The University of Leeds

SCHOOL OF COMPUTER STUDIES

Research Report Series

Balancing Space and Time Errors for Spectral Methods used with the Method of Lines for Parabolic Equations

M.Berzins

Report 91.14 **April 1991**

School of Computer Studies
University of Leeds
LEEDS LS2 9JT
U.K.

Tel. 0532–335430
JANET: editor@dcs.leeds.ac.uk

University of Leeds
SCHOOL OF COMPUTER STUDIES
RESEARCH REPORT SERIES
Report 91.14

Balancing Space and Time Errors for Spectral Methods used with the Method of Lines for Parabolic Equations

by

M Berzins
Division of Computer Science

April 1991

1. Introduction.

The recent interest in the development of general purpose codes for time-dependent partial differential equations has been documented by Machura and Sweet [15], Ortega and Voigt [17] and Hindmarsh [11]). One of the factors influencing the performance of these codes is the choice of a suitable spatial discretisation method which enable the computed solution to model accurately the exact solution to the p.d.e. The advantage of using high-order spectral spatial discretisation methods is that high accuracy can be achieved using a small number of spatial mesh points, see [3] , [4] and the recent monograph by Canuto et al [6]. Once the spatial discretisation method has been chosen, it is desirable to integrate the o.d.e. system in time with just sufficient accuracy so that the temporal error does not significantly corrupt the spatial accuracy. However, in most existing software based on the method of lines, the standard procedure is to control the local time error per step with respect to a supplied accuracy tolerance which is, in general, independent of the spatial discretisation error and also of the global time error in the computed solution.

An alternative approach is the new time error control of Lawson, Berzins and Dew [13] which has successfully been used in conjunction with low-order finite difference spatial discretisation methods to control the temporal error so that it is dominated by the spatial error. This approach controls the local time error in such a way that it is a fraction of the change in the spatial error over each time step. In this way the error tolerance used in the time integration is varied according to the size of the spatial discretisation error. The purpose of this paper is to see if this new error control strategy can be used with high-order spectral spatial discretisation methods as a means of approximately balancing the contributions of the spatial and temporal errors to the overall error in the computed solution

Such a strategy is required all the more in the case of high-order spectral methods as the high spatial accuracy achieved makes it difficult to select beforehand a local error time integration tolerance which is fine enough to allow the accuracy of the spatial discretisation to be observed yet coarse enough to allow the time integration to be efficient. The paper will show that the new strategy controls the error in the required way so that the spatial error dominates and also that realistic estimates of the pointwise globlal error may be obtained providing that a good estimate of the spatial truncation error is available. The paper superceedes earlier work of Berzins and Dew [3] in two important ways. The first way is that rather than solely estimating the combined space and time error for a given time integration tolerance the time tolerance is

chosen automatically so that the space error dominates. The second important difference is that the spatial truncation error is now estimated in a much simpler way than before by making use of the rate of decrease of the polynomial coefficients as the degree of the polynomial increases. The previous approach, [3], employed an iterative scheme to estimate the space truncation error.

An outline of the contents of the paper is as follows. Sections 2 and 3 outline the problem class to be considered and the discretisation method to be employed. Section 4 describes the general time integration procedure employed while in Section 5 the new error balancing algorithm and its implementation for high-order spectral methods is described . The key details of the estimation of the space trunction error and of the revised time integrator are also supplied in Section 5. The numerical experiments in Section 6 show the effectiveness of the new strategy in estimating and balancing the error on a simple model problem. Finally Section 7 contains a summary of the paper and highlights the areas in which further work needs to be done. It is shown that the error control strategy appears to offer an effective method of balancing the spatial and temporal errors in the method of lines but that further work is required to obtain a completely reliable and robust estimate of the spatial truncation error.

2. Problem Class and Spatial Discretisation Method .

The problem class considered here is sufficiently general to illustrate the algorithm for error estimation. The algorithm extends naturally to systems of p.d.e.s providing that the same method of lines approach is employed.

The formulas were derived by Berzins and Dew [3] using an improved form of the generalized Chebyshev method of Berzins and Dew [2]. A recent theoretical analysis of the method for steady state problems is due to Funaro [9]. Berzins and Dew [4] describe software that implements the discretisation method with a well-known time integrator.

The discretisation method is similar to the C^0 collocation methods of Diaz [8] and Leyk [12] . In all these methods the approximate solution is continuous at a set of spatial breakpoints and satisfies the differential equation at a number of collocation points between each pair of breakpoints; Diaz uses transformed Gauss or Jacobi points, while Leyk uses the zeros of a Legendre polynomial. The finite element approach of these authors is to fix the degree of the approximating polynomial and to increase the

number of spatial elements to obtain the required accuracy.

In contrast the method of Berzins and Dew is a global element method in the sense that the number of elements is normally fixed by physical constraints and it is the degree of the polynomial that can be adjusted to obtain the required accuracy. The method uses the transformed Chebyshev collocation points between each pair of breakpoints so that the method exploits the well - known approximation properties of *global* Chebyshev polynomial collocation methods , see Canuto et al [6]; thus combining the advantages of global approximation methods with the flexibility of the C^0 collocation approach in handling material interfaces and boundary conditions.

In this paper we shall make use of the discretisation method as applied to the time dependent system of NPDE partial differential equations

$$Q_k(x\ ,t\ ,\ \underline{u}\ ,\ \underline{u}_x\ ,\ \underline{u}_t) = x^{-m}\ \frac{\partial}{\partial x}(x^m R_k(x\ ,t,\ \underline{u}\ ,\ \underline{u}_x)\ )$$

$$(x,t)\ \varepsilon\ \Omega\ =\ [a,b]\times(0,t_e]\ \ ,\ a<b\ \ ,\ k=1,\ldots,NPDE. \tag{2.1}$$

The vector $\underline{u}(x\ ,t)$ is defined by

$$\underline{u}(x\ ,t) = [u_1(x,t),\ldots,u_{NPDE}(x,t)]^T\ \ , \tag{2.2}$$

the vectors $\underline{u}_x(x\ ,t)$ and $\underline{u}_t(x\ ,t)$ are similarly defined . The non-negative integer m denotes the space geometry type, when m is greater than zero, a must be greater than or equal to zero.

The function R in equation (2.1) may be thought of as a flux, e.g. $R = Ku_x$, and it is convenient to use this flux in the definition of the boundary conditions . Thus the boundary conditions are defined by

$$\beta_k(x,t)R(x\ ,t,\ \underline{u}\ ,\ \underline{u}_x) = \gamma_k(x\ ,t,\ \underline{u}\ ,\ \underline{u}_x)$$

$$\text{where}\quad k=1,\ldots,NPDE\ \text{ and }\ x=a\ \text{ or }\ x=b. \tag{2.3}$$

and the initial conditions are assumed to have the form

$$\underline{u}(x\ ,0)\ =\ \underline{k}(x)\ \ ,\ x\ \varepsilon\ [a\ ,b]\ . \tag{2.4}$$

In the case when the integer m in equation (2.1) is greater than zero we have to make special provision for the polar form of the differential operator by using the technique of Berzins and Dew [2] .

3. Outline of Chebyshev C^0 Collocation Method.

In this section a brief description of the key features of the numerical method is given . Define the spatial mesh of break points by

$$a = X_0 < X_1 < \cdots < X_{NEL} = b \, . \tag{3.1}$$

where the X_j are the breakpoints. This mesh partitions the interval $[a \, , b]$ into NEL elements ,

$$I_j = [X_{j-1} \, , X_j] \, , \text{ of length } h_j = X_j - X_{j-1} \, , \; j = 1,2,\ldots,NEL \, . \tag{3.2}$$

The breakpoints are chosen by the user to suit his application . However if the function Q in equation (2.1) has a discontinuity with respect to the spatial variable x at some point in the interval $[a \, , b]$ then this point must be one of the breakpoints.

The k th component of the solution to equations (2.1) , (2.2) and (2.3) is approximated by a continuous piecewise polynomial approximation $U_k(x \, , t)$ using a polynomial of degree N in each element. We shall consider the case $N \geq 2$ as this makes it possible to describe the method of Berzins and Dew [3] in a very straightforward manner*. The degree , N , of the polynomial is chosen by the user. The approximate solution has the form

$$U_{k,j}(x \, , t) = \sum_{i=0}^{N} a_{j,i}(t) \, T_i(W_j(x)) \, , \quad x \in I_j \quad k = 1,\ldots,NPDE. \tag{3.3}$$

where $U_{k,j}(x \, , t)$ is the restriction of $U_k(. \, , t)$ to the element I_j , $T_i(.)$ is the Chebyshev polynomial of the first kind of degree i and W_j is defined as the linear map of the interval $[X_{j-1} \, , X_j]$ onto $[-1 \, , 1]$.

For the sake of clarity we shall consider the case of one p.d.e. and drop the subscript k used in Section 2 and equation (3.3). On substituting the approximate solution (3.3) for the exact solution in equations (2.1) (2.2) and (2.3) we can use interpolation to define the piecewise polynomial approximations $Q(x,t)$ and $R(x \, , t)$. These two polynomials are of degree N in each interval and are defined by

$$\begin{aligned} Q(x_{j,i} \, , t) &= Q(x_{j,i} \, , t \, , \underline{U} \, , \underline{U}_x \, , \underline{U}_t) \\ R(x_{j,i},t) &= R(x_{j,i} \, , t, \underline{U} \, , \underline{U}_x) \\ j = 1,2,\ldots,NEL \; &; i = 0,1,2,\ldots,N \, , \end{aligned} \tag{3.4}$$

where the vectors $\underline{U}$, $\underline{U}_x$ and $\underline{U}_t$ are of length one as there is only one p.d.e. and the transformed Che-

* In the case when $N = 1$ Berzins and Dew [3] showed that the method is a lumped Galerkin finite element method with linear basis functions or a second - order finite difference method.

byshev points $\{x_{j,i}\}$ are defined by

$$W_j(x_{j,i}) = \cos\left(\frac{(N-i)\pi}{N}\right) \ , \ j = 1,2,\ldots,NEL \ , \ i = 0,1,\ldots,N \tag{3.5}$$

where $W_j(x)$ is the linear map defined in equation (3.3). Although the exact p.d.e. flux function $R(x,t)$ is assumed to be continuous at the breakpoints , the function $Q(x,t)$ is allowed to be discontinuous at these points. (In fact the *numerical* approximation to the p.d.e. flux may also happen to be discontinuous at the breakpoints, see equation (3.13) below.)

3.1. Collocation Equations.

The transformed Chebyshev points (excluding the breakpoints) are the collocation points used by Berzins and Dew [3]. In other words the computed solution and its space and time derivatives are calculated so as to satisfy the collocation-like equations:

$$Q(x_{j,i},t) - \frac{\partial R}{\partial x}(x_{j,i},t) = 0$$

$$j = 1,2,\ldots,NEL \ ; i = 1,2,\ldots,N-1 \ , \tag{3.6}$$

where the points $x_{j,i}$ are defined by equation (3.5).

In the case of just one p.d.e. and if

$$Q(....) = \frac{\partial u}{\partial t} - f(x,t,u,\frac{\partial u}{\partial x}) \tag{3.7}$$

equation (3.6) defines $\frac{\partial U}{\partial t}$ *explicitly* at the collocation point $x_{j,i}$

$$\frac{dU_{j,i}}{dt} = \frac{\partial R}{\partial x}(x_{j,i},t) + \frac{m}{x_{j,i}}R(x_{j,i},t) + f(x_{j,i},t,U_{j,i},\frac{dU_{j,i}}{dx})$$

$$\text{where} \quad U_{j,i} = U(x_{j,i},t) \ j = 1,2,\ldots,NEL \ ; \ i = 1,2,\ldots,N-1 \ .$$

3.2. Boundary and Breakpoint Conditions.

The polynomial $U(x,t)$ is continuous at each breakpoint and is required to satisfy the finite-element type orthogonality condition:

$$\int_a^b \left(Q(\cdots) - \frac{dR}{dx}(\cdots)\right)\bar{p}_j(x)\,dx = 0 \tag{3.8}$$

where $\bar{p}_j(x)$ is the linear basis (hat) functions defined by

$\bar{p}_j(X_i) = 1 \quad , i = j \ ,$

$\bar{p}_j(X_i) = 0 \quad$ otherwise and $j = 0,1, \cdots, NEL.$

Equation (3.8) is integrated by parts to get

$$\int_a^b Q(\cdots)\bar{p}_j(x)\,dx + R(\cdots)\frac{d\bar{p}_j}{dx}\,dx = [R(\cdots)\,\bar{p}_j]_a^b. \tag{3.9}$$

where the term on the right side of the equality may be seen to be zero from equation (3.9) unless $j = 0$ or $j = NEL$ in which case the boundary conditions (2.3) are used to substitute for the values of the flux $R(\cdots)$ at the boundaries. At the left-hand boundary

$$\beta(a,t) \int_a^{x_1} R\,\frac{d\bar{p}_0}{dx} + Q\,\bar{p}_0\,dx = -\gamma(a\ ,t,U\ ,\frac{\partial U}{\partial x}) \tag{3.10}$$

and at the right-hand boundary

$$\beta(b,t) \int_{x_{NEL-1}}^{b} R\,\frac{d\bar{p}_{NEL}}{dx} + Q\,\bar{p}_{NEL}\,dx = \gamma(b\ ,t,U\ ,\frac{\partial U}{\partial x}). \tag{3.11}$$

Berzins and Dew [3] showed that the integrals in equation (3.9) can be approximated by the $N+1$ point Clenshaw-Curtis quadrature rule with weights denoted by $\{\lambda_i\}_{i=0}^{N}$ to derive the equations at the boundaries. In the case when $N>1$ these equations can be further simplified by using the collocation equations (3.6) to get collocation-like equations at the boundaries.

$$\begin{aligned}
\beta(a,t)\ Q(a\ ,t) &= \beta(a,t)\,\frac{\partial R}{\partial x}(a,t) \\
&+ [\beta(a,t)R(a,t) - \gamma(a\ ,t,U\ ,\frac{\partial U}{\partial x})]\,\frac{2}{\lambda_N\,h_1} \\
\beta(b,t)\ Q(b\ ,t) &= \beta(b,t)\,\frac{\partial R}{\partial x}(b,t) \\
&+ [-\beta(b,t)R(b,t) + \gamma(b\ ,t,U\ ,\frac{\partial U}{\partial x})]\,\frac{2}{\lambda_N\,h_{NEL}}
\end{aligned} \tag{3.12}$$

In the case when $Q(...)$ has the form given by equation (3.7) it is can be seen that equations (3.12) define the time derivatives at the boundaries, providing that the functions $\beta(a,t)$ and $\beta(b,t)$ are non-zero.

In a similar way Berzins and Dew [3] showed that quadrature rules and the collocation equations (3.6) can be used to rewrite the interior breakpoint condition (3.9) as*

* Although the p.d.e. flux is assumed to be continuous the simplification in Berzins and Dew [3] incorrectly assumed that the numerical flux R (x ,t) was also continuous at the breakpoints and so neglected the bracketted term [...] in equation (3.13). As their code did not implement the simplification this did not effect their numerical results.

$$h_j\, Q(x_{j,N}\,,\,t) \;+\; h_{j+1}\, Q\,(x_{j+1,0}\,,\,t) \;=$$

$$h_j\,\frac{\partial R}{\partial x}(x_{j,N}\,,\,t) \;+\; h_{j+1}\,\frac{\partial R}{\partial x}(x_{j+1,0}\,,\,t) \;+\; [R(x_{j+1,0},t) - R(x_{j,N}\,,\,t)]\,\frac{2}{\lambda_N} \qquad (3.13)$$

Although the transformed Chebyshev points $x_{j,N}$ and $x_{j+1,0}$ are both equal to the breakpoint X_j for $j = 1,2,\ldots,NEL-1$ we denote by $Q(x_{j,n}\,,\,t)$ the value of Q evaluated at X_j using the polynomial in I_j and by $Q(x_{j+1,0}\,,\,t)$ the value of Q evaluated at X_j using the polynomial in I_{j+1} . This takes into account possible discontinuities in the functions $Q(\cdots)$ at the breakpoints $\{X_j\}$ by using the values of $Q(x\,,\,t)$ as x tends to the breakpoint from above and below. In practice it is straightforward to specify any such discontinuities , see [4] .

4. Integration Using the Method of Lines .

The success of the method of lines in solving coupled systems of ordinary and partial differential equations lies in combining efficient and general spatial discretisation methods with sophisticated o.d.e. initial value problem integrators being used to perform the time integration . The essence of the method is to spatially discretise a system of *NPDE* time - dependent partial differential equations with a spatial mesh of *NPTS* points into a system of *NPTS*NPDE* coupled ordinary differential equations of the form of .

$$\underline{F}(T\,,\,\underline{U}\,,\,\dot{\underline{U}}) = 0\,,\;\underline{U}(0) = \underline{k}\,, \qquad (4.1)$$

Each solution component of these equations defines one component of the p.d.e. solution at a single mesh point.

4.1. Ordering of the O.D.E. Solution Vector.

The following convention may be used in ordering the o.d.e. solution vector $\underline{Y}(t)$ of equation (4.1) . We assume that the system of *NPDE* p.d.e.s is discretised using *IBK* spatial breakpoints , a polynomial degree of *NPOLY*. (In the description that follows *NPOLY* rather than *N* will be used to denote the polynomial degree as this has a less ambiguous meaning .) The p.d.e. solution components are stored in the *NPDE*×*NPTS* components of the vector $\underline{U}(t)$ and *NPTS* will be defined below. The value of *NPTS* is (*IBK*−1)*(*NPOLY*) + 1 where *NPOLY* is the degree of the approximating polynomial used between each pair of spatial mesh points and where *IBK* is the number of breakpoints .

Using the above ordering the system of ordinary differential equations in time defined by the Chebyshev C^0 collocation method (e.g. equations (3.6), (3.12) and (3.13)) can equivalently be written as

equation (4.1) where the vector $\underline{U}(t)$ is defined by

$$\underline{U}(t) = \begin{bmatrix} \underline{U}_1 \\ \underline{U}_2 \\ \cdot \\ \cdot \\ \underline{U}_{NEL} \end{bmatrix} \quad , \quad \underline{U}_j = \begin{bmatrix} \underline{U}_{j,0} \\ \underline{U}_{j,1} \\ \cdot \\ \cdot \\ \underline{U}_{j,N-1} \end{bmatrix} \quad , \quad j = 1,2,\ldots,NEL-1 \quad , \quad \underline{U}_{NEL} = \begin{bmatrix} \underline{U}_{NEL,0} \\ \underline{U}_{NEL,1} \\ \cdot \\ \cdot \\ \underline{U}_{NEL,N} \end{bmatrix} \tag{4.2}$$

$$N = NPOLY, \text{ and } \underline{U}_{j,i} = \underline{U}(x_{j,i},t) \text{ where}$$

$$x_{j,i} = \tfrac{1}{2}(X_{j+1} + X_j + (X_{j+1} + X_j)\cos(\frac{N-i}{N}\pi)\ ) \tag{4.3}$$

and the vector $\underline{U}(x\ ,\ t)$ is defined as in equation (2.1). The total number of ordinary differential/algebraic equations for a system of NPDE p.d.e.s , IBK breakpoints and using a piecewise polynomial of degree NPOLY is given by NEQ where

$$NEQ \ = \ NPDE * (IBK-1) * (NPOLY) + NPDE \tag{4.4}$$

The initial condition for $\underline{U}(t)$ is found by evaluating the function $k(x)$ at the transformed Chebyshev points in each element . The value of $\underline{U}(0)$ is thus defined by substituting

$$U_{j,i} \ = \ k(x_{j,i}) \ j = 1\ ,\ \cdots\ ,NEL\ ,i = 0,\ldots\ ,N \tag{4.5}$$

into equations (4.2) .

4.2. Error Control for Stiff O.D.E.s.

In most of the codes available for solving time dependent o.d.e.s, the routines attempt to control the local time integration error in the computed solution with regards to an accuracy tolerance supplied by the user, TOL. The i.v.p. to be solved is given by equation (4.1) with the true solution $\{\underline{U}(t_n)\}_{n=0}^{p}$ approximated by $\{\underline{V}(t_n)\}_{n=0}^{p}$ at a set of discrete times $0 = t_0 < t_1 < \cdots < t_p = t_e$ by a time integration method with requested absolute local error accuracy, TOL. Using interpolation a continuous piecewise differentiable approximate solution can be constructed so that

$$\underline{z}(t,\text{TOL}) = \underline{z}_{n+1}(t,\text{TOL}) \quad , \quad t \in [t_n\ ,\ t_{n+1}] \quad , \quad n = 0,\ \cdots\ ,p-1,$$

which fits through the sequence of points $\{(\underline{V}(t_n)\ ,\ t_n)\}_{n=0}^{p}$. The following definitions can now be made. The global error in the o.d.e. solution at time t_{n+1} and for a tolerance TOL, is given by

$$\underline{ge}_{n+1}(\text{TOL}) = \underline{U}(t_{n+1}) - \underline{V}(t_{n+1}). \tag{4.6}$$

The global error associated with the continuous solution $\underline{z}(t,\text{TOL})$ is

$$\underline{ge}(t, \mathrm{TOL}) = \underline{U}(t) - \underline{z}(t, \mathrm{TOL}) \quad , \quad t \in (0 , t_e]. \tag{4.7}$$

The local solution on $[t_n , t_{n+1}]$, $\underline{y}_{n+1}(t , \mathrm{TOL})$, is the solution of the i.v.p.

$$\underline{F}(\underline{y}_{n+1}(t, \mathrm{TOL}) , \underline{y}_{n+1}'(t, \mathrm{TOL}) , t) \quad , \quad \underline{y}_{n+1}(t_n , \mathrm{TOL}) = \underline{V}(t_n). \tag{4.8}$$

The local error per step (LEPS) at t_{n+1} is given by

$$\underline{le}_{n+1}(TOL) = \underline{V}(t_{n+1}) - \underline{y}_{n+1}(t_{n+1} , \mathrm{TOL}). \tag{4.9}$$

while the local error per unit step (LEPUS) is given by

$$\frac{\underline{le}_{n+1}(TOL)}{k_n} , \tag{4.10}$$

where $k_n = t_{n+1} - t_n$.

Modern stiff o.d.e. codes such as those based on the backward differentiation formulae e.g. see Hindmarsh [11], control the local error per step, LEPS, by varying the time stepsize and also by using formulae of different orders. However the use of local error control makes it very difficult to establish a relationship between the accuracy tolerance, TOL, and the o.d.e. global error, Shampine [18]. In general, the time global error is not even proportional to the local error tolerance, TOL.

In order to balance the spatial and temporal errors, it is desirable that the error control strategy should yield a solution with a time integration error that is directly proportional (in practice less than) the error incurred by spatial discretisation. This is desirable because once we have chosen a spatial discretisation method with a particular degree of polynomial we do not wish to introduce a further larger error by time integration when this is not necessary. Niether do we want to integrate in time with a much greater accuracy than is required. Suppose that the error incurred by spatial discretisation is approximately equal to TOL. We would then like to have a time integration procedure that satisfies tolerance proportionality, that is, there exists a linear relationship between the time global error and the requested accuracy, TOL . An error control strategy is said to satisfy tolerance proportionality, Stetter [19], if the numerical solution is such that, if $\underline{ge}(t, \mathrm{TOL})$ is the global error at time t for an accuracy requirement TOL, then, for $r > 0$,

$$\underline{ge}(t , r\mathrm{TOL}) \simeq r \, \underline{ge}(t, \mathrm{TOL}). \tag{4.11}$$

The work of Stetter [19] shows that in order to obtain tolerance proportionality we must control the LEPUS rather than the LEPS. However, for stiff i.v.p.s LEPUS control may be inefficient, Lindberg [14] although the situation in the method of lines is fundamentally different in that any error control strategy

must take account of the already present spatial error. However, it is difficult to select a local error per step tolerance that will ensure that this is so, since the o.d.e. global error need not be related to the chosen accuracy tolerance. In addition, the spatial accuracy may vary with time, so any fixed tolerance used in the o.d.e. integrator is unlikely to be related to the size of the changing spatial error. Thus a local error per unit step control is needed which is related to the spatial discretisation error in some way and which can be modified accordingly as the spatial error varies.

5. The Error Balancing Approach of Lawson, Berzins and Dew [13].

In order to develop an efficient integration strategy that allows the spatial error to dominate we shall make use of the error balancing approach of Lawson, Berzins and Dew which in turn makes use of the global error indicator of Berzins, [3]. This approach is based on a form of LEPUS step control in which the o.d.e. tolerance is modified as the spatial discretisation error varies in time.

The vector of the values of the overall error at the spatial mesh points, at any time t, is defined by $\underline{E}(t)$ where

$$\underline{E}(t) = \underline{u}(t) - \underline{V}(t), \tag{5.1}$$

where $\underline{u}(t)$ is the restriction of the exact p.d.e. solution to the mesh δ i.e.

$$[\underline{u}(t)]_i = u(x_i, t) \quad i = 1, \ldots, N.$$

The vector $\underline{E}(t)$ may also be written as the sum of the restriction of the p.d.e. spatial discretisation error $\underline{es}(t)$, as defined by

$$\underline{es}(t) = \underline{u}(t) - \underline{U}(t), \tag{5.2}$$

and the o.d.e. global error $\underline{ge}(t, \mathrm{TOL})$, see equation (2.11), which represents the accumulation of the spatial discretisation error at the mesh points, The equation used by Berzins, [3], for the evolution of this spatial discretisation error is

$$A\,\dot{\underline{es}}(t) = -J\,\underline{es}(t) + \underline{TE}(t, \underline{u}(t), \dot{\underline{u}}(t)), \quad \underline{es}(0) = \underline{0}, \tag{5.3}$$

where the space truncation error vector $\underline{TE}(...)$ is defined by

$$\underline{TE}(t, \underline{u}(t), \dot{\underline{u}}(t)) = \underline{F_N}(t, \underline{u}(t), \dot{\underline{u}}(t))$$

and the matrices A and J are defined by

$$A = \frac{\partial F_N}{\partial \underline{\dot{U}}} \quad \text{and} \quad J = \frac{\partial F_N}{\partial \underline{U}} .$$

In order to compute the term $\underline{TE}$ the exact solution to the p.d.e. on the mesh δ at the time points t_n , t_{n+1} must be known. Since, in general, the exact solution is not known suitable estimates for the space truncation error must be derived.

Berzins [3] suggested that in the case of the backward differentiation formulae efficient global error estimating procedure is defined by

$$\underline{E}(t_{n+1}) = M^{-1} \left(A \, \underline{E}(t_n) + k_n \, \underline{TE}_{n+1} \right) + \underline{le}_{n+1}(\mathrm{TOL}) , \tag{5.4}$$

where the local error estimate is given by equation (3.4) of Berzins [3] and where

$$\underline{TE}_{n+1} = \underline{TE}(t_{n+1} , \underline{u}(t_{n+1}) , \underline{\dot{u}}(t_{n+1}))$$

and where M^{-1} represents the solution of a system of linear equations using the LU decomposition of the matrix $M = A - k_n \gamma J$, that is stored by the o.d.e. integrator, see [3]. Although this, in general, provides only a zero-order approximation to the true overall error we have found that this gives a good estimate of the overall error [1], [13] , providing that the estimates of the space truncation error and the local time error are reliable.

5.1. A New LEPUS Control Strategy.

The error control used by Lawson, Berzins and Dew [13] controls the local error with respect to the contribution of the spatial truncation error and the existing error from previous steps to the global error at the end of the next time step. Theorem 2 of [13] shows that by controlling the vector norm of the time local error so that it is a fraction, ε , of the vector norm of the growth in the space error over the interval $k_n = t_{n+1} - t_n$, the temporal integration error will be dominated by the spatial discretisation error. This error control thus translates as the time local error control strategy given by

$$|| \underline{le}_{n+1}(\mathrm{TOL}) || < \varepsilon \, || \underline{E}(t_{n+1}) - \underline{E}(t_n) - \underline{le}_{n+1}(\mathrm{TOL}) || \tag{5.5}$$

where $|| \; ||$ is some suitable weighted vector norm .

In practice, the control strategy (5.5) is not applied directly since the term $\underline{E}(t_{n+1})$ is known only through the global error estimator using equation (5.4). Instead equation (5.4) is used to substitute for $\underline{E}(t_{n+1})$, thus giving

$$||\underline{le}_{n+1}(\mathrm{TOL})|| < \varepsilon\, ||M^{-1}(A\,\underline{E}(t_n) + k_n\underline{TE}_{n+1}) - \underline{E}(t_n)||$$

which, using the definition of the matrix $M = A - k_n \gamma J$, in equation (5.4), gives

$$\begin{aligned} ||\underline{le}_{n+1}(\mathrm{TOL})|| &< \varepsilon\, ||M^{-1}(k_n \gamma J\underline{E}(t_n) + k_n\underline{TE}_{n+1})|| \\ &= k_n\, \varepsilon\, ||M^{-1}(\gamma J\underline{E}(t_n) + \underline{TE}_{n+1})|| \end{aligned} \tag{5.6}$$

which shows that the strategy is of the LEPUS form. In practice (5.6) is modified to cater for a zero spatial error or for a very large spatial error by enforcing upper and lower bounds of the form

$$\mathrm{TOL}_{\min} < ||M^{-1}(\gamma J\underline{E}(t_n) + \underline{TE}_{n+1})|| < \mathrm{TOL}_{\max}\ ,$$

where $\mathrm{TOL}_{\min}$ is a small number representing the minimum local error per unit tolerance allowed and $\mathrm{TOL}_{\max}$ is $O(1)$ representing the maximum local error per unit step tolerance allowed.

5.2. Derivation and Estimation of P.D.E. Truncation Error.

In the estimation of the combined error due to the spatial and temporal approximations it is necessary to estimate the error at the individual mesh points. A simple point-wise error estimate that satisfies equation (4.6) may be derived by assuming that the error in any interval may be approximated by a the next term in the polynomial expansion. Suppose that the exact solution of the p.d.e. defined by equations (2.1) , (2.3) and (2.4) on the interval I_j is given by the uniformly convergent Chebyshev series

$$u_j(x\ ,t) = \sum_{i=0}^{\infty} b_{j,i}(t)\, T_i(\,W_j(x)\,) \quad ,\quad x \in I_j. \tag{5.7}$$

and that the truncated series solution has the form given by

$$\hat{u}_j(x\ ,t) = \sum_{i=0}^{N} b_{j,i}(t)\, T_i(\,W_j(x)\,) \quad ,\quad x \in I_j. \tag{5.8}$$

A standard result e.g. see Oden et al [17] is that

$$||\ u(.\ ,t) - \hat{u}(.\ ,t)\ || \le C\, h^{\mu\,-\,1}\, N^{-p+1}\ \ ||\ u\ ||_r$$

where $||\ \ ||_r$ is a Sobolev norm and where $\mu = \min(N-1\ ,\ r)$, N is the degree of polynomial and h is the mesh spacing of the elements on which the polynomial of degree p is defined. The computed solution has the form given by

$$U_j(x\ ,t) = \sum_{i=0}^{N} a_{j,i}(t)\, T_i(\,W_j(x)\,) \quad ,\quad x \in I_j. \tag{5.9}$$

The spatial truncation error is estimated using by using three assumptions. The first of these is that the polynomial coefficients , $b_{j,i}$ converge at some power rate

$$| b_{j,i}(t) | \simeq B_j(t)\, i^{-r} \quad , \quad r > \tfrac{1}{2} \quad . \tag{5.10}$$

The second assumption is that the spatial error is approximately equal to the next term in the polynomial expansion multiplied by some appropriate constant. In other words that

$$u_j(x\,,t) \simeq U_j(x\,,t) \; + \; N^{-p}\, b_{j,N+1}(t)\, T_{N+1}(\, W_j(x)\,) \quad , \quad x \in I_j. \tag{5.11}$$

The third assumption is that the error is largest at the interior points and can be neglected at the break-points and boundaries for the purpose of estimating the spatial truncation error.

$$u_j(X_j\,,t) \simeq U_j(X_j\,,t) \tag{5.12}$$

where $j = 0, \ldots, NEL$.

The three above assumptions will now be used to devise an algorithm for estimating the truncation error. From assumption (5.10) it follows that

$$\log\left\{\frac{|b_{j,N-1}|}{|b_{j,N}|}\right\} = r \log\left\{\frac{N}{N-1}\right\} \tag{5.13}$$

As the coefficients $b_{j,N-1}$ and $b_{j,N}$ are unknown the coefficients $a_{j,N-1}$ and $a_{j,N}$ are used instead in this equation. For the values of N of interest the value of r is given by the approximation

$$r \; = \; (N-1) \log\left\{\frac{|a_{j,N-1}|}{|a_{j,N}|}\right\} \tag{5.14}$$

Although it is to be expected that $|r|$ may be greater than one half, in practice it is necessary to deal with smaller values of $|r|$. In this case the small value of $|r|$ indicates that the coefficients are only decreasing slowly as N increases and $a_{j,N+1}$ may not differ greatly from $a_{j,N}$. Similarly when the computed value of $|r|$ is greater than one the rate of convergence estimate may be over-optimistic and so a value of one may be used. The algorithm used for estimating the coefficient $b_{j,N+1}$ is then

$$N^{-p}\, b_{j,N+1} \; = \; a_{j,N}\, (N+1)^{-q}\, N^{p-q}$$

assuming that $p = q$ gives

$$N^{-p}\, b_{j,N+1} \; = \; a_{j,N}\, (N+1)^{-q} \tag{5.15}$$

where $q = 0$ if $|r| < 1/2$, $q = 1/2$ if $|r| > 1/2$ and $q = 1$ if $|r| > 1$. This provides a quite general approach for estimating the error term in the numerical solution when calculating the truncation error. The same algorithm is also used to estimate the truncation error in the solution time derivative polynomial.

The above algorithm must be modified however if either of the coefficients $a_{j,N}$ or $a_{j,N-1}$ (or the corresponding coefficients of the time derivative) are zero. In the case when $a_{j,N}$ is zero the coefficient is

estimated

by assuming that the true value of this coefficient in the exact solution, $b_{j,N}$, obeys the relationship

$$\frac{|b_{j,N}|}{|b_{j,N-1}|} \simeq \frac{|b_{j,N-1}|}{|b_{j,N-2}|}$$

and consequently, replacing unknown coefficients with known values, that

$$b_{j,N+1} = a_{j,N-1} \frac{|a_{j,N-1}|}{|a_{j,N-2}|} N^{-1} \tag{5.16}$$

Finally it is possible that $a_{j,N-2}$ may also be zero in equation (5.16) in which case it is replaced by $b_{j,N-3}$.

In order to estimate the space truncation error we approximate the exact solution by

$$v_j(x\,,t) = U_j(x\,,t) + b_{j,N+1}(t)\,T_{N+1}(W_j(x)) \quad , \quad x \in I_j.$$

and the exact time derivative by

$$\frac{\partial v_j}{\partial t}(x\,,t) = \frac{\partial U_j}{\partial t}(x\,,t) + d_{j,N+1}(t)\,T_{N+1}(W_j(x)) \quad , \quad x \in I_j.$$

where $d_{j,N+1}(t)$ is the error coefficient for the time derivative amd estimate the truncation error by substituting these polynomials in the routine used to define the residual of the o.d.e. system i.e.

$$\underline{TE}(t\,,\underline{u}(t)\,,\dot{\underline{u}}(t)) \simeq \underline{F}_N(t\,,\underline{v}(t)\,,\dot{\underline{v}}(t))$$

5.3. Modifying the Time Integration Procedure.

As most ode integrators for stiff o.d.e.s use local error per step control it is necessary to describe how the time integrator was modifed to inplement the error control described above. The time integration package used was the SPRINT software [5] with the SPGEAR time integration module. As this code was developed from the LSOD* family of codes due to Hindmarsh [10] similar comments apply to the modification of those codes also. The modifications were both few in number and simple to implement. The SPRINT codes were modified in only three areas ; the selection of the initial stepsize, the stepsize/order selection strategy in the integrator and the vector norm used throughout the code.

Almost all the important decisions made by the o.d.e. integrator are based on vector norms. In LSODE and related codes the usual vector norm is weighted by dividing each component of the vector by an individual error weight. In the unweighted norm used as part of the new strategy the error weight is always one. This vector norm is denoted by $||.||_0$. The weighted vector norm used in the new LEPUS

strategy for the timestep from t_n to t_{n+1} is given by

$$||\underline{x}(t)||_w = \frac{||\underline{x}(t)||_0}{||\underline{\Delta E}(t_{n+1})||_0} \tag{5.17}$$

where from equation (5.6)

$$\underline{\Delta E}(t_{n+1}) = \varepsilon\ (k_n\ (M^{-1}(\ \gamma J\ \underline{E}(t_n) + \underline{TE}_{n+1})).$$

and the new local error test used in the code is

$$||\underline{x}(t)||_w < 1 \tag{5.18}$$

In order to implement this method in the code used for the experiments it was assumed that $\underline{\Delta E}$ does not change by an order of magnitude from one timestep to the next. In other words the value of $\underline{\Delta E}$ calculated at t_n is used to control the error at t_{n+1} . In contrast a production code implementing the LEPUS error control would calculate $\underline{\Delta E}$ after the non-linear equations have been solved for the new solution at time t_{n+1} (and prior to the local error test at the end of the step) and use it in the time error control at t_{n+1} .

In the case of the stepsize and order selection strategy the power of the timestep present in any error estimate is reduced by one when the new weighted norm is used. In particular the factor F_l by which the stepsize could change if order l is used is now given by

$$F_l = (\ c_l\ ||\underline{E}_l||_w^p\ +\ d_l)$$

where c_l and d_l are constants , $\underline{E}_l$ is an estimate of the error at order l and $p = 1/(l+1)$ if local error per step is used and $p = 1/l$ for the new LEPUS startegy.

In the case of the algorithm to estimate the initial stepsize the algorithm must be adjusted to take account of the initial error on the first step being only order one when a first-order method is used rather than order two as is normally the case.

6. Numerical Experiments.

The first example is a p.d.e. in spherical polar co-ordinates used by Berzins and Dew [3] and is defined as Problem 1 in the Appendix.

The Chebyshev C^0 collocation method was applied to this problem using a single polynomial expansion of degree 5 , 7 , and 9 was used to represent the solution. The first table shows that o.d.e. integration with the new strategy was sufficiently accurate in each case to ensure that the spatial discretisation error

dominated. This was done by comparing the global errors obtained with the new local error per unit step control eps = 0.1 , 0.2 and 0.3 with runs using a variety of local error tolerances, abs = 1.E-8 etc with the standard local error per step control. Estimates of the polar - weighted L^2 error norm ,formed by using the trapezoidal rule with 100 equally-spaced spatial mesh points are given in Table (1).

Key to Tables 1 to 4.

NEL is the number of polynomial elements used in the spatial mesh.

N is the degree of the global polynomial used to spatially discretise the p.d.e.

ε and TOL are the parameters used in the o.d.e. integration routine.

FCN is the number of ODE function calls used by the integrator.

CPU is the amount of CPU time used, measured in seconds on the IBM 3081 at RPI.

NSTEPS is the number of time steps used in the integration of the o.d.e.'s.

N	TIME	0.01	0.25	0.50	0.75	1.00	NSTEPS	FCN	CPU
5	eps = 0.3	8.5E-5	2.9E-4	2.6E-4	2.2E-4	1.7E-4	36	213	0.56
	eps = 0.2	8.5E-5	2.8E-4	2.6E-4	2.2E-4	1.7E-4	49	316	0.73
	eps = 0.1	8.7E-5	2.9E-4	2.7E-4	2.2E-4	1.7E-4	64	348	0.81
	tol = 1.E-8	8.5E-5	2.8E-4	2.7E-4	2.2E-4	1.7E-4	185	745	1.68
	tol = 1.E-6	8.5E-5	2.8E-4	2.7E-4	2.2E-4	1.7E-4	109	510	1.18
	tol = 1.E-5	8.7E-5	3.0E-4	2.7E-4	2.2E-4	1.7E-4	78	397	0.93
7	eps = 0.3	2.8E-5	2.6E-6	2.0E-6	2.0E-6	1.2E-6	74	396	1.12
	eps = 0.2	2.6E-6	2.6E-6	2.0E-6	2.0E-6	1.2E-6	83	456	1.24
	eps = 0.1	2.5E-6	2.6E-6	2.0E-6	2.0E-6	1.2E-6	95	531	1.40
	tol = 1.E-9	3.7E-6	2.6E-6	2.0E-6	2.0E-6	1.2E-6	188	888	2.27
	tol = 1.E-7	3.0E-6	2.6E-6	2.0E-6	2.0E-6	1.2E-6	92	463	1.29
	tol = 1.E-6	2.6E-6	2.6E-6	2.0E-6	2.0E-6	1.2E-6	92	429	1.31
9	eps = 0.3	7.0E-8	6.2E-8	4.5E-8	3.4E-8	2.7E-8	102	545	1.79
	eps = 0.2	6.6E-8	5.9E-8	4.4E-8	3.4E-8	2.7E-8	132	707	2.24
	eps = 0.1	6.3E-8	5.6E-8	4.4E-8	3.4E-8	2.7E-8	136	694	2.25
	tol = 1.E-9	6.2E-8	6.2E-8	4.3E-8	3.4E-8	2.7E-8	175	924	2.76
	tol = 1.E-7	6.8E-8	5.9E-8	4.3E-8	3.4E-8	2.7E-8	117	682	2.07
	tol = 1.E-6	6.8E-8	5.6E-8	4.3E-8	3.4E-8	2.7E-8	88	522	1.62

Table (1) - Estimates of L^2 Error Norm Using the Different Time Integrators.

This example shows that the new time error control strategy does a good job of controlling the time error so that the space error dominates. The CPU times and the number of o.d.e. time steps required allow a comparison to be made of the two time error control methods. The results show that the LEPUS strategy is more or as efficient as the LEPS strategy. The choice of the parameter ε , used in the control strategy is

dictated

by the conflicting requirements of making the space error dominate the time error while taking as few time steps as possible. These results re-inforce those of [13] and indicate that we should set $\varepsilon \approx 0.3$, since this offers a good compromise between a small number of timesteps and the space error being relatively unpolluted by time integration error. For this reason in the following experiments the value of $\varepsilon = 0.3$ is used.

Summarising the results, the LEPUS strategy yields, *automatically*, solutions at least as accurate as those obtained when controlling the LEPS with tolerances chosen in order that the spatial discretisation error dominates. The user no longer has to experiment with different accuracy tolerances to find the solution for which the spatial error is dominant.

The global error estimate defined by equation (5.4) can also be compared with the true error. Tables (2) and (3) below provide this comparison using both a Chebyshev error norm and the maximum error at the mesh points.

N	TIME	0.01	0.25	0.50	0.75	1.00
5	Estimate	3.3E-4	8.1E-4	7.3E-4	6.0E-4	4.7E-4
	Exact	7.1E-4	1.5E-3	1.3E-3	1.2E-3	9.4E-4
7	Estimate	9.5E-6	7.0E-6	5.4E-6	4.0E-6	3.1E-6
	Exact	1.0E-5	7.0E-6	4.9E-6	3.1E-6	2.1E-6
9	Estimate	1.9E-7	1.5E-7	1.2E-7	9.0E-8	7.0E-8
	Exact	1.4E-7	8.3E-7	4.5E-7	2.7E-8	2.1E-8

Table (2) - Estimated and True Chebyshev Error Norms for Problem 1.

N	TIME	0.01	0.25	0.50	0.75	1.00
5	Estimate	7.0E-5	1.5E-4	1.3E-3	1.1E-4	9.3E-5
	Exact	1.7E-4	3.7E-4	7.3E-4	2.8E-4	4.7E-4
7	Estimate	1.5E-6	1.4E-5	1.1E-6	8.7E-7	6.8E-7
	Exact	1.4E-6	1.5E-5	1.1E-6	7.8E-7	5.4E-7
9	Estimate	4.7E-8	3.4E-8	2.6E-8	2.0E-8	1.6E-8
	Exact	3.2E-8	1.6E-8	9.7E-9	7.0E-9	5.2E-9

Table (3) - Estimated and True Maximum Grid Errors for Problem 1.

Tables (2) and (3) show that the error estimate is acceptably close to the true error for this problem. In order to compare the new method with that of Berzins and Dew [3] the performance of the global error estimate is measured by defining the error index :-

$$E_I(t) = \frac{||Estimated\ grid\ errors\ at\ time\ t\ ||_\infty}{||Actual\ grid\ errors\ at\ time\ t\ ||_\infty}.$$

Prob	NEL	N	TIME=	0.01	0.11	0.33	0.55	0.77	1.00
1	2	7	Old	0.10	0.19	0.36	0.18	0.27	0.23
	2	7	New	1.15	1.41	1.54	0.87	0.69	0.61
7	2	10	Old	0.50	104.	8.84	3.20	2.89	1.48
	2	10	New	2.35	2.89	2.87	2.03	1.66	1.54

Table (4) - Error Index Comparison with Berzins and Dew [3] .

The problem numbers in Table (4) refer to the problems in the Appendix. Problem 1 is also Problem 1 in Berzins and Dew [3] while Problem 7 is Problem 2 in the same paper. Table (4) shows that the error estimates are an improvement over those in Berzins and Dew [3]. In order to assess the performance of the error indicator for other test problems seven other test problems were used to illustrate the performance of the global error indicator. These test problems used are described in the Appendix. Problems 6,7 and 8 require more than one polynomial to be used in space due to travelling wave type solutions and a material interface. In these cases two polynomials were used with a breakpoint halfway across the spatial range.

N	Prob 1	Prob 2	Prob 3	Prob 4	Prob 5	N	Prob 6	Prob 7	Prob 8
5	1.70	0.16	0.82	9.5	0.75	3	0.39	1.4	0.25
7	0.86	0.33	0.17	0.95	0.59	5	0.40	2.5	0.53
9	0.56	0.43	0.97	1.2	0.77	7	0.55	2.4	0.48
11	0.11	0.41	4.6	1.2	0.96	9	0.59	2.2	0.65
13	0.30	0.44	0.54	1.0	1.30	11	0.80	3.5	0.88

Table (5) - Average Error Index for Problems 1 to 8.

The error index was calculated at the end of every time step in the integration. The average error index is the sum of the error indices divided by their number. The results show that in the majority of cases the error indicator does an adequate job of estimating the global error. There are however some exceptions to this. These exceptions appear to be caused by the poor spatial truncation error estimate rather than by the time integration method. This has been verified by using the true spatial truncation error in place of the estimates described in Section 5.

N	Prob 1	Prob 2	Prob 3	Prob 4	Prob 5	N	Prob 6	Prob 7	Prob 8
5	1.10	*	1.10	1.7	1.30	3	1.20	1.4	1.20
7	1.40	*	1.20	1.2	1.30	5	1.30	1.9	1.30
9	1.70	*	1.20	1.8	1.30	7	1.30	2.1	1.30
11	1.80	*	1.40	1.8	1.50	9	1.30	2.5	1.40
13	1.80	*	1.40	1.6	1.40	11	1.30	1.9	1.70

Table (6) - Average Error Index for Problems 1 to 8 with Exact T.E.

Table 5 shows that it is the time integration of the error which can cause the error estimates to be larger than the actual error. A interesting case is that of the artificially constructed Problem 2 in which the error estimate grows to an index of about 10^5 . In this case the stepsize taken by the main integration is too large for the error equation integration to remain stable. The precise nature of the difficulty in this case is the large truncation error close to the boundaries due to the combination of non-linear source terms and derivative boundary conditions. This large error estimate growth is avoided by the approximate indicator because of assumption (5.12) that the error from the boundaries does not dominate the error estimate. The provision of a reliable truncation error estimate for all types of boundary conditions and for all degrees of polynomials remains a difficult challenge.

7. Conclusions and Further Developments.

The aim of this work is to develop a fully automatic general purpose algorithm for the numerical solution of parabolic equations using the method of lines and spectral methods. From our practical experience, the local error control strategy introduced in Section 5, equation (5.5), appears to provide a promising starting point for the development of such an algorithm. By computing the LEPUS accuracy tolerance at each time step, not only have we enabled the error in the time integration to vary in relation to the spatial discretisation error, we have ensured that the method of lines is being used efficiently. This is in contrast to standard local error control where the tolerance is supplied by the user and experimentation is needed to balance the spatial and temporal errors. A source of difficulty with the approach proposed here is the need to have a robust and reliable estimate of the spatial truncation error for a wide range of problems and different polynomial approximations.

Acknowledgements.

The author would like to acknowledge the financial support of the Rensselaer Design Research Center while on leave from Leeds University.

References.

[1] Berzins M., (1988), Global error estimation in the method of lines for parabolic equations, SIAM J. Sci. Stat. Comput., 9, pp 687-703.

[2] Berzins M. and Dew P.M., (1981). A generalized Chebyshev method for parabolic p.d.e.s in one space variable . I.M.A. Journal of Numerical Analysis 1, pp. 469-487.

[3] Berzins M. and Dew P.M., (1987), A Note on C^0 Chebyshev Methods for Parabolic P.D.E.s, IMA J. of Numer. Anal., 7, pp 15-37.

[4] Berzins M. and Dew P.M. (1990), C^0 Chebyshev software for parabolic p.d.e.s, paper and algorithm accepted by ACM Trans. on Math. Soft.

[5] Berzins M., Dew P.M. and Furzeland R.M. (1989), Developing software for time dependent problems using the method of lines and differential algebraic integrators, Appl. Numer. Maths. 5 (1989) pp 375-397.

[6] Canuto C., Hussaini M.Y., Zang T.A. and Quateroni A., (1987), Spectral Methods in Fluid Dynamics, Springer-Verlag Series in Computational Physics.

[7] Davis S.F. and Flaherty J.E., (1982), An adaptive finite element method for initial boundary value problems for p.d.e.s, SIAM Jour. of Sci. and Stat. Comp. , Vol 3, No 1, March 1982, pp 6-26.

[8] Diaz J.C.,(1977) . A collocation-Galerkin method for the two point boundary value problem using continuous piecewise polynomial spaces. SIAM Jour. on Numerical Analysis, Vol 14, pp. 844-858.

[9] Funaro D. (1988) Domain decomposition methods for pseudo spectral approximations, part 1. second order equations in one dimension. Numer. Math. Vol 52 pp. 329-344 (1988).

[10] Hindmarsh A.C., (1978) ODEPACK - A systemised collection of o.d.e. solvers, in Advances in Computer Methods IV, R. Vichnevetsky and R.S.Stepleman, IMACS, New Brunswick, U.S.A..

[11] Hindmarsh A.C. (1986) Current methods for large stiff o.d.e. systems,
pp. 135- 144 in Numerical Mathematics and Applications, North Holland, Amsterdam, 1986. (eds) R. Vichnevetsky and J. Vignes .

[12] Leyk Z., (1986). A C^0 collocation - like Method for two point boundary value problems. Numer. Math. 49, pages 39-53.

[13] Lawson J., Berzins M. and Dew P.M., (1991) Balancing space and time errors in the method of lines solution of parabolic equations. (paper to appear in SIAM J. Sci. and Stat. Comp.)

[14] Lindberg B., (1977). Characterisation of Optimal Stepsize Sequences for Methods for Stiff Differential Equations, SIAM J. Numer. Anal., 14, pp 859-942.

[15] Machura M. and Sweet R.A. (1980). A survey of software for partial differential equations. A.C.M. Trans. on Math. Soft. Vol. 6 , No 4 , December 1980 , pp. 461-488.

[16] Oden J.T., Demkowicz L. , Rachowitz W. and Westerman T.A., (1989). Towards a universal H-P adaptive finite element strategy- Part 2 a posteriori error estimation. Comp. Meths. in Appl. Mech and Eng. 77, 113-180.

[17] Ortega J.M. and Voigt R.G. (1985) Solution of p.d.e.s on vector and parallel computers. SIAM Review, Vol 27, pp 149-240.

[18] Shampine L.F., (1987). Tolerance Proportionality in O.D.E. Codes, SMU Math. Rept. 87-8, Southern Methodist University, Texas 75275, U.S.A.

[19] Stetter H. J., (1978). Considerations Concerning a Theory for O.D.E. Solvers, Numerical Treatment of Differential Equations, ed. by R. Bulirsch, R.D. Grigorieff and J. Schroder, Lecture Notes in Mathematics 631, Springer Verlag, New York 188-200.

Appendix.

This Appendix contains the test problems used in the second part of Section 5. The seven test problems used are:

Problem 1. The p.d.e. is the one used by Berzins and Dew [3] as defined by

$$u\frac{\partial u}{\partial t} = \frac{1}{x^2}\frac{\partial}{\partial x}(x^2 u\frac{\partial u}{\partial x}) + 5u^2 + 4xu\frac{\partial u}{\partial x}$$

$$(x,t) \in [0,1] \times (0,1]$$

The left hand boundary condition is the symmetry condition $\frac{\partial u}{\partial x}(0\ ,\ t) = 0$ and the right hand Dirichlet

condition and the initial condition are consistent with the analytic solution of $u(x,t) = e^{1-x^2-t}$.

Problem 2 This problem was used by Berzins [1] to provide an example of a problem with a non-linear source term and with non-linear boundary conditions:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} - 2\left(\frac{\partial u}{\partial x}\right)^2 \frac{1}{u} - (2 + 4t^3 x)\, u^2 \quad , \quad (x\,,t) \in [0\,,1] \times (0\,,2],$$

with boundary conditions

$$\frac{\partial u}{\partial x}(0\,,t) = -u^2 t^4 \quad .$$

and

$$\frac{\partial u}{\partial x}(1\,,t) = -u^2 (-2 + t^4)$$

The initial conditions are consistent with the analytic solution

$$u(x\,,t) = \frac{1}{2 - x^2 + x\,t^4} .$$

Problem 3. This problem provides an example of a problem with a non-linear source term and a travelling wave solution:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + u^2 (1 - u)$$

with Dirichlet boundary conditions and initial conditions consistent with the analytic solution of

$$u(x\,,t) = \frac{1}{1 + e^{p(x - pt)}} ,$$

where $p = 0.5\sqrt{2}$.

Problem 4. This problem is the heat equation with Neumann boundary conditions:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \quad , \quad (x\,,t) \in [0\,,1] \times (0\,,0.25],$$

with the boundary conditions

$$\frac{\partial u}{\partial x}(x\,,t) = \pi e^{-\pi^2 t} \cos(\pi x)$$

at $x = 0$ and $x = 1$. The initial condition is consistent with the analytic solution

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

Problem 5.

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + e^{-2u} + e^{-u}, \quad x \in (0,1]$$

subject to the boundary conditions

$$u(0,t) + (t + P)\frac{\partial u}{\partial x}(0,t) = \log_e(t + P) + 1.0$$

$$u(1,t) + (1.0 + t + P)\frac{\partial u}{\partial x}(1,t) = \log_e(1.0 + t + P) + 1.0$$

The initial condition is consistent with the analytic solution of

$$= log(x + t + P)\ x>0\ .$$

and , $P = 1.0$. The problem was integrated from $t = 0$ to $t = 1.0$.

Problem 6 . A convection diffusion problem, known as Burgers' Equation, which is defined by

$$\frac{\partial u}{\partial t} = \varepsilon \frac{\partial^2 u}{\partial x^2} - u \frac{\partial u}{\partial x} \quad , \ (x\ ,t) \in (0,1)$$

where the value of $\varepsilon = 0.015$ was used in the experiments. The solution satisfies Dirichlet boundary conditions and initial conditions consistent with the analytic solution defined by

$$u(x\ ,t) = \frac{0.1A + 0.5B + C}{A + B + C}$$

where $A = e^{(-0.05(x - 0.5 + 4.95t)/\varepsilon)}$, $B = e^{(-0.25(x - 0.5 + 0.75t)/\varepsilon)}$ $C = e^{(-0.5(x - 0.375)/\varepsilon)}$.

Problem 7.

$$\frac{\partial u}{\partial t} = \frac{1}{C_1}\frac{\partial^2 u}{\partial x^2} + C_1 e^{-2u} + e^{-u}, \quad x \in [-1,0)$$

$$\frac{\partial u}{\partial t} = \frac{1}{C_2}\frac{\partial^2 u}{\partial x^2} + C_2 e^{-2u} + e^{-u}, \quad x \in (0,1]$$

subject to the boundary conditions

$$u(-1,t) = \log_e(-C_1 + t + P)$$

$$u(1,t) + (C_2 + t + P)\frac{\partial u}{\partial x}(1,t) = \log_e(C_2 + t + P) + 1.0$$

The initial condition is consistent with the analytic solution of

$$u(x,t) = log(C_1x + t + P) \; x \leq 0$$

$$= log(C_2x + t + P) \; x > 0 \; .$$

and , $P = 1.0$, $C_1 = 0.1$ $C_2 = 1.0$. The problem was integrated from $t = 0$ to $t = 1.0$. The Chebyshev C^0 collocation method was applied with two equally spaced elements . The interior break-point was situated at 0.5 .

Problem 8. The following test problem is due to Davis and Flaherty [7] .

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + f(x,t), \quad x \in (0,1]$$

where $f(x,t)$ is chosen do that the exact solution to the p.d.e. is the travelling wave form given by

$$u(x,t) = (1 - \tanh(10(x - t - 0.25)))/2$$

The initial condition and the Dirichlet boundary conditions are chosen to be consistent with this solution. The Chebyshev C^0 collocation method was applied with two equally spaced elements . The interior break-point was situated at 0.5 .