stichting
mathematisch
centrum

MC

AFDELING NUMERIEKE WISKUNDE NW 145/83 JANUARI
(DEPARTMENT OF NUMERICAL MATHEMATICS)

B.P. SOMMEIJER & P.J. VAN DER HOUWEN

SOFTWARE WITH LOW STORAGE REQUIREMENTS FOR TWO-DIMENSIONAL NONLINEAR PARABOLIC DIFFERENTIAL EQUATIONS

Preprint

kruislaan 413 1098 SJ amsterdam

Printed at the Mathematical Centre, Kruislaan 413, Amsterdam, The Netherlands.

1980 Mathematics subject classification: 65M20, 65L05, 65V05

Software with low storage requirements for two-dimensional nonlinear parabolic differential equations*)

by

B.P. Sommeijer & P.J. van der Houwen

ABSTRACT

The time-integration of a system of ODEs, originating from semi-discretization of a general, nonlinear parabolic differential equation in two space dimensions is described. The algorithm is based on a backward differentiation formula in combination with a nonlinear multigrid technique. To iterate the implicit relations in the process we employ an adapted form of nonlinear Chebyshev iteration.

The code is designed to minimize storage requirements and is provided with all the mechanisms necessary to automatic integration. The use of the code is illustrated by several examples and its behaviour is compared with PDETWO/PSETM/GEARB [10].

KEY WORDS & PHRASES: *software, parabolic differential equations multigrid methods, nonlinear Chebyshev iteration*

*) This report will be submitted for publication elsewhere.

1. INTRODUCTION

In a recent paper by Melgaard & Sincovec [9] numerical software is presented by which a rather general class of parabolic initial-boundary value problems in two space dimensions can be automatically integrated. This package (PDETWO/PSETM/GEARB) is based on the method of lines and consists of a semi-discretization part (PDETWO) which delivers the system of ordinary differential equations (ODEs) and of a time-integration part which is a modification of the widely-used and robust integrator GEARB of Hindmarsh [6]. This ODE integrator is provided with an iteration method indicator by which one can select functional iteration or some form of Newton iteration. In the first case the storage requirements are modest but this iteration usually needs excessively many right-hand side (RHS) evaluations. In the second case one may iterate with only the diagonal of the Jacobian matrix leading to an iteration process which has the same characteristics as functional iteration, or alternatively one may use the complete banded Jacobian. This last choice implies a considerable amount of storage (proportional to $3N^3$, where N is the number of meshpoints in one space direction) when dealing with two space dimensions. It should be noted that the last approach is recommended in case of one-dimensional problems [2]. In this case the Jacobian matrix usually possesses a tri-diagonal structure; hence, storage requirements are modest.

Consequently, using the package PDETWO/PSETM/GEARB, one has to choose between the "many RHS evaluations" option and "the large storage" option both of which may be unattractive from a computational point of view. The authors of PDETWO mention that it may be worth considering other methods for the solution of the matrix problem arising from applying Newton iteration to the backward differentiation formulas (BDFs) used in GEARB, for instance sparse matrix techniques [3].

Another possibility, which is the main tool of our time-integrator (henceforth indicated by BDMG) is the use of Multi Grid methods, directly applied to the nonlinear problem resulting from the BD formula.

It is the purpose of this paper to present a time-integrator for semi-discrete, parabolic initial-boundary value problems with minimal storage requirements (at most $7N^2$) and using considerably less RHS evaluations as required by the low storage options in GEARB. The numerical method is based on the BDF of second order which is solved by a nonlinear multigrid technique [8]. The coarsest-grid problem is solved by a Runge-Kutta type method with extended real stability interval and the implicit relations occurring on the finer grids are (approximately) solved by an adapted form of nonlinear Chebyshev iteration. The use of this special Runge-Kutta method and of Chebyshev iteration exploits the fact that parabolic problems give rise to Jacobian matrices with (more or less) negative eigenvalues, but at the same time restricts the applicability of our integrator BDMG to problems with such a negative spectrum. This limitation which is not shared by the PDETWO/PSETM/GEARB package of Melgaard and Sincovec, is the price we have to pay for the reduction of the amount of storage or of the computation time. Another restriction of BDMG is that only *scalar* parabolic equations can be treated.

The code BDMG, being a time-integrator for a system of ODEs, needs a semi-discrete approximation of the original parabolic problem. This semi-discretization can be performed by hand or can be left to PDETWO. In the first case the user has to write a subroutine in which a semi-discrete approximation of his problem is defined. In the second case he only has to write routines defining the partial differential equation and the boundary conditions (see the description of PDETWO [9]). We mention that the first possibility has the advantage that the execution time can be reduced and that no longer the restrictions of PDETWO apply (e.g. five-point coupling).

Finally, we remark that the code BDMG is organized in such a way that the Chebyshev part for relaxing the various implicit relations and the Runge-Kutta part for solving the coarsest-grid problem can be easily replaced by alternative subroutines which are suitable for dealing with, for instance, hyperbolic problems in which the Jacobian matrices involved have a more or less imaginary spectrum. In this connection, we observe that the basic linear multistep formula, i.e. the BDF_2, is A-stable and therefore allows for such an adaptation of BDMG. This is subject for future research.

2. THE CLASS OF PROBLEMS

Our time-integrator BDMG applies to semi-discrete parabolic equations in two space dimensions which can be written in the form

$$(2.1)\qquad \frac{du_{i,j}}{dt} = \tilde{f}_{i,j}(t,U), \qquad \begin{array}{l} i = 1,2,\ldots,NX \\ j = 1,2,\ldots,NY, \end{array} \qquad U = (u_{i,j})$$

where the (scalar) matrix elements $u_{i,j}$ are associated to the user-defined grid points (x_i,y_j) which form in the (x,y)-plane a grid with rectangular meshes wish mesh spacings $\Delta x_i = x_{i+1} - x_i$ and $\Delta y_j = y_{j+1} - y_j$ (see figure 2.1). Thus, any partial differential equation (PDE) of the form

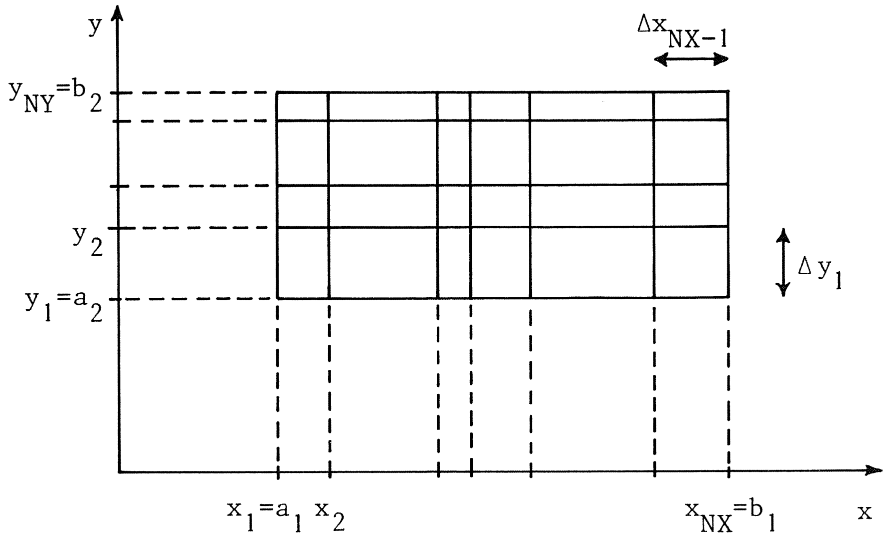

Fig. 2.1 Grid with rectangular meshes in the (x,y)-plane

$$(2.2a)\qquad \frac{\partial u}{\partial t} = f(t,x,y,u,\frac{\partial u}{\partial x},\frac{\partial u}{\partial y},\frac{\partial^2 u}{\partial x^2},\frac{\partial^2 u}{\partial x\partial y},\frac{\partial^2 u}{\partial y^2}), \qquad t \geq t_0$$

defined on a rectangle $a_1 < x < b_1$, $a_2 < y < b_2$ with initial data defined at $t = t_0$ on this rectangle and with boundary conditions of the form

$$(2.2b)\qquad A(t,x,y)u + B(t,x,y)\frac{\partial u}{\partial n} = C(t,x,y),$$

(n normal to the boundary) can be dealt with by BMDG as soon as (2.2a) and (2.2b) are semi-discretized on the grid $\{x_i,y_j\}$.

The semi-discretization of (2.2a) at interior grid points can be done by standard spatial variable differencing (see e.g. [11]). It should be noted that the system of ODEs (2.1) is defined for *all* grid points, including the boundary points. In case of Neumann or mixed (i.e. $B \neq 0$) boundary conditions the semi-discretization of (2.2b) may be achieved by differentiation with respect to t and forming an ordinary differential equation in t in the boundary points by discretization of $\partial^2 u/\partial n\, \partial t$. If $B = 0$ (i.e. Dirichlet boundary conditions) we define, analogously to Melgaard & Sincovec, the dummy equations $du_{i,j}/dt = 0$ at these points; since the value of $u_{i,j}$ is exactly determined by C/A, we do not need to integrate in time these ODEs. As a consequence of this approach, the elements $u_{i,j}$ corresponding to "Dirichlet boundary points" do not contain the approximate solution (except in case of a constant Dirichlet boundary condition). The reason for this approach is to preserve the structure of the system of ODEs. If, however, the problem (2.2) fits into the class of problems specified by Melgaard & Sincovec in their description of PDETWO (e.g. in the absence of mixed derivatives), then the semi-discretization can be achieved by linking PDETWO to BDMG (see section 4). We remark that the initial and boundary conditions are not required to be consistent. The system of ODEs (2.1) will be written more compactly as

$$(2.3) \qquad \frac{d\vec{V}}{dt} = \vec{g}(t,\vec{V}),$$

where the elements of $\vec{V}$ and $\vec{g}$ are $u_{i,j}$ and $\tilde{f}_{i,j}$, respectively, with $i = 1,2,\ldots,NX, \quad j = 1,2,\ldots,NY.$

3. THE NUMERICAL ALGORITHM

The numerical method on which BDMG is based has been fully described in [8] so that we will restrict ourselves to a rough outline of the method. Furthermore, we discuss in some detail local error estimation, step size strategy and the estimation of the spectral radius.

3.1 The BDF_2 formula

In the second and subsequent integration steps BDMG tries to solve the BDF_2 with nonuniform step size $\tau_n = t_{n+1} - t_n$, i.e.

$$(3.1) \qquad \vec{V}_{n+1} - \frac{1+q_n}{2+q_n}\tau_n\, \vec{g}(t_{n+1},\vec{V}_{n+1}) = \frac{1}{q_n(2+q_n)}\left[(1+q_n)^2\vec{V}_n - \vec{V}_{n-1}\right],$$

where $q_n = \tau_{n-1}/\tau_n$ and $\vec{V}_n$ denotes an approximation to $\vec{V}(t_n)$.

The reason for choosing a backward differentiation formula is the excellent stability behaviour of such formulas for parabolic problems. The reason for choosing only the second order method is our wish to limit the storage requirements. Moreover, in many problems a second order time integrator yields sufficient accuracy relative to the integration step used.

It will be convenient in the description of the multigrid method to write (3.1) compactly as

$$(3.1') \qquad L_n \vec{V}_{n+1} = \vec{\Sigma}_n,$$

where L_n is the operator defined by

$$L_n \vec{V} \equiv \vec{V} - b_0 \tau_n \vec{g}(t_{n+1}, \vec{V}), \qquad b_0 = \frac{1+q_n}{2+q_n}$$

and $\vec{\Sigma}_n$ is an abbreviation for the RHS of (3.1).

3.2 The multigrid method

For the numerical solution of (3.1) we apply a multigrid method which is claimed to be very efficient in elliptic problems (cf. [1,5]) and which does not require a lot of additional storage. Using the notation (3.1') we will shortly describe this technique for one particular time step (we omit the time index n if no ambiguities can arise).

Suppose we are given a sequence of M spatial grids (levels) with increasing mesh sizes on which semi-discretization of (2.2) has been performed. Applying the BDF_2 at each semi-discrete system we obtain a sequence of problems (cf. (3.1'))

$$(3.1'') \qquad L^k \vec{V}^k = \vec{\Sigma}^k, \qquad k = 1,2,\ldots,M,$$

where the upper index k is used to indicate the level, $\vec{V}^k$ is an approximation to the solution of (2.2) on level k and the operator L^k is defined according to (3.1'), i.e. $L^k \vec{V} \equiv \vec{V} - b_0 \tau_n \vec{g}^k(t, \vec{V})$. The level index 1 is associated with the coarest grid and level M with the finest grid, which we identify with the particular grid on which we want to solve our original problem (hence, (3.1") for k=M is identical to (3.1')).

There are several approaches in multigrid techniques. We follow the one which is called Full Approximation Scheme (FAS). For a thorough discussion of FAS we refer to [1]. In this process, the problems (3.1") on coarser grids (k<M) are modified by replacing their RHS $\vec{\Sigma}^k$ by $\vec{\tilde{\Sigma}}^k$, defined as

$$(3.2) \qquad \vec{\tilde{\Sigma}}^k = L^k(R\vec{v}^{k+1}) + R(\vec{\tilde{\Sigma}}^{k+1} - L^{k+1}\vec{v}^{k+1}),$$

where $\vec{v}^{k+1}$ is the current approximation to $\vec{V}^{k+1}$ on the next finer level and R denotes the Restriction operator, necessary to transfer a grid function on level k + 1 to a grid function on level k. Conversely, we will need a Prolongation operator P, transforming grid functions defined on level k into grid functions on level k + 1. Formally, both P and R may depend on the level k but for reasons of notational simplicity we omit a level index.

Starting with the extrapolation formula $\vec{v}^M = ((q_n+1)\vec{V}_n - \vec{V}_{n-1})/q_n$ (with $q_n = \tau_{n-1}/\tau_n$), being an approximation to $\vec{V}^M(=\vec{V}_{n+1})$, and using $\vec{v}^k = R\vec{v}^{k+1}$ as a first approximation on level k, we are now able to construct the row of modified right-hand side functions (3.2), successively for k = M - 1, M - 2, ..., 1.

We applied the FAS algorithm in a "fixed strategy" form, that is we
(i) solve the coarsest-grid problem

$$(3.3) \qquad L^1\vec{V}^1 = \vec{\tilde{\Sigma}}^1,$$

(ii) use, for k = 2,...,M, the approximation $\vec{v}^{k-1}$ to improve the current approximation $\vec{v}^k$ on the next finer level, i.e.

$$(3.4) \qquad \vec{v}^k := \vec{v}^k + P(\vec{v}^{k-1} - R\vec{v}^k),$$

and use this $\vec{v}^k$ as starting value in a relaxation process on level k. This relaxation is meant to smooth the error $\vec{V}^k - \vec{v}^k$.

The construction of the modified RHS functions (3.2) together with the above steps (i) and (ii) will be called a *sweep*. BDMG performs the fixed number of two sweeps. An argumentation for this choice can be found in [8]. Note that the result $\vec{v}^M$ of the first sweep is used to construct a new sequence of $\vec{\tilde{\Sigma}}^k$, k = M - 1,...,1 (cf. (3.2)) and that the resulting $\vec{v}^M$ of the second sweep is accepted as the numerical solution $\vec{V}_{n+1}$ at t_{n+1}.

3.2.1 The relaxation process

Since all RHS functions $\vec{g}^k(t,\vec{V}^k)$ originate from a parabolic problem the eigenvalues of the Jacobian matrices $\partial\vec{g}^k/\partial\vec{V}^k$ are assumed to lie in a narrow strip along the negative axis. To exploit this information we have chosen a Chebyshev type iteration process for relaxing the problems (3.1") with modified RHS (3.2). For a precise definition of this iteration process we refer to [8].

An important characteristic of the Chebyshev process is its flexibility to choose the damping interval and the factor by which the corresponding eigenvectors are damped. This factor can be decreased, simply by increasing the number of iterations. It is this flexibility that makes the Chebyshev process very well suited as a relaxation process in the multigrid method, the philosophy of which can be formulated as: "damp the low-frequency modes in the iteration error on the coarser grids". Hence, at level k, only those eigenvectors are damped which correspond to the "high-frequency" part of the spectrum at that particular level k. Damping of the low-frequency modes is postponed to the next coarser level k - 1, where these low frequencies are high frequencies, relative to level k - 1.

3.2.2 The coarsest-grid problem

During the second sweep we "solve" the coarsest-grid problem (3.3) by means of the Chebyshev process, discussed in the previous section with a small damping factor and a (relatively) large damping interval. In the first sweep, however, we follow another approach: first, we construct an ordinary differential equation the solution of which is a second order approximation to the solution of the coarsest-grid problem; next we solve this ODE using a second order, stabilized Runge-Kutta method [7].

This ODE-approach is followed because (i) a direct method to solve (3.3) may be time-consuming if the coarsest grid is still rather fine and (ii) an iterative approach usually requires a lot of iterations because the initial approximations are only of first order accuracy. A comparison of the iterative and ODE-approach can be found in [8].

3.2.3 Costs

In order to get an impression of the costs of this algorithm per integration step, we express the total number of RHS evaluations required at the various grids in terms of $\vec{g}^M$-evaluations, where we have assumed that the evaluation of $\vec{g}^k$ is four times as cheap as the evaluation of $\vec{g}^{k+1}$. Moreover, the spectral radii S^k of the Jacobian matrices $\partial\vec{g}^k/\partial\vec{V}^k$ are assumed to satisfy $S^{k+1} = 4S^k$.

In table 3.1 we list the converted equivalent of $\vec{g}^M$-evaluations for a number of M-values and for several values of $\tau_n S^M$ (in some way characteristic for the difficulty of the problem (3.1')).

$\tau_n S^M$	M = 2	M = 3	M = 4	M = 5	M = 6
100	9.3	9.5	9.1	9.1	9.1
500	12.0	11.3	10.5	10.1	9.4
1000	13.8	12.1	10.8	10.2	9.6
5000	20.3	13.6	12.1	10.8	9.8
10000	24.3	14.8	12.4	10.8	10.1
50000	35.8	16.8	13.0	11.2	10.2

Table 3.1

3.3 Starting values

The algorithm described in sections 3.1 and 3.2 needs the starting vectors $\vec{V}_0$ and $\vec{V}_1$ on the finest grid. The initial condition provides $\vec{V}_0$, the vector $\vec{V}_1$ is computed by applying exactly the same multigrid method to the one-step BDF formula (Backward Euler).

3.4 Error Control

The step sizes $\tau_n = t_{n+1} - t_n$ are chosen in such a way that the mixed error criterion

$$(3.5) \qquad (\mathrm{LTE})_n \leq \mathrm{TOL} + \mathrm{TOL}* \,\|\vec{V}_{n+1}\|, \qquad n = 0,1,\ldots$$

is satisfied, where $(\mathrm{LTE})_n$ denotes an estimate of the *local truncation error* in $[t_n,t_{n+1}]$ and TOL is a *user-provided tolerance parameter*. For $\|.\|$ we choose the divided Euclidean norm.

Apart from the above step size condition we require at each step point t_n the remaining integration interval $[t_n,t_{end}]$ to be a multiple of the current step size, i.e.

$$\tau_n := (t_{end}-t_n)/[\frac{t_{end} - t_n}{\tau_n} + 1],$$

where [·] denotes the integer part. In this way we achieve that the process arrives exactly at the prescribed endpoint t_{end}. Hence, interpolating the numerical solution at this point can be avoided as, in our experience, interpolation may be rather inaccurate when compared with the strategy described above.

3.4.1 Step size strategy

In order to minimize the overhead in calculating an estimate for the local error, all calculations involved are performed on the second finest grid (gridlevel M-1).

As an estimate of the local error for a method of order p we choose

(3.6) $$(LTE)_n = \| I\vec{V}_{n+1} - \vec{\tilde{V}}^{M-1}_{n+1} \|, \qquad n = 0,1,...,$$

where $\vec{\tilde{V}}^{M-1}_{n+1}$ is a reference solution of order p + 1 and the restrictor I is a simple Injection operator (cf. [5]).

This estimate is used for checking the step size condition (3.5) and for predicting the next integration step. Let us assume that (3.5) is satisfied by τ_n then τ_{n+1} is estimated as follows. From the definition of the reference solution we deduce

(3.7) $$\| I\vec{V}_{n+2} - \vec{\tilde{V}}^{M-1}_{n+2} \| = C_{n+1}\tau^{p+1}_{n+1} + \mathcal{O}(\tau^{p+2}_{n+1}), \qquad \text{as } \tau_{n+1} \to 0.$$

Assuming that the error constant C_{n+1} is slowly varying with n, we may write

(3.8) $$(LTE)_{n+1} = \| I\vec{V}_{n+2} - \vec{\tilde{V}}^{M-1}_{n+2} \| \simeq C_n\tau^{p+1}_{n+1} = \| I\vec{V}_{n+1} - \vec{\tilde{V}}^{M-1}_{n+1} \| . \left(\frac{\tau_{n+1}}{\tau_n}\right)^{p+1} .$$

Substitution into (3.5) and approximating $\| \vec{V}_{n+1} \|$ by $\| I\vec{V}_{n+1} \|$ yields

(3.9) $$\tau_{n+1} \le \bar{\alpha}_{n+1}\tau_n, \qquad \bar{\alpha}_{n+1} = \sqrt[p+1]{\frac{TOL + TOL* \| I\vec{V}_{n+1} \|}{\| I\vec{V}_{n+1} - \vec{\tilde{V}}^{M-1}_{n+1} \|}} .$$

We observe that $\bar{\alpha}_{n+1} \geq 1$ because τ_n was assumed to satisfy (3.5) (if $\bar{\alpha}_{n+1} < 1$ the step size τ_n is rejected, see below).

We now set

$$\tau_{n+1} = \alpha_{n+1}\tau_n,$$

where

$$\alpha_{n+1} = \begin{cases} 0.6\,\bar{\alpha}_{n+1} + 0.24 & \text{if } \begin{cases} 1 \leq \bar{\alpha}_{n+1} < 3.6 \\ 3.6 \leq \alpha_{n+1} \end{cases} \\ 2.4 & \end{cases} .$$

Because of the stability properties of the BDF_2 when applied with non-uniform step sizes, we cannot increase τ_{n+1}/τ_n unconditionally. It can be shown that the BDF_2 is A_0-stable provided that $\tau_{n+1} \leq 2.4\,\tau_n$ which leads to $\alpha_{n+1} \leq 2.4$.

If $\bar{\alpha}_{n+1} < 1$ then (3.5) is violated in the interval $[t_n, t_{n+1}]$; we reject the step and a new step size τ_n has to be calculated. The error constant C_n (cf.(3.8)) can be found from the rejected step size τ_n which leads to

$$(\tau_n)_{new} = \alpha_n . (\tau_n)_{rejected}, \qquad \alpha_n \leq \bar{\alpha}_{n+1},$$

where $\bar{\alpha}_{n+1}$ is defined in (3.9). In BDMG we choose

$$\alpha_n = \begin{cases} .1 \\ \frac{5}{9}\,\bar{\alpha}_{n+1} \end{cases} \quad \text{if} \quad \begin{cases} \bar{\alpha}_{n+1} \leq 0.18 \\ 0.18 < \bar{\alpha}_{n+1} < 1 \end{cases} .$$

The reference solution $\vec{\tilde{V}}^{M-1}_{n+1}$ in the first step is obtained by the trapezoidal rule; in the second step we choose a two-step, third order implicit linear multistep formula based on nonuniformly distributed step points and for all subsequent steps we use the third order, variable coefficient BDF formula.

3.4.2 <u>The initial step size</u>

In many codes the user is asked to provide an initial step size. Since this request is not always easily complied with, BDMG calculates an appropriate initial step size.

Following Shampine and Gordon [13] we use

$$(3.10) \qquad (\widetilde{LTE})_0 \doteq \tau_0^2\, \|\vec{g}^M(t_0,\vec{V}_0)\|$$

as an approximatition to the local error of a first order method. The step size condition (3.5) with $\vec{V}_{n+1} = \vec{V}_0$ suggests the initial step size

(3.11) $$\tau_0 = \alpha_0 \sqrt{\frac{\mathrm{TOL} + \mathrm{TOL}\,\|\vec{V}_0\|}{(\mathrm{LTE})_0}},$$

where α_0 is a strategy parameter.

In order not to be surprised by problems having an initial transient, we cautiously put $\alpha_0 = 0.1$. A detailed discussion on the estimation of the initial step size can be found in [14].

3.5 The spectral radius

As the Chebyshev iteration method (cf. section 3.2.1) is tuned to damp the high-frequency modes at level k, we need an estimate of S^k, the spectral radius of $\partial\vec{g}^k/\partial\vec{V}^k$. Besides the evaluation of S^k we also have to control its variation in time in case of nonlinear problems or time-dependent diffusion coefficients.

It should be noted that an underestimate of S^k may give rise to internal instabilities in the Chebyshev iteration (and also in the RK method to solve the coarsest-grid problem). Because these instabilities use to develop appallingly fast we have protected the code against overflow within the iteration process by controlling the norm of the iterates. If this test is violated the strong growth of the iterates is probably due to an underestimate of S^k. In that case we interrupt the current step and take action with respect to S^k (see below).

BDMG is provided with three options with regard to the estimation and control of the spectral radius:

I automatic option: with this option the code estimates and controls S^k during the entire integration interval. Its *estimation* is performed using Gerschgorin's disk theorem, assuming a five diagonal structure of the Jacobian matrix (this restriction is also made in the semi-discretization code PDETWO [9]). For the evaluation of the Jacobian elements we refer to [9], section 4.

Concerning the *control* of the S^k (again, upper indices refer to the grid level), we implemented the following strategy:

At the start of *each* integration step S_n^1, the spectral radius on the coarsest grid at $t=t_n$) is calculated and compared with S_{n-1}^1. Let $q_n \equiv S_n^1/S_{n-1}^1$. We now distinguish between two cases:

(i) $q_n \in [0.75, 1.15]$; in this case the slow variation of the spectral radius on the coarsest grid is assumed to hold on all grids and we simply set

(3.12) $$S_n^k = q_n \cdot S_{n-1}^k, \qquad k = 1,2,\ldots,M.$$

(ii) $q_n \notin [0.75,1.15]$; this fluctuation of S_n^1 suggests a strong variation in the solution, which may differ considerably on the various grids. Therefore, reevaluation of S_n^k, $k = 2,\ldots,M$ is performed (if $M \geq 3$ we calculate S_n^M by extrapolation of S_n^k, $k = 1,2,\ldots,M-1$).

Besides the above calculations of S_n^k at $t = t_n$ we also take into account the behaviour in time. In order not to be victimized by a spectral radius which is increasing in the interval $[t_n,t_{n+1}]$, we actually use the values $\tilde{S}_n^k$ defined by

(3.13) $$\tilde{S}_n^k = \max\{S_n^k,\ S_n^k + (S_n^k - S_{n-1}^k)\frac{\tau_n}{\tau_{n-1}}\}, \qquad k = 1,\ldots,M.$$

If, nevertheless, an "overflow" is detected in $[t_n,t_{n+1}]$ we verify whether the S_n^k were reevaluated at t_n or not. If so, we halve the step size and try again, otherwise we update the S_n^k - values as yet (taking care that we obtain significantly higher values than the ones causing the "overflow"). If, in the same interval again "overflow" occurs, we set $\tau_n := \tau_n/5$, repeatedly if necessary.

II <u>Semi-automatic option</u>: if the problem has a constant spectral radius (e.g. in linear problems) or if it is at least *non-increasing with time*, one may select the semi-automatic option which only calculates the spectral radius S_n^k, $k = 1,\ldots,M$ at the beginning of the integration process. No control is performed. If this situation applies this option may save a lot of RHS evaluations. Since again Gerschgorin's theorem is used one should only apply this option when the RHS function has a five-point coupling.

III <u>Non-automatic option</u>: some problems allow the user the calculation by hand of the spectral radius S_n^k so that an explicit expression in terms of t_n, the mesh sizes at level k, the current solution $\vec{V}_n$ and the current time-step τ_n can be given to be used on the interval $[t_n,t_{n+1}]$. Again, many RHS evaluations can be avoided and, if no use is made of PDETWO, the restriction to five-point coupling no longer applies. □

Whatever option is chosen, the overflow-checking is always performed. In case of options II & III, the integration process is discontinued when violating the overflow condition; control is returned to the main program and the user can take action with respect to the spectral radius.

4. USER DEFINITION OF THE PROBLEM

In defining his problem, the user is asked to provide a main program in which the (finest) grid is defined, the integration interval is specified and the initial condition is set. Because BDMG is a time-integrator for a system of ODE's it requires in addition a semi-discrete approximation to the original PDE. This semi-discretization can be performed either by hand or by PDETWO; by means of the parameter METH the user can indicate his choice. Selecting the first possibility, a subroutine is required delivering the derivative function $\vec{g}^k$ when called at level k. In case of the usage of PDETWO it suffices to supply routines defining the partial differential equation and the boundary conditions (see [9] for a detailed description of the user-required information).

To be more precise, we will shortly discuss the parameter list of BDMG; a detailed description of the parameters is given in Appendix B, where the listed comments in BDMG contain the essential details. We also refer to Appendix A where the usage of the package is illustrated by a source listing corresponding to an example discussed in section 5.

4.1. The parameter list

The parameter list, by which all information is passed from and to the integrator reads

```
BDMG(NX,NY,M,X,Y,T,TEND,V,G,SPRAD,TOL,METH,WORK,IWORK,INFO,IFLAG).
```

The meaning of the parameters is described in the following sections.

4.1.1. The grid-defining parameters

NX, NY	are the number of mesh lines in x- and y-direction, respectively (cf. section 2), corresponding to the original (in our multigrid approach called finest) grid.
M	is the number of successive grids, used in the multigrid method (see below).
X(1),...,X(NX)	contain the x-coordinates of the mesh points along each grid line in the x-direction (cf. section 2).

Y(1),...,Y(NY) contain the analogous information in the y-direction.

It is worth to go into more detail about these five parameters. The code will generate the additional grids by calculating internal values NX^k, NY^k, X^k and Y^k, defining the grid at level k. These values are necessary for several reasons, for example, to pass to the subroutine G (only if METH=1, see below), in facilitating the user in defining the derivative function on level k, but also for internal use in restrictions, prolongations etc. A grid at level k is obtained from a grid at level k + 1 by alternatingly deleting grid lines; hence, $NX^k = (NX^{k+1}+1)/2$, $X^k(1) = X^{k+1}(1)$, $X^k(2) = X^{k+1}(3)$ etc. This means that the grid lines on level k coincide with those on level k + 1. Because the values of the virtual arrays $X^k (k<M)$ are stored in the X-array too, the user should dimension this array larger than NX (see Appendix B for a precise definition). Of course, the same holds for the Y-array.

The number of grids should not be specified that large that one conflicts with one of the following situations:

(i) the above algorithm of deleting grid lines delivers a non-integer NX^k- (or NY^k-) value for some $k < M$

(ii) the grid on level 1 has less than four grid lines in each direction.

If none of these restrictions is applicable, the user is advised to choose M as large as possible, because the efficiency of the code usually increases as more grids are available. Hence, restriction (i) suggests a suitable value for NX and NY in relation to M. Finally we mention that specifying M = 1 is prohibited.

4.1.2 <u>The time-integration parameters</u>

T	is the current time; on entry T contains the initial value of the independent variable t. On exit, T contains the value TEND, unless an error has occurred.
TEND	specifies the point at which the solution is desired.
V	is an array to be dimensioned in the calling program as V(NX,NY); it containes the current solution values for all gridpoints. On entry, V should be given the initial value of the dependent variable in (2.3). On exit it contains the solution at the point T.
G	is the user-supplied subroutine defining the RHS function $\vec{g}^k$ at level k. This routine is only necessary in case of semi-discretization by hand (see the parameter METH below). Its specification reads

G(K,NXK,NYK,XK,YK,T,VK,DVK); the grid is determined by XK(I), I = 1,...,NXK and YK(J), J = 1,...,NYK, containing the coordinates of the gridlines at level k. Given the approximations at the current time T in the two-dimensional array VK, this routine should deliver the derivatives at the grid points in the two-dimensional array DVK.

SPRAD is a function the user must supply in case of the usage of the non-automatic option concerning the SPectral RADius (cf. section 3.5). SPRAD(K,NXK,NYK,XK,YK,T,VK,TAU) should deliver an estimate of the spectral radius of the Jacobian matrix of $\vec{g}^k$. The meaning of the parameters is the same as in the description of G, TAU being the current integration step.

TOL specifies the local error tolerance parameter (cf. section 3.4).

METH must be given the value 1 or 2. If METH = 1 the semi-discretization has been performed by hand and the subroutines G delivers the derivatives required by the integrator. METH = 2 means the semi-discretization has to be performed by PDETWO.

4.1.3 Auxiliary parameters

The last four parameters are discussed very briefly; a detailed description can be found in Appendix B.

WORK, IWORK	are one-dimensional work arrays for internal storage.
INFO	is an one-dimensional array containing information about the status of the integration process; some of its elements must be initialized in the main program.
IFLAG	is a flag used to indicate various error conditions.

4.1.4 Programming notes

We conclude this section with some notes:

(i) we emphasize that, using option III with respect to the spectral radius, the code relies on the user-specified estimate. There is a call to the function SPRAD at the start of each integration step and the user must be sure to deliver an estimate which holds on the whole interval $[t_n, t_{n+1}]$.

(ii) concerning the number of arrays required by BDMG, we note that the code needs five arrays on each grid plus one array on the next finest level (used for error estimation). Assuming that coarsening one level reduces the number of grid points to $\frac{1}{4}$, BDMG needs at most storage equivalent with *seven arrays on the finest grid.*

(iii) selecting METH = 2 (semi-discretization by PDETWO) requires the length of WORK to be increased with 7NX + 16 elements and usually results in an execution time much longer than in the case of METH = 1 (semi-discretization by hand).

(iv) in solving parabolic equations two types of errors arise, viz. space-discretization error and time-integration error. We mention that only the last one is controlled by BDMG (by means of the parameter TOL). Hence, in choosing a value of TOL, the user is advised to consider the accuracy of the space-discretization.

5. NUMERICAL EXAMPLES

To test the package BDMG, we solved a number of parabolic equations, two of which are described below. The main object of these examples is to illustrate the use of BDMG, to show its behaviour in reaction to a change in the process-defining parameters and to demonstrate its quite different characteristics with respect to storage and RHS evaluations when compared with PDETWO/PSETM/GEARB.

The first example mainly serves (i) to show the efficiency of the algorithm as a function of M, (the number of grids) and (ii) to demonstrate the influence of the spectral radius-option with respect to the number of RHS-functions. This example, for which also a complete source text can be found in Appendix A, is the so-called "porous-medium" equation [12]

$$(5.1) \qquad \frac{\partial u}{\partial t} = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)u^5, \qquad 0 \le t \le 1, \qquad 0 \le x,\ y \le 1.$$

The initial condition and the Dirichlet boundary conditions are taken from the exact solution

$$(5.2) \qquad u(t,x,y) = [\tfrac{4}{5}(2t+x+y)]^{\frac{1}{4}}.$$

We chose a uniform grid with $\Delta x = \Delta y = 1/24$ to semi-discretize (5.1), hence NX = NY = 25. The local error tolerance parameter TOL was set to 10^{-4}. By this choice the overall error at t = 1 is largly determined by the time-discretization error and not by the space-discretization error.

The selected grid size permitted us to choose M, the number of grids, equal to 2, 3 or 4; for these M-values results are reported. Moreover, the effect of the option concerning the spectral radius was tested. In case of selecting option III (i.e. calculation by hand) the spectral radius at level k was estimated by

$$(5.3) \qquad S^k(t) = 32(1+t).\left(\frac{1}{(\Delta x^k)^2} + \frac{1}{(\Delta y^k)^2}\right).$$

Note that the spectral radius is an increasing function in time which can be considered as a severe test of the spectral radius-control strategy, which is activated by selecting option I. The results of these tests (using METH=1, i.e. semi-discretization by hand) are presented in the tables 5.1 and 5.2, respectively where we used the abbreviations: NSTEPS for the number of steps performed including the rejected ones (in parentheses), NSE for the number of RHS-evaluations needed for the evaluation and control of the spectral radius, NGE is the total number of RHS-evaluations, including NSE. In fact, NSE and NGE are the converted equivalents of the numbers of RHS-evaluations on the finest grid. CD denotes the number of correct digits in the endpoint t = 1, defined by

$$(5.4) \qquad CD = -\log_{10}(\text{maximum of the absolute error}).$$

WK stands for the length of the work array (in words) and EXTIME for the execution time (in seconds); the calculations are performed on a CYBER 170-750.

M	NSTEPS	NSE	NGE	CD	WK	EXTIME
2	48(1)	-	652	5.23	3521	7.7
3	38(1)	-	452	5.81	3768	5.8
4	37(1)	-	395	5.54	3850	5.3

Table 5.1. BDMG for equation 5.1 with non-automatic option for the spectral radius.

M	NSTEPS	NSE	NGE	CD	WK	EXTIME
2	47(1)	75	700	5.16	3521	8.3
3	43(1)	17	517	5.86	3768	6.9
4	39(1)	5	416	5.77	3850	5.7

Table 5.2. BDGM for equation (5.1) with automatic option for the spectral radius.

From these tables we may conclude that - at least for this example - increasing the number of grids benefits the efficiency of the code, a tendency which was also observed in many other problems.

Furthermore, for this example, selection of the automatic option for the spectral radius results into a small increase of the number of RHS evaluations.

We repeated the above tests but now METH = 2 was selected (i.e. the semi-discretization was performed by PDETWO). The results of the integration (that is the numbers in the columns NSTEPS, NSE, NGE, and CD) differ slightly from the results given in the tables 5.1 and 5.2. However, a significant difference was found in the values of EXTIME, being in the average a factor 5.2 larger than the corresponding values obtained with METH = 1. Moreover, the length of the array WORK should be increased with 7 NX + 16 when using METH = 2.

We also applied PDETWO/PSETM/GEARB to the problem (5.1) in order to get an impression of the mutual relation between both codes. To that end, we have to write (5.1) in the form

$$(5.5)\qquad \frac{\partial u}{\partial t} = \frac{\partial}{\partial x}(5u^4 \frac{\partial u}{\partial x}) + \frac{\partial}{\partial y}(5u^4 \frac{\partial u}{\partial y}).$$

The GEARB routine is provided with an iteration method indicator. One can choose a "space-economic" option (either functional iteration or chord method with a diagonal approximation to the Jacobian), usually resulting into an extremely large number of RHS-evaluations. The other possibility is to select a "RHS-economic" option (chord method with complete, banded Jacobian) with excessive storage requirements as a consequence when applied to two-dimensional PDEs. The results of employing these three options are given in table 5.3. Here, NSE denotes the number of times the Jacobian matrix (or its diagonal approximation) is evaluated. Again, we set TOL = $= 10^{-4}$ and specified the initial step size equal to 10^{-5}.

method	NSTEPS	NSE	NGE	CD	WK	EXTIME
functional iteration	68980(0)	—	121119	4.52	6442	7013
diagonal Jacobian	5901(0)	6249	25857	3.60	7066	1471
banded Jacobian	33(0)	8	97	4.83	53941	19.2

Table 5.3. PDETWO/PSETM/GEARB for equation (5.5).

A comparison of both codes clearly indicates the great difference in performance: with respect to storage and execution time BDMG takes a middle-position between the extremes delivered by PDETWO/PSETM/GEARB. Therefore, we think BDMG can be a useful alternative in practical application.

The second example shows the performance of the code when it is applied to a PDE including a mixed derivative term. By "non-linearizing" an example given in [4], we constructed a problem of the form:

$$U_t = [d(t,x,y)(1+U)]^{10}[a(x,y)U_{xx} + 2b(x,y)U_{xy} + c(x,y)U_{yy}], \tag{5.6}$$

where $d(t,x,y) = (1+xy(x+y)e^{-t})^{-1}$, $a(x,y) = \frac{1}{2}x^2 + y^2$, $b(x,y) = -\frac{1}{2}(x^2+y^2)$ and $c(x,y) = x^2 + \frac{1}{2}y^2$. Equation (5.6) is defined on the unit square and the integration interval is [0,1]. The initial condition and the Dirichlet boundary conditions follow from the exact solution

$$U(t,x,y) = xy(x+y)e^{-t}. \tag{5.7}$$

We semi-discretized (5.6) on a uniform grid with mesh sizes $\Delta x = \Delta y = 1/16$. The number of grids equals 3, i.e. the maximal value allowed. We solved this problem for several values of TOL, viz. TOL = 10^{-i}, $i = 1,\dots,6$. In table 5.4. the results are listed, where NSTEPS, NGE and CD have the same meaning as in the first example of this section. Again, the number of rejected steps is put in parentheses.

TOL	NSTEPS	NGE	CD
10^{-1}	5(0)	60	2.44
10^{-2}	7(0)	79	2.60
10^{-3}	13(0)	140	3.74
10^{-4}	26(0)	257	3.93
10^{-5}	50(0)	467	4.66
10^{-6}	91(0)	810	5.21

Table 5.4. Results for problem (5.6) for several values of TOL.

No comparison can be made with PDETWO/PSETM/GEARB because of the presence of the mixed derivative term in (5.6), which is one of the restrictions to PDETWO. At the same time, however this term forces us to specify the spectral radius explicitly. We used the estimate

$6*(1/(\Delta x^k)^2 + 1/(\Delta y^k)^2) + 2/(\Delta x^k)(\Delta y^k)$, where Δx^k and Δy^k are the mesh sizes at level k in x- and y-direction, respectively.

REFERENCES

[1] BRANDT, A. & N. DINAR, *Multigrid solutions to elliptic flow problems,* in Numerical Methods for Partial Differential Equations, S.V. Parter (ed.), Academic Press, 1979.

[2] DEW, P.M. & J.E. WALSH, *A set of library routines for solving parabolic equations in one space variable,* ACM Trans. Math. Softw., Vol. 7, September 1981, pp. 295-314.

[3] EISENSTAT, C.S., M.H. SCHULTZ & A.H. SHERMAN, *Application of sparse matrix methods to partial differential equations,* in Proc. AICA Int. Symp. on Computer Methods for Partial Differential Equations, Bethlehem, Pa., June 1975.

[4] GOURLAY, A.R. & S. McKEE, *The construction of hopscotch methods for parabolic and elliptic equations in two space dimensions with a mixed derivative,* J. of Comp. and Appl. Math. 3, 1977, pp. 201-206.

[5] HEMKER, P.W., *Introduction to multi-grid methods,* Nieuw Arch. Wiskunde (3), XXIX 1981, pp. 71-101.

[6] HINDMARSH, A.C., *GEARB: Solution of ordinary differential equations having banded Jacobian,* Rep. UCID-30059, Rev. 2, Lawrence Livermore Lab., Livermore, Calif., June 1977.

[7] HOUWEN, P.J. van der & B.P. SOMMEIJER, *A special class of multistep Runge-Kutta methods with extended real stability interval,* IMA Journal of Num. Anal. 2, 1982, pp. 183-209.

[8] HOUWEN, P.J. van der & B.P. SOMMEIJER, *Analysis of Chebyshev relaxation in multigrid methods for nonlinear parabolic differential equations,* to appear in ZAMM.

[9] MELGAARD, D.K. & R.F. SINCOVEC, *General software for two-dimensional nonlinear partial differential equations*, ACM Trans. Math. Softw., Vol. 7, March 1981, pp. 106-125.

[10] MELGAARD, D.K. & R.F. SINCOVEC, *Algorithm 565, PDETWO/PSETM/GEARB: Solution of systems of two-dimensional nonlinear partial differential equations,* ACM Trans. Math. Softw., Vol. 7, March 1981, pp. 126-135.

[11] MITCHELL, A.R. & D.F. GRIFFITHS, *The Finite Difference Method in Partial Differential Equations,* Wiley, Chichester, U.K., 1980.

[12] RICHTMYER, R.D. & K.W. MORTON, *Difference methods for initial-value problems,* Interscience, New York, 1967.

[13] SHAMPINE, L.F. & M.K. GORDON, *Computer Solution of Ordinary Differential Equations: Initial Value Problems,* W.H. Freeman, San Francisco, 1975.

[14] WATTS, H.A., *Starting step size for an ODE solver,* Sandia report SAND 80-1734, Sandia National Laboratories, Albuquerque, 1982.

Appendix A.

Here, we give the main program and the subroutines corresponding to example 1 of section 5 to illustrate the usage of the package BDMG.

```
      PROGRAM EXAM1(OUTPUT,TAPE6=OUTPUT)
C-----------------------------------------------------------------------
C   MAIN PROGRAM TO ILLUSTRATE THE USAGE OF BDMG USING EXAMPLE 1
C   OF SECTION 5
C-----------------------------------------------------------------------
      DIMENSION V(25,25),X(49),Y(49),INFO(18)
      DIMENSION WORK(3850),IWORK(43)
      EXTERNAL G,SPRAD
C----------------------------------------------------------------
C   DEFINITION OF THE FINEST GRID AND OF THE NUMBER OF GRIDS
C----------------------------------------------------------------
      NX=25
      NY=25
      DX=1.0/(NX-1.0)
      DO 10 I=1,NX
   10 X(I)=(I-1)*DX
      DY=1.0/(NY-1.0)
      DO 20 I=1,NY
   20 Y(I)=(I-1)*DY
      M=4
C-----------------------------------------------------------------
C   DEFINITION OF THE INTEGRATION INTERVAL AND INITIALIZATION
C-----------------------------------------------------------------
      T=0.0
      TEND=1.0
      DO 30 J=1,NY
      DO 30 I=1,NX
   30 V(I,J)=SOL(T,X(I),Y(J))
C---------------------------------------
C   DEFINITION OF THE PROBLEM PARAMETERS
C---------------------------------------
      TOL=1.0E-4
      INFO(1)=0
      INFO(2)=2000
      INFO(3)=3
      IWORK(43)=3850
      METH=1
C--------------------------
C   CALL FOR THE INTEGRATOR
C--------------------------
      CALL BDMG(NX,NY,M,X,Y,
     +         T,TEND,V,G,SPRAD,TOL,METH,
     +         WORK,IWORK,INFO,IFLAG)
C---------------------------
C   CHECK ERRORFLAG ON RETURN
C---------------------------
      IF(IFLAG.NE.0)WRITE(6,1000)IFLAG
      IF(IFLAG.GE.1 .AND. IFLAG.LE.4) GOTO 60
```

```
C--------------------------
C   OUTPUT SOME STATISTICS
C--------------------------
      WRITE(6,1001)INFO(7),INFO(8),INFO(9)
      WRITE(6,1002)INFO(10)
C--------------------------------------------
C   PRINT THE SOLUTION AND DETERMINE THE ERROR
C--------------------------------------------
      WRITE(6,1003)T
      NXM1=NX-1
      NYM1=NY-1
      AE=-1.0
      DO 50 J=2,NYM1
      WRITE(6,1004)J
      DO 40 I=2,NXM1
      AE=AMAX1(AE,ABS(SOL(T,X(I),Y(J))-V(I,J)))
   40 CONTINUE
   50 WRITE(6,1005)(V(I,J),I=2,NXM1)
      WRITE(6,1006)-ALOG10(AE)
 1000 FORMAT(27H ERRORFLAG HAS BEEN SET TO:,I5)
 1001 FORMAT(27H NUMBER OF STEPS PERFORMED:,I6/
     +       26H NUMBER OF REJECTED STEPS:,I6/
     +       31H NUMBER OF STEPS WITH OVERFLOW:,I6)
 1002 FORMAT(44H TOTAL NUMBER OF (FINEST GRID)G-EVALUATIONS:,I6)
 1003 FORMAT(//15H SOLUTION AT T=,F10.4/)
 1004 FORMAT(I4)
 1005 FORMAT(5(E20.10))
 1006 FORMAT(34H MINIMAL NUMBER OF CORRECT DIGITS:,F7.2)
C------------------------------------
C   END OF MAIN PROGRAM FOR EXAMPLE 1
C------------------------------------
   60 STOP
      END



      FUNCTION SOL(T,X,Y)
C---------------------------------------------------
C   ANALYTIC FUNCTION USED FOR INITIALIZATION,BOUNDARY
C   CONDITIONS AND CALCULATION OF THE ERROR
C---------------------------------------------------
      SOL=(0.8*(2.0*T+X+Y))**0.25
      RETURN
      END


      FUNCTION SPRAD(K,NXK,NYK,XK,YK,T,VK,TAU)
      DIMENSION XK(NXK),YK(NYK),VK(NXK,NYK)
C-----------------------------------------------------
C   DEFINE AN UPPER ESTIMATE OF THE SPECTRAL RADIUS ON THE
C   INTERVAL [T,T+TAU]
C-----------------------------------------------------
      SPRAD=32.0*(1.0+T+TAU)*((NXK-1)**2+(NYK-1)**2)
      RETURN
      END
      SUBROUTINE G(K,NXK,NYK,XK,YK,T,VK,DVK)
      DIMENSION XK(NXK),YK(NYK),VK(NXK,NYK),DVK(NXK,NYK)
C--------------------------------------------
C   DEFINE THE DERIVATIVE FUNCTION G AT LEVEL K
C--------------------------------------------
      NXKM1=NXK-1
      NYKM1=NYK-1
C-----------------------------------
C   TREATMENT OF THE BOUNDARY POINTS
C-----------------------------------
      DO 10 J=1,NYK
      DO 10 I=1,NXK,NXKM1
      DVK(I,J)=0.0
   10 VK(I,J)=SOL(T,XK(I),YK(J))
      DO 20 J=1,NYK,NYKM1
      DO 20 I=2,NXKM1
      DVK(I,J)=0.0
   20 VK(I,J)=SOL(T,XK(I),YK(J))
C--------------------------------
C   DERIVATIVES AT INTERNAL POINTS
C--------------------------------
      DO 30 J=2,NYKM1
      DO 30 I=2,NXKM1
   30 DVK(I,J)=(VK(I+1,J)**5-2.0*VK(I,J)**5+VK(I-1,J)**5)*NXKM1**2+
     +         (VK(I,J+1)**5-2.0*VK(I,J)**5+VK(I,J-1)**5)*NYKM1**2
      RETURN
      END
```

Appendix B.

Here, a part of the listing of BDMG is printed, containing the essential details on the use of the package. The complete listing is available from the authors.

```
      SUBROUTINE BDMG(NX,NY,M,X,Y,
     *                T,TEND,V,G,SPRAD,TOL,METH,
     *                WORK,IWORK,INFO,IFLAG)
C
C-----------------------------------------------------------------------
C
C   PURPOSE AND METHOD
C
C-----------------------------------------------------------------------
C
C      BDMG IS A TIME-INTEGRATOR DESIGNED TO SOLVE A SYSTEM OF ORDINARY
C    DIFFERENTIAL EQUATIONS(ODE'S) ORIGINATING FROM SEMI-DISCRETIZATION
C    OF A (SCALAR) GENERAL PARABOLIC PARTIAL DIFFERENTIAL EQUATION(PDE)
C    IN TWO SPACE DIMENSIONS. THIS SEMI-DISCRETIZATION CAN BE PERFORMED
C    EITHER BY HAND OR BY PDETWO[2], AN INTERFACE TO FORM AND EVALUATE
C    A SEMI-DISCRETE APPROXIMATION TO THE ORIGINAL PDE.
C
C    THE METHOD ON WHICH BDMG IS BASED [1], CONSISTS OF THE APPLICATION
C    OF THE SECOND ORDER BACKWARD DIFFERENTIATION FORMULA TO THIS SEMI-
C    DISCRETE SYSTEM. THE RESULTING NON-LINEAR PROBLEM IS SOLVED USING A
C    MULTIGRID TECHNIQUE.
C
C    THE MAIN CHARACTERISTIC OF THE CODE BDMG IS ITS MINIMAL STORAGE
C    REQUIREMENTS.
C
C    THE APPLICABILITY OF BDMG IS RESTRICTED TO
C    (1)  TWO SPACE DIMENSIONS
C    (2)  RECTANGULAR DOMAINS
C    (3)  SCALAR PARTIAL DIFFERENTIAL EQUATIONS
C    (4)  PDE'S IN WHICH DIFFUSION GREATLY DOMINATES CONVECTION
C    (5)  PDE'S WITHOUT MIXED DERIVATIVES(THIS RESTRICTION ONLY APPLIES
C         IF PDETWO IS USED)
C
C-----------------------------------------------------------------------
C
C   PROBLEM DEFINITION
C
C-----------------------------------------------------------------------
C
C      BDMG APPLIES TO SEMI-DISCRETE SYSTEMS OF THE FORM
C
C
C              DV
C   (1)     [ -- = G(T,V) ]              ,I=1,...,NX
C              DT          I,J            J=1,...,NY    ,
C
C
```

```
C     WHERE THE (SCALAR) MATRIX ELEMENTS V(I,J) ARE ASSOCIATED TO THE
C     GRID POINTS (X(I),Y(J)) WHICH FORM A GRID WITH RECTANGULAR MESHES
C     IN THE (X,Y)-PLANE, INCLUDING THE BOUNDARY POINTS.
C     THIS SYSTEM IS SUPPOSED TO ORIGINATE FROM PARTIAL DIFFERENTIAL
C     EQUATIONS OF THE FORM
C
C
C     (2)      UT = F(T,X,Y,U,UX,UY,UXX,UYY,UXY) ,
C
C     WHERE UT       IS THE FIRST PARTIAL DERIVATIVE OF U WITH RESPECT TO T
C     AND   UX,UXX ARE THE FIRST AND SECOND PARTIAL DERIVATIVE OF U WITH
C                    RESPECT TO X
C           ETC.
C     THUS, ANY PDE OF THE FORM (2) DEFINED ON A RECTANGLE WITH INITIAL
C     CONDITION DEFINED ON THIS RECTANGLE AND BOUNDARY CONDITIONS OF
C     THE FORM
C
C     (3)    A(T,X,Y)*U + B(T,X,Y)*UN = C(T,X,Y),  UN IS DERIVATIVE OF U
C                                                  NORMAL TO THE BOUNDARY
C
C     CAN BE DEALT WITH BY BDMG AS SOON AS (2) AND (1) ARE SEMI-
C     DISCRETIZED ON THE GRID (X(I),Y(J)).
C
C     THE INITIAL CONDITION BEING DEFINED FOR EACH V(I,J) NEEDS NOT BE
C     CONSISTENT WITH THE BOUNDARY CONDITIONS.
C
C-----------------------------------------------------------------------
C
C    USER-REQUIRED INFORMATION
C
C-----------------------------------------------------------------------
C
C        IN DEFINING HIS PROBLEM, THE USER IS ASKED TO PROVIDE A MAIN
C     PROGRAM IN WHICH THE GRID IS DEFINED, THE INTEGRATION INTERVAL IS
C     SPECIFIED AND THE INITIAL CONDITION IS SET. BECAUSE BDMG IS A
C     TIME-INTEGRATOR FOR A SYSTEM OF ODE'S IT REQUIRES IN ADDITION A
C     SEMI-DISCRETE APPROXIMATION TO THE ORIGINAL PDE. THIS SEMI-
C     DISCRETIZATION CAN BE PERFORMED EITHER BY HAND OR BY PDETWO;
C     BY MEANS OF THE PARAMETER METH THE USER CAN INDICATE HIS CHOICE.
C     SELECTING THE FIRST POSSIBILITY, A SUBROUTINE IS REQUIRED
C     DELIVERING THE DERIVATE FUNCTION G IN (1). IN CASE OF THE USAGE
C     OF PDETWO IT SUFFICES TO SUPPLY ROUTINES DEFINING THE PARTIAL
C     DIFFERENTIAL EQUATION AND THE BOUNDARY CONDITIONS (SEE [2] FOR A
C     DETAILED DESCRIPTION OF THE USER-REQUIRED INFORMATION).
C
C-----------------------------------------------------------------------
C
C    THE PARAMETERS
C
C-----------------------------------------------------------------------
C
C        WE DISTINGUISH BETWEEN THREE TYPES OF PARAMETERS:
C
C     GRID-DEFINING PARAMETERS
C     ------------------------
C
C     NX,NY               ARE THE NUMBER OF MESH LINES IN X- AND Y-DIRECTION,
C                         RESPECTIVELY
C     M                   IS THE NUMBER OF SUCCESSIVE GRIDS, USED IN THE
C                         MULTIGRID METHOD(FOR A DISCUSSION OF M SEE BELOW).
C     X(1),...,X(NX)      CONTAIN THE X-COORDINATES OF THE MESH POINTS
C                         ALONG EACH GRID LINE IN THE X-DIRECTION.
C                         (X(1) .LT. X(2) .LT. ... .LT. X(NX)).
C     Y(1),...,Y(NY)      CONTAIN THE ANALOGOUS INFORMATION IN THE
C                         Y-DIRECTION.
C                         (Y(1) .LT. Y(2) .LT. ... .LT. Y(NY)).
C
```

```
C     NX,NY,M,X,Y ARE INPUT PARAMETERS AND HAVE TO BE INITIALIZED BY
C     THE USER.
C     NOTE THAT X AND Y ARE NOT NECESSARILY EQUIDISTANT.
C
C        IT IS WORTH TO GO INTO MORE DETAIL ABOUT THESE PARAMETERS.
C     BECAUSE THE METHOD IS BASED ON A MULTIGRID METHOD, A SEQUENCE OF M
C     SUCCESSIVE GRIDS IS NECESSARY; THESE GRIDS ARE GENERATED BY THE
C     CODE. THE LEVEL INDEX K RUNS FROM 1 (WHICH IS ASSOCIATED WITH THE
C     COARSEST GRID) UNTIL M (THE FINEST GRID, WHICH IS IDENTICAL TO THE
C     GRID CHOOSEN BY THE USER AND DEFINED BY MEANS OF THE PARAMETERS
C     NX,NY,X,Y). INTERNAL VALUES NXK, NYK ARE CALCULATED AS WELL AS
C     ARRAYS XK(I),I=1,...,NXK AND YK(J),J=1,...,NYK, DEFINING THE GRID
C     AT LEVEL K. THESE VALUES ARE USED, AMONG OTHER THINGS, TO PASS TO
C     THE SUBROUTINE G (SEE BELOW) IN WHICH THE USER HAS TO DEFINE THE
C     DERIVATIVE AT THAT PARTICULAR GRID (XK(I),YK(J)) (ONLY IF METH=1,
C     SEE BELOW).
C
C        A GRID AT LEVEL K IS OBTAINED FROM A GRID AT LEVEL K+1 (DENOTED
C     BY KP1) BY ALTERNATINGLY DELETING GRID LINES;HENCE, NXK=(NXKP1+1)/2
C     XK(1)=XKP1(1), XK(2)=XKP1(3), ETC. THIS MEANS THAT THE GRID LINES
C     ON LEVEL K COINCIDE WITH THOSE ON LEVEL K+1. BECAUSE THE VALUES OF
C     THE VIRTUAL ARRAYS XK (K<M) ARE STORED IN THE X-ARRAY TOO, THE USER
C     SHOULD DIMENSION THIS ARRAY AS X(M+(NX-1)*(2**M-1)/(2**(M-1))).
C     SIMILARLY, THE Y-ARRAY MUST BE DECLARED OF LENGTH
C     M+(NY-1)*(2**M-1)/(2**(M-1)).
C
C        THE VALUES OF NX,NY AND M SHOULD BE CHOOSEN IN A SUITABLE
C     RELATIONSHIP TO EACH OTHER. THE NUMBER OF GRIDS SHOULD NOT BE
C     SPECIFIED THAT LARGE THAT ONE CONFLICTS WITH ONE OF THE FOLLOWING
C     SITUATIONS:
C     (I)  THE ABOVE ALGORITHM OF DELETING GRID LINES DELIVERS A
C          NON-INTEGER NXK- OR NYK-VALUE FOR SOME K<M.
C     (II) THE GRID ON LEVEL 1 HAS LESS THAN FOUR GRID LINES IN EACH
C          DIRECTION, INCLUDING THE GRID LINES FORMING THE BOUNDARIES.
C
C     IF NONE OF THESE RESTRICTIONS IS APPLICABLE, THE USER IS ADVISED TO
C     CHOOSE M AS LARGE AS POSSIBLE, BECAUSE THE EFFICIENCY OF THE CODE
C     USUALLY INCREASES AS MORE GRIDS ARE AVAILABLE.
C
C     FINALLY, WE MENTION THAT SPECIFYING M=1 IS PROHIBITED.
C
C
C     TIME-INTEGRATION PARAMETERS
C     ---------------------------
C
C     T       IS THE CURRENT TIME; ON ENTRY, T CONTAINS THE INITIAL VALUE
C             OF THE INDEPENDENT VARIABLE IN (1). ON EXIT, T CONTAINS THE
C             VALUE TEND, UNLESS AN ERROR HAS OCCURRED (SEE THE PARAMETER
C             IFLAG BELOW).
C     TEND    SPECIFIES THE POINT AT WHICH THE SOLUTION IS DESIRED
C             (TEND MUST BE .GT. T ON ENTRY).
C     V       IS AN ARRAY TO BE DIMENSIONED IN THE CALLING PROGRAM AS
C             V(NX,NY); IT CONTAINS THE CURRENT SOLUTION VALUES FOR ALL
C             GRID POINTS (SEE ALSO THE REMARKS ON BOUNDARY POINTS IN THE
C             DESCRIPTION OF THE PARAMETER G). ON ENTRY, V SHOULD BE GIVEN
C             THE INITIAL VALUE OF THE DEPENDENT VARIABLE IN (1).
C             ON EXIT, V CONTAINS THE SOLUTION AT THE POINT T.
C     G       IS THE USER-SUPPLIED SUBROUTINE DEFINING THE RIGHT-HAND SIDE
C             FUNCTION G AT LEVEL K. THIS ROUTINE IS ONLY NECESSARY IN
C             CASE OF SEMI-DISCRETIZATION BY HAND (SEE THE PARAMETER METH
C             BELOW). ITS SPECIFICATION READS
C                G(K,NXK,NYK,XK,YK,T,VK,DVK)
C                DIMENSION XK(NXK),YK(NYK),VK(NXK,NYK),DVK(NXK,NYK)
```

```
C           THE GRID IS DETERMINED BY XK(I),I=1,...NXK AND YK(J),
C           J=1,...,NYK, CONTAINING THE COORDINATES OF THE GRID LINES AT
C           LEVEL K. GIVEN THE APPROXIMATIONS AT THE CURRENT TIME T IN
C           THE TWO-DIMENSIONAL ARRAY VK, THIS ROUTINE SHOULD DELIVER
C           THE DERIVATIVES AT ALL GRID POINTS IN THE TWO-DIMENSIONAL
C           ARRAY DVK, INCLUDING THE DERIVATIVES AT THE BOUNDARY POINTS.
C           G MUST BE DECLARED EXTERNAL IN THE CALLING PROGRAM.
C
C           IN ORDER TO PRESERVE THE STRUCTURE OF THE SYSTEM OF ODE'S,
C           IRRESPECTIVE OF THE TYPE OF BOUNDARY CONDITIONS, THE
C           BOUNDARY POINTS ARE INCLUDED IN THE GRID AND THUS IN THE
C           ARRAYS VK AND DVK.
C           HOWEVER, IN CASE OF DIRICHLET BOUNDARY CONDITIONS TIME-
C           INTEGRATION AT THESE POINTS SEEMS TO BE SUPERFLUOUS;
C           THEREFORE, THE DERIVATIVES AT THESE POINTS MAY BE GIVEN A
C           DUMMY VALUE. IF THIS POSSIBILITY IS USED, ONE SHOULD SPECIFY
C           THE VALUE ZERO FOR THESE DERIVATIVES.
C           AS A CONSEQUENCE OF THIS APPROACH, THE ELEMENTS IN V
C           CORRESPONDING TO BOUNDARY POINTS WHERE A DIRICHLET CONDITION
C           IS PRESCRIBED DO NOT CONTAIN THE APPROXIMATE SOLUTION AT
C           TIME T (EXCEPT WHEN THE DIRICHLET CONDITION IS A CONSTANT,
C           IN WHICH CASE THE VALUE ZERO EQUALS THE DERIVATIVE). HENCE,
C           IN CALCULATING THE DERIVATIVE IN A POINT ADJACENT TO A
C           "DIRICHLET BOUNDARY POINT" ONE SHOULD USE THE DIRICHLET
C           BOUNDARY CONDITION RATHER THAN THE VALUE OF VK.
C
C           IN CASE OF USING PDETWO TO PERFORM THE SEMI-DISCRETIZATION,
C           THE SAME APPROACH IS FOLLOWED IN "DIRICHLET BOUNDARY POINTS"
C     SPRAD IS A FUNCTION THE USER MUST SUPPLY IN CASE OF THE USAGE OF
C           THE NON-AUTOMATIC OPTION CONCERNING THE SPECTRAL RADIUS (SEE
C           THE PARAMETER INFO BELOW).
C           SPRAD(K,NXK,NYK,XK,YK,T,VK,TAU) SHOULD DELIVER AN ESTIMATE
C           OF THE THE SPECTRAL RADIUS OF THE JACOBIAN MATRIX OF THE
C           RIGHT-HAND SIDE FUNCTION G AT LEVEL K. THE MEANING OF THE
C           PARAMETERS OF SPRAD IS THE SAME AS IN THE DESCRIPTION OF G,
C           TAU BEING THE CURRENT INTEGRATION STEP.
C           SPRAD MUST BE DECLARED EXTERNAL IN THE CALLING PROGRAM.
C     TOL   SPECIFIES THE LOCAL ERROR TOLERANCE PARAMETER
C           (TOL MUST BE .GT. 0).
C     METH  MUST BE GIVEN THE VALUE 1 OR 2.
C           IF METH=1 THE SEMI-DISCRETIZATION HAS TO BE PERFORMED BY
C                     HAND AND THE SUBROUTINE G DELIVERS THE DERIVATIVES
C                     REQUIRED BY THE INTEGRATOR.
C              METH=2 MEANS THE SEMI-DISCRETIZATION HAS TO BE PERFORMED
C                     BY PDETWO.
C
C
C     AUXILIARY PARAMETERS
C     --------------------
C
C
C     WORK  IS A REAL WORK ARRAY FOR INTERNAL STORAGE. IT MUST BE
C           DIMENSIONED AS WORK(**) IN THE CALLING PROGRAM, WHERE
C           ** .GE. (NX-1)*(NY-1)*(71-5/(4**(M-2)))/12+
C                   (NX+NY-2)*(19-5/(2**(M-2)))/2+
C                   7*M+3+
C                   (7*NX+16)*(METH-1)
C
C           THE USER SHOULD BE AWARE THAT THIS EXPRESSION MUST
C           DELIVER AN INTEGER VALUE. IF NOT, ONE SHOULD CAREFULLY
C           RECONSIDER THE VALUES OF NX,NY AND M.
```

```
C     IWORK  IS AN INTEGER WORK ARRAY CONTAINING POINTERS USED TO
C            PROVIDE DYNAMIC DIMENSIONING IN BDMG. IT MUST BE DIMENSIONED
C            AS IWORK(***) IN THE CALLING PROGRAM, WHERE
C            *** .GE. 10*M+3.
C            IWORK(10*M+3) MUST BE INITIALIZED WITH THE VALUE OF
C                          ** (SEE THE PARAMETER WORK).
C     INFO   IS AN INTEGER ARRAY CONTAINING INFORMATION ABOUT THE STATUS
C            OF THE INTEGRATION PROCESS; IT MUST BE DIMENSIONED AS
C            INFO(****), WHERE
C            **** .GE. 2*M+10
C
C            INFO(1),...,INFO(3)  ARE INPUT PARAMETERS AND MUST BE
C                                 INITIALIZED IN THE CALLING PROGRAM
C            INFO(7),...,INFO(2*M+10)  ARE OUTPUT PARAMETERS.
C
C            THE ELEMENTS OF INFO HAVE THE FOLLOWING MEANINGS:
C            INFO(1)=0 TO INDICATE THAT THIS IS THE FIRST CALL TO BDMG.
C                      ON RETURN, BDMG HAS ASSIGNED INFO(1) THE APPROPRI-
C                      ATE VALUE WITH RESPECT TO SUBSEQUENT CALLS.
C            INFO(2)   MAXIMUM NUMBER OF EVALUATIONS OF THE RIGHT-HAND
C                      SIDE OF (1) TO BE SPENT DURING THE INTEGRATION
C                      PROCESS. THIS NUMBER IS THE CONVERTED EQUIVALENT
C                      OF EVALUATIONS ON THE FINEST (I.E. USER-DEFINED)
C                      GRID. TO GET AN IMPRESSION OF THE COSTS OF THE
C                      ALGORITHM WE REFER TO [1], TABLE 3.1 OF SECTION 3.
C            INFO(3)   CAN BE GIVEN THE VALUE 1,2 OR 3. IT IS USED TO
C                      SELECT THE OPTION WITH RESPECT TO THE SPECTRAL
C                      RADIUS OF THE JACOBIAN MATRIX OF THE RIGHT-HAND
C                      SIDE FUNCTION G.
C                      THE THREE OPTIONS AVAILABLE ARE:
C
C                INFO(3)=1 : AUTOMATIC OPTION.
C                            ----------------
C                      WITH THIS OPTION THE CODE ESTIMATES AND CONTROLS
C                      THE SPECTRAL RADIUS DURING THE ENTIRE INTEGRATION
C                      INTERVAL. A FIVE DIAGONAL STRUCTURE OF THE
C                      JACOBIAN MATRIX IS ASSUMED (THIS RESTRICTION
C                      IS ALSO MADE IN THE SEMI-DISCRETIZATION CODE
C                      PDETWO).
C
C                INFO(3)=2 : SEMI-AUTOMATIC OPTION.
C                            ---------------------
C                      IF THE PROBLEM HAS A CONSTANT SPECTRAL RADIUS
C                      (E.G. IN LINEAR PROBLEMS) OR IF IT IS AT LEAST
C                      NON-INCREASING WITH TIME, ONE MAY SELECT THE SEMI-
C                      AUTOMATIC OPTION WHICH ONLY CALCULATES THE
C                      SPECTRAL RADIUS AT THE BEGINNING OF THE INTEGRAT-
C                      ION PROCESS. NO CONTROL IS PERFORMED. IF THIS
C                      SITUATION APPLIES THIS OPTION MAY SAVE A LOT OF
C                      RIGHT-HAND SIDE EVALUATIONS. ONE SHOULD ONLY APPLY
C                      THIS OPTION WHEN THE RIGHT-HAND SIDE FUNCTION G
C                      HAS A FIVE-POINT COUPLING.
C
C                INFO(3)=3 : NON-AUTOMATIC OPTION.
C                            --------------------
C                      SOME PROBLEMS ALLOW THE USER THE CALCULATION
C                      BY HAND OF THE SPECTRAL RADIUS, SO THAT AN
C                      EXPLICIT EXPRESSION IN TERMS OF T, THE MESH SIZES
C                      AT LEVEL K, THE CURRENT SOLUTION V AND THE CUR-
C                      RENT TIME STEP TAU CAN BE GIVEN TO BE USED IN THE
C                      INTERVAL [T,T+TAU]. AGAIN, MANY RIGHT-HAND SIDE
C                      EVALUATIONS CAN BE AVOIDED AND, IF NO USE IS MADE
C                      OF PDETWO, THE RESTRICTION TO FIVE-POINT COUPLING
C                      NO LONGER APPLIES.
C
```

```
C                          IT SHOULD BE NOTED THAT AN UNDERESTIMATE OF THE
C                          SPECTRAL RADIUS MAY GIVE RISE TO INTERNAL
C                          INSTABILITIES IN THE INTEGRATION PROCESS. BECAUSE
C                          THESE INSTABILITIES USE TO DEVELOP APPALLINGLY
C                          FAST WE HAVE PROTECTED THE CODE AGAINST OVERFLOW.
C                          WHATEVER OPTION IS CHOSEN, THE OVERFLOW CHECKING
C                          IS ALWAYS PERFORMED. IN CASE OF OPTIONS 2 AND 3,
C                          THE INTEGRATION PROCESS IS DISCONTINUED WHEN
C                          VIOLATING THE OVERFLOW CONDITION; CONTROL IS
C                          RETURNED TO THE CALLING PROGRAM AND THE USER CAN
C                          TAKE ACTION WITH RESPECT TO THE SPECTRAL RADIUS.
C
C           INFO(4),...,INFO(6)   ARE USED FOR INTERNAL CONTROL
C
C           INFO(7)    TOTAL NUMBER OF INTEGRATION STEPS PERFORMED,
C                      INCLUDING REJECTED ONES.
C           INFO(8)    TOTAL NUMBER OF REJECTED STEPS.
C           INFO(9)    NUMBER OF TIMES A STEP HAS BEEN ABANDONED BECAUSE
C                      OF VIOLATING THE OVERFLOW-TEST.
C           INFO(10)   TOTAL NUMBER OF DERIVATIVE-EVALUATIONS, EXPRESSED
C                      IN TERMS OF EVALUATIONS ON THE ORIGINAL GRID.
C           INFO(10+K) NUMBER OF TIMES THE DERIVATIVE IS EVALUATED AT
C                      LEVEL K (K=1,...,M) USED FOR THE INTEGRATION
C                      PROCESS ONLY.
C           INFO(10+M+K)  NUMBER OF TIMES THE DERIVATIVE IS EVALUATED AT
C                      LEVEL K (K=1,...,M) USED FOR THE EVALUATION AND
C                      CONTROL OF THE SPECTRAL RADIUS.
C
C     IFLAG IS A FLAG USED TO INDICATE VARIOUS ERROR CONDITIONS.
C           ON RETURN, IT MAY HAVE THE FOLLOWING VALUES AND MEANINGS:
C
C           IFLAG=0  SUCCESSFUL INTEGRATION; THAT IS, THE END POINT TEND
C                    HAS BEEN REACHED. TO CONTINUE THE INTEGRATION
C                    IT SUFFICES TO DEFINE A NEW VALUE OF TEND AND
C                    RECALL BDMG.
C           IFLAG=1  THE VALUE OF M IS TOO LARGE IN CONNECTION WITH THE
C                    SPECIFIED VALUES OF NX AND NY.
C           IFLAG=2  X(I) .GE. X(I+1)   OR   Y(J) .GE. Y(J+1)  FOR SOME
C                    I OR J
C           IFLAG=3  THE LENGTH OF THE ARRAY WORK IS LESS THAN THE
C                    REQUIRED MINIMUM.
C           IFLAG=4  TOL .LE. 0  OR  TEND .LE. T
C
C           IN THE LAST FOUR CASES THE PROCESS IS NOT STARTED AND ONE
C           SHOULD RECONSIDER THE INPUT REQUIREMENTS GIVEN AT THE
C           DESCRIPTION OF THE APPROPRIATE PARAMETERS.
C
C           IFLAG=5  THE STEP LENGHT HAS BEEN REDUCED TO AN UNACCEPT-
C                    ABLY SMALL VALUE. THE PROBLEM SEEMS TO BE
C                    UNSOLVABLE TO BDMG. A POSSIBLE REASON MAY BE TOO
C                    STRINGENT ACCURACY REQUIREMENTS (PARAMETER TOL).
C           IFLAG=6  THE MAXIMUM NUMBER OF DERIVATIVE EVALUATIONS
C                    HAS BEEN SPENT; IF THE USER DECIDES TO CONTINUE,
C                    HE ONLY HAS TO INCREASE INFO(2) AND RECALL BDMG.
C           IFLAG=7  THE OVERFLOW-TEST WAS VIOLATED WHILE INFO(3) .NE. 1
C                    THE USER MUST TAKE ACTION WITH RESPECT TO THE
C                    SPECTRAL RADIUS OR CHANGE TO THE AUTOMATIC OPTION.
C
```

```
C-----------------------------------------------------------------------
C
C   BRIEF SUMMARY OF INPUT PARAMETERS
C
C-----------------------------------------------------------------------
C
C    NX,NY,M,X,Y     DEFINE THE GRID AND THE NUMBER OF GRIDS
C    T,TEND          DEFINE THE INTEGRATION INTERVAL
C    V               CONTAINS THE INITIAL VALUE
C    TOL             IS THE LOCAL TOLERANCE PARAMETER
C    METH            SELECTS THE OPTION CONCERNING THE SEMI-DISCRETIZATION
C    IWORK(10*M+3)   SPECIFIES THE LENGTH OF THE ARRAY WORK.
C    INFO(1)         EQUALS ZERO, ONLY AT FIRST CALL
C    INFO(2)         IS THE MAXIMUM NUMBER OF DERIVATIVE-EVALUATIONS
C    INFO(3)         SELECTS THE OPTION CONCERNING THE SPECTRAL RADIUS
C
C    THE PARAMETERS TEND, TOL, INFO(2), AND INFO(3) MAY BE CHANGED FROM
C    CALL TO CALL.
C
C-----------------------------------------------------------------------
C
C   PROGRAMMING NOTES
C
C-----------------------------------------------------------------------
C
C     -   WE EMPHASIZE THAT, USING OPTION 3 WITH RESPECT TO THE
C         SPECTRAL RADIUS (CF. INFO(3)), THE CODE RELIES ON THE USER-
C         SPECIFIED ESTIMATE. THERE IS A CALL TO THE SUBROUTINE SPRAD
C         AT THE START OF EACH INTEGRATION STEP AND THE USER MUST BE
C         SURE TO DELIVER AN ESTIMATE WHICH HOLDS ON THE WHOLE
C         INTERVAL [T,T+TAU].
C     -   CONCERNING THE NUMBER OF ARRAYS REQUIRED BY BDMG, WE NOTE
C         THAT THE CODE NEEDS AT MOST STORAGE EQUIVALENT WITH SEVEN
C         ARRAYS, THE SIZE OF WHICH CORRESPONDS WITH THE NUMBER OF
C         GRID POINTS IN THE ORIGINAL GRID.
C     -   IN SOLVING PARABOLIC EQUATIONS TWO TYPES OF ERRORS ARISE,VIZ.
C         SPACE-DISCRETIZATION ERRORS AND TIME-INTEGRATION ERRORS. WE
C         MENTION THAT ONLY THE LAST ONES ARE CONTROLLED BY BDMG (BY
C         MEANS OF THE PARAMETER TOL). HENCE, IN CHOOSING A VALUE OF
C         TOL THE USER IS ADVISED TO CONSIDER THE ACCURACY OF THE SPACE-
C         DISCRETIZATION.
C     -   THE MINIMUM STEP SIZE DEPENDS ON THE MACHINE ROUNDOFF U AND
C         THE UNDERFLOW NUMBER P, WHICH OBVIOUSLY ARE MACHINE-DEPENDENT.
C         ADAPT, IF NECESSARY, THE DATA STATEMENT BELOW, IN WHICH
C         FOURU=4*U AND TENP=10*P.
C
C-------------------------------------------------------------------
C
C   REFERENCES
C
C---------------------------------------------------------------------
C
C   [1]   HOUWEN,P.J. VAN DER AND B.P.SOMMEIJER, ANALYSIS OF CHEBYSHEV
C              RELAXATION IN MULTIGRID METHODS FOR NONLINEAR PARABOLIC
C              DIFFERENTIAL EQUATIONS, TO APPEAR IN ZAMM, 1983.
C
C   [2]   MELGAARD,D.K. AND R.F.SINCOVEC, GENERAL SOFTWARE FOR TWO-
C              DIMENSIONAL NONLINEAR PARTIAL DIFFERENTIAL EQUATIONS,
C              ACM TRANS. MATH. SOFTW., VOL 7, MARCH 1981, PP. 106-125.
C
C-----------------------------------------------------------------------
-
```