Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

B.P. Sommeijer

Numvec Fortran library manual
Chapter: parabolic PDEs
Routine: BDMG

Department of Numerical Mathematics Report NM-R8610 March

Bibliotheek
Centrum voor Wiskunde en Informatica
Amsterdam

The Centre for Mathematics and Computer Science is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

NUMVEC
is a library of NUMerical software for VECtor and parallel computers in FORTRAN.
The documentation conforms as much as possible to that of the NAG - library. A Subsection: *11.1 Vectorization information* mentions, a.o., whether the code is standard FORTRAN 77. If not, information is given about special vector-syntax used and about specific machine(s) to which the code is aimed.
The source code described can be obtained by writing to the NUMVEC Library Manager at the CWI.

Copyright © Stichting Mathematisch Centrum, Amsterdam

Numvec Fortran Library Manual

Chapter: Parabolic PDEs
Routine: BDMG

B.P. Sommeijer
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

This document describes the NUMVEC FORTRAN Library routine BDMG, which integrates in time a semi-discrete scalar parabolic partial differential equation defined over a two-dimensional rectangular region. The time integration is based on the second-order Backward Differentiation method; the resulting implicit relations are solved by employing a Multi Grid technique.

1980 Mathematics subject classification: 65M20, 65L05, 65V05
1982 CR Categories: 5.17.

Keywords & Phrases: software, parabolic differential equations, multigrid methods, nonlinear Chebyshev iteration.

Note: The implementation is available in auto-vectorizable ANSI FORTRAN 77

Report NM-R8610
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

BDMG - NUMVEC FORTRAN Library Routine Document

1. Purpose

BDMG is a time integrator designed to solve a system of ordinary differential equations (ODEs) originating from semidiscretization of a (scalar) nonlinear parabolic partial differential equation defined over a two-dimensional rectangular region.

2. Specification

```
      SUBROUTINE BDMG  (NX, NY, M, X, Y,
     +                  T, TEND, V, G, SPRAD, TOL, METHOD,
     +                  WORK, IWORK, INFO, IFAIL)
C     INTEGER  NX, NY, M, METHOD, IWORK(*), INFO(*), IFAIL
C     real  X(*), Y(*), T, TEND, V(*), SPRAD, TOL, WORK(*)
```

3. Description

BDMG applies to semidiscrete parabolic equations in two space dimensions, which can be written in the form

$$\frac{dv_{i,j}}{dt} = g_{i,j}(t,V), \qquad V = (v_{i,j}),\ i = 1,\dots,\mathrm{NX},\ j = 1,\dots,\mathrm{NY}, \tag{1}$$

where the (scalar) matrix elements $v_{i,j}$ are associated with the user-defined grid points (x_i,y_j) which form in the (x,y) plane a (not necessarily uniform) grid with rectangular meshes. Thus, any partial differential equation of the form

$$\frac{\partial u}{\partial t} = f(t,\ x,\ y,\ u, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, \frac{\partial^2 u}{\partial x^2}, \frac{\partial^2 u}{\partial x \partial y}, \frac{\partial^2 u}{\partial y^2}),\quad t \geqslant t_0 \tag{2}$$

defined on a rectangle, with initial data defined at $t=t_0$ and prescribed boundary conditions can be dealt with by BDMG as soon as (2) and the boundary conditions are semidiscretized on the grid $\{x_i,y_j\}$. Then, the resulting system (1) is integrated forwards in time using the second-order backward differentiation method; the nonlinear system emanating from this implicit scheme is (approximately) solved by a multigrid technique (cf. references [1] and [5]).

In forming the semidiscrete approximation to (2), there are two options available to the user (see also the description of the parameter METH in Section 5):

(i) the system of ODEs can be provided by a user-supplied subroutine G, defining the right-hand side function g in (1) or

(ii) the package PDETWO (cf. [2] and [3]) can be linked to BDMG, in which case the semi-discretization is automatically performed.

In selecting the last possibility, it suffices to supply routines defining the PDE and the boundary conditions, but on the other hand, its application is restricted to PDEs in which the mixed derivative is absent, resulting in five-point coupled ODEs.

The boundary conditions allowed by PDETWO are of the form

$$A(t,x,y)u + B(t,x,y)\frac{\partial u}{\partial n} = C(t,x,y), \tag{3}$$

n normal to the boundary. In case $B \equiv 0$ (i.e. Dirichlet boundary conditions), A and C may also depend on u. One should be aware of the fact that PDETWO defines the semidiscrete system (1) in *all* gridpoints, including the boundary points. In order to preserve the structure of this ODE system, the user-written subroutine G should also define ODEs at the boundary points. In case of Neumann or mixed (i.e. $B \neq 0$) conditions, the semidiscretization of (3) may be achieved by differentiation with respect to t and forming an ODE (in t) at these boundary points by discretization of $\partial^2 u / \partial n \partial t$. In case of Dirichlet conditions, dummy equations $dv_{i,j}/dt = 0$ should be specified (cf. [2]).

4. References

[1] HOUWEN, P.J. VAN DER and B.P. SOMMEIJER, Analysis of Chebyshev relaxation in multigrid methods for nonlinear parabolic differential equations, *Z. Angew. Math. Mech. 63, Vol. 3,* pp. 193-201, 1983.

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

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

[4] Numerical Algorithms Group, NAG FORTRAN Library manual-mark 11, 1984.

[5] SOMMEIJER, B.P. and P.J. VAN DER HOUWEN, Algorithm 621, Software with low storage requirements for two-dimensional nonlinear, parabolic differential equations, *ACM Trans. Math. Softw., Vol. 10,* pp. 378-396, 1984.

5. Parameters

We distinguish between three types of parameters:

5.1. Grid-defining parameters.

NX, NY - INTEGERS.

On entry, NX and NY must specify the number of mesh lines parallel to the y- and x-axis, respectively, defining a rectangular grid.
Unchanged on exit.

M - INTEGER.

On entry, M defines the number of successive grids, used in the multigrid method; for a discussion on M, see below.
Unchanged on exit.

X - *real* array.

Before entry, the first NX elements of X should contain the (x-) coordinates of the mesh lines parallel to the y-axis, with
X(1) < X(2) < . . . < X(NX).
Unchanged on exit.

Y - *real* array.
Before entry, Y contains similar information concerning the mesh lines parallel to the *x*-axis, with
Y(1) < Y(2) < . . . < Y(NY).
Unchanged on exit.

It is worth to go into more detail about these parameters. Because the method is based on a multigrid method, a sequence of M successive grids is necessary.

The level index K runs from 1 (which is associated with the coarsest grid) until M (the finest grid, which is identical to the grid chosen by the user and defined by means of the parameters NX, NY, X, Y). Apart from the finest grid, these grids are generated by the code. Internal values NXK, NYK are calculated as well as arrays XK(I), I = 1,...,NXK and YK(J), J = 1,...,NYK, defining the grid at level K. These values are used, among other things, to pass to the subroutine G (see below) in which the user has to define the derivative at that particular grid (XK(I), YK(J)) (only if METH = 1, see below).

A grid at level K is obtained from a grid at level K+1 (denoted by KP1) by alternatingly deleting grid lines; hence,

NXK=(NXKP1+1)/2, XK(1)=XKP1(1), XK(2)=XKP1(3), etc.

This means that the grid lines on level K coincide with those on level K+1. Because the values of the internally used arrays XK (for K < M) are stored in the X- array as well, the X- array should be dimensioned in the calling program of length at least $M + (NX-1)(2^M-1)2^{1-M}$. Similarly, the Y- array must be declared of length at least $M+(NY-1)(2^M-1)2^{1-M}$.

The values of NX, NY and M should be chosen in a suitable relationship to each other. 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 an non-integer NXK- or NYK- value for some K < M.
(ii) the grid on level 1 has less than four grid lines in each direction, including the grid lines forming the boundaries.

If none of these restrictions is applicable, the user is advised to choose M as large as possible, because the efficiency of the algorithm usually increases as more grids are available. However, when running in vector mode, a more efficient algorithm does not necessarily imply a better performance of the code (see also Sections 8 and 11).

Finally, we mention that specifying M = 1 is prohibited.

5.2 Time integration parameters.

T - *real.*
T is the current time. On entry, it must contain the initial value of the independent variable *t* in (1).
On exit, T contains the value TEND, unless an error has occurred (see the parameter IFAIL below).

TEND - *real.*
On entry, TEND must specify the final value of T to which the integration is to be carried out with TEND > T.
Unchanged on exit.

V - *real* array of DIMENSION (NX, NY).
V contains the current solution values for all gridpoints (see also the remarks on boundary points in the description of the parameter G). Before entry, V must contain the initial values of the dependent variable V in (1).
On exit, V contains the solution at time T.

G - SUBROUTINE, supplied by the user.
G defines the right-hand side function g of (1) at level K. This routine is only necessary in case of semidiscretization by hand (see the parameter METH below).
Its specification reads:

```
SUBROUTINE G (K, NXK, NYK, XK, YK, T, VK, DVKDT)
INTEGER K, NXK, NYK
real  XK(NXK), YK(NYK), T, VK(NXK, NYK), DVKDT(NXK, NYK)
```

The grid is determined by XK(I), I = 1,...,NXK and YK(J), J = 1,...,NYK, containing the coordinates of the grid at level K. Given the approximations at the current time T in the array VK, this routine must deliver the derivates at all gridpoints in the array DVKDT, including the derivatives at the boundary points. (Here, the subscripts (I, J) in VK and DVKDT refer to the gridpoint with coordinates (XK(I), YK(J))).
In order to preserve the structure of the system of ODEs, irrespective of the type of boundary conditions, the boundary points are included in the grid and, consequently, in the arrays VK and DVKDT. However, in case of Dirichlet boundary conditions the time integration at these points seems to be superfluous. Therefore, the derivatives at these points may be given a dummy value; the value zero is strongly recommended. As a consequence of this approach, the elements in VK corresponding to boundary points where a Dirichlet condition is prescribed, may contain a value which is *not* an approximation to the solution in this point (unless the dummy value specified equals exactly the time-derivative of the Dirichlet condition). Hence, in calculating the derivative in a point adjacent to a "Dirchlet boundary point", one should use the Dirichlet condition, rather than the value of VK in this boundary point. In case of using PDETWO, dummy differential equations $dv_{i,j}/dt=0$ are specified in such "Dirichlet boundary points".
Apart from such points, the routine G must not change any of the values in its first seven parameters.
G should be declared EXTERNAL in the calling program.

SPRAD - *real* FUNCTION supplied by the user.
In case the non-automatic option with respect to the spectral radius is selected (see the parameter INFO below), SPRAD must deliver an estimate of the spectral radius of the Jacobian matrix of the right-hand side function g in (1) at level K.
Its specification reads:

```
FUNCTION SPRAD (K, NXK, NYK, XK, YK, T, VK, TAU)
INTEGER K, NXK, NYK
real  XK(NXK), YK(NYK), VK(NXK, NYK), T, TAU
```

The meaning of the parameters is the same as given in the description of the routine G. The additional parameter TAU denotes the current timestep. SPRAD must not change the value of any of its parameters and should be declared EXTERNAL in the calling program.

TOL - *real.*

On entry, TOL specifies the local truncation error tolerance with TOL >0.
Unchanged on exit.

METH - INTEGER.

Before entry, METH must be given the value 1 or 2.
If METH = 1 the semidiscretization will be performed by hand and the subroutine G must deliver the derivatives required by the integrator.
METH = 2 means that the semidiscretization will be performed by PDETWO.
Unchanged on exit.

5.3 Auxiliary parameters

WORK - *real* array

WORK is used for internal storage. It must be dimensioned as WORK(**) in the calling program, where

$$** \geqslant (NX-1)(NY-1)\,[86-\frac{5}{4^{M-2}}]/\,12\,+$$

$$(NX+NY-2)\,[22-\frac{5}{2^{M-2}}]/\,2\,+$$

$$7M+5\,+(7NX+16)(METH-1).$$

The user should be aware that this expression must yield an integer value. If not, one should carefully reconsider the values of NX, NY and M.

IWORK - INTEGER array.

IWORK is a work array containing pointers to provide dynamic dimensioning in BDMG. It must be dimensioned as IWORK(***) in the calling program, where

$$*** \geqslant 10\,M+5.$$

IWORK(10 M+5) must be initialized with the value of ** (see the parameter WORK).

INFO - INTEGER array.

INFO contains information about the status of the integration process; it must be dimensioned as INFO(****) in the calling program, where

$$**** \geqslant 2\,M+10.$$

INFO (1),...., INFO (3) -	are input parameters and must be initialized by the user
INFO (4),...., INFO (6) -	are used for internal control
INFO (7),...., INFO (2 M+10) -	are output parameters.

The elements of INFO have the following meanings:

INFO(1) = 0

to indicate that this is the first call to BDMG. On return, BDMG has assigned INFO(1) the appropriate value with respect to subsequent calls.

INFO(2)
: maximum number of evaluations of the right-hand side of (1) to be spent during the integration process. This number is the converted equivalent of evaluations on the finest (i.e. user-defined) grid. To get an impression of the costs of the algorithm we refer to [5], Table I of Section 3.

INFO(3)
: can be given the value 1,2 or 3. It is used to select the option with respect to the spectral radius of the Jacobian matrix of the right-hand side function *g*. The three options available are:

  INFO(3) = 1 : *automatic option*
  : with this option the code estimates and controls the spectral radius during the entire integration interval. A five-diagonal structure of the Jacobian matrix is assumed (this restriction is also made in the semidiscretization code PDETWO).

  INFO(3) = 2 : *semi-automatic option*
  : 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 at the beginning of the integration process. No control is performed. If this situation applies, this option may save a lot of right-hand side evaluations. One should only apply this option when the right-hand side function *g* has a five-point coupling.

  INFO(3) = 3 : *non-automatic option*
  : some problems allow the user the calculation by hand of the spectral radius, so that an explicit expression in terms of T, the mesh sizes at level K, the current solution V and the current time step TAU can be given to be used in the interval [T, T+TAU]. Again, many right-hand side evaluations can be avoided and, if no use is made of PDETWO, the restriction to five-point coupling no longer applies. It should be noted that an underestimate of the spectral radius may give rise to internal instabilities in the integration process. Because these instabilities use to develop appallingly fast we have protected the code against overflow. Whatever option is chosen, the overflow checking is always performed. In case of options 2 and 3, the integration process is discontinued when violating the overflow condition; control is returned to the calling program and the user can take action with respect to the spectral radius.

INFO(4) ,...., INFO(6) are used for internal control

INFO(7)
: total number of integration steps performed, including rejected ones.

INFO(8)
: total number of rejected steps.

INFO(9)
: number of times a step has been abandoned because of violating the overflow test.

INFO(10)
: total number of derivative-evaluations, expressed in terms of evaluations on the original grid.

INFO(10+K)
: number of times the derivative is evaluated at level K (K=1,...,M) used for the integration process only.

INFO(10+M+K)
: number of times the derivative is evaluated at level K (K=1,...,M) used for the evaluation and control of the spectral radius.

IFAIL - INTEGER.

For this routine, the normal use of IFAIL is extended to control the printing of error and warning message as well as specifying hard or soft failure (see [4], chapter P01). Before entry, IFAIL must be set to a value with the decimal expansion *cba*, where each of the decimal digits *c*,*b* and *a* must have the value 0 or 1.

$a = 0$	specifies hard failure, otherwise soft failure;
$b = 0$	suppresses error messages, otherwise error messages will be printed (see Section 6);
$c = 0$	suppresses warning messages, otherwise warning messages will be printed (see Section 6).

The recommended value for inexperienced users is 110 (i.e. hard failure with all messages printed). Unless the routine detects an error (see Section 6), IFAIL contains 0 on exit.

6. Error Indicators and Warnings

Errors detected by the routine: -

IFAIL = 1

The value of M is too large in connection with the specified values of NX and NY. This means that for some K, with $1 \leqslant K < M$, it is not possible to construct on this K-th level a coarse grid having an *integer* number of grid lines.

IFAIL = 2

There is an error in the specification of the (finest) mesh: $X(I) \geqslant X(I+1)$ or $Y(J) \geqslant Y(J+1)$ for some I or J.

IFAIL = 3

The length of the array WORK is less than the required minimum.

IFAIL = 4

$TOL \leqslant 0$ or $TEND \leqslant T$ or $M = 1$

If IFAIL is assigned one of these values, the integration process is not started and one should reconsider the input requirements given in the description of the appropriate parameters.

The following errors are of algorithmic nature:

IFAIL = 5

The steplength has been reduced to an unacceptably small value. The problem seems unsolvable to BDMG. A possible reason may be too stringent accuracy requirements (parameter TOL).

IFAIL = 6

The maximally allowed number of derivative evaluations has been spent; if the user decides to continue, he only has to increase INFO(2) and recall BDMG.

IFAIL = 7

The overflow test was violated while $INFO(3) \neq 1$. The user must take action with respect to the (estimation of the) spectral radius of the Jacobian matrix or change to the automatic option.

7. Auxiliary routines

This routine calls the NUMVEC Library routines:
CHEB, DERIV, FIT, GERSG, INJECT, LTE, NEWST, PDETWO, PRELIM, PROLON, PROLNG, PROLNU, P01AAF, RESTR, RESTRG, RESTRU, RKR, SHIFT, SPRI, STEP, STRSET, SUMG, TAUST, X04AAF, X04ABF.

8. Timing

Evidently, the timing strongly depends on the complexity of the parabolic equation and on the accuracy requested. Due to the nature of the algorithm, a substantial part of the overall time is used to evaluate the derivative, i.e. the right-hand side function g in (1). Hence, in selecting METH = 1, the timing critically depends on the coding of the subroutine G (see also Section 11). Obviously, the timing is proportional to NX*NY; moreover, for a fixed value of M (i.e. a fixed number of grids), the number of times the derivative has to be evaluated usually increases as NX and/or NY increases. Finally, as the algorithm is based on a low-order method, a request for high accuracy may result in a very lengthy calculation.

9. Storage

There are no internally declared arrays.

10. Accuracy

In solving parabolic equations two types of errors arise, viz. space-discretization errors and time-integration errors. We emphasize that only the last ones are 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.
Furthermore, in the time-integration only the *local* errors are controlled and it cannot be guaranteed that the global error after a large number of steps will remain small, although a smaller value of TOL usually results in a more accurate global solution. The user in advised to test the effect of TOL by repeating the calculation with a different value for this parameter. As only a second-order method is used, BDMG is not suitable to produce highly accurate results.
Concerning the space-discretization, it is recommended to perform some trial computations on different grids. For this test, the parameter TOL should sufficiently be reduced so that time errors do not interfere unduly with the errors in space.

11. Further Comments

11.1 General Information

The minimum stepsize depends on the machine roundoff U and the underflow number P, which obviously are machine-dependent. Adapt, if necessary, the DATA statement in the driver BDMG, in which FOURU = 4 * U and TENP = 10 * P. The routine uses the labelled COMMON blocks: COEF, MACH, MESH, ISTORE and OFLOW; therefore, the user must avoid these names for his own purposes.

We emphasize that the routine is designed to treat parabolic equations which give rise to Jacobian matrices with a (more or less) *real-valued* spectrum; this means that an efficient application of

BDMG is restricted to parabolic equations in which the diffusion part greatly dominates the convection part. If the user decides to select option 3 with respect to the spectral radius (see also the parameter INFO(3)), the code relies on the user-specified estimate. The FUNCTION SPRAD is called at the start of each integration step and the user must be sure to deliver an estimate which holds on the whole interval [T, T+TAU].

If, on exit, IFAIL = 0 the integration has been successful and TEND is reached. To continue the process it suffices to define a new value of TEND and recall BDMG. Moreover, the parameters TOL, INFO(2) and INFO(3) may be changed from call to call.

In the stepsize control mechanism, as it is implemented in BDMG, we require at each steppoint the remaining integration interval to be an integer multiple of the current stepsize. In this way we achieve that the process arrives exactly at the prescribed endpoint TEND. Hence, interpolating the numerical solution at this point can be avoided which, in our experience, may be rather inaccurate. As a consequence of this strategy, runs with the same final endpoint but with different intermediate endpoints (output points) may produce slightly different results.

11.2 Vectorization Information

The *explicit* nature of the underlying algorithm enables us to achieve a high degree of vectorization; therefore, the routine has been written in ANSI FORTRAN 77. To guarantee autovectorization on the CYBER 205, provisions are included with respect to the maximal loop count, i.e. 65535.

In the relaxation routine CHEB, which is the most costly part of the algorithm, the majority of the loops has been vectorized; this is performed by employing over-indexing in order to collapse nested DO-loops. Furthermore, the routine is provided with a test on the user-defined spatial grid to detect whether or not it is *uniform*. Using this feature, a significant gain can be obtained in the routines which perform the restriction and the prolongation (that is, the transformation of grid-functions from one grid into another). Hence, if the problem allows to be approximated on a uniform grid, the user is strongly recommended to do so.

Concerning the choice of the parameter M (i.e. the number of successive spatial grids involved) and its influence on vectorization aspects, we have the following 'rule of thumb':

(i) in scalar mode, the value of M should be specified as large as possible (see also the discussion following the grid-defining parameters in Section 5.1), because the efficiency of both the algorithm as well as of the code usually increases as M increases.

(ii) in vector mode, however, the startup times may prevent optimal performance when the vector length is very short; this particularly applies to the CYBER 205. Therefore, using the coarsest grid which is allowed by BDMG (i.e. a 4 * 4 - grid) is not recommended to obtain optimal vector performance.

Finally, in case METH = 1 is specified, that is the subroutine G has to be supplied defining the semidiscretization, the user may decide to write this subroutine making use of explicit vector-syntax; this, of course, only applies when running on a CYBER 205.

In case the user selects METH = 2 — that is the semidiscretization will be performed by PDETWO — it should be remarked that the overall vector performance will be reduced drastically. This is due to the fact that PDETWO evaluates the semidiscrete approximation to the PDE by a call to the problem-defining routines for each gridpoint separately (see also the supplementary example in Section 13.4). Moreover, as most of the loops in PDETWO run from 1 to the number of PDEs (which equals one in our application) it is recommended to run PDETWO in scalar mode.

12. Keywords

Parabolic Partial Differential Equations, Method of Lines, Backward Differentiation Methods, Multigrid Methods, Nonlinear Chebyshev Iteration.

13. Example

The following example program is intended to illustrate some of the features of BDMG. It solves a so-called 'porous-medium' equation given by

$$\frac{\partial u}{\partial t} = \left[\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right] u^5, \quad 0 \leqslant t \leqslant 1, \tag{4}$$

defined on the unit square. At the 'upper' boundary (i.e. for $y=1$, $0<x<1$) we impose a Neumann condition by prescribing $\partial u/\partial y$. On the other boundaries we have a Dirichlet condition. All boundary conditions, as well as the initial conditions are taken from the exact solution

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

To discretize (4), we choose uniformly spaced square meshes with mesh size 1/96; hence, NX = NY = 97. The local error tolerance parameter was set to TOL $=10^{-4}$. By this choice, the overall error at $t=1$ turns out to be largely determined by the time-integration error and not by the space-discretization error. Finally, we use 5 successive grids.

13.1 Program text

```
      PROGRAM EXAM1
C-----------------------------------------------------------------------
C     EXAMPLE PROGRAM FOR BDMG
C-----------------------------------------------------------------------
      DIMENSION V(97,97), X(191), Y(191), INFO(20),
     +          WORK(68080), IWORK(55)
      COMMON /TIMINGS/ CPG, CPSPRAD, CPPDE2
      COMMON /AUXIL/ VK5(9409)
      DATA NX, NY, M, METH /97, 97, 5, 1/
      DATA CPG, CPSPRAD, CPPDE2 /0.0, 0.0, 0.0/
      DATA NOUT /6/
      EXTERNAL G, SPRAD
C-----------------------------------------------------------------------
      OPEN (UNIT = NOUT, FILE = 'OUTPUT')
      CALL X04AAF (1, NOUT)
      CALL X04ABF (1, NOUT)
      WRITE (NOUT, 99991)
C-----------------------------------------------------------------------
C     DEFINITION OF THE (FINEST) GRID
C-----------------------------------------------------------------------
      DELTAX = 1.0 / (NX-1.0)
      DO 10 I = 1, NX
   10   X(I) = (I-1) * DELTAX
```

```
      DELTAY = 1.0 / (NY-1.0)
      DO 20 I = 1, NY
   20   Y(I) = (I-1) * DELTAY
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
        V(I,J) = (0.8 * (X(I) + Y(J)) ) ** 0.25
   30   CONTINUE
C-----------------------------------------------------------------------
C     DEFINITION OF THE INTEGRATION PARAMETERS
C-----------------------------------------------------------------------
      TOL = 1.0E-4
      INFO(1) = 0
      INFO(2) = 2000
      INFO(3) = 1
      IWORK(10 * M + 5) = 68080
      IFAIL = 111
C-----------------------------------------------------------------------
C     CALL FOR THE INTEGRATOR
C-----------------------------------------------------------------------
      CPB = SECOND( )
      CALL BDMG (NX, NY, M, X, Y,
     +          T, TEND, V, G, SPRAD, TOL, METH,
     +          WORK, IWORK, INFO, IFAIL)
      CPE = SECOND( )
C-----------------------------------------------------------------------
C     CHECK IFAIL ON RETURN
C-----------------------------------------------------------------------
      IF (IFAIL .GT. 0) THEN
        WRITE (NOUT, 99992) IFAIL, T
        STOP
      ENDIF
C-----------------------------------------------------------------------
C     OUTPUT SOME STATISTICS
C-----------------------------------------------------------------------
      WRITE (NOUT, 99993) INFO(7), INFO(8), INFO(10)
      WRITE (NOUT, 99994) CPE - CPB, CPG, CPSPRAD, CPPDE2
C-----------------------------------------------------------------------
C     MEASURING THE MAXIMUM NORM OF THE ERROR
C-----------------------------------------------------------------------
      ERROR = 0.0
      DO 40 J = 2, NY
        DO 40 I = 2, NX-1
        ERROR = AMAX1(ABS(V(I,J) - (0.8 * (2*T + X(I) + Y(J))) ** 0.25), ERROR)
   40 CONTINUE
```

```
      WRITE (NOUT, 99995) T, ERROR
C-----------------------------------------------------------------------
99991 FORMAT (29H1BDMG EXAMPLE PROGRAM RESULTS,//)
99992 FORMAT (24H BDMG FAILS WITH IFAIL = , I3, 7H AT T =, F10.5)
99993 FORMAT (28H NUMBER OF STEPS PERFORMED :, 4X, I6/,
     +        28H NUMBER OF REJECTED STEPS  :, 4X, I6/,
     +        32H TOTAL NUMBER OF G-EVALUATIONS :, I6/,
     +        32H (INCLUDING THE ONES TO CONTROL /,
     +        35H THE SPECTRAL RADIUS AND CONVERTED /,
     +        31H TO THE FINEST GRID EQUIVALENT)//)
99994 FORMAT (31H TOTAL CP SECONDS USED BY BDMG:, 6X, F8.2 /,
     +        35H (INCLUDING THE SEMIDISCRETIZATION)//,
     +        37H CP SECONDS USED BY THE DERIVATIVE G:, F8.2/,
     +        27H CP SECONDS USED BY SPRAD :, 10X, F8.2/,
     +        27H CP SECONDS USED BY PDETWO:, 10X, F8.2///)
99995 FORMAT (8H AT T = , F10.5,
     +        39H  THE MAXIMUM NORM OF THE ERROR EQUALS:, E12.5)
C-----------------------------------------------------------------------
C     END OF EXAMPLE PROGRAM
C-----------------------------------------------------------------------
      STOP
      END

      FUNCTION SPRAD (K, NXK, NYK, XK, YK, T, VK, TAU)
      DIMENSION XK(NXK), YK(NYK), VK(NXK, NYK)
      COMMON /TIMINGS/ CPG, CPSPRAD, CPPDE2
      CPB = SECOND( )
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)
      CPSPRAD = CPSPRAD + SECOND( ) - CPB
C-----------------------------------------------------------------------
C     BECAUSE WE SELECTED THE AUTOMATIC OPTION WITH RESPECT
C     TO THE SPECTRAL RADIUS (INFO(3) = 1), SPRAD WILL NOT
C     BE CALLED BY BDMG; HENCE, A DUMMY FUNCTION MAY BE GIVEN
C-----------------------------------------------------------------------
      RETURN
      END

      SUBROUTINE G (K, NXK, NYK, XK, YK, T, VK, DVKDT)
      DIMENSION XK(NXK), YK(NYK), VK(NXK,NYK), DVKDT(NXK,NYK)
C-----------------------------------------------------------------------
C     DEFINE THE DERIVATIVE FUNCTION G AT LEVEL K
C-----------------------------------------------------------------------
      COMMON /AUXIL/ VK5(9409)
      COMMON /TIMINGS/ CPG, CPSPRAD, CPPDE2
      CPB = SECOND( )
      TS = T * 1.6
      H = 1.0 / (NXK-1.0)
      H2INV = (NXK-1) ** 2
C-------------------------------------------------------------------------------
C     TREATMENT OF THE DIRICHLET BOUNDARY POINTS
C-------------------------------------------------------------------------------
      DO 10 I = 1, NXK
        DVKDT(I,1) = 0.0
        VK(I,1) = ( TS + 0.8 * XK(I)) ** 0.25
   10   CONTINUE
      DO 30 J = 2, NYK
       VK(1,J) = ( TS + 0.8 * YK(J)) ** 0.25
   30  CONTINUE
      DO 40 J = 2, NYK
        VK(NXK,J) = ( TS + 0.8 * YK(J) + 0.8) ** 0.25
   40   CONTINUE
C-------------------------------------------------------------------------------
C     DERIVATIVES AT INTERNAL POINTS
C-------------------------------------------------------------------------------
      DO 50 L = 1, NXK * NYK
        VK5(L) = VK(L,1) ** 5
   50   CONTINUE
      DO 60 L = NXK + 1, (NYK-1) * NXK
        DVKDT(L,1) = (VK5(L-NXK) + VK5(L-1)- 4.0 * VK5(L) +
     +             VK5(L+1) + VK5(L+NXK) ) * H2INV
   60   CONTINUE
C-------------------------------------------------------------------------------
C     CORRECTION AT THE LEFT AND RIGHT BOUNDARY
C-------------------------------------------------------------------------------
      DO 70 J = 2, NYK
        DVKDT(1,J) = 0.0
   70   CONTINUE
      DO 80 J = 2, NYK
        DVKDT(NXK,J) = 0.0
   80   CONTINUE
C-------------------------------------------------------------------------------
C     DERIVATIVES AT THE UPPER BOUNDARY, WHERE WE HAVE A NEUMANN CONDITION
C-------------------------------------------------------------------------------
      L = (NYK-1) * NXK
      DO 90 I = 2, NXK-1
        DVKDT(I,NYK) = (VK5(L+I-1) - 2.0 * VK5(L+I) + VK5(L+I+1) +
     +             H * 2.0 * (TS + 0.8 * XK(I) + 0.8) ** (-0.75) *
     +             VK(I,NYK) ** 4 -
     +             0.625 * (VK(I,NYK) + VK(I,NYK-1)) ** 4 *
     +             (VK(I,NYK) - VK(I,NYK-1)) ) * H2INV
   90   CONTINUE
      CPG = CPG +SECOND( ) - CPB
      RETURN
      END
```

13.2 Program data

None

13.3 Program results

```
BDMG EXAMPLE PROGRAM RESULTS


NUMBER OF STEPS PERFORMED :        93
NUMBER OF REJECTED STEPS  :         1
TOTAL NUMBER OF G-EVALUATIONS :   996
(INCLUDING THE ONES TO CONTROL
THE SPECTRAL RADIUS AND CONVERTED
TO THE FINEST GRID EQUIVALENT)


TOTAL CP SECONDS USED BY BDMG:          21.89
(INCLUDING THE SEMIDISCRETIZATION)

CP SECONDS USED BY THE DERIVATIVE G:     5.50
CP SECONDS USED BY SPRAD :               0.00
CP SECONDS USED BY PDETWO:               0.00


AT T =     1.00000  THE MAXIMUM NORM OF THE ERROR EQUALS: 0.15418E-04
```

13.4 Supplementary Example

Next, we illustrate the use of the interface PDETWO. To that end, we solve the same problem (4) as described in the previous section. Therefore, the mainprogram as listed in Section 13.1 can be used, with the following modifications:

- the first DATA statement should read DATA NX, NY, M, METH / 97, 97, 5, 2 /
- The length of the working array WORK should be increased to 68775. Hence, the dimension statement and the initialization of IWORK(10 M+5) should be adapted accordingly.
- The SUBROUTINE G is no longer needed, and may be specified as a dummy routine (to satisfy the loader).

In order to fit this problem into the class of PDEs that PDETWO is capable of solving, we write (4) in the form

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}(5u^4\frac{\partial u}{\partial x}) + \frac{\partial}{\partial y}(5u^4\frac{\partial u}{\partial y}). \tag{4'}$$

Now, the following five SUBROUTINES should be supplied (c.f. [2,3]):

```
      SUBROUTINE BNDRYH (T, X, Y, U, AH, BH, CH, NPDE)
      DIMENSION U(NPDE), AH(NPDE), BH(NPDE), CH(NPDE)
C-----------------------------------------------------------------------
C     THIS ROUTINE DEFINES THE BOUNDARY CONDITIONS ALONG HORIZONTAL
C     BOUNDARIES. THESE CONDITIONS ARE OF THE FORM
C
C       AH(L) U(L)  +  BH(L) D/DY U(L)  =  CH(L),   L=1,...,NPDE,
C
C     WHERE NPDE IS THE NUMBER OF PDES (HERE, NPDE=1).
C-----------------------------------------------------------------------
      IF ( Y .EQ. 0.0 ) THEN
        AH(1) = 1.0
        BH(1) = 0.0
        CH(1) = (0.8 * (2*T + X) ) ** 0.25
      ELSE
        AH(1) = 0.0
        BH(1) = 1.0
        CH(1) = 0.2 * (0.8 * (2*T + X + 1) ) ** (-0.75)
      ENDIF
      RETURN
      END

      SUBROUTINE BNDRYV (T, X, Y, U, AV, BV, CV, NPDE)
      DIMENSION U(NPDE), AV(NPDE), BV(NPDE), CV(NPDE)
C-----------------------------------------------------------------------
C     THIS ROUTINE DEFINES THE BOUNDARY CONDITIONS ALONG VERTICAL
C     BOUNDARIES. THESE CONDITIONS ARE OF THE FORM
C
C       AV(L) U(L)  +  BV(L) D/DX U(L)  =  CV(L),   L=1,...,NPDE,
C
C     WHERE NPDE IS THE NUMBER OF PDES (HERE NPDE=1).
C-----------------------------------------------------------------------
      AV(1) = 1.0
      BV(1) = 0.0
      CV(1) = (0.8 * (2*T + X + Y) ) ** 0.25
      RETURN
      END

      SUBROUTINE DIFFH (T, X, Y, U, DH, NPDE)
      DIMENSION U(NPDE), DH(NPDE,NPDE)
C-----------------------------------------------------------------------
C     THIS ROUTINE DEFINES THE DIFFUSION COEFFICIENT IN HORIZONTAL
C     DIRECTION (I.E. IN X-DIRECTION).
C     NPDE, THE NUMBER OF PDES, EQUALS 1.
C-----------------------------------------------------------------------
      DH(1,1) = 5.0 * U(1) ** 4
      RETURN
      END
      SUBROUTINE DIFFV (T, X, Y, U, DV, NPDE)
      DIMENSION U(NPDE), DV(NPDE,NPDE)
C-----------------------------------------------------------------------
C     THIS ROUTINE DEFINES THE DIFFUSION COEFFICIENT IN VERTICAL
C     DIRECTION (I.E. IN Y-DIRECTION).
C     NPDE, THE NUMBER OF PDES, EQUALS 1.
C-----------------------------------------------------------------------
      DV(1,1) = 5.0 * U(1) ** 4
      RETURN
      END

      SUBROUTINE F (T, X, Y, U, UX, UY, DUXX, DUYY, DUDT, NPDE)
      DIMENSION U(NPDE), UX(NPDE), UY(NPDE), DUXX(NPDE,NPDE),
     +          DUYY(NPDE,NPDE), DUDT(NPDE)
C-----------------------------------------------------------------------
C     IN DUDT(L), L=1,...,NPDE, THE RIGHT-HAND SIDE OF THE PDES
C     IS DEFINED IN TERMS OF THE FIRST 8 PARAMETERS.
C     NPDE, THE NUMBER OF PDES, EQUALS 1.
C-----------------------------------------------------------------------
      DUDT(1) = DUXX(1,1) + DUYY(1,1)
      RETURN
      END
```

13.5 Program data

None

13.6 Program results

```
BDMG EXAMPLE PROGRAM RESULTS


NUMBER OF STEPS PERFORMED :          93
NUMBER OF REJECTED STEPS  :           1
TOTAL NUMBER OF G-EVALUATIONS :     996
(INCLUDING THE ONES TO CONTROL
THE SPECTRAL RADIUS AND CONVERTED
TO THE FINEST GRID EQUIVALENT)


TOTAL CP SECONDS USED BY BDMG:         228.08
(INCLUDING THE SEMIDISCRETIZATION)

CP SECONDS USED BY THE DERIVATIVE G:     0.00
CP SECONDS USED BY SPRAD :               0.00
CP SECONDS USED BY PDETWO:             211.39


AT T =     1.00000  THE MAXIMUM NORM OF THE ERROR EQUALS: 0.15640E-04
```