stichting
mathematisch
centrum

MC

AFDELING NUMERIEKE WISKUNDE
(DEPARTMENT OF NUMERICAL MATHEMATICS)

NW 52/77 DECEMBER

M. BAKKER

SOFTWARE FOR SEMI-DISCRETIZATION OF TIME-DEPENDENT PARTIAL DIFFERENTIAL EQUATIONS IN ONE SPACE VARIABLE

Preprint

2e boerhaavestraat 49 amsterdam

Printed at the Mathematical Centre, 49, 2e Boerhaavestraat, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O).

AMS(MOS) subject classification scheme (1970): 65L05

Software for semi-discretization of time-dependent partial differential equations in one space variable *)

by

M. Bakker

ABSTRACT

In this paper two FORTRAN subroutines are discussed which semi-discretize second order parabolic partial differential equations in one space variable to ordinary differential equations. Either of them may serve as an interface between the PDE and an ODE integrator.

All the user has to do is

1) the definition of a spatial grid,
2) the initialization of the function values and
3) the implementation of the three equations which determine the PDE.

In this way he is saved the time-consuming work of deriving an adequate ODE himself. Both subroutines have been tested on their portability. By means of five numerical examples their possibilites are illustrated.

KEY WORDS & PHRASES: *partial differential equations, finite element method, parabolic equations, ordinary differential equations*

---

*) This report will be submitted for publication elsewhere

1. INTRODUCTION

Software for solving partial differential equations (PDEs) exists already for years, albeit not in such an extent as the software for integrating ODEs. Examples are CARVER [4], CSENDERS [5], GARY & HELGASON [7,8], NILSEN & KARPLUS [12], POLAK et al. [14], SINCOVEC & MADSEN [17,18]. As far as parabolic PDEs are concerned, there exist two categories of software;

1) subroutines which discretize the PDE in the space variables and integrate the resulting ODE. An example is TEDDY 2 by POLAK et al. [14].
2) Subroutines which only discretize in the space variable and leave the integration of the ODE to one of the numerous robust integrators (see e.g. HINDMARSCH [12], VERWER [22]). An example is the subroutine PDEONE by SINCOVEC et al. [17]. Such a subroutine serves, in a way, as an interface between the PDE and the ODE integrator.

It now appears that, as far as I know, only the finite difference method was used to generate an interface subroutine belonging to group 2. This fact may be connected with the wide-spread belief that the finite element method generates implicit ODEs. Strictly spoken, this is true. However, by committing a variational crime (the term was first used by STRANG [19]), called *lumping*, one can remove this inconvenience. That crime consists of applying proper quadrature rules, even where exact integration is possible (see RAVIART [15], DOUGLAS & DUPONT [6], HEMKER [11], BAKKER [1,2]). Although the use of lumping is already known in the case of Cartesian coordinates, it will also be applied when the spatial coordinates are circular or spherical (see BAKKER [2]). So far, the first justification for the use of the finite element method. The second is that the F.E.M. permits a more attractive way of processing the boundary conditions.

We have striven towards a subroutine which semi-discretizes a class of second order PDEs in one space variable, which is as broad as possible. Furthermore we wanted to manufacture a subroutine in which the user need not insert self-made statements and which uses no subprograms not appearing in the parameter list; in short, a subroutine which can served as a substantive library routine and whose body need not be known to the user.

The result was the ANSI FORTRAN IV subroutines PDEF1 and PDEF2 which have been checked on their portability by the PFORT verifier by RYDER [16].

In §2, we describe the classes of PDEs which the subroutine cover.
In §3, we discuss the methods on which the two subroutines are based.
In §4, we discuss the two subroutines and their use together with an ODE integrator.
In §5, we give five numerical examples which are representative for the class of PDEs covered.
In §6, we make some concluding remarks.

2. STATEMENT OF THE PROBLEM

The class of PDEs which the subroutine PDEF1 covers is of the form

$$\frac{\partial}{\partial t} u_i = x^{-NC} \frac{\partial}{\partial x} (x^{NC} F_i(x,t,\vec{U},\vec{U}_x)) + G_i(x,t,\vec{U},\vec{U}_x), \quad a \le x \le b;$$

$$\alpha_i U_i(x,t) + \beta_i \frac{\partial U_i}{\partial x}(x,t) = \gamma_i(t,\vec{U}), \quad x = a,b; \tag{2.1}$$

$$u_i(x,0) = v_i(x), \quad i = 1,\ldots,NPDE,$$

where NC denotes the kind of spatial coordinates used (Cartesian if NC = 0, circular if NC = 1 and spherical if NC = 2) and NPDE denotes the number of partial differential equations involved. Furthermore, $\vec{U}$ and $\vec{U}_x$ are NPDE-dimensional vectors with components U_i and $\partial U_i/\partial x$, respectively.

A few mild requierements have to be satisfied with respect to the semi-discretizability of (2.1): α_i and β_i are not allowed to vanish both and the functions F_i, G_i and v_i have to be continuous in all their variables, except for some known values of x. If these requirements are fulfilled, (2.1) is semi-discretizable. Note, however, that this implies by no means that a solution of (2.1) always exists.

The class of PDEs covered by PDEF2 is also given by (2.1) with the limitation that NC = 0.

3. SEMI-DISCRETIZATION BY MEANS OF THE FINITE ELEMENT METHOD

Let

$$\Delta: a = x_1 < x_2 < \ldots < x_{NPTS} = b$$

be a grid of NPTS points, not necessarily equidistant. In order to approximate $\frac{\partial}{\partial t} U_i(x,t)$ at the grid-points, we approximate $U_i(x,t)$ by a function $U_i(x,t)$ represented by

$$U_i(x,t) = \sum_{j=1}^{NPTS} U_{i,j}\, \varphi_j(x),$$

where $\varphi_j(x)$ are properly chosen functions with the property

$$\varphi_j(x_\ell) = \delta_{j,\ell},$$

$\delta_{j,\ell}$ the Kronecker symbol. If we use for the NPDE-dimensional vectors with components $U_i(x,t)$, $\frac{\partial U_i}{\partial x}(x,t)$ and $U_{i,j}$ the notations $\vec{U}$, $\vec{U}_x$ and $\vec{U}_j$, respectively, we find that

$$\int_a^b x^{NC} \frac{\partial U_i}{\partial t}(x,t)\varphi_j(x)dx = [x^{NC} F_i(x,t,\vec{U},\vec{U}_x)\varphi_j(x)]_a^b -$$

$$- \int_a^b x^{NC}[F_i(x,t,\vec{U},\vec{U}_x)\varphi_j'(x) - G_i(x,t,\vec{U},\vec{U}_x)\varphi_j(x)]dx;$$

$$i = 1,\ldots,NPDE; \quad j = 1,\ldots,NPTS.$$

Note that the stock-term of the right hand side vanishes if $j > 1$ or $j < NPTS$.

Elaboration of (3.34) gives the (implicit) ODE

$$\sum_{\ell=1}^{NPTS} m_{j,\ell} \frac{d}{dt} U_{i,\ell} = N_{i,j}(t,\vec{U}_1,\ldots,\vec{U}_{NPTS}),$$

$$i = 1,\ldots,NPDE; \quad j = 1,\ldots,NPTS,$$

where $m_{j,\ell}$ is defined by

$$m_{j,\ell} = \int_a^b x^{NC} \varphi_j(x)\varphi_\ell(x)dx, \quad 1 \le j, \ell \le NPTS$$

and where $N_{i,j}$ is the right hand side of (3.3).

In the following sections of this § we introduce two choices of $\varphi_j(x)$ which yield easily implementable explicit ODEs with sparse Jacobians.

3.1 Piecewise linear functions

These functions (see fig.1) are defined by the properties

1) $\varphi_j(x)$ is continuous on [A,B] and linear on each segment $[x_\ell, x_{\ell+1}]$, $j = 1,\ldots,NPTS$; $\ell = 1,\ldots,NPTS-1$.
2) $\varphi_j(x_\ell) = \delta_{j,\ell}$, $1 \le j, \ell \le NPTS$.

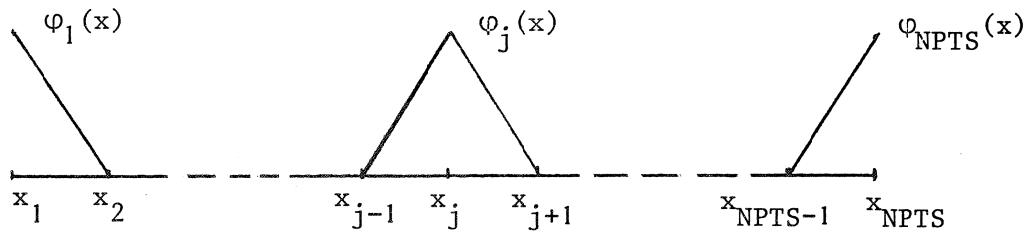

Figure 1. Piecewise linear functions

In order to obtain an easily implementable semi-discretization of (3.1) with piecewise linear functions, we apply a proper quadrature rule to evaluate (3.3a).

3.1.1 Numerical quadrature

Let λ_k and μ_k be given by the formulas

$$\lambda_k = \int_{x_k}^{x_{k+1}} x^{NC} \varphi_k(x)dx;$$

(3.9)

$$\mu_k = \int_{x_k}^{x_{k+1}} x^{NC} \varphi_{k+1}(x)dx; \quad k = 1,\ldots,NPTS-1.$$

and let

$$d_k = \begin{cases} \lambda_1 & , \ k = 1; \\ \lambda_k+\mu_{k-1}, & k = 2,\ldots,NPTS-1; \\ \mu_{NPTS-1}, & k = NPTS; \end{cases}$$

$$w_k = \lambda_k+\mu_k \quad , \ k = 1,\ldots,NPTS-1;$$

(3.10)

$$\xi_k = \frac{\lambda_k x_k+\mu_k x_{k+1}}{w_k} ; \quad k = 1,\ldots,NPTS-1;$$

$$\vec{Y}_k = \frac{\lambda_k \vec{U}_k+\mu_k \vec{U}_{k+1}}{w_k} ; \quad k = 1,\ldots,NPTS-1;$$

$$\vec{Z}_k = \frac{\vec{U}_{k+1}-\vec{U}_k}{h_k} ; \quad h_k = x_{k+1} - x_k, \quad k = 1,\ldots,NPTS-1.$$

Then we approximate $(m_{j,\ell})$ and $(N_{i,j})$ by means of the following formulas

(3.11) $$m_{j,\ell} \doteq d_\ell \, \delta_{j,\ell}; \quad 1 \le j, \ell \le NPTS;$$

(3.12) $$N_{i,1} \doteq - x_1^{NC} F_i(x_1,t,\vec{U}_1, \frac{\partial}{\partial x} \vec{U}_1) + \frac{w_1}{h_1} F_i(\xi_1,t,\vec{Y}_1,\vec{Z}_1) + d_1 G_i(\xi_1,t,\vec{Y}_1,\vec{Z}_1);$$

(3.13) $$N_{i,j} \doteq - \frac{w_{j-1}}{h_{j-1}} F_i(\xi_{j-1},t,\vec{Y}_{j-1},\vec{Z}_{j-1}) + \mu_{j-1} G_i(\xi_{j-1},t,\vec{Y}_{j-1},\vec{Z}_{j-1}) + \frac{w_j}{h_j} F_i(\xi_j,t,\vec{Y}_j,\vec{Z}_j) + \lambda_j G_i(\xi_j,t,\vec{Y}_j,\vec{Z}_j), \quad j = 2,\ldots,NPTS-1;$$

$$(3.14)\qquad N_{i,NPTS} \doteq x_{NPTS}^{NC} F_i(x_{NPTS},t,\vec{U}_{NPTS},\frac{\partial}{\partial x}\vec{U}_{NPTS})$$

$$- \frac{w_{NPTS-1}}{h_{NPTS-1}} F_i(\xi_{NPTS-1},t,\vec{Y}_{NPTS-1},\vec{Z}_{NPTS-1})$$

$$+ d_{NPTS-1}\, G_i(\xi_{NPTS-1},t,\vec{Y}_{NPTS-1},\vec{Z}_{NPTS-1}).$$

In (3.11) we use a generalization of the extended trapezoid rule, which results in the approximation of $(m_{j,\ell})$ by a *diagonal* matrix. In (3.12)-(3.14) we use a generalization of the extended mid-point rule. The advantage of this rule is that the functions F_i and G_i are to be evaluated only NPTS-1 times. We refer to BAKKER [2] for the theoretical background of these two generalizations and remark only that the accuracy of the semi-discretization remains unaltered (of $O(h^2)$, where $h = \max h_j$).

3.1.2 Boundary conditions

There are two kinds of boundary conditions, viz. of Dirichlet type ($\beta_i = 0$) and of mixed type ($\beta_i \neq 0$). The implementation of these conditions in (3.12) and (3.14) is simply done by putting

$$(3.15)\qquad \frac{\partial}{\partial x} U_{i,1} \doteq \begin{cases} (U_{i,2}-U_{i,1})/h_1, & \beta_i = 0; \\ (\gamma_i-\alpha_i U_{i,1})/\beta_i, & \beta_i \neq 0; \end{cases}$$

$$\frac{\partial}{\partial x} U_{i,NPTS} \doteq \begin{cases} (U_{i,NPTS}-U_{i,NPTS-1})/h_{NPTS-1}, & \beta_i = 0; \\ (\gamma_i-\alpha_i U_{i,NPTS})/\beta_i, & \beta_i \neq 0. \end{cases}$$

3.1.3 The resulting ODE

Summarizing, we can approximate $\partial u_i(x_j)/\partial t$ by

$$\frac{d}{dt} U_{i,j} = \begin{cases} 0, \ \beta_i = 0, \ \ j = 1 \text{ or } j = NPTS; \\ \dfrac{N_{i,j}}{d_j} \quad \text{otherwise,} \end{cases}$$

where $N_{i,j}$ and d_j are given by (3.12)-(3.15) and (3.10) respectively.

In §4 we discuss the subroutine PDEF1 in which this algorithm has been implemented.

3.2. Piecewise quadratic functions

If NC = 0, it is possible to take piecewise quadratics for $\varphi_j(x)$. A one limitation, however, is that

1) NPTS is odd and
2) the grid-points with even index lie exactly between their neighbours.

The functions $\varphi_j(x)$ (see figure 2) are defined by

1) $\varphi_j(x)$ is continuous on $[a,b]$ and quadratic on every segment $[x_{2\ell-1}, x_{2\ell+1}]$, $j = 1,\ldots,\text{NPTS}$, $\ell = 1,\ldots,\frac{\text{NPTS}-1}{2}$;
2) $\varphi_j(x_\ell) = \delta_{j,\ell}$, $1 \le j, \ell \le \text{NPTS}$.

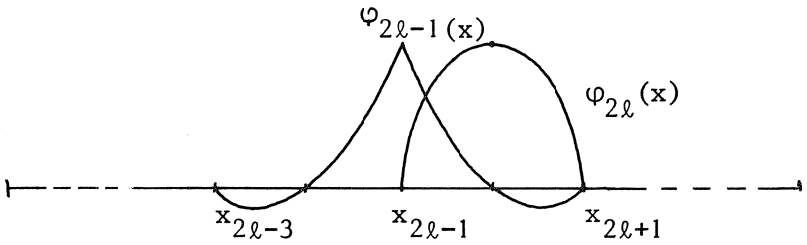

Figure 2. Piecewise Quadratics

As in the case of piecewise linear functions, we have to apply a quadrature rule to evaluate (3.3). This time we use the extended Simpson rule to approximate $(m_{j,\ell})$ and the extended two-point Gauss-Legendre rule to approximate $N_{i,j}$.

Let

$$d_j = \begin{cases} \frac{1}{6}(x_3 - x_1), & j = 1; \\ \frac{1}{6}(x_{j+2} - x_{j-2}), & j = 3,5,\ldots,\text{NPTS}-2; \\ \frac{2}{3}(x_{j+1} - x_{j-1}), & j = 2,4,\ldots,\text{NPTS}-1; \\ \frac{1}{6}(x_{\text{NPTS}} - x_{\text{NPTS}-2}), & j = \text{NPTS}; \end{cases} \tag{3.19}$$

and let

$$\xi_{2j-1} = x_{2j-1} + \frac{3-\sqrt{3}}{6} h_j; \quad h_j = x_{2j+1} - x_{2j-1}$$

$$\xi_{2j+1} = x_{2j-1} + \frac{3+\sqrt{3}}{6} (x_{2j+1} \; h_j \; x_{2j-1});$$

(3.20)
$$\vec{Y}_{2j-1} = p_1\vec{U}_{2j-1} + p_2\vec{U}_{2j} + p_3\vec{U}_{2j+1};$$

$$\vec{Y}_{2j} = p_3\vec{U}_{2j-1} + p_2\vec{U}_{2j} + p_1\vec{U}_{2j+1};$$

$$\vec{Z}_{2j-1} = (q_1\vec{U}_{2j-1} + q_2\vec{U}_{2j} + q_3\vec{U}_{2j+1})/h_j;$$

$$\vec{Z}_{2j} = -(q_3\vec{U}_{2j-1} + q_2\vec{U}_{2j} + q_1\vec{U}_{2j+1})/h_j; \quad j = 1,\ldots,\frac{NPTS-1}{2}$$

where

$$p_1 = \frac{1+\sqrt{3}}{6}; \quad p_2 = \frac{2}{3}; \quad p_3 = \frac{1-\sqrt{3}}{6}$$

$$q_1 = \frac{3+2\sqrt{3}}{6}; \quad q_2 = \frac{2}{3}\sqrt{3}; \quad q_3 = \frac{3-2\sqrt{3}}{6}.$$

Then $m_{j,\ell}$ and $N_{i,j}$ are approximated by

$$m_{j,\ell} \doteq d_j \, \delta_{j,\ell}$$

(3.21)
$$N_{i,1} \doteq - F_i(x_1,t,\vec{U}_1, \frac{\partial}{\partial x}\vec{U}_1)$$
$$- [q_1F_i(\xi_1,t,\vec{Y}_1,\vec{Z}_1) - q_3F_i(\xi_2,t,\vec{Y}_2,\vec{Z}_2)]$$
$$+ 3d_1[p_1G_i(\xi_1,t,\vec{Y}_1,\vec{Z}_1) + p_3G_i(\xi_2,t,\vec{Y}_2,\vec{Z}_2)]$$

(3.22)
$$N_{i,2j} \doteq - q_2[F_i(\xi_{2j-1},t,\vec{Y}_{2j-1},\vec{Z}_{2j-1}) - F_i(\xi_{2j},t,\vec{Y}_{2j},\vec{Z}_{2j})]$$
$$+ \tfrac{1}{2}d_{2j}[G_i(\xi_{2j-1},t,\vec{Y}_{2j-1},\vec{Z}_{2j-1}) + G_i(\xi_{2j},t,\vec{Y}_{2j},\vec{Z}_{2j})]$$
$$j = 1,\ldots,\frac{NPTS-1}{2};$$

$$(3.23)\qquad N_{i,2j+1} \doteq q_3[F_i(\xi_{2j+2},t,\vec{Y}_{2j+2},\vec{Z}_{2j+2}) - F_i(\xi_{2j-1},t,\vec{Y}_{2j-1},t,\vec{Z}_{2j-1})]$$

$$+ q_1[F_i(\xi_{2j},t,\vec{Y}_{2j},\vec{Z}_{2j}) - F_i(\xi_{2j+1},t,\vec{Y}_{2j+1},\vec{Z}_{2j+1})]$$

$$+ \tfrac{1}{2}h_j[p_3G_i(\xi_{2j-1},t,\vec{Y}_{2j-1},\vec{Z}_{2j-1}) + p_1G_i(\xi_{2j},t,\vec{Y}_{2j},\vec{Z}_{2j})]$$

$$+ \tfrac{1}{2}H_{j+1}[p_1G_i(\xi_{2j+1},t,\vec{Y}_{2j+1},\vec{Z}_{2j+1}) + p_3G_i(\xi_{2j+2},t,\vec{Y}_{2j+2},\vec{Z}_{2j+2})];$$

$$j = 1,\ldots,\frac{NPTS-3}{2};$$

$$(3.24)\qquad N_{i,NPTS} \doteq F_i(x_{NPTS},t,\vec{U}_{NPTS},\frac{\partial}{\partial x}\vec{U}_{NPTS})$$

$$- q_3F_i(\xi_{NPTS-2},t,\vec{Y}_{NPTS-2},\vec{Z}_{NPTS-2})$$

$$+ q_1F_i(\xi_{NPTS-1},t,\vec{Y}_{NPTS-1},\vec{Z}_{NPTS-1})$$

$$+ 3d_{NPTS} *$$

$$[p_3G_i(\xi_{NPTS-2},t,\vec{Y}_{NPTS-2},\vec{Z}_{NPTS-2})$$

$$+ p_1G_i(\xi_{NPTS-1},t,\vec{Y}_{NPTS-1},\vec{Z}_{NPTS-1})].$$

Again, we have to implement the boundary conditions in (3.21) and (3.24). This time, we define $\frac{\partial}{\partial x}U_{i,j}$ $(j = 1,NPTS)$ by

$$\frac{\partial}{\partial x}U_{i,1} \doteq \begin{cases} \dfrac{-3U_{i,1}+4U_{i,2}-U_{i,3}}{x_3-x_1}, & \beta_i = 0; \\ \dfrac{\gamma_i-\alpha_iU_{i,1}}{\beta_i}, & \beta_i \neq 0; \end{cases}$$

$$\frac{\partial}{\partial x}U_{i,NPTS} \doteq \begin{cases} \dfrac{U_{i,NPTS-2}-4U_{i,NPTS-1}+3U_{i,NPTS}}{x_{NPTS}-x_{NPTS-2}}, & \beta_i = 0; \\ \dfrac{\gamma_i-\alpha_iU_{i,NPTS}}{\beta_i}, & \beta_i \neq 0. \end{cases}$$

As with the roof functions, the resulting ODE becomes

$$\frac{d}{dt} U_{i,j} = \begin{cases} 0, & \beta_i = 0,\ j = 1 \text{ or } j = \text{NPTS}; \\ \dfrac{N_{i,j}}{d_j}, & \text{otherwise}, \end{cases}$$

where $N_{i,j}$ and d_j are defined by (3.21)-(3.24) and (3.19), respectively.
This algorihtm has been implemented in the subroutine PDEF2.

4. THE SUBROUTINES PDEF1 and PDEF2

The algorithms from §3 have been implemented in the subroutines PDEF1 and PDEF2. Since either of them only discretizes in the space variable and hence should be used together with an ODE integrator, a few words should be spent to such an integrator. Therefore, after the descritption of PDEF1 and PDEF2, we briefly describe an ODE integrator and its use together with the subroutines.

4.1 Description of the subroutines

The heading of PDEF1 is

```
      SUBROUTINE PDEF1(X,T,U,NPDE,NPTS,NC,FEVAL,GEVAL,BNDRY,
     * ALFA,BETA,GAMMA,UMEAN,UXMEAN,F,G)
C     INTEGER NPDE,NPTS,NC
      DIMENSION X(NPTS),U(NPDE,NPTS),ALFA(NPDE),BETA(NPDE),
     * GAMMA(NPDE),UMEAN(NPDE),UXMEAN(NPDE),F(NPDE),G(NPDE)
C     REAL T
C     EXTERNAL FEVAL,GEVAL,BNDRY

MEANING OF PARAMETERS;

X(NPTS)      :  A PARTITION OF THE X-INTERVAL;

T            :  THE TIME VARIABLE;

U(NPDE,NPTS):

ENTRY: U(I,J) IS AN APPROXIMATION OF U(I,X(J));
EXIT : U(I,J) IS A SEMI-DISCRETIZATION OF THE
       RIGHT HAND SIDE OF (2.1);

NPDE         : THE NUMBER OF P.D.E.'S;

NPTS         : THE NUMBER OF GRIDPOINTS;

NC           : A NUMBER DESIGNING WHAT KIND OF SPACE
               COORDINATES ARE USED;
               NC = 0: CARTESIAN COORDINATES;
               NC = 1: POLAR COORDINATES;
               NC = 2: SPHERICAL COORDINATES;
```

```
FEVAL        : SUBROUTINE FEVAL(X,T,U,UX,F,NPDE)
               DIMENSION U(NPDE),UX(NPDE),F(NPDE)

    THIS SUBROUTINE EVALUATES F(I,X,T,U,UX) (I = 1,...,NPDE)
    FROM THE RIGHT HAND SIDE OF (2.1) AND ASSIGNS THEM TO
    THE ARRAY F;

GEVAL        : SUBROUTINE GEVAL(X,T,U,UX,G,NPDE)
               DIMENSION U(NPDE),UX(NPDE),G(NPDE)

    THIS SUBROUTINE EVALUATES G(I,X,T,U,UX) (I = 1,...,NPDE)
    FROM THE RIGHT HAND SIDE OF (2.1) AND ASSIGNS THEM TO
    THE ARRAY G;

BNDRY        : SUBROUTINE BNDRY(T,ALFA,BETA,GAMMA,U,NPDE,LEFT)
               DIMENSION ALFA(NPDE),BETA(NPDE),GAMMA(NPDE),U(NPDE).
               LOGICAL LEFT

    THIS SUBROUTINE IMPLEMENTS THE BOUNDARY CONDITIONS
    OF THE P.D.E. (2.1);
    IF LEFT = .TRUE. , THE LEFT B.C. ARE IMPLEMENTED;
    IF LEFT = .FALSE. , THE RIGHT B.C. ARE IMPLEMENTED;
```

The heading of PDEF2 is

```
      SUBROUTINE PDEF2(X,T,U,NPDE,NPTS,FEVAL,GEVAL,BNDRY,
     * ALFA,BETA,GAMMA,UL,UR,UXL,UXR,FL,FR,GL,GR)
C     INTEGER NPDE,NPTS
      DIMENSION X(NPTS),U(NPDE,NPTS),ALFA(NPDE),BETA(NPDE),
     * GAMMA(NPDE),UL(NPDE),UR(NPDE),UXL(NPDE),UXR(NPDE),
     * FL(NPDE),FR(NPDE),GL(NPDE),GR(NPDE)
C     REAL T
C     EXTERNAL FEVAL,GEVAL,BNDRY
```

The parameters X,...,BNDRY have the same meaning as with PDEF1. Note, however, that NPTS should be *odd*. The parameters ALFA,...,GR are work arrays of dimension NPDE.

Both subroutines were written in ANSI FORTRAN and were tested on their portability by the PFORT verifyer by RYDER [16].

4.2 An ODE integrator

For the integration of the semi-discretized PDEs we used the subroutine M3RK by VERWER [22]. This is a robust explicit integrator based upon three-step multi-point Runge-Kutta schemes. Before we go on, we first explain why we used an explicit integrator and not an implicit integrator based upon Gear's method.

1) The purpose of this paper is to demonstrate the use of the semi-discretizers PDEF1 and PDEF2;
2) the integrator GEARB by HINDMARSCH [12] which exploits the sparseness of the ODE (the Jacobian is of (4*NPDE-1)-diagonal type or of block-tridiagonal type with blocks of NPDE * NPDE) was not available and other available integrators do not exploit this sparseness.

M3RK integrates autonomous ODE's of the form

$$\begin{aligned} &\frac{d\vec{Y}}{dt} = \vec{F}(\vec{Y}), \qquad t \geq t_0; \\ &\vec{Y} = \vec{Y}_0, \qquad t = t_0, \end{aligned} \tag{4.1}$$

where $\vec{Y}$, $\vec{F}$ and $\vec{Y}_0$ are N-dimensional vectors.

The ODE that M3RK has to integrate is given by an external subroutine of the form

```
SUBROUTINE DER(N,Y)
DIMENSION Y(N)
< the r.h.s. of (4.1a) is evaluated
and overwritten on Y >
RETURN
END
```

M3RK, like all robust ODE integrators, does an enormous amount of work besides the integration itself, e.g.

1) At every integration step, a new stepsize is computed based on several criteria such as accuracy and numerical stability;
2) the spectral radius, essential for the numerical stability, can be

computed automatically and regularly recomputed, as circumstances require it;

3) if the ODE tends to become "smoother", the stepsize is increased gradually; at the other hand, if the problem tends to become "nastier", the stepsize is decreased;

4) if a stepsize appears to be too large afterwards, the integration step is rejected and repeated with a smaller size.

4.3 The use of PDEF1 and PDEF2

Since M3RK integrates autonomous *vector* ODEs and PDEF1 reduces PDEs to non-autonomous *matrix* ODEs, we have to define a correspondence between the matrix U plus time and a vector $\vec{Y}$. One possibility is

$$Y_{(i-1)NPTS+j} = U_{i,j}, \quad 1 \le i \le NPDE;$$

$$Y_{NPDE*NPTS+1} = t.$$

In that case, DER may have the form

```
      SUBROUTINE DER(NEQ,Y)
      DIMENSION Y(NEQ),U(*,**),A(*),B(*),C(*),D(*),E(*),F(*),G(*)
      COMMON /LABEL/ NPDE,NPTS,NC,X(**)
C
C     FOR * AND ** THE ACTUAL VALUES OF NPDE AND NPTS HAVE TO
C     BE TAKEN; NOTE THAT NEQ = NPDE*NPTS + 1
C
      EXTERNAL FEVAL,GEVAL,BNDRY
C
      DO 10 J = 1,NPDE
        M = (J - 1)*NPTS
        DO 10 L = 1,NPTS
        U(J,L) = Y(L + M)
 10   CONTINUE
C
      T = Y(NEQ)
C
      CALL PDEF1(X,T,U,NPDE,NPTS,NC,FEVAL,GEVAL,BNDRY,A,B,C,D,E,F)
C
      DO 20 J = 1,NPDE
        M = (J - 1)*NPTS
        DO 20 L = 1,NPTS
        Y(L + M) = U(J,L)
 20   CONTINUE
C
      Y(NEQ) = 1.
      RETURN
      END
```

The hierarchical order of the subprograms in a user's program is illustrated in diagram 1.

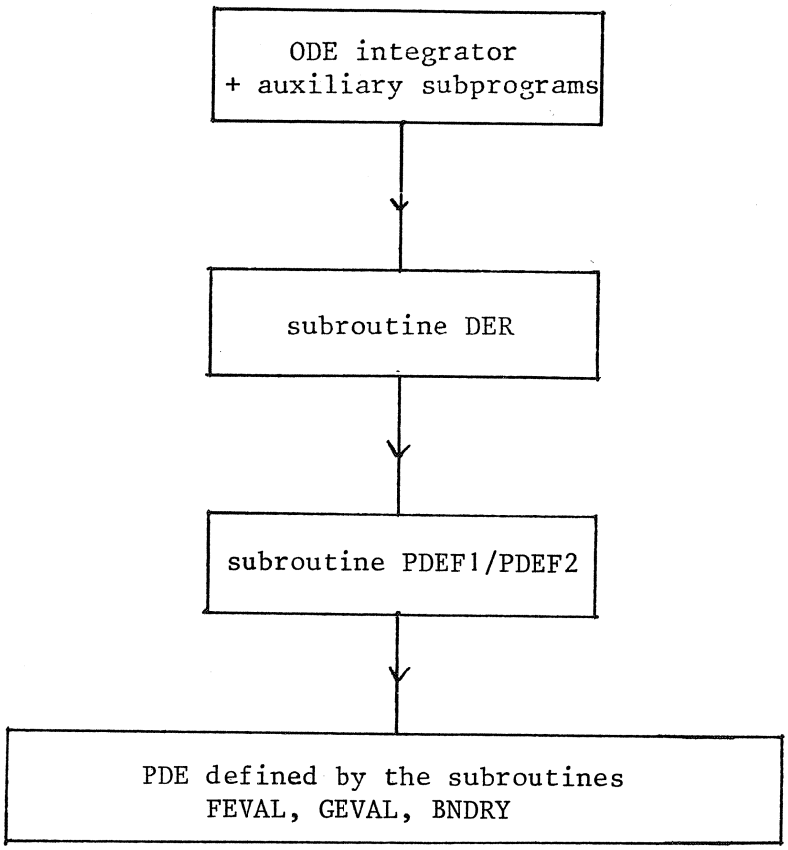

Diagram 1. Organization of a user's program

So the only things the user has to do are, roughly spoken,

1) the definition of a spatial grid;
2) the initialization of the function values and other parameters of the ODE integrator and
3) the implementation of the subroutines FEVAL, GEVAL and BNDRY.

5. NUMERICAL EXAMPLES

In this §, we give some numerical examples to demonstrate the use of PDEF1 and PDEF2.

5.1 Numerical examples for PDEF1

Example A; *a nonlinear diffusion problem.*

We consider the problem

$$\frac{\partial u}{\partial t} = x^{-2} \frac{\partial}{\partial x} (x^2 u^4 \frac{\partial u}{\partial x}), \qquad 0 < x < 1; \tag{5.1a}$$

$$\frac{\partial u}{\partial x} = 0, \quad x = 0; \tag{5.1b}$$

$$5 \int_0^1 x^2 \, u(x,t)dx + [u(1,t)]^5 = 1; \tag{5.1c}$$

$$u(x,0) = \begin{matrix} 0 & 0 \le x < 1; \\ 0, & x = 1, \end{matrix} \tag{5.1d}$$

Problems of this kind occur in non-linear diffusion theory. One easily verifies that condition (5.1c) is equivalent with $u_x + u_t = 0$, $x = 1$, but we will not use this condition.

Although, at first sight, this problem does not belong to the class described in §2, one still can reduce condition (5.1c) to a Dirichlet condition. Suppose we have a grid

$$\Delta: \ 0 = x_1 < x_2 < \ \ldots \ < x_{NPTS} = 1.$$

Then we can approximate (5.1c) by

$$5 \sum_{j=1}^{NPTS} w_j \, U_j + U_{NPTS}^5 = 1 \tag{5.2}$$

or

$$U_{NPTS}^5 + 5w_{NPTS} \, U_{NPTS} + \sum_{j=1}^{NPTS-1} w_j \, U_j - 1 = 0, \tag{5.2a}$$

where w_j ($j = 1,\ldots,NPTS$) are suitably chosen weights e.g. $w_j = d_j$, d_j defined by (3.9), (3.10) and (3.13). From (5.2a) one sees that at any time t U_{NPTS} can be calculated as a non-linear function of $U_1,\ldots,U_{NPTS-1}$, hence the right boundary is of Dirichlet type.

We also notice another property. We can rewrite (5.1a) as

$$\frac{\partial u}{\partial t} = x^{-2} \frac{\partial}{\partial x} (x^2 \frac{\partial v}{\partial x}), \quad v = \frac{u^5}{5}, \tag{5.3}$$

hence we can semi-discretize the right hand side of (5.1a) as a function of $v = 0.2*u^5$.

We divided [0,1] in 40 segments of equal length, so NPTS = 41, $x_i = (i-1)/40$, $i = 1,\ldots,41$. The semi-discretized problem (5.1) was integrated from 0 to 6.0, where the steady state was reached. As can be seen from (5.3), this steady state is a constant function u_∞. The value of u_∞ can be computed by substituting $u(x,t) \equiv u_\infty$ in (5.1c), which results in the non-linear equation

$$\frac{5}{3} u_\infty + u_\infty^5 = 1,$$

with solution $u_\infty = 0.565\ldots$.

As table I and figure 3 show, the solution behaves rather wild initially: it decreases at x = 1 but increases for x < 1.

T\X	.000	.200	.400	.600	.800	1.000
.01	.000	.000	.000	.000	.000	.942
.10	.000	.000	.000	.000	.205	.839
.50	.000	.000	.000	.031	.643	.711
1.00	.000	.000	.000	.520	.621	.653
1.50	.000	.000	.117	.547	.603	.622
2.00	.000	.000	.441	.554	.590	.603
3.00	.000	.332	.516	.557	.575	.581
4.00	.512	.526	.546	.560	.568	.571
5.00	.560	.560	.562	.564	.566	.567
6.00	.564	.565	.565	.565	.565	.566

Table I

Numerical values of u(x,t) from problem A

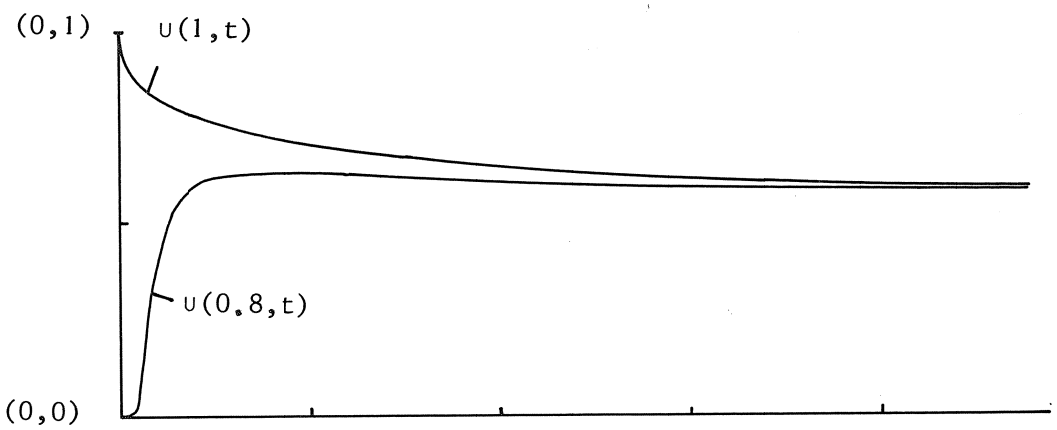

Figure 3. Graph of $u(1,t)$ and $u(0,8,t)$

Example B; *a problem with an internal boundary condition.*

The problem is given by

$$\frac{\partial u}{\partial t} = \begin{cases} 5\,x^{-2}\,\frac{\partial}{\partial x}\left(x^2\,\frac{\partial u}{\partial x}\right) - 1000\,\exp(u), & 0 < x < \tfrac{1}{2}; \\ x^{-2}\,\frac{\partial}{\partial x}\left(x^2\,\frac{\partial u}{\partial x}\right) - \exp(u), & \tfrac{1}{2} < x < 1; \end{cases} \tag{5.4a}$$

$$\frac{\partial u}{\partial x} = 0, \quad x = 0; \tag{5.4b}$$

$$u = 1, \quad x = 1; \tag{5.4c}$$

$$5 \lim_{x \uparrow 0.5} \frac{\partial u}{\partial x} = \lim_{x \downarrow 0.5} \frac{\partial u}{\partial x}\ ; \tag{5.4d}$$

$$u(x,0) = \begin{cases} 0, & x < 1; \\ 1, & x = 1. \end{cases} \tag{5.4e}$$

This problem has, besides the discontinuity of the initial values, the complication that the equation for $\partial u/\partial t$ is discontinuous in $x = 0.5$ However, this discontinuity is no problem for PDEF1. The only things the user has to do is taking care that 0.5 is one of the grid-points, and giving adequate definitions for the functions F and G. No special formula for $x = 0.5$ is needed such as a weighed mean of the left and right limit

of the function value. So although for completeness the internal boundary condition (5.4e) is also given, this condition is not actually used by PDEF1 and hence need not be processed in one of the subroutines FEVAL or GEVAL.

Since (5.4) is more singular on [0,0.5] than on [0.5,1], we divided both segments in 40 and 20 subsegments of equal length respectively. We integrated the semi-discretized problem from 0.0 until 0.5 where the steady state was reached (see table II for some results). As an extra check, we solved the steady-state problem

T\X	.00	.10	.20	.30	.40	.50	.60	.70	.80	.90
.001	-0.69	-0.69	-0.69	-0.69	-0.65	-0.47	-0.01	0.00	0.00	0.04
.005	-1.76	-1.75	-1.72	-1.64	-1.50	-1.23	-0.21	-0.02	0.05	0.35
.010	-2.22	-2.20	-2.15	-2.05	-1.88	-1.62	-0.50	-0.07	0.17	0.52
.020	-2.56	-2.54	-2.48	-2.37	-2.21	-1.96	-0.82	-0.16	0.27	0.64
.050	-2.80	-2.78	-2.72	-2.62	-2.46	-2.23	-1.10	-0.33	0.23	0.65
.100	-2.86	-2.84	-2.78	-2.68	-2.52	-2.30	-1.20	-0.42	0.16	0.62
.200	-2.87	-2.85	-2.80	-2.70	-2.54	-2.32	-1.23	-0.45	0.14	0.61
.500	-2.87	-2.85	-2.80	-2.70	-2.54	-2.32	-1.23	-0.45	0.14	0.61

Table II

Numerical values of $U(x,t)$ from problem B

by means of the Ritz-Galerkin method with a uniform grid of 100 segments. We found that the maximum error was about 1.e-3. As was already evident from the internal boundary condition (5.4d), the curve of $U(x,\infty)$ shows a crack at $x = 0.5$ (see figure 4).

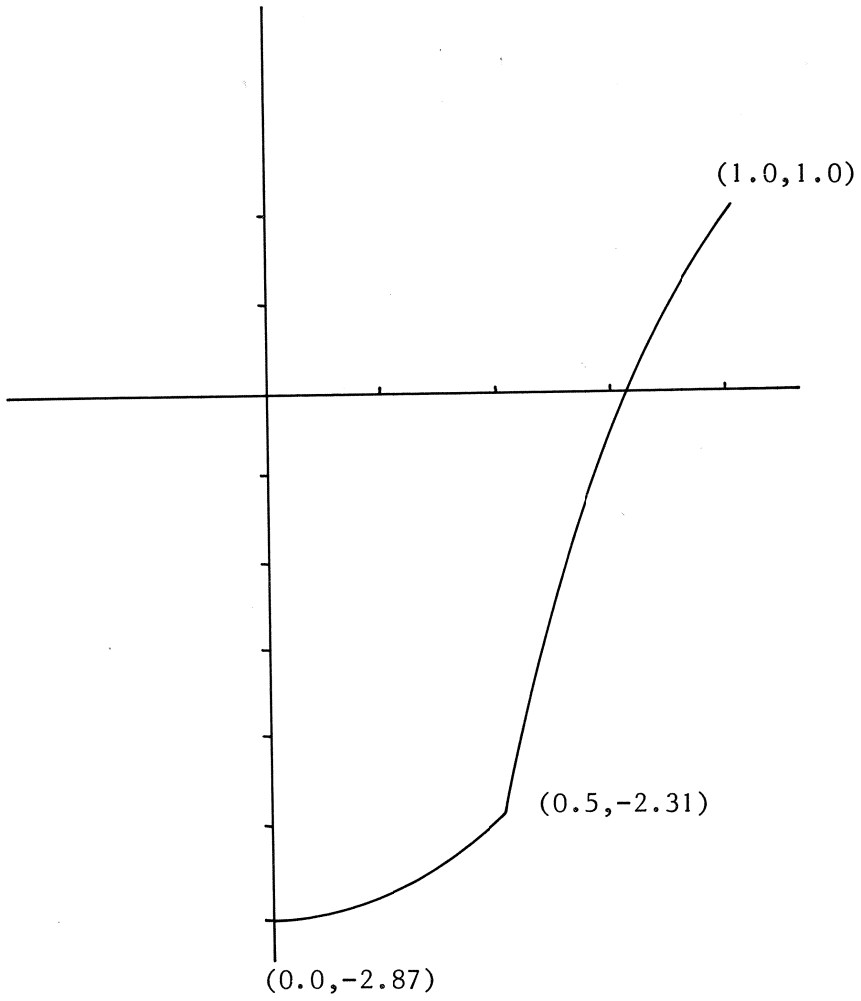

Figure 4. Steady state of problem B

Example C.

This problem was also solved by SINCOVEC & MADSEN [17]. The equation is

$$\frac{\partial u}{\partial t} = x^{-1} \frac{\partial}{\partial x}\left(x \frac{\partial u}{\partial x}\right), \quad x \in [0,1] \tag{5.5a}$$

$$u_x = \begin{cases} 0 & , \quad x = 0; \\ 1.72e\text{-}9(6.25e{+}10 - u(1,t)^4), & x = 1; \end{cases} \tag{5.5b}$$

$$u = 600, \quad t = 0. \tag{5.5c}$$

This problem has the difficulty that the right boundary condition is not compatible with the initial conditions.

T\X	0.00	0.20	0.40	0.60	0.80	1.00
.05	599.88	599.70	598.78	595.82	588.58	575.33
.10	597.45	596.59	593.80	588.30	579.37	566.90
.50	554.28	553.42	550.95	546.98	541.69	535.36
1.00	525.71	525.32	524.20	522.40	520.00	517.11
2.50	503.09	503.05	502.92	502.71	502.44	502.10
5.00	500.06	500.06	500.06	500.05	500.05	500.04
7.50	499.99	500.01	500.00	500.00	500.00	500.00

Table III

Numerical values of $u(x,t)$ from problem C

We divided [0,1] in 20 segments of equal length and integrated the ODE from 0.0 until 7.50 where $u(x,t)$ had its steady state $u(x,\infty) \equiv 500.0$ (see table III). Comparison of the results from table V with those from Sincovec & Madsen shows that they differ only in the last digit.

5.2 Numerical problems for PDEF2

Example D; *a problem from electrodynamics*

Both this problem and its steady-state problem has been treated extensively in BUS [3], pp.113-116 and TE RIELE [21], pp.38-72.

The differential equation is given by

$$\frac{\partial u}{\partial t} = 0.024 \frac{\partial^2 u}{\partial x^2} - g(u-v); \tag{5.6a}$$

$$\frac{\partial v}{\partial t} = 0.17 \frac{\partial^2 v}{\partial x^2} + g(u-v); \quad x \in [0,1]; \tag{5.6b}$$

$$g(\alpha) = \exp(5.73\alpha) - \exp(-11.46\alpha) \tag{5.6c}$$

with boundary conditions

$$u_x(0,t) = v(0,t) = 0; \tag{5.6d}$$

$$u(1,t) = 1; \; v_x(1,t) = 0 \tag{5.6e}$$

and initial values

(5.6f) $\quad u(x,0) = 1; \quad v(x,0) = 0.$

In selecting a grid for [0,1], we have to remember that (5.6) is a singular perturbation problem: the coefficients of u_{xx} and v_{xx} are small and the function $g(\alpha)$ changes rapidly with a small change of α. Consequently, boundary layers are to be expected at x = 0 and x = 1. We therefore take the following grid:

$$x_i = \begin{cases} 1.e\text{-}2 * (i-1), & i = 1,2,\ldots,11; \\ 5.0e\text{-}2 * (i-11) + 0.1, & i = 12,13,\ldots,26; \\ 1.e\text{-}2 * (i-27) + 0.9. & i = 27.28\ldots..37. \end{cases}$$

T\X	.000	.100	200	.300	.400	.500	.600	.700	.800	.900	1.000
.1	.221	.374	.475	.504	.510	.511	.512	.513	.521	.567	1.000
	.000	.271	.412	.467	.484	.488	.489	.494	.509	.552	.613
.5	.062	.148	.263	.356	.423	.472	.513	.555	.608	.692	1.000
	.000	.137	.252	.345	.415	.469	.513	.559	.614	.681	.734
1.0	.042	.106	.199	.287	.367	.441	.510	.578	.648	.733	1.000
	.000	.102	.196	.284	.364	.439	.510	.579	.649	.719	.768
1.5	.037	.094	.179	.264	.346	.425	.503	.578	.654	.740	1.000
	.000	.092	.178	.263	.345	.424	.502	.578	.653	.726	.774
2.0	.035	.090	.172	.255	.336	.417	.496	.573	.651	.739	1.000
	.000	.088	.172	.254	.336	.416	.495	.573	.649	.724	.773
2.5	.034	.088	.169	.250	.331	.411	.490	.569	.647	.736	1.000
	.000	.086	.168	.250	.331	.411	.490	.568	.645	.720	.770
3.0	.033	.087	.167	.247	.327	.407	.486	.565	.643	.733	1.000
	.000	.085	.166	.247	.327	.407	.486	.564	.642	.718	.768
3.5	.033	.086	.165	.245	.325	.404	.483	.562	.641	.731	1.000
	.000	.084	.165	.245	.325	.404	.483	.562	.640	.716	.766
4.0	.033	.086	.164	.244	.323	.403	.481	.560	.639	.730	1.000
	.000	.084	.164	.244	.323	.402	.481	.560	.638	.714	.765
4.5	.033	.085	.163	.243	.322	.401	.480	.559	.638	.729	1.000
	.000	.083	.163	.243	.322	.401	.480	.559	.637	.713	.764

Table IV

Numerical values of u and v from problem D; on the first row, u is listed, on the second row, v is listed

We integrated the semi-discretized problem from 0.0 to 4.5. From the very beginning, the boundary layer structure of problem D is visible, especially at x = 1.0 where u is very steep compared with the rest of the interval. One also sees that the steady-state curves of u and v intermingle between 0.1 and 0.9 (see firgure 5) and separate at the boundary.

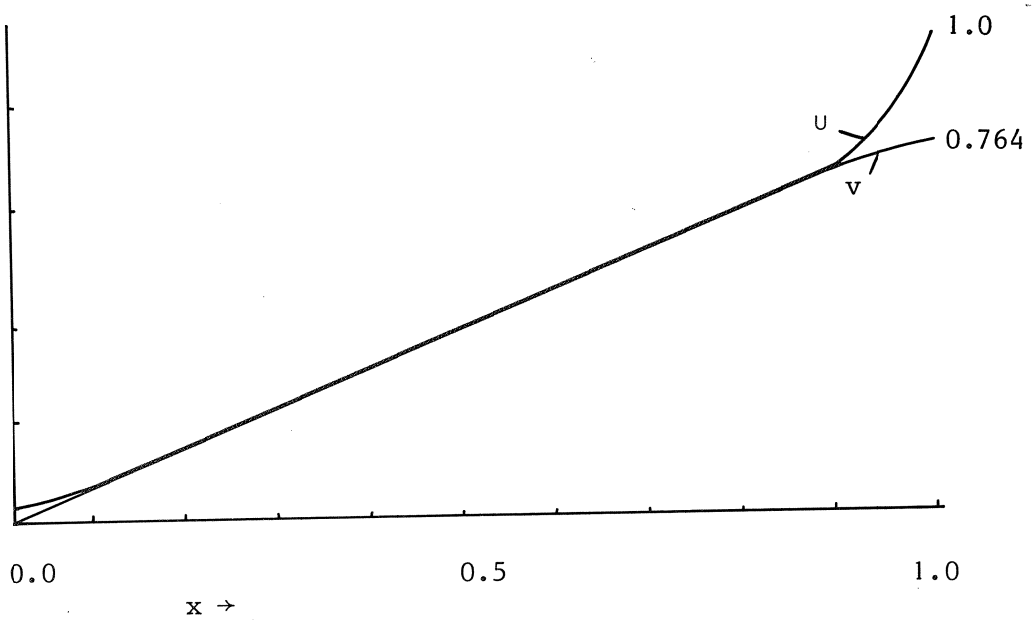

Figure 5. Steady state of problem D

Also, they practically coincide with the straight line y = 0.8x except at the boundary layers.

Example E

This problem was also solved by SINCOVEC & MADSEN [17]. The differential equation is

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(u \frac{\partial u}{\partial x}\right) - u^2, \qquad x \in [0,1];$$

$$u(0,t) = 50; \qquad u_x(1,t) = 1-\sin u(1,t); \tag{5.7}$$

$$u(x,0) = \begin{cases} 50, & x = 0; \\ 100, & x > 0. \end{cases}$$

This problem has the difficulties that

1) the initial values are discontinuous and

2) the right boundary condition is not satisfied initially.

Since PDEF2 semi-discretizes more accurately, we take a uniform grid of only 11 points: NPTS = 11. We integrated from 0.0 to 0.1 where the steady state was reached.

T\X	0.0	.2000	.4000	.6000	.8000	1.0000
.001	50.0	68.3413	79.6891	85.9974	89.0548	89.9996
.005	50.0	51.6163	53.9119	56.0277	57.4984	58.0336
.010	50.0	46.8787	45.0878	44.2096	43.8807	43.9113
.015	50.0	45.3898	42.0891	39.9345	38.7495	38.4099
.020	50.0	44.8124	40.9007	38.2334	36.7893	36.5612
.025	50.0	44.6149	40.4883	37.6285	36.0580	35.7932
.030	50.0	44.5292	40.3031	37.3427	35.6854	35.3600
.035	50.0	44.4816	40.1991	37.1765	35.4598	35.0838
.040	50.0	44.4516	40.1320	37.0675	35.3076	34.8944
.045	50.0	44.4309	40.0867	36.9934	35.2038	34.7628
.050	50.0	44.4168	40.0541	36.9411	35.1294	34.6680
.100	50.0	44.3827	39.9786	36.8162	34.9532	34.4431

Table V

Numerical values of u from problem E

The exact solution of the steady-state problem is

$$u_\infty(x) = 50.\sqrt{\cosh x\sqrt{2} - c \sinh x\sqrt{2}}$$

$$c = 0.88055353224.$$

Comparison of $u_\infty(x)$ with the values of U(x,0.1) from table V shows that the maximum error is 8.2e-3.

6. CONCLUDING REMARKS

In the previous § we succesfully semi-discretized and integrated five second order parabolic PDEs with various difficulties and pitfalls, such as

1) initial discontinuity of the function values or the spatial derivatives;

2) internal discontinuity of the spatial derivative;
3) singular perturbation;
4) strong interdependence of boundary value and interior values.

With exception of A, none of the problems required more complex action than the selection of a proper grid and of course correct implementation of the PDE. Concerning A, it is an open question whether a simpler approach had been possible.

We intentionally confined ourselves to (nearly) parabolic problems which were not too singularly perturbed. The reason is that it is uncertain whether the semi-discretization method used is also suitable for other kinds of PDEs such as first order hyperbolic (see STRANG & FIX, pp.254-256) PDEs or singularly perturbed PDEs. In the latter case it may be wiser and cheaper to use exponentially fitted splines, as HEMKER [10] proved for the steady-state problem.

All the same, in spite of these possible limitations, we can say that with PDEF1 and PDEF2 we have delivered two subroutines which can semi-discretize a broad class of time-dependent PDEs in one space variable.

APPENDIX

In this appendix, we give listings of the subroutines and of the test programs.

PDEF1.

```
      SUBROUTINE PDEF1(X,T,U,NPDE,NPTS,NC,FEVAL,GEVAL,BNDRY,
     * ALFA,BETA,GAMMA,UMEAN,UX,F,G)
      DIMENSION X(NPTS),U(NPDE,NPTS),ALFA(NPDE),BETA(NPDE),
     * GAMMA(NPDE),UX(NPDE),UMEAN(NPDE),F(NPDE),G(NPDE)
C
C     THIS SUBROUTINE SERVES AS AN INTERFACE BETWEEN  A
C     PARABOLIC PARTIAL DIFFERENTIAL EQUATION IN ONE
C     SPACE VARIABLE AND AN INTEGRATOR OF INITIAL VALUE PROBLEMS.
C
C     THE DIFFERENTIAL EQUATION SHOULD BE OF THE FORM
C
C
C     (D/DT)U(I) = (D/DX)(X**NC*F(I,X,T,U,UX)))/X**NC
C
C                                  + G(I,X,T,U,UX),I= 1,...,NPDE;
C
C
C     WITH BOUNDARY CONDITIONS
C
C
C     ALFA(I)*U(I) + BETA(I)*UX(I) = GAMMA(I) ,I = 1,...,NPDE;
C
C
C     PDEF1 TRANSFORMS THE SYSTEM OF PARTIAL DIFFERENTIAL EQUATIONS
C     INTO A SYSTEM OF ORDINARY EQUATIONS BY SEMI-DISCRETIZATION IN
C     THE SPACE VARIABLE X; THIS SEMI-DISCRETIZATION IS PERFORMED
C     BY APPLICATION OF THE FINITE ELEMENT METHOD (SEE E.G.
C     G. STRANG & G.J. FIX,AN ANALYSIS OF THE FINITE ELEMENT METHOD)
C
C
C     DESCRIPTION OF THE PARAMETERS;
C
C     INPUT:
C
C     X:    DIMENSION X(NPTS);
C           X(1),...,X(NPTS) IS A PARTITION OF [X(1),X(NPTS)]
C
C     T:    THE TIME VARIABLE;
C
C     U:    DIMENSION U(NPDE,NPTS);
C           U(I,J) IS AN APPROXIMATION OF U(I,X(J),T),I = 1,...,NPDE;
C                                                     J = 1,...,NPTS;
C
```

```
C
C
C     NPDE:    THE NUMBER OF PARTIAL DIFFERENTIAL EQUATIONS;
C
C     NPTS:    THE NUMBER OF GRIDPOINTS;
C
C     NC:    A NUMBER DESIGNING THE KIND OF SPACE COORDINATES;
C            NC = 0 : CARTESIAN COORDINATES;
C            NC = 1 : CIRCULAR COORDINATES;
C            NC = 2 : SPHERICAL COORDINATES;
C
C     FEVAL:    SUBROUTINE FEVAL(X,T,U,UX,F,NPDE)
C               DIMENSION U(NPDE),UX(NPDE),F(NPDE)
C
C          EXIT: THE VALUES OF F(I,X,T,U,UX) ARE ASSIGNED
C          TO THE ARRAY F;
C
C     GEVAL:    SUBROUTINE GEVAL(X,T,U,UX,G,NPDE)
C               DIMENSION U(NPDE),UX(NPDE),G(NPDE)
C
C          EXIT: THE VALUES OF G(I,X,T,U,UX) ARE ASSIGNED
C          TO THE ARRAY G;
C
C     BNDRY:    SUBROUTINE BNDRY(T,ALFA,BETA,GAMMA,U,NPDE,LEFT)
C               DIMENSION ALFA(NPDE),BETA(NPDE),GAMMA(NPDE),U(NPDE)
C               LOGICAL LEFT
C
C          THIS SUBROUTINE DEFINES THE BOUNDARY CONDITIONS OF THE PDE
C          IN X(1) OR X(NPTS): IF LEFT = .TRUE.,THE BOUNDARY CONDI-
C          TIONS IN X(1) ARE DEFINED,OTHERWISE THE B.C. IN X(NPTS);
C          ALFA,BETA AND GAMMA MAY DEPEND ON U AND T.
C
C     ALFA,BETA,GAMMA,UMEAN,UX,F,G:
C
C          WORK-ARRAYS OF DIMENSION NPDE.
C
C     OUTPUT:
C
C     U:    DIMENSION U(NPDE,NPTS);
C           THE SEMI-DISCRETIZED RIGHT HAND SIDE OF THE PDE OVERWRITTEN
C           ON U;
C
      WR = 0.
      XR = X(1)
      DO 150 L = 2,NPTS
C
C       THE SEGMENTWISE ASSEMBLY OF THE RIGHT HAND SIDE BEGINS;
C      AT FIRST, SOME NUMBERS CHARACTERISTIC FOR NC ARE COMPUTED.
C
      XL = XR
```

```
      L1 = L - 1
      XR = X(L)
      H = XR - XL
      IF (NC .EQ. 1) GOTO 10
      IF (NC .EQ. 2) GOTO 20
      VL = 0.5
      VR = 0.5
      GOTO 30
   10 VL = XL/3. + XR/6.
      VR = XL/6. + XR/3.
      GOTO 30
   20 XLSQ = XL*XL/12.
      XLXR = XL*XR/6.
      XRSQ = XR*XR/12.
      VL = 3.*XLSQ + XLXR + XRSQ
      VR = XLSQ + XLXR + 3.*XRSQ
   30 WL = H*VL + WR
      WR = H*VR
      VMEAN = VL + VR
      WMEAN = H*VMEAN
      PR = VR/VMEAN
      PL = 1.0 - PR
      XMEAN = XL + H*PR
      IF (L .GT. 2) GOTO 90
C
C       ON THE FIRST SEGMENT,AT FIRST THE BOUNDARY CONDITIONS ARE
C       PROCESSED;IF THEY ARE OF DIRICHLET TYPE,THE VALUE OF U(X(1),T)
C       IS UPDATED AND THE TIME DERIVATIVE IS PUT EQUAL TO ZERO;
C       IF THEY ARE OF MIXED TYPE,THE SPATIAL DERIVATIVE OF U(X(1),T)
C       OF U(X(1),T) IS EXPRESSED IN U AND T AND THAT EXPRESSION IS
C       IS IMPLEMENTED IN THE STOCK TERM
C
      DO 40 J = 1,NPDE
      UMEAN(J) = U(J,1)
      U(J,1) = 0.
   40 CONTINUE
      CALL BNDRY(T,ALFA,BETA,GAMMA,UMEAN,NPDE,.TRUE.)
      ICT = 0
      DO 60 J = 1,NPDE
      IF (BETA(J) .NE. 0.) GOTO 60
      ICT = ICT + 1
      UMEAN(J) = GAMMA(J)/ALFA(J)
   60 CONTINUE
      XPOW = 1.
      IF (NC .GT. 0) XPOW = XL**NC
      IF (XPOW .EQ. 0. .OR. ICT .EQ. NPDE) GOTO 90
      IF (ICT .GT. 0) CALL BNDRY(T,ALFA,BETA,GAMMA,UMEAN,NPDE,.TRUE.)
```

```
      DO 70 J = 1,NPDE
      IF (BETA(J) .NE. 0.)
     * UX(J) = (GAMMA(J) - ALFA(J)*UMEAN(J))/BETA(J)
      IF (BETA(J) .EQ. 0.) UX(J) = (U(J,2) - UMEAN(J))/H
   70 CONTINUE
      CALL FEVAL(XL,T,UMEAN,UX,F,NPDE)
C
C       NOW,THE STOCK TERMS ARE IMPLEMENTED,AS FAR AS THE
C       BOUNDARY CONDITIONS ARE OD MIXED TYPE.
C
      DO 80 J = 1,NPDE
   80 IF (BETA(J) .NE. 0.) U(J,1) = - XPOW*F(J)
C
C       HERE THE IMPLEMENTATION OF THE LEFT BOUNDARY CONDITIONS
C       ENDS
C
   90 IF (L .LT. NPTS) GOTO 120
C
C       AT THE LAST SEGMENT,THE RIGHT BOUNDARY CONDITIONS,AS FAR
C       AS THEY ARE OF DIRICHLET TYPE ARE IMPLEMENTED BY UPDATING
C       THE BOUNDARY VALUES;
C
      DO 100 J = 1,NPDE
      IF (BETA(J) .EQ. 0.) U(J,1) = 0.
      UX(J) = U(J,NPTS)
  100 CONTINUE
      CALL BNDRY(T,ALFA,BETA,GAMMA,UX,NPDE,.FALSE.)
      ICT = 0
      DO 110 J = 1,NPDE
      IF (BETA(J) .NE. 0.) GOTO 110
      ICT = ICT + 1
      U(J,NPTS) = GAMMA(J)/ALFA(J)
  110 CONTINUE
C
C       NOW,THE REAL ASSEMBLY BEGINS
C
  120 DO 130 J = 1,NPDE
      ULJ = UMEAN(J)
      URJ = U(J,L)
      UMEAN(J) = PL*ULJ + PR*URJ
      UX(J) = (URJ - ULJ)/H
  130 CONTINUE
      CALL FEVAL(XMEAN,T,UMEAN,UX,F,NPDE)
      CALL GEVAL(XMEAN,T,UMEAN,UX,G,NPDE)
      DO 140 J = 1,NPDE
      FMEAN = VMEAN*F(J)
      GMEAN = WMEAN*G(J)
      U(J,L1) = (U(J,L1) + FMEAN + PL*GMEAN)/WL
```

```
      UMEAN(J) = U(J,L)
      U(J,L) = - FMEAN + PR*GMEAN
  140 CONTINUE
  150 CONTINUE
C
C       FINALLY,THE PROCESSING OF THE RIGHT BOUNDARY CONDITIONS
C       IS PERFORMED.
C
      DO 160 J = 1,NPDE
  160 IF (BETA(J) .EQ. 0.) U(J,NPTS) = 0.
      IF (ICT .EQ. NPDE) RETURN
      IF (ICT .GT. 0) CALL BNDRY(T,ALFA,BETA,GAMMA,UMEAN,NPDE,.FALSE.)
      XPOW = XR**NC
      DO 170 J = 1,NPDE
  170 IF (BETA(J) .NE. 0.) UX(J) = (GAMMA(J)-ALFA(J)*UMEAN(J))/BETA(J)
      CALL FEVAL(XR,T,UMEAN,UX,F,NPDE)
      DO 180 J = 1,NPDE
  180 IF (BETA(J) .NE. 0.) U(J,NPTS) = (U(J,NPTS) + XPOW*F(J))/WR
      RETURN
      END
```

PDEF2.

```
      SUBROUTINE PDEF2(X,T,U,NPDE,NPTS,FEVAL,GEVAL,BNDRY,
     * ALFA,BETA,GAMMA,UL,UR,UXL,UXR,FL,GL,FR,GR)
      DIMENSION X(NPTS),U(NPDE,NPTS),ALFA(NPDE),BETA(NPDE),
     * GAMMA(NPDE),UL(NPDE),UR(NPDE),UXL(NPDE),UXR(NPDE),
     * FL(NPDE),GL(NPDE),FR(NPDE),GR(NPDE)
C
C     THIS SUBROUTINE SERVES AS AN INTERFACE BETWEEN  A
C     TIME-DEPENDENT PARTIAL DIFFERENTIAL EQUATION IN ONE
C     SPACE VARIABLE AND AN INTEGRATOR OF INITIAL VALUE PROBLEMS.
C
C     THE DIFFERENTIAL EQUATIONS SHOULD BE OF THE FORM
C
C
C     (D/DT)U(I) = (D/DX)F(I,X,T,U,UX))
C
C                                + G(I,X,T,U,UX),I= 1,...,NPDE;
C
C
C     WITH BOUNDARY CONDITIONS
C
C
C     ALFA(I)*U(I) + BETA(I)*UX(I) = GAMMA(I) ,I = 1,... ,NPDE;
C
C
C     PDEF2  TRANSFORMS THE SYSTEM OF PARTIAL DIFFERENTIAL EQUATIONS
C     INTO A SYSTEM OF ORDINARY EQUATIONS BY SEMI-DISCRETIZATION IN
C     THE SPACE VARIABLE X; THIS SEMI-DISCRETIZATION IS PERFORMED
C     BY APPLICATION OF THE FINITE ELEMENT METHOD (SEE E.G.
C     G. STRANG & G.J. FIX,AN ANALYSIS OF THE FINITE ELEMENT METHOD)
C
```

```
C
C     DESCRIPTION OF THE PARAMETERS;
C
C     INPUT:
C
C     X:   DIMENSION X(NPTS);
C          X(1),...,X(NPTS) IS A PARTITION OF [ X(1),X(NPTS)]
C
C     T:   THE TIME VARIABLE;
C
C     U:   DIMENSION U(NPDE,NPTS);
C          U(I,J) IS AN APPROXIMATION OF U(I,X(J),T),I = 1,...,NPDE;
C                                                  J = 1,...,NPTS;
C
C     NPDE:   THE NUMBER OF PARTIAL DIFFERENTIAL EQUATIONS;
C
C     NPTS:   THE NUMBER OF GRIDPOINTS;
C
C ---------> IMPORTANT! BECAUSE OF THE METHOD OF SEMI-DISCRETIZATION
C ---------> NPTS SHOULD DEFINITELY BE ODD!
C
C     FEVAL:   SUBROUTINE FEVAL(X,T,U,UX,F)
C              DIMENSION U(NPDE),UX(NPDE),F(NPDE)
C
C          EXIT: THE ARRAY F IS ASSIGNED THE VALUES OF THE
C          FUNCTION F(I,X,T,U,UX) OF THE P.D.E.;
C
C     GEVAL:   SUBROUTINE GEVAL(X,T,U,UX,G)
C              DIMENSION U(NPDE),UX(NPDE),G(NPDE)
C
C          EXIT: THE ARRAY G IS ASSIGNED THE VALUES OF THE
C          FUNCTION G(I,X,T,U,UX) OF THE P.D.E.;
C
C     BNDRY:   SUBROUTINE BNDRY(T,ALFA,BETA,GAMMA,U,NPDE,LEFT)
C              DIMENSION ALFA(NPDE),BETA(NPDE),GAMMA(NPDE),U(NPDE)
C              LOGICAL LEFT
C
C          THIS SUBROUTINE DEFINES THE BOUNDARY CONDITIONS OF THE PDE
C          IN X(1) OR X(NPTS): IF LEFT = .TRUE.,THE BOUNDARY CONDITIONS
C          IN X(1) ARE DEFINED,OTHERWISE THE B.C. IN X(NPTS);
C          ALFA,BETA AND GAMMA MAY DEPEND ON U AND T;
C
C     ALFA,BETA,GAMMA,UL,UR,UXL,UXR,F,G:
C
C          WORK-ARRAYS OF DIMENSION NPDE.
C
C
C     OUTPUT:
C
C     U:   DIMENSION U(NPDE,NPTS);
C          THE SEMI-DISCRETIZED RIGHT HAND SIDE OF THE PDE OVERWRITTEN
C          ON U;
```

```
C
C       PREPARATORY CALCULATIONS
C
        PL1 = 0.455341800126
        PL2 = 0.666666666667
        PL3 = 1. - PL1 - PL2
C
C       PL1 =(1. + SQRT(3.))/6.
C
        QL1 = - 2.1547005383793
        QL3 = - 0.1547005383793
        QL2 = - QL1 - QL3
C
C       QL1 = - 1. - SQRT(3.)/1.5
C       QL3 = + 1. - SQRT(3.)/1.5
C
        XR = X(1)
        HOLD = 0.0
C
C       THE ASSEMBLING BEGINS
C
        DO 110 L = 3,NPTS,2
C
        L1 = L - 1
        L2 = L - 2
        XL = XR
        XR = X(L)
        H = XR - XL
        WL = 3./(H + HOLD)
        HOLD = H
        XLM = XL + H*0.21132498654052
        XRM = XL + XR - XLM
C
C       XLM AND XRM ARE THE ABSCISSAE FOR TWO POINT
C       GAUSS-LEGENDRE QUADRATURE OVER  [X(L-2),X(L)]
C
        PL1H = PL1*H
        PL3H = PL3*H
        QL2H = QL2*0.75/H
        IF (L .GT. 3) GOTO 50
C
C       THE LEFT BOUNDARY CONDITIONS ARE IMPLEMENTED
C
        DO 10 J = 1,NPDE
   10   UR(J) = U(J,1)
        CALL BNDRY(T,ALFA,BETA,GAMMA,UR,NPDE,.TRUE.)
        ICT = 0
        DO 20 J = 1,NPDE
        IF (BETA(J) .NE. 0.0) GOTO 20
        ICT = ICT + 1
        UR(J) = GAMMA(J)/ALFA(J)
        U(J,1) = 0.0
   20   CONTINUE
```

```
      IF (ICT .EQ. NPDE) GOTO 50
      IF (ICT .GT. 0) CALL BNDRY(T,ALFA,BETA,GAMMA,UR,NPDE,.TRUE.)
      DO 30 J = 1,NPDE
      IF (BETA(J) .EQ. 0.0)
     * UXL(J) = (- 3.*UR(J) + 4.*U(J,2) - U(J,3))/H
C
C
C
      IF (BETA(J) .NE. 0.0)
     * UXL(J) = (- ALFA(J)*UR(J) + GAMMA(J))/BETA(J)
  30  CONTINUE
      CALL FEVAL(XL,T,UR,UXL,FL,NPDE)
      DO 40 J = 1,NPDE
  40  IF (BETA(J) .NE|

  40  IF (BETA(J) .NE. 0.0) U(J,1) = - FL(J)*2.
C
C     END OF IMPLEMENTATION OF LEFT B.C.
C
  50  IF (L .LT. NPTS) GOTO 80
C
C     THE RIGHT BOUNDARY CONDTIONS AS FAR AS THEY ARE
C     OF DIRICHLET TYPE,ARE IMPLEMENTED
C
      DO 60 J = 1,NPDE
      UXR(J) = U(J,NPTS)
      IF (BETA(J) .EQ. 0.0) U(J,1) = 0.0
  60  CONTINUE
      CALL BNDRY(T,ALFA,BETA,GAMMA,UXR,NPDE,.FALSE.)
      ICT = 0
      DO 70 J = 1,NPDE
      IF (BETA(J) .NE. 0.0) GOTO 70
      ICT = ICT + 1
      U(J,NPTS) = GAMMA(J)/ALFA(J)
  70  CONTINUE
C
C     NOW THE SEGMENTWISE ASSEMBLAGE BEGINS
C
  80  DO 90 J = 1,NPDE
      ULJ = UR(J)
      UMJ = U(J,L1)
      URJ = U(J,L)
      UL(J) = PL1*ULJ + PL2*UMJ + PL3*URJ
      UR(J) = PL3*ULJ + PL2*UMJ + PL1*URJ
      UXL(J) = (QL1*ULJ + QL2*UMJ + QL3*URJ)/H
      UXR(J) = - (QL3*ULJ + QL2*UMJ + QL1*URJ)/H
  90  CONTINUE
C
      CALL FEVAL(XLM,T,UL,UXL,FL,NPDE)
      CALL GEVAL(XLM,T,UL,UXL,GL,NPDE)
      CALL FEVAL(XRM,T,UR,UXR,FR,NPDE)
      CALL GEVAL(XRM,T,UR,UXR,GR,NPDE)
      DO 100 J = 1,NPDE
      FLJ = FL(J)
      FRJ = FR(J)
      GLJ = GL(J)
      GRJ = GR(J)
```

```
      U(J,L2) = WL*(U(J,L2)-QL1*FLJ+QL3*FRJ+(PL1H*GLJ+PL3H*GRJ))
      U(J,L1) = QL2H*(FRJ-FLJ) + 0.5*(GLJ+GRJ)
      UR(J) = U(J,L)
      U(J,L) = - (QL3*FLJ - QL1*FRJ) + (PL3H*GLJ + PL1H*GRJ)
  100 CONTINUE
C
  110 CONTINUE
C
C     END OF THE ASSEMBLAGE; ONLY THE RIGHT BOUNDARY
C     CONDITIONS HAVE TO BE IMPLEMENTED
C
      WL = 3./HOLD
      IF (ICT .GT. 0 .AND. ICT .LT. NPDE)
     * CALL BNDRY(T,ALFA,BETA,GAMMA,UR,NPDE,.FALSE.)
      P2 = (XR - XLM)/(XRM - XLM)
      P1 = 1. - P2
      DO 130 J = 1,NPDE
      IF (BETA(J) .EQ. 0.0) GOTO 120
      UXR(J) = (GAMMA(J) - ALFA(J))/BETA(J)
      GOTO 130
  120 U(J,NPTS) = 0.0
      UXR(J) = P1*UXL(J) + P2*UXR(J)
  130 CONTINUE
      IF (ICT .EQ. NPDE) RETURN
      CALL FEVAL(XR,T,UR,UXR,FR,NPDE)
      DO 140 J = 1,NPDE
      IF (BETA(J) .NE. 0.0) U(J,NPTS) = WL*(U(J,NPTS) + 2.*FR(J))
  140 CONTINUE
      RETURN
      END
```

PROBLEM A

```
      PROGRAM PDE(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      DIMENSION U(42),U1(42),U2(42),UOUT(42),DU(42),DU1(42),
     * SIGMA(2),INFO(15)
      EXTERNAL DER
      COMMON /ALL/ NPDE,NPTS,NC,X(41) /UPD/ W(41) /BND/ UBND
C
      NPDE = 1
      NPTS = 41
      NEQ = NPDE*NPTS + 1
      NC = 2
```

```
C
C       DEFINITION OF GRID AND INITIAL VALUES PLUS
C       ASSIGNMENT OF THE GAUSSIAN WEIGHTS NEEDED FOR
C       THE COMPUTATION OF UBND AT EACH CALL OF DER
C
      W(1) = 0.
      DO 10 L = 1,NPTS
      X(L) = FLOAT(L - 1)/FLOAT(NPTS - 1)
      U(L) = 0.
      IF (L .EQ. 1) GOTO 10
      XL = X(L-1)
      XR = X(L)
      H = (XR - XL)/12.
      W(L-1) = W(L-1) + H*(3.*XL**2 + 2.*XL*XR + XR**2)
      W(L) = H*(XL**2 + 2.*XL*XR + 3.*XR**2)
   10 CONTINUE
      U(NPTS) = 1.
      UBND = 1.
      WRITE(6,3) (X(I),I = 1,41,8)
    3 FORMAT(1H ,4H T\X,6F7.3)
C
C       FURTHER INITIALIZATION OF THE PARAMETERS OF M3RK
C
      TOL = 1.E-4
      INFO(1) = 0
      INFO(2) = 1
      INFO(3) = 20000
      SIGMA(1) = 40./X(2)**2
      T = 0.
      U(NEQ) = T
      DO 90 KL = 1,10
         READ(5,*) TE
         CALL M3RK(T,TE,NEQ,H,HMIN,SIGMA,TOL,DER,
     *        U,U1,U2,UOUT,DU,DU1,IFLAG,INFO)
         CALL UPDATE(UOUT)
         WRITE(6,1)
         WRITE(6,2) TE,(UOUT(I),I = 1,41,8)
    1    FORMAT(1H )
    2    FORMAT(1H ,F4.1,6F7.3)
   90 CONTINUE
      STOP
      END

C
      SUBROUTINE DER(NEQ,Y)
      DIMENSION Y(NEQ),U(1,41),A(1),B(1),C(1),D(1),E(1),F(1),G(1)
      EXTERNAL FEVAL,GEVAL,BNDRY
      COMMON /ALL/ NPDE,NPTS,NC,X(41)
C
C         AT FIRST, Y(NPTS) IS UPDATED
C
      CALL UPDATE(Y)
      DO 10 L = 1,NPTS
   10 U(1,L) = .2*Y(L)**5
      T = Y(NEQ)
      CALL PDEF1(X,T,U,NPDE,NPTS,NC,FEVAL,GEVAL,BNDRY,A,B,C,D,E,F,G)
      DO 20 L = 1,NPTS
   20 Y(L) = U(1,L)
      RETURN
      END
```

```
      SUBROUTINE BNDRY(T,ALFA,BETA,GAMMA,U,NPDE,LEFT)
      DIMENSION ALFA(NPDE),BETA(NPDE),GAMMA(NPDE),U(NPDE)
      LOGICAL LEFT
      COMMON /BND/ UBND
      IF (.NOT. LEFT) GOTO 10
      ALFA(1) = 0.
      BETA(1) = 1.
      GAMMA(1) = 0
      RETURN
10    ALFA(1) = 1.
      BETA(1) = 0.
      GAMMA(1) = 0.2*UBND**5
      RETURN
      END

      SUBROUTINE FEVAL(X,T,U,UX,F,NPDE)
      DIMENSION U(NPDE),UX(NPDE),F(NPDE)
      F(1) = UX(1)
      RETURN
      END

      SUBROUTINE GEVAL(X,T,U,UX,G,NPDE)
      DIMENSION U(NPDE),UX(NPDE),G(NPDE)
      G(1) = 0.
      RETURN
      END

      SUBROUTINE UPDATE(Y)
      DIMENSION Y(42)
      COMMON /UPD/ W(41) /BND/ YBND
      S = 0.
      DO 10 L = 1,40
10    S = S + W(L)*Y(L)
      C1 = W(41)
      C3 = S - 0.2

        COMPUTATION OF YBND BY MEANS OF NEWTON-RAPHSON METHOD

20    DY = (C1*YBND + 0.2*YBND**5 + C3)/(C1 + YBND**4)
      YBND = YBND - DY
      IF (ABS(DY) .GT. 5.E-3) GOTO 20
      Y(41) = YBND
      RETURN
      END
```

PROBLEM B

```
      PROGRAM PDE(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      DIMENSION U(62),U1(62),U2(62),UOUT(62),DU(62),DU1(62),
     * SIGMA(2),INFO(15)
      EXTERNAL DER
      COMMON /ALL/ NPDE,NPTS,NC,X(61)
C
      NPTS = 61
      NPDE = 1
      NEQ = NPDE*NPTS + 1
      NC = 2
C
C          DEFINITION OF GRID AND INITIALIZATION OF VALUES
C
      DO 10 I = 1,NPTS
      IF (I .LT. 41) X(I) = FLOAT(I - 1)/80.
      IF (I .GE. 41) X(I) = FLOAT(I - 41)/40. + 0.5
      U(I) = 0.
   10 CONTINUE
      U(NPTS) = 1.
C
C          FURTHER INITIALIZATION OF THE PARAMETERS OF M3RK
C
      TOL = 1.0E-4
      INFO(1) = 0
      INFO(2) = 2
      INFO(3) = 20000
      T = 0.0
      U(NEQ) = T
      WRITE(6,2)5H  T\X, (X(I),I = 1,41,8),(X(I),I=45,57,4)
    2 FORMAT(1H ,A5,10F7.3)
      WRITE(6,1)
    1 FORMAT(1H )
      DO 90 KL = 1,10
      TE = .001
      READ(5,*) TE
      CALL M3RK(T,TE,NEQ,H,HMIN,SIGMA,TOL,DER,
     * U,U1,U2,UOUT,DU,DU1,IFLAG,INFO)
      WRITE(6,4) TE, (UOUT(I),I = 1,41,8),(UOUT(I),I=45,57,4)
    4 FORMAT(1H ,F5.3,10F7.3)
   90 CONTINUE
      STOP
      END
```

```
      SUBROUTINE DER(NEQ,Y)
      DIMENSION Y(NEQ),U(1,41),A(1),B(1),C(1),D(1),E(1),F(1),G(1)
      EXTERNAL FEVAL,GEVAL,BNDRY
      COMMON /ALL/ NPDE,NPTS,NC,X(61)
      DO 10 L = 1,NPTS
   10 U(1,L) = Y(L)
      T = Y(NEQ)
      CALL PDEF1(X,T,U,NPDE,NPTS,NC,FEVAL,GEVAL,BNDRY,A,B,C,D,E,F,G)
      DO 20 L = 1,NPTS
   20 Y(L) = U(1,L)
      Y(NEQ) = 1.
      RETURN
      END
C
      SUBROUTINE BNDRY(T,ALFA,BETA,GAMMA,U,NPDE,LEFT)
      DIMENSION ALFA(NPDE),BETA(NPDE),GAMMA(NPDE),U(NPDE)
      LOGICAL LEFT
      IF (.NOT. LEFT) GOTO 10
      ALFA(1) = 0.
      BETA(1) = 1.
      GAMMA(1) = 0
      RETURN
   10 ALFA(1) = 1.
      BETA(1) = 0.
      GAMMA(1) = 1.
      RETURN
      END
C
      SUBROUTINE FEVAL(X,T,U,UX,F,NPDE)
      DIMENSION U(NPDE),UX(NPDE),F(NPDE)
      F(1) = UX(1)
      IF (X .LT. 0.5) F(1) = UX(1)*5.
      RETURN
      END
C
      SUBROUTINE GEVAL(X,T,U,UX,G,NPDE)
      DIMENSION U(NPDE),UX(NPDE),G(NPDE)
      G(1) = EXP(U(1))
      IF (X .LT. 0.5) G(1) = 1000.*G(1)
      RETURN
      END
```

PROBLEM C

```
      PROGRAM PDE(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      DIMENSION U(22),U1(22),U2(22),UOUT(22),DU(22),DU1(22),
     * SIGMA(2),INFO(15)
      EXTERNAL DER
      COMMON /ALL/ NPDE,NPTS,NC,X(21)
C
      NPTS = 21
      NPDE = 1
      NC = 1
      NEQ = NPDE*NPTS + 1
C
C         DEFINITION OF GRID AND INITIALIZATION OF VALUES
C
      DO 10 I = 1,NPTS
      X(I) = FLOAT(I - 1)/FLOAT(NPTS - 1)
      U(I) = 600.
   10 CONTINUE
C
C         FURTHER INITIALIZATION OF THE PARAMETERS OF M3RK
C
      TOL = 1.0E-4
      INFO(1) = 0
      INFO(2) = 2
      INFO(3) = 20000
      T = 0.0
      U(NEQ) = T
      WRITE(6,2) 5H  T\X, (X(I),I = 1,21,4)
    2 FORMAT(1H ,A5,6F8.2)
      WRITE(6,1)
    1 FORMAT(1H )
      DO 90 KL = 1,7
      IF (KL .EQ. 1) TE = 0.05
      IF (KL .EQ. 2) TE = 0.10
      IF (KL .EQ. 3) TE = 0.50
      IF (KL .EQ. 4) TE = 1.0
      IF (KL .EQ. 5) TE = 2.5
      IF (KL .EQ. 6) TE = 5.0
      IF (KL .EQ. 7) TE = 7.5
      WRITE(6,1)
      CALL M3RK(T,TE,NEQ,H,HMIN,SIGMA,TOL,DER,
     * U,U1,U2,UOUT,DU,DU1,IFLAG,INFO)
      WRITE(6,4) TE,(UOUT(I),I = 1,21,4)
    4 FORMAT(1H ,F5.2,6F8.2)
   90 CONTINUE
      STOP
      END
```

```
      SUBROUTINE DER(NEQ,Y)
      DIMENSION Y(NEQ),U(1,21),A(1),B(1),C(1),D(1),E(1),F(1),G(1)
      EXTERNAL FEVAL,GEVAL,BNDRY
      COMMON /ALL/ NPDE,NPTS,NC,X(21)
      DO 10 L = 1,NPTS
   10 U(1,L) = Y(L)
      T = Y(NEQ)
      CALL PDEF1(X,T,U,NPDE,NPTS,NC,FEVAL,GEVAL,BNDRY,A,B,C,D,E,F,G)
      DO 20 L = 1,NPTS
   20 Y(L) = U(1,L)
      Y(NEQ) = 1.
      RETURN
      END
C
      SUBROUTINE BNDRY(T,ALFA,BETA,GAMMA,U,NPDE,LEFT)
      DIMENSION ALFA(NPDE),BETA(NPDE),GAMMA(NPDE),U(NPDE)
      LOGICAL LEFT
      ALFA(1) = 0.
      BETA(1) = 1.
      GAMMA(1) = 0.
      IF (.NOT. LEFT) GAMMA(1) = 1.72E-9*(6.25E+10 - U(1)**4)
      RETURN
      END
C
      SUBROUTINE FEVAL(X,T,U,UX,F,NPDE)
      DIMENSION U(NPDE),UX(NPDE),F(NPDE)
      F(1) = UX(1)
      RETURN
      END
C
      SUBROUTINE GEVAL(X,T,U,UX,G,NPDE)
      DIMENSION U(NPDE),UX(NPDE),G(NPDE)
      G(1) = 0.
      RETURN
      END
```

PROBLEM D

```
      PROGRAM PDE(OUTPUT,TAPE6=OUTPUT)
      DIMENSION U(75),U1(75),U2(75),UOUT(75),DU(75),DU1(75),
     * SIGMA(2),INFO(15)
      EXTERNAL DER
      COMMON /ALL/ NPDE,NPTS,X(37)
C
      NPTS = 37
      NPDE = 2
      NEQ = NPDE*NPTS + 1
C
C        DEFINITION OF GRID AND INITIALIZATION OF VALUES
C
      DO 10 I = 1,NPTS
      IF (I .LT. 11) X(I) = FLOAT(I - 1)*.01
      IF (I .GE. 11 .AND. I .LT. 27 ) X(I) = FLOAT(I - 11)*.05 + 0.1
      IF (I .GE. 27) X(I) = FLOAT(I - 27)*.01 + 0.9
      U(I) = 1.
      U(I + NPTS) = 0.
   10 CONTINUE
C
C        FURTHER INITIALIZATION OF THE PARAMETERS OF M3RK
C
      TOL = 1.0E-4
      INFO(1) = 0
      INFO(2) = 2
      INFO(3) = 20000
      T = 0.0
      U(NEQ) = T
      WRITE(6,2) 4H T\X, X(1),(X(I),I = 11,27,2),X(37)
    2 FORMAT(1H ,A4,10F6.3,F7.3)
      WRITE(6,1)
    1 FORMAT(1H )
      DO 90 KL = 1,10
      TE = (KL - 1.)*0.5
      IF (KL .EQ. 1) TE = 0.1
      WRITE(6,1)
      CALL M3RK(T,TE,NEQ,H,HMIN,SIGMA,TOL,DER,
     * U,U1,U2,UOUT,DU,DU1,IFLAG,INFO)
      WRITE(6,4) TE,UOUT(1),(UOUT(I),I = 11,27,2),U(37)
      WRITE(6,5)   UOUT(38),(UOUT(I),I = 48,64,2),U(74)
    4 FORMAT(1H ,F4.1,10F6.3,F7.3)
    5 FORMAT(1H ,4X,10F6.3,F7.3)
   90 CONTINUE
      STOP
      END
```

```
      SUBROUTINE DER(NEQ,Y)
      DIMENSION Y(NEQ),U(2,41),A(2),B(2),C(2),D(2),E(2),F(2),G(2),
     * P(2),Q(2),R(2),S(2)
      EXTERNAL FEVAL,GEVAL,BNDRY
      COMMON /ALL/ NPDE,NPTS,X(37)
C
      DO 10 L = 1,NPTS
      U(1,L) = Y(L)
      U(2,L) = Y(L+NPTS)
   10 CONTINUE
      T = Y(NEQ)
      CALL PDEF2(X,T,U,NPDE,NPTS,FEVAL,GEVAL,BNDRY,
     *     A,B,C,D,E,F,G,P,Q,R,S)
      DO 20 L = 1,NPTS
      Y(L) = U(1,L)
      Y(L+NPTS) = U(2,L)
   20 CONTINUE
      Y(NEQ) = 1.
      RETURN
      END
C
      SUBROUTINE BNDRY(T,ALFA,BETA,GAMMA,U,NPDE,LEFT)
      DIMENSION ALFA(NPDE),BETA(NPDE),GAMMA(NPDE),U(NPDE)
      LOGICAL LEFT
      IF (.NOT. LEFT) GOTO 10
      ALFA(1) = 0.
      BETA(1) = 1.
      GAMMA(1) = 0.
      ALFA(2) = 1.
      BETA(2) = 0.
      GAMMA(2) = 0.
      RETURN
   10 ALFA(1) = 1.
      BETA(1) = 0.
      GAMMA(1) = 1.0
      ALFA(2) = 0.
      BETA(2) = 1.
      GAMMA(2) = 0.
      RETURN
      END
C
      SUBROUTINE FEVAL(X,T,U,UX,F,NPDE)
      DIMENSION U(NPDE),UX(NPDE),F(NPDE)
      F(1) = UX(1)*0.024
      F(2) = UX(2)*0.17
      RETURN
      END
C
      SUBROUTINE GEVAL(X,T,U,UX,G,NPDE)
      DIMENSION U(NPDE),UX(NPDE),G(NPDE)
      P = EXP(5.73*(U(1) - U(2)))
      G(1) = - P + 1./P**2
      G(2) = - G(1)
      RETURN
      END
```

PROBLEM E

```
      PROGRAM PDE(OUTPUT,TAPE6=OUTPUT)
      DIMENSION U(12),U1(12),U2(12),UOUT(12),DU(12),DU1(12),
     * SIGMA(2),INFO(15)
      EXTERNAL DER
      COMMON /ALL/ NPDE,NPTS,X(11)
C
      NPTS = 11
      NPDE = 1
      NEQ = NPDE*NPTS + 1
C
C          DEFINITION OF GRID AND INITIALIZATION OF VALUES
C
      DO 10 I = 1,NPTS
      X(I) = FLOAT(I - 1)/FLOAT(NPTS - 1)
      U(I) = 100.
   10 CONTINUE
      U(NPTS) = 50.
C
C          FURTHER INITIALIZATION OF THE PARAMETERS OF M3RK
C
      TOL = 1.0E-4
      INFO(1) = 0
      INFO(2) = 2
      INFO(3) = 20000
      T = 0.0
      U(NEQ) = T
      WRITE(6,2) 5H  T\X, (X(I),I = 1,11,2)
    2 FORMAT(1H ,A5,F5.1,5F8.4)
      WRITE(6,1)
    1 FORMAT(1H )
      DO 90 KL = 1,12
      TE = (KL - 1.)*0.005
      IF (KL .EQ. 1) TE = 0.001
      IF (KL .EQ. 12) TE = 0.1
      WRITE(6,1)
      CALL M3RK(T,TE,NEQ,H,HMIN,SIGMA,TOL,DER,
     * U,U1,U2,UOUT,DU,DU1,IFLAG,INFO)
      WRITE(6,4) TE,(UOUT(I),I = 1,11,2)
    4 FORMAT(1H ,F5.3,F5.1,5F8.4)
   90 CONTINUE
      STOP
      END
```

```
      SUBROUTINE DER(NEQ,Y)
      DIMENSION Y(NEQ),U(1,11),A(1),B(1),C(1),D(1),E(1),F(1),G(1),
     * P(1),Q(1),R(1),S(1)
      EXTERNAL FEVAL,GEVAL,BNDRY
      COMMON /ALL/ NPDE,NPTS,X(11)
      DO 10 L = 1,NPTS
   10 U(1,L) = Y(L)
      T = Y(NEQ)
      CALL PDEF2(X,T,U,NPDE,NPTS,FEVAL,GEVAL,BNDRY,
     *    A,B,C,D,E,F,G,P,Q,R,S)
      DO 20 L = 1,NPTS
   20 Y(L) = U(1,L)
      Y(NEQ) = 1.
      RETURN
      END
C
      SUBROUTINE BNDRY(T,ALFA,BETA,GAMMA,U,NPDE,LEFT)
      DIMENSION ALFA(NPDE),BETA(NPDE),GAMMA(NPDE),U(NPDE)
      LOGICAL LEFT
      IF (.NOT. LEFT) GOTO 10
      ALFA(1) = 1.
      BETA(1) = 0.
      GAMMA(1) = 50.
      RETURN
   10 ALFA(1) = 0.
      BETA(1) = 1.
      GAMMA(1) = 1. - SIN(U(1))
      RETURN
      END
C
      SUBROUTINE FEVAL(X,T,U,UX,F,NPDE)
      DIMENSION U(NPDE),UX(NPDE),F(NPDE)
      F(1) = U(1)*UX(1)
      RETURN
      END
C
      SUBROUTINE GEVAL(X,T,U,UX,G,NPDE)
      DIMENSION U(NPDE),UX(NPDE),G(NPDE)
      G(1) = - U(1)**2
      RETURN
      END
```

LITERATURE

[1] BAKKER, M., *On the numerical solution of parabolic equations in one space variable by the continuous time Galerkin method*, submitted elsewhere.

[2] ———— , *Galerkin methods in circular and spherical regions*, report NW 50/77, Mathematisch Centrum, Amsterdam.

[3] BUS, J.C.P. (ed.), *Colloquium numerieke programmatuur* (Dutch), MC Syllabus 29, 1b, Mathematisch Centrum, Amsterdam, 1976.

[4] CARVER, M.B., *A FORTRAN-oriented simulation system for the general solution of partial differential equations*, Proceedings of the 1973 summer computer simulation conference, Montreal.

[5] CSENDES, Z.J., *DECL - A computer language for the solution of arbitrary partial differential equations*, AICA.

[6] DOUGLAS, J. & T. DUPONT, *Galerkin methods for the two-point boundary problem using continous, piecewise polynomial spaces*, Num. Mat. 22 (1974), pp.99-109.

[7] GARY, I. & R. HELGASON, *An extension of FORTRAN containing finite difference operators*, Software - practice and experience 2 (1972), pp.321-326.

[8] ———— & ———— , *PDELAN users manual version I*, National Center for atmospheric research, Boulder, Colorado, 1975.

[9] HEMKER, P.W., *Galerkin's method and Lobatto points*, report NW 24/75, Mathematisch Centrum, Amsterdam.

[10] ———— , *A numerical study of stiff two-point boundary problems*, MC tract 80, Mathematisch Centrum, Amsterdam, 1977.

[11] HINDMARSCH, A.C., *Gear: ordinary differential equation system solver*, report UCID - 30001 Rev. 2, Lawlerence Livermore Laboratory, Livermore, California, 1972.

[12] ———— , *GEARB: solution of ordinary differential equations having banded Jacobian*, report UCID - 30059, Lawlerence Livermore Laboratory, Livermore, California, 1973.

[13] NILSEN, R.N. & W.J. KARPLUS, *Continuous system simulation languages, A state of the art survey*, Annales de l'AICA, 1 (1974).

[14] POLAK, S.J. & J. SCHROOTEN, *Preliminary Teddy 2 user manual*, Philips-ISA-UDV-DSA-SCA/SP/75/024/MW.

[15] RAVIART, P.A., *The use of numerical integration in finite element methods for solving parabolic equations*, From: J.J.H. MILLER (ed.), Topics in numerical analysis, Academic Press, London-New York, 1973.

[16] RYDER, B.G., *The PFORT verifier*, Software practice and experience 4 (1974), pp.359-378.

[17] SINCOVEC, R.F. & N.K. MADSEN, *Software for nonlinear partial differential equations*, ACM transactions on mathematical software 3 (1975), pp.232-260.

[18] ——— , *General software for partial differential equations*, to appear.

[19] STRANG, G., *Variational crimes in the finite element method*, Maryland Symposium, 1972.

[20] STRANG, G. & J.G. FIX, *An analysis of the finite element method*, Prentice-Hall, Englewood Cliffs, N.J., 1973.

[21] TE RIELE, H.J.J., *Colloquium numerieke programatuur* (Dutch), MC Syllabus 29.2, Mathematisch Centrum, Amsterdam, 1977.

[22] VERWER, J.G., *An implementation of a class of stabilized, explicit methods for the time integration of parabolic equations*, submitted elsewhere.