PDEONE

User Guide

PDEONE extends class DISCO with facilities for simulation of systems involving partial differential equations. PDEONE is a subclass of class DISCO; therefore, combined continuous and discrete systems may be modelled.

PDEONE is designed mainly for the solution of nonlinear parabolic initial value problems in one space dimension within a limited interval, XLEFT<=X<=XRIGHT. Solution of hyperbolic and elliptic problems, among others, may also be attempted. PDEONE is sufficiently general to solve a wide class of physically realistic problems.

PDEONE can solve problems characterised by the following equation:

$$\frac{dU(T,X)}{dT} = f(T,X,U,\frac{dU(T,X)}{dX}),\frac{1}{X^C}*\frac{d}{dX}(X^C*D(T,X,U)*\frac{dU(T,X)}{dX})$$

This equation represents the coupled system of partial differential equations, where U is a vector and XLEFT<X<XRIGHT. At the left boundary, X=XLEFT, the boundary condition is characterised as follows:

$$B1LEFT*U(T,X) + B2LEFT*\frac{dU(T,X)}{dX} = B3LEFT$$

At the right boundary, X=XRIGHT, the boundary condition is characterised as follows:

$$B1RIGHT*U(T,X) + B2RIGHT*\frac{dU(T,X)}{dX} = B3RIGHT$$

The initial condition (at T=T0) is characterised as follows:

$$U(T0,X) = g(X)$$

The 'diffusion coefficient', D, in the equation above is a function of T, X, and U. C is 0, 1, or 2, depending on whether the problem is Cartesian, cylindrical, or spherical, respectively. The 'left boundary coefficients', B1LEFT, B2LEFT and B3LEFT, may be functions of T, X and U. If B1LEFT and B2LEFT are both zero, the following condition is used at the left boundary:

$$\frac{dU(T,XLEFT)}{dT} = B3LEFT$$

The same is true for the 'right boundary coefficients', B1RIGHT, B2RIGHT and B3RIGHT.

In PDEONE the equations are solved using the so-called 'method of lines' in which the spatial variable, X, is "discretized", while the time variable, T=TIME, remains "continuous". Roughly speaking, the method can be described as follows. If one has a time dependent partial differential equation and discretizes the spatial variable, a semi-discrete approximating system of 'ordinary' differential equations results. One then uses an ordinary differential equation integration method, for example the Runge-Kutta-England method, for solving the resulting equations to obtain numerical approximations to the original partial differential equation.

In PDEONE the user specifies a spatial mesh consisting of points between XLEFT and XRIGHT. The points need not be uniformly spaced. The rest of the job is managed by PDEONE, that is, the splitting of the partial differential equations into systems of approximating ordinary differential equations and the numerical solution of these resulting equations.

The most important attributes of PDEONE are shown in the class outline below.

```
DISCO class PDEONE;
begin
  class PDEVARIABLE(XLEFT,XRIGHT,NPOINTS);
  real XLEFT,XRIGHT; integer NPOINTS;
  virtual: real procedure RATE,
                          INITIALSTATE,
                          B1LEFT,B2LEFT,B3LEFT,
                          B1RIGHT,B2RIGHT,B3RIGHT;
  begin
    real procedure RATE; ERROR("PDEVARIABLE: NO RATE SPECIFIED");
    real procedure INITIALSTATE;;
    real procedure B1LEFT;;  real procedure B1RIGHT;;
    real procedure B2LEFT;;  real procedure B2RIGHT;;
    real procedure B3LEFT;;  real procedure B3RIGHT;;

    integer C;
    ref(HEAD) POINTS;

    real procedure STATE; ... ;
    real procedure DX; DX:=... ;
    real procedure DDX(D); name D; real D; DDX:=... ;
    real procedure D2X; D2X:=... ;

    procedure START; ... ;
    procedure STOP; ... ;
    Boolean procedure ACTIVE; ACTIVE:=... ;
  end *** PDEVARIABLE ***;

  VARIABLE class POINT(X); real X;;

  real X;
end *** PDEONE ***;
```

Class PDEVARIABLE is used for the description of state-variables that vary according to partial differential equations. The equations are expressed by the user in subclasses of class PDEVARIABLE using the virtual procedure RATE. The two independent variables are denoted by TIME and X.

The initial and boundary conditions are specified in the virtual procedures INITIALSTATE, B1LEFT, B2LEFT, B3LEFT, B1RIGHT, B2RIGHT and B3RIGHT. The parameters XLEFT and XRIGHT denote the left and right boundary point, respectively.

The parameter NPOINTS is the number of mesh points desired at the start of the simulation. When the PDEVARIABLE-object is generated, NPOINTS equidistant mesh points are automatically placed between XLEFT and XRIGHT. These points are represented by objects of class POINT, a subclass of class VARIABLE, and are held in a list, a HEAD-object called POINTS, belonging to the PDEVARIABLE-object in question. POINTS.FIRST is the left boundary point, XLEFT, whereas POINTS.LAST is the right boundary point, XRIGHT.

The partial differential equation corresponding to a variable, say U, is expressed in the virtual procedure RATE, so that RATE computes the time derivative of U: $\frac{dU}{dT}$.

For this computation the procedures STATE, DX, DDX and D2X are provided as attributes of class PDEVARIABLE.

STATE returns the current value of the variable at the actual point, X: U(TIME,X).

DX computes an approximation to the spatial derivative of the variable: $\frac{dU}{dX}$.

DDX(D), where D is a real variable (name), computes an approximation to

$$\frac{1}{X^C} * \frac{d}{dX}(X^C * D * \frac{dU}{dX})$$

The integer attribute C of class PDEVARIABLE may be set to 0, 1, or 2, depending on whether the problem is Cartesian, cylindrical, or spherical, respectively. The parameter D of procedure DDX (the diffusion coefficient) may be a function of TIME, X and STATE.

D2X may be used to compute an approximation to the second order spatial derivative:

$$\frac{d^2U}{dX^2}$$

Note that D2X is equivalent to calling DDX(1) with C equal to zero; however, calling D2X is more efficient.

Initially a PDEVARIABLE-object is inactive, that is, its STATE is constant and equal to its INITIALSTATE at every mesh point. However, as soon its START-procedure is called, STATE will be changed "continuously" according to RATE as defined by the user. The object may be made inactive by calling its STOP-procedure.

An example

The use of PDEONE is best illustrated through an example. Consider the following partial differential equation:

$$\frac{dU}{dT} = -U * \frac{dU}{dX} + X * \frac{d}{dX}(U * \frac{dU}{dX}) + T$$

for X between 0 and 1, and for T beween 0 and 1.2. Further, assume the initial condition

```
U(0,X) = X
```

and the non-linear boundary conditions

```
U(T,XLEFT) = U(T,0) = 50
```

$$\frac{dU(T,XRIGHT)}{dT} = \frac{dU(T,1)}{dT} = 1 - \sin(U(T,1))$$

A complete PDEONE-program for solving this problem is given below.

```
 1.  PDEONE
 2.  begin
 3.    PDEVARIABLE class UVAR;
 4.    begin
 5.      real procedure RATE; RATE:=-STATE*DX+X*DDX(STATE)+TIME;
 6.      real procedure INITIALSTATE; INITIALSTATE:=X;
 7.      real procedure B1LEFT; B1LEFT:=1;
 8.      real procedure B3LEFT; B3LEFT:=50;
 9.      real procedure B2RIGHT; B2RIGHT:=1;
10.      real procedure B3RIGHT; B3RIGHT:=1-SIN(STATE);
11.    end *** UVAR ***;

12.    ref(PDEVARIABLE) U;
13.    MAXABSERROR:=0.00001; DTMIN:=0.000001; DTMAX:=0.1;
14.    U:-new UVAR(0,1,21); U.START;
15.    HOLD(1.2);
16.  end;
```

Comments:

Lines

3-11: Class UVAR describes the variable U.

5: The partial differential equation is expressed in the virtual procedure RATE.

6: Procedure INITITALSTATE defines the initial value of U.

7-10: These lines specify the boundary conditions. Note that since B2LEFT and B1RIGHT have not been specified by the user, they are zero.

11: U is generated and STARTed. The spatial interval from XLEFT=0 to XRIGHT=1 is discretized using 21 equidistant mesh points.

12: The equation is solved for 0<=TIME<=1.2.

The program does not produce any output. If, for example, we were interested in a plot of the solution at the end of the simulation (at TIME=1.2), this could be achieved by inserting the following lines after line 15:

```
begin
  ref(POINT) P;
  P:-U.POINTS.FIRST;
  while P=/=none do
  begin PLOT('*',P.X,P.STATE); P:-P.SUC; end;
end;
```

In this example the Runge-Kutta-England integration method was used. This method is not always adequate, since the problem can become "stiff" as the number of spatial mesh points increases, so that very small time-steps are required to maintain stability and accuracy. Therefore it is recommended that the version of DISCO in file 38 be used. This version includes a special method for integrating stiff differential equations (ref.19). This method is selected merely by setting the Boolean variable STIFF to true.

Since PDEONE is a subclass of DISCO, it is possible to simulate complex systems described by a mixture of partial differential equations, ordinary differential equations and discrete events. The example program from file 36 illustrates these abilities of PDEONE.

The implementation of PDEONE

Class PDEONE is on the whole a SIMULA version of the FORTRAN subroutine PDEONE by Sincovec and Madsen (ref. 1). In the implementation of the class it has been emphasised that new concepts introduced for describing partial differential equations fit into the philosophy of DISCO. Efficiency has also played a role.

PDEONE uses centred finite differences for approximating the spatial derivatives. Let X(1), X(2), ... , X(NPOINTS) denote the user specified sequence of mesh points where

```
XLEFT=X(1) < X(2) < ... < X(NPOINTS)=XRIGHT
```

and let U(j) denote the variable corresponding to the j'th partial differential equation.

The approximation formulae used at the non-boundary points, XLEFT<X<XRIGHT, are shown below.

$$\frac{dU(j)(T,X)}{dX}$$

is approximated by

$$\frac{U(j)(T,X(i+1))-U(j)(T,X(i-1))}{X(i+1)-X(i-1)}$$

and

$$\frac{1}{X^c}*\frac{d}{dX}(X^c*D(T,X,U)*\frac{dU(T,X)}{dX})$$

is approximated by

$$\frac{C+1}{(X(i+1/2)^{c+1}-X(i-1/2)^{c+1}}*$$

$$(\frac{X(i+1/2)^c*D(T,X(i+1/2),U(i+1/2))*(U(j)(T,X(i+1))-U(j)(T,X(i)))}{X(i+1)-X(i)}$$

-

$$\frac{X(i-1/2)^c*D(T,X(i-1/2),U(i-1/2))*(U(j)(T,X(i-1))-U(j)(T,X(i)))}{X(i)-X(i-1)})$$

where the following notation is used:

X(i+1/2) denotes (X(i+1)+X(i))/2,
X(i-1/2) denotes (X(i-1)+X(i))/2,

U(i+1/2) denotes the vector

```
((U(1)(T,X(i+1)+U(1)(T,X(i))/2,
 (U(2)(T,X(i+1)+U(2)(T,X(i))/2,
           .
           .
           .
 (U(NPOINTS)(T,X(i+1)+U(NPOINTS)(T,X(i))/2)
```

and U(i-1/2) denotes the vector

```
((U(1)(T,X(i-1)-U(1)(T,X(i))/2,
 (U(2)(T,X(i-1)-U(2)(T,X(i))/2,
           .
           .
           .
 (U(NPOINTS)(T,X(i-1)-U(NPOINTS)(T,X(i))/2)
```

At the boundary points, X=XLEFT and X=XRIGHT, the boundary conditions are applicable. For the left boundary, X=X(1)=XLEFT, the following difference approximations are used:

$$\frac{dU(j)(T,X)}{dX}$$

is approximated by

$$\frac{U(j)(T,X(2))-U(j(T,X(1))}{X(2)-X(1)} \quad \text{, if } B2LEFT = 0$$

$$\frac{B3LEFT-B1LEFT*U(j)(T,X(1)}{B2LEFT} \quad \text{if } B2LEFT \neq 0$$

And

$$\frac{1}{X^c} * \frac{d}{dX}(X^c * D(T,X,U) * \frac{dU(T,X)}{dX})$$

is approximated by

$$\frac{C+1}{(X(1+1/2)^{c+1} - X(1)^{c+1}} *$$

$$(X(1+1/2)^c * D(T,X(1+1/2,U(1+1/2)) -$$

$$X(1)^c * \frac{B3LEFT-B1LEFT*U(T,X(1)}{B2LEFT}) \quad \text{, if } B2LEFT \neq 0$$

$$\frac{C+1}{(X(1+1/2)^{c+1} - X(1)^{c+1}} *$$

$$(X(1+1/2)^c * D(T,X(1+1/2,U(1+1/2)) -$$

$$X(1)^c * D(T,X(1),U)) * \frac{U(T,X(2))-U(T,X(1)}{X(2)-X(1)}) \quad \text{, if } B2LEFT = 0$$

Note that when B1LEFT and B2LEFT both are zero, the behaviour at the left boundary is described by the equation

$$\frac{dU(T,X)}{dT} = B3LEFT$$

The difference approximations used for the right boundary are completely analogous.

If one wants to study the implementation of class PDEONE in detail, Sincovec and Madsen's paper (ref. 1) is recommended. It may be mentioned that derivatives for variables described by partial differential equations are computed by CONTINUOUS-objects hidden for the user. Actually, a PDEVARIABLE-object is itself a CONTINUOUS-object, which computes the derivatives, the RATEs, of all non-boundary mesh points. For the fulfilment of the boundary conditions each PDEVARIABLE-object has associated two CONTINUOUS-objects called LEFTBOUNDARY and RIGHTBOUNDARY which continuously evaluate the boundary coefficients, B1LEFT, B2LEFT, a.s.o., and, if required, update STATE or RATE of the corresponding boundary points. The PDEVARIABLE-objects have been given a comparatively low priority among the continuous processes, namely -1&10. The priorities of the continuous boundary processes are greater, namely zero. These priorities can be changed by the user, if wanted.

Reference:

1 Sincovec, R.F and Madsen, N.K.:
"Software for Nonlinear Partial Differential Equations".
ACM Transactions on Mathematical Software,
Vol. 1, No. 3, September 1975, pp. 232-260 and 261-263.