PDE2D GUI Example 3

Example 3 is a 1D time-dependent problem

$$\begin{aligned} U_t &= V \\ V_t &= -a\ V + c^2 U_{xx} \end{aligned}$$

(a=0.2,c=1) with initial conditions

$$\begin{aligned} U(x,0) &= max(0, 1 - |x - 5|) \\ V(x,0) &= 0 \end{aligned}$$

and boundary conditions $U_x = V_x = 0$ at $x = 0$ and $x = 10$. This is the damped wave equation

$$U_{tt} + a\ U_t = c^2 U_{xx}$$

reduced to a system of two equations by defining $V = U_t$.

Welcome to the PDE2D 9.4 GUI

Continue

The PDE2D GUI can be used to access the PDE2D collocation methods, which use cubic finite elements. These methods can handle 0D and 1D problems, and 2D and 3D problems in "a wide range of simple regions." For problems in complex 2D regions you should use the PDE2D Galerkin method, which is accessible only through the PDE2D Interactive Driver (type "pde2d [progname]").

If you want to see one of three prepared examples, select the example number below. If you ask to see an example, do not enter any data, just press the "Continue" buttons and look at the prepared input.

Show example number
(0 if none) 3

****Example 1. A 3D eigenvalue problem

Qxx + Qyy + Qzz - Q/sqrt(x^2+y^2+z^2) = lambda*Q

in half of a torus, with Q=0 on the curved surface of the torus and on one flat end, and dQ/dn+Q=0 (dQ/dn=normal derivative) on the other flat end. We will look for the eigenvalue closest to -1.15.

****Example 2. A 2D nonlinear steady-state problem

Uxx+Uyy = V
Vxx+Vyy = 1 + b*U^2

in an hour-glass shaped region. This is a fourth order elastic plate problem, with nonlinear loading, but reduced here to a system of two second order equations. On the left and right parabolic boundaries there are clamped boundary conditions, U=dU/dn=0, and on the flat top and bottom there are simply supported conditions, U=V=0.

****Example 3. A 1D time-dependent problem

Ut = V
Vt = -A*V + C^2*Uxx

with initial conditions U(x,0) = max(0,1-|x-5|), V(x,0)=0 and boundary conditions Ux=Vx=0 at x=0 and x=10. This is the damped wave equation Utt + A*Ut = C^2*Uxx, reduced to a system of two equations by defining V=Ut.

Continue

Begin Describing your PDE Problem

Dimension	1d	
Problem type	time-dependent	
Number of PDEs (NEQN)	2	
Precision	double	(Double strongly recommended)
Linear?		(If unsure, assume nonlinear)

Continue

Fortran expressions

You will be asked to input "Fortran expressions" to define your PDE coefficients, boundary conditions (BC) and other parameters. Fortran expressions are almost the same as MATLAB expressions, which use +, -, *, / to represent the four arithmetic operators, and recognize functions such as sin,cos,log,exp,sqrt,abs,max... One important difference is that A to the B power is represented as A**B rather than A^B. Note also that Fortran is not case-sensitive.

These expressions may reference the independent variables T,X,Y,Z,P1,P2,P3 (depending on the dimension of the problem). Some may also reference the unknowns and their first and second spatial (not time) derivatives; these are written using the names you assign, for example if you name an unknown "PHI", refer to it and its x-derivatives as PHI,PHIx,PHIxx. Your expressions may all reference the "parameters" you define, and also the parameter "pi" (3.14159...).

Expressions which cannot be written on a single line may reference Fortran functions, which you may define in a separate file. You will be prompted for the file name at the end of this session.

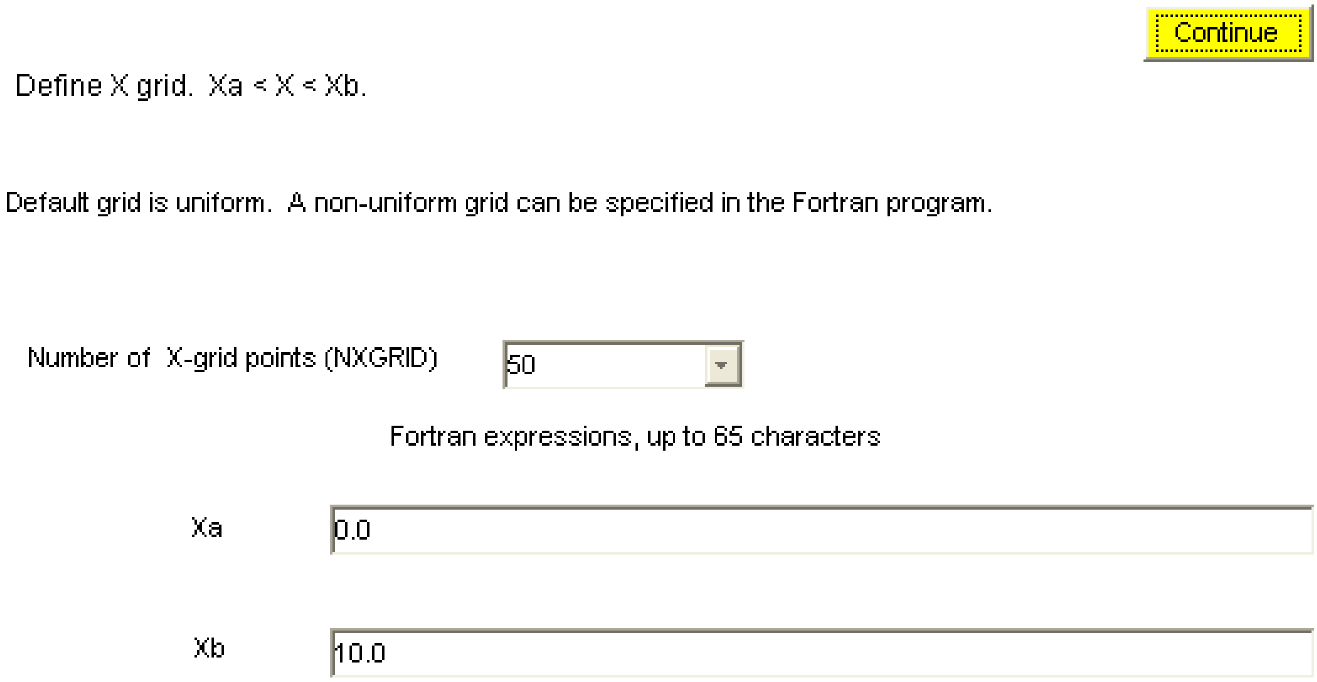
Once you have clicked "Continue" on a screen, you cannot return to correct or modify your expressions. However, you will find it easy to make such changes in the Fortran program created by the PDE2D GUI, which is well-documented. There you will also be able to modify the default values set by the GUI for numerous other options, or add further Fortran code.

Define parameters (constants)

Parameter names are 1-6 alphanumeric characters beginning with a letter. Names beginning with I,J,K,L,M,N must be integers. Parameters are constants, not functions of T,X,... Parameter definitions may reference previously-defined parameters.

Continue

Parameter Name		Value (Constant or Fortran expression, up to 65 characters)
A	=	0.2 ! (comment) A is damping coefficient
C	=	1.0 ! Wave velocity
	=	
	=	
	=	
	=	
	=	
	=	
	=	
	=	


Continue
Define X grid. Xa < X < Xb.
Default grid is uniform. A non-uniform grid can be specified in the Fortran program.
Number of X-grid points (NXGRID)
50
Fortran expressions, up to 65 characters
Xa
0.0
Xb
10.0

Continue

Define initial and final T values

Fortran expressions, up to 65 characters

TO 0.0

TF 10.0

Adaptive time step control?

Number of constant time steps (NSTEPS) 5000

If your time-dependent problem is linear AND all PDE and BC coefficients are independent of time except inhomogeneous terms, check this radio button; you may be able to dramatically cut the computer costs. But do not check it unless you are sure this is the case.

You will not be able to check this button if you request "adaptive time step control", or if you indicated earlier that your problem is nonlinear.

1D time-dependent problem

Continue

C11*dU_1/dT = F1 U_1 = U_10(X) at T=T0
C22*dU_2/dT = F2 U_2 = U_20(X)
: :

where the Cii and Fi are functions of T,X and the unknowns and their first and second derivatives.

Boundary conditions at X=Xa and X=Xb are:

G1 = 0
G2 = 0
:

where the Gi are functions of T,X and the unknowns and their first derivatives.

Continue

Name the unknowns

Names are 1-3 alphanumeric characters each, beginning with a letter A-H or O-Z.

Use these names in all future input expressions.

For example, if U_1 is named "U", then refer to this unknown and its derivatives as U,Ux,Uxx... in future Fortran expressions.

U_1 name	U
U_2 name	V
U_3 name	
U_4 name	
U_5 name	
U_6 name	
U_7 name	
U_8 name	

Integrals

Define any functions whose integrals you want to compute. They may be functions of the unknowns and their first and second derivatives, plus X,T.

Continue

Number of integrals desired (NINT) 1

Fortran expressions, up to 65 characters

Integral 1 `V**2 + C**2*Ux**2        ! Compute total energy`

Integral 2

Integral 3

Integral 4

Integral 5

Integral 6

Integral 7

Integral 8

Fortran expressions, up to 65 characters

Continue

C11	1
C22	1
C33	
C44	
C55	
C66	
C77	
C88	

F1	V
F2	-A*V + C**2*Uxx
F3	
F4	
F5	
F6	
F7	
F8	

Continue

Initial Conditions

Fortran expressions, up to 65 characters

U_10=U0	max(0.D0,1-abs(X-5))
U_20=V0	0
U_30	
U_40	
U_50	
U_60	
U_70	
U_80	

◯ Periodic BC?

For "no" BC, enter: None.

Define BCs at X=Xa

Fortran expressions, up to 65 characters

Continue

G1	Ux ! Ux=Vx=0 at both ends
G2	Vx
G3	
G4	
G5	
G6	
G7	
G8	

Define BCs at X=Xb

G1	Ux
G2	Vx
G3	
G4	
G5	
G6	
G7	
G8	

Continue

Fortran function file

If you referenced your own Fortran functions in any of your Fortran expressions, you may now supply the name of a file where these functions are defined (or will be when RUNPDE2D is executed). Remember to type functions and their arguments as "double precision" if they are double precision in the main program, and fixed-format must be used, so start lines in column 7 or later.

File name (if any)

Example Fortran function. The function below could be called (it would be referenced simply as "GB(P2,U,Unorm)") to specify boundary conditions on the perimeter (P1=B1) of a disk, if polar coordinates (X=p1*cos(p2), Y=p1*sin(p2)) are used for a 2D problem:

```
      function GB(P2,U,Unorm)
      double precision GB,P2,U,Unorm,pi
C234567 (parameters not available within functions)
      pi = 3.141592654d0
      if (P2 .le. pi/2) then
C                 for P2  < pi/2, specify BC U=1
         GB = U-1
      else
C                 for P2  > pi/2, specify BC Unorm=0
         GB = Unorm
      endif
      return
      end
```

Finish

You have finished defining your PDE problem

The PDE2D Interactive Driver will now be run using as input the "pde2d.in" file created by the PDE2D GUI, and a Fortran program will be created, which can be executed using RUNPDE2D. This program contains comments which make it easy for you to make minor corrections and modifications to your model--the input you provided during the GUI session will be clearly marked by comments "INPUT FROM GUI". There you will also be able to select many options not documented in the GUI, for example, it will be easy to modify this program to request nonuniform grids, to compute all eigenvalues of an eigenvalue problem, to compute boundary integrals of functions of the solution and its derivatives, or to add off-diagonal terms to your "C" matrix (for time-dependent problems) or "RHO" matrix (for eigenvalue problems).

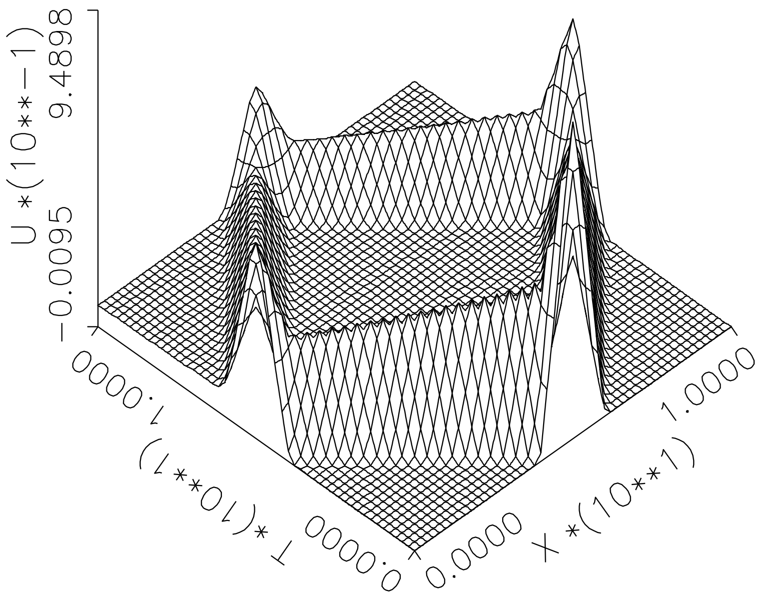
You do NOT need to be a Fortran programmer to make these modifications, and you do not need to work through another GUI session unless you have to change one of the options on the second GUI page. However, if you want to solve problems in complex 2D regions, or if you have more than eight PDEs, you will need to construct your program using the PDE2D Interactive Driver.

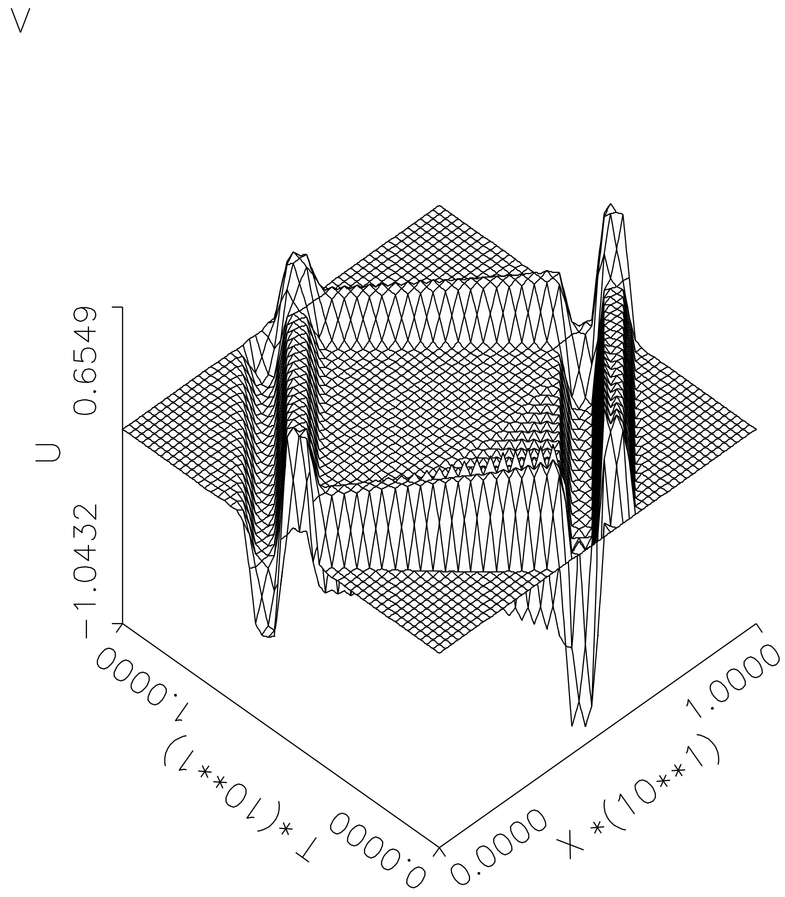
When you execute RUNPDE2D, plots of all unknowns will be made, but you can easily modify the program to postprocess the solution in many other ways, including using MATLAB plots (see subroutine POSTPR).

More information about PDE2D is contained in the book, "The Numerical Solution of Ordinary and Partial Differential Equations, second edition," Granville Sewell, John Wiley & Sons, 2005.

Integral output at $T = 10$ was 0.19975
Output graphs of U and V are shown below:

U




V
0.6549
U
−1.0432
1.0000
T *(10**1)
0.0000
0.0000
X *(10**1)
1.0000