UNIVERSIDAD DE LAS FUERZAS ARMADAS

ESPE EXTENSIÓN LATACUNGA

MATEMÁTICA SUPERIOR

EDO y EDP con MATLAB

Jácome José

IV MECATRÓNICA "A"

Latacunga-Ecuador - August 7, 2014

1 Resolución de Ecuaciones Diferenciales Parciales con Matlab

1.1 Comando pdepe

El comando *pdepe* se utiliza para resolver Ecuaciones Diferenciales Parciales con condiciones iniciales para ecuaciones de tipo parabólico y elíptico
Esta permite resolver EDP o sistemas de EDPs de una variable especial x y una variable temporal t. La forma normal de las EDPs,

```
u = pdepe(m,@pdefun,@icfun, @bcfun,x,t)
```

1.2 Forma General de las EDP para llevar a Matlab

La forma normal que se pueden resolver es de la forma:
$c(x,t,u,\frac{\partial u}{\partial x})\frac{\partial u}{\partial x} = \frac{1}{x^n}\frac{\partial}{\partial x}[x^m f(x,t,u,\frac{\partial u}{\partial x})] + s(x,t,u,\frac{\partial u}{\partial x})$
El Coeficiente c que multiplica la derivada con respecto a t es una matriz diagonal que se especifica como vector. La variable especial para reescalarse es tener: $0 < x < 1$.
El Parámetro m esta asociado a la geometría del problema: m = 0,1,2 corresponde a x como la coordenada radial en coordenadas cartesianas, cilíndricas o esféricas respectivamente.
Los terminos f, s se llaman flujo y fuente respectivamente.

1.3 Condiciones de Frontera

La condición de frontera en los extremos $x \in [0,1]$ se especifican en la forma:
$p(x,t,u) + q(x,t)f(x,t,u,\frac{\partial u}{\partial x})$
donde $x = a, b$. Es necesario especificar loas componentes de $p = (pl, pr)$ y de $q = (ql, qr)$ como funciones de t y de los valores de u en los extremos (ul, ur) en el caso de p. Note que las condiciones de frontera se especifican mediante el flujo de la derivada parcial $\frac{\partial u}{\partial x}$

1.4 Condiciones Iniciales

Se especifican en la forma de una función de x
$u = icfun(x)$

1.5 Llamada principal para resolver una EDP

```
u = pdepe(m,@pdefun,@icfun, @bcfun,x,t)
```

Sean $nx = length(x)$, $nt = lenght(t)$ las longitudes de las mallas x y t respectivamente, np es el numero de ecuaciones diferenciales, entonces:

```
a = x(1) < x(2) < ... <x(end) = b
0 = t(1) < t(2) < ... < t(end)
```

La solución u es un arreglo multidimensional de tamaño $nt*nx*np$; con $nx \geq 3$, por ejemplo $u(j,k,i)$ es la solución aproximada de la componente $i$en el punto $(t(j), x(k))$. Nótese que los índices j, k corresponden al orden de las variables (t, x).
Las funciones en el argumento de pdepe siguen la sintaxis:

1.6 Flujo

```
[c,f,s] = pdefun(x,t,u,dudx)
```

donde x, t son escalares, $u, dudx$ son vectores de dimensión np. La función regresa vectores c,f,s de de dimensión np correspondientes a la matriz diagonal $c(x, t, u, \dfrac{\partial u}{\partial x}/)$,flujo $f(x, t, u, \dfrac{\partial u}{\partial x})$ y fuente $s(x, t, u, \dfrac{\partial u}{\partial x})$. Por cada entrada del vector c nulo, la ecuación correspondiente es de tipo elíptica pero debe haber al menos una ecuación parabólica.

1.7 Condiciones de frontera

```
[pl,ql,pr,qr] = bcfun(xl,ul,xr,ur,t)
```

donde pl, ql son vectores columna de dimensión np correspondientes a la función $p(a, t, u)$ y a la matriz diagonal $q(a, t, u, \dfrac{\partial u}{\partial x})$. Note que sólo p puede depender de u. Análogamente pr, qr son vectores columna de dimensión np correspondientes a la función $p(a, t, u)$ y a la matriz diagonal $q(b, t, u, \dfrac{\partial u}{\partial x})$

1.8 Condición inicial

```
u = icfun(x)
```

donde x es un escalar, u es un vector de dimensión np.

1.9 Evaluación de la solución en puntos específicos de x

```
ui = u(:,:,i)
```

aproxima la i-ésmima componente de la solución en la malla tempo-espacial (t,x).
Para evaluar la solución y su derivada en puntos de la malla $xout$ distintos de la malla x se usa el comando.

```
[uout,duoutdx] = pdeval(m,x,ui,xout)
```

la salida son vectores $uout, duoutdx$ contienen las aproximaciones en la malla $xcout$.
Observe que se evalúa la derivada y no el flujo.

`Manejo de discontinuidades`

Se permiten discontinuidades de cos en x siempre que se incluyan en la malla x, pero el flujo deberá ser continuo. Es conveniente usar una malla más fina cerca de los puntos de discontinuidad, pero para m =2 (coordenadas esféricas) esto no es necesario.

1.10 Codigo de Matlab

```
%Resolucion de una EDP por Matlab
function pdex1
% 20 puntos de malla especial, 5 temporal
m = 0;
x = linspace(0,1,20);
t = linspace(0,2,5);
%Llamada a la rutina principal
sol = pdepe(m,@pdex1pde,@pdex1ic,@pdex1bc,x,t);
%Primera componente
u=sol(:,:,1);
%Grafico de superficie
figure;
surf(x,t,u);
title('Solución numérica con 20 puntos de malla.');
xlabel('Distancia x');
ylabel('Tiempo t');
figure;
surf(x,t,exp(-t)'*sin(pi*x));
title('Solución exacta con 20 puntos de malla. ');
xlabel('Distance x');
ylabel('Time t');
figure;
plot(x,u(end,:), 'o',x,exp(-t(end))*sin(pi*x));
title('Soluciones en t = 2. ');
legend('Numerica, 20 puntos de malla', 'Analitica',0);
xlabel('Distancia x'); ylabel('u(x,2)');
% Especificacion de la EDP
function [c,f,s] = pdex1pde(x,t,u,DuDx)
c = pi^2;
f = DuDx;
s = 0;
% Especificación de la condición inicial
function u0 = pdex1ic(x)
u0 = sin(pi*x);
% Especificación de condiciones de frontera
function [pl,ql,pr,qr] = pdex1bc(xl,ul,xr,ur,t)
pl = ul;
ql = 0;
pr = pi * exp(-t);
qr = 1;
```

1.11 Capturas de Pantalla