Rezolvitorul MATLAB pdepe

Rezolvitorul MATLAB `pdepe` rezolvă ecuaţii cu derivate parţiale simple, de tip parabolic sau eliptic în care necunoscuta $u(x,t)$ este o funcţie vectorială ce depinde de o variabilă spaţială scalară x şi o variabilă temporală scalară t. Forma generală a ecuaţiilor acceptate de rezolvitor este

$$c\left(x,t,u,\frac{\partial u}{\partial t}\right)\frac{\partial u}{\partial t}=x^{-m}\frac{\partial}{\partial x}\left(x^m f\left(x,t,u,\frac{\partial u}{\partial t}\right)\right)+s\left(x,t,u,\frac{\partial u}{\partial t}\right),$$

unde $x\in[a,b]$, $t\in[t_0,t_f]$. Parametrul întreg m specifică tipul de simetrie 0 – slab?, 1 – cilindrică, 2 – sferică. Funcţia c este dată printr-o matrice diagonală, f reprezintă fluxul, iar s sursa. Toate trei sunt funcţii vectoriale.

Condiţiile iniţiale se dau astfel: pentru $x\in[a,b]$ şi $t=t_0$ soluţia trebuie să îndeplinească condiţia $u(x,t_0)=u_0(x)$, unde u_0 este o funcţie dată. Soluţia trebuie să satisfacă condiţiile pe frontieră

$$p_a(x,t,u)+q_a(x,t)f\left(x,t,u,\frac{\partial u}{\partial x}\right)=0,\text{ pentru } x=a, t\in[t_0,t_f]$$

$$p_b(x,t,u)+q_b(x,t)f\left(x,t,u,\frac{\partial u}{\partial x}\right)=0,\text{ pentru } x=b, t\in[t_0,t_f],$$

unde p_a, q_a, p_b, q_b sunt funcţii date. Se pot impune şi alte tipuri de condiţii, vezi `doc pdepe` pentru detalii.

Apelul la `pdepe` are forma generală

```
sol = pdepe(m, @pdefun, @pdeic, xmesh, tspan, options)
```

Funcţia `pdefun` are forma

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

Ea acceptă la intrare variabilele de spaţiu şi timp şi vectorii `u` şi `DuDx`, care aproximează soluţia *u* şi derivata parţială $\partial u/\partial x$ şi returnează matricea `c`, funcţia de flux `f` şi functia sursă `s`. Condiţia iniţială se specifică prin funcţia pdeic, care are forma

```
function u0=pdeic(x)
```

Funcţia `pdebc` dă condiţiile pe frontieră şi are forma

```
function [pa, qa, pb, qb] = pdebc(xa, ua, xb, ub, t)
```

ea evalueaza p_a, q_a, p_b, q_b din condiţiile pe frontieră în `xa` = a şi `xb` = b. Vectorul `xmesh` din lista de argumente a lui `pdepe` este o mulţime de puncte din $[a,b]$, ordonată crescător, cu `xmesh(1)` = a şi `xmesh(end)` = b. El defineşte valorile variabilei spaţiale în care soluţia este aproximată. Algoritmul utilizează o discretizare de ordinul al doilea pentru variabila spaţială. Alegerea lui `xmesh` influenţează costul şi precizia soluţiei. Pe porţiunile unde soluţia oscilează mai repede în raport cu *x*, punctele se vor alege mai apropiate. Vectorul `tspan` este o mulţime de puncte din $[t_0,t_f]$, ordonată crescător, cu tspan(1) = t_0 şi tspan(end) = t_f. Integrarea ecuaţiei diferenţiale în raport cu t în `pdepe` se realizează cu `ode15s`; momentele de timp se determină dinamic, valorile în `tspan` se calculează prin interpolare şi aceasta are un impact mic asupra costului şi preciziei. Proprietăţile implicite ale rezolvitorului `ode15s` se pot modifica cu ajutorul argumentului opţional `options` care poate fi creat şi gestionat prin funcţia `odeset`.

Argumentul de ieşire sol este un tablu tridimensional, cu semnificaţia sol(j,k,i) este aproximarea celei de-a i-a componente a soluţiei u în punctul t = `tspan(j)` şi x = `xmesh(k)`. Funcţia de postprocesare pdeval permite calculul lui *u* şi $\partial u/\partial x$ în puncte care nu sunt în `xmesh`.

Vom exemplifica utilizarea lui pdepe cu ecuaţia Black-Scholes, utilizată pentru modelarea preţurilor derivatelor financiare. În trasformată şi adimensionalizată (P. Wilmott, 1995) şi cu parametrii din (Neftci, 2000), ea are forma

$$\frac{\partial u}{\partial t}=\frac{\partial^2 u}{\partial x^2}+(k-1)\frac{\partial u}{\partial x}-ku, x\in[a,b], t\in[t_0,t_f],$$

unde $k=r/(\sigma^2/2)$, $r=0.065$, $\sigma=0.8$, $a=\log\frac{2}{5}$, $b=\log\frac{7}{5}$, $t_0=0$, $t_f=5$, cu condiţiile iniţiale

$$u(x,0)=\max(e^x-1,0)$$

şi condiţiile pe frontieră

$$u(a,t)=0, u(b,t)=\frac{7-\exp(-kt)}{5}.$$

Problema are forma generală acceptată de `pdepe` cu $m = 1$,
$c(x,t,u) = 1, f\left(x,t,u,\frac{\partial u}{\partial t}\right) = \frac{\partial u}{\partial x}, s\left(x,t,u,\frac{\partial u}{\partial t}\right) = (k-1)\frac{\partial u}{\partial x} - ku$.
Condiţiile pe frontieră sunt $p(x,t,u) = u$ şi $q(x,t,u) = 0$ în $x = a$ şi $p(x,t,u) = u - (7 - 5exp(-kt))/5$ şi $q(x,t,u) = 0$ în $x = b$.
Dăm în continuare sursa, în fişierul `bs.m`.

```
function bs
%BS     Black-Scholes PDE.
%       Solves the transformed Black-Scholes equation.

m = 0;
r = 0.065;
sigma = 0.8;
k = r/(0.5*sigma^2);
a = log(2/5);
b = log(7/5);
t0 = 0;
tf = 5;

xmesh = linspace(a,b,40);
tspan = linspace(t0,tf,20);

sol = pdepe(m,@bspde,@bsic,@bsbc,xmesh,tspan);
u = sol(:,:,1);

mesh(xmesh,tspan,u)
xlabel('x','FontSize',12)
ylabel('t','FontSize',12)
zlabel('u','FontSize',12,'Rotation',0)

    function [c,f,s] = bspde(x,t,u,DuDx)
    %BSPDE  Black-Scholes PDE.
    c = 1;
    f = DuDx;
    s = (k-1)*DuDx-k*u;
    end

    function u0 = bsic(x)
    %BSIC   Initial condition at t = t0.
    u0 = max(exp(x)-1,0);
    end

    function [pa,qa,pb,qb] = bsbc(xa,ua,xb,ub,t)
    %BSBC   Boundary conditions at x = a and x = b.
    pa = ua;
    qa = 0;
    pb = ub - (7 - 5*exp(-k*t))/5;
    qb = 0;
    end

end
```

Am generat 40 de puncte după *x* şi 20 după *t* cu funcţia `linspace`. Funcţia imbricată `bspde` defineşte ecuaţia cu derivate parţiale, calculând `c`, `f` şi `s`, iar `bsic` implementează condiţiile iniţiale. Funcţia bsbc calculează `pa,qa,pb,qb` din condiţiile iniţiale.

```
bs
```

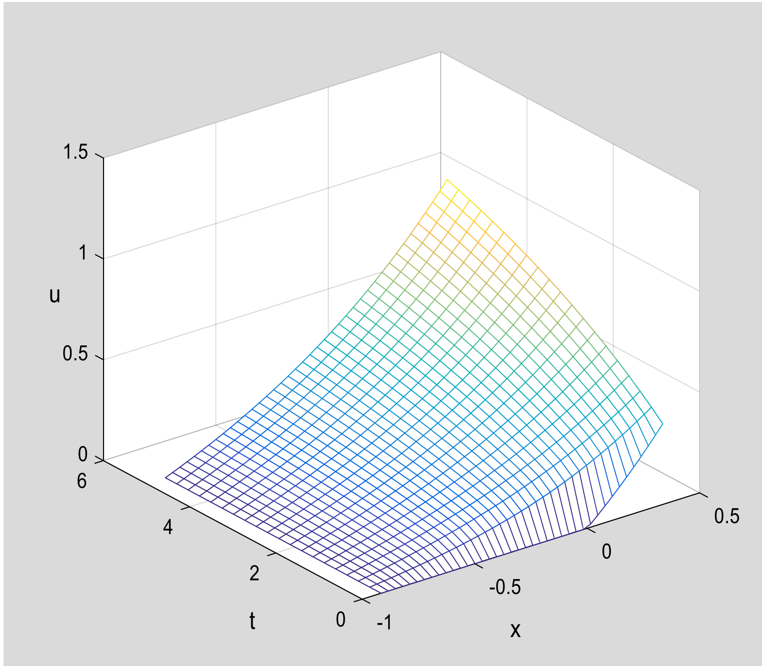

Figura 1. Ecuaţia Black-Scholes

Rezultatul execuţiei se vede în figura 1.

Exemplul următor este un sistem de două EDP de tip reacţie-difuzie, care provine din Biologia moleculară (D. S. Jones, 2003).

$$\frac{\partial u}{\partial t} = \frac{1}{2}\frac{\partial^2 u}{\partial x^2} + \frac{1}{1+v^2}$$

$$\frac{\partial v}{\partial t} = \frac{1}{2}\frac{\partial^2 v}{\partial x^2} + \frac{1}{1+u^2}$$

pentru $x \in [0,1], t \in [0,0.2]$. Condiţiile iniţiale sunt

$$u(x,0) = 1 + \frac{1}{2}\cos 2\pi x$$

$$v(x,0) = 1 - \frac{1}{2}\cos 2\pi x\,,$$

iar condiţiile pe frontieră

$$\frac{\partial u}{\partial x}(0,t) = \frac{\partial u}{\partial x}(1,t) = \frac{\partial v}{\partial x}(0,t) = \frac{\partial v}{\partial x}(1,t) = 0.$$

Ecuaţia scrisă în forma cerută de `pdepe`, cu $(u_1 \quad u_2)$ în loc de $(u \quad v)$, este

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \times \frac{\partial}{\partial t}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \frac{\partial}{\partial x}\begin{bmatrix} \frac{1}{2}\frac{\partial u_1}{\partial x} \\ \frac{1}{2}\frac{\partial u_2}{\partial x} \end{bmatrix} + \begin{bmatrix} \frac{1}{1+u_2^2} \\ \frac{1}{1+u_1^2} \end{bmatrix}.$$

Sursa MATLAB este dată în fişierul `mbiol.m`. Argumentele de ieşire `c`, `f`, `s`, `pa`, `qa`, `pb` şi `qb` ale subfuncţiilor `mbpde` şi `mbbc` sunt tablouri 2×1 deoarece avem un sistem de două ecuaţii. Energia

$$E(t) = \int_0^1 \left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2\right] dx$$

descreşte exponenţial către zero când $t \to \infty$. Pentru a verifica acest fapt numeric, am utilizat diferenţe finite şi cuadraturi. (Se putea utiliza `pdeval` pentru a aproxima $\partial u/\partial x$ şi $\partial v/\partial x$.)

```
function mbiol
%MBIOL   Reaction-diffusion system from mathematical biology.
%        Solves the PDE and tests the energy decay condition.

m = 0;
xmesh = linspace(0,1,15);
tspan = linspace(0,0.2,10);
sol = pdepe(m,@mbpde,@mbic,@mbbc,xmesh,tspan);
u1 = sol(:,:,1);
u2 = sol(:,:,2);

subplot(2,2,1)
surf(xmesh,tspan,u1)
xlabel('x','FontSize',12)
ylabel('t','FontSize',12)
title('u_1','FontSize',16)

subplot(222)
surf(xmesh,tspan,u2)
xlabel('x','FontSize',12)
ylabel('t','FontSize',12)
title('u_2','FontSize',16)

% Estimate energy integral.
dx = xmesh(2) - xmesh(1);  % Constant spacing.
energy = 0.5*sum( (diff(u1,1,2)).^2 + (diff(u2,1,2)).^2, 2)/dx;
subplot(2,2,[3:4])
plot(tspan',energy)
xlabel('t','FontSize',12)
title('Energy','FontSize',16)

% ----------------------- Subfunctions -----------------------
function [c,f,s] = mbpde(x,t,u,DuDx)
c = [1; 1];
f = DuDx/2;
s = [1/(1+u(2)^2); 1/(1+u(1)^2)];

function u0 = mbic(x);
u0 = [1+0.5*cos(2*pi*x); 1-0.5*cos(2*pi*x)];

function [pa,qa,pb,qb] = mbbc(xa,ua,xb,ub,t)
pa = [0; 0];
qa = [1; 1];
pb = [0; 0];
qb = [1; 1];

mbiol
```

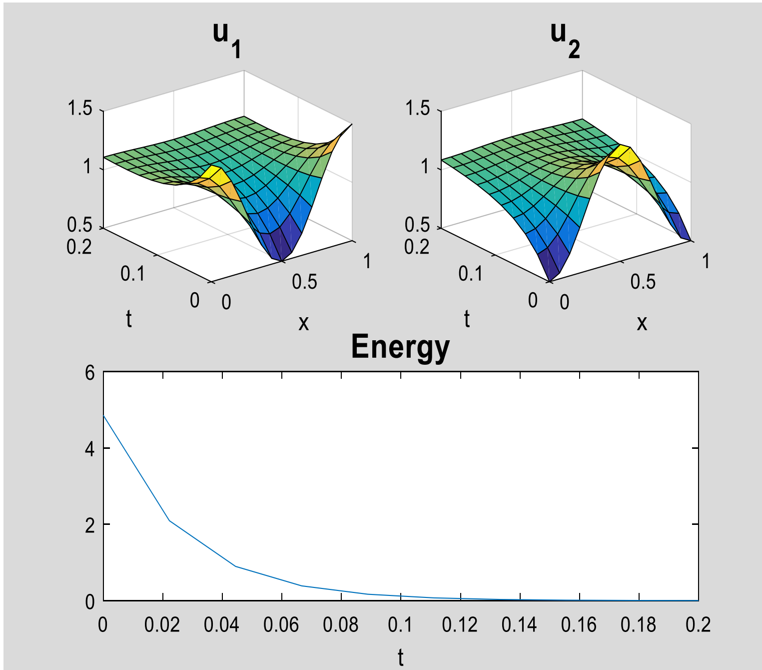

Figure 2 Soluţia unui sistem de tip reacţie difuzie cu `pdepe`

Bibliografie

D. S. Jones, S. D. (2003). *Differential Equations in Mathematical Biology.* Boca Raton, FL, USA: CRC Press.

Neftci, S. A. (2000). *An Introduction to the Mathematics of Financial Derivatives, 2nd ed.* . San Diego: Academic Press.

P. Wilmott, S. H. (1995). *The Mathematics of Financial Derivatives: A Stundent Introduction.* Cambridge: Cambridge University Press.