MAT 425: Homework 9 (04/06)

Ex 1. We consider the following boundary-value problem (BVP):

$$\begin{cases} -x^2y'' - 2xy' + 2y = -4x^2 \\ y(0) = 0\ ,\ y(1) = 0. \end{cases}$$

a) Find the variational problem associated with the BVP, i.e. find the operators $\mathcal{A}$ and L such that the solution y satisfies

$$\mathcal{A}(y, \varphi) = L(\varphi) \qquad \text{for any } \varphi \in C_0^2([0,1])$$

Hint: use $(x^2y')'$.

b) Let V a finite dimension space with a basis $\{\phi_i\}_{i=1..N}$.
Consider the matrix A and vector b defined as:

$$\begin{array}{rcll} A & = & [a_{ij}] & \text{with } a_{ij} = \mathcal{A}(\phi_i, \phi_j) \\ \mathbf{b} & = & [b_i] & \text{with } b_i = L(\phi_i). \end{array}$$

Suppose $y = \Sigma_{i=1}^N c_i\phi_i$ satisfies: $A\mathbf{c} = \mathbf{b}$. Show that:

$$\mathcal{A}(y, \varphi) = L(\varphi) \quad \text{for all } \varphi \in V$$

Hint: write $\varphi = \Sigma_{j=1}^N \alpha_j\phi_j$.

c) Fix $N = 10$ and consider the grid points: $x_i = i\Delta x$ with $\Delta x = \frac{1}{N+1}$.
We define the vector space V of piece-wise linear functions:

$$V = \{\varphi \in C_0^0([0,1]) : \quad \varphi(x) = ax + b \quad \text{for } x_i \le x \le x_{i+1}\}.$$

along with the basis function ϕ_i for $i = 1..N$:

$$\phi_i(x) = \begin{cases} \frac{x - x_{i-1}}{\Delta x} & \text{if } x_{i-1} \le x \le x_i \\ \frac{x_{i+1} - x}{\Delta x} & \text{if } x_i < x \le x_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

Find numerically the solution of the variational problem on V.

d) Find the accuracy of the method with respect to Δx.
Hint: the exact solution is $y(x) = x^2 - x$.

Ex 2. Use the finite difference method to solve the elliptic PDE:

$$\begin{cases} \partial_x^2 u + \partial_y^2 u = 4 & 0 < x < 1\ ,\ 0 < y < 2 \\ u(x,0) = x^2\ ,\ u(x,2) = (x-2)^2 & 0 \le x \le 1 \\ u(0,y) = y^2\ ,\ u(1,y) = (y-1)^2 & 0 \le y \le 2 \end{cases}$$

with $\Delta x = \Delta y = \frac{1}{10}$. Compare with the exact solution $u(x,y) = (x-y)^2$.

```
%% Solve the BVP using Finite-Element Method
%%   -(py')' + q(x) y = f(x)
%%   y(0)=y(1)=0
%%---- The Boundary Value Probem (BVP)
p = @(x) ...
q = @(x) ...
f = @(x) ...
%%--- numerical parameters: vector space V
N = ...
dx = 1/(N+1);
%%--- estimation matrix
A = zeros(N,N);
b = zeros(N,1);
%%-- construction of the matrix 'A' and vector 'b'
intX = linspace(0,1,10000);                % to estimate the integrals
for i=1:N
    % a) phi_i and phi_i'
    [phi_i,dphi_i] = Basis_Phi(intX,i,N);
    % b) b_i = L(phi_i)
    b(i) = trapz(intX,f(intX).*phi_i);
    % c) estimation of A(phi_i,phi_j)
    for j=1:N
        % phi_j and phi_j'
        [phi_j,dphi_j] = Basis_Phi(intX,j,N);
        % a_ij = A(phi_i,phi_j)
        A(i,j) = ...
    end
end
%%-------------------------------%%
%%---    Resolution system    ---%%
%%-------------------------------%%
c = A\b;
%%-- construction solution
%%  y = c(1)*phi_1 + ... + c(N)*phi_N
ySol = zeros(1,length(intX));
for i=1:N
    ...
end

%%-- plot
yExact = @(x) x.*(x-1);
plot(intX,ySol,intX,yExact(intX))
legend('Numerical Solution','Exact','location','southwest')
```

```
function [phi,dphi]=Basis_Phi(x, i,N)
  %% The piece-wise linear basis function
  %%
  %%                 |  (x-x_im)/dx      if    x_im < x <= x_i
  %%    phi(x)   =   |  (x_ip-x)/dx      if    x_i  < x <  x_ip
  %%                 |       0           otherwise
  %%
  %%                 |      1/dx      if    x_im < x <= x_i
  %%    phi'(x) =    |     -1/dx      if    x_i  < x <  x_ip
  %%                 |       0           otherwise

  %% init
  dx = 1/(N+1);
  x_im = (i-1)*dx;
  x_i  =     i*dx;
  x_ip = (i+1)*dx;

  %% value
  phi = (x-x_im)/dx.*((x-x_im)>0).*((x-x_i)<=0) ...
      + (x_ip-x)/dx.*((x-x_i)>0).*((x-x_ip)<0);
  dphi = 1/dx.*((x-x_im)>0).*((x-x_i)<=0) ...
      -1/dx.*((x-x_i)>0).*((x-x_ip)<0);

end
```

```
%% Solve the elliptic PDE
%%     u_xx + u_yy  =  f    in [a,b]x[c,d]
%%     u = g                at the frontier
%% the functions
f = @(x,y) 4;
g = @(x,y) (x-y)^2;
%% domain [a,b]x[c,d]
a = 0; b = 1;
c = 0; d = 2;
%% numerical parameters
n = 10; dx = (b-a)/n;
m = 20; dy = (d-c)/m;
%%--------------------------------%%
%%-----          FDM           -----%%
%%--------------------------------%%
lMax = (n-1)*(m-1);
A    = zeros(lMax,lMax);
vecB = zeros(lMax,1);
ij_to_l = @(i,j) i + (m-1-j)*(n-1);
for i=1:(n-1)
  for j=1:(m-1)
    %% init
    l    = ij_to_l(i,j);
    xi   = a+i*dx;
    yj   = c+j*dy;
    l_ip = ij_to_l(i+1,j);
    l_im = ij_to_l(i-1,j);
    l_jp = ij_to_l(i,j+1);
    l_jm = ij_to_l(i,j-1);
    %%---- the matrix A: stencil with 5 points
    %% diagonal
    A(l,l)  = -(2+2*dx^2/dy^2);
    vecB(l) = dx^2*f(xi,yj);
    %% extra diago: !! Danger border !!
    if (i~=1)
      A(l_im,l) = 1;
    else
      %% x_i-1=a (left)
      vecB(l) = vecB(l) - g(a,yj);
    end
    ...
    ...
  end
end
```

```
%%------------------------%%
%%---     Solution     ---%%
%%------------------------%%
vecU = ...
%%--reconstruction matrix solution
matU = zeros(n+1,m+1);
for i=0:n
  for j=0:m
    if (i==0 || i==n || j==0 || j==m)
      %% at the border
      xi = a+i*dx;
      yj = c+j*dy;
      ...
    else
      %% inside
      l = ij_to_l(i,j);
      ...
    end
  end
end

%%-- plot
x = linspace(a,b,n+1);
y = linspace(c,d,m+1);
[X,Y] = meshgrid(x,y);
surf(X,Y,matU')
xlabel('x')
ylabel('y')
title('The solution of the elliptic PDE')
```