```
function sem_MKD
% Metodou konecnych diferenci resime tuto soustavu
% y'_1 = cos(y_3)
% y'_2 = sin(y_3)
% y'_3 = ( cos(y_3) - sin(y_3)*abs( sin(y_3) ) )/y_4
% y'_4 = sin(y_3) - cos(y_3)*abs( cos(y_3) )

clc
% Okrajove podminky
ya = [0; 0; 0; 1];
yb = [0.85; 0.50; 1; 1];

% Inicializace
N = 100; % N je pocet uzlu site
h = 1/N; % Krok site
x = linspace(0,1,N+1);

X0 = (yb(1) - ya(1))*x(2:N) + ya(1);  % Vnitrni body site, n = 1,2,...,N-1
Y0 = (yb(2) - ya(2))*x(2:N) + ya(2);  % Vnitrni body site, n = 1,2,...,N-1
V0 = (yb(3) - ya(3))*x + ya(3);       % Odhad pro y_3
W0 = (yb(4) - ya(4))*x + ya(4);       % Odhad pro y_4
U = [X0'; Y0'; V0'; W0'];             % Pocatecni inicializace

A1 = sparse(2:N,1:N-1,-1,N,N-1) + sparse(1:N-1,1:N-1,1,N,N-1);
A2 = sparse(1:N,1:N,-1,N,N+1) + sparse(1:N,2:N+1,1,N,N+1);
% full(A1)

B = [A1, zeros(N,3*N+1);
     zeros(N,N-1),   A1, zeros(N,2*N+2);
     zeros(N,2*N-2), A2, zeros(N,N+1);
     zeros(N,3*N-1), A2];
% full(B)
% rank(full(B))
% det(B)
% nnz(B)
% spy(B)

tol = 1e-9;
i = 0;
pom = 0;
% for i = 1:20 % Newtonovo metoda
while norm(U-pom,inf) >= tol
        i = i + 1;
        pom = U;
        PHI = B*U - F(U,N,ya,yb);
        dPHI = jakobian(U,N,A1,A2);
    %         full(dPHI)
    %         rank(full(dPHI))
    %         c1 = condest(dPHI)
    %         c2 = cond(dPHI)
    %         figure
    %         spy(dPHI)
    %         nnz(dPHI)
        z = dPHI\(-PHI);
        U = U + z;
end
```

```
hold on
grid on
xlabel('t');
ylabel('\bf y');
title(['MKD, Pocet uzlu site : N = ',num2str(N),', Pocet iteraci : i = ',num2str(i)])

X = [ya(1); U(1:N-1); yb(1)];   % reseni y_1(t)
Y = [ya(2); U(N:2*N-2); yb(2)]; % reseni y_2(t)
    plot(X,Y,'k'); % vykresleni lana
    plot(x,X,'r'); % vykresleni y_1(t)
    plot(x,Y,'g'); %   --||--   y_2(t)

V = U(2*N-1:3*N-1); % reseni y_3(t)
    plot(x,V,'y');  % vykresleni y_3(t)
W = U(3*N:4*N);     % reseni y_4(t)
    plot(x,W,'m');  % vykresleni y_4(t)
legend('lano','y_1(t)','y_2(t)','y_3(t)','y_4(t)')


function p = F(U,N,ya,yb)
% Prava strana
h = 1/N;

V_n = U(2*N-1:3*N-2); % z U vybereme V_n az V_n-1
V_n1= U(2*N:3*N-1);   % z U vybereme V_n+1 az V_n
W_n = U(3*N:4*N-1);   % z U vybereme W_n az W_n-1
W_n1= U(3*N+1:4*N);   % z U vybereme W_n+1 az W_n

px = h*( cos(V_n) + cos(V_n1) )/2;
px(1) = px(1) + ya(1);
px(N) = px(N) - yb(1);

py = h*( sin(V_n) + sin(V_n1) )/2;
py(1) = py(1) + ya(2);
py(N) = py(N) - yb(2);

pom1 = cos(V_n) + cos(V_n1);
pom2 = ( sin(V_n) + sin(V_n1) ).*abs( sin(V_n) + sin(V_n1) )/2;
pv = h*( pom1 - pom2 )./( W_n + W_n1 );

pom3 = ( sin(V_n) + sin(V_n1) )/2;
pom4 = pom1.*abs(pom1)/4;
pw = h*( pom3 - pom4 );

p = [px; py; pv; pw];


function dPHI = jakobian(U,N,A1,A2)  % vraci matici
% Jacobiova matice
h = 1/N;

V_n = U(2*N-1:3*N-2); % z U vybereme V_n az V_n-1
V_n1= U(2*N:3*N-1);   % z U vybereme V_n+1 az V_n
W_n = U(3*N:4*N-1);   % z U vybereme W_n az W_n-1
W_n1= U(3*N+1:4*N);   % z U vybereme W_n+1 az W_n
```

```
% Maticove bloky, viz schema v pdf
% Maticovy blok : prvni radek, treti sloupec
d11 = (h/2)*sin(V_n);
d12 = (h/2)*sin(V_n1);
D1 = sparse(1:N,1:N,d11,N,N+1) + sparse(1:N,2:N+1,d12,N,N+1);
% full(D1)

% Maticovy blok : druhy radek, treti sloupec
d21 = -(h/2)*cos(V_n);
d22 = -(h/2)*cos(V_n1);
D2 = sparse(1:N,1:N,d21,N,N+1) + sparse(1:N,2:N+1,d22,N,N+1);
% full(D2)

% Maticovy blok : treti radek, treti sloupec
pom1 = sign( sin(V_n) + sin(V_n1) ).*( sin(V_n) + sin(V_n1) );
d31 = -1 + h*( sin(V_n) + pom1.*cos(V_n) )./(W_n + W_n1);
d32 =  1 + h*( sin(V_n1) + pom1.*cos(V_n1) )./(W_n + W_n1);
D3 = sparse(1:N,1:N,d31,N,N+1) + sparse(1:N,2:N+1,d32,N,N+1);
% full(D3)

% Maticovy blok : treti radek, ctvrty sloupec
pom2 = cos(V_n)+cos(V_n1) - abs( sin(V_n) + sin(V_n1) ).*( sin(V_n) + sin(V_n1) )/2;
d41 = h*pom2./( (W_n + W_n1).^2 );
d42 =  h*pom2./( (W_n + W_n1).^2 );
D4 = sparse(1:N,1:N,d41,N,N+1) + sparse(1:N,2:N+1,d42,N,N+1);
% full(D4)

% Maticovy blok : ctvrty radek, treti sloupec
pom3 = sign( cos(V_n) + cos(V_n1) ).*( cos(V_n) + cos(V_n1) );
d51 = -(h/2)*( cos(V_n) + pom3.*sin(V_n) );
d52 = -(h/2)*( cos(V_n1) + pom3.*sin(V_n1) );
D5 = sparse(1:N,1:N,d51,N,N+1) + sparse(1:N,2:N+1,d52,N,N+1);
% full(D5)

dPHI = [A1 zeros(N,N-1) D1 zeros(N,N+1);
       zeros(N,N-1) A1 D2 zeros(N,N+1);
       zeros(N,2*N-2) D3 D4;
       zeros(N,2*N-2) D5 A2];
% full(dPHI)
```