```
% cell_nut_quim.m | Solves convection-diffusion-reaction equation using MOLE

clc
close all
format short

addpath('../mole_MATLAB')

% Mimetic operator's parameters
k = 2;
m = 101;
n = 101;
o = 101;
% Domain's dimensions
a = 0;
b = 101;
c = 0;
d = 101;
e = 0;
f = 101;

% Spatial step sizes
dx = (b-a)/m;
dy = (d-c)/n;
dz = (f-e)/o;

% Mimetic operators
D = div3D(k, m, dx, n, dy, o, dz);
G = grad3D(k, m, dx, n, dy, o, dz);
I = interpol3D(m, n, o, 1, 1, 1);
L = lap3D(k, m, dx, n, dy, o, dz);

% CÃ©lulas vector
S = zeros(m+2, n+2, o+2);
%Nutrientes vector
N = ones(m+2, n+2, o+2);
%Quimiocina vector
Q = 0.05*ones(m+2, n+2, o+2);

%ParÃ¡metros
smax = 10^(5); %Valor real 10^(12)
qmax = 10^(1); %Valor real 10^(6)
nmax = 10^(-2); %Valor real 8*10^(-2)

%Difusividad de stem cells
difs = 2*10^(-1); %Valor real 2*10^(-12)
%coeficiente de muerte celular
Rd = 3*10^(-7);%Valor real 3*10^(-7)
% parÃ¡metro de Frontera libre
lamb = 10^(-2);%Valor original 10^(-9)
%ParÃ¡metros de Q multipicado por CHI+
chi = 0.008; %Valor original 8*10^(-9)
```

```
%DifusiÃ³n de quimiocina
Dq = 10^(-1); %Valor original5*10^(-11)
Rsq = 10^(-6); %Valor original 10^(-12)
Rq = 10^(-8); %Valor original 7*10^(-5)
phi = 10^(-6); %Valor original 1

%Parametros de Nutrientes
delta = 10^(-2); %Valor original 10^(-2)
%DifusiÃ³n de nutriente
Dn = 2*10^(-1); %Valor original 2*10^(-11)
%K constante media de saturaciÃ³n de nutriente
k1 = 0.05; %Valor real 5*10^(-2)
%proliferaciÃ³n
umax = 10^(-5); %tasa mÃ¡xima de proliferaciÃ³n
theta = 0.04; %constante de funciÃ³n de Hill

% Impose initial conditions ---------------------------------------------
%S=0, N=1, Q=0.5
%DefiniciÃ³n de nÃºcleo necrÃ³tico
i0=ceil(m/3)+1;
i1=floor(2*m/3);
j0=ceil(n/3)+1;
j1=floor(2*n/3);
p0=ceil(o/3)+1;
p1=floor(2*o/3);

ptab= floor(2*o/3)*ones(m+2,n+2); %arreglo del indice p1
%Boundary conditions
S(i0:i1, j1:n+2, o+2)= bc(0); %cÃ©lulas implantadas en A
S(i0:i1, j0:j1, p0:p1)= 0;
Q(i0:i1, j0:ceil((n+2)/2), p0:p1)= 0.7; %Dirichlet para Q en gamma
Q(i0:i1, ceil((n+2)/2)+1:i1, p0:p1)= 1; %Dirichlet para Q en gamma
%Condiciones de contorno externas de Neumann
N(1:(m+2),1,1:(o+2))= N(1:(m+2),2,1:(o+2));
N(1:(m+2),n+2,1:(o+2)) = N(1:(m+2),n+1,1:(o+2));
N(1,1:(n+2),1:(o+2)) = N(2,1:(n+2),1:(o+2));
N(m+2,1:(n+2),1:(o+2)) = N(m+1,1:(n+2),1:(o+2));
N(1:(m+2),1:(n+2),1) = N(1:(m+2),1:(n+2),2);
N(1:(m+2),1:(n+2),o+2) = N(1:(m+2),1:(n+2),o+1);

%Condiciones de contorno internas de Neumann
N(i0:i1,j0,p0:p1) = N(i0:i1,j0-1,p0:p1);
N(i0:i1,j1,p0:p1) = N(i0:i1,j1+1,p0:p1);
N(i0,j0:j1,p0:p1) = N(i0-1,j0:j1,p0:p1);
N(i1,j0:j1,p0:p1) = N(i1+1,j0:j1,p0:p1);
N(i0:i1,j0:j1,p0) = N(i0:j1,j0:j1,p0-1);
N(i0:i1,j0:j1,p1) = N(i0:j1,j0:j1,p1+1);

S=S(:);
N=N(:);
Q=Q(:);

%velocity field
```

```
V = chi*qmax*G*Q;
% dt based oAA=zeros(o+2,1);n von Neumann criterion
dt1 = dx^2/(3*difs)/3;
% dt based on CFL condition
dt2 = (dx/max(V))/3;
% Select minimum dt
dt = min(dt1, dt2);

iters = 3000;  % 90 = 30s  if dt = 0.3333 because CFL

%Premultiplicacion de operador para Sn
Ls = dt*D*((difs*G));
Ls = Ls - dt*Rd*speye(size(L))+ speye(size(L));
DD =  dt*D*spdiags(V,0, numel(V), numel(V))*I;

for i = 1 : iters*3

    % Solve diffusive term using FTCS scheme
    S = Ls*S;
    % Impose conditions
    S=reshape(S, m+2, n+2, o+2);
    %S(i0:i1, j0:j1, p0:p1)=0;
    S(i0:i1, j1:n+2, o+2)= bc(i*dt);
    S=S(:);

    % Solve advective term using upwind scheme
    S = S - DD*S;
    % Impose conditions
    S=reshape(S, m+2, n+2, o+2);
    %S(i0:i1, j0:j1, p0:p1)=0;
    S(i0:i1, j1:n+2, o+2)= bc(i*dt);
    S=S(:);

    %PremultiplicaciÃ³n de operador para Sn+1
    NN = N./(k1+N);
    Ms = -dt*umax*(1+k1)*spdiags(NN.*(1-S),0,numel(S), numel(S));
    Ms = speye(size(S,1))+ Ms;
    %Solve system linear
    S = Ms\S;

    %Impose conditions
    S=reshape(S, m+2, n+2, o+2);
    %S(i0:i1, j0:j1, p0:p1)=0;
    S(i0:i1,j1:n+2, o+1)= bc(i*dt);
    S=S(:);

    %Quimiocinas
    %Q=reshape(Q, m+2, n+2, o+2);
    %Q(i0:i1, j0:j1, p0:p1)=1;
    %Q=Q(:);

    %Premultiplication laplaciano de quimiocina
    Lq = dt*Dq*L;
    Lq = Lq +speye(size(Lq));
    Q=Lq*Q;

    Mq = dt*Rq*speye(size(S,1))+Rsq*spdiags(S,0,numel(S),numel(S));
    %consumo de quimiocina
    Q = Q+Mq*Q;

    %Premultiplication laplaciano de nutriente
    Ln = dt*Dn*L;
    Ln = Ln + speye(size(Ln));
    %Nutrientes
    N = Ln*N;
    %PremultiplicaciÃ³n de operador para Nn+1
    Mn = N.*N.*N.*S./((theta^4)+(N.*N.*N.*N));
    Mn = dt*spdiags(Mn,0,numel(Mn),numel(Mn));
    Mn = Mn + speye(size(Mn));

    N = Mn\N;

    % Impose conditions
    S=reshape(S, m+2, n+2, o+2);
    Q=reshape(Q, m+2, n+2, o+2);
    %Imponer condiciones para N
    N=reshape(N, m+2, n+2, o+2);
    for ii= i0:i1
       for j=j0:j1
          gamma=dz*ptab(ii,j);
          gamma=gamma-dt*smax*lamb*difs*S(ii,j,ptab(ii,j)+1)/dz; %nuevo valor de la frontera libre
          if gamma<=dz*ptab(ii,j)-(dz/2)
             ptab(ii,j)=ptab(ii,j)-1;
          end

          %Impose conditions
          S(ii, j, p0:ptab(ii,j))=0;
          if j<=ceil((n+2)/2)
             Q(ii, j, p0:ptab(ii,j))=0.7;
          else
             Q(ii, j, p0:ptab(ii,j))=1;
          end

          N(ii, j, p0:ptab(ii,j))=0; %Cero nutrientes en nucleo necrotico
          %Condicion de contorno interna de Neumann
          N(ii,j,ptab(ii,j)) = N(ii,j,ptab(ii,j)+1);

          N(i0,j,p0:ptab(ii,j)) = N(i0-1,j,p0:ptab(ii,j));
          N(i1,j,p0:ptab(ii,j)) = N(i1+1,j,p0:ptab(ii,j));
          N(ii,j0,p0:ptab(ii,j)) = N(ii,j0-1,p0:ptab(ii,j));
          N(ii,j1,p0:ptab(ii,j)) = N(ii,j1+1,p0:ptab(ii,j));

          %Condicion Neumann
          Q(i0-1,j,p0:ptab(ii,j))= Q(i0,j,p0:ptab(ii,j))+dx*phi;
          Q(i1+1,j,p0:ptab(ii,j))= Q(i1,j,p0:ptab(ii,j))+dx*phi;
          Q(ii,j0-1,p0:ptab(ii,j))=Q(ii,j0,p0:ptab(ii,j))+dy*phi;
```

```
        Q(ii,j1+1,p0:ptab(ii,j))=Q(ii,j1,p0:ptab(ii,j))+dy*phi;

        Q(ii,j,ptab(ii,j)+1)=Q(ii,j,ptab(ii,j))+dz*phi;

    end

end

N(i0:i1,j0:j1,p0) = N(i0:j1,j0:j1,p0-1);
Q(i0:i1,j0:j1,p0-1)=Q(i0:i1,j0:j1,p0)+dz*phi;
Q=Q(:);

%Actualiza V
V = chi*qmax*G*Q;
DD = dt*D*spdiags(V, 0, numel(V), numel(V))*I;

%Condiciones de contorno externas de Neumann
N(1:(m+2),2,1:(o+2))= N(1:(m+2),1,1:(o+2));
N(1:(m+2),n+1,1:(o+2)) = N(1:(m+2),n+2,1:(o+2));
N(2,1:(n+2),1:(o+2)) = N(1,1:(n+2),1:(o+2));
N(m+1,1:(n+2),1:(o+2)) = N(m+2,1:(n+2),1:(o+2));
N(1:(m+2),1:(n+2),2) = N(1:(m+2),1:(n+2),1);
N(1:(m+2),1:(n+2),o+1) = N(1:(m+2),1:(n+2),o+2);



pause(0.01)
S(i0:i1,j1:n+2, o+2)= bc(i*dt);

% Plot cell profile
slice(S, ceil(n+2)/2, ceil((m+2)/2), o+2);
shading interp
set(gca, 'XDir', 'reverse')
set(gca, 'ZDir', 'reverse')
set(gcf, 'color', 'w')
xlabel('y')
ylabel('x')
zlabel('z')
axis equal
title(['Cell concentration profile, t = ' num2str(i*dt, '%2.2f')])
colorbar
%view(90, 90)
%Plot quimiocinas
%Q1=reshape(Q, m+2, n+2, o+2);
%slice(Q1, ceil(n+2)/2, ceil((m+2)/2), o+2);
%shading interp
%set(gca, 'XDir', 'reverse')
%set(gca, 'ZDir', 'reverse')
%set(gcf, 'color', 'w')
%xlabel('y')
%ylabel('x')
%zlabel('z')
%axis equal
```

```
    %title(['Chemoquine concentration profile, t = ' num2str(i*dt, '%2.2f')])
    %colorbar
    %view(90, 90)

    %AA=zeros(o+2,1);
    %for ii=1:o+2
     %   AA(ii)=S(ceil((m+2)/2),ceil((n+2)/2),ii);
    %end
    S=S(:);
    %if i==iters*3
     %   S=reshape(S,m+2,n+2,o+2);
    %end

   % plot(AA);
   N=N(:);

end

min(S)
max(S)
```