```
% patternformation_GiererMeinhardt_system.m


t_max = 30000;

m=150;
n=150;

A=zeros(m,n);
H=A;

dA = zeros(m,n);
dH = dA;

for s=1:m
    for t=1:n
        A(s,t)=0.3+0.4*(rand);
        H(s,t)=0.3+0.4*rand;
    end
end

A_diff_coeff = 0.6;
H_diff_coeff= 230;
Prod_rate_coeff = 0.9;
A_decay_coeff = 0.25;
H_decay_coeff = 1;
A_base_coeff = 0.2;
H_base_coeff = 0.2;
hill_coeff = 2;

for t=1:t_max

    t

    for i=1:m
        for j=1:n
            a=A(i,j);
            h=H(i,j);
            neigh_ind = get_topology_neighbour_indices(i,j,m,n);

            % diffusion term

            dA(i,j) = compute_flow_from_neighbours(A,i,j,neigh_ind,A_diff_coeff);
            dH(i,j) = compute_flow_from_neighbours(H,i,j,neigh_ind,H_diff_coeff);


            % production term

            dA(i,j) = dA(i,j)+ Prod_rate_coeff*a*a/h;
            dH(i,j) = dH(i,j) + Prod_rate_coeff*a*a;


            % decay term

            dA(i,j) = dA(i,j)- A_decay_coeff*a;
            dH(i,j) = dH(i,j)- H_decay_coeff*h;


            % baseline production term

            dA(i,j) = dA(i,j) + A_base_coeff;
            dH(i,j) = dH(i,j) + H_base_coeff;


            % conditions

            if (A(i,j)<0)
                A(i,j)=0;
            end
            if (H(i,j)<0)
                H(i,j)=0;
            end

        end
    end

    sprintf('step: %d', t)

    if (mod(t,50)==0)
        figure(1);
        subplot(1,2,1);
        imagesc(A);
        colormap(gray);
        axis image;
        axis off;
        subplot(1,2,2)
        imagesc(H);
        colormap(gray);
        axis image;
        axis off;
        title('H');
        pause(0.001);
```

```
    end


    A = (A+0.001*dA);
    H = (H+0.001*dH);

    if (mod(t,500)==0)
        save curr_Matrices A H;
    end

end
```

```
% compute_flow_from_neighbours.m


function compute_flow_from_neighbours = compute_flow_from_neighbours(M,posi,posj,neigh_ind,Diff_coeff)

% PRE: The matrix M specifies the concentration of a substance, posi and
% posj point to the cell we are looking at, neigh_ind are the neighbouring indices of
% (posi, posj). Diff_coeff is the diffusion coefficient.
% POST: Returns the flow of a substance at position (posi, posj) due to diffusion

    temp_concentration = 0;
    for i=1:size(neigh_ind,2)
        if (neigh_ind(1,i)~=0)
            this_con = M(neigh_ind(1,i),neigh_ind(2,i))-M(posi,posj);
            temp_concentration = temp_concentration + this_con*Diff_coeff;
        end
    end
    compute_flow_from_neighbours = temp_concentration;
end
```

```
% get_topology_neighbour_indices.m

function get_topology_neighbour_indices = get_topology_neighbour_indices_2(posi,posj,maxi,maxj)

% PRE: Inputs are the cell indices of the cell for which the neighbour
% cells are computed, and the number of cells per side
% POST: returns the indicies of the neighbours of (posi, posj) under
% periodic boundary conditions

    temp_result = [posi posi-1 posi posi+1; posj-1 posj posj+1 posj];

    for i=1:size(temp_result,2)
        if (temp_result(1,i)==0)
            temp_result(1,i)=maxi;
        end
        if (temp_result(1,i)==maxi+1)
            temp_result(1,i)=1;
        end

        if (temp_result(2,i)==0)
            temp_result(2,i)=maxj;
        end
        if (temp_result(2,i)==maxj+1)
            temp_result(2,i)=1;
        end
    end

    get_topology_neighbour_indices = temp_result;
end
```