Mathematical Modelling in the Natural Sciences SS20
Solutions to Exercises on Sheet 3

Exercises und Lecture Notes

Contents

• Exercise 1: SIR Model, Equilibrium with Diffusion

◦ Task

Show for $N = 2$ and $M = 1$ that the SIR model with diffusion on page 144 of the lecture notes has a locally asymptotically stable equilibrium $(S_{i,j}, I_{i,j}, R_{i,j}) = (S_1^*, I_1^*, R_1^*)$ if $I_2^\star < 0$ and a locally asymptotically stable equilibrium $(S_{i,j}, I_{i,j}, R_{i,j}) = (S_2^*, I_2^*, R_2^*)$ if $I_2^\star > 0$. Bonus: Show this result for arbitrary N and M.

◦ Solution

The following argument applies independently of N and M. The system of ODEs including a discrete formulation of diffusion is

$$\begin{aligned} \boldsymbol{S}' &= \sigma L\boldsymbol{S} + \beta - (\mu + \lambda\boldsymbol{I})\boldsymbol{S} \\ \boldsymbol{I}' &= \iota L\boldsymbol{I} + (\lambda\boldsymbol{S} - \mu - \gamma)\boldsymbol{I} \\ \boldsymbol{R}' &= \rho L\boldsymbol{R} + \gamma\boldsymbol{I} - \mu\boldsymbol{R} \end{aligned}$$

where, e.g., componentwise multiplication is meant with

$$\boldsymbol{IS} = \{I_{i,j}S_{i,j} : i = 1, \ldots, N, j = 1, \ldots, M\}.$$

Recall the parameters from the script

$$S_1^\star = \frac{\beta}{\mu}, \quad I_1^\star = 0, \quad R_1^\star = 0$$
$$S_2^\star = \frac{\mu+\gamma}{\lambda}, \quad I_2^\star = \frac{\beta}{\mu+\gamma} - \frac{\mu}{\lambda}, \quad R_2^\star = \frac{\gamma}{\mu} I_2^\star.$$

Also with torus boundary conditions the discrete diffusion operator has a spectrum satisfying

$$\sigma(L) \subset (-\infty, 0], \quad 0 \in \sigma(L).$$

A direct calculation shows that

$$\boldsymbol{S}_1^\star = S_1^\star \mathbf{1}, \quad \boldsymbol{I}_1^\star = I_1^\star \mathbf{1}, \quad \boldsymbol{R}_1^\star = R_1^\star \mathbf{1}, \quad I_2^\star < 0$$
$$\boldsymbol{S}_2^\star = S_2^\star \mathbf{1}, \quad \boldsymbol{I}_2^\star = I_2^\star \mathbf{1}, \quad \boldsymbol{R}_2^\star = R_2^\star \mathbf{1}, \quad I_2^\star > 0$$

are equilbria, also for the system with diffusion. Since there is no back coupling from $\boldsymbol{R}$ in the first two equations, consider the reduced system

$$\begin{aligned} \boldsymbol{S}' &= \sigma L \boldsymbol{S} + \beta - (\mu + \lambda \boldsymbol{I}) \boldsymbol{S} \\ \boldsymbol{I}' &= \iota L \boldsymbol{I} + (\lambda \boldsymbol{S} - \mu - \gamma) \boldsymbol{I}. \end{aligned}$$

The Jacobian of the right side is

$$J(\boldsymbol{S}, \boldsymbol{I}) = \begin{bmatrix} \sigma L - \mu \mathcal{D}(\mathbf{1}) - \lambda \mathcal{D}(\boldsymbol{I}) & -\lambda \mathcal{D}(\boldsymbol{S}) \\ \lambda \mathcal{D}(\boldsymbol{I}) & \iota L + \lambda \mathcal{D}(\boldsymbol{S}) - \mu \mathcal{D}(\mathbf{1}) - \gamma \mathcal{D}(\mathbf{1}) \end{bmatrix}$$

where, e.g., $\mathcal{D}(\boldsymbol{I}) = \mathrm{diag}(\boldsymbol{I})$. An eigenvector $\boldsymbol{V} = (\boldsymbol{v}_1, \boldsymbol{v}_2)$ of $J(S\mathbf{1}, I\mathbf{1})$ with eigenvalue ϵ must satisfy

$$J(S\mathbf{1}, I\mathbf{1})\boldsymbol{V} = \begin{bmatrix} (\sigma L - \mu - \lambda I)\boldsymbol{v}_1 - \lambda S \boldsymbol{v}_2 \\ \lambda I \boldsymbol{v}_1 + (\iota L + \lambda S - \mu - \gamma)\boldsymbol{v}_2 \end{bmatrix} = \epsilon \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \end{bmatrix}$$

or

$$\begin{aligned} (\sigma L - \mu - \lambda I - \epsilon)\boldsymbol{v}_1 &= \lambda S \boldsymbol{v}_2 \\ (\iota L + \lambda S - \mu - \gamma - \epsilon)\boldsymbol{v}_2 &= -\lambda I \boldsymbol{v}_1 \end{aligned}$$

and

$$I \boldsymbol{v}_1^\top (\sigma L - \mu - \lambda I)\boldsymbol{v}_1 + S \boldsymbol{v}_2^\top (\iota L + \lambda S - \mu - \gamma)\boldsymbol{v}_2 = \epsilon(\|\boldsymbol{v}_1\|^2 + \|\boldsymbol{v}_2\|^2).$$

With $(S, I) = (S_1^\star, I_1^\star)$, $I_2^\star < 0$,

$$\epsilon \left(\|\boldsymbol{v}_1\|^2 + \|\boldsymbol{v}_2\|^2\right) = \frac{\beta}{\mu} \boldsymbol{v}_2^\top \left(\iota L + \frac{\lambda \beta}{\mu} - \mu - \gamma\right) \boldsymbol{v}_2 = \frac{\iota \beta}{\mu} \boldsymbol{v}_2^\top L \boldsymbol{v}_2 + \frac{\lambda}{\mu}(\mu + \gamma) I_2^\star \|\boldsymbol{v}_2\|^2 < 0$$

and each eigenvalue ϵ of $J(S_1^\star, I_1^\star)$ satisfies $\epsilon < 0$. With $(S, I) = (S_2^\star, I_2^\star)$, $S_2^\star, I_2^\star > 0$,

$$\epsilon \left(\|\boldsymbol{v}_1\|^2 + \|\boldsymbol{v}_2\|^2\right) = I_2^\star \boldsymbol{v}_1^\top (\sigma L - \mu - \lambda I_2^\star)\boldsymbol{v}_1 + S_2^\star \boldsymbol{v}_2^\top (\iota L)\boldsymbol{v}_2 < 0.$$

and each eigenvalue ϵ of $J(S_2^\star, I_2^\star)$ satisfies $\epsilon < 0$.

• Exercise 2: SIR Model with Quarantine Strategy

◦ Task

As on pages 143 – 149 of the lecture notes, implement an SIR model with two-dimensional diffusion and include any relevant effects, even such as loss of immunity, vaccination, etc. Use your model to develop an effective quarantine strategy.

◦ Solution

Consider the following model on the space-time domain $\Omega \times [0,\infty)$ with $\Omega = (0,1)^2$.

$$
\begin{cases}
\begin{cases} S_t &= \nabla\cdot(\sigma\nabla S) + a_1 S(a_2 - S) - a_3 SI + a_6 R - a_8 S \\ I_t &= \nabla\cdot(\iota\nabla I) + a_3 SI - a_4 I - a_5 I \\ R_t &= \nabla\cdot(\rho\nabla R) + a_5 I - a_6 R - a_7 R + a_8 S \end{cases} \qquad x\in\Omega, \quad t>0 \\
\begin{cases} S(0,y,t) = S(1,y,t) \\ I(0,y,t) = I(1,y,t) \\ R(0,y,t) = R(1,y,t) \end{cases} \quad y\in[0,1], \quad t\geq 0 \qquad \begin{cases} S(x,0,t) = S(x,1,t) \\ I(x,0,t) = I(x,1,t) \\ R(x,0,t) = R(x,1,t) \end{cases} \quad x\in[0,1], \quad t\geq 0 \\
S(x,y,0) = S_0(x,y), \quad I(x,y,0) = I_0(x,y), \quad R(x,y,0) = R_0(x,y), \quad (x,y)\in\Omega
\end{cases}
$$

With $\sigma = \iota = \rho = 0$ there is no diffusion, and the resulting point model can be compared with SIR models in the script. For instance, the case

$$a_2 \to \infty \quad \text{with} \quad a_1 a_2 = \text{constant} \quad \text{and} \quad a_6 = a_8 = 0$$

corresponds to the classical predator prey model on p. 87 in the script, where R is decoupled from the (S,I) system. The case

$$a_i > 0, \quad i \neq 6, \quad a_6 = a_8 = 0$$

corresponds to the logistic perturbation of the classical predator prey model on p. 89 in the script, where R is decoupled from the (S,I) system. Although this model is seen in the script to have only an isolated equilibrium, it will be seen below that with added coupling with R

$$a_i > 0, \quad i = 1,\ldots,8$$

a limit cycle emerges. The parameters can otherwise be interpreted as follows:

- a_1: time-scale for an otherwise logistic evolution of S
- a_2: capacity for S
- a_3: infectiousness of the disease
- a_4: mortality for the infected
- a_5: healability for the infected
- a_6: immunity loss rate for the recovered
- a_7: mortality of the recovered
- a_8: immunity gain rate from vaccination of the susceptibles

The following Matlab code is used to simulate the model for the cases illustrated graphically below.

```
% model parameters
% S' = a1 S(a2 - S) - a3 S I + a6 R - a8 S
% I' = a3 S I - a4 I - a5 I
% R' = a5 I - a6 R - a7 R + a8 S
    a1 = 0.1;
    a2 = 10;
    a3 = 10;
    a4 = 1;
    a5 = 0.1;
    a6 = 0.01;
```

```
    a7 = 0.01;
    a8 = 0.1;

% geometric parameters
    Nx   = 25;
    Ny   = 25;
    NxNy = Nx*Ny;
    xmin = 0; xmax = 1;
    ymin = 0; ymax = 1;
    x  = linspace(xmin,xmax,Nx+1);
    hx = x(2)-x(1); x = x(1:Nx) + hx/2; xx = kron(x',ones(1,Ny));
    y  = linspace(ymin,ymax,Ny+1);
    hy = y(2)-y(1); y = y(1:Ny) + hy/2; yy = kron(ones(Nx,1),y);

% diffusion with periodic BCs
    dx = spdiags(ones(Nx,1),0,Nx,Nx) ...
       - spdiags(ones(Nx,1),-1,Nx,Nx);
    dx(1,Nx) = -1;
    dx = dx/hx;
    Dx = kron(speye(Ny),dx);
    dy = spdiags(ones(Ny,1),0,Ny,Ny) ...
       - spdiags(ones(Ny,1),-1,Ny,Ny);
    dy(1,Ny) = -1;
    dy = dy/hy;
    Dy = kron(dy,speye(Nx));

% default uniform motilities: sigma (sx, sy), iota (ix,iy) and rho (rx, ry)
    sx = ones(Nx,Ny);
    sy = ones(Nx,Ny);
    ix = ones(Nx,Ny);
    iy = ones(Nx,Ny);
    rx = ones(Nx,Ny);
    ry = ones(Nx,Ny);

    example = 1;
    switch example
        case 1
% uniform state possibly with noise
            ns = 0;
            S  = 0.5*ones(Nx,Ny) + ns*randn(1);
            I  = 0.1*ones(Nx,Ny) + ns*randn(1);
            R  = zeros(Nx,Ny) + ns*randn(1);
        case 2
% add some infections to susceptibles
            S  = 1.0*rand(Nx,Ny);
            I  = zeros(Nx,Ny); I(randi(NxNy,10)) = 1;
            R  = zeros(Nx,Ny);
% criss-cross quarantine boundaries, tighter for n->1
```

```
            n = 3; Nxv = n:n:Nx; Nyv = n:n:Ny;
            sx(Nxv,:) = 0; sy(:,Nyv) = 0;
            ix(Nxv,:) = 0; iy(:,Nyv) = 0;
            rx(Nxv,:) = 0; ry(:,Nyv) = 0;
    end

% diffusion operators
    Ls = Dx'*spdiags(sx(:),0,NxNy,NxNy)*Dx + Dy'*spdiags(sy(:),0,NxNy,NxNy)*Dy;
    Li = Dx'*spdiags(ix(:),0,NxNy,NxNy)*Dx + Dy'*spdiags(iy(:),0,NxNy,NxNy)*Dy;
    Lr = Dx'*spdiags(rx(:),0,NxNy,NxNy)*Dx + Dy'*spdiags(ry(:),0,NxNy,NxNy)*Dy;

% ode system
    sir = @(t,X) [X(1:NxNy).*(a1*(a2 - X(1:NxNy))  ...
                        - a3*X((NxNy+1):(2*NxNy)))  ...
                       + a6*X((2*NxNy+1):(3*NxNy))  ...
                                    - Ls*X(1:NxNy)  ...
                                    - a8*X(1:NxNy); ...
   X((NxNy+1):(2*NxNy)).*(a3*X(1:NxNy) - a4 - a5)  ...
                         - Li*X((NxNy+1):(2*NxNy)); ...
                           a5*X((NxNy+1):(2*NxNy))  ...
              - (a6 + a7)*X((2*NxNy+1):(3*NxNy))  ...
                       - Lr*X((2*NxNy+1):(3*NxNy))  ...
                                    + a8*X(1:NxNy)];

% setup figure
    h1 = figure(1);
    close(h1);
    h1 = figure(1);
    set(h1,'Position',[10 10 500 500]);

% initial conditions
    surf(xx,yy,S,ones(size(S)))
    colormap([0 1 0;1 0 0;0 0 1])
    hold on;
    surf(xx,yy,I,ones(size(I))*2)
    surf(xx,yy,R,ones(size(R))*3)
    legend('S','I','R')
    xlabel('x'); ylabel('y'); zlabel('SIR');
    title('initial distributions')
    hold off;
    drawnow;

% solve
    X0 = [S(:);I(:);R(:)];
    tspan = [0,200];
    opts = odeset('RelTol',1.0e-10);
    [t,X] = ode15s(sir,tspan,X0,opts);
```

```
% store solution
    Nt = length(t);
    Sv = reshape(X(:,1:NxNy),Nt,Nx,Ny);
    S  = reshape(Sv(Nt,:,:),Nx,Ny);
    Iv = reshape(X(:,(NxNy+1):(2*NxNy)),Nt,Nx,Ny);
    I  = reshape(Iv(Nt,:,:),Nx,Ny);
    Rv = reshape(X(:,(2*NxNy+1):(3*NxNy)),Nt,Nx,Ny);
    R  = reshape(Rv(Nt,:,:),Nx,Ny);

    Sbar = mean(X(:,1:NxNy),2);
    Ibar = mean(X(:,(NxNy+1):(2*NxNy)),2);
    Rbar = mean(X(:,(2*NxNy+1):(3*NxNy)),2);

% dynamic densities
    plot(t,Sbar,'g',t,Ibar,'r',t,Rbar,'b');
    axis tight
    legend('S','I','R');
    xlabel('t'); ylabel('SIR');
    title('dynamic densities');

% setup figure
    h2 = figure(2);
    close(h2);
    h2 = figure(2);
    set(h2,'Position',[550 10 500 500]);

% final conditions
    surf(xx,yy,S,ones(size(S)))
    colormap([0 1 0;1 0 0;0 0 1])
    hold on;
    surf(xx,yy,I,ones(size(I))*2)
    surf(xx,yy,R,ones(size(R))*3)
    legend('S','I','R')
    xlabel('x'); ylabel('y'); zlabel('SIR');
    title('final distributions')
    hold off;

    for k=1:Nt
        S = reshape(Sv(k,:,:),Nx,Ny);
        I = reshape(Iv(k,:,:),Nx,Ny);
        R = reshape(Rv(k,:,:),Nx,Ny);
        surf(xx,yy,S,ones(size(S)))
        colormap([0 1 0;1 0 0;0 0 1])
        hold on
        surf(xx,yy,I,ones(size(I))*2)
        surf(xx,yy,R,ones(size(R))*3)
        legend('S','I','R')
        xlabel('x'); ylabel('y'); zlabel('SIR');
```

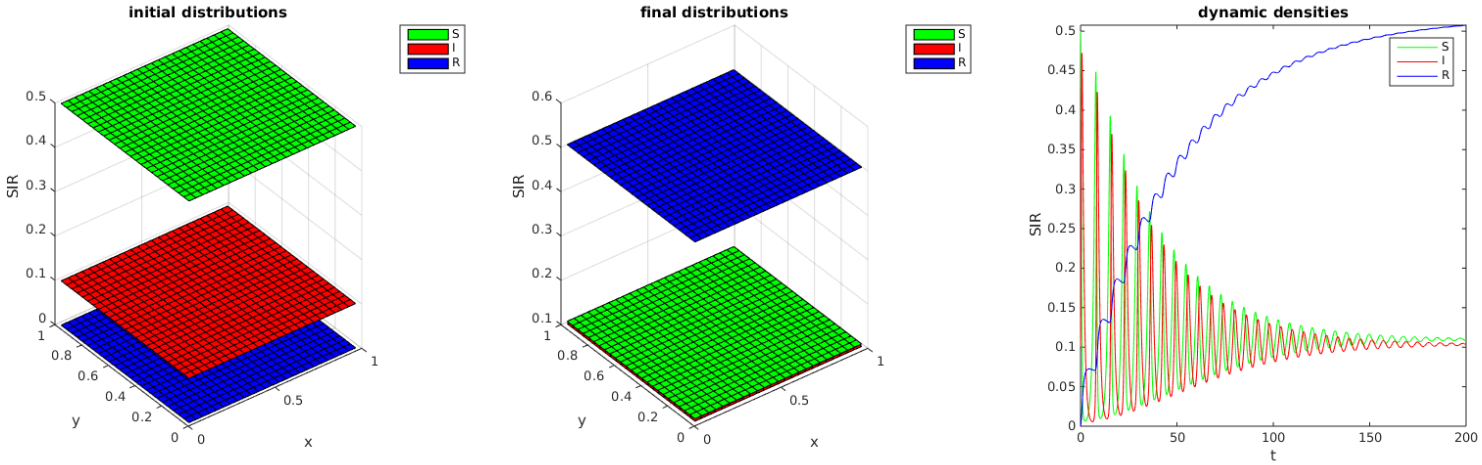
```
        title(['t=',num2str(t(k))])
        hold off;
        drawnow;
    end
```

Unless otherwise stated, for all of the following cases, the parameters are chosen as seen at the beginning of the code,

$$a_1 = 0.1, \quad a_2 = 10, \quad a_3 = 10, \quad a_4 = 1, \quad a_5 = 0.1, \quad a_6 = 0.01, \quad a_7 = 0.01, \quad a_8 = 0.$$

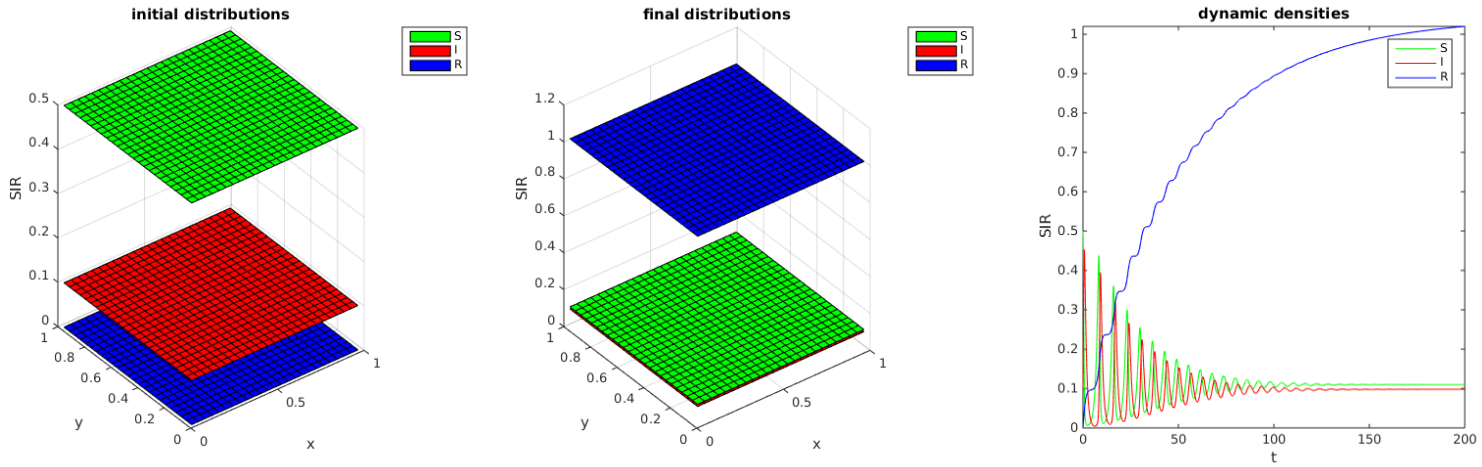
The following two cases corresond to `example = 1` in the code, i.e.,

$$\sigma = \iota = \rho = 1, \quad S_0 = 0.5, \quad I_0 = 0.1, \quad R_0 = 0.$$



Since the diffusivities are constant and the states are spatially invariant, depending only upon time, this simulation is equivalent to that of a point model without diffusion. Note the emergence of a limit cycle as mentioned above!

Now immunization is used by setting $a_8 = 0.1$ in the previous example, and the following result shows that the infection is stabilized to an equilibrium.

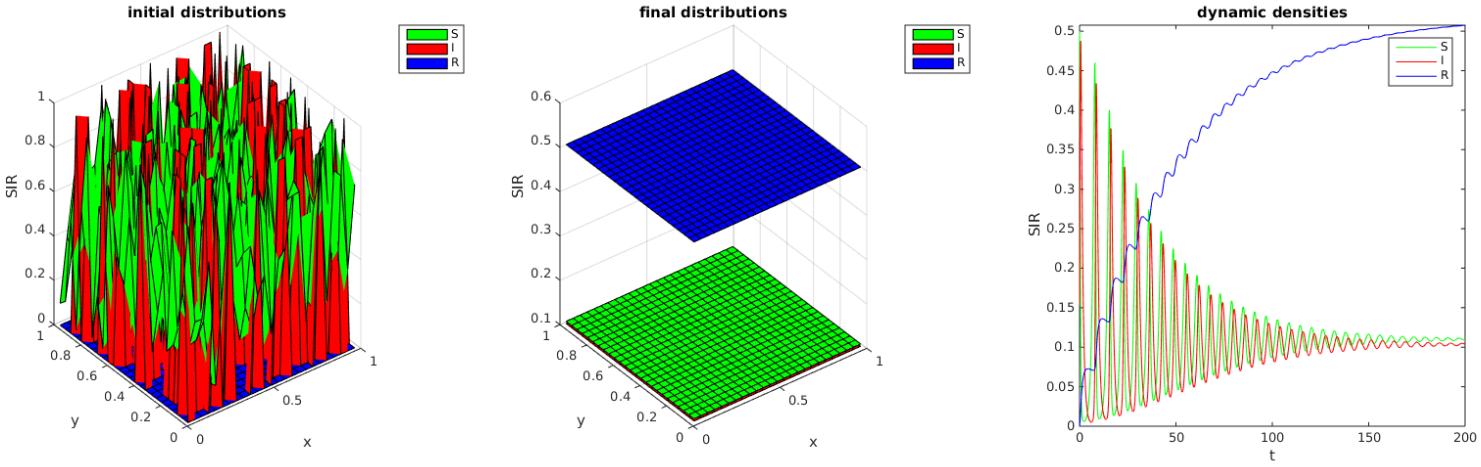


The remaining cases correspond to `example = 2` in the code, in which the initial conditions are determined according to

```
S  = 1.0*rand(Nx,Ny);
I  = zeros(Nx,Ny); I(randi(NxNy,10)) = 1;
R  = zeros(Nx,Ny);
```

The next simulation corresponds to constant diffusivitites $\sigma = \iota = \rho = 1$ where the following lines are commented out:

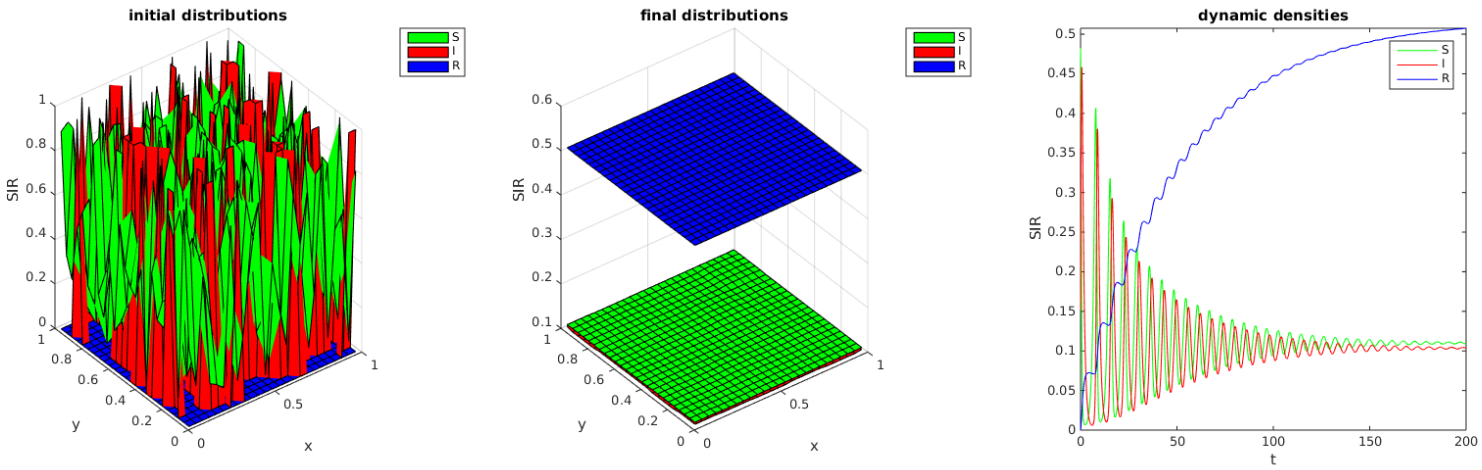
```
%   n = 3; Nxv = n:n:Nx; Nyv = n:n:Ny;
%   sx(Nxv,:) = 0; sy(:,Nyv) = 0;
%   ix(Nxv,:) = 0; iy(:,Nyv) = 0;
%   rx(Nxv,:) = 0; ry(:,Nyv) = 0;
```



Note that the final distributions are uniform and that the evolution of mean values is similar to the point model shown in the first case above.

The next case corresponds to quarantining on a coarse grid by turning off diffusivities at every fifth interface: `n = 5`.

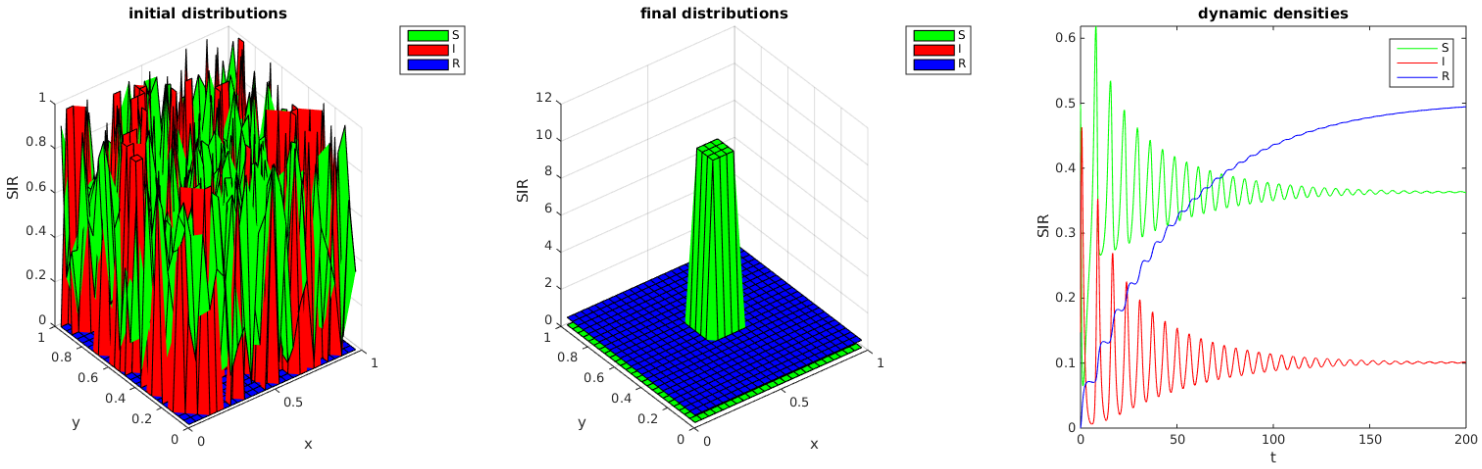
```
n = 5; Nxv = n:n:Nx; Nyv = n:n:Ny;
sx(Nxv,:) = 0; sy(:,Nyv) = 0;
ix(Nxv,:) = 0; iy(:,Nyv) = 0;
rx(Nxv,:) = 0; ry(:,Nyv) = 0;
```



Since there is still sufficient diffusion for the given frequency of initial infections, the final distributions are uniform and the evolution of mean values is similar to previous cases.

The next case corresponds to quarantining on a coarse grid by turning off diffusivities at every fourth interface: `n = 4`.

```
n = 4; Nxv = n:n:Nx; Nyv = n:n:Ny;
sx(Nxv,:) = 0; sy(:,Nyv) = 0;
ix(Nxv,:) = 0; iy(:,Nyv) = 0;
rx(Nxv,:) = 0; ry(:,Nyv) = 0;
```

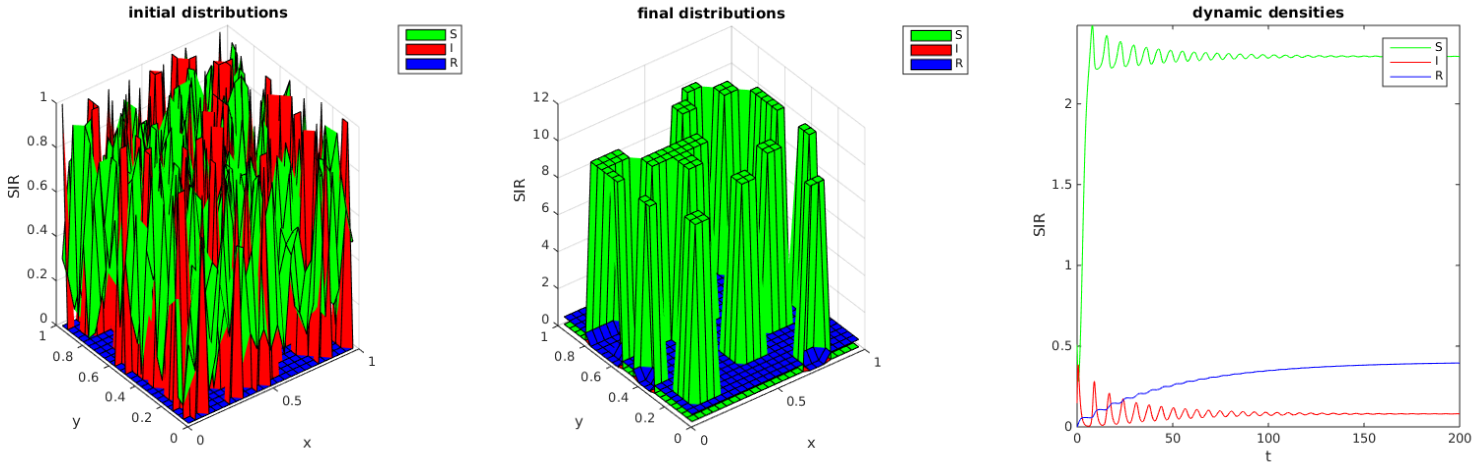


Note that in the single region in the center of the domain there happened to be no infected individuals so the susceptibles reached their capacity. The evolution of mean values shows nevertheless that the overall average number of recovered individuals exceeds the overall average numbers of susceptibles or infected.

The next case corresponds to quarantining on a coarse grid by turning off diffusivities at every third interface: `n = 3`.

```
n = 3; Nxv = n:n:Nx; Nyv = n:n:Ny;
```

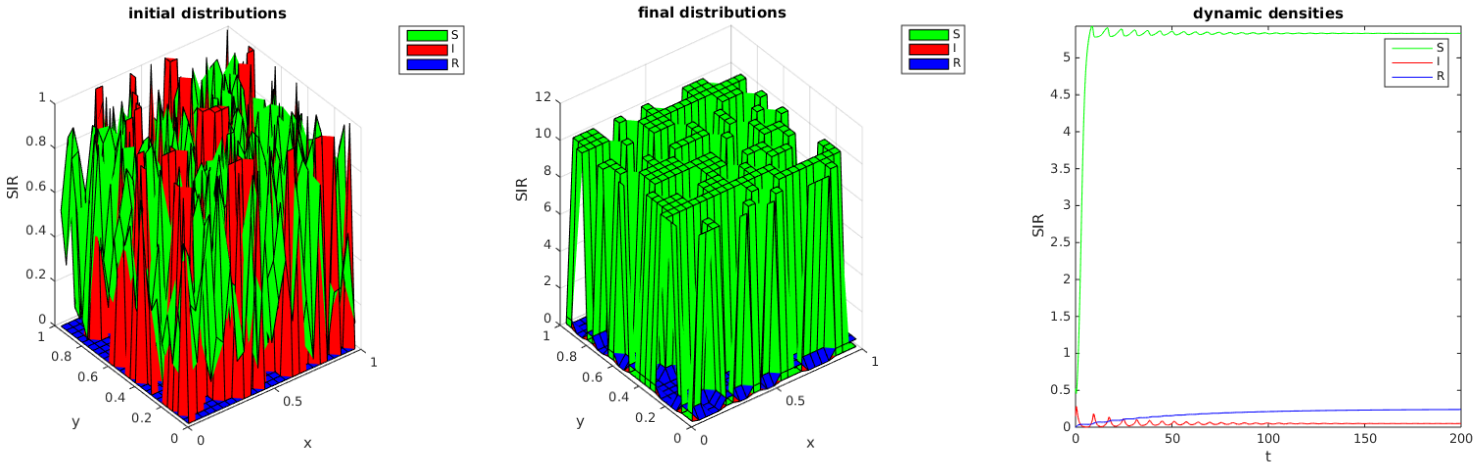
```
sx(Nxv,:) = 0; sy(:,Nyv) = 0;
ix(Nxv,:) = 0; iy(:,Nyv) = 0;
rx(Nxv,:) = 0; ry(:,Nyv) = 0;
```



Since the grid is finer here, there are several regions in which there happened to be no infected individuals. In each of these the susceptibles reached their capacity. The evolution of mean values now shows more susceptibles than recovered or infected.

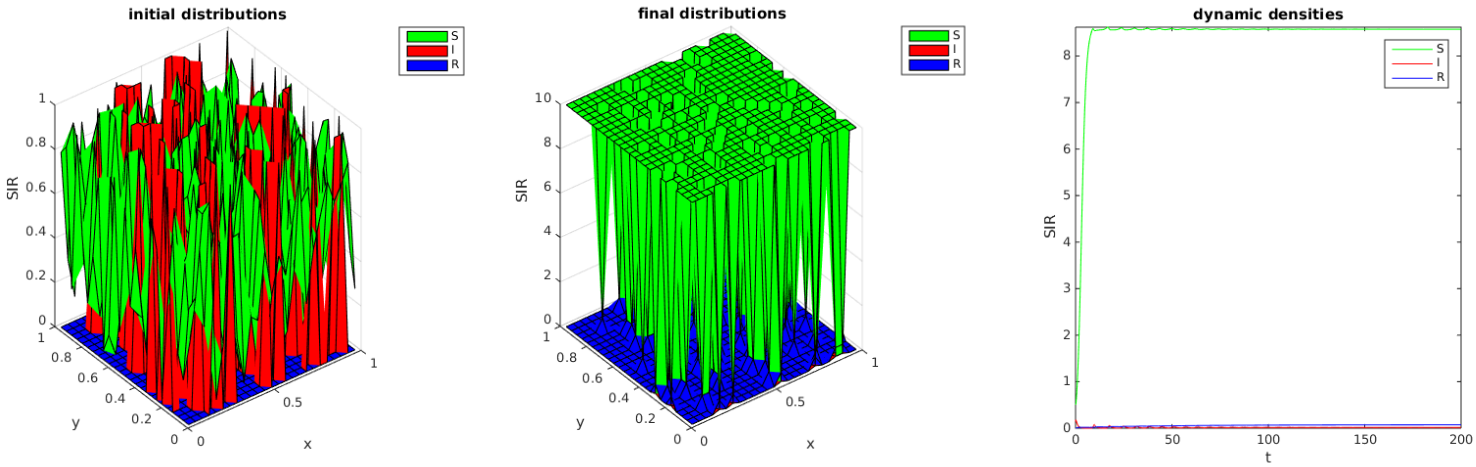
The next case corresponds to quarantining on a coarse grid by turning off diffusivities at every second interface: `n = 2`.

```
n = 2; Nxv = n:n:Nx; Nyv = n:n:Ny;
sx(Nxv,:) = 0; sy(:,Nyv) = 0;
ix(Nxv,:) = 0; iy(:,Nyv) = 0;
rx(Nxv,:) = 0; ry(:,Nyv) = 0;
```



Since the grid here is still finer, there are yet more regions in which there happened to be no infected individuals. The evolution of mean values shows that the susceptibles clearly dominate.
The next case corresponds to quarantining on a coarse grid by turning off diffusivities at every interface, `n = 1`, i.e., diffusion is set to zero everywhere.

```
n = 1; Nxv = n:n:Nx; Nyv = n:n:Ny;
sx(Nxv,:) = 0; sy(:,Nyv) = 0;
ix(Nxv,:) = 0; iy(:,Nyv) = 0;
rx(Nxv,:) = 0; ry(:,Nyv) = 0;
```



As a result, the infection is thoroughly suppressed.

Very different approaches to quarantine strategies are investigated here by Mr. Ranzinger.

• Exercise 3: Pattern Formation, Morphogenesis

◦ Task

Implement the model of morphogenesis,

$$\begin{cases} u_t &= d_1\Delta u + p - uv^2, & \Omega\times(0,T) \\ v_t &= d_2\Delta v + q - v + uv^2, & \Omega\times(0,T) \\ & u(0,y) = u(1,y), \quad u(x,0) = u(x,1), & \partial\Omega\times(0,T) \\ & v(0,y) = v(1,y), \quad v(x,0) = v(x,1), & \partial\Omega\times(0,T) \\ u &= u_0, & \Omega\times\{0\} \\ v &= v_0, & \Omega\times\{0\} \end{cases}$$

where $\Omega = (0,1)^2$, $T > 0$, $p, q, d_1, d_2 \in \mathbb{R}$, $p > q > 0$, $(p+q)^3 > (p-q)$, $0 < d_2 \ll d_1$ and u_0 and v_0 are (random) perturbations of the equilbrium state in the absence of diffusion. Demonstrate pattern formation with this model.

◦ Solution

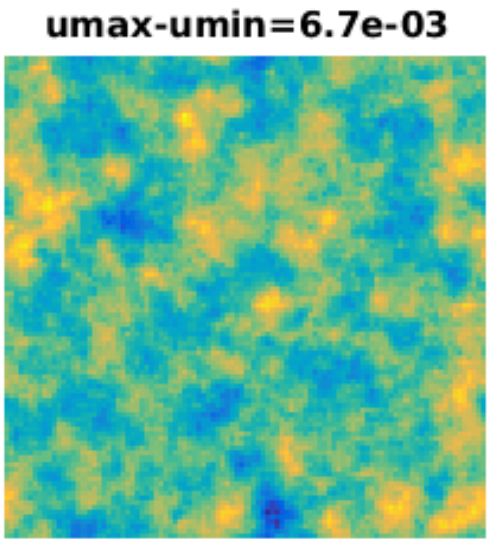
The following Matlab code is used to solve the initial and boundary value problem.

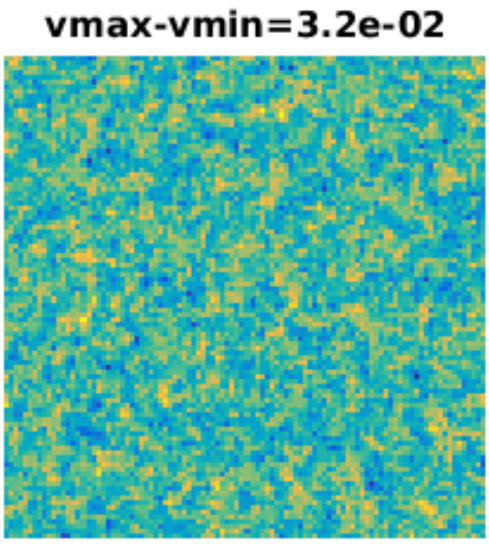
```
% setup figure
    h1 = figure(1);
    close(h1);
    h1 = figure(1);
    set(h1,'Position',[10 10 600 300]);
```

```
% temporal and geometric parameters
    T  = 1000;
    dt = 0.1; Nt = round(T/dt); dt = T/Nt;
    xmin = 0; xmax = 1;
    hx = 0.01; Nx = round((xmax - xmin)/hx); hx = (xmax - xmin)/Nx;
    x  = linspace(xmin,xmax,Nx+1); x = x(1:Nx) + hx/2;
    ymin = 0; ymax = 1;
    hy = 0.01; Ny = round((ymax - ymin)/hy); hy = (ymax - ymin)/Ny;
    y  = linspace(ymin,ymax,Ny+1); y = y(1:Ny) + hy/2;
    xx = kron(x(:),ones(1,Ny));
    yy = kron(ones(Nx,1),y(:)');

% first order derivatives with periodic BCs
    dx = spdiags(ones(Nx,1), 0,Nx,Nx) ...
       - spdiags(ones(Nx,1),-1,Nx,Nx); dx(1,Nx) = -1;
    dx = dx/hx;
    iy = speye(Ny);
    Dx = kron(iy,dx);
    dy = spdiags(ones(Ny,1), 0,Ny,Ny) ...
       - spdiags(ones(Ny,1),-1,Ny,Ny); dy(1,Ny) = -1;
    dy = dy/hy;
    ix = speye(Nx);
    Dy = kron(dy,ix);

% model parameters
% d1 >> d2 > 0
    d1 = 1.0e-2;
    d2 = 5.0e-4;
% p > q > 0, (p+q)^3 > (p-q)
    p  = 4.0e-1;
    q  = 2.0e-1;

% discrete operators
    L1 = d1*(Dx'*Dx + Dy'*Dy);
    L2 = d2*(Dx'*Dx + Dy'*Dy);
    I  = speye(Nx*Ny);

% initial conditions
    u0 = p/(p+q)^2;
    v0 = p+q;
    ns = 0.01;
    u  = u0*(1 + ns*randn(Nx,Ny)); u = u(:);
    v  = v0*(1 + ns*randn(Nx,Ny)); v = v(:);

    for k=1:Nt

% time step
```

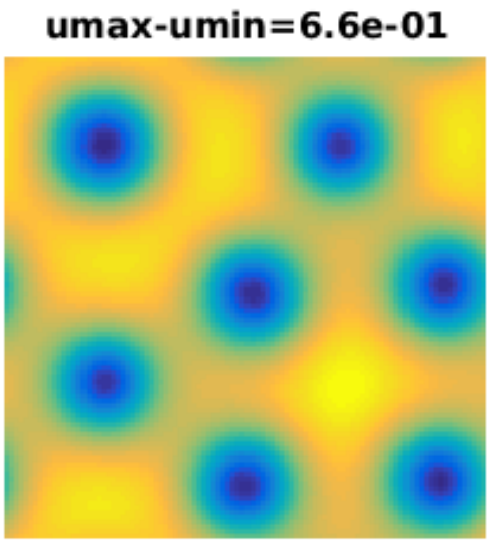
```
        f = p - u.*v.^2;
        g = q - v + u.*v.^2;
        u = (I + dt*L1) \ (u + dt*f);
        v = (I + dt*L2) \ (v + dt*g);

% plotting
        subplot(1,2,1)
        imagesc(reshape(u,Nx,Ny)); axis image; axis off;
        title(sprintf('umax-umin=%0.1e',max(u)-min(u)));
        subplot(1,2,2)
        imagesc(reshape(v,Nx,Ny)); axis image; axis off;
        title(sprintf('vmax-vmin=%0.1e',max(v)-min(v)));
        drawnow;

    end
```

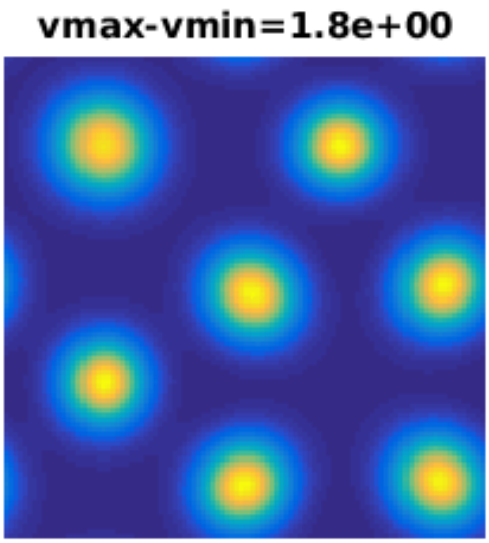
With these initial conditions on a square domain, i.e., $x_{\min} = 0 = y_{\min}$, $x_{\max} = 1 = y_{\max}$,



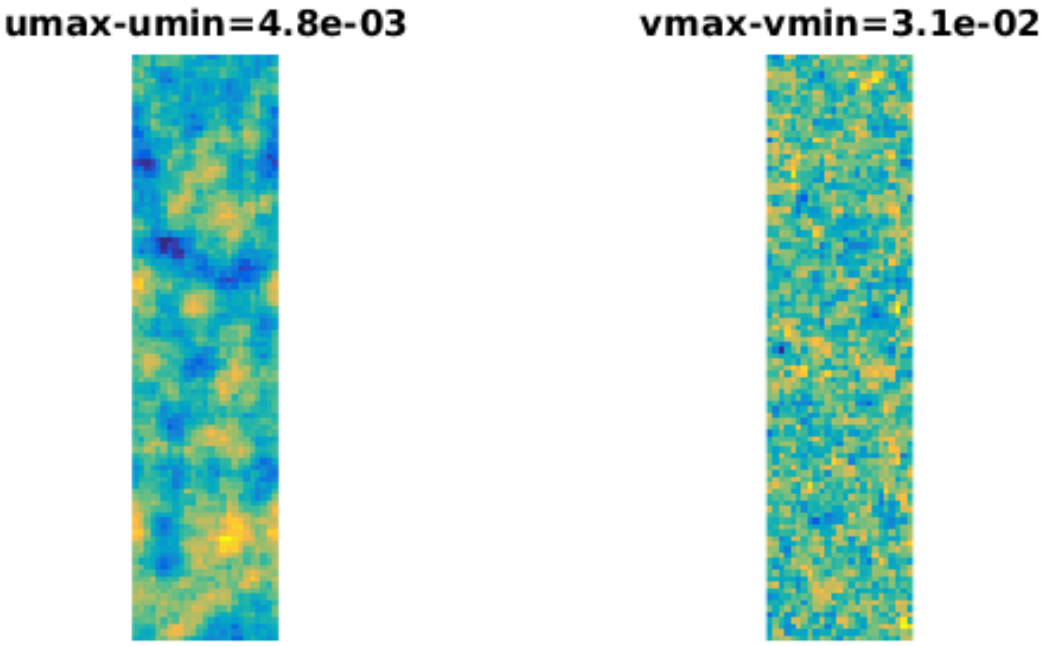


the following patterns with spots emerge:

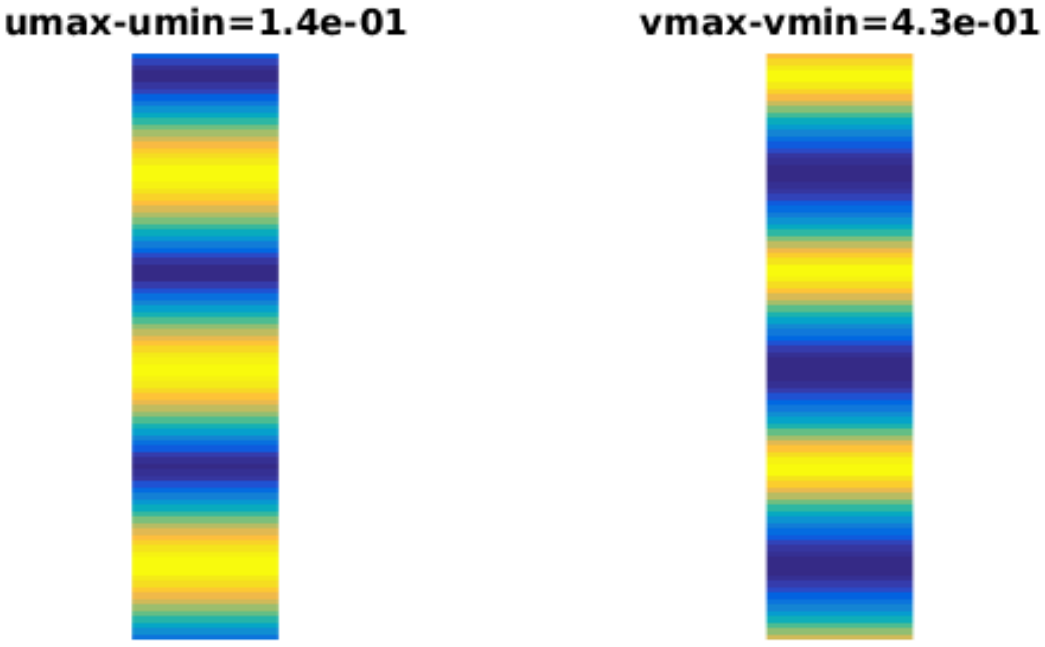




With the following initial conditions on a rectangular domain, i.e., $x_{\min} = 0 = y_{\min}$, $x_{\max} = 1$, $y_{\max} = 0.25$,



the following patterns with stripes emerge:



See also the Princeton *Shape of Mathematics* web page.

• Exercise 4: Pattern Formation, SIR

◦ Task

Implement the logistic variant of the predator-prey SIR model on page 147 of the lecture notes and demonstrate pattern formation for an appropriate choice of parameters and initial conditions.

◦ Solution

Consider the infection model for susceptibles S and infected I, which is decoupled from recovered R. The infected play the role of predators and the susceptible the role of prey. The spatial domain is $\Omega = (0,1)^2$ and the final time is $T > 0$.

$$\left\{\begin{array}{rcll} S_t &=& a_1 S(1-S/a_2) - a_3 SI/(1+S/a_4) + d_1 \Delta S, & (x,y) \in \Omega, \quad t \in (0,T] \\ I_t &=& a_5 I(1-a_6 I/S) + d_2 \Delta I, & (x,y) \in \Omega, \quad t \in (0,T] \\ S(0,y,t) = S(1,y,t), & & S(x,0,t) = S(x,1,t), & (x,y) \in \partial\Omega, \quad t \in (0,T] \\ I(0,y,t) = I(1,y,t), & & I(x,0,t) = I(x,1,t), & (x,y) \in \partial\Omega, \quad t \in (0,T] \\ S(x,y,0) = S_0(x,y), & & I(x,y,0) = I_0(x,y), & (x,y) \in \Omega, \quad t = 0 \end{array}\right.$$

With the parameter values

$$a_1 = 10, \quad a_2 = 100, \quad a_3 = 10, \quad a_4 = 10, \quad a_5 = 1, \quad a_6 = 10,$$

unstable equilibrium states are given by

$$S^\star = 5(\sqrt{41} - 1) \approx 27, \quad I^\star = (\sqrt{41} - 1)/2 \approx 2.7.$$

Perturbations of these states together with diffusivities

$$d_1 = 10^{-5} < 10^{-4} = d_2$$

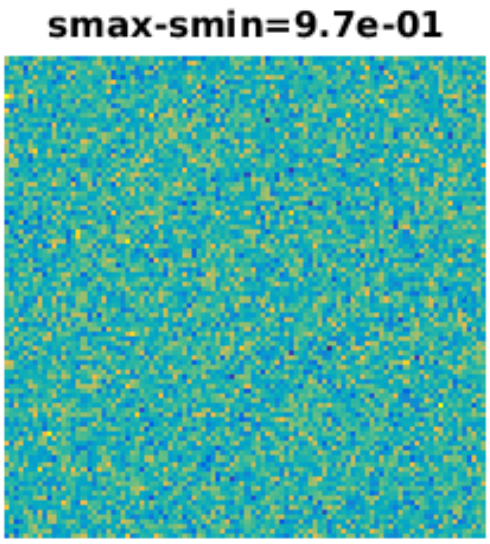
are used in the following Matlab code to solve the initial and boundary value problem and to demonstrate formation of a dynamic pattern as seen below in the graphical representation of results.

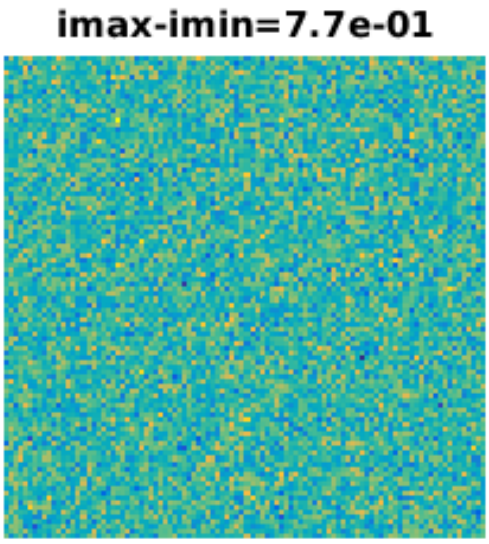
```
% setup figure
    h1 = figure(1);
    close(h1);
    h1 = figure(1);
    set(h1,'Position',[10 10 600 300]);

% temporal and geometric parameters
    T  = 1000;
    dt = 0.01; Nt = round(T/dt); dt = T/Nt;
    xmin = 0; xmax = 1;
    hx = 0.01; Nx = round((xmax - xmin)/hx); hx = (xmax - xmin)/Nx;
    x  = linspace(xmin,xmax,Nx+1);
    hx = x(2)-x(1); x = x(1:Nx) + hx/2;
    ymin = 0; ymax = 1;
    hy = 0.01; Ny = round((ymax - ymin)/hy); hy = (ymax - ymin)/Ny;
    y  = linspace(ymin,ymax,Ny+1);
    hy = y(2)-y(1); y = y(1:Ny) + hy/2;
```

```
    xx = kron(x(:),ones(1,Ny));
    yy = kron(ones(Nx,1),y(:)');

% first order derivatives with periodic BCs
    dx = spdiags(ones(Nx,1), 0,Nx,Nx) ...
       - spdiags(ones(Nx,1),-1,Nx,Nx); dx(1,Nx) = -1;
    dx = dx/hx;
    iy = speye(Ny);
    Dx = kron(iy,dx);
    dy = spdiags(ones(Ny,1), 0,Ny,Ny) ...
       - spdiags(ones(Ny,1),-1,Ny,Ny); dy(1,Ny) = -1;
    dy = dy/hy;
    ix = speye(Nx);
    Dy = kron(dy,ix);

% model parameters
    a1 = 10;
    a2 = 100;
    a3 = 10;
    a4 = 10;
    a5 = 1;
    a6 = 10;
    d1 = 1.0e-5;
    d2 = 1.0e-4;

% discrete operators
    L1 = d1*(Dx'*Dx + Dy'*Dy);
    L2 = d2*(Dx'*Dx + Dy'*Dy);
    I  = speye(Nx*Ny);

% initial conditions
    s0 = 27;
    i0 = 2.7;
    ns = 0.01;
    s  = s0*(1 + ns*randn(Nx,Ny)); s = s(:);
    i  = i0*(1 + ns*randn(Nx,Ny)); i = i(:);

    for k=1:Nt

% time step
        f = a1*s.*(1 - s/a2) - a3*s.*i./(1 + s/a4);
        g = a5*i.*(1 - a6*i./s);
        s = (I + dt*L1) \ (s + dt*f);
        i = (I + dt*L2) \ (i + dt*g);

% plotting
        subplot(1,2,1)
        imagesc(reshape(s,Nx,Ny)); axis image; axis off;
```

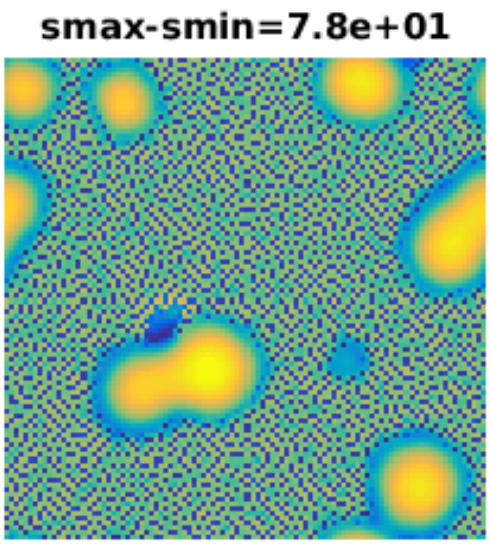
```
    title(sprintf('smax-smin=%0.1e',max(s)-min(s)));
    subplot(1,2,2)
    imagesc(reshape(i,Nx,Ny)); axis image; axis off;
    title(sprintf('imax-imin=%0.1e',max(i)-min(i)));
    drawnow;

end
```

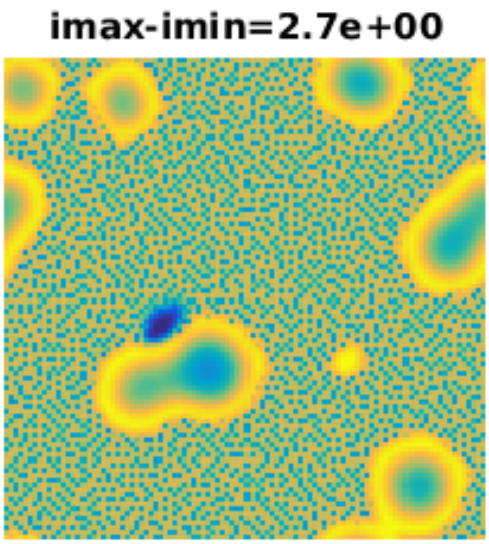
The initial conditions are displayed graphically as





and later the following dynamic pattern emerges, which cyclically changes in the concentrated regions:





• Exercise 5: Swarm Intelligence

◦ Task

Based upon the work with pattern formation implement a model of swarm intelligence.

◦ Solution

Consider the model of swarm behavior of termites for building a nest, which is presented in the work,

> Eric Bonabeau, Guy Theraulaz, J. Deneubourg, Nigel R. Franks, Oliver Rafelsberger, J. Joly and Stephane Blanco. *A model for the emergence of pillars, walls and royal chambers in termite nests.* Philosophical Transactions of the Royal Society of London B: Biological Sciences, Vol. 353, No. 1375, pp. 1561-1576, 1998.

In this model termites are carrying building material, and their movement is guided by the presence of a pheromone. Let $C(x, y, t)$ denote the density of termites loaded with building material. Let $H(x, y, t)$ be the concentration of the pheromone which is emitted as material is deposited and which diffuses freely in the air. Let P represent the amount of deposited material that is still actively emitting pheromone. The interaction among these variables is modelled according to the reaction diffusion equation,

$$\begin{cases} H_t &= k_2P - k_4H + D_H\Delta H, & (x,y)\in\Omega, & t\in(0,T] \\ C_t &= \Phi - k_1C + D_C\Delta C - \gamma\nabla\cdot(C\nabla H), & (x,y)\in\Omega, & t\in(0,T] \\ P_t &= k_1C - k_2P, & (x,y)\in\Omega, & t\in(0,T] \\ & H(0,y,t) = H(1,y,t), \quad H(x,0,t) = H(x,1,t), & (x,y)\in\partial\Omega, & t\in(0,T] \\ & C(0,y,t) = C(1,y,t), \quad C(x,0,t) = C(x,1,t), & (x,y)\in\partial\Omega, & t\in(0,T] \\ H(x,y,0) = H_0(x,y), & \quad C(x,y,0) = C_0(x,y), \quad P(x,y,0) = P_0(x,y), & (x,y)\in\Omega, & t=0. \end{cases}$$

In the partial differential equation for H, k_2P is the amount of pheromone emitted from deposited material P per unit time, so k_2 is the emission rate. The pheromone decay is represented by $-k_4H$, and $D_H\Delta H$ accounts for pheromone diffusion with diffusivity D_H.

In the partial differential equation for C, the attractiveness of the pheromone is described by $-\gamma\nabla\cdot(C\nabla H)$, which is based upon Fick's law driven by the pheremone gradient $-\gamma\nabla H$. The otherwise random motion of termites is modelled by $D_C\Delta C$ with diffusivity D_C. A spatially and temporally constant flux of termites bringing new building material into the modelled zone is denoted by Φ, and k_1 is the rate at which material is unloaded.

In the pointwise ordinary differential equation for P, k_1C is the amount of material P deposited per unit time, and the loss rate of P is the amount of pheromone emitted per unit time k_2P.

The following Matlab code is used to solve the initial and boundary value problem.

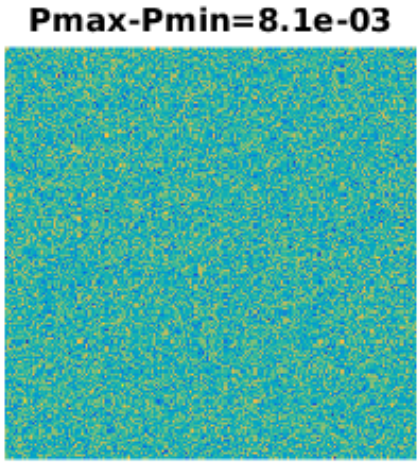
```
% setup figure
    h1 = figure(1);
    close(h1);
    h1 = figure(1);
    set(h1,'Position',[10 10 900 300]);

% temporal and geometric parameters
    T  = 1000;
    dt = 0.0003; dt = 0.1; Nt = round(T/dt); dt = T/Nt;
```

```
    xmin = 0; xmax = 1.8;
    hx = 0.01; Nx = round((xmax - xmin)/hx); hx = (xmax - xmin)/Nx;
    x  = linspace(xmin,xmax,Nx+1); x = x(1:Nx) + hx/2;
    ymin = 0; ymax = 1.8;
    hy = 0.01; Ny = round((ymax - ymin)/hy); hy = (ymax - ymin)/Ny;
    y  = linspace(ymin,ymax,Ny+1); y = y(1:Ny) + hy/2;
    xx = kron(x(:),ones(1,Ny));
    yy = kron(ones(Nx,1),y(:)');
    NxNy = Nx*Ny;

% first order derivatives with periodic BCs
    dx = spdiags(ones(Nx,1), 0,Nx,Nx) ...
       - spdiags(ones(Nx,1),-1,Nx,Nx); dx(1,Nx) = -1;
    dx = dx/hx;
    iy = speye(Ny);
    Dx = kron(iy,dx);
    dy = spdiags(ones(Ny,1), 0,Ny,Ny) ...
       - spdiags(ones(Ny,1),-1,Ny,Ny); dy(1,Ny) = -1;
    dy = dy/hy;
    ix = speye(Nx);
    Dy = kron(dy,ix);

% averaging from cells to interfaces
    ax = spdiags(ones(Nx,1), 0,Nx,Nx) ...
       + spdiags(ones(Nx,1),-1,Nx,Nx); ax(1,Nx) = +1;
    ax = ax/2;
    iy = speye(Ny);
    Ax = kron(iy,ax);
    ay = spdiags(ones(Ny,1), 0,Ny,Ny) ...
       + spdiags(ones(Ny,1),-1,Ny,Ny); ay(1,Ny) = +1;
    ay = ay/2;
    ix = speye(Nx);
    Ay = kron(ay,ix);

% model parameters
    k1 = 0.8888;
    k2 = 0.8888;
    k4 = 0.8888;
    DC = 0.01;
    DH = 0.000625;
    Ph = 3;
    ga = 0.004629;
    th = 0.5;                % th=0.5, Crank-Nicholson; th=1, backward Euler

% discrete operators
    LH = DH*(Dx'*Dx + Dy'*Dy);
    LC = DC*(Dx'*Dx + Dy'*Dy);
    I  = speye(Nx*Ny);
```

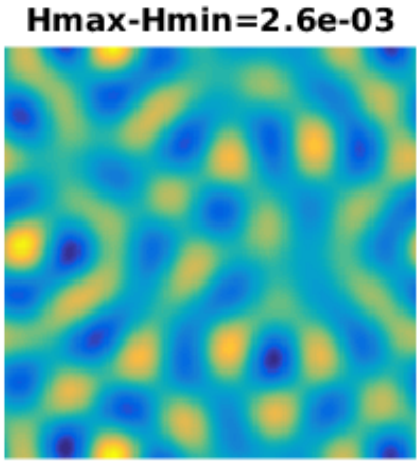
```
% initial conditions
    C0 = Ph/k1;          % equilibrium
    H0 = Ph/k4;          % state
    P0 = Ph/k2;          % with
    ns = 0.01;           % noise
    H  = H0*(1 + ns*randn(Nx,Ny)); H = H(:);
    C  = C0*(1 + ns*randn(Nx,Ny)); C = C(:);
    P  = P0*(1 + ns*randn(Nx,Ny)); P = P(:);
    H  = zeros(Nx*Ny,1);
    C  = zeros(Nx*Ny,1);
    P  = 2.333 + 0.001*randn(Nx*Ny,1);

    for k=1:Nt

% time step
% Ht = k2 P - k4 H - LH H
% Ct = Ph - k1 C - LH C + ga (Dx'(Ax C)Dx + Dy'(Ay C)Dy)H
% Pt = k1 C - k2 P

        Cx = spdiags(Ax*C,0,NxNy,NxNy);
        Cy = spdiags(Ay*C,0,NxNy,NxNy);

        dH = k2*P - k4*H;
        dC = Ph - k1*C + ga*(Dx'*Cx*Dx + Dy'*Cy*Dy)*H;
        dP = k1*C;

        H  = (I + th*dt*LH) \ ((I - (1-th)*dt*LH)*H + dt*dH);
        C  = (I + th*dt*LC) \ ((I - (1-th)*dt*LC)*C + dt*dC);
        P  = 1/(1 + th*dt*k2)*((1 - (1-th)*dt*k2)*P + dt*dP);

% plotting
        subplot(1,3,1)
        imagesc(reshape(H,Nx,Ny)); axis image; axis off;
        title(sprintf('Hmax-Hmin=%0.1e',max(H)-min(H)));
        subplot(1,3,2)
        imagesc(reshape(C,Nx,Ny)); axis image; axis off;
        title(sprintf('Cmax-Cmin=%0.1e',max(C)-min(C)));
        subplot(1,3,3)
        imagesc(reshape(P,Nx,Ny)); axis image; axis off;
        title(sprintf('Pmax-Pmin=%0.1e',max(P)-min(P)));
        drawnow;

    end
```

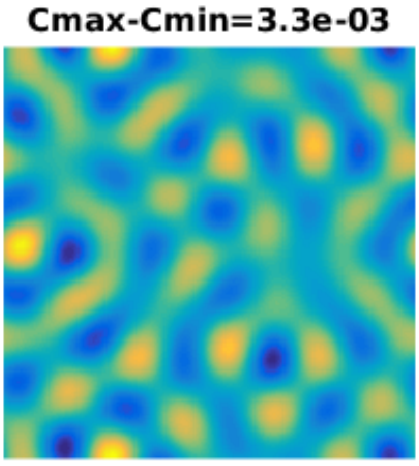
The parameters are chosen as indicated in figures 4 and 5 of the cited article. The initial conditions, $H = 0$, $C = 0$ and $P = 2.333+$ random, corresponding to those chosen and indicated in figure 4 of

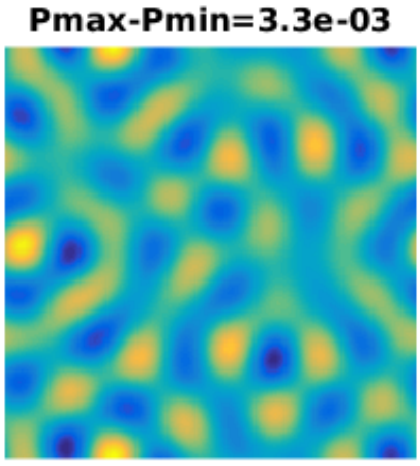
the article, are displayed graphically as follows.



A slight perturbation of this state is seen in figure 5a in the article for time $t = 4$. The time step `dt = 0.0003` and the spatial steps `hx = 0.01 = hy` are indicated in figure 4 of the article, although the authors do not write which discretizations were used. Note the finite difference approaches in the code above. A quasi-periodic state of pillars is shown in figure 5b of the article at time $t = 100$. For best efficiency, and to avoid otherwise dissipative effects, a Crank-Nicholson temporal discretization was used here with a much larger time step of `dt = 0.1` to obtain the following emergent pattern, which is comparable to that shown in the article in figure 5, although the pillar heights are much larger in the result of the article.







Very different approaches to swarm intelligence are investigated here by Ms. Freuis and Mr. Ofner.