Computational Biology 1 (FFR110), Examples sheet 3

Vitalii Iarko (930505-4711, iarko@student.chalmers.se, no ECTS)

December 10, 2014

Abstract

In this examples sheet we work with two models with diffusion. For both of them we conduct linear stability analysis of a spatially homogeneous case (without diffusion). Then we analyze traveling waves solutions and diffusion driven instability, confirming our results using numerical simulations.

14 pages overall.
The report is available online: `http://nrg3.github.io/comp_bio_3.pdf`.

Contents

1 Traveling waves.

In this task we analyze the spatial spread of the Hantavirus infection, using the model given by:

$$\frac{\partial S}{\partial t} = b(S+I) - cS - \frac{S(I+S)}{K} - aSI + D\nabla^2 S = f(S,I) + D\nabla^2 S$$

$$\frac{\partial I}{\partial t} = -cI - \frac{I(I+S)}{K} - aSI + D\nabla^2 I = g(S,I) + D\nabla^2 I$$

where a,b,c,K - positive, $b > c$.

At first we conduct linear stability analysis of the spatially homogeneous model ($D = 0$), then check whether an equation for the total population size $N = S + I$ admits a traveling wave solution and find its minimum speed. Finally, we analyze the model using numerical simulations.

1.1 a) Linear stability analysis of the spatially homogeneous model ($D = 0$).

We will do linear stability analysis of homogeneous model ($D = 0$) as described in the lecture notes [1, p. 85] (i.e. based on eigenvalues of the matrix $\underset{\sim}{A}$ - they should be smaller than 0, denote this as eigenvalues criterion).

At first we find steady states $(S^*; I^*)$ such that $\frac{\partial S^*}{\partial t} = \frac{\partial I^*}{\partial t} = 0$, they are $(0;0)$, $(K(b-c);0)$, $({}^{b}/_{a}; {}^{-b+aK(b-c)}/_{a})$, $({}^{c}/_{a}; -{}^{c}/_{a})$. Since according to the statement $a > 0$ and $c > 0$, the last one is biologically irrelevant, because its I^* is always negative.

$$\underset{\sim}{A} = \begin{pmatrix} \left.\frac{\partial f}{\partial S}\right|_{(S^*,I^*)} & \left.\frac{\partial f}{\partial I}\right|_{(S^*,I^*)} \\ \left.\frac{\partial g}{\partial S}\right|_{(S^*,I^*)} & \left.\frac{\partial g}{\partial I}\right|_{(S^*,I^*)} \end{pmatrix} = \begin{pmatrix} b - c - aI^* - \frac{S^*}{K} - \frac{I^*+S^*}{K} & b - aS^* - \frac{S^*}{K} \\ aI^* - \frac{I^*}{K} & -c - \frac{I^*}{K} + aS^* - \frac{I^*+S^*}{K} \end{pmatrix}$$

Now let's evaluate their linear stability:

1. state $(0;0)$, eigenvalues of $\underset{\sim}{A}$ are $b - c$ and $-c$ - unstable since $b > c$ according to the statement;
2. state $(K(b-c);0)$, eigenvalues are $c - b$ and $aK(b-c) - b$ - stable if $K < \frac{b}{a(b-c)}$;
3. state $({}^{b}/_{a}; {}^{-b+aK(b-c)}/_{a})$, eigenvalues are $c - b$ and $b + aK(c-b)$ - positive and stable if $K > \frac{b}{a(b-c)}$.

In order to find K_c above which both I and S in the stable steady state have positive nonzero values we need to consider the last state. It is positive and stable if $K > \frac{b}{a(b-c)}$, thus, $K_c = \frac{b}{a(b-c)}$.

1.2 b) The total population size N and traveling waves.

Let's obtain an equation of the total population size $N = S + I$:

$$\frac{\partial N}{\partial t} = \frac{\partial S}{\partial t} + \frac{\partial I}{\partial t} = b(S+I) - c(S+I) - \frac{(S+I)^2}{K} + D\nabla^2(S+I) = N(b-c) - \frac{N^2}{K} + D\nabla^2 N$$

Dimensionless variables:

$$n = \frac{N}{(b-c)K}; \quad \tau = (b-c)t; \quad \chi = \sqrt{\frac{b-c}{D}}\, x;$$

$$\frac{\partial n}{\partial \tau} = n(1-n) + \frac{\partial^2 n}{\partial \chi^2}$$

which is exactly Fisher's equation. Since it was exhaustively analyzed in the lecture notes [1, p. 159], we do not repeat the analysis here, but give only the results:

- it admits a traveling wave solution;
- minimal speed of the wave is 2 in dimensionless variables (i.e. $2\sqrt{D(b-c)}$ in dimensional).

I.e. N may propagate in a form of traveling wave over the habitat that is assumed to be initially empty, except in a narrow region that contains a small number of individuals. Its minimal speed is $2\sqrt{D(b-c)}$.

Traveling wave solution seems natural to us here. This model does not include death of individuals, they are only produced. However, there is a carrying capacity as well. Therefore, it is natural to colonize neighborhood in order to use its carrying capacity. Also one may notice light link to the spatial spread of a favoured gene in a population (and actually it is precisely this process, because Fisher equation describes it). One may consider N as a ratio of population with this gene. Then it will spread out to the neighborhood, since this gene gives advantages.

1.3 c) Numerical simulations.

Evolution of S and I vs. time t and space x are shown on Figures 1 and 2 correspondingly. As we can see population of susceptibles expand over the habitat through a traveling wave. Since they reached 50-th cell in ≈ 35 units of time, its speed is $\approx 50/35 = 1.42$. The system relaxes to the stable steady state with both nonzero susceptibles and infectives population (numerically it's $S = 10$ and $I = 5$). Infectives expand in the same manner, but with initial delay. Their speed including delay is ≈ 1, without delay - $\approx 50/38 = 1.31$.

Explanation of this infectives expansion delay is that in this model infectives only infect susceptibles and do not produce new infectives, therefore, in order to travel in space to a new place they need susceptibles there.

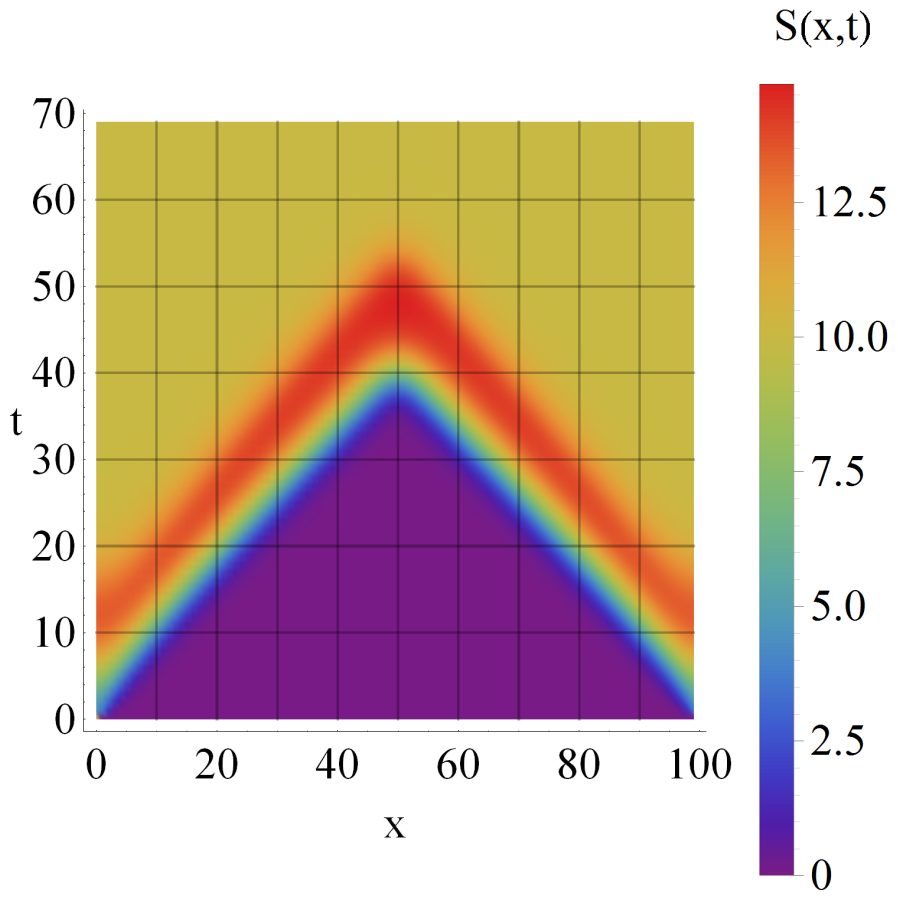


Figure 1: Evolution of S vs. time t and space x. Obtained under $\delta t = 0.01$ with periodic boundary conditions. All other parameters are as suggested in the statement.

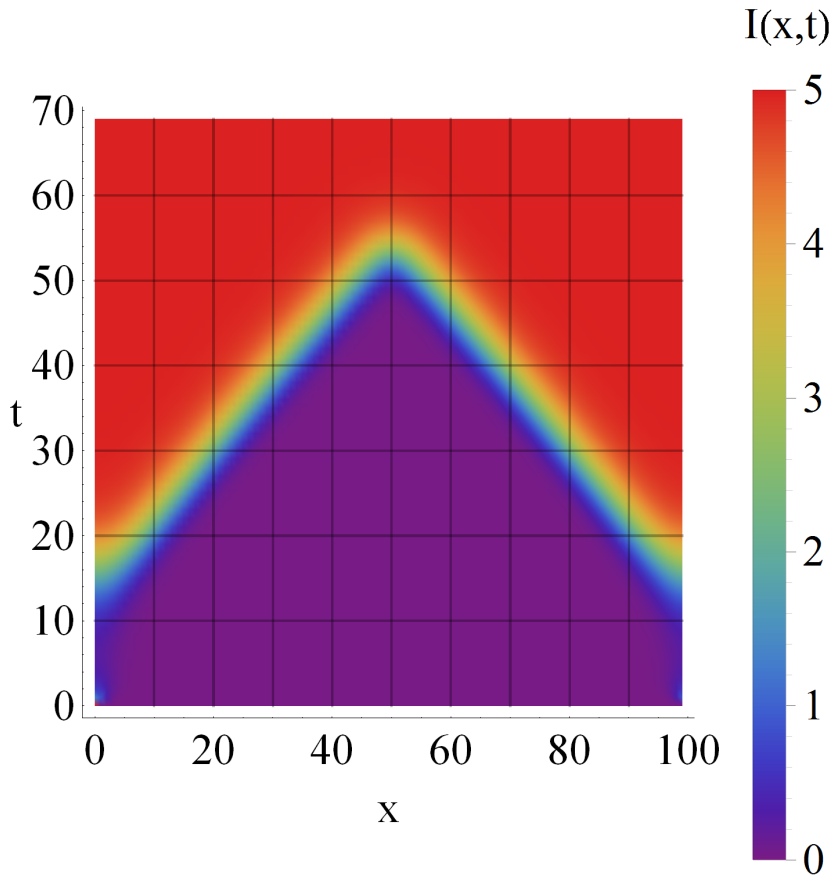


Figure 2: Evolution of I vs. time t and space x. Obtained under $\delta t = 0.01$ with periodic boundary conditions. All other parameters are as suggested in the statements.

2 Diffusion driven instability.

In this task we analyze the effect of diffusion on stability of steady states under a model for two interacting chemicals undergoing a Belousov-Zhabotinsky reaction given by

$$\frac{\partial u}{\partial t} = a - (b+1)u + u^2 v + D_u \nabla^2 u = f(u,v) + D_u \nabla^2 u$$

$$\frac{\partial v}{\partial t} = bu - u^2 v + D_v \nabla^2 v = g(u,v) + D_v \nabla^2 v$$

where $D_u, D_v, a, b > 0$.

At first we find steady states of spatially homogeneous model and then consider when they exhibit a diffusion-driven instability (in terms of value D_v). Then we do numerical simulations and analyze obtained patterns for different D_v.

2.1 a) Steady states.

Let's start with determining spatially homogeneous steady states of the system (neglecting diffusion). The condition is $f(u^*, v^*) = 0$ and $g(u^*, v^*) = 0$. This gives us one steady state $(u^*, v^*) = (a, {}^b\!/_a)$. Using eigenvalues criterion:

$$\underset{\sim}{A} = \begin{pmatrix} \left.\frac{\partial f}{\partial u}\right|_{(u^*,v^*)} & \left.\frac{\partial f}{\partial v}\right|_{(u^*,v^*)} \\ \left.\frac{\partial g}{\partial u}\right|_{(u^*,v^*)} & \left.\frac{\partial g}{\partial v}\right|_{(u^*,v^*)} \end{pmatrix} = \begin{pmatrix} -1-b+2u^*v^* & u^{*2} \\ b-2u^*v^* & -u^{*2} \end{pmatrix} = \begin{pmatrix} b-1 & a^2 \\ -b & -a^2 \end{pmatrix}$$

$$\det \underset{\sim}{A} = a^2; \quad \operatorname{tr} \underset{\sim}{A} = b - a^2 - 1;$$

$$\lambda_{1,2} = \frac{\operatorname{tr} \underset{\sim}{A} \pm \sqrt{(\operatorname{tr} \underset{\sim}{A})^2 - 4 \det \underset{\sim}{A}}}{2}$$

$$\operatorname{Re} \lambda_{1,2} < 0 \Leftrightarrow \operatorname{tr} \underset{\sim}{A} < 0 \wedge \det \underset{\sim}{A} > 0 \Leftrightarrow b < a^2 + 1$$

I.e. this state is stable if $b < a^2 + 1$.

Now we consider a diffusion-driven instability. According to the lecture notes [1, p. 204] there are four conditions when such system exhibits a diffusion-driven instability:

$$\operatorname{tr} \underset{\sim}{A} < 0; \quad d\frac{\partial f}{\partial u} + \frac{\partial g}{\partial v} > 0; \quad \det \underset{\sim}{A} > 0; \quad (d\frac{\partial f}{\partial u} + \frac{\partial g}{\partial v})^2 - 4d(\frac{\partial f}{\partial u}\frac{\partial g}{\partial v} - \frac{\partial f}{\partial v}\frac{\partial g}{\partial u}) > 0$$

with $d = D_v/D_u$.

Evaluating this at the steady state $(u^*, v^*) = (a, {}^b\!/_a)$ under $D_u = 1$, $a = 3$, $b = 8$, we obtain that $D_v > \frac{81+36\sqrt{2}}{49} \approx 2.69$.

2.2 b) Numerical simulations.

In order to simulate the dynamics, first, we initialize the field as $u(x, y, 0) = u^* \cdot \text{rand}(0.95; 1.05)$ and $v(x, y, 0) = v^* \cdot \text{rand}(0.95; 1.05) \quad \forall x, y = 0, 1, ...L-1$, where $\text{rand}(x; y)$ returns a pseudo random real number from $[x; y]$. Then, using Euler discretization and discretization of Laplacian, we obtain:

$$\frac{u(x, y, t+\delta t)}{\delta t} = f(u(x,y,t), v(x,y,t)) - 4u(x,y,t) + \sum_{(nx;ny)\in n(x,y)} u(nx, ny, t)$$

$$\frac{v(x,y,t+\delta t)}{\delta t} = g(u(x,y,t),v(x,y,t)) + D_v[-4v(x,y,t) + \sum_{(nx;ny)\in n(x,y)} v(nx,ny,t)]$$

where $n(x,y)$ is a set of neighbors of $(x;y)$ under periodic boundary conditions:

$$n(x,y) = \{(nx,ny) : (|x-nx| = 1 \vee |x-nx| = L-1) \wedge (|y-ny| = 1 \vee |y-ny| = L-1)\}$$

All simulations were done with $\delta t = 0.001$.

Results of simulation are shown on Figures 3 - 5.

As one can see, they consist with theoretical results above - under $D_v = 2.3$ technically we got patterns, but the difference between maximum and minimum of u is insignificant and this may be caused by numerical errors. Under $D_v \geq 3$ we got patterns with significant amplitude.

These patterns consist of one big connected region (background) and many smaller regions (islands), inside each region the value of u is almost the same. For smaller value $D_v = 3$ background corresponds to higher values, but then this changes - for $D_v = 5$, $D_v = 9$ background is formed by lower values. Also D_v influences appearances of islands - under $D_v = 3$ they are more connected and have greater area than for higher values of D_v.

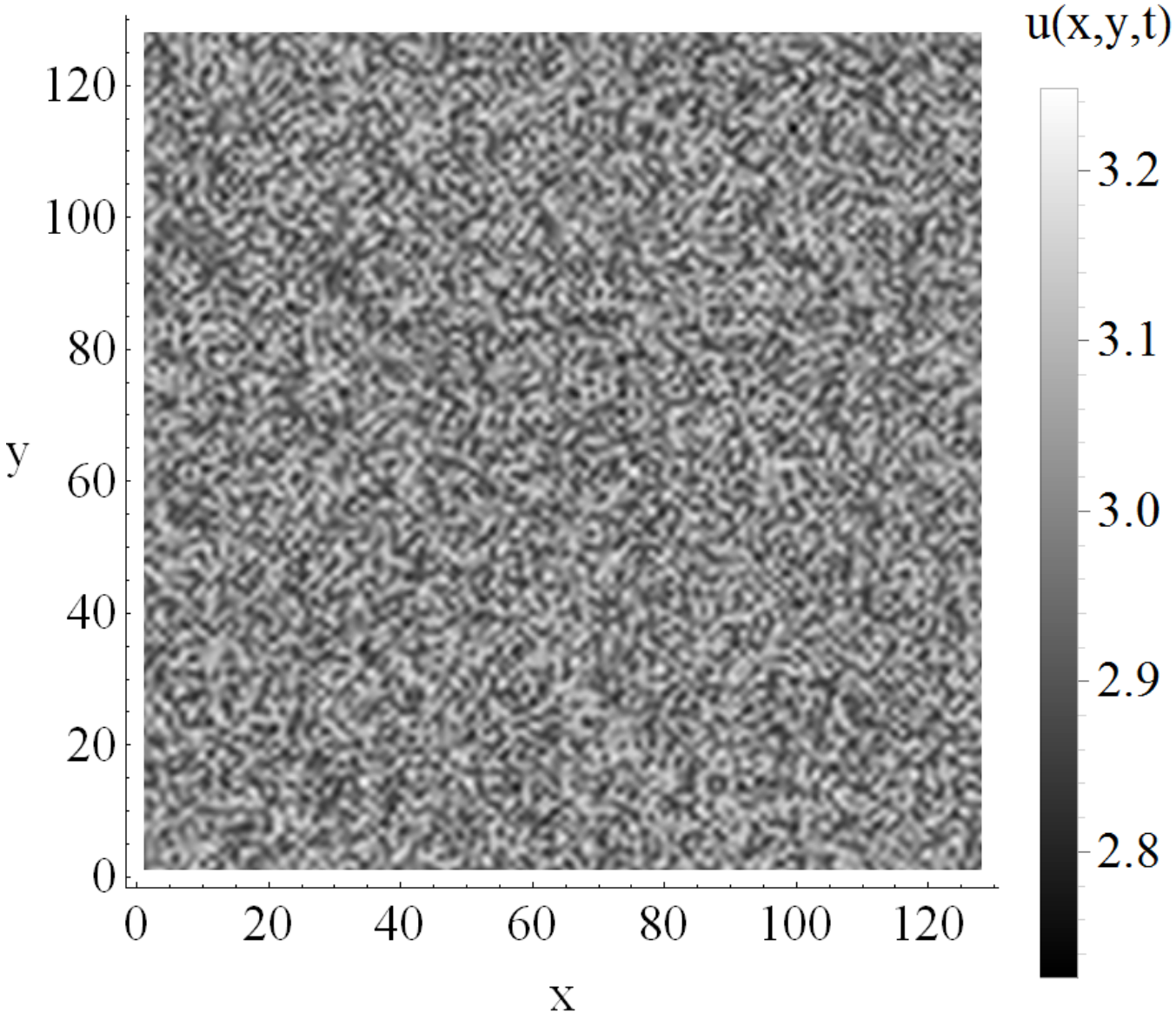
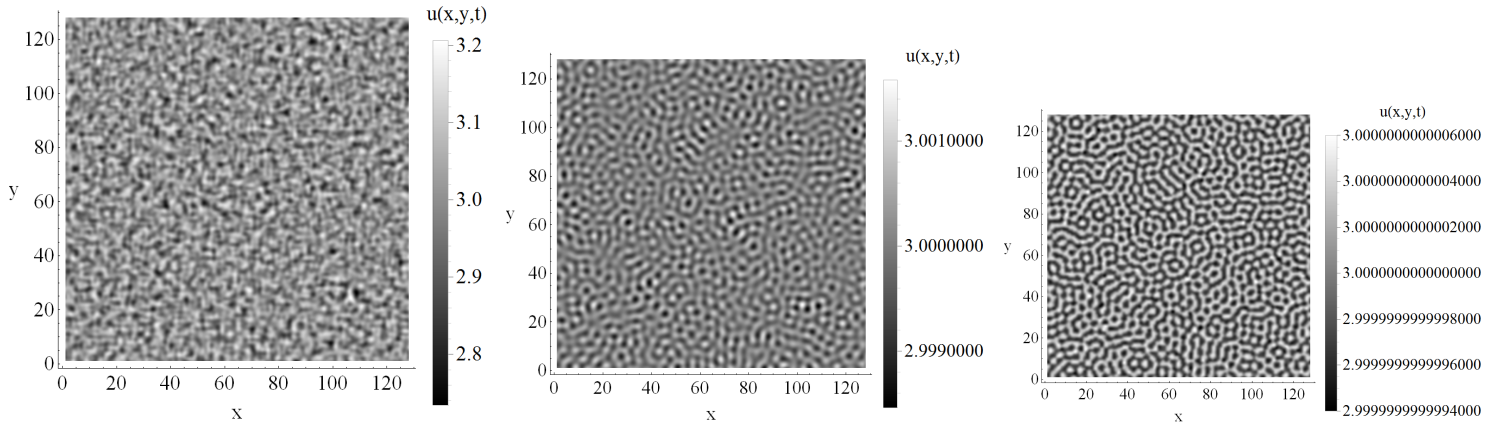
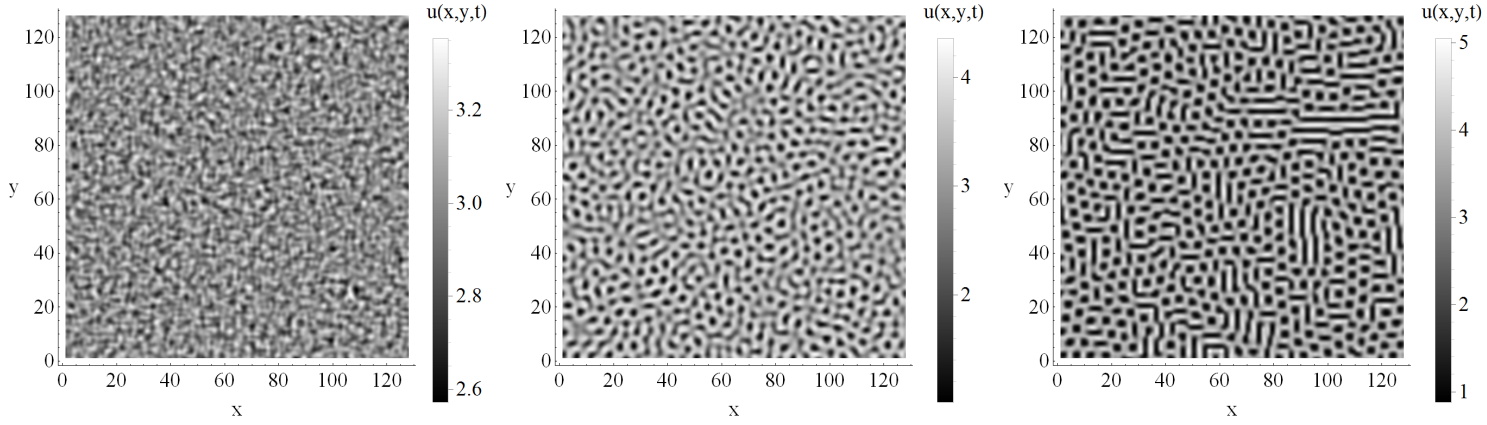


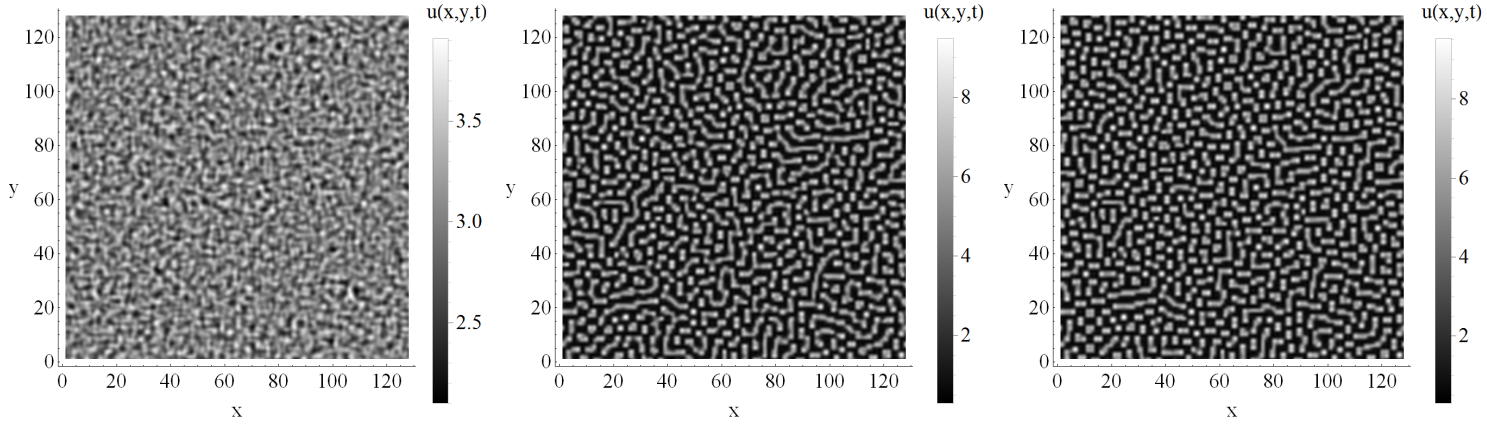
Figure 3: Initial u at $t = 0$ under seed 123.



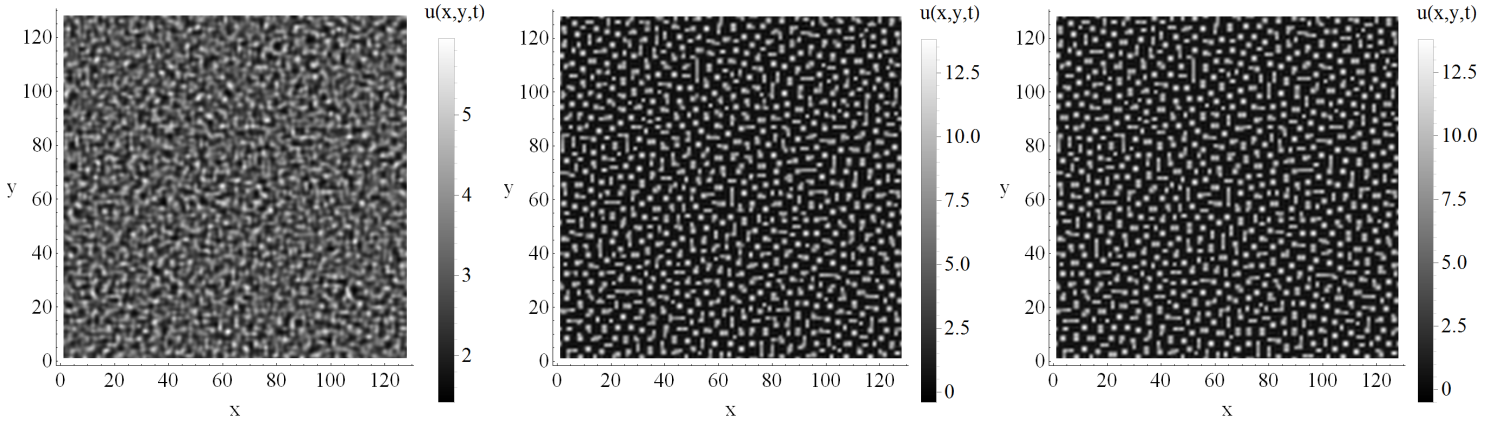
(a) $D_v = 2.3$, $t = 1$ (b) $D_v = 2.3$, $t = 10$ (c) $D_v = 2.3$, $t = 100$



(d) $D_v = 3$, $t = 1$ (e) $D_v = 3$, $t = 10$ (f) $D_v = 3$, $t = 100$

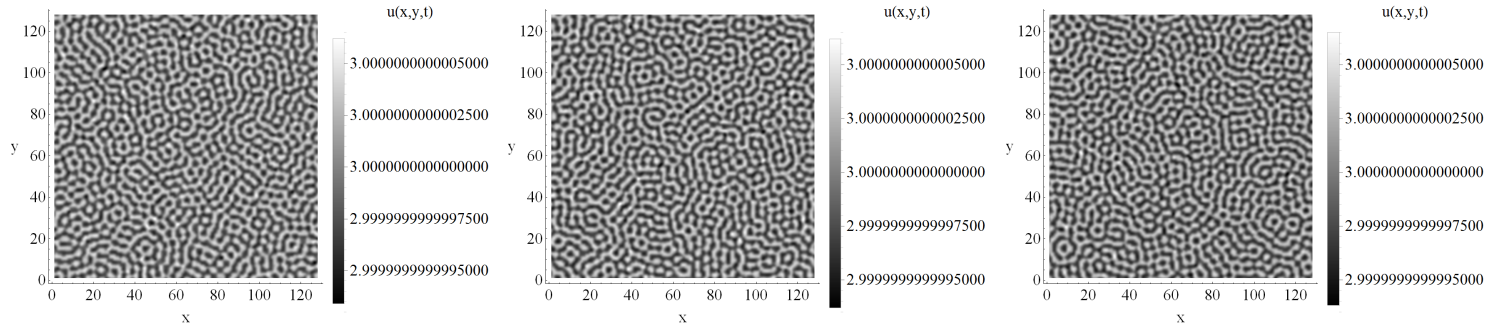


(g) $D_v = 5$, $t = 1$ (h) $D_v = 5$, $t = 10$ (i) $D_v = 5$, $t = 100$

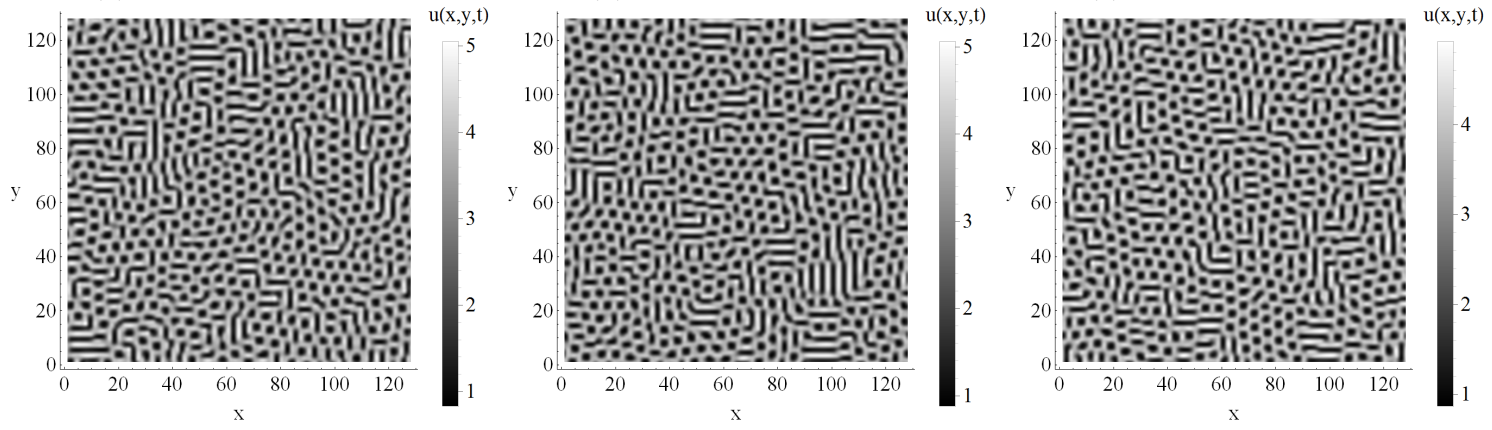


(j) $D_v = 9$, $t = 1$ (k) $D_v = 9$, $t = 10$ (l) $D_v = 9$, $t = 100$

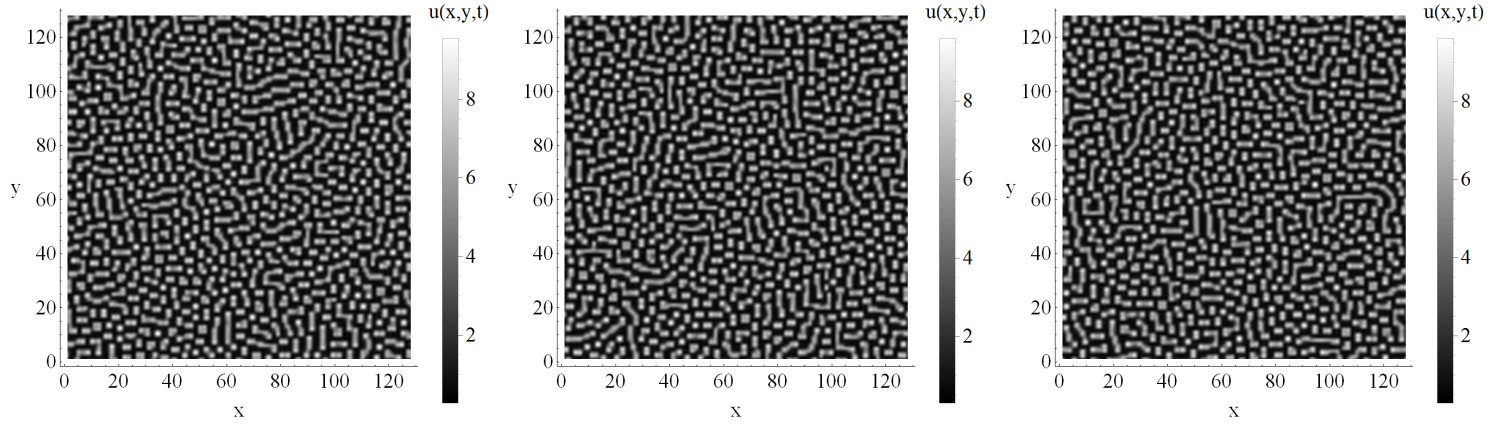
Figure 4: Dynamics of u vs. D_v and time t, seed 123. Other parameters are as suggested in the statement.



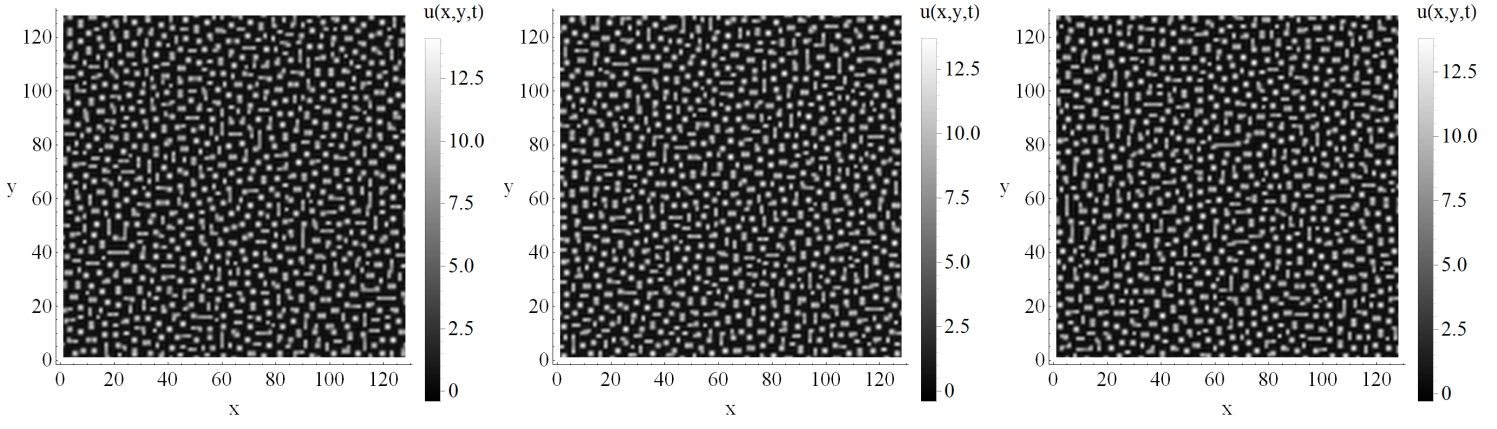
(a) $D_v = 2.3$, $seed = 124$ (b) $D_v = 2.3$, $seed = 125$ (c) $D_v = 2.3$, $seed = 126$



(d) $D_v = 3$, $seed = 124$ (e) $D_v = 3$, $seed = 125$ (f) $D_v = 3$, $seed = 126$



(g) $D_v = 5$, $seed = 124$ (h) $D_v = 5$, $seed = 125$ (i) $D_v = 5$, $seed = 126$



(j) $D_v = 9$, $seed = 124$ (k) $D_v = 9$, $seed = 125$ (l) $D_v = 9$, $seed = 126$

Figure 5: Results for other seeds: u vs. D_v, $t = 100$. Other parameters are as suggested in the statement.

References

[1] Bernhard Mehlig, *Computational Biology A: growth, morphogenesis, and ecological problems (lecture notes)*;

A File `1c.nb`:

```
1
2  codeDirectory = NotebookDirectory[];
3  CreateDirectory[codeDirectory <>"..\\report\\pics\\"];//Quiet
4  SetDirectory[codeDirectory <>"..\\report\\pics\\"];
5  hack=Function[x, (* for static relative sizes of elements in plots while rasterizing *)
6  First@ImportString[ExportString[x,"PDF"]]
7  ];
8  a=0.1;b=1;c=0.5;k=30;d=1;l=100;
9  dS=Function[{s,i},b(s+i)-c*s-s (i+s)/k-a*s*i];
10 dI=Function[{s,i},-c*i-i (i+s)/k+a*s*i];
11 Clear[s,i];
12 solutions=Solve[{dS[s,i]==0,dI[s,i]==0},{s,i}];
13 dt=0.01;
14 endTime=70;
15 patches=Table[{0,0},{l}];
16 delta=Table[{0,0},{l}];
17 patches[[1]]={s,i}/.solutions[[3]];
18
19 pointsS=Table[{0,0,0},{(endTime+1)*l}];
20 pointsI=pointsS;
21
22 last=1;
23
24 For[time=0, time <= endTime,time+=dt,
25
26 If[Mod[Floor[time*100],100]==0,
27 For[z = 1, z <= l, ++z,
28 pointsS[[last]][[1]]=z-1;
29 pointsS[[last]][[2]]=time;
30 pointsS[[last]][[3]]=patches[[z]][[1]];
31
32 pointsI[[last]][[1]]=z-1;
33 pointsI[[last]][[2]]=time;
34 pointsI[[last]][[3]]=patches[[z]][[2]];
35 ++last;
36
37
38 ];
39 ];
40
41
42 For[j=1, j <= Length[patches],++j,
43 s=patches[[j]][[1]];
44 i=patches[[j]][[2]];
45
46
47 If[j== 1, sLeft=patches[[l]][[1]],sLeft=patches[[j-1]][[1]]];
48 If[j== 1, iLeft=patches[[l]][[2]],iLeft=patches[[j-1]][[2]]];
49
50 If[j==l, sRight=patches[[1]][[1]],sRight=patches[[j+1]][[1]]];
51 If[j==l, iRight=patches[[1]][[2]],iRight=patches[[j+1]][[2]]];
52
53 delta[[j]][[1]] =dt*(dS[s,i]+sLeft-2*s+sRight);
```

```
54 delta[[j]][[2]] = dt*(dI[s,i]+iLeft-2*i+iRight);
55 ];
56
57 patches+=delta;
58 ];
59 pic1=ListDensityPlot[pointsS,Frame->{True,True,False,False},Mesh->{Table[i,{i,10,l,10}],Table
       [i,{i,10,endTime,10}]},FrameLabel->{"x","t"},FrameStyle->18,RotateLabel->False,(*
       ColorFunction\[Rule]GrayLevel,*)ColorFunction->"Rainbow",InterpolationOrder->5,
60 PlotLegends->Placed[BarLegend[Automatic,LegendMargins->{{0,0},{0,0}},LegendLabel->"S(x,t)",
       LabelStyle->{FontSize->18},LegendMarkerSize->350],Right]
61 ];
62 Export["1c_s_color.png",pic1,ImageResolution->300];
63 pic1=ListDensityPlot[pointsS,Frame->{True,True,False,False},Mesh->{Table[i,{i,10,l,10}],Table
       [i,{i,10,endTime,10}]},FrameLabel->{"x","t"},FrameStyle->18,RotateLabel->False,
       ColorFunction->GrayLevel,InterpolationOrder->5,
64 PlotLegends->Placed[BarLegend[Automatic,LegendMargins->{{0,0},{0,0}},LegendLabel->"S(x,t)",
       LabelStyle->{FontSize->18},LegendMarkerSize->350],Right]
65 ];
66 Export["1c_s_gray.png",pic1,ImageResolution->300];
67
68 pic1=ListDensityPlot[pointsI,Frame->{True,True,False,False},Mesh->{Table[i,{i,10,l,10}],Table
       [i,{i,10,endTime,10}]},FrameLabel->{"x","t"},FrameStyle->18,RotateLabel->False,(*
       ColorFunction\[Rule]GrayLevel,*)ColorFunction->"Rainbow",InterpolationOrder->5,
69 PlotLegends->Placed[BarLegend[Automatic,LegendMargins->{{0,0},{0,0}},LegendLabel->"I(x,t)",
       LabelStyle->{FontSize->18},LegendMarkerSize->350],Right]
70 ];
71 Export["1c_i_color.png",pic1,ImageResolution->300];
72 pic1=ListDensityPlot[pointsI,Frame->{True,True,False,False},Mesh->{Table[i,{i,10,l,10}],Table
       [i,{i,10,endTime,10}]},FrameLabel->{"x","t"},FrameStyle->18,RotateLabel->False,
       ColorFunction->GrayLevel,InterpolationOrder->5,
73 PlotLegends->Placed[BarLegend[Automatic,LegendMargins->{{0,0},{0,0}},LegendLabel->"I(x,t)",
       LabelStyle->{FontSize->18},LegendMarkerSize->350],Right]
74 ];
75 Export["1c_i_gray.png",pic1,ImageResolution->300];
```

B File 2b.cpp:

```
1
2 #include <iostream>
3 #include <fstream>
4
5 #include <vector>
6 using std::vector;
7
8 #include <string>
9 using std::string;
10
11 #include <sstream>
12 #include <direct.h>
13
14 typedef long double ValueType;
15
16 ValueType random() {
17   return static_cast<ValueType>(rand()) / RAND_MAX;
18 }
19
20 ValueType dif_V;
21 const ValueType a = 3, b = 8;
22
23 ValueType f(ValueType U, ValueType V) {
```

```
24   return a - U * (b+1) + U * U * V;
25 }
26
27 ValueType g(ValueType U, ValueType V) {
28   return b * U - U * U * V;
29 }
30
31 string to_string(int x) {
32   std::stringstream stream;
33   stream << x;
34   return stream.str();
35 }
36
37 void save(const vector< vector<ValueType> >& data, const string& path) {
38   std::ofstream out(path);
39
40   out.precision(15);
41   out << std::fixed;
42
43   for (int i = 0; i < data.size(); ++i) {
44     for (int j = 0; j < data[i].size(); ++j) {
45       if (j) {
46         out << " ";
47       }
48       out << data[i][j];
49     }
50     out << "\n";
51   }
52
53   out.close();
54 }
55
56 int main() {
57
58   srand(123);
59
60   ValueType dt = 0.001;
61   int L = 128;
62   ValueType pertubation = 0.1;
63   int iters = 105000;
64   ValueType end_time = dt * iters;
65
66
67   ValueType dif_Vs[] = {2.3, 3, 5, 9};
68   vector< vector< vector<ValueType> > > field(2, vector< vector<ValueType> >(L, vector<
       ValueType>(L)));
69   vector< vector< vector<ValueType> > > next = field;
70   ValueType start_U = 3, start_V = 8. / 3;
71
72   int seeds = 5;
73
74   int all = iters * 4 * seeds;
75   int done = 0;
76
77   ValueType max_dif = 1e300;
78
79   mkdir("txt");
80   for (int id = 0; id < 4; ++id) {
81     mkdir(string("txt/" + to_string(id)).c_str());
82     for (int seed = 0; seed < 5; ++seed) {
```

```
83      mkdir(string("txt/" + to_string(id) + "/" + to_string(seed + 123)).c_str());
84      srand(seed + 123);
85
86      dif_V = dif_Vs[id];
87
88
89      for (int j = 0; j < L; ++j) {
90        for (int z = 0; z < L; ++z) {
91          field[0][j][z] = start_U * (1 + (random() - 0.5) * pertubation);
92          field[1][j][z] = start_V * (1 + (random() - 0.5) * pertubation);
93        }
94      }
95
96      int dx[] = {0, 0, -1, 1};
97      int dy[] = {-1, 1, 0, 0};
98
99      int last_save = -1e9;
100     int saving_time = 1000;
101
102     int iter = 0;
103
104
105     max_dif = 1e300;
106     for (ValueType time = 0; time < end_time; time += dt) {
107
108
109
110       if (iter - last_save >= saving_time) {
111         last_save = iter;
112         save(field[0], "txt/"+ to_string(id) + "/" + to_string(seed + 123) + "/"
113                   + to_string(int(dif_V * 100)) + "_" + to_string(iter) + ".txt");
114
115         std::cerr << "\r" << 1. * done / all << " " << iter << " " << max_dif << "          \
    r";
116
117         /*
118         if (max_dif < 1e-12) {
119           done += iters - iter;
120           break;
121         }
122         */
123       }
124
125
126       max_dif = 0;
127
128       ++iter;
129       ++done;
130
131       for (int i = 0; i < L; ++i) {
132         for (int j = 0; j < L; ++j) {
133
134           ValueType sum_U = 0, sum_V = 0;
135
136           for (int z = 0; z < 4; ++z) {
137             int nx = i + dx[z];
138             int ny = j + dy[z];
139             if (nx < 0) {
140               nx = L - 1;
141             }
```

```
142                 if (nx >= L) {
143                   nx = 0;
144                 }
145                 if (ny < 0) {
146                   ny = L - 1;
147                 }
148                 if (ny >= L) {
149                   ny = 0;
150                 }
151
152                 sum_U += field[0][nx][ny];
153                 sum_V += field[1][nx][ny];
154               }
155
156               ValueType dU = (f(field[0][i][j], field[1][i][j]) + sum_U - 4 * field[0][i][j]);
157               ValueType dV = (g(field[0][i][j], field[1][i][j]) + dif_V *(sum_V - 4 * field[1][
      i][j]));
158
159               next[0][i][j] = field[0][i][j] + dt * dU;
160               next[1][i][j] = field[1][i][j] + dt * dV;
161
162               if (max_dif < fabs(dU)) {
163                 max_dif = fabs(dU);
164               }
165               if (max_dif < fabs(dV)) {
166                 max_dif = fabs(dV);
167               }
168             }
169
170
171
172           }
173           field.swap(next);
174         }
175       }
176     }
177 }
```

C File 2b.nb:

```
1  Dvs={"2.3","3","5","9"};
2  CreateDirectory["D:\\projects\\compbio3\\pics\\"]; //Quiet
3  For[id=1, id <=4, ++id,
4  For[seed = 0, seed<5,++seed,
5  folder="D:\\projects\\compbio3\\txt\\"<>ToString[id-1]<>"\\"<>ToString[seed+123]<>"\\";
6  SetDirectory[folder];
7  files=FileNames[];
8
9  For[i=1, i<= Length[files],++i,
10 l=StringSplit[files[[i]],{"_","."}];
11 iter=FromDigits[l[[2]]];
12 If[iter==0||iter==1000||iter==10000||iter==100000,
13 data=Import[files[[i]],"Table"];
14 res="rel_"<>Dvs[[id]]<>"_"<>ToString[seed+123]<>"_"<>ToString[iter *0.001];
15 res=StringReplace[res,"."->"-"];
16
17 pic=ListDensityPlot[data,
18 Frame->{True,True,False,False},
19 FrameLabel->{"x","y"},FrameStyle->18,RotateLabel->False,
20 ColorFunction->GrayLevel,(*ColorFunctionScaling\[Rule]False,*)
```

```
21 InterpolationOrder->5,
22 PlotLegends->Placed[BarLegend[Automatic,LegendMargins->{{0,0},{0,0}},LegendLabel->"u(x,y,t)",
       LabelStyle->{FontSize->18},LegendMarkerSize->350],Right]
23 ];
24 Export["D:\\projects\\compbio3\\pics\\"<>res<>".png",pic,ImageResolution->150];
25 ];
26 ];
27 ];
28 ];
```