

Брюсселятор

Содержание

1 Нераспределенный случай	1
1.1 Исследование на устойчивость нераспределенного случая	1
1.2 Предварительный анализ нераспределенного случая в Wolfram Mathematica	2
1.3 Анализ нераспределенного случая на С	3
2 Распределенный случай	5
2.1 Явная реализация	5
2.2 Подготовительные расчеты	7
2.3 Метод Кранка-Николсона	9
2.4 Граничные условия	11
2.5 Исследование устойчивости разностной схемы	12
2.5.1 Признак максимума	12
2.5.2 Энергетический критерий устойчивости	12
2.6 Исследование на устойчивость распределенного случая	13
2.7 Реализация на С	14
2.8 Анализ распределенного случая на Wolfram	18

1 Нераспределенный случай

1.1 Исследование на устойчивость нераспределенного случая

$$\begin{cases} \dot{u} = A + u^2v - (B + 1)u + D_u\Delta u, \\ \dot{v} = -u^2v + Bu + D_v\Delta v. \end{cases} \quad (1)$$

Линеаризуем эту систему:

$$\begin{cases} A + u^2v - (B + 1)u = 0, \\ u(B - uv) = 0. \end{cases} \quad (2)$$

Предположим, что коэффициенты А и В больше нуля, тогда решение этой системы:

$$\begin{cases} \bar{u} = A, \\ \bar{v} = \frac{B}{A}. \end{cases} \quad (3)$$

Линеаризуем систему вблизи этой особой точки:

$$\xi = u - \bar{u}, \quad \eta = v - \bar{v}. \quad (4)$$

$$\begin{cases} \dot{\xi} = (B - 1)\xi + A^2\eta, \\ \dot{\eta} = -B\xi - A^2\eta. \end{cases} \quad (5)$$

Найдем собственные значения матрицы коэффициентов:

$$\begin{vmatrix} B - 1 - \lambda & A^2 \\ -B & -A^2 - \lambda \end{vmatrix} = 0. \quad (6)$$

$$\lambda^2 + (A^2 + 1 - B)\lambda + A^2 = 0. \quad (7)$$

Решения этого уравнения:

$$\lambda_{1,2} = -\frac{1}{2}(A^2 + 1 - B) \pm \frac{1}{2}\sqrt{(A^2 + 1 - B)^2 - 4A^2}. \quad (8)$$

Введем обозначения: $\delta = (A^2 + 1 - B)$ и $\sigma = A^2$, тогда уравнение перепишется

$$\lambda_{1,2} = -\frac{1}{2}\delta \pm \frac{1}{2}\sqrt{\delta^2 - 4\sigma}. \quad (9)$$

Проанализируем возможные случаи:

- $\delta = 0$, тогда $\lambda_{1,2}$ чисто мнимые и решение является центром
- $\delta^2 < 4\sigma$, тогда корни мнимые и устойчивость зависит от знака δ : при $\delta > 0$ устойчивый фокус, при $\delta < 0$ неустойчивый фокус
- $\delta^2 > 4\sigma$, значит, корни действительные и простой анализ показывает, что знак $\lambda_{1,2}$ также зависит от знака δ : при $\delta > 0$ устойчивый узел, при $\delta < 0$ неустойчивый узел

$\delta = 0$ точка, в которой происходит смена характера поведения решения с устойчивости на неустойчивость, происходит бифуркация. Если $Re(\lambda_{1,2}) = 0$, а $Im(\lambda_{1,2}) \neq 0$, то происходит бифуркация Андронова-Хопфа или бифуркация рождения (исчезновения) предельного цикла. Из устойчивого фокуса при изменении параметров A, B может родиться предельный цикл.

1.2 Предварительный анализ нераспределенного случая в Wolfram Mathematica

Прежде чем программировать задачу на C или Python, проанализируем ее, используя Wolfram Mathematica.

```
Export[FileNameJoin@{NotebookDirectory[], "animation a=1 b=1.gif"}, 
Animate[
A = 1; B = 1;
sol = NDSolve[{x'[t] == A + x[t]^2 y[t] - (B + 1) x[t], 
y'[t] == x[t]^2 y[t] + B x[t], x[0] == 1, y[0] == 2}, {x[t], 
y[t]}, {t, 0, T}];
ParametricPlot[{x[t], y[t]} /. sol, {t, 0, T}], {T, 1, 100}
], "GIF"]
```

В коде можно менять параметры A и B, а также смотреть на картинку с течением времени.

Прикрепим также зависимость $x(t)$ и $y(t)$:

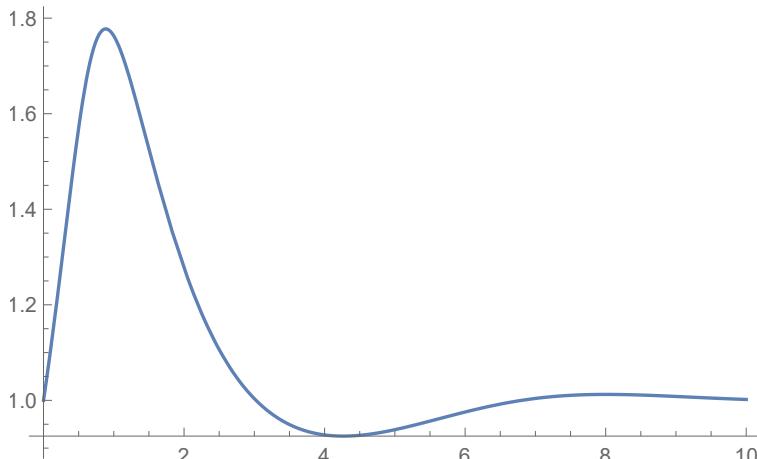


Рис. 1: Зависимость $x(t)$

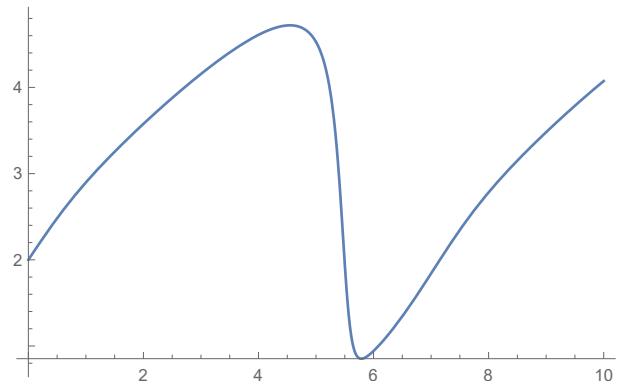


Рис. 2: Зависимость $y(t)$

Приведем характерные параметрические графики для случая $A=1, B=1$ и $A=1, B=3$. Как видно, в первом случае система устойчива, а во втором неустойчива. Это подтверждает пункт 1 отчета.

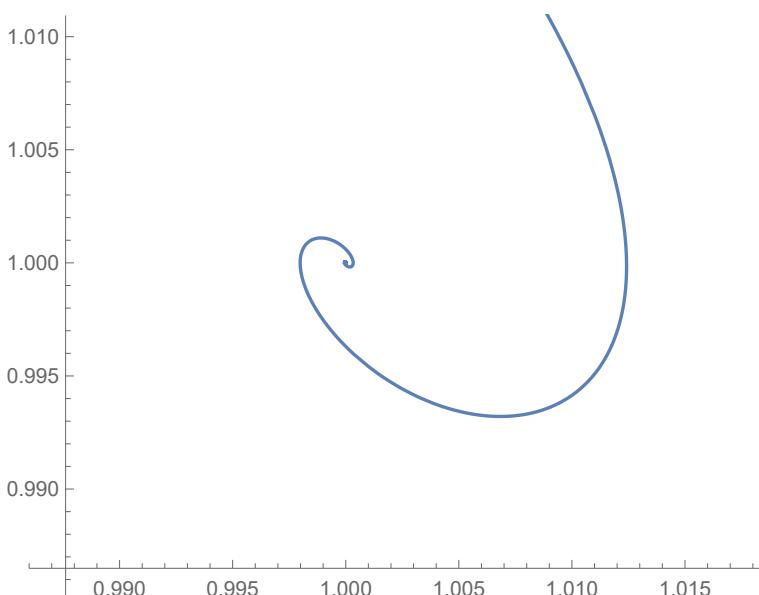


Рис. 3: A=1, B=1

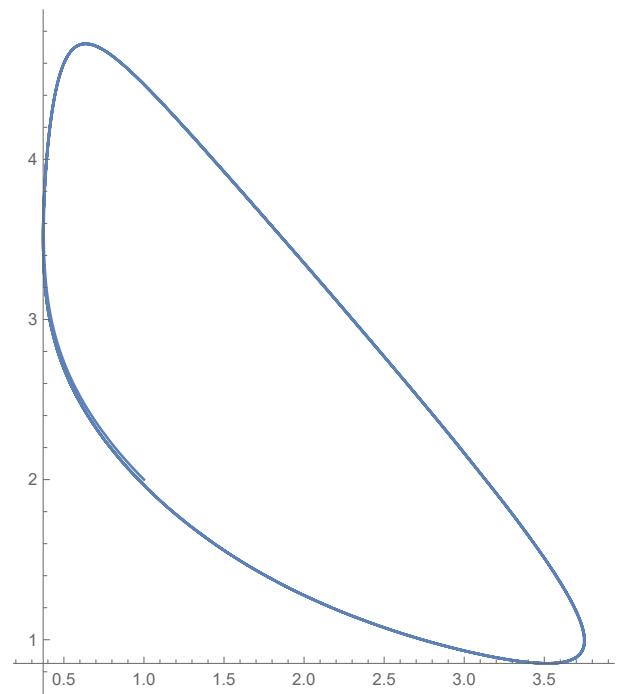


Рис. 4: A=1, B=3

1.3 Анализ нераспределенного случая на С

Будем использовать метод Рунге-Кутты.

Приведем исполняемый код:

```
#include <stdio.h>
#include<math.h>
#include<stdlib.h>

const float A = 1;
const float B = 3;

float F1(float u, float v)
{
    return (A + u*u*v * (B+1)*u);
}

float F2(float u, float v)
{
    return ( u*u*v + B*u);
}

void runge_kutt(float h, float* u, float* v, int N)
{
FILE* fp ;
fp = fopen("brus", "w") ;

for(int i = 0; i < N; i++)
{
float k11 = 0, k12 = 0, k13 = 0, k14 = 0;
float k21 = 0, k22 = 0, k23 = 0, k24 = 0;
k11 = h*F1(u[i], v[i]);
k12 = h*F2(u[i], v[i]);
k21 = h*F1(u[i] + k11/2, v[i] + k12/2);
k22 = h*F2(u[i] + k11/2, v[i] + k12/2);
k13 = h*F1(u[i] + k21/2, v[i] + k22/2);
k14 = h*F2(u[i] + k21/2, v[i] + k22/2);
k23 = h*F1(u[i] + k13, v[i] + k14);
k24 = h*F2(u[i] + k13, v[i] + k14);
u[i+1] = u[i] + (k11 + 2*k21 + 2*k31 + k41)/6;
v[i+1] = v[i] + (k12 + 2*k22 + 2*k32 + k42)/6;
}
}
```

```

k21 = h*F2(u[ i ] , v[ i ]);  

k12 = h*F1(u[ i ] + k11/2, v[ i ] + k21/2);  

k22 = h*F2(u[ i ] + k11/2, v[ i ] + k21/2);  

k13 = h*F1(u[ i ] + k12/2, v[ i ] + k22/2);  

k23 = h*F2(u[ i ] + k12/2, v[ i ] + k22/2);  

k14 = h*F1(u[ i ] + k13/2, v[ i ] + k23/2);  

k24 = h*F2(u[ i ] + k13/2, v[ i ] + k23/2);  

u[ i+1] = u[ i ] + (k11 + 2*k12 + 2*k13 + k14)/6;  

v[ i+1] = v[ i ] + (k21 + 2*k22 + 2*k23 + k24)/6;  

fprintf(fp , "%f , %f , %f \n" , h, u[ i ] , v[ i ] );  

}  

fclose (fp );  

}  
  

int main()  

{  

int n = 100000;  

float u[n];  

float v[n];  

float t[n];  
  

u[0] = 1;  

v[0] = 2;  

t[0] = 0;  

t[100] = 20;  

float tau = 0.001;  

int N = (t[100] - t[0])/tau;  

runge_kutt(tau , &u[0] , &v[0] , N);  

return 0;  

}

```

Построим три графика: $u(t)$, $v(t)$ и параметрический график u, v для двух случаев:

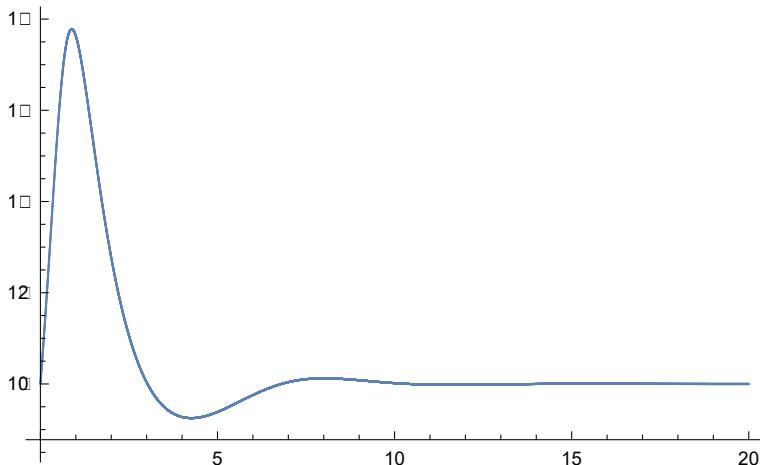


Рис. 5: $u(t)$ A=1, B=1

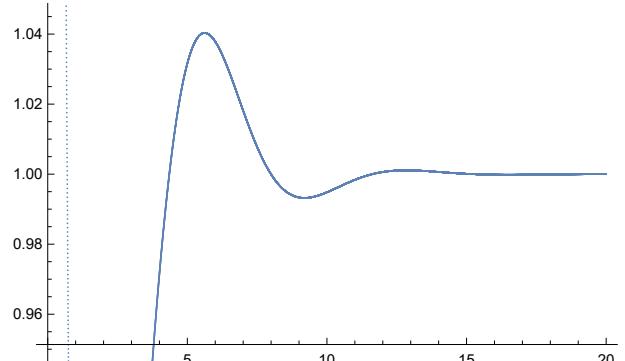


Рис. 6: $v(t)$ A=1, B=1

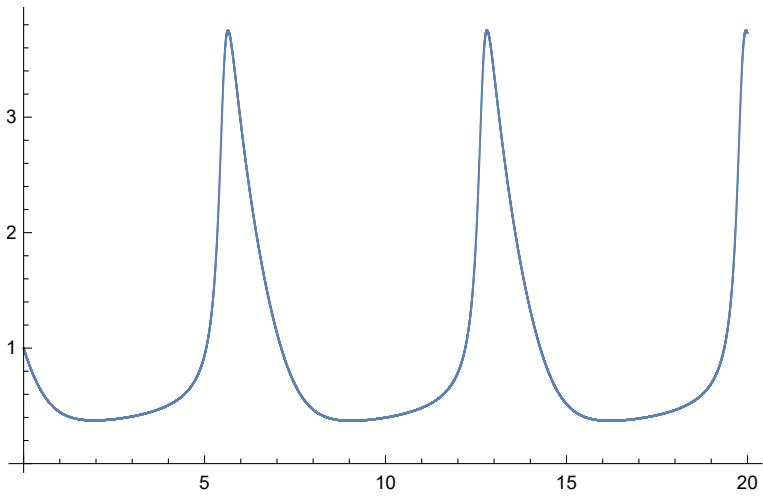


Рис. 7: $u(t)$ $A=1$, $B=3$

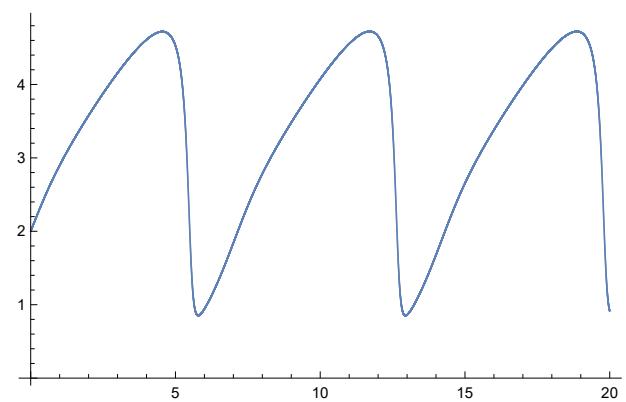


Рис. 8: $v(t)$ $A=1$, $B=3$

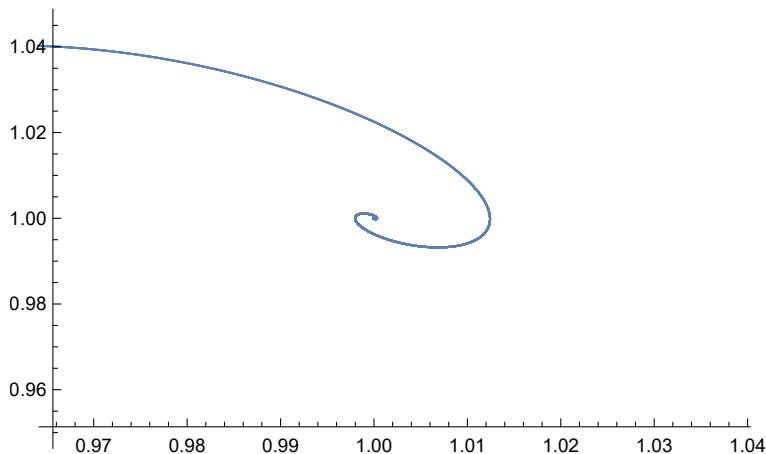


Рис. 9: $A=1$, $B=1$

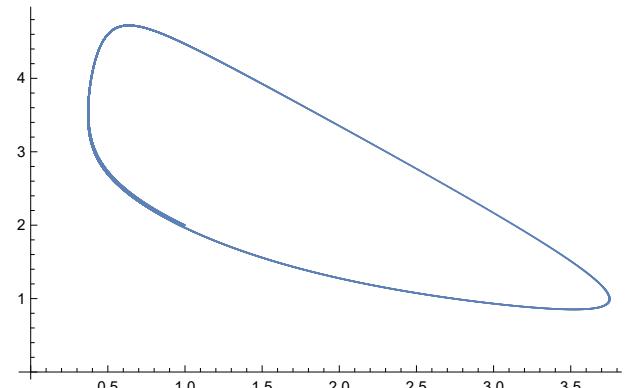


Рис. 10: $A=1$, $B=3$

Результаты совпадают с полученными в пункте 2.

2 Распределенный случай

2.1 Явная реализация

```
#include <stdio.h>
#include <stdlib.h>

const float A = 1;
const float B = 1;
const float Du = 1;
const float Dv = 100;

int main() {

    float a = 0;
    float b = 1;
    float T = 1;
    float t0 = 0;
    float h = 0.01;
```

```

float tau = 0.01;
int Nx = (b - a) / h;
int Nt = (T - t0) / tau;

float ru = Du*tau/2/h/h;
float rv = Dv*tau/2/h/h;

float fu[Nt+1][Nx+1];
float fv[Nt+1][Nx+1];

for(int j = 0; j < Nt; j++){
    for(int i = 0; i < Nx; i++){
        fu[j][i] = 0.;
        fv[j][i] = 0.;
    }
}

double noiseu = 0.005*(rand()%100);
double noisev = 0.005*(rand()%100);
for(int i = 0; i <= Nx; i++){
    fv[0][i] = B/A + noisev;
    fu[0][i] = A + noiseu;
}

for(int j = 1; j <= Nt; j++){
for(int i = 1; i < Nx; i++){
    fu[j][i] = fu[j-1][i] + tau*(A+fu[j-1][i]*fu[j-1][i]*fv[j-1][i] - (B+1)*fu[j-1][i] + Du/h/h*(fu[j-1][i+1]+fu[j-1][i-1] - 2*fu[j-1][i]));
    fv[j][i] = fv[j-1][i] + tau*( fu[j-1][i]*fu[j-1][i]*fv[j-1][i] + B*fu[j-1][i] + Dv/h/h*(fv[j-1][i+1]+fv[j-1][i-1] - 2*fv[j-1][i]));
}
    fu[j][0] = fu[j][1];
    fv[j][0] = fv[j][1];
    fv[j][Nx] = fv[j][Nx-1];
    fu[j][Nx] = fu[j][Nx-1];
}

FILE* fp;
fp = fopen("brus.dat", "w");
for(int j = 0; j <= Nt; j++){
    for(int i = 0; i <= Nx; i++){
        fprintf(fp, "%f %f %f %f\n", j*tau, i*h, fu[j][i], fv[j][i]);
    }
}
fclose(fp);
return 0;
}

```

Приведем результат выполнения программы:

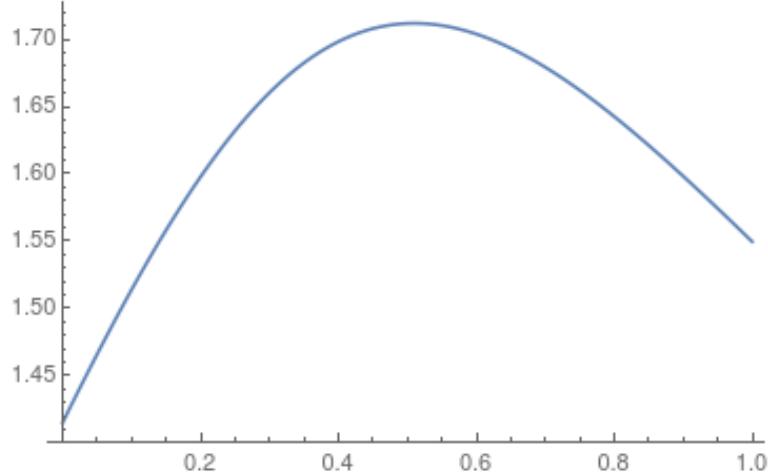


Рис. 11: Явная схема $u(t)$, $A=1$, $B=1$, $D_u = 1$, $D_v = 100$

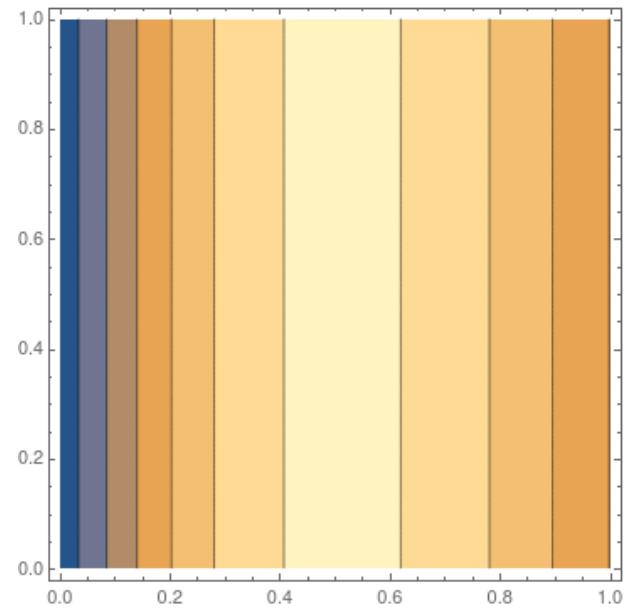


Рис. 12: Явная схема $u(t,x)$ $A=1$, $B=1$, $D_u = 1$, $D_v = 100$

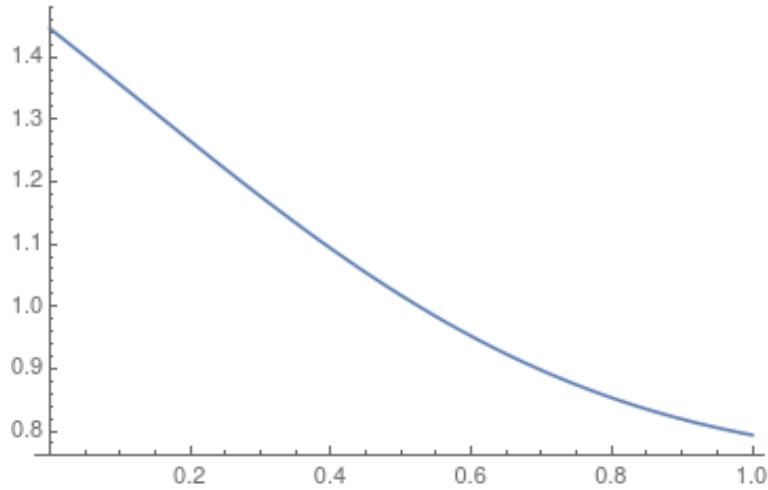


Рис. 13: Явная схема $v(t)$, $A=1$, $B=1$, $D_u = 1$, $D_v = 100$

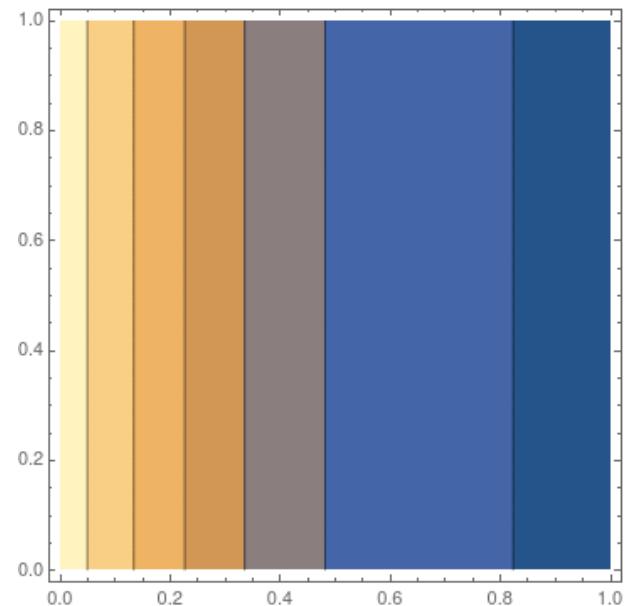


Рис. 14: Явная схема $v(t,x)$ $A=1$, $B=1$, $D_u = 1$, $D_v = 100$

2.2 Подготовительные расчеты

$$\begin{cases} \dot{u} = A + u^2v - (B+1)u + D_u\Delta u, \\ \dot{v} = -u^2v + Bu + D_v\Delta v. \end{cases} \quad (10)$$

Граничные условия:

$$\frac{\partial u}{\partial x}(0, t) = \frac{\partial u}{\partial x}(l, t) = \frac{\partial v}{\partial x}(0, t) = \frac{\partial v}{\partial x}(l, t) = 0. \quad (11)$$

Начальные условия:

$$u(x, 0) = A, \quad v(x, 0) = \frac{B}{A}. \quad (12)$$

$$u_1^{(k+1)} = u_0^{(k+1)} + \frac{\partial u}{\partial x} \Big|_0^{(k+1)} h + \frac{\partial^2 u}{\partial x^2} \Big|_0^{(k+1)} \frac{h^2}{2} \quad (13)$$

Выразим из первого уравнения брюсселятора 10 вторую производную по координате и подставим в предыдущее уравнение 13 (k - временная координата, j - пространственная):

$$\frac{\partial^2 u}{\partial x^2} = \frac{1}{D_u} \left(\frac{\partial u}{\partial t} - A - u^2 v + (B+1)u \right) \quad (14)$$

$$u_1^{(k+1)} = u_0^{(k+1)} + \frac{\partial u}{\partial x} \Big|_0^{(k+1)} h + \frac{h^2}{2D_u} \left(\frac{\partial u}{\partial t} \Big|_0^{(k+1)} - A - u^2 v \Big|_0^{(k+1)} + (B+1)u_0^{(k+1)} \right) \quad (15)$$

Выразим из 15 первую производную и приравняем ее нулю, так как граничным условием является нулевой поток.

$$\frac{\partial u}{\partial x} = \frac{1}{h} (u_1^{(k+1)} - u_0^{(k+1)}) - \frac{h^2}{2D_u} \left(\frac{\partial u}{\partial t} \Big|_0^{(k+1)} - A - u^2 v \Big|_0^{(k+1)} + (B+1)u_0^{(k+1)} \right) = 0. \quad (16)$$

Разложим $(u^2 v) \Big|_0^{(k+1)}$ по формуле Тейлора:

$$(u^2 v) \Big|_0^{(k+1)} = (u^2) \Big|_0^{(k+1)} v_0^{(k+1)} + 2u_0^{(k)} v_0^{(k)} u_0^{(k+1)} - 2(u^2) \Big|_0^{(k)} v_0^{(k)}. \quad (17)$$

При этом

$$\frac{\partial u}{\partial t} \Big|_0^{(k+1)} = \frac{u_0^{(k+1)} - u_0^{(k)}}{\tau} \quad (18)$$

Подставим 17 и 18 в 16 и сгруппируем члены при $u_1^{(k+1)}$ и $u_0^{(k+1)}$.

$$b_0 u_0^{(k+1)} + c_0 u_1^{(k+1)} = d_0, \quad j = 0, \quad (19)$$

где

$$a_0 = 0, \quad b_0 = -\frac{1}{h} - \frac{h}{2D_u} ((B+1) + \frac{1}{\tau} - 2u_0^{(k)} v_0^{(k)}), \quad c_0 = \frac{1}{h}, \quad d_0 = \frac{h^2}{2D_u} \left(-\frac{1}{\tau} u_0^{(k)} - A - (u^2) \Big|_0^{(k)} v_0^{(k+1)} + 2(u^2) \Big|_0^{(k)} v_0^{(k)} \right). \quad (20)$$

Аналогично потсупим для второй границы.

$$u_{N-1}^{(k+1)} = u_N^{(k+1)} + \frac{\partial u}{\partial x} \Big|_N^{(k+1)} h + \frac{\partial^2 u}{\partial x^2} \Big|_N^{(k+1)} \frac{h^2}{2} \quad (21)$$

$$u_{N-1}^{(k+1)} = u_N^{(k+1)} + \frac{\partial u}{\partial x} \Big|_N^{(k+1)} h + \frac{h^2}{2D_u} \left(\frac{\partial u}{\partial t} \Big|_N^{(k+1)} - A - u^2 v \Big|_N^{(k+1)} + (B+1)u_N^{(k+1)} \right) \quad (22)$$

$$\frac{\partial u}{\partial x} = \frac{1}{h} (u_{N-1}^{(k+1)} - u_N^{(k+1)}) - \frac{h^2}{2D_u} \left(\frac{\partial u}{\partial t} \Big|_N^{(k+1)} - A - u^2 v \Big|_N^{(k+1)} + (B+1)u_N^{(k+1)} \right) = 0. \quad (23)$$

$$\frac{\partial u}{\partial t} \Big|_N^{(k+1)} = \frac{u_N^{(k+1)} - u_N^{(k)}}{\tau} \quad (24)$$

$$a_N u_{N-1}^{(k+1)} + b_N u_N^{(k+1)} = d_N, \quad j = N, \quad (25)$$

где

$$a_N = \frac{1}{h}, \quad b_N = -\frac{1}{h} - \frac{h}{2D_u} ((B+1) + \frac{1}{\tau} - 2u_N^{(k)} v_N^{(k)}), \quad c_N = 0, \quad d_N = \frac{h}{2D_u} \left(-\frac{1}{\tau} u_N^{(k)} - A - (u^2) \Big|_N^{(k)} v_N^{(k+1)} + 2(u^2) \Big|_N^{(k)} v_N^{(k)} \right). \quad (26)$$

Припишем к граничным конечно-разностным уравнениям 10 в виде:

$$\frac{u_j^{(k+1)} - u_j^{(k)}}{\tau} = \frac{D_u}{h^2} (u_{j+1}^{(k+1)} - 2u_j^{(k+1)} + u_{j-1}^{(k+1)}) + A + (u^2) \Big|_j^{(k)} v_j^{(k+1)} + 2u_j^{(k)} v_j^{(k)} u_j^{(k+1)} - 2(u^2) \Big|_j^{(k)} v_j^{(k)} - (B+1)u_j^{(k+1)}, \quad j = \overline{1, N-1}. \quad (27)$$

$$a_j u_{j-1}^{(k+1)} + b_j u_j^{(k+1)} + c_j u_{j+1}^{(k+1)} = d_j, j = \overline{1, N-1}, \quad (28)$$

где

$$a_j = \frac{D_u}{h^2}, \quad b_j = -\frac{1}{\tau} - \frac{2D_u}{h^2} - (B+1) + 2u_j^{(k)}v_j^{(k)}, \quad c_j = \frac{D_u}{h^2}, \quad d_j = -(\frac{u_j^{(k)}}{\tau} + A + (u^2)_j^{(k)}v_j^{(k+1)} - 2(u^2)_j^{(k)}v_j^{(k)}). \quad (29)$$

По изложенному алгоритму поступим и со вторым уравнения брюсселятора.

$$b_0 v_0^{(k+1)} + c_0 v_1^{(k+1)} = d_0, \quad j = 0, \quad (30)$$

где

$$a_0 = 0, \quad b_0 = -\frac{1}{h} - \frac{h^2}{2D_v}(\frac{1}{\tau} + (u^2)_0^{(k)}), \quad c_0 = \frac{1}{h}, \quad d_0 = \frac{h^2}{2D_u}(-\frac{1}{\tau}v_0^{(k)} + 2u_0^{(k)}v_0^{(k)}u_0^{(k+1)} - 2(u^2)_0^{(k)}v_0^{(k)} - Bu_0^{(k+1)}),. \quad (31)$$

$$a_N v_{N-1}^{(k+1)} + b_N v_N^{(k+1)} = d_N, \quad j = N, \quad (32)$$

где

$$a_0 = \frac{1}{h}, \quad b_0 = -\frac{1}{h} - \frac{h}{2D_v}(\frac{1}{\tau} + (u^2)_N^{(k)}), \quad c_N = 0, \quad d_N = \frac{h}{2D_v}(-\frac{1}{\tau}v_N^{(k)} + 2u_N^{(k)}v_N^{(k)}u_N^{(k+1)} - 2(u^2)_N^{(k)}v_N^{(k)} - Bu_N^{(k+1)}). \quad (33)$$

$$\frac{v_j^{(k+1)} - v_j^{(k)}}{\tau} = \frac{D_v}{h^2}(v_{j+1}^{(k+1)} - 2v_j^{(k+1)} + v_{j-1}^{(k+1)}) - (u^2)_j^{(k)}v_j^{(k+1)} - 2u_j^{(k)}v_j^{(k)}u_j^{(k+1)} + 2(u^2)_j^{(k)}v_j^{(k)} + Bu_j^{(k+1)}, j = \overline{1, N-1}. \quad (34)$$

$$a_j v_{j-1}^{(k+1)} + b_j v_j^{(k+1)} + c_j v_{j+1}^{(k+1)} = d_j, j = \overline{1, N-1}, \quad (35)$$

где

$$a_j = \frac{D_v}{h^2}, \quad b_j = -\frac{1}{\tau} - \frac{2D_u}{h^2} + (u^2)_j^{(k)}, \quad c_j = \frac{D_v}{h^2}, \quad d_j = -(\frac{v_j^{(k)}}{\tau} + 2(u^2)_j^{(k)}v_j^{(k)} + 2u_j^{(k)}v_j^{(k)}u_j^{(k+1)} + Bu_j^{(k+1)}). \quad (36)$$

Для каждого из уравнений получаем СЛАУ с трехдиагональной матрицей, решаемой методом прогонки:

$$A_j = -\frac{c_j}{b_j + a_j A_{j-1}}, \quad B_j = \frac{d_j - a_j B_{j-1}}{b_j + a_j A_{j-1}}, \quad A_0 = -\frac{c_0}{b_0}, \quad B_0 = \frac{d_0}{b_0}, \quad A_N = 0, j = \overline{0, N}; \quad (37)$$

$$u_j^{(k+1)} = A_j u_{j+1}^{(k+1)} + B_j, \quad u_N^{(k+1)} = B_N, j = N, N-1, \dots, 0. \quad (38)$$

2.3 Метод Кранка-Николсона

$$\begin{cases} \dot{u} = A + u^2v - (B+1)u + D_u \Delta u, \\ \dot{v} = -u^2v + Bu + D_v \Delta v. \end{cases} \quad (39)$$

Границные условия:

$$\frac{\partial u}{\partial x}(0, t) = \frac{\partial u}{\partial x}(l, t) = \frac{\partial v}{\partial x}(0, t) = \frac{\partial v}{\partial x}(l, t) = 0. \quad (40)$$

Начальные условия:

$$u(x, 0) = A, \quad v(x, 0) = \frac{B}{A}. \quad (41)$$

Введем: p - пространственная координата, q - времененная.

Приблизим вторую производную в уравнениях разностной схемой Кранка-Николсона, перегруппируем члены и введем обозначения $r_u = \frac{D_u \tau}{2h^2}$, $r_v = \frac{D_v \tau}{2h^2}$, $f_u = f_u(u_p^{(q)}, v_p^{(q)}) = A + (u_p^{(q)})^2 v_p^{(q)} - (B+1)u_p^{(q)}$, $f_v = f_v(u_p^{(q)}, v_p^{(q)}) = -(u_p^{(q)})^2 v_p^{(q)} - Bu_p^{(q)}$ (здесь τ - шаг временненой сетки, а h - пространственной):

$$u_p^{(q+1)} - u_p^{(q)} = \frac{D_u \tau}{2h^2} (u_{p+1}^{(q+1)} - 2u_p^{(q+1)} + u_{p-1}^{(q+1)}) + \frac{D_u \tau}{2h^2} (u_{p+1}^{(q)} - 2u_p^{(q)} + u_{p-1}^{(q)}) + \tau f_u = \\ r_u (u_{p+1}^{(q+1)} - 2u_p^{(q+1)} + u_{p-1}^{(q+1)}) + r_u (u_{p+1}^{(q)} - 2u_p^{(q)} + u_{p-1}^{(q)}) + \tau f_u. \quad (42)$$

$$u_p^{(q+1)} (1 + 2r_u) - r_u (u_{p+1}^{(q+1)} + u_{p-1}^{(q+1)}) = u_p^{(q)} + r_u (u_{p+1}^{(q)} - 2u_p^{(q)} + u_{p-1}^{(q)}) + \tau f_u. \quad (43)$$

$$\begin{cases} (1 + 2r_u)u_p^{q+1} - r_u (u_{p+1}^{q+1} + u_{p-1}^{q+1}) = u_p^{(q)} + r_u (u_{p+1}^{(q)} - 2u_p^{(q)} + u_{p-1}^{(q)}) + \tau f_u, \\ (1 + 2r_v)v_p^{q+1} - r_v (v_{p+1}^{q+1} + v_{p-1}^{q+1}) = v_p^{(q)} + r_v (v_{p+1}^{(q)} - 2v_p^{(q)} + v_{p-1}^{(q)}) + \tau f_v. \end{cases} \quad (44)$$

В левой части стоят три неизвестных величины, а в правой три известных. Получили тридиагональную матрицу. Учтем граничные условия:

$$u_0^{(q+1)} - u_0^{(q)} = r_u (u_1^{q+1} - 2u_0^{(q+1)} + u_{-1}^{(q+1)}) + r_u (u_1^{(q)} - 2u_0^{(q)} + u_{-1}^{(q)}) + \tau f_u. \quad (45)$$

Так как граничным условием является нулевой поток, $u_{-1}^{(q)} = u_1^{(q)}$ и уравнение перепишется в виде

$$u_0^{(q+1)} - u_0^{(q)} = 2r_u (u_1^{q+1} - u_0^{(q+1)}) + 2r_u (u_1^{(q)} - u_0^{(q)}) + \tau f_u. \quad (46)$$

$$(1 + 2r_u)u_0^{q+1} - 2r_u u_1^{q+1} = u_0^{(q)} + 2r_u (u_1^{(q)} - u_0^{(q)}) + \tau f_u. \quad (47)$$

Аналогично для второй границы и второй переменной.

Запишем в виде матрицы:

$$\begin{pmatrix} (1 + 2r_u) & -2r_u & 0 & 0 & 0 & \dots & 0 \\ -r_u & (1 + 2r_u) & -r_u & 0 & 0 & \dots & 0 \\ 0 & -r_u & (1 + 2r_u) & -r_u & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & -r_u & (1 + 2r_u) & -r_u \\ 0 & 0 & 0 & 0 & 0 & -2r_u & (1 + 2r_u) \end{pmatrix} \cdot \begin{pmatrix} u_0^{q+1} \\ u_1^{q+1} \\ u_2^{q+1} \\ \vdots \\ u_{n-1}^{q+1} \\ u_n^{q+1} \end{pmatrix} = \quad (48)$$

$$\begin{pmatrix} u_0^{(q)} + 2r_u (u_1^{(q)} - u_0^{(q)}) + \tau f_u \\ u_1^{(q)} + r_u (u_2^{(q)} - 2u_1^{(q)} + u_0^{(q)}) + \tau f_u \\ u_2^{(q)} + r_u (u_3^{(q)} - 2u_2^{(q)} + u_1^{(q)}) + \tau f_u \\ \vdots \\ u_{n-1}^{(q)} + r_u (u_n^{(q)} - 2u_{n-1}^{(q)} + u_{n-2}^{(q)}) + \tau f_u \\ u_n^{(q)} + 2r_u (u_{n-1}^{(q)} - u_n^{(q)}) + \tau f_u \end{pmatrix} \quad (49)$$

Аналогичные выражения можно получить и для v .

$$\begin{pmatrix} (1 + 2r_v) & -2r_v & 0 & 0 & 0 & \dots & 0 \\ -r_v & (1 + 2r_v) & -r_v & 0 & 0 & \dots & 0 \\ 0 & -r_v & (1 + 2r_v) & -r_v & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & -r_v & (1 + 2r_v) & -r_v \\ 0 & 0 & 0 & 0 & 0 & -2r_v & (1 + 2r_v) \end{pmatrix} \cdot \begin{pmatrix} v_0^{q+1} \\ v_1^{q+1} \\ v_2^{q+1} \\ \vdots \\ v_{n-1}^{q+1} \\ v_n^{q+1} \end{pmatrix} = \quad (50)$$

$$\begin{pmatrix} v_0^{(q)} + 2r_v(v_1^{(q)} - v_0^{(q)}) + \tau f_v \\ v_1^{(q)} + r_v(v_2^{(q)} - 2v_1^{(1)} + v_0^{(q)}) + \tau f_v \\ v_2^{(q)} + r_v(v_3^{(q)} - 2v_2^{(1)} + v_1^{(q)}) + \tau f_v \\ \vdots \\ v_{n-1}^{(q)} + r_v(v_n^{(q)} - 2v_{n-1}^{(1)} + v_{n-2}^{(q)}) + \tau f_v \\ v_n^{(q)} + 2r_v(v_{n-1}^{(q)} - v_n^{(q)}) + \tau f_v \end{pmatrix} \quad (51)$$

Так как обе матрицы имеют диагональное преобладание, то их можно решать методом прогонки.

2.4 Границные условия

Первый способ записать граничные условия - воспользоваться методом из предыдущего пункта.

$$\dot{u} = A + u^2 v - (B + 1)u + D_u \Delta u. \quad (52)$$

Так как на границе нулевой поток, то

$$u_1^{(k)} = u_0^{(k)} + \frac{\partial u}{\partial x} \Big|_0^{(k)} h + \frac{\partial^2 u}{\partial x^2} \Big|_0^{(k)} \frac{h^2}{2} = u_0^{(k)} + \frac{\partial^2 u}{\partial x^2} \Big|_0^{(k)} \frac{h^2}{2}. \quad (53)$$

Подставляем вторую производную из 53 в 52.

$$\dot{u} = A + u^2 v - (B + 1)u + \frac{2Du}{h^2}(u_1^{(k)} - u_0^{(k)}). \quad (54)$$

Производная по времени

$$u_0^{(k+1)} = u_0^{(k)} + \tau \dot{u} \Big|_0^k. \quad (55)$$

Значит,

$$u_0^{(k+1)} = u_0^{(k)} + \tau(A + (u_0^{(k)})^2 v_0^{(k)} - (B + 1)u_0^{(k)} + \frac{2Du}{h^2}(u_1^{(k)} - u_0^{(k)})). \quad (56)$$

Аналогично:

$$u_{N-1}^{(k+1)} = u_{N-1}^{(k)} + \tau(A + (u_{N-1}^{(k)})^2 v_{N-1}^{(k)} - (B + 1)u_{N-1}^{(k)} + \frac{2Du}{h^2}(u_{N-2}^{(k)} - u_{N-1}^{(k)})). \quad (57)$$

$$v_0^{(k+1)} = v_0^{(k)} + \tau(-(u_0^{(k)})^2 v_0^{(k)} + Bu_0^{(k)} + \frac{2Du}{h^2}(v_1^{(k)} - v_0^{(k)})). \quad (58)$$

$$v_{N-1}^{(k+1)} = v_{N-1}^{(k)} + \tau(-(u_{N-1}^{(k)})^2 v_{N-1}^{(k)} + Bu_{N-1}^{(k)} + \frac{2Du}{h^2}(v_{N-2}^{(k)} - v_{N-1}^{(k)})). \quad (59)$$

Второй более простой способ записать граничные условия:

$$\frac{\partial u}{\partial x} \Big|_{x=0}^q = 0 = \frac{u_1^q - u_0^q}{h}. \quad (60)$$

Отсюда

$$u_1^q = u_0^q, \quad u_n^q = u_{n-1}^q, \quad v_1^q = v_0^q, \quad v_n^q = v_{n-1}^q. \quad (61)$$

Третий способ записи граничных условий - метод фиктивного узла - реализован в предыдущем пункте.

2.5 Исследование устойчивости разностной схемы

2.5.1 Признак максимума

Рассмотрим первое уравнение брюсселятора:

$$\frac{u_p^{(q+1)} - u_p^{(q)}}{\tau} = D(u_p^{(q+1)} + u_p^{(q)}) + f_u. \quad (62)$$

Здесь D - оператор второй производной, равный $Du_p = \frac{u_{p+1}-2u_p+u_{p-1}}{2h^2}$. Так же это уравнение можно записать

$$\sum_{Q \in O(P)} M_h(P, Q)u(Q) = f_u(P). \quad (63)$$

Мы рассматриваем сетку с шагом h окрестности точки P . Приведем к каноническому виду:

$$A(p)u(P) = \sum_{Q \in O(P) \setminus P} B(P, Q)u(Q) + f_u(P). \quad (64)$$

В качестве центрального узла шаблона P выберем точку $P(x_p, t_{q+1})$, тогда

$$O(P) \setminus P = \{Q_1(x_p, t_q), Q_2(x_{p-1}, t_q), Q_3(x_{p+1}, t_q), Q_4(x_{p-1}, t_{q+1}), Q_5(x_{p+1}, t_{q+1})\}. \quad (65)$$

Из предыдущих вычислений:

$$(1 + 2r_u)u_p^{q+1} = r_u(u_{p+1}^{q+1} + u_{p-1}^{q+1}) + r_u(u_{p+1}^{(q)} + u_{p-1}^{(q)}) + (1 - 2r_u)u_p^{(q)} + \tau f_u. \quad (66)$$

Тогда коэффициенты канонической формы разностного уравнения будут иметь вид:

$$A(P) = 1 + 2r_u, \quad B(P, Q_1) = (1 - 2r_u), \quad B(P, Q_2) = B(P, Q_3) = B(P, Q_4) = B(P, Q_5) = r_u. \quad (67)$$

Воспользуемся принципом максимума. Для устойчивости разностной схемы достаточно, чтобы $A(P) > 0$, $B(P, Q_m) \geq 0$ $\forall m$ и $D(P) = A(P) - \sum_{Q \in O(P) \setminus P} B(P, Q) \geq 0$. В нашем случае эти условия выполняются: $A(P) = 1 + 2r_u > 0$ всегда, $B(P, Q_1) = (1 - 2r_u) \geq 0$, если $r_u < \frac{1}{2}$, $B(P, Q_2) = B(P, Q_3) = B(P, Q_4) = B(P, Q_5) = r_u \geq 0$ всегда и $D(P) = 1 + 2r_u - (1 - 2r_u) - 4r_u = 0$ всегда. Аналогичные условия можно написать для второго уравнения брюсселятора. Однако это условие является достаточным, но не необходимым. Его отсутствие необязательно влечет к потери устойчивости.

2.5.2 Энергетический критерий устойчивости

Воспользуемся энергетическим критерием устойчивости. Рассмотрим первое уравнение брюсселятора:

$$\frac{u_p^{(q+1)} - u_p^{(q)}}{\tau} = \frac{D_u}{2} \Lambda_{xx}(u_p^{(q+1)} + u_p^{(q)}) + f_u. \quad (68)$$

Здесь Λ_{xx} - оператор второй производной, равный $\Lambda_{xx}u_p = \frac{u_{p+1}-2u_p+u_{p-1}}{h^2}$. Приведем к каноническому виду:

$$(E - \frac{D_u \tau}{2} \Lambda_{xx}) \frac{u_p^{(q+1)} - u_p^{(q)}}{\tau} - D_u \Lambda_{xx}u_p^{(q)} = f_u. \quad (69)$$

Необходимым и достаточным условием устойчивости будет $E - \frac{D_u \tau}{2} \Lambda_{xx} \geq -\frac{\tau D_u}{2} \Lambda_{xx}$ или $E \geq 0$, что верно при любых шагах по пространству и времени. Аналогично для второго уравнения брюсселятора. То есть наша схема Кранка-Николсона абсолютной устойчива.

2.6 Исследование на устойчивость распределенного случая

Рассмотрим более общий случай:

$$\begin{cases} \frac{\partial u}{\partial t} = P(u, v, x) + D_u \frac{\partial^2 u}{\partial x^2}, \\ \frac{\partial v}{\partial t} = Q(u, v, x) + D_v \frac{\partial^2 v}{\partial x^2}. \end{cases} \quad (70)$$

Введем ξ и η - малые отклонения от решений системы $P(u, v) = 0$, $Q(u, v) = 0$. Тогда уравнения перепишутся:

$$\begin{cases} \frac{\partial \xi}{\partial t} = a\xi + b\eta + D_u \frac{\partial^2 \xi}{\partial x^2}, \\ \frac{\partial \eta}{\partial t} = c\xi + d\eta + D_v \frac{\partial^2 \eta}{\partial x^2}, \end{cases} \quad (71)$$

где

$$a = \frac{\partial P}{\partial u}, \quad b = \frac{\partial P}{\partial v}, \quad c = \frac{\partial Q}{\partial u}, \quad d = \frac{\partial Q}{\partial v}. \quad (72)$$

Решение будем искать в виде $\xi(t, x) = \bar{A}e^{pt}e^{ikx}$, $\eta(t, x) = \bar{B}e^{pt}e^{ikx}$.

Подставим в систему и сократим на экспоненты, получим:

$$\begin{cases} \bar{A}p = a\bar{A} + b\bar{B} - D_u k^2 \bar{A}, \\ \bar{B}p = c\bar{A} + d\bar{B} - D_v k^2 \bar{B}. \end{cases} \quad (73)$$

\bar{A} и \bar{B} не равны нулю, если

$$(p - a + k^2 D_u)(p - d + k^2 D_v) - bc = 0. \quad (74)$$

Решения этого уравнения:

$$p_{1,2} = \frac{a + d - (D_u + D_v)k^2 \pm \sqrt{(a - d - k^2(D_u - D_v))^2 + 4bc}}{2}. \quad (75)$$

Знак действительной части $p_{1,2}$ показывает, будет ли решением устойчивым.

В нашем случае

$$a = \frac{\partial P}{\partial u} = B - 1, \quad b = \frac{\partial P}{\partial v} = A^2, \quad c = \frac{\partial Q}{\partial u} = -B, \quad d = \frac{\partial Q}{\partial v} = -A^2. \quad (76)$$

и дисперсионное уравнение имеет вид:

$$(p - B - 1 + k^2 D_u)(p + A^2 + K^2 D_v) + BA^2 = 0. \quad (77)$$

$$p_{1,2} = \frac{B - 1 - A^2 - (D_u + D_v)k^2 \pm \sqrt{(B - 1 - A^2 - k^2(D_u - D_v))^2 - 4A^2 B}}{2}. \quad (78)$$

Построим график $p_{1,2}(k)$:

```

Assuming[{k \[Element] Reals, k > 0},
Block[{B = 3, A = 1, Du = 1, Dv = 100},
p1[k_] := 0.5*(B - 1 - A^2 - (Du + Dv) k^2 + Sqrt[(B - 1 - A^2 - (Du + Dv) k^2)^2 - 4A^2 B]);
p2[k_] := 0.5*(B - 1 - A^2 - (Du + Dv) k^2 - Sqrt[(B - 1 - A^2 - (Du + Dv) k^2)^2 - 4A^2 B]);
realp1[k_] := Re@p1[k];
realp2[k_] := Re@p2[k];
Show[Plot[{realp1[k], realp2[k]}, {k, 0, 2}, PlotRange > {{0, 2}, {2, 2}}]]
]

```

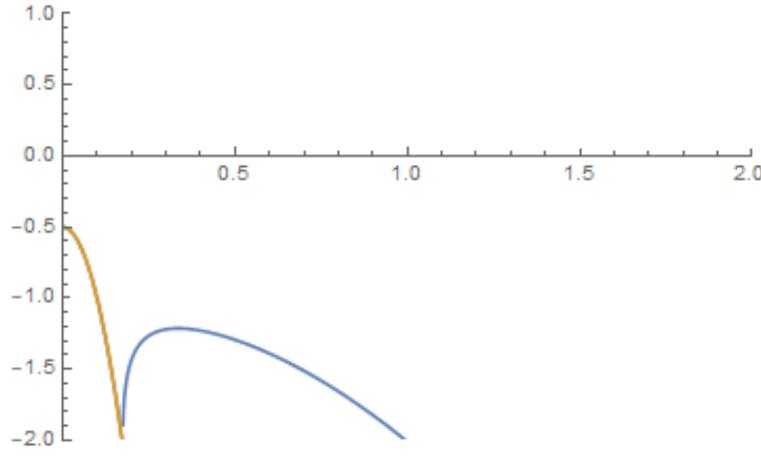


Рис. 15: Дисперсионная кривая $A=1$, $B=1$, $D_u = 1$, $D_v = 100$

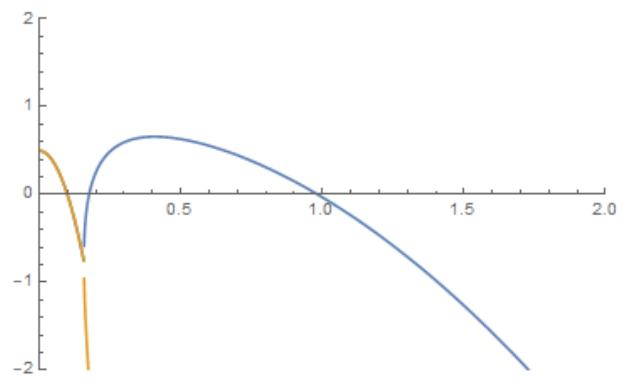


Рис. 16: Дисперсионная кривая $A=1$, $B=3$, $D_u = 1$, $D_v = 100$

В области, где $p_{1,2}$ имеют два действительных корня разного знака реализуется неустойчивость Тьюринга.

2.7 Реализация на С

Опишем алгоритм подбора параметров пространственно-временной сетки:

1. Зафиксируем параметры брюсселятора: коэффициенты A, B, D_u и D_v .
2. Строим график зависимости $Re(p_{1,2}(k))$ для данных параметров.
3. Находим k_{min} и k_{max} , соответствующие смене знака реальной части $p_{1,2}$ и, соответственно, наступлению зоны седловой неустойчивости.
4. $\lambda_{min} = \frac{2\pi}{k_{max}}$, $\lambda_{max} = \frac{2\pi}{k_{min}}$ - границы интервала длин волн, которые будут неустойчивы.
5. λ_{max} должна войти в пространственную область несколько раз, поэтому $b \approx 5\lambda_{max}$.
6. В волну должно укладываться несколько шагов сетки, тогда $h \approx \frac{\lambda_{min}}{10}$.
7. Всего шагов по пространству будет $\frac{b}{h} \approx 50 \frac{\lambda_{max}}{\lambda_{min}} \approx 100$.
8. Исходя из h подберем шаг временной сетки так, чтобы $\frac{D_v \tau}{h^2} < \frac{1}{2}$, то есть $\tau < \frac{h^2}{2D_v}$.

```
#include <stdio.h>
#include <stdlib.h>

#define mpi 3.1415926

void solve_tridiagonal(double * restrict const x, const int X, const double * restrict
    const a, const double * restrict const b, double * restrict const c) {
    /*
     solves Ax = v where A is a tridiagonal matrix consisting of vectors a, b, c
     x initially contains the input vector v, and returns the solution x. indexed
     from 0 to X - 1 inclusive
     X number of equations (length of vector x)
     a subdiagonal (means it is the diagonal below the main diagonal), indexed from 1
     to X - 1 inclusive
     b the main diagonal, indexed from 0 to X - 1 inclusive
     c superdiagonal (means it is the diagonal above the main diagonal), indexed from
     0 to X - 2 inclusive
    */
}
```

Note: contents of input vector c will be modified, making this a one time use function (scratch space can be allocated instead for this purpose to make it reusable)

*Note 2: We don't check for diagonal dominance, etc.; this is not guaranteed stable */*

```

c[0] = c[0] / b[0];
x[0] = x[0] / b[0];

/* loop from 1 to X - 1 inclusive, performing the forward sweep */
for (int ix = 1; ix < X; ix++) {
    const double m = 1.0f / (b[ix] - a[ix] * c[ix - 1]);
    c[ix] = c[ix] * m;
    x[ix] = (x[ix] - a[ix] * x[ix - 1]) * m;
}

/* loop from X - 2 to 0 inclusive (safely testing loop condition for an unsigned integer), to perform the back substitution */
for (int ix = X - 2; ix >= 0; ix--)
    x[ix] = c[ix] * x[ix + 1];
}

const double A = 1;
const double B = 1;
const double Du = 1;
const double Dv = 100;

int main() {

    double a = 0;
    double b = 10;
    double T = 1;
    double t0 = 0;
    double h = mpi/5;
    double tau = 0.0001;
    int Nx = (b - a) / h;
    int Nt = (T - t0) / tau;

    double ru = Du*tau/2/h/h;
    double rv = Dv*tau/2/h/h;

    double au[Nx+1]; double bu[Nx+1]; double cu[Nx+1];
    double av[Nx+1]; double bv[Nx+1]; double cv[Nx+1];
    au[0] = 0.; av[0] = 0.;
    au[Nx] = 2*ru;
    av[Nx] = 2*rv;

    for (int i=1; i<Nx; i++){
        av[i] = rv;
        au[i] = ru;
    }
    for (int i=0; i<=Nx; i++){
        bu[i] = 1+2*ru;
        bv[i] = 1+2*rv;
    }
}

```

```

cu[Nx] = 0.;
cv[Nx] = 0.;
cu[0] = 2*ru;
cv[0] = 2*rv;
for (int i=1; i<Nx; i++){
    cv[i] = rv;
    cu[i] = ru;
}

double fu[Nt+1][Nx+1];
double fv[Nt+1][Nx+1];

for (int j = 0; j <= Nt; j++){
    for (int i = 0; i <= Nx; i++){
        fu[j][i] = 0.;
        fv[j][i] = 0.;

    }
}
double noiseu = 0.005*(rand()%100);
double noisev = 0.005*(rand()%100);
for (int i = 0; i <= Nx; i++){
    fv[0][i] = B/A + noisev;
    fu[0][i] = A + noiseu;
}

double urightpart[Nx+1];
double vrightpart[Nx+1];

for (int j = 1; j <= Nt; j++){
    urightpart[0] = fu[j-1][0] + 2*ru*(fu[j-1][1] - fu[j-1][0]) + tau*(A + fu[j-1][0]*fu[j-1][0]*fv[j-1][0] - (B+1)*fu[j-1][0]);
    vrightpart[0] = fv[j-1][0] + 2*rv*(fv[j-1][1] - fv[j-1][0]) + tau*( fu[j-1][0]*fu[j-1][0]*fv[j-1][0] + B*fu[j-1][0]);
    urightpart[Nx] = fu[j-1][Nx] + 2*ru*(fu[j-1][Nx-1] - fu[j-1][Nx]) + tau*(A + fu[j-1][Nx]*fu[j-1][Nx]*fv[j-1][Nx] - (B+1)*fu[j-1][Nx]);
    vrightpart[Nx] = fv[j-1][Nx] + 2*rv*(fv[j-1][Nx-1] - fv[j-1][Nx]) + tau*( fu[j-1][Nx]*fu[j-1][Nx]*fv[j-1][Nx] + B*fu[j-1][Nx]);

    for (int i = 1; i < Nx; i++){
        urightpart[i] = fu[j-1][i]+ru*(fu[j-1][i+1] - 2*fu[j-1][i] + fu[j-1][i-1]) + tau*(A + fu[j-1][i]*fu[j-1][i]*fv[j-1][i] - (B+1)*fu[j-1][i]);
        vrightpart[i] = fv[j-1][i]+rv*(fv[j-1][i+1] - 2*fv[j-1][i] + fv[j-1][i-1]) + tau*( fu[j-1][i]*fu[j-1][i]*fv[j-1][i] + B*fu[j-1][i]);
    }

    solve_tridiagonal(urightpart, Nx+1, au, bu, cu);
    solve_tridiagonal(vrightpart, Nx+1, av, bv, cv);
    for (int i = 0; i < Nx+1; i++){
        fu[j][i] = urightpart[i];
        fv[j][i] = vrightpart[i];
    }
}

```

```

/*
fu[j][0] = fu[j-1][0]; // + tau*(A + fu[j][0]*fu[j][0]*fv[j][0] - (B+1)*fu[j][0]
    + 2*Du/h/h*(fu[j][1] - fu[j][0])) ;
fu[j][Nx-1] = fu[j-1][Nx-1]; // + tau*(A + fu[j][Nx-1]*fu[j][Nx-1]*fv[j][Nx-1]
    - (B+1)*fu[j][Nx-1] + 2*Du/h/h*(fu[j][Nx-2] - fu[j][Nx-1])) ;
fv[j][0] = fv[j-1][0]; // + tau*( fu[j][0]*fu[j][0]*fv[j][0] + B*fu[j][0] + 2*Dv
    /h/h*(fv[j][1] - fv[j][0])) ;
fv[j][Nx-1] = fv[j-1][Nx-1]; // + tau*( fu[j][Nx-1]*fu[j][Nx-1]*fv[j][Nx-1] + B*
    fu[j][Nx-1] + 2*Dv/h/h*(fv[j][Nx-2] - fv[j][Nx-1])) ;*/
}

FILE* fp;
fp = fopen("brus.dat", "w");
for(int j = 0; j <= Nt; j++){
    for(int i = 0; i <= Nx; i++){
        fprintf(fp, "%f %f %f %f\n", j*tau, i*h, fu[j][i], fv[j][i]);
    }
}
fclose(fp);
return 0;
}

```

Приведем результаты исполнения кода:

```

data = Import["C:/Users/Daria/Desktop/
ListPlot3D[data[[All, 1 ;; 3]]]
ListPlot3D[data[[All, {1, 2, 4}]]]

```

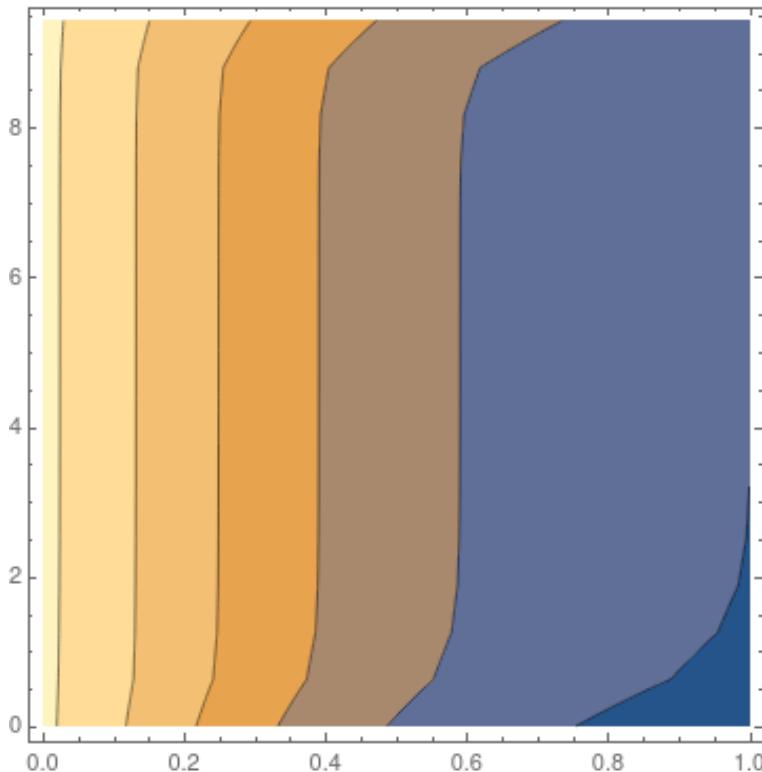


Рис. 17: $u(x,t)$ $A = B=1$, $D_u = 1$, $D_v = 100$

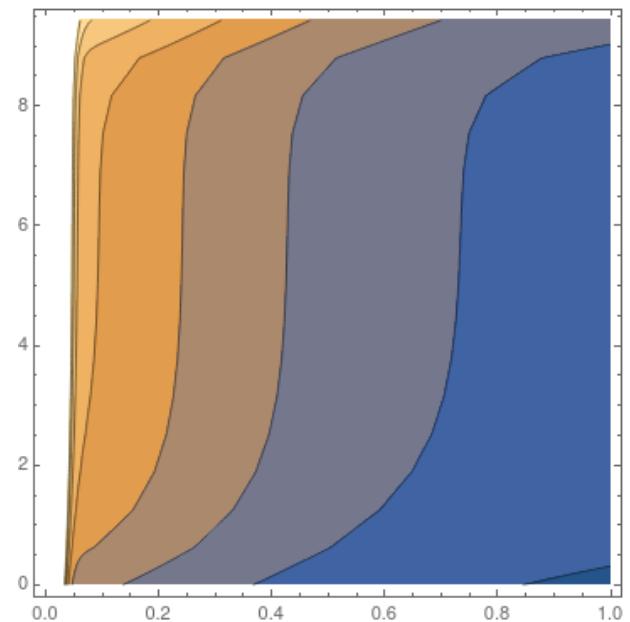


Рис. 18: $v(x,t)$ $A = B=1$, $D_u = 1$, $D_v = 100$

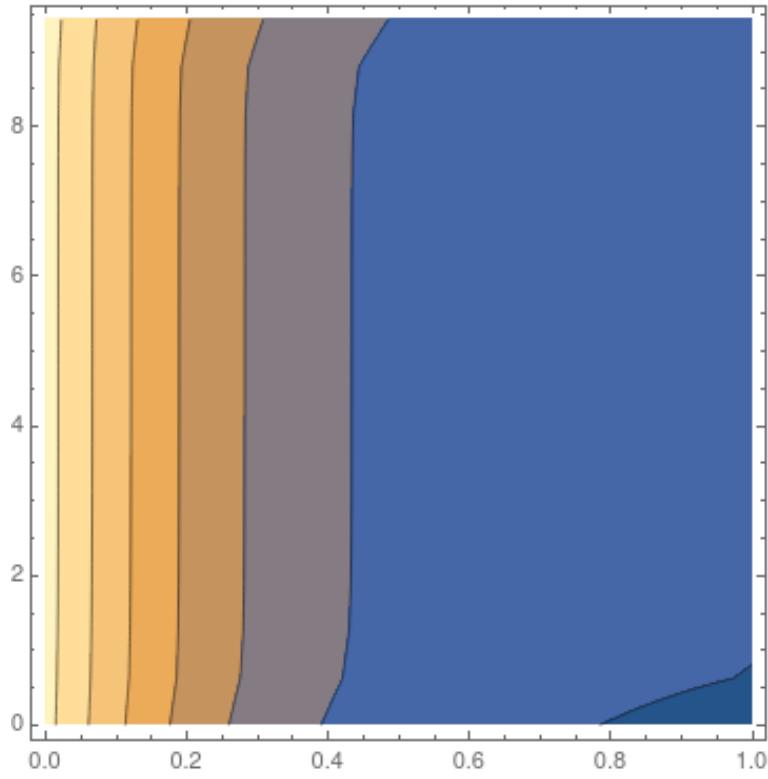


Рис. 19: $u(t)$ $A = 1$, $B = 3$, $D_u = 1$, $D_v = 100$

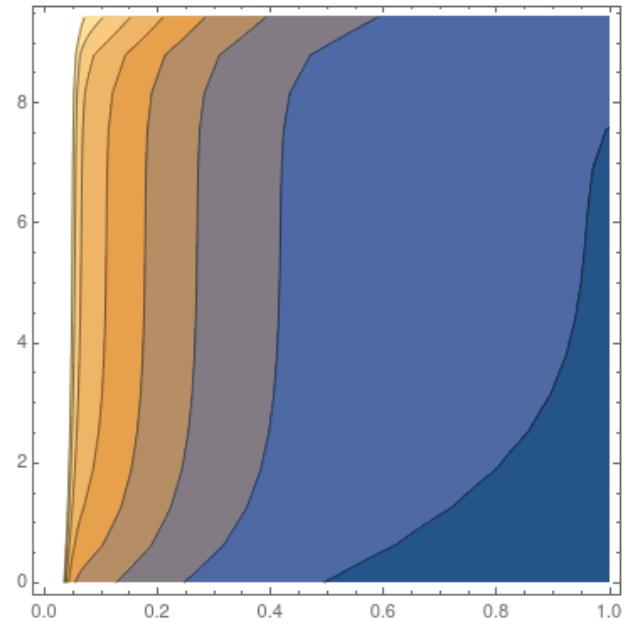


Рис. 20: $u(t)$ $A = 1$, $B = 3$, $D_u = 1$, $D_v = 100$

2.8 Анализ распределенного случая на Wolfram

```

noisefunc[const_, nperturb_: 10, noisefract_: 0.1] :=Module[{ipoints, ridx}, ipoints =
  Table[{xi, const}, {xi, 0.0, 1.0, 0.01}];
  ridx = Table[RandomInteger[{10, 90}], {nperturb}];
  Do[ipoints[[ridx[[ri]]]][[2]] += noisefract*RandomReal[{1, 1}], {ri, nperturb}];
  Return[Interpolation[ipoints, Method -> "Spline"]]];
nfu = noisefunc[1.0];
nfv = noisefunc[1.0];

soluv = NDSolve[{D[u[x, t], t] == Du *Derivative[2, 0][u][x, t] + a (b + 1) u[x, t] +
  v[x, t] (u[x, t])^2,
  D[v[x, t], t] == Dv *Derivative[2, 0][v][x, t] + b u[x, t] - v[x, t] (u[x, t])^2,
  u[x, 0] == nfu[x],
  v[x, 0] == nfv[x],
  Derivative[1, 0][u][0, t] == 0,
  Derivative[1, 0][u][L, t] == 0,
  Derivative[1, 0][v][0, t] == 0,
  Derivative[1, 0][v][L, t] == 0} /. {L > 50, Du > 1, Dv > 100, a > 1, b > 1}, {
  u, v}, {x, 0, 50}, {t, 0, 0.05 4000}, MaxSteps -> {15, Infinity}]

```

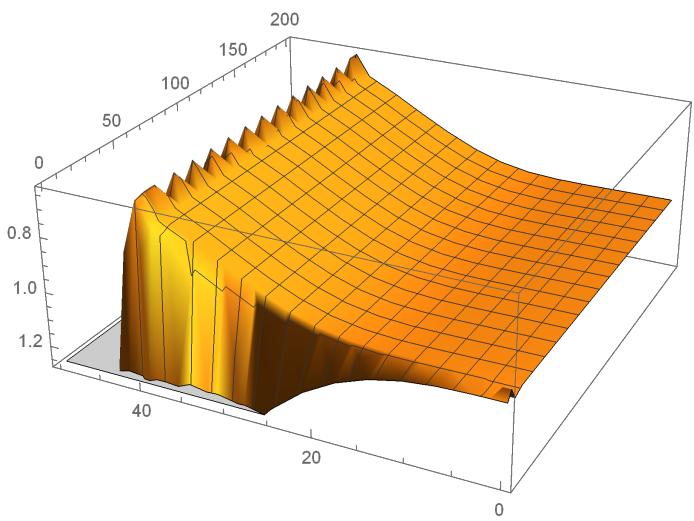


Рис. 21: $u(x,t)$ $A = B=1$, $Du = 1$, $Dv = 100$

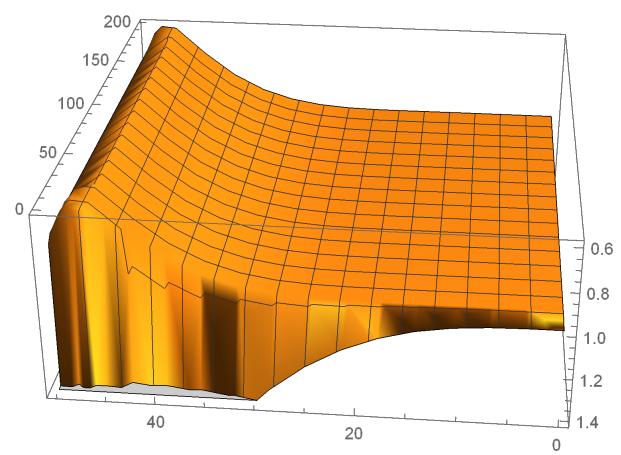


Рис. 22: $v(x,t)$ $A = B=1$, $Du = 1$, $Dv = 100$

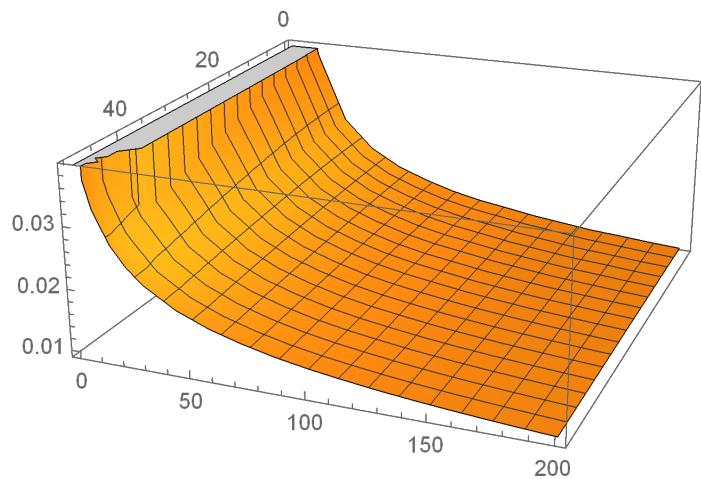


Рис. 23: $u(t)$ $A =1$, $B=1.5$, $Du = 1$, $Dv = 100$

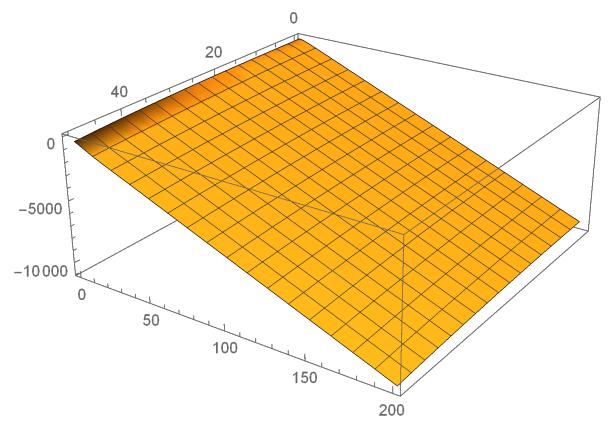


Рис. 24: $u(t)$ $A =1$, $B=1.5$, $Du = 1$, $Dv = 100$