

# 1 Постановка дифференциальной задачи

Система уравнений, описывающая нестационарное движение баротропного газа в области  $\Omega$ , выглядит следующим образом

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \frac{\partial \rho u_1}{\partial x_1} + \frac{\partial \rho u_2}{\partial x_2} &= 0, \\ \frac{\partial \rho u_1}{\partial t} + \frac{\partial \rho u_1^2}{\partial x_1} + \frac{\partial \rho u_2 u_1}{\partial x_2} + \frac{\partial p}{\partial x_1} &= \mu \left( \frac{4}{3} \frac{\partial^2 u_1}{\partial x_1^2} + \frac{\partial^2 u_1}{\partial x_2^2} + \frac{1}{3} \frac{\partial^2 u_2}{\partial x_1 \partial x_2} \right) + \rho f_1, \\ \frac{\partial \rho u_2}{\partial t} + \frac{\partial \rho u_1 u_2}{\partial x_1} + \frac{\partial \rho u_2^2}{\partial x_2} + \frac{\partial p}{\partial x_2} &= \mu \left( \frac{1}{3} \frac{\partial^2 u_1}{\partial x_1 \partial x_2} + \frac{\partial^2 u_2}{\partial x_1^2} + \frac{4}{3} \frac{\partial^2 u_2}{\partial x_2^2} \right) + \rho f_2, \end{aligned} \quad (1.1)$$

Неизвестные функции: плотность  $\rho$  и вектор скорости  $\mathbf{u}$  являются функциями переменных Эйлера

$$(t, \mathbf{x}) \in Q = [0, T] \times \bar{\Omega}.$$

Обозначим через  $\Omega_{nm}$  квадрат, координаты точек которого удовлетворяют неравенствам  $n < x < (n+1)$  и  $m < y < (m+1)$ . Множества точек, составляющие стороны квадрата  $\Omega_{nm}$  обозначим  $\Gamma_{nm}^{x-}$ ,  $\Gamma_{nm}^{x+}$ ,  $\Gamma_{nm}^{y-}$  и  $\Gamma_{nm}^{y+}$ , где индекс  $x$  или  $y$  означает какая из координат на стороне является постоянной, а  $+$  или  $-$  означает максимальное или минимальное значение принимает эта координата.

Заданная область: 9.  $\bar{\Omega} = \bar{\Omega}_{01} \cup \bar{\Omega}_{02} \cup \bar{\Omega}_{11} \cup \bar{\Omega}_{12} \cup \bar{\Omega}_{20} \cup \bar{\Omega}_{21} \cup \bar{\Omega}_{22}$ ,

Граничные условия для неизвестного решения:  $u_1|_{\Gamma_{01}^{x-} \cup \Gamma_{02}^{x-}} = w$ ,  $\frac{\partial u_2}{\partial y}|_{\Gamma_{20}^{y-}} = 0$ . На остальных участках границы функция плотности считается неизвестной и подлежит определению, а компоненты функции скорости равны нулю.

## 2 Схема А.Г.Соколова ПЛОТНОСТЬ-СКОРОСТЬ

Сеточная функция  $\mathbf{V}$ , приближающая функцию вектора скорости  $\mathbf{u}$ , определяется в узлах сетки  $\bar{Q}_{\tau\bar{h}}$ , а значения функции  $H$ , приближающей функцию плотности  $\rho$ , ищутся в узлах сетки  $Q_{\tau\bar{h}}^{1/2}$  по следующей схеме, аппроксимирующей систему (1.1)

$$\begin{aligned}
 & H_t + (\sigma_1\{\hat{H}, V_{1s_2}\}V_{1s_2})_{x_1} + (\sigma_2\{\hat{H}, V_{2s_1}\}V_{2s_1})_{x_2} = 0, \quad \mathbf{x} \in \Omega_h^{1/2}; \\
 & \hat{H}_{\bar{s}_1\bar{s}_2}V_{1t} + \hat{H}_{\bar{s}_1\bar{s}_2}\delta_1\{\hat{V}_1, V_1\} + \hat{H}_{\bar{s}_1\bar{s}_2}\delta_2\{\hat{V}_1, V_2\} + p(\hat{H}_{\bar{s}_2})_{\bar{x}_1} = \\
 & = \mu\left(\frac{4}{3}(\hat{V}_1)_{x_1\bar{x}_1} + (\hat{V}_1)_{x_2\bar{x}_2}\right) + \frac{\mu}{3}(V_2)_{x_1x_2}^0 + f_1\hat{H}_{\bar{s}_1\bar{s}_2}, \quad \text{при } \hat{H}_{\bar{s}_1\bar{s}_2} \neq 0, \\
 & \hat{V}_1 = 0, \quad \text{при } \hat{H}_{\bar{s}_1\bar{s}_2} = 0, \quad \mathbf{x} \in \Omega_h; \\
 & \hat{H}_{\bar{s}_1\bar{s}_2}V_{2t} + \hat{H}_{\bar{s}_1\bar{s}_2}\delta_1\{\hat{V}_2, V_1\} + \hat{H}_{\bar{s}_1\bar{s}_2}\delta_2\{\hat{V}_2, V_2\} + p(\hat{H}_{\bar{s}_1})_{\bar{x}_2} = \\
 & = \mu\left((\hat{V}_2)_{x_1\bar{x}_1} + \frac{4}{3}(\hat{V}_2)_{x_2\bar{x}_2}\right) + \frac{\mu}{3}(V_1)_{x_1x_2}^0 + f_2\hat{H}_{\bar{s}_1\bar{s}_2}, \quad \text{при } \hat{H}_{\bar{s}_1\bar{s}_2} \neq 0, \\
 & \hat{V}_2 = 0, \quad \text{при } \hat{H}_{\bar{s}_1\bar{s}_2} = 0, \quad \mathbf{x} \in \Omega_h.
 \end{aligned} \tag{2.1}$$

В граничных узлах  $\gamma_{\bar{h}}$  значения функции  $\mathbf{V}$  считаются известными из граничных условий. В граничных узлах, где газ втекает в область, задаются значения плотности газа, которые также берутся из граничных условий.

Разностные уравнения в индексах имеют вид:

$$\begin{aligned}
 & \frac{H_{m_1, m_2}^{n+1} - H_{m_1, m_2}^n}{\tau} + \\
 & + \frac{((\tilde{V}_2)_{m_1, m_2+1}^n - |(\tilde{V}_2)_{m_1, m_2+1}^n|)H_{m_1, m_2+1}^{n+1}}{2h_2} + \frac{((\tilde{V}_1)_{m_1+1, m_2}^n - |(\tilde{V}_1)_{m_1+1, m_2}^n|)H_{m_1+1, m_2}^{n+1}}{2h_1} + \\
 & + \left(\frac{1}{2h_1}((\tilde{V}_1)_{m_1+1, m_2}^n + |(\tilde{V}_1)_{m_1+1, m_2}^n| - (\tilde{V}_1)_{m_1, m_2}^n + |(\tilde{V}_1)_{m_1, m_2}^n|) + \right. \\
 & \left. + \frac{1}{2h_2}((\tilde{V}_2)_{m_1, m_2+1}^n + |(\tilde{V}_2)_{m_1, m_2+1}^n| - (\tilde{V}_2)_{m_1, m_2}^n + |(\tilde{V}_2)_{m_1, m_2}^n|)\right) H_{m_1, m_2}^{n+1} - \\
 & - \frac{((\tilde{V}_1)_{m_1, m_2}^n + |(\tilde{V}_1)_{m_1, m_2}^n|)H_{m_1-1, m_2}^{n+1}}{2h_1} - \frac{((\tilde{V}_2)_{m_1, m_2}^n + |(\tilde{V}_2)_{m_1, m_2}^n|)H_{m_1, m_2-1}^{n+1}}{2h_2} = 0, \\
 & 0 \leq m_1 < M_1, \quad 0 \leq m_2 < M_2, \quad n \geq 0.
 \end{aligned} \tag{2.2}$$

$$\begin{aligned}
& (\tilde{H})_{m_1, m_2}^{n+1} \left( \frac{(V_{1m_1, m_2}^{n+1} - V_{1m_1, m_2}^n)}{\tau} - \frac{|V_{2m_1, m_2}^n| + V_{2m_1, m_2}^n}{2h_2} V_{1m_1, m_2-1}^{n+1} - \right. \\
& - \frac{|V_{1m_1, m_2}^n| + V_{1m_1, m_2}^n}{2h_1} V_{1m_1-1, m_2}^{n+1} + \left( \frac{|V_{1m_1, m_2}^n|}{h_1} + \frac{|V_{2m_1, m_2}^n|}{h_2} \right) V_{1m_1, m_2}^{n+1} + \\
& + \frac{V_{1m_1, m_2}^n - |V_{1m_1, m_2}^n|}{2h_1} V_{1m_1+1, m_2}^{n+1} + \frac{V_{2m_1, m_2}^n - |V_{2m_1, m_2}^n|}{2h_2} V_{1m_1, m_2+1}^{n+1} \Big) + \\
& + \frac{p(H_{1m_1, m_2}^{n+1}) - p(H_{1m_1-1, m_2}^{n+1})}{h_1} = \\
& = \mu \left( \frac{4}{3} \frac{V_{1m_1-1, m_2}^{n+1} - 2V_{1m_1, m_2}^{n+1} + V_{1m_1+1, m_2}^{n+1}}{h_1^2} + \frac{V_{1m_1, m_2-1}^{n+1} - 2V_{1m_1, m_2}^{n+1} + V_{1m_1, m_2+1}^{n+1}}{h_2^2} \right) + \\
& + \frac{\mu}{3} \frac{V_{2m_1-1, m_2-1}^n - V_{2m_1-1, m_2+1}^n - V_{2m_1+1, m_2-1}^n + V_{2m_1+1, m_2+1}^n}{4h_1 h_2} + f_{1m_1, m_2}^{n+1} (\tilde{H})_{m_1, m_2}^{n+1}, \\
& \hat{V}_{1m_1, m_2} = 0, \quad \text{при } \tilde{H}_{\mathbf{m}}^{n+1} \neq 0, \\
& \quad \text{при } \tilde{H}_{m_1, m_2}^{n+1} = 0; \quad \mathbf{x} \in \Omega_h.
\end{aligned} \tag{2.3}$$

$$\begin{aligned}
& (\tilde{H})_{m_1, m_2}^{n+1} \left( \frac{(V_{2m_1, m_2}^{n+1} - V_{2m_1, m_2}^n)}{\tau} - \frac{|V_{2m_1, m_2}^n| + V_{2m_1, m_2}^n}{2h_2} V_{2m_1, m_2-1}^{n+1} - \right. \\
& - \frac{|V_{1m_1, m_2}^n| + V_{1m_1, m_2}^n}{2h_1} V_{2m_1-1, m_2}^{n+1} + \left( \frac{|V_{1m_1, m_2}^n|}{h_1} + \frac{|V_{2m_1, m_2}^n|}{h_2} \right) V_{2m_1, m_2}^{n+1} + \\
& + \frac{V_{1m_1, m_2}^n - |V_{1m_1, m_2}^n|}{2h_1} V_{2m_1+1, m_2}^{n+1} + \frac{V_{2m_1, m_2}^n - |V_{2m_1, m_2}^n|}{2h_2} V_{2m_1, m_2+1}^{n+1} \Big) + \\
& + \frac{p(H_{2m_1, m_2}^{n+1}) - p(H_{2m_1, m_2-1}^{n+1})}{h_2} = \\
& = \mu \left( \frac{V_{2m_1-1, m_2}^{n+1} - 2V_{2m_1, m_2}^{n+1} + V_{2m_1+1, m_2}^{n+1}}{h_1^2} + \frac{4}{3} \frac{V_{2m_1, m_2-1}^{n+1} - 2V_{2m_1, m_2}^{n+1} + V_{2m_1, m_2+1}^{n+1}}{h_2^2} \right) + \\
& + \frac{\mu}{3} \frac{V_{1m_1-1, m_2-1}^n - V_{1m_1-1, m_2+1}^n - V_{1m_1+1, m_2-1}^n + V_{1m_1+1, m_2+1}^n}{4h_1 h_2} + f_{2m_1, m_2}^{n+1} (\tilde{H})_{m_1, m_2}^{n+1}, \\
& \hat{V}_{2m_1, m_2} = 0, \quad \text{при } \tilde{H}_{\mathbf{m}}^{n+1} \neq 0, \\
& \quad \text{при } \tilde{H}_{m_1, m_2}^{n+1} = 0; \quad \mathbf{x} \in \Omega_h.
\end{aligned} \tag{2.4}$$

По этой схеме на каждом временном слое решается СЛАУ, решением которой является сеточная функция плотности  $H^{n+1}$ , а затем в любом порядке решаются СЛАУ, которые задают функции  $V_1^{n+1}$  и  $V_2^{n+1}$ . Последние две системы можно решать независимо.

### 3 Заполнение матрицы первой системы

```
void fill_first (int k, std::map<unsigned int, double> &A,
std::vector<double> &B, trio &essential)
{
    discrete_function &H = essential.m_tdfH.get_cut (k);
    discrete_function &V1 = essential.m_tdfV1.get_cut (k);
    discrete_function &V2 = essential.m_tdfV2.get_cut (k);

    double t = essential.m_tdfH.get_scale ()->get_time (k + 1);
    double tau = essential.m_tdfH.get_scale ()->get_parameters ().m_t_step;
    double h1 = essential.m_tdfH.get_grid ()->get_parameters ().m_x_step;
    double h2 = essential.m_tdfH.get_grid ()->get_parameters ().m_y_step;

    MatrixSetter A_at (A, H);
    VectorSetter B_at (B, H);

    H.do_for_each ([&] (index ij, point xy, discrete_function &)
    {
        int m1 = ij.first, m2 = ij.second;
        double x = xy.first, y = xy.second;

        if (process_H_edge (m1, m2, A_at, B_at))
            return;

        A_at(m1,m2,0,0)=1.
            +tau*(xpabs(V1.tilda(m1+1,m2))-xmabs(V1.tilda(m1,m2)))/2./h1
            +tau*(xpabs(V2.tilda(m1,m2+1))-xmabs(V2.tilda(m1,m2)))/2./h2;

        A_at(m1,m2,+1,0)=tau*xmabs(V1.tilda (m1+1,m2))/2./h1;
        A_at(m1,m2,0,+1)=tau*xmabs(V2.tilda (m1,m2+1))/2./h2;
        A_at(m1,m2,-1,0)=-tau*xpabs(V1.tilda (m1,m2))/2./h1;
        A_at(m1,m2,0,-1)=-tau*xpabs(V2.tilda (m1,m2))/2./h2;

        B_at(m1,m2)=H.val(m1,m2)+tau*f_1 (t,x,y);
    });
}
```

## 4 Заполнение матрицы второй системы

```
void fill_second (int k, std::map<unsigned int, double> &A,
std::vector<double> &B, trio &essential)
{
    discrete_function &H = essential.m_tdfH.get_cut (k + 1);
    discrete_function &V1 = essential.m_tdfV1.get_cut (k);
    discrete_function &V2 = essential.m_tdfV2.get_cut (k);

    double t = essential.m_tdfV1.get_scale ()->get_time (k + 1);
    double tau = essential.m_tdfV1.get_scale ()->get_parameters ().m_t_step;
    double h1 = essential.m_tdfV1.get_grid ()->get_parameters ().m_x_step;
    double h2 = essential.m_tdfV1.get_grid ()->get_parameters ().m_y_step;

    MatrixSetter A_at (A, V1);
    VectorSetter B_at (B, V1);

    V1.do_for_each ([&] (index ij, point xy, discrete_function &)
    {
        int m1 = ij.first, m2 = ij.second;
        double x = xy.first, y = xy.second;
        double check = (H.val(m1,m2)+H.val(m1,m2-1)+H.val(m1-1,m2)+H.val(m1-1,m2-1))/4.0;

        if (process_V_edge (m1, m2, check, A_at, B_at))
            return;

        A_at(m1,m2,0,0)=check*(1.+tau/h1*fabs(V1.val(m1,m2))+tau/h2*fabs(V2.val(m1,m2)))
            +tau*MIU*(8./3./h1/h1+2./h2/h2);

        A_at(m1,m2,-1,0)=-tau/2./h1*fpabs(V1.val(m1,m2))*check
            -4./3.*tau*MIU/h1/h1;
        A_at(m1,m2,+1,0)=tau/2./h1*fmabs(V1.val(m1,m2))*check
            -4./3.*tau*MIU/h1/h1;
        A_at(m1,m2,0,-1)=-tau/2./h2*fpabs(V2.val(m1,m2))*check
            -tau*MIU/h2/h2;
        A_at(m1,m2,0,+1)=tau/2./h2*fmabs(V2.val(m1,m2))*check
            -tau*MIU/h2/h2;

        B_at(m1,m2)=check*V1.val(m1,m2)-tau/h1*(p(H.left(m1,m2))-p(H.left(m1-1,m2)))
            +tau*MIU/12./h1/h2*(V2.val(m1+1,m2+1)-V2.val(m1+1,m2-1)-V2.val(m1-1,m2+1)+V2.val(m1-1,m2-1))
            +tau*f_2(t,x,y)*check;
    });
}
```

## 5 Заполнение матрицы третьей системы

```

void fill_third (int k, std::map<unsigned int, double> &A,
std::vector<double> &B, trio &essential)
{
    discrete_function &H = essential.m_tdfH.get_cut (k + 1);
    discrete_function &V1 = essential.m_tdfV1.get_cut (k);
    discrete_function &V2 = essential.m_tdfV2.get_cut (k);

    double t = essential.m_tdfV2.get_scale ()->get_time (k + 1);
    double tau = essential.m_tdfV2.get_scale ()->get_parameters ().m_t_step;
    double h1 = essential.m_tdfV2.get_grid ()->get_parameters ().m_x_step;
    double h2 = essential.m_tdfV2.get_grid ()->get_parameters ().m_y_step;

    MatrixSetter A_at (A, V2);
    VectorSetter B_at (B, V2);

    V2.do_for_each ([&] (index ij, point xy, discrete_function &)
    {
        int m1 = ij.first, m2 = ij.second;
        double x = xy.first, y = xy.second;
        double check = (H.val(m1,m2)+H.val(m1,m2-1)+H.val(m1-1,m2)+H.val(m1-1,m2-1))/4.0;

        if (process_V_condition (m1, m2, A_at, B_at))
            return;

        if (process_V_edge (m1, m2, check, A_at, B_at))
            return;

        A_at(m1,m2,0,0)=check*(1.+tau/h1*fabs(V1.val(m1,m2))+tau/h2*fabs(V2.val(m1,m2)))
            +tau*MIU*(2./h1/h1+8./3./h2/h2);

        A_at(m1,m2,-1,0)=-tau/2./h1*xpabs(V1.val(m1,m2))*check
            -tau*MIU/h1/h1;
        A_at(m1,m2,+1,0)=tau/2./h1*xmabs(V1.val(m1,m2))*check
            -tau*MIU/h1/h1;
        A_at(m1,m2,0,-1)=-tau/2./h2*ypabs(V2.val(m1,m2))*check
            -4./3.*tau*MIU/h2/h2;
        A_at(m1,m2,0,+1)=tau/2./h2*ymabs(V2.val(m1,m2))*check
            -4./3.*tau*MIU/h2/h2;

        B_at(m1,m2)=check*V2.val(m1,m2)-tau/h2*(p(H.right(m1,m2))-p(H.right(m1,m2-1)))
            +tau*MIU/12./h1/h2*(V1.val(m1+1,m2+1)-V1.val(m1+1,m2-1)-V1.val(m1-1,m2+1)+V1.val(m1-1,m2-1))
            +tau*f_3(t,x,y)*check;
    });
}

```

## 6 Таблицы невязок в непрерывном случае:

### 6.1 $\text{MIU} = 0.100$

Таблица 1: Функция: Н Тип невязки: С

M/N	21	42	84	168
21	1e+00	7e-01	4e-01	1e+01
42	1e+00	7e-01	4e-01	2e-01
84	1e+00	6e-01	4e-01	2e-01
168	1e+00	6e-01	4e-01	2e-01

Таблица 2: Функция: V1 Тип невязки: С

M/N	21	42	84	168
21	6e-01	3e-01	2e-01	1e+00
42	7e-01	4e-01	2e-01	8e-02
84	7e-01	4e-01	2e-01	1e-01
168	7e-01	4e-01	2e-01	1e-01

Таблица 3: Функция: V2 Тип невязки: С

M/N	21	42	84	168
21	2e-01	1e-01	1e-01	1e+00
42	2e-01	1e-01	8e-02	5e-02
84	2e-01	1e-01	8e-02	4e-02
168	2e-01	1e-01	8e-02	4e-02

Таблица 4: Функция: Н Тип невязки: L2

M/N	21	42	84	168
21	3e+00	2e+00	1e+00	9e+00
42	3e+00	1e+00	9e-01	7e-01
84	3e+00	1e+00	7e-01	5e-01
168	2e+00	1e+00	7e-01	4e-01

Таблица 5: Функция: V1 Тип невязки: L2

M/N	21	42	84	168
21	2e+00	8e-01	3e-01	9e-01
42	2e+00	1e+00	5e-01	2e-01
84	2e+00	1e+00	5e-01	2e-01
168	2e+00	1e+00	6e-01	3e-01

Таблица 6: Функция: V2 Тип невязки: L2

M/N	21	42	84	168
21	7e-01	4e-01	3e-01	1e+00
42	7e-01	4e-01	2e-01	1e-01
84	7e-01	4e-01	2e-01	1e-01
168	6e-01	4e-01	2e-01	1e-01



## 6.2 $MIU = 0.010$

Таблица 7: Функция: Н Тип невязки: С

M/N	21	42	84	168
21	1e+00	7e-01	4e-01	7e+01
42	1e+00	7e-01	4e-01	2e-01
84	1e+00	7e-01	4e-01	2e-01
168	1e+00	7e-01	4e-01	2e-01

Таблица 8: Функция: V1 Тип невязки: С

M/N	21	42	84	168
21	7e-01	3e-01	2e-01	4e+00
42	7e-01	4e-01	2e-01	9e-02
84	7e-01	4e-01	2e-01	1e-01
168	7e-01	4e-01	2e-01	1e-01

Таблица 9: Функция: V2 Тип невязки: С

M/N	21	42	84	168
21	3e-01	2e-01	1e-01	6e+00
42	3e-01	2e-01	9e-02	6e-02
84	3e-01	2e-01	9e-02	5e-02
168	3e-01	2e-01	9e-02	5e-02

Таблица 10: Функция: Н Тип невязки: L2

M/N	21	42	84	168
21	3e+00	2e+00	1e+00	3e+01
42	3e+00	1e+00	9e-01	7e-01
84	3e+00	1e+00	8e-01	5e-01
168	3e+00	1e+00	7e-01	4e-01

Таблица 11: Функция: V1 Тип невязки: L2

M/N	21	42	84	168
21	2e+00	9e-01	4e-01	3e+00
42	2e+00	1e+00	5e-01	2e-01
84	2e+00	1e+00	5e-01	2e-01
168	2e+00	1e+00	6e-01	3e-01

Таблица 12: Функция: V2 Тип невязки: L2

M/N	21	42	84	168
21	7e-01	4e-01	3e-01	8e+00
42	7e-01	4e-01	2e-01	2e-01
84	7e-01	4e-01	2e-01	1e-01
168	7e-01	4e-01	2e-01	1e-01

### 6.3 $MIU = 0.001$

Таблица 13: Функция: Н Тип невязки: С

M/N	21	42	84	168
21	1e+00	7e-01	4e-01	1e+02
42	1e+00	7e-01	4e-01	2e-01
84	1e+00	7e-01	4e-01	2e-01
168	1e+00	7e-01	4e-01	2e-01

Таблица 14: Функция: V1 Тип невязки: С

M/N	21	42	84	168
21	7e-01	3e-01	2e-01	4e+00
42	7e-01	4e-01	2e-01	9e-02
84	7e-01	4e-01	2e-01	1e-01
168	7e-01	4e-01	2e-01	1e-01

Таблица 15: Функция: V2 Тип невязки: С

M/N	21	42	84	168
21	3e-01	2e-01	1e-01	5e+00
42	3e-01	2e-01	9e-02	6e-02
84	3e-01	2e-01	9e-02	5e-02
168	3e-01	2e-01	9e-02	5e-02

Таблица 16: Функция: Н Тип невязки: L2

M/N	21	42	84	168
21	3e+00	2e+00	1e+00	3e+01
42	3e+00	1e+00	9e-01	7e-01
84	3e+00	1e+00	8e-01	5e-01
168	3e+00	1e+00	7e-01	4e-01

Таблица 17: Функция: V1 Тип невязки: L2

M/N	21	42	84	168
21	2e+00	9e-01	4e-01	4e+00
42	2e+00	1e+00	5e-01	2e-01
84	2e+00	1e+00	6e-01	3e-01
168	2e+00	1e+00	6e-01	3e-01

Таблица 18: Функция: V2 Тип невязки: L2

M/N	21	42	84	168
21	7e-01	5e-01	3e-01	9e+00
42	7e-01	4e-01	2e-01	2e-01
84	7e-01	4e-01	2e-01	1e-01
168	7e-01	4e-01	2e-01	1e-01

**6.4 Стабилизация течения в разрывном случае:**

**6.5 Примеры графиков, непрерывный случай:**

cont.png

## 6.6 Примеры графиков, разрывный случай

cont.png

## 7 Выводы