Interdisciplinary Information Sciences Vol. 22, No. 2 (2016) 199–213
©Graduate School of Information Sciences, Tohoku University
ISSN 1340-9050 print/1347-6157 online
DOI 10.4036/iis.2016.A.03

Construction of the Algorithm and Computation of the Flow of an Incompressible Fluid Based on the Supporting Operators Method

Nikita AFANASIEV*

*Faculty of Computational Mathematics and Cybernetics,
Lomonosov Moscow State University, Moscow 119991, Russia*

The paper is devoted to the specific problem of continuum mechanics - numerical computation of the solution of two-dimensional Navier–Stokes equations for viscous incompressible fluids. The author plans to use the constructed numerical methods in hemodynamics to compute blood flow in elastic vessels. The support operators technique was chosen to construct the methods because it allows to construct conservative numerical methods, which can be relatively easy implemented on unstructured meshes. These properties are very important in hemodynamics. The whole family of such conservative methods was built. One of the methods was tested on the problem of fluid flow between two plane-parallel plates with different values of Reynolds number.

KEYWORDS: fluid dynamics, finite volume method, support operators method, conservative numerical methods

1. Introduction

This paper describes one of the ways to approximate two-dimensional Navier–Stokes equations for viscous incompressible fluids, based on the finite volume method (FVM). Though there are many other ways to approach the equations, including the finite element method, the FVM was chosen because of several reasons: it allows to construct schemes with a very important property of conservativity. In other words, it allows to construct schemes which approximate different conservation laws (of mass, momentum, energy and others if needed). The FVM is also relatively easy to implement on various unstructured meshes of complex domains, especially when the boundaries of such domains are flexible. These properties of the FVM are important because the constructed schemes will be used in hemodynamics to compute blood flow in vessels, which walls can expand or reduce depending on the pressure inside and outside the vessels. Among several ways to construct the FVM the support operators method was chosen. It is based on special homogenizations of mesh functions over different areas (such as cells and regions around nodes) and on the idea that the mesh analogues of differential operators *div*, *grad* and others should have the same properties as the differential operators (for example, Gauss' theorem should work for them too). This approach was mostly inspired by the works of A. Samarski, A. Favorski, V. Tishkin [1], V. Goloviznin [2], A. Koldoba [3] and others.

The paper is divided into two parts. The first part includes the detailed construction of the family of conservative numerical methods for two-dimensional Navier–Stokes equations and describes a way of implementation of the methods. The second part includes the results of some tests for one of the constructed methods: in particularly, it is tested on the problem of fluid flow between two plane-parallel plates. The computations were conducted for flows with different parameters, special focus was given on what happens with the flows with different Reynolds numbers.

2. Navier–Stokes Equations

Navier–Stokes equations describe the motion of viscous incompressible fluid. Their form in Euler coordinates [4]:

$$div\,\mathbf{v} = 0, \tag{2.1}$$

$$\rho\frac{\partial\mathbf{v}}{\partial t} + \rho(\mathbf{v},\nabla)\mathbf{v} + grad\,p = \mu\,\Delta\mathbf{v}, \mathbf{r}\in S, t > 0, \tag{2.2}$$

$\mathbf{r}\in S, t > 0$.

Here ρ is fluid's density, $\mathbf{v} = \mathbf{v}(\mathbf{r},t)$ - fluid's velocity, $p = p(\mathbf{r},t)$ - pressure, μ - fluid's coefficient of dynamic viscosity, $\mathbf{r}$ - radius-vector of point in space, t - time, S - open domain. For the simplicity we will examine the two-

Received March 14, 2016; Accepted April 15, 2016; J-STAGE Advance published November 16, 2016
2010 Mathematics Subject Classification: Primary 62F12, Secondary 62F05.

*Corresponding author. E-mail: gandiken@yandex.ru dimensional case. Equation (2.1) is called the continuity equation and it represents the law of conservation of mass, (2.2) - the motion equation and it represents the law of conservation of momentum.

It is assumed that there are no sources of fluid inside domain S, μ is a scalar constant and there is also no external force. Some boundary conditions are given on the boundary $\partial S = C$ and some initial conditions in the domain $S \cup C$. It is required to find sufficiently differentiable functions $\mathbf{v}, p$, satisfying (2.1), (2.2) in S, boundary conditions on C and initial conditions in $S \cup C$.

Notice that in consequence of (2.1): $\Delta \mathbf{v} = grad\, div\, \mathbf{v} - rot\, rot\, \mathbf{v} = -rot\, rot\, \mathbf{v}$.

Due to the fact that usage of differential equations requires the functions to be smooth enough, equations (2.1), (2.2) do not describe the discontinuous motion of fluid. In order to construct a numerical method, which can approximate the discontinuous motion of fluid, we will use one of the integral analogues of Navier–Stokes equations for incompressible fluid [4]:

$$\oint\limits_C (\mathbf{v}(\mathbf{r}, t), \mathbf{n})\, dl = 0, \ \forall t > 0 \tag{2.3}$$

$$\rho \int\limits_S [v_i(\mathbf{r}, t + \Delta t) - v_i(\mathbf{r}, t)]\, dS + \rho \int\limits_t^{t+\Delta t} \oint\limits_C (v_i \mathbf{v}(\mathbf{r}, t'), \mathbf{n})\, dl\, dt' + \int\limits_t^{t+\Delta t} \oint\limits_C p(\mathbf{r}, t')\, n_i\, dl\, dt'$$
$$= -\mu \int\limits_t^{t+\Delta t} \oint\limits_C [rot\, \mathbf{v}(\mathbf{r}, t'), \mathbf{n}]_i\, dl\, dt', \ \forall t > 0, \ \Delta t > 0, \ i = 1, 2, \tag{2.4}$$

where $\mathbf{N}$ - external unit normal to the contour line C, n_1, n_2, v_1, v_2 - x and y-components of external unit normal and velocity in some fixed Cartesian coordinate system. System (2.1), (2.2) can be deducted from (2.3), (2.4), using two-dimensional analogue of Gauss' theorem [4], the average theorem and directing $\Delta t \to 0$.

Notice that system's (2.3), (2.4) requirements of the differentiability of functions $\mathbf{v}, p$ are less strict than in system (2.1), (2.2). Let's approximate equations (2.3), (2.4).

3. Construction of Numerical Method

3.1 Discretization of the Domain and Variables

Let's choose some bounded two-dimensional domain and build a random unstructured mesh on it with different polygons as cells (see Fig. 1).

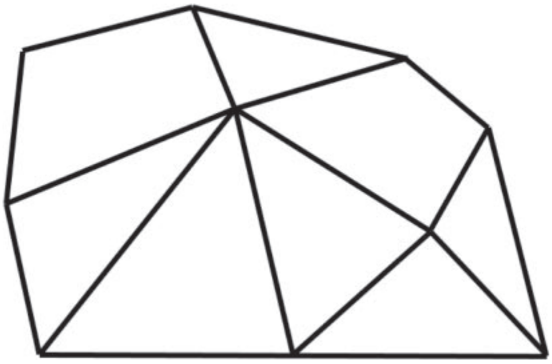

Fig. 1. Example of an unstructured mesh.

Let ω_h be a set of internal nodes of the mesh, γ_h - set of boundary nodes of the mesh, Ω_h - set of cells of the mesh, where h - maximal length of edges of the mesh.

Let's also build a time mesh: $\omega_\tau = \{t_n = n\tau \mid n = \overline{0, T}\}$, where τ - time step.

The variables in the system (2.3), (2.4) are a scalar function p and vector function $\mathbf{v}$. Let's put the values of scalar functions to the cells (mass centres of the cells in particular), values of vector functions to the nodes. In other words, we build two sets of mesh functions:

$$F_\omega = \{\mathbf{a}_\omega \mid \omega \in \omega_h \cup \gamma_h, \mathbf{a}_\omega \in \mathbb{R}^2\}$$
$$F_\Omega = \{f_\Omega \mid \Omega \in \Omega_h, \ f_\Omega \in \mathbb{R}\}$$

Then pressure $p(x \in \Omega, t_n) \approx p^n_\Omega \in F_\Omega$ and velocity $\mathbf{v}(x_\omega, t_n) \approx \mathbf{v}^n_\omega \in F_\omega$.

3.2 Approximation of Integral Equations

Due to the fact that scalars are put into cells and vectors - into nodes, it will be reasonable to approximate scalar continuity equation (2.3) for every cell $\Omega \in \Omega_h$, and vector motion equation (2.4) for every internal node $\omega \in \omega_h$.

3.2.1 Continuity Equation

Let's fix any cell Ω of the mesh. Equation (2.3) must be valid not only for domain S, but also for every subdomain of S. Let's write down equation (2.3) in the moment of time $t = t_n$ for the domain, which is bounded by chosen cell:

$$\oint_{\partial\Omega} (\mathbf{v}(\mathbf{r}, t_n), \mathbf{n})\, dl = 0 \tag{3.1}$$

Now we will approximate the contour integral in (3.1) this way: integral over every edge of the polygon is approximated using the trapezoidal rule [5] and the values of $\mathbf{v}$ in the polygon's nodes.

$$\oint_{\partial\Omega} (\mathbf{v}(\mathbf{r}, t_n), \mathbf{n})\, dl \approx \sum_{i=1}^{m_\Omega} \left(\frac{\mathbf{v}^n_{\Omega,i} + \mathbf{v}^n_{\Omega,i+1}}{2}, \mathbf{N}_{\Omega,i} \right) l_{\Omega,i}$$

Here m_Ω is a number of edges in polygon Ω, $l_{\Omega,i}$ - lenght of edge i, $\mathbf{N}_{\Omega,i}$ - external unit normal to edge i, $\mathbf{v}_{\Omega,i}$ - value of vector mesh function $\mathbf{v}$ in node i of polygon Ω, $\mathbf{v}_{\Omega,m_\Omega+1} = \mathbf{v}_{\Omega,1}$ (see Fig. 2).

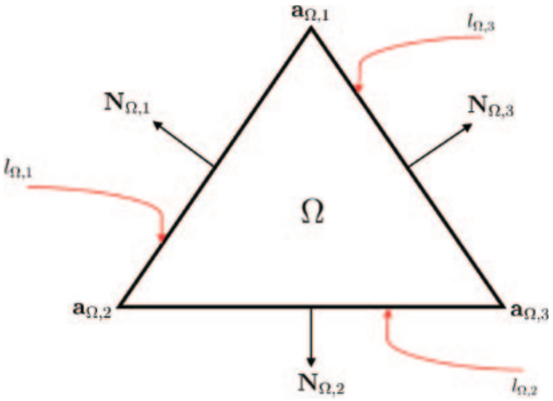

Fig. 2. Template for *DIV* ($m_\Omega = 3$).

Let's define mesh operator $DIV : F_\omega \to F_\Omega$:

$$(DIV\,\mathbf{a})_\Omega = \frac{1}{S_\Omega} \sum_{i=1}^{m_\Omega} \left(\frac{\mathbf{a}_{\Omega,i} + \mathbf{a}_{\Omega,i+1}}{2}, \mathbf{N}_{\Omega,i} \right) l_{\Omega,i}, \tag{3.2}$$

where S_Ω is the area of cell Ω. Then we have an approximation for the equation (3.1):

$$S_\Omega (DIV\,\mathbf{v}^n)_\Omega = 0, \quad \forall\Omega \in \Omega_h \tag{3.3}$$

Due to the trapezoidal rule, equation (3.3) approaches (3.1) with the third order of approximation accuracy. Notice that if one sums equations (3.3) for two adjoining cells Ω_1 and Ω_2, one gets an equation, which approximates (3.1) for the domain $\Omega_1 \cup \Omega_2$. Therefore, summing equation (3.3) for all cells of the mesh, we get the equation, which approaches (2.3) in the moment of time t_n.

3.2.2 *DIV* Operator and its Adjoint Operator

Let's examine the invariant definition of the differential divergence operator [4]:

$$div\,\mathbf{a} = \lim_{|S'|\to 0} \frac{1}{|S'|} \oint_{\partial S'} (\mathbf{a}, \mathbf{n}) dl, \tag{3.4}$$

where S' is the domain around the point, in which the operator is defined, $|S'|$ - the area of this domain, $\mathbf{n}$ - the external unit normal to the contour line $\partial S'$. Comparing (3.2) and (3.4) we can say that $(DIV\,\mathbf{v})_\Omega$ is an approximation of operator $div\,\mathbf{v}$ in cell Ω, which in general has the first order of approximation accuracy on a random unstructured mesh (on a quadrilateral mesh, where every internal node has exactly 4 adjoint cells, (3.2) has the second order of approximation accuracy).

Now let's examine the invariant definition of the differential gradient operator [4]:

$$grad\, f = \lim_{|S'|\to 0} \frac{1}{|S'|} \oint_{\partial S'} f\mathbf{n}\, dl, \tag{3.5}$$

where the designations are similar to (3.4). Notice that there is a similar to (3.5) contour integral (more precisely its components) in (2.4), so an integral of such kind has to be approximated.

It is known that for sufficiently smooth finite in domain S functions f and $\mathbf{a}$ this statement is valid:

$$\int_S f\, div\, \mathbf{a}\, dS = -\int_S (\mathbf{a}, grad\, f)\, dS \tag{3.6}$$

Let's define scalar products in the spaces of finite in domain S scalar (f, p) and vector $(\mathbf{u}, \mathbf{v})$ functions:

$$(f, p)_1 = \int_S f\, p\, dS,$$

$$(\mathbf{u}, \mathbf{v})_2 = \int_S (\mathbf{u}, \mathbf{v})\, dS.$$

Then (3.6) can be rewritten:

$$(div\, \mathbf{a}, f)_1 = -(\mathbf{a}, grad\, f)_2 \Leftrightarrow grad = -(div)^*$$

Now we need to construct an operator $GRAD$: $GRAD = -(DIV)^*$. For that we will form a so called control volume around every internal node $\omega \in \omega_h$. For every cell, which has ω as a node, we connect the mass center of the cell with the middles of those edges of the cell, which have ω as one of the endpoints. The resulting polygon around ω we will call the control volume of the node ω (see Fig. 3).

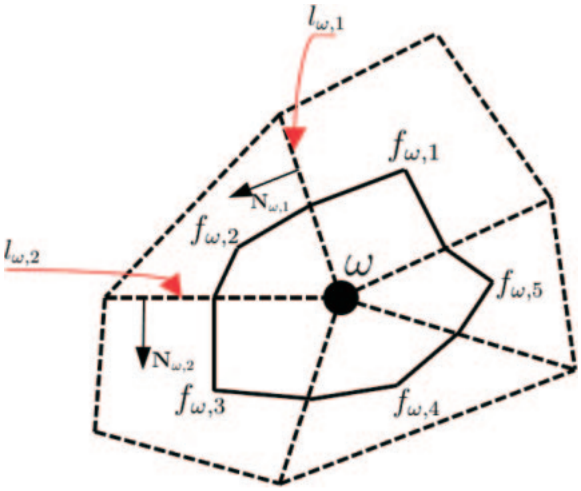

Fig. 3. Template for $GRAD$ ($m_\omega = 5$).

Let's define mesh operator $GRAD\, f : F_\Omega \to F_\omega$:

$$(GRAD\, f)_\omega = \frac{1}{S_\omega} \sum_{i=1}^{m_\omega} f_{\omega,i}\, \chi_{\omega,i},$$

where S_ω is the area of the control volume of ω, m_ω - number of cells adjoint to ω, $f_{\omega,i}$ - value of the scalar mesh function f in surrounding cell i, $\chi_{\omega,i}$ - some vector coefficients.

Then we define scalar products in the spaces of mesh functions F_Ω and F_ω:

$$(f, p)_\Omega = \sum_{\Omega \in \Omega_h} f_\Omega p_\Omega S_\Omega,$$

where the summation is conducted over all cells of the mesh,

$$(\mathbf{u}, \mathbf{v})_\omega = \sum_{\omega \in \omega_h} (\mathbf{u}_\omega, \mathbf{v}_\omega) S_\omega,$$

where the summation is conducted over all internal nodes of the mesh.

Let's choose coefficients χ_i for every internal node $\omega \in \omega_h$, so the following equation were valid:

$$(DIV\,\mathbf{a}, f)_\Omega = -(\mathbf{a}, GRAD\,f)_\omega,$$

which means:

$$\sum_{\Omega\in\Omega_h} f_\Omega \sum_{i=1}^{m_\Omega} \left(\frac{\mathbf{a}_{\Omega,i} + \mathbf{a}_{\Omega,i+1}}{2}, \mathbf{N}_{\Omega,i}\right) l_{\Omega,i} = -\sum_{\omega\in\omega_h} \mathbf{a}_\omega \sum_{i=1}^{m_\omega} f_{\omega,i}\,\chi_{\omega,i}.$$

Switching the order of summation in the left part, we finally get:

$$(GRAD\,f)_\omega = \frac{1}{S_\omega}\sum_{i=1}^{m_\omega} f_{\omega,i}\frac{1}{2}(l_{\omega,i-1}\mathbf{N}_{\omega,i-1} - l_{\omega,i}\mathbf{N}_{\omega,i}), \tag{3.7}$$

see Fig. 3 for the designations of $l_{\omega,i}$ and $\mathbf{N}_{\omega,i}$.

3.2.3 Operators *GRAD* and *grad*

Let's show that:

$$S_\omega(GRAD\,f)_\omega \approx \oint\limits_{\widehat{C}_\omega} f\mathbf{n}\,dl, \tag{3.8}$$

where $\widehat{C}_\omega$ is some contour line around node ω. Let's examine node ω and one of the adjoint to it cells with a scalar value $f_{\omega,i}$ in it (see Fig. 4).

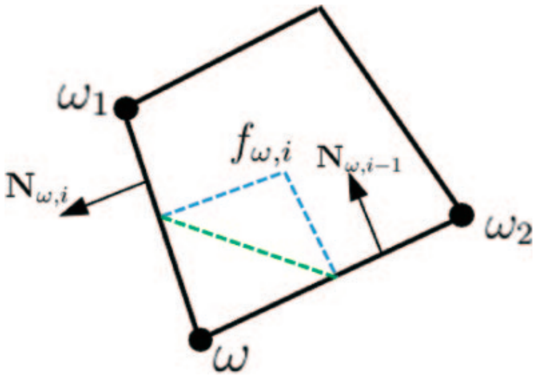

Fig. 4. A node with one adjoint cell.

It is easy to show that vector $(l_{\omega,i-1}\mathbf{N}_{\omega,i-1} - l_{\omega,i}\mathbf{N}_{\omega,i})$ is a normal to the edge, which connects nodes ω_1 and ω_2 and has the same length as this edge. Then $\frac{1}{2}(l_{\omega,i-1}\mathbf{N}_{\omega,i-1} - l_{\omega,i}\mathbf{N}_{\omega,i})$ is a normal to the interval, which connects the middles of edges $\omega\omega_1$ and $\omega\omega_2$, which is directed from node ω and has the same length as this interval, because the interval is the midline of the triangle with $\omega, \omega_1, \omega_2$ as nodes. Then expression (3.8) approximates the contour integral over the polygon around node ω, which is pictured on Fig. 5 with green colour. The expression has the first order of approximation accuracy, because in general mass centres (where values $f_{\omega,i}$ are) of surrounding ω cells are not situated on the edges of the polygon bounded by $\widehat{C}_\omega$. Notice that operator *GRAD* does not approximate differential operator *grad* on a random unstructured mesh, because contour integral in the invariant definition of *grad* (3.5) is approached with error $= O(h)$, and the area in the denominator is $= O(h^2)$.

Due to the relatively small difference between contour lines $\widehat{C}_\omega$ and C_ω (C_ω is the contour line, which bounds the control volume of node ω) we get:

$$S_\omega(GRAD\,f)_\omega = \oint\limits_{C_\omega} f\mathbf{n}\,dl + O(h)$$

Contour lines $\widehat{C}_\omega$ and C_ω are pictured with green and blue colours respectively on Figs. 4 and 5.

3.2.4 Motion Equation

Similarly to the continuity equation (2.3), the motion equation (2.4) must be valid for every subdomain of S. Let's fix some internal node $\omega \in \omega_h$ of the mesh and write down the equation for S_ω - the control volume of ω, and $t = t_n$, $\Delta t = \tau = t_{n+1} - t_n$:

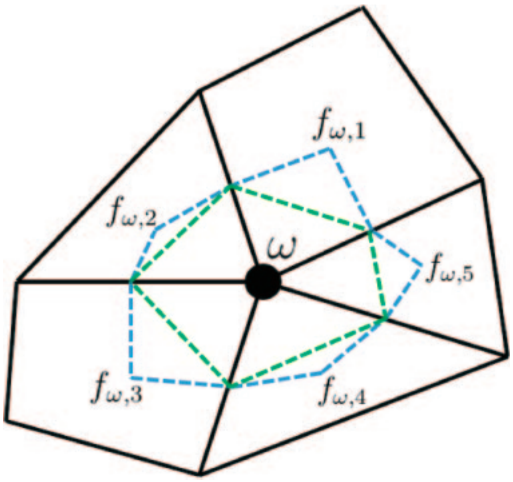

Fig. 5. Contour lines $\widehat{C}_\omega$ and C_ω.

$$\rho \int\limits_{S_\omega} [v_i(\mathbf{r}, t_{n+1}) - v_i(\mathbf{r}, t_n)]\, dS + \rho \int\limits_{t_n}^{t_{n+1}} \oint\limits_{C_\omega} (v_i \mathbf{v}(\mathbf{r}, t'), \mathbf{n})\, dl\, dt' + \int\limits_{t_n}^{t_{n+1}} \oint\limits_{C_\omega} p(\mathbf{r}, t')\, n_i\, dl\, dt'$$

$$= -\mu \int\limits_{t_n}^{t_{n+1}} \oint\limits_{C_\omega} [rot\, \mathbf{v}(\mathbf{r}, t'), \mathbf{n}]_i\, dl\, dt', \quad i = 1, 2. \tag{3.9}$$

At first we approximate integral over S and time integrals in (3.9):

$$\int\limits_{S_\omega} v_i(\mathbf{r}, t_{n+1})\, dS = S_\omega(\mathbf{v}_\omega^{n+1})_i + O(h), \quad \int\limits_{S_\omega} v_i(\mathbf{r}, t_n)\, dS = S_\omega(\mathbf{v}_\omega^n)_i + O(h),$$

$$\int\limits_{t_n}^{t_{n+1}} \oint\limits_{C_\omega} (v_i \mathbf{v}(\mathbf{r}, t'), \mathbf{n})\, dl\, dt' = \tau\sigma_1 \oint\limits_{C_\omega} (v_i \mathbf{v}(\mathbf{r}, t_{n+1}), \mathbf{n})\, dl + \tau(1 - \sigma_1) \oint\limits_{C_\omega} (v_i \mathbf{v}(\mathbf{r}, t_n), \mathbf{n})\, dl + O(\tau^2)$$

$$\int\limits_{t_n}^{t_{n+1}} \oint\limits_{C_\omega} p(\mathbf{r}, t')\, n_i\, dl\, dt' = \tau\sigma_2 \oint\limits_{C_\omega} p(\mathbf{r}, t_{n+1})\, n_i\, dl + \tau(1 - \sigma_2) \oint\limits_{C_\omega} p(\mathbf{r}, t_n)\, n_i\, dl + O(\tau^2)$$

$$\int\limits_{t_n}^{t_{n+1}} \oint\limits_{C_\omega} [rot\, \mathbf{v}(\mathbf{r}, t'), \mathbf{n}]_i\, dl\, dt' = \tau\sigma_3 \oint\limits_{C_\omega} [rot\, \mathbf{v}(\mathbf{r}, t_{n+1}), \mathbf{n}]_i\, dl + \tau(1 - \sigma_3) \oint\limits_{C_\omega} [rot\, \mathbf{v}(\mathbf{r}, t_n), \mathbf{n}]_i\, dl + O(\tau^2),$$

where $0 \leqslant \sigma_1 \leqslant 1$, $0 < \sigma_2 \leqslant 1$, $0 \leqslant \sigma_3 \leqslant 1$ are some weighting coefficients.

Now there are only 3 types of integrals to approximate:

1) $\oint\limits_{C_\omega} f\mathbf{n}\, dl$,

2) $\oint\limits_{C_\omega} (\mathbf{a}, \mathbf{n})\, dl$, (notice that the contour line is different from (3.1))

3) $\oint\limits_{C_\omega} [\mathbf{g}, \mathbf{n}]\, dl$, where $\mathbf{g} = \{0, 0, g\}$ can be considered as a scalar, which belongs to F_Ω.

We also need to approximate differential operator $rot\, \mathbf{a}$, so it could transform mesh functions from F_ω to F_Ω. Notice that the approximation of 1) has already been constructed (3.8):

$$\oint\limits_{C_\omega} f\mathbf{n}\, dl \approx S_\omega(GRAD\, f)_\omega$$

To approach 2) let's define mesh operator $DIV' : F_\omega \to F_\omega$:

$$(DIV'\, \mathbf{a})_\omega = \frac{1}{S_\omega} \sum_{i=1}^{m_\omega} \left[\left(\frac{\mathbf{a}_{\omega,i} + \mathbf{a}_\omega}{2}, \mathbf{N}_{\omega,i}^{\perp,2} \right) l_{\omega,i}^{\perp,2} + \left(\frac{\mathbf{a}_{\omega,i} + \mathbf{a}_\omega}{2}, \mathbf{N}_{\omega,i+1}^{\perp,1} \right) l_{\omega,i+1}^{\perp,1} \right], \tag{3.10}$$

where m_ω is a number of cells adjoint to node ω, $\mathbf{a}_\omega$ - value of mesh function $\mathbf{a}$ in node ω, $\mathbf{a}_{\omega,i}$ - value of mesh function $\mathbf{a}$ in node i of adjoint to ω nodes (two nodes are adjoint if they are endpoints of the same edge), $\mathbf{N}$ - unit normals, l - lenghts (see the designations and numeration on Fig. 6).

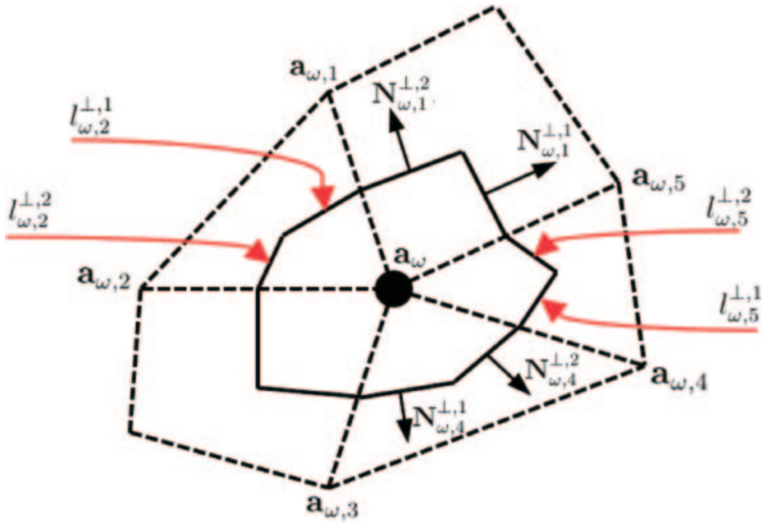

Fig. 6. Templtate for DIV' ($m_\omega = 5$).

Defining DIV' that way can guarantee that:

$$S_\omega (DIV'\,\mathbf{a})_\omega \approx \oint\limits_{C_\omega} (\mathbf{a}, \mathbf{n})\, dl, \tag{3.11}$$

because the left side of (3.10) is the approximation of the contour integral based on the interpolation of vectors **a** over the edges of cells and the rectangle method of numerical integration. Therefore, the approach (3.11) has the first order of approximation accuracy.

Now let's approximate the differential operator *rot*. For that we will need to use the invariant definition of this operator, which can be deducted from the generalized Gauss' theorem [4]:

$$rot\,\mathbf{a} = \lim_{|S'|\to 0} \frac{1}{|S'|} \oint\limits_{\partial S'} [\mathbf{a}, \mathbf{n}] dl, \tag{3.12}$$

where S' is the domain around the point, in which the operator is defined, $|S'|$ - the area of the domain, $\mathbf{n}$ - external unit normal to contour line $\partial S'$.

For every cell $\Omega \in \Omega_h$ we define mesh operator $ROD : F_\omega \to F_\Omega$:

$$(ROD\,\mathbf{a})_\Omega = \frac{1}{S_\Omega} \sum_{i=1}^{m_\Omega} \left(\frac{\mathbf{a}_{\Omega,i} + \mathbf{a}_{\Omega,i+1}}{2}, \mathbf{l}_{\Omega,i} \right), \tag{3.13}$$

where S_Ω is the area of the cell Ω, m_Ω - number of nodes in the cell, $\mathbf{l}_{\Omega,i}$ - vector directed from node i to node $i+1$ of the cell, $\mathbf{a}_{\Omega,i}$ - value of mesh function $\mathbf{a}$ in node i of the cell (see Fig. 7).

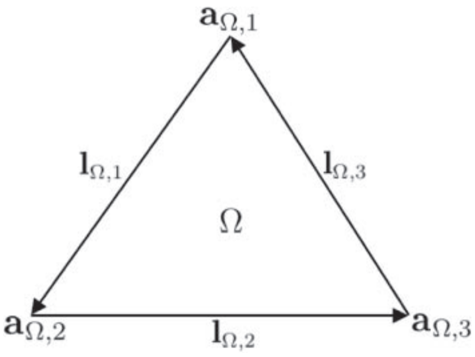

Fig. 7. Template for ROD ($m_\Omega = 3$).

Expression on the right side of (3.13) is the approximation of the third component (it matches the axis, which is orthogonal to the plane containing domain S) of the differential operator $rot\,\mathbf{a}$ in the mass center of cell Ω, because of the statement, which is valid for every two-dimensional vectors $\mathbf{a}$ and $\mathbf{n}$ (if we consider them as vectors of three-dimensional space with zero as third component):

$$[\mathbf{a},\mathbf{n}]_1 = 0,\ [\mathbf{a},\mathbf{n}]_2 = 0,\ [\mathbf{a},\mathbf{n}]_3 dl = (\mathbf{a},\mathbf{dl}),$$

and the approximation of the contour integral in (3.12) is based on the trapezoidal rule. Therefore, $ROD\,\mathbf{a}$ approaches $rot\,\mathbf{a}$ with the first order of approximation accuracy.

Now let's approximate the contour integral in 3). For that we will define mesh operator $ROG : F_\Omega \to F_\omega$:

$$(ROG\,g)_\omega = \frac{1}{S_\omega}\sum_{i=1}^{m_\omega} g_{\omega,i}\{w_{2,i}, -w_{1,i}\}, \tag{3.14}$$

where m_ω - number of cells surrounding node m_ω, $g \in F_\Omega$ is the third (and the only one non-zero) component of some vector, situated in the mass centers of cells surrounding the node,

$$\mathbf{w}_i = \{w_{1,i}, w_{2,i}\} = l^{\perp,1}_{\omega,i}\mathbf{N}^{\perp,1}_{\omega,i} + l^{\perp,2}_{\omega,i}\mathbf{N}^{\perp,2}_{\omega,i},$$

the designations are similar to the designations on Fig. 6.
Then (3.14), multiplied by the area of the control volume of the node, approaches the integral in 3) with the second order of approximation accuracy because it is based on the rectangle method of numerical integration:

$$S_\omega ROG\,g \approx \oint_{C_\omega} [\mathbf{g},\mathbf{n}]\,dl. \tag{3.15}$$

Joining (3.8), (3.11), (3.13) and (3.15), we get the approximation of the motion equation (3.9):

$$\begin{gathered}
\rho S_\omega((\mathbf{v}_\omega^{n+1})_i - (\mathbf{v}_\omega^n)_i) + \sigma_1\tau\rho S_\omega(DIV'(v_i\mathbf{v})^{n+1})_\omega + (1-\sigma_1)\tau\rho S_\omega(DIV'(v_i\mathbf{v})^n)_\omega \\
+ \sigma_2\tau S_\omega((GRAD\,p^{n+1})_\omega)_i + (1-\sigma_2)\tau S_\omega((GRAD\,p^n)_\omega)_i \\
= -\sigma_3\tau\mu S_\omega((ROG\,ROD\,\mathbf{v}^{n+1})_\omega)_i - (1-\sigma_3)\tau\mu S_\omega((ROG\,ROD\,\mathbf{v}^n)_\omega)_i, \qquad (3.16) \\
\forall\omega \in \omega_h,\ i = 1,2, \\
0 \leqslant \sigma_1 \leqslant 1,\ 0 < \sigma_2 \leqslant 1,\ 0 \leqslant \sigma_3 \leqslant 1
\end{gathered}$$

Approximation (3.16) in general has the error $= O(h) + O(\tau)$. Notice that, just like in (3.3), if we sum (3.16) for two adjoining nodes ω_1, ω_2, we get the approximation of (3.9) for the domain $S_{\omega_1} \cup S_{\omega_2}$. Therefore, summing equation (3.3) for all internal nodes of the mesh will give us the equation, which approaches (2.4).

3.3 Numerical Method's Properties

Joining (3.3) and (3.16) we get the approximation of problem (2.3), (2.4) with initial and boundary conditions:

$$\begin{gathered}
S_\Omega(DIV\,\mathbf{v}^n)_\Omega = 0, \quad \forall\Omega \in \Omega_h,\ n = \overline{1,K}, \\
\rho S_\omega((\mathbf{v}_\omega^{n+1})_i - (\mathbf{v}_\omega^n)_i) + \sigma_1\tau\rho S_\omega(DIV'(v_i\mathbf{v})^{n+1})_\omega + (1-\sigma_1)\tau\rho S_\omega(DIV'(v_i\mathbf{v})^n)_\omega \\
+ \sigma_2\tau S_\omega((GRAD\,p^{n+1})_\omega)_i + (1-\sigma_2)\tau S_\omega((GRAD\,p^n)_\omega)_i \\
= -\sigma_3\tau\mu S_\omega((ROG\,ROD\,\mathbf{v}^{n+1})_\omega)_i - (1-\sigma_3)\tau\mu S_\omega((ROG\,ROD\,\mathbf{v}^n)_\omega)_i, \\
\forall\omega \in \omega_h,\ i = 1,2,\ n = \overline{1,K}, \qquad (3.17) \\
\mathbf{v}_\omega = \mathbf{v}_0(\mathbf{r}_\omega),\ \forall\omega \in \omega_h, \\
p_\Omega = p_0(\mathbf{r}_\Omega),\ \forall\Omega \in \Omega_h, \\
\Gamma_\mathbf{v}(\mathbf{v}_\omega) = 0,\ \forall\omega \in \gamma_h, \\
\Gamma_p(p_\Omega) = 0,\ \forall\Omega \in \overline{\Omega}_h,
\end{gathered}$$

where $\mathbf{r}_\omega$ is the radius-vector of node ω, $\mathbf{r}_\Omega$ - radius-vector of the mass center of cell Ω, $\mathbf{v}_0(\mathbf{r}) = \mathbf{v}(\mathbf{r}, t_0)$, $p_0(\mathbf{r}) = p_0(\mathbf{r}, t_0)$ - initial conditions, $\Gamma_\mathbf{v}$ - some approximation of boundary conditions for $\mathbf{v}$, Γ_p - some approximation of boundary conditions for p, $\overline{\Omega}_h$ - set of cells used to implement Γ_p, which consists of extra layers of cells near the boundary. The implementation of Γ_p may be not uniform and depends on the type of the boundary conditions and the type of used mesh.

3.3.1 Method's Conservativity

As it has been noticed before, the fulfilment of (3.3) and (3.16) for every cell and internal node of the mesh results in the satisfaction of the difference analogues of conservation laws (3.1) and (3.9) for every subdomain of S, which consists of various cells and control volumes of the mesh. The satisfaction of these difference laws is called the conservativity of the method. The conservativity means that the method not only approaches the solution of the problem, but also it describes the whole new difference model of fluid - an analogue to the continuous model. This difference model has various properties, such as scheme viscosity and others, which in general are different from their continuous analogues. Such approach allows us to expect that the numerical and analytical solutions will be alike even on coarse meshes.

3.3.2 Connection to the Differential Problem

Let's modify equations (3.3) and (3.16):

$$(DIV\,\mathbf{v}^n)_\Omega = 0, \quad \forall \Omega \in \Omega_h, \tag{3.18}$$

$$\begin{aligned}\rho\frac{(\mathbf{v}_\omega^{n+1})_i - (\mathbf{v}_\omega^n)_i}{\tau} &+ \sigma_1\rho(DIV'(v_i\mathbf{v})^{n+1})_\omega + (1-\sigma_1)\rho(DIV'(v_i\mathbf{v})^n)_\omega \\ &+ \sigma_2((GRAD\,p^{n+1})_\omega)_i + (1-\sigma_2)((GRAD\,p^n)_\omega)_i \\ &= -\sigma_3\mu((ROG\,ROD\,\mathbf{v}^{n+1})_\omega)_i - (1-\sigma_3)\mu((ROG\,ROD\,\mathbf{v}^n)_\omega)_i, \\ &\forall\omega \in \omega_h,\ i = 1,2.\end{aligned} \tag{3.19}$$

Now let's compare (3.18), (3.19) to (2.1), (2.2). Equation $div\,\mathbf{v} = 0$ gives us: $(\mathbf{v},\nabla)\mathbf{v} = \{div(v_1\mathbf{v}), div(v_2\mathbf{v})\}$. Operators *DIV*, *GRAD*, *ROD*, *ROG* were constructed so they could approximate contour integrals in equations (2.3), (2.4) and in invariant definitions of the differential operators. Therefore, if the contour integrals are approached with relatively high order of approximation accuracy (higher than the second order) then the mesh operators will approximate proper differential operators and consequently (3.18), (3.19) will approach (2.1), (2.2). Notice that the constructed method approaches only equation (2.1), because the contour integral in the invariant definition of *grad* is approached only with the first order of approximation accuracy.

3.4 Implementation of the Method

3.4.1 Newton's Method

As you can see in (3.16), if $\sigma_1 \neq 0$ the motion equation's approximation is non-linear with respect to velocity $\mathbf{v}$. It was decided to solve the non-linear system of equations using Newton's iterative method [5].

If one needs to solve the system:

$$\mathbf{f}(\mathbf{x}) = \mathbf{f}(x_1,\ldots,x_n) = 0,$$

where $\mathbf{f} = \{f_1,\ldots,f_n\}, f_i$ are non-linear equations, then Newton's method is described with the formulas:

$$\mathbf{f}'(\mathbf{x}^k)[\mathbf{x}^{k+1} - \mathbf{x}^k] = -\mathbf{f}(\mathbf{x}^k), \tag{3.20}$$

$$\mathbf{f}' = \begin{pmatrix} \dfrac{\partial f_1}{\partial x_1} & \dfrac{\partial f_1}{\partial x_2} & \cdots & \dfrac{\partial f_1}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial f_n}{\partial x_1} & \dfrac{\partial f_n}{\partial x_2} & \cdots & \dfrac{\partial f_n}{\partial x_n} \end{pmatrix},$$

$\mathbf{x}^0$ is some initial point. The iterations continue until some accuracy is achieved.

The method converges if certain conditions are satisfied, including the proximity of the initial point and the solution. Its convergence is quadratic. Notice that in case of linear equations Newton's method requires only one iteration to converge, so we can can formally use Newton's method if $\sigma_1 = 0$. The solution from the previous time step is always taken as the initial point for the iterations on the next time step. That allows us to expect that the method will converge if τ is small enough.

3.4.2 Solving Linear Equations

Newton's method requires us to solve certain system of linear equations (3.20). As operators *DIV*, *GRAD*, *DIV'*, *ROD*, *ROG* have limited templates (template is a set of nodes/cells of the mesh which appear in the operator), the matrix of the system is sparse. If one reasonably enumerates the variables and the equations, the matrix will be banded with a bandwidth q, which depends on the type of used mesh and the form of domain S. Thereby, it was decided to use one of the methods of solving SLAE, which is based on LU-decomposition of the matrix of the system (library Y12M [6]).

4. Computation Results

To test the method several computations were conducted, including the case when fluid flows between two plane-parallel plates (see the domain of the problem and boundary conditions on Fig. 8).

The author used rectangular meshes with cells of sizes $h_x \times h_y$ to discretize the domain S. Notice that system (2.3), (2.4) in this domain has an analytical solution, which represents stationary laminar flow of the fluid:

$$\mathbf{v} = \{v, 0\}, \quad v = v(y) = \frac{\Delta p}{2\mu L}y(2R-y), \quad p = p(x) = p_L - \frac{\Delta p}{L}x, \tag{4.1}$$

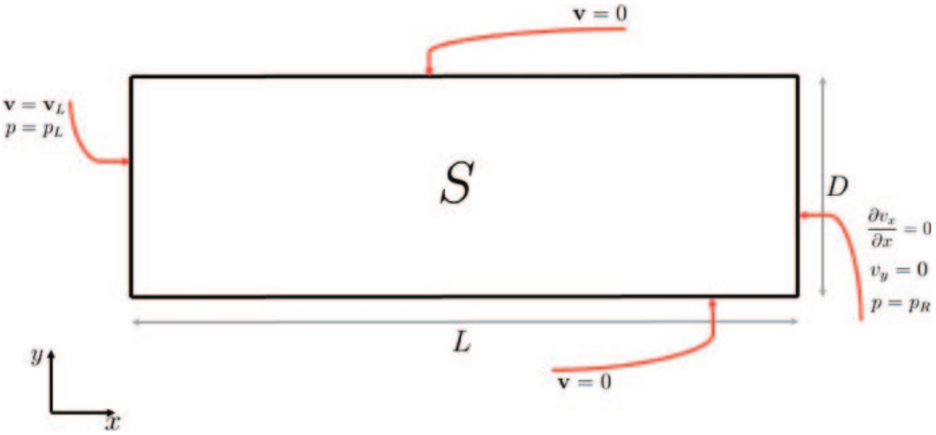

Fig. 8. Domain of the problem.

where p_L - pressure on the left boundary, Δp is pressure difference between the left and the right boundaries of the domain, L - length of the plates (according to the x-axis), $R = \frac{D}{2}$ - half of the distance between the plates.
Let's also define Reynolds number:

$$Re = \frac{v_0 D_0 \rho_0}{\mu},$$

where v_0, D_0, ρ_0 are typical dimension scales of the problem. It is known that for every type of fluid flow (flow between parallel plates, flow over ellipsoid, etc.) some value Re_{cr} exists. If $Re \ll Re_{cr}$ then the flow is laminar, if $Re \gg Re_{cr}$ than the flow can become turbulent [7].

From now on the fluid has density $\rho = 1$, numerical solution is obtained with the method (3.17) with $\sigma_1 = \sigma_2 = \sigma_3 = 1$.

Designations for all following plots: brown colour - analytical solution (4.1) for $\mathbf{v}$ (v_x in particular, because $v_y = 0$); purple colour - numerical solution for v_x; orange colour - numerical solution for v_y; green colour - numerical solution for p. On all following plots for pressure horizontal indexes of the cells act as an x-axis.

4.1 Boundary and Initial Conditions Corresponding to Analytical Solution

Let $L = 2.5$, $D = 0.6$, $h_x = 0.1$, $h_y = 0.1$, $\tau = 0.01$. Let's set pressure values p_L and p_R on the left and right boundaries and corresponding to $\Delta p = p_L - p_R$ velocity (4.1) on the left boundary and as initial condition in domain S. Then the analytical solution of (2.3), (2.4) is exactly stationary flow (4.1). Here are the results of computations for several sets of parameters:
1) $\mu = 10^{-2}$, $p_L = 1$, $p_R = 0$, $Re = 100$ (see Figs. 9 and 10)
2) $\mu = 10^{-4}$, $p_L = 10^{-2}$, $p_R = 0$, $Re = 10000$
In both cases parameters are chosen so the analytical solutions for velocity in 1) and 2) were identical.

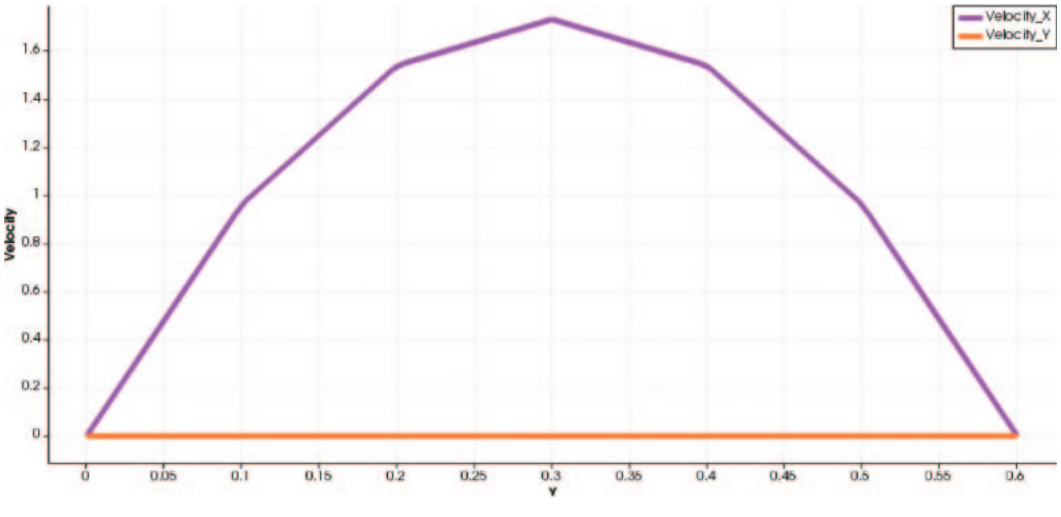

Fig. 9. Sustained velocity in case 1).

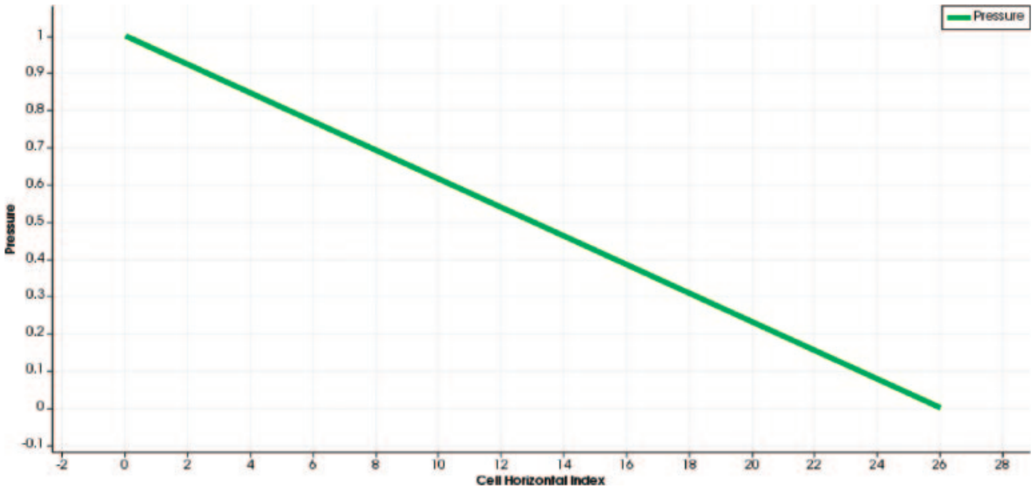

Fig. 10. Sustained pressure in case 1).

In case 1) numerical solution matches the laminar flow in which velocity doesn't depend on x, pressure is linear with respect to x. Values for velocity $\mathbf{v}$ insignificantly fluctuate near the analytical solution. Figure 9 shows velocity components on layer $x = 1.2$, $t = 500\tau$; Fig. 10 shows pressure on layer $y = 0.35$, $t = 500\tau$. Computational error at the moment $t = 500\tau$ in this case doesn't exceed 10^{-5}.

Case 2) is almost similar to case 1) apart from the computational errors, which at the moment $t = 500\tau$ do not exceed 10^{-2}.

As we see in results 1) and 2), increase of Reynolds number Re is followed by decrease in accuracy.

4.2 Boundary Conditions Corresponding to Analytical Solution and Zero Initial Conditions

Let $L = 2.5$, $D = 0.6$, $h_x = 0.1$, $h_y = 0.1$, $\tau = 0.01$. Let's set pressure values p_L and p_R on the left and right boundaries and corresponding to $\Delta p = p_L - p_R$ velocity (4.1) on the left boundary. Inside domain S we set zero initial conditions for velocity and p_R for pressure. Thereby, we have discontinuous initial conditions. Let's see if the numerical solution can give sustained flow (4.1) at various Re values.

1) $\mu = 1$, $p_L = 100$, $p_R = 0$, $Re = 1$ (see Figs. 11, 12, and 13)

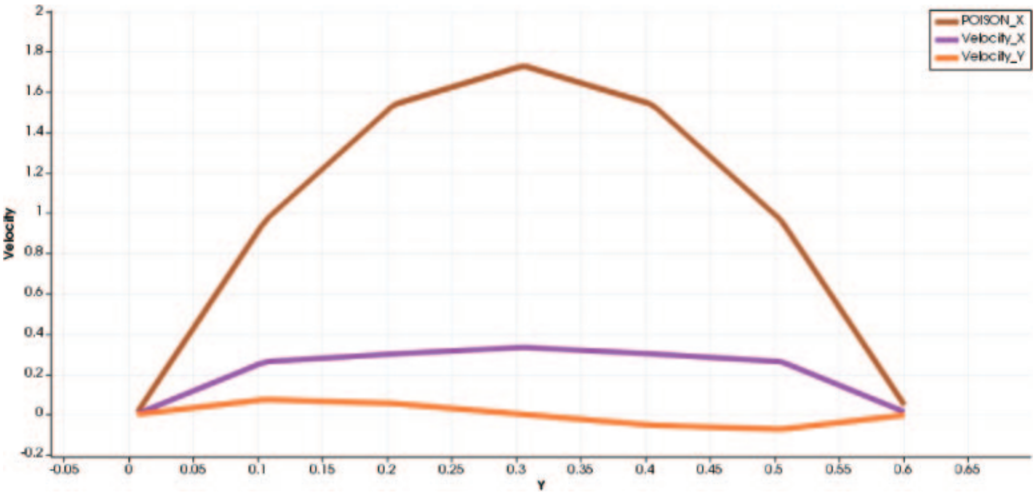

Fig. 11. Velocity at $t = \tau$ near the left boundary in case 1).

As you can see on Fig. 11, the current near the left border (layer $x = 0.1$) has non-zero y-component of velocity after the first time step. On other layers the flow is laminar. The pressure becomes linear with respect to x right after the first step. After the second step the flow in all domain becomes laminar and converges fast to the analytical solution (at $t = 25\tau$ error ≤ 0.05, at $t = 50\tau$ error $\leq 10^{-5}$).

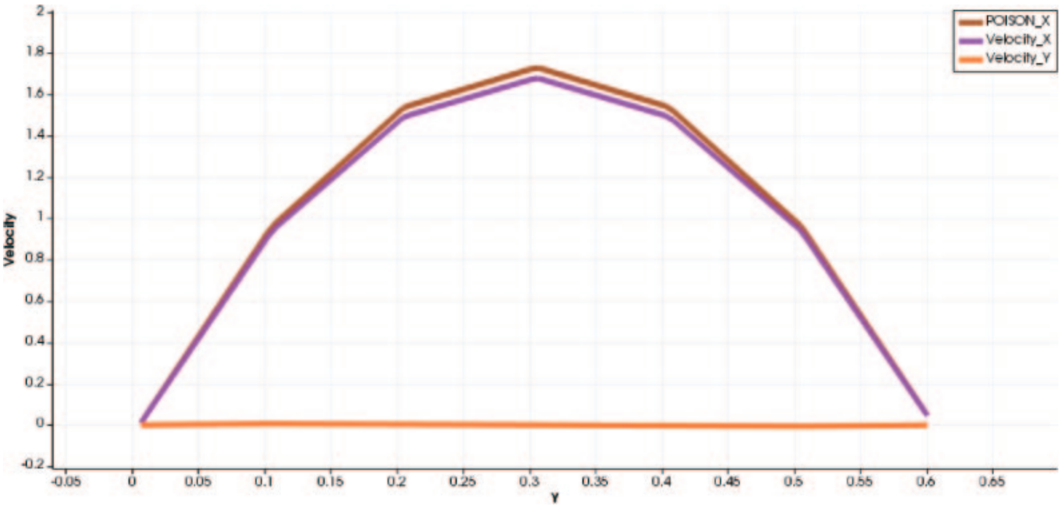

Fig. 12. Velocity at $t = 15\tau$ near the left boundary in case 1).

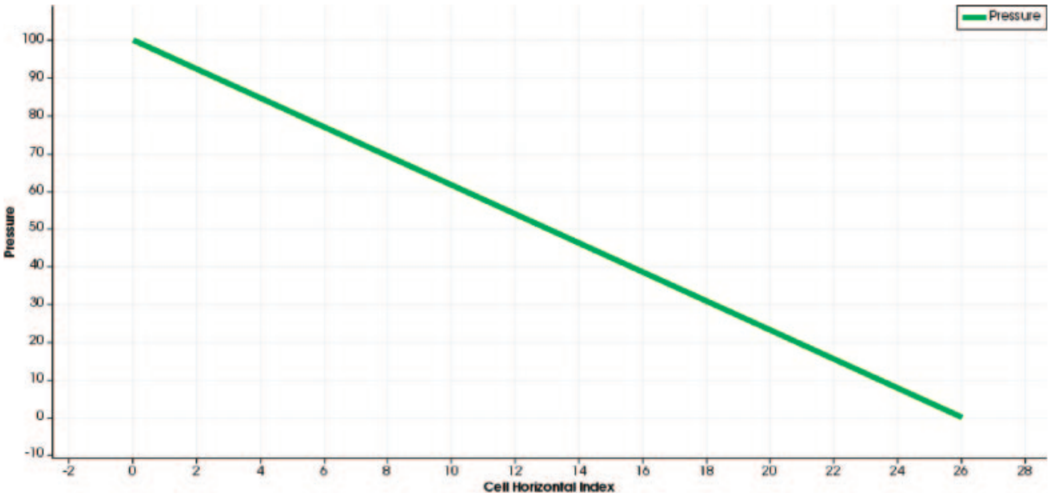

Fig. 13. Sustained pressure in case 1).

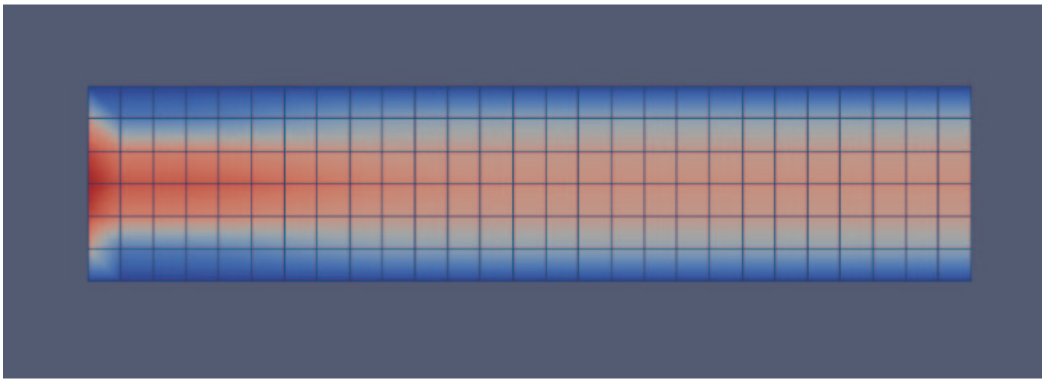

Fig. 14. Velocity magnitude at $t = 150\tau$ in case 2).

2) $\mu = 10^{-2}$, $p_L = 1$, $p_R = 0$, $Re = 100$ (see Figs. 14–17)

In case 2) y-components of velocity near the left boundary are non-zero for much longer than in case 1): about 150τ against τ. On Figs. 14, 15, 16 we have magnitude of velocity, velocity near the left boundary (layer $x = 0.1$) and pressure (layer $y = 0.35$) accordingly at the moment $t = 150\tau$. The flow at that moment still has relatively small non-zero y-components of velocity and a jump in pressure, which is caused by the discontinuity of initial conditions. After getting rid of y-components of velocity the numerical solution starts to converge relatively (for that value of Re)

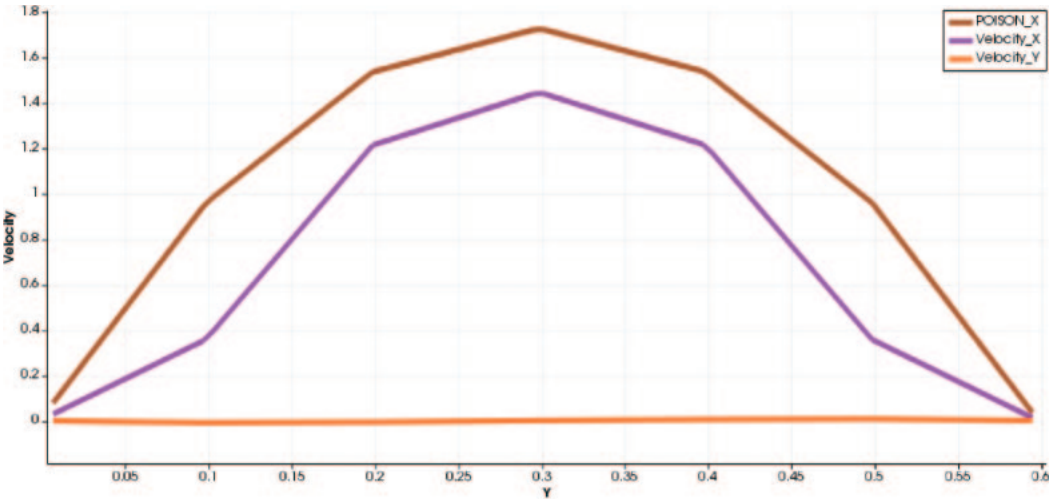

Fig. 15. Velocity at $t = 150\tau$ near the left boundary in case 2).

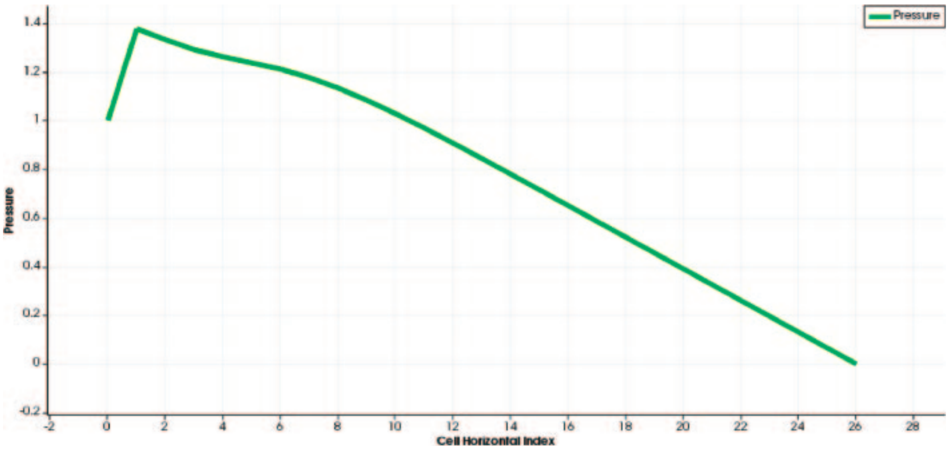

Fig. 16. Pressure at $t = 150\tau$ in case 2).

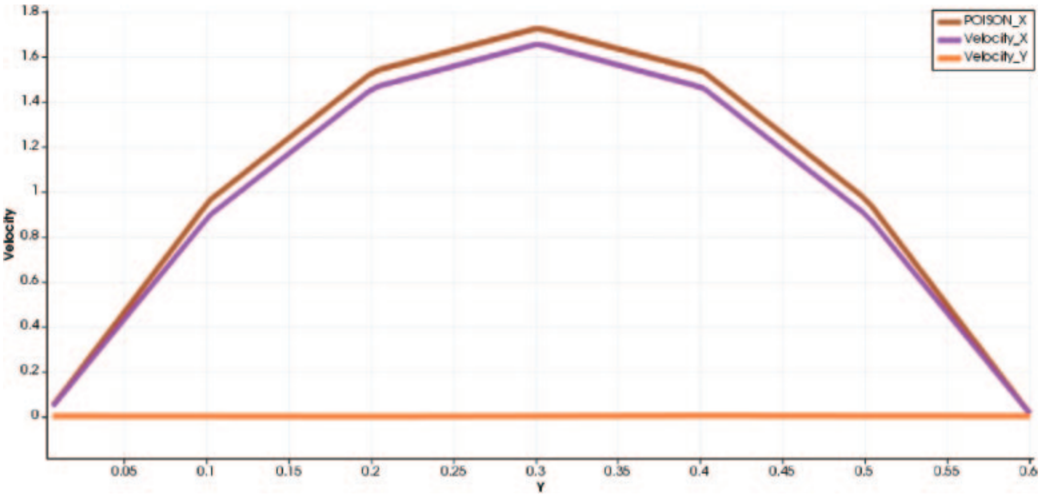

Fig. 17. Velocity at $t = 350\tau$ near the left boundary in case 2).

fast to the analytical solution. By the moment $t = 350\tau$ the current becomes laminar, pressure becomes linear and the error ≤ 0.06. By the moment $t = 1000\tau$ the error $\leq 10^{-3}$.

3) $\mu = 10^{-3}$, $p_L = 0.2$, $p_R = 0$, $Re = 1000$ (see Figs. 18–21). In this case the computations were conducted for the domain, which is twice longer than in previous tests ($L = 5$).

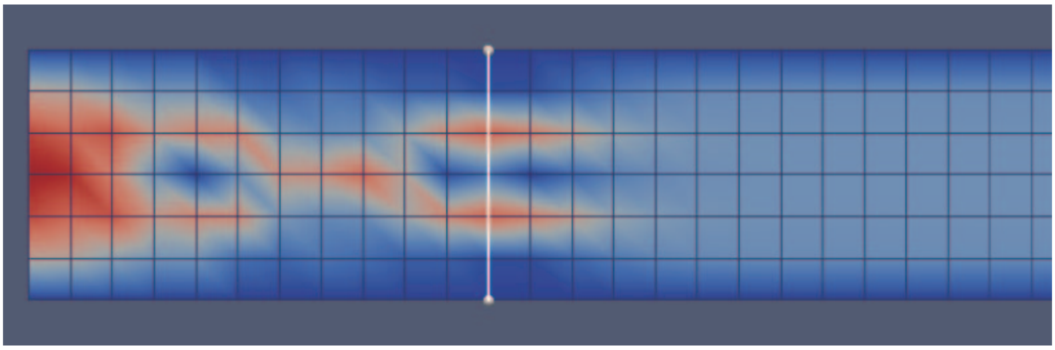

Fig. 18. Velocity magnitude at $t = 175\tau$ in case 3).

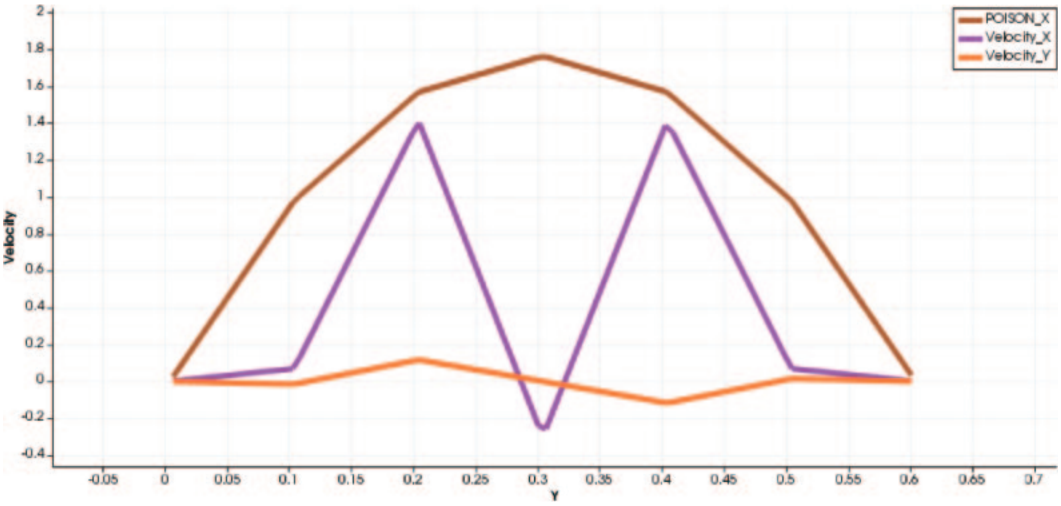

Fig. 19. Velocity at $t = 175\tau$ in case 3) (see layer on Fig. 18).

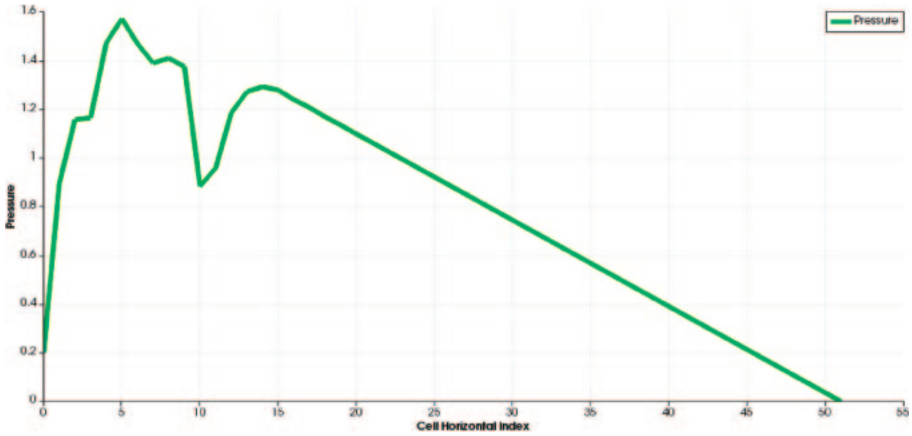

Fig. 20. Pressure at $t = 175\tau$ in case 3).

In case 3) the situation is different from cases 1) and 2). Until $t \approx 700\tau$ there are vortices which are moving from the left to the right boundary (see Figs. 18 and 19). These vortices have a noticeable impact on pressure (there are certain waves on pressure's plot, which relate to the passing vortices; see Fig. 20). At $t \approx 700\tau$ the current becomes laminar and starts to converge to the analytical solution (see Fig. 21). At $t = 850\tau$ the error ≤ 0.08 and it continues to decrease.

As a result we can state that the larger Reynolds number Re is, the more noticeable vortices appear near the left boundary and it takes more time for the method to smooth over the solution and converge to the analytical solution. Other similar tests showed that the vortices, which do not disappear near the left boundary and spread to the right, start to appear approximately at $Re \geq 200$.

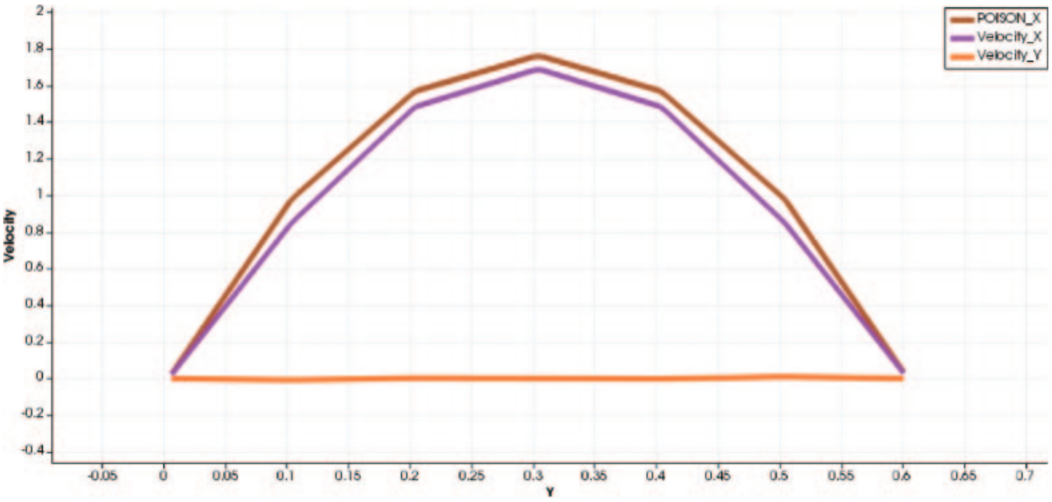

Fig. 21. Velocity at $t = 850\tau$ in case 3) (see layer on Fig. 18).

Notice that if the sustained flow from 1) or 2) (result of the scheme relatively close to the analytical solution) is taken as new boundary and initial conditions, then the numerical solution of such problem will be stationary and will approximate the analytical solution (4.1).

5. Conclusions

The whole family of conservative methods for the solution of two-dimensional Navier–Stokes equations for viscous incompressible fluids was constructed. The tests showed that the numerical solutions for some problems converge to the analytical solutions on continuous and non-continuous initial conditions. Also such properties as laminarity of the flow when $Re \ll \overline{Re_{cr}}$ are valid for the numerical solutions, and some turbulences might occur if $Re \gg \overline{Re_{cr}}$, where $\overline{Re_{cr}}$ is some property of the method, which in general is not equal to the analytical Re_{cr}. The author plans to continue testing the constructed methods on more complex domains and he also hopes to try the methods on other types of meshes, triangular or hybrid.

Acknowledgments

The author would like to thank Prof. S. Mukhin for his assistance in preparing this work.

REFERENCES

[1] Samarski, A., Tishkin, V., Favorski, A., and Shashkov, M., "Difference operator schemes," *Differential Equations*, **17**: 1317–1328 (1981).

[2] Samarski, A., Koldoba, A., Poveshenko, Y., Tishkin, V., and Favorski, A., Difference Schemes on Unstructured Meshes, Criteriy Pub. Co. (1996).

[3] Goloviznin, V., Samarski, A., and Favorski, A., "Variational approach to the construction of balanced discrete models in multidimensional hydrodynamics," *Keldysh Institute of Applied Mathematics Preprints*, **140**: 1–31 (1980).

[4] Karpov, V., Favorski, A., and Khrulenko, A., Vector and Tensor Models, Max Press Pub. Co. (2008).

[5] Samarski, A., and Gulin, A., Numerical Methods, Nauka Pub. Co. (1989).

[6] Osterbru, O., and Zlatev, Z., Exact Methods for Sparse Matrices, Nauka Pub. Co. (1989).

[7] Landau, L., and Lifshitz, E., Hydrodynamics, Nauka Pub. Co. (1986).