On the mixed approximation type pressure correction method for incompressible Navier-Stokes equations

Nikolay M. Evstigneev, Oleg I. Ryabkov
Federal Research Center "Informatics and Control", Institute for System Analysis, Russian Academy of Science, 117312, Moscow, pr. 60-letiya Oktyabrya, 9, Russia.
E-mail: evstigneevm@yandex.ru

Abstract. In the paper we consider a numerical method for the solution of the Navier–Stokes equations. We perform pressure velocity coupling in the mixed approximation type. We construct the approximation using finite element method for the pressure and finite volume/difference method for velocity variables. WENO scheme is used for velocity advection. The method is stable under common CFL criterion for advection equations. Then we demonstrate the correctness of velocity–pressure coupling for the Stokes flow. Finally the method is benchmarked against reference analytical and numerical solutions of the Navier–Stokes equations. It is shown that the method has high accuracy and can be used to perform direct numerical simulation of incompressible fluid flow.

1. Introduction
We are considering 2D or 3D incompressible Navier–Stokes equations, optionally, with external forcing:

\begin{align*}
  \mathbf{u}_t + (\mathbf{u} \cdot \nabla) \mathbf{u} + \nabla p/\rho &= \mu \Delta \mathbf{u} + \mathbf{f}, \\
  \nabla \cdot \mathbf{u} &= 0.
\end{align*}

Here \( \mathbf{u} \) is the vector–function of velocity, \( \mathbf{f} \) is the vector–function of external forcing, \( p \) is the scalar function of pressure and \( \rho \) is the constant density. Equation (1a) described the conservation of momentum and (1b) governs the conservation of mass. The latter equation reduces to forcing solenoidal filed on velocity vector–function. These equations are considered in a piecewise–continuous domain \( \Omega \). The initial conditions are posed on \( \mathbf{u} \) in \( \Omega \) at \( t = 0 \), while the boundary conditions can be gives as defining velocity components, its derivatives or given stress vector components. The pressure is determined up to a constant, but can be uniquely determined by setting the value at a spatial point or setting pressure gauge, e.g. \( \int_{\Omega} p \, d\mathbf{x} = 0 \).

Numerical simulation of the given initial boundary–value problem (IBVP) for (1) on a fine grid (with the resolution to solve all distinguishable flow scales) is called Direct Numerical Simulation (DNS). We also assume, that for a correct IBVP there exists a smooth solution of (1).

Let us introduce some discretization of temporal operator (with step \( \Delta t \)) and rewrite (1) as:

\begin{align*}
  \mathbf{u}^{l+1} &= \mathbf{u}^l - \Delta t \left( \mathbf{u}^{l+\delta} \cdot \nabla \right) \mathbf{u}^{l+\delta} - \Delta t \nabla p^{l+\eta}/\rho + \Delta t \mu \Delta \mathbf{u}^{l+\gamma} + \Delta t \mathbf{f}, \\
  \nabla \cdot \mathbf{u}^{l+1} &= 0.
\end{align*}
Parameters $\theta, \eta, \gamma, \delta$ are either 0 or 1 and define different level of implicitness. In case one wants to use fully implicit scheme one chooses $\theta = 1, \eta = 1, \gamma = 1, \delta = 1$ and uses Picard or Newton methods.

Two difficulties exist in finding the numerical solution for the Navier–Stokes equations. First is the nonlinear term that results in oscillatory behaviour of common finite difference or finite element methods for high velocity flows. To remedy this problem there are many methods exist, see [1] for example. The second difficulty is related to pressure–velocity coupling. Observe, that Navier–Stokes equations are not of Cauchy–Kovalevskaya type, so there is no direct way to find pressure from the system (2). It is the main topic of this work for a particular type pf approximation. This problem is also discussed in numerous papers, we only give a very brief review and refer readers for more detailed ones in [2, 3, 4, 5].

Let us now introduce some spatial discretization of the problem (2) with parameter $h$ (say, maximum mesh spacing size). One can apply finite element, finite volume or finite difference discretization. Then we can reformulate (2) in thee matrix form as:

$$\begin{pmatrix} A & Q \\ G & 0 \end{pmatrix} \begin{pmatrix} \hat{u} \\ \hat{p} \end{pmatrix} = \begin{pmatrix} g \\ 0 \end{pmatrix}, \quad (3)$$

where operator $A$ includes all implicit terms (or becomes identity matrix $E$ for fully explicit method), operator $Q$ is the gradient, $G$ is the divergence vector $g$ includes forcing, previous timestep value of $u$ and all explicit terms, vector $\hat{p}$ is the vector of discrete pressure and vector $\hat{u}$ is the vector of discrete velocity. One must solve discrete saddle point problem (3) in order to find the solution of (2) for every timestep. The system is solved in two steps. Observe, that the velocity can be extracted by inverting $A$:

$$\hat{u} = A^{-1}(g - Q\hat{p}). \quad (4)$$

Then, in the first step, the pressure from the continuity equation is found:

$$GA^{-1}Q\hat{p} = GA^{-1}g. \quad (5)$$

On the second step the velocity is adjusted by (4), now respecting continuity equation. We now observe two problems. First, matrix $A$ should be non–singular. This is usually true for all correct approximations. Next, we observe that in order to find $\hat{p}$, matrix $S := GA^{-1}Q$ must be non–singular. Even more, as $h \rightarrow 0$ the operator norms of these inverse matrices must be bounded, i.e. $\|A^{-1}\|_A \leq M_0 < +\infty$, $\|S^{-1}\|_S \leq M_1 < +\infty$. This is related to the famous $inf$–$sup$ condition by Ladyzhenskaya–Babuska–Brezzi (LBB):

$$\exists \gamma > 0, \text{ s.t.: } \inf_{\hat{p} \neq 0} \sup_{\hat{v} \neq 0} \frac{\int_{\Omega} \hat{p} \nabla \cdot \hat{v}}{\|\hat{v}\|_{H^1}, \|\hat{p}\|_{L^2}} \geq \gamma,$$

with $\gamma$ is independent on numerical parameters $h$ or $\Delta t$. If the latter is true, then for sufficiently smooth solutions of (1) one can conclude [2] convergence of numerical solution as:

$$\|u - \hat{u}\|_{H^1} + \|p - \hat{p}\|_{L^2} \leq C \left( h^k \|u\|_{H^{k+1}} + h^{l+1} \|p\|_{H^{l+1}} \right), \quad (7)$$

with $C$ is a constant not dependant on $h$ or $\Delta t$.

In order to satisfy (6), one must choose proper discretization for spatial operators. The easiest choice is the staggered finite difference approximation where pressure is found in the center of a cell and velocities are found in a cell’s edges, introduced in the MAC method [9] . Different appropriate finite element approximations are discussed in [6]. Then one finds the solution of
the problem in coupled way (equations (4) and (5) simultaneously) or in segregated way. The first way to solve the problem is by applying Uzawa–type method to the solution of (3). We refer reader to [6, 7] and other publications. This method is beneficial in terms of accuracy because it produces divergence free field by a single solution. But it is very difficult to formulate in parallel computational architecture which is needed for DNS computations since resulting block matrix (3) is badly conditioned and requires complex preconditioners for iterative solvers. The segregated strategy is less accurate in terms of divergence field correction process but is more easily implemented in modern parallel computers. We refer the reader to [4, 8] for overviews.

We, therefore use segregated method to solve system (2) without explicitly using LBB condition. In this work we are interested in using efficient approximation for Graphical Processing Units (GPUs). So we adopt high order finite difference methods for velocity approximation and use finite element approximation to reconstruct pressure filed that can be efficiently solved by Geometric Multigrid method. We use WENO [16] schemes in order to stabilize nonlinear term and apply compact finite differences [19] to linear operators. The idea of using WENO scheme for incompressible flows is not new and is discussed in [10, 11]. But we found no paper related to the problem of pressure–velocity coupling for such systems with WENO–type approximation. The method was designed in 2008 and used in many applications. We already discussed this approach shortly in many papers, e.g. [12], where focus was shifted towards the application problem. But in this paper it is discussed in detail.

The paper is laid out as follows. First we give details on the pressure correction method we use. We formulate the method for the Stokes and Navier–Stokes problems. Then we describe the setting of boundary conditions and perform test on the velocity–pressure correction for the Stokes problem in 2D. Next we benchmark the method against some problems that are either well studied or have analytical solution.

All simulations are performed on the author’s multiple GPU cluster, consisting of 6 GPUs assembled on one chassis. For the computational efficiency of the implementation the reader is referred to [12]. All 2D and 3D visualization is done in GMSH software [13] and MATLAB.

2. Pressure correction method

We start with discretization of Stokes operator mostly following [14] with slight modifications, reported in [7] (from here we assume, that \( \rho = 1 \) for simplicity):

\[
\begin{align*}
\partial_t \mathbf{u} + \nabla p &= \mu \Delta \mathbf{u} + \mathbf{f}, \\
\nabla \cdot \mathbf{u} &= 0,
\end{align*}
\]

in the bounded piecewise continuous domain \( \Omega \). We use some discretization method that we discuss later to translate (8) into:

\[
\begin{align*}
M \mathbf{u}_l &= -\nabla p + A \mathbf{u} + \mathbf{f}, \\
G \mathbf{u} &= 0.
\end{align*}
\]

Here \( M \) is a mass matrix, \( Q \) is a gradient matrix, \( A \) is a diffusion matrix and \( G \) is a divergence matrix. We introduce time slices with \( (\cdot)^l \) being \( l \)-the time slice with time-step \( \Delta t \) and derive the following system:

\[
\begin{align*}
(M - \Delta t A) \mathbf{u}^{l+1} + \Delta t Q p^{l+1} &= M \mathbf{u}^l + \Delta t \mathbf{f}^l, \\
G \mathbf{u}^{l+1} &= 0.
\end{align*}
\]

Since \( p^{l+1} \) is unknown on \( l \)-th time slice, we split the system as:

\[
M \mathbf{u}^* = M \mathbf{u}^l + \Delta t \mathbf{f}^l - \beta \Delta t Q p^l,
\]

(11)
and introduce velocity correction vector \( \mathbf{u}^C \) and scalar potential function \( \phi \):

\[
\mathbf{u}^* + \mathbf{u}^C = \mathbf{u}^{t+1},
\]

hence:

\[
\begin{align*}
\mathbf{u}^C &= \mathbf{u}^{t+1} - \mathbf{u}^*, \\
\mathbf{G} \mathbf{u}^C &= \mathbf{G} \mathbf{u}^*,
\end{align*}
\]

so \( \mathbf{u}^C = -\Delta t \mathbf{N}^{-1} q \phi \), and we get correction equation for the potential function \( \phi \):

\[
\mathbf{G} \mathbf{N}^{-1} q \phi = -1/\Delta t \mathbf{G} \mathbf{u}^*,
\]

where matrix \( \mathbf{S} := \mathbf{G} \mathbf{N}^{-1} \mathbf{Q} \) is called a Schur complement of the pressure [6]. After the solution of (12) we correct velocity and pressure functions in such way, that \( \mathbf{G} \mathbf{u}^{t+1} = 0 \):

\[
\begin{align*}
\mathbf{u}^{t+1} &= \mathbf{u}^* - \Delta t \mathbf{N}^{-1} q \phi, \\
p^{t+1} &= \beta p^t + \phi,
\end{align*}
\]

where \( 0 \leq \beta \leq 1 \) is a parameter. The whole solution consists of using three steps (11), (12), (13). In general this results in first order of approximation for (8) in time. The stability and consistence of the method depend on the discretization. The discretization must obey LBB condition in order to be stable and consistent. In case of the provided approximation it means, first, that \( \ker(\mathbf{G}) = \{0\} \), so that \( \mathbf{G} \mathbf{p} = 0 \) only for \( p = 0 \). Notice, that a hydrostatic pressure mode (i.e. \( p = \text{const on } \Omega \)) is not included in the verification, see [6]. Second, that \( \| (\mathbf{S}_h) \|_{S}^{-1} \leq C < +\infty, \ h \to 0 \), where \( h \) is a discretization parameter (e.g. minimal grid or element size). In general one usually takes \( \mathbf{G} := \mathbf{Q}^T \), chooses mixed finite element approximation (for finite element methods) or staggered grid (for finite difference approximation). Here we use different strategy that is dealing with different approximations for different operators, i.e. we use the altered form of correction equation: \( \mathbf{L} \phi = -1/\Delta t \mathbf{G} \mathbf{u}^* \), where \( \mathbf{L} \) is the approximation of the \( \mathbf{S} \) operator.

Let us introduce rectangular hexahedron \( \mathbf{W}_{jkl} \) that form a 3D (or 2D in case of two dimensions) tessellation of a rectangular domain \( \Omega = \bigcup \mathbf{W}_j \), such that \( \mathbf{W}_j \cap \mathbf{W}_k = \emptyset, j \neq k \), where \( j \) is a multi index with \( j \) being a center of \( \mathbf{W}_j \). We introduce another set of tessellation \( \mathbf{U}_k \) that is constructed from swapping central nodes and vertices, thus each vertex of \( \mathbf{W}_j \) becomes a center for \( \mathbf{U}_k \) and vice versa. This strategy forms staggered Arakawa B–grid [15].

We define basis functions in an element \( \mathbf{Q}_m \), where \( \mathbf{Q} \) can be \( \mathbf{W} \) or \( \mathbf{U} \), as follows:

\[
\psi_m(x) = \begin{cases} 
c_j, k, l(x), & j \in \mathbf{Q}_m, \\
0, & j \notin \mathbf{Q}_m,
\end{cases}
\]

where \( c_{j, k, l} \) is the trilinear function of a rectangular element. Let coordinates of an element be transformed as \( \xi_m = x_1/h_m \), where \( m = 1, 2, 3 \) and \( h_m \) is the length of an element in the direction \( m \). Then the reference function for a canonical element \( \bar{c} \) is defined as:

\[
\bar{c}_{v_1, v_2, v_3}(\xi_m) = \prod_{n=1}^{3} (v_n \xi_n + (1 - v_n)(1 - \xi_n)),
\]

where \( v_m = \{0, 1\} \) are functions of an element vertices, that map global indexes \( j, k, l \) into local indexes.

The following expansion in the element space for a scalar function \( P(x) \) is used:

\[
P(x) = \sum_{j \in \{Q\}} P(j)\psi_j(x),
\]
Now we consider set of points $j$ formed by the centers of $W_j$ or vertexes of $U_k$. Such elements can be considered as finite volumes, i.e. $\mathcal{F}_j = \int_{W_j} P(x)dx$. We now define the following differential operators: $Q$ and $S$ in space of nodal finite elements $W$, $A$ and $G$ in space of finite volumes $W$, we use Laplace operator $L$ for the approximation of $S$ and identity matrix for $M$. Define projection of central finite difference/volume operator to nodes as $U$. This operation is performed using (15) with $Q := W$. The gradient of the scalar function is found by the direct calculation of the gradient function in the center of an element $W_j$. In this case the scheme can be written as follows:

$$
\begin{cases}
    \mathbf{u}^* = \mathbf{u}_t + \Delta t \mathbf{u}^* + \Delta t \mathbf{A} \mathbf{u}^* - \beta \Delta t Q \mathbf{p}^t, \\
    L\phi = -1/\Delta t U Gu^*, \\
    \mathbf{u}^{t+1} = \mathbf{u}^* - \Delta t Q \phi, \\
    p^{t+1} = \beta p^t + \phi.
\end{cases}
$$

(16)

In this work we use compact finite difference scheme of 4-th order to approximate $A$ and $G$ using method of alternating directions. The approximation of $L$ and $Q$ is done using Bubnov – Galerkin projection (choosing test function from the same space of finite elements), e.g. for the second equation in (16):

$$
\int_{\Omega} \Delta \phi(x) \psi_j(x)dx = -1/\Delta t \int_{\Omega} U Gu^* \psi_j(x)dx.
$$

(17)

Inserting (15) into (17) and doing integration by parts:

$$
\phi_j \int_{\Omega} \nabla \psi(x) \nabla \psi_k(x)dx = g_{j,0} \int_{\partial \Omega_{x,0}} \psi_j \psi_k dS + g_{j,1} \int_{\partial \Omega_{x,1}} \psi_j (\nabla \psi_k, n_r) dS + 1/\Delta t G(u)_j \int_{\partial \Omega} \psi_j \psi_k dx,
$$

(18)

where $g_{j,0}$, $g_{j,1}$ are coefficients of expansion for Dirichlet and Neumann boundary conditions and $G(u)_j$ are coefficients of divergence operator projection into the space of finite elements.

The gradient operator $Q\phi$ for a scalar function $\phi$ at a central node $k_c$ that has coordinates $x_c$ is defined as:

$$
Q\phi_{k_c} := \sum_{j \in \{Q\}} P(j) \nabla \psi_j(x_c).
$$

(19)

It is a straightforward way to check the BBL condition. Trivial kernel of $Q$ is proved by considering space of finite elements. The condition number of finite element approximation can be estimated from the space of finite elements and can be shown that it has a marginal bound. The norm of the inverse Laplace operator $||L^{-1}||_L \sim O(1)$ as $h \to 0$ due to the approximation properties of nodal Finite Elements [6].

Now we return to Navier-Stokes equations by applying some approximation for the nonlinear term in (16) (defined bellow as $B$). We use fifth, seventh or nine-th order WENO schemes that have good spectral properties and guarantee TVB behavior of the solution (on each WENO stage we use Runge–Kutta 3rd order SSP method [17]), for more information see [1, 16]. Besided, WENO scheme can coupe with high gradients arising in the flow near sharp corners. This relaxes demands on mesh refinement. In order to increase the temporal accuracy we also use Runge–Kutta 3rd order explicit method [17] for which the projection step (16) is applied on every stage. The stability of the method is deduced from the Courant–Friedrichs–Lewy condition as recommended in [16] for WENO schemes, since diffusion is considered implicitly and other steps are unconditionally stable.

Then the total scheme for (2) is given as:

$$
\begin{cases}
    \left( \mathbf{u}^{t+1}, p^{t+1} \right) = \mathcal{A}(\mathbf{u}^t, p^t) = \\
    \left\{
    \begin{align*}
    \mathbf{u}^* &= \mathbf{u}^t + \Delta t \mathbf{A} \mathbf{u}^* - \beta \Delta t Q \mathbf{p}^t - \Delta t B(\mathbf{u}^t, \mathbf{u}^t, \Delta t), \\
    L\phi &= -1/\Delta t U Gu^*, \\
    \mathbf{u}^{t+1} &= \mathbf{u}^* - \Delta t Q \phi, \\
    p^{t+1} &= \beta p^t + \phi.
    \end{align*}
\right. 
\end{cases}
$$

(20)
\[
\begin{pmatrix}
(u_{n+1}^1 \\
p_{n+1}^1)
\end{pmatrix} = \mathcal{RK}(u^n, p^n) = \begin{cases}
(u^1) = A(u^n, p^n), \\
\left(\begin{array}{c}
u^2 \\
p^n
\end{array}\right) = \frac{3}{4} \left(\begin{array}{c}
u^n \\
p^n
\end{array}\right) + \frac{1}{4} A(u^1, p^1), \\
\left(\begin{array}{c}
u_{n+1}^1 \\
p_{n+1}^1
\end{array}\right) = \frac{1}{3} \left(\begin{array}{c}
u^n \\
p^n
\end{array}\right) + \frac{2}{3} A(u^2, p^2),
\end{cases}
\]

where \(\mathcal{RK}()\) denotes the Runge–Kutta method. Now the question remains is how well does the operator \(L\) approximates \(S\). We shall demonstrate this in the following sections.

3. Boundary condition details and projection verification

3.1. Boundary conditions

First we must consider imposing boundary conditions for the numerical method. Boundary conditions can be divided into no–slip boundary with Dirichlet conditions on velocity field, periodic boundary, inlet and outlet boundary with either Dirichlet or Neumann conditions imposed on velocity and symmetry boundary, where normal component of velocity is set to zero, while zero Neumann conditions are set for tangential component of velocity. Observe, that for boundary conditions including pressure difference (e.g. Poiseuille flow) one can insert constant pressure gradient and apply it as an external forcing term.

No–slip boundaries on the wall that moves with velocity \((u_{1,x}, u_{1,y}, u_{1,z}, u_{2,x}, u_{2,y}, u_{2,z})\) are set as follows: \(u_x = u_{1,x}, u_y = u_{1,y}, u_z = u_{1,z}\). This results into imposing boundary conditions near the wall - natural velocity conditions and artificial boundary conditions for the scalar function \(\phi\). Numerically, this results in setting ghost cells from the other side of the domain with full reflection. Such ghost cell approach is usually used in finite volume methods and inheritly conservative when applied with WENO schemes. From these ghost cells we deduce boundary conditions for compact differences schemes through integration, following [19]. Now we must pose zero Neumann boundary conditions on \(\phi\). In this case value \(g_j,0\) at the boundary equals 0. In this FEM setup it just means skipping values of \(\phi\) on zero Neumann boundary. The usage of this type of finite elements gives one more positive result. There is no artificial boundary layer near wall boundary, because no artificial boundary conditions are imposed on pressure while pressure gradient (19) needs no information from the boundary.

Inflow boundary conditions are set as follows: velocity on the inflow are prescribed, pressure is not defined. Alternatively, one defines pressure while velocities are set to zero Neumann boundary conditions in the direction of the inflow while other components are set to zero to insure correctness of the system solution, e.g. \(p = p_w, (u_x)_n = 0, u_y = 0, u_z = 0\). In this case artificial boundary conditions for the scalar function \(\phi\) on this boundary are of zero Dirichlet type. In this case inserting \(g_{j,0} = 1\) on this boundary in (18). Neumann boundary conditions for velocity component for the WENO scheme are set using polynomial extrapolation, for more information see [18]. Symmetry conditions are set analogously.

Outflow boundary conditions are using zero Neumann condition on the boundary for velocity. For this case we use the method of extrapolation ([18]) along with the application of the sponge zone to avoid boundary disturbances being transfered back into the domain. Dirichlet boundary conditions are set for the pressure. Zero Dirichlet boundary conditions are set for the function \(\phi\).

Finally, periodic boundary conditions are the easiest class of boundaries. In this case we associate boundary nodes from one side of periodic boundary to another for all scalar variables, while for WENO scheme and compact differences we exchange ghost cells.
3.2. Projection problem

Now let us consider some simple problem for the Stokes equations (8) with nonzero initial divergence field provided as initial conditions. The problem is described in the unit domain \([0,1]^2\) with either periodic or no–slip boundary. Initial conditions are provided by the smooth vector–function with nonzero divergence:

\[ u_j(x,0) = 1.0 \cdot e^{-30 \| x - x_0 \|^2}, \quad x_0 = (1/2, 1/2)^T, \quad j = 1, 2. \]

The intial vector field is shown in figure 2, left. For the periodic boundary conditions we also check the result with the Fourier spectral method, where velocity field is found by exact projection to the field of divergence free functions. The Poisson equation on the \(\phi\) scalar function for this test is solved by the Jacobi preconditioned CG method with target tolerance of \(1 \cdot 10^{-12}\) in MATLAB. It was enough to use this simple preconditioner, resulting in about 500–700 iterations for the grid \(160^2\). The diffusion operator is considered implicitly with \(\nu = 1/100\) and timestep \(\Delta t = 1/50\) is set.

The solution of the periodic problem for the grid \(80^2\) is provided in figure 2 and convergence history between iterations for different grids is provided in figure 1. One can observe that the flow is really divergence free up to some irreducible divergence error. This error is grater than that, obtained by the exact projection by the magnitude of \(\sim 10^2–10^3\). Observe, that the first step of divergence reduction results in tremendous divergence decrease for all grids (figure 1) with relaxation of the error. This final divergence error for each grid spacing is the error of \(S\) approximation by \(L\) operator.

Results for \(80^2\) grid are provided in figure 3 for the wall boundary conditions. We observe that the final error of operator approximation is higher (approximately by 10) then that for the periodic conditions (figure 1). However, the convergence is faster, probably, due to the influence of walls. Divergence error that is provided in figure 1 shows no degradation of the projection near no–slip walls.

In all cases we can see, that the divergence error is near the machine epsilon for the grid \(160^2\). This confirms that the constructed \(S\) operator approximation is efficient. From the computational point of view we can prescribe desired divergence tolerance and then perform iterations in (20) until the tolerance is met. It is noticed in practice, however, that at most two–three iterations is sufficient for relatively fine grid.

4. Verification

In this section we perform method verification with benchmark problems. For all further computational examples we use non–dimensionalized system of equations (1) by introduce the
Reynolds number as \( R = \frac{V^* L^*}{\nu} \), where \( V^* \) is the characteristic velocity and \( L^* \) is the characteristic length. Then the diffusion coefficient in (1) is replaced with \( R^{-1} \).

### 4.1 3D Shapiro flow with periodic boundary conditions

First, we consider a 3D flow that has an analytical solution, obtained by A. Shapiro [20]. This problem is commonly used as a benchmark to verify temporal accuracy of numerical methods for the Navier–Stokes equations. The flow is considered in a cubic domain \([0, 2\pi]^3\) with periodic boundary conditions. The suggested problem belongs to the family of Beltrami flows, i.e. \( \nabla \times \mathbf{u} = \alpha \mathbf{u} \), and \( \alpha \) is a constant. Initial conditions are prescribed as follows:

\[
\begin{align*}
    u_x(x, 0) &= -1/2 \left( \sqrt{3} \cos(x) \sin(y) \sin(z) + \sin(x) \cos(y) \cos(z) \right) e^{-\frac{2\sqrt{3}}{R}}, \\
    u_y(x, 0) &= 1/2 \left( \sqrt{3} \sin(x) \cos(y) \sin(z) - \cos(x) \sin(y) \cos(z) \right) e^{-\frac{2\sqrt{3}}{R}}, \\
    u_z(x, 0) &= (\cos(x) \cos(y) \sin(z)) e^{-\frac{2\sqrt{3}}{R}}.
\end{align*}
\]  

Figure 2. The projection Stokes flow problem with periodic boundary conditions. Initial vector field (left), result after 5 iterations (middle), divergence distribution after 5 iterations (right).

Figure 3. The projection Stokes flow test with no-slip boundary conditions, velocity field after first iteration, after 5 iterations, divergence distribution after 5 iterations.
Then the solution of the IBVP can be found as [20]:

\begin{align}
    u_x(x, t) &= -\frac{1}{4} (\cos(x) \sin(y) \sin(z) + \sin(x) \cos(y) \cos(z)) e^{-\frac{t}{\tau}}, \\
    u_y(x, t) &= \frac{1}{4} (\sin(x) \cos(y) \sin(z) - \cos(x) \sin(y) \cos(z)) e^{-\frac{t}{\tau}}, \\
    u_z(x, t) &= \frac{1}{2} (\cos(x) \cos(y) \sin(z)) e^{-\frac{t}{\tau}}.
\end{align}  \tag{24a} \tag{24b} \tag{24c}

The physical space representation of the solution (24) is presented in figure 4, left.

**Figure 4.** Shapiro solution visualization (left) and convergence test for different transient methods (right).

We solve this problem with periodic boundary conditions on mesh $300^3$ and different timesteps $\Delta t$ for $R = 1000.0$ and final time $t = 1000$. For the final time we find the $L_2$ norm of difference between vector (24) and numerical solution. The problem is solved using Euler, Runge–Kutta 2–d order and 3–d order methods. This norm is plotted in figure 4, on the right. As one can observe, the solution demonstrates claimed temporal order of approximation and shows degradation on the 3–d order Runge–Kutta method due to round–off errors.

### 4.2. 2D lid driven cavity flow

The famous 2D test of the lid driven cavity flow is considered. It is formulated as follows: domain is $[0, 1]^2$ with no–slip boundary conditions imposed. The upper section of the boundary segment $0 \leq x \leq 1$ at $y = 1$ (the “lid”) is moving with velocity $(u_x = 1, u_y = 0)$, thus a tangential flow on the boundary is imposed with zero discharge through computational domain. Canonical way to solve the problem is to use “vorticity–stream function” formulation. This problem was solved many times and very good and exact results are available, e.g. [21, 22, 23, 24] and many others. As the reference solution we take a very thorough work [23, 24], where 4-th order finite difference methods are applied to the problem for very fine grids and results are benchmarked with many common sources, including canonical work by Ghia et.al. [21]. Most importantly, that the author in [23] provides a link to his website where raw data can be found and checked. It is known, that this problem exhibits stationary solutions for super fine resolutions at least for $R \leq 21000$, see [23]. If the resolution is not sufficient, then the flow becomes unstable. The origin of this instability is beyond the scope of the paper. We fix the spatial resolution by $256^2$ and solve the problem with no grid refinement or resolution increase for high Reynolds numbers. Instantaneous streamlines are provided in figure 6. One observes formation of the
main recirculation zone with secondary recirculation vortexes. These vortexes become unstable for $R \geq 10000$ and we use averaged data to compare results.

Comparison of results is provided in figure 6 in canonical way by comparing $u_x$ and $u_y$ velocity profiles along lines passing through $y = 1/2$ and $x = 1/2$, respectively, with data from [23]. We can observe very good agreement for low Reynolds numbers and some difference in results for higher ones. This is expected, since we compare averaged unstationary solution to the stationary one.

4.3. 3D lid driven cavity flow
We consider a generalization of the 2D lid driven cavity flow for 3D domain. The domain is $[0, 1]^3$, all boundaries have no-slip conditions imposed. The upper lid, represented by the plane at $y = 1$ is moving, thus inducing flow in the domain. We consider two setups of the problem: the direct lid movement that prescribes $u_x = 0, u_y = 0, u_z = 1$ on the lid, first considered, probably by [25], and more thoroughly [26]. And oblique lid movement with $u_x = 1/\sqrt{2}, u_y = 0, u_z = 1/\sqrt{2}$ on the lid, suggested by [27]. We used grid size of $200^3$ for both problems and apply analytical coordinate transformation in the direction of walls:

$$
\xi_j = \frac{1}{2} + \frac{1}{2} \ln \left(\frac{(\gamma + 2x_j - 1)/(\gamma - 2x_j + 1)}{\ln ((\gamma + 1)/(\gamma - 1))}\right),
$$
that has an analytical inverse transformation:

\[ x_j = \frac{1 - \gamma + (1 + \gamma)((\gamma + 1)/(\gamma - 1))^{2\xi_j-1}}{2(1 + (\gamma + 1)/(\gamma - 1))^{2\xi_j-1}}, \]

in every direction \( j = 1, 2, 3 \) and we set stretching parameter \( \gamma = 1.1 \). This transformation is directly applied as Jacobi matrix and doesn’t influence spatial order of approximation. Since we use WENO schemes, we don’t have any special treatment of corner or edge singularities.

First we consider the oblique moving lid problem. Streamlines and sections of the flow field are provided in figures 7 for different Reynolds numbers. We can observe the diagonal plane of reflection symmetry for lower Reynolds number. Streamlines launched from the diagonal plane remain on this plane that can be seen in figure 7 for streamlines. We benchmark our solution against data from [28, 31], where finite volume method is used with SIMPLE pressure correction and very good data representation is available.

We compare point distribution of velocities at a central line \( x = z = 1/2 \), available in [28, 31] with our calculations, results are provided in table 1. We can observe that only slight differences can be noticed below 1%. At this central line for \( R = 1000 \) the flow is symmetric relative to the diagonal plane and so \( u_x = u_z \) up to \( 10^{-6} \) at that line. This demonstrates the reflection symmetry along that plane the flow possesses.

Then we check our results for the direct lid driven flow setup. Absolute values of velocity magnitude for different Reynolds numbers are presented in figures 8 streamlines, launched from different planes are presented in figures 9.

We compare our results with [29, 30] that provide velocity distributions along lines \( x = 1/2, z = 1/2 \) and \( x = 1/2, y = 1/2 \) for \( R = 1000 \). Comparing data and streamlines are provided in figure 9. We observe that we have very good agreement with reference data.
Figure 7. Sections of the velocity field for $R = 1000$ and $R = 2000$ and streamlines of 3D lid driven cavity flow with the oblique lid movement for $R = 1000$.

Table 1. Values of velocities and pressure for the central line in 3D oblique cavity flow, values marked * are from [28]. All values are multiplied by $10^3$.

| $y$     | $u_x^*$, $u_z^*$ | $u_{x+}, u_z$ | $u_y^*$ | $u_y$ |
|---------|------------------|---------------|---------|-------|
| 0.9766  | 417.7            | 417.812       | 5.378   | 5.391 |
| 0.9531  | 226.6            | 226.761       | 16.07   | 16.218|
| 0.8516  | 76.74            | 76.311        | 30.36   | 30.461|
| 0.7344  | 62.50            | 62.192        | 22.59   | 22.619|
| 0.6172  | 41.78            | 41.322        | 5.790   | 5.561 |
| 0.5000  | -1.398           | -1.395        | -33.95  | -34.031|
| 0.4531  | -31.54           | -31.133       | -64.70  | -64.342|
| 0.2813  | -130.7           | -131.002      | -160.2  | -160.672|
| 0.1719  | -134.7           | -134.891      | -137.9  | -138.049|
| 0.1016  | -143.1           | -143.277      | -86.78  | -87.009|

Figure 8. Absolute values of 3D lid driven cavity flow, $R = 500$, $R = 1000$, $R = 1500$ from left to right.

5. Conclusion
In the paper we describe a new mixed approximation type pressure correction method. The method is applied for Stokes and Navier–Stokes equations and can be efficiently used on high performance computing architecture, because involves all to all communication only for the
Figure 9. Streamlines of 3D lid driven cavity flow with the direct lid movement and comparison of central velocities for different lines for $R = 1000$. References denoted by crosses taken from [29].

Poisson equation. The latter can be efficiently solved on GPUs by applying a (geometric) multigrid method, e.g., [32]. The Poisson equation is formulated for the additional potential scalar function in terms of finite element method. We show that this formulation gives good approximation of the Schur complement pressure operator. We perform projection test of the Stokes flow and demonstrate efficiency of the suggested method. Next, the method is benchmarked against known problems demonstrating accuracy. It is shown that the suggested procedure is correct and can be used as a DNS solver.

Acknowledgments
This work is supported by RFBR grant no. 17-07-00116.

References
[1] G.S. Jiang and C.W. Shu. Efficient implementation of weighted ENO schemes. J. Comput. Phys. 126 (1), 202228, (1996).
[2] Girault, V., Raviart, P.A.: Finite Element Methods for Navier-Stokes Equations. Springer, (1986).
[3] Langtangen,H.P., Mardal,K-A., Winther, R.: Numerical methods for incompressible viscous flow. Advances in Water Resources, 25, 8–12, 1125–1146, (2002).
[4] Guermond, J. L., Jie Shen.: Velocity–Correction Projection Methods for Incompressible Flows. SIAM J. Numer. Anal., 41(1), 112–134., (2003).
[5] Guermond, J.L., Minev, P., Jie Shen.: An overview of projection methods for incompressible flows. Comput. Methods Appl. Mech. Engrg. 195, 8011–8045, (2006).
[6] Gresho, P.M., Sani, R.: Incompressible Flow and The Finite Element Method. Vol.1. – Wiley, New York, (1998).
[7] Nochetto, R.H., Jae-Hong Pyo.: The Gauge–Uzawa Finite Element Method. Part I: The Navier–Stokes Equations. SIAM J. Numer. Anal., 43(3), 1043–1068, (2006).
[8] Pyo, J–H.: A classification of the second order projection methods to solve the Navier–Stokes equations. Korean Journal of Mathematics 22:4, 645–658, (2014).
[9] Amsden,A., Harlow, F.: A simplified MAC technique for incompressible fluid flow calculations. Journal of Computational Physics, 6, 322–325, (1970).
[10] Yang, J.Y., Yang, S.C., Chen,Y. N., Hsu, C.A.: Implicit weighted ENO schemes for the three-dimensional incompressible Navier-Stokes equations. Journal of Computational Physics 146, 464–487, (1998).
[11] Zhang,Z., Jackson, T.L.: A high–order incompressible flow solver with WENO. Journal of Computational Physics, 228, 7, 2426–2442, (2009).
[12] Evstigneev, N.M., Ryabkov, O.I.: Application of multiGPU+CPU architecture for the direct numerical simulation of laminar-turbulent transition. Vychisl. Metody Programm., 17, 1, 55–64, (2016).

[13] Geuzaine, C., Remacle, J.-F.: Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. International Journal for Numerical Methods in Engineering 79(11), pp. 1309–1331, (2009).

[14] Oseledets V.I., A new form of writing out the Navier-Stokes equation. The Hamiltonian formalism.// Russian Math. Surveys, 44, pp. 210–212, (1989).

[15] Arakawa, A., Lamb, V.R.: Computational design of the basic dynamical processes of the UCLA general circulation model. Methods of Computational Physics. 17, pp. 173–265, (1977).

[16] Evstigneev, N.M.: On the construction and properties of WENO-schemes order five, seven, nine, eleven and thirteen. Part 1. Construction and stability. Computer Research and Modeling, 8, 5, pp. 721–753, (2016).

[17] Gottlieb, S., Chi-Wang Shu, Tadmor, E.: Strong stability-preserving high-order time discretization methods., SIAM review 43, 1, 89–112, (2001).

[18] Sirui Tan.: Boundary conditions and applications of WENO finite difference schemes for hyperbolic problems. PHD thesis in the Division of Applied Mathematics at Brown University, Providence, Rhode Island, (2012).

[19] Lele, S.K.: Compact Finite Difference Schemes with Spectral-like Resolution. Journal of Computational Physics, 103, 16–42 (1992).

[20] Shapiro, A.: The use of an exact solution of the Navier-Stokes equations in a validation test of a three-dimensional nonhydrostatic numerical model. Monthly Weather Review vol. 121, 2420-2425, (1993).

[21] Ghia, U., Ghia, K.N., Shin, C.T.: High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method. Journal of Computational Physics, 48, 387–411, (1982).

[22] Barragy E, Carey GF. Stream function-vorticity driven cavity solutions using p–finite elements. Computers and Fluids, 26, 453–468, (1997).

[23] Erturk, E., Gokcol, C.: Fourth Order Compact Formulation of Navier-Stokes Equations and Driven Cavity Flow at High Reynolds Numbers, International Journal for Numerical Methods in Fluids 2006, 50, pp 421–436.

[24] Erturk, E.: Comparison of wide and compact fourth-order formulations of the Navier–Stokes equations. International Journal for Numerical Methods in Fluids 2009, 60, pp 992–1010.

[25] Ku, H.C., Hirsh, R.S., Taylor, T.D.: A pseudospectral method for solution of the three-dimensional incompressible Navier-Stokes equations., J. Comput. Phys. 70, 439–462, (1987).

[26] Li, Q.T.m Cheng, T., Tsang, T.H.: Transient solutions for three-dimensional lid-driven cavity flows by a least-squares finite element method, Int. J. Numer. Meth. Fluids 21, 413–432, (1995).

[27] Povitsky, A.: High-incidence 3-D lid-driven cavity flow. AIAA Paper 2847, (2001).

[28] Feldman, Y., Gelfgat, A.Y.: From multi– to single–grid CFD on massively parallel computers: numerical experiments on lid–driven flow in a cube using pressure–velocity coupled formulation. Comput. Fluids 46(1), 218223, (2011).

[29] Albensoeder, S., Kuhlmann, H.C.: Accurate three-dimensional lid-driven cavity flow. Journal of Computational Physics, 206, 536–558, (2005).

[30] Zunic, Z., Hribarsek, M., Skerget, L., Ravnik. J. 3D lid driven cavity flow by mixed boundary and finite element method. ECCOMAS CFD 2006, 472, (2006).

[31] Feldman Y.: Theoretical analysis of three–dimensional bifurcated flow inside a diagonally lid–driven cavity. Theor.Comput.Fluid Dyn.29:245–261, (2015).

[32] Evstigneev, N.M.: Numerical integration of Poisson’s equation using a graphics processing unit with CUDA-technology. Vychisl. Metody Programm.,10,2, 268–274, (2009).