AN IMPLICIT ALGORITHM OF SOLVING NAVIER–STOKES EQUATIONS TO SIMULATE FLOWS IN ANISOTROPIC POROUS MEDIA

A.S. Kozelkov[1,2], S.V. Lashkin[1], V.R. Efremov[3], K.N. Volkov[4],
Yu.A. Tsibereva[1], N.V. Tarasova[1]

[1]Russian Federal Nuclear Center, Sarov, Russia

[2]Nizhny Novgorod State Technical University n.a. R. Alekseyev, Nizhny Novgorod, Russia

[3]JSC «KBP named after academician A. Shipunov», Tula, Russia

[4]Kingston University, London, United Kingdom

Abstract

A coupled computational algorithm for modified Navier–Stokes equations to simulate flows in anisotropic porous media is proposed and described. The difference from the classic SIMPLE algorithm is in the completely implicit relationship between velocity and pressure owing to the implicit terms of the pressure and mass flow gradients in the continuity equation and momentum equation. One of the attractive features of this algorithm is the possibility of completely implicit discretization of off-diagonal components of the porous medium resistance tensor in the right-hand side of the momentum equation. Implicit discretization allows reducing the number of linear iterations as compared to the SIMPLE algorithm with explicit discretization of off-diagonal components. The specific features of discretization of modified equations including the discretization of boundary conditions and components of the porous medium resistance tensor are considered. The proposed algorithm is verified in comparison with the SIMPLE algorithm on a series of benchmark problems, such as the problem of a flow through a porous insert, a flow in a divided channel, and a flow through a cylindrical porous filter. The total problem runtimes and the number of iterations required for complete convergence are given to compare the two algorithms.

Keywords

Navier–Stokes equations; porous media; Darcy law; Brinkman equations; multigrid method; pressure-based algorithm; implicit discretization.

1. Introduction

The numerical simulation of flows in complex engineering structures requires fine meshes with billions of cells regarding the structural specifics of objects, and the duration of a large-size numerical experiment may be from one to several months. In such cases the construction of a full-scale mesh model is an individual problem similar to the processing of the resultant huge data arrays. There are many examples of large-size problems in modern computational practice, one of them is the numerical simulation of a nuclear reactor core consisting of several hundred fuel elements (FEs) [Kolpakov & Selivanikova, 2009], which plays an important role for simulating both the turbulent flow structure and the integral thermal characteristics of the reactor. The simulation of car radiators, heat exchangers of different types consisting of a large number of tube bundles [Bhutta et al., 2012; Ozden& Tari, 2010], a flow through a variety of perforated plates [Malavasi et al., 2012; Ozahi, 2015], air filters in automobile industry [Elnaz, 2012] also fall into this category of problems.

One of the approaches to reduce computational loads is the porous body approximation [Ozden & Tari, 2010; Costa et al., 2004a] used instead of repetitive structures. This approach requires a mathematical model of a porous body, which takes into account geometric features of an object

to be simulated, its material properties (averaging of the thermal characteristics of the porous medium skeleton and fluid), and the flow conditions (isothermal or non-isothermal). There is a need in justification of the model application within the wide range of velocities (dependence on Reynolds number). For a certain type of porous bodies, for example, dense-packed spheres, there are valid mathematical expressions for the calculation of resistance factors [Ergun, 1952; Scheidegger, 1974]. However in most cases, the calculation of hydraulic resistance factors is based on the available experimental data on losses of pressure in a porous body depending on Reynolds number. It should be noted that in repetitive and symmetric cases data on a single area of body is sufficient for simulations [Collins, 1961].

The well-known Darcy law [Reichenberger et al., 2006; Chandesris & Jamet, 2006] is the simplest empirical rule to account for the porous medium resistance. It describes the linear relationship between the velocity of fluid flow and the pressure gradient at low velocities. The range of velocities is extended by introducing the quadratic dependence of pressure gradient on velocity using the Forchheimer law [Costa et al., 2004a, 2004b; Kulkarni et al., 2004; Shavit et al., 2003]. It is quite difficult, however, to use these equations in laminar areas with porous and open domains, where it is necessary to provide consistent solution of Navier–Stokes and Darcy (or Forchheimer) equations. To resolve this problem, there is a modification to the original Navier–Stokes equations with Darcy and Forchheimer laws in the right-hand side and with the molecular viscosity modification according to Brinkman equation. The modified equations are often called Brinkman–Forchheimer equations [Costa et al., 2004a; Kaviany, 1991]. This approach allows simulating both flows in domains completely occupied by a porous body and flows with sub-domains containing free fluid/porous body interfaces with Brinkman–Forchheimer equations in non-porous domains being degenerated into the original Navier–Stokes equations. The practice of using these equations demonstrates a good accuracy of description of flows in porous media [Costa et al., 2004a].

By present time, there have been accumulated a wide experience of solving modified Navier–Stokes equations with the finite volume method [Ferziger & Peric, 2002;Volkov et al., 2014a; Rhie & Chow, 1983; Kozelkov et al., 2016c]. The SIMPLE algorithm of the predictor–corrector type should be noted among the available computational algorithms. It allows iteratively finding the consistent velocity and pressure fields. The algorithm has been adapted to solve Brinkman–Forchheimer equations [Costa et al., 2004a]. The simulation of flows in anisotropic porous media or the use of local coordinate systems [Kaviany, 1991] leads to non-zero off-diagonal components in the resistance tensor. The classic SIMPLE algorithm implies that off-diagonal components of the resistance tensor are explicitly approximated in the momentum conservation equation, because equations are solved successively for each velocity component. An explicit approximation significantly lowers the SIMPLE algorithm convergence rate [Lashkin et al., 2016]. It is possible to speed-up the convergence by using an algorithm for the coupled velocity-pressure simulation, which is based on an implicit relationship of the momentum and continuity equations [Darwish et al., 2009]. In some studies [Darwish et al., 2009; Emans & Liebmann, 2013; Chen & Przekwas, 2010; Shterev & Stefanov, 2010; Mangani, 2010] this algorithm is used to simulate laminar flows of viscous compressible and incompressible fluids. It demonstrated a several-fold increase of the convergence rate on a sequence of condensed meshes in comparison with the SIMPLE algorithm. The coupled algorithm is also used to simulate turbulent flows [Lashkin et al., 2016; Pelinovskii et al., 2016; Kozelkov, 2016]. However, a significant dependence of the convergence rate on the relaxation parameters of turbulence equations is shown, and the use of the coupled algorithm is declared to be preferable considering the wall-clock time of computations.

State-of-the-art simulators employ numerical methods that can take advantage of multiple processors, distributed memory workstations, adaptive grid refinement strategies, and iterative techniques with linear complexity

The implicit methods have been constructed for a faster convergence to the Navier-Stokes solutions. The multigrid methods are commonly used in CFD community for solving the Euler and Navier-Stokes equations. The basic idea behind all multigrid strategies is to accelerate the solution at fine mesh by computing corrections on a coarser mesh to eliminate low-frequency errors efficiently. In general, an iterative algorithm can reduce the high-frequency errors faster than the low-frequency ones. The multiple mesh method is to make the transition between the low and high frequency modes through a change of cell size, and to eliminate the low frequency error in an even coarse mesh by increasing its spatial frequency. There are many studies about multigrid techniques and their numerical implementations (Brandt, 1982; Stuben and Trottenberg, 1982; Wesseling, 1992; Stuben, 2001). To achieve a good compromise of high efficiency and robustness for a variety of flow problems on structures and unstructured meshes different strategies to mesh coarsening, design of smoothing, restriction and prolongation operators and preconditioners have been proposed and tested in the literature (Mavriplis, 2002; Alkishriwi et al., 2006; Mavriplis, 2007; Cagnone et al., 2011; Langer , 2013; Sun et al., 2017).

For flows in porous media, an attractive feature of the coupled algorithm is that the linear resistance tensor can be approximated in a completely implicit manner, even in an anisotropic case, and the convergence rate significantly increases.

In the present study, the coupled computational algorithm for velocities and pressures based on an implicit relationship between the momentum and continuity equations is used to solve Brinkman–Forchheimer equations describing a flow in an anisotropic porous body. Finite volume discretization of the governing equations and the main steps of the computational procedure are described and presented. The implementation of the coupled implicit algorithm is verified in a series of benchmark problems on the simulation of a flow through a porous material insert, a flow in a divided channel and a flow through an anisotropic cylindrical filter. The test problems are specifically selected to make clear distinctions between the different methods.

2. Governing equations and their discretization

Brinkman–Forchheimer equations of the momentum conservation and continuity describing the laminar steady viscous flow of incompressible fluid in a porous medium, which are written relative to a real velocity, has the form [Costa et al., 2004a; Kaviany, 1991; Nield & Bejan, 2013]

$$\begin{cases} \nabla\cdot\left(\varepsilon\rho\boldsymbol{u}\right)=0, \\ \nabla\cdot\left(\varepsilon\rho\boldsymbol{u}\otimes\boldsymbol{u}\right)=-\varepsilon\nabla\cdot\left(p\boldsymbol{I}\right)+\nabla\cdot\left(\varepsilon\boldsymbol{\tau}\right)-\left(\dfrac{\varepsilon^2\mu}{\boldsymbol{K}}+\varepsilon^3 F\boldsymbol{C}\left|\boldsymbol{u}\right|\right)\cdot\boldsymbol{u}, \end{cases} \tag{1}$$

where $\varepsilon = V_{por}/V_{all}$ is the dimensionless porosity ratio; V_{por} is the pore volume; V_{all} is the total volume of cell; ρ is the density of material to be simulated; $\boldsymbol{u}=u\boldsymbol{i}+v\boldsymbol{j}+w\boldsymbol{k}$ is the real velocity vector; $\boldsymbol{i},\boldsymbol{j},\boldsymbol{k}$ are unit vectors in x, y, z directions, respectively; p is the static pressure; $\boldsymbol{I}$ is a unit tensor; μ is the viscosity; $\boldsymbol{\tau}=\mu_B\left(\nabla\boldsymbol{u}+\nabla\boldsymbol{u}^T\right)$ is the viscous stress tensor for incompressible Newtonian media; μ_B is the modified Brinkman molecular viscosity in a porous medium; $\boldsymbol{K}$ is the permeability tensor; F is Forchheimer coefficient depending on the flow parameters; $\boldsymbol{C}$ is the inertial resistance tensor. The velocity gradient tensor has the form

$$\nabla \boldsymbol{u} = \begin{pmatrix} \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial x}, \frac{\partial v}{\partial y}, \frac{\partial v}{\partial z} \\ \frac{\partial w}{\partial x}, \frac{\partial w}{\partial y}, \frac{\partial w}{\partial z} \end{pmatrix}. \quad (2)$$

Superscript T corresponds to the conjugate tensor.

The density in equations (1) remains constant. The porosity coefficient is a function of Cartesian coordinates and time. However, it does not depend on velocity and pressure. Therefore, the equation of state is not required.

To solve the system of equations (1) for incompressible fluid, the algorithm based on Rhie–Chow correction to the continuity equation is widely used [Rhie & Chow, 1983]. This correction makes it possible to solve the continuity equation with respect to the pressure field that provides a link between the continuity equation and momentum equation. A detailed description of the use of Rhie–Chow correction is given in [Zhang et al., 2014].

The diagonal components of the inertial resistance tensor, $\boldsymbol{C}$, in isotropic porous media are proportional to the value of $\rho K^{-1/2}$ [Ward, 1964], where K is the permeability. In anisotropic porous media this dependence may be invalid, and calculations of the components of the inertial stress tensor are required to take into account sizes of pores and inter pore spaces, flow regime, etc. [Muljadi et al., 2016].

The permeability tensor, $\boldsymbol{K}$, is determined by the geometrical structure of porous medium (it doesn't depend on material properties) and, in general, it is determined by the pore volume geometry. It is clear from the equation (1) that $\boldsymbol{K}$ has the squared length size and is a rough measure of the root-mean-square diameter of pores. For isotropic porous media, the permeability tensor is diagonal. This indicates a similarity of the geometric properties of medium in all directions. In the anisotropic porous media or in the case of setting the permeability value in local coordinates (for example, cylindrical) $\boldsymbol{K}$ is symmetric with filled off-diagonal coefficients indicating that permeability depends on the coordinate system direction. In heterogeneous porous media the permeability tensor may be the function of time and coordinates.

The effect of non-linear Forchheimer coefficient, F, is noticeable at local Darcy numbers above 10^{-3}, so these terms may be neglected only in some cases of a laminar flow [Prasad et al., 1985].

The modified Brinkman molecular viscosity is set in a porous medium. It depends on the medium porosity factor and is represented in the following form [Brinkman, 1952]

$$\mu_B = (1-\varepsilon)^{-0.25}. \quad (3)$$

The last two terms in the momentum equation (1) are the Darcy and Forchheimer ones. In real calculations with the use of experimental data on pressure losses it is easy to represent these terms in the form of the resistance tensor

$$\boldsymbol{P} = \frac{\varepsilon^2 \mu}{\boldsymbol{K}} + \varepsilon^3 F \boldsymbol{C} |\boldsymbol{u}| = \boldsymbol{\beta} + \boldsymbol{\alpha} |\boldsymbol{u}| = \begin{bmatrix} \beta_{11} & \beta_{12} & \beta_{13} \\ \beta_{21} & \beta_{22} & \beta_{23} \\ \beta_{31} & \beta_{32} & \beta_{33} \end{bmatrix} + \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} \\ \alpha_{31} & \alpha_{32} & \alpha_{33} \end{bmatrix} |\boldsymbol{u}|, \quad (4)$$

where $\boldsymbol{\alpha}$ is the inertial (non-linear) resistance tensor, $\boldsymbol{\beta}$ is the viscous (linear) resistance tensor. It should be noted that tensors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ always have a symmetric structure and in case of the

coordinate system coinciding with the main axes of porous medium (axes should be orthogonal) these tensors are diagonal.

Equations (1) describe flows of viscous compressible and incompressible fluids both in porous and open domains, as well as at an interface of different porous media. These equations can also be used to solve problems with a porosity ratio varying in space and time and, accordingly, a real velocity can be set.

Using the finite volume method [Ferziger & Peric, 2002] for an arbitrary cell P (Figure 1), the discrete form of the equations (1) without regard to time terms (their discretization is described in details in [Ferziger & Peric, 2002; Jasak, 1996]) in case of a laminar steady flow can be written as

$$\begin{cases} \sum\limits_{f=nb(P)} \rho_f \varepsilon_f \left(\boldsymbol{u}_f^{n+1} \cdot \boldsymbol{S}_f \right) = 0, \\ \sum\limits_{f=nb(P)} \varepsilon_f \rho_f \left(\boldsymbol{u}^{n+1} \otimes \boldsymbol{u}^n \right)_f \cdot \boldsymbol{S}_f - \sum\limits_{f=nb(P)} \varepsilon_f \boldsymbol{\tau}_f^{n+1} \cdot \boldsymbol{S}_f - \sum\limits_{f=nb(P)} \varepsilon_f p_f^{n+1} \boldsymbol{S}_f = \varepsilon_P V_P \left(\boldsymbol{\beta} + \boldsymbol{\alpha} \left| \boldsymbol{u}^n \right| \right)_P \cdot \boldsymbol{u}_P^{n+1}. \end{cases} \quad (5)$$

Here, the subscript f signifies that a given quantity belongs to a face separating control volumes of the computational mesh; $\boldsymbol{S}_f = \boldsymbol{n}_f S_f$ is the area-vector of the face; $\boldsymbol{n}_f$ is the unit normal vector to the face; V is the volume of cell P. The superscript $n+1$ signifies the value of quantity at a new iterative layer, and the superscript n is related to the previous iteration. The summation in equations (5) is performed over all faces $f = nb(P)$ which form a control volume P.

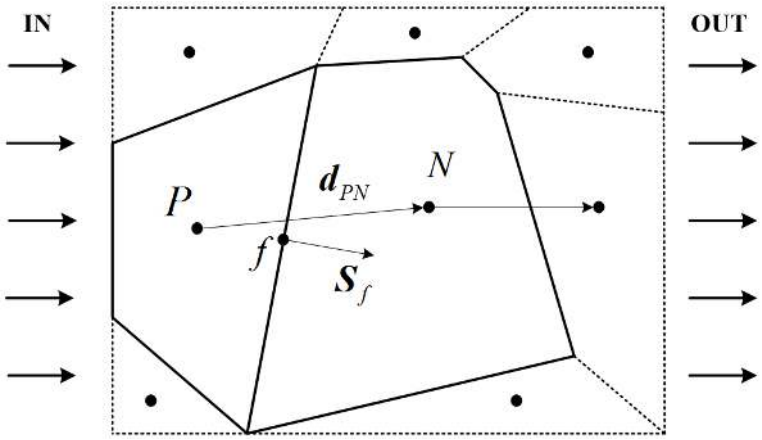

Figure 1. Two neighboring control volumes P and N (f is the face separating cells P and N, $\boldsymbol{d}_{PN}$ is the vector connecting the central points of two control volumes)

Linearization of the original system of equations (5) is performed with the simple iteration method [Samarsky & Gulin, 1989]. Using the coupled implicit algorithm [Lashkin et al., 2016; Darwish et al., 2009] to solve equations (5), the equations are represented as a system of linear algebraic equations (SLAE) for an arbitrary control volume P. It is more convenient to represent it in the block form

$$\begin{bmatrix} a_P^{pp} & a_P^{pu} & a_P^{pv} & a_P^{pw} \\ a_P^{up} & a_P^{uu} & a_P^{uv} & a_P^{uw} \\ a_P^{vp} & a_P^{uv} & a_P^{vv} & a_P^{vw} \\ a_P^{wp} & a_P^{wu} & a_P^{wv} & a_P^{ww} \end{bmatrix} \begin{bmatrix} p_P^{n+1} \\ u_P^{n+1} \\ v_P^{n+1} \\ w_P^{n+1} \end{bmatrix} + \sum_{N=nb(P)} \begin{bmatrix} a_N^{pp} & a_N^{pu} & a_N^{pv} & a_N^{pw} \\ a_N^{up} & a_N^{uu} & a_N^{uv} & a_N^{uw} \\ a_N^{vp} & a_N^{uv} & a_N^{vv} & a_N^{vw} \\ a_N^{wp} & a_N^{wu} & a_N^{wv} & a_N^{ww} \end{bmatrix} \begin{bmatrix} p_N^{n+1} \\ u_N^{n+1} \\ v_N^{n+1} \\ w_N^{n+1} \end{bmatrix} = \begin{bmatrix} b_P^p \\ b_P^u \\ b_P^v \\ b_P^w \end{bmatrix}. \quad (6)$$

Here, a_P and a_N are the diagonal and off-diagonal coefficients of the SLAE block matrix, respectively, and b_P is the right-hand side of the system including the volume sources. Coefficients with the subscript N represent the relationship between control volume P and its

neighboring control volume *N*. The first row in (6) is the continuity equation, which is solved relative to absolute pressure. The next three rows are the momentum equations for the three velocity components. Equation (6) is written for each cell (Figure1) and the number of such equations equals the number of control volumes in computational model.

To discretize the convective term in (5) and to improve the stability of the method, the deferred correction method is used [Ferziger & Peric, 2002; Jasak, 1996]

$$\left(\boldsymbol{u}_f^{HO}\right)^{n+1} = \left(\boldsymbol{u}_f^{UD}\right)^{n+1} + \beta\left(\boldsymbol{u}_f^{HO} - \boldsymbol{u}_f^{UD}\right)^{n}. \quad (7)$$

Here, $\boldsymbol{u}_f^{HO}$ is the velocity vector on face according to the higher order scheme; $\boldsymbol{u}_f^{UD}$ is the velocity vector on face according to the first order scheme; $0 \le \beta \le 1$ is the numerical scheme mixing factor. This approach is applicable to any higher-order scheme, including those with the TVD gradient limiters [Jasak, 1996]. By varying the values of β, it is possible to increase numerical diffusion and, thereby, improve the computation stability. With $\beta = 0$ the given scheme becomes the first-order scheme, and with $\beta = 1$ it becomes a higher-order scheme. The description of discretization schemes for flows with convection can be found in [Kozelkov et al., 2015a, 2016a; Volkov et al., 2014a; Kozelkov & Kurulin, 2015].

The original equations (5) use the non-orthogonal correction, which is written for an arbitrary scalar quantity φ [Jasak, 1996]

$$\nabla\varphi_f \cdot \boldsymbol{S}_f = \left(\varphi_N - \varphi_P\right)\frac{\boldsymbol{S}_f \cdot \boldsymbol{S}_f}{\boldsymbol{S}_f \cdot \boldsymbol{d}_{PN}} + \overline{\nabla\varphi}_f \cdot \boldsymbol{S}_f - \left(\overline{\nabla\varphi_f} \cdot \boldsymbol{d}_{PN}\right)\frac{\boldsymbol{S}_f \cdot \boldsymbol{S}_f}{\boldsymbol{S}_f \cdot \boldsymbol{d}_{PN}}, \quad (8)$$

where $\overline{\nabla\varphi_f} = \dfrac{\nabla\varphi_P + \nabla\varphi_N}{2}$ is the averaged gradient of φ on a face.

To summarize, the process of filling and generating coefficients of the matrix SLAE (6) can be represented as a sequence of successive steps.

Step 1. Coefficients for the convective and diffusive terms of the vector algebraic equation of momentum conservation are generated

$$\begin{aligned} a_N^{uu} &= a_N^{vv} = a_N^{ww} = \varepsilon_f \mu_f S_f^* + \min\left(m_f, 0\right), \\ a_P^{uu} &= a_P^{vv} = a_P^{ww} = \varepsilon_f \mu_f S_f^* + \max\left(m_f, 0\right), \\ b_P^u &= b_P^v = b_P^w = \varepsilon_f \mu_f \left(\nabla\boldsymbol{u}_f + \nabla\boldsymbol{u}_f^T\right) \cdot \boldsymbol{S}_f - \varepsilon_f \mu_f S_f^* \left(\nabla\boldsymbol{u}_f \cdot \boldsymbol{d}_{PN}\right), \end{aligned} \quad (9)$$

where $S_P^* = \dfrac{\boldsymbol{S}_f \cdot \boldsymbol{S}_f}{\boldsymbol{S}_f \cdot \boldsymbol{d}_{PN}}$ is the reference area also used in the non-orthogonal correction; $m_f = \left(\rho_f \boldsymbol{u}_f\right) \cdot \boldsymbol{S}_f$ is the mass flux on a face. Functions $\min$ and $\max$ are used to find minimum and maximum of two arguments.

Step 2. Coefficients for the term of pressure forces in the momentum equation are calculated. In the coupled algorithm this term is calculated implicitly in contrast to the SIMPLE algorithm, in which the fully implicit simulation is impossible, because the pressure in the original equation is a parameter rather than a desired quantity. That's why the pressure forces in the SIMPLE algorithm are discretized explicitly, and the pressure coefficients are introduced to the right-hand side from the previous iteration. For the coupled algorithm, the block matrix coefficients are modified

$$a_N^{up} = \varepsilon_f \left(1-\lambda_f\right) S_f^x \quad a_N^{vp} = \varepsilon_f \left(1-\lambda_f\right) S_f^y \quad a_N^{wp} = \varepsilon_f \left(1-\lambda_f\right) S_f^z,$$
$$a_P^{up} = \varepsilon_f \lambda_f S_f^x \quad a_P^{vp} = \varepsilon_f \lambda_f S_f^y \quad a_P^{wp} = \varepsilon_f \lambda_f S_f^z, \tag{10}$$

For the SIMPLE algorithm, the right-hand side of the system is modified

$$b_N^u = \varepsilon_f p_f^n S_f^x \quad b_N^v = \varepsilon_f p_f^n S_f^y \quad b_N^w = \varepsilon_f p_f^n S_f^z,$$
$$b_P^u = \varepsilon_f p_f^n S_f^x \quad b_P^v = \varepsilon_f p_f^n S_f^y \quad b_P^w = \varepsilon_f p_f^n S_f^z, \tag{11}$$

where $\varepsilon_f = \left(1-\lambda_f\right)\varepsilon_P + \lambda_f \varepsilon_N$ is the porosity factor on face; $\lambda_f = \dfrac{a_N}{a_P + a_N}$ is the interpolation factor based on the diagonal coefficients in the momentum equation matrix [Rhie & Chow, 1983]; $a_{P,N} = \left(\boldsymbol{a}_{P,N} \cdot \boldsymbol{n}_f\right) \cdot \boldsymbol{n}_f$ is an averaged diagonal coefficient in the momentum equation; $\boldsymbol{a}_{P,N}$ is a diagonal coefficient in the momentum equation represented in the form of tensor in the coupled solver; $S_f^{x,y,z}$ are the area vector components in directions *x*, *y*, *z*, respectively.

Step 3. The last step in generating coefficients for the momentum equation is the key stage of the porous body model implementation. In addition to the above described coefficients of the block SLAE, terms of the non-linear and linear parts of the permeability tensor are summarized

$$\begin{aligned}
&a_P^{uu} \mathrel{+}= \left(\beta_{11} + \alpha_{11}|\boldsymbol{u}|\right)V_P \quad a_P^{uv} \mathrel{+}= \left(\beta_{12} + \alpha_{12}|\boldsymbol{u}|\right)V_P \quad a_P^{uw} \mathrel{+}= \left(\beta_{13} + \alpha_{13}|\boldsymbol{u}|\right)V_P,\\
&a_P^{vu} \mathrel{+}= \left(\beta_{21} + \alpha_{21}|\boldsymbol{u}|\right)V_P \quad a_P^{vv} \mathrel{+}= \left(\beta_{22} + \alpha_{22}|\boldsymbol{u}|\right)V_P \quad a_P^{vw} \mathrel{+}= \left(\beta_{23} + \alpha_{23}|\boldsymbol{u}|\right)V_P,\\
&a_P^{wu} \mathrel{+}= \left(\beta_{31} + \alpha_{31}|\boldsymbol{u}|\right)V_P \quad a_P^{wv} \mathrel{+}= \left(\beta_{32} + \alpha_{32}|\boldsymbol{u}|\right)V_P \quad a_P^{ww} \mathrel{+}= \left(\beta_{33} + \alpha_{33}|\boldsymbol{u}|\right)V_P.
\end{aligned} \tag{52}$$

The linear part of permeability tensor (52) is fully implicit, in contrast to the SIMPLE algorithm, where the implicit discretization is impossible, because the solution is performed successively for the three velocity components and, hence, there is no implicit relationship between the velocity components.

After all coefficients for the momentum equation have been generated (the lower three rows of the block matrix have been filled), the matrix coefficients for the continuity equation are calculated (the upper row). Here, the key point of discretization is the fully implicit simulation of the mass flux, in contrast to the SIMPLE algorithm, where the mass flux is taken from the previous iteration layer and introduced to the right-hand side. Besides the mass flux, diagonal elements and the right-hand side with respect to pressure are also generated (similarly to the SIMPLE algorithm) and the implicit mass flux for the coupled algorithm

$$\begin{aligned}
&a_N^{pp} = -\varepsilon_f^2 \rho_f D_f S_f^* \quad a_P^{pp} = -\sum_{f=nb(P)} a_N^{pp},\\
&b_P^p = \varepsilon_f^2 \rho_f D_f S_f^* \left(\nabla p_N \cdot \boldsymbol{d}_{Pf} - \nabla p_P \cdot \boldsymbol{d}_{Nf}\right),\\
&a_N^{pu} = \lambda_f \varepsilon_f \rho_f S_f^x \quad a_P^{pu} = \left(1-\lambda_f\right)\varepsilon_f \rho_f S_f^x,\\
&a_N^{pv} = \lambda \varepsilon_f \rho_f S_f^y \quad a_P^{pv} = \left(1-\lambda_f\right)\varepsilon_f \rho_f S_f^y,\\
&a_N^{pw} = \lambda \varepsilon_f \rho_f S_f^z \quad a_P^{pw} = \left(1-\lambda_f\right)\varepsilon_f \rho_f S_f^z,
\end{aligned} \tag{13}$$

where $D_f = \dfrac{V_P + V_N}{a_p + a_N}$ is the mean harmonic averaging of the Rhie–Chow correction coefficient [Rhie & Chow, 1983]; ρ_f is the density on face *f*, which is calculated using the given scheme (in this study, the specific discretization scheme in use is not important in view of an incompressible material). It should be noted that velocity $\boldsymbol{u}_f$ on face f (for the mass flux in the continuity

equation) and pressure (for the pressure forces in the momentum equation) are calculated with opposite interpolation factors

$$\begin{aligned} \boldsymbol{u}_f &= \left(1-\lambda_f\right)\boldsymbol{u}_P + \lambda_f \boldsymbol{u}_N, \\ p_f &= \lambda_f p_P + \left(1-\lambda_f\right) p_N. \end{aligned} \tag{14}$$

This approach allows making the velocity and pressure values consistent according to the mean harmonic averaging [Rhie & Chow, 1983].

3. Discretization of boundary conditions

Specification of the boundary conditions is required to close and solve equations (5). In view of the fact that two equations in (5) are rigidly bound by means of Rhie–Chow correction [Rhie & Chow, 1983], at one boundary it is sufficient to specify boundary conditions for one equation only. However, in general the boundary conditions should be specified for all equations.

The boundary condition implementation in the coupled algorithm to solve equations (5) significantly differs from the process in the SIMPLE algorithm. First of all, this concerns the mass flux in the continuity equation and pressure forces in the momentum equation. The boundary coefficients for an arbitrary control volume P depend on the type of boundary conditions, such as inlet, outlet, wall, etc.

3.1. Inlet (velocity components)

The inlet boundary condition implies setting only three velocity components for the momentum equation. Thus, the implicit discretization of the convective term is the only possible way. The diffusive term and pressure gradient discretization is performed implicitly, similarly to interior faces.

The explicit way of setting velocity components and mass flux does not imply the use of Rhie–Chow correction and, hence, the calculated mass flux with regard to the porosity factor is put to the right-hand side of equation.

For the inlet boundary condition in a porous body, it is necessary to extrapolate the pressure from the boundary cell central point. Usually, the extrapolation with the pressure gradient from the previous iteration step is used. In that case the explicit term is added to the right-hand side of the momentum equation. With regard to the foresaid, matrix coefficients are as follows

$$\begin{aligned} a_P^{uu} &= a_P^{vv} = a_P^{ww} = \varepsilon_f \mu_f S_f^* - \min\left(m_f, 0\right), \\ a_P^{uu} &= \sum_{f=nb(P)} a_N^{uu} \quad a_P^{vv} = \sum_{f=nb(P)} a_N^{vv} \quad a_P^{ww} = \sum_{f=nb(P)} a_N^{ww}, \\ b_P^u &= \varepsilon_f \mu_f \left(\nabla \boldsymbol{u}_P + \nabla \boldsymbol{u}_P^T\right)\cdot \boldsymbol{S}_f - \varepsilon_f \mu_f S_f^* \left(\nabla \boldsymbol{u}_P \cdot \boldsymbol{d}_{PN}\right), \\ b_P^v &= \varepsilon_f \mu_f \left(\nabla \boldsymbol{v}_P + \nabla \boldsymbol{v}_P^T\right)\cdot \boldsymbol{S}_f - \varepsilon_f \mu_f S_f^* \left(\nabla \boldsymbol{v}_P \cdot \boldsymbol{d}_{PN}\right), \\ b_P^w &= \varepsilon_f \mu_f \left(\nabla \boldsymbol{w}_P + \nabla \boldsymbol{w}_P^T\right)\cdot \boldsymbol{S}_f - \varepsilon_f \mu_f S_f^* \left(\nabla \boldsymbol{w}_P \cdot \boldsymbol{d}_{PN}\right), \\ b_P^p &= -\varepsilon_P m_f, \\ b_P^{u,v,w} &+= \varepsilon_P \left(\nabla p_P \cdot \boldsymbol{d}_{Pf}\right) S_f^{x,y,z}. \end{aligned} \tag{15}$$

3.2. Outlet (mass flux)

For incompressible flows, there is a possibility of setting the outlet boundary condition as a mass flux equal to the total inlet mass. This approach requires specification of the reference pressure because in case of its absence the resultant matrix is uncertain and has an infinite number of solutions (the determinant equals zero). It is possible to set the reference pressure either by selecting an arbitrary cell with the specified pressure and further accounting it in the whole model, or by using the matrix way with the explicit specification of the solution in the basic cell. In any case, coefficients are calculated similarly to the inlet boundary condition.

3.3. Outlet (static pressure)

Another general-purpose outlet boundary condition is the specified static pressure. It can be used both for compressible and incompressible flows. Though such approach does not ensure equal inlet and outlet masses, it allows avoiding the uncertain matrix solution. The convective term and pressure gradient in the momentum equation are implemented explicitly. Similarly to interior faces, Rhie–Chow correction [Rhie & Chow, 1983] is used for the continuity equation.

$$
\begin{aligned}
&a_P^{uu} = a_P^{vv} = a_P^{ww} = \varepsilon_f \mu_f S_f^* - \min\left(m_f, 0\right),\\
&a_P^{uu} = \sum_{f=nb(P)} a_N^{uu} \quad a_P^{vv} = \sum_{f=nb(P)} a_N^{vv} \quad a_P^{ww} = \sum_{f=nb(P)} a_N^{ww},\\
&b_P^u = \varepsilon_f \mu_f \left(\nabla \boldsymbol{u}_P + \nabla \boldsymbol{u}_P^T\right) \cdot \boldsymbol{S}_f - \varepsilon_f \mu_f S_f^* \left(\nabla \boldsymbol{u}_P \cdot \boldsymbol{d}_{PN}\right),\\
&b_P^v = \varepsilon_f \mu_f \left(\nabla \boldsymbol{v}_P + \nabla \boldsymbol{v}_P^T\right) \cdot \boldsymbol{S}_f - \varepsilon_f \mu_f S_f^* \left(\nabla \boldsymbol{v}_P \cdot \boldsymbol{d}_{PN}\right),\\
&b_P^w = \varepsilon_f \mu_f \left(\nabla \boldsymbol{w}_P + \nabla \boldsymbol{w}_P^T\right) \cdot \boldsymbol{S}_f - \varepsilon_f \mu_f S_f^* \left(\nabla \boldsymbol{w}_P \cdot \boldsymbol{d}_{PN}\right),\\
&a_P^{pu} = \varepsilon_P S_f^x \quad a_P^{pv} = \varepsilon_P S_f^y \quad a_P^{pw} = \varepsilon_P S_f^z.
\end{aligned}
\tag{16}
$$

It should be noted that this outlet boundary condition can also be used as the inlet boundary condition.

3.4. No-slip wall

The most significant difference between the coupled and SIMPLE algorithms consists in the implementation of the no-slip wall boundary condition. The SIMPLE algorithm uses the known deferred correction method [Ferziger & Peric, 2002] leading to the similarity of diagonal coefficients of the three SLAE velocities. This approach provides stability of simulations. However, it lowers the rate of convergence owing to the explicit way of accounting the velocity components. The coupled algorithm uses the known implicit approach [Darwish et al., 2009]. The implicit approach idea is to decompose the tangential velocity with respect to the outward normal vector components and take into account the off-diagonal coefficients coupling the velocity components with each other.

The surface friction force is calculated from the equality

$$\boldsymbol{F} = \boldsymbol{\tau}_w \cdot \boldsymbol{S}_f, \tag{17}$$

where $\tau_w = \mu_l \dfrac{\boldsymbol{u}_t - \boldsymbol{u}_w}{\boldsymbol{d}_{Pf}}$ is the viscous stress tensor on wall; $\boldsymbol{u}_t = \boldsymbol{u}_P - \left(\boldsymbol{u}_P \cdot \boldsymbol{n}_w\right)\boldsymbol{n}_w$ is the tangential velocity component in cell (the velocity parallel to the wall); $\boldsymbol{n}_w$ is the outward normal to the wall; $\boldsymbol{u}_w$ is the moving wall velocity.

Thus, the near-wall cell coefficients are specified by the following equalities

$$a_P^{uu} = \mu_f S_f^* \left(1-\left(n_w^x\right)^2\right) \quad a_P^{uv} = \mu_f S_f^* n_w^x n_w^y \quad a_P^{uv} = \mu_f S_f^* n_w^x n_w^z,$$
$$a_P^{vu} = \mu_f S_f^* n_w^y n_w^x \quad a_P^{vv} = \mu_f S_f^* \left(1-\left(n_w^y\right)^2\right) \quad a_P^{vw} = \mu_f S_f^* n_w^y n_w^z, \tag{18}$$
$$a_P^{wu} = \mu_f S_f^* n_w^z n_w^x \quad a_P^{wv} = \mu_f S_f^* n_w^z n_w^y \quad a_P^{ww} = \mu_f S_f^* \left(1-\left(n_w^z\right)^2\right),$$
$$b_P^u = \varepsilon_P \mu_f S_f^* u_w^x \quad b_P^v = \varepsilon_P \mu_f S_f^* u_w^v \quad b_P^w = \varepsilon_P \mu_f S_f^* u_w^z.$$

This boundary condition implies no variations in the continuity equation coefficients.

3.5. Slip wall

The implementation of the slip wall boundary condition is similar to the implementation of the no-slip wall boundary condition with regard to zero tangential stress ($\boldsymbol{\tau}_w = 0$). The wall velocity is not set and equals the tangential velocity component in the near-boundary cell. Pressure and any other scalar quantity are equalized with the values of quantities in the near-boundary cell. This boundary condition requires accounting the pressure gradient in the momentum equation

$$a_P^{pu} = \varepsilon_P S_f^x \quad a_P^{pv} = \varepsilon_P S_f^y \quad a_P^{pw} = \varepsilon_P S_f^z. \tag{19}$$

As a result, the described discretization steps for equations (5) with the use of a completely implicit algorithm allow solving problems of compressible and incompressible flows in porous media with a varying porosity factor relative to the real velocity. The discretization using the completely implicit algorithm allows significantly reducing the total number of iterations and the total problem runtime, and such reduction is demonstrated below for benchmark problems.

4. Computational speedup

The general performance of the coupled algorithm depends on an important factor, such as the SLAE solution stage in the computational process. The application of Krylov's subspace iterative solvers [Saad, 2003] significantly increases the number of nested iterations (the number of iterations of SLAE solution) and hence the problem runtime, which may be several dozens of thousands, and in most cases leads to the solution divergence. First of all, this fact can be attributed to large enough conditioning numbers of the original SLAE [Darwish et al., 2009; Kozelkov et al., 2016c] and, hence, it seems reasonable to use multigrid solvers, a variety of which is available [Brandt, 1982; Stuben & Trottenberg, 1982]. In the present study the multigrid solver with the aggregative way of coarsening is used [Vanek et al., 1996]. It is described in detail for a parallel case in [Kozelkov et al., 2016c]. The studies of [Volkov et al., 2013, 2014b] describe in detail the choice of optimal parameters (such as the type of cycle, the number of cells for coarsening) for the given version.

To solve SLAEs, the coupled algorithm uses matrices consisting of blocks which size is 4×4, and the solution vector with four unknowns (6). The original scalar multigrid solver [Kozelkov et al., 2016c] is performed by increasing the size of linear arrays of matrix coefficients and replacing the scalar multiplication and division operations by the corresponding matrix function. The implementation of such approach allows solving SLAE with an arbitrary block size and it is of a great importance for multiphase flows, where the size of blocks depends on the number of phases [Moukalled et al., 2003].

The principal idea of the multigrid method is in the hierarchical structure and preserved sequence of embedded coarse SLAE matrices and operators of random transition from one SLAE matrix to another [Volkov et al., 2013, 2014b; Yvan, 2010]. The upper left block coefficient responsible for the pressure field equation is taken for the leading element to perform coarsening. The resultant number of coarsened SLAE matrices (levels) is random and depends on the coarsening parameters. The solution process starts with solving the coarsest SLAE (level 0) followed by the

obtained solution and residual interpolation to finer levels. Such iterative process allows reducing the number of internal iterations until the required solution accuracy is achieved.

Consider the main steps of the algebraic method using the classic SLAE solution problem

$$A_h x_h = b_h, \tag{20}$$

where A_h is the original matrix of size $n \times n$; x_h and b_h are the vector of unknowns and the right-hand side of the system of size *n*, respectively. The subscript *h* indicates that equations belong to a fine grid.

The operator *P* of interpolation from coarse grid H to fine grid h allows representing matrix A_H on the coarse grid in the form

$$A_H = RA_h P, \tag{21}$$

where $R = P^T$ is the operator of interpolation from a fine grid to a coarse grid. The correction step has the form

$$x_h^{new} = x_h^{old} + Pe_H. \tag{22}$$

The solution correction e_H is the exact solution to equation

$$A_H e_H = r_H, \tag{23}$$

where $r_H = Rr_h$ is the residual at the coarse level, and $r_h = b_h - A_h x_h^{old}$ is the residual at the finest level.

An iteration of the multigrid method using the solution correction scheme is the following sequence of steps.

1. Do n iterations of preliminarily smoothing the solution on grid h using the ILU method.
2. Residual $r^h = b^h - A_h x_{old}^h$ is calculated at the current level.
3. Approximate solution $A_H e^H = r^H$ is found on a coarse grid. For this purpose, γ cycles of the multigrid scheme are recursively fulfilled.
4. Correction e_H is interpolated onto a fine grid and the solution is corrected on the fine grid, $x_{new}^h = x_{old}^h + Pe^H$.
5. Do n iterations of finally smoothing the solution on the fine grid to suppress interpolation errors.

Different types of cycles are identified at each grid level depending on the number of recursive calls of solver, V-cycle and W-cycle take place with $\gamma = 1$ and $\gamma = 2$, respectively. If at each level one W-cycle and then one V-cycle are recursively called, F-cycle is performed. The use of one or another type of cycles (V-, W- or F-cycle) affects the convergence rate and solution stability [Volkov et al., 2013, 2014b].

Smoother is an important factor in the multigrid solver. In calculations ILU(0) method of [Pommerell & Fichtner, 1994] is used as a smoother. The multigrid solver in combination with ILU(0) smoother allows suppressing high-frequency and low-frequency errors with a maximum efficiency, while providing a high convergence rate [Darwish et al., 2009].

Consider the specific features of storing the block matrix of SLAE by the example of a problem with four cells (Figure 2).

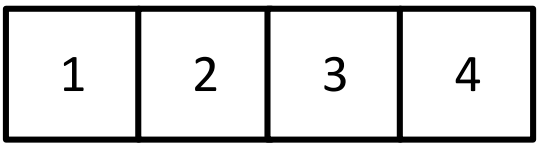

Figure 2. An example of computational model with four cells

Use of the described algorithm of discretization of the continuity and momentum equations gives SLAE of the block type with the corresponding matrix portrait

$$\begin{bmatrix} \begin{pmatrix} p_{11}^{pp} & p_{11}^{pu} & p_{11}^{pv} & p_{11}^{pw} \\ u_{11}^{up} & u_{11}^{uu} & u_{11}^{uv} & u_{11}^{uw} \\ v_{11}^{vp} & v_{11}^{vu} & v_{11}^{vv} & v_{11}^{vw} \\ w_{11}^{wp} & w_{11}^{wu} & w_{11}^{wv} & w_{11}^{ww} \end{pmatrix} & \begin{pmatrix} p_{12}^{pp} & p_{12}^{pu} & p_{12}^{pv} & p_{12}^{pw} \\ u_{12}^{up} & u_{12}^{uu} & u_{12}^{uv} & u_{12}^{uw} \\ v_{12}^{vp} & v_{12}^{vu} & v_{12}^{vv} & v_{12}^{vw} \\ w_{12}^{wp} & w_{12}^{wu} & w_{12}^{wv} & w_{12}^{ww} \end{pmatrix} & \begin{pmatrix} \dots \\ \dots \\ \dots \\ \dots \end{pmatrix} & \begin{pmatrix} \dots \\ \dots \\ \dots \\ \dots \end{pmatrix} \\ \begin{pmatrix} p_{21}^{pp} & p_{21}^{pu} & p_{21}^{pv} & p_{21}^{pw} \\ u_{21}^{up} & u_{21}^{uu} & u_{21}^{uv} & u_{21}^{uw} \\ v_{21}^{vp} & v_{21}^{vu} & v_{21}^{vv} & v_{21}^{vw} \\ w_{21}^{wp} & w_{21}^{wu} & w_{21}^{wv} & w_{21}^{ww} \end{pmatrix} & \begin{pmatrix} p_{22}^{pp} & p_{22}^{pu} & p_{22}^{pv} & p_{22}^{pw} \\ u_{22}^{up} & u_{22}^{uu} & u_{22}^{uv} & u_{22}^{uw} \\ v_{22}^{vp} & v_{22}^{vu} & v_{22}^{vv} & v_{22}^{vw} \\ w_{22}^{wp} & w_{22}^{wu} & w_{22}^{wv} & w_{22}^{ww} \end{pmatrix} & \begin{pmatrix} \dots \\ \dots \\ \dots \\ \dots \end{pmatrix} & \begin{pmatrix} \dots \\ \dots \\ \dots \\ \dots \end{pmatrix} \\ (\dots \quad \dots \quad \dots \quad \dots) & (\dots \quad \dots \quad \dots \quad \dots) & (\dots) & (\dots) \\ (\dots \quad \dots \quad \dots \quad \dots) & (\dots \quad \dots \quad \dots \quad \dots) & (\dots) & (\dots) \end{bmatrix} \times \begin{bmatrix} \begin{pmatrix} p_P \\ u_1 \\ v_1 \\ w_1 \end{pmatrix} \\ \begin{pmatrix} p_2 \\ u_2 \\ v_2 \\ w_2 \end{pmatrix} \\ (\dots) \\ (\dots) \end{bmatrix} = \begin{bmatrix} \begin{pmatrix} b_P^p \\ b_1^u \\ b_1^v \\ b_1^w \end{pmatrix} \\ \begin{pmatrix} b_2^p \\ b_2^u \\ b_2^v \\ b_2^w \end{pmatrix} \\ (\dots) \\ (\dots) \end{bmatrix}. \tag{24}$$

$$\begin{bmatrix} [\times] & [\times] & \cdot & \cdot \\ [\times] & [\times] & [\times] & \cdot \\ \cdot & [\times] & [\times] & [\times] \\ \cdot & \cdot & [\times] & [\times] \end{bmatrix}, \; where \; [\times] = \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}$$

The symbols [×] correspond to the blocks of cells which are coupled. The symbols • correspond to blocks of cells without coupling. The symbols × correspond to a non-zero element inside one block. Note that coefficients for velocity components with superscripts *uv*, *uw* and *vw* are non-zero for near-wall cells only. In all the rest cases these coefficients equal zero.

The aggregative multigrid solver allows using computational resources in a most optimal way from the point of view of both memory consumption and speed of computations.

5. Resuls and discussion

The results of the numerical simulation of flows in porous domains are presented. The proposed approach is implemented in the LOGOS code used to solve coupled 3D problems of the heat and mass transport with convection, aerodynamics and hydrodynamics on parallel computers [Lashkin et al., 2016; Kozelkov et al., 2016c]. The LOGOS code has been successfully verified and demonstrates good results for a series of different CFD problems, including the simulation of turbulent and unsteady flows [Kozelkov et al., 2016c, 2015a, 2016a; Volkov et al., 2014a; Kozelkov & Kurulin 2015], as well as geographical problems [Kozelkov et al., 2015b, 2015c, 2016b; Pelinovskii et al., 2016]. The multigrid method [Brandt, 1982; Volkov et al., 2013; Lashkin et al., 2016] is used for simulations. It ensures a significant speedup of the computational process and allows efficiently using several hundreds of computing cores.

5.1. A fluid flow through a porous material insert

A 2D flow in a plane-parallel channel of height H and length $L = 7H$ [Costa et al., 2004a] is considered. A rectangular insert of length $2H$ made of a homogeneous isotropic porous material is placed in the channel at a distance of $3H$ to the channel inlet. The channel length L has been

chosen sufficiently large to eliminate the impact of steady state flow regions on the final results. The domain of interest is illustrated by the Figure 3. The 2D problem formulation is considered, and the channel length is eliminated from the consideration by imposing the symmetric boundary conditions. Horizontal boundaries are rigid walls with adhesion.

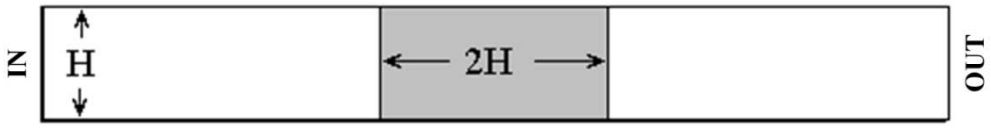

Figure 3. A computational domain

The dimensionless parameters of the problem are Reynolds number, $\mathrm{Re} = \dfrac{\rho U H}{\mu} = 1$, where U is the inlet velocity, and Darcy number, $Da = 10^{-3}$. These numbers are used to select dimensional quantities such as fluid properties and resistance factors for the porous body. The porosity factor ε is set equal to 1.

To solve the problem, three uniform meshes have been used. Their parameters are given in the Table 1.

Table 1. Computational meshes

No	Type of mesh	Typical streamwise size, m	Number of cells
1	Uniform	0.025	6400
2	Uniform	0.0125	25600
3	Uniform	0.00625	102400

Simulations are performed using the coupled and segregate solvers for the steady state problem. The specified minimum level of the residual is set to 10^{-6}. It is found from the expression

$$res_m = \sum_{i=1}^{N} \left(\sum_{f=nb(P)} \rho_f S_f \left(\mathbf{u}_f \cdot \mathbf{n}_f \right) \right), \tag{25}$$

where N is the total number of cells.

The convergence histories are presented in the Figure 4 for both solvers (with standard relaxation parameters).

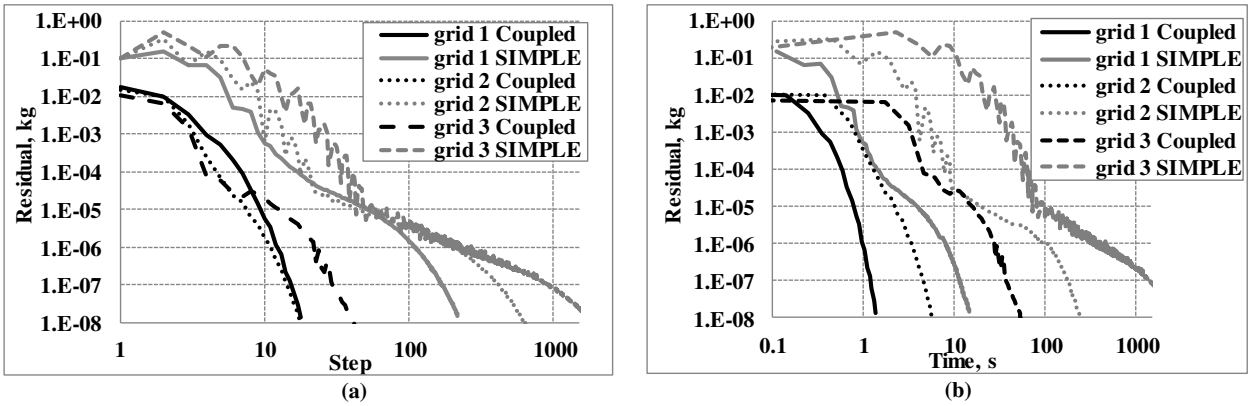

Figure 4. Residual of mass as a function of step size (a) and as a function of time (b)

Comparison of the results computed with the SIMPLE and coupled solvers for the given problem are summarized in the Table 2. The reported computational costs is measured as a number of steps required to reach the same level of residual for two computational solvers, SIMPLE solver and coupled solver.

Table 2. Computational results

No	Number of steps			Time, s		
	Coupled solver	SIMPLE solver	SIMPLE/ Coupled	Coupled solver	SIMPLE solver	SIMPLE/ Coupled
1	13	111	8.5	1.03	8.04	7.8
2	12	203	16.9	3.69	106.90	29.0
3	23	199	8.7	28.52	340.76	11.9

The results obtained show that the coupled algorithm on a fine mesh provides 17 times reduction of the total number of iterations and almost 29 times reduction of the wall-clock time of simulations.

The examination of results includes the comparison of velocity and pressure profiles along the centerline of the channel. The comparison of the calculated results and those from [Elnaz, 2012] is given in the Figure 5.

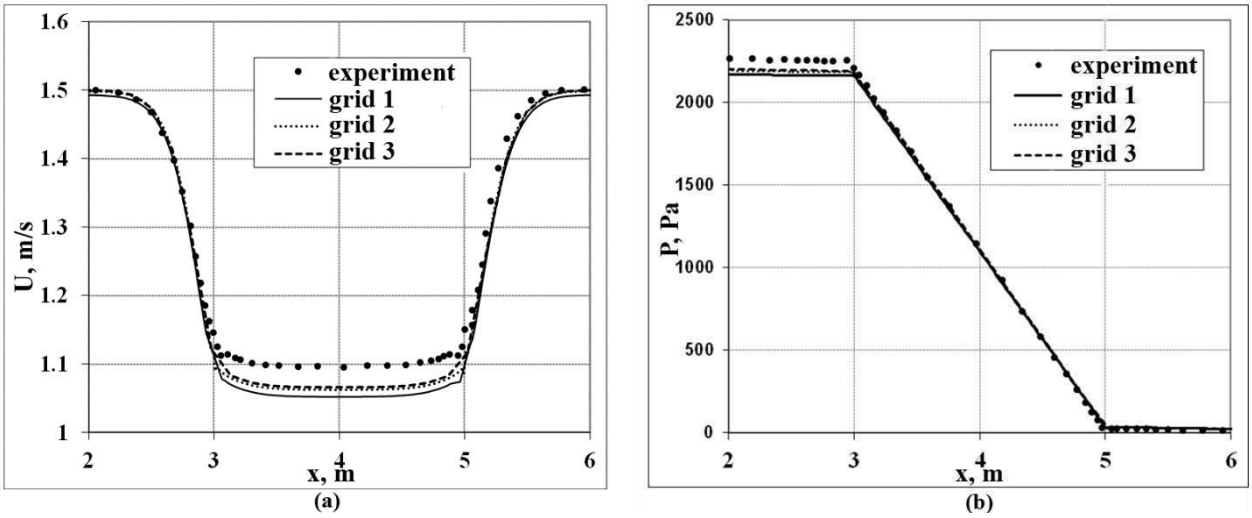

Figure 5. Velocity profiles (a) and pressure profiles (b) along the centerline

The obtained velocity and pressure-difference values are almost the same for all meshes. They are in a good agreement with the numerical results from [Costa et al., 2004a].

5.2. A fluid flow in a planar divided channel

The problem presented in [Costa et al., 2004a] describes incompressible viscous fluid flow in a plane-parallel channel of height $2H$ and length $L >> 2H$. The lower domain of height, H, is a porous medium. The channel length, L, is selected sufficiently large to eliminate the impact of the steady state flow regions on the final results. The physical region of interest is schematically represented in the Figure 6.

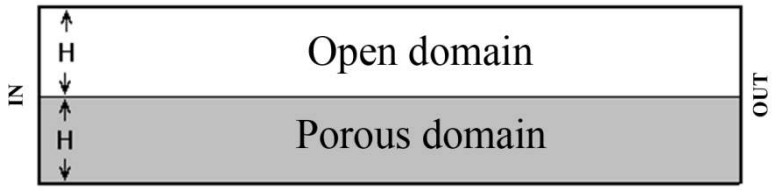

Figure 6. A computational domain

The dimensionless parameters for the given problem are Reynolds number in an open domain, $\mathrm{Re} = \frac{\rho UH}{\mu} = 1$, and Darcy number, $Da = 10^{-2}$. The porosity factor ε is set equal to 1.

To solve the problem, three uniform meshes, which parameters are given in the Table 3, are used.

Table 3. Computational meshes

No	Type of mesh	Typical size across the flow, m	Number of cells
1	Uniform	0.04	10000
2	Uniform	0.02	20000
3	Uniform	0.01	80000

Simulations were performed using the coupled and SIMPLE solvers for the steady state problem. The minimum level of residual is set to 10^{-6}. The iterative process convergence rates are shown in the Figure 7 for the coupled and segregate solvers (with standard relaxation parameters).

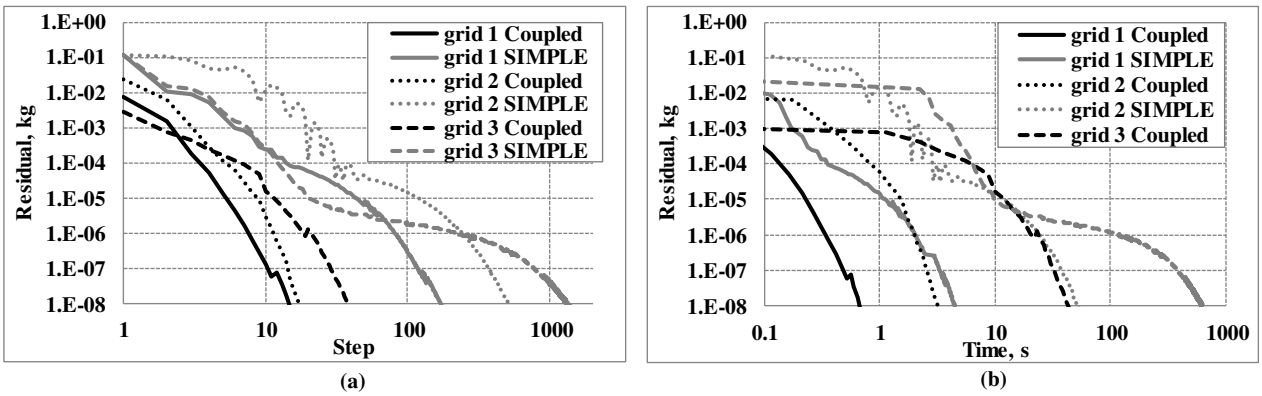

Figure 7. Residual of mass as a function of step size (a) and as a function of time (b)

The comparison of the results for the SIMPLE and coupled solvers for the given problem is given in the Table 4.

Table 4. Computational results

No	Number of steps			Time, s		
	Coupled solver	SIMPLE solver	SIMPLE/ Coupled	Coupled solver	SIMPLE solver	SIMPLE/ Coupled
1	8	82	10.25	0.36	2.04	5.67
2	12	241	20.08	2.12	24.44	11.53
3	19	218	11.47	21.85	114.17	5.22

The results obtained show that the coupled algorithm on a fine mesh provides 20 times reduction of the total number of iterations and 11 times reduction of the wall-clock time of simulations.

The results are compared with those from [Costa et al., 2004a] with respect to the horisontal velocity component profiles along the channel cross-section in the steady state flow region. Figure 8 illustrates the streamwise velocity profiles computed with different meshes in comparison with those from [Costa et al., 2004a]. The axis *x* shows the channel height, and the axis *y* shows the streamwise velocity.

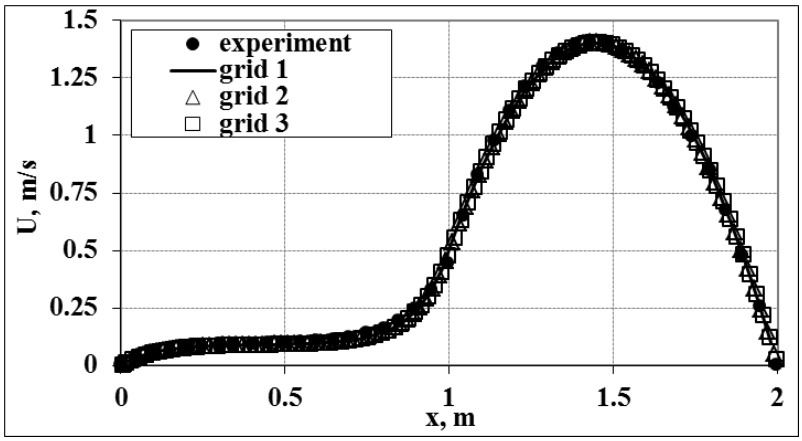

Figure 8. The streamwise velocity profiles along the central part of the channel

There is a good agreement of the computed results with results from [Costa et al., 2004a] for each of the three meshes.

5.3. A fluid flow through a cylindrical filter

The problem demonstrates a possibility of simulating a fluid flow in a symmetric cylindrical region occupied by a porous body. The problem of an air flow in a car engine, where filter may be represented as a cylinder, is an example. The cause of the main difficulty in solving problems of such kind is inconsistency between the resistance tensor coefficients specified for the local cylindrical and the global Cartesian frame of reference. The fact is that in the local system this tensor is represented by the diagonal part only, and in the global system it transforms and contains values in off-diagonal elements of the resistance tensor. The coupled algorithm developed in the study allows discretization of the original off-diagonal elements in a completely implicit manner.

The flow in a porous medium is set in the radial direction alone. This becomes possible owing to the resistance coefficients augmentation in the transversal and longitudinal directions relative to the radial direction. The diagonal resistance tensor in cylindrical coordinates is set so that the fluid flows in radial direction only. Such effect is possible, for example, with the following values of resistance tensors

$$\boldsymbol{P}_{local} = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 10000 & 0 \\ 0 & 0 & 10000 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 100 \end{bmatrix} \cdot \left| \boldsymbol{u}_{local} \right|. \tag{26}$$

The resistance coefficients have no significant effect on the flow pattern (they affect the pressure difference only), whereas of a great importance is 100 times difference between the coefficient values in the radial direction and their values in the transversal and longitudinal directions.

With the transition from the local system to the global one, the resistance tensor transforms according to the rule of transition from one coordinate system to another

$$\boldsymbol{P}_{global} = M \cdot P_{local} \cdot M \,, \tag{27}$$

where M is the transformation matrix. For example, for the given problem the transformation of tensor into the global coordinate system looks like

$$\boldsymbol{P}_{global} = \begin{bmatrix} P_{xx}\cdot\cos(\varphi)\cdot\cos(\varphi) + P_{yy}\cdot\sin(\varphi)\cdot\sin(\varphi) & \cos(\varphi)\cdot\sin(\varphi)\cdot(P_{xx}-P_{yy}) & 0 \\ \cos(\varphi)\cdot\sin(\varphi)\cdot(P_{xx}-P_{yy}) & P_{xx}\cdot\sin(\varphi)\cdot\sin(\varphi) + P_{yy}\cdot\cos(\varphi)\cdot\cos(\varphi) & 0 \\ 0 & 0 & P_{zz} \end{bmatrix}, \quad (28)$$

where φ is the angle of revolution in cylindrical coordinates. So, it has been demonstrated that off-diagonal coefficients do not equal zero, and the way of discretizing them affects the convergence rate.

The computational domain and mesh are presented in the Figure 9. A uniform radial mesh containing 315,000 cells is used.

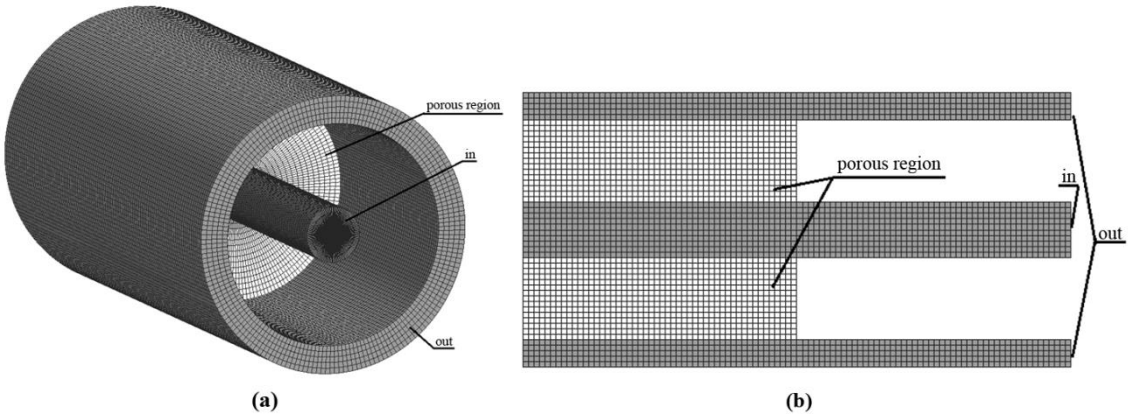

Figure 9. A computational domain (a) and its cross-section (b)

The mass flow rate is fixed at the inlet boundary, and the static pressure is fixed at the outlet boundary. The Reynolds number is $\mathrm{Re} = \dfrac{\rho U D}{\mu} = 1000$, where D is the cylinder diameter.

To solve the problem, two uniform meshes are used. Their parameters are given in the Table 5.

Table 5. Computational meshes

No	Type of mesh	Typical size, m	Number of cells
1	Uniform	0.02	125000
2	Uniform	0.01	500000

Simulations are performed using the coupled and SIMPLE solvers for the steady state problem. The minimum level of residual is set to 10^{-6}. The convergence histories of iterative process are presented in the Figure 10 for the coupled and segregate solvers (with standard relaxation parameters).

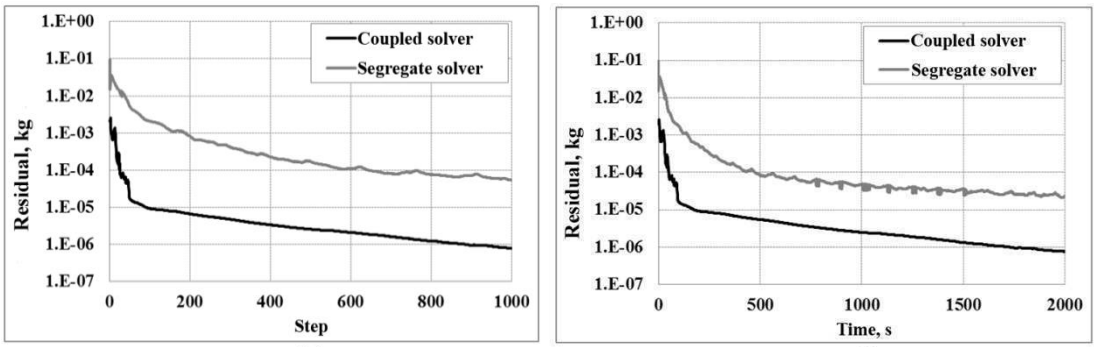

Figure 10. Residual of mass as a function of step size (a) and as a function of time (b)

The comparison of results based on the coupled and segregate solvers for the given problem are presented in the Table 6.

Table 6. Computational results

No	Number of steps			Time, s		
	Coupled solver	SIMPLE solver	SIMPLE/ Coupled	Coupled solver	SIMPLE solver	SIMPLE/ Coupled
1	913	7421	8.1	575.6	1696.7	2.9
2	2123	10637	5.01	701.3	5943.3	8.5

The results obtained demonstrate that the coupled algorithm on a fine mesh provides 8 times reduction in the number of iterations on a coarse mesh, and 8.5 times reduction of wall-clock time of simulations on a fine mesh.

The pressure differences at the input and output are the same and equal 970 Pa for the coupled and SIMPLE solvers.

6. Conclusions

The coupled computational algorithm for Brinkman–Forchheimer equations to simulate flows in anisotropic porous media is developed. In contrast to the SIMPLE algorithm, the coupled algorithm couples the velocity and pressure in the algebraic form in a completely implicit manner. Coupling is performed owing to the implicit pressure gradient and mass flux terms in the momentum and continuity equations, respectively. When solving Brinkman–Forchheimer equations with the coupled algorithm, it is possible to discretize implicitly off-diagonal coefficients of the porous resistance tensor in the right-hand side of the momentum equation. The numerical calculations demonstrate that such implicit discretization allows reducing the number of nonlinear iterations in comparison with the SIMPLE algorithm and the wall-clock time of the problem solution. The time gain is 4 to 30 and depends on the mesh size and problem type. For the problem of a flow through a cylindrical filter the time gain is 4 in comparison with the SIMPLE algorithm.

Comparison of a SIMPLE algorithm and the coupled solver has been made for some benchmark cases including fluid flow through a porous material insert, a fluid flow in a planar divided channel and a fluid flow through a cylindrical filter. On the one side, these test cases are typical for flow problems appearing in nuclear industry. On the other side, the test cases chosen are simple to avoid influence of other factors on comparison of two solvers, SIMPLE solver and coupled solver. Although the robustness of the developed numerical methods was only demonstrated for some simple benchmark flow cases through a porous media, the algorithm can easily be extended and applied to more complex flow types and geometries. The main motivation

in this study was to accurately predict the flow through porous media based on Brinkman–Forchheimer equations in simple geometry.

Acknowledgments

This work was partially supported by the Russian Foundation for Basic Research (project 16-01-00267) and the RF President's Grant for the governmental support of leading scientific schools of the Russian Federation (project NSH-6637.2016.5).

References

G.N. Kolpakov, O.V. Selivanikova (2009). Designs of fuels elements, channels, and cores of power reactors. Tomsk, Tomsk Technological Untiversity, 118 p.

M.A. Bhutta, N. Hayat, M.H. Bashir, A.R. Khan, K.N. Ahmad, S.Khan (2012). CFD applications in various heat exchangers designs: a review. Applied Thermal Engineering, 32, 1–12.

E. Ozden, I. Tari (2010). Shell side CFD analysis of a small shell-and-tube heat exchanger. Energy Conversion and Management, 51, 1004–1014.

S. Malavasi, G. Messa, U. Fratino, A. Pagano (2012). On the pressure losses through perforated plates. Flow Measurement and Instrumentation, 28, 57–66.

E. Ozahi (2015). An analysis of the pressure loss through perforated plates at moderate Reynolds numbers in turbulent flow regime. Flow Measurement and Instrumentation, 43, 6–13.

S. Elnaz (2012). CFD and experimental analysis of diesel particulate filter. Chalmers University of Technology. MSc Thesis in Solid and Fluid Mechanics.

V.F. Costa, L.A. Oliveira, B.R. Baliga, A.C.M. Sousa (2004a). Simulation of coupled flows in adjacent porous and open domains using a control-volume finite-element method. Numerical Heat Transfer, 45, 675–697.

S. Ergun (1952). Fluid flow through packed columns. Chemical Engineering Progress, 48(2), 89–94.

A.E. Scheidegger (1974). The physics of flow through porous media. Toronto, University of Toronto Press, 353 p.

R. Collins (1961). Flow of fluids through porous material. New York, Reinhold Publishing Corporation, 270 p.

V. Reichenberger, H. Jakobs, P. Bastian, R. Helmig (2006). A mixed-dimensional finite volume method for two-phase flow in fractured porous media. Advances in Water Resources, 29, 1020–1036.

M. Chandesris, D. Jamet (2006). Boundary conditions at a planar fluid–porous interface for a Poiseuille flow. International Journal of Heat and Mass Transfer, 49, 2137–2150.

V. Costa, M. Oliveira, A. Sousa (2004b). Numerical simulation of non-Darcian flows through spaces partially filled with a porous medium. Computers and Structures, 82, 1535–1541.

V. Kulkarni, G. Quadir, K. Seetharamu, P. Aswathanarayana, A. Aziz (2004). Characteristic based split algorithm used in underfilling encapsulation process. Proceedings of the Fourth European Congress on Computational Methods in Applied Sciences and Engineering, 24–28 July 2004, Jyväskylä, Finland.

U. Shavit, R. Rosenzweig, S. Assouline (2003). Free flow at the interface of porous surfaces: a generalization of the Taylor brush configuration. Transport in Porous Media, 1835, 1–16.

M. Kaviany (1991). Principles of heat transfer in porous media. New York, Springer-Verlag, 626 p.

J.H. Ferziger, M. Peric (2002). Computational methods for fluid dynamics. Berlin, Springer, 423 p.

M. Darwish, I. Sraj, F. Moukalled (2009). A coupled finite volume solver for the solution of incompressible flows on unstructured grids. Journal of Computational Physics, 228, 180–201.

M. Darwish, F. Moukalled (2000). Unified formulation of the segregated class of algorithms for fluid flow at all speeds. Numerical Heat Transfer, 37(1), 103–139.

S. Zhang, X. Zhao, S. Bayyuk (2014). Generalized formulation for the Rhie–Chow interpolation. Journal of Computational Physics, 258, 880–914.

K. Shterev, S. Stefanov (2010). Pressure based finite volume method for calculation of compressible viscous gas flows. Journal of Computational Physics, 229(2), 461–480.

L. Mangani (2010). A coupled finite volume solver for the solution of laminar/turbulent incompressible and compressible flows. Proceedings of the Fifth OpenFOAM Workshop, 22–24 June 2010, Gothenburg, Sweden.

M. Emans, M. Liebmann (2013). Velocity-pressure coupling on GPU. FB-Report, 19 p.

Z.J. Chen, A.J. Przekwas (2010). A coupled pressure-based computational method for incompressible/compressible flows. Journal of Computational Physics, 229, 9150–9165.

D.A. Nield, A. Bejan (2013). Convection in porous media. New York, Springer Science, 778 p.

V. Prasad, F.A. Kulacki, M. Keyhani (1985). Natural convection in porous media. Fluid Mechanics, 150, 89–119.

H. Brinkman (1952). The viscosity of concentrated suspensions and solutions. Journal of Chemical Physics, 4(20), 571–584.

J.C. Ward (1964). Turbulent flow in porous media. ASCE Journal of Hydraulics Division, 90 (HY5), 1–12.

H. Jasak (1996). Error analysis and estimation for the finite volume method with applications to fluid flow. London, 394 p.

A.A. Samarsky, A.V. Gulin (1989). Numerical Methods. Moscow, Nauka, 432 p.

B.P. Muljadi, M.J. Blunt, A.Q. Raeini, B. Bijeljic (2016). The impact of porous media heterogeneity on non-Darcy flow behaviour from pore-scale simulation. Advances in Water Resources, 95, 329–340.

A. Kozelkov, V. Kurulin, V. Emelyanov, E. Tyatyushkina, K. Volkov (2016a). Comparison of convective flux discretization schemes in detached-eddy simulation of turbulent flows on unstructured meshes. Journal of Scientific Computing, 67, 176–191.

K.N. Volkov, Yu.N. Deryugin, A.S. Kozelkov, V.N. Emelyanov, I.V. Teterina (2014a). Difference schemes in gas dynamics problems on unctructured grids. Moscow, Fizmatlit, 416 p.

A.S. Kozelkov, V.V. Kurulin (2015). Eddy resolving numerical scheme for simulation of turbulent incompressible flows. Computational Mathematics and Mathematical Physics, 55(7), 1255–1266.

A.S. Kozelkov, O.L. Krutyakova, A.A. Kurkin, V.V. Kurulin, E.S. Tyatyushkina (2015a). Zonal RANS–LES approach based on an algebraic Reynolds stress model. Fluid Dynamics, 50(5), 621–628.

C. Rhie, W. Chow (1983). A numerical study of the turbulent flow past an isolated airfoil with trailing edge separation. AIAA Journal, 21, 1525–1532.

Y. Saad (2003). Iterative methods for sparse linear systems. Minneapolis, SIAM, 568 p.

A.S. Kozelkov, V.V. Kurulin, S.V. Lashkin, R.M. Shagaliev, A.V. Yalozo (2016c). Investigation of supercomputer capabilities for the scalable numerical simulation of computational fluid dynamics problems in industrial applications. Computational Mathematics and Mathematical Physics, 56(8), 1506–1516.

A. Brandt (1982). Guide to multigrid development. Lectures Notes in Mathematics, 960, 220–312.

K. Stuben, U. Trottenberg (1982). Multigrid methods: fundamental algorithms, model problem analysis and applications. Berlin, Springer, 176 p.

P. Vanek, J. Mandel, M. Brezina (1996). Algebraic multigrid based on smoothed aggregation for second and fourth order problems. Computing, 1(56), 179–196.

K.N. Volkov, Yu.N. Deryugin, V.N. Emelyanov, A.G. Karpenko, A.S. Kozelkov, I.V. Teterina (2013). Acceleration methods for fluid dynamics simulation on unstructured grids. Moscow, Fizmatlit, 536 p.

K.N. Volkov, Yu.N. Deryugin, V.N. Emelyanov, A.S. Kozelkov, I.V. Teterina (2014b). An algebraic method in comptational physics problems. Computational Methods and Programming, 15, 183–200.

F. Moukalled, M. Darwish, B. Sekar (2003). A pressure-based algorithm for multi-phase flow at all speeds. Journal of Computational Physics, 190, 550–571.

N. Yvan (2010). An aggregation-based algebraic multigrid method. Electronic Transactions on Numerical Analysis, 37, 123–146.

C. Pommerell, W. Fichtner (1994). Memory aspects and performance of iterative solvers. SIAM Journal of Scientific and Statistical Computing, 15, 460–473.

A.S. Kozelkov, A.A. Kurkin, E.N. Pelinovskii, V.V. Kurulin (2015b). Modeling of the cosmogenic tsunami within the framework of the Navier–Stokes equations with sources of different types. Fluid Dynamics, 50(2),306–313.

A.S. Kozelkov, A.A. Kurkin, E.N. Pelinovskii, V.V. Kurulin, E.S. Tyatyushkina (2015b). Modeling the disturbances in the lake Chebarkul caused by the fall of the meteorite in 2013. Fluid Dynamics, 50(6), 828–840.

A.S. Kozelkov, A.A. Kurkin, E.N. Pelinovskii (2016b). Effect of the angle of water entry of a body on the generated wave heights. Fluid Dynamics, 51(2), 288–298.

S.V. Lashkin, A.S. Kozelkov, D.P. Meleshkina, A.V. Yalozo, N.V. Tarasova (2016). Numerical simulation ofviscous incompressible flow by a segregated and coupled SIMPLE-type algorithm. Mathematical Modeling, 28(6), 64–76.

E. Pelinovsky, A. Kozelkov, A. Kurkin (2016). Huge waves of meteorite origin. Geophysical Research Abstracts, 18, EGU2016-1213.

A.S. Kozelkov, A.A. Kurkin, E.N. Pelinovsky, E.S. Tyatyushkina, V.V. Kurulin, N.V. Tarasova (2016). Lands lide-type tsunami modelling based on the Navier–Stokes equations. Science of Tsunami Hazards. 35(3), 106–144.

P. Wesseling (1992). An introduction to multigrid methods. New York, Willey.

K. Stuben (2001). A review of algebraic multigrid. Journal of Computational and Applied Mathematics, 128(1-2), 281–309.

D.J. Mavriplis (2002) An assessment of linear versus nonlinear multigrid methods for unstructured mesh solvers. Journal of Computational Physics, 175(1), 302–325.

N. Alkishriwi, M. Meinke, W. Schroder (2006). A large-eddy simulation method for low Mach number flows using preconditioning and multigrid. Computers and Fluids, 35(10), 1126–1136.
D.J. Mavriplis (2007). Unstructured mesh discretizations and solvers for computational aerodynamics. AIAA Paper, 2007-3955.

J.S. Cagnone, K. Sermeus, S.K. Nadarajah, E. Laurendeau (2011). Implicit multigrid schemes for challenging aerodynamic simulations on block-structured grids. Computers and Fluids, 44(1), 314–327.

S. Langer (2013) Application of a line implicit method to fully coupled system of equations for turbulent flow problems. International Journal of Computational Fluid Dynamics, 27, 131–150.

W. Sun, S. Jiang, K. Xu (2017). An Implicit unified gas kinetic scheme for radiative transfer with equilibrium and non-equilibrium diffusive limits. Communications in Computational

Physics, 22(4), . 889–912.