Engineering Analysis with Boundary Elements 99 (2019) 222–232

Contents lists available at ScienceDirect

Engineering Analysis with Boundary Elements

journal homepage: www.elsevier.com/locate/enganabound

Fast boundary-domain integral method for heat transfer simulations

J. Tibaut*, J. Ravnik

Faculty of Mechanical Engineering, University of Maribor, Smetanova street 17, Maribor SI-2000, Slovenia

ARTICLE INFO

Keywords:
Modified Helmholtz equation
Boundary-Domain Integral Method
Velocity–vorticity formulation
Adaptive Cross Approximation
Fluid flow
Heat transfer

ABSTRACT

In this paper, we present a method to decrease the computational cost of the Boundary-Domain Integral Method. We focus on the solution of the velocity–vorticity formulation of the Navier–Stokes equation for incompressible 3D fluid flow. The objective is to accelerate the solution of the boundary vorticity values. In order to reduce the computational cost of the Boundary-Domain Integral Method, we employ the Adaptive Cross Approximation algorithm in combination with the hierarchical matrix structure. The hierarchical matrix structure enables higher compression rates when individual matrix parts are approximated by low-rank matrices. We use the fundamental solution of the modified Helmholtz equation to improve further the approximation accuracy when solving the boundary vorticity values.

The developed algorithm is used to simulate natural convection in a closed cavity. By comparing the simulation results with a published benchmark, we are able to assess the influence of the approximation techniques and give a recommendation on parameter values that lead to optimal compression characteristics when simulating the heat and mass transfer processes.

1. Introduction

Even though we are witnessing exponential growth in computing power, the need to develop efficient numerical algorithms is always present. In this study, we propose an algorithm to accelerate the Boundary-Domain Integral Method (BDIM) and test the new method by performing a number of fluid flow and heat transfer simulations.

Fluid flow is governed by a system of partial differential equations. Different numerical methods were introduced in order to solve such systems. The most widely used method in fluid mechanics is the Finite Volume Method (FVM). The FVM is based on the discretization of the domain.

An alternative approach is the Boundary Element Method (BEM). It is based on the discretization of only the boundary and the use of a fundamental solution of the underlying problem. Several researchers have worked on using BEM for simulation of transport phenomena. Wang and Ang [1] implemented the boundary element method solve a steady-state advective-diffusion-reaction equation. Ghadimi et al. [2] solved Poisson's equation by employing an analytical boundary element integration. The boundary element method is widely used in heat transfer problems. Li et al. [3] used the Galerkin boundary element to perform a steady heat conduction analysis. Liao and Chwang [4] used BEM for the solution of unsteady non-linear heat transfer problems. Yu et al. [5] presented a Radial Integral Boundary Element Method (RIBEM) with a precise integration method to solve transient heat conduction problems with variable thermal conduction. Recently Cui et al. [6] used the same method for the solution of transient heat conduction problems with heat sources and variable thermal conduction and in [7] the same authors solved a heat conductivity problem. On the other hand Yang et al. [8] used the virtual boundary element method in conjunction with the gradient algorithm to solve a three-dimensional inverse heat conduction problem.

In this paper, we develop an alternative approach, which is based on the Boundary-Domain Integral Method. We develop the fast BDIM to solve a non-linear heat transfer problem. The BDIM is a numerical method, which is based on the Boundary Element Method. Ravnik et al. [9] proposed BDIM for simulation of 3D flow and heat transfer problems. They used the Laplace fundamental solution to develop the algorithm. In this work, we propose to use the modified Helmholtz fundamental solution, which was used for the acceleration of the BDIM algorithm.

The Boundary-Domain Integral Method is based on the application of Green's second theorem. By using the fundamental solution of the governing partial differential equation, a boundary-domain integral equation may be derived [10]. Our main objective is to solve the velocity–vorticity formulation of the Navier–Stokes equations. In this formulation, the continuity equation is reformed into a kinematics equation and the momentum conservation equation is recast into a vorticity equation. Accounting for Green's second identity, we can write the integral form of the two equations. Škerget and Kuhn [11] have observed that this form of the Navier–Stokes equations can be solved efficiently by the BDIM. The first advantage of using the velocity–vorticity formula-

* Corresponding author.
E-mail addresses: jan.tibaut@um.si (J. Tibaut), jure.ravnik@um.si (J. Ravnik).

https://doi.org/10.1016/j.enganabound.2018.12.003
Received 5 October 2018; Received in revised form 14 November 2018; Accepted 5 December 2018

tion is the computational decoupling of the kinematics and kinetics fluid motion from pressure computation. Since the pressure does not appear explicitly in the field functions, the well-known difficulty with the computation of the boundary pressure values in incompressible fluid motion is avoided. Next, the boundary vorticity is computed directly, and not through the use of some approximation formula. Lastly, the convection dominated fluid motion suffers from numerical instability. In BDIM, the problem can be avoided by the use of Green's functions of appropriate linear differential operators [12]. However, the computational cost in memory and CPU time is of the order $O(N^2)$ and, therefore, the method is very costly. Thus, in this work, we develop a fast Boundary-Domain Integral Method to reduce the computational cost and storage requirements.

The fast Boundary-Domain Integral Method is a term that we use for the Boundary Domain Integral Methods that are faster than the conventional BDIM. Since the BDIM was developed from the Boundary Element Method, the techniques employed to accelerate the method are very similar to the ones used with the Boundary Element Method. Several acceleration approaches were introduced. Kalman [13] considered using the Singular Value Decomposition method (SVD) to approximate full matrices with a low-rank matrix approximation. The SVD is an efficient compression technique, however its computational cost is very high. Computational cost is lower with other approximation methods, such as the Fast Multipole Method (FMM) [14], wavelet transform [15,16] and the Adaptive Cross Approximation (ACA) [17].

In this paper, we focus on the implementation of the Adaptive Cross Approximation method in order to accelerate the Boundary-Domain Integral Method. The ACA algorithm is algebraic, and can be used without any information on the origin of matrix entries. Several improvements have been proposed, the extended Adaptive Cross Approximation (ACA+) [18] and the Hybrid Cross Approximation (HCA) [19]. The ACA+ has an optimized pivoting procedure, and the HCA uses a degenerate kernel function in order to interpolate between the elements in the matrix. However, the ACA+ and HCA are very complex approximation techniques. The simplicity of the Adaptive Cross Approximation enables the algorithm to be used in different engineering applications. For example, it has been used to solve the eddy current problem by Smajic et al. [20]. Schröder et al. [21] used the Adaptive Cross Approximation to solve the current distribution in an electromagnetic field. On the other hand, Grytsenko and Galybin [22] used the Adaptive Cross Approximation to solve a large number of cracks on a plate with the singular integral method, and Maerton [23] employed ACA to solve a 3D elasticity problem. Kurz et al. [24] used the ACA for the acceleration of the Boundary Element Method, and Van et al. [25] implemented the ACA to solve the electromagnetic field distribution on different applications. Recently, Campos et al. [26] implemented the Adaptive Cross Approximation algorithm to solve potential problems with non-uniform boundary conditions. The fast BEM is considered an alternative for the FEM methods used for magnetic field computations [27].

Hackbusch [28] introduced a recursive hierarchical decomposition of the domain in order to write the boundary element matrix using such a $\mathcal{H}$-structure. Each matrix part is tested using an admissibility condition, to assess which parts may be approximated using a low-rank matrix approximation [29]. A more extended version of the $\mathcal{H}$-structure was introduced by Börm [30]. He introduced the $\mathcal{H}^2$ matrix formulation, which further reduces the number of arithmetical operations in the matrix-vector multiplication and the needed memory storing space.

Tibaut et al. [31] used the elliptic fundamental solution in combination with the Adaptive Cross Approximation, for the simulation of a lid-driven cavity test case and discovered poor compression characteristics. In order to avoid the use of the the elliptic fundamental solution we propose to use the false transient approach, that introduces an artificial time derivative into the steady-state formulation of the kinematics equation and enables the use of the modified Helmholtz fundamental solution. Due to the fact that the shape of the modified Helmholtz fundamental solution depends on the time step, we are able to choose a shape which yields improved approximation characteristics.

The false transient approach has been used by other authors as well. Mallinson and Davis [32] introduced the false transient approach for the solution of a coupled elliptic equation. Guj and Stella [33] used the false transient in order to solve a lid-driven cavity test case with the finite difference scheme. The proposed procedure has shown great promise among other methods. Behnia et al. [34] used the same approach so simulate a buoyancy-driven flow with thermocapillary convection. On the other hand, Škerget and Rek [35], have observed the impact of the modified Helmholtz fundamental solution on the solution of the fluid flow in a lid-driven cavity at different Reynolds numbers. Recently, Kocutar et al. [36] used the same function in order to solve an unsteady turbulent fluid flow with a hybrid turbulent model.

The source of fluid movement in natural convection type problems are the changes in fluid density, which are caused by temperature or concentration differences. Natural convection appears in nature as wind or a sea current. It also appears in a variety of different engineering applications. In the past, natural convection was studied in different forms for different problems. We have focused on the natural convection in a closed cavity. Phillips [37] has performed a variety of simulations. He solved the problem of numerical instability with an automatic method that determines the step size. Quere and Aliziary de Roquefort [38] performed a two-dimensional simulation of the natural convection in a closed cavity. They solved the problem of numerical instability, with the use of the Chebyshev polynomials and the second-order time-stepping scheme. However, the time integration was only conditionally stable. This resulted in a long computing time. Tric et al. [39] observed the structures of the three-dimensional natural convection from an accurate numerical solution. They changed the Rayleigh number in the range of 10^3 to 10^7 and observed the influence of the mesh density on the solution of the flow. Markatos and Pericleous [40] observed the laminar and turbulent natural convection in a two-dimensional closed cavity. Their solver was built on the Finite Volume Method. They observed different flow structures that appear at Rayleigh numbers 10^3 to 10^{16}.

The aim of this work is to develop an algorithm that integrates the Adaptive Cross Approximation method with the hierarchical matrix structure and the modified Helmholtz fundamental solution into the Boundary-Domain Integral Method fluid flow solver. To accelerate the solution of the kinematics equation, we present a modification of the kinematics equation from the elliptic partial differential to a parabolic partial differential equation. The newly developed algorithm is assessed in terms of accuracy and effectiveness.

2. Velocity–vorticity formulation

In the present work, we consider an incompressible, Newtonian, and laminar fluid flow. The velocity–vorticity formulation of the Navier–Stokes equations is used. Let $\vec{v}$ be the velocity and let $\vec{\omega} = \vec{\nabla} \times \vec{v}$ be the vorticity of the fluid flow. The continuity equation can be reformulated into the following form:

$$\nabla^2 \vec{v} + \vec{\nabla} \times \vec{\omega} = 0. \tag{1}$$

Eq. (1) is the kinematics equation, and represents a connection between the velocity and vorticity vector field. In order to accelerate the convergence and the stability of the coupled velocity–vorticity field, the false transient approach is applied to Eq. (1):

$$\nabla^2 \vec{v} - \frac{1}{\alpha} \frac{\partial \vec{v}}{\partial t} + \vec{\nabla} \times \vec{\omega} = 0, \tag{2}$$

where $\alpha[\frac{\mathrm{m}^2}{\mathrm{s}}]$ is a relaxation parameter. The false transient kinematics equation reverts back to the kinematics equation only at steady state ($t \to \infty$), when the artificial time derivative term vanishes. When a finite difference approximation is used for the time derivative, a modified Helmholtz type equation emerges, for which the modified Helmholtz fundamental solution is known. By using a finite difference approximation of the time derivative $\Delta t = t - t_{F-1}$, where Δt is the time step, we can write the inhomogeneous modified Helmholtz kinematics

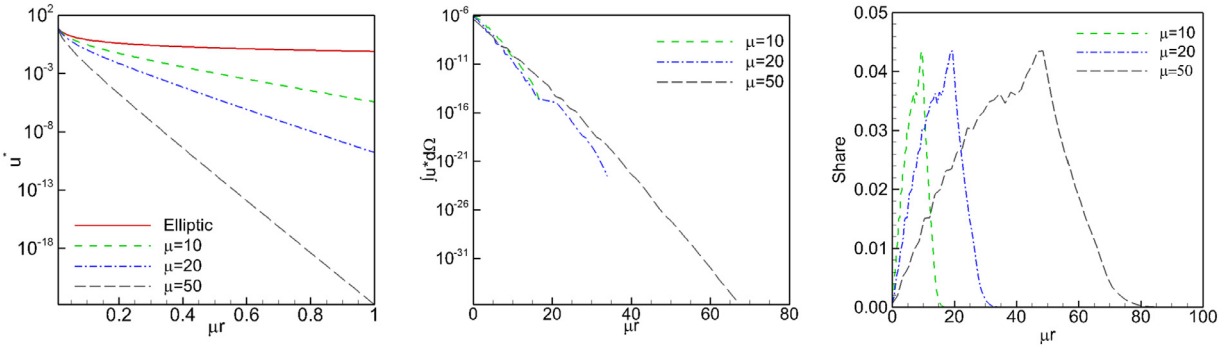

Fig. 1. The shape of the modified Helmholtz fundamental solution is shown in the left panel at different values of the μ parameter and distances between the collocation and field points r. When $\mu \to 0$ the Helmholtz fundamental solution limits towards the elliptic fundamental solution. In the central panel the absolute value of matrix entries is shown. In the right panel we present the share of mesh nodes that is at a distance μr. The mesh density was 25^3.

equation:

$$\left(\nabla^2 - \frac{1}{\alpha \Delta t}\right)\vec{v} = \frac{\vec{v}_{F-1}}{\alpha \Delta t} - \vec{\nabla} \times \vec{\omega}. \tag{3}$$

In our algorithm, we solve Eq. (3) for the boundary vorticity values and for the domain velocity values. The fluid movement is governed by the vorticity transport equation written in the non-dimensional form:

$$(\vec{v} \cdot \vec{\nabla})\vec{\omega} = (\vec{\omega} \cdot \vec{\nabla})\vec{v} + Pr \Delta \vec{\omega} - Pr\ Ra\ \vec{\nabla} \times \theta \vec{g}, \tag{4}$$

where $\theta = (T - T_\infty)/(T_w - T_\infty)$ is the non-dimensional temperature, T_w is the wall temperature and T_∞ the fluid temperature, $Ra = \frac{g\beta(T-T_\infty)l^3}{\gamma \nu}$ is the Rayleigh number, g is gravitational acceleration, l is the characteristic length scale, ν kinematic fluid viscosity, β thermal expansion, $\gamma = \frac{k}{\rho c_p}$ is the thermal diffusivity and $Pr = \frac{\nu}{\gamma}$ is the Prandtl number. The fluid properties are constant. Steady state formulation of the equation is given, as we consider only steady flows in this paper. The vorticity transport equation is employed to solve the domain vorticity of the flow. In order to solve the temperature field function θ, we consider the steady energy conservation equation:

$$(\vec{v} \cdot \vec{\nabla})\theta = \gamma \nabla^2 \theta. \tag{5}$$

In the following, we present a fast implementation of the Boundary-Domain Integral Method for the solution of the nonlinear system of Eqs. (3)–(5). A domain decomposition approach is used to solve the vorticity transport equation and the energy equation. However, Eq. (3) cannot be solved with the domain decomposition method. This is because of the Biot–Sawart law, that states the whole domain must be considered in order to keep the vorticity field divergence free [41]. Thus, we developed an algorithm to accelerate the calculation of the unknown boundary vorticity by using the modified Helmholtz kinematics equation.

2.1. The modified Helmholtz fundamental solution

Let u^* be the fundamental solution of the modified Helmholtz equation:

$$\left(\nabla^2 - \mu^2\right)u^* + \delta(\vec{\xi}, \vec{r}) = 0, \tag{6}$$

where $\delta(\vec{r}, \vec{\xi})$ is the Kronecker delta, $\vec{r}$ is the position vector of a point in the domain and $\vec{\xi}$ is the position of a source point. The modified Helmholtz fundamental solution is of the form, $u^* = e^{-\mu r}/(4\pi r)$, where r is the absolute distance between the domain point and the source point and μ is a constant. When the kinematics equation is considered $\mu = \frac{1}{\sqrt{\alpha\ \Delta t}}$ and, thus, the time step Δt and the relaxation parameter α determine the shape of the fundamental solution. In the case of large time steps, $\Delta t \to \infty$ the parameter $\mu = \frac{1}{\sqrt{\alpha \Delta t}} \to 0$ and the modified Helmholtz fundamental solution reverts into the elliptic fundamental solution $u^* = 1/(4\pi r)$. On the other hand, for short time steps, the modified Helmholtz fundamental solution becomes more local, diminishing quickly with the growing distance away from the source point. In Fig. 1 (left panel) we illustrate the shape of fundamental solutions at different μ values and compare it to the elliptic fundamental solution exposing the local character of the modified Helmholtz fundamental solution at large μ. In the right panel of Fig. 1 we present the share of matrix elements, which are calculated by integration at the distance μr. We can observe that at large μ values most of the matrix elements are calculated with a fundamental solution, which is very small, and the resulting matrix elements are also small (central panel in Fig. 1). Thus, we can expect more efficient approximation when using high μ values.

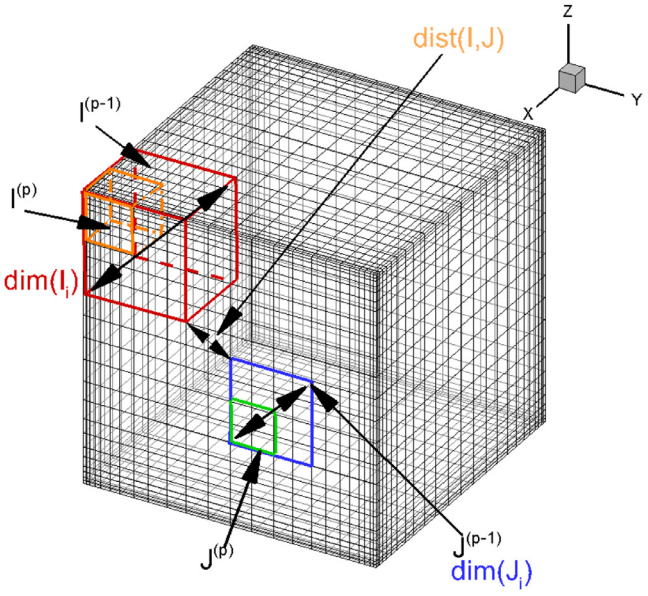

Fig. 2. Illustration of a mesh with 25^3 nodes is shown. Nodes are compressed towards the walls using a geometrical series. The domain and boundary clusters of elements are shown along with the definition of cluster size.

2.2. Integral formulation of the modified Helmholtz kinematics equation

Let us consider a domain Ω with a position vector $\vec{r} \in \mathbb{R}^3$. The boundary of the domain is $\Gamma = \partial\Omega$. The integral formulation of the modified Helmholtz kinematics equation (3) can be formed with the implementation of Green's second theorem. The following integral form, without the derivatives of the velocity and vorticity vector field, can be written:

$$c(\vec{\xi})\vec{v}(\vec{\xi}) + \int_\Gamma \vec{v}(\vec{n} \cdot \vec{\nabla})u^* d\Gamma = \int_\Gamma \vec{v} \times [(\vec{n} \times \vec{\nabla})u^*] d\Gamma + \int_\Omega (\vec{\omega} \times \vec{\nabla}u^*) d\Omega + \mu^2 \int_\Omega u^* \vec{v}_{F-1} d\Omega, \tag{7}$$

where $\vec{v}_{F-1}$ is the velocity of the previous time step, u^* is the modified Helmholtz fundamental solution and the parameter $\mu^2 = \frac{1}{\alpha\Delta t}$. Eq. (7) is the boundary-domain integral formulation of the domain-velocity modified Helmholtz kinematics equation. In order to use the modified Helmholtz kinematics equation to obtain the boundary vorticity values, we must rewrite Eq. (7) into a tangential form. This is done by multiplying the system with a normal vector of the source point $\vec{n}(\vec{\xi})$, as was proposed by Škerget et al. [41]. The following equation is obtained:

$$\begin{aligned} c(\vec{\xi})\vec{n}(\vec{\xi})\times\vec{v}(\vec{\xi}) + \vec{n}(\vec{\xi})\times\int_\Gamma \vec{v}(\vec{n}\cdot\vec{\nabla})u^* d\Gamma \\ = \vec{n}(\vec{\xi})\times\int_\Gamma \vec{v}\times[(\vec{n}\times\vec{\nabla})u^*]d\Gamma + \vec{n}(\vec{\xi}) \\ \times\int_\Omega(\vec{\omega}\times\vec{\nabla}u^*)d\Omega + \vec{n}(\vec{\xi})\times\mu^2\int_\Omega u^*\vec{v}_{F-1}d\Omega. \end{aligned} \tag{8}$$

Eq. (8) is the boundary-domain integral formulation of the boundary-vorticity modified Helmholtz kinematics equation.

Both, the domain-velocity and the boundary-vorticity modified Helmholtz kinematics equations are reformulated into discrete forms, in order to solve the domain velocity and boundary vorticity vector fields of the fluid flow.

3. Solution algorithms

During the simulation of coupled fluid flow and heat transfer problem, we solve the boundary-vorticity modified Helmholtz kinematics equation (7), the domain-velocity modified kinematics equation (8), the vorticity equation (4) and the energy equation (5). The individual algorithms are explained in detail below.

3.1. Discretization of the modified Helmholtz kinematics equation

In order to write the discrete form of the boundary-vorticity and domain-velocity modified Helmholtz kinematics equation, we divide the domain into domain elements $\Omega = \sum_i^D \Omega_i$ and the boundary into boundary elements $\Gamma = \sum_i^B \Gamma_i$. The domain elements are hexahedra with 27 nodes for quadratic interpolation of function. The boundary elements are quadrilaterals with 9 function nodes for quadratic interpolation of function. Let the boundary and domain shape functions be denoted by φ_m and ϕ_m. When the collocation point is placed in node k, the following integrals must be calculated

$$\begin{aligned} h_{kl} &= \int_{\Gamma_b}\varphi_m(\vec{n}\cdot\vec{\nabla})u^* d\Gamma_b, \qquad \vec{h}^t_{kl} = \int_{\Gamma_b}\varphi_m(\vec{n}\times\vec{\nabla})u^* d\Gamma_b, \\ \vec{d}_{kl} &= \int_{\Omega_d}\phi_l\vec{\nabla}u^* d\Omega_D, \qquad b_{kl} = \mu^2\int_{\Omega_d}\phi_m u^* d\Omega_d \end{aligned} \tag{9}$$

The matrix entries are stored in row k and column l, which is the node number of mth node in bth element. We denote the matrices compiled with entries from Eq. (9) as $[H]$, $[\vec{H}^t]$, $[\vec{D}]_\Gamma$, $[\vec{D}]_{\Omega/\Gamma}$ and $[B]$. Let the computation grid include n boundary nodes and m interior domain nodes. The matrices have the following number of rows and columns: Matrices $[H]$, $[\vec{H}^t]$, $[\vec{D}]_\Gamma$ have $n\times n$ rows and columns, $[\vec{D}]_{\Omega/\Gamma}$ have $n\times m$ and $[B]$ has $n\times(n+m)$.

3.2. Solution of the boundary-vorticity modified Helmholtz kinematics equation

The first stage of the algorithm is to solve for the vorticity at the boundary. In the following, we propose a fast BDIM for the boundary-vorticity modified Helmholtz equation. In order to obtain the boundary vorticity vector field from Eq. (8), we have to separate the unknown boundary vorticity values. The vorticity vector has to be written as a sum of boundary and domain vorticity as $\{\omega_i\} = \{\omega_i\}_\Gamma + \{\omega_i\}_{\Omega/\Gamma}$. From this, we can split the integral formulation $\vec{n}(\vec{\xi})\times\int_\Omega(\vec{\omega}\times\vec{\nabla}u^*)d\Omega$ in Eq. (8) into boundary and domain integral parts. Thus, we are able to form systems of linear equations for the boundary vorticity components $\{\omega_i\}_\Gamma$. A non-singular system is obtained when the normal derivative operator is formed as $([n_x][D_x]_\Gamma + [n_y][D_y]_\Gamma + [n_z][D_z]_\Gamma)$. Considering this and the discretization procedure, the following three systems of linear equations are written for the unknown boundary vorticity components:

$$\begin{aligned} &([n_x][D_x] + [n_y][D_y] + [n_z][D_z])\{\omega_x\}_\Gamma \\ &= ([n_y][H^t_y] + [n_z][H^t_z])\{v_x\}_\Gamma + ([n_y][H] - [n_z][H^t_x])\{v_z\}_\Gamma \\ &\quad -([n_x][H] + [n_y][H^t_x])\{v_y\}_\Gamma + [n_z][D_x]\{\omega_z\}_\Gamma + [n_y][D_x]\{\omega_y\}_\Gamma \\ &\quad +[n_x][D_x]\{\omega_x\}_\Gamma - ([n_y][D_y]_{\Omega/\Gamma} + [n_z][D_z]_{\Omega/\Gamma})\{\omega_x\}_{\Omega/\Gamma} \\ &\quad +[n_y][D_x]_{\Omega/\Gamma}\{\omega_y\}_{\Omega/\Gamma} + [n_z][D_x]_{\Omega/\Gamma}\{\omega_z\}_{\Omega/\Gamma} - [n_y][B]\{v^{F-1}_z\}_\Omega \\ &\quad +[n_z][B]\left\{v^{F-1}_y\right\}_\Omega, \end{aligned} \tag{10}$$

$$\begin{aligned} &([n_x][D_x] + [n_y][D_y] + [n_z][D_z])\{\omega_y\}_\Gamma \\ &= ([n_z][H^t_z] + [n_x][H^t_x])\{v_y\}_\Gamma + ([n_x][H] - [n_x][H^t_y])\{v_x\}_\Gamma \\ &\quad -([n_x][H] + [n_z][H^t_y])\{v_z\}_\Gamma + [n_z][D_y]\{\omega_z\}_\Gamma + [n_x][D_y]\{\omega_x\}_\Gamma \\ &\quad +[n_y][D_y]\{\omega_y\}_\Gamma - ([n_z][D_z]_{\Omega/\Gamma} + [n_x][D_x]_{\Omega/\Gamma})\{\omega_y\}_{\Omega/\Gamma} \\ &\quad +[n_z][D_y]_{\Omega/\Gamma}\{\omega_z\}_{\Omega/\Gamma} + [n_x][D_y]_{\Omega/\Gamma}\{\omega_x\}_{\Omega/\Gamma} - [n_z][B]\{v^{F-1}_x\}_\Omega \\ &\quad +[n_x][B]\{v^{F-1}_z\}_\Omega, \end{aligned} \tag{11}$$

$$\begin{aligned} &([n_x][D_x] + [n_y][D_y] + [n_z][D_z])\{\omega_z\}_\Gamma \\ &= ([n_x][H^t_x] + [n_y][H^t_y])\{v_z\}_\Gamma + ([n_x][H] - [n_y][H^t_z])\{v_y\}_\Gamma \\ &\quad -([n_y][H] + [n_x][H^t_z])\{v_x\}_\Gamma + [n_y][D_z]\{\omega_z\}_\Gamma + [n_x][D_z]\{\omega_x\}_\Gamma \\ &\quad +[n_z][D_z]\{\omega_z\}_\Gamma - ([n_x][D_x]_{\Omega/\Gamma} + [n_y][D_y]_{\Omega/\Gamma})\{\omega_z\}_{\Omega/\Gamma} \\ &\quad +[n_x][D_z]_{\Omega/\Gamma}\{\omega_x\}_{\Omega/\Gamma} + [n_y][D_z]_{\Omega/\Gamma}\{\omega_y\}_{\Omega/\Gamma} - [n_x][B]\left\{v^{F-1}_y\right\}_\Omega \\ &\quad +[n_y][B]\{v^{F-1}_x\}_\Omega, \end{aligned} \tag{12}$$

where the symbol $\{v_i\}_\Gamma$ is the boundary velocity vector field. Matrices $[n_x]$, $[n_y]$, $[n_z]$ are diagonal matrices, that include the components of the normal vector $\vec{n} = \{n_x, n_y, n_z\}$.

Since BDIM matrices are fully populated, we implemented the ACA algorithm with the $\mathcal{H}$-structure on the domain matrices $[D_x]_{\Omega/\Gamma}$, $[D_y]_{\Omega/\Gamma}$, $[D_z]_{\Omega/\Gamma}$ and $[B]$ in order to reduce the storage requirements and CPU time needed for matrix-vector multiplications. The systems are solved with the LU decomposition with a diagonal preconditioning.

3.2.1. Fast ACA matrix-vector multiplication

Let us consider one of the integral matrices $[D_i]_{\Omega/\Gamma}$ and denote it as D. Its size is $n\times m$ and its entries are real numbers, $D\in\mathbb{R}^{n\times m}$. In order to accelerate the algorithm, we implemented the Adaptive Cross Approximation and a $\mathcal{H}$-structure. Firstly, the $\mathcal{H}$-structure is built, and each part of the structure is tested for admissibility. Admissible parts are then approximated using the Adaptive Cross Approximation algorithm. This procedure approximates the matrix $D|_{n\times m}$ by splitting it into matrix parts $\hat{D}|_{\hat{n}\times\hat{m}}$ and approximating admissible parts via the low rank matrix approximation $\tilde{D}|_{r(\hat{n}+\hat{m})}$.

3.2.2. Construction of the $\mathcal{H}$-matrix structure

The computational domain is meshed with hexahedral elements. In order to construct the $\mathcal{H}$-structure we have to build two cluster trees. We will denote the cluster tree that is built from boundary elements as J and the domain cluster tree that is built from the domain elements as I. The root node of cluster trees is the whole domain and/or boundary. At the leaf level each mesh element represents a leaf node. The trees are formed using a bottom-up approach [31] combining neighbouring

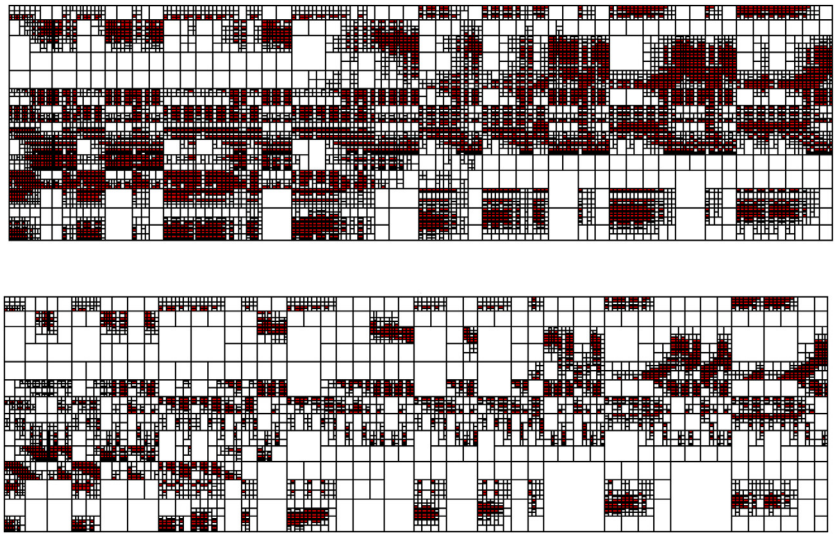

Fig. 3. Representation of the boundary-domain $\mathcal{H}$-matrix structure. White blocks are admissible parts, red blocks are inadmissible parts. Mesh with 25^3 nodes was used with $\eta = 2$ (top panel) and $\eta = 5$ (bottom panel).

Table 1
The share of inadmissible matrix blocks $1 - \frac{N_{adm}}{N}$. A cubic geometry was considered, meshed with elements concentrated at the walls. See Fig. 2 for an illustration of the mesh design.

η	17^3	25^3	33^3	41^3
1	0.95	0.45	0.34	0.21
2	0.53	0.25	0.17	0.10
3	0.38	0.17	0.13	0.069
4	0.31	0.12	0.097	0.054
5	0.30	0.11	0.085	0.043

element clusters together to form larger clusters in the next level. An example of a boundary and a domain cluster is shown in Fig. 2.

We combine the two cluster trees to build a boundary-domain block cluster tree $I \times J$. Each cluster is then tested for admissibility using the following admissibility condition:

$$min\left\{dim(I_i), dim(J_j)\right\} \leq \eta\ dist(I_i, J_i), \tag{13}$$

where $dim(I_i)$ and $dim(J_j)$ are the cluster sizes defined in Fig. 2. The distance $dist(I_i, J_j)$ is the minimal distance between nodes in the boundary and domain clusters. The admissibility parameter η is defined by the user. The admissibility condition (13) determines the admissible and inadmissible clusters.

When all clusters in the boundary-domain cluster tree are tested for admissibility, the $\mathcal{H}$-matrix is formed. In Fig. 3, we present an example of the $\mathcal{H}$-matrix structure. The inadmissible matrix parts are red blocks, and the admissible matrix parts are white blocks. Comparison of the structure at $\eta = 2$ and $\eta = 5$ reveals that the number of inadmissible matrix parts decreases at higher value of η.

In Table 1, we present the effect of the admissibility parameter η and mesh density on the number of matrix entries in admissible matrix parts. In the table, $N = n \times m$ is the number of entries in the matrix D, and N_{adm} is the number of entries which are in admissible parts. The ratio $1 - \frac{N_{adm}}{N}$ represents the share of inadmissible matrix blocks and, as such, the limit compression ratio for the approximation.

The geometry, the mesh density and the parameter η have a high influence on the construction of the $\mathcal{H}$-matrix and the number of admissible matrix parts. After the $\mathcal{H}$-matrix is constructed a low rank-approximation technique can be employed.

3.2.3. Adaptive Cross Approximation (ACA)

The Adaptive Cross Approximation is a method that approximates the matrix parts $\hat{D}|_{\hat{n}\times\hat{m}}$ that meet the admissibility condition, using a low rank matrix approximation (RK-matrix) [42]. Let the rank of the matrix $\hat{D}$: be k, and let us denote the approximation rank as r. The approximation rank r is the number of rows in matrix A and the number of columns in matrix B^T. The Adaptive Cross Approximation algorithm is written bellow:

- State $\hat{R}^0 = \hat{D}|_{\hat{n}\times\hat{m}}$
- For $r = 0, 1, 2, 3, \ldots, k$
  1. $(i^*, j^*)^r = ArgMax|(\hat{R}^r)|$
  2. $\gamma^{r+1} = (\hat{R}^r_{i^*,j^*})^{-1}$
  3. $a_{r+1} = \gamma^{r+1}\hat{R}^r_{i,j^*},\ b_{r+1} = (\hat{R}^r_{i^*,j})^T$
  4. $\hat{R}^{r+1} = \hat{R}^r - a_{r+1}b_{r+1}$
- If $(\|a^T_{r+1}\|_F \|b^T_{r+1}\|_F \leq \epsilon \|S^{r+1}\|_F \vee r = k)$ Stop
- EndFor

At the start of the algorithm, we define the residual matrix. Next, we start a loop that performs the approximation. Firstly, the largest element in the matrix $\hat{R}^u$ is found. Next, the inverse of the largest element $\hat{R}^u_{i^*,j^*}$ is calculated. In the third step the vectors a_{u+1} and b_{u+1} are defined. These two vectors build a cross around the element $\hat{R}^u_{i^*,j^*}$. Finally, we calculate a new residual matrix $\hat{R}^{u+1}$. The algorithm ends when the stopping criteria is satisfied, or if the maximal rank is reached. We use the Frobenius norm $\|\cdot\|_F$ and a user prescribed stopping parameter ϵ. The stopping condition was implemented as proposed by Bebendorf and Grzibovski [43]:

$$\left\|S^{r+1}\right\|_F = \|S^r\|_F + \sum_{j=1}^{r} a^T_{r+1}a_r b_r b^T_{r+1} + \left\|a^T_{r+1}\right\|_F \left\|b^T_{r+1}\right\|_F. \tag{14}$$

Based on the approximation rank r we can calculate the compression ratio

$$\varphi = \frac{\sum_i \hat{n}_i \cdot \hat{m}_i + \sum_j r_j(\hat{n}_j + \hat{m}_j)}{n \cdot m}, \tag{15}$$

which measures the memory usage of the approximated matrix versus the original matrix. It is the sum of the matrix elements in admissible matrix parts $\tilde{D}|_{r(\hat{n}+\hat{m})}$ plus the elements in inadmissible matrix parts $\hat{D}|_{\hat{n}\times\hat{m}}$, and the number of elements in the original matrix $D|_{n\times m}$.

The ACA algorithm reformulates the full matrix formulation into a low rank *RK*-matrix approximation. In order to store the *RK*-matrix into memory and perform matrix-vector multiplications, we need $O(r(\hat{n} + \hat{m}))$ computational resources, [44]. Furthermore, the complexity of the

BDIM is reduced from O(N^2) to O($N\log N$), with the ACA algorithm and the $\mathcal{H}$-structure.

3.3. Solution of the domain-velocity modified Helmholtz kinematics equation

The second stage of the flow simulation algorithm is the solution of the domain-velocity vector field by solving equation (7). We employed the subdomain boundary-domain method, that was proposed by Ravnik et al. [9].

The subdomain Boundary-Domain Integral Method is based on domain decomposition. It splits the domain into smaller domain parts and applies BDIM on each part. Compatibility boundary conditions are prescribed between subdomains. In this paper, we used each mesh element as a subdomain. The resulting integral matrices are sparse and have a finite-element type structure. The following systems of linear equations are produced:

$$\begin{aligned}
[H]\{v_x\} &= [H_z^t]\{v_y\} - [H_y^t]\{v_z\} + [D_z]\{\omega_y\} - [D_y]\{\omega_z\} + [B]\{v_x^{F-1}\},\\
[H]\{v_y\} &= [H_x^t]\{v_z\} - [H_z^t]\{v_x\} + [D_x]\{\omega_z\} - [D_z]\{\omega_x\} + [B]\left\{v_y^{F-1}\right\},\\
[H]\{v_z\} &= [H_y^t]\{v_x\} - [H_x^t]\{v_y\} + [D_y]\{\omega_x\} - [D_x]\{\omega_y\} + [B]\{v_z^{F-1}\}.
\end{aligned} \tag{16}$$

The source point is placed into each node in each subdomain. This means that the system of Eq. (16) is over-determined. A least squares solver is used to find a solution [45].

3.4. Solution of vorticity and energy transport equations

The final two steps in the flow solution algorithm is the solution of Eqs. (4) and (5) for the unknown vorticity domain field and the temperature field. Subdomain BDIM is used, as proposed by Ravnik et al. [9].

3.5. Flow solution algorithm

Due to the nonlinear nature of the governing equations of flow and heat transfer, we propose an iterative algorithm with under-relaxation. We propose to implement a nonlinear loop and, within each nonlinear step, a false transient loop to resolve the false time. The solution procedure is presented in the following algorithm:

- While nonlinear loop, $||\vec{v}-\vec{v}'|| > \kappa \quad \vee \quad ||\vec{\omega}-\vec{\omega}'|| > \kappa \quad \vee \quad ||T-T'|| > \kappa$
  1. Solve Eq. (12) for the boundary vorticity vector using ACA accelerated BDIM
     - While false transient loop, $||\vec{v}-\vec{v}_{F-1}|| > \kappa$
       (a) Solve Eq. (16) for the domain velocity vector field using subdomain BDIM
     - EndWhile
  2. Solve Eq. (4) for the domain vorticity vector field using subdomain BDIM
  3. Solve Eq. (5) for the temperature field using subdomain BDIM
  4. Apply under-relaxation
- EndWhile

Firstly, the algorithm solves the tangential formulation of the modified Helmholtz kinematics equation to find the boundary vorticity values. These serve as boundary conditions for the vorticity transport equation. The solution of Eq. (12) could be placed inside or outside of the false transient loop, due to the fact it is always inside of the nonlinear loop. When it is placed outside, even though at the start of the nonlinear loop the values of boundary vorticity are inaccurate, the overall convergence properties are better. Next, the modified Helmholtz kinematics equation is solved in a false transient loop to calculate the domain velocity values. When the difference between the velocity vector field of the previous time step $\vec{v}_F$ and the velocity vector field of the current time step $\vec{v}$ converges the solution of the domain velocity field is equal to the domain velocity vector field of the kinematics equation, for the same boundary vorticity. Secondly, the vorticity transport equation is solved for domain vorticity. In the third step, the velocity and vorticity vector fields are used to solve the energy equation for unknown temperature values. Lastly under-relaxation of the solved vector and scalar fields is applied to stabilize the numerical calculation. The value of the under-relaxation parameter ranged between 0.5 to 0.01 in the simulations performed in this study. The temperature and the velocity boundary conditions are known, while boundary vorticity values are obtained as a part of the algorithm. The false transient loop and the nonlinear loop run until convergence is achieved. The difference between subsequent iterations, expressed as RMS norm, is compared with convergence criterion $\kappa = 10^{-6}$.

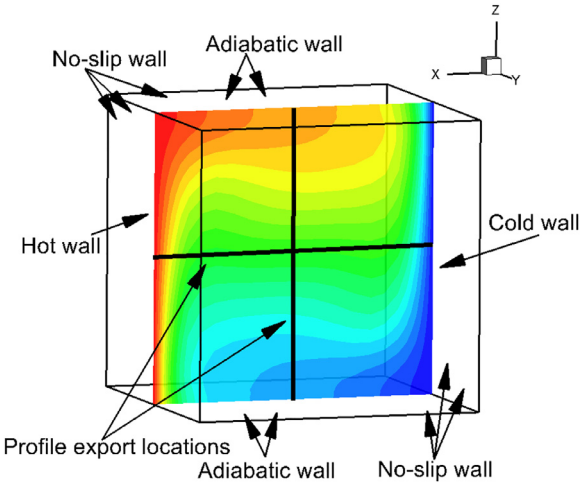

Fig. 4. The boundary conditions for the natural convection in a closed cavity test case. A $Ra = 10^5$ temperature field is shown on the $x - z$ cross-section. The profile export locations are shown on the $y = 0.5$ plane.

4. Numerical example

Natural convection occurs due to differences in fluid temperature, which results in density differences and buoyancy forces. A standard natural convection benchmark test case is the flow in a closed differentially heated cavity. This case has already been studied by different authors, so results for comparison are available [37–39].

The simulation is performed in a three-dimensional cube. In Fig. 4, we present an illustration of the geometry, boundary conditions and profile export locations. The no-slip velocity boundary conditions are employed on all walls. Two opposite walls have constant hot and cold temperatures prescribed, while all other walls are adiabatic. The vorticity boundary conditions are unknown, they are calculated by solving the Adaptive Cross Approximation accelerated modified Helmholtz kinematics equation as a part of the solution algorithm.

All numerical simulations were performed on an Intel Xeon 64-bit processor using 64 GB of memory. We simulated the fluid flow and heat transfer at different Rayleigh numbers (Ra) in order to observe the influence of the ACA approximation on the solution accuracy. The Rayleigh number was set to 10^3, 10^4 and 10^5. A Prandtl number value of $Pr = 0.71$ was used, which represents air at 20 °C.

In order to facilitate the comparison between simulations with and without matrix approximation, we based the choice of the fine mesh density on the availability of the computer storage space to store full uncompressed matrices. The mesh structure was built from hexagonal non-uniform mesh elements. In order to have a well-resolved boundary layer, the size of elements increases with the distance from the domain

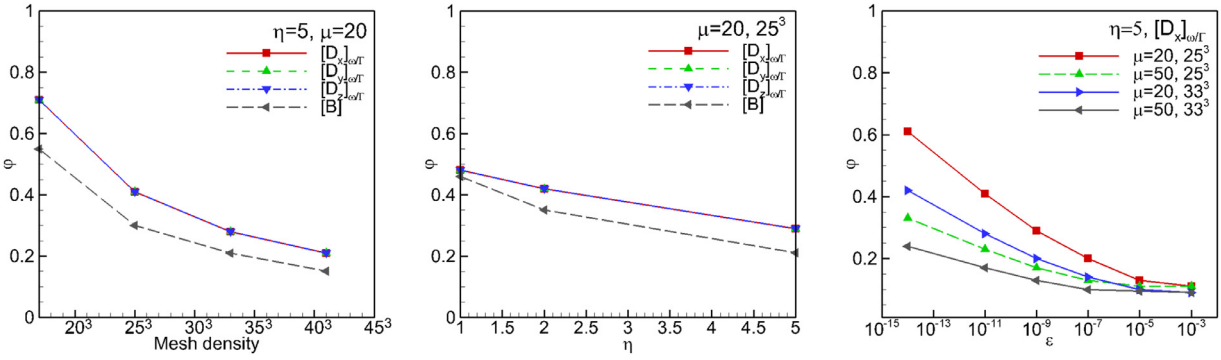

Fig. 5. Compression ratio φ dependence on the mesh density (left), on the parameter η (centre) and on the stopping condition ϵ (right).

walls using a geometrical progression. Four different mesh resolutions were employed to perform the simulations. The meshes had 17^3, 25^3, 33^3, and 41^3 nodes. The ratio between the longest and shortest element size was 1:3 in mesh 17^3 and 1:6.5 in others.

In order to determine the maximal approximation rank r, and its relationship to the flow solution accuracy, we used Eq. (15) with different stopping parameters ϵ. The stopping parameter ϵ was varied from 10^{-2} to 10^{-20}.

The Adaptive Cross Approximation, in combination with the $\mathcal{H}$-matrix, was implemented on matrices $[D_x]_{\Omega/\Gamma}$, $[D_y]_{\Omega/\Gamma}$, $[D_z]_{\Omega/\Gamma}$ and $[B]$. In order to measure the influence of the approximation on the results and find the dependence on the flow structure and mesh density, we used an RMS norm to measure the difference in vorticity, velocity and temperature fields:

$$RMS_\omega = \left(\frac{\sum_{j=x,y,z} \sum_i (\omega_{ji} - \omega_{Aji})^2}{\sum_{j=x,y,z} \sum_i (\omega_{Aji})^2} \right)^{\frac{1}{2}},$$

$$RMS_V = \left(\frac{\sum_{j=x,y,z} \sum_i (v_{ji} - v_{Aji})^2}{\sum_{j=x,y,z} \sum_i (v_{Aji})^2} \right)^{\frac{1}{2}}, \quad RMS_T = \left(\frac{\sum_i (T_i - T_{Ai})^2}{\sum_i (T_{Ai})^2} \right)^{\frac{1}{2}}, \tag{17}$$

where ω_{ji} is the jth component of vorticity in the ith node calculated without an approximation, and the ω_{Aji} is its approximated counterpart. The same notation is used for the velocity and temperature fields.

4.1. Results and discussion

In this subsection, we examine the properties of the newly presented algorithm by performing simulations of the natural convection phenomenon in a closed cavity.

The choice of the ACA stopping parameter ϵ, the fundamental solution parameter μ, the computational mesh density and the admissibility criterion η determine the compression ratio, the amount of memory used, the CPU time needed to perform matrix operations and the accuracy of the final solution. The ultimate goal is to provide guidelines for the choice of parameter values at which the accuracy of the final solution is unaffected by the introduction of the fast method and at which the compression ratio the highest possible.

In Fig. 5, we present the compression ratio φ obtained for different computational grids, admissibility parameters η, and ACA stopping parameters ϵ. We can observe that the compression of the matrices depends on the mesh resolution, where denser meshes enable higher compression ratios. The parameter μ also has an effect on the compression of the matrices. Choosing a higher value of the parameter μ leads to an increase of the compression ratio. We have observed that the compression of the matrices $[D_x]_{\Omega/\Gamma}$, $[D_y]_{\Omega/\Gamma}$, $[D_z]_{\Omega/\Gamma}$ is very similar, and that the matrix $[B]$ was compressed even more. This happened due to the fact that the integral entries of the matrices $[D_x]_{\Omega/\Gamma}$, $[D_y]_{\Omega/\Gamma}$, $[D_z]_{\Omega/\Gamma}$ include the gradient of the fundamental solution for its kernel, while the entries of the matrix $[B]$ are obtained by using the fundamental solution as the kernel. In consequence, the ACA algorithm approximated the matrices differently. The ACA stopping parameter ϵ is used to set the compression ratio, where large values lead to high compression ratio. The compression ratio φ also depends on the $\mathcal{H}$-matrix structure. Higher values of η lead to stronger compression.

The compression ratio is linked with the memory consumption. In Fig. 6 we present the amount of memory that has to be assigned in order to store all of the matrices arising from the discretization of the kinematics equation. In the left panel we present how the choice of the μ parameter affects the memory consumption. We observe, that at a given ACA stopping condition ϵ, using high values of μ leads to lower memory usage. In the right panel, we compare memory usage at different mesh densities. We observe that for all choices of μ and ϵ the growth of memory usage is approximately linear, while the growth of the original BDIM method is quadratic.

In Fig. 1, we illustrate the shape of the fundamental solution at different μ values, exposing the more local character of the fundamental solution at high μ values. This observation is confirmed by looking (Fig. 5) at the compression ratio φ of the matrices, which is higher at $\mu = 50$ than at $\mu = 20$. In consequence the memory consumption is lower at high μ values (Fig. 6). Thus, the shape of the fundamental solution has an influence on the compression ratio of the matrices.

Next, we study the influence of the parameter η by simulating the problem at $Ra = 10^5$ and keeping $\mu = 20$ (Fig. 7). We present norms RMS_ω, RMS_T and RMS_v versus compression ratio obtained with different choices of the ϵ parameter. We observe that, for low compression ratio, the result accuracy is independent of the admissibility parameter η. The error introduced by the ACA approximation is smaller than the discretization and linear solver errors. We observe a sharp decrease of solution accuracy at a critical compression ratio, when the error introduced by the matrix approximation begins to dominate. This critical compression ratio does depend on the admissibility parameter; the higher the admissibility parameter, the lower the critical compression ratio. Since at the critical compression ratio, we have the best compression ratio and still no influence of the matrix approximation on the solution result, we denote this compression ratio as the optimal compression ratio. When simulations are run at the optimal compression ratio, we will achieve maximal possible compression at negligible influence on the results.

In Fig. 8, we present the dependence of the RMS_ω on the Rayleigh number and the type of fundamental solution used. We observe that, when elliptic fundamental solution is used, the optimal compression ratio does not seem to depend on the Rayleigh number value (non-linear nature of the problem). On the other hand, when modified Helmholtz fundamental solution is used, dependence on Rayleigh number value is evident, especially at high μ values.

In Fig. 9, we present the total number of iterations in the nonlinear and in the false transient loops at different compression ratios. We observe that the total number of iterations depends on the Rayleigh number, the compression ratio φ and the shape of the fundamental solutions.

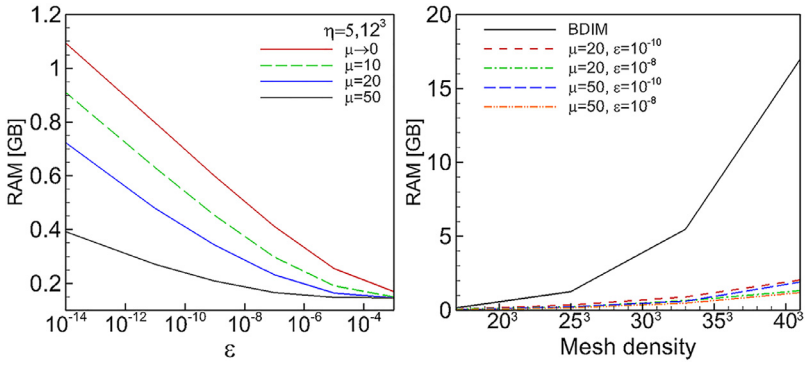

Fig. 6. Memory usage of all of the kinematics matrices. Influence of the μ parameter is examined in the left panel, dependence on the mesh density in the right panel. BDIM denotes the original algorithm, which does not use fast compression techniques.

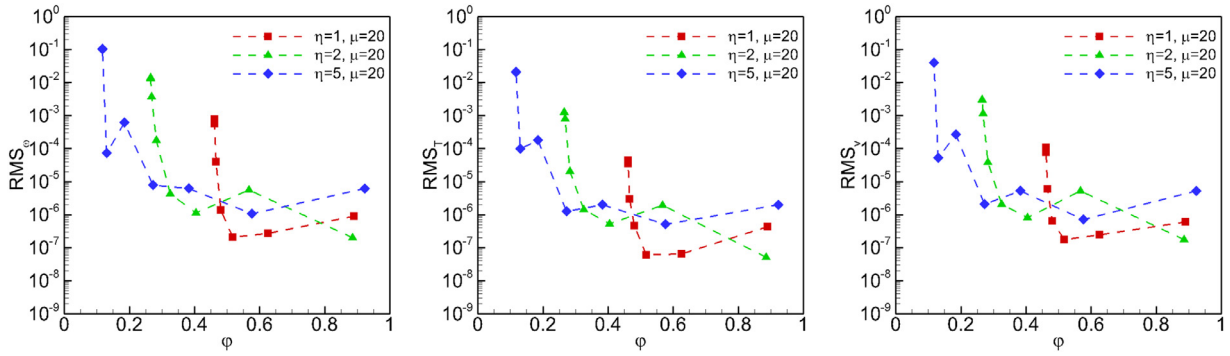

Fig. 7. Norms RMS_ω (left), RMS_T (center) and RMS_v (right) versus compression ratio obtained with different choices of the ϵ parameter. Three different values of admissibility parameter are considered. The mesh density was 25^3, $Ra = 10^5$.

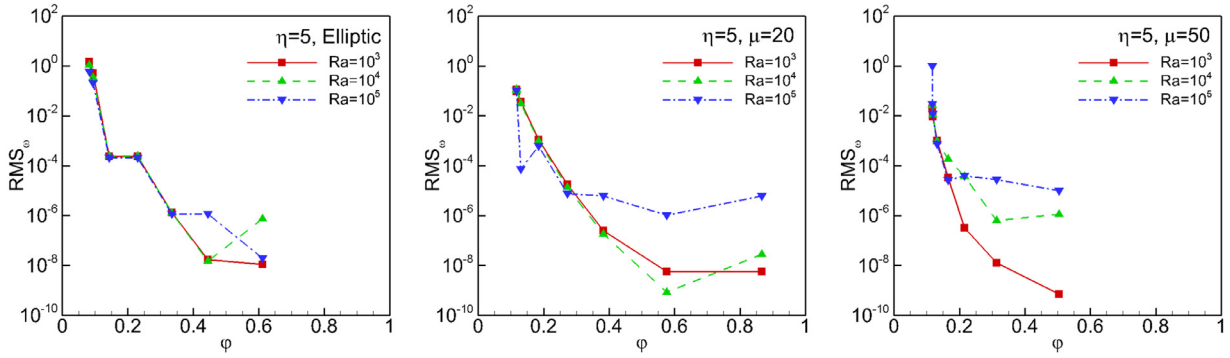

Fig. 8. Norms RMS_ω versus compression ratio obtained using the elliptic fundamental solution (left), modified Helmholtz fundamental solution $\mu = 20$ (center) and $\mu = 50$ (right) at different Rayleigh numbers. Mesh density was 25^3.

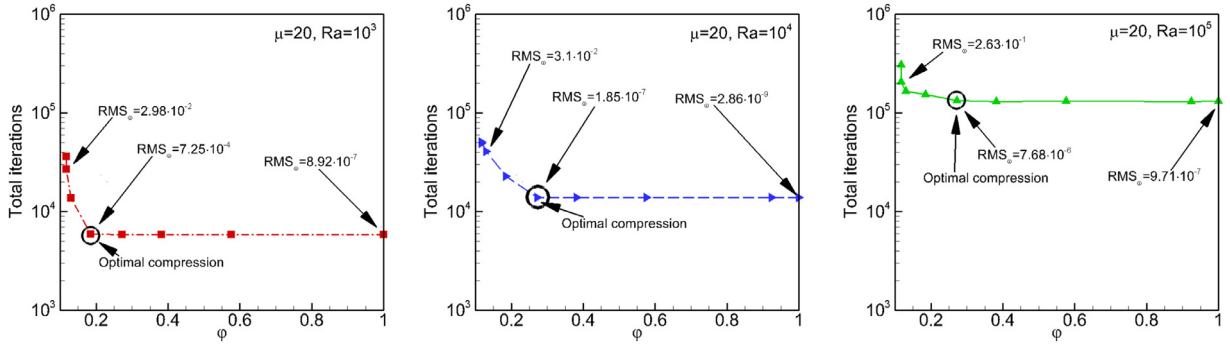

Fig. 9. Number of iterations needed to obtain a converged solution of the flow field for different compression ratios, $\mu = 20$. Rayleigh number values $Ra = 10^3$ (left panel), $Ra = 10^4$ (centre panel) and $Ra = 10^5$ (right panel).

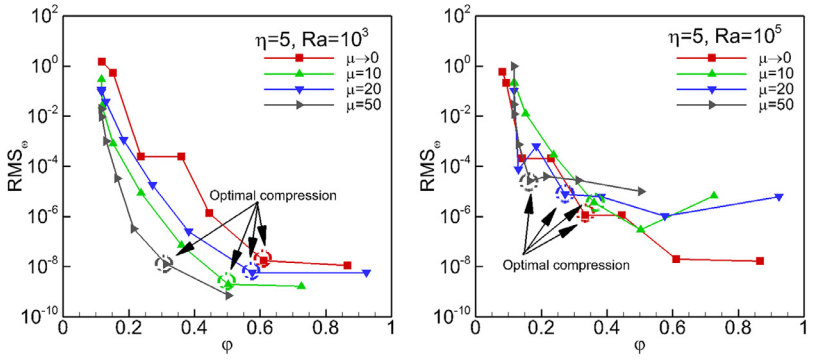

Fig. 10. We examine how the type of fundamental solution and the μ value effect the accuracy of the solution expressed using RMS_ω norm. Rayleigh number 10^3 is shown in the left panel and $Ra = 10^5$ in the right panel. The mesh density was 25^3.

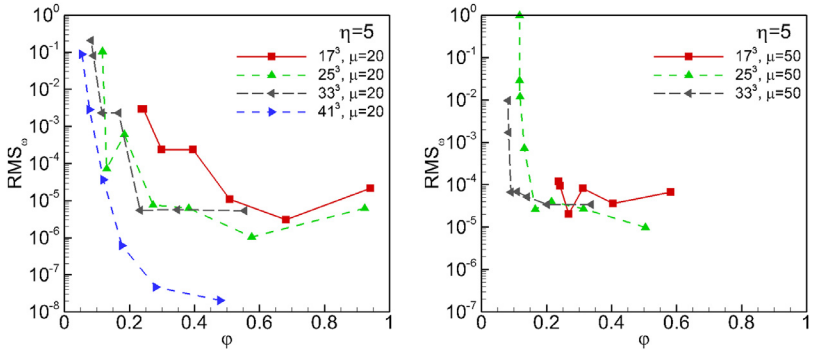

Fig. 11. Dependence of the mesh density and parameter μ on the compression ratio φ of the matrices. The Rayleigh number for all cases was 10^5 and the mesh density was 25^3.

Even though the increase of the Rayleigh number requires an increased number of iterations to resolve the nonlinear nature of the flow, the error introduced into the flow by the ACA matrix approximation does not propagate to the increase the number of iterations. When the ACA stopping parameter ϵ is set below the optimal compression ratio, we observe a sharp increase in the number of iterations. This results in poor solution accuracy, and increases the CPU time needed to reach the solution.

The shape of the fundamental solution has an influence on the approximation of the matrices. In Fig. 10, we illustrate the influence of μ on the compression ratio and the RMS_ω. The integral matrices, which are using the modified Helmholtz fundamental solution, can be approximated better by the ACA than the ones using the elliptic fundamental solution. This is due to the more local character of the modified Helmholtz fundamental solution, which is governed by the parameter μ. To show this we marked points that represent the optimal compression of the matrices in the figure. The optimal compression point moves to a lower compression ratio φ at a higher μ value.

At low Rayleigh number values the solution accuracy at the optimal compression ratio is independent of the type of fundamental solution. At high Rayleigh number values, when the non-linearity of the problem is more pronounced, the accuracy at the optimal compression ratio drops with increasing μ value. This can be explained by the fact that the error introduced by ACA approximation is multiplied by the Rayleigh number value and, thus, has a stronger effect at high Rayleigh number values.

In Fig. 11, we illustrate the dependence of the mesh density on the solution of the vorticity vector field. A better resolved mesh contributes to the approximation of the matrix. The RMS_ω is lower at a better resolved mesh. Thus, the optimal compression ratio is improved.

In Fig. 12, we present the temperature and velocity profiles at the $y = 0.5$ plane. In order to verify that without using compression, the algorithm yields the same result, regardless of the μ value employed, we present a comparison of profiles in the left panel of Fig. 12. We observe good agreement between results obtained using the elliptic fundamental solution and results obtained using the modified Helmholtz fundamental solution at different μ values. In the central and in right panels we display temperature and velocity profiles obtained at different compression ratios. The optimal compression was $\varphi = 0.42$, where virtually no difference can be observed to the uncompressed solution. When using sub-optimal compression ($\varphi < 0.42$) small differences can be observed. The differences are larger in the velocity field and smaller in the temperature field.

Table 2
Influence of μ and ϵ on the average Nusselt number $\overline{Nu}$ at different Rayleigh number values and comparison with the benchmark results of Tric et al. [39] and Lo et al. [46]. BDIM denotes the solution obtained without compression ($\epsilon = 0$). Simulations were performed with $\eta = 5$.

Mesh density	μ	BDIM	$\epsilon = 10^{-10}$	$\epsilon = 10^{-6}$	$\epsilon = 10^{-2}$
$Ra = 10^3$					
17^3	20	1.0710	1.0708	1.0708	1.0709
25^3	20	1.0710	1.0710	1.0705	1.0694
41^3	20	1.0710	1.0710	1.0695	1.0666
17^3	50	1.0709	1.0709	1.0710	1.0710
25^3	50	1.0709	1.0709	1.0708	1.0707
Tric et al. [39]		1.0700			
Lo et al. [46]		1.0710			
$Ra = 10^4$					
17^3	20	2.0574	2.0574	2.0569	2.0575
25^3	20	2.0571	2.0571	2.0552	2.0501
41^3	20	2.0572	2.0573	2.0500	2.0457
17^3	50	2.0592	2.0592	2.0591	2.0591
25^3	50	2.0544	2.0544	2.0530	2.0521
Tric et al. [39]		2.0542			
Lo et al. [46]		2.0537			
$Ra = 10^5$					
17^3	20	4.3499	4.3499	4.3485	4.5077
25^3	20	4.3600	4.3600	4.3561	4.3411
41^3	20	4.3706	4.3705	4.3584	4.3888
17^3	50	4.3671	4.3672	4.3671	4.3671
25^3	50	4.3362	4.3361	4.3295	4.3280
Tric et al. [39]		4.3375			
Lo et al. [46]		4.3329			

In Table 2, we present the heat flux at the hot wall expressed as the average Nusselt number value $\overline{Nu}$. The approximation error that is introduced by the ACA algorithm into the fluid flow does not have a

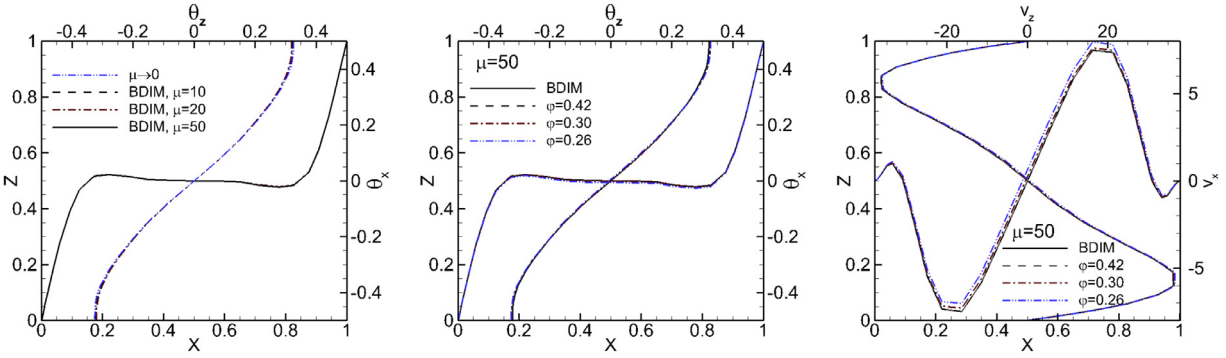

Fig. 12. In the left panel we show the temperature profiles across the $y = 0.5$ plane for different fundamental solution types without compression. Next, we compare the temperature profiles (center) and the velocity profiles (right) at different compression ratios using $\mu = 50$ and $\eta = 5$. The Rayleigh number was 10^5 and the mesh density was 25^3. BDIM denotes the solution obtained without the compression.

substantial effect on the average $\overline{Nu}$, even when sub-optimal compression is used. The average Nusselt number increases with a higher compression of the matrices. However, the ACA has little effect on the average Nusselt number at a better-resolved mesh. Comparison with benchmark data provided by Tric et al. [39] and Lo et al. [46] is good.

5. Conclusions

In this study, we developed an Adaptive Cross Approximation algorithm with the modified Helmholtz fundamental solution in order to accelerate the Boundary-Domain Integral Method. The developed algorithm was used in an incompressible flow solver, which solves the velocity–vorticity formulation of the Navier–Stokes equations in three-dimensions. We have shown that the Adaptive Cross Approximation algorithm can be combined with the modified Helmholtz fundamental solution to reduce computational demands of the flow solver.

We investigated the influence of the shape of the fundamental solution, the nonlinearity of the problem, and the admissibility criterion, to establish the optimal compression ratio. We have shown that, at the optimal compression ratio, the solution accuracy and the convergence properties are unaffected by the introduction of approximation of integral matrices.

Due to its more local shape, the modified Helmholtz fundamental solution enables better optimal compression ratios compared to the elliptic fundamental solution. Thus, the use of the modified Helmholtz fundamental solution is preferred in terms of computer storage. On the other hand, in order to employ the modified Helmholtz fundamental solution in the solution procedure for determination of boundary vorticity, we had to introduce a false transient loop into the algorithm. For this reason, in terms of CPU time, the advantage of the modified Helmholtz fundamental solution is less pronounced.

Four main conclusions can be derived from our observations on the influence of the ACA approximation algorithm and modified Helmholtz fundamental solution on the solution of the fluid flow. Firstly, the modified Helmholtz fundamental solution contributes to the Adaptive Cross Approximation algorithm in the way that a greater compression can be achieved. Secondly, the modified Helmholtz fundamental solution has a positive effect on the error that is introduced by the Adaptive Cross Approximation algorithm on the solution of the fluid flow. Thirdly, the approximation of the matrices depends on the $\mathcal{H}$-matrix formulation, and, finally, a better-resolved mesh can support higher compression ratio to achieve a target solution accuracy.

Acknowledgments

The authors acknowledge the financial support from the Slovenian Research Agency (research core funding No. P2-0196) and the Deutsche Forschungsgemeinschaft (project STE 544/58).

References

[1] Wang X, Ang W-t. A complex variable boundary element method for solving a steady-state advection diffusion reaction equation. Appl Math Comput 2018;321:731–44. doi:10.1016/j.amc.2017.11.016.
[2] Ghadimi P, Dashtimanesh A, Hosseinzadeh H. Solution of Poisson's equation by analytical boundary element integration. Appl Math Comput 2010;217(1):152–63. doi:10.1016/j.amc.2010.05.034.
[3] Li Q, Chen S, Luo X. Steady heat conduction analyses using an interpolating element-free Galerkin scaled boundary method. Appl Math Comput 2017;300:103–15. doi:10.1016/j.amc.2016.12.007.
[4] Liao Sj, Chwang TA. General boundary-element method for unsteady nonlinear heat transfer problems. Numer Heat Transf Part B Fundam 1999;35(2):225–42.
[5] Yu B, Yao W, Gao Q. A precise integration boundary-element method for solving transient heat conduction problems with variable thermal conductivity. Numer Heat Transf Part B Fundam 2014;65(5):37–41. doi:10.1080/10407790.2013.873311.
[6] Cui M, Xu B-b, Feng W-z, Zhang Y, Gao X-w. A radial integration boundary element method for solving transient heat conduction problems with heat sources and variable thermal conductivity. Numer Heat Transf Part B Fundam 2018a;73(1):1–18.
[7] Cui M, Peng H-f, Xu B-b, Gao X-w, Zhang Y. A new radial integration polygonal boundary element method for solving heat conduction problems. Int J Heat Mass Transf 2018b;123:251–60.
[8] Yang D, Yue X, Yang Q. Virtual boundary element method in conjunction with conjugate gradient algorithm for three-dimensional inverse heat conduction problem. Numer Heat Transf Part B Fundam 2017;71(6):Pages421–30.
[9] Ravnik J, Škerget L, Zunič Z. Velocity–vorticity formulation for 3D natural convection in an inclined enclosure by BEM. Int J Heat Mass Transf 2008;51(17–18):4517–4527.
[10] Wrobel LC. The Boundary Element Method. John Willey and Sons, Ltd.; 2002.
[11] Škerget L, Alujevic A, Brebia C, Kuhn G. Natural and forced convection simulation using the velocity–vorticity approach. Top Bound Elem Res 1989;5:49–86.
[12] Škerget L, Hriberšek M, Kuhn G. Computational fluid dynamics by boundary-domain integral method. Int J Numer Methods Eng 2000;46:1291–311.
[13] Kalman D. A singularly valuable decomposition: the SVD of a matrix. Col Math J 1996;27(1):2–23.
[14] Darve E. The fast multipole method: numerical implementation. J Comput Phys 2000;160:195–240. doi:10.1006/jcph.2000.6451.
[15] Daubechies I. Ten Lectures on Wavelets. Society for Industrial Applied Mathematics; 1992.
[16] Ravnik J, Skerget L. Fast single domain – subdomain BEM algorithm for 3D incompressible fluid flow and heat transfer. Int J Numer Methods Eng 2009;77:1627–45. doi:10.1002/nme.
[17] Bebendorf M. Approximation of boundary element matrices. Numer Math 2000;86(4):565–89.
[18] Rjasanow S. Adaptive cross approximation of dense matrices. In: Proceedings of the IABEM 2002 Symposium; 2002. p. 1–12.
[19] Börm S, Grasedyck L. Hybrid cross approximation of integral operators. Numer Math 2005;101(2):221–49.
[20] Smajic J, Andjelic Z, Bebendorf M. Fast BEM for eddy-current problems using H-matrices and adaptive cross approximation. IEEE Trans Magnet 2007;43(4):1269–72.
[21] Schröder A, Brüns H-D, Schuster C. Fast evaluation of electromagnetic fields using a parallelized adaptive cross approximation. IEEE Trans Antennas Propag 2014;62(5):2818–22.
[22] Grytsenko T, Galybin AN. Numerical analysis of multi-crack large-scale plane problems with adaptive cross approximation and hierarchical matrices. Eng Anal Bound Elem 2010;34(5):501–10.
[23] Maerten F. Adaptive cross-approximation applied to the solution of system of equations and post-processing for 3D elastostatic problems using the boundary element method. Eng Anal Bound Elem 2010;34(5):483–91.
[24] Kurz S, Rain O, Rjasanow S. The adaptive cross-approximation technique for the 3-D boundary-element method. IEEE Trans Magnet 2002;38(2 I):421–4. doi:10.1109/20.996112.

[25] Van T, Suzuki LRC, Latypov D, Holle JV, Voss T, Van T, et al. Fast algebraic methods in computational electromagnetics. In: Proceedings of the IEEE 2010 National Aerospace and Electronics Conference (NAECON); 2010. p. 230–6. Dayton
[26] Campos LS, de Albuquerque EL, Wrobel LC. An ACA accelerated isogeometric boundary element analysis of potential problems with non-uniform boundary conditions. Eng Anal Bound Elem 2017;80:108–15. doi:10.1016/j.enganabound.2017.04.004.
[27] Ravnik J, Črnec D, Hriberšek M, Zadravec M. Magnetic susceptibility determination based on microparticles sedimentation analysis. Int J Simul Model 2017;16(2):275–288.
[28] Hackbusch W. A sparse matrix arithmetic based on H-matrices. Computing 1999;108:89–108.
[29] Grasedyck L, Hackbusch W. Hierarchical Matrices. Leipzig: Max-Planck-Institut für Mathematik in den Naturwissenschaften; 2006.
[30] Börm S. Approximation of integral operators by H2-matrices with adaptive bases. Computing 2005;74(3):249–71.
[31] Tibaut J, Škerget L, Ravnik J. Acceleration of a BEM based solution of the velocity–vorticity formulation of the Navier–Stokes equations by the cross approximation method. Eng Anal Bound Elem 2017;82(C):17–26.
[32] Mallinson GD, Davis DEV. The Method of the False Transient for the Solution Coupled Elliptic Equations. J Comput Phys 1973;435–461(12):435–61.
[33] Guj G, Stella F. A vorticity–velocity method for the numerical of 30 incompressible flows. J Comput Phys 1993;106:286–98.
[34] Behnia M, Stella F, Guj G. A numerical study of the three-dimensional combined buoyancy and thermocapillary convection. Int J Multiph Flow 1995;21(3):529–542.
[35] Škerget L, Rek Z. Boundary-domain integral method using a velocity–vorticity formulation. Int J Numer Methods Eng 1995;15:359–70.
[36] Kocutar P, Škerget L, Ravnik J. Hybrid LES/URANS simulation of turbulent natural convection by BEM. Eng Anal Bound Elem 2015;61:16–26. doi:10.1016/j.enganabound.2015.06.005.
[37] Phillips TN. Natural convection in an enclosed cavity. J Comput Phys 1984;54:365–381.
[38] Quere PL, Alziary de Roquefort T. Computation of natural convection in two- dimensional cavities with Chebyshev polynomials. J Comput Phys 1985;57:210–28.
[39] Tric E, Labrosse G, Betrouni M. A first incursion into the 3D structure of natural convection of air in a differentially heated cubic cavity, from accurate numerical solutions. Int J Heat Mass Transf 2000;43(21):4043–56. doi:10.1016/S0017-9310(00)00037-5.
[40] Markatos NC, Pericleous KA. Laminar and turbulent natural convection in an enclosed cavity. Int J Heat Mass Transf 1984;27(5):755–72.
[41] Škerget L, Hriberšek M, Žunic Z. Natural convection flows in complex cavities by BEM. Int J Numer Methods Heat Fluid Flow 2003;13(6):720–35.
[42] Börm S, Grasedyck L, Hackbusch W. Introduction to hierarchical matrices with applications. Eng Anal Bound Elem 2003;27(5):405–22.
[43] Bebendorf M, Grzibovski R. Accelerating Galerkin BEM for Linear Elasticity using Adaptive Cross Approximation. Leipzig: Max-Planck-institut für Mathematik in den Naturwissenschaften; 2006.
[44] Rjasanow S, Steinbach O. The Fast Solution of Boundary Integral Equations. New York: Springer Science + Business Media; 2007.
[45] Paige CC, Saunders MA. LSQR: an algorithm for sparse linear equations and sparse least squares. ACM Trans Math Softw 1982;8(1):43–71.
[46] Lo DC, Young DL, Murugesan K, Tsai CC, Gou MH. Velocity vorticity formulation for 3D natural convection in an inclined cavity by DQ method. Int J Heat Mass Transf 2007;50:479–91. doi:10.1016/j.ijheatmasstransfer.2006.07.025.