Engineering Analysis with Boundary Elements 82 (2017) 17–26

Contents lists available at ScienceDirect

Engineering Analysis with Boundary Elements

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

Acceleration of a BEM based solution of the velocity–vorticity formulation of the Navier–Stokes equations by the cross approximation method

J. Tibaut[a,*], L. Škerget[b], J. Ravnik[a]

[a] Faculty of Mechanical Engineering, University of Maribor, Smetanova ulica 17, SI-2000 Maribor, Slovenia
[b] Wessex Institute, Ashurst Lodge, Ashurst, Southampton SO40 7AA, UK

ARTICLE INFO

2010 MSC:
65N38
74S15
76M15
76D05

Keywords:
Boundary-domain integral method
Velocity–vorticity formulation
Adaptive cross approximation
Fast algorithms

ABSTRACT

In this paper, we present a method to decrease the computational demand of the boundary-domain integral equation based 3D flow solver. We focus on the solution of the velocity–vorticity formulation of the Navier–Stokes equation, which governs incompressible viscous fluid flow. We introduce the cross approximation into the solution of the boundary vorticity values. This problem is governed by the Poisson type kinematics equation and presents a computational bottleneck of the algorithm. In order to accelerate the flow solver, we approximate the domain contribution of the kinematics integral equation by the cross approximation algorithm. The cross approximation method is used in combination with the hierarchical decomposition of the domain boundary combined by the hierarchical decomposition of domain interior. We propose to specify the approximation extent by controlling the depth of the hierarchical decomposition and the rank of the approximated integral matrix parts.

The developed algorithm is tested using the Arnold–Beltrami–Childress and lid driven cavity flows. We study the accuracy of boundary vorticity estimation and of the flow solution for different flow complexities (Reynolds number values), computational mesh densities and cross approximation settings. We found that that by using the cross approximation technique in the flow solver, we were able to reduce the computational demands of storing matrices to approximately 30% of the storage space of the original matrices. Furthermore, we showed that achieved approximation extent depends on the complexity of the simulated flow problem.

1. Introduction

There are many numerical methods that can be used to solve a differential partial equations (PDE). In computational fluid dynamics, the final volume method is often employed. To write the discrete counterpart of the governing PDE, the volume-based methods require discretization of the entire domain. The procedure results in a sparse system of linear equations for the unknown field functions. The boundary element method (BEM) is based on the Green's second theorem and by using the fundamental solution of the governing PDE a boundary integral equation may be derived. Thus, only the discretization of the boundary is necessary, [1]. However, due to the non-local nature of the fundamental solution, the resulting system of linear equations is fully populated. Due to this, the computational demands of the method scale with the square of the number of boundary unknowns, $O(N^2)$.

The fast boundary element method is a generic term describing a wide range of numerical algorithms, which aim to reduce computational demands from $O(N^2)$ to $O(N\log N)$ or even to $O(N)$. To accelerate the solution process and reduce the computational demand, we present an approximation procedure with a hierarchical structure. The hierarchical or $\mathcal{H}$ structure is a form that divides the full matrix formulation into smaller matrices on which an admissibility condition can be applied. Börm et al. [2] introduced the application of hierarchical matrices and the bounding box method for determining cluster trees. Hackbusch [3] wrote the arithmetic based on $\mathcal{H}$ matrices. Another hierarchical matrix form is the $\mathcal{H}^2$ matrix. Börm [4] used the $\mathcal{H}^2$ matrix formulation to construct efficient approximations of discretized integral formulation. There are many approximation procedures. The singular value decomposition is an efficient approximation method [5]. However, the computational demand to perform the SVD scales with $O(N^3)$ and is therefore not suitable for more demanding problems [6]. This makes the method unsuitable for larger problems. Another approach is the fast multipole method (FMM) that was presented by Rokhlin [7]. The FMM is based on the expansion of the integral kernel and thus requires a new implementation when the integral kernel is changed. The wavelet transform technique is another approximation method [8], which works algebraically and can be used to compress a wide variety of data and signals.

In the past years the cross approximation algorithm was developed. Different authors proposed different forms and variants of the algorithm.

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

http://dx.doi.org/10.1016/j.enganabound.2017.05.013
Received 13 March 2017; Received in revised form 10 May 2017; Accepted 22 May 2017
0955-7997/© 2017 Elsevier Ltd. All rights reserved.

The most widely used form is the algebraic cross approximation algorithm. The algebraic form includes the adaptive cross approximation ACA [9] and the extended adaptive cross approximation ACA+ [10]. A more extended formulation of the algebraic form is the multilevel adaptive cross approximation (MLACA) that was proposed by Tamayo [11]. MLACA is a procedure that divides the matrix into levels and applies the ACA algorithm on each level. In our work, we used the algebraic cross approximation algorithm for approximation of matrices for a collocation BEM discretization. Similarly, Bebendorf [12] used the cross approximation for the collocation BEM, while Rjasanow [13] used the cross approximation for the Galerkin BEM. Another form of the method is the hybrid cross approximation of integral operators (HCA) that was introduced by Börm and Grasedyck [14]. They have employed HCA on the $\mathcal{H}$ matrix representation of the finite element stiffens matrix. The $\mathcal{H}$ matrix structure is a hierarchical decomposition of the full matrix formulation and is used with all forms of the cross approximation. In this paper we propose the use of hierarchical decomposition approach of collocation BEM matrices and their approximation with the cross approximation algorithm.

The adaptive cross approximation is the most often used form of cross approximation. In recent years different physical problems were solved with this procedure. In this work we propose to use the cross approximation algorithm for the simulation of fluid flows. Other authors have applied it to other problems. For example, it was used to solve the eddy current problem by Smajic et al. [15]. Schröder et al. [16] used the adaptive cross approximation to solve the current distribution in an electromagnetic field. A coupled BEM and fine element numerical method was accelerated with the adaptive cross approximation to solve the distribution of a symmetric electromagnetic field by Kurz and Rjasanow [17]. Van et al. [18] used the ACA on different geometries, for solving the electromagnetic field with the coupled fine element-boundary integral numerical method. On the other hand Grytsenko and Galybin [19] used the adaptive cross approximation to solve a large number of cracks on a plate with the singular integral method and Maerton [20] employed ACA to solve a 3D elasticity problem. Besides the adaptive cross approximation, MLACA was also employed by Tamayo et.al [21] to solve the electromagnetic field distribution between to differently charged spheres. Zhao and Vouvakis [22] used the same technique to solve the magnetic behaviour of the ground plane design. Recently Wei et al. [23] solved the Laplace equation to predict the temperature distribution in a 2D domain by combining ACA with the singular boundary integral method. However, it has to be noted that Heldring et al. [24,25] revisited the problem of the ACA stopping criteria and proposed an alternative solution. In all publications, the authors discovered that the cross approximation method works efficiently towards the goal of reducing computational demand and storage requirements.

The aim of this work is to implement the cross approximation algorithm to accelerate a BEM based fluid flow solver. The fluid flow solver [26,27] has been developed for the velocity–vorticity formulation of the Navier–Stokes equations. To write this form we introduce the vorticity into the Navier–Stokes equations. The continuity equation is reformed into the kinematics equation and the equation of momentum conservation turns into the vorticity equation. Accounting for the Green's second identity, we can write the integral form of the two equations. The collocation scheme is employed and fully populated systems of linear equations must be solved.

To accelerate the solution of the kinematics equation we present an implementation of the cross approximation algorithm with a hierarchical structure. Our goal was to decrease the CPU time and memory storage demands. We test the implementation and assess its usefulness using two well-known fluid dynamics problems: the Arnold–Beltrami–Childress (ABC) flow [28] and the lid driven cavity problem [29,30]. The paper is divided into four sections, in the first chapter we introduce the governing equations, in the second we introduce the cross approximation and the $\mathcal{H}$ matrix structure. Finally, we test the implementation in the third section and summarize the results in the last section.

2. Governing equations

The governing equations for laminar incompressible fluid flow written in velocity–vorticity formulation are the kinematics equation and the vorticity transport equation. Let $\vec{v}$ be the velocity field and $\vec{\omega} = \vec{\nabla} \times \vec{v}$ the vorticity field. For a constant density fluid, the velocity field and the vorticity field are divergence free. With this approximation the kinematics equation may be stated as [26]:

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

Eq. (1) represents a connection between the velocity and vorticity vector field. It is an elliptic partial differential equation. The fluid movement is governed by the vorticity transport equation, written in a non-dimensional form as:

$$\frac{\partial \vec{\omega}}{\partial t} + (\vec{v} \cdot \vec{\nabla})\vec{\omega} = (\vec{\omega} \cdot \vec{\nabla})\vec{v} + \frac{1}{Re}\nabla^2 \vec{\omega}, \tag{2}$$

where $Re = vl/\nu$ is the Reynolds number and l the characteristic length scale. The fluid viscosity ν is also considered constant. The velocity boundary conditions are known, while the vorticity boundary conditions are calculated from (1).

The Eqs. (1) and (2) are solved in an boundary element based iterative algorithm presented in [31]. A domain decomposition approach is used to avoid the fully populated system of linear equations resulting from the vorticity transport equation (2). Domain decomposition can not be used for the solution of the kinematics equation for the unknown vorticity boundary conditions due to the fact that the Biot–Savart law states that the whole domain must be considered in order to keep the vorticity field divergence free.

Thus, in the following, we develop the cross approximation technique to accelerate the solution of the kinematics equation for the unknown vorticity boundary conditions.

2.1. The flow solver

Incompressible fluid flow, governed by a system of non-linear PDE's (1) and (2), is simulated using an iterative algorithm proposed by Ravnik et al. [27]. In this paper, we present an accelerated version of the first step - the estimation of boundary vorticity values (Section 2.2). In the second step, using sub-domain BEM solution of the kinematics equation (1) the velocity in the domain is calculated. Lastly, the vorticity transport equation (2) is solved for domain vorticity values using sub-domain BEM. The procedure is repeated until convergence for all field functions is achieved. To guarantee convergence under-relaxation is used. A value of 0.1 is used for problems with low Reynolds number value and 0.01 for problems with high Reynolds number value.

2.2. The kinematics equation for the boundary vorticity

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 form of the kinematics equation (1) without derivatives of the velocity and vorticity fields takes the following form, [27]:

$$c(\vec{\xi})\vec{v}(\vec{\xi}) + \int_\Gamma \vec{v}\,\vec{\nabla} u^\star \cdot \vec{n} d\Gamma = \int_\Gamma \vec{v} \times (\vec{n} \times \vec{\nabla}) u^\star d\Gamma + \int_\Omega (\vec{\omega} \times \vec{\nabla} u^\star) d\Omega. \tag{3}$$

where $u^\star(\vec{r}) = \frac{1}{4\pi|\vec{r}-\vec{\xi}|}$ is the fundamental solution of the Laplace equation and $\vec{\xi}$ is the source point. In order to use the kinematics equation to obtain boundary vorticity values, we must rewrite the Eq. (3) in a tangential form by multiplying the system with a normal in the source point $\vec{n}(\vec{\xi})$. This yields the following integral equation,

$$c(\vec{\xi})\vec{n}(\vec{\xi}) \times \vec{v}(\vec{\xi}) + \vec{n}(\vec{\xi}) \times \int_\Gamma \vec{v}\vec{\nabla} u^\star \cdot \vec{n} d\Gamma$$
$$= \vec{n}(\vec{\xi}) \times \int_\Gamma \vec{v} \times (\vec{n} \times \vec{\nabla}) u^\star d\Gamma + \vec{n}(\vec{\xi}) \times \int_\Omega (\vec{\omega} \times \vec{\nabla} u^\star) d\Omega. \tag{4}$$

which results in a non-singular system of linear equations for boundary vorticity values.

In order to write the discrete form of the kinematics equation, we divide the domain into elements $\Omega = \sum_{i=1}^{d} \Omega_i$ and the boundary into boundary elements $\Gamma = \sum_{i=1}^{b} \Gamma_i$, where d is the number of domain elements and b the number of boundary elements. At each mesh node, we interpolate the field functions using domain ϕ_i or boundary Φ_i shape functions. Quadratic interpolation of functions is used by 27-noded box domain elements and 9-noded quadrilateral boundary elements.

For each collocation node at the boundary $\vec{\xi}_f$, we calculate the following integrals over boundary element c:

$$h_{fcl} = \delta_{fcl} c(\xi_f) + \sum_{l=1}^{9} \int_{\Gamma_c} \Phi_l n_i \frac{\partial u^\star}{\partial x_i} d\Gamma_c, \tag{5}$$

$$h_{fcl}^{ij} = \sum_{l=1}^{9} \int_{\Gamma_c} \Phi_l \left[n_i \frac{\partial u^\star}{\partial x_j} - n_j \frac{\partial u^\star}{\partial x_i} \right] d\Gamma_c, \tag{6}$$

and the following integrals over domain elements e

$$d_{fel}^{i} = \sum_{l=1}^{27} \int_{\Omega_e} \phi_l \frac{\partial u^\star}{\partial x_i} d\Omega_e. \tag{7}$$

The indexes i, j in Eqs. (5)–(7) stand for the coordinates x, y, z. For the integral in Eq. (6) $i \neq j$. The symbol δ_{fcl} is the Kronecker delta, which is equal to 1 when source point and the mesh point are equal.

Components h_{fcl} are parts of matrix $[H]$, h_{fcl}^{ij} defines the matrices $[H_{ij}]$ and components d_{fel}^{i} define the matrix $[D_i]$. Considering this, we can write a system of equations, where matrices are shown as $[\,\cdot\,]$ and vectors are shown as $\{\,\cdot\,\}$:

$$\begin{aligned}
[H]\{v_x\} &= [H_{yx}]\{v_y\} - [H_{zx}]\{v_z\} + [D_z]\{\omega_y\} - [D_y]\{\omega_z\} \\
[H]\{v_y\} &= [H_{zy}]\{v_z\} - [H_{yx}]\{v_x\} + [D_x]\{\omega_z\} - [D_z]\{\omega_x\} \\
[H]\{v_z\} &= [H_{xz}]\{v_x\} - [H_{zy}]\{v_y\} + [D_y]\{\omega_x\} - [D_x]\{\omega_y\}
\end{aligned} \tag{8}$$

In order to separate the unknown boundary vorticity values, the vorticity vector can be written as a sum of boundary and domain vorticity as $\{\omega_i\} = \{\omega_i\}_\Gamma + \{\omega_i\}_{\Omega/\Gamma}$. The vector $\{\omega_i\}_\Gamma$ includes the vorticity at nodes on the boundary and the vector $\{\omega_i\}_{\Omega/\Gamma}$ the vorticity at nodes in the domain. Hence, we can write the system of equations like this:

$$\begin{aligned}
&([n_x][D_x] + [n_y][D_y] + [n_z][D_z])\{\omega_x\}_\Gamma = ([n_y][H_{zx}] + [n_z][H_{xy}])\{v_x\} \\
&\quad + ([n_y][H] - [n_z][H_{yz}])\{v_z\} - ([n_x][H] + [n_y][H_{yz}])\{v_y\} \\
&\quad + [n_z][D_x]\{\omega_z\}_\Gamma + [n_y][D_x]\{\omega_y\}_\Gamma + [n_x][D_x]\{\omega_x\}_\Gamma - ([n_y][D_y]_{\Omega/\Gamma} \\
&\quad + [n_z][D_z]_{\Omega/\Gamma})\{\omega_x\}_{\Omega/\Gamma} + [n_y][D_x]_{\Omega/\Gamma}\{\omega_y\}_{\Omega/\Gamma} + [n_z][D_x]_{\Omega/\Gamma}\{\omega_z\}_{\Omega/\Gamma},
\end{aligned} \tag{9}$$

$$\begin{aligned}
&([n_x][D_x] + [n_y][D_y] + [n_z][D_z])\{\omega_y\}_\Gamma = ([n_z][H_{xy}] + [n_x][H_{zy}])\{v_y\} \\
&\quad + ([n_x][H] - [n_x][H_{zx}])\{v_x\} - ([n_x][H] + [n_z][H_{zx}])\{v_z\} \\
&\quad + [n_z][D_y]\{\omega_z\}_\Gamma + [n_x][D_y]\{\omega_x\}_\Gamma + [n_y][D_y]\{\omega_y\}_\Gamma - ([n_z][D_z]_{\Omega/\Gamma} \\
&\quad + [n_x][D_x]_{\Omega/\Gamma})\{\omega_y\}_{\Omega/\Gamma} + [n_z][D_y]_{\Omega/\Gamma}\{\omega_z\}_{\Omega/\Gamma} + [n_x][D_y]_{\Omega/\Gamma}\{\omega_x\}_{\Omega/\Gamma},
\end{aligned} \tag{10}$$

$$\begin{aligned}
&([n_x][D_x] + [n_y][D_y] + [n_z][D_z])\{\omega_z\}_\Gamma = ([n_x][H_{yz}] + [n_y][H_{xz}])\{v_z\} \\
&\quad + ([n_x][H] - [n_y][H_{xy}])\{v_y\} - ([n_y][H] + [n_x][H_{xy}])\{v_x\} \\
&\quad + [n_y][D_z]\{\omega_z\}_\Gamma + [n_x][D_z]\{\omega_x\}_\Gamma + [n_z][D_z]\{\omega_z\}_\Gamma - ([n_x][D_x]_{\Omega/\Gamma} \\
&\quad + [n_y][D_y]_{\Omega/\Gamma})\{\omega_z\}_{\Omega/\Gamma} + [n_x][D_z]_{\Omega/\Gamma}\{\omega_x\}_{\Omega/\Gamma} + [n_y][D_z]_{\Omega/\Gamma}\{\omega_y\}_{\Omega/\Gamma}.
\end{aligned} \tag{11}$$

Matrices $[n_x]$, $[n_y]$, $[n_z]$ are diagonal matrices, that include the components of the normal vector $\vec{n} = \{n_x,\ n_y,\ n_z\}$. The size of matrices $[H]$ and $[H_{ij}]$ is $n \times n$, where n is the number mesh nodes on the boundary.

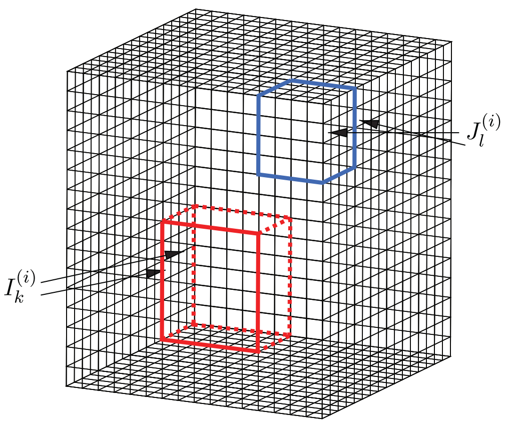

Fig. 1. A graphical representation of the boundary and domain cluster three forming algorithm.

since we decomposed the vorticity vector into boundary and domain parts we get two types of domain matrices. The first are $[D_i]$ which are of the size $n \times n$, while the second $[D_i]_{\Omega/\Gamma}$ are of the size $n \times m$. Here m is the number of interior domain nodes equal to the number of all nodes minus the number of boundary nodes.

As the size of $[D_i]_{\Omega/\Gamma}$ matrices is one order of magnitude greater than all other matrices, we implemented the cross approximation algorithm for these matrices.

3. Acceleration of the solution of the kinematics equations

Let us consider one of the integral matrices $[D_i]_{\Omega/\Gamma}$ and denote it as D. It's size is $n \times m$ and its entries are real numbers, $D \in \mathrm{IR}^{n \times m}$. By defining a geometrical condition, based on the properties of the fundamental solution and the domain shape, we can split the matrix $D|_{n \times m}$ into smaller matrices $\hat{D}|_{\hat{n} \times \hat{m}}$. As the matrix forms due to the source point at the boundary and field point in the domain, the splitting of both the domain and the boundary must be performed.

3.1. Block cluster trees

The computational domain is meshed with box elements. Each element represent a leaf on a tree of clusters of elements. Let these elements represent cluster at level 1 of the cluster tree. In order to build the cluster tree, we propose a recursive algorithm, which groups neighbouring elements into higher generation clusters. The procedure is repeated until a single cluster emerges, which included all elements in the computational mesh. The total number of levels p defines the depth of a cluster tree. The following algorithm was used to build the domain cluster tree T_I

- for all levels $i = 1$ to p
- for all clusters $I_k^{(i)}$ at level i
- find $I_k^{(i)}$'s eight neighbouring clusters and join them into a new cluster at level $i + 1$

The number of clusters in each level increases like this, $a = b = 8^{(p-i)}$, $(i \leq p)$. This algorithm yields a cluster tree with a moderate number of generations p and a high number of clusters in each generation. The clusters at level i are sons of a father, which is a cluster at level $i + 1$.

This algorithm is used again to build the boundary cluster tree T_J by starting with boundary elements as first level leaves. The cluster forming algorithm is shown in Fig. 1. Boundary T_J and domain T_I cluster trees are joined by combining clusters of both trees at the same levels to form a block cluster tree $T_{I \times J}$.

For each domain cluster in T_I cluster tree we define a bounding box $Q_{dm} \subseteq \mathrm{IR}^3$, which encompassed all domain nodes in the cluster as proposed by Börm [2]. Similarly, for each boundary cluster in the T_J cluster

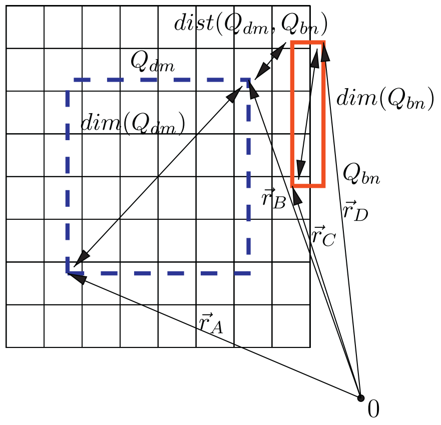

Fig. 2. The implementation of the bounding box method and determination of distances between clusters.

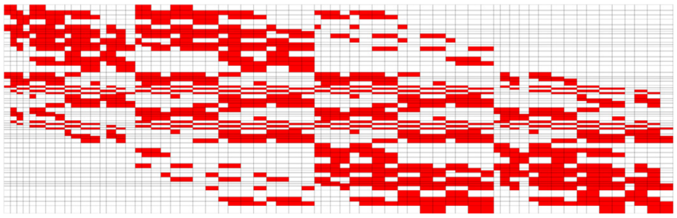

Fig. 3. In this Figure we can see the $\mathcal{H}$ structure of the 25^3 nodes mesh, where $\eta = 1$ was used in the admissibility condition.

tree we define a bounding box $Q_{bn} \subseteq \mathrm{IR}^3$ from the boundary nodes in the cluster. Fig. 2 shows the boundary box definition on a 2D cross-section of the domain.

3.2. $\mathcal{H}$-matrix

From the block cluster tree we build a $\mathcal{H}$ matrix, which has the recursive structure of the block cluster tree and represents an approximation of the original matrix. The matrix structure includes several parts in which we try to approximate the original matrix part. We decide, which part is admissible for approximation by employing the admissibility condition. The admissibility condition is:

$$min\left\{dim(Q_{dm}), dim(Q_{bn})\right\} \le \eta\ dist(Q_{dm}, Q_{bn}). \tag{12}$$

Expressions $dim(Q_{dm}) = |\vec{r}_A - \vec{r}_B|$ and $dim(Q_{bn}) = |\vec{r}_C - \vec{r}_D|$ are the diameters of boundary and domain bounding boxes. The distance $dist(Q_{dm}, Q_{dn})$ is the minimal distance between nodes in the boundary and domain clusters. These dimensions are shown in Fig. 2.

The parameter η is the admissibility parameter, which is defined by the user. This parameter enables the user to vary the admissibility condition and control the number of approximated matrix parts. In our simulations we used $\eta = 1$, based on the recommendation given by Wei et al. [23].

The blocks $(I_k \times J_l)^{(i)}$, which meet the admissibility condition are called admissible leaves and cross approximation is used to approximate this matrix part.

In Fig. 3 we show an illustration of a $\mathcal{H}$ matrix structure. The red boxes are the matrices that do not fulfil the admissibility condition and the white boxes are the matrices which are admissible.

3.3. Cross approximation

Cross approximation is a method that approximates matrices $\hat{D}|_{\hat{n}\times\hat{m}}$, that meet the admissibility condition in Eq. (12), into a *RK*-form. Let the rank of the matrix $\hat{D}$ be k and let us denote the approximation rank as r. Every matrix that has the rank $k > 1$, can be approximated into a *RK*-matrix using approximation rank r in the following way [2]:

$$\widetilde{D}|_{r(\hat{n}+\hat{m})} \approx \hat{A}\hat{B}^T,\ \hat{A} \in \mathbb{R}^{\hat{n}\times r},\ \hat{B} \in \mathbb{R}^{r\times\hat{m}}. \tag{13}$$

In order to store the *RK*-matrix into memory and perform matrix-vector multiplications, we need $O(r(\hat{n} \times \hat{m}))$ computational resources, [10]. The approximation rank r is the number of rows in matrix A and the number of columns in matrix B^T. The cross approximation algorithm is written bellow:

```
1. $\hat{R}^0 = \hat{D}|_{\hat{n}\times\hat{m}}$
2. The loop u=0,1,2,3,..., $r$
   (a) $(i^*, j^*)^u = ArgMax\left|(\hat{R}^u)\right|$
   (b) $\gamma^{u+1} = (\hat{R}^u_{i^*,j^*})^{-1}$
   (c) $a^{u+1} = \gamma^{u+1}\hat{R}^u_{i,j^*},\ b^{u+1} = (\hat{R}^u_{i^*,j})^T$
   (d) $\hat{R}^{u+1} = \hat{R}^u - a^{u+1}b^{u+1}$
```

In point (1) of the algorithm, we define the residual matrix. The second point (2) is a start of a loop that preforms the approximation. In (a) the largest element in the matrix $\hat{R}^u$ is found. Next, in (b), the inverse of the largest element $\hat{R}^u_{i^*,j^*}$ is calculated. In (c) vectors a^{u+1} and b^{u+1} are defined. This two vectors build a cross around the element $\hat{R}^u_{i^*,j^*}$. (d) calculates a new residual matrix $\hat{R}^{u+1}$.

The described cross approximation algorithm is an iterative method, for which we need a way to determine the approximation rank r in order to stop the iterations. Bebendorf [32] proposed an adaptive determination of the stopping criteria that is most widely used also by other authors. Heldring revised the problem of the stopping criteria in [24,25]. We propose to use a user defined compression factor α, which defines the approximation rank r in the following way:

$$r = k \cdot \alpha, \tag{14}$$

where k is the rank of the matrix part $\hat{D}|_{\hat{n}\times\hat{m}}$. Based on r we can calculate the compression ratio φ

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

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

4. Numerical experiments

We tested the proposed algorithm using two established test cases: the Arnold–Beltrami–Childress (ABC) flow [28] and the lid driven cavity problem [29,30]. The domain in both cases was a unit cube. We sought steady state flow solutions using four different mesh densities, having $17^3, 25^3, 41^3$ and 49^3 nodes. In order to facilitate the comparison between simulations with and without compression, we based the choice of fine mesh density on the availability of computer storage to store the full uncompressed matrices. We chose the middle and coarse mesh based on previous experiences with simulation of lid driven cavity flows, [31]. The locations of nodes were concentrated towards the corner of the domain using a geometric series with the ratio between the longest and shortest element size being 1: 3 in the 17^3 mesh and 1: 6.5 in others.

4.1. Arnold–Beltrami–Childress flow

The ABC flow is a time independent solution of the Euler's equations for three dimensional incompressible inviscid fluid flow. The flow is a prototype for the research of the turbulent flow [28]. The velocity vector field is defined like this,

$$\begin{aligned} v_x &= A \cdot sin(zv) + C \cdot cos(yv), \\ v_y &= B \cdot sin(xv) + A \cdot cos(zv), \\ v_z &= C \cdot sin(yv) + B \cdot cos(xv), \end{aligned} \tag{16}$$

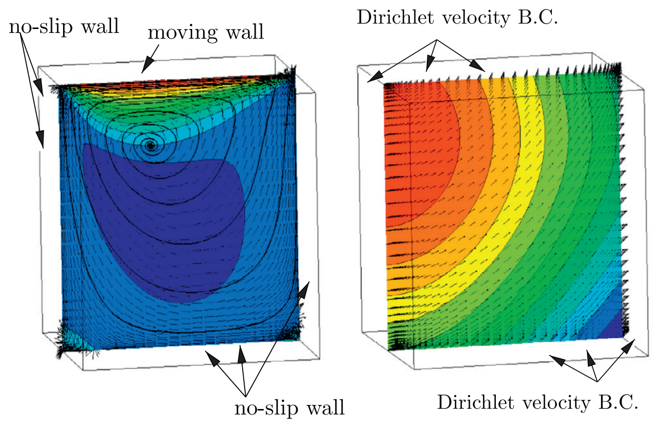

Fig. 4. Flow structure and boundary conditions for the lid driven cavity ($Re = 100$) (left) and Arnold–Beltrami–Childress flow ($\nu = 2$) (right). Colour denotes velocity component magnitude.

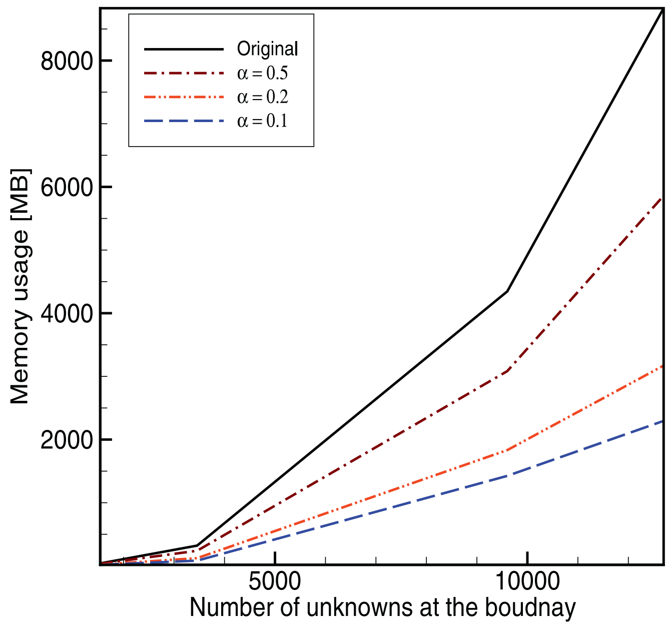

Fig. 5. Memory demand versus the number of boundary unknowns. Memory usage of the original uncompressed matrix is compared with approximated matrices at different compression factor α.

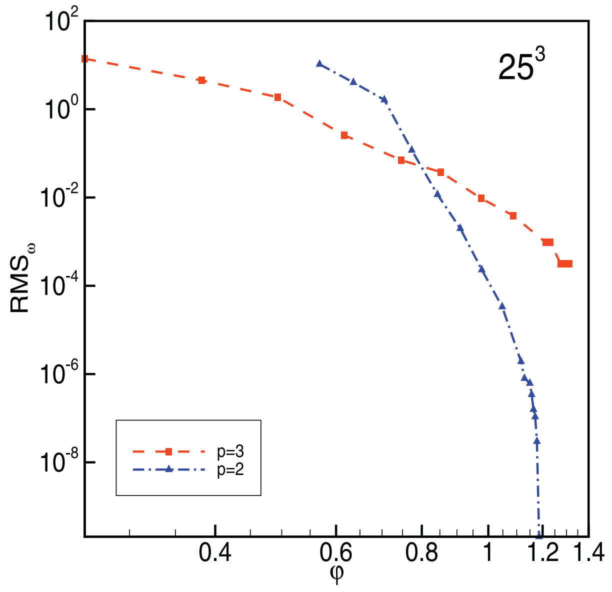

Fig. 6. Influence of the block cluster tree depth on the simulation accuracy at different compression ratios. A deeper block cluster tree yields lower RMS_{ω} due to an increased number of admissible blocks. Results of the ABC flow test case are shown on the 25^3 mesh for $A = B = C = 1$ and $\nu = 0.1$.

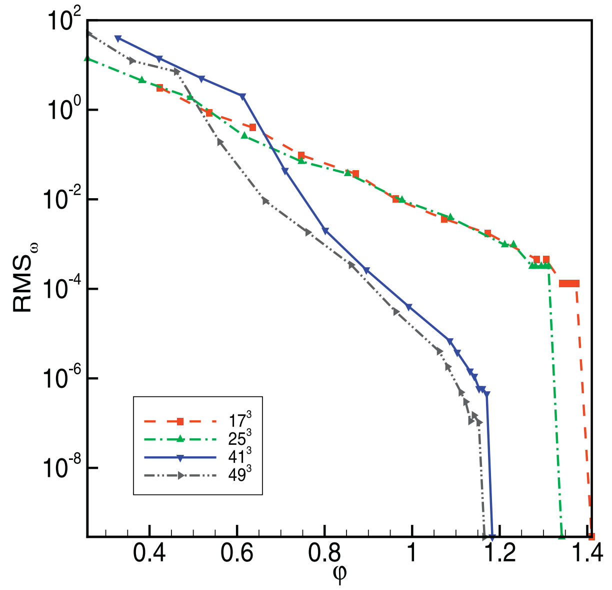

Fig. 7. Norm RMS_{ω} for ABC flow at different mesh densities. Mesh independent behaviour is observed for high compression. Norm limits towards machine precision at zero compression.

while the analytical expression for the vorticity field may be readily derived. Symbols A, B, C in Eq. (16) are the amplitudes and ν is the frequency. We used the analytical ABC flow solution and based on the known domain vorticity and velocity calculated the boundary vorticity values by using the developed cross approximation accelerated BEM for the kinematics equation.

We were manipulating the frequency and the amplitude to get different structures of the ABC flow. The frequency parameter of ABC flow determines the number and size of vortices in the problem domain. The amplitude parameter defines the maximal velocity of the fluid. In order to make a parametric study, we chose frequencies from $0.001, 0.1, 1$ to 2, at a constant amplitude A, B, C equal to 1.0. The amplitude was varied from $0.001, 0.1, 1$ to 2 at a constant frequency of $\nu = 0.1$.

4.2. Lid driven cavity flow

Flow in a 3D lid driven cavity is one of the standard benchmark test cases used in the development of flow solvers. The domain as well as the boundary conditions are unambiguously defined and do not change with the Reynolds number. The flow exhibits a wide variety of phenomena, such as eddies, complex three-dimensional patterns and instabilities (Shankar and Deshpande [33]). The research of lid driven cavity flow started with the observations of Koseff and Street [34], who were able to observe the inherent 3D nature of flow phenomena in the cavity.

The simulation was performed on a cavity with no-slip velocity boundary conditions are employed on all wall except the top wall, where a constant velocity in x direction is prescribed. Dirichlet type boundary conditions are used for the vorticity transport equation. Vorticity on the boundary is obtained by the solution of the cross approximation accelerated kinematics equation for all directions and walls, except for $\omega_x = 0$ on left and right walls, $\omega_y = 0$ on the front and back walls and $\omega_z = 0$ on top and bottom walls. The flow was simulated at three Reynolds number

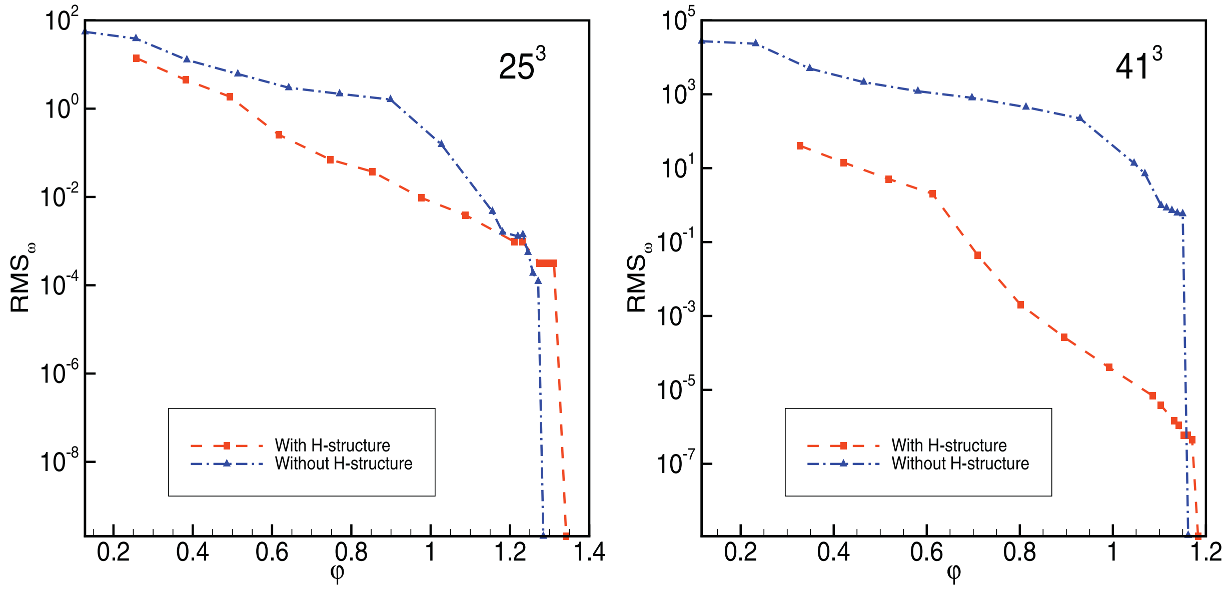

Fig. 8. Importance of the hierarchical matrix structure for the accuracy of the cross approximation.

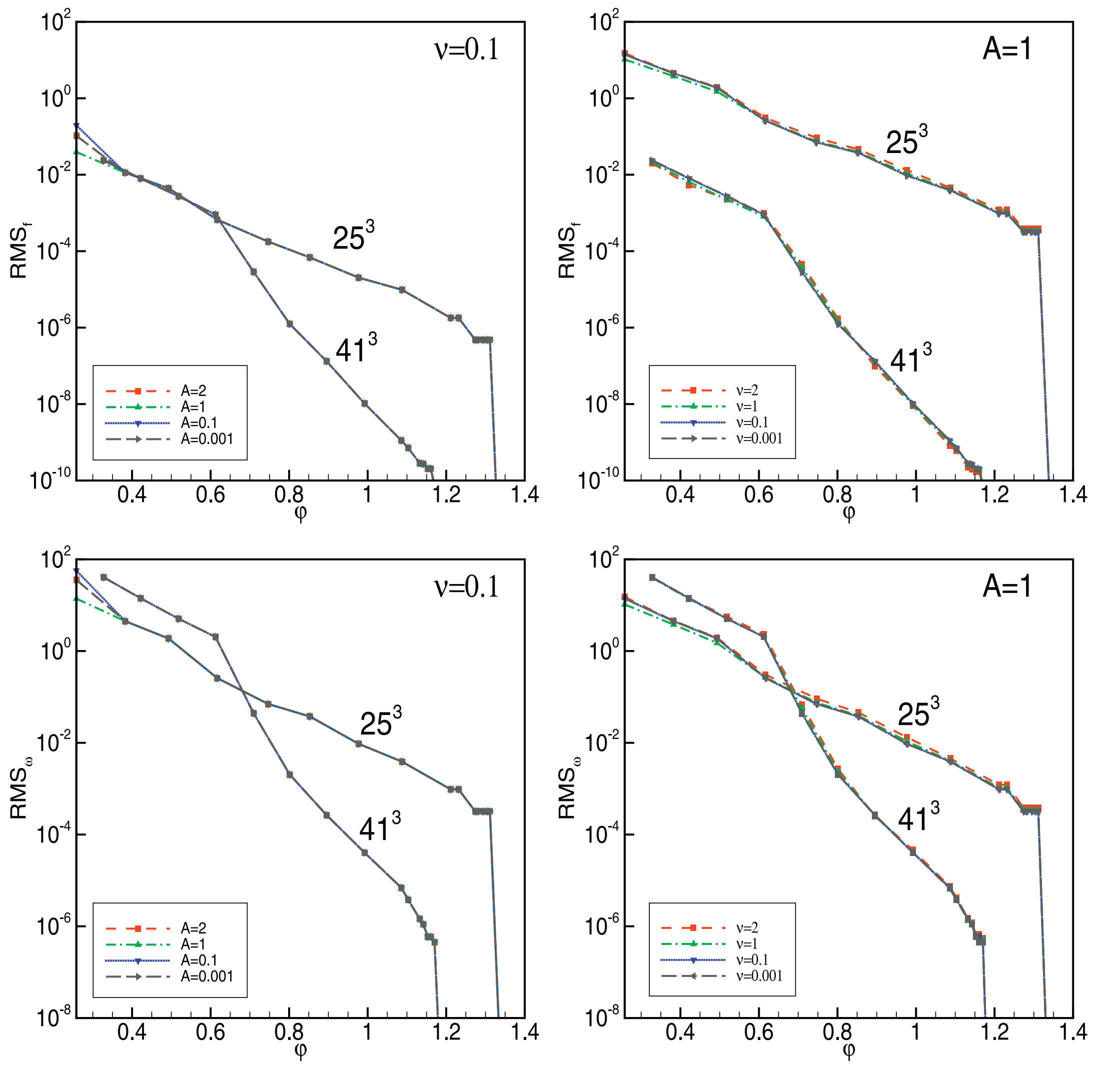

Fig. 9. Influence of the ABC flow structure on the RMS_f and RMS_ω norms. A course and a fine mesh are considered with changing amplitude $A = B = C$ and frequency ν of the vortical structures.

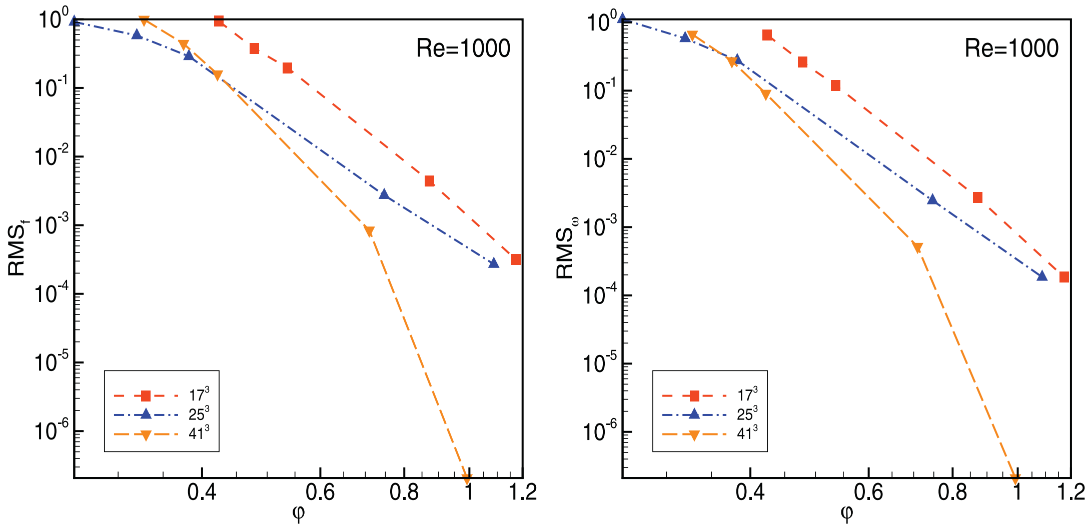

Fig. 10. Lid driven cavity test case at $Re = 1000$. Norms versus compression ratio are shown for three meshes having 17^3, 25^3 and 41^3 nodes.

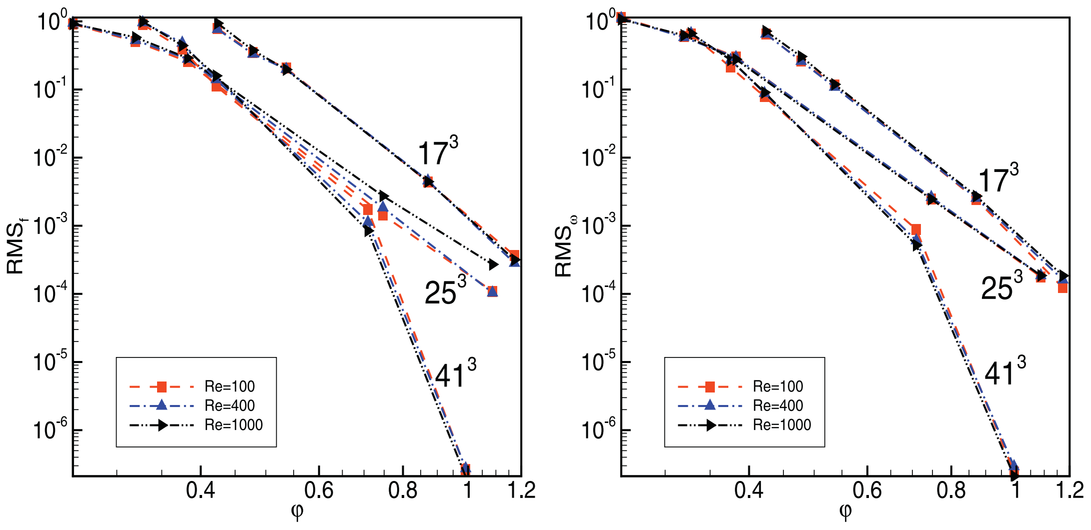

Fig. 11. Influence of the Reynolds number on the accuracy of solution of the boundary vorticity values in the lid driven cavity problem. The RMS_f and RMS_ω are shown in two panels. Three computational grids are considered having 17^3, 25^3 and 41^3 nodes for three Reynolds number values $Re = 100, 400$ and 1000.

values: $Re = 100$, 400 and 1000. Boundary conditions and a representation of the flow structure is shown in Fig. 4 for both test cases.

Due to inherent non-linearity of the flow problem, the driven cavity problem was solved using an iterative procedure. The iterations were stopped when the difference between subsequent iteration of all field functions dropped below 10^{-6}.

4.3. Definition of norms

The cross approximation was used on the matrices $[D_x]_{\Omega/\Gamma}$, $[D_y]_{\Omega/\Gamma}$, $[D_z]_{\Omega/\Gamma}$ in Eqs. (9)–(11). In order to measure the influence of the approximation on the result and find its dependence on the flow structure and mesh density, we used two norms. We measure the difference between boundary vorticity values as

$$RMS_\omega = \left(\sum_{j=x,y,z} \frac{\sum_i (\omega_{ji} - \omega_{Aji})^2}{\sum_i (\omega_{Aji})^2} \right)^{\frac{1}{2}} \tag{17}$$

where ω_{ji} is the jth component of vorticity in i^{th} boundary node calculated without an approximation and the ω_{Aji} its approximated counterpart. In addition, we calculate

$$RMS_f = \sqrt{\frac{\sum_{i=1}^{N} (f_{Ai} - f_i)^2}{\sum_{i=1}^{N} (f_i)^2}}, \tag{18}$$

which measures the difference between the right hand side of the linear system of equations for the z component of boundary vorticity (11). f_{Ai} is the right hand side calculated with the use of cross approximation, f_i its non approximated counterpart.

5. Results and discussion

In this section, we illustrate the impact of the cross approximation method on the ABC and lid driven cavity flow solutions. All simulations were performed in double precision on a workstation with an Intel Xenon 64-*bit* processor and 64 *GB* of memory. Flow structures were changed, to see how a different compression factor α, effects the norms RMS_ω and RMS_f.

Fig. 5 illustrates the memory demands needed to store $[D_z]_{\Omega/\Gamma}$ matrix at different compression factors as well as the original fully populated matrix. We observe that the memory demands of the original matrix

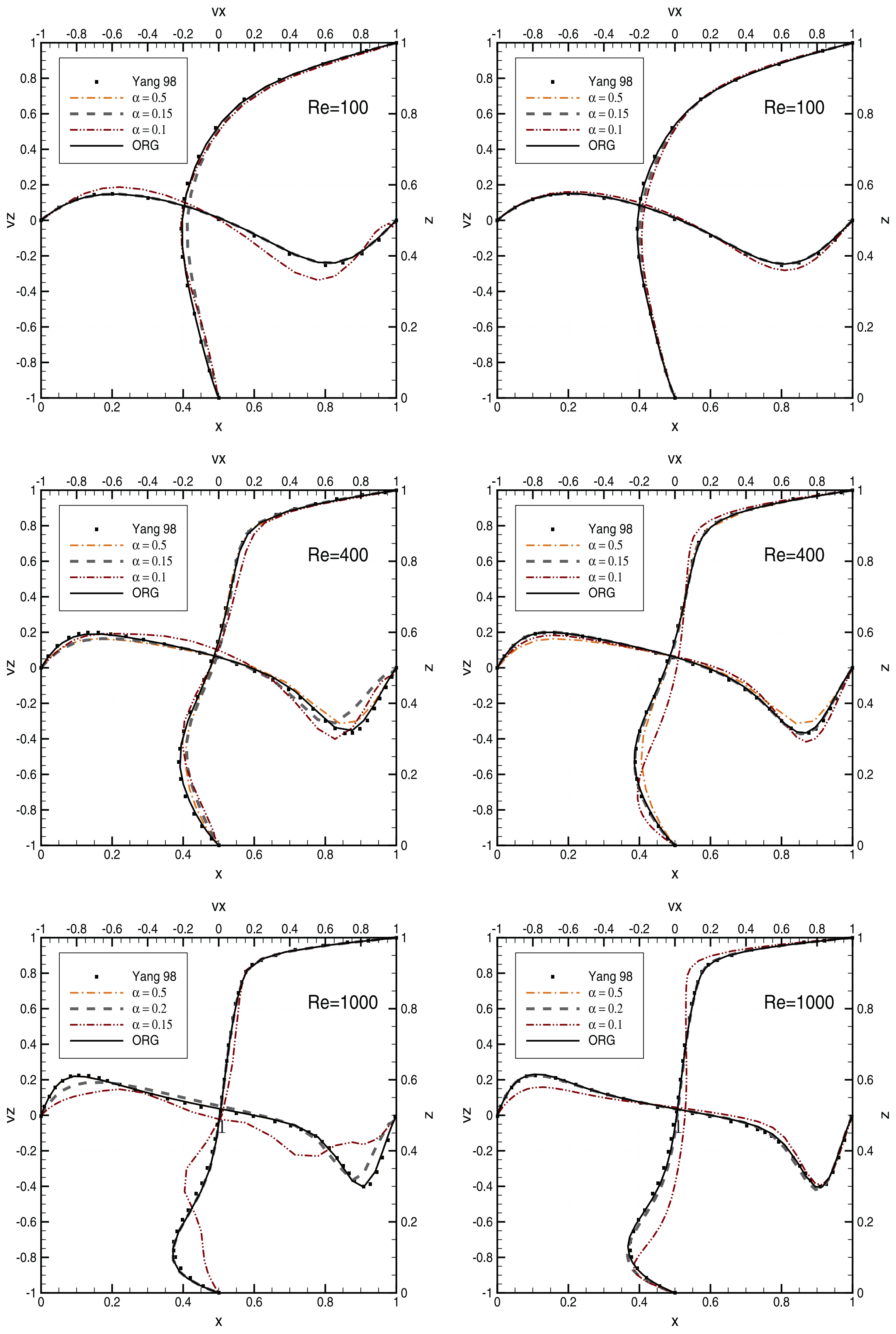

Fig. 12. Velocity profiles $v_x(z)$ and $v_z(x)$ at the $y = 0.5$ plane, for different compressions ratios and Reynolds numbers. Results obtained using the 25^3 mesh are shown left, 41^3 mesh is on the right.

scale quadratically, while the compressed matrices exhibit quasilinear dependence.

Each block cluster tree $T_{I\times J}$ can have a different number of generations p. The number of generations is defined by the stopping criterion. In Fig. 6 we show how a different generation depth p of a cluster tree effects the RMS_{ω}. It can be seen that the RMS_{ω} difference at a chosen compression ratio is lower when a deeper hierarchical matrix structure is used.

In Fig. 7 we present the RMS_{ω} for the ABC flow at different mesh densities. The flow was simulated with $A = B = C$=1.0 and $\nu = 0.1$. We observe similar behaviour on all meshes at high compression ratio ($\varphi < 0.5$), while at low compression the dense meshes yield significantly more accurate results than the coarse meshes. When considering a coarse mesh, the cluster tree depth is limited by the number of domain elements. As shown in Fig. 6 the cluster tree depth also has an influence on RMS_{ω}.

Employing hierarchical matrix structure is important for achieving high accuracy of the approximation. We compare two approximations one with the hierarchical matrix structure and one without. A course and a fine mesh were considered. The result is illustrated in Fig. 8. It can be seen that the approximation with the hierarchical structure is more accurate at a given compression ratio. In Fig. 9 we show how different features of the ABC flow influence the norm RMS_f by changing the amplitude and frequency of the vortical flow structures. We consider changing of the amplitude while keeping the frequency ν constant and in vice-versa. Plots of norms RMS_f and RMS_{ω} reveal that flow structure has virtually no influence on the norms. As expected the norms decrease with compression and reach machine precision at zero compression. Comparing these results on course and dense meshes also gives no difference. The reason for this is the fact that the approximation is based on the domain geometry and integral calculation and does not depend on the flow. As the boundary vorticity for the ABC flow case is solved by a single step solution using analytical domain flow field, the solution accuracy is expected to be independent of the flow. This observation is clearly confirmed by the norms in Fig. 9.

In Fig. 10 we consider the lid driven cavity test case and show the RMS_f and RMS_{ω} norms for Reynold number $Re = 1000$ and three mesh densities. In all cases, the norms decrease with decreasing compression. At a given Reynolds number, we observe that the norms are lower for a fine mesh as compared to a coarse mesh. This means that a simulation on a fine mesh is able to use more compression to reach the same accuracy of results compared to a simulation on a coarse mesh.

In Fig. 11 we compare the norms at three Reynolds number values, $Re = 100$, $Re = 400$ and $Re = 1000$. A change in the Reynolds number of the flow significantly alters the flow structure in the cavity. With an increase in Re the main vortex in the cavity dominates and gives rise to secondary and tertiary vortices at the bottom corners of the cavity. A look at the norms in Fig. 11 reveals that the boundary vorticity solution accuracy is unaffected by the Reynolds number.

Even though the increase of the Reynolds number requires the iterative algorithm to resolve the non-linear nature of the flow by an increased number of iteration, the error made by the cross approximation algorithm does not propagate through the iterative algorithm and thus remains unaffected by flow the Reynolds number.

Using norms we focused on the impact of the cross approximation algorithm on the solution of the boundary vorticity values. The approximation of the boundary vorticity values affects the whole velocity field. In Fig. 12 we show velocity profile through the centre of the cavity. We compare the benchmark results of Yang et al. [30], our method, where approximation has not been used and the results obtained using the cross approximation. We observe very good agreement between the benchmark and the results obtained without approximation. When approximation is used, the difference in profiles increases. We observe that the difference also increases with Reynolds number and decrease with mesh density. Two main conclusions may be drawn: for a given flow problem (Reynolds number) a denser mesh enables higher compression ratios and for a given mesh and increase of Reynolds number requires a decrease of the compression ratio.

6. Conclusions

In this study we developed a cross approximation accelerated boundary element based solution of the kinematics equation for evaluation of the boundary vorticity values using a hierarchical integral matrix structure. The developed algorithm is included into a boundary element incompressible flow solver, which solves the velocity-vorticity formulation of the Navier–Stokes equation in 3D. We have shown that the cross approximation algorithm can be successfully introduced to accelerate the solution of the kinematics equation. The implementation was tested using an analytical Arnold–Beltrami–Childress flow example and the 3D lid driven cavity test case with the Reynolds number up to 1000.

The results showed that for a chosen solution accuracy the extent of integral matrix compression depends on the computational mesh used. A finer mesh allows usage of higher compression ratios. We propose to control the error introduced into the boundary vorticity by the approximation by specifying the rank of the approximated matrix parts.

Even though the one over r elliptic integral kernel, which is found in the integral form of the kinematics equation, is not very well suited for approximation, we can conclude that the application of the cross approximation enables the usage of the flow solver on finer meshes at reduced computational cost and thus enables simulation of more complex flow problems.

Acknowledgements

The authors acknowledge the financial support from the Slovenian Research Agency (research core funding no. P2-0196).

References

[1] Wrobel LC. The boundary element method. John Wiley and Sons Ltd; 2002.
[2] Börm S, Grasedyck L, Hackbusch W. Introduction to hierarchical matrices with applications. Eng Anal Bound Elem 2003;27(5):405–22.
[3] Hackbusch W, Leipzig. A sparse matrix arithmetic based on H -matrices. Computing 1999;108:89–108.
[4] Börm S. Approximation of integral operators by H2-matrices with adaptive bases. Computing 2005;74(3):249–71.
[5] Grasedyck L, Hackbusch W. Hierarchical matrices, 21. Leipzig: Max-Planck-institut für Mathematik in den Naturwissenschaften; 2006.
[6] Kalman D. A singularly valuable decomposition : the SVD of a matrix. Washington: The America University; 2002.
[7] Rokhlin V. Rapid solution of integral equations of classical potential theory. J Comput Phys 1985;60(2):187–207.
[8] Daubechies I. Ten lectures on wavelets. Soc Industr Appl Math; 1992. p. 357.
[9] Bebendorf M. Approximation of boundary element matrices. Numer Math 2000;86(4):565–89.
[10] Rjasanow S, Steinbach O. The fast solution of boundary integral equations. New York: Springer Science + Business Media; 2007.
[11] Tamayo J, Heldring A, Rius J. Multilevel adaptive cross approximation (MLACA). In: IEEE antennas and propagation society international symposium n antennas and propagation. Charleston, SC: Dept. of Signal Theory; 2009a. p. 2008–11.
[12] Bebendorf M, Rjasanow S. Adaptive low-rank approximation of collocation matrices. Computing 2003;70(1):1–24.
[13] Rjasanow S. Adaptive cross approximation of dense matrices. In: Proceedings of the IABEM 2002 Symposium; 2002. p. 1–12.
[14] Börm S, Grasedyck L. Hybrid cross approximation of integral operators. Numer Math 2005;101(2):221–49.
[15] Smajic J, Andjelic Z, Bebendorf M. Fast BEM for eddy-current problems using H-matrices and adaptive cross approximation. IEEE Trans Magn 2007;43(4):1269–72.
[16] 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.
[17] Kurz S, Rain O, Rjasanow S. Application of the adaptive cross approximation technique for the coupled BE-FE solution of symmetric electromagnetic problems 2003;32(4):423–9.
[18] Van T, Suzuki LRC, Latypov D, Holle JV, Voss T, Van T, et al. Fast Algebraic Methods in Computational Electromagnetics. In: Proceedings of the national IEEE aerospace and electronics conference (NAECON), Dayton; 2010. p. 230–6.
[19] 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.

[20] 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.
[21] Tamayo JM, Heldring A, Rius JM. Application of multilevel adaptive cross approximation (MLACA) to electromagnetic scattering and radiation problems. In: Proceedings of the international conference on electromagnetics in advanced applications, 2009. ICEAA '09, Torino; 2009b. p. 178–81.
[22] Zhao K, Vouvakis M. Application of the multilevel adaptive cross-approximation on ground plane designs. Electromagn Compat 2004;1:124–7.
[23] Wei X, Chen B, Chen S, Yin S. An ACA-SBM for some 2D steady-state heat conduction problems. Eng Anal Bound Elem 2016;71:101–11.
[24] Heldring A, Ubeda E, Rius J. On the convergence of the ACA algorithm for radiation and scattering problems. IEEE Trans Antennas Propag 2014;62(7):3806–9.
[25] Heldring A, Ubeda E, Rius JM. Stochastic estimation of the Frobenius norm in the ACA convergence criterion. IEEE Trans Antennas Propag 2015;63(3):1155–8.
[26] 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.
[27] 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):4517–27.
[28] Ershkov SV. About existence of stationary points for the Arnold–Beltrami–Childress (ABC) flow. Appl Math Comput 2016;276:379–83.
[29] Botella O, Peyret R. Benchmark spectral results on the lid-driven cavity flow. Comput Fluids 1998;27(4):421–33.
[30] Yang J-y, Yang S-c, Chen Y-n, Hsu C-a. Implicit weighted ENO schemes for the three--dimensional incompressible Navier–Stokes equations 1998;487:464–87.
[31] Ravnik J, Škerget L, Zunič Z. Combined single domain and subdomain BEM for 3D laminar viscous flow. Eng Anal Bound Elem 2009;33(3):420–4.
[32] Bebendorf M, Grzibovski R. Accelerating Galerkin BEM for linear elasticity using adaptive cross approximation. Leipzig: Max-Planck-institut für Mathematik in den Naturwissenschaften; 2006.
[33] Shankar P N and Deshpande MD. Fluid mechanics in the driven cavity. Ann Fluid Mech 2000;32:93–136.
[34] Koseff J R and Street RL. The lid-driven cavity flow: a synthesis of qualitative and quantitative observations. J Fluids Eng 1984;106:385–9.