ELSEVIER

Engineering Analysis with Boundary Elements 28 (2004) 1303–1314

ENGINEERING ANALYSIS *with* BOUNDARY ELEMENTS

www.elsevier.com/locate/enganabound

The wavelet transform for BEM computational fluid dynamics

J. Ravnik*, L. Škerget, M. Hriberšek

Faculty of Mechanical Engineering, Institute of Power, Process and Environmental Engineering, University of Maribor, Smetanova 17, SI-2000 Maribor, Slovenia

Received 30 January 2004; revised 27 April 2004; accepted 11 May 2004

Available online 10 July 2004

Abstract

A wavelet matrix compression technique was used to solve systems of linear equations resulting from BEM applied to fluid dynamics. The governing equations were written in velocity–vorticity formulation and solutions of the resulting systems of equations were obtained with and without wavelet matrix compression. A modification of the Haar wavelet transform, which can transform vectors of any size, is proposed. The threshold, used for making fully populated matrices sparse, was written as a product of a user defined factor κ and the average value of absolute matrix elements values. Numerical tests were performed to assert, that the error caused by wavelet compression depends linearly on the factor κ, while the dependence of the error on the share of thresholded elements in the system matrix is highly non-linear. The results also showed that the increasing non-linearity (higher *Ra* and *Re* number values) limits the extent of compression. On the other hand, higher mesh density enables higher compression ratios.

Keywords: Boundary elements; Velocity–vorticity formulation; Discrete wavelet transform; Modified Haar wavelet transform; Linear systems of equations; System matrix compression; Driven cavity; Natural convection

1. Introduction

One of the main impediments of the boundary element method is the need for solving large fully populated linear systems of equations resulting from numerical discretization of the governing non-linear partial differential equations. The numerical solution of viscous fluid flow requires very large numbers of boundary elements and internal cells in order to capture all physical effects. Unfortunately, large systems of equations require enormous amounts of computation time and computer storage. The fast development of computer technology enables us to solve more and more complex problems, however, a vast number of problems still remain unsolved due to computation time and storage limitations.

The wavelet transform is a recent mathematical tool, developed specially for saving computational time and computer storage. It has been widely used for image compression and signal processing and recently for providing faster solutions of boundary element algorithms [2,7,8,11,12]. In the present work, we will use a modification of the Haar wavelet transform with a BEM numerical scheme, based on a numerical algorithm developed by Škerget et al. [16] for solving equations of fluid flow and heat transport. After a brief overview of the BEM algorithm in Section 2, we will introduce the modification of the discrete wavelet transform, which will transform vectors of arbitrary size.

The wavelet solution method for solving systems of linear equations was previously proposed, among others, by Bucher et al. [2]. The *thresholding* operation makes a fully populated matrix sparse. Beylkin et al. [1] proposed the fast wavelet transform (FWT) algorithm, which ensures O($n$log$n$) non-zero elements at a fixed threshold, using Daubechies [3] wavelets. Koro and Abe [12–14] proposed a threshold determination strategy that optimizes matrix sparsity and calculation error using their non-orthogonal spline wavelets. In Section 4, we introduce the threshold as a product of a user defined factor and the average matrix element value. This results in linear dependence of the user defined factor and the final solution field RMS error.

Finally, the flow kinematics equation is solved with wavelets repeatedly for two numerical test cases to provide an insight into the possibilities and limitations of the use of

* Corresponding author.

E-mail addresses: jure.ravnik@uni-mb.si (J. Ravnik), leo@uni-mb.si (L. Škerget), matjaz.hribersek@uni-mb.si (M. Hriberšek).

0955-7997/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.enganabound.2004.05.002

the technique in the proposed velocity–vorticity BEM numerical scheme. Special emphasis is given to the investigation of the effects of changing values of Reynolds and Rayleigh number, as wells as changing computational mesh density, on the error caused by wavelet solution of the flow kinematics equation.

2. The Navier–Stokes equations

The analytical description of motion of a continuous fluid medium is based on conservation of mass, momentum and energy, the associated equations of state and constitutive relations. With the assumptions of incompressibility within Boussinesq approximation, a velocity–vorticity formulation was developed by Škerget et al. [16–18] and Hribersek and Škerget [9,10]. The dynamics of a viscous incompressible fluid is partitioned into its kinematic and kinetic aspect through the use of derived vector vorticity field function $\omega_i(r_j,t)$, obtained as a curl of the compatibility velocity field $v_i(r_j,t)$:

$$\omega_i = e_{ijk}\frac{\partial v_k}{\partial x_j}, \qquad \frac{\partial \omega_i}{\partial x_j} = 0, \tag{1}$$

which is a solenoidal vector by definition, and e_{ijk} is the permutation unit tensor. By applying a curl to the vorticity definition (1) and using the continuity equation for incompressible flow $\vec{\nabla}\cdot\vec{v} = 0$, the following vector elliptic Poisson equation for velocity vector is obtained:

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

Eq. (2) represents the kinematics of an incompressible fluid motion, expressing the compatibility and restriction conditions between velocity and vorticity field functions.

The kinetic aspect is governed by the parabolic diffusion–convection vorticity transport equation, obtained by applying the curl operator to the momentum equation. In the case of two-dimensional flow, the vorticity vector has just one component, which is perpendicular to the plane of the flow. Thus, we obtain a scalar transport equation for vorticity

$$\frac{\mathrm{D}\omega}{\mathrm{D}t} = \nu\frac{\partial^2\omega}{\partial x_j\partial x_j} + e_{ij}g_j\frac{\partial F_B}{\partial x_i}, \tag{3}$$

e_{ij} being the permutation unit symbol and $\frac{\mathrm{D}}{\mathrm{D}t}$ the Stokes derivative. The vorticity transport equation is non-linear due to the product of velocity and vorticity in the convective term. The buoyancy source term $F_B = -\beta_T(T - T_0)$ couples the vorticity transport equation with the energy conservation equation:

$$\frac{\mathrm{D}T}{\mathrm{D}t} = a\frac{\partial^2 T}{\partial x_j\partial x_j}, \tag{4}$$

where β_T is the thermal volume expansion coefficient, T_0 is the reference temperature, $a = \frac{\lambda}{\rho c_p}$ denotes the thermal diffusivity. The material properties such as mass density ρ, specific isobaric heat c_p, kinematic viscosity ν and heat conductivity λ are assumed to be constant parameters.

Eqs. (2)–(4) form a closed system of partial differential equations, which must be solved in order to obtain the resulting velocity, vorticity and temperature fields. To apply the boundary element method, we must rewrite the equations in integral form. Making use of the Green fundamental solutions, the following integral representation for the two-dimensional plane kinematics can be derived [16]:

$$c(\xi)v_i(\xi) + \int_\Gamma v_i\frac{\partial u^*}{\partial n}\,\mathrm{d}\Gamma = \int_\Gamma v_j\frac{\partial u^*}{\partial t}\,\mathrm{d}\Gamma - e_{ij}\int_\Omega \omega\frac{\partial u^*}{\partial x_j}\,\mathrm{d}\Omega, \tag{5}$$

where u^* is the elliptic Laplace fundamental solution, ξ the source point and Γ the boundary of the domain Ω.

Deriving integral representations of the vorticity transport equation (3) and heat transport equation (4), one has to consider the parabolic diffusion–convection character of the transport equation. The final integral statements are [16]:

$$\begin{aligned} c(\xi)\omega(\xi,t_F) &+ \nu\int_\Gamma\int_{t_{F-1}}^{t_F}\omega\frac{\partial u^*}{\partial n}\,\mathrm{d}t\,\mathrm{d}\Gamma \\ &= \nu\int_\Gamma\int_{t_{F-1}}^{t_F}\frac{\partial\omega}{\partial n}u^*\,\mathrm{d}t\,\mathrm{d}\Gamma - \int_\Gamma\int_{t_{F-1}}^{t_F}\omega v_n u^*\,\mathrm{d}t\,\mathrm{d}\Gamma \\ &+ \int_\Omega\int_{t_{F-1}}^{t_F}\omega v_j\frac{\partial u^*}{\partial x_j}\mathrm{d}t\,\mathrm{d}\Omega + e_{ij}\int_\Gamma\int_{t_{F-1}}^{t_F} n_i g_j F_B u^*\,\mathrm{d}t\,\mathrm{d}\Gamma \\ &- e_{ij}\int_\Omega\int_{t_{F-1}}^{t_F} g_j F_B\frac{\partial u^*}{\partial x_i}\mathrm{d}t\,\mathrm{d}\Omega + \int_\Omega \omega_{F-1}u^*_{F-1}\,\mathrm{d}\Omega \end{aligned} \tag{6}$$

for the vorticity transport equation (3) and

$$\begin{aligned} c(\xi)T(\xi,t_F) &+ a\int_\Gamma\int_{t_{F-1}}^{t_F} T\frac{\partial u^*}{\partial n}\,\mathrm{d}t\,\mathrm{d}\Gamma \\ &= a\int_\Gamma\int_{t_{F-1}}^{t_F}\frac{\partial T}{\partial n}u^*\,\mathrm{d}t\,\mathrm{d}\Gamma - \int_\Gamma\int_{t_{F-1}}^{t_F} T v_n u^*\,\mathrm{d}t\,\mathrm{d}\Gamma \\ &+ \int_\Omega\int_{t_{F-1}}^{t_F} T v_j\frac{\partial u^*}{\partial x_j}\mathrm{d}t\,\mathrm{d}\Omega + \int_\Omega T_{F-1}u^*_{F-1}\,\mathrm{d}\Omega, \end{aligned} \tag{7}$$

for the heat transport equation (4). In Eqs. (6) and (7), the u^* is the parabolic diffusion fundamental solution and $\Delta t = t_F - t_{F-1}$ the corresponding time increment.

Analytical solutions of integral equations for velocity, vorticity and temperature exist only for simple cases of elementary geometries or basic forms of boundary conditions. The boundary element method enables us to find approximate solutions of the integral equations by writing them in a discrete form. To obtain discrete forms of equations, we divide the boundary Γ into boundary elements with N_e nodes and the domain Ω into internal cells with N_c

nodes. The unknowns of the system are nodal values of ω, v_x, v_y, $\frac{\partial\omega}{\partial n}$, T and $\frac{\partial T}{\partial n}$.

The flow kinematics equation (5) is written in discrete form by collocating the boundary nodes with the source point ξ. This produces an implicit system matrix for unknown boundary nodes only. Eq. (5) has to be written in its tangential or normal form to produce a non-singular system matrix [18]. The final discrete forms of the flow kinematics equation are normal (8) and tangential (9) forms:

$$([H_1]+[H_{t1}])\{v_n\} = +([H_{t2}]-[H_2])\{v_t\}+[D_t]\{\omega\}, \quad (8)$$

$$([H_1]+[H_{t1}])\{v_t\} = -([H_{t2}]-[H_2])\{v_n\}+[D_n]\{\omega\}, \quad (9)$$

where $\{v_n\}$, $\{v_t\}$ and $\{\omega\}$ are column vectors of node point transformed velocities and vorticities. The solution of systems of Eqs. (8) and (9) gives boundary values of velocity and vorticity. Velocities in the domain are computed explicitly from the discrete form of the integral kinematics equation (5).

For equations of vorticity transport (3) and heat transport (4) all nodes (boundary and domain) are included in the system equation. Boundary and domain vorticities and temperatures are computed implicity through the solution of the system of linear equations. The final discrete forms of Eqs. (3) and (4) are:

$$\begin{aligned}[H_k]\{\omega\}_F = [G_k]\left\{\frac{\partial\omega}{\partial n}\right\}_F - \frac{1}{\nu}[G_k]\{v_n\omega\}_F \\ + \frac{1}{\nu}([D_{kx}]\{v_x\omega\}_F + [D_{ky}]\{v_y\omega\}_F) \\ + \frac{1}{\nu}[B_k]\{\omega\}_{F-1}\end{aligned} \quad (10)$$

for vorticity transport and

$$\begin{aligned}[H_e]\{T\}_F = [G_e]\left\{\frac{\partial T}{\partial n}\right\}_F - \frac{1}{\alpha}[G_e]\{v_nT\}_F \\ + \frac{1}{\alpha}([D_{ex}]\{v_xT\}_F + [D_{ey}]\{v_yT\}_F) \\ + \frac{1}{\alpha}[B_e]\{T\}_{F-1}\end{aligned} \quad (11)$$

for heat transport. For specified boundary and initial conditions, this non-linear set of Eqs. (8)–(11) is solved by an iterative procedure with under-relaxation. Through the iterative process, large systems of linear equations must be repeatedly solved. The described discretization procedure was presented among others by Hriberšek and Škerget [9], Škerget et al. [17], and Wrobel [19].

3. The discrete wavelet transform

The aim of this section is to show how to use the properties of the discrete wavelet transform to accelerate the solution of large linear systems of equations. One can find another account of using this technique, among others in Bucher et al. [2] and Koro and Abe [11–13]. The matrices produced by discrete integral equations (5)–(7) are fully populated and non-symmetric. We will use the modified Haar wavelet transform, introduced below, to compress the matrices i.e. to make them sparse. Writing the matrices in compressed form (compressed row storage was used in present work) will enable the solver to find the solutions in less computational time.

3.1. The fast wavelet transform

The FWT algorithm of Beylkin et al. [1] uses a pyramidal scheme to transform a vector into a wavelet basis. It employs compact support wavelets of Daubechies [3] with M vanishing moments. Each wavelet family is characterized by $2M$ wavelet filter coefficients (h_i, g_i), which were tabulated by Daubechies up to $M = 10$ in her 1988 paper [3]. With $M = 1$ we have the Haar wavelets, which have a constant scaling function and non-overlapping support.

Given a vector with components s_k^0; $k = 1,\ldots,N = 2^{n+1}$, the following formulae are used recursively to make the transform:

$$s_k{}^j = \sum_{l=1}^{l=2M} h_l s_{l+2k-2}^{j-1} \quad (12)$$

$$d_k{}^j = \sum_{l=1}^{l=2M} g_l s_{l+2k-2}^{j-1} \quad (13)$$

where $s_k{}^j$ and $d_k{}^j$ are viewed as periodic sequences with the period 2^{n-j}. The formulae (12) and (13) map coefficients s_k^{j-1} with $k = 1,\ldots,2^{n-j+1}$ into $s_k{}^j$ and $d_k{}^j$ with $k = 1,\ldots,2^{n-j}$. The inverse mapping is given by

$$\begin{aligned}s_k^{j-1} &= \sum_{k=1}^{k=M} h_{2k} s_{n-k+1}^j + \sum_{k=1}^{k=M} g_{2k} d_{n-k+1}^j, \\ d_k^{j-1} &= \sum_{k=1}^{k=M} h_{2k-1} s_{n-k+1}^j + \sum_{k=1}^{k=M} g_{2k-1} d_{n-k+1}^j.\end{aligned} \quad (14)$$

Using formulae (12) and (13) recursively for all $j = 1,\ldots,n$, $k = 1,\ldots,2^{n-j}$ starting with s_k^0, $k = 1,\ldots,N$ we are able to evaluate all $s_k{}^j$ and $d_k{}^j$ with a cost proportional to N. Storing $d_k{}^j$ and the one element in s_1^n, we can perform the inverse transform by recursively using Eq. (14) for $j = n, n-1,\ldots,0$.

3.2. FWT for vectors of arbitrary length

The requirement that the number of vector components N is a power of 2, $N = 2^{n+1}$, steams form the fact that complete wavelet orders must be used in the transform. All wavelets are formed by translation and dilation of a prototype wavelet or a mother wavelet. The mother wavelet is the zero-order wavelet denoted by $\psi_{0,1}$. Higher order wavelets are denoted by $\psi_{k,l}$, where k is the wavelet order and l its position. Wavelets of order k are dilated into $l = 1,\ldots,2^k$ positions. Starting with the zero-order mother

wavelet, which only has one position and including the scaling function, we have 2^{n+1} wavelets, if the maximal wavelet order is n.

The boundary-domain integral method solves viscous fluid flow problems by solving large systems of linear equations for equations of flow kinematics, vorticity transport and heat transport. The sizes of the system matrices are the number of boundary nodes and the number of boundary and internal nodes for equations of kinematics and both transport equations, respectively. For the wavelet transform to be complete, all wavelets up to a maximal order must be included into the wavelet matrix. Therefore, the wavelet transform can work only on vectors (or columns, rows of a system matrix), which have a power of two components. This poses a limitation for our formulation. It is not possible to have the number of boundary nodes and the number of boundary and internal nodes both of 2^n form. Also, the difference in computational effort needed to solve linear systems with 2^n and 2^{n+1} equations becomes drastic at large values of n, hence making the choice of computational mesh density very limited.

Let a vector $\vec{s}^0$ have an arbitrary number of components $s_k^0, k = 1, \ldots, L$ and let the first power of 2 larger than L be N. We will expand the vector with additional $K = N - L$ components x_i, chosen is such manner, that the last K wavelet coefficients d_k^1 will end up zero. The wavelet transform of the expanded vector will be:

$$(s_1^0, s_2^0, \ldots, s_{L-K+1}^0, x_1, s_{L-K+2}^0, x_2, \ldots, s_L^0, x_K) \rightarrow (s_0^n, d_1^n, \ldots, d_1^1, d_2^1, \ldots, d_{N/2-K}^1, 0, \ldots, 0), \tag{15}$$

where there are K zero and L non-zero wavelet coefficients in the transformed vector. Also, $N/2$ is always larger than K, because N is the first power of 2 number larger than L. In the case of $K = N/2$ we are dealing with a trivial case of the original vector having a power of 2 components $L = N/2$, so there is no need for expansion. Eq. (13) defines the values d_k^1. Let us rewrite it for $j = 1$:

$$d_k^1 = \sum_{l=1}^{l=2M} g_l s_{l+2k-2}^0, \qquad k = 1, 2, \ldots, N/2 \tag{16}$$

where s_i^0 are the expanded vector defined on the left hand side of Eq. (15). We would like to have the last K coefficients equal to zero, $d_k^1 = 0$. To achieve this, the following system of equations must be solved for x_i :

$$d_k^1 = \sum_{l=1}^{l=2M} g_l s_{l+2k-2}^0 = 0, \qquad k = N/2 - K, \ldots, N/2 \tag{17}$$

where $s^0 = (s_1^0, s_2^0, \ldots, s_{L-K+1}^0, x_1, s_{L-K+2}^0, x_2, \ldots, s_L^0, x_K)$. The system (17) has K equations for x_k $(k = 1, \ldots, K)$ unknowns. It is a M upper diagonal system, where non-zero elements of the system matrix are located only on M upper diagonals. For $M = 3$ and $K = 6$ the system is such:

$$\begin{matrix} g_2 & g_4 & g_6 & 0 & 0 & 0 & x_1 \\ 0 & g_2 & g_4 & g_6 & 0 & 0 & x_2 \\ 0 & 0 & g_2 & g_4 & g_6 & 0 & x_3 \\ 0 & 0 & 0 & g_2 & g_4 & g_6 & x_4 \\ 0 & 0 & 0 & 0 & g_2 & g_4 & x_5 \\ 0 & 0 & 0 & 0 & 0 & g_2 & x_6 \end{matrix} = \vec{R}, \tag{18}$$

where the right hand side vector $\vec{R}$ is equal to

$$R_k = \begin{cases} -\sum_{i=1}^{M} g_{2i-1} s_{N-2K-1+k+i}^0 & k \le K - (M-1) \\ \left\{ -\sum_{i=1}^{\alpha} g_{2i-1} s_{N-2K-1+k+i}^0 - \sum_{i=2\alpha+1}^{2M} g_i s_{i-2\alpha}^0 \right\} & k > K - (M-1) \end{cases}, \tag{19}$$

and $\alpha = M - [k - \{K - (M-1)\}]$. Solution of the system (17) may be obtained recursively, when starting from the bottom and solving for x_K first. The following formulae should be used:

$$x_k = \begin{cases} \frac{1}{g_2}\left\{ R_k - \sum_{i=2}^{M} g_{2i} x_{k+i-1} \right\} & k \le K - (M-1) \\ \frac{1}{g_2}\left\{ R_k - \sum_{i=1}^{K-k} g_{2i+2} x_{k+i} \right\} & k > K - (M-1) \end{cases}. \tag{20}$$

for $k = K, K-1, \ldots, 1$. The solution (20) of the system (17) is specially simple for the Haar wavelets ($M = 1$, $g_1 = -g_2$), namely

$$x_k = s_{L-K+k}^0, \qquad k = 1, \ldots, K, M = 1. \tag{21}$$

Let us estimate the size of coefficients x_k for Daubechies wavelets ($M > 1$). Since the application of the formulate (20) is recursive, we approximately have $x_{k-1} \approx \frac{x_k}{g_2}$. This makes the order $O(x_1) \approx (\frac{1}{g_2})^K$. The absolute value of the wavelet coefficient is $g_2 = 0.22$ for $M = 2$ and quickly decreases towards zero as M increases [3]. This makes x_1 a huge number for virtually any combination K and $M > 1$, which soon cannot be represented by a double precision number. Daubechies wavelets ($M > 1$) can be used only, if K is sufficiently small. In practice this means, that one can expand the vector by only a few additional coefficients. Due to this limitation of Daubechies wavelets, we have used the Haar wavelets for the solution of the linear systems of equations of fluid flow.

3.3. The Haar wavelet transform for vectors of arbitrary length

Here, we introduce the modified Haar wavelet transform, which can transform vectors of any size, using a full matrix form instead of the pyramid scheme presented above. Let the Haar wavelet matrix be denoted by H. It is a square matrix with $2^{k_{\max}+1} \times 2^{k_{\max}+1}$ elements, including all wavelets up to the $k_{\max}$ order. We would like to be able to transform a vector $\vec{f}$ of N components by multiplying it with the wavelet matrix. The number of components N is larger than $2^{k_{\max}}$, but smaller than $2^{k_{\max}+1}$. We can always choose such $k_{\max}$ that this inequality holds.

The strategy of our modification of the Haar wavelet transform is to expand the vector, which has N components, to $2^{k_{\max}+1}$ components in such a way, that the transformed vector $\hat{f} = H\vec{f}$ will have only N non-zero components. The expansion is done by duplicating the last components of the vector (see Eq. (21)). The highest order wavelets in the bottom half of the Haar wavelet matrix have only two non-zero elements of the same magnitude and opposite sign. When those rows are multiplied with a vector with duplicated components, the result is zero. The expansion is a linear process and can be written in matrix form $E\vec{f} = \tilde{f}$. The expansion matrix E has $2^{k_{\max}+1}$ rows and N columns. An example of the expansion for $N = 5$ and $k_{\max} = 2$ is given in Eq. (22):

$$E\vec{f} = \begin{matrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{matrix} \cdot \begin{matrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{matrix} = \begin{matrix} f_1 \\ f_2 \\ f_3 \\ f_3 \\ f_4 \\ f_4 \\ f_5 \\ f_5 \end{matrix} = \vec{f}_e. \tag{22}$$

Applying the Haar wavelet matrix H on the expanded vector $\vec{f}_e$ yields a vector with only N non-zero elements:

$$H\vec{f} = H\vec{f}_e = |\hat{f}_1, \hat{f}_2, \hat{f}_3, \hat{f}_4, \hat{f}_5, 0, 0, 0|^{\mathrm{T}} = \hat{f}_e. \tag{23}$$

Since all $(\hat{f}_e)_i$ components for $i > N$ are zero, one can introduce contraction linear operator C, which will transform $\hat{f}_e$ into $\hat{f}$ by disregarding the zeros. Writing C in matrix form gives

$$CHE\vec{f} = C\hat{f}_e = \begin{matrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{matrix} \cdot \begin{matrix} \hat{f}_1 \\ \hat{f}_2 \\ \hat{f}_3 \\ \hat{f}_4 \\ \hat{f}_5 \\ 0 \\ 0 \\ 0 \end{matrix} = \begin{matrix} \hat{f}_1 \\ \hat{f}_2 \\ \hat{f}_3 \\ \hat{f}_4 \\ \hat{f}_5 \end{matrix} = \hat{f}. \tag{24}$$

The matrix product CHE is a square matrix of $N \times N$ elements. The modified Haar or the CHE transform of a vector of arbitrary length can be written as $\hat{f} = CHE\vec{f}$. The inverse CHE transform is obtained by multiplying the wavelet coefficients with the inverse $(CHE)^{-1}$ matrix:

$$\vec{f} = (CHE)^{-1}\hat{f} = E^{-1}H^{\mathrm{T}}C^{\mathrm{T}}\hat{f}, \tag{25}$$

$$(CHE)^{-1}(CHE) = E^{-1}H^{\mathrm{T}}C^{\mathrm{T}}CHE = \underline{I}, \tag{26}$$

where E^{-1} is a matrix that shortens the vector by disregarding duplicate elements, H^{T} is a transpose Haar wavelet matrix and C^{T} is a transpose C matrix, which expands a vector by adding zeros to it. Matrices C, C^{T}, E and E^{-1} are identity matrices, if the vector length N is equal to the wavelet matrix dimension $N = 2^{k_{\max}+1}$.

The CHE transform is still in its essence the Haar wavelet transform. Before the Haar transformation, the vector is modified in such manner, that just the right number of wavelet coefficient end up zero. Not storing zeros makes it possible to apply the CHE transform to a vector with an arbitrary number of components and store only the same number of wavelet coefficients.

4. Solving linear systems of equations using the modified Haar wavelet transform

The discretization procedure described in the beginning of this paper yields large systems of linear equations for equations of flow kinematics, vorticity transport and heat transport. Let us write the systems symbolically

$$\underline{a}\vec{x} = \vec{b}, \tag{27}$$

where the unknown values are $\vec{x} = x_1, x_2, \ldots, x_n$. The right hand side vector $\vec{b} = b_1, b_2, \ldots, b_n$ and the system matrix $\underline{a}$ with elements a_{ij} are both known. Multiplying with the CHE wavelet matrix on both sides of Eq. (27) gives

$$CHE\underline{a}\vec{x} = CHE\vec{b}. \tag{28}$$

The expression $(CHE)^{-1}(CHE)$ is equal to identity (Eq. (26)), so it can be inserted into Eq. (28) to obtain

$$CHE\underline{a}(CHE)^{-1}CHE\vec{x} = CHE\vec{b}. \tag{29}$$

The new system matrix can now be defined as

$$\underline{\hat{a}} = CHE\underline{a}(CHE)^{-1} = CHE\underline{a}E^{-1}H^{\mathrm{T}}C^{\mathrm{T}}. \tag{30}$$

The product $CHE\underline{a}$ is the wavelet transform of all columns in $\underline{a}$ while $(CHE\underline{a})(CHE)^{-1}$ transforms all rows in the product $CHE\underline{a}$. Thus, the majority of information is written in large elements of $\underline{\hat{a}}$, while the redundant information of $\underline{a}$ is represented in small elements in $\underline{\hat{a}}$.

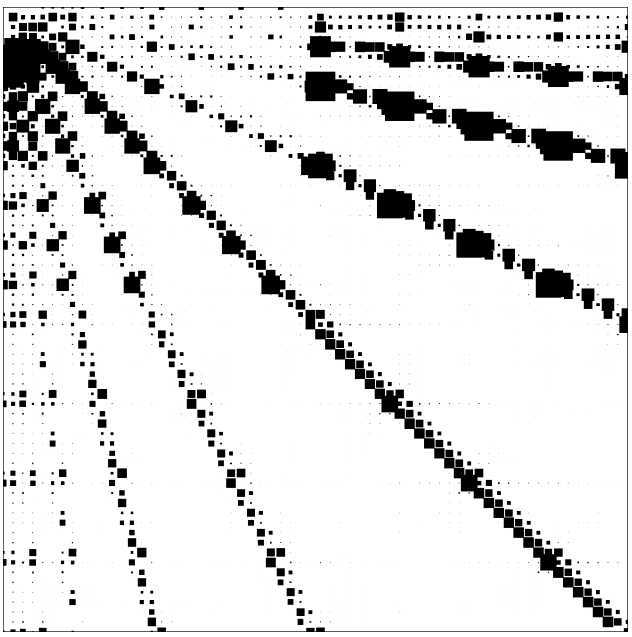

Fig. 1. The magnitude of absolute element values of the system matrix $\hat{\underline{a}}$. The largest elements lie on the diagonal and were, for the purpose of clearer presentation, decreased by a factor of 10. The matrix was taken from a flow kinematics calculation described in Section 5.

In Fig. 1, we displayed the magnitude of absolute element values of the system matrix $\hat{\underline{a}}$. The shown system matrix solves the flow kinematics equation with the velocity prescribed as the boundary condition. The numerical example is described in more detail below. The difference between absolute values of small and large elements in the matrix is several orders of magnitude.

The vectors $\vec{x}$ and $\vec{b}$ in Eq. (29) are transformed according to Eq. (24). Finally, we have a new system of linear equations

$$\hat{\underline{a}}\hat{x} = \hat{b} \tag{31}$$

governed by the new system matrix $\hat{\underline{a}}$. When the system is solved for $\hat{x}$, we obtain the final solution vector with the inverse wavelet transform (Eq. (25)) $\vec{x} = (CHE)^{-1}\hat{x}$.

The main advantage of this procedure is that the unknown vector in the new linear system of equations $\hat{x}$ consists of wavelet coefficients of the unknown vector $\vec{x}$ of the original system of equations. Let us examine the inverse transformation, which calculates $\vec{x}$ out of coefficients $\hat{x}$. Only a few coefficients $\hat{x}_i$ are needed to obtain a good approximation for $\vec{x}$. The vast number of other coefficients only make an already good approximation even better and if we use all of them, we obtain the perfect reconstruction of $\vec{x}$. Even if our new system of equations calculates $\hat{x}$ imperfectly, the inverse wavelet transformation will still give us a good approximation of $\vec{x}$. The strategy for the imperfect calculation of $\hat{x}$ was already proposed among others by Bucher et al. [2]. It suggests that elements of the system matrix $\hat{\underline{a}}$, that have small absolute values, could be *thresholded*, because they carry redundant information. The thresholding operation proposed by Bucher et al. [2] is

$$\text{threshold}(\hat{\underline{a}}) = \begin{cases} \hat{\underline{a}}_{ij}; & |\hat{\underline{a}}_{ij}| \geq \alpha \\ 0; & \text{otherwise} \end{cases} \tag{32}$$

Let $\bar{m}$ be the average value of absolute elements of $N \times N$ system matrix defined as

$$\bar{m} = \frac{1}{N^2} \sum_i \sum_j |\hat{\underline{a}}_{ij}|. \tag{33}$$

We chose factors κ of $\bar{m}$ to be the thresholds $\alpha = \kappa\bar{m}$ for the thresholding of elements in Eq. (32). Since the user has but little knowledge of the magnitude of elements in the system matrix it is convenient to express thresholding limit in terms of the factor κ. Although one could set the thresholding limit in terms of the number of thresholded elements, choosing the value of κ has three advantages.

- The thresholding operation algorithm is easier to develop.
- The RMS difference between solutions of the linear systems of equations with and without thresholding is linearly proportional to the factor κ and exponentially proportional to the share of thresholded elements.
- The dependence of the RMS difference to κ is independent of the size of the system of equations, whereas the RMS difference at a fixed share of thresholded elements depends on the system matrix size.

To show this advantages we devised the following numerical test. Three flow kinematics system matrices with 120, 240 and 360 equations were prepared. All were taken from the driven cavity flow test case, which is described in detail in Section 5. We solved the three systems for 100 random right hand side vectors for a wide range of κ values, calculating the RMS difference between solution vector obtained at $\kappa = 0$ and vectors obtained at non-zero κ. The average of the 100 RMS values is shown in Fig. 2. Please note that the factor κ in Fig. 2a is plotted in log–log scale and therefore the function RMS(κ) is linear and independent of the number of equations in the system. At values of κ larger than $\kappa \approx 10$, only the diagonal of the system matrix is retained and the RMS difference becomes flat and independent of κ. On the other hand, the dependence of RMS on the share of thresholded elements, plotted in graph in Fig. 2b, is exponential and depends on the number of equations.

We shall now review this section by writing a solution algorithm for a system of linear equations using the wavelet matrix compression technique. The boundary element method (or any other numerical scheme) produces the system matrix $\underline{a}$ and the right hand side vector $\vec{b}$. The user chooses the factor κ as one of the input parameters of the numerical scheme.

1. We calculate wavelet transform of the right hand side vector $\hat{b} = CHE\vec{b}$.

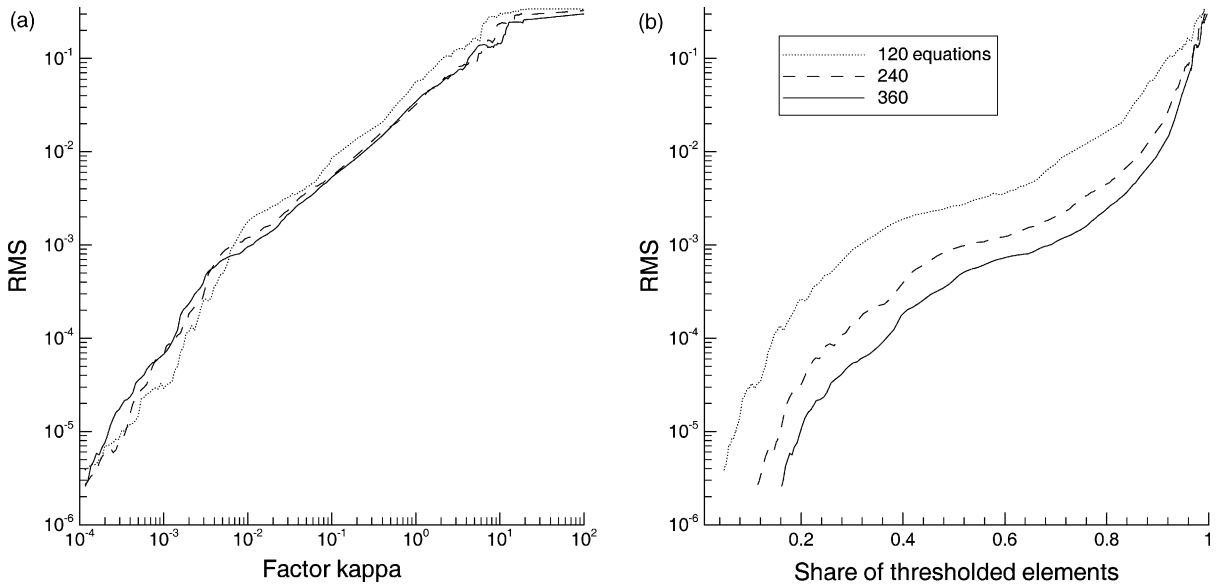

Fig. 2. The RMS difference between solution vectors obtained with and without thresholding in dependence of (a) the factor κ and (b) the share of thresholded elements.

2. Next we have to calculate the new system matrix via the following matrix product: $\hat{\underline{a}} = CHE\underline{a}E^{-1}W^{\mathrm{T}}C^{\mathrm{T}}$. Although the fast pyramid scheme of Beylkin et al. [1] can be applied, this product still takes a substantial amount of CPU time.
3. We zero out elements in $\hat{\underline{a}}$, that have absolute values less than the chosen threshold $\alpha = \kappa\bar{m}$ (Eq. (32)).
4. The thresholded matrix is now written in a compressed form. We used the compressed row storage technique [10].
5. We solve $\hat{\underline{a}}\hat{x} = \hat{b}$ for $\hat{x}$ instead of $\underline{a}\vec{x} = \vec{x}$ for $\vec{x}$.
6. Finally, we calculate the inverse wavelet transform $\vec{x} = (CHE)^{-1}\hat{x}$ and obtain the solution vector.

Only step 5 is needed, when solving the system without usage of the wavelet transform. It is therefore imperative that steps 1–4 and 6 take less computation time than it is saved by the faster solution of the system in step 5.

The calculation of the new system matrix in step 2 (Eq. (30)) takes up the majority of CPU time. Each row and column must be extended (E), then transformed (H) and finally compressed (C). We measured CPU time needed for step 2 for systems with different number of equations. Results are shown in Fig. 3b, where it can be seen that the shape of the time versus number of equations curve is parabolic. Hence, the CPU time needed for the calculation

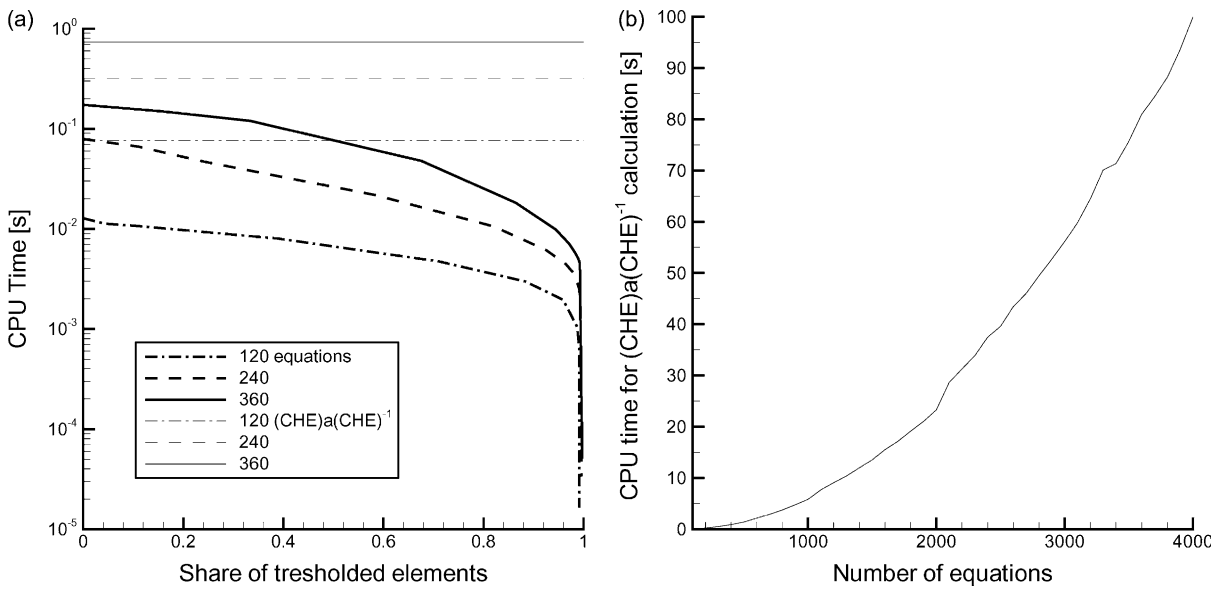

Fig. 3. The CPU time needed for (a) the solution of three wavelet transformed systems of equations and three $(CHE)\underline{a}(CHE)^{-1}$ calculations versus the share of thresholded elements; (b) $(CHE)\underline{a}(CHE)^{-1}$ calculations versus the number of equations N.

of the new system matrix is proportional to the square of the number of equations.

The wavelet solution algorithm saves CPU time by solving the system of linear equations faster (step 5 in the above algorithm). To be able to see how much CPU time can be saved, we repeated the numerical test described above. Three kinematics equation systems (120, 240 and 360 equations) were solved for random right hand side vectors. The *BICGSTAB* [15] solver with diagonal preconditioning was used. Both the CPU time needed for the solution versus share of thresholded elements and the time needed for the calculation of the new system matrix (which is independent of the share of thresholded elements) are shown in Fig. 3a. The sharp decrease in CPU time at very large shares of thresholded elements is due to the fact, that only the diagonal elements are non-zero and hence the solution of the system is trivial. Unfortunately, we can observe, that the CPU time needed for the calculation of the new system matrix is longer than the time required for the iterative solution of the original system. Thus, the CPU time cannot be saved, if the system needs to be solved for only one right hand side vector. It was already reported by González et al. [7] and Beylkin et al. [1] that the cost of iterative solvers for sparse systems is of the order O($N \log N$), while the wavelet compression of the system matrix has the cost of O(N^2). Therefore, CPU time can be saved only, if we are solving the system for more than one right hand side vector.

In order to implement Eq. (30) to calculate the wavelet transformed system matrix $\hat{\underline{a}}$ one needs the whole system matrix $\underline{a}$ in storage. The implementation itself requires only one additional vector to store the temporary wavelet coefficients. Since, Eq. (30) has to be solved prior to the iterative process only once, the original system matrix may be stored outside the core memory. After compression, the original matrix is disregarded and only the compressed matrix, which should fit into the core memory, is stored and used throughout the iterative process to obtain solutions for different right hand side vectors.

Our work focuses mainly on the accelerated solution of linear systems of equations. However, wavelet compression may be used on any linear equation involving matrices and vectors. Storage requirements for explicit calculations like $\vec{x} = \sum \underline{m}_i \cdot \vec{v}_i$ may be efficiently reduced with wavelet compression of matrices $\hat{\underline{m}}_i = CHE\underline{m}_i(CHE)^{-1}$ and solving the equation for wavelet transformed vector $\hat{x} = \sum \hat{\underline{m}}_i \cdot \hat{v}_i$. Such schemes are especially useful in cases when storage is more of a problem than CPU time.

5. Wavelet transform for flow kinematics

The use of the boundary element method, to solve the governing equations of fluid flow written in the velocity–vorticity formulation, requires a coupled iterative solution procedure. Equations of fluid kinematics (8) and (9), vorticity transport (10) and heat transport (11) are to be solved for each iterative step. This procedure is described in detail in Škerget et al. [16]. The main difference between the flow kinematics and kinetics equations is, that the system matrix $\underline{a}$ changes for each iterative step for equations of vorticity and heat transport, yet it is kept constant for the flow kinematics equation. The reason for this is that the velocity field, which changes every iteration, is needed to calculate convective transport in kinetics equations (10) and (11). This means that the most CPU intensive step in the wavelet solution algorithm (2) needs to be calculated only once for the flow kinematics equation, and in each iterative step for the transport equations. As it was shown in Section 4, wavelet matrix preparation takes longer than the solution of the system, so using wavelet transform for the solution of kinetics equations would prolong computation time. For this reason, we focused our attention on the wavelet solution of the flow kinematics. In special cases, e.g. creeping flow, the kinetics equations can also be kept constant by neglecting convective terms in Eqs. (10) and (11). In the sub-domain technique [10], the kinetics system matrix is temporary frozen in the iteration process and changes only every 10 or so iterations. Using the wavelet transform with fluid kinetics in the sub-domain technique will be investigated in forthcoming research.

The use of wavelet compression for the solution of the flow kinematics equation produces inaccurate boundary vorticity values each iterative step. Those are used for the solution of the transport equations. The *BICGSTAB* [15] solver with diagonal preconditioning was used for solving all systems of linear equations. When the solution vorticity field changes between iterations for less than the convergence criteria (RMS difference between vorticity fields in subsequent iterations is less than $\epsilon = 5 \times 10^{-6}$), the final solution is written and the calculation process is stopped. To get a quantitative representation of the error that is caused by the wavelet compression we used the root mean square (RMS) measure between the solution fields obtained with and without the use of wavelets. The RMS measure is the square root of the sum of squared differences between the fields divided by the sum of squared values of the field calculated without wavelets.

Two numerical test examples were investigated: the onset of natural convection in a closed cavity and isothermal fluid flow in a square driven cavity. Natural convection onsets in a closed square cavity because of the buoyancy force, which is approximated via the Boussinesq approximation in our formulation. The test configuration is shown in Fig. 4a. The left wall is heated constantly at temperature T_1, while the right wall is cooled and kept at T_0. The bottom and the top walls are adiabatic, while the gravity force acts in a downward direction. The fluid is at rest on all walls. This test was covered extensively by Davies [4] and Davies and Jones [5]. The benchmark Nusselt number $Nu = \int_0^1 \frac{\partial T}{\partial n} dy$ for the middle vertical plane was used for quantitative comparison of the results.

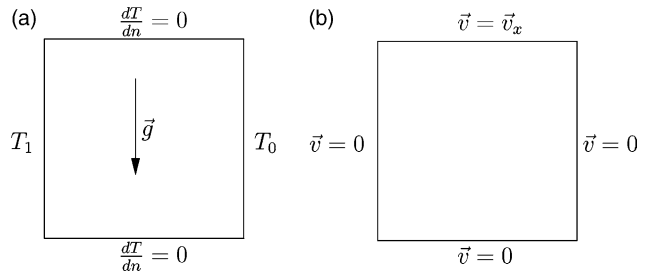

Fig. 4. Presentation of the two tests: (a) the onset of natural convection in a closed cavity and (b) isothermal flow in a square driven cavity.

The second test example investigated was the driven cavity test. The fluid in the cavity is isothermal, so it is not necessary to solve the heat transport equation. We are seeking an iterative solution of only fluid kinematic equation and vorticity transport equation. The driven cavity is presented in Fig. 4b. On the top wall, the velocity is prescribed in the x direction $\vec{v}=\vec{v}_x$, whereas on the other walls, the flow velocity is zero. Ghia et al. [6] benchmark velocity profiles were used for comparison.

Our aim is to use the two tests to establish the dependence of the RMS error of the final temperature and velocity fields on the share of thresholded elements for different Rayleigh (Ra) and Reynolds (Re) number values and for different mesh densities.

5.1. Share of thresholded elements

The first set of calculations was made to establish the dependence on the share of thresholded elements. We chose a 120 equation linear system for the solution of fluid kinematics. The onset of natural convection was calculated for $Ra=10^4$ and the fluid in the driven cavity was at $Re=400$. Altogether we made six calculations of each test, changing the share of thresholded elements by making use of the factor κ. The first calculation was made without using wavelet compression ($\kappa=0$). The RMS difference of the temperature and velocity fields calculated with and without the use of wavelets is presented, along with other parameters of the calculations, in Table 1. In the Table and in the graph in Fig. 5, we can see that the RMS difference increases approximately linearly with κ, which is in agreement with the dependence on κ of RMS errors caused by wavelet transforms of a single vector, discussed in Section 5. The resulting temperature field and the velocity profiles are shown on Fig. 6.

Natural convection in a cavity is strong along the walls of the cavity, where velocity and its gradients are large. In the center of the cavity, diffusion is the dominant heat transport process. Wavelet solution of fluid kinematics introduces a numerical error, which is large in the convection-governed part of the cavity and small in the center of the cavity, where diffusion dominates. The increasing numerical error with κ is also reflected in the increasing difference of the Nusselt number to the benchmark in the driven cavity test.

Table 1
Error analysis of wavelet compressed solutions of the discrete fluid kinematic equation using linear systems of 120 equations for the onset of natural convection test and the driven cavity test

120 equation linear system, $Ra=10^4$; $Re=400$						
κ	Share	Natural convection			Driven cavity	
		RMS(T)	Nu	Nu_b	RMS($\vec{v}_x$)	RMS($\vec{v}_y$)
0.0	0.000	0.0	2.24504	2.243	0.0	0.0
10^{-3}	0.109	3.62×10^{-6}	2.24502	2.243	1.02×10^{-6}	1.10×10^{-6}
10^{-2}	0.387	2.42×10^{-5}	2.24515	2.243	6.14×10^{-5}	8.92×10^{-5}
10^{-1}	0.710	0.90×10^{-4}	2.24512	2.243	1.18×10^{-3}	1.51×10^{-3}
10^{+0}	0.883	1.78×10^{-3}	2.24971	2.243	9.38×10^{-3}	1.37×10^{-2}
10^{+1}	0.987	0.76×10^{-2}	2.26865	2.243	7.58×10^{-2}	1.05×10^{-1}

The RMS columns show the RMS difference between the final temperature and velocity fields solved with and without the use of wavelets. The middle vertical plane benchmark Nusselt number $Nu=\int_0^1 \frac{\partial T}{\partial n}\mathrm{d}y$ is shown alongside the benchmark Nusselt number Nu_b [4].

Looking at the velocity profiles in graphs b and c in Fig. 6, we can also confirm, that the profiles calculated with wavelets deviate from benchmark profiles mostly in the areas of high velocity and gradients. For every driven cavity test examined we found that RMS($\vec{v}_y$) is always larger than RMS($\vec{v}_x$). This is due to the geometry of the driven cavity test.

5.2. Reynolds and Rayleigh number values

Let us now examine the effect of Ra and Re number values on the error introduced by wavelet compressed solutions of the fluid kinematics equation. Again, a 120 linear equation system was chosen. The factor κ was

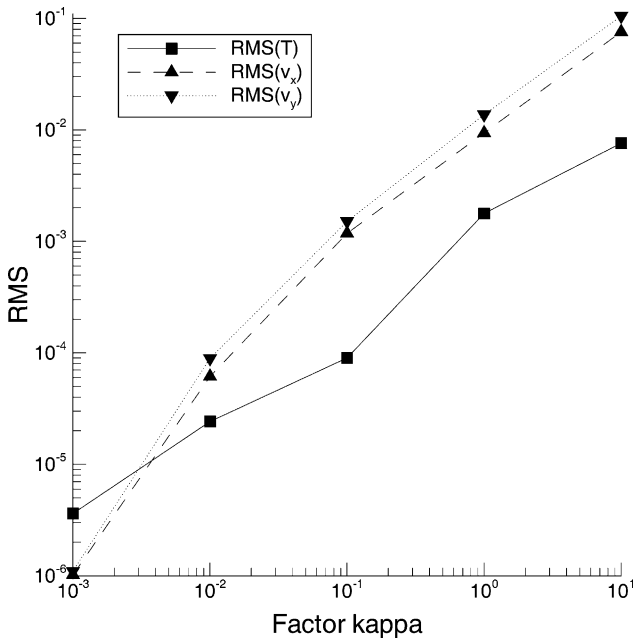

Fig. 5. The RMS error of the temperature field in the onset of natural convection test and the RMS error of the velocity field in the driven cavity test plotted versus the factor κ.

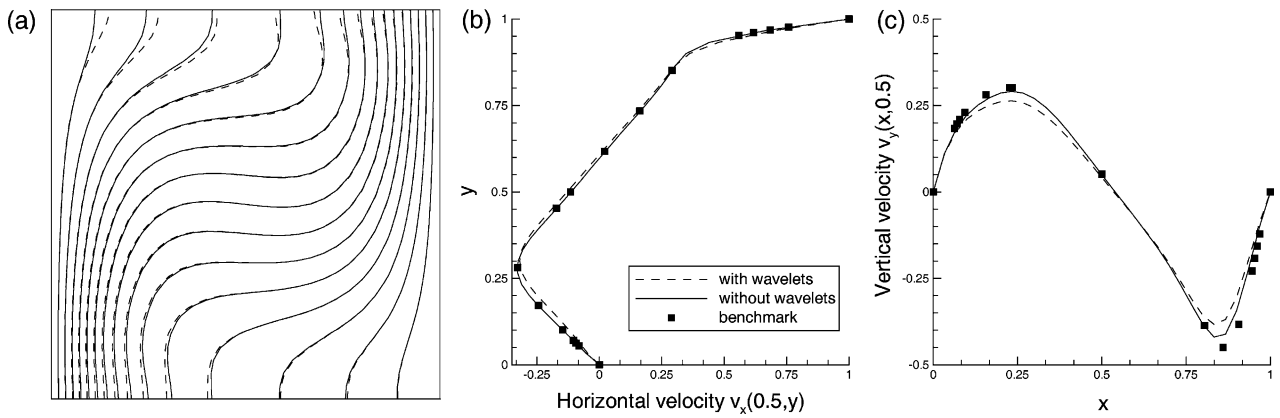

Fig. 6. The lines of constant temperature (a) in the onset of natural convection test and horizontal (b) and vertical (c) velocity profiles in the driven cavity test for $\kappa = 0$ (solid lines) and $\kappa = 10$ (dashed lines). Ghia et al. [6] benchmark velocity profiles are shown with squares.

Table 2
Error analysis with regard to changing *Ra* and *Re* number values

Discrete form of flow kinematics has 120 equations, $\kappa = 8.0$							
Natural convection					Driven cavity		
Ra	RMS(T)	*Nu*	Nu_0	Nu_b	*Re*	RMS($\vec{v}_x$)	RMS($\vec{v}_y$)
10^3	2.239×10^{-3}	1.124	1.118	1.118	100	3.123×10^{-2}	6.057×10^{-2}
10^4	8.666×10^{-3}	2.277	2.245	2.243	400	4.881×10^{-2}	6.902×10^{-2}
10^5	39.42×10^{-3}	4.656	4.539	4.519	1000	11.61×10^{-2}	14.48×10^{-2}

Nu_0 is the Nusselt number value of the solution calculated without wavelets. Other notation is the same as in Table 1.

kept constant for all calculations and was set to $\kappa = 8.0$. This value was chosen based on the linear RMS versus κ dependence established in Section 5.1. This choice leaves 256 non-zero system matrix elements out of 14,400, making the share of thresholded elements 0.9822. The onset of natural convection was evaluated for three Rayleigh number values $Ra = 10^3$, 10^4, 10^5, while the velocity field in a driven cavity was evaluated for three Reynolds number values $Ra = 100$, 400, 1000. The RMS differences between resulting temperature and velocity fields obtained with and without the use of wavelets are shown in Table 2.

The rate of convective heat transport in the cavity is characterized by the Rayleigh number value. The higher the value the stronger the convection. At higher *Ra* number value, velocities and velocity gradients are larger than at lower *Ra* number value. Thus, the inaccuracies in the solution of fluid kinematics equation at higher *Ra* number values, which are due to the use of wavelets, have stronger influence on the solution of transport equations and hence on the final temperature field. Looking at the RMS(T) column of Table 2 it is evident, that the RMS difference rises with increasing Rayleigh number value. In Fig. 7, the resulting temperature fields are plotted for all three Rayleigh number values.

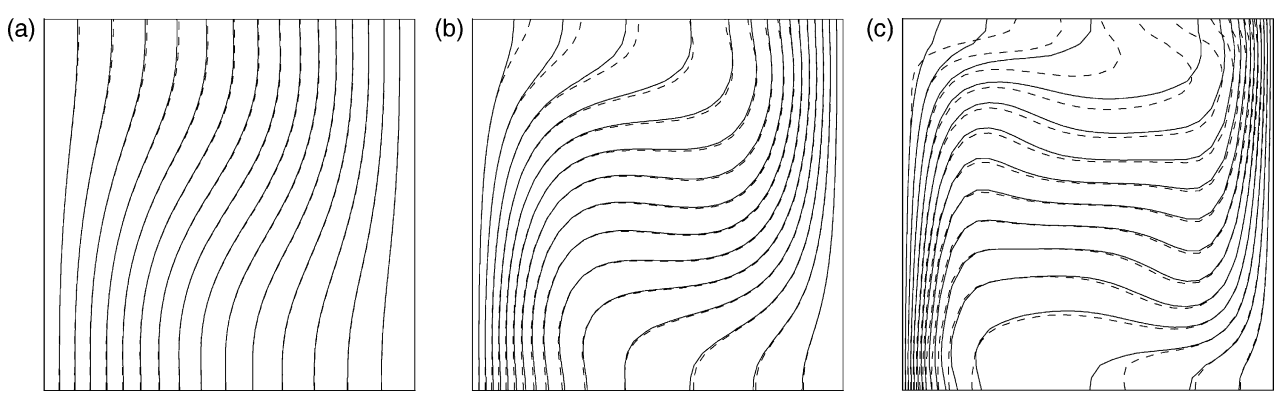

Fig. 7. The lines of constant temperature in a closed square cavity. The solid lines are temperature field solutions calculated without wavelet compression, dashed lines were calculated using wavelets at $\kappa = 8.0$ and (a) $Ra = 10^3$; (b) $Ra = 10^4$; (c) $Ra = 10^5$.

Table 3
Error analysis with regard to increasing computational mesh density

$\kappa = 8.0$, $Ra = 10^4$, $Re = 400$

Number of equation	Share	Natural convection				Driven cavity	
		RMS(T)	Nu	Nu_0	Nu_b	RMS($\vec{v}_x$)	RMS($\vec{v}_y$)
80	0.9826	11.82×10^{-3}	2.244	2.248	2.243	18.26×10^{-2}	24.31×10^{-2}
120	0.9822	8.666×10^{-3}	2.277	2.245	2.243	4.881×10^{-2}	6.902×10^{-2}
160	0.9792	4.301×10^{-3}	2.267	2.245	2.243	2.879×10^{-2}	3.293×10^{-2}

Notation is the same as in Table 2.

Decreasing solution accuracy with increasing Ra number value is obvious. Similarly in the driven cavity test, the higher Reynolds number value increases the RMS difference between velocity fields. Non-linearity of the momentum transport equation is more severe at higher Re number value and its solution more susceptible to the inaccuracies of the boundary conditions, which are set by the wavelet solution of the fluid kinematics equation.

5.3. Computational mesh density

To be able to understand the effect of computational mesh density on the employability of the wavelet compressed solutions of fluid kinematics equation, we repeated the two numerical tests with different mesh densities. We chose $\kappa = 8.0$, $Ra = 10^4$, $Re = 400$ and three mesh densities: the discrete form of the fluid kinematics equation had 80, 120 and 160 equations. The RMS differences between resulting temperature and velocity fields obtained with and without the use of wavelets are shown in Table 3.

All RMS error values decrease with increasing mesh density when the factor κ is kept constant. We have shown in Fig. 2, that the RMS error caused by preforming an inverse wavelet transform at constant κ on a vector of arbitrary size is constant. Hence, the boundary vorticities obtained by solving the flow kinematics equation with wavelets have the same RMS error regardless of the mesh density. When this constant error is introduced through boundary conditions to the solution of flow transport equations, it has a larger effect on the solution fields at lower mesh densities and a smaller effect at high mesh densities. This effect is visualized in Fig. 8, where the vorticity distributions of the natural convection test were plotted for all three cases. The increase of solution accuracy with increased mesh density can readily be observed.

This work was focused mainly on saving computational time by compressing an already formed full system matrix. Further work will investigate the possibility of completely abandoning the full system matrix and using the wavelet compressed matrix also for right hand side vector computations and for explicit computations of the fields inside of the domain.

6. Conclusions

In this paper, we presented an implementation of wavelet transform for acceleration of solutions of systems of linear equations. The velocity–vorticity [16–18] formulation of

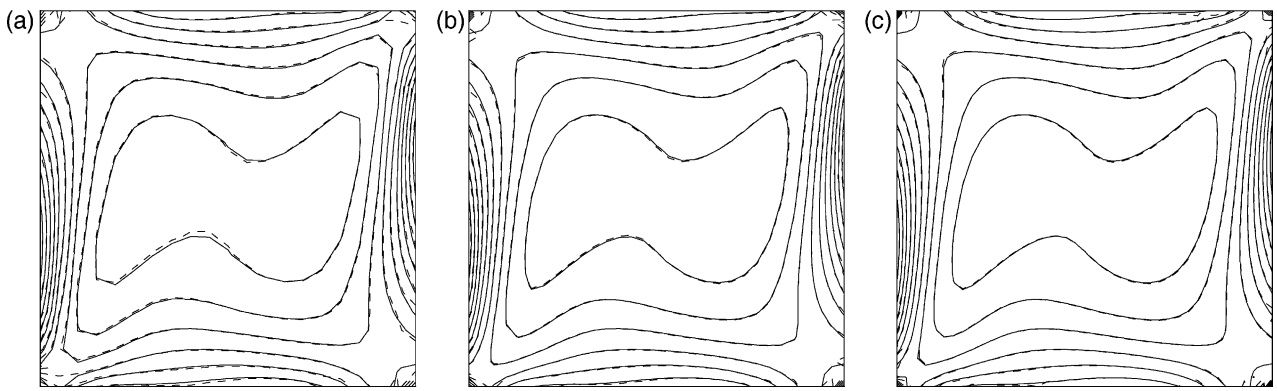

Fig. 8. Vorticity distribution in a closed square cavity for the onset of natural convection test. The solid lines are calculated without wavelet compression, dashed lines were calculated using wavelets at $\kappa = 8.0$. The flow kinematics equation was discreteized by (a) 80 equations, (b) 120 equations and (c) 160 equations.

the governing equations of fluid flow and heat transport was used. The boundary element method solving the governing equations was upgraded with the wavelet matrix compression technique. A modification of the discrete wavelet transform, which transforms vectors with arbitrary number of elements, was introduced.

In the wavelet solution algorithm, the system matrix is thresholded by zeroing out elements, which have absolute values less than a chosen threshold α. We proposed the threshold to be a factor κ of the mean absolute value of all matrix elements $\bar{m}$; $\alpha = \kappa\bar{m}$ It was shown, that the error caused by thresholding elements is linearly dependent on the factor κ, while the dependence on the share of thresholded system matrix elements is highly non-linear.

Fluid and heat flow in two numerical tests was investigated. The standard onset of natural convection in a square cavity and the flow in the driven cavity tests were solved with and without the use of wavelets for the flow kinematics equation for different *Ra*, *Re*, mesh densities and factors κ. It was shown that the RMS difference between the final solution field calculated with wavelets and the one obtained without wavelets increases linearly with the factor κ. The RMS difference also increases with increasing Rayleigh and Reynolds number values at the same share of thresholded elements. On the other hand, increasing computation mesh density enables us to threshold more elements to end up with the same RMS difference.

References

[1] Beylkin G, Coifman R, Rokhlin V. Fast wavelet transforms and numerical algorithms. Communications on Pure and Applied Mathematics 1991;44:141–83.

[2] Bucher HF, Wrobel LC, Mebe JM, Magluta C. A novel approach to applying fast wavelet transforms in boundary element method. Electronic Journal of Boundary Elements, BETEQ 2002;2001(2):187–95.

[3] Daubechies I. Orthonormal bases of compactly supported wavelets. Communications on Pure and Applied Mathematics 1988;41:909–96.

[4] Davies GDV. Natural convection of air in a square cavity: a bench mark numerical solution. International Journal for Numerical Methods in Fluids 1983;3:249–64.

[5] Davies GDV, Jones IP. Natural convection in a square cavity: a comparison excercise. International Journal for Numerical Methods in Fluids 1983;3:227–48.

[6] Ghia U, Ghia KN, Shin CT. High-resolutions for incompressible flow using the Navier–Stokes equations and a multigrid method. Journal of Computational Physics 1982;48:387–411.

[7] González P, Cabaleiro JC, Pena TF. Parallel iterative solvers involving fast wavelet transforms for the solution of BEM systems. Advances in Engineering Software 2002;33(7–10):417–26.

[8] Harbrecht H, Konik M, Schneider R. Fully discrete wavelet Galerkin schemes. Engineering Analysis with Boundary Elements 2003;27(5): 423–37.

[9] Hriberšek M, Škerget L. Iterative methods in solving Navier–Stokes equations by the boundary element method. International Journal for Numerical Methods in Engineering 1996;39:115–39.

[10] Hriberšek M, Škerget L. Fast boundary-domain integral algorithm for the computation of incompressible fluid flow problems. International Journal for Numerical Methods in Fluids 1999;31:891–907.

[11] Koro K, Abe K. H-Hierarchical Adaptive BEM with Haar Wavelet Functions for Two-dimensional Laplace Problems. Boundary Elements XXII, Southampton: WIT Press; 1999. pp. 229–238.

[12] Koro K, Abe K. Non-orthogonal spline wavelets for boundary element analysis. Engineering Analysis with Boundary Elements 2001;25(3): 149–64.

[13] Koro K, Abe K. Determination of optimal threshold for matrix compression in wavelet BEM. In: Becsos DE, Brebbia CA, Katsikadelis JT, Manolis GD, editors. Boundary Elements XXIII. Southampton: WIT Press; 2001. p. 455–64.

[14] Koro K, Abe K. A practical determination strategy of optimal threshold parameter for matrix compression in wavelet BEM. International Journal for Numerical Methods in Engineering 2003; 57:169–91.

[15] Sleijpen GLG, Fokkema DR. BICGSTAB(l) for linear equations involving unsymmetric matrices with complex spectrum. Electronic Transactions on Numerical Analysis 1993;1:11–32.

[16] Škerget L, Hriberšek M, Žunič Z. Natural convection flows in complex cavities by BEM. International Journal of Numerical Methods for Heat and Fluid Flow 2003;13(6):720–35.

[17] Škerget L, Hriberšek M, Kuhn G. Computational fluid dynamics by boundary-domain integral method. International Journal for Numerical Methods in Engineering 1999;46:1291–311.

[18] Škerget L, Alujevič A, Brebbia CA, Kuhn G. Natural and forced convection simulation using velocity–vorticity approach. In: Brebbia CA, editor. Topics in Boundary Element Research, vol. 5. Berlin: Springer; 1989. Chapter 4.

[19] Wrobel LC. The Boundary Element Method. Applications in Thermo-fluids and Acoustics, vol. 1. New York: Wiley; 2002.