INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2007; **53**:1473–1494
Published online 27 September 2006 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1372

Edge-based data structures for a symmetric stabilized finite element method for the incompressible Navier–Stokes equations with heat transfer

R. A. Kraft[1,‡], A. L. G. A. Coutinho[1,*,†] and P. A. B. de Sampaio[2,§]

[1]*Programa de Engenharia Civil—COPPE, Universidade Federal do Rio de Janeiro, CP 68506, Rio de Janeiro-RJ, 21945-970, Brazil*
[2]*Instituto de Engenharia Nuclear—Comissão Nacional de Energia Nuclear, Rio de Janeiro-RJ, 21945-970, Brazil*

SUMMARY

This work presents a new implementation of the De Sampaio–Coutinho formulation (*Int. J. Numer. Meth. Fluids* 1999; **29**:289–309), a segregated Petrov–Galerkin/generalized least-squares method, for the solution of the incompressible Navier–Stokes equations with heat transfer. Such a formulation produces symmetric, positive-definite matrices, allowing the use of a preconditioned conjugate gradient solver for each unknown. The formulation also intrinsically introduces streamline *upwinding* by the choice of adequate time steps, providing a suitable description of convective dominated flows. The code was primarily written using element-based data structures, making use of parallel/vector techniques and mesh adaptivity. However, schemes based on edge-based data structures have been introduced with the aim of reducing flop count, memory demands and indirect addressing. In this work, the De Sampaio–Coutinho formulation has been re-written considering an edge-based arrangement. The effectiveness of the new scheme was observed solving some standard test cases, as shown in this paper, with a comparatively high gain in computational efficiency. Copyright © 2006 John Wiley & Sons, Ltd.

Received 14 March 2006; Revised 1 August 2006; Accepted 1 August 2006

KEY WORDS: finite element method; edge-based data structures; incompressible Navier–Stokes equations; Petrov–Galerkin methods

*Correspondence to: A. L. G. A. Coutinho, Programa de Engenharia Civil—COPPE, Universidade Federal do Rio de Janeiro, CP 68506, Rio de Janeiro-RJ, 21945-970, Brazil.
†E-mail: alvaro@nacad.ufrj.br
‡E-mail: r_kraft@coc.ufrj.br
§E-mail: sampaio@ien.gov.br

Contract/grant sponsor: CNPq; contract/grant numbers: 301251/2005-3, 301045/2003-8

Copyright © 2006 John Wiley & Sons, Ltd.

1. INTRODUCTION

In the finite element method, discrete equations can be obtained either from variational principles or from weighted residual methods. According to De Sampaio in Reference [1], the latter are mostly used in fluid mechanics as variational principles do not always exist. Apart from this, as the differential operators are non-self-adjoint, the Galerkin method loses its best approximation property and other formulations have to be used. Petrov–Galerkin methods, which make use of weighting functions applied to the residuals that are different from the interpolating functions used to expand the approximated solution, are used instead in the description of convective-diffusive phenomena, producing in addition streamline *upwinding*, providing an adequate description of convective dominated flows.

According to Löhner [2], in recent years, besides the significant progress on the development of numerical algorithms for the solution of Navier–Stokes equations, the use of unstructured meshes have allowed a better discretization of more complex geometries and eased the adaptation of the mesh to improve the solution. However, such unstructured meshes require the storage of the mesh connectivity, leading to an increase in the computer memory and the use of indirect addressing to retrieve nearest neighbour information. This implies that numerical algorithms will run slower on unstructured grids than on structured ones.

With the aim of reducing float point operations and indirect addressing, schemes based on edge-based data structures have been introduced, such as in the works by Barth [3], Luo *et al.* [4], Peraire *et al.* [5], and Venkatakrishnan and Mavriplis [6]. Additionally, as presented in Reference [7], more sophisticated data structures such as stars, super edges and chains have been developed. Catabriga and Coutinho [8] have shown that element matrices can be disassembled into their edge contributions, independently of the nature of the problem to be solved. According to these works, the use of edge-based data structures present significant savings in processing time for three-dimensional problems when compared with element-based ones. Another advantage of edge-based data structures pointed out in Reference [4] is its easy implementation, for both two- and three-dimensional problems, due to its similarity with the one-dimensional scheme, in which the connectivity is given by the initial and final nodes of the edge. Recently edge-based schemes have been introduced for incompressible flows, as in Reference [9], where an implicit second-order accurate monolithic scheme has been used. In that implementation the mass, gradients, and Laplacian matrices are computed and stored only once at the beginning of the run by using a standard compressed sparse row (CSR) format. According to the authors, the reduction in the total CPU time is mostly due to the higher speed with which LHS and RHS are constructed in the edge-based arrangement if compared with the element-based one.

Particularly in the case of the incompressible Navier–Stokes equations, the De Sampaio–Coutinho formulation, a segregated Petrov–Galerkin/generalized least-squares method type, presented here produces symmetric, positive-definite matrices, allowing the use of a preconditioned conjugate gradient solver for each unknown. The formulation also intrinsically introduces streamline *upwinding* by the choice of adequate local time steps, providing a suitable description of convective dominated flows. It is also developed such that C_0 class shape functions can be used, circumventing, this way, the Babuška–Brezzi condition. Another advantage is that the problem is solved in terms of primitive variables, i.e. pressure, components of velocity and temperature.

As in the original element-based implementation, optimal local time steps are used to advance the solution, whilst a time interpolation scheme is employed to provide the necessary synchronization [10]. The combined local time steps/time-interpolation scheme has shown good time accuracy

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

in several applications involving coupled flow and heat transfer [10] and in fluid–structure interaction problems [11]. Recently, De Sampaio [12, 13] has shown that the combined use of local time steps and time-interpolation, in the present context, is equivalent to the use of the Galerkin least-squares (GLS) method [14, 15], where the local time steps play the role of the stabilization parameter whilst the interpolation time step plays the role of the time step used to advance the solution in time. Note also that differently from Soto *et al.* [9], here the edge coefficients coming from nodes *ij* are not different from nodes *ji*, due to the symmetry of the underlying formulation.

The edge-based version of the De Sampaio–Coutinho formulation here presented preserves all the essential characteristics of the original element-based version, allowing the use of a preconditioned conjugate gradient solver in a segregated procedure for each of the variables involved, starting with the solution of pressure, and then updating velocity and temperature fields. There is only a minor difference regarding the evaluation of the local time steps. In the element-based implementation the local time steps are computed elementwise, whilst in this work they are computed elementwise and then projected to the edges. Because this is only a minor change in the representation of the local time step field, the differences observed in the computations using the element-based and the edge-based arrangements are negligible. The present work presents a comparison of both element-based and edge-based codes showing the effectiveness of the latter.

The choice of the edge-based data structure adoption in this work is reinforced by the results obtained by Ribeiro and Coutinho [16]. They show that, in a two-dimensional diffusion problem, with a domain discretized with an unstructured mesh composed of linear triangular elements, edge-by-edge (EDS) operations are more advantageous than those with other arrangements, such as element-by-element (EBE), CSR or compressed storage row with non-symmetric implementation, identified by CSR*, being only less attractive than CSR* when it comes to the number of indirect addressing operations. However, the computing times for routines involving Matrix–Vector (Matvec) products are the lowest with EDS and EDS* (so written to identify when groups of edges are used). It is also noticeable that better performances are obtained with EDS and EDS* when no strategy for renumbering nodes in the mesh to improve data locality is used.

The remainder of this article is organized as follows. In the next section, the governing equations for the incompressible Navier–Stokes equations coupled with heat transfer are presented. Following, the discrete formulation developed by De Sampaio and Coutinho is shown in Section 3, and in the sequence, the same formulation is presented in terms of edge-based data arrangement in Section 4. Numerical examples are then shown in Section 5 and a comparison of results between element-based and edge-based data structures is presented. Section 6 presents the conclusions of the present work.

2. GOVERNING EQUATIONS

The governing equations are written using the summation convention for $a=1,\ldots,nsd$ and $b=1,\ldots,nsd$, in Cartesian coordinates, being nsd the number of space dimensions. Thus, the incompressible continuity equation, in indicial notation, is given by

$$\frac{\partial(\rho_0 v_a)}{\partial x_a}=0 \tag{1}$$

The momentum conservation equation for each Cartesian component is given by

$$\rho_0\left(\frac{\partial v_a}{\partial t}+v_b\frac{\partial v_a}{\partial x_b}\right)-\frac{\partial \tau_{ab}}{\partial x_b}+\frac{\partial p}{\partial x_a}+\rho_0\beta g_a\theta=0 \tag{2}$$

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

and the energy conservation equation is

$$\rho_0 c_v \left(\frac{\partial \theta}{\partial t} + u_b \frac{\partial \theta}{\partial x_b} \right) + \frac{\partial q_b}{\partial x_b} = 0 \tag{3}$$

where, the viscous stress and the heat flux are written as

$$\tau_{ab} = \mu \left(\frac{\partial v_a}{\partial x_b} + \frac{\partial v_b}{\partial x_a} \right) \tag{4}$$

and

$$q_b = -\kappa \frac{\partial \theta}{\partial x_b} \tag{5}$$

with c_v being the specific heat, β, the thermal expansion coefficient, μ, the viscosity and κ, the thermal conductivity for the fluid. The fluid density at reference temperature $\theta = 0$ is designated by ρ_0, and v_a, p and θ represent, respectively, the velocity, pressure and temperature fields, and g_a, the Cartesian components of gravity.

Finally, the model is completed by the introduction of boundary and initial conditions, in which velocity and tractions are prescribed in regions of the boundary Γ_{va} and Γ_{ta}, such that $\Gamma_{va} \cup \Gamma_{ta} = \Gamma$ and $\Gamma_{va} \cap \Gamma_{ta} = \emptyset$,

$$v_a = \overline{v}_a(\mathbf{x}, t), \quad \mathbf{x} \in \Gamma_{va} \tag{6}$$

$$(-p\delta_{ab} + \tau_{ab}) n_b = \overline{t}_a(\mathbf{x}, t), \quad \mathbf{x} \in \Gamma_{ta} \tag{7}$$

where δ_{ab} is the Kronecker delta and n_b, the unit outward normal vector at the boundary.

The boundary conditions for temperature and heat flux are prescribed in the regions Γ_θ and Γ_q, such that $\Gamma_\theta \cup \Gamma_q = \Gamma$ and $\Gamma_\theta \cap \Gamma_q = \emptyset$,

$$\theta = \overline{\theta}(\mathbf{x}, t), \quad \mathbf{x} \in \Gamma_\theta \tag{8}$$

$$q_b n_b = \overline{q}(\mathbf{x}, t), \quad \mathbf{x} \in \Gamma_q \tag{9}$$

and pressure boundary conditions and mass flux, which are associated with the mass balance in the domain, are prescribed in the partitions Γ_p and Γ_G, such that $\Gamma_p \cup \Gamma_G = \Gamma$ and $\Gamma_p \cap \Gamma_G = \emptyset$,

$$p = \overline{p}(\mathbf{x}, t), \quad \mathbf{x} \in \Gamma_p \tag{10}$$

$$\rho_0 v_b n_b = \overline{G}(\mathbf{x}, t), \quad \mathbf{x} \in \Gamma_G \tag{11}$$

As noted in Reference [10], since the incompressible model involves only pressure gradients and not pressure itself, at least one value of pressure must be prescribed in order to define a unique pressure field.

3. DE SAMPAIO–COUTINHO FORMULATION

As presented in Reference [10], this finite element formulation is derived from the minimization of the momentum and energy squared residuals:

$$S = \int_\Omega \hat{F}_a \hat{F}_a \, \mathrm{d}\Omega \tag{12}$$

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

$$R = \int_\Omega \hat{E}^2 \, \mathrm{d}\Omega \tag{13}$$

where,

$$\hat{F}_a = \rho_0 \left(\frac{\hat{v}_a^{n+1} - \hat{v}_a^n}{\Delta t} + \hat{v}_b \frac{\partial \hat{v}_a^{n+1/2}}{\partial x_b} \right) - \frac{\partial \tau_{ab}^n}{\partial x_b} + \frac{\partial \hat{p}^{n+1/2}}{\partial x_a} + \rho_0 \beta g_a \hat{\theta}^n \tag{14}$$

and,

$$\hat{E} = \rho_0 c \left(\frac{\hat{\theta}^{n+1} - \hat{\theta}^n}{\Delta t} + \hat{v}_b \frac{\partial \hat{\theta}^{n+1/2}}{\partial x_b} \right) + \frac{\partial q_b^n}{\partial x_b} \tag{15}$$

The variables $\hat{v}_a$, $\hat{p}$ and $\hat{\theta}$ represent the discretized fields, interpolated by shape functions N_i. Nodal values for velocity, pressure and temperature are denoted, respectively, by v_{a_i}, p_i and θ_i. The superscripts n, $n+1$ and $n+1/2$ stand for the time level and Δt is the time step. Viscous stress, τ_{ab}^n, and heat flux, q_b^n, evaluated at time level n, are firstly taken as source terms, since their calculation involves second-order spatial derivatives (while the shape functions used are C_0 class). Such quantities, as indicated ahead, will be evaluated inside the elements, and not along element interfaces [10].

According to Reference [10], at time level $n+1$, the minimization of (12) with respect to v_a and after some manipulation leads to

$$\begin{aligned}
&\int_\Omega \left[\frac{\rho_0}{\Delta t} \left(N_i + \frac{\Delta t}{2} \hat{v}_c^n \frac{\partial N_i}{\partial x_c} \right) \left(\hat{v}_a^{n+1} + \frac{\Delta t}{2} \hat{v}_b^n \frac{\partial \hat{v}_a^{n+1}}{\partial x_b} \right) + \frac{\mu}{2} \frac{\partial N_i}{\partial x_b} \frac{\partial \hat{v}_a^{n+1}}{\partial x_b} \right] \mathrm{d}\Omega \\
&= \int_\Omega \left[\frac{\rho_0}{\Delta t} \left(N_i + \frac{\Delta t}{2} \hat{v}_c^n \frac{\partial N_i}{\partial x_c} \right) \left(\hat{v}_a^n - \frac{\Delta t}{2} \hat{v}_b^n \frac{\partial \hat{v}_a^n}{\partial x_b} \right) - \frac{\mu}{2} \frac{\partial N_i}{\partial x_b} \frac{\partial \hat{v}_a^n}{\partial x_b} \right] \mathrm{d}\Omega \\
&\quad - \int_\Omega \frac{\partial N_i}{\partial x_a} \hat{p}^{n+1/2} \, \mathrm{d}\Omega - \int_\Omega \frac{\Delta t}{2} \hat{v}_c^n \frac{\partial N_i}{\partial x_c} \frac{\partial \hat{p}^{n+1/2}}{\partial x_a} \, \mathrm{d}\Omega \\
&\quad + \int_{\Gamma_{ta}} N_i \bar{t}_a \, \mathrm{d}\Gamma - \int_\Omega \left(N_i + \frac{\Delta t}{2} \hat{v}_c^n \frac{\partial N_i}{\partial x_c} \right) \rho_0 \beta g_a \hat{\theta}^n \, \mathrm{d}\Omega \\
&\quad + \int_\Omega \frac{\Delta t}{2} \hat{v}_c^n \frac{\partial N_i}{\partial x_c} \frac{\partial \tau_{ab}^n}{\partial x_c} \, \mathrm{d}\Omega \quad \forall \text{ free } v_{a_i}^{n+1}
\end{aligned} \tag{16}$$

minimizing (12) again with respect to p, combining the result with the mass conservation equation (1) and after some mathematical manipulation the following equation is obtained:

$$\begin{aligned}
\int_\Omega \Delta t \frac{\partial N_i}{\partial x_a} \frac{\partial \hat{p}^{n+1/2}}{\partial x_a} \, \mathrm{d}\Omega &= -\int_\Omega \Delta t \frac{\partial N_i}{\partial x_a} \rho_0 \hat{v}_b^n \frac{\partial \hat{v}_a^n}{\partial x_b} \, \mathrm{d}\Omega - \int_\Omega \rho_0 N_i \frac{\partial \hat{v}_a^n}{\partial x_a} \, \mathrm{d}\Omega + \int_\Omega \Delta t \frac{\partial N_i}{\partial x_a} \frac{\partial \tau_{ab}^n}{\partial x_b} \, \mathrm{d}\Omega \\
&\quad - \int_\Omega \Delta t \frac{\partial N_i}{\partial x_a} \rho_0 \beta g_a \hat{\theta}^n \, \mathrm{d}\Omega - \int_{\Gamma_G} N_i (\overline{G}^{n+1} - \overline{G}^n) \, \mathrm{d}\Gamma \quad \forall \text{ free } p_i^{n+1/2}
\end{aligned} \tag{17}$$

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

The minimization of (13) with respect to θ, also at time level $n+1$, and again after some manipulation, provides:

$$\int_\Omega \left[\frac{\rho_0 c}{\Delta t} \left(N_i + \frac{\Delta t}{2} \hat{v}_c^n \frac{\partial N_i}{\partial x_c} \right) \left(\hat{\theta}^{n+1} + \frac{\Delta t}{2} \hat{v}_b^n \frac{\partial \hat{\theta}^{n+1}}{\partial x_b} \right) + \frac{\kappa}{2} \frac{\partial N_i}{\partial x_b} \frac{\partial \hat{\theta}^{n+1}}{\partial x_b} \right] \mathrm{d}\Omega$$

$$= \int_\Omega \left[\frac{\rho_0 c}{\Delta t} \left(N_i + \frac{\Delta t}{2} \hat{v}_c^n \frac{\partial N_i}{\partial x_c} \right) \left(\hat{\theta}^{n} - \frac{\Delta t}{2} \hat{v}_b^n \frac{\partial \hat{\theta}^{n}}{\partial x_b} \right) - \frac{\kappa}{2} \frac{\partial N_i}{\partial x_b} \frac{\partial \hat{\theta}^{n}}{\partial x_b} \right] \mathrm{d}\Omega$$

$$- \int_{\Gamma_q} N_i \overline{q} \, \mathrm{d}\Gamma - \int_\Omega \frac{\Delta t}{2} \hat{v}_c^n \frac{\partial N_i}{\partial x_c} \frac{\partial q_b^n}{\partial x_b} \, \mathrm{d}\Omega \quad \forall \text{ free } \theta_i^{n+1} \tag{18}$$

The traction and heat flux boundary conditions given by (7) and (9) are approximated by:

$$\int_{\Gamma_{ta}} N_i (-p\delta_{ab} + \tau_{ab}) n_b \, \mathrm{d}\Gamma = \int_{\Gamma_{ta}} N_i \overline{t}_a \, \mathrm{d}\Gamma \tag{19}$$

$$\int_{\Gamma_q} N_i q_b^n n_b \, \mathrm{d}\Gamma = \int_{\Gamma_q} N_i \overline{q} \, \mathrm{d}\Gamma \tag{20}$$

At this point, viscous terms and heat flux must be replaced in Equations (16)–(18) by their corresponding discretized forms. In terms of discretized velocity and temperature they become:

$$\hat{\tau}_{ab}^n = \mu \left(\frac{\partial \hat{v}_a}{\partial x_b} + \frac{\partial \hat{v}_b}{\partial x_a} \right)^n \tag{21}$$

and,

$$\hat{q}_b^n = -\kappa \frac{\partial \hat{\theta}^n}{\partial x_b} \tag{22}$$

By doing so, the equivalence of Equations (16)–(18) with the least-squares method is lost, since it would be necessary to use of C_1 class shape functions. However, the present formulation inherits from the former the properties of symmetry and positive definiteness.

As stated before, the resulting system of equations is solved in a segregated solution procedure, in which pressure is computed first and then velocity and temperature fields are updated. Equations (16)–(18) lead to systems of equations that can be written in matrix form as:

$$\mathbf{A}_P \mathbf{p}^{n+1/2} = \mathbf{b}_P \tag{23}$$

$$\mathbf{A}_{v_x} \mathbf{v}_x^{n+1} = \mathbf{b}_x \tag{24}$$

$$\mathbf{A}_{v_y} \mathbf{v}_y^{n+1} = \mathbf{b}_y \tag{25}$$

$$\mathbf{A}_\theta \boldsymbol{\theta}^{n+1} = \mathbf{b}_\theta \tag{26}$$

Matrices $\mathbf{A}_p$, $\mathbf{A}_{v_x}$, $\mathbf{A}_{v_y}$ and $\mathbf{A}_\theta$ are symmetric positive-definite and allow the use of a Jacobi-preconditioned conjugate gradient solver for each unknown. The EBE matrix–vector product needed

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

in the iterative solver can be expressed by

$$\mathbf{Ax} = \sum_{e=1}^{n_{\text{elem}}} \mathbf{A}_e \mathbf{x}_e \tag{27}$$

where n_{elem} stands for the number of elements in the mesh. This multiplication in an EBE scheme can be summarized in the algorithm below:

```
do e = 1, n_elem
    gather          p_e from p
    compute         ap = A_e p
    scatter + add   p = p + ap
end do
```

Another important aspect of the method is that the weighting functions derived from the minimization of (12) and which are applied to Equations (16) and have the *streamline upwind Petrov–Galerkin* structure [17]:

$$W_i = N_i + \frac{\Delta t}{2} \hat{v}_c^n \frac{\partial N_i}{\partial x_c} \tag{28}$$

According to Reference [10], for linear finite triangular elements, an adequate amount of *streamline upwinding* is introduced in the momentum balance by choosing the time step as

$$\Delta t = \left[\coth\left(\frac{Re}{2}\right) - \frac{2}{Re} \right] \frac{h_e}{\|\mathbf{v}^n\|} \tag{29}$$

with $\|\mathbf{v}^n\|$ being the velocity modulus and h_e, the characteristic element size, calculated as the square root of the element area. The element Reynolds number is determined by $Re = h_e \|\mathbf{v}^n\| \rho / \mu$.

It can also be noted that when the flow tends to pure convection, that is, $Re \to \infty$, the time step in (29) tends to,

$$\Delta t = h_e / \|\mathbf{v}^n\| \tag{30}$$

whilst, for pure diffusion, i.e. $Re = 0$, it is given by

$$\Delta t = \rho h_e^2 / 6\mu \tag{31}$$

Such a time step is also called *intrinsic time scale*. It must be also noted that in order to introduce optimal *upwinding* in the fluid energy, the element Reynolds number in (29) must be replaced by the element Peclet number, which is given by $Pe = RePr$, where $Pr = \mu c / \kappa$ is the Prandtl number.

Since the time step given by (29) varies in space with mesh size, physical properties and velocities, and, furthermore, if optimal *upwinding* is to be introduced in both momentum and energy equations, then distinct intrinsic time scales are to be used. In Reference [10], time steps to advance the solution and to provide an adequate streamline *upwinding* are taken as being the same. Such time steps are determined along the loop for the segregated solution of each degree of freedom, for each node, which implies that different values are obtained. Therefore, in order to allow each degree of freedom to advance in time according to its own local time step while results are outputted at fixed times, a procedure involving freezing 'inactive' variables, treating them as boundary conditions and solving 'active' equations and interpolating results for the desired output time is adopted.

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

As an alternative procedure, the use of a unique time step for the mesh is proposed in References [12, 13], with adjustments to the equations of pressure and momentum such as to embed local time step and the synchronization phase mentioned above. Local time steps are used in this new formulation only with stabilization purposes and the algorithm turns out to be more easily implemented.

4. THE EDGE-BASED SCHEME

Equations (16) and (17) can be discretized using finite elements and matrix–vector operations can be performed in a standard EBE scheme, as indicated in Reference [10]. In the present work, data is arranged such that operations involving the matrix–vector multiplication and RHS evaluation are performed EDS.

Edge matrices are written considering the contributions of the elements adjacent to the edge under consideration, as indicated in Figure 1. The resulting scheme showing how an element matrix for a linear triangular element with one degree of freedom per node can be written in terms of its edges contributions is shown below:

$$\underbrace{\begin{bmatrix} \otimes & \otimes & \otimes \\ \otimes & \otimes & \otimes \\ \otimes & \otimes & \otimes \end{bmatrix}}_{\text{Element } e} = \underbrace{\begin{bmatrix} \otimes & \otimes & 0 \\ \otimes & \otimes & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{\text{Edge } ij} + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & \otimes & \otimes \\ 0 & \otimes & \otimes \end{bmatrix}}_{\text{Edge } jk} + \underbrace{\begin{bmatrix} \otimes & 0 & \otimes \\ 0 & 0 & 0 \\ \otimes & 0 & \otimes \end{bmatrix}}_{\text{Edge } ki} \tag{32}$$

and thus, from (32), the matrix corresponding to the edge s can be written in a schematic way as

$$\underbrace{\begin{bmatrix} \otimes & \otimes \\ \otimes & \otimes \end{bmatrix}}_{\text{Edge } s} = \underbrace{\begin{bmatrix} \oplus & \oplus \\ \oplus & \oplus \end{bmatrix}}_{\text{Edge } e} + \underbrace{\begin{bmatrix} \oplus & \oplus \\ \oplus & \oplus \end{bmatrix}}_{\text{Edge } f} \tag{33}$$

All matrices involved are then written in terms of edge data aiming to allow that a code based only on the edge information is written.

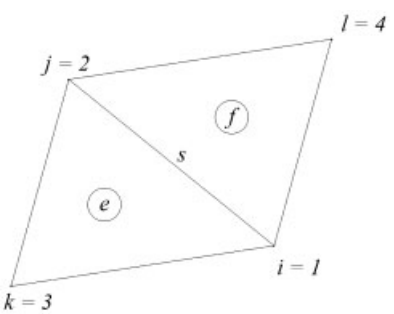

Figure 1. Elements e and f adjacent to edge s.

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

In matrix notation, the equation of pressure (17) can be expressed as

$$\mathbf{TDDp}^{n+1} = -\rho_0 \mathbf{TB}_x \mathbf{v}_x^n - \frac{\rho_0}{\Delta t}\mathbf{TA}_x \mathbf{v}_x^n - \rho_0 \mathbf{TB}_y \mathbf{v}_y^n - \frac{\rho_0}{\Delta t}\mathbf{TA}_y \mathbf{v}_y^n - \rho_0 g_x \mathbf{TA}_x^{\mathrm{T}} \boldsymbol{\theta}^n - \rho_0 g_y \mathbf{TA}_y^{\mathrm{T}} \boldsymbol{\theta}^n \text{—boundary terms} \tag{34}$$

In our formulations, elements e and f are considered as having their connectivity given by local nodes 1–2–3 and 1–4–2, respectively. In order to keep consistency, shape functions derivatives for each element are written as

$$\mathbf{B}^e = \frac{\partial N_i^e}{\partial x_a} = \begin{bmatrix} \dfrac{\partial N_i}{\partial x} \\ \dfrac{\partial N_i}{\partial y} \end{bmatrix} = \frac{1}{2A_{\mathrm{el}}^e} \begin{bmatrix} b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{bmatrix} \tag{35}$$

and,

$$\mathbf{B}^f = \frac{\partial N_i^f}{\partial x_a} = \begin{bmatrix} \dfrac{\partial N_i}{\partial x} \\ \dfrac{\partial N_i}{\partial y} \end{bmatrix} = \frac{1}{2A_{\mathrm{el}}^f} \begin{bmatrix} b_5 & b_4 & b_6 \\ c_5 & c_4 & c_6 \end{bmatrix} \tag{36}$$

where the area of each element is

$$A_{\mathrm{el}}^e = \frac{a_1 + a_2 + a_3}{2} \tag{37}$$

$$A_{\mathrm{el}}^f = \frac{a_4 + a_5 + a_6}{2} \tag{38}$$

and,

$$\begin{aligned} a_1 &= x_2 y_3 - x_3 y_2 \\ b_1 &= y_2 - y_3 \\ c_1 &= x_3 - x_2 \end{aligned} \tag{39}$$

$$\begin{aligned} a_2 &= x_3 y_1 - x_1 y_3 \\ b_2 &= y_3 - y_1 \\ c_2 &= x_1 - x_3 \end{aligned} \tag{40}$$

$$\begin{aligned} a_3 &= x_1 y_2 - x_2 y_1 \\ b_3 &= y_1 - y_2 \\ c_3 &= x_2 - x_1 \end{aligned} \tag{41}$$

In the case where there are two elements adjacent to edge s, the coefficients relative to this second element are determined by

$$\begin{aligned} a_4 &= x_2 y_1 - x_1 y_2 \\ b_4 &= y_2 - y_1 \\ c_4 &= x_1 - x_2 \end{aligned} \tag{42}$$

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

$$a_5 = x_4 y_2 - x_2 y_4$$
$$b_5 = y_4 - y_2 \tag{43}$$
$$c_5 = x_2 - x_4$$

$$a_6 = x_1 y_4 - x_4 y_1$$
$$b_6 = y_1 - y_4 \tag{44}$$
$$c_6 = x_4 - x_1$$

Considering the matrix arrangement in (33), the pressure matrix on the left-hand side of (34) is written as

$$\mathbf{TDD}_{ij} = \mathbf{TDD}^e_{ij} + \mathbf{TDD}^f_{ij} = \begin{bmatrix} -T_{ij} & T_{ij} \\ T_{ij} & -T_{ij} \end{bmatrix} \tag{45}$$

In the calculations, a loop is carried out along the edges, for the current edge, in terms of local nodal numbering, we can write,

$$\mathbf{TDD}_{12} = \mathbf{TDD}^e_{12} + \mathbf{TDD}^f_{12} = \begin{bmatrix} -T_{12} & T_{12} \\ T_{12} & -T_{12} \end{bmatrix} \tag{46}$$

where the coefficient T_{12} is calculated by

$$T_{12} = T^e_{12} + T^f_{12} = \frac{b_1 b_2 + c_1 c_2}{4A^e_{\text{el}}} + \frac{b_5 b_6 + c_5 c_6}{4A^f_{\text{el}}} \tag{47}$$

For the matrices on the right-hand side of (34), due to the non-symmetry of the obtained matrices, the following approximations have been made:

$$\mathbf{A}_x = \frac{1}{6}\begin{bmatrix} b_1 & b_2 & b_3 \\ b_1 & b_2 & b_3 \\ b_1 & b_2 & b_3 \end{bmatrix} = \frac{1}{6}\begin{bmatrix} \frac{b_1}{2} & b_2 & 0 \\ b_1 & \frac{b_2}{2} & 0 \\ 0 & 0 & 0 \end{bmatrix} + \frac{1}{6}\begin{bmatrix} \frac{b_1}{2} & 0 & b_3 \\ 0 & 0 & 0 \\ b_1 & 0 & \frac{b_3}{2} \end{bmatrix} + \frac{1}{6}\begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{b_2}{2} & b_3 \\ 0 & b_2 & \frac{b_3}{2} \end{bmatrix} \tag{48}$$

$$\mathbf{TA}_x = \frac{1}{6}\begin{bmatrix} \frac{b_1 + b_5}{2} & b_2 + b_6 \\ b_1 + b_5 & \frac{b_2 + b_6}{2} \end{bmatrix} \tag{49}$$

$$\mathbf{TA}_y = \frac{1}{6}\begin{bmatrix} \frac{c_1 + c_5}{2} & c_2 + c_6 \\ c_1 + c_5 & \frac{c_2 + c_6}{2} \end{bmatrix} \tag{50}$$

$$\mathbf{TB}_x = \frac{1}{12A^e_{\text{el}}}\begin{bmatrix} b_1(b_1(u_1+u_2+u_3)+c_1(v_1+v_2+v_3))/2 & b_1(b_2(u_1+u_2+u_3)+c_2(v_1+v_2+v_3)) \\ b_2(b_1(u_1+u_2+u_3)+c_1(v_1+v_2+v_3)) & b_2(b_2(u_1+u_2+u_3)+c_2(v_1+v_2+v_3))/2 \end{bmatrix}$$

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

$$+\frac{1}{12A_{\mathrm{el}}^{f}}\begin{bmatrix} b_5(b_5(u_1+u_2+u_4)+c_5(v_1+v_2+v_4))/2 & b_5(b_6(u_1+u_2+u_4)+c_6(v_1+v_2+v_4)) \\ b_6(b_5(u_1+u_2+u_4)+c_5(v_1+v_2+v_4)) & b_6(b_6(u_1+u_2+u_4)+c_6(v_1+v_2+v_4))/2 \end{bmatrix} \tag{51}$$

$$\mathbf{TB}_y=\frac{1}{12A_{\mathrm{el}}^{e}}\begin{bmatrix} c_1(b_1(u_1+u_2+u_3)+c_1(v_1+v_2+v_3))/2 & c_1(b_2(u_1+u_2+u_3)+c_2(v_1+v_2+v_3)) \\ c_2(b_1(u_1+u_2+u_3)+c_1(v_1+v_2+v_3)) & c_2(b_2(u_1+u_2+u_3)+c_2(v_1+v_2+v_3))/2 \end{bmatrix}$$
$$l+\frac{1}{12A_{\mathrm{el}}^{f}}\begin{bmatrix} c_5(b_5(u_1+u_2+u_4)+c_5(v_1+v_2+v_4))/2 & c_5(b_6(u_1+u_2+u_4)+c_6(v_1+v_2+v_4)) \\ c_6(b_5(u_1+u_2+u_4)+c_5(v_1+v_2+v_4)) & c_6(b_6(u_1+u_2+u_4)+c_6(v_1+v_2+v_4))/2 \end{bmatrix} \tag{52}$$

The second term at the right side of (51) and (52) is considered only if element f, with local connectivity 1–4–2, exists. In the case of velocity equations, (16) can be written in matrix form as

$$\begin{aligned}&\left[\frac{\rho_0}{\Delta t}\mathbf{TM}+\frac{\rho_0}{2}(\mathbf{TCC}+\mathbf{TCC}^{\mathrm{T}})+\frac{\rho_0}{2}\frac{\Delta t}{2}\mathbf{TEE}+\frac{\mu}{2}\mathbf{TDD}\right]\mathbf{v}_a^{n+1}\\ &\quad=\left(\frac{\rho_0}{\Delta t}\mathbf{TM}-\frac{\rho_0}{2}(\mathbf{TCC}-\mathbf{TCC}^{\mathrm{T}})-\frac{\rho_0\Delta t}{4}\mathbf{TEE}-\frac{\mu}{2}\mathbf{TDD}\right)\mathbf{v}_a^{n}\\ &\qquad-\left(\mathbf{TA}_a+\frac{\Delta t}{2}\mathbf{TB}_a^{\mathrm{T}}\right)\mathbf{p}^{n+1/2}\\ &\qquad-\rho_0\beta g_a\left(\mathbf{TM}+\frac{\Delta t}{2}\mathbf{TCC}^{\mathrm{T}}\right)\boldsymbol{\theta}^{n}\text{—boundary terms}\end{aligned} \tag{53}$$

Similarly to the equations of pressure, the matrices on the left-hand side of (53) are symmetric and can be calculated accordingly to (32) and (33) by

$$\mathbf{TM}=\frac{A_{\mathrm{el}}^{e}}{12}\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix} \tag{54}$$

$$\mathbf{TCC}=\frac{1}{24}\begin{bmatrix}\dfrac{b_1(2u_1+u_2+u_3)+c_1(2v_1+v_2+v_3)}{2} & b_2(2u_1+u_2+u_3)+c_2(2v_1+v_2+v_3)\\ b_1(u_1+2u_2+u_3)+c_1(v_1+2v_2+v_3) & \dfrac{b_2(u_1+2u_2+u_3)+c_2(v_1+2v_2+v_3)}{2}\end{bmatrix} \tag{55}$$

and, if element f exists, the coefficients relative to it must be added to each matrix,

$$\mathbf{TM}=\mathbf{TM}+\frac{A_{\mathrm{el}}^{f}}{12}\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix} \tag{56}$$

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

$$\mathbf{TCC} = \mathbf{TCC} + \frac{1}{24}\begin{bmatrix} \dfrac{b_5(2u_1+u_2+u_4)+c_5(2v_1+v_2+v_4)}{2} & b_6(2u_1+u_2+u_4)+c_6(2v_1+v_2+v_4) \\ b_5(u_1+2u_2+u_4)+c_5(v_1+2v_2+v_4) & \dfrac{b_6(u_1+2u_2+u_4)+c_6(v_1+2v_2+v_4)}{2} \end{bmatrix} \tag{57}$$

It must be noted that, despite (57) is not symmetric, when summed up to its transpose on the LHS of (53), symmetric matrix is produced and the conjugate gradient solving strategy can be kept.

Matrix **TEE** can be written as

$$\mathbf{TEE} = \mathbf{TEE}^e + \mathbf{TEE}^f = \begin{bmatrix} -TEE_{12} & TEE_{12} \\ TEE_{12} & -TEE_{12} \end{bmatrix} \tag{58}$$

where the coefficient TEE_{12} is obtained from

$$\begin{aligned} TEE_{12} = {} & \frac{(u_1b_1 + v_1c_1)}{48A^e_{\text{el}}}[b_2(2u_1 + u_2 + u_3) + c_2(2v_1 + v_2 + v_3)] \\ & + \frac{(u_2b_1 + v_2c_1)}{48A^e_{\text{el}}}[b_2(u_1 + 2u_2 + u_3) + c_2(v_1 + 2v_2 + v_3)] \\ & + \frac{(u_3b_1 + v_3c_1)}{48A^e_{\text{el}}}[b_2(u_1 + u_2 + 2u_3) + c_2(v_1 + v_2 + 2v_3)] \end{aligned} \tag{59}$$

and, if element f exists, the following must be added to (59)

$$\begin{aligned} TEE_{12} = {} & TEE_{12} + \frac{(u_1b_5 + v_1c_5)}{48A^f_{\text{el}}}[b_6(2u_1 + u_2 + u_4) + c_6(2v_1 + v_2 + v_4)] \\ & + \frac{(u_2b_5 + v_2c_5)}{48A^f_{\text{el}}}[b_6(u_1 + 2u_2 + u_4) + c_6(v_1 + 2v_2 + v_4)] \\ & + \frac{(u_4b_5 + v_4c_5)}{48A^f_{\text{el}}}[b_6(u_1 + u_2 + 2u_4) + c_6(v_1 + v_2 + 2v_4)] \end{aligned} \tag{60}$$

Matrices on the right-hand side of (53) have already been determined, having just to be noted that in $\mathbf{TA}_a$ and $\mathbf{TB}_a$ the subscript a stands for the Cartesian component of velocity being evaluated. In edge terms, the temperature equation (18) can be written as

$$\begin{aligned} & \left[\rho_0 c\mathbf{TM} + \frac{\rho_0 c\Delta t}{2}(\mathbf{TCC} + \mathbf{TCC}^{\mathrm{T}}) + \frac{\rho_0 c\Delta t^2}{4}\mathbf{TEE} + \frac{\kappa\Delta t}{2}\mathbf{TDD}\right]\boldsymbol{\theta}^{n+1} \\ & = \left[\rho_0 c\mathbf{TM} - \frac{\rho_0 c\Delta t}{2}(\mathbf{TCC} - \mathbf{TCC}^{\mathrm{T}}) - \frac{\rho_0 c\Delta t^2}{4}\mathbf{TEE}\right. \\ & \quad \left. - \frac{\kappa}{2}\mathbf{TDD}\right]\boldsymbol{\theta}^n — \text{boundary terms} \end{aligned} \tag{61}$$

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

where all matrices indicated have already been defined in the previous items. Once the matrices involved are defined, the strategy to solve the system of resulting equations is discussed.

Time steps to advance the solution and to provide an adequate streamline *upwinding* are taken here as in Reference [10] and a similar procedure for treating 'active' and 'inactive' variables is adopted for the solution advance in time. Soto *et al.* [9] describe the determination of an intrinsic time used in the stabilization of the formulation there presented which is different from the time step used to advance the solution in the simulation. Nodal evaluations are made to determine the time to stabilize convective terms and edge evaluations are made to determine the adequate time to provide stabilization to the incompressible continuity terms.

Though other procedures are possible, in the present work, if there are two elements adjacent to an edge, the time step associated with it is determined as being the average of the time steps evaluated in the barycentre of each element separately, as follows:

$$\Delta t = \frac{\Delta t^e + \Delta t^f}{2} \tag{62}$$

with element time steps calculated as shown in Section 3.

Since matrices obtained on the left-hand side of all systems of equations are symmetric positive definite, an EDS procedure using a Jacobi-preconditioned conjugate gradient solver is adopted. To allow parallelization of calculations in cache-based shared memory systems, the mesh is coloured as described in Reference [10]. Although important, no edge or node reordering as used in References [16, 18] to improve data locality is considered.

As mentioned before, a segregated method of solution is adopted, where systems of equations as summarized in Equations (23)–(26) are obtained. The matrix–vector product in the edge-based arrangement is performed according to

$$\mathbf{Ax} = \sum_{s=1}^{n_{\text{edges}}} \mathbf{A}_s \mathbf{x}_s \tag{63}$$

where n_{edges} is the number of edges in the mesh. Calculations are then performed similarly to the EBE scheme.

Comparing the operations involved in the EBE matrix–vector product algorithm and the EDS one, in two-dimensional problems, since we are calculating 4 degrees of freedom per node and as the number of edges in unstructured meshes tends to be 1.5 times the number of elements in the mesh, we arrive to the number of floating point operations (flop) and indirect addressing (I/A) estimates in Table I.

From the analysis of the data in Table I, it is possible to compare how the edge-based data arrangement can contribute to diminish computing time.

Table I. Estimate of indirect addressing and number of floating point operations in the EBE and EDS arrangements.

Parameter	Element-by-element (EBE)	Edge-by-edge (EDS)
I/A	$4 * 9 * n_{\text{elem}} = 36 * n_{\text{elem}}$	$4 * 6 * 1.5 * n_{\text{elem}} = 36 * n_{\text{elem}}$
Flop	$4 * 18 * n_{\text{elem}} = 72 * n_{\text{elem}}$	$4 * 8 * 1.5 * n_{\text{elem}} = 48 * n_{\text{elem}}$

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

Table II. Memory demand comparison in the Matvec product.

Data structure	With symmetry	Symmetry + conservation
Element-by-element	$24 * n_{\text{elem}}$	$21 * n_{\text{elem}}$
Edge-by-edge	$12 * 1.5 * n_{\text{elem}} = 18 * n_{\text{elem}}$	$10 * 1.5 * n_{\text{elem}} = 15 * n_{\text{elem}}$

Though this work does not treat three-dimensional problems, it is worth to notice that since an edge incidence is always the same in both two- and three-dimensional domains, the algorithm for the matrix–vector product is therefore always the same, involving the same number of indirect addressing and floating point operations. That means that the gains in three-dimensional problems are potentially even bigger.

Concerning memory use, taking into consideration the operations involved in the calculation of coefficients of the matrices on the left-hand side of equations for pressure, velocities and temperature for both edge and element-based arrangements, it is possible to summarize the memory demand for each arrangement considering the whole mesh as shown in Table II.

In this table we computed memory positions using the property of symmetry only and symmetry associated with the conservation property of the shape-functions derivatives [2], in order to reduce the positions storage to a minimum.

5. NUMERICAL EXAMPLES

Here results for some problems using both element-based and edge-based data arrangements are shown. For convenience, the governing equations are written in non-dimensional form. The relationship between dimensional and non-dimensional variables is defined by $x'_a = x'_a/L$; $t' = \mu t/\rho_o L^2$; $v'_a = \rho_o v_a L/\mu$; $\theta' = \theta/\Delta\theta$ and $p' = \rho_o p L^2/\mu^2$, being $\Delta\theta$ and L, respectively, the reference temperature difference and the length scale for the problem under consideration. Thus, Equations (1)–(3) are rewritten as

$$\frac{\partial v'_a}{\partial x'_a} = 0 \tag{64}$$

$$\frac{\partial v'_a}{\partial t'} + v'_b \frac{\partial v'_a}{\partial x'_b} - \frac{\partial}{\partial x_b}\left(\frac{\partial v'_a}{\partial x'_b} + \frac{\partial v'_b}{\partial x'_a}\right) + \frac{\partial p'}{\partial x'_a} + Gr\gamma_a\theta' = 0 \tag{65}$$

$$Pr\left(\frac{\partial \theta'}{\partial t'} + v'_b \frac{\partial \theta'}{\partial x'_b}\right) - \frac{\partial}{\partial x'_b}\left(\frac{\partial \theta'}{\partial x'_b}\right) = 0 \tag{66}$$

where

$$\gamma_a = \frac{g_a}{\|\mathbf{g}\|} \tag{67}$$

being the Grashof and Prandtl numbers calculated as

$$Gr = \frac{\rho_0^2 \beta \|\mathbf{g}\| \Delta\theta L^3}{\mu^2} \tag{68}$$

$$Pr = \frac{c\mu}{\kappa} \tag{69}$$

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

Next, three examples involving free and forced convection are shown. In the simulations performed, a single processor computer using Athlon XP 3000+, with 1 Gb DDR 400 MHz memory and OS Windows 2000 has been used.

5.1. Thermal stratification in a square cavity

In this example, the fluid inside a square cavity is initially at rest and under thermal equilibrium, when a sudden constant temperature difference is applied and kept along two opposite vertical walls. Temperature on the left wall is increased by $0.5\Delta\theta$, while the temperature on the right is decreased by the same amount. The simulation is carried out during the time interval $t' = 0$–5.0, considering the product $GrPr = 10^4$ and $Pr = 0.72$.

The domain is discretized by means of a fixed, unstructured mesh containing 7477 nodes and 14 632 elements. The number of edges in the mesh is 22 108. Mesh and boundary conditions are shown in Figure 2.

Results for temperature distribution and velocity field at the end of the simulation are shown in Figures 3 and 4. It can be noticed that the temperature distribution shown at Figure 3 presents a similar pattern to results shown in Reference [10], though the fluid properties are slightly different there and an adaptive mesh has been used.

Table III shows that the number of floating point operations performed with the EDS arrangement is 67.15% of the one with EBE and memory demand, 71.99%. Though the number of indirect addressing operations is 1.064 times bigger with EDS, it can be noted that EDS is much more

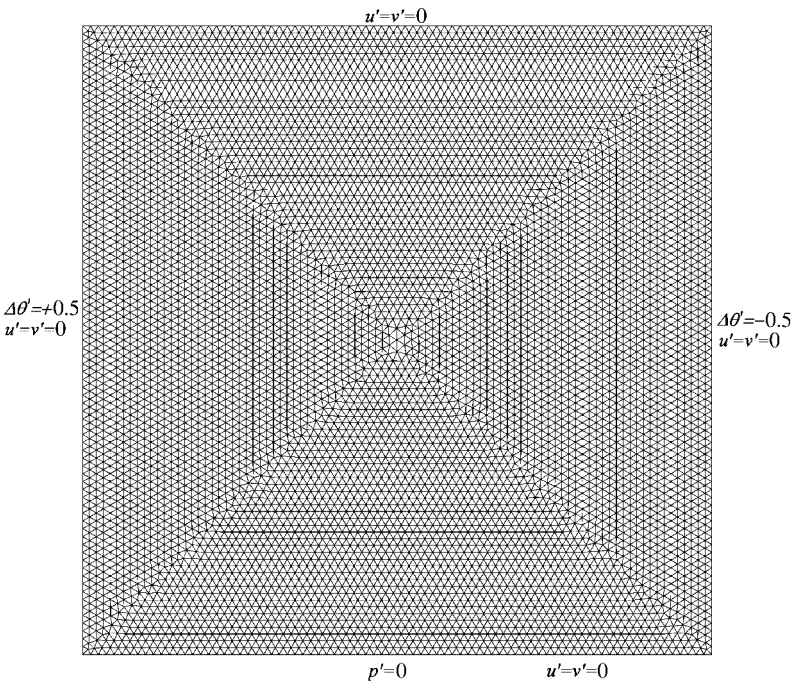

Figure 2. Mesh and boundary conditions for the thermal stratification problem in a square cavity.

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

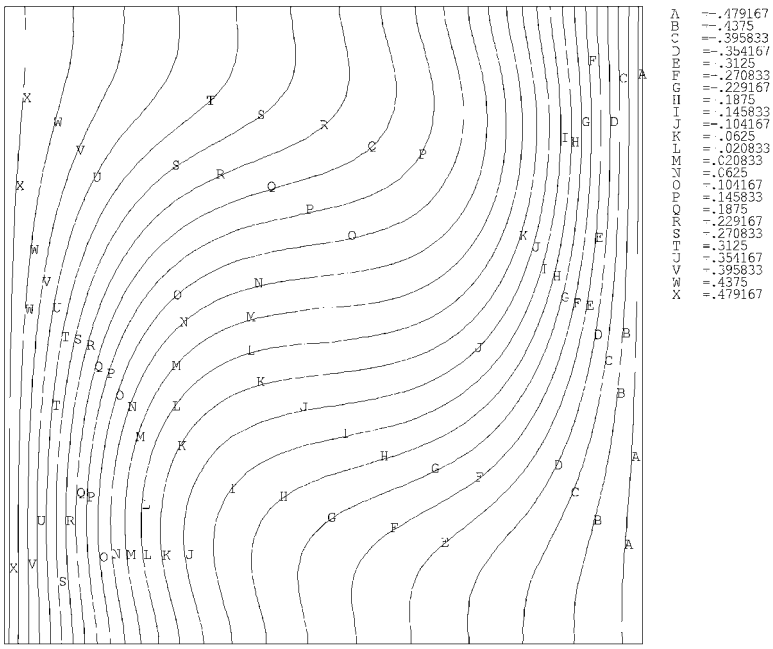

Figure 3. Temperature distribution at $t' = 5.0$.

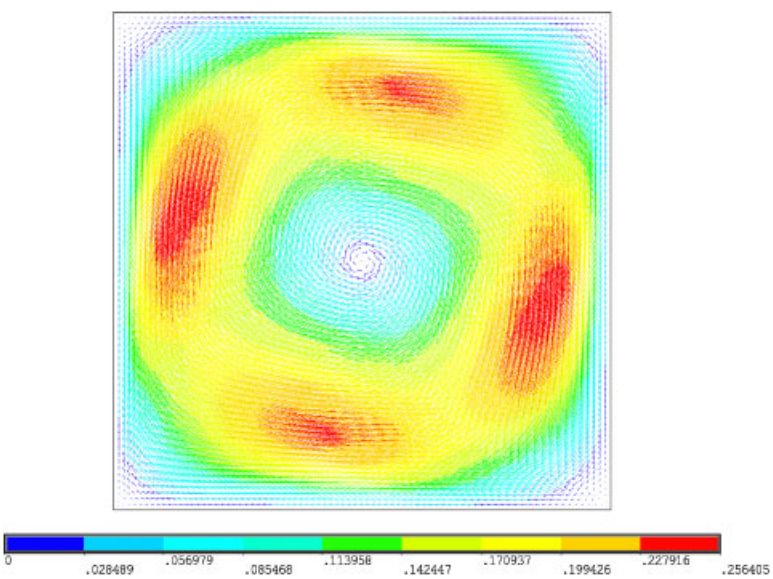

Figure 4. Velocity field at $t' = 5.0$.

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

Table III. Element-by-element and edge-by-edge performance for the thermal stratification problem in a square cavity.

Data arrangement	Element-by-element	Edge-by-edge
Number of time steps	16 351	16 351
CPU time (s)	44 975.859	16 564.000
Memory demand (Mwords)	0.307	0.221
Floating point operations (Mflop)	1.0535	0.7074
I/A $(*10E+6)$	0.5268	0.5306

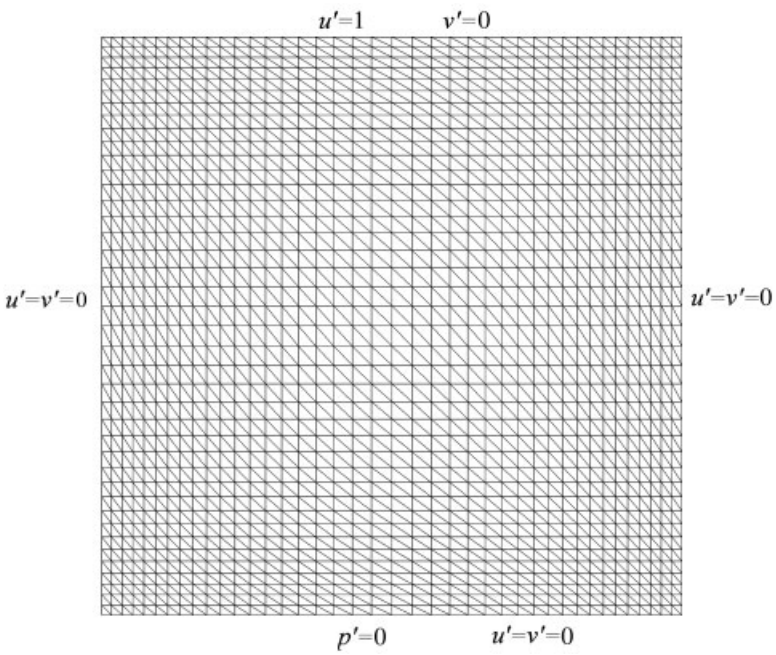

Figure 5. Mesh and boundary conditions for the lid-driven cavity problem.

advantageous, with the time required to reach the end of the simulation being 36.83% of the time using EBE routines with the same number of time steps.

5.2. *Lid-driven cavity*

In the lid-driven cavity flow problem, a fixed mesh containing 1681 nodes, 3200 elements and 4880 edges is used. Non-dimensional variables have been used here again. To the top of the domain, a square cavity with length $L'=1$, $u'=1$ and $v'=0$ velocity components have been applied, and along the other faces, null velocity components, $u'=v'=0$. Figure 5 shows mesh and boundary conditions for the problem. The simulation was carried out in the time interval $t'=0$–5, considering $Re=1000$.

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

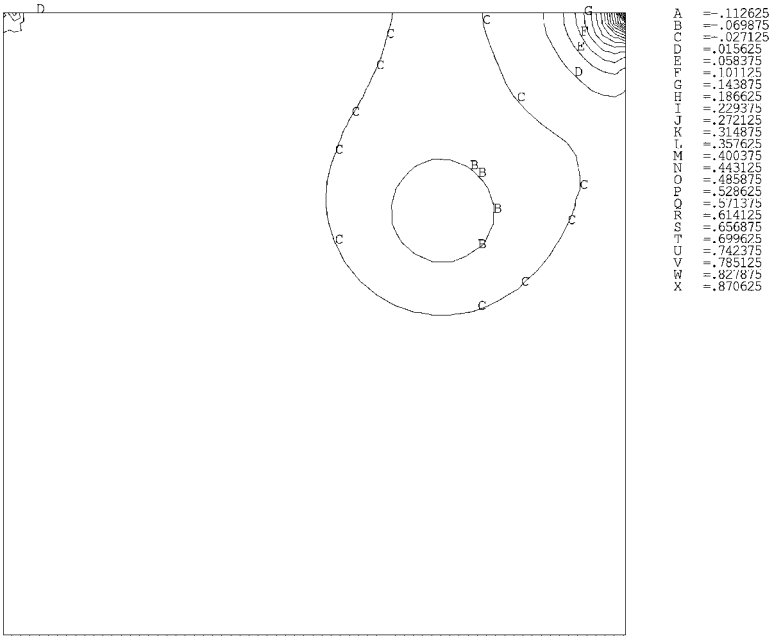

Figure 6. Pressure field at $t' = 5$.

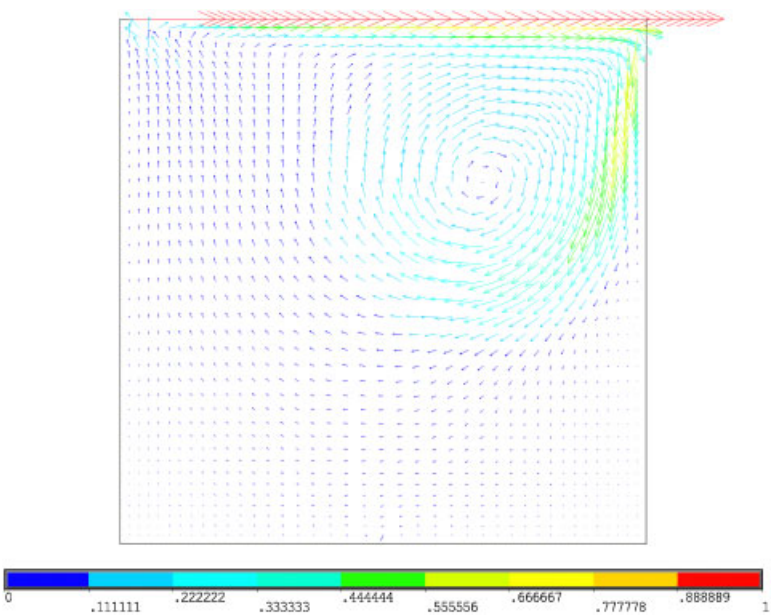

Figure 7. Velocity field at $t' = 5$.

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

Table IV. Element-by-element and edge-by-edge performance for the lid-driven cavity problem.

Data arrangement	Element-by-element	Edge-by-edge
Number of time steps	400	411
CPU time (s)	196.969	145.172
Memory demand (Mwords)	0.0672	0.0488
Floating point operations (Mflop)	0.240	0.156
I/A $(*10E+6)$	0.115	0.117

Figure 8. Mesh and boundary conditions for the backward facing step problem.

Table V. Element-by-element and edge-by-edge performance for the backward facing step problem—$Re=250$.

Data arrangement	Element-by-element	Edge-by-edge
Number of time steps	132	129
CPU time (s)	803.297	438.891
Memory demand (Mwords)	0.294	0.2144
Floating point operations (Mflop)	1.050	0.6861
I/A $(*10E+6)$	0.5040	0.5146

Figures 6 and 7 present pressure and velocity fields for $t'=5.0$. Given that at $t'=5.0$ results have reached stationary solution, it is possible to compare them to Reference [19] and conclude that they are in good agreement with results shown for the same problem at $Re=1000$.

Table IV shows a comparison between performances with EBE and EDS strategies.

Table IV allows us to see that memory demand with the EDS solution is 72.61% of the one with EBE and number of floating point operations is 65.00% of the one with EBE while indirect addressing operations are 1.74% bigger with EDS. It can be noted that EDS requires a larger number of time steps than the EBE solution. This is due to the fact that time steps are calculated differently in each arrangement. Again, the time required for the simulation is smaller with EDS, being 73.70% of the time with EBE.

5.3. Backward facing step

The third example shows the backward facing step problem. It has been used a fixed structured mesh with 7421 nodes and 14 000 elements, resulting in 21 440 edges. The simulation covered the time interval between $t'=0$ and 5. Reynolds number was assumed 250. Mesh and boundary conditions are shown in Figure 8. Table V shows a comparison between performances using EBE and EDS.

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

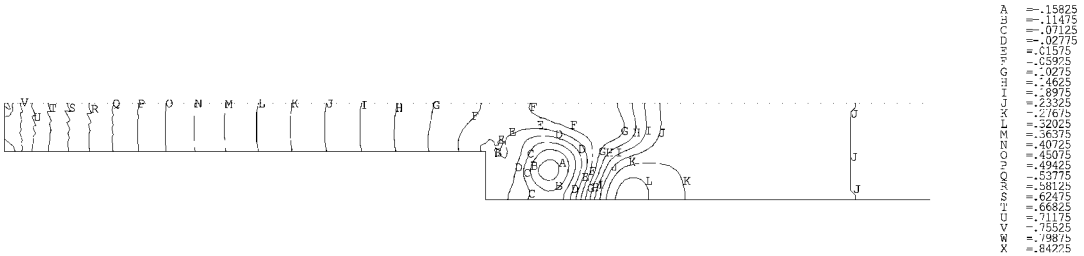

Figure 9. Pressure distribution at $t' = 5$.

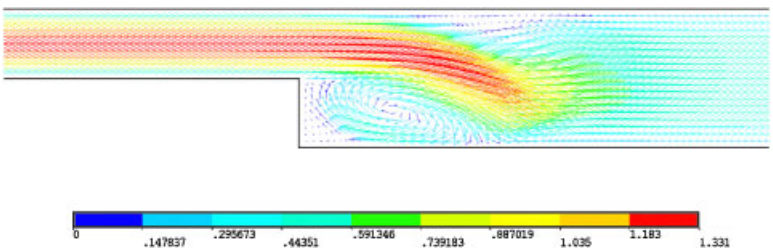

Figure 10. Detail for velocity field at $t' = 5$.

Figures 9 and 10 show pressure and velocity fields at the end of the simulation. Here again, $t' = 5$ represents a time when stationary results are obtained, allowing us to compare and conclude that the obtained results are comparable to the ones found in Reference [19], considering vortices positions and lengths.

In Table V we can see that memory demand with the EDS arrangement is 72.93% of the one with EBE. The number of floating point operations with EDS is 65.34% of the one with EBE and, again, the number of indirect addressing operations with EDS is larger than with EBE by 2.10%. For this problem, again because of the differences in the calculation of the time advance in each strategy, the number of time steps with EDS was smaller than with EBE. The time consumed to reach the end of the simulation with EDS was 54.64% of the one with EBE.

6. CONCLUDING REMARKS

Results show that the use of the De Sampaio–Coutinho formulation associated with an edge-based data arrangement provides an important gain in computing time for the finite element solution of the incompressible Navier–Stokes equations coupled with heat transfer. Results have been compared with the ones obtained with an element-based code.

Numerical comparisons show that results obtained with both data arrangements are virtually the same in all examples. Practically negligible differences (of the order of less than 5% in mixed and convection-forced problems, and 0% in free-convection problems) are observed due to the use of local time steps which are calculated for elements as shown in Section 3 and projected for edges as in Section 4. This procedure may cause the time step to be different for the same degree of freedom

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

in each of the data arrangements used. This is observed in both transient and stationary solutions. Time-accurate solutions with local time steps in transient solutions are obtained as demonstrated in Reference [20].

Though a larger number of indirect addressing operations have been performed with a reduced number of floating-point operations, when compared with the element-based scheme, the edge based arrangement shows still more advantages than the former, requiring less memory and a smaller number of floating point operations, as shown in the examples and predicted in Tables I and II.

Also, ratifying our choice concerning data arrangement, considering the analyses performed in References [16, 18], it is possible to conclude that the best strategy for data arrangement has been adopted to solve two-dimensional problems governed by the formulation here presented. We remark that better performances may be achieved if we consider the edge and node reordering techniques given in Reference [18].

The present formulation associated with edge-based data arrangements has great potential to be extended to 3D problems. Future implementations include mesh adaptation, use of special edge arrangements and adoption of a unique time step to advance solution [12], besides data locality improvements when suitable, as shown in References [16, 18].

ACKNOWLEDGEMENTS

This work is partially supported by CNPq grants 301251/2005-3 and 301045/2003-8.

REFERENCES

1. de Sampaio PAB. Petrov–Galerkin finite element formulations for incompressible viscous flows. *Ph.D. Thesis*, Department of Civil Engineering, University of Wales, Swansea, 1991.
2. Löhner R. *Applied CFD Techniques—An Introduction Based on Finite Element Methods*. Wiley: Chichester, 2001.
3. Barth TJ. Numerical aspects of computing viscous high Reynolds number flows on unstructured meshes. *29th Aerospace Sciences Meeting and Exhibit*, Reno. AIAA Paper 91-0721, 1991.
4. Luo H, Baum JD, Löhner R, Cabello J. Adaptive edge-based finite element schemes for the Euler and Navier–Stokes equations on unstructured grids. *31st Aerospace Sciences Meeting and Exhibit*, Reno. AIAA Paper 93-0336, 1993.
5. Peraire J, Peiro J, Morgan K. Multigrid Solution of the 3-D compressible Euler equations on unstructured tetrahedral grids. *International Journal for Numerical Methods in Engineering* 1993; **36**(6):1029–1044.
6. Venkatakrishnan V, Mavriplis DJ. Implicit solvers for unstructured meshes. *Journal of Computational Physics* 1993; **105**:83–91.
7. Löhner R. Edges, stars, superedges and chains. *Computer Methods in Applied Mechanics and Engineering* 1994; 255–263.
8. Catabriga L, Coutinho ALGA. Implicit SUPG solution of Euler equations using edge-based data structures. *Computer Methods in Applied Mechanics and Engineering* 2002; **191**(32):3477–3490.
9. Soto O, Löhner R, Cebral J, Camelli F. A stabilized edge-based implicit incompressible flow formulation. *Computer Methods in Applied Mechanics and Engineering* 2004; **60**:641–660.
10. de Sampaio PAB, Coutinho ALGA. Simulation of free and forced convection incompressible flows using an adaptive parallel/vector finite element procedure. *International Journal for Numerical Methods in Fluids* 1999; **29**:289–309.
11. de Sampaio PAB, Hallak PH, Coutinho ALGA, Pfeil MA. A stabilized finite element procedure for turbulent fluid–structure interaction using adaptive time-space refinement. *International Journal for Numerical Methods in Fluids* 2004; **44**:673–693.
12. de Sampaio PAB. A finite element formulation for transient incompressible viscous flows stabilized by local time-steps. *Computer Methods in Applied Mechanics and Engineering* 2005; **194**:2095–2108.

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

13. de Sampaio PAB. A stabilized finite element method for incompressible flow and heat transfer: a natural derivation based on the use of local time-steps. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**(44–47):6177–6190.
14. Hughes TJR, Franca LP, Hulbert GM. A new finite element formulation for computational fluid dynamics: VIII. The Galerkin least-squares method for advective–diffusive equations. *Computer Methods in Applied Mechanics and Engineering* 1989; **73**:173–189.
15. Franca LP, Frey SL. Stabilized finite element methods: II. The incompressible Navier–Stokes equations. *Computer Methods in Applied Mechanics and Engineering* 1992; **99**:209–233.
16. Ribeiro FLB, Coutinho ALGA. Comparison between element, edge and compressed storage schemes for iterative solutions in finite element analyses. *International Journal for Numerical Methods in Engineering* 2005; **63**: 569–588.
17. Brooks A, Hughes TJR. Streamline upwind/Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations. *Computer Methods in Applied Mechanics and Engineering* 1982; **32**:199–259.
18. Coutinho ALGA, Martins MAD, Sydenstricker RM, Elias RN. Performance comparison of data-reordering algorithms for sparse matrix-vector multiplication in edge-based unstructured grid computations. *International Journal for Numerical Methods in Engineering* 2006; **66**:431–460.
19. Griebel M, Dornseifer T, Neunhoeffer T. *Numerical Simulation in Fluid Dynamics—A Practical Introduction.* SIAM: Philadelphia, PA, 1997.
20. de Sampaio PAB. Transient solutions of the incompressible Navier–Stokes equations in primitive variables employing optimal local time stepping. In *Proceedings of the 8th International Conference on Numerical Methods in Laminar and Turbulent Flow*, Taylor C (ed.), vol. VIII. Pineridge Press: Swansea, 1993; 1493–1504.

Copyright © 2006 John Wiley & Sons, Ltd.

DOI: 10.1002/fld