Parallel Adaptive Simulation of Coupled Incompressible Viscous Flow and Advective-Diffusive Transport Using Stabilized FEM Formulation

André L. Rossa
Engineering Simulation and Scientific Software,
Rio de Janeiro, Brazil, 20210-031,
andre.rossa@esss.com.br

and

Alvaro L.G.A. Coutinho
High-Performance Computing Center, Department of Civil Engineering, UFRJ
Rio de Janeiro, Brazil, 21941-972
alvaro@nacad.ufrj.br

Abstract

In this work we study coupled incompressible viscous flow and advective-diffusive transport of a scalar. Both the Navier-Stokes and transport equations are solved using an Eulerian approach. The SUPG/PSPG stabilized finite element formulation is applied for the governing equations. The implementation is held using the libMEsh finite element library which provides support for parallel adaptive mesh refinement and coarsening. The Rayleigh-Bénard natural convection and the planar lock-exchange density current problems are solved to assess the adaptive parallel performance of the numerical solution.

Keywords: Stabilized FEM formulation, incompressible flows, adaptive meshes, parallel computing.

1 Introduction

The numerical simulation of current engineering problems would not be feasible without the advent of parallel computing. Even with the development of techniques for mesh adaptation, the fastest available processors are not able to solve, within a practical period of time, problems that have large amounts of degrees of freedom. High-performance computing (HPC) have enabled the solution of problems with a large number of unknowns and high complexity (often involving multiple scales and multiple physics) by clusters of computers installed in universities as well in industry research centers.

To make HPC be efficiently used, a set of algorithms and computational methods have been developed over the last decade that made possible high fidelity solutions of complex problems. Processors with multiple cores with shared memory, clusters of personal computers in which each processor has its own memory (distributed memory) and more recently, the hybrid memory architectures are present on the daily work of engineers and researchers.

Although improving processing capacity, parallel computation have added complexity to computer codes programing. According to [1] scaling performance is particularly problematic because the vision of seamless scalability cannot be achieved without having the applications scale automatically as the number of processors increases. However, for this to happen, the applications have to be programmed to exploit parallelism efficiently. Therefore, parallel computing resources should be used rationally in order to obtain compatible performances.

Good simulation practice suggests that the applications and algorithms employed in HPC should be optimized for this purpose. Currently there are available (mostly freely distributed) several programs to perform different tasks inherent to parallel computing. The domain partitioning and load balancing, information exchange between processors, algebraic operations and linear preconditioned systems solving are

some of the necessary operations and have specific computational libraries that can be incorporated into the implementation of a numerical simulator.

In order to keep the focus on the issues related to the numerical problem, we use the `libMesh` framework, which is a C++ library for parallel adaptive mesh refinement/coarsening numerical multiphysics simulations based on the finite element method [2]. The library has been developed since 2002 by a group of researchers from CFDLab, Department of Aerospace Engineering and Engineering Mechanics, University of Texas at Austin, and is available as open source software (`http://libmesh.sourceforge.net/`).

In this work we implement stabilized finite element formulations for the Navier-Stokes and advective-diffusive transport in `libMesh`. Parallel adaptive simulations of coupled problems confirm the mesh size reduction potential and the ability to capture the solution small scales as well the development of the interface between the fluids in evolution problems.

The remainder of this work is organized as follows. In the next section the dimensionless governing equations for the coupled viscous flow and transport is presented together with correspondent stabilized SUPG/PSPG finite element method (FEM) formulation. Details of the adaptive mesh refinement/coarsening (AMR/C) in the context of the `libMesh` library as well some aspects of the parallel solution of precondiotioned linear systems are presented in Section 3. Section 4 presents the results of the parallel adaptive simulation of the Rayleigh-Bénard natural convection and a density current in a planar lock-exchange configuration. The paper ends with the main conclusions.

2 Mathematical Formulation

2.1 Governing Equations

Assuming an unsteady incompressible viscous flow and the Bousinessq approximation, the dimensionless Navier-Stokes, continuity and scalar transport equations[1] can be written in a non conservative form following a Eulerian description as

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\nabla\mathbf{u} - \frac{1}{Re}\nabla^2\mathbf{u} + \nabla p = \frac{Gr}{Re^2}\phi\mathbf{e} \quad \text{in } \Omega \times [0,t] \ , \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega \times [0,t] \ , \tag{2}$$

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla\phi - \frac{1}{D}\nabla^2\phi = 0 \quad \text{in } \Omega \times [0,t] \ . \tag{3}$$

defined in the simulation domain Ω with a smooth boundary Γ. The time is t, $\mathbf{u} = (u, v, w)^T$ is the velocity field, p is the pressure and ϕ the scalar being transported and $\mathbf{e}$ is an unit vector aligned with the gravity.

In (1) Re and Gr are the Reynolds and Grashof numbers. The inverse of the parameter D in equation (3) represents a dimensionless diffusive constant depending on the nature of the scalar being transported (e.g., the Peclet number for the temperature transport).

The essential and natural boundaries conditions (BCs) are:

$$\begin{aligned} \mathbf{u} &= \mathbf{g} \quad && \text{on } \Gamma_{\mathbf{g}} \ , \\ \mathbf{n} \cdot \left[\frac{1}{Re}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T \right) - p\mathbf{I} \right] &= \mathbf{h} \quad && \text{on } \Gamma_{\boldsymbol{\sigma}} \ , \\ \phi &= \overline{\phi} \quad && \text{on } \Gamma_{\phi}, \\ -\mathbf{n} \cdot \left(\frac{1}{D}\nabla\phi \right) &= q \quad && \text{on } \Gamma_q \end{aligned} \tag{4}$$

where $\mathbf{g}$ and $\overline{\phi}$ are functions with prescribed values for the velocity vector and scalar defined on the regions $\Gamma_{\mathbf{g}}$ and Γ_{ϕ} of the boundary where the essential BCs are imposed. Furthermore $\mathbf{h}$ and q represent the natural BCs acting on the regions $\Gamma_{\boldsymbol{\sigma}}$ and Γ_q. Generally $\Gamma_i \subset \Gamma$. $\mathbf{n}$ is the unit outward normal vector on the boundary and $\mathbf{I}$ is the 3×3 identity matrix.

The initial conditions are:

$$\begin{aligned} \mathbf{u}(\mathbf{x}, 0) &= \mathbf{u_0} \ , \\ \phi(\mathbf{x}, 0) &= \phi_0 \end{aligned} \tag{5}$$

where the initial velocity field $\mathbf{u_0}$ is divergent free.

[1] Details on how the physical quantities may be normalized in order to arrive at dimensionless equations can be found at [3].

In gravity current problems, the concept of buoyancy velocity u_b is largely used (see [4]). It may be defined as

$$u_b := \sqrt{g' h_v} \tag{6}$$

where h_v is a scale length related to the vertical dimension of the simulation domain (usually taken as the domain height) and $g^{'}$ is called the reduced gravity given by

$$g^{'} := g\frac{\rho_1 - \rho_2}{\rho_\infty} \tag{7}$$

where g is the absolute value of the gravitational acceleration, ρ_∞ is the reference density, ρ_1 is the density of the "heavy" fluid and ρ_2 is the density of the "light" fluid.

When one uses the buoyancy velocity as a reference velocity and h_v as the scale length, the Reynolds number may be computed directly from the Grashof as follow

$$Re = \sqrt{Gr}\ . \tag{8}$$

For particle-driven problems (a class of gravity current phenomenon), where the transported scalar is the density ρ, the diffusivity constant is given by the product of the Schmidt Sc and Grashof numbers. Taking it into account, the Navier-Stokes and advective-diffusive equations may be rewritten as

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\nabla\mathbf{u} - \frac{1}{\sqrt{Gr}}\nabla^2\mathbf{u} + \nabla p = \rho\mathbf{e}\ , \tag{9}$$

$$\frac{\partial \rho}{\partial t} + \mathbf{u}\cdot\nabla\rho - \frac{1}{ScGr}\nabla^2\rho = 0\ . \tag{10}$$

2.2 Stabilized Finite Element Formulation

Given a suitably defined finite-dimensional trial solution and weight functions spaces for velocity and pressure

$$\begin{aligned} S^h_{\mathbf{u}} &= \left\{ \mathbf{u}^h \mid \mathbf{u}^h \in \left[H^{1h}(\Omega)\right]^3,\ \ \mathbf{u}^h \doteq \mathbf{g}^h \ \text{ em } \ \Gamma_{\mathbf{g}} \right\}\ , \\ V^h_{\mathbf{w}} &= \left\{ \mathbf{w}^h \mid \mathbf{w}^h \in \left[H^{1h}(\Omega)\right]^3,\ \ \mathbf{w}^h \doteq 0 \ \text{ em } \ \Gamma_{\mathbf{g}} \right\}\ , \\ S^h_p &= V^h_p = \left\{ q^h \mid q^h \in H^{1h}(\Omega) \right\} \end{aligned} \tag{11}$$

where $H^{1h}(\Omega)$ is the finite-dimensional space function square integrable into the element domain, the stabilized SUPG/PSPG FEM formulation for the non-dimensional Navier-Stokes and continuity equations (1) and (2) can be written as: Find $\mathbf{u}^h \in S^h_{\mathbf{u}}$ and $p^h \in S^h_p$ such as, $\forall \mathbf{w}^h \in V^h_{\mathbf{w}}$ and $\forall q^h \in V^h_p$,

$$\begin{aligned} &\int_\Omega \mathbf{w}^h \cdot \left[\left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h\nabla\mathbf{u}^h \right) - \mathbf{l}^h \right] d\Omega + \frac{1}{Re}\int_\Omega \left(\nabla\mathbf{w}^h\right)^T \cdot \nabla\mathbf{u}^h \mathbf{I} d\Omega - \\ &\int_\Omega \nabla\mathbf{w}^h p^h \mathbf{I} d\Omega - \int_\Gamma \mathbf{w}^h \cdot \mathbf{h}^h d\Gamma + \int_\Omega q^h \nabla\cdot\mathbf{u}^h d\Omega + \\ &\sum_{e=1}^{n_el} \int_{\Omega^e} \left(\tau_{SUPG}\mathbf{u}^h\nabla\mathbf{w}^h\right) \cdot \left[\left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h\nabla\mathbf{u}^h \right) + \nabla p^h - \mathbf{l}^h \right] d\Omega^e + \\ &\sum_{e=1}^{n_el} \int_{\Omega^e} \left(\tau_{PSPG}\nabla q^h\right) \cdot \left[\left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h\nabla\mathbf{u}^h \right) + \nabla p^h - \mathbf{l}^h \right] d\Omega^e = 0\ . \end{aligned} \tag{12}$$

The first four integrals in (12) arise from the classical Galerkin weak formulation for the Navier-Stokes equations. The fifth integral represents the classical Galerkin formulation for the continuity equation. The summations over the elements are the SUPG and the PSPG stabilizations for the Navier-Stokes equation. The parameters adopted for both stabilizations were obtained from [5] and are defined as follows

$$\tau_{SUPG} = \tau_{PSPG} = \left[\left(2\frac{\left\|\mathbf{u}^h\right\|}{\mathfrak{h}} \right)^2 + 9\left(\frac{4}{Re\mathfrak{h}^2} \right)^2 \right]^{-\frac{1}{2}}\ . \tag{13}$$

The dimensionless stabilizations parameters are local (element level) so, the velocity modulus $\left\|\mathbf{u}^h\right\|$ is calculated for each element e and $\mathfrak{h}$ is an element length measure based in its volume $\mathfrak{V}$ as shown bellow

$$\mathfrak{h} = \sqrt[3]{\frac{6\mathfrak{V}}{\pi}} \, . \tag{14}$$

The discretized dimensionless body force are represented by $\mathbf{l}^h$.

For the dimensionless advective-diffusive transport we adopt the same assumptions, so given the following finite-dimensional trial solution and weight functions spaces for the scalar

$$\begin{aligned} S_\phi^h &= \left\{ \phi^h \mid \phi^h \in H^{1h}(\Omega), \;\; \phi^h \doteq \overline{\phi}^h \;\; \text{in} \;\; \Gamma_\phi \right\} , \\ V_w^h &= \left\{ w^h \mid w^h \in H^{1h}(\Omega), \;\; w^h \doteq 0 \;\; \text{em} \;\; \Gamma_\phi \right\} \end{aligned} \tag{15}$$

the stabilized FEM formulation can be written as: Find $\phi^h \in S_\phi^h$ such as, $\forall w^h \in V_w^h$,

$$\begin{aligned} &\int_\Omega w^h \cdot \left(\frac{\partial \phi^h}{\partial t} + \mathbf{u}^h \cdot \nabla \phi^h \right) d\Omega + \frac{1}{D} \int_\Omega \left(\nabla w^h \right)^T \cdot \nabla \phi^h d\Omega - \int_\Gamma w^h q d\Gamma + \\ &\sum_{e=1}^{n_e l} \int_{\Omega^e} \left(\tau_{SUPG} \mathbf{u}^h \cdot \nabla w^h \right) \cdot \left[\left(\frac{\partial \phi^h}{\partial t} + \mathbf{u}^h \cdot \nabla \phi^h \right) \right] d\Omega^e = 0 \, . \end{aligned} \tag{16}$$

The three first integrals in (16) come from the Galerkin weak formulation. The integral into the summation over the elements is the SUPG stabilization. The non-dimensional stabilization parameter is computed similarly to (13), that is:

$$\tau_{SUPG} = \left[\left(2 \frac{\left\| \mathbf{u}^h \right\|}{\mathfrak{h}} \right)^2 + 9 \left(\frac{4}{D \mathfrak{h}^2} \right)^2 \right]^{-\frac{1}{2}} \, . \tag{17}$$

In the stabilized formulations (16), an additional stabilization is added to handle instabilities in the numerical solution of flows with presence of strong gradients of the scalar being transported. In [6] is presented a discontinuity capturing term which is calculated as follows:

$$\sum_{e=1}^{n_e l} \int_{\Omega^e} \delta \left(\phi^h \right) \nabla w^h \cdot \nabla \phi^h d\Omega^e \, . \tag{18}$$

Because the δ parameter is a function of the scalar, (16) can be understood as a nonlinear diffusion operator. In this work, the δ parameter was adapted from [6] as follows in dimensionless form

$$\delta \left(\phi^h \right) = \left| \frac{1}{\phi^*} R \left(\phi^h \right) \right| \left(\sum_{i=1}^{3} \left| \frac{1}{\phi^*} \frac{\partial \phi^h}{\partial x_i} \right|^2 \right)^{\beta/2-1} \frac{\mathfrak{h}}{2}^\beta \tag{19}$$

where ϕ^* is a dimensionless value of the scalar (usually taken as 1) and $R\left(\phi^h\right)$ is an approximation for the actual residual defined as:

$$R \left(\phi^h \right) = \frac{\partial \phi^h}{\partial t} + \mathbf{u}^h \cdot \nabla \phi^h \, . \tag{20}$$

The β parameter can be set as 1 or 2.

2.3 Discretized Systems

Adopting the implicit backward Euler scheme for the time discretization together with a fixed point linearization, the final discrete system of (12) and (16) results in

$$\begin{aligned} &\left(\mathbf{M} + \mathbf{M}_\tau \right) \mathbf{u}^{n+1,k+1} + \Delta t \left(\mathbf{N} \left(\mathbf{u}^{n+1,k} \right) + \mathbf{N}_\tau \left(\mathbf{u}^{n+1,k} \right) + \mathbf{K} \right) \mathbf{u}^{n+1,k+1} - \\ &\Delta t \left(\mathbf{G} - \mathbf{G}_\tau \right) \mathbf{p}^{n+1,k+1} = \Delta t \left(\mathbf{f} \left(\boldsymbol{\phi}^n \right) + \mathbf{f}_\tau \left(\boldsymbol{\phi}^n \right) \right) + \left(\mathbf{M} + \mathbf{M}_\tau \right) \mathbf{u}^n \, , \end{aligned} \tag{21}$$

$$\begin{aligned} &\Delta t \mathbf{G}^T \mathbf{u}^{n+1,k+1} + \mathbf{M}_\xi \mathbf{u}^{n+1,k+1} + \Delta t \left(\mathbf{N}_\xi \left(\mathbf{u}^{n+1,k} \right) \mathbf{u}^{n+1,k+1} + \mathbf{G}_\xi \mathbf{p}^{n+1,k+1} \right) = \\ &\Delta t \mathbf{f}_\xi \left(\boldsymbol{\phi}^n \right) + \mathbf{M}_\xi \mathbf{u}^n \, , \end{aligned} \tag{22}$$

$$
\begin{aligned}
&\left(\mathbf{M}+\mathbf{M}_{\tau}\right)\boldsymbol{\phi}^{n+1,k+1}+\\
&\Delta t\left(\mathbf{N}\left(\mathbf{u}^{n+1}\right)+\mathbf{N}_{\tau}\left(\mathbf{u}^{n+1}\right)+\mathbf{K}+\mathbf{K}_{\delta}\left(\boldsymbol{\phi}^{n+1,k}\right)\right)\boldsymbol{\phi}^{n+1,k+1}=\\
&\left(\mathbf{M}+\mathbf{M}_{\tau}\right)\boldsymbol{\phi}^{n}\ .
\end{aligned}
\tag{23}
$$

In the matrix systems (21), (22) and (23) $\mathbf{u}$, $\mathbf{p}$ and $\boldsymbol{\phi}$ are the nodal vectors of the correspondent unknowns $\mathbf{u}^h$, $\mathbf{p}^h$ and $\boldsymbol{\phi}^h$, and Δt stands for the time-step size. The super indexes $n+1$ and n mean the current and previous time-steps while $k+1$ and k are respectively the current and previous nonlinear iterations counter.

For the matrices where the advective operator appears, i.e., Galerkin advection, SUPG mass and advection, and PSPG advection, the velocity components are evaluated at each integration point. $\mathbf{M}$ is the mass matrix, $\mathbf{K}$ is the viscous/diffusive matrix, $\mathbf{N}(\mathbf{u})$ is the nonlinear advection matrix, $\mathbf{G}$ and $\mathbf{G}^T$ are the gradient and its transpose (divergent) matrices. $\mathbf{f}$ is the body force vector. $\mathbf{K}_{\delta}(\boldsymbol{\phi})$ is the nonlinear discontinuity capturing matrix. The matrices and vectors with the subscripts τ and ξ mean the SUPG and PSPG terms.

3 Computational Aspects

3.1 Mesh Adaptivity

The mesh adaptivity together with high-performance computing (parallel processing) play a key role to enable numerical simulations of actual engineering/industrial problems within an acceptable time without exhausting the processing capacity of current computers.

Particularly for the density current problem, AMR/C is a important tool to capture and track the flow structure at the front, where the Kelvin-Helmholtz billows occur. From an initial coarse mesh, the adaptivity refinement process begins near the interface between the two fluids and it follows its development.

In `libMesh` the mesh refinement can be accomplished by element subdivision (h-refinement), increasing the local polynomial degree (p-refinement) as well a combination of both methods (hp-refinement). Although there is an extensive literature devoted to obtaining reliable a posteriori estimators that are more closely linked to the operators and governing equations [7, 8], in `libMesh` the error indicator is focused on local indicators that are essentially independent of the physics [2].

`libMesh` uses a statistical refinement/coarsening scheme based on the ideas presented in [9] where the mean μ and the standard deviation σ of the error indicator "population" are computed. Using refinement and coarsening fractions (r_f and r_c), the elements are flagged for refinement and coarsening as showed in Fig. (1).

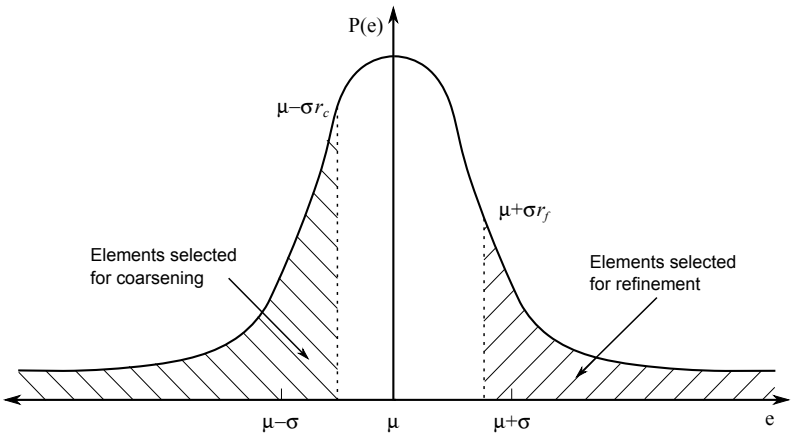


Figure 1: Statistical refinement: elements in hatched areas are flagged to AMR/C process

This scheme is suitable for evolution problems where, in the beginning, a small error is evenly distributed. Throughout the simulation the error distribution spreads and the AMR/C process starts. Whether the solution approaches its steady-state, the distribution of error also reaches the steady-state, stopping the AMR/C process.

The elements are refined through a "natural refinement" scheme: elements of dimension d, with the exception of the pyramids, produce 2^d elements of the same type after refinement. The degrees of freedom are constrained at the hanging nodes on element interfaces. This approach yields a tree data structure

formed by the "parents" and their "children" elements. An example of an element level hierarchy resulting from the application of this scheme for a hexahedral mesh is presented in Fig. 2.

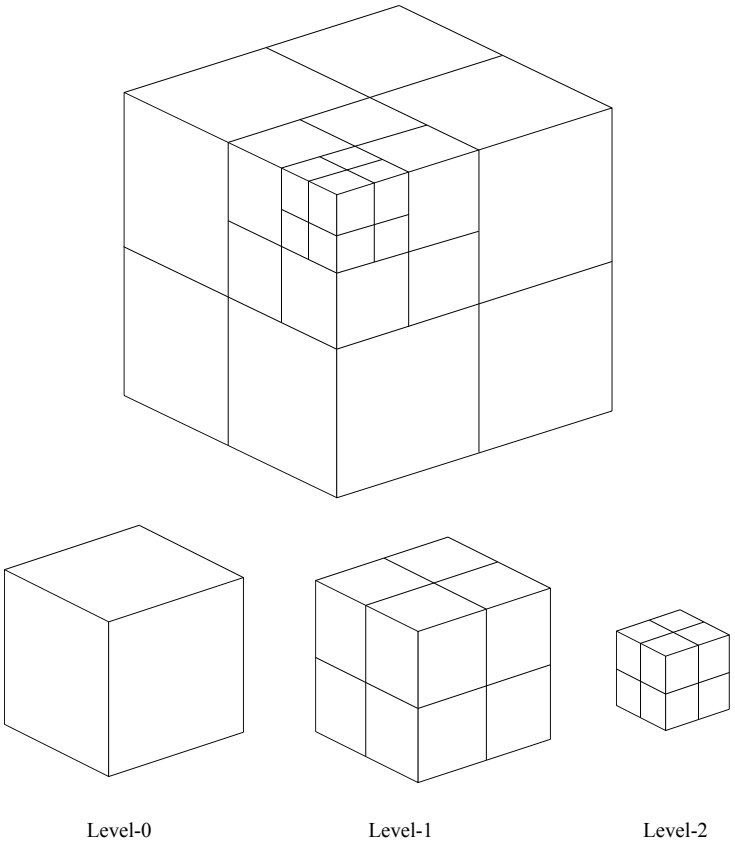


Figure 2: Hierarchy levels of the refinement of an hexahedral element

The elements present in the initial mesh (level-0 elements) have no parents as well the active elements (those that are part of the current simulation mesh) have no children, so the latter are the current high-level elements. The element level is determined recursively from its parents and the user should determine the maximum refinement level.

The frequency of refinement/coarsening is an user's responsibility. When the mesh adapts it is optimized for the state at the current time [10]. One should find a frequency of adaptivity which will balance the computational effort and quality of results as there is a computational cost associated with the adaptivity process. When the mesh is adapted, the field solution should be projected (interpolated) and a simulation should be performed on the new mesh.

In this work we use a class of errors estimators based on derivative jump (or flux jump) of the transported scalar calculated at the elements interface called Kelly's Error Estimator [11] to perform the h-refinement. The refinement and coarsening fractions for the statistical strategy as well the adaptivity frequency are set independently for each simulation problem.

3.2 Domain Decomposition

In this paper, we consider the standard partitioning domain without overlap, as shown in Fig. (3), where the elements related to each of the sub-domains are assigned to different processors. That is, the simulation domain Ω^h is divided into a discrete set of sub-domains Ω_p^h such as $\bigcup \Omega_p^h = \Omega^h$ and $\bigcap \Omega_p^h = \emptyset$.

In AMR/C computations at a new adaptation stage, regions of the domain will have an increase in mesh element density while in others, the number of elements will decrease. These dynamic mesh adjustments result for some processors in significant increasing (or decreasing) work therefore causing load unbalancing [1].

Libraries such as METIS and ParMETIS were developed aiming implementing efficient partitioning mesh schemes. The first is a serial mesh partitioning library, while the second is based on parallel MPI. Both can also reorder the unknowns in unstructured grids to minimize the fill-in during LU factorization. The ParMETIS extends the functionality provided by METIS and includes routines for parallel computations with adaptive meshes refinement and large-scale numerical simulations.

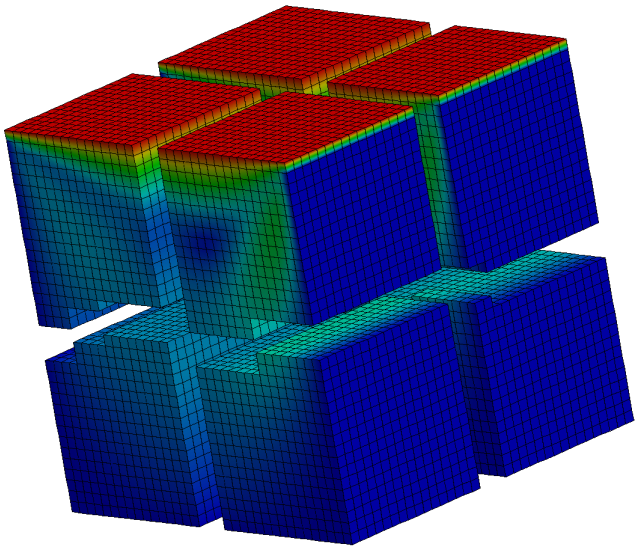


Figure 3: Simulation domain decomposition in 8 sub-domains

3.3 Parallel Solution of Preconditioned Linear Systems

The support for the numerical solution of the resulting system of linear equations in the parallel architecture environment is provided by PETSc. It provides structures for efficient storage (vectors, arrays, for example), as well ways to handle it. The `libMesh` uses compressed sparse row (CSR) data structure to store the sparse matrices. PETSc has a number of methods for solving linear sparse system as GMRES and BiConjugate Gradient method (BiCG) and several types of preconditioners as ILU(k) and Block-Jacobi. Options for reordering the linear system as the Reverse Cuthill-McKee method (RCM, [12]) are also available.

In this work we use Block-Jacobi sub-domain preconditioning. It is one of the most widely used schemes due to its easiness of implementation. There are no overlap between the blocks. The incomplete factorization may be applied to each of them without extra communication costs. However, in adaptive simulations, the new local factorization should be performed every time the mesh is modified since the adaptivity changes the group of elements residing in each sub-domain [13]. It is usual to refer to the Block-Jabobi preconditioning strategy in conjunction with the ILU factorization with a certain level k of fill-in by BILU(k) [14].

The communication between the processors, such as required for the algebraic operations or during assembly of arrays of elements are supported by a set of PETSc library routines which is designed for parallel computing using the MPI API.

4 Numerical Results

The simulations were performed in a SGI Altix ICE 8400 cluster with 640 cores (Intel Nehalem). This machine has 1.28 TB of distributed memory. The processing nodes are connected by InfiniBand. The cluster is located at the High-Performance Computing Center (NACAD) of the Federal University of Rio de Janeiro, Brazil.

4.1 Parallel Adaptive Simulation of the Rayleigh-Bénard Problem

In this example we consider the Rayleigh-Bénard natural convection in a container with geometric domain $\Omega = [0, 4] \times [0, 1] \times [0, 1]$. This problem consists to solve a natural convection phenomenon of a fluid which initially at rest ($t = 0$) produces a sequence of adjacent convection cells along the longitudinal direction (x axis) due to the temperature difference between its upper (cold) and lower (hot) walls.

No-slip boundary conditions are imposed in all the walls and the pressure is prescribed as $p(2.0, 0.5, 0.0) = 0.0$. The dimensionless cold temperature is $T_c = -0.5$ and the hot $T_h = 0.5$. The physical problem is defined setting the Reynolds Number as $Re = 4,365$, Grashof number as $Gr = 41,666.66$, the Peclet number as $Pe = 3,142.8$ and Froude number[2] as $Fr = 0.6432$.

[2]Besides not introduced in the Navier-Stokes equations presented in section 2.1, the Froude number is used here to take into account the fluid's weight in the calculation. More details about how to incorporate the Froude number in the dimensionless Navier-Stokes equations may be found in [3].

Only one refinement/coarsening level was allowed at every 25 time-steps. For the statistical adaptivity scheme the refinement fraction is $r_f = 0.6$ and coarsening fraction is $r_c = 0.01$. The linear tolerance for GMRES(30) together with the BILU(1) and reordering by RCM method is 1.0×10^{-6}. The nonlinear tolerance is 1.0×10^{-5} and the constant time step size is $\Delta t = 5.0$.

The steady-state velocity vectors are shown in Fig.(4) and the temperature over the final adapted mesh is plotted in Fig.(5).

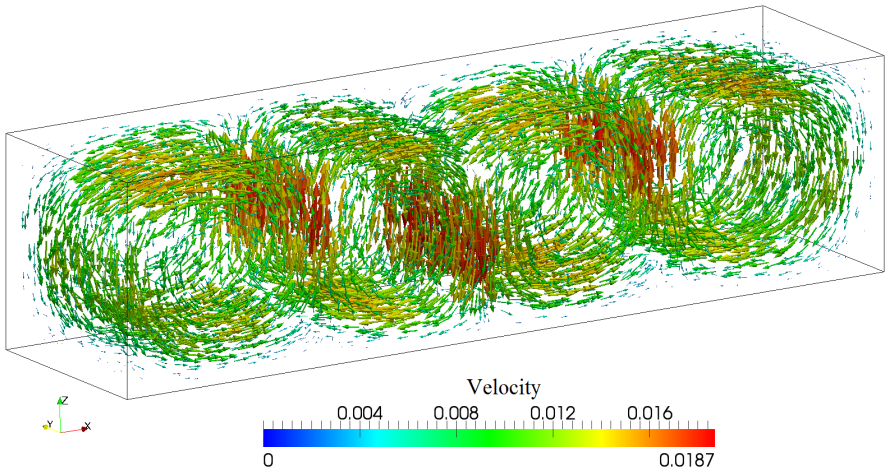


Figure 4: Steady-state velocity vectors

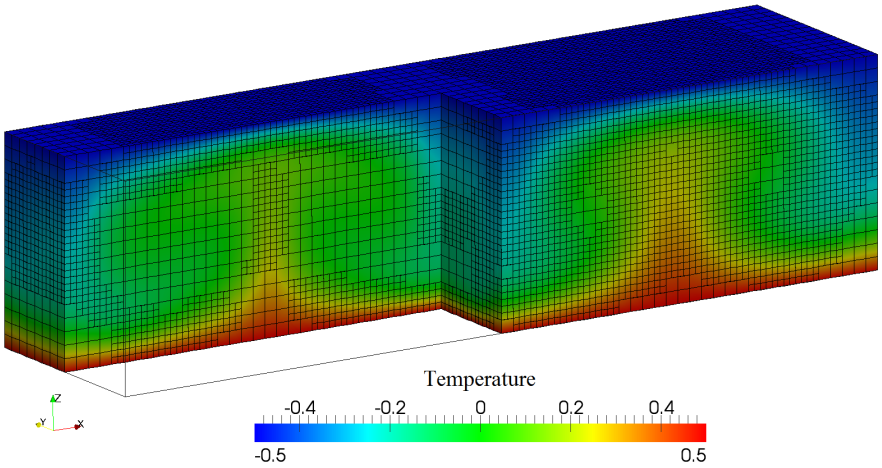


Figure 5: Temperature at steady-state and final adapted mesh

The Fig. (6) presents the speedup for the total simulation time, the total time for solving the Navier-Stokes and transport problems and the AMR/C procedure considering in the numerator the time spent with 16 CPU's, i.e.,

$$S_p = \frac{\tau_{16}}{\tau_p} \ . \tag{24}$$

The AMR/C time does not reach 10% of the total simulation time. We may observe from the results of Fig.(6) that the present simulation achieves a good parallel performance, that is, speed up around 3 for the total simulation with 64-cores run with respect to 16-cores run (over 3 for the Navier-Stokes simulation).

Despite the good overall performance, it is observed that the adaptive procedure does not scale as well as the linear solvers ($S_{64} = 1.22$). The poor performance of the AMR/C procedure in the current `libMesh` release is due to the fact that all mesh data are replicated on all cores, which increases memory requirements and communication per core.

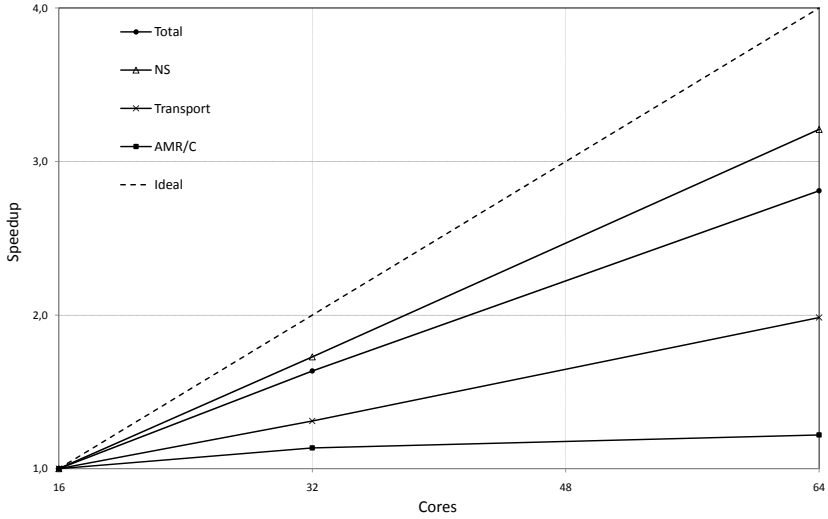
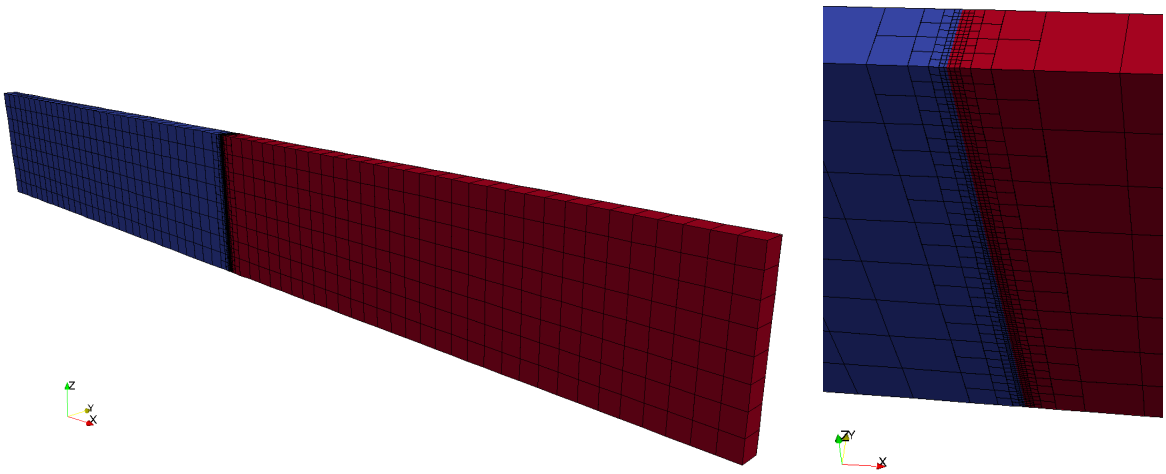


Figure 6: Speedup for the Rayleigh-Bénard problem

4.2 Temperature-driven gravity current with AMR/C

For the simulation of a temperature-driven gravity current with mesh adaptivity, we consider a slice domain[3] $\Omega = [0, L] \times [0, H/8] \times [0, H]$ where the nondimensional length and height are $L = 0.8$ and $H = 0.1$. The left half of the channel is initially filled with the cold fluid and the right half is filled with hot fluid. The Fig. (7) shows the initial configuration and a detail of the refinement at the center of the domain.



(a) (b)

Figure 7: Initial lock-exchange configuration: (a) View of the slice domain and (b) Mesh detail

The dimensionless cold temperature is set to $T_c = -0.5$ and the hot $T_h = 0.5$. We consider no-slip boundary conditions on the bottom, left and right walls. Free-slip boundary conditions are imposed on the top one. Thus free and no slip fronts may be considered in the same simulation.

We do not consider the reduced gravity for the definitions for the dimensionless parameters and set the Reynolds number as $Re = 1.0 \times 10^6$ and the Grashof is set to $Gr = 1.0 \times 10^{10}$. For this simulation, we disregard the diffusion term of the transport equation (3). So, the Peclet number does not need to be defined. We set the exponent of the nonlinear diffusion operator (19) as $\beta = 1$ and the time step is set to $\Delta t = 0.025$.

We compare the results from the present adaptive simulation with those obtained using fixed structured mesh with characteristic length $l = 0.00078125$ (given by the hexahedron edge). The dimensionless distance

[3]To emulate a 2D simulation domain from a mesh composed of 3D elements, the slice domain is positioned parallel to the xz plane and the perpendicular direction $(0, y, 0)$ is discretized with only one element except at regions where the mesh adapts. For all nodes on the mesh $v_y = 0.0$ is imposed.

of the front head between the two fluids $X = |x - 0.4|$ is tracked over time t. The distance from the initial position ($x = 0.4$) of both fronts using the fixed mesh is plotted in Fig. (8)

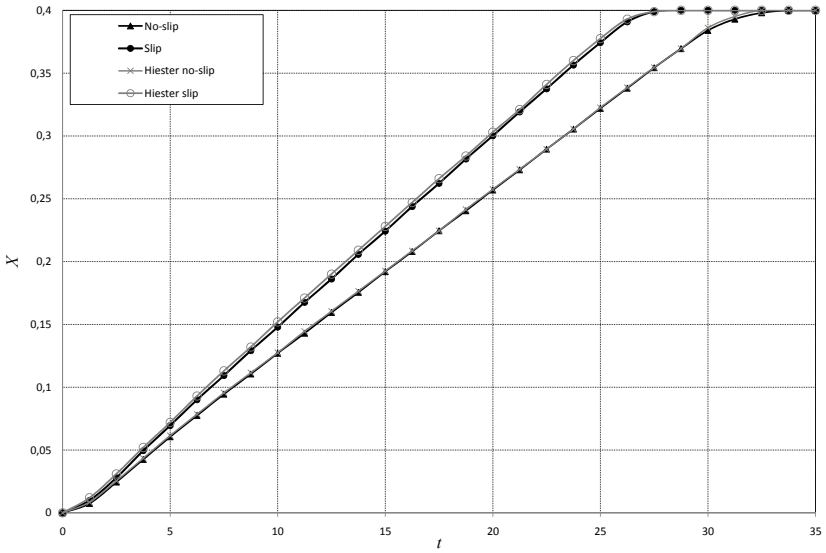
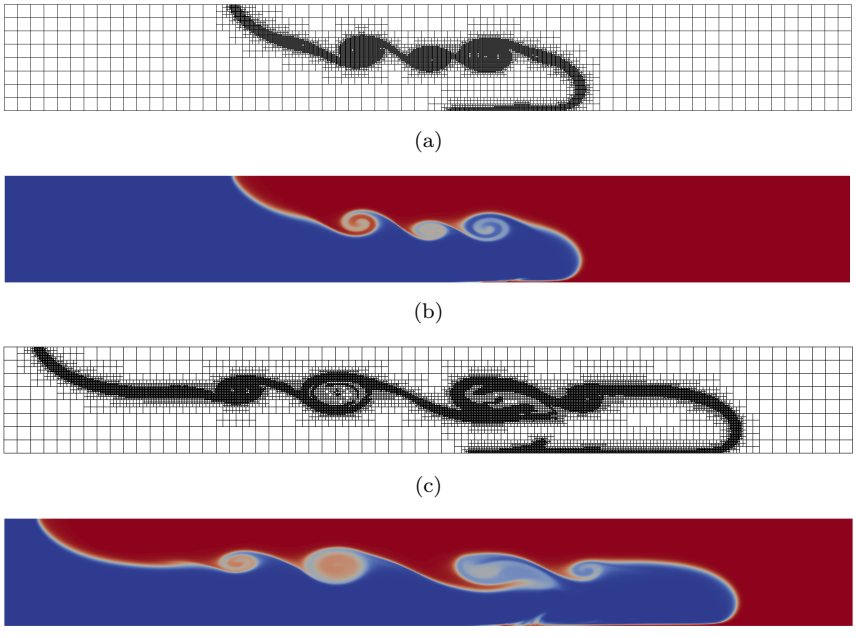


Figure 8: Plot of the dimensionless distance of top and bottom fronts

As expected, free-slip front (top) reaches the vertical wall before the no-slip (bottom) does. Figure (8) shows a good agreement between our results and those obtained by the reference [10]. The results from [10] were obtained using a fixed 2D mesh formed by triangles which characteristic length is $l = 0.00025$.

The Fig. (9) shows the temperature distributions and the meshes at two different times with AMR/C at every 10 time-steps. The refinement fraction is set as $r_f = 0.95$ and the coarsening fraction is $r_c = 0.01$. In order to prevent the size of the elements become too small, we allowed only 4 refinement-levels. The Kelvin-Helmholtz billows are captured by the mesh as the front evolves after the release.



(a) (b) (c) (d)

Figure 9: Adaptive meshes and temperature distribution: (a) Adaptive mesh at $t = 12.5$, (b) Temperature distribution at $t = 12.5$, (c) Adaptive mesh at $t = 25.0$ and (d) Temperature distribution at $t = 25.0$

Through the mesh adaptivity simulation, the largest number of elements reached is approximately 30,000. If a fixed structured mesh had been used, to achieve the same refinement level, it would take approximately

131,000 hexaedrons. Therefore, with mesh adaptation we can, in this problem, compute a solution with one order of magnitude less elements without compromising the solution accuracy.

5 Conclusions

The `libMesh` framework was used to implement the stabilized SUPG/PSPG finite element formulation for the parallel adaptive solution of incompressible viscous flow and advective-diffusive transport using a trilinear hexahedral element. Good numerical results were obtained for parallel executions with adaptive meshes. AMR/C allows the representation of multiple flow scales and improves resolution where needed. It was possible to track the Kelvin-Helmholtz billows present in the temperature-driven gravity current problem.

Acknowledgment

This work is partially supported by CNPq and ANP. We are also indebted to Engineering Simulation and Scientific Software (ESSS) to their partial support to A. Rossa in his PhD studies at COPPE/UFRJ. Computer time for the simulations in this work was provided by the High Performance Computing Center at COPPE/UFRJ.

References

[1] J. Dongarra, I. Foster, G. Fox, W. Gropp, K. Kennedy, L. Torczon, and A. White, *Sourcebook of Parallel Computing*. San Francisco, CA, USA: Morgan Kaufmann Publishers, 2003.

[2] B. S. Kirk, J. W. Peterson, R. H. Stogner, and G. F. Carey, "`libMesh`: A C++ Library for Parallel Adaptive Mesh Refinement/Coarsening Simulations," *Engineering with Computers*, vol. 22, no. 3–4, pp. 237–254, Nov. 2006, http://dx.doi.org/10.1007/s00366-006-0049-3.

[3] M. Griebel, T. Dornseifer, and T. Neuhoeffer, *Numerical Simulation in Fluid Dynamics: A Practical Introduction*. Philadelphia, PA, USA: SIAM, 1997.

[4] C. Härtel, E. Meiburg, and F. Necker, "Analysis and direct numerical simulation of the flow at a gravity-current head. Part 1. Flow topology and front speed for slip and no-slip boundaries," *Journal of Fluid Mechanics*, vol. 418, pp. 189–212, Apr. 2000.

[5] T. Tezduyar, "Stabilized finite element formulations for incompressible flow computations," ser. Advances in Applied Mechanics, J. W. Hutchinson and T. Y. Wu, Eds. Elsevier, 1991, vol. 28, pp. 1 – 44. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0065215608701534

[6] Y. Bazilevs, V. Calo, T. Tezduyar, and T. Hughes, "$YZ\beta$ discontinuity capturing for advection-dominated processes with application to arterial drug delivery," *International Journal for Numerical Methods in Fluids*, vol. 54, pp. 593–608, 2007.

[7] R. E. Bank and B. D. Welfert, "A posteriori error estimates for the Stokes problem," *Journal on Numerical Analysis*, vol. 28, pp. 591–623, Jun. 1991.

[8] M. Ainsworth and J. Oden, *A Posteriori Error Estimation in Finite Element Analysis*, ser. Pure and Applied Mathematics. Wiley, 2000. [Online]. Available: http://books.google.com.br/books?id=_HligpOKcjoC

[9] G. Carey, *Computational Grids: Generations, Adaptation & Solution Strategies*, ser. Series in Computational and Physical Processes in Mechanics and Thermal sciEnces. Taylor & Francis, 1997. [Online]. Available: http://books.google.com.br/books?id=0w4iXh4FXKYC

[10] H. Hiester, M. Piggott, and P. Allison, "The impact of mesh adaptivity on the gravity current front speed in a two-dimensional lock-exchange," *Ocean Modelling*, vol. 38, no. 1 - 2, pp. 1 – 21, Jan. 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1463500311000060

[11] D. W. Kelly, J. P. De S. R. Gago, O. C. Zienkiewicz, and I. Babuska, "A posteriori error analysis and adaptive processes in the finite element method: Part I - Error analysis," *International Journal for Numerical Methods in Engineering*, vol. 19, no. 11, pp. 1593–1619, Nov. 1983. [Online]. Available: http://dx.doi.org/10.1002/nme.1620191103

[12] W. Liu and A. Sherman, "Comparative analysis of the Cuthill-McKee and the reverse Cuthill-McKee ordering algorithms for sparse matrices," *Journal on Numerical Analysis*, vol. 13, no. 2, pp. 198–213, Apr. 1976.

[13] J. Camata, A. Rossa, A. Valli, L. Catabriga, G. Carey, and A. Coutinho, "Reordering and incomplete preconditioning in serial and parallel adaptive mesh refinement and coarsening flow solutions," *International Journal for Numerical Methods in Fluids*, vol. 69, no. 4, pp. 802–823, Jun. 2012. [Online]. Available: http://dx.doi.org/10.1002/fld.2614

[14] M. Benzi, "Preconditioning techniques for large linear systems: A survey," *Journal of Computational Physics*, vol. 182, no. 2, pp. 418–477, Nov. 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021999102971767