Towards large-scale three-dimensional blood flow simulations in realistic geometries

Céline Caldini Queiros, Vincent Chabannes, Mourad Ismail, Gonçalo Pena, Christophe Prud'Homme, Marcela Szopos, Ranine Tarabay

▶ **To cite this version:**

Céline Caldini Queiros, Vincent Chabannes, Mourad Ismail, Gonçalo Pena, Christophe Prud'Homme, et al.. Towards large-scale three-dimensional blood flow simulations in realistic geometries. ESAIM: Proceedings, 2013, 43, pp.195-212. 10.1051/proc/201343013 . hal-00786556

HAL Id: hal-00786556
https://hal.science/hal-00786556

Submitted on 11 Feb 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ESAIM: PROCEEDINGS, Vol. ?, 2013, 1-10
Editors: Will be set by the publisher

TOWARDS LARGE-SCALE THREE-DIMENSIONAL BLOOD FLOW SIMULATIONS IN REALISTIC GEOMETRIES

CÉLINE CALDINI-QUEIROS[1], VINCENT CHABANNES[2], MOURAD ISMAIL[3], GONCALO PENA[4], CHRISTOPHE PRUD'HOMME[5], MARCELA SZOPOS[5] AND RANINE TARABAY[5]

Abstract. This paper addresses the numerical approximation of fluid dynamics problems using various finite element methods including high order methods and high order geometry. The paper is divided in three parts. The first part concerns the various problem formulations and discretization methods we are interested in. Using the Stokes equations as model, several different types of boundary conditions are presented and discussed. The second part deals with describing the high performance framework FEEL++ with which we obtained the various numerical results including scalability studies. Finally we display numerical results: we start with convergence properties of the various formulations and associated discretization choices including high order geometries and we finish with a Navier-Stokes simulation within the cerebral venous system.

1. INTRODUCTION

Developing mathematical and numerical models for simulation of biofluids has been an active research field in the last few years (see, for instance [10] for a sound and up-to-date monograph on cardiovascular blood flow simulations or [13] for a recent synthesis on air flow simulations in the human lung, and the references therein). Intensive efforts were motivated by the fact that these phenomena are very complex, involving different scales (spatially and temporally), and combining multi-physics. The long term goal is to develop a framework able to provide information difficult or even impossible to obtain *in vivo* on patients and thus to support physicians in the diagnosis and/or treatment of diseases.

In the context of such applications, several difficult questions need to be addressed: the generation of meshes (for complex geometries), which is currently a challenge; the accurate simulation of blood flow, with a model best suited to the problem and an appropriate choice of boundary conditions; the validation of this model, allowing for calibration and uncertainty quantification. Moreover, since nowadays, recent developments of models, algorithms and computational environment make possible to use medical images (CT scan, MRI) as an input in numerical computations, a flexible framework is needed to include all these data. In this context, not only numerical models must account for standard discretization errors but also for the various approximations done during the process to obtain the input data — computational meshes, velocity profiles, pressure or flow rate measurements.

In the present work, we focus our attention on the simulation step, using for now the standard newtonian incompressible Navier-Stokes equations, and we are interested in particular in (*i)* handling various boundary conditions settings allowing for a flexible framework with respect to the type of input data (velocity, pressure, flow rate, ...); (*ii)* handling of the discretization errors not only with respect to the physical fields (velocity and pressure) but also with respect to the geometry; (*iii)* dealing with the associated large computational cost, requiring high performance computing, through strong and weak scalability studies. We would like to emphasize that the accuracy with which the

[1] Laboratoire de Mathématiques de Besançon, UMR CNRS 6623, Université de Franche-Comté,16 route de Gray 25030 Besançon Cedex, France
[2] Laboratoire Jean Kuntzmann, Université Joseph Fourier Grenoble 1, BP53 38041 Grenoble Cedex 9, France, Tel.: +33476635497, Fax: +33476631263
[3] Université Grenoble 1 / CNRS, Laboratoire Interdisciplinaire de Physique / UMR 5588. Grenoble, F-38041, France
[4] CMUC, Department of Mathematics, Apartado 3008, EC Santa Cruz, 3001-501 Coimbra, Portugal
[5] Université de Strasbourg / CNRS, IRMA / UMR 7501. Strasbourg, F-67000, France

© EDP Sciences, SMAI 2013

boundary of the physical domain is approximated has a considerable impact on the quality of the numerical approximation, see for instance [14]. In particular, in the context of blood flow approximation, this takes particular relevance since the boundary might also be an unknown of the problem and needs to be accurately approximated.

To these ends, our computational framework builds upon the FEEL++ library [15, 16], which allows for arbitrary order continuous and discontinuous Galerkin methods in 1D, 2D and 3D, seamlessly in parallel, on simplices and hypercubes. Moreover, we highlight the fact that the mathematical kernel of the library can handle a large variety of numerical methods and it has been designed such that new ones can be easily included. In particular, in this work, we use and compare low order as well as high order approximations including for the geometry in 3D.

The outline of the article is as follows. We start by reminding the governing equations in a general setting and by explaining in more detail the difficulties related to the treatment of the boundary conditions in Section 2. The discussion is held both on the continuous and on the discrete level. The high performance computing issues are described in Section 3, where in particular a scalability analysis is performed. Section 4 is devoted to various numerical experiments, in order to validate the numerical method and also to apply it on a realistic geometry. Conclusions and some prospects are finally presented in Section 5.

2. Mathematical models and methods

Let $\Omega \subset \mathbb{R}^d$, $d \geq 1$, denote a bounded connected domain, representing the lumen of the vessel, or system of vessels, under investigation. In the context of blood flow, it is assumed that the unsteady incompressible Navier-Stokes equations hold. They read as:

$$\rho \frac{\partial \mathbf{u}}{\partial t} - 2\mathbf{div}(\mu \mathbf{D}(\mathbf{u})) + \rho(\mathbf{u} \cdot \boldsymbol{\nabla})\mathbf{u} + \nabla p = \mathbf{0}, \quad \text{in } \Omega \times I \tag{1}$$

$$\mathrm{div}(\mathbf{u}) = 0, \quad \text{in } \Omega \times I \tag{2}$$

where $I = (0, T]$, $\mathbf{u}$ and p are the velocity and pressure of the fluid, respectively, ρ and μ are the density and the dynamic viscosity of the fluid, respectively, and $\mathbf{D}(\mathbf{u})$ is the linear fluid deformation tensor (given by the expression $\frac{1}{2}(\nabla \mathbf{u} + \nabla \mathbf{u}^T)$). These notations allow to define the stress tensor $\boldsymbol{\sigma}(\mathbf{u}, p) = -p\mathbf{I} + 2\mu\mathbf{D}(\mathbf{u})$, where $\mathbf{I}$ is the identity tensor. System (1)-(2) is completed with appropriate initial and boundary conditions.

2.1. Variational Formulations

Our goal is to impose different kinds of boundary conditions (for both velocity and pressure) to system (1)-(2). Since these are independent of the presence of a convective term or a time derivative, in the following we focus only on the Stokes equations.

We are interested in fluid flows in different 2D and 3D geometries, mainly, (straight and curved) pipes and realistic vessels geometries. In all these cases, the domain of interest is denoted by Ω and its boundary $\partial\Omega$ is decomposed into three parts: Γ_w (where we will consider an adherence boundary condition), Γ_{in} (inlet) and Γ_{out} (outlet). Note that inlet and outlet can have several locally connected components (from the topological point of view), see e.g. Section 4.2.

The model (differential) problem is then written as: find $(\boldsymbol{u}, p)$ such that

$$-2\mu\mathbf{div}(\mathbf{D}(\mathbf{u})) + \nabla p = \mathbf{0} \text{ in } \Omega, \tag{3}$$

$$\mathrm{div}(\mathbf{u}) = 0 \text{ in } \Omega, \tag{4}$$

with boundary conditions that will be detailed later. To write formally a variational formulation of the problem (3)-(4), let us denote by $\mathbb{V}$ and $\mathbb{M}$ the functional spaces for the velocity and pressure fields, respectively. These spaces will be setted later according to the specific choices of boundary conditions.We will take, for the moment, $\mathbb{V} = [H^1(\Omega)]^d$ and $\mathbb{M} = L^2(\Omega)$.

Taking the scalar product of equation (3) by a test function $\mathbf{v} \in \mathbb{V}$, multiplying equation (4) by a test function $q \in \mathbb{M}$ and integrating the resulting equalities over Ω, we are lead to the following weak formulation: find $(\mathbf{u}, p) \in \mathbb{V} \times \mathbb{M}$ such that

$$-2\mu \int_\Omega \mathbf{div}(\mathbf{D}(\mathbf{u})) \cdot \mathbf{v}\, dx + \int_\Omega \nabla p \cdot \mathbf{v}\, dx = 0, \quad \forall \mathbf{v} \in \mathbb{V}, \tag{5}$$

$$\int_\Omega q\, \mathrm{div}(\mathbf{u})\, dx = 0, \quad \forall q \in \mathbb{M}. \tag{6}$$

We integrate by parts the first integral of equation (5). We obtain

$$2\mu \int_\Omega \mathbf{D}(\mathbf{u}) : \boldsymbol{\nabla}\mathbf{v}\, dx - 2\mu \int_{\partial\Omega} \mathbf{D}(\mathbf{u})\mathbf{n}\cdot\mathbf{v}\, ds + \int_{\partial\Omega} p\, \mathbf{v}\cdot\mathbf{n}\, ds - \int_\Omega p\, \mathrm{div}(\mathbf{v})\, dx = 0.$$

Note that, for symmetry reasons, the equality $\mathbf{D}(\mathbf{u}) : \boldsymbol{\nabla}\mathbf{v} = \mathbf{D}(\mathbf{u}) : \mathbf{D}(\mathbf{v})$ holds. Thus, the variational formulation of (3)-(4) can be written as: find $(\mathbf{u}, p) \in \mathbb{V} \times \mathbb{M}$ such that

$$2\mu \int_\Omega \mathbf{D}(\mathbf{u}) : \mathbf{D}(\mathbf{v})\, dx - 2\mu \int_{\partial\Omega} \mathbf{D}(\mathbf{u})\mathbf{n}\cdot\mathbf{v}\, ds + \int_{\partial\Omega} p\, \mathbf{v}\cdot\mathbf{n}\, ds - \int_\Omega p\, \mathrm{div}(\mathbf{v})\, dx = 0, \quad \forall \mathbf{v} \in \mathbb{V}, \tag{7}$$

$$\int_\Omega q\, \mathrm{div}(\mathbf{u})\, dx = 0, \quad \forall q \in \mathbb{M}, \tag{8}$$

or, equivalently: find $(\mathbf{u}, p) \in \mathbb{V} \times \mathbb{M}$ such that

$$2\mu \int_\Omega \mathbf{D}(\mathbf{u}) : \mathbf{D}(\mathbf{v})\, dx - \int_{\partial\Omega} \boldsymbol{\sigma}(\mathbf{u}, p)\mathbf{n}\cdot\mathbf{v}\, ds - \int_\Omega p\, \mathrm{div}(\mathbf{v})\, dx = 0, \quad \forall \mathbf{v} \in \mathbb{V}, \tag{9}$$

$$\int_\Omega q\, \mathrm{div}(\mathbf{u})\, dx = 0, \quad \forall q \in \mathbb{M}. \tag{10}$$

We have not yet incorporated the boundary conditions in the weak formulation. To do so, we start by the common boundary condition to all our following simulations which is the *no slip* condition on Γ_w:

$$\mathbf{u} = \mathbf{0} \text{ on } \Gamma_w.$$

The standard manner to deal with this essential boundary condition is to choose the functional space $\mathbb{V}$ as

$$\mathbb{V} = \{\mathbf{v} \in [H^1(\Omega)]^d \mid \mathbf{v} = \mathbf{0} \text{ on } \Gamma_w\}. \tag{11}$$

Within this functional setting for the velocity field, equations (9)-(10) are rewritten as: find $(\mathbf{u}, p) \in \mathbb{V} \times \mathbb{M}$ such that

$$2\mu \int_\Omega \mathbf{D}(\mathbf{u}) : \mathbf{D}(\mathbf{v})\, dx - \int_{\Gamma_{in}\cup\Gamma_{out}} \boldsymbol{\sigma}(\mathbf{u}, p)\mathbf{n}\cdot\mathbf{v}\, ds - \int_\Omega p\, \mathrm{div}(\mathbf{v})\, dx = 0, \quad \forall \mathbf{v} \in \mathbb{V}, \tag{12}$$

$$\int_\Omega q\, \mathrm{div}(\mathbf{u})\, dx = 0, \quad \forall q \in \mathbb{M}. \tag{13}$$

In the weak formulation (12)-(13), we have to take into account the boundary conditions on the inlet and outlet parts of the boundary. Recall that in this work we are interested in simulating fluid flows in straight and curved pipes and in realistic geometries of blood vessels. In all these cases, the computational domain is only a part of the physical one. Therefore, inlets and outlets are in fact the sections that separate the computational and physical domains. Consequently, special attention needs to be given on the boundary conditions enforced at the inlets or outlets to not change the physics of the problem.

In the following, let us consider different types of boundary conditions for the inlet and outlet sections. We start by the more classical boundary condition, the case where we know the velocity profiles at inlet and outlet (for example, Poiseuille profiles). The second case we consider is the free outlet condition. Finally, we focus on less classical boundary conditions that correspond to the case where we know only the pressure at the inlet and outlet sections. These nonstandard boundary conditions are very useful if we want to make a comparison between the simulations performed and the experimental data. Indeed, in physical experiments it is easier to impose pressure than velocity on both inlet and outlet.

2.1.1. *Dirichlet-Dirichlet boundary conditions*

Let us suppose that we know the velocity profiles at the inlets and outlets and that they are described by two functions $\mathbf{u}_{in} \in [H^{\frac{1}{2}}(\Gamma_{in})]^d$ and $\mathbf{u}_{out} \in [H^{\frac{1}{2}}(\Gamma_{out})]^d$ such that

$$\mathbf{u} = \mathbf{u}_{in} \quad \text{on } \Gamma_{in}, \tag{14}$$

$$\mathbf{u} = \mathbf{u}_{out} \quad \text{on } \Gamma_{out}. \tag{15}$$

In this case, in order to have a well posed problem, it is sufficient to choose

$$\mathbb{V} = \{\mathbf{v} \in [H^1(\Omega)]^d \mid \mathbf{v} = \mathbf{0} \text{ on } \Gamma_w,\ \mathbf{v} = \mathbf{u}_{in} \text{ on } \Gamma_{in},\ \mathbf{v} = \mathbf{u}_{out} \text{ on } \Gamma_{out}\} \tag{16}$$

and $\mathbb{M} = L^2_0(\Omega)$, where $L^2_0(\Omega)$ denotes the set of functions in $L^2(\Omega)$ with zero mean value. With this choice of functional spaces, problem (12)-(13) becomes: find $(\mathbf{u}, p) \in \mathbb{V} \times \mathbb{M}$ such that

$$2\mu \int_\Omega \mathbf{D}(\mathbf{u}) : \mathbf{D}(\mathbf{v})\, dx - \int_\Omega p\, \mathbf{div}(\mathbf{v})\, dx = 0, \quad \forall \mathbf{v} \in [H^1_0(\Omega)]^d, \tag{17}$$

$$\int_\Omega q\, \mathrm{div}(\mathbf{u})\, dx = 0, \quad \forall q \in L^2_0(\Omega). \tag{18}$$

The additional restriction of zero mean value to $L^2(\Omega)$ function allows to uniquely define the pressure in $\mathbb{M}$ and may be integrated in the variational formulation (17)-(18) by adding a suitable Lagrange multiplier. The final variational formulation, with the Lagrange multiplier, reads as: find $(\mathbf{u}, p, \zeta) \in \mathbb{V} \times \mathbb{M} \times \mathbb{R}$ such that

$$2\mu \int_\Omega \mathbf{D}(\mathbf{u}) : \mathbf{D}(\mathbf{v})\, dx - \int_\Omega p\, \mathbf{div}(\mathbf{v})\, dx = 0, \quad \forall \mathbf{v} \in [H^1_0(\Omega)]^d, \tag{19}$$

$$\int_\Omega (q\, \mathrm{div}(\mathbf{u}) + \zeta q)\, dx = 0, \quad \forall q \in \mathbb{M}, \tag{20}$$

$$\int_\Omega p\xi\, dx = 0, \quad \forall \xi \in \mathbb{R}. \tag{21}$$

2.1.2. *Dirichlet-Neumann boundary conditions*

The previous Dirichlet-Dirichlet boundary conditions do not correspond to the most common real situation because, in general, we do not know exactly the velocity profile at the outlet sections, even if we assume that we know it at the inlets. Indeed, it is difficult to predict the velocity profile at outlets since it depends on the channel geometry or the number of outlets sections in a vessel network, for example. Therefore, we can consider the so-called *free outlet* conditions. These boundary conditions can be formalized as

$$\mathbf{u} = \mathbf{u}_{in} \qquad \text{on } \Gamma_{in}, \tag{22}$$

$$\boldsymbol{\sigma}(\mathbf{u}, p)\mathbf{n} = \mathbf{0} \qquad \text{on } \Gamma_{out}. \tag{23}$$

In this case, we retrieve the same variational formulation as in (17)-(18), but taking

$$\mathbb{V} = \{\mathbf{v} \in [H^1(\Omega)]^d \mid \mathbf{v} = \mathbf{0} \text{ on } \Gamma_w,\ \mathbf{v} = \mathbf{u}_{in} \text{ on } \Gamma_{in}\} \quad \text{and} \quad \mathbb{M} = L^2_0(\Omega). \tag{24}$$

It appears that this choice of boundary conditions is not in agreement with our problem since the velocity field is not necessarily orthogonal to the outlet section. See [13] for some studies on these boundary conditions in the framework of human lung modeling and some numerical experiments showing this defect or [17] for some numerical examples in the case of blood flow simulation.

2.1.3. *Neumann-Neumann boundary conditions*

Boundary conditions involving the pressure and the stress tensor: To overcome the difficulties related to the free outlet boundary conditions, we assume now that we know the exact value of the normal stress tensor at inlets and outlets. Due to the definition of the stress tensor, it is sufficient to know both the pressure and the velocity at inlets and outlets to enforce such boundary conditions. They read as

$$\boldsymbol{\sigma}(\mathbf{u}, p)\mathbf{n} = \boldsymbol{\sigma}_{in}\mathbf{n} = -p_{in}\mathbf{n} + 2\mu\mathbf{D}(\mathbf{u_{in}})\mathbf{n}, \qquad \text{on } \Gamma_{in}, \tag{25}$$

$$\boldsymbol{\sigma}(\mathbf{u}, p)\mathbf{n} = \boldsymbol{\sigma}_{out}\mathbf{n} = -p_{out}\mathbf{n} + 2\mu\mathbf{D}(\mathbf{u_{out}})\mathbf{n}, \qquad \text{on } \Gamma_{out}. \tag{26}$$

The corresponding variational formulation is then written as: find $(\mathbf{u}, p) \in \mathbb{V} \times \mathbb{M}$ such that

$$2\mu \int_\Omega \mathbf{D}(\mathbf{u}) : \mathbf{D}(\mathbf{v})\, dx - \int_\Omega p\, \mathbf{div}(\mathbf{v})\, dx = \int_{\Gamma_{in}} \boldsymbol{\sigma}_{in}\mathbf{n} \cdot \mathbf{v}\, ds + \int_{\Gamma_{out}} \boldsymbol{\sigma}_{out}\mathbf{n} \cdot \mathbf{v}\, ds, \quad \forall \mathbf{v} \in \mathbb{V}, \tag{27}$$

$$\int_\Omega q\, \mathrm{div}(\mathbf{u})\, dx = 0, \quad \forall q \in \mathbb{M}, \tag{28}$$

where

$$\mathbb{M} = L^2(\Omega) \quad \text{and} \quad \mathbb{V} = \{\mathbf{v} \in [H^1(\Omega)]^d \mid \mathbf{v} = \mathbf{0} \text{ on } \Gamma_w\}. \tag{29}$$

Boundary conditions involving the pressure without the stress tensor: It is obvious that in the most common situations, the normal stress tensor is not known at the inlets or outlets. However, in some physical experiments we have direct access to the pressure at inlets and outlets because it is imposed by the experience. Therefore, we suppose the existence of two functions $p_{in} \in H^{-\frac{1}{2}}(\Gamma_{in})$ and $p_{out} \in H^{-\frac{1}{2}}(\Gamma_{out})$ such that

$$\boldsymbol{\sigma}(\mathbf{u}, p)\mathbf{n} = -p_{in}\mathbf{n} \quad \text{on } \Gamma_{in}, \tag{30}$$
$$\boldsymbol{\sigma}(\mathbf{u}, p)\mathbf{n} = -p_{out}\mathbf{n} \quad \text{on } \Gamma_{out}. \tag{31}$$

The corresponding variational formulation is: find $(\mathbf{u}, p) \in \mathbb{V} \times \mathbb{M}$ such that

$$2\mu \int_\Omega \mathbf{D}(\mathbf{u}) : \mathbf{D}(\mathbf{v})\, dx - \int_\Omega p\, \mathbf{div}(\mathbf{v})\, dx = -\int_{\Gamma_{in}} p_{in}\mathbf{n} \cdot \mathbf{v}\, ds - \int_{\Gamma_{out}} p_{out}\mathbf{n} \cdot \mathbf{v}\, ds, \quad \forall \mathbf{v} \in \mathbb{V}, \tag{32}$$
$$\int_\Omega q\, \text{div}(\mathbf{u})\, dx = 0, \quad \forall q \in \mathbb{M}, \tag{33}$$

where $\mathbb{M} = L^2(\Omega)$ and $\mathbb{V} = \{\mathbf{v} \in [H^1(\Omega)]^d \mid \mathbf{v} = \mathbf{0} \text{ on } \Gamma_w\}$.

The variational formulation (32)-(33) gives an appropriate framework for imposing boundary conditions on the pressure. However, this option may only make sense if the stress tensor is diagonal at inlets and outlets, meaning that the computational domain Ω is separated by inlets and outlets from a perfect gas-like media at pressure p_{in} and p_{out} respectively. However, this is obviously not the case for blood flow simulations, where Ω is the computational domain, corresponding to a part of the circulatory system and this is why this formulation cannot be used in our case. For example, the formulation (32)-(33) cannot even recover the Poiseuille flow in a straight pipe!

2.1.4. *Mixed boundary conditions*

To overcome the problems related to the previous variational formulations, especially the one given by equations (32)-(33), we introduce in this section an additional boundary condition for the velocity on inlets and outlets. These nonstandard boundary conditions involving the pressure have been studied, from a theoretical point of view, in [8], for the first time, and recently in [2]. More precisely, we impose $\mathbf{u} \times \mathbf{n} = \mathbf{0}$ (or $\mathbf{u} \cdot \mathbf{t} = 0$ where $\mathbf{t}$ stands for the unit tangent vector for $d = 2$) on $\Gamma_{in} \cup \Gamma_{out}$. The new variational formulation is then written as: find $(\mathbf{u}, p) \in \mathbb{V} \times \mathbb{M}$ such that

$$2\mu \int_\Omega \mathbf{D}(\mathbf{u}) : \mathbf{D}(\mathbf{v})\, dx - \int_\Omega p\, \mathbf{div}(\mathbf{v})\, dx = -\int_{\Gamma_{in}} p_{in}\mathbf{n} \cdot \mathbf{v}\, ds - \int_{\Gamma_{out}} p_{out}\mathbf{n} \cdot \mathbf{v}\, ds, \quad \forall \mathbf{v} \in \mathbb{V}, \tag{34}$$
$$\int_\Omega q\, \text{div}(\mathbf{u})\, dx = 0, \quad \forall q \in \mathbb{M}, \tag{35}$$

where

$$\mathbb{M} = L^2(\Omega) \quad \text{and} \quad \mathbb{V} = \{\mathbf{v} \in [H^1(\Omega)]^d \mid \mathbf{v} = \mathbf{0} \text{ on } \Gamma_w,\ \mathbf{v} \times \mathbf{n} = \mathbf{0} \text{ on } \Gamma_{in} \cup \Gamma_{out}\}. \tag{36}$$

This formulation allows to impose the pressure on inlets and outlets while, at the same time, enforcing the velocity field to be parallel to to the outward normal vector at inlets and outlets.

From the numerical point of view, the constraints $\mathbf{u} \times \mathbf{n} = \mathbf{0}$ on $\Gamma_{in} \cup \Gamma_{out}$ (or $\mathbf{u} \cdot \mathbf{t} = 0$ on $\Gamma_{in} \cup \Gamma_{out}$ for $d = 2$) can be taken into account by using Lagrange multipliers or penalty techniques. In the case of Lagrange multipliers,

the problem reads as: find $(\mathbf{u}, p, \lambda_{in}, \lambda_{out}) \in \mathbb{V} \times \mathbb{M} \times \Lambda_{in} \times \Lambda_{out}$ such that

$$
\begin{aligned}
2\mu \int_\Omega \mathbf{D}(\mathbf{u}) : \mathbf{D}(\mathbf{v})\, dx - \int_\Omega p\, \mathbf{div}(\mathbf{v})\, dx & \\
- \int_{\Gamma_{in}} \lambda_{in} \cdot \mathbf{v} \times \mathbf{n}\, ds - \int_{\Gamma_{out}} \lambda_{out} \cdot \mathbf{v} \times \mathbf{n}\, ds &= - \int_{\Gamma_{in}} p_{in} \mathbf{n} \cdot \mathbf{v}\, ds - \int_{\Gamma_{out}} p_{out} \mathbf{n} \cdot \mathbf{v}\, ds, \quad \forall \mathbf{v} \in \mathbb{V}, \quad (37)\\
\int_\Omega q\, \mathrm{div}(\mathbf{u})\, dx &= 0, \quad \forall q \in \mathbb{M}, \quad (38)\\
\int_{\Gamma_{in}} \eta_{in} \cdot \mathbf{u} \times \mathbf{n}\, ds &= 0, \quad \forall \eta_{in} \in \Lambda_{in}, \quad (39)\\
\int_{\Gamma_{out}} \eta_{out} \cdot \mathbf{u} \times \mathbf{n}\, ds &= 0, \quad \forall \eta_{out} \in \Lambda_{out}, \quad (40)
\end{aligned}
$$

where $\mathbb{M} = L^2(\Omega)$, $\mathbb{V} = \{\mathbf{v} \in [H^1(\Omega)]^d \mid \mathbf{v} = \mathbf{0} \text{ on } \Gamma_w\}$, $\Lambda_{in} = [H^{-\frac{1}{2}}(\Gamma_{in})]^d$ and $\Lambda_{out} = [H^{-\frac{1}{2}}(\Gamma_{out})]^d$.

2.2. Discretisation choices

We introduce in the sequel the discretisation strategy, following the framework presented in [7]. Let δ be a discretisation parameter. We define $\hat{K} \subset \mathbb{R}^d$ $(d = 1, 2, 3)$ a reference elementary convex, *e.g.* a simplex or a hypercube. We denote by $\mathcal{T}_\delta$ a finite collection of nonempty, disjoint open simplices or hypercubes $\mathcal{T}_\delta \equiv \mathcal{T}_{(h,k)} = \{K = \boldsymbol{\varphi}^{\mathrm{geo}}_{K,k}(\hat{K})\}$ forming a partition of Ω such that $h = \max_{K \in \mathcal{T}_\delta} h_K$, with h_K denoting the diameter of the element $K \in \mathcal{T}_\delta$ and $\boldsymbol{\varphi}^{\mathrm{geo}}_{K,k}$ is the polynomial of degree k that maps $\hat{K}$ to K which is also called the geometric transformation. The partition $\mathcal{T}_\delta$ induces a discretization of Ω, denoted Ω_δ, defined as the union of the closure of all elements in this partition. Note that if Ω is a polyhedral domain then $\Omega_\delta = \Omega$. Following these notations, we denote $\Gamma_{in,\delta}$, $\Gamma_{out,\delta}$, $\Gamma_{w,\delta}$ the discretization of $\Gamma_{in}, \Gamma_{out}, \Gamma_w$ respectively.

We say that a hyperplanar closed subset F of $\overline{\Omega_\delta}$ is a *mesh face* if it has positive $(d-1)$-dimensional measure and if either there exist K_1, $K_2 \in \mathcal{T}_\delta$ such that $F = \partial K_1 \cap \partial K_2$ (in this case F is called an *internal face*) or there exists $K \in \mathcal{T}_\delta$ such that $F = \partial K \cap \partial \Omega_\delta$ (and F is called a *boundary face*). Internal faces are collected in the set $\mathcal{F}^i_\delta$, boundary faces in $\mathcal{F}^b_\delta$ and we let $\mathcal{F}_\delta := \mathcal{F}^i_\delta \cup \mathcal{F}^b_\delta$. For all $F \in \mathcal{F}_\delta$, we define $\mathcal{T}_F := \{K \in \mathcal{T}_\delta \mid F \subset \partial K\}$. For every interface $F \in \mathcal{F}^i_\delta$ we introduce two associated normals to the elements in $\mathcal{T}_F$ and we have $\mathbf{n}_{K_1,F} = -\mathbf{n}_{K_2,F}$, where $\mathbf{n}_{K_i,F}$, $i \in \{1, 2\}$, denotes the unit normal to F pointing out of $K_i \in \mathcal{T}_F$. On a boundary face $F \in \mathcal{F}^b_\delta$, $\mathbf{n}_F = \mathbf{n}_{K,F}$ denotes the unit normal pointing out of Ω_δ.

Without loss of generality we suppose from now on that we work with simplicial elements. Given a positive integer N, we denote by $\mathbb{P}^N(\hat{K})$ and $\mathbb{P}^N(K)$ the spaces of polynomials of total degree less or equal than N defined in $\hat{K}$ and K respectively. We define $P^N_c(\Omega_\delta \equiv \Omega_{(h,k)})$ and $[P^N_c(\Omega_\delta \equiv \Omega_{(h,k)})]^d$ with $k \geqslant 1$:

$$
P^N_c(\Omega_\delta) = \{v \in C^0(\Omega_\delta) \mid v \circ \boldsymbol{\varphi}^{\mathrm{geo}}_{K,k} \in \mathbb{P}^N(\hat{K})\ \forall K \in \mathcal{T}_\delta\}, \quad [P^N_c(\Omega_\delta)]^d = \prod_1^d P^N_c(\Omega_\delta). \tag{41}
$$

We choose the generalized Taylor-Hood finite element for the velocity-pressure discretisation, that is to say we look for the velocity in $[P^{N+1}_c(\Omega_{(h,k_{\mathrm{geo}})})]^d$ and the pressure in $P^N_c(\Omega_{(h,k_{\mathrm{geo}})})$. We shall use from now on the notation $\mathbb{P}_{N+1}\mathbb{P}_N\mathbb{G}_{k_{\mathrm{geo}}}$ to specify exactly the discretisation spaces used for the velocity, pressure and geometry, respectively. The resulting approximate velocity and pressure fields are denoted by $\mathbf{u}_\delta$ and p_δ, respectively.

Regarding the formulation (37)-(40) in Section 2.1.4, we follow the same discretization process as previously for velocity, pressure and geometry. Denote $\Gamma_\delta = \Gamma_{in,\delta} \cup \Gamma_{out,\delta}$ and we look for the discrete Lagrange multiplier in

$$
[P^N_c(\Gamma_\delta)]^d = \prod_1^d P^N_c(\Gamma_\delta), \quad \text{where } P^N_c(\Gamma_\delta) = \{v \in C^0(\Gamma_\delta) \mid v \circ \boldsymbol{\varphi}^{\mathrm{geo}}_{F,k} \in \mathbb{P}^N(\hat{F})\ \forall F \in \mathcal{F}^b_\delta \cap \Gamma_\delta, \}, \tag{42}
$$

$\hat{F} \subset \mathbb{R}^d$ is a reference elementary convex of topological dimension $d-1$ (*i.e.*corresponding to a face of $\hat{K}$) and $\boldsymbol{\varphi}^{\mathrm{geo}}_{F,k}$ is the polynomial of degree k_{geo} that maps $\hat{F}$ to F. A natural choice for the polynomial degree of the Lagrange multipliers is to take $N + k_{\mathrm{geo}}$ where N is the velocity polynomial order in order to ensure that (39)- (40) are satisfied exactly. This will require further analysis. However the code, see Listing 1, and some promising initial numerical results, see Figure 1, are already available.

3. HPC WITH FEEL++

3.1. General description

In order to perform our numerical experiments, we use the FEEL++ library, *Finite Element Embedded Language in* C++, see [15, 16], which provides a clear and easy interface to solve complex PDE systems. It aims at bringing the scientific community a tool for the implementation of advanced numerical methods and high performance computing. Some recent applications of FEEL++ to multiphysics problems in blood flow context can be found in the literature, see *e.g.* [7, 9, 14].

FEEL++ relies on a so-called *domain specific embedded language* (DSEL) designed to closely match the Galerkin mathematical framework. In computer science, DS(E)Ls are used to partition complexity and, in our case, the DSEL splits low level mathematics and computer science on one side (leaving the FEEL++ developer to enhance them) and high level mathematics as well as physical applications to the other side (left to the FEEL++ user). This allows FEEL++ to be used for teaching purposes, solving complex problems with multiple physics and scales or rapid prototyping of new methods, schemes or algorithms.

The DSEL on FEEL++ provides access to powerful, yet with a simple and seamless interface, tools such as interpolation or the clear translation of a wide range of variational formulations into the variational embedded language. Combined with this robust engine lie also state of the art arbitrary order finite elements — including handling high order geometrical approximations, — high order quadrature formulas and robust nodal configuration sets. The tools at the user's disposal grant the flexibility to implement numerical methods that cover a large combination of choices from meshes, function spaces or quadrature points using the same integrated language and control at each stage of the solution process of the numerical approximations.

To illustrate, Listing 1 displays the full code to implement the formulation described in Section 2.1.4. It shows the power of the DSEL to write this complex and non-standard mathematical formulation in a very compact and expressive way.

LISTING 1. FEEL++ code to implement the mixed boundary conditions description in Section 2.1.4 using Lagrange multipliers

```
    Environment env( _argc=argc, _argv=argv,
                     _desc=feel_options(),
                     _about=about(_name="poiseuille",
                                  _author="Feel++ Consortium",
                                  _email="feelpp-devel@feelpp.org"));
auto mesh = Cylinder<1>();
auto Vh = Pch<2,Vectorial>( _mesh=mesh ); // velocity
auto Qh = Pch<1>( mesh ); // pressure
auto Lmesh = merge( mesh->trace(markedfaces(mesh,"inlet")),
                    mesh->trace(markedfaces(mesh,"outlet")) );
auto Lh = Pch<2,Vectorial>( _mesh=Lmesh ); // Lagrange mult.
auto Xh = Vh*Qh*Lh;
auto U = Xh->element();
auto u = U.element<0>();
auto p = U.element<1>();
auto l = U.element<2>();

auto a = form2( _trial=Xh, _test=Xh );
a = integrate(_range=elements(mesh),
              _expr=sym(gradt(u))*trans(sym(grad(u))) );
a+= integrate(_range=elements(mesh),
              _expr=-div(u)*idt(p)-divt(u)*id(p));
a+= integrate(_range=markedfaces(mesh,{"inlet","outlet"}),
              _expr=-trans(cross(idt(u),N()))*id(l)-trans(cross(id(u),N()))*idt(l));
auto l = form1( _test=Vh );
l = integrate(_range=markedfaces(mesh,"inlet"),
              _expr=-trans(p_in*N())*id(v));
l+= integrate(_range=markedfaces(mesh,"outlet"),
              _expr=-trans(p_out*N())*id(v));
a+=on(_range=markedfaces(mesh,"wall"), _rhs=l, _element=u,
      _expr=zero<3,1>());
```

```
a.solve(_rhs=l,_solution=U);
```

Figure 1 displays the results of the above code using the Stokes mixed boundary condition formulation of Section 2.1.4.

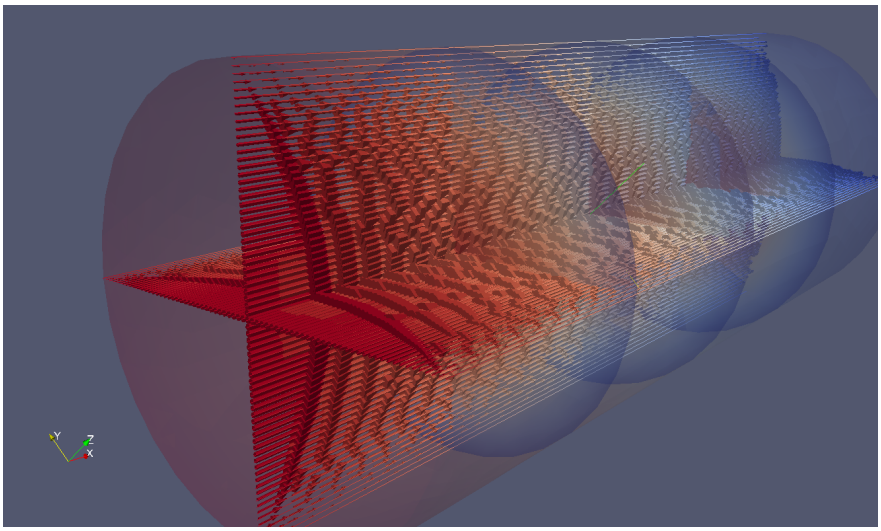


FIGURE 1. Streamlines, velocity arrows and pressure isosurfaces of a Poseuille flow in a cylinder using mixed boundary conditions

In this paper, we use recent developments which allow to operate on large-scale parallel infrastructures. The general strategy used is *parallel data* framework using MPI and thanks to DSEL the MPI communications are seamless to the user: (*i*) we start with automatic mesh partitioning using GMSH [11] (Chaco/Metis) — adding information about ghost cells with communication between neighbor partition;— (*ii*) the FEEL++ parallel data structures such as meshes, (elements of) function spaces — create a parallel degrees of freedom table with local and global views; — (*iii*) and finally we use the library PETSC [3–5] which provides access to a Krylov subspace solvers(KSP) coupled with PETSC preconditioners such as Block-Jacobi, ASM, GASM.

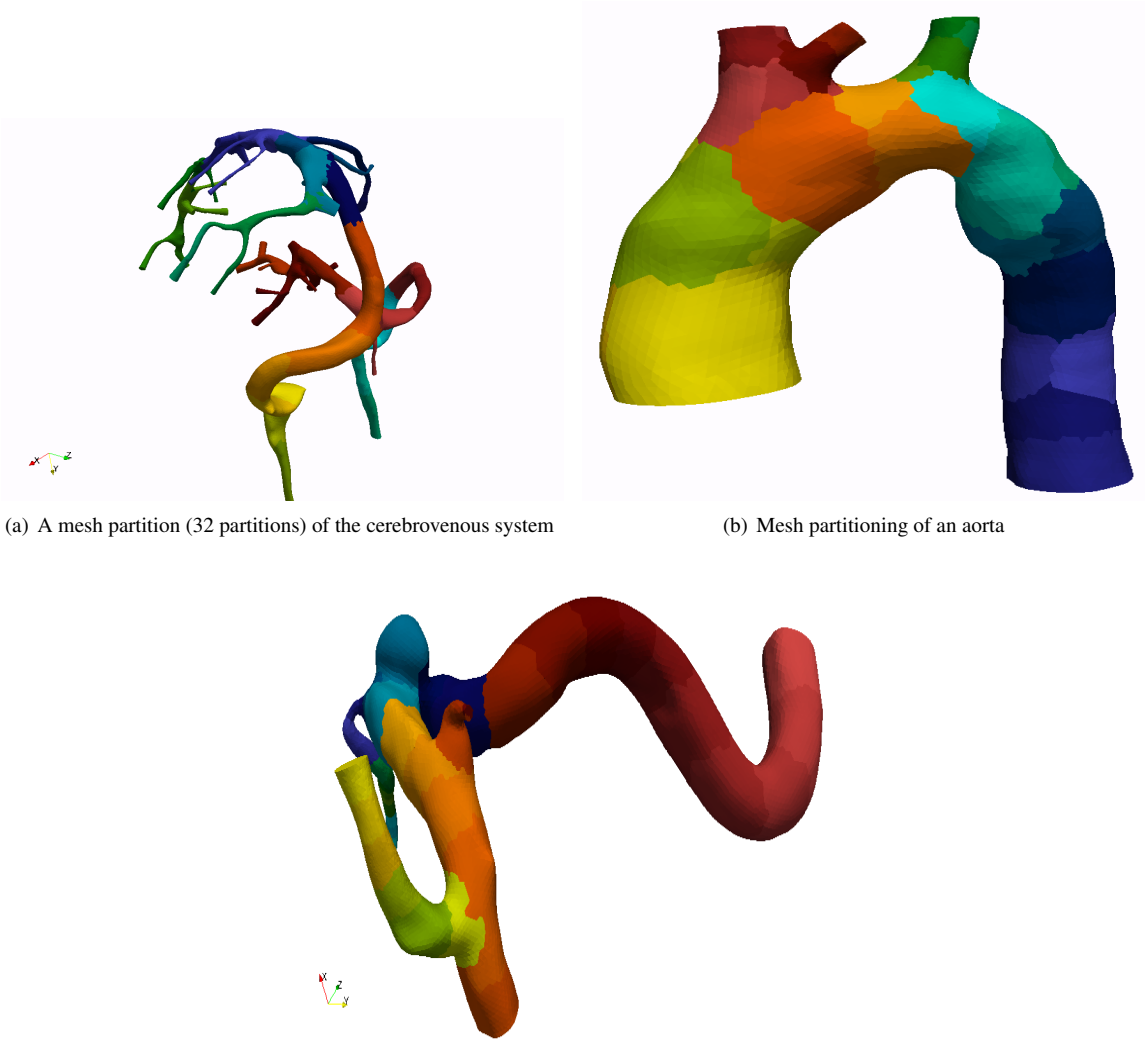
Remark 3.2. *The last preconditioner is an additive variant of the Schwarz alternating method for the case of many subregions, see [18]. For each sub-preconditioners (in the subdomains),* PETSC *allows to choose a wide range of sequential preconditioners such as LU, ILU, JACOBI, ML. Moreover, precondioner ASM or GASM can be used with or without an algebraic overlap. Other parallel preconditioners are available in* PETSC *but not used here. In particular we would like to mention the* MUMPS *direct parallel solver [1]. We use it both as solver and preconditioner for iterative solves.* FIELDSPLIT *preconditioners are also of notice for the applications we have: they allow to exploit the structure of block matrix.*

From now on, we shall denote GASM1 (resp GASM2) for a preconditioner GASM with an overlap of size 1 (respectively, 2). A complete description of the FEEL++ high performance framework — with blood flow applications — is available in the thesis [6].

3.3. Challenges in handling complex geometries

Geometries that describe blood flow are complex with three-dimensional structures, often asymmetric and presenting a variable pattern. From these medical images, a complex process can be developed to build computational meshes, which the objective of many research and development projects. Figure 2(a) shows a computational mesh obtained through this process from MRA images, see for details [17]. Other realistic meshes can be found on GMSH [11] website, see for example Figures 2(b) and 2(c) that display an aorta and an artery with an aneurysm. The mesh generation was done using GMSH [12].

Once we have the computational meshes are built, they are partitioned, see Section 3.1, and the blood flow computation is distributed using MPI. One of the crucial points in parallel computation is to check if our strategy is robust and scalable over a large number of processors.



(a) A mesh partition (32 partitions) of the cerebrovenous system

(b) Mesh partitioning of an aorta

(c) Mesh partitioning of an artery with an aneurysm

FIGURE 2. Computational meshes of the cerebrovenous system, of a realistic aorta and of an artery with aneurysm geometries.

3.4. **Scalability analysis**

We now turn to a scalability analysis applied to problems in fluid mechanics using FEEL++. We propose here to study two configurations: *(i)* a Stokes model with a simple geometry (a cylinder), *(ii)* a Navier-Stokes model with complex and realistic geometry (an aorta), presented in Figure 2(b), in order to check the strong scalability — increase processing units for a fixed size problem — and weak scalability — increase processing units along with the problem size. Each test will measure *(i)* the assembly CPU time for matrices and vectors as well as the necessary MPI communications, *(ii)* the linear and/or nonlinear algebraic system CPU time.

Regarding the strong scalability, the problem size stays fixed while the number of processing elements increases from $N^{\text{core}} = N^{\text{core}}_{\text{min}}, \ldots, N^{\text{core}}_{\text{max}}$ and we are interested in the *speed-up* $= t_{N^{\text{core}}_{\text{min}}}/t_{N^{\text{core}}}$ obtained with N^{core} cores where $t_{N^{\text{core}}_{\text{min}}}$ is the elapsed wall-clock time with $N^{\text{core}}_{\text{min}}$ cores and $t_{N^{\text{core}}}$ the elapsed wall-clock time with N^{core} processors.

As to the weak scalability, the problem size increases with the number of processing units such that the problem size assigned to each processing element remains constant throughout all computations and we measure the *efficiency* $= 100(t_{N^{\text{core}}_{\text{min}}}/t_{N^{\text{core}}})$.

3.4.1. *Stokes model*

To perform the scalability test, we consider a cylinder of length L_C and radius 0.5. We impose Dirichlet-Neumann boundary conditions, *i.e.*, a parabolic known velocity profile at the inlet and a zero normal stress at the outlet, see Section 2.1.2.

(a) Strong scalability settings

MESH	TETRAHEDRONS	DOF
M8	111,475	530,275
M9	151,443	712,545
M10	210,526	980,105

(b) Weak scalability settings

N^{core}	L_C	DOF(M4)	DOF(M5)	DOF(M6)
32	16.0	297,372	398,181	563,574
16	8.0	146,094	197,262	290,423
8	4.0	77,255	105,296	148,027
4	2.0	39,834	56,075	75,005
2	1.0	23,950	31,971	41,869
1	0.5	15,705	19,207	26,523

TABLE 1. Configurations used for the scalability tests with the Stokes model and approximation spaces $\mathbb{P}_2\mathbb{P}_1\mathbb{G}_1$.

The scalability tests of Stokes model were made using the $\mathbb{P}_2\mathbb{P}_1\mathbb{G}_1$ approximation spaces. The different configurations used are displayed in Table 1(a) for the strong scalability and in Table 1(b) for the weak scalability. Figure 3 displays all the results obtained. We use several meshes associated to a characteristic mesh size which we denote by M4, M5, M6, etc. The larger the index, the finer the mesh. We also compare two parallel preconditioners GASM1 and GASM2. Both precondtioners use a direct subsolver (LU with MUMPS) within the subdomains.

Figures 3(a) and 3(b) display the elapsed wall-clock CPU time to solve the linear system for strong and weak scalability. We see that the preconditioner GASM1 is more efficient than GASM2 for each scalability test. Figure 3(c) plots the number of solver iterations associated to the strong scalability test: the preconditioner GASM2 reduces the number of iteration which is due to the overlap. However, in both cases, the number of iterations increases significantly as we increase the number of cores. A similar behaviour for weak scalability is observed in Figure 3(c). In particular when using 32 cores the number of solver iterations increases a lot and we have slow convergence. Figures 3(e) and 3(f) plot the assembly time of the algebraic structures. The performances are perfect with for both scalability tests up to 16 processors and it deteriorates at 32 cores as the communication become dominant. We note however that the speed-up is improving with the finest meshes.

To conclude the overall — the sum of assembly and solves times — measured speed-up is very good, even above an ideal speed-up, see Figure 3(g). As to weak scalability, see Figure 3(h), performance decreases with increasing the number of cores which is explained by the strong increase in the number of iterations.

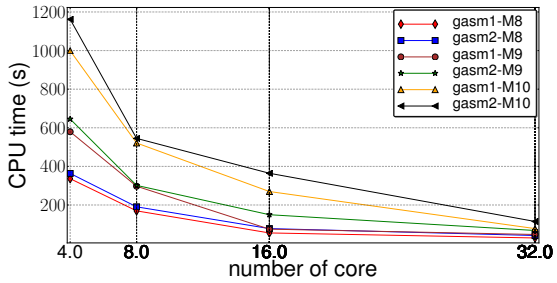
3.4.2. *Navier-Stokes model*

We now turn to the strong scalability test for the Navier-Stokes strategy on a realistic geometry — an aorta, — see Figure 2(b). The formulation of this nonlinear model is described with equations (1)-(2). Regarding the problem setting, we have a nonzero value of normal stress tensor at the inlet, a zero normal stress is set on the four outlets and a no-slip condition on the wall, see Section 2.1.3. We measure the computational time only during the first time step.

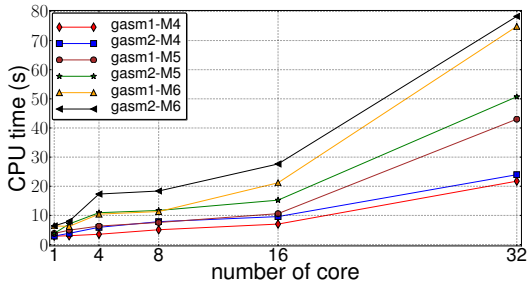
MESH	TETRAHEDRONS	DOF
M2	42,744	199,371
M3	109,997	497,525
M4	223,120	989,237

TABLE 2. Configurations used for the Navier-Stokes model with $\mathbb{P}_2\mathbb{P}_1\mathbb{G}_1$ approximations.

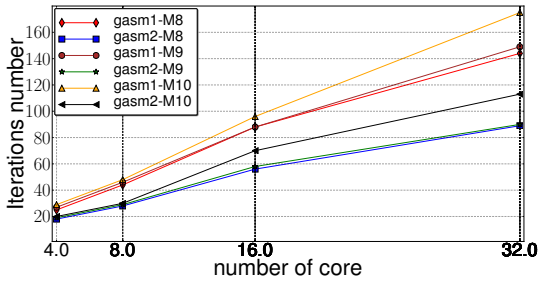
The strong scalability tests are obtained using $\mathbb{P}_2\mathbb{P}_1\mathbb{G}_1$ approximations. The number of mesh elements and degrees of freedom are reported in Table 2 and the scalability results in Figure 4. The speed-up for assembly of the algebraic structures is ideal up to 16 processors but at 32 cores performance stagnates because communications become dominant compared to the cost of local assembly. Indeed we note that speed-up improves with the finest meshes, i.e. the local assembly cost is still important. As to the speed-up of the solver strategy, it is excellent, see Figure 4(b). Indeed,



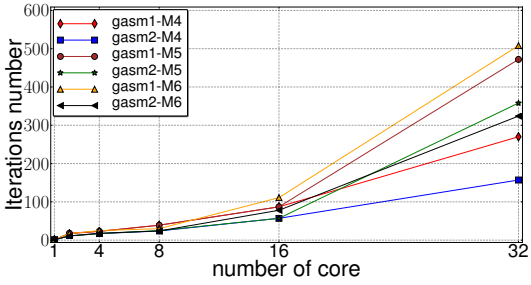
(a) CPU time to solve the algebraic system (strong scalability)



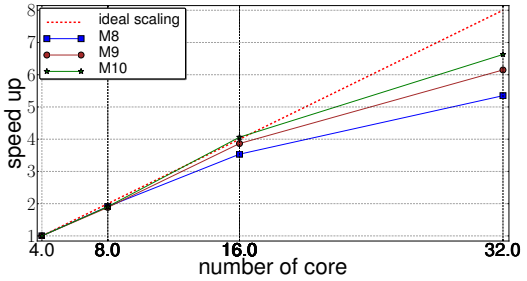
(b) CPU time to solve the algebraic system (weak scalability)



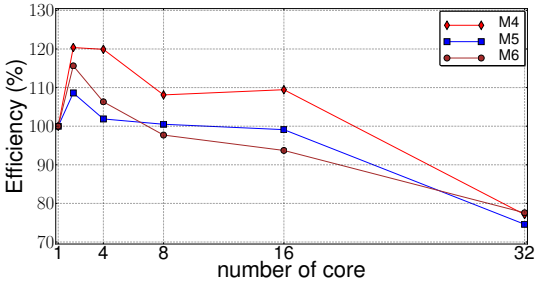
(c) Number of solver iterations (strong scalability)



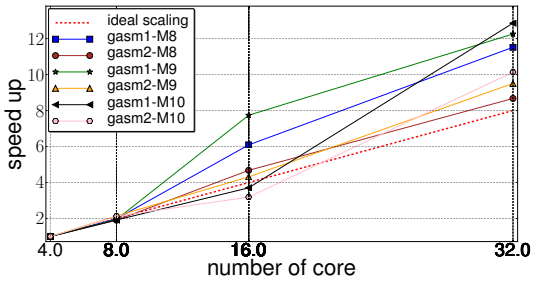
(d) Number of solver iterations (weak scalability)



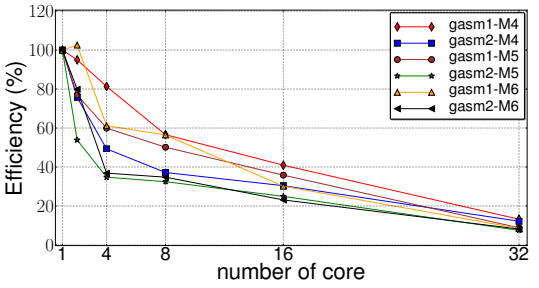
(e) Assembly speed-up (strong scalability)



(f) Assembly efficiency (weak scalability)



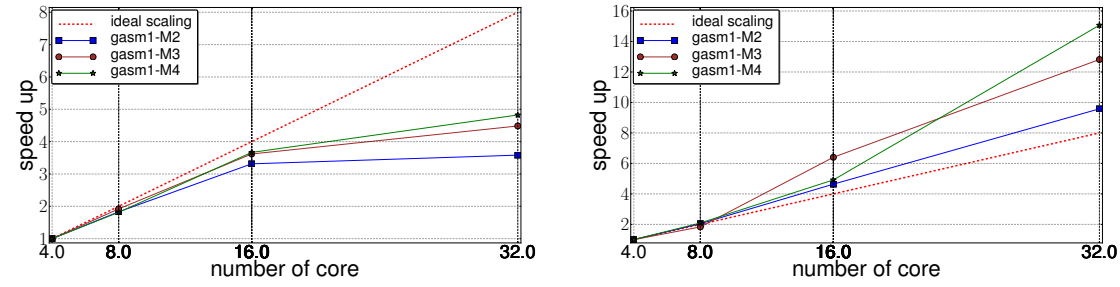
(g) Global speed-up (strong scalability)



(h) Global efficiency (weak scalability)

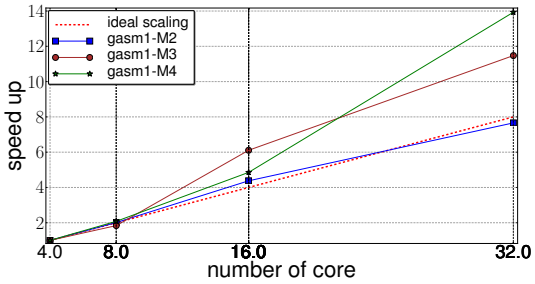
FIGURE 3. Results for the scalability tests with the Stokes model.

the number of iterations of the linear solves remains relatively low, see Figure 4(d). Also note that the scalability of this application is dominated by the solver, see Figure 4(d), and not by assembly: we obtain, in overall, an excellent global strong scalability.

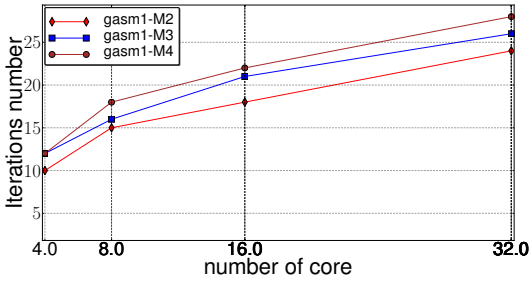


(a) Assembly speed-up

(b) Non-linear solves speed-up



(c) Global speed-up



(d) Number of iterations needed by the first linear solve

FIGURE 4. Results for the strong scalability tests with the Navier-Stokes model.

4. NUMERICAL RESULTS

4.1. **Convergence analysis of the Stokes formulations**

In this Section we verify that the various formulations (presented in Section 2 after discretization) converge with the proper rates. We check with a simple 3D Poiseuille flow in a cylinder with a base of radius $r = 1$ and length $L = 5$ centered at $(2.5, 0, 0)$, see Figure 5. The exact solution for this benchmark is

$$\boldsymbol{u}_{ex} = \left(\frac{(p_{in} - p_{out})^2 r^2}{4\mu L} \left(1 - \frac{y^2 + z^2}{r^2} \right), 0, 0 \right), \quad p_{ex} = \frac{p_{out} - p_{in}}{L} x + p_{in}. \tag{43}$$

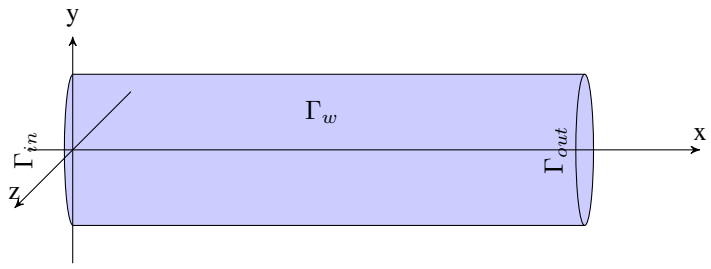


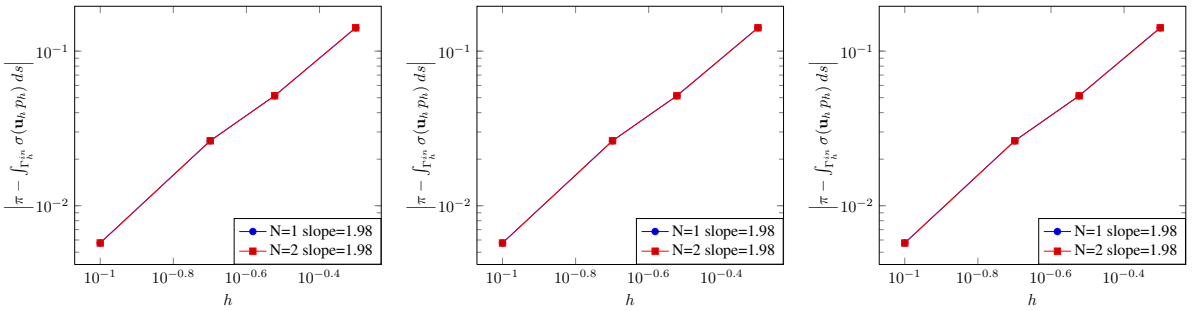
FIGURE 5. Geometry for convergence rates verification Ω

We note that the discrete geometry, Ω_δ, will be different from the exact one, Ω. Hence, the geometry approximation will play a crucial role in the verification process. To measure the geometric approximation error, not only we measure the error in the standard $L^2(\Omega_\delta)$ and $H^1(\Omega_\delta)$ norms but also the applied forces by integrating the stress tensor on one of the curved boundaries, here Γ_{in}. That is to say, we compute

$$\boldsymbol{F}_\delta = \int_{\Gamma_{in,\delta}} \boldsymbol{\sigma}(\mathbf{u}, p)\mathbf{n}\, ds. \tag{44}$$

Thanks to (43), we can compute by hand the exact counterpart $\boldsymbol{F}_{ex} = \int_{\Gamma_{in}} \boldsymbol{\sigma}(\mathbf{u_{ex}}, p_{ex})\mathbf{n}\, ds$ and hence $\|\boldsymbol{F}_{ex} - \boldsymbol{F}_\delta\|_2$ is measured. This is the simplest way to take properly into account the error in the geometry as well as in velocity and pressure. Indeed we have otherwise only access to $L^2(\Omega_\delta)$ and $H^1(\Omega_\delta)$ norms and not $L^2(\Omega)$ and $H^1(\Omega)$. We remark that a similar test case was used in [14] to check for the dependence of the geometry approximation in the numerical convergence verification process.

We start first with the standard finite element approximations using first order geometry, namely $\mathbb{P}_2\mathbb{P}_1\mathbb{G}_1$ and $\mathbb{P}_3\mathbb{P}_2\mathbb{G}_1$, applied to the Dirichlet-Dirichlet (see Section 2.1.1), Dirichlet-Neumann (see Section 2.1.2) and Neumann-Neumann (see Section 2.1.3). The results are displayed in Table 3 and are quite interesting: even though the geometry is not properly approximated, velocity and pressure error norms are 0 up to machine precision in all cases. This can be explained by the facts that *(i)* the exact velocity is quadratic and the exact pressure linear, see (43), *(ii)* the geometric transformation is first order and all volume and surface numerical integrals are computed exactly thanks to FEEL++, hence the only possible solution in the velocity and pressure spaces are the exact velocity and exact pressure respectively which we see in the $L^2(\Omega_\delta)$ and $H^1(\Omega_\delta)$ error norms. These results are however somewhat deceptive as they are not taking into account the geometrical approximation as mentioned above. Indeed if we now look at the error in $\|\boldsymbol{F}_{ex} - \boldsymbol{F}_\delta\|$ it displays only an order 2 convergence rate with respect to h — as expected when using order 1 geometry — not withstanding the polynomial order of the velocity and pressure, see Table 3 and Figure 6.



(a) Dirichlet-Dirichlet (b) Dirichlet-Neumann (c) Neumann-Neumann

FIGURE 6. Output convergence using $\mathbb{P}_{N+1}\mathbb{P}_N\mathbb{G}_1$ approximation spaces, $N = 1, 2$

In order to improve the approximation properties of $\boldsymbol{F}_\delta$, we now turn to the second order geometry approximation. The results are displayed in Table 4. Again the results are interesting: even though the exact velocity and pressure are quadratic and linear respectively the finite element approximations are not exact. First recall that, see (41), it is not the finite element approximations in the real element that are polynomials of degree N but the finite element approximations in the reference element $\hat{K}$ and second that the geometric transformation is not longer linear, hence the numerical integrations are not exact as the integrands are no longer polynomial when derivatives are involved — thanks to the chain rule. — We recover however very good convergence rates and in fact we have super convergence — one order more than expected. — This is due to the symmetries of the cylinder. Finally, we plot the $\boldsymbol{F}_\delta$ convergence in Figure 7 for $\mathbb{P}_{N+1}\mathbb{P}_N\mathbb{G}_N$, $N = 1, 2$ as well as $\mathbb{P}_2\mathbb{P}_1\mathbb{G}_2$. The later case shows that increasing just the geometrical approximation improves already $\boldsymbol{F}_\delta$ tremendously. To summarize, to handle non linear geometries, *i.e.*$\Omega_\delta \neq \Omega$, we not only need to increase the order of approximations in velocity and pressure if we want to improve the accuracy of our simulations but we need also to increase the order of approximation of the geometry. This, of course, comes at a cost which the FEEL++ framework allows to alleviate to find a good balance between h, N and k_{geo}.

4.2. **Flow simulation on cerebrovenous system**

To exercise our full fledged framework, we perfom now a (incompressible) Navier-Stokes simulation in a realistic geometry, the cerebrovenous system. This geometry represented by Figure 2(a) has 29 inlets and 2 outlets. The boundary conditions imposed in this application are not physiological: on each inlet, we impose $\boldsymbol{\sigma}(\mathbf{u}, p)\mathbf{n} = g_{in}\mathbf{n}$ with

$$g_{in} = -0.5 \cdot 10^5 \left(1 - \cos\left(\frac{\pi t}{0.0015}\right)\right), \tag{45}$$

and at the outlets, we set : $\boldsymbol{\sigma}(\mathbf{u}, p)\mathbf{n} = \mathbf{0}$ which corresponds to the Neumann-Neumann formulation described in Section 2.1.3. Regarding the physical parameters, we take the density ρ equal to 1kg/m^3 and the dynamic viscosity μ equal $0.003\text{N}\cdot\text{s/m}^2$. The time step is $\Delta t = 10^{-5}\text{s}$ and we perform a simulation up to $t = 0.003\text{s}$. The approximation

(a) Dirichlet-Dirichlet: $\mathbb{P}_2\mathbb{P}_1\mathbb{G}_1$

h	$\|\mathbf{u}-\mathbf{u}_\delta\|_{0,\Omega_\delta}$	$\|p-p_\delta\|_{0,\Omega_\delta}$	$\|\mathbf{u}-\mathbf{u}_\delta\|_{1,\Omega_\delta}$	$\|\|\boldsymbol{F}_{ex}-\boldsymbol{F}_\delta\|\|$	Slope
0.5	$2.0\cdot10^{-16}$	$2.1\cdot10^{-15}$	$4.9\cdot10^{-16}$	$1.4\cdot10^{-1}$	
0.3	$1.2\cdot10^{-16}$	$3.6\cdot10^{-15}$	$5.5\cdot10^{-16}$	$5.1\cdot10^{-2}$	1.98
0.2	$3.3\cdot10^{-16}$	$1.3\cdot10^{-14}$	$8.4\cdot10^{-16}$	$2.6\cdot10^{-2}$	1.65
0.1	$1.3\cdot10^{-16}$	$1.5\cdot10^{-14}$	$1.7\cdot10^{-15}$	$5.7\cdot10^{-3}$	2.2

(b) Dirichlet-Dirichlet: $\mathbb{P}_3\mathbb{P}_2\mathbb{G}_1$

h	$\|\mathbf{u}-\mathbf{u}_\delta\|_{0,\Omega_\delta}$	$\|p-p_\delta\|_{0,\Omega_\delta}$	$\|\mathbf{u}-\mathbf{u}_\delta\|_{1,\Omega_\delta}$	$\|\|\boldsymbol{F}_{ex}-\boldsymbol{F}_\delta\|\|$	Slope
0.5	$1.5\cdot10^{-16}$	$8.4\cdot10^{-15}$	$1.1\cdot10^{-15}$	$1.4\cdot10^{-1}$	
0.3	$1.8\cdot10^{-16}$	$4.9\cdot10^{-15}$	$1.2\cdot10^{-15}$	$5.1\cdot10^{-2}$	1.98
0.2	$1.9\cdot10^{-16}$	$9.2\cdot10^{-15}$	$1.8\cdot10^{-15}$	$2.6\cdot10^{-2}$	1.65
0.1	$2.1\cdot10^{-15}$	$4.6\cdot10^{-14}$	$4.3\cdot10^{-15}$	$5.7\cdot10^{-3}$	2.2

(c) Dirichlet-Neumann: $\mathbb{P}_2\mathbb{P}_1\mathbb{G}_1$

h	$\|\mathbf{u}-\mathbf{u}_\delta\|_{0,\Omega_\delta}$	$\|p-p_\delta\|_{0,\Omega_\delta}$	$\|\mathbf{u}-\mathbf{u}_\delta\|_{1,\Omega_\delta}$	$\|\|\boldsymbol{F}_{ex}-\boldsymbol{F}_\delta\|\|$	Slope
0.5	$8.1\cdot10^{-17}$	$2.3\cdot10^{-15}$	$3.9\cdot10^{-16}$	$1.4\cdot10^{-1}$	
0.3	$8.4\cdot10^{-17}$	$1.4\cdot10^{-15}$	$5.1\cdot10^{-16}$	$5.1\cdot10^{-2}$	1.98
0.2	$1.7\cdot10^{-16}$	$2.0\cdot10^{-15}$	$8.5\cdot10^{-16}$	$2.6\cdot10^{-2}$	1.65
0.1	$1.6\cdot10^{-16}$	$3.9\cdot10^{-15}$	$1.7\cdot10^{-15}$	$5.7\cdot10^{-3}$	2.2

(d) Dirichlet-Neumann: $\mathbb{P}_3\mathbb{P}_2\mathbb{G}_1$

h	$\|\mathbf{u}-\mathbf{u}_\delta\|_{0,\Omega_\delta}$	$\|p-p_\delta\|_{0,\Omega_\delta}$	$\|\mathbf{u}-\mathbf{u}_\delta\|_{1,\Omega_\delta}$	$\|\|\boldsymbol{F}_{ex}-\boldsymbol{F}_\delta\|\|$	Slope
0.5	$2.0\cdot10^{-16}$	$5.4\cdot10^{-15}$	$1.1\cdot10^{-15}$	$1.4\cdot10^{-1}$	
0.3	$2.1\cdot10^{-16}$	$6.7\cdot10^{-15}$	$1.4\cdot10^{-15}$	$5.1\cdot10^{-2}$	1.98
0.2	$2.1\cdot10^{-16}$	$7.1\cdot10^{-15}$	$1.9\cdot10^{-15}$	$2.6\cdot10^{-2}$	1.65
0.1	$4.9\cdot10^{-16}$	$1.3\cdot10^{-14}$	$4.2\cdot10^{-15}$	$5.7\cdot10^{-3}$	2.2

(e) Neumann-Neumann: $\mathbb{P}_2\mathbb{P}_1\mathbb{G}_1$

h	$\|\mathbf{u}-\mathbf{u}_\delta\|_{0,\Omega_\delta}$	$\|p-p_\delta\|_{0,\Omega_\delta}$	$\|\mathbf{u}-\mathbf{u}_\delta\|_{1,\Omega_\delta}$	$\|\|\boldsymbol{F}_{ex}-\boldsymbol{F}_\delta\|\|$	Slope
0.5	$2.3\cdot10^{-16}$	$3.2\cdot10^{-15}$	$4.6\cdot10^{-16}$	$1.4\cdot10^{-1}$	
0.3	$1.8\cdot10^{-16}$	$3.1\cdot10^{-15}$	$5.5\cdot10^{-16}$	$5.1\cdot10^{-2}$	1.98
0.2	$1.9\cdot10^{-16}$	$2.7\cdot10^{-15}$	$8.3\cdot10^{-16}$	$2.6\cdot10^{-2}$	1.65
0.1	$1.7\cdot10^{-16}$	$3.6\cdot10^{-15}$	$1.5\cdot10^{-15}$	$5.7\cdot10^{-3}$	2.2

(f) Neumann-Neumann: $\mathbb{P}_3\mathbb{P}_2\mathbb{G}_1$

h	$\|\mathbf{u}-\mathbf{u}_\delta\|_{0,\Omega_\delta}$	$\|p-p_\delta\|_{0,\Omega_\delta}$	$\|\mathbf{u}-\mathbf{u}_\delta\|_{1,\Omega_\delta}$	$\|\|\boldsymbol{F}_{ex}-\boldsymbol{F}_\delta\|\|$	Slope
0.5	$2.2\cdot10^{-16}$	$4.2\cdot10^{-15}$	$1.0\cdot10^{-15}$	$1.4\cdot10^{-1}$	
0.3	$1.8\cdot10^{-16}$	$4.5\cdot10^{-15}$	$1.3\cdot10^{-15}$	$5.1\cdot10^{-2}$	1.98
0.2	$2.0\cdot10^{-16}$	$6.3\cdot10^{-15}$	$1.8\cdot10^{-15}$	$2.6\cdot10^{-2}$	1.65
0.1	$8.0\cdot10^{-16}$	$1.8\cdot10^{-14}$	$3.9\cdot10^{-15}$	$5.7\cdot10^{-3}$	2.2

TABLE 3. Dirichlet-Dirichlet, Dirichlet-Neumann and Neumann-Neumann formulations using first order geometry approximation

used is $\mathbb{P}_2\mathbb{P}_1\mathbb{G}_1$. Table 5 shows the number of tetrahedrons in the mesh as well as the number of degrees of freedom associated to the $\mathbb{P}_2\mathbb{P}_1\mathbb{G}_1$ approximation.

As a verification, we monitor the sum of the flow rates D_{in} at the inlets and the sum of the flow rates D_{out} at the outlets. Since the fluid is incompressible, $|D_{in}+D_{out}|$ must be zero. In our simulations, this estimate varies between

(a) Dirichlet-Dirichlet $\mathbb{P}_3\mathbb{P}_2\mathbb{G}_2$

h	$\|\mathbf{u}-\mathbf{u}_\delta\|_{0,\Omega_\delta}$	$\|p-p_\delta\|_{0,\Omega_\delta}$	SlopeP	$\|\mathbf{u}-\mathbf{u}_\delta\|_{1,\Omega_\delta}$	SlopeU	$\|\|\boldsymbol{F}_{ex}-\boldsymbol{F}_\delta\|\|$	Slope
0.5	$6.3\cdot10^{-6}$	$5.8\cdot10^{-5}$		$2.2\cdot10^{-5}$		$4.6\cdot10^{-4}$	
0.3	$6.2\cdot10^{-7}$	$8.5\cdot10^{-6}$	3.76	$2.5\cdot10^{-6}$	4.28	$5.7\cdot10^{-5}$	4.1
0.2	$1.2\cdot10^{-7}$	$2.1\cdot10^{-6}$	3.48	$8.9\cdot10^{-7}$	2.52	$1.5\cdot10^{-5}$	3.31
0.1	$6.4\cdot10^{-9}$	$1.4\cdot10^{-7}$	3.88	$7.0\cdot10^{-8}$	3.67	$6.2\cdot10^{-7}$	4.59

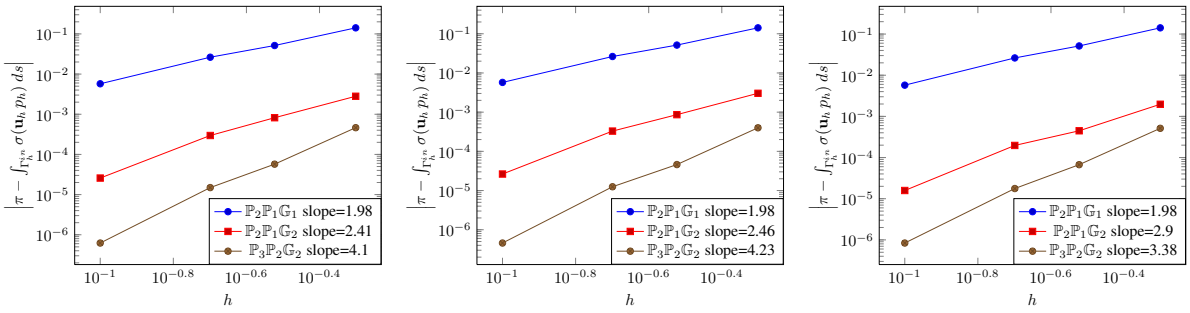
(b) Dirichlet-Neumann $\mathbb{P}_3\mathbb{P}_2\mathbb{G}_2$

h	$\|\mathbf{u}-\mathbf{u}_\delta\|_{0,\Omega_\delta}$	$\|p-p_\delta\|_{0,\Omega_\delta}$	SlopeP	$\|\mathbf{u}-\mathbf{u}_\delta\|_{1,\Omega_\delta}$	SlopeU	$\|\|\boldsymbol{F}_{ex}-\boldsymbol{F}_\delta\|\|$	Slope
0.5	$6.3\cdot10^{-6}$	$1.0\cdot10^{-4}$		$2.2\cdot10^{-5}$		$4.0\cdot10^{-4}$	
0.3	$6.4\cdot10^{-7}$	$1.7\cdot10^{-5}$	3.55	$2.3\cdot10^{-6}$	4.36	$4.6\cdot10^{-5}$	4.23
0.2	$1.3\cdot10^{-7}$	$3.9\cdot10^{-6}$	3.6	$8.8\cdot10^{-7}$	2.4	$1.2\cdot10^{-5}$	3.21
0.1	$6.8\cdot10^{-9}$	$2.6\cdot10^{-7}$	3.91	$7.1\cdot10^{-8}$	3.62	$4.6\cdot10^{-7}$	4.77

(c) Neumann-Neumann $\mathbb{P}_3\mathbb{P}_2\mathbb{G}_2$

h	$\|\mathbf{u}-\mathbf{u}_\delta\|_{0,\Omega_\delta}$	$\|p-p_\delta\|_{0,\Omega_\delta}$	SlopeP	$\|\mathbf{u}-\mathbf{u}_\delta\|_{1,\Omega_\delta}$	SlopeU	$\|\|\boldsymbol{F}_{ex}-\boldsymbol{F}_\delta\|\|$	Slope
0.5	$8.9\cdot10^{-6}$	$5.2\cdot10^{-5}$		$2.3\cdot10^{-5}$		$5.1\cdot10^{-4}$	
0.3	$1.1\cdot10^{-6}$	$4.3\cdot10^{-6}$	4.85	$2.4\cdot10^{-6}$	4.45	$6.7\cdot10^{-5}$	3.98
0.2	$2.3\cdot10^{-7}$	$1.3\cdot10^{-6}$	2.92	$8.9\cdot10^{-7}$	2.45	$1.8\cdot10^{-5}$	3.29
0.1	$1.2\cdot10^{-8}$	$7.0\cdot10^{-8}$	4.25	$7.1\cdot10^{-8}$	3.65	$8.4\cdot10^{-7}$	4.4

TABLE 4. Dirichlet-Dirichlet, Dirichlet-Neumann and Neumann-Neumann formulations using second order geometry approximation



(a) Dirichlet-Dirichlet test case (b) Dirichlet-Neumann test case (c) Neumann-Neumann test case

FIGURE 7. $\boldsymbol{F}_\delta$ convergence using $\mathbb{P}_{N+1}\mathbb{P}_N G_N$, $N=1,2$ approximations and $\mathbb{P}_2\mathbb{P}_1 G_2$ approximation.

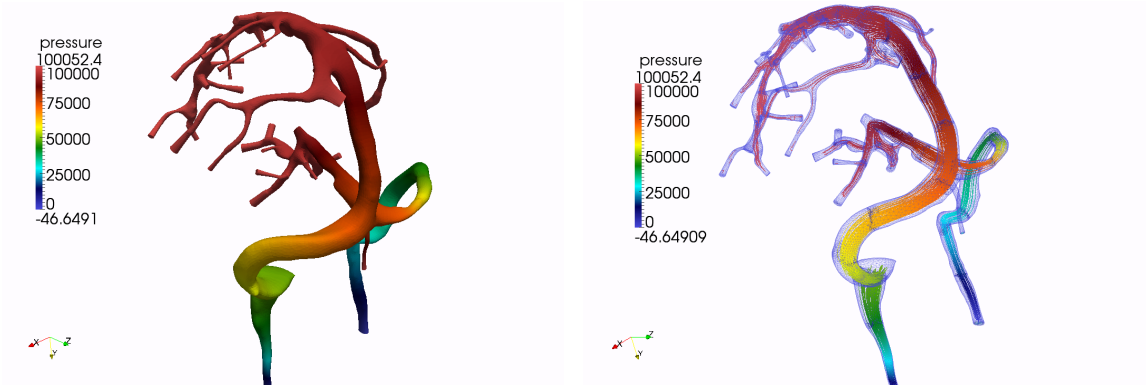
TETRAHEDRONS	DOF($\mathbf{u}_\delta$)	DOF(p_δ)	DOF(Total)
237,438	1,119,411	54,183	1,173,594

TABLE 5. Number of elements and degrees of freedom using $\mathbb{P}_2\mathbb{P}_1\mathbb{G}_1$ approximations.

$1e^{-10}$ and $1e^{-13}$. Finally, the Figure 4.2 displays two screenshots of a solution at time $t = 0.00151$s computed with 32 processors.

5. CONCLUSIONS AND OUTLOOK

In this paper we have proposed a flexible framework to answer some modeling and computational issues in order to perform large-scale three-dimensional blood flow simulations in realistic geometries. In particular, we have presented different strategies to handle boundary conditions appropriate to blood flow simulation as well as their variational



(a) Pressure field (b) Streamlines colored with pressure

FIGURE 8. Numerical solution on the cerebrovenous system at time $t = 0.00151$s.

formulation. While this work is a preliminary step, the results obtained with respect to convergence rates for both velocity and pressure for the Stokes problem are encouraging but require further investigation, for example in the case of mixed boundary conditions. We have also obtained good scalability results indicating that the strategy considered should now be taken to the next level and tested on hundreds or thousands of processors. Finally, we displayed initial numerical results on a realistic geometry which now need to be backed up by further mathematical and bio-mechanical modeling.

ACKNOWLEDGEMENTS

Vincent Chabannes and Christophe Prud'homme acknowledge the financial support of the Région Rhône-Alpes through the project ISLE/CHPID. Abdoulaye Samake and Christophe Prud'homme acknowledge the financial support of the project ANR HAMM ANR-2010-COSI-009. Gonçalo Pena acknowledges the financial support of the *Centro de Matemática da Universidade de Coimbra* (CMUC), funded by the European Regional Development Fund through the program COMPETE and by the Portuguese Government through the FCT - *Fundação para a Ciência e a Tecnologia* under the project PEst-C/MAT/UI0324/2011. Marcela Szopos and Ranine Tarabay acknowledge the financial support of the University of Strasbourg. Céline Caldini-Queiros acknowledges the financial support of the *Laboratoire de Mathématiques de Besançon* (LMB). Finally the authors wish to thank the Cemracs 2012 and its organizers.

REFERENCES

[1] Patrick R. Amestoy, Abdou Guermouche, and Stphane Pralet. Hybrid scheduling for the parallel solution of linear systems. *Parallel Computing*, 32:136–156, 2006.

[2] C. Amrouche and N. Seloula. Stokes equations and elliptic systems with non standard boundary conditions. *C. R. Acad. Sci. Paris, Ser. I*, 340, 2010.

[3] Satish Balay, Jed Brown, Kris Buschelman, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc Web page. http://www.mcs.anl.gov/petsc, 2012.

[4] Satish Balay, Kris Buschelman, Victor Eijkhout, William Gropp, Dinesh Kaushik, Matt Knepley, Lois C. Mcinnes, Barry Smith, and Hong Zhang. PETSc Users Manual, 2012.

[5] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. Efficient management of parallelism in object-oriented numerical software libraries. In *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhauser Press, 1997.

[6] Vincent Chabannes. *Vers la Simulation des Écoulements Sanguins*. PhD thesis, 2013.

[7] Vincent Chabannes, Gonçalo Pena, and Christophe Prud'homme. High-order fluid-structure interaction in 2D and 3D: Application to blood flow in arteries. *Journal of Computational and Applied Mathematics*, 2012. accepted for publication.

[8] C. Conca, F. Murat, and O. Pironneau. The Stokes and Navier-Stokes equations with boundary conditions involving the pressure. *Japan. J. Math.*, 20(2):279–318, 1994.

[9] V. Doyeux, Y. Guyot, V. Chabannes, C. Prud'homme, and M. Ismail. Simulation of two-fluid flows using a finite element/level set method. application to bubbles and vesicle dynamics. *Journal of Computational and Applied Mathematics*, 2012. accepted for publication.

[10] Luca Formaggia, Alfio Quarteroni, and Alessandro Veneziani, editors. *Cardiovascular mathematics*, volume 1 of *MS&A. Modeling, Simulation and Applications*. Springer-Verlag Italia, Milan, 2009. Modeling and simulation of the circulatory system.

[11] C. Geuzaine and J.-F. Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre-and post-processing facilities. *International Journal for Numerical Methods*, 79(11):1309–1331, 2009.

[12] E. Marchandise, C. Geuzaine, and J.F. Remacle. Cardiovascular and lung mesh generation based on centerlines. *International Journal for Numerical Methods in Biomedical Engineering*. submitted in 2013.
[13] B. Maury. *The Respiratory System in Equations*. Springer, 2013.
[14] Gonçalo Pena, Christophe Prud'homme, and Alfio Quarteroni. High Order Methods for the Approximation of the Incompressible Navier-Stokes Equations in a Moving Domain. *Computer Methods in Applied Mechanics and Engineering*, 209-212:197–211, 2012.
[15] Christophe Prud'homme. Life: Overview of a unified C++ implementation of the finite and spectral element methods in 1D, 2D and 3D. In *Workshop On State-Of-The-Art In Scientific And Parallel Computing, Lecture Notes in Computer Science*, page 10. Springer-Verlag, 2007.
[16] Christophe Prudhomme, Vincent Chabannes, Vincent Doyeux, Mourad Ismail, Abdoulaye Samake, and Gonçalo Pena. Feel++: A Computational Framework for Galerkin Methods and Advanced Numerical Methods. In *ESAIM Proc., CEMRACS'11: Multiscale Coupling of Complex Models in Scientific Computing*, volume 38, pages 429–455, December 2012.
[17] Salmon, Stéphanie, Sy, Soyibou, and Szopos, Marcela. Cerebral blood flow simulations in realistic geometries. *ESAIM: Proc.*, 35:281–286, 2012.
[18] B. Smith, P. Bjorstad, and W. Gropp. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, 1996.