Toward a Real Time, Image Based CFD

Marc Garbey and Bilel Hadri

Computer Science Department, University of Houston.
{garbey,hadri}@cs.uh.edu

Summary. We present a method to combine fluid dynamics and image analysis into a single fast simulation environment. Our target applications are hemodynamic studies. Our method combines an NS solver that relies on the L_2 penalty approach pioneered by Caltagirone and co-workers, and a level set method based on the Mumford-Shah energy model. Working in Cartesian coordinates regardless of grid no matter the complexity of the geometry, one can use fast parallel domain decomposition solvers in a fairly robust and consistent way. The input of the simulation tool is a set of JPEG images, and the output can be various flow components as well as shear stress indicators on the vessel or domain wall. In two space dimensions the code runs close to real time.

1 Introduction and Motivation

The objective of this work is to use angiogram medical images to produce flow simulations by a very robust and fast method. The emphasis is not on high accuracy, since there are many sources of errors or variability in medical data. From medical imaging, we extract the geometry of large vessels. Our algorithm provides a first order approximation of some main quantities of interest in cardiovascular disease: the shear stress and the pressure on the wall, as well as the flow components in the artery.

We present a fast, versatile and robust NS solver that relies heavily on the L_2 penalty approach pioneered by Caltagirone and co-workers [2] and combines nicely with a level set method based on the Mumford-Shah energy model [4].

The wall boundary condition is immersed in the Cartesian mesh thanks to the penalty term added to the momentum equation. We use the domain decomposition (DD) algorithm of [3] that has high numerical efficiency and scales well with parallel computers in order to take full advantage of the regular data structure of the problem. This DD is coupled with a sub-domain solver that is tuned to provide the fastest result on the computer available for the run. In this paper we present simulations in two space dimensions while results in three space dimensions will be reported elsewhere.

2 Navier-Stokes Flow Solver

Since we concentrate our study on large vessels, we use an incompressible NS fluid flow model [8, 11].

In this paper we will use the penalty method introduced by Caltagirone and co-workers [2] since it is simpler to implement than our previous boundary fitted methods [7] and applies naturally to flow in complex domains with moving walls [10].

The flow of incompressible fluid in a rectangular domain $\Omega = (0, L_x) \times (0, L_y)$ with prescribed values of the velocity on $\partial\Omega$ obeys the NS equations:

$$\partial_t U + (U.\nabla)U + \nabla p - \nu \nabla.(\nabla U) = f, \text{ in } \Omega$$

$$div(U) \; = \; 0, \text{ in } \Omega, U = g \text{ on } \partial\Omega,$$

We denote by $U(x, y, t)$ the velocity with components (u_1, u_2) and by $p(x, y, t)$ the normalized pressure of the fluid. ν is a kinematic viscosity.

With an immersed boundary approach the domain Ω is decomposed into a fluid subdomain Ω_f and a wall subdomain Ω_w. In the L_2 penalty method the right hand side f is a forcing term that contains a mask function Λ_{Ω_w}

$$\Lambda_{\Omega_w}(x, y) = 1, \text{ if } (x, y) \in \Omega_w, \text{ 0 elsewhere,}$$

and is defined as follows

$$f = -\frac{1}{\eta} \Lambda_{\Omega_w} \; \{U - U_w(t)\}.$$

U_w is the velocity of the moving wall and η is a small positive parameter that goes to zero.

A formal asymptotic analysis helps us to understand how the penalty method matches the no slip boundary condition on the interface $S^f_w = \bar{\Omega}_f \bigcap \bar{\Omega}_w$ as $\eta \rightarrow 0$. Let us define the following expansion:

$$U = U_0 \; + \; \eta \, U_1, \; p \; = p_0 \; + \; \eta \, p_1.$$

Formally, in first order, we obtain,

$$\frac{1}{\eta} \Lambda_{\Omega_w} \; \{U_0 - U_w(t)\} = 0,$$

that is

$$U_0 = U_w, \text{ for } (x, y) \in \Omega_w.$$

The leading order terms U_0 and p_0 in the fluid domain Ω_f satisfy the standard set of NS equations:

$$\partial_t U_0 + (U_0.\nabla)U_0 + \nabla p_0 - \nu \nabla.(\nabla U_0) = 0, \text{ in } \Omega_f$$

$$div(U_0) \; = \; 0, \text{ in } \Omega.$$

At the next order we have in Ω_w,

$$\nabla p_0 \; + \; U_1 \; + Q_w = 0, \tag{1}$$

where

$$Q_w \;=\; \partial_t U_w + (U_w.\nabla)U_w - \nu\nabla.(\nabla U_w).$$

Further the wall motion U_w must be divergence free. Continuing to the next order we have in Ω_f,

$$\partial_t U_1 + (U_0.\nabla)U_1 + (U_1.\nabla)U_0 + \nabla p_1 - \nu\nabla.(\nabla U_1) = 0,$$

with

$$div(U_1) = 0.$$

In the simplest situation where $U_w \equiv 0$, we observe that the motion of the flow is driven by the pressure following a classical Darcy law. η stands for a small permeability. To summarize as $\eta \to 0$, the flow evolution is dominated by the NS equations in the artery, and by the Darcy law with very small permeability in the wall. This actually corresponds to a standard multiscale model of blood flow in the main arteries. From the analytical point of view it was shown in [1] for a fixed wall, i.e. $U_w \equiv 0$, that the convergence order of the penalty method is of order $\eta^{\frac{3}{4}}$, in the fluid domain, and $\eta^{\frac{1}{4}}$ in the wall.

The mask function Λ_{Ω_w} is obtained with an image segmentation technique that is a level set method. Since the contours of the image are not necessarily sharp, it is interesting to use the level set method presented in [4] and based on the Mumford-Shah Model.

Regarding the resolution of the equation, we use a projection method for the time step as follows :

- Step 1: prediction of the velocity $\hat{u}^{k+1}$ by solving either :

$$\frac{\hat{u}^{k+1} - u^{k,*}}{\Delta t} - \nu\Delta u^k = f^{k+1} - \nabla p^k \; or; \frac{\hat{u}^{k+1} - u^{k,*}}{\Delta t} - \nu\Delta u^{k+1} = f^{k+1} - \nabla p^k$$

  in $(0, L_x) \times (0, L_y)$ with the boundary condition $\hat{u}^{k+1} = g$ on $\partial\Omega$. We denote that $u^{k,*}$ is obtained thanks to the method of characteristics.
- Step 2: projection of the predicted velocity to the space of divergence free functions.

$$-div\nabla\delta p = -\frac{1}{\Delta t} div\hat{u}^{k+1}; u^{k+1} = \hat{u}^{k+1} - \Delta t\delta p$$

$$p^{k+1} = p^k + \delta p\,.$$

The NS calculation decomposes into three steps: the prediction of the flow speed components, the solution of a Poisson problem for the pressure, and eventually the computation of the shear stress along the wall. The momentum equations can be solved quickly, while the performance of the code is dominated by the Poisson solver for the pressure. We detail this part of the algorithm in the next section.

3 Multi-Algorithm for the Pressure Solver

The pressure equation can be integrated with a number of existing fast Poisson solvers since the discretization grid is regular. It is convenient for example to use a (full) multigrid solver here. The arithmetic complexity of this solver is optimum. Further the iterative solver converges extremely fast for those grid points that are in the solid wall. However the pressure equation has to be solved at every time step and what turns out to be the faster solver may depend on the computer architecture. In the following we use the framework of the Aitken- Additive Schwarz method to fine tune the subdomain solver [5, 6]. The main reason being that on a standard Beowulf system with fast ethernet switch, the high latency of the network significantly lowers the performance of the multigrid solver. On the contrary the Aitken-Schwarz algorithm may double the number of flops compared to an optimal solver but is highly insensitive to the latency of the network.

Interface software [6] has been written to reuse a broad variety of existing linear algebra software for each subdomain such as LU factorization, a large number of Krylov methods with incomplete LU preconditioner, and geometric or algebraic multigrid solvers.

Only experiments can provide the fastest method for the resolution of a linear system.

Let us restrict ourselves to three software systems: Linpack for LU, Sparskit for iterative solver, Hypre for algebraic multigrid. We build a surface response model [9] based upon the least square quadratic polynomial approximation of the elapsed time as a function of the grid size (n_x, n_y):

$$T(n_x, n_y) = \beta_0 + \beta_1 n_x + \beta_2 n_y + \beta_3 n_x n_y + \beta_4 n_x^2 + \beta_5 n_y^2 \,.$$

The performance modeling can be done in principle with any linear algebra software. This model to predict the elapsed time for the resolution of a linear solver with LU or a Krylov solver with a relative prediction error of the prediction less than a few percent. To build the model we need on the order of 10 test runs with various grid configurations that cover the region of prediction. For Hypre, this model does not give good predictions in general. One observes that the elapsed time is very sensitive to the size.

Based on the surface response model, one can then decide what is the optimum solver for a given subdomain dimension. For illustration purpose let us restrict ourselves to the 2D Poisson problem that corresponds to the pressure solver. One can notice that LU performs well for small sizes, while, iterative solvers such as BiCGStab (Krylov method) and AMG-GMRES (multigrid method) give the fastest results for large grid sizes. More details on this study can be found in [6].

Figures 1 and 2 give the performances on different processor architectures, a dual processor AMD 1800+ with 2GB of RAM and a dual processor 900 MHz Itanium2 with 3 GB of RAM, respectively. We clearly see the difference of the region of the area where LU is faster than the iterative method. The automatic tuning through the model of the solver helps us to choose wisely the fastest solver for each subdomain solver.

It should be observed from Figures 3 and 4 that the optimum choice of the subdomain solver depends weakly on the number of processors. In the framework of the AS algorithm, we have observed that message passing favors an iterative solver versus a direct solver when the difference on performance between the two solvers is moderate.

Let us now illustrate our parallel algorithm on the incompressible Navier-Stokes flow that uses the pressure solver of this section.

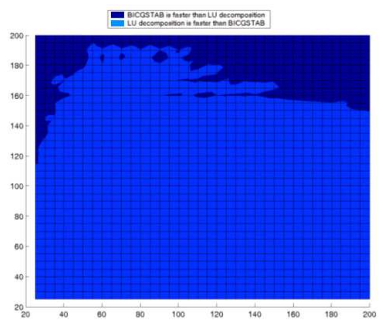


Fig. 1. Comparison between BiCGStab and LU decomposition on a 32bit AMD processor

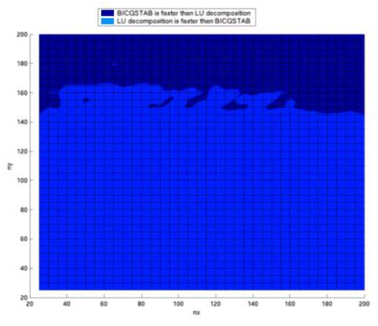


Fig. 2. Comparison between BiCGStab and LU decomposition on a 64 bit Itanium

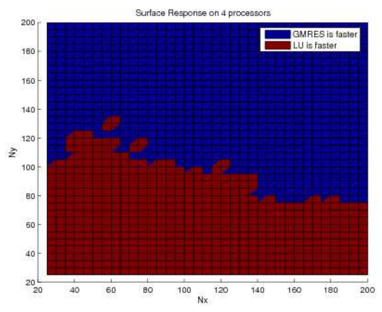


Fig. 3. Surface response with 4 processors

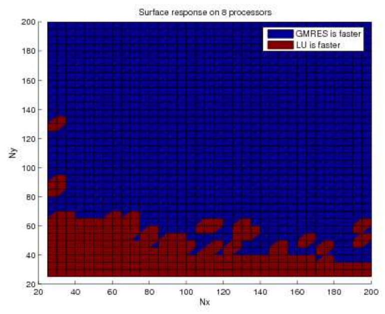


Fig. 4. Surface response with 8 processors

4 Parallel Performances of the NS Code

Let us first present a two dimensional simulation obtained from an x-ray. Figure 5 shows the frontal projection of a carotid in the brain area during an angiogram procedure. Figure 6 shows an example for a steady flow calculation in the region of interest with a Reynolds number of order 330. The flow comes from the left side. The size of the grid in this simulation is about 210×170.

The simulation of one cardiac cycle with an 8-way Opteron machine, for the grid sizes, 300×100 and 450×150, takes around 3 and 9 seconds, respectively. The time for the image segmentation is less. These simulation are then close to real time.

Figure 7 shows the speedup of this code with different grid sizes. Until four processors, the code has a linear speedup, and then the speed up deteriorates.

Actually this is mainly due to the design of the crossbar architecture of the low cost Opteron system which does not scale from 4 to 8 processors.

The results on scalability of our code are better.

In figure 8 we keep the aspect ratio of the grid $\frac{h_x}{h_y}$ the same: three tests have been performed respectively with a problem of size 141×567 for two processors, 200×801

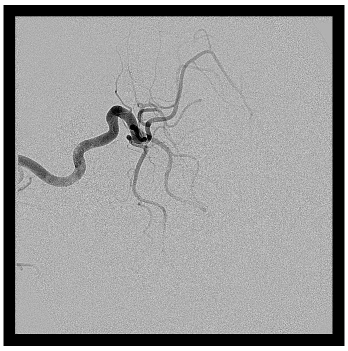


Fig. 5. Benchmark problem

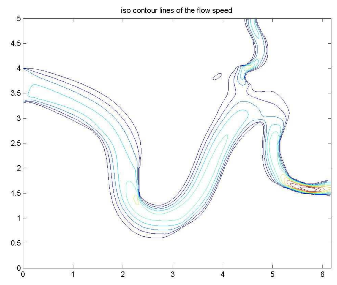


Fig. 6. Contour u

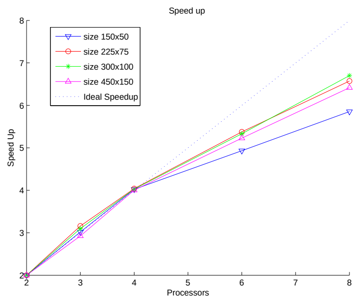
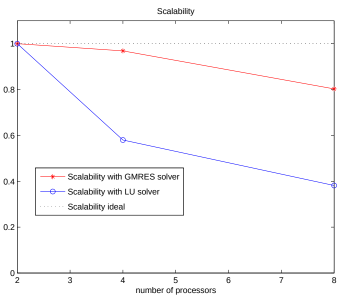


Fig. 7. Speedup of the Navier-Stokes code.



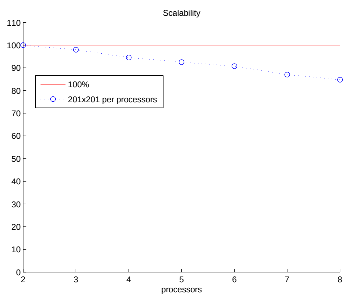


Fig. 8. Scalability of the Navier-Stokes code

Fig. 9. Scalability of the Navier-Stokes code with LU solver

on 4 processors, and 283×1129 on 8 processors. GMRES gives a better scalability result compared to LU because the growth of the bandwidth of the matrix as the number of subdomain increases penalize the LU solver. On the contrary if we keep the size of the subdomain fixed, here 201×201 in figure 9, we obtain a very good scalability of the code no matter the subdomain solver.

5 Conclusion

We have presented an image based CFD algorithm designed for hemodynamic simulation. In two space dimension we obtain a code that can be easily optimized for a specific parallel computer architecture. Robustness and simplicity of the solver are key elements to make the simulation applicable to clinical conditions. We have developed recently a three dimension version of the method that will be reported elsewhere. Our technique may not be appropriate for turbulent flow for high Reynolds numbers, but there are a number of cardiovascular problems that corresponds to unsteady situations with relatively modest Reynolds numbers [8, 11].

Acknowledgement. This research was partially supported by Awards 0305405 and 0521527 from the National Science Foundation.

References

[1] P. Angot, C.H. Bruneau, and P. Fabrie. A penalisation method to take into account obstacles in viscous flows. *Numer. Math.*, 81(4):497–520, 1999.
[2] E. Arquis and J.P. Caltagirone. Sur les conditions hydrodynamiques au voisinage d'une interface milieu fluide-milieux poreux: Application à la convection naturelle. *C. R. Acad. Sci. Paris Sér. II*, 299:1–4, 1984.
[3] N. Barberou, M. Garbey, M. Hess, M. Resch, T. Rossi, J. Toivanen, and D. Tromeur Dervout. Efficient metacomputing of elliptic linear and non-linear problems. *J. Parallel Distrib. Comput.*, 63:564–577, 2003.
[4] T.F. Chan and L.A. Vese. Active contours without edges. *IEE Transaction on Image Processing*, 10 (2):266–277, 2001.
[5] M. Garbey and D. Tromeur Dervout. On some Aitken like acceleration of the Schwarz method. *Internat. J. Numer. Methods Fluids*, 40:1493–1513, 2002.
[6] M. Garbey, W. Shyy, B. Hadri, and E. Rougetet. Efficient solution techniques for CFD and heat transfer. In *ASME Heat Transfer/Fluids Engineering Summer Conference*, 2004.
[7] M. Garbey and Yu.V. Vassilevski. A parallel solver for unsteady incompressible 3D Navier-Stokes equations. *Parallel Comput.*, 27(4):363–389, 2001.
[8] D.A. McDonald. *Bloof Flow in Arteries.* Edward Arnold, 3rd edition, 1990.
[9] D.C. Montgomery and R.H. Myers. *Response Surface Methodology: Process and Product Optimization Using Designed Experiments.* Wiley, 2nd edition, 2002.
[10] K. Schneider and M. Farge. Numerical simulation of the transient flow behaviour in tube bundles using a volume penalisation method. *J. of Fluids and Structures*, 20(4):555–566, 2005.
[11] X.Y. Xu, M.W. Collins, and C.J.H. Jones. Flow studies in canine aortas. *ASME J. Biomech. Engrg.*, 114(11):505–511, 1992.