COMMUNICATIONS IN COMPUTATIONAL PHYSICS
Vol. **4**, No. 5, pp. 1151-1169

Commun. Comput. Phys.
November 2008

A Scalable Domain Decomposition Method for Ultra-Parallel Arterial Flow Simulations[†]

Leopold Grinberg and George Em Karniadakis[*]

Division of Applied Mathematics, Brown University, Providence 02912, USA.

Received 18 March 2008; Accepted (in revised version) 10 July 2008

Available online 9 September 2008

Abstract. Ultra-parallel flow simulations on hundreds of thousands of processors require new multi-level domain decomposition methods. Here we present such a new two-level method that has features both of discontinuous and continuous Galerkin formulations. Specifically, at the coarse level the domain is subdivided into several big patches and within each patch a spectral element discretization (fine level) is employed. New interface conditions for the Navier-Stokes equations are developed to connect the patches, relaxing the C^0 continuity and minimizing data transfer at the patch interface. We perform several 3D flow simulations of a benchmark problem and of arterial flows to evaluate the performance of the new method and investigate its accuracy.

AMS subject classifications: 52B10, 65D18, 68U05, 68U07

Key words: Bioflows, spectral elements, discontinuous Galerkin, parallel computing.

1 Introduction

Current and projected advances in computer architectures involving hundreds of thousands of processors cannot be exploited for large-scale simulations of the human arterial tree [1, 2] (or of many other physical and biological problems) based on existing domain decomposition algorithms and corresponding parallel paradigms. Not only we have to address the tremendous complexity associated with data transfer amongst thousands of processors, but more fundamentally the solution of linear systems with billions degrees of freedom (DOFs) and corresponding condition number exceeding one million is a rather formidable task.

In this paper we develop a significant extension of the spectral/*hp* element method (SEM) for large-scale simulation of arterial blood flow dynamics. In particular, we adopt

[†]Dedicated to Professor Xiantu He on the occasion of his 70th birthday.
[*]Corresponding author. *Email addresses:* `lgrinb@dam.brown.edu` (L. Grinberg), `gk@cfm.brown.edu` (G. E. Karniadakis)

two levels of discretization by introducing coarse-level patches to decompose the computational domain. SEM, similarly to finite element method, is based on discretization of the computational domain into non-overlapping elements. Within each element the solution is approximated with a high-order (spectral) polynomial expansion. The total number of DOFs depends on the number of elements and the order of polynomial expansion within each element. The two common approaches for solution of partial differential equations with SEM are [3]: (a) Discontinuous Galerkin method (DG), where discontinuity of the numerical solution at the interfaces of elements is allowed; and (b) Continuous Galerkin method, where the boundary degrees of freedom defined at the interfaces of elements are shared, hence enforces C^0 continuity of the numerical solution. In the C^0 approximation global linear operators are constructed from the local ones by static condensation and due to sharing of the boundary degrees of freedom the rank of the global operator is lower than the total number of local DOFs.

In 3D large-scale simulations, the number of spectral elements can be well over a million, and due to the high-order polynomial expansion the number of DOFs may be over several billions. For example, the aorta domain in Fig. 1 has 325,795 tetrahedral elements and includes only 17 arteries while a domain to discretize 65 major cranial arteries [4] has 459,250 tetrahedral elements. To resolve the complex patterns of unsteady blood flow such as secondary flows, turbulence and recirculation, high-order spatial resolution is required. Hence, in the aorta domain employing sixth-order polynomial expansions leads to 187,657,920 number of unknowns per variable[§] while in the cranial domain it leads to 264,528,000 number of unknowns. In bigger domains with 10 millions elements and sixth-order polynomial approximation the number of unknowns for each variable would be 5.76 billions or more than 20 billions DOFs for all four variables (3D velocity vector plus pressure).

This large number of unknowns leads to construction of a global linear operator matrix with very high rank and consequently with very large condition number. Decoupling of the interior degrees of freedom by applying Schur decomposition leads to reduction in the size of the linear operator that must be inverted, however, the rank of the Schur complement is still very large. In the current study we use the parallel solver NEKTAR [5], which employs a Preconditioned Conjugate Gradient (PCG) algorithm to solve the four linear systems for the velocity and pressure. Among the different preconditioners we have tested for parallel computations, the so-called Low Energy Basis Preconditioner (LEBP) [6–8] is the most effective. In Fig. 2 we plot the performance of NEKTAR (using LEBP) on the CRAY XT3 for a simulation involving 120,813 elements. The scaling is favorable for high-order polynomial approximation. In the parallel LEBP the coarse linear vertex preconditioner is implemented in two steps: In the first step, the global operator constructed from the linear (vertex) modes, which are shared by different partitions, is constructed and inverted in parallel. In the second step, the local operator constructed from linear modes within each partition is inverted. The size of the global operator is

[§]Here we define the number of unknowns as the number of quadrature points required for exact integration of the linear terms in the Navier-Stokes equation.

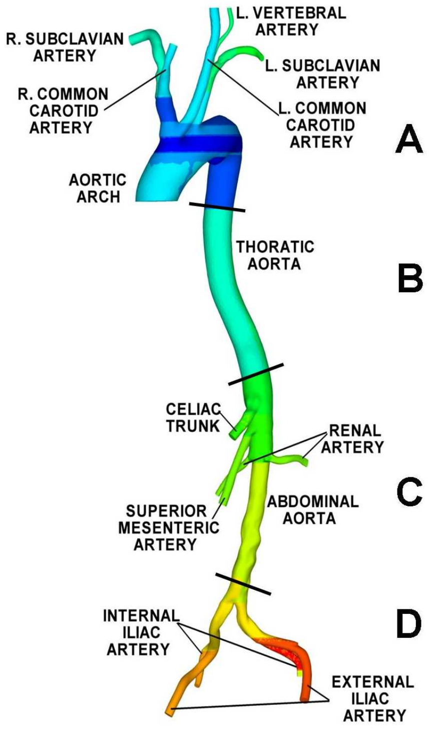

Figure 1: Computational domain of aorta consisting of 325,795 tetrahedral elements with edge length of 0.4 to 1.5 mm. The geometry obtained from CT images; colors represent different patches of the domain.

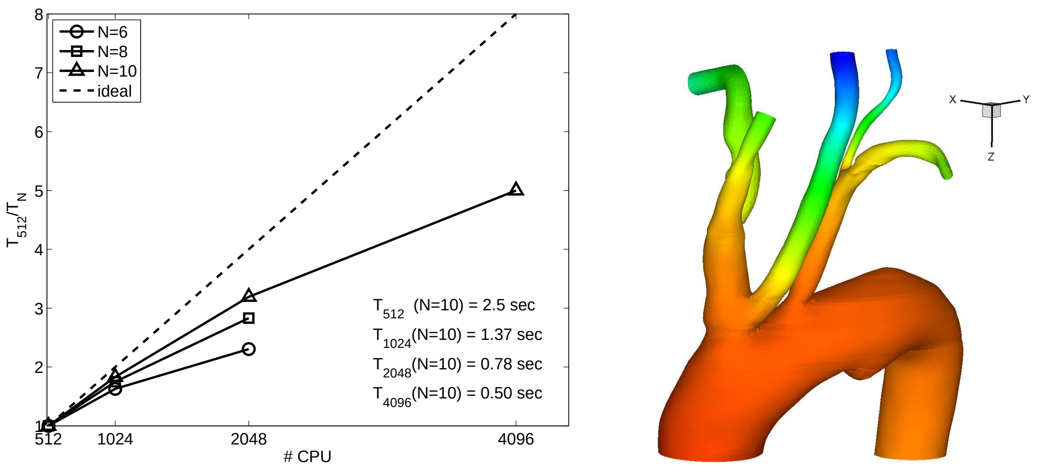

Figure 2: Performance of NEKTAR with LEBP: Left: Parallel speed-up. Problem size: 120,813 tetrahedral elements with 6th (circle), 8th (square) and 10th (triangle) polynomial-order approximation. The table shows the mean cpu-time per time step for solution of a problem with polynomial order $P=10$ and 226,161,936 unknowns. Right: geometry of the computational domain; color corresponds to pressure values. Computation performed on CRAY XT3 (ERDC) and CRAY XT3 (PSC). The result for 4096 CPUs was obtained on the CRAY XT3 of PSC, which is about 5-10% slower than the CRAY XT3 of ERDC.

increasing with the number of partitions, and this imposes two problems: (1) Inversion of a full matrix with large rank. For example, in a domain of 162,909 tetrahedral elements the rank of the coarse linear vertex operator for the pressure solver is 1000 when 256 processes are used and almost 27,000 when the domain is sub-divided (using METIS [9]) onto 3072 partitions. It grows almost linearly with an increase in the number of processes. (2) The volume of computation performed by each process decreases relatively to the volume of communication between processes required by parallel matrix vector multiply; hence the parallel efficiency of the preconditioner degrades. This last point is illustrated in Fig. 2, which shows the degradation in parallel efficiency of the iterative solver as the number of processors increases.

To overcome the aforementioned problems, we propose to decompose the arterial network into a series of *weakly coupled* sub-domains or patches of *manageable* size for which high parallel efficiency can be achieved on a sub-cluster of processors. The continuity of numerical solution across different patches is achieved by providing appropriate interface conditions, based on a *hybrid* C^0-DG-like approach. The method preserves the advantages of C^0 discretization, namely low number of DOFs compared to a full DG discretization and implicit treatment of the viscous terms of Navier-Stokes equation within a patch. The entire simulation can be performed in two levels: (i) in the inner level, we solve a series of tightly coupled problems in each sub-domain using semi-implicit time stepping scheme and C^0 polynomial approximation, and (ii) in the outer level, we explicitly solve the weakly coupled problems by imposing interface boundary conditions on the sub-domains interfaces with a DG-like approach.

The paper is organized as follows: In Section 2 we briefly overview the numerical method for spatial and temporal discretization and present the interface conditions. In Section 3 we present results of numerical simulations using the two-level approach and investigate its accuracy in solving steady and unsteady flow problems. Finally, in Section 4 we conclude with a summary.

2 The new method

The numerical approach we propose is based on a two-level domain decomposition (2DD), where the large computational domain is initially decomposed into several patches (coarse level) and SEM domain decomposition (1DD) is applied within each patch. For example, in the illustration of Fig. 1 the computational domain is subdivided into four non-overlapping patches A-D, with the thick black lines depicting the location of the patch interfaces; here we denote the interface between two patches by Γ. At every time step the tightly coupled problem defined within each patch is solved using the numerical scheme presented in the Section 2.1. The boundary conditions at the interface are provided by exchanging values of the numerical solution across Γ as we explain in Section 2.2.

2.1 Discretization within a patch

We use SEM discretization within a patch implemented in our code NEKTAR [5]. The computational domain used by NEKTAR consists of polymorphic elements e.g., tetrahedra, hexahedra, prisms, pyramids or a combination of these. Within each element the solution is approximated in terms of hierarchical, mixed-order, semi-orthogonal Jacobi polynomial expansions [3]. They are hierarchical in a sense that the modes are separated into vertex (linear term) Φ_k, edge Ψ_k, face Θ_k and interior or bubble modes Λ_k. The polynomial approximation of a field $V(t,\mathbf{x})$ at any point $\mathbf{x}_j$ is given by

$$V(t,\mathbf{x}_j)=\sum_{k=1}^{Nv}\hat{V}_k^V(t)\Phi_k(\mathbf{x}_j)+\sum_{k=1}^{Ne}\hat{V}_k^E(t)\Psi_k(\mathbf{x}_j)+\sum_{k=1}^{Nf}\hat{V}_k^F(t)\Theta_k(\mathbf{x}_j)+\sum_{k=1}^{Ni}\hat{V}_k^I(t)\Lambda_k(\mathbf{x}_j)+\Re, \quad (2.1)$$

where Nv is the number of vertex modes and $Ne=(P-1)$(number of edges), $Nf=(P-2)(P-1)/2$(number of faces) and $Ni=(P-1)(P-2)(P-3)/6$ are the number of the edge, face and interior modes respectively and P is polynomial order of the expansion (here a tetrahedral element is considered); the $\Re$ term represents the truncation error. In Fig. 3 we provide an illustration of the domain decomposition and the polynomial bases employed in NEKTAR. The boundary degrees of freedom, corresponding to adjacent elements, are coupled due to the C^0-continuity. The interior modes have zero support on the elemental boundaries, thus the boundary and interior degrees of freedom are solved in a decoupled manner by a technique known as substructuring (also referred to as static condensation) [3].

For the temporal discretization, a second-order accurate semi-implicit time integration scheme is employed based on a high-order splitting scheme [10], that decouples the velocity and pressure fields, i.e.,

$$\mathbf{v}^*=\sum_{k=0}^{Je-1}\alpha_k\mathbf{v}^{n-k}-\Delta t\left(\sum_{k=0}^{Je-1}\beta_k(\mathbf{nl})^{n-k}+\mathbf{f}\right),\quad \mathbf{nl}=\mathbf{v}\cdot(\nabla\mathbf{v}), \quad (2.2a)$$

$$\nabla^2 p=\frac{1}{\Delta t}\nabla\cdot\mathbf{v}^*, \quad (2.2b)$$

$$\gamma_0\mathbf{v}^{n+1}=\mathbf{v}^*+\Delta t(-\nabla p+\nu\nabla^2\mathbf{v}^{n+1}). \quad (2.2c)$$

By applying Galerkin projection, we obtain the weak formulations of Eqs. (2.2b) and (2.2c), namely

$$\mathbf{L}\hat{p}=-\frac{1}{\Delta t}(\nabla\cdot\mathbf{v}^*,\phi)+\left(\frac{\partial p}{\partial n},\phi\right) \quad (2.3a)$$

and

$$\mathbf{H}\hat{\mathbf{v}}^{n+1}=\frac{1}{\gamma_0}(\mathbf{v}^*-\Delta t\nabla p,\phi)+\frac{\Delta t\nu}{\gamma_0}\left(\frac{\partial\mathbf{v}}{\partial n},\phi\right)^{n+1}, \quad (2.3b)$$

where $\mathbf{H}=\mathbf{M}-\frac{\Delta t\nu}{\gamma_0}\mathbf{L}$, and $\mathbf{M}$ and $\mathbf{L}$ are the mass and stiffness matrices, respectively. The quantities with hat denote modal amplitudes, as the linear solves are performed in the

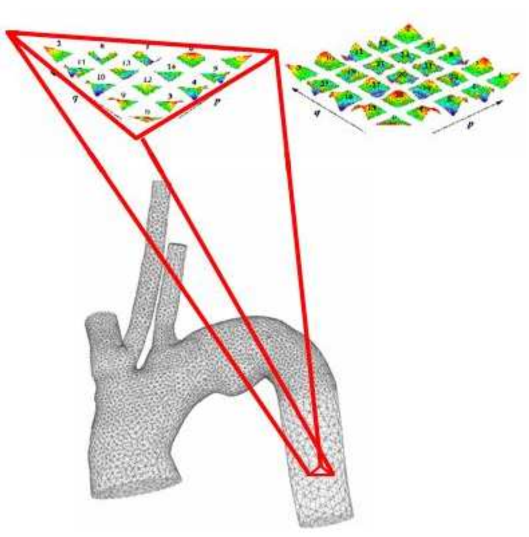

Figure 3: Illustration of the unstructured mesh and the polynomial spectral bases employed in NEKTAR. The solution domain is decomposed into nonoverlaping elements. Within each element the solution is approximated by vertex, edge, face and interior modes. The shape functions associated with the vertex, edge and face modes for fourth-order polynomial expansion on surface triangular and quadrilateral elements are shown in color.

modal domain. The Neumann pressure boundary condition at the boundaries with prescribed velocity are computed from

$$\frac{\partial p}{\partial n} = \sum_{k=0}^{Je-1} \left[\beta_k \left(-\frac{\partial \mathbf{v}}{\partial t} - \mathbf{nl} - \nu \nabla \times (\nabla \times \mathbf{v}) \right) \cdot \mathbf{n} \right]^{n-k}. \tag{2.4}$$

According to this scheme, the provisional field $\mathbf{v}^*$ is computed explicitly (see formula (2.2a)), then the Poisson equation is solved to compute the pressure (2.3a), and at the final step three Helmholtz solvers are employed for implicit solution for the three velocity vector components (2.3b). The solution of three Helmholtz and one Poisson equations is the bottleneck in scaling efficient numerical parallel solvers for use with thousands of processors.

2.2 Inter-patch conditions

The inter-patch conditions (IPC), required for coupling 3D patches, are computed explicitly and include the velocity boundary condition at the inlets along with pressure and velocity fluxes at the outlets. An illustration of two patches coupled by IPC is presented in Fig. 4. The outlet of patch A marked as Γ^- conforms with the inlet of patch B marked as Γ^+.

To impose IPC we follow a procedure similar to the discontinuous Galerkin (DG) method [11]. The hyperbolic component of the Navier-Stokes equation dictates the choice of interface condition for the velocity based on the upwinding principle. Assuming that

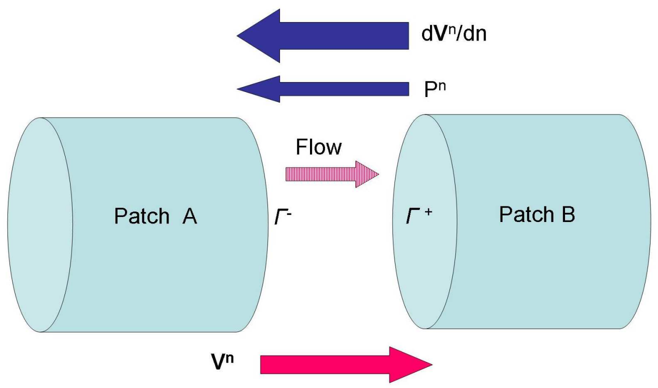

Figure 4: Two Level Domain Decomposition: two patches A and B are connected by the interface boundary conditions. Velocity computed at the outlet of A (Γ^-) is imposed as Dirichlet B.C. at the inlet of B (Γ^+); pressure and velocity flux computed at Γ^+ are imposed as B.C. at Γ^-.

$\mathbf{v}\cdot\mathbf{n}\geq 0$ (with $\mathbf{n}$ pointing outward) at the patch outlet we impose the inlet velocity condition in patch B as

$$\mathbf{v}^{n+1}|_{\Gamma^+}=\mathbf{v}^n|_{\Gamma^-}, \tag{2.5}$$

where the superscripts denote time steps. The velocity flux at the patch A outlet is computed as weighted average of the normal velocity derivatives from both sides of the interface, i.e.,

$$\left.\frac{d\mathbf{v}^{n+1}}{dn}\right|_{\Gamma^-}=c_1\left.\frac{d\mathbf{v}^n}{dn}\right|_{\Gamma^-}-(1-c_1)\left.\frac{d\mathbf{v}^n}{dn}\right|_{\Gamma^+}, \tag{2.6}$$

where the coefficient c_1 is in the range $0\leq c_1<1$. The choice of c_1 is important and will be investigated systematically in future work. Alternative choices of numerical fluxes may be considered; for example, imposing the total flux for the velocity at Γ^+ was advocated in [12, 13]. Also, different choices for the flux in the *DG* formulation can be found in [14, 15].

The pressure at the patch outlet is given by

$$p^{n+1}|_{\Gamma^-}=F(t-t_0)\bar{p}+(1-F(t-t_0))p_{IC}, \tag{2.7}$$

where $\bar{p}=0.5(p^n|_{\Gamma^-}+p^n|_{\Gamma^+})$, $F(t-t_0)=(1-e^{-\alpha(t^{n+1}-t^0)})^\beta$ with $\alpha>0$ and $\beta>0$ and p_{IC} is the initial conditions for the pressure; in our simulations we used $\alpha=20$ and $\beta=2$. The role of p_{IC} is explained bellow. The filter function $F(t-t_0)$ suppresses erroneous large pressure oscillations. The pressure oscillations at the inlet are due to the numerical scheme and incompatible initial conditions; e.g., in the beginning of the simulation the initial velocity field may not satisfy the continuity equation $\nabla\cdot\mathbf{v}=0$. In simulations on a single domain (1DD), the pressure oscillations at the beginning of a simulation do not affect the stability of the solver. However, in the 2DD formulation, the pressure computed at the inlet Γ^+ is imposed as Dirichlet B.C. at the outlet Γ^-, thus oscillations in p_{Γ^+} propagate to the

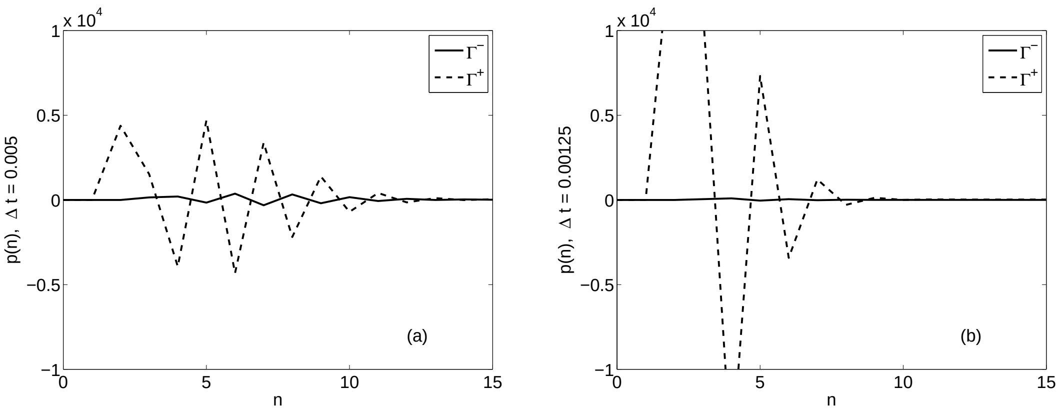

Figure 5: Steady flow simulation with 2DD: High amplitude oscillations of the pressure at the patch interface in the beginning of a simulation performed in a convergent pipe domain subdivided as illustrated in Fig. 8 (left). Solution is computed with third-order approximation in space and different size of time steps: (a) $\Delta t = 0.005$, and (b) $\Delta t = 0.00125$. High amplitude oscillations in p_{Γ^+} are reduced by the filter function $F(t-t_0)$ resulting in low amplitude oscillations in p_{Γ^-}.

adjacent patch. In the case of multiple interfaces required in complex arterial networks, out-of-phase pressure oscillations at outlets may lead to catastrophic results. Large oscillations in pressure may also appear when the simulation is restarted; in this case, the last term of Eq. (2.7) works to reduce the oscillations and, during the first few time steps, it keeps the pressure values at the patch interface close to p_{IC}. In Fig. 5 we show the oscillations of the mean pressure $\tilde{p} = (A_\Gamma)^{-1} \int_\Gamma p dA$, computed at the patch interface for two choices of Δt. The dramatic reduction in the amplitude of the oscillations at Γ^- is a result of applying the filter function $F(t-t_0)$ of Eq. (2.7).

The error introduced by the explicit treatment of the interface boundary conditions is controlled by the size of time step. In order to minimize the error, an iterative procedure for computing $\mathbf{v}^{n+1}$ and p^{n+1} may be applied. However, from the computational standpoint, such procedure is very inefficient in a sense that the computational time will grow linearly with the number of iterations required for convergence of the velocity and the pressure at patch interfaces. In large scale simulations of blood flow in the human arterial tree, computing the solution over one cardiac cycle typically takes 1-3 days at the present time, thus such iterations are computationally prohibitive.

An alternative way for enhancing the numerical accuracy is to approximate the boundary condition at the time step $t^{(n+1)}$ by applying a penalty term, i.e.,

$$\mathbf{v}^{n+1}|_{\Gamma^+} = \mathbf{v}^n|_{\Gamma^-} + \alpha_{BC} F(t-t_0)(\mathbf{v}^n|_{\Gamma^-} - \mathbf{v}^n|_{\Gamma^+}) \tag{2.8}$$

or by extrapolation, i.e.,

$$\mathbf{v}^{n+1}|_{\Gamma^+} = \mathbf{v}^n|_{\Gamma^-} + \alpha_{BC} F(t-t_0)(\mathbf{v}^n - \mathbf{v}^{n-1})|_{\Gamma^-}, \tag{2.9}$$

where $0 \le \alpha_{BC} \le 1$ is a relaxation parameter and $0 \le F(t-t_0) \le 1$. The choice of α_{BC} affects both accuracy and stability, however, the latter dictates its value.

In the SEM approach, the solution for velocity and pressure is performed in *modal* space, hence, the values of velocity and pressure can be transferred and imposed as B.C. in modal space bypassing expensive transformation from modal to physical space and vise versa. Of course, this can be done only when the computational meshes at Γ^- and Γ^+ are conforming. Imposing B.C. in modal space has an additional advantage: we can exploit the hierarchical structure of the base functions and reduce communication by imposing IPC by transferring only the most energetic modes. Although a small reduction in accuracy may occur, from the computational standpoint, limiting the number of modes to be collected from one subset of processors to another leads to shorter messages and consequently to reduction in computation time associated with imposing IPC. The latter is very important in ultra-parallel simulations.

3 Results

In this section we compare the parallel efficiency of two solvers based on the 1DD and 2DD approaches. Subsequently, we investigate the accuracy of numerical simulations of steady and unsteady flow performed with the 1DD and 2DD methods.

3.1 Performance

From the computational standpoint, one of the advantages of the 2DD approach over the 1DD approach is the minimization of blocking communication between processes, which enhances parallel efficiency. In the first simulation we use a computational domain consisting of 67456 tetrahedral elements sub-divided into two equal size patches. Separate runs are performed using the 1DD and the 2DD approaches, and in Fig. 6 we plot the mean-cpu time per time step. The computational saving with the 2DD approach is clear even for this relatively small size problem.

The potentially great advantage of the 2DD approach is in simulating very large scale problems. There are two difficulties in the solution of a such problems: (a) limited available memory per processor, and (b) tight communication between thousands of processors. Limited memory requires the use of many processors, which decreases the volume of computation versus volume of communication ratio. Implementation of coarse partitioning of a computational domains into M patches leads to partitioning of the communicator into M non-overlapping groups of processes [16]. The tight communication between processes is performed as intra-group message passing only. The communication between groups is required to exchange data across patch interfaces and it is limited to a small number of processes. This communication is non-blocking and is performed once in a time step. The overall computational efficiency is strongly affected by the parallel efficiency in the solution of a problem within a patch. In Fig. 7 we show the parallel efficiency of NEKTAR for the domain shown in Fig. 1. On a coarse level the domain is

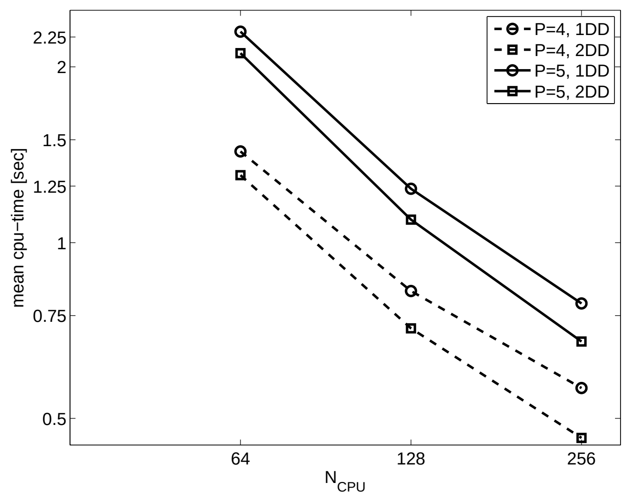

Figure 6: Simulation with 1DD and 2DD: performance. Y-axis is the mean-cpu time required per time step. Problem size: 67456 tetrahedral elements, polynomial order $P=4$ (dash line) and $P=5$ (solid line). Computation performed on the CRAY XT3 supercomputer.

partitioned into four patches. The parallel efficiency E_p is computed as:

$$E_p(N) = \frac{T_n/T_N}{N/n},$$

where N is the number of processes employed in simulation, n is the reference (minimum) number of processes used for parallel solution of a given problem and T_N (T_n) is the mean cpu-time per time step required for simulation on N (n) processes. The results of Fig. 7 verify that the overall parallel efficiency depends on the parallel efficiency of each subproblem on the different patches. Here patches A and C suffer from relatively low efficiency. In terms of optimizing the overall performance, this implies that we should optimize the parallel performance on the individual patches — a much simpler task than dealing with tens or hundreds of thousands of processors involved in the solution of the overall problem.

Table 1: Solution of large scale problem, computational complexity: flow simulation in the domain of Fig. 1. On a coarse level the computational domain is subdivided into four patches A-D. Nel — number of spectral elements. DOF — number of degrees of freedom, computed as a total number of quadrature points: $DOF = Nel*(P+3)(P+2)^2$ required for exact integration of linear terms.

patch	*Nel*	P	# DOF
A	120,813	6	69,588,288
B	20,797	6	11,979,072
C	106,219	6	61,182,144
D	77,966	6	44,908,416
total	**325,795**	**6**	**187,657,920**

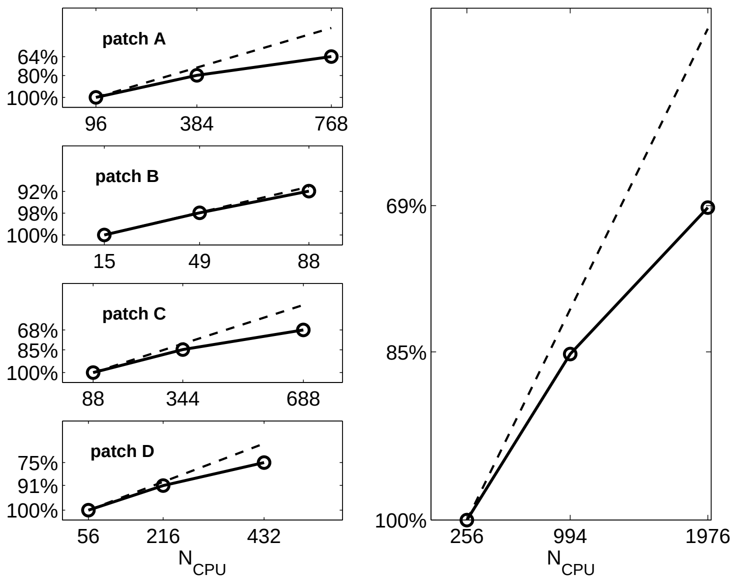

Figure 7: Simulation with 2DD: parallel efficiency. Steady flow simulation in the human aorta. The domain is sub-divided into four patches as shown in Fig. 1. Details on computational complexity are summarized in Table 1. Y-axis - parallel efficiency of a solver, E_p. Left: parallel efficiency in solution of tightly coupled system within a patch. Right: overall parallel efficiency. Computation performed on the CRAY XT3 supercomputer.

3.2 Accuracy verification

Our objective is to evaluate the potential loss of spectral accuracy in the numerical solution obtained using the 2DD approach. The accuracy degrades due to two factors: a) explicit treatment of the IPC; and b) incomplete transfer of the velocity and pressure modes in imposing the inter-patch conditions.

We use the following notations:
P — order of polynomial expansion;
P_{VBC} — maximum order of polynomial expansion in imposing the *velocity* IPC;
P_{PBC} — maximum order of polynomial expansion in imposing the *pressure* IPC;
P_{FBC} — maximum order of polynomial expansion in imposing the *velocity flux* IPC.

3.2.1 Benchmark problem

Numerical simulations were performed in a simple computational domain whose shape is similar to a converging blood vessel. Although the domain is axisymmetric, we use full 3D solver and non axisymmetric mesh to perform the simulations.

The domain consists of two blocks A and B as illustrated in Fig. 8. For reference, we also combined the two blocks into one, i.e., block AB. In the first configuration, the patch interface is normal to the z-axis and in the second configuration it intersects the z-axis at 80 degrees. First, we performed a steady flow simulation with Reynolds number $Re{=}D_AU_m/\nu{=}350$ where D_A is a diameter of the inlet of patch A, U_m is the mean velocity

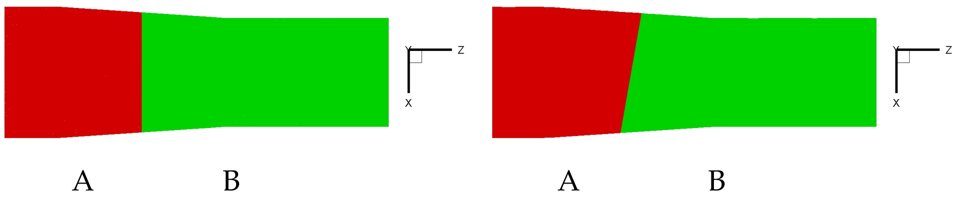

Figure 8: Illustration of two configurations of two-level domain decomposition into patches A and B. The interfaces are centered at $z=5$. Left: interface is normal to the pipe axis. Patch A has 4203 tetrahedral spectral elements and patch B has 8906 elements. Right: interface is at an angle of 80^o to the pipe-axis. The sizes of patches A and B are 7246 and 11331 tetrahedral spectral elements, respectively. Primary flow direction $(z-)$ is from left to right.

at the inlet and ν is a kinematic viscosity. Poiseuille flow is imposed at the entrance of the block A. At the outlet of the patch B a fully developed flow is assumed defined by zero velocity flux and constant pressure. Second, we simulate unsteady flow by imposing the Womersley velocity profile at the inlet of patch A. In this simulation the main characteristics of the flow are $Re=350$ and Womersley number $Ws=(D_A/2)^2\sqrt{\omega/\nu}=4.375$, which are typical values for the arterial flow in vessels with diameter of 4 to 5mm.

3.2.2 Convergence of flowrate and mean pressure at patch interface

We define the error in mass conservation across the interface as:

$$\epsilon_Q=\left|\frac{Q|_{\Gamma^+}-Q|_{\Gamma^-}}{Q|_{\Gamma^+}}\right|, \tag{3.1}$$

where $Q=|\int_\Gamma \mathbf{v}\cdot\mathbf{n}dA|$ and $\mathbf{n}$ is an unit vector normal to Γ. The error in the mean pressure $\tilde{p}$ is computed from

$$\epsilon_p=\left|\frac{\tilde{p}|_{\Gamma^+}-\tilde{p}|_{\Gamma^-}}{\tilde{p}|_{\Gamma^+}}\right|. \tag{3.2}$$

First, we consider steady flow simulation with relatively low spatial resolution *within* a patch, $P=3$. In Fig. 9 we plot $\tilde{p}(t)$ at the interface. In the first simulation the pressure IPC were imposed with $P_{PBC}=1$ (i.e., using vertex modes only) and relatively large size of time step was used i.e., $\Delta t=0.005$. Low spatial and temporal resolution result in high frequency oscillations in $\tilde{p}(t)$, whose amplitude is of order Δt at Γ^+ and considerably lower at Γ^-, as shown in Fig. 9(a). The next two simulations are performed with smaller size of the time step $\Delta t=0.00125$ (Fig. 9(b)) or with higher spatial accuracy in imposing the pressure IPC, i.e., $P_{PBC}=2$ (Fig. 9(c)); we observe that the oscillations can be removed by either reducing the size of a time step or by increasing P_{PBC}.

To investigate further the effect of spatial under-resolution we consider a steady flow simulation in the domain illustrated in Fig. 8(left) performed with higher order of accuracy i.e., $P=5$ and $\Delta t=0.00125$. We focus on the effects of spatial under-resolution in imposing IPC. In Table 2 we summarize details of simulations where the velocity IPC were imposed with $P_{VBC}=1,2,3$ and pressure IPC with $P_{PBC}=1,2$. Exponential

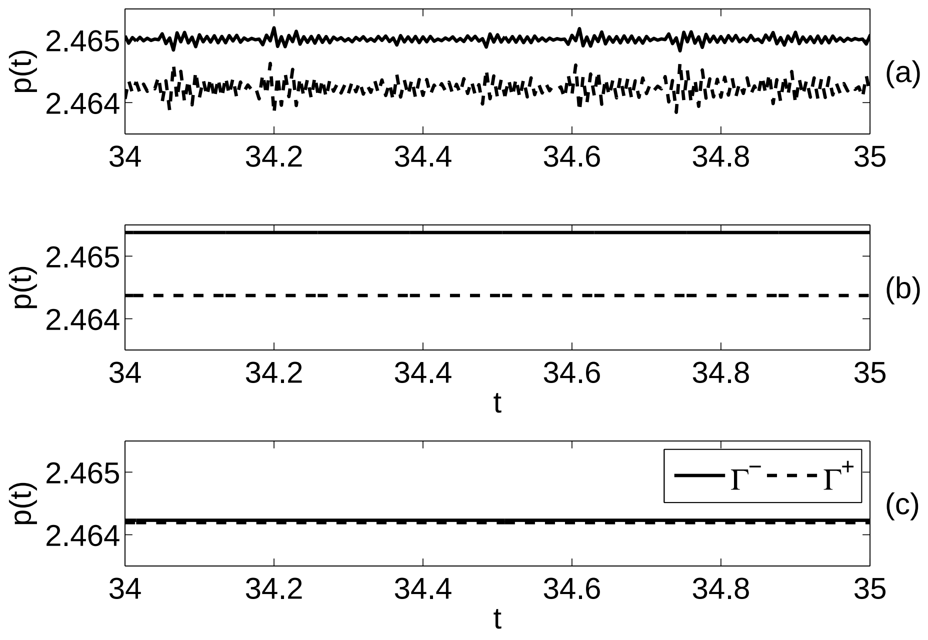

Figure 9: Steady flow simulation with 2DD: Simulation performed in domain of convergent pipe, sub-divided as illustrated in Fig. 8 (left). Convergence of the mean pressure at the patch interface. Solution computed with $P=3$. Velocity IPC is imposed with $P_{VBC}=3$.
(a) - $\Delta t=0.005$ pressure IPC is imposed with $P_{PBC}=1$.
(b) - $\Delta t=0.00125$ pressure IPC is imposed with $P_{PBC}=1$.
(c) - $\Delta t=0.005$ pressure IPC is imposed with $P_{PBC}=2$.

convergence in ϵ_Q and $\epsilon_{\tilde{p}}$ is observed. Simulations (a), (b) and (c) also show that imposing velocity IPC with higher order of accuracy has significant effect on *both* ϵ_p and ϵ_Q. We should note that the time splitting scheme (2.2a)-(2.2c) introduces an error in mass conservation. For example, the error in mass conservation in simulation (d) was $\epsilon_Q=|(Q_{inlet}-Q_{outlet})/Q_{inlet}|=6.1e\text{–}7$ in patch A and $\epsilon_Q=1.4e\text{–}6$ in patch B, which is comparable to the error introduced by incomplete transfer of modes in imposing the velocity IPC.

Decoupling the solution for the velocity and the pressure allows to approximate the two fields with the same polynomial order. The solution for the Poisson equation for the pressure, supplemented with Dirichlet boundary condition at the outlet and Neumann boundary condition at the Dirichlet-velocity boundaries, is unique and the spurious pressure modes are eliminated [3]. However, imposing outlet pressure boundary condition with $P_{PBC}=P$ or $P_{PBC}=P-1$ may lead to an instability. In our tests, we observed that using $P_{PBC}=P$ or $P_{PBC}=P-1$ (in the domain of Fig. 8) led to unstable simulations. In steady flow simulation, only $P_{PBC}=P$ led to instability. This issue deserves further investigation in the future. In simulations (c) and (d) (see Table 2) we observed that using $P_{PBC}=P-1$ gave practically the same result as $P_{PBC}=P-2$ as far as the mass conservation is concerned.

Table 2: Steady flow simulation with 2DD: exponential convergence in the error of a flow rate Q and mean pressure $\tilde{p}$ computed across the patch interface. Simulation performed in a domain of convergent pipe, sub-divided as illustrated in Fig. 8(left).

simulation	N	Δt	N_{VBC}	N_{PBC}	N_{FBC}	ϵ_Q	ϵ_P
a	5	1.25E-3	**1**	1	5	3.3E-2	1.0E-2
b	5	1.25E-3	**2**	1	5	8.2E-4	6.8E-4
c	5	1.25E-3	**3**	1	5	1.3E-7	3.1E-4
d	5	1.25E-3	**3**	**2**	5	1.6E-7	1.7E-5

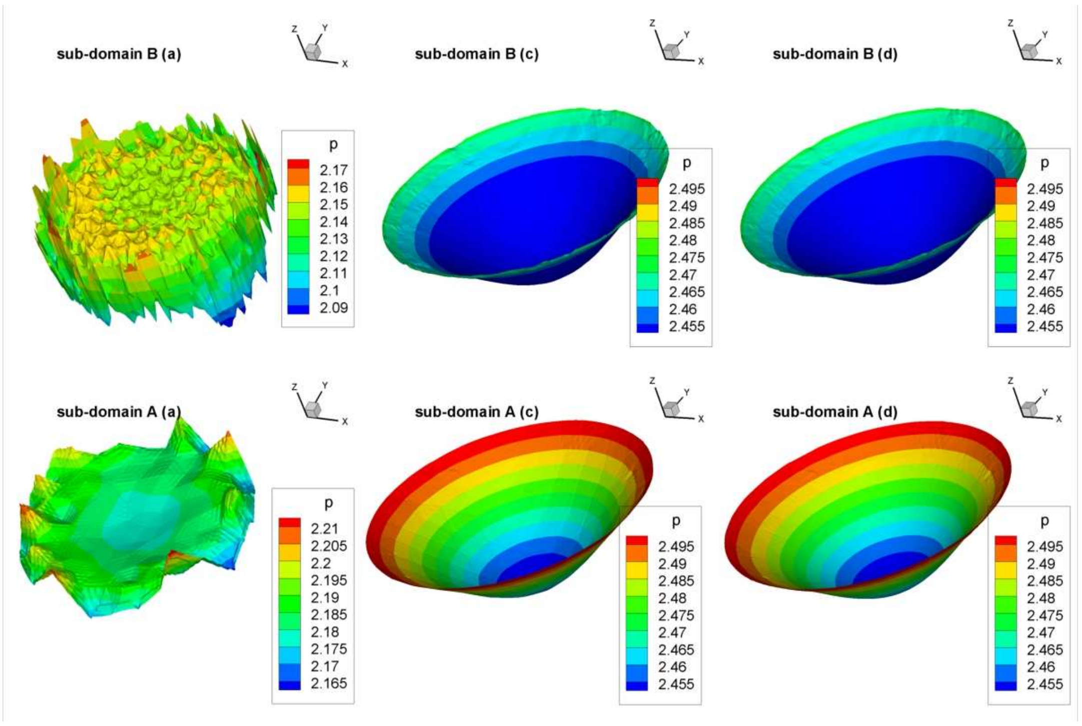

Figure 10: Steady flow simulation with 2DD: pressure distribution from both sides of patch interface. Simulation performed in a domain of convergent pipe, sub-divided as illustrated in Fig. 8(left) with $P=5$ and $\Delta t=0.00125$. The set-up is consistent with Table 2. Top plots: $p|_{\Gamma^+}$; bottom plots: plots $p|_{\Gamma^-}$. Left: case (a), $P_{VBC}=1$, $P_{PBC}=1$. Center: case (c), $P_{VBC}=3$, $P_{PBC}=1$. Right: case (d), $P_{VBC}=3$, $P_{PBC}=2$.

In Fig. 10 we plot the pressure distribution from both sides of the interface. The upper plots show the computed pressure at the inlet of the patch B, $p|_{\Gamma^+}$, while the lower plots show the pressure imposed as IPC, $p|_{\Gamma^-}$ at the outlet of patch A. To understand the origin of high frequency oscillations in pressure we refer to formula (2.4). Under-resolution in imposing velocity boundary condition results in significant discontinuity in the second derivatives of velocity computed at the inlet, and this induces pressure oscillations due to pressure-velocity coupling through the pressure B.C. In contrast, incomplete transfer of pressure modes from Γ^+ to Γ^- is equivalent to applying a cut-off filter and results

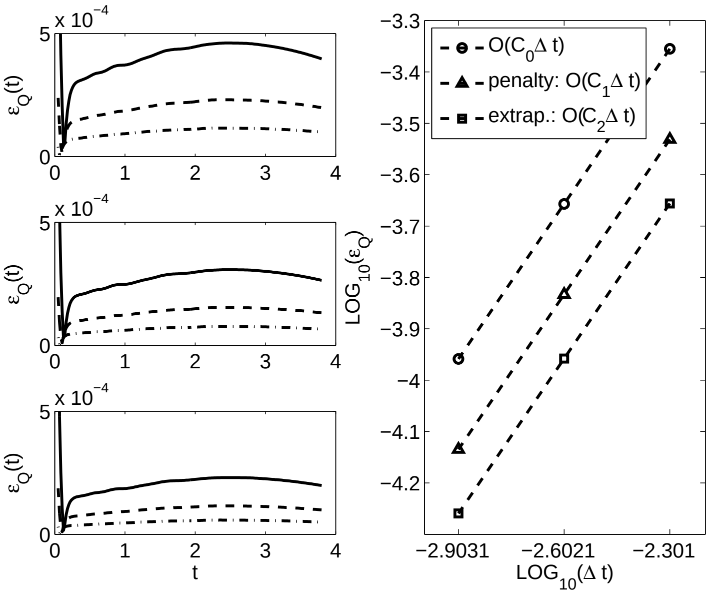

Figure 11: Unsteady flow simulation with 2DD: Convergence of flow rates at the patch interface. Simulation performed in a domain of convergent pipe, sub-divided as illustrated in Fig. 8 (left). Left upper plot: $\alpha=0.0$. Left center plot: penalty formulation, $\alpha=0.5$. Left lower plot: extrapolation, $\alpha=0.5$. Solid line $\Delta t=0.005$, dash line $\Delta t=0.0025$, dash-dot line $\Delta t=0.00125$. Right: convergence rate of numerical error ϵ_Q at time $t=1.9$.

in considerable smoothing of the pressure field particularly in a case of under-resolved velocity field.

The effect of explicit treatment of velocity IPC on the mass conservation across the interface was investigated also using unsteady flow simulation, where third-order accuracy in space ($P=3$) and second-order accuracy in time discretization within each patch was employed. Velocity and flux IPC were imposed with full resolution ($P_{VBC}=3$, $P_{FBC}=3$) and pressure IPC with $P_{PBC}=1$. Velocity and flux boundary conditions were imposed using three different methods: a) according to formula (2.5); b) using the penalty formulation (2.8) with $\alpha=0.5$; and c) using the extrapolation formula (2.9) with $\alpha=0.5$. The results are summarized in Fig. 11. All three methods are based on first-order (in time) explicit scheme in imposing IPC, however the coefficients (C_i) of the leading term in the truncation error are different as we show in Fig. 11(right).

3.2.3 Errors due to inter-patch interface

Here we compare the error in the pressure and vorticity fields obtain with the 1DD and 2DD approaches. In Fig. 12 we show the computational domain and the location where the pressure field was extracted for comparison. In Fig. 13 we compare the pressure computed with the two methods. We observe that the error in the numerical solution is greatest in the vicinity of patch interface and rapidly decays upstream where it is of order Δt, and downstream, where it converges to the imposed value of the pressure at the outlet, i.e., $p=0$. The localization of numerical error at the vicinity of the patch interface

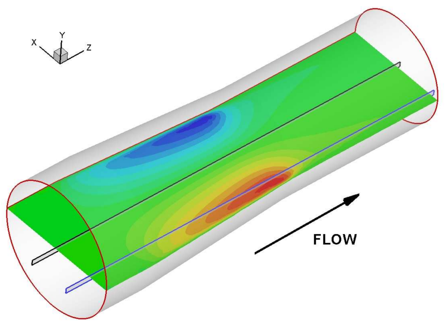

Figure 12: Illustration of computational domain and location of slice $y=0$ and lines $x=0$, $y=0$ (black) and $x=-1.6$, $y=0$ (blue); colors represent the non-dimensional u-velocity.

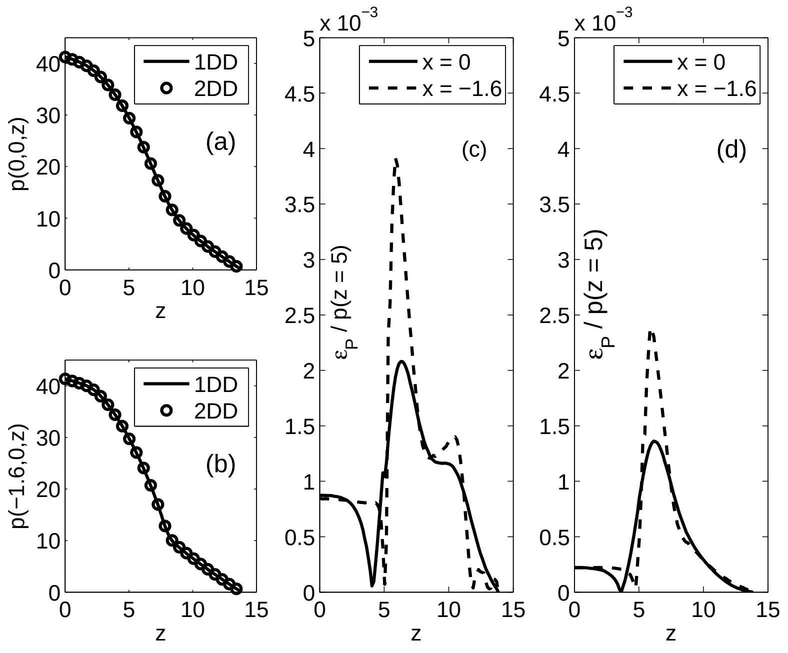

Figure 13: Unsteady flow simulation with 2DD in the computational domain of Fig. 8(right). Pressure along lines $y=0$, $x=0$ and $y=0$, $x=-1.6$ as marked in Fig. 12. (a) and (b) non-dimensional pressure values computed with 1DD and 2DD. (c) and (d) normalized difference between the pressure computed with 2DD and 1DD; (c) - $P_{VBC}=3$, $P_{PBC}=1$, (d) $P_{VBC}=3$, $P_{PBC}=2$. $P=5$, $\Delta t=0.0005$.

is due to reduced space for imposing IPC and also due to explicit treatment of the IPC. Next we compare the vorticity field. In Fig. 14 we plot the y-component of the vorticity, ω_y computed with 1DD and 2DD. We define the maximum deviation in the vorticity field by:

$$\epsilon_\omega = \frac{MAX(|\omega(1DD)-\omega(2DD)|)}{\omega_s(1DD)},$$

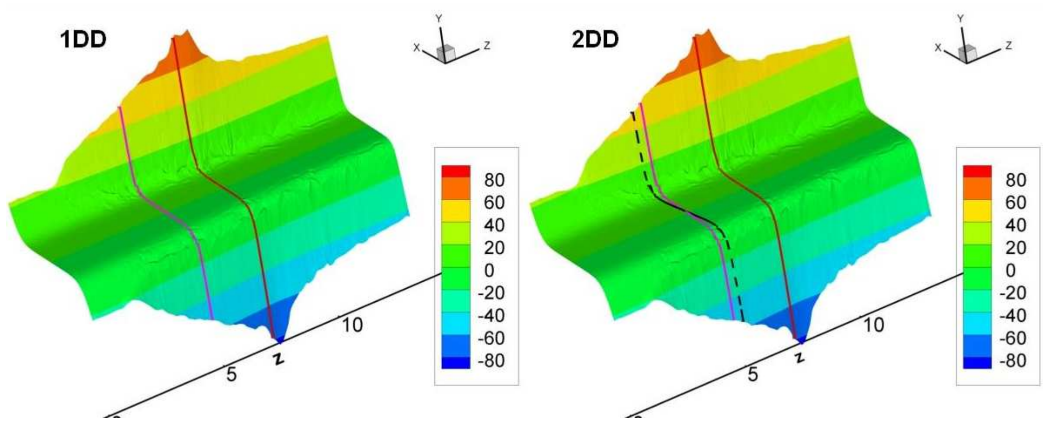

Figure 14: Unsteady flow simulation with 2DD: comparison of vorticity filed computed with 1DD and 2DD: Y-component of vorticity field (ω_y) contours at slices $y=0$. Y-axis is ω_y. Solid lines represent location ($z=5$ and $z=7.5$) where ω_y was extracted. Dash line depicts the location of patch interface. $P=5$, $\Delta t=0.0005$.

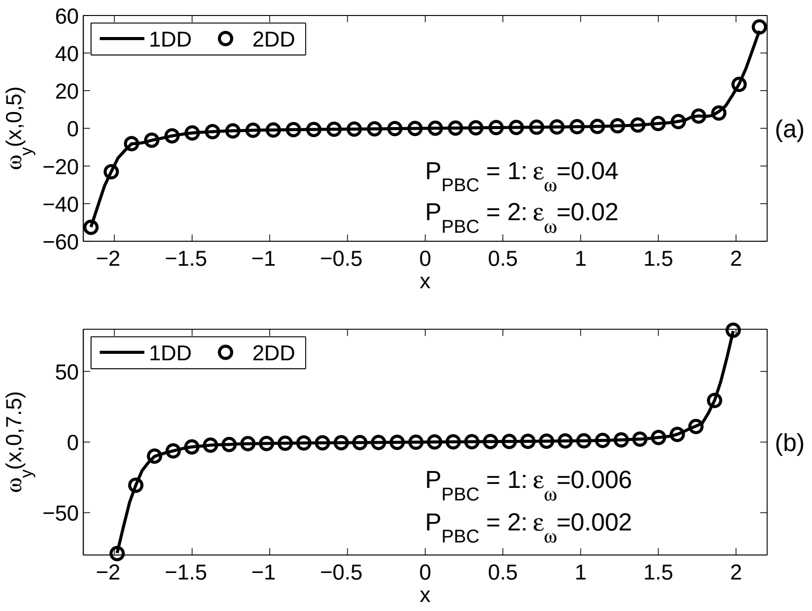

Figure 15: Unsteady flow simulation with 2DD: comparison of vorticity filed computed with 1DD and 2DD. Computational domain is illustrated in Fig. 8(right). Y-component of vorticity field, ω_y, extracted at (a) - $y=0,z=5$ and (b) - $y=0,z=7.5$. ϵ_ω - deviation in ω_y. $P_{VBC}=3$, $P_{PBC}=1,2$, $P=5$, $\Delta t=0.0005$.

here the value of a scaling factor $\omega_s(1DD)$ is computed at a point where the difference $|\omega(1DD)-\omega(2DD)|$ is maximum; results are presented in Fig. 15. The maximum value of ϵ_{ω_y} in simulation with $P_{PBC}=1$ ($P_{PBC}=2$) is about 4% (2%) and decreases by an order of magnitude a short distance from the interface.

4 Summary

Currently none of the existing domain decomposition methods for the Navier-Stokes equations scale well beyond a few thousand processors, and hence their anticipated performance on petaflop-size systems will be very poor. We have presented a two-level decomposition method that can potentially scale well to hundreds of thousands of processors. The main advantage of the suggested method is in splitting a large domain, where the solution is approximated with C^0 polynomial expansion, into a set of patches of manageable size with negligible increase in the number of degrees of freedom. Continuity of the solution at patch interfaces is obtained by implementing a DG-like approach both for the advection and diffusion contributions. There is a potential loss of accuracy due to the patch interfaces but more elaborate conditions can be applied to enhance the accuracy as we have demonstrated here for a benchmark problem. One possibility is the use of overlapped domains, and we are currently working along this direction. A theoretical analysis of the accuracy and the stability of the new method is also required in the near future.

From the computational standpoint, solution of a set of small tightly coupled problems is more efficient than solution of a single much larger problem. In particular, the new two-level method we presented matches well with the new petaflop hybrid-type architectures, consisting of a few hundreds of "fat nodes", with each node containing several thousand processors. We can envision, therefore, a natural mapping of patches and nodes, with all intensive data transfers taking place efficiently within each node. This new parallel paradigm can be readily implemented using an MPI/OpenMP approach [17].

Acknowledgments

This work was supported by the National Science Foundation CI-TEAM grant. Computations were performed on CRAY XT3 of Pittsburgh Supercomputing Center with the help of David O'Neal and on the Ranger supercomputer of Texas Advanced Computing Center with the help of John Cozes.

References

[1] `www.physiome.org`.

[2] S. Dong, J. Insley, N. T. Karonis, M. Papka, J. Binns and G. E. Karniadakis, Simulating and visualizing the human arterial system on the TeraGrid. Future generation computer systems, Int. J. Grid Comput.: Theor., Meth. Appl., 22 (2006), 1011-1017.

[3] G. E. Karniadakis and S. J. Sherwin, Spectral/hp Element Methods for CFD, 2nd ed., Oxford University Press, Oxford, 2005.

[4] L. Grinberg, Multiscale Modelling and Parallel Algorithms for the Cerebrovasculature, Ph.D Thesis, Brown University, in progress.

[5] R. M. Kirby, T. C. Warburton, S. J. Sherwin, A. Beskok and G. E. Karniadakis, The Nektar code: Dynamic simulations without remeshing, Proc. 2nd International Conference on Computational Technologies for Fluid/Thermal/Chemical Systems with Industrial Applications, August 1-5, 1999.

[6] I. Bica, Iterative Substructiring Algorithm for p-version Finite Element Method for Elliptic Problems, Ph.D Thesis, Courant Institute, NYU, 1997.

[7] S. Sherwin and M. Casarin, Low-energy basis preconditioning for elliptic substructured solvers based on unstructured spectral/hp element discretization, J. Comput. Phys., 171(1) (2001), 394-417.

[8] L. Grinberg, D. Pekurovsky, S. J. Sherwin and G. E. Karniadakis, Parallel performance of a low energy basis preconditioner for spectral/hp elements, Parallel Computing, under review.

[9] `http://glaros.dtc.umn.edu/gkhome/views/metis`.

[10] G. E. Karniadakis, M. Israeli and S. A. Orszag, High-order splitting methods for the incompressible Navier-Stokes equations, J. Comput. Phys., 97(1) (1991), 414-443.

[11] B. Cockburn, G. E. Karniadakis and C. W. Shu, Discontinuous Galerkin Methods: Theory, Compuration and Applications, Springer, 2000.

[12] T. J. R. Hughes, G. Scovazzi, P. B. Bochev and A. Buffa, A multiscale discontinuous Galerkin method with the computational structure of a continuous Galerkin method, Comput. Methods Appl. Mech. Engrg., 195 (2006), 2761-2787.

[13] M. Gander, C. Japhet, Y. Maday and F. Nataf, A new cement to glue nonconforming grids with Robin interface conditions: The finite element case, in: Domain Decomposition Methods in Science and Engineering Series: Lecture Notes in Computational Science and Engineering, 40(4) (2006), 259-266.

[14] M. Zhang and C. W. Shu, An analysis of three different formulations of the discontinuous Galerkin method for diffusion equations, Math. Model. Meth. Appl. Sci., 13(3) (2003), 395-413.

[15] R. M. Kirby and G. E. Karniadakis, Selecting the numerical flux in discontinuous Galerkin methods for diffusion problems, J. Sci. Comput., 22-23(3) (2005), 385-411.

[16] L. Grinberg, B. Toonen, N. Karonis, G. E. Karniadakis, A New Domain Decomposition Technique for TeraGrid Simulations, TG07, 2007.

[17] S. Dong and G. E. Karniadakis, Dual-level parallelism for high-order CFD methods, Parallel Computing, 30(1) (2004), 1-20.