A Domain Decomposition Solver for the Steady Navier-Stokes Equations

E. M. Rønquist*

Abstract

We present two new domain decomposition solvers in the context of conforming spectral element discretizations. The first is a domain decomposition solver for the discrete steady convection-diffusion equation, while the second is a domain decomposition solver for the discrete steady Stokes or Navier-Stokes equations. The solution algorithms are both based on the additive Schwarz method in the context of nonoverlapping subdomains. The key ingredients are: (i) a coarse global system; (ii) a set of local, independent subproblems associated with the subdomains (or spectral elements); (iii) a system associated with the unknowns on the subdomain interfaces; and (iv) a Krylov method such as the CG algorithm or the GMRES algorithm. We present numerical results that demonstrate the convergence properties of the new solvers, as well as the applicability of the methods to solve heat transfer and incompressible fluid flow problems.

Key words: spectral element, domain decomposition, additive Schwarz, convection-diffusion, Navier-Stokes.

AMS subject classifications: 76D05, 65N30, 65N35, 65N55.

1 Introduction

In this paper we shall discuss the solution of the steady convection-diffusion equation as well as the solution of the steady, incompressible Navier-Stokes equations. In terms of spatial discretization, our primary focus will be the use of conforming spectral elements [39, 32], however, the general framework should also apply to the p- or h-p- type finite element method [5, 3, 4, 18, 37, 36]. The spectral element method is very similar to the p-type finite element method, but with a particular emphasis on tensor-product forms: tensor-product nodal bases, tensor-product Gauss quadratures, and tensor-product sum-factorization techniques for efficient matrix-vector product evaluations [38, 32].

The solution of the resulting set of algebraic equations poses a special challenge for high-order methods due to the long-range couplings and the severe conditioning associated with these methods. Direct methods are very computer intensive and therefore rarely practical, especially when considering general three-dimensional geometries and general elemental decompositions. An iterative approach seems to be the only viable alternative for such problems.

For the steady Stokes problem, a popular approach has been to use a form of the Uzawa procedure [26, 27, 31]. The attractive side of this approach is that it decouples a saddle problem into two symmetric, positive (semi)-definite forms, one for the pressure and one for the velocity. The solution can thus be obtained by solving a series of elliptic problems, with each elliptic problem solved with a standard conjugate gradient like method.

For the steady convection-diffusion problem, the presence of the nonsymmetric convection term has prevented an efficient iterative solution of the discrete, steady equations in the past. The most popular approach for spectral element discretizations has been to solve an unsteady problem, and integrate these equations until a steady state has been reached [32, 33]. Following such an approach, the nonsymmetric convection term is typically treated explicitly, while the symmetric diffusion term is treated implicitly, thus avoiding a linear system of equations with a nonsymmetric matrix. A similar approach has also been applied for solving the steady Navier-Stokes equations.

Iterative techniques for nonsymmetric problems, such as the GMRES algorithm [46], has earlier been used in the context of solving the fully coupled, discrete Navier-Stokes equations. However, the availability of good preconditioners is still very limited. In the context of low-order finite element discretizations, the most common precondition-

*Nektonics, Inc., 875 Main St., Cambridge, MA 02139

ICOSAHOM'95: Proceedings of the Third International Conference on Spectral and High Order Methods.

ers are either based upon a diagonal scaling [50], or some form of element-by-element preconditioning [51, 42]. In the context of high-order finite element/spectral element discretizations, even less progress has been made in terms of constructing efficient preconditioners.

The work we present in this paper is an attempt to address this deficiency. The algorithms we propose are inspired by recent progress in domain decomposition techniques, in particular, iterative substructuring techniques [8, 10, 11, 21]. Although an impressive development has taken place over the past few years [35, 47, 22, 20, 30], including nonsymmetric problems [53, 14, 15], only very limited results seem to have been reported in the area of solving Stokes and Navier-Stokes problems [9]. Although our algorithms cannot claim to have a polylogarithmic convergence rate (at least not yet), we believe that they nonetheless represent a significant advance compared to current iterative methods for solving steady, incompressible fluid flow problems.

Our approach will be as follows: As a point of departure we shall use an additive Schwarz method without overlap, that is, we shall use what is also referred to as an iterative substructuring method. Recently, polylogarithmic convergence rates have been reported for elliptic problems in the context of three-dimensional spectral element discretizations using this class of solution methods [40, 41]. The method we propose for the elliptic kernel in this study will, however, be less optimal than the solution method proposed in [40, 41]. The reason for this is that the method we propose for the interface system is very simple and easy to invert, and that it can readily be extended as a building block for the Navier-Stokes solver that we propose.

The outline of the paper is as follows: In Section 2 we present spectral element discretizations for the Poisson problem, the steady Stokes problem, the steady convection-diffusion problem, and the steady Navier-Stokes problem. In Section 3 we propose iterative substructuring methods for the resulting discrete systems, and in Section 4 we present two-dimensional and three-dimensional numerical results. The major conclusions from this study are presented in Section 5.

2 Spectral element discretizations

2.1 The Poisson equation

We consider here the solution of the Poisson problem in a domain Ω,

$$-\nabla^2 u = f \quad \text{in } \Omega, \tag{1}$$

$$u = 0 \quad \text{on } \partial\Omega, \tag{2}$$

where f is the given data and u is the solution. In deriving the set of discrete equations we shall assume that Ω is a two-dimensional domain. This assumption simplifies the definition and discussion of the spectral element method, and is used for reasons of exposition only. Fully three-dimensional cases will be considered later.

As a point of departure for our numerical discretization we consider the equivalent variational formulation of problem (1)-(2): Find $u \in \mathcal{V} \equiv H_0^1(\Omega)$ such that

$$a(u,v) = f(v) \qquad \forall v \in \mathcal{V}, \tag{3}$$

where the bilinear form $a : \mathcal{V} \times \mathcal{V} \rightarrow \mathbf{R}$ and the linear form $f : \mathcal{V} \rightarrow \mathbf{R}$ are defined as

$$a(u,v) = \int_\Omega \nabla u \cdot \nabla v \, d\Omega, \tag{4}$$

$$f(v) = \int_\Omega f \, v \, d\Omega. \tag{5}$$

Here, $\mathcal{V} = H_0^1(\Omega)$ is the standard Sobolev space.

Next, we assume that the domain Ω is broken up into K non-overlapping and geometrically conforming quadrilateral elements (or subdomains) Ω_k, $1 \leq k \leq K$. This implies that the intersection of two elements Ω_k and Ω_l is either empty or reduced to a common vertex or a common edge; in the latter case we define the open interval $\Gamma_{k,l}$ as

$$\overline{\Gamma_{k,l}} = \overline{\partial\Omega}_k \cap \overline{\partial\Omega}_l. \tag{6}$$

The discretization of problem (3) consists of choosing a finite-dimensional space V that approximates $\mathcal{V}$: Find $u_\delta \in V$ such that

$$a(u_\delta, v_\delta) = f(v_\delta), \qquad \forall v_\delta \in V. \tag{7}$$

Before we define the discrete space V, we first define the space $Q_N(\hat{\Omega})$ to be the set of all polynomials of degree less than or equal to N in each spatial direction on the reference domain $\hat{\Omega} =]-1,1[^2$ in $\mathbf{R}^2$. Let $F_k(\zeta_1,\zeta_2)$ be the affine transformation (or isoparametric mapping) from the reference domain $\hat{\Omega}$ onto Ω_k. The polynomial approximation space $Q_N(\Omega_k)$ is then defined as

$$Q_N(\Omega_k) = \{ v(F_k^{-1}(\zeta_1,\zeta_2)),\ v \in Q_N(\hat{\Omega}) \}. \tag{8}$$

We now proceed by defining the space $\tilde{V}$ of piecewise polynomials as

$$\tilde{V} = \{ v : \ \forall k,\ 1 \leq k \leq K,\ v_k = v_{|\Omega_k} \in Q_N(\Omega_k) \}. \tag{9}$$

The finite-dimensional space V is then defined as

$$V = \tilde{V} \cap H_0^1(\Omega). \tag{10}$$

In order to derive a set of algebraic equations, we need to define a quadrature rule in order to evaluate the integrals (4)-(5) in the variational form, and we also need to define a basis for the discrete space V. It is natural to use quadrature formulas of the Gauss-Lobatto Legendre type [32, 43], constructed from the zeros ξ_j, $0 \le j \le N$ in the interval $\Lambda =]-1,1[$ of the polynomial $(1-\zeta^2)L'_N(\zeta)$. Here, L_N denotes the Legendre polynomial of degree N over Λ. The quadrature rules for the multi-dimensional case are then constructed as the tensor-product extension of the one-dimensional Gauss-Lobatto Legendre (GLL) points. For the two-dimensional case, the set of points $\xi_{pq} = (\xi_p, \xi_q)$, $0 \le p,q \le N$ refer to the GLL points on the reference domain $\hat{\Omega} = \Lambda \times \Lambda =]-1,1[^2$. These points are then mapped via the affine transformation (or isoparametric mapping) $F_k(\zeta_1,\zeta_2)$ onto Ω_k, defining the points $\xi^k_{pq} = (\xi^k_p, \xi^k_q)$, $0 \le p,q \le N$, $1 \le k \le K$.

A typical integral over the subdomain Ω_k in the variational form is then evaluated in the following way:

$$\begin{aligned}\int_{\Omega_k} o(x,y)\,dxdy &= \int_{-1}^{1}\int_{-1}^{1} \phi(F_k^{-1}(\zeta_1,\zeta_2)) \mid J_k \mid d\zeta_1 d\zeta_2 \\ &\cong \sum_{\alpha=0}^{N}\sum_{\beta=0}^{N} \rho_\alpha \rho_\beta\, \phi(\xi^k_\alpha,\xi^k_\beta) \mid J(\xi^k_\alpha,\xi^k_\beta) \mid .\end{aligned}$$

Here, J_k is the Jacobian associated with the affine transformation F_k, and ρ_j, $0 \le j \le N$ are the GLL quadrature weights associated with the GLL points ξ_j, $0 \le j \le N$. We remark that the numerical quadrature formula is exact for polynomials $\phi \in Q_{2N-1}(\Omega_k)$ [49].

Having defined a numerical quadrature rule we can now pose the discrete problem as: Find $u_\delta \in V$ such that

$$a_\delta(u_\delta, v_\delta) = f_\delta(v_\delta)\,, \qquad \forall v_\delta \in V\,,$$

where a and f in (7) have been replaced by a_δ and f_δ to indicate integration of the bilinear and linear form by GLL quadrature.

The GLL points are also used to define a tensor-product, Lagrangian interpolant basis [32, 43]. These basis functions are defined over each Ω_k as polynomials $H^k_{pq} \in Q_N(\Omega_k)$ that satisfy

$$\forall p,q,p',q',\ 0 \le p,q,p',q' \le N,\ H^k_{pq}(\xi^k_{p'},\xi^k_{q'}) = \delta_{pp'}\delta_{qq'}\,.$$

In order to define a basis for the space $\tilde{V}$, these polynomials are extended by zero in all the other subdomains. An element $v \in \tilde{V}$ can then be expressed as

$$v = \sum_{k=1}^{K}\sum_{p=0}^{N}\sum_{q=0}^{N} V^k_{pq} H^k_{pq}\,, \tag{11}$$

where

$$\forall k,\ 1 \le k \le K,\ v_k = v_{|\Omega_k},\ V^k_{pq} = v_k(\xi^k_{pq})\,.$$

The basis (11) represents a tensor-product, Lagrangian interpolant basis where the degrees-of-freedom of elements in $\tilde{V}$ are the nodal values $V^k_{pq} = v_k(\xi^k_{pq})$, $0 \le p,q \le N$, $1 \le k \le K$. In order to represent an element in the discrete space V, we also need to honor the C^0 continuity requirement across the elemental boundaries $\Gamma = \{\Gamma_{k,l}\}$, as well as the homogeneous boundary conditions along $\partial\Omega$.

Choosing appropriate test functions, we are now in a position to derive a set of algebraic equations which can be expressed in matrix form as

$$\underline{A}\,\underline{u} = \underline{f}\,. \tag{12}$$

Here, $\underline{A}$ is a symmetric positive definite (SPD) matrix representing the discrete Laplace operator, $\underline{u}$ is a vector representing the nodal unknowns, and $\underline{f}$ represents the discrete right hand side.

Remark 2.1 *The extension to three-dimensional domains follows readily from the application of tensor-product forms [38, 32, 43].*

Remark 2.2 *For problems including non-homogeneous Dirichlet boundary conditions, a standard approach is to act on these boundary values with a discrete Laplacian corresponding to Neumann boundary conditions. The result is then subtracted from the right-hand side $\underline{f}$, and we arrive at a system similar to (12), to be solved for the internal nodal values $\underline{u}$.*

Remark 2.3 *In the case of Neumann boundary conditions, the variational form naturally results in surface integrals due to the integration by parts [48]. The given Neumann boundary conditions are then inserted into these surface integrals, and the result is absorbed into the right hand side.*

2.2 The steady Stokes equations

We now turn to the discretization of the steady Stokes equations

$$-\mu\nabla^2\mathbf{u} + \nabla p = \mathbf{f} \qquad \text{in } \Omega\,, \tag{13}$$

$$\nabla\cdot\mathbf{u} = 0 \qquad \text{in } \Omega\,, \tag{14}$$

$$\mathbf{u} = \mathbf{0} \qquad \text{on } \partial\Omega\,. \tag{15}$$

Here, $\mathbf{u}$ is the fluid velocity, p is the pressure, μ is the viscosity, and $\mathbf{f}$ is a body force. Again, for reasons of exposition, we assume that Ω is a two-dimensional domain.

A spectral element discretization of (13) - (15) is based on the equivalent weak form. For two-dimensional problems and homogeneous velocity boundary conditions we can formulate this problem as: Find $\mathbf{u} \in \mathcal{V}^2 = [H_0^1(\Omega)]^2$ and $p \in \mathcal{W} = L_0^2(\Omega)$ such that

$$\begin{aligned} a(\mathbf{u},\mathbf{v}) + b(\mathbf{v},p) &= f(\mathbf{v}) \quad \forall \mathbf{v} \in \mathcal{V}^2 , \\ b(\mathbf{u},q) &= 0 \quad \forall q \in \mathcal{W} . \end{aligned}$$

Here, the bilinear form $a : \mathcal{V}^2 \times \mathcal{V}^2 \rightarrow \mathbf{R}$, the bilinear form $b : \mathcal{V}^2 \times \mathcal{W} \rightarrow \mathbf{R}$, and the linear form $f : \mathcal{V}^2 \rightarrow \mathbf{R}$ are defined as:

$$a(\mathbf{u},\mathbf{v}) = \mu \int_\Omega \nabla \mathbf{u} \cdot \nabla \mathbf{v} \, d\Omega , \tag{16}$$

$$b(\mathbf{u},q) = -\int_\Omega (\nabla \cdot \mathbf{u}) \, q \, d\Omega , \tag{17}$$

$$f(\mathbf{v}) = \int_\Omega \mathbf{f} \, \mathbf{v} \, d\Omega . \tag{18}$$

Here, $\mathcal{W} = L_0^2(\Omega)$ is the space of all functions which are square integrable and have zero average over Ω.

The discretization of the steady Stokes problem now consists of choosing a discrete velocity space V^2 that approximates $\mathcal{V}^2$ and a discrete pressure space W that approximates $\mathcal{W}$. For the discrete velocity space V^2, we shall consider the space V as defined in (10) for each velocity component. As a pressure space W we need to choose a compatible space that honors the Brezzi-Babuška (inf-sup) condition [12, 2]. For spectral element discretizations, a good choice is to use the discrete pressure space [6, 32, 34]

$$W = \{ w : \ \forall k, \ 1 \le k \le K, \ \ w_k = w_{|\Omega_k} \in Q_{N-2}(\Omega_k), \ \int_\Omega w \, d\Omega = 0 \ \},$$

that is, the polynomial degree for the pressure is two orders lower than for the velocity inside each subdomain (or spectral element). We remark that since the pressure needs only be square integrable, no continuity requirement for the pressure is enforced between the elements.

As for the Poisson problem, we evaluate all the integrals in the variational form by a tensor-product GLL quadrature rule, and we can pose the discrete problem as: Find $\mathbf{u}_\delta \in V^2$ and $p_\delta \in W$ such that

$$a_\delta(\mathbf{u}_\delta,\mathbf{v}_\delta) + b_\delta(\mathbf{v}_\delta,p_\delta) = f_\delta(\mathbf{v}_\delta) \quad \forall \mathbf{v}_\delta \in V^2 , \tag{19}$$

$$b_\delta(\mathbf{u}_\delta,q_\delta) = 0 \quad \forall q_\delta \in W . \tag{20}$$

where a, b, and f in (16)-(18) have been replaced by a_δ, b_δ, and f_δ in order to indicate integration of the bilinear and linear forms by GLL quadrature.

The basis for an element in V (e.g., a single velocity component) is the same as the one defined for the discrete Poisson problem, i.e., a tensor-product, Lagrangian interpolant basis associated with the GLL points. The basis for an element in W (e.g., the pressure) is also taken to be a tensor-product, Lagrangian interpolant basis, however, this basis is associated with the *internal* GLL points [1]. Specifically, the basis functions are defined over each Ω_k as polynomials $\tilde{H}_{pq}^k \in Q_{N-2}(\Omega_k)$ that satisfy

$$\forall p,q,p',q', \ \ 1 \le p,q,p',q' \le N-1, \ \ \tilde{H}_{pq}^k(\xi_{p'}^k,\xi_{q'}^k) = \delta_{pp'}\delta_{qq'} .$$

In order to define a basis for the space W, these polynomials are extended by zero in all the other subdomains. An element $w \in W$ can then be expressed as

$$w = \sum_{k=1}^{K} \sum_{p=1}^{N-1} \sum_{q=1}^{N-1} W_{pq}^k \tilde{H}_{pq}^k ,$$

where

$$\forall k, \ 1 \le k \le K, \ w_k = w_{|\Omega_k}, \ W_{pq}^k = w_k(\xi_{pq}^k) .$$

Choosing appropriate test functions, we arrive at a set of algebraic equations which can be expressed in matrix form as

$$\underline{\mathbf{A}}\,\underline{\mathbf{u}} - \underline{\mathbf{D}}^T \underline{p} = \underline{\mathbf{f}} , \tag{21}$$

$$\underline{\mathbf{D}}\,\underline{\mathbf{u}} = \underline{0} . \tag{22}$$

Here, $\underline{\mathbf{A}}$ is the discrete viscous operator, $\underline{\mathbf{D}}$ is the discrete divergence operator, and its transpose $\underline{\mathbf{D}}^T$ is the discrete gradient operator. The vector $\underline{\mathbf{u}}$ contains the nodal velocity values, $\underline{p}$ represents the nodal pressure values, and $\underline{\mathbf{f}}$ are the nodal forces.

Remark 2.4 *The extension to three-dimensional domains follows readily from the application of tensor-product forms [38, 32, 43].*

Remark 2.5 *In the case of non-homogeneous Dirichlet velocity boundary conditions, we follow a similar procedure as for the Poisson problem.*

Remark 2.6 *The non-staggered discretization procedure outlined above is valid for polynomial approximations $N \ge 2$, that is, the coarsest discretization represents the use of a Q_2/Q_0 element.*

2.3 The steady convection-diffusion equation

We now consider the steady, scalar convection-diffusion problem

$$-\alpha_0 \nabla^2 \phi + \mathbf{u} \cdot \nabla \phi = f \quad \text{in } \Omega , \tag{23}$$

$$\phi = 0 \quad \text{on } \partial\Omega , \tag{24}$$

where f is the given data, α_0 is the diffusivity, $\mathbf{u}$ is a given convecting velocity field, and ϕ is the solution.

Using a similar procedure as for the Poisson problem, a Galerkin formulation of (23) can be expressed as: Find $\phi_\delta \in V$ such that

$$a(\phi_\delta, v_\delta) + c(\phi_\delta, v_\delta) = f(v_\delta)\,, \qquad \forall v_\delta \in V\,,$$

where the bilinear form $c : \mathcal{V} \times \mathcal{V} \rightarrow \mathbf{R}$ is defined as

$$c(\phi, \psi) = \int_\Omega \psi\, \mathbf{u} \cdot \nabla \phi \, d\Omega\,. \tag{25}$$

A spectral element discretization of the steady convection-diffusion problem (23) results in a set of discrete equations which is linear and nonsymmetric, and which can be expressed in matrix form as

$$[\underline{A} + \underline{C}]\,\underline{\phi} = \underline{f}\,. \tag{26}$$

Here, the matrix $\underline{A}$ represents the discrete Laplace operator (linear and symmetric), while $\underline{C}$ represents the discrete convection operator (linear and nonsymmetric); the vector $\underline{\phi}$ represents the nodal values of the discrete solution ϕ_δ.

Remark 2.7 *Equation (25) represents the convective form of the convection operator. There are alternative forms that can be used [44], however, we shall not consider these here.*

Remark 2.8 *No upwinding is used in constructing the discrete, spectral element convection operator.*

2.4 The steady Navier-Stokes equations

We shall treat each component of the advection term in a similar fashion as the convection term in the steady convection-diffusion equation. Otherwise, we follow the same procedure as outlined for the steady Stokes problem. We then arrive at a set of discrete equations which can be expressed in matrix form as

$$\underline{\mathbf{A}}\,\underline{\mathbf{u}} + \underline{\mathbf{C}}(\underline{\mathbf{u}})\,\underline{\mathbf{u}} - \underline{\mathbf{D}}^T \underline{p} = \underline{\mathbf{f}}\,, \tag{27}$$

$$\underline{\mathbf{D}}\,\underline{\mathbf{u}} = \underline{0}\,, \tag{28}$$

where $\underline{\mathbf{C}}(\underline{\mathbf{u}})$ represents the discrete, nonlinear, nonsymmetric advection operator.

3 Iterative substructuring methods

Iterative substructuring methods are solution methods based on a decomposition of the original domain into nonoverlapping subdomains [10, 11, 21, 47]. This class of domain decomposition methods has reached a high degree of maturity over the past few years, in particular, for symmetric, positive definite, elliptic problems [20]. A general and powerful domain decomposition approach for solving the discrete Poisson problem (7) consists of first decomposing the finite-dimensional space V into a sum of $M+1$ subspaces [21, 41],

$$V = \sum_{i=0}^{M} V_i\,,$$

and then consider the solution of new, smaller subproblems associated with these subspaces. The space V_0 typically represents a global, coarse space, while V_i, $i = 1, ..., M$ are subspaces associated with the individual (local) subdomains, both interior and interfaces.

In terms of matrix algebra we can summarize the approach as follows: Instead of solving the original system of algebraic equations, equation (12), we consider the solution of a preconditioned (or transformed) system

$$\underline{B}^{-1}\underline{A}\,\underline{u} = \underline{B}^{-1}\,\underline{f}\,, \tag{29}$$

where the preconditioner $\underline{B}^{-1}$ is defined as

$$\underline{B}^{-1} = \sum_{i=0}^{M} \underline{B}_i^{-1} \tag{30}$$

with

$$\underline{B}_i^{-1} = \underline{R}_i\, \tilde{\underline{A}}_i^{-1}\, \underline{R}_i^T\,.$$

Here $\underline{A}$ is the matrix version of the symmetric, positive definite, bilinear form $a : V \times V \rightarrow \mathbf{R}$, while $\tilde{\underline{A}}_i$ is the matrix version of a symmetric, positive definite, bilinear form $\tilde{a}_i : V_i \times V_i \rightarrow \mathbf{R}$. The operator (or matrix) R_i extends the nodal representation of an element in the subspace V_i to an element in the global space V, while the operator $\underline{R}_i^T$ represents the associated restriction operator.

For each subspace V_i we also introduce the operator $T_i : V \rightarrow V_i$ such that $\forall v_\delta \in V$, $T_i v_\delta \in V_i$ is the solution of the following problem on V_i

$$\tilde{a}_i(T_i v_\delta, w_\delta) = a(v_\delta, w_\delta) \qquad \forall w_\delta \in V_i\,.$$

In the case that $\tilde{a}_i(\cdot,\cdot) = a(\cdot,\cdot)$ the operator T_i represents an orthogonal projection from V onto V_i. However, this framework also allows for the consideration of letting $\tilde{a}_i(\cdot,\cdot)$ represent an approximation to $a(\cdot,\cdot)$, a possibility that we will later exploit in several different ways.

If $\underline{v}$ is the nodal representation of an element $v_\delta \in V$, the above projection can be expressed in matrix form as

$$\underline{T}_i \underline{v} = \underline{B}_i^{-1} \underline{A} \underline{v} .$$

In particular, we can express the projection of the exact solution as

$$\underline{T}_i \underline{u} = \underline{\hat{f}}_i$$

where

$$\underline{\hat{f}}_i = \underline{B}_i^{-1} \underline{f} .$$

Equation (29) can now be expressed as

$$\underline{T}\,\underline{u} = \underline{\hat{f}}, \tag{31}$$

where $\underline{T} = \sum_{i=0}^{M} \underline{T}_i$ and $\underline{\hat{f}} = \sum_{i=0}^{M} \underline{\hat{f}}_i$.

The preconditioned (or transformed) system (31) is now typically solved by a Krylov method such the conjugate gradient method. We remark that if the operators $\underline{B}_i$ (or $\underline{T}_i$) are well chosen, the operator $\underline{T} = \underline{B}^{-1} \underline{A}$ will be quite well conditioned, and the iterative procedure will converge rapidly. From (30) we see that each preconditioning step consists of solving $M+1$ subproblems. Because of the additive nature of the preconditioner, the solution of these subproblems can be performed in parallel. As we shall discuss more later, the use of inexact solvers for the local subproblems will also allow us to make each preconditioning step inexpensive relative to the cost of performing global matrix vector products, resulting in a cost-effective solution algorithm.

In the next section we shall apply the additive Schwarz procedure outlined above to solve the Poisson problem (12) in the context of spectral element discretizations. In Section 3.2 we shall propose an extension of the above procedure to solve the steady Stokes problem, and in Section 3.3 and Section 3.4 we shall propose further extensions in order to solve the steady convection-diffusion equation and the steady Navier-Stokes equations, respectively.

3.1 An iterative substructuring method for the Poisson problem

Here, we consider the solution of the Poisson problem (12) discretized using spectral elements. The method we propose employs the following decomposition of the discrete space V:

$$V = V_0 + \sum_{k=1}^{K} V_k + V_\Gamma . \tag{32}$$

With V defined in (10), we also define the particular realization V_{N_0} to mean that a fixed polynomial approximation N_0 is used in every element. With this notation, the space V_0 can be defined as

$$V_0 = V_{N_0} \qquad 1 \le N_0 < N . \tag{33}$$

The space V_0 is thus associated with a coarse discretization of the original problem. Previous studies have demonstrated the importance of including a coarse problem as part of the preconditioner in order to allow for a global information transfer mechanism [52].

The subspace V_k is associated with an individual spectral element (or subdomain), and is defined as

$$V_k = \{v : \ v \in Q_N(\Omega_k), \ v_{|\partial\Omega_k} = 0\} . \tag{34}$$

Finally, the space V_Γ is defined as

$$V_\Gamma = \{v : \ v = w_{|\Gamma}, \ w \in V\} , \tag{35}$$

where Γ refers to the collection of all the edges $\Gamma_{k,l}$ defined in (6) for two-dimensional problems, and faces for three-dimensional problems.

In terms of the basis for the subspaces V_0, $V_k, k = 1, ..., K$, and V_Γ, we use a nodal, Lagrangian interpolant basis defined in terms of the tensor product Gauss-Lobatto Legendre points, similar to the basis for the global space V. Note that an element in the space V_Γ is extended by zero from the element interfaces to the GLL nodes in the interior of the elements.

We now discuss the approximate projection operators T_0, $T_k, k = 1, ..., K$, and T_Γ associated with these subspaces. First, we let T_0 represent an orthogonal projection from V to V_0, i.e., $\tilde{a}_0(\cdot,\cdot) = a(\cdot,\cdot)$. In matrix form this means that we can express $\underline{\tilde{A}}_0$ as

$$\underline{\tilde{A}}_0 = \underline{R}_0^T \underline{A}\, \underline{R}_0 . \tag{36}$$

Here the prolongation operator $\underline{R}_0$ represents an operator which takes an element in V_0 (the coarse global space) and represents it in terms of the basis for the global space V. In practice, this is done by taking a global coarse solution and performing an interpolation in each spectral element from a polynomial order N_0 to a polynomial order N.

Next, we let T_k represent an approximate projection from V_k to V. In particular, we let $\tilde{a}_k(\cdot,\cdot)$ (or $\underline{\tilde{A}}_k$ in matrix form) represent a linear finite element approximation associated with the GLL points,

$$\underline{\tilde{A}}_k = \underline{A}_{k,FE} . \tag{37}$$

In two space dimensions we use linear triangular elements based on the GLL nodes, while in three space dimensions

we use linear tetrahedral elements based on the GLL nodes. Earlier studies have shown that such a finite element preconditioner is spectrally close to the original local spectral operator (denoted in matrix form as $\underline{A}_k$), with a condition number bounded by a constant as the polynomial degree N increases [19]. The reason for using a finite element preconditioner is that it reduces the computational complexity associated with solving the subproblems for the individual spectral elements, while still resulting in a good conditioning of the transformed problem (31).

Finally, we consider the approximate projection operator T_Γ. In order to compute the nodal values along the subdomain interfaces, we shall simply use the diagonal of the discrete Laplace operator, i.e.,

$$\tilde{\underline{A}}_\Gamma = diag(\underline{A})_{|\Gamma} \,. \tag{38}$$

A better choice would, of course, be to use an approximation to the Schur complement on the subdomain interfaces. However, as we shall see later, the simple diagonal preconditioner (38) gives remarkably good results. This is particularly true when we later on consider solution algorithms for the steady Stokes problem and for the steady Navier-Stokes problem.

In summary, the preconditioner $\underline{B}^{-1}$ that we use can be expressed as

$$\underline{B}^{-1} = \underline{B}_0^{-1} + \sum_{k=1}^{K} \underline{B}_k^{-1} + \underline{B}_\Gamma^{-1} \tag{39}$$

where

$$\begin{aligned} \underline{B}_0^{-1} &= \underline{R}_0 \, \tilde{\underline{A}}_0^{-1} \underline{R}_0^T \;, \\ \underline{B}_k^{-1} &= \underline{R}_k \, \tilde{\underline{A}}_k^{-1} \underline{R}_k^T \;, \; k = 1, ..., K \;, \\ \underline{B}_\Gamma^{-1} &= \underline{R}_\Gamma \, \tilde{\underline{A}}_\Gamma^{-1} \underline{R}_\Gamma^T \;. \end{aligned}$$

The system (29) is now solved by the conjugate gradient method. We remark that $K + 2$ subproblems need to be solved for each iteration, see (39).

The coarse system matrix $\tilde{\underline{A}}_0$ is explicitly assembled and then factored using a banded direct solver for symmetric systems from the LINPACK library. Hence, only back-substitution is needed during the iteration. If N_0 is small, both the number of unknowns and the bandwidth will be small.

The local systems matrices $\tilde{\underline{A}}_k, k = 1, ..., K$ are also explicitly assembled and then factored using a banded direct solver for symmetric systems (from LINPACK). We remark that the bandwidth for the finite element approximation is a factor of N smaller than the bandwidth for the original local spectral operators $\underline{A}_k, k = 1, ..., K$. The operator $\underline{R}_k$ extends the solution in Ω_k by zero to all the other subdomains. Hence, $\underline{R}_k$ represents the identity operator for the nodal values associated with subdomain Ω_k, and the zero operator for the nodal values associated with the rest of the computational domain.

The matrix $\tilde{\underline{A}}_\Gamma$ is diagonal, which makes the inversion of this operator trivial. The operator $\underline{R}_\Gamma$ represents the identity operator for the degrees-of-freedom associated with the element interfaces, and the zero operator for the degrees-of-freedom associated with the interior of the elements.

3.2 An iterative substructuring method for the steady Stokes problem

Iterative substructuring methods have shown great promise for solving symmetric and nonsymmetric systems of equations [35, 47, 14, 53, 41, 16]. However, there are still very limited results and experience from applying such algorithms directly to solving the discrete Stokes or Navier-Stokes equations [4].

In the past the most commonly used methods to solve (21) - (22) have either been iterative methods based on a global Uzawa decoupling procedure [26, 27, 31] or direct solvers. The large bandwidth typcially associated with spectral element discretizations makes direct solvers practical only for relatively small two-dimensional problems. In order to solve three-dimensional systems, it is imperative to have good iterative solvers. Even though a global Uzawa procedure results in a relatively well-conditioned system for regular geometries [31], the convergence rate can deteriorate significantly for irregular computational domains (e.g., large aspect ratios).

In this section we shall propose an iterative substructuring method for solving the steady Stokes equations in two or three space dimensions. Throughout the rest of this section, we assume that the Stokes problem is discretized using spectral elements, that is, we assume a decomposition of the original domain into K spectral elements (or subdomains), and a high-order, tensor-product, polynomial approximation inside each element. However, we remark that the method we propose should in general work for systems based upon p- or h-p-type finite element methods [5, 3, 4, 37].

The general approach will be similar to the additive Schwarz method for the elliptic Poisson problem, namely, a decomposition of the finite-dimensional velocity and pressure spaces V^d $(d = 2, 3)$ and W in order to define smaller and more tractable subproblems: (i) a coarse Stokes problem defined over the entire domain Ω; (ii) local Stokes problems associated with the individual spectral elements Ω_k, $k = 1, ..., K$; and (iii) a Poisson type subproblem asso-

ciated with the subdomain interfaces Γ. Finally, (iv) due to the fact that the (fully coupled) steady Stokes problem represents an indefinite saddle problem, a global iterative scheme will be based on the GMRES method [46] .

We start by proposing the following decomposition of the finite-dimensional spaces V^d and W for the velocity and the pressure in d space dimensions.

$$V^d = V_0^d + \sum_{k=1}^{K} V_k^d + V_\Gamma^d ,$$
$$W = W_0 + \sum_{k=1}^{K} W_k .$$

A coarse Stokes problem is associated with the subspaces V_0^d and W_0. Similar to the definition of V_0 in (33), we define the coarse velocity and pressure spaces as

$$V_0^d = V_{N_0}^d \qquad 2 \le N_0 < N ,$$
$$W_0 = W_{N_0-2} .$$

With this definition of V_0^d and W_0, the coarse Stokes problem is defined as the particular realization of the original Stokes problem, defined by K conforming spectral elements and assuming a (fixed) polynomial degree N_0 for the velocity and $N_0 - 2$ for the pressure inside each spectral element (or subdomain). In other words, the coarse Stokes problem is the standard $\mathbf{P}_N/\mathbf{P}_{N-2}$ method [34, 1] with $N = N_0 \ge 2$ in each spectral element.

In practice, the polynomial degree N_0 that we use for the coarse Stokes problem cannot be too high; a typical value for N_0 is 2 or 3. Hence, a typical spectral element for the coarse Stokes problem is either a Q_2/Q_0 element, or a Q_3/Q_1 element. The main reason for this choice is that larger values for N_0 make the solution of the coarse Stokes problem too expensive. This is particularly true when considering a direct solver for the coarse problem. For three-dimensional problems only a quadratic element might be practical. However, in this case the Q_2/Q_0 spectral element is expected to be inferior to the Q_2/P_1 finite element [25]. We shall therefore also consider the use of low order finite elements for the coarse problem, see Section 4 for numerical results.

We now proceed by considering the subproblems associated with the individual spectral elements. For each spectral element (or subdomain) Ω_k we define a local Stokes problem with homogeneous velocity boundary conditions. That is, for each spectral element $\Omega_k, k = 1, ..., K$ we search for a discrete velocity $\mathbf{u}_{\delta,k} \in V_k^d$ and a discrete pressure $p_{\delta,k} \in W_k$, where

$$V_k^d = [Q_N(\Omega_k) \cap H_0^1(\Omega_k)]^d , \; k = 1, ..., K$$

$$W_k = \{ w_k : \; w_k \in Q_{N-2}(\Omega_k), \; \int_{\Omega_k} w_k \, d\Omega = 0 \} .$$

The space V_Γ^d is defined as

$$V_\Gamma^d = \{ v : \; v = w_{|\Gamma}, \; w \in V^d \} .$$

Hence, V_Γ^d is defined similarly to V_Γ for the Poisson problem. We remark that there are no pressure degrees-of-freedom along Γ.

We are now in a position to propose an additive Schwarz algorithm for the Stokes problem. We start by first expressing the original discrete Stokes equations (21)-(22) in the compact form

$$\underline{\mathbf{S}}\,\underline{\mathbf{x}} = \underline{\mathbf{g}} \tag{40}$$

where

$$\underline{\mathbf{S}} = \begin{pmatrix} \underline{\mathbf{A}} & -\underline{\mathbf{D}}^T \\ \underline{\mathbf{D}} & \underline{\mathbf{0}} \end{pmatrix}$$
$$\underline{\mathbf{x}} = [\underline{\mathbf{u}}, \underline{p}]^T$$
$$\underline{\mathbf{g}} = [\underline{\mathbf{g}}_u, \underline{g}_p]^T \equiv [\underline{\mathbf{f}}, \underline{0}]^T .$$

We then consider the preconditioned (transformed) Stokes system

$$\underline{\mathbf{Q}}^{-1}\underline{\mathbf{S}}\,\underline{\mathbf{x}} = \underline{\mathbf{Q}}^{-1}\underline{\mathbf{g}} , \tag{41}$$

where $\underline{\mathbf{Q}}^{-1}$ represents the Stokes preconditioner, that is, an operator that approximates the inverse of the original discrete Stokes operator and is relatively inexpensive to evaluate. Before we discuss the Stokes preconditioner, we remark that the indefinite, nonsymmetric system (41) is solved using a global iterative procedure based upon GMRES. For each global iteration we need to perform *two* global matrix-vector products of the type $\mathbf{y} = \underline{\mathbf{Q}}^{-1}\underline{\mathbf{S}}\,\mathbf{x}$. If the preconditioner is well chosen, the number of iterations will be small, and not very sensitive to the number of spectral elements K or the polynomial degree N associated with each element (or subdomain) $\Omega_k, k = 1, ..., K$.

We now proceed with discussing the Stokes preconditioner $\underline{\mathbf{Q}}^{-1}$, which we define as

$$\underline{\mathbf{Q}}^{-1} = \underline{\mathbf{Q}}_0^{-1} + (\mathbf{I} + \underline{\mathbf{Q}}_\Gamma^{-1}\underline{\mathbf{G}}) [\sum_{k=1}^{K} \underline{\mathbf{Q}}_k^{-1}] + \underline{\mathbf{Q}}_\Gamma^{-1} \tag{42}$$

where

$$\underline{\mathbf{Q}}_0^{-1} = \underline{\mathbf{R}}_0 \tilde{\underline{\mathbf{S}}}_0^{-1} \underline{\mathbf{R}}_0^T ,$$
$$\underline{\mathbf{Q}}_k^{-1} = \underline{\mathbf{R}}_k \tilde{\underline{\mathbf{S}}}_k^{-1} \underline{\mathbf{R}}_k^T , \; k = 1, ..., K ,$$
$$\underline{\mathbf{Q}}_\Gamma^{-1} = \underline{\mathbf{R}}_\Gamma \tilde{\underline{\mathbf{S}}}_\Gamma^{-1} \underline{\mathbf{R}}_\Gamma^T .$$

The first part of the preconditioner represents the solution of a coarse Stokes problem similar to the solution of a coarse Poisson problem in (30). A coarse version of the original Stokes problem can be expressed as

$$\underline{\mathbf{S}}_0 \, \underline{\mathbf{x}}_0 = \underline{\mathbf{g}}_0 \, .$$

Here the subscript zero indicates that we are searching for a solution in the subspaces V_0^d and W_0 instead of the original spaces V^d and W. In addition, the subscript zero indicates that a low (fixed) polynomial degree N_0 is used in order to construct the individual discrete operators in $\underline{\mathbf{S}}_0$ as well as the right hand side $\underline{\mathbf{g}}_0$. The operator $\tilde{\underline{\mathbf{S}}}_0$ that we use in the preconditioner (42) is simply

$$\tilde{\underline{\mathbf{S}}}_0 = \underline{\mathbf{S}}_0 \, .$$

The prolongation operator $\underline{\mathbf{R}}_0$ can be expressed as

$$\underline{\mathbf{R}}_0 = \begin{pmatrix} \underline{\mathbf{R}}_{u,0} & \underline{\mathbf{0}} \\ \underline{\mathbf{0}} & \underline{R}_{p,0} \end{pmatrix} .$$

Here, $\underline{\mathbf{R}}_{u,0}$ represents an operator which takes an element in V_0^d and represents it in terms of the basis for the global space V^d, while $\underline{R}_{p,0}$ represents an operator which takes an element in W_0 and represents it in terms of the basis for the global space W. In practice, this is done by taking a global coarse Stokes solution $(\underline{\mathbf{u}}_0, \underline{p}_0)$, and performing an interpolation in each spectral element from a polynomial order $(N_0, N_0 - 2)$ to a polynomial order $(N, N-2)$ in all Ω_k, $k = 1, ..., K$ (in the case of spectral elements).

We remark that when we apply this coarse Stokes operator as part of the preconditioner (42), the associated right hand side $\underline{\mathbf{g}}_0 = [\underline{\mathbf{g}}_{u,0}, \underline{g}_{p,0}]^T$ will in general be nonzero, including $\underline{g}_{p,0}$. In our implementation, the coarse Stokes operator is explicitly assembled and then factored using a banded direct solver from the LINPACK library. Hence, only back substitution is necessary during each GMRES iteration.

We now proceeed by expressing the local (spectral) Stokes problems associated with the individual spectral elements:

$$\underline{\mathbf{S}}_k \, \underline{\mathbf{x}}_k = \underline{\mathbf{g}}_k \, , \; k = 1, ..., K \, , \tag{43}$$

where

$$\underline{\mathbf{S}}_k = \begin{pmatrix} \underline{\mathbf{A}}_k & -\underline{\mathbf{D}}_k^T \\ \underline{\mathbf{D}}_k & \underline{\mathbf{0}} \end{pmatrix} . \tag{44}$$

Note that subscript k here refers to a particular subdomain Ω_k, and should not be confused with summation over repeated indices.

The operator $\tilde{\underline{\mathbf{S}}}_k$ that we use in the preconditioner (42) will be based upon a modified (approximate) version of $\underline{\mathbf{S}}_k$ in (44), defined as

$$\tilde{\underline{\mathbf{S}}}_k = \begin{pmatrix} \tilde{\underline{\mathbf{A}}}_k & -\underline{\mathbf{D}}_k^T \\ \underline{\mathbf{D}}_k & \underline{\mathbf{0}} \end{pmatrix} .$$

The original (spectral) viscous operator in (44) is here replaced by a finite element operator; this operator is derived by using linear finite elements on the GLL nodes in a similar fashion as the subdomain preconditioner for the Poisson problem. Hence, instead of solving (43) we solve

$$\tilde{\underline{\mathbf{S}}}_k \, \tilde{\underline{\mathbf{x}}}_k = \underline{\mathbf{g}}_k \, , \; k = 1, ..., K \, . \tag{45}$$

Again, we remark that the right hand side $\underline{\mathbf{g}}_k = [\underline{\mathbf{g}}_{u,k}, \underline{g}_{p,k}]^T$ will in general be nonzero, including $\underline{g}_{p,k}$. The prolongation operators $\underline{\mathbf{R}}_k$, $k = 1, ..., K$ are defined in an analogous fashion to the Poisson problem: each operator represents an identity operator for the degrees-of-freedom associated with Ω_k, and the zero operator for the remaining degrees-of-freedom.

We solve the coupled, saddle Stokes system (45) by first applying a Uzawa procedure, that is, by applying a block 2×2 Gaussian elimination. The result is a decoupling of the pressure and the velocity into a positive semi-definite system for the pressure and a positive definite system for the velocity:

$$\tilde{\underline{U}}_k \underline{p}_k = \underline{g}_{p,k} - \underline{\mathbf{D}}_k \, \tilde{\underline{\mathbf{A}}}_k^{-1} \underline{\mathbf{g}}_{u,k} \, , \tag{46}$$

$$\underline{\mathbf{A}}_k \, \underline{\mathbf{u}}_k = \underline{\mathbf{g}}_{u,k} + \underline{\mathbf{D}}_k^T \, \underline{p}_k \, , \tag{47}$$

where the Uzawa pressure operator $\tilde{\underline{U}}_k$ is defined as

$$\tilde{\underline{U}}_k = \underline{\mathbf{D}}_k \, \tilde{\underline{\mathbf{A}}}_k^{-1} \, \underline{\mathbf{D}}_k^T \, . \tag{48}$$

We construct $\tilde{\underline{U}}_k$ explicitly, and factor the matrix using a symmetric solver from the LINPACK library. Hence, only back substitution is necessary during the global iteration. Note that the Uzawa pressure system is singular, reflecting the fact that the pressure $\underline{p}_k$ is only determined up to a constant. In order to obtain solvability we therefore fix the pressure to be zero at a single (interior) GLL point, and later adjust the pressure level such that the average pressure is zero in each subdomain. The correct pressure level in each subdomain is actually provided by the *coarse* problem. We now make the important observation that the coarse problem is not only necessary in order to improve the conditioniong of the transformed problem (41); the coarse problem is, in fact, *essential* in order to compute the correct solution.

Once the pressure $\underline{p}_k$ has been computed, we can solve for the velocity by inverting the viscous operator. Again,

as for the Poisson problem, we form explicitly the *scalar*, finite element based Poisson operator. Next, we factor this scalar operator using a banded, direct solver from LINPACK. The velocity $\underline{\mathbf{u}}_k$ is then computed by performing d back-substitutions, one for each velocity component.

Next, having computed $\underline{\mathbf{x}}_k = [\underline{\mathbf{u}}_k, \underline{p}_k]^T$ for all Ω_k, $k = 1, ..., K$, we apply the gradient operator $\underline{\mathbf{G}}$ defined as

$$\underline{\mathbf{G}} = \begin{pmatrix} \underline{\mathbf{0}} & +\underline{\mathbf{D}}^T \\ \underline{\mathbf{0}} & \underline{\mathbf{0}} \end{pmatrix} . \tag{49}$$

The result from this operation, restricted to Γ, is added to the original contribution along the interfaces Γ. The nodal values along Γ (velocity degrees-of-freedom only) are then computed by inverting the diagonal of the viscous operator, that is,

$$\tilde{\underline{\mathbf{S}}}_\Gamma = diag(\underline{\mathbf{A}})_{|\Gamma} . \tag{50}$$

Finally, the prolongation operator $\underline{\mathbf{R}}_\Gamma$ represents the identity operator for the *velocity* degrees-of-freedom along Γ, and the zero operator for the remaining degrees-of-freedom in the domain. Again, we remark that there are no pressure degrees-of-freedom along the interface Γ.

3.3 An iterative substructuring method for the steady convection-diffusion equation

We are here interested in solving the discrete system (26) using an iterative substructuring approach. As a starting point we shall use the algorithm presented for the Poisson problem, which corresponds to solving the system (26) without any convection.

The addition of the linear convection term modifies the Poisson algorithm in two ways. First, the system is no longer symmetric, so we have to replace the conjugate gradient algorithm with a GMRES algorithm. This means that two global matrix-vector products are required for each iteration (as opposed to one for the conjugate gradient algorithm).

Second, the Poisson preconditioner (39) is modified in the following way: The coarse problem corresponds to a coarse discretization of the original convection-diffusion problem, including the convection term, but the local problems and the interface problem are left unchanged.

It is well known that using a coarse, low order discretization to resolve convection-diffusion problems will produce wiggles. For the coarse problem we therefore add an anisotropic diffusion term [28], which is equivalent to the incorporation of a streamline upwinding procedure [13].

The modified diffusivity can be expressed as

$$\alpha_{ij} = \alpha_0 + \tilde{\alpha}_{ij}$$

where α_0 is the original (isotropic) diffusivity in (23). The added symmetric diffusivity tensor $\tilde{\alpha}_{ij}$ at a particular (integration) point in space can be expressed as in [28]

$$\tilde{\alpha}_{ij} = c \frac{U H}{2} (\frac{u_i u_j}{U^2}) .$$

Here, H is the local mesh spacing, U is the magnitude of the velocity, and $u_i, i = 1, .., d$ are the corresponding velocity components. The constant c is chosen such that the grid Peclet number is less than 2 everywhere on the coarse grid. We note that there is no diffusion in the direction perpendicular to a streamline, hence the name streamline diffusion (or streamline upwinding).

3.4 An iterative substructuring method for the steady Navier-Stokes problem

We start by first expressing the original, nonsymmetric, nonlinear, discrete steady Navier-Stokes system (27)-(28) in the compact form

$$\underline{\mathbf{F}}\,\underline{\mathbf{x}} = \underline{\mathbf{g}} , \tag{51}$$

where

$$\begin{aligned} \underline{\mathbf{F}} &= \begin{pmatrix} (\underline{\mathbf{A}} + \underline{\mathbf{C}}(\mathbf{u})) & -\underline{\mathbf{D}}^T \\ \underline{\mathbf{D}} & \underline{\mathbf{0}} \end{pmatrix} , \\ \underline{\mathbf{x}} &= [\underline{\mathbf{u}}, \underline{p}]^T , \\ \underline{\mathbf{g}} &= [\underline{\mathbf{g}}_u, \underline{g}_p]^T \equiv [\underline{\mathbf{f}}, \underline{0}]^T . \end{aligned}$$

As usual we linearize the system (51) and perform a Newton iteration. For each iteration we have to solve a system of the form

$$\underline{\mathbf{N}}\,\delta\underline{\mathbf{x}}^n = \underline{\mathbf{g}} - \underline{\mathbf{F}}\,\underline{\mathbf{x}}^{n-1} \tag{52}$$

where $\underline{\mathbf{N}}$ represents the linearized Navier-Stokes operator, $\underline{\mathbf{x}}^n$ is the solution after n Newton iterations, and $\delta\underline{\mathbf{x}}^n = \underline{\mathbf{x}}^n - \underline{\mathbf{x}}^{n-1}$.

The iterative substructuring algorithm we now present is for the system (52). Our method can therefore be described as a Newton-Krylov method. We proceed by directly considering the preconditioned (transformed), linearized, steady Navier-Stokes system

$$\underline{\mathbf{M}}^{-1}\underline{\mathbf{N}}\,\delta\underline{\mathbf{x}}^n = \underline{\mathbf{M}}^{-1}(\underline{\mathbf{g}} - \underline{\mathbf{F}}\,\underline{\mathbf{x}}^{n-1}) ,$$

where $\underline{\mathbf{M}}^{-1}$ represents the Navier-Stokes preconditioner.

The Navier-Stokes preconditioner can be expressed in a form similar to that for the steady Stokes equations,

$$(53)\quad \underline{\mathbf{M}}^{-1} = \underline{\mathbf{M}}_0^{-1} + (\mathbf{I} + \underline{\mathbf{M}}_\Gamma^{-1}\underline{\mathbf{G}})\,[\sum_{k=1}^{K} \underline{\mathbf{M}}_k^{-1}] + \underline{\mathbf{M}}_\Gamma^{-1}$$

where

$$\begin{aligned} \underline{\mathbf{M}}_0^{-1} &= \underline{\mathbf{R}}_0\,\tilde{\underline{\mathbf{N}}}_0^{-1}\underline{\mathbf{R}}_0^T \ , \\ \underline{\mathbf{M}}_k^{-1} &= \underline{\mathbf{R}}_k\,\tilde{\underline{\mathbf{N}}}_k^{-1}\underline{\mathbf{R}}_k^T \ , \ k = 1, ..., K \ , \\ \underline{\mathbf{M}}_\Gamma^{-1} &= \underline{\mathbf{R}}_\Gamma\,\tilde{\underline{\mathbf{N}}}_\Gamma^{-1}\underline{\mathbf{R}}_\Gamma^T \ . \end{aligned}$$

Our choice for the individual components of this preconditioner will be:

$$\begin{aligned} \tilde{\underline{\mathbf{N}}}_0 &= \underline{\mathbf{N}}_{0,FE,SU} \ , \\ \tilde{\underline{\mathbf{N}}}_k &= \tilde{\underline{\mathbf{S}}}_k \ \ k = 1, ..., K \ , \\ \tilde{\underline{\mathbf{N}}}_\Gamma &= diag(\underline{\mathbf{A}})_{|\Gamma} \ . \end{aligned}$$

Here, $\underline{\mathbf{N}}_{0,FE,SU}$ represents a coarse discretization of the original, linearized Navier-Stokes operator. For this coarse discretization we use low-order finite elements (or low-order spectral elements) on the original spectral element decomposition. In addition, we also add streamline diffusion in a similar fashion as for the convection-diffusion problem. Hence, our Navier-Stokes preconditioner is based upon a *hierarchy* of discrete, spatial operators, starting with a linearized Navier-Stokes operator for the coarse, global problem, a steady Stokes (mixed) operator for each individual, local problem, and finally, an elliptic (Poisson type) operator for the interface problem.

Our experience has been that using streamline upwinding on the coarse grid is perhaps most useful as a means of obtaining a good initial condition at a very low computational cost. A good initial condition reduces the initial residual on the fine grid (and thus the overall cost), and it also provides a good starting point for the initial linearization in (52). Using upwinding in the construction of $\tilde{\underline{\mathbf{N}}}_0$ does not always seem to make a substantial difference. However, more testing is necessary in order to quantify this effect more precisely.

4 Numerical results

The purpose of this section is to explore the behavior of the algorithms that we have just presented. We will study the conditioning of the elliptic systems together with the convergence rate for the Stokes systems. We shall also study the steady convection-diffusion problem as well as the full Navier-Stokes problem. Finally, we will compute the error of some model problems in order to verify that we indeed end up with the correct solution when we apply these algorithms.

4.1 The Poisson problem

We shall first study the solution of the Poisson problem (1)-(2) in a domain $\Omega =]0,1[^d$, $d = 2, 3$. We choose a forcing function f such that the exact solution u can be expressed as

$$\begin{aligned} u(x,y) &= u_1(x) \otimes u_1(y) && (d=2) \\ u(x,y,z) &= u_1(x) \otimes u_1(y) \otimes u_1(z) && (d=3) \end{aligned}$$

where

$$u_1(t) = t\,(1 - e^{\beta(t-1)}) \ .$$

We have chosen f such that the exact solution represent a tensor-product, "boundary-layer" type solution. In all the numerical experiments we use a value $\beta = 10$. We remark that the solution cannot be represented exactly by polynomials, and does not represent an eigenfunction of the Poisson operator.

We break up the computational domain Ω into K square or cubic spectral elements, each element being of order N. For each discretization (characterized by K and N), we shall compute the condition number $\kappa^B = \lambda_{max}/\lambda_{min}$ for the preconditioned system (29). This is equivalent to considering the following eigenvalue problem:

$$\underline{A}\,\underline{\chi} = \lambda\,\underline{B}\,\underline{\chi} \ ,$$

where $\underline{B}$ is the preconditioner defined in (39), $\underline{\chi}$ represents an eigenvector, and λ represents the corresponding eigenvalue.

We start by first looking at the special case $K = 1$, i.e, the pure spectral case. In this case the preconditioner $\underline{B}$ does not include any interface system $\underline{B}_\Gamma$. For illustration we shall not include any coarse system either. Hence, the preconditioner $\underline{B}$ consists entirely of a finite element system constructed as a triangulation/tetrahedrazation based on the tensor-product Gauss-Lobatto Legendre nodes.

Finite element preconditioning of spectral systems has been used with success in different contexts [19, 19, 24], and we shall here verify the good conditioning properties for this Galerkin based discretization of the Poisson problem.

Table 1 reports the computed condition numbers that we obtain in $d = 2$ and $d = 3$ space dimensions for practical values of the polynomial degree N associated with the spectral operator. As expected, the condition number is $\mathcal{O}(1)$.

N	d=2	d=3
3	1.34	1.59
4	1.55	2.01
5	1.70	2.31
6	1.79	2.51
7	1.82	2.44
8	1.89	2.48
9	1.95	2.52
10	1.99	2.57
11	2.03	2.61
12	2.06	2.65

Table 1: Condition number κ^B $(K = 1)$

We proceed by now considering the multi-domain case. Unless otherwise stated we construct a coarse, global problem based upon K elements, each of order $N_0 = 2$. Table 2 reports the condition number for the two-dimensional case, while Table 3 reports the condition number for the three-dimensional case. The results indicate that the condition number κ^B is independent of the number of elements, K, and grows approximately like N^2. This is also consistent with our experience that the number of conjugate gradient iterations grows approximately linearly with N.

N	$K = 16$	$K = 64$	$K = 256$
3	3.65	3.70	3.65
4	5.14	5.20	5.22
5	7.48	7.55	7.59
6	10.3	10.4	10.5
7	13.4	13.5	13.6
8	17.2	17.3	17.4
9	21.2	21.3	21.4
10	25.8	26.0	26.1
11	30.7	30.9	
12	36.3	36.5	

Table 2: Condition number κ^B $(d = 2)$

N	$K = 27$	$K = 64$	$K = 125$
3	5.49	5.60	5.65
4	7.88	7.97	7.99
5	12.2	12.1	12.1
6	17.4	17.6	17.5
7	23.6	23.6	23.5
8	31.0	31.0	31.0
9	39.2	39.2	39.2

Table 3: Condition number κ^B $(d = 3)$

In order to verify that the solution algorithm indeed computes the correct solution, we also compute the error $\| u - u_\delta \|$ between the exact solution u and the numerical solution u_δ in the relative semi-norm. For the two-dimensional case $(d = 2)$ we use $K = 4$ spectral elements, each of order N. For the three-dimensional case $(d = 3)$ we use $K = 8$ spectral elements, each of order N. For the error calculation we use the discrete semi-norm, however, we use a finer mesh in order to avoid quadrature errors. For the results presented for the Poisson equation, we use a polynomial degree $M = N + 3$ inside each element in the error calculation. The results are reported in Table 4. As expected, exponential convergence is achieved as the polynomial order N is increased with K fixed. It is interesting to notice that the relative error is essentially independent of the number of spatial dimensions; this is most likely due to the tensor-product form of the exact solution.

N	$d = 2$	$d = 3$
3	$2.23 \cdot 10^{-1}$	$2.25 \cdot 10^{-1}$
4	$6.55 \cdot 10^{-2}$	$6.62 \cdot 10^{-2}$
5	$1.62 \cdot 10^{-2}$	$1.64 \cdot 10^{-2}$
6	$3.44 \cdot 10^{-3}$	$3.46 \cdot 10^{-3}$
7	$6.30 \cdot 10^{-4}$	$6.34 \cdot 10^{-4}$
8	$1.02 \cdot 10^{-4}$	$1.02 \cdot 10^{-4}$
9	$1.46 \cdot 10^{-5}$	$1.47 \cdot 10^{-5}$

Table 4: Discretization error $\| u - u_\delta \| / \| u \|$

Finally, we look at the Poisson model problem in a "stretched" two-dimensional domain $\Omega =]0, \alpha[\times]0, 1[$. Hence, the domain aspect ratio is equal to α. The computational domain is now broken up into K rectilinear spectral elements, each of order N and with an element aspect ratio equal to α_k, $k = 1, ..., K$.

In the first experiment, we choose the domain aspect ratio $\alpha = 10$ and the element (subdomain) aspect ratio $\alpha_k = 1$, $k = 1, ..., K$. In order to realize this choice, we choose $K_1 = 40$ elements in the x-direction, and $K_2 = 4$ elements in the y-direction. Hence, the total number of elements is $K = K_1 \times K_2 = 160$. In the second experiment we choose $\alpha = 10$ and $\alpha_k = 10$, $k = 1, ..., K$. Here, we use $K_1 = K_2 = 4$, i.e., $K = 16$. In each case we compute the condition number κ^B for the preconditioned system (29). The results are reported in Table 5.

We see that the results for the first case, with $\alpha = 10$ and $\alpha_k = 1$, are almost identical to the results reported in Table 2 for the case $\alpha = 1$ and $\alpha_k = 1$. Hence, we conclude that the condition number of the preconditioned system is insensitive to the *domain* aspect ratio. However, the second case, with $\alpha = 10$ and $\alpha_k = 10$, indicates that the condition number is strongly dependent upon the *subdomain* aspect ratio.

N	$\alpha_k = 1$	$\alpha_k = 10$
3	3.70	53.9
4	5.20	75.9
5	7.58	110
6	10.4	150
7	13.6	193
8	17.4	244
9	21.4	299
10	26.1	363
11	31.0	429
12	36.6	505

Table 5: Condition number κ^B ($\alpha = 1$)

4.2 The steady Stokes problem

In this section we shall study the convergence rate for the preconditioned Stokes system (41). We shall use the standard driven cavity problem in $d = 2$ and $d = 3$ space dimensions as a sample problem. Since the steady Stokes system represents a saddle problem, we will not report the condition number as we did for the Poisson problem, but rather the number of GMRES iterations, m^Q, required in order to reduce the initial residual with five orders of magnitude.

While the original grid is based on K spectral elements, each of order N, the coarse grid will be based on K finite elements of the type Q_2/P_1. As discussed earlier, our experience has been that the Q_2/P_1 element is, in general, better than the Q_2/Q_0 element, which is the lowest order spectral element that we can use. This finding is also consistent with previous studies [25]. As expected, our experience is also that a Q_3/Q_1 element is even better. However, this element is expensive to use for large three-dimensional Stokes problems given the fact that we are using a direct banded solver for the coarse, global problem. For the results that we report in the following, the coarse grid is based upon using Q_2/P_1 elements on the original spectral element decomposition.

N	$K = 16$	$K = 64$	$\mathcal{N}_{d.o.f.}$
3	17	17	1,314
4	21	21	2,498
5	22	23	4,066
6	26	27	6,018
7	30	30	8,354
8	32	34	11,074
9	35	38	14,178
10	38	41	17,666

Table 6: Number of iterations m^Q ($d = 2$)

N	$K = 27$	$K = 64$	$\mathcal{N}_{d.o.f.}$	CPU
3	25	25	4,505	5 min.
4	31	32	11,853	9 min.
5	35	39	24,673	18 min.
6	43	46	44,501	34 min.

Table 7: Number of iterations m^Q ($d = 3$)

In Table 6 and Table 7 we report our results. We notice that, similar to the Poisson problem, the number of GMRES iterations, m^Q, seems to be rather insensitive to the number of elements (or subdomains), K. The number of iterations seems to grow approximately linearly with respect to the element order, N. In Table 6 and Table 7 we have also included the number of velocity and pressure degrees-of-freedom, $\mathcal{N}_{d.o.f.}$, for the case with the largest number of elements ($K = 64$). For the three-dimensional case, see Table 7, we have also included the total CPU time required in order to solve the for the corresponding number of degrees-of-freedom, starting from a zero initial condition. The computer we used for these experiments was a Sun Sparc II workstation with 64 MB of memory. All the calculations were done in double precision. It is interesting to notice that the CPU time per d.o.f. stays almost constant.

4.3 The steady convection-diffusion problem

We shall here solve the standard two-dimensional driven cavity problem as well as an associated convection-diffusion heat transfer problem; the computational domain is $\Omega =]0,1[^2$. The boundary conditions for the heat transfer problem is $u = 1$ along $x = 1$, $u = 0$ along $x = 0$, and insulated (zero Neumann) conditions along $y = 0$ and $y = 1$. We shall solve the problem corresponding to a Reynolds number $Re = 100$ and a Peclet number $Pe = 100$. We first solve the fluid problem, and then solve the associated steady heat transfer problem.

In Table 8 we report the number of iterations, m^C, required in order to reduce the initial residual for the scalar convection-diffusion problem with 5 orders of magnitude starting with a zero initial condition. We show the results for three different meshes, $K = 16$, $K = 64$, and $K = 256$, and for different values of N.

For the case $K = 16$ we also show the number of iterations, for the pure diffusion case ($Pe = 0$); we know from the results in Table 2 that these results are independent of K. We notice that the number of iterations decreases as K increases, and approaches the result for the pure diffusion case. This is due to the fact that, as K increases, the

coarse, global problem resolves the exact solution better, that is, the grid Peclet number decreases. These results are consistent with previous findings for nonsymmetric problems [53].

N	$K = 16$	$K = 64$	$K = 256$	$K = 16$ $(Pe = 0)$
3	24	15	11	10
4	22	17	13	12
5	25	21	15	14
6	30	24	17	16
7	34	27	19	18

Table 8: Number of iterations m^C $(d = 2,\ Pe = 100)$

4.4 The steady Navier-Stokes problem

We now illustrate the spatial convergence rate associated with the spectral element discretization of the steady two-dimensional Navier-Stokes equations. Kovasznay [29] gives an analytical solution to the Navier-Stokes equations which is similar to the two-dimensional flow field behind a periodic array of cylinders:

$$\begin{aligned} u_x &= 1 - e^{-\lambda x} cos(2\pi y) \\ u_y &= \frac{\lambda}{2\pi} e^{-\lambda x} sin(2\pi y) \\ \lambda &= \frac{1}{2} Re \pm \sqrt{\frac{1}{4} Re^2 + 4\pi^2}\,, \end{aligned}$$

where Re is the Reynolds number based on the mean flow velocity and separation between vortices. We solve this problem numerically in the case of $Re = 40$, $\lambda = \frac{1}{2} Re - \sqrt{\frac{1}{4} Re^2 + 4\pi^2}$, imposing the analytical velocity solution on the domain boundary.

We break up the computational domain $\Omega =]-0.5, 1.0] \times]-0.5, 1.5[$ into $K = 6$ equal quadrilateral spectral elements, each of order N. We then solve the discrete system of equations using the Newton-Krylov algorithm proposed in Section 3.4. The main reason for doing this test is to confirm that the algorithm computes the correct solution.

N	Q_2/P_1 (upwinding)	Q_3/Q_1
4	$6.84 \cdot 10^{-2}$	$6.84 \cdot 10^{-2}$
5	$1.25 \cdot 10^{-2}$	$1.25 \cdot 10^{-2}$
6	$2.09 \cdot 10^{-3}$	$2.09 \cdot 10^{-3}$
7	$3.10 \cdot 10^{-4}$	$3.10 \cdot 10^{-4}$
8	$4.08 \cdot 10^{-5}$	$4.08 \cdot 10^{-5}$
9	$4.73 \cdot 10^{-6}$	$4.73 \cdot 10^{-6}$
10	$5.01 \cdot 10^{-7}$	$5.01 \cdot 10^{-7}$

Table 9: Discretization error $\| \mathbf{u} - \mathbf{u}_\delta \| / \| \mathbf{u} \|$

In Table 9 we show the (relative) velocity error in the discrete semi-norm as a function of the polynomial order N. The results clearly demonstrate that exponential convergence is achieved, both in the case of using a coarse grid based upon Q_2/P_1 finite elements with streamline upwinding, as well as Q_3/Q_1 spectral elements without any upwinding. For a fixed N, the error in both these cases is the same.

5 Conclusions and final comments

We have presented iterative substructuring algorithms for the Poisson problem, the steady convection-diffusion problem, the steady Stokes problem, and the steady Navier-Stokes problem in the context of using spectral element discretizations and an additive Schwarz method without overlap. The preconditioners for these problems have three main components: (i) the solution of a coarse, global problem; (ii) the solution of independent, local problems associated with the individual spectral elements (or subdomains); (iii) the solution of a system for the unknowns on the element (subdomain) interfaces.

Associated with the three components of the Navier-Stokes preconditioner is a *hierarchy* of operators: (a) a steady, linearized Navier-Stokes operator (including streamline diffusion) for the coarse, global problem; (b) a steady Stokes operator for each individual, local problem; (c) an elliptic (Poisson type) operator for the interface problem.

Earlier studies have demonstrated the importance of including a coarse, global problem in order to obtain rapid convergence for elliptic problems. For the Stokes and Navier-Stokes algorithm presented here, the coarse, global problem is not only important for the convergence rate; it is *essential* in order to compute the correct solution.

The steady Stokes operator used for each individual, local problem provides an example of using *mixed* discrete operators. Here, the viscous term is treated using a linear, triangular or tetrahedral elements on the tensor-product Gauss-Lobatto nodes, while the divergence and gradient operator represent the original spectral element operators.

Numerical experiments indicate that the convergence rate is independent of the number of spectral elements (subdomains), and also independent of the domain aspect ratio. The number of iterations grows approximately linearly with the polynomial order inside the elements as well as the element (subdomain) aspect ratio.

We would like to mention that the solution algorithms presented in this paper have also been extended to: (i) spectral elements of different polynomial order (including

nonconforming matching conditions [7]); (ii) problems with variable properties (non-Newtonian flows); and (iii) problems using the full stress formulation (including the specification of Neumann boundary conditions). However, due to space limitation, these results will be reported in a separate paper together with illustrative examples [45].

Future work will focus on improving the preconditioning of the interface system; this part seems to be the weakest part of the proposed algorithms, in particular, for meshes with large *subdomain* aspect ratios. It would also be interesting to try other types of Schwarz algorithms, such as the the additive or multiplicative Schwarz algorithms including overlap [15, 22]. In terms of new application areas we plan to extend the current algorithms to solve unsteady problems, thus allowing for fully implicit time stepping procedures.

In order to better understand the proposed solution methods, as well as to suggest further improvements, we hope that the algorithms and the numerical results that we have presented in this paper will be followed up with a theoretical analysis.

6 Acknowledgments

The author would like to thank NASA Goddard Space Flight Center for the financial support of most of this work (contract NAS5-38075). The author would also like to thank Prof. Olof B. Widlund for his comments on an earlier version of the manuscript.

References

[1] M. Azaiez and G. Coppoletta, *Calcul de la pression dans le probleme de Stokes par une methode spectrale de quasi collocation a grille unique*, Annales Maghrebine de l'ingenieur (1992).

[2] I. Babuška, *The finite element method with Lagrangian multipliers*, Numer. Math., **20**, pp. 179-192 (1973).

[3] I. Babuška and M.R. Dorr, *Error estimates for combined h and p versions of the finite element method*, Numer. Math, **37**, pp. 257-277 (1981).

[4] I. Babuška and M. Suri, *The p- and h − p version of the finite element method, an overview*, Comp. Meth. Appl. Mech. Engrg., **80**, pp. 5-26 (1990).

[5] I. Babuška, B.A. Szabo and I.N. Katz, *The p-version of the finite element method*, SIAM J. Numer. Anal., **18**, pp. 515-545 (1981).

[6] C. Bernardi, Y. Maday and B. Metivet, *Spectral approximation of the periodic-nonperiodic Navier-Stokes equations*, Numer. Math., **51**, pp. 655-700 (1987).

[7] C. Bernardi, Y. Maday and A.T. Patera, *A new nonconforming approach to domain decomposition; the mortar element method*, Publications du Laboratoire D'Analyse Numerique, Universite Pierre et Marie Curie, No. R 89027 (1990).

[8] P.E. Bjørstad, and O.B. Widlund, *Iterative methods for the solution of elliptic problems on regions partitioned into substructures* SIAM J. Numer. Anal., **23**, pp. 1097-1120 (1986).

[9] J.H. Bramble and J.E. Pasciak, *A domain decomposition technique for Stokes problems*, Appl. Numer. Math., **6**, pp. 251-261 (1989/90).

[10] J.H. Bramble, J.E. Pasciak, and A.H. Schatz, *The construction of preconditioners for elliptic problems by substructuring, I.*, Math. Comput., **47**(175), pp. 103-134 (1986).

[11] J.H. Bramble, J.E. Pasciak, and A.H. Schatz, *The construction of preconditioners for elliptic problems by substructuring, IV.*, Math. Comput., **53**, pp. 1-24 (1989).

[12] F. Brezzi, *On the existence, uniqueness and approximation of saddle-point problems arising from Lagrange multipliers*, R.A.I.R.O. Anal. Numer., **8**, R2, pp. 129-151 (1974).

[13] A.N. Brooks and T.J.R. Hughes, *Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations*, Comput. Meth. Appl. Mech. Engng., **32**, pp. 199-259 (1982).

[14] X.C. Cai, W.D. Gropp, and D.E. Keyes, *A comparison of some domain decomposition algorithms and ILU preconditioned iterative methods for nonsymmetric elliptic problems*, Numer. Lin. Alg. Appl., **1**(5) (1994).

[15] X.C. Cai and O.B. Widlund, *Multiplicative Schwarz algorithms for some nonsymmetric and indefinite problems*, SIAM J. Numer. Anal., **30**(4), pp. 936-952 (1993).

[16] T.F. Chan and B.F. Smith, *Domain decomposition and multigrid algorithms for elliptic problems on unstructured meshes*, in Proc. Seventh Int. Conf. on Domain Decomposition Methods for PDE's, SIAM (to appear).

[17] P. Demaret and M.O. Deville, *Chebyshev pseudospectral solution of the Stokes equations using finite element preconditioning*, J. Comput. Phys., **83**(2), pp. 463-484 (1989).

[18] L. Demkowicz, J.T. Oden, W. Rachowicz, and O. Hardy *Toward a universal $h-p$ adaptive finite element strategy, Part 1. Constrained approximation and data structure*, Comp. Meth. Appl. Mech. Engrg., **77**, pp. 79-112 (1989).

[19] M.O. Deville and E.H. Mund, *Finite-element preconditioning for pseudospectral solutions of ellipic problems*, SIAM J. Sci. Stat. Comput., **11**(2), pp. 311-342 (1990).

[20] M. Dryja, B.F. Smith and O.B. Widlund, *Schwarz analysis of iterative substructuring algorithms for elliptic problems in three dimensions*, SIAM J. Numer. Anal., **31**(6), pp. 1662-1694 (1994).

[21] M. Dryja and O.B. Widlund, *Towards a unified theory of domain decomposition algorithms for elliptic problems*, in T.F. Chan, R. Glowinski, J. Periaux and O.B. Widlund, editors, Third Int. Symp. on Domain Decomposition Methods for PDE's, SIAM, pp. 3-21 (1990).

[22] M. Dryja and O.B. Widlund, *Domain decomposition algorithms with small overlap*, SIAM J. Sci. Comput., **15**(3), pp. 604-620 (1994).

[23] P.F. Fischer and A.T. Patera, *Parallel spectral element solution to the Stokes problem*, J. Comput. Phys., **92**(2), pp. 380-421 (1991).

[24] P.F. Fischer and E.M. Rønquist, *Spectral element methods for large scale parallel Navier-Stokes calculations*, Comp. Meth. Appl. Mech. Engrg., **116**, pp. 69-76 (1994).

[25] M. Fortin and A. Fortin, *Experiments with several elements for viscous incompressible flows*, Int. J. Numer. Meth. Fluids, **5**, pp. 911-928 (1985).

[26] M. Fortin and R. Glowinski, Augmented Lagrangian Methods, North-Holland (1983).

[27] R. Glowinski, Numerical Methods for Nonlinear Variational Problems, Springer Verlag (1984).

[28] D.W. Kelly, S. Nakazawa, O.C. Zienkiewicz and J.C. Heinrich, *A note on upwinding and balancing dissipation in finite element approximations to convection diffusion problems*, Int. J. Numer. Meth. Engng., **15**, pp. 1705-1711 (1980).

[29] L.I.G Kovasznay, *Laminar flow behind a two-dimensional grid*, Proceedings of the Cambridge Philosophical Society, **44**, pp. 58-62 (1948).

[30] P. Le Tallec, *Domain decomposition methods in computational mechanics*, Computational Mechanics Advances, **1**(2) (1994).

[31] Y. Maday, D. Meiron, A.T. Patera and E.M. Rønquist, *Analysis of iterative methods for the steady and unsteady Stokes problem: Application to spectral element discretizations*, SIAM J. Sci. Stat. Comput., **14**(2), pp. 310-337 (1993).

[32] Y. Maday and A.T. Patera, *Spectral element methods for the Navier-Stokes equations*, in State of the Art Surveys in Computational Mechanics, edited by A. K. Noor, ASME, New York, pp. 71-143 (1989).

[33] Y. Maday, A.T. Patera and E.M. Rønquist, *An operator-integration-factor splitting method for time-dependent problems: application to incompressible fluid flow*, J. Sci. Comput., **5**(4), pp. 263-292 (1991).

[34] Y. Maday, A. T. Patera and E. M. Rønquist, *The $P_N \times P_{N-2}$ method for the approximation of the Stokes problem*, Numer. Math., to appear.

[35] J. Mandel, *Iterative solvers by substructuring for the p-version finite element method*, Comput. Meth. Appl. Mech. Engrg., **80**, pp. 117-128 (1990).

[36] J.T. Oden, *Theory and implementation of high-order adaptive $h-p$ methods for the analysis of incompressible viscous flows*, in S.N. Atluri, editor, Computational Nonlinear Mechanics in Aerospace Engineering, AIAA Progress in Aeronautics and Astronautics' Series, **146**, pp. 321-363 (1992).

[37] J.T. Oden and L. Demkowicz, *$h-p$ adaptive finite element methods in computational fluid dynamics*, Comp. Meth. Appl. Mech. Engrg., **89**, pp. 11-40 (1991).

[38] S.A. Orszag, *Spectral methods for problems in complex geometries*, J. Comput. Phys., **37**, pp. 70-92 (1980).

[39] A.T. Patera, *A spectral element method for fluid dynamics; Laminar flow in a channel expansion*, J. Comput. Phys., **54**, pp. 468-488 (1984).

[40] L.F. Pavarino and O.B. Widlund, *A polylogarithmic bound for an iterative substructuring method for spectral elements in three dimensions*, Technical Report 661, Dept. of Computer Science, Courant Institute, March 1994. To appear in SIAM J. Numer. Anal.

[41] L.F. Pavarino and O.B. Widlund, *Iterative substructuring methods for spectral elements: problems in three dimensions based on numerical quadrature*, Technical Report 663, Dept. of Computer Science, Courant Institute, May 1994. To appear in Int. J. Comp. Math. Appl.

[42] M.P. Reddy and L.G. Reifschneider, *Accuracy and convergence of element-by-element iterative solvers for incompressible fluid flows using penalty finite element model*, Int. J. Numer. Meth. Fluids, **17**, pp. 1019-1033 (1993).

[43] E.M. Rønquist, Optimal Spectral Element Methods for the Unsteady Three-dimensional Incompressible Navier-Stokes Equations, Ph.D. Thesis, Massachusetts Institute of Technology, 1988.

[44] E.M. Rønquist, *Convection treatment using spectral elements of different order*, Int. J. Numer. Meth. Fluids, to appear.

[45] E.M. Rønquist, *A domain decomposition solver for the incompressible Navier-Stokes equations*, in preparation.

[46] Y. Saad and M.H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., **7**(3), pp. 856-869 (1986).

[47] B.F. Smith, *A parallel implementation of an iterative substructuring algorithm for problems in three dimensions*, SIAM J. Sci. Comput., **14**(2), pp. 406-423 (1993).

[48] G. Strang and G. Fix, An Analysis of the Finite Element Method, Prentice-Hall (1973).

[49] A.H. Stroud and D. Secrest, Gaussian Quadrature Formulas, Prentice Hall (1966).

[50] M. Behr and T.E. Tezduyar, *Finite element solution strategies for large-scale flow simulations*, Comput. Meth. Appl. Mech. Engrg., **112**, pp. 3-24 (1994).

[51] J. Liou and T.E. Tezduyar, *Clustered element-by-element computations for fluid flow*, in Horst D. Simon (editor), Parallel Computational Fluid Dynamics - Implementation and Results, Scientific and Engineering Computation Series, MIT Press, pp. 167-187 (1992).

[52] O.B. Widlund, *Iterative substructuring methods: algorithms and theory for elliptic problems in the plane*, in First Int. Symp. on Domain Decomposition Methods for PDE's (edited by R. Glowinski et. al.), SIAM, Philadelphia (1988).

[53] J. Xu and X.C. Cai, *A preconditioned GMRES method for nonsymmetric or indefinite problems*, Math. Comput., **59**(200), pp. 311-319 (1992).