BDDC Preconditioning for High-Order Galerkin Least Squares Methods using Inexact Solvers

Masayuki Yano*,[a], David L. Darmofal[a]

[a] *Aerospace Computational Design Laboratory, Massachusetts Institute of Technology, Building 37, Room 442, Cambridge, MA 02139, USA*

Abstract

A high-order Galerkin Least-Squares (GLS) finite element discretization is combined with a Balancing Domain Decomposition by Constraints (BDDC) preconditioner and inexact local solvers to provide an efficient solution technique for large-scale, convection-dominated problems. The algorithm is applied to the linear system arising from the discretization of the two-dimensional advection-diffusion equation and Euler equations for compressible, inviscid flow. A Robin-Robin interface condition is extended to the Euler equations using entropy-symmetrized variables. The BDDC method maintains scalability for the high-order discretization of the diffusion-dominated flows, and achieves low iteration count in the advection-dominated regime. The BDDC method based on inexact local solvers with incomplete factorization and $p = 1$ coarse correction maintains the performance of the exact counterpart for the wide range of the Peclet numbers considered while at significantly reduced memory and computational costs.

Key words: Galerkin least-squares, high-order methods, domain decomposition methods, BDDC, preconditioners

1. Introduction

Stabilized finite element methods, which enable high-order accurate discretization of advection-dominated flows on unstructured meshes, have been extensively studied for high-fidelity CFD simulations on geometrically complex domains. [1, 2, 3, 4, 5]. However, as the high-order discretization captures a wide range of

*Corresponding author
Email addresses: myano@mit.edu (Masayuki Yano), darmofal@mit.edu (David L. Darmofal)

spatial scales, the method imposes a stringent restriction on the time step for explicit time marching, making an explicit method impractical for the calculation of steady state problems. While implicit time integration removes the time step restriction, the condition number of the discrete system worsens rapidly with the interpolation order and mesh resolution. The solution strategy for the linear system is further complicated by the increasing parallelism of the modern computers. Thus, designing an efficient, parallel preconditioner for the high-order discretization is a crucial issue.

This work employs the Balancing Domain Decomposition by Constraints (BDDC) algorithm [6] to solve the linear systems arising from the discretization of the advection-dominated problems via the Galerkin Least-Squares method [7]. The BDDC method constructs a coarse, global space from a set of selected constraints, and achieves a condition number independent of the number of subdomains when applied to the Poisson equation, having the same spectrum as the FETI-DP methods [8, 9, 10]. The condition number of the system only weakly depends on the interpolation order, and its effectiveness for high-order discretizations has recently been verified by Klawonn et. al. for spectral methods using the Gauss-Lobatto-Legendre quadrature nodes [11]. This work employs high-order simplex elements and the numerical results confirm that the BDDC remains an attractive choice for the discretization.

To apply the domain decomposition algorithms to the advection-diffusion equation, Achdou et. al. introduced the Robin-Robin interface condition, which modifies the local bilinear form to ensure the positiveness of the local problem [12]. The idea has been applied successfully to the FETI [13] and BDDC algorithms [14]. This work generalizes the Robin-Robin framework to systems of conservation laws considering the local energy stability and applies the modification to the Euler equations.

This work also presents the BDDC method based on approximate local solvers. The use of inexact solvers for the BDDC method has been considered recently in [15, 16]. In these works, the subdomain problems or the partially assembled system arising from the linear finite element discretization is solved using an incomplete factorization or h-multigrid. The requirement for the inexact local solver for the current work differs from the previous work in two regards. First, the approximate solver must remain effective for the advection-dominated problems with highly anisotropic features. Second, the solver must work well for the high-order discretization. In order to address these issues, this work employs a two-level multiplicative preconditioner

as a local solver. The first component is the dual-threshold incomplete factorization (ILUT) [17] with the minimum discarded fill (MDF) reordering [18, 19], which orders the unknowns appropriately such that the incomplete factorization captures the long-range propagation of the perturbations along the characteristics. The second component is the $p = 1$ correction for the elliptic mode, which is a natural choice for the construction of the coarse space for the high-order discretization [19, 20].

This paper is organized as follows. Section 2 describes the high-order GLS discretization. Section 3 introduces the BDDC preconditioner, with the generalization of the Robin-Robin interface condition to systems of equations. Section 4 presents the BDDC algorithm based on the inexact local solvers designed to maintain the scalability of the exact counterpart at a reduced cost. Section 5 presents the result of applying the BDDC algorithm to the advection-diffusion equation with a wide range of Peclet numbers and the Euler equations. Finally, Section 6 discusses conclusions and ongoing work.

2. The Galerkin Least-Squares Method

2.1. Variational Form of Conservation Laws

Let $\Omega \in \mathbb{R}^d$ be an open, bounded domain, where d is the number of spatial dimensions. In general, a system of time-dependent conservation laws is expressed in semilinear form as

$$\tilde{u}_{k,t} + (F^{\text{inv}}_{ik}(\tilde{u}, x, t))_{,x_i} - (F^{\text{vis}}_{ik}(\tilde{u}, \nabla\tilde{u}, x, t))_{,x_i} = f_k(x, t), \quad \text{in } \Omega$$

where $k \in \{1, \ldots, m\}$ is the component index of the governing equations, $i \in \{1, \ldots, d\}$ is the spatial index, $(\cdot)_{,t}$ denote the temporal derivative, and $(\cdot)_{,x_i}$ denote the spatial derivatives with respect to x_i. The conservative variable $\tilde{u}$, inviscid flux $F^{\text{inv}}(\tilde{u}, x, t)$, viscous flux $F^{\text{vis}}(\tilde{u}, \nabla\tilde{u}, x, t)$, and the source term $f(x, t)$, characterize the governing equations to be solved. For conservation laws possessing an entropy function[21, 22], a set of entropy variables, u, can be defined such that

$$A_{0kl} = \frac{\partial \tilde{u}_k}{\partial u_l}, \quad A_{ikl} = \frac{\partial F^{\text{inv}}_{ik}}{\partial u_l}, \quad \text{and} \quad K_{ijkl} u_{l,x_j} = F^{\text{vis}}_{ik}$$

are all symmetric with respect to k and l, and $A_{0\cdot\cdot}$ is positive definite.

Let a linear operator $\mathcal{L}_{k,w}\cdot$, parametrized by w, be defined as

$$\mathcal{L}_{k,w}u \equiv A_{0kl}|_w u_{l,t} + A_{ikl}|_w u_{l,x_i} - (K_{ijkl}|_w u_{l,x_j})_{,x_i}, \tag{1}$$

Using the operator $\mathcal{L}_{k,w}$, the governing equations in entropy variables can be expressed as,

$$\mathcal{L}_{k,u}u \equiv A_{0kl}u_{l,t} + A_{ikl}u_{l,x_i} - (K_{ijkl}u_{l,x_j})_{,x_i} = f_k.$$

This form of the equations is convenient for analyzing the energy stability of the variational form.

The finite element discretization of the problem is performed on a space of functions

$$V_h = \{v \in [H^1(\Omega)]^m : v|_K \in [\mathcal{P}_p(K)]^m, \forall K \in \mathcal{T}_h\}$$

where $\mathcal{T}_h$ is the triangulation of domain Ω into non-overlapping elements, K, such that $\bar{\Omega} = \cup_{K\in\mathcal{T}_h}\bar{K}$, and $\mathcal{P}_p(K)$ is the space of p-th order polynomial on K. The superscript m implies the spaces are vector-valued. The finite element variational problem consists of finding $u \in V_h$ such that

$$(A_{0kl}u_{l,t}, v_k)_\Omega + \mathcal{R}_{gal}(u,v) + \mathcal{R}_{ls}(u,v) = 0 \quad \forall v \in V_h, \tag{2}$$

where

$$\begin{aligned}
\mathcal{R}_{gal}(u,v) &= -(F^{\text{inv}}_{ik}, v_{k,x_i})_\Omega + (F^{\text{vis}}_{ik}, v_{k,x_i})_\Omega - (f_k, v_k)_\Omega + (\hat{F}_k(u, \text{B.C.data}, n), v_k)_{\partial\Omega}, \\
\mathcal{R}_{ls}(u,v) &= (\mathcal{L}_{l,u}v, \tau_{lk}(\mathcal{L}_{k,u}u - f_k))_{\Omega,\mathcal{T}_h},
\end{aligned}$$

where $(\cdot,\cdot)_\Omega : L^2(\Omega)\times L^2(\Omega) \to \mathbb{R}$ and $(\cdot,\cdot)_{\partial\Omega} : L^2(\partial\Omega)\times L^2(\partial\Omega) \to \mathbb{R}$ denote the L^2 inner product over the domain and the boundary of the domain, respectively. The numerical flux function, $\hat{F}$, uses the interior state and the boundary condition to define the appropriate flux at the boundary. τ is the stabilization parameter. $(\cdot,\cdot)_{\Omega,\mathcal{T}_h}$ denotes the summation of the element-wise L^2 inner product, $\sum_K (\cdot,\cdot)_K$.

For the advection-diffusion equation, $\tilde{u} = u$ and the inviscid and viscous fluxes are given by

$$F_i^{\text{inv}} = \beta_i u \quad \text{and} \quad F_i^{\text{vis}} = \kappa u_{,x_i},$$

where β is the advection field and κ is the diffusivity coefficient.

The Euler equations are formulated using Hughes' entropy-symmetrized variables[22] with the scaling proposed by Barth[23]. The entropy-symmetrized state and the flux functions are given by

$$u = \begin{pmatrix} \frac{-s+\gamma+1}{\gamma-1} - \frac{\rho E}{p} \\ \frac{\rho v_1}{p} \\ \frac{\rho v_2}{p} \\ -\frac{\rho}{p} \end{pmatrix} \quad \text{and} \quad F_i^{\text{inv}} = \begin{pmatrix} \rho v_i \\ \rho v_1 v_i + p\delta_{i1} \\ \rho v_2 v_i + p\delta_{i2} \\ \rho H v_i \end{pmatrix},$$

where ρ is the fluid density, v is the velocity vector, p is the pressure, and E is the specific stagnation internal energy. The entropy, s, is given by $s = \log(p/\rho^\gamma)$, the specific stagnation enthalpy, H, is given by $H = E + p/\rho$, and the pressure is given by $p = (\gamma - 1)\rho(E - \|v\|^2/2)$, where γ is the ratio of specific heats.

2.2. Stabilization Parameter τ

The stabilization parameter controls the amount of stabilization added to the Galerkin Least-Squares discretization. From the variational multiscale perspective, it can be thought of as adding the dissipation that would have been provided by the unresolved, subgrid scale [24, 25]. It is well known [26, 2, 27] that, in order to attain optimal convergence rate for the advection-diffusion equation with the element size h, the stabilization parameter must scale as

$$\tau = \mathcal{O}(h/|\beta|), \quad Pe \gg 1$$
$$\tau = \mathcal{O}(h^2/\kappa), \quad Pe \ll 1.$$

The first condition is necessary to obtain stability and optimal convergence in the streamline derivative in advection-dominated cases, and the second condition is necessary to maintain the optimality in diffusion-

dominated case. As the conservation laws of interest consist of inviscid and viscous parts, τ may be conveniently expressed as the sum of these two parts. In particular, the scaling relation can be satisfied by choosing

$$\tau^{-1} = \tau_{\text{inv}}^{-1} + \tau_{\text{vis}}^{-1},$$

where $\tau_{\text{inv}} = \mathcal{O}(h/|\beta|)$ and $\tau_{\text{vis}} = \mathcal{O}(h^2/\kappa)$. In addition to satisfying the scaling relations, the τ matrix must be symmetric and positive-definite for a system of equations [3].

Generalization of the inviscid stabilization parameter to a multidimensional system of equations follows from [23]. Let the directional vectors associated with the mapping from physical space, described by coordinate x, to the reference space, described by barycentric coordinate ξ, be

$$n^i = (\xi_{i,x_1}\xi_{i,x_2})^T, \quad i = 1, 2, 3,$$

and the unit normal vector be $\hat{n}^i = n^i/|n^i|$. For example, in two-dimensions, the normal vectors are

$$\begin{aligned} n^1 &= (\xi_{1,x_1}, \xi_{1,x_2})^T \\ n^2 &= (\xi_{2,x_1}, \xi_{2,x_2})^T \\ n^3 &= -(\xi_{1,x_1} + \xi_{2,x_1}, \xi_{1,x_2} + \xi_{2,x_2})^T. \end{aligned}$$

The directional flux Jacobian with respect to the conservative variable, $\tilde{u}$, is given by

$$\tilde{A}(n^i) = n_j^i \tilde{A}_j,$$

where $\tilde{A}_j$ is the conservative flux Jacobian in the x_j coordinate direction, i.e. $\tilde{A}_j \equiv \partial F_j^{\text{inv}}/\partial\tilde{u}$. The inviscid stabilization parameter is defined as

$$\tau_{\text{inv}}^{-1} = |n^i||\tilde{A}(\hat{n}^i)|A_0,$$

where $|\tilde{A}(\hat{n}^i)|$ is the matrix absolute value of the conservative flux Jacobian evaluated in the $\hat{n}^i$ direction. Note, the directional flux Jacobian, $\tilde{A}(\hat{n})$, has a real set of eigenvalues for any $\hat{n}$, as the inviscid problem constitutes a hyperbolic system.

The viscous stabilization parameter is chosen to be

$$\tau_{\text{vis}}^{-1} = \frac{p^2}{h_s^2} K_{ii},$$

where h_s is the shortest edge of an element, p is the interpolation order, and K_{ii} is the sum of the block diagonal entries of the viscosity tensor. The choice of the shortest edge, h_s, as the viscous scaling length follows from the analysis in [28, 29, 30]. The p-dependent reduction of the stabilization parameter is motivated by recalling that the role of τ is to recover the dissipation that would have been provided by the subgrid, unresolved features. As pointed out by Hughes [25], p-refinement captures forms of the dissipation effects present at subgrid scales. Thus, the proposed approach rescales the viscous scaling length based on the interpolation order.

2.3. Discrete Systems

Once a suitable basis for V_h is chosen, a solution $u \in V_h$ can be expressed as $u = U_j \phi_j$, where $\{\phi_j\}_{j=1}^{\text{dof}}$ is the set of basis functions. Then, Eq. (2) can be expressed as a system of ODE's

$$M \frac{dU}{dt} + R(U) = 0,$$

where $R(U)_i = \mathcal{R}_{gal}(u, \phi_i) + \mathcal{R}_{ls}(u, \phi_i)$ are the discrete nonlinear residuals, and $M = (A_0 \phi_j, \phi_i)_\Omega$ is the mass matrix. The steady state solution to the conservation laws is given at $R(U) = 0$. In order to solve for the steady state, a damped Newton method based on the backward Euler differencing of the ODE is used, i.e.

$$U^{n+1} = U^n + \left(\frac{1}{\Delta t} M + \left. \frac{dR}{dU} \right|_{U=U^n} \right)^{-1} R(U^n).$$

Even though this work is concerned with steady problems, the pseudo-time stepping improves the robustness of the solver for nonlinear equations. After the initial transient, $\Delta t \to \infty$, and the method approaches the

full Newton method. The pseudo-time marching scheme requires the solution of a linear equation at each time step, i.e.

$$\left(\frac{1}{\Delta t}M + \left.\frac{dR}{dU}\right|_{U=U^n}\right)\Delta U^n = -R(U^n).$$

For a small Δt, the linear equation is well-conditioned as the mass matrix dominates. On the other hand, as $\Delta t \to \infty$, the linear system becomes harder to solve as the condition number of the system increases. Note, the matrix $\frac{1}{\Delta t}M + \frac{dR}{dU}$ plays the role of the stiffness matrix for the Newton step. Thus, the stiffness matrix, denoted by A, instead of the Jacobian $\frac{1}{\Delta t}M + \frac{dR}{dU}$, is used to describe the linear solver algorithm in the following sections to be consistent with other literature on domain decomposition methods.

2.4. High-Order C^0 Basis Functions

Throughout this study, triangular elements with a Lagrange basis are used to construct the approximation space. A set of equally-spaced nodes is used to define the basis functions on a reference element.

High-order geometry representation is achieved through polynomial mapping based on the Lagrange points, i.e. $x = x_j\phi_j(\xi)$ where ϕ_j is the Lagrange basis function corresponding to node j, x_j is the physical coordinate of the j-th Lagrange node, and ξ is the coordinate on the reference triangle. Note, as the mapping $T : \xi \to x$ is nonlinear, the basis functions in the physical space are not polynomials for curved elements.

The basis functions for the high-order continuous Galerkin method must be C^0 continuous across the element boundaries. In order to enforce the continuity, it is convenient to categorize the basis functions into one of the three types of modes in two dimensions, as shown in Figure 1. The first is the node mode, which has support on all elements sharing the node. For a given node, this mode always has a single degree of freedom regardless of the polynomial order. This is the only active mode for $p = 1$ discretization. The second type of basis is the edge mode, which has support on two elements sharing an edge. The degrees of freedom associated with a given edge are equal to $p - 1$. The third type of basis is the element mode, which has support on a single element. The degrees of freedom associated with a given element are equal to $\frac{1}{2}(p-2)(p-1)$. Except on the boundary, the node mode has the largest support out of all basis types.

3. Balancing Domain Decomposition by Constraints

3.1. The BDDC Preconditioner

Let $\{\Omega_i\}_{i=1}^N$ be a decomposition of domain Ω into N non-overlapping subdomains such that $\Omega_i \cap \Omega_j = \emptyset, i \neq j$ and $\bar{\Omega} = \cup_{i=1}^N \bar{\Omega}_i$. The decomposition is assumed to align with the finite element triangulation, $\mathcal{T}_h$. The interface of a subdomain Ω_i is denoted by Γ_i and defined as $\Gamma_i \equiv \partial\Omega_i \setminus \partial\Omega$. The collection of the subdomain interfaces is denoted by Γ, and is defined as $\Gamma \equiv \cup_{i=1}^N \Gamma_i$. The space of finite element functions on Ω_i is denoted by $V_h^{(i)}$. The subset of these functions that vanish on Γ_i is denoted by $V_I^{(i)}$, and the subset that have nonzero traces on Γ_i is denoted by $V_\Gamma^{(i)}$. In order to define the local Schur complement system, the local degrees of freedom are decomposed into the interior part $u_I^{(i)} \in V_I^{(i)}$ and the interface part $u_\Gamma^{(i)} \in V_\Gamma^{(i)}$. The local stiffness matrix, the solution vector, and the load vector are denoted by

$$A^{(i)} = \begin{pmatrix} A_{II}^{(i)} & A_{I\Gamma}^{(i)} \\ A_{\Gamma I}^{(i)} & A_{\Gamma\Gamma}^{(i)} \end{pmatrix}, \quad u^{(i)} = \begin{pmatrix} u_I^{(i)} \\ u_\Gamma^{(i)} \end{pmatrix}, \quad \text{and} \quad f^{(i)} = \begin{pmatrix} f_I^{(i)} \\ f_\Gamma^{(i)} \end{pmatrix},$$

where the subscript I and Γ denote the interior and interface degrees of freedom, respectively.

In order to relate the local problem on $V_h^{(i)}$ to the global problem on V_h, let $R^{(i)} : V_h \to V_h^{(i)}$ be the restriction operator that extracts the degrees of freedom associated with $V_h^{(i)}$ from V_h. The global stiffness matrix and the load vector are obtained by assembling the interface degrees of freedom, i.e.

$$A = \sum_{i=1}^N (R^{(i)})^T A^{(i)} R^{(i)} \quad \text{and} \quad f = \sum_{i=1}^N (R^{(i)})^T f^{(i)}.$$

The global system $Au = f$ is solved using the Generalized Minimum Residual (GMRES) algorithm [31], with the BDDC preconditioner.

In order to define the BDDC algorithm, the space of local interface degrees of freedom, $V_\Gamma^{(i)}$, is decomposed into two parts, i.e. $V_\Gamma^{(i)} = V_\Delta^{(i)} \oplus V_\Pi^{(i)}$. The local primal variables, $V_\Pi^{(i)}$, are a selected few degrees of freedom on the interface, which constitute the global coarse space for the BDDC preconditioner. The local dual variables, $V_\Delta^{(i)}$, consists of functions in $V_\Gamma^{(i)}$ that vanish on the primal degrees of freedom. For convenience, let the product of the interior and the dual space be denoted by $V_r^{(i)} \equiv V_I^{(i)} \oplus V_\Delta^{(i)}$. Then, the local stiffness

matrix, the solution vector, and the load vector can be partitioned as

$$A^{(i)} = \begin{pmatrix} A_{rr}^{(i)} & A_{r\Pi}^{(i)} \\ A_{\Pi r}^{(i)} & A_{\Pi\Pi}^{(i)} \end{pmatrix}, \quad u^{(i)} = \begin{pmatrix} u_r^{(i)} \\ u_\Pi^{(i)} \end{pmatrix}, \quad \text{and} \quad f^{(i)} = \begin{pmatrix} f_r^{(i)} \\ f_\Pi^{(i)} \end{pmatrix},$$

where

$$A_{rr}^{(i)} = \begin{pmatrix} A_{II}^{(i)} & A_{I\Delta}^{(i)} \\ A_{\Delta I}^{(i)} & A_{\Delta\Delta}^{(i)} \end{pmatrix}, \quad A_{r\Pi}^{(i)} = \begin{pmatrix} A_{I\Pi}^{(i)} \\ A_{I\Delta}^{(i)} \end{pmatrix}, \quad \text{and} \quad A_{\Pi r}^{(i)} = \begin{pmatrix} A_{\Pi I}^{(i)} & A_{\Pi\Delta}^{(i)} \end{pmatrix}.$$

The space $V_\Pi^{(i)}$ is chosen such that the problem defined on $V_r^{(i)}$ has a unique solution, i.e. $A_{rr}^{(i)}$ is non-singular. For example, for the Poisson equation, it is sufficient to choose the degrees of freedom on the corners of the subdomain as primal variables, which prevents the subdomains from floating. Two coarse primal variable spaces are considered in this study: the first is the space spanned by coarse basis functions defined on the subdomain corners, and the second is the space spanned by coarse basis functions on the subdomain corners and edges (i.e. the average value along an subdomain edge). When the edge average constraints are employed, the basis functions of $V_h^{(i)}$ are modified such that all primal and dual degrees of freedom are explicit, using the method described in [32, 9]. For either set of the primal variables, the degrees of freedom of $V_\Pi^{(i)}$ is on the order of the number of subdomains connected to Ω_i. While the use of the corner only constraints is sufficient to produce a preconditioned operator with the condition number independent of the number of subdomains, the addition of the edge average constraints further improves the conditioning of the operator.

The problem on $A_{rr}^{(i)}$ is refered to as the constrained Neumann problem from here on, as, for the Poisson equation, the inversion of $A_{rr}^{(i)}$ corresponds to solving the problem with the Neumann boundary condition almost everywhere on Γ, except at locations where the boundary values are required to vanish due to primal constraints.

The key space used in the BDDC algorithm is the partially assembled space, $\tilde{V}$, defined by

$$\tilde{V} = \left(\oplus_{i=1}^{N} V_r^{(i)} \right) \oplus V_\Pi,$$

where V_Π is the primal variable space, which is continuous across the interface. By construction, the degrees of freedom of V_Π is on the order of the number of subdomains. Let $\bar{R}^{(i)} : \tilde{V} \to V_h^{(i)}$ be the restriction operator that extracts the degrees of freedom associated with $V_h^{(i)}$ from $\tilde{V}$. The stiffness matrix on $\tilde{V}$ is given by

$$\tilde{A} = \sum_{i=1}^{N} (\bar{R}^{(i)})^T A^{(i)} \bar{R}^{(i)} = \begin{pmatrix} A_{rr}^{(1)} & & & \tilde{A}_{r\Pi}^{(1)} \\ & \ddots & & \vdots \\ & & A_{rr}^{(N)} & \tilde{A}_{r\Pi}^{(N)} \\ \tilde{A}_{\Pi r}^{(1)} & \cdots & \tilde{A}_{\Pi r}^{(N)} & \tilde{A}_{\Pi\Pi} \end{pmatrix} \equiv \begin{pmatrix} A_{rr} & \tilde{A}_{r\Pi} \\ \tilde{A}_{\Pi r} & \tilde{A}_{\Pi\Pi} \end{pmatrix},$$

where

$$\tilde{A}_{r\Pi}^{(i)} \equiv \begin{pmatrix} A_{I\Pi}^{(i)} R_\Pi^{(i)} \\ A_{\Delta\Pi}^{(i)} R_\Pi^{(i)} \end{pmatrix}, \quad \tilde{A}_{\Pi r}^{(i)} \equiv \begin{pmatrix} (R_\Pi^{(i)})^T A_{\Pi I}^{(i)} & (R_\Pi^{(i)})^T A_{\Pi\Delta}^{(i)} \end{pmatrix},$$
$$\text{and} \quad \tilde{A}_{\Pi\Pi} = \sum_{i=1}^{N} (R_\Pi^{(i)})^T A_{\Pi\Pi}^{(i)} R_\Pi^{(i)}.$$

In other words, $\tilde{A}$ is created by assembling the local spaces with respect to only the primal variables.

In order to inject a function from the partially assembled space, $\tilde{V}_\Gamma$, into the fully assembled space, V_Γ, a scaling operator that takes a weighted average of the interface variables must be defined [6]. For this study, the scaling operator $\delta_i^\dagger$ is simply set to $1/\mathcal{N}_x$, where $\mathcal{N}_x$ is the number of subdomains sharing the same degree of freedom on the interface. For an elliptic problem with discontinuous coefficients, it is known that the scaling parameter should be weighted by the coefficient of neighboring subdomains [33]. The scaling factors, $\delta_i^\dagger$, are collected to form a diagonal matrix, $D^{(i)}$, which operates on $V_\Gamma^{(i)}$. The weighted interface restriction operator, $\tilde{R}_{D,\Gamma}^{(i)}$, is obtained by multiplying the rows of $R^{(i)}$ corresponding to the dual variables by $\delta_i^\dagger$ and applying an identity mapping to the primal variables.

The BDDC preconditioner is defined as

$$M_{\text{BDDC}}^{-1} = \tilde{\mathcal{H}}_1 \tilde{A}^{-1} \tilde{\mathcal{H}}_2,$$

where the extension operators are given by

$$\tilde{\mathcal{H}}_1 = \begin{pmatrix} I & A_{II}^{-1}\tilde{A}_{I\Gamma}J_{D,\Gamma} \\ & \tilde{R}_{D,\Gamma}^T \end{pmatrix} \qquad \text{and} \qquad \tilde{\mathcal{H}}_2 = \begin{pmatrix} I & \\ J_{D,\Gamma}\tilde{A}_{\Gamma I}A_{II}^{-1} & \tilde{R}_{D,\Gamma} \end{pmatrix},$$

where $A_{II} \equiv \text{diag}_{i=1}^N \left(A_{II}^{(i)}\right)$ and $J_{D,\Gamma}$ is the jump operator on the interface variables defined by $J_{D,\Gamma} = I_\Gamma - \tilde{R}_{D,\Gamma}R_\Gamma^T$. Recall, because the primal variables V_Π are continuous across the subdomain interfaces, the jump operator maps the primal variables to zero. Thus, the operator $A_{II}^{-1}\tilde{A}_{I\Gamma}J_{D,\Gamma}$ extends the jump in the dual variables to the interior using the discrete harmonic extension, which is the minimum energy extension for the symmetric positive definite systems [34].

3.2. Implementation of the BDDC Algorithm

A single step of the BDDC preconditioning requires the inversion of the partially assembled matrix, $\tilde{A}$, and the application of extension operators, $\tilde{\mathcal{H}}_1$ and $\tilde{\mathcal{H}}_2$. Most of the operations required for the application of these operators parallelize, making the algorithm suitable for massively parallel architectures.

With a Cholesky-like factorization, the inverse of the partially assembled matrix can be expressed as

$$\tilde{A}^{-1} = \begin{pmatrix} A_{rr}^{-1} & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} -A_{rr}^{-1}\tilde{A}_{r\Pi} \\ I \end{pmatrix} S_\Pi^{-1} \begin{pmatrix} -\tilde{A}_{\Pi r}A_{rr}^{-1} & I \end{pmatrix},$$

where $S_\Pi = A_{\Pi\Pi} - A_{\Pi r}A_{rr}^{-1}A_{r\Pi}$ is called the primal Schur complement. The application of A_{rr} corresponds to solving the constrained Neumann problem on each subdomain, which parallelizes completely as the continuity across the subdomain interface is not enforced in $\oplus_{i=1}^N V_r^{(i)}$. The inversion of the primal Schur complement is a globally coupled problem; however, the primal Schur complement problem is much smaller than the local problems, as the degrees of freedom associated with V_Π is approximately equal to the number of subdomains when only the corner constraints are employed and is approximately three times the number of subdomains when both the corner and edge average constrains are employed. Moreover, S_Π is sparse as only the primal degrees of freedom that share the same subdomain connect to each other. Thus, in this work, S_Π is factored exactly.

Similarly, application of the extension operators, $\tilde{\mathcal{H}}_1$ and $\tilde{\mathcal{H}}_2$, can be carried out almost completely in parallel; the communication is required only for calculating the action of the jump operator, $J_{D,\Gamma}$. The majority of the cost comes from the application of A_{II}^{-1}, which corresponds to solving the local Dirichlet problem on each subdomain.

At first glance, the application of $\tilde{A}^{-1}$ seems to require solving the constrained Neumann problem, A_{rr}, twice. However, when the factorization of the local stiffness matrix, $A^{(i)}$, is available, $\tilde{A}^{-1}$ can be applied using a forward substitution, followed by the inversion of S_Π, and finally a backward substitution. Thus, the cost of applying $\tilde{A}^{-1}$ is approximately equal to the cost of solving a single constrained Neumann problem. Similarly, when the factorization of the local problem is available, the extension operators $\tilde{\mathcal{H}}_1$ and $\tilde{\mathcal{H}}_2$ can be applied by single backward solve and forward solve, respectively. Thus, the total cost of a single step of the BDDC algorithm is approximately equal to the cost of solving the local problem twice. The implementation details of the BDDC method are discussed in [35].

3.3. Robin-Robin Interface Condition

As the BDDC preconditioner was originally designed for symmetric positive-definite systems, the method must be modified for nonsymmetric systems. A local bilinear form that conserves the energy stability property of the global form is first derived for the advection equation. Then, the approach is generalized to a symmetrized system of conservation laws such as the Euler equations.

3.3.1. Advection Equation

The variational form of the time-dependent advection equation is: find $u \in V \equiv H^1(\Omega \times I)$, with $I = (0, T)$, such that

$$a(u, v) = \ell(v), \quad \forall v \in V$$

where

$$\begin{aligned} a(u, v) &= (v, u_{,t})_{\Omega\times I} - (v_{,x_j}, \beta_j u)_{\Omega\times I} + (v, \beta_j u n_j)_{\partial\Omega^+\times I} \\ \ell(v) &= (v, f)_{\Omega\times I} - (v, \beta_j g n_j)_{\partial\Omega^-\times I}, \end{aligned}$$

with $(\cdot,\cdot)_{\Omega\times I}$ denoting the L^2 inner product over the space-time domain. The inflow and outflow boundaries are denoted by $\partial\Omega^- = \{x \in \partial\Omega | \beta_j(x) n_j(x) < 0\}$ and $\partial\Omega^+ = \{x \in \partial\Omega | \beta_j(x) n_j(x) > 0\}$, respectively. Integrating by parts the temporal and spatial derivative terms, the energy stability of the advection equation follows from

$$\begin{aligned}
&\tfrac{1}{2}\|u|_{t=T}\|^2_{L^2(\Omega)} + \tfrac{1}{2}(u, u\beta_{j,x_j})_{\Omega\times I} + \tfrac{1}{2}(u, u|\beta_j n_j|)_{\partial\Omega\times I} \\
&\quad = \tfrac{1}{2}\|u|_{t=0}\|^2_{L^2(\Omega)} + (u, f)_{\Omega\times I} + (u, g|\beta_j n_j|)_{\partial\Omega^-\times I}.
\end{aligned}$$

In other words, assuming $\beta_{j,x_j} \geq 0$, the solution at $t = T$ is bounded by the initial state, the source function, and the inflow condition.

Consider designing a local bilinear form, $\check{a}(\cdot,\cdot) : V(\check{\Omega}) \times V(\check{\Omega}) \to \mathbb{R}$, with the criterion that the local form maintains the energy stability of the global bilinear form on a subset $\check{\Omega} \subset \Omega$. Although the analysis generalizes to the case where $\check{\Omega}$ is any subset of Ω, the setting that is most relevant to the domain decomposition method is the case where the subset is the subdomain ($\check{\Omega} = \Omega_i$) and the local bilinear form acts on the subdomain, i.e. $\check{a}(\cdot,\cdot) = a_i(\cdot,\cdot)$ with $a_i(\cdot,\cdot) : V^{(i)} \times V^{(i)} \to \mathbb{R}$. First, consider the local bilinear form and the linear form defined by restricting the global bilinear form and the linear form, respectively, to the subdomain, i.e.

$$\begin{aligned}
\hat{a}_i(u, v) &= (v, u_{,t})_{\Omega_i\times I} - (v_{,x_j}, \beta_j u)_{\Omega_i\times I} + (v, \beta_j u n_j)_{(\partial\Omega_i\cap\partial\Omega^+)\times I} \\
\hat{\ell}(v) &= (v, f)_{\Omega_i\times I} - (v, \beta_j g n_j)_{(\partial\Omega_i\cap\partial\Omega^-)\times I}.
\end{aligned}$$

The energy statement arising from the bilinear form $\hat{a}_i(\cdot,\cdot)$ is

$$\begin{aligned}
&\tfrac{1}{2}\|u|_{t=T}\|^2_{L^2(\Omega_i)} + \tfrac{1}{2}(u, u\beta_{j,x_j})_{\Omega_i\times I} + \tfrac{1}{2}(u, u|\beta_j n_j|)_{(\partial\Omega_i\cap\partial\Omega)\times I} \\
&\quad - \tfrac{1}{2}(u, u|\beta_j n_j|)_{\Gamma_i^+\times I} + \tfrac{1}{2}(u, u|\beta_j n_j|)_{\Gamma_i^-\times I} \\
&\quad = \tfrac{1}{2}\|u|_{t=0}\|^2_{L^2(\Omega_i)} + (u, f)_{\Omega_i\times I} + (u, g|\beta_j n_j|)_{(\partial\Omega_i\cap\partial\Omega^-)\times I}.
\end{aligned}$$

Note, the local problem does not maintain the energy conservation property of the global problem due to the presence of the term $-\frac{1}{2}(u, u|\beta_j n_j|)_{\Gamma_i^+\times I} + \frac{1}{2}(u, u|\beta_j n_j|)_{\Gamma_i^-\times I}$. In particular, the local problem can be

unstable due to the negative term on Γ_i^+. In order to preserve the energy stability property of the global problem, the terms on the interface Γ_i must be eliminated by adding $\frac{1}{2}(u, u\beta_j n_j)_{\Gamma_i \times I}$. The resulting bilinear form is

$$a_i(u, v) = \hat{a}_i(u, v) + \tfrac{1}{2}(u, v\beta_j n_j)_{\Gamma_i \times I}.$$

As the interface term added to one side of the interface is subtracted from the neighboring subdomain, this does not modify the global bilinear form, i.e. $\sum_i^N a_i(u, v) = a(u, v)$. The same modification to the interface condition is proposed in [12] to make the local bilinear form positive definite. The resulting interface condition is called the Robin-Robin interface condition as the resulting local problem involves the Robin boundary condition. The method has been also successfully applied to solve the advection-diffusion equation using Finite Element Tearing and Interconnect [13] and recently BDDC [14].

3.3.2. Interface Condition for Symmetrized System of Equations

The Robin-Robin interface condition can be generalized to a system of symmetrized nonlinear equations. Recalling a linearized system is solved at each Newton step, the objective is to specify the interface condition such that the local linearized problem maintains the energy stability of the global linearized problem. For a time-dependent hyperbolic system of conservation laws, the semilinear form is expressed as

$$\mathcal{R}(u, v) = (A_{0kl} u_{l,t}, v_k)_{\Omega \times I} - (F_{jk}^{\text{inv}}, v_{k,x_j})_{\Omega \times I} + (\hat{F}_k(u, \text{B.C.data}, n), v_k)_{\partial\Omega \times I}.$$

and the linearized equation is

$$\mathcal{R}^{\text{lin}}(u, v; w) = (A_{0kl}|_w u_{l,t}, v_k)_{\Omega \times I} - (A_{jkl}|_w u_l, v_{k,x_j})_{\Omega \times I} + (A_{bc,kl}|_w u_l, v_k)_{\partial\Omega \times I},$$

where $A_{bc} \equiv \frac{\partial \hat{F}}{\partial u}$ and the bilinear form is parametrized about the previous state, i.e. $w = u^n$. As the linearization is always about the previous state u^n in this section, the explicit notation $(\cdot)|_{u^n}$ is dropped for

the rest of the section for brevity. The energy statement of the global equation is of the form

$$\begin{aligned}&\tfrac{1}{2}\|u|_{t=T}\|^2_{A_0,\Omega} - \tfrac{1}{2}(A_{0kl,t}u_l, u_k)_{\Omega\times I} + \tfrac{1}{2}(A_{jkl,x_j}u_l, u_k)_{\Omega\times I} - \tfrac{1}{2}(A(n)_{kl}u_l, u_k)_{\partial\Omega\times I}\\ &\quad + (A_{bc,kl}u_l, v_k)_{\partial\Omega\times I} = \tfrac{1}{2}\|u|_{t=0}\|^2_{A_0,\Omega}\end{aligned}$$

where $\|u\|_{A_0,\Omega} \equiv (A_0 u, u)_\Omega$ is the energy norm weighted by the symmetric positive definite matrix A_0. Note, for a linear system of equations with the boundary flux $\hat{F}_k = A^+_{kl}u_l + A^-_{kl}g_l$, the boundary flux Jacobian is given by $A_{bc} = A^+$, and the integrals on the boundary can be simplified as $-\frac{1}{2}(A(n)_{kl}u_l, u_k)_{\partial\Omega\times I} + (A_{bc,kl}u_l, v_k)_{\partial\Omega\times I} = \frac{1}{2}(|A|u, u)_{\partial\Omega\times I}$.

In order to develop an appropriate local bilinear form, first consider the form obtained by simply restricting the global linearized form to the local space, i.e.

$$\hat{\mathcal{R}}^{\text{lin}}_i(u, v) = (A_{0kl}u_{l,t}, v_k)_{\Omega_i\times I} - (A_{jkl}u_l, v_{k,x_j})_{\Omega_i\times I} + (A_{bc,kl}u_l, v_k)_{(\partial\Omega_i\cap\partial\Omega)\times I}$$

Integrating by parts the temporal and spatial derivative terms and taking advantage of the symmetry of A_0 and A_j, the energy statement for the local bilinear form is

$$\begin{aligned}&\tfrac{1}{2}\|u|_{t=T}\|^2_{A_0,\Omega_i} - \tfrac{1}{2}(A_{0kl,t}u_l, u_k)_{\Omega_i\times I} + \tfrac{1}{2}(A_{jkl,x_j}u_l, u_k)_{\Omega_i\times I}\\ &\quad - \tfrac{1}{2}(A(n)_{kl}u_l, u_k)_{(\partial\Omega_i\cap\partial\Omega)\times I} + (A_{bc,kl}u_l, v_k)_{(\partial\Omega_i\cap\partial\Omega)\times I} - \tfrac{1}{2}(A(n)_{kl}u_l, u_k)_{\Gamma_i\times I}\\ &\qquad = \tfrac{1}{2}\|u|_{t=0}\|^2_{A_0,\Omega_i}\end{aligned}$$

Again, due to the presence of the term $-\frac{1}{2}(A_{kl}(n)u_l, u_k)_{\Gamma_i\times I}$, the local energy statement is not consistent with the global energy statement. Using the same approach as the advection equation case, the consistent local energy statement is derived by modifying the local linearized form to

$$\mathcal{R}^{\text{lin}}_i(u, v) = \hat{\mathcal{R}}^{\text{lin}}_i(u, v) + \tfrac{1}{2}(A_{kl}(n)u_l, v_k)_{\Gamma_i\times I}.$$

A local semilinear form that has the desired linearized form is

$$\mathcal{R}_i(u,v) = -(F_{jk}^{\text{inv}}, v_{k,x_j})_{\Omega_i} + \tfrac{1}{2}(F_{jk}^{\text{inv}} n_i, v_k)_{\Gamma_i} + (\hat{F}_k(u, \text{B.C.data}, n), v_k)_{\partial\Omega\cap\partial\Omega_i}.$$

Thus, for a system of nonlinear equations with a symmetric flux Jacobian, the Robin-Robin interface condition corresponds to adding half of the nonlinear flux on the interfaces.

4. Inexact BDDC

For a large scale problem, the cost of computing and storing the exact factorization of $A_{rr}^{(i)}$ and $A_{II}^{(i)}$ can be prohibitively high, and the cost only escalates for three-dimensional problems. In this section, the inexact BDDC preconditioner is constructed by replacing the solutions to the Dirichlet and the partially subassembled problem with actions of inexact local solvers. The inexact BDDC algorithms have been previously applied to linear systems arising from the linear finite element discretization of elliptic systems [15, 16]. In [15], the exact local solvers are replaced by the action of h-multigrid V-cycles and applied to the Poisson equation, for which the multigrid converges uniformly. In [16], the incomplete factorization is employed to approximate the solutions to local problems.

The local solver used in this work is a two-level multiplicative preconditioner consisting of the dual-threshold incomplete factorization (ILUT) [17] and a $p = 1$ coarse correction, following the approach used in [19] for the high-order discontinuous Galerkin discretization. The use of the two-level preconditioner is motivated by the presence of the convective mode and the elliptic mode in the equations of interest. The incomplete factorization, with a proper reordering, have been shown to work well for advection-dominated problems [19, 20]. This work uses the minimum discarded fill (MDF) algorithm introduced in [18], and generalized to block matrices in [19]. The method orders the unknown that produces the least discarded fill-in first and repeats the process in a greedy manner. The incomplete factorization is performed block-wise to take advantage of the block structure of the stiffness matrix arising from the high-order discretization of system of equations. The Frobenius norm is used to measure the contribution of a given block to the factorization. The coarse correction is performed by projecting the high-order solution to the $p = 1$ space, and solving the coarse problem exactly. The prolongation operator $P : V_{h,1} \to V_{h,p}, p > 1$ is defined by a

simple interpolation.

The two-level preconditioner induced by the $p = 1$ variational coarse correction followed by the ILUT smoothing is

$$M_{MG,1}^{-1} = PA_0^{-1}P^T + M_s^{-1}(I - APA_0^{-1}P^T), \tag{3}$$

where $A_0 \equiv P^T AP$ is the coarse correction matrix and M_s^{-1} represents the action of the ILUT preconditioner. Alternatively, the order of the multiplicative projectors can be commutated, resulting in the preconditioner given by

$$M_{MG,2}^{-1} = PA_0^{-1}P^T + (I - PA_0^{-1}P^T A)M_s^{-1}. \tag{4}$$

When a preconditioner defined by Eq. (3) or (4) is used as the approximate local solver for the BDDC algorithm, the matrix A is replaced by local matrices as described in Section 4.1 and 4.2.

As shown in Table 1, the stiffness matrix for the linear finite element discretization is significantly smaller than that for the high-order discretization, making the direct solution to the coarse problem practical at least for the problems considered in two dimensions. The coarse problem arising from h-multigrid based on the variational correction is known to be unstable due to the mesh-dependent bilinear form of the GLS discretization [36, 37]. However, a similar problem has not been observed for the p-based coarse correction despite the presence of the p-dependent stabilization parameter, τ.

4.1. Inexact Discrete Harmonic Extensions

The inexact discrete harmonic extension operators are given by

$$\tilde{\mathcal{H}}_1 \approx \begin{pmatrix} I & M_{II,1}^{-1}\tilde{A}_{I\Gamma}J_{D,\Gamma} \\ & \tilde{R}_{D,\Gamma}^T \end{pmatrix} \quad \text{and} \quad \tilde{\mathcal{H}}_2 \approx \begin{pmatrix} I & \\ J_{D,\Gamma}\tilde{A}_{\Gamma I}M_{II,2}^{-1} & \tilde{R}_{D,\Gamma} \end{pmatrix},$$

where $M_{II,1}^{-1}$ and $M_{II,2}^{-1}$ are approximate solvers. When ILUT is used as the approximate local solver, $M_{II,1}^{-1} = M_{II,2}^{-1} = M_{II,\mathrm{ILUT}}^{-1}$, where $M_{II,\mathrm{ILUT}}^{-1}$ denote the operator obtained by applying the incomplete

factorization to A_{II}. Similar to the exact factorization case discussed in Section 3.2, the approximate extension operators $\tilde{\mathcal{H}}_1$ and $\tilde{\mathcal{H}}_2$ can be applied by a single backward solve and forward solve, respectively, of the approximation factorization.

When ILUT with the $p = 1$ coarse correction (ILUT-$p1$) is used, the preconditioners are given by

$$M_{II,1}^{-1} = PA_{II,0}^{-1}P^T + (I - PA_{II,0}^{-1}P^T A_{II})M_{II,\mathrm{ILUT}}^{-1}$$
$$M_{II,2}^{-1} = PA_{II,0}^{-1}P^T + M_{II,\mathrm{ILUT}}^{-1}(I - A_{II}PA_{II,0}^{-1}P^T)$$

The order of the coarse correction and the smoothing are different for the two cases to exploit the structure of the ILUT factorization generated. Namely, when the projectors are applied in this order, the extension operator $\tilde{\mathcal{H}_1}$ and $\tilde{\mathcal{H}_2}$ require a single backward solve and forward solve, respectively, to apply the ILUT smoother, $M_{II,\mathrm{ILUT}}^{-1}$.

Note, due to the block structure of A_{II}, both the ILUT smoothing and the coarse correction can be performed in parallel. In other words, applying the ILUT-$p1$ preconditioner to each local problem results in the ILUT-$p1$ preconditioning of the global problem.

4.2. Inexact Partially Subassembled Solve

Two types of the inexact partially assembled solvers are considered in this study. The first is based on performing the ILUT factorization on the partially assembled matrix, i.e.

$$\tilde{A}^{-1} \approx \tilde{M}_{\mathrm{ILUT}}^{-1}.$$

The second is based on the ILUT factorization with the $p = 1$ coarse correction, i.e.

$$\tilde{A}^{-1} \approx \tilde{P}\tilde{A}_0^{-1}\tilde{P}^T + \tilde{M}_{\mathrm{ILUT}}^{-1}(I + \tilde{A}\tilde{P}\tilde{A}_0^{-1}\tilde{P}^T),$$

where $\tilde{A}_0 \equiv \tilde{P}^T\tilde{A}\tilde{P}$ is the $p = 1$ correction of the partially assembled system and $\tilde{M}_{\mathrm{ILUT}}^{-1}$ is the ILUT factorization of the partially assembled system. The prolongation operator, $\tilde{P}$, interpolates the $p = 1$ solution on the partially assembled space to the higher-order partially assembled space.

5. Results

5.1. BDDC with Exact Local Solver

5.1.1. Advection-Diffusion Equation: Isotropic Mesh

The advection-diffusion equation is solved on a unit square domain to evaluate the performance of the parallel preconditioner applied to the high-order GLS discretization. The cross-stream boundary layer problem solved has the solution that is essentially 1 everywhere, except it decreases to 0 in the vicinity of the $y = 0$ boundary, as shown in Figure 2(a). The boundary layer thickness is inversely proportional to the square root of the Peclet number. The solution is initialized to random values, which results in a random right hand side vector for the linear system, and the system is solved using BDDC preconditioned GMRES until the 2-norm of the preconditioned residual reduces by a factor of 10^{13}. Three different values of the viscosity are considered: the diffusion-dominated case ($\kappa = 10^2$), the balanced advection-diffusion case ($\kappa = 10^{-2}$), and the advection-dominated case ($\kappa = 10^{-6}$). The Robin-Robin interface condition is used for all cases.

The result of solving the advection-diffusion equation using the BDDC preconditioner with exact local solvers is shown in Table 2. In the upper half of the table, the size of the subdomain (H/h) is fixed to 8×8 (128 triangular elements), and the number of subdomains is varied from four to 256. In the lower half of the table, the number of subdomains is fixed to 16 and the size of the subdomain is varied from $H/h = 4$ to $H/h = 16$. For each case, the interpolation order is varied from $p = 1$ to 4.

In the diffusion-dominated case ($\kappa = 10^2$), the iteration count is independent of the number of subdomains for $N \geq 64$ for the corner constraints case and $N \geq 16$ for the corner and edge average constraints case. When the size of subdomains is varied while keeping number of subdomains fixed, the iteration count increases slowly, as expected from the $(1 + \log(H/h))^2$ dependence of the condition number on the size of the subdomain. [38] In general, using the higher-order interpolation increases the iteration count, but the scalability is maintained for all values of p.

For the advection-dominated case, the iteration count increases with the maximum number of the subdomains that a characteristic passes through, N_{char}, as the hyperbolic equation does not possess the smoothing property of the elliptic equation. In particular, the residual does not decay significantly for the first $N_{\text{char}}/2$ iterations, but decays rapidly after $N_{\text{char}}/2$ iterations; this is consistent with the theoretical analysis of the

Robin-Robin interface condition conducted in [12]. Note, the iteration count is largely independent of the size of the subdomain, the interpolation order, and the choice of constraints. Thus, in the advection limit, a partitioning strategy based on capturing the strong characteristics, such as those used in [39] and [40], is expected to improve the performance of the preconditioner. However, these strategies are sequential in nature and tend to increase the communication volume, especially in three-dimensions.

5.1.2. Advection-Diffusion Equation: Anisotropic Mesh

In realistic problems, highly-anisotropic meshes are employed to resolve the thin boundary layer of high Peclet number flows. The meshes used in this section have the exponential y-spacing such that the elements on the boundary have the aspect ratio approximately equal to $1/\sqrt{\kappa}$. Thus, for $\kappa = 10^{-6}$, elements on the boundary have the aspect ratio of 1000, and the area of an element on the boundary is less than 1/1000 of the largest element in the domain.

Table 3 shows the scalability result using the series of anisotropic meshes. Comparing to Table 2, the iteration count increases in general. In particular, the performance degrades considerably on the fine mesh used for the 256-subdomain partition. The difference in the iteration count for the corner only and the corner and edge average constraints suggests that the problem exhibits diffusive behavior in the boundary layer region even for $\kappa = 10^{-6}$ with the highly anisotropic mesh.

5.1.3. Euler Equation

The performance of the BDDC method applied to the Euler equations is assessed in this section. The equation is solved on a $(-10, 10)\times(0, 10)$ rectangular domain with a Gaussian bump, $y = 1/(\sigma\sqrt{2\pi})\exp(-x/(2\sigma^2))$, with $\sigma = 5/6$, as shown in Figure 2(b). The stagnation pressure, stagnation temperature, and the flow angle are specified at the inflow, and the static pressure is specified at the outflow such that the freestream Mach number is $M_\infty = 0.2$. The boundary state for the subsonic inflow and the outflow are reconstructed from the compatibility relations of the Riemann invariants. The flow tangency condition is set on the upper and the lower walls.

The problem is solved using the subdomain-wise block Jacobi, BDDC with corner constraints, and BDDC with corner and edge average constraints. Both naturally arising and Robin-Robin interface condition are

considered. The size of the subdomain is fixed to 160 elements, and increasingly larger problems are solved as more subdomains are added. The linear system arising from the Jacobian for the converged solution is used, and the CFL number is set to infinity so that there is no mass matrix contribution to the linear system.

Table 4 shows the result of the comparison. For all types of the preconditioners considered, the Robin-Robin interface condition improves the performance of the preconditioner significantly. Unlike the advection-dominated case of the advection-diffusion equation on uniform meshes, the iteration count is dependent on the type of constraints for the BDDC method, and using both the corner and edge average constraints reduces the iteration count, especially when a large number of subdomains is employed. The difference suggests the underlying ellipticity of the acoustic modes in steady, subsonic flow.

Table 5 shows the variation in the iteration count with the size of the subdomains using eight subdomains for the BDDC method with the Robin-Robin interface condition and the corner and edge average constraints. There is no significant increase in the iteration count as the subdomain size grows, particularly when 640 or more elements are employed per subdomain for all interpolation orders considered.

5.2. BDDC with Inexact Local Solvers

This section presents the result of solving the advection-diffusion equation using the inexact BDDC algorithm. Before using inexact solvers with the BDDC algorithm, the performance of the inexact solvers on a single subdomain is studied. Table 6 shows the result of solving the linear system arising from the $p = 4$ discretization of the advection-diffusion equations on uniform meshes for three different values of viscosity: $\kappa = 10^2, 10^{-2}, 10^{-6}$. In the advection-dominated cases ($\kappa = 10^{-6}$), the ILUT with or without the $p = 1$ coarse correction achieves low iteration count, as the MDF algorithm orders the unknowns along the characteristics and the factorization becomes nearly exact. This is similar to the result obtained for the DG discretization in [19], despite the worse connectivity structure of the stiffness matrix arising from the high-order GLS discretization. In the diffusion-dominated case, the ILUT with $p = 1$ coarse correction performs significantly better than the method without the coarse correction. In the absence of the coarse correction, the iteration count increases linearly with $1/h$, whereas the iteration count is independent of h with the $p = 1$ coarse correction.

Table 7 shows the result of applying the inexact BDDC algorithm discussed in Section 4 to the $p = 4$

discretization of the advection-diffusion equation. The result of solving the equation using the exact local solver is reproduced for convenience. Similar to the single domain case, the incomplete factorization with or without the $p = 1$ coarse correction achieves low iteration count for the advection-dominated cases. However, in the diffusion-dominated case, the performance of the BDDC algorithm degrades significantly when only the ILUT is employed as the local solver. With the $p = 1$ correction, the inexact BDDC algorithm achieves a similar iteration count as the exact counterpart. For some cases, the BDDC preconditioner with ILUT-$p1$ local solvers achieves lower iteration count than the exact local solvers due to the slight difference in the convergence criteria, as the preconditioned residual is used as the convergence criterion.

5.2.1. Cost Estimate for the Inexact BDDC

In order to assess the benefit of using the BDDC algorithm based on approximate local solvers, the cost of solving a linear system is estimated in terms of the number of floating point operations (FLOPS). The objective of this section is to obtain an order of magnitude estimate of the computational cost; characterization of the precise execution time is not attempted as it is highly dependent on the implementation and the hardware architecture.

In general, the key difference in the computational cost for the exact and inexact local solver is its scaling with the size of the local problem. If n is the degrees of freedom associated with the local problem, the computational cost and the memory requirements scale as $\mathcal{O}(n^{3/2})$ and $\mathcal{O}(n \log(n))$, respectively, for the exact factorization using optimal reordering in two dimensions. The scaling worsens in three dimensions, with the computational cost and the memory equation scaling as $\mathcal{O}(n^2)$ and $\mathcal{O}(n^{4/3})$, respectively. On the other hand, both the computational cost and the memory requirement for an incomplete factorization scales as $\mathcal{O}(n)$. Thus, the difference between the exact and inexact BDDC algorithm is more pronounced for large problems.

As an example, consider a $p = 4$ discretization of the advection-diffusion equation on a subdomain consisting of 32768 triangular elements (128×128). Approximately 50MB of memory is required to store the local stiffness matrix, which is a moderate requirement for modern computers. The computational cost for a linear solve is consisting of two parts: the preprocessing stage and the iterative stage. In the preprocessing stage, the local matrices and the primal Schur complement, $S_{\Pi\Pi}$, are factorized, and the $p = 1$ matrix for the

two-level multiplicative preconditioner is formed and factorized (if ILUT-$p1$ is used). In the iterative stage, the BDDC algorithm requires application of the discrete harmonic extensions and solution to the partially assembled system $\tilde{A}$, as well as the matrix-vector multiplication and Gram-Schmidt orthogonalization for GMRES.

The major operations required in the preprocessing stage and the iterative stage are summarized in Table 8. The cost for each of the operation is calculated for the $p = 4$ discretization of the advection-diffusion equation mentioned above. The total linear solver cost is estimated assuming the GMRES iteration count does not change significantly from the 128 element per subdomain case. The result of the comparison is shown in Figure 3. For both the diffusion-dominated case and the advection-dominated case, the inexact BDDC based on the two-level multiplicative local solver reduces the estimated computed cost by an order of magnitude compare to the BDDC based on the exact local solver. The local solver based on the ILUT alone does not scale well in the diffusion-dominated case as the cost increases with the GMRES iteration count. While the ILUT local solver achieves the lowest cost in the advection-dominated case, the ILUT-$p1$ local solver reduces the cost for the wide range of the flow regimes considered.

It is worth mentioning that the computational cost for the factorization of $S_{\Pi\Pi}$ is approximately five and three orders of magnitude less than the direct and ILUT factorization, respectively, of $A^{(i)}$ even for the case with 256 processors. The cost of solving the $S_{\Pi\Pi}$ in each iteration step is also approximately three orders of magnitude less than the cost for the rest of the operations for computing the preconditioned operator. Thus, the estimate confirms that the BDDC algorithm is highly scalable and is well suited for massively parallel systems.

6. Conclusion

This work presented the high-order accurate Galerkin Least-Squares method combined with a BDDC method to provide efficient solutions on massively parallel architectures. The BDDC method was applied to the high-order discretization of the advection-diffusion equation and the Euler equations. In the diffusion-dominated case, the BDDC method achieves an iteration count independent of the number of subdomains, as predicted by the theory. In the advection-dominated case, the method maintains low iteration count when

the Robin-Robin interface condition is employed such that the local problem inherit the energy stability of the global problem. The Robin-Robin interface condition was generalized to the Euler equations using the entropy symmetrization theory, and the BDDC method is shown to achieve good performance for the subsonic inviscid flow considered.

The paper also presented the inexact BDDC method based on inexact local solvers. Two inexact local solvers are used in this study: the ILUT preconditioner and the two-level multiplicative preconditioner based on the ILUT and $p = 1$ coarse correction. On a single domain case, the ILUT preconditioner with the minimum discarded fill ordering achieves low iteration counts in the advection-dominated case. In diffusion-dominated cases, the use of a $p = 1$ coarse correction is needed to maintain an iteration count independent of the grid size. When combined with the BDDC preconditioner, the ILUT preconditioner with the MDF ordering was sufficient to maintain low iteration counts in the advection-dominated case. However, the $p = 1$ coarse correction was necessary to maintain scalability in the diffusion-dominated cases. The inexact BDDC method based on the ILUT-$p1$ local solver provides an effective strategy to maintain good parallel scaling for high-order discretization over a wide range of flow regimes.

Acknowledgment

This work was supported by funding from The Boeing Company with technical monitor Dr. Mori Mani.

References

[1] A. N. Brooks, T. J. R. Hughes, Streamline upwind / petrov-galerkin formulations for convection dominated flows with particular emphasis on the incompressible navier-stokes equations, Computer methods in applied mechanics and engineering 32 (1982) 199–259.

[2] T. J. R. Hughes, T. E. Tezduyar, Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible Euler equations, Comput. Methods Appl. Mesh. Engrg. 45 (1984) 217–284.

[3] T. J. R. Hughes, M. Mallet, A new finite element formulation for computational fluid dynamics: III The generalized streamline operator for multidimensional advective-diffusive systems, Comput. Methods Appl. Mech. Engrg. 58 (1986) 305–328.

[4] T. J. Hughes, G. Engel, L. Mazzei, M. G. Larson, The continuous Galerkin method is locally conservative, Journal of Computational Physics 163 (2000) 467–488.

[5] V. Venkatakrishnan, S. R. Allmaras, D. S. Kamenetskii, F. T. Johnson, Higher order schemes for the compressible Navier-Stokes equations, AIAA Paper 2003-3987 (2003).

[6] C. R. Dohrmann, A preconditioner for substructuring based on constrained energy minimization, SIAM Journal on Scientific Computing 25 (1) (2003) 246–258.

[7] T. J. R. Hughes, L. P. Franca, G. M. Hulbert, A new finite element formulation for computational fluid dynamics: VIII. the Galerkin/least-squares method for advective-diffusive equations, Comput. Methods Appl. Mech. Eng. 73 (1989) 173–189. doi:http://dx.doi.org/10.1016/0045-7825(89)90111-4.

[8] J. Mandel, C. R. Dohrmann, R. Tezaur, An algebraic theory for primal and dual substructuring methods by constraints, Applied Numerical Mathematics 54 (2005) 167–193.

[9] J. Li, O. B. Widlund, FETI-DP, BDDC, and block Cholesky methods, International Journal for Numerical Methods in Engineering 66 (2006) 250–271.

[10] S. C. Brenner, L. yeng Sung, BDDC and FETI-DP without matrices or vectors, Comput. Methods Appl. Mech. Engrg. 196 (2007) 1429–1435.

[11] A. Klawonn, L. F. Pavarino, O. Rheinbach, Spectral element FETI-DP and BDDC preconditioners with multi-element subdomains, Comput. Methods Appl. Mech. Engrg. 198 (2008) 511–523.

[12] Y. Achdou, P. L. Tallec, F. Nataf, M. Vidrascu, A domain decomposition preconditioner for an advection-diffusion problem, Computer Methods in Applied Mechanics and Engineering 184 (2000) 145–170.

[13] A. Toselli, FETI domain decomposition methods for scalar advection-diffusion problems, Computer Methods in Applied Mechanics and Engineering 190 (2001) 5759–5776.

[14] X. Tu, J. Li, A balancing domain decomposition method by constraints for advection-diffusion problems, Communications in Applied Mathematics and Computational Science 3 (1) (2008) 25–60.

[15] J. Li, O. B. Widlund, On the use of inexact subdomain solvers for BDDC algorithms, Computational Methods in Applied Mechanics and Engineering 196 (2007) 1415–1428.

[16] C. R. Dohrmann, An approximate BDDC preconditioner, Numerical Linear Algebra with applications 14 (2007) 149–168.

[17] Y. Saad, ILUT: a dual threshold incomplete LU factorization, Numerical Linear Algebra with Applications 1 (4) (1994) 387–402.

[18] E. F. D'Azevedo, P. A. Forsyth, W.-P. Tang, Ordering methods for preconditioned conjugate gradient methods applied to unstructured grid problems, SIAM J. matrix anal. appl. 13 (3) (1992) 944–961.

[19] P.-O. Persson, J. Peraire, Newton-GMRES preconditioning for Discontinuous Galerkin discretizations of the Navier-Stokes equations, SIAM Journal on Scientific Computing 30 (6) (2008) 2709–2722.

[20] L. T. Diosady, D. L. Darmofal, Preconditioning methods for discontinuous Galerkin solutions of the Navier-Stokes equations, Journal of Computational Physics.

[21] A. Harten, On the symmetric form of systems of conservation laws with entropy, Journal of computational physics 49 (1983) 151–164.

[22] T. J. R. Hughes, L. Franca, M. Mallet, A new finite element formulation for computational fluid dynamics: I Symmetric forms of the compressible Euler and Navier-Stokes equations and the second law of thermodynamics, Comput. Methods Appl. Mech. Engrg. 54 (1986) 223–234.

[23] T. J. Barth, Numerical methods for gasdynamic systems on unstructured meshes, in: D. Kroner, M. Olhberger, C. Rohde (Eds.), An Introduction to Recent Developments in Theory and Numerics for Conservation Laws, Springer-Verlag, 1999, pp. 195 – 282.

[24] T. J. R. Hughes, Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods, Comput. Methods Appl. Mech. Engrg. 127 (1995) 387–401.

[25] T. J. R. Hughes, G. R. Feijoo, L. Mazzei, J.-B. Quincy, The variational multiscale method - a paradigm for computational mechanics, Comput. Methods Appl. Mech. Engrg. 166 (1998) 3–24.

[26] T. J. R. Hughes, A simple shcme for developing upwind finite elements, International journal for numerical methods in engineering 12 (1978) 1359–1365.

[27] T. J. R. Hughes, M. Mallet, A. Mizukami, A new finite element formulation for computational fluid dynamics: II Beyond SUPG, Comput. Methods Appl. Mech. Engrg. 54 (1986) 341–355.

[28] I. Harari, T. J. R. Hughes, What are c and h? : Inequalities for the analysis and design of finite element methods, Comput. Methods Appl. Mech. Engrg. 97 (1992) 157–192.

[29] T. Apel, G. Lube, Anisotropic mesh refinement in stabilized Galerkin methods, Numer. Math. 74 (1996) 261–282.

[30] S. Micheletti, S. Perotto, M. Picasso, Stabilized finite elements on anisotropic meshes: A priori error estimates for the advection-diffusion and the stokes problems, SIAM J. Numer. Anal. 41 (3) (2003) 1131–1162.

[31] Y. Saad, M. H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM Journal on Scientific and Statistical Computing 7 (3) (1986) 856–869.

[32] A. Klawonn, O. B. Widlund, Dual-primal FETI methods for linear elasticity, Communications on Pure and Applied Mathematics 59 (2006) 1523–1572.

[33] A. Klawonn, O. B. Widlund, M. Dryja, Dual-primal FETI methods for three-dimensional elliptic problems with heterogeneous coefficients, SIAM Journal of Numerical Analysis 43 (1) (2002) 159–179.

[34] A. Toselli, O. Widlund, Domain Decomposition Methods Algorithm and Theory, Springer-Verlag, 2005.

[35] M. Yano, Massively parallel solver for the high-order galerkin least-squares method, Masters thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics (May 2009).

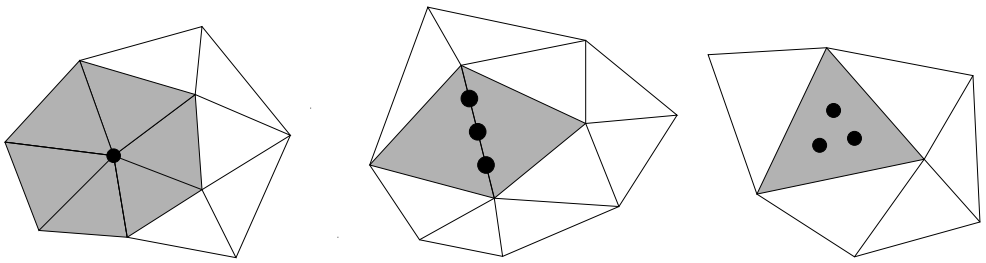
[36] T. O. Okusanya, Algebraic multigrid for stabilized finite element discretizations of the Navier-Stokes equations, PhD dissertation, M.I.T., Department of Aeronautics and Astronautics (June 2002).

[37] T. Okusanya, D. Darmofal, J. Peraire, Algebraic multigrid for stabilized finite element discretizations of the Navier-Stokes equations, Computational Methods in Applied Mechanics and Engineering 193 (1) (2004) 3667–3686.

[38] J. Mandel, C. R. Dohrmann, Convergence of a balancing domain decomposition by constraints and energy minimization, Numerical Linear Algebra with Applications 10 (2003) 639–659.

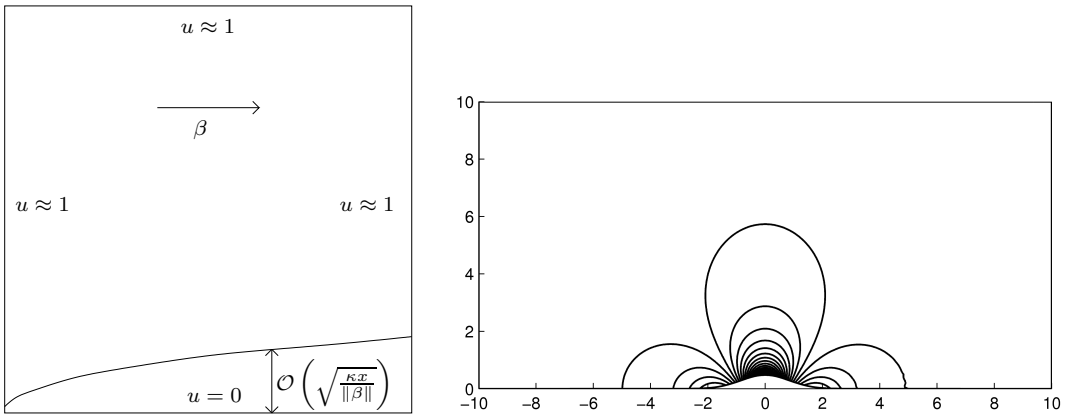
[39] L. T. Diosady, A linear multigrid preconditioner for the solution of the Navier-Stokes equations using a discontinuous Galerkin discretization, Masters thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics (May 2007).

[40] P.-O. Persson, Scalable parallel Newton-Krylov solvers for discontinuous Galerkin discretizations, AIAA Paper 2009-606 (2009).



(a) Node mode (b) Edge mode (c) Element mode

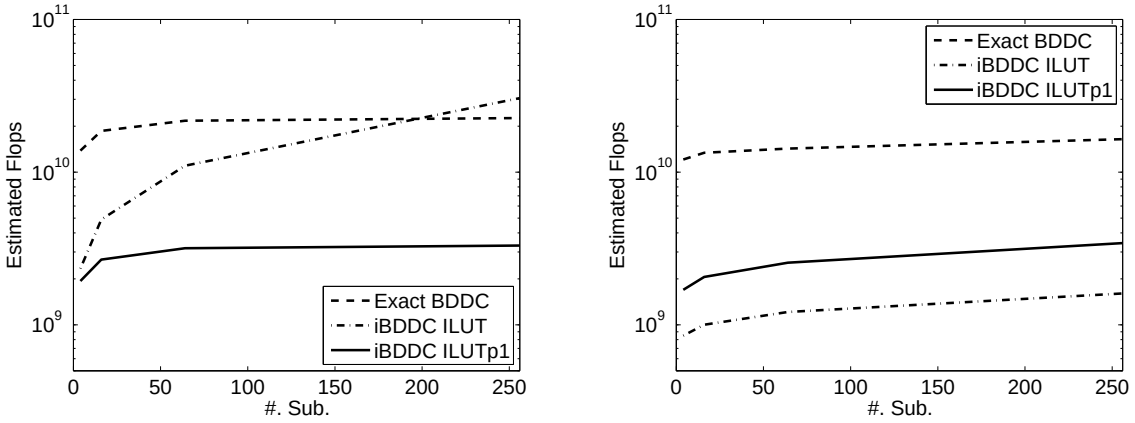
Figure 1: Diagram of basis modes and their support. The degree of freedom for each mode for $p = 4$ discretization is shown in dots.



(a) Cross-stream boundary layer

(b) Euler Gaussian bump

Figure 2: Advection-diffusion problem on a unit square domain and inviscid subsonic flow over a Gaussian bump.



(a) $\kappa = 10^2$ (b) $\kappa = 10^{-6}$

Figure 3: Estimated cost of the linear solve for the $p = 4$ discretization of the advection-diffusion equation with each subdomain having 32768 elements.

p	1	2	3	4	5
Elemental basis	3	6	10	15	21
DOF/elem.	0.5	2.0 (4)	4.5 (9)	8.0 (16)	12.5 (25)
Matrix nnz/elem.	3.5	23.0 (6.6)	76.5 (21.9)	188.0 (53.7)	387.5 (110.7)

Table 1: The degrees of freedom associated with high-order continuous Galerkin discretization on an average mesh. Numbers in the parenthesis are the ratios with respect to the $p = 1$ discretization.

		$\kappa = 10^2$				$\kappa = 10^{-2}$				$\kappa = 10^{-6}$			
#. Sub.	H/h	$p = 1$	2	3	4	1	2	3	4	1	2	3	4
4	8	5	7	7	7	5	8	8	8	3	3	3	3
16		11	15	17	18	8	10	10	11	7	6	6	6
64		16	21	24	25	11	13	14	16	10	9	8	8
256		17	22	24	27	19	22	23	25	14	13	12	13
16	4	10	13	15	16	8	10	11	11	10	10	9	9
	8	11	15	17	18	8	10	10	11	7	6	6	6
	16	13	17	18	18	8	10	11	12	5	5	5	5

(a) Corner constraints only

		$\kappa = 10^2$				$\kappa = 10^{-2}$				$\kappa = 10^{-6}$			
#. Sub.	H/h	$p = 1$	2	3	4	1	2	3	4	1	2	3	4
4	8	4	6	6	7	6	8	8	8	4	4	4	4
16		7	10	11	12	8	10	10	11	8	7	7	7
64		8	11	12	14	10	12	13	14	11	10	10	9
256		8	10	12	13	10	12	14	16	15	14	14	13
16	4	6	8	10	11	8	10	11	11	10	10	9	9
	8	7	10	11	12	8	10	10	11	8	7	7	7
	16	9	11	12	13	8	10	11	12	6	6	6	6

(b) Corner and edge average constraints

Table 2: The GMRES iteration count for the BDDC method based on exact local solvers for the boundary layer problem on uniform meshes.

		$\kappa = 10^{-2}$				$\kappa = 10^{-4}$				$\kappa = 10^{-6}$			
#. Sub.	H/h	$p = 1$	2	3	4	1	2	3	4	1	2	3	4
4	8	7	9	9	8	7	9	9	10	8	9	10	11
16		9	10	12	13	10	11	13	15	9	11	11	12
64		18	20	21	22	17	18	18	19	14	15	17	19
256		37	40	41	43	45	38	37	36	34	36	41	46
16	4	9	11	10	11	11	11	12	14	12	12	12	12
	8	9	10	12	13	10	11	13	15	9	11	11	12
	16	9	12	13	15	10	12	14	15	9	10	12	13

(a) Corner constraints only

		$\kappa = 10^{-2}$				$\kappa = 10^{-4}$				$\kappa = 10^{-6}$			
#. Sub.	H/h	$p = 1$	2	3	4	1	2	3	4	1	2	3	4
4	8	7	9	9	8	7	9	10	11	7	8	9	10
16		8	10	11	12	9	11	12	13	9	10	11	12
64		12	14	16	17	14	15	17	18	12	13	15	17
256		19	23	25	27	31	26	26	28	26	28	30	32
16	4	8	11	10	10	10	11	12	13	11	12	12	13
	8	8	10	11	12	9	11	12	13	9	10	11	12
	16	9	11	13	14	10	12	13	14	8	10	12	12

(b) Corner and edge average constraints

Table 3: The GMRES iteration count for the BDDC method based on exact local solvers for the boundary layer problem on anisotropic meshes.

	Subdomain Block Jacobi			BDDC (Corner only)			BDDC (Corner + edge)		
#. Sub.	$p=1$	$p=2$	$p=3$	$p=1$	$p=2$	$p=3$	$p=1$	$p=2$	$p=3$
2	50	87	113	34	57	75	30	55	74
8	190	359	565	156	284	428	223	276	455
32	914	-	-	580	-	-	476	-	-

(a) Naturally arising interface condition

	Subdomain Block Jacobi			BDDC (Corner only)			BDDC (Corner + edge)		
#. Sub.	$p=1$	$p=2$	$p=3$	$p=1$	$p=2$	$p=3$	$p=1$	$p=2$	$p=3$
2	37	50	63	16	20	21	15	18	20
8	92	155	199	51	77	106	38	66	89
32	209	334	415	90	130	180	52	82	116
128	478	-	-	158	220	309	65	106	159

(b) The Robin-Robin interface condition

Table 4: The GMRES iteration count for the Euler bump problem using exact local solvers. (160 elem. per subdomain, $\Delta t = \infty$)

Elem. per sub.	$p = 1$	$p = 2$	$p = 3$
40	30	49	71
160	38	66	89
640	46	74	100
2560	52	78	102

Table 5: Variation in the GMRES iteration count with the size of subdomains using the BDDC method with the Robin-Robin interface condition and exact local solvers. (8 subdomains, corner and edge constraints)

	ILUT($10^{-8}, 2$)			ILUT($10^{-8}, 2$)-$p1$		
h	$\kappa = 10^2$	10^{-2}	10^{-6}	10^2	10^{-2}	10^{-6}
1/8	22	9	6	10	7	5
1/16	40	11	6	10	7	6
1/32	74	17	7	9	8	6
1/64	145	32	8	8	7	7
1/128	290	66	8	7	7	7

Table 6: The GMRES iteration count for $p = 4$ discretization of the advection-diffusion equation using the ILUT and ILUT with $p = 1$ coarse correction on a single domain (uniform mesh).

		ILUT($10^{-8}, 2$)			ILUT($10^{-8}, 2$)-$p1$			Exact		
# Sub.	H/h	$\kappa = 10^2$	10^{-2}	10^{-6}	10^2	10^{-2}	10^{-6}	10^2	10^{-2}	10^{-6}
4	8	38	12	12	13	9	11	7	8	3
16		73	17	15	19	12	14	18	11	6
64		135	31	19	23	17	18	25	16	8
256		266	61	26	24	24	25	27	25	13
16	4	37	12	17	18	11	16	16	11	9
	8	73	17	15	19	12	14	18	11	6
	16	146	33	14	21	14	14	18	12	5

Table 7: The GMRES iteration count for the $p = 4$ discretization of the advection-diffusion equation using the inexact BDDC method with the Robin-Robin interface condition.

Method	Preprocessing	Iterative
Direct	LU factorize $A^{(i)}$ LU factorize $S_{\Pi\Pi}$	Solve $LU(A_{II}^{(i)})$ and $LU(A_{rr}^{(i)})$ Solve $S_{\Pi\Pi}$ Multiply $A^{(i)}$ and apply Gram-Schmidt
ILUT	$ILUT$ factorize $A_{II}^{(i)}$ and $A_{rr}^{(i)}$ LU factorize $S_{\Pi\Pi}$	Solve $ILUT(A_{II}^{(i)})$ and $ILUT(A_{rr}^{(i)})$ Solve $S_{\Pi\Pi}$ Multiply $A^{(i)}$ and apply Gram-Schmidt
ILUT-p1	$ILUT$ factorize $A^{(i)}$ and $A_{rr}^{(i)}$ LU factorize $S_{\Pi\Pi}$ Form $A_0^{(i)} = P^T A^{(i)} P$ LU factorize $A_{II,0}^{(i)}$ and $A_{rr,0}^{(i)}$	Solve $ILUT(A_{II}^{(i)})$ and $ILUT(A_{rr}^{(i)})$ Solve $LU(A_{II,0}^{(i)})$ twice and $LU(A_{rr,0}^{(i)})$ once Apply $p = 1$ restriction/prolongation three times Solve $S_{\Pi\Pi}$ and $S_{\Pi\Pi,0}$ Multiply $A^{(i)}$ and apply Gram-Schmidt

Table 8: Operations considered for the cost estimate of the BDDC algorithm