Massively Parallel Solver for the High-Order Galerkin Least-Squares Method

by

Masayuki Yano

B.S., Aerospace Engineering (2007)
Georgia Institute of Technology

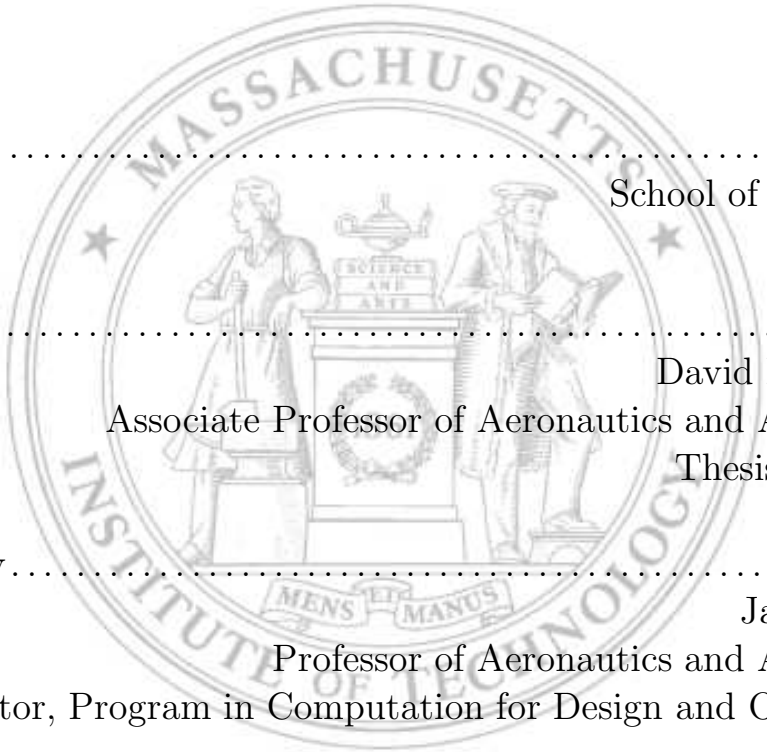
Submitted to the School of Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Computation for Design and Optimization

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2009



Author .....
School of Engineering
May, 2009

Certified by .....
David L. Darmofal
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Accepted by .....
Jaime Peraire
Professor of Aeronautics and Astronautics
Director, Program in Computation for Design and Optimization

Massively Parallel Solver for the High-Order Galerkin Least-Squares Method

by

Masayuki Yano

Submitted to the School of Engineering
on May, 2009, in partial fulfillment of the
requirements for the degree of
Master of Science in Computation for Design and Optimization

Abstract

A high-order Galerkin Least-Squares (GLS) finite element discretization is combined with massively parallel implicit solvers. The stabilization parameter of the GLS discretization is modified to improve the resolution characteristics and the condition number for the high-order interpolation. The Balancing Domain Decomposition by Constraints (BDDC) algorithm is applied to the linear systems arising from the two-dimensional, high-order discretization of the Poisson equation, the advection-diffusion equation, and the Euler equation. The Robin-Robin interface condition is extended to the Euler equation using the entropy-symmetrized variables. The BDDC method maintains scalability for the high-order discretization for the diffusion-dominated flows. The Robin-Robin interface condition improves the performance of the method significantly for the advection-diffusion equation and the Euler equation. The BDDC method based on the inexact local solvers with incomplete factorization maintains the scalability of the exact counterpart with a proper reordering.

Thesis Supervisor: David L. Darmofal
Title: Associate Professor of Aeronautics and Astronautics

Acknowledgments

I would like to thank all those who made this thesis possible. First, I would like to thank my advisor, Professor David Darmofal, for his guidance and encouragement throughout this research and for giving me the opportunity to work with him. I look forward to continue on our work together. I would also like to thank the Project X team (Julie Andren, Garret Barter, Laslo Diosady, Krzysztof Fidkowski, Bob Haimes, Josh Krakos, Eric Liu, JM Modisette, Todd Oliver, and Huafei Sun) for their support during development of the Galerkin Least-Squares code used in this work and numerous insightful discussions on high-order methods and linear solver strategies. Special thanks go to Laslo Diosady, Xun Huan, JM Modisette, and Huafei Sun for the help during the drafting of this thesis and Thomas Richter for helping me get started in the lab. I would also like to thank everyone at ACDL for making the last two years a lot of fun.

Finally, I would like to thank my family—Mom, Dad, and Hiro—for all their support, without which I would not have gotten this far.

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

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

As Computational Fluid Dynamics (CFD) has matured significantly over the past decades, the complexity of the problems that can be simulated has also increased dramatically. Driven by the desire for higher fidelity simulations, the model equations have evolved from the potential equation, to the Euler equations, and to the Navier-Stokes equations with turbulence models, e.g. Reynolds-Average Navier Stokes (RANS) and Large Eddy Simulations (LES). The geometry of the problems has also become increasingly complex, ranging from airfoils to full aircraft configurations. The evolution of the CFD capability has been realized through both algorithmic development and increased computational power.

However, there remains a number of challenging problems that are beyond the current CFD capabilities. In [61], Mavriplis lists some Grand Challenges in the aerospace community, including: a complete flight-envelope characterization, full engine simulations, and probabilistic computational optimization. Mavriplis points out that the biggest impediment to solving these problems is not the hardware capability, which has been increasing exponentially, but rather the lack of a robust, high-fidelity solver that can take advantage of massively parallel architectures that will deliver the computational power needed. In fact, today's most powerful computers house more than

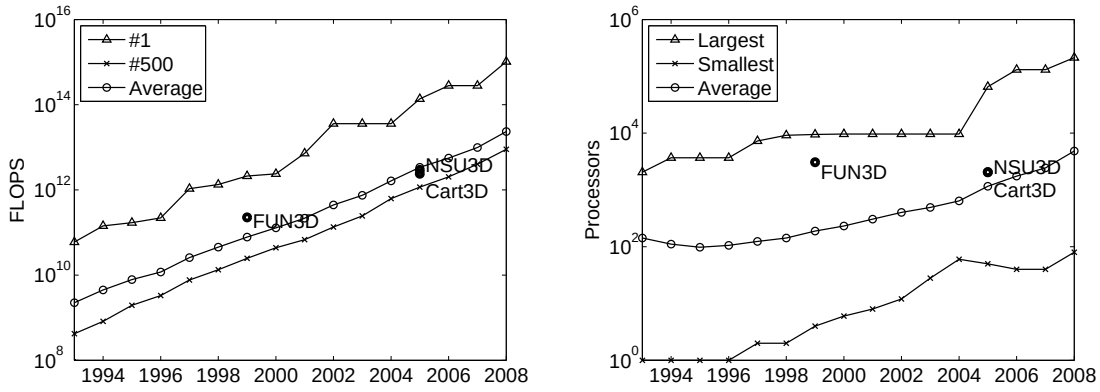


Figure 1-1: The trend of the Top 500 computers in the past 15 years [62] and the large scale computation in the aerospace community [5, 60].

100,000 processors, and the trend of massive parallelization is expected to continue (see Figure 1-1).

The difficulty in high-fidelity, efficient CFD simulations arise from the large range of temporal and spatial scales present in the flow structures; the scale of turbulence structures and the aircraft body can easily vary by more than six orders of magnitude. Thus, the discretization must be capable of efficiently capturing the widest range of scales and, given the geometric complexity, handle unstructured meshes. To meet the requirements, the work presented in this thesis employs the Galerkin Least-Squares method, which enables arbitrarily high-order accurate discretization on unstructured meshes.

Furthermore, the stiff problem, which results from the wide range of scales present, necessitates the use of an implicit method for a robust simulation at a reasonable cost. The solver must also be highly scalable to take advantage of the massively parallel computers. To address these problems, the Balancing Domain Decomposition by Constraints, which was initially developed for large-scale structural dynamics problems, is adopted for a system of conservation laws and employed to solve the linear systems arising from the high-order discretization of advection-dominated flows.

1.2 Background

1.2.1 Stabilized Finite Element Methods

Stabilized finite element methods have been developed extensively for hyperbolic and parabolic conservation laws, including the Euler equations and the Navier-Stokes equations. These methods provide consistent, locally conservative [42, 78], and arbitrarily high-order accurate discretization on unstructured meshes. The original stabilized method, the Streamline-Upwind Petrov-Galerkin (SUPG) method, was developed to provide upwinding effect in finite element methods using the Petrov-Galerkin framework [21, 47]. The method provides improved numerical stability for advection-dominated flows while maintaining consistency. The convergence analysis of the SUPG method applied to the advection-diffusion equation was elaborated in [44]. The method was extended for systems of hyperbolic equations using the generalized streamline operator, and applied to Euler equations [41, 38]. The symmetrization theory for hyperbolic conservation laws played a key role in extending the method to systems of equations [32, 37]. At the same time, the nonlinear operators for shock capturing were designed for scalar equations and systems of equations [40, 39, 45].

The SUPG method was generalized to the Galerkin Least-Squares (GLS) method, which provided a general framework for improving the stability of the classical Galerkin method using the least-squares operator [36]. The GLS method is equivalent to SUPG in the hyperbolic limit but is conceptually simpler in the presence of diffusion. Later, generalized framework for analyzing the stabilized finite element methods, including those based on the bubble functions, were provided by the variational multiscale concept [20, 19, 34, 35]. In the variational multiscale framework, the least-square operator in the GLS method is viewed as a model for the dissipation present in the subgrid scale.

1.2.2 Domain Decomposition Methods

Massively parallel solvers that are most relevant to the current work are non-overlapping domain decomposition methods, known as iterative substructuring methods. These methods were developed to solve symmetric, positive-definite linear systems arising from finite element discretization of elliptic systems in parallel environments [76, 13, 14, 15, 16]. The original substructuring method was the Neumann-Neuamnn method proposed in [12]. The Balancing Domain Decomposition (BDD) method, introduced in [53], significantly improved the scalability by introducing a coarse space correction, which made the condition number of the preconditioned operator independent of the number of subdomains. The BDD method was further modified to accomodate problems with large jumps in the coefficient across the subdomain interfaces [54, 27], making the method capable of handling larger classes of structural dynamics problems.

The BDD method further evolved into the Balancing Domain Decomposition by Constraints (BDDC), in which the coarse, global component is constructed from a set of selected primal constraints [25]. The convergence theory of the BDDC method was developed in [55], and the BDDC preconditioned operator is proved to have a condition number that is independent of the number of subdomains. Meanwhile, the BDDC method and the dual-primal Finite Element Tearing and Interconnecting (FETI-DP) method [28] have been shown to have the same set of eigenvalues except possibly those equal to 0 or 1, assuming the same set of the primal constraints are employed [56, 51, 18]. The use of inexact solvers for the BDDC method has been considered recently in [52, 26]. In these work, the subdomain problems or the partially assembled system is solved using an incomplete factorization or multigrid. The BDDC method for spectral elements using the Gauss-Lobatto-Legendre quadrature nodes has also appeared recently in [48].

Although the iterative substructuring methods were originally designed for symmetric, positive-definite systems, the methods have been applied to the advection-diffusion equation to a lesser extent. In [1], the typical Neumann-Neumann interface condition of the elliptic problem is replaced with the Robin-Robin interface condition

to maintain the positivity of the local bilinear form. The interface condition has also been applied in the FETI [75] and BDDC [77] frameworks to solve the advection-diffusion equation.

The iterative substructing methods have been implemented and tested in the production level code. In particular, a group at Sandia National Laboratory has run FETI and FETI-DP algorithms on ASCI-Red and ASCI-White with more than 1,000 processors to solve large-scale structural dynamics problems [10, 11, 70]. Their largest case includes the real-world structural analysis with more than 100 million degrees of freedom on 3,375 processor ASCI-White.

1.2.3 Massively Parallel Solvers in Aerospace Applications

The aerospace community has also been active in designing massively parallel solvers. In 1999, the finite volume Navier-Stokes solver, FUN3D [4], was ported to the ASCI-Red machine with 3,072 dual-processor nodes [5]. The code used matrix-free Newton-Krylov method with additive Schwarz preconditioner (i.e. subdomain-wise block Jacobi). The finite volume Euler equations solver, Cart3D [2], has also been used to simulate the flow over the Space Shuttle Launch Vehicle recently on the NASA Columbia supercomputer using 2,016 processors [60]. The code employs the multigrid-accelerated Runge-Kutta method to reach steady state. In the same study, the RANS equations solver, NSU3D, was used to simulate full aircraft configurations. The NSU3D solver uses a multigrid-accelerated explicit method, with implicit line smoothing in boundary layers. [57, 59]. While the explicit solvers used in Cart3D and NSU3D achieve high-parallel efficiency, these methods are not as robust or efficient as fully implicit solver for stiff problems.

1.3 Outline of Thesis

This thesis is organized as follows. The GLS discretization of conservation laws is presented in Chapter 2. The BDDC algorithm and the Robin-Robin interface condition

for a system of nonlinear equations are developed in Chapter 3. The BDDC algorithm that uses an inexact local solvers and the choice of the local solvers are discussed in Chapter 4. The numerical results are presented in Chapter 5, where the quality of the high-order GLS discretization and the performance fo the BDDC algorithm are evaluated for the Poisson equation, the advection-diffusion equation, and the Euler equations. The performance of the BDDC algorithm using the inexact factorization is also assessed.

Chapter 2

The Galerkin Least-Squares Method

This chapter develops the Galerkin Least-Squares discretization for a system of conservation laws. Particular attention is paid to a design of the stabilization parameter, τ, for a high-order discretization. The linear system arising from the high-order continuous Galerkin discretization is also discussed.

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 as

$$u_{k,t} + (F_{ik}^{\text{inv}})_{,x_i} - (F_{ik}^{\text{vis}})_{,x_i} = f_k, \quad \text{in } \Omega \tag{2.1}$$

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 inviscid flux $F^{\text{inv}} = F^{\text{inv}}(u, x, t)$, viscous flux $F^{\text{vis}} = F^{\text{vis}}(u, \nabla u, x, t)$, and the source term, $f(x, t)$, characterize the governing equations to

be solved. The quasi-linear form of the governing equation is given by

$$\mathcal{L}u \equiv u_{k,t} + A_{ikl}u_{l,x_i} - (K_{ijkl}u_{l,x_j})_{,x_i} = f_k, \tag{2.2}$$

where the inviscid flux Jacobian and viscous flux tensor are defined to satisfy

$$A_{ikl} = \frac{\partial F_{ik}^{\text{inv}}}{\partial u_l} \quad \text{and} \quad K_{ijkl}u_{l,x_j} = F_{ik}^{\text{vis}}. \tag{2.3}$$

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

$$V_h = \{u \in [H^1(\Omega)]^m : u|_K \in [\mathcal{P}_p(K)]^m, \forall K \in \mathcal{T}_h\} \tag{2.4}$$

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

$$(u_{k,t}, v_k)_\Omega + \mathcal{R}_{gal}(u, v) = 0 \quad \forall v \in V_h,$$

where

$$R_{gal}(u, v) = -(F_{ik}^{\text{inv}}, v_{k,x_i})_\Omega + (F_{ik}^{\text{vis}}, v_{k,x_i})_\Omega - (f_k, v_k)_\Omega + (\hat{F}_k(u, \text{B.C.data}, n), v_k)_{\partial\Omega},$$

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. It is well known that the standard Galerkin method becomes unstable for a large grid Peclet number, P_e, and exhibits spurious oscillations in the vicinity of unresolved internal and boundary layers. The Galerkin Least-Squares (GLS) method remedies this problem by directly controlling

the strong form of the residual. The GLS problem consists of finding $u \in V_h$ such that

$$(u_{k,t}, v_k)_\Omega + \mathcal{R}_{gal}(u, v) + \mathcal{R}_{ls}(u, v) = 0 \quad \forall v \in V_h, \tag{2.5}$$

where the least-squares residual is given by

$$\mathcal{R}_{ls}(u, v) = ((\mathcal{L}v)_l, \tau_{lk}(\mathcal{L}u - f)_k)_{\Omega, \mathcal{T}_h},$$

where $\mathcal{L}$ is the linear differential operator defined in Eq. (2.2), and τ is the stabilization parameter. The choice of stabilization parameter is discussed in detail in Section 2.2. $(\cdot, \cdot)_{\Omega, \mathcal{T}_h}$ denotes the summation of the element-wise L^2 inner product, $\sum_K (\cdot, \cdot)_K$. In the limit of $F^{\text{vis}} \to 0$, the stabilization term, $R_{ls}(\cdot, \cdot)$, adds viscosity only in the streamwise direction, and the scheme is equivalent to the Streamline Upwind Petrov-Galerkin method [36]. It is important to note that the discretization is consistent in the sense that the true solution satisfies the discrete equations.

2.1.1 Advection-Diffusion Equation

The advection-diffusion equation is characterized by inviscid and viscous fluxes 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. For simplicity, assume the Dirichlet boundary condition is imposed everywhere on $\partial\Omega$. In order to impose the boundary condition strongly, the finite element spaces for the trial function $u \in V_h$ and the test function $v \in V_{h,0}$ are chosen as

$$V_h = \{u \in H^1(\Omega) : u|_K \in \mathcal{P}_p(K), \forall K \in \mathcal{T}_h, u|_{\partial\Omega} = g\}$$
$$V_{h,0} = \{u \in H^1(\Omega) : u|_K \in \mathcal{P}_p(K), \forall K \in \mathcal{T}_h, u|_{\partial\Omega} = 0\},$$

where g is the Dirichlet boundary data.

2.1.2 Euler Equations

In two dimension, the Euler equations are characterized by the conservative state u^{cons} and the inviscid flux F^{inv} given by

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

where ρ is the fluid density, v is the velocity vector, p is the pressure, and E is the specific stagnation internal energy. The 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. Then, the time-dependent conservation law is stated as

$$u^{\text{cons}}_{,t} + (F_i^{\text{inv}})_{,x_i} = 0. \tag{2.6}$$

Alternatively, the Euler equations can be formulated using the entropy-symmetrized variables. The two sets of entropy variables commonly used for the Euler equations are Harten's variables [32] and Hughes' variables [37]. When applied to the Navier-Stokes equations, Hughes' variables have the advantage that they also symmetrize the viscosity tensor. The entropy variables used in this study are the scaled version of Hughes' variables used in [8], i.e.

$$u^{\text{entropy}} = \begin{pmatrix} \frac{-s+\gamma+1}{\gamma-1} - \frac{\rho E}{p} \\ \frac{\rho v_1}{p} \\ \frac{\rho v_2}{p} \\ -\frac{\rho}{p} \end{pmatrix},$$

where the entropy s is given by $s = \log(p/\rho^\gamma)$. The transformation matrix from the entropy variable to the conservative variables, A_0, the conservative flux Jacobian, $\hat{A}_i$, and the entropy flux Jacobian, A_i, are defined as

$$A_0 = \frac{\partial u^{\text{cons}}}{\partial u^{\text{entropy}}}, \quad \hat{A}_i = \frac{\partial F_i^{\text{inv}}}{\partial u^{\text{cons}}} \quad \text{and} \quad A_i = \hat{A}_i A_0$$

Note that A_0 is symmetric positive-definite and A_i is symmetric [37].

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 [34, 35]. It is well known [33, 41, 40] that, in order to attain optimal convergence rate 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, $\|\beta_i u_{,x_i}\|_{L^2(\Omega)}$, in advection-dominated cases. 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 equa-

tions [38]. Generalization of the inviscid stabilization parameter to a multidimensional system of equations follows from [8]. 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 is given by

$$A(n^i) = n^i_j A_j,$$

where A_j is the flux Jacobian in the x_j coordinate direction. The inviscid stabilization parameter is defined as

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

where $|A(\hat{n}^i)|$ is the matrix absolute value of the flux Jacobian evaluated in the $\hat{n}^i$ direction. Note, the directional flux Jacobian, $A(\hat{n})$, 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 and its p-dependence is discussed in Section 2.2.1.

2.2.1 High-Order Correction of τ

The proof of the optimal convergence of the GLS method in the diffusion-dominated cases requires a choice of τ that satisfies

$$\kappa\tau\|\Delta(u-u_h)\|^2_{\Omega,\mathcal{T}_h} \leq C\|\nabla(u-u_h)\|^2_{\Omega}, \tag{2.7}$$

where C is a parameter independent of h [36, 30]. From an inverse estimate, there exists c such that

$$h_e^2\|\Delta v\|^2_K \leq c\|\nabla v\|^2_K, \quad \forall v \in V, \tag{2.8}$$

where h_e is a characteristic length of the element. By the inverse estimate, Eq. (2.7) is satisfied asymptotically as long as $\tau = \mathcal{O}(h_e^2)$. However, in practice, h_e is far from zero, and the choice of the scale of the stabilization parameter affects the solution quality. In a practical setting, two questions of interests are the choice of h_e for an anisotropic element and the scaling of c with the interpolation order.

The inverse estimate for a quadratic and cubic interpolation on a triangular element has been studied in detail in [31]. The analysis shows that the appropriate element size, h_e, in Eq. (2.8) scales with the length of the shortest edge for an anisotropic, straight-edged element. Similar results for the GLS methods on highly anisotropic mesh are reported in [6, 63].

The inverse estimate for a spectral-like discretization is introduced in [7], and c in Eq (2.8) is shown to scale with p^4 asymptotically. However, as the interpolation orders considered ($p = 1, \ldots, 5$) are relatively low, the use of the asymptotic behavior

for the design of τ is not appropriate. In [31], the choice of c for $p = 2$ and $p = 3$ interpolations is calculated by analytically solving a Rayleigh quotient problem,

$$\lambda = \max_{v \in \mathcal{P}^p(K)} \frac{\|\Delta v\|_K^2}{\|\nabla v\|_K^2},$$

and setting $c = \lambda h_e^2$. However, this approach becomes difficult for $p > 3$. On the other hand, an explicit calculation of c for each element is alluded to in [30], but not pursued. Note, an explicit calculation would make the discretization nonlinear even when it is applied to a linear equation.

The proposed approach is a simplification of the approach taken in [31]. Recall, the role of τ is to recover the dissipation that would have been provided by the subgrid, unresolved features. As pointed out in [35], p-refinement generates approximate subgrid scale Green's function, and some of the dissipation effects that the subgrid scale has on the resolved scale are captured. Thus, the proposed approach is obtained by rescaling the viscous scaling length based on the interpolation order.

Consider the Poisson equation in one dimension,

$$\mathcal{L}u \equiv -\kappa u_{,xx} = f, \qquad x \in \Omega \subset \mathbb{R}^1.$$

For simplicity, assume Dirichlet boundary condition with $u|_{\partial\Omega} = 0$ is enforced. The GLS discretization of the problem is to find $u \in H_0^1(\Omega)$ such that

$$(\kappa v_{,x}, u_{,x})_\Omega + (\mathcal{L}v, \tau(\mathcal{L}u - f))_{\Omega, \mathcal{T}_h} = 0 \qquad \forall v \in H_0^1(\Omega).$$

Assuming $u \in H^4(\Omega)$, applying integration by parts to the stabilization term twice reveals the modified equation arising from the GLS discretization of the problem, i.e.

$$-\kappa u_{,xx} + \tau\kappa^2 u_{,xxxx} = f, \qquad x \in \Omega.$$

Applying Fourier transform $x \to \eta$, the resulting equation in η domain is

$$\kappa\eta^2\hat{u}(\eta) + \tau\kappa^2\eta^4\hat{u}(\eta) = \hat{f}(\eta), \qquad \eta \in \mathbb{C}$$

and the eigenvalues of the modified equation are

$$\lambda = \kappa\eta^2 + \tau\kappa^2\eta^4.$$

For high-order discretization, assume $\max(|\eta|) \sim \frac{1}{h/p} = \frac{p}{h}$. The condition number of the problem is

$$\frac{\lambda_{\max}}{\lambda_{\min}} = \frac{\kappa\left(\frac{p}{h}\right)^2 + \tau\kappa^2\left(\frac{p}{h}\right)^4}{\kappa + \tau\kappa^2} = \frac{1 + \tau\kappa\left(\frac{p}{h}\right)^2}{1 + \tau\kappa}\left(\frac{p}{h}\right)^2. \tag{2.9}$$

In order for the condition number of the stabilized operator to behave like that of the unstabilized operator (i.e. $\tau = 0$), the first term of Eq. (2.9) must satisfy

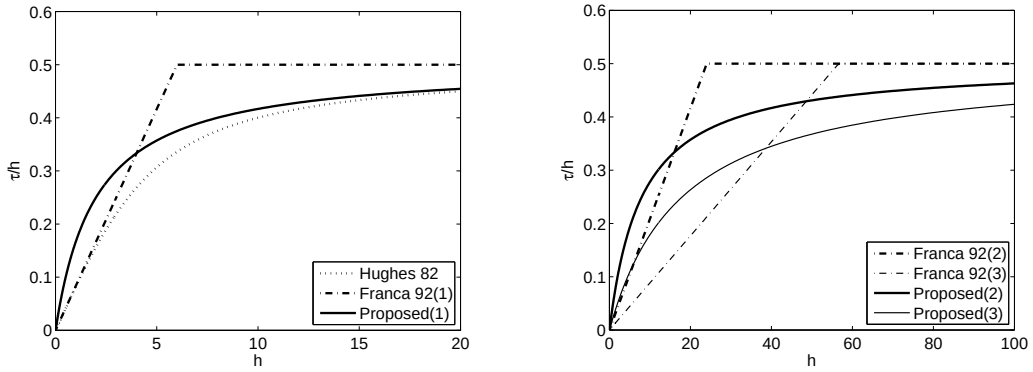
$$\frac{1 + \tau\kappa\left(\frac{p}{h}\right)^2}{1 + \tau\kappa} = \mathcal{O}(1), \quad \text{as } h \to 0$$
$$\frac{1 + \tau\kappa\left(\frac{p}{h}\right)^2}{1 + \tau\kappa} = \mathcal{O}(1), \quad \text{as } p \to \infty$$

In order to satisfy these conditions, the appropriate τ scaling in viscous limit is

$$\tau \sim \frac{1}{\kappa\eta^2} \sim \frac{h^2}{\kappa p^2}.$$

Thus, in additional to satisfying the h^2 scaling of the linear finite element, τ must scale as $1/p^2$ in diffusion dominated cases for high-order elements.

Figure 2-1 shows the comparison of the stabilization parameters for the advection-diffusion equation with $\|\beta\| = \kappa = 1$. Hughes 82 is the τ that gives the nodally exact solution for linear finite element in one dimension [21]. For higher-order interpolations, Franca 92(p) refers to the method proposed in [30] using the inverse estimate coefficient, c, derived for $p = 2$ and $p = 3$ interpolations in [31]. The proposed approach is



(a) Linear finite element

(b) $p = 2$ and $p = 3$

Figure 2-1: Comparison of the stabilization parameter for high-order discretization.

more diffusive than Franca $92(p)$ in diffusion dominated case.

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. 2.5 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 $M_h = (\phi_j, \phi_i)_\Omega$ 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 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.

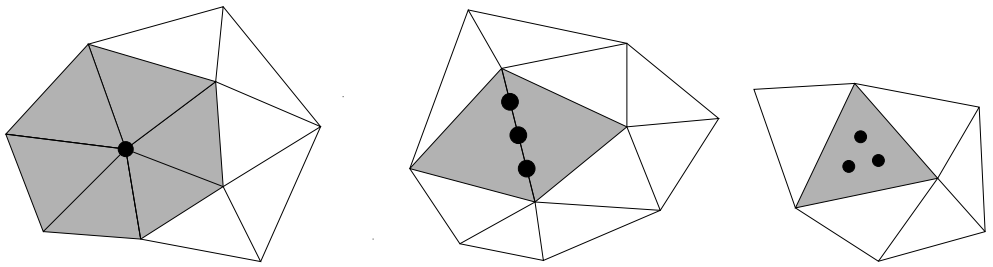
It is worth mentioning the relationship between the Jacobian matrix, $\frac{dR}{dU}$, and the stiffness matrix arising from a finite element discretization of a linear PDE. Although the boundary residual and the least-squares residual can be included readily, these terms are neglected for simplicity. The Jacobian matrix can be expressed as

$$\begin{aligned}
\frac{dR_i}{dU_j} &= \frac{d}{dU_j}\left(\mathcal{R}_{gal}(U_k\phi_k, \phi_i)\right) \\
&= \frac{d}{dU_j}\left(-(F_s^{\text{inv}}, \phi_{i,x_s})_\Omega + (F_s^{\text{vis}}, \phi_{i,x_s})_\Omega - (S, \phi_i)_\Omega\right) \\
&= -\left(A_s\phi_j, \phi_{i,x_s}\right)_\Omega + \left(\frac{\partial K_{sr}}{\partial u}\phi_j u_{,x_r} + K_{sr}\phi_{j,x_r}, \phi_{i,x_s}\right)_\Omega - \left(\frac{\partial f}{\partial u}\phi_j, \phi_i\right)_\Omega
\end{aligned}$$

where A and K are the flux Jacobian and viscous tensor defined in Eq. 2.3. Defining $\bar{A}_s = A_s + \frac{\partial K_{sr}}{\partial u}u_{,x_r}$ and $C = \frac{\partial f}{\partial u}$, the linearized equations solved in each Newton step is equivalent to a linear system arising from a discretization of advection-diffusion-reaction equation, i.e.

$$\mathcal{R}^{\text{lin}}(u, v) = -\left(\bar{A}_s u, v_{,x_s}\right)_\Omega + \left(K_{sr}u_{i,x_r}, v_{i,x_s}\right)_\Omega - \left(C_k u, v\right)_\Omega .$$

For this reason, the term Jacobian matrix and the stiffness matrix are used interchangeably in the following sections. Similarly, the linear solver algorithm is described using the bilinear form, $a(\cdot,\cdot)$, instead of the linearized form $\mathcal{R}^{\text{lin}}(\cdot,\cdot)$, to be consistent with other literatures on domain decomposition methods.



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

Figure 2-2: Diagram of basis modes and their support. The degree of freedom for each mode for $p = 4$ discretization is shown in dots.

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. For an even higher order interpolation, it would be necessary to use a node distribution clustered in the extremities to improve the numerical conditioning, e.g. Fekete points [74].

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 $T : \xi \rightarrow x$ 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 2-2. 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. Except on the boundary, the node mode has the largest support out of all basis types. The second type of basis is the edge mode, which has

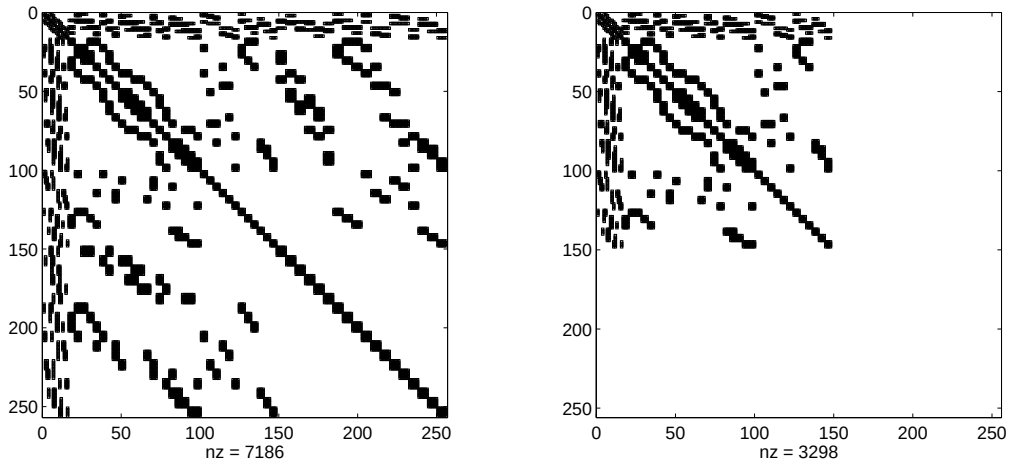
p	1	2	3	4	5
Elemental basis	3	6	10	15	21
DOF/elem.	0.5 (0.5)	2.0 (2.0)	4.5 (3.5)	8.0 (5.0)	12.5 (6.5)
Matrix nnz/elem.	3.5 (3.5)	23.0 (23.0)	76.5 (66.5)	188.0 (143.0)	387.5 (261.5)

Table 2.1: The degree of freedom associated with high-order continuous Galerkin discretization on an average mesh.

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 $\frac{1}{2}(p-2)(p-1)$. As the element mode has elementally compact support, this mode can be eliminated locally prior to solving the global linear system via static condensation.

The ratio of the number of global modes and the number of elements is shown in Table 2.1. The number of nonzero entries in the stiffness matrix per element is also shown. The number in the parenthesis are based on the global system after the static condensation of the elemental modes. The calculation is based on an average mesh with six elements sharing a node. For a high-order discretization, the elemental static condensation significantly reduces the size of the global vector and the size of the stiffness matrix. The static condensation not only leads to a smaller preconditioner size, but also reduces the size of Krylov space vectors for GMRES.

The comparison of the original system and the system after the elemental static condensation is shown in Figure 2-3. For the original system, the node modes are ordered first, followed by the edge modes, and then the element modes. The static condensation removes the elemental modes, but does not alter the sparsity pattern of the remaining matrix.



(a) Original system

(b) Statically condensed system

Figure 2-3: Sparsity pattern of the stiffness matrix arising from $p = 5$ discretization on a 3×3 structured mesh with 18 triangular elements.

Chapter 3

Balancing Domain Decomposition by Constraints

This chapter develops the Balancing Domain Decomposition by Constraints (BDDC) method. The generalization of the Robin-Robin interface condition to a system of symmetrized hyperbolic equations is also discussed.

3.1 Schur Complement System

This section presents the algebraic formulation of the Schur complement system. Throughout the section, a suitable set of basis functions representing the finite element spaces is assumed to have been chosen, and no distinction is made between the finite element spaces and the spaces of coefficients for the basis, e.g. $V_h(\Omega) \subset H^1(\Omega)$ and $V_h = \mathbb{R}^{\mathrm{dof}(V_h(\Omega))}$.

Let $\{\Omega_i\}_{i=1}^N$ be a decomposition of domain Ω into N non-overlapping subdomains such that

$$\Omega_i \cap \Omega_j = \emptyset \quad i \neq j$$
$$\bar{\Omega} = \cup_{i=1}^N \bar{\Omega}_i.$$

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 space of traces on Γ_i of the functions in $V_h^{(i)}$ 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

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

Let $R^{(i)} : V_h \to V_h^{(i)}$ be the restriction operator that extracts the degrees of freedom associated with $V_h^{(i)}$. 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 discrete harmonic extension, $\mathcal{H}_i : V_\Gamma^{(i)} \to V_h^{(i)}$, and the adjoint discrete harmonic extension, $\mathcal{H}_i^* : V_\Gamma^{(i)} \to V_h^{(i)}$, correspond to extending the interface values to the interior such that the interior residual vanishes with respect to $a_i(\cdot,\cdot) : V_h^{(i)} \times V_h^{(i)} \to \mathbb{R}$, i.e.

$$\begin{aligned} a_i(\mathcal{H}_i u_\Gamma^{(i)}, v_I^{(i)}) = 0, \quad \forall u_\Gamma^{(i)} \in V_\Gamma^{(i)}, \forall v_I^{(i)} \in V_I^{(i)} \\ a_i(w_I^{(i)}, \mathcal{H}_i^* \psi_\Gamma^{(i)}) = 0, \quad \forall \psi_\Gamma^{(i)} \in V_\Gamma^{(i)}, \forall w_I^{(i)} \in V_I^{(i)}. \end{aligned}$$

The equivalent operators in matrix form are given by

$$\mathcal{H}_i = \begin{pmatrix} -(A_{II}^{(i)})^{-1} A_{I\Gamma}^{(i)} \\ I_\Gamma^{(i)} \end{pmatrix} \quad \text{and} \quad \mathcal{H}_i^* = \begin{pmatrix} -(A_{II}^{(i)})^{-T} (A_{\Gamma I}^{(i)})^T \\ I_\Gamma^{(i)} \end{pmatrix}.$$

For elliptic problems, the discrete harmonic extension is the minimum energy extension of the interface variables to the interior [76]. The local Schur complement operator is given by restricting the local bilinear form to the space of discrete harmonic extensions, i.e.

$$\begin{aligned} S^{(i)} &= (\mathcal{H}_i^*)^T A^{(i)} \mathcal{H}_i = A_{\Gamma\Gamma}^{(i)} - A_{\Gamma I}^{(i)} (A_{II}^{(i)})^{-1} A_{I\Gamma}^{(i)} \\ g^{(i)} &= (\mathcal{H}_i^*)^T f^{(i)} = f_\Gamma^{(i)} - A_{\Gamma I}^{(i)} (A_{II}^{(i)})^{-1} f_I^{(i)}. \end{aligned} \tag{3.1}$$

The global Schur complement operator is defined as the assembly of the local Schur complement operators. Let the operator $R_\Gamma^{(i)} : V_\Gamma \to V_\Gamma^{(i)}$ be the restriction of the global interface variables, V_Γ, to the local interface variables, $V_\Gamma^{(i)}$. The global Schur complement matrix, $S : V_\Gamma \to V_\Gamma$, and the vector, $g \in V_\Gamma$, are

$$S = \sum_{i=1}^{N} (R_\Gamma^{(i)})^T S^{(i)} R_\Gamma^{(i)} \quad \text{and} \quad g = \sum_{i=1}^{N} (R_\Gamma^{(i)})^T g^{(i)},$$

and the Schur complement problem is

$$S u_\Gamma = g.$$

In practice, the Schur complement system is not formed explicitly. Instead, the action of S on $v \in V_\Gamma$ is computed when necessary by solving the local Dirichlet problems defined in Eq. (3.1) in parallel and by assembling the resultant vectors. Thus, a preconditioned GMRES [71, 73], which only requires the action of a matrix onto a vector, is used to solve the Schur complement problem.

3.2 BDDC Preconditioner

In order to define the BDDC preconditioner, 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. The local

dual variables, $V_\Delta^{(i)}$, consists of functions in $V_\Gamma^{(i)}$ that vanish on the primal degrees of freedom. The space $V_\Pi^{(i)}$ is chosen such that the local Schur complement system restricted to $V_\Delta^{(i)}$ is invertible. The partially assembled space is defined as

$$\tilde{V}_\Gamma = \left(\oplus_{i=1}^N V_\Delta^{(i)}\right) \oplus \tilde{V}_\Pi, \tag{3.2}$$

where $\tilde{V}_\Pi$ is a coarse primal variable space. The functions in $\tilde{V}_\Pi$ are continuous across the subdomain interfaces. 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. Notice, because $\tilde{V}_\Pi$ is continuous across the interface and the functions in $V_\Delta^{(i)}$ vanish at the primal degrees of freedom, the partially assembled space $\tilde{V}_\Gamma$ is continuous at the primal degrees of freedom. However, the function in $\tilde{V}_\Gamma$ are discontinuous elsewhere as $\oplus_{i=1}^N V_\Delta^{(i)}$ is discontinuous across the interface.

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 weighted average of the interface variables must be defined [25]. 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 [50]. The scaling factors, $\delta_i^\dagger$, are collected to form a diagonal matrix, $D^{(i)}$, which operates on $V_\Gamma^{(i)}$.

Assume 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 [49, 51], i.e. $v = (v_I, v_\Delta, v_\Pi)^T \in V_I^{(i)} \oplus V_\Delta^{(i)} \oplus V_\Pi^{(i)}$. After the change of basis, the local stiffness matrix can be written as

$$A^{(i)} = \begin{pmatrix} A_{rr}^{(i)} & A_{r\Pi}^{(i)} \\ A_{\Pi r}^{(i)} & A_{\Pi\Pi}^{(i)} \end{pmatrix},$$

where

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

Let $R_{\Delta\Gamma} : V_{\Gamma}^{(i)} \to V_{\Delta}^{(i)}$ be the restriction operator that extracts the dual degrees of freedom on the interface. The local dual Schur complement operator is given by

$$S_{\Delta}^{(i)} = A_{\Delta\Delta}^{(i)} - A_{\Delta I}^{(i)}(A_{II}^{(i)})^{-1}A_{I\Delta}^{(i)}. \tag{3.3}$$

Similar to the Schur complement, the matrix $S_{\Delta}^{(i)}$ is not formed explicitly. Instead, the application of its inverse, $(S_{\Delta}^{(i)})^{-1}$, to a vector $w_{\Delta}^{(i)} \in V_{\Delta}^{(i)}$ is computed by solving a Neumann problem with a few additional constraints, i.e. $v_{\Delta}^{(i)} = (S_{\Delta}^{(i)})^{-1}w_{\Delta}^{(i)}$ is given by solving

$$\begin{pmatrix} A_{II}^{(i)} & A_{I\Delta}^{(i)} \\ A_{\Delta I}^{(i)} & A_{\Delta\Delta}^{(i)} \end{pmatrix} \begin{pmatrix} v_{I}^{(i)} \\ v_{\Delta}^{(i)} \end{pmatrix} = \begin{pmatrix} 0 \\ w_{\Delta}^{(i)} \end{pmatrix}. \tag{3.4}$$

A sufficient number of coarse constraints is assumed to have been chosen such that this constrained Neumann problem is well-posed.

Let $R_{\Pi}^{(i)} : V_{\Pi} \to V_{\Pi}^{(i)}$ be the restriction operator that extracts the local primal degrees of freedom from the global primal degrees of freedom. The local coarse basis, $\Psi^{(i)} : V_{\Pi}^{(i)} \to V_{h}^{(i)}$, and the adjoint coarse basis, $\Psi^{*(i)} : V_{\Pi}^{(i)} \to V_{h}^{(i)}$, take a value of 1 at a primal degree of freedom and extends to interior and dual degrees of freedom such that residual against them are zero, i.e.

$$\begin{aligned} a_i(\Psi^{(i)}u_{\Pi}^{(i)}, v) = 0 \quad & \forall u^{(i)} \in V_{\Pi}^{(i)}, \forall v \in V_{I}^{(i)} \oplus V_{\Delta}^{(i)} \\ a_i(w, \Psi^{*(i)}\psi^{(i)}) = 0 \quad & \forall \psi^{(i)} \in V_{\Pi}^{(i)}, \forall w \in V_{I}^{(i)} \oplus V_{\Delta}^{(i)}. \end{aligned}$$

In matrix form, these problems are equivalent to finding $\Psi^{(i)} = ((\Psi_{I}^{(i)})^T, (\Psi_{\Delta}^{(i)})^T, I_{\Pi}^T)^T$

and $\Psi^{*(i)} = ((\Psi_I^{*(i)})^T, (\Psi_\Delta^{*(i)})^T, I_\Pi^T)$, each in $\mathbb{R}^{\dim(V_h^{(i)}) \times \dim(V_\Pi^{(i)})}$, such that

$$\begin{pmatrix} A_{II}^{(i)} & A_{I\Delta}^{(i)} \\ A_{\Delta I}^{(i)} & A_{\Delta\Delta}^{(i)} \end{pmatrix} \begin{pmatrix} \Psi_I^{(i)} \\ \Psi_\Delta^{(i)} \end{pmatrix} = - \begin{pmatrix} A_{I\Pi}^{(i)} \\ A_{\Delta\Pi}^{(i)} \end{pmatrix}$$

and

$$\begin{pmatrix} A_{II}^{(i)} & A_{I\Delta}^{(i)} \\ A_{\Delta I}^{(i)} & A_{\Delta\Delta}^{(i)} \end{pmatrix}^T \begin{pmatrix} \Psi_I^{*(i)} \\ \Psi_\Delta^{*(i)} \end{pmatrix} = - \begin{pmatrix} (A_{\Pi I}^{(i)})^T \\ (A_{\Pi\Delta}^{(i)})^T \end{pmatrix}.$$

For elliptic problems, the local coarse basis function is the energy minimizing extension of the primal degrees of freedom to the interior and the dual degrees of freedom [25]. In order to construct the local coarse basis function, the constrained Neumann problem, Eq. (3.4), is solved $\dim(V_\Pi^{(i)})$ times. The restriction of the local coarse basis function to the interface space is denoted by $\Psi_\Gamma^{(i)} \equiv ((\Psi_\Delta^{(i)})^T, I_\Pi^T)^T$ and $\Psi_\Gamma^{*(i)} \equiv ((\Psi_\Delta^{*(i)})^T, I_\Pi^T)^T$. The local primal Schur complement is defined as a restriction of the global Schur complement to the space of local primal variables, i.e.

$$S_\Pi^{(i)} = (\Psi_\Gamma^{*(i)})^T S^{(i)} \Psi_\Gamma^{(i)} = A_{\Pi\Pi}^{(i)} - A_{\Pi r}^{(i)} (A_{rr}^{(i)})^{-1} A_{r\Pi}^{(i)}.$$

The global primal Schur complement is the assembly of local primal Schur complements, i.e. $S_\Pi = \sum_{i=1}^N (R_\Pi^{(i)})^T S^{(i)} R_\Pi^{(i)}$. The BDDC preconditioner is given by

$$M_{\text{BDDC}}^{-1} = \sum_{i=1}^N (R_\Gamma^{(i)})^T D^{(i)} \left(T_{\text{sub}}^{(i)} + T_{\text{coarse}}^{(i)} \right) D^{(i)} R_\Gamma^{(i)},$$

where

$$T_{\text{sub}}^{(i)} = (R_{\Delta\Gamma}^{(i)})^T \left(S_\Delta^{(i)} \right)^{-1} R_{\Delta\Gamma}^{(i)}$$
$$T_{\text{coarse}}^{(i)} = (\Psi^{(i)} R_\Pi^{(i)}) S_\Pi^{-1} (\Psi^{*(i)} R_\Pi^{(i)})^T$$

Note the application of the subdomain correction, T_{sub} can be performed completely

in parallel. The coarse grid correction, S_{Π}^{-1}, is a globally coupled problem, but the degrees of freedom associated with the problem is on the same order as the number of subdomains.

Assuming the standard shape regularity conditions of substructuring methods hold [17], the BDDC preconditioned operator applied to the Poisson equation has the condition number bound

$$\kappa(M_{\text{BDDC}}^{-1}A) \leq C(1+\log(H/h))^2,$$

where the constant C is independent of the subdomain size H and the element size h [26]. Thus, the condition number is independent of the number of the subdomains and is weakly dependent on the size of the subdomains.

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 symmetrized system of conservation laws.

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

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

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 follow 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 conservation property of the global bilinear form on a subet $\check{\Omega}\subset\Omega$. 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.

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

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

$$\begin{aligned}
&\tfrac{1}{2}\|u|_{t=T}\|^2_{L^2(\Omega_i)} + \tfrac{1}{2}(u, u\beta_{j,x_j})_{\Omega_i\times I} + \tfrac{1}{2}(u, u|\beta_j n_j|)_{(\partial\Omega_i\cap\partial\Omega)\times I} \\
&\quad - \tfrac{1}{2}(u, u|\beta_j n_j|)_{\Gamma_i^+\times I} + \tfrac{1}{2}(u, u|\beta_j n_j|)_{\Gamma_i^-\times I} \\
&\qquad = \tfrac{1}{2}\|u|_{t=0}\|^2_{L^2(\Omega_i)} + (u, f)_{\Omega_i\times I} + (u, g|\beta_j n_j|)_{(\partial\Omega_i\cap\partial\Omega^-)\times I},
\end{aligned} \tag{3.5}$$

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 $\frac{1}{2}(u, u\beta_j n_j)_{\Gamma_i^+\times I}$ to the interfaces. The resulting bilinear form is

$$a_i(u,v) = \hat{a}_i(u,v) + \frac{1}{2}(u, u\beta_j n_j)_{\Gamma_i^+\times I}. \tag{3.6}$$

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 [1] 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 [75] and recently BDDC [77].

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 using the linearized form presented in Section 2.3. For a time-dependent hyperbolic system of conservation laws, the semilinear form is expressed

as

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

and the linearized equation solved at a given step of the Newton iteration is

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

where $A_0 = du^{\text{cons}}/du^{\text{entropy}}$ is the matrix mapping the entropy symmetrized variables to the conservative variables, A_j is the flux Jacobian in x_j direction, and $A_{bc} \equiv \frac{\partial \hat{F}}{\partial u}$. 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 $\|u\|_{A_0,\Omega} = (A_0u, u)_\Omega$ 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.

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

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

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

A local semilinear form that has the desired linearized form is

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

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.

Chapter 4

Inexact Solver

This chapter develops an inexact BDDC preconditioner following the approach taken in [52]. Inexact local solvers based on the dual threshold incomplete factorization [72] and the minimum discarded fill reordering [23] are also discussed.

4.1 Inexact BDDC Preconditioner

The BDDC algorithm described in Chapter 3 requires solutions to the local Dirchlet problems, Eq. (3.1), and the constrained Neumann problems, Eq. (3.4). In each GMRES step, the local Dirichlet problem is solved to obtain the action of the local Schur complement, $S^{(i)}$, on a vector $v_\Gamma^{(i)} \in V_h^{(i)}$. The action of the inverse of the local dual Schur complement, $(S_\Delta^{(i)})^{-1}$, on a vector $v_\Delta^{(i)} \in V_\Delta^{(i)}$ requires the solution to the constrained Neumann problem. Similarly, in pre-processing stage, the local coarse basis functions, $\Psi^{(i)}$, and the primal Schur complement, S_Π, are constructed by solving the constrained Neumann problems.

For a large scale problem, the cost of computing and storing the exact factorizations of the local problems can be prohibitively high, and the cost only escalate for three-dimensional problems. Thus, the inexact BDDC preconditioner is constructed by replacing the solutions to the Dirichlet and the constrained Neumann problems by action of inexact local solvers.

In order to employ inexact solvers, the BDDC formulation needs to be modified slightly. Recall, the BDDC algorithm creates a preconditioner for the Schur complement system. The preconditioned interface problem is to find $u_\Gamma \in V_\Gamma$ such that

$$M^{-1}_{\text{BDDC}} S u_\Gamma = M^{-1}_{\text{BDDC}} g.$$

However, in the absence of the exact factorizations to the local stiffness matrices, $A^{(i)}$, the action of discrete harmonic extensions cannot be computed exactly. Thus, the original right hand side, $f \in V_h$, cannot be statically condensed to form the interface right hand side, $g = f_\Gamma - A_{\Gamma I} A_{II}^{-1} f_I \in V_\Gamma$. Therefore, the preconditioner based on local inexact solvers, M^{-1}_{iBDDC}, must act on the entire finite element space, V_h, i.e. the preconditioned problem must have a form

$$M^{-1}_{\text{iBDDC}} A u = M^{-1}_{\text{iBDDC}} f,$$

where $u, f \in V_h$.

Ref. [52] shows the construction of preconditioned operator, $M^{-1}_{\text{iBDDC}} A$, that has the same spectrum as the BDDC preconditioned operator, $M^{-1}_{\text{BDDC}} S$, with the exception of some additional eigenvalues equal to 1. The preconditioner, M^{-1}_{iBDDC}, is given by

$$M^{-1}_{\text{iBDDC}} = \tilde{\mathcal{H}} \tilde{A}^{-1} (\tilde{\mathcal{H}}^*)^T. \tag{4.1}$$

The matrix $\tilde{A}$ and the extension operators $\tilde{\mathcal{H}}$ and $\tilde{\mathcal{H}}^*$ will be defined shortly. The matrix $\tilde{A}$ operates on the partially assembled space, $\tilde{V}$, defined as the product of the partially assembled interface space, $\tilde{V}_\Gamma$, and the interior degrees of freedom, i.e.

$$\tilde{V} = (\oplus_{i=1}^N V_I^{(i)}) \oplus \tilde{V}_\Gamma.$$

Let $\bar{R}$ be an injection operator from the partially assembled space, $\tilde{V}$, to the unassembled space, $\oplus_{i=1}^N V^{(i)}$. The partially assembled matrix, $\tilde{A}$, is obtained by assembling

the local stiffness matrices, $A^{(i)}$, with respect to only the primal variables, i.e.

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

where

$$A_{rr}^{(i)} \equiv \begin{pmatrix} A_{II}^{(i)} & A_{I\Delta}^{(i)} \\ A_{\Delta I}^{(i)} & A_{\Delta\Delta}^{(i)} \end{pmatrix}, \qquad \tilde{A}_{r\Pi}^{(i)} \equiv \begin{pmatrix} A_{I\Pi}^{(i)} R_{\Pi}^{(i)} \\ A_{\Delta\Pi}^{(i)} R_{\Pi}^{(i)} \end{pmatrix},$$

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

Note, due to the block structure of $\tilde{A}$, most of the operations required to calculate its inverse can be carried out in parallel. More precisely, 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 $S_{\Pi} = A_{\Pi\Pi} - A_{\Pi r} A_{rr}^{-1} A_{r\Pi}$. The majority of the cost of the inversion comes from the application of A_{rr}^{-1}, which parallelizes completely.

The extension operators $\tilde{\mathcal{H}}$ and $\tilde{\mathcal{H}}^*$ in Eq. (4.1) average the interface variables V_Γ and add the harmonic extension of the jump in the interface variables to the interior, i.e.

$$\tilde{\mathcal{H}} = \begin{pmatrix} I & A_{II}^{-1} \tilde{A}_{I\Gamma} J_{1,D,\Gamma} \\ & \tilde{R}_{D,\Gamma}^T \end{pmatrix} \quad \text{and} \quad \tilde{\mathcal{H}}^* = \begin{pmatrix} I & A_{II}^{-T} \tilde{A}_{\Gamma I}^T J_{2,D,\Gamma}^T \\ & \tilde{R}_{D,\Gamma}^T \end{pmatrix} \tag{4.2}$$

with

$$J_{1,D,\Gamma} = I - \tilde{R}_\Gamma \tilde{R}_\Gamma^T \tilde{D} \quad \text{and} \quad J_{2,D,\Gamma} = I - \tilde{D} \tilde{R}_\Gamma \tilde{R}_\Gamma^T.$$

The majority of the cost of the extension operators comes from applying A_{II}^{-1}, which can be carried out in parallel.

The preconditioned operator, $M_{\text{iBDDC}}^{-1} A$, has the same spectrum as the BDDC preconditioned operator, $M_{\text{BDCC}}^{-1} S$, when $\tilde{A}^{-1}$, $\tilde{\mathcal{H}}$, and $\tilde{\mathcal{H}}^*$ are computed exactly [52]. The inexact solver proposed for this study is obtained by replacing $\tilde{A}_{rr}^{-1}$ in $\tilde{A}^{-1}$ and A_{II}^{-1} in $\tilde{\mathcal{H}}$ and $\tilde{\mathcal{H}}^*$ with the action of inexact solvers.

4.2 Local Inexact Solver

The inexact BDDC algorithm has been previously applied to linear systems arising from the linear finite element discretization of elliptic systems [52, 26]. In [52], the solution to the partially assembled system $\tilde{A}$ and the local Dirchlet problems are replaced by the action of multigrid V-cycles, and convergence properties are proved for the Poisson equation, for which the multigrid method converges uniformly. The numerical experiment confirms that the method preserves the scaling properties of the BDDC method using the exact local solvers. However, the theory of multigrid applied to hyperbolic equations is much less established. The issue is further complicated by the presence of the mesh dependent stabilization parameter in the GLS discretization, which results in an unstable coarse system if the least-squares operator is not scaled properly [65, 64]. The p-multigrid strategy suffers from a similar problem, as the modified stabilization parameter is a function of the interpolation order [24]. While the multigrid methods have been applied successfully to systems of conservation laws in aerospace applications (e.g. [43, 69, 58]), the use of the method is not pursued in this study.

Incomplete factorizations, with a proper reordering, have been shown to work well for advection-dominated problems [67, 24]. In order for the incomplete factorization to

perform well, the reordering needs to ensure that the factorization captures the strong couplings among the elements. Example of the succesful reordering for hyperbolic systems includes the line creation algorithm motivated by the physics of the problem [65, 29] and the algebraic algorithms based on minimizing the magnitude of the discarded fills [23, 67]. Thus, an incomplete factorization, with a suitable reordering, is used as the inexact solver for the BDDC algorithm in this study.

4.2.1 Incomplete Factorization

BDDC using Incomplete Factorizations

Two different strategies, called the two-matrix method and the one-matrix method, are considered for the BDDC preconditioner using the inexact local solvers. Recall, the inexact solver is used to solve the local Dirichlet problem for the discrete harmonic extensions and to solve the constrained Neumann problem for the partially assembled system. The two-matrix method simply stores the incomplete factorizations for the Dirichlet problem and the constrained Neumann problem separately, i.e.

$$
\begin{aligned}
P_D^{(i)} A^{(i)} (P_D^{(i)})^T &= \begin{pmatrix} P_{II}^{(i)} & \\ & I_\Gamma^{(i)} \end{pmatrix} \begin{pmatrix} A_{II}^{(i)} & A_{I\Gamma}^{(i)} \\ A_{\Gamma I}^{(i)} & A_{\Gamma\Gamma}^{(i)} \end{pmatrix} \begin{pmatrix} (P_{II}^{(i)})^T & \\ & I_\Gamma^{(i)} \end{pmatrix} \\
&\approx \begin{pmatrix} L_{A_{II}^{(i)}} & \\ A_{\Gamma I}^{(i)} U^{-1}_{A_{II}^{(i)}} & I_\Gamma^{(i)} \end{pmatrix} \begin{pmatrix} U_{A_{II}^{(i)}} & L^{-1}_{A_{II}^{(i)}} A_{I\Gamma}^{(i)} \\ & S_\Gamma^{(i)} \end{pmatrix}
\end{aligned}
$$

and

$$
\begin{aligned}
P_N^{(i)} A^{(i)} (P_N^{(i)})^T &= \begin{pmatrix} P_{rr}^{(i)} & \\ & I_\Pi^{(i)} \end{pmatrix} \begin{pmatrix} A_{rr}^{(i)} & A_{r\Pi}^{(i)} \\ A_{\Pi r}^{(i)} & A_{\Pi\Pi}^{(i)} \end{pmatrix} \begin{pmatrix} (P_{rr}^{(i)})^T & \\ & I_\Pi^{(i)} \end{pmatrix} \\
&\approx \begin{pmatrix} L_{A_{rr}^{(i)}} & \\ A_{\Pi r}^{(i)} U^{-1}_{A_{rr}^{(i)}} & I_\Pi^{(i)} \end{pmatrix} \begin{pmatrix} U_{A_{rr}^{(i)}} & L^{-1}_{A_{rr}^{(i)}} A_{r\Pi}^{(i)} \\ & S_\Pi^{(i)} \end{pmatrix}
\end{aligned}
$$

The reordering provided by $P_{II}^{(i)}$ and $P_{rr}^{(i)}$ are optimized for solving the local Dirichlet problem and the constrained Neumann problem, respectively. In particular, $P_{rr}^{(i)}$ does not distinguish the interior and dual degrees of freedom, and these degrees of freedoms are mixed in the factorization process. The reordering strategy used in this work is discussed in detail in Section 4.2.1. The first factorization is used to solve the action of extension operators, $\tilde{\mathcal{H}}$ and $\tilde{\mathcal{H}}^*$, and the second factorization is used to used to solve the partially assembled problem, $\tilde{A}^{-1}$. A similar strategy has been employed in [26], in which different local solvers are used to solve the Dirichlet and the constrained Neumann problems.

The one-matrix method only stores a single incomplete factorization of the local matrix, and uses it to solve both the Dirichlet and the constrained Neumann problem. In order to use the same factorization for both problems, the reordering must preserve the interior, dual, and primal order of the degrees of freedom, i.e. the incomplete factorization applies to the reordered matrix given by

$$P^{(i)}A^{(i)}(P^{(i)})^T = \begin{pmatrix} P_I^{(i)} & & \\ & P_\Delta^{(i)} & \\ & & I_\Pi^{(i)} \end{pmatrix} \begin{pmatrix} A_{II}^{(i)} & A_{I\Delta}^{(i)} & A_{I\Gamma}^{(i)} \\ A_{\Delta I}^{(i)} & A_{\Delta\Delta}^{(i)} & A_{\Delta\Gamma}^{(i)} \\ A_{\Gamma I}^{(i)} & A_{\Gamma\Delta}^{(i)} & A_{\Gamma\Gamma}^{(i)} \end{pmatrix} \begin{pmatrix} (P_I^{(i)})^T & & \\ & (P_\Delta^{(i)})^T & \\ & & I_\Pi^{(i)} \end{pmatrix}.$$

The resulting incomplete factorization for the constrained Neumann problem, $A_{rr}^{(i)}$, is of lower quality than that used in the two-matrix method, as the reordering is constrained. However, the one-matrix method is memory efficient, as it requires half as much storage as the two-matrix method.

Formation of the Primal Schur Complement

Regardless of whether the one-matrix or the two-matrix method is used, the incomplete factorization of a local stiffness matrix, $A^{(i)}$, with respect to the interior and dual

variables results in

$$A^{(i)} = \begin{pmatrix} A_{rr}^{(i)} & A_{r\Pi}^{(i)} \\ A_{\Pi r}^{(i)} & A_{\Pi\Pi}^{(i)} \end{pmatrix} \approx \begin{pmatrix} L_{A_{rr}^{(i)}} & \\ A_{\Pi r}^{(i)} U_{A_{rr}^{(i)}}^{-1} & I_{\Pi}^{(i)} \end{pmatrix} \begin{pmatrix} U_{A_{rr}^{(i)}} & L_{A_{rr}^{(i)}}^{-1} A_{r\Pi}^{(i)} \\ & S_{\Pi}^{(i)} \end{pmatrix},$$

where $A_{rr}^{(i)} \approx L_{A_{rr}^{(i)}} U_{A_{rr}^{(i)}}$. The approximate local primal Schur complement, $S_{\Pi}^{(i)}$, is formed implicitly as a by-product of the factorization because the basis functions of $V_h^{(i)}$ are changed to make the dual and primal degrees of freedom explicit. This is different from the approach in [26], where the primal coarse basis functions are computed explicitly by solving the constrained minimization problems. At pre-processing stage, the local primal Schur complement matrices are assembled to create the global primal Schur complement, S_{Π}. The global primal Schur complement matrix is factorized exactly in this study.

Application of Inexact BDDC Preconditioner

Computing the action of the inexact BDDC preconditioner based on Eq. (4.1) requires solving the local Dirichlet problem twice and the local constrained Neumann problem once. At first glance, this appears to require three pairs of forward and backward substitutions. However, due to the structure of the matrices generated in the factorization of $A^{(i)}$, the harmonic extension and the adjoint harmonic extension can be computed in single backward and forward substitution, respectively. The implementation details of the inexact BDDC preconditioner is discussed are Appendix B.

Block Threshold ILU

Approximate solutions to the local problems are obtained from the incomplete factorization of the local matrices. If the governing equation has multiple components or the discretization order is third order or higher, then the resulting stiffness matrix is a block matrix. The inexact factorization used is a generalization of the dual threshold incomplete LU factorization, ILUT(τ, p), proposed in [72] for block matrices.

In the proposed method, the removal and introduction of the fill-ins are performed block-wise. The parameters that control the quality of the factorization are the drop tolerance, τ, and the number of additional block fill-ins per row, p. The incomplete factorization is performed row-wise to minimize the memory reallocation, as suggested in [72]. The Frobenius norm is used to measure the contribution of a given block to the factorization. For example, after the factorization of the i-th block row, all blocks that satisfy

$$\|A_{ik}\|_F \leq \tau \sum_{j \in \text{nnz}(i)} \|A^0_{ij}\|_F \tag{4.3}$$

are dropped, where $A^0_{i,:}$ is the i-th row of the matrix before factorization. Similarly, the size of blocks within a row are compared using the Frobenius norm, and at most the p largest additional block fill-ins are introduced.

Reordering

The performance of the incomplete factorization depends strongly on the ordering of the unknowns [9]. This is particularly true for a strongly nonsymmetric matrix, which arises from the discretization of advection-dominated flows. The reordering algorithm used in this work is based on the minimum discarded fill (MDF) algorithm presented in [23]. The method orders the unknown that produces the least discarded fill-in first and repeats the process in a greedy manner. The method is modified for a block matrix according to the approach in [67]. Namely, each block of the matrix is reduced to a scalar value by taking the Frobenius norm, and the MDF algorithm is applied to the scalar weight matrix.

For the one-matrix method, the degrees of freedom must be ordered such that all interior variables come first, followed by dual variables, and finally the primal variables. Thus, when selecting the variable with the least discard, the search is initially restricted to the interior variables. Once all interior variables are ordered, the next unknown is chosen from the dual variables.

Chapter 5

Results

This chapter first presents numerical experiments to assess the quality of the high-order GLS discretization developed in Chapter 2. The parallel preconditioners developed in Chapter 3 and 4 are then applied to the linear system arrising from the GLS discretization to evaluate their performance.

5.1 Assessment of High-Order GLS

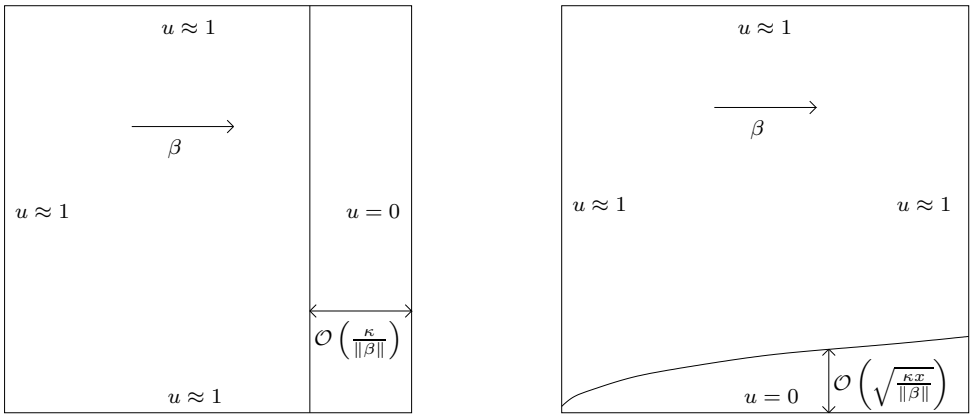
5.1.1 Advection-Diffusion Equation

Streamwise Boundary Layer Problem

In order to assess the performance of the p-scaled stabilization parameter, τ, discussed in Section 2.2.1, the streamwise boundary layer problem is solved on a domain $\Omega = (0,1) \times (0,1)$. The advection field and the viscosity parameter are set to $\beta = (1,0)^T$ and $\kappa = 0.01$, respectively. The exact solution to the problem is

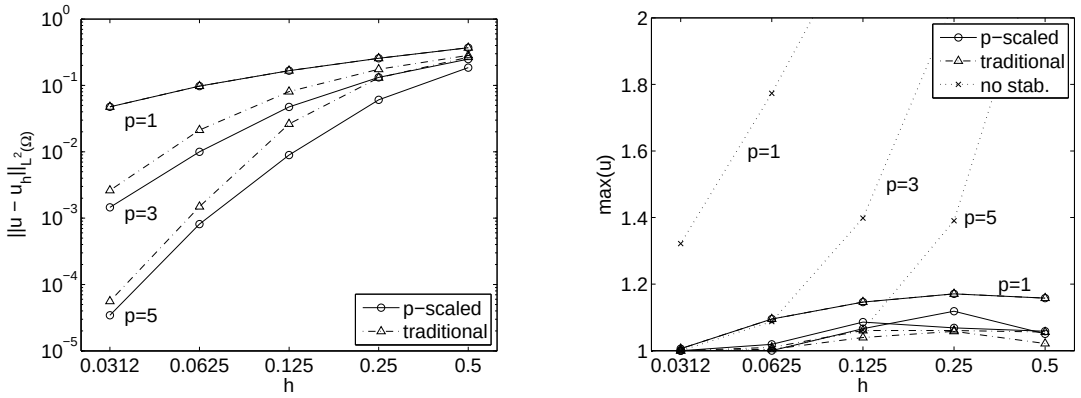
$$u(x,y) = \frac{1 - \exp(-(1-x)/\kappa)}{1 - \exp(-1/\kappa)}.$$

As shown in Figure 5-1(a), the solution is 1 almost everywhere in the domain, except in the boundary layer region of thickness $\mathcal{O}(\kappa/\|\beta\|)$ near $x = 1$. The Dirichlet boundary



(a) Streamwise boundary layer (b) Cross-stream boundary layer

Figure 5-1: Definition of the streamwise and cross-stream boundary layer problems.

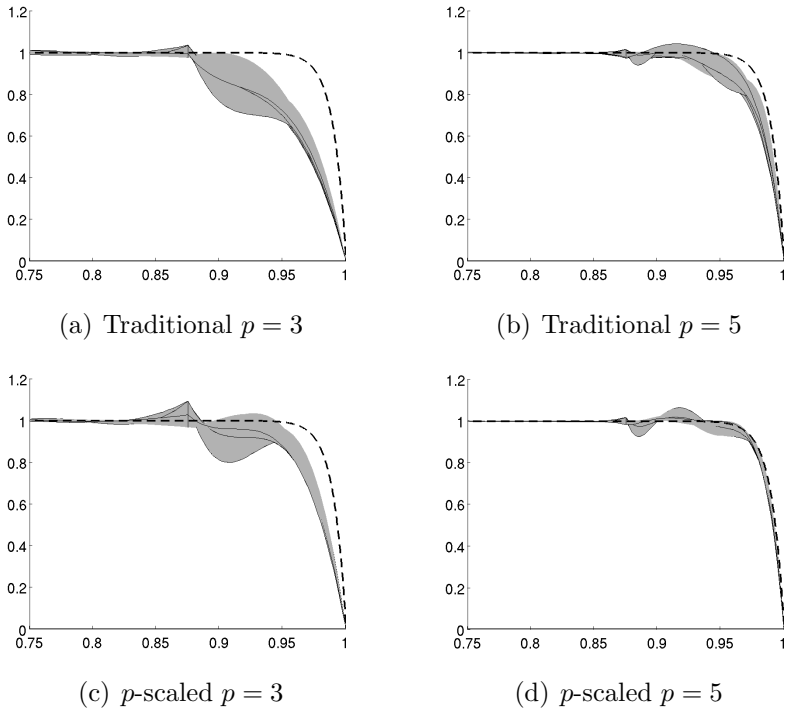


(a) Error (b) Overshoot

Figure 5-2: Comparison of the traditional and p-scaled τ for the streamwise boundary layer problem.

condition is enforced strongly everywhere on the domain boundary. Solutions are obtained on successively refined uniform, structured, isotropic meshes.

Figure 5-2 shows the result of the comparison. The quality of the solution is assessed in two manners: the L^2 norm of the error and the overshoot in the solution. The proposed τ with p-scaling produces smaller L^2 error than the traditional τ, particularly for $h = 0.25$ and 0.125. For these values of h, the mesh is not fine enough to resolve the boundary layer region using the linear finite elements; however, the higher-order interpolations resolve some of the subgrid-scale features. When τ is not scaled with p, the excess diffusion reduces the benefit of the high-order interpolation.



(a) Traditional $p = 3$ (b) Traditional $p = 5$

(c) p-scaled $p = 3$ (d) p-scaled $p = 5$

Figure 5-3: Solutions to the streamwise boundary layer problem projected onto the $y = 0$ plane for $h = 1/8$. The exact solution is shown in dotted line.

Figure 5-3 shows the solutions to the problem for $h = 1/8$ projected onto the $y = 0$ plane. The figure confirms that the p-scaled τ provides a higher resolution for this moderate value of the grid size. Thus, the primary benefit of the p-scaled τ is its superior resolution characteristics for the moderate values of the grid Peclet numbers, uh/κ, rather than the formal accuracy in the asymptotic range.

As the p-scaled τ adds less diffusion than the traditional τ, the stability of the scheme must be assessed by measuring the overshoot based on the maximum value u. Note, the true solution has $\max(u) = 1$. Figure 5-2(b) shows that the standard Galerkin scheme is unstable and produces unacceptably large overshoot. However, for $p = 5$, the scheme becomes stable for $h \geq 0.125$, as the high-order interpolation captures the subgrid-scale dissipation. Both the traditional and p-scaled stabilized schemes are stable, and the overshoot never exceeds 20%. All high-order stabilized schemes produce lower overshoot than the $p = 1$ stabilized scheme, indicating that a more aggressive reduction in the τ parameter with p could be used to further improve

the resolution characteristics of the scheme. This is consistent with the comparison in Section 2.2.1, in which the proposed τ is shown to be more diffusive than those based on the detailed analysis of the inverse estimate for $p = 2$ and $p = 3$ elements.

Cross-stream boundary Layer Problem on Anisotropic Mesh

In order to assess the quality of the high-order GLS discretization on highly anisotropic mesh often encountered in viscous flows, the high-Peclet number cross-stream boundary layer problems are solved on a domain $\Omega = (0,1)\times(0,1)$. The traditional boundary layer problem is modified by adding a source term such that the exact solution of the following form exists:

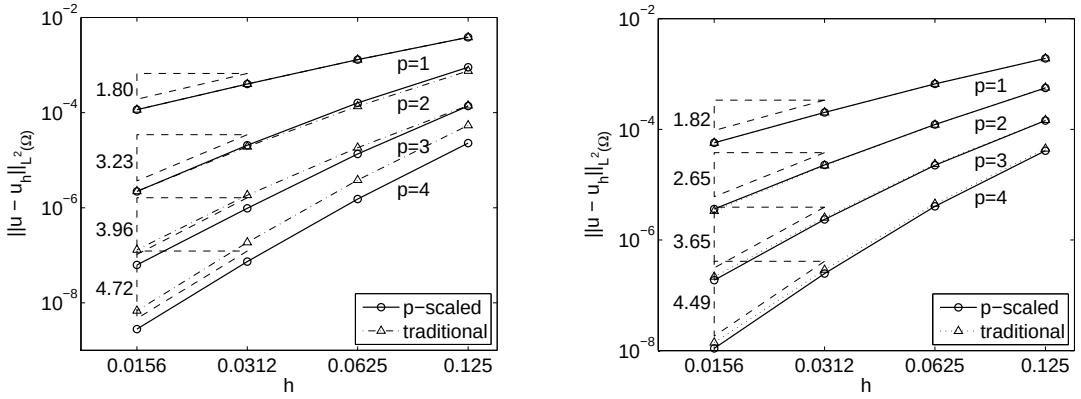
$$u = 1 - \exp\left(-\frac{y}{\sqrt{c\kappa(x+0.05)}}\right),$$

with $c = 1.3$. As shown in Figure 5-1(b), the solution is essentially 1 everywhere, except it decreases exponentially to 0 in the vicinity of the boundary $y = 0$. Note, the boundary layer thickness varies as the square root of the Peclet number. The advection field is set to $\beta = (1,0)^T$, and the two values of the viscosity considered are $\kappa = 10^{-2}$ and 10^{-6}. The anisotropic meshes are uniformly spaced in the x-direction and exponentially scaled in the y-direction such that the element on the lower boundary has the aspect ratio equal to $1/\sqrt{\kappa}$.

Figure 5-4 shows the L^2-norm of the error measured for the two problems. Asymptotically, both the p-scaled τ and the traditional τ converge at the rate of h^{p+1}, but the p-scaled τ produces lower error for high-order interpolation particularly for the $\kappa = 10^{-2}$. The modification has a smaller effect for the $\kappa = 10^{-6}$ case, as the viscous contribution to the τ parameter is small in advection-dominated case.

5.1.2 Euler Equation

The accuracy of the GLS discretization applied to the Euler equations is assessed in this section. The problem is solved on a $(-10,10)\times(0,10)$ rectangular domain with



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

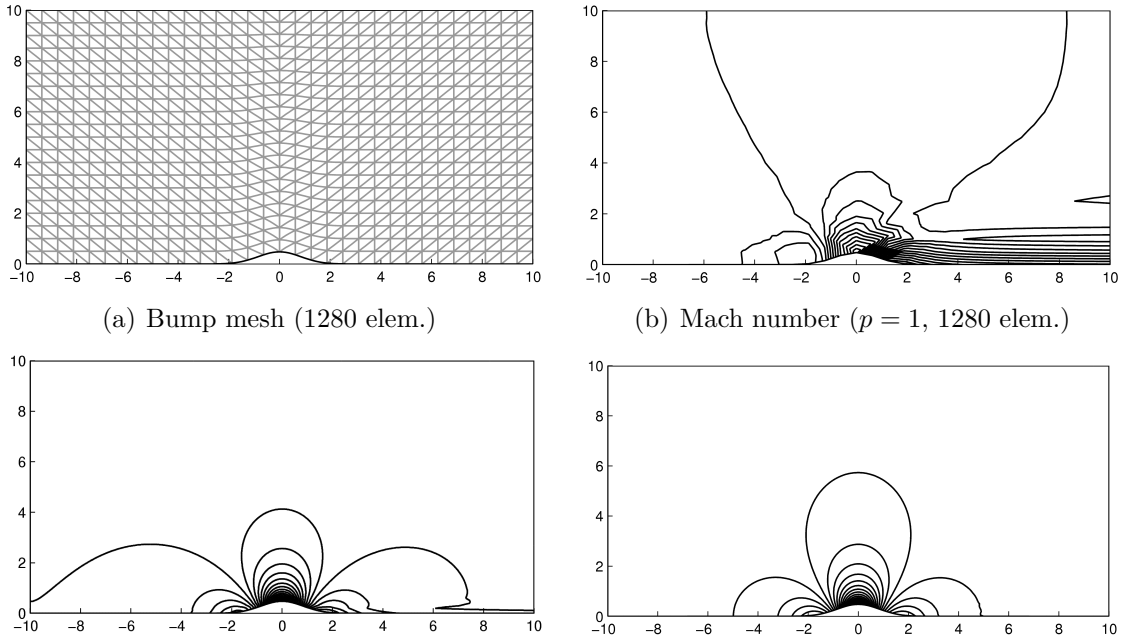
Figure 5-4: Comparison of the traditional and p-scaled τ for the cross-stream boundary layer problem.

a Gaussian bump,

$$y = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{x}{2\sigma^2}\right),$$

where $\sigma = 5/6$. 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.

Figure 5-5 shows examples of the Mach contour for the Euler Gaussian bump problem. The three cases shown are: 1) $p = 1$, 1280 elements, 2) $p = 1$, 32000 elements, and 3) $p = 5$, 1280 elements. The degrees of freedom for the discretizations are 693, 16261, and 16261, respectively. Obtaining a symmetric Mach contour is a challenging problem, as the entropy generated by the numerical dissipation affects the solution downstream of the bump. The $p = 1$, 1280-element case is completely underresolved. Even with 32000 elements, the $p = 1$ discretization still suffers from large numerical dissipation. On the other hand, the $p = 5$ solution shows nearly symmetric Mach contour. The high-order interpolation, along with the high-order geometry representation, significantly reduces the entropy generation in the bump



(a) Bump mesh (1280 elem.)

(b) Mach number ($p = 1$, 1280 elem.)

(c) Mach number ($p = 1$, 32000 elem.)

(d) Mach number ($p = 5$, 1280 elem.)

Figure 5-5: The mesh for the Gaussian bump problem and the solutions.

region. In fact, while having the same degrees of freedom, the entropy error of the $p = 5$ discretization on the 1280-element mesh is approximately 50 times less than the error of the $p = 1$ discretization on the 32000-element mesh.

Figure 5-6 shows the grid convergence of the entropy error for different order discretization on hierarchical grids with 80, 320, 1280, 5120, and 20480 elements. Asymptotically, the entropy error converges at $h^{p+1/2}$. The convergence rate is 1/2 order less than the optimal convergence rate based on the interpolation theory , but is consistent with the theory of SUPG for purely advective equations [44].

5.2 BDDC with Exact Local Solver

5.2.1 Poisson Equation

To evaluate the performance of the parallel preconditioners for elliptic problems, the Poisson equation is solved on a unit square domain. The solution is initialized to random values, which results in random initial residual. In other words, the right

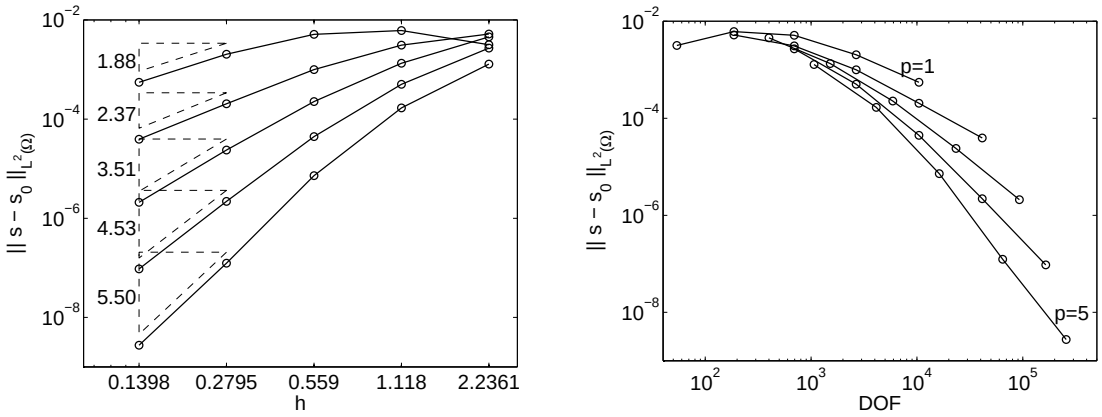
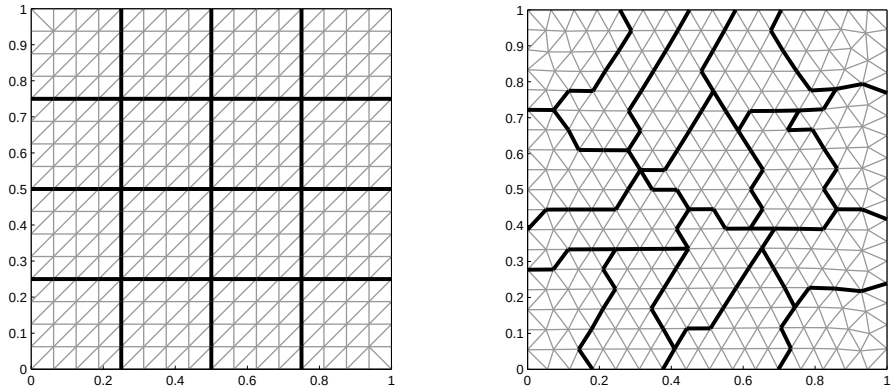


Figure 5-6: Entropy convergence for the Euler Gaussian bump problem for $p = 1, \ldots, 5$.



(a) Structured mesh and partition (b) Unstructured mesh and partition

Figure 5-7: Example of structured and unstructured mesh and partitioning.

hand side of the linear system is a random vector. The system is solved using the BDDC preconditioned GMRES until the 2-norm of the residual reduces by a factor of 10^{-13}. The condition number of the preconditioned matrix is estimated from the smallest and the largest eigenvalues obtained from the Arnoldi iteration. Two different sets of coarse primal constraints are considered; the first set only uses the degrees of freedom on the corners of the subdomains, and the second set uses both the corners and the edge averages.

BDDC Preconditioner

Table 5.1 shows the result of solving the Poisson equation using the BDDC preconditioner on a series of uniform, structured mesh shown in Figure 5-7(a). In the upper half of the table, the size of a 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 subdomains is varied from $H/h = 4$ (32 elements) to $H/h = 16$ (512 elements). For each case, the interpolation order is varied from $p = 1$ to 4.

For a fixed size of submain, the condition number is nearly independent of the number of subdomains for $N > 16$. Using both the corner and edge average constraints reduces the iteration count to approximately half of the corner only cases. When the size of subdomains varies while keeping the number of subdomains fixed, the condition number increases slowly, as expected from the $(1 + \log(H/h))^2$ dependence of the condition number on the size of subdomains. In general, using the higher order interpolation increases the condition number and the iteration count, but the scalability is maintained for all values of p.

The GMRES convergence history for several different cases are shown in Figure 5-8. As the preconditioned problem is well-conditioned, GMRES converges exponentially with the number of iterations. Thus, if the residual convergence criterion is relaxed from 10^{-13}, the number of GMRES iterations would decrease proportionally.

Lumped FETI-DP Preconditioner

Recall, the BDDC preconditioned operator on the full finite element space is given by $M^{-1}_{\text{iBDDC}} = \tilde{\mathcal{H}}\tilde{A}^{-1}(\tilde{\mathcal{H}}^*)^T$. This preconditioner extends the interface jumps resulting from solving the partially assembled problem, $\tilde{A}$, to the interior using the discrete harmonic extension. A simpler preconditioner discussed in [52] extends the interface value by zero to the interior. The resulting preconditioned operator, $M^{-1}_{\text{FETI-DP}}A = (\tilde{R})^T A^{-1}\tilde{R}A$, has the same spectrum, with exceptions of additional 0 and 1, as the preconditioned FETI-DP operator with lumped preconditioner [52]. The preconditioner

# Sub.	H/h	$p=1$		$p=2$		$p=3$		$p=4$	
		it.	κ	it.	κ	it.	κ	it.	κ
4	8	4	(1.28)	5	(1.60)	5	(1.88)	6	(2.12)
16		13	(2.22)	16	(3.42)	18	(4.35)	29	(5.13)
64		19	(2.45)	25	(3.80)	29	(4.85)	31	(5.73)
256		20	(2.51)	26	(3.90)	30	(4.97)	33	(5.87)
16	4	11	(1.63)	14	(2.58)	16	(3.38)	17	(4.06)
	8	13	(2.22)	16	(3.42)	18	(4.35)	19	(5.13)
	16	15	(2.96)	18	(4.39)	19	(5.46)	21	(6.34)

(a) Corner constraints only

# Sub.	H/h	$p=1$		$p=2$		$p=3$		$p=4$	
		it.	κ	it.	κ	it.	κ	it.	κ
4	8	5	(1.09)	5	(1.25)	6	(1.43)	6	(1.59)
16		9	(1.15)	12	(1.39)	13	(1.62)	15	(1.82)
64		10	(1.18)	12	(1.45)	15	(1.70)	16	(1.91)
256		10	(1.18)	13	(1.46)	15	(1.72)	17	(1.94)
16	4	7	(1.04)	9	(1.20)	11	(1.38)	12	(1.55)
	8	9	(1.15)	12	(1.39)	13	(1.62)	14	(1.82)
	16	11	(1.32)	13	(1.64)	15	(1.91)	16	(2.14)

(b) Corner and edge average constraints

Table 5.1: The GMRES iteration counts and the condition numbers for the BDDC method for the Poisson problem on structured meshes.

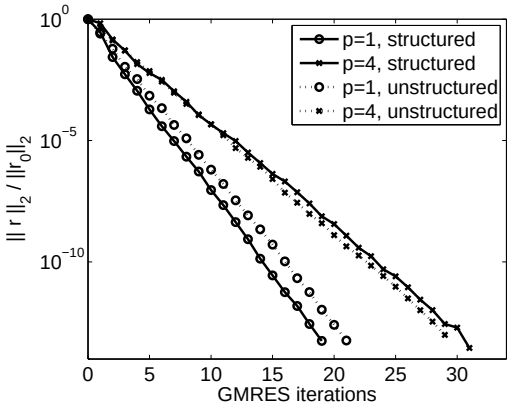


Figure 5-8: Typical GMRES convergence history for the Poisson problem. (64 subdomains, $H/h = 8$, corner constraints)

# Sub.	H/h	$p=1$ it.	$p=1$ κ	$p=2$ it.	$p=2$ κ	$p=3$ it.	$p=3$ κ	$p=4$ it.	$p=4$ κ
4	8	14	(4.41)	22	(11.29)	28	(18.87)	32	(27.53)
16		28	(9.08)	41	(30.01)	52	(56.88)	58	(90.20)
64		41	(10.17)	68	(34.19)	87	(65.18)	102	(103.77)
256		45	(10.43)	80	(35.18)	107	(67.15)	131	(106.98)
16	4	19	(3.74)	31	(12.24)	40	(23.76)	46	(38.30)
	8	28	(9.08)	41	(30.01)	52	(56.88)	58	(90.20)
	16	39	(22.63)	55	(72.05)	68	(133.45)	76	(208.51)

(a) Corner constraints only

# Sub.	H/h	$p=1$ it.	$p=1$ κ	$p=2$ it.	$p=2$ κ	$p=3$ it.	$p=3$ κ	$p=4$ it.	$p=4$ κ
4	8	12	(1.89)	18	(4.09)	25	(6.57)	29	(9.38)
16		17	(2.14)	28	(5.05)	37	(8.40)	44	(12.24)
64		18	(2.20)	31	(5.29)	42	(8.85)	51	(12.93)
256		18	(2.22)	32	(5.34)	43	(8.95)	52	(13.10)
16	4	11	(1.31)	19	(2.50)	26	(4.11)	31	(5.89)
	8	17	(2.14)	28	(5.05)	37	(8.40)	44	(12.24)
	16	25	(4.17)	40	(10.54)	52	(17.54)	60	(25.52)

(b) Corner and edge average constraints

Table 5.2: The GMRES iteration counts and the condition numbers for the lumped FETI-DP method for the Poisson problem on structured meshes.

requires only one application of the factorized matrix instead of two solves for the BDDC preconditioner using the implementation in Appendix B. In fact, the application of the lumped FETI-DP preconditioner is only as expensive as the subdomain-wise block Jacobi preconditioner, M_J, for which the condition number deteriorate with the number of subdomain, i.e. $\kappa(M_J^{-1}A) = \mathcal{O}(\log(1+(H/h))^2/H^2)$.

The result of solving the Poisson equation using the lumped FETI-DP preconditioner is shown in Table 5.2. For a fixed size of the subdomains, the condition number is essentially constant for $N > 16$. However, as the size of the subdomains increases, the condition number increases rapidly. Similarly, the conditioner number worsens rapidly with the interpolation order, due to the increase in the degrees of freedom per subdomain. In realistic problems, the degrees of freedom per subdomain are much larger than the values considered here, and the performance of the preconditioner is expected to degrade further. Therefore, the lumped FETI-DP preconditioner is not

		$p=1$		$p=2$		$p=3$		$p=4$	
# Sub.	H/h	it.	κ	it.	κ	it.	κ	it.	κ
4	8	11	(1.45)	11	(1.57)	11	(1.68)	12	(1.79)
16		17	(2.29)	19	(3.02)	20	(3.54)	21	(3.99)
64		21	(2.84)	25	(3.85)	27	(4.58)	29	(5.17)
256		25	(3.81)	29	(4.99)	31	(5.87)	33	(6.58)
16	4	16	(2.30)	19	(3.07)	21	(3.71)	22	(4.26)
	8	17	(2.29)	19	(3.02)	20	(3.54)	21	(3.99)
	16	18	(2.66)	21	(3.55)	22	(4.24)	23	(4.81)

(a) Corner constraints only

		$p=1$		$p=2$		$p=3$		$p=4$	
# Sub.	H/h	it.	κ	it.	κ	it.	κ	it.	κ
4	8	10	(1.25)	10	(1.35)	11	(1.41)	11	(1.46)
16		13	(1.85)	13	(1.95)	14	(1.95)	15	(1.99)
64		15	(1.89)	15	(2.01)	16	(2.02)	16	(2.07)
256		16	(1.90)	17	(2.00)	18	(2.17)	19	(2.43)
16	4	12	(1.73)	13	(1.88)	14	(1.91)	15	(1.96)
	8	13	(1.85)	13	(1.95)	14	(1.95)	15	(1.99)
	16	14	(1.69)	15	(1.76)	16	(1.80)	17	(1.92)

(b) Corner and edge average constraints

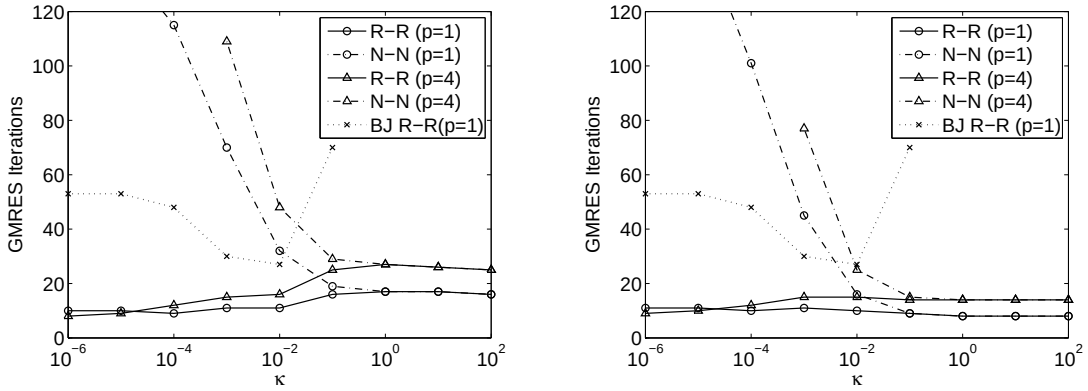
Table 5.3: The GMRES iteration counts and the condition numbers for the BDDC method for the Poisson problem on unstructured meshes.

considered for the rest of the study.

Unstructured Mesh and Partitioning

The Poisson problem is solved on unstructured meshes. The unstructured meshes are generated using the Distmesh package [68] and partitioned using the METIS package [46], with the objective of balancing the number of elements in each subdomain and minimizing the interface degrees of freedom. Figure 5-7(b) shows an example of the unstructured mesh and the partitioning. The measure of H/h is based on the average number of the element per subdomain. Thus, for a subdomain with an irregular shape, H/h can be larger than the numbers shown.

A comparison of Table 5.3 and 5.1 shows that the $p = 1$ interpolation experiences approximately 50% increase in the iteration count with the unstructured partitionings, but the $p = 4$ interpolation is essentially unaffected by the unstructured partitionings.



(a) Corner only

(b) Corner and edge

Figure 5-9: Comparison of the Robin-Robin and the Neumann-Neumann interface conditions. (64 subdomains, $H/h = 8$)

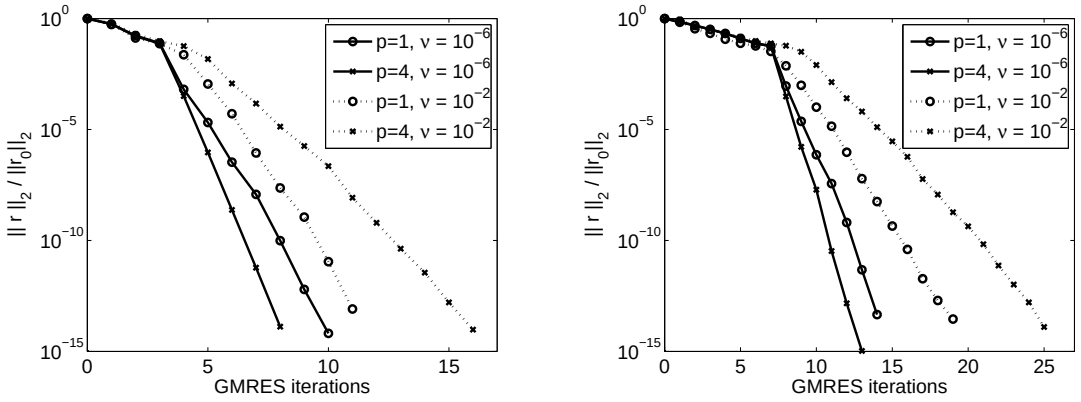
Thus, using an unstructured partitioning has a bigger impact for the linear elements than for the higher-order elements. Nevertheless, the BDDC preconditioner maintains a good scaling for all interpolation orders considered.

5.2.2 Advection-Diffusion Equation

Interface Conditions

In this section the cross-stream advection-diffusion boundary layer problem, shown in Figure 5-1(b), is solved on a uniform mesh to assess the performance of the parallel preconditioners for a wide range of the physical Peclet numbers, uL/κ, with the viscosity varying from 10^2 (diffusion-dominated) to 10^{-6} (advection-dominated). A structured partitioning with 64 subdomains and $H/h = 8$ is used. The methods compared are the BDDC method using the Robin-Robin interface condition discussed in Section 3.3, the BDDC method using the Neumann-Neumann interface condition that arises naturally from the discretization of elliptic problems, and the subdomain-wise block Jacobi preconditioner with the Robin-Robin interface condition.

Figure 5-9 shows the result of the comparison. In the diffusion-dominated case, the Robin-Robin and the Neumann-Neumann interface conditions are identical, and the BDDC method based on either interface condition performs well. The subdomain-



(a) 64 subdomains (8×8) (b) 256 subdomains (16×16)

Figure 5-10: Typical GMRES convergence histories for the advection-diffusion equation. ($H/h = 8$, corner constraints)

wise block Jacobi preconditioner becomes ill-conditioned in the diffusion limit as the interior subdomains float in the absence of the constraints. In the advection-dominated case, the Robin-Robin BDDC maintains its performance. On the other hand, the Neumann-Neumann BDDC performs poorly, due to the loss of the positivity of the local bilinear forms. In fact, a simple subdomain-wise block Jacobi preconditioner with the Robin-Robin interface condition performs better than the Neumann-Neumann BDDC in the advection-dominated cases. The performance of the Robin-Robin BDDC is independent of the choice of the constraints in the advection-dominated cases, while the use of both the corner and the edge average constraints roughly halves the iteration count in the diffusion-dominated cases.

The GMRES convergence histories for several cases using the BDDC method with the Robin-Robin interface condition is shown in Figure 5-10. Unlike the convergence history for the Poisson equation shown in Figure 5-8, the residual does not decay significantly for the first $N_{\text{char}}/2$ iterations, where N_{char} is the maximum number of the subdomains that a characteristic passes through. The theoretical study of the Robin-Robin type domain decomposition method is conducted in details in [1] by considering the quasi-one-dimensional flow on vertical strips. The numerical results obtained in this section is consistent with the theoretical results presented in [1].

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

(a) Corner constraints only

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

(b) Corner and edge average constraints

Table 5.4: The GMRES iteration count for the BDDC method with the Robin-Robin interface condition for the boundary layer problem on uniform meshes.

Scalability

Table 5.4 shows the result of the scalability study for the advection-diffusion equation for $\kappa = 10^{-2}$, 10^{-4}, and 10^{-6}. In the advection-dominated cases, the iteration count increases with the maximum number of subdomains that a characteristic passes through, N_{char}, as the hyperbolic equation does not possess the smoothing property of the elliptic equation. On the other hand, the iteration count is independent of the size of the subdomain, the interpolation order, and the choice of constraints in the advection-dominated case. Thus, in the advection-limit, a partitioning strategy based on capturing the strong characteristics, such as those used in [24] and [66], 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.

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

(a) Corner constraints only

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

(b) Corner and edge average constraints

Table 5.5: The GMRES iteration count for the BDDC method with the Robin-Robin interface condition for the boundary layer problem on anisotropic meshes.

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 $\kappa = 10^{-6}$, the 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 5.5 shows the scalability result using the series of anisotropic meshes. Comparing to Table 5.4, 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 elliptic behavior even for $\kappa = 10^{-6}$

#. Sub.	Subdomain Block Jacobi $p=1$	$p=2$	$p=3$	BDDC (Corner only) $p=1$	$p=2$	$p=3$	BDDC (Corner + edge) $p=1$	$p=2$	$p=3$
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) Neumann-Neumann Interface Condition

#. Sub.	Subdomain Block Jacobi $p=1$	$p=2$	$p=3$	BDDC (Corner only) $p=1$	$p=2$	$p=3$	BDDC (Corner + edge) $p=1$	$p=2$	$p=3$
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) Robin-Robin Interface Condition

Table 5.6: The GMRES iteration count for the Euler bump problem. (160 elem. per subdomain, $\Delta t = \infty$)

with the highly anisotropic mesh.

5.2.3 Euler Equation

In this section, the Euler equation problem with a Gaussian bump discussed in Section 5.1.2 is solved using the subdomain-wise block Jacobi, BDDC with corner constraints, and BDDC with corner and edge average constraints using the Neumann-Neumann (i.e. naturally arising) and Robin-Robin interface conditions. The size of the subdomain is fixed to 160 elements, and increasingly larger problem is 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 5.6 shows the result of the comparison. For all types of preconditioners considered, the Robin-Robin interface condition performs significantly better than the Neumann-Neumann interface condition. Similar to the result obtained for the advection-dominated case of the advection-diffusion equation, a simple subdomain-wise block Jacobi preconditioner with the the Robin-Robin interface condition out-

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

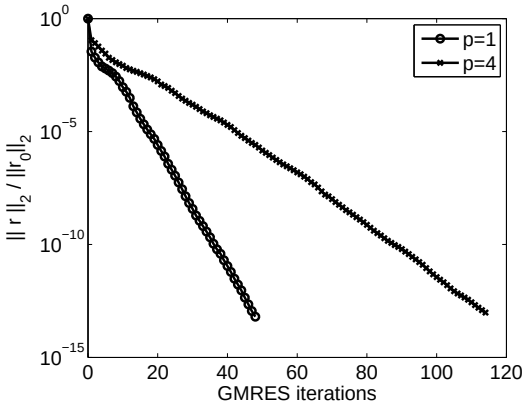


Figure 5-11: Typical GMRES convergence history for the Euler problem. (32 subdomains, 160 elem. per subdomain, corner and edge constraints)

performs the BDDC preconditioner with the Neumann-Neumann interface condition. However, unlike the advection-dominated case of the advection-diffusion equation, the iteration count is dependent on the type of constraints for the BDDC method, and using both the corner and edge average constraints significantly improves the performance of the solver, especially when a large number of subdomains is employed. The difference suggests the underlying ellipticity of the acoustic modes in steady, subsonic flow.

Table 5.7 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.

The GMRES convergence histories for a typical subsonic Euler equation case is

shown in Figure 5-11. The linear residual decays exponentially with the number of iterations. Since the convergence criterion of the linear problem is typically relaxed in the early stage of Newton iteration, the number of GMRES iteration would decrease proportionally in the early stage of the nonlinear solve.

5.3 BDDC with Inexact Local Solvers

5.3.1 Advection-Diffusion Equation

Serial Preconditioner

The quality of the incomplete factorizations are assessed by solving the advection-diffusion equation for the advection-dominated case ($\kappa = 10^{-6}$) and balanced advection-diffusion case ($\kappa = 10^{-2}$) on the uniform and anisotropic meshes. The ILUT factorization with minimum discarded fill (MDF) and approximate minimum degree (AMD) [3] orderings are used as the inexact local solvers. The maximum block fill per row is varied from 0 to 10, and the traditional and p-scaled τ are used for stabilization.

Table 5.8(a) and 5.8(b) show the results of solving the problem on a uniform mesh with 512 elements. The MDF reordering performs significantly better than the AMD reordering for advection-dominated cases using ILU(0). However, as the number of maximum fill-ins increases, the difference in the MDF and AMD reordering become insignificant. For the $\kappa = 10^{-2}$ case, the p-scaled τ provides slightly better convergence rate than the traditional τ; the p-scaling results in no differences for the $\kappa = 10^{-6}$ case as the viscous contribution to the τ parameter is small.

Table 5.8(c) and 5.8(d) show the result for solving the same problems on anisotropic meshes with the element on the wall having aspect ratios of 10 and 1000 for $\kappa = 10^{-2}$ and $\kappa = 10^{-6}$, respectively. The benefit of the MDF reordering as well as the p-scaled τ is more pronounced than the isotropic mesh cases. However, the iteration count is generally higher than in the isotropic mesh cases due to the presence of elliptic modes in the highly anisotropic boundary layer region; in order to efficiently precondition

	MDF ($\tau_{p-\text{scaled}}$)				MDF ($\tau_{\text{traditional}}$)				AMD ($\tau_{p-\text{scaled}}$)			
fill	0	2	5	10	0	2	5	10	0	2	5	10
$p=1$	11	6	4	3	11	6	4	3	22	13	10	6
$p=2$	12	8	6	5	17	11	10	7	22	13	10	8
$p=3$	16	11	8	5	27	19	10	7	27	16	12	10
$p=4$	18	11	8	5	24	12	9	6	31	19	12	10

(a) Isotropic 512 elem., $\kappa = 10^{-2}$

	MDF ($\tau_{p-\text{scaled}}$)				MDF ($\tau_{\text{traditional}}$)				AMD ($\tau_{p-\text{scaled}}$)			
fill	0	2	5	10	0	2	5	10	0	2	5	10
$p=1$	9	6	5	3	9	6	5	3	34	16	11	7
$p=2$	12	6	6	4	12	6	6	4	51	16	10	6
$p=3$	12	10	6	6	12	10	6	6	58	18	10	6
$p=4$	10	6	5	4	10	6	5	4	61	20	10	7

(b) Isotropic 512 elem., $\kappa = 10^{-6}$,

	MDF ($\tau_{p-\text{scaled}}$)				MDF ($\tau_{\text{traditional}}$)				AMD ($\tau_{p-\text{scaled}}$)			
fill	0	2	5	10	0	2	5	10	0	2	5	10
$p=1$	12	8	6	4	12	8	6	4	29	15	11	7
$p=2$	15	11	9	7	16	11	10	8	46	17	11	9
$p=3$	18	13	11	9	26	18	12	9	53	18	13	10
$p=4$	22	14	10	8	34	19	14	11	55	22	15	11

(c) Anisotropic 512 elem., $AR = 10$, $\kappa = 10^{-2}$,

	MDF ($\tau_{p-\text{scaled}}$)				MDF ($\tau_{\text{traditional}}$)				AMD ($\tau_{p-\text{scaled}}$)			
fill	0	2	5	10	0	2	5	10	0	2	5	10
$p=1$	10	7	5	4	10	7	5	4	32	16	11	7
$p=2$	16	12	9	8	14	11	8	6	49	16	10	8
$p=3$	16	11	10	8	17	11	9	8	58	18	11	8
$p=4$	17	12	10	8	22	12	10	8	61	19	12	9

(d) Anisotropic 512 elem., $AR = 1000$, $\kappa = 10^{-6}$

Table 5.8: The GMRES iteration count for the ILUT preconditioner at various fill-levels applied to the advection-diffusion equation on a single domain.

	8192 elem.			32768 elem.		
fill	5	10	∞	5	10	∞
Factorization (sec)	0.094	0.099	0.598	0.507	0.834	4.941
Application of Preconditioner (sec)	0.010	0.012	0.049	0.037	0.094	0.233
Memory ($\text{nnz}(M^{-1})/\text{nnz}(A)$)	2.261	2.723	6.594	2.377	3.705	8.531

(a) $p = 1$

	512 elem.			2048 elem.		
fill	5	10	∞	5	10	∞
Factorization (sec)	0.074	0.087	0.976	0.371	0.465	46.334
Application of Preconditioner (sec)	0.000	0.006	0.030	0.020	0.025	0.356
Memory ($\text{nnz}(M^{-1})/\text{nnz}(A)$)	1.605	2.010	3.637	1.663	2.207	7.579

(b) $p = 4$

Table 5.9: The time for performing the incomplete factorizations, the time for applying the preconditioner, and the memory requirement for storing the factored matrix.

these modes, the use of a multigrid-type coarse correction should be considered.

In order to quantify the cost of the factorization, the timing and memory requirement for the incomplete factorizations ILUT(10^{-8},5) and ILUT(10^{-8},10) and the exact factorization are compared. The viscosity is set to $\kappa = 10^{-2}$, and the MDF reordering is used for the incomplete factorization cases. The exact factorization is performed using the same ILUT algorithm but using the AMD ordering to minimize the fill. This results in the exact factorization algorithm that is slower than the sophisticated algorithm that takes advantage of the memory hierarchy of the modern computers (e.g. UMFPACK [22]), but allows for more consistent comparison of the inexact solver and the exact solver.

The timing and memory requirements for representative cases are shown in Table 5.9. In general, the cost of the exact factorization rises rapidly, both in terms of the time and the memory requirement, with the size of the problem and the number of neighbors. For instance, the incomplete factorization are approximately six times faster for the $p = 1$, 8192-element case, but is more than 100 times faster for the $p = 4$, 2048-element case. The inexact factorizations also reduces the memory requirement and the time for applying the preconditioner, which is the major cost in the GMRES algorithm.

BDDC with Inexact Solver

The result for the BDDC method based on the ILUT factorization of local stiffness matrices is shown in Table 5.10. The performance of both the one-matrix method and the two-matrix method, discussed in 4.2.1, are assessed. For the infinite fill case, the drop tolerance is set to zero such that the local solvers become direct solvers; for all other cases, the drop tolerance is set to 10^{-8}.

In general, the two-matrix method performs significantly better than the one-matrix method, especially when the number of maximum allowed fill-in is low. For the $\kappa = 10^{-2}$ case on the isotropic mesh, the two-matrix method incurs less than 30% increase in the iteration count compared to the exact local solver with as few as five additional fill-ins per row, outperforming the one-matrix method with 20-fills. The experiment reasserts the importance of the reordering when an incomplete factorization is employed.

When the anisotropic meshes are employed, the performance of the BDDC method using the incomplete factorizations tend to degrade quicker than the isotropic mesh cases due to the presence of the elliptic modes in the boundary layer region. In particular, for $\kappa = 10^{-2}$ case, the number of fill-ins required to achieve the iteration count close to the case with the exact local solver is significantly higher. The use of a multigrid-type correction in the local solver, which can capture the elliptic modes, is expected to improve the result significantly.

Scalability

The scaling result for the inexact BDDC method using the ILUT(10^{-8}, 5) local solvers with the MDF reordering is shown in Table 5.11. Compared to the result obtained for the same case using the exact local solver shown in Table 5.4, the iteration counts increase in general. The degradation of the iteration count is more pronounced for the one-matrix method than the two-matrix method due to the poorer inexact factorization. The case with $\kappa = 10^{-2}$ suffers from a larger increase in the iteration count due to the stronger elliptic behavior of these flows. However, considering the orders

	Two-matrix method						One-matrix method					
fill	2	5	10	15	20	∞	2	5	10	15	20	∞
$p=1$	15	13	12	12	12	11	24	18	15	13	12	11
$p=2$	22	19	16	15	15	13	30	23	19	17	16	13
$p=3$	33	24	18	16	16	14	42	31	24	20	19	14
$p=4$	31	26	18	17	16	16	46	35	25	22	20	16

(a) Isotropic mesh, $\kappa = 10^{-2}$

	Two-matrix method						One-matrix method					
fill	2	5	10	15	20	∞	2	5	10	15	20	∞
$p=1$	14	11	11	10	10	10	18	15	11	10	10	10
$p=2$	21	14	13	13	13	9	21	19	17	15	13	9
$p=3$	19	13	9	9	9	9	22	18	17	14	12	9
$p=4$	18	14	13	13	12	9	29	21	16	15	14	9

(b) Isotropic mesh, $\kappa = 10^{-6}$

	Two-matrix method						One-matrix method					
fill	2	5	10	15	20	∞	2	5	10	15	20	∞
$p=1$	22	20	19	18	18	18	33	21	19	18	18	18
$p=2$	34	26	22	21	20	20	69	49	34	28	25	20
$p=3$	43	31	24	22	22	20	78	61	45	37	31	21
$p=4$	48	34	26	24	24	22	95	68	48	38	32	22

(c) Anisotropic mesh, $AR = 10$, $\kappa = 10^{-2}$

	Two-matrix method						One-matrix method					
fill	2	5	10	15	20	∞	2	5	10	15	20	∞
$p=1$	19	16	15	14	14	14	26	18	15	14	14	14
$p=2$	29	20	17	16	16	16	76	50	33	27	24	16
$p=3$	32	25	20	18	18	17	72	52	38	32	29	17
$p=4$	36	26	21	21	20	20	84	66	37	30	30	20

(d) Anisotropic mesh, $AR = 1000$, $\kappa = 10^{-6}$

Table 5.10: The GMRES iteration count for the BDDC method with the ILUT local solvers (64 subdomains, $H/h = 8$, corner constraints)

		$\kappa = 10^{-6}$				$\kappa = 10^{-2}$			
#. Sub.	H/h	$p = 1$	2	3	4	1	2	3	4
4	8	5	7	7	8	6	10	10	10
16		9	10	9	10	9	12	14	15
64		11	14	13	14	13	19	24	26
256		19	22	21	21	25	34	45	45
16	4	11	12	12	11	9	11	11	11
	8	9	10	9	10	9	12	14	15
	16	8	11	9	11	11	19	25	28

(a) Two-matrix method

		$\kappa = 10^{-6}$				$\kappa = 10^{-2}$			
#. Sub.	H/h	$p = 1$	2	3	4	1	2	3	4
4	8	7	9	9	10	7	10	11	11
16		11	15	14	17	12	15	17	19
64		15	19	18	21	18	23	31	35
256		24	26	25	29	29	41	56	58
16	4	12	14	14	14	10	13	14	14
	8	11	15	14	17	12	15	17	19
	16	11	17	17	20	16	22	28	32

(b) One-matrix method

Table 5.11: The GMRES iteration counts for the advection-diffusion equation on isotropic mesh using the ILUT(10^{-8}, 5) local solvers.

of magnitude improvement in the factorization time and the preconditioner application time shown in Table 5.9, the inexact solver improves the overall solution time, especially for large problems. The inexact algorithm also benefit from the substantially reduced memory requirement. Thus, for advection-dominated flows, the inexact BDDC algorithm based on ILUT with appropriate ordering outperforms the direct method even for the two-dimensional problem, and the benefit is expected to increase further for three-dimensional problems.

Chapter 6

Conclusion

This work presents the high-order accurate Galerkin Least-Squares method combined with massively parallel implicit solvers. The highlight of the thesis includes: a simple stabilization parameter correction for the high-order discretization of advective-diffusive systems, the use of non-overlapping domain decomposition methods for high-order triangular elements, the extension of the Robin-Robin interface condition to a system of hyperbolic system, and the assessment of the BDDC method using local solvers based on incomplete factorizations.

The stabilization parameter was adjusted to account for the subgrid-scale resolution provided by the high-order elements. The modified p-scaled stabilization improves not only the resolution characteristics while maintaining the stability, but also the quality of the incomplete factorization applied to the discrete systems. The optimum h^{p+1} convergence was observed for advection-diffusion problems, and $h^{p+1/2}$ convergence was observed for subsonic Euler flows.

The BDDC method was applied to the high-order discretization using triangular elements for the Poisson equation, the advection-diffusion equation, and the Euler equation. For the Poisson equation, the condition number of the BDDC preconditioned operator increases with the order of interpolation, but the scalability is maintained. The gap in the iteration count for the lower and higher order interpolations was found to be smaller on unstructured meshes. On the other hand, the FETI-DP method with

lumped preconditioner was not competitive for high-order interpolation due to the large degrees of freedom associated with each subdomain.

For the advection-dominated flows, the iteration count is governed by the maximum number of subdomains that a characteristic crosses, and the count is independent of the size of the subdomain or the interpolation order. The BDDC preconditioner remained effective on the exponentially-scaled boundary layer meshes with highly anisotropic elements. The Robin-Robin interface condition was extended to the Euler equation using the entropy-symmetrized formulation. The resulting preconditioner outperformed other preconditioners considered. The Robin-Robin interface condition treated the advection modes, while the global coarse correction treated the acoustic modes present in the system. A further improvement in the interface condition and the global coarse correction may be required to effectively precondition nearly incompressible or supersonic flows.

While the BDDC method with exact local solvers performs well, the inexact BDDC method needs further improvement. The two-matrix method demonstrated that when separate reorderings are employed to solve the Dirichlet and the constrained Neumann problem, the inexact BDDC maintains the performance of the exact counterpart even with a small fill for the advection-dominated cases. However, the two-matrix method is memory inefficient, and the performance of the one-matrix method must be improved for the BDDC method to be competitive in practical aerospace applications. Moreover, the incomplete factorizations used in this work failed to capture the elliptic modes, which degraded the performance of the parallel algorithm. The use of either p- or h-multigrid correction within the local solver is the first step in improving the performance of the inexact BDDC preconditioner. The BDDC method should also be extended to three dimensions and tested on viscous flows over complex geometries.

Appendix A

Parallel Implementation

The grid partitioning is performed using the METIS library [46], with an objective of balancing the number of elements per subdomain while minimizing the communication volume (i.e. minimizing the interface degrees of freedom). Note, this is different from the approaches taken in [24] and [66], in which the strength of coupling among the degrees of freedom are considered in partitioning. Keeping degrees of freedom with strong couplings to a single subdomain was important in these work, because the parallel solver used in the work applied ILU to the global stiffness matrix while ignoring the coupling between the elements in different subdomains. For advection-dominated flows, coupling-weighted methods tend to produce highly-stretched subdomains with large interfaces. These features are undesirable for the domain decomposition method based on Schur complement, as they produce a large Schur complement system and nonuniform partitioning [76]. Thus, the method used in the current approach focused on minimizing the interface degrees of freedom.

All parallel communications are performed using the Message Passing Interface (MPI) library. The non-blocking communications are used whenever possible to minimize the idle time. The global primal system, $S_\Pi u_\Pi = g_\Pi$, is solved using a direct sparse solver UMFPACK [22].

Appendix B

Implementation of Inexact BDDC

As discussed in Chapter 4, an inexact BDDC preconditioner can be developed from the inexact application of $\tilde{A}^{-1}$ and the inexact discrete harmonic extensions. This section presents a detailed implementation of BDDC based on inexact factorization that requires only single factorization of the local stiffness matrix, $A^{(i)}$. An inexact factorization of the stiffness matrix produces

$$\begin{pmatrix} L_{A_{rr}^{(i)}} & \\ A_{\Pi r}^{(i)} U_{A_{rr}^{(i)}}^{-1} & I_{\Pi}^{(i)} \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} U_{A_{rr}^{(i)}} & L_{A_{rr}^{(i)}}^{-1} A_{r\Pi}^{(i)} \\ & S_{\Pi}^{(i)} \end{pmatrix}.$$

The local primal Schur complements, which are formed as bi-products of the factorizations, are gathered to the root processor and assembled to form S_{Π}. The matrix S_{Π} is factored exactly using a sparse direct solver.

B.1 Application of Inexact $\tilde{A}^{-1}$

This section considers application of $\tilde{A}^{-1}$ to a vector $\tilde{v}$ to generate $\tilde{u} = \tilde{A}^{-1}\tilde{v}$. Note, $\tilde{u}, \tilde{v} \in \tilde{V}$.

1. Using forward substitution, solve

$$\begin{pmatrix} L_{A_{rr}^{(i)}} & \\ A_{\Pi r}^{(i)} U^{-1}_{A_{rr}^{(i)}} & I_{\Pi}^{(i)} \end{pmatrix} \begin{pmatrix} w_r^{(i)} \\ \mathring{w}_{\Pi}^{(i)} \end{pmatrix} = \begin{pmatrix} v_r^{(i)} \\ 0_{\Pi}^{(i)} \end{pmatrix}$$

   The solution is given by

$$w_r^{(i)} = L^{-1}_{A_{rr}^{(i)}} v_r^{(i)} \quad \text{and} \quad \mathring{w}_{\Pi}^{(i)} = -A_{\Pi r}^{(i)} (A_{rr}^{(i)})^{-1} v_r^{(i)}$$

2. Construct global primal vector w_{Π}

$$w_{\Pi} = v_{\Pi} + \sum_{i=1}^{N} (R_{\Pi}^{(i)})^T \mathring{w}_{\Pi}^{(i)} = \sum_{i=1}^{N} (R_{\Pi}^{(i)})^T \left[D_{\Pi}^{(i)} v_{\Pi}^{(i)} + \mathring{w}_{\Pi}^{(i)} \right]$$

   The action of $\sum_{i=1}^{N} (R_{\Pi}^{(i)})^T \cdot$ corresponds to global sum of primal variables. Thus, all primal variables are gathered to the root processor.

3. Solve $S_{\Pi} u_{\Pi} = w_{\Pi}$ for u_{Π} on the root processor.

4. Extract local primal variables $u_{\Pi}^{(i)} = R_{\Pi}^{(i)} u_{\Pi}$. The operation corresponds to scattering variables.

5. Using back substitution, solve

$$\begin{pmatrix} U_{A_{rr}^{(i)}} & L^{-1}_{A_{rr}^{(i)}} A_{r\Pi}^{(i)} \\ & I_{\Pi}^{(i)} \end{pmatrix} \begin{pmatrix} u_r^{(i)} \\ u_{\Pi}^{(i)} \end{pmatrix} = \begin{pmatrix} w_r^{(i)} \\ u_{\Pi}^{(i)} \end{pmatrix}$$

   The solution is given by

$$\begin{aligned} u_{\Pi}^{(i)} &= u_{\Pi}^{(i)} \\ u_r^{(i)} &= U^{-1}_{A_{rr}^{(i)}} (w_r^{(i)} - L^{-1}_{A_{rr}^{(i)}} A_{r\Pi}^{(i)} A_{r\Pi}^{(i)} u_r^{(i)}) = (A_{rr}^{(i)})^{-1} (v_r^{(i)} - A_{r\Pi}^{(i)} u_{\Pi}^{(i)}) \end{aligned}$$

   The resulting vector is $\tilde{u} = \tilde{A}^{-1} \tilde{v}$.

B.2 Application of Inexact $\tilde{\mathcal{H}}_i$

Consider the application of the extension operator $\tilde{\mathcal{H}} : \tilde{V} \rightarrow V$

$$\tilde{\mathcal{H}}_i = \begin{pmatrix} I & A_{II}^{-1}\tilde{A}_{I\Gamma}J_{1,D,\Gamma} \\ & \tilde{R}_{D,\Gamma}^T \end{pmatrix} = \begin{pmatrix} I_I & A_{II}^{-1}A_{I\Delta}(I_\Delta - R_\Delta R_{D,\Delta}^T) & \\ & R_{D,\Delta}^T & \\ & & I_\Pi \end{pmatrix}.$$

The application of $\tilde{\mathcal{H}}$ to create $u = \tilde{\mathcal{H}}\tilde{v}$, $u \in V$, $\tilde{v} \in \tilde{V}$, is performed as follows

1. Compute $w_\Delta = R_\Delta R_{D,\Delta}^T v_\Delta$

   $$w_\Delta^{(i)} = \sum_{j=1}^{N} (R_{D,\Delta}^{(j)})^T v_\Delta^{(j)} = \sum_{j=1}^{N} (R_\Delta^{(j)})^T D^{(j)} v_\Delta^{(j)},$$

   which corresponds to weighted averaging operation. Set $u_\Delta^{(i)} = w_\Delta^{(i)}$.

2. Compute the difference term $q_\Delta = (I_\Delta - R_\Delta R_{D,\Delta}^T)v_\Delta$ as

   $$q_\Delta^{(i)} = v_\Delta^{(i)} - w_\Delta^{(i)}$$

3. Apply $A_{II}^{-1}A_{I\Delta}$ by solving

   $$\begin{pmatrix} U_{A_{II}}^{(i)} & L_{A_{II}^{(i)}}^{-1}A_{I\Delta}^{(i)} \\ & I_\Delta^{(i)} \end{pmatrix} \begin{pmatrix} u_I^{(i)} \\ * \end{pmatrix} = \begin{pmatrix} 0 \\ -q_\Delta^{(i)} \end{pmatrix}$$

   the solution is given by

   $$u_I^{(i)} = (A_{II}^{(i)})^{-1}A_{I\Delta}^{(i)}q_\Delta^{(i)}.$$

   The resulting vector is $u = \tilde{\mathcal{H}}\tilde{v}$.

B.3 Application of Inexact $(\tilde{\mathcal{H}}^*)^T$

Consider the application of the extension operator $(\tilde{\mathcal{H}}^*)^T : V \to \tilde{V}$

$$(\tilde{\mathcal{H}}^*)^T = \begin{pmatrix} I & \\ J_{2,D,\Gamma}\tilde{A}_{\Gamma I}A_{II}^{-1} & \tilde{R}_{D,\Gamma} \end{pmatrix} = \begin{pmatrix} I_I & & \\ (I_\Delta - R_{D,\Delta}R_\Delta^T)A_{\Delta I}A_{II}^{-1} & R_{D,\Delta} & \\ & & I_\Pi \end{pmatrix}.$$

The application of $(\tilde{\mathcal{H}}^*)^T$ to create $\tilde{u} = (\tilde{\mathcal{H}}^*)^T v$, $\tilde{u} \in \tilde{V}$, $v \in V$, is performed as follows

1. Compute effect of $A_{\Delta I}A_{II}^{-1}$ by solving

$$\begin{pmatrix} L_{A_{II}^{(i)}} & \\ A_{\Delta I}^{(i)}U^{-1}_{A_{II}^{(i)}} & I_\Delta^{(i)} \end{pmatrix} \begin{pmatrix} * \\ w_\Delta^{(i)} \end{pmatrix} = \begin{pmatrix} -v_I^{(i)} \\ 0_\Delta \end{pmatrix}$$

   The solution is given by

$$w_\Delta^{(i)} = A_{\Delta I}^{(i)}(A_{II}^{(i)})^{-1}v_I^{(i)}.$$

2. Compute $q_\Delta = R_{D,\Delta}R_\Delta^T w_\Delta$ in parallel as

$$q_\Delta^{(i)} = D_\Delta^{(i)}R_\Delta^{(i)}\sum_{j=1}^{N}(R_\Delta^{(j)})^T w_\Delta^{(j)},$$

   which corresponds to summing the dual variables and weighting them by $D_\Delta^{(i)}$.

3. Compute $u_\Delta = (I_\Delta - R_{D,\Delta}R_\Delta^T)w_\Delta + R_{D,\Delta}v_\Delta$

$$u_\Delta^{(i)} = w_\Delta^{(i)} - q_\Delta^{(i)} + D^{(i)}v_\Delta^{(i)}$$

   The resulting vector is $\tilde{u} = (\tilde{\mathcal{H}}^*)^T v$.

Bibliography

[1] Yves Achdou, Patrick Le Tallec, Frederic Nataf, and Marina Vidrascu. A domain decomposition preconditioner for an advection-diffusion problem. *Computer Methods in Applied Mechanics and Engineering*, 184:145–170, 2000.

[2] M. J. Aftosmis, M. J. Berger, and G. Adomavicius. A parallel multilevel method for adaptively refined cartesian grids with embedded boundaries. AIAA Paper 2000-0808, 2000.

[3] Patrick R. Amestoy, Timothy A. Davis, and Iain S. Duff. An approximate minimum degree ordering algorithm. *SIAM J. Matrix Anal. Appl.*, 17(4):886–905, 1996.

[4] W. Anderson, R. Rausch, and D. Bonhaus. Implicit multigrid algorithms for incompressible turbulent flows on unstructured grids. Number 95-1740-CP. In Proceedings of the 12th AIAA CFD Conference, San Diego CA, 1995.

[5] W. K. Anderson, W. D. Gropp, D. K. Kaushik, D. E. Keyes, and B. F. Smith. Achieving high sustained performance in an unstructured mesh CFD application. Proceedings of SC99, Portland, OR, November 1999, 1999.

[6] Thomas Apel and Gert Lube. Anisotropic mesh refinement in stabilized Galerkin methods. *Numer. Math.*, 74:261–282, 1996.

[7] I. Babuska, B. A. Szabo, and I. N. Katz. The p-version of the finite element method. *SIAM Journal on Numerical Analysis*, 18(3):515–545, 1981.

[8] Timothy J. Barth. Numerical methods for gasdynamic systems on unstructured meshes. In D. Kroner, M. Olhberger, and C. Rohde, editors, *An Introduction to Recent Developments in Theory and Numerics for Conservation Laws*, pages 195 – 282. Springer-Verlag, 1999.

[9] Michele Benzi, Daniel B. Szyld, and Arno van Duin. Orderings for incomplete factorization preconditioning of nonsymmetric problems. *SIAM Journal on Scientific Computing*, 20(5):1652–1670, 1999.

[10] Manoj Bhardwaj, David Day, Charbel Farhat, Michel Lesoinne, Kendall Pierson, and Daniel Rixen. Application of the FETI method to ASCI problems - scalability

results on 1000 processors and discussion on highly heterogeneous problems. *Int. J. Numer. Meth. Engng*, 47:513–535, 2000.

[11] Manoj Bhardwaj, Kendall Pierson, Garth Reese, Tim Walsh, David Day, Ken Alvin, James Peery, Charbel Farhat, and Michel Lesoinne. Salinas: A scalable software for high-performance structural and solid mechanics simulations. In *In proceedings of the 2002 ACM/IEEE conference on supercomputing*, Baltimore, Maryland, 2002.

[12] Jean-Francois Bourgat, Roland Glowinski, Patrick Le Tallec, and Marina Vidrascu. Variational formulation and algorithm for trace operator in domain decomposition calculations. In Tony Chan, Roland Glowinski, Jacques Periaux, and Olof Widlund, editors, *Domain decomposition methods. Second international symposium on domain decomposition methods*, pages 3–16. SIAM, 1988.

[13] J. H. Bramble, J. E. Pasciak, and A. H. Schatz. The construction of preconditioners for elliptic problems by substructuring. i. *Mathematics of Computation*, 47(175):103–134, July 1986.

[14] J. H. Bramble, J. E. Pasciak, and A. H. Schatz. The construction of preconditioners for elliptic problems by substructuring. ii. *Mathematics of Computation*, 49(179):1–16, July 1987.

[15] J. H. Bramble, J. E. Pasciak, and A. H. Schatz. The construction of preconditioners for elliptic problems by substructuring. iii. *Mathematics of Computation*, 51(184):415–430, October 1988.

[16] J. H. Bramble, J. E. Pasciak, and A. H. Schatz. The construction of preconditioners for elliptic problems by substructuring. iv. *Mathematics of Computation*, 53(187):1–24, July 1989.

[17] Susanne C. Brenner and L. Ridgway Scott. *The Mathematical Thoery of Finite Element Methods, Third Edition*. Springer, New York, 2008.

[18] Susanne C. Brenner and Li yeng Sung. BDDC and FETI-DP without matrices or vectors. *Comput. Methods Appl. Mech. Engrg.*, 196:1429–1435, 2007.

[19] F. Brezzi, L. P. Franca, T. J. R. Hughes, and A. Russo. $b = \int g$. *Comput. Methods Appl. Mech. Engrg.*, 145:329–339, 1997.

[20] Franco Brezzi, Marie-Odile Bristeau, and Leopoldo P. Franca. A relationship between stabilized finite element methods and the Galekin method with bubble functions. *Comput. Methods Appl. Mech. Engrg.*, 96:117–129, 1992.

[21] Alexander N. Brooks and Thomas 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*, 32:199–259, 1982.

[22] Timothy A. Davis. Algorithm 832: UMFPACK V4.3 - An unsymmetric-pattern multifrontal method. *ACM Transactions on mathematical software*, 30(2):196–199, 2004.

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

[24] Laslo 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.

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

[26] Clark R. Dohrmann. An approximate BDDC preconditioner. *Numerical Linear Algebra with applications*, 14:149–168, 2007.

[27] Maksymilian Dryja. Schwarz methods of Neumann-Neumann type for three-dimensional elliptic finite element problems. *Communications on pure and applied mathematics*, 48:121–155, 1995.

[28] Charbel Farhat, Michel Lesoinne, Patrick LeTallec, Kendall Pierson, and Daniel Rixen. FETI-DP: a dual-primal unified FETI method - part I: A faster alternative to the two-level FETI method. *International journal for numerical methods in engineering*, 50:1523–1544, 2001.

[29] Krzysztof J. Fidkowski. A high-order discontinuous Galerkin multigrid solver for aerodynamic applications. Masters thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, June 2004.

[30] Leopoldo P. Franca, Sergio L. Frey, and Thomas J. R. Hughes. Stabilized finite element methods: I. Application to the advective-diffusive model. *Comput. Methods Appl. Mech. Engrg.*, 95:253–276, 1992.

[31] Isaac Harari and Thomas J. R. Hughes. What are c and h? : Inequalities for the analysis and design of finite element methods. *Comput. Methods Appl. Mech. Engrg.*, 97:157–192, 1992.

[32] Amiram Harten. On the symmetric form of systems of conservation laws with entropy. *Journal of computational physics*, 49:151–164, 1983.

[33] Thomas J. R. Hughes. A simple shcme for developing upwind finite elements. *International journal for numerical methods in engineering*, 12:1359–1365, 1978.

[34] Thomas 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.*, 127:387–401, 1995.

[35] Thomas J. R. Hughes, Gonzalo R. Feijoo, Luca Mazzei, and Jean-Baptiste Quincy. The variational multiscale method - a paradigm for computational mechanics. *Comput. Methods Appl. Mech. Engrg.*, 166:3–24, 1998.

[36] Thomas J. R. Hughes, L. P. Franca, and 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.*, 73:173–189, 1989.

[37] Thomas J. R. Hughes, L.P. Franca, and 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.*, 54:223–234, 1986.

[38] Thomas J. R. Hughes and 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.*, 58:305–328, 1986.

[39] Thomas J. R. Hughes and M. Mallet. A new finite element formulation for computational fluid dynamics: IV A discontinuity capturing operator for multidimensional advective-diffusive systems. *Comput. Methods Appl. Mech. Engrg.*, 58:329–336, 1986.

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

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

[42] Thomas J.R. Hughes, Gerald Engel, Luca Mazzei, and Mats G. Larson. The continuous Galerkin method is locally conservative. *Journal of Computational Physics*, 163:467–488, 2000.

[43] A. Jameson. Solution of the Euler equations for two-dimensional transonic flow by a multigrid method. *Applied Mathematics and Computation*, 13:327–356, 1983.

[44] Claes Johnson, Uno Navert, and Juhani Pitkaranta. Finite element methods for linear hyperbolic problmes. *Comput. Methods Appl. Mech. Engrg.*, 45:285–312, 1984.

[45] Claes Johnson, Anders Szepessy, and Peter Hansbo. On the convergence of shock-capturing streamline diffusion finite element methods for hyperbolic conservation laws. *Math. Comp.*, 54(189):107–129, 1990.

[46] George Karypis. Parmetis: Parallel graph partitioning and sparse matrix ordering library, 2006. http://glaros.dtc.umn.edu/gkhome/views/metis/parmetis.

[47] D. W. Kelly, S. Nakazawa, and O. C. Zienkiewicz. A note on upwinding and anisotropic balancing dissipation in finite element approximations to convection diffusion problems. *International journal for numerical methods in engineering*, 15:1705–1711, 1980.

[48] Axel Klawonn, Luca F. Pavarino, and Oliver Rheinbach. Spectral element FETI-DP and BDDC preconditioners with multi-element subdomains. *Comput. Methods Appl. Mech. Engrg.*, 198:511–523, 2008.

[49] Axel Klawonn and Olof B. Widlund. Dual-primal FETI methods for linear elasticity. *Communications on Pure and Applied Mathematics*, 59:1523–1572, 2006.

[50] Axel Klawonn, Olof B. Widlund, and Maksymilian Dryja. Dual-primal FETI methods for three-dimensional elliptic problems with heterogeneous coefficients. *SIAM Journal of Numerical Analysis*, 43(1):159–179, 2002.

[51] Jing Li and Olof B. Widlund. FETI-DP, BDDC, and block Cholesky methods. *International Journal for Numerical Methods in Engineering*, 66:250–271, 2006.

[52] Jing Li and Olof B. Widlund. On the use of inexact subdomain solvers for BDDC algorithms. *Computational Methods in Applied Mechanics and Engineering*, 196:1415–1428, 2007.

[53] Jan Mandel. Balancing domain decomposition. *Communications in Numerical Methods in Engineering*, 9:233–241, 1993.

[54] Jan Mandel and Marian Brezina. Balancing domain decomposition for problems with large jumps in coefficients. *Mathematics of Computation*, 65(216):1387–1401, 1996.

[55] Jan Mandel and Clark R. Dohrmann. Convergence of a balancing domain decomposition by constraints and energy minimization. *Numerical Linear Algebra with Applications*, 10:639–659, 2003.

[56] Jan Mandel, Clark R. Dohrmann, and Radek Tezaur. An algebraic theory for primal and dual substructuring methods by constraints. *Applied Numerical Mathematics*, 54:167–193, 2005.

[57] D. J. Mavriplis. Multigrid strategies for viscous flow solvers on anisotropic unstructured meshes. *Journal of Computational Physics*, 145:141–165, 1998.

[58] D. J. Mavriplis. An assessment of linear versus nonlinear multigrid methods for unstructured mesh solvers. *Journal of Computational Physics*, 175:302–325, 2001.

[59] Dimitri J. Mavriplis. Large-scale parallel viscous flow computations using an unstructured multigrid algorithm. ICASE XSReport TR-99-44, 1999.

[60] Dimitri J. Mavriplis, Michael J. Aftosmis, and Marsha Berger. High resolution aerospace applications using the NASA Columbia supercomputer. In proceedings of the 2005 ACM/IEEE conference on supercomputing, 2005.

[61] Dimitri. J. Mavriplis, David Darmofal, David Keyes, and Mark Turner. Petaflops opportunities for the NASA fundamental aeronautics program. AIAA Paper 2007-4084, 2007.

[62] Hans Meuer, Erich Strohmaier, Jack Dongarra, and Horst Simon. Top500 supercomputer sites, 2009.

[63] Stefano Micheletti, Simona Perotto, and Marco 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):1131–1162, 2003.

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

[65] Tolulope 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.

[66] Per-Olof Persson. Scalable parallel Newton-Krylov solvers for discontinuous Galerkin discretizations. AIAA Paper 2009-606, 2009.

[67] Per-Olof Persson and Jaime Peraire. Newton-GMRES preconditioning for Discontinuous Galerkin discretizations of the Navier-Stokes equations. *SIAM Journal on Scientific Computing*, 30(6):2709–2722, 2008.

[68] Per-Olof Persson and Gilbert Strang. A simple mesh generator in MATLAB. *SIAM review*, 46(2):329–345, 2004.

[69] Niles A. Pierce and Michael B. Giles. Preconditioned multigrid methods for compressible flow calculations on stretched meshes. *Journal of Computational Physics*, 136:425–445, 1997.

[70] Kendall H. Pierson, Garth M. Reese, Manoj K. Bhardwaj, Timothy F. Walsh, and David M. Day. Experiences with FETI-DP in a production level finite element application. Technical Report SAND2002-1371, Sandia National Laboratories, 2002.

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

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

[73] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, 1996.

[74] M. A. Taylor, B. A. Wingate, and R. E. Vincent. An algorithm for computing fekete points in the triangles. *SIAM J. Numer. Anal.*, 38(5):1707–1720, 2000.

[75] Andrea Toselli. FETI domain decomposition methods for scalar advection-diffusion problems. *Computer Methods in Applied Mechanics and Engineering*, 190:5759–5776, 2001.

[76] Andrea Toselli and Olof Widlund. *Domain Decomposition Methods Algorithm and Theory*. Springer-Verlag, 2005.

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

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