SIAM J. SCI. COMPUT.
Vol. 38, No. 5, pp. C504–C532

A MATRIX DEPENDENT/ALGEBRAIC MULTIGRID APPROACH FOR EXTRUDED MESHES WITH APPLICATIONS TO ICE SHEET MODELING*

R. TUMINARO†, M. PEREGO†, I. TEZAUR†, A. SALINGER†, AND S. PRICE‡

Abstract. A multigrid method is proposed that combines ideas from matrix dependent multigrid for structured grids and algebraic multigrid for unstructured grids. It targets problems where a three-dimensional mesh can be viewed as an extrusion of a two-dimensional, unstructured mesh in a third dimension. Our motivation comes from the modeling of thin structures via finite elements and, more specifically, the modeling of ice sheets. Extruded meshes are relatively common for thin structures and often give rise to anisotropic problems when the *thin* direction mesh spacing is much smaller than the *broad* direction mesh spacing. Within our approach, the first few multigrid hierarchy levels are obtained by applying matrix dependent multigrid to semicoarsen in a structured thin direction fashion. After sufficient structured coarsening, the resulting mesh contains only a single layer corresponding to a two-dimensional, unstructured mesh. Algebraic multigrid can then be employed in a standard manner to create further coarse levels, as the anisotropic phenomena is no longer present in the single layer problem. The overall approach remains fully algebraic, with the minor exception that some additional information is needed to determine the extruded direction. This facilitates integration of the solver with a variety of different extruded mesh applications.

Key words. ice sheets, iterative solvers, algebraic method

AMS subject classifications. 65F08, 65Y05, 65M55, 65N55

DOI. 10.1137/15M1040839

1. Introduction.

The 2007 Intergovernmental Panel on Climate Change (IPCC) produced a highly influential 4th Assessment Report (AR4) heightening awareness on the importance of improved modeling of ice sheet dynamics [44], particularly for providing better estimates of potential future sea-level rise from large ice sheets. The report highlighted the inability of (the then) current ice sheet models to accurately explain observed phenomena, such as the acceleration and thinning of several large outlet glaciers located in Greenland. While many improvements have occurred in ice sheet models since that time, the IPCC's more recent 5th Assessment Report [45] continues to stress the importance of improved ice sheet modeling frameworks for reducing uncertainties in projections of future sea-level rise. Since the AR4, a number

*Submitted to the journal's Software and High-Performance Computing section September 22, 2015; accepted for publication (in revised form) July 1, 2016; published electronically October 6, 2016.

http://www.siam.org/journals/sisc/38-5/M104083.html

Funding: Support was provided by the Scientific Discovery through Advanced Computing (SciDAC) program funded by the U.S. Department of Energy (DOE), Office of Science, Advanced Scientific Computing Research, and Biological and Environmental Research. Support was also provided by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research, Applied Mathematics program.

†Sandia National Laboratories, Albuquerque, NM 87185 (rstumin@sandia.gov, mperego@sandia.gov, ikalash@sandia.gov, agsalin@sandia.gov). Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the National Nuclear Security Administration of the U.S. Department of Energy under contract DE-AC04-94AL85000.

‡Los Alamos National Laboratory, Los Alamos, NM 87545 (sprice@lanl.gov). Los Alamos National Laboratory is operated by Los Alamos National Security, LLC, for the National Nuclear Security Administration of the U.S. Department of Energy under contract DE-AC52-06NA25396.

of "next generation" ice sheet models [26, 25, 9, 23, 16, 7, 10, 46] have been developed. Many of these models include features such as unstructured and/or adaptive meshes at subkilometer resolutions, which were not common in prior models. Current simulations already require thousands of processing units on high-performance computing systems. Most recently, sophisticated numerical techniques are beginning to be explored in areas such as optimization, data assimilation, and uncertainty quantification to more rigorously treat the numerous unknown model inputs. It is clear that further algorithmic advances are required for continued progress on accurate modeling of ice sheets, especially when considering the large, multidimensional, poorly constrained, or unknown sets of physical parameters they require as inputs (see, e.g., [20]).

This paper focuses on solving linear systems associated with ice sheet modeling, as frequently the solution of these systems is the most time consuming component within larger ice sheet simulations, requiring that numerous linear systems be solved within each Newton and time step of the model. The most prominent challenge associated with the linear systems is the anisotropic nature of the problem. Anisotropic problems often lead to iterative method difficulties, as weak coupling directions give rise to oscillatory modes with small eigenvalues and poorly conditioned linear systems. For example, a standard five-point discretization of $\epsilon u_{xx} + u_{yy}$ on an $n \times n$ uniform mesh gives a matrix with eigenvalues

$$4(n+1)^2 \left[\epsilon \sin(\pi i/(2n+2))^2 + \sin(\pi j/(2n+2))^2\right], \qquad i, j = 1, \ldots, n,$$

which yields many relatively small eigenvalues (for *all* values of i and small values of j) when ϵ is small. In our case, anisotropy comes from the thin vertical scale of the domain relative to the horizontal scale, typically resulting in elements or grid cells with high aspect ratios. The highly anisotropic meshes lead to discretization matrices, where terms associated with vertical coupling/differentiation are much larger in magnitude than horizontal coupling/differentiation terms. The use of semicoarsening to address anisotropic problems is now well known in the multigrid literature [38, 14, 8, 43]. Algebraic multigrid (AMG) attempts to mimic semicoarsening ideas using notions of strong and weak matrix connections. While this approach is often effective, the performance of a smoothed aggregation AMG (SA-AMG) solver [49] was disappointing on the corresponding finite element matrices generated by our ice sheet model based on the first-order Stokes equations [46]. To address this, a new multigrid solver is proposed that combines matrix dependent multigrid ideas, semicoarsening, and algebraic multigrid. The new algorithm takes advantage of the fact that almost all ice sheet modeling codes employ extruded meshes. These extruded three-dimensional (3D) meshes logically correspond to a tensor product of a two-dimensional (2D) unstructured mesh with a one-dimensional (1D) mesh in the z direction. The new algorithm first applies structured multigrid techniques to coarsen in the z direction only, thereby creating a hierarchy of meshes that are identical in the horizontal direction. To do this, matrix dependent multigrid ideas are generalized to cases where two of the three spatial dimensions are unstructured. Additionally, an aggressive coarsening version of the semicoarsening algorithm is developed so that the ratio of the number of unknowns between consecutive levels is much greater than two. This is often advantageous on high-performance computing systems, as parallel efficiency sometimes suffers on multigrid hierarchies with many levels. When the resulting coarse mesh contains only one vertical layer, further coarsening is applied in the horizontal direction using standard AMG techniques. Thus, the entire multigrid hierarchy is built by matrix dependent grid transfers on finer grids and AMG transfers on coarser grids.

Extruded meshes occur frequently in geophysical problems (e.g., atmospheric and oceanic circulation, oil/gas modeling). Multigrid algorithms have been proposed that take advantage of the extruded direction [3, 34, 14], but most of these efforts focus on geometric multigrid, where considerations are somewhat different. In particular, these algorithms employ line relaxation in the vertical (extruded) direction as we do but perform coarsening only in the horizontal (nonextruded) direction. The line relaxation provides sufficient smoothing so that horizontal coarsening can be effective. In the algebraic context, initial horizontal coarsening is cumbersome, as it requires restructuring an existing AMG algorithm and code. Specifically, all grid layers must be coarsened consistently so that vertical lines are preserved on all hierarchy levels (and then line relaxation can be employed at all levels). Furthermore, generation of the grid transfers would also require suitable modifications. For example, SA-AMG effectively applies a point Jacobi algorithm to smooth grid transfer basis functions. However, adaptation of this algorithm would be needed, as the horizontal weak coupling complicates the smoothing of grid transfer basis functions in the horizontal direction. Such adaptations might be possible, but they are fairly intrusive and require care to avoid large fill-in within the grid transfer stencil. These concerns are not considerations within a geometric multigrid algorithm.

The algorithm proposed here is fully algebraic, which facilitates its integration into a variety of different application simulations. In the context of AMG, two recent papers follow a somewhat similar notion of coarsening first in the vertical direction. In [21], a more standard smoothed aggregation method is considered but first restricting all coarsening to the vertical direction. This was shown to significantly enhance the performance of the method over standard isotropic coarsening.[1] A matrix dependent multigrid is also considered in [22] based on the idea of semicoarsening in an extruded direction. This proposed method can be viewed as an extension of Schaffer's algorithm [38] for structured grids. The nonzero pattern of the interpolation operators is identical to those in our matrix dependent/AMG hybrid when the coarse grid is defined by every even plane (orthogonal to the extruded direction). However, the actual interpolation coefficients are computed in a very different fashion and require solutions to the plane problems, which can be problematic in a parallel setting. Plane relaxation is also advocated for the smoothing phase of the algorithm. Furthermore, the algorithm in [22] does not allow for aggressive coarsening, which is highly desirable in a parallel setting.

A secondary contribution of this paper is to clarify issues and misconceptions associated with linear systems arising from ice sheet modeling. For example, there are situations in which the underlying linear systems are not difficult to solve. We discuss this phenomenon in the context of Green's functions and thin domains for nonsliding ice. ILU preconditioners are also touched upon, as they are commonly employed in this area. Finally, large ice shelves—extensive regions of floating ice originating from and connected to land-based ice sheets, most common in Antarctica—can generate problematic linear systems. Here, the associated Green's function is very different from that for nonsliding ice at a frozen ice/bed interfacegrounded ice and can lead to singular systems if the mesh (constructed from datasets representing highly irregular coastlines) does not exclude islands and certain thin peninsulas.

Section 2 briefly summarizes the first-order-accurate Stokes approximation model, while section 3 discusses Green's functions for thin domain applications. Section 4 highlights the multigrid challenges associated with anisotropic problems, and

[1] Suggested by R. Tuminaro while preparing the current paper.

section 5 proposes a new hybrid matrix dependent/AMG solver to address these challenges. Section 6 considers difficulties associated with ILU orderings on anisotropic problems and singularities that could arise when meshing irregular coastlines. An algorithm is proposed for detecting and removing these potentially harmful mesh components. In section 7 numerical results are given demonstrating the solver efficacy on our ice sheet modeling simulation. While our motivation and numerical examples are driven by ice sheets, the fully algebraic solver can generally be applied to other anisotropic problems that arise from extruded meshes.

2. Stokes formulation and discretization. Ice sheets can be modeled as an incompressible, power-law viscous fluid in a low-Reynolds number flow. For the solution of the momentum balance equations, a range of different approximations has been considered, including "shallow ice" [19], "shallow-shelf" [29], "hybrid" [9, 35, 17], and "higher-order" approximations [31]. While so-called "full" Stokes models [23, 24, 16] are generally accepted as providing the highest fidelity simulations, in this paper we pursue the first-order-accurate Stokes approximation model [31, 2, 15] (also commonly referred to as the "Blatter–Pattyn" approximation) discussed in [46]. This model is fully 3D and sufficiently accurate for simulating the flow over most parts of an ice sheet while being considerably less computationally expensive than simulating the full nonlinear Stokes equations. The corresponding partial differential equations (PDEs) are given by the following elliptic system:

$$\begin{cases} -\nabla \cdot (2\mu \dot{\boldsymbol{\epsilon}}_1) + \rho g \frac{\partial s}{\partial x} &= 0, \\ -\nabla \cdot (2\mu \dot{\boldsymbol{\epsilon}}_2) + \rho g \frac{\partial s}{\partial y} &= 0. \end{cases} \tag{2.1}$$

The $\boldsymbol{\epsilon}_i$ are first-order approximations to the effective strain rate tensors,

$$\dot{\boldsymbol{\epsilon}}_1^T = \begin{pmatrix} 2\dot{\epsilon}_{xx} + \dot{\epsilon}_{yy}, & \dot{\epsilon}_{xy}, & \dot{\epsilon}_{xz} \end{pmatrix} \text{ and } \dot{\boldsymbol{\epsilon}}_2^T = \begin{pmatrix} \dot{\epsilon}_{xy}, & \dot{\epsilon}_{xx} + 2\dot{\epsilon}_{yy}, & \dot{\epsilon}_{yz} \end{pmatrix}, \tag{2.2}$$

where

$$\dot{\epsilon}_{xx} = \frac{\partial u}{\partial x}, \quad \dot{\epsilon}_{yy} = \frac{\partial v}{\partial y}, \quad \dot{\epsilon}_{xy} = \frac{1}{2}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right), \quad \dot{\epsilon}_{xz} = \frac{1}{2}\frac{\partial u}{\partial z}, \quad \dot{\epsilon}_{yz} = \frac{1}{2}\frac{\partial v}{\partial z}. \tag{2.3}$$

One is interested in solving for u and v, the respective x and y components of the ice velocity. In (2.1), g is gravitational acceleration, ρ is ice density, and $s \equiv s(x, y)$ denotes the upper boundary surface. Nonlinearity arises from the "effective" viscosity, which is approximated by

$$\mu = \frac{1}{2} A^{-\frac{1}{n}} \dot{\epsilon}_e^{-\frac{2}{n}} \tag{2.4}$$

using Glen's law [12, 30] to model the ice rheology. Here, $\dot{\epsilon}_e$ is the effective strain rate given by

$$\dot{\epsilon}_e^2 \equiv \dot{\epsilon}_{xx}^2 + \dot{\epsilon}_{yy}^2 + \dot{\epsilon}_{xx}\dot{\epsilon}_{yy} + \dot{\epsilon}_{xy}^2 + \dot{\epsilon}_{xz}^2 + \dot{\epsilon}_{yz}^2, \tag{2.5}$$

and A is the strongly temperature dependent flow law rate factor that can be described through an Arrhenius relation [12]. The exponent n typically takes values between 1 and 4. In this work we take $n = 3$, as commonly done in the literature. Notice that there exists a 3D null space of (2.1) when boundary conditions are not imposed, which corresponds to rigid body modes. This space is spanned by functions associated with

horizontal translations ($u = 1$, $v = 0$) and ($u = 0$, $v = 1$) and rotation ($u = -y$, $v = x$); see, e.g., [39].

This first-order Stokes model is a simplification of the nonlinear Stokes flow equations [15, 40]. It is derived under an assumption of a small aspect ratio, δ, between characteristic length scales for the vertical and horizontal dimensions and the assumption that normal vectors to the upper and lower surfaces are nearly vertical. These assumptions allow one to effectively neglect $\mathcal{O}(\delta^2)$ terms in the Stokes equations and to treat the pressure as hydrostatic (see, e.g., [15] for details) Boundary conditions are required to complete the formulation. A homogeneous Neumann condition is prescribed on the upper boundary, i.e., $\dot{\boldsymbol{\epsilon}}_1 \cdot \mathbf{n} = \dot{\boldsymbol{\epsilon}}_2 \cdot \mathbf{n} = 0$, where $\mathbf{n}$ is the outward facing normal vector to the upper surface. A Robin condition is used on the lower boundary,

$$2\mu\dot{\boldsymbol{\epsilon}}_1 \cdot \mathbf{n} + \beta u = 0, \quad 2\mu\dot{\boldsymbol{\epsilon}}_2 \cdot \mathbf{n} + \beta v = 0, \tag{2.6}$$

where $\beta \equiv \beta(x, y) \geq 0$ is the basal sliding (or friction) coefficient, which in this paper can be viewed as an already known field.[2] Large β (e.g., $\beta = 10^4$ kPa yr m^{-1}) corresponds to a quasi–no-slip condition, while small β implies a weak frictional force corresponding to a thawed ice/bed interface that allows for some degree of slip tangential to the bedrock. Under floating ice shelves, β is often taken identically equal to zero, corresponding to free slip over a frictionless boundary. On the lateral boundary, the following dynamic Neumann condition (referred to as "open-ocean" or "floating ice") is used in this paper:

$$\begin{aligned} 2\mu\dot{\boldsymbol{\epsilon}}_1 \cdot \mathbf{n} - \rho g(s - z)\mathbf{n} &= \rho_w g \max(z, 0)\mathbf{n}, \\ 2\mu\dot{\boldsymbol{\epsilon}}_2 \cdot \mathbf{n} - \rho g(s - z)\mathbf{n} &= \rho_w g \max(z, 0)\mathbf{n}, \end{aligned} \tag{2.7}$$

where ρ_w denotes the density of water and z is the elevation above sea level. This condition is derived under a hydrostatic equilibrium assumption between the ice shelf and the air (or water) that surrounds it [27].

This first-order-accurate Stokes model is then discretized with a Galerkin finite element method using bilinear or trilinear basis functions on either hexahedral or tetrahedral elements. A combination of Newton's method and continuation (on μ) generates a sequence of linear systems, the efficient solution of which is the main focus of this paper. We omit further details of the underlying discretization and refer interested readers to [46] and [47].

3. Locality and thin structures. Before developing solution techniques, the character of elliptic systems on thin domains is examined with an eye towards understanding the potential of local preconditioners. To do this, consider the PDE

$$\begin{aligned} u_{xx} + u_{yy} &= f, \qquad 0 < x < 1, \quad 0 < y < \epsilon, \\ u_y(x, 0) = u_y(x, \epsilon) &= 0, \qquad u(0, y) = u(1, y) = 0, \end{aligned} \tag{3.1}$$

where $\epsilon \ll 1$, and $f(x, y) = \delta(1/2, \epsilon/2)$ is the Dirac delta function. Thus, Neumann conditions are imposed on the top/bottom boundaries (which are our focus here). Dirichlet conditions hold on the left/right boundaries, and the value of ϵ determines

[2] In practice, β is obtained through an inverse problem formulation to *best* match measurements while retaining certain desirable properties [33].

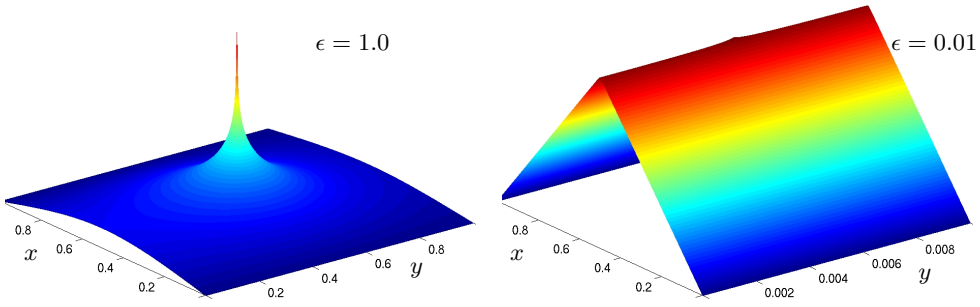

FIG. 3.1. *Solution with two Neumann and two Dirichlet boundary conditions for two ϵ values.*

the degree to which the domain is thin. In this case, the analytic solution (see page 435 of [28]) is

$$u(x,y) = \frac{1}{2\pi} \ln \frac{|1 - e^{\pi(y+ix-\epsilon/2+i/2)}||1 - e^{\pi(y+ix+\epsilon/2+i/2)}|}{|1 - e^{\pi(y+ix-\epsilon/2-i/2)}||1 - e^{\pi(y+ix+\epsilon/2-i/2)}|} + 2\sum_{n=1}^{\infty} \frac{\cosh(n\pi y)\cosh(n\pi\frac{\epsilon}{2})\sin(n\pi x)\sin(n\frac{\pi}{2})}{n^2\pi^2 e^{n\pi\epsilon}\sinh(n\pi\epsilon)} \tag{3.2}$$

that is depicted in Figure 3.1 for $\epsilon = 1.0$ (left) and $\epsilon = 0.01$ (right). For the square domain, one sees a typical Green's function decay associated with 2D Laplace operators. For the thin strip, the decay is much less rapid. In particular, the function is nearly constant in the thin direction, while in the thick direction it resembles a 1D Green's function for the Laplace equation (which decays linearly across the entire domain). The key points are that the solution within regions far from the central point is still significant even though it decays noticeably for the leftmost plot and that the decay of the solution away from the Dirac delta function is particularly modest for the thin domain case when Neumann conditions are applied to the top and bottom surfaces.

Now replace the bottom boundary condition with a Dirichlet condition by considering

$$u_y(x,\epsilon) = 0 \qquad \text{and} \qquad u(x,0) = u(0,y) = u(1,y) = 0. \tag{3.3}$$

In this case, the solution (see page 435 of [28]) is

$$u(x,y) = \frac{1}{2\pi} \ln \frac{|1 - e^{\pi(y+ix-\epsilon/2+i/2)}||1 - e^{\pi(y+ix+\epsilon/2-i/2)}|}{|1 - e^{\pi(y+ix-\epsilon/2-i/2)}||1 - e^{\pi(y+ix+\epsilon/2+i/2)}|} + 2\sum_{n=1}^{\infty} \frac{\sinh(n\pi y)\sinh(n\pi\frac{\epsilon}{2})\sin(n\pi x)\sin(n\frac{\pi}{2})}{n\pi e^{n\pi\epsilon}\cosh(n\pi\epsilon)}. \tag{3.4}$$

Figures 3.2 and 3.3 depict solutions for different values of ϵ. One can most easily see the effect of the Neumann and Dirichlet conditions in Figure 3.3. Notice that the solution extends across the entire vertical direction for all values of ϵ. However, the decay in the horizontal direction can be seen in the $\epsilon = 0.1$ case and is indeed quite pronounced for the $\epsilon = 0.01$ situation. That is, the solution character now decays rapidly in the horizontal direction for small values of ϵ, unlike the Figure 3.1 case.

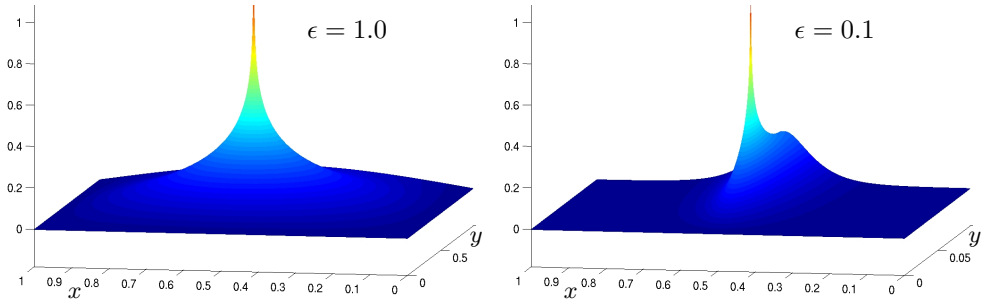

FIG. 3.2. *Solution with Dirichlet (Neumann) condition at $y = 0$ ($y = 1$) for two ϵ values.*

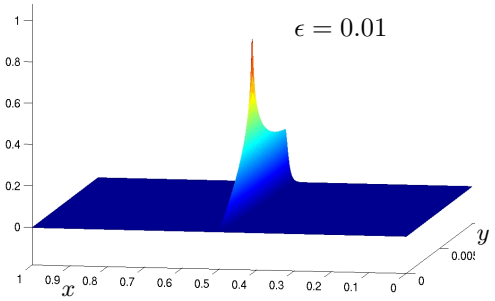

FIG. 3.3. *Solution with Dirichlet (Neumann) condition at $y = 0$ ($y = 1$) for $\epsilon = 0.01$.*

Intuitively, the additional Dirichlet boundary condition pins the Green's function along the bottom surface, inducing rapid decay within the interior.

Green's functions and matrix inverses are closely related. For a PDE discretized by a standard first-order local finite element technique (e.g., using nodal Lagrange basis functions), the global/local nature of the associated Green's functions and the matrix inverse should roughly mirror each other. Specifically, nonzeros within a row have smaller magnitude for columns corresponding to mesh nodes further from the node associated with the diagonal. This matrix decay generally mirrors the continuous Green's function and is more pronounced when the continuous function decay is rapid. As preconditioners approximate matrix inverses, it should not be surprising that a preconditioner with a spatially global character only in the vertical direction might be sufficient for the problem associated with Figure 3.3 but insufficient for both cases in Figure 3.1. This insufficiency can be seen by considering the application of a Krylov solver to a linear system. At the jth iteration, such a solver finds the *best* solution within a Krylov subspace given by

$$\mathcal{K}_j(M^{-1}A, b) = \{b,\ M^{-1}Ab,\ (M^{-1}A)^2 b,\ \ldots,\ (M^{-1}A)^j b\},$$

where A is the discrete matrix, b is the discrete right-hand side, M^{-1} is the left preconditioner, and the initial guess is assumed to be the zero vector. In our example, b has only a few nonzeros to represent a discrete approximation to the Dirac delta function. This implies that if both A and M^{-1} do not globally propagate information, then only a modest number of entries in the first Krylov vectors will be nonzero. That is, a small Krylov space (i.e., few Krylov iterations) *might be sufficient* when M^{-1} globally propagates information vertically to accurately approximate solutions such as

in Figure 3.3. However, a large Krylov space is necessary to capture solutions such as in Figure 3.1. Thus, a large number of Krylov iterations would be needed when using a vertical line Jacobi/Gauss–Seidel preconditioner when both top and bottom boundaries correspond to Neumann conditions. Nonetheless, a line preconditioner could *potentially* be effective when the bottom boundary is given by a Dirichlet condition and the domain is thin vertically.

As previously mentioned for ice sheets, a homogeneous Neumann condition is prescribed on the upper ice surface and a Robin condition is used for the lower ice boundary. The Robin boundary mimics a Dirichlet condition (frozen ice/bed interface or nonsliding ice) for large values of $\beta(x, y)$, while it more closely mimics a Neumann condition (ice shelves or sliding ice) for small $\beta(x, y)$. Thus, depending on the setting, the problem might more closely resemble Figure 3.1 or Figure 3.3. The implications of this section will be revisited in section 6.

4. Anisotropic meshes and linear solvers. Multigrid for anisotropic applications has been heavily studied; see, e.g., [48, 38, 14, 8, 43] and the references therein. Anisotropic phenomena might be present in the PDE coefficients (e.g., material characteristics with strong directional dependence). Anisotropic phenomena might also arise from a highly stretched mesh, which can conceptually be transformed by rescaling coordinate directions so that the mesh no longer appears stretched, but the transformed PDE now has anisotropic coefficients. To examine the solver difficulties, consider the model problem

$$\epsilon u_{xx} + u_{yy} = f, \tag{4.1}$$

where $\epsilon \ll 1$. Here, the dominant direction is oriented along the y axis. This dominant direction is aligned with a regular mesh oriented along the standard coordinate directions. In a 3D thin domain setting, anisotropic phenomena occurs due to the relatively small thin direction mesh spacing. In this case, the dominant direction is in the thin dimension and is aligned when a 2D mesh is extruded in the thin dimension to create a 3D mesh. Generally, the aligned case is much easier for solvers.

From a matrix perspective, the associated discrete operator has relatively small eigenvalues associated with eigenvectors that represent oscillatory functions in weak directions. Application of standard point relaxation methods (such as Gauss–Seidel) do not smooth errors in nondominant directions, as they are generally inefficient at reducing errors associated with small eigenvalues. The left side of Figure 4.1 plots errors after applying 10-point Jacobi sweeps to a five-point finite difference representation of (4.1) on a square domain with Dirichlet boundary conditions and a random initial guess. Errors are smooth (oscillatory) in the y (x) direction.

Two general multigrid strategies have been considered. The first improves the relaxation technique so that errors are smoothed in all directions. This involves some form of line or plane relaxation. The right side of Figure 4.1 plots errors after 10 line Jacobi sweeps for the same model problem, where lines are oriented in the y coordinate direction. Errors are now smooth in all directions, and so standard multigrid coarsening can be effective. The second remedy continues with point relaxation (i.e., left side of Figure 4.1) but only coarsens (or semicoarsens) in directions where smoothing is effective. In our model example, errors in the x direction cannot be well approximated by a mesh that is coarser in the x direction. However, errors can be well approximated by a mesh coarser in the y direction. In the ice sheet context, this corresponds to only vertical coarsening.

Different combinations of line smoothing and semicoarsening have been considered

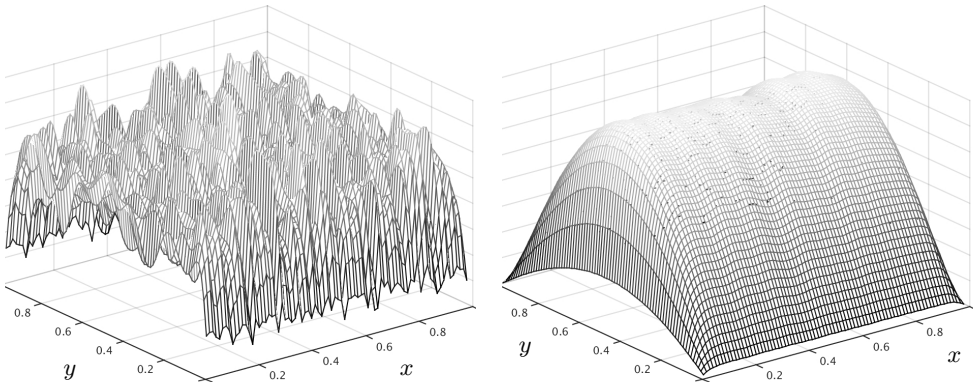

FIG. 4.1. *Solution error after point Jacobi and line Jacobi with* $f = 0$ *and a random initial solution guess on the left and right, respectively. Both smoothers use a damping factor of* 0.7.

for structured grids. For the most part, AMG methods focus exclusively on semicoarsening. This is natural given AMG's attractiveness on fully unstructured meshes. In particular, AMG methods do not generally assume an easily identifiable direction associated with anisotropic phenomena. Thus, line techniques are somewhat awkward. Instead, AMG methods center on automatically detecting anisotropic directions using notions of strength-of-connection. Selective coarsening is then accomplished by altering the matrix graph during the AMG coarsening phase to exclude graph edges that are determined to be weak connections (see, for example, [5, 37, 41, 6]). On model structured grid problems, these selective coarsening techniques mimic the original semicoarsening ideas for structured grids. Of course, it is important to recognize that these algorithms are applicable to unstructured problems, and so they can be applied to semicoarsen in more general ways, as may be needed for complex problems.

In principle, standard AMG semicoarsening techniques could be applied to thin domain problems discretized with finite elements on meshes with large element aspect ratios, e.g., an order of magnitude or more greater than one. The algorithm proposed in this paper is motivated by the extremely disappointing performance of a standard smoothed aggregation multigrid method on our application. As is known, most common AMG strength-of-connection measures are fragile when applied to finite element examples. This is due to a smearing effect of stencil coefficients and is related to the presence of a nondiagonal mass matrix (see [41] for a detailed discussion). Automatic detection is further complicated by nonuniform z spacing, which is a function of the varying ice sheet thicknesses and the relatively large matrix coefficients associated with large β's used for the basal Robin boundary conditions when the ice is well grounded. A new solver is proposed in the next section that avoids these AMG challenges by exploiting the extruded mesh structure.

5. Operator dependent multigrid and semicoarsening.

5.1. Semicoarsened multigrid hierarchy. The proposed AMG algorithm assumes that the underlying PDE mesh is extruded as depicted in Figure 5.1. In particular, this mesh is created by first constructing a 2D grid, which might be completely unstructured, and then extruding this into the third dimension. It is not assumed, however, that the mesh spacing in this third dimension is necessarily uniform in any

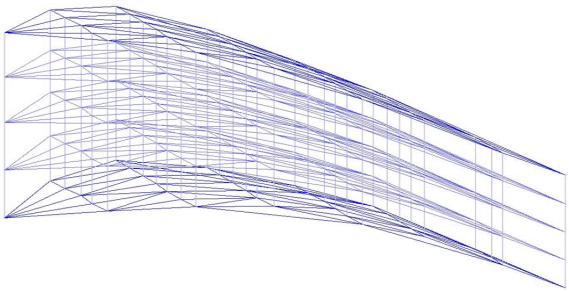

FIG. 5.1. *Sample of an unstructured 2D mesh that is extruded vertically.*

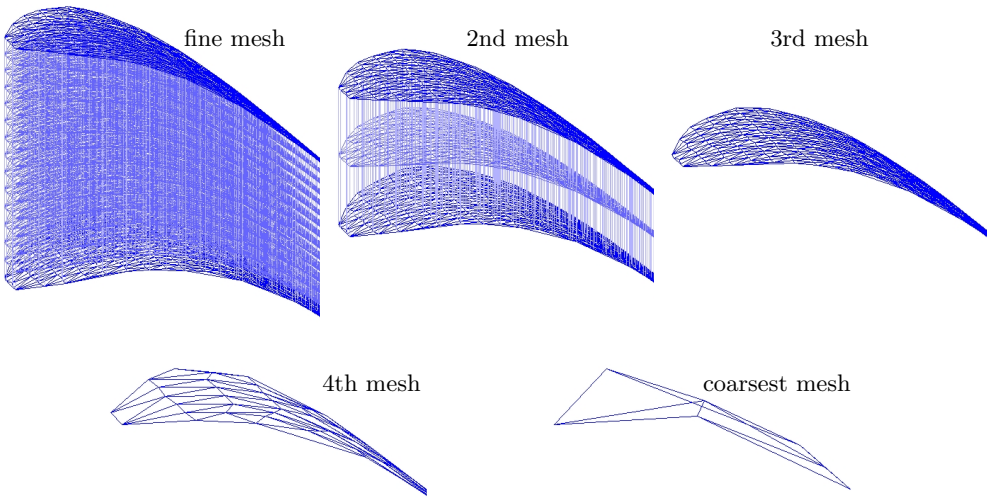

FIG. 5.2. *Sample mesh hierarchy with finest level (top row, left), two semicoarsening created levels (top row, right), and two SA-AMG created levels (bottom row).*

way or that points residing on the same mesh layer have the same z coordinate value. For a number of practical reasons, vertically extruded meshes are commonly employed in ice sheet modeling. One could argue as well that extruded meshes are natural given the underlying assumptions made when deriving the first-order Stokes approximation, which effectively relies on large disparities between vertical and horizontal scales. In addition to ice sheet modeling, extruded meshes are heavily used in other geophysical modeling applications (e.g., atmospheric and oceanic, oil/gas, carbon sequestration) and arise frequently in engineering simulations involving thin structures.

To illustrate our proposed semicoarsening strategy, a sample multigrid hierarchy is depicted in Figure 5.2. As opposed to a standard AMG method that automatically detects coarsening directions, we leverage the extruded nature of the mesh to explicitly prescribe coarsening directions. In particular, finer levels of the multigrid hierarchy are created by regular coarsening in the vertical direction. This regular coarsening corresponds to choosing a subset of layers from the finer mesh to define the next coarser mesh. This is illustrated by the three images in the upper row of Figure 5.2, where three layers are chosen for the second mesh (middle image) and one layer is chosen for the third mesh (rightmost image). Notice that the second mesh has far

fewer layers than the finest mesh (leftmost image); i.e., the ratio of the number of layers on two consecutive levels is much greater than two. We denote this as *aggressive* semicoarsening.

To complete the multilevel solver, AMG is employed after a sufficient problem dimension reduction via semicoarsening. Ideally, this should occur when the matrix equations no longer appear anisotropic. As the thin direction anisotropic phenomena is no longer present, application of standard AMG is no longer problematic. For many thin structure problems, such as ice sheet modeling, this arises when there is only a single layer (or plane) of points remaining. However, the switch to AMG could take place earlier. As AMG coarsening typically does not preserve any mesh structure, it is no longer possible to apply any further semicoarsening once an AMG coarsened level has been created within the hierarchy. The bottom two images in Figure 5.2 are obtained by applying a smoothed aggregation (SA-AMG) coarsening algorithm (plotting a central point of each aggregate). Thus, the entire multigrid hierarchy is generated by a hybrid method. The finer levels employ semicoarsening, and the coarser levels are obtained by a standard AMG approach.

Before concluding this subsection, we note that there are alternative ways to leverage structure in an extruded mesh. One scheme might use vertical line relaxation (similar to the right side of Figure 4.1), so that smoothing is effective in all directions. In this way, more standard multigrid coarsening could be used. Tensor product multigrid [3, 34, 14] adopts this strategy using geometric multigrid to coarsen only in horizontal directions in conjunction with vertical line smoothing. While this idea has attractive features, we do not pursue this approach primarily because it is not easily adapted to a fully algebraic capability. Specifically, it is necessary that the extruded nature of the mesh is maintained on all hierarchy levels to recursively apply the line smoothing. One could consider different ways to adapt an AMG code so that it only coarsens horizontally, but we view this as a fairly intrusive/nontrivial adaptation, especially with smoothed SA-AMG. As is generally known, an AMG approach is desirable when integrating a solver with several different large simulation codes. For this reason, our adaptations center on semicoarsening (or strong coupling) in the vertical direction, which is more typical with fully algebraic methods.

5.2. Matrix dependent grid transfers: One-dimensional case. To construct grid transfers, we propose a matrix dependent semicoarsening strategy that resembles ideas found in [38, 14, 1, 51, 13, 8]. It can be motivated from a physical perspective or by purely algebraic considerations. In this paper, we focus on the algebraic perspective but do make reference to more physical arguments in the discussion.

We begin by recalling matrix dependent multigrid ideas in the purely 1D case. Most standard 1D discretizations give rise to banded linear systems, assuming matrix rows are ordered so that consecutive rows come from neighboring mesh points. Banded systems are easily and effectively solved by direct methods, e.g., the Thomas algorithm for tridiagonal matrices. One direct approach for solving these 1D systems is a generalization of cyclic reduction, which in the 1D case can be viewed as nested dissection. Connections between cyclic reduction and multigrid are well known. Specifically, consider the grid shown in Figure 5.3 with subdomains depicted by rounded rectangles. Discretization of a scalar PDE on this mesh typically leads to a band matrix. For first-order discretizations (e.g., with linear finite element basis functions), the associated matrix would normally have a tridiagonal structure. The matrix can be reordered so that all unknowns inside the rectangles (numbering from

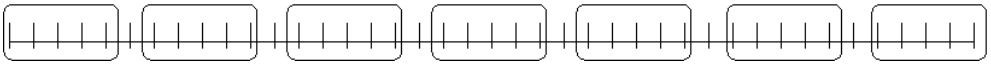

FIG. 5.3. *1D mesh with internal subdomains depicted by rounded rectangles.*

left to right) appear before all nonrectangle unknowns (numbering from left to right). This induces a matrix with the structure

$$A = \begin{pmatrix} D & C \\ G & B \end{pmatrix}, \tag{5.1}$$

where D (B) represents coupling between unknowns within (outside of) rectangles and C and G capture cross couplings between unknowns within rectangles and unknowns outside of rectangles. If the original matrix (before reordering) has a tridiagonal structure, then the Schur complement defined by $S = B - GD^{-1}C$ is also tridiagonal. Thus, the idea can be recursively applied. That is, S's unknowns can be categorized as either belonging to a new set of rectangles or lying outside of rectangles. This new ordering induces another Schur complement operator that is again tridiagonal. In this fashion, one generates a hierarchy of matrices with decreasing dimension. In fact, this process corresponds to a multigrid procedure in which the number of unknowns within rectangles determines the aggressiveness of the multigrid coarsening. The Schur complement matrices are the coarse level discretizations within a multigrid hierarchy, and the grid transfers are defined through the Gaussian elimination procedure. In our example, we have

$$P = \begin{bmatrix} -D^{-1}C \\ I \end{bmatrix} \quad \text{and} \quad R = \begin{bmatrix} -GD^{-1} & I \end{bmatrix}, \tag{5.2}$$

where P is the interpolation operator to the finest grid and R is the restriction operator from the finest grid. Notice that the product AP is identically zero for all rows corresponding to rectangle interiors. That is, the interpolation basis functions satisfy the discrete PDE operator within the rectangles. This essentially implies that the energy of the interpolation basis functions (as measured in the A norm) is low, which is often a key assumption for several multigrid convergence theories; see, e.g., [4]. As D is block diagonal with tridiagonal matrices for blocks, it is easily inverted. Notice that if A is symmetric, then $R = P^T$, and then $S = RAP$ is a standard Galerkin projection of A giving rise to a symmetric S. In the nonsymmetric case, $R \neq P^T$, leading to a Petrov–Galerkin projection.

5.3. Multigrid convergence in the one-dimensional setting. In the special 1D case, the multigrid operator is exact if employed with a suitable relaxation scheme. That is, this multigrid procedure converges in one iteration regardless of the PDE operator or the size of the rectangles. If, for example, A is a discretization of a PDE that depends on a variable or even highly discontinuous field, the multigrid process still converges in one iteration, even when more aggressive coarsening is used to generate the multigrid hierarchy. Furthermore, the method converges in one iteration regardless of the value of β if the 1D PDE includes a Robin condition (similar to the basal boundary condition). Figure 5.4 shows resulting interpolation basis functions for a 1D Poisson problem with 32 fine nodes (shown by ticks) and three coarse nodes (shown by circles). In this specific case, the computed interpolation operator is a 32×3 matrix, and the three illustrated hat functions correspond, respectively, to the plots of

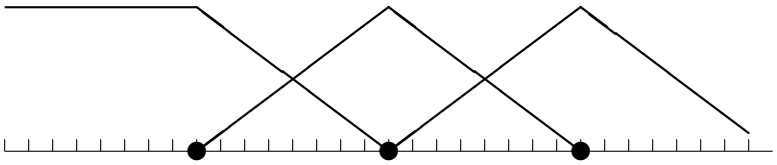

FIG. 5.4. *Interpolation basis functions computed by matrix dependent algorithm for a 1D Poisson operator with a Neumann (Dirichlet) condition on the left (right) boundary.*

the three columns. This Poisson problem includes two Robin conditions. The leftmost boundary has a β value of zero (effectively a Neumann condition), while the rightmost boundary has a β value of 10^6 (effectively a Dirichlet condition). One can see that the leftmost and rightmost basis functions satisfy the homogeneous form of the boundary conditions. That is, the derivative of the leftmost basis function is zero at the left boundary, and the rightmost basis function is zero at the right boundary. The exact convergence property is a consequence of the fact that the coarse operator corresponds to the exact Schur complement. Exactness also requires relaxation to be of either block Jacobi or block Gauss–Seidel variety, where each block consists of all degrees-of-freedom belonging to a rectangular block in Figure 5.3. These relaxation choices effectively apply the inverse of the D operator from (5.1) within the preconditioner. As mentioned earlier, the above choices for multigrid operators can often be motivated by a physical interpretation. The use of harmonic averages and properly capturing discontinuities on coarse levels has been considered for simple variable coefficient Poisson operators, e.g., $(\rho(x)u_x)_x$ [50, 1], and upwind or convection following grid transfers have been considered for model convection-diffusion systems [51].

5.4. Semicoarsened matrix dependent grid transfers in three dimensions. The term *matrix dependent* multigrid refers to a class of multigrid algorithms for constructing grid transfers using matrix entries *while also taking advantage of the structure of the underlying mesh* [43]. Loosely speaking, different matrix dependent algorithms are generalizations of the ideal 1D Gaussian elimination process to higher dimensions. These generalizations use approximations to avoid the computational expense of exact recursive Schur complements in higher dimensions. One such semicoarsening generalization can be obtained by collapsing the original PDE to a unidirectional operator aligned along a single coordinate direction via

$$A_z \leftarrow Collapse(A),$$

where $Collapse(A)$ takes an $n \times n$ matrix and replaces it with another $n \times n$ matrix that corresponds to a three-point vertical stencil approximation to only z derivatives in A. Thus, A_z includes no horizontal coupling, and so the 1D procedure can be applied to it to generate unidirectional semicoarsening grid transfers for each vertical grid line which, when combined, can be used to project A. For a purely algebraic solver, $Collapse()$ should not require applications to supply the z derivative finite difference approximation. The main observation is that this can be done algebraically without requiring any structure in the direction orthogonal to the coarsening direction.

While there are many possibilities, a simple $Collapse()$ procedure has proven effective. To facilitate the exposition, we describe the scalar PDE case. PDE systems extensions are straightforward. The 3D mesh must correspond to n_z stacked layers, where each layer contains the same set of (x, y) points (see, e.g., Figure 5.1) and A's stencils must extend only to nearest neighboring layers. First, define the function

Algorithm 1: Compute mth column of P.

```
/* Assumptions:                                                          */
/*  • vertices within extruded mesh lines are consecutively ordered      */
/*  • P_{i,m} ≠ 0 only for first_m ≤ i ≤ last_m                          */
/*      first_m is the first layer associated with the lower rectangular region. */
/*      last_m is the last layer associated with the upper rectangular region.   */
T = 0,  s = 0 ;
for i = first_m, ..., last_m do
    k = i − first_m + 1;
    if i lies in coarse layer coinciding with m then
        T_{k,k} = 1;                    // Dirichlet condition forcing
        s_k = 1;                        // T^{-1}s to be one at the kh point
    else
        T_{k,k} = Σ_{j∈ mid_i} a_ij;
        if i ≠ first_m then T_{k,k-1} = Σ_{j∈ bot_i} a_ij;
        if i ≠ last_m then T_{k,k+1} = Σ_{j∈ top_i} a_ij;
    end
end
P_{G,m} = T^{-1}s;      // G defined by integer set {i | first_m ≤ i ≤ last_m}
```

$layer(i)$, which returns the layer associated with the ith node. That is, the (x_i, y_i, z_i)th grid point resides on the $layer(i)$ stacked plane, where adjacent layers are numbered consecutively (starting from the bottommost layer). Then, consider the sets

$$\begin{aligned} top_i &\equiv \{j \in [1, \ldots, n] \mid a_{ij} \neq 0 \ \ \& \ \ layer(j) = layer(i) + 1\}, \\ mid_i &\equiv \{j \in [1, \ldots, n] \mid a_{ij} \neq 0 \ \ \& \ \ layer(j) = layer(i) \quad \}, \\ bot_i &\equiv \{j \in [1, \ldots, n] \mid a_{ij} \neq 0 \ \ \& \ \ layer(j) = layer(i) - 1\}. \end{aligned}$$

Then, a simple mechanism for constructing the three-point stencil for A_z's ith row is

$$\left[\sum_{j \in bot_i} a_{ij} \qquad \sum_{j \in mid_i} a_{ij} \qquad \sum_{j \in top_i} a_{ij} \right], \tag{5.3}$$

which is truncated to two entries for i's within the topmost or bottommost plane. The idea is the same for PDE systems, but in this case A_z is a block tridiagonal matrix. Some properties of $Collapse()$ are given in section 5.5 highlighting preservation of null space components and the ability to filter out matrix contributions associated with differentiation in the x/y plane.

A code fragment is given in Algorithm 1 to determine the mth column of P, which is equivalent to applying the 1D algorithm to each of the independent vertical line problems associated with A_z. This construction builds a tridiagonal matrix T and a right-hand side s, and it computes $T^{-1}s$. Rows of A_z are computed on-the-fly and immediately inserted into T. Connecting this fragment with the previous 1D case, Figure 5.3 should be rotated so that it corresponds to one vertical line in a 3D mesh.

The row/column subset of A_z inserted into T corresponds to equations associated with two adjacent rectangular regions in the rotated figure. Now, unknowns within rectangular regions reside in horizontal layers that are not represented on the next coarsest mesh. The mesh point sandwiched between two rectangles is an unknown represented on the coarse mesh. The mth grid transfer basis function takes the value of 1 at this point (corresponding to a single row of the identity block in (5.2)). This is accomplished by inserting a Dirichlet condition into T, forcing $T^{-1}s$ to be 1 at this sandwich point. This computation is equivalent to (5.2) using A_z instead of A, as the product of A_z and the mth grid transfer column is identically zero for all rows corresponding to rectangle interiors, and the basis function is 1 at the sandwich point. It should be noted that here the unknown ordering within a single line is consecutive (as opposed to (5.2), where coarse points are ordered after fine points). Conceptually, the algorithm for constructing restriction in the nonsymmetric case is identical to applying Algorithm 1 to A^T (instead of A) to compute a $\tilde{P}$ and then transposing the result, i.e., $R = \tilde{P}^T$.

As noted, the proposed algebraic scheme can address large $\beta(x, y)$, e.g., $\beta(x, y) \sim \mathcal{O}(10^7)$, that we have found problematic for standard smoothed aggregation algebraic multigrid. In particular, large $\beta(x, y)$ leads to large coefficient matrix stencils resembling finite element mass matrices at Robin boundaries. One can see potential AMG difficulties by considering piecewise constant (vertical line) grid transfers. In this case, the resulting coarse level stencils via Galerkin projection correspond to a simple averaging of fine level stencils from vertical neighbors. Unfortunately, large matrix stencils dominate these averages so that interior PDE terms have little impact on the resulting coarse stencils. Of course, averaging results from the simple choice of piecewise constants, but it underscores the importance of interpolation near Robin boundary conditions, where large $\beta(x, y)$ should lead to corresponding small interpolation weights. As illustrated by the rightmost boundary of Figure 5.4, the proposed matrix dependent transfers satisfy this property that is particularly important for our application.

As mentioned, the matrix dependent algorithm naturally supports aggressive coarsening. This is helpful to avoid inefficiencies associated with highly parallel architectures. With standard coarsening (e.g., coarse mesh defined by doubling the mesh spacing in the z direction), the matrix dimension diminishes by a factor of 2 (as opposed to 8 with full coarsening) and by a factor of 3 (as opposed to 27 with full coarsening) when the mesh spacing triples. This leads to many hierarchy levels, each requiring potentially expensive communication. Thus, aggressive semicoarsening can be used to effectively control the number of coarse levels within the multigrid hierarchy and, consequently, limit the number of messages per sweep of a multigrid cycle. We currently require an entire vertical line to reside within a single computational core. This simplifies coding, though it imposes a burden on the partitioning software. The key kernel during grid transfer construction and multigrid line smoothing consists of independent band/tridiagonal solves that do not require any communication. The vertical lines can be derived by our software from user-supplied coordinates or if users indicate certain types of consistent orderings (e.g., vertical line-by-line or horizontal layer-by-layer) used to define the matrix.

To complete the method definition, relaxation must be chosen as well as an AMG scheme for when the anisotropic behavior is no longer present or when it is not possible to further semicoarsen (e.g., single layer mesh as shown in the rightmost top image of Figure 5.2). While any AMG method could be applied after semicoarsening is completed, we prefer SA-AMG because near null space information (such as rigid

body modes) is easily incorporated. In our particular case, smoothed aggregation's interpolation can then be oriented so that x translations, y translations, and the $x-y$ rotations which are components of the null space for purely floating ice, are accurately transferred between hierarchy levels. To use rigid body modes supplied by an application, it is necessary that the semicoarsening project fine level modes to the single layer level. In our implementation, this is done by injecting modes from finer to coarser levels during semicoarsening. For relaxation, standard choices are of the Gauss–Seidel variety when the coarsening rate is not particularly aggressive. However, line Jacobi or line Gauss–Seidel relaxation is natural when using *aggressive* semicoarsening. The efficiency of this smoothing is tied into the degree to which the mesh is anisotropic in the extruded dimension. For ice sheets, the anisotropic nature of the mesh is quite pronounced and is a direct consequence of the relative thin height of an ice sheet when compared to its horizontal extent.

5.5. Grid transfer properties. The $P^{(h)}$ produced by Algorithm 1 applied to a discretization matrix $A^{(h)}$ is identical to applying the 1D algorithm to $Collapse(A^{(h)})$. Therefore, $P^{(h)}$ satisfies

$$[Collapse(A^{(h)})P^{(h)}]_{\tilde{\mathcal{S}}} = 0 \tag{5.4}$$

and

$$[P^{(h)}]_{\mathcal{S}} = I, \tag{5.5}$$

analogues to the 1D case where AP is identically zero for rows corresponding to rectangle interiors in Figure 5.3 and to the form of P given in (5.2). Here, $\tilde{\mathcal{S}}$ refers to the set of unknowns residing in a layer not represented on the coarse mesh (the counterpart to rectangle unknowns in Figure 5.3) and $\mathcal{S}$ is the set of fine unknowns remaining on the coarse mesh. Thus, $\mathcal{S} \cap \tilde{\mathcal{S}} = \emptyset$, and $\mathcal{S} \cup \tilde{\mathcal{S}}$ includes all unknowns. Additionally, the stencils associated with $Collapse(A^{(h)})$ are formed by summing stencils associated with $A^{(h)}$. When boundary conditions are not imposed, the sum of each row's entries within $A^{(h)}$ are identically zero, and so $Collapse(A^{(h)})$'s rows also have zero sums.

Using the above properties, it is possible to understand a few important characteristics of $P^{(h)}$. In particular, one can show that the fine discretization's null space lies within the range space of $P^{(h)}$, which is generally understood to be a desirable multigrid property. Recall from section 2 that the null space of (2.1) is spanned by the three sets of vectors (1) $u = 1$, $v = 0$; (2) $u = 0$, $v = 1$; and (3) $u = -y$, $v = x$ when boundary conditions are not imposed. As semicoarsening occurs only in the z direction and each of these three vector sets is constant in the z direction, it suffices to show that the range of $P^{(h)}$ includes all vectors associated with functions that are constant in the z direction.

THEOREM 5.1. *Consider two functions $g_1(x_i, y_i)$ and $g_2(x_i, y_i)$ that do not depend on the z coordinate, i.e., constant functions in the z direction. Define fine and coarse vectors that can be applied to discrete representations of the two PDE system* (2.1) *when boundary conditions are not imposed:*

$$[w^{(h)}]_i = \begin{cases} g_1(x_i, y_i), & i \text{ is odd}, \quad i \in \tilde{\mathcal{S}} \cup \mathcal{S}, \\ g_2(x_i, y_i), & i \text{ is even}, \quad i \in \tilde{\mathcal{S}} \cup \mathcal{S}, \end{cases}$$

and

$$[w^{(H)}]_i = \begin{cases} g_1(x_i, y_i), & i \text{ is odd}, \quad i \in \mathcal{S}, \\ g_2(x_i, y_i), & i \text{ is even}, \quad i \in \mathcal{S}. \end{cases}$$

Here, x_i and y_i denote x and y coordinates associated with the ith degree of freedom,[3] *and it is assumed that degrees-of-freedom are numbered by node (and not by component). That is, degrees-of-freedom corresponding to $g_1()$ ($g_2()$) are associated with odd (even) numbering. Then, the interpolation operator $P^{(h)}$ associated with applying Algorithm 1 to $A^{(h)}$ satisfies the relation*

$$w^{(h)} = P^{(h)} w^{(H)}.$$

That is, $P^{(h)}$ preserves functions that are constant in the z direction.

Proof. $w^{(h)}$ lies in the null space of $Collapse(A^{(h)})$, as $Collapse(A^{(h)})$'s stencils only combine components in the z direction, $Collapse(A^{(h)})$ has zero row sums when boundary conditions are not imposed, and $w^{(h)}$ is constant in the z direction. From (5.5), $[P^{(h)} w^{(H)}]_{\mathcal{S}} = [w^{(h)}]_{\mathcal{S}}$. That is, taking $P^{(h)} w^{(H)}$ to be $w^{(h)}$ satisfies (5.5). As $w^{(h)}$ lies in the null space of $Collapse(A^{(h)})$, it follows that taking $P^{(h)} w^{(H)}$ to be $w^{(h)}$ also satisfies (5.4), i.e., $[Collapse(A^{(h)}) P^{(h)} w^{(H)}]_{\tilde{\mathcal{S}}} = 0$. As these two properties uniquely define $P^{(h)} w^{(H)}$, it must be that $w^{(h)} = P^{(h)} w^{(H)}$, as this satisfies both properties. ∎

From Theorem 5.1, it follows that the three null space vectors, each of which is independent of the z coordinate, are exactly preserved by interpolation.

By defining a 1D operator via the $Collapse()$ function, (5.4) is satisfied. However, ideally one would prefer

$$[A^{(h)} P^{(h)}]_{\tilde{\mathcal{S}}} = 0, \tag{5.6}$$

as this would effectively imply that the associated Galerkin projection corresponds to a Schur complement operator as in the 1D case. While (5.6) does not generally hold, it is possible to illustrate some related characteristics by making some additional assumptions. For example, suppose that the extruded mesh is such that the z coordinate on each layer is constant. That is, $z_i = z_j$ for all i and j such that $layer(i) = layer(j)$. Define the following vectors that are now constant in the x/y plane:

$$[\bar{w}^{(h)}]_i = \begin{cases} g_1(z_i), & i \text{ is odd}, \quad i \in \tilde{\mathcal{S}} \cup \mathcal{S}, \\ g_2(z_i), & i \text{ is even}, \quad i \in \tilde{\mathcal{S}} \cup \mathcal{S}, \end{cases}$$

and

$$[\bar{w}^{(H)}]_i = \begin{cases} g_1(z_i), & i \text{ is odd}, \quad i \in \mathcal{S}, \\ g_2(z_i), & i \text{ is even}, \quad i \in \mathcal{S}. \end{cases}$$

Suppose now that we can split the discretization matrix into two pieces,

$$A^{(h)} = A^{(h)}_{xy} + A^{(h)}_{z},$$

where $A^{(h)}_{xy}$ is associated with differentiation in the x and y directions and $A^{(h)}_{z}$ approximates differentiation in the z direction. This effectively assumes that there are no mixed derivatives in z and x—or in y, which does not hold for the first-order Stokes model described in section 2. As $A^{(h)}_{xy} \bar{w}^{(h)}$ approximates differentiation in the x/y plane, it must be identically zero for consistent discretizations. By choosing different $\bar{w}^{(h)}$ that are only nonzero within a single layer, it follows that

$$\sum_{j \in\ bot_i} (A^{(h)}_{xy})_{ij} = 0, \qquad \sum_{j \in\ mid_i} (A^{(h)}_{xy})_{ij} = 0, \quad \text{and} \qquad \sum_{j \in\ top_i} (A^{(h)}_{xy})_{ij} = 0.$$

[3] Recall that all degrees-of-freedom within a vertical line share the same x and y coordinates.

Thus, contributions from x/y differentiation cancel in (5.3), and so

$$Collapse(A^{(h)}) = Collapse(A_z^{(h)}).$$

When $A_z^{(h)}$ corresponds to a three-point vertical stencil, then (5.4) becomes

$$[A_z^{(h)} P^{(h)}]_{\tilde{\mathcal{S}}} = 0,$$

which is somewhat closer to (5.6).

If we additionally assume that layerwise constants are perfectly preserved by interpolation, i.e., $\bar{w}^{(h)} = P^{(h)} \bar{w}^{(H)}$, then (5.4) becomes

$$[A^{(h)} P^{(h)} \bar{w}^{(H)}]_{\tilde{\mathcal{S}}} = 0. \tag{5.7}$$

This effectively states that interpolated layerwise constant functions satisfy the discrete PDE within layers that are not represented on the coarse mesh. Here, (5.7) relies on $Collapse(A^{(h)}) w^{(h)} = A^{(h)} w^{(h)}$, which follows from the definition of $Collapse()$ and the fact that $w^{(h)}$ is layerwise constant. Of course, (5.7) assumes that $P^{(h)} w^{(H)}$ yields a layerwise constant. This occurs when matrix stencils of $Collapse(A_z^{(h)})$ do not vary for rows within the same layer, as this leads to $P^{(h)}$ with stencils that do not vary for rows within the same layer. $Collapse(A_z^{(h)})$ will generally have this property when z derivative terms in the underlying PDE have no x/y dependence (e.g., $(g(z)u_z)_z$) and when the underlying mesh is uniform in the x/y plane.

Finally, we note that when additionally $A_z^{(h)} u^{(h)}$ is a standard approximation of u_{zz}, the resulting grid transfer operator corresponds to linear interpolation in the z direction. Thus, convergence results associated with semicoarsening and linear interpolation would apply in this particular case.

6. Practical considerations. This section briefly discusses two solver aspects that influence the numerical comparisons presented in the next section. The first concerns the use of ILU preconditioners for "thin" problems like that considered here, and the second centers on meshing and irregular boundaries.

6.1. Node ordering. ILU preconditioners have been employed with some success within ice sheet modeling. This success can be understood by recalling the earlier Green's function arguments. Specifically, consider injecting a Green's function with very rapid horizontal decay onto a mesh. In this case, nodal values outside of a small horizontal neighborhood will have negligible magnitude relative to those near the function peak. On a mesh of modest resolution, this neighborhood might include only a few adjacent vertices. This implies that a preconditioner need not approximate long distance horizontal couplings. Thus, a vertical line Jacobi, line Gauss–Seidel, or properly ordered ILU preconditioner might be acceptable. To realize acceptable convergence with processor-local ILU, vertical coupling must be accurately approximated. Thus, equations must be partitioned so that each processor is assigned an entire vertical line. Furthermore, equations within a processor must be ordered layerwise (i.e., equations within a processor associated with the kth layer are numbered before equations associated with layer $k+1$th). To understand the advantages of layerwise ordering, consider the matrix

$$A = \begin{pmatrix} A_{1,1} & A_{1,2} & & & & & \\ A_{2,1} & A_{2,2} & A_{2,3} & & & & \\ & \cdot & \cdot & \cdot & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & A_{n_z,n_z-1} & A_{n_z,n_z} \end{pmatrix},$$

where $A_{i,j}$ represents coupling between unknowns within the ith and jth layers. An ILU(0) factorization produces a lower (upper) block bidiagonal L (U) factor. The product $\hat{A} = LU$ is also block tridiagonal, and so its stencil for a point lying within the kth layer only includes entries from neighboring layers. That is, the stencil width in the vertical direction has not grown (in contrast to when using a different unknown ordering). By the definition of ILU(0), $\hat{a}_{ij} = a_{ij}$ for all i, j such that a_{ij} is nonzero,[4] and so one can argue that the vertical coupling is captured by the incomplete factors. One can further show (though details are omitted for brevity) that if $A_{i,j} = \alpha_{i,j} A_{1,1}$, then $\hat{A}_{i,j} = \alpha_{i,j} \hat{A}_{1,1}$, i.e., the vertical $\alpha_{i,j}$ coupling is perfectly captured. This ideal situation can occur for discretizations of a homogeneous medium on an extruded mesh with simple boundary conditions and where the z direction spacing is constant across all layers. Though the $A_{i,j}$'s are not generally *exact* scalar multiples of each other, they are not vastly far from this case, and so ILU(0) accurately captures vertical coupling.

6.2. Removal of islands and hinged peninsulas. The second concern arises from iterative solver difficulties and ice sheet meshing challenges. In particular, mesh construction typically requires processing raw satellite data associated with irregular coastlines. If care is not given to this process, the imposition of a mesh on the satellite images leads to isolated or partially isolated regions of ice. Figure 6.1 illustrates small regions of meshes obtained by a typical algorithm for converting raw satellite data into meshes. Most conversion algorithms do remove some troublesome features, e.g., isolated mesh points that cannot be connected with neighbors to generate a finite element with nontrivial area. While some irregularities are removed, one can still see highly intricate structures, including isolated regions and narrow peninsulas. Similar features could be generated dynamically when running time dependent simulations continually during model run time, e.g., by the modeling of physical processes that lead to iceberg generation ("calving") at ice sheet margins that are in contact with the ocean. The "floating" islands that result (i.e., areas for which $\beta(x, y) = 0$) lead to matrix singularities, as rigid body translations and $x-y$ plane rotations of the island correspond to null space components. Rigid body translations are not null space components for floating peninsulas. However, rotations lead to a singularity for floating peninsulas that are attached to the mainland at just one horizontal point (hinged peninsulas). In this case, the peninsula is free to swing around the attachment point, similarly to a door on a hinge. One island and two hinged peninsulas can be seen in Figure 6.1 (right).

To avoid singularities, we developed a new algorithm for detecting and removing floating islands and hinged peninsulas, as these artifacts add little to the overall solution.[5] Alternatively, one can artificially impose $\beta(x, y) \geq \epsilon_t$, where ϵ_t is a positive minimum value. While this removes the singularity, solution quality and severe ill-conditioning (with small ϵ_t) should be studied. Our new algorithm for identifying islands and hinges is adapted from standard graph schemes for detecting connected components. The complete details of the algorithm are provided in a supplementary appendix, as a high level of care is required to properly cover all possible cases. The algorithm/code given in the supplementary material has been tested on a number of hard cases involving multiple connected hinged peninsulas and complex mesh regions with just two mainland connections in order to stress the hinged peninsula test. For

[4] Nonexactness comes from the presence of nonzero $\hat{a}_{ij}$'s in some cases where a_{ij} is zero.

[5] The removal of icebergs is fairly standard practice, though the icebergs are still accounted for by the ice sheet and climate model in terms of conserving mass and latent heat of the ice.

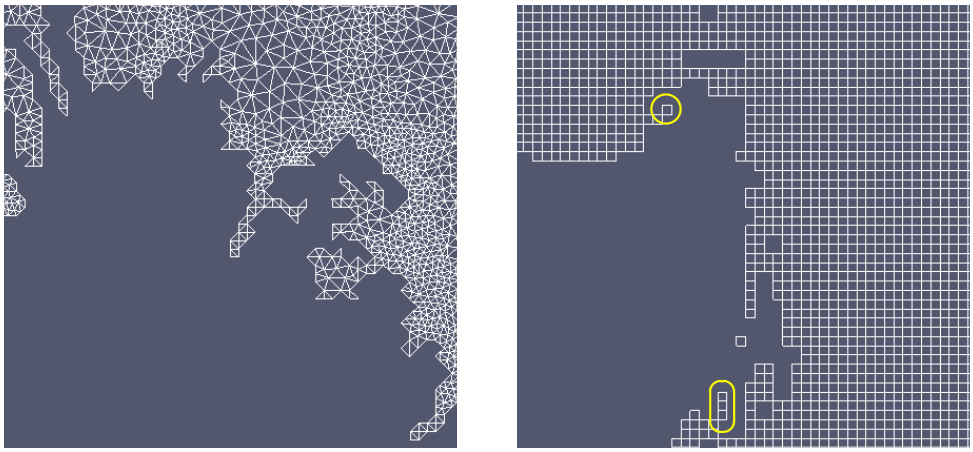

FIG. 6.1. *Coastlines: Greenland triangle mesh (left) and Antarctica quadrilateral mesh (right). Two hinges are highlighted with yellow ovals. (See online version for color.)*

efficiency, it is only necessary to consider the subgraph associated with elements that have at least one vertex with a $\beta(x, y) = 0$. A standard algorithm can then be applied to this subgraph to find a connected component starting from a vertex with a positive $\beta(x, y)$. This connected component is removed, and the process is repeated until no vertices with positive $\beta(x, y)$ remain. This final graph contains all floating island vertices that should be removed.

While the island detection method is a fairly straightforward adaptation of classical algorithms for finding connected components within a graph, finding hinged peninsulas requires the development of a more sophisticated approach. Our algorithm for hinged peninsulas starts by coloring the graph. The algorithm uses a modest number of colors and assigns one color to each vertex such that no two adjacent vertices share the same color. For each color, temporarily remove vertices of the *selected color* and apply the island detection algorithm to the color-removed subgraph. These *color-removed islands* are hinged peninsulas in the original graph if they are adjacent to only one *selected color* vertex that is connected to the mainland. That is, all other adjacent *selected color* vertices are surrounded by vertices from either the *color-removed island* or already removed hinged peninsulas. This process is repeated for each color. As mentioned above, the details are subtle, and so a MATLAB code is supplied in the supplementary material. In the next section, results are given for $\beta(x, y)$ fields with zero values on postprocessed meshes, where all hinged peninsulas have been removed. For a standard Antarctica mesh, this removal typically eliminates about 10 elements. Additionally, solution data will also be given for regularized $\beta(x, y)$ fields ($\epsilon_t > 0$) on the original meshes (without removing harmful components).

7. Numerical results.

7.1. Three-dimensional variable coefficient Poisson problem. A series of experiments are first performed to assess solver sensitivity to various problem and algorithm parameters. To do this, we consider a 3D variable coefficient Poisson problem

$$(q(x, y)u_x)_x + (q(x, y)u_y)_y + (s(z)u_z)_z = 0$$

on a box domain defined by $0 \le x, y \le 1$, and $0 \le z \le z_{max}$ with boundary conditions

$$\begin{array}{lll} u(0,y,z)=0, & u(x,0,z)=0, & \beta\ u(x,y,0)+u_z(x,y,0)=0, \\ u(1,y,z)=1, & u(x,1,z)=1, & u_z(x,y,z_{max})=1. \end{array}$$

A discrete problem is obtained by a standard seven-point finite different stencil on a completely uniform structured mesh. For example, $(q(x,y)u_x)_x$ is approximated by

$$\left[q(x_{i+\frac{1}{2},j,k})u^h_{i+1,j,k}+q(x_{i-\frac{1}{2},j,k})u^h_{i-1,j,k}-\left[q(x_{i+\frac{1}{2},j,k})+q(x_{i-\frac{1}{2},j,k})\right]u^h_{i,j,k}\right]/h_x^2$$

at the (i,j,k)th mesh point. Here, $u^h_{i,j,k}$ refers to the discrete solution at the (i,j,k)th mesh point, while h_x gives the mesh spacing in the x direction. For all model problem tests, a conjugate-gradient (CG) solver is employed in conjunction with a V cycle multigrid preconditioner, and iterations are reported corresponding to a reduction in the 2-norm of the initial residual by a factor of 10^{-6} (starting with a 0 initial guess). Furthermore, all tests are performed on a $161\times161\times161$ mesh. Coarse multigrid meshes are obtained by coarsening in the z direction using equispace coarse points (though equispace points are not necessary for the software), where coarsening rates correspond to powers of 3.[6] A direct solver is used on the coarsest multigrid mesh, which always corresponds to a single z layer. Table 7.1 illustrates iterations as a function of different z_{max} and β choices as well as different coarsening rates for $q(x,y)=1$ and $s(z)=1$. When $z_{max}=1.0$, the problem is isotropic and the method performs poorly, with the most aggressive coarsening rate version being the worst. It should be noted that in the case of a Poisson operator, the grid transfer operators within the domain interior correspond to linear interpolation in the z direction, which is natural for a Poisson operator. The reason for poor convergence is not necessarily the grid transfer, but rather inadequate z direction smoothing. In fact, with modest coarsening rates (i.e., the 5-level method), the coarse level discretizations have much finer mesh spacing in the x and y directions than in the z direction (due to z coarsening). That is, they appear anisotropic in the opposite direction for which the method was designed. Convergence is noticeably better once the mesh spacing ratio between the x/y directions and the z direction is 5 to 1 (requiring less than 20 iterations for the 5-level method with large β). Once the mesh spacing ratio is 25 or greater, convergence is quite rapid. For mildly anisotropic problems ($z_{max}\geq0.2$), the modest coarsening rate method generally converges about twice as fast as the method which immediately coarsens to a single level. When communication dominates the cost per iteration, the more aggressively coarsened hierarchies may require less overall time, though these experiments are carried out on a serial platform. Finally, one can see that the solver is relatively insensitive to β variations. Convergence is generally a bit more rapid for larger β. This is consistent with the earlier discussion related to Green's function decay.

To clarify the impact of using a line smoother as opposed to a point smoother, Table 7.2 considers the same parameter choices with the only exception being that one pre- and one postsmoothing sweep of point symmetric Gauss–Seidel smoothing is now used on all the finer levels. One can see that there are cases when the point smoother requires fewer or almost the same number of iterations as the line smoother. These situations generally correspond to the modest anisotropic cases with the least aggressive coarsening. However, line smoothing generally pays off when the problem is more anisotropic or when more aggressive coarsening is employed.

[6]The number of z layers within the multigrid hierarchy is given by $3^{-k}(161+1)-1$, where $k=0,1,2,3,4$ for the 5-level method, $k=0,2,4$ for the 3-level method, and $k=0,4$ for the 2-level method.

TABLE 7.1

CG iterations for $q(x,y)=1$ and $s(z)=1$ using 1 pre- and 1 postsmoothing sweep of symmetric line Gauss–Seidel.

z_{max}	β				
	0.0	10^2	10^4	10^6	Method
1.0	51	44	41	41	5-level multigrid where
0.2	21	15	13	14	the # of z direction
0.04	5	4	4	4	points from the finest
0.008	3	2	2	2	to the coarsest level is
0.0016	2	1	1	1	$161, 53, 17, 5, 1$
1.0	68	61	57	57	3-level multigrid where
0.2	30	21	18	18	the # of z direction
0.04	6	5	5	5	points from the finest
0.008	3	3	2	2	to the coarsest level is
0.0016	2	2	1	1	$161, 17, 1$
1.0	95	86	81	80	2-level multigrid where
0.2	41	30	26	26	the # of z direction
0.04	9	7	7	7	points from the finest
0.008	4	3	3	3	to the coarsest level is
0.0016	2	2	1	1	$161, 1$

TABLE 7.2

CG iterations for $q(x,y)=1$ and $s(z)=1$ using 1 pre- and 1 postsmoothing sweep of symmetric point Gauss–Seidel.

z_{max}	β				
	0.0	10^2	10^4	10^6	Method
1.0	41	38	38	38	5-level multigrid where
0.2	19	18	18	18	the # of z direction
0.04	8	8	8	8	points from the finest
0.008	7	7	7	7	to the coarsest level is
0.0016	7	7	7	7	$161, 53, 17, 5, 1$
1.0	61	58	58	54	3-level multigrid where
0.2	30	27	27	27	the # of z direction
0.04	18	16	16	10	points from the finest
0.008	20	17	16	8	to the coarsest level is
0.0016	19	17	16	8	$161, 17, 1$
1.0	117	105	99	70	2-level multigrid where
0.2	156	114	102	40	the # of z direction
0.04	178	141	126	22	points from the finest
0.008	141	139	116	21	to the coarsest level is
0.0016	87	86	82	19	$161, 1$

To understand the solver's sensitivity to coefficient variations in the x/y plane, we consider $q(x,y) = \sin(k\pi x)\ \sin(k\pi y) + c$ for different values of k and c in Table 7.3. Overall, the iteration counts do not vary significantly for different values of k and c. It should be kept in mind that the $c = 0$ ($c = 1$) case gives rise to weaker (stronger) x/y coupling when compared with the $q(x,y) = 1$ situation. Table 7.4 considers the case of coefficient variation in the z direction. Here, $q(x,y) = 1$, while $s(z) = \sin(k\pi x) + c$. The $c = 1$ case shows virtually no variation in iteration counts, as k is varied. However, the $c = 0$ case gives rise to some variation and noticeably poor convergence rates when compared to the constant coefficient case. Here, we point out that while $s(z) \geq 0$, it is small for many values of z. These small regions correspond to weak coupling in the z direction, which again is not the regime that the algorithm is designed to tackle.

TABLE 7.3
CG iterations for $q(x,y) = \sin(k\pi x)\ \sin(k\pi y)+c$ and $s(z) = 1$ using 1 pre- and 1 postsmoothing sweep of symmetric line Gauss–Seidel.

z_{max}	$c = 0.0$			$c = 1.0$			# MG levels
	k = 2	k = 4	k = 8	k = 2	k = 4	k = 8	
1.0	53	56	54	61	63	61	5
0.2	23	23	22	28	28	26	5
0.04	6	6	5	8	7	7	5
0.008	3	3	3	3	3	3	5
0.0016	2	2	2	2	2	2	5
1.0	73	79	74	83	86	81	3
0.2	33	33	31	40	40	38	3
0.04	9	8	8	11	10	10	3
0.008	3	3	3	4	4	4	3
0.0016	2	2	2	2	2	2	3
1.0	103	111	104	116	121	113	2
0.2	46	46	44	56	57	53	2
0.04	12	11	11	15	15	14	2
0.008	4	4	4	4	4	4	2
0.0016	2	2	2	2	2	2	2

TABLE 7.4
CG iterations for $q(x,y) = 1$ and $s(z) = \sin(k\pi z/z_{max}) + c$ using 1 pre- and 1 postsmoothing sweep of symmetric line Gauss–Seidel.

z_{max}	$c = 0.0$			$c = 1.0$			# MG levels
	k = 2	k = 4	k = 8	k = 2	k = 4	k = 8	
1.0	89	77	78	83	83	83	5
0.2	46	50	48	32	32	32	5
0.04	31	25	19	7	7	7	5
0.008	10	6	4	3	3	3	5
0.0016	3	3	3	2	2	2	5
1.0	116	101	98	59	59	59	3
0.2	64	67	66	23	23	23	3
0.04	43	34	27	5	5	5	3
0.008	14	9	6	3	3	3	3
0.0016	8	3	3	2	2	2	3
1.0	151	140	138	41	43	43	2
0.2	84	96	94	16	16	16	2
0.04	60	48	38	4	4	4	2
0.008	20	13	9	4	3	3	2
0.0016	5	4	3	2	2	2	2

7.2. Greenland and Antarctica first-order Stokes problems. All remaining experiments use the *Albany/FELIX* finite element code[7] to discretize the first-order-accurate Stokes equations (2.1) via first-order nodal finite elements on vertically extruded meshes. Realistic $\beta(x,y)$ and bed topography fields were obtained from a deterministic inversion procedure minimizing the discrepancy between modeled and observed surface velocities [33]. For Greenland simulations, a realistic 3D temperature field, originally calculated for [42], provided realistic flow-law rate factors. For Antarctica, we obtained the same from unpublished simulations using another large-scale Antarctic ice sheet model (from personal communication with Dan Martin). A sample solution of Antarctica ice sheet surface speeds is shown in Figure 7.1 (left).

A sequence of linear systems is produced from a Newton-based solver applied to the nonlinear equations. Due to nonlinear convergence difficulties associated with

[7]For a detailed discussion of the *Albany/FELIX* code, the reader is referred to [46].

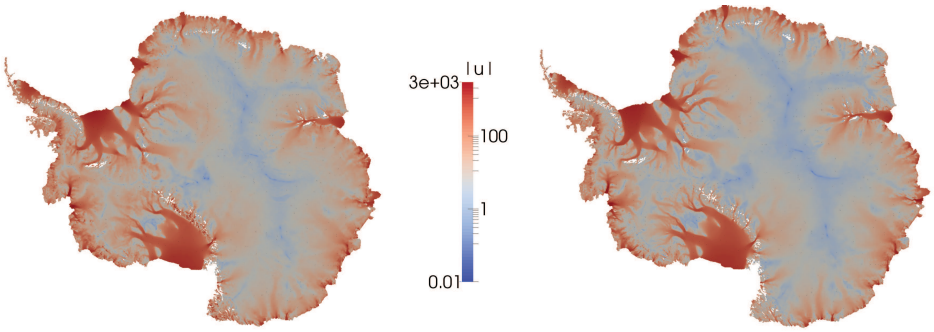

FIG. 7.1. *Surface velocity magnitudes [m/yr] of Antarctica ice sheet computed using the first-order Stokes method (left) or obtained by observations (right).*

TABLE 7.5
CG iterations/timings for Greenland using 4, 32, 256, 2048, *and* 16384 *cores on respectively larger meshes. The largest matrix is of dimension greater than* 1.1 *billion.*

Horizontal resolution	# layers	ILU # lin. sys.	ILU Avg. its. per solve	ILU Total lin. seconds	MDSC–AMG # lin sys.	MDSC–AMG Avg. its. per solve	MDSC–AMG Total lin. seconds
8 km	5	36	15.1	73.9	36	13.3	78.2
4 km	10	34	34.0	116.6	34	12.7	76.6
2 km	20	35	68.5	194.3	35	12.3	77.9
1 km	40	36	139.6	353.3	36	13.3	100.0
500 m	80	35	320.3	748.7	37	17.4	134.2

poor initial guesses of the effective strain rate, (2.4) is modified by introducing a homotopy continuation parameter, $1 \gg \gamma > 0$:

$$\mu_\gamma = \frac{1}{2} A^{-\frac{1}{3}} \left(\dot{\epsilon}_e^2 + \gamma\right)^{-\frac{1}{3}}. \tag{7.1}$$

Thus, a sequence of nonlinear problems is solved for different values of decreasing γ. The nonlinear solver and homotopy are implemented in the *NOX* and *LOCA Trilinos* [18] packages, respectively. A more complete description of *Albany/FELIX* and these algorithms can be found in [46, 47].

As already mentioned, substantial investigations with standard SA-AMG preconditioning [49] were extremely disappointing. In particular, a Krylov method preconditioned with SA-AMG simply does not converge within any reasonable number of iterations unless an ILU smoother (where equations are ordered layerwise) is used within the hierarchy. Even with this smoother, the resulting multilevel preconditioner offered no run time improvement over using just an ILU preconditioner (without multigrid). Experimentation with different parameters such as strength-of-connection thresholds did little to improve the situation.

In contrast to our smoothed aggregation experience, Table 7.5 illustrates performance improvements obtained with the new matrix dependent semicoarsening–AMG algorithm (MDSC–AMG) compared to an ILU(0) preconditioner. All reported timings were obtained on the Hopper Cray XE6 computer at the National Energy Research Scientific Computing (NERSC) Center. In these runs, the inner linear solver is terminated if either the 2-norm of the residual is reduced by 10^{-6} when compared to the initial residual or if 400 iterations are reached (which did occur 10 times within the 500 m ILU(0) simulation, but not for the other runs). The MDSC–AMG results

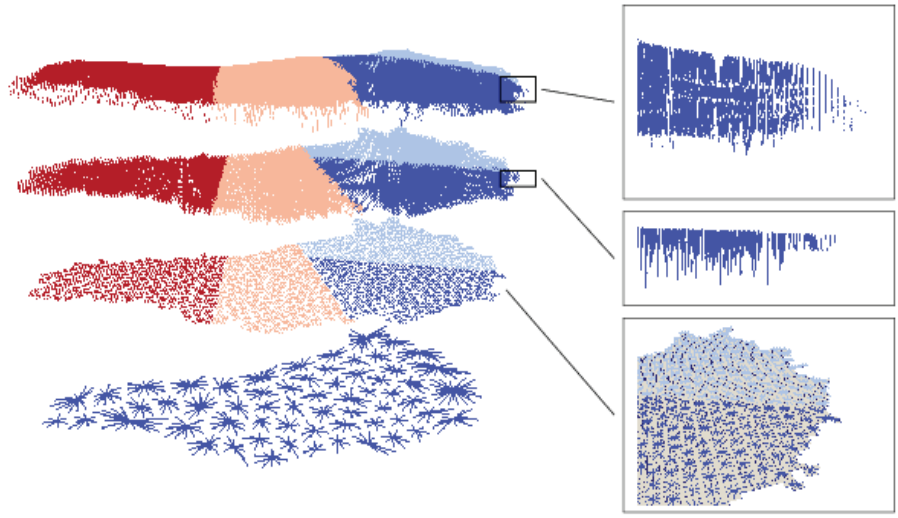

FIG. 7.2. *Visualization of MDSC–AMG coarsening. Colors represent different processors. Each aggregate consists of lines drawn from the root/central vertex to its neighbors.*

use the *Trilinos ML* package, though MDSC–AMG is also available from the *Trilinos MueLu* package [36]. The preconditioned MDSC–AMG iterations and timings are not far from exhibiting scalable behavior. In particular, the matrix dimensions vary from 334,476 to 1,132,268,706, holding the number of equations per core relatively fixed as the number of processors is increased. Over this 3000^+ factor increase in problem size, there is a slight increase in iterations, and the total time rises by approximately a factor of 2. Furthermore, MDSC–AMG leads to generally faster solution times than the ILU(0) preconditioner. For example, one can see a factor of 5 improvement on the most refined mesh. As a point of comparison, one of the better parameter choices for an SA-AMG preconditioner applied to the 2 km problem using an ILU(0) smoother (with properly ordered equations) required an average of 27.7 Krylov iterations per solve and a total of 284.2 seconds, i.e., $\approx 1.5x$ slower than just using an ILU(0) preconditioner. A pictorial representation of the coarsening process is illustrated in Figure 7.2 for a small four processor case. One can see that aggregates are vertically (horizontally) oriented on finer (coarser) levels. For these runs, the requested vertical coarsening rate is 12. This is used as a loose target by the software in determining the number of coarse layers while semicoarsening. Here, the MDSC–AMG preconditioner semicoarsens twice to reach a one layer grid when applied to the 1 km and 500 m meshes. The lower resolution problems employ semicoarsening once. A line Jacobi smoother ($\omega = 0.55$) is used on the two finest levels, while a Chebyshev smoother is used on the remaining coarse levels, with the exception of the coarsest level direct solver. The number of pre- and postsweeps on the finest levels is one, while it is four on all remaining levels. These particular choices were made to optimize scalability, but overall MDSC–AMG is not too sensitive to different algorithm choices. Additionally, it is apparent from the results that ILU(0) with layerwise ordering performs well over a range of meshes. In fact, ILU(0) leads to the fastest solution times on the coarsest mesh. However, on the finest mesh it is much slower than the MDSC–AMG preconditioner. We note that layerwise ordering is critical to ILU(0), as other orderings generally do not lead to convergence on the larger mesh problems.

TABLE 7.6
GMRES iterations/timings for Antarctica using 16, 128, 1024, *and* 8192 *cores on respectively larger meshes. The largest matrix is of dimension greater than* 1.1 *billion.*

		ILU			MDSC–AMG		
Horizontal resolution	# layers	# lin. sys.	Avg. its. per solve	Total lin. seconds	# lin. sys.	Avg its. per solve	Total lin. seconds
8 km	5	33	84.0	877.8	33	11.1	254.3
4 km	10	32	159.8	1953.0	33	13.0	284.9
2 km	20	34	709.9	10943.3	31	23.7	482.0
1 km	40	-	-	-	28	34.0	668.4

TABLE 7.7
GMRES results for Antarctica using 16, 128, 1024, *and* 8192 *cores on respectively larger meshes.*

		ILU			MDSC–AMG		
Horizontal resolution	# layers	# lin. sys.	Avg. its. per solve	Total lin. seconds	# lin sys.	Avg. its. per solve	Total lin. seconds
8 km	5	33	71.2	692.5	33	9.2	220.2
4 km	10	32	160.4	1969.2	32	11.6	245.0
2 km	20	30	426.1	5576.3	30	14.6	293.5
1 km	40	27	881.4	15715.8	27	20.2	377.8

Similar data are shown in Tables 7.6 and 7.7 for Antarctica employing geometry and temperature data from modeling done in [11, 32]. Table 7.6 corresponds to a mesh with hinged peninsulas but where $\beta(x, y)$ is altered to avoid zero values. Table 7.7 considers unaltered $\beta(x, y)$, but with hinged peninsulas eliminated. In these tests the maximum linear iterations is set to 1000 and the residual reduction factor is 10^{-7}. Several ILU solves hit this maximum for the 1 km and 2 km cases. The case denoted by "-" ran for over seven hours without making sufficient progress within either the linear or nonlinear solver. GMRES was used for these results, as it generally performed better than CG, requiring fewer iterations and total time. This was not the case for the Greenland tests and might be a result of severe ill-conditioning associated with the ice shelves, though further study would be needed to confirm this. The iterations associated with MDSC–AMG are generally low, though not completely scalable. The run time improvements relative to ILU(0) are greater in this case than with the Greenland test. This is likely due to the wider Green's function support associated with floating ice shelves (not present in the Greenland test). Furthermore, one can see that the convergence/timings are noticeably better without an artificially perturbed $\beta(x, y)$ field using the mesh where hinged peninsulas are removed. In these examples, the number of elements removed by the island/hinge procedure was about 20 on the finest resolution mesh. As noted, the artificial $\beta(x, y)$ field tends to artificially introduce a significant level of ill-conditioning that seriously affects both the linear and the nonlinear solver, though we have not performed a detailed analysis. It is worth noting that all solvers failed when the unaltered $\beta(x, y)$ field is used with the original mesh where hinged peninsulas have not been removed.

8. Conclusion. A new hybrid matrix dependent semicoarsening algebraic multigrid (MDSC–AMG) preconditioner has been presented. MDSC–AMG was motivated by a complete lack of success with a standard smoothed aggregation AMG (SA-AMG) solver due to difficulties associated with strong anisotropic phenomena. MDSC–AMG can be applied to linear systems where the underlying three-dimensional (3D) mesh has been generated by extruding a two-dimensional (2D) mesh into a third dimension. MDSC–AMG creates the finer level hierarchy meshes by semicoarsening in the

extruded direction in a matrix dependent multigrid fashion. That is, the coarsening and grid transfer operators take advantage of the fact that the mesh is structured in the extruded direction. When repeated semicoarsening constructs a one layer mesh, a standard AMG method can be used to further coarsen and produce additional hierarchy levels. MDSC–AMG has been deployed within an ice sheet modeling simulation. In these calculations, the most important feature from a solver perspective is the presence of very thin finite elements in the extruded direction. These thin elements lead to anisotropic behavior, and this complicates the linear solution process. The hybrid multigrid solver effectively leverages knowledge about the anisotropic direction, and in so doing yields nearly scalable performance over a range of meshes. It generally outperforms an ILU(0) preconditioner and completely outperforms a standard AMG preconditioner. While the solver is not completely optimal (i.e., does not exhibit mesh independent convergence), the performance improvement over ILU(0) is pronounced (a factor greater than 10 in run time gain on larger problems). The more pronounced gain for Antarctica is likely due to its large floating ice shelves that give rise to Green's functions with only a modest horizontal decay (and so ILU is less effective).

In addition to a new multigrid solver, this paper contributes a new algorithm for removing problematic mesh features associated with irregular coastlines, clarifies ordering considerations for ILU, and provides insight into the global/local solution character for thin domains with either a sliding ice/bed interface or a nonsliding/frozen ice/bed interface. In the case of problematic mesh features, singularities arise if an entire piece of floating ice is not connected to the mainland or connected only at a single vertex. Here, we demonstrate that removal of these small components significantly improves the performance of the linear solver. Finally, we note that the MDSC–AMG has not only proved useful within ice sheet modeling, but also has significantly outperformed our standard AMG solver on some thin devices used in engineering analysis.

Acknowledgment. We thank Ali Pinar for suggesting the idea of combining coloring and connected component algorithms to detect hinged peninsulas within graphs.

REFERENCES

[1] R. E. Alcouffe, A. Brandt, J. E. Dendy, Jr., and J. W. Painter, *The multi-grid method for the diffusion equation with strongly discontinuous coefficients*, SIAM J. Sci. Statist. Comput., 2 (1981), pp. 430–454, doi:10.1137/0902035.

[2] H. Blatter, *Velocity and stress fields in grounded glaciers: A simple algorithm for including deviatoric stress gradients*, J. Glaciol., 41 (1995), pp. 333–344.

[3] S. Borm and R. Hiptmair, *Analysis of tensor product multigrid*, Numer. Algorithms, 26 (1999), pp. 200–201.

[4] J. Bramble, J. Pasciak, J. Wang, and J. Xu, *Convergence estimates for multigrid algorithms without regularity assumptions*, Math. Comp., 57 (1991), pp. 23–45.

[5] A. Brandt, S. McCormick, and J. Ruge, *Algebraic multigrid (AMG) for sparse matrix equations*, in Sparsity and Its Applications, D. Evans, ed., Cambridge University Press, Cambridge, UK, 1985, pp. 257–284.

[6] J. Brannick, M. Brezina, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge, *An energy-based AMG coarsening strategy*, Numer. Linear Algebra Appl., 13 (2006), pp. 133–148.

[7] D. Brinkerhoff and J. Johnson, *Data assimilation and prognostic whole ice sheet modelling with the variationally derived, higher order, open source, and fully parallel ice sheet model VarGlaS*, The Cryosphere, 7 (2013), pp. 1161–1184.

[8] P. N. Brown, R. D. Falgout, and J. E. Jones, *Semicoarsening multigrid on distributed memory machines*, SIAM J. Sci. Comput., 21 (2000), pp. 1823–1834, doi:10.1137/S1064827598339141.

[9] E. Bueler and J. Brown, *Shallow shelf approximation as a "sliding law" in a thermodynamically coupled ice sheet model*, J. Geophys. Res., 114 (2009), F03008.
[10] S. Cornford, D. Martin, D. Graves, D. Ranken, A. Le Brocq, R. Gladstone, A. Payne, E. Ng, and W. Lipscomb, *Adaptive mesh, finite volume modeling of marine ice sheets*, J. Comput. Phys., 232 (2013), pp. 529–549.
[11] S. L. Cornford, D. F. Martin, A. J. Payne, E. G. Ng, A. M. Le Brocq, R. M. Gladstone, T. L. Edwards, S. R. Shannon, C. Agosta, M. R. van den Broeke, H. H. Hellmer, G. Krinner, S. R. M. Ligtenberg, R. Timmermann, and D. G. Vaughan, *Century-scale simulations of the response of the west Antarctic ice sheet to a warming climate*, The Cryosphere, 9 (2015), pp. 1579–1600.
[12] K. Cuffey and W. Paterson, *The Physics of Glaciers*, 4th ed., Butterworth-Heinemann, Oxford, UK, 2010.
[13] J. Dendy, *Black box multigrid for nonsymmetric problems*, Appl. Math. Comput., 13 (1983), pp. 261–283.
[14] J. E. Dendy, Jr., M. P. Ida, and J. M. Rutledge, *A semicoarsening multigrid algorithm for SIMD machines*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 1460–1469, doi:10.1137/0913082.
[15] J. Dukowicz, S. Price, and W. Lipscomb, *Consistent approximations and boundary conditions for ice-sheet dynamics from a principle of least action*, J. Glaciol., 56 (2010), pp. 480–496.
[16] O. Gagliardini, T. Zwinger, F. Gillet-Chaulet, G. Durand, L. Favier, B. de Fleurian, R. Greve, M. Malinen, C. Martín, P. Råback, J. Ruokolainen, M. Sacchettini, M. Schäfer, H. Seddik, and J. Thies, *Capabilities and performance of Elmer/Ice, a new-generation ice sheet model*, Geosci. Model Develop., 6 (2013), pp. 1299–1318.
[17] D. Goldberg and O. Sergienko, *Data assimilation using a hybrid ice flow model*, The Cryosphere, 5 (2011), pp. 315–327.
[18] M. Heroux, R. Bartlett, V. Howle, R. Hoekstra, J. Hu, T. Kolda, R. Lehoucq, K. Long, R. Pawlowski, E. Phipps, A. Salinger, H. Thornquist, R. Tuminaro, J. Willenbring, A. Williams, and K. Stanley, *An overview of the Trilinos project*, ACM Trans. Math. Softw., 31 (2005), pp. 397–423.
[19] K. Hutter, *Theoretical Glaciology: Material Science of Ice and the Mechanics of Glaciers and Ice Sheets*, Math. Approaches Geophys., Springer Netherlands, Dordrecht, 1983.
[20] T. Isaac, N. Petra, G. Stadler, and O. Ghattas, *Scalable and efficient algorithms for the propagation of uncertainty from data through inference to prediction for large-scale problems, with application to flow of the Antarctic ice sheet*, J. Comput. Phys., 296 (2015), pp. 348–368.
[21] T. Isaac, G. Stadler, and O. Ghattas, *Solution of nonlinear Stokes equations discretized by high-order finite elements on nonconforming and anisotropic meshes, with application to ice sheet dynamics*, SIAM J. Sci. Comput., 37 (2015), pp. B804–B833, doi:10.1137/140974407.
[22] S. Jachec and J. Jones, *Semicoarsening multigrid for z-level grids*, submitted.
[23] E. Larour, H. Seroussi, M. Morlighem, and E. Rignot, *Continental scale, high order, high spatial resolution ice sheet modeling using the ice sheet system model (ISSM)*, J. Geophys. Res., 117 (2012), F00214.
[24] W. Leng, L. Ju, M. Gunzburger, S. Price, and T. Ringler, *A parallel high-order accurate finite element nonlinear Stokes ice sheet model and benchmark experiments*, J. Geophys. Res., 117 (2012), F00196.
[25] W. Lipscomb, R. Bindschadler, E. Bueler, D. Holland, J. Johnson, and S. Price, *A community ice sheet model for sea level prediction: Building a next-generation community ice sheet model; Los Alamos, New Mexico,* 18–20 *August,* 2008, EOS, 90 (2009), p. 23.
[26] C. Little, M. Oppenheimer, R. Alley, V. Balaji, G. Clarke, T. Delworth, R. Hallberg, D. Holland, C. Hulbe, S. Jacobs, J. Johnson, H. Levy, W. Lipscomb, S. Marshall, B. Parizek, A. Payne, G. Schmidt, R. Stouffer, D. Vaughan, and M. Winton, *Toward a new generation of ice sheet models*, EOS, 88 (2007), pp. 578–579.
[27] D. MacAyeal, C. Hulbe, P. Huybrechts, V. Rommelaere, J. Determann, and C. Ritz, *An ice-shelf model test based on the Ross Ice Shelf, Antarctica*, Ann. Glaciol., 23 (1996), pp. 46–51.
[28] Y. Melnikov, *Influence Functions and Matrices*, Marcel Dekker, New York, 1998.
[29] L. Morland, *Unconfined ice-shelf flow*, in Dynamics of the West Antarctic Ice Sheet, Glaciology and Quaternary Geology 4, C. J. Van der Veen and J. Oerlemans, eds., Springer Netherlands, Dordrecht, 1987, pp. 99–116.

[30] J. Nye, *The distribution of stress and velocity in glaciers and ice-sheets*, Proc. R. Soc. London Ser. A, 239 (1957), pp. 113–133.
[31] F. Pattyn, *A new three-dimensional higher-order thermomechanical ice-sheet model: Basic sensitivity, ice stream development, and ice flow across subglacial lakes*, J. Geophys. Res., 108 (2003), pp. 1–15.
[32] F. Pattyn, *Antarctic subglacial conditions inferred from a hybrid ice sheet/ice stream model*, Earth Planetary Sci. Lett., 295 (2010), pp. 451–461.
[33] M. Perego, S. Price, and G. Stadler, *Optimal initial conditions for coupling ice sheet models to earth system models*, J. Geophys. Res., 119 (2014), pp. 1894–1917.
[34] C. Pflaum, *Fast and Robust Multilevel Algorithms*, Habilitation, Universität Würzburg, Würzburg, Germany, 1998.
[35] D. Pollard and R. Deconto, *Modelling West Antarctic ice sheet growth and collapse through the past five million years*, Nature, 458 (2009), pp. 329–332.
[36] A. Prokopenko, J. Hu, T. Wiesner, C. Siefert, and R. Tuminaro, *MueLu User's Guide* 1.0 *(Trilinos Version* 11.12*)*, Technical Report SAND2014-18874, Sandia National Laboratories, Albuquerque, NM, 2014.
[37] J. W. Ruge and K. Stüben, *Algebraic multigrid*, in Multigrid Methods, Frontiers Appl. Math. 3, S. F. McCormick, ed., SIAM, Philadelphia, 1987, pp. 73–130, doi:10.1137/1.9781611971057.
[38] S. Schaffer, *A semicoarsening multigrid method for elliptic partial differential equations with highly discontinuous and anisotropic coefficients*, SIAM J. Sci. Comput., 20 (1998), pp. 228–242, doi:10.1137/S1064827595281587.
[39] C. Schoof, *Coulomb friction and other sliding laws in a higher-order glacier flow model*, Math. Models Methods Appl. Sci., 20 (2010), pp. 157–189.
[40] C. Schoof and A. Hindmarsh, *Thin-film flows with wall slip: An asymptotic analysis of higher order glacier flows*, Quart. J. Mech. Appl. Math., 63 (2010), pp. 73–114.
[41] J. Schroder, R. Tuminaro, and L. Olson, *A new perspective on strength measures in algebraic multigrid*, Numer. Linear Algebra Appl., 17 (2010), pp. 713–733.
[42] S. Shannon, A. Payne, I. Bartholomew, M. van den Broeke, T. Edwards, X. Fettweis, O. Gagliardini, F. Gillet-Chaulet, H. Goelzer, M. Hoffman, P. Huybrechts, D. Mair, P. Nienow, M. Perego, S. Price, C. Smeets, A. Sole, R. van de Wal, and T. Zwinger, *Enhanced basal lubrication and the contribution of the Greenland ice sheet to future sea-level rise*, Proc. Natl. Acad. Sci. USA, 110 (2013), pp. 14156–14161.
[43] Y. Shapira, *Matrix-Based Multigrid: Theory and Applications*, 2nd ed., Springer, New York, 2008.
[44] S. Solomon, D. Qin, M. Manning, Z. Chen, M. Marquis, K. Averyt, M. Tignor, and H. Miller, eds., *Climate Change* 2007*: The Physical Science Basis. Contribution of Working Group* I *to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change,* Cambridge University Press, Cambridge, UK, 2007.
[45] T. Stocker, D. Qin, G. Plattner, M. Tignor, S. Allen, J. Boschung, A. Nauels, Y. Xia, V. Bex, and P. Midgley, eds., *Climate Change* 2013*: The Physical Science Basis. Contribution of Working Group* I *to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*, Cambridge University Press, Cambridge, UK, 2013.
[46] I. Tezaur, M. Perego, A. Salinger, R. Tuminaro, and S. Price, *Albany/FELIX: A parallel, scalable and robust finite element higher-order Stokes ice sheet solver built for advance analysis*, Geosci. Model Develop., 8 (2015), pp. 1197–1220.
[47] I. Tezaur, R. Tuminaro, M. Perego, A. Salinger, and S. Price, *On the scalability of the Albany/FELIX first-order Stokes approximation ice sheet solver for large-scale simulations of the Greenland and Antarctic ice sheets*, Procedia Comput. Sci., 51 (2015), pp. 2026–2035.
[48] U. Trottenberg, C. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, London, 2001.
[49] P. Vaněk, J. Mandel, and M. Brezina, *Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems*, Computing, 56 (1996), pp. 179–196.
[50] P. Wesseling, *An Introduction to Multigrid Methods*, Wiley, West Sussex, 1992.
[51] P. De Zeeuw, *Matrix-dependent prolongations and restrictions in a blackbox multigrid solver*, J. Comput. Appl. Math., 33 (1990), pp. 1–27.