Robust solution of singularly perturbed problems using multigrid methods: analysis and results in one and two dimensions

ROBUST SOLUTION OF SINGULARLY PERTURBED PROBLEMS USING MULTIGRID METHODS; ANALYSIS AND NUMERICAL RESULTS IN ONE AND TWO DIMENSIONS*

SCOTT MACLACHLAN† AND NIALL MADDEN‡

Abstract. We consider the problem of solving linear systems of equations that arise in the numerical solution of singularly perturbed ordinary and partial differential equations of reaction-diffusion type. Standard discretization techniques are not suitable for such problems and, so, specially tailored methods are required, usually involving adapted or fitted meshes that resolve important features such as boundary and/or interior layers. In this study, we consider classical finite difference schemes on the layer adapted meshes of Shishkin and Bakhvalov. We show that standard direct solvers exhibit poor scaling behaviour when solving the resulting linear systems. We investigate the use of standard robust multigrid preconditioners for these linear systems, and we propose and prove optimality of a new block-structured preconditioning approach.

Key words. Boundary-fitted meshes, robust multigrid, preconditioning

AMS subject classifications. 65F10, 65N06, 65N22, 65N55

1. Introduction. This study addresses the problem of solving linear systems that arise when computing numerical solutions to certain linear singularly perturbed boundary value problems in one and two dimensions. These differential equations are characterised by a small positive parameter, usually denoted as ε, multiplying the highest derivative. The perturbation is "singular" in the sense that, as $\varepsilon \to 0$, the problem becomes ill-posed since the order of the differential equation is reduced but number of boundary conditions remains the same. (For a more formal definition, see [29, Chap. 1].)

The simplest example of a singularly perturbed problem is

$$-\varepsilon^2 u'' + a(x)u'(x) + b(x)u = f(x) \text{ on } (0,1), \quad u(0) = 0, u(1) = 0. \tag{1.1}$$

When $b \equiv 0$ this is known as a *convection-diffusion* problem, whereas if $a \equiv 0$ and $b \neq 0$, it is of *reaction-diffusion* type. Such problems, and their higher-dimensional analogues, are common in mathematical models. Convection-diffusion problems are widespread in many formulations of fluid-flow problems (e.g., in linearised Navier-Stokes equations, and transport problems), and simulation of semi-conductor devices: see [41] and [34] for a more exhaustive list. Coupled systems of reaction-diffusion equations are standard in many biological applications, simulation of chemical reactions, and in hydrodynamic stability. In each case, the solution to the singularly perturbed problem is characterised by the presence of boundary or interior *layers*: narrow regions of the domain where the solution changes rapidly.

The numerical solution of these problems is of significant mathematical interest. Classical numerical schemes that are suitable when ε is $\mathcal{O}(1)$ are often inappropriate as $\varepsilon \to 0$, unless the number of degrees of freedom in one dimension, N, satisfies a relation such as $N = \mathcal{O}(\varepsilon^{-1})$: without this, they may fail the resolve layers—usually the region of most interest; the order of convergence may be diminished for small ε, a phenomenon some times referred to as *locking* [3]; in the case of convection-diffusion problems, the method may become unstable and fail to yield any useful information.

Much of the difficulty in applying numerical methods to problems such as (1.1) stems from the fact that the derivatives of these solutions depend on negative powers of the perturbation parameter. Since estimates for the errors in numerical solution generated by classical schemes depend on bounds for these derivatives, they are not *parameter robust* meaning that they do not hold for arbitrarily small values of the perturbation parameter.

The development of algorithms that are robust with respect to the perturbation parameter, and resolve any layers present, is a very active field of endeavour. See, for example, [18, 29, 33, 41, 45], and the many references there-in. Such so-called *parameter robust* (also known as *uniformly convergent* or "ε-uniform") methods guarantee that the computational effort required to obtain a certain accuracy is the same, for

*This research was supported in part by the Institute for Mathematics and its Applications with funds provided by the National Science Foundation. The research of SM was partially supported by the National Science Foundation under grant DMS-0811022. The research of NM was supported by the Science Foundation of Ireland under Grants No. 08/RFP/CMS1205 and Mathematics Initiative 07/MI/007.

†Department of Mathematics, Tufts University, 503 Boston Avenue, Medford, MA 02155. Email: scott.maclachlan@tufts.edu

‡School of Mathematics, Statistics and Applied Mathematics, National University of Ireland, Galway, Ireland. Email: Niall.Madden@NUIGalway.ie

example, when $\varepsilon = 10^{-6}$ as when $\varepsilon = 10^{-2}$. The majority of these papers consider the application of standard schemes, such as finite difference and finite element methods, on specially constructed (*a priori* or *a posteriori*) fitted meshes, most often the piecewise uniform meshes introduced by Shishkin [33] or the graded meshes devised by Bakhvalov [4], described below in Section 2.2.

These algorithms produce linear systems that must be solved, but it is notable that there are relatively few studies concerning their numerical solution.This is significant because, particularly for problems in more than one dimension, standard solvers are unlikely to be useful; the development of fast robust solvers is important and challenging. Moreover, most studies usually assume that the computational effort for solving the linear systems is independent of ε; few have considered the issue of solving the linear systems with efficiency that is robust with respect to ε. We argue that this should not be taken for granted. Firstly, direct solvers—whose performance should depend only on the matrix structure, and not its entries—are of limited use for problems in more that one dimension. Furthermore, as outlined below in Section 3, such solvers can be surprisingly inefficient for singularly perturbed partial differential equations. Finally, the performance and analysis of most iterative schemes, particularly those involving robust preconditioners, is highly dependent on both the scheme and the underlying differential equation.

Therefore, it is surprising that there is so little in the literature on development of solvers for discretization schemes designed for singularly perturbed problems. For convection-diffusion problems, several studies exist, including [40], which considers the conditioning of the resulting discretizations on certain meshes and the effects of diagonal scaling; Farrell and Shishkin give a short analysis of a Gauss-Seidel method for a convection diffusion problem in [19]; while, in [2], results of experiments with ILU-based preconditioners are reported. Multigrid methods for convection diffusion problems on Shishkin meshes are discussed in [21, 20], where a scalable multigrid scheme is introduced.

For reaction-diffusion problems, most of the multigrid literature focuses on the case of a singularly perturbed problem discretized on a uniform or quasi-uniform mesh. For example, in [36], it is shown that a standard multigrid method applied to the two- and three-dimensional analogues of (1.1) on a quasi-uniform mesh converges with bound independent of mesh size or ε; only a small remark is made about lack of accuracy within the boundary layers. A hierarchical basis approach is discussed in [47]; however, the restriction in this work that the mesh aspect ratios be uniformly bounded is not satisfied by the tensor-products of fitted meshes considered here. In contrast, there is extensive literature on the combination of multigrid methods with adaptive refinement algorithms, ranging from fundamental principles discussed in [7] to recent work on achieving efficiency in massively parallel environments [5]; clearly such approaches yield efficient solution algorithms for the problems considered here. Our interests, however, are in the cases where we have enough *a priori* knowledge to avoid the costs of grid adaptation, but still require efficient solvers on highly non-uniform fitted meshes.

While we focus here on the case of singularly perturbed problems, we note that our approaches could also be applied to other problems where there is a substantial mismatch between the scaling of terms in the discrete equations over different parts of the domain. One such case, of key current interest, arises in the simulation of time-harmonic wave propagation. In the development of discretizations for either the Helmholtz equation or Maxwell's equations, special attention is always paid to the treatment of discrete approximations of the Sommerfeld radiation condition for outgoing waves. In many cases, in order to attain meaningful solutions over a physical domain of interest, a much larger volume needs to be modeled, implementing some form of absorbing boundary layer to prevent unphysical reflections of outgoing waves at the edges of the discretized domain. While this may be achieved simply by introducing attenuative terms within the layer [50, 46], a more common approach is to couple discretized attenuative terms with a significant increase in grid spacing [14, 16, 35, 24, 54]. While some work has been done on the development of efficient multigrid approaches for these grid structures [54, 39], the methods proposed here may lead to simpler algorithms with improved efficiency for these grids.

1.1. Model problems and examples. In this paper, we consider *reaction-diffusion problems* in one and two dimensions. The general form of the one-dimensional problem is:

$$L_\varepsilon u := -\varepsilon^2 u'' + b(x)u = f(x), \quad \text{on } \Omega = (0,1), \tag{1.2a}$$

subject to the boundary conditions

$$u(0) = 0, u(1) = 0. \tag{1.2b}$$

We shall assume that $\varepsilon \in (0,1]$ and that there exists β such that $0 < \beta^2 < b(x)$ for all $x \in \Omega$. It is easy to show that the operator L_ε satisfies a maximum principle (see, e.g., [38]), and that the problem possesses a unique solution. If $\varepsilon \ll 1/\beta$, then the problem is singularly perturbed, and we expect its solution to exhibit boundary layers, with rapid changes in $u(x)$, near $x = 0$ and $x = 1$.

As an example, consider the problem

$$-\varepsilon^2 u'' + u = e^x \text{ on } (0,1), \quad u(0) = u(1) = 0. \tag{1.3}$$

When $\varepsilon < 1$, the solution can be expressed as

$$u(x) = \frac{e^{-x/\varepsilon}(e^{1-1/\varepsilon} - 1) + e^{-(1-x)/\varepsilon}(e^{1/\varepsilon} - e)}{(1-\varepsilon^2)(1-e^{-2/\varepsilon})} + \frac{e^x}{1-\varepsilon^2}.$$

One can consider the expressions $\exp(-x/\varepsilon)$ and $\exp(-(1-x)/\varepsilon)$ as representing the layer components. A typical solution exhibiting these layers is shown in Figure 1.1(a).

Our model two-dimensional problem is

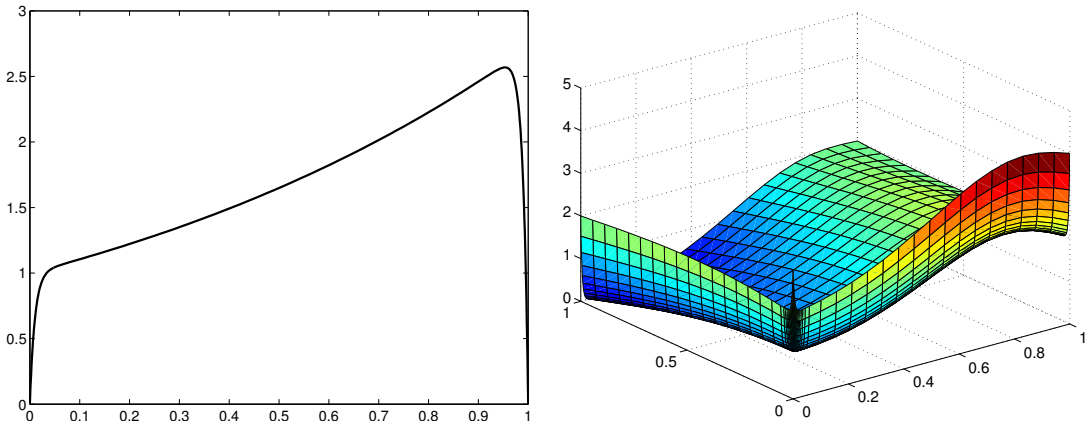
$$-\varepsilon^2 \Delta u + b(x,y)u = f(x,y) \quad \text{ on } \Omega := (0,1)^2, \qquad u(x,y) = g(x,y) \ \text{ on } \partial\Omega, \tag{1.4}$$

where, again, we assume that there is a positive β such that $0 < \beta^2 < b(x,y)$ for all $(x,y) \in \Omega$. Subject to sufficient regularity and compatibility of b, f and g, this problem has a unique solution: we refer readers to, e.g., [25] for technical details. When ε is small, the solution may have boundary and corner layers.

As an example of a two-dimensional problem, we consider a variant on a standard test problem (see, e.g., [12]). Although, in general, one would expect solutions to (1.4) to have four boundary and four corner layers, for simplicity of exposition, we have constructed one that has only two boundary layers, near the edges $x = 0$ and $y = 0$, and a corner layer near $(0,0)$. We take $b(x,y) = 1$, choose f and g so that

$$u = x^3(1+y^2) + \sin(\pi x^2) + \cos(\pi y/2) + (1+x+y)\big(e^{-2x/\varepsilon} + e^{-2y/\varepsilon}\big). \tag{1.5}$$

This solution, in the case $\varepsilon = 10^{-2}$, is shown in Figure 1.1(b).



(a) Solution to (1.3) with $\varepsilon = 10^{-2}$ (b) Solution to (1.5) with $\varepsilon = 10^{-2}$

FIG. 1.1. *Examples of solutions to one- and two-dimensional singularly perturbed reaction-diffusion problems*

1.2. Outline. In Section 2, we introduce the standard finite difference schemes for one- and two-dimensional problems, demonstrating that they are not parameter robust if employed on a uniform mesh. This motivates the introduction of two standard parameter robust finite difference methods: one based on the piecewise uniform meshes of Shishkin, and the other on the graded meshes of Bakhvalov. In Section 3, we show that standard direct solvers perform poorly for these discretizations, with the CPU time required depending badly on ε. This motivates the introduction of iterative methods in Section 4, beginning with multigrid methods and a boundary-layer preconditioning approach for one-dimensional problems. We then

discuss multigrid methods for two-dimensional problems, focusing on the Algebraic Multigrid (AMG) and Black Box Multigrid (BoxMG) methods, before proposing a two-dimensional boundary-layer preconditioning approach in Section 4.2.4. Stopping criteria for these iterative methods are discussed in Section 4.3, followed by the results of numerical experiments in Section 5. A brief discussion of generalisations to higher-dimensional problems appears in Section 6, with conclusions and some comments on other extensions of this work in Section 7.

1.2.1. Notation. We denote by N the number of intervals in one dimension of a mesh, and by c and C generic positive constants that are independent of ε and N. Since we are mainly concerned with discretization by finite difference methods, we use $\|\cdot\|_\infty$ to denote the discrete maximum norm on a given mesh:

$$\|u\|_\infty = \begin{cases} \max\limits_{x_i \in \omega^N} |u(x_i)| & \text{on the one-dimensional mesh } \omega^N, \\ \max\limits_{(x_i,y_j)\in\omega^{N\times N}} |u(x_i,y_j)| & \text{on the two-dimensional mesh } \omega^{N\times N}. \end{cases}$$

The continuous analogue on the domain Ω is denoted $\|\cdot\|_{\Omega,\infty}$. Other norms used are the discrete ℓ_2 norm, $\|\cdot\|_2$, and the A-norm for symmetric and positive-definite matrices, A, $\|V\|_A = \left(V^T A V\right)^{1/2}$ for any vector, V. We reserve the use of $\|\cdot\|$ for denoting a generic norm, which may be any of the above.

We will use two parameters to measure the singularly perturbed nature of a discrete problem. Define $\delta_N = (\varepsilon N/\beta)^2$ to indicate if a problem is singularly perturbed relative to a mesh with N points, when $\delta_N \ll 1$. For the purposes of the theory developed in Section 4, we use the term *boundary-fitted mesh* to mean a mesh that is uniform in the interior of the domain, but condenses near the boundary. This uniform mesh-width away from boundaries is denoted h_I, and we define $\delta_h = (\varepsilon/(h_I\beta))^2$ to indicate the diagonal dominance of the matrix over the interior degrees of freedom.

2. Parameter robust methods. The robust solution of singularly perturbed problems can be achieved using fitted operator schemes (i.e., specially designed methods, but used, say, on a uniform mesh) or fitted mesh methods—standard schemes employed on specially designed meshes. The latter approach has received most attention of late, not least because they are easier to generalise to high-dimensional problems. These fitted mesh methods are categorised as either *a priori* or *a posteriori* (equivalently, "fitted" or "adaptive"). An *a priori* method is constructed based on a careful analysis of the asymptotic properties of the solution and its derivative; most published work considers such schemes. Alternatively, adaptive schemes may be generated based on *a posteriori* analysis: see, for example, [27, 9]. In this paper, we consider only fitted mesh methods. However, those meshes generated by adaptive schemes tend to be very similar to the Bakhvalov meshes we discuss below and, as such, we expect that similar techniques could be used for the meshes generated by adaptive schemes.

2.1. Finite difference scheme for one dimensional problems. Given an arbitrary one-dimensional grid $\omega^N = \{0 = x_0 < x_1 < \cdots < x_N = 1\}$, with $h_i = x_i - x_{i-1}$ for $i = 1, \ldots N$, the natural second-order finite-difference discretization of problem (1.2) is given by

$$-\frac{\varepsilon^2}{\bar{h}_i}\left(\frac{U_{i+1}-U_i}{h_{i+1}} - \frac{U_i - U_{i-1}}{h_i}\right) + b(x_i)U_i = f(x_i), \quad \text{for } i = 1, \ldots N-1,$$
$$U_i = 0 \quad \text{for } i \in \{0, N\},$$

where $\bar{h}_i = (h_{i+1} + h_i)/2$. As this provides a potentially unsymmetric discretization of a Hermitian operator (since $\bar{h}_i \neq \bar{h}_{i+1}$ on an arbitrary mesh), a natural approach is to symmetrise this operator, both for consistency with the continuum equations and for the many advantages of dealing with symmetric and positive-definite linear systems. The symmetrised finite-difference method for the problem (1.2) is given by multiplying by a diagonal matrix with entries $\bar{h}_i$, giving

$$\begin{aligned} -\varepsilon^2\left(\frac{U_{i+1}-U_i}{h_{i+1}} - \frac{U_i - U_{i-1}}{h_i}\right) + \bar{h}_i b(x_i)U_i &= \bar{h}_i f(x_i), \quad \text{for } i = 1, \ldots N-1, \\ U_i &= 0 \quad \text{for } i \in \{0, N\}. \end{aligned} \tag{2.1}$$

In matrix form, we write the symmetrically scaled equations, with boundaries eliminated, as $AU = F$.

If the mesh ω^N is uniform then, in both theory and practise, one must make the unreasonable assumption that N is $\mathcal{O}(\varepsilon^{-1})$ in order to obtain a convergent, layer-resolving method. In Section 2.3 below, we give an example of numerical results obtained on a uniform mesh, demonstrating this point. Intuitively, it seems likely that a robust scheme could be generated if h_i is $\mathcal{O}(\varepsilon)$, but only in the region of the layers. This is indeed the case, but the construction of such meshes must be very careful if one is to rigorously prove robustness. We consider two examples of such meshes below: a simple piecewise uniform mesh, and a more accurate graded mesh.

2.2. Fitted meshes for one dimensional problems.

2.2.1. Shishkin meshes. The most popular boundary-fitted mesh for singularly perturbed problems to be found in the mathematical literature is certainly the piecewise uniform mesh of Shishkin [33]. For a problem such as (1.2), it may be formed as follows: assuming the number of mesh intervals N is divisible by 4, define the *mesh transition point* to be

$$\tau_S = \min\{\frac{1}{4}, 2\frac{\varepsilon}{\beta}\ln N\}, \tag{2.2}$$

and divide $[0,1]$ into subintervals $[0,\tau_S]$, $[\tau_S, 1-\tau_S]$ and $[1-\tau_S, 1]$. A piecewise-uniform mesh is constructed by subdividing $[\tau_S, 1-\tau_S]$ into $N/2$ equidistant mesh intervals, and subdividing each of $[0,\tau_S]$ and $[1-\tau_S, 1]$ into $N/4$ equidistant mesh intervals, as shown in Figure 2.1. The construction can also be described in terms of a *mesh generating function*.

DEFINITION 2.1. *A function* $\psi : [0,1] \to [0,1]$ *is a* mesh generating function *for the mesh* $\omega^N = \{x_0, x_1, \ldots, x_N\}$ *if* $x_i = \psi(i/N)$. Following [29], a suitable Shishkin mesh for (1.2) can be formed by taking the transition point τ_S as defined in (2.2), and then

$$\psi(t) = \begin{cases} 4t\tau_S & t \le \frac{1}{4}, \\ 2(1-\tau_S)(t-\frac{1}{4}) + 2\tau_S(\frac{3}{4}-t) & \frac{1}{4} < t < \frac{3}{4}, \\ 4(1-\tau_S)(1-t) + 4\tau_S(t-\frac{3}{4}) & t \ge \frac{3}{4}. \end{cases}$$

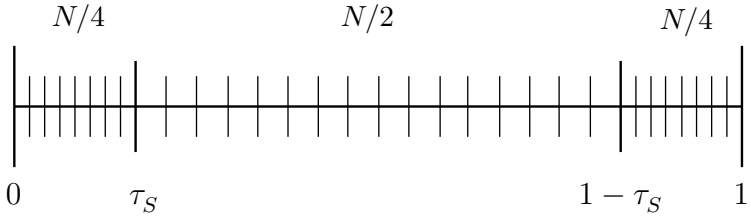


FIG. 2.1. *A Shishkin mesh for a reaction-diffusion problem*

This mesh was first proposed in [44], and an accessible analysis of the uniform convergence of the finite difference method (2.1) applied to the linear reaction-diffusion problem (1.2) is given in [33, Chap. 6]. That analysis uses ideas involving decomposition of the solution into regular and layer parts and exploits the fact that the continuous and discrete operators satisfy maximum principles. A unified treatment based on discrete Green's functions is given in [29], from which it quickly follows that there is a constant C which is independent of both N and ε such that that the solution to the finite difference scheme (2.1) satisfies

$$\|u - U\|_\infty = \max_{i=0,1,\ldots,N} |u(x_i) - U_i| \le CN^{-2}\ln^2 N. \tag{2.3}$$

Since C does not depend on ε, this qualifies as a parameter robust estimate. It is not immediately obvious from (2.3) that the numerical solution also resolves the boundary layers. However, one can also show (see, e.g., [29, Thm. 6.12]) that

$$\|u - \bar{U}\|_{\Omega,\infty} = \max_{0\le x\le 1} |u(x) - \bar{U}(x)| \le CN^{-2}\ln^2 N,$$

where $\bar{U}$ is the piecewise linear interpolant to U.

2.2.2. Bakhvalov meshes. In the case where ε is $\mathcal{O}(1)$, the scheme (2.1) applied on a uniform mesh should yield a numerical solution with error that is bounded by terms of $\mathcal{O}(N^{-2})$. The logarithmic factor that spoils (2.3) slightly is the price one pays for the simplicity of employing a piecewise uniform mesh. To regain full second-order convergence, one could use the more sophisticated nonuniform boundary-fitted mesh of Bakhvalov [4]. Like the Shishkin mesh, it is uniform over the interior of the domain (a fact which simplifies our analysis later in Section 4.1.3), but is graded within the boundary layer. The mesh generating function, ψ, is

$$\psi(t) = \begin{cases} \chi(t) := -\dfrac{\sigma\varepsilon}{\beta}\ln\left(1-\dfrac{t}{q}\right) & \text{for } t \in [0,\tau_B], \\ \phi(t) := \chi(\tau_B) + \psi'(\tau_B)(t-\tau_B) & \text{for } t \in [\tau_B, 1/2], \\ 1-\psi(1-t) & \text{for } t \in (1/2,1], \end{cases}$$

where the Bakhvalov mesh transition point τ_B is chosen so that $\psi \in C^1[0,1]$. This is done by solving the (simple) nonlinear problem $(1-2\tau_B)\phi'(\tau_B) = 1-2\chi(\tau_B)$: for example the iteration

$$\tau_0 = 0, \qquad \chi'(\tau_{i+1}) = \frac{1-\chi(\tau_i)}{1-\tau_i}, i = 0, 1, \ldots,$$

converges rapidly (see [29, p. 7]). The mesh parameters q and σ are user-chosen and control, respectively, the proportion of mesh points in the layer regions, and the grading of the mesh within the layer. A diagram of such a mesh is shown in Figure 2.2, and the mesh generating functions for both the Shishkin and Bakhvalov meshes are shown in Figure 2.3.

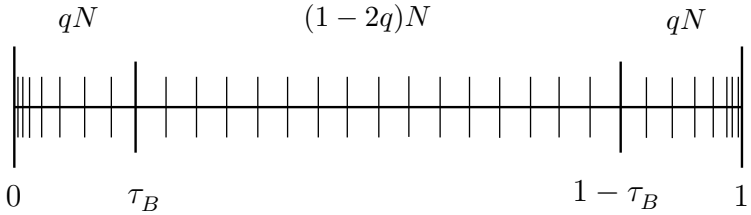


FIG. 2.2. *A Bakhvalov mesh for a reaction-diffusion problem*

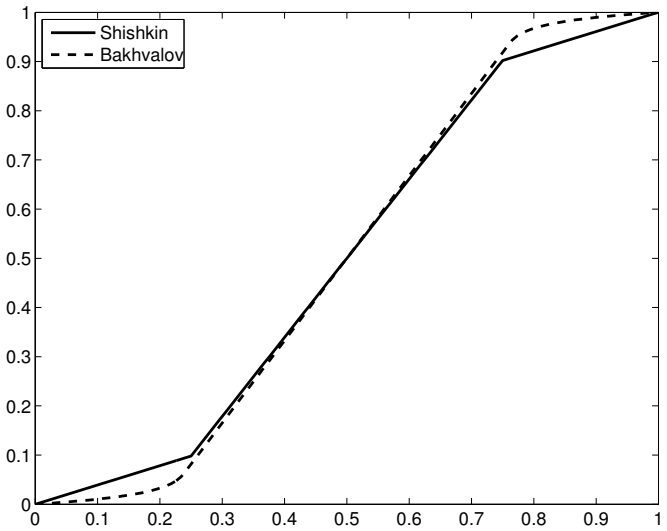


FIG. 2.3. *Mesh generating functions for Shishkin and Bakhvalov meshes with* $N = 2^7$ *and* $\varepsilon = 10^{-2}$. *The parameters for the Bakhvalov mesh are* $q = 1/2$ *and* $\sigma = 2$

For the Bakhvalov mesh just described, it can be proved [29] that there is a constant C independent of ε and N such that

$$\|u - U\|_\infty \leq CN^{-2},$$

and, furthermore, as with the Shishkin mesh, the piecewise linear interpolant to the Bakhvalov solution is second-order convergent.

2.3. Fitted methods for two dimensional problems. For the two dimensional problem (1.4), we employ the natural extension of the method (2.1): a standard finite difference scheme on a tensor-product grid. Let ω_x^N and ω_y^N be arbitrary meshes, each with N intervals on $[0,1]$. Set $\omega^{N\times N} = \{(x_i, y_j)\}_{i,j=0}^N$ to be the Cartesian product of ω_x^N and ω_y^N. Taking $h_i = x_i - x_{i-1}$, $k_j = y_j - y_{j-1}$, $\bar{h}_i = (x_{i+1} - x_{i-1})/2$, and $\bar{k}_j = (y_{j+1} - y_{j-1})/2$, we define the symmetrised 5-point second-order central difference operator:

$$\Delta^N := \begin{pmatrix} & \frac{\bar{h}_i}{k_{j+1}} & \\ \frac{\bar{k}_j}{h_i} & -\Big(\bar{k}_j\big(\frac{1}{h_i} + \frac{1}{h_{i+1}}\big) + \bar{h}_i\big(\frac{1}{k_j} + \frac{1}{k_{j+1}}\big)\Big) & \frac{\bar{k}_j}{h_{i+1}} \\ & \frac{\bar{h}_i}{k_j} & \end{pmatrix}.$$

The resulting numerical scheme is:

$$\begin{aligned} \big(-\varepsilon^2\Delta^N + \bar{h}_i\bar{k}_j b(x_i,y_j)\big)U_{i,j} = \bar{h}_i\bar{k}_j f(x_i,y_j) \qquad & i = 1,\ldots,N-1, j = 1,\ldots N-1, \\ U_{i,j} = g(x_i,y_j), \qquad & i \in \{0,N\},\ j \in \{0,N\}. \end{aligned} \tag{2.4}$$

Again, we write the linear system as $AU = F$.

As in one dimension, this scheme will not generate satisfactory numerical solutions if employed on a uniform mesh. Consider Table 2.1 below, which gives the maximum pointwise errors in the numerical solution to (1.5) for various values of N and ε on a uniform mesh. When ε is $\mathcal{O}(1)$, the method is second order accurate as expected. However, when ε is small, the reported error *increases* as N increases. This is because, if $N \ll 1/\varepsilon$, the boundary layers are not resolved. However, as N increases one obtains more mesh points close to, or within, the layers. Since these are the regions where the problem is most difficult to solve, the observed error increases. The most interesting row in Table 2.1 occurs for $\varepsilon^2 = 10^{-6}$, where we see an initial increase in the maximum errors as N increases, until we reach the point where $N = \mathcal{O}(\varepsilon^{-1})$. As we increase N from here, the pointwise error decreases, falling off as N^{-2}.

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	9.762×10^{-5}	2.441×10^{-5}	6.103×10^{-6}	1.526×10^{-6}	3.814×10^{-7}	9.447×10^{-8}
10^{-2}	1.098×10^{-3}	2.750×10^{-4}	6.878×10^{-5}	1.720×10^{-5}	4.299×10^{-6}	1.075×10^{-6}
10^{-4}	1.133×10^{-1}	3.147×10^{-2}	8.094×10^{-3}	2.053×10^{-3}	5.142×10^{-4}	1.286×10^{-4}
10^{-6}	3.149×10^{-2}	1.126×10^{-1}	2.315×10^{-1}	1.607×10^{-1}	4.715×10^{-2}	1.298×10^{-2}
10^{-8}	3.301×10^{-4}	1.313×10^{-3}	5.212×10^{-3}	2.052×10^{-2}	7.702×10^{-2}	2.054×10^{-1}
10^{-10}	3.302×10^{-6}	1.316×10^{-5}	5.253×10^{-5}	2.099×10^{-4}	8.382×10^{-4}	3.343×10^{-3}
10^{-12}	3.302×10^{-8}	1.316×10^{-7}	5.253×10^{-7}	2.099×10^{-6}	8.393×10^{-6}	3.356×10^{-5}

TABLE 2.1

Maximum pointwise errors for problem (1.5) *solved by a finite difference method on a uniform mesh*

2.3.1. A Shishkin mesh for a two dimensional problem. We shall now construct fitted meshes which can be used to generate uniformly convergent numerical solutions. Because the test problem (1.4) features one layer in each coordinate direction, we modify slightly the construction of the Shishkin mesh of Section 2.2.1 for the one-dimensional problem. Similar to (2.2), we choose the transition point $\tau_S = \min\{1/2, 2(\varepsilon/\beta)\ln N\}$, and take both ω_x^N and ω_y^N to be piecewise uniform meshes with $N/2$ intervals on each of $[0,\tau_S]$ and $[\tau_S,1]$. The resulting Cartesian product mesh is shown in Figure 2.4(a).

The error estimate for the one dimensional problem can be extended to the two dimensional case. In particular, in [12], the following parameter robust error estimate is proved: there is a constant C independent of N and ε such that

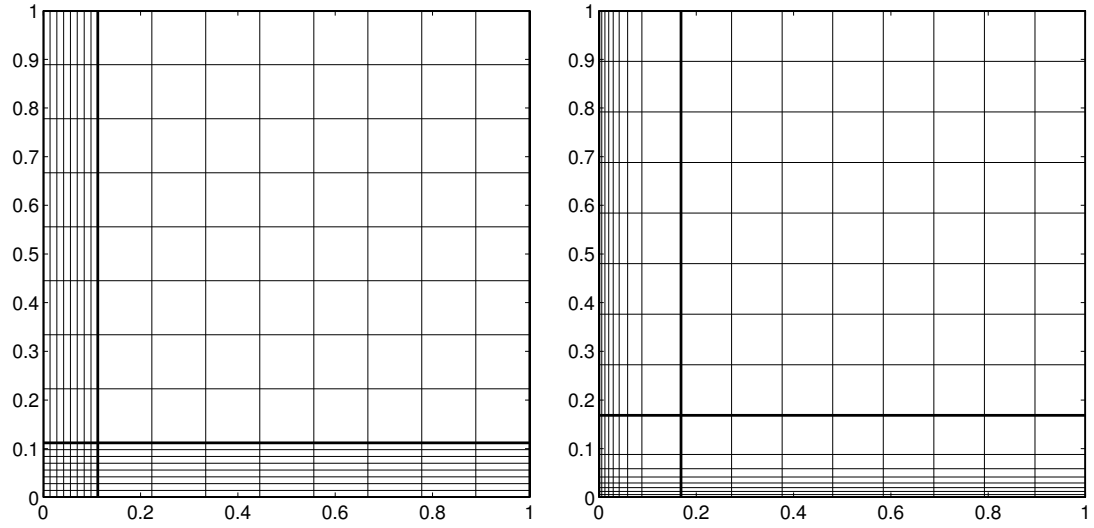
$$\|u - U\|_\infty \leq CN^{-2}\ln^2 N. \tag{2.5}$$

In Table 2.2, we report the maximum pointwise errors when (1.4) is solved on the Shishkin mesh just described. When ε is large, the mesh is uniform and, so, the results are identical to those reported in Table 2.1. For fixed N, we notice that the error initially increases as ε decreases; this corresponds to the mesh transitioning from being uniform to being piecewise uniform. As predicted by the theory, although the rate of convergence initially falls from $\mathcal{O}\big(N^{-2}\big)$ for $\varepsilon = 1$ to $\mathcal{O}\big(N^{-2}\ln^2 N\big)$ for smaller ε, the accuracy with respect to N is otherwise guaranteed.

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	9.762×10^{-5}	2.441×10^{-5}	6.103×10^{-6}	1.526×10^{-6}	3.814×10^{-7}	9.447×10^{-8}
10^{-2}	1.098×10^{-3}	2.750×10^{-4}	6.878×10^{-5}	1.720×10^{-5}	4.299×10^{-6}	1.075×10^{-6}
10^{-4}	4.996×10^{-3}	1.648×10^{-3}	5.227×10^{-4}	1.614×10^{-4}	4.883×10^{-5}	1.453×10^{-5}
10^{-6}	5.105×10^{-3}	1.684×10^{-3}	5.354×10^{-4}	1.654×10^{-4}	5.008×10^{-5}	1.490×10^{-5}
10^{-8}	5.116×10^{-3}	1.692×10^{-3}	5.370×10^{-4}	1.658×10^{-4}	5.023×10^{-5}	1.495×10^{-5}
10^{-10}	5.116×10^{-3}	1.692×10^{-3}	5.372×10^{-4}	1.660×10^{-4}	5.025×10^{-5}	1.495×10^{-5}
10^{-12}	5.116×10^{-3}	1.692×10^{-3}	5.372×10^{-4}	1.660×10^{-4}	5.025×10^{-5}	1.496×10^{-5}

TABLE 2.2
Maximum pointwise errors for problem (1.5) *solved on a Shishkin mesh*

2.3.2. A Bakhvalov for a two dimensional problem. Taking $\omega_x^N = \omega_y^N$ to be the one dimensional graded mesh described Section 2.2.2, adjusted to account for the fact that there is only one layer in each coordinate direction, and taking the Cartesian product mesh, $\omega^{N\times N}$, generates the two-dimensional Bakhvalov mesh, shown in Figure 2.4(b).



(a) Shishkin mesh (b) Bakhvalov mesh

FIG. 2.4. *Fitted meshes for problem* (1.5) *when* $N = 16$, $\varepsilon = 2 \times 10^{-2}$

Kellogg et al. [26] prove the ε-uniform error estimate: there is a constant C independent of ε and N such that

$$\|u - U\|_\infty \leq CN^{-2}. \tag{2.6}$$

This bound is seen to be sharp in the results of numerical experiments for Problem (1.5) reported in Table 2.3 below. The errors are clearly uniform in ε, and the method is fully second-order: the spoiling logarithmic factor associated with the Shishkin mesh is absent and, so, the errors are smaller than those reported in Table 2.2.

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	9.762×10^{-5}	2.441×10^{-5}	6.103×10^{-6}	1.526×10^{-6}	3.814×10^{-7}	9.447×10^{-8}
10^{-2}	9.907×10^{-5}	2.479×10^{-5}	6.196×10^{-6}	1.549×10^{-6}	3.873×10^{-7}	9.685×10^{-8}
10^{-4}	2.858×10^{-5}	7.170×10^{-6}	1.794×10^{-6}	4.486×10^{-7}	1.122×10^{-7}	2.802×10^{-8}
10^{-6}	2.864×10^{-5}	7.188×10^{-6}	1.803×10^{-6}	4.511×10^{-7}	1.128×10^{-7}	2.818×10^{-8}
10^{-8}	2.880×10^{-5}	7.241×10^{-6}	1.814×10^{-6}	4.537×10^{-7}	1.135×10^{-7}	2.840×10^{-8}
10^{-10}	2.881×10^{-5}	7.245×10^{-6}	1.815×10^{-6}	4.543×10^{-7}	1.136×10^{-7}	2.841×10^{-8}
10^{-12}	2.881×10^{-5}	7.245×10^{-6}	1.815×10^{-6}	4.543×10^{-7}	1.137×10^{-7}	2.850×10^{-8}

TABLE 2.3
Maximum pointwise errors for problem (1.5) *solved by a finite difference method on a Bakhvalov mesh*

3. Solving a 2D problem with a Direct Solver. Solving the one-dimensional problem is straightforward with a direct solver, as the resulting tridiagonal matrices can be factored with no fill-in. Thus, here, we consider the challenges of applying direct solvers to the two-dimensional problem.

Our programmes that generate the results throughout this paper were coded in C and executed on a Beowulf cluster using a single core of a node with an AMD Opteron 2427, 2200 MHz processor with 32Gb of RAM. In this and the previous section, we use `CHOLMOD` (supernodal sparse Cholesky factorization and update/downdate) Version 1.7.1 to solve the sparse symmetric linear systems; see [11, 13]. In Table 3.1, we show the time in seconds, averaged over three runs, required to solve the linear systems on Bakhvalov meshes that yield results given in Table 2.3. For a given ε, we see growth in these times that, for large N, scales as N^3 (increasing by factors of roughly eight when N is doubled), as expected given the known $\mathcal{O}(N^3)$ complexity of the nested dissection algorithm for these grids [22] and the $\mathcal{O}(N^3)$ lower bound for this complexity [23]. For fixed N, however, we observe that, rather remarkably, the amount of time required to solve the linear system depends quite badly on the perturbation parameter. This is in spite of the fact that, for a given N, the matrices for different ε have exactly the same size and structure; the only difference is the scaling of some entries due to the difference in both ε and the local mesh width. Similar results have been observed with other direct solvers, including MA57 [17], on different processors and architectures, and with different meshes, including uniform and Shishkin meshes. As we now explain, the degradation in the performance of the direct solvers is not related to their implementation but, rather, the specific nature of the discretized singularly perturbed problem.

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	0.08	0.41	2.80	19.32	203.25	1736.27
10^{-2}	0.07	0.39	2.80	19.36	203.62	1735.94
10^{-4}	0.07	0.40	2.82	19.50	203.71	1737.89
10^{-6}	0.07	1.00	12.11	93.06	891.72	7716.35
10^{-8}	0.16	1.34	11.29	76.75	495.31	2787.91
10^{-10}	0.19	1.24	8.82	49.42	353.53	1581.63
10^{-12}	0.19	1.16	7.10	38.49	264.58	1211.50

TABLE 3.1
Cholesky (CHOLMOD) solve times for linear systems generated by a finite difference method applied on a Bakhvalov mesh

Writing the linear system for the finite difference solution to (2.4) as $AU = F$, while neglecting boundary conditions, gives A as an $(N-1)^2 \times (N-1)^2$, banded, symmetric and positive-definite matrix with bandwidth of $N-1$. It should be easily solved using a Cholesky factorisation, $A = L^T L$, though the factors will experience fill-in within the band. When considering the rows/columns of these matrices in which the off-diagonal entries are much smaller than the diagonals (i.e., those where the mesh spacing $h_I \gg \varepsilon$), the successive fill-ins are decaying in magnitude, scaling as $(\varepsilon^2/h_I^2)^k$ for the k^{th} band of fill. If the bandwidth is small, then this poses

no problem for floating-point calculation. If, on the other hand, N is large, then floating-point underflows may occur.

In IEEE standard double precision, numbers are typically represented with 64 bits as $\pm X \times 2^{Y-1023}$ where 52 bits are used to store the significand, X, 11 bits are used to store the exponent, Y, and the remaining bit stores the sign of X. If $0 < Y < 2047$ (a "normal" number), then X is assumed to be a binary decimal with a leading 1 (an implied 53^{rd} bit); the smallest number that can be represented this way is, then, when $Y = 1$ and $X = 0$, giving $2^{-1022} \approx 10^{-308}$. When $Y = 0$, the implied bit in X is taken to be a leading 0 and the exponent is fixed at -1022, allowing representation of nonzero numbers as small as is $2^{-52} \times 2^{-1022} \approx 5 \times 10^{-324}$ when the binary decimal in X has only a single one, in its last entry; anything smaller than this is rounded to zero (when $X = 0$ and $Y = 0$). The use of such "subnormal" numbers allows for gradual reduction in the precision of stored numbers. For a full discussion of IEEE double precision see, for example, [37]. Most processors, however, do not provide hardware support for arithmetic with subnormal numbers and, instead, a compiler must rely on a software implementation, which is significantly slower [28]. The variation in timings seen with ε in Table 3.1 are due to the introduction of subnormal numbers in the fill-in generated by Cholesky.

To demonstrate this effect, in Table 3.2, we give the number of nonzero entries in the Cholesky factors produced by CHOLMOD for a range of values of N and ε, corresponding to those shown in Table 3.1, as well as the number of subnormal entries. For $\varepsilon \leq 10^{-2}$, there are no subnormal numbers, and the scaling in N is, notably, worse than $\mathcal{O}(N^2)$. For smaller ε, the number of subnormals increases and the number of nonzero numbers decreases. (This latter observation is important since these "new" zeros result from underflow involving subnormals and, so, are expensive to compute.) Although they are relatively few compared to the number of nonzero entries, they are sufficient to greatly increase the computation time.

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	350,112	1,833,813	9,425,559	45,671,436	183,759,251	831,532,333
	0	0	0	0	0	0
10^{-2}	350,112	1,833,813	9,425,559	45,671,436	183,759,251	831,532,333
	0	0	0	0	0	0
10^{-4}	350,112	1,833,813	9,425,559	45,671,436	183,759,251	831,532,333
	0	0	0	0	0	0
10^{-6}	350,112	1,828,215	9,293,727	44,499,256	179,511,201	808,690,367
	0	4,338	22,596	108,387	573,033	2,852,019
10^{-8}	347,351	1,717,341	8,266,871	37,946,547	147,162,291	625,420,613
	1,146	8,488	56,295	316,104	1,121,348	4,956,624
10^{-10}	335,322	1,614,213	7,535,505	33,695,760	130,437,185	544,870,886
	1,915	10,008	77,691	283,348	1,111,292	4,422,916
10^{-12}	322,935	1,534,747	7,019,889	31,076,314	120,736,814	504,478,967
	2,176	11,467	58,065	305,428	991,728	3,803,770

TABLE 3.2
Number of nonzero entries (top) and subnormal numbers (bottom) in Cholesky factors generated by CHOLMOD.

This phenomenon is a natural feature of discretizations of multidimensional problems. For problems in 1D, on the other hand, the efficiency of direct solvers should be independent of ε for standard discretizations, because of the typical tridiagonal structure of the resulting matrices. Since there is no zero band in the original matrices, there can be no fill-in of the factors and, so, no subnormal numbers will be computed. However, the difficulties highlighted above are likely to be exacerbated further for problems in three or more dimensions where larger bandwidths occur.

It is worth noting that this behaviour can be overcome with compiler directives. For the GCC family of compilers, the `-funsafe-math-optimizations` option turns on certain optimizations that violate the IEEE floating point standard, including the treatment of subnormal numbers. With this option enabled, we see general decreases in the factorization and solve times as ε decreases and the number of nonzero entries retained in the factors drops, with no observable degradation in the solution accuracy. However, we now see large (up to 60%) increases in these times for the smallest values of ε that are not consistent with the number of nonzeros in the factors. This suggests that simply ignoring subnormal entries in the factors may still not be enough to ensure a favorable scaling of direct solvers with ε, especially as doing so relies on compiler and

architecture dependent implementations of variations to the IEEE standard.

4. Iterative Methods.

4.1. Solving the one-dimensional system. From a practical viewpoint, there is no benefit to be gained in terms of computational time or memory requirements from considering iterative approaches to solving discretizations of one-dimensional problems such as (1.1) by a standard 3-point scheme. The tridiagonal structure of the discretization matrices ensures that sparse direct solvers are optimal, and specialized approaches, such as cyclic reduction or the Thomas algorithm, are well-known in the literature. It is, nonetheless, worth considering iterative approaches for these matrices to inform the development of iterative approaches for two- (or multi-) dimensional problems. In particular, the theory for the boundary-layer preconditioner developed in Section 4.1.3 develops the key ideas needed for the two-dimensional analysis in Section 4.2.4.

4.1.1. Geometric Multigrid. Multigrid methods are widely regarded as being among the most efficient iterative approaches for solving discretizations of elliptic PDEs, such as those considered here. The key to their efficiency lies in the combination of two processes, relaxation and coarse-grid correction, that effectively damp complementary components of the error in any approximation to the solution of the discrete system. For uniformly elliptic operators discretized on uniform meshes, simple analysis shows that standard relaxation approaches, such as the Jacobi or Gauss-Seidel iterations, effectively damp errors that are oscillatory on the scale of the grid, while coarse-grid correction effectively resolves the smooth components. In this setting, coarse-grid correction typically involves discretizing the problem on a hierarchy of successively coarser meshes and combining relaxation on all levels of this hierarchy with simple geometric operations to transfer residuals from finer grids to coarser ones and interpolate corrections from coarser grids to finer ones. See [8, 49] for full details.

In the context of singularly perturbed problems and non-uniform grids, more careful treatment must be given to the coarse-grid correction process to obtain optimal efficiency, even for one-dimensional problems. While the partitioning of errors into "smooth" and "oscillatory" components is still intuitive, much more technical theory is needed to prove optimality [53, 6]. In practice, however, only small changes are needed in the multigrid algorithm. In this paper, we consider iterations and preconditioners based on the multigrid V-cycle; thus, a single iteration of the multigrid cycle can be written in recursive form as

Algorithm 1: $U^{(1)} = MG(U^{(0)}, F, N)$.

1. Apply relaxation to $AU = F$ with initial guess $U^{(0)}$, producing $U^{(r)}$.
2. Compute $F_c = R(F - AU^{(r)})$, for restriction matrix, R.
3. Compute $U_c = MG(0, F_c, N/2)$.
4. Compute $U^{(c)} = U^{(r)} + PU_c$, for interpolation matrix, P.
5. Apply relaxation to $AU = F$ with initial guess $U^{(c)}$, producing $U^{(1)}$.

Thus, the algorithm takes, as input, an initial guess, $U^{(0)}$, and right-hand side, F, of length $N-1$ (in one dimension). On any level, relaxation is applied based on the matrix, A, which is taken to be the discretization of the given differential equation with $N-1$ degrees of freedom after elimination of boundary conditions. A coarse-grid right-hand side is computed by restricting the fine-grid residual, and the algorithm is applied recursively to compute a coarse-grid representation of the error (which is the solution to $A_cU_c = F_c$), with a zero initial guess for U_c. The computed correction, U_c, is interpolated to the fine grid and added to the approximation, $U^{(r)}$, from after the initial relaxation sweep, and a second relaxation sweep is performed.

For one-dimensional problems, we use "red-black" Gauss-Seidel relaxation, where, in Step 1 of Algorithm 1, the odd-indexed nodes are first processed (relative to 0-indexing in C), followed by the even-indexed nodes. The opposite ordering is used in Step 5 to ensure symmetry. To create the hierarchy of meshes, we begin with a chosen fine mesh with N intervals (uniformly spaced or fitted), and create the first coarse mesh with $N/2$ intervals by aggregating intervals pairwise (or, equivalently, by discarding every other node in the fine mesh). This process is repeated recursively until a grid with fewer than 5 nodes is reached. We assume the initial N is chosen so that this pairwise aggregation never fails. On each mesh, the matrix A is created by symmetrising the unequally spaced finite-difference approximation, as described above. The interpolation operator, P, between two such meshes is defined by linear interpolation to neighbouring nodes, with the interpolation weights suitably adjusted for the unequal spacing of nodes. The restriction operator, R, is chosen to be the transpose of P, leading to a weighted averaging of residuals between neighbouring nodes, emphasizing the closer neighbour on an unequally spaced mesh.

This geometric multigrid approach offers excellent performance and scalability on both uniformly spaced and smoothly varying meshes, yielding errors at the level of discretization error in $\ln(N)$ iterations. However, on meshes with sharp contrasts in grid spacing, such as in the case of Shishkin meshes, the performance of the stationary iteration suffers greatly, as relaxation is slow to reduce certain error modes that change rapidly near the changes in grid spacing, and these modes are slowly corrected by coarse-grid correction. Such modes are, however, provably few in number [52] and, so, accelerating the multigrid convergence by use of the preconditioned conjugate gradient iteration is an effective strategy. In this case, the V-cycle given by Algorithm 1 defines a preconditioning matrix, M, that yields a preconditioned system, $MAU = MF$ where the spectrum of MA has a tight cluster of eigenvalues around unity, with only a few outlying eigenvalues caused by the sharp transitions.

4.1.2. Algebraic Multigrid. An alternate approach to using geometric multigrid as a preconditioner for conjugate gradient is to adjust the interpolation operators to account for the sharp mesh transitions, thereby accurately approximating the associated error modes in the coarse-grid correction phase. A standard approach to developing operator-dependent interpolation schemes is the algebraic multigrid (AMG) method, which supplements the multigrid V-cycle given in Algorithm 1 with a preliminary setup stage in which the coarse meshes and interpolation, restriction, and coarse-grid operators are computed based on the given fine-grid operator.

In general, AMG is a much more flexible algorithm than is needed to develop multigrid approaches for the tridiagonal matrices that arise in one dimension. From the given fine-grid operator, a first coarse mesh is chosen, then interpolation, P, is defined, with restriction fixed as $R = P^T$, the coarse-grid operator is given by the Galerkin triple product $A_c = P^T AP$. This process repeats recursively until a suitably small mesh is reached, where a direct solver can be cheaply applied. At each level, a coarse mesh is chosen based on augmenting a maximal independent subset of the graph of the matrix at that level, with the augmentation chosen to ensure suitably quality of interpolation. For the symmetric tridiagonal matrices considered here, this process generates the same coarse grids as are chosen geometrically, selecting every other node in the mesh, which is equivalent to aggregating the finer mesh pairwise. With this coarsening pattern, interpolation to each fine-mesh node from its two coarse-mesh neighbours is defined by diagonal scaling of the off-diagonal connections in A. Thus, for fine-mesh node i, $(PU_c)_i = (-a_{i,i-1}(U_c)_{i-1} - a_{i,i+1}(U_c)_{i+1})/a_{i,i}$, where entries $a_{i,j}$ are those in row i and column j of A, and all indices are relative to the fine-mesh numbering. With this choice of P and the resulting Galerkin definition of $A_c = P^T AP$, it is straightforward to show that the AMG V-cycle recreates the cyclic reduction direct solver for the tridiagonal system. Thus, AMG for these problems converges in a single iteration, for any stopping tolerance above the floating point limits.

4.1.3. A one-dimensional boundary-layer preconditioning approach. While use of AMG, or similar methods based on cyclic reduction, provides a natural solution algorithm for the tridiagonal matrices arising from the one-dimensional problem, these approaches do not take advantage of the singularly perturbed nature of the problem, and the tridiagonal structure is not preserved for problems in higher dimensions. Thus, in this section, we develop an algorithm that takes advantage of the singularly perturbed nature and allows generalization to higher dimensions. In Section 4.2.4, we develop the two-dimensional analogue and prove its optimality.

When $\delta_N \ll 1$, the fitted meshes for one-dimensional problems described in Section 2.2 condense in the region of the two boundary layers and are uniform in the interior. From the linear algebraic point-of-view, the matrix corresponding to a discretization on such a mesh naturally partitions into three pieces: the left boundary layer, the interior (not including the transition points), and the right boundary layer. This gives a partitioned matrix of the form

$$A = \begin{bmatrix} A_{LL} & A_{LI} & 0 \\ A_{IL} & A_{II} & A_{IR} \\ 0 & A_{RI} & A_{RR} \end{bmatrix}, \tag{4.1}$$

where we use the natural subscripts L, I, and R to denote the three regions. Notably, in the symmetrically scaled system, we have the relations that $A_{LI} = A_{IL}^T$ and $A_{RI} = A_{IR}^T$, and we see that each of these matrices has only one non-zero entry, giving the coefficient that relates the last point in the layer (the transition point) to the first point in the interior. Defining the uniform interior mesh width to be h_I, these entries are all $-\varepsilon^2/h_I$.

Considering A_{II}, we notice that this matrix has diagonal entries given by $2\varepsilon^2/h_I + h_I b(x_j)$, for the row corresponding to node x_j, while the off-diagonal coefficients are $-\varepsilon^2/h_I$. In the singularly perturbed case, when $\delta_h = \big(\varepsilon/(h_I\beta)\big)^2 \ll 1$, the diagonal of A_{II} strongly dominates all other entries in the rows and columns corresponding to interior (I) points. In this case, it is intuitive to approximate these rows and columns by just the dominant diagonal values given by the reaction term, $h_I b(x_j)$. Theorem 4.1 shows that this approximation is accurate, in the spectral sense, when $\delta_h \ll 1$.

THEOREM 4.1. *Let h_I denote the (uniform) interior mesh-spacing in the standard 3-point finite difference discretization of $-\varepsilon^2 u''(x) + b(x)u(x)$ on a boundary-fitted mesh, where $b(x) > \beta^2$ for all $x \in [0,1]$, with symmetrised discretization matrix A. Order the rows and columns of A according to the boundary layer structure in* (4.1)*. Define*

$$A_D = \begin{bmatrix} A_{LL} & 0 & 0 \\ 0 & D_{II} & 0 \\ 0 & 0 & A_{RR} \end{bmatrix},$$

where the partitioning matches that of A in (4.1)*, and the entries of diagonal matrix D_{II} are given by $h_I b(x_j)$ for the row/column corresponding to node x_j. Then*

$$(1 - 2\delta_h)V^T A_D V \le V^T A V \le (1 + 6\delta_h)V^T A_D V,$$

for all vectors V.

Proof. Writing a generic vector $V = \begin{bmatrix} V_L & V_I & V_R \end{bmatrix}^T$, we see that

$$V^T A V = V_L^T A_{LL} V_L + 2V_L^T A_{LI} V_I + V_I^T A_{II} V_I + 2V_R^T A_{RI} V_I + V_R^T A_{RR} V_R,$$

using the symmetry of A. Bounding $V_I^T A_{II} V_I$ is straightforward, since $A_{II} = D_{II} + L_{II}$, where L_{II} is the symmetrised uniform-grid Laplacian matrix, with diagonal entry $2\varepsilon^2/h_I$ and off-diagonal entries $-\varepsilon^2/h_I$. Thus, by Geršgorin's theorem,

$$V_I^T D_{II} V_I \le V_I^T A_{II} V_I \le V_I^T \left(D_{II} + (4\varepsilon^2/h_I)I\right) V_I \le (1 + 4\delta_h) V_I^T D_{II} V_I$$

for any V_I, since $V_I^T V_I \le 1/(\beta^2 h_I) V_I^T D_{II} V_I$.

Bounding the cross terms, $V_L^T A_{LI} V_I$ and $V_R^T A_{RI} V_I$, is slightly more difficult, owing to the need to bound these in terms of $V_L^T A_{LL} V_L$, $V_I^T D_{II} V_I$, and $V_R^T A_{RR} V_R$. Here, we focus on bounds for $V_L^T A_{LI} V_I$; the bounds for $V_R^T A_{RI} V_I$ are derived by the same argument.

To bound $V_L^T A_{LI} V_I$, we write $A_{LL} = D_{LL} + L_{LL}$, where D_{LL} is the diagonal matrix with entry $\bar{h}_j b(x_j)$ in the row/column corresponding to node j, $\bar{h}_j$ is the average of the widths of the two cells adjacent to node j, and L_{LL} is the symmetrised (positive-definite) Laplacian operator in the left boundary layer. Then, by the Cauchy-Schwarz inequality,

$$\left|V_L^T A_{LI} V_I\right| \le \left\| D_{LL}^{-1/2} A_{LI} V_I \right\|_2 \cdot \left\| D_{LL}^{1/2} V_L \right\|_2,$$

for any V_L and V_I. Now, since $V_L^T D_{LL} V_L \le V_L^T A_{LL} V_L$ for all V_L,

$$\left|V_L^T A_{LI} V_I\right| \le \left\| D_{LL}^{-1/2} A_{LI} V_I \right\|_2 \left(V_L^T A_{LL} V_L\right)^{1/2},$$

for any V_L and V_I.

To bound the remaining term, $\left(V_I^T A_{IL} D_{LL}^{-1} A_{LI} V_I\right)^{1/2}$, note that there is only one non-zero entry in each of A_{IL} and A_{LI}, so that there is only one non-zero entry in $A_{IL} D_{LL}^{-1} A_{LI}$, given by $(\varepsilon^2/h_I)^2/\big(\bar{h}_j b(x_j)\big)$, where x_j is the last node in the left boundary layer. Since this node is adjacent to an interior mesh cell with width h_I, $h_I/2 < \bar{h}_j$, and the entry in $A_{IL} D_{LL}^{-1} A_{LI}$ satisfies

$$(\varepsilon^2/h_I)^2/\big(\bar{h}_j b(x_j)\big) \le (\varepsilon^2/h_I)^2 \frac{2}{h_I \beta^2} \le (\varepsilon^2/h_I)^2 \frac{2}{h_I^2 \beta^4} h_I b(x_{j+1}) = 2\delta_h^2 h_I b(x_{j+1}),$$

where x_{j+1} is the first node in the interior region. Thus, $V_I^T A_{IL} D_{LL}^{-1} A_{LI} V_I \leq 2\delta_h^2 V_I^T D_{II} V_I$ for all V_I, giving

$$\left|V_L^T A_{LI} V_I\right| \leq \sqrt{2}\delta_h \left(V_I^T D_{II} V_I\right)^{1/2} \left(V_L^T A_{LL} V_L\right)^{1/2},$$

for all V_L and V_I. Since $2ab \leq a^2 + b^2$ for any real a, b, we have

$$2\left|V_L^T A_{LI} V_I\right| \leq \delta_h V_I^T D_{II} V_I + 2\delta_h V_L^T A_{LL} V_L,$$

for all V_L and V_I. From this, and the corresponding bound that

$$2\left|V_R^T A_{RI} V_I\right| \leq \delta_h V_I^T D_{II} V_I + 2\delta_h V_R^T A_{RR} V_R,$$

for all V_R and V_I, the stated bound follows. □

Note 1: While the lower bound holds for all δ_h, the bound is only useful in the case that $\delta_h < 1/2$, since a lower bound of $0 \leq V^T A V$ naturally holds. When $\delta_h < 1/2$, the theorem establishes spectral equivalence bounds that show that A_D is an excellent preconditioner for A as $\delta_h \to 0$.

Note 2: A slight improvement of the upper bound, from a constant of $1+6\delta_h$ to $1+5\delta_h$ can be found by combining the bounds of the two cross terms. However, since $\delta_h \ll 1$ is the case of interest, this improvement is largely irrelevant.

COROLLARY 4.2. *Under the assumptions of Theorem 4.1, if M_{LL} and M_{RR} are spectrally equivalent to A_{LL} and A_{RR}, respectively, meaning that there are constants c_0 and c_1 such that*

$$c_0 V_L^T M_{LL} V_L \leq V_L^T A_{LL} V_L \leq c_1 V_L^T M_{LL} V_L \text{ for all } V_L$$
$$\text{and } c_0 V_R^T M_{RR} V_R \leq V_R^T A_{RR} V_R \leq c_1 V_R^T M_{RR} V_R \text{ for all } V_R,$$

then the matrix

$$A_M = \begin{bmatrix} M_{LL} & 0 & 0 \\ 0 & D_{II} & 0 \\ 0 & 0 & M_{RR} \end{bmatrix},$$

satisfies

$$\min(1-2\delta_h, c_o(1-2\delta_h)) V^T A_M V \leq V^T A V \leq \max\left(1+6\delta_h, c_1(1+2\delta_h)\right) V^T A_M V$$

for all V.

Note 3: Corollary 4.2 is particularly relevant to the case where M_{LL} and M_{RR} are the inverses of effective multigrid preconditioners for A_{LL} and A_{RR}, respectively. If the stationary multigrid cycle represented by $I - M_{LL}^{-1} A_{LL}$ has spectral radius $\alpha < 1$, then

$$(1-\alpha) V_L^T M_{LL} V_L \leq V_L^T A_{LL} V_L \leq (1+\alpha) V_L^T M_{LL} V_L$$

for all V_L. If the same bounds hold for the right boundary layer, then the bound in Corollary 4.2 becomes

$$(1-\alpha)(1-2\delta_h) V^T A_M V \leq V^T A V \leq \max\left(1+6\delta_h, (1+\alpha)(1+2\delta_h)\right) V^T A_M V$$

for all V. Since we typically expect $\alpha \approx 0.1$ and $\delta_h \ll 1$, a better rule of thumb would be

$$(1-\alpha)(1-2\delta_h) V^T A_M V \leq V^T A V \leq (1+\alpha)(1+2\delta_h) V^T A_M V$$

for all V. This suggests that the convergence of a preconditioner consisting of one multigrid V-cycle applied to each of the boundary layer regions plus diagonal scaling of the interior region should be very similar to that of multigrid applied directly to the boundary layer regions alone. In the case of Shishkin meshes, this means that we expect to recover the optimal scaling of multigrid on uniform meshes, while we expect similar behaviour for Bakhvalov meshes, due to the smooth grid stretching.

4.2. Solving the two-dimensional system. In contrast to the one-dimensional problem, there is the potential for true practical benefit in the development of optimally efficient multigrid-based solvers for two-dimensional problems on tensor-product fitted meshes, to address the complexity issues of direct methods discussed in Section 3. It is well known that simple multigrid approaches are optimal or near-optimal for uniformly elliptic problems on uniform tensor-product meshes. In this section, we discuss the twin challenges of efficiently dealing with tensor products of highly nonuniform fitted meshes and singularly perturbed problems on these meshes.

4.2.1. Geometric Multigrid. The success of simple geometric multigrid with pointwise relaxation as a good preconditioner for the one-dimensional problem, with only mild dependence on N and ε, is not recreated in two dimensions, due to the nature of the tensor product of fitted meshes. As noted in [55, 54], the key parameter in determining the effectiveness of pointwise relaxation in geometric multigrid is the mesh aspect ratio; when the ratio of the largest to smallest edges of a mesh cell is much larger than unity, multigrid convergence suffers. In contrast to the one-dimensional case, however, the space of slow-to-converge modes is much larger in two dimensions, leading to poor behaviour also as a preconditioner for conjugate gradients, as the performance of pointwise relaxation degrades with ε and the number of slow-to-converge modes grows with N. Table 4.1 shows the number of iterations needed for geometric multigrid preconditioned CG to reduce the maximum norm of the error in the iterative solution to that of the direct solution of the linear system. Two well-known antidotes to this degradation are to consider the use of mesh-adapted coarsening approaches or the use of line-wise (coupled) relaxation techniques [8, 49].

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$
1	11	11	12	12
10^{-2}	14	16	18	19
10^{-4}	60	96	140	178
10^{-6}	76	152	300	557
10^{-8}	81	170	348	712
10^{-10}	88	179	364	748
10^{-12}	92	186	387	811

TABLE 4.1

Iteration counts for geometric multigrid preconditioned CG to achieve theoretically optimal error reduction on Bakhvalov meshes.

In the former category, semicoarsening techniques are commonly used when the bad aspects ratios result from the use of tensor-product grids of vastly different meshwidths (as, for example, might be the case for two-dimensional problem with a one-dimensional boundary layer). These techniques can also be adapted to tensor-products of graded or fitted meshes [54], but require much more involved development of the semicoarsening approach. In Section 4.2.2, we discuss the application of algebraic multigrid (AMG) to the two-dimensional problem, which results in coarse grids adapted to the fitted mesh structure, suitable for use with pointwise relaxation.

By contrast, the use of line relaxation techniques can allow the continued use of regular geometric coarsening in both directions, at the cost of a more expensive (but still optimally scaling, due to the optimal scaling in cost of tridiagonal solves) relaxation technique. Because the tensor product of fitted meshes involves cells with bad aspect ratios in two orientations (some with their long edges aligned with the x-axis and some with their long edges aligned with the y-axis), alternating-direction line relaxation is necessary to adequately damp all types of oscillatory error. In order to gain added robustness, we use the Black Box Multigrid method of Dendy [1, 15], coupling alternating-direction line relaxation with variational coarsening, discussed in Section 4.2.3.

The choice to focus on AMG and BoxMG as solution strategies is based on two criteria. First, they both employ variational coarsening strategies based on Galerkin projection. Thus, variable reaction coefficients are naturally projected onto coarse meshes, and the same solvers can be easily applied to other discretizations, such as bilinear finite-element discretizations on fitted meshes (see, e.g, [30] for an error analysis for a two-dimensional reaction diffusion problem on a Shishkin mesh). Secondly, they are both readily available in software packages that can easily be incorporated into simulation codes beyond those used in the test cases presented in Section 5. The downside of using these solvers is that they do not represent tuned software that

takes full advantage of the structure of the differential operators, discretization, and fitted meshes considered here. Thus, small improvements in overall time-to-solution could be realized by more specialized software, but at the expense of applicability and ease of use. In contrast, in Section 4.2.4, we present an algorithm that is tuned specifically for the fitted meshes considered here, but which is still applicable to both finite difference and finite element discretizations of singularly perturbed reaction-diffusion equations.

4.2.2. Algebraic Multigrid. For two-dimensional problems, the coarse-grid selection stage of AMG plays an important role and, notably, does not reduce to a special case as in 1D. With general selections of coarse grids, the definition of interpolation operators is, also, important and non-trivial. We briefly review these aspects here; for a fuller treatment, consult [8, 43, 48]. While these details are important for the AMG setup phase, we note that only a small change is made in the solve phase detailed in Algorithm 1, where the size of the coarse-grid problem is now determined by the number of coarse-grid points selected, changing the notation, but not the substance, of the recursive step. For relaxation, we use an ordered Gauss-Seidel sweep that, in Step 1 of Algorithm 1, loops lexicographically over the points selected for the coarse mesh first, then over those that are not selected for the coarse mesh. In Step 5 of Algorithm 1, the opposite ordering is used, to preserve the symmetry of the preconditioner.

The coarse-grid selection algorithm of [43] is used here. First, a reduced matrix is computed by thresholding the off-diagonal values relative to the largest off-diagonal in each row. In experiments reported on in Section 5, a standard M-matrix based criterion is used, where "strong" connections are defined as those negative off-diagonal entries whose magnitude is at least 0.25 times that of the largest negative off-diagonal entry. From this, a maximal independent subset of the graph of the reduced matrix is computed as the first pass at determining the set of coarse-grid points. In a second pass, this set is augmented, adding a locally minimal number of points to the coarse-grid set to ensure that the AMG interpolation operator, detailed below, is well-defined. This process ensures that the coarse-grid set is much smaller than the original fine mesh, yet is still large enough to ensure good correction is possible to a range of errors that are slow to be reduced by relaxation.

Once the coarse-grid set has been chosen, interpolation is determined based on the principles that the errors that are slow to be reduced by relaxation also yield small residuals, and that these errors can be accurately interpolated from nodes with a strong connection. Interpolation of a correction from a coarse-grid node to its fine-grid counterpart is done directly, with unit weight. For each fine-grid node, i, the non-zero entries in row i of the matrix are sorted into those that are strong connections in the coarse-grid set, C_i, which must be non-empty by the initial choice of a maximal independent subset for the coarse-grid set, strong connections in the fine-grid set, F_i, and weak connections, W_i. Based on this decomposition, and the assumption that an error, E, to be interpolated give small residuals, $(AE)_i \approx 0$, we base interpolation on the decomposition that

$$a_{ii}E_i = -\sum_{k\in C_i} a_{ik}E_k - \sum_{j\in F_i} a_{ij}E_j - \sum_{\ell\in W_i} a_{i\ell}E_\ell.$$

Strong connections in C_i are natural to use in interpolation, but we seek to compensate the weights of direct interpolation, $-a_{ik}/a_{ii}$, to account for connections to other points. Weak connections are assumed to play little role in interpolation and, so, their contributions to this decomposition are lumped onto the diagonal, writing $E_\ell \approx E_i$. Strong connections to F_i are treated by indirect interpolation. For each $j \in F_i$, we approximate $E_j \approx \sum_{k\in C_i} w_{jk}E_k$, where the weights $w_{jk} = a_{jk}/\sum_{m\in C_i} a_{jm}$ are chosen to approximate E_j by an average of the values in C_i, weighted towards those points in C_i to which j is strongly connected. The second pass of the coarsening scheme ensures that $\sum_{m\in C_i} a_{jm} \neq 0$ for all $j \in F_i$, by adding points to the coarse-grid set for any i, j pair for which this doesn't already hold. With these choices, the interpolation weights in P are given by

$$(PU_c)_i = \sum_{k\in C_i} \frac{-a_{ik} - \sum_{j\in F_i}\left(\frac{a_{ij}a_{jk}}{\sum_{m\in C_i} a_{jm}}\right)}{a_{ii} + \sum_{\ell\in W_i} a_{i\ell}}(U_c)_k,$$

for a general coarse-grid correction U_c. With these weights, we use the variational definitions, giving $R = P^T$, and $A_c = P^T AP$, to recursively define all levels of the multigrid hierarchy.

4.2.3. Black Box Multigrid. In contrast to AMG, the Black Box Multigrid (BoxMG) algorithm [1, 15] focuses on maintaining the regular, tensor-product grid structure of the fine mesh, and choosing more robust relaxation techniques, in combination with a similar operator-dependent definition of interpolation, to achieve robustness in the multigrid algorithm. The algorithmic choice to maintain structured grids on all levels of the multigrid hierarchy avoids the expensive graph processing and indirect addressing of AMG; as a result, BoxMG typically achieves much faster total times to solution for problems where both AMG and BoxMG can be readily applied, about six times faster for some two-dimensional problems [10], and ten to fifteen times faster for some three-dimensional problems [32].

The key to maintaining robustness in BoxMG for two-dimensional problems is the use of alternating-direction line relaxation. The rectangular grid is decomposed twice, into alternating lines in both the x- and y-directions. In a single sweep of relaxation (Step 1 of Algorithm 1), residuals are calculated first for alternating lines parallel to the x-axis, starting from the second row of the grid, and an update is calculated for each of these lines to simultaneously zero the residual at all points along these lines, then the same procedure is applied to alternating lines parallel to the x-axis, starting from the first row of the grid, then these two stages are repeated for alternating lines parallel to the y-axis. The opposite ordering is used in Step 5 of Algorithm 1 to ensure symmetry.

The coarse-grid correction phase of BoxMG is simpler than that of AMG, due to the choice of regularly structured coarse grids, but still uses an operator-induced interpolation scheme and variational coarsening. Because of the regular structures, a simple symbolic calculation shows that the non-zero pattern of the matrix on all grids is confined to the usual nine-point stencil, so long as the fine-grid matrix is, including when the fine-grid matrix has the usual five-point stencil. For any point that is on the coarse mesh, its value is interpolated to the fine mesh with unit weight. For fine-grid points along the lines in which the coarse grid is embedded, the fine-grid stencil is collapsed orthogonal to this line to yield a one-dimensional three-point stencil that yields interpolation as in the one-dimensional case of AMG. For fine-grid points that are in the interior of coarse-grid cells, the assumption of interpolation of a zero residual is again used; however, since interpolation coefficients have already been defined for all neighbouring grid-points, these weights are used directly to define interpolation to the interior points. The full specification of BoxMG interpolation is reviewed in [31], including a discussion of its relationship to AMG interpolation.

4.2.4. A two-dimensional boundary-layer preconditioning approach. For the two-dimensional problem with $\delta_N \ll 1$, we extend the preconditioning technique from Section 4.1.3 to handle tensor-products of fitted meshes, by partitioning the 2D mesh into four pieces: the high-resolution corner, two edges with resolved meshes in one dimension (along the x- and y-axes), and the interior region. In the corner, connections in both coordinate directions are significant, requiring the full power of a multigrid method. In the edges, the problems are effectively one-dimensional, and tridiagonal solvers can be used within an effective preconditioner. In the interior, as in one dimension, diagonal scaling is effective. Thus, the two-dimensional boundary-layer preconditioner can be expressed by partitioning the degrees of freedom in U and, consequently, the rows and columns of A as

$$A = \begin{bmatrix} A_{CC} & A_{CE} & 0 \\ A_{EC} & A_{EE} & A_{EI} \\ 0 & A_{IE} & A_{II} \end{bmatrix}, \tag{4.2}$$

where the subscripts indicate the block structure of corners, C, edge layers, E, and interior points, I. The preconditioner, A_D, is defined in the same partitioning by

$$A_D = \begin{bmatrix} A_{CC} & 0 & 0 \\ 0 & T_{EE} & 0 \\ 0 & 0 & D_{II} \end{bmatrix}, \tag{4.3}$$

where the matrix D_{II} has diagonal entries given by the scaled reaction coefficient, and the matrix T_{EE} has three nonzero entries per row: in the diagonal position, and in the two off-diagonals corresponding to neighbours along the direction with the smallest meshwidth. Thus, T_{EE} could be itself partitioned into two parts, one corresponding to the edge layer along the x-axis, and one corresponding to the edge layer along

the y-axis. Depending on the ordering of the degrees of freedom (in x then y, or in y then x), one of these blocks will be tridiagonal, and one will be block-tridiagonal with diagonal blocks. The neglected entries in A_{II} and A_{IE} or A_{EI} are essentially the same as those treated in the one-dimensional case, but scaled by another factor of h_I from the symmetrisation, giving $-\varepsilon$, relative to the diagonal values $h_I^2 b(x_i, y_j)$. Thus, a similar argument will be used to bound their contribution. The neglected entries in A_{CC}, A_{CE}, and A_{EC} are slightly more complicated, due to the potentially uneven mesh spacing in the boundary layers, but a similar argument bounds their contribution.

THEOREM 4.3. *Let h_I denote the interior mesh-spacing in the standard 5-point finite-difference discretization of $-\varepsilon^2\Delta u(x,y)+b(x,y)u(x,y)$ on a boundary-fitted mesh, where $b(x,y) > \beta^2$ for all $(x,y) \in [0,1]^2$ and boundary conditions such that there are only two boundary layers, along the x- and y-axes (the South and West faces, respectively), with symmetrised discretization matrix A. Order the rows and columns of A according to the boundary layer structure in* (4.2) *and define A_D in the same ordering as in* (4.3)*, where the entries in the diagonal matrix D_{II} are given by the reaction coefficients, $h_I^2 b(x_i, y_j)$, for the row/column corresponding to node (x_i, y_j) and the off-diagonal coefficients of $A_{EE} - T_{EE}$ are of the form $-\varepsilon^2 \bar{k}_j/h_I$ (for edges along the x-axis) or $-\varepsilon^2 \bar{h}_j/h_I$ (for edges along the y-axis), with twice the magnitude of these values on the diagonal (so the $A_{EE} - T_{EE}$ is a zero row-sum operator except for the first and last rows). Then,*

$$(1 - 3\delta_h)V^T A_D V \leq V^T A V \leq (1 + 9\delta_h)V^T A_D V,$$

for all vectors V.

Proof. Writing generic vector $V = \begin{bmatrix} V_C & V_E & V_I \end{bmatrix}$ (noting that here V_C is the component of V associated with the refined corner of the mesh, and should not be confused with the notation of Sections 4.1.1 and 4.1.2 where V_c would denote the coarse-grid analogue of V) we see that

$$V^T A V = V_C^T A_{CC} V_C + 2V_C^T A_{CE} V_E + V_E^T A_{EE} V_E + 2V_E^T A_{EI} V_I + V_I^T A_{II} V_I,$$

using the symmetry of A. We seek to bound this above and below by a scalar (depending only on δ_h) times the quantity

$$V^T A_D V = V_C^T A_{CC} V_C + V_E^T T_{EE} V_E + V_I^T D_{II} V_I.$$

Bounding $V_I^T A_{II} V_I$ is straight-forward, since $A_{II} = D_{II} + L_{II}$ where L_{II} is the symmetrised uniform-grid Laplacian matrix with diagonal entry $4\varepsilon^2$ and off-diagonal entries $-\varepsilon^2$. Thus, by Geršgorin's theorem,

$$V_I^T D_{II} V_I \leq V_I^T A_{II} V_I \leq V_I^T \left(D_{II} + (8\varepsilon^2) I\right) V_I \leq (1 + 8\delta_h) V_I^T D_{II} V_I$$

for any V_I, since $V_I^T V_I \leq 1/(\beta^2 h_I^2) V_I^T D_{II} V_I$.

Considering $V_E^T A_{EE} V_E$ is slightly more complicated because of the tridiagonal structure of $A_{EE} - T_{EE}$; however, we can consider the spectral equivalence by analyzing a generic line in the edges of the mesh. To do this, first note that A_{EE} is reducible into two blocks, one containing connections within the layer along the x-axis, and one containing connections within the layer along the y-axis. Considering one of these pieces, along the x-axis, we decompose the stencil in (2.4) into two pieces, writing

$$\begin{pmatrix} & -\frac{h_I}{k_{j+1}}\varepsilon^2 & \\ -\frac{\bar{k}_j}{h_I}\varepsilon^2 & \left(\frac{2\bar{k}_j}{h_I} + h_I\left(\frac{1}{k_j} + \frac{1}{k_{j+1}}\right)\right)\varepsilon^2 + \bar{k}_j h_I b(i,j) & -\frac{\bar{k}_j}{h_I}\varepsilon^2 \\ & -\frac{h_I}{k_j}\varepsilon^2 & \end{pmatrix}$$

$$= \begin{pmatrix} & -\frac{h_I}{k_{j+1}}\varepsilon^2 & \\ 0 & h_I\left(\frac{1}{k_j} + \frac{1}{k_{j+1}}\right)\varepsilon^2 + \bar{k}_j h_I b(i,j) & 0 \\ & -\frac{h_I}{k_j}\varepsilon^2 & \end{pmatrix} + \begin{pmatrix} & 0 & \\ -\frac{\bar{k}_j}{h_I}\varepsilon^2 & \frac{2\bar{k}_j}{h_I}\varepsilon^2 & -\frac{\bar{k}_j}{h_I}\varepsilon^2 \\ & 0 & \end{pmatrix},$$

where the first term on the right-hand side represents the terms kept in T_{EE}, and the second term represents $A_{EE} - T_{EE}$. From this decomposition, we immediately see that $V_E^T T_{EE} V_E \leq V_E^T A_{EE} V_E$ for any V_E, since the remainder term is positive definite. To find the upper bound, note that the entries in $A_{EE} - T_{EE}$ depend only on the position in the y-direction, through $\bar{k}_j$, and not on the position in the x-direction. Thus, defining the diagonal matrix M, with dimension given by the number of points within the boundary layer and entries given by $(\bar{k}_j/h_I)\varepsilon^2$, for the row/column associated with the point j nodes in from the boundary, we can write

$$(A_{EE} - T_{EE})_x = \begin{bmatrix} 2M & -M & & & & & \\ -M & 2M & -M & & & & \\ & -M & 2M & -M & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -M & 2M & -M & \\ & & & & -M & 2M & \end{bmatrix},$$

assuming an ordering of points first in lines parallel to the y-axis, then in lines parallel to the x-axis, where the subscript x is used on $A_{EE} - T_{EE}$ to indicate that we consider only the edge layer along the x-axis. The matrix $(A_{EE} - T_{EE})_x$ has block dimension equal to the number of points on the grid that are *not* in the boundary layer, and the quantity $(V_E)_x^T (A_{EE} - T_{EE})_x (V_E)_x$ can easily be expressed by writing $(V_E)_x = \begin{bmatrix} V_1 & V_2 & \dots \end{bmatrix}$, giving

$$(V_E)_x^T (A_{EE} - T_{EE})_x (V_E)_x = 2 \sum_\ell V_\ell^T M V_\ell - 2 \sum_\ell V_\ell^T M V_{\ell+1},$$

where the first sum extends over all ℓ, and the second over all but the last ℓ. Bounding the terms in the second sum, using Cauchy-Schwarz, gives

$$|V_\ell^T M V_{\ell+1}| \leq \left\| M^{1/2} V_\ell \right\|_2 \left\| M^{1/2} V_{\ell+1} \right\|_2 \leq \frac{1}{2} \left(V_\ell^T M V_\ell + V_{\ell+1}^T M V_{\ell+1} \right);$$

this, in turn, bounds

$$\left| (V_E)_x^T (A_{EE} - T_{EE})_x (V_E)_x \right| \leq 2 \sum_\ell V_\ell^T M V_\ell + 2 \sum_\ell |V_\ell^T M V_{\ell+1}| \leq 4 \sum_\ell V_\ell^T M V_\ell.$$

For one of these terms, however, the diagonal entries in M, $(\bar{k}_j)/h_I \varepsilon^2$ are bounded above by $\delta_h \bar{k}_j h_I b(x_i, y_j)$ for any i. Thus,

$$V_\ell^T M V_\ell \leq \delta_h V_\ell^T D_\ell V_\ell$$

for all V_ℓ for any ℓ, where the diagonal matrices D_ℓ are taken to have entries $\bar{k}_j h_I b(x_{i_\ell}, y_j)$ where index i_ℓ corresponds to the line indexed by ℓ. Taken together, with the analogous bounds for the edge layer along the y-axis, this gives

$$V_E^T (A_{EE} - T_{EE}) V_E \leq 4 \delta_h V_E^T D_{EE} V_E,$$

for any V_E, where D_{EE} is the diagonal matrix with entries $\bar{k}_j h_I b(x_i, y_j)$ for nodes (i, j) in the edge layer along the x-axis and analogous values for nodes in the edge layer along the y-axis. Since $V_E^T D_{EE} V_E \leq V_E^T T_{EE} V_E$ for any V_E, which follows from the fact that the difference $T_{EE} - D_{EE}$ is positive definite, this gives

$$V_E^T T_{EE} V_E \leq V_E^T A_{EE} V_E \leq V_E^T T_{EE} V_E + 4 \delta_h V_E^T D_{EE} V_E \leq (1 + 4\delta_h) V_E^T T_{EE} V_E.$$

We next look to bound the mixed terms, $2 V_C^T A_{CE} V_E$ and $2 V_E^T A_{EI} V_I$. Considering the first of these, we again employ the Cauchy-Schwarz inequality, writing

$$\left| V_C^T A_{CE} V_E \right| \leq \left\| D_{CC}^{-1/2} A_{CE} V_E \right\|_2 \cdot \left\| D_{CC}^{1/2} V_C \right\|_2,$$

for any V_C and V_E, where D_{CC} is the diagonal matrix with entry $\bar{h}_i \bar{k}_j b(x_i, y_j)$ in the row/column corresponding to node (i, j). Since $A_{CC} = D_{CC} + L_{CC}$ for positive-definite Laplacian matrix L_{CC}, $V_C^T D_{CC} V_C \leq V_C^T A_{CC} V_C$, and

$$\left| V_C^T A_{CE} V_E \right| \leq \left\| D_{CC}^{-1/2} A_{CE} V_E \right\|_2 \cdot \left(V_C^T A_{CC} V_C \right)^{1/2}.$$

To bound the remaining term, notice that there is at most one non-zero entry per row in A_{CE} and, consequently, in $D_{CC}^{1/2}A_{CE}$ and $A_{EC}D_{CC}^{-1}A_{CE}$. The nonzero entries in $A_{EC}D_{CC}^{-1}A_{CE}$ occur on the diagonals of the rows/columns corresponding to nodes that are adjacent to the highly resolved corner, and are given by $\left((\bar{k}_j/h_I)\varepsilon^2\right)^2/(\bar{h}_i\bar{k}_j b(x_i,y_j)$ for a node $(i+1,j)$ in the edge layer along the x-axis (so that node (i,j) is in the corner region), with a similar expression for a node in the edge layer along the y-axis. Since $x_{i+1}-x_i=h_I$, $\bar{h}_i\geq h_I/2$, giving

$$\left((\bar{k}_j/h_I)\varepsilon^2\right)^2/(\bar{h}_i\bar{k}_j b(x_i,y_j)\leq(\varepsilon^2/h_I)^2\frac{2}{h_I\beta^2}\bar{k}_j\leq(\varepsilon^2/h_I)^2\frac{2}{h_I^2\beta^4}h_I\bar{k}_j b(x_{i+1},y_j)=2\delta_h^2 h_I\bar{k}_j b(x_{i+1},y_j),$$

with an analogous bound for nodes in the edge layer along the y-axis. Thus, $V_E^TA_{EC}D_{CC}^{-1}A_{CE}V_E\leq 2\delta_h^2V_E^TD_{EE}V_E$ for all V_E and, thus, $V_E^TA_{EC}D_{CC}^{-1}A_{CE}V_E\leq 2\delta_h^2V_E^TT_{EE}V_E$ for all V_E. This gives

$$\left|V_C^TA_{CE}V_E\right|\leq\sqrt{2}\delta_h\left(V_E^TT_{EE}V_E\right)^{1/2}\left(V_C^TA_{CC}V_C\right)^{1/2},$$

for all V_C and V_E. Since $2ab\leq a^2+b^2$, this gives

$$2\left|V_C^TA_{CE}V_E\right|\leq\delta_hV_E^TT_{EE}V_E+2\delta_hV_C^TA_{CC}V_C,\tag{4.4}$$

for all V_C and V_E.

Finally, we look to bound $2V_E^TA_{EI}V_I$ using similar techniques. The Cauchy-Schwarz bound becomes

$$\left|V_E^TA_{EI}V_I\right|\leq\left\|D_{EE}^{-1/2}A_{EI}V_I\right\|_2\cdot\left\|D_{EE}^{1/2}V_E\right\|_2$$

for all V_E and V_I, giving

$$\left|V_E^TA_{EI}V_I\right|\leq\left\|D_{EE}^{-1/2}A_{EI}V_I\right\|_2\cdot\left(V_E^TT_{EE}V_E\right)^{1/2}$$

for all V_E and V_I, since $V_E^TD_{EE}V_E\leq V_E^TT_{EE}V_E$ for all V_E. Again, there is only one entry per row of $A_{IE}D_{EE}^{-1}A_{EI}$, on the diagonal of rows/columns adjacent to the edge layers. For a node $(i,j+1)$ adjacent to the edge layer along the x-axis, the entry would be

$$\varepsilon^4/(h_I\bar{k}_j b(x_i,y_j))\leq\varepsilon^4\frac{2}{h_I^2\beta^2}\leq 2\delta_h^2h_I^2b(x_i,y_{j+1}),$$

where we use the bound $\bar{k}_j\geq h_I/2$, since $y_{j+1}-y_j=h_I$. This gives

$$\left|V_E^TA_{EI}V_I\right|\leq 2\delta_h^2\left(V_I^TD_{II}V_I\right)^{1/2}\left(V_E^TT_{EE}V_E\right)^{1/2}$$

for all V_E and V_I. Consequently,

$$2\left|V_E^TA_{EI}V_I\right|\leq\delta_hV_I^TD_{II}V_I+2\delta_hV_E^TT_{EE}V_E\tag{4.5}$$

for all V_E and V_I.

Assembling these bounds, we have

$$\begin{aligned}
V^TAV&=V_C^TA_{CC}V_C+2V_C^TA_{CE}V_E+V_E^TA_{EE}V_E+2V_E^TA_{EI}V_I+V_I^TA_{II}V_I\\
&\geq V_C^TA_{CC}V_C-2\left|V_C^TA_{CE}V_E\right|+V_E^TT_{EE}V_E-2\left|V_E^TA_{EI}V_I\right|+V_I^TD_{II}V_I\\
&\geq V_C^TA_{CC}V_C-\left(\delta_hV_E^TT_{EE}V_E+2\delta_hV_C^TA_{CC}V_C\right)+V_E^TT_{EE}V_E\\
&\qquad\qquad-\left(\delta_hV_I^TD_{II}V_I+2\delta_hV_E^TT_{EE}V_E\right)+V_I^TD_{II}V_I\\
&=(1-2\delta_h)V_C^TA_{CC}V_C+(1-3\delta_h)V_E^TT_{EE}V_E+(1-\delta_h)V_I^TD_{II}V_I
\end{aligned}$$

for all V, establishing the lower bound. For the upper bound, we have

$$\begin{aligned}
V^TAV&=V_C^TA_{CC}V_C+2V_C^TA_{CE}V_E+V_E^TA_{EE}V_E+2V_E^TA_{EI}V_I+V_I^TA_{II}V_I\\
&\leq V_C^TA_{CC}V_C+2\left|V_C^TA_{CE}V_E\right|+(1+4\delta_h)V_E^TT_{EE}V_E+2\left|V_E^TA_{EI}V_I\right|+(1+8\delta_h)V_I^TD_{II}V_I\\
&\leq V_C^TA_{CC}V_C+\left(\delta_hV_E^TT_{EE}V_E+2\delta_hV_C^TA_{CC}V_C\right)+(1+4\delta_h)V_E^TT_{EE}V_E\\
&\qquad\qquad+\left(\delta_hV_I^TD_{II}V_I+2\delta_hV_E^TT_{EE}V_E\right)+(1+8\delta_h)V_I^TD_{II}V_I\\
&=(1+2\delta_h)V_C^TA_{CC}V_C+(1+7\delta_h)V_E^TT_{EE}V_E+(1+9\delta_h)V_I^TD_{II}V_I
\end{aligned}$$

for all V. □

Note 4: As in the 1D case, the lower bound holds for all δ_h, but the bound is only useful in the case that $\delta_h < 1/3$, since a lower bound of $0 \leq V^T A V$ naturally holds. When $\delta_h < 1/3$, the theorem establishes spectral equivalence bounds that show that A_D is an excellent preconditioner for A as $\delta_h \to 0$.

Note 5: Slight variations in these bounds are easily derived by varying the division of the constants in Equations (4.4) and (4.5). In particular, the upper bound can be slightly decreased at the expense of the constant in the lower bound and vice-versa. However, since $\delta_h \ll 1$ is the case of interest, these variations are largely irrelevant.

COROLLARY 4.4. *Under the assumptions of Theorem 4.3, if M_{CC} is spectrally equivalent to A_{CC}, meaning that*

$$c_0 V_C^T M_{CC} V_C \leq V_C^T A_{CC} V_C \leq c_1 V_C^T M_{CC} V_C \text{ for all } V_C,$$

then the matrix

$$A_M = \begin{bmatrix} M_{CC} & 0 & 0 \\ 0 & T_{EE} & 0 \\ 0 & 0 & D_{II} \end{bmatrix}$$

satisfies

$$\min(1 - 3\delta_h, c_o(1 - 2\delta_h)) V^T A_M V \leq V^T A V \leq \max(1 + 9\delta_h, c_1(1 + 2\delta_h)) V^T A_M V$$

for all V.

Note 6: Corollary 4.4 is particularly relevant to the case where M_{CC} is the inverse of an effective multigrid preconditioner for A_{CC}. If the stationary multigrid cycle represented by $I - M_{CC}^{-1} A_{CC}$ has spectral radius $\alpha < 1$, then

$$(1 - \alpha) V_C^T M_{CC} V_C \leq V_C^T A_{CC} V_C \leq (1 + \alpha) V_C^T M_{CC} V_C$$

for all V_C. Under this assumption, the bound in Corollary 4.4 becomes

$$\min\big(1 - 3\delta_h, (1 - \alpha)(1 - 2\delta_h)\big) V^T A_M V \leq V^T A V \leq \max\big(1 + 9\delta_h, (1 + \alpha)(1 + 2\delta_h)\big) V^T A_M V$$

for all V. Since we typically expect $\alpha \approx 0.1$ and $\delta_h \ll 1$, a better rule of thumb would be

$$(1 - \alpha)(1 - 2\delta_h) V^T A_M V \leq V^T A V \leq (1 + \alpha)(1 + 2\delta_h) V^T A_M V$$

for all V. This suggests that the convergence of a preconditioner consisting of one multigrid V-cycle applied to the corner region, appropriately ordered tridiagonal solves applied to the edge layers plus diagonal scaling of the interior region should be very similar to that of multigrid applied directly to the boundary layer regions alone. In the case of Shishkin meshes, this means that we expect to recover the optimal scaling of multigrid on uniform meshes, while we expect similar behaviour for Bakhvalov meshes when a robust multigrid approach is used.

In light of Corollary 4.4, in Section 5, we present results for a boundary-layer preconditioner that makes use of a single V-cycle of BoxMG in the corner region, appropriately ordered tridiagonal solves to treat the edge layers, and diagonal scaling in the interior. For a tensor product of fitted meshes with $N/2$ points in each layer (including the transition points), this implies that one-quarter of the grid will be treated with a preconditioner that has the same cost as BoxMG, while the remaining three-quarters will be treated in a much cheaper manner. Estimating the work of a BoxMG V-cycle as four full line relaxations on each level times a factor of four-thirds (to account for the hierarchy of levels of sizes N^2, $N^2/4$, $N^2/16$, etc.), this suggests that on three-quarters of the grid, the boundary-layer preconditioner does three-sixteenths of the work of the multigrid V-cycle. The total work estimate is, then $1/4 + (3/4)(3/16) = 25/64$ times the work of a multigrid V-cycle on the full problem. This is, clearly, an over-estimate, since there is extra savings from the lack of interpolation and restriction, and from the cost of diagonal scaling compared to line solves, but provides a rough estimate for the improvement possible from a preconditioner that is tuned to the full problem structure.

4.3. Stopping Criteria. Given a parameter-robust discretization for a singularly perturbed problem, such as those given by the fitted Shishkin or Bakhvalov meshes, our aim is to find an approximate solution of the discrete system that accurately approximates the true (continuum) solution. It is easy to do this if we "oversolve" the discrete system, bounding the norm of the discrete approximation error, $\|U - \tilde{U}\|$, by some vanishingly small quantity (such as the machine round-off precision) for an approximation, $\tilde{U}$, to the exact solution of the discrete problem, U. This gives the natural bound that

$$\|u - \tilde{U}\| \leq \|u - U\| + \|U - \tilde{U}\| \approx \|u - U\|,$$

whenever $\|U - \tilde{U}\| \ll \|u - U\|$. When considering iterative methods, however, we generate a sequence of increasingly better approximate solutions, denoted $U^{(k)}$, to U, in the sense that $\|U - U^{(k)}\| \leq \|U - U^{(k-1)}\|$, although this norm may not match the natural bound between the continuum solution, u, and its "best" discrete approximation, U. Since there is a non-trivial cost of generating each new iterate, it is most efficient to stop the iteration when the norm of the discrete approximation error, $\|U - U^{(k)}\|$, is of the same order as the discretization error, $\|u - U\|$, giving

$$\|u - U^{(k)}\| \leq \|u - U\| + \|U - U^{(k)}\| \leq C\|u - U\|,$$

for a moderate constant, C.

In the case of finite-difference approximations on Shishkin and Bakhvalov meshes, we know that the discretization error bound takes the form

$$\|u - U\|_\infty \leq g(N),$$

for $g(N) = CN^{-2}\ln^2(N)$ for Shishkin, and $g(N) = CN^{-2}$ for Bakhvalov, where the constants are independent of ε.

Naturally, the discrete approximation error, $E^{(k)} = U - U^{(k)}$, is unknown during the iteration; thus, we must measure this error indirectly. For stationary iterations, the best indicator of this error is the residual, $R^{(k)} = F - AU^{(k)} = AE^{(k)}$, since $AU = F$. The simplest possible bound, then, would be to write $E^{(k)} = A^{-1}R^{(k)}$, giving

$$\|U - U^{(k)}\|_\infty \leq \|A^{-1}\|_\infty \|R^{(k)}\|_\infty.$$

Practical bounds on $\|A^{-1}\|_\infty$ arise from M-matrix theory; Theorem 2.7 in [41] states that

$$\|A^{-1}\|_\infty \leq \frac{\|V\|_\infty}{\min_k (AV)_k},$$

where V is majorising vector with $(AV)_k > 0$ for all k. For the discretization matrices considered here for the two-dimensional problem, the choice of V as the vector of all ones is natural, giving $\min_k(AV)_k$ as the Geršgorin bound on the smallest eigenvalue of A; this is bounded from below by $\beta^2 h_{\min}^2$, where $h_{\min}$ is the minimal mesh spacing on the grid. Numerical experiments show this bound to be sharp for small ε. With this bound, we can then guarantee discrete approximation error comparable to the discretization error by asking for $\|R^{(k)}\|_\infty \leq \beta^2 h_{\min}^2 g(N)$, ensuring $\|U - U^{(k)}\|_\infty < g(N)$. For small ε, however, $h_{\min}$ scales as $h_{\min} \backsim \varepsilon \ln(N)/(N\beta)$ for Shishkin meshes, and as $h_{\min} \backsim \varepsilon/(N\beta)$ for Bakhvalov meshes, requiring $\|R^{(k)}\|_\infty \leq C\varepsilon^2(\ln^3 N)/N^3$ to achieve discretization accuracy on Shishkin meshes, and $\|R^{(k)}\|_\infty \leq C\varepsilon^2/N^3$ to achieve discretization accuracy on Bakhvalov meshes. For small ε and moderate N, these values will be much less than machine precision, implying that it is difficult to compute $\|R^{(k)}\|_\infty$ accurately enough to guarantee convergence.

Instead, we look to make use of the standard stopping criterion for preconditioned conjugate gradients, that bounds the inner product of $R^{(k)}$ with the preconditioned residual, $Z^{(k)} = MR^{(k)}$, where M represents the preconditioning matrix applied. When M is a good preconditioner in the spectral sense that MA (or, more precisely, $A^{1/2}MA^{1/2}$, where $A^{1/2}$ is the principal square root of A) is spectrally equivalent to the identity, this inner product accurately estimates $\|E^{(k)}\|_A^2$ as

$$\left(Z^{(k)}\right)^T R^{(k)} = \left(E^{(k)}\right)^T AMAE^{(k)} \approx \|E^{(k)}\|_A^2,$$

To bound $\|E^{(k)}\|_\infty$ by $\|E^{(k)}\|_A$, we make use of an intermediate bound, trading the max-norm for a discrete ℓ_2-norm, noting that

$$\frac{1}{\sqrt{n}}\|E^{(k)}\|_2 \leq \|E^{(k)}\|_\infty \leq \|E^{(k)}\|_2,$$

for vectors of length n. A reasonable, although not rigorous, estimate is to assume near-equality with the lower bound, rather than the often-pessimistic upper bound, under the assumption that the discrete approximation error is nearly equi-distributed across the mesh, so that the error is not concentrated at relatively few mesh points.

With this assumption, a more tractable bound is available, writing

$$\|E^{(k)}\|_\infty \approx \frac{c}{\sqrt{n}}\|E^{(k)}\|_2 \leq \frac{c}{\sqrt{n}}\|A^{-1/2}\|_2\|E^{(k)}\|_A,$$

where $A^{-1/2}$ is the principal square root of the symmetric and positive-definite matrix, A^{-1}. This bound presents two advantages. First, $\|E^{(k)}\|_A$ is both naturally estimated and minimized by the preconditioned conjugate gradient algorithm and, as such, makes a natural stopping criterion for the iterative approach. Secondly, as we show below, the natural bound on $\|A^{-1/2}\|_2$ needed to determine the stopping criterion is much smaller than that of $\|A^{-1}\|_\infty$ (and $\|A^{-1}\|_2$), yielding a stopping criterion that is also less susceptible to problems with round-off errors.

Since $A^{-1/2}$ is the principle square root of A^{-1}, we have $\|A^{-1/2}\|_2^2 = \|A^{-1}\|_2$, as the eigenvalues of A^{-1} are the squares of those of $A^{-1/2}$. Thus, we can estimate the needed norm by applying Geršgorin's theorem to A to estimate its smallest eigenvalue, deriving the same bound on $\|A^{-1}\|_2$ as was given for $\|A^{-1}\|_\infty$ above. This implies that $\|A^{-1/2}\|_2 \leq 1/(\beta h_{\min})$ for the two-dimensional problems. Thus, we have

$$\|E^{(k)}\|_\infty \approx \frac{c}{\sqrt{n}}\|E^{(k)}\|_2 \leq \frac{c}{N}\|A^{-1/2}\|_2\|E^{(k)}\|_A \leq \frac{c}{N}\frac{1}{\beta h_{\min}}\|E^{(k)}\|_A.$$

Similar bounds can be derived for the one-dimensional problems for both $\|A^{-1}\|_\infty$ and $\|A^{-1/2}\|_A$, with changes that the matrices are of dimension $n \approx N$ instead of N^2 and a scaling of $h_{\min}\beta^2$ for the minimal eigenvalue estimate.

Given a bound of this form, the stopping criterion for $\|E^{(k)}\|_A$ to guarantee $\|E^{(k)}\|_\infty < g(N)$ (up to the assumption relating $\|E^{(k)}\|_\infty$ and $\|E^{(k)}\|_2$) is given by

$$\|E^{(k)}\|_A \leq C\frac{\beta N h_{\min}}{c}g(N);$$

with this, we have

$$\|u - U^{(k)}\|_\infty \lesssim Cg(N),$$

ensuring optimality of the approximate discrete solution, $U^{(k)}$. For example, given the N^{-2} scaling of the discretization error on Bakhvalov meshes, in two dimensions the relevant bound is that if

$$\|E^{(k)}\|_A \leq \beta N^{-1} h_{\min}/c, \tag{4.6}$$

then

$$\|u - U^{(k)}\|_\infty \leq \|u - U\|_\infty + \|U - U^{(k)}\|_\infty \lesssim CN^{-2} + \frac{c}{N}\frac{1}{\beta h_{\min}}\|E^{(k)}\|_A \leq (C+1)N^{-2}.$$

What this analysis doesn't treat is the size of the constants, C and c, or their variation in N and ε. While these constants are uniformly bounded independently of N and ε, numerical experiments in 2D show that when ε is large, stricter convergence bounds are needed to obtain the best possible accuracy (that is, to be almost identical to that of a direct solver). Thus, for the results presented in Section 5.2, we use a graduated stopping tolerance of 10^{-3} times the above bounds when $\varepsilon = 1$, 10^{-2} when $\varepsilon = 10^{-1}$ and 10^{-1} when $\varepsilon = 10^{-2}$. For smaller values of epsilon, we take the constant, $c = 1$, with no further scaling. For one-dimensional problems, the scaling is more uniform, and we use universally a factor of $c = 100$, scaling

the above bounds by 0.01. In all results, we take β as an input parameter, matching what is used for the grid generation, and use a computed value of $h_{\min}$.

Finally, we note that the required bounds on $\|E^{(k)}\|_A$ through this analysis are dependent on both N and ε. For many preconditioners, including standard approaches such as incomplete factorization or sparse approximate inverses, this would imply that the necessary number of iterations, k, would grow like some power of N or ε^{-1}. Here, we instead focus on multigrid methods and boundary-layer preconditioners that lead to reduction of $\|E^{(k)}\|_A$ by factors bounded below unity in each iteration, where these bounds are independent of both N and ε. This implies that, at worst, the number of iterations required for convergence grows like $\ln(N)$ or $-\ln(\varepsilon)$. More detailed analysis generally confirms this expectation, showing that the initial norms, $\|E^{(0)}\|_A$, are reasonably well-behaved, with mild dependence on N and ε. A significant reduction in these norms is achieved already by the first multigrid relaxation step and, consequently, $\|E^{(1)}\|_A$ is much reduced from its initial values, typically by a factor proportional to at least $1/\varepsilon$. Further reduction in these norms is generally seen to be independent of N, leading to the mild increases in iteration counts and computing time reported in Section 5.

5. Numerical Results. All numerical results in this section are computed using code (both C and Fortran) compiled with full optimization, executed on a Beowulf cluster using a single core of a node with an AMD Opteron 2427, 2200 MHz processor with 32Gb of RAM (the same as was used for results in Section 3). The main driver routines are written in C to compute the fitted meshes and assemble the matrices, while the CHOLMOD [11], BoxMG [15], and our own AMG libraries are linked to provide linear solvers. Both BoxMG and AMG provide their own PCG wrappers. The boundary-layer preconditioners in one and two dimensions are implemented (separately) in C within their own PCG wrappers.

We do not report errors for the solutions computed here. Instead, we fix convergence tolerances as discussed above (with a small constant factor for BoxMG, detailed below) such that the error in the iterative solutions generated by geometric multigrid, AMG, BoxMG, and, for suitably small δ_h, the boundary-layer preconditioners, matches that of the direct solver (as reported, for example, in Table 2.3 for the Bakhvalov mesh in 2D) to three significant digits. We present here only results for Bakhvalov meshes, as results for Shishkin meshes are nearly identical.

5.1. One-dimensional problems. We consider the solution of the linear systems arising from the finite difference approximation to the test problem (1.3). Because the computing times for solves on reasonable fitted grids in 1D are so small, we solve the resulting linear systems 500 times in succession and report the aggregate solve times here.

As discussed above, there are two reasons for using geometric multigrid as a preconditioner to solve these problems, given the difficulties for convergence caused by non-uniform grids (particularly in the Shishkin case) and the difficulties of computing ℓ_2 norms of the residuals with enough accuracy in double precision to meet the more stringent stopping tolerances of the stationary iteration. Thus, in Tables 5.1 and 5.2, we show the aggregated solve times and (unaggregated) iteration counts for geometric multigrid preconditioned CG applied to Bakhvalov meshes.

ε^2	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$	$N = 2^{13}$	$N = 2^{14}$
1	0.78	1.61	3.22	7.02	14.17	28.72
10^{-2}	0.82	1.63	3.40	6.95	13.95	30.84
10^{-4}	0.86	1.80	3.51	6.97	15.25	30.87
10^{-6}	0.94	1.81	3.50	7.77	15.32	33.60
10^{-8}	0.89	1.74	3.84	7.69	15.41	33.79
10^{-10}	0.96	1.74	3.87	7.74	16.86	33.72
10^{-12}	0.83	1.70	3.86	7.71	16.81	33.99

TABLE 5.1

CPU times for geometric multigrid preconditioned CG to achieve theoretically optimal error reduction on 1D Bakhvalov meshes, aggregated over 500 runs.

The solve times in Table 5.1 show the near-linear scaling in N expected of multigrid, as well as very slow growth with ε. Both of these observations are also demonstrated in the iteration counts in Table 5.2, showing slight growth in the iteration counts as ε decreases and as N increases. Both of these are consistent with multigrid achieving a fixed reduction in the norm of the preconditioned residual per iteration, given the

ε^2	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$	$N = 2^{13}$	$N = 2^{14}$
1	5	6	6	7	7	7
10^{-2}	6	6	7	7	7	8
10^{-4}	6	7	7	7	8	8
10^{-6}	7	7	7	8	8	9
10^{-8}	7	7	8	8	8	9
10^{-10}	7	7	8	8	9	9
10^{-12}	7	7	8	8	9	9

TABLE 5.2

Iteration counts for geometric multigrid preconditioned CG to achieve theoretically optimal error reduction on 1D Bakhvalov meshes.

behaviour of the stopping tolerance with N and ε. Overall, we note that the solve times in Table 5.1 are about a factor of 3 slower than the direct solves on the finest grids, as expected due to the optimal scaling of the Cholesky factorization for these problems.

Nonetheless, it is interesting to compare the speedup realized by the boundary-layer preconditioner with the times shown in Table 5.1. Table 5.3 shows the values of δ_h realized on the 1D Bakhvalov meshes; following Theorem 4.1 and Corollary 4.2, we present results for the cases where $\delta_h < 0.1$, with aggregate CPU times in Table 5.4 and iteration counts in Table 5.5.

ε^2	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$	$N = 2^{13}$	$N = 2^{14}$
1	2.65×10^5	1.06×10^6	4.24×10^6	1.70×10^7	6.78×10^7	2.71×10^8
10^{-2}	2.65×10^3	1.06×10^4	4.24×10^4	1.70×10^5	6.78×10^5	2.71×10^6
10^{-4}	1.21×10^1	4.83×10^1	1.93×10^2	7.73×10^2	3.09×10^3	1.24×10^4
10^{-6}	7.28×10^{-2}	2.91×10^{-1}	1.17×10^0	4.66×10^0	1.86×10^1	7.46×10^1
10^{-8}	6.71×10^{-4}	2.68×10^{-3}	1.07×10^{-2}	4.29×10^{-2}	1.72×10^{-1}	6.87×10^{-1}
10^{-10}	6.63×10^{-6}	2.65×10^{-5}	1.06×10^{-4}	4.24×10^{-4}	1.70×10^{-3}	6.79×10^{-3}
10^{-12}	6.62×10^{-8}	2.65×10^{-7}	1.06×10^{-6}	4.24×10^{-6}	1.70×10^{-5}	6.78×10^{-5}

TABLE 5.3

Computed values of $\delta_h = \varepsilon^2/(h_1^2\beta^2)$ for Bakhvalov meshes on $[0,1]$ with two boundary layers and $\beta = 0.99$.

ε^2	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$	$N = 2^{13}$	$N = 2^{14}$
10^{-6}	0.41					
10^{-8}	0.30	0.74	1.62	3.61		
10^{-10}	0.25	0.63	1.64	3.23	7.36	14.78
10^{-12}	0.28	0.72	1.64	3.17	7.40	14.87

TABLE 5.4

CPU times for boundary layer preconditioned CG to achieve theoretically optimal error reduction on 1D Bakhvalov meshes, aggregated over 500 runs.

Comparing times between Tables 5.1 and 5.4, we see uniform improvements by factors of 2 or slightly better for the boundary-layer preconditioning approach over standard geometric multigrid preconditioning. This is, in general, consistent with our cost expectation, since the boundary layer approach is applying a regular geometric multigrid cycle (adapted only to account for the implied boundary condition at the transition points) to one-half of the grid, and a much cheaper diagonal scaling operation to the other half. Comparing iteration counts between Tables 5.2 and 5.5, we see that they are essentially identical, with the boundary-layer preconditioner occasionally taking one more iteration to fulfill the stopping criterion when δ_h is large.

5.2. Two-dimensional problems. Recall the test problem (1.5). In contrast to the one-dimensional case, it is interesting to first compare the performance of AMG (as a preconditioner for CG) with the times for the direct solvers reported in Table 3.1. Table 5.6 reports the times for AMG solution of the resulting linear systems, averaged over 3 runs. Comparing these two tables, we see that the AMG solution times are

ε^2	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$	$N = 2^{13}$	$N = 2^{14}$
10^{-6}	8					
10^{-8}	7	7	8	9		
10^{-10}	7	7	8	8	9	9
10^{-12}	7	7	8	8	9	9

TABLE 5.5

Iteration counts for boundary layer preconditioned CG to achieve theoretically optimal error reduction on 1D Bakhvalov meshes.

generally greater than the best times for CHOLMOD until $N = 1024$ when the two algorithms are roughly comparable. For larger N, however, the near-optimal scaling of AMG coupled with the clearly sub-optimal scaling of CHOLMOD results in much better performance when comparing AMG to the best-case solve times of CHOLMOD, and substantial improvements when CHOLMOD is slowed by sub-normal arithmetic, as discussed in Section 3.

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	0.26	1.09	4.90	21.12	95.30	429.53
10^{-2}	0.23	1.01	4.57	19.90	91.84	371.54
10^{-4}	0.19	0.90	3.92	17.18	74.40	327.46
10^{-6}	0.17	0.84	3.86	16.67	75.13	323.07
10^{-8}	0.18	0.81	3.74	16.59	72.45	319.21
10^{-10}	0.18	0.95	4.35	20.16	86.49	396.19
10^{-12}	0.20	0.95	4.32	20.05	86.72	397.23

TABLE 5.6

CPU times for AMG preconditioned CG to achieve theoretically optimal error reduction on Bakhvalov meshes, averaged over 3 runs.

Hidden in the AMG iteration times is some noteworthy variation in the number of PCG iterations needed for each problem. Table 5.7 reports iteration counts for AMG preconditioned CG on Bakhvalov meshes; since the iteration counts are the same for each run (as a zero initial guess and fixed right-hand side are used for each value of N and ε), no averaging is needed in these results. Here, we notice that for fixed ε, there is a slight increase in the number of iterations needed to achieve the stopping tolerance with N, roughly proportional to $\log_2 N$. This is naturally predicted by the assumption of a fixed error reduction per iteration of AMG preconditioned CG, given that the stopping criteria decrease with N. Scaling with ε is less clear, where we see an initial decrease in iteration counts as ε decreases, followed by an increase as ε reaches 10^{-5} and 10^{-6}. For fitted meshes, decreases in ε match increases in the worst-case mesh aspect ratio, which are typically correlated with convergence of AMG. This variation in ε can, to some extents, be ameliorated by varying the value of the AMG parameter defining the relative magnitude of strong connections in each row. For worse mesh-aspect ratios, larger values of the parameter (up to 1.0, where only the largest magnitude connections are taken to be strong) are needed to stabilize AMG convergence, while only a fixed value (0.25) has been used here.

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	8	7	8	9	11	13
10^{-2}	6	6	7	8	10	10
10^{-4}	5	6	6	7	8	9
10^{-6}	4	5	6	7	8	9
10^{-8}	5	5	6	7	8	9
10^{-10}	5	7	8	10	11	13
10^{-12}	6	7	8	10	11	13

TABLE 5.7

Iteration counts for AMG preconditioned CG to achieve theoretically optimal error reduction on Bakhvalov meshes.

As seen in other cases, BoxMG clearly outperforms AMG on these problems, with averaged solution times

reported in Table 5.8. Comparing with Table 5.6, we see that CPU times range from about 6 times faster for small grids to 3-4 times faster for large grids. Iteration counts, shown in Table 5.9, again show $\log_2 N$ scaling as problem size increases, but notably less variation in ε. This is likely due to the effectiveness of the alternating-direction line relaxation used within BoxMG, although there is no existing theoretical verification of this scaling. Once again, the times are notably shorter (and better scaling) than those for the direct solver in Table 3.1, as well as showing no sensitivity to subnormal numbers (as one would expect, since a direct solver is only applied on the coarsest of grids). We note that, to achieve this performance, we use a slightly stricter stopping tolerance for BoxMG than we do for AMG or BLPCG, decreasing those described above by a factor of 10. Such small variations are expected, as the component bounds employed are true to within constant factors that have been neglected.

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	0.03	0.15	0.80	4.61	23.06	111.95
10^{-2}	0.03	0.15	0.81	5.02	23.06	121.13
10^{-4}	0.03	0.13	0.75	4.61	24.94	121.11
10^{-6}	0.01	0.11	0.66	4.20	21.20	102.92
10^{-8}	0.02	0.12	0.66	4.20	19.34	102.96
10^{-10}	0.02	0.12	0.67	4.22	19.37	103.06
10^{-12}	0.03	0.12	0.68	4.22	19.41	103.32

TABLE 5.8

CPU times for BoxMG preconditioned CG to achieve theoretically optimal error reduction on Bakhvalov meshes, averaged over 3 runs.

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	8	9	10	10	11	11
10^{-2}	8	9	10	11	11	12
10^{-4}	7	8	9	10	12	12
10^{-6}	6	7	8	9	10	10
10^{-8}	6	7	8	9	9	10
10^{-10}	6	7	8	9	9	10
10^{-12}	6	7	8	9	9	10

TABLE 5.9

Iteration counts for BoxMG preconditioned CG to achieve theoretically optimal error reduction on Bakhvalov meshes.

The boundary-layer preconditioner discussed in Section 4.2.4 is expected to yield even faster times, but only when $\delta_h \ll 1$. Table 5.10 shows the values of δ_h for various choices of N and ε; as expected, for fixed ε, δ_h increases with increasing N (since h_I decreases) while, for fixed N, δ_h decreases with decreasing ε. Note, however, that the decrease is not linear in ε^2, as h_I is dependent on ε.

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
1	1.65×10^4	6.62×10^4	2.65×10^5	1.06×10^6	4.24×10^6	1.69×10^7
10^{-2}	1.24×10^2	4.99×10^2	2.00×10^3	7.98×10^3	3.19×10^4	1.28×10^5
10^{-4}	5.12×10^{-1}	2.05×10^0	8.19×10^0	3.27×10^1	1.31×10^2	5.24×10^2
10^{-6}	4.27×10^{-3}	1.71×10^{-2}	6.83×10^{-2}	2.73×10^{-1}	1.09×10^0	4.37×10^0
10^{-8}	4.15×10^{-5}	1.66×10^{-4}	6.65×10^{-4}	2.66×10^{-3}	1.06×10^{-2}	4.25×10^{-2}
10^{-10}	4.14×10^{-7}	1.66×10^{-6}	6.62×10^{-6}	2.65×10^{-5}	1.06×10^{-4}	4.24×10^{-4}
10^{-12}	4.14×10^{-9}	1.66×10^{-8}	6.62×10^{-8}	2.65×10^{-7}	1.06×10^{-6}	4.24×10^{-6}

TABLE 5.10

Computed values of $\delta_h = \varepsilon^2/(h_I^2\beta)$ for Bakhvalov meshes on $[0,1]^2$ with a single boundary layer and $\beta = 0.99$.

In Tables 5.11 and 5.12, we only report results for cases where $\delta_h < 0.1$, consistent with the spectral equivalence bounds in Theorem 4.3 and Corollary 4.4. We note that, when $\delta_h < 0.01$, there is no notable difference between the errors in the iterative solutions generated by BLPCG with the stopping criterion as

given above from those generated by the direct solver. For $0.01 \leq \delta_h < 0.1$, the errors generated by BLPCG are somewhat larger. For $N = 512$ and $\varepsilon^2 = 10^{-6}$, where $\delta_h = 0.0683$, the relative increase in the BLPCG error is just 0.1%; for $N = 2048$ and $\varepsilon^2 = 10^{-8}$, where $\delta_h = 0.0425$, the relative increase in the BLPCG error is a more significant 15.4%. These values could be decreased by a tighter stopping tolerance, but this would lead to oversolving in the cases where δ_h is smaller.

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
10^{-6}	0.01	0.05	0.27			
10^{-8}	0.01	0.05	0.25	1.31	6.82	29.34
10^{-10}	0.01	0.04	0.24	1.31	6.82	29.35
10^{-12}	0.01	0.05	0.24	1.31	6.82	29.29

TABLE 5.11

*CPU times for preconditioned CG with the boundary-layer preconditioner to achieve (near) theoretically optimal error reduction on Bakhvalov meshes, averaged over 3 runs. Times marked as * indicate those below timing threshold.*

ε^2	$N = 2^7$	$N = 2^8$	$N = 2^9$	$N = 2^{10}$	$N = 2^{11}$	$N = 2^{12}$
10^{-6}	6	6	8			
10^{-8}	6	6	7	8	8	8
10^{-10}	5	6	7	8	8	8
10^{-12}	5	6	7	8	8	8

TABLE 5.12

Iteration counts for preconditioned CG with the boundary-layer preconditioner to achieve (near) theoretically optimal error reduction on Bakhvalov meshes.

Comparing Table 5.11 with Tables 5.6 and 5.8, we see that, when the BLPCG approach is effective, it offers a substantial further reduction in the time-to-solution on large grids, by slightly more than a factor of 3 over BoxMG, and a factor of 10 or more over AMG. This is slightly better than predicted by the rough cost estimate given in Section 4.2.4, but not surprisingly so, especially as BLPCG typically converges in fewer iterations than BoxMG. The iteration counts in Table 5.12 are steady and show only small dependency on ε.

6. Extensions to three-dimensional problems. While we do not directly treat either the theory or practice of three-dimensional problems in this paper, we note that the techniques proposed here for two dimensions all naturally extend to three. Both AMG and BoxMG can be directly applied to the seven-point stencils of finite-difference discretizations in 3D, and past comparisons show BoxMG to be about ten to fifteen times faster than AMG for some diffusion problems [32]. In three dimensions, BoxMG makes use of plane-based relaxation where, because of the non-optimal scaling of direct solvers for two-dimensional problems, relaxation is composed of a single two-dimensional V-cycle along each plane of grid points in each direction of the mesh.

Similarly, a three-dimensional analogue of the boundary-layer preconditioner is also natural. Considering a problem with one boundary layer in each direction and a tensor-product of meshes with $N/2$ points in the boundary layer and $N/2$ evenly spaced points in the interior, there is one highly resolved corner with $N^3/8$ points, to which we would apply a standard 3D BoxMG V-cycle, with cost roughly 1/8 of that of the V-cycle on all N^3 points. Along each of the x-, y-, and z-axes, away from the corner, there are layers that are highly resolved in two directions, but not the third. For example, along the x-axis away from the origin, there is high resolution in the y- and z-directions, but not in the x-direction. Each of these regions has $N^3/8$ points in it, and can be effectively treated by a single plane solve, instead of the six plane solves (in three alternating directions, both before and after coarse-grid correction) used in the standard BoxMG Vcycle. Thus, the cost of treating these regions at each iteration is $(3/8)(1/6)(7/8)$, where the factor of 7/8 comes from the treatment of only the finest grid, and not grids of N^3, $N^3/8$, $N^3/64$, etc., as in the multigrid V-cycle. Next, along each face in the xy-, xz-, and yz-planes, there is a layer that is highly resolved in one direction (orthogonal to the plane), but not along the plane. In these regions, line solves are sufficient. In the remaining $N^3/8$ points in the interior, diagonal scaling is sufficient, but we will estimate the cost of line solves here, too, for simplicity. Thus, for $N^3/2$ points, we make use of a single line solve (oriented in the direction of the mesh refinement) or point solve, with cost 1/4 of the relaxation in a plane solve, which has relative complexity of

4/3, and of which there are six in a 3D BoxMG V-cycle, with relative complexity of 8/7. This gives a cost of (1/2)(1/4)(3/4)(1/6)(7/8) for the treatment of these regions, relative to the cost of a full multigrid V-cycle. Adding these costs gives

$$1/8 + (3/8)(1/6)(7/8) + (1/2)(1/4)(3/4)(1/6)(7/8) = 99/512$$

of the cost of a full 3D BoxMG V-cycle. While the non-uniform coarsening used by AMG should naturally coarsen in ways consistent with this proposed boundary-layer preconditioner, the expectation that BoxMG by itself outperforms AMG implies that the boundary-layer preconditioner should be the fastest approach.

7. Conclusions. This paper focuses on the solution of the linear systems of equations that arise from finite-difference discretizations of singularly perturbed reaction-diffusion equations on fitted meshes. We show that standard direct solvers scale poorly with both the mesh size (as is well known) and the perturbation parameter (due to slow performance of IEEE floating-point arithmetic in this regime). In contrast, robust multigrid methods offer nearly parameter-uniform efficiency in solving these linear systems, and the carefully derived stopping criteria ensure accurate, and efficient, approximations to the discrete solution. Existing software implementing algebraic and black-box multigrid approaches is demonstrated to offer nearly scalable performance. A new boundary-layer preconditioner is also proposed and is shown, both theoretically and numerically, to offer near-optimal scaling at still lower cost. Overall, we see speedups of factors of 40 or greater for problems with small perturbation parameters.

The approaches proposed here can clearly be extended to a number of other situations. Bilinear finite-element discretizations on fitted meshes generate similar stencils to those considered here and, as such, it is expected that simple extensions of these approaches can be applied in this case. Other layer-adapted meshes, such as hybrid Bakhvalov-Shishkin meshes (e.g., [42]), and variants due to Vulanović (e.g., [51]), create similar structures; in these cases, very similar approaches should also yield near-optimal scaling solution algorithms for the resulting linear systems. The developed theory for the boundary-layer preconditioner, in particular, is directly applicable to finite-difference discretizations on any fitted mesh that yields uniform meshwidth away from the boundary and corner layers, while the stopping criterion derived depends only on the minimum mesh width and existing theoretical error estimates for the discretization.

8. Acknowledgments. The authors would like to thank Iain Duff for his helpful discussions that exposed the issue of subnormal numbers in causing the slowdown observed with sparse direct methods for these problems.

REFERENCES

[1] R. E. Alcouffe, A. Brandt, J. E. Dendy, and J. W. Painter, *The multigrid method for the diffusion equation with strongly discontinuous coefficients*, SIAM J. Sci. Stat. Comput., 2 (1981), pp. 430–454.
[2] A. R. Ansari and A. F. Hegarty, *A note on iterative methods for solving singularly perturbed problems using non-monotone methods on Shishkin meshes.*, Comput. Methods Appl. Mech. Eng., 192 (2003), pp. 3673–3687.
[3] I. Babuška and M. Suri, *On locking and robustness in the finite element method.*, SIAM J. Numer. Anal., 29 (1992), pp. 1261–1293.
[4] N. Bakhvalov, *Towards optimization of methods for solving boundary value problems in the presence of boundary layers*, Zh. Vychisl. Mat. i Mat. Fiz., 9 (1969), pp. 841–859.
[5] B. Bergen, G. Wellein, F. Hülsemann, and U. Rüde, *Hierarchical hybrid grids: achieving TERAFLOP performance on large scale finite element simulations*, Int. J. Parallel Emergent Distrib. Syst., 22 (2007), pp. 311–329.
[6] J. H. Bramble, J. E. Pasciak, J. Wang, and J. Xu, *Convergence estimates for multigrid algorithms without regularity assumptions*, Math. Comp., 57 (1991), pp. 23–45.
[7] A. Brandt, *Multi–level adaptive techniques (MLAT) for singular–perturbation problems*, in Numerical Analysis of Singular Perturbation Problems, P. W. Hemker and J. J. H. Miller, eds., Academic Press, New York, 1979, pp. 53–142.
[8] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A Multigrid Tutorial*, SIAM Books, Philadelphia, 2000. Second edition.
[9] N. M. Chadha and N. Kopteva, *A robust grid equidistribution method for a one-dimensional singularly perturbed semi-linear reaction-diffusion problem*, IMA J. Numer. Anal., 31 (2011), pp. 188–211.
[10] D. Chen, S. MacLachlan, and M. Kilmer, *Iterative parameter choice and algebraic multigrid for anisotropic diffusion denoising*, SIAM J. Sci. Comput., 33 (2011), pp. 2972–2994.
[11] Y. Chen, T. A. Davis, W. W. Hager, and S. Rajamanickam, *Algorithm 887: CHOLMOD, supernodal sparse cholesky factorization and update/downdate*, ACM Trans. Math. Softw., 35 (2008), pp. 22:1–22:14.
[12] C. Clavero, J. Gracia, and E. O'Riordan, *A parameter robust numerical method for a two dimensional reaction-diffusion problem.*, Math. Comput., 74 (2005), pp. 1743–1758.

[13] T. A. Davis, *Direct methods for sparse linear systems.*, Fundamentals of Algorithms 2. Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM). xii, 217. , 2006.
[14] S. Davydycheva, V. Druskin, and T. Habashy, *An efficient finite-difference scheme for electromagnetic logging in 3D anisotropic inhomogeneous media*, Geophysics, 68 (2003), pp. 1525–1536.
[15] J. E. Dendy, *Black box multigrid*, J. Comput. Phys., 48 (1982), pp. 366–386.
[16] V. Druskin and L. Knizhnerman, *Gaussian spectral rules for the three-point second differences. I. A two-point positive definite problem in a semi-infinite domain*, SIAM J. Numer. Anal., 37 (2000), pp. 403–422.
[17] I. S. Duff, *MA57 – a code for the solution of sparse symmetric definite and indefinite systems.*, ACM Trans. Math. Softw., 30 (2004), pp. 118–144.
[18] P. A. Farrell, A. F. Hegarty, J. J. H. Miller, E. O'Riordan, and G. I. Shishkin, *Robust Computational Techniques for Boundary Layers*, no. 16 in Applied Mathematics, Chapman & Hall/CRC, Boca Raton, U.S.A., 2000.
[19] P. A. Farrell and G. I. Shishkin, *On the Convergence of Iterative Methods for Linear Systems arising from Singularly Perturbed Equations*, in Proc. Copper Mountain Conf. on Iterative Methods, 1998, pp. 1–7.
[20] F. Gaspar, C. Clavero, and F. Lisbona, *Some numerical experiments with multigrid methods on Shishkin meshes.*, J. Comput. Appl. Math., 138 (2002), pp. 21–35.
[21] F. Gaspar, F. Lisbona, and C. Clavero, *Multigrid Methods and Finite Difference Schemes for 2D Singularly Perturbed Problems*, in Numerical Analysis and Its Applications, L. Vulkov, P. Yalamov, and J. Wasniewski, eds., vol. 1988 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2001, pp. 128–135.
[22] A. George, *Nested dissection of a regular finite element mesh*, SIAM J. Numer. Anal., 10 (1973), pp. 345–363.
[23] A. J. Hoffman, M. S. Martin, and D. J. Rose, *Complexity bounds for regular finite difference and finite element grids*, SIAM J. Numer. Anal., 10 (1973), pp. 364–369.
[24] T. B. Jönsthövel, *Improvement of a multigrid solver for 3D EM diffusion*, Master's thesis, Delft University of Technology, 2006.
[25] R. Kellogg, N. Madden, and M. Stynes, *A parameter-robust numerical method for a system of reaction-diffusion equations in two dimensions.*, Numer. Methods Partial Differ. Equations, 24 (2008), pp. 312–334.
[26] R. B. Kellogg, T. Linss, and M. Stynes, *A finite difference method on layer-adapted meshes for an elliptic reaction-diffusion system in two dimensions.*, Math. Comput., 77 (2008), pp. 2085–2096.
[27] N. Kopteva and M. Stynes, *A Robust Adaptive Method for a Quasi-Linear One-Dimensional Convection- Diffusion Problem*, SIAM Journal on Numerical Analysis, 39 (2001), pp. 1446–1467.
[28] O. Lawlor, H. Govind, I. Dooley, M. Breitenfeld, and L. Kale, *Performance degradation in the presence of subnormal floating-point values*, in Proceedings of the International Workshop on Operating System Interference in High Performance Applications, September 2005.
[29] T. Linss, *Layer-adapted meshes for reaction-convection-diffusion problems.*, Lecture notes in Mathematics 1985. Berlin: Springer. xi, 320 p., 2010.
[30] F. Liu, N. Madden, M. Stynes, and A. Zhou, *A two-scale sparse grid method for a singularly perturbed reaction–diffusion problem in two dimensions*, IMA J. Numer. Anal., 29 (2009), pp. 986–1007.
[31] S. P. MacLachlan, J. D. Moulton, and T. P. Chartier, *Robust and adaptive multigrid methods: comparing structured and algebraic approaches*, Numer. Linear Alg. Appl., 19 (2012), pp. 389–413.
[32] S. P. MacLachlan, J. M. Tang, and C. Vuik, *Fast and robust solvers for pressure-correction in bubbly flow problems*, J. Comput. Phys., 227 (2008), pp. 9742–9761.
[33] J. J. H. Miller, E. O'Riordan, and G. I. Shishkin, *Fitted numerical methods for singular perturbation problems – error estimates in the maximum norm for linear problems in one and two dimensions*, World Scientific, 1996.
[34] K. Morton, *Numerical solution of convection-diffusion problems.*, Applied Mathematics and Mathematical Computation. 12. London: Chapman & Hall. xii, 372 p., 1996.
[35] W. Mulder, *Geophysical modelling of 3D electromagnetic diffusion with multigrid*, Comput. Visual. Sci., 11 (2008), pp. 129–138.
[36] M. A. Olshanskii and A. Reusken, *On the convergence of a multigrid method for linear reaction-diffusion problem*, Computing, 65 (2000), pp. 193–202.
[37] M. L. Overton, *Numerical Computing with IEEE Floating Point Arithmetic*, Society for Industrial and Applied Mathematics, 2001.
[38] M. H. Protter and H. F. Weinberger, *Maximum principles in differential equations*, Springer-Verlag, New York, 1984. Corrected reprint of the 1967 original.
[39] B. Reps, W. Vanroose, and H. bin Zubair, *On the indefinite Helmholtz equation: complex stretched absorbing boundary layers, iterative analysis, and preconditioning*, J. Comput. Phys., 229 (2010), pp. 8384–8405.
[40] H.-G. Roos, *A note on the conditioning of upwind schemes on Shishkin meshes.*, IMA J. Numer. Anal., 16 (1996), pp. 529–538.
[41] H.-G. Roos, M. Stynes, and L. Tobiska, *Robust Numerical Methods for Singularly Perturbed Differential Equations*, vol. 24 of Springer Series in Computational Mathematics, Springer-Verlag, Berlin, 2nd ed., 2008.
[42] H.-G. Roos and H. Zarin, *A second-order scheme for singularly perturbed differential equations with discontinuous source term*, J. Numer. Math., 10 (2002), pp. 275–289.
[43] J. W. Ruge and K. Stüben, *Algebraic multigrid (AMG)*, in Multigrid Methods, S. F. McCormick, ed., vol. 3 of Frontiers in Applied Mathematics, SIAM, Philadelphia, PA, 1987, pp. 73–130.
[44] G. I. Shishkin, *Grid approximation of singularly perturbed elliptic and parabolic equations*. Second doctoral thesis, Keldysh Institute, Moscow, 1991. In Russian.
[45] G. I. Shishkin and L. Shishkina, *Difference methods for singular perturbation problems*, Monographs and Surveys in Pure and Applied Math, Chapman & Hall/CRC, 2008.
[46] I. Singer and E. Turkel, *A perfectly matched layer for the Helmholtz equation in a semi-infinite strip*, J. Comput. Phys., 201 (2004), pp. 439–465.

[47] R. Stevenson, *A robust hierarchical basis preconditioner on general meshes*, Numerische Mathematik, 78 (1997), pp. 269–303. 10.1007/s002110050313.
[48] K. Stüben, *An introduction to algebraic multigrid*, in Multigrid, U. Trottenberg, C. Oosterlee, and A. Schüller, eds., Academic Press, London, 2001, pp. 413–528.
[49] U. Trottenberg, C. W. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, London, 2001.
[50] N. Umetani, S. P. MacLachlan, and C. W. Oosterlee, *A multigrid-based shifted-Laplacian preconditioner for a fourth-order Helmholtz discretization*, Numer. Linear Alg. Appl., 16 (2009), pp. 603–626.
[51] R. Vulanović, *A priori meshes for singularly perturbed quasilinear two-point boundary value problems.*, IMA J. Numer. Anal., 21 (2001), pp. 349–366.
[52] J. Xu and Y. Zhu, *Uniform convergent multigrid methods for elliptic problems with strongly discontinuous coefficients*, Math. Models Methods Appl. Sci., 18 (2008), pp. 77–105.
[53] J. Xu and L. Zikatanov, *The method of alternating projections and the method of subspace corrections in Hilbert space*, J. Amer. Math. Soc., 15 (2002), pp. 573–597.
[54] H. bin Zubair, S. P. MacLachlan, and C. W. Oosterlee, *A geometric multigrid method based on L-shaped coarsening for PDEs on stretched grids*, Numer. Linear Alg. Appl., 17 (2010), pp. 871–894.
[55] H. bin Zubair, C. W. Oosterlee, and R. Wienands, *Multigrid for high-dimensional elliptic partial differential equations on non-equidistant grids*, SIAM J. Sci. Comput., 29 (2007), pp. 1613–1636.