PROJECTION MULTILEVEL METHODS FOR QUASILINEAR ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS: NUMERICAL RESULTS

THOMAS A. MANTEUFFEL[†], STEPHEN F. MCCORMICK[†], OLIVER RÖHRLE[†], AND JOHN RUGE[†]

Abstract. The goal of this paper is to introduce a new multilevel solver for two-dimensional elliptic systems of nonlinear partial differential equations (PDEs), where the nonlinearity is of the type $u\partial v$. The incompressible Navier-Stokes equations are an important representative of this class and are the target of this study. Using a first-order system least-squares (FOSLS) approach and introducing a new variable for ∂v, for this class of PDEs we obtain a formulation in which the nonlinearity appears as a product of two different dependent variables. The result is a system that is linear within each variable but nonlinear in the cross terms. In this paper, we introduce a new multilevel method that treats the nonlinearities directly. This approach is based on a multilevel projection method (PML [23]) applied to the FOSLS functional. The implementation of the discretization process, relaxation, coarse-grid correction, and cycling strategies is discussed, and optimal performance is established numerically. A companion paper [22] establishes a two-level convergence proof for this new multilevel method.

Key words. Projection Method, Multigrid, Least Squares, Finite Elements, Quasilinear PDEs, Navier-Stokes

AMS subject classifications. 35J60, 65N12, 65N30, 65N55

1. Introduction.

The goal of nonlinear solution techniques is to solve the discretized nonlinear PDEs efficiently and accurately. Many popular, efficient methods for this purpose are based on multilevel strategies and all require a linearization process somewhere in the algorithm. These methods can be grouped into two broad categories, depending on when and how they apply the linearization step: global linearization such as Newton-type methods (cf. [14, 24]) and local linearization such as Brandt's FAS (Full Approximation Scheme; cf. [6]) or Hackbusch's similar NMGM scheme (Nonlinear Multigrid Method; cf. [16]).

Global linearization methods usually involve the solution of large linear systems of equations. Since substantial multigrid research is directed on developing robust, fast, and efficient linear solvers, there is an extensive repertoire of algorithms and knowledge to draw upon in this category of techniques. On the other hand, it is well known that the basin of attraction for efficient global linearization methods can be relatively small. Some of these problems might be handled by a full multigrid or nested iteration process that uses coarse-level processing to provide fine-level initial guesses. But problems with very small basins of attraction might need more expensive global search methods. Local linearization methods tend to have a bigger basin of attraction, but often rely on rediscretizing each of the coarser levels separately. This might result in some loss of robustness since, for some problems with strong nonlinearities, the discretization on coarse levels might not accurately reflect the finer-level properties.

Although most research for nonlinear multilevel methods certainly focuses on these two main categories, there also exist other nonlinear solution techniques. For

[†]Department of Applied Mathematics, Campus Box 526, University of Colorado at Boulder, Boulder, CO, 80309–0526. *Email*: {tmanteuf, stevem, roehrle, jruge}@colorado.edu. This work was sponsored by the Department of Energy under grant numbers DE-FC02-01ER25479 and DE-FG02-03ER25574, Lawrence Livermore National Laboratory under contract number B533502, Sandia National Laboratory under contract number 15268, and the National Science Foundation under VIGRE grant number DMS-9810751.

example, Yavneh and Dardyk [13] propose a nonlinear multigrid method that combines global and local linearization. Apparently, it is *"at least as good as the more suitable of these two approaches, and often better than both.* [13]*"*

This paper introduces a new multilevel method that fits into neither of these two categories, since we do not appeal to a linearization process anywhere in the algorithm. To achieve this direct approach, we focus on PDEs with nonlinear terms of type $u\partial v$, especially the incompressible Navier-Stokes equations, that we reformulate as a least-squares problem. Least-squares methods are based on a minimization principle for a functional constructed by taking the residual of the governing equations in some Hilbert norm. The intent is to ensure that the minimizer is the solution of the original set of equations and that the formulation is well posed.

Least-squares methods for the Navier-Stokes equations have been addressed, for example, by Bochev and Gunzburger [4], Jiang [18], and Bochev, Cai, Manteuffel, and McCormick [1, 2]. In this paper, we consider a first-order system least-squares (FOSLS) method, where the functional is constructed by taking the L^2-norm of each interior first-order equation.

Instead of using FAS, Newton, or Newton-like methods to solve the resulting algebraic equations, we want to develop a new multigrid algorithm that can treat the nonlinearity directly and, thus, potentially more effectively. To this end, we consider a *projection multilevel method* (PML; cf. [23]) that solves an optimization problem by correcting a current approximation using projections onto various subspaces. In the context of FOSLS, the solution to a PDE is the minimizer of the FOSLS functional. So, naturally, we choose the minimization of the FOSLS functional as the optimization problem for our projection method. The minimization is done by corrections from certain finite element subspaces by way of the natural embedding of these spaces into the fine-grid space. The projection of the error that this defines is orthogonal with respect to the inner product associated with the functional, because it is defined as the approximation to the error from the given subspace that is best in the sense of minimizing the functional.

The multilevel projection method idea is not new. Projection multilevel methods have been used by Mandel and McCormick for eigenvalue problems (cf. [21]), by Gelman and Mandel for constraint optimization problems (cf. [15]), and by McCormick for parameter estimation, transport equations, general eigenvalue problems, Riccati equations, finite volume element methods, and image reconstruction (cf. [23]). In fact, under certain circumstances, these methods relate to specific forms of classical multilevel methods. Consider, for example, the standard fully-variational multigrid method applied to the Poisson problem in two dimensions, as given in [26], with Gauß-Seidel as the smoother, full coarsening, bilinear interpolation, and a 9-point stencil. This classical algorithm could also be classified as a projection multilevel method. It can be interpreted, at each stage, as a Rayleigh-Ritz method applied to minimizing the energy functional, where the optimization is taken as a correction over the continuous space projected onto certain subspaces of the fine-grid finite element space. This exemplifies that there exist, under certain circumstances, similarities and relations between a standard multigrid and a projection multilevel method. In fact, PML exhibits the same basic principles as any other multilevel algorithm. Such principles include appropriate discretizations for the fine-grid problem, relaxation, coarsening, coarse-grid solves, interpolation, and cycling strategies.

The challenge in developing such a scheme is to ensure that the cost of processing coarse levels is less expensive in total than that of the fine grid. The major task in addressing this challenge is to cast the coarse subspace projection in terms of coarse-level computation. This ability we call coarse-grid realizability. We show below how this is done for our scheme applied to the Navier-Stokes equations.

To illustrate the basic ideas and principles of this new PML method, we introduce in Section 2 a projection-based discretization process. Based on this process, we derive in Section 3 an abstract framework for PML. In Section 4, we discuss how coarse-grid realizability can be done efficiently for quasilinear PDEs, for which the highest-order terms are linear. Additionally, we show that this is also feasible for different relaxation types and higher-order discretizations. We conclude this paper by giving numerical results (Section 5) for model problems in two dimensions and making a few general remarks. While the numerical results in Section 5 show optimal convergence properties, we provide in the companion paper [22] a two-level convergence proof.

2. Embedding Operators and Discretization by Projection.

As for any numerical scheme that discretely approximates continuous problems, the discretization process plays an important role. This process is even more important for multilevel schemes since they use a sequence of coarse-grid discretizations that must in some sense be compatible with the fine-grid discretization. For our particular PML method, we want to exploit a natural discretization process by using the same approach on all levels. Even though this seems to be the most natural and straightforward way to obtain discretizations for all levels, there exist other methodologies for which it is more advantageous to use a variational type of discretization process instead. Algebraic multigrid (AMG) is just one example. On coarser levels, AMG applied to a discretized PDE obtains matrices that often differ from what one would obtain using a discretization process that is analogous to that used on the finest level. For further details on AMG, see [7, 9, 25].

One way to relate a continuous PDE to a discrete problem is to think of the discretization process as a projection from an infinite-dimensional space onto a discrete one, with some nodal or finite element representation. (Here we restrict ourselves to a finite element representation.) To illustrate this process, consider a partial differential operator, $\mathcal{L}$, which maps between two infinite-dimensional spaces, $\boldsymbol{\mathcal{V}}$ and $\boldsymbol{\mathcal{V}}_{\mathcal{L}}$ ($\mathcal{L} : \boldsymbol{\mathcal{V}} \to \boldsymbol{\mathcal{V}}_{\mathcal{L}}$). For a specific $\mathsf{g} \in \boldsymbol{\mathcal{V}}_{\mathcal{L}}$ and domain Ω, we formally obtain a PDE, which we denote by

$$\mathcal{L}(\mathbf{x}) = \mathsf{g}, \qquad \text{in } \Omega. \tag{2.1}$$

For equation (2.1) to be properly defined, it may need to be taken in the weak sense, but this would complicate the discussion. We use the strong form here for simplicity. Note the use of bold face type for unknown $\mathbf{x}$ and source term g. We do this to allow for different types of principal variables, such as pressure, temperature, and velocity. When we want to emphasize this possibility, we write these variables in component form, such as $\mathbf{x} = (\mathrm{x}_1, \dots, \mathrm{x}_v)^t$.

Now let $\mathcal{S}^h$ be a finite-dimensional subset of $\boldsymbol{\mathcal{V}}$ (e.g., a standard finite element space associated with an approximate mesh size, h). Then denote the natural embedding operator by $\mathcal{P}^h : \mathcal{S}^h \hookrightarrow \boldsymbol{\mathcal{V}}$. This operator leads to a natural discretization of our functional minimization problem as follows. Consider the least-squares functional associated with (2.1):

$$\mathcal{F}(\mathbf{x}; \mathsf{g}) = \left\| \mathcal{L}(\mathbf{x}) - \mathsf{g} \right\|_{0,\Omega}^2, \qquad \forall \mathbf{x} \in \boldsymbol{\mathcal{V}}. \tag{2.2}$$

Note that $\mathcal{F}(\,\cdot\,; \mathsf{g})$ is a mapping from $\boldsymbol{\mathcal{V}}$ to $\mathbb{R}$. A discrete functional is obtained by defining $\mathcal{F}^h(\mathbf{x}^h; \mathsf{g}) := \mathcal{F}(\mathcal{P}^h \mathbf{x}^h; \mathsf{g})$, with $\mathbf{x}^h \in \mathcal{S}^h$ and $\mathcal{P}^h \mathbf{x}^h \in \boldsymbol{\mathcal{V}}$. Discretization is thus simply a matter of restricting the functional to the discrete space. This is the essence of Rayleigh-Ritz. Note that since $\mathcal{F}^h(\,\cdot\,;\mathrm{g})$ is a mapping from $\mathcal{S}^h$ to $\mathbb{R}$, notations $\mathcal{F}(\mathbf{x}^h;\mathrm{g})$ and $\mathcal{F}^h(\mathbf{x}^h;\mathrm{g})$ are equivalent. From now on, we refer to $\mathcal{F}(\mathbf{x}^h;\mathrm{g})$ as the discrete functional.

The abstract discretization process only depends on the choice of the embedding operator, $\mathcal{P}^h$, and the associated finite element space, $\mathcal{S}^h$. Hence, for coarser levels, we can define the discrete functional in the same way. Let $\mathcal{S}^{2h}$ be a finite-dimensional space (associated with an approximate mesh size, $2h$) and let $\mathcal{P}^{2h}:\mathcal{S}^{2h}\hookrightarrow\boldsymbol{\mathcal{V}}$ be the natural embedding from $\mathcal{S}^{2h}$ into $\boldsymbol{\mathcal{V}}$. Then the coarse-grid discretization of functional (2.2) is given by $\mathcal{F}(\mathbf{x}^{2h};\mathrm{g}) := \mathcal{F}(\mathcal{P}^{2h}\mathbf{x}^{2h};\mathrm{g})$, with $\mathbf{x}^{2h}\in\mathcal{S}^{2h}$. In our framework, for consecutive coarser levels, we typically choose nested spaces, so that $\mathcal{S}^{2^L h}\subset \ldots \mathcal{S}^{2h}\subset\mathcal{S}^h\subset\boldsymbol{\mathcal{V}}$. In this way, the interlevel transfer operators are induced in a natural, straightforward, and advantageous way and are easy to implement within PML. Furthermore, the coarse-grid problems are ensured to be compatible with the procedures used to define the fine-level problem, with the difference being that the coarse-level unknown is an approximation to the fine-level error and not to its solution; that is, the coarse-level correction is of the form $\mathbf{x}^h+\mathbf{c}^{2h}$. (Since $\mathbf{x}^h=\mathcal{P}^h\mathbf{x}^h$ for $\mathbf{x}^h\in\mathcal{S}^h$ and $\mathbf{c}^{2h}=\mathcal{P}^{2h}\mathbf{c}^{2h}$ for $\mathbf{c}^{2h}\in\mathcal{S}^{2h}$, we omit the embedding operators $\mathcal{P}^h$ and $\mathcal{P}^{2h}$ here and henceforth.)

Note that relaxation also depends on the choice of subspaces (and, hence, on the embeddings). We thus have to be particularly careful in picking the underlying subspaces for relaxation and coarsening.

3. Abstract Framework of PML.

To describe the general framework of a PML method applied to a functional minimization principle, let $\mathcal{F}(\mathbf{x};\mathrm{g}):\boldsymbol{\mathcal{V}}\to\mathbb{R}$ and assume that we have a conforming finite element structure in the sense that $\mathcal{S}^{2h}\subset\mathcal{S}^h\subset\boldsymbol{\mathcal{V}}$. The aim of this section is to develop a multilevel framework that applies directly to

$$\mathcal{F}(\mathbf{x}^h_*;\mathrm{g}) = \min_{\mathbf{x}^h\in\mathcal{S}^h}\mathcal{F}(\mathbf{x}^h;\mathrm{g}), \qquad \mathbf{x}^h_*\in\mathcal{S}^h. \tag{3.1}$$

To do this, we focus on two important ingredients of multilevel methods: relaxation and coarsening. Relaxation is a generic term for an iterative process that is typically very inexpensive to use but is effective only at reducing certain 'oscillatory' error components. Coarsening refers to the process of determining a coarse-level correction that hopefully eliminates the 'smooth' errors that relaxation leaves behind.

We first provide a general framework for a point or nodal relaxation scheme on the finest level. To maintain a certain form of generality in this section, let $\{\phi^h_n\}_{n=1}^{m_0}$ be a basis for $\mathcal{S}^h$, where m_0 is the dimension of $\mathcal{S}^h$. Then write $\mathcal{S}^h$ as a direct sum of the one-dimensional subspaces, $\mathcal{S}^h_n=\operatorname{span}\{\phi^h_n\}$, $1\le n\le m_0$: $\mathcal{S}^h=\mathcal{S}^h_1\oplus\ldots\oplus\mathcal{S}^h_{m_0}$. (Higher-dimensional subspaces can be considered for relaxation processes that update several variables at once, e.g., line or box relaxation. However, we consider only the one-dimensional case here for simplicity.)

These definitions set the stage for an abstract definition of a relaxation scheme to approximately solve for $\mathbf{x}^h_*\in\mathcal{S}^h$ in (3.1) by PML. To do so, we want to improve an initial guess, $\mathbf{x}^h$, by corrections, $\mathbf{c}^h\in\mathcal{S}^h_n$, $1\le n\le m_0$. Thus, one sweep of relaxation consists of performing the following correction steps for each $n=1,2,\ldots,m_0$ in turn:

$$\begin{cases} \mathcal{F}(\mathbf{x}^h+\mathbf{c}^h_n;\mathrm{g}) = \displaystyle\min_{\mathbf{c}^h_n\in\mathcal{S}^h_n}\mathcal{F}(\mathbf{x}^h+\mathbf{c}^h_n;\mathrm{g}), \\ \mathbf{x}^h\leftarrow\mathbf{x}^h+\mathbf{c}^h_n. \end{cases} \tag{3.2}$$

Next, consider the coarse-grid correction process, first in terms of an exact solve, then as an iterative process. Let $\mathbf{x}^h \in \mathcal{S}^h$ be a random initial guess or a current iterate for our PML scheme. Then, the exact coarse-grid solve is described by

$$\mathcal{F}(\mathbf{x}^h + \mathbf{c}_*^{2h}; \mathrm{g}) = \min_{\mathbf{c}^{2h} \in \mathcal{S}^{2h}} \mathcal{F}(\mathbf{x}^h + \mathbf{c}^{2h}; \mathrm{g}), \qquad \mathbf{c}_*^{2h} \in \mathcal{S}^{2h}. \tag{3.3}$$

To develop an iterative version of (3.3), we proceed in analogy to fine-grid relaxation. Let $\{\phi_n^{2h}\}_{n=1}^{m_1}$ be a basis for $\mathcal{S}^{2h}$. Then, write $\mathcal{S}^{2h}$ as a direct sum of the one-dimensional subspaces, $\mathcal{S}_n^{2h} = \mathrm{span}\{\phi_n^{2h}\}$, $1 \le n \le m_1$: $\mathcal{S}^{2h} = \mathcal{S}_1^{2h} \oplus \ldots \oplus \mathcal{S}_{m_1}^{2h}$. Then one coarse-grid relaxation sweep consists of performing the following correction steps for each $n = 1, 2, \ldots, m_1$ in turn:

$$\begin{cases} \mathcal{F}(\mathbf{x}^h + \mathbf{c}_n^{2h}; \mathrm{g}) = \min\limits_{\mathbf{c}_n^{2h} \in \mathcal{S}_n^{2h}} \mathcal{F}(\mathbf{x}^h + \mathbf{c}_n^{2h}; \mathrm{g}), \\ \mathbf{x}^h \leftarrow \mathbf{x}^h + \mathbf{c}_n^{2h}. \end{cases} \tag{3.4}$$

Our notation is at the crux of our ability to make PML practical. Iterative methods are commonly formulated as a processes that directly updates the original approximation, $\mathbf{x}^h$. Our choice of the more complicated correction form in (3.2) was made for consistency with (3.4). We complicate this notation further below by writing the respective fine- and coarse-level iterative processes as corrections to the approximate solutions, $\mathbf{c}^h$ and $\mathbf{c}^{2h}$, of (3.2) and (3.4). (To avoid further complication, we use $\mathbf{c}^h$ and $\mathbf{c}^{2h}$ to denote either the exact solutions or their approximations, a distinction that is clear by context.) Furthermore, we use these formulations to allow the multiple corrections that come from yet coarser levels. Hopefully, the three-term correction forms that we use in what follows are enough to expose the mechanisms needed to make the process efficient. The key is to write relaxation on the level $4h$ correction as a process that only involves changes to level $4h$ vectors.

To compute the corrections in (3.2) and (3.4), we use fine- and coarse-level relaxation processes. For $\mathbf{x}^h \in \mathcal{S}^h$ fixed and $\mathbf{c}^h \in \mathcal{S}^h$, the current approximation to the exact correction defined in (3.2), the nth step of a fine-grid relaxation sweep is defined by solving

$$s = \operatorname*{argmin}_{t \in \mathbb{R}} \mathcal{F}(\mathbf{x}^h + \mathbf{c}^h + t\mathbf{d}^h; \mathrm{g}), \qquad s \in \mathbb{R}. \tag{3.5}$$

and forming the update,

$$\mathbf{c}^h \leftarrow \mathbf{c}^h + s\mathbf{d}^h, \tag{3.6}$$

where $\mathbf{d}^h = \phi_n^h$, $1 \le n \le m_0$. Note that (3.5) and (3.6) describe a basic line search method, in direction $\mathbf{d}^h$, with optimal step length s. For simplicity, we combine (3.5) and (3.6) and refer to it as a *directional iteration step*. For a given $\mathbf{x}^h$, $\mathbf{c}^h$, and $\mathbf{d}^h$, we denote the operator describing (3.5) and (3.6) by

$$\mathbf{c}^h \leftarrow \mathcal{D}_{\mathbf{x}^h}(\mathbf{c}^h, \mathbf{d}^h). \tag{3.7}$$

In an analogous way, coarse-grid relaxation is defined for $\mathbf{x}^h \in \mathcal{S}^h$ and $\mathbf{c}^{2h} \in \mathcal{S}^{2h}$ by $\mathbf{c}^{2h} \leftarrow \mathcal{D}_{\mathbf{x}^h}(\mathbf{c}^{2h}, \mathbf{d}^{2h})$, where $\mathbf{d}^{2h} = \phi_n^{2h}$, $1 \le n \le m_1$. Successive application of this process yields an abstract formulation of a general multilevel projection method. Assume that there are $L + 1$ distinct grid levels corresponding to mesh sizes $2^l h$, $l = 0, \ldots, L$. (We label the finest level by superscript h and the coarsest one by superscript $2^L h$.) Assume that each level is defined by a finite-dimensional subspace, $\mathcal{S}^{2^l h}$, that is nested in the sense that $\mathcal{S}^{2^{l+1}h} \subset \mathcal{S}^{2^l h}$, $l = 0, \ldots, L-1$. Suppose that these spaces are written as a direct sum of one-dimensional subspaces: $\mathcal{S}^{2^l h} = \mathcal{S}_1^{2^l h} \oplus \ldots \oplus \mathcal{S}_{m_l}^{2^l h}, l = 0, \ldots, L$. Then one $V(0,1)$-PML cycle is defined as follows:

```
$\mathbf{c}^{2^l h} \leftarrow \mathbf{0}, \qquad l = 0, \ldots, L$
For $l = L, \ldots, 1$:        *(coarse-grid process)*
    For: $n = 0, \ldots, m_l$
        $\mathbf{c}^{2^l h} \leftarrow \mathcal{D}_{\mathbf{x}^h}(\mathbf{c}^{2^l h}, \mathbf{d}_n^{2^l h}), \qquad \mathbf{d}_n^{2^l h} \in \mathcal{S}_n^{2^l h},$
    $\mathbf{c}^{2^{l-1}h} = \mathbf{c}^{2^l h}$                                   (3.8)
For $l = 0$:                   *(fine-grid process)*
    For: $n = 0, \ldots, m_0$
        $\mathbf{c}^h \leftarrow \mathcal{D}_{\mathbf{x}^h}(\mathbf{c}^h, \mathbf{d}_n^h), \qquad \mathbf{d}_n^h \in \mathcal{S}_n^h,$
$\mathbf{x}^h \leftarrow \mathbf{x}^h + \mathbf{c}^h.$
```

4. Coarse-grid Realizability, Different Relaxation Types, and Higher-Order Discretizations.

The key to obtaining an efficient multigrid-optimal PML implementation from (3.8) is the capability to perform the directional iteration step efficiently on coarse levels. Since the directional iteration step is based on functional evaluations, we focus now on how to do this efficiently on coarse levels.

4.1. Coarse-grid Realizability.

To obtain optimality, our multigrid algorithm must achieve two key objectives. First, we must be able to compute the FOSLS functional efficiently. Thus, update $\mathbf{c}^{2h}$ and the resulting new functional value must be computed quickly. Essentially, all level $2h$ calculations should in effect be performed on grid $2h$, not on grid h. Second, it must be possible to go from level $2h$ to level $4h$ without first updating the approximations on grid h.

To show how the first objective can be achieved, we need some additional notation and definitions. For simplicity, we choose the discretization to be the space, $\mathcal{S}^h$, of continuous piecewise-linear functions and the domain, $\Omega \subset \mathbb{R}^2$, to be two-dimensional, simply-connected, and polygonal so that it can be partitioned into triangles. We consider here only triangulations by standard linear Lagrange triangles (cf. [5]). We need to maintain a certain block-structured grid in order to obtain an efficient multigrid-optimal implementation of PML in two dimensions. Each level is defined by a finite-dimensional subspace, $\mathcal{S}^{2^l h}$, nested in the sense that $\mathcal{S}^{2^{l+1}h} \subset \mathcal{S}^{2^l h}$, $l = 0, \ldots, L-1$. For this section, we consider $\mathbf{x}^h = (\mathrm{x}_1^h, \ldots, \mathrm{x}_v^h)^t \in \mathcal{S}^h$ to be an arbitrary but fixed fine-grid approximation to the solution of the PDE. The $\mathbf{x}^h$ components, $\mathrm{x}_i^h : \Omega \to \mathbb{R}$ $(i = 1, \ldots, v)$, represent the different principal PDE variables (e.g., pressure, temperature, energy, and velocity). Further, we denote with $\mathbf{c}^{2^l h}$ a correction to fine-grid approximation $\mathbf{x}^h$ on level l (with approximate mesh size $2^l h$). Each component of correction $\mathbf{c}^{2^l h} = (\mathrm{c}_1^{2^l h}, \ldots, \mathrm{c}_v^{2^l h})^t \in \mathcal{S}^{2^l h}$ is a continuous piecewise-linear function and can be written, restricted to an element $\Omega_j^{2^l h}$, as a linear function as follows:

$$\mathrm{c}_i^{2^l h}(x,y)\Big|_{\Omega_j^{2^l h}} = s_1^{(i,j,l)} + s_2^{(i,j,l)} x + s_3^{(i,j,l)} y. \tag{4.1}$$

This representation holds for all $i = 1, \ldots, v$ and $j = 1, 2, ..., N^{(l)}$, where $N^{(l)}$ is the total number of elements on level l. The coefficients, $s_p^{(i,j,l)}$ with $p = 1, 2, 3$, are determined uniquely on level l by solving on each element, $\Omega_j^{2^l h}$ $(j = 1, 2, ..., N^{(l)})$, and for each $i = 1, \ldots, v$ the corresponding linear interpolation problem. In contrast to standard finite element practice, (4.1) can be seen as an alternative way to obtain a representation of $\mathbf{c}^h$ in $\mathcal{S}^h$.

We next show how $\mathcal{F}(\mathbf{x}^h + \mathbf{c}^h; \mathsf{g})$ is computed for a modifiable fine-grid correction, $\mathbf{c}^h$, and an arbitrary but fixed approximation, $\mathbf{x}^h$. This is an essential step towards a multigrid-optimal algorithm and provides the basis for computing the FOSLS functional efficiently on coarser levels. For simplicity, we focus on one fine-grid element, Ω_j^h. This can be done without any loss of generality, since the sum of all fine-grid element contributions, $\mathcal{F}_{\Omega_j^h}(\mathbf{x}^h + \mathbf{c}^h; \mathsf{g})$, is the functional value, $\mathcal{F}(\mathbf{x}^h + \mathbf{c}^h; \mathsf{g})$. Further, we represent the fine-grid correction, $\mathbf{c}^h$ (level $l = 0$), as in (4.1) and consider its coefficients, $s_p^{(i,j,0)}$, as unknowns. In a next step, we use this representations to express the functional contribution, $\mathcal{F}_{\Omega_j^h}(\mathbf{x}^h + \mathbf{c}^h; \mathsf{g})$, in terms of the coefficients, $s_p^{(i,j,0)}$. Due to the nature of our quasilinear first-order system and its L^2 least-squares functional, it is possible that the expansion of $\mathcal{F}_{\Omega_j^h}(\mathbf{x}^h + \mathbf{c}^h; \mathsf{g})$, with respect to the coefficients of $\mathbf{c}^h$, includes product terms of the coefficients, $s_p^{(i,j,0)}$, of up to order four. In the context of our new PML method, we regard all these terms as separate unknowns and store them as a matrix, $\mathbf{C}_j^h$. In this way, we are able to write the expansion of $\mathcal{F}_{\Omega_j^h}(\mathbf{x} + \mathbf{c}^h; \mathsf{g})$ as a matrix inner product of the form $\mathbf{A}_j^h : \mathbf{C}_j^h$. In the following, we refer to $\mathbf{A}_j^h$ as the local functional matrix and to $\mathbf{C}_j^h$ as the local coefficient matrix for element Ω_j^h on grid h. Now, whenever $\mathbf{c}^h$ changes on Ω_j^h, we obtain the new functional value, $\mathcal{F}_{\Omega_j^h}(\mathbf{x}^h + \mathbf{c}^h; \mathsf{g})$, by recomputing the local coefficient matrix and by evaluating the matrix inner product.

To show that we can compute the functional on coarser levels by coarse-grid calculations (the first objective), we assume a regular-structured grid. We further assume that four fine-grid elements always form one coarse-grid element. Denote with $\mathbf{C}_k^{2h}$ the coefficient matrix for $\mathbf{c}^{2h}$ on coarse-grid element Ω_k^{2h}. Further, let $\mathbf{C}_j^h$, with $j \in \{i | \Omega_i^h \subset \Omega_k^{2h}\}$, be the coefficient matrices for $\mathbf{c}^{2h}$ restricted to the fine-grid elements, Ω_j^h. The key observation now is to recognize that $\mathbf{C}_k^{2h} = \mathbf{C}_j^h$ for all $j \in \{i | \Omega_i^h \subset \Omega_k^{2h}\}$. Then, we obtain the functional contribution for coarse-grid element Ω_k^{2h} as follows:

$$\mathcal{F}_{\Omega_k^{2h}}(\mathbf{x}^h + \mathbf{c}^{2h}; \mathsf{g}) = \sum_j \mathbf{A}_j^h : \mathbf{C}_j^h = \left(\sum_j \mathbf{A}_j^h \right) : \mathbf{C}_k^{2h} = \mathbf{A}_k^{2h} : \mathbf{C}_k^{2h}, \tag{4.2}$$

where $j \in \{i | \Omega_i^h \subset \Omega_k^{2h}\}$. Having all local fine-grid functional matrices available, we obtain the local coarse-grid functional matrices by a simple element-by-element addition of the respective fine-grid functional matrices. In this way, we can compute the functional, $\mathcal{F}(\mathbf{x}^h + \mathbf{c}^{2h}; \mathsf{g})$, for fixed fine-grid approximation $\mathbf{x}^h \in \mathcal{S}^h$ and any $\mathbf{c}^{2h} \in \mathcal{S}^{2h}$, entirely by grid $2h$ computations. We remark that, because of the nonlinear nature of our problems, these coarse-grid functional evaluations are only possible for approximations of the form $\mathbf{x}^h + \mathbf{c}^{2^l h}$, $l = 0, \ldots, L$, and not, for example, for approximations of the form $\mathbf{x}^h + \mathbf{c}^{2h} + \mathbf{c}^{4h}$. But, to fulfill the second objective, we would have to be able to compute approximations of the second type. To circumvent this drawback, we simply restrict ourselves to $V(0,\nu)$-cycles. Such a cycle is characterized for our new PML method by the following steps: fix the current approximation/initial guess; compute all local fine-grid functional matrices, $\mathbf{A}_j^h$; calculate the corresponding local coarse-grid functional matrices for all coarse levels; start the relaxation process on the coarsest level by applying ν sweeps there to compute the correction, $\mathbf{c}^{2^L h}$; interpolate this correction to the next finer level; use the interpolated correction as an initial value for relaxation on this next finer level; repeat the steps for $l = L, \dots, 0$; and update the current approximation, $\mathbf{x}^h$, by $\mathbf{c}^h$.

By introducing local functional matrices and restricting ourselves to structured grids and $V(0,\nu)$-cycles, we can fulfill all the objectives for an efficient and multigrid optimal algorithm. Note that the same efficiency and optimality is retained if, instead of regular-structured grids, we use block-structured grids. Such grids allow the coarsest level to be unstructured, while the subsequent finer levels exhibit a regular structure.

4.2. Relaxation. The description of a $V(0,1)$-PML cycle defines relaxation in general as $\mathbf{c}^{2^l h} \leftarrow \mathcal{D}_{\mathbf{x}^h}(\mathbf{c}^{2^l h}, \mathbf{d}_n^{2^l h})$, with $n = 0, \dots, m_l$, where $\mathbf{x}^h$ is the current fine-grid approximation, $\mathbf{c}^{2^l h}$ is a coarse-grid correction on level $2^l h$, $\mathbf{d}_n^{2^l h} \in \mathcal{S}_n^{2^l h}$ is a search direction, and $\mathcal{S}_n^{2^l h}$, $n = 0, \dots, m_l$, are spaces that decompose $\mathcal{S}^{2^l h}$. We clearly see that picking $\mathcal{S}_n^{2^l h}$ characterizes the type of relaxation. Before illustrating relationships between $\mathcal{S}_n^{2^l h}$ and different relaxation types, we first comment on issues concerning the realization and implementation of directional iteration or relaxation steps in general.

The classical approach for relaxation schemes, such as Gauß-Seidel or damped Jacobi, are based on a finite number of either explicitly given linear equations, typically written in matrix form, or nonlinear equations. Since relaxation for our PML method is based on a nonlinear functional minimization principle, we cannot use them in the same way that most standard approaches present them. To implement relaxation so that we keep the overall promise of avoiding linearization while obtaining an efficient algorithm, we restrict ourselves to a first-order system least-squares functional for quasilinear PDEs. For this class of PDE formulations, the nonlinearity appears in the functional as a cross product of two different variables, which implies linearity of the weak form with respect to each variable.

To illustrate this linearity, consider a least-squares functional consisting of the product of two variables: $\mathcal{F}([u,v]^t;0) = \|uv\|_{0,\Omega}^2$. (For clarity, we use u and v instead of x_1 and x_2.) Let $\mathcal{S}^h$ be a standard finite element space with approximate mesh size h. Then choose a relaxation direction for each variable: $\mathbf{d}_1^h = [d_{u^h}^h, 0]^t \in \mathcal{S}^h$ and $\mathbf{d}_2^h = [0, d_{v^h}^h]^t \in \mathcal{S}^h$. Relaxing on each variable of $\mathcal{F}([u^h, v^h]^t;0)$ separately, we obtain for $\mathbf{x}^h = [u^h, v^h]^t \in \mathcal{S}^h$ the following relaxation process:

$$\left\{ \begin{array}{l} s_1 = \underset{t\in\mathbb{R}}{\operatorname{argmin}}\ \mathcal{F}(\mathbf{x}^h + t\mathbf{d}_1^h;0) = \underset{t\in\mathbb{R}}{\operatorname{argmin}}\ \mathcal{F}([u^h + t d_{u^h}^h, v^h]^t;0) =: \underset{t\in\mathbb{R}}{\operatorname{argmin}}\ \bar{\mathcal{F}}_1(t), \\ u^h \leftarrow u^h + s_1 d_1^h, \end{array} \right. \tag{4.3}$$

and

$$\left\{ \begin{array}{l} s_2 = \underset{t\in\mathbb{R}}{\operatorname{argmin}}\ \mathcal{F}(\mathbf{x}^h + t\mathbf{d}_2^h;0) = \underset{t\in\mathbb{R}}{\operatorname{argmin}}\ \mathcal{F}([u^h, v^h + t d_2^h]^t;0) =: \underset{t\in\mathbb{R}}{\operatorname{argmin}}\ \bar{\mathcal{F}}_2(t), \\ v^h \leftarrow v^h + s_2 d_1^h. \end{array} \right. \tag{4.4}$$

For our class of PDEs, functions $\bar{\mathcal{F}}_1(t)$ and $\bar{\mathcal{F}}_2(t)$ defined in (4.3) and (4.4) are quadratic polynomials in the scalar, t. To obtain the quadratic formulation for $\bar{\mathcal{F}}_1(t)$ (or $\bar{\mathcal{F}}_2(t)$), we evaluate $\bar{\mathcal{F}}_1(t)$ (or $\bar{\mathcal{F}}_2(t)$) at three different locations and fit the functional values quadratically. In this way, the quadratic polynomial fits $\bar{\mathcal{F}}_1(t)$ (or $\bar{\mathcal{F}}_2(t)$) exactly. Actually, we only need to evaluate $\bar{\mathcal{F}}_1(t)$ (or $\bar{\mathcal{F}}_2(t)$) at two locations because the current functional value ($t = 0$) is known. Also, after computing the optimal step length, which is the minimum of the quadratic polynomial, we obtain the new current functional value by plugging s_1 (or s_2) into our quadratically-fitted curve.

Even though we only illustrated one fine-grid relaxation step for two scalar unknowns, we can apply the same techniques for more than two variables, for unknowns that are vector functions, and on coarser levels. Moreover, at this point, we see why it is extremely important to be able to compute functional values on all levels efficiently. Relaxation is the main contributor to the overall computational cost and is almost solely based on functional evaluations.

We can relax on the unknowns in an alternating fashion, as described by (4.3) and (4.4), for almost any choice of relaxation subspaces, $\mathcal{S}_n^{2^l h}$, and discretization. These choices only affect the type of relaxation. In what follows, we give two examples for different relaxation types, a Richardson-like scheme and a Gauß-Seidel-like scheme. Although we describe the different relaxation types as if the functional had only one unknown, we still relax on the unknowns in an alternating way.

To obtain a Richardson-like relaxation scheme, we choose $m_l = 1$ on all levels. This means that there is only one relaxation step per sweep. As the single direction, $\mathbf{d}_1^{2^l h} \in \mathcal{S}_1^{2^l h} = \mathcal{S}^{2^l h}$, we make the natural choice of 'steepest' descent given by the gradient of the functional with respect to the unknown. We compute the gradient of our nonlinear functional numerically: its value at node n is determined by the forward-difference formula, $\big(\mathcal{F}(\mathbf{x}^{2^l h} + s\,\mathbf{e}_n^{2^l h}; \mathrm{g}) - \mathcal{F}(\mathbf{x}^{2^l h}; \mathrm{g})\big)/s$, where $\mathbf{e}_n^{2^l h}$ is the n-th nodal finite element basis function (with value one at grid point n and zero elsewhere) and s is sufficiently small; the discrete representation of the gradient, $\mathbf{d}_1^{2^l h}$, is then just the continuous piecewise polynomial in $\mathcal{S}^{2^l h}$ that has these nodal values.

If we now choose our relaxation subspaces as the span of individual basis or nodal finite element basis functions (with a value of one at a single node and zero at all other nodes), we obtain a coordinate minimization or nonlinear Gauß-Seidel relaxation process. Hence, we choose m_l to be equal to the number of nodes on level l, $\mathbf{d}_n^{2^l h}$ as the nodal finite element basis function, and $\mathcal{S}_n^{2^l h}$ as the space spanned by the nodal basis function of node n. This means that we minimize consecutively over all nodes, n, by computing the step length, $s = \min_{t\in\mathbb{R}} \mathcal{F}(\mathbf{x}^h + \mathbf{c}^{2^l h} + t\mathbf{d}_n^{2^l h}; \mathrm{g})$, and the resultant update, $\mathbf{c}^{2^l h} \leftarrow \mathbf{c}^{2^l h} + s\mathbf{d}_n^{2^l h}$. Note that this is a local process in that the approximation, $\mathbf{x}^h$, only changes at one node per step of the sweep.

Gauß-Seidel is typically a more efficient smoother than a gradient or Richardson-type process. This is confirmed numerically for our application in Table 5.1 of Section 5.2.

4.3. Higher-Order Discretizations. Many engineering problems require more than just a linear finite element discretization. For example, the numerical solution to the Navier-Stokes equations obtained by using linear finite elements and a triangular discretization in a FOSLS formulation usually does not conserve mass very well. This section shows the potential of using higher-order finite elements in our framework of PML. For simplicity, however, we limit ourselves in this section to standard quadratic Lagrange triangles, which generate the space, $\mathcal{S}_Q^h$, of continuous piecewise-quadratic finite elements. It should be noted that any other higher-order discretization or other element type can be implemented in a similar way. Mimicking the representation of linear functions over elements, we describe approximations or corrections in $\mathcal{S}_Q^h$ restricted to an element (in this case, we use a less cumbersome notation as we restrict our correction to a reference element, Ω_{ref}^h) by

$$c^h(x,y)\Big|_{\Omega_{ref}^h} = s_0^h + s_1^h x + s_2^h y + s_3^h xy + s_4^h x^2 + s_5^h y^2. \tag{4.5}$$

Similar to fine-grid level h, we introduce quadratic finite element spaces on coarser levels: $\mathcal{S}_Q^{2^l h}$, $l = 1, \dots, L$. The subscript Q indicates the use of quadratic ansatz functions to generate the space. We stress that the lack of such a subscript signifies linear ansatz functions. For the multilevel implementation with quadratic finite elements, we use an unstructured triangulation for the coarsest level. All finer levels are obtained by subdividing each coarser-grid-level triangle into four equal triangles. To this end, we consider two different coarse-grid correction processes that differ by the choice of the coarse-grid correction subspaces:

1. On all levels, corrections are obtained from the quadratic finite element subspaces, $\mathcal{S}_Q^{2^l h}$ $(l = 0, \dots, L)$.
2. Only $\mathcal{S}_Q^h$ is used for the fine-grid corrections, while the coarse-grid process uses corrections from the linear finite element subspaces, $\mathcal{S}^{2^l h}$ $(l = 1, \dots, L)$.

To achieve for both approaches an efficient or even multigrid-optimal algorithm for quadratic finite elements, we mimic ideas and techniques from previous discussions on linear finite elements. There, we introduced local functional matrices, which led to an efficient way of handling modifications to coarse-grid corrections. These local functional matrices can be computed for quadratic finite elements in a similar way. The key observation, which led to multigrid-optimality, is to recognize the need to use $V(0,\nu)$-cycles instead of $V(\mu,\nu)$-cycles for our PML method.

For the first approach, coarse-grid functional matrices are obtained as in the case for linear finite elements by adding up the respective fine-grid functional matrices. At the end of each cycle, we update the current approximation of the solution, $\mathbf{x}_Q^h$, by $\mathbf{c}_Q^h$, compute the new local functional matrices, and repeat the cycle. For the second approach, we first alter the triangulation from quadratic Lagrange triangles to linear Lagrange triangles. Then we apply a standard $V(0,\nu)$-cycle that involves relaxation on corrections represented by continuous piecewise-linear functions. At the end of this $V(0,\nu)$-cycle, we project the current (piecewise-linear) correction onto the original triangulation with quadratic Lagrange triangles. We continue the cycle by performing ν further relaxation sweeps on the correction, now represented by continuous piecewise-quadratic functions, by updating the current approximation of the solution, $\mathbf{x}_Q^h$, by $\mathbf{c}_Q^h$, and by computing the new local functional matrices to repeat the cycle.

Compared to the first approach, the second has two advantages. First, it allows reuse of most of the code for linear finite elements. Second, the coarsening process is independent of the order of the fine-grid discretization. This property becomes increasingly important as we choose increasingly higher-order discretizations on the finest level. To illustrate this, recall that our PML method treats the nonlinearity directly, without any kind of linearization process. Thus, our local functional matrices are growing rapidly in complexity for higher-order discretizations. (The complexity grows for quasilinear problems even more than for linear ones.) With the second approach, we use a multilevel strategy to compute a piecewise-linear approximation to the fine-grid correction. Having this approximation on the finest level available, we have the advantage of no longer being constrained to use local functional matrices or the same relaxation process as on coarser levels. In principle, we could consider the fine-level correction as a separate minimization process, with the advantage of having a good coarse-grid corrected initial guess. This is very appealing in particular for high-order finite elements.

However, low-order spaces do not always provide an effective coarse-level correction for high-order spaces in the same elements. An alternative is to partition the elements defining the high-order discretization into several smaller elements that could then be used to define the linear correction space (cf. [17, 20]). In our context of standard quadratic finite elements, we could split each quadratic Lagrange triangle into four linear Lagrange triangles. Although, this would violate our assumption of nested finite-dimensional subspaces, we would still obtain a good low-order approximation to the (high-order) correction on the finest level. This is very appealing for high-order element types, in particular, since this allows us, again, to consider the (high-order) fine-level problem as a separate minimization process.

5. Numerical Results.

Here we report on numerical results for some test problems. First, for verification, we compare PML convergence factors for different types of relaxation, discretization, and cycling strategies applied to (linear) Poisson problems. We then study performance on a set of nonlinear test problems by adding a simple nonlinear term to the Laplace operator, using a coefficient, α, that allows us to adjust the strength of nonlinearity. We finish this section by presenting numerical results for our target application, the incompressible Navier-Stokes equations, with particular focus on the so-called Kovasznay flow.

5.1. Measuring Convergence Factors.

Before we provide numerical results on some test problems, we first address the issue of how to measure convergence factors for our method since they play an especially important role in analyzing and evaluating a multigrid iteration.

Let $\mathcal{F}(\mathbf{x}^{2^l h}; \mathsf{g})$ be the discrete nonlinear functional for a nonlinear PDE written in first-order system least-squares form, and let $\mathbf{x}_*^{2^l h}$ be the minimizer of $\mathcal{F}(\mathbf{x}^{2^l h}; \mathsf{g})$ on level l. Superscript $2^l h$ does not play an essential role here, but we use it anyway to emphasize that the operator stems from a discretization on a certain level, l. Taking our cue from the linear case, we write the functional norm defect of our current approximation as

$$\hat{\delta}_k^{2^l h} = \sqrt{\mathcal{F}(\mathbf{x}_k^{2^l h}; \mathsf{g}) - \mathcal{F}(\mathbf{x}_*^{2^l h}; \mathsf{g})}, \tag{5.1}$$

where $\mathbf{x}_k^{2^l h}$ is the approximation to the exact solution, $\mathbf{x}_*^{2^l h}$, after the k-th iteration step. Note that $\hat{\delta}_k^{2^l h}$ is a positive real number because $\mathbf{x}_*^{2^l h}$ is the minimizer of $\mathcal{F}(\mathbf{x}^{2^l h}; \mathsf{g})$. In analogy to computing convergence factors for linear systems (cf. [9, 27]), we define the convergence factor for the k-th iteration step on level l by

$$\widehat{CF}_k^{(l)} := \frac{\hat{\delta}_k^{2^l h}}{\hat{\delta}_{k-1}^{2^l h}} = \sqrt{\frac{\mathcal{F}(\mathbf{x}_k^{2^l h}; \mathsf{g}) - \mathcal{F}(\mathbf{x}_*^{2^l h}; \mathsf{g})}{\mathcal{F}(\mathbf{x}_{k-1}^{2^l h}; \mathsf{g}) - \mathcal{F}(\mathbf{x}_*^{2^l h}; \mathsf{g})}}. \tag{5.2}$$

Since $\mathcal{F}(\mathbf{x}_*^{2^l h}; \mathsf{g})$ is unknown, equation (5.2) cannot be used directly to compute the convergence factor. Thus, instead of considering the defect, $\hat{\delta}_k^{2^l h}$, as in (5.1), we take the approach of defining the defect of two consecutive approximations:

$$\delta_k^{2^l h} = \sqrt{\mathcal{F}(\mathbf{x}_{k-1}^{2^l h}; \mathsf{g}) - \mathcal{F}(\mathbf{x}_k^{2^l h}; \mathsf{g})}. \tag{5.3}$$

The attendant convergence factor estimate is then given by

$$CF_k^{(l)} := \frac{\delta_k^{2^l h}}{\delta_{k-1}^{2^l h}} = \sqrt{\frac{\mathcal{F}(\mathbf{x}_{k-1}^{2^l h}; \mathrm{g}) - \mathcal{F}(\mathbf{x}_k^{2^l h}; \mathrm{g})}{\mathcal{F}(\mathbf{x}_{k-2}^{2^l h}; \mathrm{g}) - \mathcal{F}(\mathbf{x}_{k-1}^{2^l h}; \mathrm{g})}}. \tag{5.4}$$

Note, this measures requires care with respect to machine precision and numerical cancellation. For example, if $\mathcal{F}(\mathbf{x}_{k-1}^{2^l h}; \mathrm{g})$ and $\mathcal{F}(\mathbf{x}_k^{2^l h}; \mathrm{g})$ in (5.4) are the same up to near machine precision, then convergence factors can give the impression of degenerating performance.

5.2. Verifying Uniformly-Bounded Linear Convergence for a Poisson Problem. We begin our numerical tests on Poisson problems to confirm that we get optimal standard multigrid performance and optimal finite element approximation properties using linear finite elements. The Poisson problem with pure Dirichlet boundary conditions on $\Omega = [0,1] \times [0,1]$ is given by

$$\begin{aligned} -\Delta p &= f_{\Omega} && \text{in } \Omega, \\ p &= f_{\Gamma} && \text{on } \Gamma_{\Omega}. \end{aligned} \tag{5.5}$$

A first-order system least-squares (FOSLS) formulation for (5.5) is given by

$$\begin{aligned} \nabla p - \mathbf{u} =&\ \mathbf{0} && \text{in } \Omega, \\ -\nabla \cdot \mathbf{u} =&\ f_{\Omega} && \text{in } \Omega, \\ \nabla \times \mathbf{u} =&\ 0 && \text{in } \Omega, \\ p =&\ f_{\Gamma} && \text{on } \Gamma_{\Omega}, \\ \mathbf{n} \times \mathbf{u} =&\ \mathbf{n} \times \nabla f_{\Gamma} && \text{on } \Gamma_{\Omega}, \end{aligned} \tag{5.6}$$

where $\mathbf{n}$ is the unit outward normal on the boundary, Γ_{Ω}. For further details on this formulation, see [10] and [11]. For all of our tests, we use Dirichlet boundary conditions, strongly enforced by imposing them on the finite element space. We construct the first-order system least-squares functional by taking the L^2-norm of each interior equation:

$$\mathcal{F}(p, \mathbf{u}; \mathrm{g}) = \|p - \nabla \mathbf{u}\|_{0,\Omega}^2 + \|\nabla \cdot \mathbf{u} + f_{\Omega}\|_{0,\Omega}^2 + \|\nabla \times \mathbf{u}\|_{0,\Omega}^2, \tag{5.7}$$

where $\mathrm{g} = (\mathbf{0}, f_{\Omega}, 0)^t$ is the combined right side of the FOSLS formulation.

We start by examining asymptotic V-cycle convergence factors with a varying number of post-smoothing relaxation sweeps. To facilitate this test, we choose the homogeneous Laplace problem with zero boundary conditions ($f_{\Omega}(x,y) = 0$ and $f_{\Gamma}(x,y) = 0$ in (5.6)). The triangulation of the unit square consists of a regular grid with 2 113 nodes and 4 096 elements. For the discretization, we use standard piecewise-linear finite elements. Table 5.1 depicts numerical results for our PML scheme using different V$(0,\nu)$-cycles. We use either a Richardson-like or Gauß-Seidel-like relaxation method. The initial guess is chosen randomly and we iterate with 20 V-cycles to ensure sharp estimates of the asymptotic convergence factors, $CF_{20}^{(l)}$, which we measure according to (5.4). In addition, we report in Table 5.1 on the effective convergence factor, defined as the ν-th root of $CF_{20}^{(l)}$. Our Gauß-Seidel process uses a C/F ordering of the nodes, where we first sweep over all coarse-grid points, and then over the remaining fine-grid points.

$V(0,\nu)$-Cycle	Asymptotic convergence factor ($CF_{20}^{(l)}$)			
	Richardson-like Relaxation		Gauß-Seidel-like Relaxation	
	$CF_{20}^{(l)}$	effective CF	$CF_{20}^{(l)}$	effective CF
$\nu = 1$	0.696	0.696	0.418	0.418
$\nu = 2$	0.395	0.628	0.250	0.500
$\nu = 4$	0.288	0.732	0.133	0.603
$\nu = 6$	0.256	0.796	0.089	0.668
$\nu = 8$	0.245	0.838	0.067	0.713

TABLE 5.1
Comparison of asymptotic convergence factors for our nonlinear PML scheme using Richardson and Gauß-Seidel smoothers.

These results show typical multigrid convergence behavior. As expected, a Gauß-Seidel-like scheme performs better than Richardson-like relaxation. Also, we note that increasing the number of post-relaxation sweeps decreases the convergence factors, although with diminishing returns as is typical of multigrid solvers. This is reflected for both smoother types in an increase of the effective convergence factors. Although the effective convergence factors favor $V(0,2)$-cycles for a Richardson-like relaxation and $V(0,1)$-cycles for a Gauß-Seidel-like relaxation, they neglect the overhead of our PML method for computing new local functional matrices. In fact, from our numerical experiments, we observed the best results in terms of efficiency for $V(0,2)$ or $V(0,4)$-cycles, with a Gauß-Seidel-like smoother.

Next, we are interested in how our method performs for the Laplace problem with a nonzero right side. For the exact solution, we choose $p(x,y) = x^2 + y^2$. Since this solution cannot be represented exactly by our finite element space, the functional cannot converge to zero, but rather stagnates as the scheme reaches the level of discretization error on each grid. For piecewise linear finite elements with sufficiently smooth solution, we expect functional reduction by an asymptotic factor of about 2 as the resolution doubles. (This is consistent with finite element approximation theory.) For this test, we start with a regular triangulation (level $l = 6$) of 41 nodes and 64 elements. Each finer grid is obtained by dividing each triangle into four equal triangles. This leads to the finest level ($l = 0$) with 131 585 nodes and 262 144 elements. To step through the grid levels, we use a nested iteration approach, in the sense that we fix the number of V-cycles per level and use, as initial guess on each grid, the interpolated approximation from the previous (coarser) grid level (except for the coarsest grid, on which we use a random initial guess). On each level, we perform 5 $V(0,4)$-cycles with Gauß-Seidel as the smoother. Table 5.2 reports, for each level $l = 0, \ldots, 6$, the functional norm, $\mathcal{F}(p_5^{2^l h}, \mathbf{u}_5^{2^l h}; \mathrm{g})^{\frac{1}{2}}$, after 5 cycles, the functional reduction factor as the resolution doubles defined by

$$\beta_n^{(l)} = \sqrt{\frac{\mathcal{F}(p_n^{2^{l+1}h}, \mathbf{u}_n^{2^{l+1}h}; \mathrm{g})}{\mathcal{F}(p_n^{2^l h}, \mathbf{u}_n^{2^l h}; \mathrm{g})}}, \qquad l = 0, \ldots, L-1, \tag{5.8}$$

and the convergence factor, $CF_5^{(l)}$, defined as in (5.4). The purpose of this test is to give numerical evidence that we recover uniform convergence factors and optimal finite element approximation properties.

The last column in Table 5.2 show that the convergence factors are approximately the same on all levels. This property of approximate grid-independent convergence

Level l	Nodes/Elements	Functional norm $\mathcal{F}(p_5^{2^l h}, \mathbf{u}_5^{2^l h}; \mathbf{g})^{\frac{1}{2}}$	Functional reduction factor $\beta_5^{(l)}$	$CF_5^{(l)}$
6	41/64	1.416285e-01		0.054
5	145/256	7.174930e-02	1.975	0.091
4	545/1 024	3.602167e-02	1.992	0.108
3	2 113/4 096	1.803306e-02	1.998	0.116
2	8 321/16 384	9.019794e-03	2.000	0.117
1	33 025/65 536	4.510366e-03	2.000	0.096
0	131 585/262 144	2.255249e-03	2.000	0.098

TABLE 5.2

Convergence history for Poisson problem (5.6) *with* $f_\Omega(x, y) = -4$*, linear finite elements, and a standard* $V(0, 4)$*-cycle.*

is one key characteristics of multigrid that we aim to achieve. Note that the grid used to generate Table 5.1 is identical to that used for level $l = 3$ in Table 5.2. On this level, we see similar convergence factors, with a slight difference (0.133 versus 0.116) due to earlier termination here of the sequence of V-cycles. We also seem to have achieved optimal finite element approximation properties. On each level, the functional norm stagnates at the level of discretization error. The fact that we reached the level of discretization error is also supported by the functional reduction factors. For continuous piecewise-linear finite elements for our problem, standard theory (cf. [12]) establishes asymptotic $O(h)$ H^1 error bounds, so H^1 ellipticity of our functional yields an $O(h)$ functional-norm bound. We might, thus, expect about a factor of 2 in functional-norm reduction from one level to the next finer one. The numerically computed functional reduction factors, $\beta_5^{(l)}$, are reported in column 4 of Table 5.2 and coincide with the theoretical results. For this test problem, we reach the level of discretization error most of the time after 2 or 3 V-cycles. By performing 5 V-cycles, we ensure that the measured factors, $CF_5^{(l)}$, give us a better approximation to the asymptotic convergence factors.

5.3. A Nonlinear Model Problem. Section 5.2 shows typical numerical behavior of a multilevel algorithm applied to a Poisson problem. The next step is to test the new algorithm in the presence of nonlinearities. We, thus, modify the PDE given in (5.5) by adding a nonlinear term, pp_x. Additionally, we introduce parameter α to allow variation of the strength of the nonlinearity. This model represents a simple nonlinear PDE with a type of nonlinearity that is at the focus of this research. Its first-order system least-squares formulation is given as follows:

$$\begin{aligned} \nabla p - \mathbf{u} &= \mathbf{0} && \text{in } \Omega, \\ -\frac{1}{\alpha}\nabla\cdot \mathbf{u} + pu_1 &= f_\Omega && \text{in } \Omega, \\ \frac{1}{\alpha}\nabla\times \mathbf{u} &= 0 && \text{in } \Omega, \\ p &= f_\Gamma && \text{on } \Gamma_\Omega, \\ \mathbf{n}\times\mathbf{u} &= \mathbf{n}\times f_\Gamma && \text{on } \Gamma_\Omega. \end{aligned} \tag{5.9}$$

where $\nabla p = (u_1, u_2)^t$, $\mathbf{n}$ is the unit outward normal on boundary Γ_Ω, and Ω is the unit square. We choose $p(x, y) = x^2 + y^2$ as the exact solution and thus obtain $f_\Omega = -4/\alpha + (2x^3 + 2xy^2)$ as the right side. For all of our experiments, we use Dirichlet boundary conditions derived from the exact solution. We enforce the boundary conditions strongly by imposing them on the finite element space. Note that (5.9) arises from a more favorable scaling of the first-order system derived from the PDE, $\Delta p + \alpha\, pp_x = \tilde{f}_\Omega$. Hence, parameter α allows us to vary the strength of nonlinear term pp_x. Its first-order system least-squares functional is constructed by taking the L^2-norm of each interior equation:

$$\mathcal{F}(p, \mathbf{u}; \mathrm{g}) = \|\nabla p - \mathbf{u}\|^2_{0,\Omega} + \|-\frac{1}{\alpha}\nabla\cdot\mathbf{u} + pu_1 + \frac{4}{\alpha} - (2x^3 + 2xy^2)\|^2_{0,\Omega} + \|\frac{1}{\alpha}\nabla\times\mathbf{u}\|^2_{0,\Omega}, \quad (5.10)$$

where $\mathrm{g} = (\mathbf{0}, f_\Omega, 0)$. The grids are based on a regular triangulation of Ω by 16 elements and 13 grid points. This coarsest level is denoted by $l = 7$, with an approximate mesh size $2^l h$, where h is the approximate mesh size with respect to the finest level. Level 6 is formed by taking every element of level 7 and subdividing it into 4 equal triangles. The midpoint of the coarse-grid element sides are the new fine-grid points. Successively finer levels are constructed in the same way. This refinement leads to 131 585 nodes (with 3 degrees of freedom per node) and 262 144 elements on level 0. A nested iteration algorithm with 10 $V(0,4)$-Gauß-Seidel relaxation sweeps on each level is used to minimize $\mathcal{F}(p, \mathbf{u}; \mathrm{g})$ in (5.10) over the space consisting of continuous piecewise-linear functions. Table 5.3 depicts the functional norms, $\mathcal{F}(p_{10}^{2^l h}, \mathbf{u}_{10}^{2^l h}; \mathrm{g})^{\frac{1}{2}}$, obtained on each level for the linear Poisson problem and α varying between 1 and 10 000. Table 5.4 reports on the corresponding final convergence factors, $CF_{10}^{(l)}$, computed according to (5.4). Here, we choose again to report the convergence factor of the last iteration, since it tends to be the worst in our numerical tests.

Level	Linear Poisson	Nonlinearity parameter α 1	10	100	1 000	10 000
	Functional norm	Functional norm	Functional norm	Functional norm	Functional norm	Functional norm
7	2.7635e-01	2.7635e-01	2.6843e-01	2.5509e-01	2.5357e-01	2.5342e-01
6	1.4162e-01	1.4207e-01	1.4046e-01	1.3540e-01	1.3460e-01	1.3452e-01
5	7.1749e-02	7.1800e-02	7.1545e-02	7.0292e-02	7.0032e-02	7.0007e-02
4	3.6021e-02	3.6027e-02	3.5992e-02	3.5762e-02	3.5714e-02	3.5710e-02
3	1.8033e-02	1.8033e-02	1.8029e-02	1.8007e-02	1.8032e-02	1.8036e-02
2	9.0197e-03	9.0198e-03	9.0193e-03	9.0259e-03	9.0756e-03	9.0822e-03
1	4.5103e-03	4.5103e-03	4.5103e-03	4.5160e-03	4.5750e-03	4.5833e-03
0	2.2552e-03	2.2552e-03	2.2552e-03	2.2583e-03	2.3232e-03	2.3331e-03

TABLE 5.3

Measured functional norm (5.10)*,* $\mathcal{F}(p_{10}^{2^l h}, \mathbf{u}_{10}^{2^l h}; \mathrm{g})^{\frac{1}{2}}$*, for different* α *using a linear finite element discretization and 10* $V(0,4)$*-cycles with Gauss-Seidel as smoother.*

Note that, on each level, we obtain accuracy close to discretization level within 10 $V(0,4)$-cycles. We have not used any special technique (e.g., streamline relaxation) to address the changing character of the operator as α increases. Thus, as expected, the final convergence factors degrade as the nonlinearity increases in dominance, but they remain grid-independent. Though one might argue that the convergence factors in the last column of Table 5.4 ($\alpha = 10\,000$) do not exhibit grid-independent convergence factors, it is believed that grid-independent convergence factors are obtained once a sufficiently small mesh size is reached.

Table 5.5 depicts the functional reduction factors, $\beta_{10}^{(l)}$, as defined in (5.8) for different levels and strengths of nonlinearity. For all levels and strengths of nonlinearity,

Level	Linear Poisson	Nonlinearity parameter α				
		1	10	100	1 000	10 000
	$CF_{10}^{(l)}$	$CF_{10}^{(l)}$	$CF_{10}^{(l)}$	$CF_{10}^{(l)}$	$CF_{10}^{(l)}$	$CF_{10}^{(l)}$
7	0.031	0.029	0.128	0.258	0.272	0.274
6	0.054	0.063	0.318	0.602	0.632	0.635
5	0.091	0.068	0.443	0.776	0.809	0.812
4	0.108	0.092	0.538	0.825	0.865	0.866
3	0.116	0.098	0.581	0.872	0.891	0.893
2	0.117	0.097	0.595	0.893	0.926	0.928
1	0.117	0.090	0.599	0.908	0.944	0.946
0	0.108	0.096	0.599	0.918	0.955	0.957

TABLE 5.4
Convergence factors, $CF_{10}^{(l)}$, for the same experiments as in Table 5.3

	Linear	Nonlinearity parameter α				
	Poisson	1	10	100	1 000	10 000
Level	$\beta_{10}^{(l)}$	$\beta_{10}^{(l)}$	$\beta_{10}^{(l)}$	$\beta_{10}^{(l)}$	$\beta_{10}^{(l)}$	$\beta_{10}^{(l)}$
7	—	—	—	—	—	—
6	1.95	1.95	1.91	1.88	1.88	1.88
5	1.97	1.98	1.96	1.93	1.92	1.93
4	1.99	1.99	1.99	1.97	1.96	1.96
3	2.00	2.00	2.00	1.99	1.98	1.98
2	2.00	2.00	2.00	2.00	1.99	1.99
1	2.00	2.00	2.00	2.00	1.98	1.98
0	2.00	2.00	2.00	2.00	1.97	1.97

TABLE 5.5
Functional reduction factors, $\beta_{10}^{(l)}$, based on functional norms reported in Table 5.3

we observe functional reduction factors of about 2, which is consistent with the use of continuous piecewise-linear finite elements.

Next, we analyze error reduction factors as we step through the different levels. Consider again the same FOSLS formulation, levels, and number of V-cycles per level used for the results in Table 5.3. We now compare the numerically obtained solution, $p^{2^l h}$, with the exact solution, $p = x^2 + y^2$, for each level and for each α ($\alpha = 1$, 10, 100, 1 000, and 10 000), measured by the H^1 and L^2 norms. In Figure 5.1, we depict the H^1-error norm versus the number of elements. The L^2-error norm versus the number of elements is shown in Figure 5.2. Since we use a regular refinement strategy to step through the levels (with each refinement, we increasing the number of elements by a factor of 4 and, therefore, halve our mesh size), reporting on the number of elements is the same as reporting on the mesh size. For both figures, we use a logarithmic scale for the number of elements (abscissa) and the error-norm (ordinate). For each α, the H^1-error norm (or L^2-error norm) is measured for each level and indicated with marks, which are connected in such a way that each line displays one nested iteration process for some α. Additionally, we include in Figure 5.1 a supporting line with slope 1 and in Figure 5.2 two supporting lines with slopes 1 and 2. These supporting lines should help retrieving an estimate of the error-reduction factors directly out of the graph. Note that a slope of $\mathbf{s}$ in Figures 5.1

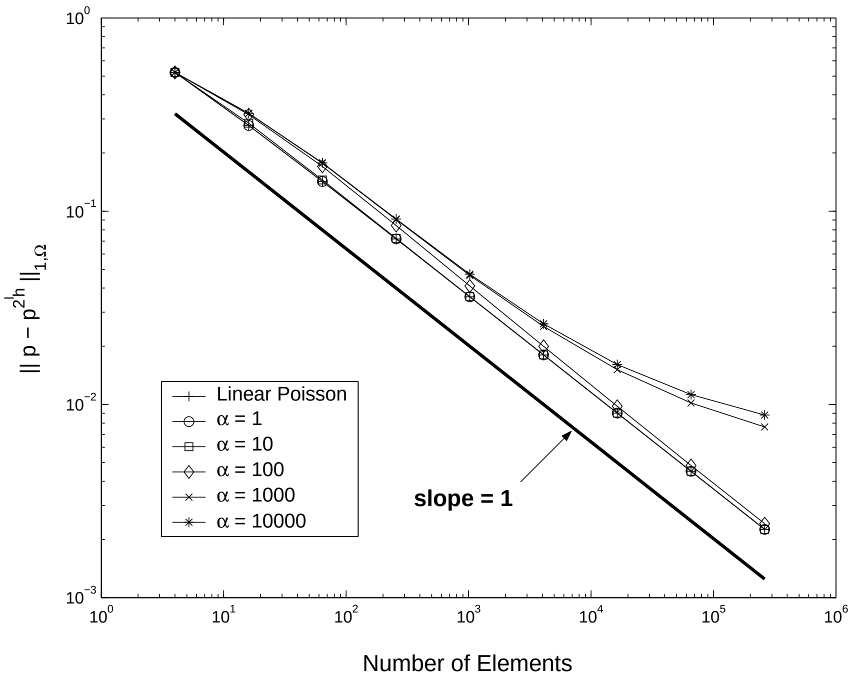

FIG. 5.1. *H^1-error, $\|p-p^{2^l h}\|_{1,\Omega}$, versus the number of elements for the linear Poisson problem and* (5.9) *with $\alpha =$ 1, 10, 100, 1 000, and 1 0000.*

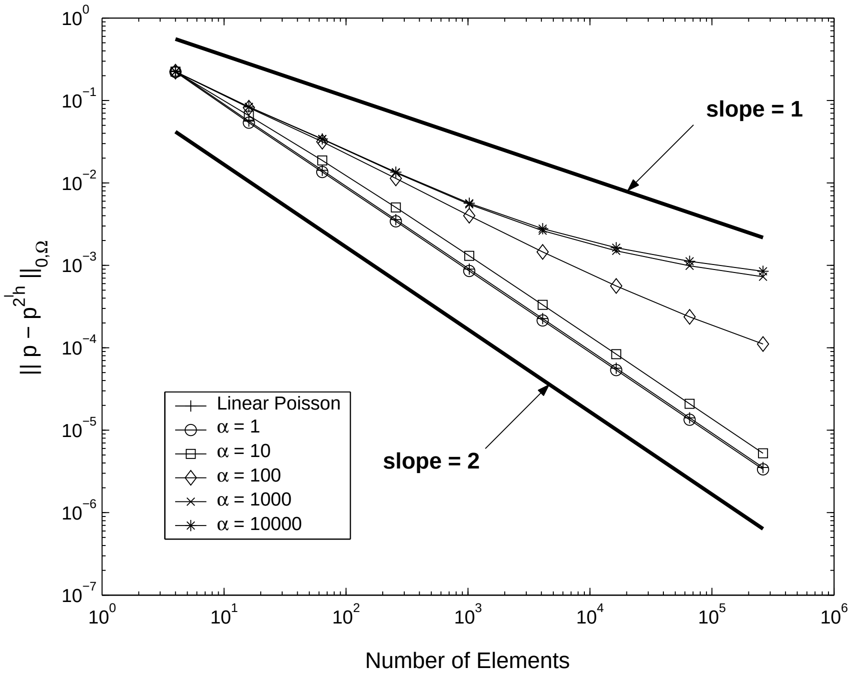

FIG. 5.2. *L^2-error, $\|p-p^{2^l h}\|_{0,\Omega}$, versus the number of elements for the linear Poisson problem and* (5.9) *with $\alpha =$ 1, 10, 100, 1 000, and 10 000.*

and 5.2 means that $\frac{\|p-p^{2^l h}\|}{\|p-p^{2^{l+1} h}\|} \approx \left(\frac{2^{l+1}h}{2^l h}\right)^{\mathbf{s}} = 2^{\mathbf{s}}$. Hence, slope s translates to an error-reduction factor of 2^s. Analyzing Figure 5.1, the error-reduction factor from one level to the next is about 2 for every α. This coincides well with the reported functional reduction factors, $\beta_{10}^{(l)}$, in Table 5.5, and are considered to be optimal for linear finite elements. From the excellent agreement of the FOSLS functional norm and the H^1-error reduction factors, we conclude that the functional in (5.10) appears to be H^1-elliptic. This numerical observation coincides with the theoretical results of the companion paper [22], where we establish H^1-ellipticity of the FOSLS functional based on the Navier-Stokes equations and anticipate it for other quasilinear PDEs of that class.

In Figure 5.2, we display the L^2-error norms in the same way as the H^1-error in Figure 5.1. We now observe strongly deteriorating L^2-error reduction factors with increasing strength of nonlinearity. One possible explanation for this might involve the Nitsche Trick (cf. [8]), which relates two different error norms to each other (in this case, the H^1-error norm and the L^2-error norm). Its proof is based on the assumption that the exact solution, $\mathbf{x}_*^{2^l h}$ is found on each level. With a nested iteration scheme, we compute on each level only an approximation to $\mathbf{x}_*^{2^l h}$; here, for example, we approximate $\mathbf{x}_*^{2^l h}$ by $\mathbf{x}_{10}^{2^l h}$. In separate experiments, we have been able to recover near-optimal L^2-error reduction factors by using 100 V-cycles instead of 10 on each level. This shows that better algebraic accuracy is needed on each level to control the L^2-error. This should be expected since greater L^2 accuracy is obtained from the discretization on each level, so nested iteration should have to work harder than for H^1 accuracy to achieve it. Development of effective criteria for a nested iteration strategy that efficiently produces small H^1 and L^2 errors is still an open question.

5.4. Kovaszany Flow. While system (5.9) provides an important problem to test the behavior of the algorithm, our ultimate goal is to solve the Navier-Stokes equations. For concreteness, we focus on the steady-state incompressible Navier-Stokes equations in velocity-pressure formulation given as follows:

$$\begin{aligned} -\frac{1}{Re}\Delta\mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} + \nabla p = \mathbf{0} \qquad & \text{in } \Omega, \\ \nabla\cdot\mathbf{u} = 0 \qquad & \text{in } \Omega. \end{aligned} \tag{5.11}$$

Velocity vector variable $\mathbf{u} = (u_1, u_2)^t$ and pressure scalar variable p are non-dimensionalized. Re denotes the Reynolds number defined as $Re = (U_{ref}L)/\nu$, where L is a reference length, U_{ref} a reference velocity, and ν the kinematic viscosity (see [18]). Note that the source terms in this system are all zero. We could easily incorporate nonzero terms, but choose this simplification instead because our primary focus is on the algebraic solver and because inhomogeneities are incorporated in the boundary conditions in any case.

To obtain a first-order system from (5.11), we introduce a new velocity-flux tensor variable, $\mathbf{U} = (U_{i,j})_{2\times 2} = (\partial u_j/\partial x_i)_{2\times 2} = \nabla\mathbf{u}^t$. (See [1] for details on the FOSLSization of equations (5.11).) We thus obtain the following first-order velocity-flux form of the Navier-Stokes equations:

$$\begin{aligned} \nabla \mathbf{u}^t - \mathbf{U} &= \mathbf{0} && \text{in } \Omega, \\ -\frac{1}{Re}\,(\nabla\cdot \mathbf{U})^t + \mathbf{U}^t\mathbf{u} + \nabla p &= \mathbf{0} && \text{in } \Omega, \\ \nabla\cdot \mathbf{u} &= 0 && \text{in } \Omega, \\ \frac{2}{Re}\nabla\times \mathbf{U} &= 0 && \text{in } \Omega. \end{aligned} \tag{5.12}$$

The difference between this system and that proposed in [1] is the factor of 2 in the last equation and the missing trace term, $\nabla tr(\mathbf{U})$. The additional factor is a simple weighting of this equation that, by our empirical observations, results in slightly better numerical results. Concerning the trace term, because of the incompressibility condition expressed by $\partial_x u_1 + \partial_y u_2 = U_{11} + U_{22} = 0$, we are able to eliminate one of the variables by setting $U_{11} = -U_{22}$, which in turn enforces $\nabla tr(\mathbf{U}) = 0$ and therefore makes this trace equation unnecessary. Of course, system (5.12) offers but one approach to reducing the second-order problem to first order. Other choices are given, for example, in [3] and [18]. In any case, the solution of our first-order system is the minimizer of the least-squares functional given by

$$\begin{aligned} \mathcal{F}(\mathbf{u},\mathbf{U},p\,;\mathbf{g}) = \|\nabla \mathbf{u}^t - \mathbf{U}\|^2_{0,\Omega} + \|-\frac{1}{Re}\,(\nabla\cdot \mathbf{U})^t + \mathbf{U}^t\mathbf{u} + \nabla p\|^2_{0,\Omega} \\ + \|\nabla\cdot \mathbf{u}\|^2_{0,\Omega} + \|\frac{2}{Re}\nabla\times \mathbf{U}\|^2_{0,\Omega}, \end{aligned} \tag{5.13}$$

where $\mathbf{g} = (\mathbf{0},\mathbf{0},0)$ is the combined right side of the equations in (5.12).

As a model problem for our algorithm applied to the Navier-Stokes equations, we turn to Kovasznay flow. This particular system is named after L.I.G. Kovasznay, who derived in [19] an analytic solution for the steady-state incompressible Navier-Stokes equations for a special laminar flow problem. We choose this problem as a test case, since it is posed on a rectangular domain, $\Omega = [-.5, 2.0]\text{x}[-.5, 1.5]$, has a smooth solution, and exhibits no singularities. Knowledge of the analytical solution allows us to impose the exact boundary conditions strongly. Actually, for accurate error estimates, we need not appeal to an exact analytic solution, since the FOSLS functional itself provides naturally a sharp error measurement. But use of an exact solution gives a somewhat tighter estimate of any error measure we choose to use.

In Table 5.6, we give the convergence history using continuous piecewise-quadratic functions for the Kovasznay flow problem with a Reynolds number of 40. For the cycling strategy, we choose to use quadratic finite elements for the fine-grid corrections and linear finite element subspaces for the coarse-grid process (see the second approach of Section 4.3). The grids are based on a regular triangulation of Ω by 16 elements and 41 nodes. Again, we use a nested iteration approach to step through the levels. On each level, we apply 10 $V(0, 4)$-PML cycles, with Gauß-Seidel as smoother. For each level, we report on final functional norm values, $\mathcal{F}(\mathbf{x}_{10}^{2^l h};\mathbf{g})^{\frac{1}{2}}$, the functional reduction factor, $\beta_{10}^{(l)}$, defined as in (5.8), and the final convergence factor, $CF_{10}^{(l)}$, defined as in (5.4).

The results in Table 5.6 show that we also obtain approximate grid-independent convergence factors for the FOSLS formulation of the Navier-Stokes problem and nearly optimal finite element approximation properties. The fact that the functional reduction factor, $\beta_{10}^{(l)}$, is hovering around 3.7 instead of an optimal factor of 4 for

Level l	Nodes/Elements	Functional norm $\mathcal{F}(\mathbf{x}_{10}^{2^l h}; \mathbf{g})^{\frac{1}{2}}$	Functional reduction factor $\beta_{10}^{(l)}$	$CF_{10}^{(l)}$
5	41/16	4.212367e+00		0.713
4	145/64	1.635538e+00	2.57	0.788
3	545/256	4.854122e-01	3.37	0.854
2	2 113/1 024	1.379767e-01	3.51	0.880
1	8 321/4 096	3.760596e-02	3.67	0.892
0	33 025/16 384	9.993762e-03	3.78	0.897

Table 5.6

Convergence summary for Kovasznay flow with $Re = 40$, a nested-iteration PML approach with 10 $V(0, 4)$-cycles per level, Gauß-Seidel as smoother, and quadratic finite elements.

quadratic Lagrange finite elements is probably due to mostly the approximations not yet being in the asymptotic range. Note the increase in these factors with decreasing h. (Our tests that increased the number of V-cycles showed only marginal increase in the functional reduction factors.)

Though we report here only on results for $Re = 40$ (the classical setting for the Kovasznay flow), we have done experiments for much higher Reynolds numbers. We obtained similar results, although the convergence factors naturally degraded since the PML scheme was not designed for convection-dominated problems.

REFERENCES

[1] P. Bochev, Z. Cai, T. Manteuffel, and S. McCormick, *Analysis of velocity-flux first-order system least-squares principles for the Navier-Stokes equations: Part I*, SIAM J. Numer. Anal, 35 (1998), pp. 990–1009.

[2] ———, *Analysis of velocity-flux least-squares principles for the Navier-Stokes equations: Part II*, SIAM J. Numer. Anal., 36 (1999), pp. 1125–1144.

[3] P. B. Bochev, *Negative norm least-squares methods for the velocity-vorticity-pressure Navier-Stokes equations*, Numer. Methods Partial Differential Equations, 15 (1999), pp. 237–256.

[4] P. B. Bochev and M. D. Gunzburger, *Finite element methods of least-squares type*, SIAM Rev., 40 (1998), pp. 789–837 (electronic).

[5] D. Braess, *Finite elements*, Cambridge University Press, Cambridge, 2001.

[6] A. Brandt, *Multi-level adaptive solutions to boundary-value problems*, Math. Comp., 31 (1977), pp. 333–390.

[7] A. Brandt, S. McCormick, and J. Ruge, *Algebraic multigrid (AMG) for sparse matrix equations*, in Sparsity and its applications (Loughborough, 1983), Cambridge Univ. Press, Cambridge, 1985, pp. 257–284.

[8] S. C. Brenner and L. R. Scott, *The Mathematical Theory of Finite Element Methods*, vol. 15 of Texts in Applied Mathematics, Springer-Verlag, New York, 2002.

[9] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A multigrid tutorial*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, second ed., 2000.

[10] Z. Cai, R. Lazarov, T. A. Manteuffel, and S. F. McCormick, *First-order system least squares for second-order partial differential equations. I*, SIAM J. Numer. Anal., 31 (1994), pp. 1785–1799.

[11] Z. Cai, T. A. Manteuffel, and S. F. McCormick, *First-order system least squares for second-order partial differential equations. II*, SIAM J. Numer. Anal., 34 (1997), pp. 425–454.

[12] P. G. Ciarlet, *The Finite Element Method for Elliptic Problems*, vol. 40 of Classics in Applied Mathematics, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2002.

[13] G. Dardyk and I. Yavneh, *A Robust Nonlinear Multigrid Method.* Technion - Israel Institute of Technology, September 2001.

[14] J. E. Dennis, Jr. and R. B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*, vol. 16 of Classics in Applied Mathematics, Society for Industrial

and Applied Mathematics (SIAM), Philadelphia, PA, 1996. Corrected reprint of the 1983 original.

[15] E. Gelman and J. Mandel, *On multilevel iterative methods for optimization problems*, Math. Programming, 48 (1990), pp. 1–17.

[16] W. Hackbusch, *Multigrid methods and applications*, vol. 4 of Springer Series in Computational Mathematics, Springer-Verlag, Berlin, 1985.

[17] J. J. Heys, T. A. Manteuffel, S. F. McCormick, and L. N. Olson, *Algebraic Multigrid (AMG) for Higher-Order Finite Elements*, Journal of Computational Physics, to appear.

[18] B.-n. Jiang, *The least-squares finite element method*, Scientific Computation, Springer-Verlag, Berlin, 1998. Theory and applications in computational fluid dynamics and electromagnetics.

[19] L. I. G. Kovasznay, *Laminar flow behind two-dimensional grid*, Proc. Cambridge Philos. Soc., 44 (1948), pp. 58–62.

[20] J. Lottes and P. Fischer, *Hybrid Multigrid/Schwarz Algorithms for the Spectral Element Method*, SIAM J. Sci. Comput., to appear.

[21] J. Mandel and S. McCormick, *A multilevel variational method for* $A\mathbf{f}u = \lambda B\mathbf{f}u$ *on composite grids*, J. Comput. Phys., 80 (1989), pp. 442–452.

[22] T. A. Manteuffel, S. F. McCormick, and O. Röhrle, *Projection Multilevel Methods for Quasilinear Elliptic Partial Differential Equations: Theoretical Results*, manuscript.

[23] S. F. McCormick, *Multilevel projection methods for partial differential equations*, vol. 62 of CBMS-NSF Regional Conference Series in Applied Mathematics, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1992.

[24] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer Series in Operations Research, Springer-Verlag, New York, 1999.

[25] J. W. Ruge and K. Stüben, *Algebraic multigrid*, in Multigrid methods, vol. 3 of Frontiers Appl. Math., SIAM, Philadelphia, PA, 1987, pp. 73–130.

[26] K. Stüben and U. Trottenberg, *Multigrid methods: fundamental algorithms, model problem analysis and applications*, in Multigrid methods (Cologne, 1981), vol. 960 of Lecture Notes in Math., Springer, Berlin, 1982, pp. 1–176.

[27] U. Trottenberg, C. W. Oosterlee, and A. Schüller, *Multigrid*, Academic Press Inc., San Diego, CA, 2001.