SIAM J. SCI. COMPUT.
Vol. 16, No. 6, pp. 1308-1332, November 1995

MULTI-P METHODS: ITERATIVE ALGORITHMS FOR THE P-VERSION OF THE FINITE ELEMENT ANALYSIS*

NING HU† AND I. NORMAN KATZ†

Abstract. Motivated by classical multigrid methods, which are based on either the finite difference method or the h-version of the finite element method, we study multi-p methods, which are based on the p-version of the finite element method and hierarchical shape functions. We have developed a class of multi-p V-cycle algorithms: standard V-cycle, modified V-cycle and varying V-cycle. We have also studied nested multi-p methods. By combining the nested multi-p methods with the V-cycle methods, we obtain the so-called "accelerated multi-p V-cycle methods." The convergence of multi-p V-cycle methods have been investigated and an error estimate is given for the nested multi-p methods.

Numerical experiments on representative sample problems have been conducted and indicate that our multi-p methods are very efficient.

Key words. multi-p methods; p-version of the finite element method

AMS subject classifications. 65N22, 65N30, 65F10

1. Introduction. Modern engineering design relies heavily on computer simulations. In particular, in structural elastic analysis, it is necessary to solve (sometimes nonlinear) partial differential equations on complicated geometries. To solve such partial differential equations numerically on computers, discretization techniques are used. Such discretizations may involve thousands or hundreds of thousands of unknowns, and efficient numerical methods are indispensable. Two common discretization methods are used: the finite difference method and the finite element method. In general, there are two kinds of finite element methods: the h-version and the p-version. In the h-version, polynomial basis functions are fixed over each element of diameter h and accuracy is achieved by refining the mesh [24]. In the p-version, the mesh is fixed and accuracy is achieved by increasing the degree p of polynomial basis functions [1], [2], [9], [31]. The h-p version is the combination of both versions [3], [4], [6], [7], [8], [10], [11]. The p-version and h-p version of the finite element methods have been successfully applied to solve industrial problems on serial computers. Indeed, at least four corporations have adopted the p-version for commercial marketing, and at least one of them uses the p-version exclusively. Efficient iterative algorithms are now needed for more complicated cases that involve hundreds of thousands of degrees of freedom, which are becoming more common and more realistic.

Multigrid methods have been regarded as one of the most promising iterative methods for solving equations arising from the discretization of partial differential equations by either the finite difference method or the h-version of the finite element method [12], [15], [19], [27], [29]. For example, for the Poisson equation in a rectangular domain, the spectral radius of a multigrid V-cycle method is bounded away from one independent of mesh size h [13]. For general problems, the spectral radius of a multigrid V-cycle method is bounded by [14]

$$1 - \frac{c}{\left(\log \frac{1}{h}\right)^{(1-\alpha)/\alpha}},$$

*Received by the editors March 21, 1994; accepted for publication (in revised form) September 20, 1994. This research was supported by Air Force Office of Scientific Research under grant number AFOSR 92-J-0043. The results presented here are part of the doctoral dissertation of Ning Hu.

†Department of Systems Science and Mathematics, Washington University in St. Louis, St. Louis, Missouri 63130 (hu@badger.wustl.edu, katz@zach.wustl.edu).

where $0 < \alpha \leq 1$ is a regularity parameter associated with original partial differential equations.

Motivated by multigrid methods, some researchers applied such multigrid ideas to solve equations arising from the p-version of the finite element analysis [17]. In [5], Babuska et al. proposed the name "multi-p methods" and conducted numerical tests. Based on the simulation results, they indicated that the multi-p V-cycle methods converge faster than classical Gauss-Seidel and SOR methods but not as fast as multigrid methods. In [16], Brussino et al. tested a multi-p V-cycle algorithm on a parallel computer, and their numerical results confirmed the same observation as in [5]. Foresti et al. in [18] tested a multi-p V-cycle algorithm by using a block Gauss-Seidel scheme as the smoother. The numerical results indicated a better convergence rate, but the speed up ratio of the parallel implementation deviated significantly from the ideal. All these results are based on numerical studies; there is no theoretical analysis. As we point out in this paper, although the multigrid ideas can be adopted, the theoretical results cannot be applied directly to analyze multi-p methods.

In this paper, we have systematically investigated the multi-p methods. The goal of this paper is to develop efficient iterative numerical algorithms for solving equations arising from the p-version of the finite element analysis. We have developed a class of multi-p V-cycle methods including standard V-cycle, modified V-cycle, and varying V-cycle. We have also studied nested multi-p methods. The convergence of the multi-p methods have been investigated, and numerical simulations on representative sample problems have been conducted. By combining the nested multi-p algorithms with the V-cycle algorithms, we obtain the so-called "accelerated V-cycle algorithms"; numerical simulations indicate that the accelerated multi-p V-cycle methods have a much faster convergence rate than other multi-p algorithms. In our simulations, we have used two iterative schemes as smoothers. One is the classical Gauss-Seidel scheme. The other is a newly developed iterative scheme called the textured decomposition scheme [23], [20]. The Gauss-Seidel scheme has been successfully used in multigrid methods, and it is also very suitable as a smoother for the multi-p methods [21]. The reason we use a textured decomposition scheme as a smoother is that it is very suitable for parallel implementations when used recursively [23], [32]. The ultimate goal of this research is to develop efficient parallel multi-p methods for solving large-scale problems arising from the p-version of the finite element analysis.

The paper is organized as follows: in §2, we briefly discuss some of the properties of the p-version of the finite element method. In §3, we formulate the multi-p V-cycle algorithms and present some convergence results. We discuss the nested multi-p algorithms in §4 and derive an error estimate. The accelerated multi-p V-cycle algorithms are formulated in §5, and numerical simulation results are given in §6. Finally, a concluding remark is given in §7.

2. The p-version of the finite element method and hierarchical shape functions. In the p-version of the finite element method, hierarchical shape functions are used [31]. We discuss briefly here a set of hierarchical shape functions that are associated with quadrilateral elements. A set of hierarchical shape functions that are associated with triangular elements will not be discussed here but can be found in [31].

Let $K = (-1, 1) \times (-1, 1)$ be a standard quadrilateral element as shown in Fig. 1. There are four nodal shape functions associated with each vertex:

$$N_1(\zeta, \eta) = \frac{1}{4}(1 - \zeta)(1 - \eta),$$

$$N_2(\zeta, \eta) = \frac{1}{4}(1 + \zeta)(1 - \eta),$$

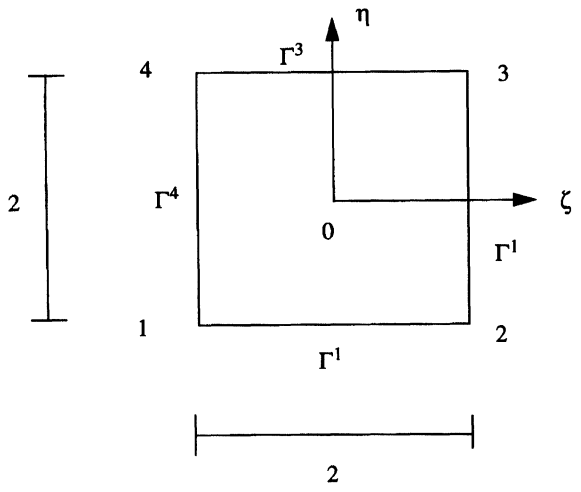

FIG. 1. *The standard quadrilateral element.*

$$N_3(\zeta,\eta)=\frac{1}{4}(1+\zeta)(1+\eta),$$

$$N_4(\zeta,\eta)=\frac{1}{4}(1-\zeta)(1+\eta).$$

For $p\geq 2$, there are $p-1$ shape functions associated with every side:

$$N_i^{[1]}(\zeta,\eta)=\frac{1}{2}(1-\eta)\Phi_i(\zeta),$$

$$N_i^{[2]}(\zeta,\eta)=\frac{1}{2}(1+\zeta)\Phi_i(\eta),$$

$$N_i^{[3]}(\zeta,\eta)=\frac{(-1)^i}{2}(1+\eta)\phi_i(\zeta),$$

$$N_i^{[4]}(\zeta,\eta)=\frac{(-1)^i}{2}(1-\zeta)\phi_i(\eta),\qquad i=2,3,\ldots,p;$$

where

$$\phi_j(\zeta)=\sqrt{\frac{2i-1}{2}}\int_{-1}^{\zeta}P_{j-1}(t)\,dt$$

and $P_{j-1}(t)$ is the Legendre polynomial of degree $j-1$, $j=2,3,\ldots,p$.

For $p\geq 4$, there are $(p-2)(p-3)/2$ internal shape functions:

$$N_{i,j}^{[0]}(\zeta,\eta)=(1-\zeta^2)(1-\eta^2)P_i(\zeta)P_j(\eta),\qquad 0\leq i+j\leq p-4.$$

This set of shape functions is hierarchic; that is, the finite element space V_{p-1} that is spanned by the above polynomial shape functions with degree up to $p-1$ is completely embedded into the space V_p that is spanned by the above shape functions with degree up to p. Based on this set of hierarchical shape functions, the stiffness matrices and load vectors also have hierarchical structures. For example, let A_p and b_p be the assembled global stiffness matrix

and load vector. If A_p and b_p correspond to degree $\leq p$, then we can write $A_p x = b_p$ in block form as

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{12}^{\mathrm{T}} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, \tag{2.1}$$

where A_{11} and b_1 correspond to degree $\leq p-1$, i.e., $A_{p-1} = A_{11}$ and $b_{p-1} = b_{11}$.

Let $\Omega \subset R^2$ be a problem domain, which is bounded by piecewise smooth boundaries. For any integer $k \geq 0$, let $H^k(\Omega)$ (or $H_0^k(\Omega)$) be the standard Sobolev space. For a noninteger $k > 0$, let the space $H^k(\Omega)$ (or $H_0^k(\Omega)$) be defined by usual interpolation procedures.

Consider the second-order elliptic partial differential equation

$$-\nabla(a\nabla u) + bu = g \quad \text{in } \Omega,$$

$$u = 0 \quad \text{on } \Gamma^D,$$

$$\frac{\partial u}{\partial n} = h \quad \text{on } \Gamma^N,$$

where Ω is a bounded domain with piecewise smooth boundary $\partial\Omega$ and $\Gamma^D \cup \Gamma^N = \partial\Omega$; $a \in C^1(\bar{\Omega})$; $b \in C(\Omega)$; $a(x) > \alpha > 0$; $b(x) \geq 0$; and $g \in L^2(\Omega)$.

Define

$$H(\Omega) = \{u | u \in H^1(\Omega),\ u = 0 \text{ on } \Gamma^D\}.$$

The exact solution. Find $u \in H(\Omega)$ such that

$$a(u, v) = f(v)$$

for all $v \in H(\Omega)$, where

$$a(u, v) = \int_\Omega (a\nabla u \cdot \nabla v + buv)\, dx,$$

$$f(v) = \int_\Omega gv\, dx + \int_{\Gamma^N} hv\, ds.$$

The exact solution exists and is unique under common assumptions for g, h, and $\partial\Omega$. Let $V_p \subset H(\Omega)$ be the p-version of the finite element space based on the hierarchical shape functions $\{\varphi_i\}$. Then we have

$$V_1 \subset V_2 \subset \cdots \subset V_{p-1} \subset V_p \subset H(\Omega).$$

The finite element solution in the *p*-version. Find a function $u \in V_p \subset H(\Omega)$ such that

$$a(u, v) = f(v) \quad \forall v \in V_p.$$

Let U_p be the corresponding coefficient space with Euclidean inner product $(\ ,\ )_p$; then $U_p = R^{N_p}$, where $N_p = \dim(U_p)$. For each $u \in V_p$, $\exists (x_1, \ldots, x_{N_p}) \in U_p$, such that

$$u = \sum_{i=1}^{N_p} x_i \varphi_i.$$

Define

$$\mathcal{A}_p : U_p \to V_p \quad \text{as } (x_1, \ldots, x_{N_p}) \to u, \qquad p = 1, 2, \ldots.$$

Then $\mathcal{A}_p$ is one-to-one and onto.

The prolongation (interpolation) mapping $I_{p-1}^p : U_{p-1} \to U_p$ (from level $p-1$ to p) is defined by the injection mapping

$$I_{p-1}^p = \mathcal{A}_p^{-1} \cdot \mathcal{A}_{p-1}. \tag{2.2}$$

The projection (restriction) mapping $I_p^{p-1} : U_p \to U_{p-1}$ (from level p to $p-1$) is defined as

$$(I_{p-1}^p x_{p-1}, y_p)_p = (x_{p-1}, I_p^{p-1} y_p)_{p-1} \tag{2.3}$$

for all $x_{p-1} \in U_{p-1}$, $y_p \in U_p$, i.e.,

$$I_p^{p-1} = (I_{p-1}^p)^{\mathrm{T}}. \tag{2.4}$$

Based on the p-version and the hierarchical shape functions, the defined prolongation and projection mappings can be expressed in matrix form as

$$I_{p-1}^p = \begin{bmatrix} I_{p-1} \\ 0 \end{bmatrix}, \qquad I_p^{p-1} = [I_{p-1} \;\; 0], \tag{2.5}$$

where I_{p-1} is the $N_{p-1} \times N_{p-1}$ identity matrix.

In comparison with multigrid methods, the prolongation and projection here are very simple and independent of domain geometry. Also notice that, by using (2.1), it is easy to verify that the prolongation and projection satisfy the relation

$$A_{p-1} = I_p^{p-1} A_p I_{p-1}^p. \tag{2.6}$$

Figure 2 shows the relationship among all the aforementioned mappings.

Now, the question we want to ask is: Is there any other prolongation and projection that can be used? To answer this question, we need to note that the purpose of using prolongation and projection is to move data and computations from one level to another level. Based on the p-version and hierarchical shape functions, the stiffness matrices and load vectors also have the hierarchical property. This property provides us with all the lower level stiffness matrices and load vectors if we already have obtained the stiffness matrix and load vector at the highest level. This saves computer storage and computations. In order to make use of this property, a prolongation and projection have to satisfy condition (2.6). Based on such considerations, we can answer the question in the following proposition. Let S_{p-1}^p and S_p^{p-1} be a pair of prolongation and projection.

PROPOSITION 2.1. *There are at least two pairs of prolongations and projections* (S_{p-1}^p, S_p^{p-1}) *satisfying*

$$A_{p-1} = S_p^{p-1} A_p S_{p-1}^p,$$

where A_p *and* A_{p-1} *are stiffness matrices corresponding to the shape functions with* degree $\leq p$ *and* degree $\leq p-1$, *respectively.*

Proof. For simplicity, let

$$S_{p-1}^p = \begin{pmatrix} S_1 \\ S_2 \end{pmatrix}, \qquad S_p^{p-1} = (S_{p-1}^p)^{\mathrm{T}};$$

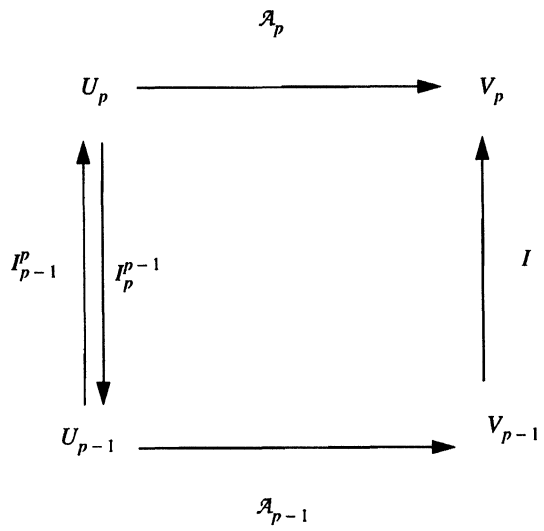

FIG. 2. *Relationship between prolongation and projection.*

then,

$$S_p^{p-1} A_p S_{p-1}^p = \begin{bmatrix} S_1^T & S_2^T \end{bmatrix} \begin{bmatrix} A_{11} & A_{12} \\ A_{12}^T & A_{22} \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \end{bmatrix}$$
$$= S_1^T A_{11} S_1 + S_1^T A_{12} S_2 + S_2^T A_{12}^T S_1 + S_2^T A_{22} S_2.$$

Note that $A_{p-1} = A_{11}$. We already know that by choosing

$$S_1 = I_{p-1}, \qquad S_2 = 0$$

we obtain the standard prolongation and projection I_{p-1}^p and I_p^{p-1}, which satisfy (2.6). Therefore, we only need to find another pair of prolongation and projection that satisfy (2.6). If we choose

$$S_1 = I_{p-1}, \qquad S_2 = -2A_{22}^{-1} A_{12}^T, \tag{2.7}$$

then

$$S_{p-1}^p = \begin{bmatrix} I_{p-1} \\ -2A_{22}^{-1} A_{12}^T \end{bmatrix}, \qquad S_p^{p-1} = [I_{p-1} \ -2A_{12} A_{22}^{-1}] \tag{2.8}$$

and it is easy to check that the pair (2.8) satisfies (2.6). □

A natural question is: Which pair of prolongation and projection is more efficient when used in multilevel computations? It is not easy to give a theoretical analysis to this question. Numerical studies in §7 indicate that the prolongation and projection (2.5) is better than the pair of prolongation and projection (2.8).

We will use the hierarchical property and the prolongation and projection mappings (2.5) to formulate our multi-p V-cycle methods in the following sections.

3. Multi-*p* V-cycle methods. Let $F(u, f)$ be a linear stationary iterative scheme, such as Gauss–Seidel or SOR in operator form, that is, $F(u_j, f_j) : V_j \to V_j, u_j \in V_j, j = 1, 2, \ldots, p$. And let an energy norm be defined as

$$\|u\|_E = (a(u \cdot u))^{1/2}.$$

The problem that we want to solve is to find the function $u \in V_p CH(\Omega)$ such that

$$a(u, v) = f(v) \quad \forall v \in V_p, \tag{3.1}$$

where $a(u, v)$ and $f(v)$ are defined as in §2.

In analogy with the multigrid V-cycle algorithm, we have a multi-p V-cycle algorithm. We call it the standard multi-p V-cycle algorithm (SMPV).

ALGORITHM. STANDARD MULTI-p V-CYCLE: SMPV (u_p, f_p).

Let $\mu \geq 1$ be a given integer. Let u_p^0 be an initial guess.

(i) If $p = 1$, solve (3.1) directly;

(ii) If $p \geq 2$, then

(1) For $i = 1, 2, \ldots, \mu$, compute $u_p^i = F(u_p^{i-1}, f_p)$;

(2) Perform one cycle of SMPV (v_{p-1}, f_{p-1}) algorithm at level $p-1$ to the following correction problem starting with initial guess $v_{p-1}^0 = 0$: Find $v_{p-1} \in V_{p-1}$ such that

$$a(v_{p-1}, w_{p-1}) = f(w_{p-1}) - a(u_p^\mu, w_{p-1}) \quad \text{for all } w_{p-1} \in V_{p-1}. \tag{3.2}$$

Let the solution be $\tilde{v}_{p-1}$;

(3) Let $u_p^{\mu+1} = u_p^\mu + \tilde{v}_{p-1}$;

(4) For $i = \mu + 2, \mu + 3, \ldots, 2\mu + 1$, compute

$$u_p^i = F(u_p^{i-1}, f_p);$$

At highest level p, if the residual is within the given tolerance, stop; otherwise, let $u_p^0 = u_p^{2\mu+1}$ and go back to (1).

Remark. The SMPV (u_p, f_p) algorithm looks exactly like the multigrid V-cycle algorithm; the only difference is that the multilevel computations here are conducted at levels of different degrees of shape functions, not at different grids. The mesh in the p-version is fixed. This difference can be seen more clearly in matrix form.

Let A_p and b_p be the assembled global stiffness matrix and load vector, respectively. If A_p and b_p correspond to degree $\leq p$, then we can write $A_p x = b_p$ in the block form

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{12}^{\mathrm{T}} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, \tag{3.3}$$

where A_{11} and b_1 correspond to degree $\leq p - 1$. Let $F(x, b)$ be a linear stationary scheme in matrix form.

A two-level standard multi-p V-cycle algorithm in matrix form can be stated as follows.

ALGORITHM.

Let an initial guess be $[x_1^0 x_2^0]^{\mathrm{T}}$.

(1) Perform μ iterations of $F(x, b)$ on (3.3), and let the solution be $[x_1^\mu x_2^\mu]^{\mathrm{T}}$;

(2) Solve

$$A_{11} x_1 = b_1 - A_{12} x_2^\mu \quad \text{(Projection)}$$

directly (or iteratively with initial guess x_1^μ) to obtain $x_1^{\mu+1}$;

(3) Perform μ iterations on (3.3) with initial guess $[x_1^{\mu+1} x_2^\mu]^{\mathrm{T}}$ (Prolongation) to obtain $[x_1^{2\mu+1} x_2^{2\mu}]^{\mathrm{T}}$;

(4) If the residual is within the tolerance, stop; otherwise let $[x_1^{2\mu+1} x_2^{2\mu}]^T \Rightarrow [x_1^0 x_2^0]^T$ and return to (1).

From the above algorithm, it is easy to see that the lower level computations only improve components in x_1. This is more significant in multilevel cases, and this observation leads us to a modified V-cycle algorithm. Because of the hierarchical property of V_p, it can be decomposed as

$$V_p = V_{p-1} \oplus V_{-p},$$

where V_{-p} is the set of polynomials that are of degree p but not of lower degree.

ALGORITHM. MODIFIED MULTI-p V-CYCLE: MMPV (u_p, f_p).
Let $\mu \geq 1$ and $\sigma \geq 1$ be given integers. Let u_p^0 be an initial guess.

(i) If $p = 1$, solve (3.1) directly;

(ii) If $p \geq 2$, then

(1) For $i = 1, 2, \ldots, \mu$, compute

$$u_p^i = F(u_p^{i-1}, f_p);$$

(2) Perform one cycle of MMPV (v_{p-1}, f_{p-1}) algorithm at level $p-1$ to the following correction problem starting with initial guess $v_{p-1}^0 = 0$:
Find $v_{p-1} \in V_{p-1}$ such that

$$a(v_{p-1}, w_{p-1}) = f(w_{p-1}) - a(u_p^\mu, w_{p-1}) \text{ for all } w_{p-1} \in V_{p-1}.$$

Let the solution be $\tilde{v}_{p-1}$;

(3) Let $u_p^{\mu+\frac{1}{2}} = u_p^\mu + \tilde{v}_{p-1}$;

(4) Perform σ iterations of $F(u, f)$ to the following problem with initial guess $u_{-p}^0 = 0$: Find $u_{-p} \in V_{-p}$ such that

$$a(u_{-p}, v_{-p}) = f(v_{-p}) - a\left(u_p^{\mu+\frac{1}{2}}, v_{-p}\right) \text{ for all } v_{-p} \in V_{-p}. \tag{3.4}$$

Let the solution be $\tilde{u}_{-p}$;

(5) Let $u_p^{\mu+1} = u_p^{\mu+\frac{1}{2}} + \tilde{u}_{-p}$;

(6) For $i = \mu+2, \mu+3, \ldots, 2\mu+1$, compute

$$u_p^i = F(u_p^{i-1}, f_p);$$

At highest level p, if the residual is within the given tolerance, stop; otherwise, let $u_p^0 = u_p^{2\mu+1}$ and go back to (1).

A two-level modified V-cycle algorithm in matrix form can be stated as follows.

ALGORITHM.
Let an initial guess be $[x_1^0 x_2^0]^T$.

(1) Perform μ iterations of $F(x, b)$ on the equation (3.3), let the solution be $[x_1^\mu x_2^\mu]^T$;

(2) Solve

$$A_{11}x_1 = b_1 - A_{12}x_2^\mu \quad \text{(Projection)}$$

directly (or iteratively with initial guess x_1^μ) to obtain $x_1^{\mu+1}$;

(3) Perform σ iterations of $F(x, b)$ on the following equations:

$$A_{22}x_2 = b_2 - A_{12}^T x_1^{\mu+1}$$

with initial guess x_2^μ, and let the solution be $x_2^{\mu+1}$;

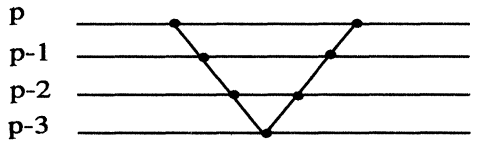

FIG. 3. *The standard* V*-cycle.*

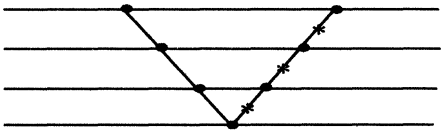

FIG. 4. *The modified* V*-cycle.*

(4) Perform μ iterations on (3.3) with initial guess $[x_1^{\mu+1}x_2^{\mu+1}]^T$ (Prolongation) to obtain $[x_1^{2\mu+1}x_2^{2\mu+1}]^T$;
(5) If the residual is within the tolerance, stop; otherwise, let $[x_1^{2\mu+1}x_2^{2\mu+1}]^T \Rightarrow [x_1^0x_2^0]^T$ and return to (1).

Figures 3 and 4 show the differences between the standard V-cycle and the modified V-cycle algorithms. Although the standard multi-p V-cycle algorithm looks exactly like a multigrid V-cycle, we can not directly adopt the convergence analysis of the multigrid V-cycle for our multi-p V-cycle. In multigrid V-cycle analysis, a basic assumption is the quasi-uniform mesh [12], [13], [15]. This assumption provides a nice result, that is, the spectral radius $\rho(A_h)$ of the underlying matrix A_h satisfies [24]

$$\rho(A_h) \leq ch^{-2}.$$

This inequality is essential to many convergence analyses of multigrid V-cycle methods [12], [13], [15], [28]. In the p-version, the corresponding condition should read $\rho(A_p) \leq cp^2$ in two dimensions, but the following lemma indicates that this might not be true.

Associated with each space V_p are eigenvalues $\lambda_i^{(p)}$ and eigenvectors $\psi_i^{(p)}$, $1 \leq i \leq N_p$, satisfying

$$a(\psi_i^{(p)}, v) = \lambda_i^{(p)}(\psi_i^{(p)}, v)_{L^2} \quad \text{for all } v \in V_p.$$

LEMMA 3.1. *In two dimensions, the maximum eigenvalue* $\lambda_{\max}(a)$ *of* $a(u, v)$ *in the p-version satisfies*

$$c_1 p^3 \leq \lambda_{\max}(a) \leq c_2 p^4,$$

where $c_1 > 0$ *and* $c_2 > 0$ *are constants independent of* p.

Proof. Without loss of generality, we can assume that the problem domain is the standard square element $\Omega = (-1, 1) \times (-1, 1)$. In order to prove this lemma, we need the following properties of the Legendre polynomials [30]:

$$\int_{-1}^{1} P_m(x) \cdot P_n(x)\, dx = \begin{cases} 0 & \text{if } m \neq n, \\ \dfrac{2}{2n+1} & \text{if } m = n; \end{cases}$$

$$P_n'(x) = \sum_{k=0}^{n-2k-1\geq 0} (2n - 4n - 1)P_{n-2k-1}(x).$$

Based on these two properties, it can be shown that

$$\int_{-1}^{1} [P_n'(x)]^2\, dx = n(n+1).$$

By using these properties, we have

$$\begin{aligned} a(P_n, P_n) &= \int_\Omega (a\nabla P_n \cdot \nabla P_n + bP_nP_n)\,dx \\ &= \int_{-1}^{1}\int_{-1}^{1} a[P_n'(x)]^2\,dx\,dy + \int_{-1}^{1}\int_{-1}^{1} b[P_n(x)]^2\,dx\,dy \\ &\geq 2\alpha \int_{-1}^{1} [P_n'(x)]^2\,dx \\ &= 2\alpha n(n+1), \end{aligned}$$

where $\alpha > 0$ is the parameter defined in the partial differential equation of our model problem. We also have

$$\begin{aligned} (P_n, P_n)_{H^0(\Omega)} &= \int_{-1}^{1}\int_{-1}^{1} [P_n(x)]^2\,dx\,dy \\ &= \frac{4}{2n+1}. \end{aligned}$$

For the left-hand inequality, by choosing Legendre polynomials $P_p(x)$ with degree p as special functions, we have

$$\begin{aligned} \lambda_{\max}(a) &= \max_{u\neq 0,\, u\in V_p} \frac{a(u,u)}{(u,u)_{H^0(\Omega)}} \\ &\geq \frac{a(P_p, P_p)}{(P_p, P_p)_{H^0(\Omega)}} \\ &\geq \frac{\alpha}{2} p(p+1)(2p+1) \\ &\geq c_1 p^3. \end{aligned}$$

For the right-hand side inequality, by using Lemma 3.32 of [1], that is, for each integer $k \geq 0$,

$$\|u_p\|_{H^k(\Omega)} \leq c(k)p^{2k}\|u_p\|_{H^0(\Omega)},$$

we have, for $k = 1$,

$$a(u_p, u_p) \leq \|u_p\|^2_{H^1(\Omega)} \leq c(1)p^4\|u_p\|^2_{H^0(\Omega)}$$

and the inequality follows. □

Remark. From the proof, it is easy to see that the inequality holds for the problem domain in R^d, $d \geq 1$, arbitrary.

Remark. The relationship between the spectral radius $\rho(A_p)$ of a stiffness matrix and $\lambda_{\max}(a)$ can be obtained, which indicates that, in general, the inequality $\rho(A_p) \leq cp^2$ may not hold in the p-version. In particular, if the eigenfunctions of $a(u, v)$ defined above are chosen to be the basis functions, then we have $\rho(A_p) = \lambda_{\max}(a) \geq c_1 p^3$. Therefore, in general, the analysis of multigrid V-cycle cannot be adopted for the multi-p V-cycle. However, for multi-p V-cycle algorithms, we have the following theorems.

THEOREM 3.1. *If the linear iterative scheme $F(u, f)$ satisfies*

$$\|u_p - F(u_p^{i-1}, f_p)\|_E \leq \delta_p \|u_p - u_p^{i-1}\|_E \tag{3.5}$$

where $u_p \in V_p$ is the exact solution to (3.1), $u_p^i \in V_p$, $\delta_p < 1$, $p = 1, 2, \ldots, i = 1, 2, \ldots$, *and $u_p^0 \in V_p$ is an initial guess. Then, the solution $u_p^{2\mu+1}$ obtained after one cycle of the standard multi-p V-cycle algorithm satisfies*

$$\|u_p - u_p^{2\mu+1}\|_E \le \delta_p^{2\mu} \eta_p \|u_p - u_p^0\|_E \tag{3.6}$$

where $0 \le \eta_p < 1$.

Proof. The proof is by induction. For $p = 1$, the theorem holds obviously because (3.1) is solved directly. Assume now that the theorem holds for $p - 1$ with $p > 1$; we want to show that it still holds for p. Let $e_p^\mu = u_p - u_p^\mu$; then

$$\begin{aligned}
\|u_p - u_p^{2\mu+1}\|_E^2 &\le \delta_p^{2\mu} \|u_p - u_p^{\mu+1}\|_E^2 \\
&= \delta_p^{2\mu} \|u_p - u_p^\mu - \tilde{v}_{p-1}\|_E^2 \\
&= \delta_p^{2\mu} \|e_p^\mu - v_{p-1} + v_{p-1} - \tilde{v}_{p-1}\|_E^2,
\end{aligned}$$

where v_{p-1} is the finite element solution to (3.2). From (3.2), we have

$$\begin{aligned}
a(v_{p-1}, w_{p-1}) &= f(w_{p-1}) - a(u_p^\mu, w_{p-1}) \\
&= a(u_p, w_{p-1}) - a(u_p^\mu, w_{p-1}) \\
&= a(e_p^\mu, w_{p-1}),
\end{aligned}$$

i.e.,

$$a(e_p^\mu - v_{p-1}, w_{p-1}) = 0 \quad \text{for all } w_{p-1} \in V_{p-1}. \tag{3.7}$$

From (3.7), it is easy to see that

$$\|e_p^\mu\|_E^2 = \|e_p^\mu - v_{p-1}\|_E^2 + \|v_{p-1}\|_E^2$$

or

$$\|e_p^\mu - v_{p-1}\|_E^2 = \|e_p^\mu\|_E^2 - \|v_{p-1}\|_E^2. \tag{3.8}$$

Therefore, since $v_{p-1}^0 = 0$, we have

$$\begin{aligned}
\|u_p - u_p^{2\mu+1}\|_E^2 &\le \delta_p^{2\mu} \left[\|e_p^\mu - v_{p-1}\|_E^2 + \|v_{p-1} - \tilde{v}_{p-1}\|_E^2\right] \\
&\le \delta_p^{2\mu} \left[\|e_p^\mu\|_E^2 - \|v_{p-1}\|_E^2 + \|v_{p-1} - \tilde{v}_{p-1}\|_E^2\right] \\
&\le \delta_p^{2\mu} \left[\|e_p^\mu\|_E^2 - \|v_{p-1}\|_E^2 + \delta_{p-1}^{4\mu} \|v_{p-1}\|_E^2\right] \\
&\le \delta_p^{2\mu} \left[\|e_p^\mu\|_E^2 - (1 - \delta_{p-1}^{4\mu}) \|v_{p-1}\|_E^2\right] \\
&\le \delta_p^{2\mu} \left[1 - (1 - \delta_{p-1}^{4\mu}) \frac{\|v_{p-1}\|_E^2}{\|e_p^\mu\|_E^2}\right] \|e_p^\mu\|_E^2 \\
&\le \delta_p^{4\mu} \left[1 - (1 - \delta_{p-1}^{4\mu}) \xi_p^2\right] \|e_p^0\|_E^2,
\end{aligned}$$

where $0 \le \xi_p = \|v_{p-1}\|_E / \|e_p^\mu\|_E \le 1$ from (3.8). Therefore, we get

$$\|u_p - u_p^{2\mu+1}\|_E \le \delta_p^{2\mu} \eta_p \|u_p - u_p^0\|_E$$

where $\eta_p = [1 - (1 - \delta_{p-1}^{4\mu}) \xi_p^2]^{1/2}$. □

Remark. If we just apply 2μ number of iterations of $F(u, f)$ at the level p without any lower-level computations, we can only get a convergence factor of $\delta_p^{2\mu}$. Now, Theorem 3.1 shows that, by adding lower-level computations as described in the algorithm, we get a better convergence factor of $\delta_p^{2\mu}\eta_p$. What remains to be determined is whether this faster convergence can compensate for the extra computations that are required. The numerical results in §6 indicate that it does indeed compensate. The next theorem is concerned with the convergence of the modified V-cycle algorithm.

THEOREM 3.2. *Under the same assumption as in Theorem* 3.1, *the solution* $u_p^{2\mu+1}$ *after one cycle of the modified multi-p V-cycle algorithm satisfies*

$$\|u_p - u_p^{2\mu+1}\|_E \leq \delta_p^{2\mu}\tilde{\eta}_p\|u_p - u_p^0\|_E,$$

where $0 \leq \tilde{\eta}_p < 1$.

Proof. It is easy to see that

$$\begin{aligned}\|u_p - u_p^{2\mu+1}\|_E^2 &\leq \delta_p^{2\mu}\|u_p - u_p^{\mu+1}\|_E^2 \\ &= \delta_p^{2\mu}\|u_p - u_p^{\mu+1/2} - \tilde{u}_{-p}\|_E^2 \\ &= \delta_p^{2\mu}\|u_p - u_p^{\mu} - \tilde{v}_{p-1} - \tilde{u}_{-p}\|_E^2.\end{aligned}$$

From (3.4), we have

$$\begin{aligned}a(u_{-p}, w_{-p}) &= f(w_{-p}) - a(u_p^{\mu+1/2}, w_{-p}) \\ &= a(u_p, w_{-p}) - a(u_p^{\mu} + \tilde{v}_{p-1}, w_{-p}) \\ &= a(e_p^{\mu} - \tilde{v}_{p-1}, w_{-p})\end{aligned}$$

for any $w_{-p} \in V_{-p}$; thus,

$$a(e_p^{\mu} - \tilde{v}_{p-1} - u_{-p}, w_{-p}) = 0 \quad \text{for all } w_{-p} \in V_{-p}, \tag{3.9}$$

and by taking $w_{-p} = u_{-p}$, we have

$$a(u_{-p}, u_{-p}) = a(e_p^{\mu} - \tilde{v}_{p-1}, u_{-p}). \tag{3.10}$$

Using (3.9), we get

$$\|u_p - u_p^{2\mu+1}\|_E^2 \leq \delta_p^{2\mu}\left(\|e_p^{\mu} - \tilde{v}_{p-1} - u_{-p}\|_E^2 + \|u_{-p} - \tilde{u}_{-p}\|_E^2\right), \tag{3.11}$$

where u_{-p} is the exact solution of (3.4).

Now, consider the first term of (3.11):

$$\begin{aligned}\|e_p^{\mu} - \tilde{v}_{p-1} - u_{-p}\|_E^2 &= a\left(e_p^{\mu} - \tilde{v}_{p-1} - u_{-p}, e_p^{\mu} - \tilde{v}_{p-1} - u_{-p}\right) \\ &= a\left(e_p^{\mu} - \tilde{v}_{p-1} - u_{-p}, e_p^{\mu} - \tilde{v}_{p-1}\right) - a\left(e_p^{\mu} - \tilde{v}_{p-1} - u_{-p}, -u_{-p}\right).\end{aligned}$$

Therefore, by using (3.9) and (3.10), we have

$$\begin{aligned}\|e_p^{\mu} - \tilde{v}_{p-1} - u_{-p}\|_E^2 &= a\left(e_p^{\mu} - \tilde{v}_{p-1} - u_{-p}, e_p^{\mu} - \tilde{v}_{p-1}\right) \\ &= a\left(e_p^{\mu} - \tilde{v}_{p-1}, e_p^{\mu} - \tilde{v}_{p-1}\right) - a\left(u_{-p}, e_p^{\mu} - \tilde{v}_{p-1}\right) \\ &= \|e_p^{\mu} - \tilde{v}_{p-1}\|_E^2 - a(u_{-p}, u_{-p}) \\ &= \|e_p^{\mu} - \tilde{v}_{p-1}\|_E^2 - \|u_{-p}\|_E^2.\end{aligned}$$

From the proof of Theorem 3.1, we know that

$$\|e_p^{\mu} - \tilde{v}_{p-1}\|_E^2 \leq \delta_p^{2\mu}\eta_p^2\|u_p - u_p^0\|_E^2;$$

thus, we get

$$\|e_p^{\mu} - \tilde{v}_{p-1} - u_{-p}\|_E^2 \leq \delta_p^{2\mu}\eta_p^2\|u_p - u_p^0\|_E^2 - \|u_{-p}\|_E^2.$$

For the second term of (3.11), it is easy to see that

$$\|u_{-p} - \tilde{u}_{-p}\|_E^2 \leq \delta_{-p}^{2\sigma}\|u_{-p}\|_E^2.$$

Therefore, we get

$$\begin{aligned}
\|u_p - u_p^{2\mu+1}\|_E &\leq \delta_p^{2\mu}\left[\delta_p^{2\mu}\eta_p^2\|u_p - u_p^0\|_E^2 - \|u_{-p}\|_E^2 + \delta_{-p}^{2\sigma}\|u_{-p}\|_E^2\right] \\
&\leq \delta_p^{2\mu}\left[\delta_p^{2\mu}\eta_p^2\|u_p - u_p^0\|_E^2 - \left(1 - \delta_{-p}^{2\sigma}\right)\|u_{-p}\|_E^2\right] \\
&\leq \delta_p^{4\mu}\left[\eta_p^2 - \frac{\left(1 - \delta_{-p}^{2\sigma}\right)}{\delta_p^{2\mu}}\frac{\|u_{-p}\|_E^2}{\|u_p - u_p^0\|_E^2}\right]\|u_p - u_p^0\|_E^2 \\
&\leq \delta_p^{4\mu}\left[\eta_p^2 - \frac{\left(1 - \delta_{-p}^{2\sigma}\right)}{\delta_p^{2\mu}}\tilde{\xi}_p^2\right]\|u_p - u_p^0\|_E^2,
\end{aligned}$$

where $\tilde{\xi}_p = \|u_{-p}\|_E/\|u_p - u_p^0\|_E$. Let

$$\tilde{\eta}_p = \left[\eta_p^2 - \frac{\left(1 - \delta_{-p}^{2\sigma}\right)}{\delta_p^{2\mu}}\tilde{\xi}_p^2\right]^{1/2} \quad \left(\text{we show later that } \eta_p^2 - \frac{\left(1 - \delta_{-p}^{2\sigma}\right)}{\delta_p^{2\mu}}\tilde{\xi}_p^2 \geq 0\right);$$

then we get

$$\|u_p - u_p^{2\mu+1}\|_E^2 \leq \delta_p^{4\mu}\tilde{\eta}_p^2\|u_p - u_p^0\|_E^2$$

or

$$\|u_p - u_p^{2\mu+1}\|_E \leq \delta_p^{2\mu}\tilde{\eta}_p\|u_p - u_p^0\|_E.$$

Now we show $0 \leq \tilde{\eta}_p < 1$. For the right-hand inequality, we have

$$\tilde{\eta}_p < \eta_p < 1.$$

For the left-hand inequality, we must show that

$$\eta_p^2 - \frac{\left(1 - \delta_{-p}^{2\,\sigma}\right)}{\delta_p^{2\mu}}\tilde{\xi}_p^2 \geq 0.$$

Using the Schwartz inequality

$$|a(u, v)|^2 \leq a(u, u) \cdot a(v, v)$$

and (3.10), we get

$$\|u_{-p}\|_E^2 \leq \|e_p^{\mu} - \tilde{v}_{p-1}\|_E\|u_{-p}\|,$$

so

$$\|u_{-p}\|_E \leq \|e_p^{\mu} - \tilde{v}_{p-1}\|_E \leq \delta_p^{\mu}\eta_p\|u_p - u_p^0\|_E$$

or

$$\frac{\|u_{-p}\|_E}{\|u_p - u_p^0\|_E} \leq \delta_p^{\mu}\eta_p;$$

therefore,

$$\begin{aligned}\tilde{\eta}_p &= \left[\eta_p^2 - \frac{\left(1-\delta_{-p}^{2\sigma}\right)}{\delta_p^{2\mu}}\left(\frac{\|u_{-p}\|_E}{\|u_p - u_p^0\|_E}\right)^2\right]^{1/2} \\ &\geq \left[\eta_p^2 - \frac{\left(1-\delta_{-p}^{2\sigma}\right)}{\delta_p^{2\mu}}\delta_p^{2\mu}\eta_p^2\right]^{1/2} \\ &= \eta_p\delta_{-p}^{\sigma} \\ &\geq 0.\end{aligned}$$

Thus, the proof is complete. □

Remark. It is easy to see that

$$\tilde{\eta}_p < \eta_p.$$

This indicates that by adding extra computations, we get a better error reduction. The numerical results in §6 agree with our theoretical results as can be seen in Table 1.

TABLE 1
Convergence comparison.

Method	Criterion: Residual $\|r\| < 10^{-n}$		
	10^{-6}	10^{-7}	10^{-8}
SMPV	32 cycles	46 cycles	61 cycles
MMPV	27 cycles	40 cycles	52 cycles

Remark. From Theorems 3.1 and 3.2, it is easy to see that the assumptions required here are very simple and can be satisfied easily. We do not require any regularity assumptions or assumptions on domain geometry as required by convergence analysis in multigrid V-cycle methods [12], [13], [15], [26], [29]. The regularity-free convergence analysis of multigrid methods is studied by some researchers [25], where the convergence factors have the asymptotic behavior $(1 - \text{const} \cdot q^m)$, $0 < q < 1$. It is possible to get better convergence results for our multi-p methods if some additional assumptions are imposed. But this may restrict the applicability of multi-p methods to a smaller class of problems.

We now introduce another variant of the standard V-cycle.

ALGORITHM. VARYING MULTI-p V-CYCLE: VMPV (u_p, f_p).
Let $\mu \geq 1$ and $\sigma \geq 1$ be given integers. Let u_p^0 be an initial guess.
(i) If $p = 1$, solve (3.1) directly;
(ii) If $p \geq 2$.
(1) Sub-cycle 1: Do one cycle of SMPV (u_p, f_p) (or MMPV (u_p, f_p)) with highest level p and lowest level 1;

(2) Sub-cycle 2: Do one cycle of SMPV (u_p, f_p) (or MMPV (u_p, f_p)) at all the even levels $1 < j < p$ including highest level p and lowest level 1;
(3) Sub-cycle 3: Do one cycle of SMPV (u_p, f_p) (or MMPV (u_p, f_p)) at all the odd levels $1 < j < p$ including highest level p and lowest level 1;
(4) If the residual is within the given tolerance, stop; otherwise, return to (1);

End

It is obvious that, by Theorems 3.1 and 3.2, the convergence of a varying V-cycle algorithm is guaranteed. The purpose here is to reduce the amount of computational work and attempt to improve the efficiency. Numerical simulations indicate that the varying V-cycle is more efficient as discussed in §6. From now on, when we say multi-p V-cycle, or MPV (u_p, f_p), we refer to any one of the three V-cycle algorithms discussed above. Figure 5 shows the computational processes of the varying V-cycle algorithm.

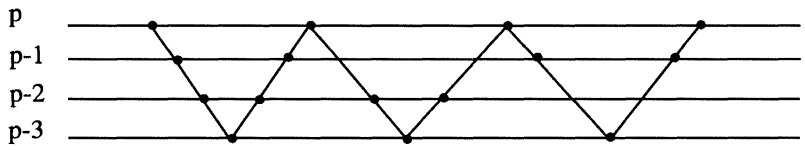

FIG. 5. *The varying V-cycle.*

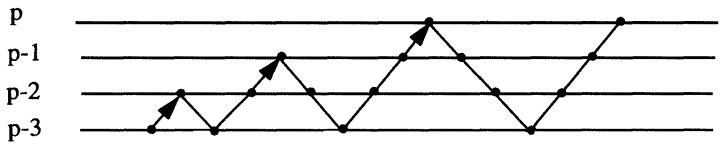

FIG. 6. *The nested multi-p algorithm.*

4. Nested multi-*p* methods. The nested multi-p algorithms are different from the V-cycles in that the computation starts from the lowest level and goes up to the highest level. At each level, we need an efficient iterative scheme to produce an approximate solution to the equation at that level. Since the V-cycle algorithms are more efficient than the classical iterative scheme, it is natural to use V-cycle algorithms at each level. The nested multi-p algorithms are given below. Here, we assume that $p = 1$ is the lowest level. It may be more efficient sometimes to choose a lowest level other than $p = 1$.

ALGORITHM. NESTED MULTI-p: NMP (u_p, f_p).
Let $p > 1, n \geq 1, \mu \geq 1$, and $\sigma \geq 1$ be given integers.
(i) At level $p = 1$, solve equation $a(u_1, v_1) = f(v_1)$ directly.
Let the solution be denoted by $\tilde{u}_1$.
(ii) For $j = 2$; increment 1; until p; do
Let $u_j^0 = \tilde{u}_{j-1}$.
For $i = 1$; increment 1; until n; do
MPV(u_p, f_p)
End;
End.

Figure 6 shows the computational process of the nested multi-p algorithms.

The rest of this section will be devoted to the convergence analysis of nested multi-p methods. In order to do this, we need an approximation property of the p-version of the finite element method.

If $u \in H^k(\Omega)$ satisfies

$$a(u, v) = f(v) \quad \text{for all } v \in H^k(\Omega)$$

and $u_p \in V_p$ is the finite element approximation, then [1], [2]

$$\|u - u_p\|_{H^k(\Omega)} \leq c(k) p^{-(k-1)} \|u\|_{H^k(\Omega)}, \tag{4.1}$$

where $k > 1$. Using (4.1), it is easy to show the following result.

LEMMA 4.1. *Let $k > 1$, $p > 1$, and u_p and u_{p-1} be the finite element approximations. Then*

$$\|u_p - u_{p-1}\|_{H^k(\Omega)} \leq c_1(k) p^{-(k-1)}, \tag{4.2}$$

where $c_1(k)$ is a constant independent of p.

The proof of this lemma is very easy and is omitted.

Let $p > 1$ be a given integer. Let ρ_j $(1 \leq j \leq p)$ be the convergence factor of the multi-p V-cycle algorithm used in a nested multi-p method at level j. Define

$$\tilde{\rho} = \max_{1 \leq j \leq p} \rho_j. \tag{4.3}$$

THEOREM 4.1. *Let $k > 1$, $p > 1$, and $n \geq 1$ be given integers. Let the solution produced by a nested multi-p algorithm at level p be denoted by $\tilde{u}_p$. Then $\tilde{u}_p$ satisfies*

$$\|u_p - \tilde{u}_p\|_E \leq c_1(k) \tilde{\rho}^n c_2(\tilde{\rho}, n) p^{-\alpha}, \tag{4.4}$$

where

$$c_2(\tilde{\rho}, n) = \frac{1}{1 - 2^{\alpha} \tilde{\rho}^n}, \tag{4.5}$$

$$\alpha = \min\{k - 1, -\beta \log_2 \tilde{\rho}^n\}, \tag{4.6}$$

and $\beta = 1 - \varepsilon$, $0 < \varepsilon < 1$, is a very small parameter.

Remark. In Theorem 4.1, if we choose n large enough so that $\alpha = k - 1$, then we have

$$\|u_p - \tilde{u}_p\|_E \leq c_1(k) \tilde{\rho}^n c_2(\tilde{\rho}, n) p^{-(k-1)}, \tag{4.7}$$

$$c_2(\tilde{\rho}, n) = \frac{1}{1 - 2^{k-1} \tilde{\rho}^n} \tag{4.8}$$

and it is easy to see that $2^{k-1} \tilde{\rho}^n < 1$ because

$$\log_2 2^{k-1} \tilde{\rho}^n = k - 1 + \log_2 \tilde{\rho}^n \leq -\beta \log_2 \tilde{\rho}^n + \log_2 \tilde{\rho}^n = (1 - \beta) \log_2 \tilde{\rho}^n < 0.$$

In this case, the theorem says if n is such that $2^{k-1} \tilde{\rho}^n < 1$, then $\tilde{u}_p$ satisfies (4.7) and (4.8). Since $k - 1 < -\beta \log_2 \tilde{\rho}^n$ in this case, we have

$$p^{-(k-1)} > p^{\beta \log_2 \tilde{\rho}^n};$$

therefore, (4.7) can be written as

$$\|u_p - \tilde{u}_p\|_E \leq c_1(k) \tilde{\rho}^n c_2(\tilde{\rho}, n) \max\{p^{-(k-1)}, p^{\beta \log_2 \tilde{\rho}^n}\}. \tag{4.9}$$

On the other hand, if n is such that $\alpha = -\beta \log_2 \tilde{\rho}^n$ for any $\varepsilon > 0$ small, then we have

$$\|u_p - \tilde{u}_p\|_E \le c_1(k)\tilde{\rho}^n c_2(\tilde{\rho}, n) p^{\beta \log_2 \tilde{\rho}^n} \tag{4.10}$$

and

$$c_2(\tilde{\rho}, n) = \frac{1}{1 - 2^{-\beta \log_2 \tilde{\rho}^n} \tilde{\rho}^n}. \tag{4.11}$$

Note that it is always true that $2^{-\beta \log_2 \tilde{\rho}^n} \tilde{\rho}^n < 1$ because

$$\log_2 \left(2^{-\beta \log_2 \tilde{\rho}^n} \tilde{\rho}^n\right) = -\beta \log_2 \tilde{\rho}^n + \log_2 \tilde{\rho}^n = (1 - \beta) \log_2 \tilde{\rho}^n < 0.$$

Note also that, in this case, $2^{k-1}\tilde{\rho}^n \ge 1$ because, if we suppose that $2^{k-1}\tilde{\rho}^n < 1$, then we have $k - 1 + \log_2 \tilde{\rho}^n < 0$ or $k - 1 < -\log_2 \tilde{\rho}^n$. Choose

$$\beta = \frac{1}{2}\left(1 + \frac{k-1}{-\log_2 \tilde{\rho}^n}\right);$$

then $0 < \beta < 1$ and

$$-\beta \log_2 \tilde{\rho}^n = \frac{1}{2}\left[-\log_2 \tilde{\rho}^n + (k-1)\right] > \frac{1}{2}[(k-1) + (k-1)] = k - 1.$$

That is, for such a β, we have $\alpha = k - 1$, which is a contradiction. Therefore, in this case, the theorem says if $2^{k-1}\tilde{\rho}^n \ge 1$, then $\tilde{u}_p$ satisfies (4.10) and (4.11). Since $k - 1 > -\beta \log_2 \tilde{\rho}^n$ in this case, we have

$$p^{-(k-1)} < p^{\beta \log_2 \tilde{\rho}^n}$$

and (4.10) can be written as

$$\|u_p - \tilde{u}_p\|_E \le c_1(k)\tilde{\rho}^n c_2(\tilde{\rho}, n) \max\{p^{-(k-1)}, p^{\beta \log_2 \tilde{\rho}^n}\}. \tag{4.12}$$

From the preceding discussion, we see that (4.4) can always be written as (4.12) for any given $n \ge 1$.

Proof of Theorem 4.1. The proof is by induction. Equation (4.4) holds for $p = 1$ because the problem is solved directly. Let (4.4) hold for $p - 1$ with $p > 1$. We have

$$\begin{aligned}
\|u_p - \tilde{u}_p\|_E &\le \tilde{\rho}^n \|u_p - u_p^0\|_E \\
&= \tilde{\rho}^n \|u_p - \tilde{u}_{p-1}\| \\
&\le \tilde{\rho}^n \left[\|u_p - u_{p-1}\|_E + \|u_{p-1} - \tilde{u}_{p-1}\|_E\right] \\
&\le \tilde{\rho}^n \left[c_1(k)p^{-(k-1)} + c_1(k)\tilde{\rho}^n c_2(\tilde{\rho}, n)(p-1)^{-\alpha}\right] \\
&\le \tilde{\rho}^n \left[c_1(k)p^{-(k-1)} + c_1(k)\tilde{\rho}^n c_2(\tilde{\rho}, n)2^{\alpha} p^{-\alpha}\right] \\
&\le c_1(k)\tilde{\rho}^n \left[p^{-(k-1)+\alpha} + \tilde{\rho}^n c_2(\tilde{\rho}, n)2^{\alpha}\right] p^{-\alpha} \\
&\le c_1(k)\tilde{\rho}^n \left[p^{-(k-1)+\alpha} + \tilde{\rho}^n c_2(\tilde{\rho}, n)2^{\alpha}\right] p^{-\alpha}.
\end{aligned}$$

In the following, we want to show that

$$\left[p^{-(k-1)+\alpha_p} + \tilde{\rho}^n c_2(\tilde{\rho}, n)2^{\alpha}\right] \le c_2(\tilde{\rho}, n). \tag{4.13}$$

Since

$$p^{-(k-1)+\alpha} + \tilde{\rho}^n c_2(\tilde{\rho}, n)2^\alpha = \frac{1}{p^{(k-1)-\alpha}} + 2^\alpha \tilde{\rho}^n c_2(\tilde{\rho}, n)$$

$$= \frac{1}{p^{(k-1)-\alpha}} + \frac{2^\alpha \tilde{\rho}^n}{1 - 2^\alpha \tilde{\rho}^n}$$

$$\leq 1 + \frac{2^\alpha \tilde{\rho}^n}{1 - 2^\alpha \tilde{\rho}^n}$$

$$= c_2(\tilde{\rho}, n),$$

(4.13) is true and (4.4) follows. □

Remark. From this theorem, we see that, when using the nested multi-p algorithms, we can just apply one cycle of MPV at each level, i.e., $n = 1$. In practical computations, it is probably more efficient to choose $n = 1$ than to choose $n > 1$ when p is large because, from (4.4), we see that the term $p^{-\alpha}$ will be the dominating term for error reductions when p is large. The numerical simulations on sample problems confirm this observation and are given in §6.

5. Accelerated multi-*p* V-cycle methods. The numerical results given in §6 indicate that, for lower accuracy, the nested multi-p algorithms are more efficient than the multi-p V-cycle algorithms, but for higher accuracy, the V-cycle algorithms are more efficient than the nested multi-p algorithms. Therefore, we can take advantage of both methods by first using nested multi-p to produce a rough solution, and then using this solution as an initial guess to start a V-cycle iteration. This observation leads us to the so-called "accelerated multi-p V-cycle algorithms."

ALGORITHM. ACCELERATED MULTI-p V-CYCLE: AMPV (u_p, f_p).
Let $n \geq 1$, $\mu \geq 1$, and $\sigma \geq 1$ be given integers.
(i) If $p = 1$, solve equation (3.1) directly.
(ii) If $p \geq 2$, then
(1) Do
NMP (u_p, f_p)
Let the solution be denoted by $\tilde{u}_p$
End
(2) Let $u_p^0 = \tilde{u}_p$, do
MPV (u_p, f_p)
Let the solution be denoted by $\tilde{u}_p$
If the residual is within the given tolerance, stop; otherwise, return to (2)
End

Remark. In this algorithm, we do not specify which V-cycle algorithm is used; therefore, it can be any one of the three V-cycle algorithms discussed in §3.

THEOREM 5.1. *Let $\tilde{u}_p^j$ be the solution produced after j cycles of the accelerated standard multi-p V-cycle algorithm, where $j \geq 1$ is an integer. Then $\tilde{u}_p^j$ satisfies*

$$\|u_p - \tilde{u}_p^j\|_E \leq c_1(k)\tilde{\rho}_p^n c_2(\tilde{\rho}_p, n) p^{-\alpha} (\delta_p^{2\mu} \eta_p)^j, \tag{5.1}$$

where $c_2(\tilde{\rho}_p, n)$ is defined by (4.5) *and α is defined by* (4.6).

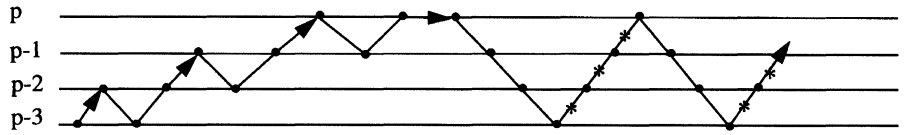

FIG. 7. *The combination of simplified nested multi-p algorithm with V-cycle.*

Proof. Using Theorem 3.1, we have

$$\|u_p - \tilde{u}_p^j\|_E \leq (\delta_p^{2\mu}\eta_p)^j \|u_p - \hat{u}_p\|_E$$

where $\hat{u}_p$ is the solution produced by the nested multi-p algorithm. Thus, using Theorem 4.1, we have

$$\|u_p - \tilde{u}_p^j\|_E \leq (\delta_p^{2\mu}\eta_p)^j c_1(k)\tilde{\rho}_p^n c_2(\tilde{\rho}_p, n) p^{-\alpha}. \qquad \square$$

Remark. If we replace η_p in (5.1) and (5.2) by $\tilde{\eta}_p$, then we get a convergence result for accelerated modified V-cycle algorithm.

The purpose of the nested multi-p process in the accelerated multi-p V-cycle algorithm is to produce an initial guess for a V-cycle algorithm efficiently. In practice, we found that a simplified nested multi-p process is more efficient. The simplification is as follows. During the nested multi-p process, at each level, instead of performing an entire V-cycle, we use a two-level V-cycle (see Fig. 7). In this way, the amount of the two-level V-cycle computational work is reduced but still produces a good initial guess for a V-cycle algorithm, and the overall computations are more efficient.

6. Numerical results. In this section, we present numerical simulation results based on two model problems. The first is a smooth problem and the second has a geometric singularity. In order to compare different algorithms, we need to find a measure. For this purpose, we introduce the so-called work unit as follows.

DEFINITION. *One work unit is defined as the equivalent amount of work required for performing one iteration of $F(u, f)$ on equation $Ax = b$ at the highest level, where A and b are the assembled global stiffness matrix and load vector.*

For example, if $F(u, f)$ represents the Gauss-Seidel scheme and, at the highest level p, the number of degrees of freedom is N_p, then one work unit is equivalent to N_p^2 operations.

We use two different linear iterative schemes as smoothers in our simulations. One scheme is the classical Gauss-Seidel scheme, and the other is a newly developed linear iterative scheme called the textured decomposition scheme [20], [23]. The reason we use this second scheme is that the textured decomposition scheme, when used recursively, is very suitable for parallel implementations [32]. Here, we give a brief introduction of a textured decomposition scheme. For more details, we refer to [18] and the references therein.

Let A be an $n \times n$ matrix. A multiple splitting

$$A = D_1 - E_1 = D_2 - E_2 = \cdots = D_m - E_m$$

is called a textured decomposition if D_i is nonsingular for $i = 1, 2, \ldots, m$. Each such splitting (D_i, E_i) is called a leaf.

Suppose we have an m-leaf textured decomposition, where $m \geq 2$ is a given integer. Then a textured decomposition iteration scheme for solving $Ax = b$ is as follows:

$$
\begin{aligned}
D_1 x(k+1) &= E_1 x(k) + b, \\
D_2 x(k+2) &= E_2 x(k+1) + b, \\
&\cdots \\
D_m x(k+m) &= E_m x(k+m-1) + b, \qquad k = 0, 1, 2, \ldots,
\end{aligned}
$$

where $x(0)$ is an initial guess.

In our simulations, we use a two-leaf textured decomposition scheme.

Model Problem 1. Consider the two-dimensional boundary value problem (see [27])

$$
\begin{aligned}
-\Delta u &= 2[(1-6x^2)y^2(1-y^2) + (1-6y^2)x^2(1-x^2)], \\
u &= 0 \quad \text{on } \partial\Omega,
\end{aligned}
$$

where Ω is the unit square $= (0, 1) \times (0, 1)$.

We solve this problem by standard multi-p V-cycle (SMPV), modified multi-p V-cycle (MMPV), and varying multi-p V-cycle (VMPV) algorithms combined with a two-leaf textured decomposition scheme (TD) as a smoother. The numerical results are given in Table 2.

TABLE 2
Model problem 1.

Criterion	Residual: $\|r\| < 10^{-6}$			
Method	TD	SMPV	MMPV	VMPV
Cycles	12	4	3	3
Work units	12	9.59	8.65	8.14
Residual	9.93×10^{-7}	6.28×10^{-7}	7.05×10^{-7}	2.93×10^{-7}

From Table 2, we observe that, in terms of the speed of convergence, the modified and varying V-cycle algorithms converge faster than the standard V-cycle algorithm and that all three V-cycle algorithms converge faster than the underlying iterative scheme (in this case, the TD scheme). These numerical results agree with our theoretical results. If we examine the efficiency in terms of work units and residual, we can see that the varying V-cycle is the most efficient algorithm among the three V-cycle algorithms and that they are all more efficient than the underlying iterative scheme. Since this problem is smooth, the numerical results based on this problem are not enough to support our assertion. Next, we test our method on a nonsmooth problem that has been a good benchmark problem for the p-version of the finite element analysis [4].

Model Problem 2. Consider the L-shaped domain problem (see [4])

$$
\Delta u = 0 \quad \text{in } \Omega,
$$

where Ω is an L-shaped domain as shown in Fig. 8. The boundary conditions are

$$
\begin{aligned}
u &= 0 \quad \text{on } \Gamma^1 \text{ and } \Gamma^2, \\
\frac{\partial u}{\partial n} &= 1 \quad \text{on } \Gamma^4, \Gamma^5, \Gamma^6, \text{ and } \Gamma^7, \\
\frac{\partial u}{\partial n} &= 0 \quad \text{on } \Gamma^3 \text{ and } \Gamma^8.
\end{aligned}
$$

The finite element mesh used for solving this problem is shown in Fig. 9.

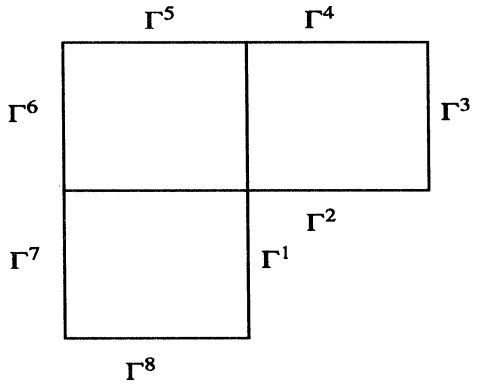

FIG. 8. *L-shaped domain.*

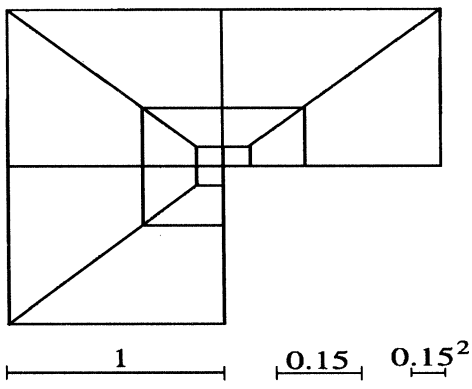

FIG. 9. *The mesh in the p-version.*

TABLE 3
L-shaped domain problem.

Criterion	Residual: $\|r\| < 10^{-6}$			
Method	G-S	SMPV	MMPV	VMPV
Cycles	245	32	27	34
Work units	245	143.71	129.67	126.79
Timing (sec)	321.6	152.3	141.9	139.6
Residual	9.90×10^{-7}	8.83×10^{-7}	9.64×10^{-7}	7.88×10^{-7}

TABLE 4
L-shaped domain problem.

Criterion	Residual: $\|r\| < 10^{-7}$			
Method	G-S	SMPV	MMPV	VMPV
Cycles	346	46	40	49
Work units	346	206.58	185.24	182.74
Timing (sec)	439.2	209.5	200.0	193.5
Residual	9.87×10^{-8}	9.82×10^{-8}	8.87×10^{-8}	7.53×10^{-8}

TABLE 5
L-shaped domain problem.

Criterion	Residual: $\|r\| < 10^{-8}$			
Method	G-S	SMPV	MMPV	VMPV
Cycles	447	61	52	64
Work units	447	273.95	240.81	238.65
Timing (sec)	561.2	271.2	253.9	247.6
Residual	9.86×10^{-9}	9.38×10^{-9}	9.85×10^{-9}	7.23×10^{-9}

We use the classical Gauss-Seidel iterative scheme as a smoother. First, we solve this L-shaped domain problem by standard multi-p V-cycle, modified V-cycle, and varying V-cycle algorithms. The numerical results are given in Tables 3, 4, and 5.

Based on Tables 3, 4, and 5, we plot a diagram as shown in Fig. 10. It is obvious from Fig. 10 that the modified V-cycle is more efficient than the standard V-cycle and that the varying V-cycle is the most efficient algorithm.

Next, we apply the nested multi-p algorithms to the L-shaped domain problem. The results are presented in Table 6.

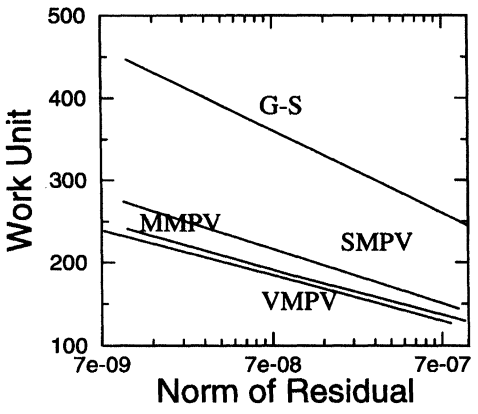

FIG. 10. *Comparison of V-cycles.*

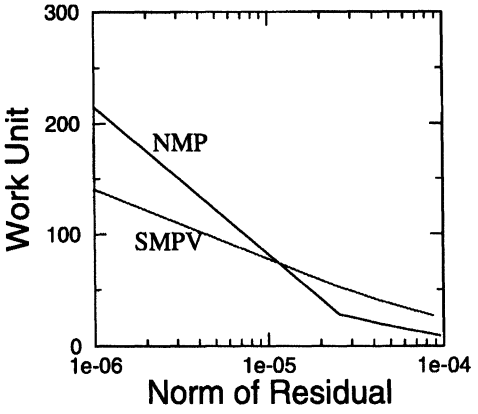

FIG. 11. *Comparison of NMP with SMPV.*

TABLE 6
Nested multi-p methods for L-shaped domain problem.

Number of V-cycles at each level	1	2	3	23
Residual	9.54×10^{-5}	4.61×10^{-5}	2.55×10^{-5}	9.89×10^{-7}
Work units	9.38	18.76	28.13	215.69

TABLE 7
Standard V-cycle method for L-shaped domain problem.

Number of V-cycles	6	9	12	32
Residual	8.73×10^{-5}	4.35×10^{-5}	2.41×10^{-5}	8.83×10^{-7}
Work units	26.95	40.42	53.89	143.71

TABLE 8
L-shaped domain problem.

Criterion	Residual: $\|r\| < 10^{-6}$			
Method	G-S	ASMPV	AMMPV	AVMPV
Cycles	245	11	9	13
Work units	245	55.18	47.46	54.25
Timing (sec)	321.6	70.6	65.9	72.5
Residual	9.90×10^{-7}	9.01×10^{-7}	8.93×10^{-7}	5.53×10^{-7}

It is not easy to determine whether multi-p V-cycle or nested multi-p algorithms are more efficient theoretically. Therefore, we want to get some ideas from numerical studies. Table 7 contains such numerical results.

Based on Tables 6 and 7, we plot the data in Fig. 11. From Fig. 11, we can see that, for low accuracy, the nested multi-p algorithm is more efficient in terms of work units, but the standard V-cycle algorithm is more efficient for high accuracy.

Finally, the L-shaped problem is solved by the accelerated multi-p V-cycle algorithms. The numerical results are given in Tables 8, 9, and 10. Based on Tables 8, 9, and 10, we plot a diagram as shown in Fig. 12.

It is obvious that the accelerated V-cycles indeed accelerate the convergence speed compared with the corresponding V-cycle algorithms. For this problem, we achieve an 80% improvement (for AMMPV over G-S when $\|r\| < 10^{-6}$) in terms of computational time and work units.

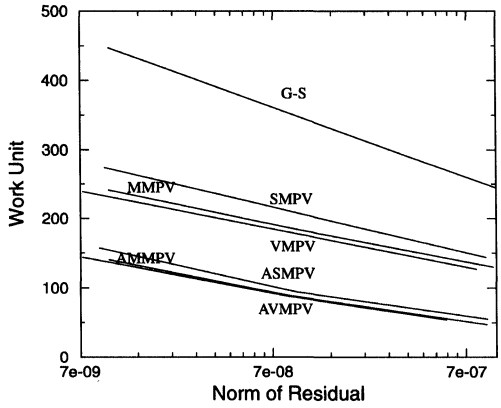

FIG. 12. *Comparison of V-cycles with accelerated V-cycles.*

TABLE 9
L-shaped domain problem.

Criterion	Residual: $\|r\| < 10^{-7}$			
Method	G-S	ASMPV	AMMPV	AVMPV
Cycles	346	21	18	22
Work units	346	94.31	89.13	87.81
Timing (sec)	439.2	111.7	107.0	100.4
Residual	9.37×10^{-8}	9.28×10^{-8}	8.46×10^{-8}	8.53×10^{-8}

TABLE 10
L-shaped domain problem.

Criterion	Residual: $\|r\| < 10^{-8}$			
Method	G-S	ASMPV	AMMPV	AVMPV
Cycles	447	35	29	37
Work units	447	157.18	140.08	143.75
Timing (sec)	561.2	169.3	156.9	154.0
Residual	9.86×10^{-9}	8.79×10^{-9}	9.00×10^{-9}	7.22×10^{-9}

TABLE 11
Numerical studies of prolongations and projections.

Criterion	Residual: $\|r\| < 10^{-6}$	
Method	SMPV	New
Cycles	51	50
Timing (sec)	202.8	357.2

7. Conclusion. Before we summarize the results that are discussed in this paper, we want to point out one thing. All the multi-p algorithms we have presented are based on the prolongation and projection defined by (2.5). But Proposition 2.1 provides another pair of prolongation and projection defined by (2.8). It is not easy to compare the standard prolongation and projection (2.5) with the pair (2.8) through theoretical analysis, so we turn to numerical studies of their performance. For simplicity, we use a two-level standard multi-p V-cycle algorithm with Gauss-Seidel smoother. We apply this algorithm to the L-shaped domain problem and get the data presented in Table 11.

From Table 11, we can see that the two-level standard V-cycle algorithm based on the standard prolongation and projection (2.5) is much more efficient in terms of timing than the

same algorithm based on the prolongation and projection (2.8). This is also true for general multi-p V-cycle algorithms. This seems to be obvious because the new pair of prolongation and projection (2.8) requires the solutions of some linear equations, and this adds to the cost of computation.

To summarize, we have systematically studied multi-p methods through both theoretical analysis and numerical experiments. Numerical results agree and support our theoretical results and indicate that the accelerated multi-p V-cycle methods are very efficient for solving linear equations arising from the p-version of the finite element method.

The presentation of multi-p methods in this paper is based on the model problems in two dimensions. The extension to three-dimensional problems is straightforward. First, all the algorithms formulated here will be the same for three-dimensional problems if the finite element space V_p is constructed based on a problem domain Ω in three dimensions. Next, all the proofs are still valid in three dimensions except for the proof of the nested multi-p methods (Theorem 4.1), where we have used the p-version of the finite element approximation property in two dimensions. It is easy to see that the proof of Theorem 4.1 will still go through if, instead, the three-dimensional finite element approximation property is used. Third, implementations of our multi-p methods for three-dimensional problems will be the same if stiffness matrices and load vectors with hierarchical structure have been obtained.

Currently we are investigating the efficiency of conjugate gradient methods preconditioned by our multi-p processes. Some of the results were reported in [22]. We are also investigating the application of the multi-p methods to condensed finite elements. Initial results indicate that this is a very efficient approach to solving linear systems of equations that arise from the p-version of the finite element analysis; the results will be reported elsewhere.

REFERENCES

[1] I. BABUSKA, B. A. SZABO, AND I. N. KATZ, *The p-version of the finite element method*, SIAM J. Numer. Anal., 18 (1981), pp. 515-545.

[2] I. BABUSKA AND M. SURI, *The optimal convergence rate of the p-version of the finite element methods*, SIAM J. Numer. Anal., 24 (1987), pp. 750-776.

[3] ———, *The p- and h-p versions of the finite element method, An overview*, Comput. Methods Appl. Mech. Engrg., 80 (1990), pp. 5-26.

[4] ———, *The p- and h-p Versions of the Finite Element Method: Basic Principles and Properties*, Tech. report, 1993, Institute for Physical Science and Technology, University of Maryland College Park, College Park, MD.

[5] I. BABUSKA, M. GRIEBEL, AND J. PITKARANTA, *The Problem of Selecting the Shape Functions for a p-Type Finite Element*, Tech. report, 1988, Institute for Physical Science and Technology, University of Maryland College Park, College Park, MD.

[6] I. BABUSKA AND B. GUO, *The h-p version of the finite element method for problems with nonhomogeneous essential boundary condition*, Comput. Methods Appl. Mech. Engrg., 74 (1989), pp. 1-28.

[7] I. BABUSKA, B. GUO, AND E. P. STEPHAN, *The h-p version of the boundary element method with geometric mesh on polygonal domains*, Comput. Methods Appl. Mech. Engrg., 74 (1990), pp. 319-325.

[8] ———, *On the exponential convergence of the h-p version of the boundary element Galerkin methods on polygons*, Math. Methods Appl. Sci., 12 (1990), pp. 413-427.

[9] I. BABUSKA, B. GUO, AND M. SURI, *Implementation of nonhomogeneous Dirichlet boundary conditions in the p-version of the finite element method*, Impact Comput. in Sci. Engrg., 1 (1989), pp. 36-63.

[10] I. BABUSKA AND T. JANIK, *The h-p version of the finite element method for parabolic equations. Part I. The p-version in time*, Numer. Methods Partial Differential Equations, 5 (1989), pp. 363-399.

[11] I. BABUSKA AND T. JANIK, *The h-p version of the finite element method for parabolic equations. Part II. The h-p version in time*. Numer. Methods Partial Differential Equations, 6 (1990), pp. 343-369.

[12] R. E. BANK AND T. DUPONT, *An optimal order process for solving finite element equations*, Math. Comp., 36 (1981), pp. 35-51.

[13] D. BRAESS AND W. HACKBUSH, *A new convergence proof for the multigrid method including the V-cycle*, SIAM J. Numer. Anal., 20 (1983), pp. 967-975.

[14] J. BRAMBLE AND J. E. PASCIAK, *New convergence estimates for multigrid algorithms*, Math. Comp., 49 (1987), pp. 311-329.
[15] A. BRANDT, *Multi-level adaptive solutions to boundary value problems*, Math. Comp., 31 (1977), pp. 333-390.
[16] G. BRUSSINO, R. HERBIN, Z. CHRISTIDIS, AND V. SONNAD, *Parallel multilevel finite element method with hierarchical basis functions*, Tech. report, IBM Kingston, NY, 1989.
[17] A. W. CRAIG AND O. C. ZIENKIEWICZ, *A multigrid algorithm using a hierarchical finite element basis*, in Multigrid Methods for Integral and Differential Equations, D. J. Paddon and Holstein, eds., Clarendon Press, Oxford, 1985, pp. 310-312.
[18] S. FORESTI, G. BRUSSINO, S. HASSANZADEH, AND V. SONNAD, *Multilevel solution method for the p-version of the finite elements*, Comput. Phys. Comm. 53 (1989), pp. 349-355.
[19] W. HACKBUSH, *Multigrid Methods and Applications*, Springer-Verlag, Berlin, 1985.
[20] N. HU, AND I. N. KATZ, *Multi-p methods based on a textured decomposition scheme*, paper presented at SIAM Annual Meeting, Los Angeles, CA, 1992.
[21] N. HU AND I. N. KATZ, *Multi-p V-cycle methods accelerated by a nested multi-p procedure*, paper presented at SIAM Annual Meeting, Philadelphia, PA, 1993.
[22] N. HU AND I. N. KATZ, *Conjugate gradient methods preconditioned by multi-p processes*, paper presented at SIAM Annual Meeting, San Diego, CA, 1994.
[23] G. HUANG, W. K. TSAI, AND W. LU, *Fast parallel linear equation solvers based on textured decomposition*, in Proc. 28th Conference on Decision and Control, Los Angeles, CA, December 1987, pp. 1450-1454.
[24] C. JOHNSON, *Numerical Solution of Partial Differential Equations by the Finite Element Method*, Cambridge University Press, Cambridge, New York, 1990.
[25] J. F. MAITRE AND F. MUSY, *Multigrid methods: Convergence theory in a variational framework*; SIAM J. Numer. Anal., 21 (1984), pp. 657-671.
[26] S. F. MCCORMICK, *Multigrid methods for variational problems: general theory for the V-cycle*, SIAM J. Numer. Anal., 22 (1985), pp. 634-643.
[27] S. F. MCCORMICK, ED., *Multigrid Methods*, Frontiers Appl. Math. 3, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1987.
[28] J. MANDEL, S. MCCORMICK, AND J. RUGE, *An algebraic theory for multigrid methods for variational problems*, SIAM J. Numer. Anal., 25 (1988), pp. 91-110.
[29] J. MANDEL, *Some recent advances in multigrid methods*, Adv. Electron. Electron Phys., 82, Academic Press, London, 1991, pp. 327-377.
[30] G. SANSONE, *Orthogonal Functions*, Interscience Publishers, Inc., New York, 1959.
[31] B. A. SZABO AND I. BABUSKA, *Finite Element Analysis*, J. Wiley & Sons, New York, 1991.
[32] Y. ZHU AND I. N. KATZ, *A parallel implementation of the p-version of the finite element method*, SIAM J. Sci. Comput., 17 (1996), to appear.