EXERCISES

10.5.1 Use Taylor expansion (Theorem 10.1.2) to give a proof of Theorem 10.5.3.

10.5.2 Give an alternative to Theorem 10.5.3 when $F\colon X \to Y$ has the additional structure

$$F(u) = Au + B(u),$$

where A has the maximum principle property and B is monotone increasing (see Section 10.1).

10.5.3 Use the general residual indicator given by Theorem 10.5.4 to derive a residual indicator for

$$-\nabla \cdot (\epsilon \nabla u) = f \text{ in } \Omega, \quad u = 0 \text{ on } \partial\Omega, \quad \epsilon > 0.$$

10.5.4 Use the general residual indicator given by Theorem 10.5.4 to derive a residual indicator for

$$-\nabla \cdot (\epsilon \nabla u) + bu = f \text{ in } \Omega, \quad \epsilon \nabla u \cdot n = g \text{ on } \partial\Omega, \quad \epsilon, b > 0.$$

10.6 ITERATIVE METHODS FOR DISCRETIZED LINEAR EQUATIONS

In this section we give a survey of classical and modern techniques for iterative solution of linear systems involving matrices arising from any of the discretization techniques considered earlier in this chapter. Our focus will be primarily on fast (optimal or nearly optimal complexity) linear solvers based on multilevel and domain decomposition methods. Our goal here is to develop a basic understanding of the structure of modern optimal and near-optimal complexity methods based on space and/or frequency decompositions, including domain decomposition and multilevel methods. To this end, we first review some basic concepts and tools involving self-adjoint linear operators on a finite-dimensional Hilbert space. The results required for the analysis of linear methods, as well as conjugate gradient methods, are summarized. We then develop carefully the theory of classical linear methods for operator equations. The conjugate gradient method is then considered, and the relationship between the convergence rate of linear methods as preconditioners and the convergence rate of the resulting preconditioned conjugate gradient method is explored in some detail. We then consider linear two-level and multilevel methods as recursive algorithms, and examine various forms of the error propagator that have been key tools for unlocking a complete theoretical understanding of these methods over the last 20 years.

Since our focus has now turned to linear (and in Section 10.7, nonlinear) algebraic systems in finite-dimensional spaces, a brief remark about notation is in order. When

we encountered a sequence in a general Banach space X earlier in the chapter, we used a fairly standard notation to denote the sequence, $\{u_j\}_{j=1}^{\infty}$, with j the sequence index. Now that we will be working entirely with sequences in finite-dimensional spaces, it is standard to use a subscript to refer to a particular component of a vector in $\mathbb{R}^n$. Moreover, it will be helpful to use a subscript on a matrix or vector to refer to a particular discrete space when dealing with multiple spaces. Therefore, rather than keep track of three distinct subscripts when we encounter sequences of vectors in multiple discrete spaces, we will place the sequence index as a superscript, for example, $\{u^j\}_{j=1}^{\infty}$. There will be no danger of confusion with the exponentiation operator, as this convention is only used on vectors in a finite-dimensional vector space analogous to $\mathbb{R}^n$. When encountering a sequence of real numbers, such as the coefficients in an expansion of a finite-dimensional basis $\{u^j\}_{j=1}^{n}$, we will continue to denote the sequence using subscripts for the index, such as $\{c_j\}_{j=1}^{n}$. The expression for the expansion would then be $u = \sum_{j=1}^{n} c_j u^j$.

Linear Iterative Methods

When finite element, wavelet, spectral, finite volume, or other standard methods are used to discretize the second-order linear elliptic partial differential equation $Au = f$, a set of linear algebraic equations results, which we denote as

$$A_k u_k = f_k. \tag{10.6.1}$$

The subscript k denotes the discretization level, with larger k corresponding to a more refined mesh, and with an associated mesh parameter h_k representing the diameter of the largest element or volume in the mesh Ω_k. For a self-adjoint strongly elliptic partial differential operator, the matrix A_k produced by finite element and other discretizations is SPD. In this section we are primarily interested in linear iterations for solving the matrix equation (10.6.1) which have the general form

$$u_k^{i+1} = (I - B_k A_k) u_k^i + B_k f_k, \tag{10.6.2}$$

where B_k is an SPD matrix approximating A_k^{-1} in some sense. The classical stationary linear methods fit into this framework, as well as domain decomposition methods and multigrid methods. We will also make use of nonlinear iterations such as the conjugate gradient method, but primarily as a way to improve the performance of an underlying linear iteration.

Linear Operators, Spectral Bounds, and Condition Numbers. We briefly compile some material on self-adjoint linear operators in finite-dimensional spaces which will be used throughout the section. (See Chapters 4 and 5 for a more lengthy and more general exposition.) Let $\mathcal{H}$, $\mathcal{H}_1$, and $\mathcal{H}_2$ be real finite-dimensional Hilbert spaces equipped with the inner product $(\cdot,\cdot)$ inducing the norm $\|\cdot\| = (\cdot,\cdot)^{1/2}$. Since we are concerned only with finite-dimensional spaces, a Hilbert space $\mathcal{H}$ can be thought of as the Euclidean space $\mathbb{R}^n$; however, the preliminary material below and the algorithms we develop are phrased in terms of the unspecified space $\mathcal{H}$, so

that the algorithms may be interpreted directly in terms of finite element spaces as well.

If the operator $A\colon \mathcal{H}_1 \to \mathcal{H}_2$ is linear, we denote this as $A \in \mathcal{L}(\mathcal{H}_1, \mathcal{H}_2)$. The *(Hilbert) adjoint* of a linear operator $A \in \mathcal{L}(\mathcal{H}, \mathcal{H})$ with respect to $(\cdot, \cdot)$ is the unique operator A^T satisfying $(Au, v) = (u, A^T v)\,,\ \forall u, v \in \mathcal{H}$. An operator A is called *self-adjoint* or *symmetric* if $A = A^T$; a self-adjoint operator A is called *positive definite* or simply *positive* if $(Au, u) > 0, \forall u \in \mathcal{H},\ u \neq 0$.

If A is self-adjoint positive definite (SPD) with respect to $(\cdot, \cdot)$, then the bilinear form $A(u, v) = (Au, v)$ defines another inner product on $\mathcal{H}$, which we sometimes denote as $(\cdot, \cdot)_A = A(\cdot, \cdot)$ to emphasize the fact that it is an inner product rather than simply a bilinear form. The A-inner product then induces the A-norm in the usual way: $\|\cdot\|_A = (\cdot, \cdot)_A^{1/2}$. For each inner product the Cauchy-Schwarz inequality holds:

$$|(u, v)| \leqslant (u, u)^{1/2}(v, v)^{1/2}, \qquad |(u, v)_A| \leqslant (u, u)_A^{1/2}(v, v)_A^{1/2}, \qquad \forall u, v \in \mathcal{H}.$$

The adjoint of an operator M with respect to $(\cdot, \cdot)_A$, the *A-adjoint*, is the unique operator M^* satisfying $(Mu, v)_A = (u, M^* v)_A, \forall u, v \in \mathcal{H}$. From this definition it follows that

$$M^* = A^{-1} M^T A. \tag{10.6.3}$$

An operator M is called *A-self-adjoint* if $M = M^*$, and it is called *A-positive* if $(Mu, u)_A > 0, \forall u \in \mathcal{H}, u \neq 0$.

If $N \in \mathcal{L}(\mathcal{H}_1, \mathcal{H}_2)$, then the adjoint satisfies $N^T \in \mathcal{L}(\mathcal{H}_2, \mathcal{H}_1)$ and relates the inner products in $\mathcal{H}_1$ and $\mathcal{H}_2$ as follows:

$$(Nu, v)_{\mathcal{H}_2} = (u, N^T v)_{\mathcal{H}_1}\ , \quad \forall u \in \mathcal{H}_1\ , \quad \forall v \in \mathcal{H}_2\ .$$

Since it is usually clear from the arguments which inner product is involved, we shall drop the subscripts on inner products (and norms) throughout the section, except when necessary to avoid confusion.

For the operator M we denote the eigenvalues satisfying $Mu_i = \lambda_i u_i$ for eigenfunctions $u_i \neq 0$ as $\lambda_i(M)$. The spectral theory for self-adjoint linear operators states that the eigenvalues of the self-adjoint operator M are real and lie in the closed interval $[\lambda_{\min}(M), \lambda_{\max}(M)]$ defined by the Rayleigh quotients:

$$\lambda_{\min}(M) = \min_{u \neq 0} \frac{(Mu, u)}{(u, u)}, \qquad \lambda_{\max}(M) = \max_{u \neq 0} \frac{(Mu, u)}{(u, u)}.$$

Similarly, if an operator M is A-self-adjoint, then the eigenvalues are real and lie in the interval defined by the Rayleigh quotients generated by the A-inner product:

$$\lambda_{\min}(M) = \min_{u \neq 0} \frac{(Mu, u)_A}{(u, u)_A}, \qquad \lambda_{\max}(M) = \max_{u \neq 0} \frac{(Mu, u)_A}{(u, u)_A}.$$

We denote the set of eigenvalues as the spectrum $\sigma(M)$ and the largest of these in absolute value as the spectral radius as $\rho(M) = \max(|\lambda_{\min}(M)|, |\lambda_{\max}(M)|)$. For SPD (or A-SPD) operators M, the eigenvalues of M are real and positive, and the

powers M^s for real s are well-defined through the spectral decomposition; see, for example, [89]. Finally, recall that a matrix representing the operator M with respect to any basis for $\mathcal{H}$ has the same eigenvalues as the operator M.

Linear operators on finite-dimensional spaces are bounded, and these bounds define the operator norms induced by the norms $\|\cdot\|$ and $\|\cdot\|_A$:

$$\|M\| = \max_{u\neq 0} \frac{\|Mu\|}{\|u\|}, \qquad \|M\|_A = \max_{u\neq 0} \frac{\|Mu\|_A}{\|u\|_A}.$$

A well-known property is that if M is self-adjoint, then $\rho(M) = \|M\|$. This property can also be shown to hold for A-self-adjoint operators. The following lemma can be found in [7] (as Lemma 4.1), although the proof there is for A-normal matrices rather than A-self-adjoint operators.

Lemma 10.6.1. *If A is SPD and M is A-self-adjoint, then $\|M\|_A = \rho(M)$.*

Proof. We simply note that

$$\|M\|_A = \max_{u\neq 0} \frac{\|Mu\|_A}{\|u\|_A} = \max_{u\neq 0} \frac{(Mu, Mu)_A^{1/2}}{(u,u)_A^{1/2}} = \max_{u\neq 0} \frac{(M^*Mu, u)_A^{1/2}}{(u,u)_A^{1/2}}$$
$$= \lambda_{\max}^{1/2}(M^*M),$$

since M^*M is always A-self-adjoint. Since by assumption M itself is A-self-adjoint, we have that $M^* = M$, which yields $\|M\|_A = \lambda_{\max}^{1/2}(M^*M) = \lambda_{\max}^{1/2}(M^2) = (\max_i[\lambda_i^2(M)])^{1/2} = \max[|\lambda_{\min}(M)|, |\lambda_{\max}(M)|] = \rho(M)$. □

Finally, we define the *A-condition number* of an invertible operator M by extending the standard notion to the A-inner product:

$$\kappa_A(M) = \|M\|_A \|M^{-1}\|_A.$$

In Lemma 10.6.9 we will show that if M is an A-self-adjoint operator, then in fact the following simpler expression holds for the generalized condition number:

$$\kappa_A(M) = \frac{\lambda_{\max}(M)}{\lambda_{\min}(M)}.$$

The Basic Linear Method and Its Error Propagator. Assume that we are faced with the operator equation $Au = f$, where $A \in \mathcal{L}(\mathcal{H}, \mathcal{H})$ is SPD, and we desire the unique solution u. Given a *preconditioner* (an approximate inverse operator) $B \approx A^{-1}$, consider the equivalent *preconditioned system* $BAu = Bf$. The operator B is chosen so that the simple linear iteration

$$u^1 = u^0 - BAu^0 + Bf = (I - BA)u^0 + Bf,$$

which produces an improved approximation u^1 to the true solution u given an initial approximation u^0, has some desired convergence properties. This yields the following basic linear iterative method, which we study in the remainder of this section.

Algorithm 10.6.1 (Basic Linear Method for Solving $Au = f$).

`Form` u^{i+1} `from` u^i `using the affine fixed point iteration:`

$$u^{i+1} = u^i + B(f - Au^i) = (I - BA)u^i + Bf.$$

Subtracting the iteration equation from the identity $u = u - BAu + Bf$ yields the equation for the error $e^i = u - u^i$ at each iteration:

$$e^{i+1} = (I - BA)e^i = (I - BA)^2 e^{i-1} = \cdots = (I - BA)^{i+1} e^0. \tag{10.6.4}$$

The convergence of Algorithm 10.6.1 is determined completely by the spectral radius of the error propagation operator $E = I - BA$.

Theorem 10.6.1. *The condition $\rho(I - BA) < 1$ is necessary and sufficient for convergence of Algorithm 10.6.1 for an arbitrary initial approximation $u^0 \in \mathcal{H}$.*

Proof. See, for example, [115] or [169]. □

Since $|\lambda|\|u\| = \|\lambda u\| = \|Mu\| \leqslant \|M\|\,\|u\|$ for any norm $\|\cdot\|$, it follows that $\rho(M) \leqslant \|M\|$ for all norms $\|\cdot\|$. Therefore, $\|I - BA\| < 1$ and $\|I - BA\|_A < 1$ are both sufficient conditions for convergence of Algorithm 10.6.1. In fact, it is the norm of the error propagation operator which will bound the reduction of the error at each iteration, which follows from (10.6.4):

$$\|e^{i+1}\|_A \leqslant \|I - BA\|_A \|e^i\|_A \leqslant \|I - BA\|_A^{i+1} \|e^0\|_A. \tag{10.6.5}$$

The spectral radius $\rho(E)$ of the error propagator E is called the *convergence factor* for Algorithm 10.6.1, whereas the norm of the error propagator $\|E\|$ is referred to as the *contraction number* (with respect to the particular choice of norm $\|\cdot\|$).

We now establish some simple properties of the error propagation operator of an abstract linear method. We note that several of these properties are commonly used, especially in the multigrid literature, although the short proofs of the results seem difficult to locate. The particular framework we construct here for analyzing linear methods is based on the work of Xu [178] and the papers referenced therein, on the text by Varga [169], and on [100].

An alternative sufficient condition for convergence of the basic linear method is given in the following lemma, which is similar to *Stein's Theorem* (see [139] or [184]).

Lemma 10.6.2. *If E^* is the A-adjoint of E, and if the operator $I - E^*E$ is A-positive, then $\rho(E) \leqslant \|E\|_A < 1$.*

Proof. By hypothesis, $(A(I - E^*E)u, u) > 0\ \forall u \in \mathcal{H}$. This then implies that $(AE^*Eu, u) < (Au, u)\ \forall u \in \mathcal{H}$, or $(AEu, Eu) < (Au, u)\ \forall u \in \mathcal{H}$. But this last inequality implies that

$$\rho(E) \leqslant \|E\|_A = \left(\max_{u \neq 0} \frac{(AEu, Eu)}{(Au, u)} \right)^{1/2} < 1.$$

□

We now state three very simple lemmas that we use repeatedly in the following sections.

Lemma 10.6.3. *If A is SPD, then BA is A-self-adjoint if and only if B is self-adjoint.*

Proof. Simply note that $(ABAx, y) = (BAx, Ay) = (Ax, B^T Ay)\ \forall x, y \in \mathcal{H}$. The lemma follows since $BA = B^T A$ if and only if $B = B^T$. □

Lemma 10.6.4. *If A is SPD, then $I - BA$ is A-self-adjoint if and only if B is self-adjoint.*

Proof. Begin by noting that $(A(I - BA)x, y) = (Ax, y) - (ABAx, y) = (Ax, y) - (Ax, (BA)^* y) = (Ax, (I - (BA)^*)y),\ \forall x, y \in \mathcal{H}$. Therefore, $E^* = I - (BA)^* = I - BA = E$ if and only if $BA = (BA)^*$. But by Lemma 10.6.3, this holds if and only if B is self-adjoint, so the result follows. □

Lemma 10.6.5. *If A and B are SPD, then BA is A-SPD.*

Proof. By Lemma 10.6.3, BA is A-self-adjoint. Since B is SPD, and since $Au \neq 0$ for $u \neq 0$, we have $(ABAu, u) = (BAu, Au) > 0, \forall u \neq 0$. Therefore, BA is also A-positive, and the result follows. □

We noted above that the property $\rho(M) = \|M\|$ holds in the case that M is self-adjoint with respect to the inner product inducing the norm $\|\cdot\|$. If B is self-adjoint, the following theorem states that the resulting error propagator $E = I - BA$ has this property with respect to the A-norm.

Theorem 10.6.2. *If A is SPD and B is self-adjoint, then $\|I - BA\|_A = \rho(I - BA)$.*

Proof. By Lemma 10.6.4, $I - BA$ is A-self-adjoint, and by Lemma 10.6.1, the result follows. □

REMARK. Theorem 10.6.2 will be exploited later since $\rho(E)$ is usually much easier to compute numerically than $\|E\|_A$, and since it is the energy norm $\|E\|_A$ of the error propagator E which is typically bounded in various convergence theories for iterative processes.

The following simple lemma, similar to Lemma 10.6.2, will be very useful later.

Lemma 10.6.6. *If A and B are SPD, and if the operator $E = I - BA$ is A-nonnegative, then $\rho(E) = \|E\|_A < 1$.*

Proof. By Lemma 10.6.4, E is A-self-adjoint. By assumption, E is A-nonnegative, so from the discussion earlier in the section we see that E must have real nonnegative eigenvalues. By hypothesis, $(A(I - BA)u, u) \geqslant 0\ \forall u \in \mathcal{H}$, which implies that $(ABAu, u) \leqslant (Au, u)\ \forall u \in \mathcal{H}$. By Lemma 10.6.5, BA is A-SPD, and we have that

$$0 < (ABAu, u) \leqslant (Au, u) \quad \forall u \in \mathcal{H},\ \ u \neq 0,$$

which implies that $0 < \lambda_i(BA) \leqslant 1\ \forall \lambda_i \in \sigma(BA)$. Thus, since we also have that $\lambda_i(E) = \lambda_i(I - BA) = 1 - \lambda_i(BA)\ \forall i$, we have

$$\rho(E) = \max_i \lambda_i(E) = 1 - \min_i \lambda_i(BA) < 1.$$

Finally, by Theorem 10.6.2, we have $\|E\|_A = \rho(E) < 1$. □

The following simple lemma relates the contraction number bound to two simple inequalities; it is a standard result which follows directly from the spectral theory of self-adjoint linear operators.

Lemma 10.6.7. *If A is SPD and B is self-adjoint, and $E = I - BA$ is such that*

$$-C_1(Au, u) \leqslant (AEu, u) \leqslant C_2(Au, u), \quad \forall u \in \mathcal{H},$$

for $C_1 \geqslant 0$ and $C_2 \geqslant 0$, then $\rho(E) = \|E\|_A \leqslant \max\{C_1, C_2\}$.

Proof. By Lemma 10.6.4, $E = I - BA$ is A-self-adjoint, and by the spectral theory outlined at the beginning of the earlier section on linear iterative methods, the inequality above simply bounds the most negative and most positive eigenvalues of E with $-C_1$ and C_2, respectively. The result then follows by Theorem 10.6.2. □

Corollary 10.6.1. *If A and B are SPD, then Lemma 10.6.7 holds for some $C_2 < 1$.*

Proof. By Lemma 10.6.5, BA is A-SPD, which implies that the eigenvalues of BA are real and positive by the discussion earlier in the section. By Lemma 10.6.4, $E = I - BA$ is A-self-adjoint, and therefore has real eigenvalues. The eigenvalues of E and BA are related by $\lambda_i(E) = \lambda_i(I - BA) = 1 - \lambda_i(BA)\ \forall i$, and since $\lambda_i(BA) > 0\ \forall i$, we must have that $\lambda_i(E) < 1\ \forall i$. Since C_2 in Lemma 10.6.7 bounds the largest positive eigenvalue of E, we have that $C_2 < 1$. □

Convergence Properties of the Linear Method. The generalized condition number κ_A is employed in the following lemma, which states that there is an optimal relaxation parameter for a basic linear method, and gives the best possible convergence estimate for the method employing the optimal parameter. This lemma has appeared many times in the literature in one form or another; see [141].

Lemma 10.6.8. *If A and B are SPD, then*

$$\rho(I - \alpha BA) = \|I - \alpha BA\|_A < 1$$

if and only if $\alpha \in (0, 2/\rho(BA))$. Convergence is optimal (the norm is minimized) when $\alpha = 2/[\lambda_{\min}(BA) + \lambda_{\max}(BA)]$, giving

$$\rho(I - \alpha BA) = \|I - \alpha BA\|_A = 1 - \frac{2}{1 + \kappa_A(BA)} < 1.$$

Proof. Note that $\rho(I - \alpha BA) = \max_\lambda |1 - \alpha\lambda(BA)|$, so that $\rho(I - \alpha BA) < 1$ if and only if $\alpha \in (0, 2/\rho(BA))$, proving the first part of the lemma. We now take $\alpha = 2/[\lambda_{\min}(BA) + \lambda_{\max}(BA)]$, which gives

$$\begin{aligned}\rho(I - \alpha BA) &= \max_\lambda |1 - \alpha\lambda(BA)| = \max_\lambda (1 - \alpha\lambda(BA)) \\ &= \max_\lambda \left(1 - \frac{2\lambda(BA)}{\lambda_{\min}(BA) + \lambda_{\max}(BA)}\right) \\ &= 1 - \frac{2\lambda_{\min}(BA)}{\lambda_{\min}(BA) + \lambda_{\max}(BA)} \\ &= 1 - \frac{2}{1 + \dfrac{\lambda_{\max}(BA)}{\lambda_{\min}(BA)}}.\end{aligned}$$

Since BA is A-self-adjoint, by Lemma 10.6.9 we have that the condition number is $\kappa_A(BA) = \lambda_{\max}(BA)/\lambda_{\min}(BA)$, so that if $\alpha = 2/[\lambda_{\min}(BA) + \lambda_{\max}(BA)]$, then

$$\rho(I - \alpha BA) = \|I - \alpha BA\|_A = 1 - \frac{2}{1 + \kappa_A(BA)}.$$

To show that this is optimal, we must solve the mini-max problem: $\min_\alpha[\max_\lambda |1 - \alpha\lambda|]$, where $\alpha \in (0, 2/\lambda_{\max})$. Note that each α defines a polynomial of degree zero in λ, namely $P_o(\lambda) = \alpha$. Therefore, we can rephrase the problem as

$$P_1^{\text{opt}}(\lambda) = \min_{P_o} \left[\max_\lambda |1 - \lambda P_o(\lambda)|\right].$$

It is well-known that the scaled and shifted Chebyshev polynomials give the solution to this "mini-max" problem (see Exercise 10.5.2):

$$P_1^{\text{opt}}(\lambda) = 1 - \lambda P_o^{\text{opt}} = \frac{T_1\left(\dfrac{\lambda_{\max} + \lambda_{\min} - 2\lambda}{\lambda_{\max} - \lambda_{\min}}\right)}{T_1\left(\dfrac{\lambda_{\max} + \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}}\right)}.$$

Since $T_1(x) = x$, we have simply that

$$P_1^{\text{opt}}(\lambda) = \frac{\dfrac{\lambda_{\max} + \lambda_{\min} - 2\lambda}{\lambda_{\max} - \lambda_{\min}}}{\dfrac{\lambda_{\max} + \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}}} = 1 - \lambda\left(\frac{2}{\lambda_{\min} + \lambda_{\max}}\right),$$

showing that, in fact, $\alpha_{\text{opt}} = 2/[\lambda_{\min} + \lambda_{\max}]$. □

Note that if we wish to reduce the initial error $\|e^0\|_A$ by the factor ϵ, then equation (10.6.5) implies that this will be guaranteed if

$$\|E\|_A^{i+1} \leqslant \epsilon.$$

Taking natural logarithms of both sides and solving for i (where we assume that $\epsilon < 1$), we see that the number of iterations required to reach the desired tolerance, as a function of the contraction number, is given by

$$i \geqslant \frac{|\ln \epsilon|}{|\ln \|E\|_A|}. \tag{10.6.6}$$

If the bound on the norm is of the form in Lemma 10.6.8, then to achieve a tolerance of ϵ after i iterations will require that

$$i \geqslant \frac{|\ln \epsilon|}{\left|\ln\left(1 - \dfrac{2}{1+\kappa_A(BA)}\right)\right|} = \frac{|\ln \epsilon|}{\left|\ln\left(\dfrac{\kappa_A(BA)-1}{\kappa_A(BA)+1}\right)\right|}. \tag{10.6.7}$$

Using the approximation

$$\begin{aligned} \ln\left(\frac{a-1}{a+1}\right) = \ln\left(\frac{1+(-1/a)}{1-(-1/a)}\right) &= 2\left[\left(\frac{-1}{a}\right) + \frac{1}{3}\left(\frac{-1}{a}\right)^3 + \frac{1}{5}\left(\frac{-1}{a}\right)^5 + \cdots\right] \\ &< \frac{-2}{a}, \end{aligned} \tag{10.6.8}$$

we have $|\ln[(\kappa_A(BA)-1)/(\kappa_A(BA)+1)]| > 2/\kappa_A(BA)$. Thus, we can guarantee (10.6.7) holds by enforcing

$$i \geqslant \frac{1}{2}\kappa_A(BA)|\ln \epsilon| + 1.$$

Therefore, the number of iterations required to reach an error on the order of the tolerance ϵ is then

$$i = \mathcal{O}\left(\kappa_A(BA)|\ln \epsilon|\right).$$

If a single iteration of the method costs $\mathcal{O}(N)$ arithmetic operations, then the overall complexity to solve the problem is $\mathcal{O}(|\ln \|E\|_A|^{-1} N|\ln \epsilon|)$, or $\mathcal{O}(\kappa_A(BA)N|\ln \epsilon|)$. If the quantity $\|E\|_A$ can be bounded by a constant which is less than 1, where the constant is independent of N, or alternatively, if $\kappa_A(BA)$ can be bounded by a constant which is independent of N, then the complexity is near optimal $\mathcal{O}(N|\ln \epsilon|)$.

Note that if E is A-self-adjoint, then we can replace $\|E\|_A$ by $\rho(E)$ in the discussion above. Even when this is not the case, $\rho(E)$ is often used above in place of $\|E\|_A$ to obtain an estimate, and the quantity $R_\infty(E) = -\ln \rho(E)$ is referred to as the *asymptotic convergence rate* (see [169, 184]). In [169], the *average rate of convergence of m iterations* is defined as the quantity $R(E^m) = -\ln(\|E^m\|^{1/m})$, the meaning of which is intuitively clear from equation (10.6.5). Since we have that $\rho(E) = \lim_{m\to\infty} \|E^m\|^{1/m}$ for all bounded linear operators E and norms $\|\cdot\|$ (see [116]), it then follows that $\lim_{m\to\infty} R(E^m) = R_\infty(E)$. While $R_\infty(E)$ is considered the standard measure of convergence of linear iterations (it is called the "convergence rate"; see [184]), this is really an asymptotic measure, and the convergence behavior for the early iterations may be better monitored by using the norm of the propagator E directly in (10.6.6); an example is given in [169], for which $R_\infty(E)$ gives a poor estimate of the number of iterations required.

The Conjugate Gradient Method

Consider now the linear equation $Au = f$ in the space $\mathcal{H}$. The conjugate gradient method was developed by Hestenes and Stiefel [92] for linear systems with symmetric positive definite operators A. It is common to *precondition* the linear system by the SPD *preconditioning operator* $B \approx A^{-1}$, in which case the generalized or preconditioned conjugate gradient method results. Our purpose in this section is to briefly examine the algorithm, its contraction properties, and establish some simple relationships between the contraction number of a basic linear preconditioner and that of the resulting preconditioned conjugate gradient algorithm. These relationships are commonly used, but some of the short proofs seem unavailable.

In [8], a general class of conjugate gradient methods obeying three-term recursions is studied, and it is shown that each instance of the class can be characterized by three operators: an inner product operator X, a preconditioning operator Y, and the system operator Z. As such, these methods are denoted as CG(X,Y,Z). We are interested in the special case that $X = A$, $Y = B$, and $Z = A$, when both B and A are SPD. Choosing the *Omin* [8] algorithm to implement the method CG(A,B,A), the *preconditioned conjugate gradient method* results. In order to present the algorithm, which is more complex than the basic linear method (Algorithm 10.6.1), we will employ some standard notation from the algorithm literature. In particular, we will denote the start of a complex fixed point-type iteration involving multiple steps using the standard notion of a "Do"-loop, where the beginning of the loop, as well as its duration, is denoted with a "Do X" statement, where X represents the conditions for continuing or terminating the loop. The end of the complex iteration will be denoted simply by "End do."

Algorithm 10.6.2 (Preconditioned Conjugate Gradient Algorithm).

Let $u^0 \in \mathcal{H}$ be given.
$r^0 = f - Au^0$, $s^0 = Br^0$, $p^0 = s^0$.
Do $i = 0, 1, \ldots$ until convergence:
  $\alpha_i = (r^i, s^i)/(Ap^i, p^i)$
  $u^{i+1} = u^i + \alpha_i p^i$
  $r^{i+1} = r^i - \alpha_i Ap^i$
  $s^{i+1} = Br^{i+1}$
  $\beta_{i+1} = (r^{i+1}, s^{i+1})/(r^i, s^i)$
  $p^{i+1} = s^{i+1} + \beta_{i+1} p^i$
End do.

If the dimension of $\mathcal{H}$ is n, then the algorithm can be shown to converge in n steps since the preconditioned operator BA is A-SPD [8]. Note that if $B = I$, then this algorithm is exactly the Hestenes and Stiefel algorithm.

Convergence Properties of the Conjugate Gradient Method. Since we wish to understand a little about the convergence properties of the conjugate gradient method and how these will be affected by a linear method representing the preconditioner B, we will briefly review a well-known conjugate gradient contraction bound. To begin, it is not difficult to see that the error at each iteration of Algorithm 10.6.2

can be written as a polynomial in BA times the initial error:

$$e^{i+1} = [I - BAp_i(BA)]e^0,$$

where $p_i \in \mathcal{P}_i$, the space of polynomials of degree i. At each step the energy norm of the error $\|e^{i+1}\|_A = \|u - u^{i+1}\|_A$ is minimized over the *Krylov subspace*:

$$K_{i+1}(BA, Br^0) = \text{span}\{Br^0, (BA)Br^0, (BA)^2Br^0, \ldots, (BA)^iBr^0\}.$$

Therefore,

$$\|e^{i+1}\|_A = \min_{p_i \in \mathcal{P}_i} \|[I - BAp_i(BA)]e^0\|_A.$$

Since BA is A-SPD, the eigenvalues $\lambda_j \in \sigma(BA)$ of BA are real and positive, and the eigenvectors v_j of BA are A-orthonormal. By expanding $e^0 = \sum_{j=1}^n \alpha_j v_j$, we have

$$\begin{aligned}
\|[I - BAp_i(BA)]e^0\|_A^2 &= (A[I - BAp_i(BA)]e^0, [I - BAp_i(BA)]e^0) \\
&= (A[I - BAp_i(BA)] \\
&\qquad \cdot (\sum_{j=1}^n \alpha_j v_j), [I - BAp_i(BA)](\sum_{j=1}^n \alpha_j v_j)) \\
&= (\sum_{j=1}^n [1 - \lambda_j p_i(\lambda_j)]\alpha_j \lambda_j v_j, \sum_{j=1}^n [1 - \lambda_j p_i(\lambda_j)]\alpha_j v_j) \\
&= \sum_{j=1}^n [1 - \lambda_j p_i(\lambda_j)]^2 \alpha_j^2 \lambda_j \\
&\leqslant \max_{\lambda_j \in \sigma(BA)} [1 - \lambda_j p_i(\lambda_j)]^2 \sum_{j=1}^n \alpha_j^2 \lambda_j \\
&= \max_{\lambda_j \in \sigma(BA)} [1 - \lambda_j p_i(\lambda_j)]^2 \sum_{j=1}^n (A\alpha_j v_j, \alpha_j v_j) \\
&= \max_{\lambda_j \in \sigma(BA)} [1 - \lambda_j p_i(\lambda_j)]^2 (A \sum_{j=1}^n \alpha_j v_j, \sum_{j=1}^n \alpha_j v_j) \\
&= \max_{\lambda_j \in \sigma(BA)} [1 - \lambda_j p_i(\lambda_j)]^2 \|e^0\|_A^2.
\end{aligned}$$

Thus, we have that

$$\|e^{i+1}\|_A \leqslant \left(\min_{p_i \in \mathcal{P}_i} \left[\max_{\lambda_j \in \sigma(BA)} |1 - \lambda_j p_i(\lambda_j)| \right] \right) \|e^0\|_A.$$

The scaled and shifted Chebyshev polynomials $T_{i+1}(\lambda)$, extended outside the interval $[-1, 1]$ as in Appendix A of [12], yield a solution to this *mini-max* problem (see

Exercises 10.5.2 and 10.5.3). Using some simple well-known relationships valid for $T_{i+1}(\cdot)$, the following contraction bound is easily derived:

$$\|e^{i+1}\|_A \leqslant 2 \left(\frac{\sqrt{\dfrac{\lambda_{\max}(BA)}{\lambda_{\min}(BA)}} - 1}{\sqrt{\dfrac{\lambda_{\max}(BA)}{\lambda_{\min}(BA)}} + 1} \right)^{i+1} \|e^0\|_A = 2\, \delta_{\mathrm{cg}}^{i+1}\, \|e^0\|_A. \tag{10.6.9}$$

The ratio of the extreme eigenvalues of BA appearing in the bound is often mistakenly called the (spectral) condition number $\kappa(BA)$; in fact, since BA is not self-adjoint (it is A-self-adjoint), this ratio is not in general equal to the condition number (this point is discussed in detail in [7]). However, the ratio does yield a condition number in a different norm. The following lemma is a special case of a more general result [7].

Lemma 10.6.9. *If A and B are SPD, then*

$$\kappa_A(BA) = \|BA\|_A \|(BA)^{-1}\|_A = \frac{\lambda_{\max}(BA)}{\lambda_{\min}(BA)}. \tag{10.6.10}$$

Proof. For any A-SPD M, it is easy to show that M^{-1} is also A-SPD, so from the material in the earlier section on linear iterative methods we know that both M and M^{-1} have real, positive eigenvalues. From Lemma 10.6.1 it then holds that

$$\begin{aligned} \|M^{-1}\|_A = \rho(M^{-1}) &= \max_{u \neq 0} \frac{(AM^{-1}u, u)}{(Au, u)} = \max_{u \neq 0} \frac{(AM^{-1/2}u, M^{-1/2}u)}{(AMM^{-1/2}u, M^{-1/2}u)} \\ &= \max_{v \neq 0} \frac{(Av, v)}{(AMv, v)} = \left[\min_{v \neq 0} \frac{(AMv, v)}{(Av, v)} \right]^{-1} = \lambda_{\min}(M)^{-1}. \end{aligned}$$

By Lemma 10.6.5, BA is A-SPD, which together with Lemma 10.6.1 implies that $\|BA\|_A = \rho(BA) = \lambda_{\max}(BA)$. We have then $\|(BA)^{-1}\|_A = \lambda_{\min}(BA)^{-1}$, implying that the A-condition number is given as the ratio of the extreme eigenvalues of BA as in equation (10.6.10). □

More generally, it can be shown that if the operator D is C-normal for some SPD inner product operator C, then the generalized condition number given by the expression $\kappa_C(D) = \|D\|_C \|D^{-1}\|_C$ is equal to the ratio of the extreme eigenvalues of the operator D. A proof of this fact is given in [7], along with a detailed discussion of this and other relationships for more general conjugate gradient methods. The conjugate gradient contraction number δ_{cg} can now be written as

$$\delta_{\mathrm{cg}} = \frac{\sqrt{\kappa_A(BA)} - 1}{\sqrt{\kappa_A(BA)} + 1} = 1 - \frac{2}{1 + \sqrt{\kappa_A(BA)}}.$$

The following lemma is used in the analysis of multigrid and other linear preconditioners (it appears for example in [177]) to bound the condition number of the

operator BA in terms of the extreme eigenvalues of the linear preconditioner error propagator $E = I - BA$. We have given our own short proof of this result for completeness.

Lemma 10.6.10. *If A and B are SPD, and $E = I - BA$ is such that*

$$-C_1(Au, u) \leqslant (AEu, u) \leqslant C_2(Au, u), \quad \forall u \in \mathcal{H},$$

for $C_1 \geqslant 0$ and $C_2 \geqslant 0$, then the inequality above must in fact also hold with $C_2 < 1$, and it follows that

$$\kappa_A(BA) \leqslant \frac{1 + C_1}{1 - C_2}.$$

Proof. First, since A and B are SPD, by Corollary 10.6.1 we have that $C_2 < 1$. Since $(AEu, u) = (A(I - BA)u, u) = (Au, u) - (ABAu, u)$, $\forall u \in \mathcal{H}$, it is immediately clear that

$$-C_1(Au, u) - (Au, u) \leqslant -(ABAu, u) \leqslant C_2(Au, u) - (Au, u), \quad \forall u \in \mathcal{H}.$$

After multiplying by minus 1, we have

$$(1 - C_2)(Au, u) \leqslant (ABAu, u) \leqslant (1 + C_1)(Au, u), \quad \forall u \in \mathcal{H}.$$

By Lemma 10.6.5, BA is A-SPD, and it follows from the material in the section on linear iterative methods that the eigenvalues of BA are real and positive, and lie in the interval defined by the Rayleigh quotients generated by the A-inner product. From above, we see that the interval is given by $[(1-C_2), (1+C_1)]$, and by Lemma 10.6.9 the result follows. □

The next corollary may be found in [177].

Corollary 10.6.2. *If A and B are SPD, and BA is such that*

$$C_1(Au, u) \leqslant (ABAu, u) \leqslant C_2(Au, u), \quad \forall u \in \mathcal{H},$$

for $C_1 \geqslant 0$ and $C_2 \geqslant 0$, then the above must hold with $C_1 > 0$, and it follows that

$$\kappa_A(BA) \leqslant \frac{C_2}{C_1}.$$

Proof. This follows easily from the argument used in the proof of Lemma 10.6.10. □

The following corollary, which relates the contraction property of a linear method to the condition number of the operator BA, appears without proof in [178].

Corollary 10.6.3. *If A and B are SPD, and $\|I - BA\|_A \leqslant \delta < 1$, then*

$$\kappa_A(BA) \leqslant \frac{1 + \delta}{1 - \delta}. \tag{10.6.11}$$

Proof. This follows immediately from Lemma 10.6.10 with $\delta = \max\{C_1, C_2\}$. □

Preconditioners and the Acceleration of Linear Methods. We comment briefly on an interesting implication of Lemma 10.6.10, which was pointed out in [177]. It seems that even if a linear method is not convergent, for example if $C_1 > 1$ so that $\rho(E) > 1$, it may still be a good preconditioner. For example, if A and B are SPD, then by Corollary 10.6.1 we always have $C_2 < 1$. If it is the case that $C_2 << 1$, and if $C_1 > 1$ does not become too large, then $\kappa_A(BA)$ will be small and the conjugate gradient method will converge rapidly. A multigrid method (see below) will often diverge when applied to a problem with discontinuous coefficients unless special care is taken. Simply using the conjugate gradient method in conjunction with the multigrid method often yields a convergent (even rapidly convergent) method without employing any of the special techniques that have been developed for these problems; Lemma 10.6.10 gives some insight into this behavior.

The following result from [178] connects the contraction number of the linear method used as the preconditioner to the contraction number of the resulting conjugate gradient method, and it shows that the conjugate gradient method always accelerates a linear method, justifying the terminology "CG acceleration."

Theorem 10.6.3. *If A and B are SPD, and $\|I - BA\|_A \leqslant \delta < 1$, then $\delta_{\text{cg}} < \delta$.*

Proof. An abbreviated proof appears in [178]; we fill in the details here for completeness. Assume that the given linear method has contraction number bounded as $\|I - BA\|_A < \delta$. Now, since the function

$$\frac{\sqrt{\kappa_A(BA)} - 1}{\sqrt{\kappa_A(BA)} + 1}$$

is an increasing function of $\kappa_A(BA)$, we can use the result of Lemma 10.6.10, namely $\kappa_A(BA) \leqslant (1+\delta)/(1-\delta)$, to bound the contraction rate of preconditioned conjugate gradient method as follows:

$$\begin{aligned}\delta_{\text{cg}} &\leqslant \left(\frac{\sqrt{\kappa_A(BA)} - 1}{\sqrt{\kappa_A(BA)} + 1}\right) \leqslant \left(\frac{\sqrt{\dfrac{1+\delta}{1-\delta}} - 1}{\sqrt{\dfrac{1+\delta}{1-\delta}} + 1}\right) \cdot \left(\frac{\sqrt{\dfrac{1+\delta}{1-\delta}} - 1}{\sqrt{\dfrac{1+\delta}{1-\delta}} - 1}\right) \\ &= \frac{\dfrac{1+\delta}{1-\delta} - 2\sqrt{\dfrac{1+\delta}{1-\delta}} + 1}{\dfrac{1+\delta}{1-\delta} - 1} = \frac{1 - \sqrt{1-\delta^2}}{\delta}.\end{aligned}$$

Note that this last term can be rewritten as

$$\delta_{\text{cg}} \leqslant \frac{1 - \sqrt{1-\delta^2}}{\delta} = \delta\left(\frac{1}{\delta^2}[1 - \sqrt{1-\delta^2}]\right).$$

Now, since $0 < \delta < 1$, clearly $1 - \delta^2 < 1$, so that $1 - \delta^2 > (1-\delta^2)^2$. Thus, $\sqrt{1-\delta^2} > 1 - \delta^2$, or $-\sqrt{1-\delta^2} < \delta^2 - 1$, or finally, $1 - \sqrt{1-\delta^2} < \delta^2$. Therefore,

$(1/\delta^2)\left[1-\sqrt{1-\delta^2}\right] < 1$, or

$$\delta_{\mathrm{cg}} \leq \delta\left(\frac{1}{\delta^2}\left[1-\sqrt{1-\delta^2}\right]\right) < \delta.$$

A more direct proof follows by recalling from Lemma 10.6.8 that the *best* possible contraction of the linear method, when provided with an optimal parameter, is given by

$$\delta_{\mathrm{opt}} = 1 - \frac{2}{1+\kappa_A(BA)},$$

whereas the conjugate gradient contraction is

$$\delta_{\mathrm{cg}} = 1 - \frac{2}{1+\sqrt{\kappa_A(BA)}}.$$

Assuming that $B \neq A^{-1}$, then we always have $\kappa_A(BA) > 1$, so we must have that $\delta_{\mathrm{cg}} < \delta_{\mathrm{opt}} \leqslant \delta$. □

This result implies that it always pays in terms of an improved contraction number to use the conjugate gradient method to accelerate a linear method; the question remains, of course, whether the additional computational labor involved will be amortized by the improvement. This is not clear from the analysis above, and is problem dependent in practice.

Note that if a given linear method requires a parameter α as in Lemma 10.6.8 in order to be competitive, one can simply use the conjugate gradient method as an accelerator for the method without a parameter, avoiding the possibly costly estimation of a good parameter α. Theorem 10.6.3 guarantees that the resulting method will have superior contraction properties, without requiring the parameter estimation. This is exactly why additive multigrid and domain decomposition methods (which we discuss in more detail below) are used almost exclusively as preconditioners for conjugate gradient methods; in contrast to the multiplicative variants, which can be used effectively without a parameter, the additive variants always require a good parameter α to be effective, unless used as preconditioners.

To finish this section, we remark briefly on the complexity of Algorithm 10.6.2. If a tolerance of ϵ is required, then the computational cost to reduce the energy norm of the error below the tolerance can be determined from the expression above for δ_{cg} and from equation (10.6.9). To achieve a tolerance of ϵ after i iterations will require that

$$2\,\delta_{\mathrm{cg}}^{i+1} = 2\left(\frac{\sqrt{\kappa_A(BA)}-1}{\sqrt{\kappa_A(BA)}+1}\right)^{i+1} < \epsilon.$$

Dividing by 2 and taking natural logarithms (and assuming that $\epsilon < 1$) yields

$$i \geqslant \frac{\left|\ln\frac{\epsilon}{2}\right|}{\left|\ln\left(\frac{\sqrt{\kappa_A(BA)}-1}{\sqrt{\kappa_A(BA)}+1}\right)\right|}. \tag{10.6.12}$$

Using (10.6.8) we have $|\ln[(\kappa_A^{1/2}(BA)-1)/(\kappa_A^{1/2}(BA)+1)]| > 2/\kappa_A^{1/2}(BA)$. Thus, we can ensure that (10.6.12) holds by enforcing

$$i \geqslant \frac{1}{2}\kappa_A^{1/2}(BA)\left|\ln\frac{\epsilon}{2}\right| + 1.$$

Therefore, the number of iterations required to reach an error on the order of the tolerance ϵ is

$$i = \mathcal{O}\left(\kappa_A^{1/2}(BA)\left|\ln\frac{\epsilon}{2}\right|\right).$$

If the cost of each iteration is $\mathcal{O}(N)$, which will hold in the case of the sparse matrices generated by standard discretizations of elliptic partial differential equations, then the overall complexity to solve the problem is $\mathcal{O}(\kappa_A^{1/2}(BA)N|\ln[\epsilon/2]|)$. If the preconditioner B is such that $\kappa_A^{1/2}(BA)$ can be bounded independently of the problem size N, then the complexity becomes (near) optimal order $\mathcal{O}(N|\ln[\epsilon/2]|)$.

We make some final remarks regarding the idea of *spectral equivalence*.

Definition 10.6.1. *The SPD operators $A \in \mathcal{L}(\mathcal{H},\mathcal{H})$ and $M \in \mathcal{L}(\mathcal{H},\mathcal{H})$ are called spectrally equivalent if there exist constants $C_1 > 0$ and $C_2 > 0$ such that*

$$C_1(Au,u) \leqslant (Mu,u) \leqslant C_2(Au,u), \quad \forall u \in \mathcal{H}.$$

In other words, A defines an inner product which induces a norm equivalent to the norm induced by the M-inner product. If a given preconditioner B is spectrally equivalent to A^{-1}, then the condition number of the preconditioned operator BA is uniformly bounded.

Lemma 10.6.11. *If the SPD operators B and A^{-1} are spectrally equivalent, then*

$$\kappa_A(BA) \leqslant \frac{C_2}{C_1}.$$

Proof. By hypothesis, we have $C_1(A^{-1}u,u) \leq (Bu,u) \leq C_2(A^{-1}u,u)$, $\forall u \in \mathcal{H}$. But this can be written as

$$\begin{aligned} C_1(A^{-1/2}u, A^{-1/2}u) &\leqslant (A^{1/2}BA^{1/2}A^{-1/2}u, A^{-1/2}u) \\ &\leqslant C_2(A^{-1/2}u, A^{-1/2}u) \end{aligned}$$

or

$$C_1(\tilde{u},\tilde{u}) \leqslant (A^{1/2}BA^{1/2}\tilde{u},\tilde{u}) \leqslant C_2(\tilde{u},\tilde{u}), \quad \forall \tilde{u} \in \mathcal{H}.$$

Now, since $BA = A^{-1/2}(A^{1/2}BA^{1/2})A^{1/2}$, we have that BA is similar to the SPD operator $A^{1/2}BA^{1/2}$. Therefore, the inequality above bounds the extreme eigenvalues of BA, and as a result the lemma follows by Lemma 10.6.9. □

Moreover, if any of the following (equivalent) norm equivalences hold:

$$\begin{aligned}
C_1(Au,u) \leqslant \quad & (ABAu,u) \quad \leqslant C_2(Au,u),\\
C_1(Bu,u) \leqslant \quad & (BABu,u) \quad \leqslant C_2(Bu,u),\\
C_1(A^{-1}u,u) \leqslant \quad & (Bu,u) \quad \leqslant C_2(A^{-1}u,u),\\
C_1(B^{-1}u,u) \leqslant \quad & (Au,u) \quad \leqslant C_2(B^{-1}u,u),\\
C_2^{-1}(Au,u) \leqslant \quad & (B^{-1}u,u) \quad \leqslant C_1^{-1}(Au,u),\\
C_2^{-1}(Bu,u) \leqslant \quad & (A^{-1}u,u) \quad \leqslant C_1^{-1}(Bu,u),
\end{aligned}$$

then by similar arguments one has

$$\kappa_A(BA) \leqslant \frac{C_2}{C_1}.$$

Of course, since all norms on finite-dimensional spaces are equivalent (which follows from the fact that all linear operators on finite-dimensional spaces are bounded), the idea of spectral equivalence is only important in the case of infinite-dimensional spaces, or when one considers how the equivalence constants behave as one increases the sizes of the spaces. This is exactly the issue in multigrid and domain decomposition theory: As one decreases the mesh size (increases the size of the spaces involved), one would like the quantity $\kappa_A(BA)$ to remain uniformly bounded (in other words, one would like the equivalence constants to remain constant or grow only slowly). A discussion of these ideas appears in [141].

Domain Decomposition Methods

Domain decomposition methods were first proposed by H. A. Schwarz as a theoretical tool for studying elliptic problems on complicated domains, constructed as the union of simple domains. An interesting early reference not often mentioned is [109], containing both analysis and numerical examples and references to the original work by Schwarz. Since the development of parallel computers, domain decomposition methods have become one of the most important practical methods for solving elliptic partial differential equations on modern parallel computers. In this section we briefly describe basic overlapping domain decomposition methods; our discussion here draws much from [66, 100, 178] and the references cited therein.

Given a domain Ω and coarse triangulation by J regions $\{\Omega_k\}$ of mesh size H_k, we refine (several times) to obtain a fine mesh of size h_k. The regions defined by the initial triangulation Ω_k are then extended by δ_k to form the "overlapping subdomains" Ω_k'. Let $\mathcal{V}$ and $\mathcal{V}_0$ denote the finite element spaces associated with the h_k and H_k triangulation of Ω, respectively. Examples of overlapping subdomains constructed in this way over existing coarse simplicial meshes, designed for building piecewise-linear finite element subdomain spaces $\mathcal{V}_k = H_0^1(\Omega_k') \cap \mathcal{V}$, are shown in Figure 10.10.

To describe overlapping domain decomposition methods, we focus on the following variational problem in $\mathcal{V}$:

$$\text{Find } u \in \mathcal{V} \text{ such that } a(u,v) = f(v), \quad \forall v \in \mathcal{V}, \tag{10.6.13}$$

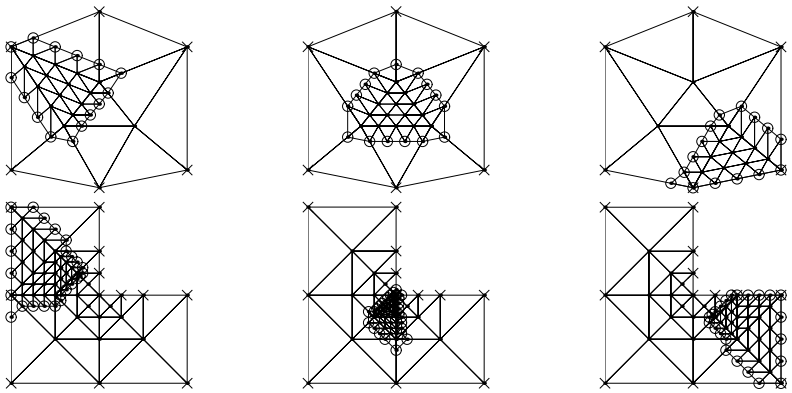

Figure 10.10 Unstructured overlapping subdomain collections for two example domains. The large triangles in the coarse mesh form the nonoverlapping subdomains Ω_k, and the refined regions form the overlapping subdomains Ω_k'. The symbols $\times$ denote nodes lying on the boundary of the global domain Ω, whereas the symbols $\circ$ denote nodes lying on the boundary of a particular subdomain Ω_k'.

where the form $a(\cdot, \cdot)$ is bilinear, symmetric, coercive, and bounded, whereas $f(\cdot)$ is linear and bounded. An overlapping domain decomposition method involves first solving (10.6.13) restricted to each overlapping subdomain Ω_k':

$$\text{Find } u_k \in \mathcal{V}_k \text{ such that } a(u_k, v_k) = f(v_k), \quad \forall v_k \in \mathcal{V}_k, \tag{10.6.14}$$

and then combining the results to improve an approximation over the entire domain Ω. Since the global problem over Ω was not solved, this procedure must be repeated until it converges to the solution of the global problem (10.6.13). Therefore, overlapping domain decomposition methods can be viewed as iterative methods for solving the variational problem (10.6.13), where each iteration involves approximate projections of the error onto subspaces of $\mathcal{V}$ associated with the overlapping subdomains Ω_k', which is accomplished by solving the subspace problem (10.6.14).

It is useful to reformulate problems (10.6.13) and (10.6.14) as operator equations in the function spaces defined over Ω and Ω_k'. Let $\mathcal{V}_k = H_0^1(\Omega_k') \cap \mathcal{V}$, $k = 1, \ldots, J$; it is not difficult to show that $\mathcal{V} = \mathcal{V}_1 + \cdots + \mathcal{V}_J$, where the coarse space $\mathcal{V}_0$ may also be included in the sum. Through the Riesz representation theorem and the Bounded Operator Theorem of Section 4.8, we can associate with the problem above an abstract operator equation $Au = f$, where A is SPD. We denote as A_k the restriction of the operator A to the space $\mathcal{V}_k$, corresponding to (any) discretization of the original problem restricted to the subdomain Ω_k'. Algebraically, it can be shown that $A_k = I_k^T A I_k$, where I_k is the natural inclusion of $\mathcal{V}_k$ into $\mathcal{V}$ and I_k^T is the corresponding projection of $\mathcal{V}$ onto $\mathcal{V}_k$. The property that I_k is the natural inclusion and I_k^T is the corresponding projection holds for both the finite element space $\mathcal{V}_k$ as

well as the Euclidean space $\mathbb{R}^{n_k}$. In other words, domain decomposition methods automatically satisfy the so-called *variational condition*:

$$A_k = I_k^T A I_k \tag{10.6.15}$$

in the subspaces $\mathcal{V}_k$, $k \neq 0$, for *any* discretization. Recall that A-orthogonal projection from $\mathcal{V}$ onto $\mathcal{V}_k$ can be written as $P_k = I_k(I_k^T A I_k)^{-1} I_k^T A$, which becomes simply $P_k = I_k A_k^{-1} I_k^T A$ when A_k satisfies the variational condition (10.6.15). If $R_k \approx A_k^{-1}$, we can define the *approximate* A-orthogonal projector from $\mathcal{V}$ onto $\mathcal{V}_k$ as $T_k = I_k R_k I_k^T A$. The case of $R_k = A_k^{-1}$ corresponds to an exact solution of the subdomain problems, giving $T_k = P_k$.

A *multiplicative Schwarz overlapping domain decomposition method*, employing *successive* approximate projections onto the subspaces $\mathcal{V}_k$ and written in terms of the operators A and A_k, has the following form.

Algorithm 10.6.3 (Multiplicative Schwarz Method: Implementation Form).

```
Set u^{i+1} = MS(u^i, f), where u^{i+1} = MS(u^i, f) is defined as:
Do k = 1, ..., J
     r_k = I_k^T (f - A u^i)
     e_k = R_k r_k
     u^{i+1} = u^i + I_k e_k
     u^i = u^{i+1}
End do.
```

Note that the first step through the loop in $MS(\cdot, \cdot)$ gives

$$\begin{aligned} u^{i+1} &= u^i + I_1 e_1 \\ &= u^i + I_1 R_1 I_1^T (f - A u^i) \\ &= (I - I_1 R_1 I_1^T A) u^i + I_1 R_1 I_1^T f. \end{aligned}$$

Continuing in this fashion, and by defining $T_k = I_k R_k I_k^T A$, we see that after the full loop in $MS(\cdot, \cdot)$ the solution transforms according to

$$u^{i+1} = (I - T_J)(I - T_{J-1}) \cdots (I - T_1) u^i + Bf,$$

where B is a quite complicated combination of the operators R_k, I_k, I_k^T, and A. By defining $E_k = (I - T_k)(I - T_{k-1}) \cdots (I - T_1)$, we see that $E_k = (I - T_k) E_{k-1}$. Therefore, since $E_{k-1} = I - B_{k-1} A$ for some (implicitly defined) B_{k-1}, we can identify the operators B_k through the recursion $E_k = I - B_k A = (I - T_k) E_{k-1}$, giving

$$\begin{aligned} B_k A &= I - (I - T_k) E_{k-1} = I - (I - B_{k-1} A) + T_k (I - B_{k-1} A) \\ &= B_{k-1} A + T_k - T_k B_{k-1} A = B_{k-1} A + I_k R_k I_k^T A - I_k R_k I_k^T A B_{k-1} A \\ &= \left[B_{k-1} + I_k R_k I_k^T - I_k R_k I_k^T A B_{k-1} \right] A, \end{aligned}$$

so that $B_k = B_{k-1} + I_k R_k I_k^T - I_k R_k I_k^T A B_{k-1}$. But this means that Algorithm 10.6.3 is equivalent to the following.

Algorithm 10.6.4 (Multiplicative Schwarz Method: Operator Form).

```
Define:
     u^{i+1} = u^i + B(f - Au^i) = (I - BA)u^i + Bf,
where the error propagator E is defined by:
     E = I - BA = (I - T_J)(I - T_{J-1}) ··· (I - T_1),
     T_k = I_k R_k I_k^T A,   k = 1, ..., J.
The implicit operator B ≡ B_J obeys the recursion:
     B_1 = I_1 R_1 I_1^T,   B_k = B_{k-1} + I_k R_k I_k^T - I_k R_k I_k^T A B_{k-1},   k = 2, ..., J.
```

An *additive Schwarz overlapping domain decomposition method*, employing *simultaneous* approximate projections onto the subspaces $\mathcal{V}_k$, has the form:

Algorithm 10.6.5 (Additive Schwarz Method: Implementation Form).

```
Set u^{i+1} = AS(u^i, f), where u^{i+1} = AS(u^i, f) is defined as:
r = f - Au^i
Do k = 1, ..., J
     r_k = I_k^T r
     e_k = R_k r_k
     u^{i+1} = u^i + I_k e_k
End do.
```

Since each loop iteration depends only on the original approximation u^i, we see that the full correction to the solution can be written as the sum

$$u^{i+1} = u^i + B(f - Au^i) = u^i + \sum_{k=1}^{J} I_k R_k I_k^T (f - Au^i),$$

where the preconditioner B has the form $B = \sum_{k=1}^{J} I_k R_k I_k^T$, and the error propagator is $E = I - BA$. Therefore, Algorithm 10.6.5 is equivalent to the following.

Algorithm 10.6.6 (Additive Schwarz Method: Operator Form).

```
Define:
     u^{i+1} = u^i + B(f - Au^i) = (I - BA)u^i + Bf,
where the error propagator E is defined by:
     E = I - BA = I - Σ_{k=1}^{J} T_k,
     T_k = I_k R_k I_k^T A,   k = 1, ..., J.
The operator B is defined explicitly as:
     B = Σ_{k=1}^{J} I_k R_k I_k^T .
```

Therefore, the multiplicative and additive domain decomposition methods fit exactly into the framework of a basic linear method (Algorithm 10.6.1) or can be viewed as methods for constructing preconditioners B for use with the conjugate gradient method (Algorithm 10.6.2). If $R_k = A_k^{-1}$, where A_k satisfies the variational condition (10.6.15), then each iteration of the algorithms involves removal of the A-orthogonal projection of the error onto each subspace, either successively (the multiplicative method) or simultaneously (the additive method). If R_k is an approximation to A_k^{-1}, then each step is an approximate A-orthogonal projection.

Multilevel Methods

Multilevel (or *multigrid*) methods are highly efficient numerical techniques for solving the algebraic equations arising from the discretization of partial differential equations. These methods were developed in direct response to the deficiencies of the classical iterations such as the Gauss-Seidel and SOR methods. Some of the early fundamental papers are [18, 40, 84, 162], as well as [17, 19, 185], and a comprehensive analysis of the many different aspects of these methods is given in [85, 178]. The following derivation of two-level and multilevel methods in a recursive operator framework is motivated by some work on finite element-based multilevel and domain decomposition methods, represented, for example, by [38, 66, 100, 178]. Our notation follows the currently established convention for these types of methods; see [100, 178].

Linear Equations in a Nested Sequence of Spaces. In what follows we are concerned with a nested sequence of spaces $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots \subset \mathcal{H}_J \equiv \mathcal{H}$, where $\mathcal{H}_J$ corresponds to the finest or largest space and $\mathcal{H}_1$ the coarsest or smallest. Each space $\mathcal{H}_k$ is taken to be a Hilbert space, equipped with an inner product $(\cdot,\cdot)_k$ which induces the norm $\|\cdot\|_k$. Regarding notation, if $A \in \mathcal{L}(\mathcal{H}_k, \mathcal{H}_k)$, then we denote the operator as A_k. Similarly, if $A \in \mathcal{L}(\mathcal{H}_k, \mathcal{H}_i)$, then we denote the operator as A_k^i. Finally, if $A \in \mathcal{L}(\mathcal{H}_k, \mathcal{H}_k)$ but its operation somehow concerns a specific subspace $\mathcal{H}_i \subset \mathcal{H}_k$, then we denote the operator as $A_{k;i}$. For quantities involving the finest space $\mathcal{H}_J$, we will often leave off the subscripts without danger of confusion.

Now, given such a nested sequence of Hilbert spaces, we assume that associated with each space $\mathcal{H}_k$ is an SPD operator A_k, which defines a second inner product $(\cdot,\cdot)_{A_k} = (A_k\cdot,\cdot)_k$, inducing a second norm $\|\cdot\|_{A_k} = (\cdot,\cdot)_{A_k}^{1/2}$. The spaces $\mathcal{H}_k$ are connected by *prolongation* operators $I_{k-1}^k \in \mathcal{L}(\mathcal{H}_{k-1}, \mathcal{H}_k)$ and *restriction* operators $I_k^{k-1} \in \mathcal{L}(\mathcal{H}_k, \mathcal{H}_{k-1})$, where we assume that the null space of I_{k-1}^k contains only the zero vector, and usually that $I_k^{k-1} = (I_{k-1}^k)^T$, where the (Hilbert) adjoint is with respect to the inner products on the sequence of spaces $\mathcal{H}_k$:

$$(u_k, I_{k-1}^k v_{k-1})_k = ((I_{k-1}^k)^T u_k, v_{k-1})_{k-1}, \quad \forall u_k \in \mathcal{H}_k, \quad \forall v_{k-1} \in \mathcal{H}_{k-1}. \tag{10.6.16}$$

We are given the operator equation $Au = f$ in the finest space $\mathcal{H} \equiv \mathcal{H}_J$, where $A \in \mathcal{L}(\mathcal{H}, \mathcal{H})$ is SPD, and we are interested in iterative algorithms for determining the unique solution u which involves solving problems in the coarser spaces $\mathcal{H}_k$ for $1 \leqslant k < J$. If the equation in $\mathcal{H}$ has arisen from finite element or similar discretization of an elliptic partial differential equation, then operators A_k (and the associated coarse problems $A_k u_k = f_k$) in coarser spaces $\mathcal{H}_k$ for $k < J$ may be defined naturally with the same discretization on a coarser mesh. Alternatively, it is convenient (for theoretical reasons which we discuss later in the chapter) to take the so-called *variational approach* of constructing the coarse operators, where the operators $A_k \in \mathcal{L}(\mathcal{H}_k, \mathcal{H}_k)$ satisfy

$$A_{k-1} = I_k^{k-1} A_k I_{k-1}^k, \qquad I_k^{k-1} = (I_{k-1}^k)^T. \tag{10.6.17}$$

The first condition in (10.6.17) is sometimes referred to as the *Galerkin condition*, whereas the two conditions (10.6.17) together are known as the *variational conditions*, due to the fact that both conditions are satisfied naturally by variational or Galerkin (finite element) discretizations on successively refined meshes. Note that if A_k is SPD, then A_{k-1} produced by (10.6.17) will also be SPD.

In the case that $\mathcal{H}_k = \mathcal{U}_k = \mathbb{R}^{n_k}$, the prolongation operator I_{k-1}^k typically corresponds to d-dimensional interpolation of u_{k-1} to $u_k = I_{k-1}^k u_{k-1}$, where u_{k-1} and u_k are interpreted as grid functions defined over two successively refined (box or finite element) discretizations Ω_{k-1} and Ω_k of the domain $\Omega \subset \mathbb{R}^d$. Since the coarse grid function space has by definition smaller dimension than the fine space, I_{k-1}^k takes the form of a rectangular matrix with more rows than columns. A positive scaling constant $c \in \mathbb{R}$ will appear in the second condition in (10.6.17), which will become $I_k^{k-1} = c(I_{k-1}^k)^T$, due to taking I_k^{k-1} to be the adjoint of I_{k-1}^k with respect to the inner product (10.5.53). This results from $h_k < h_{k-1}$ on two successive spaces, and the subsequent need to scale the corresponding discrete inner product to preserve a discrete notion of volume; this scaling allows for comparing inner products on spaces with different dimensions.

In the case that $\mathcal{H}_k = \mathcal{V}_k$, where $\mathcal{V}_k$ is a finite element subspace, the prolongation corresponds to the natural inclusion of a coarse space function into the fine space, and the restriction corresponds to its natural adjoint operator, which is the L^2-projection of a fine space function onto the coarse space. The variational conditions (10.6.17) then hold for the abstract operators A_k on the spaces $\mathcal{V}_k$, with inclusion and L^2-projection for the prolongation and restriction (see the proof in [85]). In addition, the stiffness matrices representing the abstract operators A_k also satisfy the conditions (10.6.17), where now the prolongation and restriction operators are as in the case of the space $\mathcal{U}_k$. However, we remark that this is true only with *exact evaluation* of the integrals forming the matrix components; the conditions (10.6.17) are violated if quadrature is used. "Algebraic multigrid" are methods based on enforcing (10.6.17) algebraically using a product of sparse matrices; one can develop a strong two-level theory for this class of methods in the case of M-matrices (see, for example, [41, 151]), but it is difficult to develop theoretical results for multilevel versions of these methods.

Many important results have been obtained for multilevel methods in the spaces $\mathcal{H}_k = \mathcal{V}_k$, which rely on certain operator recursions (we point out in particular the papers [36, 38, 177, 178]). Some of these results [38, 178] are "regularity-free" in the sense that they do not require the usual regularity or smoothness assumptions on the solution to the problem, which is important since these are not valid for problems such as those with discontinuous coefficients. As a result, we will develop multilevel algorithms in a recursive form in the abstract spaces $\mathcal{H}_k$.

Two-Level Methods. As we noted earlier, the convergence rate of the classical methods (Gauss-Seidel and similar methods) deteriorate as the mesh size $h_k \to 0$; we examine the reasons for this behavior for a model problem later in this section. However, using the same spectral analysis, one can easily see that the components of the error corresponding to the small eigenvalues of the error propagation operator are

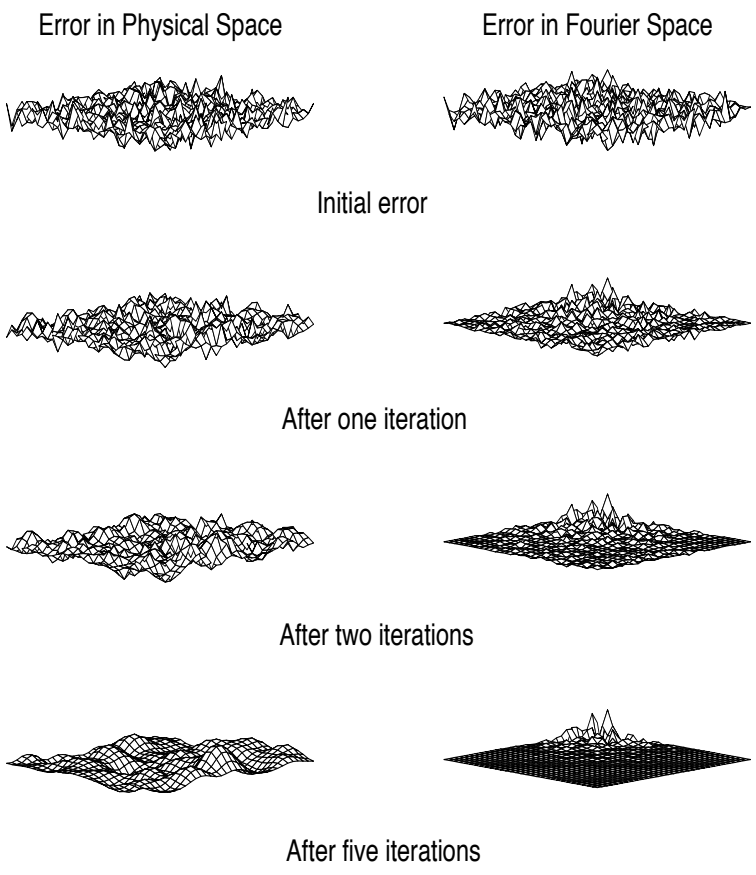

Figure 10.11 Error-smoothing effect of Gauss-Seidel iteration. The error in both physical and Fourier (or frequency) space is shown initially and after one, two, and five iterations. Low-frequency components of the error appear at the rear of the Fourier plots; high-frequency components appear at far left, far right, and in the foreground.

actually being decreased quite effectively even as $h_k \to 0$; these are the rapidly varying or *high-frequency* components in the error. This effect is illustrated graphically in Figure 10.11 for Gauss-Seidel iteration applied to the two-dimensional Poisson equation on the unit square. In the figure, the error in both physical and Fourier (or frequency) space is shown initially and after one, two, and five iterations. In the Fourier space plots, the low-frequency components of the error are found in the rear, whereas the high-frequency components are found to the far left, the far right, and in

the foreground. The source function for this example was constructed from a random field (to produce all frequencies in the solution) and the initial guess was taken to be zero.

The observation that classical linear methods are very efficient at reducing the high-frequency modes is the motivation for the multilevel method: A classical linear method can be used to handle the high-frequency components of the error (or to *smooth* the error), and the low-frequency components can be eliminated efficiently on a coarser mesh with fewer unknowns, where the low-frequency modes are well represented.

For the equation $A_k u_k = f_k$ on level k, the smoothing method takes the form of Algorithm 10.6.1 for some operator R_k, the *smoothing operator*, as the approximate inverse of the operator A_k:

$$u_k^{i+1} = u_k^i + R_k(f_k - A_k u_k^i). \tag{10.6.18}$$

In the case of two spaces $\mathcal{H}_k$ and $\mathcal{H}_{k-1}$, the error equation $e_k = A_k^{-1} r_k$ is solved approximately using the coarse space, with the *coarse-level correction operator* $C_k = I_{k-1}^k A_{k-1}^{-1} I_k^{k-1}$ representing the exact solution with A_{k-1}^{-1} in the coarse-level subspace $\mathcal{H}_{k-1}$. The solution is then adjusted by the correction

$$u_k^{i+1} = u_k^i + C_k(f_k - A_k u_k^i). \tag{10.6.19}$$

There are several ways in which these two procedures can be combined.

By viewing multilevel methods as compositions of the simple linear methods (10.6.18) and (10.6.19), a simple yet complete framework for understanding these methods can be constructed. The most important concepts can be discussed with regard to two-level methods and then generalized to more than two levels using an implicit recursive definition of an approximate coarse-level inverse operator.

Consider the case of two nested spaces $\mathcal{H}_{k-1} \subset \mathcal{H}_k$, and the following two-level method:

Algorithm 10.6.7 (Nonsymmetric Two-Level Method).

$v_k = u_k^i + C_k(f_k - A_k u_k^i)$. `[Coarse-level correction]`
$u_k^{i+1} = v_k + R_k(f_k - A_k v_k)$. `[Post-smoothing]`

The coarse-level correction operator has the form $C_k = I_{k-1}^k A_{k-1}^{-1} I_k^{k-1}$, and the smoothing operator is one of the classical iterations. This two-level iteration, a composition of two linear iterations of the form of Algorithm 10.6.1, can itself be written in the form of Algorithm 10.6.1:

$$\begin{aligned} u_k^{i+1} &= v_k + R_k(f_k - A_k v_k) \\ &= u_k^i + C_k(f_k - A_k u_k^i) + R_k f_k - R_k A_k(u_k^i + C_k(f_k - A_k u_k^i)) \\ &= (I - C_k A_k - R_k A_k + R_k A_k C_k A_k) u_k^i + (C_k + R_k - R_k A_k C_k) f_k \\ &= (I - B_k A_k) u_k^i + B_k f_k. \end{aligned}$$

The *two-level operator* B_k, the approximate inverse of A_k which is implicitly defined by the nonsymmetric two-level method, has the form:

$$B_k = C_k + R_k - R_k A_k C_k. \tag{10.6.20}$$

The error propagation operator for the two-level method has the usual form $E_k = I - B_k A_k$, which now can be factored due to the form for B_k above:

$$E_k = I - B_k A_k = (I - R_k A_k)(I - C_k A_k). \tag{10.6.21}$$

In the case that ν post-smoothing iterations are performed in step (2) instead of a single post-smoothing iteration, it is not difficult to show that the error propagation operator takes the altered form

$$I - B_k A_k = (I - R_k A_k)^\nu (I - C_k A_k).$$

Now consider a symmetric form of the above two-level method:

Algorithm 10.6.8 (Symmetric Two-Level Method).

$w_k = u_k^i + R_k^T(f_k - A_k u_k^i)$. `[Pre-smoothing]`
$v_k = w_k + C_k(f_k - A_k w_k)$. `[Coarse-level correction]`
$u_k^{i+1} = v_k + R_k(f_k - A_k v_k)$. `[Post-smoothing]`

As in the nonsymmetric case, it is a simple task to show that this two-level iteration can be written in the form of Algorithm 10.6.1:

$$u_k^{i+1} = (I - B_k A_k) u_k^i + B_k f_k,$$

where after a simple expansion as for the nonsymmetric method above, the *two-level operator* B_k implicitly defined by the symmetric method can be seen to be

$$B_k = R_k + C_k + R_k^T - R_k A_k C_k - R_k A_k R_k^T - C_k A_k R_k^T + R_k A_k C_k A_k R_k^T.$$

It is easily verified that the factored form of the resulting error propagator E_k^s for the symmetric algorithm is

$$E_k^s = I - B_k A_k = (I - R_k A_k)(I - C_k A_k)(I - R_k^T A_k).$$

Note that the operator $I - B_k A_k$ is A_k-self-adjoint, which by Lemma 10.6.4 is true if and only if B_k is symmetric, implying the symmetry of B_k. The operator B_k constructed by the symmetric two-level iteration is always symmetric if the smoothing operator R_k is symmetric; however, it is also true in the symmetric algorithm above when general nonsymmetric smoothing operators R_k are used, because we use the adjoint R_k^T of the post-smoothing operator R_k as the pre-smoothing operator. The symmetry of B_k is important for use as a preconditioner for the conjugate gradient method, which requires that B_k be symmetric for guarantee of convergence.

REMARK. Note that this alternating technique for producing symmetric operators B_k can be extended to multiple nonsymmetric smoothing iterations, as suggested in [37]. Denote the variable nonsymmetric smoothing operator $R_k^{(i)}$ as

$$R_k^{(j)} = \begin{cases} R_k, & j \text{ odd}, \\ R_k^T, & j \text{ even}. \end{cases}$$

If ν pre-smoothings are performed, alternating between R_k and R_k^T, and ν post-smoothings are performed alternating in the opposite way, then a tedious computation shows that the error propagator has the factored form

$$I - B_k A_k = \left(\prod_{j=\nu}^{1} (I - R_k^{(j)} A_k) \right) (I - C_k A_k) \left(\prod_{j=1}^{\nu} (I - (R_k^{(j)})^T A_k) \right),$$

where we adopt the convention that the first terms indexed by the products appear on the left. It is easy to verify that $I - B_k A_k$ is A_k-self-adjoint, so that B_k is symmetric.

Variational Conditions and A-Orthogonal Projection. Up to this point, we have specified the approximate inverse corresponding to the coarse-level subspace correction only as $C_k = I_{k-1}^k A_{k-1}^{-1} I_k^{k-1}$, for some coarse-level operator A_{k-1}. Consider the case that the variational conditions (10.6.17) are satisfied. The error propagation operator for the coarse-level correction then takes the form

$$I - C_k A_k = I - I_{k-1}^k A_{k-1}^{-1} I_k^{k-1} A_k = I - I_{k-1}^k [(I_{k-1}^k)^T A_k I_{k-1}^k]^{-1} (I_{k-1}^k)^T A_k.$$

This last expression is simply the A_k-orthogonal projector $I - P_{k;k-1}$ onto the complement of the coarse-level subspace, where the unique orthogonal and A_k-orthogonal projectors $Q_{k;k-1}$ and $P_{k;k-1}$ projecting $\mathcal{H}_k$ onto $I_{k-1}^k \mathcal{H}_{k-1}$ can be written as

$$Q_{k;k-1} = I_{k-1}^k [(I_{k-1}^k)^T I_{k-1}^k]^{-1} (I_{k-1}^k)^T,$$
$$P_{k;k-1} = C_k A_k = I_{k-1}^k [(I_{k-1}^k)^T A_k I_{k-1}^k]^{-1} (I_{k-1}^k)^T A_k.$$

In other words, if the variational conditions are satisfied, and the coarse-level equations are solved exactly, then the coarse-level correction projects the error onto the A_k-orthogonal complement of the coarse-level subspace. It is now not surprising that successively refined finite element discretizations satisfy the variational conditions naturally, since they are defined in terms of A_k-orthogonal projections.

Note the following interesting relationship between the symmetric and nonsymmetric two-level methods, which is a consequence of the A_k-orthogonal projection property.

Lemma 10.6.12. *If the variational conditions (10.6.17) hold, then the nonsymmetric and symmetric propagators E_k and E_k^s are related by*

$$\|E_k^s\|_{A_k} = \|E_k\|_{A_k}^2.$$

Proof. Since $I - C_k A_k$ is a projector, we have $(I - C_k A_k)^2 = I - C_k A_k$. It follows that

$$\begin{aligned} E_k^s &= (I - R_k A_k)(I - C_k A_k)(I - R_k^T A_k) \\ &= (I - R_k A_k)(I - C_k A_k)(I - C_k A_k)(I - R_k^T A_k) = E_k E_k^*, \end{aligned}$$

where E_k^* is the A_k-adjoint of E_k. Therefore, the convergence of the symmetric algorithm is related to that of the nonsymmetric algorithm by:

$$\|E_k^s\|_{A_k} = \|E_k E_k^*\|_{A_k} = \|E_k\|_{A_k}^2.$$

□

REMARK. The relationship between the symmetric and nonsymmetric error propagation operators in Lemma 10.6.12 was first pointed out by McCormick in [131], and has been exploited in many papers; see [36, 100, 178]. It allows one to use the symmetric form of the algorithm as may be necessary for use with conjugate gradient methods while exploiting the relationship above to work only with the nonsymmetric error propagator E_k in analysis, which may be easier to analyze.

Multilevel Methods. Consider now the full nested sequence of Hilbert spaces $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots \subset \mathcal{H}_J \equiv \mathcal{H}$. The idea of the multilevel method is to begin with the two-level method, but rather than solve the coarse-level equations exactly, yet another two-level method is used to solve the coarse-level equations approximately, beginning with an initial approximation of zero on the coarse-level. The idea is applied recursively until the cost of solving the coarse system is negligible, or until the coarsest possible level is reached. Two nested simplicial mesh hierarchies for building piecewise-linear finite element spaces in the case $\mathcal{H}_k = \mathcal{V}_k$ are shown in Figure 10.12.

The following is a recursively defined multilevel algorithm which corresponds to the form of the algorithm commonly implemented on a computer. For the system $Au = f$, the algorithm returns the approximate solution u^{i+1} after one iteration of the method applied to the initial approximate u^i.

Algorithm 10.6.9 (Nonsymmetric Multilevel Method: Implementation Form).

```
Set:
     u^{i+1} = ML(J, u^i, f)
where u_k^{i+1} = ML(k, u_k^i, f_k) is defined recursively as:
     If (k = 1) Then:
          u_1^{i+1} = A_1^{-1} f_1 .                          [Direct solve]
     Else:
          v_k = u_k^i + I_{k-1}^k (ML(k-1, 0, I_k^{k-1}(f_k - A_k u_k^i))) .   [Correction]
          u_k^{i+1} = v_k + R_k (f_k - A_k v_k) .             [Post-smoothing]
     End.
```

As with the two-level Algorithm 10.6.7, it is a straightforward calculation to write the multilevel Algorithm 10.6.9 in the standard form of Algorithm 10.6.1, where now

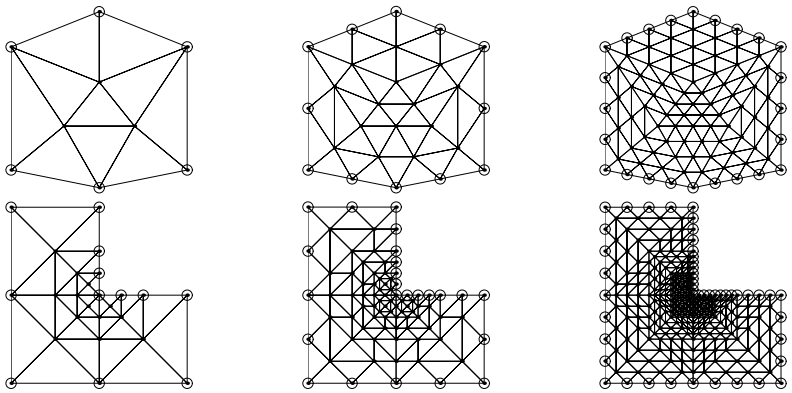

Figure 10.12 Unstructured three-level mesh hierarchies for two example domains. The nested refinements are achieved by successive quadra-section (subdivision into four similar subtriangles). Nested hierarchies of finite element spaces are then built over theses nested triangulations.

the *multilevel operator* $B \equiv B_J$ is defined recursively. To begin, assume that the approximate inverse of A_{k-1} at level $k-1$ implicitly defined by Algorithm 10.6.9 has been explicitly identified and denoted as B_{k-1}. The coarse-level correction step of Algorithm 10.6.9 at level k can then be written as

$$v_k = u_k^i + I_{k-1}^k B_{k-1} I_k^{k-1}(f_k - A_k u_k^i).$$

At level k, Algorithm 10.6.9 can be thought of as the two-level Algorithm 10.6.7, where the two-level operator $C_k = I_{k-1}^k A_{k-1}^{-1} I_k^{k-1}$ has been replaced by the approximation $C_k = I_{k-1}^k B_{k-1} I_k^{k-1}$. From (10.6.20) we see that the expression for the multilevel operator B_k at level k in terms of the operator B_{k-1} at level $k-1$ is given by

$$B_k = I_{k-1}^k B_{k-1} I_k^{k-1} + R_k - R_k A_k I_{k-1}^k B_{k-1} I_k^{k-1}. \tag{10.6.22}$$

We can now state a second multilevel algorithm, which is mathematically equivalent to Algorithm 10.6.9, but which is formulated explicitly in terms of the recursively defined multilevel operators B_k.

Algorithm 10.6.10 (Nonsymmetric Multilevel Method: Operator Form).

Set: $u^{i+1} = u^i + B(f - Au^i)$,
where the operator $B \equiv B_J$ is defined recursively:
Let $B_1 = A_1^{-1}$, and assume that B_{k-1} has been defined.
$B_k = I_{k-1}^k B_{k-1} I_k^{k-1} + R_k - R_k A_k I_{k-1}^k B_{k-1} I_k^{k-1}, \quad k = 2, \ldots, J.$

REMARK. Recursive definition of multilevel operators B_k apparently first appeared in [36], although operator recursions for the error propagators $E_k = I - B_k A_k$ appeared earlier in [125]. Many of the results on finite element-based multilevel methods depend on the recursive definition of the multilevel operators B_k.

As was noted for the two-level case, the error propagator at level k can be factored as:

$$E_k = I - B_k A_k = (I - R_k A_k)(I - I_{k-1}^k B_{k-1} I_k^{k-1} A_k). \tag{10.6.23}$$

It can be shown (see [39, 87, 100, 175, 178]) that the multilevel error propagator can actually be factored into a full product.

Lemma 10.6.13. *If variational conditions (10.6.17) hold, the error propagator E of Algorithm 10.6.10 can be factored:*

$$E = I - BA = (I - T_J)(I - T_{J-1}) \cdots (I - T_1), \tag{10.6.24}$$

where

$$T_1 = I_1 A_1^{-1} I_1^T A, \qquad T_k = I_k R_k I_k^T A, \qquad k = 2, \ldots, J,$$

with

$$I_J = I, \quad I_k = I_{J-1}^J I_{J-2}^{J-1} \cdots I_{k+1}^{k+2} I_k^{k+1}, \quad k = 1, \ldots, J-1.$$

Moreover, one has the additional variational condition

$$A_k = I_k^T A I_k. \tag{10.6.25}$$

Proof. Let us begin by expanding the second term in (10.6.23) more fully and then factoring again:

$$\begin{aligned} I - I_{k-1}^k B_{k-1} I_k^{k-1} A_k &= I - I_{k-1}^k (I_{k-2}^{k-1} B_{k-2} I_{k-1}^{k-2} + R_{k-1} \\ &\qquad - R_{k-1} A_{k-1} I_{k-2}^{k-1} B_{k-2} I_{k-1}^{k-2}) I_k^{k-1} A_k \\ &= I - I_{k-2}^k B_{k-2} I_k^{k-2} A_k - I_{k-1}^k R_{k-1} I_k^{k-1} A_k \\ &\qquad + I_{k-1}^k R_{k-1} (I_k^{k-1} A_k I_{k-1}^k) I_{k-2}^{k-1} B_{k-2} I_k^{k-2} A_k \\ &= I - I_{k-2}^k B_{k-2} I_k^{k-2} A_k - I_{k-1}^k R_{k-1} I_k^{k-1} A_k \\ &\qquad + (I_{k-1}^k R_{k-1} I_k^{k-1} A_k)(I_{k-2}^k B_{k-2} I_k^{k-2} A_k) \\ &= (I - I_{k-1}^k R_{k-1} I_k^{k-1} A_k)(I - I_{k-2}^k B_{k-2} I_k^{k-2} A_k), \end{aligned}$$

where we have assumed that the first part of the variational conditions (10.6.17) holds. In general, we have

$$I - I_{k-i}^k B_{k-i} I_k^{k-i} A_k = (I - I_{k-i}^k R_{k-i} I_k^{k-i} A_k)(I - I_{k-i-1}^k B_{k-i-1} I_k^{k-i-1} A_k).$$

Using this result inductively, beginning with $k = J$, the error propagator $E \equiv E_J$ takes the *product form*:

$$E = I - BA = (I - T_J)(I - T_{J-1}) \cdots (I - T_1).$$

The second part of the variational conditions (10.6.17) implies that the T_k are A-self-adjoint and have the form

$$T_1 = I_1 A_1^{-1} I_1^T A, \qquad T_k = I_k R_k I_k^T A, \qquad k = 2, \ldots, J.$$

That (10.6.25) holds follows from the definitions. □

Note that this lemma implies that the multilevel error propagator has precisely the same form as the multiplicative Schwarz domain decomposition error propagator. One can also define an additive version via the sum

$$E = I - BA = T_1 + T_2 + \cdots + T_J, \tag{10.6.26}$$

where B is now an additive preconditioner, again identical in form to the additive Schwarz domain decomposition error propagator. Lemma 10.6.13 made it possible to consider multilevel and domain decomposition methods as particular instances of a general class of *Schwarz methods*, which allowed for the development of a very general convergence theory; see, for example, [66, 87, 93, 178] for more detailed expositions of this convergence theory framework.

The V-Cycle, the W-Cycle, and Nested Iteration. The methods we have just described are standard examples of *multigrid* or *multilevel methods* [85], where we have introduced a few restrictions for convenience, such as equal numbers of pre- and post-smoothings, one coarse space correction per iteration, and pre-smoothing with the adjoint of the post-smoothing operator. These restrictions are unnecessary in practice, but are introduced to make the analysis of the methods somewhat simpler, and to result in a symmetric preconditioner as required for combination with the conjugate gradient method.

The procedure just outlined involving correcting with the coarse space once each iteration is referred to as the *V-cycle* [40]. A similar procedure is the *Variable V-cycle*, whereby the number of smoothing iterations in one cycle is increased as coarser spaces are visited [38]. Another variation is termed the *W-cycle*, in which two coarse space corrections are performed per level at each iteration. More generally, the *p-cycle* would involve p coarse space corrections per level at each iteration for some integer $p \geqslant 1$. The *full multigrid method* [40] or *nested iteration technique* [85] begins with the coarse space, prolongates the solution to a finer space, performs a p-cycle, and repeats the process until a p-cycle is performed on the finest level. The methods can be depicted as in Figure 10.13.

Complexity of Classical, CG, DD, and Multilevel Methods

We compare the complexity of multilevel methods to some classical linear iterations for discrete elliptic equations $Au = f$ on the space U (omitting the subscript k here and below since only one space is involved), where A is an SPD matrix. Our purpose is to explain briefly the motivation for considering the more complex domain decomposition and multilevel methods as essential alternatives to the classical methods.

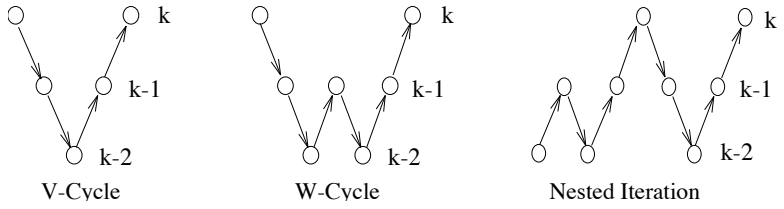

Figure 10.13 The V-cycle, the W-cycle, and nested iteration.

Convergence and Complexity of Classical Methods. Since A is SPD, we may write $A = D - L - L^T$, where D is a diagonal matrix and L a strictly lower-triangular matrix. The Richardson variation of Algorithm 10.6.1 takes λ^{-1} as the approximate inverse $B \approx A^{-1}$ of A, where λ is a bound on the largest eigenvalue of A:

$$u^{i+1} = (I - \lambda^{-1}A)u^i + \lambda^{-1}f. \tag{10.6.27}$$

The Jacobi variation of Algorithm 10.6.1 takes D^{-1} as the approximate inverse B:

$$u^{i+1} = (I - D^{-1}A)u^i + D^{-1}f. \tag{10.6.28}$$

In the Gauss-Seidel variant, the approximate inverse is taken to be $(D-L)^{-1}$, giving

$$u^{i+1} = (I - (D - L)^{-1}A)u^i + (D - L)^{-1}f. \tag{10.6.29}$$

The SOR variant takes the approximate inverse as $\omega(D - \omega L)^{-1}$, giving

$$u^{i+1} = (I - \omega(D - \omega L)^{-1}A)u^i + \omega(D - \omega L)^{-1}f. \tag{10.6.30}$$

When the model problem of the Poisson equation on a uniform mesh is considered, then the eigenvalues of both A and the error propagation matrix $I - BA$ can be determined analytically. This allows for an analysis of the convergence rates of the Richardson, Jacobi, and Gauss-Seidel iterations.

To give an example of the convergence results which are available for these classical methods, first recall that for the real square matrix A, the splitting $A = M - R$ is called a *regular splitting* (see [169]) of A if $R > 0$, M is nonsingular, and $M^{-1} \geqslant 0$. Note that an alternative construction of the Jacobi and Gauss-Seidel methods is through matrix splittings. For example, given the particular matrix splitting $A = M - R = D - (L + U)$, which corresponds to the Jacobi iteration, the resulting iteration can be written in terms of M and R as follows:

$$\begin{aligned} u^{i+1} &= (I - D^{-1}A)u^i + D^{-1}f = (I - M^{-1}(M - R))u^i + M^{-1}f \\ &= M^{-1}Ru^i + M^{-1}f. \end{aligned}$$

Therefore, for a splitting $A = M - R$, the convergence of the resulting linear method is governed completely by the spectral radius of the error propagation matrix, $\rho(M^{-1}R)$. The following standard theorem gives a sufficient condition for

convergence of the Jacobi and Gauss-Seidel iterations, which can be considered to be regular splittings of A.

Theorem 10.6.4. *If A is an M-matrix, and M is obtained from A by setting off-diagonal elements of A to zero, then the splitting $A = M - R$ is regular and the corresponding linear iteration defined by the splitting is convergent: $\rho(M^{-1}R) < 1$.*

Proof. This follows from Theorem 10.6.1; see also [169]. □

Given that λ is the largest eigenvalue (or an upper bound on the largest eigenvalue) of A, we remark that Richardson's method is always trivially convergent since each eigenvalue $\lambda_j(E)$ of E is bounded by 1:

$$\lambda_j(E) = \lambda_j(I - BA) = \lambda_j(I - \lambda^{-1}A) = 1 - \lambda^{-1}\lambda_j(A) < 1.$$

However, the following difficulty makes these classical linear methods impractical for large problems. Consider the case of the three-dimensional Poisson's equation on the unit cube with zero Dirichlet boundary conditions, discretized with the box-method on a uniform mesh with m meshpoints in each mesh direction ($n = m^3$) and mesh spacing $h = 1/(m+1)$. It is well-known that the eigenvalues of the resulting matrix A can be expressed in closed form

$$\lambda_j = \lambda_{\{p,q,r\}} = 6 - 2\cos p\pi h - 2\cos q\pi h - 2\cos r\pi h, \quad p, q, r = 1, \ldots, m.$$

Clearly, the largest eigenvalue of A is $\lambda = 6(1 - \cos m\pi h)$, and the smallest is $\lambda_1 = 6(1 - \cos \pi h)$. It is not difficult to show (see [169] or [184] for the two-dimensional case) that the largest eigenvalue of the Jacobi error propagation matrix $I - D^{-1}A$ is in this case equal to $\cos \pi h$. It is also well-known that for consistently ordered matrices with *Property $\mathcal{A}$* (see [184]), the spectral radius of the Gauss-Seidel error propagation matrix is the square of the Jacobi matrix spectral radius; more generally, the relationship between the Jacobi and Gauss-Seidel spectral radii is given by the *Stein-Rosenberg Theorem* (again see [169], or [184]). An expression for the spectral radius of the SOR error propagation matrix can also be derived; the spectral radii for the classical methods are then:

- Richardson: $\rho(E) = 1 - 6\lambda^{-1}(1 - \cos \pi h) \approx 1 - 3\lambda^{-1}\pi^2 h^2 = 1 - \mathcal{O}(h^2)$
- Jacobi: $\rho(E) = \cos \pi h \approx 1 - \frac{1}{2}\pi^2 h^2 = 1 - \mathcal{O}(h^2)$
- Gauss-Seidel: $\rho(E) = \cos^2 \pi h \approx 1 - \pi^2 h^2 = 1 - \mathcal{O}(h^2)$
- SOR: $\rho(E) \approx 1 - \mathcal{O}(h)$

The same dependence on h is exhibited for one- and two-dimensional problems. Therein lies the problem: As $h \to 0$, then for the classical methods, $\rho(E) \to 1$, so that the methods converge more and more slowly as the problem size is increased.

REMARK. An alternative convergence proof for the Jacobi and Gauss-Seidel iterations follows simply by noting that the matrix $I - E^*E$ is A-positive for both the Jacobi and Gauss-Seidel error propagators E, and by employing Lemma 10.6.2,

or the related Stein's Theorem. Stein's Theorem is the basis for the proof of the Ostrowski-Reich SOR convergence theorem (see [139]).

In the case of a uniform $m \times m \times m$ mesh and the standard box-method discretization of Poisson's equation on the unit cube, the resulting algebraic system is of dimension $N = m^3$. It is well-known that the computational complexities of dense, banded, and sparse Gaussian elimination are $\mathcal{O}(N^3)$, $\mathcal{O}(N^{7/3})$, and $\mathcal{O}(N^2)$, respectively, with storage requirements that are also worse than linear (even if the matrix A itself requires only storage linear in N). In order to understand how the iterative methods we have discussed in this chapter compare to direct methods as well as to each other in terms of *complexity*, we must translate their respective known convergence properties for the model problem into a complexity estimate.

Assume now that the discretization error is $\mathcal{O}(h^s)$ for some $s > 0$, which yields a practical linear iteration tolerance of $\epsilon = \mathcal{O}(h^s)$. As remarked earlier, if the mesh is shape-regular and quasi-uniform, then the mesh size h is related to the number of discrete unknowns N through the dimension d of the spatial domain as $h = \mathcal{O}(N^{-1/d})$. Now, for the model problem, we showed above that the spectral radii of the Richardson, Jacobi, and Gauss-Seidel behave as $1 - \mathcal{O}(h^2)$. Since $-\ln(1 - ch^2) \approx ch^2 + \mathcal{O}(h^4)$, we can estimate the number of iterations required to solve the problem to the level of discretization error from (10.6.6) as follows:

$$n \geqslant \frac{|\ln \epsilon|}{|\ln \rho(E)|} = \frac{|\ln h^s|}{|\ln(1 - ch^2)|} \approx \frac{|s \ln h|}{h^2} = \mathcal{O}\left(\frac{|\ln N^{-1/d}|}{N^{-2/d}}\right) = \mathcal{O}(N^{2/d} \ln N).$$

Assuming that the cost of each iteration is $\mathcal{O}(N)$ due to the sparsity of the matrices produced by standard discretization methods, we have that the total computational cost to solve the problem using any of the three methods above for $d = 3$ is $\mathcal{O}(N^{5/3} \ln N)$. A similar model problem analysis can be carried out for other methods.

Convergence and Complexity of Multilevel Methods. Let us now examine the complexity of multilevel methods. Multilevel methods first appeared in the Russian literature in [73]. In his 1961 paper Fedorenko, described a two-level method for solving elliptic equations, and in a second paper from 1964 [74] proved convergence of a multilevel method for Poisson's equation on the square. Many theoretical results have been obtained since these first two papers. In short, what can be proven for multilevel methods under reasonable conditions is that the convergence rate or contraction number (usually, the energy norm of the error propagator E^s) is bounded by a constant below 1, independent of the mesh size and the number of levels, and hence the number of unknowns:

$$\|E^s\|_A \leqslant \delta_J < 1. \tag{10.6.31}$$

In more general situations (such as problems with discontinuous coefficients), analysis yields contraction numbers which decay as the number of levels employed in the method is increased.

If a tolerance of ϵ is required, then the computational cost to reduce the energy norm of the error below the tolerance can be determined from (10.6.6) and (10.6.31):

$$i \geqslant \frac{|\ln \epsilon|}{|\ln \delta_J|} \geqslant \frac{|\ln \epsilon|}{|\ln \|E^s\|_A|}.$$

The discretization error of $\mathcal{O}(h_J^s)$ for some $s > 0$ yields a practical tolerance of $\epsilon = \mathcal{O}(h_J^s)$. As remarked earlier, for a shape-regular and quasi-uniform mesh, the mesh size h_J is related to the number of discrete unknowns n_J through the dimension d of the spatial domain as $n_J = \mathcal{O}(h_J^{-d})$. Assuming that $\delta_J < 1$ independently of J and h_J, we have that the maximum number of iterations i required to reach an error on the order of discretization error is

$$i \geqslant \frac{|\ln \epsilon|}{|\ln \delta_J|} = \mathcal{O}(|\ln h_J|) = \mathcal{O}(|\ln n_J^{-1/d}|) = \mathcal{O}(\ln n_J). \tag{10.6.32}$$

Consider now that the operation count o_J of a single (p-cycle) iteration of Algorithm 10.6.9 with J levels is given by

$$\begin{aligned} o_J &= po_{J-1} + Cn_J = p(po_{J-2} + Cn_{J-1}) + Cn_J = \cdots \\ &= p^{J-1}o_1 + C\sum_{k=2}^{J} p^{J-k}n_k, \end{aligned}$$

where we assume that the post-smoothing iteration has cost Cn_k for some constant C independent of the level k, and that the cost of a single coarse-level correction is given by o_{k-1}. Now, assuming that the cost to solve the coarse problem o_1 can be ignored, then it is not difficult to show from the expression for o_J above that the computational cost of each multilevel iteration is $\mathcal{O}(n_J)$ if (and only if) the dimensions of the spaces $\mathcal{H}_k$ satisfy

$$n_{k_1} < \frac{C_1}{p^{k_2-k_1}} n_{k_2}, \quad \forall k_1, k_2, \quad k_1 < k_2 \leqslant J,$$

where C_1 is independent of k. This implies both of the following:

$$n_k < \frac{C_1}{p} n_{k+1}, \qquad n_k < \frac{C_1}{p^{J-k}} n_J, \qquad k = 1, \ldots, J-1.$$

Consider the case of nonuniform Cartesian meshes which are successively refined, so that $h_{k_1} = 2^{k_2-k_1}h_{k_2}$ for $k_1 < k_2$, and in particular $h_{k-1} = 2h_k$. This gives

$$\begin{aligned} n_{k_1} &= C_2 h_{k_1}^{-d} = C_2(2^{k_2-k_1}h_{k_2})^{-d} = C_2 2^{-d(k_2-k_1)}(C_3 n_{k_2}^{-1/d})^{-d} \\ &= \frac{C_2 C_3^{-d}}{(2^d)^{k_2-k_1}} n_{k_2}. \end{aligned}$$

Therefore, if $2^{d(k_2-k_1)} > p^{k_2-k_1}$, or if $2^d > p$, which is true in two dimensions ($d = 2$) for $p \leqslant 3$, and in three dimensions ($d = 3$) for $p \leqslant 7$, then each multilevel

Table 10.1 Model problem computational complexities of various solvers.

Method	$3D$	$3D$
Dense Gaussian elimination	$\mathcal{O}(N^3)$	$\mathcal{O}(N^3)$
Banded Gaussian elimination	$\mathcal{O}(N^2)$	$\mathcal{O}(N^{2.33})$
Sparse Gaussian elimination	$\mathcal{O}(N^{1.5})$	$\mathcal{O}(N^2)$
Richardson's method	$\mathcal{O}(N^2 \ln N)$	$\mathcal{O}(N^{1.67} \ln N)$
Jacobi iteration	$\mathcal{O}(N^2 \ln N)$	$\mathcal{O}(N^{1.67} \ln N)$
Gauss-Seidel iteration	$\mathcal{O}(N^2 \ln N)$	$\mathcal{O}(N^{1.67} \ln N)$
SOR	$\mathcal{O}(N^{1.5} \ln N)$	$\mathcal{O}(N^{1.33} \ln N)$
Conjugate gradient methods (CG)	$\mathcal{O}(N^{1.5} \ln N)$	$\mathcal{O}(N^{1.33} \ln N)$
Preconditioned CG	$\mathcal{O}(N^{1.25} \ln N)$	$\mathcal{O}(N^{1.17} \ln N)$
Multilevel methods	$\mathcal{O}(N \ln N)$	$\mathcal{O}(N \ln N)$
Nested multilevel methods	$\mathcal{O}(N)$	$\mathcal{O}(N)$
Domain decomposition methods	$\mathcal{O}(N)$	$\mathcal{O}(N)$

iteration has complexity $\mathcal{O}(n_J)$. In particular, one V-cycle ($p = 1$) or W-cycle ($p = 2$) iteration has complexity $\mathcal{O}(n_J)$ for nonuniform Cartesian meshes in two and three dimensions.

If these conditions on the dimensions of the spaces are satisfied, so that each multilevel iteration has cost $\mathcal{O}(n_J)$, then combining this with equation (10.6.32) implies that the overall complexity to solve the problem with a multilevel method is $\mathcal{O}(n_J \ln n_J)$. By using the nested iteration, it is not difficult to show using an inductive argument (see [85]) that the multilevel method improves to optimal order $\mathcal{O}(n_J)$ if $\delta_J < 1$ independent of J and h_J, meaning that the computational cost to solve a problem with n_J pieces of data is Cn_J, for some constant C which does not depend on n_J. Theoretical multilevel studies first appeared in the late 1970s and continuing up through the present have focused on extending the proofs of optimality (or near optimality) to larger classes of problems.

To summarize, the complexities of the methods we have discussed in this chapter plus a few others are given in Table 10.1. The complexities for the conjugate gradient methods applied to the model problem may be found in [12]. The entry for domain decomposition methods is based on the assumption that the complexity of the solver on each subdomain is linear in the number of degrees of freedom in the subdomain (usually requiring the use of a multilevel method), and on the assumption that a global coarse space is solved to prevent the decay of the condition number or contraction constant with the number of subdomains. This table states clearly the motivation for considering the use of multilevel and domain decomposition methods for the numerical solution of elliptic partial differential equations.

EXERCISES

10.6.1 *Derivation of the conjugate gradient method.*

1. The Cayley-Hamilton Theorem states that a square $n \times n$ matrix M satisfies its own characteristic equation:

$$P_n(M) = 0.$$

Using this result, prove that if M is also nonsingular, then the matrix M^{-1} can be written as a matrix polynomial of degree $n-1$ in M, or

$$M^{-1} = Q_{n-1}(M).$$

2. Given an SPD matrix A, show that it defines a new inner product

$$(u, v)_A = (Au, v) = \sum_{i=1}^{n} (Au)_i v_i, \quad \forall u, v \in \mathbb{R}^n,$$

called the *A-inner product*; that is, show that $(u, v)_A$ is a "true" inner product, in that it satisfies the inner product axioms.

3. Recall that the transpose M^T of an $n \times n$ matrix M is defined as

$$M^T_{ij} = M_{ji}.$$

We observed in Section 3.4 that an equivalent characterization of the transpose matrix M^T is that it is the unique *adjoint* operator satisfying

$$(Mu, v) = (u, M^T v), \quad \forall u, v \in \mathbb{R}^n,$$

where $(\cdot, \cdot)$ is the usual Euclidean inner product,

$$(u, v) = \sum_{i=1}^{n} u_i v_i.$$

The A-adjoint of a matrix M, denoted M^*, is defined as the adjoint in the A-inner product; that his, the unique matrix satisfying

$$(AMu, v) = (Au, M^* v), \quad \forall u, v \in \mathbb{R}^n.$$

Show that that an equivalent definition of M^* is

$$M^* = A^{-1} M^T A.$$

4. Consider now the matrix equation

$$Au = f,$$

where A is an $n \times n$ SPD matrix, and u and f are n-vectors. It is common to "precondition" such an equation before attempting a numerical solution, by multiplying by an approximate inverse operator $B \approx A^{-1}$ and then solving the *preconditioned system*:

$$BAu = Bf.$$

If A and B are both SPD, under what conditions is BA also SPD? Show that if A and B are both SPD, then BA is A-SPD (symmetric and positive in the A-inner product).

5. Given an initial guess u^0 for the solution of $BAu = Bf$, we can form the initial residuals

$$r^0 = f - Au^0, \quad s^0 = Br^0 = Bf - BAu^0.$$

Do a simple manipulation to show that the solution u can be written as

$$u = u^0 + Q_{n-1}(BA)s^0,$$

where $Q(\cdot)$ is the matrix polynomial representing $(BA)^{-1}$. In other words, you have established that the solution u lies in a *translated Krylov space*:

$$u \in u^0 + K_{n-1}(BA, s^0),$$

where

$$K_{n-1}(BA, s^0) = \text{span}\{s^0, BAs^0, (BA)^2 s^0, \ldots, (BA)^{n-1} s^0\}.$$

Note that we can view the Krylov spaces as a sequence of expanding subspaces

$$K_0(BA, s^0) \subset K_1(BA, s^0) \subset \cdots \subset K_{n-1}(BA, s^0).$$

6. We will now try to construct an iterative method (the CG method) for finding u. The algorithm determines the best approximation u_k to u in a subspace $K_k(BA, s^0)$ at each step k of the algorithm, by forming

$$u^{k+1} = u^k + \alpha_k p^k,$$

where p^k is such that $p^k \in K_k(BA, s^0)$ at step k, but $p^k \notin K_j(BA, s^0)$ for $j < k$. In addition, we want to enforce minimization of the error in the A-norm,

$$\|e^{k+1}\|_A = \|u - u^{k+1}\|_A,$$

at step k of the algorithm. The next iteration expands the subspace to $K_{k+1}(BA, s^0)$, finds the best approximation in the expanded space, and so on, until the exact solution in $K_{n-1}(BA, s^0)$ is reached.

To realize this algorithm, let us consider how to construct the required vectors p^k in an efficient way. Let $p^0 = s^0$, and consider the construction of an A-orthogonal basis for $K_{n-1}(BA, s^0)$ using the standard Gram-Schmidt procedure:

$$p^{k+1} = BAp^k - \sum_{i=0}^{k} \frac{(BAp^k, p^i)_A}{(p^i, p^i)_A} p^i, \quad k = 0, \ldots, n-2.$$

At each step of the procedure, we will have generated an A-orthogonal (orthogonal in the A-inner product) basis $\{p^0, \ldots, p^k\}$ for $K_k(BA, s^0)$. Now, note that by construction,

$$(p^k, v)_A = 0, \quad \forall v \in K_j(BA, s^0), \;\; j < k.$$

Using this fact and the fact you established previously that BA is A-self-adjoint, show that the Gram-Schmidt procedure has only three nonzero terms in the sum; namely, for $k = 0, \ldots, n-1$, it holds that

$$p^{k+1} = BAp^k - \frac{(BAp^k, p^k)_A}{(p^k, p^k)_A} p^k - \frac{(BAp^k, p^{k-1})_A}{(p^{k-1}, p^{k-1})_A} p^{k-1}.$$

Thus, there exists an efficient three-term recursion for generating the A-orthogonal basis for the solution space. Note that this three-term recursion is possible due to the fact that we are working with orthogonal (matrix) polynomials!

7. We can nearly write down the CG method now, by attempting to expand the solution in terms of our cheaply generated A-orthogonal basis. However, we need to determine how far to move in each "conjugate" direction p^k at step k after we generate p^k from the recursion. As remarked earlier, we would like to enforce minimization of the quantity

$$\|e^{k+1}\|_A = \|u - u^{k+1}\|_A$$

at step k of the iterative algorithm. It is not difficult to show that this is equivalent to enforcing

$$(e^{k+1}, p^k)_A = 0.$$

Let's assume that we have somehow enforced

$$(e^k, p^i)_A = 0, \quad i < k,$$

at the previous step of the algorithm. We have at our disposal $p^k \in K_k(BA, s^0)$, and let's take our new approximation at step $k+1$ as

$$u^{k+1} = u^k + \alpha_k p^k,$$

for some step length $\alpha_k \in \mathbb{R}$ in the direction p^k. Thus, the error in the new approximation is simply

$$e^{k+1} = e^k + \alpha_k p^k.$$

Show that in order to enforce $(e^{k+1}, p^k)_A = 0$, we must choose α_k to be

$$\alpha_k = \frac{(r^k, p^k)}{(p^k, p^k)_A}.$$

The final algorithm is now as follows.

The Conjugate Gradient Algorithm
Let $u^0 \in \mathcal{H}$ be given.
$r^0 = f - Au^0, \;\; s^0 = Br^0, \;\; p^0 = s^0.$
Do $k = 0, 1, \ldots$ until convergence:
$\alpha_k = (r^k, p^k)/(p^k, p^k)_A$
$u^{k+1} = u^k + \alpha_k p^k$
$r^{k+1} = r^k - \alpha_k A p^k$
$s^{k+1} = B r^{k+1}$
$\beta_{k+1} = -(BAp^k, p^k)_A/(p^k, p^k)_A$
$\gamma_{k+1} = -(BAp^k, p^{k-1})_A/(p^{k-1}, p^{k-1})_A$
$p^{k+1} = BAp^k + \beta_{k+1} p^k + \gamma_{k+1} p^{k-1}$
End do.

8. Show that equivalent expressions for some of the parameters in CG are:
   (a) $\alpha_k = (r^k, s^k)/(p^k, p^k)_A$
   (b) $\delta_{k+1} = (r^{k+1}, s^{k+1})/(r^k, s^k)$
   (c) $p^{k+1} = s^{k+1} + \delta_{k+1} p^k$

   In other words, the CG algorithm you have derived from first principles in this exercise, using only the idea of orthogonal projection onto an expanding set of subspaces, is mathematically equivalent to Algorithm 10.6.2.

   Remark: The CG algorithm that appears in most textbooks is formulated to employ these equivalent expressions due to the reduction in computational work of each iteration.

10.6.2 *Properties of the conjugate gradient method.*

In this exercise, we will establish some simple properties of the CG method derived in Exercise 10.6.1. (Although this analysis is standard, you will have difficulty finding all of the pieces in one text.)

1. It is not difficult to show that the error in the CG algorithm propagates as

$$e^{k+1} = [I - BAp_k(BA)]e^0,$$

where $p_k \in \mathcal{P}_k$, the space of polynomials of degree k. By construction, we know that this polynomial is such that

$$\|e^{k+1}\|_A = \min_{p_k \in \mathcal{P}_k} \|[I - BAp_k(BA)]e^0\|_A.$$

Now, since BA is A-SPD, we know that it has real positive eigenvalues $\lambda_j \in \sigma(BA)$, and further, that the corresponding eigenvectors v_j of BA are orthonormal. Using the expansion of the initial error

$$e^0 = \sum_{j=1}^{n} \alpha_j v_j,$$

establish the inequality

$$\|e^{k+1}\|_A \leqslant \left(\min_{p_k \in \mathcal{P}_k} \left[\max_{\lambda_j \in \sigma(BA)} |1 - \lambda_j p_k(\lambda_j)| \right] \right) \|e^0\|_A.$$

The polynomial which minimizes the maximum norm above is said to solve a *mini-max problem*.

2. It is well-known in approximation theory that the Chebyshev polynomials

$$T_k(x) = \cos(k \arccos x)$$

solve mini-max problems of the type above, in the sense that they deviate least from zero (in the *max-norm* sense) in the interval $[-1, 1]$, which can be shown to be due to their unique *equi-oscillation* property. (These facts can be found in any introductory numerical analysis text.) If we extend the Chebyshev polynomials outside the interval $[-1, 1]$ in the natural way, it can be shown that shifted and scaled forms of the Chebyshev polynomials solve the mini-max problem above. In particular, the solution is simply

$$1 - \lambda p_k(\lambda) = \tilde{p}_{k+1}(\lambda) = \frac{T_{k+1}\left(\dfrac{\lambda_{\max} + \lambda_{\min} - 2\lambda}{\lambda_{\max} - \lambda_{\min}} \right)}{T_{k+1}\left(\dfrac{\lambda_{\max} + \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}} \right)}.$$

Use an obvious property of the polynomials $T_{k+1}(x)$ to conclude that

$$\|e^{k+1}\|_A \leqslant \left[T_{k+1}\left(\frac{\lambda_{\max} + \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}} \right) \right]^{-1} \|e_0\|_A.$$

3. Use one of the Chebyshev polynomial results given in Exercise 10.6.3 below to refine this inequality to

$$\|e^{k+1}\|_A \leqslant 2\left(\frac{\sqrt{\dfrac{\lambda_{\max}(BA)}{\lambda_{\min}(BA)}}-1}{\sqrt{\dfrac{\lambda_{\max}(BA)}{\lambda_{\min}(BA)}}+1}\right)^{k+1}\|e^0\|_A.$$

Now, recall that the *A-condition number* of the matrix BA is defined just as the normal condition number, except employing the A-norm:

$$\kappa_A(BA) = \|BA\|_A\|(BA)^{-1}\|_A.$$

Since the matrix BA is A-self-adjoint, it can be shown that, in fact,

$$\kappa_A(BA) = \|BA\|_A\|(BA)^{-1}\|_A = \frac{\lambda_{\max}(BA)}{\lambda_{\min}(BA)},$$

so that the error reduction inequality above can be written more simply as

$$\begin{aligned}\|e^{k+1}\|_A &\leqslant 2\left(\frac{\sqrt{\kappa_A(BA)}-1}{\sqrt{\kappa_A(BA)}+1}\right)^{k+1}\|e^0\|_A \\ &= 2\left(1-\frac{2}{1+\sqrt{\kappa_A(BA)}}\right)^{k+1}\|e^0\|_A.\end{aligned}$$

4. Assume that we would like to achieve the following accuracy in our iteration after some number of steps n:

$$\frac{\|e^{n+1}\|_A}{\|e^0\|_A} < \epsilon.$$

Using the approximation

$$\begin{aligned}\ln\left(\frac{a-1}{a+1}\right) &= \ln\left(\frac{1+(-1/a)}{1-(-1/a)}\right) \\ &= 2\left[\left(\frac{-1}{a}\right)+\frac{1}{3}\left(\frac{-1}{a}\right)^3+\frac{1}{5}\left(\frac{-1}{a}\right)^5+\cdots\right] \\ &< \frac{-2}{a},\end{aligned}$$

show that we can achieve this error tolerance if n satisfies

$$n = \mathcal{O}\left(\kappa_A^{1/2}(BA)\left|\ln\frac{\epsilon}{2}\right|\right).$$

5. Many types of matrices have $\mathcal{O}(1)$ nonzeros per row (for example, finite element and other discretizations of ordinary and partial differential equations.) If A is an $n \times n$ matrix, then the cost of one iteration of CG (Algorithm 10.6.2) will be $\mathcal{O}(n)$, as would one iteration of the basic linear method (Algorithm 10.6.1). What is the overall complexity [in terms of n and $\kappa_A(BA)$] to solve the problem to a given tolerance ϵ? If $\kappa_A(BA)$ can be bounded by a constant, independent of the problem size n, what is the complexity? Is this then an optimal method?

10.6.3 *Properties of the Chebyshev polynomials.*

The Chebyshev polynomials are defined as

$$t_n(x) = \cos(n \cos^{-1} x), \quad n = 0, 1, 2, \ldots.$$

Taking $t_0(x) = 1$, $t_1(x) = x$, it can be shown that the Chebyshev polynomials are an orthogonal family that can be generated by the standard recursion (which holds for any orthogonal polynomial family):

$$t_{n+1}(x) = 2t_1(x)t_n(x) - t_{n-1}(x), \quad n = 1, 2, 3, \ldots.$$

Prove the following extremely useful relationships:

$$t_k(x) = \frac{1}{2}\left[\left(x + \sqrt{x^2 - 1}\right)^k + \left(x - \sqrt{x^2 - 1}\right)^k\right], \quad \forall x, \quad (10.6.33)$$

$$t_k\left(\frac{\alpha + 1}{\alpha - 1}\right) > \frac{1}{2}\left(\frac{\sqrt{\alpha} + 1}{\sqrt{\alpha} - 1}\right)^k, \quad \forall \alpha > 1. \quad (10.6.34)$$

These two results are fundamental in the convergence analysis of the conjugate gradient method in the earlier exercises in the section. [Hint: For the first result, use the fact that $\cos k\theta = (e^{ik\theta} + e^{-ik\theta})/2$. The second result will follow from the first after some algebra.]

10.7 METHODS FOR NONLINEAR EQUATIONS

Building on the material assembled in Section 10.1 on nonlinear equations and calculus in Banach spaces, we now consider some of the classical nonlinear iterations and nonlinear conjugate gradient methods for solving nonlinear equations in finite-dimensional Hilbert spaces. Newton-like methods are then reviewed, including inexact variations and global convergence modifications. We then discuss damped inexact Newton multilevel methods, which involve the coupling of damped Newton methods with linear multilevel methods for approximate solution of the linearized systems. We then combine the damping (or *backtracking*) parameter selection and linear iteration tolerance specification to ensure global superlinear convergence. We also describe nonlinear multilevel methods proposed by Hackbusch and others, which do not involve an outer Newton iteration.

While we only have space to cover a few of the main ideas, our discussion in this section follows closely some of the standard references for nonlinear equations in $\mathbb{R}^n$, such as [78, 140], as well as standard references for generalizations to Banach spaces, such as [108, 188]. For Newton multilevel-type methods, we also follow material from the research monographs [63, 85], as well as the articles [21, 22] and several other references cited in the text.

Standard Methods for Nonlinear Equations in $\mathbb{R}^n$

Let $\mathcal{H}$ be a Hilbert space, endowed with an inner product $(\cdot, \cdot)$ which induces a norm $\|\cdot\|$. Given a map $F\colon \mathcal{H} \to \mathcal{H}$ such that $F(u) = Au + B(u)$, where $B\colon \mathcal{H} \to \mathcal{H}$ is a nonlinear operator and where $A\colon \mathcal{H} \to \mathcal{H}$ is an invertible linear operator, we are interested in solutions to the following mathematically equivalent problems: Find $u \in \mathcal{H}$ such that any of the following hold:

$$F(u) = 0, \tag{10.7.1}$$

$$Au + B(u) = 0, \tag{10.7.2}$$

$$u = T(u), \tag{10.7.3}$$

where

$$F(u) = Au + B(u), \qquad T(u) = -A^{-1}B(u), \tag{10.7.4}$$

with $T\colon \mathcal{H} \to \mathcal{H}$. These three familiar-looking equations also arose at the end of Section 10.1 in our discussions of fixed-point theorems and ordered Banach spaces. In this section, we are interested in iterative algorithms for solving equation (10.7.1) or (10.7.2) in the setting of a finite-dimensional Hilbert space $\mathcal{H}$. We will focus entirely on general iterations of the form

$$u^{i+1} = T(u^i), \tag{10.7.5}$$

where T is as in (10.7.4), or more generally is any mapping which is constructed to have as its fixed point the unique solution u of (10.7.1) and (10.7.2).

The nonlinear extensions of the classical linear methods fit into this framework, as well as the Newton-like methods. Our interest in improved convergence, efficiency, and robustness properties will lead us to damped inexact Newton multilevel methods and nonlinear multilevel methods. We are particularly interested in the nonlinear equations which arise from discretizations of the types of semilinear elliptic partial differential equations we considered in detail in Section 10.4, leading to equations which have the additional structure (10.7.2). It will be useful to consider the following variation of (10.7.2), which obviously can be rewritten in the form of (10.7.2) by suitably redefining the operator B:

$$A_k u_k + B_k(u_k) = f_k. \tag{10.7.6}$$

These types of equations will arise from a box or finite element discretization of the types of semilinear elliptic partial differential equations we encountered in Section 10.4, as discussed in some detail in Section 10.5. The space of grid functions u_k

with values at the nodes of the mesh will be denoted as $\mathcal{U}_k$, and equation (10.7.6) may be interpreted as a nonlinear algebraic equation in the space $\mathcal{U}_k$. Equation (10.7.6) may also be interpreted as an abstract operator equation in the finite element space $\mathcal{V}_k$, as discussed in detail in Sections 10.5 and 10.6. In either case, the operator A_k is necessarily symmetric positive definite (SPD) for the problems and discretization methods we consider, while the form and properties of the nonlinear term $B_k(\cdot)$ depend on the particular problem.

To discuss algorithms for (10.7.6), and in particular multilevel algorithms, we will need a nested sequence of finite-dimensional spaces $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots \mathcal{H}_J \equiv \mathcal{H}$, which are connected by prolongation and restriction operators, as discussed in detail in Section 10.6. We are given the abstract nonlinear problem in the finest space $\mathcal{H}$:

$$\text{Find } u \in \mathcal{H} \text{ such that } Au + B(u) = f, \tag{10.7.7}$$

where $A \in \mathcal{L}(\mathcal{H}, \mathcal{H})$ is SPD and $B(\cdot)\colon \mathcal{H} \to \mathcal{H}$ is a nonlinearity which yields a uniquely solvable problem, and we are interested in iterative algorithms for determining the unique solution u which involves solving problems of the form:

$$\text{Find } u_k \in \mathcal{H}_k \text{ such that } A_k u_k + B_k(u_k) = f_k, \tag{10.7.8}$$

in the coarser spaces $\mathcal{H}_k$ for $1 \leqslant k < J$. When only the finest space $\mathcal{H}$ is employed, we will omit the subscripts on functions, operators, and spaces to simplify the notation.

Nonlinear Extensions of Classical Linear Methods. In this section we review nonlinear conjugate gradient methods and Newton-like methods; we also make a few remarks about extensions of the classical linear methods. We discuss at some length the one-dimensional line search required in the Fletcher-Reeves nonlinear conjugate gradient method, which we will use later for computing a global convergence damping parameter in nonlinear multilevel methods.

In Section 4.8 we discussed three distinct notions of convergence in a Banach or Hilbert space: *strong convergence* (often called simply *convergence*), *weak convergence*, and *weak-** convergence. One result we showed was that strong convergence implies weak convergence (Theorem 4.8.5), although the reverse is generally not true. However, in the setting of practical algorithms for linear and nonlinear equations, which take place in finite-dimensional Banach spaces, these three notions of convergence are all equivalent; therefore, here we are interested simply in *strong convergence* of sequences generated by iterative algorithms. Let X be a Banach space, and for ease of exposition denote the norm on X as $\|\cdot\| = \|\cdot\|_X$. Recall that the sequence $\{u^i\}$ with $u^i \in X$ is said to *converge strongly* to $u \in X$ if $\lim_{i\to\infty} \|u - u^i\| = 0$. In Section 4.8 we also defined several distinct notions of the *rate of (strong) convergence* of a sequence, such as Q-linear, Q-superlinear, Q-order(p), and R-order(p). We are interested primarily here in *fixed point iterations* of the form (10.7.5), where the nonlinear mapping $T(\cdot)$ is the *fixed point mapping*. If $T(\cdot)$ represents some iterative technique for obtaining the solution to a problem, it is important to understand what are necessary or at least sufficient conditions for

this iteration to converge to a solution, and what its convergence properties are (such as *rate*). Recall that in Chapter 4 and also in Section 10.1 we examined *contraction operators*, which are maps $T\colon U \subset X \to U \subset X$ having the property

$$\|T(u) - T(v)\| \leqslant \alpha\|u - v\|, \quad \forall u, v \in U, \quad \alpha \in [0, 1),$$

for some *contraction constant* $\alpha \in [0, 1)$. In Section 4.4 we stated and proved the Banach Fixed-Point Theorem (or Contraction Mapping Theorem). The version of the theorem we gave ensured that a fixed point iteration involving a contraction in a closed subset of a Banach space will converge to a unique fixed point. However, in the proof given in Chapter 4, two results were established as intermediate steps that were not stated as separate conclusions of the theorem, but in fact have importance here; the first is that the convergence rate of the iteration is actually Q-linear, and the second is that the error at each iteration may be bounded by the contraction constant. In Section 10.1 we have restated a version of the Banach Fixed-Point Theorem from Chapter 4, but with these two additional conclusions emphasized.

The classical linear methods discussed in Section 10.6, such as Jacobi and Gauss-Seidel, can be extended in the obvious way to nonlinear algebraic equations of the form (10.7.6). In each case, the method can be viewed as a fixed point iteration of the form (10.7.5). Of course, implementation of these methods, which we refer to as nonlinear Jacobi and nonlinear Gauss-Seidel methods, now requires the solution of a sequence of one-dimensional nonlinear problems for each unknown in one step of the method. A variation that works well, even compared to newer methods, is the nonlinear SOR method. The convergence properties of these types of methods, as well as a myriad of variations and related methods, are discussed in detail in [140]. Note, however, that the same difficulty arising in the linear case also arises here: As the problem size is increased (the mesh size is reduced), these methods converge more and more slowly. As a result, we consider alternative methods, such as nonlinear conjugate gradient methods, Newton-like methods, and nonlinear multilevel methods.

Note that since the one-dimensional problems arising in the nonlinear Jacobi and nonlinear Gauss-Seidel methods are often solved with Newton's method, the methods are also referred to as Jacobi-Newton and Gauss-Seidel-Newton methods, meaning that the Jacobi or Gauss-Seidel iteration is the main or outer iteration, whereas the inner iteration is performed by Newton's method. Momentarily we will consider the other situation: The use of Newton's method as the outer iteration, and a linear iterative method such as multigrid for solution of the linear Jacobian system at each outer Newton iteration. We refer to this method as a Newton multilevel method.

Nonlinear Conjugate Gradient Methods. As we have seen in Sections 10.1 and 10.4, the following minimization problem:

$$\text{Find } u \in \mathcal{H} \text{ such that } J(u) = \min_{v \in \mathcal{H}} J(v), \text{ where } J(u) = \frac{1}{2}(Au, u) + G(u) - (f, u)$$

is equivalent to the associated zero-point problem:

$$\text{Find } u \in \mathcal{H} \text{ such that } F(u) = Au + B(u) - f = 0,$$

where $B(u) = G'(u)$. We assume here that both problems are uniquely solvable. An effective approach for solving the zero-point problem, by exploiting the connection with the minimization problem, is the *Fletcher-Reeves* version [76] of the nonlinear conjugate gradient method, which takes the form:

Algorithm 10.7.1 (Fletcher-Reeves Nonlinear CG Method).

```
Let u^0 ∈ H be given.
r^0 = f − B(u^0) − Au^0,      p^0 = r^0.
Do i = 0, 1, ... until convergence:
     α_i = (see below)
     u^{i+1} = u^i + α_i p^i
     r^{i+1} = r^i + B(u^i) − B(u^{i+1}) − α_i A p^i
     β_{i+1} = (r^{i+1}, r^{i+1})/(r^i, r^i)
     p^{i+1} = r^{i+1} + β_{i+1} p^i
End do.
```

The expression for the residual r^{i+1} is from

$$
\begin{aligned}
r^{i+1} &= -F(u^{i+1}) && (10.7.9)\\
&= f - B(u^{i+1}) - Au^{i+1} && (10.7.10)\\
&= (f - B(u^i) - Au^i) + B(u^i) - B(u^{i+1}) + A(u^i - u^{i+1}) && (10.7.11)\\
&= r^i + B(u^i) - B(u^{i+1}) - \alpha_i A p^i, && (10.7.12)
\end{aligned}
$$

where $u^i - u^{i+1} = \alpha_i p^i$ has been used to obtain the last expresion; this holds by the definition of u^{i+1} in the second step of the "Do Loop" in Algorithm 10.7.1. The directions p^i are computed from the previous direction and the new residual, and the steplength α_i is chosen to minimize the associated functional $J(\cdot)$ in the direction p^i. In other words, α_i is chosen to minimize $J(u^i + \alpha_i p^i)$, which is equivalent to solving the one-dimensional zero-point problem:

$$\frac{dJ(u^i + \alpha_i p^i)}{d\alpha_i} = 0.$$

Given the form of $J(\cdot)$ above, we have that

$$J(u^i + \alpha_i p^i) = \frac{1}{2}(A(u^i + \alpha_i p^i), u^i + \alpha_i p^i) + G(u^i + \alpha_i p^i) - (f, u^i + \alpha_i p^i).$$

A simple differentiation with respect to α_i (and some simplification) gives

$$\frac{dJ(u^i + \alpha_i p^i)}{d\alpha_i} = \alpha_i(Ap^i, p^i) - (r^i, p^i) + (B(u^i + \alpha_i p^i) - B(u^i), p^i),$$

where $r^i = f - B(u^i) - Au^i = -F(u^i)$ is the nonlinear residual. The second derivative with respect to α_i will be useful also, and is easily seen to be

$$\frac{d^2 J(u^i + \alpha_i p^i)}{d\alpha_i^2} = (Ap^i, p^i) + (B'(u^i + \alpha_i p^i)p^i, p^i).$$

Now, Newton's method for solving the zero-point problem for α_i takes the form

$$\alpha_i^{m+1} = \alpha_i^m - \delta^m,$$

where

$$\begin{aligned}\delta^m &= \frac{dJ(u^i + \alpha_i^m p^i)/d\alpha_i}{d^2 J(u^i + \alpha_i^m p^i)/d\alpha_i^2} \\ &= \frac{\alpha_i^m (Ap^i, p^i) - (r^i, p^i) + (B(u^i + \alpha_i^m p^i) - B(u^i), p^i)}{(Ap^i, p^i) + (B'(u^i + \alpha_i^m p^i)p^i, p^i)}.\end{aligned}$$

The quantities (Ap^i, p^i) and (r^i, p^i) can be computed once at the start of each line search for α_i, each requiring an inner product (Ap^i is available from the CG iteration). Each Newton iteration for the new α_i^{m+1} then requires evaluation of the nonlinear term $B(u^i + \alpha_i^m p^i)$ and inner product with p^i, as well as evaluation of the derivative mapping $B'(u^i + \alpha_i p^i)$, application to p^i, followed by inner product with p^i.

In the case that $B(\cdot)$ arises from the discretization of a nonlinear partial differential equation and is of *diagonal form*, meaning that the j-th component function of the vector $B(\cdot)$ is a function of only the j-th component of the vector of nodal values u, or $B_j(u) = B_j(u_j)$, then the resulting Jacobian matrix $B'(\cdot)$ of $B(\cdot)$ is a diagonal matrix. This situation occurs with box-method discretizations or mass-lumped finite element discretizations of semilinear problems. As a result, computing the term $(B'(u^i + \alpha_i p^i)p^i, p^i)$ can be performed with fewer operations than two inner products.

The total cost for each Newton iteration (beyond the first) is then evaluation of $B(\cdot)$ and $B'(\cdot)$, and something less than three inner products. Therefore, the line search can be performed fairly inexpensively in certain situations. If alternative methods are used to solve the one-dimensional problem defining α_i, then evaluation of the Jacobian matrix can be avoided altogether, although as we remarked earlier, the Jacobian matrix is cheaply computable in the particular applications we are interested in here.

Note that if the nonlinear term $B(\cdot)$ is absent, then the zero-point problem is linear and the associated energy functional is quadratic:

$$F(u) = Au - f = 0, \qquad J(u) = \frac{1}{2}(Au, u) - (f, u).$$

In this case, the Fletcher-Reeves CG algorithm reduces to exactly the Hestenes-Stiefel [92] linear conjugate gradient algorithm (Algorithm 10.6.2 with the preconditioner $B = I$). The exact solution to the linear problem $Au = f$, as well as to the associated minimization problem, can be reached in no more than n_k steps (in exact arithmetic, that is), where n_k is the dimension of the space $\mathcal{H}$ (see, for example, [140]). The calculation of the steplength α_i no longer requires the iterative solution of a one-dimensional minimization problem with Newton's method, since

$$\frac{dJ(u^i + \alpha_i p^i)}{d\alpha_i} = \alpha_i (Ap^i, p^i) - (r^i, p^i) = 0$$

yields an explicit expression for the α_i which minimizes the functional J in the direction p^i:

$$\alpha_i = \frac{(r^i, p^i)}{(Ap^i, p^i)}.$$

See Exercise 10.6.1 for a guided derivation of the linear conjugate gradient method from first principles.

Newton's Method. We now consider one of the most powerful techniques for solving nonlinear problems: Newton's method. A classic reference for much of the following material is [78]. Given the nonlinear map $F\colon D \subset \mathcal{H} \to \mathcal{H}$ for some finite-dimensional Hilbert space $\mathcal{H}$, where $F \in C^2(\mathcal{H})$, we can derive Newton's method by starting with the generalized Taylor expansion (Theorem 10.1.2):

$$F(u+h) = F(u) + F'(u)h + \mathcal{O}(\|h\|^2). \tag{10.7.13}$$

One wants to find $u \in D \subset \mathcal{H}$ such that $F(u) = 0$, but have only an initial approximation $u^0 \approx u$. If the Taylor expansion could be used to determine h such that $F(u^0 + h) = 0$, then the problem would be solved by taking $u = u^0 + h$. Although the Taylor expansion is an infinite series in h, we can solve approximately for h by truncating the series after the first two terms, leaving

$$0 = F(u^0 + h) = F(u^0) + F'(u^0)h. \tag{10.7.14}$$

Writing this as an iteration leads to

$$\begin{aligned} F'(u^i)h^i &= -F(u^i) \\ u^{i+1} &= u^i + h^i. \end{aligned}$$

In other words, the Newton iteration is simply the fixed point iteration

$$u^{i+1} = T(u^i) = u^i - F'(u^i)^{-1}F(u^i). \tag{10.7.15}$$

By viewing the Newton iteration as a fixed point iteration, a very general convergence theorem can be proven in a general Banach space X.

Theorem 10.7.1 (Newton Kantorovich Theorem)**.** *Let X be a Banach space, let $D \subset X$ be an open set, and let $F \in C^1(D; X)$. If there exists $u^0 \in D$ and an open ball $B_\rho(u^0) \subset D$ of radius $\rho > 0$ about u^0 such that*

(1) $F'(u^0)$ is nonsingular, with $\|F'(u^0)^{-1}\|_{\mathcal{L}(X,X)} \leqslant \beta$,
(2) $\|u^1 - u^0\|_X = \|F'(u^0)^{-1}F(u^0)\|_X \leqslant \alpha$,
(3) $\|F'(u) - F'(v)\|_X \leqslant \gamma\|u - v\|_X$, $\forall u, v \in B_\rho(u^0)$,
(4) $\alpha\beta\gamma < \frac{1}{2}$, and $\rho \leqslant [1 - \sqrt{1 - 2\alpha\beta\gamma}]/[\beta\gamma]$,

then the Newton iterates produced by (10.7.15) converge strongly at a q-linear rate to a unique $u^* \in B_\rho(u^0) \subset D$.

Proof. See, for example [108, 140, 188]. □

If one assumes the existence of the solution $F(u^*) = 0$, then theorems such as the following one (see also [111]) give an improved rate of convergence.

Theorem 10.7.2 (Quadratic Convergence of Newton's Method)**.** *Let X be a Banach space, let $D \subset X$ be an open set, and let $F \in C^1(D; X)$. If there exists $u^* \in D$ and an open ball $B_\rho(u^*) \subset D$ of radius $\rho > 0$ about u^* such that*

(1) $F(u^*) = 0$,

(2) $F'(u^*)$ *is nonsingular, with* $\|F'(u^*)^{-1}\|_{\mathcal{L}(X,X)} \leqslant \beta$,

(3) $\|F'(u) - F'(v)\|_X \leqslant \gamma\|u - v\|_X$, $\forall u, v \in B_\rho(u^*)$,

(4) $\rho\beta\gamma < \frac{2}{3}$,

then for any $u^0 \in B_\rho(u^*) \subset D$, the Newton iterates produced by (10.7.15) are well-defined and remain in $B_\rho(u^*)$, and converge strongly at a q-quadratic rate to $u^* \in B_\rho(u^0) \subset D$.

Proof. Since by assumption the ball $B_\rho(u^*)$ is already contained in D, we can take $\theta = \rho\beta\gamma < 2/3$ in the Inverse Perturbation Lemma (Lemma 10.1.2), to extend the bound on the inverse of F' in assumption (2) to all of $B_\rho(u^*)$:

$$\|[F'(u)]^{-1}\|_{\mathcal{L}(X,Y)} \leqslant \frac{\beta}{1 - \rho\beta\gamma}, \quad \forall u \in B_\rho(u^*). \tag{10.7.16}$$

We now consider the behavior of the error in the Newton iteration:

$$\begin{aligned} u^{n+1} - u^* &= -[F'(u^n)]^{-1}F(u^n) - u^* \\ &= [F'(u^n)]^{-1}[F(u^*) - F(u^n) - F'(u^n)u^*] \\ &= [F'(u^n)]^{-1}[F(u^n + h) - \{F(u^n) + F'(u^n)(u^n + h)\}], \end{aligned} \tag{10.7.17}$$

where we have defined $h = u^* - u^n$ and used the fact that $F(u^*) = 0$. Taking norms of both sides of (10.7.17) and employing (10.7.16) and the Linear Approximation Lemma (Lemma 10.1.1), gives

$$\begin{aligned} \|u^{n+1} - u^*\|_X &= \|[F'(u^n)]^{-1}[F(u^n + h) - \{F(u^n) + F'(u^n)(u^n + h)\}]\| \\ &= \|[F'(u^n)]^{-1}\|_{\mathcal{L}(X,X)} \\ &\qquad \cdot \|[F(u^n + h) - \{F(u^n) + F'(u^n)(u^n + h)\}]\|_{\mathcal{L}(X,X)} \\ &\leqslant \frac{\beta\gamma}{2(1 - \rho\beta\gamma)}\|u^* - u^n\|_X^2. \end{aligned}$$

Since $\|u^* - u^n\|_X \leqslant \rho$ and

$$0 < \frac{\beta\gamma}{2(1 - \rho\beta\gamma)} \leqslant \frac{1}{\rho}\left(\frac{\rho\beta\gamma}{2(1 - \rho\beta\gamma)}\right) \leqslant \frac{1}{\rho}\left(\frac{1}{2} \cdot \frac{2}{3} \cdot 3\right) \leqslant \frac{1}{\rho},$$

we have $\|u^{n+1} - u^*\|_X \leqslant \|u^* - u^n\|_X \leqslant \rho$, giving $u^{n+1} \in B_\rho(u^*)$, which completes the proof. □

There are several variations of the standard Newton iteration (10.7.15) commonly used for nonlinear algebraic equations which we mention briefly. A *quasi*-Newton method refers to a method which uses an approximation to the true Jacobian matrix for solving the Newton equations. A *truncated*-Newton method uses the true Jacobian matrix in the Newton iteration, but solves the Jacobian system only approximately, using an iterative linear solver in which the iteration is stopped early or *truncated*. These types of methods are referred to collectively as *Inexact* or *approximate* Newton methods, where in the most general case an approximate Newton direction is produced in some unspecified fashion. It can be shown that the convergence behavior of these inexact Newton methods is similar to the standard Newton's method, and theorems similar to (10.7.1) can be established (see [108] and the discussions below).

Global Inexact Newton Iteration

For our purposes here, the inexact Newton approach will be of interest, for the following reasons. First, in the case of semilinear partial differential equations which consist of a leading linear term plus a nonlinear term which does not depend on derivatives of the solution, the nonlinear algebraic equations generated often have the form

$$F(u) = Au + B(u) - f = 0.$$

The matrix A is SPD, and the nonlinear term $B(\cdot)$ is often simple, and in fact is often of *diagonal form*, meaning that the j-th component of the vector function $B(u)$ is a function of only the j-th entry of the vector u, or $B_j(u) = B_j(u_j)$; this occurs, for example, in the case of a box-method discretization, or a mass-lumped finite element discretization of semilinear equations. Further, it is often the case that the derivative $B'(\cdot)$ of the nonlinear term $B(\cdot)$, which will be a diagonal matrix due to the fact that $B(\cdot)$ is of diagonal form, can be computed (and applied to a vector) at low expense. If this is the case, then the true Jacobian matrix is available at low cost:

$$F'(u) = A + B'(u).$$

A second reason for our interest in the inexact Newton approach is that the efficient multilevel methods described in Section 10.6 for the linearized semilinear equations can be used effectively for the Jacobian systems; this is because the Jacobian $F'(u)$ is essentially the linearized semilinear operator, where only the diagonal Helmholtz-like term $B'(\cdot)$ changes from one Newton iteration to the next.

Regarding the assumptions on the function $F(\cdot)$ and the Jacobian $F'(\cdot)$ appearing in Theorem 10.7.1, although they may seem unnatural at first glance, they are essentially the minimal conditions necessary to show that the Newton iteration, viewed as a fixed point iteration, is a *contraction*, so that a contraction argument may be employed (see [108]). Since a contraction argument is used, no assumptions on the existence or uniqueness of a solution are required. A disadvantage of proving Newton convergence through a contraction argument is that only Q-linear convergence is shown. This can be improved to R-quadratic through the idea of *majorization* [108].

If additional assumptions are made, such as the existence of a unique solution, then Q-quadratic convergence can be shown; see [108, 140].

Global Newton Convergence Through Damping. As noted in the preceding section, Newton-like methods converge if the initial approximation is "close" to the solution; different convergence theorems require different notions of closeness. If the initial approximation is close enough to the solution, then superlinear or Q-order(p) convergence occurs. However, the fact that these theorems require a good initial approximation is also indicated in practice: it is well known that Newton's method will converge slowly or fail to converge at all if the initial approximation is not good enough.

On the other hand, methods such as those used for unconstrained minimization can be considered to be "globally" convergent methods, although their convergence rates are often extremely poor. One approach to improving the robustness of a Newton iteration without losing the favorable convergence properties close to the solution is to combine the iteration with a global minimization method. In other words, we can attempt to force global convergence of Newton's method by requiring that

$$\|F(u^{i+1})\| < \|F(u^i)\|,$$

meaning that we require a decrease in the value of the function at each iteration. But this is exactly what global minimization methods, such as the nonlinear conjugate gradient method, attempt to achieve: progress toward the solution at each step.

More formally, we wish to define a minimization problem, such that the solution of the zero-point problem we are interested in also solves the associated minimization problem. Let us define the following two problems:

Problem 1: Find $u \in \mathcal{H}$ such that $F(u) = 0$.
Problem 2: Find $u \in \mathcal{H}$ such that $J(u) = \min_{v \in \mathcal{H}} J(v)$.

We assume that Problem 2 has been defined so that the unique solution to Problem 1 is also the unique solution to Problem 2; note that in general there may not exist a *natural* functional $J(\cdot)$ for a given $F(\cdot)$, although we will see in a moment that it is always possible to construct an appropriate functional $J(\cdot)$.

A *descent direction* for the functional $J(\cdot)$ at the point u is any direction v such that the directional derivative of $J(\cdot)$ at u in the direction v is negative, or $J'(u)(v) = (J'(u), v) < 0$. If v is a descent direction, then it is not difficult to show that there exists some $\lambda > 0$ such that

$$J(u + \lambda v) < J(u). \tag{10.7.18}$$

This follows from generalized Taylor expansion (Theorem 10.1.2), since

$$J(u + \lambda v) = J(u) + \lambda(J'(u), v) + \mathcal{O}(\lambda^2).$$

If λ is sufficiently small and $(J'(u), v) < 0$ holds (v is a descent direction), then clearly $J(u + \lambda v) < J(u)$. In other words, if a descent direction can be found at the

current solution u^i, then an improved solution u^{i+1} can be found for some steplength in the descent direction v, that is, by performing a one-dimensional line search for λ until (10.7.18) is satisfied.

Therefore, if we can show that the Newton direction is a descent direction, then performing a one-dimensional line search in the Newton direction will always guarantee progress toward the solution. In the case that we define the functional as

$$J(u) = \frac{1}{2}\|F(u)\|^2 = \frac{1}{2}(F(u), F(u)),$$

we can show that the Newton direction is a descent direction. While the following result is easy to show for $\mathcal{H} = \mathbb{R}^n$, we showed more generally in Lemma 10.1.3 that it is also true in the general case of an arbitrary Hilbert space when $\|\cdot\| = (\cdot,\cdot)^{1/2}$:

$$J'(u) = F'(u)^T F(u).$$

Now, the Newton direction at u is simply $v = -F'(u)^{-1}F(u)$, so if $F(u) \neq 0$, then

$$(J'(u), v) = -(F'(u)^T F(u), F'(u)^{-1}F(u)) = -(F(u), F(u)) < 0.$$

Therefore, the Newton direction is always a descent direction for this particular choice of $J(\cdot)$, and by the introduction of the damping parameter λ, the Newton iteration can be made globally convergent in the sense described above.

Damped Inexact Newton Multilevel Methods. Given the problem of n_k nonlinear algebraic equations and n_k unknowns

$$F(u) = Au + B(u) - f = 0,$$

for which we desire the solution u, the ideal algorithm for this problem is one that (1) always converges, and (2) has optimal complexity, which in this case means $\mathcal{O}(n_k)$.

As we have just seen, Newton's method can be made essentially globally convergent with the introduction of a damping parameter. In addition, close to the root, Newton's method has at least superlinear convergence properties. If a method with linear convergence properties is used to solve the Jacobian systems at each Newton iteration, and the complexity of the linear solver is the dominant cost of each Newton iteration, then the complexity properties of the linear method will determine the complexity of the resulting Newton iteration asymptotically, as long as the number of Newton iterations does not grow with the size of the discretization. This last property can in fact be shown for Newton iterations; see [4].

We have discussed in detail in Section 10.6 the convergence and complexity properties of multilevel methods; in many situations they can be shown to have optimal complexity, and in many others this behavior can be demonstrated empirically. With an efficient inexact solver such as a multilevel method for the early damped iterations, employing a more stringent tolerance for the later iterations as the root is approached, a very efficient yet robust nonlinear iteration should result. Following [21, 22], here we combine the robust damped Newton methods with the fast linear multilevel solvers developed in Section 10.6 for inexact solution of the Jacobian systems.

The conditions for linear solver tolerance to ensure superlinear convergence have been given in [60, 61]. Guidelines for choosing damping parameters to ensure global convergence and yet allow for superlinear convergence have been established in [21]. Combination with linear multilevel iterative methods for the semiconductor problem has been considered in [22], along with questions of complexity. We outline the basic algorithm below, specializing it to the particular form of a nonlinear problem of interest. We then give some results on damping and inexactness tolerance selection strategies.

We restrict our discussion here to the following nonlinear problem, which has arisen, for example, from the discretization of a nonlinear elliptic problem:

$$F(u) = Au + B(u) - f = 0.$$

The derivative has the form

$$F'(u) = A + B'(u).$$

The damped inexact Newton iteration for this problem takes the form:

Algorithm 10.7.2 (Damped Inexact Newton Method).

$$\begin{array}{ll} \left[A + B'(u^i)\right] v^i = f - Au^i - B(u^i). & \texttt{[Inexact solve]} \\ u^{i+1} = u^i + \lambda_i v^i. & \texttt{[Correction]} \end{array}$$

We can employ the linear multilevel methods of Section 10.6 in step (1) of Algorithm 10.7.2. A convergence analysis of the undamped method is given in [85]. A detailed convergence analysis of the damped method is given in [22]. Below, we outline what guidelines exist for selection of the damping parameters and the linear iteration tolerance.

Note that due to the special form of the nonlinear operator, the damping step can be implemented in a surprisingly efficient manner. During the one-dimensional line search for the parameter λ_i, we continually check for satisfaction of the inequality

$$\|F(u^i + \lambda_i v^i)\| < \|F(u^i)\|.$$

The term on the right is available from the previous Newton iteration. The term on the left, although it might appear to involve computing the full nonlinear residual, in fact can avoid the operator-vector product contributed by the linear term. Simply note that

$$\begin{aligned} F(u^i + \lambda_i v^i) &= A[u^i + \lambda_i v^i] + B(u^i + \lambda_i v^i) - f \\ &= [Au^i - f] + \lambda_i [Av^i] + B(u^i + \lambda_i v^i). \end{aligned}$$

The term $[Au^i - f]$ is available from the previous Newton iteration, and $[Av^i]$ needs to be computed only once at each Newton step. Computing $F(u^i + \lambda_i v^i)$ for each damping step beyond the first requires only the operation $[Au^i - f] + \lambda_i [Av^i]$ for the new damping parameter λ_i, and evaluation of the nonlinear term at the new damped solution, $B(u^i + \lambda_i v^i)$.

Local and Global Superlinear Convergence. Quasi-Newton methods are studied in [61], and a "characterization" theorem is established for the sequence of approximate Jacobian systems. This theorem establishes sufficient conditions on the sequence $\{B_i\}$, where $B_i \approx F'$, to ensure superlinear convergence of a quasi-Newton method. An interesting result which they obtained is that the "consistency" condition is not required, meaning that the sequence $\{B_i\}$ need not converge to the true Jacobian $F'(\cdot)$ at the root of the equation $F(u) = 0$, and superlinear convergence can still be obtained.

In [61], a characterization theorem shows essentially that the full or true Newton step must be approached, asymptotically, in both length and direction, to attain superlinear convergence in a quasi-Newton iteration.

Inexact Newton methods are studied directly in [60]. Their motivation is the use of iterative solution methods for approximate solution of the true Jacobian systems. They establish conditions on the accuracy of the inexact Jacobian solution at each Newton iteration which will ensure superlinear convergence. The inexact Newton method is analyzed in the form

$$F'(u^i)v^i = -F(u^i) + r^i, \qquad \frac{\|r^i\|}{\|F(u^i)\|} \leqslant \eta_i,$$
$$u^{i+1} = u^i + v^i.$$

In other words, the quantity r^i, which is simply the residual of the Jacobian linear system, indicates the inexactness allowed in the approximate solution of the linear system, and is exactly what one would monitor in a linear iterative solver. It is established that if the *forcing sequence* $\eta_i < 1$ for all i, then the method above is locally convergent. Their main result is the following theorem.

Theorem 10.7.3 (Dembo-Eisenstat-Steihaug)**.** *Assume that there exists a unique u^* such that $F(u^*) = 0$, that $F(\cdot)$ is continuously differentiable in a neighborhood of u^*, that $F'(u^*)$ is nonsingular, and that the inexact Newton iterates $\{u^i\}$ converge to u^*. Then:*

(1) The convergence rate is superlinear if $\lim_{i\to\infty} \eta_i = 0$.

(2) The convergence rate is Q-order at least $1 + p$ if $F'(u^*)$ is Hölder continuous with exponent p, and

$$\eta_i = \mathcal{O}(\|F(u^i)\|^p), \text{ as } i \to \infty.$$

(3) The convergence rate is R-order at least $1 + p$ if $F'(u^*)$ is Hölder continuous with exponent p, and if $\{\eta_i\} \to 0$ with R-order at least $1 + p$.

Proof. See [60]. □

As a result of this theorem, they suggest the tolerance rule:

$$\eta_i = \min\left\{\frac{1}{2}, C\|F(u^i)\|^p\right\}, \quad 0 < p \leqslant 1, \tag{10.7.19}$$

which guarantees Q-order convergence of at least $1 + p$; a similar criterion is

$$\eta_i = \min\left\{\frac{1}{i}, \|F(u^i)\|^p\right\}, \quad 0 < p \leqslant 1. \tag{10.7.20}$$

Necessary and Sufficient Conditions for Inexact Descent. Note the following subtle point regarding the combination of inexact Newton methods and damping procedures for obtaining global convergence properties: Only the *exact* Newton direction is guaranteed to be a descent direction. Once inexactness is introduced into the Newton direction, there is no guarantee that damping will achieve global convergence in the sense outlined above. However, the following simple result gives a necessary and sufficient condition on the tolerance of the Jacobian system solution for the inexact Newton direction to be a descent direction.

Theorem 10.7.4. *The inexact Newton method (Algorithm 10.7.2) for $F(u) = 0$ will generate a descent direction v at the point u if and only if the residual of the Jacobian system $r = F'(u)v + F(u)$ satisfies*

$$(F(u), r) < (F(u), F(u)).$$

Proof. (See, for example, [101].) We remarked earlier that an equivalent minimization problem (appropriate for Newton's method) to associate with the zero point problem $F(u) = 0$ is given by $\min_{u\in\mathcal{H}} J(u)$, where $J(u) = (F(u), F(u))/2$. We also noted that the derivative of $J(u)$ can be written as $J'(u) = F'(u)^T F(u)$. Now, the direction v is a descent direction for $J(u)$ if and only if $(J'(u), v) < 0$. The exact Newton direction is $v = -F'(u)^{-1}F(u)$, and as shown earlier is always a descent direction. Consider now the inexact direction satisfying

$$F'(u)v = -F(u) + r \quad \text{or} \quad v = F'(u)^{-1}[r - F(u)].$$

This inexact direction is a descent direction if and only if:

$$\begin{aligned}(J'(u), v) &= (F'(u)^T F(u), F'(u)^{-1}[r - F(u)]) \\ &= (F(u), r - F(u)) \\ &= (F(u), r) - (F(u), F(u)) \\ &< 0,\end{aligned}$$

which is true if and only if the residual of the Jacobian system r satisfies

$$(F(u), r) < (F(u), F(u)).$$

□

This leads to the following very simple sufficient condition for descent.

Corollary 10.7.1. *The inexact Newton method (Algorithm 10.7.2) for $F(u) = 0$ yields a descent direction v at the point u if the residual of the Jacobian system $r = F'(u)v + F(u)$ satisfies*

$$\|r\| < \|F(u)\|.$$

Proof. (See, for example, [101].) From the proof of Theorem 10.7.4 we have

$$(J'(u), v) = (F(u), r) - (F(u), F(u)) \leqslant \|F(u)\|\|r\| - \|F(u)\|^2,$$

where we have employed the Cauchy-Schwarz inequality. Therefore, if we have $\|r\| < \|F(u)\|$, then the rightmost term is clearly negative [unless $F(u) = 0$], so that v is a descent direction. □

Note that most stopping criteria for the Newton iteration involve evaluating $F(\cdot)$ at the previous Newton iterate u^i. The quantity $F(u^i)$ will have been computed during the computation of the previous Newton iterate u^i, and the tolerance for u^{i+1} which guarantees descent requires that $(F(u^i), r) < (F(u^i), F(u^i))$ by Theorem 10.7.4. This involves only the inner product of r and $F(u^i)$, so that enforcing this tolerance requires only an additional inner product during the Jacobian linear system solve, which for n_k unknowns introduces an additional n_k multiplications and n_k additions. In fact, a scheme may be employed in which only a residual tolerance requirement for superlinear convergence is checked until an iteration is reached in which it is satisfied. At this point, the descent direction tolerance requirement can be checked, and additional iterations will proceed with this descent stopping criterion until it too is satisfied. If the linear solver reduces the norm of the residual monotonically (such as any of the linear methods of Section 10.6), then the first stopping criterion need not be checked again.

In other words, this adaptive Jacobian system stopping criterion, enforcing a tolerance on the residual for local superlinear convergence *and* ensuring a descent direction at each Newton iteration, can be implemented at the same computational cost as a simple check on the norm of the residual of the Jacobian system.

Alternatively, the sufficient condition given in Corollary 10.7.1 may be employed at no additional cost, since only the norm of the residual needs to be computed, which is also what is required to ensure superlinear convergence using Theorem 10.7.3.

Global Superlinear Convergence. In [21], an analysis of inexact Newton methods is performed, where a damping parameter has been introduced. Their goal was to establish selection strategies for both the linear solve tolerance and the damping parameters at each Newton iteration, in an attempt to achieve global superlinear convergence of the damped inexact Newton iteration. It was established, similar to the result in [61], that the Jacobian system solve tolerance must converge to zero (exact solve in the limit), and the damping parameters must converge to 1 (the full Newton step in the limit), for superlinear convergence to be achieved. There are several technical assumptions on the function $F(\cdot)$ and the Jacobian $F'(\cdot)$ in their paper; we summarize one of their main results in the following theorem, as it applies to the inexact Newton framework we have constructed in this section.

Theorem 10.7.5 (Bank and Rose)**.** *Suppose that $F\colon D \subset \mathcal{H} \to \mathcal{H}$ is a homeomorphism on $\mathcal{H}$. Assume also that $F(\cdot)$ is differentiable on closed bounded sets D, that $F'(u)$ is nonsingular and uniformly Lipschitz continuous on such sets D, and that the closed level set*

$$S_o = \{u \mid \|F(u)\| \leqslant \|F(u^0)\|\}$$

is a bounded set. Suppose now that the forcing and damping parameters η_i and λ_i satisfy

$$\eta_i \leqslant C\|F(x^i)\|^p, \quad \eta_i \leqslant \eta_0, \quad \eta_0 \in (0,1),$$

$$\lambda_i = \frac{1}{1 + K_i\|F(x^i)\|}, \quad 0 \leqslant K_i \leqslant K_0, \quad \textit{so that} \quad \lambda_i \leqslant 1.$$

Then there exists $u^* \in \mathcal{H}$ such that $F(u^*) = 0$, and with any $u^0 \in \mathcal{H}$, the sequence generated by the damped inexact Newton method

$$F'(u^i)v^i = -F(u^i) + r^i, \qquad \frac{\|r^i\|}{\|F(u^i)\|} \leqslant \eta_i, \tag{10.7.21}$$

$$u^{i+1} = u^i + \lambda_i v^i, \tag{10.7.22}$$

converges to $u^* \in S_0 \subset \mathcal{H}$. In addition, on the set S_0, the sequence $\{u^i\}$ converges to u^* at rate Q-order at least $1 + p$.

Proof. See [22]. □

Note that by forcing $\eta_i \leqslant \eta_0 < 1$, it happens that the residual of the Jacobian system in Theorem 10.7.5 satisfies $\|r^i\| \leqslant \eta_i\|F(u^i)\| \leqslant \|F(u^i)\|$, which by Corollary 10.7.1 always ensures that the inexact Newton direction produced by their algorithm is a descent direction. The sequence $\{K_i\}$ is then selected so that each parameter is larger than a certain quantity [inequality 2.14 in [22]], which is a guarantee that an appropriate steplength for actual descent is achieved, without line search. We remark that there is also a weaker convergence result in [22] which essentially states that the convergence rate of the damped inexact Newton method above is R-linear or Q-order$(1 + p)$ on certain sets which are slightly more general than the set S_0. The parameter selection strategy suggested in [22] based on Theorem 10.7.5 is referred to as *Algorithm Global*. The idea of the algorithm is to avoid the typical searching strategies required for other global methods by employing the sequence K_i above.

Backtracking for Sufficient Descent. One of the standard choices for backtracking in Algorithm 10.7.2 to ensure global convergence is to successively reduce the size of the damping parameter λ_i at step i of the Newton iteration according to

$$\lambda_i = \frac{1}{2^k}, \quad k = 0, 1, 2, \ldots,$$

where k is incrementally increased from $k = 0$ (giving the full Newton step with $\lambda_i = 1$) to a sufficiently large number until descent (10.7.18) occurs. However, consider the following example from [78]. Let $F\colon \mathbb{R} \to \mathbb{R}$ be given as $F(u) = u$, and

take $\lambda_i = 1/2^{i+1}$, with $u^0 = 2$. The Newton direction at each step remains constant at $v^i = u^i$, $\forall i$, which generates the sequence $\{2, 1, 3/4, 27/32, \ldots\}$, converging to approximately 0.58, yet the solution to $F(u) = 0$ in this case is $u = 0$. The failure of convergence is due to the damping; it is a result of $\lambda_i \to 0$ while $v^i \to v \neq 0$.

To avoid this problem of stalling during the damping procedure, one can enforce a stronger *sufficient descent condition*. By analyzing a linear model of F (see [78]), one can show if $F \in C^1(\mathcal{H})$, then for a fixed $\mu \in (0, 1)$, the following condition can always be satisfied for $\lambda_i \in (0, 1]$ sufficiently small:

$$\|F(u^i + \lambda_i v^i)\| \leqslant (1 - \lambda_i \mu)\|F(u^i)\|. \tag{10.7.23}$$

The result of enforcing this condition is that if $\lambda_i \not\to 0$, then descent cannot stall unless $\|F(u^i)\| \to 0$.

We now describe a globally convergent inexact Newton algorithm that is fairly easy to understand and implement, motivated by the simple necessary and sufficient descent conditions established in the preceding section, as well as the stronger sufficient descent condition described above.

Algorithm 10.7.3 (Damped Inexact Newton method).

```
Do:

     F'(u^i)v^i = -F(u^i) + r^i,   TEST(r^i) = TRUE,      [Inexact solve]
     u^{i+1} = u^i + λ_i v^i,                             [Correction]

where parameters λ_i and TEST(r^i) are defined as:
     (1) TEST(r^i):
          If:   ||r^i|| ⩽ C||F(u^i)||^{p+1},   C > 0,   p > 0,
          And:  (F(u^i), r^i) < (F(u^i), F(u^i)),
          Then: TEST ≡ TRUE;
          Else: TEST ≡ FALSE.
     (2) For fixed μ ∈ (0,1), find λ_i by line search so that:
              ||F(u^i + λ_i v^i)|| ⩽ (1 - λ_i μ)||F(u^i)||.
          Always possible if TEST(r^i) = TRUE.
          Full inexact Newton step λ = 1 always tried first.

An alternative TEST(r^i) is as follows:
     (1') TEST(r^i):
          If:   ||r^i|| ⩽ C||F(u^i)||^{p+1},   C > 0,   p > 0,
          And:  ||r^i|| < ||F(u^i)||,
          Then: TEST ≡ TRUE;
          Else: TEST ≡ FALSE.
```

In Algorithm 10.7.3, the damping parameters λ_i selected in (2) ensure the enforcement of the stronger sufficient descent condition described above, to avoid having the backtracking procedure stall before reaching the solution. The second condition in (1) is the necessary and sufficient condition for the inexact Newton direction to be a descent direction, established in Theorem 10.7.4. The second condition in (1′) of Algorithm 10.7.3 is the weaker sufficient condition established in Corollary 10.7.1. Note that in early iterations when Q-order$(1 + p)$ for $p > 0$ is not to be expected,

just satisfying one of the descent conditions is (necessary and) sufficient for progress toward the solution. The condition $\eta_i < 1$ in Theorem 10.7.5 implies that the inexact Newton directions produced by the algorithm are, by Corollary 10.7.1, descent directions. Algorithm 10.7.3 decouples the descent and superlinear convergence conditions and would allow for the use of only the weakest possible test of $(F(u^i), r^i) < (F(u^i), F(u^i))$ far from the solution, ensuring progress toward the solution with the least amount of work per Newton step.

Note also that the Q-order$(1 + p)$ condition

$$\|r^i\| \leqslant C\|F(u^i)\|^{p+1}$$

does *not* guarantee a descent direction, so that it is indeed important to satisfy the descent condition separately. The Q-order$(1 + p)$ condition *will* impose descent if

$$C\|F(u^i)\|^{p+1} < \|F(u^i)\|,$$

which does not always hold. If one is close to the solution, so that $\|F(u^i)\| < 1$, and if $C \leqslant 1$, then the Q-order$(1 + p)$ condition will imply descent. By this last comment, we see that if $\|F(u^i)\| < 1$ and $C \leq 1$, then the full inexact Newton step is a descent direction, and since we attempt this step first, we see that the algorithm reduces to the algorithm studied in [60] near the solution; therefore, Theorem 10.7.3 applies to Algorithm 10.7.3 near the solution without modification.

Nonlinear Multilevel Methods.

Nonlinear multilevel methods were developed originally in [40, 83]. These methods attempt to avoid Newton linearization by accelerating nonlinear relaxation methods with multiple coarse problems. We are again concerned with the problem

$$F(u) = Au + B(u) - f = 0.$$

Let us introduce the notation $M(\cdot) = A + B(\cdot)$, which yields the equivalent problem:

$$M(u) = f.$$

While there is no direct analogue of the linear error equation in the case of a nonlinear operator $M(\cdot)$, a modified equation for e^i can be used. Given an approximation u^i to the true solution u at iteration i, the equations

$$r^i = f - M(u^i), \qquad M(u) = M(u^i + e^i) = f,$$

where r^i and e^i are the residual and error, give rise to the expressions

$$u^i = M^{-1}(f - r^i), \qquad e^i = M^{-1}(f) - u^i,$$

which together give an expression for the error:

$$e^i = (u^i + e^i) - u^i = M^{-1}(f) - M^{-1}(f - r^i).$$

This expression can be used to develop two- and multiple-level methods as in the linear case.

Nonlinear Two-Level Methods. Consider now the case of two nested finite-dimensional spaces $\mathcal{H}_{k-1} \subset \mathcal{H}_k$, where $\mathcal{H}_k$ is the fine space and $\mathcal{H}_{k-1}$ is a lower-dimensional coarse space, connected by a prolongation operator $I_{k-1}^k \colon \mathcal{H}_{k-1} \to \mathcal{H}_k$ and a restriction operator $I_k^{k-1} \colon \mathcal{H}_k \to \mathcal{H}_{k-1}$. These spaces may, for example, correspond to either the finite element spaces $\mathcal{V}_k$ or the grid function spaces $\mathcal{U}_k$ arising from the discretization of a nonlinear elliptic problem on two successively refined meshes, as discussed above.

Assuming that the error can be smoothed efficiently as in the linear case, then the error equation can be solved in the coarser space. If the solution is transferred to the coarse space as $u_{k-1}^i = I_k^{k-1} u_k^i$, then the coarse space source function can be formed as $f_{k-1} = M_{k-1}(u_{k-1}^i)$. Transferring the residual r_k to the coarse space as $r_{k-1}^i = I_k^{k-1} r_k^i$, the error equation can then be solved in the coarse space as

$$e_{k-1}^i = I_k^{k-1} u_k^i - M_{k-1}^{-1}(M_{k-1}(I_k^{k-1} u_k^i) - I_k^{k-1} r_k^i).$$

The solution is corrected as

$$\begin{aligned} u_k^{i+1} &= u_k^i + I_{k-1}^k e_{k-1}^i \\ &= u_k^i + I_{k-1}^k [I_k^{k-1} u_k^i - M_{k-1}^{-1}(M_{k-1}(I_k^{k-1} u_k^i) - I_k^{k-1}[f_k - M_k(u_k^i)])] \\ &= K_k(u_k^i, f_k). \end{aligned}$$

Therefore, the nonlinear coarse space correction can be viewed as a fixed point iteration.

The algorithm implementing the nonlinear error equation is known as the *full approximation scheme* [40] or the *nonlinear multigrid method* [85]. The two-level version of this iteration can be formulated as:

Algorithm 10.7.4 (Nonlinear Two-Level Method).

$v_k = K_k(u_k^i, f_k)$. `[Correction]`
$u_k^{i+1} = S_k(v_k, f_k)$. `[Post-smoothing]`

Algorithm 10.7.4 will require a nonlinear relaxation operator $S_k(\cdot)$ in step (2), and restriction and prolongation operators as in the linear case, as well as the solution of the nonlinear coarse space equations, to apply the mapping $K_k(\cdot)$ in step (1).

Nonlinear Multilevel Methods. We consider now a nested sequence of finite-dimensional spaces $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots \subset \mathcal{H}_J \equiv \mathcal{H}$, where $\mathcal{H}_J$ is the finest space and $\mathcal{H}_1$ the coarsest space, each space being connected to the others via prolongation and restriction operators, as discussed above.

The *multi*-level version of Algorithm 10.7.4 would employ another two-level method to solve the coarse space problem in step (1), and can be described recursively as follows:

Algorithm 10.7.5 (Nonlinear Multilevel Method).

Do:

$u^{i+1} = NML(J, u^i, f)$.

where $u_k^{\text{NEW}} = NML(k, u_k^{\text{OLD}}, f_k)$ is defined recursively:

If $(k = 1)$ Then:
$u_1^{\text{NEW}} = M_1^{-1}(f_1)$. [Direct solve]
Else:
$r_{k-1} = I_k^{k-1}(f_k - M_k(u_k^{\text{OLD}}))$, [Restrict residual]
$u_{k-1} = I_k^{k-1} u_k^{\text{OLD}}$ [Restrict solution]
$f_{k-1} = M_{k-1}(u_{k-1}) - r_{k-1}$ [Coarse source]
$w_{k-1} = u_{k-1} - NML(k-1, u_{k-1}, f_{k-1})$ [Coarse solution]
$w_k = I_{k-1}^k w_{k-1}$ [Coarse correction]
$\lambda =$ (see below) [Damping parameter]
$v_k = u_k^{\text{OLD}} + \lambda w_k$ [Correction]
$u_k^{\text{NEW}} = S_k(v_k, f_k)$. [Post-smoothing]
End.

The practical aspects of this algorithm and variations are discussed in [40]. A convergence theory has been discussed in [85] and in the sequence of papers [88, 146].

Damping Parameter. Note that we have introduced a damping parameter λ in the coarse space correction step of Algorithm 10.7.5, analogous to the damped inexact Newton multilevel method discussed earlier. In fact, without this damping parameter, the algorithm fails for difficult problems such as those with exponential or rapid nonlinearities (this is also true for the Newton iteration without damping).

To explain how the damping parameter is chosen, we refer back to the earlier discussion of nonlinear conjugate gradient methods. We begin with the following energy functional:

$$J_k(u_k) = \frac{1}{2}(A_k u_k, u_k)_k + B_k(u_k) - (f_k, u_k)_k.$$

As we have seen, the resulting minimization problem:

$$\text{Find } u_k \in \mathcal{H}_k \text{ such that } J_k(u_k) = \min_{v_k \in \mathcal{H}_k} J_k(v_k)$$

is equivalent to the associated zero-point problem:

$$\text{Find } u_k \in \mathcal{H}_k \text{ such that } F_k(u_k) = A_k u_k + B_k(u_k) - f_k = 0,$$

where $B_k(u_k) = G_k'(u_k)$. In other words, $F_k(\cdot)$ is a gradient mapping of the associated energy functional $J_k(\cdot)$, where we assume that both problems above are uniquely solvable.

In [88] it is shown [with suitable conditions on the nonlinear term $B_k(\cdot)$ and satisfaction of a nonlinear form of the variational conditions] that the prolongated

coarse space correction $w_k = I_{k-1}^k w_{k-1}$ is a descent direction for the functional $J_k(\cdot)$. Therefore, there exists some $\lambda > 0$ such that

$$J_k(u_k + \lambda w_k) < J_k(u_k).$$

Minimization of $J_k(\cdot)$ along the descent direction w_k is equivalent to solving the following one-dimensional problem:

$$\frac{dJ(u_k + \lambda w_k)}{d\lambda} = 0.$$

As in the discussion of the nonlinear conjugate gradient method, the one-dimensional problem can be solved with Newton's method:

$$\lambda^{m+1} = \lambda^m - \frac{\lambda^m (A_k w_k, w_k)_k - (r_k, w_k)_k + (B_k(u_k + \lambda^m w_k) - B_k(u_k), w_k)_k}{(A_k w_k, w_k)_k + (B_k'(u_k + \lambda^m w_k) w_k, w_k)_k}.$$

Now, recall that the "direction" from the coarse space correction has the form $w_k = I_{k-1}^k w_{k-1}$. Defining the quantities

$$\begin{aligned} A_1 &= \lambda^m (A_k I_{k-1}^k w_{k-1}, I_{k-1}^k w_{k-1})_k, \\ A_2 &= (r_k, I_{k-1}^k w_{k-1})_k, \\ A_3 &= (B_k(u_k + \lambda^m I_{k-1}^k w_{k-1}) - B_k(u_k), I_{k-1}^k w_{k-1})_k, \end{aligned}$$

the Newton correction for λ then takes the form

$$\frac{A_1 - A_2 + A_3}{(A_k I_{k-1}^k w_{k-1}, I_{k-1}^k w_{k-1})_k + (B_k'(u_k + \lambda^m I_{k-1}^k w_{k-1}) I_{k-1}^k w_{k-1}, I_{k-1}^k w_{k-1})_k}.$$

If the linear variational conditions are satisfied:

$$A_{k-1} = I_k^{k-1} A_k I_{k-1}^k, \qquad I_k^{k-1} = (I_{k-1}^k)^T, \tag{10.7.24}$$

and we define the quantities

$$\begin{aligned} B_1 &= \lambda^m (A_{k-1} w_{k-1}, w_{k-1})_{k-1}, \\ B_2 &= (r_{k-1}, w_{k-1})_{k-1}, \\ B_3 &= (I_k^{k-1}(B_k(u_k + \lambda^m I_{k-1}^k w_{k-1}) - B_k(u_k)), w_{k-1})_{k-1}, \end{aligned}$$

then this expression becomes

$$\frac{B_1 - B_2 + B_3}{(A_{k-1} w_{k-1}, w_{k-1})_{k-1} + (I_k^{k-1} B_k'(u_k + \lambda^m I_{k-1}^k w_{k-1}) I_{k-1}^k w_{k-1}, w_{k-1})_{k-1}}.$$

It can be shown [88] that as in the linear case, a conforming finite element discretization of the nonlinear elliptic problem we are considering, on two successively refined meshes, satisfies the following so-called *nonlinear variational conditions*:

$$A_{k-1} + B_{k-1}(\cdot) = I_k^{k-1} A_k I_{k-1}^k + I_k^{k-1} B_k(I_{k-1}^k \cdot), \qquad I_k^{k-1} = (I_{k-1}^k)^T. \tag{10.7.25}$$

As in the linear case, these conditions are usually required [88] to show theoretical convergence results about nonlinear multilevel methods. Unfortunately, unlike the linear case, there does not appear to be a way to enforce these conditions algebraically [at least for the strictly nonlinear term $B_k(\cdot)$] in an efficient way. Therefore, if we employ discretization methods other than finite element methods, or cannot approximate the integrals accurately (such as if discontinuities occur within elements on coarser levels) for assembling the discrete nonlinear system, then the variational conditions will be violated. With the algebraic approach, we will have to be satisfied with violation of the nonlinear variational conditions, at least for the strictly nonlinear term $B_k(\cdot)$, in the case of the nonlinear multilevel method.

In [88] an expression is given for λ in an attempt to avoid solving the one-dimensional minimization problem. Certain norm estimates are required in their expression for λ, which depends on the particular nonlinearity; therefore, the full line search approach may be more robust, although more costly. There is an interesting result regarding the damping parameter in the linear case, first noticed in [88]. If the nonlinear term $B(\cdot)$ is absent, the zero-point problem is linear and the associated energy functional is quadratic:

$$F_k(u_k) = A_k u_k - f_k = 0, \qquad J_k(u_k) = \frac{1}{2}(A_k u_k, u_k)_k - (f_k, u_k)_k.$$

As in the conjugate gradient algorithm, the calculation of the steplength λ no longer requires the iterative solution of a one-dimensional minimization problem with Newton's method, since

$$\frac{dJ(u_k + \lambda w_k)}{d\lambda} = \lambda(A_k w_k, w_k)_k - (r_k, w_k)_k = 0$$

yields an explicit expression for λ which minimizes the functional $J_k(\cdot)$ in the direction w_k:

$$\lambda = \frac{(r_k, w_k)_k}{(A_k w_k, w_k)_k}.$$

Since $w_k = I_{k-1}^k w_{k-1}$, we have that

$$\begin{aligned}
\lambda &= \frac{(r_k, w_k)_k}{(A_k w_k, w_k)_k} \\
&= \frac{(r_k, I_{k-1}^k w_{k-1})_k}{(A_k I_{k-1}^k w_{k-1}, I_{k-1}^k w_{k-1})_k} \\
&= \frac{((I_{k-1}^k)^T r_k, w_{k-1})_{k-1}}{((I_{k-1}^k)^T A_k I_{k-1}^k w_{k-1}, w_{k-1})_{k-1}}.
\end{aligned}$$

Therefore, if the variational conditions (10.7.24) are satisfied, the damping parameter can be computed cheaply with only coarse space quantities:

$$\lambda = \frac{(I_k^{k-1} r_k, w_{k-1})_{k-1}}{(I_k^{k-1} A_k I_{k-1}^k w_{k-1}, w_{k-1})_{k-1}} = \frac{(r_{k-1}, w_{k-1})_{k-1}}{(A_{k-1} w_{k-1}, w_{k-1})_{k-1}}.$$

Note that in the two-level case, $w_{k-1} = A_{k-1}^{-1} r_{k-1}$, so that $\lambda = 1$ always holds. Otherwise, numerical experiments show that $\lambda \geqslant 1$, and it is argued [88] that this is always the case. Adding the parameter λ to the linear multilevel algorithms of Section 10.6 guarantees that the associated functional $J_k(\cdot)$ is minimized along the direction defined by the coarse space correction. A simple numerical example in [88] illustrates that, in fact, the convergence rate of the linear algorithm can be improved to a surprising degree by employing the damping parameter.

Stopping Criteria for Nonlinear Iterations.

As in a linear iteration, there are several quantities which can be monitored during a nonlinear iteration to determine whether a sufficiently accurate approximation u^{i+1} to the true solution u^* has been obtained. Possible choices, with respect to any norm $\|\cdot\|$, include:

(1) Nonlinear residual:	$\|F(u^{i+1})\|$	$\leqslant FTOL$
(2) Relative residual:	$\|F(u^{i+1})\|/\|F(u^0)\|$	$\leqslant RFTOL$
(3) Iterate change:	$\|u^{i+1} - u^i\|$	$\leqslant UTOL$
(4) Relative change:	$\|u^{i+1} - u^i\|/\|u^{i+1}\|$	$\leqslant RUTOL$
(5) Contraction estimate:	$\|u^{i+1} - u^i\|/\|u^i - u^{i-1}\|$	$\leqslant CTOL.$

We also mention a sixth option, which attempts to obtain an error estimate from the Contraction Mapping Theorem (Theorem 10.1.14) by estimating the contraction constant α of the nonlinear fixed point mapping $T(\cdot)$ associated with the iteration. The constant is estimated as follows:

$$\alpha = \frac{\|u^{i+1} - u^i\|}{\|u^i - u^{i-1}\|} = \frac{\|T(u^i) - T(u^{i-1})\|}{\|u^i - u^{i-1}\|},$$

and the Contraction Mapping Theorem gives the error estimate-based criterion:

$$\text{(6) Error estimate: } \|u^* - u^{i+1}\| \leqslant \frac{\alpha}{1-\alpha}\|u^{i+1} - u^i\| \leqslant ETOL.$$

There are certain difficulties with employing any of these conditions alone. For example, if the iteration has temporarily stalled, then criteria (3) and (4) would prematurely halt the iteration. On the other hand, if the scaling of the function $F(\cdot)$ is such that $\|F(\cdot)\|$ is always very small, then criterion (1) could halt the iteration early. Criterion (2) attempts to alleviate this problem in much the same way as a relative stopping criterion in the linear case. However, if the initial approximation u^0 is such that $\|F(u^0)\|$ is extremely large, then (3) could falsely indicate that a good approximation has been reached. Criterion (5) cannot be used to halt the iteration alone, as it gives no information about the quality of the approximation; it would be useful in a Newton iteration to detect when the region of fast convergence has been entered.

Criterion (6) may be the most reliable stand-alone criterion, although it depends on the accuracy of the contraction number estimate. If the contraction number is constant (linear convergence) over many iterations or goes to zero monotonically (superlinear convergence), then this should be reliable; otherwise, the contraction

estimate may have no bearing on the true contraction constant for the mapping $T(\cdot)$, and the error estimate may be poor.

Several dual criteria have been proposed in the literature. For example, the combination of (4) and (5) was suggested in [20], since (4) attempts to detect if convergence has been reached, whereas (5) attempts to ensure that (4) has not been satisfied simply due to stalling of the iteration. In [62], the combination of (4) and (1) is suggested, where (1) attempts to prevent halting on (4) due to stalling. The idea of scaling the components of u^{i+1} in (1) and $F(u^{i+1})$ in (2) is also recommended in [62], along with use of the maximum norm $\|\cdot\|_\infty$. In [78], other combinations are suggested [with an optimization orientation, some combinations involving the associated functional $J(\cdot)$].

EXERCISES

10.7.1 Let X and Y be Banach spaces and let $F \in C^2(X, Y)$. Use only the mean value theorem (Theorem 10.1.3) to derive the following Taylor-series expansion with integral remainder:

$$F(u+h) = F(u) + F'(u)h + \int_0^1 (1-t)F''(u+th)(h,h)\,dt.$$

[*Hint:* Expand $F'(u+h)$ using one of the formulas from Theorem 10.1.3, and then differentiate with respect to h using the chain rule.]

10.7.2 Find $J'(u)$ (a row vector function), $\nabla J(u)$ (a column vector function), and $\nabla^2 J(u)$ (the symmetric Hessian matrix of J) for the following functions of n variables.

(1) $J(u) = (1/2)u^T Au - u^T f$, where $A \in \mathbb{R}^{n\times n}$.
(2) $J(u) = (1/2)u^T Au - u^T f$, where $A \in \mathbb{R}^{n\times n}$, *and also* $A = A^T$.
(3) $J(u) = (1/2)u^T A^T Au - u^T Af$, where $A \in \mathbb{R}^{m\times n}$, and $f \in \mathbb{R}^m$.
(4) $J(u) = \|u\|_{l^2} = \left(\sum_{i=1}^n u_i^2\right)^{1/2}$.

[*Hint:* Do not use the information that you are working with the particular normed space $\mathbb{R}^n$; just think of $\mathbb{R}^n$ as an arbitrary Hilbert space H, and compute the derivatives using the convenient expression for the **G**-variation in (10.1.5).]

10.7.3 In [78], the sufficient descent condition (10.7.23) is derived by requiring the reduction in $\|F(u)\|_X$ be no worse than μ times the reduction in the *linear model* of F given by Taylor expansion $F(u^i + h) = F(u^i) + F'(u^i)h + \mathcal{O}(\|h\|_X^2)$. Setting $w = u^i + h$, we can write the expansion as a linear model $L^i(w)$ plus a remainder:

$$F(w) = L^i(w) + \mathcal{O}(\|h\|_X^2),$$

where $L^i(w) = F(u^i) + F'(u^i)(w - u^i)$. Prove that condition (10.7.23) is equivalent to

$$\frac{\|F(u^i)\|_X - \|F(u^i + \lambda_i v^i)\|_X}{\|L^i(u^i)\|_X - \|L^i(u^i + \lambda_i v^i)\|_X} \geqslant \mu.$$

10.7.4 Prove that Newton's method converges Q-linearly by using the Banach Fixed-Point Theorem. If you assume the existence of a solution, then you need to simply give conditions on F and F' which guarantee that the fixed point operator defined by the Newton iteration is a contraction on a sufficiently small ball around the solution. Can you construct a proof using the Banach Fixed-Point Theorem that also gives existence of the solution, without assuming it *a priori*? Can you recover something faster than Q-linear convergence?

10.7.5 *For Fun:* Construct a Newton iteration for computing the reciprocal of a positive real number without performing division. (This has been a standard algorithm for doing division in computer arithmetic units, together with a lookup table of good initial approximations.)

REFERENCES AND ADDITIONAL READING

1. R. Abraham, J. E. Marsden, and T. Ratiu, *Manifolds, Tensor Analysis, and Applications*, Springer-Verlag, New York, 1988.
2. R. A. Adams and J. F. Fornier, *Sobolev Spaces*, 2nd ed., Academic Press, San Diego, CA, 2003.
3. P. Allen, A. Clausen, and J. Isenberg, Near-constant mean curvature solutions of the Einstein constraint equations with non-negative Yamabe metrics, 2007, available as arXiv:0710.0725 [gr-qc].
4. E. L. Allgower, K. Böhmer, F. A. Potra, and W. C. Rheinboldt, A mesh-independence principle for operator equations and their discretizations, *SIAM J. Numer. Anal.*, 23(1):160–169, 1986.
5. D. Arnold, R. Falk, and R. Winther, Finite element exterior calculus: From hodge theory to numerical stability, *Bull. Amer. Math. Soc. (N.S.)*, 47:281–354, 2010.
6. D. Arnold, A. Mukherjee, and L. Pouly, Locally adapted tetrahedral meshes using bisection, *SIAM J. Sci. Statist. Comput.*, 22(2):431–448, 1997.
7. S. Ashby, M. Holst, T. Manteuffel, and P. Saylor, The role of the inner product in stopping criteria for conjugate gradient iterations, *BIT*, 41(1):26–53, 2001.
8. S. F. Ashby, T. A. Manteuffel, and P. E. Saylor, A taxonomy for conjugate gradient methods, *SIAM J. Numer. Anal.*, 27(6):1542–1568, 1990.
9. K. Atkinson and W. Han, *Theoretical Numerical Analysis*, Springer Verlag, New York, 2001.
10. J. Aubin, *Approximation of Elliptic Boundary-Value Problems*, Wiley, New York, 1972.
11. T. Aubin, *Nonlinear Analysis on Manifolds: Monge-Ampére Equations*, Springer-Verlag, New York, 1982.
12. O. Axelsson and V. Barker, *Finite Element Solution of Boundary Value Problems*, Academic Press, Orlando, FL, 1984.
13. A. K. Aziz, *The Mathematical Foundations of the Finite Element Method with Applications to Partial Differential Equations*, Academic Press, New York, 1972.

14. I. Babuška, The finite element method for elliptic equations with discontinuous coefficients, *Computing*, 5(3):207–213, 1970.

15. I. Babuška and W. Rheinboldt, Error estimates for adaptive finite element computations, *SIAM J. Numer. Anal.*, 15:736–754, 1978.

16. R. E. Bank, *PLTMG: A Software Package for Solving Elliptic Partial Differential Equations, Users' Guide 8.0*, Software, Environments and Tools, Vol. 5, SIAM, Philadelphia, 1998.

17. R. E. Bank and T. F. Dupont, Analysis of a two-level scheme for solving finite element equations, Tech. Rep. CNA–159, Center for Numerical Analysis, University of Texas at Austin, 1980.

18. R. E. Bank and T. F. Dupont, An optimal order process for solving finite element equations, *Math. Comp.*, 36(153):35–51, 1981.

19. R. E. Bank, T. F. Dupont, and H. Yserentant, The hierarchical basis multigrid method, *Numer. Math.*, 52:427–458, 1988.

20. R. E. Bank and D. J. Rose, Parameter selection for Newton-like methods applicable to nonlinear partial differential equations, *SIAM J. Numer. Anal.*, 17(6):806–822, 1980.

21. R. E. Bank and D. J. Rose, Global Approximate Newton Methods, *Numer. Math.*, 37:279–295, 1981.

22. R. E. Bank and D. J. Rose, Analysis of a multilevel iterative method for nonlinear finite element equations, *Math. Comp.*, 39(160):453–465, 1982.

23. R. E. Bank and D. J. Rose, Some error estimates for the box method, *SIAM J. Numer. Anal.*, 24(4):777–787, 1987.

24. R. E. Bank and R. K. Smith, A posteriori error estimates based on hierarchical bases, *SIAM J. Numer. Anal.*, 30(4):921–935, 1993.

25. R. E. Bank and A. Weiser, Some a posteriori error estimators for elliptic partial differential equations, *Math. Comp.*, 44(170):283–301, 1985.

26. E. Bänsch, Local mesh refinement in 2 and 3 dimensions, *Impact Comput. Sci. Engr.*, 3:181–191, 1991.

27. R. Bartnik and J. Isenberg, The constraint equations, in *The Einstein Equations and Large Scale Behavior of Gravitational Fields* (P. Chruściel and H. Friedrich, Eds.), pp. 1–38, Birkhäuser, Berlin, 2004.

28. P. Bastian, K. Birken, K. Johannsen, S. Lang, N. Neuss, H. Rentz-Reichert, and C. Wieners, UG - A flexible software toolbox for solving partial differential equations, in *Computing and Visualization in Science*, pp. 27–40, 1997.

29. R. Beck, B. Erdmann, and R. Roitzsch, KASKADE 3.0: An object-oriented adaptive finite element code, Tech. Rep. TR95–4, Konrad-Zuse-Zentrum for Informationstechnik, Berlin, 1995.

30. J. Bey, Tetrahedral grid refinement, *Computing*, 55(4):355–378, 1995.

31. J. Bey, Adaptive grid manager: AGM3D manual, Tech. Rep. 50, SFB 382, Mathematics Institute, University of Tubingen, Tubingen, Germany, 1996.

32. D. Braess, *Nonlinear Approximation Theory*, Springer-Verlag, Berlin, 1980.

33. D. Braess, *Finite Elements*, Cambridge University Press, Cambridge, MA, 1997.

34. D. Braess and W. Hackbusch, A new convergence proof for the multigrid method including the V-cycle, *SIAM J. Numer. Anal.*, 20(5):967–975, 1983.

35. J. Bramble and J. King, A finite element method for interface problems in domains with smooth boundaries and interfaces, *Adv. Comput. Math.*, 6(1):109–138, 1996.

36. J. H. Bramble and J. E. Pasciak, New convergence estimates for multigrid algorithms, *Math. Comp.*, 49(180):311–329, 1987.

37. J. H. Bramble and J. E. Pasciak, The analysis of smoothers for multigrid algorithms, *Math. Comp.*, 58(198):467–488, 1992.

38. J. H. Bramble, J. E. Pasciak, J. Wang, and J. Xu, Convergence estimates for multigrid algorithms without regularity assumptions, *Math. Comp.*, 57:23–45, 1991.

39. J. H. Bramble, J. E. Pasciak, J. Wang, and J. Xu, Convergence estimates for product iterative methods with applications to domain decomposition and multigrid, *Math. Comp.*, 57:1–21, 1991.

40. A. Brandt, Multi-level adaptive solutions to boundary-value problems, *Math. Comp.*, 31:333–390, 1977.

41. A. Brandt, Algebraic multigrid theory: The symmetric case, *Appl. Math. Comput.*, 19:23–56, 1986.

42. S. C. Brenner and L. R. Scott, *The Mathematical Theory of Finite Element Methods*, 2nd ed., Springer-Verlag, New York, 2002.

43. F. Brezzi, Mathematical theory of finite elements, in *State-of-the-Art Surveys on Finite Element Technology* (A. K. Noor and W. D. Pilkey, Eds.), pp. 1–25, The American Society of Mechanical Engineers, New York, NY, 1985.

44. F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods*, Springer-Verlag, New York, 1991.

45. J. M. Briggs and J. A. McCammon, Computation unravels mysteries of molecular biophysics, *Comput. Phys.*, 6(3):238–243, 1990.

46. Z. Cai, J. Mandel, and S. F. McCormick, The finite volume element method for diffusion equations on general triangulations, *SIAM J. Numer. Anal.*, 28:392–402, 1991.

47. C. Carstensen, Convergence of adaptive FEM for a class of degenerate convex minimization problem, *Preprint*, 2006.

48. L. Chen, A new class of high order finite volume methods for second order elliptic equations, *SIAM J. Numer. Anal.*, 47(6):4021–4043, 2010.

49. L. Chen, M. Holst, and J. Xu, The finite element approximation of the nonlinear Poisson-Boltzmann Equation, *SIAM J. Numer. Anal.*, 45(6):2298–2320, 2007, available as arXiv:1001.1350 [math.NA].

50. Y. Choquet-Bruhat, Sur l'intégration des équations de la relativité générale, *J. Rational Mech. Anal.*, 5:951–966, 1956.

51. Y. Choquet-Bruhat, Einstein constraints on compact n-dimensional manifolds, *Class. Quantum Grav.*, 21:S127–S151, 2004.

52. P. G. Ciarlet, *The Finite Element Method for Elliptic Problems*, North-Holland, New York, 1978.

53. P. Clément, Approximation by finite element functions using local regularization, *RAIRO Anal. Numer.*, 2:77–84, 1975.

54. A. Cohen, *Numerical Analysis of Wavelet Methods*, North-Holland, New York, 2003.
55. G. Cook and S. Teukolsky, Numerical relativity: Challenges for computational science, in *Acta Numerica*, Vol. 8 (A. Iserles, Ed.), pp. 1–44, Cambridge University Press, New York, 1999.
56. I. Daubechies, *Ten Lectures on Wavelets*, SIAM, Philadelphia, 1992.
57. P. J. Davis, *Interpolation and Approximation*, Dover, New York, 1963.
58. L. Debnath and P. Mikusiński, *Introduction to Hilbert Spaces with Applications*, Academic Press, San Diego, CA, 1990.
59. P. Debye and E. Hückel, Zur Theorie der Elektrolyte: I. Gefrierpunktserniedrigung und verwandte Erscheinungen, *Phys. Z.*, 24(9):185–206, 1923.
60. R. S. Dembo, S. C. Eisenstat, and T. Steihaug, Inexact Newton Methods, *SIAM J. Numer. Anal.*, 19(2):400–408, 1982.
61. J. E. Dennis, Jr. and J. J. Moré, Quasi-Newton methods, motivation and theory, *SIAM Rev.*, 19(1):46–89, 1977.
62. J. E. Dennis, Jr. and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, NJ, 1983.
63. P. Deuflhard, *Newton Methods for Nonlinear Problems*, Springer-Verlag, Berlin, 2004.
64. R. A. DeVore and G. G. Lorentz, *Constructive Approximation*, Springer-Verlag, New York, 1993.
65. P. Drabek and J. Milota, *Methods of Nonlinear Analysis: Applications to Differential Equations*, Birkhäuser, Berlin, 2000.
66. M. Dryja and O. B. Widlund, Towards a unified theory of domain decomposition algorithms for elliptic problems, in *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations* (T. F. Chan, R. Glowinski, J. Périaux, and O. B. Widlund, Eds.), pp. 3–21, SIAM, Philadelphia, 1989.
67. I. Ekeland and R. Temam, *Convex Analysis and Variational Problems*, North-Holland, New York, 1976.
68. D. Estep, A posteriori error bounds and global error control for approximations of ordinary differential equations, *SIAM J. Numer. Anal.*, 32:1–48, 1995.
69. D. Estep, M. Holst, and M. Larson, Generalized Green's functions and the effective domain of influence, *SIAM J. Sci. Comput.*, 26(4):1314–1339, 2005.
70. L. C. Evans, *Partial Differential Equations*, Vol. 19 of *Graduate Studies in Mathematics*, American Mathematical Society, Providence, RI, 2010.
71. R. Eymard, T. Gallouët, and R. Herbin, Convergence of finite volume schemes for semilinear convection diffusion equations, *Numer. Math.*, 82(1):91–116, 1999.
72. R. Eymard, T. Gallouët, and R. Herbin, Finite volume methods, in *Handbook of numerical analysis, Vol. VII*, pp. 713–1020, North-Holland, Amsterdam, 2000.
73. R. P. Fedorenko, A relaxation method for solving elliptic difference equations, *USSR Comput. Math. Math. Phys.*, 1(5):1092–1096, 1961.
74. R. P. Fedorenko, The speed of convergence of one iterative process, *USSR Comput. Math. Math. Phys.*, 4(3):227–235, 1964.
75. FETK, The Finite Element ToolKit, `http://www.FETK.org`.

76. R. Fletcher and C. Reeves, Function minimization by conjugate gradients, *Comput. J.*, 7:149–154, 1964.

77. S. Fucik and A. Kufner, *Nonlinear Differential Equations*, Elsevier Scientific, New York, 1980.

78. P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*, Academic Press, London and New York, 1981.

79. M. S. Gockenbach, *Partial Differential Equations: Analytical and Numerical Methods*, SIAM, Philadelphia, 2002.

80. M. Griebel, Multilevel algorithms considered as iterative methods on indefinite systems, in *Proceedings of the Second Copper Mountain Conference on Iterative Methods* (T. Manteuffel, Ed.), 1992.

81. M. Griebel and M. A. Schweitzer, A particle-partition of unity method for the solution of elliptic, parabolic, and hyperbolic PDEs, *SIAM J. Sci. Statist. Comput.*, 22(3):853–890, 2000.

82. P. Grisvard, *Elliptic Problems in Nonsmooth Domains*, Pitman, Marshfield, MA, 1985.

83. W. Hackbusch, On the fast solutions of nonlinear elliptic equations, *Numer. Math.*, 32:83–95, 1979.

84. W. Hackbusch, Multi-grid convergence theory, in *Multigrid Methods: Proceedings of Köln-Porz Conference on Multigrid Methods, Lecture notes in Mathematics 960* (W. Hackbusch and U. Trottenberg, Eds.), Springer-Verlag, Berlin, Germany, 1982.

85. W. Hackbusch, *Multi-grid Methods and Applications*, Springer-Verlag, Berlin, 1985.

86. W. Hackbusch, On first and second order box schemes, *Computing*, 41(4):277–296, 1989.

87. W. Hackbusch, *Iterative Solution of Large Sparse Systems of Equations*, Springer-Verlag, Berlin, 1994.

88. W. Hackbusch and A. Reusken, Analysis of a damped nonlinear multilevel method, *Numer. Math.*, 55:225–246, 1989.

89. P. R. Halmos, *Finite-Dimensional Vector Spaces*, Springer-Verlag, Berlin, 1958.

90. S. W. Hawking and G. F. R. Ellis, *The Large Scale Structure of Space-Time*, Cambridge University Press, Cambridge, UK, 1973.

91. E. Hebey, *Sobolev Spaces on Riemannian Manifolds*, Vol. 1635 of *Lecture Notes in Mathematics*, Springer-Verlag, Berlin, New York, 1996.

92. M. R. Hestenes and E. Stiefel, Methods of conjugate gradients for solving linear systems, *J. Res. of NBS*, 49:409–435, 1952.

93. M. Holst, An algebraic Schwarz theory, Tech. Rep. CRPC-94-12, Applied Mathematics and CRPC, California Institute of Technology, 1994.

94. M. Holst, Adaptive numerical treatment of elliptic systems on manifolds, *Adv. Comput. Math.*, 15(1–4):139–191, 2001, available as arXiv:1001.1367 [math.NA].

95. M. Holst, J. McCammon, Z. Yu, Y. Zhou, and Y. Zhu, Adaptive finite element modeling techniques for the Poisson-Boltzmann equation, to appear in Comm. Comput. Phys. Available as arXiv:1009.6034 [math.NA].

96. M. Holst, G. Nagy, and G. Tsogtgerel, Far-from-constant mean curvature solutions of Einstein's constraint equations with positive Yamabe metrics, *Phys. Rev. Lett.*, 100(16):161101.1–161101.4, 2008, available as arXiv:0802.1031 [gr-qc].

97. M. Holst, G. Nagy, and G. Tsogtgerel, Rough solutions of the Einstein constraints on closed manifolds without near-CMC conditions, *Comm. Math. Phys.*, 288(2):547–613, 2009, available as arXiv:0712.0798 [gr-qc].

98. M. Holst and E. Titi, Determining projections and functionals for weak solutions of the Navier-Stokes equations, in *Recent Developments in Optimization Theory and Nonlinear Analysis*, Vol. 204 of *Contemporary Mathematics* (Y. Censor and S. Reich, Eds.), American Mathematical Society, Providence, RI, 1997, available as arXiv:1001.1357 [math.AP].

99. M. Holst, G. Tsogtgerel, and Y. Zhu, Local convergence of adaptive methods for nonlinear partial differential equations, submitted for publication. Available as arXiv:1001.1382 [math.NA].

100. M. Holst and S. Vandewalle, Schwarz methods: To symmetrize or not to symmetrize, *SIAM J. Numer. Anal.*, 34(2):699–722, 1997, available as arXiv:1001.1362 [math.NA].

101. M. J. Holst, *The Poisson-Boltzmann Equation: Analysis and Multilevel Numerical Solution*, Ph.D. dissertation, University of Illinois at Urbana-Champaign, 1994.

102. G. Hsiao and W. Wendland, *Boundary Integral Equations*, Vol. 164 of *Applied Mathematical Sciences Series*, Springer-Verlag, Berlin, 2008.

103. T. J. R. Hughes, *The Finite Element Method*, Dover, New York, 2000.

104. J. Isenberg, Constant mean curvature solution of the Einstein constraint equations on closed manifold, *Class. Quantum Grav.*, 12:2249–2274, 1995.

105. J. Isenberg and V. Moncrief, A set of nonconstant mean curvature solution of the Einstein constraint equations on closed manifolds, *Class. Quantum Grav.*, 13:1819–1847, 1996.

106. J. W. Jerome and T. Kerkhoven, A finite element approximation theory for the drift diffusion semiconductor model, *SIAM J. Numer. Anal.*, 28(2):403–422, 1991.

107. C. Johnson, *Numerical Solution of Partial Differential Equations by the Finite Element Method*, Cambridge University Press, Cambridge, UK, 1987.

108. L. V. Kantorovich and G. P. Akilov, *Functional Analysis*, Pergamon Press, New York, 1982.

109. L. V. Kantorovich and V. I. Krylov, *Approximate Methods of Higher Analysis*, P. Noordhoff, Groningen, The Netherlands, 1958.

110. T. Kato, *Perturbation Theory for Linear Operators*, Springer-Verlag, Berlin, 1980.

111. H. B. Keller, *Numerical Methods in Bifurcation Problems*, Tata Institute of Fundamental Research, Bombay, India, 1987.

112. T. Kerkhoven, Piecewise linear Petrov-Galerkin analysis of the box-method, *SIAM J. Numer. Anal.*, 33(5):1864–1884, 1996.

113. S. Kesavan, *Topics in Functional Analysis and Applications*, Wiley, New York, 1989.

114. A. N. Kolmogorov and S. V. Fomin, *Introductory Real Analysis*, Dover, New York, 1970.

115. R. Kress, *Linear Integral Equations*, Springer-Verlag, Berlin, 1989.

116. E. Kreyszig, *Introductory Functional Analysis with Applications*, Wiley, New York, 1990.

117. A. Kufner, O. John, and S. Fucik, *Function Spaces*, Noordhoff International, Leyden, The Netherlands, 1977.

118. J. Lee and T. Parker, The Yamabe problem, *Bull. Amer. Math. Soc.*, 17(1):37–91, 1987.

119. J. M. Lee, *Riemannian Manifolds*, Springer-Verlag, New York, 1997.

120. L. Lehner, Numerical relativity: A review, *Class. Quantum Grav.*, 18:R25–R86, 2001.

121. R. J. LeVeque, *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems*, SIAM, Philadelphia, 2007.

122. Y. Y. Li and L. Zhang, A Harnack type inequality for the Yamabe equation in low dimensions, *Calc. Var.*, 20:133–151, 2004.

123. A. Liu and B. Joe, Quality local refinement of tetrahedral meshes based on bisection, *SIAM J. Sci. Statist. Comput.*, 16(6):1269–1291, 1995.

124. J. Liu and W. Rheinboldt, A posteriori finite element error estimators for indefinite elliptic boundary value problems, *Numer. Funct. Anal. Optim.*, 15(3):335–356, 1994.

125. J.-F. Maitre and F. Musy, Multigrid methods: Convergence theory in a variational framework, *SIAM J. Numer. Anal.*, 21(4):657–671, 1984.

126. J. E. Marsden and T. J. R. Hughes, *Mathematical Foundations of Elasticity*, Dover, New York, 1994.

127. R. Mattheij, W. Rienstra, and J. ten Thije Boonkkamp, *Partial Differential Equations: Modeling, Analysis, Computation*, SIAM, Philadelphia, 2005.

128. J. Maubach, Local bisection refinement for N-simplicial grids generated by relection, *SIAM J. Sci. Statist. Comput.*, 16(1):210–277, 1995.

129. D. Maxwell, Rough solutions of the Einstein constraint equations on compact manifolds, *J. Hyp. Differential Equations*, 2(2):521–546, 2005.

130. D. Maxwell, A class of solutions of the vacuum Einstein constraint equations with freely specified mean curvature, 2008, available as arXiv:0804.0874 [gr-qc].

131. S. F. McCormick, Multigrid methods for variational problems: Further results, *SIAM J. Numer. Anal.*, 21(2):255–263, 1984.

132. D. Mitrović and D. Zubrinić, *Fundamentals of Applied Functional Analysis*, Pitman Monographs and Surveys in Pure and Applied Mathematics, Longman Scientific & Tecnical, Wiley, New York, 1998.

133. A. Mukherjee, *An Adaptive Finite Element Code for Elliptic Boundary Value Problems in Three Dimensions with Applications in Numerical Relativity*, Ph.D. dissertation, Department of Mathematics, The Pennsylvania State University, 1996.

134. R. Nochetto, K. Siebert, and A. Veeser, Theory of adaptive finite element methods: An introduction, in *Multiscale, Nonlinear and Adaptive Approximation* (R. DeVore and A. Kunoth, Eds.), pp. 409–542, Springer, 2009, dedicated to Wolfgang Dahmen on the Occasion of His 60th Birthday.

135. J. T. Oden and L. F. Demkowicz, *Applied Functional Analysis*, CRC Series in Computational Mechanics and Applied Analysis, CRC Press, Boca Raton, FL, 1996.

136. J. T. Oden and J. N. Reddy, *An Introduction to The Mathematical Theory of Finite Elements*, Wiley, New York, 1976.

137. N. O'Murchadha and J. York, Jr., Existence and uniqueness of solutions of the Hamiltonian constraint of general relativity on compact manifolds, *J. Math. Phys.*, 14(11):1551–1557, 1973.

138. E. G. Ong, Uniform refinement of a tetrahedron, Tech. Rep. CAM 91-01, Department of Mathematics, UCLA, 1991.

139. J. M. Ortega, *Numerical Analysis: A Second Course*, Academic Press, New York, 1972.

140. J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

141. P. Oswald, *Multilevel Finite Element Approximation*, B. G. Teubner, Stuttgart, Germany, 1994.

142. R. Palais, *Foundations of Global Non-linear Analysis*, W. A. Benjamin, New York, 1968.

143. J. Pousin and J. Rappaz, Consistency, stability, a priori and a posteriori errors for Petrov-Galerkin methods applied to nonlinear problems, *Numer. Math.*, 69(2):213–231, 1994.

144. J. Rappaz, Numerical approximation of PDEs and Clément's interpolation, *Partial Differential Equations Func. Anal.*, 168:237–250, 2006.

145. M. Renardy and R. C. Rogers, *An Introduction to Partial Differential Equations*, Springer-Verlag, New York, 1993.

146. A. Reusken, Convergence of the multilevel full approximation scheme including the V-cycle, *Numer. Math.*, 53:663–686, 1988.

147. M. Rivara, Algorithms for refining triangular grids suitable for adaptive and multigrid techniques, *Internat. J. Numer. Methods Engrg.*, 20:745–756, 1984.

148. I. Rosenberg and F. Stenger, A lower bound on the angles of triangles constructed by bisecting the longest side, *Math. Comp.*, 29:390–395, 1975.

149. S. Rosenberg, *The Laplacian on a Riemannian Manifold*, Cambridge University Press, Cambridge, UK, 1997.

150. W. Rudin, *Real and Complex Analysis*, McGraw-Hill, New York, 1987.

151. J. W. Ruge and K. Stüben, Algebraic multigrid (AMG), in *Multigrid Methods*, Vol. 3 of *Frontiers in Applied Mathematics* (S. F. McCormick, Ed.), pp. 73–130, SIAM, Philadelphia, 1987.

152. G. Savare, Regularity results for elliptic equations in Lipschitz domains, *J. of Func. Anal.*, 152(1):176–201, 1998.

153. A. H. Schatz, An oberservation concerning Ritz-Galerkin methods with indefinite bilinear forms, *Math. Comp.*, 28(128):959–962, 1974.

154. A. H. Schatz and J. Wang, Some new error estimates for Ritz-Galerkin methods with minimal regularity assumptions, *Math. Comp.*, 62:445–475, 2000.

155. G. Schwarz, *Hodge Decomposition: A Method for Solving Boundary Value Problems*, Springer-Verlag, New York, 1991.

156. L. R. Scott and S. Zhang, Finite element interpolation of nonsmooth functions satisfying boundary conditions, *Math. Comp.*, 54(190):483–493, 1990.

157. L. A. Segel, *Mathematics Applied to Continuum Mechanics*, Dover, New York, 1977.

158. E. Seidel, New developments in numerical relativity, *Helv. Phys. Acta*, 69:454–471, 1996.

159. R. E. Showalter, *Hilbert Space Methods for Partial Differential Equations*, Pitman, Marshfield, MA, 1979.

160. E. M. Stein, *Singular Integrals and Differentiability Properties of Functions*, Princeton Mathematical Series, No. 30, Princeton University Press, Princeton, N.J., 1970.

161. G. Strang and G. Fix, *An Analysis of the Finite Element Method*, Prentice-Hall, Englewood Cliffs, NJ, 1973.

162. K. Stüben and U. Trottenberg, Multigrid methods: Fundamental algorithms, model problem analysis and applications, in *Multigrid Methods: Proceedings of Köln-Porz Conference on Multigrid Methods, Lecture notes in Mathematics 960* (W. Hackbusch and U. Trottenberg, Eds.), Springer-Verlag, Berlin, Germany, 1982.

163. M. Stynes, On faster convergence of the bisection method for all triangles, *Math. Comp.*, 35:1195–1201, 1980.

164. B. Szabó and I. Babuška, *Finite Element Analysis*, Wiley, New York, 1991.

165. C. Tanford, *Physical Chemistry of Macromolecules*, John Wiley & Sons, New York, NY, 1961.

166. M. E. Taylor, *Partial Differential Equations*, Vol. III, Springer-Verlag, New York, 1996.

167. V. Thomee, *Galerkin Finite Element Methods for Parabolic Problems*, Springer-Verlag, New York, 1997.

168. H. Triebel, *Interpolation Theory, Function Spaces, and Differential Operators*, North-Holland, Amsterdam, 1978.

169. R. S. Varga, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1962.

170. A. Veeser, Convergent adaptive finite elements for the nonlinear Laplacian, *Numer. Math.*, 92:743–770, 2002.

171. R. Verfürth, A posteriori error estimates for nonlinear problems: Finite element discretizations of elliptic equations, *Math. Comp.*, 62(206):445–475, 1994.

172. R. Verfürth, *A Review of* A Posteriori *Error Estimation and Adaptive Mesh-Refinement Techniques*, Wiley, New York, 1996.

173. E. L. Wachspress, *Iterative Solution of Elliptic Systems and Applications to the Neutron Diffusion Equations of Reactor Physics*, Prentice-Hall, Englewood Cliffs, NJ, 1966.

174. R. M. Wald, *General Relativity*, University of Chicago Press, Chicago, IL, 1984.

175. J. Wang, Convergence analysis without regularity assumptions for multigrid algorithms based on SOR smoothing, *SIAM J. Numer. Anal.*, 29(4):987–1001, 1992.

176. J. Wloka, *Partial Differential Equations*, Cambridge University Press, Cambridge, MA, 1992.

177. J. Xu, *Theory of Multilevel Methods*, Ph.D. dissertation, Department of Mathematics, Penn State University, 1989, technical Report AM 48.

178. J. Xu, Iterative methods by space decomposition and subspace correction, *SIAM Rev.*, 34(4):581–613, 1992.

179. J. Xu and A. Zhou, Local and parallel finite element algorithms based on two-grid discretizations, *Math. Comp.*, 69:881–909, 2000.

180. J. Xu and L. Zikatanov, Some observations on Babuška and Brezzi theories, *Numer. Math.*, 94:195–202, 2003.

181. J. Xu and Q. Zou, Analysis of linear and quadratic simplicial finite volume methods for elliptic equations, *Numer. Math.*, 111(3):469–492, 2009.

182. J. W. York, Jr., Kinematics and dynamics of general relativity, in *Sources of Gravitational Radiation* (L. L. Smarr, Ed.), pp. 83–126, Cambridge University Press, Cambridge, MA, 1979.

183. K. Yosida, *Functional Analysis*, Springer-Verlag, Berlin, 1980.

184. D. M. Young, *Iterative Solution of Large Linear Systems*, Academic Press, New York, 1971.

185. H. Yserentant, On the multi-level splitting of finite element spaces, *Numer. Math.*, 49:379–412, 1986.

186. Z. Yu, M. Holst, Y. Cheng, and J. McCammon, Feature-preserving adaptive mesh generation for molecular shape modeling and simulation, *J. of Mol. Graph. Model.*, 26:1370–1380, 2008.

187. Z. Yu, M. Holst, and J. McCammon, High-fidelity geometric modeling for biomedical applications, *Finite Elem. Anal. Des.*, 44(11):715–723, 2008.

188. E. Zeidler, *Nonlinear Functional Analysis and Its Applications*, Vol. I: Fixed Point Theorems, Springer-Verlag, New York, 1991.

189. E. Zeidler, *Nonlinear Functional Analysis and Its Applications*, Vol. III: Variational Methods and Optimization, Springer-Verlag, New York, 1991.

190. S. Zhang, *Multi-level Iterative Techniques*, Ph.D. dissertation, Department of Mathematics, Pennsylvania State University, 1988.

191. Y. Zhou, M. Holst, and J. McCammon, Nonlinear elastic modeling of macromolecular conformational change induced by electrostatic forces, *J. Math. Anal. Appl.*, 340(1):135–164, 2008, available as arXiv:1001.1371 [math.AP].

192. Z. Zhou, P. Payne, M. Vasquez, N. Kuhn, and M. Levitt, Finite-difference solution of the Poisson-Boltzmann equation: Complete elimination of self-energy, *J. Comput. Chem.*, 11(11):1344–1351, 1996.

193. O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method*, 5th ed., Vol. 1: The Basis, Butterworth-Heinemann, 2000.

194. O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method*, 5th ed., Vol. 2: Solid Mechanics, Butterworth-Heinemann, 2000.

195. O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method*, 5th ed., Vol. 3: Fluid Dynamics, Butterworth-Heinemann, 2000.

INDEX