Iterative Methods for Solving Linear Systems
(from book to be published by SIAM)

Anne Greenbaum

February 23, 1997

Chapter 1

Introduction

The subject of this book is what at first appears to be a very simple problem – how to solve a system of linear equations $Ax = b$, where A is an n by n nonsingular matrix and b is a given n-vector. One well-known method is Gaussian elimination. In general, this requires storage of all n^2 entries of the matrix A and it requires approximately $2n^3/3$ arithmetic operations (additions, subtractions, multiplications, and divisions). Matrices that arise in practice, however, often have special properties that can be only partially exploited by Gaussian elimination. For example, the matrices that arise from differencing partial differential equations are often *sparse*, having only a few nonzeros per row. If the (i, j)-entry of matrix A is zero whenever $|i - j| > m$, then a *banded* version of Gaussian elimination can solve the linear system by storing only the approximately $2mn$ entries inside the band (i.e., those with $|i - j| \leq m$) and performing about $2m^2n$ operations. The algorithm cannot take advantage of any zeros inside the band, however, as these *fill-in* with nonzeros during the process of elimination.

In contrast, sparseness and other special properties of matrices can often be used to great advantage in matrix-vector multiplication. If a matrix has just a few nonzeros per row, then the number of operations required to compute the product of that matrix with a given vector is just a few n, instead of the $2n^2$ operations required for a general dense matrix-vector multiplication. The storage required is only that for the nonzeros of the matrix, and, if these are sufficiently simple to compute, even this can be avoided. For certain special dense matrices, a matrix-vector product can also be computed with just $O(n)$ operations. For example, a Cauchy matrix is one whose (i, j)-entry is $1/(z_i - z_j)$, for $i \neq j$, where $z_1, \ldots, z_n$ are some complex numbers. The product of this matrix with a given vector can be computed in time $O(n)$ using the fast multipole method [68] and the actual matrix entries need never be computed or stored. This leads one to ask whether the system of linear equations $Ax = b$ can be solved (or an acceptably good approximate solution obtained), using matrix-vector multiplications. If this can be accomplished with a moderate number of matrix-vector multiplications and little additional work, then the iterative procedure that does this may far outperform Gaussian elimination in terms of both work and storage.

One might have an initial guess for the solution, but if this is not the case, then the only vector associated with the problem is the right-hand side vector b. Without computing any matrix-vector products, it seems natural to take some multiple of b as the first approximation to the solution:

$$x_1 \in \text{span}\{b\}.$$

One then computes the product Ab, and takes the next approximation to be some linear combination of b and Ab:

$$x_2 \in \text{span}\{b, Ab\}.$$

This process continues so that the approximation at step k satisfies

$$x_k \in \text{span}\{b, Ab, \ldots, A^{k-1}b\}, \quad k = 1, 2, \ldots. \tag{1.1}$$

The space represented on the right in (1.1) is called a *Krylov subspace* for the matrix A and initial vector b.

Given that x_k is to be taken from the Krylov space in (1.1), one must ask the following two questions:

(i) How good an approximate solution is contained in the space (1.1)?

(ii) How can a good (optimal) approximation from this space be computed with a moderate amount of work and storage?

These questions are the subject of Part I of this book.

If it turns out that the space (1.1) does not contain a good approximate solution for any moderate size k, or if such an approximate solution cannot be easily computed, then one might consider modifying the original problem to obtain a better Krylov subspace. For example, one might use a *preconditioner* M and effectively solve the modified problem

$$M^{-1}Ax = M^{-1}b$$

by generating approximate solutions $x_1, x_2, , \ldots$ satisfying

$$x_k \in \operatorname{span}\{M^{-1}b, (M^{-1}A)M^{-1}b, \ldots, (M^{-1}A)^{k-1}M^{-1}b\}. \tag{1.2}$$

At each step of the preconditioned algorithm, it is necessary to compute the product of M^{-1} with a vector, or, equivalently, to solve a linear system with coefficient matrix M, so M should be chosen so that such linear systems are much easier to solve than the original problem. The subject of finding good preconditioners is a very broad one on which much research has focused in recent years, most of it designed for specific classes of problems; e.g., linear systems arising from finite element or finite difference approximations for elliptic partial differential equations. Part II of this book deals with the topic of preconditioners.

1.1 Brief Overview of the State of the Art

In dealing with questions (i) and (ii), one must consider two types of matrices — *Hermitian* and *non-Hermitian*. These questions are essentially solved for the Hermitian case, but remain wide-open in the case of non-Hermitian matrices.

A caveat here, is that we are now referring to the *preconditioned* matrix. If A is a Hermitian matrix and the preconditioner M is Hermitian and *positive definite*, then instead of using left preconditioning as described above one can (implicitly) solve the modified linear system

$$L^{-1}AL^{-H}y = L^{-1}b, \quad x = L^{-H}y \tag{1.3}$$

where $M = LL^H$, and the superscript ^{H} denotes the complex conjugate transpose ($L^H_{ij} = \bar{L}_{ji}$). Of course, as before, one does not actually form the matrix $L^{-1}AL^{-H}$, but approximations to the solution y are considered to come from the Krylov space based on $L^{-1}AL^{-H}$ and $L^{-1}b$, so that the computed approximations to x come from the space in (1.2). If the preconditioner M is *indefinite*, then the preconditioned problem cannot be cast in the form (1.3) and treated as a Hermitian problem. In this case, methods for non-Hermitian problems may be needed. The subject of special methods for Hermitian problems with Hermitian indefinite preconditioners is an area of current research.

Throughout the remainder of this section, we will let A and b denote the already preconditioned matrix and right-hand side. The matrix A is Hermitian if the original problem is Hermitian and the preconditioner is Hermitian and positive definite, and otherwise it is non-Hermitian.

1.1.1 Hermitian Matrices

For *real symmetric* or *complex Hermitian* matrices A, there is a known short recurrence for finding the 'optimal' approximation of the form (1.1), if 'optimal' is taken to mean the approximation whose residual, $b - Ax_k$, has the smallest Euclidean norm. An algorithm that generates this optimal approximation is called MINRES (minimal residual) [104]. If A is also positive definite, then one might instead minimize the A-norm of the error, $\|e_k\|_A \equiv \langle A^{-1}b - x_k, b - Ax_k\rangle^{1/2}$. The conjugate gradient (CG) algorithm [73] generates this approximation. For each of these algorithms, the work required at each iteration is the computation of one matrix-vector product (which is always required to generate the next vector in the Krylov space) plus a few vector inner products, and only a few vectors need be stored. Since these methods find the 'optimal' approximation with little extra work and storage, beyond that required for the matrix-vector multiplication, they are almost always the methods of choice for Hermitian problems. [Of course, one can never really make such a blanket statement about numerical methods. On some parallel machines, for instance, inner products are very expensive, and methods that avoid inner products, even if they generate non-optimal approximations, may be preferred.]

Additionally, we can describe precisely how good these optimal approximations are (for the worst possible right-hand side vector b), in terms of the *eigenvalues* of the matrix. Consider the 2-norm of the residual in the MINRES algorithm. It follows from (1.1) that the residual $r_k \equiv b - Ax_k$ can be written in the form

$$r_k = P_k(A)b,$$

where P_k is a certain kth degree polynomial with value 1 at the origin, and, for any other such polynomial p_k, we have

$$\|r_k\| \leq \|p_k(A)b\|. \tag{1.4}$$

Writing the eigendecomposition of A as $A = Q\Lambda Q^H$, where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ is a diagonal matrix of eigenvalues and Q is a unitary matrix whose columns are eigenvectors, expression (1.4) implies

$$\|r_k\| \leq \|Qp_k(\Lambda)Q^H b\| \leq \|p_k(\Lambda)\| \; \|b\|,$$

and since this holds for any kth degree polynomial p_k with $p_k(0) = 1$, we have

$$\|r_k\|/\|b\| \leq \min_{p_k} \max_{i=1,\ldots,n} |p_k(\lambda_i)| \tag{1.5}$$

It turns out that the bound (1.5) on the size of the MINRES residual at step k is *sharp* — that is, for each k there is a vector b for which this bound will be attained [59, 63, 79]. Thus the question of the size of the MINRES residual at step k is reduced to a problem in approximation theory — how well can one approximate zero on the set of eigenvalues of A using a kth degree polynomial with value 1 at the origin? One can answer this question precisely with a complicated expression involving all of the eigenvalues of A or one can give simple bounds in terms of just a few of the eigenvalues of A. The important point is that the norm of the residual (for the worst-case right-hand side vector) is completely determined by the eigenvalues of the matrix, and we have at least intuitive ideas of what constitute good and bad eigenvalue distributions. The same reasoning shows that the A-norm of the error $e_k \equiv A^{-1}b - x_k$ in the CG algorithm satisfies

$$\|e_k\|_A / \|A^{-1}b\|_A \leq \min_{p_k} \max_{i=1,\ldots,n} |p_k(\lambda_i)|, \tag{1.6}$$

and, again, this bound is sharp.

Thus, for Hermitian problems, we not only have good algorithms for finding the optimal approximation from the Krylov space (1.1) but we can also say just how good that approximation will be, based on simple properties of the coefficient matrix (i.e., the eigenvalues). It is therefore fair to say that the iterative solution of Hermitian linear systems is well-understood — except for one thing. All of the above discussion assumes *exact arithmetic*. It is well-known that in finite precision arithmetic the CG and MINRES algorithms do *not* find the optimal approximation from the Krylov space (1.1) or necessarily anything close to it! In fact, the CG algorithm originally lost favor partly because it did not behave the way exact arithmetic theory predicted [41]. More recent work [61, 66, 35, 34] has gone a long way toward explaining the behavior of the MINRES and CG algorithms in finite precision arithmetic, although open problems remain. This work is discussed in Chapter 4.

1.1.2 Non-Hermitian Matrices

While the methods of choice and convergence analysis for Hermitian problems are well-understood in exact arithmetic, and the results of Chapter 4 go a long way towards explaining the behavior of these methods in finite precision arithmetic, the state-of-the-art for *non-Hermitian* problems is not nearly so well-developed. One difficulty is that no method is known for finding the optimal approximation from the space (1.1), while performing only $O(n)$ operations per iteration, in addition to the matrix-vector multiplication. In fact, a theorem due to Faber and Manteuffel [43] shows that for most non-Hermitian matrices A there is no short recurrence for the optimal approximation from the Krylov space (1.1). To fully understand this result, one must consider the statement and hypotheses carefully, and we give more of these details in Chapter 6. Still, the current options for non-Hermitian problems are either to perform extra work ($O(nk)$ operations at step k) and use extra storage ($O(nk)$ words to perform k iterations) to find the optimal approximation from the Krylov space or to settle for a nonoptimal approximation. The (full) GMRES (generalized minimal residual) algorithm [110] (and other mathematically equivalent algorithms) find the approximation of the form (1.1) for which the 2-norm of the residual is minimal, at the cost of this extra work and storage, while other non-Hermitian iterative methods (e.g., BCG [48], CGS [114], QMR [51], BiCGSTAB [124], restarted GMRES [110], hybrid GMRES [96], etc.) generate nonoptimal approximations.

Even if one does generate the optimal approximation of the form (1.1), we are still unable to answer question (i). That is, there is no known way to describe how good this optimal approximation will be (for the worst-case right-hand side vector), in terms of simple properties of the coefficient matrix. One might try an approach based on eigenvalues, as was done for the Hermitian case. Assume that A is diagonalizable and write an eigendecomposition of A as $A = V \Lambda V^{-1}$, where Λ is a diagonal matrix of eigenvalues and the columns of V are eigenvectors. Then it follows from (1.4) that the GMRES residual at step k satisfies

$$\|r_k\| \leq \min_{p_k} \|V p_k(\Lambda) V^{-1} b\| \leq \kappa(V) \ \min_{p_k} \max_{i=1,\ldots,n} |p_k(\lambda_i)| \ \|b\|, \qquad (1.7)$$

where $\kappa(V) = \|V\| \cdot \|V^{-1}\|$ is the condition number of the eigenvector matrix. The scaling of the columns of V can be chosen to minimize this condition number. When A is a *normal* matrix, so that V can be taken to have condition number one, it turns out that the bound (1.7) is sharp, just as for the Hermitian case. Thus for normal matrices, the analysis of GMRES again reduces to a question of polynomial approximation — how well can one approximate zero on the set of (complex) eigenvalues using a kth degree polynomial with value 1 at the origin. When A is nonnormal, however, the bound (1.7) may *not* be sharp. If the condition number of V is huge, the right-hand side of (1.7) may also be large, but this does not necessarily imply that GMRES converges poorly. It may simply mean that the bound (1.7) is a large overestimate of the actual GMRES residual norm. An interesting open question is to determine when an ill-conditioned eigenvector matrix implies poor convergence for GMRES and when it simply means that the bound (1.7) is a large overestimate. If eigenvalues are not the key, then one would like to be able to describe the behavior of GMRES in terms of some other characteristic properties of the matrix A. Some ideas along these lines are discussed in Section 3.2.

Finally, since the full GMRES algorithm may be impractical if a fairly large number of iterations are required, one would like to have theorems relating the convergence of some nonoptimal methods (that do not require extra work and storage) to that of GMRES. Unfortunately, no fully satisfactory theorems of this kind are currently available, and this important open problem is discussed in Chapter 6.

1.1.3 Preconditioners

The tools used in the derivation of preconditioners are much more diverse than those applied to the study of iteration methods. There are some general results concerning comparison of preconditioners and optimality of preconditioners of certain forms (e.g., block-diagonal), and these are described in Chapter 10. Many of the most successful preconditioners, however, have been derived for special problem classes, where the origin of the problem suggests a particular type of preconditioner. Multigrid and domain decomposition methods are examples of this type of preconditioner and are discussed in Chapter 12. Still other preconditioners are designed for very specific physical problems, such as the transport equation. Since one cannot assume familiarity with every scientific application, a complete survey is impossible. Chapter 9 contains two example problems, but the problem of generalizing application-specific preconditioners to a broader setting remains an area of active research.

Chapter 2

Some Iteration Methods

2.1 Simple Iteration

Given a preconditioner M for the linear system $Ax = b$, a natural idea for generating approximate solutions is the following. Since a preconditioner is designed so that $M^{-1}A$ in some sense approximates the identity, $M^{-1}(b - Ax_k)$ can be expected to approximate the error $A^{-1}b - x_k$ in an approximate solution x_k. A better approximate solution x_{k+1} might therefore be obtained by taking

$$x_{k+1} = x_k + M^{-1}(b - Ax_k). \tag{2.1}$$

This procedure of starting with an initial guess x_0 for the solution and generating successive approximations using (2.1) for $k = 0, 1, \ldots$ is sometimes called *simple iteration*, but more often it goes by different names according to the choice of M. For M equal to the diagonal of A, it is called *Jacobi* iteration; for M equal to the lower triangle of A, it is the *Gauss-Seidel* method; and for M of the form $\omega^{-1}D - L$, where D is the diagonal of A, L is the strict lower triangle of A, and ω is a relaxation parameter, it is the successive overrelaxation or *SOR* method. Preconditioners will be discussed in Part II of this book, but our concern in this section is to describe the behavior of iteration (2.1) for a given preconditioner M, in terms of properties of the preconditioned matrix $M^{-1}A$.

We will see in later sections that the simple iteration procedure

(2.1) can be improved upon in a number of ways. Still, it is not to be abandoned. All of these improvements require some extra work, and if the iteration matrix $M^{-1}A$ is sufficiently close to the identity, this extra work may not be necessary. Multigrid methods, which will be discussed in Chapter 12, can be thought of as very sophisticated preconditioners used with the simple iteration (2.1).

An actual implementation of (2.1) might use the following algorithm:

Algorithm 1. Simple Iteration.

Given an initial guess x_0, compute $r_0 = b - Ax_0$, and solve $Mz_0 = r_0$.
For $k = 1, 2, \ldots,$

Set $x_k = x_{k-1} + z_{k-1}$.

Compute $r_k = b - Ax_k$.

Solve $Mz_k = r_k$.

Let $e_k \equiv A^{-1}b - x_k$ denote the error in approximation x_k. It follows from (2.1) that

$$e_k = (I - M^{-1}A)e_{k-1} = \ldots = (I - M^{-1}A)^k e_0. \tag{2.2}$$

Taking norms on both sides in (2.2), we find

$$|||e_k||| \leq |||(I - M^{-1}A)^k||| \cdot |||e_0|||, \tag{2.3}$$

where $|||\cdot|||$ can be any vector norm, provided the matrix norm is taken to be the one *induced* by the vector norm, $|||B||| \equiv \max_{|||y|||=1} |||By|||$. In this case, the bound in (2.3) is *sharp*, since there is an initial error e_0 for which equality holds.

The norm of the error will approach zero and x_k will approach $A^{-1}b$, for *every* initial error e_0 if and only if

$$\lim_{k \to \infty} |||(I - M^{-1}A)^k||| = 0.$$

It was shown in Section 1.3.1 that, independent of the matrix norm used in (2.3), the quantity $|||(I - M^{-1}A)^k|||^{1/k}$ approaches the *spectral radius*, $\rho(I - M^{-1}A)$ as $k \to \infty$. Thus we have the result that

Theorem 2.1.1. The iteration (2.1) converges to $A^{-1}b$ for every initial error e_0 if and only if $\rho(I - M^{-1}A) < 1$.

Having established necessary and sufficient conditions for convergence, we must now consider the *rate* of convergence. How many iterations will be required to obtain an approximation that is within, say, δ of the true solution? This question is not so easy to answer in general.

Taking norms on each side in (2.2), we can write

$$|||e_k||| \leq |||I - M^{-1}A||| \cdot |||e_{k-1}|||, \tag{2.4}$$

from which it follows that if $|||I - M^{-1}A||| < 1$, then the error is reduced by at least this factor at each iteration. The error will satisfy $|||e_k\|/|||e_0||| \leq \delta$, provided

$$k \geq \log \delta / \log |||I - M^{-1}A|||.$$

It was shown in Section 1.3.1 that for any $\epsilon > 0$, there is a matrix norm such that $|||I - M^{-1}A||| < \rho(I - M^{-1}A) + \epsilon$. Hence if $\rho(I - M^{-1}A) < 1$, then there is a norm for which the error is reduced *monotonically* and convergence is at least *linear* with a reduction factor approximately equal to $\rho(I - M^{-1}A)$. Unfortunately, however, this norm is sometimes a very strange one (as might be deduced from the proof of Theorem 1.3.3, since the matrix D_t involved an exponential scaling), and it is unlikely that one would really want to measure convergence in terms of this norm!

It is usually the 2-norm or the ∞-norm or some closely related norm of the error that is of interest. For the class of *normal* matrices (diagonalizable matrices with a complete set of orthonormal eigenvectors), the 2-norm and the spectral radius coincide. Thus if $I - M^{-1}A$ is a normal matrix, then the 2-norm of the error is reduced by at least the factor $\rho(I - M^{-1}A)$ at each step. For nonnormal matrices, however, it is often the case that $\rho(I - M^{-1}A) < 1 < \|I - M^{-1}A\|$. In this case, the error may grow over some finite number of steps, and it is impossible

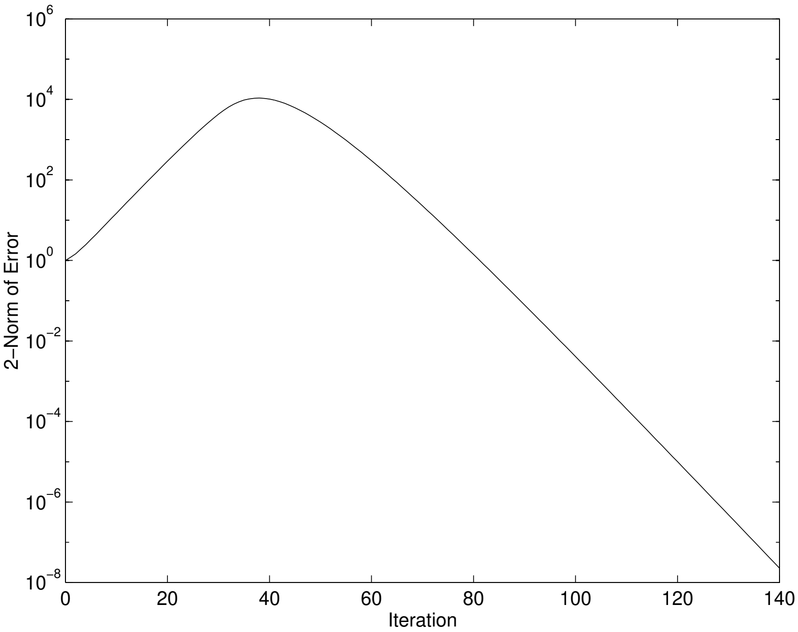

Figure 2.1: Simple iteration for a highly nonnormal matrix $\rho = .74$.

to predict the number of iterations required to obtain a given level of accuracy, knowing only the spectral radius.

An example is shown in Figure 2.1. The matrix A was taken to be

$$A = \begin{pmatrix} 1 & -1.16 & & \\ .16 & \ddots & \ddots & \\ & \ddots & \ddots & -1.16 \\ & & .16 & 1 \end{pmatrix}$$

and M was taken to be the lower triangle of A. For problem size $n = 30$, the spectral radius of $I - M^{-1}A$ is about .74, while the 2-norm of $I - M^{-1}A$ is about 1.4. As the figure shows, the 2-norm of the error increases by about four orders of magnitude over its original value, before starting to decrease.

While, in general, the spectral radius does not determine the convergence rate of early iterations, it does describe the *asymptotic* convergence rate of (2.1). We will prove this only in the case where $M^{-1}A$ is diagonalizable and has a single eigenvalue of largest absolute value.

Then, writing the eigendecomposition of $M^{-1}A$ as $V\Lambda V^{-1}$, where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ and, say, $\rho(I - M^{-1}A) = |1-\lambda_1| > \max_{i>1} |1-\lambda_i|$, we have

$$V^{-1}e_k = (I - \Lambda)^k (V^{-1}e_0).$$

Assuming that the first component of $V^{-1}e_0$ is nonzero, for k sufficiently large, the largest component of $V^{-1}e_k$ will be the first, $(V^{-1}e_k)_1$. At each subsequent iteration this dominant component is multiplied by the factor $(1-\lambda_1)$ and so we have

$$\begin{aligned} |||e_{k+j}||| &\approx |(V^{-1}e_{k+j})_1| = \rho(I - M^{-1}A)\ |(V^{-1}e_{k+j-1})_1| \\ &\approx \rho(I - M^{-1}A)\ |||e_{k+j-1}|||, \quad j = 1, 2, \ldots. \end{aligned}$$

A more general proof can be carried out using the Jordan form of $I - M^{-1}A$, as described in Section 1.3.2. Note in Figure 2.1 that eventually the error decreases by about the factor $\rho(I - M^{-1}A) = .74$ at each step.

2.2 Orthomin(1) and Steepest Descent

In this section we discuss methods to improve on the simple iteration (2.1) by introducing dynamically computed parameters into the iteration. For ease of notation, here and throughout the remainder of Part I of this book, we avoid explicit reference to the preconditioner M and consider A and b to be the coefficient matrix and right-hand side vector for the already preconditioned system. Sometimes we will assume that the preconditioned matrix is Hermitian. If the original coefficient matrix is Hermitian and the preconditioner M is Hermitian and *positive definite*, then one obtains a Hermitian preconditioned matrix by (implicitly) working with the modified linear system

$$L^{-1}AL^{-H}y = L^{-1}b, \quad x = L^{-H}y,$$

where $M = LL^H$ and superscript H denotes the Hermitian transpose. If the original problem is Hermitian but the preconditioner is *indefinite*, then we consider this as a non-Hermitian problem.

One might hope to improve the iteration (2.1) by introducing a parameter a_k and setting

$$x_{k+1} = x_k + a_k(b - Ax_k) \tag{2.5}$$

Since the residual satisfies $r_{k+1} = r_k - a_k A r_k$, one can minimize the 2-norm of r_{k+1} by choosing

$$a_k = \frac{\langle r_k, Ar_k \rangle}{\langle Ar_k, Ar_k \rangle}. \tag{2.6}$$

If the matrix A is Hermitian and positive definite, one might instead minimize the A-norm of the error, $\|e_{k+1}\|_A \equiv \langle e_{k+1}, Ae_{k+1} \rangle^{1/2}$. Since the error satisfies $e_{k+1} = e_k - a_k r_k$, the coefficient that minimizes this error norm is

$$a_k = \frac{\langle e_k, Ar_k \rangle}{\langle r_k, Ar_k \rangle} = \frac{\langle r_k, r_k \rangle}{\langle r_k, Ar_k \rangle}. \tag{2.7}$$

For Hermitian positive definite problems, the iteration (2.5) with coefficient formula (2.7) is called the method of *steepest descent*, because if the problem of solving the linear system is identified with that of minimizing the quadratic form $x^H Ax - 2b^H x$ (which has its minimum where $Ax = b$), then the negative gradient or direction of steepest descent of this function at $x = x_k$ is $r_k = b - Ax_k$. The coefficient formula (2.6), which can be used with arbitrary nonsingular matrices A, does not have a special name like steepest descent, but it is a special case of a number of different methods. In particular, it can be called Orthomin(1).

By choosing a_k as in (2.6), the Orthomin(1) method produces a residual r_{k+1} that is equal to r_k minus its *projection* onto Ar_k. It follows that $\|r_{k+1}\| \leq \|r_k\|$, with equality if and only if r_k is already orthogonal to Ar_k. Recall the definition of the *field of values* $\mathcal{F}(\mathcal{B})$ of a matrix B as the set of all complex numbers of the form $y^H By/y^H y$, where y is any complex vector other than the zero vector.

Theorem 2.2.1 The 2-norm of the residual in iteration (2.5) with coefficient formula (2.6) decreases strictly monotonically for every initial vector r_0 if and only if $0 \notin \mathcal{F}(A^H)$.

Proof: If $0 \in \mathcal{F}(A^H)$ and r_0 is a nonzero vector satisfying $\langle r_0, Ar_0 \rangle \equiv r_0^H A^H r_0 = 0$, then $\|r_1\| = \|r_0\|$. On the other hand, if $0 \notin \mathcal{F}(A^H)$, then $\langle r_k, Ar_k \rangle$ cannot be 0 for any k and $\|r_{k+1}\| < \|r_k\|$. $\Box$

Since the field of values of A^H is just the complex conjugate of the field of values of A, the condition in the theorem can be replaced by $0 \notin \mathcal{F}(A)$.

Suppose $0 \notin \mathcal{F}(A^H)$. To show that the method (2.5–2.6) converges to the solution $A^{-1}b$, we will show not only that the 2-norm of the residual is reduced at each step but that it is reduced by at least some fixed factor, independent of k. Since the field of values is a closed set, if $0 \notin \mathcal{F}(A^H)$ then there is a positive number d — the distance of $\mathcal{F}(A^H)$ from the origin — such that $|\frac{y^H A^H y}{y^H y}| \geq d$ for all complex vectors $y \neq 0$. From (2.5) it follows that

$$r_{k+1} = r_k - a_k A r_k,$$

and taking the inner product of r_{k+1} with itself, we have

$$\langle r_{k+1}, r_{k+1} \rangle = \langle r_k, r_k \rangle - \frac{|\langle r_k, Ar_k \rangle|^2}{\langle Ar_k, Ar_k \rangle},$$

which can be written in the form

$$\|r_{k+1}\|^2 = \|r_k\|^2 \left(1 - \left| \frac{r_k^H A^H r_k}{r_k^H r_k} \right|^2 \cdot \left(\frac{\|r_k\|}{\|Ar_k\|} \right)^2 \right). \tag{2.8}$$

Bounding the last two factors in (2.8) independently, in terms of d and $\|A\|$, we have

$$\|r_{k+1}\|^2 \leq \|r_k\|^2 \left(1 - d^2/\|A\|^2\right)$$

We have proved the following theorem:

Theorem 2.2.2. The iteration (2.5) with coefficient formula (2.6) converges to the solution $A^{-1}b$ for all initial vectors r_0 if and only if $0 \notin \mathcal{F}(A^H)$. In this case, the 2-norm of the residual satisfies

$$\|r_{k+1}\| \leq \sqrt{1 - d^2/\|A\|^2}\ \|r_k\| \tag{2.9}$$

for all k, where d is the distance from the origin to the field of values of A^H.

In the special case where A is real and the Hermitian part of A, $H(A) \equiv (A + A^H)/2$, is positive definite, the distance d in (2.9) is just the smallest eigenvalue of $H(A)$. This is because the field of values of $H(A)$ is the real part of the field of values of A^H, which is convex and symmetric about the real axis and hence has its closest point to the origin on the real axis.

The bound (2.9) on the rate at which the 2-norm of the residual is reduced is not necessarily sharp, since the vectors r_k for which the first factor in (2.8) is equal to d^2 are not necessarily the ones for which the second factor is $1/\|A\|^2$. Sometimes a stronger bound can be obtained by noting that, because of the choice of a_k,

$$\|r_{k+1}\| \le \|I - \alpha A\| \cdot \|r_k\|, \tag{2.10}$$

for *any* coefficient α. In the special case where A is Hermitian and positive definite, consider $\alpha = 2/(\lambda_n + \lambda_1)$, where λ_n is the largest and λ_1 the smallest eigenvalue of A. Inequality (2.10) then implies

$$\|r_{k+1}\| \le \max_{i=1,\ldots,n} \left| 1 - \frac{2\lambda_i}{\lambda_n + \lambda_1} \right| \cdot \|r_k\| \le \left(\frac{\kappa - 1}{\kappa + 1} \right) \|r_k\|, \tag{2.11}$$

where $\kappa = \lambda_n/\lambda_1$ is the condition number of A. (For the Hermitian positive definite case, expression (2.9) gives the significantly weaker bound $\|r_{k+1}\| \le \sqrt{1 - \kappa^{-2}}\ \|r_k\| \approx (1 - \frac{1}{2}\kappa^{-2})\ \|r_k\|$.)

The same argument applied to the *steepest descent* method for Hermitian positive definite problems shows that for that algorithm

$$\|e_{k+1}\|_A \le \left(\frac{\kappa - 1}{\kappa + 1} \right) \|e_k\|_A. \tag{2.12}$$

2.3 Orthomin(2) and CG

The Orthomin(1) and steepest descent iterations of the previous section can be written in the form

$$x_{k+1} = x_k + a_k p_k$$

where the *direction vector* p_k is equal to the residual r_k. The new residual and error vectors then satisfy

$$r_{k+1} = r_k - a_k A p_k, \quad e_{k+1} = e_k - a_k p_k,$$

where the coefficient a_k is chosen so that either r_{k+1} is orthogonal to Ap_k (Orthomin(1)) or, in the case of Hermitian positive definite A, so that e_{k+1} is A-orthogonal to p_k (steepest descent). Note, however, that r_{k+1} is not orthogonal to the previous vector Ap_{k-1} in the Orthomin(1) method and e_{k+1} is not A-orthogonal to the previous vector p_{k-1} in the steepest descent method.

If, in the Orthomin iteration, instead of subtracting off the projection of r_k in the direction Ar_k, we subtracted off its projection in a direction like Ar_k but orthogonal to Ap_{k-1}; i.e., in the direction $A\tilde{p}_k$, where

$$\tilde{p}_k = r_k - \frac{\langle Ar_k, Ap_{k-1} \rangle}{\langle Ap_{k-1}, Ap_{k-1} \rangle} p_{k-1},$$

then we would have

$$\langle r_{k+1}, A\tilde{p}_k \rangle = \langle r_{k+1}, Ap_{k-1} \rangle = 0.$$

Now the residual norm is minimized in the *plane* spanned by Ar_k and Ap_{k-1}, since we can write

$$r_{k+1} = r_k - a_k A r_k + a_k b_{k-1} A p_{k-1},$$

and the coefficients force orthogonality between r_{k+1} and span$\{Ar_k, Ap_{k-1}\}$.

The new algorithm, known as Orthomin(2), is the following:

> Given an initial guess x_0, compute $r_0 = b - Ax_0$ and set $p_0 = r_0$. For $k = 1, 2, \ldots$,
>
> Compute Ap_{k-1}.
>
> Set $x_k = x_{k-1} + a_{k-1} p_{k-1}$, where $a_{k-1} = \frac{\langle r_{k-1}, Ap_{k-1} \rangle}{\langle Ap_{k-1}, Ap_{k-1} \rangle}$.
>
> Compute $r_k = r_{k-1} - a_{k-1} A p_{k-1}$.
>
> Set $p_k = r_k - b_{k-1} p_{k-1}$, where $b_{k-1} = \frac{\langle Ar_k, Ap_{k-1} \rangle}{\langle Ap_{k-1}, Ap_{k-1} \rangle}$

The new algorithm can also fail. If $\langle r_0, Ar_0 \rangle = 0$, then r_1 will be equal to r_0, and p_1 will be 0. An attempt to compute the coefficient a_1 will result in dividing 0 by 0. As for Orthomin(1), however, it can be shown that Orthomin(2) cannot fail if $0 \notin \mathcal{F}(A^H)$. If an Orthomin(2) step does succeed, then the 2-norm of the residual is reduced by at least as much as it would be reduced by an Orthomin(1) step from the same point. This is because the residual norm is minimized over a larger space — $r_k + \text{span}\{Ar_k, Ap_{k-1}\}$ instead of $r_k + \text{span}\{Ar_k\}$. It follows that the bound (2.9) holds also for Orthomin(2) when $0 \notin \mathcal{F}(A^H)$. Unfortunately, no stronger bounds on the residual norm are known for Orthomin(2) applied to a general matrix whose field of values does not contain the origin, although, in practice, it may perform significantly better than Orthomin(1).

In the special case when A is *Hermitian*, if the vectors at steps 1 through $k+1$ of the Orthomin(2) algorithm are defined, then r_{k+1} is minimized not just over the two-dimensional space $r_k + \text{span}\{Ar_k, Ap_{k-1}\}$, but over the $(k+1)$-dimensional space $r_0 + \text{span}\{Ap_0, \ldots, Ap_k\}$.

Theorem 2.3.1. Suppose A is Hermitian and the coefficients $a_0, \ldots, a_{k-1}$ are nonzero and the vectors $r_1, \ldots, r_{k+1}$ and $p_1, \ldots, p_{k+1}$ in the Orthomin(2) algorithm are defined. Then

$$\langle r_{k+1}, Ap_j \rangle = \langle Ap_{k+1}, Ap_j \rangle = 0, \quad \forall j \leq k.$$

It follows that of all vectors in the affine space

$$r_0 + \text{span}\{Ar_0, A^2 r_0, \ldots, A^{k+1} r_0\}, \tag{2.13}$$

r_{k+1} has the smallest Euclidean norm. It also follows that if $a_0, \ldots, a_{n-2}$ are nonzero and $r_1, \ldots, r_n$ and $p_1, \ldots, p_n$ are defined, then $r_n = 0$.

Proof: By construction, we have $\langle r_1, Ap_0 \rangle = \langle Ap_1, Ap_0 \rangle = 0$. Assume $\langle r_k, Ap_j \rangle = \langle Ap_k, Ap_j \rangle = 0$, $\forall j \leq k-1$. The coefficients at step $k+1$ are chosen to force $\langle r_{k+1}, Ap_k \rangle = \langle Ap_{k+1}, Ap_k \rangle = 0$. For $j \leq k-1$, we have

$$\langle r_{k+1}, Ap_j \rangle = \langle r_k - a_k Ap_k, Ap_j \rangle = 0$$

by the induction hypothesis. Also

$$\begin{aligned}\langle Ap_{k+1}, Ap_j\rangle &= \langle A(r_{k+1} - b_k p_k), Ap_j\rangle \\ &= \langle Ar_{k+1}, a_j^{-1}(r_j - r_{j+1})\rangle \\ &= \bar{a}_j^{-1}\langle Ar_{k+1}, p_j + b_{j-1}p_{j-1} - p_{j+1} - b_j p_j\rangle \\ &= \bar{a}_j^{-1}\langle r_{k+1}, A(p_j + b_{j-1}p_{j-1} - p_{j+1} - b_j p_j)\rangle \\ &= 0,\end{aligned}$$

with the next to last equality holding because $A = A^H$. It is justified to write a_j^{-1} since by assumption $a_j \neq 0$.

It is easily checked by induction that r_{k+1} lies in the space (2.13) and that $\text{span}\{Ap_0, \ldots, Ap_k\} = \text{span}\{Ar_0, \ldots, A^{k+1}r_0\}$. Since r_{k+1} is orthogonal to $\text{span}\{Ap_0, \ldots, Ap_k\}$, it follows that r_{k+1} is the vector in the space (2.13) with minimal Euclidean norm. For $k = n-1$, this implies $r_n = 0$. $\Box$

The assumption in Theorem 2.3.1 that $r_1, \ldots, r_{k+1}$ and $p_1, \ldots, p_{k+1}$ be defined is actually implied by the other hypothesis. It can be shown that these vectors are defined provided $a_0, \ldots, a_{k-1}$ are defined and nonzero and r_k is nonzero.

An algorithm that approximates the solution of a Hermitian linear system $Ax = b$ by minimizing the residual over the affine space (2.13) is known as the *MINRES* algorithm. It should not be implemented in the form given here, however, unless the matrix is positive (or negative) definite, because, as noted, this iteration can fail if $0 \in \mathcal{F}(A)$. An appropriate implementation of the MINRES algorithm for Hermitian indefinite linear systems is derived in Section 2.5.

In a similar way, the steepest descent method for Hermitian positive definite matrices can be modified so that it eliminates the A-projection of the error in a direction that is already A-orthogonal to the previous direction vector; i.e., in the direction

$$\tilde{p}_k = r_k - \frac{\langle r_k, Ap_{k-1}\rangle}{\langle p_{k-1}, Ap_{k-1}\rangle} p_{k-1}.$$

Then we have

$$\langle e_{k+1}, A\tilde{p}_k\rangle = \langle e_{k+1}, Ap_{k-1}\rangle = 0,$$

and the A-norm of the error is minimized over the two-dimensional affine space $e_k + \text{span}\{r_k, p_{k-1}\}$. The algorithm that does this is called the *conjugate gradient* method (CG). It is usually implemented with slightly different (but equivalent) coefficient formulas, as below:

Algorithm 2. Conjugate Gradient Method (CG).
(For Hermitian Positive Definite Problems)

Given an initial guess x_0, compute $r_0 = b - Ax_0$ and set $p_0 = r_0$.
For $k = 1, 2, \ldots,$

Compute Ap_{k-1}.

Set $x_k = x_{k-1} + a_{k-1}p_{k-1}$, where $a_{k-1} = \frac{\langle r_{k-1}, r_{k-1}\rangle}{\langle p_{k-1}, Ap_{k-1}\rangle}$.

Compute $r_k = r_{k-1} - a_{k-1}Ap_{k-1}$.

Set $p_k = r_k + b_{k-1}p_{k-1}$, where $b_{k-1} = \frac{\langle r_k, r_k\rangle}{\langle r_{k-1}, r_{k-1}\rangle}$.

It is left as an exercise for the reader to prove that these coefficient formulas are equivalent to the more obvious expressions

$$a_{k-1} = \frac{\langle r_{k-1}, p_{k-1}\rangle}{\langle p_{k-1}, Ap_{k-1}\rangle}, \quad b_{k-1} = -\frac{\langle r_k, Ap_{k-1}\rangle}{\langle p_{k-1}, Ap_{k-1}\rangle} \tag{2.14}$$

Since the CG algorithm is used only with positive definite matrices, the coefficients are always defined, and it can be shown, analogously to the MINRES method, that the A-norm of the error is actually minimized over the much larger affine space $e_0 + \text{span}\{p_0, p_1, \ldots, p_k\}$.

Theorem 2.3.2. Assume A is Hermitian and positive definite. The CG algorithm generates the exact solution to the linear system $Ax = b$ in at most n steps. The error, residual, and direction vectors generated before the exact solution is obtained are well-defined and satisfy

$$\langle e_{k+1}, Ap_j\rangle = \langle p_{k+1}, Ap_j\rangle = \langle r_{k+1}, r_j\rangle = 0, \quad \forall j \leq k.$$

It follows that of all vectors in the affine space

$$e_0 + \mathrm{span}\{Ae_0, A^2 e_0, \ldots, A^{k+1} e_0\}, \tag{2.15}$$

e_{k+1} has the smallest A-norm.

Proof: Since A is positive definite, it is clear that the coefficients in the CG algorithm are well-defined unless a residual vector is zero, in which case the exact solution has been found. Assume that $r_0, \ldots, r_k$ are nonzero. By the choice of a_0, it is clear that $\langle r_1, r_0 \rangle = \langle e_1, Ap_0 \rangle = 0$, and from the choice of b_0 it follows that

$$\begin{aligned}
\langle p_1, Ap_0 \rangle &= \langle r_1, Ap_0 \rangle + \frac{\langle r_1, r_1 \rangle}{\langle r_0, r_0 \rangle} \langle p_0, Ap_0 \rangle \\
&= \langle r_1, a_0^{-1}(r_0 - r_1) \rangle + a_0^{-1} \langle r_1, r_1 \rangle = 0,
\end{aligned}$$

with the last equality holding because $\langle r_1, r_0 \rangle = 0$ and a_0^{-1} is real. Assume

$$\langle e_k, Ap_j \rangle = \langle p_k, Ap_j \rangle = \langle r_k, r_j \rangle = 0, \quad \forall j \leq k-1.$$

Then we also have

$$\langle p_k, Ap_k \rangle = \langle r_k + b_{k-1} p_{k-1}, Ap_k \rangle = \langle r_k, Ap_k \rangle$$

$$\langle r_k, p_k \rangle = \langle r_k, r_k + b_{k-1} p_{k-1} \rangle = \langle r_k, r_k \rangle,$$

and so, by the choice of a_k, it follows that

$$\langle r_{k+1}, r_k \rangle = \langle r_k, r_k \rangle - a_k \langle r_k, Ap_k \rangle = \langle r_k, r_k \rangle - \langle r_k, r_k \rangle = 0$$

$$\langle e_{k+1}, Ap_k \rangle = \langle r_{k+1}, p_k \rangle = \langle r_k, p_k \rangle - a_k \langle Ap_k, p_k \rangle = \langle r_k, r_k \rangle - \langle r_k, r_k \rangle = 0.$$

From the choice of b_k, we have

$$\langle p_{k+1}, Ap_k \rangle = \langle r_{k+1}, Ap_k \rangle + \frac{\langle r_{k+1}, r_{k+1} \rangle}{\langle r_k, r_k \rangle} \langle p_k, Ap_k \rangle$$

$$= \langle r_{k+1}, a_k^{-1}(r_k - r_{k+1}) \rangle + a_k^{-1} \langle r_{k+1}, r_{k+1} \rangle = 0.$$

For $j \leq k-1$, we have

$$\langle e_{k+1}, Ap_j \rangle = \langle e_k - a_k p_k, Ap_j \rangle = 0$$

$$\langle r_{k+1}, r_j \rangle = \langle r_k - a_k A p_k, r_j \rangle = -a_k \langle p_k, A(p_j - b_{j-1} p_{j-1}) \rangle = 0$$

$$\langle p_{k+1}, Ap_j \rangle = \langle r_{k+1} + b_k p_k, Ap_j \rangle = \langle r_{k+1}, a_j^{-1}(r_j - r_{j+1}) \rangle = 0,$$

and so by induction the desired equalities are established.

It is easily checked by induction that e_{k+1} lies in the space (2.15) and that $\mathrm{span}\{p_0, \ldots, p_k\} = \mathrm{span}\{Ae_0, \ldots, A^{k+1}e_0\}$. Since e_{k+1} is A-orthogonal to $\mathrm{span}\{p_0, \ldots, p_k\}$, it follows that e_{k+1} is the vector in the space (2.15) with minimal A-norm, and it also follows that if the exact solution is not obtained before step n, then $e_n = 0$. □