19. A matrix A is *pentadiagonal* if $a_{ij} = 0$ whenever $|i - j| > 2$.
    (a) Construct an algorithm to efficiently compute the Crout decomposition of a pentadiagonal matrix.
    (b) How many operations are required by the algorithm from part (a)?
    (c) How many operations are needed to carry out forward and backward substitution using the decomposition obtained from part (a)?

3.8 ITERATIVE TECHNIQUES FOR LINEAR SYSTEMS: BASIC CONCEPTS AND METHODS

Having just devoted several sections to the development of direct techniques for linear systems—techniques that produce an answer in a fixed number of operations—it is natural to ask why we would want or even need to develop iterative techniques. For systems of small dimension, there is no need. Direct techniques will perform very efficiently. However, linear systems arising from practical applications will frequently be quite large. The coefficient matrices associated with these systems also tend to be sparse, meaning that only a small percentage of the entries are nonzero. We will encounter systems of this type in Chapter 9 when we treat the solution of elliptic partial differential equations.

For systems with large, sparse coefficient matrices, direct techniques are often less efficient than iterative techniques. Even though multiple iterations need to be performed to achieve convergence, an iterative solution will typically require fewer total operations than a direct solution. It will often happen that the nonzero elements in the coefficient matrix will exhibit a well-defined pattern. In these cases, an iterative solution will not require the storage of the coefficient matrix at all—only the structure of the equations will be needed. As an added bonus, iterative techniques are generally insensitive to roundoff error.

Basic Concepts

Basic iterative techniques for the solution of linear systems of equations are analogous to the fixed-point techniques which were discussed in Chapter 2. The original linear system $A\mathbf{x} = \mathbf{b}$, which can be interpreted as the rootfinding problem

$$\text{find the } n\text{-vector } \mathbf{x} \text{ so that } A\mathbf{x} - \mathbf{b} = \mathbf{0},$$

is first converted to the fixed point problem

$$\text{find the } n\text{-vector } \mathbf{x} \text{ so that } \mathbf{x} = T\mathbf{x} + \mathbf{c},$$

for some matrix T and vector $\mathbf{c}$. Next, starting from some initial approximation to the solution of the fixed point problem, $\mathbf{x}^{(0)}$, a sequence of vectors $\{\mathbf{x}^{(k)}\}$ is computed according to the rule

$$\mathbf{x}^{(k+1)} = T\mathbf{x}^{(k)} + \mathbf{c}. \tag{1}$$

Within this context, the matrix T is called the *iteration matrix*. The functional iteration is terminated when some appropriate measure of the difference between

successive vectors in the sequence, $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(k+1)}$, falls below a user specified tolerance.

The analysis of the functional iteration scheme given by (1) boils down to four important questions. First, what conditions guarantee a unique solution to the fixed point problem? Second, under what conditions will the sequence generated by (1) converge to this unique fixed point? Third, when the sequence generated by (1) converges, how quickly does it converge? Fourth, what conditions must the matrix T and the vector $\mathbf{c}$ satisfy in order for the fixed point problem to be consistent with the original rootfinding problem (i.e., for the two problems to have the same solution)?

The following theorem from general matrix theory plays a major role in establishing answers to these questions.

Theorem. Let A be an $n \times n$ matrix. Then the following statements are equivalent:

1. $\rho(A) < 1$, where $\rho(A)$ denotes the spectral radius of A;
2. $A^k \to 0$ as $k \to \infty$; and
3. $A^k\mathbf{x} \to \mathbf{0}$ as $k \to \infty$ for any vector $\mathbf{x}$.

A proof of this result can be found in Isaacson and Keller [1].

Let's start with the question of the uniqueness of the solution of the fixed point problem. Manipulating the fixed-point equation, we find

$$\begin{aligned} \mathbf{x} = T\mathbf{x} + \mathbf{c} &\Leftrightarrow \mathbf{x} - T\mathbf{x} = \mathbf{c} \\ &\Leftrightarrow (I - T)\mathbf{x} = \mathbf{c}. \end{aligned}$$

From this last equation, it follows that the fixed point problem has a unique solution if and only if the matrix $I - T$ is nonsingular. A sufficient condition for $I - T$ to be nonsingular is $\rho(T) < 1$ (see Exercise 10 from Section 3.3). Hence, the fixed point problem is guaranteed to have a unique solution whenever $\rho(T) < 1$.

To establish convergence, let $\mathbf{x}^\star$ denote the solution to the fixed point problem, and define the iteration error vector by $\mathbf{e}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}^\star$. Subtracting the equation $\mathbf{x}^\star = T\mathbf{x}^\star + \mathbf{c}$ from equation (1) yields the error evolution equation

$$\mathbf{e}^{(k+1)} = T\mathbf{e}^{(k)}.$$

Working backward through this equation, we find

$$\begin{aligned} \mathbf{e}^{(k+1)} &= T\mathbf{e}^{(k)} \\ &= T(T\mathbf{e}^{(k-1)}) = T^2\mathbf{e}^{(k-1)} \\ &= T^2(T\mathbf{e}^{(k-2)}) = T^3\mathbf{e}^{(k-2)} \\ &= \cdots \\ &= T^{k+1}\mathbf{e}^{(0)}. \end{aligned}$$

Ideally, $\mathbf{e}^{(k+1)}$ should approach $\mathbf{0}$ as $k \to \infty$ for any choice of initial vector $\mathbf{x}^{(0)}$, that is, for any initial error vector $\mathbf{e}^{(0)}$. The theorem stated above indicates that this will happen if and only if $\rho(T) < 1$. Hence, the iteration scheme defined by equation (1) will converge for any choice of the initial vector $\mathbf{x}^{(0)}$ if and only if $\rho(T) < 1$.

From the error evolution equation, we find

$$\|\mathbf{e}^{(k+1)}\| \leq \|T\| \, \|\mathbf{e}^{(k)}\|$$

for any vector norm $\| \cdot \|$ and associated natural matrix norm. Provided $\|T\| < 1$, it can be shown that

$$\|\mathbf{e}^{(k)}\| \leq \frac{\|T\|^k}{1 - \|T\|} \|\mathbf{x}^{(1)} - \mathbf{x}^{(0)}\|$$

(see Exercise 14). These two inequalities imply that the sequence $\{\mathbf{x}^{(k)}\}$ converges linearly with an asymptotic error constant that is less than or equal to $\|T\|$. Carrying out a more precise analysis, it can be shown that the asymptotic error constant is equal to $\rho(T)$. The proof is based on the fact that $\rho(T)$ is the greatest lower bound for all natural matrix norms of T. See Ortega [2] for details. Thus, the smaller the spectral radius of the iteration matrix, the faster the convergence of the corresponding iterative scheme.

The final preliminary issue to discuss is that of consistency. In order for the iteration defined by (1) to be of any practical use, the solution of the fixed point problem, $\mathbf{x}^* = (I - T)^{-1}\mathbf{c}$, must be identical to the solution of the original linear system, $\mathbf{x}^* = A^{-1}\mathbf{b}$. Hence, when constructing the fixed point problem from the linear system, we must be certain that T and $\mathbf{c}$ satisfy the relation

$$(I - T)^{-1}\mathbf{c} = A^{-1}\mathbf{b}.$$

Splitting Methods

A broad class of consistent iterative methods, known as splitting methods, can be constructed by introducing the notion of a splitting.

> **Definition.** Let A be a given $n \times n$ matrix. If M and N are $n \times n$ matrices with M nonsingular and $A = M - N$, then the pair (M, N) is called a SPLITTING of the matrix A.

So let's suppose that (M, N) forms a splitting of the matrix A. Then

$$A\mathbf{x} = \mathbf{b} \quad \text{is equivalent to} \quad (M - N)\mathbf{x} = \mathbf{b}.$$

Clearing the parentheses and transposing the term involving the matrix N to the right-hand side of the equation yields

$$M\mathbf{x} = N\mathbf{x} + \mathbf{b}.$$

Finally, premultiplying by M^{-1} produces

$$\mathbf{x} = M^{-1}N\mathbf{x} + M^{-1}\mathbf{b}.$$

Hence the splitting $A = M - N$ determines the fixed point problem $\mathbf{x} = T\mathbf{x} + \mathbf{c}$ and associated iteration scheme $\mathbf{x}^{(k+1)} = T\mathbf{x}^{(k)} + \mathbf{c}$, where

$$T = M^{-1}N \quad \text{and} \quad \mathbf{c} = M^{-1}\mathbf{b}.$$

To establish that splitting methods are always consistent, first note that with $T = M^{-1}N$

$$\begin{aligned} I - T &= I - M^{-1}N \\ &= M^{-1}(M - N) \\ &= M^{-1}A. \end{aligned}$$

Therefore, $(I - T)^{-1} = A^{-1}M$. Finally, with $\mathbf{c} = M^{-1}\mathbf{b}$,

$$\begin{aligned} (I - T)^{-1}\mathbf{c} &= A^{-1}M \cdot M^{-1}\mathbf{b} \\ &= A^{-1}\mathbf{b}, \end{aligned}$$

as required. Descriptions of the three most commonly used splitting methods are presented below. The convergence properties for these three methods will then be discussed at the end of the section.

The Jacobi Method, Gauss-Seidel Method, and SOR Method

To identify the splittings associated with the Jacobi method, the Gauss-Seidel method, and the SOR method, first express the coefficient matrix A in the form

$$A = D - L - U.$$

Here, D is the diagonal part of A, $-L$ is the strictly lower triangular part of A, and $-U$ is the strictly upper triangular part. It is important to keep in mind that the matrices L and U used here are in no way related to the LU decomposition of the coefficient matrix. As an example, suppose

$$A = \begin{bmatrix} 5 & 1 & 2 \\ -3 & 9 & 4 \\ 1 & 2 & -7 \end{bmatrix}.$$

Then

$$D = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & -7 \end{bmatrix}, \quad L = \begin{bmatrix} 0 & 0 & 0 \\ 3 & 0 & 0 \\ -1 & -2 & 0 \end{bmatrix} \quad \text{and} \quad U = \begin{bmatrix} 0 & -1 & -2 \\ 0 & 0 & -4 \\ 0 & 0 & 0 \end{bmatrix}.$$

The Jacobi method is based on the splitting $M = D$ and $N = L + U$. In order for M to be nonsingular, it must be the case that, for each i, $d_{i,i} \equiv a_{i,i} \neq 0$. If this relationship does not hold for even a single value of i, then the equations in the system must be reordered before the Jacobi method can be applied. With

the specific choice of splitting indicated above, the iteration scheme for the Jacobi method is defined by

$$\mathbf{x}^{(k+1)} = T_{jac}\mathbf{x}^{(k)} + \mathbf{c}_{jac}, \tag{2}$$

where

$$T_{jac} = D^{-1}(L+U) \quad \text{and} \quad \mathbf{c}_{jac} = D^{-1}\mathbf{b}.$$

Taking into account the structure of the iteration matrix, T_{jac}, and the vector $\mathbf{c}_{jac}$, the individual components of equation (2) can be written as

$$x_i^{(k+1)} = \frac{1}{a_{i,i}}\left[b_i - \sum_{j=1}^{i-1} a_{i,j}x_j^{(k)} - \sum_{j=i+1}^{n} a_{i,j}x_j^{(k)}\right]. \tag{3}$$

Hence, the Jacobi method is equivalent to solving the i-th equation in the system for the unknown x_i.

Since, in general, the value of $x_i^{(k)}$ will be needed to compute $x_j^{(k+1)}$ for each $j = i+1, i+2, \ldots, n$, the value of $x_i^{(k)}$ cannot be overwritten by the newly computed value of $x_i^{(k+1)}$. This implies that when implementing the Jacobi method, two storage arrays will have to be maintained, one for the old approximation vector $\mathbf{x}^{(k)}$ and one for the new approximation vector $\mathbf{x}^{(k+1)}$. It also follows that the components of $\mathbf{x}^{(k+1)}$ can be computed in any order and that, on a parallel or vector machine, all components of $\mathbf{x}^{(k+1)}$ can be computed simultaneously. For this reason, the Jacobi method is often called the method of Simultaneous Relaxation.

EXAMPLE 3.22 The Jacobi Method in Action

Consider the system of equations

$$\begin{array}{rcrcrcr} 5x_1 & + & x_2 & + & 2x_3 & = & 10 \\ -3x_1 & + & 9x_2 & + & 4x_3 & = & -14 \\ x_1 & + & 2x_2 & - & 7x_3 & = & -33\,. \end{array}$$

The Jacobi method, when applied to this system, will produce the sequence of approximations $\{\mathbf{x}^{(k)}\}$ according to the rules

$$\begin{aligned} x_1^{(k+1)} &= \frac{1}{5}\left[10 - x_2^{(k)} - 2x_3^{(k)}\right] \\ x_2^{(k+1)} &= \frac{1}{9}\left[-14 + 3x_1^{(k)} - 4x_3^{(k)}\right] \\ x_3^{(k+1)} &= -\frac{1}{7}\left[-33 - x_1^{(k)} - 2x_2^{(k)}\right]. \end{aligned}$$

If we start with $\mathbf{x}^{(0)} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$, then the components of $\mathbf{x}^{(1)}$ are

$$x_1^{(1)} = \frac{1}{5}\left[10 - x_2^{(0)} - 2x_3^{(0)}\right] = 2$$
$$x_2^{(1)} = \frac{1}{9}\left[-14 + 3x_1^{(0)} - 4x_3^{(0)}\right] = -\frac{14}{9}$$
$$x_3^{(1)} = -\frac{1}{7}\left[-33 - x_1^{(0)} - 2x_2^{(0)}\right] = \frac{33}{7}.$$

The following table summarizes the 14 iterations of the Jacobi method that were needed for $\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|_\infty$ to fall below 5×10^{-4}. Other stopping criteria can be imposed, but this is the most common. The exact solution to this problem is

$$\mathbf{x} = \begin{bmatrix} 1 & -3 & 4 \end{bmatrix}^T.$$

Hence, $\|\mathbf{x} - \mathbf{x}^{(14)}\|_\infty = 2.43 \times 10^{-4}$.

k	$\mathbf{x}^{(k)}$		
0	$\begin{bmatrix} 0.000000 & 0.000000 & 0.000000 \end{bmatrix}^T$		
1	$\begin{bmatrix} 2.000000 & -1.555556 & 4.714286 \end{bmatrix}^T$		
2	$\begin{bmatrix} 0.425397 & -2.984127 & 4.555556 \end{bmatrix}^T$		
3	$\begin{bmatrix} 0.774603 & -3.438448 & 3.922449 \end{bmatrix}^T$		
4	$\begin{bmatrix} 1.118710 & -3.040665 & 3.842530 \end{bmatrix}^T$		
5	$\begin{bmatrix} 1.071121 & -2.890443 & 4.005340 \end{bmatrix}^T$		
6	$\begin{bmatrix} 0.975953 & -2.978666 & 4.041462 \end{bmatrix}^T$		
7	$\begin{bmatrix} 0.979148 & -3.026443 & 4.002660 \end{bmatrix}^T$		
8	$\begin{bmatrix} 1.004225 & -3.008133 & 3.989466 \end{bmatrix}^T$		
9	$\begin{bmatrix} 1.005840 & -2.993910 & 3.998280 \end{bmatrix}^T$		
10	$\begin{bmatrix} 0.999470 & -2.997289 & 4.002574 \end{bmatrix}^T$		
11	$\begin{bmatrix} 0.998428 & -3.001321 & 4.000699 \end{bmatrix}^T$		
12	$\begin{bmatrix} 0.999985 & -3.000835 & 3.999398 \end{bmatrix}^T$		
13	$\begin{bmatrix} 1.000408 & -2.999738 & 3.999759 \end{bmatrix}^T$		
14	$\begin{bmatrix} 1.000044 & -2.999757 & 4.000133 \end{bmatrix}^T$		

An obvious improvement that can be made to the Jacobi method is to use the value of $x_i^{(k+1)}$ as soon as it has been calculated in the computation of all subsequent entries in the vector $\mathbf{x}^{(k+1)}$, rather than waiting until the next iteration. After all, $x_i^{(k+1)}$ is supposed to be a better approximation to x_i than $x_i^{(k)}$. This modification amounts to changing equation (3) to

$$x_i^{(k+1)} = \frac{1}{a_{i,i}}\left[b_i - \sum_{j=1}^{i-1} a_{i,j}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{i,j}x_j^{(k)}\right]; \tag{4}$$

the only difference between the equations is that the superscript on x in the first summation is now $k+1$. The iteration scheme corresponding to equation (4) is known as the Gauss-Seidel method. Note that the Gauss-Seidel method is not vectorizable. The entries in $\mathbf{x}^{(k+1)}$ must be computed in succession. Hence, the Gauss-Seidel method is also known as Successive Relaxation.

Working backward from equation (4), we find that the splitting upon which the Gauss-Seidel method is based is

$$M = D - L \quad \text{and} \quad N = U.$$

Thus, the iteration matrix for the Gauss-Seidel method is given by

$$T_{gs} = (D - L)^{-1}U,$$

and the vector $\mathbf{c}$ is given by

$$\mathbf{c}_{gs} = (D - L)^{-1}\mathbf{b}.$$

The necessary and sufficient condition for the matrix M to be nonsingular is the same as above: for each i, we must have $d_{i,i} \equiv a_{i,i} \neq 0$.

EXAMPLE 3.23 The Gauss-Seidel Method in Action

Reconsider the system of equations

$$\begin{array}{rcrcrcr} 5x_1 & + & x_2 & + & 2x_3 & = & 10 \\ -3x_1 & + & 9x_2 & + & 4x_3 & = & -14 \\ x_1 & + & 2x_2 & - & 7x_3 & = & -33. \end{array}$$

The Gauss-Seidel method, when applied to this system, will produce the sequence of approximations $\{\mathbf{x}^{(k)}\}$ according to the rules

$$\begin{aligned} x_1^{(k+1)} &= \frac{1}{5}\left[10 - x_2^{(k)} - 2x_3^{(k)}\right] \\ x_2^{(k+1)} &= \frac{1}{9}\left[-14 + 3x_1^{(k+1)} - 4x_3^{(k)}\right] \\ x_3^{(k+1)} &= -\frac{1}{7}\left[-33 - x_1^{(k+1)} - 2x_2^{(k+1)}\right]. \end{aligned}$$

If we start with $\mathbf{x}^{(0)} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$, then the components of $\mathbf{x}^{(1)}$ are

$$\begin{aligned} x_1^{(1)} &= \frac{1}{5}\left[10 - x_2^{(0)} - 2x_3^{(0)}\right] = 2 \\ x_2^{(1)} &= \frac{1}{9}\left[-14 + 3x_1^{(1)} - 4x_3^{(0)}\right] = -\frac{8}{9} \\ x_3^{(1)} &= -\frac{1}{7}\left[-33 - x_1^{(1)} - 2x_2^{(1)}\right] = \frac{299}{63}. \end{aligned}$$

The following table summarizes the 10 iterations of the Gauss-Seidel method that were needed for $\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|_\infty$ to fall below 5×10^{-4}. Recall that the exact solution to this problem is

$$\mathbf{x} = \begin{bmatrix} 1 & -3 & 4 \end{bmatrix}^T.$$

Hence, $\|\mathbf{x} - \mathbf{x}^{(10)}\|_\infty = 7.8 \times 10^{-5}$. Note that convergence is obtained with the Gauss-Seidel method in roughly 30% fewer iterations than the Jacobi method. Further, the error in the final Gauss-Seidel approximation is roughly one-third the error in the final Jacobi approximation.

k	$\mathbf{x}^{(k)}$		
0	$\begin{bmatrix} 0.000000 & 0.000000 & 0.000000 \end{bmatrix}^T$		
1	$\begin{bmatrix} 2.000000 & -0.888889 & 4.746032 \end{bmatrix}^T$		
2	$\begin{bmatrix} 0.279365 & -3.571781 & 3.733686 \end{bmatrix}^T$		
3	$\begin{bmatrix} 1.220882 & -2.808011 & 4.086409 \end{bmatrix}^T$		
4	$\begin{bmatrix} 0.927039 & -3.062724 & 3.971656 \end{bmatrix}^T$		
5	$\begin{bmatrix} 1.023883 & -2.979442 & 4.009286 \end{bmatrix}^T$		
6	$\begin{bmatrix} 0.992174 & -3.006736 & 3.996958 \end{bmatrix}^T$		
7	$\begin{bmatrix} 1.002564 & -2.997793 & 4.000997 \end{bmatrix}^T$		
8	$\begin{bmatrix} 0.999160 & -3.000723 & 3.999673 \end{bmatrix}^T$		
9	$\begin{bmatrix} 1.000275 & -2.999763 & 4.000107 \end{bmatrix}^T$		
10	$\begin{bmatrix} 0.999910 & -3.000078 & 3.999965 \end{bmatrix}^T$		

The final iterative technique that we will discuss in this section is the SOR method. An explanation for the name of the method will be provided shortly. This technique attempts to improve upon the convergence of the Gauss-Seidel method by computing $x_i^{(k+1)}$ as a weighted average of $x_i^{(k)}$ and the value produced by the Gauss-Seidel method, as given in equation (4). Let the weighting parameter, also known as a relaxation parameter, be denoted by ω. Then the analogue of equations (3) and (4) for the SOR method is

$$x_i^{(k+1)} = (1-\omega)x_i^{(k)} + \frac{\omega}{a_{i,i}}\left[b_i - \sum_{j=1}^{i-1} a_{i,j}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{i,j}x_j^{(k)}\right]. \tag{5}$$

Note that when $\omega = 1$, the SOR method reduces to the Gauss-Seidel method. Typically, there exists a range of ω values for which the SOR method will converge faster than the Gauss-Seidel method. The splitting associated with the SOR method is

$$M = \frac{1}{\omega}D - L \quad \text{and} \quad N = \left(\frac{1}{\omega} - 1\right)D + U.$$

Therefore,

$$T_{sor} = \left(\frac{1}{\omega}D - L\right)^{-1}\left[\left(\frac{1}{\omega} - 1\right)D + U\right]$$

and

$$\mathbf{c}_{sor} = \left(\frac{1}{\omega}D - L\right)^{-1}\mathbf{b}.$$

EXAMPLE 3.24 The SOR Method in Action

One more time we will consider the system of equations

$$\begin{array}{rcrcrcr} 5x_1 & + & x_2 & + & 2x_3 & = & 10 \\ -3x_1 & + & 9x_2 & + & 4x_3 & = & -14 \\ x_1 & + & 2x_2 & - & 7x_3 & = & -33\,. \end{array}$$

With $\omega = 0.9$, the iteration equations for the SOR method become

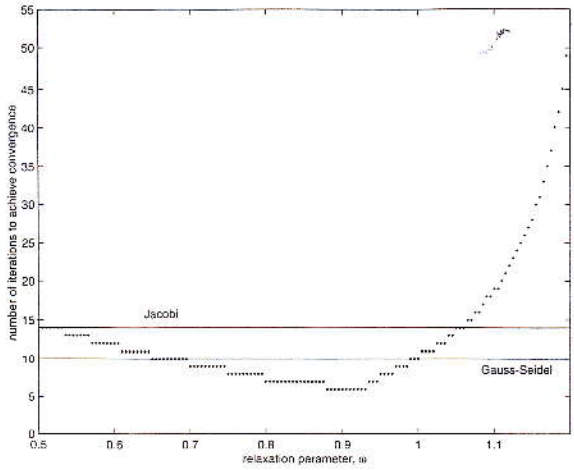
$$\begin{aligned} x_1^{(k+1)} &= 0.1x_1^{(k)} + \frac{0.9}{5}\left[10 - x_2^{(k)} - 2x_3^{(k)}\right] \\ x_2^{(k+1)} &= 0.1x_2^{(k)} + \frac{0.9}{9}\left[-14 + 3x_1^{(k+1)} - 4x_3^{(k)}\right] \\ x_3^{(k+1)} &= 0.1x_3^{(k)} - \frac{0.9}{7}\left[-33 - x_1^{(k+1)} - 2x_2^{(k+1)}\right]. \end{aligned}$$

If we start with $\mathbf{x}^{(0)} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$, then the components of $\mathbf{x}^{(1)}$ are

$$\begin{aligned} x_1^{(1)} &= 0.1x_1^{(0)} + \frac{0.9}{5}\left[10 - x_2^{(0)} - 2x_3^{(0)}\right] = 1.8 \\ x_2^{(1)} &= 0.1x_2^{(0)} + \frac{0.9}{9}\left[-14 + 3x_1^{(1)} - 4x_3^{(0)}\right] = -0.86 \\ x_3^{(1)} &= 0.1x_3^{(0)} - \frac{0.9}{7}\left[-33 - x_1^{(1)} - 2x_2^{(1)}\right] = 4.253143\,. \end{aligned}$$

The table shown below summarizes the 6 iterations of the SOR method that were needed for $\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|_\infty$ to fall below 5×10^{-4}. Note that $\|\mathbf{x} - \mathbf{x}^{(6)}\|_\infty = 6.0 \times 10^{-5}$.

k	$\mathbf{x}^{(k)}$		
0	[0.000000	0.000000	0.000000]^T
1	[1.800000	−0.860000	4.253143]^T
2	[0.603669	−3.006157	3.972774]^T
3	[0.971276	−2.998342	3.994011]^T
4	[0.998985	−2.997743	3.999851]^T
5	[0.999546	−2.999851	3.999965]^T
6	[0.999940	−2.999989	3.999992]^T



The selection of the parameter ω is crucial to the performance of the SOR method. The figure above displays the number of iterations required for the SOR method to converge to within a tolerance of 5×10^{-4} as a function of ω. The horizontal lines indicate the number of iterations required by the Gauss-Seidel method (bottom line) and the Jacobi method (top line). We see that there is a range of values, roughly $0.65 \leq \omega \leq 1.0$, for which the performance of the SOR method is as good as or better than the performance of the Gauss-Seidel method. As expected, the SOR method outperforms the Jacobi method over a broader range of ω values.

For the example problem we have been examining in this section, the SOR method performed better than the Gauss-Seidel method primarily for $\omega < 1$. However, for many of the practical problems to which the SOR method is applied (such as the systems of equations associated with the solution of elliptic partial differential equations which we will encounter in Chapter 9), performance is better than the Gauss-Seidel method for $\omega > 1$. When ω is selected greater than 1, the iterative method is referred to as an overrelaxation scheme. Since the Gauss-Seidel method is successive relaxation, this new technique is Successive overrelaxation, or the SOR method for short.

Specific Convergence Properties for the Jacobi, Gauss-Seidel, and SOR Methods

We know that the general iterative method $\mathbf{x}^{(k)} = T\mathbf{x}^{(k)} + \mathbf{c}$ converges if and only if $\rho(T) < 1$. Are there conditions that, when imposed upon the coefficient matrix A, will guarantee that the Jacobi, Gauss-Seidel, and SOR methods converge? The answer to this question is yes; unfortunately, there is no general theory, just a collection of special cases. For example, it is known that strict diagonal dominance

of A is sufficient to guarantee that both the Jacobi method and the Gauss-Seidel method will converge for any choice of the initial vector $\mathbf{x}^{(0)}$. The proof for the Jacobi method is considered in Exercise 15, while the proof for the Gauss-Seidel method can be found in Ortega [2].

The following results regarding the Gauss-Seidel method are also useful in practice.

Theorem.

1. If A is real and symmetric with all positive diagonal elements, then the Gauss-Seidel method converges if and only if A is positive definite.
2. If A is positive definite, then the Gauss-Seidel method will converge for any choice of the initial vector $\mathbf{x}^{(0)}$.

A proof of the first part of this theorem can be found in Isaacson and Keller [1]. Consult Ortega [2] or Ralston and Rabinowitz [3] for a proof of the second part. For the SOR method, the most important convergence results are as follows:

Theorem.

1. If A has all nonzero diagonal elements, then $\rho(T_{sor}) \geq |\omega - 1|$. Therefore, the SOR method can converge only if $0 < \omega < 2$.
2. If A is positive definite and $0 < \omega < 2$, then the SOR method will converge for any choice of the initial vector $\mathbf{x}^{(0)}$.

See Ortega [2], Young [4], or Golub and Ortega [5] for details.

What about the speed of convergence? For the sample problem treated in this section, it was found that the Gauss-Seidel method converged in fewer iterations than the Jacobi method. Will this relative performance hold in general? The answer to this question is no. There are, in fact, coefficient matrices for which the Jacobi method will converge, but the Gauss-Seidel method will not—see Exercise 13. Furthermore, there is no general theory to indicate which method, Jacobi or Gauss-Seidel, will perform best on an arbitrary problem, just a collection of special cases. For example (see Ralston and Rabinowitz [3] or Young [4] for a proof).

Theorem. Suppose A is an $n \times n$ matrix. If $a_{i,i} > 0$ for each i and $a_{i,j} \leq 0$ whenever $i \neq j$, then one and only one of the following statements holds:

1. $0 \leq \rho(T_{gs}) = \rho(T_{jac}) < 1$;
2. $1 < \rho(T_{jac}) < \rho(T_{gs})$;
3. $\rho(T_{jac}) = \rho(T_{gs}) = 0$;
4. $\rho(T_{jac}) = \rho(T_{gs}) = 1$.

Thus, under the hypotheses of this theorem, when one method converges, so will the other method, with the Gauss-Seidel method converging faster. On the other hand, when one method diverges, so will the other, with the Gauss-Seidel method diverging faster.

The final issue we will address is that of the choice of the relaxation parameter for the SOR method. In practice, one of the most important special cases is the following.

Theorem. If A is positive definite and tridiagonal, then $\rho(T_{gs}) = [\rho(T_{jac})]^2 < 1$, and the optimal choice of the relaxation parameter, ω, for the SOR method is

$$\omega = \frac{2}{1 + \sqrt{1 - [\rho(T_{jac})]^2}}.$$

With this choice of ω, $\rho(T_{sor}) = \omega - 1$.

Ortega [2] provides a proof of this result. There are more general conditions under which the formula for the optimal value of ω given in the above theorem holds, but these require more advanced matrix theory concepts than we have developed here. For details consult Stoer and Bulirsch [6], Young [4], or Varga [7]. The optimal choice for ω for an arbitrary linear system remains an open question. Methods for computing the optimal value of ω during the iterative process are discussed by Hageman and Young [8].

References

1. E. Isaacson and H. B. Keller, *Analysis of Numerical Methods*, John Wiley and Sons, New York, 1966.
2. J. M. Ortega, *Numerical Analysis: A Second Course*, Academic Press, New York, 1972.
3. A. Ralston and P. Rabinowitz, *A First Course in Numerical Analysis*, 2nd edition, McGraw-Hill, New York, 1978.
4. D. M. Young, *Iterative Solution of Large Linear Systems*, Academic Press, New York, 1971.
5. G. H. Golub and J. M. Ortega, *Scientific Computing and Differential Equations: An Introduction to Numerical Methods*, Academic Press, Boston, 1992.
6. J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, Springer-Verlag, New York, 1980.
7. R. S. Varga, *Matrix Iterative Analysis*, Prentice Hall, Englewood Cliffs, 1962.
8. L. A. Hageman and D. M. Young, *Applied Iterative Methods*, Academic Press, New York, 1981.

EXERCISES

In Exercises 1–4,

(a) Compute T_{jac} and T_{gs} for the given matrix.
(b) Determine the spectral radius of each iteration matrix from part (a).
(c) Will the Jacobi method converge for any choice of initial vector $\mathbf{x}^{(0)}$? Will the Gauss-Seidel method converge for any choice of initial vector $\mathbf{x}^{(0)}$? Explain.