Iterative Solutions of Linear Systems

Consistent linear systems in real life are solved in one of two ways: by direct calculation (using a matrix factorization, for example) or by an iterative procedure that generates a sequence of vectors that approach the exact solution. When the coefficient matrix is large and *sparse* (with a high proportion of zero entries), iterative algorithms can be more rapid than direct methods and can require less computer memory. Also, an iterative process may be stopped as soon as an approximate solution is sufficiently accurate for practical work. However, iterative methods can also fail or be extremely slow for some problems.

The simple iterative methods below were discovered long ago, but they provided the foundation for what today is an active area of research at the interface of mathematics and computer science.

General Framework for an Iterative Solution of $A\mathbf{x} = \mathbf{b}$

Throughout the section, A is an invertible matrix. The goal of an iterative algorithm is to produce a sequence of vectors,

$$\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(k)}, \ldots$$

that converges to the unique solution of $A\mathbf{x} = \mathbf{b}$, say $\mathbf{x}^*$, in the sense that the entries in $\mathbf{x}^{(k)}$ are as close as desired to the corresponding entries in $\mathbf{x}^*$ for all k sufficiently large.

To describe a recursion algorithm that produces $\mathbf{x}^{(k+1)}$ from $\mathbf{x}^{(k)}$, we write $A = M - N$ for suitable matrices M and N, and then we rewrite the equation $A\mathbf{x} = \mathbf{b}$ as $M\mathbf{x} - N\mathbf{x} = \mathbf{b}$ and

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

If a sequence $\{\mathbf{x}^{(k)}\}$ satisfies

$$\boxed{M\mathbf{x}^{(k+1)} = N\mathbf{x}^{(k)} + \mathbf{b} \qquad (k = 0, 1, \ldots)} \tag{1}$$

and if the sequence converges to some vector $\mathbf{x}^*$, then it can be shown that $A\mathbf{x}^* = \mathbf{b}$. [The vector on the left in (1) approaches $M\mathbf{x}^*$, while the vector on the right in (1) approaches $N\mathbf{x}^* + \mathbf{b}$. This implies that $M\mathbf{x}^* = N\mathbf{x}^* + \mathbf{b}$ and $A\mathbf{x}^* = \mathbf{b}$.]

For $\mathbf{x}^{(k+1)}$ to be uniquely specified in (1), M must be invertible. Also, M should be chosen so that $\mathbf{x}^{(k+1)}$ is easy to calculate. The iterative methods below illustrate two simple choices for M.

Jacobi's Method

This method assumes that the diagonal entries of A are all nonzero. Let D be the diagonal matrix formed from the diagonal entries of A. Jacobi's method uses D for M and $D - A$ for N in (1), so that

$$D\mathbf{x}^{(k+1)} = (D - A)\mathbf{x}^{(k)} + \mathbf{b} \qquad (k = 0, 1, \ldots)$$

In a real-life problem, available information may suggest a value for $\mathbf{x}^{(0)}$. For simplicity, we take the zero vector as $\mathbf{x}^{(0)}$.

EXAMPLE 1 Apply Jacobi's method to the system

$$\begin{aligned} 10x_1 + x_2 - x_3 &= 18 \\ x_1 + 15x_2 + x_3 &= -12 \\ -x_1 + x_2 + 20x_3 &= 17 \end{aligned} \tag{2}$$

Take $\mathbf{x}^{(0)} = (0, 0, 0)$ as an initial approximation to the solution, and use six iterations (that is, compute $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(6)}$).

Solution For some k, denote the entries in $\mathbf{x}^{(k)}$ by (x_1, x_2, x_3) and the entries in $\mathbf{x}^{(k+1)}$ by (y_1, y_2, y_3). The recursion

$$\begin{bmatrix} 10 & 0 & 0 \\ 0 & 15 & 0 \\ 0 & 0 & 20 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 1 \\ -1 & 0 & -1 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 18 \\ -12 \\ 17 \end{bmatrix}$$

$$\uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow$$

$$D \quad \mathbf{x}^{(k+1)} \quad (D - A) \quad x^{(k)} \quad \mathbf{b}$$

can be written as

$$\begin{aligned} 10y_1 &= -x_2 + x_3 + 18 \\ 15y_2 &= -x_1 - x_3 - 12 \\ 20y_3 &= x_1 - x_2 + 17 \end{aligned}$$

and

$$\begin{aligned} y_1 &= (-x_2 + x_3 + 18)/10 \\ y_2 &= (-x_1 - x_3 - 12)/15 \\ y_3 &= (x_1 - x_2 + 17)/20 \end{aligned} \tag{3}$$

A faster way to get (3) is to solve the first equation in (2) for x_1, the second equation for x_2, the third for x_3, and then rename the variables on the left as y_1, y_2, and y_3, respectively.

For $k = 0$, take $\mathbf{x}^{(0)} = (x_1, x_2, x_3) = (0, 0, 0)$, and compute

$$\mathbf{x}^{(1)} = (y_1, y_2, y_3) = (18/10, -12/15, 17/20) = (1.8, -.8, .85)$$

For $k = 1$, use the entries in $\mathbf{x}^{(1)}$ as x_1, x_2, x_3 in (3) and compute the new y_1, y_2, y_3:

$$\begin{aligned} y_1 &= [-(-.8) + (.85) + 18]/10 = 1.965 \\ y_2 &= [-(1.8) - (.85) - 12]/15 = -.9767 \\ y_3 &= [(1.8) - (-.8) + 17]/20 = .98 \end{aligned}$$

Thus $\mathbf{x}^{(2)} = (1.965, -.9767, .98)$. The entries in $\mathbf{x}^{(2)}$ are used on the right in (3) to compute the entries in $\mathbf{x}^{(3)}$, and so on. Here are the results, with calculations using MATLAB and results reported to four decimal places:

$\mathbf{x}^{(0)}$	$\mathbf{x}^{(1)}$	$\mathbf{x}^{(2)}$	$\mathbf{x}^{(3)}$	$\mathbf{x}^{(4)}$	$\mathbf{x}^{(5)}$	$\mathbf{x}^{(6)}$
$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 1.8 \\ -.8 \\ .85 \end{bmatrix}$	$\begin{bmatrix} 1.965 \\ -.9767 \\ .98 \end{bmatrix}$	$\begin{bmatrix} 1.9957 \\ -.9963 \\ .9971 \end{bmatrix}$	$\begin{bmatrix} 1.9993 \\ -.9995 \\ .9996 \end{bmatrix}$	$\begin{bmatrix} 1.9999 \\ -.9999 \\ .9999 \end{bmatrix}$	$\begin{bmatrix} 2.0000 \\ -1.0000 \\ 1.0000 \end{bmatrix}$

If we decide to stop when the entries in $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(k-1)}$ differ by less than .001, then we need five iterations ($k = 5$).

The Gauss–Seidel Method

This method uses the recursion (1) with M the *lower triangular part* of A. That is, M has the same entries as A on the diagonal and below, and M has zeros above the diagonal. See Fig. 1. As in Jacobi's method, the diagonal entries of A must be nonzero in order for M to be invertible.

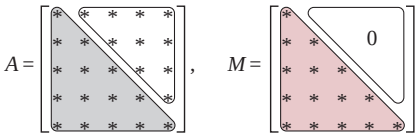

FIGURE 1 The lower triangular part of A.

EXAMPLE 2 Apply the Gauss–Seidel method to the system in Example 1 with $\mathbf{x}^{(0)} = \mathbf{0}$ and six iterations.

$$\begin{aligned} 10x_1 + x_2 - x_3 &= 18 \\ x_1 + 15x_2 + x_3 &= -12 \\ -x_1 + x_2 + 20x_3 &= 17 \end{aligned} \tag{4}$$

Solution With M the lower triangular part of A and $N = M - A$, the recursion relation is

$$\underset{M}{\begin{bmatrix} 10 & 0 & 0 \\ 1 & 15 & 0 \\ -1 & 1 & 20 \end{bmatrix}} \underset{\mathbf{x}^{(k+1)}}{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}} = \underset{(M-A)}{\begin{bmatrix} 0 & -1 & 1 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}} \underset{x^{(k)}}{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}} + \underset{\mathbf{b}}{\begin{bmatrix} 18 \\ -12 \\ 17 \end{bmatrix}}$$

or

$$\begin{aligned} 10y_1 &= -x_2 + x_3 + 18 \\ y_1 + 15y_2 &= -x_3 - 12 \\ -y_1 + y_2 + 20y_3 &= 17 \end{aligned}$$

We will work with one equation at a time. When we reach the second equation, y_1 is already known, so we can move it to the right side. Likewise, in the third equation y_1 and y_2 are known, so we move them to the right. Dividing by the coefficients of the terms

remaining on the left, we obtain

$$\begin{aligned} y_1 &= (-x_2 + x_3 + 18)/10 \\ y_2 &= (-y_1 - x_3 - 12)/15 \\ y_3 &= (y_1 - y_2 + 17)/20 \end{aligned} \tag{5}$$

Another way to view (5) is to solve each equation in (4) for x_1, x_2, x_3, respectively, and regard the highlighted x's as the *new* values:

$$\begin{aligned} x_1 &= (-x_2 + x_3 + 18)/10 \\ x_2 &= (-x_1 - x_3 - 12)/15 \\ x_3 &= (x_1 - x_2 + 17)/20 \end{aligned} \tag{6}$$

Use the first equation to calculate the *new* x_1 [called y_1 in (5)] from x_2 and x_3. Then use this *new* x_1 along with x_3 in the second equation to compute the *new* x_2. Finally, in the third equation, use the *new values* for x_1 and x_2 to compute x_3. In this way, the latest information about the variables is used to compute new values. [A computer program would use statements corresponding to the equations in (6).]

From $\mathbf{x}^{(0)} = (0, 0, 0)$, we obtain

$$\begin{aligned} x_1 &= [-(0) + (0) + 18]/10 = 1.8 \\ x_2 &= [-(1.8) - (0) - 12]/15 = -.92 \\ x_3 &= [+(1.8) - (-.92) + 17]/20 = .986 \end{aligned}$$

Thus $\mathbf{x}^{(1)} = (1.8, -.92, .986)$. The entries in $\mathbf{x}^{(1)}$ are used in (6) to produce $\mathbf{x}^{(2)}$, and so on. Here are the MATLAB calculations reported to four decimal places:

$$\begin{array}{ccccccc} \mathbf{x}^{(0)} & \mathbf{x}^{(1)} & \mathbf{x}^{(2)} & \mathbf{x}^{(3)} & \mathbf{x}^{(4)} & \mathbf{x}^{(5)} & \mathbf{x}^{(6)} \\ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} & \begin{bmatrix} 1.8 \\ -.92 \\ .986 \end{bmatrix} & \begin{bmatrix} 1.9906 \\ -.9984 \\ .9995 \end{bmatrix} & \begin{bmatrix} 1.9998 \\ -.9999 \\ 1.0000 \end{bmatrix} & \begin{bmatrix} 2.0000 \\ -1.0000 \\ 1.0000 \end{bmatrix} & \begin{bmatrix} 2.0000 \\ -1.0000 \\ 1.0000 \end{bmatrix} & \begin{bmatrix} 2.0000 \\ -1.0000 \\ 1.0000 \end{bmatrix} \end{array}$$

Observe that when k is 4, the entries in $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(k-1)}$ differ by less than .001. The values in $\mathbf{x}^{(6)}$ in this case happen to be accurate to eight decimal places.

There exist examples where Jacobi's method is faster than the Gauss–Seidel method, but usually a Gauss–Seidel sequence converges faster, as in Example 2. (If parallel processing is available, Jacobi might be faster because the entries in $\mathbf{x}^{(k)}$ can be computed simultaneously.) There are also examples where one or both methods fail to produce a convergent sequence, and other examples where a sequence is convergent, but converges too slowly for practical use.

Fortunately, there is a simple condition that guarantees (but is not essential for) the convergence of both Jacobi and Gauss–Seidel sequences. This condition is often satisfied, for instance, in large-scale systems that can occur during numerical solutions of partial differential equations (such as Laplace's equation for steady-state heat flow).

An $n \times n$ matrix A is said to be **strictly diagonally dominant** if the absolute value of each diagonal entry exceeds the *sum* of the absolute values of the other entries in the same row. In this case it can be shown that A is invertible and that both the Jacobi and Gauss–Seidel sequences converge to the unique solution of $A\mathbf{x} = \mathbf{b}$, for any initial $\mathbf{x}^{(0)}$. (The speed of the convergence depends on how much the diagonal entries dominate the corresponding row sums.)

The coefficient matrix in Examples 1 and 2 is strictly diagonally dominant, but the following matrix is not. Examine each row:

$$\begin{bmatrix} -6 & 2 & -3 \\ 1 & 4 & -2 \\ 3 & -5 & 8 \end{bmatrix} \qquad \begin{aligned} |-6| &> |2| + |-3| \\ |4| &> |1| + |-2| \\ |8| &= |3| + |-5| \end{aligned}$$

The problem lies in the third row, because $|8|$ is not *larger* than the sum of the magnitudes of the other entries. The practice problem below suggests a trick that sometimes works when a system is not strictly diagonally dominant.

Practice Problem

Show that the Gauss–Seidel method *will* produce a sequence converging to the solution of the following system, provided the equations are arranged properly:

$$\begin{aligned} x_1 - 3x_2 + x_3 &= -2 \\ -6x_1 + 4x_2 + 11x_3 &= 1 \\ 5x_1 - 2x_2 - 2x_3 &= 9 \end{aligned}$$

2.6 Exercises

Solve the systems in Exercises 1–4 using Jacobi's method, with $\mathbf{x}^{(0)} = \mathbf{0}$ and three iterations. [**M**] Repeat the iterations until two successive approximations agree within a tolerance of .001 in each entry.

1. $\begin{aligned} 4x_1 + x_2 &= 7 \\ -x_1 + 5x_2 &= -7 \end{aligned}$

2. $\begin{aligned} 10x_1 - x_2 &= 25 \\ x_1 + 8x_2 &= 43 \end{aligned}$

3. $\begin{aligned} 3x_1 + x_2 &= 11 \\ -x_1 - 5x_2 + 2x_3 &= 15 \\ 3x_2 + 7x_3 &= 17 \end{aligned}$

4. $\begin{aligned} 50x_1 - x_2 &= 149 \\ x_1 - 100x_2 + 2x_3 &= -101 \\ 2x_2 + 50x_3 &= -98 \end{aligned}$

In Exercises 5–8, use the Gauss–Seidel method, with $\mathbf{x}^{(0)} = \mathbf{0}$ and two iterations. [**M**] Compare the number of iterations needed by Gauss–Seidel and Jacobi to make two successive approximations agree within a tolerance of .001.

5. The system in Exercise 1.

6. The system in Exercise 2.

7. The system in Exercise 3.

8. The system in Exercise 4.

Determine which of the matrices in Exercises 9 and 10 are strictly diagonally dominant.

9. a. $\begin{bmatrix} 5 & 4 \\ 4 & 3 \end{bmatrix}$ b. $\begin{bmatrix} 9 & -5 & 2 \\ 5 & -8 & -1 \\ -2 & 1 & 4 \end{bmatrix}$

10. a. $\begin{bmatrix} 3 & -2 \\ 2 & 3 \end{bmatrix}$ b. $\begin{bmatrix} 5 & 3 & 1 \\ 3 & 6 & -4 \\ 1 & -4 & 7 \end{bmatrix}$

Neither iteration method of this section works for the systems as written in Exercises 11 and 12. Compute the first three iterates for Gauss–Seidel, with $\mathbf{x}^{(0)} = \mathbf{0}$. Then rearrange the equations to make Gauss–Seidel work, and compute two iterations. [**M**] Find an approximation accurate to three decimal places.

11. $\begin{aligned} 2x_1 + 6x_2 &= 4 \\ 5x_1 - x_2 &= 6 \end{aligned}$

12. $\begin{aligned} -x_1 + 4x_2 - x_3 &= 3 \\ 4x_1 - x_2 &= 10 \\ -x_2 + 4x_3 &= 6 \end{aligned}$

The systems in Exercises 13 and 14 are not strictly diagonally dominant, and no rearrangement of the equations will make them so. Nevertheless, both Jacobi and Gauss–Seidel produce convergent sequences. [**M**] Find how many Gauss–Seidel iterations are needed to produce an approximation accurate to three decimal places. Set $\mathbf{x}^{(0)} = \mathbf{0}$.

13.
$$\begin{aligned} 4x_1 - 2x_2 &= 10 \\ -5x_1 + 4x_2 &= -2 \end{aligned}$$

14.
$$\begin{aligned} 5x_1 - 3x_2 + 2x_3 &= -19 \\ -x_1 + 5x_2 + 3x_3 &= 10 \\ x_1 - 4x_2 - 4x_3 &= 18 \end{aligned}$$

[M] In Exercises 15 and 16, try both Jacobi's method and the Gauss–Seidel method and describe what happens. Set $\mathbf{x}^{(0)} = \mathbf{0}$. If a sequence seems to converge, try to find two successive approximations that agree within a tolerance of .001.

15.
$$\begin{aligned} x_1 \quad + \quad x_3 &= 6 \\ x_1 - \ x_2 \quad &= 3 \\ x_1 + 2x_2 - 3x_3 &= 9 \end{aligned}$$

16.
$$\begin{aligned} 3x_1 - 2x_2 + 2x_3 &= 10 \\ -2x_1 + 3x_2 - 2x_3 &= -7 \\ 2x_1 - 2x_2 + 3x_3 &= 4 \end{aligned}$$

17. The main focus of Section 2.7 will be the equation $\mathbf{x} = C\mathbf{x} + \mathbf{d}$. Under suitable hypotheses on C, this equation can be solved by an iterative scheme using the recursion relation

$$\mathbf{x}^{(k+1)} = C\mathbf{x}^{(k)} + \mathbf{d} \qquad (7)$$

a. Show that (7) is a special case of (1). What are M, N, and A?

b. Take $\mathbf{x}^{(0)} = \mathbf{0}$ and use (7) to find a formula for $\mathbf{x}^{(2)}$ not involving $\mathbf{x}^{(1)}$.

c. Verify by induction that if $\mathbf{x}^{(0)} = \mathbf{0}$, then

$$\mathbf{x}^{(k)} = \mathbf{d} + C\mathbf{d} + C^2\mathbf{d} + \cdots + C^{k-1}\mathbf{d} \qquad (k = 1, 2, \ldots)$$

18. Suppose $\mathbf{x}^*$ satisfies $A\mathbf{x} = \mathbf{b}$ and $\mathbf{x}^{(k)}$ is determined by the recursion relation (1), where A is invertible, $A = M - N$, and M is invertible. The vector $\mathbf{e}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}^*$ is the error in the approximation of $\mathbf{x}^*$ by $\mathbf{x}^{(k)}$. The steps below imply that if $(M^{-1}N)^k \to 0$ as $k \to \infty$, then $\mathbf{e}^{(k)} \to \mathbf{0}$, which means that the sequence $\{\mathbf{x}^{(k)}\}$ converges to the solution $\mathbf{x}^*$.

a. Show that $\mathbf{e}^{(k+1)} = (M^{-1}N)\mathbf{e}^{(k)}$.

b. Prove by induction that $\mathbf{e}^{(k)} = (M^{-1}N)^k\mathbf{e}^{(0)}$.

Solution to Practice Problem

The system is not strictly diagonally dominant, so neither Jacobi nor Gauss–Seidel is guaranteed to work. In fact, both iterative methods produce sequences that fail to converge, even though the system has the unique solution $x_1 = 3$, $x_2 = 2$, $x_3 = 1$. However, the equations can be rearranged as

$$\begin{aligned} 5x_1 - 2x_2 - \ 2x_3 &= \ 9 \\ x_1 - 3x_2 + \ x_3 &= -2 \\ -6x_1 + 4x_2 + 11x_3 &= \ 1 \end{aligned}$$

Now the coefficient matrix is strictly diagonally dominant, so we know Gauss–Seidel works with any initial vector. In fact, if $\mathbf{x}^{(0)} = \mathbf{0}$, then $\mathbf{x}^{(8)} = (2.9987, 1.9992, .9996)$.