Iteration of Linear Systems

Iteration, or repeated application of a function, appears in a surprisingly wide range of applications. Discrete dynamical systems, in which the continuous time variable has been “quantized” in individual units (seconds, days, years, etc.) are modeled by iterative systems. Most numerical solution methods, for both linear and nonlinear systems, are based on an iterative procedure. Starting with an initial guess, the successive iterates lead to closer and closer approximations to the true solution. For linear systems of equations, iterative solution methods can be used as an attractive alternative to Gaussian elimination, and are particularly effective for solving the very large, sparse systems arising in the numerical solution to both ordinary and partial differential equations. In probability theory, population dynamics and other applications, iterative models known as Markov processes govern basic probabilistic processes. All practical methods for computing eigenvalues and eigenvectors are based on a form of iteration.

In this chapter, we concentrate on the iteration of linear systems. As always, proper understanding of the linear situation is an essential prerequisite for tackling the more challenging nonlinear systems, which will be deferred until Chapter 19. Linear iteration coincides with multiplication by successive powers of a matrix. The convergence of the iterates depends on the magnitude of the eigenvalues of the coefficient matrix. The largest eigenvalue (in modulus) is known as the “spectral radius” of the matrix, and convergence requires a spectral radius smaller than one. While accurate computation of the eigenvalues is not an easy task, the simple but effective Gerschgorin Circle Theorem yields useful estimates, that can, in favorable situations, readily ensure convergence. Matrix norms are another practical alternative, since iterative methods with coefficient matrices of norm less than one are guaranteed to converge.

We will then turn our attention to the three most important iterative schemes used to accurately approximate the solutions to linear systems of algebraic equations. The classical Jacobi method is the simplest, while an evident modification leads to the popular Gauss–Seidel method. Completely general conditions ensuring convergence of these schemes to the solution of the original system are hard to formulate, although convergence is assured for the important class of diagonally dominant matrices that arise in many applications. A simple modification of the Gauss–Seidel scheme known as Successive Over-Relaxation (SOR) can dramatically speed up the convergence rate, and is the method of choice in many modern applications.

In the final section we discuss the computation of eigenvalues and eigenvectors of matrices. Needless to say, we completely avoid trying to solve (or even write down) the characteristic polynomial equation. The simple power method and its variants, all based
on linear iteration, provide an effective means of approximating selected eigenvalues. For constructing a complete system of eigenvalues and eigenvectors, the remarkable QR algorithm, which is based of the Gram–Schmidt orthogonalization procedure, is the method of choice, and we shall close with a new proof of its convergence.

10.1. Linear Iterative Systems.

We begin with the basic definition of an iterative system of linear equations.

**Definition 10.1.** A linear iterative system takes the form

\[ u^{(k+1)} = T u^{(k)}, \quad u^{(0)} = \mathbf{a}. \]  \hspace{1cm} (10.1)

The coefficient matrix \( T \) has size \( n \times n \). We will consider both real and complex systems, and so the iterates \( u^{(k)} \) are vectors either in \( \mathbb{R}^n \) (which assumes that the coefficient matrix \( T \) is also real) or in \( \mathbb{C}^n \). A linear iterative system can be viewed as a discretized version of a first order system of linear ordinary differential equations, as in (8.9), in which the state of system, as represented by the vector \( u^{(k)} \), changes at discrete time intervals, labeled by the index \( k \). For \( k = 1, 2, 3, \ldots \), the solution \( u^{(k)} \) is uniquely determined by the initial conditions \( u^{(0)} = \mathbf{a} \).

**Scalar Systems**

As usual, one begins with an analysis of the scalar version. Consider the iterative equation

\[ u^{(k+1)} = \lambda u^{(k)}, \quad u^{(0)} = \mathbf{a}. \]  \hspace{1cm} (10.2)

The general solution to (10.2) is easily found:

\[ u^{(1)} = \lambda u^{(0)} = \lambda \mathbf{a}, \quad u^{(2)} = \lambda u^{(1)} = \lambda^2 \mathbf{a}, \quad u^{(3)} = \lambda u^{(2)} = \lambda^3 \mathbf{a}, \]

and, in general,

\[ u^{(k)} = \lambda^k \mathbf{a}. \]  \hspace{1cm} (10.3)

If the initial condition is \( a = 0 \), then the solution \( u^{(k)} \equiv 0 \) is constant. Therefore, 0 is a fixed point or equilibrium solution for the iterative system.

**Example 10.2.** Banks add interest to a savings account at discrete time intervals. For example, if the bank offers 5\% interest compounded yearly, this means that the account balance will increase by 5\% each year. Thus, assuming no deposits or withdrawals, the balance \( u^{(k)} \) after \( k \) years will satisfy the iterative equation (10.2) with \( \lambda = 1 + r \) where \( r \) is the interest rate, and the 1 indicates that the money in the account remains there. For example, if your initial deposit is \( u^{(0)} = a = \$1,000 \), after 1 year your account has \( u^{(1)} = \$1,050 \), after 10 years \( u^{(10)} = \$1,628.89 \), after 50 years \( u^{(50)} = \$11,467.40 \), and after 200 years \( u^{(200)} = \$17,292,580.82 \).

When the compounding is done monthly, the interest rate is still quoted on a yearly basis, and so you receive \( \frac{1}{12} \) of the interest compounded each month. If \( \tilde{u}^{(k)} \) denotes the balance after \( k \) months, then, after \( n \) years, the account balance is \( \tilde{u}^{(12n)} = (1 + \frac{1}{12} r)^{12n} a \). Thus, when the interest rate of 5\% is compounded monthly, your account balance is
Figure 10.1. One Dimensional Real Linear Iterative Systems.

\[ \hat{u}^{(12)} = 1,051.16 \text{ after 1 year, } \hat{u}^{(120)} = 1,647.01 \text{ after 10 years, } \hat{u}^{(600)} = 12,119.38 \text{ after 50 years, and } \hat{u}^{(2400)} = 21,573,572.66 \text{ dollars after 200 years.} \]

So, if you wait sufficiently long, compounding has a dramatic effect. Daily compounding replaces 12 by 365.25, the number of days in a year.

Let us analyze the solutions of iterative equations when \( \lambda \in \mathbb{R} \) is a real constant. Apart from the equilibrium solution, the iterates exhibit five qualitatively different behaviors, depending on the size of the coefficient \( \lambda \).

(a) If \( \lambda = 0 \), the solution immediately becomes zero, and stays there, so \( u^{(k)} = 0 \) for all \( k \geq 1 \).

(b) If \( 0 < \lambda < 1 \), then the solution is of one sign, and tends monotonically to zero, so \( u^{(k)} \to 0 \) as \( k \to \infty \).

(c) If \( -1 < \lambda < 0 \), then the solution tends to zero, \( u^{(k)} \to 0 \) as \( k \to \infty \). Successive iterates have alternating signs.

(d) If \( \lambda = 1 \), the solution is constant, \( u^{(k)} = a \), for all \( k \geq 0 \).

(e) If \( \lambda = -1 \), the solution switches back and forth between two values; \( u^{(k)} = (-1)^k a \).

(f) If \( 1 < \lambda < \infty \), then the iterates \( u^{(k)} \) become unbounded. If \( a > 0 \), they go monotonically to \( +\infty \); if \( a < 0 \), to \( -\infty \).

(g) If \( -\infty < \lambda < -1 \), then the iterates \( u^{(k)} \) also become unbounded. Successive iterates have alternating signs.

In Figure 10.1 we exhibit representative scatter plots for the nontrivial cases \( (b - g) \). The horizontal axis is the index \( k \) and the vertical axis the solution value \( u \).

To describe the different scenarios, we adopt a terminology that already appeared in the continuous realm. In the first three cases, the fixed point \( u = 0 \) is said to be \emph{globally asymptotically stable} since all solutions tend to 0 as \( k \to \infty \). In cases \( (d) \) and \( (e) \), the zero solution is \emph{stable}, since solutions with nearby initial data, \( |a| \ll 1 \), remain nearby. In the final two cases, the zero solution is \emph{unstable}; any nonzero initial data \( a \neq 0 \) — no
matter how small — will give rise to a solution that eventually goes arbitrarily far away from equilibrium.

Let us next consider the case of a complex scalar iterative system. The coefficient $\lambda$ and the initial data $a$ in (10.2) are allowed to be complex numbers. The solution is the same, (10.3), but now we need to know what happens when we raise a complex number $\lambda$ to a high power. The secret is to write $\lambda = r e^{i\theta}$ in polar form (3.79), where $r = |\lambda|$ is its modulus and $\theta = \text{ph} \lambda$ its angle or phase. Then $\lambda^k = r^k e^{ik\theta}$. Since $|e^{ik\theta}| = 1$, we have $|\lambda^k| = |\lambda|^k$, and so the solutions (10.3) have modulus $|u^{(k)}| = |\lambda^k a| = |\lambda|^k |a|$. As a result, $u^{(k)}$ will remain bounded if and only if $|\lambda| \leq 1$, and will tend to zero as $k \to \infty$ if and only if $|\lambda| < 1$.

We have thus established the basic stability criteria for scalar, linear systems.

**Theorem 10.3.** The zero solution to a (real or complex) scalar iterative system

$$u^{(k+1)} = \lambda u^{(k)}$$

is

(a) asymptotically stable if and only if $|\lambda| < 1$,
(b) stable if and only if $|\lambda| \leq 1$,
(c) unstable if and only if $|\lambda| > 1$.

**Powers of Matrices**

The solution to the general linear matrix iterative system

$$u^{(k+1)} = T u^{(k)}, \quad u^{(0)} = a,$$

is also, at least at first glance, immediate. Clearly,

$$u^{(1)} = T u^{(0)} = T a, \quad u^{(2)} = T u^{(1)} = T^2 a, \quad u^{(3)} = T u^{(2)} = T^3 a,$$

and, in general,

$$u^{(k)} = T^k a.$$  \hspace{1cm} (10.5)

Thus, the iterates are simply determined by multiplying the initial vector $a$ by the successive powers of the coefficient matrix $T$.

However, unlike real or complex scalars, the general formulae and qualitative behavior of the powers of a square matrix are not nearly so immediately apparent. (Before continuing, the reader is urged to experiment with simple $2 \times 2$ matrices, and try to detect patterns.) To resolve this dilemma, recall that we managed to solve linear systems of differential equations by suitably adapting the known exponential solution from the scalar version. In the discrete case, we no longer have exponentials, but rather powers, in our scalar solution formula (10.3). This motivates us to try the power ansatz

$$u^{(k)} = \lambda^k v,$$  \hspace{1cm} (10.6)

where $\lambda$ is a scalar and $v$ is a fixed vector, as a possible solution. We find

$$u^{(k+1)} = \lambda^{k+1} v, \quad \text{while} \quad T u^{(k)} = T (\lambda^k v) = \lambda^k T v.$$
These two expressions will be equal if and only if
\[ T v = \lambda v. \]
Therefore, (10.6) is a nontrivial solution to (10.4) if and only if \( \lambda \) is an eigenvalue of \( T \) and \( v \) an associated eigenvector.

Thus, to each eigenvector and eigenvalue of the coefficient matrix, we can construct a solution to the iterative system. We can then use linear superposition, as in Theorem 7.29, to combine the basic power solutions to form more general solutions. In particular, if the coefficient matrix is complete, then this method will, as in the case of linear ordinary differential equations, produce the general solution.

**Theorem 10.4.** If the coefficient matrix \( T \) is complete, then the general solution to the linear iterative system \( u^{(k+1)} = T u^{(k)} \) is given by
\[ u^{(k)} = c_1 \lambda_1^k v_1 + c_2 \lambda_2^k v_2 + \cdots + c_n \lambda_n^k v_n, \quad (10.7) \]
where \( v_1, \ldots, v_n \) are the linearly independent eigenvectors and \( \lambda_1, \ldots, \lambda_n \) the corresponding eigenvalues of \( T \). The coefficients \( c_1, \ldots, c_n \) are arbitrary scalars, and are uniquely prescribed by the initial conditions \( u^{(0)} = a \).

**Proof:** Since we already know that (10.7) is a solution to the system for arbitrary \( c_1, \ldots, c_n \), it suffices to show that we can match any prescribed initial conditions. We need to solve the linear system
\[ u^{(0)} = c_1 v_1 + \cdots + c_n v_n = a. \quad (10.8) \]
Completeness of \( T \) implies that its eigenvectors form a basis of \( \mathbb{C}^n \), and hence (10.8) always admits a solution. In matrix form, we can rewrite (10.8) as
\[ S c = a, \quad \text{so that} \quad c = S^{-1} a, \]
where \( S = (v_1 \ v_2 \ldots \ v_n) \) is the (nonsingular) matrix whose columns are the eigenvectors. \( Q.E.D. \)

**Remark:** Incomplete cases rely on the Jordan canonical form of Section 8.6; see Exercise \( \blacklozenge \) for details.

**Example 10.5.** Consider the iterative system
\[ x^{(k+1)} = \frac{3}{10} x^{(k)} + \frac{1}{10} y^{(k)}, \quad y^{(k+1)} = \frac{1}{10} x^{(k)} + \frac{3}{10} y^{(k)}, \quad (10.9) \]
with initial conditions
\[ x^{(0)} = a, \quad y^{(0)} = b. \quad (10.10) \]
The system can be rewritten in our matrix form (10.4) with
\[ T = \begin{pmatrix} .3 & .1 \\ .1 & .3 \end{pmatrix}, \quad u^{(k)} = \begin{pmatrix} x^{(k)} \\ y^{(k)} \end{pmatrix}, \quad a = \begin{pmatrix} a \\ b \end{pmatrix}. \]
Solving the characteristic equation 
\[ \det(T - \lambda I) = \lambda^2 - .6\lambda - .08 = 0 \]
produces the eigenvalues \( \lambda_1 = .4, \lambda_2 = .2 \). We then solve the associated linear systems 
\( (T - \lambda_j I)v_j = 0 \) for the corresponding eigenvectors:

\[ \begin{align*}
\lambda_1 = .4, & \quad v_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \\
\lambda_2 = .2, & \quad v_2 = \begin{pmatrix} -1 \\ 1 \end{pmatrix}.
\end{align*} \]

Therefore, the basic power solutions are
\[ \begin{align*}
u_1^{(k)} &= (.4)^k \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \\
u_2^{(k)} &= (.2)^k \begin{pmatrix} -1 \\ 1 \end{pmatrix}.
\end{align*} \]

Theorem 10.4 tells us that the general solution is given as a linear combination,
\[ u^{(k)} = c_1 u_1^{(k)} + c_2 u_2^{(k)} = c_1 (.4)^k \begin{pmatrix} 1 \\ 1 \end{pmatrix} + c_2 (.2)^k \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} c_1 (.4)^k - c_2 (.2)^k \\ c_1 (.4)^k + c_2 (.2)^k \end{pmatrix}, \]
where \( c_1, c_2 \) are arbitrary scalars, whose values are determined by the initial conditions:
\[ u^{(0)} = \begin{pmatrix} c_1 - c_2 \\ c_1 + c_2 \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}, \quad \text{and hence} \quad c_1 = \frac{a + b}{2}, \quad c_2 = \frac{b - a}{2}. \]

Therefore, the explicit formula for the solution to (10.9), (10.10) is
\[ \begin{align*}
x^{(k)} &= (.4)^k \frac{a + b}{2} - (.2)^k \frac{b - a}{2}, \\
y^{(k)} &= (.4)^k \frac{a + b}{2} + (.2)^k \frac{b - a}{2}.
\end{align*} \]

In particular, as \( k \to \infty \), the iterates \( u^{(k)} \to 0 \) converge to zero at a rate governed by the larger eigenvalue \( \lambda_1 = .4 \). Thus, (10.9) defines a stable iterative system.

**Example 10.6.** The **Fibonacci numbers** are defined by the second order iterative scheme
\[ u^{(k+2)} = u^{(k+1)} + u^{(k)}, \quad (10.11) \]
with initial conditions
\[ u^{(0)} = a, \quad u^{(1)} = b. \quad (10.12) \]

The classical Fibonacci integers follow from \( a = 0, b = 1 \). Thus, to obtain the next Fibonacci number, we add the previous two; the first few Fibonacci integers are
\[ u^{(0)} = 0, \quad u^{(1)} = 1, \quad u^{(2)} = 1, \quad u^{(3)} = 2, \quad u^{(4)} = 3, \quad u^{(5)} = 5, \quad u^{(6)} = 8, \quad u^{(7)} = 13, \quad \ldots. \]

The Fibonacci integers occur in a surprising range of natural objects, including leaves, flowers, and fruit, [11]. They were originally introduced by the Renaissance mathematician Fibonacci (Leonardo of Pisa) as a crude model of the growth of a population of rabbits. In Fibonacci’s model, the \( k^{th} \) Fibonacci number \( u^{(k)} \) measures the total number of pairs of rabbits at year \( k \). We start the process with a single juvenile pair\(^\dagger\) at year 0. Once a

\(^\dagger\) We ignore important details like the sex of the offspring.
year, each pair of rabbits produces a new pair of offspring, but it takes a year for a rabbit pair to mature enough to produce offspring of their own.

Just as every higher order ordinary differential equation can be replaced by an equivalent first order system, so every higher order iterative system can be replaced by a first order iterative system. In this particular case, we define the vector

\[ \mathbf{u}^{(k)} = \begin{pmatrix} u^{(k)} \\ u^{(k+1)} \end{pmatrix} \in \mathbb{R}^2, \]

and note that (10.11) is equivalent to the matrix system

\[ \begin{pmatrix} u^{(k+1)} \\ u^{(k+2)} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} u^{(k)} \\ u^{(k+1)} \end{pmatrix}, \quad \text{or} \quad \mathbf{u}^{(k+1)} = T \mathbf{u}^{(k)}, \quad \text{where} \quad T = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}. \]

To find an explicit formula for the Fibonacci numbers, we need to determine the eigenvalues and eigenvectors of the coefficient matrix \( T \). A straightforward computation produces

\[ \lambda_1 = \frac{1 + \sqrt{5}}{2} = 1.618034 \ldots, \quad \lambda_2 = \frac{1 - \sqrt{5}}{2} = -0.618034 \ldots, \]

\[ \mathbf{v}_1 = \left( \begin{array}{c} \frac{-1 + \sqrt{5}}{2} \\ 1 \end{array} \right), \quad \mathbf{v}_2 = \left( \begin{array}{c} \frac{-1 - \sqrt{5}}{2} \\ 1 \end{array} \right). \]

Therefore, according to (10.7), the general solution to the Fibonacci system is

\[ \mathbf{u}^{(k)} = \begin{pmatrix} u^{(k+1)} \\ u^{(k)} \end{pmatrix} = c_1 \left( \frac{1 + \sqrt{5}}{2} \right)^k \begin{pmatrix} \frac{-1 + \sqrt{5}}{2} \\ 1 \end{pmatrix} + c_2 \left( \frac{1 - \sqrt{5}}{2} \right)^k \begin{pmatrix} \frac{-1 - \sqrt{5}}{2} \\ 1 \end{pmatrix}. \quad (10.13) \]

The initial data

\[ \mathbf{u}^{(0)} = c_1 \begin{pmatrix} \frac{-1 + \sqrt{5}}{2} \\ 1 \end{pmatrix} + c_2 \begin{pmatrix} \frac{-1 - \sqrt{5}}{2} \\ 1 \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix} \]

uniquely specifies the coefficients

\[ c_1 = \frac{2a + (1 + \sqrt{5})b}{2\sqrt{5}}, \quad c_2 = \frac{-2a + (1 - \sqrt{5})b}{2\sqrt{5}}. \]

The first entry of the solution vector (10.13) produces the formula

\[ u^{(k)} = \frac{-1 + \sqrt{5}}{2\sqrt{5}} a + 2b \left( \frac{1 + \sqrt{5}}{2} \right)^k + \frac{1 + \sqrt{5}}{2\sqrt{5}} a - 2b \left( \frac{1 - \sqrt{5}}{2} \right)^k \quad (10.14) \]

for the \( k \)th Fibonacci number. For the particular initial conditions \( a = 0, \ b = 1 \), formula (10.14) reduces to the classical Binet formula

\[ u^{(k)} = \frac{1}{\sqrt{5}} \left[ \left( \frac{1 + \sqrt{5}}{2} \right)^k - \left( \frac{1 - \sqrt{5}}{2} \right)^k \right] \quad (10.15) \]
for the $k^{th}$ Fibonacci integer. It is a remarkable fact that, for every value of $k$, all the $\sqrt{5}$'s cancel out, and the Binet formula does indeed produce the Fibonacci integers tabulated above. Another useful observation is that since

$$0 < |\lambda_2| = \frac{\sqrt{5} - 1}{2} < 1 < \lambda_1 = \frac{1 + \sqrt{5}}{2},$$

the terms involving $\lambda_1^k$ go to $\infty$ (and so the zero solution to this iterative system is unstable) while the terms involving $\lambda_2^k$ go to zero. Therefore, even for $k$ moderately large, the first term in (10.14) is an excellent approximation (and one that gets more and more accurate with increasing $k$) to the $k^{th}$ Fibonacci number.

The dominant eigenvalue $\lambda_1 = \frac{1}{2} (1 + \sqrt{5}) = 1.618034\ldots \equiv \phi$ is known as the golden ratio and plays an important role in spiral growth in nature, as well as in art, architecture and design, [11]. It describes the overall growth rate of the Fibonacci integers, and, in fact, any sequence of Fibonacci numbers with initial conditions $b_6 = \frac{1}{2} - 1 - \frac{1}{2}$.

**Example 10.7.** Let $T = \begin{pmatrix} -3 & 1 & 6 \\ 1 & -1 & -2 \\ -1 & -1 & 0 \end{pmatrix}$ be the coefficient matrix for a three-dimensional iterative system $u^{(k+1)} = T u^{(k)}$. The eigenvalues and corresponding eigenvectors are

$$\lambda_1 = -2, \quad \lambda_2 = -1 + i, \quad \lambda_3 = -1 - i,$$

$$v_1 = \begin{pmatrix} 4 \\ -2 \\ 1 \end{pmatrix}, \quad v_2 = \begin{pmatrix} 2 - i \\ -1 \\ 1 \end{pmatrix}, \quad v_3 = \begin{pmatrix} 2 + i \\ -1 \\ 1 \end{pmatrix}.$$

Therefore, according to (10.7), the general complex solution to the iterative system is

$$u^{(k)} = b_1 (-2)^k \begin{pmatrix} 4 \\ -2 \\ 1 \end{pmatrix} + b_2 (-1 + i)^k \begin{pmatrix} 2 - i \\ -1 \\ 1 \end{pmatrix} + b_3 (-1 - i)^k \begin{pmatrix} 2 + i \\ -1 \\ 1 \end{pmatrix},$$

where $b_1, b_2, b_3$ are arbitrary complex scalars.

If we are only interested in real solutions, we can, as in the case of systems of differential equations, break up any complex solution into its real and imaginary parts, each of which constitutes a real solution. (This is another manifestation of the general Reality Theorem 7.47, but is not hard to prove directly.) We begin by writing $\lambda_2 = -1 + i = \sqrt{2} e^{3 \pi i/4}$, and hence

$$( -1 + i)^k = 2^{k/2} e^{3 k \pi i/4} = 2^{k/2} \left( \cos \frac{3}{4} k \pi + i \sin \frac{3}{4} k \pi \right).$$

Therefore, the complex solution

$$(-1 + i)^k \begin{pmatrix} 2 - i \\ -1 \\ 1 \end{pmatrix} = 2^{k/2} \begin{pmatrix} 2 \cos \frac{3}{4} k \pi \sin \frac{3}{4} k \pi \\ -\cos \frac{3}{4} k \pi \\ \frac{3}{4} k \pi \end{pmatrix} + i 2^{k/2} \begin{pmatrix} 2 \sin \frac{3}{4} k \pi \sin \frac{3}{4} k \pi \\ -\sin \frac{3}{4} k \pi \\ \frac{3}{4} k \pi \end{pmatrix}.$$
is a complex combination of two independent real solutions. The complex conjugate eigenvalue \( \lambda_3 = -1 - i \) leads, as before, to the complex conjugate solution — and the same two real solutions. The general real solution \( u^{(k)} \) to the system can be written as a linear combination of the three independent real solutions:

\[
\begin{pmatrix}
4 \\
-2 \\
1
\end{pmatrix}
\]

\[
c_1 (-2)^k + c_2 2^{k/2} \begin{pmatrix}
2 \cos \frac{3}{4} k \pi + \sin \frac{3}{4} k \pi \\
- \cos \frac{3}{4} k \pi \\
\cos \frac{3}{4} k \pi
\end{pmatrix}
\]

\[
+ c_3 2^{k/2} \begin{pmatrix}
2 \sin \frac{3}{4} k \pi - \cos \frac{3}{4} k \pi \\
- \sin \frac{3}{4} k \pi \\
\sin \frac{3}{4} k \pi
\end{pmatrix},
\]

where \( c_1, c_2, c_3 \) are arbitrary real scalars, uniquely prescribed by the initial conditions.

### 10.2. Stability.

With the solution formula (10.7) in hand, we are now in a position to understand the qualitative behavior of solutions to (complete) linear iterative systems. The most important case for applications is when all the iterates converge to \( 0 \).

**Definition 10.8.** The equilibrium solution \( u^* = 0 \) to a linear iterative system (10.1) is called *asymptotically stable* if and only if all solutions \( u^{(k)} \rightarrow 0 \) as \( k \rightarrow \infty \).

Stability of the solutions to an iterative system relies on the following property of the coefficient matrix.

**Definition 10.9.** A matrix \( T \) is called *convergent* if its powers \( T^k \rightarrow O \) converge to the zero matrix as \( k \rightarrow \infty \).

We note that convergence of a sequence of matrices or vectors is equivalent to convergence of their individual entries. The equivalence of the convergence condition and stability of the iterative system follows immediately from the solution formula (10.5).

**Proposition 10.10.** The linear iterative system \( u^{(k+1)} = T u^{(k)} \) has asymptotically stable zero solution if and only if \( T \) is a convergent matrix.

Indeed, since \( u^{(k)} = T^k a \) and the initial condition \( a \) is arbitrary, the only way that all solutions tend to zero as \( k \rightarrow \infty \) is if the coefficient matrices \( T^k \rightarrow O \).

For the analysis of convergence, we shall adopt a norm \( \| \cdot \| \) on our underlying vector space, \( \mathbb{R}^n \) or \( \mathbb{C}^n \). The reader may be inclined to choose the Euclidean (or Hermitian) norm, but, in practice, the \( L^\infty \) norm

\[
\| u \|_\infty = \max \left\{ | u_1 |, \ldots, | u_n | \right\},
\]

prescribed by the vector’s maximal entry (in modulus) is usually much easier to work with. Convergence of the iterates is equivalent to convergence of their norms:

\( u^{(k)} \rightarrow 0 \) if and only if \( \| u^{(k)} \| \rightarrow 0 \) as \( k \rightarrow \infty \).

(See also Section 12.5 for additional details on convergence in finite-dimensional vector spaces.)

The fundamental stability criterion relies on the magnitude of the eigenvalues of the coefficient matrix.
Theorem 10.11. A linear iterative system (10.1) has asymptotically stable zero solution if and only if all its (complex) eigenvalues have modulus strictly less than one: 

\[ |\lambda_j| < 1. \]

Proof: Let us prove this result assuming that the coefficient matrix \( T \) is complete. (The proof in the incomplete case relies on the Jordan canonical form, and is outlined in the exercises.) If \( \lambda_j \) is an eigenvalue such that \( |\lambda_j| < 1 \), then the corresponding basis solution \( u_j^{(k)} = \lambda_j^k v_j \) tends to zero as \( k \to \infty \); indeed,

\[ \| u_j^{(k)} \| = \| \lambda_j^k v_j \| = |\lambda_j|^k \| v_j \| \to 0 \quad \text{since} \quad |\lambda_j| < 1. \]

Therefore, if all eigenvalues are less than 1 in modulus, all terms in the solution formula (10.7) tend to zero, which proves asymptotic stability: \( u^{(k)} \to 0 \). \( \text{Q.E.D.} \)

Consequently, the necessary and sufficient condition for asymptotic stability of a linear iterative system is that all the eigenvalues of the coefficient matrix lie strictly inside the unit circle in the complex plane: \( |\lambda_j| < 1 \). Let us formalize this key result.

Definition 10.12. The \textit{spectral radius} of a matrix \( T \) is defined as the maximal modulus of all of its real and complex eigenvalues:

\[ \rho(T) = \max \{ |\lambda_1|, \ldots, |\lambda_k| \}. \]

We can restate the Stability Theorem 10.11 as follows.

Theorem 10.13. The matrix \( T \) is convergent if and only if its spectral radius is strictly less than one: \( \rho(T) < 1 \).

If \( T \) is complete, then we can apply the triangle inequality to (10.7) to estimate

\[
\| u^{(k)} \| = \| c_1 \lambda_1^k v_1 + \cdots + c_n \lambda_n^k v_n \|
\leq |\lambda_1|^k \| v_1 \| + \cdots + |\lambda_n|^k \| v_n \|
\leq \rho(T)^k \left( \| v_1 \| + \cdots + |c_n| \| v_n \| \right) = C \rho(T)^k,
\]

for some constant \( C > 0 \) that depends only upon the initial conditions. In particular, if \( \rho(T) < 1 \), then

\[
\| u^{(k)} \| \leq C \rho(T)^k \to 0 \quad \text{as} \quad k \to \infty,
\]

in accordance with Theorem 10.13. Thus, the spectral radius prescribes the rate of convergence of the solutions to equilibrium. The smaller the spectral radius, the faster the solutions converge to 0.

If \( T \) has only one largest (simple) eigenvalue, so \( |\lambda_1| > |\lambda_j| \) for all \( j > 1 \), then the first term in the solution formula (10.7) will eventually dominate all the others: \( \| \lambda_1^k v_1 \| \gg \| \lambda_j^k v_j \| \) for \( j > 1 \) and \( k \gg 0 \) large. Therefore, provided \( c_1 \neq 0 \), the solution (10.7) has the asymptotic formula

\[
u^{(k)} \approx c_1 \lambda_1^k v_1,
\]

\[ ^\dagger \text{Note that this is \textit{not} the same as the stability criterion for ordinary differential equations, which requires the eigenvalues of the coefficient matrix to lie in the left half plane.} \]
and so most solutions end up parallel to the dominant eigenvector \( \mathbf{v}_1 \). In particular, if \( |\lambda_1| = \rho(T) < 1 \), such a solution approaches \( \mathbf{0} \) along the direction of the dominant eigenvector \( \mathbf{v}_1 \) at a rate governed by the modulus of the dominant eigenvalue.

The exceptional solutions, with \( c_1 = 0 \), tend to \( \mathbf{0} \) at a faster rate, along one of the other eigendirections. However, in practical computations, one rarely observes the exceptional solutions. Indeed, even if one begins with initial conditions for which there is no dominant eigenvector component, round off error will almost inevitably introduce a small component in the direction of \( \mathbf{v}_1 \), which will, if you wait long enough, eventually dominate the computation.

Remark: The inequality (10.18) only applies to complete matrices. In the general case, one can prove that the solution satisfies the slightly weaker inequality

\[
\| \mathbf{u}^{(k)} \| \leq C \sigma^k \quad \text{for all} \quad k \geq 0, \quad \text{where} \quad \sigma > \rho(T) \quad (10.21)
\]

is any number larger than the spectral radius, while \( C > 0 \) is a positive constant (that may depend on how close \( \sigma \) is to \( \rho \)).

**Example 10.14.** According to Example 10.7, the matrix

\[
T = \begin{pmatrix}
-3 & 1 & 6 \\
1 & -1 & -2 \\
-1 & -1 & 0
\end{pmatrix}
\]

has eigenvalues \( \lambda_1 = -2, \lambda_2 = -1 + i, \lambda_3 = -1 - i \).

Since \( |\lambda_1| = 2 > |\lambda_2| = |\lambda_3| = \sqrt{2} \), the spectral radius is \( \rho(T) = |\lambda_1| = 2 \). We conclude that \( T \) is not a convergent matrix. As the reader can check, either directly, or from the solution formula (10.16), the vectors \( \mathbf{u}^{(k)} = T^k \mathbf{u}^{(0)} \) obtained by repeatedly multiplying any nonzero initial vector \( \mathbf{u}^{(0)} \) by \( T \) rapidly go off to \( \infty \), at a rate roughly equal to \( \rho(T)^k = 2^k \).

On the other hand, the matrix

\[
\tilde{T} = -\frac{1}{3} T = \begin{pmatrix}
1 & -\frac{1}{3} & -2 \\
-\frac{1}{3} & \frac{1}{3} & \frac{2}{3} \\
\frac{1}{3} & \frac{1}{3} & 0
\end{pmatrix}
\]

has spectral radius \( \rho(\tilde{T}) = \frac{2}{3} \), and hence is a convergent matrix. According to (10.20), if we write the initial data \( \mathbf{u}^{(0)} = c_1 \mathbf{v}_1 + c_2 \mathbf{v}_2 + c_3 \mathbf{v}_3 \) as a linear combination of the eigenvectors, then, provided \( c_1 \neq 0 \), the iterates have the asymptotic form \( \mathbf{u}^{(k)} \approx c_1 \left( -\frac{2}{3} \right)^k \mathbf{v}_1 \), where \( \mathbf{v}_1 = (4, -2, 1)^T \) is the eigenvector corresponding to the dominant eigenvalue \( \lambda_1 = -\frac{2}{3} \).

Thus, for most initial vectors, the iterates end up decreasing in length by a factor of almost exactly \( \frac{2}{3} \) and become eventually parallel to the dominant eigenvector. This is borne out by a sample computation; starting with \( \mathbf{u}^{(0)} = (1, 1, 1)^T \), the first ten iterates are

\[
\begin{pmatrix}
-0.0936 \\
0.0462 \\
-0.0231
\end{pmatrix}, \quad \begin{pmatrix}
-0.0627 \\
0.0312 \\
-0.0158
\end{pmatrix}, \quad \begin{pmatrix}
-0.0416 \\
0.0208 \\
-0.0105
\end{pmatrix}, \quad \begin{pmatrix}
-0.0275 \\
0.0138 \\
-0.0069
\end{pmatrix}, \quad \begin{pmatrix}
-0.0182 \\
0.0091 \\
-0.0046
\end{pmatrix},
\]

2/25/04

359

© 2004 Peter J. Olver
(−0.0121, −0.0081, −0.0054, −0.0036, −0.0024),
(0.0061, 0.0040, 0.0027, 0.0018, 0.0012),
(−0.0030, −0.0020, −0.0013, −0.0009, −0.0006),

**Fixed Points**

The zero vector \( \mathbf{0} \) is always a fixed point for a linear iterative system \( \mathbf{u}^{(k+1)} = T \mathbf{u}^{(k)} \).

Are there any others? The answer is immediate: \( \mathbf{u}^\ast \) is a fixed point if and only if \( \mathbf{u}^\ast = T \mathbf{u}^\ast \), and hence any nonzero \( \mathbf{u}^\ast \) must be an eigenvector of \( T \) with eigenvalue 1. Thus, the system has a nonzero fixed point if and only if the coefficient matrix \( T \) has 1 as an eigenvalue. Since any scalar multiple of the eigenvector \( \mathbf{u}^\ast \) is also an eigenvector, in such cases the system admits infinitely many fixed points.

The stability of such fixed points, at least if the coefficient matrix is complete, is governed by the same solution formula (10.7). If the eigenvalue \( \lambda_1 = 1 \) is simple, and all other eigenvalues are less than one in modulus, so \( |\lambda_2|, \ldots, |\lambda_n| < 1 \), then the solution takes the asymptotic form

\[
\mathbf{u}^{(k)} = c_1 \mathbf{v}_1 + c_2 \lambda_2^k \mathbf{v}_2 + \cdots + c_n \lambda_n^k \mathbf{v}_n \to c_1 \mathbf{v}_1, \quad \text{as} \quad k \to \infty, \quad (10.22)
\]

converging to one of the fixed points, i.e., a multiple of the eigenvector \( \mathbf{v}_1 \). The actual multiple \( c_1 \) is determined by the initial conditions, as in (10.8). The rate of convergence is governed by the modulus \( |\lambda_2| \) of the subdominant eigenvalue.

The general convergence result governing the stability of fixed points for general coefficient matrices follows.

**Theorem 10.15.** Suppose that \( T \) has a simple (or, more generally, complete) eigenvalue \( \lambda_1 = 1 \), and, moreover, all other eigenvalues satisfy \( |\lambda_j| < 1 \), for \( j \geq 2 \). Then all solutions to the linear iterative system \( \mathbf{u}^{(k+1)} = T \mathbf{u}^{(k)} \) converge to a vector \( \mathbf{v} \in V_1 \) in the eigenspace for the eigenvalue \( \lambda_1 = 1 \).

**Remark:** If \( \lambda = 1 \) is an incomplete eigenvalue, then the solutions do not, in general, converge.

**Example 10.16.** For the matrix \( T = \begin{pmatrix} \frac{3}{2} & -\frac{1}{2} & -3 \\ -\frac{1}{2} & \frac{1}{2} & 1 \\ \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix} \), the eigenvalues and corresponding eigenvectors are

\[
\lambda_1 = 1, \quad \lambda_2 = \frac{1 + i}{2}, \quad \lambda_3 = \frac{1 - i}{2},
\]

\[
\mathbf{v}_1 = \begin{pmatrix} 4 \\ -2 \\ 1 \end{pmatrix}, \quad \mathbf{v}_2 = \begin{pmatrix} 2 - i \\ -1 \\ 1 \end{pmatrix}, \quad \mathbf{v}_3 = \begin{pmatrix} 2 + i \\ -1 \\ 1 \end{pmatrix}.
\]

Since \( \lambda_1 = 1 \), any multiple of the eigenvector \( \mathbf{v}_1 \) is a fixed point. The fixed points are stable since the remaining eigenvalues have modulus \( |\lambda_2| = |\lambda_3| = \frac{1}{2} \sqrt{2} \approx 0.7071 < 1 \). Thus, the iterates \( \mathbf{u}^{(k)} = T^k \mathbf{a} \to c_1 \mathbf{v}_1 \) will eventually converge, at a rate of about .7, to a
multiple of the first eigenvector. For example, starting with $u^{(0)} = (1, 1, 1)^T$, leads to the iterates\[^\dagger\]

\[
\begin{align*}
  u^{(5)} &= \begin{pmatrix} -9.5 \\ 4.75 \\ -2.75 \end{pmatrix}, & u^{(10)} &= \begin{pmatrix} -7.9062 \\ 3.9062 \\ -1.9062 \end{pmatrix}, & u^{(15)} &= \begin{pmatrix} -7.9766 \\ 4.0 \\ -2.0 \end{pmatrix}, \\
  u^{(20)} &= \begin{pmatrix} -8.0088 \\ 4.0029 \\ -2.0029 \end{pmatrix}, & u^{(25)} &= \begin{pmatrix} -7.9985 \\ 3.9993 \\ -1.9993 \end{pmatrix}, & u^{(30)} &= \begin{pmatrix} -8.0001 \\ 4.0001 \\ -2.0001 \end{pmatrix},
\end{align*}
\]

which are slowly converging to the particular eigenvector $(-8, 4, -2)^T = -2v_1$. This can be predicted in advance by decomposing the initial condition into a linear combination of eigenvectors:

\[
  u^{(0)} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = -2 \begin{pmatrix} 4 \\ -2 \\ 1 \end{pmatrix} + \frac{3 + 3i}{2} \begin{pmatrix} 2 - i \\ -1 \\ 1 \end{pmatrix} + \frac{3 - 3i}{2} \begin{pmatrix} 2 + i \\ -1 \\ 1 \end{pmatrix},
\]

whence

\[
  u^{(k)} = \begin{pmatrix} -8 \\ 4 \\ -2 \end{pmatrix} + \frac{3 + 3i}{2} \left( \frac{1 + i}{2} \right)^k \begin{pmatrix} 2 - i \\ -1 \\ 1 \end{pmatrix} + \frac{3 - 3i}{2} \left( \frac{1 - i}{2} \right)^k \begin{pmatrix} 2 + i \\ -1 \\ 1 \end{pmatrix},
\]

and so $u^{(k)} \to (-8, 4, -2)^T$ as $k \to \infty$.

### 10.3. Matrix Norms.

The convergence of a linear iterative system is governed by the spectral radius of the coefficient matrix, and hence knowledge of its eigenvalues is essential. Unfortunately, a priori information on the eigenvalues is not so easy to come by. Indeed, computing accurate approximations to the eigenvalues of a general matrix is a difficult computational problem, and completely satisfactory general numerical algorithms are not known. Indeed, the simplest way to determine the spectral radius is, in fact, to explicitly iterate the matrix and observe how fast the resulting vectors grow or decay. But this defeats its purpose!

An alternative, more practical approach to convergence is based on the concept of a matrix norm. Matrix norms are a natural class of norms on the vector space of $n \times n$ matrices. They often provide comparable convergence information for linear iterative systems, and are simpler to compute.

We work exclusively with real $n \times n$ matrices in this section, although the results straightforwardly extend to complex $n \times n$ matrices. Let us fix a norm $\| \cdot \|$ on $\mathbb{R}^n$. The norm may or may not come from an inner product — this is irrelevant as far as the construction goes. Roughly speaking, the matrix norm tells us how far the matrix stretches vectors relative to the given norm.

\[^\dagger\] Since the convergence is slow, we only display every fifth one.
Theorem 10.17. If \( \| \cdot \| \) is any norm on \( \mathbb{R}^n \), then the quantity

\[
\| A \| = \max \left\{ \| Au \| \mid \| u \| = 1 \right\}
\]

(10.23)
defines a norm on the vector space \( \mathcal{M}_{n \times n} \) of all \( n \times n \) matrices, called the natural matrix norm associated with the given norm \( \| \cdot \| \) on \( \mathbb{R}^n \).

Proof: First note that \( \| A \| < \infty \) since the maximum is taken on a closed and bounded subset, namely the unit sphere \( S_1 = \{ \| u \| = 1 \} \) of the given norm. To show that (10.23) defines a norm, we need to verify the three basic axioms of Definition 3.12. Non-negativity, \( \| A \| \geq 0 \), is immediate. Suppose \( \| A \| = 0 \). This means that, for every unit vector, \( \| Au \| = 0 \), and hence \( Au = 0 \) whenever \( \| u \| = 1 \). If \( 0 \neq v \in \mathbb{R}^n \) is any nonzero vector, then \( u = v/r \), where \( r = \| v \| \), is a unit vector, and so

\[
Av = A(ru) = rAu = 0.
\]

(10.24)

Therefore, \( Av = 0 \) for every \( v \in \mathbb{R}^n \), which implies \( A = 0 \) is the zero matrix. This serves to prove the positivity property. As for homogeneity, if \( c \in \mathbb{R} \) is any scalar,

\[
\| cA \| = \max \left\{ \| cAu \| \right\} = \max \left\{ \| c \| \| Au \| \right\} = \| c \| \| A \|.
\]

Finally, to prove the triangle inequality, we use the fact that the maximum of the sum of quantities is bounded by the sum of their individual maxima. Therefore, since the norm on \( \mathbb{R}^n \) satisfies the triangle inequality,

\[
\| A + B \| = \max \left\{ \|Au + Bu\| \right\} \leq \max \left\{ \|Au\| + \|Bu\| \right\}
\]

\[
\leq \max \left\{ \|Au\| \right\} + \max \left\{ \|Bu\| \right\} = \| A \| + \| B \|.
\]

This completes the proof that the matrix norm satisfies the three basic axioms. \( Q.E.D. \)

The property that distinguishes a matrix norm over a generic norm on the space of matrices is the fact that it obeys a very useful product inequality.

Theorem 10.18. A natural matrix norm satisfies

\[
\| Av \| \leq \| A \| \| v \|,
\]

for all \( A \in \mathcal{M}_{n \times n}, \ v \in \mathbb{R}^n \).

Furthermore,

\[
\| AB \| \leq \| A \| \| B \|,
\]

for all \( A, B \in \mathcal{M}_{n \times n} \).

Proof: Note first that, by definition \( \| Au \| \leq \| A \| \) for all unit vectors \( \| u \| = 1 \). Then, letting \( v = ru \) where \( u \) is a unit vector and \( r = \| v \| \), we have

\[
\| Av \| = \| A(ru) \| = r \| Au \| \leq r \| A \| = \| v \| \| A \|,
\]

proving the first inequality. To prove the second, we apply the first to compute

\[
\| AB \| = \max \left\{ \| A(Bu) \| \right\} = \max \left\{ \| A(Bu) \| \right\}
\]

\[
\leq \max \left\{ \| A \| \| Bu \| \right\} = \| A \| \max \left\{ \| Bu \| \right\} = \| A \| \| B \|.
\]

This completes the proof. \( Q.E.D. \)
Remark: A norm on the vector space of $n \times n$ matrices is called a matrix norm if it also satisfies the multiplicative inequality (10.26). Most, but not all, matrix norms used in applications come from norms on the underlying vector space.

The multiplicative inequality (10.26) implies, in particular, that $\| A^2 \| \leq \| A \|^2$; equality is not necessarily true. More generally,

**Lemma 10.19.** If $A$ is a square matrix, then $\| A^k \| \leq \| A \|^k$. In particular, if $\| A \| < 1$, then $\| A^k \| \to 0$ as $k \to \infty$, and hence $A$ is a convergent matrix: $A^k \to 0$.

The converse is not quite true; a convergent matrix does not necessarily have matrix norm less than 1, or even $\leq 1$ — see Example 10.24 below for an explicit example. An alternative proof of Lemma 10.19 can be based on the following useful estimate.

**Theorem 10.20.** The spectral radius of a matrix is bounded by its matrix norm:

$$\rho(A) \leq \| A \|.$$  \hfill (10.27)

**Proof:** If $\lambda$ is a real eigenvalue, and $u$ a corresponding unit eigenvector, so that $Au = \lambda u$ with $\| u \| = 1$, then

$$\| Au \| = \| \lambda u \| = |\lambda| \| u \| = |\lambda|.$$  \hfill (10.28)

Since $\| A \|$ is the maximum of $\| Au \|$ over all possible unit vectors, this implies that

$$|\lambda| \leq \| A \|.$$  \hfill (10.29)

If all the eigenvalues of $A$ are real, then the spectral radius is the maximum of their absolute values, and so it too is bounded by $\| A \|$, proving (10.27).

If $A$ has complex eigenvalues, then we need to work a little harder. Let $\lambda = r e^{i\theta}$ be a complex eigenvalue with complex eigenvector $z = x + iy$. Define

$$m = \min \left\{ \| \text{Re} e^{i\varphi} z \| = \| (\cos \varphi) x - (\sin \varphi) y \| \mid 0 \leq \varphi \leq 2\pi \right\}.$$  \hfill (10.30)

Since the indicated subset is a closed curve that does not go through the origin†, $m > 0$.

Let $\varphi_0$ denote the value of the angle that produces the minimum, so

$$m = \| (\cos \varphi_0) x - (\sin \varphi_0) y \| = \| \text{Re} (e^{i\varphi_0} z) \|.$$  

Define the real unit vector

$$u = \frac{\text{Re} (e^{i\varphi_0} z)}{m} = \frac{(\cos \varphi_0) x - (\sin \varphi_0) y}{m},$$  

so that $\| u \| = 1$.

Then

$$Au = \frac{1}{m} \text{Re} \left( e^{i\varphi_0} Az \right) = \frac{1}{m} \text{Re} \left( re^{i\varphi_0} e^{i\theta} z \right) = \frac{r}{m} \text{Re} \left( e^{i(\varphi_0 + \theta)} z \right).$$

Therefore, using the fact that $m$ is the minimal value in (10.30),

$$\| A \| \geq \| Au \| = \frac{r}{m} \| \text{Re} (e^{i(\varphi_0 + \theta)} z) \| \geq r = |\lambda|,$$  \hfill (10.31)

and so (10.29) also holds for complex eigenvalues.  \hfill Q.E.D.

† This relies on the fact that $x, y$ are linearly independent, which was shown in Exercise 1.
Explicit Formulae

Let us now determine the explicit formulae for the matrix norms corresponding to our most important vector norms, introduced in Example 3.13. Let us begin with the $\infty$ matrix norm.

**Definition 10.21.** The $i^{th}$ absolute row sum of a matrix $A$ is the sum of the absolute values (moduli) of the entries in the $i^{th}$ row:

$$s_i = |a_{i1}| + \cdots + |a_{in}| = \sum_{j=1}^{n} |a_{ij}|. \quad (10.32)$$

**Proposition 10.22.** The $1$ matrix norm of a matrix $A$ is equal to the maximal absolute row sum:

$$\|A\|_1 = \max\{s_1, \ldots, s_n\} = \max \left\{ \sum_{j=1}^{n} |a_{ij}| \ 1 \leq i \leq n \right\}. \quad (10.33)$$

**Proof:** Let $s = \max\{s_1, \ldots, s_n\}$ denote the right hand side of (10.33). Given any $v \in \mathbb{R}^n$, we compute

$$\|Av\|_\infty = \max \left\{ \left| \sum_{j=1}^{n} a_{ij}v_j \right| \right\} \leq \max \left\{ \sum_{j=1}^{n} |a_{ij}v_j| \right\} \leq \max \left\{ \sum_{j=1}^{n} |a_{ij}| \right\} \max \{ |v_j| \} = s \|v\|_\infty.$$

In particular, by specializing to $\|v\|_\infty = 1$, we deduce that $\|A\|_\infty \leq s$.

On the other hand, suppose the maximal absolute row sum occurs at row $i$, so

$$s_i = \sum_{j=1}^{n} |a_{ij}| = s. \quad (10.34)$$

Let $u$ be defined so that $u_j = +1$ if $a_{ij} > 0$, while $u_j = -1$ if $a_{ij} < 0$. Then $\|u\|_\infty = 1$. Moreover, the $i^{th}$ entry of $Au$ is equal to the $i^{th}$ row sum (10.34). This implies that

$$\|A\|_\infty \geq \|Au\|_\infty \geq s. \quad Q.E.D.$$

**Corollary 10.23.** If $A$ has maximal absolute row sum strictly less than 1, then $\|A\|_\infty < 1$ and hence $A$ is a convergent matrix.

This is an immediate consequence of Lemma 10.19.
Example 10.24. Consider the symmetric matrix \( A = \begin{pmatrix} \frac{1}{2} & -\frac{3}{5} \\ -\frac{3}{5} & \frac{1}{4} \end{pmatrix} \). Its two absolute row sums are \( |\frac{1}{2}| + |-\frac{1}{3}| = \frac{5}{6} \), \( |-\frac{1}{3}| + |\frac{1}{4}| = \frac{7}{12} \) and so
\[
\| A \|_\infty = \max \{ \frac{5}{6}, \frac{7}{12} \} = \frac{5}{6} \approx 0.8333\ldots.
\]
Since the norm is less than 1, \( A \) is a convergent matrix. Indeed, its eigenvalues are
\[
\lambda_1 = \frac{9 + \sqrt{73}}{24} \approx 0.7310\ldots, \quad \lambda_2 = \frac{9 - \sqrt{73}}{24} \approx 0.0190\ldots,
\]
and hence the spectral radius is
\[
\rho(A) = \frac{9 + \sqrt{73}}{24} \approx 0.7310\ldots,
\]
which is slightly smaller than its \( \infty \) norm.

The row sum test for convergence is not always conclusive. For example, the matrix
\[
A = \begin{pmatrix} \frac{1}{2} & -\frac{3}{5} \\ \frac{3}{5} & \frac{1}{4} \end{pmatrix}
\]
has matrix norm \( \| A \|_\infty = \frac{11}{10} > 1 \).

On the other hand, its eigenvalues are \( (15 \pm \sqrt{601})/40 \), and hence its spectral radius is
\[
\rho(A) = \frac{15 + \sqrt{601}}{40} \approx 0.98788\ldots,
\]
which implies that \( A \) is (just barely) convergent, even though its maximal row sum is larger than 1.

The Euclidean matrix norm relies on the singular value decomposition of Theorem 8.33.

Proposition 10.25. The matrix norm corresponding to the Euclidean norm is its maximal singular value
\[
\| A \|_2 = \max\{\sigma_1, \ldots, \sigma_n\}.
\]

Proof: We use the singular value decomposition (8.41) to write
\[
A = Q_1 \Sigma Q_2^T
\]
where \( Q_1 \) and \( Q_2 \) are orthogonal matrices, while \( \Sigma = \text{diag}(\sigma_1, \ldots, \sigma_n) \) is the diagonal matrix containing the singular values of \( A \). Using the Euclidean norm-preserving property (7.32) of orthogonal matrices, we have
\[
\| A \mathbf{u} \|_2 = \| Q_1 \Sigma Q_2^T \mathbf{u} \|_2 = \| \Sigma Q_2^T \mathbf{u} \|_2.
\]
Now, if \( \mathbf{u} \) is a unit vector, \( \| \mathbf{u} \|_2 = 1 \), then so is \( \tilde{\mathbf{u}} = Q_2^T \mathbf{u} \). Therefore,
\[
\| A \|_2 = \max \{ \| A \mathbf{u} \|_2 \mid \| \mathbf{u} \|_2 = 1 \}
= \max \{ \| \Sigma Q_2^T \mathbf{u} \|_2 \mid \| \mathbf{u} \|_2 = 1 \}
= \max \{ \| \Sigma \tilde{\mathbf{u}} \|_2 \mid \| \tilde{\mathbf{u}} \|_2 = 1 \}.
\]
If we order the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$, then

$$
\| \Sigma \tilde{u} \|_2 = \sqrt{\sigma_1^2 \tilde{u}_1^2 + \cdots + \sigma_n^2 \tilde{u}_n^2} \leq \sigma_1 \sqrt{\tilde{u}_1^2 + \cdots + \tilde{u}_n^2} = \sigma_1 \quad \text{since} \quad \| \tilde{u} \|_2 = 1.
$$

On the other hand, if we set $\tilde{u} = e_1$ to be the first standard basis vector, then $\| \Sigma e_1 \|_2 = \| \sigma_1 e_1 \|_2 = \sigma_1$. When put together, these imply that $\| A \|_2 = \sigma_1$, which proves the result. \textit{Q.E.D.}

\textbf{Corollary 10.26.} If $A$ is symmetric, its Euclidean matrix norm is equal to its spectral radius.

\textit{Proof:} This follows directly from the fact, proved in Proposition 8.31, that the singular values of a symmetric matrix are just the absolute values of its eigenvalues. \textit{Q.E.D.}

\textbf{Example 10.27.} Consider the matrix $A = \begin{pmatrix} 0 & -\frac{1}{3} & \frac{1}{3} \\ \frac{1}{4} & 0 & \frac{1}{2} \\ \frac{2}{5} & \frac{1}{5} & 0 \end{pmatrix}$. The Gram matrix $A^T A = \begin{pmatrix} 0.2225 & 0.0800 & 0.1250 \\ 0.0800 & 0.1511 & -0.1111 \\ 0.1250 & -0.1111 & 0.3611 \end{pmatrix}$, has eigenvalues $\lambda_1 = 0.4472, \lambda_2 = 0.2665, \lambda_3 = 0.0210$, and hence the singular values of $A$ are the square roots: $\sigma_1 = 0.6687, \sigma_2 = 0.5163, \sigma_3 = 0.1448$. The Euclidean matrix norm of $A$ is the largest singular value, and so $\| A \|_2 = 0.6687$, proving that $A$ is a convergent matrix. Note that, as always, the matrix norm overestimates the spectral radius $\rho(A) = .5$.

Unfortunately, as we discovered in Example 10.24, matrix norms are not a foolproof test of convergence. There exist convergent matrices such that $\rho(A) < 1$ and yet have matrix norm $\| A \| \geq 1$. In such cases, we will not be able to predict the convergence of the iterative system based on the matrix, although we would expect the convergence to be quite slow. Although such pathology might show up in one particular matrix norm, it turns out that one can always find some matrix norm which is less than 1. A proof of this result can be found in [119].

\textbf{Theorem 10.28.} Let $A$ have spectral radius $\rho(A)$. If $\varepsilon > 0$ is any positive number, then there exists a matrix norm $\| \cdot \|$ such that

$$
\rho(A) \leq \| A \| < \rho(A) + \varepsilon. \quad (10.36)
$$

\textbf{Corollary 10.29.} If $A$ is a convergent matrix, then there exists a matrix norm such that $\| A \| < 1$.

\textit{Proof:} By definition, $A$ is convergent if and only if $\rho(A) < 1$. Choose $\varepsilon > 0$ such that $\rho(A) + \varepsilon < 1$. Any norm that satisfies (10.36) has the desired property. \textit{Q.E.D.}
Remark: Based on the accumulated evidence, one might be tempted to speculate that the spectral radius itself defines a matrix norm. Unfortunately, this is not the case. For example, the nonzero matrix \( A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \) has zero spectral radius, \( \rho(A) = 0 \), violating a basic norm axiom.

The Gerschgorin Circle Theorem

In general, precisely computing the eigenvalues, and hence the spectral radius of a matrix is not easy, and, in most cases, must be done through a numerical eigenvalue routine. In many applications, though, one does not need their exact numerical values, but only their approximate locations. The Gerschgorin Circle Theorem serves to restrict the eigenvalues to a certain well-defined region in the complex plane. In favorable situations, this information, which is relatively easy to obtain, is sufficient to demonstrate the convergence of the matrix.

Definition 10.30. Let \( A \) be an \( n \times n \) matrix, either real or complex. For each \( 1 \leq i \leq n \), define the Gerschgorin disk

\[
D_i = \{ \| z - a_{ii} \| \leq r_i \mid z \in \mathbb{C} \}, \quad \text{where} \quad r_i = \sum_{i=1}^{n} |a_{ij}|. \tag{10.37}
\]

The Gerschgorin domain \( D = \bigcup_{i=1}^{n} D_i \subset \mathbb{C} \) is the union of the Gerschgorin disks.

Thus, the \( i \)th Gerschgorin disk \( D_i \) is centered at the \( i \)th diagonal entry \( a_{ii} \), and has radius \( r_i \) equal to the sum of the absolute values of the off-diagonal entries that are in the \( i \)th row of \( A \).

Theorem 10.31. All real and complex eigenvalues of the matrix \( A \) lie in its Gerschgorin domain \( D \).

Example 10.32. The matrix \( A = \begin{pmatrix} 2 & -1 & 0 \\ 1 & 4 & -1 \\ -1 & -1 & -3 \end{pmatrix} \) has Gerschgorin disks

\[
D_1 = \{ \| z - 2 \| \leq 1 \}, \quad D_2 = \{ \| z - 4 \| \leq 2 \}, \quad D_3 = \{ \| z + 3 \| \leq 2 \},
\]

which are plotted in Figure 10.2. The eigenvalues of \( A \) are

\[
\lambda_1 = 3, \quad \lambda_2 = 3.1623 \ldots, \quad \lambda_3 = -3.1623 \ldots.
\]

Observe that \( \lambda_1 \) belongs to both \( D_1 \) and \( D_2 \), while \( \lambda_2 \) lies in \( D_2 \), and \( \lambda_3 \) in \( D_3 \). We thus confirm that all three eigenvalues are in the Gerschgorin domain \( D = D_1 \cup D_2 \cup D_3 \).

Proof of Theorem 10.31: Let \( v \) be an eigenvector of \( A \) with eigenvalue \( \lambda \). Let \( u = v / \| v \|_\infty \) be the corresponding unit eigenvector with respect to the \( \infty \) norm, so

\[
\| u \|_\infty = \max\{ |u_1|, \ldots, |u_n| \} = 1.
\]
Let $u_i$ be an entry of $\mathbf{u}$ that achieves the maximum: $|u_i| = 1$. Writing out the eigenvalue equation $A\mathbf{u} = \lambda \mathbf{u}$ in components, we find

$$
\sum_{j=1}^{n} a_{ij} u_j = \lambda u_i, \quad \text{which we rewrite as} \quad \sum_{j=1}^{n} a_{ij} u_j = (\lambda - a_{ii}) u_i.
$$

Therefore, since all $|u_i| \leq 1$,

$$
|\lambda - a_{ii}| |u_i| \leq \left| \sum_{j \neq i} a_{ij} u_j \right| \leq \sum_{j \neq i} |a_{ij}| |u_j| \leq \sum_{j \neq i} |a_{ij}| = r_i.
$$

Since we chose $u_i$ so that $|u_i| = 1$, we conclude that $\lambda$ satisfies

$$
|\lambda - a_{ii}| \leq r_i,
$$

and hence $\lambda \in D_i \subset D$ belongs to the $i$th Gerschgorin disk. \ \ \ \ Q.E.D.

The Gerschgorin Theorem 10.31 can be used to give a direct proof of Corollary 10.23. If $A$ is any matrix, then the modulus of all points $z \in D_i$ contained in its $i$th Gerschgorin disk is bounded by the $i$th absolute row sum,

$$
|z| \leq |z - a_{ii}| + |a_{ii}| \leq r_i + |a_{ii}| = s_i,
$$

where the final equality follows by comparison of (10.37) and (10.32). Thus, every point $z \in D$ in the Gerschgorin set has modulus

$$
|z| \leq \max\{s_1, \ldots, s_n\} = \|A\|_\infty,
$$

bounded by the maximal row sum. Since all eigenvalues $\lambda_j$ of $A$ are contained in $D$, they too satisfy

$$
|\lambda_j| \leq \|A\|_\infty, \quad \text{and hence} \quad \rho(A) \leq \|A\|_\infty.
$$

By hypothesis, $1 > \|A\|_\infty \geq \rho(A)$, and hence $A$ is a convergent matrix.
As a second application, we give a simple direct test that guarantees invertibility of a matrix without requiring Gaussian elimination or computing determinants. Recall that a matrix is nonsingular if and only if it does not have a zero eigenvalue. Thus, if its Gerschgorin domain does not contain 0 \( \notin D \), then the matrix cannot have 0 as an eigenvalue, and hence is necessarily invertible. This condition requires that the matrix have large diagonal entries, as quantified by the following definition.

**Definition 10.33.** A square matrix \( A \) is called strictly diagonally dominant if

\[
|a_{ii}| > \sum_{\substack{i=1\atop i\neq j}}^{n} |a_{ij}|, \quad \text{for all} \quad i = 1, \ldots, n. \quad (10.39)
\]

In other words, for \( A \) to be diagonally dominant, its diagonal entry must be larger, in absolute value, than the sum of all the other entries in its row. For example, the matrix

\[
A = \begin{pmatrix}
3 & 1 & -1 \\
1 & -4 & 2 \\
-2 & -1 & 5
\end{pmatrix}
\]

is strictly diagonally dominant since

\[
|3| > |1| + |-1|, \quad |-4| > |1| + |2|, \quad |5| > |-2| + |-1|.
\]

Diagonally dominant matrices arise in many applications, particularly in finite difference and finite element methods for numerically solving boundary value problems. As we shall see, they are the most common class of matrices to which iterative solution methods can be successfully applied.

**Proposition 10.34.** A strictly diagonally dominant matrix is nonsingular.

**Proof:** The diagonal dominance inequalities (10.39) imply that the radius of the \( i \text{th} \) Gerschgorin disk is strictly less than the modulus of its center: \( r_i < |a_{ii}|. \) Thus, the disk cannot contain 0; indeed, if \( z \in D_i \), then, by the triangle inequality

\[
r_i > |z - a_{ii}| \geq |a_{ii}| - |z| > r_i - |z|, \quad \text{and hence} \quad |z| > 0.
\]

Thus, 0 \( \notin D \) does not lie in the Gerschgorin domain and hence cannot be an eigenvalue.

**Warning:** The converse is obviously not true. There are plenty of nonsingular matrices that are not diagonally dominant.

### 10.4. Markov Processes.

A discrete process in which the probability of a system being in a particular state during a given time period depends only its state in the immediately preceding time period is known as a **Markov chain**, in honor of the pioneering studies of the Russian mathematician Andrei Markov. Markov chains are the beginning of the theory of stochastic processes. They are described by linear iterative systems whose coefficient matrices have a special form, and hence can be analyzed by our eigenvalue methods.
To take a very simple example, suppose you are interested in predicting whether the weather in your city on a particular day will be either sunny or cloudy. Consulting weather records over the past decade, you determine that

(i) If today is sunny, there is a 70% chance that tomorrow will also be sunny,
(ii) But, if today is cloudy, the chances are 80% that tomorrow is also cloudy.

Question: given that today is sunny, what is the probability that next Saturday’s weather will also be sunny?

To mathematically formulate this process, we let $s^{(k)}$ denote the probability that day $k$ is sunny and $c^{(k)}$ the probability that it is cloudy. If we assume that these are the only possibilities, then the individual probabilities must sum to 1, so

$$s^{(k)} + c^{(k)} = 1.$$ 

According to our data, the probability that the next day is sunny or cloudy is expressed by the equations

$$s^{(k+1)} = .7 s^{(k)} + .2 c^{(k)}, \quad c^{(k+1)} = .3 s^{(k)} + .8 c^{(k)}. \quad (10.40)$$

Indeed, day $k + 1$ could be sunny either if day $k$ was, with a 70% chance, or, if day $k$ was cloudy, there is still a 20% chance of day $k + 1$ being sunny. We rewrite (10.40) in a more convenient matrix form:

$$u^{(k+1)} = T u^{(k)}, \quad \text{where} \quad T = \begin{pmatrix} .7 & .2 \\ .3 & .8 \end{pmatrix}, \quad u^{(k)} = \begin{pmatrix} s^{(k)} \\ c^{(k)} \end{pmatrix}. \quad (10.41)$$

In a Markov process, the vector of probabilities $u^{(k)}$ is known as the $k^{th}$ state vector and the matrix $T$ is known as the transition matrix, whose entries fix the transition probabilities between the states.

By assumption, our initial state vector is $u^{(0)} = (1, 0)^T$, since we know for certain that today is sunny. Rounding off to three decimal places, the subsequent state vectors are

$$u^{(1)} = \begin{pmatrix} .7 \\ .3 \end{pmatrix}, \quad u^{(2)} = \begin{pmatrix} 0.55 \\ 0.45 \end{pmatrix}, \quad u^{(3)} = \begin{pmatrix} 0.475 \\ 0.525 \end{pmatrix}, \quad u^{(4)} = \begin{pmatrix} 0.438 \\ 0.563 \end{pmatrix},$$

$$u^{(5)} = \begin{pmatrix} 0.419 \\ 0.581 \end{pmatrix}, \quad u^{(6)} = \begin{pmatrix} 0.410 \\ 0.591 \end{pmatrix}, \quad u^{(7)} = \begin{pmatrix} 0.405 \\ 0.595 \end{pmatrix}, \quad u^{(8)} = \begin{pmatrix} 0.402 \\ 0.598 \end{pmatrix}.$$ 

The iterates converge fairly rapidly to $(.4, .6)^T$, which is a fixed point for the iterative system (10.41). Thus, in the long run, 40% of the days will be sunny and 60% will be cloudy. Let us explain why this happens.

**Definition 10.35.** A vector $u = (u_1, u_2, \ldots, u_n)^T \in \mathbb{R}^n$ is called a probability vector if all its individual entries lie between 0 and 1, so $0 \leq u_i \leq 1$, and, moreover, the sum of its entries is unity: $u_1 + \cdots + u_n = 1.$
For example, the possible probability vectors $\mathbf{u} \in \mathbb{R}^3$ fill the equilateral triangle plotted in Figure 10.3. We interpret the entry $u_i$ of a probability vector as the probability the system is in state number $i$. The fact that the entries add up to 1 means that they represent a complete list of probabilities for the possible states of the system.

**Remark:** Any nonzero vector $\mathbf{0} \neq \mathbf{v} = (v_1, v_2, \ldots, v_n)^T$ with all non-negative entries: $v_i \geq 0$ for $i = 1, \ldots, n$, can be converted into a parallel probability vector by dividing by the sum of its entries:

$$
\mathbf{u} = \frac{\mathbf{v}}{v_1 + \cdots + v_n}.
$$

For example, if $\mathbf{v} = (3, 2, 0, 1)^T$, then $\mathbf{u} = \left( \frac{1}{2}, \frac{1}{3}, 0, \frac{1}{6} \right)^T$ is the corresponding probability vector.

In general, a *Markov chain* is represented by a first order linear iterative system

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

The *transition matrix*

$$
T = (t_{ij}), \quad 0 \leq t_{ij} \leq 1, \quad t_{1j} + \cdots + t_{nj} = 1,
$$

contains all the transitional probabilities. The entry $t_{ij}$ represents the probability that the system will switch from state $j$ to state $i$. (Note the reversal of indices.) Since this covers all possible transitions, the *column sums* of the transition matrix are all equal to 1, and hence each column of $T$ is a probability vector. An easy Exercise [94] shows that if $\mathbf{u}^{(k)}$ is a probability vector, so is $\mathbf{u}^{(k+1)} = T \mathbf{u}^{(k)}$. Thus, the solution $\mathbf{u}^{(k)} = T^k \mathbf{u}^{(0)}$ to the Markov process represents a sequence or “chain” of probability vectors.

It can be proved, [94] that every transition matrix $T$ is complete, and hence admits an eigenvector basis $\mathbf{v}_1, \ldots, \mathbf{v}_n$ with associated eigenvalues $\lambda_1, \ldots, \lambda_n$ (some of which may be repeated). Therefore, by Theorem 10.4, the solution to the Markov process (10.43) is

$$
\mathbf{u}^{(k)} = T^k \mathbf{u}^{(0)} = c_1 \lambda_1^k \mathbf{v}_1 + \cdots + c_n \lambda_n^k \mathbf{v}_n.
$$

2/25/04 371 © 2004 Peter J. Olver
where $\lambda_1, \ldots, \lambda_n$ are the eigenvalues and $v_1, \ldots, v_n$ the corresponding eigenvectors.

Let us now investigate the convergence of the Markov chain. This will not happen in general, but requires some additional mild restrictions on the transition matrix.

**Definition 10.36.** A transition matrix (10.44) is *regular* if some power $T^k$ contains no zero entries. In particular, if $T$ itself has no transition probabilities equal to 0, then it is regular.

*Warning:* The term “regular transition matrix” is *not* the same as our earlier term “regular matrix”, which was used to describe matrices with an $LU$ factorization.

The asymptotic behavior of a regular Markov chain is governed by the following key result.

**Theorem 10.37.** If $T$ is a regular transition matrix, then it admits a unique probability eigenvector $u^*$ with eigenvalue $\lambda_1 = 1$. Moreover, any Markov chain with coefficient matrix $T$ will converge to the distinguished probability vector: $u^{(k)} \to u^*$ as $k \to \infty$.

The proof of this result appears at the end of this section.

**Example 10.38.** For the weather transition matrix (10.41), the eigenvalues and eigenvectors are

\[
\begin{align*}
\lambda_1 &= 1, & v_1 &= \left( \frac{2}{3}, 1 \right), \\
\lambda_2 &= .5, & v_2 &= \left( -1, 1 \right).
\end{align*}
\]

The first eigenvector is then converted into a probability vector via (10.42):

\[
u^* = u_1 = \frac{1}{1 + \frac{2}{3}} \left( \frac{2}{3}, 1 \right) = \left( \frac{2}{5}, \frac{3}{5} \right).
\]

This distinguished probability eigenvector represents the final asymptotic state of the system after many iterations, *no matter what the initial state*. Thus, our earlier observation that about 40% of the days will be sunny and 60% will be cloudy holds *no matter what the initial weather is*.

**Example 10.39.** A taxi company in Minnesota serves the cities of Minneapolis and St. Paul, as well as the nearby suburbs. Records indicate that, on average, 10% of the customers taking a taxi in Minneapolis go to St. Paul and 30% go to the suburbs. Customers boarding in St. Paul have a 30% chance of going to Minneapolis and 30% chance of going to the suburbs, while suburban customers choose Minneapolis 40% of the time and St. Paul 30% of the time. The owner of the taxi company is interested in knowing where the taxis will end up, on average. We write this as a Markov process. The entries of the state vector $u^{(k)} = (u_1^{(k)}, u_2^{(k)}, u_3^{(k)})^T$ tell what proportion of the taxi fleet is, respectively, in Minneapolis, St. Paul and the suburbs. Using the data, we construct the relevant transition matrix

\[
T = \begin{pmatrix} .6 & .3 & .4 \\
.1 & .4 & .3 \\
.3 & .3 & .3 \end{pmatrix}.
\]
Figure 10.4. Gerschgorin Disks for a Transition Matrix.

Note that $T$ regular since it has no zero entries. The probability eigenvector

$$u^* = (0.471429 \ldots \ 0.228571 \ldots \ 0.3)^T$$

corresponding to the unit eigenvalue $\lambda_1 = 1$ is found by first solving the linear system $(T - I)v = 0$ and then converting the solution $v$ into a valid probability vector by use of formula (10.42). According to Theorem 10.37, no matter how the taxis are initially distributed, ultimately about 47% of the taxis will be in Minneapolis, 23% in St. Paul, and 30% in the suburbs. This can be confirmed by running numerical experiments on the system.

Remark: The convergence rate of the Markov chain to its steady state is governed by the size of the subdominant or second largest eigenvalue $\lambda_2$. The smaller $|\lambda_2|$ is to 0, the faster the process converges. In the taxi example, $\lambda_2 = .3$ (and $\lambda_3 = 0$) and so the convergence to steady state is fairly rapid.

Proof of Theorem 10.37: We begin the proof by replacing $T$ by its transpose $M = T^T$, keeping in mind that every eigenvalue of $T$ is also an eigenvalue of $M$, cf. Exercise 10.44. The conditions (10.44) tell us that the matrix $M$ has entries $0 \leq m_{ij} = t_{ji} \leq 1$, and, moreover, the (absolute) row sums $s_i = \sum_{i=1}^{n} m_{ij} = 1$ of $M$, being the same as the corresponding column sums of $T$, are all equal to 1. Since $M^k = (T^k)^T$, regularity of $T$ implies that some power $M^k$ has all positive entries.

---

$^\dagger$ Theorem 10.37 guarantees an eigenvector $v$ with all non-negative entries.
According to Exercise 1, if \( \mathbf{z} = (1, \ldots, 1)^T \) is the column vector all of whose entries are equal to 1, then the entries of \( M \mathbf{z} \) are the row sums of \( M \). Therefore, \( M \mathbf{z} = \mathbf{z} \), which implies that \( \mathbf{z} \) is an eigenvector of \( M \) with eigenvalue \( \lambda_1 = 1 \). As a consequence, \( T \) also has 1 as an eigenvalue — although it is associated with a different eigenvector, not necessarily a multiple of \( \mathbf{z} \).

Let us next prove that \( \lambda_1 = 1 \) is a simple eigenvalue. Since \( T \) is complete, this is equivalent to the statement that the only vectors satisfying \( M \mathbf{v} = \mathbf{v} \) are those with all equal entries \( v_1 = \cdots = v_n = a \), and hence \( \mathbf{v} = a \mathbf{z} \) is a scalar multiple of the particular eigenvector \( \mathbf{z} \). Let us first prove this assuming all of the entries of \( M \) are positive, and so \( 0 < m_{ij} = t_{ji} < 1 \) for all \( i, j \). Suppose \( \mathbf{v} \) is an eigenvector with not all equal entries. Let \( v_k \) be the minimal entry of \( \mathbf{v} \), so \( v_k \leq v_i \) for all \( i \neq k \) and at least one inequality is strict, say \( v_k < v_j \). Then the \( k^{th} \) entry of the eigenvector equation \( \mathbf{v} = M \mathbf{v} \) is

\[
v_k = \sum_{j=1}^{n} m_{kj} v_j < \sum_{j=1}^{n} m_{kj} v_k = v_k,
\]

where the strict inequality follows from the positivity of the entries of \( M \), and the final equality follows from the fact that \( M \) has unit row sums. Thus, we are led to a contradiction, and the claim follows. If \( M \) has one or more 0 entries, but \( M^k \) has all positive entries, then we apply the previous argument to the equation \( M^k \mathbf{v} = \mathbf{v} \) which follows from \( M \mathbf{v} = \mathbf{v} \).

Finally, let us prove that all the other eigenvalues of \( M \) are less than 1 in modulus. For this we appeal to the Gerschgorin Circle Theorem 10.31. The Gerschgorin disk \( D_i \) is centered at \( m_{ii} \) and has radius \( r_i = s_i - m_{ii} = 1 - m_{ii} \). Thus the disk lies strictly inside the open unit disk \( |z| < 1 \) except for a single boundary point at \( z = 1 \); see Figure 10.4. The Circle Theorem 10.31 implies that all eigenvalues except the unit eigenvalue \( \lambda_1 = 1 \) must lie strictly inside the unit disk, and so \( |\lambda_j| < 1 \) for \( j \geq 2 \).

Therefore, the matrix \( M \), and, hence, also \( T \) satisfy the hypotheses of Theorem 10.15. We conclude that the iterates \( \mathbf{u}^{(k)} = T^k \mathbf{u}^{(0)} \to \mathbf{u}^* \) converge to a multiple of the unit eigenvector of \( T \). If the initial condition \( \mathbf{u}^{(0)} \) is a probability vector, then is every subsequent state vector \( \mathbf{u}^{(k)} \), and so their limit \( \mathbf{u}^* \) must also be a probability vector. This completes the proof of the theorem. Q.E.D.

10.5. Iterative Solution of Linear Systems.

In this section, we introduce several basic iterative methods that are used to approximate the solution of certain classes of linear systems

\[
A \mathbf{u} = \mathbf{b},
\]

consisting of \( n \) equations in \( n \) unknowns. The resulting algorithms will provide an attractive alternative to Gaussian elimination, particularly when dealing with the large, sparse systems that arise in the numerical solution to differential equations. One major advantage of an iterative technique is that it produces progressively more and more accurate approximations to the solution, and hence, by prolonging the iterations, one can, in principle, compute the solution to any desired order of accuracy — although, in practice, the
round-off errors due to the finite precision of the computer will eventually be an issue. Moreover, even performing just a few iterations may produce a reasonable approximation to the true solution — in stark contrast to Gaussian elimination, where one must continue the algorithm through to the bitter end before any useful information can be extracted. A partially completed Gaussian elimination is of scant use! On the other hand, specific iterative schemes are not universally applicable to all linear systems, and their design relies upon the detailed structure of the coefficient matrix.

We shall be attempting to solving (10.46) by an iterative system of the form

\[ u^{(k+1)} = T u^{(k)} + c, \quad u^{(0)} = u_0, \quad (10.47) \]

where \( T \) is a fixed \( n \times n \) matrix and \( c \) a fixed vector. This is a slight generalization of the linear iterative system (10.1), in that the right hand side is now an affine function of \( u^{(k)} \). If the solutions to the affine iterative system converge, \( u^{(k)} \to u^* \) as \( k \to \infty \), then \( u^* \) solves the fixed-point equation

\[ u^* = T u^* + c. \quad (10.48) \]

Indeed, both \( u^{(k)} \) and \( u^{(k+1)} \) in (10.47) converge to the same \( u^* \), and so the system converges to the limiting fixed point equation (10.48). Thus we need to design our system so that

(a) The solution to the fixed-point system (10.48) coincides with the solution to the original system (10.46), and

(b) The iterates defined by (10.47) converge to the solution \( u^* \).

Before exploring these issues in depth, let us look at a simple example.

**Example 10.40.** Consider the linear system

\[ 3x + y - z = 3, \quad x - 4y + 2z = -1, \quad -2x - y + 5z = 2, \quad (10.49) \]

which we rewrite in matrix form \( A u = b \), with

\[ A = \begin{pmatrix} 3 & 1 & -1 \\ 1 & -4 & 2 \\ -2 & -1 & 5 \end{pmatrix}, \quad u = \begin{pmatrix} x \\ y \\ z \end{pmatrix}, \quad b = \begin{pmatrix} 3 \\ -1 \\ 2 \end{pmatrix}. \]

One easy way to rewrite the system in fixed-point form (10.48) is to set

\[ T = I - A = \begin{pmatrix} -2 & -1 & 1 \\ -1 & 5 & -2 \\ 2 & 1 & -4 \end{pmatrix}, \quad c = b = \begin{pmatrix} 3 \\ -1 \\ 2 \end{pmatrix}. \quad (10.50) \]

Clearly, \( A u = b \) if and only if \( T u + b = (I - A)u + b = u \), and hence the fixed point coincides with the solution to the original system. The resulting iterative system \( u^{(k+1)} = T u^{(k)} + c \) has the explicit form

\[ x^{(k+1)} = -2x^{(k)} - y^{(k)} + z^{(k)} + 3, \]
\[ y^{(k+1)} = -x^{(k)} + 5y^{(k)} - 2z^{(k)} - 1, \]
\[ z^{(k+1)} = 2x^{(k)} + y^{(k)} - 4z^{(k)} + 2. \]
Another possibility is to solve the first equation in (10.49) for $x$, the second for $y$ and the third for $z$, so that

\[ x = -\frac{1}{3} y + \frac{1}{3} z + 1, \quad y = \frac{1}{4} x + \frac{1}{2} z + \frac{1}{4}, \quad z = \frac{2}{5} x + \frac{1}{5} y + \frac{2}{5}. \]

The solution to this fixed point system also coincide with that of the original linear system. The corresponding iteration takes the form

\[
\begin{align*}
x^{(k+1)} &= -\frac{1}{3} y^{(k)} + \frac{1}{3} z^{(k)} + 1, \\
y^{(k+1)} &= \frac{1}{4} x^{(k)} + \frac{1}{2} z^{(k)} + \frac{1}{4}, \\
z^{(k+1)} &= \frac{2}{5} x^{(k)} + \frac{1}{5} y^{(k)} + \frac{2}{5}.
\end{align*}
\]

(10.51)

In matrix notation, this becomes

\[
u^{(k+1)} = \hat{T} u^{(k)} + \hat{c}, \quad \text{where} \quad \hat{T} = \begin{pmatrix} 0 & -\frac{1}{3} & \frac{1}{3} \\ \frac{1}{4} & 0 & \frac{1}{2} \\ \frac{2}{5} & \frac{1}{5} & 0 \end{pmatrix}, \quad \hat{c} = \begin{pmatrix} 1 \\ 0 \\ \frac{2}{5} \end{pmatrix}.
\]

(10.52)

Do the resulting iterative schemes (10.47) converge to the solution $x = y = z = 1$? The results, starting with initial guess $u^{(0)} = (0, 0, 0)$, appear in the following table.

<table>
<thead>
<tr>
<th>$k$</th>
<th>$u^{(k+1)} = T u^{(k)} + b$</th>
<th>$u^{(k+1)} = \hat{T} u^{(k)} + \hat{c}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1, 0.25, 0.4</td>
</tr>
<tr>
<td>2</td>
<td>0, -13</td>
<td>1.05, 0.7, 0.85</td>
</tr>
<tr>
<td>3</td>
<td>15, -64, -7</td>
<td>1.05, 0.9375, 0.96</td>
</tr>
<tr>
<td>4</td>
<td>30, -322, -4</td>
<td>1.0075, 0.9925, 1.0075</td>
</tr>
<tr>
<td>5</td>
<td>261, -1633, -244</td>
<td>1.005, 1.00562, 1.0015</td>
</tr>
<tr>
<td>6</td>
<td>870, -7939, -133</td>
<td>0.9986, 1.002, 1.0031</td>
</tr>
<tr>
<td>7</td>
<td>6069, -40300, -5665</td>
<td>1.0004, 1.0012, 0.9999</td>
</tr>
<tr>
<td>8</td>
<td>22500, -196240, -5500</td>
<td>0.9995, 1.0000, 1.0004</td>
</tr>
<tr>
<td>9</td>
<td>145743, -992701, -129238</td>
<td>1.0001, 1.0001, 0.9998</td>
</tr>
<tr>
<td>10</td>
<td>571980, -4850773, -184261</td>
<td>0.9999, 0.9999, 1.0001</td>
</tr>
<tr>
<td>11</td>
<td>3522555, -24457324, -2969767</td>
<td>1.0000, 1.0000, 1.0000</td>
</tr>
</tbody>
</table>

For the first scheme, the answer is no — the iterations become successively wilder and wilder. Indeed, this occurs no matter how close the initial guess is to the actual solution — unless it happens to be exactly equal: $u^{(0)} = u^*$. (And even then, numerical errors could creep in and send the iterations off to $\infty$.) In the second case, the convergence is quite good, and it does not take too long, even starting from a bad initial guess, to obtain an accurate approximation to the solution.
Of course, in such a simple example, it would be silly to use iteration, when Gaussian elimination can be done by hand and produces the solution almost immediately. However, we use the small examples for illustrative purposes, reserving the full-fledged application of the iterative schemes to the large linear systems arising in applications.

The convergence of solutions to (10.47) to the fixed point $u^*$ is based on the behavior of the error vectors

$$e^{(k)} = u^{(k)} - u^*, \quad (10.53)$$

which measure how close the iterates are to the actual solution. Let us find out how the successive error vectors are related. We compute

$$e^{(k+1)} = u^{(k+1)} - u^* = (T u^{(k)} + a) - (T u^* + a) = T(u^{(k)} - u^*) = T e^{(k)}. \quad (10.54)$$

Therefore, the error vectors satisfy a linear iterative system

$$e^{(k+1)} = T e^{(k)},$$

with the same coefficient matrix $T$. Therefore, the errors are given by the explicit formula $e^{(k)} = T^k e^{(0)}$. Now, the solutions to (10.47) converge to the fixed point, $u^{(k)} \to u^*$, if and only if the error vectors $e^{(k)} \to 0$ as $k \to \infty$. Consequently, our convergence results for linear iterative systems, as summarized in Proposition 10.10, imply the following basic result.

**Proposition 10.41.** The iterative system (10.47) will converge to the solution to the fixed point equation (10.48) if and only if $T$ is a convergent matrix: $\rho(T) < 1$.

For example, in the two iterative schemes presented in Example 10.40, the spectral radii of the coefficient matrices are found to be

$$\rho(T) = 4.9675\ldots, \quad \rho(\hat{T}) = 0.5.$$ 

Therefore, $T$ is not a convergent matrix, which explains the behavior of its iterates, whereas $\hat{T}$ is convergent, and one expects the error to roughly decrease by a factor of $\frac{1}{2}$ with each new iterate.

The spectral radius $\rho(T)$ of the coefficient matrix will govern the speed of convergence. Therefore, the main goal is to construct an iterative scheme whose coefficient matrix has as small a spectral radius as possible. At the very least, the spectral radius must be less than 1.

**The Jacobi Method**

The first general iterative scheme for solving linear systems is based on the same simple idea used in our illustrative Example 10.40. Namely, we solve the $i$th equation in the system $A u = b$, which is

$$\sum_{j=1}^{n} a_{ij} u_j = b_i,$$
for the \(i\)th variable. To do this, we need to assume that all the diagonal entries of \(A\) are nonzero: \(a_{ii} \neq 0\). The result is

\[
u_i = -\frac{1}{a_{ii}} \sum_{i \neq j = 1}^n a_{ij} u_j + \frac{b_i}{a_{ii}} = \sum_{j = 1}^n t_{ij} u_j + c_i,
\]

(10.55)

where

\[
t_{ij} = \begin{cases} \frac{-a_{ij}}{a_{ii}}, & i \neq j, \\ 0, & i = j,
\end{cases}
\]

and \(c_i = \frac{b_i}{a_{ii}}\).

(10.56)

Equation (10.55) can be rewritten in fixed point form \(\mathbf{u} = T\mathbf{u} + \mathbf{c}\), and forms the basis of the Jacobi method

\[
\mathbf{u}^{(k+1)} = T\mathbf{u}^{(k)} + \mathbf{c}, \quad \mathbf{u}^{(0)} = \mathbf{u}_0,
\]

(10.57)

named after the influential nineteenth century German analyst Carl Jacobi. The explicit form of the Jacobi iterative scheme is

\[
u_i^{(k+1)} = -\frac{1}{a_{ii}} \sum_{i \neq j = 1}^n a_{ij} u_j^{(k)} + \frac{b_i}{a_{ii}}.
\]

(10.58)

Let us rederive the Jacobi method in a direct matrix form. We begin by decomposing the coefficient matrix

\[
A = L + D + U,
\]

(10.59)

into the sum of a strictly lower triangular matrix \(L\), a diagonal matrix \(D\), and a strictly upper triangular matrix \(U\), each of which is uniquely specified. For example, in the case of the coefficient matrix

\[
A = \begin{pmatrix} 3 & 1 & -1 \\ 1 & -4 & 2 \\ -2 & -1 & 5 \end{pmatrix},
\]

(10.60)

the decomposition (10.59) yields

\[
L = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ -2 & -1 & 0 \end{pmatrix}, \quad D = \begin{pmatrix} 3 & 0 & 0 \\ 0 & -4 & 0 \\ 0 & 0 & 5 \end{pmatrix}, \quad U = \begin{pmatrix} 0 & 1 & -1 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{pmatrix}.
\]

**Warning:** The \(L, D, U\) in the elementary additive decomposition (10.59) have nothing to do with the \(L, D, U\) in factorizations arising from Gaussian elimination. The latter play no role in the iterative solution methods considered here.

We then rewrite the system

\[
A\mathbf{u} = (L + D + U)\mathbf{u} = \mathbf{b} \quad \text{in the alternative form} \quad D\mathbf{u} = -(L + U)\mathbf{u} + \mathbf{b}.
\]

The Jacobi fixed point equation amounts to solving for

\[
\mathbf{u} = T\mathbf{u} + \mathbf{c}, \quad \text{where} \quad T = -D^{-1}(L + U), \quad \mathbf{c} = D^{-1}\mathbf{b}.
\]

(10.61)
For the example (10.60), we recover the Jacobi iteration matrix by

\[ T = -D^{-1}(L + U) = \begin{pmatrix}
0 & -\frac{1}{3} & \frac{1}{3} \\
\frac{1}{4} & 0 & \frac{1}{2} \\
\frac{2}{5} & \frac{1}{5} & 0
\end{pmatrix}. \]

Deciding whether or not the Jacobi method converges for a specific matrix is not an easy task. However, it can be shown that Jacobi iteration will always converge for matrices that have large diagonal terms: the diagonally dominant matrices of Definition 10.33.

**Theorem 10.42.** If \( A \) is strictly diagonally dominant, then the associated Jacobi iteration scheme converges.

**Proof:** We shall prove that \( \|T\|_\infty < 1 \), and so Corollary 10.23 implies that \( T \) is a convergent matrix. The row sums of the Jacobi matrix \( T = -D^{-1}(L + U) \) are, according to (10.56),

\[ s_i = \sum_{j=1}^{n} |t_{ij}| = \frac{1}{|a_{ii}|} \sum_{i \neq j=1}^{n} |a_{ij}| < 1 \]

(10.62)

because \( A \) is strictly diagonally dominant. Thus, \( \|T\|_\infty = \max\{s_1, \ldots, s_n\} < 1 \), and the result follows.

**Q.E.D.**

**Example 10.43.** Consider the linear system

\[
\begin{align*}
4x + y + w &= 1, \\
x + 4y + z + v &= 2, \\
y + 4z + w &= -1, \\
x + z + 4w + v &= 2, \\
y + w + 4v &= 1.
\end{align*}
\]

The Jacobi method solves the respective equations for \( x, y, z, w, v \), leading to the iterative scheme

\[
\begin{align*}
x^{(k+1)} &= -\frac{1}{4} y^{(k)} - \frac{1}{4} w^{(k)} + 1, \\
y^{(k+1)} &= -\frac{1}{4} x^{(k)} - \frac{1}{4} z^{(k)} - \frac{1}{4} v^{(k)} + 2, \\
z^{(k+1)} &= -\frac{1}{4} y^{(k)} - \frac{1}{4} w^{(k)} - 1, \\
w^{(k+1)} &= -\frac{1}{4} x^{(k)} - \frac{1}{4} z^{(k)} - \frac{1}{4} v^{(k)} + 2, \\
v^{(k+1)} &= -\frac{1}{4} y^{(k)} - \frac{1}{4} w^{(k)} + 1.
\end{align*}
\]

The coefficient matrix of the original system

\[ A = \begin{pmatrix}
4 & 1 & 0 & 1 & 0 \\
1 & 4 & 1 & 0 & 1 \\
0 & 1 & 4 & 1 & 0 \\
1 & 0 & 1 & 4 & 1 \\
0 & 1 & 0 & 1 & 4
\end{pmatrix}. \]
is diagonally dominant, and so we are guaranteed that the Jacobi iterations will eventually converge to the solution. Indeed, the Jacobi scheme takes the iterative form (10.61), with

\[
T = \begin{pmatrix}
0 & -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \\
-\frac{1}{4} & 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\
0 & -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \\
-\frac{1}{4} & 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\
0 & -\frac{1}{4} & 0 & -\frac{1}{4} & 0
\end{pmatrix}, \quad c = \begin{pmatrix}
\frac{1}{2} \\
\frac{1}{2} \\
\frac{1}{2} \\
\frac{1}{2} \\
\frac{1}{2}
\end{pmatrix}.
\]

Note that \( \| T \|_\infty = \frac{3}{4} < 1 \), and hence the convergence rate of the iterates to the solution is at least \( 0.75 \), which slightly overestimates the true convergence rate, as determined by the spectral radius \( \rho(T) = 0.6124 \). To obtain four decimal place accuracy in the solution, we anticipate\footnote{If we were to use the matrix norm instead of the spectral radius, we would overestimate the proposed number of iterates to be \( \log(0.5 \times 10^{-4})/\log 0.75 \approx 34 \).} about \( \log(0.5 \times 10^{-4})/\log 0.6124 \approx 20 \) iterations. Indeed, starting with the initial guess \( x^{(0)} = y^{(0)} = z^{(0)} = w^{(0)} = v^{(0)} = 0 \), the Jacobi iterates converge to the exact solution \( x = -1 \), \( y = 0.7 \), \( z = -0.6 \), \( w = 0.7 \), \( v = -0.1 \), to four decimal places in exactly 20 iterations.

**The Gauss–Seidel Method**

The Gauss–Seidel method relies on a slightly more sophisticated implementation of the Jacobi process. To understand how it works, it will help to write out the Jacobi iteration scheme (10.57) in full detail:

\[
\begin{align*}
 u_1^{(k+1)} &= t_{12} u_2^{(k)} + t_{13} u_3^{(k)} + \cdots + t_{1,n-1} u_{n-1}^{(k)} + t_{1n} u_n^{(k)} + c_1, \\
 u_2^{(k+1)} &= t_{21} u_1^{(k)} + t_{23} u_3^{(k)} + \cdots + t_{2,n-1} u_{n-1}^{(k)} + t_{2n} u_n^{(k)} + c_2, \\
 u_3^{(k+1)} &= t_{31} u_1^{(k)} + t_{32} u_2^{(k)} + \cdots + t_{3,n-1} u_{n-1}^{(k)} + t_{3n} u_n^{(k)} + c_3, \\
 &\vdots \\
 u_n^{(k+1)} &= t_{n1} u_1^{(k)} + t_{n2} u_2^{(k)} + t_{n3} u_3^{(k)} + \cdots + t_{n,n-1} u_{n-1}^{(k)} + t_{nn} u_n^{(k)} + c_n,
\end{align*}
\]

where we are explicitly noting the fact that the diagonal entries of \( T \) vanish. Observe that we are using the entries of \( u^{(k)} \) to compute all of the updated values of \( u^{(k+1)} \). Presumably, if the iterates \( u^{(k)} \) are converging to the solution \( u^* \), then their individual entries are also converging, and so each \( u_j^{(k+1)} \) should be a better approximation to \( u_j^* \) than \( u_j^{(k)} \) is. Therefore, if we begin the \( k \)-th Jacobi iteration by computing \( u_1^{(k+1)} \) using the first equation, then we are tempted to use this new value instead of the previous, less accurate value \( u_1^{(k)} \) in each of the subsequent equations. In particular, we employ the modified equation

\[
 u_2^{(k+1)} = t_{21} u_1^{(k+1)} + t_{23} u_3^{(k)} + \cdots + t_{1n} u_n^{(k)} + c_2
\]
to update the second component of our iterate. This more accurate value should then be used to update $u^{(k+1)}_3$, and so on.

The upshot of these considerations is the Gauss–Seidel iteration scheme

$$u^{(k+1)}_i = t_{i1} u^{(k+1)}_1 + \cdots + t_{i,i-1} u^{(k+1)}_{i-1} + t_{i,i} u^{(k+1)}_{i+1} + \cdots + t_{in} u^{(k+1)}_n + c_i,$$

(10.64)

named after Gauss (as usual!) and the German astronomer/mathematician Philipp von Seidel. At the $k$th stage of the iteration, we use (10.64) to compute the updated entries $u^{(k+1)}_1$, $u^{(k+1)}_2$, \ldots, $u^{(k+1)}_n$ in their numerical order. Once an entry has been updated, the new value is immediately used in all subsequent computations.

**Example 10.44.** For the linear system

$$3x + y - z = 3, \quad x - 4y + 2z = -1, \quad -2x - y + 5z = 2,$$

the Jacobi iteration method was given in (10.51). To obtain the corresponding Gauss–Seidel scheme we use updated values of $x, y$ and $z$ as they become available. Explicitly,

$$x^{(k+1)} = -\frac{1}{3} y^{(k)} + \frac{1}{3} z^{(k)} + 1,$$
$$y^{(k+1)} = \frac{1}{4} x^{(k+1)} + \frac{1}{2} z^{(k)} + \frac{1}{4},$$
$$z^{(k+1)} = \frac{2}{5} x^{(k+1)} + \frac{1}{5} y^{(k+1)} + \frac{2}{5},$$

(10.65)

The resulting iterates starting with $u^{(0)} = 0$ are

$$u^{(1)} = \begin{pmatrix} 1.0000 \\ 0.5000 \\ 0.9000 \end{pmatrix}, \quad u^{(2)} = \begin{pmatrix} 1.1333 \\ 0.9833 \\ 1.0500 \end{pmatrix}, \quad u^{(3)} = \begin{pmatrix} 1.0222 \\ 1.0306 \\ 1.0150 \end{pmatrix}, \quad u^{(4)} = \begin{pmatrix} 0.9948 \\ 1.0062 \\ 0.9992 \end{pmatrix},$$

$$u^{(5)} = \begin{pmatrix} 0.9977 \\ 0.9990 \\ 0.9989 \end{pmatrix}, \quad u^{(6)} = \begin{pmatrix} 1.0000 \\ 0.9994 \\ 0.9999 \end{pmatrix}, \quad u^{(7)} = \begin{pmatrix} 1.0001 \\ 1.0000 \\ 1.0001 \end{pmatrix}, \quad u^{(8)} = \begin{pmatrix} 1.0000 \\ 1.0000 \\ 1.0000 \end{pmatrix}.$$

The iterations have converged to the solution, to 4 decimal places, after only 8 iterations — as opposed to the 11 iterations required by the Jacobi method. In this example, the Gauss–Seidel method is converging roughly 50% faster.

The Gauss–Seidel iteration scheme is particularly suited to implementation on a serial computer, since one can immediately replace each component $u^{(k)}_i$ by its updated value $u^{(k+1)}_i$, thereby also saving on storage in the computer’s memory. In contrast, the Jacobi scheme requires us to retain all the old values $u^{(k)}$ until all of the new values in $u^{(k+1)}$ have been computed. Moreover, Gauss–Seidel typically (although not always) converges faster than Jacobi, making it the iterative algorithm of choice for serial processors. On the other hand, with the advent of parallel processing machines, variants of the Jacobi scheme have been making a comeback. Whereas Gauss–Seidel necessitates performing only one entry update at a time, the Jacobi method can be more easily parallelized.

What is Gauss–Seidel really up to? Let us rewrite the basic iterative equation (10.64) by multiplying by $a_{ii}$ and moving the terms involving $u^{(k+1)}$ to the left hand side. In view
of the formula (10.56) for the entries of $T$, the resulting equation is
\[ a_{i1} u_1^{(k+1)} + \cdots + a_{i,i-1} u_{i-1}^{(k+1)} + a_{ii} u_i^{(k+1)} = -a_{i,i+1} u_{i+1}^{(k)} - \cdots - a_{in} u_n^{(k)} + b_i. \]

In matrix form, taking (10.59) into account, this reads
\[ (L + D)u^{(k+1)} = -U u^{(k)} + b, \]  
(10.66)

and so can be viewed as a linear system of equations for $u^{(k+1)}$ with lower triangular coefficient matrix $L+D$. Note that the fixed point of (10.66), namely $(L+D)u = -U u + b$, coincides with the solution to the original system $A u = (L + D + U) u = b$. The Gauss–Seidel procedure is merely implementing Forward Substitution to solve the lower triangular system (10.66) for the next iterate:
\[ u^{(k+1)} = (L + D)^{-1} U u^{(k)} + (L + D)^{-1} b. \]

The latter is in our more usual iterative form
\[ u^{(k+1)} = \tilde{T} u^{(k)} + \tilde{c}, \]
where $\tilde{T} = -(L + D)^{-1} U$, $\tilde{c} = (L + D)^{-1} b$.

Consequently, the convergence of the Gauss–Seidel iterates is governed by the spectral radius of its coefficient matrix $\tilde{T}$.

For example, in the case of the coefficient matrix in Example 10.44, we have
\[ A = \begin{pmatrix} 3 & 1 & -1 \\ 1 & -4 & 2 \\ -2 & -1 & 5 \end{pmatrix}, \quad L + D = \begin{pmatrix} 3 & 0 & 0 \\ 1 & -4 & 0 \\ -2 & -1 & 5 \end{pmatrix}, \quad U = \begin{pmatrix} 0 & 1 & -1 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{pmatrix}. \]

Therefore, the Gauss–Seidel coefficient matrix is
\[ \tilde{T} = -(L + D)^{-1} U = \begin{pmatrix} 0 & -0.3333 & 0.3333 \\ 0 & -0.0833 & 0.5833 \\ 0 & -0.1500 & 0.2500 \end{pmatrix}. \]

The matrix $\tilde{T}$ has eigenvalues 0 and $0.0833 \pm 0.2444i$, and hence its spectral radius is $\rho(\tilde{T}) \approx 0.2582$. This is roughly the square of the Jacobi spectral radius of 0.5, and tell us that the Gauss-Seidel iterations will converge about twice as fast to the solution, in accordance with our earlier observation. Indeed, although examples exist where the Jacobi method converges faster, in many practical situation, the Gauss–Seidel scheme tends to converge roughly twice as fast.

General conditions guaranteeing the convergence of the Gauss–Seidel method are hard to establish. But, like the Jacobi method, diagonally dominant matrices are still handled well.

**Theorem 10.45.** If $A$ is strictly diagonally dominant, then the Gauss–Seidel iteration scheme converges.
Proof: Let \( e^{(k)} = u^{(k)} - u^* \) denote the \( k \)-th Gauss–Seidel error vector. As in (10.54), the error vectors satisfy the homogeneous iteration \( e^{(k+1)} = T e^{(k)} \). We write out this equation in components:

\[
e^{(k+1)}_i = t_{i1} e^{(k+1)}_1 + \cdots + t_{i,i-1} e^{(k+1)}_{i-1} + t_{i,i+1} e^{(k)}_{i+1} + \cdots + t_{in} e^{(k)}_n.
\] (10.67)

Let

\[
m^{(k)} = \| e^{(k)} \|_\infty = \max \{ |e^{(k)}_1|, \ldots, |e^{(k)}_n| \}
\] (10.68)

denote the \( \infty \) norm of the error vector. We claim that diagonal dominance of \( A \) implies that

\[
m^{(k+1)} \leq s m^{(k)}, \quad \text{where} \quad s = \| T \|_\infty < 1
\] (10.69)

denotes the \( \infty \) matrix norm of the Jacobi matrix, which, by (10.62), is less than 1. We infer that \( m^{(k)} \leq s^k m^{(0)} \rightarrow 0 \) as \( k \rightarrow \infty \). Thus, \( e^{(k)} \rightarrow 0 \), demonstrating the theorem.

To prove (10.69), we use induction on \( i = 1, \ldots, n \). Our induction hypothesis is

\[
|e^{(k+1)}_j| \leq s m^{(k)} \leq m^{(k)} \quad \text{for} \quad j = 1, \ldots, i - 1.
\]

Moreover, by (10.68),

\[
|e^{(k)}_j| \leq m^{(k)} \quad \text{for all} \quad j = 1, \ldots, n.
\]

We use these two inequalities to estimate \( |e^{(k+1)}_i| \) from (10.67):

\[
|e^{(k+1)}_i| \leq |t_{i1}| |e^{(k)}_1| + \cdots + |t_{i,i-1}| |e^{(k)}_{i-1}| + |t_{i,i+1}| |e^{(k)}_{i+1}| + \cdots + |t_{in}| |e^{(k)}_n| \\
\leq \left( |t_{i1}| + \cdots + |t_{in}| \right) m^{(k)} = s_i m^{(k)} \leq s m^{(k)},
\]

which completes the induction step. As a result, the maximum

\[
m^{(k+1)} = \max \{ |e^{(k+1)}_1|, \ldots, |e^{(k+1)}_n| \} \leq s m^{(k)}
\]

also satisfies the same bound, and hence (10.69) follows. \( Q.E.D. \)

**Example 10.46.** For the linear system considered in Example 10.43, the Gauss–Seidel iterations take the form

\[
x^{(k+1)} = -\frac{1}{4} y^{(k)} - \frac{1}{4} w^{(k)} + 1, \\
y^{(k+1)} = -\frac{1}{4} x^{(k+1)} - \frac{1}{4} z^{(k)} - \frac{1}{4} v^{(k)} + 2, \\
z^{(k+1)} = -\frac{1}{4} y^{(k+1)} - \frac{1}{4} w^{(k)} - 1, \\
w^{(k+1)} = -\frac{1}{4} x^{(k+1)} - \frac{1}{4} z^{(k+1)} - \frac{1}{4} v^{(k)} + 2, \\
v^{(k+1)} = -\frac{1}{4} y^{(k+1)} - \frac{1}{4} w^{(k+1)} + 1.
\]

Starting with \( x^{(0)} = y^{(0)} = z^{(0)} = w^{(0)} = v^{(0)} = 0 \), the Gauss–Seidel iterates converge to the solution \( x = -1, y = .7, z = -6, w = .7, v = -.1 \), to four decimal places in 11 iterations, again roughly twice as fast as the Jacobi scheme.
Indeed, the convergence rate is governed by the corresponding Gauss-Seidel matrix $\tilde{T}$, which is

$$
\begin{pmatrix}
4 & 0 & 0 & 0 & 0 \\
1 & 4 & 0 & 0 & 0 \\
0 & 1 & 4 & 0 & 0 \\
1 & 0 & 1 & 4 & 0 \\
0 & 1 & 0 & 1 & 4
\end{pmatrix}
= 
\begin{pmatrix}
0 & 1 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 4
\end{pmatrix}

Its spectral radius is $\rho(\tilde{T}) = 0.3936$, which is, as in the previous example, approximately the square of the spectral radius of the Jacobi coefficient matrix. This explains its doubly fast rate of convergence.

**Successive Over-Relaxation (SOR)**

As we know, the smaller the spectral radius (or matrix norm) of the coefficient matrix, the faster the convergence of the iterative method. The goal of researchers in numerical linear algebra is to design new methods for accelerating the convergence. In his 1950 thesis, the American mathematician David Young discovered a simple modification of the Jacobi and Gauss-Seidel methods that can, in many common applications, lead to a dramatic speed up in the rate of convergence. The method, known as *successive over-relaxation*, and often abbreviated as SOR, has become the iterative method of choice in many modern applications. In this subsection, we give a brief overview of the SOR iterative scheme.

In practice, designing the optimal iterative algorithm to solve a given linear system is as hard a solving the system itself. Therefore, one relies on a few tried and true techniques for building a good iterative scheme that works in a range of examples. Every decomposition

$$
A = M - N
$$

of the coefficient matrix of the system $Au = b$ into the difference of two matrices leads to an equivalent system of the form

$$
Mu = Nu + b.
$$

Provided we take $M$ to be invertible, we can rewrite the system in the fixed point form

$$
u = M^{-1}Nu + M^{-1}b = Tu + c,$$

where $T = M^{-1}N$, $c = M^{-1}b$.

Now, we are free to choose any such $M$, which then specifies $N = A-M$ uniquely. However, for the resulting iterative scheme $u^{(k+1)} = T u^{(k)} + c$ to be practical we must arrange that

1. $T = M^{-1}N$ is a convergent matrix, and
2. $M$ can be easily inverted.

The second requirement ensures that the iterative equations

$$
Mu^{(k+1)} = Nu^{(k)} + b
$$

can be solved for $u^{(k+1)}$ with minimal computational effort. Typically, this requires that $M$ be either a diagonal matrix, in which case the inversion is immediate, or upper or lower triangular, in which case one employs back or forward substitution to solve for $u^{(k+1)}$.
With this in mind, we now introduce the SOR method. It relies on a slight general-
ization of the Gauss–Seidel decomposition (10.66) of the matrix into lower plus diagonal
and upper triangular parts. The starting point is to write
\[ A = L + D + U = \left[ L + \alpha D \right] - \left[ (\alpha - 1) D - U \right], \] (10.73)
where \( \alpha \) is an adjustable scalar parameter. We decompose the system \( Au = b \) as
\[ (L + \alpha D)u = \left[ (\alpha - 1) D - U \right]u + b. \] (10.74)
It turns out to be slightly more convenient to divide (10.74) through by \( \alpha \), and write the
resulting iterative system in the form
\[ \left( \omega L + D \right)u^{(k+1)} = \left[ (1 - \omega) D - \omega U \right]u^{(k)} + \omega b, \] (10.75)
where \( \omega = 1/\alpha \) is called the relaxation parameter. Assuming, as usual, that all diagonal
entries of \( A \) are nonzero, the matrix \( \omega L + D \) is an invertible lower triangular matrix, and
so we can use forward substitution to solve the iterative system (10.75) to recover \( u^{(k+1)} \).
The explicit formula for its \( i \)th entry is
\[ u_i^{(k+1)} = \omega t_{i1} u_1^{(k+1)} + \cdots + \omega t_{i,i-1} u_{i-1}^{(k+1)} + (1 - \omega) u_i^{(k)} + \]
\[ + \omega t_{i,i+1} u_{i+1}^{(k)} + \cdots + \omega t_{in} u_n^{(k)} + \omega c_i, \] (10.76)
where \( t_{ij} \) and \( c_i \) denote the original Jacobi values (10.56). As in the Gauss–Seidel approach,
we update the entries \( u_i^{(k+1)} \) in numerical order \( i = 1, \ldots, n \). Thus, to obtain the SOR
scheme (10.76), we merely multiply the right hand side of the Gauss–Seidel scheme (10.64)
by the adjustable relaxation parameter \( \omega \) and append the diagonal term \( (1 - \omega) u_i^{(k)} \).
In particular, if we set \( \omega = 1 \), then the SOR method reduces to the Gauss–Seidel method.
Choosing \( \omega < 1 \) leads to an under-relaxed method, while \( \omega > 1 \), known as over-relaxation,
is the choice that works in most practical instances.

To analyze the convergence rate of the SOR scheme (10.75), we rewrite it in the fixed
point form
\[ u^{(k+1)} = T_\omega u^{(k)} + c_\omega, \] (10.77)
where
\[ T_\omega = (\omega L + D)^{-1} \left[ (1 - \omega) D - \omega U \right], \quad c_\omega = (\omega L + D)^{-1} \omega b. \] (10.78)
The rate of convergence of the SOR method is governed by the spectral radius of its
coefficient matrix \( T_\omega \). The goal is to choose the relaxation parameter \( \omega \) so as to make the
spectral radius of \( T_\omega \) as small as possible. As we will see, a clever choice of \( \omega \) will result
in a dramatic speed up in the convergence of the iterative method. Before stating some
general facts (albeit without proof) let us analyze a simple example.

**Example 10.47.** Consider the matrix \( A = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix} \), which we write as \( L+D+U \),
where
\[ L = \begin{pmatrix} 0 & 0 \\ -1 & 0 \end{pmatrix}, \quad D = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}, \quad U = \begin{pmatrix} 0 & -1 \\ 0 & 0 \end{pmatrix}. \]
Jacobi iteration uses the coefficient matrix $T = -D^{-1}(L + U) = \begin{pmatrix} 0 & \frac{1}{3} \\ \frac{2}{3} & 0 \end{pmatrix}$. The Jacobi spectral radius is $\rho(T) = .5$, and hence it takes, on average, roughly $3.3 \approx -1/\log_{10}.5$ iterations to produce each new decimal place of accuracy in the solution.

The SOR scheme (10.75) takes the explicit form

$$
\begin{pmatrix} 2 & 0 \\ -\omega & 2 \end{pmatrix} u^{(k+1)} = \begin{pmatrix} 2(1 - \omega) & \omega \\ 0 & 2(1 - \omega) \end{pmatrix} u^{(k)} + \omega b,
$$

where Gauss–Seidel is the particular case $\omega = 1$. The SOR coefficient matrix is

$$
T_\omega = \begin{pmatrix} 2 & 0 \\ -\omega & 2 \end{pmatrix}^{-1} \begin{pmatrix} 2(1 - \omega) & \omega \\ 0 & 2(1 - \omega) \end{pmatrix} = \begin{pmatrix} 1 - \omega & \frac{1}{2} \omega \\ \frac{1}{2} \omega(1 - \omega) & \frac{1}{4}(2 - \omega)^2 \end{pmatrix}.
$$

To compute the eigenvalues of $T_\omega$, we form its characteristic equation

$$
0 = \det(T_\omega - \lambda I) = \lambda^2 - (2 - 2\omega + \frac{1}{2} \omega^2)\lambda + (1 - \omega)^2 = (\lambda + \omega - 1)^2 - \frac{1}{4}\lambda \omega^2.
$$

(10.79)

Our goal is to choose $\omega$ so that

(a) Both eigenvalues are less than 1 in modulus, so $|\lambda_1|, |\lambda_2| < 1$. This is the minimal requirement for convergence of the method.

(b) The largest eigenvalue (in modulus) is as small as possible. This will give the smallest spectral radius for $T_\omega$ and hence the fastest convergence rate.

By (8.26), the product of the two eigenvalues is the determinant,

$$
\lambda_1 \lambda_2 = \det T_\omega = (1 - \omega)^2.
$$

If $\omega \leq 0$ or $\omega \geq 2$, then $\det A \geq 1$, and hence least one of the eigenvalues would have modulus larger than 1. Thus, in order to ensure convergence, we must require $0 < \omega < 2$. For Gauss–Seidel, at $\omega = 1$, the eigenvalues are $\lambda_1 = 0$, $\lambda_2 = \frac{1}{4}$, and the spectral radius is $\rho(T_1) = .25$. This is exactly the square of the Jacobi spectral radius, and hence the Gauss–Seidel iterates converge twice as fast — it only takes, on average, about 1.65 Gauss–Seidel iterations to produce a new decimal place of accuracy. It can be shown (Exercise 2) that as $\omega$ increases above 1, the two eigenvalues move together, the larger one decreasing in size. They are equal when

$$
\omega = \omega_* = 8 - 4\sqrt{3} \approx 1.07.
$$

At that point, $\lambda_1 = \lambda_2 = .07 = \rho(T_\omega)$, which is the convergence rate of the optimal SOR scheme. Each iteration produces slightly more than one new decimal place in the solution, which represents a significant improvement over the Gauss–Seidel convergence rate of .25. It takes about twice as many Gauss–Seidel iterations (and four times as many Jacobi iterations) to produce the same accuracy as this optimal SOR method.

† In Exercise 2, the reader is asked to complete the proof of optimality.
Of course, in such a simple $2 \times 2$ example, it is not so surprising that we can construct the optimal relaxation parameter by hand. In his 1950 thesis, cf. [154], Young found the optimal value of the relaxation parameter for a broad class of matrices that includes most of those arising in the finite difference and finite element numerical solutions to ordinary and partial differential equations. For the matrices in Young’s class, the Jacobi eigenvalues occur in signed pairs. If $\pm \mu$ are a pair of eigenvalues for the Jacobi method, then the corresponding eigenvalues of the SOR iteration matrix satisfy the quadratic equation

$$\left(\lambda + \omega - 1\right)^2 = \lambda \omega^2 \mu^2. \tag{10.80}$$

If $\omega = 1$, so we have standard Gauss–Seidel, then $\lambda^2 = \lambda \mu^2$, and so the corresponding Gauss–Seidel eigenvalues are $\lambda = 0, \lambda = \mu^2$. The Gauss–Seidel spectral radius is therefore the square of the Jacobi spectral radius, and so (at least for matrices in the Young class) its iterates converge twice as fast. The quadratic equation (10.80) has the same properties as in the $2 \times 2$ version (10.79) (which corresponds to the case $\mu = \frac{1}{2}$), and hence the optimal value of $\omega$ will be the one at which the two roots are equal,

$$\lambda_1 = \lambda_2 = \omega - 1, \quad \text{which occurs when} \quad \omega = \frac{2 - 2\sqrt{1 - \mu^2}}{\mu^2} = \frac{2}{1 + \sqrt{1 - \mu^2}}. \tag{10.81}$$

Therefore, if $\rho_J = \max |\mu|$ denotes the spectral radius of the Jacobi method, then the Gauss–Seidel has spectral radius $\rho_{GS} = \rho_J^2$, while the SOR method with optimal relaxation parameter

$$\omega_* = \frac{2}{1 + \sqrt{1 - \rho_J^2}}, \quad \text{has spectral radius} \quad \rho_* = \omega_* - 1. \tag{10.81}$$

For example, if $\rho_J = .99$, which is quite slow convergence (but common for iterative solutions of partial differential equations), then $\rho_{GS} = 0.9801$, which is twice as fast, but still quite slow, while SOR with $\omega_* = 1.7527$ has $\rho_* = 0.7527$, which is dramatically faster. Indeed, since $\rho_* \approx (\rho_{GS})^{14} \approx (\rho_J)^{28}$, it takes about 14 Gauss–Seidel (and 28 Jacobi) iterations to produce the same accuracy as one SOR step. The fact that such a simple idea can have such a dramatic effect on the convergence rate is amazing.

Conjugate Gradients

So far, we have established two broad classes of algorithms for solving linear systems. The first, the direct methods, based on some version of Gaussian elimination or matrix factorization, eventually obtain the solution, but must be carried through to completion before any useful information is obtained. The alternative, iterative methods discussed in the present chapter, lead to closer and closer approximations of the solution, but never reach the actual value exactly. One might ask whether there are algorithms that combine the best of both: semi-direct methods that give closer and closer approximations to the solution, but are guaranteed to eventually terminate with the exact solution in hand.

† This assumes that we are dealing with a fully accurate implementation, i.e., without round-off or other numerical error. For this discussion, numerical instability will be left aside as a separate, albeit ultimately important, issue.
For instance, one might ask for an algorithm that successively computes each entry of the solution vector $u_\ast$. This seems unlikely, but if we recall that the entries of the solution are merely its coordinates with respect to the standard basis $e_1, \ldots, e_n$, then one might try instead to compute the coordinates $t_1, \ldots, t_n$ of $u_\ast = t_1 v_1 + \cdots + t_n v_n$ with respect to some basis that is specially adapted to the linear system. Ideally, $v_1, \ldots, v_n$ should be an orthogonal basis — but orthogonality with respect to the standard Euclidean dot product is not typically relevant. A better idea is to arrange that the basis be orthogonal with respect to an inner product that is adapted to the system under consideration. In particular, if the linear system to be solved takes the form

$$K u = f,$$

in which the coefficient matrix is positive definite, as occurs in many applications, then orthogonality with respect to the induced inner product

$$\langle v, w \rangle = v^T K w = v \cdot K w$$

is very natural. Vectors that are orthogonal with respect to the inner product induced by the coefficient matrix $K$ are known as conjugate vectors, which explain half the name of the conjugate gradient algorithm, first introduced by Hestenes and Stiefel, [78].

The term “gradient” stems from the minimization principle. According to Theorem 4.1, the solution $u_\ast$ to the positive definite linear system (10.82) is the unique minimizer of the quadratic function

$$p(u) = \frac{1}{2} u^T K u - u^T f.$$ 

Thus, one way to solve the system is to minimize $p(u)$. Suppose we find ourselves at a point $u$ which is not the minimizer. In which direction should we travel to find $u_\ast$? The key result from multivariable calculus is that the gradient vector $\nabla p(u)$ of a (nonlinear) function points in the direction of its steepest increase at the point, while its negative $-\nabla p(u)$ points in the direction of steepest decrease. Our discussion of gradient flow systems (9.19) used the same idea; full details appear in Section 19.3. For the particular quadratic function (10.84), its negative gradient is easily found:

$$-\nabla p(u) = f - K u = r,$$

where $r$ is known as the residual vector for the point $u$. Note that $r = 0$ if and only if $u = u_\ast$ is the solution, and so the size of $r$ measures, in a certain sense, how accurately $u$ comes to solving the system. Moreover, the residual vector indicates the direction of steepest decrease in the quadratic function, and is thus a good choice of direction to head off in search of the true minimizer.

The initial result is the gradient descent algorithm, in which each successive approximation $u_k$ to the solution is obtained by going a certain distance in the residual direction:

$$u_{k+1} = u_k + t_k r_k,$$

where

$$r_k = f - K u_k.$$ 

† Here, we include an irrelevant factor of $\frac{1}{2}$ for later convenience.
The scalar factor \( t_k \) can be specified by the requirement that \( p(u_{k+1}) \) is as small as possible; in Exercise [exercise number] you are asked to find this value. A second option is to make the residual vector at \( u_{k+1} \) as small as possible. The initial guess \( u_0 \) for the solution can be chosen as desired, with \( u_0 = 0 \) the default choice. Gradient descent is a reasonable algorithm, and will lead to the solution in favorable situations. It is also used to minimize more general nonlinear functions. However, in many circumstances, the iterative method based on gradient descent can take an exceedingly long time to converge to an accurate approximation to the solution, and so is often not a competitive algorithm.

However, if we supplement the gradient descent idea by the use of conjugate vectors, we are led to a very powerful semi-direct solution algorithm. We shall construct the solution \( u_k \) by successive approximation, with the \( k \)th iterate having the form

\[
  u_k = t_1 v_1 + \cdots + t_k v_k,
\]

so that 

\[
  u_{k+1} = u_k + t_{k+1} v_{k+1},
\]

where, as advertised, the conjugate vectors \( v_1, \ldots, v_n \) form a \( K \)-orthogonal basis. The secret is not to try to specify the conjugate basis vectors in advance, but rather to successively construct them during the course of the algorithm. We begin, merely for convenience, with an initial guess \( u_0 = 0 \) for the solution. The residual vector \( r_0 = f - K u_0 = f \) indicates the direction of steepest decrease of \( p(u) \) at \( u_0 \), and we update our original guess by moving in this direction, taking \( v_1 = r_0 = f \) as our first conjugate direction. The next iterate is 

\[
  u_1 = u_0 + t_1 v_1 = t_1 v_1,
\]

and we choose the parameter \( t_1 \) so that the corresponding residual vector

\[
  r_1 = f - K u_1 = r_0 - t_1 K v_1
\]

is as close to \( 0 \) (in the Euclidean norm) as possible. This occurs when \( r_1 \) is orthogonal to \( r_0 \) (why?), and so we require

\[
  0 = r_0 \cdot r_1 = \| r_0 \|^2 - t_1 r_0 \cdot K v_1 = \| r_0 \|^2 - t_1 \langle r_0 ; v_1 \rangle = \| r_0 \|^2 - t_1 \langle v_1 ; v_1 \rangle.
\]

Therefore we set

\[
  t_1 = \frac{\| r_0 \|^2}{\langle v_1 ; v_1 \rangle}
\]

and so 

\[
  u_1 = u_0 + \frac{\| r_0 \|^2}{\langle v_1 ; v_1 \rangle} v_1
\]

is the new approximation to the solution.

**Note:** We will consistently use \( \| v \| \) to denote the standard Euclidean norm, and \( \langle v ; w \rangle \) the adapted inner product (10.83), which has its own norm \( \sqrt{\langle v ; v \rangle} \).

The gradient descent algorithm would tell us to update \( u_1 \) by moving in the residual direction \( r_1 \). But in the conjugate gradient algorithm, we choose a direction \( v_2 \) which is conjugate, meaning \( K \)-orthogonal to the first direction \( v_1 = r_0 \). Thus, we slightly modify the residual direction by setting \( v_2 = r_1 + s_1 v_1 \), where the scalar factor \( s_1 \) is determined by the orthogonality requirement

\[
  0 = \langle v_2 ; v_1 \rangle = \langle v_2 ; r_1 + s_1 v_1 \rangle = \langle r_1 ; v_1 \rangle + s_1 \langle v_1 ; v_1 \rangle,
\]

so 

\[
  s_1 = -\frac{\langle r_1 ; v_1 \rangle}{\langle v_1 ; v_1 \rangle}.
\]
Now, using (10.87) twice,
\[
\left\langle r_1 : v_1 \right\rangle = r_1 \cdot K v_1 = \frac{1}{t_1} r_1 \cdot (r_0 - r_1) = -\frac{1}{t_1} \| r_1 \|^2,
\]
\[
\left\langle v_1 : v_1 \right\rangle = v_1 \cdot K v_1 = \frac{1}{t_1} \| r_0 \|^2,
\]
and therefore the second conjugate direction is
\[
v_2 = r_1 + s_1 v_1, \quad \text{where} \quad s_1 = \frac{\| r_1 \|^2}{\| r_0 \|^2}.
\]
We then update
\[
u_2 = u_1 + t_2 v_2 = t_1 v_1 + t_2 v_2
\]
so as to make the corresponding residual vector
\[
r_2 = f - K u_2 = r_1 - t_2 K v_2
\]
as small as possible, which is accomplished by requiring it to be orthogonal to \( r_1 \). Thus, using the \( K \)–orthogonality of \( v_1 \) and \( v_2 \),
\[
0 = r_1 \cdot r_2 = \| r_1 \|^2 - t_2 r_1 \cdot K v_2 = \| r_1 \|^2 - t_2 \left\langle v_2 : v_2 \right\rangle, \quad \text{and so} \quad t_2 = \frac{\| r_1 \|^2}{\left\langle v_2 : v_2 \right\rangle}.
\]
Continuing in this manner, at the \( k \)th stage, we have already constructed the conjugate vectors \( v_1, \ldots, v_k \), and the solution approximation \( u_k \) as a suitable linear combination of them. The next conjugate direction is given by
\[
v_{k+1} = r_k + s_k v_k, \quad \text{where} \quad s_k = \frac{\| r_k \|^2}{\| r_{k-1} \|^2} \quad (10.89)
\]
is a result of the \( K \)–orthogonality requirement \( \left\langle v_i : v_k \right\rangle = 0 \) for \( i < k \). The updated solution approximation
\[
u_{k+1} = u_k + t_{k+1} v_{k+1} \quad \text{where} \quad t_{k+1} = \frac{\| r_k \|^2}{\left\langle v_{k+1} : v_{k+1} \right\rangle} \quad (10.90)
\]
is then fixed so as to make the corresponding residual \( r_{k+1} = f - K u_{k+1} = r_k - t_{k+1} K v_{k+1} \) as small as possible, which requires that it be orthogonal to \( r_k \). Starting with an initial guess \( u_0 \), the iterative equations (10.89), (10.90) implement the complete conjugate gradient algorithm. Note that the only matrix operation required is a multiplication \( K v_k \) in the computation of \( t_k \); all the other operations are fast Euclidean dot products. Unlike Gaussian elimination, the method produces a sequence of successive approximations \( u_1, u_2, \ldots \) to the solution \( u^* \), and so the method can be stopped when a desired solution accuracy is reached. On the other hand, unlike purely iterative methods, the algorithm does eventually terminate at the exact solution, because, as remarked at the outset, there are at most \( n \) conjugate directions, forming an \( K \) orthogonal basis of \( \mathbb{R}^n \). Therefore,
Conjugate Gradient Method for Solving $Ku = f$

\begin{verbatim}
start
    choose $u_0$, e.g. $u_0 = 0$
    for $k = 1$ to $m$
        set $r_{k-1} = f - Ku_{k-1}$
        if $k = 1$ set $v_1 = r_0$
        else set $v_k = r_{k-1} + \frac{\|r_{k-1}\|^2}{\|r_{k-2}\|^2} v_{k-1}$
        set $u_k = u_{k-1} + \frac{\|r_{k-1}\|^2}{v_k \cdot K v_k} v_k$
    next $k$

$u_n = t_1 v_1 + \cdots + t_n v_n = u_*$ must be the solution since its residual $r_n = f - Ku_n$ is orthogonal to all the $v_i$, and hence equal to $0$.

A pseudocode program is attached; at each stage $u_k$ represents the updated approximation to the solution. The initial guess $u_0$ can be chosen by the user, with $u_0 = 0$ the default. The iteration number $m \leq n$ can be chosen by the user in advance; alternatively, one can impose a stopping criterion based on the size of the residual vector, $\|r_{k-1}\|$, or, alternatively, the distance between successive iterates, $\|u_k - u_{k-1}\|$. If the process is carried on to the bitter end, i.e., for $m = n$, then, in the absence of round-off errors, the result is the exact solution to the system.

Example 10.48. Consider the linear system $Ku = f$ with

$$K = \begin{pmatrix} 3 & -1 & 0 \\ -1 & 2 & 1 \\ 0 & 1 & 1 \end{pmatrix}, \quad f = \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix}.$$ 

The exact solution is $u_* = (2, 5, -6)^T$. In order to implement the method of conjugate gradients, we start with the initial guess $u_0 = (0, 0, 0)^T$. The corresponding residual vector is merely $r_0 = f - Ku_0 = (1, 2, -1)^T$. The first conjugate direction is $v_1 = r_0 = f = (1, 2, -1)^T$, and we use (10.88) to obtain the updated approximation to the solution

$$u_1 = u_0 + \frac{\|r_0\|^2}{\langle v_1, v_1 \rangle} v_1 = \frac{6}{4} \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix} = \begin{pmatrix} 3/2 \\ 3/2 \\ -3/2 \end{pmatrix},$$

noting that $\langle v_1, v_1 \rangle = v_1 \cdot K v_1 = 4$. In the next stage of the algorithm, we compute the
corresponding residual \( r_1 = f - Ku_1 = (-\frac{1}{2}, -1, -\frac{5}{2})^T \). The conjugate direction is

\[
v_2 = r_1 + \frac{\|r_1\|^2}{\|r_0\|^2} v_1 = \begin{pmatrix} -\frac{1}{2} \\ -1 \\ -\frac{5}{2} \end{pmatrix} + \frac{15}{6} \begin{pmatrix} 1 \\ 2 \\ -1 \end{pmatrix} = \begin{pmatrix} \frac{3}{2} \\ \frac{3}{2} \\ -\frac{15}{4} \end{pmatrix}.
\]

We note that, as designed, it satisfies the conjugacy condition \( \langle v_1; v_2 \rangle = v_1 \cdot Ku_2 = 0 \).

Each entry of the new approximation

\[
u_2 = u_1 + \frac{\|r_1\|^2}{\langle v_2; v_2 \rangle} v_2 = \begin{pmatrix} \frac{3}{2} \\ 3 \\ -\frac{3}{2} \end{pmatrix} + \frac{15}{27} \begin{pmatrix} \frac{3}{2} \\ \frac{3}{2} \\ -\frac{15}{4} \end{pmatrix} = \begin{pmatrix} \frac{7}{3} \\ \frac{14}{3} \\ -\frac{17}{3} \end{pmatrix} = \begin{pmatrix} 2.333 \ldots \\ 4.666 \ldots \\ -5.666 \ldots \end{pmatrix}
\]

is within a \( \frac{1}{3} \) of the exact solution \( u_* \).

Since we are dealing with a \( 3 \times 3 \) system, we will recover the exact solution by one more iteration of the algorithm. The new residual is \( r_2 = f - Ku_2 = (-\frac{4}{3}, \frac{2}{3}, 0)^T \). The last conjugate direction is

\[
v_3 = r_2 + \frac{\|r_2\|^2}{\|r_1\|^2} v_2 = \begin{pmatrix} -\frac{4}{3} \\ \frac{2}{3} \\ 0 \end{pmatrix} + \frac{20}{15} \begin{pmatrix} \frac{3}{4} \\ \frac{3}{2} \\ -\frac{15}{4} \end{pmatrix} = \begin{pmatrix} -\frac{10}{9} \\ -\frac{10}{9} \\ -\frac{10}{9} \end{pmatrix},
\]

which, as you can check, is conjugate to both \( v_1 \) and \( v_2 \). The solution is obtained from

\[
u_3 = u_2 + \frac{\|r_2\|^2}{\langle v_3; v_3 \rangle} v_3 = \begin{pmatrix} \frac{7}{3} \\ \frac{14}{3} \\ -\frac{17}{3} \end{pmatrix} + \frac{20}{27} \begin{pmatrix} -\frac{10}{9} \\ \frac{10}{9} \\ -\frac{10}{9} \end{pmatrix} = \begin{pmatrix} 2 \\ 5 \\ -6 \end{pmatrix}.
\]

Of course, in larger examples, one would not carry through the algorithm to the bitter end — indeed reverting to ordinary Gaussian elimination is probably a better strategy in that case — since a decent approximation to the solution is typically obtained with only a few iterations. The result is a substantial saving in computational time and effort to produce a reasonable approximation to the solution.


The importance of the eigenvalues of a square matrix for both continuous and discrete dynamical systems has been amply demonstrated in this chapter and its predecessor. However, finding the eigenvalues and associated eigenvectors is not an easy task. The classical method of constructing the characteristic equation of the matrix through the determinantal formula, then solving the resulting polynomial equation for the eigenvalues, and finally producing the eigenvectors by solving the associated homogeneous linear system, is hopelessly inefficient, fraught with difficulty and numerical dangers. We are in need of a completely new approach if we have any hopes of designing efficient, numerical approximation schemes for computing eigenvalues and eigenvectors.
In this section, we present a few of the most basic numerical schemes for accurately computing eigenvalues and eigenvectors of matrices. The most direct are based on the connections between the eigenvalues and the high powers of a matrix. A more sophisticated technique is based on the $QR$ factorization that we learned in Section 5.3, and will be presented at the end of the section.

**The Power Method**

We have already noted the role played by the eigenvalues and eigenvectors in the solution to linear iterative systems. Now we are going to turn the tables, and use the iterative system as a mechanism for approximating the eigenvalues, or, more correctly, selected eigenvalues of the coefficient matrix. The resulting computational procedure is known as the *power method*.

We assume, for simplicity, that $A$ is a complete $n \times n$ matrix. Let $v_1, \ldots, v_n$ denote the eigenvector basis, and $\lambda_1, \ldots, \lambda_n$ the corresponding eigenvalues. As we have learned, the solution to the linear iterative system

$$ v^{(k+1)} = Av^{(k)}, \quad v^{(0)} = v, \tag{10.91} $$

is obtained by multiplying the initial vector $v$ by the successive powers of the coefficient matrix: $v^{(k)} = A^k v$. If we write the initial vector in terms of the eigenvector basis

$$ v = c_1 v_1 + \cdots + c_n v_n, \tag{10.92} $$

then the solution takes the explicit form given in Theorem 10.4, namely

$$ v^{(k)} = A^k v = c_1 \lambda_1^k v_1 + \cdots + c_n \lambda_n^k v_n. \tag{10.93} $$

Suppose further that $A$ has a single dominant *real* eigenvalue, $\lambda_1$, that is larger than any other in magnitude, so

$$ |\lambda_1| > |\lambda_j| \quad \text{for all} \quad j > 1. \tag{10.94} $$

The largest eigenvalue will completely dominate the iteration (10.93). Indeed, since

$$ |\lambda_1|^k \gg |\lambda_j|^k \quad \text{for all} \quad j > 1 \quad \text{and all} \quad k \gg 0, $$

the first term in the iterative formula (10.93) will eventually be much larger than all the rest, and so, provided $c_1 \neq 0$,

$$ v^{(k)} \approx c_1 \lambda_1^k v_1 \quad \text{for} \quad k \gg 0. $$

Therefore, under the assumption (10.94), the solution to the iterative system (10.91) will, almost always, end up being a multiple of the first eigenvector of the coefficient matrix.

---

$\dagger$ This is not a very severe restriction. Most matrices are complete. Moreover, perturbations caused by numerical inaccuracies will almost always make an incomplete matrix complete.
Furthermore, the entries of \( v^{(k)} \) are given by \( v_i^{(k)} = \lambda_i^k v_i \). Hence, for any nonzero eigenvector component \( v_i \neq 0 \), we can recover the eigenvalue \( \lambda_i \) itself by taking a ratio between the \( i \)th components of successive iterates:

\[
\lambda_i \approx \frac{v_i^{(k+1)}}{v_i^{(k)}}, \quad \text{provided} \quad v_i^{(k)} \neq 0. \tag{10.95}
\]

**Example 10.49.** Consider the matrix

\[
A = \begin{pmatrix}
-1 & 2 & 2 \\
-1 & -4 & -2 \\
-3 & 9 & 7
\end{pmatrix}.
\]

As the reader can check, its eigenvalues and eigenvectors are

\[
\lambda_1 = 3, \quad v_1 = \begin{pmatrix} 1 \\ -1 \\ 3 \end{pmatrix}, \quad \lambda_2 = -2, \quad v_2 = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}, \quad \lambda_3 = 1, \quad v_3 = \begin{pmatrix} -1 \\ 1 \\ -2 \end{pmatrix}.
\]

Repeatedly multiplying the particular initial vector \( v = (1, 0, 0)^T \) by \( A \) results in the vectors \( v^{(k)} = A^k v \) listed in the accompanying table. The last column indicates the ratio \( v_1^{(k)} \) and \( v_1^{(k-1)} \) between the first components of successive iterates. (One could equally well use the second or third components.) These ratios are converging to the third and largest eigenvalue \( \lambda_3 = 3 \), while \( v^{(k)} \) is converging to a very large multiple of the corresponding eigenvector \( v_3 \).

The success of the power method requires that \( A \) have a unique dominant eigenvalue of maximal modulus, which, by definition, equals its spectral radius: \( |\lambda_1| = \rho(A) \). The rate of convergence of the method is governed by the ratio \( |\lambda_2/\lambda_1| \) between the subdominant
and dominant eigenvalues. Thus, the further the dominant eigenvalue lies away from the rest, the faster the power method converges.

Since complex eigenvalues of real matrices come in complex conjugate pairs, of the same modulus, matrices whose dominant eigenvalue is complex are not covered by the method. Indeed, one could hardly expect to compute a complex eigenvalue as a ratio of real vectors! However, a slightly more sophisticated version of the method can handle the cases when there is a single complex-conjugate pair of dominant eigenvalues; see Exercise 10.50. We also assumed that the initial vector $v^{(0)}$ includes a nonzero multiple of the dominant eigenvector, i.e., $c_1 \neq 0$. As we do not know the eigenvectors, it is not so easy to guarantee this in advance, although one must be quite unlucky to make such a poor choice of initial vector. (Of course, the stupid choice $v^{(0)} = 0$ is not counted.) Moreover, even if $c_1$ happens to be 0 initially, numerical round-off error will typically come to one’s rescue, since it will almost inevitably introduce a tiny component of the eigenvector $v_1$ into some iterate, and this component will eventually dominate the computation. The trick is to wait long enough for it to show up!

Since the iterates of $A$ are, typically, getting either very large — when $\rho(A) > 1$ — or very small — when $\rho(A) < 1$ — the iterated vectors will be increasingly subject to round-off error, if not numerical over- or under-flow. One way to avoid this problem is to work with unit vectors $u^{(k)} = \|v^{(k)}\|^{-1}v^{(k)}$, whose entries cannot get too large, and so are less likely to cause numerical errors in the computations. Here $\| \cdot \|$ is any convenient norm — the 1 and $\infty$ norms being slightly easier to compute than the Euclidean norm. The unit vectors $u^{(k)}$ can be computed directly by the iterative scheme

$$u^{(0)} = \frac{v^{(0)}}{\|v^{(0)}\|}, \quad \text{and} \quad u^{(k+1)} = \frac{Au^{(k)}}{\|Au^{(k)}\|}.$$  \hspace{1cm} (10.96)

If the largest eigenvalue $\lambda_1 > 0$ is positive, then $u^{(k)} \rightarrow u_1$ will converge to one of the two unit eigenvectors (the other is $-u_1$) corresponding to the eigenvalue $\lambda_1$. If $\lambda_1 < 0$, then the iterates will switch back and forth between the two eigenvectors $u^{(k)} \approx (-1)^k u_1$. In either case, the eigenvalue $\lambda_1$ is obtained as a limiting ratio between nonzero entries of $u^{(k)}$ and $Au^{(k)}$. If some other behavior is observed, it means that one of our assumptions is not valid; either $A$ has more than one dominant eigenvalue of maximum modulus, or it is not complete.

**Example 10.50.** For the matrix considered in Example 10.49, if we multiply the initial vector $u^{(k)} = (1, 0, 0)^T$ by $A$, the resulting unit vectors $u^{(k)} = Au^{(k-1)}/\|Au^{(k-1)}\|$ for the Euclidean norm are given in the table. The last column, being the ratio between the first components of $u^{(k)}$ and $Au^{(k)}$, converges to the dominant eigenvalue $\lambda_1 = 3$.

Variants of the power method for computing the other eigenvalues of the matrix are explored in the exercises.

*The QR Algorithm*

As stated, the power method only produces the largest eigenvalue of a matrix $A$. The inverse power method of Exercise 10.48 can be used to find the smallest eigenvalue. Additional eigenvalues can be found by using the shifted inverse power method of Exercise 10.50 or the
The most important scheme for simultaneously approximating all the eigenvalues of a matrix $A$ is the remarkable $QR$ algorithm, first proposed in 1961, by Francis, [61], and Kublanovskaya, [99]. The underlying idea is simple but surprising. The first step is to factor the matrix

$$A = A_0 = Q_0 R_0$$

into a product of an orthogonal matrix $Q_0$ and a positive upper triangular matrix (i.e., with all positive entries along the diagonal) $R_0$ using the Gram–Schmidt orthogonalization procedure of Theorem 5.24. Next, multiply the two factors together in the wrong order! The result is the new matrix

$$A_1 = R_0 Q_0.$$

We then repeat these two steps. Thus, we next factor

$$A_1 = Q_1 R_1$$

using the Gram–Schmidt process, and then multiply the factors in the reverse order to produce $A_2 = R_2 Q_2$. The general algorithm is

$$A = Q_0 R_0, \quad A_{k+1} = R_k Q_k = Q_{k+1} R_{k+1}, \quad k = 0, 1, 2, \ldots, \quad (10.97)$$

where $Q_k, R_k$ come from the previous step, and the subsequent orthogonal matrix $Q_{k+1}$ and positive upper triangular matrix $R_{k+1}$ are computed using the numerically stable form of the Gram–Schmidt algorithm.

The astonishing fact is that, for many matrices $A$, the iterates $A_k \rightarrow V$ converge to a upper triangular matrix $V$ whose diagonal entries are the eigenvalues of $A$. Thus, after

<table>
<thead>
<tr>
<th>$k$</th>
<th>$u^{(k)}$</th>
<th>$\lambda$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-0.3015</td>
<td>-0.3015</td>
</tr>
<tr>
<td>2</td>
<td>-0.2335</td>
<td>0.3669</td>
</tr>
<tr>
<td>3</td>
<td>-0.3319</td>
<td>0.2257</td>
</tr>
<tr>
<td>4</td>
<td>-0.2788</td>
<td>0.3353</td>
</tr>
<tr>
<td>5</td>
<td>-0.3159</td>
<td>0.2740</td>
</tr>
<tr>
<td>6</td>
<td>-0.2919</td>
<td>0.3176</td>
</tr>
<tr>
<td>7</td>
<td>-0.3080</td>
<td>0.2899</td>
</tr>
<tr>
<td>8</td>
<td>-0.2973</td>
<td>0.3089</td>
</tr>
<tr>
<td>9</td>
<td>-0.3044</td>
<td>0.2965</td>
</tr>
<tr>
<td>10</td>
<td>-0.2996</td>
<td>0.3048</td>
</tr>
<tr>
<td>11</td>
<td>-0.3028</td>
<td>0.2993</td>
</tr>
<tr>
<td>12</td>
<td>-0.3007</td>
<td>0.3030</td>
</tr>
</tbody>
</table>
a sufficient number of iterations, say \( k \), the matrix \( A_k \) will have very small entries below the diagonal, and one can read off a complete system of (approximate) eigenvalues along its diagonal. For each eigenvalue, the computation of the corresponding eigenvector can be done by solving the appropriate homogeneous linear system, or by applying the shifted inverse power method of Exercise \[ \text{Exercise} \].

**Example 10.51.** Consider the matrix \( A = \begin{pmatrix} 2 & 1 \\ 2 & 3 \end{pmatrix} \). The initial Gram–Schmidt factorization \( A = Q_0 R_0 \) yields

\[
Q_0 = \begin{pmatrix} .7071 & .7071 \\ -.7071 & .7071 \end{pmatrix}, \quad R_0 = \begin{pmatrix} 2.8284 & 2.8284 \\ 0 & 1.4142 \end{pmatrix}.
\]

These are multiplied in the reverse order to give \( A_1 = R_0 Q_0 = \begin{pmatrix} 4 & 0 \\ 0 & 1 \end{pmatrix} \). We refactor \( A_1 = Q_1 R_1 \) via Gram–Schmidt, and then reverse multiply to produce

\[
Q_1 = \begin{pmatrix} .9701 & -.2425 \\ .2425 & .9701 \end{pmatrix}, \quad R_1 = \begin{pmatrix} 4.1231 & .2425 \\ 0 & .9701 \end{pmatrix}, \quad A_2 = R_1 Q_1 = \begin{pmatrix} 4.0588 & -.7647 \\ .2353 & .9412 \end{pmatrix}.
\]

The next iteration yields

\[
Q_2 = \begin{pmatrix} .9983 & -.0579 \\ .0579 & .9983 \end{pmatrix}, \quad R_2 = \begin{pmatrix} 4.0656 & -.7090 \\ 0 & .9839 \end{pmatrix}, \quad A_3 = R_2 Q_2 = \begin{pmatrix} 4.0178 & -.9431 \\ .0569 & .9822 \end{pmatrix}.
\]

Continuing in this manner, after 9 iterations we find, to four decimal places

\[
Q_9 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad R_9 = \begin{pmatrix} 4 & -1 \\ 0 & 1 \end{pmatrix}, \quad A_{10} = R_9 Q_9 = \begin{pmatrix} 4 & -1 \\ 0 & 1 \end{pmatrix}.
\]

The eigenvalues of \( A \), namely 4 and 1, appear along the diagonal of \( A_{10} \). Additional iterations produce very little further change, although they can be used for increasing the accuracy of the computed eigenvalues.

If the original matrix \( A \) happens to be symmetric and positive definite, then the limiting matrix \( A_k \to V = \Lambda \) is, in fact, the diagonal matrix containing the eigenvalues of \( A \). Moreover, if, in this case, we recursively define

\[
S_k = S_{k-1} Q_k = Q_0 Q_1 \cdots Q_{k-1} Q_k,
\]

then \( S_k \to S \) have, as their limit, the orthogonal matrix appearing in the Spectral Theorem 8.25, whose columns are the orthonormal eigenvector basis of \( A \).
Example 10.52. Consider the symmetric matrix $A = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 3 & -1 \\ 0 & -1 & 6 \end{pmatrix}$. The initial $A = Q_0 R_0$ factorization produces

$$S_0 = Q_0 = \begin{pmatrix} .8944 & -.4082 & -.1826 \\ .4472 & .8165 & .3651 \\ 0 & -.4082 & .9129 \end{pmatrix}, \quad R_0 = \begin{pmatrix} 2.2361 & 2.2361 & -.4472 \\ 0 & 2.4495 & -3.2660 \\ 0 & 0 & 5.1121 \end{pmatrix},$$

and so

$$A_1 = R_0 Q_0 = \begin{pmatrix} 3 & 1.0954 & 0 \\ 1.0954 & 3.3333 & -2.0870 \\ 0 & -2.0870 & 4.6667 \end{pmatrix}.$$

We refactor $A_1 = Q_1 R_1$ and reverse multiply to produce

$$Q_1 = \begin{pmatrix} .9393 & -.2734 & -.2071 \\ .3430 & .7488 & .5672 \\ 0 & -.6038 & .7972 \end{pmatrix}, \quad R_1 = \begin{pmatrix} 3.1937 & 2.1723 & -.7158 \\ 0 & 3.4565 & -4.3804 \\ 0 & 0 & 2.5364 \end{pmatrix},$$

$$A_2 = \begin{pmatrix} 3.7451 & 1.1856 & 0 \\ 1.1856 & 5.2330 & -1.5314 \\ 0 & -1.5314 & 2.0219 \end{pmatrix}, \quad S_1 = \begin{pmatrix} .7001 & -.4400 & -5.623 \\ .7001 & .2686 & .6615 \\ -.1400 & -.8569 & .4962 \end{pmatrix},$$

where $S_1 = S_0 Q_1 = Q_0 Q_1$. Continuing in this manner, after 10 iterations we find

$$Q_{10} = \begin{pmatrix} 1.0000 & -.0067 & 0 \\ .0067 & 1.0000 & 0.0001 \\ 0 & -.0001 & 1.0000 \end{pmatrix}, \quad R_{10} = \begin{pmatrix} 6.3229 & .0647 & 0 \\ 0 & 3.3582 & -0.006 \\ 0 & 0 & 1.3187 \end{pmatrix},$$

$$A_{11} = \begin{pmatrix} 6.3232 & 0.0224 & 0 \\ .0224 & 3.3581 & -.0002 \\ 0 & -.0002 & 1.3187 \end{pmatrix}, \quad S_{10} = \begin{pmatrix} .0753 & -.5667 & -.8205 \\ .3128 & -.7679 & .5591 \\ -.9468 & -.2987 & .1194 \end{pmatrix}.$$

After 20 iterations, the process has completely settled down, and

$$Q_{20} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad R_{20} = \begin{pmatrix} 6.3234 & .0001 & 0 \\ 0 & 3.3579 & 0 \\ 0 & 0 & 1.3187 \end{pmatrix},$$

$$A_{21} = \begin{pmatrix} 6.3234 & 0 & 0 \\ 0 & 3.3579 & 0 \\ 0 & 0 & 1.3187 \end{pmatrix}, \quad S_{20} = \begin{pmatrix} .0710 & -.5672 & -.8205 \\ .3069 & -.7702 & .5590 \\ -.9491 & -.2915 & .1194 \end{pmatrix}.$$

The eigenvalues of $A$ appear along the diagonal of $A_{21}$, while the columns of $S_{20}$ are the corresponding orthonormal eigenvector basis, both correct to 4 decimal places.

We will devote the remainder of this section to a justification of the $QR$ algorithm. The secret is that the method is, in fact, intimately connected with the more primitive power method. To keep the exposition under control, let us make the simplifying assumption
that the matrix $A > 0$ is symmetric and positive definite with distinct positive eigenvalues that we label in decreasing order:

$$\lambda_1 > \lambda_2 > \cdots > \lambda_n. \quad (10.99)$$

The corresponding eigenvectors $u_1, \ldots, u_n$ can be chosen to form an orthonormal basis of $\mathbb{R}^n$. While not the most general case to which the QR algorithm applies, the positive definite matrices are among the most important subclass, and the ones for which the basic algorithm applies as stated.

If one were to implement a version of the power method to capture all the eigenvectors and eigenvalues of $A$, one might think of iterating a complete basis $u_1^{(0)}, \ldots, u_n^{(0)}$ of $\mathbb{R}^n$ instead of just one individual vector. The problem is that, for almost all vectors, the matrix power iterates $u_j^{(k)} = A^k u_j$ all tend to a multiple of the eigenvector $u_1$ corresponding to the dominant eigenvalue. Normalizing the vectors at each step of the algorithm, as in (10.96), is not any better, since then they merely converge to one of the two dominant unit eigenvectors $\pm u_1$. However, if, inspired by the form of the basis, we orthonormalize the vectors at each step, then we effectively keep them separate and so prevent them from all accumulating at the dominant unit eigenvector, and so, with some luck, they may decide to converge to the other eigenvectors. This inspired hope is the heart of the QR algorithm!

Thus, we start with any orthonormal basis, which, for simplicity, we take to be the standard basis vectors of $\mathbb{R}^n$, and so $u_j^{(0)} = e_j$, $j = 1, \ldots, n$. At the $k$th stage of the algorithm, we set $u_1^{(k)}, \ldots, u_n^{(k)}$ to be the orthonormal vectors that result from applying the Gram–Schmidt algorithm to the power vectors $v_j^{(k)} = A^k e_j$. In matrix language, the vectors $v_1^{(k)}, \ldots, v_n^{(k)}$ are merely the columns of $A^k$, and the orthonormal basis $u_1^{(k)}, \ldots, u_n^{(k)}$ are the columns of the orthogonal matrix $S_k$ in the QR decomposition of the $k$th power of $A$, which we denote by

$$A^k = S_k P_k, \quad (10.100)$$

where $P_k$ is positive upper triangular. Note that, in view of (10.97)

$$A = Q_0 R_0, \quad A^2 = Q_0 R_0 Q_0 R_0 = Q_0 Q_1 R_1 R_0, \quad A^3 = Q_0 R_0 Q_0 R_0 Q_0 R_0 = Q_0 Q_1 Q_1 R_1 R_1 R_0 = Q_0 Q_1 Q_2 R_2 R_1 R_0,$$

and, in general,

$$A^k = (Q_0 Q_1 \cdots Q_{k-1} Q_k) \left( R_k R_{k-1} \cdots R_1 R_0 \right). \quad (10.101)$$

The product of orthogonal matrices is also orthogonal, and the product of positive upper triangular matrices is also positive upper triangular. Comparing (10.100), (10.101), we conclude that

$$S_k = Q_0 Q_1 \cdots Q_{k-1} Q_k = S_{k-1} Q_k, \quad P_k = R_k R_{k-1} \cdots R_1 R_0 = R_k P_{k-1}, \quad (10.102)$$

since the QR factorization is unique once one requires that all the diagonal entries of $R$ be positive.
Let \( S = (u_1 \ u_2 \ \ldots \ u_n) \) denote the orthogonal eigenvector matrix. The Spectral Theorem 8.25 tells us that
\[
A = S \Lambda S^T, \quad \text{where} \quad \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)
\]
is the diagonal eigenvalue matrix. Substituting the spectral factorization into (10.100), we find
\[
A^k = S \Lambda^k S^T = S_k P_k.
\]
We now make one additional assumption on the eigenvectors by requiring that \( S^T \) be a regular matrix, and so, by Gaussian elimination, admits a factorization \( S^T = LU \) into a product of special lower and upper triangular matrices. This holds generically, and is the analog of the condition that our original vector in the basic power method includes a nonzero component of the dominant eigenvector. (The reader can trace through the argument in the general case that \( S^T \) requires a permuted \( LU \) factorization.) Under this assumption,
\[
A^k = S \Lambda^k L U = S_k P_k, \quad \text{and hence} \quad S \Lambda^k L = S_k P_k U^{-1}.
\]
Multiplying on the right by \( \Lambda^{-k} \) we obtain
\[
S \Lambda^k L \Lambda^{-k} = S_k T_k, \quad \text{where} \quad T_k = P_k U^{-1} \Lambda^{-k}
\]
is also a positive upper triangular matrix.

Now consider what happens as \( k \to \infty \). Since \( L \) is lower triangular, the entries of the matrix \( \Lambda^k L \Lambda^{-k} \) below the diagonal are \( l_{ij} (\lambda_j/\lambda_i)^k \to 0 \), since \( i > j \), and, by (10.99), \( 0 < \lambda_j/\lambda_i < 1 \). Its diagonal entries are all equal to 1, and therefore, in the limit,
\[
\Lambda^k L \Lambda^{-k} \to I,
\]
with convergence rate governed by the ratio \( \lambda_2/\lambda_1 \) between the subdominant and the dominant eigenvalues. As a consequence, in the limit as \( k \to \infty \), the left hand side of (10.103) tends to the orthogonal eigenvector matrix \( S \):
\[
S_k T_k \to S. \quad \text{(10.104)}
\]
We now make use of the following result, whose proof will be given after we finish the justification of the \( QR \) algorithm.

**Lemma 10.53.** The products of orthogonal and positive upper triangular matrices have an orthogonal limit, as in (10.104), if and only if the individual matrices have limits
\[
S_k \to S, \quad T_k \to I. \quad \text{(10.105)}
\]

Therefore, as claimed, the orthogonal matrices \( S_k \) do converge to the orthogonal eigenvector matrix. Moreover, by (10.102), (10.103),
\[
R_k = P_k P_{k-1}^{-1} = (T_k \Lambda^k U^{-1}) (T_{k-1} \Lambda^{k-1} U^{-1})^{-1} = T_k \Lambda T_{k-1}^{-1}.
\]
Since both \( T_k \) and \( T_{k-1} \) converge to the identity matrix, in the limit \( R_k \to \Lambda \) converges to the diagonal eigenvalue matrix, as claimed. The eigenvalues appear in decreasing order along the diagonal — this follows from our regularity assumption on the transposed eigenvector matrix \( S^T \).
Theorem 10.54. If $A$ is symmetric, satisfies (10.99), and $S^T$ is a regular matrix, then the matrices $S_k \rightarrow S$ and $R_k \rightarrow \Lambda$ in the QR algorithm converge to, respectively the eigenvector matrix and the diagonal eigenvalue matrix. The rate of convergence is governed by the ratio between the subdominant and dominant eigenvalues.

The last remaining detail is a proof of Lemma 10.53. We write $S = (u_1, u_2, \ldots, u_n)$, $S_k = (u_1^{(k)}, \ldots, u_n^{(k)})$ in columnar form, and let $t_{ij}^{(k)}$ denote the matrix entries of the positive upper triangular matrix $T_k$. The last column of the limiting equation (10.104) reads $t_{nn}^{(k)} u_n^{(k)} \rightarrow u_n$. Since both $u_n^{(k)}$ and $u_n$ are unit vectors, and $t_{nn}^{(k)} > 0$, the norm of the limit implies $t_{nn}^{(k)} \rightarrow 1$ and hence the last column $u_n^{(k)} \rightarrow u_n$. The next to last column of (10.104) reads

$$t_{n-1,n}^{(k)} u_{n-1}^{(k)} + t_{n-1,n}^{(k)} u_{n-1}^{(k)} \rightarrow u_{n-1}.$$ 

Taking the inner product with $u_n^{(k)} \rightarrow u_n$ and using orthonormality, we deduce $t_{n-1,n}^{(k)} \rightarrow 0$, and so $t_{n-1,n}^{(k)} u_{n-1}^{(k)} \rightarrow u_{n-1}$, which, by the previous reasoning, implies $t_{n-1,n}^{(k)} \rightarrow 1$ and $u_{n-1}^{(k)} \rightarrow u_{n-1}$. The proof is completed through a reverse induction on the columns, and the remaining details are left to the reader.

Tridiagonalization

In practical implementations, the direct QR algorithm is not very efficient, and takes too long to provide reasonable approximations to the eigenvalues of large matrices. Fortunately, the algorithm can be made efficient by a simple preprocessing step. The key observation is that the QR algorithm preserves the class of symmetric tridiagonal matrices, cf. Exercise $\blacksquare$, and, moreover, like Gaussian elimination, is much faster when applied to this class of matrices. Alston Householder devised a simple method that converts any symmetric matrix into tridiagonal form while preserving all the eigenvalues. Thus, by first applying the Householder tridiagonalization algorithm, and then applying the QR method to the resulting tridiagonal matrix, the result is an efficient and practical algorithm for computing eigenvalues of symmetric matrices.

Consider the Householder or elementary reflection matrix

$$H = I - 2u u^T$$

(10.106)

in which $u$ is a unit vector (in the Euclidean norm). The matrix $H$ represents a reflection of vectors through the plane perpendicular to $u$, as illustrated in Figure 10.5. According to Exercise $\blacksquare$, $H$ is a symmetric orthogonal matrix, and so

$$H^T = H, \quad H^2 = I, \quad H^{-1} = H.$$ 

The proof is straightforward: symmetry is immediate, while

$$H H^T = H^2 = (I - 2u u^T) (I - 2u u^T) = I - 4u u^T + 4u (u^T u) u^T = I$$

since, by assumption, $u^T u = \|u\|^2 = 1$. By suitably prescribing the unit vector $u$, we can construct an elementary reflection matrix that interchanges any two vectors of the same length.
Lemma 10.55. Let $x, y \in \mathbb{R}^n$ with $\|x\| = \|y\|$. Set $u = \frac{x - y}{\|x - y\|}$ and let $H = I - 2uu^T$ be the corresponding elementary reflection matrix. Then $Hx = y$ and $Hy = x$.

Proof: Keeping in mind that $x$ and $y$ have the same Euclidean norm, we compute

$$Hx = (I - 2uu^T)x = x - 2 \frac{(x - y)(x - y)^T x}{\|x - y\|^2}$$

$$= x - 2 \frac{(x - y)(\|x\|^2 - y \cdot x)}{2 \|x\|^2 - 2x \cdot y} = x - (x - y) = y.$$

The proof of the second equation is similar. Q.E.D.

Given a symmetric $n \times n$ matrix $A$, our goal is to devise a similar tridiagonal matrix by applying Householder reflections. We begin by setting

$$x_1 = \begin{pmatrix} 0 \\ a_{21} \\ a_{31} \\ \vdots \\ a_{n1} \end{pmatrix}, \quad y_1 = \begin{pmatrix} 0 \\ \pm r_1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad \text{where} \quad r_1 = \|x_1\| = \|y_1\|;$$

so that $x_1$ consists of all off-diagonal entries of the first column of $A$. Let

$$H_1 = I - 2u_1u_1^T,$$

where $u_1 = \frac{x_1 - y_1}{\|x_1 - y_1\|}$

be the corresponding elementary reflection matrix that maps $x_1$ to $y_1$. Either $\pm$ sign in the formula for $y_1$ works in the algorithm; the practical choice is to set it to be the opposite of the sign of the entry $a_{21}$, which minimizes the possible effects of round-off error when
computing the unit vector \( \mathbf{u}_1 \). A direct computation, based on Lemma 10.55 and the fact that the first entry of \( \mathbf{u}_1 \) is zero, proves that

\[
A_2 = H_1 A H_1 = \begin{pmatrix}
a_{11} & r_1 & 0 & \cdots & 0 \\
r_1 & \bar{a}_{22} & \bar{a}_{23} & \cdots & \bar{a}_{2n} \\
0 & \bar{a}_{32} & \bar{a}_{33} & \cdots & \bar{a}_{3n} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \bar{a}_{n2} & \bar{a}_n & \cdots & \bar{a}_{nn}
\end{pmatrix}
\]

(10.107)

for some \( \tilde{a}_{ij} \). Thus, by a single Householder transformation, we can arrange that the first row and column of \( A \) are of tridiagonal form. At the next stage, we work on the second row and column of the new matrix \( A_2 \). We set

\[
\mathbf{x}_2 = \begin{pmatrix}
0 \\
0 \\
\bar{a}_{32} \\
\bar{a}_{42} \\
\vdots \\
\bar{a}_{n2}
\end{pmatrix}, \quad \mathbf{y}_1 = \begin{pmatrix}
0 \\
0 \\
\pm r_2 \\
\vdots \\
0
\end{pmatrix}, \quad \text{where} \quad r_2 = \| \mathbf{x}_2 \| = \| \mathbf{y}_2 \|,
\]

and the sign is chosen to be the same as that of \( \bar{a}_{32} \). We set \( H_2 = I - 2 \mathbf{u}_2 \mathbf{u}_2^T \), leading to

\[
A_3 = H_2 A_2 H_2 = \begin{pmatrix}
a_{11} & r_1 & 0 & 0 & \cdots & 0 \\
r_1 & \bar{a}_{22} & r_2 & 0 & \cdots & 0 \\
0 & r_2 & \bar{a}_{33} & \bar{a}_{34} & \cdots & \bar{a}_{3n} \\
0 & 0 & \bar{a}_{43} & \bar{a}_{44} & \cdots & \bar{a}_{4n} \\
\vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\
0 & 0 & \bar{a}_{n3} & \bar{a}_{n4} & \cdots & \bar{a}_{nn}
\end{pmatrix},
\]

whose first two rows and columns are in tridiagonal form. The remaining steps in the algorithm should now be clear; once they are reached, the final two columns need not be modified since the resulting matrix will be in tridiagonal form. Let us illustrate the method by an example.

**Example 10.56.** To tridiagonalize matrix \( A = \begin{pmatrix}
4 & 1 & -1 & 2 \\
1 & 4 & 1 & -1 \\
-1 & 1 & 4 & 1 \\
2 & -1 & 1 & 4
\end{pmatrix} \), we begin with its first column. We set \( \mathbf{x}_1 = \begin{pmatrix}
0 \\
1 \\
-1 \\
2
\end{pmatrix} \), and so \( \mathbf{y}_1 = \begin{pmatrix}
0 \\
\sqrt{6} \\
0 \\
0
\end{pmatrix} \approx \begin{pmatrix}
0 \\
2.4495 \\
0 \\
0
\end{pmatrix} \). Therefore,
the unit vector is $u_1 = \frac{x_1 - y_1}{\|x_1 - y_1\|} = \begin{pmatrix} 0 \\ .8391 \\ -.2433 \\ .4865 \end{pmatrix}$, with corresponding Householder matrix

$$H_1 = I - 2 u_1 u_1^T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1.082 & .4082 & -.8165 \\ 0 & .4082 & .8816 & .2367 \\ 0 & -.8165 & .2367 & .5266 \end{pmatrix}$$

and so

$$A_2 = H_1 A H_1 = \begin{pmatrix} 4.0000 & -2.4495 & 0 & 0 \\ -2.4495 & 2.3333 & -.3865 & -.8599 \\ 0 & -.3865 & 4.9440 & -.1246 \\ 0 & -.8599 & -.1246 & 4.7227 \end{pmatrix}$$

In the next phase, $x_2 = \begin{pmatrix} 0 \\ 0 \\ -.3865 \\ -.8599 \end{pmatrix}$, $y_2 = \begin{pmatrix} 0 \\ 0 \\ -.9428 \\ 0 \end{pmatrix}$, so $u_2 = \begin{pmatrix} 0 \\ 0 \\ -.8396 \\ -.5431 \end{pmatrix}$, and

$$H_2 = I - 2 u_2 u_2^T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -.4100 & -.9121 \\ 0 & 0 & -.9121 & .4100 \end{pmatrix}$$.

The resulting matrix

$$T = A_3 = H_2 A_2 H_2 = \begin{pmatrix} 4.0000 & -2.4495 & 0 & 0 \\ -2.4495 & 2.3333 & .9428 & 0 \\ 0 & .9428 & 4.6667 & 0 \\ 0 & 0 & 0 & 5.0000 \end{pmatrix}$$

is now in tridiagonal form.

Now, the key point is that, because the Householder matrices are their own inverses, the resulting matrices

$$A = A_1, \quad A_2 = H_1 A H_1^{-1}, \quad A_3 = H_2 A_2 H_2^{-1} = (H_1 H_2) A (H_1 H_2)^{-1}, \quad \ldots$$

are all similar and hence have the same eigenvalues. Thus the final result is a tridiagonal matrix $T = A_n$ that has the *same eigenvalues* as the original symmetric matrix $A$.

We may then apply the $QR$ algorithm to the final tridiagonal matrix $T$ to approximate its eigenvalues, and hence the eigenvalues of $A$. According to Exercise [1], in the resulting Gram–Schmidt procedure, the iterates $A_k$ are all tridiagonal. Moreover, the number of arithmetic operations is relatively small; in Exercise [2] you are asked to quantify this. For instance, in the preceding example, after we apply 20 iterations of the $QR$ algorithm
directly to $T$, the upper triangular factor has become

$$ R_{20} = \begin{pmatrix} 6.0000 & -0.0065 & 0 & 0 \\ 0 & 4.5616 & 0 & 0 \\ 0 & 0 & 5.0000 & 0 \\ 0 & 0 & 0 & .4384 \end{pmatrix}, $$

The eigenvalues of $T$ and hence also $A$ are along the diagonal, and are correct to 4 decimal places.

Finally, if $A$ is not symmetric, then the Householder transformations proceed as before, but the result is no longer tridiagonal, but rather an upper Hessenberg matrix, which means that all its entries below the subdiagonal are zero. Thus, a $5 \times 5$ upper Hessenberg matrix looks like

$$
\begin{pmatrix}
* & * & * & * & * \\
* & * & * & * & * \\
0 & * & * & * & * \\
0 & 0 & * & * & * \\
0 & 0 & 0 & * & *
\end{pmatrix}
$$

where the starred entries can be anything. The $QR$ algorithm maintains the upper Hessenberg form, and, while not nearly as efficient as the tridiagonal case, still produces a significant savings in computational effort required to find the eigenvalues.