

Chapter 12

More Numerical Linear Algebra: Eigenvalues, Singular Values, and Iterative Methods for Solving Linear Systems

12.1 Eigenvalue Problems

Let A be an n by n matrix. We wish to find a real or complex number λ , called an *eigenvalue*, and a nonzero vector \mathbf{v} , called an *eigenvector*, such that $A\mathbf{v} = \lambda\mathbf{v}$. Eigenvalue problems arise in many areas, such as the study of *resonance* of musical instruments, or the stability of fluid flows. We saw in the previous chapter how eigenvalue analysis can be used to study the behavior of a system of linear ODE's, $\mathbf{y}' = A\mathbf{y}$. If λ is an eigenvalue of A with corresponding eigenvector \mathbf{v} , then if $\mathbf{y}(0) = \mathbf{v}$, then $\mathbf{y}(t) = e^{\lambda t}\mathbf{v}$. More generally, if $\lambda_1, \dots, \lambda_n$ are the eigenvalues of A with corresponding eigenvectors $\mathbf{v}_1, \dots, \mathbf{v}_n$, then if $\mathbf{y}(0) = \sum_{j=1}^n c_j \mathbf{v}_j$, then $\mathbf{y}(t) = \sum_{j=1}^n c_j e^{\lambda_j t} \mathbf{v}_j$.

The eigenvalues of A are the roots of its *characteristic polynomial*:

$$\det(A - \lambda I) = 0.$$

To find the eigenvalues of

$$A = \begin{pmatrix} 1 & 2 \\ 4 & 3 \end{pmatrix}$$

we can write

$$\det(A - \lambda I) = (1 - \lambda)(3 - \lambda) - 8 = \lambda^2 - 4\lambda - 5 = 0,$$

and solve the quadratic equation to find

$$\lambda = -1 \text{ or } \lambda = 5.$$

Even real matrices can have *complex* eigenvalues, as the following example shows:

$$A = \begin{pmatrix} 1 & 2 \\ -4 & 3 \end{pmatrix}, \quad \det(A - \lambda I) = (1 - \lambda)(3 - \lambda) + 8 = \lambda^2 - 4\lambda + 11,$$

$$\lambda = \frac{4 \pm \sqrt{-28}}{2} = 2 \pm i\sqrt{7}.$$

In most cases, there is no analytic formula for the eigenvalues of a matrix, since the eigenvalues are the roots of the characteristic polynomial, and Abel proved in 1824 that there can be no formula (involving rational numbers, addition, subtraction, multiplication, division, and k th roots) for the roots of a general polynomial of degree 5 or higher. Hence the best one can hope to do is to approximate the eigenvalues numerically. One possible approach would be to determine the coefficients of the characteristic polynomial and then use a root finding routine, as described in Chapter 2, to approximate its roots. Unfortunately, this strategy is a bad one because the polynomial root finding problem may be ill-conditioned, even when the underlying eigenvalue problem is not. A tiny change in the coefficients of the polynomial can make a large change in its roots. We will find other methods for approximating eigenvalues.

Associated with each eigenvalue λ is a collection of nonzero vectors \mathbf{v} , called eigenvectors, satisfying $A\mathbf{v} = \lambda\mathbf{v}$. The eigenvectors \mathbf{v} associated with an eigenvalue λ , together with the zero vector, form a vector space, sometimes called the *eigenspace* of λ . In many cases, this space is one-dimensional and \mathbf{v} is determined up to nonzero scalar multiples; clearly if \mathbf{v} is an eigenvector with eigenvalue λ , then so is $c\mathbf{v}$ for any nonzero scalar c , since $A\mathbf{v} = \lambda\mathbf{v} \Rightarrow A(c\mathbf{v}) = \lambda(c\mathbf{v})$. Sometimes there are two or more linearly independent eigenvectors, say $\mathbf{v}_1, \dots, \mathbf{v}_m$, associated with an eigenvalue λ , and in this case, for any scalars c_1, \dots, c_m not all zero, the vector $\sum_{j=1}^m c_j \mathbf{v}_j$ is also an eigenvector for λ since $A(\sum_{j=1}^m c_j \mathbf{v}_j) = \sum_{j=1}^m c_j A\mathbf{v}_j = \sum_{j=1}^m c_j \lambda \mathbf{v}_j = \lambda(\sum_{j=1}^m c_j \mathbf{v}_j)$. The dimension of the eigenspace is called the *geometric multiplicity* of λ . The eigenspace of λ is the same as the null space of $A - \lambda I$, and to find the eigenvectors associated with λ we can look for nonzero vectors \mathbf{v} that satisfy $(A - \lambda I)\mathbf{v} = \mathbf{0}$.

In the first example above, for instance, to find the eigenvector(s) associated with the eigenvalue $\lambda = -1$, we look for nonzero vectors $\mathbf{v} = (v_1, v_2)^T$ satisfying

$$\begin{pmatrix} 1 & 2 \\ 4 & 3 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = - \begin{pmatrix} v_1 \\ v_2 \end{pmatrix},$$

or, equivalently,

$$\begin{pmatrix} 2 & 2 \\ 4 & 4 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

If we begin to solve this set of equations by Gaussian elimination, subtracting 2 times the first row from the second, then we find:

$$\left(\begin{array}{cc|c} 2 & 2 & 0 \\ 4 & 4 & 0 \end{array} \right) \longrightarrow \left(\begin{array}{cc|c} 2 & 2 & 0 \\ 0 & 0 & 0 \end{array} \right),$$

and the solution set consists of all vectors \mathbf{v} such that $2v_1 + 2v_2 = 0$; i.e., such that $v_2 = -v_1$. We can take v_1 to have any nonzero value, say, $v_1 = 1$, and then the eigenvector is $(1, -1)^T$.

Following is a 3 by 3 example:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}.$$

To find the eigenvalues of A , we form the characteristic polynomial, $\det(A - \lambda I)$:

$$\det(A - \lambda I) = \det \begin{pmatrix} 1 - \lambda & 2 & 3 \\ 4 & 5 - \lambda & 6 \\ 7 & 8 & 9 - \lambda \end{pmatrix} =$$

$$(1 - \lambda) \det \begin{pmatrix} 5 - \lambda & 6 \\ 8 & 9 - \lambda \end{pmatrix} - 2 \det \begin{pmatrix} 4 & 6 \\ 7 & 9 - \lambda \end{pmatrix} + 3 \det \begin{pmatrix} 4 & 5 - \lambda \\ 7 & 8 \end{pmatrix} =$$

$$(1 - \lambda)[(5 - \lambda)(9 - \lambda) - 6 \cdot 8] - 2[4(9 - \lambda) - 6 \cdot 7] + 3[4 \cdot 8 - (5 - \lambda) \cdot 7] = -\lambda(\lambda^2 - 15\lambda + 18).$$

The roots of this polynomial are:

$$\lambda = 0 \quad \text{and} \quad \lambda = \frac{15 \pm 3\sqrt{17}}{2}.$$

Let us find the eigenvector(s) associated with the eigenvalue 0. To do this we must find all nonzero vectors $\mathbf{v} = (v_1, v_2, v_3)^T$ satisfying

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

Proceeding by Gaussian elimination we find

$$\left(\begin{array}{ccc|c} 1 & 2 & 3 & 0 \\ 4 & 5 & 6 & 0 \\ 7 & 8 & 9 & 0 \end{array} \right) \rightarrow \left(\begin{array}{ccc|c} 1 & 2 & 3 & 0 \\ 0 & -3 & -6 & 0 \\ 0 & -6 & -12 & 0 \end{array} \right) \rightarrow \left(\begin{array}{ccc|c} 1 & 2 & 3 & 0 \\ 0 & -3 & -6 & 0 \\ 0 & 0 & 0 & 0 \end{array} \right).$$

The last equation ($0 \cdot v_1 + 0 \cdot v_2 + 0 \cdot v_3 = 0$) holds for any \mathbf{v} , the second equation ($-3v_2 - 6v_3 = 0$) holds provided $v_2 = -2v_3$, and the first equation ($v_1 + 2v_2 + 3v_3 = 0$) holds as well provided $v_1 = -2v_2 - 3v_3 = v_3$. Setting v_3 arbitrarily to 1 gives the eigenvector $\mathbf{v} = (1, -2, 1)^T$. You should check that $A\mathbf{v}$ is indeed equal to $0 \cdot \mathbf{v}$.

An n by n matrix A that has n linearly independent eigenvectors is said to be *diagonalizable*. If $\mathbf{v}_1, \dots, \mathbf{v}_n$ are a set of linearly independent eigenvectors with corresponding eigenvalues $\lambda_1, \dots, \lambda_n$, then we can write

$$A(\mathbf{v}_1, \dots, \mathbf{v}_n) = (\lambda_1 \mathbf{v}_1, \dots, \lambda_n \mathbf{v}_n) = (\mathbf{v}_1, \dots, \mathbf{v}_n) \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}.$$

If $V \equiv (\mathbf{v}_1, \dots, \mathbf{v}_n)$ is the matrix whose columns are the eigenvectors $\mathbf{v}_1, \dots, \mathbf{v}_n$, and Λ is the diagonal matrix of eigenvalues, then this becomes

$$AV = V\Lambda, \quad \text{or } A = V\Lambda V^{-1}.$$

A *similarity transformation* $A \rightarrow W^{-1}AW$, where W is any nonsingular matrix, preserves eigenvalues since if $A\mathbf{v} = \lambda\mathbf{v}$, then $W^{-1}AW(W^{-1}\mathbf{v}) = W^{-1}A\mathbf{v} = \lambda(W^{-1}\mathbf{v})$, but the eigenvectors of $W^{-1}AW$ are W^{-1} times the eigenvectors of A . A similarity transformation corresponds to a change of variables. For example, consider the linear system $A\mathbf{x} = \mathbf{b}$, where $A = V\Lambda V^{-1}$. Making the change of variables $\mathbf{y} = V^{-1}\mathbf{x}$ and $\mathbf{c} = V^{-1}\mathbf{b}$, this becomes the diagonal system $V^{-1}AV\mathbf{y} \equiv \Lambda\mathbf{y} = \mathbf{c}$. Following are two important theorems about classes of matrices that are diagonalizable:

Theorem 12.1.1. *If A is an n by n matrix with n distinct eigenvalues then A is diagonalizable.*

Proof. Let $\lambda_1, \dots, \lambda_n$ be the distinct eigenvalues of A , and let $\mathbf{v}_1, \dots, \mathbf{v}_n$ be corresponding eigenvectors. If $\mathbf{v}_1, \dots, \mathbf{v}_n$ were not linearly independent, then there would be a linear combination of them with nonzero coefficients that was equal to the zero vector. Let $\sum_{j=1}^m c_j \mathbf{v}_j$ be the shortest such linear combination (i.e., the one involving the fewest \mathbf{v}_j 's). Then we have the two equations $\sum_{j=1}^m c_j \mathbf{v}_j = \mathbf{0}$ and $A \sum_{j=1}^m c_j \mathbf{v}_j = \sum_{j=1}^m c_j \lambda_j \mathbf{v}_j = \mathbf{0}$. Subtracting λ_1 times the first equation from the second gives $\sum_{j=2}^m c_j (\lambda_j - \lambda_1) \mathbf{v}_j = \mathbf{0}$, but this is a contradiction since this is a shorter linear combination with nonzero coefficients (since $c_j \neq 0$ and $\lambda_j - \lambda_1 \neq 0$, $j = 2, \dots, m$) that is equal to $\mathbf{0}$. Therefore there must be no such linear combination and $\mathbf{v}_1, \dots, \mathbf{v}_n$ must be linearly independent. \square

Theorem 12.1.2. *If A is real and symmetric ($A = A^T$) then the eigenvalues of A are real and A is diagonalizable via an orthogonal similarity transformation; that is $A = Q\Lambda Q^T$, where $Q^T = Q^{-1}$ and Λ is the diagonal matrix of eigenvalues.*

Not all n by n matrices have n linearly independent eigenvectors. Consider, for example, the following 2 by 2 matrix:

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

The characteristic polynomial is $\det(A - \lambda I) = (1 - \lambda)^2$, which has a double root at $\lambda = 1$. The eigenspace associated with $\lambda = 1$ is the set of solutions to

$$(A - I)v = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

which consists of vectors \mathbf{v} for which $v_2 = 0$; i.e., the one-dimensional space of scalar multiples of $(1, 0)^T$.

While not all square matrices are diagonalizable, it can be shown that every square matrix is similar to a matrix in *Jordan canonical form*:

Theorem 12.1.3 (Jordan Canonical Form). *Every n by n matrix A is similar to one of the form*

$$J = \begin{pmatrix} J_1 & & \\ & \ddots & \\ & & J_m \end{pmatrix},$$

where each block J_i has the form

$$J_i = \begin{pmatrix} \lambda_i & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_i \end{pmatrix}.$$

The number of linearly independent eigenvectors is the number of blocks m . The matrix is diagonalizable if and only if $m = n$. The geometric multiplicity of an eigenvalue λ_i is the number of Jordan blocks with eigenvalue λ_i . The algebraic multiplicity of λ_i (i.e., its degree as a root of the characteristic polynomial) is the sum of the orders of all Jordan blocks with eigenvalue λ_i .

While the Jordan form is an important theoretical tool, it is less useful in computations. A tiny change in a matrix can make a huge change in its Jordan form. For example, suppose we have a single n by n Jordan block with eigenvalue 1. Changing the diagonal entries by arbitrarily small but distinct amounts ϵ_i , we obtain a matrix with distinct eigenvalues $1 + \epsilon_i$, $i = 1, \dots, n$, which, by Theorem 10.1.1, is diagonalizable; i.e., has a Jordan form consisting of n one by one blocks instead of one n by n block.

Often the most useful similarity transformations are those that can be carried out with orthogonal or, more generally, unitary matrices. Recall that an orthogonal matrix is a real matrix Q such that $Q^T = Q^{-1}$. The complex analog is a *unitary matrix*: a complex matrix for which the complex conjugate transpose (the matrix Q^* whose i, j entry is equal to the complex conjugate of Q_{ji}) is equal to Q^{-1} . We state one more important theorem, called *Schur's theorem*, saying that every square matrix is unitarily similar to an upper triangular matrix.

Theorem 12.1.4 (Schur Form). *Every square matrix A can be written in the form $A = QTQ^*$ where Q is a unitary matrix and T is upper triangular.*

For certain types of matrices, the eigenvalues and eigenvectors are easy to find. For example, the eigenvalues of a *diagonal* matrix

$$A = \begin{pmatrix} d_1 & & \\ & \ddots & \\ & & d_n \end{pmatrix}$$

are just the diagonal entries d_1, \dots, d_n , and the corresponding eigenvectors are the unit vectors, $\mathbf{e}_1, \dots, \mathbf{e}_n$, where \mathbf{e}_j has a 1 in position j and 0's everywhere else; for we have $A\mathbf{e}_j = d_j\mathbf{e}_j$. If A

is *triangular*, either upper or lower triangular, then its eigenvalues are again equal to its diagonal entries, and the eigenvectors can be determined by back substitution. For example, if

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 4 & 5 \\ 0 & 0 & 6 \end{pmatrix}$$

then its eigenvalues are 1, 4, and 6. The eigenvector associated with 1 is $\mathbf{e}_1 = (1, 0, 0)^T$. The eigenvector associated with 4 satisfies

$$\begin{pmatrix} -3 & 2 & 3 \\ 0 & 0 & 5 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix},$$

and so must have $v_3 = 0$ and $-3v_1 + 2v_2 = 0$, say, $\mathbf{v} = (2/3, 1, 0)^T$. The eigenvector associated with 6 satisfies

$$\begin{pmatrix} -5 & 2 & 3 \\ 0 & -2 & 5 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix},$$

and so v_3 can be any nonzero number, while $v_2 = (5/2)v_3$ and $v_1 = (2/5)v_2 + (3/5)v_3 = (8/5)v_3$, giving, for example, $\mathbf{v} = (8/5, 5/2, 1)^T$.

A very useful theorem for obtaining bounds on eigenvalues, without actually doing much computation, is *Gerschgorin's theorem*. We will prove the first part of this theorem.

Theorem 12.1.5 (Gerschgorin). *Let A be an n by n matrix with entries a_{ij} and let r_i denote the sum of the absolute values of the off-diagonal entries in row i : $r_i = \sum_{\substack{j=1 \\ j \neq i}}^n |a_{ij}|$. Let D_i denote the disk in the complex plane centered at a_{ii} and of radius r_i :*

$$D_i = \{z \in \mathbf{C} : |z - a_{ii}| \leq r_i\}.$$

Then all eigenvalues of A lie in the union $\cup_{i=1}^n D_i$ of the Gerschgorin disks. If m of these disks are connected and disjoint from the others, then exactly m eigenvalues of A lie in this connected component.

Proof of first part. Let λ be an eigenvalue of A with corresponding eigenvector \mathbf{v} . Then for each $i = 1, \dots, n$, we have $\sum_{j=1}^n a_{ij}v_j = \lambda v_i$, or equivalently, $(\lambda - a_{ii})v_i = \sum_{\substack{j=1 \\ j \neq i}}^n a_{ij}v_j$. Let v_k be the component of \mathbf{v} of largest absolute value. Then using this formula with $i = k$ and dividing each side by v_k we find

$$\lambda - a_{kk} = \sum_{\substack{j=1 \\ j \neq k}}^n a_{kj}(v_j/v_k),$$

and hence, taking absolute values on each side,

$$|\lambda - a_{kk}| \leq \sum_{\substack{j=1 \\ j \neq k}}^n |a_{ij}| \cdot |v_i/v_k| \leq \sum_{\substack{j=1 \\ j \neq k}}^n |a_{ij}|.$$

□

The theorem is stated using the *Gerschgorin row disks*. Since the eigenvalues of A^T are the same as those of A , however, (because if $A = SJS^{-1}$ where J is in Jordan form then $A^T = (S^T)^{-1}J^T S^T$ and the eigenvalues of J^T are the same as those of J), one can state a similar result using the *Gerschgorin column disks*:

Let c_j denote the sum of the absolute values of the off-diagonal entries in column j : $c_j = \sum_{\substack{i=1 \\ i \neq j}}^n |a_{ij}|$.

Let E_j denote the disk in the complex plane centered at a_{jj} and of radius c_j :

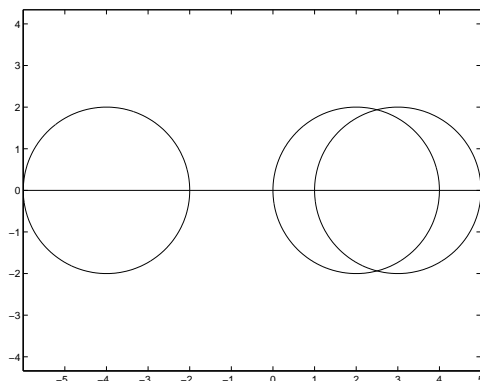
$$E_j = \{z \in \mathbf{C} : |z - a_{jj}| \leq c_j\}.$$

Then all eigenvalues of A lie in the union $\cup_{j=1}^n E_j$ of the Gerschgorin column disks. If m of these disks are connected and disjoint from the others, then exactly m eigenvalues of A lie in this connected component.

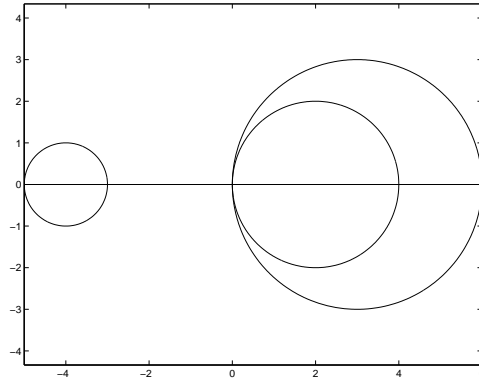
Example. Consider the matrix

$$A = \begin{pmatrix} 3 & 1 & 1 \\ 2 & 2 & 0 \\ 1 & -1 & -4 \end{pmatrix}.$$

The Gerschgorin row disks, $D_1 = \{z \in \mathbf{C} : |z - 3| \leq 2\}$, $D_2 = \{z \in \mathbf{C} : |z - 2| \leq 2\}$, and $D_3 = \{z \in \mathbf{C} : |z + 4| \leq 2\}$, are pictured below. According to the theorem, one eigenvalue lies in the disk centered at -4 , while two eigenvalues lie in the union of the disks centered at 2 and 3.



The Gerschgorin column disks, $E_1 = \{z \in \mathbf{C} : |z - 3| \leq 3\}$, $E_2 = \{z \in \mathbf{C} : |z - 2| \leq 2\}$, and $E_3 = \{z \in \mathbf{C} : |z + 4| \leq 1\}$, are pictured below.



Combining information from each of the two figures, we obtain the stronger result that one eigenvalue lies in E_3 while two lie in $D_1 \cup D_2$.

12.1.1 The Power Method for Computing the Largest Eigenpair

Suppose one starts with a vector \mathbf{w} and applies the matrix A to it many, many times, obtaining $A^k \mathbf{w}$, $k = 1, 2, \dots$. Let us assume that A is *diagonalizable*, meaning that it has n linearly independent eigenvectors $\mathbf{v}_1, \dots, \mathbf{v}_n$ forming a basis for \mathbf{C}^n . The vector \mathbf{w} can be expressed as a linear combination of these eigenvectors: $\mathbf{w} = \sum_{j=1}^n c_j \mathbf{v}_j$, for some scalars c_1, \dots, c_n . Then $A\mathbf{w} = \sum_{j=1}^n c_j A\mathbf{v}_j = \sum_{j=1}^n c_j \lambda_j \mathbf{v}_j$, where λ_j is the eigenvalue corresponding to \mathbf{v}_j . Multiplying by A again, we find that $A^2 \mathbf{w} = \sum_{j=1}^n c_j \lambda_j A\mathbf{v}_j = \sum_{j=1}^n c_j \lambda_j^2 \mathbf{v}_j$, and continuing in this way we see that for each $k = 1, 2, \dots$,

$$A^k \mathbf{w} = \sum_{j=1}^n c_j \lambda_j^k \mathbf{v}_j. \quad (12.1)$$

Now suppose that $|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_n|$. Then, assuming that $c_1 \neq 0$, the term in the sum (12.1) that dominates the others when k is very large is the first term: $c_1 \lambda_1^k \mathbf{v}_1$. Hence as k increases, $A^k \mathbf{w}$ looks more and more like a multiple of the first eigenvector \mathbf{v}_1 . Put another way, if we divide each side of (12.1) by λ_1^k , then we can write

$$\lambda_1^{-k} A^k \mathbf{w} = c_1 \mathbf{v}_1 + \sum_{j=2}^n c_j (\lambda_j / \lambda_1)^k \mathbf{v}_j, \quad (12.2)$$

and each of the terms $c_j (\lambda_j / \lambda_1)^k \mathbf{v}_j$, $j = 2, \dots, n$, approaches 0 as $k \rightarrow \infty$, since $|\lambda_j / \lambda_1| < 1$. Hence $\lim_{k \rightarrow \infty} \lambda_1^{-k} A^k \mathbf{w} = c_1 \mathbf{v}_1$.

The *power method* for computing the eigenvector of A corresponding to the eigenvalue of largest absolute value is usually implemented as follows:

Given a nonzero vector \mathbf{w} , set $\mathbf{y}^{(0)} = \mathbf{w} / \|\mathbf{w}\|$, and for $k = 1, 2, \dots$:

Set $\mathbf{y}^{(k)} = A\mathbf{y}^{(k-1)} / \|A\mathbf{y}^{(k-1)}\|$.

By normalizing the vector $\mathbf{y}^{(k)}$, one avoids possible problems with overflow or underflow, without affecting the convergence of the method. Since $\mathbf{y}^{(k)}$ converges to the eigenvector $\hat{\mathbf{v}}_1 \equiv \pm \mathbf{v}_1 / \|\mathbf{v}_1\|$, the corresponding eigenvalue λ_1 can be approximated in several different ways. If $\mathbf{y}^{(k)}$ were the true eigenvector, then the ratio of any component of $A\mathbf{y}^{(k)}$ to the corresponding component of $\mathbf{y}^{(k)}$ would be equal to λ_1 . Hence one way to approximate λ_1 would be to take the ratio $(A\mathbf{y}^{(k)})_i / \mathbf{y}^{(k)}_i$, for any nonzero component $i = 1, \dots, n$. Another way is to take the inner product $\langle A\mathbf{y}^{(k)}, \mathbf{y}^{(k)} \rangle \equiv \mathbf{y}^{(k)T} A\mathbf{y}^{(k)}$, which would be equal to $\langle \lambda_1 \hat{\mathbf{v}}_1, \hat{\mathbf{v}}_1 \rangle = \lambda_1 \langle \hat{\mathbf{v}}_1, \hat{\mathbf{v}}_1 \rangle = \lambda_1$, if $\mathbf{y}^{(k)}$ were the true normalized eigenvector $\hat{\mathbf{v}}_1$. The latter method is usually preferred, and the power method becomes:

Power Method for computing the eigenvalue of largest absolute value and the corresponding normalized eigenvector.

Given a nonzero vector \mathbf{w} , set $\mathbf{y}^{(0)} = \mathbf{w} / \|\mathbf{w}\|$. For $k = 1, 2, \dots$,
 Set $\tilde{\mathbf{y}}^{(k)} = A\mathbf{y}^{(k-1)}$.
 Compute $\lambda^{(k-1)} = \langle \tilde{\mathbf{y}}^{(k)}, \mathbf{y}^{(k-1)} \rangle$.
 Set $\mathbf{y}^{(k)} = \tilde{\mathbf{y}}^{(k)} / \|\tilde{\mathbf{y}}^{(k)}\|$.

We have argued that the power method converges to the eigenvector corresponding to the eigenvalue of largest absolute value, *provided* the absolute value of this eigenvalue is strictly greater than that of the others. It can be seen from (12.2) that the *rate* of convergence depends on the ratio $|\lambda_2/\lambda_1|$; i.e.,

$$\|\lambda_1^{-k} A^k \mathbf{w} - c_1 \mathbf{v}_1\| = O(|\lambda_2/\lambda_1|^k).$$

If one could decrease the ratio $|\lambda_2/\lambda_1|$, then one could achieve faster convergence.

Suppose, for example, that A has eigenvalues 10, 9, 8, 7, 6, 5. Then the power method applied to A will converge to the eigenvector corresponding to 10, with a convergence factor of .9. Consider the shifted matrix $A - 7I$, whose eigenvectors are the same as those of A and whose eigenvalues are 3, 2, 1, 0, -1, -2, as pictured below.



If the power method is applied to this shifted matrix, then it will converge to the same eigenvector (now corresponding to eigenvalue 3 in the shifted matrix), but the rate of convergence will be governed by the ratio $2/3 \approx .667$. We can recover the eigenvalues of the original matrix from those of the shifted matrix just by adding 7.

This leads to the idea of the *power method with a shift*.

Power Method with shift s for computing the eigenvalue farthest from s and the corresponding normalized eigenvector.

Given a shift s and a nonzero vector \mathbf{w} , set $\mathbf{y}^{(0)} = \mathbf{w}/\|\mathbf{w}\|$. For $k = 1, 2, \dots$,
 Set $\tilde{\mathbf{y}}^{(k)} = (A - sI)\mathbf{y}^{(k-1)}$.
 Compute $\lambda^{(k-1)} = \langle \tilde{\mathbf{y}}^{(k)}, \mathbf{y}^{(k-1)} \rangle + s$.
 Set $\mathbf{y}^{(k)} = \tilde{\mathbf{y}}^{(k)}/\|\tilde{\mathbf{y}}^{(k)}\|$.

This iteration converges to the eigenvector corresponding to the eigenvalue of $A - sI$ with largest absolute value. Note that in the previous example, if we shift by more than 7.5, say, we shift by 8 giving eigenvalues 2, 1, 0, -1, -2, -3, then the eigenvalue of largest absolute value is now the algebraically *smallest* eigenvalue, -3. In this case, the shifted power method converges to the smallest eigenvalue, 5, and corresponding eigenvector of the original matrix. Thus, by appropriately choosing shifts, the power method can be made to converge to the largest or smallest eigenvalue of a matrix with real eigenvalues. It cannot find the other eigenvalues, however, because the shift can never make one of the interior eigenvalues lie farthest from the origin. To find these eigenvalues one can use a method called *inverse iteration*. Another way to accomplish this when A is symmetric is through *deflation*.

Deflation

Assume that A is symmetric with eigenvalues $\lambda_1, \dots, \lambda_n$ satisfying $|\lambda_1| > |\lambda_2| > |\lambda_3| \geq \dots \geq |\lambda_n|$, and suppose that we have used the power method to compute the eigenvalue λ_1 of largest absolute value and the corresponding normalized eigenvector \mathbf{v}_1 . Suppose we have used an initial vector $\mathbf{w} = \sum_{j=1}^n c_j \mathbf{v}_j$. Knowing \mathbf{v}_1 , we can construct an initial vector $\hat{\mathbf{w}}$ that is a linear combination of only $\mathbf{v}_2, \dots, \mathbf{v}_n$:

$$\hat{\mathbf{w}} = \mathbf{w} - \langle \mathbf{w}, \mathbf{v}_1 \rangle \mathbf{v}_1 = \sum_{j=2}^n c_j \mathbf{v}_j.$$

This works because the eigenvectors of the symmetric matrix A are orthogonal; orthogonalizing any vector \mathbf{w} against one set of eigenvectors produces a vector that is a linear combination of only the other eigenvectors. This process of modifying the initial vector or other vectors generated by the algorithm so that they are orthogonal to the already computed eigenvector(s) is called *deflation*.

If the power method is run with initial vector $\hat{\mathbf{w}}$, then instead of converging to \mathbf{v}_1 , it will converge to \mathbf{v}_2 (assuming $c_2 \neq 0$), since

$$A^k \hat{\mathbf{w}} = \sum_{j=2}^n c_j A^k \mathbf{v}_j = \sum_{j=2}^n c_j \lambda_j^k \mathbf{v}_j = \lambda_2^k [c_2 \mathbf{v}_2 + \sum_{j=3}^n c_j (\lambda_j/\lambda_2)^k \mathbf{v}_j],$$

and each of the coefficients $(\lambda_j/\lambda_2)^k$, $j \geq 3$, converges to 0 as $k \rightarrow \infty$.

Now, in practice, we do not know \mathbf{v}_1 exactly, so the vector $\hat{\mathbf{w}}$ constructed above will likely have at least a small component in the direction of \mathbf{v}_1 . Even if it did not, rounding errors in the computation of powers of A times $\hat{\mathbf{w}}$ would perturb subsequent vectors so that they had components

in the direction of \mathbf{v}_1 , and the method would likely end up converging to \mathbf{v}_1 again. To prevent this, it is necessary to periodically orthogonalize the vectors in the power method against \mathbf{v}_1 . That is, if $\tilde{\mathbf{y}}^{(k)}$ is the vector produced at the k th step of the power method, then we replace $\tilde{\mathbf{y}}^{(k)}$ by

$$\tilde{\mathbf{y}}^{(k)} - \langle \tilde{\mathbf{y}}^{(k)}, \mathbf{v}_1 \rangle \mathbf{v}_1.$$

This need not be done at every step, but only occasionally to prevent the reappearance of the eigenvector \mathbf{v}_1 .

12.1.2 Inverse Iteration

Suppose the power method is applied to the matrix A^{-1} or, more generally, to $(A - sI)^{-1}$, where s is a given shift. Again let $\lambda_1, \dots, \lambda_n$ denote the eigenvalues of A and $\mathbf{v}_1, \dots, \mathbf{v}_n$ corresponding eigenvectors. Then the eigenvectors of $(A - sI)^{-1}$ are the same as those of A , and the corresponding eigenvalues are $(\lambda_1 - s)^{-1}, \dots, (\lambda_n - s)^{-1}$, since

$$A\mathbf{v}_j = \lambda_j\mathbf{v}_j \Rightarrow (A - sI)\mathbf{v}_j = (\lambda_j - s)\mathbf{v}_j \Rightarrow \frac{1}{\lambda_j - s}\mathbf{v}_j = (A - sI)^{-1}\mathbf{v}_j.$$

The eigenvalue of $(A - sI)^{-1}$ that is *farthest* from the origin is the one for which $|\lambda_j - s|$ is smallest. Thus if A has eigenvalues $1, 2, \dots, 10$, then A^{-1} has eigenvalues $1, 1/2, \dots, 1/10$, and the power method applied to A^{-1} will converge to 1. Moreover, its rate of convergence is governed by the ratio of the second largest eigenvalue of A^{-1} to the largest, which is $1/2$. Note that this is a faster rate of convergence than we would have obtained by using the power method on A with a shift; had we shifted by, say, 6 so that $A - 6I$ had eigenvalues $-5, -4, \dots, 4$, then the power method applied to the shifted matrix would have converged to the eigenpair corresponding to 1 in the original matrix, but the rate would have been governed by the ratio $4/5$. Thus the method of *inverse iteration* may provide faster convergence to the smallest eigenvalue of A .

Moreover, with a properly chosen shift, the inverse iteration method can be made to converge to *any* of the eigenvalues of A . For instance, suppose we choose $s = 3.2$. Then the eigenvalues of $A - sI$ are $-2.2, -1.2, -0.2, 0.8, \dots, 6.8$, so the eigenvalues of $(A - sI)^{-1}$ are approximately $-0.45, -0.83, -5, 1.25, \dots, .15$. The one that is farthest from the origin is -5 , and this corresponds to the interior eigenvalue 3 of A . Thus the inverse iteration method can be made to converge to interior eigenpairs of A . Additionally, if we already have a fairly good estimate of the eigenvalue that we are looking for, then the rate of convergence of the inverse iteration method can be greatly enhanced. Since the shift $s = 3.2$ is already fairly close to the eigenvalue 3, when we shift A by s , the shifted eigenvalue -0.2 will be much closer to the origin than the next closest one, and when we take the inverse of $A - sI$, the inverse of the shifted eigenvalue $1/(-0.2)$ will be much farther from the origin than the next farthest one. In this case, the next farthest eigenvalue of $(A - sI)^{-1}$ from the origin is 1.25 , and so the rate of convergence of inverse iteration is governed by the ratio $1.25/5 = .15$. Had we chosen an even closer shift, say, $s = 3.01$, then convergence would have been even faster. The eigenvalues of $(A - 3.01I)^{-1}$ are $-1/2.01, -1/1.01, -100, 1/.99, \dots, 1/6.99$, so the convergence rate of the power method applied to this matrix is given by the ratio $(1/.99)/100 \approx .01$.

The inverse iteration algorithm can be written as follows:

Inverse Iteration with shift s for computing the eigenvalue of A that is closest to s and the corresponding normalized eigenvector.

Given a shift s and a nonzero vector \mathbf{w} , set $\mathbf{y}^{(0)} = \mathbf{w}/\|\mathbf{w}\|$. For $k = 1, 2, \dots$,
Solve $(A - sI)\tilde{\mathbf{y}}^{(k)} = \mathbf{y}^{(k-1)}$ for $\tilde{\mathbf{y}}^{(k)}$.
Compute $\lambda^{(k-1)} = 1/\langle \tilde{\mathbf{y}}^{(k)}, \mathbf{y}^{(k-1)} \rangle + s$.
Set $\mathbf{y}^{(k)} = \tilde{\mathbf{y}}^{(k)}/\|\tilde{\mathbf{y}}^{(k)}\|$.

Note that we do not actually compute the inverse of $A - sI$ but we solve the linear system $(A - sI)\tilde{\mathbf{y}}^{(k)} = \mathbf{y}^{(k-1)}$ to obtain $\tilde{\mathbf{y}}^{(k)} = (A - sI)^{-1}\mathbf{y}^{(k-1)}$. Recall from Chapter 5 that this requires less work and is more stable than actually computing the inverse.

Computing Eigenvectors with Inverse Iteration

Often inverse iteration is used to compute eigenvectors, once the eigenvalues have been computed by other means, such as the *QR algorithm* to be described later. If the shift s is a very good approximation to an eigenvalue, then inverse iteration will converge to the corresponding eigenvector very quickly, as explained above. Note, however, that if s were an exact eigenvalue, then the matrix $A - sI$ would be singular, and if s is very close to an eigenvalue then one might expect rounding errors to prevent one from obtaining an accurate solution to the linear system: $(A - sI)\tilde{\mathbf{y}}^{(k)} = \mathbf{y}^{(k-1)}$. It turns out that although this is indeed the case, the error in solving this linear system lies mostly in the direction of the eigenvector that we are approximating. Hence this error does not detract from the performance of inverse iteration.

12.1.3 Rayleigh Quotient Iteration

Suppose one wishes to use inverse iteration to approximate an eigenpair, but one does not know a good shift s that is close to the desired eigenvalue. Since the inverse iteration method generates vectors that approximate an eigenvector of A , one can approximate the corresponding eigenvalue in the same way that this is done in the power method: $\lambda^{(k)} = \langle A\mathbf{y}^{(k)}, \mathbf{y}^{(k)} \rangle$. These values can then be used as successive shifts in the inverse iteration algorithm. This is called *Rayleigh quotient iteration*:

Rayleigh Quotient Iteration for computing an eigenvalue and corresponding eigenvector of A .

Given a nonzero vector \mathbf{w} , set $\mathbf{y}^{(0)} = \mathbf{w}/\|\mathbf{w}\|$ and $\lambda^{(0)} = \langle A\mathbf{y}^{(0)}, \mathbf{y}^{(0)} \rangle$. For $k = 1, 2, \dots$,
Solve $(A - \lambda^{(k-1)}I)\tilde{\mathbf{y}}^{(k)} = \mathbf{y}^{(k-1)}$ for $\tilde{\mathbf{y}}^{(k)}$.
Set $\mathbf{y}^{(k)} = \tilde{\mathbf{y}}^{(k)}/\|\tilde{\mathbf{y}}^{(k)}\|$.
Compute $\lambda^{(k)} = \langle A\mathbf{y}^{(k)}, \mathbf{y}^{(k)} \rangle$.

Note that the Rayleigh quotient, $\langle A\mathbf{y}^{(k)}, \mathbf{y}^{(k)} \rangle$, could be computed and used as an approximate eigenvalue in the inverse iteration algorithm as well, and sometimes this is done. It does require an

extra matrix vector multiplication, however.

How good an approximation to an eigenvalue is the Rayleigh quotient $\langle A\mathbf{v}, \mathbf{v} \rangle / \langle \mathbf{v}, \mathbf{v} \rangle$? Suppose $\mathbf{v} = \gamma\mathbf{v}_j + \mathbf{u}$, where \mathbf{v}_j is a normalized eigenvector of A corresponding to eigenvalue λ_j and \mathbf{u} is orthogonal to \mathbf{v}_j . Then

$$\frac{\langle A\mathbf{v}, \mathbf{v} \rangle}{\langle \mathbf{v}, \mathbf{v} \rangle} = \frac{\langle \gamma\lambda_j\mathbf{v}_j + A\mathbf{u}, \gamma\mathbf{v}_j + \mathbf{u} \rangle}{\langle \gamma\mathbf{v}_j + \mathbf{u}, \gamma\mathbf{v}_j + \mathbf{u} \rangle} = \frac{\gamma^2\lambda_j + \bar{\gamma}\langle A\mathbf{u}, \mathbf{v}_j \rangle + \langle A\mathbf{u}, \mathbf{u} \rangle}{\gamma^2 + \langle \mathbf{u}, \mathbf{u} \rangle}.$$

If A is *symmetric*, then $\langle A\mathbf{u}, \mathbf{v}_j \rangle = \langle \mathbf{u}, A\mathbf{v}_j \rangle = \langle \mathbf{u}, \lambda_j\mathbf{v}_j \rangle = 0$, and so this becomes

$$\frac{\langle A\mathbf{v}, \mathbf{v} \rangle}{\langle \mathbf{v}, \mathbf{v} \rangle} = \frac{\gamma^2\lambda_j + \langle A\mathbf{u}, \mathbf{u} \rangle}{\gamma^2 + \langle \mathbf{u}, \mathbf{u} \rangle} = \lambda_j + \frac{\langle A\mathbf{u}, \mathbf{u} \rangle - \lambda_j\langle \mathbf{u}, \mathbf{u} \rangle}{\|\mathbf{v}\|^2},$$

and the difference between the Rayleigh quotient and the eigenvalue λ_j satisfies

$$\left| \frac{\langle A\mathbf{v}, \mathbf{v} \rangle}{\langle \mathbf{v}, \mathbf{v} \rangle} - \lambda_j \right| \leq (\|A\| + |\lambda_j|)(\|\mathbf{u}\|/\|\mathbf{v}\|)^2 \leq 2\|A\|(\|\mathbf{u}\|/\|\mathbf{v}\|)^2.$$

The error in the Rayleigh quotient approximation to λ_j is proportional to the *square* of the norm of the portion of \mathbf{u} orthogonal to \mathbf{v}_j . Thus if \mathbf{v} is a fairly good approximation to the eigenvector \mathbf{v}_j , say, $\|\mathbf{v}\| = 1$ and $\|\mathbf{u}\| = .1$, then the Rayleigh quotient gives a much better approximation to λ_j , with error about $2\|A\|$ times .01. For a general nonsymmetric matrix, the term $\langle A\mathbf{u}, \mathbf{v}_j \rangle$ does not vanish and the error in the Rayleigh quotient approximation to λ_j is on the same order, $\|\mathbf{u}\|/\|\mathbf{v}\|$, as the error in the eigenvector approximation. For this reason, Rayleigh quotient iteration is used mainly for symmetric eigenproblems.

12.1.4 The QR Algorithm

We have thus far looked at methods for approximating selected eigenvalues and corresponding eigenvectors. Can one find all of the eigenvalues/vectors at once? The *QR algorithm* applies a sequence of orthogonal similarity transformations to A until it approaches an upper triangular matrix. Then the approximate eigenvalues are taken to be the diagonal entries of this (nearly) upper triangular matrix.

Recall the QR decomposition of a matrix, described in Section 5.6.2: Every n by n matrix A can be written in the form $A = QR$, where Q is an *orthogonal matrix* (its columns are orthonormal) and R is upper triangular. This decomposition was carried out using the *Gram-Schmidt algorithm*. We will later see a different way to compute the QR decomposition of a matrix. Consider now the following procedure: First factor $A \equiv A_0$ in the form $A_0 = QR$; then form the matrix $A_1 = RQ$. Note that $A_1 = (Q^T Q)RQ = Q^T(QR)Q = Q^T A_0 Q$, so that A_1 is similar to A_0 since $Q^T = Q^{-1}$. We can repeat this process of doing a QR factorization and then multiplying the factors in reverse order to obtain a sequence of matrices that are all orthogonally similar to A . This is called the *QR algorithm*:

Let $A_0 = A$. For $k = 0, 1, \dots$,

 Compute the QR factorization of A_k : $A_k = Q_k R_k$.

 Form the product: $A_{k+1} = R_k Q_k$.

The following theorem, which we state without proof, gives the basic convergence result for the QR algorithm:

Theorem 12.1.6. *Suppose the eigenvalues $\lambda_1, \dots, \lambda_n$ of A satisfy $|\lambda_1| > |\lambda_2| > \dots > |\lambda_n|$. Then the matrices A_k produced by the QR algorithm converge to an upper triangular matrix whose diagonal entries are the eigenvalues of A . If, in addition, $A = V\Lambda V^{-1}$, where V has an LU decomposition (i.e., pivoting is not required for Gaussian elimination on a matrix V of eigenvectors), then the rate of convergence to 0 of the elements $a_{ij}^{(k)}$, $i > j$, in the strict lower triangle is given by*

$$|a_{ij}^{(k)}| = O\left(\left|\frac{\lambda_i}{\lambda_j}\right|^k\right).$$

The technical condition that V have an LU decomposition ensures that the eigenvalues appear on the diagonal in descending order of magnitude. This is the usual case.

Hessenberg Form

The QR algorithm described so far is too expensive to be practical for a general matrix A . The first problem is that each step requires $O(n^3)$ operations. To reduce the number of operations per step, one can first apply a similarity transformation to put A into a form where the QR factorization is not so expensive. A general matrix A can first be reduced to *upper Hessenberg* form:

$$\begin{pmatrix} * & \dots & \dots & * \\ * & \ddots & & \vdots \\ & \ddots & \ddots & \vdots \\ & & & * & * \end{pmatrix},$$

which has only one nonzero diagonal below the main diagonal.

We will show how the QR factorization of an upper Hessenberg matrix can be computed using only $O(n^2)$ operations. Moreover, we will show that when the product RQ is formed it maintains this upper Hessenberg form. First, to reduce A to upper Hessenberg form, we will use *Householder reflections*.

A *hyperplane* is an $n - 1$ dimensional subspace of \mathbf{R}^n . It consists of all n -vectors that are orthogonal to some given vector \mathbf{w} . Thus, a hyperplane in 2-space is a line through the origin, and if \mathbf{w} is orthogonal to this line, then the line can be described as the set of all 2-vectors that are orthogonal to \mathbf{w} . A hyperplane in 3-space is a plane through the origin, and given a vector $\mathbf{w} \in \mathbf{R}^3$, the set of all vectors orthogonal to \mathbf{w} forms a (hyper)plane, as pictured below:

Suppose that \mathbf{v} is a vector not in the hyperplane orthogonal to \mathbf{w} . Then \mathbf{v} can be written as a linear combination of two vectors – one in the direction of \mathbf{w} and one orthogonal to \mathbf{w} and hence in the hyperplane. This second vector is the *orthogonal projection* of \mathbf{v} onto the hyperplane. Thus, assuming that \mathbf{w} is a unit vector ($\|\mathbf{w}\| = 1$), we can write

$$\mathbf{v} = \langle \mathbf{v}, \mathbf{w} \rangle \mathbf{w} + \mathbf{y},$$

where \mathbf{y} is the projection of \mathbf{v} onto the hyperplane; hence $\mathbf{y} = \mathbf{v} - \langle \mathbf{v}, \mathbf{w} \rangle \mathbf{w}$. Suppose that we wish to find the *reflection* of \mathbf{v} in the hyperplane orthogonal to \mathbf{w} . Then instead of subtracting the projection of \mathbf{v} onto \mathbf{w} from \mathbf{v} , as above, we should subtract twice this vector to obtain

$$\mathbf{v} - 2\langle \mathbf{v}, \mathbf{w} \rangle \mathbf{w}.$$

This reflected vector is pictured above.

Now, let P denote the n by n matrix $I - 2\mathbf{w}\mathbf{w}^T$, where again we assume that $\|\mathbf{w}\| = 1$. Then the formula above for the reflection of \mathbf{v} through the hyperplane orthogonal to \mathbf{w} is just $P\mathbf{v}$, since

$$P\mathbf{v} = (I - 2\mathbf{w}\mathbf{w}^T)\mathbf{v} = \mathbf{v} - 2\mathbf{w}(\mathbf{w}^T\mathbf{v}) = \mathbf{v} - 2\langle \mathbf{w}, \mathbf{v} \rangle \mathbf{w}.$$

Note that if \mathbf{v} already lies in the hyperplane; i.e., if \mathbf{v} is orthogonal to \mathbf{w} , then $P\mathbf{v} = \mathbf{v}$. Also, if we reflect \mathbf{w} in the hyperplane, then we just obtain $P\mathbf{w} = -\mathbf{w}$.

Note also that P is symmetric since $P^T = (I - \mathbf{w}\mathbf{w}^T)^T = I - \mathbf{w}\mathbf{w}^T = P$, and P is an orthogonal matrix since

$$P^T P = (I - 2\mathbf{w}\mathbf{w}^T)^T (I - 2\mathbf{w}\mathbf{w}^T) = (I - 2\mathbf{w}\mathbf{w}^T)^2 = I - 4\mathbf{w}\mathbf{w}^T + 4\mathbf{w}(\mathbf{w}^T\mathbf{w})\mathbf{w}^T = I.$$

How can reflections be used to reduce a matrix A to upper Hessenberg form via similarity transformations? Given an arbitrary vector \mathbf{v} , we can choose the hyperplane so that the reflection of \mathbf{v} in this hyperplane aligns with one of the coordinate axes; that is, so that $P\mathbf{v}$ has only one nonzero component, say, the first.

12.1.5 Google's PageRank

A popular and often effective form of information acquisition is submitting queries to Google.com. In fact, in the past second an average¹ of just over 1,200 searches were conducted on Google which supplies results to AOL, Yahoo, Netscape and Earthlink [?].

Suppose that we submit the query `mathematics` to Google. In the summer of 2006, Google returned the following pages (plus thousands more) in order:

- Mathematics in the Yahoo! Directory, dir.yahoo.com/Science/Mathematics/
- Eric Weisstein's World of Mathematics, mathworld.wolfram.com/
- Mathematics WWW Virtual Library, <http://www.math.fsu.edu/Virtual/index.php>
- Mathematics Archives, archives.math.utk.edu/

The mathematics category in the Yahoo! directory is deemed the "best" page related to the query. Being at the top of the list is an enviable position, since being listed 40th or 50th can essentially bury a page into obscurity, as it relates to the query.

Google searches billions of pages to create a linear-algebra problem that yields an "importance" ranking of pages. We will see that eigenvectors are deeply integrated into Google and the results that are returned from a query.

¹using data from January 2003

Lining Up the Web by Rank

A component of Google's success is the PageRank™ algorithm developed by Google's founders, Larry Page and Sergey Brin, who were graduate students at Stanford University when they developed the foundational ideas of Google [?]. The results of the algorithm are determined by the link structure of the WWW and involve no content of any page. As a historical note, Jon Kleinberg first suggested link analysis and spectral graph theory [?] as a means to improve web search, although the development and analysis differed from what Page and Brin would later develop.²

How does the PageRank™ algorithm work? A page A is considered more “important” if more pages link to it. However, Google also considers the importance of the pages that link to page A; links from “important” pages are given more weight. In the end, pages with high weight sums are given a high PageRank, which Google uses to order results of a query. PageRank is combined with text-matching algorithms to find pages that are both “important” and relevant to the query. Keeping the rankings updated presents its own issues since the structure of the Web changes continually as links are altered and pages are added and deleted.

How does the PageRank™ algorithm determine the weights of links? Try the following exercise:

Start at one of your favorite pages. Randomly select a link and click it. On the next page, randomly select a link. Continue this process of following random links.

There are two possible outcomes:

- You visit a page with no outgoing links—a *dangling node*, a dead end.
- Otherwise, you form a cycle by revisiting a page. You cannot visit new pages indefinitely, since there are finitely many pages.

Surfing with Markov

Your random walk in the exercise above can be modeled mathematically with a Markov chain. A Markov chain modeling a discrete systems requires determining the states of the system, and for each ordered pair of states i and j , the probability m_{ij} of moving from state i to state j . The Markov transition matrix is $M = (m_{ij})$.

For surfing the Web, the states are the indexed pages. Markov processes model the behavior of a random system whose probability distribution of possible next states depends only on the current state. The random walk (“surf”) that we performed had this attribute. The pages that we visit depend in part on the links available at the current page. For Google's purposes, it is undesirable for the Markov chain to get stuck at a dangling node, so we also include the possibility that the surfer jumps to another page not linked from the current one uniformly at random. Once we specify the probability of this random jump, we have a complete conceptual description of the Markov chain used by Google. But determining the transition matrix—the *Google matrix*³, with all the probabilities of moving from page i to page j , seems a monumental undertaking!

²Note, the citation for the reference to Kleinberg's work is later than that of Brin and Page's seminal paper. Brin and Page, however, cite Kleinberg's work as it appeared in 1998 in the *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*.

³Enter "Google matrix" as a query in Google and see what you find!

A Googled Matrix

Let W be the set of web pages contained in Google's index. Let n be the cardinality of W (the number of web pages in W). Notice that n varies with time. As of September 2005, n was about 8 billion and growing fast. Google's indexing of billions of web pages is a non-trivial task. We assume that such a set W has been constructed.

Consider a directed graph of the WWW where a vertex represents a web page, and an edge from vertex i to vertex j corresponds to a link from page i to page j . Let G be the $n \times n$ adjacency matrix of this graph; that is, $g_{ij} = 1$ if there is a hyperlink from page i to page j and 0 otherwise. The number of nonzeros in row i is the number of hyperlinks on page i , out of the very large number n of elements in the column. Hence, most entries in every row are zeros, and the matrix is sparse. Recall the size of n . The sparsity of G will be an asset in manipulating it on a computer. In particular, only the nonzero entries, along with column and row information, are stored. Because the average out-degree of pages on the web is about seven [?], this saves a factor on the order of half a billion in storage space and since n is growing over time while the average number of links on each page appear to remain about constant, the savings will only increase over time.

The information captured in G leads to the transition matrix. Let c_j and r_i be the column and row sums of G , respectively. That is,

$$c_j = \sum_{1 \leq i \leq n} g_{ij}, \quad r_i = \sum_{1 \leq j \leq n} g_{ij}. \quad (12.3)$$

Then c_k and r_k are the indegree and outdegree of the state (page) k .

Let p be the fraction of time that a random walk follows a link available on a page, so that $(1-p)$ is the fraction of time that a surfer jumps to a random web page chosen uniformly at random from W . Google typically sets $p = 0.85$.

The transition matrix M has elements

$$m_{ij} = p \left(\frac{g_{ij}}{r_i} \right) + \frac{1-p}{n}, \quad (12.4)$$

where m_{ij} is the probability that we visit page j in the next step, given that we are currently at page i .

Google's Eigenvectors

A non-negative left eigenvector that satisfies $\mathbf{v}M = \mathbf{v}$ (right-multiplication by the transition matrix is common notation for Markov processes) is called a *steady-state vector* of the Markov process (where \mathbf{v} is normalized such that $\sum \mathbf{v}_i = 1$, which results in a vector of probabilities). The theory of Markov chains shows that such a vector exists and, from any starting state, the limiting probability of visiting page i at time t as $t \rightarrow \infty$ is \mathbf{v}_i .

Google defines the PageRank of page i to be \mathbf{v}_i . Therefore, the largest element of \mathbf{v} corresponds to the page with the highest PageRank, the second largest to the page with the second highest PageRank, and so on. The limiting frequency that an infinitely dedicated random surfer visits any particular page is that page's PageRank.

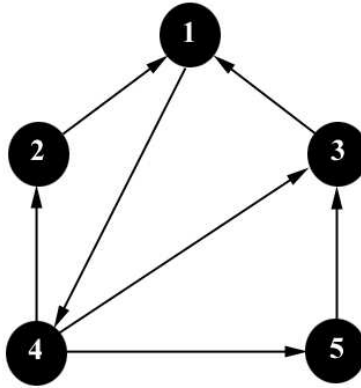


Figure 12.1: A small network of web pages.

The following theorem guarantees the uniqueness of the steady-state vector and that it will have positive entries:

Theorem 12.1.1 (Perron). *Every real square matrix P whose entries are all positive has a unique eigenvector with all positive entries, its corresponding eigenvalue has multiplicity one, and it is the dominant eigenvalue, in that every other eigenvalue has strictly smaller magnitude.*

Recall that the rows of M sum to 1. Therefore, $M\mathbf{1} = \mathbf{1}$, where $\mathbf{1}$ is the column vector of all ones. That is, $\mathbf{1}$ is a right eigenvector of M associated with the eigenvalue 1, most notably for our purposes having all positive entries. Perron's Theorem ensures that $\mathbf{1}$ is the unique right eigenvector with all positive entries, and hence its eigenvalue must be the dominant one. The right and left eigenvalues of a matrix are the same, therefore 1 is the dominant left eigenvalue as well. So, there exists a unique steady-state vector \mathbf{v} that satisfies $\mathbf{v}M = \mathbf{v}$. Normalizing this eigenvector so that $\sum v_i = 1$ gives a steady-state vector.

With the theory behind the PageRank™ algorithm in place, we apply it to the small network in Figure 12.1.

The adjacency matrix G of this network is

$$G = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

Recall that r_k is the outdegree of page k . Therefore, we see from G that page 1 has one link that points to page 4.

From G , we form the transition matrix. Note from (12.4) that if $g_{ij} = 0$, then $m_{ij} = (1-p)/n = .15/5 = .03$. Let us consider m_{14} since $g_{14} = 1 \neq 0$. From (12.4), we have $m_{14} = (0.85) \left(\frac{1}{1}\right) + 0.03 =$

0.88. Similarly, $m_{42} = (0.85) \left(\frac{1}{3}\right) + 0.03 \approx 0.3133$. Therefore,

$$M = \begin{pmatrix} .0300 & .0300 & .0300 & .8800 & .0300 \\ .8800 & .0300 & .0300 & .0300 & .0300 \\ .8800 & .0300 & .0300 & .0300 & .0300 \\ .0300 & .3133 & .3133 & .0300 & .3133 \\ .0300 & .0300 & .8800 & .0300 & .0300 \end{pmatrix}.$$

Each row of M sums to 1; (12.4) guarantees this. Since m_{ij} is the probability of moving from page i to page j , if a surfer is currently on page 1, there is an 88% chance that the surfer will visit page 4 next. What about the 3% probabilities that appear in M ? There is a 3% chance that a surfer will jump to page 2 without following a link on page 1.

Solving $\mathbf{v}M = \mathbf{v}$ yields

$$\mathbf{v} = (.2959, .1098, .2031, .2815, .1098).$$

Therefore, a random surfer will visit page 1 approximately 30% of the time and page 2 11% of the time. The PageRanks (the elements of \mathbf{v}) for this small Web are given in 12.1.

Page	PageRank
1	.2959
4	.2815
3	.2031
2	.1098
5	.1098

Table 12.1: PageRanks for the pages in the Web of Figure 12.1.

Compare these PageRanks to the network in Figure 12.1. Pages 1 and 3 have the same indegree and outdegree, yet page 1 is linked from pages that in the end have higher PageRank. Page 4 receives a high PageRank because it is linked from page 1. If a surfer lands on page 1 (which occurs about 29% of the time), then 85% of the time the surfer will follow a link. Page 1 links only to page 4. Therefore, the high PageRank of page 1 boosts the probability that page 4 is visited.

Getting Practical

This subsection has simplified many of the issues of information retrieval. As such, we now introduce some of the complexities inherent to a problem of this magnitude.

Note, the PageRankTM algorithm requires finding only the dominant eigenvector; a fact that will serve as a huge asset computationally. In particular, the power method can be used to find this vector.

Will the power method converge? The answer will be yes if M satisfies two conditions specified in the Perron-Frobenius theorem. First, the matrix must be irreducible. This occurs if every web page is reachable from every other page via a path of hyperlinks. This may not be satisfied

GOOGLE BOMBS



White House photo by Eric Draper



Much can be gained for those who can use knowledge of the underlying mathematics and computer science in information retrieval to their advantage. For example, in October of 2003, George Johnston initiated a *Google bomb* called the “miserable failure” bomb, which targeted President George W. Bush and detonated by late November of that same year. The result was Google returning the official White House Biography of the President as the highest ranked result to the query **miserable failure**. Putting together a Google bomb was relatively easy, it involved several web pages linking to a common web page with an agreed upon anchor text (the text that one clicks to go to the hyperlink). In the case of the “miserable failure” project, of the over 800 links that pointed to the Bush biography, only 32 used the phrase “miserable failure” in the anchor text. [?] This ignited a sort of political sparring match among the web saavy, and by January 2004, a “miserable failure” query returned results for Michael Moore, President Bush, Jimmy Carter and Hillary Clinton in the top four positions. As expected, Google is fully aware of such tactics and works to outsmart these and other initiatives that can dilute the effectiveness of its results. For more information, search the internet on “Google bombs” or “link farms” or see the recent text on information retrieval by Langville and Meyer [?].

in the connectivity matrix of the network as dangling nodes alone prohibit such a property in this matrix. However, the presence of the $(1 - p)/n$ term in (12.4) modeled a surfer directly jumping from any page to another. In such a way, the resulting matrix M is indeed irreducible and simultaneously aperiodic, which satisfies the second condition of the Perron-Frobenius theorem. Therefore, convergence is guaranteed.

For simplicity of exposition, assume $|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_n|$. Then, from (??), we know that the rate of convergence of the Power Method is $|\lambda_2|/|\lambda_1|$. From Perron’s theorem, we know that $\lambda_1 = 1$ and $\lambda_i < 1$ for $i > 1$. In [?], the rate of convergence of the power method is shown to be $|\lambda_2|/|\lambda_1|$ with λ_2 bounded above by $p = 0.85$. Note, this implies the scalability of this method even as the size of the underlying problem increases.

Even more powerful is the fact that the entire matrix M need not be stored for the power method. In fact, it is enough to know p , n , the nonzeros of G , and the out-degree of each page. For more information on this, see [?]. This paper supplies an extensive survey of research on all facets of PageRank, which involves many complexities beyond the scope of this paper.

Remarks before Logging Out

To determine the PageRanks of the pages that it indexes, Google solves a linear-algebra problem of massive proportions—finding an eigenvector of a matrix of order 8 billion.

Again, keep in mind that PageRank alone does not determine the order of the list returned by Google for a query. For example, <http://mathworld.wolfram.com/> has a lower PageRank than <http://www.google.com>. However, our query `mathematics` does not produce <http://www.google.com> in the first 100 pages returned from Google. Why? Using text-matching algorithms, Google determines the relevance of a page to a query and combines this with a page's PageRank to determine the final ranking for a query.

To reflect the subtlety of language and the challenges of text-matching, suppose that we submit the query `show boat`. The page that tops the list returned is <http://www.imdb.com/title/tt0044030/>, which gives details on the 1951 film “Show Boat” based on the musical by Jerome Kern and Oscar Hammerstein II. For the query `boat show`, topping the list is <http://www.showmanagement.com/>, containing information on the Show Management company that produces boat shows throughout the year.

By inverting the order of the words in our query, we changed the list of pages returned. This is what we want. If we are looking for information on boat shows, we probably do not want information on the musical “Show Boat.” Carefully designed text-matching algorithms play the role in differentiating between pages that have the words “boat” and “show” in their content.

How do text-matching algorithms work? How is relevance to a string quantified? These questions step into thought-provoking areas beyond the scope of this textbook. The answers, however, may simply be a query on Google away.

