Chapter 4

Linear Algebra I, Theory and Conditioning

4.1 Introduction

Linear algebra and calculus are the basic tools of quantitative science. The operations of linear algebra include solving systems of equations, finding subspaces, solving least squares problems, factoring matrices, and computing eigenvalues and eigenvectors. In practice most of these operations will be done by software packages that you buy or download. This chapter discusses formulation and condition number of the main problems in computational linear algebra. Chapter 5.1 discusses algorithms.

Conditioning is the primary concern in many practical linear algebra computations. Easily available linear algebra software is *stable* in the sense that the results are as accurate as the conditioning of the problem allows. Unfortunately, condition numbers as large as 10^{18} occur in not terribly large or rare practical problems. The results of such a calculation in double precision would be completely unreliable.

If a computational method for a well conditioned problem is unstable (much less accurate than its conditioning allows), it is likely because one of the subproblems is ill conditioned. For example, the problem of computing the matrix exponential, e^A, may be well conditioned while the problem of computing the eigenvalues and eigenvectors of A is ill conditioned. A stable algorithm for computing e^A in that case must avoid using the eigenvalues and eigenvectors of A, see Exercise 12.

The condition number measures how small perturbations in the data affect the answer. This is called *perturbation theory* in linear algebra. Suppose[1] A is a matrix and $f(A)$ is the solution of a linear algebra problem involving A, such as x that satisfies $Ax = b$, or λ and v that satisfies $Av = \lambda v$. Perturbation theory seeks to estimate $\Delta f = f(A + \Delta A) - f(A)$ when ΔA is small. Usually, this amounts to calculating the derivative of f with respect to A. We often do this by applying implicit differentiation to the relevant equations (such as $Ax = b$).

It often is helpful to simplify the results of perturbation calculations using simple bounds that involve vector or matrix *norms*. For example, suppose we want to say that all the entries in ΔA or Δv are small. For a vector, v, or a matrix, A, the norm, $\|v\|$ or $\|A\|$, is a number that characterizes the size of v or A. Using norms, we can say that the relative size of a perturbation in A is $\|\Delta A\| / \|A\|$.

The condition number of a problem involving A depends on the problem as well as on A. For example, the condition number of $f(A) = A^{-1}b$, the problem of finding x so that $Ax = b$, informally[2] is given by

$$\|A\| \left\|A^{-1}\right\| . \tag{4.1}$$

The problem of finding the eigenvectors of A has a condition number that does

[1]This notation replaces our earlier $A(x)$. In linear algebra, A always is a matrix and x never is a matrix.

[2]To get this result, we not only maximize over ΔA but also over b. If the relative error really were increased by a factor on the order of $\|A\| \left\|A^{-1}\right\|$ the finite element method, which is the main computational technique for structural analysis, would not work.

not resemble (4.1). For example, finding eigenvectors of A can be well conditioned even when $\|A^{-1}\|$ is infinite (A is singular).

There are several ways to represent an $n \times n$ matrix in the computer. The simplest is to store the n^2 numbers in an $n \times n$ array. If this direct storage is efficient, we say A is *dense*. Much of the discussion here and in Chapter 5.1 applies mainly to dense matrices. A matrix is *sparse* if storing all its entries directly is inefficient. A modern (2006) desktop computer has enough memory for n^2 numbers if if n is less than about[3] $50,000$. This makes dense matrix methods impractical for solving systems of equations with more than $50,000$ variables. The computing time solving $n = 50,000$ linear equations in this way would be about a day. Sparse matrix methods can handle larger problems and often give faster methods even for problems that can be handled using dense matrix methods. For example, finite element computations often lead to sparse matrices with orders of magnitude larger n that can be solved in minutes

One way a matrix can be sparse is for most of its entries to be zero. For example, discretizations of the Laplace equation in three dimensions have as few as seven non-zero entries per row, so that $7/n$ is the fraction of entries of A that are not zero. Sparse matrices in this sense also arise in circuit problems, where a non-zero entry in A corresponds to a direct connection between two elements in the circuit. Such matrices may be stored in *sparse matrix format*, in which we keep lists noting which entries are not zero and the values of the non-zero elements. Computations with such sparse matrices try to avoid *fill in*. For example, they would avoid explicit computation of A^{-1} because most of its entries are not zero. Sparse matrix software has heuristics that often do very well in avoiding fill in. The interested reader should consult the references.

In some cases it is possible to compute the *matrix vector product* $y = Ax$ for a given x efficiently without calculating the entries of A explicitly. One example is the *discrete Fourier transform* (DFT) described in Chapter 1. This is a *full* matrix (every entry different from zero) with n^2 non-zero entries, but the FFT (*fast Fourier transform*) algorithm computes $y = Ax$ in $O(n \log(n))$ operations. Another example is the *fast multipole method* that computes forces from mutual electrostatic interaction of n charged particles with b bits of accuracy in $O(nb)$ work. Many finite element packages never *assemble* the stiffness matrix, A.

Computational methods can be *direct* or *iterative*. A direct method would get the exact answer in exact arithmetic using a predetermined number of arithmetic operations. For example, Gauss elimination computes the *LU* factorization of A using $O(n^3)$ operations. Iterative methods produce a sequence of approximate solutions that converge to the exact answer as the number of iterations goes to infinity. They usually are faster than direct methods for very large problems, particularly when A is sparse.

[3]With n^2 floating point numbers and 8 bytes per number (double precision), we need $50,000^2 \times 8 = 2 \cdot 10^{10}$ bytes, which is 20GBytes.

4.2 Review of linear algebra

This section recalls some aspects of linear algebra we make use of later. It is not a substitute for a course on linear algebra. We make use of many things from linear algebra, such as matrix inverses, without explanation. People come to scientific computing with vastly differing points of view in linear algebra. This section should give everyone a common language.

4.2.1 Vector spaces

Linear algebra gets much of its power through the interaction between the abstract and the concrete. Abstract linear transformations are represented by concrete arrays of numbers forming a matrix. The set of solutions of a *homogeneous* system of equations forms an abstract subspace of R^n that we can try to characterize. For example, a basis for such a subspace may be computed by factoring a matrix in a certain way.

A vector space is a set of elements that may be added and multiplied by *scalar* numbers[4] (either real or complex numbers, depending on the application). Vector addition is commutative ($u + v = v + u$) and associative ($(u + v) + w = u + (v + w)$). Multiplication by scalars is distributive over vector addition ($a(u + v) = au + av$ and $(a + b)u = au + bu$ for scalars a and b and vectors u and v). There is a unique zero vector, 0, with $0 + u = u$ for any vector u.

The standard vector spaces are R^n (or C^n), consisting of column vectors

$$u = \begin{pmatrix} u_1 \\ u_2 \\ \cdot \\ \cdot \\ u_n \end{pmatrix}$$

where the *components*, u_k, are arbitrary real (or complex) numbers. Vector addition and scalar multiplication are done componentwise.

If V is a vector space and $V' \subset V$, then we say that V' is a *subspace* of V if V' is also a vector space with the same vector addition and scalar multiplication operations. We may always add elements of V' and multiply them by scalars, but V' is a subspace if the result always is an element of V'. We say V' is a subspace if it is *closed* under vector addition and scalar multiplication. For example, suppose $V = R^n$ and V' consists of all vectors whose components sum to zero ($\sum_{k=1}^n u_k = 0$). If we add two such vectors or multiply by a scalar, the result also has the zero sum property. On the other hand, the set of vectors whose components sum to one ($\sum_{k=1}^n u_k = 1$) is not closed under vector addition or scalar multiplication.

[4]Physicists use the word "scalar" in a different way. For them, a scalar is a number that is the same in any coordinate system. The components of a vector in a particular basis are not scalars in this sense.

A *basis* for vector space V is a set of vectors f_1, …, f_n so that any $u \in V$ may be written in a unique way as a *linear combination* of the vectors f_k:

$$u = u_1 f_1 + \cdots + u_n f_n \ ,$$

with scalar *expansion coefficients* u_k. The standard vector spaces R^n and C^n have standard bases e_k, the vector with all zero components but for a single 1 in position k. This is a basis because

$$u = \begin{pmatrix} u_1 \\ u_2 \\ \cdot \\ \cdot \\ u_n \end{pmatrix} = u_1 \begin{pmatrix} 1 \\ 0 \\ \cdot \\ \cdot \\ 0 \end{pmatrix} + u_2 \begin{pmatrix} 0 \\ 1 \\ \cdot \\ \cdot \\ 0 \end{pmatrix} + \cdots + u_n \begin{pmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ 1 \end{pmatrix} = \sum_{k=1}^{n} u_k e_k \ .$$

In view of this, there is little distinction between coordinates, components, and expansion coefficients, all of which are called u_k. If V has a basis with n elements, we say the *dimension* of V is n. It is possible to make this definition because of the theorem that states that every basis of V has the same number of elements. A vector space that does not have a finite basis is called *infinite dimensional*[5]. The vector space of all polynomials (with no limit on their degree) is infinite dimensional.

Polynomials provide other examples of vector spaces. A polynomial in the variable x is a linear combination of powers of x, such as $2 + 3x^4$, or 1, or $\frac{1}{3}(x-1)^2(x^3-3x)^6$. We could multiply out the last example to write it as a linear combination of powers of x. The degree of a polynomial is the highest power that it contains. The complicated product above has degree 20. One vector space is the set, P_d, of all polynomials of degree not more than d. This space has a basis consisting of $d+1$ elements:

$$f_0 = 1 \ , \ \ f_1 = x \ , \ \ \ldots \ , \ \ f_d = x^d \ .$$

Another basis of P_3 consists of the "Hermite" polynomials

$$H_0 = 1 \ , \ \ H_1 = x \ , \ \ H_2 = x^2 - 1 \ , \ \ H_3 = x^3 - 3x \ .$$

These are useful in probability because if x is a standard normal random variable, then they are uncorrelated:

$$E\left[H_j(X)H_k(X)\right] = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} H_j(x)H_k(x)e^{-x^2/2}dx = 0 \quad \text{if } j \neq k.$$

Still another basis of P_3 consists of Lagrange interpolating polynomials for the points 1, 2, 3, and 4:

$$l_1 = \frac{(x-2)(x-3)(x-4)}{(1-2)(1-3)(1-4)} \ , \ \ l_2 = \frac{(x-1)(x-3)(x-4)}{(2-1)(2-3)(2-4)} \ ,$$

[5]An infinite dimensional vector space might have an infinite basis, whatever that might mean.

$$l_3 = \frac{(x-1)(x-2)(x-4)}{(3-1)(3-2)(3-4)} \;, \quad l_4 = \frac{(x-1)(x-2)(x-3)}{(3-1)(3-2)(3-4)} \;.$$

These are useful for interpolation because, for example, $l_1(1) = 1$ while $l_2(1) = l_3(1) = l_4(1) = 0$. If we want $u(x)$ to be a polynomial of degree 3 taking specified values $u(1) = u_1$, $u(2) = u_2$, $u(3) = u_3$, and $u(4) = u_4$, the answer is

$$u(x) = u_1 l_1(x) + u_2 l_2(x) + u_3 l_3(x) + u_4 l_4(x) \;.$$

For example, at $x = 2$, the l_2 term takes the value u_2 while the other three terms are zero. The Lagrange interpolating polynomials are linearly independent because if $0 = u(x) = u_1 l_1(x) + u_2 l_2(x) + u_3 l_3(x) + u_4 l_4(x)$ for all x then in particular $u(x) = 0$ at $x = 1$, 2, 3, and 4, so $u_1 = u_2 = u_3 = u_4 = 0$.

If $V' \subset V$ is a subspace of dimension m of a vector space of dimension n, then it is possible to find a basis, f_k, of V so that the first m of the f_k form a basis of V'. For example, if $V = P_3$ and V' is the polynomials that vanish at $x = 2$ and $x = 3$, we can take

$$f_1 = (x-2)(x-3) \;, \quad f_2 = x(x-2)(x-3) \;, \quad f_3 = 1 \;, \quad f_4 = x \;.$$

Note the general (though not universal) rule that the dimension of V' is equal to the dimension of V minus the number of constraints or conditions that define V'. Whenever V' is a *proper* subspace of V, there is some $u \in V$ that is not in V', $m < n$. One common task in computational linear algebra is finding a *well conditioned* basis for a subspace specified in some way.

4.2.2 Matrices and linear transformations

Suppose V and W are vector spaces. A *linear transformation* from V to W is a function that produces an element of W for any element of V, written $w = Lv$, that is linear. Linear means that $L(v_1 + v_2) = Lv_1 + Lv_2$ for any two vectors in V, and $Lxv = xLv$ for any scalar x and vector $v \in V$. By convention we write Lv instead of $L(v)$ even though L represents a function from V to W. This makes algebraic manipulation with linear transformations look just like algebraic manipulation of matrices, deliberately blurring the distinction between linear transformations and matrix multiplication. The simplest example is the situation of $V = R^n$, $W = R^m$, and $Lu = A \cdot u$, for some $m \times n$ matrix A. The notation $A \cdot u$ means that we should multiply the vector u by the matrix A. Most of the time we leave out the dot.

Any linear transformation between finite dimensional vector spaces may be represented by a matrix. Suppose $f_1, \ldots, f_n$ is a basis for V, and $g_1, \ldots, g_m$ is a basis for W. For each k, the linear transformation of f_k is an element of W and may be written as a linear combination of the g_j:

$$Lf_k = \sum_{j=1}^{m} a_{jk} g_j \;.$$

Because the transformation is linear, we can calculate what happens to a vector $u \in V$ in terms of its expansion $u = \sum_k u_k f_k$. Let $w \in W$ be the *image* of u, $w = Lu$, written as $w = \sum_j w_j g_j$. We find

$$w_j = \sum_{k=1}^{n} a_{jk} u_k \ ,$$

which is ordinary matrix-vector multiplication.

The matrix that represents L depends on the basis. For example, suppose $V = P_3$, $W = P_2$, and L represents differentiation:

$$L\left(p_0 + p_1 x + p_2 x^2 + p_3 x^3\right) = \frac{d}{dx}\left(p_0 + p_1 x + p_2 x^2 + p_3 x^3\right) = p_1 + 2p_2 x + 3p_3 x^2 \ .$$

If we take the basis 1, x, x^2, x^3 for V, and 1, x, x^2 for W, then the matrix is

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3 \end{pmatrix}$$

The matrix would be different if we used the Hermite polynomial basis for V. (See Exercise 1).

Conversely, an $m \times n$ matrix, A, represents a linear transformation from R^n to R^m (or from C^n to C^m). We denote this transformation also by A. If $v \in R^m$ is an m component column vector, then $w = Av$, the matrix vector product, determines $w \in R^n$, a column vector with n components. As before, the notation deliberately is ambiguous. The matrix A is the matrix that represents the linear transformation A using standard bases of R^n and R^m.

A matrix also may represent a change of basis within the same space V. If $f_1, \ldots, f_n$, and $g_1, \ldots, g_n$ are different bases of V, and u is a vector with expansions $u = \sum_k v_k f_k$ and $u = \sum_j w_j g_j$, then we may write

$$\begin{pmatrix} v_1 \\ \cdot \\ \cdot \\ v_n \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & \cdot & a_{1n} \\ a_{21} & a_{22} & \cdot & a_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & \cdot & a_{nn} \end{pmatrix} \begin{pmatrix} w_1 \\ \cdot \\ \cdot \\ w_n \end{pmatrix}$$

As before, the matrix elements a_{jk} are the expansion coefficients of g_j with respect to the f_k basis[6]. For example, suppose $u \in P_3$ is given in terms of Hermite polynomials or simple powers: $u = \sum_{j=0}^{3} v_j H_j(x) = \sum_{k=0}^{3} w_j x^j$, then

$$\begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix}$$

[6]We write a_{jk} for the (j,k) entry of A.

We may reverse the change of basis by using the inverse matrix:

$$\begin{pmatrix} w_1 \\ \cdot \\ \cdot \\ w_n \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & \cdot & a_{1n} \\ a_{21} & a_{22} & \cdot & a_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & \cdot & a_{nn} \end{pmatrix}^{-1} \begin{pmatrix} v_1 \\ \cdot \\ \cdot \\ v_n \end{pmatrix}$$

Two bases must have the same number of elements because only square matrices can be invertible.

Composition of linear transformations corresponds to matrix multiplication. If L is a linear transformation from V to W, and M is a linear transformation from W to a third vector space, Z, then ML is the *composite* tranformation that takes V to[7] W. The composite of L and M is defined if the *target* (or *range*) of L, is the same as the *source* (or *domain*) of M, W in this case. If A is an $m \times n$ matrix and and B is $p \times q$, then the target of A is $W = R^n$ and the source of B is R^q. Therefore, the composite AB is defined if $n = p$. This is the condition for the matrix product $A \cdot B$ (usually written without the dot) to be defined. The result is a transformation from $V = R^m$ to $Z = R^p$, i.e., the $m \times p$ matrix AB.

For vector spaces V and W, the set of linear transformations from V to W forms a vector space. We can add two linear transformations and multiply a linear transformation by a scalar. This applies in particular to $m \times n$ matrices, which represent linear transformations from R^n to R^m. The entries of $A + B$ are $a_{jk} + b_{jk}$. An $n \times 1$ matrix has a single column and may be thought of as a column vector. The product Au is the same whether we consider u to be a column vector or an $n \times 1$ matrix. A $1 \times n$ matrix has a single row and may be thought of as a *row vector*. If r is such a row vector, the product rA makes sense, but Ar does not. It is useful to distinguish between row and column vectors although both are determined by n components. The product ru is a 1×1 matrix ($(1 \times n) \cdot (n \times 1)$ gives 1×1), i.e., a scalar.

If the source and targets are the same, $V = W$, or $n = m = p = q$, then both composites LM and ML both are defined, but they probably are not equal. Similarly, if A and B are $n \times n$ matrices, then AB and BA are defined, but $AB \neq BA$ in general. If A, B, and C are matrices so that the products AB and BC both are defined, then the products $A \cdot (BC)$ and $(AB) \cdot C$ also are defined. The *associative* property of matrix multiplication is the fact that these are equal: $A(BC) = (AB)C$. In actual computations it may be better to compute AB first then multiply by C than to compute BC first the compute $A(BC)$. For example suppose u is a $1 \times n$ row vector, and B and C are $n \times n$ matrices. Then $uB = v$ is the product of a row vector and a square matrix, which takes $O(n^2)$ additions and multiplications (*flops*) to compute, and similarly for $vC = (uB)C$. On the other hand, BC is the product of $n \times n$ matrices, which takes $O(n^3)$ flops to compute. Thus $(uB)C$ takes an order of magnitude fewer flops to compute than $u(BC)$.

[7]First v goes to $w = Lv$, then w goes to $z = Mw$. In the end, v has gone to $M(Lv)$.

If A is an $m \times n$ matrix with real entries a_{jk}, the *transpose* of A, written A^*, is the $n \times m$ matrix whose (j,k) entry is $a^*_{jk} = a_{kj}$. If A has complex entries, then A^*, the *adjoint* of A, is the $n \times m$ matrix wth entries $a^*_{jk} = \overline{a}_{kj}$ ($\overline{a}$ is the complex conjugate of a.). If the entries of A are real, the adjoint and transpose are the same. The transpose often is written A^t, but we use the same notation for adjoint and transpose. The transpose or adjoint of a column vector is a row vector with the same number of entries, and vice versa. By convention, the vector spaces R^n and C^n consist of n component column vectors. A real square ($n = m$) matrix is *symmetric* if $A = A^*$. A complex square matrix is *self-adjoint* if $A^* = A$. We often use the term "self-adjoint" for either case.

If u and v are n component column vectors ($u \in C^n$, $v \in C^n$), their *standard inner product* is

$$\langle u, v \rangle = u^* v = \sum_{k=1}^{n} \overline{u}_k v_k \; . \tag{4.2}$$

The inner product is *antilinear* in its first argument, which means that

$$\langle xu + yv, w \rangle = \overline{x} \langle u, w \rangle + \overline{y} \langle v, w \rangle \; ,$$

for complex numbers x and y. It is linear in the second argument:

$$\langle u, xv + yw \rangle = x \langle u, w \rangle y + \langle v, w \rangle \; .$$

It determines the standard *euclidean norm* $\|u\| = \langle u, u \rangle^{1/2}$. The complex conjugates are not needed when the entries of u and v are real. The reader should check that $\overline{\langle u, v \rangle} = \langle v, u \rangle$. The adjoint of A is determined by the requirement that $\langle A^* u, v \rangle = \langle u, Av \rangle$ for all u and v.

4.2.3 Vector norms

A *norm* is a way to describe the size of a vector. It is a single number extracted from the information describing u, which we write $\|u\|$ (read "the norm of u"). There are several different norms that are useful in scientific computing. We say $\|u\|$ is a norm if it has the following properties. First, $\|u\| \geq 0$, with $\|u\| = 0$ only when $u = 0$. Second, it should be *homogeneous*: $\|xu\| = |x| \, \|u\|$ for any scalar, x. Third, it should satisfy the *triangle inequality*, $\|u + v\| \leq \|u\| + \|v\|$, for any two vectors u and v.

There are several simple norms for R^n or C^n that have names. One is the l^1 norm

$$\|u\|_1 = \|u\|_{l^1} = \sum_{k=1}^{n} |u_k| \; .$$

Another is the l^∞ norm, also called the *sup* norm or the *max* norm:

$$\|u\|_\infty = \|u\|_{l^\infty} = \max_{k=1,\ldots,n} |u_k| \; .$$

Another is the l^2 norm, also called the *Euclidian* norm

$$\|u\|_2 = \|u\|_{l^2} = \left(\sum_{k=1}^{n} |u_k|^2 \right)^{1/2} = \langle u, u \rangle^{1/2} .$$

The l^2 norm is natural for vectors representing positions or velocities in three dimensional space. If the components of $u \in R^n$ represent probabilities, the l^1 norm might be more appropriate. In some cases we may have a norm defined indirectly or with a definition that is hard to turn into a number. For example in the vector space P_3 of polynomials of degree 3, we can define a norm

$$\|p\| = \max_{a \leq x \leq b} |p(x)| . \tag{4.3}$$

There is no simple formula for $\|p\|$ in terms of the coefficients of p.

The notion of vector norms is not perfect. For one thing, the choice of norm seems arbitrary in many cases . For example, what norm should we use for the two dimensional subspace of P_3 consisting of polynomials that vanish when $x = 2$ and $x = 3$? Another criticism is that norms may not make dimensional sense if the different components of u have different units. This might happen, for example, if the components of u represent different factors (or variables) in a linear regression. The first factor, u_1, might be age of a person, the second, u_2, income, the third the number of children. In some units, we might get (because the computer stores only numbers, not units)

$$u = \begin{pmatrix} 45 \\ 50000 \\ 2 \end{pmatrix} \tag{4.4}$$

In situations like these we can define for example, a *dimensionless* version of the l^1 norm:

$$\|u\| = \sum_{k=1}^{n} \frac{1}{\overline{u}_k} \cdot |u_k| ,$$

where $\overline{u}_k$ is a typical value of a quantity with the units of u_k in the problem at hand.

We can go further and use the basis

$$f_k = \overline{u}_k e_k \tag{4.5}$$

of R^n. In this basis, the components of u are $\widetilde{u}_k = \frac{1}{\overline{u}_k} u_k$, which are dimensionless. This is *balancing* or *diagonal scaling.* For example, if we take a typical age to be $\overline{u}_1 = 40$(years), a typical salary to be $\overline{u}_2 = 60,000$(dollars/year), and a typical number of children to be $\overline{u}_3 = 2.3$ (US national average), then the normalized components are comparable well scaled numbers: $(f_1, f_2, f_3) = (1.125, .833, .870)$. The matrix representing a linear transformation in the f_k basis is likely to be better conditioned than the one using the e_k basis.

4.2.4 Norms of matrices and linear transformations

Suppose L is a linear transformation from V to W. If we have norms for the spaces V and W, we can define a corresponding norm of L, written $\|L\|$, as the largest amount by which it stretches a vector:

$$\|L\| = \max_{u \neq 0} \frac{\|Lu\|}{\|u\|} \ . \tag{4.6}$$

The norm definition (4.6) implies that for all u,

$$\|Lu\| \leq \|L\| \cdot \|u\| \ . \tag{4.7}$$

Moreover, $\|L\|$ is the *sharp constant* in the inequality (4.7) in the sense that if $\|Lu\| \leq C \cdot \|u\|$ for all u, then $C \geq \|L\|$. Thus, (4.6) is equivalent to saying that $\|L\|$ is the sharp constant in (4.7).

The different vector norms give rise to different matrix norms. The matrix norms corresponding to certain standard vector norms are written with corresponding subscripts, such as

$$\|L\|_{l^2} = \max_{u \neq 0} \frac{\|Lu\|_{l^2}}{\|u\|_{l^2}} \ . \tag{4.8}$$

For $V = W = R^n$, it turns out that (for the linear transformation represented in the standard basis by A)

$$\|A\|_{l^1} = \max_k \sum_j |a_{jk}| \ ,$$

and

$$\|A\|_{l^\infty} = \max_j \sum_k |a_{jk}| \ .$$

Thus, the l^1 matrix norm is the maximum *column sum* while the max norm is the maximum *row sum*. Other norms are hard to compute explicitly in terms of the entries of A. People often say that $\|L\|_{l^2}$ is the largest *singular value* of L, but this is not too helpful because (4.8) is the definition of the largest singular value of L.

Any norm defined by (4.6) in terms of vector norms has several properties derived from corresponding properties of vector norms. One is homogeneity: $\|xL\| = |x| \, \|L\|$. Another is that $\|L\| \geq 0$ for all L, with $\|L\| = 0$ only for $L = 0$. The triangle inequality for vector norms implies that if L and M are two linear transformations from V to W, then $\|L + M\| \leq \|L\| + \|M\|$. Finally, we have $\|LM\| \leq \|L\| \, \|M\|$. This is because the composite transformation stretches no more than the product of the individual maximum stretches:

$$\|M(Lu)\| \leq \|M\| \, \|Lu\| \leq \|M\| \, \|L\| \, \|u\| \ .$$

Of course, all these properties hold for matrices of the appropriate sizes.

All of these norms have uses in the theoretical parts of scientific computing, the l^1 and l^∞ norms because they are easy to compute and the l^2 norm because it is invariant under orthogonal transformations such as the discrete Fourier transform. The norms are not terribly different from each other. For example, $\|A\|_{l^1} \leq n\|A\|_{l^\infty}$ and $\|A\|_{l^\infty} \leq n\|A\|_{l^1}$. For $n \leq 1000$, this factor of n may not be so important if we are asking, for example, about catastrophic ill-conditioning.

4.2.5 Eigenvalues and eigenvectors

Let A be an $n \times n$ matrix, or a linear transformation from V to V. We say that λ is an *eigenvalue*, and $r \neq 0$ the corresponding (right) *eigenvector* if

$$Ar = \lambda r \; .$$

Eigenvalues and eigenvectors are useful in understanding dynamics related to A. For example, the differential equation $\frac{du}{dt} = \dot{u} = Au$ has solutions $u(t) = e^{\lambda t}r$. Moreover, eigenvalues and their relatives, *singular values*, are the basis of *principal component analysis* in statistics. In general, eigenvalues and eigenvectors may be complex even though A is real. This is one reason people work with complex vectors in C^n, even for applications that seem to call for R^n.

Although their descriptions seem similar, the *symmetric eigenvalue problem* (A symmetric for real A or self-adjoint for complex A), is vastly different from the general, or *unsymmetric* problem. This contrast is detailed below. Here I want to emphasize the differences in conditioning. In some sense, the eigenvalues of a symmetric matrix are always well-conditioned functions of the matrix – a rare example of uniform good fortune. By contrast, the eigenvalues of an unsymmetric matrix may be so ill-conditioned, even for n as small as 20, that they are not computable (in double precision arithmetic) and essentially useless. Eigenvalues of unsymmetric matrices are too useful to ignore, but we can get into trouble if we ignore their potential ill-conditioning. Eigenvectors, even for symmetric matrices, may be ill-conditioned, but the consequences of this ill-conditioning seem less severe.

We begin with the unsymmetric eigenvalue problem. Nothing we say is wrong if A happens to be symmetric, but some of the statements might be misleading. An $n \times n$ matrix may have as many as n eigenvalues, denoted λ_k, $k = 1, \ldots, n$. If all the eigenvalues are distinct, the corresponding eigenvectors, denoted r_k, with $Ar_k = \lambda_k r_k$ must be linearly independent, and therefore form a basis for R^n. These n linearly independent vectors can be assembled to form the columns of an $n \times n$ eigenvector matrix that we call the *right eigenvector* matrix.

$$R = \begin{pmatrix} \vdots & & & & \vdots \\ r_1 & \cdot & \cdot & \cdot & r_n \\ \vdots & & & & \vdots \end{pmatrix} . \tag{4.9}$$

We also consider the diagonal eigenvalue matrix with the eigenvalues on the diagonal and zeros in all other entries:

$$\Lambda = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix} .$$

The eigenvalue – eigenvector relations may be expressed in terms of these matrices as

$$AR = R\Lambda \; . \tag{4.10}$$

To see this, check that multiplying R by A is the same as multiplying each of the columns of R by A. Since these are eigenvectors, we get

$$\begin{aligned} A \begin{pmatrix} \vdots & & & & \vdots \\ r_1 & \cdot & \cdot & \cdot & r_n \\ \vdots & & & & \vdots \end{pmatrix} &= \begin{pmatrix} \vdots & & & & \vdots \\ \lambda_1 r_1 & \cdot & \cdot & \cdot & \lambda_n r_n \\ \vdots & & & & \vdots \end{pmatrix} \\ &= \begin{pmatrix} \vdots & & & & \vdots \\ r_1 & \cdot & \cdot & \cdot & r_n \\ \vdots & & & & \vdots \end{pmatrix} \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix} \\ &= R\Lambda \; . \end{aligned}$$

Since the columns of R are linearly independent, R is invertible, we can multiply (4.10) from the right and from the left by R^{-1} to get

$$R^{-1}ARR^{-1} = R^{-1}R\Lambda R^{-1} \; ,$$

then cancel the $R^{-1}R$ and RR^{-1}, and define[8] $L = R^{-1}$ to get

$$LA = \Lambda L \; .$$

This shows that the k^{th} row of L is an eigenvector of A if we put the A on the right:

$$l_k A = \lambda_k l_k \; .$$

Of course, the λ_k are the same: there is no difference between "right" and "left" eigenvalues. The matrix equation we used to define L, $LR = I$, gives useful relations between left and right eigenvectors. The (j,k) entry of LR is the product of row j of L with row k of R. When $j = k$ this product should be a diagonal entry of I, namely one. When $j \neq k$, the product should be zero. That is

$$\left. \begin{aligned} l_k r_k &= 1 \\ l_j r_k &= 0 \qquad \text{if } j \neq k. \end{aligned} \right\} \tag{4.11}$$

[8]Here L refers to a matrix, not a general linear transformation.

These are called *biorthogonality* relations. For example, r_1 need not be orthogonal to r_2, but it is orthogonal to l_2. The set of vectors r_k is not orthogonal, but the two sets l_j and r_k are biorthogonal. The left eigenvectors are sometimes called *adjoint eigenvectors* because their transposes form right eigenvectors for the adjoint of A:

$$A^* l_j^* = \lambda_j l_j^* \; .$$

Still supposing n distinct eigenvalues, we may take the right eigenvectors to be a basis for R^n (or C^n if the entries are not real). As discussed in Section 2.2, we may express the action of A in this basis. Since $Ar_j = \lambda_j r_j$, the matrix representing the linear transformation A in this basis will be the diagonal matrix Λ. In the framework of Section 2.2, this is because if we expand a vector $v \in R^n$ in the r_k basis, $v = v_1 r_1 + \cdots + v_n r_n$, then $Av = \lambda_1 v_1 r_1 + \cdots + \lambda_n v_n r_n$. For this reason finding a complete set of eigenvectors and eigenvalues is called *diagonalizing* A. A matrix with n linearly independent right eigenvectors is *diagonalizable.*

If A does not have n distinct eigenvalues then there may be no basis in which the action of A is diagonal. For example, consider the matrix

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

Clearly, $J \neq 0$ but $J^2 = 0$. A diagonal or diagonalizable matrix cannot have this property: if $\Lambda^2 = 0$ then $\Lambda = 0$, and if the relations $A \neq 0$, $A^2 = 0$ in one basis, they hold in any other basis. In general a *Jordan block* with eigenvalue λ is a $k \times k$ matrix with λ on the diagonal, 1 on the *superdiagonal* and zeros elsewhere:

$$\begin{pmatrix} \lambda & 1 & 0 & \cdots & 0 \\ 0 & \lambda & 1 & 0 & 0 \\ \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \lambda & 1 \\ 0 & 0 & \cdots & 0 & \lambda \end{pmatrix} \; .$$

If a matrix has fewer than n distinct eigenvalues, it might or might not be diagonalizable. If A is not diagonalizable, a theorem of linear algebra states that there is a basis in which A has *Jordan form.* A matrix has Jordan form if it is *block diagonal* with Jordan blocks of various sizes and eigenvalues on the diagonal. It can be difficult to compute the Jordan form of a matrix numerically, as we will see.

The eigenvalue – eigenvector problem for symmetric or self-adjoint matrices is different and in many ways simpler than the general nonsymmetric eigenvalue problem. The eigenvalues are real. The left eigenvectors are transposes of the right eigenvectors: $l_k = r_k^*$. There are no Jordan blocks; every symmetric matrix is diagonalizable even if the number of distinct eigenvalues is less than n. The biorthogonality relations (4.11) become the normalization conditions

$$r_j^* r_j = 1 \; , \qquad (4.12)$$

and the orthogonality relations

$$r_j^* r_k = 0 \tag{4.13}$$

A set of vectors satisfying both (4.12) and (4.13) is called *orthonormal*. A complete set of eigenvectors of a symmetric matrix forms an orthonormal basis. It is easy to check that the orthonormality relations are equivalent to the matrix R in (4.9) satisfying $R^*R = I$, or, equivalently, $R^{-1} = R^*$. Such a matrix is called *orthogonal*. The matrix form of the eigenvalue relation (4.10) may be written $R^*AR = \Lambda$, or $A = R\Lambda R^*$, or $R^*A = \Lambda R^*$. The latter shows (yet again) that the rows of R^*, which are r_k^*, are left eigenvectors of A.

4.2.6 Differentiation and perturbation theory

The main technique in the perturbation theory of Section 4.3 is implicit differentiation. We use the formalism of *virtual perturbations* from mechanical engineering, which is related to tangent vectors in differential geometry. It may seem roundabout at first, but it makes actual calculations quick.

Suppose $f(x)$ represents m functions, $f_1(x), \ldots, f_m(x)$ of n variables, $x_1, \ldots, x_n$. The Jacobian matrix[9], $f'(x)$, is the $m \times n$ matrix of partial derivatives $f'_{jk}(x) = \partial_{x_k} f_j(x)$. If f is differentiable (and f' is Lipschitz continuous), then the first derivative approximation is (writing x_0 for x to clarify some discussion below)

$$f(x_0 + \Delta x) - f(x_0) = \Delta f = f'(x_0)\Delta x + O\left(\|\Delta x\|^2\right) . \tag{4.14}$$

Here Δf and Δx are column vectors.

Suppose s is a scalar "parameter" and $x(s)$ is a differentiable curve in R^n with $x(0) = x_0$. The function $f(x)$ then defines a curve in R^m with $f(x(0)) = f(x_0)$. We define two vectors, called *virtual perturbations*,

$$\dot{x} = \frac{dx(s)}{ds}(0) \quad , \quad \dot{f} = \frac{df(x(s))}{ds}(0) .$$

The multivariate calculus chain rule implies the virtual perturbation formula

$$\dot{f} = f'(x_0)\dot{x} . \tag{4.15}$$

This formula is nearly the same as (4.14). The virtual perturbation strategy is to calculate the linear relationship (4.15) between virtual perturbations and use it to find the approximate relation (4.14) between actual perturbations. For this, it is important that any $\dot{x} \in R^n$ can be the virtual perturbation corresponding to some curve: just take the straight "curve" $x(s) = x_0 + s\dot{x}$.

The Leibnitz rule (product rule) for matrix multiplication makes virtual perturbations handy in linear algebra. Suppose $A(s)$ and $B(s)$ are differentiable

[9]See any good book on multivariate calculus.

curves of matrices and compatible for matrix multiplication. Then the virtual perturbation of AB is given by the product rule

$$\left.\frac{d}{ds}AB\right|_{s=0} = \dot{A}B + A\dot{B} \; . \tag{4.16}$$

To see this, note that the (jk) entry of $A(s)B(s)$ is $\sum_l a_{jl}(s)b_{lk}(s)$. Differentiating this using the ordinary product rule then setting $s = 0$ yields

$$\sum_l \dot{a}_{jl}b_{lk} + \sum_l a_{jl}\dot{b}_{lk} \; .$$

These terms correspond to the terms on the right side of (4.16). We can differentiate products of more than two matrices in this way.

As an example, consider perturbations of the inverse of a matrix, $B = A^{-1}$. The variable x in (4.14) now is the matrix A, and $f(A) = A^{-1}$. Apply implicit differentiation to the formula $AB = I$, use the fact that I is constant, and we get $\dot{A}B + A\dot{B} = 0$. Then solve for $\dot{B}$ (and use $A^{-1} = B$, and get

$$\dot{B} = -A^{-1}\dot{A}A^{-1} \; .$$

The corresponding actual perturbation formula is

$$\Delta\left(A^{-1}\right) = -A^{-1}\,\Delta A\;A^{-1} + O\left(\|\Delta A\|^2\right) \; . \tag{4.17}$$

This is a generalization of the fact that the derivative of $1/x$ is $-1/x^2$, so $\Delta(1/x) \approx -1/x^2\Delta x$. When x is replaced by A and ΔA does not commute with A, we have to worry about the order of the factors. The correct order is (4.17).

For future reference we comment on the case $m = 1$, which is the case of one function of n variables. The $1 \times n$ Jacobian matrix may be thought of as a row vector. We often write this as $\bigtriangledown f$, and calculate it from the fact that $\dot{f} = \bigtriangledown f(x) \cdot \dot{x}$ for all $\dot{x}$. In particular, x is a stationary point of f if $\bigtriangledown f(x) = 0$, which is the same as $\dot{f} = 0$ for all $\dot{x}$. For example, suppose $f(x) = x^*Ax$ for some $n \times n$ matrix A. This is a product of the $1 \times n$ matrix x^* with A with the $n \times 1$ matrix x. The Leibnitz rule (4.16) gives, if A is constant,

$$\dot{f} = \dot{x}^*Ax + x^*A\dot{x} \; .$$

Since the 1×1 real matrix $\dot{x}^*Ax$ is equal to its transpose, this is

$$\dot{f} = x^*(A + A^*)\dot{x} \; .$$

This implies that (both sides are row vectors)

$$\bigtriangledown\;(\;x^*Ax\;) = x^*(A + A^*) \; . \tag{4.18}$$

If $A^* = A$, we recognize this as a generalization of $n = 1$ formula $\frac{d}{dx}(ax^2) = 2ax$.

4.2.7 Variational principles for the symmetric eigenvalue problem

A *variational principle* is a way to find something by solving a maximization or minimization problem. The *Rayleigh quotient* for an $n \times n$ matrix is

$$Q(x) = \frac{x^* A x}{x^* x} = \frac{\langle x, Ax \rangle}{\langle x, x \rangle} \ . \tag{4.19}$$

If x is real, x^* is the transpose of x, which is a row vector. If x is complex, x^* is the adjoint. In either case, the denominator is $x^* x = \sum_{k=1}^n |x_k|^2 = \|x\|_{l^2}^2$. The Rayleigh quotient is defined for $x \neq 0$. A vector r is a *stationary point* if $\bigtriangledown Q(r) = 0$. If r is a stationary point, the corresponding value $\lambda = Q(r)$ is a *stationary value.*

Theorem 1 *If A is a real symmetric or a complex self-adjoint matrix, each eigenvector of A is a stationary point of the Rayleigh quotient and the corresponding eigenvalue is the corresponding stationary value. Conversely, each stationary point of the Rayleigh quotient is an eigenvector and the corresponding stationary value the corresponding eigenvalue.*

Proof: We give the proof for real x and real symmetric A. The complex self-adjoint case is an exercise. Underlying the theorem is the calculation (4.18) that if $A^* = A$ (this is where symmetry matters) then $\bigtriangledown \ (\ x^* A x \ ) = 2 x^* A$. With this we calculate (using the quotient rule and (4.18) with $A = I$)

$$\bigtriangledown Q = 2 \left(\frac{1}{x^* x} \right) x^* A - 2 \left(\frac{x^* A x}{(x^* x)^2} \right) x^* \ .$$

If x is a stationary point ($\bigtriangledown Q = 0$), then $x^* A = \left(\frac{x^* A x}{x^* x} \right) x^*$, or, taking the transpose,

$$Ax = \left(\frac{x^* A x}{x^* x} \right) x \ .$$

This shows that x is an eigenvector with

$$\lambda = \frac{x^* A x}{x^* x} = Q(x)$$

as the corresponding eigenvalue. Conversely if $Ar = \lambda r$, then $Q(r) = \lambda$ and the calculations above show that $\bigtriangledown Q(r) = 0$. This proves the theorem.

A simple observation shows that there is at least one stationary point of Q for Theorem 1 to apply to. If α is a real number, then $Q(\alpha x) = Q(x)$. We may choose α so that[10] $\|\alpha x\| = 1$. This shows that

$$\max_{x \neq 0} Q(x) = \max_{\|x\|=1} Q(x) = \max_{\|x\|=1} x^* A x \ .$$

[10]In this section and the next, $\|x\| = \|x\|_{l^2}$.

A theorem of analysis states that if $Q(x)$ is a continuous function on a compact set, then there is an r so that $Q(r) = \max_x Q(x)$ (the max is attained). The set of x with $\|x\| = 1$ (the *unit sphere*) is compact and Q is continuous. Clearly if $Q(r) = \max_x Q(x)$, then $\bigtriangledown Q(r) = 0$, so r is a stationary point of Q and an eigenvector of A.

The key to finding the other $n-1$ eigenvectors is a simple orthogonality relation. The principle also is the basis of the *singular value decomposition.* If r is an eigenvector of A and x is orthogonal to r, then x also is orthogonal to Ar. Since $Ar = \lambda r$, and x is orthogonal to r if $x^* r = 0$, this implies $x^* Ar = x^* \lambda r = 0$. More generally, suppose we have $m < n$ and eigenvectors $r_1, \ldots, r_m$. eigenvectors $r_1, \ldots, r_m$ with $m < n$. Let $V_m \subseteq R^n$ be the set of $x \in R^n$ with $x^* r_j = 0$ for $j = 1, \ldots, m$. It is easy to check that this is a subspace of R^n. The orthogonality principle is that if $x \in V_m$ then $Ax \in V_m$. That is, if $x^* r_j = 0$ for all j, then $(Ax)^* r_j = 0$ for all j. But $(Ax)^* r_j = x^* A^* r_j = x^* A r_j = x^* \lambda_j r_j = 0$ as before.

The variational and orthogonality principles allow us to find $n-1$ additional orthonormal eigenvectors as follows. Start with r_1, a vector that maximizes Q. Let V_1 be the vector space of all $x \in R^n$ with $r_1^* x = 0$. We just saw that V_1 is an *invariant subspace* for A, which means that $Ax \in V_1$ whenever $x \in V_1$. Thus A defines a linear transformation from V_1 to V_1, which we call A_1. Chapter 5.1 gives a proof that A_1 is symmetric in a suitable basis. Therefore, Theorem 1 implies that A_1 has at least one real eigenvector, r_2, with real eigenvalue λ_2. Since $r_2 \in V_1$, the action of A and A_1 on r_2 is the same, which means that $Ar_2 = \lambda_2 r_2$. Also since $r_2 \in V_1$, r_2 is orthogonal to r_1. Now let $V_2 \subset V_1$ be the set of $x \in V_1$ with $x^* r_2 = 0$. Since $x \in V_2$ means $x^* r_1 = 0$ and $x^* r_2 = 0$, V_2 is an invariant subspace. Thus, there is an $r_3 \in V_2$ with $Ar_3 = A_1 r_3 = A_2 r_3 = \lambda_3 r_3$. And again r_3 is orthogonal to r_2 because $r_3 \in V_2$ and r_3 is orthogonal to r_1 because $r_3 \in V_2 \subset V_1$. Continuing in this way, we eventually find a full set of n orthogonal eigenvectors.

4.2.8 Least squares

Suppose A is an $m \times n$ matrix representing a linear transformation from R^n to R^m, and we have a vector $b \in R^m$. If $n < m$ the linear equation system $Ax = b$ is *overdetermined* in the sense that there are more equations than variables to satisfy them. If there is no x with $Ax = b$, we may seek an x that minimizes the *residual*

$$r = Ax - b \; . \tag{4.20}$$

This *linear least squares* problem

$$\min_x \|Ax - b\|_{l^2} \;\; , \tag{4.21}$$

is the same as finding x to minimize the sum of squares

$$SS = \sum_{j=1}^{n} r_j^2 \; .$$

Linear least squares problems arise through *linear regression* in statistics. A *linear regression model* models the response, b, as a linear combination of m *explanatory* vectors, a_k, each weighted by a *regression coefficient*, x_k. The residual, $R = (\sum_{k=1}^m a_k x_k) - b$, is modeled as a Gaussian random variable[11] with mean zero and variance σ^2. We do n *experiments.* The explanatory variables and response for experiment j are a_{jk}, for $k = 1. \ldots, m$, and b_j, and the residual (for given regression coefficients) is $r_j = \sum_{k=1}^m a_{jk} x_k - b_j$. The *log likelihood* function is (r depends on x through (4.20) $f(x) = -\sigma^2 \sum_{j=1}^n r_j^2$. Finding regression coefficients to maximize the log likelihood function leads to (4.21).

The *normal equations* give one approach to least squares problems. We calculate:

$$\begin{aligned}
\|r\|_{l^2}^2 &= r^* r \\
&= (Ax - b)^* (Ax - b) \\
&= x^* A^* A x - 2x^* A^* b + b^* b \ .
\end{aligned}$$

Setting the gradient to zero as in the proof of Theorem 1 leads to the normal equations

$$A^* A x = A^* b \ , \tag{4.22}$$

which can be solved by

$$x = (A^* A)^{-1} A^* b \ . \tag{4.23}$$

The matrix $M = A^* A$ is the *moment matrix* or the *Gram matrix.* It is symmetric, and *positive definite* if A has rank m, so the *Choleski decomposition* of M (see Chapter 5.1) is a good way to solve (4.22). The matrix $(A^* A)^{-1} A^*$ is the *pseudoinverse* of A. If A were square and invertible, it would be A^{-1} (check this). The normal equation approach is the fastest way to solve dense linear least squares problems, but it is not suitable for the subtle ill-conditioned problems that arise often in practice.

The singular value decomposition in Section 4.2.9 and the QR decomposition from Section 5.4 give better ways to solve ill-conditioned linear least squares problems.

4.2.9 Singular values and principal components

Eigenvalues and eigenvectors of a symmetric matrix have at many applications. They can be used to solve dynamical problems involving A. Because the eigenvectors are orthogonal, they also determine the l^2 norm and condition number of A. Eigenvalues and eigenvectors of a non-symmetric matrix are not orthogonal so the eigenvalues do not determine the norm or condition number. *Singular values* for non-symmetric or even non-square matrices are a partial substitute.

Let A be an $m \times n$ matrix that represents a linear transformation from R^n to R^m. The *right singular vectors*, $v_k \in R^n$ form an orthonormal basis for

[11]See any good book on statistics for definitions of Gaussian random variable and the log likelihood function. What is important here is that a systematic statistical procedure, the *maximum likelihood* method, tells us to minimize the sum of squares of residuals.

R^n. The *left singular vectors*, $u_k \in R^m$, form an orthonormal basis for R^m. Corresponding to each v_k and u_k pair is a non-negative *singular value*, σ_k with

$$Av_k = \sigma_k u_k \ . \tag{4.24}$$

By convention these are ordered so that $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$. If $n < m$ we interpret (4.24) as saying that $\sigma_k = 0$ for $k > n$. If $n > m$ we say $\sigma_k = 0$ for $k > m$.

The non-zero singular values and corresponding singular vectors may be found one by one using variational and orthogonality principles similar to those in Section 4.2.7. We suppose A is not the zero matrix (not all entries equal to zero). The first step is the variational principle:

$$\sigma_1 = \max_{x \neq 0} \frac{\|Ax\|}{\|x\|} \ . \tag{4.25}$$

As in Section 4.2.7, the maximum is achieved, and $\sigma_1 > 0$. Let $v_1 \in R^n$ be a maximizer, normalized to have $\|v_1\| = 1$. Because $\|Av_1\| = \sigma_1$, we may write $Av_1 = \sigma_1 u_1$ with $\|u_1\| = 1$. This is the relation (4.24) with $k = 1$.

The optimality condition calculated as in the proof of Theorem 1 implies that

$$u_1^* A = \sigma_1 v_1^* \ . \tag{4.26}$$

Indeed, since $\sigma_1 > 0$, (4.25) is equivalent to[12]

$$\begin{aligned} \sigma_1^2 &= \max_{x \neq 0} \frac{\|Ax\|^2}{\|x\|^2} \\ &= \max_{x \neq 0} \frac{(Ax)^*(Ax)}{x^* x} \\ \sigma_1^2 &= \max_{x \neq 0} \frac{x^*(A^*A)x}{x^* x} \ . \end{aligned} \tag{4.27}$$

Theorem 1 implies that the solution to the maximization problem (4.27), which is v_1, satisfies $\sigma^2 v_1 = A^* A v_1$. Since $Av_1 = \sigma u_1$, this implies $\sigma_1 v_1 = A^* u_1$, which is the same as (4.26).

The analogue of the eigenvalue orthogonality principle is that if $x^* v_1 = 0$, then $(Ax)^* u_1 = 0$. This is true because

$$(Ax)^* u_1 = x^* \left(A^* u_1\right) = x^* \sigma_1 v_1 = 0 \ .$$

Therefore, if we define $V_1 \subset R^n$ by $x \in V_1$ if $x^* v_1 = 0$, and $U_1 \subset R^m$ by $y \in U_1$ if $y^* u_1 = 0$, then A also defines a linear transformation (called A_1) from V_1 to U_1. If A_1 is not identically zero, we can define

$$\sigma_2 = \max_{\substack{x \in V_1 \\ x \neq 0}} \frac{\|Ax\|^2}{\|x\|^2} = \max_{\substack{x^* v_1 = 0 \\ x \neq 0}} \frac{\|Ax\|^2}{\|x\|^2} \ ,$$

[12]These calculations make constant use of the associativity of matrix multiplication, even when one of the matrices is a row or column vector.

and get $Av_2 = \sigma_2 u_2$ with $v_2^* v_1 = 0$ and $u_2^* u_1 = 0$. This is the second step constructing orthonormal bases satisfying (4.24). Continuing in this way, we can continue finding orthonormal vectors v_k and u_k that satisfy (4.24) until reach $A_k = 0$ or $k = m$ or $k = n$. After that point, the may complete the v and u bases arbitrarily as in Chapter 5.1 with remaining singular values being zero.

The *singular value decomposition* (*SVD*) is a matrix expression of the relations (4.24). Let U be the $m \times m$ matrix whose columns are the left singular vectors u_k (as in (4.9)). The orthonormality relations $u_j^* u_k = \delta_{jk}$ are equivalent to U being an orthogonal matrix: $U^*U = I$. Similarly, we can form the orthogonal $n \times n$ matrix, V, whose columns are the right singular vectors v_k. Finally, the $m \times n$ matrix, Σ, has the singular values on its diagonal (with somewhat unfortunate notation), $\sigma_{jj} = \sigma_j$, and zeros off the diagonal, $\sigma_{jk} = 0$ if $j \neq k$. With these definitions, the relations (4.24) are equivalent to $AV = U\Sigma$, which more often is written

$$A = U\Sigma V^* \ . \tag{4.28}$$

This the singular value decomposition. Any matrix may be factored, or decomposed, into the product of the form (4.28) where U is an $m \times m$ orthogonal matrix, Σ is an $m \times n$ diagonal matrix with nonnegative entries, and V is an $n \times n$ orthogonal matrix.

A calculation shows that $A^*A = V\Sigma^*\Sigma V^* = V\Lambda V^*$. This implies that the eigenvalues of A^*A are given by $\lambda_j = \sigma_j^2$ and the right singular vectors of A are the eigenvectors of A^*A. It also implies that $\kappa_{l^2}(A^*A) = \kappa_{l^2}(A)^2$. This means that the condition number of solving the normal equations (4.22) is the square of the condition number of the original least squares problem (4.21). If the condition number of a least squares problem is $\kappa_{l^2}(A) = 10^5$, even the best solution algorithm can amplify errors by a factor of 10^5. Solving using the normal equations can amplify rounding errors by a factor of 10^{10}.

Many people call singular vectors u_k and v_k *principal components*. They refer to the singular value decomposition as *principal component analysis*, or *PCA*. One application is *clustering*, in which you have n objects, each with m measurements, and you want to separate them into two clusters, say "girls" and "boys". You assemble the data into a matrix, A, and compute, say, the largest two singular values and corresponding left singular vectors, $u_1 \in R^m$ and $u_2 \in R^m$. The data for object k is $a_k \in R^m$, and you compute $z_k \in R^2$ by $z_{k1} = u_1^* a_k$ and $z_{k2} = u_2^* a_k$, the components of a_k in the principal component directions. You then plot the n points z_k in the plane and look for clusters, or maybe just a line that separates one group of points from another. Surprising as may seem, this simple procedure does identify clusters in practical problems.

4.3 Condition number

Ill-conditioning can be a serious problem in numerical solution of problems in linear algebra. We take into account possible ill-conditioning when we choose computational strategies. For example, the matrix exponential $\exp(A)$ (see exercise 12) can be computed using the eigenvectors and eigenvalues of A. We will see in Section 4.3.3 that the eigenvalue problem may be ill conditioned even when the problem of computing $\exp(A)$ is fine. In such cases we need a way to compute $\exp(A)$ that does not use the eigenvectors and eigenvalues of A.

As we said in Section 2.7 (in slightly different notation), the condition number is the ratio of the change in the answer to the change in the problem data, with (*i*) both changed measured in relative terms, and (*ii*) the change in the problem data being small. Norms allow us to make this definition more precise in the case of multivariate functions and data. Let $f(x)$ represent m functions of n variables, with Δx being a change in x and $\Delta f = f(x + \Delta x) - f(x)$ the corresponding change in f. The size of Δx, relative to x is $\|\Delta x\| / \|x\|$, and similarly for Δf. In the multivariate case, the size of Δf depends not only on the size of Δx, but also on the direction. The norm based condition number seeks the worst case Δx, which leads to

$$\kappa(x) = \lim_{\epsilon \to 0} \max_{\|\Delta x\| = \epsilon} \frac{\frac{\|f(x+\Delta x) - f(x)\|}{\|f(x)\|}}{\frac{\|\Delta x\|}{\|x\|}} \quad . \tag{4.29}$$

Except for the maximization over directions Δx with $\|\Delta x\| = \epsilon$, this is the same as the earlier definition 2.8.

Still following Section 2.7, we express (4.29) in terms of derivatives of f. We let $f'(x)$ represent the $m \times n$ *Jacobian* matrix of first partial derivatives of f, as in Section 4.2.6, so that, $\Delta f = f'(x)\Delta x + O\left(\|\Delta x\|^2\right)$. Since $O\left(\|\Delta x\|^2\right) / \|\Delta x\| = O\left(\|\Delta x\|\right)$, the ratio in (4.29) may be written

$$\frac{\|\Delta f\|}{\|\Delta x\|} \cdot \frac{\|x\|}{\|f\|} = \frac{\|f'(x)\Delta x\|}{\|\Delta x\|} \cdot \frac{\|x\|}{\|f\|} + O\left(\|\Delta x\|\right) \quad .$$

The second term on the right disappears as $\Delta x \to 0$. Maximizing the first term on the right over Δx yields the norm of the matrix $f'(x)$. Altogether, we have

$$\kappa(x) = \|f'(x)\| \cdot \frac{\|x\|}{\|f(x)\|} \quad . \tag{4.30}$$

This differs from the earlier condition number definition (2.10) in that it uses norms and maximizes over Δx with $\|\Delta x\| = \epsilon$.

In specific calculations we often use a slightly more complicated way of stating the definition (4.29). Suppose that P and Q are two positive quantities and there is a C so that $P \le C \cdot Q$. We say that C is the *sharp* constant if there is no C' with $C' < C$ so that $P \le C' \cdot Q$. For example, we have the inequality $\sin(2\epsilon) \le 3 \cdot \epsilon$ for all x. But $C = 3$ is not the sharp constant because the inequality also is true with $C' = 2$, which is sharp. But this definition is clumsy, for example, if we ask for the sharp constant in the inequality $\tan(\epsilon) \le C \cdot \epsilon$. For one thing, we must restrict to $\epsilon \to 0$. The sharp constant should be $C = 1$ because if $C' > 1$, there is an ϵ_0 so that if $\epsilon \le \epsilon_0$, then $\tan(\epsilon) \le C' \cdot \epsilon$. But the inequality is not true with $C = 1$ for any ϵ. Therefore we write

$$P(\epsilon) \underset{\sim}{\le} CQ(\epsilon) \quad \text{as } \epsilon \to 0 \tag{4.31}$$

if

$$\lim_{\epsilon \to 0} \frac{P(\epsilon)}{Q(\epsilon)} \leq C \; .$$

In particular, we can seek the sharp constant, the smallest C so that

$$P(\epsilon) \leq C \cdot Q(\epsilon) + O(\epsilon) \quad \text{as } \epsilon \to 0.$$

The definition (**??**) is precisely that $\kappa(x)$ is the sharp constant in the inequality

$$\frac{\|\Delta f\|}{\|f\|} \lesssim \frac{\|\Delta x\|}{\|x\|} \quad \text{as } \|x\| \to 0. \tag{4.32}$$

4.3.1 Linear systems, direct estimates

We start with the condition number of calculating $b = Au$ in terms of u with A fixed. This fits into the general framework above, with u playing the role of x, and Au of $f(x)$. Of course, A is the Jacobian of the function $u \to Au$, so (4.30) gives

$$\kappa(A, u) = \|A\| \cdot \frac{\|u\|}{\|Au\|} \; . \tag{4.33}$$

The condition number of solving a linear system $Au = b$ (finding u as a function of b) is the same as the condition number of the computation $u = A^{-1}b$. The formula (4.33) gives this as

$$\kappa(A^{-1}, b) = \left\|A^{-1}\right\| \cdot \frac{\|b\|}{\|A^{-1}b\|} = \|A{-}1\| \cdot \frac{\|Au\|}{\|u\|} \; .$$

For future reference, not that this is not the same as (4.33).

The traditional definition of the condition number of the Au computation takes the worst case relative error not only over perturbations Δu but also over vectors u. Taking the maximum over Δu led to (4.33), so we need only maximize it over u:

$$\kappa(A) = \|A\| \cdot \max_{u \neq 0} \frac{\|u\|}{\|Au\|} \; . \tag{4.34}$$

Since $A(u + \Delta u) - Au = A\Delta u$, and u and Δu are independent variables, this is the same as

$$\kappa(A) = \max_{u \neq 0} \frac{\|u\|}{\|Au\|} \cdot \max_{\Delta u \neq 0} \frac{\|A\Delta u\|}{\|\Delta u\|} \; . \tag{4.35}$$

To evaluate the maximum, we suppose A^{-1} exists.[13] Substituting $Au = v$, $u = A^{-1}v$, gives

$$\max_{u \neq 0} \frac{\|u\|}{\|Au\|} = \max_{v \neq 0} \frac{\left\|A^{-1}v\right\|}{\|v\|} = \left\|A^{-1}\right\| \; .$$

[13]See exercise 8 for a the l^2 condition number of the $u \to Au$ problem with singular or non-square A.

Thus, (4.34) leads to

$$\kappa(A) = \|A\| \left\|A^{-1}\right\| \tag{4.36}$$

as the worst case condition number of the forward problem.

The computation $b = Au$ with

$$A = \begin{pmatrix} 1000 & 0 \\ 0 & 10 \end{pmatrix}$$

illustrates this discussion. The error amplification relates $\|\Delta b\| \,/\, \|b\|$ to $\|\Delta u\| \,/\, \|u\|$. The worst case would make $\|b\|$ small relative to $\|u\|$ and $\|\Delta b\|$ large relative to $\|\Delta u\|$: amplify u the least and Δu the most. This is achieved by taking $u = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ so that $Au = \begin{pmatrix} 0 \\ 10 \end{pmatrix}$ with amplification factor 10, and $\Delta u = \begin{pmatrix} \epsilon \\ 0 \end{pmatrix}$ with $A\Delta u = \begin{pmatrix} 1000\epsilon \\ 0 \end{pmatrix}$ and amplification factor 1000. This makes $\|\Delta b\| \,/\, \|b\|$ 100 times larger than $\|\Delta u\| \,/\, \|u\|$. For the condition number of calculating $u = A^{-1}b$, the worst case is $b = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ and $\Delta b = \begin{pmatrix} \epsilon \\ 0 \end{pmatrix}$, which amplifies the error by the same factor of $\kappa(A) = 100$.

The informal condition number (4.34) has advantages and disadvantages over the more formal one (4.33). At the time we design a computational strategy, it may be easier to estimate the informal condition number than the formal one, as we may know more about A than u. If we have no idea what u will come up, we have a reasonable chance of getting something like the worst one. Moreover, $\kappa(A)$ defined by (4.34) determines the convergence rate of iterative strategies for solving linear systems involving A. On the other hand, there are cases, particularly when solving partial differential equations, where $\kappa(A)$ is much more pessimistic than $\kappa(A, u)$. For example, in Exercise 11, $\kappa(A)$ is on the order of n^2, where n is the number of unknowns. The truncation error for the second order discretization is on the order of $1/n^2$. A naive estimate using (4.34) might suggest that solving the system amplifies the $O(n^{-2})$ truncation error by a factor of n^2 to be on the same order as the solution itself. This does not happen because the u we seek is smooth, and not like the worst case.

The informal condition number (4.36) also has the strange feature than $\kappa(A) = \kappa(A^{-1})$, since $\left(A^{-1}\right)^{-1} = A$. For one thing, this is not true, even using informal definitions, for nonlinear problems. Moreover, it is untrue in important linear problems, such as the heat equation.[14] Computing the solution at time $t > 0$ from "initial data" at time zero is a linear process that is well conditioned and numerically stable. Computing the solution at time zero from the solution at time $t > 0$ is so ill conditioned as to be essentially impossible. Again, the more precise definition (4.33) does not have this drawback.

[14] See, for example, the book by Fritz John on partial differential equations.

4.3.2 Linear systems, perturbation theory

If $Au = b$, we can study the dependence of u on A through perturbation theory. The starting point is the perturbation formula (4.17). Taking norms gives

$$\|\Delta u\| \lessapprox \left\|A^{-1}\right\| \|\Delta A\| \|u\| \ , \qquad \text{(for small } \Delta A\text{)}, \tag{4.37}$$

so

$$\frac{\|\Delta u\|}{\|u\|} \lessapprox \left\|A^{-1}\right\| \|A\| \cdot \frac{\|\Delta A\|}{\|A\|} \tag{4.38}$$

This shows that the condition number satisfies $\kappa \leq \left\|A^{-1}\right\| \|A\|$. The condition number is equal to $\left\|A^{-1}\right\| \|A\|$ if the inequality (4.37) is (approximately for small ΔA) sharp, which it is because we can take $\Delta A = \epsilon I$ and u to be a maximum stretch vector for A^{-1}.

We summarize with a few remarks. First, the condition number formula (4.36) applies to the problem of solving the linear system $Au = b$ both when we consider perturbations in b and in A, though the derivations here are different. However, this is not the condition number in the strict sense of (4.29) and (4.30) because the formula (4.36) assumes taking the worst case over a family of problems (varying b in this case), not just over all possible small perturbations in the data. Second, the formula (4.36) is independent of the size of A. Replacing A with cA leaves $\kappa(A)$ unchanged. What matters is the ratio of the maximum to minimum stretch, as in (4.35).

4.3.3 Eigenvalues and eigenvectors

The eigenvalue relationship is $Ar_j = \lambda_j r_j$. Perturbation theory allows to estimate the changes in λ_j and r_j that result from a small ΔA. These perturbation results are used throughout science and engineering. We begin with the symmetric or self-adjoint case, it often is called *Rayleigh Schödinger* perturbation theory[15] Using the virtual perturbation method of Section 4.2.6, differentiating the eigenvalue relation using the product rule yields

$$\dot{A}r_j + A\dot{r}_j = \dot{\lambda}_j r_j + \lambda_j \dot{r}_j \ . \tag{4.39}$$

Multiply this from the left by r_j^* and use the fact that r_j^* is a left eigenvector[16] gives

$$r_j^* \dot{A}_j r_j = \dot{\lambda}_l r_j^* r_j \ .$$

If r_j is normalized so that $r_j^* r_j = \|r_j\|_{l^2}^2 = 1$, then the right side is just $\dot{\lambda}_j$. Trading virtual perturbations for actual small perturbations turns this into the famous formula

$$\Delta\lambda_j \ = \ r_j^* \, \Delta A \, r_j \ + \ O\left(\|\Delta A\|^2\right) \ . \tag{4.40}$$

[15]Lord Rayleigh used it to study vibrational frequencies of plates and shells. Later Erwin Schrödinger used it to compute energies (which are eigenvalues) in quantum mechanics.

[16]$Ar_j = \lambda_j r_j \Rightarrow (Ar_j)^* = (\lambda_j r_j)^* \Rightarrow r_j^* A = \lambda_j r_j^*$ since $A^* = A$ and λ_j is real.

We get a condition number estimate by recasting this in terms of relative errors on both sides. The important observation is that $\|r_j\|_{l^2} = 1$, so $\|\Delta A \cdot r_j\|_{l^2} \leq \|\Delta A\|_{l^2}$ and finally

$$\left| r_j^* \; \Delta A \; r_j \right| \overset{<}{\approx} \|\Delta A\|_{l^2} \; .$$

This inequality is sharp because we can take $\Delta A = \epsilon r_j r_j^*$, which is an $n \times n$ matrix with (see exercise 7) $\left\|\epsilon r_j r_j^*\right\|_{l^2} = |\epsilon|$. Putting this into (4.40) gives the also sharp inequality,

$$\left| \frac{\Delta \lambda_j}{\lambda_j} \right| \leq \frac{\|A\|_{l^2}}{|\lambda_j|} \frac{\|\Delta A\|_{l^2}}{\|A\|_{l^2}} \; .$$

We can put this directly into the abstract condition number definition (4.29) to get the conditioning of λ_j:

$$\kappa_j(A) = \frac{\|A\|_{l^2}}{|\lambda_j|} = \frac{|\lambda|_{max}}{|\lambda_j|} \tag{4.41}$$

Here, $\kappa_j(A)$ denotes the condition number of the problem of computing λ_j, which is a function of the matrix A, and $\|A\|_{l^2} = |\lambda|_{max}$ refers to the eigenvalue of largest absolute value.

The condition number formula (4.41) predicts the sizes of errors we get in practice. Presumably λ_j depends in some way on all the entries of A and the perturbations due to roundoff will be on the order of the entries themselves, multiplied by the machine precision, ϵ_{mach}, which are on the order of $\|A\|$. Only if λ_j is very close to zero, by comparison with $|\lambda_{max}|$, will it be hard to compute with high relative accuracy. All of the other eigenvalue and eigenvector problems have much worse condition number difficulties.

The eigenvalue problem for non-symmetric matrices can by much more sensitive. To derive the analogue of (4.40) for non-symmetric matrices we start with (4.39) and multiply from the left with the corresponding left eigenvector, l_j. After simplifying, the result is

$$\dot{\lambda}_j = l_j \dot{A} r_j \; , \quad \Delta\lambda_j = l_j \Delta A r_j + O\left(\|\Delta A\|^2\right) \; . \tag{4.42}$$

In the non-symmetric case, the eigenvectors need not be orthogonal and the eigenvector matrix R need not be well conditioned. For this reason, it is possible that l_k, which is a row of R^{-1} is very large. Working from (4.42) as we did for the symmetric case leads to

$$\left| \frac{\Delta \lambda_j}{\lambda_j} \right| \leq \kappa_{LS}(R) \frac{\|A\|}{|\lambda_j|} \frac{\|\Delta A\|}{\|A\|} \; .$$

Here $\kappa_{LS}(R) = \left\|R^{-1}\right\| \|R\|$ is the linear systems condition number of the right eigenvector matrix. Therefore, the condition number of the non-symmetric eigenvalue problem is (again because the inequalities are sharp)

$$\kappa_j(A) = \kappa_{LS}(R) \frac{\|A\|}{|\lambda_j|} \; . \tag{4.43}$$

Since A is not symmetric, we cannot replace $\|A\|$ by $|\lambda|_{max}$ as we did for (4.41). In the symmetric case, the only reason for ill-conditioning is that we are looking for a (relatively) tiny number. For non-symmetric matrices, it is also possible that the eigenvector matrix is ill-conditioned. It is possible to show that if a family of matrices approaches a matrix with a Jordan block, the condition number of R approaches infinity. For a symmetric or self-adjoint matrix, R is orthogonal or unitary, so that $\|R\|_{l^2} = \|R^*\|_{l^2} = 1$ and $\kappa_{LS}(R) = 1$.

The eigenvector perturbation theory uses the same ideas, with the extra trick of expanding the derivatives of the eigenvectors in terms of the eigenvectors themselves. We expand the virtual perturbation $\dot{r}_j$ in terms of the eigenvectors r_k. Call the expansion coefficients m_{jk}, and we have

$$\dot{r}_j = \sum_{l=1}^{n} m_{jl} r_l \ .$$

For the symmetric eigenvalue problem, if all the eigenvalues are distinct, the formula follows from multiplying (4.39) from the left by r_k^*:

$$m_{jk} = \frac{r_k^* \dot{A} r_j}{\lambda_j - \lambda_k} \ ,$$

so that

$$\Delta r_j = \sum_{k \neq j} \frac{r_k^* \Delta A r_j}{\lambda_j - \lambda_k} + O\left(\|\Delta A\|^2\right) \ .$$

(The term $j = k$ is omitted because $m_{jj} = 0$: differentiating $r_j^* r_j = 1$ gives $r_j^* \dot{r}_j = 0$.) This shows that the eigenvectors have condition number "issues" even when the eigenvalues are well-conditioned, if the eigenvalues are close to each other. Since the eigenvectors are not uniquely determined when eigenvalues are equal, this seems plausible. The unsymmetric matrix perturbation formula is

$$m_{kj} = \frac{l_j \dot{A} r_k}{\lambda_j - \lambda_k} \ .$$

Again, we have the potential problem of an ill-conditioned eigenvector basis, combined with the possibility of close eigenvalues. The conclusion is that the eigenvector conditioning can be problematic, even though the eigenvalues are fine, for closely spaced eigenvalues.

4.4 Software

4.4.1 Software for numerical linear algebra

There have been major government-funded efforts to produce high quality software for numerical linear algebra. This culminated in the public domain software package *LAPack*. LAPack is a combination and extension of earlier packages *Eispack*, for solving eigenvalue problems, and *Linpack*, for solving systems of equations. Many of the high quality components of Eispack and Linpack also were incorporated into Matlab, which accounts for the high quality of Matlab linear algebra routines.

Our software advice is to use LAPack or Matlab software for computational linear algebra whenever possible. It is worth the effort in coax the LAPack `make` files to work in a particular computing environment. You may avoid wasting time coding algorithms that have been coded thousands of times before, or you may avoid suffering from the subtle bugs of the codes you got from next door that came from who knows where. The professional software often does it better, for example using *balancing* algorithms to improve condition numbers. The professional software also has clever condition number estimates that often are more sophisticated than the basic algorithms themselves.

4.4.2 Test condition numbers

Part of the error flag in linear algebra computation is the condition number. Accurate computation of the condition number may be more expensive than the problem at hand. For example, computing $\|A\| \left\|A^{-1}\right\|$ is more several times more expensive than solving $Ax = b$. However, there are cheap heuristics that generally are reliable, at least for identifying severe ill-conditioning. If the condition number is so large that all relative accuracy in the data is lost, the routine should return an error flag. Using such condition number estimates makes the code slightly slower, but it makes the computed results much more trustworthy.

4.5 Resources and further reading

If you need to review linear algebra, the Schaum's Outline review book on Linear Algebra may be useful. For a practical introduction to linear algebra written by a numerical analyst, try the book by Gilbert Strang. More theoretical treatments may be found in the book by Peter Lax or the one by Paul Halmos. An excellent discussion of condition number is given in the SIAM lecture notes of Lloyd N. Trefethen. Beresford Parlett has a nice but somewhat out-of-date book on the theory and computational methods for the symmetric eigenvalue problem. The Linear Algebra book by Peter Lax also has a beautiful discussion of eigenvalue perturbation theory and some of its applications. More applications may be found in the book *Theory of Sound* by Lord Rayleigh (reprinted by Dover Press) and in any book on quantum mechanics. I do not know an elementary book that covers perturbation theory for the non-symmetric eigenvalue problem in a simple way.

The software repository *Netlib*, `http://netlib.org`, is a source for LAPack. Other sources of linear algebra software, including Mathematica and *Numerical Recipes*, are not recommended. Netlib also has some of the best available software for solving sparse linear algebra problems.

4.6 Exercises

1. Let L be the differentiation operator that takes P_3 to P_2 described in Section 4.2.2. Let $f_k = H_k(x)$ for $k = 0, 1, 2, 3$ be the Hermite polynomial basis of P_3 and $g_k = H_k(x)$ for $k = 0, 1, 2$ be the Hermite basis of P_2. What is the matrix, A, that represents this L in these bases?

2. Suppose L is a linear transformation from V to V and that f_1, …, f_n, and g_1, …, g_n are two bases of V. Any $u \in V$ may be written in a unique way as $u = \sum_{k=1}^{n} v_k f_k$, or as $u = \sum_{k=1}^{n} w_k g_k$. There is an $n \times n$ matrix, R that relates the f_k expansion coefficients v_k to the g_k coefficients w_k by $v_j = \sum_{k=1}^{n} r_{jk} w_k$. If v and w are the column vectors with components v_k and w_k respectively, then $v = Rw$. Let A represent L in the f_k basis and B represent L in the g_k basis.

   (a) Show that $B = R^{-1}AR$.

   (b) For $V = P_3$, and $f_k = x^k$, and $g_k = H_k$, find R.

   (c) Let L be the linear transformation $Lp = q$ with $q(x) = \partial_x(xp(x))$. Find the matrix, A, that represents L in the monomial basis f_k.

   (d) Find the matrix, B, that represents L in the Hermite polynomial basis H_k.

   (e) Multiply the matrices to check explicitly that $B = R^{-1}AR$ in this case.

3. If A is an $n \times m$ matrix and B is an $m \times l$ matrix, then AB is an $n \times l$ matrix. Show that $(AB)^* = B^*A^*$. Note that the incorrect suggestion A^*B^* in general is not compatible for matrix multiplication.

4. Let $V = R^n$ and M be an $n \times n$ real matrix. This exercise shows that $\|u\| = (u^*Mu)^{1/2}$ is a vector norm whenever M is *positive definite* (defined below).

   (a) Show that $u^*Mu = u^*M^*u = u^*\left(\frac{1}{2}(M + M^*)\right)u$ for all $u \in V$. This means that as long as we consider functions of the form $f(u) = u^*Mu$, we may assume M is symmetric. For the rest of this question, assume M is symmetric. Hint: u^*Mu is a 1×1 matrix and therefore equal to its transpose.

   (b) Show that the function $\|u\| = (u^*Mu)^{1/2}$ is homogeneous: $\|xu\| = |x| \, \|u\|$.

   (c) We say M is positive definite if $u^*Mu > 0$ whenever $u \neq 0$. Show that if M is positive definite, then $\|u\| \geq 0$ for all u and $\|u\| = 0$ only for $u = 0$.

   (d) Show that if M is symmetric and positive definite (SPD), then $|u^*Mv| \leq \|u\| \, \|v\|$. This is the *Cauchy Schwartz inequality.* Hint (a famous old trick): $\phi(t) = (u + tv)^*M(u + tv)$ is a quadratic function of t that is non-negative for all t if M is positive definite. The Cauchy Schwartz inequality follows from requiring that the minimum value of ϕ is not negative, assuming $M^* = M$.

(e) Use the Cauchy Schwartz inequality to verify the triangle inequality in its squared form $\|u+v\|^2 \leq \|u\|^2 + 2\,\|u\|\,\|u\| + \|v\|^2$.

(f) Show that if $M = I$ then $\|u\|$ is the l^2 norm of u.

5. Verify that $\|p\|$ defined by (4.3) on $V = P_3$ is a norm as long as $a < b$.

6. Suppose A is the $n \times n$ matrix that represents a linear transformation from R^n to R^n in the standard basis e_k. Let B be the matrix of the same linear transformation in the scaled basis (4.5).

   (a) Find a formula for the entries b_{jk} in terms of the a_{jk} and $\overline{u}_k$.

   (b) Find a matrix formula for B in terms of A and the *diagonal scaling* matrix $W = \text{diag}(\overline{u}_k)$ (defined by $w_{kk} = \overline{u}_k$, $w_{jk} = 0$ if $j \neq k$) and W^{-1}.

7. Show that if $u \in R^m$ and $v \in R^n$ and $A = uv^*$, then $\|A\|_{l^2} = \|u\|_{l^2} \cdot \|v\|_{l^2}$. Hint: Note that $Aw = mu$ where m is a scalar, so $\|Aw\|_{l^2} = |m| \cdot \|u\|_{l^2}$. Also, be aware of the *Cauchy Schwarz* inequality: $|v^* w| \leq \|v\|_{l^2}\,\|w\|_{l^2}$.

8. Suppose that A is an $n \times n$ invertible matrix. Show that

$$\left\|A^{-1}\right\| = \max_{u \neq 0} \frac{\|u\|}{\|Au\|} = \left(\min_{u\neq 0} \frac{\|Au\|}{\|u\|}\right)^{-1} .$$

9. The *symmetric part* of the real $n \times n$ matrix is $M = \frac{1}{2}\left(A + A^*\right)$. Show that $\triangledown\left(\ \frac{1}{2}x^* Ax\ \right) = Mx$.

10. The *informal* condition number of the problem of computing the action of A is

$$\kappa(A) = \max_{x\neq 0,\ \Delta x \neq 0} \frac{\frac{\|A(x+\Delta x) - Ax\|}{\|Ax\|}}{\frac{\|x+\Delta x\|}{\|x\|}} .$$

Alternatively, it is the sharp constant in the estimate

$$\frac{\|A(x+\Delta x) - Ax\|}{\|Ax\|} \leq C \cdot \frac{\|x+\Delta x\|}{\|x\|} ,$$

which bounds the worst case relative change in the answer in terms of the relative change in the data. Show that for the l^2 norm,

$$\kappa = \sigma_{max}/\sigma_{min} ,$$

the ratio of the largest to smallest singular value of A. Show that this formula holds even when A is not square.

11. We wish to solve the *boundary value problem* for the differential equation

$$\frac{1}{2}\partial_x^2 u = f(x) \quad \text{for } 0 < x < 1, \tag{4.44}$$

with *boundary conditions*

$$u(0) = u(1) = 0\ . \tag{4.45}$$

We *discretize* the interval $[0,1]$ using a uniform *grid* of points $x_j = j\Delta x$ with $n\Delta x = 1$. The $n-1$ unknowns, U_j, are approximations to $u(x_j)$, for $j = 1, \ldots, n-1$. If we use a second order approximation to $\frac{1}{2}\partial_x^2 u$, we get discrete equations

$$\frac{1}{2}\frac{1}{\Delta x^2}\left(U_{j+1} - 2U_j + U_{j-1}\right) = f(x_j) = F_j\ . \tag{4.46}$$

Together with boundary conditions $U_0 = U_n = 0$, this is a system of $n-1$ linear equations for the vector $U = (U_1, \ldots, U_{n-1})^*$ that we write as $AU = F$.

(a) Check that there are $n-1$ distinct eigenvectors of A having the form $r_{kj} = \sin(k\pi x_j)$. Here r_{kj} is the j component of eigenvector r_k. Note that $r_{k,j+1} = \sin(k\pi x_{j+1}) = \sin(k\pi(x_j + \Delta x))$, which can be evaluated in terms of r_{kj} using trigonometric identities.

(b) Use the eigenvalue information from part (a) to show that $\left\|A^{-1}\right\| \to 2/\pi^2$ as $n \to \infty$ and $\kappa(A) = O(n^2)$ (in the informal sense) as $n \to \infty$. All norms are l^2.

(c) Suppose $\widetilde{U}_j = u(x_j)$ where $u(x)$ is the exact but unknown solution of (4.44), (4.45). Show that if $u(x)$ is smooth then the *residual*[17], $R = A\widetilde{U} - F$, satisfies $\|R\| = O(\Delta x^2) = O(1/n^2)$. For this to be true we have to adjust the definition of $\|U\|$ to be consistent with the L^2 integral $\|u\|_{L^2}^2 = \int_{x=0}^1 u^2(x)dx$. The discrete approximation is $\|U\|_{l^2}^2 = \Delta x \sum_{k=1}^n U_j^2$.

(d) Show that $A\left(U - \widetilde{U}\right) = R$. Use part (b) to show that $\left\|U - \widetilde{U}\right\| = O(\Delta x^2)$ (with the Δx modified $\|\cdot\|$).

(e) (harder) Create a fourth order five point central difference approximation to $\partial_x^2 u$. You can do this using Richardson extrapolation from the second order three point formula. Use this to find an A so that solving $AU = F$ leads to a fourth order accurate U. The hard part is what to do at $j = 1$ and $j = n-1$. At $j = 1$ the five point approximation to $\partial_x^2 u$ involves U_0 and U_{-1}. It is fine to take $U_0 = u(0) = 0$. Is it OK to take $U_{-1} = -U_1$?

[17]Residual refers to the extent to which equations are not satisfied. Here, the equation is $AU = F$, which $\widetilde{U}$ does not satisfy, so $R = A\widetilde{U} - F$ is the residual.

(f) Write a program in Matlab to solve $AU = F$ for the second order method. The matrix A is symmetric and *tridiagonal* (has nonzeros only on three *diagonals*, the *main diagonal*, and the immediate sub and super diagonals). Use the Matlab matrix operation appropriate for symmetric positive definite tridiagonal matrices. Do a convergence study to show that the results are second order accurate.

(g) (extra credit) Program the fourth order method and check that the results are fourth order when $f(x) = \sin(\pi x)$ but not when $f(x) = \max(0, .15 - (x - .5)^2)$. Why are the results different?

12. This exercise explores conditioning of the non-symmetric eigenvalue problem. It shows that although the problem of computing the fundamental solution is well-conditioned, computing it using eigenvalues and eigenvectors can be an unstable algorithm because the problem of computing eigenvalues and eigenvectors is ill-conditioned. For parameters $0 < \lambda < \mu$, there is a *Markov chain transition rate* matrix, A, whose entries are $a_{jk} = 0$ if $|j - k| > 1$ If $1 \leq j \leq n - 2$, $a_{j,j-1} = \mu$, $a_{jj} = -(\lambda + \mu)$, and $a_{j,j+1} = \lambda$ (taking j and k to run from 0 to $n - 1$). The other cases are $a_{00} = -\lambda$, $a_{01} = \lambda$, $a_{n-1,n-1} = -\mu$, and $a_{n-1,n-2} = \mu$. This matrix describes a continuous time Markov process with a random walker whose position at time t is the integer $X(t)$. Transitions $X \to X + 1$ happen with rate λ and transitions $X \to X - 1$ have rate μ. The transitions $0 \to -1$ and $n - 1 \to n$ are not allowed. This is the *M/M/1 queue* used in operations research to model queues ($X(t)$ is the number of customers in the queue at time t, λ is the rate of arrival of new customers, μ is the service rate. A customer arrival is an $X \to X + 1$ transition.). For each t, we can consider the row vector $p(t) = (p_1(t), \ldots, p_n(t))$ where $p_j(t) = \text{Prob}(X(t) = j)$. These probabilities satisfy the differential equation $\dot{p} = \frac{d}{dt}p = pA$. The solution can be written in terms of the *fundamental solution*, $S(t)$, which in an $n \times n$ matrix that satisfies $\dot{S} = SA$, $S(0) = I$.

(a) Show that if $\dot{S} = SA$, $S(0) = I$, then $p(t) = p(0)S(t)$.

(b) The *matrix exponential* may be defined through the Taylor series $\exp(B) = \sum_{k=0}^{\infty} \frac{1}{k!} B^k$. Use matrix norms and the fact that $\left\|B^k\right\| \leq \|B\|^k$ to show that the infinite sum of matrices converges.

(c) Show that the fundamental solution is given by $S(t) = \exp(tA)$. Top do this, it is enough to show that $\exp(tA)$ satisfies the differential equation $\frac{d}{dt}\exp(tA) = \exp(tA)A$ using the infinite series, and show $\exp(0A) = I$.

(d) Suppose $A = L\Lambda R$ is the eigenvalue and eigenvector decomposition of A, show that $\exp(tA) = L\exp(t\Lambda)R$, and that $\exp(t\Lambda)$ is the obvious diagonal matrix.

(e) Use the Matlab function `[R,Lam] = eig(A);` to calculate the eigenvalues and right eigenvector matrix of A. Let r_k be the k^{th} column of R. For $k = 1, \ldots, n$, print r_k, Ar_k, $\lambda_k r_k$, and $\|\lambda_k - Ar_k\|$ (you choose the norm). Mathematically, one of the eigenvectors is a multiple of the vector **1** defined in part h. The corresponding eigenvalue is $\lambda = 0$. The computed eigenvalue is not exactly zero. Take $n = 4$ for this, but do not hard wire $n = 4$ into the Matlab code.

(f) Let $L = R^{-1}$, which can be computed in Matlab using `L=R^(-1);`. Let l_k be the k^{th} row of L, check that the l_k are left eigenvectors of A as in part e. Corresponding to $\lambda = 0$ is a left eigenvector that is a multiple of p_∞ from part h. Check this.

(g) Write a program in Matlab to calculate $S(t)$ using the eigenvalues and eigenvectors of A as above. Compare the results to those obtained using the Matlab built in function `S = expm(t*A);`. Use the values $\lambda = 1$, $\mu = 4$, $t = 1$, and n ranging from $n = 4$ to $n = 80$. Compare the two computed $\widehat{S}(t)$ (one using eigenvalues, the other just using `expm`) using the l^1 matrix norm. Use the Matlab routine `cond(R)` to compute the condition number of the eigenvector matrix, R. Print three columns of numbers, n, error, condition number. Comment on the quantitative relation between the error and the condition number.

(h) Here we figure out which of the answers is correct. To do this, use the known fact that $\lim_{t\to\infty} S(t) = S_\infty$ has the simple form $S_\infty = \mathbf{1}p_\infty$, where **1** is the column vector with all ones, and p_∞ is the row vector with $p_{\infty,j} = ((1-r)/(1-r^n))r^j$, with $r = \lambda/\mu$. Take $t = 3 * n$ (which is close enough to $t = \infty$ for this purpose) and the same values of n and see which version of $S(t)$ is correct. What can you say about the stability of computing the matrix exponential using the ill conditioned eigenvalue/eigenvector problem?

13. This exercise explores eigenvalue and eigenvector perturbation theory for the matrix A defined in exercise 12. Let B be the $n \times n$ matrix with $b_{jk} = 0$ for all (j,k) except $b_{00} = -1$ and $b_{1,n-1} = 1$ (as in exercise 12, indices run from $j = 0$ to $j = n-1$). Define $A(s) = A + sB$, so that $A(0) = A$ and $\frac{dA(s)}{ds} = B$ when $s = 0$.

(a) For $n = 20$, print the eigenvalues of $A(s)$ for $s = 0$ and $s = .1$. What does this say about the condition number of the eigenvalue eigenvector problem? All the eigenvalues of a real tridiagonal matrix are real[18] but that $A(s = .1)$ is not tridiagonal and its eigenvalues are not real.

(b) Use first order eigenvalue perturbation theory to calculate $\dot{\lambda}_k = \frac{d}{ds}\lambda_k$ when $s = 0$. What size s do you need for $\Delta\lambda_k$ to be accurately approximated by $s\dot{\lambda}_k$? Try $n = 5$ and $n = 20$. Note that first order perturbation theory always predicts that eigenvalues stay real, so $s = .1$ is much too large for $n = 20$.

[18]It is easy to see that if A is tridiagonal then there is a diagonal matrix, W, so that WAW^{-1} is symmetric. Therefore, A has the same eigenvalues as the symmetric matrix WAW^{-1}.