Scientific Computing
Chapter III
Numerical Linear Algebra I, Review and Conditioning

Jonathan Goodman
Courant Institute of Mathemaical Sciences

Last revised February 25, 2002

1 Introduction

Linear algebra and calculus are the basic tools of quantitative science. The previous chapter is about how to carry out the operations of calculus, differentiation and integration. Here we discuss the operations of linear algebra: 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 canned software you buy or download. Therefore we spend more time here on problem formulation and conditioning than on specific computational algorithms.

In practical application of linear algebra, one often encounters ill conditioned problems. The best available linear algebra software is good enough usually to give results nearly as accurately as possible, so our practical worries in routine linear algebra computations will be with conditioning. Condition number concerns the impact on the solution of small perturbations in the data. In linear algebra, the data, and the perturbations, are vectors or matrices. Thus, to discuss condition number, we must first discuss ways to measure the size of a matrix or vector. This leads to the notion of the "norm" of a vector or matrix. Once we adopt norms as measures of size, we quickly find the famous condition number formula $\kappa = \|A\| \left\|A^{-1}\right\|$. This is the condition number for solving a system of equations with matrix A. Other problems, such as least squares or eigenvalue problems, have different condition numbers. For example, it may be easy to find the eigenvalues of A even though it is hard to solve a system of equations involving A, or the other way around.

This chapter begins with a review of linear algebra in order to state the various problems we might want to solve. We also review matrix factorizations and their computational applications. Then we discuss the conditioning of several problems. In some cases, a problem may converted into a different but mathematically equivalent problem for solution. The choice of approach may determined by conditioning; one formulation may be better conditioned than another. The best known example of this least squares, which may be solved directly or by first converting to a system of "normal equations". For well conditioned problems, the normal equations formulation has advantages. However, the condition number of the matrix appearing in the normal equations is roughly the square of the condition number of the least squares problem stated directly. This makes the direct approach preferable for problems with worse condition numbers.

It is important to be aware of the vast and important areas of numerical linear algebra that are not covered in this chapter or elsewhere in the notes. First, we neglect many details of the standard algorithms of elimination and substitution. This is for reasons of space and time, and because readily available software makes use of much of this.

More importantly, we completely neglect specialized techniques for large and "sparse" matrices. The generic "dense matrix" algorithm for solving a system of linear equations $Ax = b$ with an $n \times n$ matrix A requires $O(n^3)$ operations, even though there are only $O(n^2)$ elements of A. As matrices get big, say $n \geq 1000$, run time can become a serious issue. Fortunately, many large matrices have a majority of elements equal to zero. Sparse matrix algorithms only store and do arithmetic on the non zero elements, leading to great savings in work in many cases. Good sparse matrix software is widely available.

More generally, there are matices, A, that we are unable to store explicitly, even in sparse matrix format. In those cases we may be able to perform linear algebra using only procedures that compute Ax for a vector, x. For example, in solving partial differential equations, it is common to have $n > 10^6$, and algorithms to compute Ax from x in $O(n)$ or $O(n\log(n))$ work. Linear algrbra algorithms using only these operations are called "iterative methods". They are the most common approaches to truly large problems. The best iterative methods make use of problem specific approximate solutions called "preconditioners". Good preconditioners are based on physical or mathematical properties of the specific problem rather than on general recipes. This is the last I have to say about them.

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. I make use of many things from linear algebra, such as matrix inverses, without explanation. In my experience, people come to scientific computing with vastly differing points of view in linear algebra. This section should give everyone a common language.

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 (either real or complex numbers, depending on the application). Vector addition is supposed to be commutitive ($u + v = v + u$) and associative ($(u + v) + w = u + (v + w)$). Multiplication by scalars should be distributive over vector addition ($x(u+v) = xu+xv$ for scalar x and vector u), and conversely ($(x + y)u = xu + yu$). There should be 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" when the result is always an element of V'. 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 I 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 component sum is one ($\sum_{k=1}^n u_k = 1$) is not closed under vector addition or scalar multiplication.

A "basis" for vector space V is a set of vectors $f_1, \ldots, f_n$ 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, being the vectors with all zero components but for a single 1 in position k. This is a basis because

$$u = \begin{pmatrix} u_1 \\ u_2 \\ \cdot \\ \cdot \\ u_n \end{pmatrix} = u_1 \begin{pmatrix} 1 \\ 0 \\ \cdot \\ \cdot \\ 0 \end{pmatrix} + u_2 \begin{pmatrix} 0 \\ 1 \\ \cdot \\ \cdot \\ 0 \end{pmatrix} + \cdots + u_n \begin{pmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ 1 \end{pmatrix} \ .$$

In view of this, there is not much 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"[1].

Polynomials provide examples of abstract 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 would have to multiply out the last example to write it as a linear combination of powers of x, and it is a polynomial (by our definition) because this is possible. 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 consists of Lagrange interpolating polynomials for the points 1, 2, 3, and 4:

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

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

These are useful for interpolation because, for examaple, $l_1(1) = 1$ while $l_2(1) = l_3(1) = l_4(1) = 0$. If I 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 $0 = u_1 l_1 + u_2 l_2 + u_3 l_3 + u_4 l_4$ then the sum is equal to the zero polynomial, which is equal to zero for any value of x, and particularly, at $x = 1$, 2, 3, and 4, so $u_1 = 1$, $u_2 = 0$, $u_3 = 0$, and $u_4 = 0$. The set of all polynomials with no restriction on degree is a vector space without a finite basis, an infinite dimensional space.

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

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 ov 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.

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 vector 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 $u = \sum_k u_u f_k$. 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 multiplication.

If the linear transformation, L, is between abstract finite dimensional vector spaces that are not R^n, the matrix representing L depends on the basis. For example, suppose $V = P_3$ $W = P_2$, and L represents differentiation:

$$L(p_0 + p_1 x + p_2 x^2 + p_3 x^3) = p_1 + 2p_2 x + 3p_3 x^2\ .$$

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.

A matrix may also 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, and is a vector with expansions $v = \sum_k u_k f_k$ and $v = \sum_j w_j g_j$, then we may write

$$\begin{pmatrix} u_1 \\ \cdot \\ \cdot \\ u_n \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & \cdot & a_{1n} \\ a_{21} & a_{22} & \cdot & a_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & \cdot & a_{nn} \end{pmatrix} \begin{pmatrix} w_1 \\ \cdot \\ \cdot \\ w_n \end{pmatrix}$$

The matrix elements a_{jk} are the expansion coefficients of g_j with respect to the f_k basis. We use the convention that a_{jk} represents the (j,k) entry of the matrix A. For example, suppose $u \in P_3$ is given in terms of Hermite polynomials or simple powers: $u = \sum_{j=0}^{3} u_j H_j(x) = \sum_{k=0}^{3} w_j x^j$, then

$$\begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix}$$

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

$$\begin{pmatrix} w_1 \\ \cdot \\ \cdot \\ w_n \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & \cdot & a_{1n} \\ a_{21} & a_{22} & \cdot & a_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & \cdot & a_{nn} \end{pmatrix}^{-1} \begin{pmatrix} u_1 \\ \cdot \\ \cdot \\ u_n \end{pmatrix}$$

Two bases must have the same number of elements because only square matrixes are invertable. Of course, any $m \times n$ matrix defines a linear transformation from R^n or C^n to R^m or C^m. Even if the entries in the matrix are real, it might be better to think of C^n rather than R^n, for example if we are looking for eigenvalues that might be complex.

2.3 Vector norms

As explained above, in numerical linear algebra we need measures of the sizes of vectors and matrices. We want to extract a single number from the coordinates of a vector to say how big it is. There are several different ways to do this, but most of them are "norms", which we now describe. A norm on a vector space gives a number, written $\|u\|$, the norm of u for each $u \in V$. This norm should have several natural properties. First, it should be positive or zero, and 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 in V.

There are several "natural" 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:

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

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

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

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 of n events, 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 if our vector space is a 3 dimensional subspace of R_0^3, with a basis f_1, f_2, and f_3, it might take some work to compute the l^1 norm of a vector $u = u_1 f_1 + u_2 f_2 + u_3 f_3$ in terms of u_1, u_2, and u_3.

The notion of vector norms, though very useful, is not perfect. For one thing, the choice of norms 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)

$$u = \begin{pmatrix} 45 \\ 500000 \\ 2 \end{pmatrix}$$

In situations like these, the condition number of matrices is often greatly improved by "balancing" or "diagonal scalings", which amount to choosing units in which all the components are of the same order of magnitude.

2.4 Norms of matrices and linear transformations

Suppose A is a matrix representing a linear transformation, L, from V to W. For this, we must suppose that we have a basis for V and a basis for W. If we have norms for the spaces V and W, we can define a corresponding norm of L, written $||L||$, or $||A||$. In some cases it will be easy to calculate $||A||$ from the entries of A, but in other cases, it may involve more computing, such as a singular value decomposition in the case of l^2 norms. I find myself using the theoretical definition and properties of matrix or linear transformation norms much more than computing them.

The norm of a linear transformation is the largest amount by which it stretches a vector:

$$||L|| = \max_{u \neq 0} \frac{||Lu||}{||u||} \quad . \tag{1}$$

We often use the related inequality

$$||Lu|| \leq ||L|| \, ||u|| \quad . \tag{2}$$

The norm of L is the smallest number we can multiply $||u||$ by in (2) to get an inequality that is valid for every $u \in V$. For this reason, we have an inequality about products of matrices:

$$||AB|| \leq ||A|| \, ||B|| \quad ,$$

the maximum stretch of the product, first apply B the A, is not more than the maxumum stretch from the first operation, B, multiplied by the maximum stretch from A. If $V = R^n$ and $W = R^m$ and we use the same type of norm for both, then we label the matrix norm accordingly. For example

$$||A||_{l^2} = \max_{u \neq 0} \frac{||Au||_{l^2}}{||u||_{l^2}} \quad .$$

The following formulae are well known and easy to derive:

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

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

the maximum "row sum" or "column sum" respectively. We can get the l^2 norm of A, from the "singular value decomposition" of A but there is no simple direct formula for it.

The reader might be annoyed that there are so many different matrix and vector norms. For the purpose of studying stability and condition number, the three norms $||A||_{l^2}$, $||A||_{l^1}$, and $||A||_{l^\infty}$ are nearly equivalent, since they only differ by a factor of the dimension, n. In practical situations, if a problem is well or ill conditioned with respect to one of these norms, it probably is also with respect to the other two, since $n \leq 1000$ on modern computers if we explicitly store the matrix. For very large matrices not stored explicitly, the differences among norms can matter.

2.5 Eigenvalues and eigenvectors

Let A be an $n \times n$ matrix. We say that λ is an "eigenvalue", and $r \neq 0$ the corresponding "eigenvector" if

$$Ar = \lambda r \ .$$

Eigenvalues and eigenvectors are useful in understanding dynamics related to A. For example, "applying" A repeatedly to an eigenvector is the same as multiplying that vector repeatedly by λ:

$$A^k r = \lambda^k r \ .$$

Ordinarily it might be hard to say much about what happens when we apply a matrix many times. Eigenvalues and their relatives, "singular values", are the basis of "principal component analysis" in statistics. In general, the eigenvalues and eigenvectors may be complex even though the linear transformation, A, is real. This is one reason people work with complex vectors, C^n, even for applications that seemed to call for R^n.

Even though their descriptions seem similar, the "symmetric eigenvalue problem" (A symmetric), is vastly different from the general, or "unsymmetric" problem. This contrast is detailed below. Here I want to emphasize the differences in conditioning. 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 uncomputable (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 here is wrong if A happens to be symmetric, but some of the statements might be misleading. A 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. 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.

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

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

$$AR = R\Lambda \; . \tag{3}$$

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 invertable, we can multiply (3) 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 $L = R^{-1}$ to get

$$LA = \Lambda L \; . \tag{4}$$

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

$$\left.\begin{array}{ll} l_k r_k = 1 & \\ l_j r_k = 0 & \text{if } j \neq k. \end{array}\right\} \tag{5}$$

These are called "biorthogonality"relations. For examaple,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 they (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.

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 $\Lambda = 0$. 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 (have a complete set of n linearly independent eigenvectors). 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 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 (5) become the orthogonality relations:

$$\left.\begin{array}{ll} r_j^* r_j = 1 & \\ r_j^* r_k = 0 & \text{for } j \neq k. \end{array}\right\}$$

This may be restated by saying that the eigenvector matrix, R, is an orthogonal matrix:

$$L = R^{-1} = R^* \;\; , \;\; \text{or} \qquad R^* R = I \; .$$

2.6 Matrix decompositions and factorizations

A matrix factorization is an expression of a given matrix as a product of other matrices of specified types. Many important algorithms of numerical linear algebra may be expressed as methods for computing or using certain matrix factorizations. For example, Gauss elimination computes the LU decomposition of a general matrix square or the Choleski decomposition of an SPD matrix (definitions below). The QR factorization of a rectangular matrix solves linear least squares problems and finds bases for subspaces. Eigenvalues and eigenvectors may be organized into another factorization. The "singular value decomposition" contains the information of a principal component analysis in statistics.

A matrix, L is "lower triangular" if all its entries above the diagonal are zero: $l_{jk} = 0$ for $k > j$. A matrix, U is "upper triangular" if its entries vanish below the diagonal: $u_{jk} = 0$ if $k < j$. The matrix is "strictly" upper or lower triangular if the diagonal entries also are zero. Upper and lower triangular matrices need not be square. A matrix is a "permutation matrix" if there is a single one in each row and in each column. Applying a permutation matrix to a column vector just permutes the entries of the vector.

Any nonsingular matrix, A has an LU factorization. This almost, but not quite, means that we may find a square lower triangular L and upper triangular U so that $A = LU$. First, we usually normalize to make the diagonal entries of U be one. With this normalization, the LU factors, if they exist, are uniquely determined by A. Second, we may need to reorder (permute) the rows and/or columns of A before the factorization is possible. For example, it is impossible to write

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} l_{11} & 0 \\ l_{21} & l_{22} \end{pmatrix} \begin{pmatrix} 1 & u_{12} \\ 0 & 1 \end{pmatrix} .$$

3 Condition number

The condition number of a problem depends on what the problem is. A matrix may have different condition numbers depending on what we are trying to do with the matrix. When someone says "the condition number of A", he or she is usually referring to the conditioning of solving a system of equations

$$Ax = b \tag{6}$$

where the vector b and matrix A are given and we want x. We might also want the condition number of the eigenvalue problem for A. Finding the eigenvalues might be better conditioned than finding the eigenvectors, etc. For example, if A is a singular matrix, then the linear system (6) is not solvable ("infinitely ill conditioned"?) but the eigenvalue problem is fine, just with a zero eigenvalue. For a symmetric matrix that is very near the identity matrix, the eigenvalue problem is fine but eigenvectors are an ill conditioned function of the matrix.

3.1 Linear systems, direct estimates

For solving a linear system (6), the data are the matrix A and the right hand side vector, b. We want to measure the relative change in the answer, x caused by a small relative change in b or A. We measure the size of a vector using its norm, which was the point of introducing norms. The result for small perturbations in b is

$$\frac{\|\Delta x\|}{\|x\|} \leq \kappa(A)\frac{\|\Delta b\|}{\|b\|} \ , \tag{7}$$

where the condition number, $\kappa(A)$, is given by

$$\kappa(A) = \|A\| \left\|A^{-1}\right\| \ . \tag{8}$$

As with the norm, the condition may be defined as the smallest number that makes the inequality (7) true for any b and sufficiently small Δb. An interesting feature of the formula (8) is that $\kappa(A)$ does not change if I multiply each element of A by the same number. If I use $321 \cdot A$ instead of A, then x and Δx change by the same factor, 321, but their ratio does not change. Even though we have an explicit formula for $\|A\|$ in some cases, I don't know of any explicit formula for $\kappa(A)$ for any norm. The l^2 condition number can be computed in terms of the singular values of A: $\kappa(A)_{l^2} = \sigma_{\text{max}}(A)/\sigma_{\text{min}}(A)$

The formula (8) states that the right side of (8) is the smallest possible number that makes the error bound (7) true. This is really two statements, first that the inequality (7) is always true for κ given by (8), and, second, that no smaller constant will do, the estimate is "sharp". Do do this, we start with the equations (6) and the perturbation of it

$$A(x + \Delta x) = b + \Delta b \ .$$

Subtracting these gives the equation for the perturbation:

$$A\Delta x = \Delta b$$

which we rewrite as

$$\Delta x = A^{-1}\Delta b \tag{9}$$

To establish a bound for the ratio $\|\Delta x\| / \|x\|$ we need to show that the numerator ($\|\Delta x\|$) is small while the denominator ($\|x\|$) is large. For this we use the two norm relations, first from (9):

$$\|\Delta x\| \leq \left\|A^{-1}\right\| \|\Delta b\| \ , \tag{10}$$

then from (6),

$$\|b\| \leq \|A\| \|x\| \ .$$

The latter shows that $\|x\|$ is larger than something, so it may be inverted to show that $1/\|x\|$ is smaller than something:

$$\frac{1}{\|x\|} \leq \frac{\|A\|}{\|b\|} \ . \tag{11}$$

Multiplying the bounds (10) and (11) together gives the bound (7).

The "sharpness" statement is that if

$$\frac{\|\Delta x\|}{\|x\|} \leq C\frac{\|\Delta b\|}{\|b\|} \ ,$$

then $C \geq \|A\| \left\|A^{-1}\right\|$. This is because the definition of the matrix norm calls for sharp inequalities. That is, there is some[2] $x_* \neq 0$ so that (with $b_* = Ax_*$)

$$\|b_*\| = \|A\| \, \|x_*\| \ ,$$

and there is a Δb_* (with $\Delta x_* = A^{-1} \Delta b_*$)[3] so that

$$\|\Delta x_*\| = \left\|A^{-1}\right\| \|\Delta b_*\| \ .$$

For these vectors we have

$$\frac{\|\Delta x_*\|}{\|x_*\|} = \|A\| \left\|A^{-1}\right\| \frac{\|\Delta b_*\|}{\|b_*\|} \ .$$

This shows that no constant smaller than (8) is possible.

The above argument may be restated in simple terms. We want to compute x by solving $Ax = b$. The right hand side, b is not known exactly, either through roundoff or measurement error. Thus, instead of our intended b b we have $b + \Delta b$ and know little about Δb except for it's rough size. Then look for the worst thing that can happen with $\|\Delta b\| \leq \epsilon \|b\|$ (relative error of order ϵ) but otherwise arbitrary. The relative error is $\|\Delta x\| / \|x\|$. How large this ratio can be is determined by what A^{-1} does to the norms of vectors, how much A^{-1} "stretches" them. The worst case is maximal stretching for Δb and minimal stretching (maximal contraction) for b. The maximal stretching for b is given by $\left\|A^{-1}\right\|$. The minimal stretching for A^{-1} is the maximal stretching for A, which is $\|A\|$. The ratio (max stretch)/(min stretch) is the condition number. The identity matrix or multiples of it apply the same stretch to every vector so their condition number is one. A diagonal matrix such as

$$\begin{pmatrix} 100000 & 0 \\ 0 & 10 \end{pmatrix}$$

has a large condition number, $100000/10 = 10000$ in this case.

3.2 Linear systems, perturbation theory

The discussion of conditioning in the previous subsection is rather different from the generic discussion of conditioning given earlier. The generic condition number depends on the derivative of the answer with respect to the data, but there did not seem to be any differentiation above. This is because the dependence of x on b is linear, so there is little difference between x as a function of b and the derivative of this function. Most other dependences are nonlinear and push us back to the generic point of view from previous chapters.

[2]There is a subtle point here for the mathematically minded reader. The maximum value in the norm definition (1) is actually attained. This is because we may as well maximize over the set of u with $\|u\| = 1$, which is a compact set. Since the ratio is a continuous function of u for $u \neq 0$, its maximum is achieved.

[3]This Δb_* is not a perturbation of b_*, but is just some vector, the worst possible perturbation.

In particular, the dependence of x on A is nonlinear. If we have[4] $Ax_1 = b_1$ and $Ax_2 = b_2$, then $A(x_1 + x_2) = b_1 + b_2$; the dependence of x on b is linear. However, if $A_1 x_1 = b$ and $A_2 x_2 = b$, then a simple calculation shows that it is not true that $(A_1 + A_2)(x_1 + x_2) = b$; the dependence of x on A is nonlinear. If we want to understand the conditioning of x as a function of A, we must differentiate the function that defines x as a function of A, as in the generic condition number discussion.

Taking derivatives of the answer with respect to the data, particularly in linear algebra, is often called "perturbation theory": we study how the answer changes when we perturb the data. One of the many approaches to perturbation theory is to imagine that the data, A in this case, depends on an artificial parameter, s. When $s = 0$ we have the "original" or "unperturbed" A. When s is close to zero, there is a $\Delta A \approx A'\Delta s = A's$. We denote differentiation with respect to s with a prime and leave out the argument s when $s = 0$. Thus, $A\prime$ means $\frac{d}{ds}A(s = 0)$. Be careful, in Matlab A' means the transpose of A. We write either A^t or A^* to represent the transpose, but never A'. In principle, we do not need the s parameter, and could directly approximate the Δx resluting from a ΔA. Physicists usually proceed in this direct way. Our approach leads to the same formula for Δx without rehashing arguments from basic calculus, but at the expense of adding an extra parameter.

The actual calculation is simple. We suppose that $A(s)$ depends in a smooth way on the parameter s with the unperturbed A equal to $A(0)$. Then, for each s, we consider[5] the solution of the system $A(s)x(s) = b$. Note that $x(s)$ depends on s but b does not. Differentiating this equation using the product rule and then setting $s = 0$ gives

$$A'x + Ax' = 0 \ ,$$

which leads to

$$x' = -A^{-1}A'x \ . \tag{12}$$

Next we multiply both sides by $s = \Delta s$ (perturbing about $s = 0$) and use the "first order" (but second order accurate) approximations $\Delta x \approx x'\Delta s$, and $\Delta A \approx A'\Delta s$. This gives our perturbation formula (using a $\cdot$ on the left for matrix and vector multiplication):

$$\Delta x \approx -A^{-1} \cdot \Delta A \cdot x \ .$$

Finally, we put in norms on both sides and use the multiplicative property of norms (??) to get

$$\|\Delta x\| \leq \left\|A^{-1}\right\| \|\Delta A\| \|x\| \ ,$$

[4]The subscripts 1 and 2 refer to two different vectors rather than to the first two components of a vector x.

[5]Physicists use the term "gedanken experiment", or "thought experiment" when they try to guess laws of nature by asking what would happen in some experiment that might not be possible to perform. For example, the young Albert Einstein is supposed to have started on the path that lead to relativity by trying to imaging what it would be like to run as fast as a light wave. Here, we need not write a computer program to solve the equations, but only to know that the solutions exist.

which may be rewritten to bring out the relative sizes of the perturbations:

$$\frac{\|\Delta x\|}{\|x\|} \leq \left\|A^{-1}\right\| \|A\| \frac{\|\Delta A\|}{\|A\|} \; . \tag{13}$$

This formula is only approximately true and then only when Δa is small enough. It shows that (8) gives the conditioning of x as a function of A as well as b. If we have a perturbation of A that is too large for (13) to be accurate, then probably Δx is too large to be acceptable. The relative error bound (13) is still not in the generic condition number setting. We can be exactly in that setting by working with the derivatives directly. If we take the norm of both sides of (12), and manipulate a bit, we get

$$\frac{\|x'\|}{|x\|} \leq \left\|A^{-1}\right\| \|A\| \frac{\|A'\|}{\|A\|} \; ,$$

which, but for the use of norms, is exactly as in the earlier chapter. Finally, the interested reader can show that (13) is approximately sharp for small Δx, just as we showed (8) and (7) is sharp.

3.3 Eigenvalues and eigenvectors

Perturbation theory provides a way to understand the conditioning of eigenvalues and eigenvectors. We use the formalism from the previous section. We have a matrix A and a small perturbation ΔA. We imagine a smooth function $A(s)$ depending on some parameter, s with $\Delta A = A'\Delta s$. Then we seek expressions for λ_k' or r_k' so that we may estimate, for example, $\Delta\lambda_k \approx \lambda_k'\Delta s$. In the end, everything will be expressed in terms of ΔA, which is what was given.

We start with perturbation theory of eigenvalues of a symmetric matrix. The famous formula[6] is

$$\lambda_k' = r_k^* A' r_k \; , \tag{14}$$

which may be rewritten

$$\Delta\lambda_k \approx r_k^* \Delta A r_k \; . \tag{15}$$

We can turn this into a condition number estimate, which means recasting it in terms of relative errors on both sides. The important observation is that $\|r_k\|_{l^2} = 1$, so $\|\Delta A r_k\| \leq \|\Delta a\|_{l^2}$ and finally

$$|r_k^* \Delta A r_k| \leq \|\Delta A\|_l^2 \; .$$

This inequality is sharp because we could take ΔA proportional to $r_k r_k^*$, which is an $n \times n$ matrix. Putting this into (**??**) gives the also sharp inequality,

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

[6] This is sometimes called "Rayleigh Schrödinger" perturbation theory. It was used by Lord Rayleigh to study vibrational frequencies of plates and shells and later by Schrödinger to compute energies (which are eigenvalues) in quantum mechanics.

from which we can read off the condition number

$$cond = \frac{\|A\|_{l^2}}{|\lambda_k|} = \frac{|\lambda|_{\max}}{|\lambda_k|} \tag{16}$$

As in Section ??, $\|A\| = |\lambda|_{\max}$ refers to the the eigenvalue of largest absolute value.

The condition number formula (16) is essentially be best possible condition number. Presumably λ_k 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, which are one the order of $\|A\|$. Only if λ_k 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. It is easy to guess and not so hard to show (see below) that the anologue of (14), when the eigenvalues are distinct, is

$$\lambda'_k = l_k A' r_k \ . \tag{17}$$

In the nonsymmetric 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 (17) as we did for the symmetric case leads to

$$\left| \frac{\Delta\lambda_k}{\lambda_k} \right| \leq \kappa(R) \frac{\|A\|}{|\lambda_k|} \frac{\|\Delta A\|}{\|A\|} \ .$$

Here $\kappa(R) = \left\|R^{-1}\right\| \|R\|$ is the linear systems condition number of the right eigenvector matrix. In the symmetric case, the only reason for ill conditioning is that we are looking for a (relatively) tiny number. For nonsymmetric matrices, it is also possible that the eigenvector matrix is ill conditioned. If a family of matrices approaches a matrix with a Jordan block, the condition number of R approaches infinity (Take my word for this). For a symmetric matrix, the condition number of R, using l^2 norms, is one; since R is an orthogonal matrix, both it and it's inverse have norm one.

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 seek to expand r'_k in terms of the eigenvectors r_k. Call the expansion coefficients m_{kj}, and we have

$$r'_k = \sum_j m_{kj} r_j \ .$$

For the symmetric eigenvalue problem, if all the eigenvalues are distince, the formula is

$$m_{kj} = r_j^* A' r_k \lambda_j - \lambda_k \ .$$

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 plausable. The unsymmetric matrix perturbation formula is

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

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 Resources

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 (still more theoretical) the one by Paul Halmos. An excellent discussion of condition number is given in the SIAM lecture notes of Lloyd N. Trefethen. A somewhat overcomplete guide to the computational algorithms may be found in the book of Gene Golub and Charles vanLoan.