Basic Linear Algebra

Definitions

Here I want to introduce some notation that I use (which may be non-standard) and some ideas which are basic and well known, but helpful to keep in mind when reading the ML literature. The vector $\mathbf{u}$ with n elements is either a horizontal or vertical collection of n numbers. In the former $\mathbf{u}$ is *row* vector (and I note its dimension $\mathbf{u} \equiv (1 \times n)$) and the latter it is a *column* vector ($\mathbf{u} \equiv (n \times 1)$). By default all vectors are column vectors. A column vector $\mathbf{u}$ can be written as a row by tranposing (ie $\mathbf{u}^T$). We define the two vector-vector products:

Vector Products

The *dot product* of two vectors $\mathbf{u}$ and $\mathbf{v}$ is the scalar

$$\mathbf{u} \cdot \mathbf{v} = \mathbf{u}^T \mathbf{v} = \sum \mathbf{u}_i \mathbf{v}_i \tag{1}$$

The *vector product* of two vectors $\mathbf{u}$ and $\mathbf{v}$ is the matrix

$$\mathbf{u}\mathbf{v}^T = \begin{pmatrix} u_1 v_1 & u_1 v_2 & \dots & u_1 v_n \\ u_2 v_1 & u_2 v_2 & \dots & u_2 v_n \\ \vdots & \vdots & \ddots & \vdots \\ u_n v_n & u_1 v_n & s & u_n v_n \end{pmatrix} \tag{2}$$

The vectors $\mathbf{u}$ and $\mathbf{v}$ are *orthogonal* if their dot products is zero, ie $\mathbf{u} \cdot \mathbf{v} = 0$. The $m \times n$ matrix A, where

$$A = \begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & A_{m2} & \cdots & A_{mn} \end{pmatrix} \tag{3}$$

is often decomposed into its m rows

$$A = \begin{pmatrix} \mathtt{r}\,(A)_1 \\ \mathtt{r}\,(A)_2 \\ \vdots \\ \mathtt{r}\,(A)_m \end{pmatrix} \tag{4}$$

or n columns

$$A = \begin{pmatrix} \mathtt{c}\,(A)_1 & \mathtt{c}\,(A)_2 & \dots & \mathtt{c}\,(A)_n \end{pmatrix} \tag{5}$$

so as to better represent matrix operations.

Matrix Products

There are a number of ways to represent matrix-vector and matrix-matrix products, and different representations are usefull in different contexts. Here I will go through some of the main ideas. It is good to go through each a couple of times using a (2×2) or (3×3) to convince yourself of the structure of the following products.

The matrix-vector product $A\mathbf{x}$ is most commonly viewed as the weighted sum of the columns of A

ie

$$A\mathbf{x} = \sum x_i \mathtt{c}\,(A)_i \tag{6}$$

or it can be interpreted as a sequence of dot products using the *rows* of A

$$A\mathbf{x} = \begin{pmatrix} \mathtt{r}\,(A)_1 \cdot \mathbf{x} \\ \mathtt{r}\,(A)_2 \cdot \mathbf{x} \\ \vdots \\ \mathtt{r}\,(A)_N \cdot \mathbf{x} \end{pmatrix} \tag{7}$$

As might be expected, the vector-matrix product $x^T A$ is a sequence of dot products using the *columns* of A

$$x^T A = \left(\mathbf{x}^T \cdot \mathtt{c}\,(A)_1 , \mathbf{x}^T \cdot \mathtt{c}\,(A)_2 , \ldots , \mathbf{x}^T \cdot \mathtt{c}\,(A)_n \right) \tag{8}$$

Matrix products AB where $A \equiv (m \times n)$ and $B \equiv (n \times m)$ are a matrix of dot products

$$AB = \begin{pmatrix} \mathtt{r}\,(A)_1 \cdot \mathtt{c}\,(B)_1 & \mathtt{r}\,(A)_1 \cdot \mathtt{c}\,(B)_2 & \cdots & \mathtt{r}\,(A)_1 \cdot \mathtt{c}\,(B)_m \\ \mathtt{r}\,(A)_2 \cdot \mathtt{c}\,(B)_1 & \mathtt{r}\,(A)_2 \cdot \mathtt{c}\,(B)_2 & \cdots & \mathtt{r}\,(A)_2 \cdot \mathtt{c}\,(B)_m \\ \vdots & \vdots & \ddots & \vdots \\ \mathtt{r}\,(A)_m \cdot \mathtt{c}\,(B)_m & \mathtt{r}\,(A)_1 \cdot \mathtt{c}\,(B)_m & \cdots & \mathtt{r}\,(A)_m \cdot \mathtt{c}\,(B)_m \end{pmatrix} \tag{9}$$

This representation is the most common way for me to work towards proofs of various matrix properties. It is also usefull to note that matrix-matrix products can be seen as a collection of the matrix-vector or vector-matrix products we have just seen. The matrix-matrix product AB is a row of matrix-vector products using the columns of B

$$AB = \left(A\mathtt{c}\,(B)_1 , A\mathtt{c}\,(B)_2 , \ldots , A\mathtt{c}\,(B)_n \right) \tag{10}$$

or it is a column of vector-matrix products using the rows of B

$$AB = \begin{pmatrix} \mathtt{r}\,(A)_1 B \\ \mathtt{r}\,(A)_2 B \\ \vdots \\ \mathtt{r}\,(A)_n B \end{pmatrix} \tag{11}$$

All of these interpretations will be used in the following notes (I think).

Special Matrices

There are a number of matrices with special structure which are found repeatedly in numerical analysis and machine learning. These are diagonal matrices, triangular matrices, symmetric matrices, orthogonal matrices and symmetric positive definite matrices.

Diagonal Matrices

Diagonal Matrices are denoted D and are all zero except the main diagonal, ie $D_{ij} = 0$ if $i \neq j$. Diagonal matrices are one of two types of *elementary matrices*

- Diagonal or Scaling matrices
- Permutation matrices

though we will delay discussion of permulation matrices until a little later. It is good to note that as elementary matrices, both diagonal and permulation matrices should be seen as matrices which perform basic transformations of other matrices, such as scaling columns/rows or by switching

rows/columns around. Diagonal matrices do that former - ie *Diagonal matrices are a scaling operation*. Diagonal matrices are denoted D, and permutation matrices P. A general elementary matrix is denoted E. Multiplication by an elementary matrix E transforms either rows or columns of its operand depending on whether we pre- or post- multiply (ie EA or AE).

Operation	Effect
pre-multiply (ie EA)	row transformation
post-multiply (ie AE)	column transformation

Here, put in some examples of elementary matrices.

As with other *elementary matrices*, pre-multiplying is a row transformation, post-multiplying is a column transformation. To see this, recognise the matrix product as a matrix of dot products. In the case of pre-multiplication, where D is diagonal with $D_{ii} = di$

$$\begin{aligned}(DA)_{ij} &= \mathtt{r}\,(D)_i \cdot \mathtt{c}\,(A)_j \\ &= D_{ii}A_{ij} \\ &= d_i A_{ij}\end{aligned}$$

where the second line occurs because all elements in $\mathtt{r}\,(D)_i = 0$ except the i^{th} element, so only the i^{th} element in $\mathtt{c}\,(A)_i$ (ie A_{ij}), survives the dot product. The result is the whole of the i^{th} row of A is multiplied through by d_i. The analgous result happens for post-multiplication. If A is *both* pre and post multiplied we will end up with all pairwise combinations of the diagonals D_{ii}, ie

$$(DAD)_{ij} = d_i d_j A_{ij} \tag{12}$$

This is usefull in particular with correlation and covariance matrices.

Triangular matrices are ubiquitious in computation linear algebra, because they can be solved with little effort. Matrices can be upper or lower triangular:

- *Upper Triangular:* U is upper triangular if all $U_{ij} = 0$ when $i < j$
- *Lower Triangular:* L is lower triangular if all $L_{ij} = 0$ when $i > j$

I often refer to lower and upper as being types of triangular matrices. Assume everywhere that T is a triangular matrix of some type, and U/L are upper/lower triangular matrices unless otherwise specified. Triangular matrices (aside from diagonal matrices, which are of course both upper and lower triangular) are very common because the systems of equations they represent are very easy to solve. Note that

- If T_1, T_2 have the same type, then the sum $T = T_1 + T_2$ also has the same type.
- If T_1, T_2 have the same type, then the product $T = T_1 T_2$ also has the same type.

Symmetric Matrices have corresponding elements across the main diagonal equal, ie $S_{ij} = S_{ji}$. This simple structure results in many useful properties, and many important matrices, such as correlation matrices are symmetric.

Orthogonal Matrices are matrices made up of orthogonal columns. Remember that two vectors $\mathbf{u}$ and $\mathbf{v}$ are orthogonal if $\mathbf{u} \cdot \mathbf{v} = 0$. If a matrix Q is orthogonal (often Q is used to represent orthogonal matrices) $\mathtt{c}\,(Q)_i \cdot \mathtt{c}\,(Q)_j = 0$ for any columns i, j. The special case Q where $\mathtt{c}\,(Q)_i \cdot \mathtt{c}\,(Q)_i = 1$ is referred to as *orthonormal*. Orthogonal matrices have excellent numerical properties, so many common numerical methods rely on finding orthogonal matrices. The critical feature of orthogonal

matrices is the product $Q^TQ = I$, since the ij^{th} element in the product

$$\begin{aligned}(Q^TQ)_{ij} &= \mathsf{c}(Q)_i^T \cdot \mathsf{c}(Q)_j \\ &= 0 \text{ if } i \neq j\end{aligned}$$

The critical point is that its transpose (which is easy to find) is its inverse. We will discuss inverses shortly.

Symmetric Positive Definite Matrices or *SPD* matrices are common and incredibly useful in both numerical and analytic work. An SPD matrix is a special case of a symmetric matrix (ie if S is SPD it is symmetric, though the converse is not necessarily true), where the *quadratic form* $x^TSx > 0$ is positive for any x. The *quadratic form* is a quantity that reappears a number of times in different contexts, so we should look at it briefly. The quadratic form is a function of x, and is essentially a weighted sum of squares of x where the elements in S make up the weights. For example the 2×2 case

$$\begin{aligned}x^TAx &= \begin{pmatrix} x_1 & x_2 \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ &= x_1^2 A_{11} + x_1 x_2 (A_{12} + A_{21}) + x_2^2\end{aligned}$$

More generally, the vector-matrix-vector product $\mathbf{y}^TAx$, which is a more general expression than the quadratic form, can be written as a sum of all elements in A in the form

$$\mathbf{y}^TAx = \sum_i \sum_j A_{ij} y_i x_j \tag{13}$$

which is shown below as it is a quite usefull algebra exercise

$$\begin{aligned}\mathbf{y}^TAx &= \left(\mathbf{y}^T\mathsf{c}(A)_1, \ \ldots, \ \mathbf{y}^T\mathsf{c}(A)_n\right)\mathbf{x} \\ &= x_1 (y_1A_{11} + y_2A_{21} + \ldots + y_nA_{n1}) + \ldots + x_n (y_nA_{1n} + y_2A_{2n} + \ldots + y_nA_{nn})\end{aligned}$$

where each element A_{ij} appears once in the sum in an expression $A_{ij}y_ix_j$, which gives the above identity. The special case here, where $\mathbf{y} = \mathbf{x}$ gives

$$\mathbf{x}^TAx = \sum_i \sum_j A_{ij} x_i x_j \tag{14}$$

Last, a common matrix expression in statistics/ML is the matrix $S = A^TA$, which is often called the Grammian. It is a particularly useful symmetric positive definite matrix. It is symmetric, since

$$\begin{aligned}S^T &= (A^TA)^T \\ &= A^T (A^T)^T \\ &= A^TA \\ &= S\end{aligned}$$

and positive definite, since

$$\begin{aligned}\mathbf{x}^TS\mathbf{x} &= \mathbf{x}^TA^TA\mathbf{x} \\ &= A\mathbf{x}^TA\mathbf{x} \\ &= \mathbf{u}^T\mathbf{u} \\ &= \sum u_i^2 \geq 0\end{aligned}$$

where $\mathbf{u} = A\mathbf{x}$. This actually guarantees *semi-positive definiteness* only, but if the columns of A are independant, then $S = A^T A$ is positive definite. We will look at independance in the next section, but note that being *positive definite* requires the following is non-zero (it cant be negative)

$$\begin{aligned}\mathbf{x}^T A^T A\mathbf{x} &= \mathbf{u}^T\mathbf{u} \\ &= |\mathbf{u}| \\ &= |A\mathbf{x}| > 0\end{aligned}$$

which is only zero when $A\mathbf{x} = 0$. If we have independance in A, we are guaranteed $S = A^T A$ is symmetric positive definite.

Column Space, Rank and Indepdance

Return the matrix-vector product $A\mathbf{x}$, which we saw is the weighted sum of the columns of A. The set of all possible combinations of the columns defines the *column space* of A - that is, all possible outputs of the operation $A\mathbf{x}$. This idea is coupled to the more general idea of a *vector space*. More specifically, the *vector space* generated by a set of n vectors $\{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$ is the set of all possible linear combinations of the $\mathbf{u}_i$. If we combine these vectors to form the columns of a matrix A, the we get the linear combination from the above matrix-vector product $A\mathbf{x}$. The column space of a matrix A then is the *vector space* generated by its columns. This *vector space*, which is the *column space* of A is often also called the *range* of A, ie

$$\text{range}\,(A) = \{\mathbf{u} \in \mathcal{R}^n \text{ where there exists } \mathbf{x} \text{ with } A\mathbf{x} = \mathbf{u}\} \tag{15}$$

since it defines which vectors can be produced by summing the columns in A. The *null space* is a particular set of non-zero weights $\mathbf{x}$ which produce the zero vector, ie

$$\text{null}\,(A) = \{\text{all } \mathbf{u} \text{ where } A\mathbf{u} = 0\} \tag{16}$$

Not all matrices have a non-empty null space. If such an $\mathbf{u}$ exists, then at least one column can be written as a linear combination of at least one of the others. This has significant implications - it means that one of the rows in A is a redundant piece of information. If no such combination exists - that is, the null space is empty, then the columns are *linearly independant* - no columns can be written as a combination of the others. This concept is again common more generally to sets of vectors - ie a set of vectors is linearly independant if no linear combination of the vectors produces a zero vector. Spaces which are *not* linearly independant include redundant vectors.

The *column rank* of matrix A is the largest number of independant columns in A. If all columns are independant, the matrix has *full rank*. It is denoted colrank (A). The *row rank* is defined similarly, but is used much less frequently - the *rank* of a matrix rank (A) usually refers to the column rank. Some examples would be good.

Consider a matrix A and some set of vectors $\begin{pmatrix}\mathbf{u}_1 & \mathbf{u}_2 & \ldots & \mathbf{u}_n\end{pmatrix}$. For every possible combination of the columns of A (ie every $\mathbf{v} = A\mathbf{u}$), it may be possible to choose some combination of the vectors $\mathbf{u}_i$ that produces the same $\mathbf{v}$, and vica versa. In this sense, the set of vectors $\begin{pmatrix}\mathbf{u}_1 & \mathbf{u}_2 & \ldots & \mathbf{u}_n\end{pmatrix}$ can substitute for the matrix A, since they can both generate the exact same output (ie they have the same range). This might be useful if the vectors $\mathbf{u}_i$ had properties that were more desireable than the columns of A. Indeed, this is done very commonly in numerical methods in linear algebra. If we want to do this substitution however, we shouldnt choose any redundant vectors in the $\mathbf{u}_i$ - we should only use *linearly independant* vectors. Such a set of vectors is said to form a *basis* for A.

Matrix Inverse and Non-Singular Matrices

Definitions

Recall the matrix-vector product $A\mathbf{x} = \mathbf{y}$. Is it possible to work out the particular $\mathbf{x}$ which results in a particular $\mathbf{y}$? Clearly $\mathbf{x}$ should be in column space of A, ie $\mathbf{x} \in \mathtt{c}(A)$. Also, if $\mathbf{x} \in \mathcal{R}^n$ so that we have n variables to find, we need n degrees of freedom - there can be no redundant information in the columns of A. Therefore A should have full rank, or equivalently have an empty null space. Let us formalise some of these ideas below.

An $(n \times n)$ matrix A is *non-singular* if there exists another matrix B where $AB = I$. If such a matrix exists is it unique, and we denote it $B = A^{-1}$. To see this, assume instead that $A^{-1} = B_1$ and $A^{-1} = B_2$ where $B_1 \neq B_2$ - that is, there are two different matrices which are inverses of A. By definition

$$AB_1 = I = B_1A$$
$$AB_2 = I = B_2A$$

and by multiplying the second equation by B_1 we see

$$(B_1A)\,B_2 = B_1I \tag{17}$$

where the bracketed term is I and we have $B_2 = B_1$. The previous definitions of *rank*, *range* and *null space* are usefull here. When determing if A^{-1} exists, remember

$$A^{-1} \iff \text{null}(A) = \{\varnothing\}$$
$$A^{-1} \iff \text{range}(A) = \mathcal{R}^n$$
$$A^{-1} \iff \text{rank}(A) = n$$

When finding if A is singular analytically, I ususally check if

$$\text{null}(A) = \{\varnothing\} \tag{18}$$

Some important identities that are used regularly are

$$(AB)^{-1} = B^{-1}A^{-1}$$
$$(A^T)^{-1} = \left(A^{-1}\right)^T$$

To check the first, let $M = AB$, and note that

$$B^{-1}A^{-1}M = B^{-1}\left(A^{-1}A\right)B = I$$

so that $M^{-1} = B^{-1}A^{-1} = (AB)^{-1}$. For the second, consider $(A^{-1})^T A^T = (AA^{-1})^T = I$

Note that if A is *diagonal* or *triangular*, its determinant is particular simple

$$\mathtt{det}(A) = \prod A_{ii} \tag{19}$$

Since A is non-singular if $\mathtt{det}(A) = 0$ diagonal and triangular matrices can have no zeros on the

main diagonal. If all diagonal elements are non-zero then A^{-1} exists. Also, if A^{-1} exists, then

$$\begin{aligned} A \text{ is diagonal} &\iff A^{-1} \text{ is diagonal} \\ A \text{ is upper triangular} &\iff A^{-1} \text{ is upper triangular} \\ A \text{ is lower triangular} &\iff A^{-1} \text{ is lower triangular} \end{aligned}$$

The inverse of a diagonal matrix D has

$$\left(D^{-1}\right)_{ii} = 1/D_{ii} \tag{20}$$

which is what we might expect when viewing D as a scaling operation (ie to make the inverse, just use the opposite scaling factor!).

Analytic Matrix Inverse

Here, add a discussion of elementary matrix operations, and finding inverses.

Numeric Matrix Inverse

Include a section on why we dont actually computin matrix inverses, conditioning etc.

Most numeric methods for solving systems of equations

$$A\mathbf{x} = \mathbf{y} \tag{21}$$

involve decomposing A into a product of two matrices (ie perform a *matrix decomposition*), one of which is triangular (or has some other simple form which makes solving fast and easy). The solution then usually proceeds in two steps. The most basic is the LU decomposition, where we find lower and upper triangular matrices L, U where $A = LU$. The above equation then reduces to

$$\begin{aligned} A\mathbf{x} &= \mathbf{y} \\ L\left(U\mathbf{x}\right) &= \mathbf{y} \\ L\mathbf{v} &= \mathbf{y} \end{aligned}$$

where $U\mathbf{x} = \mathbf{v}$. Here we solve for $\mathbf{v}$, which is very fast since L is triangular (as we will see below). Once we have $\mathbf{v}$, we use it to solve for $\mathbf{x}$ in $U\mathbf{x} = \mathbf{v}$. Again, since U is triangular this is very fast.

Solving Triangular Matrices

Triangular matrices can be solved very quickly using backward/forward substitution. Consider solving $U\mathbf{x} = \mathbf{y}$. The bottom row only has one one non-zero element $U_{nn}x_n = y_n$, so

$$x_n = y_n / U_{nn} \tag{22}$$

. The next row has two:

$$\begin{aligned} U_{n-1,n-1}x_{n-1} + U_{n,n}x_n &= y_{n-1} \\ x_{n-1} &= \left(y_{n-1} - U_{n,n}x_n\right)/U_{n-1,n-1} \end{aligned}$$

and we know x_n from the previous step. We continue on recursively. An upper triangular matrix is solved the identical way except starting at the top row instead of the bottom. Clearly the process breaks down if T has a zero element on the diagonal, but we expected that since this would cause $\texttt{det}\,(T) = 0$. A triangular matrix of size N will take $order N^2$ operations to solve in this way. Consider a particular row in the algorithm described above. The operations are

$$\begin{aligned} y_i &= \sum_{j=1}^{N} U_{ij} x_j \\ &= \sum_{j=i}^{N} U_{ij} x_j \\ &= U_{ii} x_i + \sum_{j=i+1}^{N} U_{ij} x_j \\ x_i &= \left(y_i - \sum_{j=i+1}^{N} U_{ij} x_j \right) / U_{ii} \end{aligned}$$

where the second line follows since the first elements U_{i1} to $U_{i,i-1}$ are all zero, and in last last recognise that we have already found the next elements we are solving for, so that for x_i we already have found $x_{i+1}, x_{i+2}, \ldots, x_N$. Computation of x_i involves the summation of $U_{ij} x_j$ for $j = i+1, \ldots, N$, which is $\mathcal{O}\,(N)$. This calculation is performed N times, once for each of the N rows, so in total we require $N\mathcal{O}\,(N) = \mathcal{O}\left(N^2\right)$ operations.

Creating Triangular Matrices with LU decompositions

The LU decomposition of a matrix A are the upper and lower triangular matrices U and L where $A = LU$. Conventionally, the main diagonal of L is $L_{ii} = 1$. This ensures uniqueness, since A has n^2 elements, while L has

$$\begin{aligned} \text{Elements} &= n + (n-1) + \ldots + 2 + 1 \\ &= \sum_{i=1}^{n} i \\ &= n(n+1)/2 \end{aligned}$$

non-zero elements, as does U. So we have $n(n+1)$ non-zero variables in total to set using the n^2 elements of A. To make this match set n of them, the diagonal of U, to be all one. When computing L and U analytically, we use sequences of transformations via elementary matrices. In practice LU are computed numerically in $\mathcal{O}\,(()\,N^3)$ operations. Here, I should add both numeric and analytic examples of LU decompositions. Note that many authors refer to decompositions where rows have been swapped (ie *pivoting*) as a *LU decomposition with pivoting*. If a LU decomposition is attempted directly without any pivoting the algorithm will fail anywhere with a zero element on the main diagonal. For example the algorithm will fail to decompose

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{23}$$

where it is obvious that exchanging the rows leads to an easy solution. If the pivot operations on A are collected in the matrix P, the the decomposition is $PA = LU$. Here P is *not*-unique, but once P is specified LU is.

The benefit of the LU decomposition $PA = LU$ is that we can use forward and backward substitution to very quickly solve linear systems. Consider the system $A\mathbf{x} = \mathbf{y}$. To solve, find P

which decomposes $PA = LU$, and solve $PA\mathbf{x} = LU\mathbf{x} = P\mathbf{y}$ in two steps. The trick is to let $U\mathbf{x} = \mathbf{v}$, and solve in two steps as follows

$$\begin{aligned} A\mathbf{x} &= \mathbf{y} \\ PA\mathbf{x} &= P\mathbf{y} \\ L(U\mathbf{x}) &= P\mathbf{y} \\ L\mathbf{v} &= P\mathbf{y} \end{aligned}$$

for $\mathbf{v}$, and then solve

$$U\mathbf{x} = \mathbf{v} \tag{24}$$

for $\mathbf{x}$, as required.

Eigenvalues and Eigenvectors

Definitions

Eigenvectors and *Eigenvalues* of a matrix A are fundamental quantities used regularly in matrix operations. The Eigenvector/value pair corresponding to a matrix A is the pair $\mathbf{v}$,λ with the property that

$$A\mathbf{v} = \lambda\mathbf{v} \tag{25}$$

The geometric interpretation of this is nice, but I've never found it particularly usefull. In two dimensions $\mathbf{v}$ is a line in the $x - y$ plane, and $A\mathbf{x}$ is a linear transformation that changes the direction $\mathbf{v}$ points, as well as its length. If $\mathbf{v}$ is an is an eigenvector of A, then the transformation $A\mathbf{v}$ only changes the length of the line $\mathbf{v}$, and not its direction. What is the practical importance of that? I'm not really sure. It is quite tedious to find eigenvalue/vector pairs analytically (include a few examples). Computing eigenvectors/values numerically is not as simple as the LU decomposition described earlier either. Since they are so critically important in all areas of applied maths/statistics, the computation of eigenvalues is a huge field of study. Note that if λ is an eigenvalue with a corresponding eigenvector $\mathbf{v}$, then any multiple of λ is also an eigenvalue. As such two vectors $\mathbf{u}$ and $\mathbf{v}$ with the above property, where $\mathbf{u} = c\mathbf{v}$ (ie one is just a multiple of the other) do not count as separate eigenvectors. To see this note that

$$\begin{aligned} A\mathbf{v} &= \lambda\mathbf{v} \\ A(c\mathbf{v}) &= (c\lambda)\mathbf{v} \\ A\mathbf{u} &= \lambda_u\mathbf{u} \end{aligned}$$

where $\mathbf{u} = c\mathbf{v}$ and $\lambda_u = c\lambda$.

To finding eigenvalue/vector pairs, note that if λ is an eigenvalue of A, then

$$\begin{aligned} A\mathbf{v} &= \lambda\mathbf{v} \\ (A - \lambda I)\mathbf{v} &= 0 \end{aligned}$$

which means the vector $\mathbf{v}$ is in the null-space of the matrix $A - \lambda I$. If such a vector exists, so that the null-space is non-empty, we know $(A - \lambda I)$ is a singular matrix. Since the purpose is to find such a $\mathbf{v}$, we need to find the conditions for which $(A - \lambda I)$ is singular. The most obvious is that it has a zero determinant; ie

$$\texttt{det}(A - \lambda I) = 0 \tag{26}$$

This results in a polynomial equation for λ, which may have multiple solutions, some of which might be repeated. This equation is refered to as the *characteristic polynomial* of A. Once we have all the solutions λ_i, we can work out the corresponding $\mathbf{v}_i$. These are the eigenvalue/vector pairs. This is quite a tedious process! I should put in an example or two here.

Since computing the λ_i reduces to working with determinants, the special structure of some matrices means that computing the λ_i is straight forward. The subtraction $A - \lambda I$ only modifies the diagonal of A, many matrices retain the special structure after the subtraction. Most importantly:

- **Diagonal Matrices** If D is diagonal, then $A - \lambda I$ is diagonal too. The determinant of a diagonal matrix is $\texttt{det}(D) = \prod D_{ii}$, so the characterist polynomial of $A - \lambda I$ is $\texttt{det}(A - \lambda I) = \prod (D_{ii} - \lambda) = 0$. As such the diagonal elements D_{ii} are themselves the eigenvalues of D.
- **Triangular Matrices** If T is triangular, then $A - \lambda I$ is triangular too. The determinant of a triangular matrix is $\texttt{det}(T) = \prod T_{ii}$, so the characterist polynomial of $A - \lambda I$ is $\texttt{det}(A - \lambda I) = \prod (T_{ii} - \lambda) = 0$. As such the diagonal elements T_{ii} are themselves the eigenvalues of T.

Eigenvector/value pairs give a great deal of information about the matrix they were computed from. Let $(\lambda_i, \mathbf{v}_i)$ be the eigenvalue/vector pairs of the matrix A.

- If any $\lambda_i = 0$, then A is singular. Remember that if A is non-singular, then it has an empty null-space. That is, there can be no non-trivial vector $\mathbf{v}$ where $A\mathbf{v} = 0$.
  However, if $\lambda = 0$ is an eigenvalue, and $\mathbf{v}$ its corresponding eigenvector, then $A\mathbf{v} = \lambda\mathbf{v} = 0$.
  This means the null-space of A is not empty (ie the eigenvector $\mathbf{v}$ is in it), so A is singular.
- An $(n \times n)$ matrix A has exactly n eigenvalues. Some may be repeated, and some may be complex.
- Every eigenvalue has at least one *eigenvector*.
- If $\lambda_1 \neq \lambda_2$ then their corresponding eigenvectors are linearly independant.
- If A has an eigenvalue with multiplicity m, it has at least one eigenvector, but may have less than m eigenvectors. This means that A may have n eigenvectors which are all linearly independant, but it may have less if any eigenvalues were repeated.
- If $(\lambda, \mathbf{v})$ are an evalue/vector pair from a non-singular A, then $(\lambda, \mathbf{v})$ are an evalue/vector pair from A^{-1}.

Interestingly, the eigenvectors/values of symmetric matrices have a number of additional useful properties.

- If S is symmetric, then S has a full set of n eigenvectors.
- If S is symmetric, each of its n eigvvalues are real.
- If S is symmetric, then all n of these eigenvectors are orthogonal.
- If S is symmetric, then its eigenvectors v_i and $\mathbf{v}_j$ are othogonal if the corresponding eigenvalues are not repeated (ie $\lambda_i \neq \lambda_j$).

Diagonalisation

A particularly useful application of eigenvalues and eigenvectors is the following decomposition. The matrix A is diagonaliseable if A can be written

$$A = VBV^{-1} \tag{27}$$

where

$$\begin{aligned} V &\rightarrow \text{columns are e-vectors of } A \\ B &\rightarrow \text{diagonal matrix where } B_{ii} \text{ are e-values of } A \end{aligned}$$

Note that by writting $A = VBV^{-1} \rightarrow AV = BV$, we see the above link between diagonalisation and eigenvalue/vector pairs

$$\begin{aligned} A &= VBV^{-1} \\ AV &= VB \\ \left(A\mathbf{v}_1,\, A\mathbf{v}_2,\, \ldots,\, A\mathbf{v}_n\right) &= \left(B_1\mathbf{v}_1,\, B_2\mathbf{v}_2,\, \ldots,\, B_n\mathbf{v}_n\right) \end{aligned}$$

and if we equate each column on the left and right hand sides, we have

$$A\mathbf{v}_i = B_i\mathbf{v}_i \tag{28}$$

ie $\mathbf{v}_i$ and B_i and are eigenvector/value pair. For A to be diagonalisable we require V^{-1} to exist - since this is a matrix whose columns are the evectors of A, we require all n eigenvectors to be linearly independant.

Extra Results for SPD Matrices

The additional eigenvector/value results for symmetric matrices also provide additional properties regarding diagonalisation. If S is symmetric, we are guaranteed to have n eigenvalues, and its n eigenvectors are all orthogonal. As such, if S is symmetric then it is diagonalisable, ie $S = VBV^{-1}$, and since the columns of V are orthogonal, V is an orthogonal matrix $V = Q$ and $Q^{-1} = Q^T$. The diagonal form of a symmetric matrix is useally written

$$S = QBQ^T \tag{29}$$

If all the eigenvalues of S are positive, then S is SPD. This converse is also true - if S is SPD then its eigenvalues are all positive. Since all $\lambda_i > 0$ we are guaranteed that S is non-singular, and that S^{-1} is also SPD. This makes sense - when $\lambda_i, \mathbf{v}_i$ are the eigenvalue/vector pairs of S, then $1/\lambda_i, \mathbf{v}$ are the eigenvalue/vector pairs of S^{-1}. Since $\lambda_i > 0$ then $1/\lambda_i$ exists and is positive. So S^{-1} is symmetric, and all its eigenvalues are positive - it is SPD!

Finally, SPD matrices can be factorised in a particularly efficient way - the *Cholesky decomposition*. The Cholesky decomposition is another very common matrix decomposition applicable only to SPD matrices. In practice, the Choleksy algorithm determines whether a given matrix is in fact SPD. The decomposition of a SPD matrix S is

$$S = U^TU \tag{30}$$

where U is upper triangular. If S is symmetric and such a decomposition exists, then it must be SPD, since

$$\begin{aligned} \mathbf{x}^TS\mathbf{x} &= \mathbf{x}^TU^TU\mathbf{x} \\ &= (U\mathbf{x})^T U\mathbf{x} \\ &= |U\mathbf{x}|^2 > 0 \end{aligned}$$

As with our discussion of LU decomposition, a Cholesky decomposition allows fast solutions to

systems $A\mathbf{x} = \mathbf{y}$ via forward and back subsitutions, since U is triangular. The particular form of the Cholesky decomposition means the *Cholesky factor* U is often described as the square root of a matrix, which we will see soon in the section on random variables and simulation.

Multivariate Random Variables

Basics

Consider a set $\{x_1, x_2, \ldots, x_m\}$ of random variables. The moments we will focus on are mean, variance, covariance, and correlation which are defined

$$\begin{aligned}
\text{mean: } \mu_i &= \mathbb{E}(x_i) \\
\text{variance: } \sigma_i^2 &= \mathbb{E}\left[(x_i - \mu_i)^2\right] \\
\text{covariance: } \sigma_{ij}^2 &= \mathbb{E}\left[(x_i - \mu_i)(x_j - \mu_j)\right] \\
\text{correlation: } \rho_{ij} &= \frac{\sigma_{ij}^2}{\sigma_i \sigma_j}
\end{aligned}$$

Both the covariances and correlations can be collected into $(m \times m)$ matrices Σ and Ω where

$$\begin{aligned}
\Sigma_{ij} &= \sigma_{ij} \\
\Omega_{ij} &= \rho_{ij}
\end{aligned}$$

The matrix Σ is built by taking the expectation of

$$\Sigma = \mathbb{E}\left[(\mathbf{x} - \mu)^T (\mathbf{x} - \mu)\right] \tag{31}$$

since each element $\Sigma_{ij} = \sigma_{ij}$ involves the product $(x_i - \mu_i)(x_j - \mu_j)$. The correlation matrix is easily computeable from Σ, by noting that it just requires scaling by each pairwise combination of standard deviations σ_i and σ_j. Using the diagonal matrix D, where $D_{ii} = \sigma_i$ we can write

$$D\Omega D = \Sigma \tag{32}$$

where the first D multiplies each *row* of correlations by σ_i, and the second post multiplication multiplies each *column* by σ_j. More commonly we write

$$\Omega = D^{-1}\Sigma D^{-1} \tag{33}$$

since usually the variances Σ are computed first.

The sample estimates of these quantities can be efficiently computed using linear algebra. Let ι be a $(n \times 1)$ vector of ones, and collect our n observation of each $\mathbf{x}_i$ into a matrix X, where $\mathsf{c}(X)_i$ has the n observations of $\mathbf{x}_i$, then

$$\begin{aligned}
\hat{\mu} &= \iota^T \mathbf{x}/n \\
\hat{\sigma}_i^2 &= (\mathbf{x}_i - \hat{\mu}_i)^T (\mathbf{x}_i - \hat{\mu}_i) / (n-1) \\
\hat{\Sigma} &= (X - \hat{\boldsymbol{\mu}})^T (X - \hat{\boldsymbol{\mu}}) / (n-1)
\end{aligned}$$

Writing these formulas using linear algebra may seem a little unnecessary, but they are important for two reasons:

- Many analytic results in statistics and ML follow from these representations
- Linear algebra packages (eg Intel MKL) are highly optimised to compute matrix-vector operations

It is common to ompuate the covariance of random variables which themselves are combinations of the variables $\{x_1, x_2, \ldots, x_m\}$. Again collect the x_i into a vector $\mathbf{x}$, and define $z_1 = \mathbf{a}^T\mathbf{x}$ and $z_2 = \mathbf{b}^T\mathbf{x}$. For simplicity assume that $\mu_i = 0$ and let's compute $\mathbb{C}(z_1, z_2)$, as follows:

$$\begin{aligned}
\mathbb{C}(z_1, z_2) &= \mathbb{C}\left[\left(\sum a_i x_i\right), \left(\sum b_i x_i\right)\right] \\
&= \mathbb{E}\left(\sum_i \sum_j a_i b_j x_i x_j\right) \\
&= \sum_i \sum_j a_i b_j \mathbb{E}\left(x_i x_j\right) \\
&= \sum_i \sum_j a_i b_j \Sigma_{ij}
\end{aligned}$$

We can build this quadratic-form like sum using $\mathbb{E}(\mathbf{x}\mathbf{x}^T)$ to get the required cross-products so that

$$\begin{aligned}
\mathbb{C}(z_1, z_2) &= \mathbf{a}^T \mathbb{E}(\mathbf{x}\mathbf{x}^T)\mathbf{b} \\
&= \mathbf{a}^T \Sigma \mathbf{b}
\end{aligned}$$

More generally, consider the random variables $\{y_1, y_2, ..., y_m\}$ created from linear combinations of the x_i. Collect the rvs into $\mathbf{x}$ and $\mathbf{y}$ and after setting

$$\mathbf{y} = A\mathbf{x} \tag{34}$$

the objective is to compute Σ_y using Σ_x, as we did above in the case of two linear combinations z_1 and z_2 for $\mathbf{y}$

$$\begin{aligned}
\Sigma_y &= \mathbb{E}\left[\left(\mathbf{y} - \boldsymbol{\mu}_y\right)\left(\mathbf{y} - \boldsymbol{\mu}_y\right)^T\right] \\
&= \mathbb{E}\left[M(\mathbf{x} - \boldsymbol{\mu})\left(M(\mathbf{x} - \boldsymbol{\mu})\right)^T\right] \\
&= \mathbb{E}\left[M(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T M^T\right] \\
&= M\,\mathbb{E}\left[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T\right] M^T \\
&= M\Sigma_x M^T
\end{aligned}$$

This property is critical - it is the recipe for generate one set of random variables from another. If

$$\mathbf{x} \sim (\boldsymbol{\mu}_x, \Sigma_x) \tag{35}$$

then $\mathbf{y} = \mathbf{a} + M\mathbf{x}$ has distribution

$$\mathbf{y} \sim (\mathbf{a} + \boldsymbol{\mu}_x, M\Sigma_x M^T) \tag{36}$$

More on Correlation Matrices

As shown above, correlation matrices provide the recipe for transorming one set of random variables into another. It is critical to note that the $(n \times n)$ correlation matrix Σ is SPD. First, it is clearly symmetric, since it is computed from the product $\mathbf{x}\mathbf{x}^T$. To show it is positive definite, we

are required to show $\mathbf{x}^T\Sigma\mathbf{x} > 0$. Return to the previous computation

$$\mathbb{C}\left[\left(\sum a_i x_i\right), \left(\sum b_i x_i\right)\right] = \mathbf{a}^T\Sigma\mathbf{b} \tag{37}$$

and set $a_i = b_i$. Then the covariance calculation is a variance, ie

$$\begin{aligned}\mathbb{C}\left[\left(\sum a_i x_i\right), \left(\sum a_i x_i\right)\right] &= \mathbb{V}\left(\sum a_i x_i\right) \\ &= \mathbf{a}^T\Sigma\mathbf{a} \\ &> 0\end{aligned}$$

Since a variance is clearly non-negative. So we have shown that the covariance matrix Σ is SPD. From here, many properties follow. Using a number of these properties, we can show that Σ has a cholesky decomposition, and infer that the Cholesky factor provides the means to generate multivariate random variables with a certain required covariance matrix. To see this, diagonalise Σ - we are guaranteed to be able to do this since Σ is symmetric ie

$$\Sigma = VBV^{-1} \tag{38}$$

where B is diagonal with B_{ii} the eigenvalues of Σ, and $\mathsf{c}\left(V\right)_i$ the corresponding eigenvectors. We are guaranteed the all $B_{ii} > 0$, so we can define A to be another symmetric matrix with $A_{ii} = \sqrt{B_{ii}}$. Finally, note that since Σ is symmetric, its eigenvalues are orthogonal - so write $V = Q$ to make the orthogonality of the eigenvectors making up the columns of V more clear. Since $Q^{-1} = Q^T$, we can write

$$\begin{aligned}\Sigma &= VBV^{-1} \\ &= QAAQ^T \\ &= QA\left(QA^T\right)^T \\ &= QA\left(QA\right)^T \\ &= UU^T\end{aligned}$$

where $U = QB^*$.