13 Bases

Motivation. **Digital images** are composed of discrete pixels. For a grey-scale picture, every pixel is represented by a one-byte number, from 0 (for black) to 255 (for white). In a colour picture, every pixel is represented by three one-byte numbers, one each for red, green, and blue. Here is a 20×20 pixel colour image, greatly magnified:

This snippet was taken from a 1368×855 pixel colour image. To store all its pixels we require $1368 \times 855 \times 3 = 3,508,920$ bytes, or approximately 3.5MB. Storing the image in the `bmp` ("bitmap") format takes this amount of storage space.

If this image is transformed into the `jpeg` format, then only 600KB are required, a nearly six-fold saving. The translation between the `bmp` and the `jpeg` representation is pure linear algebra and in this chapter we will find out how it works.

To start with, a digital image (from now on let's always think of a grey-scale one) is just a very long tuple of natural numbers, in our example it is an element of $1368 \times 855 = 1,169,640$-dimensional "space". So where in the previous two chapters I have encouraged you to think of tuples as either the coordinates of points, or as the components of a straight-line movement, now we are interpreting them as digital images. This is no big deal but perhaps we should remember that not every tuple can be interpreted in this way, for example, negative entries are meaningless as there is no such thing as "negative luminescence".

13.1 Generating an algebra of vectors

We already touched upon this topic in Section 11.6 but let us be more general now. If we are given a collection $\vec{v}_1, \ldots, \vec{v}_n$ of vectors, then we can look at the algebra of vectors generated by them. The generation is done by forming all **linear combinations** of the given vectors, that is, we compute

$$a_1 \cdot \vec{v}_1 + \ldots + a_n \cdot \vec{v}_n$$

for all choices of the scalars $a_1, \ldots, a_n$. We will have to compute a lot of these expressions in this chapter, so let's introduce a more compact notation for them:

$$\sum_{i=1}^{n} a_i \cdot \vec{v}_i \;=\; a_1 \cdot \vec{v}_1 + \ldots + a_n \cdot \vec{v}_n$$

To get some practice in this new notation, let's check again that the collection of all linear combinations (also known as the **span** of the vectors $\vec{v}_1, \ldots, \vec{v}_n$) is itself an algebra of vectors. That's because linear combinations can be added:

142

and the result is itself a linear combination. Likewise, a linear combination can be multiplied with a scalar:

143

and again we get a linear combination of the original n vectors $\vec{v}_1, \ldots, \vec{v}_n$. Combining addition and scalar multiplication, we can say that a linear combination of linear combinations is again a linear combination of the original vectors.

13.2 Linear independence and bases

Now imagine that we have some algebra of vectors in mind and we are seeking to generate it from a small number of "basic" vectors. That's exactly what we did in Section 11.3, when we represented all the movements along a straight line as multiples of a basic vector $\vec{v}$. It makes sense that we should like to employ as few vectors as possible for the generation process. Therefore, the following properties should hold about our collection $\vec{v}_1, \ldots, \vec{v}_n$:

- the null vector $\vec{0}$ does not occur in the list $\vec{v}_1, \ldots, \vec{v}_n$ (because it doesn't generate anything);
- none of the v_i is a linear combination of the remaining $n-1$ vectors (because that would be a redundancy).

If these two conditions are satisfied, then we say that the collection $\vec{v}_1, \ldots, \vec{v}_n$ is **linearly independent**. We can express the two conditions very neatly in a single criterion:

Theorem 8 *A collection $\vec{v}_1, \ldots, \vec{v}_n$ of vectors is linearly independent exactly if*

$$\sum_{i=1}^{n} a_i \cdot \vec{v}_i = \vec{0} \quad \textit{happens only in case} \quad a_1 = a_2 = \ldots = a_n = 0 \,.$$

If every element of an algebra V of vectors can be written as a linear combination of the vectors $\vec{v}_1, \ldots, \vec{v}_n$ and if the v_i are linearly independent, then we call $\vec{v}_1, \ldots, \vec{v}_n$ a **basis** of V.

Time for an example. Consider the algebra of all four-tuples of (rational) numbers (interpreted, if you wish, as the movements of four-dimensional space, or as 2×2 digital images). The following is the **standard basis** for this algebra:

144

Linear independence is easy to check for these four vectors using the criterion of Theorem 8:

145

How about the following four vectors?

$$\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}, \quad \begin{pmatrix} 1 \\ 1 \\ -1 \\ -1 \end{pmatrix}, \quad \begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \end{pmatrix}, \quad \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \end{pmatrix}$$

This is where Gaussian elimination can help. The equation $\sum_{i=1}^{4} a_i \cdot \vec{v}_i = \vec{0}$ amounts to four linear equations in the four unknowns a_1, a_2, a_3 and a_4. Since the right-hand side of each equation equals zero, we always have $a_1 = a_2 = a_3 = a_4 = 0$ as one solution, so the system can't be contradictory. If there is also a non-zero solution, then Gaussian elimination will find it:

146

We get a perfect echelon form which shows that there is only one solution; that solution is $a_1 = a_2 = a_3 = a_4 = 0$. (In a moment we will have an even better criterion for the linear independence of these four vectors which will allow us to avoid Gaussian elimination.)

13.3 Coordinates

Whenever we have a basis $\vec{v}_1, \ldots, \vec{v}_n$ for an algebra of vectors, any given vector $\vec{w}$ can be written as a linear combination of the $\vec{v}_i$. That's because the basis generates every vector. So if

$$\vec{w} = \sum_{i=1}^{n} a_i \cdot \vec{v}_i$$

then we can call the numbers $a_1, a_2, \ldots, a_n$ the **coordinates** of $\vec{w}$ with respect to the basis $\vec{v}_1, \ldots, \vec{v}_n$. Now you might wonder whether the coordinates are uniquely determined, or whether it is possible that $\vec{w}$ has two different representations:

$$\vec{w} = \sum_{i=1}^{n} a_i \cdot \vec{v}_i = \sum_{i=1}^{n} b_i \cdot \vec{v}_i$$

The first option is correct and the second option cannot happen. Here is the argument:

147

So once we have a basis, the coordinates of each vector are uniquely determined, and in fact, we can then think of the algebra as an algebra of n-tuples. This is exactly what we have been doing in Section 11 when we discussed movements in the plane and in 3D space with respect to a fixed basis (which at the time we called a "coordinate system").

13.4 Dimension of an algebra of vectors

If we have found a basis $\vec{v}_1, \ldots, \vec{v}_n$ for an algebra of vectors, is it possible that there is another one with fewer elements? That is not the case. The number of elements in a basis is an intrinsic feature of the algebra; it is called its **dimension**. Let us state this formally:

Theorem 9 *Any two bases of an algebra of vectors have the same number of elements.*

This may be intuitive but a proof is not so easy to come by. We give one in the case that the dimension is finite. So assume that both $\vec{v}_1, \ldots, \vec{v}_n$ and $\vec{w}_1, \ldots, \vec{w}_m$ are a basis for an algebra and our aim is to show that $n = m$. The crucial proof idea is contained in the following:

Lemma 10 *Let both $\vec{v}_1, \ldots, \vec{v}_n$ and $\vec{w}_1, \ldots, \vec{w}_m$ be a basis for an algebra of vectors. Let $\vec{v}_k$ be a member of the first basis. Then there exists a member $\vec{w}_p$ of the second basis that can replace $\vec{v}_k$ in the sense that $\vec{v}_1, \ldots, \vec{v}_{k-1}, \vec{w}_p, \vec{v}_{k+1}, \ldots, \vec{v}_n$ is also a basis.*

With this powerful lemma we can now prove our theorem: Given two bases, we can one-by-one replace all the $\vec{v}$'s with a $\vec{w}$. Doing so, we will never use the same $\vec{w}$ twice (or the result would not be a basis) and it follows that we must have started with at least as many $\vec{w}$'s as there were $\vec{v}$'s. Doing the same in the opposite direction, we conclude that there must have been at least as many $\vec{v}$'s as there were $\vec{w}$'s. These two statements can only be true if $n = m$, that is, both bases had the same number of elements.

13.5 All bases are not equal

So now we know that any two bases for the same algebra have the same number of basis vectors. However, depending on what we are trying to achieve, one basis may be a lot more convenient than another one. This is exactly the message that `jpeg`-compression is telling us. Let's start with an example of a tiny 2×2 digital (grey-scale) image, written as a four-tuple. In the standard basis, we can read off the coordinates immediately:

$$\begin{pmatrix} 10 \\ 10 \\ 11 \\ 12 \end{pmatrix} = 10 \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} + 10 \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} + 11 \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} + 12 \cdot \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

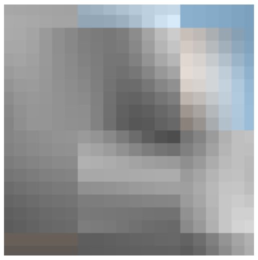
but let's look at the same "image" represented in the basis given in Box 145:

148

We see that the first coordinate is much bigger than the other three and in fact, if we "forgot" about the last two, we would still get a reasonable approximation to the original image:

149

This example already illustrates the main idea behind `jpeg`-compression: Represent the image vector via its coordinates with respect to an alternative basis, *better suited to images*, and suppress all coordinates that are smaller than some threshold. In `jpeg` the transformation is done for each block of 8×8 pixels separately, something that is very visible if we set a very high compression factor:



(This is the same image fragment as displayed at the beginning of the chapter, but now highly compressed.)

13.6 Orthogonal and orthonormal bases

Wait a moment, you will say, where do we get the coordinates of the image vector with respect to the alternative basis from? Especially, since `jpeg`-compression works with 64-dimensional tuples. The answer comes from orthogonality as defined by the inner product.

So assume that we have a basis $\vec{v}_1, \dots, \vec{v}_n$ where any two vectors are orthogonal to each other, that is, we have

$$\langle \vec{v}_i, \vec{v}_j \rangle = 0 \quad \text{whenever} \quad i \neq j$$

Then I claim that the coordinates a_k of a given vector $\vec{w}$ with respect to the $\vec{v}$'s are simply computed as

$$a_k = \frac{\langle \vec{w}, \vec{v}_k \rangle}{\langle \vec{v}_k, \vec{v}_k \rangle},$$

which is the formula we saw a lot in the previous chapter already. Why is this the case? We assumed that the $\vec{v}$'s form a basis and hence we know that

$$\vec{w} = \sum_{i=1}^{n} a_i \cdot \vec{v}_i$$

for *some* scalars $a_1, \dots, a_n$. To show that the scalars have the form given above, we simply evaluate an inner product:

150

so we get the formula we claimed by solving for a_k.

If the computations are done on a computer, then we can make sure that each basis vector has length 1, so the division by $\langle \vec{v}_k, \vec{v}_k \rangle$ is not necessary when computing the coordinates. In that case one speaks of an **orthonormal basis**. If computations are done on paper, then it's better to avoid the surds[15] and stay with the merely *orthogonal basis*. You can check that the basis given in Box 148 is orthogonal and that the coordinates given in Box 149 can be computed by the formula above.

However, we have been a bit nonchalant in our computations above. For the moment, there is nothing to tell us that the inner product of a vector with itself is different from zero, and so the term $\frac{\langle \vec{w}, \vec{v}_k \rangle}{\langle \vec{v}_k, \vec{v}_k \rangle}$ may not exist. For the algebras of vectors considered in the last two chapters (i.e., tuples of rational numbers), this problem does not exist because here the following is true:

Positive definiteness of the inner product

$$\begin{array}{rcl} \langle \vec{v}, \vec{v} \rangle & \geq & 0 \\ \langle \vec{v}, \vec{v} \rangle = 0 & \Longrightarrow & \vec{v} = \vec{0} \end{array}$$

Over GF(2) we still have the inner product defined as in the last chapter, but it is not positive definite as the following example shows:

151

$$\left\langle \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\rangle =$$

Indeed, there is no positive definite inner product for vectors over GF(2) and orthogonal bases may not exist.

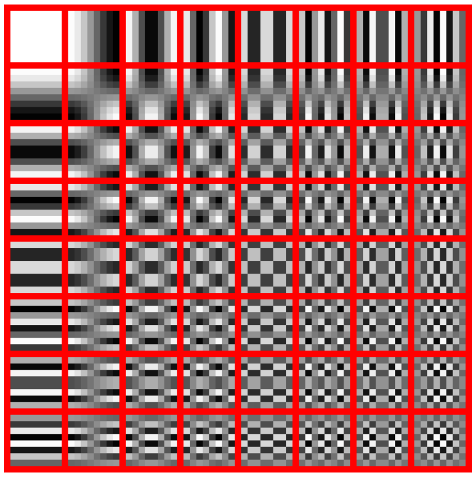
Creating an orthogonal basis. If we have a positive definite inner product then we can turn any basis into an orthogonal one. The idea is simple: we use the projection property (Section 12.4) to remove from each basis vector that part which is not orthogonal to the others. More formally, let $\vec{v}_1, \ldots, \vec{v}_n$ be a basis of our algebra. We define a new basis $\vec{w}_1, \ldots, \vec{w}_m$ whose members are orthogonal to each other:

$$\begin{array}{lcl} \vec{w}_1 & \overset{\text{def}}{=\!=} & \vec{v}_1 \\ \vec{w}_2 & \overset{\text{def}}{=\!=} & \vec{v}_2 - \frac{\langle \vec{v}_2, \vec{w}_1 \rangle}{\langle \vec{w}_1, \vec{w}_1 \rangle} \cdot \vec{w}_1 \\ & \vdots & \\ \vec{w}_n & \overset{\text{def}}{=\!=} & \vec{v}_n - \sum_{i=1}^{n-1} \frac{\langle \vec{v}_n, \vec{w}_i \rangle}{\langle \vec{w}_i, \vec{w}_i \rangle} \cdot \vec{w}_i \end{array}$$

It's a good exercise to show that each $\vec{w}_k$ is orthogonal to all the previous ones, i.e., to $\vec{w}_1, \ldots, \vec{w}_{k-1}$.

On a computer we would now divide each $\vec{w}_k$ by its length to obtain an orthonormal basis.

The 64 basis vectors used in `jpeg`-compression form an orthonormal basis. Here is a visual representation of them all (taken from Wikipedia):



[15] Expressions containing roots

To avoid any misunderstandings, the power of `jpeg`-compression does *not* come from the fact that these vectors form an orthonormal basis; this only helps in the calculations. After all, the standard basis is already orthonormal. It is rather that the pixels of real-life images are not random: If several neighbouring ones have similar values then it is likely that many more have similar values (e.g., a patch of blue sky), and if several neighbouring pixels show some oscillation then it is likely that the oscillation continues further (e.g., the leaves of a tree, or waves on a lake). The orthonormal basis of `jpeg`-compression picks up on these "oscillations" which is why it is also said that it represents the image in the "frequency domain". What we have here is an example of a **discrete Fourier transform**. There are many variations on this theme with multitude applications in computer science, but the underlying principle is always the same: representation in an alternate (orthonormal) basis.

Exercises

1. Show that the five vectors $\begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$, $\begin{pmatrix} 2 \\ -1 \\ -1 \end{pmatrix}$, $\begin{pmatrix} 0 \\ -1 \\ 3 \end{pmatrix}$, $\begin{pmatrix} 3 \\ -2 \\ 0 \end{pmatrix}$, and $\begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix}$ are linearly dependent.
2. Select three of them which form a basis for the algebra of three-tuples.
3. Turn it into an orthogonal basis using the procedure of Section 13.5 above.
4. Compute the coordinates of the vector $\vec{u} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ with respect to your orthogonal basis.

Practical advice

In the exam I expect you to be able to

- test whether a given set of vectors (over any field) is linearly independent;
- complete a given set of linearly independent vectors to a basis;
- transform a given basis into an orthogonal one.