Pattern Analysis

A Deep Learning Approach

S. S. Chandra Ph.D

July 2023

Draft

Version: 0.31

"A mathematician, like a painter or a poet, is a maker of patterns. If his patterns are more permanent than theirs, it is because they are made with ideas."

Godfrey H. Hardy (1877-1947)

Abstract

Acknowledgements

Contents

List of Figures

List of Tables

To our families ...

Preface

Part I

Patterns

CHAPTER 1

Introduction

> "Nature is the source of all true knowledge. She has her own logic, her own laws, she has no effect without cause nor invention without necessity."
>
> Leonardo da Vinci (1452-1519)

Patterns in nature exist in many forms. We find them beautiful and pleasing, and in some ways we are designed to naturally sense them. From musical masterpieces to pretty flowers and snowflakes. We are easily able to identify them as pleasing and spot them in a scene full of objects without issue. We have an underlying intuitive understanding of patterns.

A few questions arise

1. What do we consider as a pattern?
2. How do we quantify a pattern?
3. What can be classed as having a pattern?

In this book, we will explore the fundamental meaning of patterns and how they can be recognised.

CHAPTER 2

Symmetry

> "So our problem is to explain where symmetry comes from. Why is nature so nearly symmetrical? No one has any idea why."
>
> Richard Feynman (1918-1988)

Have you ever asked why flowers are beautiful? What gives them their beauty. No doubt that colour plays a role, but there is one aspect that has the strongest effect. It is because they are symmetrical. In fact, the reason why flowers are symmetrical is because bees like symmetry, while they don't even see any colour! Bees are one of the most wonderful of species in that they employ symmetry is nearly everything they do. In their honeycombs, the way they cover an entire area to find pollen and how they build their hives.

The key principle governing patterns in nature is symmetry. In this chapter, we will see how we can quantify symmetry scientifically and exhaustively. We will see that there is an optimal way to represent symmetry and mathematician Évariste Galois [1830] (1811-1832) invented it, just before he died in duel at the early age of twenty! We will begin by observing symmetry and the most intuitive way to do this is to visualise it in what we experience in a daily lives - Geometry.

2.1 Geometry

We can consider everything around us as geometry, objects moving in three dimensional (3D) space. For our purposes, we can consider the simpler two dimensional (2D) case and begin with the simplest object, a line (or more technically a line segment) as shown in figure 2.1 on the following page(a). The line exists even without a coordinate system, but we are accustomed to viewing it within such as system as shown in figure 2.1(b). A number of areas of science depend on the properties of this simple line, such as the theory of gravity, which we will explore later in this book.

For now consider the simple act of trying to measure the length c of this line. We could use Pythagoras' theorem $a^2 + b^2 = c^2$, but that depends on knowing the lengths a and

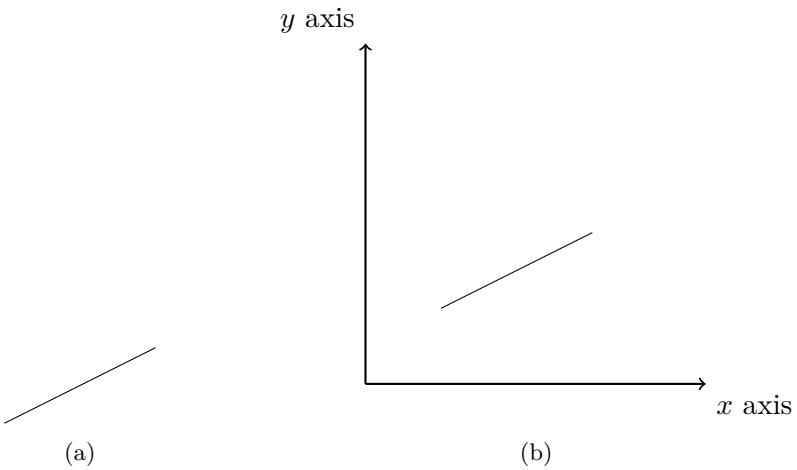


Figure 2.1: The simplest non-trivial geometric object, a line.

b, which in turn depends on the coordinate system we choose to define. However, there is no unique way to do this, as we can define the unit distance of the coordinate with many differing lengths. Consider that even the unit of measure we use called a metre is not absolute, it was in fact defined by a platinum-iridium bar of fixed length at a fixed temperature in the mid 20th century, and currently defined as the distance travelled by light in some fraction of a second. But even the speed of light varies depending on the medium it is travelling in, which is the same phenomenon that causes light to refract.

We can mitigate the differences between coordinate systems by determining the scaling between the various systems. For example, let us assume that there are three different coordinate systems and the line has integer coordinates, so that the length of the line is also an integer. This is called a Pythagorean triple, such as $a = 3$, $b = 4$ and $c = 5$. Let one of the other coordinate systems be scaled 2 times larger than your coordinate system and the other 3 times more. Relative to your point of view, where $a_1^2 + b_1^2 = c_1^2$, we can view the other coordinate systems as scaled versions

$$\begin{aligned} \frac{a_2^2}{2} + \frac{b_2^2}{2} &= \frac{c_2^2}{2} \\ \frac{a_3^2}{3} + \frac{b_3^2}{3} &= \frac{c_3^2}{3} \end{aligned}$$

This normalisation effectively allows you to convert between the different coordinate systems provided you know the scaling factors. Note that this scaling of integer coordinates also provides a way to introduce rational numbers through geometry rather than the number line, since the coefficients of a, b and c are integer ratios. These types of descriptions for lines and points, as well as precise definitions of parallel lines, were first constructed by Euclid [300BCE].

However, this method becomes very cumbersome if the scalings are different per dimension and when other considerations need to be taken into account, such as motion. The elegant solution to these considerations is to use coordinate free systems for representing

geometry and we will cover this later via tensors. But for the moment, we have a way of ensuring that all observers can agree on the length of the line regardless of their coordinate systems via scaling.

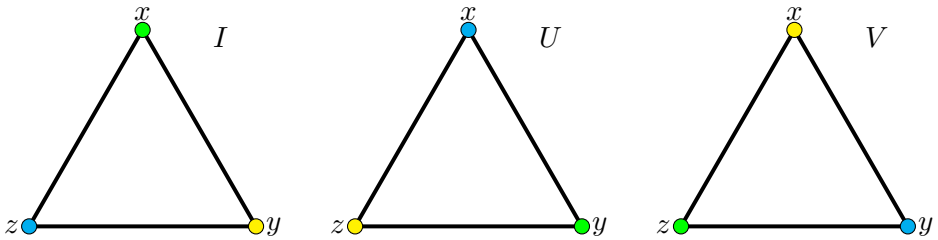
2.2 Invariance

Having agreed on the length of the line, the multiple observers can experiment with the line, apply operations and measure the outcomes. If we assume that the coordinates of the end points of the line are defined as (1,1) and (3,2), we can observe the following about the line:

1. moving the line anywhere (e.g. adding (4,4) to both end points) does not alter the length of the line
2. rotating the line by 180 degrees does not alter the line at all

We can say that the line we have defined is *invariant* under those operations. Those operations leave the line unchanged and expose to us the symmetry of the line, namely the line is symmetrical along the direction of the line.

To provide a more concrete example, consider a triangle constructed from three of our lines, whose vertices are coloured green, yellow and blue. Given the fixed positions x, y and z, we can explore rotations around the centre of the triangle as shown as figure 2.2. The



(a) (b) (c)

Figure 2.2: A triangle with coloured vertices showing rotational symmetry. (b) and (c) shows (clockwise and anti-clockwise) rotations of the triangle around its centre with respect to the fixed points x, y and z.

rotation of the triangle as shown in figure 2.2 leaves the triangle unchanged but the vertices permuted. This invariance to a rotation of 60 degrees reveals a symmetry of the object. We have uncovered a clue that applying some operations may leave our geometry objects unchanged and therefore reveal the symmetries involved. What operations do we mean?

2.3 Transformations

The experiments above were limited to translation (movement) and rotation. These operations are examples of what we call in mathematics as transformations. They are a mapping

of coordinates to new coordinates just like applying a function to a variable in algebra. Felix Klein [1893] constructed a generalised form of geometry that superseded that constructed by Euclid [300BCE]. He defined symmetry in geometry as those properties of geometric objects that remain invariant to transformations.

For example, let us go back to the triangle shown in figure 2.2 on the previous page. We saw that the triangle has rotational symmetry around the centre. We can also observe that the triangle has reflective symmetry around a line drawn perpendicular to each side through each vertex (see dotted lines in figure 2.2 on the preceding page). Each of these operations can be seen as a transformation of the triangle. For example, a clockwise rotation of the triangle will move the green vertex to the yellow vertex starting from x and ending up at y (from figure 2.2 on the previous page(a) to figure 2.2 on the preceding page(b)). Likewise, the reflections of the triangle along the perpendicular (dotted) lines with respect to the vertices are shown in figure 2.3. This gives us a total of six unique symmetries that

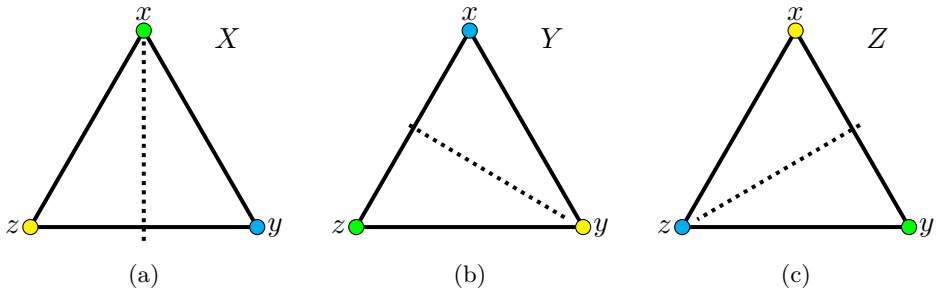


Figure 2.3: A triangle with coloured vertices showing reflective symmetry. (a)-(c) shows reflections of the triangle along the perpendicular (dotted) line with respect to the fixed points x, y and z.

leave the triangle unchanged after transformations, where we have included the identity transformation that does nothing. But does this cover *all* the possible transformations? What happens when we compose transformations together?

If we label the operations of reflective symmetry about x, y and z as X, Y, Z respectively and U, V as the rotations clockwise and anti-clockwise respectively (see figures 2.2 on the preceding page and 2.3), we can draw a table of the possible pairs of transformations and note the resulting configuration of the triangle obtained from them and label this accordingly. We can then observe any patterns that show up. This composition or 'multiplication' table, also known as a Cayley table is shown in table 2.1 on the facing page. Similarly, the reader is left to verify that table 2.2 on the next page is the Cayley table for a rectangle.

We can observe a startlingly fact that all possible composed pairs of transformations simply lead to the result of another known transformation. For example, two counter-clockwise rotations V, read from the table as the row V and the column V is equivalent to a single clockwise rotation U. Or that a reflection X and a reflection Y is equivalent to a rotation V.

	I	U	V	X	Y	Z
I	I	U	V	X	Y	Z
U	U	V	I	Y	Z	X
V	V	I	U	Z	X	Y
X	X	Z	Y	I	U	V
Y	Y	X	Z	V	I	U
Z	Z	Y	X	U	V	I

Table 2.1: The mapping of transformation pairs for the triangle, where X, Y and Z represent the rotations to those fixed points starting from X. The symbols I, U and V represent the identity (nothing happens), reflection and rotation respectively.

	I	R	H	V
I	I	R	H	V
R	R	I	V	H
H	H	V	I	R
V	V	H	R	I

Table 2.2: The mapping of transformation pairs for the rectangle, where R represents the rotation (either direction is equivalent) to those fixed points. The symbols I, H and V represent the identity (nothing happens), horizontal flip and vertical flips respectively.

In fact, the table we have constructed is exhaustive and let's us map any possible transformation of the object that leave the object unchanged. In other words, the composition $A \cdot B$ of two operations A and B is always in table 2.1. This is effectively a quantification of that objects *entire* symmetry as a single construct called a Group. This is the central idea of the theory of Galois Groups or simply Groups or in our case a symmetry Group.

2.4 Groups

The goal is to create a concrete abstract structure that can quantify all symmetries of an object, but also the relationships between all the possible symmetries. Moreover, the structure has to be as compact and minimal as possible as well.

Therefore, we need some way to collect all of the possible unique symmetries. In mathematics, the simplest structure to do this is called a set. However, a set does not have any way of associating relationships between its elements. Thus, we also need some way to do basic algebra between elements, so that we may write the relationships between the various symmetries. Such a structure was first constructed by Galois [1830] and it is called a Group.

1 Definition (Groups [Galois, 1830])
A Group G is composed of a set of elements and a composition or product operation $\cdot$, which abide by the following properties:

1. *The product result $C = A \cdot B$ is also part of the same set.*
2. *There is an identity element I in the set, so that $A \cdot I = A$.*
3. *There exists an inverse operation, so that $A \cdot A^{-1} = I$.*
4. *The product is associative, so that $A \cdot (B \cdot C) = (A \cdot B) \cdot C$.*

By placing the set and the composition operation together, we define a consistent algebra. The identity operation is important because it completes the algebra, making the construct consistent, by ensuring that inverse operation is possible among all elements. As a consequence, the identity symmetry must also always exist with the set. It is important to note that the product operation is used in a general sense and in no way related to the actual product or multiplication operation of numbers. It is purely a contextual operation between elements and can be anything. For example in our geometry examples, the elements of the group are the rotations and reflections possible, while the product is the composition of these elements. The entire 'space' of the Group can be quantified by the Cayley tables 2.1 on the previous page and 2.2 we have seen in our geometric examples.

In the next section, we will give examples of how scientists have used Groups to solve some very difficult, but extremely fundamental mysteries found in nature.

2.5 Conservation Laws

Noether [1918]

2.6 Applications

2.6.1 Standard Model of Particle Physics

2.6.2 Rubik's Cube

CHAPTER 5

Transform Domains

> "All mathematics is is a language that is well tuned, finely honed, to describe patterns; be it patterns in a star, which has five points that are regularly arranged, be it patterns in numbers like 2, 4, 6, 8, 10 that follow very regular progression."
>
> Brian Greene (1963-)

The fundamental idea of transforms is to change the way data is represented, i.e. transform the data into a (hopefully more) useful domain, whose properties can be exploited to:

1. make processing easier or computationally more efficient
2. improve the separation or clustering
3. reduce the problem

Ultimately, finding the right domain can make finding patterns in data more easier, even straight forward. The most basic way to change the underlying representation of your data or image is to change the coordinate system representing the data.

5.1 Change of Basis

A coordinate system is simply a choice of basis vectors to represent our data. The linear combination of these basis vectors allows us to represent the entire space of the coordinate system however we wish. Turns out, the choice of coordinate system is not unique and no one coordinate system is universal. This is the fundamental concept behind one of the most successful theories in science - Einstein's Theories of Special and General Relativity.

5.1.1 Basis Vectors

In linear algebra, we can write any vector space via a set of basis vectors $\{e_j\}$, where $j = 1, \ldots, n$ and n is the dimension of the space. The basis vectors are generally of length unity and orthogonal to each other. Orthogonality means the vectors are linearly independent and thus have no common components. We can understand this concept of Orthogonality by understanding the dot product of vectors.

5.1.2 Dot Product

The key idea behind coordinate system basis vectors is the concept of the dot (or inner) product. The dot product $\boldsymbol{a} \cdot \boldsymbol{b}$ between the vector $\boldsymbol{a}$ and vector $\boldsymbol{b}$ of dimension n can be written as

$$\boldsymbol{a} \cdot \boldsymbol{b} = \sum_{j=0}^{n-1} a_j b_j \tag{5.1}$$

If the vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ are orthonormal, then $\boldsymbol{a} \cdot \boldsymbol{b} = \delta_{ij}$, where δ_{ij} is the Dirac delta function defined as

$$\delta_{ij} = \begin{cases} 0, & i \neq j \\ 1, & i = j. \end{cases} \tag{5.2}$$

For basis vectors, this function simply states that the only vector that has a common component to a vector like e_j is the vector e_j itself.

Thus, we are not limited to conventional coordinate systems and we should in fact change to a coordinate system, i.e. to a new basis set, which is more 'convenient' in your scenario. If we 'transform' our coordinate system from one to another, we can change our representation. If we change our representation, we might find that our pattern analysis is simplified.

Consider the simple example of data that follows an exponential trend. We could fit an exponential curve or function to the data and that will work fine. However, an exponential is a non-linear function, computing it computationally expensive and it is difficult to plot or visualise because of the growth of values. But if we transform our data into the logarithmic domain, the exponential data becomes linear and the analysis of a line is much simpler!

The problem arises: what representation is 'best' to use and how do we find them? This is not an easy question, however there are special types of transforms that a generic tools that can be used for most types of data and this chapter will cover some of them. The first such transform is one based on physical systems and oscillations called the Fourier transform.

5.2 Fourier Transform

The Fourier transform (FT) is the change of basis to a harmonic representation of the data. In other words, we fit trigonometric functions, the so called sines and cosines, to our data to determine the harmonic representation or Fourier space of the data. This transformation is a natural process and consequence of complex numbers and traversing a unit circle.

5.2.1 Unit Circle

A complex number $z = x + iy$ with $i^2 = -1$ can also be written in polar coordinates as

$$z = re^{i\theta}. \tag{5.3}$$

This is Euler's formula and is shown in figure 5.1. Multiplying two complex numbers

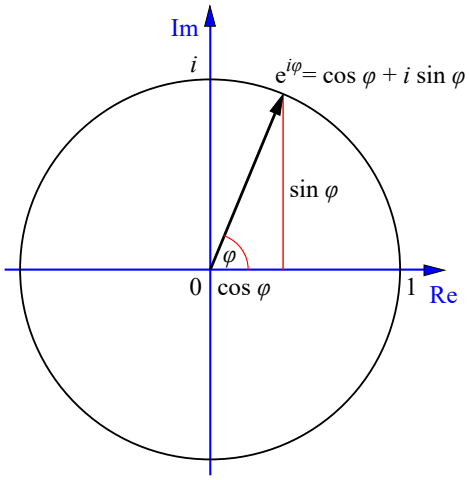


Figure 5.1: Euler's formula for the complex number $z = x + iy$ so that $e^{i\theta} = \cos\theta + i\sin\theta$ and $z = re^{i\theta}$.

($z_1 = r_1 e^{i\theta}$ and $z_2 = r_2 e^{i\phi}$) of unit length is therefore equivalent to rotating around the unit circle

$$\begin{aligned} z &= z_1 \cdot z_2 && (5.4)\\ &= r_1 e^{i\theta} \cdot r_2 e^{i\phi} && (5.5)\\ &= r_1 r_2 e^{i(\theta+\phi)} && (5.6)\\ &= e^{i(\theta+\phi)}. && (5.7) \end{aligned}$$

This is the same as adding the angles of the two vectors together and since the length is unity, this results in the vector z_1 rotating around the origin by an additional ϕ radians or vice versa for z_2.

5.2.2 Roots of Unity

The data that we usually deal with is discretised, i.e. we have a fixed number of samples. If this is of a physical system like a ball being thrown under the influence of Earth's gravity, it might represent a fixed number of samples of the ball position in space. However, the ball follows a perfect parabolic trajectory when ignoring air resistance, but our samples represent snapshots of this trajectory over time.

In discrete Fourier analysis, we study the rate that our data oscillates around the unit circle. Therefore, if we have N sample points, we usually divide the circle into N equal

parts. This division of the circle is algebraically known as the Nth root of unity $\omega = e^{2\pi i/N}$. You can show that by traversing around the unit circle with each Nth root of unity steps, one will trace out a sine or cosine curve depending on starting position. The rate at which you traverse the unit circle (perhaps by multiples of ω) will trace out sine curves of different frequencies.

5.2.3 Harmonics

The harmonics are then just traversals of the unit circles at different rates of the root of unit ω. Traversing the unit circle and tracing the resulting point over time will reveal of sine curve. If one traverses and traces out every point corresponding to a angle caused by the root of unit ϕ_ω starting from the point (1,0), we will get a sine curve with a wavelength of 1. Figure 5.2 shows the various harmonics generated by traversing the unit circle by every increasing rates of multiples of ω.

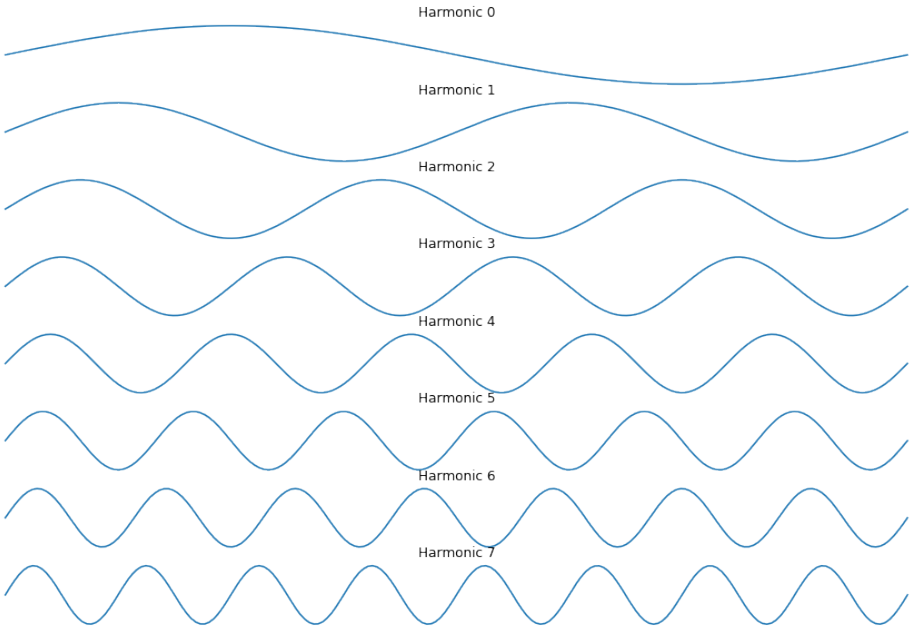


Figure 5.2: Each harmonic $n-1$ that results from traversing the unit circle at the rate ω^n.

To visualise this process for $N = 8$, consider the construction of a matrix with each row representing different rates of unit circle traversal, which is equivalent to different powers

of ω. We get the following Fourier matrix

$$\boldsymbol{F} = \begin{bmatrix} \omega^0 & \omega^0 & \omega^0 & \omega^0 & \omega^0 & \omega^0 & \omega^0 & \omega^0 \\ \omega^0 & \omega^1 & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\ \omega^0 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} & \omega^{14} \\ \omega^0 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} & \omega^{21} \\ \omega^0 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} & \omega^{28} \\ \omega^0 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} & \omega^{35} \\ \omega^0 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} & \omega^{42} \\ \omega^0 & \omega^7 & \omega^{14} & \omega^{21} & \omega^{28} & \omega^{35} & \omega^{42} & \omega^{49} \end{bmatrix} \tag{5.8}$$

Notice that the powers of ω is a different multiple in each row. The multiple is in fact the row index k. But why? This is the speed at which ω goes around the unit circle creating the harmonic at that frequency.

Let us examine each harmonic starting from row 1 noting that row 0 gives $\omega^0 = 1$ and this is just a constant, thus given us the sum of the signal (or DC offset). At row 1

$$\boldsymbol{F}_1 = \begin{bmatrix} \omega^0 & \omega^1 & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \end{bmatrix} \tag{5.9}$$

Using the fact that multiplication of complex exponentials results in the sum of their angles, the above can be done considering only the unit circle as $\theta = \pi/4$. This is because $N = 8$ divides the circle into 8 parts with every second octant making a quadrant that intersects with the imaginary and real axis, i.e. the values are either $1, i, -1, -i$ and back to 1. This process is shown in figure 5.3

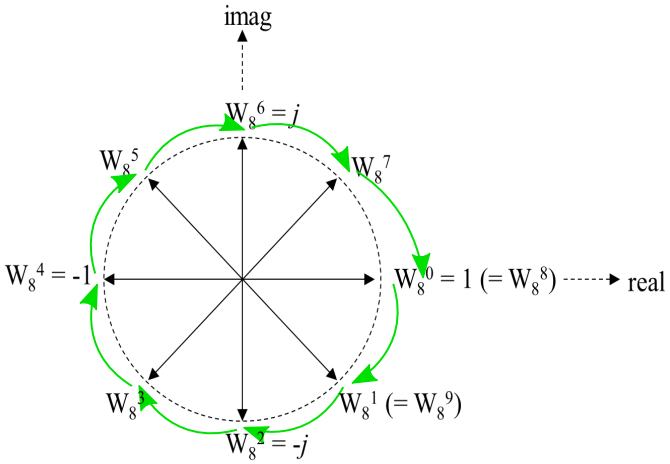


Figure 5.3: For $N = 8$ the circle is divided into 8 parts with every second octant making a quadrant that intersects with the imaginary and real axis, i.e. the values are either $1, i, -1, -i$ and back to 1. Sometimes the symbol j is used as a convention for the imaginary unit i.

Examining the second harmonic on row 2

$$\boldsymbol{F}_2 = \begin{bmatrix} \omega^0 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} & \omega^{14} \end{bmatrix} \tag{5.10}$$

Since $\omega = 0.707 - 0.707i$ for $N = 8$, the second row of $\boldsymbol{F}$ is

$$\begin{bmatrix} 1+0.i & 0-1.i & -1-0.i & 0+1.i & 1+0.i & 0-1.i & -1-0.i & 0+1.i \end{bmatrix}$$

Using the same multiplication property of complex exponentials, the above can be computed as rotation around the unit circle with $\theta = \pi/2$. This is twice the rate as for row 1 with each multiplication resulting with each value being the quadrants that intersects with the imaginary and real axis, i.e. the values are either $1, i, -1, -i$ and back to 1 twice. This is shown in figure 5.4.

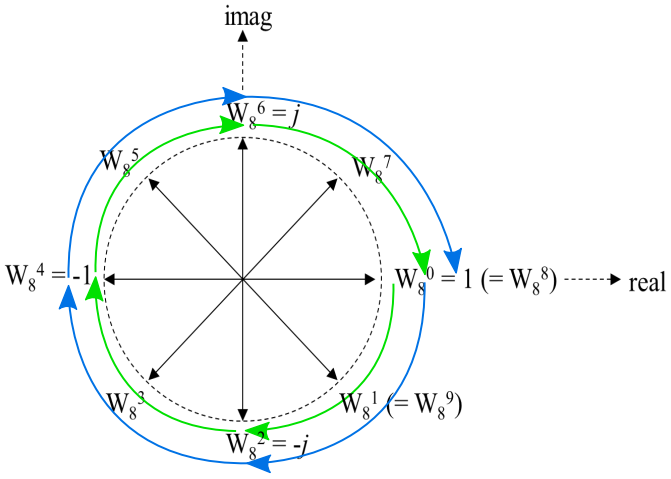


Figure 5.4: Traversing the unit circle at twice the rate as for row 1 shown in figure 5.3 on the previous page with each multiplication resulting with each value being the quadrants that intersects with the imaginary and real axis, i.e. the values are either $1, i, -1, -i$ and back to 1 twice.

Examining the second harmonic on row 3

$$\boldsymbol{F}_2 = \begin{bmatrix} \omega^0 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} & \omega^{21} \end{bmatrix} \tag{5.11}$$

Since $\omega = 0.707 - 0.707i$ for $N = 8$, the third row of $\boldsymbol{F}$ is

$$\begin{bmatrix} 1+0.i & -0.707-0.707i & 0+1.i & 0.707-0.707i & -1-0.i & 0.707+0.707i & 0-1.i & -0.707+0.707i \end{bmatrix}$$

The above can be computed as rotation around the unit circle three times the rate as for row 1 with each multiplication resulting with each value being every third octant. This is shown in figure 5.5 on the facing page.

The harmonics share two important properties that make them ideal as basis functions. The harmonics are both orthogonal to each other and of unit magnitude, i.e. they are orthonormal by construction, which are properties that can be easily verified by the reader. This allows harmonics to be used a basis functions of our transform and all we need to do is to compute the dot product with these harmonic functions.

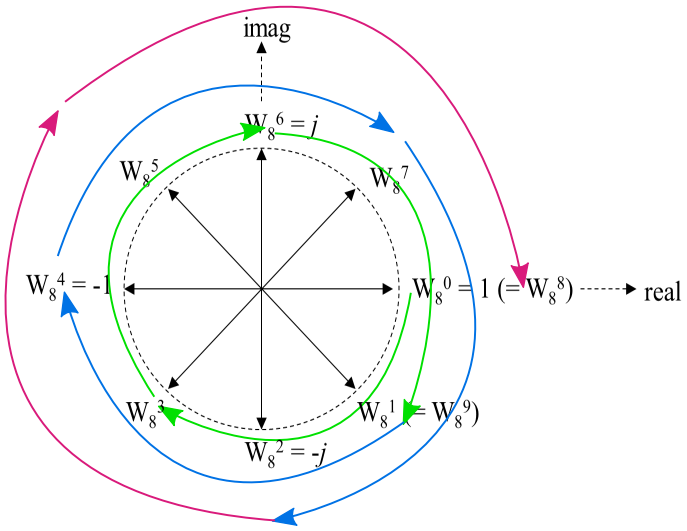


Figure 5.5: Traversing the unit circle at three times the rate as for row 1 shown in figure 5.3 on page 21 with each multiplication resulting with each value being every third octant.

5.2.4 Discrete Fourier Transform

The discrete Fourier transform (DFT) is the projection of our data (in vector form) onto the harmonic functions we described in the previous section. Since the root of unity ω is fixed and known for the size N. The harmonics of the Fourier matrix $\boldsymbol{F}$ can be predefined. Thus, we simply need to compute the inner product with our data/signal $\boldsymbol{x}$. This can be viewed as an inner product between two N-vectors, i.e. vectors of dimension N (in a linear algebra sense). For example, vectors representing 3D space would be 3-vectors. Therefore, the DFT $\boldsymbol{X} = X_k$ of the signal $\boldsymbol{x} = x_n$ can be defined as

$$X_k = \sum_{n=0}^{N-1} x_n \cdot \omega^{kn}, \tag{5.12}$$

where the Nth root of unity is defined as

$$\omega = e^{2\pi i/N}. \tag{5.13}$$

The inverse DFT is simply the traversal of the unit circle in the opposite direction

$$x_n = \sum_{k=0}^{N-1} X_k \cdot \omega^{-kn}, \tag{5.14}$$

It is usually more efficient to use the fast Fourier transform (FFT) implementation of the DFT in practice. This is a highly optimised algorithm for the DFT that uses the symmetry in the Fourier matrix $\boldsymbol{F}$ to reduce the complexity of the algorithm from $O(N^2)$ to $O(N \log_2 N)$.

The DFT can reveal hidden harmonic signatures, especially those caused by physical systems, such as sound or motion. In fact, the DFT is an ideal transform for signals or data originating from physical sources, since most physical systems have some oscillatory phenomena behind them.

5.3 Principal Component Analysis

Ideally, our data would be represented using the basis set that gives us the most "compact" representation. By compact we mean representing our data with as few basis vectors as possible. This compact representation can only be obtained by identifying the patterns in our data and one way to do this is to compute its principal components. This representation involving principal components is called principal component analysis (PCA). A form of PCA was first introduced by Pearson [1901] and also known as the Karhunen–Loève Transform.

Consider the extreme examples shown in figures 5.6 and 5.7 on the facing page. The data

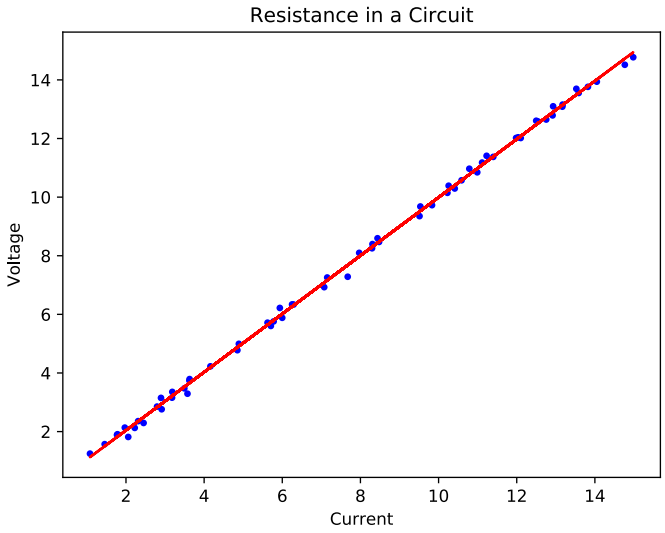
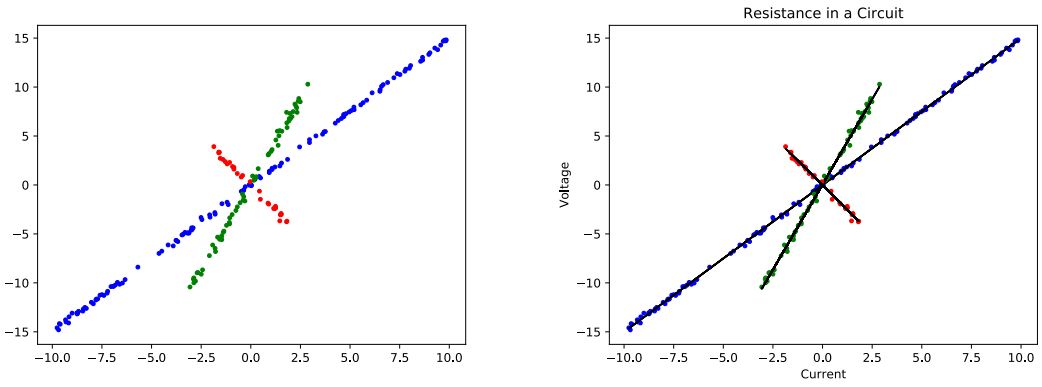


Figure 5.6: Example dataset of 64 data points with an embedded linear direction.

points in figure 5.6 has a linear structure, while the data points in figure 5.7 on the facing page appears to have three axes or directions as shown by the linear fits in figure 5.7(b).

In these cases, it is clear that we should exploit the linear aspects we can see in the data. If we look at the individual data points that make up data set in figure 5.6, we can represent each data points as a vector $\boldsymbol{x}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$ and therefore be able to create the data matrix by vertically stacking all s data points $\boldsymbol{x}_i$

$$\boldsymbol{X} = \begin{bmatrix} x_0 & x_1 & x_2 & \dots & x_{s-1} \\ y_0 & y_1 & y_2 & & y_{s-1} \end{bmatrix} \tag{5.15}$$



(a) (b)

Figure 5.7: Example datasets of 128, 64 and 32 data points with multiple embedded linear directions.

If we ignore the small "noise" in our data points from figure 5.6 on the facing page, our most "compact" representation is that of a line. Therefore, we should be able to represent each point as simply scalings b of a vector $\boldsymbol{u}$ with the slope corresponding to the line. Thus in theory, we can represent the entire data matrix $\boldsymbol{X}$ with a new matrix $\boldsymbol{U}$ of k columns, so that $k \ll s$ and $k = 1$ in our case giving

$$\boldsymbol{U} = \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \tag{5.16}$$

Likewise for the dataset in figure 5.7, we can represent the entire data matrix $\boldsymbol{X}$ with a new matrix $\boldsymbol{U}$ of $k = 3$ columns

$$\boldsymbol{U} = \begin{bmatrix} u_0 & u_1 & u_2 \\ v_0 & v_2 & v_2 \end{bmatrix} \tag{5.17}$$

with each of $\boldsymbol{u}_i$ having the directions corresponding to the slope of each line. We can then recover each data point in the dataset by creating a linear combination of these $\boldsymbol{u}_i$ vectors as

$$\boldsymbol{x}_i = \sum_{i=0}^{k-1} b_i \boldsymbol{u}_i. \tag{5.18}$$

Notice that here we have reduced the number of columns of our original dataset from s to k. This is in essence the notion of the dimensionality reduction that PCA provides. The $\boldsymbol{u}_i$ vectors are known as eigenvectors and form the principal components. Computing these vectors is based on the fundamental concept of eigen-decomposition.

5.3.1 Eigen-decomposition

Recall that in the chapter on symmetry, we defined symmetry as a property that remains unchanged after a transformation. Eigen-decomposition provides a method to do this with

respect to basis vectors of a given dataset. Given a transformation $\boldsymbol{A}$ of vectors of n dimensions, the goal of eigen-decomposition is to find vectors $\boldsymbol{u}$ such that

$$\boldsymbol{A}\boldsymbol{u} = \lambda \boldsymbol{u}, \tag{5.19}$$

where λ is a scalar value called the eigenvalue. In other words, these eigenvectors $\boldsymbol{u}$ are vectors that remain invariant to any rotations induced by transformation $\boldsymbol{A}$. The vectors $\boldsymbol{u}$ do not change direction when $\boldsymbol{A}$ is applied to them.

In our examples from figures 5.6 on page 24 and 5.7 on the previous page, these eigenvectors were oriented along the lines shown. Any point in those datasets can be represented by a linear combination of scaled version of the eigenvectors shown in equation (5.18). Now using linear algebra, we can extend the concept to any number of training samples s and points N.

To compute this eigen-decomposition, we can use the singular value decomposition (SVD) algorithm that decomposes the matrix $\boldsymbol{A}$ as

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{W}\boldsymbol{V}^T, \tag{5.20}$$

where $\boldsymbol{W}$ is a diagonal matrix whose entries are the singular values in descending order that correspond to the eigenvalues. Note that the data matrix should be centered, i.e. $\boldsymbol{A} = \hat{\boldsymbol{X}} = \boldsymbol{X} - \boldsymbol{\mu}$ where $\boldsymbol{\mu}$ is the mean of the training vectors or columns. The matrix $\boldsymbol{U}$ corresponds to the eigenvectors of the decomposition.

5.3.2 Dimensionality Reduction

In terms pattern recognition, we determine the eigen-decomposition a rectangular $(N \times s)$ data matrix $\boldsymbol{X} = \boldsymbol{x}_j$ (see algorithm 1). Thus, meaningful variations are obtained for each row in $\boldsymbol{X}$ that represents the same regions of the data.

Algorithm 1 PCA Algorithm [Pearson, 1901, adapted]

1: $\boldsymbol{X} \leftarrow$ dataset
2: $\boldsymbol{\mu} \leftarrow$ mean vector from $\boldsymbol{X}$.
3: $\hat{\boldsymbol{X}} \leftarrow \boldsymbol{X} - \boldsymbol{\mu}$ ▷ Each column consists of variational points
4: Compute the SVD of $\hat{\boldsymbol{X}}$ to get estimate of $\boldsymbol{U}$.
5: **return** $\boldsymbol{U}$ and λ_i ▷ The PCA model

The PCA produces an eigen-decomposition of the data matrix $\hat{\boldsymbol{X}}$, where $\hat{\boldsymbol{X}}$ represents the training data with the mean vector $\boldsymbol{\mu}$ removed. The eigen-decomposition results in a set of eigenvectors $\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_s] = \boldsymbol{u}_i$ and eigenvalues λ_i that effectively represent a set of modes that model how each data point varies from the mean data point given the training vectors. One can then project a training vector $\boldsymbol{S} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_s] = \boldsymbol{v}_i$ as

$$\boldsymbol{b}_i = \boldsymbol{U}^T \left(\boldsymbol{v}_i - \boldsymbol{\mu}_i \right), \tag{5.21}$$

into the variation (PCA) model to acquire the variation coefficients $\boldsymbol{b}_i$. The data vector $\boldsymbol{S}$ can then be reconstructed as a new vector $\hat{\boldsymbol{S}} = \hat{\boldsymbol{v}}_i$ within the constraints of the training set

as

$$\hat{\boldsymbol{v}}_i = \sum_j^k b_{ji} u_{ji} + \boldsymbol{\mu}_i \approx \boldsymbol{v}_i, \tag{5.22}$$

where k is the number of modes, so that $k \leqslant s$ and usually $k \ll s$. The fact that k ls is the fundamental concept behind dimensionality reduction. In general, only the most significant modes are used, up to a precision of representation $P : 0 < P \leqslant 1$, for computational efficient and stable reconstructions.

5.3.3 PCA Variants

The robust PCA of Skočaj et al. [2007] that can handle missing or corrupted data (hereon referred to as missing data) using binary weights $\boldsymbol{w}_{ij}$ to represent missing or known data for each point in each training surface. In this robust PCA, the variations are modelled from the mean of the known data as

$$\boldsymbol{\mu}_i = \frac{1}{\sum_j^N \boldsymbol{w}_{ij}} \sum_j^N \boldsymbol{w}_{ij} \boldsymbol{x}_{ij}. \tag{5.23}$$

Any detected outliers are replaced by iteratively reconstructed values from a model of the reliable or known data using an expectation maximisation (EM) algorithm (see algorithm 2).

Algorithm 2 Iterative Reconstruction PCA Algorithm [Skočaj et al., 2007]

$\boldsymbol{X} \leftarrow$ known data.
$\boldsymbol{W} \leftarrow$ binary weights representing missing and known data. ▷ Can accommodate partial data in training
$\boldsymbol{\mu} \leftarrow$ weighted mean from known data from equation (5.23).
$\hat{\boldsymbol{X}} \leftarrow \boldsymbol{X} - \boldsymbol{\mu}$
Apply PCA on known data in $\hat{\boldsymbol{X}}$ via algorithm 1. ▷ Initialise the algorithm with the known model
$\boldsymbol{Y} \leftarrow \hat{\boldsymbol{X}}$
while not converged **do**
    E-step: Project known data with the pseudo-inverse of weighted $\boldsymbol{U}$. ▷ Element-wise product with $\boldsymbol{W}$
    M-step 1: Reconstruct $\hat{\boldsymbol{X}}$ with current model.
    M-step 2: $\boldsymbol{Y} \leftarrow \hat{\boldsymbol{X}}$ with missing data replaced by reconstructed values. ▷ Recovers missing data here
    M-step 3: Re-estimate $\boldsymbol{U}$ by PCA on $\boldsymbol{Y}$.
end while
return $\boldsymbol{U}$ and λ_i ▷ The robust shape model

The spatially weighted PCA of Thomaz et al. [2010][1] is simpler to implement than the robust PCA as can be seen from comparing algorithms 2 and 3. The weights

$$\boldsymbol{w}_i = \left[\sqrt{w}_1, \sqrt{w}_2, \ldots, \sqrt{w}_N\right], \tag{5.24}$$

are determined from regions that of the data that needs to be contoured and are applied to all training samples via the correspondences present. To reduce computation complexity in the projection stage, only the first $k \leqslant s$ need to be used.

[1] See also the related work of Thomaz and Giraldi [2010].

Algorithm 3 Spatially Weighted PCA Algorithm [Thomaz et al., 2010]

1: $\boldsymbol{X} \leftarrow$ dataset.
2: $\boldsymbol{w}_i \leftarrow$ square root of the weights representing regions. ▷ Weights represent important regions
3: $\boldsymbol{\mu} \leftarrow$ weighted mean from equation (5.23).
4: $\hat{\boldsymbol{X}} \leftarrow \boldsymbol{X} - \boldsymbol{\mu}$
5: Multiply each row of $\hat{\boldsymbol{X}}$ with $\boldsymbol{w}_i$ to get $\boldsymbol{Y}$. ▷ Weight each corresponding point
6: Apply PCA on $\boldsymbol{Y}$ via algorithm 1.
7: **return** $\boldsymbol{U}$ and λ_i ▷ The focused shape model

The spatially weighted PCA of Thomaz et al. [2010] has important distinctions to the temporally and spatially weighted PCAs proposed by Skočaj et al. [2007]. The temporally weighted PCA weights each training shape (or column of the training matrix), which is suitable for handling unreliable or low quality training shapes as a whole. The spatially weighted PCA of Skočaj et al. [2007] is a general scheme solved using an EM algorithm for weighting arbitrary points in any of the training shapes and not just each corresponding point (or row of the training matrix) as with the spatially weighted PCA of Thomaz et al. [2010], so that $\boldsymbol{w}$ is a matrix and not a vector. In the spatially weighted PCA of Thomaz et al. [2010], entire regions maybe weighted and their variations from the mean shape scaled as desired. This weighting of a corresponding point for all training surfaces such as (5.24) is not possible with the spatially weighted PCA of Skočaj et al. [2007] as the weightings are lost from the eigenvectors in the process of removing the weighted mean from the data matrix[2].

The last consideration when using the weighted PCA is that it cannot accommodate missing data in the same way as the robust PCA. This means that zero-valued weights cannot be used, but can be handled by using very small values (e.g. 1×10^{-6}) instead.

5.4 Radon Transform

Svalbe [1989]

5.5 Wavelet Transforms

[2]Equations (7) and (8) of the work of Skočaj et al. [2007] separates the weights from the eigenvectors and so no weights (such as (5.24)) are directly encoded in the resulting eigenvectors leading to no weighted dimensionality reduction normally afforded to us by the PCA.

Part IV

Appendix

Abbreviations

FT Fourier transform 18
FFT fast Fourier transform 23
DFT discrete Fourier transform 23
EM expectation maximisation 27
PCA principal component analysis 24
SVD singular value decomposition 26
2D two dimensional 5
3D three dimensional 5
RF random forest 29
CNN convolutional neural network vi

Citation Index

Bibliography

Earman, J., Glymour, C., 1978. Lost in the tensors: Einstein's struggles with covariance principles 1912 ??1916. Studies in History and Philosophy of Science Part A 9, 251 – 278. URL: http://www.sciencedirect.com/science/article/pii/0039368178900080, doi:https://doi.org/10.1016/0039-3681(78)90008-0.

Euclid, 300BCE. The Elements.

Galois, É., 1830. Analyse d un mémoire sur la résolution algébrique des équations. Bulletin des sciences mathématiques physiques et chimiques 13, 171–172.

Ho, T.K., 1995. Random decision forests, in: Proceedings of 3rd International Conference on Document Analysis and Recognition, pp. 278–282 vol.1. doi:10.1109/ICDAR.1995.598994.

Klein, F., 1893. Vergleichende betrachtungen über neuere geometrische forschungen. Mathematische Annalen 43, 63–100. URL: https://doi.org/10.1007/BF01446615, doi:10.1007/BF01446615.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks, in: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (Eds.), Advances in Neural Information Processing Systems 25. Curran Associates, Inc., pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Lecun, Y., Bengio, Y., 1995. Convolutional networks for images, speech, and time-series, in: The handbook of brain theory and neural networks. MIT Press.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al., 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE 86, 2278–2324.

Lorenz, E.N., 1963. Deterministic Nonperiodic Flow. Journal of the Atmospheric Sciences 20, 130–141. doi:10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2.

Mandelbrot, B., 1982. The Fractal Geometry of Nature. 1st edition edition ed., W. H. Freeman and Company, San Francisco.

Noether, E., 1918. Invariante variationsprobleme. Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse 1918, 235–257. URL: http://eudml.org/doc/59024.

Olshausen, B.A., Field, D.J., 1996. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature 381, 607–609. URL: https://doi.org/10.1038/381607a0, doi:10.1038/381607a0.

Pearson, K., 1901. On lines and planes of closest fit to systems of points in space. Philosophical Magazine 2, 559–572.

Ringach, D.L., 2002. Spatial structure and symmetry of simple-cell receptive fields in macaque primary visual cortex. Journal of Neurophysiology 88, 455–463. URL: https://doi.org/10.1152/jn.2002.88.1.455, doi:10.1152/jn.2002.88.1.455, arXiv:https://doi.org/10.1152/jn.2002.88.1.455. pMID: 12091567.

Skočaj, D., Leonardis, A., Bischof, H., 2007. Weighted and robust learning of subspace representations. Pattern Recognition 40, 1556–1569. doi:10.1016/j.patcog.2006.09.019.

Svalbe, I.D., 1989. Natural representations for straight lines and the hough transform on discrete arrays. Pattern Analysis and Machine Intelligence, IEEE Transactions on 11, 941–950. doi:10.1109/34.35497.

Thomaz, C., Giraldi, G., da Costa, J., Gillies, D., 2010. A Simple and Efficient Supervised Method for Spatially Weighted PCA in Face Image Analysis. Technical Report. Department of Computing, Imperial College London.

Thomaz, C.E., Giraldi, G.A., 2010. A new ranking method for principal components analysis and its application to face image analysis. Image and Vision Computing 28, 902–913. doi:10.1016/j.imavis.2009.11.005.