Essential Mathematics for Neuroscience

Fabian Sinz, Jakob Macke & Philipp Lies

December 11, 2009

Contents

Chapter 1

Basics

This chapter serves as an introduction to a few basic elements that will be needed troughout the course. We begin by reviewing basic families of functions like *linear functions*, *polynomials* and *trigonometric functions*, as well as some of their properties. Afterwards we will look at some elementary calculus on those types of functions.

1.1 Essential Functions

A function is a rule that relates to sets of quantities, the *inputs* and the *outputs*. Each input x is deterministially related to an output $f(x)$. For example, $f(x)$ might temperature on day x, or the firing rate of a neuron in response to a stimulus x. Thus, functions can be used as mathematical models of processes in which one quantity is transformed into another in a deterministic way. Even when the process of transformation is not deterministic, usually an underlying deterministic process corrupted by random noise can be used. In the example above, the firing rate of the neuron could be $f(x) + \varrho$, where ϱ is a noise-term. In contrast to a deterministic function, $f(x) + \varrho$ denotes a whole set of values for a given x since the random term ϱ can take different values for each trial. Therefore, $f(x) + \varrho$ is not a function in the strict sense. The reason is that, functions—by definition—are rules how to assign elements x of one set to *unique* elements $f(x)$ of another set. Only if the target elements are unique, the assignment rule is called *function*. When defining a function, we have to specify the two *sets* between the function is mapping and the *rule* that transforms an element of the target set to an element of the input set. For example, if we want to define a function f that is transforming elements of a set A into elements of a set B according to the rule r, we would write this as

$$\begin{aligned} f: \quad & A \to B \\ & a \mapsto r(a). \end{aligned}$$

Here, a is an element of A (written $a \in A$) and $r(b)$ is an element of B (i.e. $r(b) \in B$). The set A is usually called *domain of* f while B is called *the co-domain of* f. The arrow "$\to$" is used to denote the mapping between the two sets, while "$\mapsto$" denotes the mapping from an element of the domain to an specific element of the co-domain. This means that "$\to$" tells us what kind of objects are mapped into another and "$\mapsto$" specifies the assignment rule.

The rule r can be anything that can be done with elements of A. For example, if the function f simply doubles any real number, we would write

$$\begin{aligned} f: \quad & \mathbb{R} \to \mathbb{R} \\ & x \mapsto 2 \cdot x \quad x \in \mathbb{R}. \end{aligned}$$

In most cases, the inputs and outputs of function will be numbers, but this does not necessarily have to be the case (i.e. the elements of the domain A and the co-domain B do not need to be numbers).

Although, in principle, there are infinitely many functions on the real numbers, knowing only a few of them is usually enough to get along well in most natural sciences. The reason is that most complex functions are built by adding, multiplying, or composing simpler ones. It is important that you get comfortable with those simper functions since they are your toolbox to understand and build more complex functions. Once you have and intuition how those simple functions behave, it is often not too difficult to get a feeling for a more complicated one. In this section we will review the most important simple functions and present their most important properties.

1.1.1 Polynomials and Powers

Polynomials is a very common class of functions. The two most widely known kinds of polynomials are the parabola $f(x) = x^2$ and the more general quadratic function $f(x) = ax^2+bx+c$. In general, polynomials consist of a sum of positive integer powers k of x with coefficients a_k:

$$f(x) \quad = \quad a_n x^n + ... + a_1 x + a_0.$$

The single terms in the sum are called *monomials.* The *degree* of the polynomial is the largest exponent of its monomials. The polynomial above has a degree of n. Polynomials have nice properties like e.g. the *derivatives* and *anti-derivatives* of polynomials are easy to calculate and yield polynomials again. One frequent use of polynomials is to approximate any function at a certain location. This approximation is called *Taylor-Expansion.* We will discuss the Taylor-Expansion and many properties of polynomials in later chapters.

This is a good point to introduce the notation for sums over several elements: Instead of indicating the entire sum by three dots "..." we use the greek uppercase letter sigma Σ (like sum) to indicate a sum over all terms directly after the sigma. These terms are usually indexed and the range of the index is written below and above the Σ. Since $x^0 = 1$ for all $x \in \mathbb{R}$ we write the polynomial from above as

$$\begin{aligned} f(x) \quad &= \quad a_n x^n + ... + a_1 x + a_0 \\ &= \quad \sum_{k=0}^{n} a_k x^k. \end{aligned}$$

While polynomials have exponents $k \in \mathbb{N}_0$ (where $\mathbb{N}_0$ denotes the set of natural number including 0), exponents can in principle be in $\mathbb{R}$ as well. There are two most important cases: when the exponent is negative and when it is a rational number (i.e. a number that can be written as a fraction). A negative exponent of a number is merely a shortcut for $x^{-a} = \frac{1}{x^a}$. In many cases, for example when calculating derivatives, the notation with negative exponent is useful. A fraction in the exponent is another way of writing roots. For example the square root $\sqrt{x}$ is equivalently written as $x^{\frac{1}{2}}$. In general, the nth root of x can be written as $\sqrt[n]{x} = x^{\frac{1}{n}}$.

We conclude this section by stating a few calculation rules for powers for $x, a \in \mathbb{R}$. You should know all of them by heart and be able to use them effortlessly.

Calculation Rules for Powers
The following rules apply to any $x, a \in \mathbb{R}$:

1. Anything to the power of zero is one: $x^0 = 1$
2. Multiplying two terms with the same basis is equivalent to adding their exponents: $x^a \cdot x^b = x^{a+b}$
3. Dividing two terms with the same basis is equivalent to subtracting their exponents: $\frac{x^a}{x^b} = x^a \cdot x^{-b} = x^{a-b}$
4. Exponentiating a term is equivalent to multiplying its exponents: $(x^a)^b = x^{a \cdot b}$
5. A special case of rule 3. is given by $\frac{1}{x^a} = x^{-a}$
6. The a^{th} root of x is given by $\sqrt[a]{x} = x^{\frac{1}{a}}$ for $x \geq 0$.

1.1.2 Linear Functions

Linear functions are among the simplest functions one can imagine. You can imagine a linear function as a line (plane, or hyperplane) through the origin. Algebraically, their key property is that the function value of a sum $x + y$ of elements x, y equals the sum of their function values $f(x) + f(y)$.The same is true for multiples of input elements, i.e. the function value of some multiple $a \cdot x$ of an element x from the domain is the multiple of the function value $a \cdot f(x)$. If any function fulfills these two properties, it is linear by definition.

Definition (Linear Function) A function $f : \mathbb{R} \to \mathbb{R}$ is said to be *linear* if it fulfills the following two properties:

$$f(x + y) = f(x) + f(y) \quad \text{for all } x, y \in \mathbb{R} \tag{1.1}$$
$$f(a \cdot x) = a \cdot f(x) \quad \text{for all } a, x \in \mathbb{R}. \tag{1.2}$$

$\Diamond$

These properties have remarkable consequences. While for general functions, a single input-output pair of values $(x, f(x))$ does not tell anything about the value of the function at other locations $y \neq x$, a single such pair is enough to know the value of a linear function at any location: Assume we are given the input-output pair $(x, f(x))$ and we know that f is linear. In order to calucate the value of f at another location y, we search for a scalar a that scales x into y, i.e. $y = a \cdot x$. Clearly, this scalar is easy is given by $a = \frac{y}{x}$. Once we know a, we can compute $f(y)$ via

$$\begin{aligned} f(y) &= f(a \cdot x) \\ &= a \cdot f(x). \end{aligned}$$

Example (Mathematical Modelling of Receptive Fields) For some neurons, it is often assumed that their responses, i.e. the spike rate $r(x)$, depends linearly on the stimulus x.

Assume our cell responds to a visual image I_1 with a spike rate of $r_1 = 20$ spikes per second and to another image I_2 with $r_2 = 60$ spikes per second. What spike rate would we expect to the mean of I_1 and I_2, if our neuron was truely linear in the stimuli? The answer is easy to calculate. Let $r : \mathcal{I} \to \mathbb{R}$ denote the function from images (denoted by $\mathcal{I}$) to spike rate. We already know $r(I_1) = r_1 = 20\frac{sp}{s}$ and $r(I_2) = \nu_2 = 60\frac{sp}{s}$. Then the response to the mean of the two images is

$$\begin{aligned} r\left(\frac{1}{2}I_1 + \frac{1}{2}I_2\right) &= r\left(\frac{1}{2}I_1\right) + r\left(\frac{1}{2}I_2\right) \\ &= \frac{1}{2}r(I_1) + \frac{1}{2}r(I_2) \\ &= \frac{1}{2}r_1 + \frac{1}{2}r_2 \\ &= 10\frac{sp}{s} + 30\frac{sp}{s} \\ &= 40\frac{sp}{s} \end{aligned}$$

This property does not only hold for two input stimuli. It holds for an arbitrary number of stimuli. If the rate function r realized of our neuron is linear, then response to the mean of n images is just the mean response to the single images.

$$\begin{aligned} r\left(\frac{1}{n}\sum_{k=1}^{n} I_k\right) &= \frac{1}{n}\sum_{k=1}^{n} r\left(I_k\right) \\ &= \frac{1}{n}\sum_{k=1}^{n} \nu_k \end{aligned}$$

Question:

Of course, real neurons are not truly linear. If a neuron was indeed linear, for inputs, this would lead to some very unrealistic conclusions. Name two of them!

Answer:

For some stimuli, the spike rate would be negative. Also, for stimuli with very high input, the spike-rate would be arbitrarily large, i.e. the neuron's rate would not saturate.

◁

1.1.3 Trigonometric Functions

Trigonometric functions are functions of an angle ϑ. The most common trigonometric functions are $\sin(\vartheta)$, $\cos(\vartheta)$, $\tan(\vartheta)$ and $\text{cotan}(\vartheta)$.

In general, there are two natural ways to think about trigonometric functions: the geometrical view quantities and the periodic signal view.

1.1.3.1 The geometric view

In the geometric view, $\cos(\vartheta)$ and $\sin(\vartheta)$ represent the x-coordinate and the y-coordinate of a point on the intersection between a circle with radius one centered at the origin, and a line through the origin that encloses an angle of ϑ with the x-axis. In this view, $\tan(\vartheta)$ and $\text{cotan}(\vartheta)$ have a natural interpretation as well, namely the length of the line, touching the circle, between the upper leg of the angle and the x-axis or the y-axis, respectively. Alternatively, $\tan(\theta)$ is the ratio between the $x-$and the $y-$coordinate (see Figure 1.1).

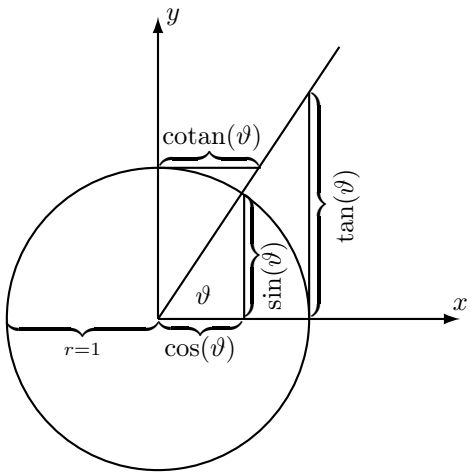
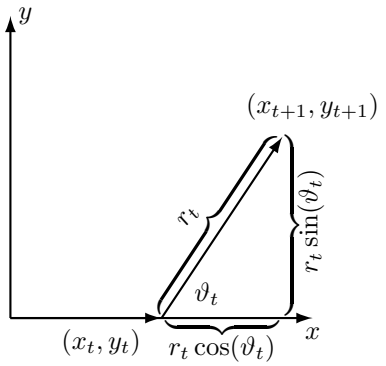


Figure 1.1: Geometrical view of $\sin(\vartheta)$, $\cos(\vartheta)$, $\tan(\vartheta)$ and $\text{cotan}(\vartheta)$. For a given angle ϑ, $(\cos(\vartheta), \sin(\vartheta))$ are the coordinates of the point on the intersection between a circle of radius $r = 1$ and the upper leg of the angle. $\tan(\vartheta)$ is the length of the line between the x-axis and the upper leg of the angle, that "touches" the unit circle (therefore the name *tangens* from *lat. tangere = to touch*). $\text{cotan}(\vartheta)$ is defined analogoulsy for the line touching the circle from above.

Example (Path Integration) The term *path integration* denotes the ability of moving organisms (such as ants) to remember the direction and length of the vector to its home while moving in the environment. We will now just look at a special case of updating the home vector in world coordinates, i.e. a global fixed coordinate system. This means that the home base is assigned the coordinates $(0, 0)$ and the moving organism stores its position according to a global coordinate frame.

Imagine you are an ant living in a completely flat world. For the sake of simplicity we further imagine that you have a compass and you know the length of your steps, so that you can measure the angles you turn and the distance you walked. Now imagine that you already explored your environment for a while and that you are now standing at position (x_t, y_t) looking along the x-axis. If you know turn by an angle of ϑ_t and move into the new direction by a distance of r_t, what is your new position (x_{t+1}, y_{t+1})?



Looking again at figure 1.1 and the figure above shows that the new direction in which you are walking is $(\cos(\vartheta_t), \sin(\vartheta_t))$. Since you are walking along that direction for a distance of r_t, the total displacement is $r \cdot (\cos(\vartheta_t), \sin(\vartheta_t))$. If we add this displacement to the old position, we get the new position

$$(x_{t+1}, y_{t+1}) \quad = \quad (x_t, y_t) + r_t \cdot (\cos(\vartheta_t), \sin(\vartheta_t))$$

◁

Due to the geometrical property of $\sin(\vartheta)$ and $\cos(\vartheta)$ we can derive a very useful equality by employing Pythagoras theorem.

Theorem (Pythagoras) For a right angle triangle with side lengths a, b and c, where c is the length of the longest leg, while the legs with length a and b enclose an angle of $90°$ (or $\frac{\pi}{2}$ in radians), the following equality holds:

$$a^2 + b^2 \quad = \quad c^2$$

◊

In the case of $\sin(\vartheta)$ and $\cos(\vartheta)$, we know the value of c for any given ϑ. It is simply $c = 1$, since the point $p = (\cos(\vartheta), \sin(\vartheta))$ lies on the circle of radius $r = 1$. Therefore, we know that

$$\cos(\vartheta)^2 + \sin(\vartheta)^2 \quad = \quad 1$$

for all angles ϑ.

There is another useful equality that we can read of figure 1.1. Assume we want to know the scaling factor s that transforms $\sin(\vartheta)$ into $\tan(\vartheta)$, i.e. $\sin(\vartheta) \cdot s = \tan(\vartheta)$. From looking at figure 1.1 we know that it is the same scaling factor that scales $\cos(\vartheta)$ into 1, i.e. $\cos(\vartheta) \cdot s = 1$. In this case, s is easy to calculate: It is simply $s = \frac{1}{\cos(\vartheta)}$. Therefore we have found a formula how to calculate $\tan(\vartheta)$ from $\sin(\vartheta)$ and $\cos(\vartheta)$:

$$\tan(\vartheta) = \frac{\sin(\vartheta)}{\cos(\vartheta)}.$$

In an analogous manner we can also derive the equality

$$\operatorname{cotan}(\vartheta) = \frac{\cos(\vartheta)}{\sin(\vartheta)}.$$

Since $\tan(\vartheta)$ and $\operatorname{cotan}(\vartheta)$ can be written in terms of a quotient, the radius of the circle does not even have to be one. Assume we want to know the tangens between the x-axis and the leg $(0, p)$ for a point $p = (x, y)$ that lies on a circle with radius r. Since we can write p equivalently as $p = (x, y) = r \cdot (\cos(\vartheta), \sin(\vartheta))$ for an appropriate value of r, we have that

$$\frac{y}{x} = \frac{r \cdot \sin(\vartheta)}{r \cdot \cos(\vartheta)} = \tan(\vartheta).$$

Therefore, $\tan(\vartheta)$ is simply the quotient of the opposite leg and the adjacent leg of a right angle triangle. Similarily we can get

$$\frac{x}{y} = \frac{r \cdot \cos(\vartheta)}{r \cdot \sin(\vartheta)} = \operatorname{cotan}(\vartheta).$$

Further, sine and cosine can also be defined in terms of quotients for circles with an arbitrary radius. According to the intercept theorem the ratio of $\cos(\vartheta)$ to 1 is equal to the ratio of x to r for every point $p = (x, y)$ on a circle wirh radius r. Equally, the ratio of $\sin(\vartheta)$ to 1 is equal to the ratio of y to r. Therefore, we get

$$\begin{aligned} \sin(\vartheta) &= \frac{y}{r} \\ \cos(\vartheta) &= \frac{x}{r} \end{aligned}$$

Other useful properties that can also be read off from the geometric view. Among them are the symmetry properties of $\sin(\vartheta)$, $\cos(\vartheta)$, $\tan(\vartheta)$ and $\operatorname{cotan}(\vartheta)$:

$$\begin{aligned} \cos(-\vartheta) &= \cos(\vartheta) \\ \sin(-\vartheta) &= -\sin(\vartheta) \\ \tan(-\vartheta) &= -\tan(\vartheta) \\ \operatorname{cotan}(-\vartheta) &= -\operatorname{cotan}(\vartheta) \end{aligned}$$

Sometimes we do not want to compute the sine or cosine of an angle but the angle itself. For example, we take point $p = (x, y)$ and want to know what

the angle between the x-axis and the line from the origin to p is. We know that $\tan(\vartheta) = \frac{y}{x}$, but what is ϑ? To solve for ϑ, we need the inverse trigonometric functions. For every trigonometric function there is an inverse trigonometric function: $\arccos(x)$, $\arcsin(x)$, $\arctan(x)$, $\operatorname{arccotan}(x)$. In some texts they are confusingly written as $\cos^{-1}(\vartheta)$ whereas $\cos(\vartheta)^{-1}$ means $1/\cos(\vartheta)$, so you should always use the arc-notation. Now, by using the arctan function we can compute the angle between p and the x-axis as $\vartheta = \arctan(\frac{y}{x})$.

Remark There are two units for measuring angles which are commonly used and get quite often mixed up: *radians* and *degrees*. A full circle of 360° corresponds to 2π radians. In computer programs, the default unit is radians. Therefore to compute the sine of 45° one has to compute $\sin(\pi/4)$. Likewise if you computed an angle by using $\arccos(x)$ your output is in radians. To convert from radians to degrees, you have to multiply your result by $\frac{180}{\pi}$ and similarly converting from degrees to radians by multiplying with $\frac{\pi}{180}$.

◁

1.1.3.2 The Periodic Signal View

The periodic signal view of thinking about $\sin(\vartheta)$ and $\cos(\vartheta)$ is especially useful when dealing with signals such as e.g. membrane potentials of neurons or stripes of natural images. It is also closely related to techniques such as Fourier- or Spectral Analysis, which are indispensable tools for the analysis of neurophysiological signals. In order to get from the geometric to the periodic signal view, just imagine a point on the unit circle that is moving with constant speed counterclockwise along the circle. If we plot the time t against the point's x-coordinate $x(t) = \cos(\vartheta(t))$, we get a strongly periodic function. Same applies to the y-coordinate $y(t) = \sin(\vartheta(t))$. Figure 1.2 shows the graphs of the two functions. For the moment we wrote ϑ as a function of time in order to be able to say that the point is moving with constant speed. The faster the point is moving along the circle the more cycles we can get in one fixed time interval. This is expressed by the frequency ω of the sine or cosine, respectively. It tells us how many cycles our point does in one unit time interval. We can therefore equivalently write $x(t) = \cos(\omega \cdot t)$ and $y(t) = \sin(\omega \cdot t)$. From now on we will drop the dependence on time. The frequency then tells us how many cycles of our point fit in the interval $[0, 2\pi]$.

Example (Sine and Cosine Gratings) Assume that you want measure the orientation selectivity a V1-cell you are recording from with an electrode. A simple experiment would be to present gratings with different orientations and varying amount of bars in one unit interval, i.e. different spatial frequency. The most common way of producing such patterns is to use *sine* and *cosine gratings*. Figure 1.3 shows such a grating.

Since an image can be seen as a function,that assigns a graylevel value to each pixel ($f(x)$is the gray value of pixel x) those gratings are simply sine or

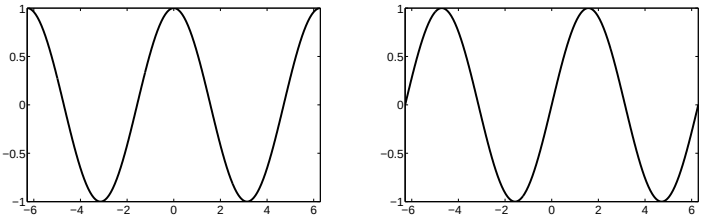


Figure 1.2: $\cos(\omega t)$ (left) and $\sin(\omega t)$ (right) in the interval of $t \in [-2\pi, 2\pi]$ with a frequency of $\omega = 1$.

cosine functions over $\mathbb{R}^2$. In order to get to the graylevel values at the different pixels, the function is discretized, i.e. is only evaluated at certain locations that correspond to the pixels. At the moment we shall only look at how to produce vertical or horizontal gratings. We will see how to produce gratings of arbitrary orientation later.

For a vertical grating, we know that the graylevel value along the vertical axis must be constant. If $I(x, y)$ denotes the function that represents the image, i.e. the functions that assigns a graylevel value to a position (x, y), we can produce a vertical grating by $I(x, y) = \sin(\omega x)$. Instead of using the sine we could as well have used the cosine function. A horizontal grating can be generated in exactly the same way by replacing x by y inside the sine function. Here is the matlab code to produce a horizontal grating of frequency $\omega = 2$:

```
>> [X,Y] = meshgrid([-2*pi:0.01:2*pi]); % get the sample points
>> omega = 2; % set the frequency
>> imagesc(sin(omega*X)) % display grating
>> colormap(gray)  % set colormap to gray values
>> axis off % switch off the axes
```

At the moment all our gratings have a fixed contrast, since $-1 \leq \sin(x), \cos(x) \leq 1$ for all $x \in \mathbb{R}$. However, we can vary the contrast by varying the amplitude of the sine. This is done by premultiplying an appropriate scaling factor $I(x, y) = A \cdot \sin(\omega x)$. In order to build that into the matlab code above, you must specify the maximum and the minimum gray value when calling the function `imagesc`, since it automatically scales the gray values otherwise

```
>> [X,Y] = meshgrid([-2*pi:0.01:2*pi]); % get the sample points
>> omega = 2; % set the frequency
>> A = 3; % set the contrast
>> maxA = 10; % set maximal contrast
>> imagesc(A*sin(omega*X),[-maxA,maxA]) % display grating
>> colormap(gray)  % set colormap to gray values
>> axis off % switch off the axes
```

◁

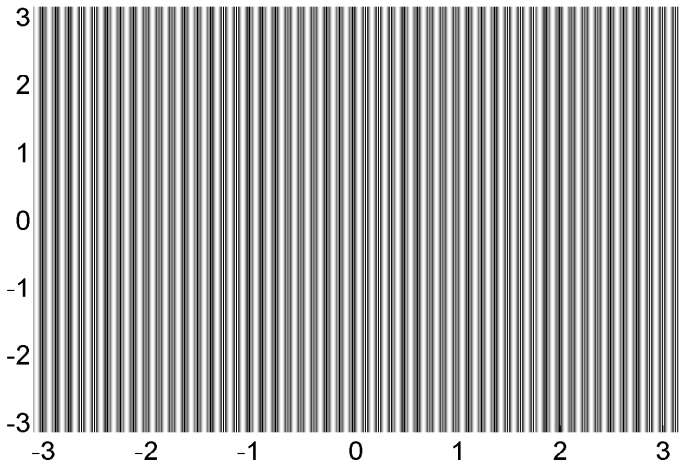


Figure 1.3: Example of a vertical sine grating with a spatial frequency of approx. 8Hz along the x-axis. The graylevel value at each position (x, y) is given by $I(x, y) = \sin(50 \cdot x)$.

Up to now we saw how to control the frequency and the amplitude of sine and cosine functions. In order to complete this subsection we will also see how to shift the sine and the cosine functions along the x-axis. Let us look at a sine function with a certain frequency $\sin(\omega t)$. At $t = 0$ also $\sin(\omega t) = 0$. If we want to shift the sine function along the x-axis by an offset of ϕ, we must ensure that the shifted version is zero at $t = \phi$ and not at $t = 0$. However, this will be the case, if we subtract ϕ from the argument of the sine. Therefore, a sine that is shifted by an offset of ϕ along the x-axis is given by $\sin(\lambda t - \phi)$. Cosine functions are shifted in exactly the same way. This offset ϕ is called *phase* of the signal. Now we are able to write down the general form of a sine or cosine signal with a given amplitude A and phase ϕ: It is

$$A \cdot \sin(\omega t - \phi) \quad \text{and} \quad A \cdot \cos(\omega t - \phi).$$

Before finishing this section about trigonometric functions we just want to mention two equalities that are useful when calculating with sine and cosine. These equalities are called the *Addition Theorems*.

Theorem (Addition Theorems) The following equalities hold for all $x, y \in \mathbb{R}$:

$$\begin{aligned} \cos(x + y) &= \cos(x)\cos(y) - \sin(x)\sin(y) \\ \sin(x + y) &= \sin(x)\cos(y) + \cos(x)\sin(y) \end{aligned}$$

$\Diamond$

Using the symmetry properties of $\sin(x)$ and $\cos(x)$ one can also derive similar expressions for $\cos(x-y)$ and $\sin(x-y)$.

Exercise

E

Write down the the corresponding expressions for $\cos(x-y)$ and $\sin(x-y)$.

◁

Important Rules for Trigonometric Functions
The following rules apply to any $x, y, \vartheta \in \mathbb{R}$:

- Pythagoras's Theorem: $\cos(\vartheta)^2 + \sin(\vartheta)^2 = 1$
- Symmetry Properties:

$$\begin{aligned} \cos(-\vartheta) &= \cos(\vartheta) \\ \sin(-\vartheta) &= -\sin(\vartheta) \\ \tan(-\vartheta) &= -\tan(\vartheta) \\ \text{cotan}(-\vartheta) &= -\text{cotan}(\vartheta) \end{aligned}$$

- Addition Theorems

$$\begin{aligned} \cos(x+y) &= \cos(x)\cos(y) - \sin(x)\sin(y) \\ \sin(x+y) &= \sin(x)\cos(y) + \cos(x)\sin(y) \end{aligned}$$

1.1.4 The e-function and the Logarithm

1.1.4.1 The Exponential Function

The exponential or e-function $f(x) = e^x = \exp(x)$ is one of the most frequently occurring and important function in the everyday life of a natural scientist. The number denoted by "e" is an irrational number called *Euler's number*. Its first digits are $e \approx 2.7183$. You should remember the approximate value of its inverse $\frac{1}{e} \approx 0.37$ because it is used to define time constants of neural signal transduction.

The exponential function appears in many probability distribution in statistics, in solution of differential equations and will appear in many mathematical model of neural processes. The exponential function has some very nice properties. One of them is that it is its own derivative $f'(x) = (e^x)' = e^x$.

The following examples show three cases, in which the exponential function naturally occurs.

Examples

1. One central probability density in statistics is the *Normal* or *Gaussian Distribution* $\mathcal{N}(\mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$. In many experiments that involve

noisy measurements, the noise is assumed to be Gaussian with mean $\mu = 0$. One possible justification for this assumption is that sums of random quantities with other probability distributions tend to be Gaussian if the total number of this quantities increases. Since we think of the noise in an experiment as the superposition of many other processes that we are not interested in, the assumption that there are a large number of them which are linearly superimposed motivates the Gaussian noise assumption.
Apart from that, the Gaussian distribution is frequently used because it has a lot of properties that make it possible to calculate analytical solutions of the respective statistical problems.

2. The potential change $\Delta V_m(t)$ over time at a passive neuron membrane after applying a rectangular current pulse can be described by the following equation

$$\Delta V_m(t) \quad = \quad I_m R \left(1 - e^{-\frac{t}{\tau}}\right).$$

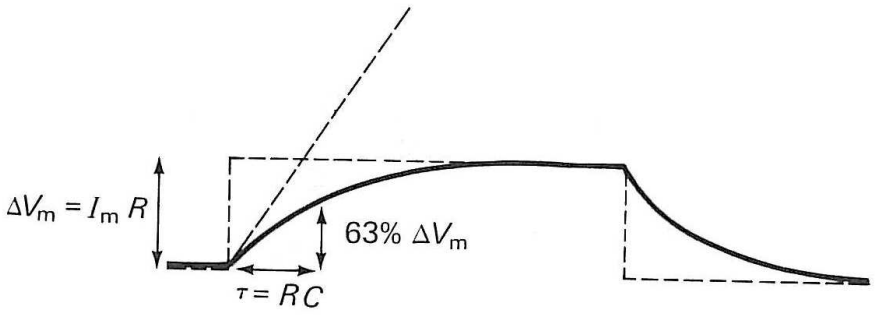


Figure 1.4: Potential change $\Delta V_m(t)$ (solid line) over time at a passive neuron membrane after applying a rectangular current pulse (dashed line) [Figure from [1]].

Figure 1.4 shows the time course of the potential. Here I_m is the current, that has been injected, R is the membrane resistance and τ is the *time constant* of the membrane. It tells us how fast the membrane potential follows the rectangular pulse. The greater τ is, the longer it takes until $\Delta V_m(t) = I_m R$, where $I_m R$ is the potential change induced by injecting the current I_m. In some sense τ defines a time scale for that membrane. τ is the time needed until the potential change reaches $0.63 \cdot I_m R = (1 - 0.37) \cdot I_m R = (1 - \frac{1}{e}) I_m R$, i.e. 63% of the potential change induced by the rectangular pulse. One can show that $\tau = RC$, where C is the membrane capacitance, i.e. its ability to buffer charge.

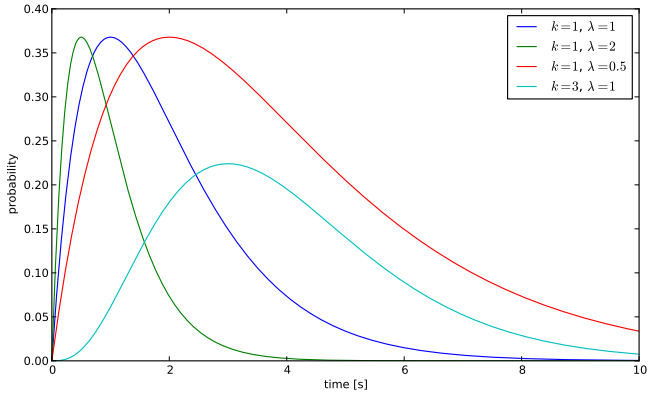


Figure 1.5: Poisson Process with 4 different sets of parameters.

3. A simple stochastical model for spike generation by a neuron is the *Poisson Process.* In this model, time is divided into a large number of bins. In each bin a spike occurs with probability p independent of whether a spike occurred in the bin before or not. If p is sufficiently small and the average spiking rate is λ, the probability of observing exactly k spikes in a time window of length Δt is given by the distribution of the Poisson Process with rate λ:

$$P(k, \Delta t) \quad = \quad \frac{e^{-\lambda \Delta t}(\lambda \Delta t)^k}{k!}.$$

Figure 1.5 gives an example how a Poisson Process looks like for different sets of parameters. The symbol expression "$k!$" denotes the *factorial function* $k! = k \cdot (k-1) \cdot ... \cdot 2 \cdot 1$. We can use a notation similar to the Σ for sums to denote a product of several components: The uppercase Greek letter Π (for product) denotes a product of all elements following it. The indices are written in the same way as for sums (see also Appendix 3.1). Therefore, we can write the factorial function as $k! = \prod_{n=1}^{k} n$.

4. If a spike train is generated by a *Poisson Process,* then the distribution of the *inter-spike-intervals (ISIs)* is an *exponential distribution.* That is, if we observe a spike at time 0, then the probability (density) of observing the next spike at time s is given by $p(s) = \mu e^{-\mu s}$, for $\mu = 1/\lambda$ in the example above.

◁

1.1.4.2 Logarithms

Logarithms and their derivatives often occur in statistics. Estimating the parameters of a statistical model is often done via maximum likelihood estimation.

This involves taking the derivative of the log of the likelihood function.

Here we introduce logarithms. More advanced examples will be discussed in later chapters. For now, we start with a small example:

Example In this example, we look again at the time course of the membrane potential after the application of a rectangular current pulse $\Delta V_m(t) = I_m R\left(1 - e^{-\frac{t}{\tau}}\right)$. Assume that we want to measure the time constant of a certain membrane. For this purpose we excited the membrane with a rectangular current pulse and measured $\Delta V_m(t)$ at several points t_k, $k = 1, ..., n$ in time. Now we want to solve $\Delta V_m(t) = I_m R\left(1 - e^{-\frac{t}{\tau}}\right)$ for τ at each time step t_k and get n values τ_k that we average to get your final estimation $\hat{\tau} = \frac{1}{n}\sum_{k=1}^{n} \tau_k$ of the membrane's time constant. How do we solve for τ_k? The first step is easy: You rearrange the the terms to get

$$\begin{aligned} \Delta V_m(t) = I_m R\left(1 - e^{-\frac{t}{\tau_k}}\right) \quad &\Leftrightarrow \quad I_m R - \Delta V_m(t_k) == I_m R \cdot e^{-\frac{t_k}{\tau_k}} \\ &\Leftrightarrow \quad \frac{I_m R - \Delta V_m(t_k)}{I_m R} = e^{-\frac{t_k}{\tau_k}}. \end{aligned}$$

Now we somehow have to extract $-\frac{t_k}{\tau_k}$ from the exponent. This means that you are searching for a number a, such that $e^a = \frac{I_m R - \Delta V_m(t_k)}{I_m R}$. This is exactly the definition of the *natural logarithm* $\ln(x)$: It is the number that you have to put in the exponent of e in order to obtain x. Now you can solve for τ:

$$\begin{aligned} \frac{I_m R - \Delta V_m(t_k)}{I_m R} = e^{-\frac{t_k}{\tau_k}} \quad &\Leftrightarrow \quad \ln\left(\frac{I_m R - \Delta V_m(t_k)}{I_m R}\right) = -\frac{t_k}{\tau_k} \\ &\Leftrightarrow \quad -\frac{\ln\left(\frac{I_m R - \Delta V_m(t_k)}{I_m R}\right)}{t_k} = \frac{1}{\tau_k} \\ &\Leftrightarrow \quad -\frac{t_k}{\ln\left(\frac{I_m R - \Delta V_m(t_k)}{I_m R}\right)} = \tau_k \end{aligned}$$

◁

We just saw that the natural logarithm is the function that cancels the exponential function, i.e. $\ln(e^x) = e^{\ln(x)} = x$. In general, a function g that cancels another function f is called *inverse function* of f and is denoted by $g = f^{-1}$. Not all functions have inverses. Some of them only have an inverse on a restricted range. We will discuss inverse function in more detail in the chapter about analysis.

So far we have seen how to solve e^x for x by using the natural logarithm. What if we want to solve an equation like 2^x for x? Here we cannot use the natural logarithm since $\ln(x)$ is only the inverse function of e, not 2. Here we must use a logarithm that fits to 2. This logarithm is denoted by $\log(x)$ and has the property that $2^{\log(x)} = \log(2^x) = x$. In general there is a logarithm for

any number b that is the inverse of the function $f(x) = b^x$. The number b is called *base* of the logarithm. If the base is not $b = 2$ or $b = e$, we indicate the base in the subscript of $\log_b(x)$. However, the only frequently used logarithms are $\log(x) = \log_2(x)$ and $\ln(x) = \log_e(x)$.

Remark It happens quite often that the base of of the logarithm is not specified, either because it does not matter or it is clear from the context. In this case, the notation $\log(x)$ is usually used. Usually this means that the base is e (e.g. Physics), sometimes it means that the base is 2 (e.g. in Information Theory) and sometimes it is used for base 10 (e.g. Economics). In the remaining part of the script we simply use $\log(x)$ to denote any logarithm. The base should always be clear from the context or it does not matter. If we want to emphasize a certain base we will write it in the subscript or use the explicit notation $\ln(x) = \log_e(x)$ or $\lg(x) = \log_{10}(x)$.

There is a neat trick how to calculate logarithms of arbitrary bases by using logarithms of another base:

$$\log_b(x) \quad = \quad \frac{\ln(x)}{\ln(b)} = \frac{\log(x)}{\log(b)}.$$

Until now we skipped an important detail of logarithms. When only dealing with real numbers, the logarithm is only defined on the strictly positive part of $\mathbb{R}$. We denote this set by $\mathbb{R}^+$. The reason for this restriction is easy to see. If we remember that $\log_b x$ is the number that has to be put in the exponent of b in order to obtain x. If x is negative, there can generally be no such real number since b^x is positive. (The concept of logarithm can be extended to negative and imaginary numbers, but does lead to some complications, so we will not cover it here.)

We conclude this section with a few calculation rules. Most of them follow directly from the calculation rules of powers or the definition of the logarithm.

Calculation Rules for Logarithms
The following rules apply to any logarithm:

- $\log_b(b^x) = x$
- $\log_b(x \cdot y) = \log_b(x) + \log_b(y)$ for $x, y \in \mathbb{R}^+$
- $\log_b\left(\frac{x}{y}\right) = \log_b(x) - \log_b(y)$ for $x, y \in \mathbb{R}^+$
- $\log_b(x) = \frac{\ln(x)}{\ln(b)} = \frac{\log(x)}{\log(b)}$ for $x, b \in \mathbb{R}^+$
- $\log_b(x^y) = y \cdot \log_b(x)$ for $x \in \mathbb{R}^+$, $y \in \mathbb{R}$

1.1.5 Lines (Affine Functions)

Most people think about lines when they think about linear functions. However, lines are not generally linear functions. Only the lines that include the origin are strictly speaking linear functions. Lines versions of linear functions that are shifted along the y-axis. The general equation for a line is

$$f(x) \quad = \quad mx + t,$$

where m is called the *slope* of f. It is the first derivative or, equivalently, the amount about f changes if we increase x by one, i.e. $m = f(x+1) - f(x)$. The value of t determines the y coordinate of the point where f cuts through the y-axis. This can easily been seen by evaluating f at $x = 0$. Obviously, the function value $f(0) = t$ must be the location on the y-axis where f hits it.

From the general form of lines we can also see why they are not strictly linear if $t \neq 0$. If they were linear, $f(x+y) = f(x) + f(y)$ would have to hold for all $x, y \in \mathbb{R}$. However, it is easy to check that this is not the case:

$$\begin{aligned} f(x) + f(y) &= mx + t + my + t \\ &= m(x+y) + 2t \\ &\neq m(x+y) + t \\ &= f(x+y). \end{aligned}$$

Therefore, lines with $t \neq 0$ are not linear. But we can always make them linear by subtracting t. This yields a line with the same slope m, which is shifted along the y-axis such that it cuts through $(0, 0)$, which make it a truly linear function.

Functions of the form $f(x) = mx + t$ are also called *affine functions*.

1.1.6 Piecewise Defined Functions

Sometimes, it is convenient to define a function by using two or more other functions. This is useful if we want to change the behaviour of the function on certain parts of the domain. Achieving a certain behaviour with a single expression might be difficult. It is then usually easier to use an several expressions, one for each part of the domain. These functions are called *piecewise defined functions*. We just mention a few important examples here.

Examples

1. The Heaviside function is defined as:

$$f(x) = \begin{cases} 0 & x \leq 0 \\ 1 & x > 0 \end{cases}.$$

2. The absolute value is given by

$$f(x) = \begin{cases} -x & x \leq 0 \\ x & x > 0 \end{cases}$$

3. The maximum-function is defined as

$$f(x,y) = \begin{cases} y & x \leq y \\ x & x > y \end{cases}$$

◁

Piecewise defined functions can for example be used to define a more realistic model of neurons.

Example: Neurons that are linear over some range.

Earlier, we saw that neurons are linear for some stimuli x, but clearly not for all: Firstly, the firing rate of a neuron can not be negative, and secondly, there is a maximal firing rate that can not be exceeded. Let us suppose that a neuron responds to a one-dimensional stimulus x with firing rate $f(x)$.

$$f(x) = \begin{cases} 0 & x < 0 \\ x & 0 \leq x \leq f_{max} \\ f_{max} & x > f_{max} \end{cases}$$

◁

Piecewise functions are just like any other function. Within each interval, the functions can be differentiated and integrate like other functions. Nevertheless, there is some care needed at the points at which the intervals meet. In particular, it is often (but not always) the case that piecewise functions are not continuous of differentiable at these points.

1.1.7 Sketching Functions

Nowadays, computers are around almost everywhere allowing us to plot functions whenever needed. Nevertheless, being able to imagine how functions look like and sketch them is a useful ability because it gives us a better intuition for what those functions do. There are some simple tricks for imagining and drawing functions which we briefly present in this section. They can basically be classified into two categories. The first is for functions that are transformations of certain basic functions which one usually knows by, like the exponential function, sine and cosine function or easy polynomials like the parabola. The second is for functions that are compositions of known basis functions.

1.1.7.1 Adapting Functions

Everyone knows how to sketch the parabola $f(x) = x^2$: It is opened upwards, symmetric, equals one for $x = \pm 1$ and diverges to infinity for $x \to \pm\infty$. But what about the function $f(x) = -\frac{1}{2}(x-2)^2 + 5$? We will see that it is easy to adapt $f(x) = x^2$ to make it look like $f(x) = -\frac{1}{2}(x-2)^2 + 5$. The first question, we have to answer, is in which order we want to introduce the changes to x^2 to

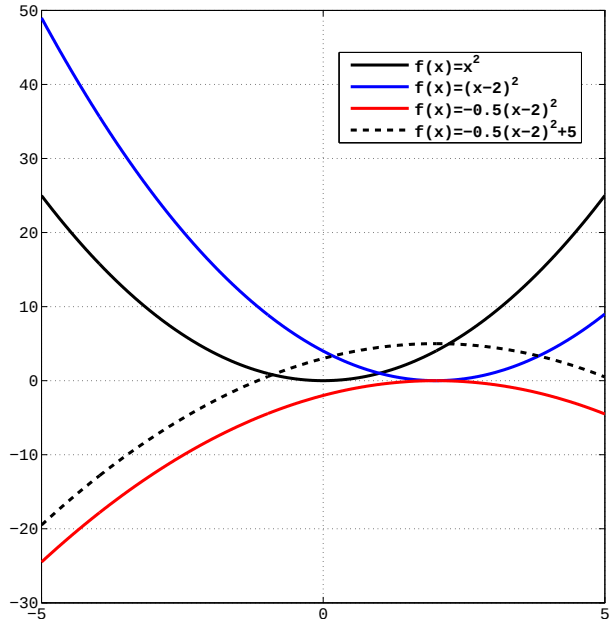


Figure 1.6: Different steps of transforming $f(x) = x^2$ into $f(x) = -\frac{1}{2}(x-2)^2+5$.

transform it into $-\frac{1}{2}(x-2)^2+5$. The answer is: We do that in exactly that order in which we would compute the result of $-\frac{1}{2}(x-2)^2+5$. This means, we first subtract 2, then square the result, then premultiply $-\frac{1}{2}$ and finally add 5. If we would not do that we would end up with a different function since we would violate calculation rules at some point in the process.

So let us start to transform x^2. You can follow the different steps graphically in Figure 1.6.As we just mentioned, the first step is to transform x^2 into $(x-2)^2$. Here, we get our first rule: The graph of $f(x-a)$ is simply the graph of $f(x)$ shifted by a to the right. Of course, if a is negative, shifting by a becomes a shift to the left. Why is that the case? If you think about it, $x-a$ can also be seen as shifting the whole x-axis by $-a$, i.e. a to the left. This, however, is equivalent to shifting f to the right by a. Applied to your example, this means that we have to shift x^2 to the right by 2 in order to obtain the graph of $(x-2)^2$.

The next step is to include the factor $-\frac{1}{2}$. We already know the graph of $(x-2)^2$, how does the graph of $-\frac{1}{2}(x-2)^2$ look like? Well, premultiplying -1 surely reflects the graph along the x-axis. The factor $\frac{1}{2}$ squeezes the result. For example, y-values that used to be -1 are now $-\frac{1}{2}$, y-values that used to be -2 are now -1, and so on.

Now, we are almost there. The last step is to include the additive constant $+5$. This is easy: It simply shifts the graph of $-\frac{1}{2}(x-2)^2$ upwards by 5. This is it! We arrived at the graph of $-\frac{1}{2}(x-2)^2+5$ by a few simple adaptation rules for the graph of x^2. With this few simple rules, you can already sketch a

decent amount of functions.

Rules for adapting functions

- The graph of $f(x-a)$ is simply the graph of $f(x)$ shifted by a.
- The graph of $-f(x)$ is the graph of $f(x)$ flipped along the x-axis.
- The graph of $a \cdot f(x)$ is the graph of $f(x)$ stretched ($a > 1$) or squeezed ($0 \leq a < 1$) by a.
- The graph of $f(x) + b$ is the graph of $f(x)$ shifted by b along the y-axis.

1.1.7.2 Compositions of Functions

Sketching compositions of functions is a little bit more art, but for a lot of examples it is not so difficult. As an example, we use the function $f(x) = \exp\left(-(x-2)^2\right)$ which is just a nicer way of writing $f(x) = e^{-(x-2)^2}$. The rules from above are not sufficient to sketch this function. There is, however, one rule that we can use: If we know the graph of $f(x) = \exp\left(-x^2\right)$, we know that we arrive at $f(x) = \exp\left(-(x-2)^2\right)$ by shifting the graph to the right by 2. Therefore, we look at how to sketch the graph of $f(x) = \exp\left(-x^2\right)$ in the following.

The first step, you can always to is to check whether there are function values which are easy to compute and help drawing the graph. Usually, it is a good idea to look at the behavior of $f(x)$ at $x = 0$, $x = \pm 1$ and what $f(x)$ does if x goes to $\pm\infty$. In our case, the interesting cases are $x = 0$ and $x \to \pm\infty$. The position $x = 0$ is interesting since anything to the power of 0 equals 1. Therefore $f(0) = 1$. When we ask how $f(x)$ behaves for $x \to \pm\infty$ we can first observe that the behavior will be the same at both sides since the squaring operation cancels out the sign. So let us look at what happens when $x \to \infty$. Let us advance step by step, just as before. First, when $x \to \infty$, surely we will get $x^2 \to \infty$ as well. By that, we can immediately see that $-x^2 \to -\infty$. Therefore, we only need to know what happens to $\exp(z)$ if its argument z assumes a very large negative value. We can rewrite the problem a bit by using one of the calculation rules for exponentials: $e^{-x} = \frac{1}{e^x}$. Now, the answer should be easy. If $-x^2 \to -\infty$, there will be a large value in the denominator and, therefore, $f(x) = \exp\left(-x^2\right)$ will approach zero. We can even say a bit more, namely that it will approach zero from above since $\exp(z)$ can never become negative if $z \in \mathbb{R}$.

In a similar manner, you can sketch functions of the form $f(x) = g(x) + h(x)$ or $f(x) = \frac{g(x)}{h(x)}$. First, find a few points where the function value is easy to compute. Then check what happens if x approaches points, where one of the functions goes to zero or infinity. Then the question is usually, which of the functions "wins", i.e. which approaches zero or infinity faster. For example, $f(x) = x^2 - x$ will definitely diverge to infinity for $x \to \infty$ since x^2 grows much faster than $-x$ is able to drag it into the negative side. Similarly $f(x) = \frac{x}{\exp(x)}$ will approach zero for $x \to \infty$ since $\exp(x)$ grows faster than x does.

Drawing compositions of functions takes a bit of practice, but is a useful tool for understanding how functions look like and what they do.

1.2 Basic Calculus

In this section we will cover basic rules for calculating derivatives and simple integrals. Along with introducing the different rules we will also introduce the derivatives of all the functions covered in the section before. In order to keep the equations simple, we will from now on leave out the brackets for functions like sin, cos, log, ... as long as the argument of the function is clear from the context.

1.2.1 Derivatives

The derivative $f'(x_0) = \frac{df}{dx}(x_0)$ (denoted with a "$'$", $\frac{df}{dx}$, or $\frac{d}{dx}f$) of a function $f(x)$ has two intuitive meanings:

1. It measures the rate of change of a function at a certain location x_0.
2. It is the slope of the line touching the function f at a the point $(x_0, f(x_0))$. This line is called *tangent line* or simply *tangent*.

Using the first intuition, $f'(x_0)$ is an approximation of how the function value of $f(x)$ changes when going from x_0 to $x_0 + 1$. If f is a linear function, the approximation will be exact, that means f will change exactly by $f'(x_0)$. If f is not linear we will make some error, but we still can use $f'(x)$ to construct the best linear approximation of f at x_0. But first, we will start with an example of the exact case.

Example Consider the linear function $f(x) = 3x$. According to the first intuition, the derivative $f'(x)$ is the rate of change of f, i.e. the change in the function value of f divided by the change in the value of x. Consider two arbitrary points x_0 and x_1. The rate of change is then given by:

$$\begin{aligned} f'(x) &= \frac{f(x_1) - f(x_0)}{x_1 - x_0} \\ &= \frac{3x_1 - 3x_0}{x_1 - x_0} \\ &= \frac{3(x_1 - x_0)}{(x_1 - x_0)} \\ &= 3. \end{aligned}$$

Since any linear function can be written as $f(x) = ax$, we just showed that the first derivative $f'(x)$ of a linear function does not depend on x. This means that it is the same everywhere. This is what we expect intuitively from a line. Secondly we verified the second intuition for linear functions. The first derivative at a point x_0 is the slope of the tangent line of f at $(x_0, f(x_0))$. Since the tangent line is simply the linear function itself, the first derivative $f'(x)$ is the slope of the linear function.

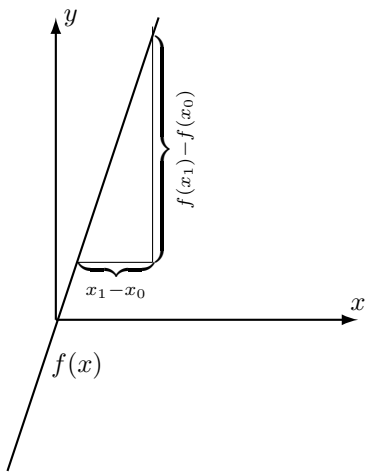


Figure 1.7: Geometrical picture for calculating derivatives of linear functions.

◁

The situation changes when we consider arbitrary functions $f(x)$. In that case the value of the rate of change, given by the quotient $\frac{f(x_1)-f(x_0)}{x_1-x_0}$, will depend on the choices of x_1 and x_0. This raises the question how we could define the rate of change in a meaningful way? The second intuition can help us here: In order to get the first derivative at a point x_0 we approximate $f(x)$ at x_0 with a line and define the first derivative to be its slope. Since we are merely interested in computing the slope of that line, we do not need to compute the full line equation but start with the slope right away. Remember, given a line $g(x) = ax+t$, we can compute its slope via $\frac{g(x_1)-g(x_0)}{(x_1-x_0)} = \frac{ax_1+t-ax_0-t}{(x_1-x_0)} = a$ where x_1 and x_0 are two arbitrary points. Now imagine we have a line $g(x) = ax + t$ that contains the two points $(x_0, f(x_0))$ and $(x_1, f(x_1))$ (see Figure 1.8). In order to compute its slope we do not need the full line equation. Instead we can simply use the quotient from above and compute

$$\begin{aligned} a &= \frac{g(x_1) - g(x_0)}{x_1 - x_0} \\ &= \frac{f(x_1) - f(x_0)}{x_1 - x_0}. \end{aligned}$$

The slope a of this line is not yet the first derivative since g contains $(x_0, f(x_0))$ *and* $(x_1, f(x_1))$. This means, that (in most cases) it will intersect with f and

not *touch* it at $(x_0, f(x_0))$. However, we can achieve this goal by moving x_1 close to x_0. Once it is infinitely, called *infinitesimally*, close to x_0, $a = \frac{f(x_1)-f(x_0)}{x_1-x_0}$ will be the slope of the tangent line, i.e. $a = f'(x_0)$. Mathematically this is expressed in terms of a limit. We do not go into the details of limits here, but merely demonstrate, how the derivative of a function f at x_0 is defined. The rate of change with a infinitesimal close point $x_1 = x_0 + h$ can be written in terms of limits as

$$f'(x) = \lim_{h \to 0} \frac{f(\overbrace{x_0 + h}^{=x_1}) - f(x_0)}{\underbrace{h}_{=x_1 - x_0}}.$$

The "lim" expresses that we let h come infinitesimally close to zero and therefore $x_1 = x_0 + h$ infinitesimally close to x_0. The expression $\lim_{h\to 0} \frac{f(x_0+h)-f(x_0)}{h}$ is called *differential quotient* of f. Note that, in order to obtain a unique notion of a derivative at a point x_0, the direction from which $x_1 = x_0 + h$ approaches x_0 should not matter. This means that h could be negative (x_1 approaches x_0 from the left) or positive (x_1 approaches x_0 from the right).

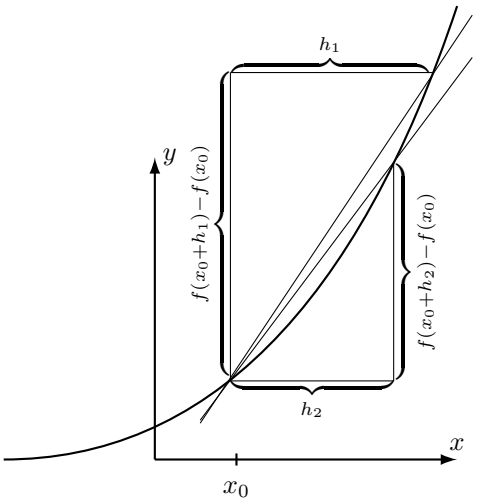


Figure 1.8: Geometrical picture of the differential quotient for arbitrary functions. As h becomes smaller, slope of the line through $(x_0, f(x_0))$ and $(x_0 + h, f(x_0 + h))$ converges to the derivative of f at x_0.

Looking at Figure 1.8, we can see that not every function has a derivative at any point x_0. If the function makes a step at x_0 such that there is a gap

between the function value at x_0 and the function value at x_0 plus or minus an infinitesimal h, then the value of the differential quotient at x_0 depends on the direction from which $x_0 + h$ approaches x_0 and the derivative would not be unique. Therefore, functions are only differentiable at points where the function does not make such a step. This property is expressed in the concept of *continuity*.

Definition (Continuous Function) A function f is said to be *continuous at a point* x_0 if

$$\lim_{h \to 0} f(x_0 + h) = \lim_{h \to 0} f(x_0 - h) \quad = \quad f(x_0),$$

i.e. if getting infinitesimal close to x_0 (no matter from which side) implies getting infinitesimal close to $f(x_0)$.

A function f is said to be *continuous*, if it is continuous in every point of its domain.

$\Diamond$

If a function is continuous, we can calculate the value of the differential quotient $\lim_{h \to 0} \frac{f(x_0+h)-f(x_0)}{h}$. If the function is not continuous, the derivative at that point is classified as *not defined*. In a similar fashion, functions that have a kink at x_0 give rise to an undefined derivative at x_0. The reason is that because of the kink approaching x_0 from the left yields a different slope than approaching x_0 from the right. For example $f(x) = |x| = \text{abs}(x)$ is not differentiable at $x_0 = 0$. At all other points, however, it is. We will see how to differentiate $f(x) = |x|$ in an example below.

However, if we can find a linear function that with its slope equal to the value of the differential quotient, no matter if h approaches zero from the left ($h < 0$) or from the right ($h > 0$), the function is called *differentiable*.

Definition (Differentiable Function) A function f is said to be *differentiable at a point* x_0 if there exists a linear function $L(x)$ such that

$$L(x_0) - \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h} \quad = \quad 0,$$

no matter from which side h approaches zero.

A function f is said to be *differentiable*, if it is differentiable in every point of its domain.

$\Diamond$

Fortunately, we do not need to go through the tedious process of calculating the value of the differential quotient every time. There are easier ways to compute the slope of the tangent at a point x_0. In the following we will review the

most important rules how to compute derivatives. In most cases those rules are sufficient to compute derivatives of functions that you are dealing with.

We start with the simplest rule: Calculating the derivatives of powers. If $f(x) = ax^b$, then the derivative of f is given by

$$f'(x) \quad = \quad abx^{b-1}.$$

As we can see, the rule is fairly easy: We just premultiply the exponent b, subtract 1 from the exponent of x and leave the factor a untouched. Before looking at a small example, we just introduce another rule: The derivative of a sum equals the derivatives of the single terms:

$$f(x) = g_1(x) + g_2(x) \Rightarrow f'(x) = g_1'(x) + g_2'(x)$$

Or in general:

$$f(x) = \sum_{k=1}^{n} g_k(x) \quad \Rightarrow \quad f'(x) = \left(\sum_{k=1}^{n} g_k(x) \right)' = \sum_{k=1}^{n} g_k'(x).$$

Here, g_k are arbitrary differentiable functions.

Example Assume that $f(x) = \frac{1}{2}x^2 + 5x + 10$. In order to apply the rule, we must rewrite the constant 10. Since we already know that $x^0 = 1$ we can write $10 = 10x^0$ and f becomes $f(x) = \frac{1}{2}x^2 + 5x + 10x^0$. Now we can apply our rules step by step:

$$\begin{aligned} f'(x) &= \left(\frac{1}{2}x^2 + 5x + 10x^0 \right)' \\ &= \left(\frac{1}{2}x^2 \right)' + (5x)' + (10x^0)' \\ &= x + 5. \end{aligned}$$

As we can see, the last term vanishes since premultiplying 0 cancels the whole term. Since we can write any constant b that does not depend on x as $b = bx^0$, constants always vanish when calculating the derivative.

◁

We can generalize our example to arbitrary polynomials:

$$f(x) = \sum_{k=0}^{n} a_k x^k \quad \Rightarrow \quad f'(x) = \sum_{k=1}^{n} a_k k x^{k-1}.$$

Note, that the index k of $f'(x)$ starts at one instead of zero. This is because the last term $a_0 x^0 = a_0$ vanishes when differentiating.

We can also use this rule to calculate the derivative of roots and ratios.

Examples

1. Let f be $f(x) = \sqrt{x}$. Since we can write f as $f(x) = x^{\frac{1}{2}}$, the derivative of f is given by $f'(x) = \frac{1}{2}x^{-\frac{1}{2}} = \frac{1}{2\sqrt{x}}$.

2. Let f be $f(x) = \frac{1}{x^a}$. Since we can write f as $f(x) = x^{-a}$, the derivative is given by $f'(x) = -ax^{-a-1} = -\frac{a}{x^{a+1}}$

◁

As those examples show, we can already calculate the derivative of a fair amount of functions. However, so far we cannot differentiate functions that are compositions of other functions. For example, we cannot get $f'(x)$ for function $f(x) = \sqrt{\frac{1}{2}x^2 + 5x + 10}$ with our current set of rules, since $f(x) = g_2(g_1(x))$ is the composition of $g_2(y) = \sqrt{y}$ and $y = g_1(x) = \frac{1}{2}x^2 + 5x + 10$. Therefore, we introduce a new rule, called *chain rule:* Let f be $f(x) = g_2(g_1(x))$ with two arbitrary differentiable functions g_1 and g_2, then the derivative $f'(x)$ is given by

$$f'(x) \quad = \quad g_2'(g_1(x)) \cdot g_1'(x).$$

According to this rule we first differentiate $g_2(y)$ while treating $y = g_1(x)$ as a variable on its own. After that, we multiply the result with the derivative $g_1'(x)$ of g_1 with respect to g_1. Let us illustrate this rule by differentiating $f(x) = \sqrt{\frac{1}{2}x^2 + 5x + 10}$.

Example Let f be $f(x) = \sqrt{\frac{1}{2}x^2 + 5x + 10}$. As already mentioned before, f is the composition of $g_2(y) = \sqrt{y}$ and $y = g_1(x) = \frac{1}{2}x^2 + 5x + 10$. We already calculated the derivative of those functions:

$$\begin{aligned} g_2'(y) &= \frac{1}{2\sqrt{y}} \\ g_1(x) &= x + 5. \end{aligned}$$

Applying the chain rule therefore yields

$$\begin{aligned} f'(x) &= g_2'(g_1(x)) \cdot g_1'(x) \\ &= \frac{1}{2\sqrt{\frac{1}{2}x^2 + 5x + 10}} \cdot g_1'(x) \\ &= \frac{1}{2\sqrt{\frac{1}{2}x^2 + 5x + 10}} \cdot (x + 5). \end{aligned}$$

◁

Now, there is only one rule left: How to differentiate products $f(x) = g_1(x) \cdot g_2(x)$ of two differentiable functions g_1 and g_2. If f is the product of two functions g_1 and g_2, the derivative is given by

$$f'(x) = g_1'(x) \cdot g_2(x) + g_1(x) \cdot g_2'(x).$$

This rule is called *product rule*. It tells us to first differentiate $g_1(x)$ and multiply the result with $g_2(x)$, then do it the other way round and sum the results in the end. Again, an example will illustrate this rule.

Example Let f be $f(x) = (x+1)(\frac{1}{2}x^2 + 5)$. Apparently, f is the product of $g_1(x) = (x+1)$ and $g_2(x) = (\frac{1}{2}x^2 + 5)$. It is easy to calculate their derivatives:

$$\begin{aligned} g_1'(x) &= 1 \\ g_2'(x) &= x. \end{aligned}$$

Therefore, by applying the product rule we get the derivative of f:

$$\begin{aligned} f'(x) &= (x+1)' \cdot \left(\frac{1}{2}x^2 + 5\right) + (x+1) \cdot \left(\frac{1}{2}x^2 + 5\right)' \\ &= \frac{1}{2}x^2 + 5 + (x+1)x \\ &= \frac{3}{2}x^2 + x + 5. \end{aligned}$$

In this particular case, we can check the rule by expanding

$$\begin{aligned} f(x) &= (x+1)(\frac{1}{2}x^2 + 5) \\ &= \frac{1}{2}x^3 + \frac{1}{2}x^2 + 5x + 5. \end{aligned}$$

Using the rule for polynomials, yields:

$$\begin{aligned} f'(x) &= \left(\frac{1}{2}x^3 + \frac{1}{2}x^2 + 5x + 5\right)' \\ &= \frac{3}{2}x^2 + x + 5, \end{aligned}$$

which is the same results as the one from the product rule.

◁

In most cases, a fourth rule for differentiating quotients of functions is specified. However, we can derive the *quotient rule* from the rules we already know. Before stating the general quotient rule, we look at a small example.

Example Let f be $f(x) = \frac{(x+1)}{(\frac{1}{2}x^2+5)}$. For calculating the derivative of f we reformulate it first into $f(x) = (x+1)(\frac{1}{2}x^2+5)^{-1}$. Now we can apply the product rule and the chain rule to differentiate f. Let us look at the calculation step by step:

$$\begin{aligned}
f'(x) &\overset{\text{Product Rule}}{=} (x+1)' \left(\frac{1}{2}x^2+5\right)^{-1} + (x+1)\left(\left(\frac{1}{2}x^2+5\right)^{-1}\right)' \\
&\overset{\text{Chain Rule}}{=} (x+1)' \left(\frac{1}{2}x^2+5\right)^{-1} - (x+1)\left(\frac{1}{2}x^2+5\right)^{-2}\left(\frac{1}{2}x^2+5\right)' \\
&= \frac{(x+1)'}{\left(\frac{1}{2}x^2+5\right)} - \frac{(x+1)\left(\frac{1}{2}x^2+5\right)'}{\left(\frac{1}{2}x^2+5\right)^2} \\
&= \frac{(x+1)'(\frac{1}{2}x^2+5) - (x+1)\left(\frac{1}{2}x^2+5\right)'}{\left(\frac{1}{2}x^2+5\right)^2} \\
&= \frac{-\frac{1}{2}x^2 - x + 5}{\left(\frac{1}{2}x^2+5\right)^2}.
\end{aligned}$$

◁

In general, it might be easier to calculate all the derivatives first. In this example, however, the derivatives were only resolved at the end to show an important aspect: When looking at the fourth line of the calculation, we see that it consists exclusively of terms that already appeared in the product. We can generalize this, to get the quotient rule: Let f be a quotient of two functions $f(x) = \frac{g_1(x)}{g_2(x)}$. By rewriting $f(x) = g_1(x) \cdot g_2(x)^{-1}$ and applying the product rule, the chain rule and the rule for powers, we arrive at

$$f'(x) = \frac{g_1'(x)g_2(x) - g_1(x)g_2'(x)}{g_2(x)^2}.$$

Here is a summary of all the rules we just saw:

Differentiation Rules

- Derivatives of constant functions: The derivative of any constant function $f(x) = a$ is $f'(x) = 0$.
- Summation Rule: Let $f(x) = \sum_{k=1}^{n} g_k(x)$ a sum of arbitrary differentiable functions. Then $f'(x)$ is given by:

$$f'(x) = \sum_{k=1}^{n} g_k'(x).$$

- Power Rule: Let f be $f(x) = ax^b$, then $f'(x)$ is give by:

$$f'(x) = abx^{b-1}.$$

- Chain Rule: Let $f(x) = g_2(g_1(x))$ be a composition of arbitrary differentiable functions. Then $f'(x)$ is given by:

$$f'(x) = g_2'(g_1(x)) \cdot g_1'(x).$$

- Product Rule: Let $f(x) = g_1(x)g_2(x)$ be a product of arbitrary differentiable functions. Then $f'(x)$ is given by:

$$f'(x) = g_1'(x) \cdot g_2(x) + g_1(x) \cdot g_2'(x).$$

- Quotient Rule: Let $f(x) = \frac{g_1(x)}{g_2(x)}$ be a quotient of two arbitrary differentiable functions. Then $f'(x)$ is given by:

$$f'(x) = \frac{g_1'(x)g_2(x) - g_1(x)g_2'(x)}{g_2(x)^2}.$$

We conclude this section by stating derivatives of important functions that you cannot compute using the rules above. Since they occur quite often, you should know them by heart.

Derivatives of important functions

- $f(x) = \sin(x) \Rightarrow f'(x) = \cos(x)$
- $f(x) = \cos(x) \Rightarrow f'(x) = -\sin(x)$
- $f(x) = e^x \Rightarrow f'(x) = e^x$
- $f(x) = \ln(x) = \log_e(x) \Rightarrow f'(x) = \frac{1}{x}$

Example

Let us calculate the derivative of a little bit more advanced case: $f(x) = |x| = \text{abs}(x)$. Figure 1.9 show the graph of the function.

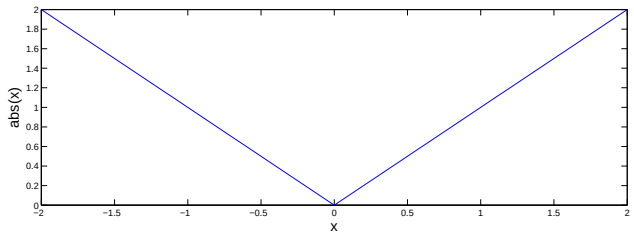


Figure 1.9: Graph of the function $f(x) = |x|$.

We can either use the definition of $|x|$ as a piecewise defined function, or use the fact that $f(x) = \sqrt{x^2}$. In both cases the derivative is $f'(x) = -1$ for $x < 0$ and $f'(x) = 1$ for $x > 0$. But what happens at $x_0 = 0$. If we use the piecewise definition

$$|x| = \begin{cases} -x & x \leq 0 \\ x & x > 0 \end{cases},$$

then the value of the differential quotient $\lim_{h\to 0} \frac{|x_0+h|-|x_0|}{h}$ is -1 if we approach $x_0 = 0$ from the left (i.e. $h < 0$) and 1 if we approach $x_0 = 0$ from the right (i.e. $h > 0$). Since there cannot be a linear function which has the slope -1 and 1 at the same time, the derivative is not defined at $x_0 = 0$. A similar thing happens when we use $|x| = \sqrt{x^2}$. Since $\left(\sqrt{x^2}\right)' = \frac{x}{\sqrt{x^2}}$ we cannot choose x to be zero since this would make the denominator zero and the fraction would not be defined. Therefore the derivative of $f(x) = |x|$ is given by

$$f(x) = \begin{cases} -1 & x < 0 \\ \text{undefined} & x = 0 \\ 1 & x > 0 \end{cases}.$$

◁

Example

There is an easy numerical way to check and compute derivatives with matlab based on the differential quotient:

$$f'(x) = \lim_{h\to 0} \frac{f(\overbrace{x_0+h}^{=x_1}) - f(x_0)}{\underbrace{h}_{=x_1-x_0}}.$$

The idea is that $\frac{\overbrace{f(x_0+h)}^{=x_1}-f(x_0)}{\underbrace{h}_{=x_1-x_0}}$ already sufficiently approximates $f'(x)$ for really small h. This means, that you choose at a set of closely spaced points $x_1, ..., x_i, x_{i+1}, ..., x_n$ and compute the function values $f(x_1), ..., f(x_i), f(x_{i+1}), ..., f(x_n)$ at those points and compute $f'(x_i) \approx \frac{f(x_{i+1})-f(x_i)}{x_{i+1}-x_i}$. Fortunately, there is a built-in matlab function that takes the differences for you. Here is an example for how to compute the numerical derivative of $f(x) = \sin(x)$:

```
>> h = .001; x = 0:h:2*pi; % define h and the base points
>> f = sin(x); % sample the function at x
>> df = diff(f)/h; % compute the differences
>> plot(x,f,'k'), hold on; % plot the function
>> plot(x(1:end-1),df,'r'); % plot the approximation
>> plot(x(1:end-1),cos(x(1:end-1)),'g'); % plot the true derivative
```

You surely noticed that we were only able to compute $f'(x)$ for all but the last x. The reason is, that there is no x_{n+1} and $f(x_{n+1})$ that we could have used for computing $f'(x_n)$. However, there is also a little more involved matlab function that computed the derivatives at all base points. Using that function the example becomes:

```
>> h = .001; x = 0:h:2*pi; % define h and the base points
>> f = sin(x); % sample the function at x
>> dfdh = gradient(f,h); % compute the first derivative
>> plot(x,f,'k'), hold on; % plot the function
>> plot(x,dfdh,'r'); % plot the approximation
>> plot(x,cos(x),'g'); % plot the true derivative
```

You can (and should) use this function to check derivatives that you computed analytically.

◁

1.2.2 Higher-Order Derivatives

The term *higher-order derivatives* comprises the result of iterated differentiation. Since we can treat $f'(x)$ as a function on its own, there is really nothing new here. If $f'(x)$ is differentiable, then $f(x)$ twice differentiable. This can be extended to higher-orders: The third order derivative which is written $f'''(x)$ or sometimes $f^{(3)}(x)$ is just the derivative of the second order derivative.

The second derivative, however, has a special meaning. Geometrically, it is the slope of the slope, i.e. how do the linear approximations at $f(x)$ vary with x. This give us a notion of *curvature* of $f(x)$. Intuitively, if the linear approximations at $f(x_0)$ bend away very quickly in the area around x_0, then $f(x)$ must have a high curvature at x_0.

The geometric intuition behind higher-order derivatives is somewhat harder. In particular, our visual system is not very good at judging higher-order derivatives. By visual inspection, it is pretty hard to say if e.g. the fourth derivative is positive or negative.

Example

Suppose that the function $f(t)$ is the position of an object in space at time t. Then, $f'(t)$ gives us the rate at which the object changes its position, i.e. the object's speed at time t. The second order derivative $f''(t)$ gives us the rate at which the object changes its speed, which is simply the acceleration of the object. Similarly, if $f(t)$ is the concentration of Ca^{2+}-ions in a neuron at time t, then $f'(t)$ gives you the rate at which ions enter or leave the cell at time t, and $f''(t)$ indicates whether this rate is getting bigger or smaller.

◁

1.2.3 Finding Maxima/Minima of a Function

Derivatives can be used to find global minima or maxima of functions. If $f(x)$ has a local maximum or minimum at a point x_0, the tangent line is horizontal. This means that the slope of the tangent and, therefore, the derivative of the function at that point x_0 must be $f'(x_0) = 0$. Mathematically speaking, $f'(x_0) = 0$ is a *necessary condition* for the function f having a maximum or a minimum, i.e. "x_0 is maximum/minimum" $\Rightarrow f'(x_0)$. However, it can happen, that $f'(x_0) = 0$ but not x_0 is not a maximum or a minimum. Those points are called saddle-points. They arise for example, when $f(x)$ increases, becomes more and more flat when approaching x_0, is completely flat at x_0, and increases again afterwards. Therefore, since $f'(x_0) = 0$ does not imply a maximum or a minimum, we must find a condition that does that, i.e. we need to find a *sufficient condition,* one for which "condition"⇒"x_0 is maximum/minimum".

Since we already know that we only need to look at points x_0 where $f'(x_0) = 0$, we can think about what must happen for x_0 to be—say—a maximum. If x_0 is a maximum, then the slope of tangent lines of points to the right of x_0 must become more and more negative. At the same time the slope of tangent lines at points left to x_0 must become less and less negative and approach zero at x_0. If you think about it, this is exactly the same as requiring $f''(x_0) < 0$. Similarly, $f'(x_0) = 0$ and $f''(x_0) > 0$ imply a minimum of f at x_0. If $f'(x_0) = 0$, $f''(x_0) = 0$, but $f'''(x_0) \neq 0$ then f has a saddle-point at x_0. In general, if the first n derivatives are zero and the $(n+1)$th derivative $f^{(n+1)}(x_0)$ is not equal to zero, then x_0 is a minimum, if $n+1$ is even and $f^{(n+1)}(x_0) > 0$, a maximum if $n+1$ is even and $f^{(n+1)}(x_0) < 0$, and a saddle-point if $n+1$ is odd and $f^{(n+1)}(x_0) \neq 0$.

Finding Maxima, Minima, and Saddle-Points
Let f be a $(n+1)$ times differentiable function. Then f has a

- *minimum* at x_0 if and only if $f^{(k)}(x_0) = 0$ for $k = 1, ..., n$ and $f^{(n+1)}(x_0) > 0$ with $(n+1)$ even.
- *maximum* at x_0 if and only if $f^{(k)}(x_0) = 0$ for $k = 1, ..., n$ and $f^{(n+1)}(x_0) < 0$ with $(n+1)$ even.
- *saddle-point* x_0 if and only if $f^{(k)}(x_0) = 0$ for $k = 1, ..., n$ and $f^{(n+1)}(x_0) \neq 0$ with $(n+1)$ odd.

It should be pointed out that this procedure only yields *local* maxima, and not necessarily *global* ones.

Example

Consider the function $f(x) = (x-1)^2$. Then $f'(x) = 2(x-1)$ and $f''(x) = 2$. In order to find candidates for a maximum or a minimum, we set $f'(x) = 0$ and solve for x:

$$\begin{aligned} f'(x) = 0 &\Leftrightarrow 2(x-1) = 2 \\ &\Leftrightarrow x = 1. \end{aligned}$$

Since $f''(x) > 0$ for all x (therefore also for $x = 1$), $x = 1$ must be a minimum of f.

$\lhd$

Example: Estimating the rate of a Poisson distribution

A probabilistic model for a neuron that generates completely random spike trains with no temporal structure is the homogeneous *Poisson process*. Since there is no temporal structure, it serves as a baseline model that other—more advanced—models for spike trains can compare to. You can generate spike trains from that model as follows: Bin the time axis into sufficiently small bins (say $1ms$). For each bin you randomly place a spike with a certain small probability. The matlab code looks like this

```
>> t = [0:0.001:10];
>> p = 0.05; % spiking probability in each bin
>> ind = find( rand(size(t)) <= p); % sample spike times
>> s = zeros(size(t)); % generate spike train
>> s(ind) = 1; % insert spikes
>> plot(t,s,'k')
```

of observing k spikes in a time window of length $\Delta t = 1$ is given by the Poisson distribution

$$p(k \text{ spikes}) = \frac{\lambda^k e^{-\lambda}}{k!}$$

or simply

$$p(k) = \frac{\lambda^k e^{-\lambda}}{k!}.$$

λ is called the rate of the Poisson distribution. It tells us how many spike we expect in one second on average. The question that we want to solve in this example is how to estimate λ from a number of observed spike trains. But lets first look at our matlab example again and check, whether our spike counts in one second really follow a Poisson distribution. For that reason, we generate a large number of spike trains of length $1s$, count the spikes in each spike train and look at the empirical histogram. The matlab code looks like this:

```
>> t = [0:0.001:1];
>> p = 0.05; % probability of spike in each bin
>> m = 5000; % number of spike trains
>> U = rand(m,length(t));
>> S = zeros(size(U)); % generate spike train matrix (trains X time)
>> S(U <= p) = 1;
>> figure
>> imagesc(1-S); colormap gray;
>> xlabel('time bins'); ylabel('spike trains')
>> figure, C = sum(S,2); % get spike counts
>> K = [1:max(C)];
>> hist(C,K); % plot histogram
>> hold on
>> lambda = p*length(t);
>> plot(K, m*poisspdf(K,lambda),'r','LineWidth',2); hold off
```

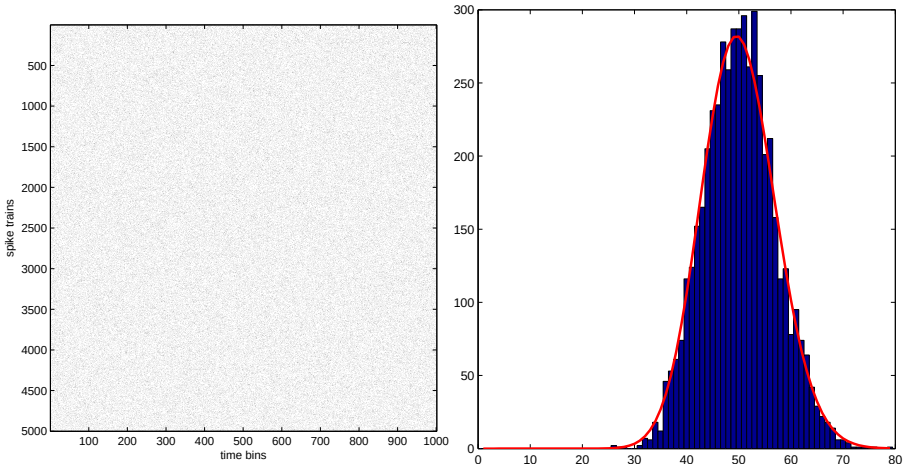


Figure 1.10: Poisson spikes with rate $\lambda = 50.05$.

A typical output of the matlab code fragment is shown in Figure 1.10. Before we see how to estimate λ from the data, let us first spend a thought about what values we expect. The answer is not too difficult. Since we see a spike with probability $p = 0.05$ in each bin, all bins are independent, we expect to see a total number of $\lambda = n_{bins} \cdot p$ spikes per second where n_{bins} is the number of bins per second. Strictly speaking, the rate λ is defined as the number of bins times the probability of spiking per bin in the limit of infinite n_{bins}. The idea is, that the larger the number of bins gets, the lower the probability of observing a spike in a specific bin becomes, i.e. it goes to zero for n_{bins} to infinity. Therefore, the product of both number can be something finite, i.e. the rate λ.

Let us know turn to the question of how to estimate λ. Assume that you are only given the number of spikes $k_1, ..., k_m$ in each of the spike trains of length $1s$. Certainly you also do not know p. However, you do know (or do assume) that the spikes have been generated as described above and therefore the spike counts k must be Poisson distributed. We further assume that each spike train has been generated independently of the others.

Given a distribution and a number of observations, one principled way of estimating the parameters of that distribution (in our case the rate λ) is the *maximum likelihood* method. The idea behind that method is simple: We choose λ such that the probability of our observations is maximized:

$$\hat{\lambda} \quad = \quad \text{argmax}_{\lambda \in \mathbb{R}^+} p(k_1, ..., k_m | \lambda).$$

Since we assumed that the observations are independent from each other, the probability of observing all spike counts jointly, is the product of the probabilities of observing each single spike count:

$$p(k_1, ..., k_m | \lambda) \quad = \quad \prod_{i=1}^{m} p(k_i | \lambda).$$

For each single spike count, we know that k_i is Poisson distributed and therefore

$$\begin{aligned} p(k_1, ..., k_m | \lambda) &= \prod_{i=1}^{m} p(k_i | \lambda) \\ &= \prod_{i=1}^{m} \frac{\lambda^{k_i} e^{-\lambda}}{k_i!}. \end{aligned}$$

Now, we only need to find that maximum of $f(\lambda) = \prod_{i=1}^{m} \frac{\lambda^{k_i} e^{-\lambda}}{k_i!}$ with respect to λ. Unfortunately, taking the derivative of $f(\lambda) = \prod_{i=1}^{m} \frac{\lambda^{k_i} e^{-\lambda}}{k_i!}$ is very complicated. Luckily, there is a simple trick that we can apply: By taking the log of both sides changes the function values, but leaves the position of the maxima unchanged (the reason for that is, that log is a strictly increasing continuous function). However, taking the log makes the right hand side considerably easier

to deal with:

$$\begin{aligned}\log f(\lambda) &= \log\left(\prod_{i=1}^{m}\frac{\lambda^{k_i}e^{-\lambda}}{k_i!}\right)\\ &= \sum_{i=1}^{m}\log\frac{\lambda^{k_i}e^{-\lambda}}{k_i!}\\ &= \sum_{i=1}^{m}\left(\log\lambda^{k_i}+\log e^{-\lambda}-\log k_i!\right)\\ &= \sum_{i=1}^{m}\left(k_i\log\lambda-\lambda-\log k_i!\right).\end{aligned}$$

Let us now compute the maximum of $\sum_{i=1}^{m} k_i\log\lambda-\lambda-\log k_i!$. For that reason, we first compute the first derivative

$$\begin{aligned}\frac{d}{d\lambda}\left(\sum_{i=1}^{m}k_i\log\lambda-\lambda-\log k_i!.\right) &= \sum_{i=1}^{m}\left(k_i\frac{d}{d\lambda}\log\lambda-\frac{d}{d\lambda}\lambda-\frac{d}{d\lambda}\log k_i!\right)\\ &= \sum_{i=1}^{m}\left(k_i\frac{1}{\lambda}-1\right)\\ &= -m+\frac{1}{\lambda}\sum_{i=1}^{m}k_i.\end{aligned}$$

Setting the first derivative to zero and solving for λ yields

$$\begin{aligned}-m+\frac{1}{\lambda}\sum_{i=1}^{m}k_i=0 \quad&\Leftrightarrow\quad \frac{1}{\lambda}\sum_{i=1}^{m}k_i=m\\ &\Leftrightarrow\quad \lambda=\frac{1}{m}\sum_{i=1}^{m}k_i.\end{aligned}$$

In order to check that it is really a maximum, we compute the second derivative

$$\frac{d}{d\lambda}\left(-m+\frac{1}{\lambda}\sum_{i=1}^{m}k_i\right) = -\frac{1}{\lambda^2}\sum_{i=1}^{m}k_i.$$

Since rates are always positive (i.e. $\lambda > 0$) the second derivative is always smaller than zero and, therefore, $\hat{\lambda}=\frac{1}{m}\sum_{i=1}^{m}k_i$ is really a maximum. It also has a very intuitive interpretation. The maximum likelihood estimate for the rate of a Poisson distribution is the mean of the observed spike counts. In that sense it is also not surprising, that the mean of a Poisson distribution is λ.

You should include the maximum likelihood estimate in the matlab example from above and compare it to the expected rate $\lambda = p\cdot n_{bins}$. The code fragment, you have to add, is:

```
>> lambda_est = mean(C);
```

◁

1.2.4 Approximating Functions Locally by Lines and Polynomials

Example: Finding a linear approximation to the function sin(x) for x near 0

Suppose $f(x) = \sin(x)$. If we want to approximate $f(x)$ at $x_0 = 0$ by a line, i.e. a function of the form $g(x) = wx + b$, such that $g(x)$ is a good approximation for x near 0. From the definition of the derivative, we saw that the derivate of a function at x_0 is really the slope of a linear approximation of $f(x)$ at x_0. Therefore we can just compute $f'(x) = \cos(x)$ and evaluate it at $x_0 = 0$. This yields the value of w, since w if the derivative of $g(x)$ and we want the derivatives of the function f and its linear approximation g to be the same. In our case,

$$w \quad = \quad f'(x_0) = \cos(0) = 1.$$

Now, we must ensure, that $g(x_0) = f(x_0)$. We do that by adjusting b. The offset b shifts the function horizontally. If we want g to have the same value at x_0 as f, we just need to add $f(x_0)$. Therefore $b = f(x_0)$. Since $f(x_0) = \sin(0) = 0$, we set $b = 0$ and get $g(x) = wx$ as linear approximation of $f(x)$ at x_0. Figure 1.11 plots the linear approximation.

◁

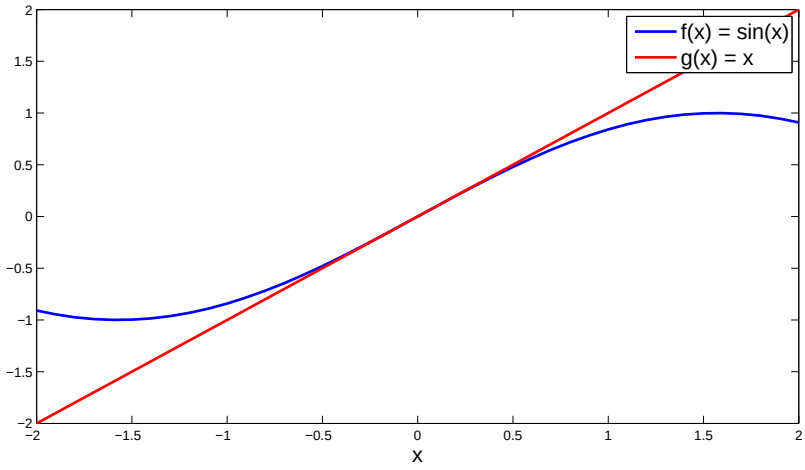


Figure 1.11: Graph of the function $f(x) = \sin(x)$ and its linear approximation $g(x) = x$ at $x_0 = 0$.

In the example, we implicitly used that the function is approximated at $x_0 = 0$ when computing the offset b. In general, if $x_0 \neq 0$, we cannot simply use $b = f(x_0)$, since

$$\begin{aligned} g(x_0) &= wx_0 + b \\ &= f'(x_0)x_0 + f(x_0) \\ &\neq f(x_o) \text{ for } w, x_0 \neq 0. \end{aligned}$$

How can we compute the offset b in the general case $x_0 \neq 0$? We can answer this question by some geometrical thoughts. First of all, imagine we set $b = 0$. The approximating function $g(x) = wx = f'(x_0) \cdot x$ would be a true linear function that had the same slope as f at x_0 but not the same function value $g(x_0) = wx_0 \neq f(x_0)$. Since we know that b lifts g along the y-axis we need to find out by how much we must lift it in order to obtain $g(x_0) = f(x_0)$. The answer is: We need to lift it by the difference between the true function value $f(x_0)$ and the linear function $g(x_0) = wx_0$ that has the same slope but the wrong offset, i.e. $b = f(x_0) - wx_0$. By substituting $w = f'(x_0)$ and $b = f(x_0) - wx_0$ in the general line equation, we end up with

$$\begin{aligned} g(x) &= f'(x_0)x + f(x_0) - f'(x_0)x_0 \\ &= f(x_0) + f'(x_0)(x - x_0) \end{aligned}$$

as the best "linear" approximation (it is not linear, is is a line) of f at x_0.

The quality of the approximation depends, of course, on the properties of $f(x)$. Clearly, in the example above, the approximation is really bad for e.g. $x = \pi$.

Example

The linear approximation to $f(x) = e^x$ at $x_o = 0$ is given by

$$\begin{aligned} g_0(x) &= f(x) + f'(x_o)(x - x_o) \\ &= e^0 + e^0 \cdot x \\ &= 1 + x. \end{aligned}$$

The approximation at $x_o = 1$ is given by:

$$\begin{aligned} g_1(x) &= f(x) + f'(x_o)(x - x_o) \\ &= e^1 + (x - 1) \cdot e^1 \\ &= xe. \end{aligned}$$

Figure 1.12 shows the graphs of these functions.

$\triangleleft$

As we can see in figure 1.12, the linear approximations to e^x are not very good. Can we find a better approximation by using a quadratic approximation, i.e. one that uses a polynomial of degree 2?

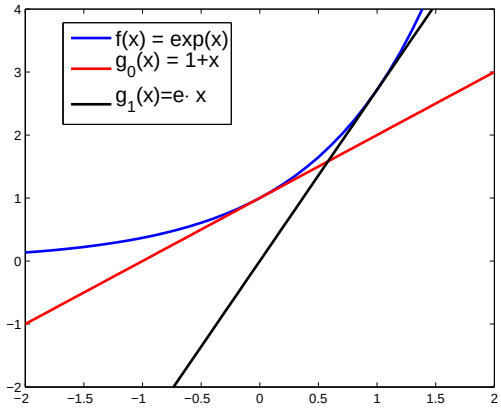


Figure 1.12: Graph of the functions $f(x) = \exp(x)$ and its linear approximation $g_0(x) = 1 + x$ and $g_1(x) = ex$ at $x_0 = 0$ and $x_0 = 1$.

The answer is yes. We simply add a quadratic term to our approximating function g. So far g had the form $g(x) = w(x - x_0) + b$. To turn it into an quadratic approximation, we need to add a term, that is quadratic in $(x - x_0)$. For reasons that will become clear soon this additional term is $\frac{1}{2}v(x - x_0)^2$ yielding $g(x) = \frac{1}{2}v(x-x_0)^2 + w(x-x_0) + b$. The additional work we have to do in comparison to a simple linear approximation is to determine the value of v. Analogously to determining the value of w, the value of v is simply $v = f''(x_0)$. By making this choice, we enforce that *the value*, *the slope* and *the curvature* of g and f match at x_0, i.e.

$$\begin{aligned} f(x_0) &= g(x_0) \\ f'(x_0) &= g'(x_0) \\ f''(x_0) &= g''(x_0). \end{aligned}$$

The reason why the quadratic term is multiplied with $\frac{1}{2}$ is to cancel the exponent 2 when taking the derivatives. It is an instructive exercise to check that the function value and the first two derivatives really match at x_0 for $w = f'(x_0)$ and $v = f''(x_0)$.

Example

For our example $f(x) = e^x$, the best quadratic approximation is $g(x) = 1 + x + \frac{x^2}{2}$.

◁

This example can be generalized to polynomials of arbitrary order. We can find better local approximations by matching the (higher order) derivatives. If we want to match the first three derivatives, then we have to use approximations by a polynomial of degree 3, 4 and so on. It is a remarkable fact that any function that is sufficiently differentiable can (locally) be approximated arbitrarily well by polynomials:

Theorem (Taylor/MacLaurin)

For a differentiable function f and a x near x_0, $f(x)$ can be approximated by

$$f(x) \approx f(x_0) + (x - x_0)f'(x_0) + \frac{1}{2}(x - x_0)^2 f''(x_0) + \frac{1}{6}(x - x_0)^3 f'''(x_0) + \dots$$

or in general

$$f(x) \approx f_{(n)}(x) = \sum_{k=0}^{n} \frac{1}{k!}(x - x_0)^k f^{(k)}(x_0),$$

where $f^{(k)}(x_0)$ denotes the k-th order derivative of f, and $f_{(n)}$is referred to as the *Taylor series* approximation to $f(x)$ at x_0, or simply the *Taylor approximation of order k* at x_0.

$\Diamond$

The theorem states, that every function can locally be approximated by a polynomial. The higher the order of the polynomial, the more precise is the approximation. The linear approximation we had above is therefore really a special case for a polynomial of degree 1.

Examples

1. The cubic approximation to $f(x) = e^x$at $x_0 = 0$ is given by

$$\begin{aligned} g(x) &= f(x_0) + f'(x_0)(x - x_0) + f''(x_0)\frac{1}{2}(x - x_0)^2 + f'''(x_0)\frac{1}{6}(x - x_0)^3 \\ &= e^0 + e^0 x + \frac{1}{2}e^0 x^2 + \frac{1}{6}e^0 x^3 \\ &= 1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3. \end{aligned}$$

   We can generalize that to polynomials of any given order n: The n-th order approximation to $f(x) = e^x$ is given by

$$g_n(x) = \sum_{k=0}^{n} \frac{1}{k!}(x - x_0)^k.$$

2. The linear and the quadratic approximation to $\cos(x)$ for $x_0 = 0$ are given by

$$\begin{aligned} g_{lin}(x) &= \cos(x_0) - \sin(x_0) \cdot x \\ &= 1 \end{aligned}$$

and

$$\begin{aligned} g_{quadr}(x) &= \cos(x_0) - \sin(x_0) \cdot x - \frac{1}{2}\cos(x_0)x^2 \\ &= 1 - x^2. \end{aligned}$$

3. The quadratic approximation to $\sin(x)$ at $x_0 = 0$ is given by

$$\begin{aligned} g(x) &= \sin(x_0) + \cos(x_0) \cdot x - \frac{1}{2}\sin(x_0) \cdot x^2 \\ &= x. \end{aligned}$$

4. Here is a small piece of matlab code that plots the Taylor approximation of e^x at $x_0 = 1$ up to the order of $n = 10$.

```
>> t = [-1:0.01:2]; % define the range where we want to plot exp
>> x0 = 1; % set x0 = 1; you can change that
>> plot(t,exp(t),'k-','LineWidth',3), hold on % plot exp
>> fk = exp(x0); % compute constant part
>> n = 1; % set the polynomial order to 1
>> for k = 1:10 % loop over
     plot(t,fk,'-r','LineWidth',3); % plot the best approx. in red
     pause % wait until someone presses a key
     plot(t,fk,'-b','LineWidth',3); % plot the same curve in blue
     n = n*k; % update the factorial function
     fk = fk + 1/n*(t-x0).^k.*exp(x0); % get the next polynomial
   end
>> hold off
```

Note that this code only works for the function $f(x) = \exp(x)$ since exp is its own derivative.

◁

Exercise E

Find the quadratic approximation to $f(x)=x^4 + 3x^3 + x^2 + x$ at $x_0 = 1$.

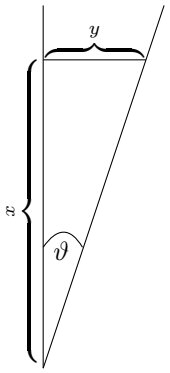
◁

Being able to approximate functions by polynomials is extremely useful, and done very often. Because many functions are hard to analyze, people work with their linear or quadratic approximations in many cases . By using these approximations, one can often get away by just using polynomials. Furthermore, finding Taylor approximations is often not hard, you just have to be able to differentiate. You should keep in mind though that the approximation is only local, i.e. for x near x_0, and you should be careful that the approximation is really appropriate for your particular application.

Here are two applications where the Taylor expansion is used.

Example: Estimating small angles

A calculation that has to be done for many psychophysical experiments it to compute the degree of visual angle. A part of the problem (it is still part of the exercises which) is to figure out the angle ϑ in the following setup:



The natural way to do so is simply by $\tan\vartheta = \frac{y}{x}$ and, therefore, $\vartheta = \operatorname{atan}\frac{y}{x}$. However, if you do not have a computer at your disposal (e.g. when reading the methods section of a paper on much more comfortable couch) you would still like to have an idea what ϑ is, even if you cannot invert the tan function in your head. Since the usual setup involves large x and small y, we expect ϑ to be very small, i.e. close to zero. Therefore, we can simply start by computing the first order Taylor approximation of tan around 0. First, we need the derivative of

course:

$$\begin{aligned}(\tan\vartheta)' &= \left(\frac{\sin\vartheta}{\cos\vartheta}\right)' \\ &= \frac{\cos^2\vartheta + \sin^2\vartheta}{\cos^2\vartheta} \\ &= \frac{1}{\cos^2\vartheta}.\end{aligned}$$

At zero, the first derivative is simply one. Since $\tan 0 = 0$, the first order Taylor approximation has the simple form

$$\tan\vartheta \approx f_{(1)}(\vartheta) = \vartheta.$$

Fortunately, this function is really easy to invert: It is simply the function itself $f^{-1}_{(1)}(\vartheta) = \vartheta$. This shows that we can use $\frac{y}{x}$ for small y and large x itself as good estimate for the angle ϑ.

$\lhd$

Example: An error analysis of depth perception

Very often a physical system shows a very specific error behavior, i.e. how the error in the inputs shows up in the outputs of a system. This error transformation is important to know, since it tells us how an error in the input will affect the quality of the system's output. A good example for that is depth perception from disparity. In this case the error in the estimated depth will grow quadratically with depth. This is what we will show in this example.

In the simplest case of two parallel pinhole cameras with a focal length f a distance of b between them, the distance of a point $\mathbf{x}$ space to the view planes of the cameras can be estimated by $d(\varrho) = \frac{fb}{\varrho}$ where $\varrho = x_l - x_r$ the disparity of the two images of $\mathbf{x}$, i.e. the distance between the x-coordinates of the image of $\mathbf{x}$ in the two view planes of the cameras. Consider we have measured a certain disparity $\hat{\varrho}$ which is δ away from the true disparity ϱ, i.e. our measurement error was δ, or $\hat{\varrho} = \varrho + \delta$. To see how this error affects the depth estimation d we make a first order Taylor expansion around the true value ϱ:

$$\begin{aligned}d(\hat{\varrho}) &\approx d_{(1)}(\hat{\varrho}) \\ &= \frac{fb}{\varrho} - \frac{fb}{\varrho^2}(\hat{\varrho} - \varrho) \\ &= \frac{fb}{\varrho} - \frac{fb}{\varrho^2}(\varrho + \delta - \varrho) \\ &= \frac{fb}{\varrho} - \frac{fb}{\varrho^2}\delta.\end{aligned}$$

The true depth error is $d(\hat{\varrho}) - d(\varrho)$. Using the Taylor expansion instead of d, we get $d(\hat{\varrho}) - d(\varrho) \approx \frac{fb}{\varrho^2}\delta$. From $d(\varrho) = \frac{fb}{\varrho}$ we can read off that the depth d is

inversely proportional to the true disparity, i.e. $d \sim \frac{1}{\rho}$. Plugging that into our expression for the depth estimation error, we obtain $d(\hat{\varrho}) - d(\varrho) \sim fbd^2\rho$, which shows that the influence of an error in the inputs grows quadratically with the depth that we want to estimate.

$\triangleleft$

1.3 Integrals

1.3.1 Introduction and Definition of Integral

Sometimes we face the situation that we are only given the rate of change $\dot{x}(t)$ of some quantity of interest. How can we convert this rate into an absolute value? One simple idea would be to start with a fixed value $x_0 := x(t_0)$ and add up the changes $\dot{x}(t)\Delta t$ for small intervals Δt until we arrive at the point t_n where we want to know the value of x:

$$\begin{aligned} x(t_n) &= \sum_{i=0}^{n-1} \dot{x}(t_i)\Delta t_i \\ \text{with} \quad & \Delta t_i = t_{i+1} - t_i. \end{aligned}$$

However, in most cases we will make an error, since the rate of change within an interval Δt_i might vary. By only looking at the rate at the interval borders t_i, we completely ignore that. We could make the intervals shorter to keep the error small, but in general we will always get an answer $x(t_n)$ that is slightly wrong. How can we avoid those errors?

In the previous section, we saw that taking the derivative of a function is calculating the rate of change of this function at every point. In this situation we were given the function and wanted to know the rate of change. Now, the situation is reverse. Therefore, what we would really like to do is to reverse the process of differentiation, i.e. to get from the rate of change to the actual function value. This is the subject of this chapter. It will turn out that integration is not as easy as differentiation. However, it has a nice geometrical interpretation that we already read of the introductory example above. Figure 1.14 shows a picture of our approximate integration from above. When looking at the sum $\sum_{i=0}^{n-1} \dot{x}(t_i)\Delta t_i$ we can interpret Δt_i as the width of a rectangle with height $\dot{x}(t_i)$. Summing up their products then means summing up the areas of those rectangles. We saw in earlier sections, that taking the derivative is the same as computing the rate of change of a function for infinitesimal small steps along the x-axis. This infinitesimal small step was incorporated via taking the limit of the differential quotient, when letting the step size go to zero. Integration involves a similar operation that we just mention here. When integrating, you really apply the approximate summation from above, but with infinitesimal small Δt_i. This means that an integral is the limit of the sum from above when letting Δt_i go to zero. This limit of the sum is then denoted by a new symbol "$\int$", a stylized "S" for "sum". The Δt is replaced by the symbol dt, which indicates that Δt is infinitesimally small.

Now we are ready to give the geometrical interpretation of the integral $\int_{t_0}^{t_n} \dot{x}(t)dt$: By making the width of the rectangles infinitesimally small, the area of all rectangles together is the area under the curve $\dot{x}(t)$.

Definition:

The integral of the function $f(x)$ from a to b is denoted by $\int_a^b f(x)dx$.

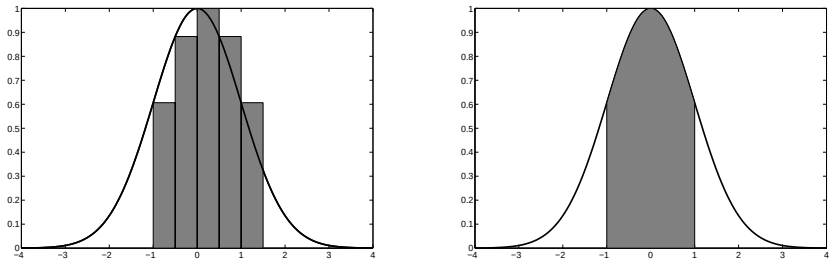


Figure 1.13: Left: Geometrical interpretation of the approximate integration: Areas of width Δt_i and height $\dot{x}(t_i)$ are summed up. Right: Geometrical interpretation of integration: The integral of the displayed function in the interval $[-1, 1]$ is the area under the curve of that function.

$\Diamond$

We can read of another important property of the integral from the geometrical interpretation. When computing our approximate integral, we did not make sure that the function values of $\dot{x}(t)$ where positive. Since Δt was always negative, it could in principle happen, that the area $\dot{x}(t_i)\Delta t_i$ has a negative value. In this case the area would really be subtracted from the whole area, instead of being added. Therefore, our approximate integral was really the signed area under the curve. This carries over to the real integral. *The integral is the signed area under the graph of $f(x)$. All parts of $f(x)$ that lie under the x-axis will be subtracted from the whole area.* Therefore, if we really want to compute the area between the graph and the x-axis, we have to find out where $f(x)$ crosses the x-axis and must compute the integrals between those points separately, putting a minus in front of the integral where $f(x)$ is under the x-axis.

Example

If a car is moving at constant speed v, the distance it travels in time t_0 is given by vt_0. If we draw a plot of v against t_0, we can see that the distance traveled between times 0 and t_0 is given by the following shaded region. This is also true if the speed v is not constant, but a function of time $v(t)$.

Therefore, the distance traveled between times 0 and t_0 can be computed by solving the integral $\int_0^{t_0} v(t)dt$. In the case of constant speed, the velocity v does not depend on t. In this case we can already compute the integral geometrically. As already said, it is the area under the constant curve between 0 and t_0. This is simply vt_0. So far, we do not know how to compute the integral for the case where v depends on t. We first have to introduce a few rules how to solve integrals. This is what we are going to do in the following paragraphs.

$\triangleleft$

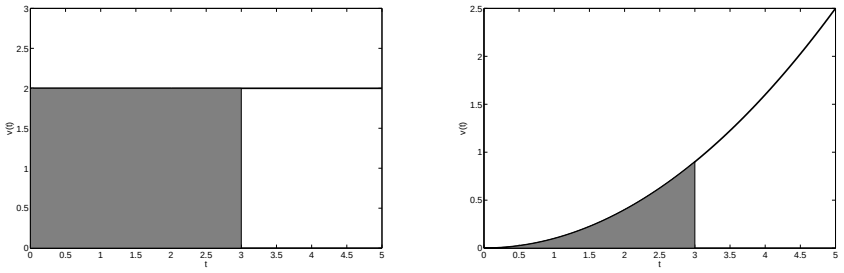


Figure 1.14: Distance traveled at time in the time interval $[0, 3]$ with constant speed (left) and with varying speed (right).

Most rules for solving simple integrals are actually the inverted rules for differentiation. Before reversing this rules we must make sure that integrating is indeed the inverse of differentiation.

Proposition (Integrating is the inverse of differentiating)

Suppose that the function $F(x)$ is defined by $F(x) = \int_0^x f(s)ds$, i.e. $F(x)$ is the area under the curve from 0 to x. Then, $F'(x) = f(x)$, provided that $f(x)$ is continuous at x.

Proof:

For simplicity, suppose that $f(s)$ is increasing near x. Then

$$\begin{aligned} f(x)\Delta x \quad &\leq \textstyle\int_x^{x+\Delta x} f(s)ds \quad &\leq f(x+\Delta x)\Delta x \\ f(x) \quad &\leq \frac{\int_x^{x+\Delta x} f(s)ds}{\Delta x} \quad &\leq f(x+\Delta x). \end{aligned}$$

Now, we can observe that

$$F(x+\Delta x) - F(x) = \int_x^{x+\Delta x} f(s)ds,$$

so

$$f(x) \leq \frac{F(x+\Delta x) - F(x)}{\Delta x} \leq f(x+\Delta x).$$

If we let $\Delta x \to 0$, $f(x+\Delta x) \to f(x)$ by continuity of $f(x)$, and $\frac{F(x+\Delta x)-F(x)}{\Delta x} \to F'(x)$ by definition of derivative. In summary,

$$f(x) \leq F'(x) \leq f(x),$$

which implies that $F'(x) = f(x)$.

□

The proposition shows, that integrating is really inverse differentiation. That is why the integral is also sometimes called *anti-derivative.* So far we have only discussed so called *definite integrals,* i.e. integrals that are computed between two points on the x-axis. Often, we can also inverse the process of differentiation first and plug in the borders of the integral later. The notation for this kind of integral is the same except that the boundary points are left out at the integral sign. Now, we can collect all our kinds of integrals in a mathematical definition.

Definition (Anti-Derivative)

Any function $F(x)$ with the property that $F'(x) = f(x)$ is called an *anti-derivative* or *indefinite integral* of $f(x)$. A indefinite integral is denoted by $F(x) = \int f(s)ds$. Once we have the indefinite integral, calculating any definite integral can be computed via:

$$\int_a^b f(s)ds = F(b) - F(a).$$

If $F(x)$ is an anti-derivative, so is $F(x)+C$ for any constant C. This constant does not matter, because if evaluate the integral, the constants cancel: $F(b) + C - (F(a) + C) = F(b) - F(a) + C - C = F(b) - F(a)$.

In many cases, the shorthand $[F(x)]_a^b = F(b) - F(a)$ is used.

$\Diamond$

The relationship between integrating and differentiating allows us to calculate integrals. Furthermore, once we have calculated an indefinite integral, we can easily check whether it is correct by just differentiating it again. If we get the original function we started with, everything is correct, otherwise there is a mistake somewhere. One could even just guess the integral, and then verify whether the guess is correct. So, even if integrating can be hard sometimes, checking whether the result is correct is almost always easy.

Example

Even without knowing any rule how to integrate so far, we know the anti-derivative of $f(x) = e^x$ is again $F(x) = e^x + C$, since $(e^x)' = e^x$. The area under the graph of $f(x)$ between 0 and some other point is given by the integral

$$\begin{aligned}\int_0^T f(x)dx &= F(T) - F(0) \\ &= e^T - e^0 \\ &= e^T - 1.\end{aligned}$$

For $T \to \infty$ the integral diverges, which means that the area under $f(x) = e^x$ is not finite.

$\triangleleft$

Exercise

E

Calculate the derivative of $F(x) = \frac{1}{n+1}x^{n+1}$ and thus verify that $F(x)$ is the anti-derivative of $f(x) = x^n$. For which values of n is this not valid?

◁

If you have solved the exercise, you have verified that the indefinite integral of $f(x) = x^n$ is $F(x)$ for all $n \neq -1$. This is the only rule we will introduce here. More advanced rules will be introduced in the next section. Like for differentiation, there are a few anti-derivatives that we will not prove here and which you should just remember, because they occur very often when calculating integrals.

Anti-Derivatives of important functions

- $f(x) = a \Rightarrow F(x) = ax + C$
- $f(x) = x^a \Rightarrow F(x) = \frac{1}{a+1}x^{a+1} + C$ for $a \neq -1$
- $f(x) = \sin(x) \Rightarrow F(x) = -\cos(x) + C$
- $f(x) = \cos(x) \Rightarrow F(x) = \sin(x) + C$
- $f(x) = e^x \Rightarrow F(x) = e^x + C$
- $f(x) = \frac{1}{x} \Rightarrow F(x) = \ln(x) + C$

Example (Approximating Integrals with Matlab)

Using a similar idea as for taking numerical derivatives we can also numerically compute approximate values of integrals in matlab. This idea is to use the definition of an integral as the sum of the areas of rectangles with height $f(x)$ and width Δx in the limit of infinitesimally small Δx. In order to give you a function, that you can reuse, we will use a bit more advanced matlab programming. Assume you are given a matlab function f which returns the function values at the points that you call it on. You can realize that in matlab via a so called *function handle*. A function handle is a variable, which is really a function. That means you can simply call the variable as if it would be a function. The important point is, that you can write code that uses a function for which you do not know what it will be exactly. For now, we simply a assign a function to a function handle `f`. This seems completely pointless now, but will be become clear later. You can assign functions to variable by using the "`@`"-symbol.

```
 >> f = @exp % assign f to be the exponential function
f =
    @exp
```

Now we go on with approximating our integral. First, we choose a certain small step width Δx, integral border between which we want to compute the integral, and base points at which we will evaluate the function to get the height of the rectangles.

```
>> dx = .01; % set stepwidth
>> a = 1; b = 2; % set integral borders
>> x = [a:dx:b]; % get base points
```

Next, we evaluate the function `f` (which is exp at the moment) at the base points.

```
>> fval = f(x); % evaluate function handle at x
```

Now, we use the fact that

$$\begin{aligned}\int_a^b f(x)dx &\approx \sum_{i=1}^{n} f(x_i)\Delta x \\ &= \Delta x \sum_{i=1}^{n} f(x_i),\end{aligned}$$

where is the number of our base points x_i and $f(x_i)$ are the values contained in the array `fval`. Translating that formula into matlab yields:

```
>> I = sum(fval)*dx; % compute approximate integral
```

Now, we can use the fact that we did not directly specify what function `f` exactly is to write a matlab function which we can reuse. For that purpose, we create a file with the name of the function by:

```
>> edit integrate.m
```

In that file we specify the function.

```
function I = integrate(f,a,b,dx)
    x = [a:dx:b]; % get base points
    fval = f(x); % evaluate function handle at x
    I = sum(fval)*dx; % compute approximate integral
```

The code above defines a function named `integrate` that takes four input arguments: the function handle `f`, the integral boundaries `a` and `b` and the step width `dx`. It returns the approximate value of the integral $\int_a^b f(x)dx$ as we compute it above. However, we can call it on different functions and different integral boundaries now, without having to redefine `a` and `b` all the time. Our example from above can be reproduced via:

```
>> I = integrate(@exp,1,2,.01)
I =
    4.7213
```

Other uses of the function could look like

```
>> I = integrate(@cos,0,2,.01)
I =
    0.9122
>> I = integrate(@log,1,3,.001)
I =
    1.2964
```

◁

1.3.2 Evaluating and Transforming Integrals

1.3.2.1 Integrals and Sums

Above, we have seen that the integral of f is the limit of a sum of infinitesimal narrow rectangles with heights given by the function values $f(x)$. If f is the sum of two functions $f(x) = g_1(x) + g_2(x)$, we can split this sum into two. Therefore, the integral of a sum of two functions is the sum of their integrals:

$$\int_a^b (f(x) + g(x))dx = \int_a^b f(x)dx + \int_a^b g(x)dx.$$

This rule is of course also valid for a sum of an arbitrary number of functions.

By the same reasoning, constant factors can be pulled out of the integral, because constant factors can be pulled out of sums:

$$\int_a^b c \cdot f(x)dx = c \cdot \int_a^b f(x)dx.$$

1.3.2.2 Integrals of Polynomials

Now we are equipped with enough rules to be able to calculate the anti-derivative of polynomials. We show how to do this with a small example.

Example

We calculate the anti-derivative of $f(x) = 2x^2 + x + 1$:

$$\begin{aligned} F(x) &= \int f(x)dx \\ &= \int 2x^2 + x + 1dx \\ &= \int 2x^2 dx + \int x dx + \int 1dx \\ &= \frac{2}{3}x^3 + \frac{1}{2}x^2 + x. \end{aligned}$$

We can use this anti-derivative to compute the integral $\int_1^2 f(x)dx$:

$$\begin{aligned} \int_1^2 f(x)dx &= F(2) - F(1) \\ &= \frac{16}{3} + 1 + 2 - \frac{2}{3} - \frac{1}{2} - 1 \\ &= \frac{37}{6}. \end{aligned}$$

◁

Example

Before coming to more advanced rules for calculating integrals, we show a more sophisticated example: What is the integral of $f(x) = |x|$ from -10 to 10? At first glance, it is not clear how we should integrate $|x|$. Therefore, we apply a simple trick. Since an integral is simply an infinite sum of infinitesimal small rectangle, it does not matter if we split an integral into two "sums". Therefore, we can split the integral $\int_{-10}^{10} |x|dx$ into $\int_{-10}^{10} |x|dx = \int_{-10}^{0} |x|dx + \int_0^{10} |x|dx$. Between those boundaries, we can use the piecewise definition of $|x|$:

$$\begin{aligned} \int_{-10}^{10} |x|dx &= \int_{-10}^{0} -x\,dx + \int_0^{10} x\,dx \\ &= [-\frac{1}{2}x^2]_{-10}^{0} + [\frac{1}{2}x^2]_0^{10} \\ &= \frac{100}{2} + \frac{100}{2} = 100. \end{aligned}$$

◁

1.3.2.3 Integrals and Change of Variables

Sometimes, it is easy to integrate $f(s)$, but we actually want to integrate $g(x) = f(x+s)$. The new function $g(x)$ can be seen as a shifted version of $f(x)$, so we can integrate by shifting the limits of integration as well. This leads the rule:

$$\int_a^b f(x+s)dx = \int_{a+s}^{b+s} f(x)dx.$$

Example

$$\int_0^{10} (x-5)^2 dx = \int_{-5}^{5} x^2 dx = \frac{2}{3}5^3$$

◁

Similarly, we might want to integrate a function $g(x) = f(sx)$ where the x-axis has been rescaled. In this case, we also have to adjust the limits of integration, but also get an extra factor in front of the integral.

$$\int_a^b f(sx)dx = \frac{1}{s}\int_{as}^{bs} f(x)dx$$

Example

$$\int_0^3 (2x)^2 dx = \frac{1}{2}\int_0^6 x^2 dx = 2 \cdot \frac{1}{3} \cdot 8 \cdot 27$$

$$\int_0^1 e^{2x} dx = \frac{1}{2}\int_0^2 e^x dx = \frac{1}{2}\left(e^2 - 1\right)$$

$$\int_0^{2\pi} \sin\left(\frac{x}{2}\right) = 2\int_0^{\pi} \sin(x)dx = 2 \cdot 2.$$

◁

The following box provides a summary of all transformation rules, that we have seen so far.

Transformations rules for integrals

- Summation Rule: The integral of a sum of is the sum of the integrals:

$$\int_a^b \sum_{i=1}^{n} g_i(x)dx \quad = \quad \sum_{i=1}^{n} \int_a^b g_i(x)dx.$$

- Constant factors can be pulled out of the integral:

$$\int_a^b c \cdot f(x)dx \quad = \quad c \cdot \int_a^b f(x)dx.$$

- Integrating from a to c is the same as integrating from a to b, and then b to c, and adding the two integrals (This is useful for evaluating integrals over piecewise functions):

$$\int_a^b \cdot f(x)dx \quad + \int_b^c f(x)dx = \quad \int_a^c f(x)dx.$$

- Shift Rule:

$$\int_a^b f(x+s)dx = \int_{a+s}^{b+s} f(x)dx.$$

- Rescaling Rule:

$$\int_a^b f(sx)dx = \frac{1}{s}\int_{as}^{bs} f(x)dx$$

1.3.2.4 Integrals and Statistics

Integrals are extremely important in probability theory and statistics. The expected value of a random variable X is simply the integral $E(X) = \int_\Omega x \cdot p(x)dx$, where $p(x)$ is the probability density function of x. The subscript Ω indicates that we choose the boundaries of the integral such that all possible values of the random variable are covered. The integral is often not too hard to do. Unfortunately, for our favorite distribution, the *Normal distribution* or *Gaussian*, calculating the expectation involves some techniques that are beyond the scope of this course[1].

Examples

1. **The Expectation of a Uniform Distribution:**
   The pdf of a Uniform Distribution on $[0, 1]$ is $f(x) = 1$. The expectation is $\mathbb{E}[X] = \int_0^1 xdx = \frac{1}{2}$

[1] namely either transforming to polar coordinates or using contour integration

2. **The Variance of a Uniform Distribution**
   The Variance is $\mathbb{V}[X] = \mathbb{E}[X^2] - [\mathbb{E}[X]]^2$. So, for the Uniform distribution, $\mathbb{V}[X] = \int_o^1 x^2 dx - \frac{1}{4} = \frac{1}{3} - \frac{1}{4} = \frac{1}{12}$.

3. **The expectation function $\mathbb{E}$ is a linear function for any distribution:**
   Consider the random variables X and Y and their joint distribution $p(x, y)$. The expectation of X is given by $\mathbb{E}[X] = \int xp(x)dx$, the expectation of Y by $\mathbb{E}[y] = \int yp(y)dy$. By not specifying any integral boundaries we silently assume that the integrals are taken over the whole range of possible values of each random variable. The single distributions $p(x)$ and $p(y)$, called *marginal distributions* or simply *marginals*, can be computed from the joint distribution $p(x, y)$ by $p(x) = \int p(x, y)dy$ and $p(y) = \int p(x, y)dx$. This computation is usually referred to as "integrating out variables". What it means is, that when writing $p(x)$ we do not care about the value of y. For that reason we have to collect all the probability mass of the ys that occur together with that x. This is exactly what we do by integrating.
   Even though, we do not know anything about the distribution $p(x, y)$, by simply using properties of integrals, we can show that the expectation is a linear function. For that purpose we have to show that (i) the expectation of the sum of X and Y is the sum of the expectations and (ii) the expectation of scaled versions of X is the scaled expectation of X. For the sake of compactness, we merge (i) and (ii) in a single computation. We show that $\mathbb{E}[aX + bY] = a\mathbb{E}[X] + b\mathbb{E}[Y]$ for $a, b \in \mathbb{R}$.

$$
\begin{aligned}
\mathbb{E}[aX + bY] &= \int\int (ax + by)p(x, y)\, dxdy \\
&= \int\int axp(x, y)\, dxdy + \int\int byp(x, y)\, dxdy \\
&= a\int\int xp(x, y)\, dxdy + b\int\int yp(x, y)\, dxdy \\
&= a\int x\int p(x, y)\, dydx + b\int y\int p(x, y)\, dxdy \\
&= a\int xp(x)\, dx + b\int yp(y)dy \\
&= a\mathbb{E}[X] + b\mathbb{E}[Y]
\end{aligned}
$$

◁

1.4 Derivatives in $\mathbb{R}^n$

1.4.1 Partial Derivatives

Example (Least Squares)

Consider the following problem: We are given a number of noisy measurements $y_1, ..., y_m \in \mathbb{R}$ that we measured at corresponding base points $x_1, ..., x_m \in \mathbb{R}$. Now we want to model the y_i as an affine function of x_i, i.e. we want to find a line $g(x) = wx + b$ such that $y_i \approx g(x_i)$. Since your measurements are noisy, there is probably not a line that contains all points (x_i, y_i). From all the other lines that match the points more or less well we have to choose one that matches our points (x_i, y_i) best. The meaning of "matching best" is usually expressed by means of a loss function. The most common choice is the sum of the squared deviations from that line

$$\ell(w, b) \quad = \quad \sum_{i=1}^{m} (y_i - (wx_i + b))^{\mathbf{2}}. \tag{1.3}$$

This *squared loss* is a function of the line parameters w and b. The approach of finding a line by minimizing (1.3) is known as *least squares* and goes back to Gauß, who became famous for finding a lost comet with this approach.

◁

Our loss is a function of two parameters, i.e. $\ell : \mathbb{R}^{\mathbf{2}} \to \mathbb{R}$. So far, we only looked at finding the optima of functions $f : \mathbb{R} \to \mathbb{R}$. However, there are only a few changes in the rules for taking derivatives or finding optima when dealing with *multivariate functions* $f : \mathbb{R}^n \to \mathbb{R}$.

The first change is obvious. Since our function has more than one variable, we must take the derivative with respect to more than one variable now. This kind of derivative is know as *partial derivative*.

Definition (Partial Derivative):

Let $f : \mathbb{R}^n \to \mathbb{R}$, $(x_1, ..., x_n) \mapsto f(x_1, ..., x_n)$ be a multivariate function. The partial derivative with respect to a x_i is taking the normal derivative of f with respect to x_i while treating all other x_j, $j \neq i$ as constants. The partial derivative with respect to x_i at the point $\mathbf{z} = (z_1, ..., z_n)$ is denotes with the symbol $\frac{\partial f}{\partial x_i}(\mathbf{z})$, $\frac{\partial}{\partial x_i} f(\mathbf{z})$ or sometimes $\frac{\partial}{\partial x_i}|_{\mathbf{z}} f$.

◊

When dealing with a function of n variables, we can compute n partial derivatives. Computing all partial derivatives and assembling them in a vector gives us the so called *gradient*.

1.4.2 Gradient and Optima in $\mathbb{R}^n$

Definition (Gradient):

The n-dimensional vector containing all partial derivatives of a function $f : \mathbb{R}^n \to \mathbb{R}$ at the position $\mathbf{z}$

$$\nabla f(\mathbf{z}) \quad = \quad \left(\frac{\partial f}{\partial x_1}(\mathbf{z}), ..., \frac{\partial f}{\partial x_n}(\mathbf{z}) \right)$$

is called gradient of f at $\mathbf{z}$. It is denoted with $\nabla f(\mathbf{z})$ or sometimes $\nabla|_{\mathbf{z}} f$.

$\Diamond$

Similarly to finding optima of *univariate functions* $f : \mathbb{R} \to \mathbb{R}$, where $f'(x) = 0$ was a necessary condition for x being an optimum, a necessary condition for $\mathbf{x} = (x_1, ..., x_n)$ being an optimum is that all partial derivatives vanish, i.e. $\nabla f(\mathbf{x}) = (0, ..., 0)$.

Remark (Gradient Descent)

The gradient of a function f at a point $\mathbf{x}$ has a remarkable property. It always points into the direction of steepest ascent of the function f at $\mathbf{x}$. This property is used in the so called *gradient descent* algorithm. Starting at a random position $\mathbf{x}^{(0)}$, gradient descent computes the gradient $\nabla f(\mathbf{x}^{(0)})$ of f at $\mathbf{x}^{(0)}$ and makes a small step in its direction (or in the opposite direction, when minimizing a function). Let us assume that we want to minimize a function (for maximizing, the gradient is added and not subtracted). We start at a random position $\mathbf{x}^{(0)}$, compute the gradient $\nabla f(\mathbf{x}^{(0)})$ and obtain a new value for $\mathbf{x}$ via $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} - \alpha \nabla f(\mathbf{x}^{(0)})$. At $\mathbf{x}^{(1)}$, we repeat this procedure. By that we obtain the general update rule

$$\mathbf{x}^{(t+1)} \quad = \quad \mathbf{x}^{(t)} - \alpha \nabla f(\mathbf{x}^{(t)}).$$

The art, somehow, is to choose the scaling constant α. This is usually done by *line search* algorithms, that we do not describe here. However, choosing a small constant value for α also works for simple examples. The procedure is repeated until the gradient becomes zero, i.e. $\nabla f(\mathbf{x}^{(t)}) = (0, ..., 0)$. Gradient descent does not yield a global minimum. It simply gives you the local minimum that you reach when running down the surface defined by the function f starting at $\mathbf{x}^{(0)}$. Intuitively, you can imagine gradient descent as a small ball, that rolls down a surface until it gets stuck in the next valley.

$\triangleleft$

Example (Least Squares cont'd)

Let us look again at our least squares problem. Since we want to find the minimum of

$$\ell(w, b) \quad = \quad \sum_{i=1}^{m} (y_i - (wx_i + b))^2,$$

we need to compute the gradient $\nabla \ell(w, b)$, set it to zero and solve for w and b.

$$
\begin{aligned}
\frac{\partial \ell}{\partial w} &= \frac{\partial}{\partial w} \sum_{i=1}^{m} \left(y_i - (w x_i + b)\right)^2 \\
&\overset{\text{Summation Rule}}{=} \sum_{i=1}^{m} \frac{\partial}{\partial w} \left(y_i - (w x_i + b)\right)^2 \\
&\overset{\text{Chain Rule}}{=} \sum_{i=1}^{m} 2\left(y_i - (w x_i + b)\right) \cdot \frac{\partial}{\partial w} \left(y_i - (w x_i + b)\right) \\
&= \sum_{i=1}^{m} 2\left(y_i - (w x_i + b)\right) \cdot -x_i \\
&= 2 \sum_{i=1}^{m} w {x_i}^2 + b x_i - x_i y_i \\
\frac{\partial \ell}{\partial b} &= \frac{\partial}{\partial b} \sum_{i=1}^{m} \left(y_i - (w x_i + b)\right)^2 \\
&= \sum_{i=1}^{m} 2\left(y_i - (w x_i + b)\right) \cdot \frac{\partial}{\partial b} \left(y_i - (w x_i + b)\right) \\
&= \sum_{i=1}^{m} 2\left(y_i - (w x_i + b)\right) \cdot -1 \\
&= \sum_{i=1}^{m} 2\left(w x_i + b - y_i\right).
\end{aligned}
$$

If we denote the mean of x_i with $\mu_x = \frac{1}{m} \sum_{i=1}^{m} x_i$, the mean over y_i with $\mu_y = \frac{1}{m} \sum_{i=1}^{m} y_i$, and splitting the sums, we can simplify the equations to

$$
\begin{aligned}
\frac{\partial \ell}{\partial w} &= 2\left(w \sum_{i=1}^{m} {x_i}^2 + b m \mu_x - \sum_{i=1}^{m} x_i y_i\right) \\
\frac{\partial \ell}{\partial b} &= 2\left(w m \mu_x + m b - m \mu_y\right)
\end{aligned}
$$

Setting $\frac{\partial \ell}{\partial b} = 0$ and solving for b then yields

$$
\begin{aligned}
2\left(w m \mu_x + m b - m \mu_y\right) &= 0 \\
\Leftrightarrow b &= \mu_y - w \mu_x.
\end{aligned}
$$

This equation has a simple interpretation. b is the mean deviation of y from our line $g(x) = wx$ without offset b.

Plugging that into $\frac{\partial \ell}{\partial w}$ yields

$$\begin{aligned} 2\left(w\sum_{i=1}^{m} {x_i}^{\mathbf{2}} + bm\mu_x - \sum_{i=1}^{m} x_i y_i\right) &= 2\left(w\sum_{i=1}^{m} {x_i}^{\mathbf{2}} + m\mu_x(\mu_y - w\mu_x) - \sum_{i=1}^{m} x_i y_i\right) \\ &= 2\left(w\sum_{i=1}^{m} {x_i}^{\mathbf{2}} + m\mu_x\mu_y - wm\mu_x^2 - \sum_{i=1}^{m} x_i y_i\right). \end{aligned}$$

If we remember that the variance ${\sigma_x}^{\mathbf{2}}$ of the x_i can be computed via ${\sigma_x}^{\mathbf{2}} = \frac{1}{m}\sum_{i=1}^{m} {x_i}^{\mathbf{2}} - {\mu_x}^{\mathbf{2}}$ and if we denote the *covariance* between x and y with $\sigma_{xy} = \frac{1}{m}\sum_{i=1}^{m} x_i y_i - \mu_x\mu_y$, we can simplify the equation to

$$2\left(w\sum_{i=1}^{m} {x_i}^{\mathbf{2}} + m\mu_x\mu_y - wm\mu_x^2 - \sum_{i=1}^{m} x_i y_i\right) = 2\left(wm{\sigma_x}^{\mathbf{2}} - m\sigma_{xy}\right).$$

Setting that equation to zero and solving for w yields

$$\begin{aligned} 2\left(wm{\sigma_x}^{\mathbf{2}} - m\sigma_{xy}\right) &= 0 \\ \Leftrightarrow w &= \frac{\sigma_{xy}}{{\sigma_x}^{\mathbf{2}}}. \end{aligned}$$

The term $\frac{\sigma_{xy}}{{\sigma_x}^{\mathbf{2}}}$ has again a simple interpretation: It is the correlation coefficient $\rho = \frac{\sigma_{xy}}{\sqrt{{\sigma_x}^{\mathbf{2}}}\sqrt{{\sigma_y}^{\mathbf{2}}}}$ times the the standard deviation $\sigma_y = \sqrt{{\sigma_y}^{\mathbf{2}}}$ of the y_i divided by the standard deviation of the x_i, i.e. $w = \rho\frac{\sigma_y}{\sigma_x}$. Plugging that into the equation for b yields

$$\begin{aligned} b &= \mu_y - w\mu_x \\ &= \mu_y - \rho\frac{\sigma_y}{\sigma_x}\mu_x \end{aligned}$$

and we are left with the line equation

$$\begin{aligned} g(x) &= wx + b \\ &= \rho\frac{\sigma_y}{\sigma_x}x + \mu_y - \rho\frac{\sigma_y}{\sigma_x}\mu_x \\ &= \rho\frac{\sigma_y}{\sigma_x}(x - \mu_x) + \mu_y. \end{aligned}$$

If you think about it, this equation makes perfect sense. In order to obtain a estimate for the associated y for a given x, we first subtract the mean from x as estimated by our sample, normalize its variance to one by dividing by the standard deviation, multiply the result with the correlation coefficient ρ, multiply the result with the standard deviation of y to get the scale right and finally add the mean μ_y of y.

$\lhd$

Unfortunately, we cannot check if the values for w and b really correspond to a minimum, yet. In order to do so we would need the concept of matrices and eigenvalues. Just for the sake of completeness, we quickly mention how to generalize the sufficient conditions for a minimum and a maximum to more than one dimension.

The second derivative of a function $f : \mathbb{R}^n \to \mathbb{R}$ is a $\mathbb{R}^{n \times n}$ matrix. This is not surprising, since the first derivative with respect to all variables was a vector (the gradient), or a function from $\mathbb{R}^n$ into $\mathbb{R}^n$. Taking the derivatives with respect to all parameters again, yields a matrix. This matrix is called the *Hessian matrix* or simply the *Hessian*.

Definition (Hessian):

The $\mathbb{R}^{n \times n}$ matrix H containing all the second derivatives of a function $f : \mathbb{R}^n \to \mathbb{R}$ at a position $\mathbf{z}$

$$H(\mathbf{z}) = \begin{pmatrix} \frac{\partial^2 f}{\partial {x_1}^2}(\mathbf{z}) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(\mathbf{z}) & & \\ \frac{\partial^2 f}{\partial x_1 \partial x_2}(\mathbf{z}) & & & \\ & & & \frac{\partial^2 f}{\partial x_{n-1} \partial x_n}(\mathbf{z}) \\ & & \frac{\partial^2 f}{\partial x_{n-1} \partial x_n}(\mathbf{z}) & \frac{\partial^2 f}{\partial {x_n}^2}(\mathbf{z}) \end{pmatrix}$$

is called *Hessian.*

$\Diamond$

The sufficient condition for a function f to have a minimum or maximum at $\mathbf{x}$ is that $H(\mathbf{x})$ is *positive definite* or *negative definite*, respectively. This means that $H(\mathbf{x})$ has only *positive* or *negative* eigenvalues, respectively. You will see what that means in the following part about linear algebra.

Chapter 2

Linear Algebra

Linear Algebra plays an important role in many branches of mathematics and natural sciences. Even when not visible at first glance, linear algebra can help you to understand the biggest portion of mathematics that you will ever have to deal with. This is basically due to the fact that most of linear algebra can be interpreted geometrically which is a helpful source of intuition. However, linear algebra comes with the price that it is confusing for most people at the beginning. The reason for this is basically that linear algebra introduces a lot of new definitions and uses a special kind of notation, the matrix notation, for its operations. But as soon as one got used to the linear algebra way of thinking, it is a very helpful and natural tool which one does not want to miss.

We will start this section by introducing *vectors*

$$\mathbf{x} \quad = \quad \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix},$$

which will allow us to efficiently represent n inputs in a single variable $\mathbf{x}$. Along with them we will introduce a number of useful operations that can be done with vectors. In 1.1.2 we saw that a linear function is completely determined by a single input-output pair (x, y) with $y = f(x)$. At that point, the linear function took only one input argument x. In this chapter we will look at linear functions that take several inputs $\mathbf{x}_1, ..., \mathbf{x}_n$.

It will turn out that, similar to one-dimensional linear functions, a n-dimensional linear function is completely determined by n input output pairs $(y_i, \mathbf{x}_i)$ with $i = 1, ..., n$. This will enable us to introduce matrices, a extremely powerful tool when dealing with linear functions of more than one dimension.

We will finish this chapter with advanced operations and properties of matrices, such as eigenvectors and eigenvalues.

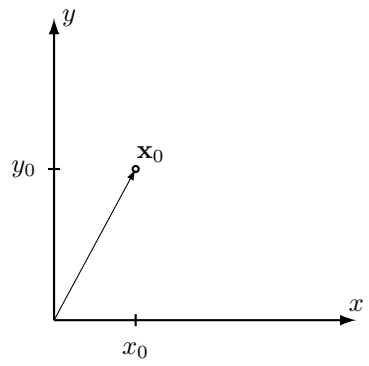
2.1 Vectors

A vector is a shorthand notation for a ordered set of numbers $(x_1, ..., x_n)$. As a convention, we denote vectors with a lower case Latin letter in bold font

$$\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}.$$

Examples

- The x and y coordinates of a point in 2-D $\mathbf{x}_0 = \binom{x}{y} \in \mathbb{R}^2$ is a vector. Similarly, any point in in 3-D $\mathbf{x}_0 = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ can be written as a vector. So, every point in space can be written as a vector.



Vectors are sometimes drawn as points and sometimes drawn as arrows, depending on what one intends to indicate. When drawing it as a point one wants to emphasize that the elements of the vector are coordinates in space. When drawing it as an arrow, one wants to emphasize that the vector indicates a *direction,* i.e. the direction from the center of the coordinate system to the point $\mathbf{x}_0 = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$.

- The activity, i.e the firing rates, of n simultaneously recorded neurons can be described as a n-dimensional vector $\mathbf{r} = \begin{pmatrix} r_1 \\ \vdots \\ r_n \end{pmatrix}$.

- The color of a pixel, e.g. in the RGB color space, can be written as a vector $\mathbf{c} = \begin{pmatrix} r \\ g \\ b \end{pmatrix}$.

- The pixel intensities of an $n \times m$ image

$$\mathbf{I} = \begin{pmatrix} I_{11} & I_{12} & \dots & & I_{1m} \\ I_{21} & & & & \vdots \\ \vdots & & \ddots & & \\ & & & & I_{(n-1)m} \\ I_{n1} & & \dots & I_{n(n-1)} & I_{nm} \end{pmatrix}$$

  can be described as a vector $\mathbf{v} \in \mathbb{R}^{n \times m}$ by stacking the columns of $\mathbf{I}$ upon each other:

$$\mathbf{v} = \begin{pmatrix} I_{11} \\ I_{21} \\ \vdots \\ I_{ij} \\ I_{(i+1)j} \\ \vdots \\ I_{nm} \end{pmatrix}.$$

  It might be tempting to think of an image as a matrix. In matlab, they are usually stored as matrices, and that is certainly useful for looking at them. However, *mathematically,* it is really much more natural to think of an image as a vector. Matrices (as we will see later) are really transformations which can be multiplied–does it really make sense to multiply to images? When talking about operation on images in the following, unless explicitly stated otherwise, we always implicitly assume that a image is represented as this kind of stacked vector. Note that stacking columns on top of each other is also the way matlab represents matrices as vectors. If `A` is a $m \times n$ matrix, then `A(:)` is a $nm \times 1$ dimensional column vector. If you converted an image `I` into a vector `v` via `v=I(:)`, you can transform it back into matrix form via `I2=reshape(v,m,n)`, if `I` was a $m \times n$ image.

- Functions$f(x)$ can often be viewed as vectors, provided that we only consider a fixed set of inputs $x_1, \dots x_n$. Then, we the function $f(x)$ can be

thought of as a vector

$$\mathbf{f} = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix}.$$

◁

As we just saw, quite a few objects can naturally be described by vectors. Additionally, it is possible to carry out certain operations with these objects. E.g. we can always add two images or multiply it with a scalar to increase or decrease its luminance. Of course, the same operations can also be carried out with vectors. Let us just summarize vectors and their properties in a definition.

Definition (Vectors) A n-dimensional vector $\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ is a ordered set of n values $x_1, ..., x_n$. In order to indicate that a symbol denotes a vector, it is usually written in bold font. The set of all n-dimensional vectors with real entries is denoted $\mathbb{R}^n$. A set like $\mathbb{R}^n$ is also called *vector space*. There are a few basic operations that can be applied to vectors:

- *Addition:* Let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ be two n-dimensional vectors. The sum of two vectors is simply the sum of the single entries

$$\begin{aligned} \mathbf{x} + \mathbf{y} &= \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \\ &= \begin{pmatrix} x_1 + y_1 \\ \vdots \\ x_n + y_n \end{pmatrix}. \end{aligned}$$

- *Multiplication with a scalar:* Let $\mathbf{x} \in \mathbb{R}^n$ be a n-dimensional vector. The product of $\mathbf{x}$ and a scalar $a \in \mathbb{R}$ is defined as

$$\begin{aligned} a \cdot \mathbf{x} &= a \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \\ &= \begin{pmatrix} ax_1 \\ \vdots \\ ax_n \end{pmatrix}. \end{aligned}$$

- *Transposition:* So far, each vector was written as a vertical stack of its entries $\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$. However, for a reason that will become clear soon,

vectors can also be written as an array of numbers $\mathbf{x} = (x_1, ..., x_n)$. The operation of converting a column into a row vector and vice versa is denoted by the symbol "$\top$" in the superscript, i.e.

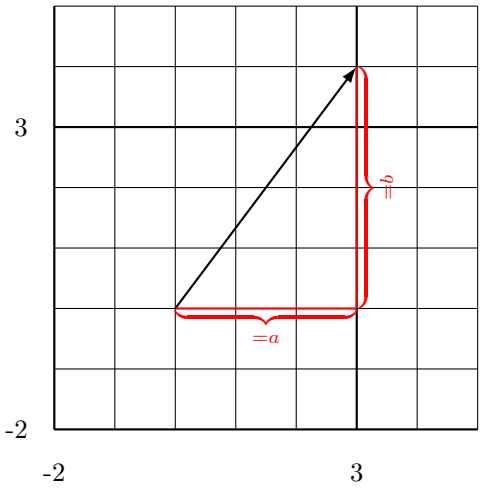
$$\begin{aligned} \mathbf{x} &= \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \\ \Leftrightarrow \mathbf{x}^\top &= (x_1, ..., x_n) \\ \Leftrightarrow (\mathbf{x}^\top)^\top &= \mathbf{x}. \end{aligned}$$

$\Diamond$

2.1.1 Length of a vector: The euclidean norm

A very basic property of a vector that we can compute is its length. In general, the length of a vector is called the *norm of a vector*. Since the norm is a very frequently occurring function of a vector $\mathbf{x}$ it has its own notation: It is indicated by $||\mathbf{x}||$.

Example Assume we are given a 2D vector $\mathbf{x} = \binom{3}{4}$. Since $\mathbf{x} \in \mathbb{R}^2$ we can use Pythagoras theorem to compute its length $||\mathbf{x}|| = \sqrt{3^2 + 4^2} = \sqrt{25} = 5$.



$\Diamond$

This way of computing a vector generalizes to n dimensions. In n dimensions, the *Euclidean norm* of a vector is the square root of the sum of its squared entries:

$$||\mathbf{v}|| = \sqrt{\sum_{i=1}^{n} v_i^2}.$$

The Euclidean norm has a few properties, that are useful when dealing with it.

Lemma (Properties of the Euclidean norm) The Euclidean norm

$$\begin{aligned} ||\cdot|| : \mathbb{R}^n &\to \mathbb{R} \\ \mathbf{x} &\mapsto ||\mathbf{x}|| = \sqrt{\textstyle\sum_{i=1}^n x_i^2} \end{aligned}$$

has the following properties:

$$||\mathbf{x}|| > 0 \quad \text{whenever} \quad \mathbf{x} \neq \mathbf{0} \tag{2.1}$$

$$||\mathbf{x}|| = 0 \quad \text{whenever} \quad \mathbf{x} = \mathbf{0} \tag{2.2}$$

$$||a\mathbf{x}|| = |a| \cdot ||\mathbf{x}|| \quad \text{for all} \quad a \in \mathbb{R} \tag{2.3}$$

$$||\mathbf{x} + \mathbf{y}|| \leq ||\mathbf{x}|| + ||\mathbf{y}|| \quad \text{for all} \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^n. \tag{2.4}$$

$\Diamond$

Examples

- In order to scale a vector $\mathbf{v}$ to length one, we just need to divide all its entries by $||\mathbf{v}||$. We can do this, since $||\mathbf{v}||$ is just a scalar. The resulting vector will have length one, since

$$\begin{aligned} \left|\left| \frac{1}{||\mathbf{v}||}\mathbf{v} \right|\right| &\overset{\text{2.1 and 2.3}}{=} \frac{1}{||\mathbf{v}||}||\mathbf{v}|| \\ &= 1. \end{aligned}$$

  The operation of scaling a vector to length one is called *normalization.*

- Dividing by the norm of a vector can also be used to normalize the mean luminance of a set of images $\mathbf{I}_1, ..., \mathbf{I}_m$. Assume you want to fix the luminance of $\mathbf{I}_1, ..., \mathbf{I}_m$ to a fixed value a. All you have to do is to multiply each image $\mathbf{I}_k$ by $\frac{a}{||\mathbf{I}_k||}$, where we think of $\mathbf{I}_k$ as a vector with the stacked columns of the kth image as in the example above.

- Variance: The variance of a set of numbers $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is given by $Var(\mathbf{x}) = \frac{1}{n}\sum_i (x_i - \mathrm{E}(\mathbf{x}))^2$, where $\mathrm{E}(x) = \frac{1}{n}\sum_i x_i$ is the average of x. This can be rewritten as $\mathrm{Var}(x) = \frac{1}{n}\|\mathbf{x} - \mathrm{E}(\mathbf{x})\|^2$.
  (Subtracting the number $E(x)$ from the vector $\mathbf{x}$ is slightly sloppy notation, though)

$\Diamond$

As you might guess, the Euclidean norm is not the only function that fulfills the properties (2.1), (2.2), (2.3) and (2.4). In fact, those properties are usually used to define a norm on a vector space. Since there are several possible norms on $\mathbb{R}^n$, the Euclidean norm is also sometimes written with a two in the subscript, i.e. $||\mathbf{x}||_2$. Since we will deal with the Euclidean norm for most of the time, we will only use the subscript when it is not clear from the context which norm is meant.

2.1.2 Projection and Scalar Product

A very important operation between two vectors is the *scalar product*[1] or *dot product*:

Definition (Scalar Product)

The *scalar* or *dot product* between two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ is defined as

$$\langle \mathbf{x}, \mathbf{y} \rangle \quad = \quad \langle \mathbf{y}, \mathbf{x} \rangle = \sum_{i=1}^{n} x_i y_i.$$

$\Diamond$

At this point we can already introduce a small bit of matrix notation: By definition, the product of a row vector $\mathbf{x}^\top$ and a column vector $\mathbf{y}$ equals the scalar product between them:

$$\mathbf{x}^\top \mathbf{y} \quad = \quad \langle \mathbf{x}, \mathbf{y} \rangle.$$

This will make it easies to understand the general matrix-matrix or matrix-vector products that we introduce later. Unlike in the multiplication of scalar values, the order of the elements does matter in matrix multiplication. Therefore, it is important that the row vector appears first and the column vector appears second.

Examples

- The mean of the entries $x_1, ..., x_n$ can be written in terms of the scalar product as

$$\begin{aligned} \frac{1}{n} \sum_{i=1}^{n} x_i &= \frac{1}{n} \langle \mathbf{1}_n, \mathbf{x} \rangle \\ &= \frac{1}{n} \mathbf{1}_n^\top \mathbf{x}. \end{aligned}$$

- In 1.3 we saw, that an integral is the sum of the areas of infinitely narrow rectangle with their height given by the function values at the respective position:

$$\int_a^b f(x)dx \quad = \quad \lim_{h \to 0} \sum_{k=0}^{n-1} f(a + k \cdot h) \cdot h,$$

  where n is the number of intervals of width h between a and b. Assume that we need to compute the value of the integral in a program we write. If the function f is not too nasty and we do not need the exact value

[1] not to be confused with a product by a scalar...

of the integral, we can just choose a very small h, e.g. $h = \frac{b-a}{2^{-m}}$ for an appropriate m, and just use a sum to approximate the integral, i.e.

$$\int_a^b f(x)dx \quad \approx \quad \sum_{k=0}^{2^m-1} f(a+k\cdot h)\cdot h.$$

But this sum is simply the scalar product between two 2^m-dimensional vectors $\mathbf{h} = h \cdot \mathbf{1}_{2^m}$ and $\mathbf{f} = \begin{pmatrix} f(a) \\ \vdots \\ f(a+(2^m-1)h) \end{pmatrix}$, i.e.

$$\int_a^b f(x)dx \quad \approx \quad \langle \mathbf{h}, \mathbf{f} \rangle.$$

For programs like Matlab this is an easy and elegant way to write code for that approximate integral. E.g. the code for computing the integral $\int_0^{2\pi} \sin(\phi)d\phi$ would look like this:

```
>> h = (2*pi)/2^10; % define increment
>> phi = [0:h:2*pi]; % get angles between 0 and 2*pi
>> f = sin(phi); % get the function values

>> h = h*ones(length(f),1); % make h into a vector
>> f*h % compute the approximate integral

ans =

-4.5996e-18
```

- In general, it is often useful to imagine an integral like $\int f(x)g(x)dx$ as a dot product between two very long vectors, because a lot of the geometrical intuition carries over. In fact, for a certain type of functions, a dot product between two functions can be defined as $\langle f, g \rangle_{\mathcal{F}} = \int_{-\infty}^{+\infty} f(x)g(x)dx$. Here the subscript $\mathcal{F}$ indicates that this dot product is not the normal dot product as we know it. Another situation where the dot product intuition is useful is the convolution of two functions, which we will introduce later.

- Spectral sensitivity of cones: The frequency spectrum of any light source can be characterized by a vector $\mathbf{l} = (l_1, l_2, \cdots, l_n)$, where, $l_k$denotes the power of the light source for some wavelength which we index as k. Similarly, the spectral sensitivity of any cone in the retina can be described by a spectral sensitivity curve $\mathbf{s} = (s_1, s_2, \ldots, s_n)$. For example, for an L-cone, the curve $\mathbf{s}$ would have its maximum at the index k which corresponds to the wavelength $560nm$. Then, the response of the receptor to any light

source $\mathbf{l}$ can be computed by calculating the dot product between $\mathbf{l}$ and $\mathbf{s}$:

$$R = \langle \mathbf{l}, \mathbf{s} \rangle = \sum_{i=1}^{n} l_i s_i.$$

- More generally, the firing rate of some neurons in the visual system can be modeled as the scalar product between the input image $\mathbf{I}$ and the receptive field Ψ. In fact, neurons for which this holds are usually referred to as *simple cells*, and neurons for which it does not are called *complex cells*. If we imagine the receptive field as an image of the same size as $\mathbf{I}$, which is only non-zero in the area, which the neuron responds to, we can write the answer as $\sum_{i=1}^{m} \sum_{j=1}^{n} \Psi_{ij} \cdot I_{ij}$, which is the scalar product between the stacked vectors of $\mathbf{I}$ and Ψ, which we introduced before. The only problem is, that this dot product can still have negative values and, therefore, the model would allow for negative firing rates. We can fix this by simply setting everything below zero to the value zero. This is done via the max-function. We thereby obtain this simple response model:

$$r \quad = \quad \max \left\{ 0, \sum_{i=1}^{m} \sum_{j=1}^{n} \Psi_{ij} \cdot I_{ij} \right\}.$$

- Covariance: The covariance between two data vectors $\mathbf{x}$ and $\mathbf{y}$ is defined to be $\mathrm{Cov}(x, y) = \frac{1}{n} \sum_i x_i y_i - E(x)E(y)$. So, if both x and y have zero mean, the covariance is exactly the same as the dot-product between the vectors (divided by n).

$\lhd$

So far, the scalar product may seem as no more than a useful notational shortcut. However, it has a lot of nice properties and it is used to compute a lot of interesting properties of vectors. Therefore, it is very important to get a good intuition for it. In the following we develop a few.

Lemma (Angle between two vectors) The cosine of the angle between the two vectors is given by their dot-product divided by the product of their norms:

$$\frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\| \|\mathbf{y}\|} \quad = \quad \cos \measuredangle \{\mathbf{x}, \mathbf{v}\}.$$

$\Diamond$

We know that the cosine of any angle must be between -1,and 1. As a consequence, we can see that the term on the left hand size, $\frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\| \|\mathbf{y}\|}$, must

always be at least -1, and at most 1. Thus, we obtain the inequality

$$\begin{aligned} -1 &\leq \tfrac{\langle \mathbf{x},\mathbf{y}\rangle}{\|\mathbf{x}\|\|\mathbf{y}\|} &\leq 1 \\ -\|\mathbf{x}\|\|\mathbf{y}\| \leq & \quad \langle \mathbf{x},\mathbf{y}\rangle & \leq \|\mathbf{x}\|\|\mathbf{y}\| \end{aligned}$$

In other words, the dot-product between any two vectors can never be bigger than the product of their norms, and never be smaller than minus that. This inequality is known as the Cauchy Schwarz inequality.

Example: Correlation coefficients are between -1 and 1

We can use the Cauchy Schwarz inequality to show that the correlation coefficients between any two vectors of data $\mathbf{x}$ and $\mathbf{y}$ must be between -1 and 1 : The correlation coefficients is defined to be

$$\begin{aligned} \text{Corr}(\mathbf{x},\mathbf{y}) &= \frac{\text{Cov}(\mathbf{x},\mathbf{y})}{\sqrt{\text{Var}\mathbf{x}\text{Var}\mathbf{y}}} \\ \text{where Cov}(\mathbf{x},\mathbf{y}) &= \frac{1}{n}\sum_i x_i y_i - \bar{\mathbf{x}}\bar{\mathbf{y}} \\ \text{and Var}\mathbf{x} &= \sqrt{\frac{1}{n}\sum_i x_i^2 - \bar{\mathbf{x}}} \\ \bar{\mathbf{x}} &= \frac{1}{n}\sum_i x_i. \end{aligned}$$

Without loss of generality, we can assume that the means of $\mathbf{x}$ and $\mathbf{y}$ are both zero. (If they are not, one can simply subtract off the mean.) Then, the formula for the correlation coefficient becomes

$$\begin{aligned} \text{Corr}(\mathbf{x},\mathbf{y}) &= \frac{\frac{1}{n}\sum_i x_i y_i}{\sqrt{\frac{1}{n}\sum_i x_i^2}\sqrt{\frac{1}{n}\sum_i y_i^2}} \\ &= \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}} \\ &= \frac{\langle \mathbf{x},\mathbf{y}\rangle}{\|\mathbf{x}\|\|\mathbf{y}\|.} \end{aligned}$$

Now, by the Cauchy-Schwarz inequality, we can see that correlation-coefficients always have to be between -1 and 1.

Orthogonality

Finally, if $\langle x, y\rangle = 0$, we know that the cosine of the angle between them must be 0, which implies that the angle is 90^o, or $\pi/2$ in radians. In this case, $\mathbf{x}$ and $\mathbf{v}$ are called *orthogonal.*

The second important intuition about scalar products can be explained in terms of *projections*. Let us start with a small example to motivate projections.

Example Assume that you have an image

$$\mathbf{I} = \begin{pmatrix} I_{11} & I_{12} & \dots & & I_{1m} \\ I_{21} & & & & \vdots \\ \vdots & & \ddots & & \\ & & & & I_{(n-1)m} \\ I_{n1} & & \dots & I_{n(n-1)} & I_{nm} \end{pmatrix}$$

which you want to use to approximate another image $\mathbf{J}$, i.e. you want to describe the image $\mathbf{J}$ by a single value λ such that $\lambda\mathbf{I}$ is closest to $\mathbf{J}$ or, in other words, such that$||\lambda\mathbf{I} - \mathbf{J}||^2$ is minimal. The vector $\mathbf{I}$ is called *projection of* $\mathbf{J}$ *onto* $\mathbf{I}$.

◁

In a more mathematical language, the idea of a projection can be described as follows: Assume that you have two vectors $\mathbf{x}$ and $\mathbf{v}$, and your goal is to express $\mathbf{x}$ as good as possible in terms of $\mathbf{v}$. If $\mathbf{x}$ and $\mathbf{v}$ are not pointing in the same directions, you will not be able to express $\mathbf{x}$ in terms of $\mathbf{v}$. However, you could search for the scalar λ such that $||\lambda\mathbf{v} - \mathbf{x}||^2$ is minimal. If $\mathbf{v}$ happens to have length one, computing the value of λ is easy: It is just $\lambda = \langle\mathbf{v}, \mathbf{x}\rangle$. If $\mathbf{v}$ does not have length one, we can simply normalize it and use $\lambda = \langle\frac{1}{||\mathbf{v}||}\mathbf{v}, \mathbf{x}\rangle$. In that case $\frac{\lambda}{||\mathbf{v}||}\mathbf{v}$ best approximates $\mathbf{x}$.

Lemma (Projection) Let $\mathbf{x}, \mathbf{v} \in \mathbb{R}^n$. The projection of $\mathbf{x}$ onto $\mathbf{v}$, i.e. the vector $\frac{\lambda}{||\mathbf{v}||}\mathbf{v}$ that minimizes $||\frac{\lambda}{||\mathbf{v}||}\mathbf{v} - \mathbf{x}||^2$ is given by $\frac{\lambda}{||\mathbf{v}||}\mathbf{v} = \langle\frac{1}{||\mathbf{v}||}\mathbf{v}, \mathbf{x}\rangle \cdot \mathbf{v}$.

◊

Interpreting the scalar product as projection is a very important intuition that helps to understand equations involving scalar products: If $\mathbf{v}$ has length one the dot product $\langle\mathbf{v}, \mathbf{x}\rangle$ is the amount $\mathbf{v}$ has at $\mathbf{x}$. If $\langle\mathbf{v}, \mathbf{x}\rangle$ has a large positive or negative value compared to the length of $\mathbf{x}$, then $\mathbf{x}$ can be well approximated by $\langle\mathbf{v}, \mathbf{x}\rangle \cdot \mathbf{v}$. The smaller the absolute value $|\langle\mathbf{v}, \mathbf{x}\rangle|$, the less $\mathbf{v}$ has in common with $\mathbf{x}$. Therefore, we can think of $\langle\mathbf{v}, \mathbf{x}\rangle$ as a *similarity measure* between $\mathbf{x}$ and $\mathbf{v}$. This also fits very well to the cosine-property of the scalar product. The smaller the angle between two vectors the larger the cosine and therefore the larger the dot product.

◁

We have seen, that the scalar product plays an important role when dealing with vectors. Therefore it is good to know a few calculation rules that make it easy to modify it.

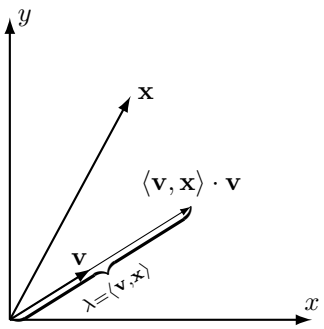


Figure 2.1: Sketch of a projection: If $\mathbf{v}$ has length one, $\lambda\mathbf{v} = \langle \mathbf{x}, \mathbf{v}\rangle\mathbf{v}$ is the closest $\mathbf{v}$-approximation of $\mathbf{x}$.

Lemma (Properties of the scalar or dot product) The scalar or dot product of two n-dimensional vectors $\mathbf{x}$ and $\mathbf{v}$

$$\langle \mathbf{x}, \mathbf{v}\rangle \quad = \quad \sum_{i=1}^{n} x_i v_i$$

has the following properties:

1. *Symmetry:* The scalar product is symmetric, i.e.

   $$\langle \mathbf{x}, \mathbf{v}\rangle \quad = \quad \langle \mathbf{v}, \mathbf{x}\rangle,$$

   for all $\mathbf{x}, \mathbf{v} \in \mathbb{R}^n$.

2. *Bi-linearity:* $\langle \mathbf{x}, \mathbf{v}\rangle$ is linear in both arguments, i.e.

   $$\begin{aligned} \langle a\mathbf{x} + b\mathbf{y}, c\mathbf{v} + d\mathbf{w}\rangle &= \langle a\mathbf{x} + b\mathbf{y}, c\mathbf{v}\rangle + \langle a\mathbf{x} + b\mathbf{y}, d\mathbf{w}\rangle \\ &= \langle a\mathbf{x}, c\mathbf{v}\rangle + \langle b\mathbf{y}, c\mathbf{v}\rangle + \langle a\mathbf{x}, d\mathbf{w}\rangle + \langle b\mathbf{y}, \mathbf{w}\rangle \\ &= ac\langle \mathbf{x}, \mathbf{v}\rangle + bc\langle \mathbf{y}, \mathbf{v}\rangle + ad\langle \mathbf{x}, \mathbf{w}\rangle + bd\langle \mathbf{y}, \mathbf{w}\rangle, \end{aligned}$$

   for all $\mathbf{x}, \mathbf{y}, \mathbf{v}, \mathbf{w} \in \mathbb{R}^n$ and $a, b, c, d \in \mathbb{R}$. Note, however, that $\langle \mathbf{x}, \mathbf{x}\rangle$ is not a linear function of $\mathbf{x}$ since it appears in both arguments.

3. *Cosine:* If $||\mathbf{x}|| = ||\mathbf{v}|| = 1$, the scalar product is the cosine of the angle between the two vectors:

$$\langle \mathbf{x}, \mathbf{v} \rangle \quad = \quad \cos \measuredangle\{\mathbf{x}, \mathbf{v}\}.$$

4. *Scalar product and norm:* The norm of $\mathbf{x}$ can be written in terms of the scalar product as follows: $||\mathbf{x}|| = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$.

$\Diamond$

Especially the second property is very important, since it basically states that we can drag out sums from the scalar product.

2.1.3 Linear (In)Dependence

We just saw, that the best approximation of a vector $\mathbf{x}$ by another vector $\mathbf{v}$ is given by $\left\langle \frac{1}{||\mathbf{v}||}\mathbf{v}, \mathbf{x} \right\rangle \mathbf{v}$. If $\mathbf{x}$ and $\mathbf{v}$ point in the same direction, we can perfectly express $\mathbf{x}$ in terms of $\mathbf{v}$, i.e. $\left\langle \frac{1}{||\mathbf{v}||}\mathbf{v}, \mathbf{x} \right\rangle \mathbf{v} = \lambda \mathbf{v} = \mathbf{x}$. In this case, when $\mathbf{v}$ can be transformed into $\mathbf{x}$ by a multiplication with a scalar λ, the vectors $\mathbf{x}$ and $\mathbf{v}$ are called *linearly dependent.* If this is not possible, i.e. $\left\langle \frac{1}{||\mathbf{v}||}\mathbf{v}, \mathbf{x} \right\rangle \mathbf{v} \neq \mathbf{x}$, the vectors $\mathbf{x}$ and $\mathbf{v}$ are called *linearly independent.* The extreme case of linear independence is *orthogonality*, i.e. if $\langle \mathbf{x}, \mathbf{v} \rangle = 0$.

In terms of projections and approximating $\mathbf{x}$ by $\mathbf{v}$, this means that the best approximation of $\mathbf{x}$, or any of its multiples, by $\mathbf{v}$ is not using $\mathbf{v}$ at all.

2.1.4 Orthonormal Bases of a Vector Space

An important questions that arises in this context is how many vectors $\mathbf{v}_1, ..., \mathbf{v}_m$ suffice to express an arbitrary n-dimensional vector $\mathbf{x}$ and how to choose them. An important theorem says that n linearly independent vectors are enough to describe any vector in an n-dimensional vector space. Such a set of n linearly independent vectors is called *a basis of* $\mathbb{R}^n$. In general, choosing an arbitrary set of n linearly independent vectors as basis is not the best way to choose a basis for a vector space. We will later explain why. However, there is one particular type of bases that are easy to handle: *orthonormal bases.* We will first look at an example and then state a rigorous definition.

Example The simplest basis for $\mathbb{R}^2$ is the so called *canonical basis* $\mathbf{e}_1 = \binom{1}{0}$ and $\mathbf{e}_2 = \binom{0}{1}$. Given an arbitrary vector $\mathbf{x} \in \mathbb{R}^n$ we can express it as a sum of the basis elements

$$\begin{aligned} \mathbf{x} &= \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ &= x_1 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + x_2 \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \end{aligned}$$

Defining the canonical basis $\mathbf{e}_1, ..., \mathbf{e}_n$ of $\mathbb{R}^n$ in exactly the same way, i.e. the ith coordinate of $\mathbf{e}_i$ is one and all others are zero, every vector $\mathbf{x} \in \mathbb{R}^n$ can then be written as

$$\mathbf{x} = \sum_{i=1}^{n} x_i \mathbf{e}_i.$$

◁

You might have noted that the canonical basis has two important properties. First of all, all basis vectors have length one, i.e. $||\mathbf{e}_i|| = 1$ for all $i = 1, ..., n$. Secondly, the basis elements are mutually orthogonal, i.e.

$$\langle \mathbf{e}_i, \mathbf{e}_j \rangle = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{else} \end{cases}.$$

There is a notational shortcut to for "$\begin{cases} 1 & \text{if } i = j \\ 0 & \text{else} \end{cases}$", which is called the *Kronecker Delta.* It is just defined as

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{else} \end{cases}.$$

It is used a lot, since it causes less effort to write. You can think of delta functions as the continuous equivalent of the Kronecker Delta. Using it, we would write orthogonality as

$$\langle \mathbf{e}_i, \mathbf{e}_j \rangle = \delta_{ij}.$$

In the following we will use the Kronecker Delta for notational convenience.

These two properties, i.e. normalized basis vectors and mutual orthogonality, make a basis especially easy to deal with. For that reason, those basis have an own name.

Definition (Orthonormal basis) A set of vectors $\mathbf{v}_1, ..., \mathbf{v}_n \in \mathbb{R}^n$ is called an *orthonormal basis* if each of them has unit length, i.e. $||\mathbf{v}_i|| = 1$ for $i = 1, ..., n$, and they are all mutually orthogonal:

$$\langle \mathbf{v}_i, \mathbf{v}_j \rangle = \delta_{ij}.$$

◊

Definition (Basis Expansion and Coordinates) Writing an arbitrary vector $\mathbf{x}$ as a sum of basis vectors $\mathbf{x} = \sum_{i=1}^{n} \lambda_i \mathbf{v}_i$ is called *basis expansion of* $\mathbf{x}$ *according to the basis* $\mathbf{v}_1, ..., \mathbf{v}_n$. The coefficients λ_i are called *coordinates of* $\mathbf{x}$ *according to the basis* $\mathbf{v}_1, ..., \mathbf{v}_n$.

◊

The reason, why orthonormal bases are so useful is that the coordinates of a vector are easy to compute: They are simply given by the projection of $\mathbf{x}$ onto each single basis vector. Therefore, the basis expansion of $\mathbf{x}$ with respect to an arbitrary orthonormal basis $\mathbf{v}_1, ..., \mathbf{v}_n \in \mathbb{R}^n$ is given by:

$$\begin{aligned} \mathbf{x} &= \sum_{i=1}^{n} \lambda_i \mathbf{v}_i \\ &= \sum_{i=1}^{n} \langle \mathbf{x}, \mathbf{v}_i \rangle \mathbf{v}_i. \end{aligned}$$

The term $\sum_{i=1}^{n} \langle \mathbf{x}, \mathbf{v}_i \rangle \mathbf{v}_i$ is called *orthonormal expansion* of $\mathbf{x}$.

Examples

- The orthonormal expansion of a vector $\mathbf{x} \in \mathbb{R}^3$ is given by

$$\begin{aligned} \mathbf{x} &= \left\langle \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \right\rangle \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + \left\langle \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \right\rangle \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \\ &\quad + \left\langle \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \right\rangle \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \\ &= x_1 \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + x_2 \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + x_3 \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} x_1 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ x_2 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ x_3 \end{pmatrix} \\ &= \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}. \end{aligned}$$

- For any fixed ϕ, the basis $\mathbf{r}_1 = \begin{pmatrix} \cos(\phi) \\ \sin(\phi) \end{pmatrix}, \mathbf{r}_2 = \begin{pmatrix} -\sin(\phi) \\ \cos(\phi) \end{pmatrix}$ is an orthonormal basis, since

$$\langle \mathbf{r}_1, \mathbf{r}_2 \rangle = -\sin(\phi)\cos(\phi) + \sin(\phi)\cos(\phi) = 0$$

and

$$\begin{aligned} ||\mathbf{r}_i|| &= \sqrt{\sin(\phi)^2 + \cos(\phi)^2} \\ &= 1 \text{ for } i = 1, 2. \end{aligned}$$

The coordinates of a vector $\mathbf{x}$ are given by $\lambda_1 = \langle \mathbf{r}_1, \mathbf{x} \rangle = x_1 \cos(\phi) + x_2 \sin(\phi)$ and $\lambda_2 = \langle \mathbf{r}_2, \mathbf{x} \rangle = -x_1 \sin(\phi) + x_2 \cos(\phi)$.

◁

From the example above, we can learn two important facts. First of all, $x_1, ..., x_n$ are the coordinates of the vector $\mathbf{x}$ according to the canonical basis. That sounds pretty trivial. However, it is important when we change from the canonical basis $\mathbf{e}_1, ..., \mathbf{e}_n$ to another orthonormal basis $\mathbf{v}_1, ..., \mathbf{v}_n$. This is the second observation: If we know which basis $\mathbf{v}_1, ..., \mathbf{v}_n$ we use, it is enough to remember the coordinates λ_i for $i = 1, ..., n$ and write them in vector notation $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_n)^\top$. We can just use these new coordinates $\boldsymbol{\lambda}$ to do our computations with $\mathbf{x}$ without bothering about the actual underlying basis. If we want to get back the representation in the canonical basis we can write it in the basis expansion $\mathbf{x} = \sum_{i=1}^{n} \lambda_i \mathbf{v}_i$. We will hear more about change of basis in the section about matrices.

2.1.5 A Note on Bases which are Not Orthonormal

As mentioned above, basically every set of linearly independent vectors can serve a basis. When working with a non-orthonormal basis $\mathbf{w}_1, ..., \mathbf{w}_n$, the basis can have non-normalized elements (i.e. $||\mathbf{w}_i|| \neq 1$), non-orthogonal elements (i.e. $\langle \mathbf{w}_i, \mathbf{w}_j \rangle \neq 0$ for $i \neq j$), or both.

Having only non-normalized basis vectors does not cause problems, since we can always normalize them by dividing by their norm and get a new orthonormal basis $\frac{1}{||\mathbf{w}_1||}\mathbf{w}_1, ..., \frac{1}{||\mathbf{w}_n||}\mathbf{w}_n$. Even if we do not do that, the new coordinates can simply be computed via $\lambda_i = \frac{1}{||\mathbf{w}_i||^2}\langle \mathbf{w}_i, \mathbf{x} \rangle$.

If the basis is non-orthogonal, the situation get more tricky. The problem is, that we cannot simply project our vector $\mathbf{x}$ onto the basis components to build the basis expansion. The reason is, that we also gets a bit of "$\mathbf{v}_i$-part" of $\mathbf{x}$ when computing the coordinate via $\lambda_j = \langle \mathbf{v}_j, \mathbf{x} \rangle$, if $\mathbf{v}_i$ and $\mathbf{v}_j$ are not orthogonal, i.e. not independent. We will not go into detail here, but show an example of what can happen, instead.

Example Consider the basis $\mathbf{w}_1 = \binom{1}{0}$, $\mathbf{w}_2 = \frac{1}{\sqrt{2}}\binom{1}{1}$. The basis has normal elements, since $||\mathbf{w}_1|| = 1$ and $\mathbf{w}_2 = \sqrt{2 \cdot \left(\frac{1}{\sqrt{2}}\right)^2} = 1$, but not orthogonal, since $\langle \mathbf{w}_1, \mathbf{w}_2 \rangle = \frac{1}{\sqrt{2}} \neq 0$. Assume we want to write the vector $\binom{0}{1}$ in as a orthonormal expansion in the basis $\mathbf{w}_1, \mathbf{w}_2$ we get

$$\begin{aligned}
\left\langle \mathbf{w}_1, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\rangle \mathbf{w}_1 + \left\langle \mathbf{w}_2, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\rangle \mathbf{w}_2 &= 0 \cdot \mathbf{w}_1 + \frac{1}{\sqrt{2}}\mathbf{w}_2 \\
&= \frac{1}{\sqrt{2}} \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \\
&= \frac{1}{2} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\
&\neq \begin{pmatrix} 0 \\ 1 \end{pmatrix}.
\end{aligned}$$

This show that we cannot use the scalar product trick to compute the coordinates of a vector according to a non-orthogonal basis.

◁

2.2 Matrices

Matrices are really nothing but a very convenient notation for (linear) transformations on vectors. For example, the transformations of stretching a vector, or rotating it, or reflecting it can all be written as matrix products.

Example 1 In the photoreceptors of the retina, color is represented by the activation of L, M and S cones (which are selective to 'red', 'green' and 'blue' light, respectively.) In a crude approximation, the classical view of color vision is that, at later processing stages, the color of a stimulus is rather represented by two 'color-opponent' channels, and a 'luminance' channel. Luminance could be computed by simply adding the activities of all cones: $Lum = L + M + S$. the 'red-green' channel could be computed by subtracting the M cone activity from the S cones, $RG = L - M$, and the 'blue yellow' channel by subtracting a combination of M and L from S: $BY = S - (M + L)$. To sum up:

$$\begin{aligned} RG &= L - M + 0 \cdot S \\ BY &= -L - M + S \\ Lum &= L + M + S. \end{aligned}$$

This can be written in matrix notation as

$$\begin{pmatrix} RG \\ BY \\ Lum \end{pmatrix} = \begin{pmatrix} 1 & -1 & 0 \\ -1 & -1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} L \\ M \\ S \end{pmatrix}.$$

So, multiplying the vector of cone-activities by this matrix returns the vector of channel-activities. This mapping from one 3 dimensional vector to another 3 dimensional vector can be summarized succinctly by a matrix product.

Now, we will formally define what we mean by a matrix, and by a matrix-vector product:

Definition (Matrix and matrix-vector product)

A matrix A of size m by n is a collection of numbers of the form

$$\mathbf{A} = \begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & & A_{2n} \\ \vdots & \vdots & \ldots & \vdots \\ A_{m1} & A_{m2} & \ldots & A_{mn} \end{pmatrix}.$$

The matrix product $y = Ax$ between an $m \times n$ matrix and an n dimensional vector x is defined to be

$$y = Ax = \begin{pmatrix} \sum_{i=1}^{n} A_{1i}x_i \\ \sum_{i=1}^{n} A_{2i}x_i \\ \vdots \\ \sum_{i=1}^{n} A_{mi}x_i \end{pmatrix}.$$

We often say that x is 'pre-multiplied' by A.

Note

- We can see that multiplying a vector by a matrix is really nothing but a couple of dot products: To compute the first element of y, we simply compute the dot product between x and the first row of $A : y_1 = \sum_{i=1}^{n} A_{1i}x_i = \langle A_{(1)}, x \rangle$, where we define $A_{(1)} = (A_{11}, A_{12}, \ldots, A_{1n})$.
- For matrix-vector multiplication to make sense, the sizes have to match: If x has n elements, we can only compute Ax if A is of size $m \times n$! Thus, if $m \neq n$, Ax is defined, but xA would not be defined!
- The result of multiplying an $m \times n$ matrix to a vector of size $n \times 1$ is always a vector of size $m \times 1$.

2.2.1 Reading a matrix

The 'canonical basis vectors' are defined to be those vectors that have exactly one entry equal to 1, and 0 everywhere else. For example, in $2D$, the canonical basis vectors are $e_1 = \binom{1}{0}$ and $e_2 = \binom{0}{1}$, i.e the vectors that define the x and $y-$axis, respectively. If we multiply a matrix M by e_1, we get back the first column of M : For example,

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

Therefore, the columns of a matrix are exactly the 'images' of the canonical basis vectors under matrix multiplication. From these images, we can calculate the image of any arbitrary vector x, as

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = x_1 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + x_2 \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Therefore,

$$\begin{aligned} Mx &= M\left(x_1 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + x_2 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) \\ &= x_1 \begin{pmatrix} 2 \\ 1 \end{pmatrix} + x_2 \begin{pmatrix} 3 \\ 1 \end{pmatrix}. \end{aligned}$$

Hence, the image of any vector x under multiplication by the matrix M can always be written as a linear combination of the columns of M.

Examples of matrices

- The identify matrix is a boring matrix which does not do anything. Multiplying any vector by it leaves it unchanged:

$$\begin{aligned}\mathbf{I} &= (\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_n) \\ &= \begin{pmatrix} 1 & 0 & \dots & 0 \\ \vdots & 1 & \ddots & 0 \\ 0 & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}.\end{aligned}$$

  Despite its boringness, this matrix occurs so frequently that we give it its own symbol, namely uppercase **I**. This should not be confused with images, that we also denote by **I**. However, it should be clear from the context what we mean by **I**.
  The result of multiplying a vector $\mathbf{x}$ with the identity matrix is just the vector itself:

$$\begin{aligned}\mathbf{Ix} &= \begin{pmatrix} 1 & 0 & \dots & 0 \\ \vdots & 1 & \ddots & 0 \\ 0 & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{e}_1^\top \mathbf{x} \\ \vdots \\ \mathbf{e}_n^\top \mathbf{x} \end{pmatrix} \\ &= \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \mathbf{x}.\end{aligned}$$

- Matrices that look very similar to the identity matrix, are *permutation matrices*. A permutation simply changes the order of coordinates of a vector $\mathbf{x}$. Let us look at a 3D example. Let $\mathbf{x} = (x_1, x_2, x_3)^\top$ and let f be the permutation that swaps x_1 and x_2, i.e. $f(\mathbf{x}) = (x_2, x_1, x_3)$. The corresponding permutation matrix is given by

$$\mathbf{P} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Multiplying a three dimensional vector $\mathbf{x} = (x_1, x_2, x_3)^\top$ with our example of a permutation matrix yields

$$\begin{aligned}\mathbf{Px} &= \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{e}_2^\top \mathbf{x} \\ \mathbf{e}_1^\top \mathbf{x} \\ \mathbf{e}_3^\top \mathbf{x} \end{pmatrix} \\ &= \begin{pmatrix} x_2 \\ x_1 \\ x_3 \end{pmatrix}.\end{aligned}$$

- In geometry, *rotation matrices* play an important role: For a fixed a rotation angle φ, rotating vectors is linear mapping, since it does not matter if we add two vectors, rotate the result by φ, or do it the other way around. In 2D there is only one rotation matrix for a given angle φ:

$$\mathbf{R} = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix}.$$

  In 3D we have three coordinate planes where we can rotate a vector in, i.e. the x_1x_2-, the x_1x_3- and the x_2x_3-plane. In ND we have even more. However, the corresponding rotation matrices are easy to remember. You can build them by taking the identity matrix and replacing the entries corresponding to the coordinate plane you want to rotate in by the entries that you would use for the 2D rotation matrix. For example, the rotation matrix for the x_1x_2-plane in 3D is given by

$$\mathbf{R} = \begin{pmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

  the matrix for a rotation in the x_1x_3-plane is given by

$$\mathbf{R} = \begin{pmatrix} \cos\varphi & 0 & -\sin\varphi \\ 0 & 1 & 0 \\ \sin\varphi & 0 & \cos\varphi \end{pmatrix}.$$

  Multiplying a vector with a rotation matrix, means rotating this vector. Let us look at one simple example in detail: Rotating the vector $\binom{1}{0} \in \mathbb{R}^2$ about 90°. From their definition, we can get the appropriate rotation matrix:

$$\begin{aligned}\mathbf{R} &= \begin{pmatrix} \cos 90° & -\sin 90° \\ \sin 90° & \cos 90° \end{pmatrix} \\ &= \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}.\end{aligned}$$

Now we can apply it to our vector $(1,0)^\top$:

$$\mathbf{R}\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix}.$$

We see that the resulting vector has been rotated by 90°. Multiplying it again with $\mathbf{R}$, yields

$$\begin{aligned}\mathbf{R}\mathbf{R}\begin{pmatrix}1\\0\end{pmatrix} &= \mathbf{R}\begin{pmatrix}0\\1\end{pmatrix}\\ &= \begin{pmatrix}0\\-1\end{pmatrix},\end{aligned}$$

which is indeed a rotation about $2 \cdot 90° = 180°$. In general this choice of $\mathbf{R}$ will rotate any vector in $\mathbb{R}^2$ about 90° around the origin. If we choose φ differently, the resulting matrix will rotate any vector in $\mathbb{R}^2$ about φ.
In 3D the example works just the same. You might want to try it with a few examples in order to get used to multiplying vectors with matrices.

- The matrix that corresponds to the projection of a vector $\mathbf{x} \in \mathbb{R}^3$ onto the x_1x_2-plane, is given by

$$\mathbf{P}_{12} = \begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\end{pmatrix}.$$

Note, that the third column is zero since projecting the vector $(0,0,1)^\top$ onto the x_1x_2-plane yields the vector $(0,0)^\top$.
Applying the matrix, that projects every vector $\mathbf{x} \in \mathbb{R}^3$ onto the x_1x_2-plane, to an arbitrary vector $\mathbf{x} \in \mathbb{R}^3$ yields

$$\begin{aligned}\mathbf{P}_{12}\mathbf{x} &= \begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\end{pmatrix}\begin{pmatrix}x_1\\x_2\\x_3\end{pmatrix}\\ &= \begin{pmatrix}x_1\\x_2\end{pmatrix},\end{aligned}$$

which is indeed the projection of $\mathbf{x} = (x_1, x_2, x_3)^\top$ onto the x_1x_2-plane.

- Any scalar product can be thought of as a matrix product: The scalar product between two vectors v and w is defined to be

$$\langle v, w\rangle = \sum_i v_i w_i.$$

We can see that this is equivalent to $v^\top w = \sum_i v_i w_i$, i.e. of pre-multiplying w be the transpose of v,$v^\top$.

2.2.2 Every linear function can be written as a matrix product:

Example Remember our very simple linear response model for a V1 simple cell. Since the number of pixels of the presented image does not really matter, let us only consider 2×2 input images. As shown in one of the examples above, we can represent any of those input images by a four-dimensional vector $\mathbf{v} = (I_{11}, I_{21}, I_{12}, I_{22})^\top$. In contrast to the previous example, we now consider the spiking rate of three instead of only one neuron. By assuming that the relation between input image and spiking rate of our neurons is linear, our model for the spiking rates given an image is a linear function $f : \mathbb{R}^4 \to \mathbb{R}^3$.

Assume now that we measured the response of our three neurons for the four images

$$\begin{aligned} \mathbf{I}_1 &= \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \\ \mathbf{I}_2 &= \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \\ \mathbf{I}_3 &= \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \\ \mathbf{I}_4 &= \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}, \end{aligned}$$

which yielded us four spike rate measurements $\mathbf{r}_1 = \begin{pmatrix} r_{11} \\ \vdots \\ r_{31} \end{pmatrix}$, $\mathbf{r}_2 = \begin{pmatrix} r_{12} \\ \vdots \\ r_{32} \end{pmatrix}$, $\mathbf{r}_3 = \begin{pmatrix} r_{13} \\ \vdots \\ r_{33} \end{pmatrix}$ and $\mathbf{r}_4 = \begin{pmatrix} r_{14} \\ \vdots \\ r_{34} \end{pmatrix}$.

Note that the vectors $\mathbf{v}_1, ..., \mathbf{v}_4$, which correspond to the four input images $\mathbf{I}_1, ..., \mathbf{I}_n$, are really just the canonical basis of $\mathbb{R}^4$. Assume now that we get another image $\tilde{\mathbf{I}}$ and we are asked to predict the responses of our three neurons. In the linear model, we can use the already measured responses $\mathbf{r}_1, ..., \mathbf{r}_4$ to predict the response $\tilde{\mathbf{r}}$. The strategy for that is as follows: First we express the vector notation $\tilde{\mathbf{v}}$ of the new image as a linear combination of the vector notations $\mathbf{v}_1, ..., \mathbf{v}_4$ of the images $\mathbf{I}_1, ..., \mathbf{I}_4$:

$$\begin{aligned} \tilde{\mathbf{v}} &= \sum_{i=1}^{4} \tilde{v}_i \mathbf{v}_i \\ &= \tilde{I}_{11} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} + \tilde{I}_{21} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} + \tilde{I}_{12} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} + \tilde{I}_{22} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}. \end{aligned}$$

Then we compute the response of our three neurons via:

$$\begin{aligned} f\left(\sum_{i=1}^{4}\tilde{v}_i\mathbf{v}_i\right) &\overset{f \text{ linear}}{=} \sum_{i=1}^{4}\tilde{v}_i f\left(\mathbf{v}_i\right) \\ &= \sum_{i=1}^{4}\tilde{v}_i\mathbf{r}_i \\ &= \begin{pmatrix} \sum_{i=1}^{4}\tilde{v}_i r_{1i} \\ \sum_{i=1}^{4}\tilde{v}_i r_{2i} \\ \sum_{i=1}^{4}\tilde{v}_i r_{3i} \end{pmatrix}. \end{aligned}$$

As we can see, using the linearity of f, four measurements of linear independent inputs are enough to determine the output of an arbitrary four dimensional image. Therefore, our linear function is completely determined by our four measurements.

◁

This example shows a very important aspect of n-dimensional linear functions. Now, we state this observation in a more general way.

We can always define the matrix of a mapping with respect to any basis: Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be a linear function which takes in an n dimensional vector and returns an m dimensional one. We know that each n-dimensional vector $\mathbf{x} \in \mathbb{R}^n$ can be described in terms of a linear combination $\mathbf{x} = \sum_{i=1}^{n}\lambda_i\mathbf{b}_i$ of n basis vectors $\mathbf{b}_1, ..., \mathbf{b}_n$. Since we can exchange the summation and function symbol for linear functions we can do the same operation as for the one-dimensional case:

$$\begin{aligned} f_k(\mathbf{x}) &= f_k\left(\sum_{i=1}^{n}\lambda_i\mathbf{b}_i\right) \\ &= \sum_{i=1}^{n}\lambda_i f_k(\mathbf{b}_i), \text{ for } k = 1, ..., n. \end{aligned}$$

Here, $f_k(\mathbf{x})$ denotes the kth coordinate of the output vector $f(\mathbf{x})$. Therefore, we can determine the function value of an arbitrary vector $\mathbf{x}$ by first expressing $\mathbf{x}$ in terms of the $\mathbf{b}_1, ..., \mathbf{b}_n$, therefore getting the coordinates $\lambda_1, ..., \lambda_n$, and then building a linear combination $\sum_{i=1}^{n}\lambda_i f_k(\mathbf{b}_i)$ of the known function values $f_k(\mathbf{b}_i)$. This is the basic mathematical mechanic behind matrices.

Definition (Matrix of a linear mapping) Given an n-dimensional linear function $f : \mathbb{R}^n \to \mathbb{R}^m$ and a basis for $\mathbf{b}_1, ..., \mathbf{b}_n \in \mathbb{R}^n$, the matrix of the function

f according to the basis $\mathbf{b}_1, ..., \mathbf{b}_n$

$$\mathbf{A} = \begin{pmatrix} f_1(\mathbf{b}_1) & f_1(\mathbf{b}_2) & f_1(\mathbf{b}_{n-1}) & f_1(\mathbf{b}_n) \\ \vdots & f_2(\mathbf{b}_2) & & f_2(\mathbf{b}_n) \\ f_{m-1}(\mathbf{b}_1) & \vdots & \dots & \vdots \\ f_m(\mathbf{b}_1) & f_m(\mathbf{b}_2) & \dots & f_m(\mathbf{b}_n) \end{pmatrix}$$

stores the function values $f_k(\mathbf{b}_i)$ for $i = 1, ..., n$ and $k = 1, ..., m$, i.e. the jth column is the output of f on the jth basis vector. These function values are everything that is needed to compute the outcome of the linear function for an arbitrary input.

$\Diamond$

This means that each n-dimensional linear function $f : \mathbb{R}^n \to \mathbb{R}^m$ can be expressed in terms of a $m \times n$ matrix if we choose a basis $\mathbf{b}_1, ..., \mathbf{b}_n$. On the other hand, each $m \times n$ matrix, i.e. an arbitrary $m \times n$ grid of numbers, determines a linear function for a given basis $\mathbf{b}_1, ..., \mathbf{b}_n$.

It is important to understand that a matrix is always with respect to a certain basis. If the basis changes, the matrix changes. The linear function, however, does not.

From now on, unless explicitly noted, we use the canonical basis of $\mathbf{e}_1, ..., \mathbf{e}_n \in \mathbb{R}^n$ when dealing with matrices, i.e.

$$\mathbf{A} = \begin{pmatrix} f_1(\mathbf{e}_1) & f_1(\mathbf{e}_2) & f_1(\mathbf{e}_{n-1}) & f_1(\mathbf{e}_n) \\ \vdots & f_2(\mathbf{e}_2) & \ddots & f_2(\mathbf{e}_n) \\ f_{m-1}(\mathbf{e}_1) & \vdots & \dots & \vdots \\ f_m(\mathbf{e}_1) & f_m(\mathbf{e}_2) & \dots & f_m(\mathbf{e}_n) \end{pmatrix}.$$

2.2.3 Multiplying two matrices

$\triangleleft$

2.2.3.1 Matrix-Matrix Product

In the above example about the rotation matrices, we rotated a vector by 180° degrees by rotating it twice by 90° degrees with our rotation matrix

$$\begin{aligned} \mathbf{R} &= \begin{pmatrix} \cos 90° & -\sin 90° \\ \sin 90° & \cos 90° \end{pmatrix} \\ &= \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}. \end{aligned}$$

The way we did this was multiplying a vector with $\mathbf{R}$ and doing the same operation with the outcome again. However, there is also another way for

computing this outcome: By multiplying the matrix $\mathbf{R}$ with itself and obtain a new matrix $\mathbf{R}'$ that corresponds to a rotation about 180° degrees. Speaking in terms of functions, the matrix $\mathbf{R}' = \mathbf{R} \cdot \mathbf{R}$ corresponds to the linear function $rot_{180}(\mathbf{x}) = rot_{90}(rot_{90}(\mathbf{x}))$ if rot_φ denotes the function that rotates a vector by φ. This means that we obtain the matrix of a composition of two functions by multiplying their matrices. Multiplying two matrices is fairly easy: We just treat each of its columns as a vector on its own and multiply it with the matrix. The result of this multiplication determines one column of the matrix-matrix product.

Definition (Matrix-Matrix Product) The product of a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ with another matrix $\mathbf{B}^{n \times k}$ is given by

$$\begin{aligned}
\mathbf{A} \cdot \mathbf{B} &= \begin{pmatrix} \sum_{i=1}^n \mathbf{A}_{1i}\mathbf{B}_{i1} & \sum_{i=1}^n \mathbf{A}_{1i}\mathbf{B}_{i2} & \dots & \sum_{i=1}^n \mathbf{A}_{1i}\mathbf{B}_{in} \\ \sum_{i=1}^n \mathbf{A}_{2i}\mathbf{B}_{i1} & \ddots & & \vdots \\ \vdots & & \ddots & \sum_{i=1}^n \mathbf{A}_{(m-1)i}\mathbf{B}_{in} \\ \sum_{i=1}^n \mathbf{A}_{mi}\mathbf{B}_{i1} & \dots & \sum_{i=1}^n \mathbf{A}_{mi}\mathbf{B}_{i(n-1)} & \sum_{i=1}^n \mathbf{A}_{mi}\mathbf{B}_{in} \end{pmatrix} \\
&= \begin{pmatrix} \langle \mathbf{a}^1, \mathbf{b}_1 \rangle & \langle \mathbf{a}^1, \mathbf{b}_2 \rangle & \dots & \langle \mathbf{a}^1, \mathbf{b}_n \rangle \\ \langle \mathbf{a}^2, \mathbf{b}_1 \rangle & \ddots & & \vdots \\ \vdots & & \ddots & \langle \mathbf{a}^{m-1}, \mathbf{b}_n \rangle \\ \langle \mathbf{a}^m, \mathbf{b}_1 \rangle & \dots & \langle \mathbf{a}^m, \mathbf{b}_{n-1} \rangle & \langle \mathbf{a}^m, \mathbf{b}_n \rangle \end{pmatrix},
\end{aligned}$$

where $\mathbf{a}^i$ denotes the ith row and $\mathbf{b}_j$ the jth column of $\mathbf{A}$ and $\mathbf{B}$, respectively. We can see from the definition of the matrix-matrix product that the number of columns of $\mathbf{A}$ and the number of rows of $\mathbf{B}$ must match.

$\Diamond$

Notes

- We can multiply two matrices if the 'length' of the first matches the 'height' of the second: $C = AB$ is only defined if A is of size $m \times n$ and B is of size $n \times k$. Then, the outcome C will be of size $m \times k$. ('Inner dimensions must match and drop out').
- Matrix multiplication is not commutative, in general, $AB \neq BA$! Thinking of matrix multiplication as applying a linear functions, this makes a lot of sense: For example, suppose that our matrix A swaps the first and second dimension of a two-dimensional vector, and that B doubles its first dimension. Clearly, it does matter whether we switch dimensions before or after doubling the first dimension.

Examples

- Let

$$\mathbf{R} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

as above. The product of $\mathbf{R}$ with itself is given by

$$\begin{aligned} \mathbf{R}\cdot\mathbf{R} &= \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \\ &= \begin{pmatrix} \left\langle \binom{0}{-1}, \binom{0}{1} \right\rangle & \left\langle \binom{0}{-1}, \binom{-1}{0} \right\rangle \\ \left\langle \binom{1}{0}, \binom{0}{1} \right\rangle & \left\langle \binom{1}{0}, \binom{-1}{0} \right\rangle \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \end{aligned}$$

- Let $A = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}, B = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, then

$$\begin{aligned} AB &= \begin{pmatrix} 0 & 2 \\ 1 & 0 \end{pmatrix}, \\ BA &= \begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix}. \end{aligned}$$

- Matrices do not have to be 'square' to be multiplied:

$$\begin{pmatrix} 1 & 0 & 1 \\ 3 & 2 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 2 & 2 \\ 6 & 4 \end{pmatrix}.$$

- Again, the identity matrix likes to be boring: Multiplying a $m \times n$ matrix $\mathbf{A}$ with the identity matrix $\mathbf{I}_m$ from the left or $\mathbf{I}_n$ from the right (n and m denote the dimension of the identity matrix) leaves the matrix $\mathbf{A}$ unchanged, i.e. $\mathbf{A}\mathbf{I}_n = \mathbf{I}_m\mathbf{A} = \mathbf{A}$.

- We can derived the addition theorems for sine and cosine from the rule of matrix multiplication: Clearly, rotating a vector first by an angle of α, and then by β, should yield exactly the same result as rotating it by $\alpha+\beta$ in one go. If we denote the matrix which rotates by α by $R(\alpha)$, we get

$$\begin{aligned} R(\alpha+\beta) &= R(\alpha)R(\beta) \\ \begin{pmatrix} \cos(\alpha+\beta) & -\sin(\alpha+\beta) \\ \sin(\alpha+\beta) & \cos(\alpha+\beta) \end{pmatrix} &= \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix}\begin{pmatrix} \cos(\beta) & -\sin(\beta) \\ \sin(\beta) & \cos(\beta) \end{pmatrix} \\ &= \begin{pmatrix} \cos(\alpha)\cos(\beta) - \sin(\alpha)\sin(\beta) & -\cos(\alpha)\sin(\beta) - \cos(\beta)\sin(\alpha) \\ \cos(\alpha)\sin(\beta) + \cos(\beta)\sin(\alpha) & \cos(\alpha)\cos(\beta) - \sin(\alpha)\sin(\beta) \end{pmatrix} \end{aligned}$$

Equation coefficients, we recover the addition theorems

$$\begin{aligned}\cos(\alpha+\beta) &= \cos(\alpha)\cos(\beta)-\sin(\alpha)\sin(\beta)\\ \sin(\alpha+\beta) &= \sin(\alpha)\cos(\beta)+\cos(\alpha)\sin(\beta).\end{aligned}$$

- Assume that we are searching for a matrix that projects a vector $\mathbf{x}\in\mathbb{R}^3$ onto the x_1x_2-plane and rotates the result by 30°around the origin. We can get this matrix $\mathbf{C}$ by multiplying the rotation matrix

$$\mathbf{R} = \begin{pmatrix}\cos(30°) & -\sin(30°)\\ \sin(30°) & \cos(30°)\end{pmatrix}$$

with the projection matrix

$$\mathbf{P} = \begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\end{pmatrix}.$$

The result is given by

$$\mathbf{R}\cdot\mathbf{P} = \begin{pmatrix}\cos(30°) & -\sin(30°) & 0\\ \sin(30°) & \cos(30°) & 0\end{pmatrix}.$$

Note that the order in which we multiply the matrices is the reverse order in which we apply the transformations, i.e. projection and rotation. The reason is simply that a vector $\mathbf{x}$ is multiplied with $\mathbf{R}\cdot\mathbf{P}$ from the right, which means that it is multiplied with $\mathbf{P}$ first and with $\mathbf{R}$ afterwards. This is exactly the order in which we wanted to apply the transformations.

◁

With the rule for matrix-matrix multiplication we can also multiply a vector from the left to a matrix. However, since the number of columns of the first matrix must match the number of row of the second matrix in a product, we cannot use column vectors, but only row vectors. This is one example, where it makes a difference if we have a row or a column vector. Since a row vector $\mathbf{x}=(x_1,...,x_m)$ can be treated as a $1\times m$-matrix, we can apply the rule for matrix-matrix multiplication and obtain

$$\begin{aligned}\mathbf{xA} &= (x_1,...,x_n)\cdot\mathbf{A}\\ &= (\langle\mathbf{x},\mathbf{a}_1\rangle,...,\langle\mathbf{x},\mathbf{a}_k\rangle),\end{aligned}$$

if $\mathbf{a}_i$ denotes the ith column of $\mathbf{A}$ as before.

2.2.3.2 Matrix Transposition

It is often convenient (or necessary) to 'flip' a matrix or a vector by exchanging its rows and columns. For example, we might want to write a column vector $n\times 1$ as a row vector of size $1\times n$. Formally, this operation is called 'taking the transpose':

Definition (Matrix Transpose) The *transpose* of a matrix

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & & \mathbf{A}_{1n} \\ \mathbf{A}_{21} & & & \\ \vdots & & & \mathbf{A}_{(m-1)n} \\ \mathbf{A}_{m1} & & \mathbf{A}_{m(n-1)} & \mathbf{A}_{mn} \end{pmatrix}$$

is given by

$$\mathbf{A}^\top = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{21} & & \mathbf{A}_{m1} \\ \mathbf{A}_{12} & & & \\ \vdots & & & \mathbf{A}_{m(n-1)} \\ \mathbf{A}_{1n} & & \mathbf{A}_{(m-1)n} & \mathbf{A}_{mn} \end{pmatrix}.$$

If A is of size $m \times n$, $A^\top$is of size $n \times m$.

Note

- The usual dot-product between two vectors can be written as $\langle v, w \rangle = v^\top w$.
- If we take the transpose of a product, we have to reverse the order: $(ABC)^\top = C^\top B^\top A^\top$.

$\Diamond$

2.2.3.3 Symmetric Matrices, Covariance Matrix and Outer Product of Vectors and Matrices

In this section we look at a special way of using the matrix notation that might not be immediately obvious when looking at the calculation rules for the first time. This way of using the matrix notation is especially useful for computing covariance matrices. Therefore, will develop the idea of the *outer product* with the example of computing a covariance matrix in matrix notation. Let us start with the definition of a covariance matrix.

Definition (Covariance Matrix) Let $\mathsf{X} = (\mathsf{X}_1, ..., \mathsf{X}_n)^\top$ be an n-dimensional random variable. The covariance of dimension i with dimension j is given by

$$\begin{aligned} Cov(\mathsf{X}_i, \mathsf{X}_j) &= E\left((\mathsf{X}_i - E(\mathsf{X}_i)) \cdot (\mathsf{X}_j - E(\mathsf{X}_j))\right) \\ &= E(\mathsf{X}_i \cdot \mathsf{X}_j) - E(\mathsf{X}_i) \cdot E(\mathsf{X}_j). \end{aligned}$$

The covariance tells us how X_i behaves if we change X_j and vice versa. The single covariances can be put together in the $n \times n$ *covariance matrix*

$$\begin{aligned}\mathbf{C} &= \begin{pmatrix} Cov(\mathsf{X}_1, \mathsf{X}_1) & \dots & Cov(\mathsf{X}_1, \mathsf{X}_n) \\ \vdots & \ddots & \vdots \\ Cov(\mathsf{X}_n, \mathsf{X}_1) & \dots & Cov(\mathsf{X}_n, \mathsf{X}_n) \end{pmatrix} \\ &= \begin{pmatrix} Var(\mathsf{X}_1) & \dots & Cov(\mathsf{X}_1, \mathsf{X}_n) \\ \vdots & \ddots & \vdots \\ Cov(\mathsf{X}_n, \mathsf{X}_1) & \dots & Var(\mathsf{X}_n) \end{pmatrix}.\end{aligned}$$

Empirically, i.e. given N sample vectors $\mathbf{x}_1, ..., \mathbf{x}_n$, the single covariances can be computed via

$$\begin{aligned}\hat{Cov}(\mathsf{X}_i, \mathsf{X}_j) &= \frac{1}{N} \sum_{\ell=1}^{N} (x_{\ell i} - \mu_i)(x_{\ell j} - \mu_j) \\ &= \hat{\mathbf{C}}_{ij}.\end{aligned}$$

$\Diamond$

For simplicity, let us assume that our sample has mean zero, i.e. $\boldsymbol{\mu} = 0$. This is no restriction since we can always subtract $\boldsymbol{\mu}$ from each $\mathbf{x}_i$ in order to center them around zero. For $\boldsymbol{\mu} = 0$, the above equation for the empirical mean becomes

$$\begin{aligned}\hat{Cov}(\mathsf{X}_i, \mathsf{X}_j) &= \frac{1}{N} \sum_{\ell=1}^{N} x_{\ell i} x_{\ell j} \\ &= \hat{\mathbf{C}}_{ij}.\end{aligned}$$

Here, we use the hat on a variable to indicate that it has been estimated from empirical data. Covariance matrices have a certain property: They are *symmetric*, i.e. the original matrix equals its transpose $\mathbf{C} = \mathbf{C}^\top$. Symmetric matrices have certain nice properties that we will encounter later in the section about eigenvalues and eigenvectors.

Now let us return to the question of how to express the computation of the empirical covariance matrix with a single matrix multiplication. We do that in two steps: first we see how to generate the inner term $x_{\ell i} x_{\ell j}$ of the sum $\frac{1}{N} \sum_{\ell=1}^{N} x_{\ell i} x_{\ell j}$ and then expand this idea in order to get the full empirical covariance matrix. To answer the first question, we must find an operation of a vector $\mathbf{x}_\ell$ with itself such that the result is a matrix $\tilde{\mathbf{C}}^{(\ell)}$ with the entries $\tilde{\mathbf{C}}^{(\ell)}_{ij} = x_{\ell i} x_{\ell j}$. The crucial observation for that is that we can interpret a single entry $x_{\ell i}$ as a 1×1 vector. With this in mind, we can apply the matrix product rule to the term $\mathbf{x}_\ell \mathbf{x}_\ell^\top$. The dot product between the row and the column vector simply becomes a multiplication of two scalars $x_{\ell i} x_{\ell j}$. Note, that the vectors have the reverse order than in the dot product, i.e. the column vector is the first

factor and the row vector is the second factor. The result of this *outer product* is the desired matrix:

$$\begin{aligned}\tilde{\mathbf{C}}^{(\ell)} &= \mathbf{x}_\ell \mathbf{x}_\ell^\top \\ &= \begin{pmatrix} x_{\ell 1}x_{\ell 1} & \dots & x_{\ell 1}x_{\ell n} \\ \vdots & \ddots & \vdots \\ x_{\ell n}x_{\ell 1} & \dots & x_{\ell n}x_{\ell n} \end{pmatrix}.\end{aligned}$$

Looking at the term for the empirical covariance matrix, we see that it can now be computed via

$$\hat{\mathbf{C}} = \tfrac{1}{N} \sum_{\ell=1}^{N} \tilde{\mathbf{C}}^{(\ell)}.$$

In order to be able to express this sum as multiplication of two matrices, we must arrange our data in an matrix such that the sum of the matrix multiplication equals the above sum. Placing all our measurements $\mathbf{x}_1, ..., \mathbf{x}_N$ as rows in a $N \times n$ matrix $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_N)^\top$, we can see that

$$\mathbf{X}^\top \mathbf{X} = \sum_{\ell=1}^{N} \tilde{\mathbf{C}}^{(\ell)}.$$

In order to see this, we must realize that an entry $(\mathbf{X}^\top \mathbf{X})_{ij}$ is given by the dot product between an N-dimensional vector that contains the ith entry of all our measurements and a vector of the same size that contains the jth entry of all our measurements, i.e.

$$(\mathbf{X}^\top \mathbf{X})_{ij} = (x_{1i}, x_{2i}, ..., x_{Ni}) \cdot (x_{1j}, x_{2j}, ..., x_{Nj}) = \sum_{\ell=1}^{N} x_{\ell i} x_{\ell j}.$$

This is exactly what we intended. Therefore, computing the covariance matrix from N measurements from an n-dimensional random variable by a single matrix multiplication can be done via

$$\hat{\mathbf{C}} = \frac{1}{N} \mathbf{X}^\top \mathbf{X}.$$

2.3 Invertibility, Inverses and Rank

2.3.1 The Inverse of a matrix

Suppose we have given a linear system of equations, $y = Mx$, where M is a given $n \times n$ matrix, and y is some knows vector. For example, lets say we want to find a vector x such that

$$\binom{1}{0} = \begin{pmatrix} 2 & 3 \\ -1 & 4 \end{pmatrix} \binom{x_1}{x_2}.$$

How can we find x? We know that the outcome of multiplying x with the matrix M gives us the vector $y = \binom{1}{0}$, but is that enough for determining x? Matrices for which we can 'recover' x from knowing the outcome of the matrix multiplication Mx are said to be invertible. Formally:

Definition:

A square matrix M is said to be invertible if there exists a second matrix, called M^{-1}, which is such that

$$MM^{-1} = M^{-1}M = I_n.$$

Notes:

- If M is invertible, the linear system of equations $y = Mx$ has the unique solution $x = M^{-1}y$.
- In the example above, $M^{-1} = \frac{1}{11}\begin{pmatrix} 4 & -3 \\ 1 & 2 \end{pmatrix}$, so $x = M^{-1}y = \frac{1}{11}\begin{pmatrix} 4 & -3 \\ 1 & 2 \end{pmatrix}\binom{1}{0} = \frac{1}{11}\binom{4}{1}$.

2.3.2 Inverses and Determinants

For matrices of size 2×2, we can directly derive the inverse by hand: Given a matrix $M = \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix}$, we want to find $M^{-1} = \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix}$such that $MM^{-1} = I$, i.e.

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

Writing this matrix product out line by line yields for equations in four unknowns, which can be solved yielding

$$M^{-1} = \frac{1}{m_{11}m_{22} - m_{12}m_{21}}\begin{pmatrix} m_{22} & -m_{12} \\ -m_{21} & m_{11} \end{pmatrix}.$$

Clearly, this formula only makes sense if $m_{11}m_{22} - m_{12}m_{21}$ does not equal zero. So, by computing this term, we can immediately see whether a 2×2 matrix

is invertible, or not. It is therefore given a special name, it is the *determinant* of M.

Definition: Determinant

The determinant of a 2×2 matrix M is defined as $\det(M) = m_{11}m_{22} - m_{12}m_{21}$. In general, a matrix is invertible if and only if its determinant is non-zero.

Notes:

- For a 3×3 matrix A, we have that

$$\det \mathbf{A} = A_{11}A_{22}A_{33} + A_{12}A_{23}A_{31} + A_{13}A_{21}A_{32} - A_{31}A_{22}A_{13} - A_{32}A_{23}A_{11} - A_{33}A_{21}A_{12}.$$

- If $\mathbf{D}$ is an $n \times n$ diagonal matrix, i.e. a matrix that is only non-zero on the diagonal, then the determinant is given by the product of the diagonal entries:

$$\det \mathbf{D} = \prod_{i=1}^{n} D_{ii}.$$

  This statements makes sense: The inverse of a diagonal matrix $D = \begin{pmatrix} d_{11} & 0 & 0 \\ 0 & d_{22} & 0 \\ 0 & 0 & d_{33} \end{pmatrix}$ can be written directly as $D^{-1} = \begin{pmatrix} 1/d_{11} & 0 & 0 \\ 0 & 1/d_{22} & 0 \\ 0 & 0 & 1/d_{33} \end{pmatrix}$, but this construction only works if all diagonal entries are different from zero. On the other hand, as the determinant is the product of all diagonal entries, it will only be non-zero if all of the diagonal entries are non-zero.

- If $\mathbf{A}$ is an $n \times n$ matrix, for which all entries below or above the diagonal are zero, then the determinant is also given by the product of the diagonal entries:

$$\det \mathbf{A} = \prod_{i=1}^{n} A_{ii}.$$

Examples

1. The determinant of the matrices

$$\mathbf{A}_1 = \begin{pmatrix} 2 & 3 \\ 4 & 2 \end{pmatrix}$$

   and

$$\mathbf{A}_2 = \begin{pmatrix} 2 & 4 \\ 1 & 2 \end{pmatrix},$$

   are $\det \mathbf{A}_1 = 2 \cdot 2 - 4 \cdot 3 = -8$ and $\det \mathbf{A}_2 = 2 \cdot 2 - 1 \cdot 4 = 0$.

2. The determinant of the identity matrix $\mathbf{I}$ is $\det \mathbf{I} = 1$, since the product over the diagonal elements is one.

3. This last example shows a case where determinants are used in statistics: The density function of a multivariate n-dimensional Gaussian with mean $\boldsymbol{\mu}$ and covariance $\mathbf{C}$ is given by

$$p(\mathsf{X}|\boldsymbol{\mu}, \mathbf{C}) \quad = \quad \frac{1}{(2\pi)^{\frac{n}{2}}\sqrt{\det \mathbf{C}}} \exp\left(\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right).$$

   Here, the square root of the determinant, i.e. the volume spanned by the column vectors of $\mathbf{C}$ is used to normalize the probability density. This is very similar to the one-dimensional case, where the square root of the variance, i.e. $\sigma = \sqrt{\sigma^2}$, takes over this role.

Properties of determinants: Let $\mathbf{C} = \mathbf{A} \cdot \mathbf{B}$, then

1. $\det \mathbf{C} = \det(\mathbf{AB}) = \det \mathbf{A} \cdot \det \mathbf{B}$
2. $\det(\mathbf{A}^{-1}) = \det(\mathbf{A})^{-1}$
3. $\det \mathbf{A} = \det \mathbf{A}^\top$.

2.3.3 When is a matrix invertible?

In the above, we stated a general rule for determining whether a matrix is invertible, namely computing its determinant. Now, we want to get more intuition into what 'makes' a matrix invertible. If a matrix is invertible, then given any vector y, we can always find a unique x such that $y = Mx$. When could this fail?

Suppose that we have any vector x that is such that $Mx = 0$. (With 0, we here mean a vector that has all entries equal to 0). Then clearly, $M(2x) = 0$ also, or $M(\alpha x) = 0$ for any number α. In this case, we would have non-uniqueness, as $Mx = M(2x)$ but clearly $x \neq 2x$, for $x \neq 0$.

Alternatively, one possible scenario is that we have two vectors x and z which are such that $y = Mx = Mz$. If that is the case, if we are only given the outcome y, there is know way of determining whether x or z went into the matrix-multiplication. Actually, this scenario is the same as the previous one: If $Mx = My$, then $M(x-y) = 0$, so we have found a vector $x - y$ which is not equal to 0, but $M(x-y)$ is 0.

In general, we have the statement that

Invertibility and the null-space:

A square matrix M is not invertible exactly we can find a vector $x \neq 0$ for which $Mx = 0$. A vector is said to belong to the *null-space* of M is $Mx = 0$.

An example

Lets consider the matrix $A = \begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix}$. A maps the first canonical basis vector $\binom{1}{0}$ to $\binom{1}{2}$, and the second basis vector $\binom{0}{1}$ is mapped to $\binom{2}{4} = 2\binom{1}{2}$. As Ax can always be written as $Ax = x_1\binom{1}{2} + x_2\binom{2}{4} = (x_1 + 2x_2)\binom{1}{2}$, we can see that the image of any vector will be proportional to $\binom{1}{2}$. This implies, e.g. that $A\binom{-2}{1} = (-2+2)\binom{1}{2} = 0$. So, we can see that a matrix if not invertible if its columns do not 'fill the space'. In two dimensions, this is the case if one column is a multiple of the other column. In general, this will be the case if $\binom{A_{11}}{A_{21}} = \alpha\binom{A_{12}}{A_{22}}$, for some number α, i.e. if $A_{11} = \alpha A_{12}$ and $A_{21} = \alpha A_{22}$. Getting rid of α, we get the condition that $A_{11} = \frac{A_{21}}{A_{22}}A_{12}$,or $A_{11}A_{22} = A_{21}A_{22}$. If this condition is satisfied, $A_{11}A_{22} - A_{21}A_{22} = \det(A) = 0$.

In three dimensions, a matrix is not invertible if its columns do not 'fill the space', in the sense that either all three columns lie on a line, or in a 2-dimensional plane. We can formalize this concept using the notion of linear independence:

2.3.4 Linear independence

Definition:

A set of vectors $v_1 \dots v_m$ is said to be linearly independent if we can not find number $\alpha_1, \alpha_2, \dots, \alpha_m$, where at least one of the α's is not equal to 0, such that

$$\alpha_1 v_1 + \alpha_2 v_2 + \dots \alpha_m v_m = 0.$$

If we can find such numbers, then the set of vector is said to be linearly dependent.

In other words, a set of vectors is linearly independent if we can not find a (non-trivial) linear combination of them that is 0. If vectors $v_1 \dots v_m$ are linearly dependent, then we can find some number such that $\alpha_1 v_1 + \alpha_2 v_2 + \dots \alpha_m v_m = 0$, where, lets say, $\alpha_1 \neq 0$. So, in this case, we can write

$$v_1 = \frac{-1}{\alpha_1}(\alpha_2 v_2 + \dots + \alpha_m v_m),$$

i.e. we can express (at least) one of the vectors as a linear combination of the others.

Examples

- Any single, non-zero vector v is linearly independent: Clearly, $\alpha v = 0$ is only 0 if $\alpha = 0$.
- The canonical basis vectors are independent: If we write, e.g.

$$\alpha_1 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} + \alpha_2 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} + \alpha_3 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix},$$

then the only way this could be equal to $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$ if $\alpha_1 = \alpha_2 = \alpha_3 = 0$.

- The two vectors $\binom{2}{4}$ and $\binom{1}{2}$ are linearly dependent, as $\binom{2}{4} = 2\binom{1}{2}$.
- The vectors $\begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 2 \\ 0 \\ 3 \end{pmatrix}, \begin{pmatrix} 0 \\ -2 \\ 3 \end{pmatrix}$ are linearly dependent, as
$$2\begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 2 \\ 0 \\ 3 \end{pmatrix} + \begin{pmatrix} 0 \\ -2 \\ 3 \end{pmatrix} = 0$$
- Any set of vectors which includes a zero vector is linearly dependent, e.g. $1 \cdot \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + 0v_2 + 0v_{,m} = 0$.

2.3.5 Rank of a matrix

Consider an $n \times n$ matrix A with columns $A_1 \dots A_n$. We showed early that the image of any vector x can be written as

$$Ax = x_1A_1 + \dots x_nA_n.$$

A matrix is not invertible if and only if there exists a vector x which is non-zero, but which maps to 0,i.e. for which

$$0 = x_1A_1 + \dots x_nA_n.$$

In other words, a matrix is invertible if and only if its columns are linearly independent. The maximal number of linearly independent columns of a matrix is called the *rank*. A matrix is said to have *full rank* if its rank is n, i.e. if all of its columns are linearly independent, otherwise it is said to be *rank-deficient.*

Examples

- Any non-zero vector v can be considered as a rank 1 matrix.
- The matrix $\begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix}$ has rank 1, as its columns are not independent (so rank$<$2), but the vector $\binom{1}{2}$ is non-zero, so rank$\geq$ 1.
- The matrix $\begin{pmatrix} -1 & 2 & 0 \\ 1 & 0 & -2 \\ 0 & 3 & 3 \end{pmatrix}$ has rank 2, as its columns are not independent (as shown above), so rank$<$3, but e.g. the two vectors $\begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 2 \\ 0 \\ 3 \end{pmatrix}$ are linearly independent, so rank$\geq$ 2.

- For any diagonal matrix, the rank is equal to the number of non-zero diagonal entries.
- The matrix
$$\begin{pmatrix} 1 & 3 & 5 & 7 & 10 \\ 2 & 4 & 6 & 8 & 9 \end{pmatrix}$$
has rank 2, as both rows are linearly independent (so rank ≥ 2), but there are only two rows (so rank ≤ 2).
- The outer product of two (non zero) vectors, $xx^\top$, has rank 1: For example,
$$\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} (1,2,3) = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ 3 & 6 & 9 \end{pmatrix}$$
has rank one, as (by construction), each row is a multiple of the others.
- Similarly, a sum of outer products of m n-dimensional, linearly independent vectors $x_1 \dots x_m$has rank m if $m \leq n$,and rank n otherwise: $C = \sum_i x_i x_i^\top$. As covariance matrices are of this form, this means that covariance matrices are rank-deficient if one has not collected enough data.

Note

- The definition of 'invertible' in terms of linearly independent columns often makes it easy to determine whether a matrix is invertible by inspecting its columns:
  - If any column is zero, the matrix is not invertible
  - If two columns are the same, or one column is a multiple of the other, it is not invertible.
- In fact, the rank of any matrix A is the same as the rank of its transpose, $A^\top$. As a consequence, we can replace the 'columns' in the definition of rank by 'rows'. For determining the rank of a matrix, it is sometimes easier to look at the columns, and sometimes easier to look at the rows.

$\Diamond$

In the following section we will look deeper into the mechanics of matrices and their properties. Especially, we will look at eigenvalues and eigenvectors, an important concept of matrices which can be used for understanding and analysing a big portion of all matrix equations. For example, it plays an important role in *Principal Component Analysis*, known as *PCA*, which is one of the most common algorithms for analysis data. We will also look at this algorithm in more detail.

2.4 Change of Basis

In the previous sections, we have heard that a matrix to a linear function is always with respect to some basis and that the matrix changes, if the basis does. Fortunately, changing a vector of coordinates or a matrix from one basis to another is a linear function. Therefore, we can find a matrix to this function. This is what we will look at in more detail in this section.

2.4.1 Coordinate Change of a Vector under a Change of Basis

Let us develop the coordinate change along a simple example. Previously, we have seen that

$$\mathbf{e}_1 = \begin{pmatrix}1\\0\end{pmatrix}, \mathbf{e}_2 = \begin{pmatrix}0\\1\end{pmatrix} \quad \text{and} \quad \mathbf{r}_1 = \begin{pmatrix}\cos\phi\\\sin\phi\end{pmatrix}, \mathbf{r}_2 = \begin{pmatrix}-\sin\phi\\\cos\phi\end{pmatrix}$$

form orthonormal bases of $\mathbb{R}^2$. If we now look at the vector $\mathbf{x} = \binom{\cos\phi}{\sin\phi}$, we can see that its coordinates under the first basis are $\cos\phi$ and $\sin\phi$, since

$$\begin{aligned}\mathbf{x} &= \cos\phi \cdot \begin{pmatrix}1\\0\end{pmatrix} + \sin\phi \cdot \begin{pmatrix}0\\1\end{pmatrix}\\ &= \cos\phi \cdot \mathbf{e}_1 + \sin\phi \cdot \mathbf{e}_2.\end{aligned}$$

Under the second basis the coordinates are 1 and 0, since

$$\begin{aligned}\mathbf{x} &= 1 \cdot \begin{pmatrix}\cos\phi\\\sin\phi\end{pmatrix} + 0 \cdot \begin{pmatrix}-\sin\phi\\\cos\phi\end{pmatrix}\\ &= 1 \cdot \mathbf{r}_1 + 0 \cdot \mathbf{r}_2.\end{aligned}$$

Hence, if we are looking for a matrix, that transforms the coordinates with respect to the basis $\mathbf{e}_1, \mathbf{e}_2$ into coordinates of the same vector with respect to basis $\mathbf{r}_1, \mathbf{r}_2$, this matrix should transform the coordinates $\binom{\cos\phi}{\sin\phi}$ into the coordinates $\binom{1}{0}$.

In order to find this matrix, we have to remember that a matrix of a linear mapping must contain the results of the linear function on the basis vectors in its columns. Therefore, what we are looking for is a matrix, that contains the basis vectors of the old basis $\mathbf{e}_1, \mathbf{e}_2$ written in terms of the new basis $\mathbf{r}_1, \mathbf{r}_2$. If we want to change from $\mathbf{e}_1, \mathbf{e}_2$ to $\mathbf{r}_1, \mathbf{r}_2$, we must express $\mathbf{e}_1, \mathbf{e}_2$ in terms of $\mathbf{r}_1$ and $\mathbf{r}_2$. Since both bases are orthonormal bases, we can use the dot product trick, i.e.

$$\begin{aligned}\mathbf{e}_1 &= \langle \mathbf{e}_1, \mathbf{r}_1\rangle \mathbf{r}_1 + \langle \mathbf{e}_1, \mathbf{r}_2\rangle \mathbf{r}_2\\ &= \cos\phi \cdot \mathbf{r}_1 - \sin\phi \mathbf{r}_2\end{aligned}$$

and

$$\begin{aligned}\mathbf{e}_2 &= \langle \mathbf{e}_2, \mathbf{r}_1\rangle \mathbf{r}_1 + \langle \mathbf{e}_2, \mathbf{r}_2\rangle \mathbf{r}_2\\ &= \sin\phi \cdot \mathbf{r}_1 + \cos\phi \mathbf{r}_2.\end{aligned}$$

Therefore, the coordinates of $\mathbf{e}_1$ and $\mathbf{e}_2$ with respect to the basis $\mathbf{r}_1, \mathbf{r}_2$ are $(\cos\phi, -\sin\phi)^\top$ and $(\sin\phi, \cos\phi)^\top$. If we write the new coordinates in the column of a matrix, we get the matrix $\mathbf{B}$ that changes the coordinates from basis $\mathbf{e}_1, \mathbf{e}_2$ to $\mathbf{r}_1, \mathbf{r}_2$:

$$\mathbf{B} = \begin{pmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{pmatrix}.$$

Note, that $\mathbf{B}$ is just the transpose of the matrix $\mathbf{R} = (\mathbf{r}_1, \mathbf{r}_2)$, which contains the new basis vectors $\mathbf{r}_1, \mathbf{r}_2$ as column vectors, i.e. $\mathbf{B} = \mathbf{R}^\top$. This is no surprise, since

$$\mathbf{R}^\top \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \langle \mathbf{e}_1, \mathbf{r}_1 \rangle \\ \langle \mathbf{e}_1, \mathbf{r}_2 \rangle \end{pmatrix}$$

and

$$\mathbf{R}^\top \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \langle \mathbf{e}_2, \mathbf{r}_1 \rangle \\ \langle \mathbf{e}_2, \mathbf{r}_2 \rangle \end{pmatrix},$$

i.e. the multiplication of the coordinates of $\mathbf{e}_1$ and $\mathbf{e}_2$ with $\mathbf{R}^\top$ contains the projections of $\mathbf{e}_1$ and $\mathbf{e}_2$ onto $\mathbf{r}_1$ and $\mathbf{r}_2$. From this observation, we can read of another important theorem.

If we want to reverse the basis change, and change the coordinates of $\mathbf{x}$ with respect to $\mathbf{r}_1, \mathbf{r}_2$ into coordinates with respect to $\mathbf{e}_1, \mathbf{e}_2$, we need a matrix that contains the coordinates of $\mathbf{r}_1$ and $\mathbf{r}_2$ with respect to $\mathbf{e}_1, \mathbf{e}_2$ in its columns. But this is just $\mathbf{R}$. Let us now state this finding in a general theorem.

Theorem If $\mathbf{v}_1, ..., \mathbf{v}_n$ and $\mathbf{w}_1, ..., \mathbf{w}_n$ are two orthonormal bases, then the matrix that changes the coordinates of a vector with respect to the first basis into coordinates with respect to the second basis, can be found by the following steps:

1. Express the basis $\mathbf{w}_1, ..., \mathbf{w}_n$ in terms of the basis $\mathbf{v}_1, ..., \mathbf{v}_n$ and write the coordinates into the column of a matrix $\mathbf{R}$.
2. The matrix, that changes the coordinates from $\mathbf{v}_1, ..., \mathbf{v}_n$ to $\mathbf{w}_1, ..., \mathbf{w}_n$ is then given by $\mathbf{R}^\top$.
3. The matrix that changes the coordinates from $\mathbf{w}_1, ..., \mathbf{w}_n$ to $\mathbf{v}_1, ..., \mathbf{v}_n$ is given by $\mathbf{R}$.

$\Diamond$

Additionally to this theorem, we can find another useful theorem, that tells us that the inverse of so called *orthonormal matrix*, are easy to find.

Definition (Orthonormal Matrix) A matrix, which has orthonormal columns or rows, is called *orthonormal matrix*.

$\Diamond$

Theorem The inverse of a orthonormal matrix $\mathbf{B}$ is given by its transpose, i.e. $\mathbf{B}^{-1} = \mathbf{B}^\top$.

Proof:

The ijth entry of the matrix $\mathbf{B}^\top\mathbf{B}$ contains the dot product between the column vectors $\mathbf{b}_i$ and $\mathbf{b}_j$, which are orthonormal by definition, i.e. $\langle \mathbf{b}_i, \mathbf{b}_j \rangle = \delta_{ij}$. Therefore, the matrix only contains ones on the diagonal, i.e. where $i = j$. On the other hand, consider the identity matrix $\mathbf{I}$. This matrix is an orthonormal matrix, that contains the canonical basis in its columns. Therefore, the matrix $\mathbf{B}\mathbf{B}^\top\mathbf{I}$ can be seen as changing from the orthonormal basis into the basis given by the columns of $\mathbf{B}$ and back. Since nothing happens, when changing bases back and forth, we have that $\mathbf{B}\mathbf{B}^\top\mathbf{I} = \mathbf{I}$ and, therefore $\mathbf{B}\mathbf{B}^\top = \mathbf{B}^\top\mathbf{B} = \mathbf{I}$, which shows that $\mathbf{B}^\top$ is the inverse of $\mathbf{B}$ and vice versa.

□

2.4.2 Changing the Basis of a Matrix

Now that we have the tool to change the coordinates of a vector under the change of coordinates, it is easy to adjust a matrix appropriately under a change of the basis. Consider the two basis $\mathbf{e}_1, \mathbf{e}_2$ and $\mathbf{r}_1, \mathbf{r}_2$ of $\mathbb{R}^2$ again. Assume, that we are given a matrix $\mathbf{A}_E$ with respect to the first basis, but the coordinates $(y_1, y_2)^\top$ of a vector $\mathbf{y} = y_1 \cdot \mathbf{r}_1 + y_2\mathbf{r}_2$ with respect to the second. In order to be able to use the matrix $\mathbf{A}_E$ for $(y_1, y_2)^\top$, we need to transform it appropriately. But since we know how to transform coordinates, it is easy to obtain the new matrix $\mathbf{A}_R$ with respect to $\mathbf{r}_1, \mathbf{r}_2$. The idea is to change the coordinates $(y_1, y_2)^\top$ into the basis $\mathbf{e}_1, \mathbf{e}_2$, apply $\mathbf{A}_E$ and change the coordinates back. Each operation can be done via a matrix multiplication. Therefore, we obtain the new matrix $\mathbf{A}_R$ by multiplying the single matrices.

Theorem

1. Let $\mathbf{v}_1, ..., \mathbf{v}_n$ and $\mathbf{w}_1, ..., \mathbf{w}_n$ be two orthonormal bases of $\mathbb{R}^n$, and let $\mathbf{A}_V \in \mathbb{R}^{n\times n}$ be a square matrix with respect to the basis $\mathbf{v}_1, ..., \mathbf{v}_n$. We obtain the equivalent matrix $\mathbf{A}_W$ with respect to the basis $\mathbf{w}_1, ..., \mathbf{w}_n$ by the following operation:

$$\mathbf{A}_W = \underbrace{\mathbf{V}}_{change\ V\to W} \underbrace{\mathbf{A}_V}_{apply\ \mathbf{A}_V} \underbrace{\mathbf{V}^\top}_{change\ W\to V},$$

   where $\mathbf{V}$ is the matrix that contains the coordinate vectors of $\mathbf{v}_1, ..., \mathbf{v}_n$ with respect to the basis $\mathbf{w}_1, ..., \mathbf{w}_n$, i.e. $\mathbf{V}^\top$ changes the coordinates from $\mathbf{w}_1, ..., \mathbf{w}_n$ to $\mathbf{v}_1, ..., \mathbf{v}_n$.

2. If the matrix $\mathbf{A}$ is not square, i.e. $\mathbf{A} \in \mathbb{R}^{m\times n}$, we have two different bases: A basis $\mathbf{v}_1, ..., \mathbf{v}_n$ for the inputs and a basis $\mathbf{a}_1, ..., \mathbf{a}_m$ for the outputs, since the linear function $f_\mathbf{A}$ of $\mathbf{A}$ maps from $\mathbb{R}^n$ to $\mathbb{R}^m$. If we want to change the matrix $\mathbf{A}$ such that is does the equivalent operation, but with respect

to two different basis $\mathbf{w}_1, ..., \mathbf{w}_n$ and $\mathbf{b}_1, ..., \mathbf{b}_m$, we must multiply a matrix $\mathbf{M}_A^B$ from the left, that changes the coordinates from basis $\mathbf{a}_1, ..., \mathbf{a}_m$ to $\mathbf{b}_1, ..., \mathbf{b}_m$, and a matrix $\mathbf{M}_V^W$ from the right, that changes the coordinates from $\mathbf{v}_1, ..., \mathbf{v}_n$ to $\mathbf{w}_1, ..., \mathbf{w}_n$.

$\Diamond$

Let us look at a few examples.

Examples

1. Consider the bases

$$\mathbf{e}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \mathbf{e}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \text{and} \quad \mathbf{r}_1 = \begin{pmatrix} \cos\phi \\ \sin\phi \end{pmatrix}, \mathbf{r}_2 = \begin{pmatrix} \sin\phi \\ \cos\phi \end{pmatrix}$$

of $\mathbb{R}^2$again. If we apply the theorem above in order to change the identity matrix $\mathbf{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ from the canonical basis to the basis $\mathbf{r}_1, \mathbf{r}_2$, we obtain with $\mathbf{R} = (\mathbf{r}_1, \mathbf{r}_2)$:

$$\begin{aligned} \mathbf{I}_R &= \mathbf{R}\mathbf{I}\mathbf{R}^\top \\ &= \mathbf{R}\mathbf{I}\mathbf{R}^\top \\ &= \mathbf{I}, \end{aligned}$$

since $\mathbf{R}$ is the inverse of $\mathbf{R}^\top$and vice versa. This shows that the identity matrix is always given by $\mathbf{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ no matter which orthonormal basis we choose.

2. Consider the bases $\mathbf{e}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \mathbf{e}_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \mathbf{e}_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ of $\mathbb{R}^3$ and $\mathbf{e}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \mathbf{e}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ and $\mathbf{r}_1 = \begin{pmatrix} \cos\phi \\ \sin\phi \end{pmatrix}$, $\mathbf{r}_2 = \begin{pmatrix} \sin\phi \\ \cos\phi \end{pmatrix}$ of $\mathbb{R}^2$. If we want to adapt the projection matrix $\mathbf{P} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$ such that it takes coordinates with respect to $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ and yields coordinates with respect to $\mathbf{r}_1, \mathbf{r}_2$, we need to do the following multiplication:

$$\begin{aligned} \mathbf{P}_E^R &= \mathbf{R}^\top \mathbf{P} \\ &= \begin{pmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \end{pmatrix}, \end{aligned}$$

which transforms the outcome of $\mathbf{P}$ into coordinates with respect to $\mathbf{r}_1, \mathbf{r}_2$.

2.5 Eigenvalues and Eigenvectors

In this section we will look at a very important concept of matrices: *eigenvalues* and *eigenvectors*. Eigenvalues and eigenvectors are mathematical objects that can be derived from square matrices and which are tightly linked to the intrinsic mechanic of a matrix. Eigenvalues play an important role in data analysis, since they are a key ingredient of the *Principal Component Analysis (PCA)* algorithm. We will first introduce eigenvalues and eigenvectors with an example, then make a few general comments about eigenvalues and eigenvectors and their properties, and then introduce PCA.

2.5.0.1 Eigenvalues and Eigenvectors

In one of the first sections we saw that a one-dimensional linear function can be expressed by a single number: We compute the result of a linear function $f(x)$ on some value, say $x_0 = 1$, obtain $\lambda = f(x_0) = f(1)$ and compute any other value by $f(x) = f(a \cdot x_0) = af(x_0) = \lambda \cdot a$. Here, x_0 serves as our basis in $\mathbb{R}^1$. We express x in terms of x_0, get the coordinate a and use the linearity so get it into the form $f(x) = \lambda \cdot x$.

Now we could ask the question, whether there are direction in space, such that a multi-dimensional linear function can also be described by a single number for every input along that direction. Those would be directions, that are tightly linked to the mechanic of that function, since it takes this especially easy form in that direction.

Let us have a look at, how those directions could look like.

Example Consider the matrix

$$\mathbf{A} = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}.$$

This matrix doubles the x_1-coordinate and leaves the x_2-coordinate untouched, i.e.

$$\mathbf{A}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2x_1 \\ x_2 \end{pmatrix}.$$

What are the directions in $\mathbb{R}^2$ that are tightly linked with the transformation carried out by $\mathbf{A}$? Considering the transformation of a vector $\mathbf{x} = \binom{x_1}{x_2}$, we can see that these two directions are the first coordinate direction (i.e. the direction of the first basis vector) and the second coordinate direction, since $\mathbf{A}$ doubles the length of a vector along the first and leaves a vector untouched along the second direction. In other words, if a vector is given by $\mathbf{y} = \binom{a}{0}$, then the result of $\mathbf{Ay} = \binom{2a}{0} = 2\mathbf{y}$ is just an elongated version of the input. The same is true for an input $\mathbf{z} = \binom{0}{b}$, since $\mathbf{Az} = \binom{0}{b} = \mathbf{z}$ does not change the input vector $\mathbf{z}$.

◁

If we generalize the example, we see that the directions from the example before are exactly the ones that we were looking for: Given the direction and an input vector $\mathbf{x}$ that has the same orientation, the linear mapping can be described by one number, i.e. the scaling vector along that direction. This means that the directions we are searching for, that are tightly linked to the transformation carried out by a matrix $\mathbf{A}$, are those directions in which a vector is only scaled, but not rotated, i.e. the direction that does only affect the length, but not the orientation of a vector when multiplied with $\mathbf{A}$. This is the general definition of a eigenvector. The amount of scaling in that direction is called eigenvalue of $\mathbf{A}$.

Definition Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be a square matrix. Any vector $\mathbf{v} \neq \mathbf{0}$ that fulfills $\mathbf{Av} = \lambda\mathbf{v}$ is called *eigenvector* of $\mathbf{A}$. The value λ is called *eigenvalue* of $\mathbf{A}$. Each eigenvector has its eigenvalue. However, the eigenvalues for different eigenvectors might be the same.

Note, that if $\mathbf{v}$ is an eigenvector, then $a\mathbf{v}$ with $\mathbf{v} \in \mathbb{R}$ is an eigenvector, too. Therefore, we can assume without loss of generality that eigenvectors have length one, i.e. $||\mathbf{v}|| = 1$.

$\Diamond$

How do we compute eigenvectors? In order to answer this question, let us rewrite the definition of an eigenvector a little bit:

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

This shows, that we are interested in a vector $\mathbf{v} \in \mathbb{R}^n$ and a scalar $\lambda \in \mathbb{R}$ such that the linear function given by the matrix $(\mathbf{A} - \lambda\mathbf{I})$ maps $\mathbf{v}$ onto the zero-vector. A trivial solution for this would be, to set $\mathbf{v} = \mathbf{0}$, but this is not allowed by the definition of an eigenvector. If $\mathbf{v} \neq \mathbf{0}$, we must adjust λ and $\mathbf{v}$ such that $\mathbf{v}$ lives in the *nullspace* of $\mathbf{A}$.

Definition (Nullspace) The nullspace of a matrix $\mathbf{A}$ is the set of all vectors $\mathbf{v}$ that are mapped onto the zero vector, i.e.

$$\mathcal{N}(\mathbf{A}) = \{\mathbf{v} \in \mathbb{R}^n | \, \mathbf{Av} = \mathbf{0}\}.$$

$\Diamond$

Example Consider the matrix

$$\mathbf{P} = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

No matter what vector we feed into $\mathbf{P}$, the x_3-coordinate always gets mapped into 0. This means that all vectors along the direction $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ are mapped onto the zero vector. Therefore, the nullspace of $\mathbf{P}$ is given by

$$\mathcal{N}(\mathbf{P}) \quad = \quad \{a \cdot (0,0,1)^\top \,|\, a \in \mathbb{R}\}.$$

$\lhd$

If the nullspace of a matrix contains any other element than zero, the matrix is not invertible anymore. This is simply because the zero vector is always mapped onto the zero vector by linear mappings. If another vector is mapped into the zero vector, we would not know which vector we should assign to the zero vector when inverting the linear mapping. Therefore, it cannot be invertible.

This brings us back to the question of how to compute the eigenvector and the eigenvalues. We know now that $(\mathbf{A} - \lambda\mathbf{I})$ must not be invertible. We already saw that determinants can be used to check whether a matrix is invertible or not. We can use that here. If a matrix is not invertible, then the determinant must be zero. Therefore, we are searching for all $\lambda \in \mathbb{R}$ such that

$$\det(\mathbf{A} - \lambda\mathbf{I}) \quad = \quad 0.$$

Once we have the solutions to that equation, we can search for the eigenvectors belonging to each solution. But let us look at a few examples first:

Examples

1. Let us start by the example from above

   $$\mathbf{A} \quad = \quad \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}.$$

   The eigenvalues of $\mathbf{A}$ are given by the solution of

   $$\det\left(\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right) \quad = \quad \det\left(\begin{pmatrix} 2-\lambda & 0 \\ 0 & 1-\lambda \end{pmatrix}\right)$$
   $$0.$$

   Determinants of diagonal matrices are easy to compute:

   $$\det\left(\begin{pmatrix} 2-\lambda & 0 \\ 0 & 1-\lambda \end{pmatrix}\right) \quad = \quad (2-\lambda)\cdot(1-\lambda).$$

   We see that the determinant gives us a polynomial in λ. Since it is already factorized we can read off the solutions as $\lambda_1 = 2$ and $\lambda_2 = 1$. Therefore the eigenvalues of $\mathbf{A}$ are given by $\lambda_1 = 2$ and $\lambda_2 = 1$.

2. Consider

$$\mathbf{A} = \begin{pmatrix} 6 & 4 \\ 2 & 4 \end{pmatrix}.$$

Let us do the same steps as in the example before:

$$\begin{aligned} \det\left(\begin{pmatrix} 6 & 4 \\ 2 & 4 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right) &= \det\begin{pmatrix} 6-\lambda & 4 \\ 2 & 4-\lambda \end{pmatrix} \\ &= (6-\lambda)(4-\lambda) - 2 \cdot 4 \\ &= 24 - 4\lambda - 6\lambda + \lambda^2 - 8 \\ &= \lambda^2 - 10\lambda + 16 \\ &= 0. \end{aligned}$$

The solution can be found by solving this quadratic equation:

$$\begin{aligned} \lambda_{1,2} &= \frac{10 \pm \sqrt{100-64}}{2} \\ &= \frac{10 \pm 6}{2} \\ &= \frac{10 \pm 6}{2} \\ &= 5 \pm 3. \end{aligned}$$

3. Consider the triangular matrix

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 2 & 2 \\ 0 & 0 & 3 \end{pmatrix}.$$

Since we know that the determinant of a triangular matrix is again just the product of the diagonal terms

$$\det(\mathbf{A} - \lambda\mathbf{I}) = (1-\lambda)(2-\lambda)(3-\lambda).$$

If we set the determinant to zero, the cubic equation has the solutions $\lambda_1 = 1$, $\lambda_2 = 2$ and $\lambda_3 = 3$.

◁

After the computation of the eigenvalues, we need to find the eigenvectors. However, this is simple task. We can just use the definition of an eigenvector

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

Since we know λ now, the left hand side is just a matrix-vector multiplication, or seen differently, a linear equation system. In order to get the eigenvector, we must solve that for $\mathbf{v}$. Let us look at our examples again.

Examples

1. We already know that the eigenvalues of

$$\mathbf{A} = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}$$

are given by $\lambda_1 = 2$ and $\lambda_2 = 1$. In order to get the eigenvector for λ_1, we must solve the following equation system

$$\begin{aligned} \left(\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix} - \lambda_1 \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right) &= \left(\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix} - \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix} \right) \\ &= \begin{pmatrix} 0 & 0 \\ 0 & -1 \end{pmatrix} \mathbf{v} \\ &= \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \end{aligned}$$

Written down as an equation system

$$\begin{aligned} 0 \cdot v_1 + 0 \cdot v_2 &= 0 \\ 0 \cdot v_1 - v_2 &= 0. \end{aligned}$$

Every vector $\mathbf{v} = \binom{a}{0}$ for arbitrary a is a solution to this equation. Since we normalize eigenvectors for convenience, the solution is given by $\mathbf{v}_1 = \binom{1}{0}$. Therefore the eigenvector to the eigenvalue $\lambda_1 = 2$ is $\mathbf{v}_1 = \binom{1}{0}$. An analogous computation yields $\mathbf{v}_2 = \binom{0}{1}$.

2. Before, we saw that the eigenvalues of

$$\mathbf{A} = \begin{pmatrix} 6 & 4 \\ 2 & 4 \end{pmatrix}$$

are given by

$$\lambda_{1,2} = 5 \pm 3.$$

Now, we do the same steps as in the example before. The linear equation system for the first eigenvalue $\lambda_1 = 8$ is given by

$$\begin{aligned} 6v_1 + 4v_2 &= 8v_1 \\ 2v_1 + 4v_2 &= 8v_2, \end{aligned}$$

which is equivalent to

$$\begin{aligned} -2v_1 + 4v_2 &= 0 \\ 2v_1 - 4v_2 &= 0. \end{aligned}$$

Both equations are the same. Solving the first for v_1 yields

$$v_1 = 2v_2.$$

Therefore, the normalized eigenvector to the eigenvalue $\lambda_1 = 8$ is given by $\mathbf{v}_1 = \frac{1}{\sqrt{3}}\binom{2}{1}$. An analogous computation yields the eigenvector of $\lambda_2 = 2$.

◁

2.5.0.2 Eigen Decomposition

Eigenvalues and eigenvectors are important tools to understand the function behind a matrix $\mathbf{A}$. Once we know along which directions $\mathbf{v}_i$ only scales the input, we get insight in what omit$\mathbf{A}$ is actually doing. In order to make this more apparent, a matrix can be decomposed into a product of matrices, that are built from eigenvectors and eigenvalues.

Theorem (Eigen Decomposition) Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be an arbitrary square matrix and the eigenvalues of $\mathbf{A}$ be mutually distinct, i.e. $\lambda_i \neq \lambda_j$ for $i \neq j$. Then there exists a diagonal matrix

$$\mathbf{D} = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & & \\ & & \ddots & \\ 0 & & & 0 \\ & \dots & 0 & \lambda_n \end{pmatrix}$$

that hold the eigenvalues in the diagonal and a matrix $\mathbf{U} = (\mathbf{u}_1, ..., \mathbf{u}_n)$ that contains the eigenvectors to the eigenvalues $\lambda_1, ..., \lambda_n$ as column vectors, such that

$$\mathbf{AU} = \mathbf{UD}.$$

Even if the eigenvalues are not distinct, i.e. $\lambda_{k_1} = ... = \lambda_{k_r}$ for $k_i \neq k_j$, then there still exists such a decomposition if the eigenvectors corresponding to those eigenvalues are linearly independent.

$\Diamond$

Theorem (Eigen Basis) Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be an arbitrary square matrix. Eigenvectors $\mathbf{v}_i, \mathbf{v}_j$ to distinct eigenvalues $\lambda_i \neq \lambda_j$ are linearly independent. This implies that the eigenvectors to n mutually distinct eigenvalues form a basis of $\mathbb{R}^n$.

$\Diamond$

These theorems sound very abstract. However, we can get an intuition for them with the following considerations. Let us first look at the matrix $\mathbf{U}$. If we assume, that the eigenvalues of $\mathbf{A}$ are mutually distinct, then the columns of $\mathbf{U}$ for a - not necessarily orthogonal- basis of $\mathbb{R}^n$. This means that multiplying a vector $\mathbf{x}$ with $\mathbf{U}$ as we would treat the entries of $\mathbf{x}$ as coordinates with respect to the basis $\mathbf{U}$ and change its coordinates to the basis of $\mathbf{A}$ by multiplying $\mathbf{x}$ with $\mathbf{U}$. This holds true, even if the eigenvectors do not form an orthonormal basis. Interpreted in that way the term $\mathbf{AUx}$ means, that we first change the coordinates in $\mathbf{x}$ to the coordinates of $\mathbf{A}$, by $\mathbf{Ux}$ and the apply the transformation carried out by $\mathbf{A}$ via $\mathbf{AUx}$.

The theorem tells us, that this is the same as first scaling the coordinates in $\mathbf{x}$ by multiplying it with $\mathbf{D}$ and then changing the basis to that of $\mathbf{A}$. The result of

$$\mathbf{Dx} = \begin{pmatrix} \lambda_1 x_1 \\ \lambda_2 x_2 \\ \vdots \\ \lambda_n x_n \end{pmatrix}$$

is just $\mathbf{x}$, but with every entry i scaled by the eigenvalue λ_i. This is exactly the way how the eigenvalues and eigenvectors are linked to the transformation of $\mathbf{A}$. If we express a vector with respect to the eigenvectors, then the transformation is only a rescaling of each coordinate.

Let us look at a few examples and then see how we can use this result.

Examples

1. The identity matrix $\mathbf{I} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & & \\ & & \ddots & 0 \\ 0 & & 0 & 1 \end{pmatrix}$ is already diagonal. Therefore, its eigenvectors are $\lambda_1 = ... = \lambda_n = 1$ and the matrix $\mathbf{U}$ is the matrix containing the canonical basis, i.e. $\mathbf{U} = (\mathbf{e}_1, ..., \mathbf{e}_n) = \mathbf{I}$.

2. Consider the matrix

$$\begin{aligned} & \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{pmatrix} \\ = & \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} 2\cos\phi & 2\sin\phi \\ -\sin\phi & \cos\phi \end{pmatrix} \\ = & \begin{pmatrix} 2\cos^2\phi + \sin^2\phi & 2\cos\phi\sin\phi - \cos\phi\sin\phi \\ 2\cos\phi\sin\phi - \cos\phi\sin\phi & 2\sin^2\phi + \cos^2\phi \end{pmatrix}. \end{aligned}$$

   This matrix changes the basis from $\mathbf{e}_1, \mathbf{e}_2$ to $\mathbf{r}_1, \mathbf{r}_2$ (see examples before), doubles the first coordinate and changes the coordinates of the results back to $\mathbf{e}_1, \mathbf{e}_2$. Therefore, the eigenvalues are $\lambda_1 = 2$ and $\lambda_2 = 1$ and the corresponding eigenvectors are $\mathbf{r}_1, \mathbf{r}_2$.

3. Consider the matrix $\begin{pmatrix} 3 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{pmatrix}$. This matrix multiplies the first coordinate by three, doubles the second coordinate and sets the third coordinate to zero. Since the matrix is already diagonal, the eigenvectors are simply the canonical basis vectors. The eigenvalues are given by $\lambda_1 = 3$, $\lambda_2 = 2$ and $\lambda_3 = 0$.

◁

From the third example we can see another important use of eigenvalues. The matrix in example does not have full rank, since the third column is the zero vector and therefore linearly dependent on both others. At the same time, the eigenvalue corresponding to the third column is zero as well. We can look at that as if the zero eigenvalue switches off the corresponding eigenvector. In general, if we compute the eigendecomposition of a matrix and one or more of its eigenvalues is zero, then the matrix does not have full rank, since all information about that direction in space gets lost, when multiplying it with the diagonal matrix, that contains the eigenvalues. There is a slightly more formal way of stating this result.

Assume that we decompose $\mathbf{A}$ such that $\mathbf{AU} = \mathbf{UD}$ or, equivalently, $\mathbf{A} = \mathbf{UDU}^{-1}$. If the eigenvalue of $\mathbf{A}$ are mutually distinct, we can invert $\mathbf{U}$, since its column vectors, the eigenvectors of $\mathbf{A}$, are linearly independent. If $\mathbf{A}$ had full rank, then its determinant must not be zero. But if one of the eigenvalues equals zero, the determinant $\det \mathbf{D} = \prod_{i=1}^{n} \lambda_i$ of the diagonal matrix will be zero and so will be the determinant of $\mathbf{A}$, since

$$\begin{aligned} \det \mathbf{A} &= \det \mathbf{UDU}^{-1} \\ &= \det \mathbf{U} \cdot \det \mathbf{D} \cdot \det \mathbf{U}^{-1}. \end{aligned}$$

Therefore, as soon as one eigenvalue of $\mathbf{A}$ is zero, $\mathbf{A}$ does not have full rank and is not invertible.

However, given the eigenvalues and eigenvectors of a matrix, we can do even more than just saying if $\mathbf{A}$ is invertible or not. If it is, i.e. if $\mathbf{A}$ has mutually distinct eigenvalues $\lambda_1 \neq ... \neq \lambda_n \neq 0$, then we can immediately compute the inverse of $\mathbf{A}$. It is given by $\mathbf{A}^{-1} = \mathbf{UD}^{-1}\mathbf{U}^{-1}$, where

$$\mathbf{D}^{-1} = \begin{pmatrix} \frac{1}{\lambda_1} & 0 & \dots & 0 \\ 0 & \frac{1}{\lambda_2} & & \\ & & \ddots & \\ 0 & & & 0 \\ & \dots & 0 & \frac{1}{\lambda_n} \end{pmatrix}.$$

It is easy to see that this is indeed the inverse of $\mathbf{A}$, since

$$\begin{aligned} \mathbf{AA}^{-1} &= \mathbf{UDU}^{-1}\mathbf{UD}^{-1}\mathbf{U}^{-1} \\ &= \mathbf{UDD}^{-1}\mathbf{U}^{-1} \\ &= \mathbf{UU}^{-1} \\ &= \mathbf{I}. \end{aligned}$$

The same holds true for $\mathbf{A}^{-1}\mathbf{A}$.

2.5.0.3 Eigenvalues and Eigenvectors of Symmetric Matrices

You might have noticed, that all matrices in the examples above, were symmetric. This has a special reason: For a general eigenvalues decomposition, there are two basic problems. First, there is no reason why the eigenvectors should

be orthogonal to each other in the general case. This is unfortunate, since it would be nice to express vectors in terms of the eigenvectors of a matrix, since the operation of a matrix would become very simple in that case. However, we have seen that non-orthogonal bases are difficult do deal with, so for general matrices, changing into the eigenbasis is not an option.

The second problem is, that the eigenvalues might not be in $\mathbb{R}$. For example, if consider any rotation matrix $\mathbf{R}$, it is difficult to find a vector $\mathbf{v}$ and a scalar λ such that $\mathbf{Rv} = \lambda\mathbf{v}$. However, if $\lambda \in \mathbb{C}$, the set of all complex numbers, then this is possible. However, for us it is more interesting to have eigenvalues in $\mathbb{R}$.

Fortunately there is a class of matrices, that gives us both: Orthogonality of the eigenvectors and real eigenvalues.

Theorem (Eigen Decomposition of Symmetric Matrices) Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be an arbitrary symmetric square matrix. Then there exists a diagonal matrix $\mathbf{D} = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & & \\ & & \ddots & \\ 0 & & & 0 \\ & \dots & 0 & \lambda_n \end{pmatrix}$ that hold the eigenvalues $\lambda_i \in \mathbb{R}$ in the diagonal and an orthonormal matrix $\mathbf{U} = (\mathbf{u}_1, ..., \mathbf{u}_n)$ that contains the eigenvectors to the eigenvalues $\lambda_1, ..., \lambda_n$ as column vectors, such that

$$\mathbf{AU} = \mathbf{UD}$$

or equivalently

$$\mathbf{A} = \mathbf{UDU}^\top,$$

i.e. the transformation with $\mathbf{A}$ is equivalent to changing into the eigenbasis, rescaling the coordinates with the eigenvalues and changing the basis back.

$\Diamond$

This property of symmetric matrices is the most important ingredient of the principal component algorithm that we will look at now.

2.6 Principal Component Analysis (PCA)

Example: PCA on chromatic pixels The first plot of Figure 2.2 shows the histograms of the red (R), green (G) and blue (B) channel of a natural image. We can see that the value are spread out over the whole dynamic range $[0, 1]$. The other plot in Figure 2.2 shows the RGB values as points in 3D. While the histograms suggested that the whole dynamic range of the cube $[0, 1]^{\mathbf{3}}$ should be covered, we can see that this is definitely not the case. In fact, the RGB values are correlated and cover only a part of the color cube. The main axis of correlation is vaguely $(1, 1, 1)^\top$, which corresponds to the luminance of the three pixels.

$\Diamond$

Image we wanted to encode each pixel value with a single neuron, i.e. we wanted to find an axis along which we would get most of the information about the three RGB values. This means we want to find a direction $\mathbf{v} \in \mathbb{R}^{\mathbf{3}}$ such that the projection $v_i = \langle \mathbf{v}, \mathbf{x}_i \rangle$ of pixel values $\mathbf{x}_i = (x_i^{red}, x_i^{green}, x_i^{blue})^\top$ catches most of the information about all our pixels $\mathbf{x}_i$ in a Euclidean norm sense

$$\mathbf{v} \quad = \quad \text{argmin}_{\mathbf{v}} \frac{1}{m} \sum_{i=1}^{m} ||\mathbf{x}_i - \langle \mathbf{v}, \mathbf{x}_i \rangle \mathbf{v}||^{\mathbf{2}}. \tag{2.5}$$

Which direction should we take? Before we compute the direction from equation (2.5), let us quickly spent some thought on what we would expect. Intuitively, most information about a color pixel is preserved when we convert it into a single gray value. However, this would mean that the axis, we are searching for, is exactly the axis $(1, 1, 1)^\top$ along which the color values have the largest variance. We will see in the following that this is indeed the case.

Before we compute our direction $\mathbf{v}$, we need to transform equation (2.5) a little bit

$$\begin{aligned} & \frac{1}{m} \sum_{i=1}^{m} ||\mathbf{x}_i - \langle \mathbf{v}, \mathbf{x}_i \rangle \mathbf{v}||^{\mathbf{2}} \\ = \quad & \frac{1}{m} \sum_{i=1}^{m} \langle \mathbf{x}_i - \langle \mathbf{v}, \mathbf{x}_i \rangle \mathbf{v}, \mathbf{x}_i - \langle \mathbf{v}, \mathbf{x}_i \rangle \mathbf{v} \rangle \\ = \quad & \frac{1}{m} \sum_{i=1}^{m} \left(\mathbf{x}_i^\top \mathbf{x}_i - 2 \langle \mathbf{v}, \mathbf{x}_i \rangle^2 + \langle \mathbf{v}, \mathbf{x}_i \rangle^2 \mathbf{v}^\top \mathbf{v} \right). \end{aligned}$$

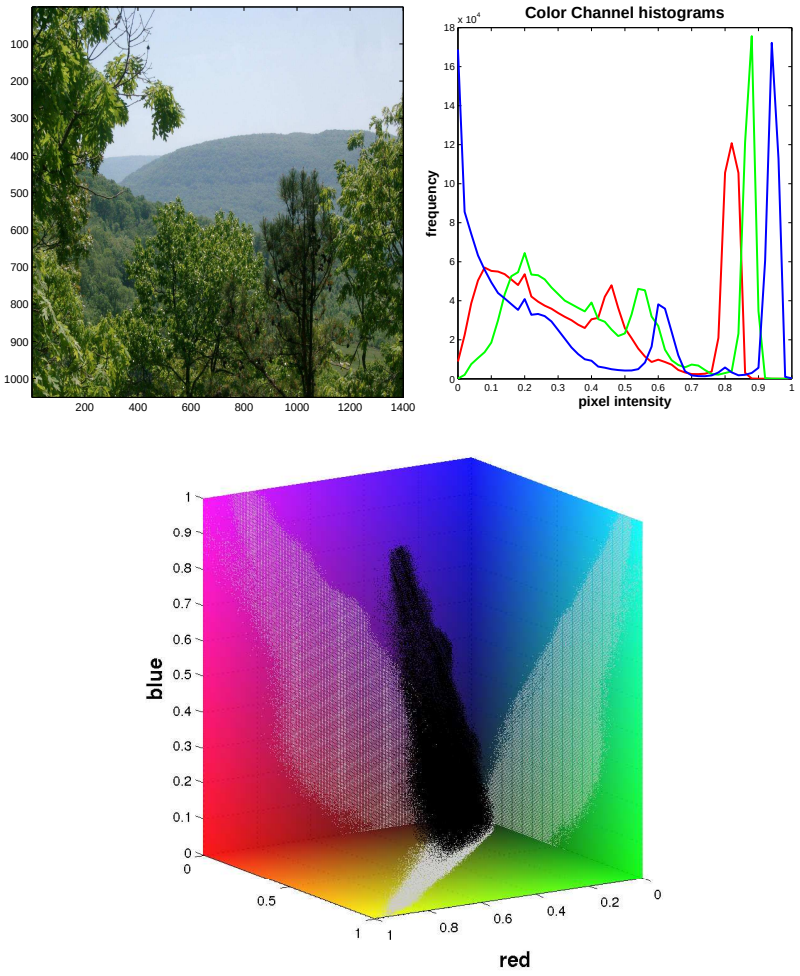


Figure 2.2: **Left Top:** Original natural image. **Right Top:** Histograms over the red, green and blue channel of a natural image. **Center Bottom:**Scatter plot of color pixel values in RGB space. The gray points on the side of the cube are the marginal scatter plots, i.e. the projections on the respective coordinate plane.

Since we are only interested in a direction, we can assume choose our vector $\mathbf{v}$ to have length one $||\mathbf{v}|| = 1$. In fact we already implicitly assumed this when we wrote the reconstruction of $\mathbf{x}_i$ through $\mathbf{v}$ as $\langle \mathbf{v}, \mathbf{x}_i \rangle \mathbf{v}$. With $||\mathbf{v}|| = 1$, the

equation becomes

$$\begin{aligned} & \frac{1}{m}\sum_{i=1}^{m}\left(\mathbf{x}_i^\top\mathbf{x}_i - 2\langle\mathbf{v},\mathbf{x}_i\rangle^2 + \langle\mathbf{v},\mathbf{x}_i\rangle^2\mathbf{v}^\top\mathbf{v}\right) \\ = & \frac{1}{m}\sum_{i=1}^{m}\left(\mathbf{x}_i^\top\mathbf{x}_i - 2\langle\mathbf{v},\mathbf{x}_i\rangle^2 + \langle\mathbf{v},\mathbf{x}_i\rangle^2\right) \\ = & \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i^\top\mathbf{x}_i - \frac{1}{m}\sum_{i=1}^{m}\langle\mathbf{v},\mathbf{x}_i\rangle^2. \end{aligned}$$

Note that if the $\mathbf{x}_i$ had mean zero, i.e. $\frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i = 0$, then $\frac{1}{m}\sum_{i=1}^{m}\langle\mathbf{v},\mathbf{x}_i\rangle^2$ would be the variance of the projections of the $\mathbf{x}_i$ onto our direction $\mathbf{v}$. We can further rewrite this term into

$$\begin{aligned} \frac{1}{m}\sum_{i=1}^{m}\langle\mathbf{v},\mathbf{x}_i\rangle^2 &= \frac{1}{m}\sum_{i=1}^{m}\langle\mathbf{v},\mathbf{x}_i\rangle\langle\mathbf{x}_i,\mathbf{v}\rangle \\ &= \frac{1}{m}\sum_{i=1}^{m}\mathbf{v}^\top\mathbf{x}_i\cdot\mathbf{x}_i^\top\mathbf{v} \\ &= \mathbf{v}^\top\left(\frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i\cdot\mathbf{x}_i^\top\right)\mathbf{v}. \end{aligned}$$

Again, for mean zero $\mathbf{x}_i$, $\frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i\cdot\mathbf{x}_i^\top$ would be the covariance matrix of the $\mathbf{x}_i$.

Now, let us turn to the question how we compute $\mathbf{v}$. After all our transformations we are left with the problem

$$\begin{aligned} \text{minimize}_{\mathbf{v}} \quad & \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i^\top\mathbf{x}_i - \mathbf{v}^\top\left(\frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i\cdot\mathbf{x}_i^\top\right)\mathbf{v} \\ & \text{s.t. } ||\mathbf{v}||^2 = 1. \end{aligned}$$

Note that the first term $\frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i^\top\mathbf{x}_i$ does not depend on $\mathbf{v}$ at all, so we can drop it from the optimization. Furthermore, note that minimizing $-\mathbf{v}^\top\left(\frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i\cdot\mathbf{x}_i^\top\right)\mathbf{v}$ is the same as maximizing $\mathbf{v}^\top\left(\frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i\cdot\mathbf{x}_i^\top\right)\mathbf{v}$. Denoting $\mathbf{C} = \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i\cdot\mathbf{x}_i^\top$ we are therefore left with the final version of the problem

$$\begin{aligned} \text{maximize}_{\mathbf{v}} \quad & \mathbf{v}^\top\mathbf{C}\mathbf{v} \\ & \text{s.t. } ||\mathbf{v}||^2 = 1. \end{aligned}$$

The straightforward way for computing $\mathbf{v}$ would be to compute the derivative, set it to zero and solve for $\mathbf{v}$. However that does not tell us how we can deal with the constraint $||\mathbf{v}|| = 1$. In order to take it into account we need a method called *Lagrange multipliers*. The Lagrange multiplier method is a way of solving exactly those problems above. We will not formally introduce the method here, but only give a rough intuition for the idea behind it.

Remember that the gradient of a function always points into the direction of maximal ascent. Looking at the constraint $||\mathbf{v}||^2 = 1$ or, equivalently, $||\mathbf{v}||^2 - 1 = 0$ we can see that it describes a contour line of the function $||\mathbf{v}||^2 - 1$. Intuitively, since function is constant along the contour line, the gradient of $||\mathbf{v}||^2 - 1$ will be orthogonal to that contour line. Optimizing the function under the constraint now means searching for the optimum along the contour line.

We now state the condition for optimality and then spent some though on why that makes sense. The condition for optimality under the constraint can be written as

$$\nabla \left(\mathbf{v}^\top \mathbf{C}\mathbf{v}\right) = \lambda \nabla(||\mathbf{v}||^2 - 1).$$

The condition means that at the optimum, the gradient of the constraint must have the same orientation as the gradient of the function. Why does that make sense? First, imagine that the optimum of $\mathbf{v}^\top \mathbf{C}\mathbf{v}$ without lies on the contour line. Then, the gradient $\nabla \left(\mathbf{v}^\top \mathbf{C}\mathbf{v}\right)$ is zero and we simply set $\lambda = 0$ and the condition is fulfilled. Now, assume that the optimum of $\mathbf{v}^\top \mathbf{C}\mathbf{v}$ is not on the contour line. Then, at some point, the only way to improve the function $\mathbf{v}^\top \mathbf{C}\mathbf{v}$ is to leave the constraint line, which is equivalent of saying that $\nabla \left(\mathbf{v}^\top \mathbf{C}\mathbf{v}\right)$ is orthogonal to the constraint line. However, we noted before that $\nabla(||\mathbf{v}||^2 - 1)$ is always orthogonal to the constraint line. This means that the two gradients have the same orientation and the condition is fulfilled as well.

Finally, note that we can write the condition for optimality as

$$\text{maximize}_{\mathbf{v}} \qquad \mathbf{v}^\top \mathbf{C}\mathbf{v} - \lambda \left(||\mathbf{v}||^2 - 1\right)$$

because taking the derivative of $\mathbf{v}^\top \mathbf{C}\mathbf{v} - \lambda \left(||\mathbf{v}||^2 - 1\right)$ and setting it to zero exactly yields $\nabla \left(\mathbf{v}^\top \mathbf{C}\mathbf{v}\right) = \lambda \nabla(||\mathbf{v}||^2 - 1)$. The nice feature of the optimization problem above is now, that it does not have constraints and we can simply solve it by setting the derivative to zero and solve for $\mathbf{v}$. In theory we also needed to compute the derivative with respect to lambda and solve for it. However, in this case there is a simple way.

The derivative with respect to $\mathbf{v}$ is given by

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{v}} \left(\mathbf{v}^\top \mathbf{C}\mathbf{v} - \lambda \left(||\mathbf{v}||^2 - 1\right)\right) &= \frac{\partial}{\partial \mathbf{v}} \left(\mathbf{v}^\top \mathbf{C}\mathbf{v} - \lambda \mathbf{v}^\top \mathbf{v} - \lambda\right) \\
&= \frac{\partial}{\partial \mathbf{v}} \mathbf{v}^\top \mathbf{C}\mathbf{v} - \lambda \frac{\partial}{\partial \mathbf{v}} \mathbf{v}^\top \mathbf{v} \\
&= 2\mathbf{C}\mathbf{v} - 2\lambda \mathbf{v}.
\end{aligned}$$

Setting it to zero yields

$$\begin{aligned}
2\mathbf{C}\mathbf{v} - 2\lambda \mathbf{v} &= 0 \\
\Leftrightarrow \mathbf{C}\mathbf{v} &= \lambda \mathbf{v}.
\end{aligned}$$

This is exactly the condition for $\mathbf{v}$ being an eigenvector corresponding to the largest eigenvalue λ of $\mathbf{C}$. Luckily there are many algorithms that carry out

this computation for us. For example, we can use the command `eig` in matlab. If we assume again that the mean of the $\mathbf{x}_i$ is zero, we can even compute the variance of $\langle \mathbf{x}_i, \mathbf{v} \rangle$:

$$\begin{aligned} \frac{1}{m} \sum_{i=1}^{m} \langle \mathbf{x}_i, \mathbf{v} \rangle^2 &= \mathbf{v}^\top \left(\frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i \cdot \mathbf{x}_i^\top \right) \mathbf{v} \\ &= \mathbf{v}^\top \mathbf{C} \mathbf{v} \\ &= \lambda \mathbf{v}^\top \mathbf{v} \\ &= \lambda. \end{aligned}$$

Since λ is the maximal eigenvalue of $\mathbf{C}$, for mean zero signals finding the most informative $\mathbf{v}$ is the same as finding the direction of maximal variance. This is exactly what *Principle Component Analysis (PCA)* does. If the mean of the $\mathbf{x}_i$ is not zero, then, in general, $\mathbf{v}$ will not be the direction of maximal variance but also point in the direction of the mean. That is the reason why the data is usually centered, i.e. the mean is subtracted, before computing PCA.

In its complete version, the PCA algorithm usually computes all eigenvectors instead of simply the largest one. As we saw in the last chapter about eigenvalues, symmetric matrices like $\mathbf{C}$ have eigenvectors that are mutually orthogonal to each other. In terms of PCA this means that after computing the direction of the largest variance, we compute the direction of largest variance which is orthogonal to the first one and so on.

Example: PCA on chromatic pixels (con'd) Figure 2.3 shows the centered pixels and the pixels that have been transformed into the PCA basis. For visualization we also transformed the RGB color planes along with the data. Looking at the right plot of Figure 2.3 shows that the first principle component (now the x-axis) indeed corresponds to the luminance axis. Furthermore, we can observe that the second and third principle component (now the y- and the z-axis, respectively) roughly corresponds to the blue-yellow and the red-green channel, respectively. This is not an coincidence. In fact, this finding is very stable of many natural images.

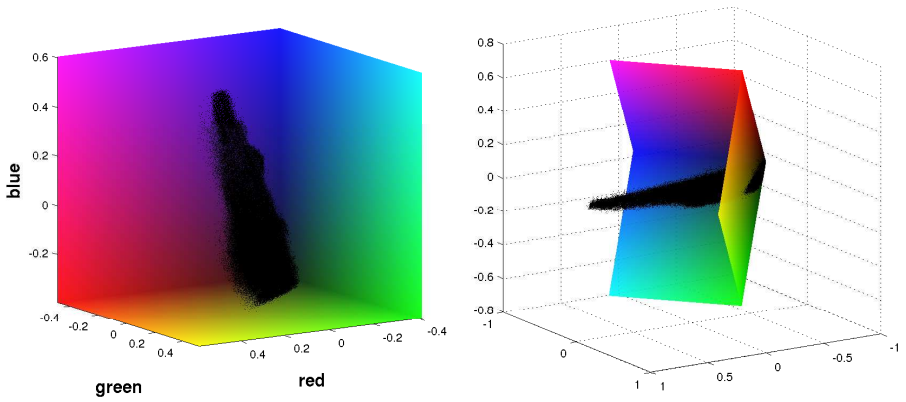


Figure 2.3: **Left:** Scatter plot of centered chromatic natural image pixels. **Right:** Scatter plot of color pixel values in PCA space. We can see that the y-axis in the PCA basis roughly corresponds to the blue-yellow channel whereas the z-axis roughly corresponds to the red-green channel. Both channels are also found in the early visual system.
In both plots the color planes have been shifted with the pixels.

Interestingly, the same color opponencies are also found in the early visual system. What could be the advantage of those channels? PCA can give as an answer to that. Assume that $\mathbf{C}$ is the covariance matrix (i.e. the $\mathbf{x}_i$ have mean zero). Remember from the last chapter, that we can write a symmetric matrix in terms of its eigenbasis as follows $\mathbf{C} = \mathbf{V}\Lambda\mathbf{V}^\top$, where the columns of $\mathbf{V}$ correspond to the eigenvectors and Λ is a diagonal matrix that contains the eigenvalues. Now, we can ask the question how $\mathbf{C}$ looks like in the basis $\mathbf{V}$. A quick computation shows that the transformation that has to be applied is $\mathbf{C} \mapsto \mathbf{V}^\top\mathbf{C}\mathbf{V}$. However, since $\mathbf{C} = \mathbf{V}\Lambda\mathbf{V}^\top$ and $\mathbf{V}^\top\mathbf{V} = \mathbf{V}\mathbf{V}^\top = \mathbf{I}$ we know the answer: In the PCA basis $\mathbf{V}^\top\mathbf{C}\mathbf{V} = \Lambda$, i.e. the covariance matrix is diagonal. This means that all off-diagonal terms are zero and, therefore, the signals are uncorrelated. How could the visual system profit from uncorrelated signals?

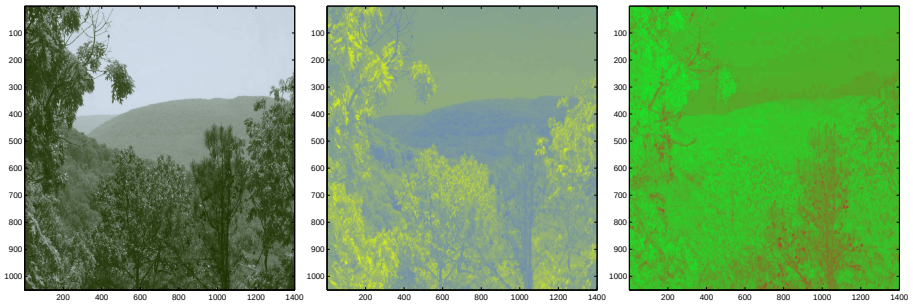


Figure 2.4: Image only represented by the first (left), second (middle) and third (right) principal component in color space.

There are basically two reasons. Imagine, we use three neurons to signal the value of each color pixel, either on neuron for each RGB channel or one neuron for each principle component. Now we make two reasonable assumption about our neurons: Firstly, our neurons have limited expressive power. This means that they can only signal a finite number of pixel values. Secondly, our neuron has limited capacity. This means it can only transmit a limited amount of information in a given time window. For both cases, signalling in the PCA basis is favorable for the neuron. Why is that the case? We already saw, that the histograms in the RGB space covered the full dynamic range $[0, 1]$. A neuron, which has limited expressive power, must distribute the pixel values, that it wants to transmit over the whole dynamic range for all three color channels in RGB space. This means that the resolution is very coarse. In the PCA basis however, we can see from 2.3 that the histograms for the blue-yellow and the red-green channel will be much more compressed. This means that the neurons responsible for those channels, can spend their finite amount of states on that range, thereby obtaining a finer resolution.

The PCA basis is also more favorable in terms of capacity. If a neuron can only transmit a finite amount of information in a given time window, it is not favorable if the three different signals are correlated or, in other words, redundant (ignoring noise issues for the moment). The reason is simply that by transmitting two correlated signals, each signal also transmits some information about the other signal. Since our neurons can only signal a limited amount of information in a given time window, the total amount of information that can be transmitted becomes less when the signals are correlated. However, as we have already seen above, signals are uncorrelated in the PCA basis. This means that signalling in that representation is also favorable in terms of capacity.

$\Diamond$

Chapter 3

Appendix

3.1 Notation and Symbols

- The capital Greek letter sigma "$\sum$" (sigma like sum) denotes a sum over several elements. Usually the components of the sum are indexed with lower case roman letters starting from "i". The starting index is indicated below the "$\sum$" and final index is indicated on top. For example, the sum over n real numbers $x_1, ..., x_n \in \mathbb{R}$ is denoted by

$$x_1 + ... + x_n \quad = \quad \sum_{i=1}^{n} x_i.$$

  Sometimes, when summing over all elements of a set, the set is indicated below the sigma. For example, summing all elements of the set $A = \{1, 2, 3, 4, ..., 15\}$, could be written as $\sum_{x \in A} x$ as well as $\sum_{n=1}^{15} n$.

- The capital Greek letter pi "Π" is used in an analogous manner for products, i.e.

$$x_1 \cdot ... \cdot x_n \quad = \quad \prod_{k=1}^{n} x_k.$$

Bibliography

[1] Eric C. Kandel and James H. Schwartz. *Principles of Neural Science*. Elsevier Science Publishing Co., Inc., 2 edition, 1985.