Machine Learning I
–
Probabilistic Unsupervised Learning

Lecture Notes

Lecture notes for the course Machine Learning I – Probabilistic Unsupervised Learning. The notes are subject to constant changes and do not *provide a definitive reference for the course (the lecture may sometimes provide more, sometimes less information). You will find that some parts of the notes are already relatively well revised. However, other parts are definitely NOT revised – and we are aware that they need revision. Those parts may even contain wrong statements. Also, there is (almost) no literature worked in, yet. The lecture slides/videos are usually in a much better shape. So, as discussed, only use this draft for internal usage, and use it as something it is: a preliminary draft. However, we hope that these preliminary lecture notes will provide some valuable support.* *Lecture and lecture notes by Jörg Lücke Copyright 2010–2021.*

1 Elementary Probability Theory

1.1 An Introductory Experiment

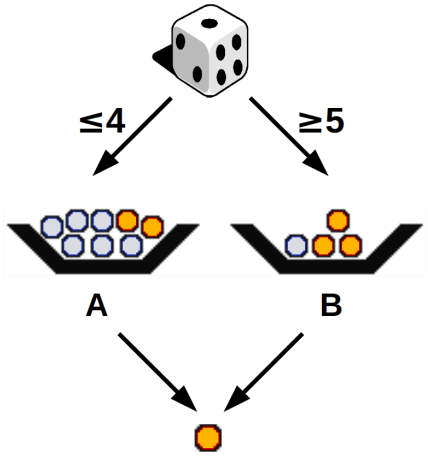


Figure 1: Illustration of the first experiment. We pick fruits from two bowls.

Consider Fig. 1 which illustrates an experiment. The experiment consists of drawing fruits from two bowls according the following procedure:

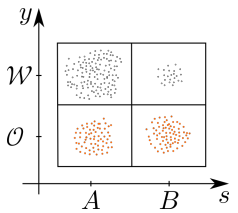
- roll a dice
- if we obtain a number smaller/equal 4 then we choose bowl **A**
- if we obtain a number larger/equal 5 then we choose bowl **B**

- using the chosen bowl, we randomly pick a fruit (i.e., we pick a fruit from it without looking)
- we show/record the result of the experiment
- we put the fruit back into the bowl we drew it from and start over

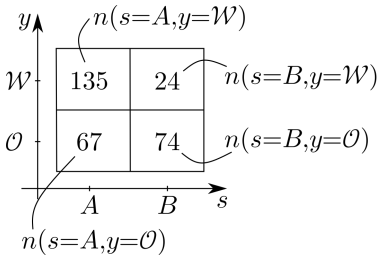
There are two types of fruits: whiteberries and oranges (assumed here to be of the same size). From the experiment displayed in Fig. 1, one could already determine certain probabilities, e.g., the probability to draw a whiteberry from bowl **B**. Alternatively, we can also estimate such a probability (and others) if we have sufficiently many results (i.e., data points) that were generated using the experiment.

experiment n	bowl s	fruit y
1	A	$\mathcal{O}$
2	A	$\mathcal{W}$
3	B	$\mathcal{O}$
4	A	$\mathcal{W}$
5	B	$\mathcal{O}$
6	A	$\mathcal{O}$
7	A	$\mathcal{W}$
⋮	⋮	⋮

(a)



(b)



(c)

Figure 2: (a) Table listing the result of $N = 300$ experiments using the setup of Fig. 1. (b) Representation of the experiments (the data) using a plot with four boxes for each potential outcome and one point for each experiment. (c) Representation of the data as numbers of times each outcome is measured.

Let us follow the latter, i.e., let us consider a number of data points, say $N = 300$, generated using the experiment. These data points will, for instance, look like displayed in Fig. 2(a). Instead of listing the data point as in Fig. 2(a), we can also visualize the data as shown in Fig. 2(b) or Fig. 2(c). Based on the data, we may be interested in different questions that we are able to ask about the experiment. For instance:

- What is the probability that we will pick an orange?
- What is the probability that we will pick a whiteberry from bowl **A**?
- What is the probability that we will pick from bowl **B**?
- Given that we choose bowl **B**, what is the probability that we will pick a whiteberry?
- Given that we picked an orange, what is the probability that we picked it from bowl **B**?

We will be able to answer these and other questions once we have equipped ourselves with the appropriate mathematical tools. In order to do so, let us denote the bowls by s with $s \in \{A, B\}$, and let us denote the fruits by y where $y \in \{\mathcal{W}, \mathcal{O}\}$[1].

Considering the data as displayed in Fig. 2(c), we can already deduce some probabilities.

$$p(s = A) \approx \frac{n(s = A, y = \mathcal{W}) + n(s = A, y = \mathcal{O})}{N} = \frac{202}{300} \approx \frac{2}{3}$$

$$p(s = B) \approx \frac{n(s = B, y = \mathcal{W}) + n(s = B, y = \mathcal{O})}{N} = \frac{98}{300} \approx \frac{1}{3}$$

[1] Variables x and y usually denote outcomes of experiments, and s (or c or z) often stands for something like *source*.

$$p\,(s=A, y=\mathcal{W}) \approx \frac{n\,(s=A, y=\mathcal{W})}{N} = \frac{135}{300} = \frac{45}{100}$$

$$p\,(s=A, y=\mathcal{O}) \approx \frac{n\,(s=A, y=\mathcal{O})}{N} = \frac{67}{300} \approx \frac{22}{100}$$

$$p\,(s=B, y=\mathcal{W}) \approx \frac{n\,(s=B, y=\mathcal{W})}{N} = \frac{24}{300} = \frac{8}{100}$$

$$p\,(s=B, y=\mathcal{O}) \approx \frac{n\,(s=B, y=\mathcal{O})}{N} = \frac{74}{300} \approx \frac{1}{4}$$

$$p\,(y=\mathcal{O}\,|\,s=A) \approx \frac{n\,(s=A, y=\mathcal{O})}{n\,(s=A, y=\mathcal{W}) + n\,(s=A, y=\mathcal{O})} = \frac{67}{202} \approx \frac{1}{3}$$

$$p\,(y=\mathcal{O}\,|\,s=B) \approx \frac{n\,(s=B, y=\mathcal{O})}{n\,(s=B, y=\mathcal{W}) + n\,(s=B, y=\mathcal{O})} = \frac{74}{98} \approx \frac{3}{4}$$

$$p\,(y=\mathcal{W}\,|\,s=A) \approx \frac{n\,(s=A, y=\mathcal{W})}{n\,(s=A, y=\mathcal{W}) + n\,(s=A, y=\mathcal{O})} = \frac{135}{202} \approx \frac{2}{3}$$

$$p\,(y=\mathcal{W}\,|\,s=B) \approx \frac{n\,(s=B, y=\mathcal{W})}{n\,(s=B, y=\mathcal{W}) + n\,(s=B, y=\mathcal{O})} = \frac{24}{98} \approx \frac{1}{4}$$

For the above probabilities, we have used our intuition that a probability of an event X in an experiment is given by the number of times that event is observed divided by the number of times that the experiment is performed. From the examples above, we can determine more general relations between probabilities such as $p\,(s=A)$, $p\,(s=A, y=\mathcal{W})$ or $p\,(y=\mathcal{W}\,|\,s=B)$. For the first relation note that:

$$\begin{aligned} p\,(s=A) &\approx \frac{n\,(s=A, y=\mathcal{W}) \;+\; n\,(s=A, y=\mathcal{O})}{N} \\ &= \frac{n\,(s=A, y=\mathcal{W})}{N} + \frac{n\,(s=A, y=\mathcal{O})}{N} \quad \approx p\,(s=A, y=\mathcal{W}) + p\,(s=A, y=\mathcal{O}) \end{aligned}$$

So we can motivate the following relation that directly relates probabilities (known as **sum rule**):

$$p\,(s=A) = p\,(s=A, y=\mathcal{W}) + p\,(s=A, y=\mathcal{O})\,. \tag{1}$$

Similarly, we obtain another relation between probabilities as follows:

$$\begin{aligned} p\,(s=A, y=\mathcal{W}) &\approx \frac{n\,(s=A, y=\mathcal{W})}{N} \\ &= \frac{n\,(s=A, y=\mathcal{W})}{n\,(s=A, y=\mathcal{W}) + n\,(s=A, y=\mathcal{O})} \frac{n\,(s=A, y=\mathcal{W}) + n\,(s=A, y=\mathcal{O})}{N} \\ &\approx p\,(y=\mathcal{W}|s=A)\,p\,(s=A) \end{aligned}$$

So we can motivate the following rule that directly relates probabilities (known as **product rule**):

$$p\,(s=A, y=\mathcal{W}) = p\,(y=\mathcal{W}\,|\,s=A)\;p\,(s=A)\,. \tag{2}$$

Equations 1 and 2 represent two very important and useful relations between the different types of probabilities. The rules are usually stated in a much more abbreviated form. Using generic variables x and y instead of s and y that common form is as follows:

$$p(x) \;=\; \sum_y p(x, y) \qquad \underline{\textbf{sum rule}} \tag{3}$$

$$p(x, y) \;=\; p(y\,|\,x)\,p(x) \;=\; p(x\,|\,y)\,p(y), \qquad \underline{\textbf{product rule}} \tag{4}$$

where we have stated the product rule in the two possible variants that can be derived. Also not again that we have dropped the somewhat redundant notation of the form $p(y = y_o)$ by replacing $p(y = y_o) \to p(y)$. The meaning however is unchanged: $p(\tilde{y})$ denotes the probability that a random variable distributed according to $p(y)$ takes on value $\tilde{y}$.

Using Eqns. 3 and 4 we can now derive a third very important relation between probabilities:

$$p(y \,|\, x) = \frac{p(x,y)}{p(x)} = \frac{p(x \,|\, y)\, p(y)}{p(x)} \tag{5}$$

$$= \frac{p(x \,|\, y)\, p(y)}{\sum_y p(x,y)} = \frac{p(x \,|\, y)\, p(y)}{\sum_{y'} p(x \,|\, y')\, p(y')} \tag{6}$$

$$\text{and similarly} \quad p(x \,|\, y) = \frac{p(y \,|\, x) p(x)}{\sum_{x'} p(y \,|\, x') p(x')} \qquad \underline{\textbf{Bayes' rule}} \tag{7}$$

But what is e.g. Eqn. 7 good for? Well, remember the third question regarding the first example:

Look! I've picked an orange. What is the probability that I've found it in bowl **B**?

With our mathematical notation, the probability we are looking for is $p(s = B \,|\, y = \mathcal{O})$. In words, probability of $s = B$ given that $y = \mathcal{O}$. This is a difficult conditional probability we do not have direct access to. However, by considering Bayes' theorem, observe that the theorem 'switches' the conditional probabilities, i.e., we can use the theorem to express the (difficult) conditional probability that we seek in terms of (simpler) conditional probabilities.

In the notation for our experiment, Bayes' rule says:

$$p(s = B \,|\, y = \mathcal{O}) = \frac{p(y = \mathcal{O} \,|\, s = B)\, p(s = B)}{\sum_{s'} p(y = \mathcal{O} \,|\, s = s')\, p(s = s')} \tag{8}$$

$$= \frac{p(y = \mathcal{O} \,|\, s = B)\, p(s = B)}{p(y = \mathcal{O} \,|\, s' = A)\, p(s' = A) \,+\, p(y = \mathcal{O} \,|\, s' = B)\, p(s' = B)}. \tag{9}$$

But all these probabilities we have already estimated. Hence, we can finally answer the question by inserting them:

$$p(s = B \,|\, y = \mathcal{O}) = \frac{\frac{3}{4}\,\frac{1}{3}}{\frac{1}{3}\,\frac{2}{3} + \frac{3}{4}\,\frac{1}{3}} = \frac{\frac{3}{4}}{\frac{2}{3} + \frac{3}{4}} \tag{10}$$

$$= \frac{\frac{9}{12}}{\frac{8}{12} + \frac{9}{12}} = \frac{9}{17} \approx \underline{53\%}. \tag{11}$$

So we have our answer: if we have picked an orange in the above experiment, there is a 53% probability that we have picked from bowl $\mathbf{B}$. The conditional probability $p(s = B \,|\, y = \mathcal{O})$ is in this context called *a-posteriori probability* or *posterior probability*. In contrast, the probability $p(s = B)$ is called the *a-priori probability* or *prior probability*. The terminology can be motivate with the experiment as follows: suppose I have done the experiment and hold a fruit in my hands. Then *before* I show you the fruit (i.e., *prior* to me showing you the fruit) the probability that I have picked from bowl **B** is given by $p(s = B)$. However, *after* (*posterior* means 'after') I have shown you that the fruit is an orange, the probability that I have picked from bowl **B** changes. It is now given by the posterior $p(s = B \,|\, y = \mathcal{O})$. For our example, the prior probability of having picked from bowl B has been $\frac{1}{3}$, i.e. about 33%. After we have seen an orange, it raises to 53% , i.e., it is now more likely that we have picked from bowl **B** than having picked from bowl **A**.

1.2 Generalizations and Notations

The introductory experiment of Fig. 1 can easily be generalized in different ways. For instance, we could use more than two berries. If we were using more than one bowl, say six, the random variable s could

take on one of six values instead of two: $S \in \{A, B, C, D, E, F\}$. The second obvious way to generalize the experiment would be to use more than two types of fruits. If we used strawberries (symbol $\mathcal{S}$) and blueberries (symbol $\mathcal{B}$) in addition to whiteberries and oranges then the random variable y could take on any value in the set $\{\mathcal{W}, \mathcal{O}, \mathcal{S}, \mathcal{B}\}$. Of course, both generalizations are possible simultaneously (more bowls and more fruits).

Note a subtle but important notational convention in this respect. If we denote the probability $p(s = B)$ we mean the probability that the random variable s takes on the value B. Later on, we will be concerned with models that use a parameter for each of the probabilities $p(s = A)$ to $p(s = F)$ (or similar), e.g., $p(s = A) = \pi_A$, $p(s = B) = \pi_B$, etc. with $\pi_A, \pi_B, \ldots \in [0, 1]$. If we leave the specific value open, the notation is usually $p(s = S)$ and the parameter π_S. So we denote with s the random variable, and with S the value it takes on (although the value is left unspecified). The distinction of random variable and the concrete value it takes on is always there but we do (as is very common) often simply write $p(s)$ instead of $p(s = S)$, and moreover often write $p(s) = \pi_s$ although this is at close inspection a slight abuse of notation.

The set of possible values that a random variable x can take on is often denoted by Ω_x, i.e., for our experiment

$$\Omega_s \;=\; \{A, B, C, D, E, F\} \;\text{ and }\; \Omega_y \;=\; \{\mathcal{W}, \mathcal{O}, \mathcal{S}, \mathcal{B}\}. \tag{12}$$

Now, considering the 'derivation' of sum rule and product rule above, we could do the same derivation we did for the introductory experiment also for the more general case – and we would arrive at the same results. In these more general setting we consequently get for the **sum rule**:

$$p(s = S) \;=\; \sum_{Y \in \Omega_y} p(s = S, y = Y) \quad \text{and abbreviated} \quad p(s) \;=\; \sum_{y} p(s, y) \tag{13}$$

$$p(y = Y) \;=\; \sum_{S \in \Omega_s} p(s = S, y = Y) \quad \text{and abbreviated} \quad p(y) \;=\; \sum_{s} p(s, y) \tag{14}$$

For the sum rules, the sum goes over all possible values that the variable can take on. If nothing is said about the range of the sum (like on the right-hand-side above), the implicit meaning is usually that it goes over all possible values.

For the **product rule** we get:

$$p(s = S, y = Y) \;=\; p(s = S \,|\, y = Y)\, p(y = Y) \quad \text{and abbreviated} \quad p(s, y) \;=\; p(s \,|\, y)\, p(y) \tag{15}$$

$$p(s = S, y = Y) \;=\; p(y = Y \,|\, s = S)\, p(s = S) \quad \text{and abbreviated} \quad p(s, y) \;=\; p(y \,|\, s)\, p(s) \tag{16}$$

Bayes' Rule can now be derived along the same lines as above and is given by:

$$p(s = S \,|\, y = Y) \;=\; \frac{p(y = Y \,|\, s = S)\, p(s = S)}{\sum_{S' \in \Omega_s} p(y = Y \,|\, s = S')\, p(s = S')} \tag{17}$$

$$\text{and abbreviated} \quad p(s \,|\, y) \;=\; \frac{p(y \,|\, s)\, p(s)}{\sum_{s'} p(y \,|\, s')\, p(s')} \tag{18}$$

And I emphasise again that the abbreviated form is stretching the abbreviation maybe a bit too far.

1.3 Probability Densities

So far we have considered discrete random variables: both s and y could only take on discrete and finitely many values. For continuous variables, consider the experiment of Fig. 3. The experiment depicts an imaginary factory that processes fruits. At one stage of this process, the fruits drop from the ceiling onto a factory floor. There is some air disturbance so that each fruit drops slightly differently, and ends up at a slightly different location on the floor. We would be interested now in knowing what the probability of

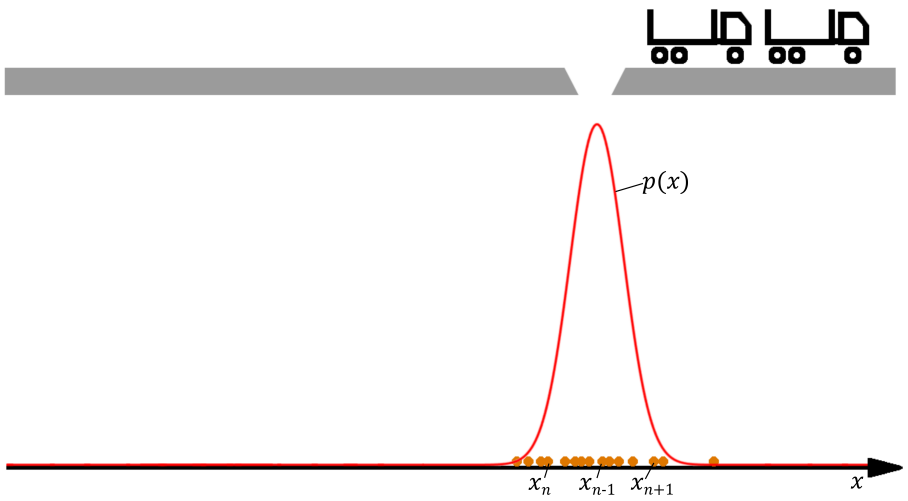


Figure 3: Fruit Factory. An example experiment of a fruit factory. Lorries transport oranges that drop to a one dimensional factory floor. The floor is modeled by the real line $\mathbb{R}$, and a position on the floor is denoted by x.

a fruit is to end up at a certain location X. We assume x to be the corresponding random variable for the position. If we assume the factory floor to be one-dimensional and to extend to infinity on both sides, then the domain of possible values of x is the real line, i.e. $\Omega_x = \mathbb{R}$.

Now thinking briefly about the question how probable it is that x takes on one specific value in Ω_x, we quickly realize that such a probability must be zero (given a specific location X there are infinitely many other locations that the fruit will drop on). Finite probabilities we do get, however, if we ask about the probability of a fruit to land within an interval $[a, b]$. The corresponding notation is $p(x \in [a, b])$ and this probability is between zero and one.

The central mathematical tool to describe continuous random variables is the so-called *probability density function* (sometimes abbreviated by 'probability density', 'density' or 'pdf'). For all examples of this book and for by far most real-world applications, we model a continuous random variable by a probability density function[2].

A probability density is a function that maps all possible values that x can take on (i.e. Ω_x) to a real number: $p : \Omega_x \to \mathbb{R}$. A typical example of a probability density function is, for instance, sketched in Fig. 3 (red bell-like curve). Having a probability density function means that essentially all properties of the corresponding random variable are captured by that function.

But how is the probability density $p(x)$ of our example in Fig. 3 related to the finite probability of a fruit $p(x \in [a, b])$ to lie in an interval $[a, b]$? The answer is given by the following equation, i.e., by an integration of the probability density function:

$$p(x \in [a, b]) \quad = \quad \int_a^b p(x')dx' \tag{19}$$

with p being the probability density function. Obviously, if we make the interval larger, the probability of a fruit to end up in the interval can only increase. This implies that $p(x)$ can not be negative: $p(x) \geq 0$. Furthermore, if the interval extends from minus infinity to plus infinity in our case, it is certain (probability one) that the fruit ends up in the interval:

$$\int_{-\infty}^{\infty} p(x)dx \; = \; 1. \tag{20}$$

Both properties are true not only for one dimensional continuous variables but also for two-dimensional

[2]Probability theory also knows continuous random variables that do not have a probability density to describe them.

(e.g., real two-dim factory floor) and many-dimensional random variables. For general probability densities for these continuous variables, the properties of our example generalize and are given by:

$$\text{for all } \vec{x} \in \Omega_{\vec{x}} \text{ applies:} \quad p(\vec{x}) \;\geq\; 0 \quad \text{and} \quad \int_{\vec{x}\in\Omega_{\vec{x}}} p(\vec{x})\, d\vec{x} \;=\; 1. \tag{21}$$

Note that the notational convention in Machine Learning is to denote the volume element by $d\vec{x}$, e.g. $\mathrm{d}\vec{x} = \mathrm{d}x\mathrm{d}y$ for a two dimensional Euclidean space (in Physics $\mathrm{d}\vec{x}$ can mean an infinitisimal line segment of a path integral). Furthermore, note that we can write (e.g. in two dimensions) $p(\vec{x}) = p(x, y)$ which looks like a joint probability (we will come back to this point later on).

Considering (21) note again the notational conventions used. We use p to denote probabilities as well as probability densities. If x is a discrete random variable, then $p(x)$ denotes (as abbreviated notation) a 'normal' probability (e.g., the probability to obtain a certain number with a given dice). If x is a continuous random variable, then $p(x)$ denotes a probability density: x can take on real numbers ($x \in \mathbb{R}$) for our example, and 'normal' probabilities we only get by integrating w.r.t. x.

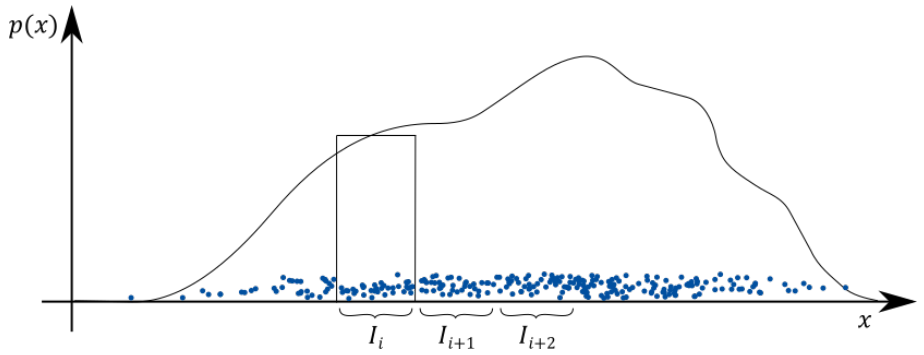


Figure 4: Binning. Characterizing a distribution of fruits on the factory floor using disjoint intervals (bins). The blue dots represent the positions where different fruits have dropped. That is, the data we get is $x_1, x_2, \ldots, x_N$ for a number of N fruits that have dropped to the floor. The function $p(x)$ is supposed to be the probability density function for the random variable x.

1.4 A first estimation of probability densities (data binning / histogram)

Let us briefly discuss how a data distribution can be modeled using data binning (also known as histogram approach). Data binning is a non-parametric approach in the sense that we do not assume any standard shape (like a Gaussian shape) for the distribution. The approach is based on discretizing the data space in order to obtain discrete estimations to the continuous values an underlying density $p(x)$ can take on. A 'bin' is basically another name for interval. We will consider non-overlapping intervals that cover the whole real line, and one particular interval we will denote by I_i. Let us further denote the position in the middle of a given interval I_i by $\bar{x}_i$ (i.e., for an interval $I_i = [a, b]$ the position is $\bar{x}_i = (a + b)/2$). For each interval I_i we can count the number of fruits with a position within the interval, and we denote the number by n_i (in Fig. 4 the interval I_i contains thirty-seven fruits, i.e. $n_i = 37$).

Using the notation, we can for sufficiently many fruits estimate the probability of a fruit to end up in interval I_i: the estimation is given by $\frac{n_i}{N}$. As $p(x \in I_i)$ is representing just his probability, we can use Eqn. 19 to relate the fruit positions $x_1, x_2, \ldots, x_N$ (our data) to the probability density $p(x)$:

$$\frac{n_i}{N} \approx p(x \in I_i) = \int_{I_i} p(x)dx \approx p(\bar{x}_i) \int_{I_i} dx \approx p(x_i)\Delta_i, \tag{22}$$

where we have assumed the intervals to be small such that $p(x)$ does not vary much within an interval. Δ_i is the size of interval I_i (i.e., for $I_i = [a, b]$ we have $\Delta_i = b - a$). Using (22) we consequently arrive

at the following equation which estimates the values of $p(x)$ at the positions $\bar{x}_i$:

$$\Rightarrow \quad p(\bar{x}_i) = \frac{n_i}{N\Delta_i}. \tag{23}$$

The values $p(\bar{x}_i)$ computed with this formula from data $x_1, x_2, \ldots, x_N$ are usually plotted using bars with width Δ_i. The resulting figure is then often referred to as a *histogram*. The procedure itself is often called *binning* but other naming conventions exist. Often, all intervals are assumed to have the same size.

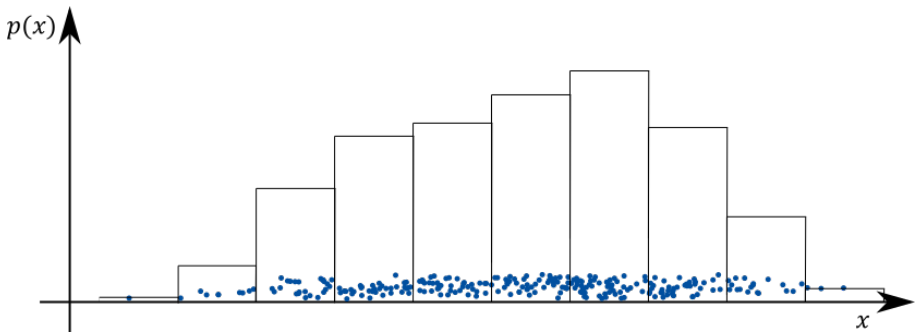


Figure 5: Full histogram of the same data as in Fig. 4, the N data points $x_1, x_2, \ldots, x_N$ are denoted as blue dots. The bins height is determined by the number of fruits which are located in the interval divided by the Number of fruits $\frac{n_i}{N}$.

1.5 Expectation and Covariance

We will later discuss concrete examples of probability densities or probabilities for discrete latents. For both, a number of important properties are defined including mean, variance, skewness, etc. A central definition for all these properties is the *expectation value*.

Given that x is a continuous random variable with probability density $p(x)$, and $f(x)$ a function that depends on x, then the *expectation value*, denoted by $\langle f \rangle_p$, is defined as follows:

$$\langle f \rangle_p \quad = \quad \int_{x' \in \Omega_x} p(x') f(x') dx', \tag{24}$$

where Ω_x is again the set of all possible values of x. The one-dimensional integral becomes a higher-dimensional (volume) integral if the variable is a vector $\vec{x}$.

Similarly, for a discrete random variable x the expectation value is defined as follows:

$$\langle f \rangle_p \quad = \quad \sum_{X' \in \Omega_x} p(x = X') f(X'), \tag{25}$$

If f is a constant, i.e. $f(x) = f_o$, than the expectation value is:

$$\langle f \rangle_p \quad = \quad f_o, \tag{26}$$

in the continuous as well as in the discrete case (show why as an exercise).

The simplest non-trivial expectation value is the one of the identity function, $f(x) = x$, which is also known as the *mean* of the random variable:

$$\text{Mean}(x) \;=\; \langle x \rangle_p \quad = \quad \int_{x' \in \Omega_x} p(x') x' dx', \quad \text{or} \;\; \text{Mean}(x) \;=\; \langle x \rangle_p \;=\; \sum_{X' \in \Omega_x} p(x = X')\, X'. \tag{27}$$

As a first example, we can think of a discrete random variable which has as possible output the numbers $\Omega_x = \{1, 2, 3, 4, 5, 6\}$ all appearing with the same probability of $1/6$ (think of a dice). Inserting $p(x = X') = \frac{1}{6}$ for all $X' \in \Omega_x$ we obtain:

$$\langle x \rangle_p = \sum_{x' \in \Omega_x} p(x')x' = \sum_{x'=1}^{6} \frac{1}{6} x' = \frac{21}{6} = 3.5 \tag{28}$$

So the mean of a dice (or the expectation $\langle x \rangle_p$) is 3.5. If we roll a number lower than 3.5 this is below our expectation; if we roll a number above 3.5 this is above our expectation.

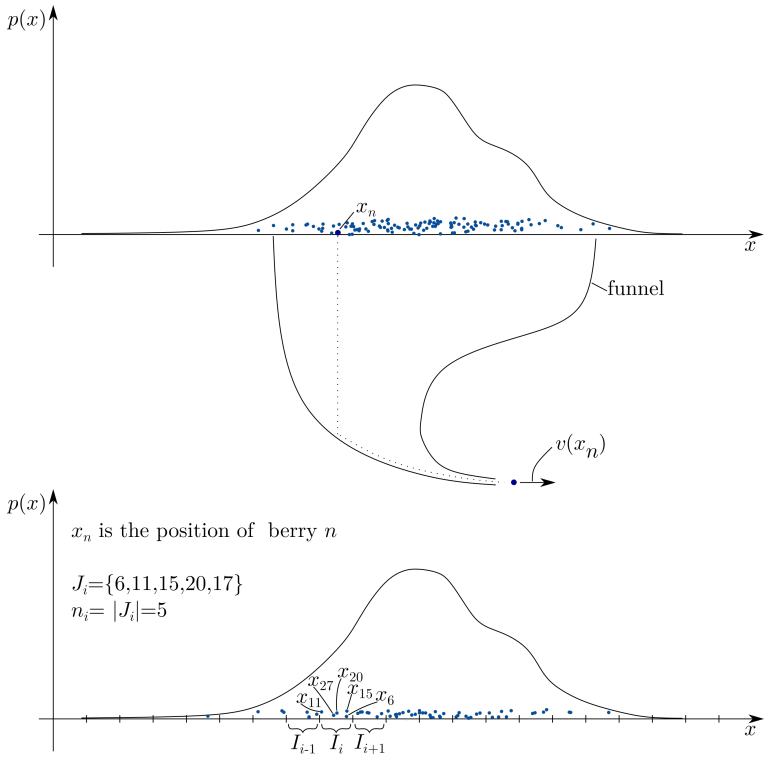


Figure 6: Fruits through a funnel.

A second example is a contiuous random variable the can take one any value on the real line between the numbers $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}$ (we assume $\alpha < \beta$). A corresponding probability density is then given by:

$$p(x|\alpha, \beta) \quad = \quad \begin{cases} \frac{1}{\beta - \alpha} & x \in [\alpha, \beta] \\ 0 & \text{otherwise} \end{cases} \tag{29}$$

which is called the *uniform distribution*. It follows:

$$\langle x \rangle_p \quad = \quad \int_{\mathcal{R}} p(x')x'dx' = \frac{1}{\beta - \alpha} \int_{\alpha}^{\beta} x'dx' \tag{30}$$

$$= \quad \frac{1}{\beta - \alpha} \frac{1}{2} x^2 \Big|_{\alpha}^{\beta} = \frac{1}{2} \frac{1}{\beta - \alpha} (\beta^2 - \alpha^2) = \frac{1}{2} \frac{1}{\beta - \alpha} (\beta - \alpha)(\beta + \alpha) = \frac{\alpha + \beta}{2} \tag{31}$$

Similar to the dice, a getting a number below $\frac{\alpha+\beta}{2}$ is lower than our expectation. Getting a number larger than $\frac{\alpha+\beta}{2}$ is above our expectation.

The two examples above may be considered as intuition for the definition of the expectation value w.r.t. x. To develop an intuition for the meaning of a general expectation value w.r.t. a general function $v(x)$. Consider the experiment of Fig. 6. The function $v(x)$ specifies how the position x of the random variable translates into a velocity. What would be the meaning of the expectation value $\langle v \rangle_p$ if the variable x is distributed according to p (i.e., if variable x has $p(x)$ as probability density)? For this, we proceed similarly (but more generally) as for the binning approach above. Note that Ω_x is assumed equal to the real line $\mathbb{R}$. We then start by considering again a partition $\{I_i\}_{i \in N}$ of the real line in the form of disjoint intervals I_i of the same size: $\bigcup_i I_i = \mathbb{R}$ and $I_i \cap I_j = \emptyset$ for all $i \neq j$. Let us for simplicity assume that there are always enough data points in an interval I_i. At the same time, all the intervals will be assumed small enough such that the functions $p(x)$ and $v(x)$ only vary little across all values of x in an interval. We can then derive:

$$
\begin{aligned}
\langle v \rangle_p &= \int_{x \in \mathbb{R}} p(x) v(x) dx \;=\; \int_{-\infty}^{+\infty} p(x) v(x) dx && (32)\\
&= \sum_i \int_{x \in I_i} p(x) v(x) dx && (33)\\
&\approx \sum_i \int_{x \in I_i} v(\overline{x}_i) p(\overline{x}_i) dx \quad (v \text{ and } p \text{ change little from } x \in I_i \text{ to } \overline{x}_i) && (34)\\
&= \sum_i v(\overline{x}_i) p(\overline{x}_i) \int_{x \in I_i} dx \;=\; \sum_i v(\overline{x}_i) p(\overline{x}_i) |I_i| && (35)\\
&= \sum_i v(\overline{x}_i) \frac{n_i}{N} \quad \text{(formula from binning)} && (36)\\
&= \sum_i v(\overline{x}_i) \frac{1}{N} \sum_{n \in J_i} 1 \quad \text{(rewrite } n_i\text{)} && (37)\\
&= \frac{1}{N} \sum_i \sum_{n \in J_i} v(\overline{x}_i) \quad \text{(rearrange)} && (38)\\
&\approx \frac{1}{N} \sum_i \sum_{n \in J_i} v(x_n) \quad (v \text{ changes little from } x_n \text{ to } \overline{x}_i) && (39)\\
&= \frac{1}{N} \sum_n v(x_n)\,. && (40)
\end{aligned}
$$

So the expectation value $\langle v \rangle_p$ actually is the mean velocity of the fruits which exit the funnel (the average of velocity across all fruits). The above derivation, on the one hand, does make the definition of an expectation value more plausible (the expectation value acquires a physical meaning for the experiment in Fig. 6). On the other hand, the formula is not using any specifics of the experiment, it is very generally true. Indeed, the formula is of central importance in Machine Learning and Computational Statistics. It is sometimes referred to as *Sampling Formula* and is given in its general form by:

$$
\langle v \rangle_p \;=\; \int_{\vec{x} \in \Omega_{\vec{x}}} p(\vec{x})\, v(\vec{x})\, d\vec{x} \;\approx\; \frac{1}{N} \sum_n v(\vec{x}_n) \quad \text{where } \vec{x}_n \text{ is distributed according to } p(\vec{x}). \tag{41}
$$

For scalar functions of a random vector $\vec{x}$ (with D entries) $v : \mathbb{R}^D \to \mathbb{R}$ applies analogously

$$
\langle v \rangle_p = \int_{\vec{x} \in \mathbb{R}^D} p(\vec{x}) v(\vec{x})\, d\vec{x} \;\approx\; \frac{1}{N} \sum_{n=1}^{N} v(\vec{x}^{(n)}) \tag{42}
$$

For non-scalar functions (vectors and matrices) the approximations can be generalized in a straightforward way because the expectation values of vectors and matrix-valued functions are defined elementwise.

One possibility to recognize the importance of the Eqn. 42 is to notice that it connects two 'worlds': On the one hand, the left-hand-side and middle of (42) are purely mathematical objects, i.e. functions

and integrals; on the other hand, the right-hand-side of (42) contains the data, i.e. its value can be directly computed from the data. This property makes Eqn. 42 very useful to model data with probability functions and to use these functions for algorithms.

Let us go back to the scalar case and Eqn. 41. The function $v(x)$ can be any function. For the choice of $v(x) = x$, we called the expectation value $\langle v \rangle_p$ the *mean*, which is something like the center or center-of-mass of the distribution (Exercise: how could that be shown?). Another quantity that will be helpful in specifying properties of random variables is the *variance* of a random variable, $\mathrm{Var}(x)$. It is defined by:

$$\begin{aligned} \mathrm{Mean}(x) &= \langle x \rangle_p = \int p(x) x dx \\ \mathrm{Var}(x) &= \left\langle (x - \langle x \rangle_p)^2 \right\rangle_p = \int p(x)(x - \langle x \rangle_p)^2 dx \end{aligned} \tag{43}$$

Just as the mean value gives us information about the 'center' of a probability density function, the variance tells us the dispersion around the 'center'. The variance is a quantitative way to say how concentrated or broad our distribution is.

1.6 Mean and variance: two examples

To go further in this direction, we will consider first a vector of random variables. Then, we can write for example : $p(\vec{X}) = p(X, Y)$. At the same time, the covariance between X, Y is:

$$\text{Cov}(x, y) = \langle (x - \langle x \rangle_p)(y - \langle y \rangle_p) \rangle_p = \int p(x, y)(x - \langle x \rangle_p)(y - \langle y \rangle_p) dxdy \tag{44}$$

This number is telling us about the co-occurrence of X and Y, it answers quantitatively the question 'What has X to do with Y?' The correlation is its normalized equivalent:

$$\text{Cor}(x, y) = \frac{\text{Cov}(x, y)}{\sqrt{\text{Var}(x)\text{Var}(y)}} \tag{45}$$

For example, if X and Y are independent i.e. $p(x, y) = p(x)p(y)$, we have:

$$\begin{aligned} \text{Cov}(x, y) &= \int p(x)p(y)(x - \langle X \rangle)(y - \langle Y \rangle) dxdy \\ &= \left(\int p(x)xdx - \langle X \rangle \right) \left(\int p(y)ydy - \langle Y \rangle \right) = 0 \end{aligned} \tag{46}$$

in the independent case, 'X has nothing to do with Y'. But be careful, because the converse is not true! For a general vector $\vec{X} = (X_1, ..., x^{(n)})$ of random variables, we have,

$$\begin{aligned} \text{Cov}(x_i, x_j) &= \int p(x_1, ..., x_{\mathcal{D}})(x_i - \langle X_i \rangle)(x_j - \langle X_j \rangle) dx_1 ... dx_{\mathcal{D}}. \\ &= \int p(x_1, ..., x_{\mathcal{D}}) dx_1 ... dx_{i-1} dx_{i+1} ... dx_{j-1} dx_{j+1} ... dx_{\mathcal{D}} (x_i - \langle X_i \rangle)(x_j - \langle X_j \rangle) dx_i dx_j. \\ &= \int p(x_i, x_j)(x_i - \langle X_i \rangle)(x_j - \langle X_j \rangle) dx_i dx_j \end{aligned} \tag{47}$$

$p(x_i, x_j)$ is the joint probability density function of the random variables X_i and X_j. It was obtained by integrating out the rest of the random variables.
[NOTE: TEXT TO BE REVISED HERE]
called marginalized function, and the associated correlation function is defined as above.

For completeness, we would like to mention another useful magnitude, defined in a similar way to the previously seen mean value, called conditional expectation:

$$\langle f \mid y \rangle = \int_{x \in X} p(x \mid y) f(x) dx \tag{48}$$

1.7 Probability Density Functions and Distributions

We have already considered two probability distributions above: a constant discrete probability distribution for a fair dice, and continuous probability distribution for with a constant probability in an interval $[a, b]$ (the uniform distribution). Let us here mention a number of further probability distributions.

The most well known continuous distribution is presumably the Gaussian probability distribution. Its probability density $p(x \mid \mu, \sigma^2)$ is defined based on two real parameters: μ and $\sigma^2 > 0$.

$$p(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}) \tag{49}$$

The standard parameterization (49) of the Gaussian distribution (also referred to as the *normal distribution*) is such that the parameter μ is the distribution mean and the parameter σ^2 is the distribution variance. This is in contrast to, say, the standard parameterization of the uniform distribution, see Eqn. 29. One reason for the importance of the Gaussian distribution is the central limit theorem which, in its reduced form, states that the mean of N copies of the same random variable tend to a normal distribution as N goes to infinity.

Further distributions are given below:

Bernoulli (discrete):

$$\begin{aligned} p(x=1|\pi) &= \pi;\ p(x=0|\pi) = 1-\pi \\ i.e.\ \ p(x|\pi) &= \pi^x(1-\pi)^{1-x} \end{aligned} \tag{50}$$

A possible example is tossing a coin. In this case (provided that the coin is fair) $\pi = 0.5$.

Poisson (discrete):

$$p(x|\lambda) = \exp(-\lambda)\frac{\lambda^x}{x!} \tag{51}$$

It was developed to model a number of events (outcome of the same process) that take place in a fixed temporal window, with a known average and their occurrence is independent of the time elapsed since the previous event. One example could be, the number of people that go to a given bank, from 9 in the morning to 3 P.M. (during a normal day, without a financial crisis), provided that we know the average number of people per day.

Binomial (discrete):

$$p(x|n, p_0) = \binom{n}{x} p_0^x (1-p_0)^{n-x} \tag{52}$$

This distribution represent the probability of obtaining x successes out of n independent trials. Being p_0 the probability of success on each trial.

Beta (continuous):

$$p(x|\alpha, \beta) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1}(1-x)^{\beta-1} \tag{53}$$

2 Decision Theory

So far, we have seen different probability densities, joint probabilities, and different useful operations on them. Now, we want to see them in action in a particular example. Suppose that we have an input vector $\vec{x}$ together with a corresponding vector $\vec{t}$ of target variables, and our goal is to predict $\vec{t}$ given a new value of $\vec{x}$. The joint probability distribution $p(\vec{x}, \vec{t})$ provides a complete summary of the uncertainty associated with these variables.

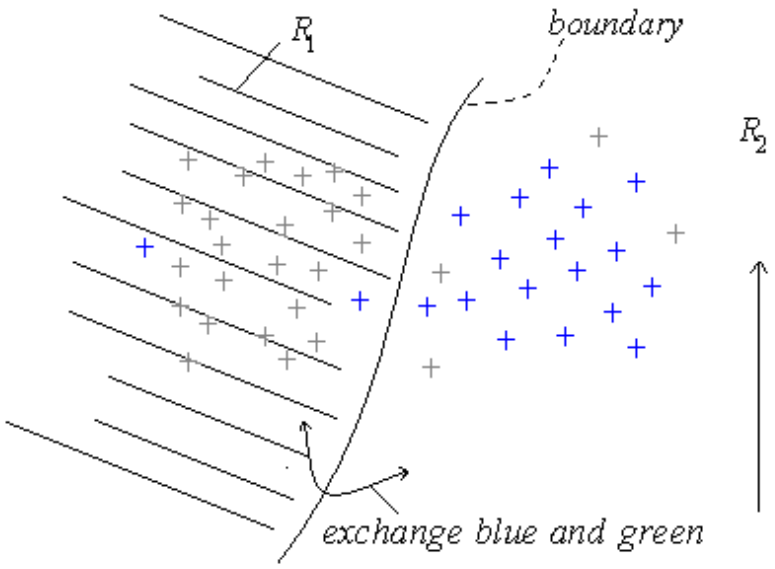


Figure 7: Schematic representation of the clustering process

In what follows, we will think of $\vec{t}$ as a discrete vector, what transforms our problem into one of clustering. As a first example we will consider our vector to be a single number, one or two that points to the cluster to which each data $\vec{x}$ belongs to. Let us consider also, that the different $\vec{x}$ are distributed all over the real plane as depicted in figure 7, all we know about the data is that they belong to two different categories (blueberries (1) and green berries (2), for example) and that we want to separate them into the two groups R_1 and R_2. One way to do it in this case, is to reduce the misclassification error.

What is misclassification, then?

$$P^{\text{mis}} = p(\vec{x} \in \mathcal{R}_1, c = 2) + p(\vec{x} \in \mathcal{R}_2, c = 1) \tag{54}$$

$$= \int_{\mathcal{R}_1} p(\vec{x}, c = 2) d\vec{x} + \int_{\mathcal{R}_2} p(\vec{x}, c = 1) d\vec{x} \tag{55}$$

We are free to choose the decision rule which assigns each point $\vec{x}$ to one of the two classes. To minimize $p(mis)$ we should make sure that each $\vec{x}$ is assigned to the class that has the smaller value of the integrand in (55). To illustrate this point, suppose that $p(\vec{x}, 2) \geq p(\vec{x}, 1)$. Let us shift the area $R1$, by a volume ρ, such that the point $\vec{x}$ doesn't belong to $R1$ any more, as depicted in figure **??**, so now we have:

$$\mathcal{R}_1^{new} = \mathcal{R}_1 \setminus \rho$$
$$\mathcal{R}_2^{new} = \mathcal{R}_2 \cup \rho$$

Since in this process we have removed a point that naturally belongs to 2 from $\mathcal{R}_1$, the P^{mis} decreases. We can see it analytically in equation, where we start from the new areas, and show how the newly defined P^{mis} is lower than the old one.

$$
\begin{aligned}
P_{\text{mis}} &= \int_{\mathcal{R}_1} p(c=2|\vec{x})p(\vec{x})d\vec{x} + \int_{\mathcal{R}_2} p(c=1|\vec{x})p(\vec{x})d\vec{x} && (56)\\
&= \int_{\mathcal{R}_1\setminus\rho} p(c=2|\vec{x})p(\vec{x})d\vec{x} + \int_{\rho} p(c=2|\vec{x})p(\vec{x})d\vec{x} + \int_{\mathcal{R}_2} p(c=1|\vec{x})p(\vec{x})d\vec{x} && (57)\\
&\geq \int_{\mathcal{R}_1\setminus\rho} p(c=2|\vec{x})p(\vec{x})d\vec{x} + \int_{\rho} p(c=1|\vec{x})p(\vec{x})d\vec{x} + \int_{\mathcal{R}_2} p(c=1|\vec{x})p(\vec{x})d\vec{x} && (58)\\
&= \int_{\mathcal{R}_1\setminus\rho} p(c=2|\vec{x})p(\vec{x})d\vec{x} + \int_{\mathcal{R}_2\cup\rho} p(c=1|\vec{x})p(\vec{x})d\vec{x} && (59)\\
&= \int_{\mathcal{R}_1^{new}} p(c=2|\vec{x})p(\vec{x})d\vec{x} + \int_{\mathcal{R}_2^{new}} p(c=1|\vec{x})p(\vec{x})d\vec{x} \;=\; P_{\text{mis}}^{new} && (60)
\end{aligned}
$$

The above inequality holds for any region ρ for which applies

$$p(c=2|\vec{x}) \quad \geq \quad p(c=1|\vec{x}) \quad \text{for all } \vec{x} \text{ in region } \rho \tag{61}$$

because in this case it follows:

$$
\begin{aligned}
p(c=2|\vec{x}) &\geq p(c=1|\vec{x}) && \text{for all } \vec{x} \text{ in region } \rho && (62)\\
\Rightarrow p(c=2|\vec{x})\,p(\vec{x}) &\geq p(c=1|\vec{x})\,p(\vec{x}) && \text{for all } \vec{x} \text{ in region } \rho && (63)\\
\Rightarrow \int_{\rho} p(c=2|\vec{x})\,p(\vec{x})d\vec{x} &\geq \int_{\rho} p(c=1|\vec{x})\,p(\vec{x})d\vec{x} && && (64)
\end{aligned}
$$

Considering the derivation above, we observe that we can always decrease the misclassification error P_{mis} to a smaller error P_{mis}^{new} if we can find a region for which condition (61) holds. This observation provokes the following question:

Can we define regions $\mathcal{R}_1$ and $\mathcal{R}_2$ such that it is impossible for us to find a region ρ that satisfies (61)?

Let us try. We simply use the knowledge we have gained and define regions as follows:

$$
\begin{aligned}
\mathcal{R}_1^{opt} &= \{\vec{x}\,|\,p(c=1|\vec{x}) > p(c=2|\vec{x})\} && (65)\\
\mathcal{R}_2^{opt} &= \{\vec{x}\,|\,p(c=2|\vec{x}) > p(c=1|\vec{x})\} && (66)
\end{aligned}
$$

These are well-defined disjunct regions, and there is a misclassification error defined by these regions which we will call P_{mis}^{opt}. Now, by definition of $\mathcal{R}_1^{opt}$ there is no ρ (and in fact no single $\vec{x}$) which satisfies (61). Hence, there is no redefinition of the regions $\mathcal{R}_1^{opt}$ (and $\mathcal{R}_2^{opt}$) possible that decrease the misclassification error. We can, therefore, conclude that the regions (67) are *optimal* in the sense that they minimize the misclassification error.

Having defined the optimal regions $\mathcal{R}_1^{opt}$ and $\mathcal{R}_2^{opt}$, the points $\vec{x}$ that do separate the regions are given by

$$\mathcal{S}^{opt} \quad = \quad \{\vec{x}\,|\,p(c=1|\vec{x}) = p(c=2|\vec{x})\} \tag{67}$$

The set $\mathcal{S}$ is often referred to as *decision boundary* and $\mathcal{S}^{opt}$ as optimal decision boundary. If the data space is scalar (as for our one-dimensional example), one simply drops the vector ($\vec{x} \rightarrow x$). In the one-dimensional case, the set $\mathcal{S}^{opt}$ consists of single points.

But how can we now find the point that represents the decision boundary for our $x \in \mathbb{R}$ example?

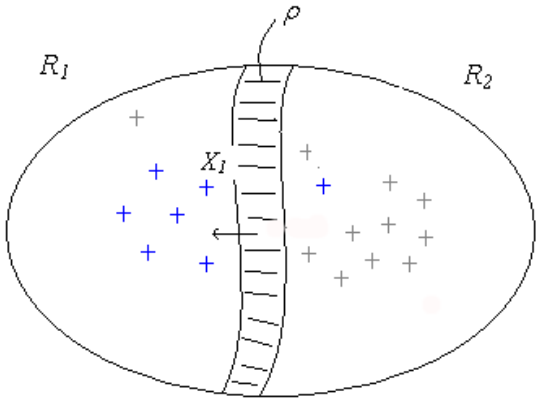


Figure 8: Boundary shift. ρ represents the changed volume

Well, first we need to compute $p(c = 1|\vec{x})$ and $p(c = 2|\vec{x})$. These are conditional probabilities which we are not given directly. However, we can use Bayes' Rule and obtain:

$$p(c = 1|\vec{x}) = \frac{p(\vec{x} \,|\, c = 1)\; p(c = 1)}{\sum_{C'=1}^{2} p(\vec{x} \,|\, c = C')p(c = C')} \tag{68}$$

$$p(c = 2|\vec{x}) = \frac{p(\vec{x} \,|\, c = 2)\; p(c = 2)}{\sum_{C'=1}^{2} p(\vec{x} \,|\, c = C')p(c = C')} \tag{69}$$

For the one-dimensional case, we again drop the vector ($\vec{x} \to x$). Now Eqns. 68 and 69 are functions of the position x. We can plot these functions. The set $\mathcal{S}^{opt}$ is then made up of those points where the two functions become equal. We have thus solved our task.

Once we have computed the set $\mathcal{S}^{opt}$ we realize that it consists of more than one point. Why? What would be the intuition?

For vectors $\vec{x} \in \mathbb{R}^2$, e.g., for a real two-dimensional factory floor, the set $\mathcal{S}^{opt}$ consists (except of boundary cases) of one-dimensional subspaces. These subspaces separate the two-dimensional regions.

The whole derivation can be generalized, of course, to the case of more than two berry types (more than two classes). In such a case it is better to maximize the probability of correct classification rather than to minimize the misclassification probability.

Take-home-messages

- The posterior emerges as the important object for optimal decisions. More generally the posterior will turn out to be the crucial object for inference and learning.

- Did we learn how to do clustering? What is missing?
  Answer : $p(\vec{x} \mid c_1)$, $p(\vec{x} \mid c_2)$ and $p(c_1)$ ($p(c_2) = 1 - p(c_1)$).

Different Options for Classification

- **Generative Models** (the approach above); try to find / learn $p(c)$, $p(\vec{x} \mid c)$ and then compute $p(c \mid \vec{x})$.

- **Discriminative Approaches**; try to find a good model (approximation) for $p(c \mid \vec{x})$ directly. (extreme case is to find $f(\vec{x}) \in \{1, 2\}$; called hard decision)

Question : Why doesn't it work with $p(\vec{x}|c = c_i)p(c = c_i)$? Answer : Well, we do have to know how $p(c = c_i)$ behaves.

Let us rethink what we have been doing up to now. Suppose we are given data as depicted in **??**,

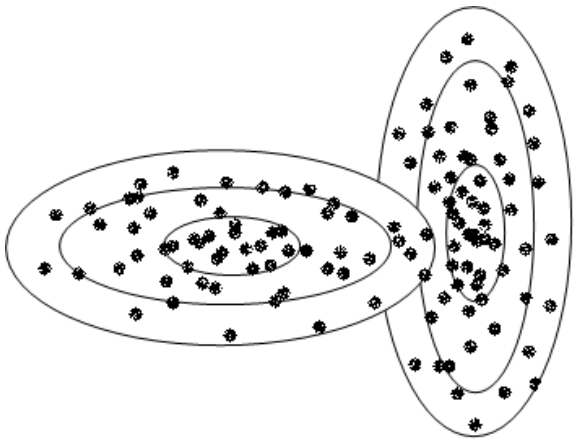


Figure 9: Visualization of a clustering process, data points are given as black dots. Our partition, due to the clustering process is given by the drawn ellipses.

and that we know $p(c = 1)$ and $p(c = 2)$ as well as $p(\vec{x} \,|\, c = 1)$ and $p(\vec{x} \,|\, c = 2)$, are we done then? The answer is yes, we can construct the quantities we need to take the decision by plugging them into Bayes' formulas, i.e., by using Eqns. 68 and 69. Using $p(c = 1 \,|\, \vec{x})$ and $p(c = 2 \,|\, \vec{x})$ we can then compute $\mathcal{R}_i$ using Eqns. 65 and 66 – so far so good.

However, note that we do not have that distributions $p(c)$ and $p(\vec{x} \,|\, c)$, we only have the data. So how can we obtain $p(\vec{x}|c)$ and $p(c)$? This is an important question and the answer will be developed during the whole course. For our example remember, first of all, that the following magnitudes, that help us to characterize distributions have discrete analogues (or approximations) based only on the observed data points.

Density Estimation

Remember: $\langle \nu(x) \rangle := \int p(x)\nu(x)dx \approx \frac{1}{N} \sum_{n=1}^{N} \nu(x^{(n)})$

Mean value: $\nu(x) = x \Rightarrow \langle x \rangle \approx \frac{1}{N} \sum_{n=1}^{N} x^{(n)} = \bar{x}$

Variance: $\nu(x) = (x - \bar{x})^2 \Rightarrow \langle \nu(x) \rangle \approx \frac{1}{N} \sum_{n=1}^{N} (x^{(n)} - \bar{x})^2$

Parametric Methods

A possible way to estimate these quantities is to assume a function or group of functions together with some adjustable parameters. This assumption is useful when we know something about the process which generates the data. For example, suppose we knew that we have a Gaussian distribution (and the points

are independently generated), with mean μ and variance σ^2. In this case we can simply use:

$$p(x \mid c=1) = \frac{1}{\sqrt{2\pi\sigma_1^2}} \exp(-\frac{1}{2}\frac{(x-\mu_1)^2}{\sigma_1^2}), \tag{70}$$

$$p(x \mid c=2) = \frac{1}{\sqrt{2\pi\sigma_2^2}} \exp(-\frac{1}{2}\frac{(x-\mu_2)^2}{\sigma_2^2}). \tag{71}$$

Because of the sampling formula, we can estimate μ_c and σ_c^2 from the data. To do so, recall that a data point is of the form $(x^{(n)}, c^{(n)})$, where $x^{(n)}$ is the position of data point and where $c^{(n)}$ is its *label*. There are N data points ($n = 1, \ldots, N$) and n is called the *index* of the data point. Let us assume that all blueberries have label $c = 1$ and all oranges have label $c = 2$. Then we define the index set $\mathcal{I}_1$ to consist of those indices n that have label $c^{(n)} = 1$, and let the set $\mathcal{I}_2$ consist of all those indices that have label $c^{(n)} = 2$:

$$\mathcal{I}_1 = \{n : c^{(n)} = 1\}, \quad \text{and} \quad \mathcal{I}_2 = \{n : c^{(n)} = 2\}. \tag{72}$$

In other words: the set $\mathcal{I}_1$ contains the blueberries, the set $\mathcal{I}_2$ contains the oranges. Now, the positions $x^{(n)}$ for all $n \in \mathcal{I}_1$ can be assumed to follow a Gaussian distribution (with parameters μ_1 and σ_1), and all positions $x^{(n)}$ with $n \in \mathcal{I}_2$ can be assumed to follow another Gaussian distribution (with parameters μ_2 and σ_2). Using the sampling formula, we can consequently estimate:

$$\mu_c = \frac{1}{N}\sum_{n \in \mathcal{I}_c}^{N} x^{(n)}, \tag{73}$$

$$\sigma_c^2 = \frac{1}{N}\sum_{n=1}^{N} (x^{(n)} - \mu_c)^2. \tag{74}$$

Note that we have assumed a principle form for $p(x|c)$ that was parameterized by μ and σ^2. We therefore often call such a method *parametric method*. Note that, there are also non-parametric methods. For instance, in principle one could use a histogram/binning approach to estimate $p(x \mid c)$.

Finally, we also need estimates for $p(c = 1)$ and $p(c = 2)$. Using the index sets, these are simply given by the number of blueberries divided by the number of all fruits (for $c = 1$) and correspondingly for the oranges:

$$p(c=1) = \frac{|\mathcal{I}_1|}{|\mathcal{I}_1| + |\mathcal{I}_2|}; \;\; p(c=2) = \frac{|\mathcal{I}_2|}{|\mathcal{I}_1| + |\mathcal{I}_2|}, \tag{75}$$

where $\mathcal{I}_c$ is simply the number of indices in the set $\mathcal{I}_c$ (number of fruits), formally this is:

$$|\mathcal{I}_c| = \sum_{n \in \mathcal{I}_c} 1. \tag{76}$$

3 Maximum Likelihood

We did consider the problem of fitting a parametric model to a set of data points. Examples of such models were the uniform distribution and the Gaussian distribution. We used the sampling formula, which gave us the parameters of the respective distributions. However, not all distributions are as conveniently defined as the Uniform or the Gaussian distributions, i.e., in many cases we can not easily use the sampling formula to find corresponding parameters. To be able to treat more difficult cases, we need a more general method to find parameters also for very intricate probability distributions. One such more general method is based on the so called *data likelihood*. If we assume the data points x_1 to $x^{(n)}$ to be independently and identically distributed (iid), the *data likelihood* is defined as follows:

$$L(\Theta) = p(x^{(1)}, ..., x^{(n)} \mid \Theta) = \prod_{n=1}^{N} p(x^{(n)}|\Theta) \tag{77}$$

where Θ denotes the set of all parameters of the distribution (e.g. for a Gaussian $\Theta = (\mu, \sigma^2)$).

Does this definition make sense? Suppose $p(x|\Theta)$ is (1-dim) Gaussian. What is the maximum of $L(\Theta)$?

Instead of maximizing $L(\Theta)$ directly, let us maximize $\log(L(\Theta))$ instead. Since $\log(\cdot)$ is a well behaved monotonically increasing function, and as $L(\Theta)$ is always a positive real number, a maximum of $L(\Theta)$ is always also a maximum of $\log(L(\Theta))$ and vice versa. The function $\mathcal{L}(\Theta) = \log(L(\Theta))$ is commonly referred to as *data log-likelihood* or just *log-likelihood*. If we want to make salient that it also depends on the data points, we write $\mathcal{L}(x^{(1)}, \ldots, x^{(N)}; \Theta)$ or $\mathcal{L}(x^{(1):N)}; \Theta)$ instead of $\mathcal{L}(\Theta)$.

For the Gaussian with parameters $\Theta = \big(\mu, \sigma^2\big)$ we first derive the expression of the function $\mathcal{L}(x^{(1)}, \ldots, x^{(N)}; \Theta)$:

$$\begin{aligned}
\mathcal{L}(x^{(1)}, \ldots, x^{(N)}; \Theta) &= \sum_{n=1}^{N} \log\Big(p\big(x^{(n)} \,|\, \Theta\big)\Big) \;=\; \sum_{n=1}^{N} \log\Big(\mathcal{N}\big(x^{(n)};\, \mu, \sigma^2\big)\Big) \\
&= \sum_{n=1}^{N} \log\Big(\frac{1}{\sqrt{2\pi\sigma^2}} \exp\big(-\frac{1}{2\sigma^2}(x^{(n)} - \mu)^2\big)\Big) \\
&= \sum_{n=1}^{N} \Big(-\frac{1}{2}\log(2\pi\sigma^2) - \frac{(x^{(n)} - \mu)^2}{2\sigma^2}\Big) \\
&= -\frac{N}{2}\log(2\pi) - \frac{N}{2}\log(\sigma^2) - \frac{1}{2}\sum_{n=1}^{N} \frac{(x^{(n)} - \mu)^2}{\sigma^2}
\end{aligned}$$

Now for finding the maximum,

$$\begin{aligned}
\frac{\partial}{\partial \mu}\, \mathcal{L}(x^{(1)}, \ldots, x^{(N)}; \Theta) &\stackrel{!}{=} 0 \\
\frac{\partial}{\partial \sigma^2}\, \mathcal{L}(x^{(1)}, \ldots, x^{(N)}; \Theta) &\stackrel{!}{=} 0
\end{aligned} \tag{78}$$

We first look at the derivative w.r.t. the mean μ:

$$\begin{aligned}
0 \;&\stackrel{!}{=}\; \frac{\partial}{\partial \mu}\mathcal{L}(x^{(1)}, \ldots, x^{(N)}; \Theta) = \sum_{n=1}^{N} \frac{\partial}{\partial \mu} \log(p(x^{(n)}|\mu, \sigma^2)). \\
&= \sum_{n=1}^{N} \frac{\partial}{\partial \mu}(-\frac{1}{2}\log(2\pi) - \frac{1}{2}\log\sigma^2 - \frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}) = \sum_{n=1}^{N} \frac{1}{\sigma^2}(x^{(n)} - \mu) \\
\Rightarrow\; &\sum_{n=1}^{N} x^{(n)} - \mu \sum_{n=1}^{N} 1 = 0 \\
\Rightarrow\; &\mu = \frac{1}{N}\sum_{n=1}^{N} x^{(n)}.
\end{aligned}$$

Next, we take derivatives w.r.t. σ. It is easier to first replace σ^2 by $\tilde{\sigma}$ and to consider the derivative w.r.t. $\tilde{\sigma}$:

$$\begin{aligned}
0 \stackrel{!}{=} \frac{\partial}{\partial\tilde{\sigma}}\mathcal{L}(x^{(1)},\ldots,x^{(N)};\Theta) &= \sum_{n=1}^{N}\frac{\partial}{\partial\tilde{\sigma}}\log(p(x^{(n)}|\mu,\tilde{\sigma})).\\
&= \sum_{n=1}^{N}\frac{\partial}{\partial\tilde{\sigma}}(-\frac{1}{2}\log(2\pi)-\frac{1}{2}\log\tilde{\sigma}-\frac{1}{2}\frac{(x-\mu)^2}{\tilde{\sigma}}).\\
&= \sum_{n=1}^{N}(-\frac{1}{2})(\frac{1}{\tilde{\sigma}}+(x^{(n)}-\mu)^2(-\frac{1}{\tilde{\sigma}})^2).\\
\Rightarrow &\sum_{n=1}^{N}(-\frac{1}{2})(1-\frac{(x^{(n)}-\mu)^2}{\tilde{\sigma}})=0.\\
\Rightarrow &\; N-\frac{1}{\tilde{\sigma}}\sum_{n=1}^{N}(x^{(n)}-\mu)^2=0.\\
\Rightarrow &\; \sigma^2=\frac{1}{N}\sum_{n=1}^{N}(x^{(n)}-\mu)^2.
\end{aligned}$$

Using the log-likelihood and given a set of N data points, we hence obtain the same estimations for μ and σ as we obtained using the sampling formula. The advantage of the log-likelihood is that it can be applied more generally. The log-likelihood is well grounded in probability theory and can, for our purposes, be thought of as a measure for similarity between data points and a data model (the data model being given by $p(x\,|\,\Theta)$).

Equivalences

- Maximizing the likelihood of a Gaussian mode $\Leftrightarrow$
- Minimizing the squared error cost function $\Leftrightarrow$
- Minimizing the data coding cost in bits.

For most densities we can not solve the maximum likelihood equation analytically. In those cases we could use a '*gradient ascent method*'.There is a huge collection of literature on such methods. For the Maximum Likelihood approach there is one method which is of particular importance: *The Expectation Maximization* algorithm (**EM**).

4 The Expectation Maximization (EM) Algorithm

THIS CHAPTER HAS TO BE REVISED - USE SEPARATE TEX FILE TO IMPORT DERIVATIONS

Given a set of N independent data points, we want to find the parameters Θ that maximize the likelihood function. Since the logarithm is a continuous and monotonically increasing function, we can find the parameters in an easier way by maximizing the log-likelihood function L.

$$
\begin{aligned}
L &= \sum_{n=1}^{N} \log\{p(x^{(n)}|\Theta)\} && (79)\\
&= \sum_{n} \log\{\sum_{c} p(x^{(n)}, c|\Theta)\} \\
&= \sum_{n} \log\{\sum_{c} q_n(c) \frac{p(x^{(n)}, c|\Theta)}{q_n(c)}\} \\
&\geq \sum_{n} \{\sum_{c} q_n(c) \log(\frac{p(x^{(n)}, c|\Theta)}{q_n(c)})\} =: F(q, \Theta) && (80)
\end{aligned}
$$

In the first step above we use the fact that the data points are independent. In the second, we use the sum rule. The numbers $q_n(c), c \in \{1, ..., C\}$ are positive and one and add up to 1. This construction allows us to use Jensen's inequality in the next, arriving at an expression in which the summations are outside the logarithms. The expression $F(q, \Theta)$ is called *free energy function* and we can now maximize F, which will prove to be more convenient than maximizing L. We do so by repeating systematically the following two steps until convergence , (see figures 10 and 11 below)

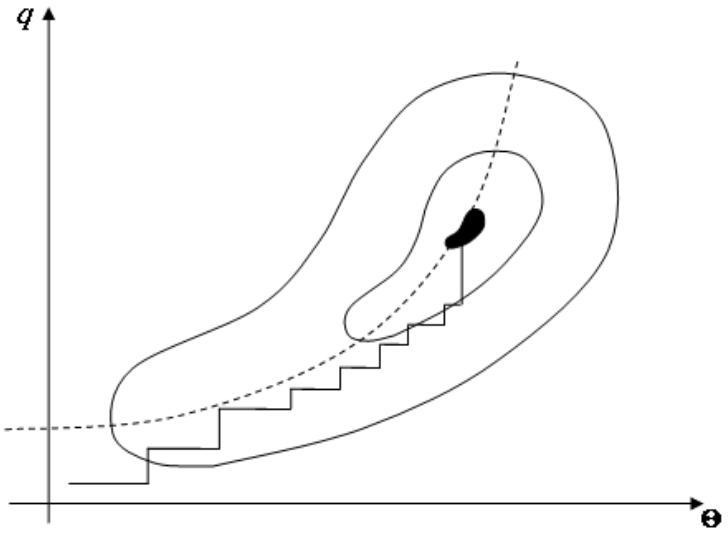


Figure 10: *Schematic representation of the EM algorithm. The algorithm updates the q's during the E-step (vertical line), and the Θ parameters during the M-step (horizontal line), until it converges to a maximum (black area)*

- E-step: maximize $F(q, \Theta)$ with respect to q (while keeping Θ fixed)
- M-step: maximize the resultant expression $F(q, \Theta)$ with respect to Θ (while keeping q fixed)

How can we find the q that maximizes $F(q, \Theta^{old})$? Let's have a look.
As was proven in equation 79, the following inequality holds:

$$\begin{aligned} L(\Theta^{old}) &\geq F(q, \Theta^{old}) \\ \Rightarrow L(\Theta^{old}) &- F(q, \Theta^{old}) \geq 0 \end{aligned} \tag{81}$$

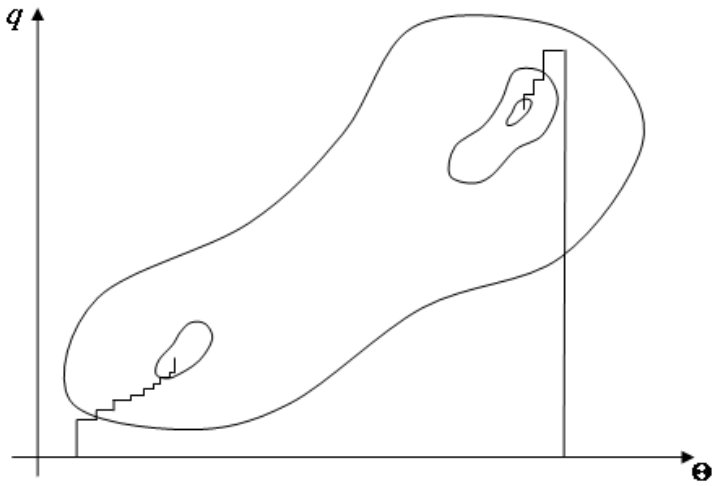


Figure 11: *Schematic representation of the EM algorithm. Convergence to different local maxima depending on the initial conditions*

We want to find q that minimizes the expression above (can't get smaller than 0).
Consider the equation 82 below:
[NOTE: TEXT TO BE REVISED HERE. hint: define KL-divergence]

$$\begin{aligned} 0 \leq\ & L(\Theta^{old}) - F(q, \Theta^{old}) \\ =\ & \sum_n \log p(x^{(n)}|\Theta^{old}) - \sum_n \sum_c q_n(c) \log\{\frac{p(x^{(n)}, c|\Theta^{old})}{q_n(c)}\} \\ =\ & \sum_n \log p(x^{(n)}|\Theta^{old}) - \sum_n \sum_c q_n(c) \log\{\frac{q_{\text{new}}(c; x^{(n)}, \Theta^{old}) p(x^{(n)}|\Theta^{old})}{q_n(\Theta^{old})}\} \\ =\ & \sum_n \log p(x^{(n)}|\Theta^{old}) - \sum_n \sum_c q_n(c) \log\{\frac{q_{\text{new}}(c; x^{(n)}, \Theta^{old})}{q_n(c)}\} - \sum_n \sum_c q_n(c) \log(p(x^{(n)}|\Theta^{old})) \\ =\ & \sum_n \log p(x^{(n)}|\Theta^{old}) - \sum_n \log p(x^{(n)}|\Theta^{old}) \underbrace{\sum_c q_n(c)}_{1} - \sum_n \sum_c q_n(c) \log\{\frac{q_{\text{new}}(c; x^{(n)}, \Theta^{old})}{q_n(c)}\} \\ =\ & -\sum_n \sum_c q_n(c) \log\{\frac{q_{\text{new}}(c; x^{(n)}, \Theta^{old})}{q_n(c)}\} \\ =\ & D_{\text{KL}}(q_n(c), q_{\text{new}}(c; x^{(n)}, \Theta^{old})) \end{aligned} \tag{82}$$

$$D_{\mathrm{KL}}(q(c), p(c)) \quad = \quad -\sum_c q(c) \log \left(\frac{p(c)}{q(c)}\right) \tag{83}$$

$$p(\vec{x}, c \,|\, \Theta) \quad = \quad p(\vec{x} \,|\, c, \Theta)\, p(c \,|\, \Theta) \tag{84}$$

$$\vec{\mu}^{\,\mathrm{old}} \;=\; \vec{\mu}^{\,\mathrm{new}} \tag{85}$$

Where in the first line, we rewrote the expression from equation 79. In the second we use the product rule to rewrite the numerator inside the logarithm. We split as a sum in the next line, the term inside the logarithm which does not contain c. The obtained term (last one) is rearranged in the next step to obtain the second in the fourth line (note that only $q_n(c)$ depend on c inside the second summation). Finally the first two expressions cancel out giving as a result the (so-called) Kullback-Leibler divergence between the functions $q(\cdot)$ and $p(\cdot \,|\, x^{(n)}, \Theta^{old})$.
Since the KL-divergence is zero if and only if $q_n(c) = q_{\mathrm{new}}(c; x^{(n)}, \Theta^{old})$, we have:

$$q_n^{max}(\Theta^{old}) = q_{\mathrm{new}}(c; x^{(n)}, \Theta^{old}) \quad \text{E-step} \tag{86}$$

Thus the evolution of the likelihood during an EM iteration can be understood as follows:

$$L(\Theta^{old}) \underbrace{=}_{\texttt{E-step}} F(q^{max}_{\Theta^{old}}, \Theta^{old}) \underbrace{\leq}_{\texttt{M-step}} F(q^{max}_{\Theta^{old}}, \Theta^{new}) \underbrace{\leq}_{\texttt{Jensen's}} L(\Theta^{new}) \tag{87}$$

This process never decreases the likelihood! In practice it usually increases the likelihood to a likelihood maximum (which can however be local).

We already know how to find the maximal q for $F(q, \Theta)$, the E-step; $q_n(c) := p(c|x^{(n)}, \Theta)$. But how do we find the parameters Θ^{new} that maximize $F(q, \Theta)$ in the M-step?
Assuming smoothness we know that in the maximum we have:

$$\frac{\partial F(q, \Theta)}{\partial \Theta} \quad = \quad 0 \tag{88}$$

Let us first rewrite F:

$$\begin{aligned}
F(q^{max}_{\Theta^{old}}, \Theta) \quad &= \quad \sum_n \sum_c q_{\mathrm{new}}(c; x^{(n)}, \Theta^{old}) \log\left(\frac{p(x^{(n)}, c|\Theta)}{q_{\mathrm{new}}(c; x^{(n)}, \Theta^{old})}\right) \\
&= \quad \sum_n \sum_c q_{\mathrm{new}}(c; x^{(n)}, \Theta^{old})(\log p(x^{(n)}, c|\Theta) - \log q_{\mathrm{new}}(c; x^{(n)}, \Theta^{old})) \\
&= \quad \underbrace{\sum_n \sum_c q_{\mathrm{new}}(c; x^{(n)}, \Theta^{old})(\log p(x^{(n)}|c, \Theta) + \log p(c|\Theta))}_{Q(\Theta)} \\
&\quad + \sum_n \underbrace{(-\sum_c q_{\mathrm{new}}(c; x^{(n)}, \Theta^{old}) \log q_{\mathrm{new}}(c; x^{(n)}, \Theta^{old}))}_{H(p(.|x^{(n)}, \Theta^{old}))\ \texttt{or}\ H_n(C)\ \texttt{where}\ C \sim p(.|x^{(n)}, \Theta^{old})}
\end{aligned}$$

The first term above $Q(\Theta)$ contains all the dependencies on the parameters Θ. Note that this term can also be rewritten as an expectation value $Q(\Theta) = \sum_n E[\log p(x^{(n)}, c|\Theta)]$. We want to maximize this

expectation value, therefore the name *Expectation Maximization* (see Dempster 1977, and Neal + Hinton 1998). Thus:

$$\frac{\partial}{\partial \Theta_i}\mathcal{F}(q^{new},\Theta) \quad = \quad 0 \text{ for all parameters } \Theta_i \text{ simultaneously} \tag{89}$$

$$\Rightarrow \frac{\partial}{\partial \Theta_i}F(q^{max}_{\Theta^{old}},\Theta) = 0 \Leftrightarrow \frac{\partial}{\partial \Theta_i}Q(\Theta) = 0 \tag{90}$$

$$\frac{\partial}{\partial \Theta_i}F(q^{mew},\Theta) \quad = \quad \frac{\partial}{\partial \Theta_i}(Q(\Theta) + \sum_n H_n(p(.|x^{(n)},\Theta^{old}))) = \frac{\partial}{\partial \Theta_i}Q(\Theta) \tag{91}$$

$$\Rightarrow \frac{\partial}{\partial \Theta_i}F(q^{max}_{\Theta^{old}},\Theta) = 0 \Leftrightarrow = \frac{\partial}{\partial \Theta_i}Q(\Theta) = 0 \tag{92}$$

Where Θ_i is one of the $\Theta = \{\Theta_1, ..., \Theta_p\}$ parameters.
We can see that $Q(\Theta)$ is the relevant term for this step, sometimes it is also written as $Q(\Theta, \Theta^{old})$.
Important remark: EM also works more general, in particular $x_1, ..., x^{(n)}$ do not have to be independent (e.g. time series).

To summaries, the following steps are involved in the **EM Algorithm**:

Randomly initialize the parameters Θ^{init}
Set $\Theta^{old} = \Theta^{init}$
repeat
  Compute $q_{\text{new}}(c; x^{(n)}, \Theta^{old})$ for all *hidden causes* c and *data points* $x^{(n)}$ (E-step)
  Compute the parameters Θ^{new} for which $\frac{\partial}{\partial \Theta_i}Q(\Theta) = 0$ for all i (*additionally, you should actually make sure that* Θ^{new} *is a maximum of* $Q(\Theta)$) (M-step)
  Set $\Theta^{old} = \Theta^{new}$
until *until convergence*;

---

5 Gaussian Mixture Models for Clustering — A First Derivation of a Concrete EM Learning Algorithm

Let us model the situation shown in Fig. 11 using the following generative model:

$$p(c\,|\,\Theta) \quad = \quad \pi_c \tag{93}$$

$$p(x\,|\,c,\Theta) \quad = \quad \frac{1}{\sqrt{2\pi\sigma_c^2}}\exp(-\frac{1}{2\sigma_c^2}(x-\mu_c)^2) \tag{94}$$

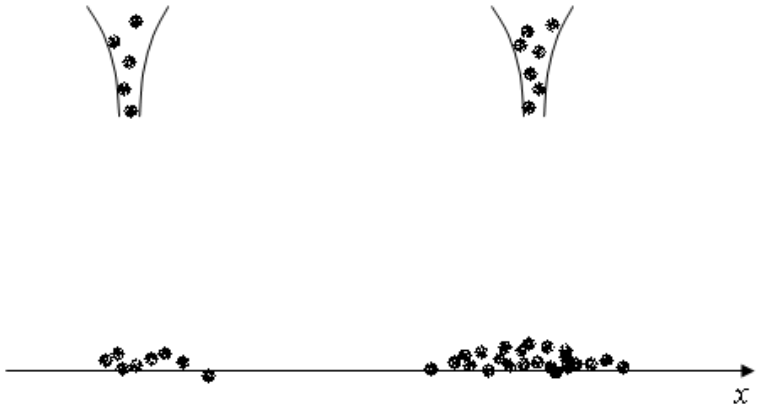


Figure 12: *Blueberries dropping from the ceiling*

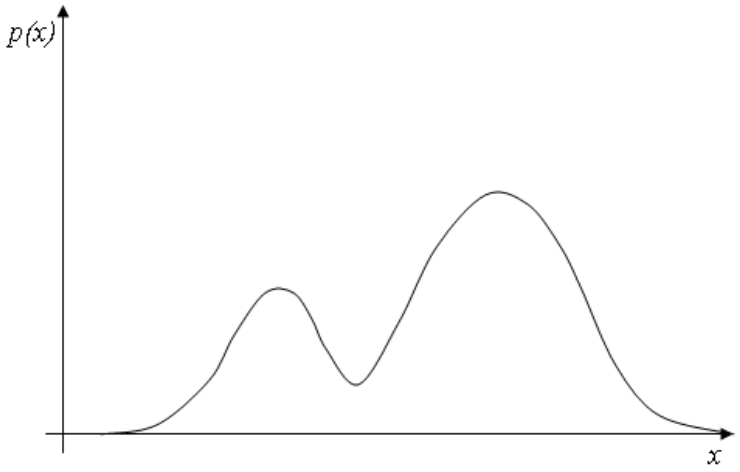


Figure 13: *Schematic representation of the optimal pdf for the problem above*

where $\Theta = (\pi_1, \pi_2, \mu_1, \mu_2, \sigma_1^2, \sigma_2^2)$ is the set of parameters of the model. Note that, $\pi_1 + \pi_2 = 1$. Using the sum rule and then the product rule, the probability density function over the real axis then takes the form:

$$\begin{aligned} p(x \,|\, \Theta) &= \sum_c p(x, c|\Theta) = \sum_c p(x|c, \Theta) p(c|\Theta) \\ &= \sum_c \pi_c \frac{1}{\sqrt{2\pi\sigma_c^2}} \exp(-\frac{1}{2\sigma_c^2}(x - \mu_c)^2) \end{aligned} \tag{95}$$

We want to find those parameters Θ that maximize the likelihood of N data points given the above generative model, i.e., we want to find the maximum of the function

$$\mathcal{L}(\Theta) = \sum_{n=1}^{N} \log p(x^{(n)}|\Theta).$$

Because of the sum over c inside the logarithm, the maximization of $\mathcal{L}(\Theta)$ is difficult (exercise). We will use EM instead. That is, we maximize the lower bound $F(q, \Theta)$ (free energy function) and we expect the algorithm to arrive at some parameters Θ which result in a pdf similar to the one shown in figure 12

We will apply the EM algorithm (derived in general in the previous section) to the case of the Gaussian Mixture Model (GMM) given by (93) and (94).

5.1 The Free Energy of the GMM

The first step to derive an EM algorithm for a GMM is to write down its corresponding free energy function. The part of the free energy that directly depends on the generative model is the logarithm of the joint, which can be reformulated (using product and logarithm rules):

$$\log\big(p(x, c \,|\, \Theta)\big) \;=\; \log\big(p(x, c \,|\, \Theta)\big) \;=\; \log\big(p(x \,|\, c, \Theta) p(c \,|\, \Theta)\big) \;=\; \log\big(p(x \,|\, c, \Theta)\big) + \log\big(p(c \,|\, \Theta)\big)$$

For the generative model (93) and (94) we have:

$$\log\big(p(c \,|\, \Theta)\big) = \log\big(\pi_c\big) \tag{96}$$

$$\log\big(p(x^{(n)} \,|\, c, \Theta)\big) = -\frac{1}{2}\log(2\pi) - \frac{1}{2}\log(\sigma_c^2) - \frac{1}{2\sigma_c^2}(x^{(n)} - \mu_c)^2. \tag{97}$$

The free energy is consequently given by:

$$\begin{aligned} \mathcal{F}(q, \Theta) &= \sum_n \sum_{c'} q^{(n)}(c')(\frac{1}{2}\log(2\pi) - \frac{1}{2}\log(\sigma_{c'}^2) - \frac{1}{2}\frac{(x^{(n)} - \mu_{c'})^2}{\sigma_{c'}^2} + \log(\pi_{c'})) \\ &\quad - \sum_n \sum_{c'} q^{(n)}(c') \log\big(q^{(n)}(c')\big). \end{aligned} \tag{98}$$

5.2 The E-step

Given the free energy (98), the first task to accomplish for the EM-Algorithm is the computation of the E-step, i.e.,

$$q_{\text{new}}^{(n)}(c) = \text{argmax}_q\{\mathcal{F}(q, \Theta^{\text{old}})\} \tag{99}$$

But we do know the solution for this maximization task, which we derived in general in the last section. The solution is given by setting $q^{(n)}_{\text{new}}(c)$ equal to the posterior probability $p(c \,|\, xN, \Theta^{\text{old}})$ for each N:

$$\begin{aligned} q^{(n)}_{\text{new}}(c) &= p(c \,|\, x^{(n)}, \Theta^{old}) \\ &= \frac{p(x^{(n)} \,|\, c, \Theta^{old})\, p(c \,|\, \Theta^{old})}{\sum_{c'} p(x^{(n)} \,|\, c', \Theta^{old})\, p(c' \,|\, \Theta^{old})} \\ &= \frac{(2\pi(\sigma_c^{old})^2)^{-\frac{1}{2}} \exp(-\frac{1}{2(\sigma_c^{old})^2}\left(x^{(n)} - \mu_c^{old}\right)^2)\pi_c^{old}}{\sum_{c'} \left(2\pi(\sigma_{c'}^{old})^2\right)^{-\frac{1}{2}} \exp(-\frac{1}{2(\sigma_{c'}^{old})^2}\left(x^{(n)} - \mu_{c'}^{old}\right)^2)\pi_{c'}^{old}} \\ &=: q_{\text{new}}(c; x^{(n)}, \Theta^{old}), \end{aligned} \tag{100}$$

where we have introduced $q_{\text{new}}(c; x^{(n)}, \Theta^{old})$ to make the dependencies on data point and parameters explicit (note that q_{new} is now the same function for all data points and parameters but with accrodingly different arguments). The E-step consists of computing $q_{\text{new}}(c; x^{(n)}, \Theta^{old})$ for all data points $x^{(n)}$ and all c (these are N times C values, and for $C = 2$ in total $2N$ floating point numbers between zero and one. The free energy depends of the $q_{\text{new}}(c; x^{(n)}, \Theta^{old})$ distributions of all data points, which is what we mean by writing q_{new} in the argument of $\mathcal{F}(q_{\text{new}}, \Theta)$, i.e.:

$$q_{\text{new}} = \left(q_{\text{new}}(c; x^{(1)}, \Theta^{old}), q_{\text{new}}(c; x^{(2)}, \Theta^{old}), \ldots, q_{\text{new}}(c; x^{(N)}, \Theta^{old})\right) \tag{101}$$

5.3 The M-step

For the M-step of the EM-algorithm, we have to solve the second maximization problem given by:

$$\Theta^{\text{new}} = \text{argmax}_\Theta\{\mathcal{F}(q_{\text{new}}, \Theta)\} \tag{102}$$

Now the q_{new} is held fixed and the Θ are allowed to vary. We exploit a necessary condition for a maximum, i.e., we demand the derivatives w.r.t. all parameters to be zero:

$$\frac{\partial}{\partial \Theta_i}\mathcal{F}(q_{\text{new}}, \Theta) \stackrel{!}{=} 0 \quad \text{for all parameters } \Theta_i. \tag{103}$$

Before we consider the derivatives w.r.t. the paramters, we have to insert the distribution q_{new} of Eqn. 100 which was computed in the E-step into the formula (98) for the GMM free energy. The free energy $\mathcal{F}(q_{\text{new}}, \Theta)$ is then given by:

$$\begin{aligned} \mathcal{F}(q_{\text{new}}, \Theta) = &\sum_n \sum_{c'} q_{\text{new}}(c'; x^{(n)}, \Theta^{old})(\frac{1}{2}\log(2\pi) - \frac{1}{2}\log(\sigma_{c'}^2) - \frac{1}{2}\frac{(x^{(n)} - \mu_{c'})^2}{\sigma_{c'}^2} + \log(\pi_{c'})) \\ &- \sum_n \sum_{c'} q_{\text{new}}(c'; x^{(n)}, \Theta^{old}) \log\left(q_{\text{new}}(c'; x^{(n)}, \Theta^{old})\right). \end{aligned}$$

Note that we (for later convenience) sum over c' instead of c.

The first derivative we consider is the derivative with respect to μ_c. Note that $q_{\text{new}}(c'; x^{(n)}, \Theta^{old})$ depends on the old parameters not the new ones. $q_{\text{new}}(c'; x^{(n)}, \Theta^{old})$ can consequently be treated as a constant factor. Furthermore, the structure of the free energy results in many of it terms being zero if we take the

derivative:

$$\begin{aligned}
\frac{\partial}{\partial \mu_c}\mathcal{F}(q_{\text{new}},\Theta) &= \sum_n \sum_{c'} q_{\text{new}}(c';x^{(n)},\Theta^{old})(-\frac{1}{2\sigma_{c'}^2}\frac{\partial}{\partial \mu_c}(x^{(n)}-\mu_{c'})^2) \\
&= \sum_n \sum_{c'} q_{\text{new}}(c';x^{(n)},\Theta^{old})(-\frac{1}{2\sigma_{c'}^2})(-1)2(x^{(n)}-\mu_{c'})\delta_{cc'} \\
&= \sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old})(\frac{1}{\sigma_c^2})(x^{(n)}-\mu_c) = 0 \\
\Rightarrow \sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old})x^{(n)} &- \mu_c \sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old}) = 0 \\
\Rightarrow \mu_c &= \frac{\sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old})x^{(n)}}{\sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old})}. \qquad (104)
\end{aligned}$$

During the first step, we only keep the terms on the r.h.s. which depend on μ_c. In the second line the summation over the index c' is performed. Note that in the third step the quantity σ_c^2 does not depend on the index n and so can be put outside the summation.

Instead of performing the derivative with respect to σ_c we take a shortcut by doing it with respect to $\tilde{\sigma}_c = \sigma_c^2$:

$$\begin{aligned}
\frac{\partial}{\partial \tilde{\sigma}_c}\mathcal{F}(q_{\text{new}},\Theta) &= \sum_n \sum_{c'} q_{\text{new}}(c';x^{(n)},\Theta^{old})(-\frac{1}{2}\frac{1}{\tilde{\sigma}_c}\delta_{cc'} - \frac{1}{2}(x^{(n)}-\mu_c)^2(-\frac{1}{\tilde{\sigma}_c^2})\delta_{cc'}) \\
&= \sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old})(-\frac{1}{2}\frac{1}{\tilde{\sigma}_c} + \frac{1}{2}(x^{(n)}-\mu_c)^2(\frac{1}{\tilde{\sigma}_c^2})) = 0 \\
\Rightarrow \sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old})((x^{(n)}-\mu_c)^2 &- \tilde{\sigma}_c) = 0 \\
\Rightarrow \sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old})(x^{(n)}-\mu_c)^2 &- \tilde{\sigma}_c \sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old}) = 0 \\
\Rightarrow \tilde{\sigma}_c &= \frac{\sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old})(x^{(n)}-\mu_c)^2}{\sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old})} \\
\Rightarrow \sigma_c^2 &= \frac{\sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old})(x^{(n)}-\mu_c)^2}{\sum_n q_{\text{new}}(c;x^{(n)},\Theta^{old})}
\end{aligned}$$

For $\frac{\partial}{\partial \pi_c}\mathcal{F}(q_{\text{new}},\Theta)$, we can not perform the derivative directly since the pi's are not independent of each other. Instead we have to perform an optimization of $\mathcal{F}(q_{\text{new}},\Theta)$ under the constraint that the following function is zero:

$$g(\vec{\pi}) = \sum_{c'} \pi_{c'} - 1 \stackrel{!}{=} 0, \qquad (105)$$

i.e., that the constraint $\sum_{c'} \pi_{c'} = 1$ is fulfilled. Following the methodology of constraint optimization with Lagrange multipliers (REFER OPTIONALLY TO APPENDIX), a necessary condition for $\mathcal{F}(q_{\text{new}},\Theta)$ to take on a maximum w.r.t. π_c is given by:

$$\frac{\partial}{\partial \pi_c}\mathcal{F}(q_{\text{new}},\Theta) + \lambda(\frac{\partial}{\partial \pi_c}g(\vec{\pi})) = 0, \qquad (106)$$

where λ is the Lagrange multiplier. Using the GMM free energy (...) we derive accordingly:

$$\begin{aligned}
\frac{\partial}{\partial \pi_c}\mathcal{F}(q_{\text{new}},\Theta) + \lambda(\frac{\partial}{\partial \pi_c}g(\vec{\pi})) &= \sum_n \sum_{c'} q_{\text{new}}(c'; x^{(n)}, \Theta^{old})\frac{1}{\pi_c}\delta_{cc'} + \lambda\frac{\partial}{\partial \pi_c}(\sum_{c'}\pi_{c'} - 1) \\
&= \sum_n q_{\text{new}}(c; x^{(n)}, \Theta^{old})\frac{1}{\pi_c} + \lambda = 0 \\
&\Rightarrow \quad \lambda\pi_c = -\sum_n q_{\text{new}}(c; x^{(n)}, \Theta^{old}) \qquad \forall\, c
\end{aligned}$$

To compute λ, we will add up all the equations above for all the different $c's$ in order to use the contraint. We arrive at:

$$\begin{aligned}
\lambda \underbrace{\sum_c \pi_c}_{1} &= -\sum_n \underbrace{\sum_c q_{\text{new}}(c; x^{(n)}, \Theta^{old})}_{1} \\
&\Rightarrow \quad \lambda = -N \\
&\Rightarrow \quad \pi_c = \frac{1}{N}\sum_n q_{\text{new}}(c; x^{(n)}, \Theta^{old}) \qquad (107)
\end{aligned}$$

The equations 104, 120 and 107 are the update rules for the one dimensional GMM. We can thus summarize the steps above as:

EM - Algorithm for the mixture of Gaussians model

- randomly initialize the parameters $\Theta = (\pi_1, ...\pi_c, \mu_1, ..., \mu_c, \tilde{\sigma}_1, ..\tilde{\sigma}_c)$
- compute the E-step:

$$q_{\text{new}}(c; x^{(n)}, \Theta^{old}) \;=\; p(c \,|\, x^{(n)}, \Theta^{old}) \quad = \quad \frac{(2\pi\tilde{\sigma}_c^{old})^{-\frac{1}{2}} \exp(-\frac{1}{2\tilde{\sigma}_c^{old}}\left(x^{(n)} - \mu_c^{old}\right)^2)\pi_c^{old}}{\sum_{c'}(2\pi\tilde{\sigma}_{c'}^{old})^{-\frac{1}{2}} \exp(-\frac{1}{2\tilde{\sigma}_{c'}^{old}}\left(x^{(n)} - \mu_{c'}^{old}\right)^2)\pi_{c'}^{old}}$$

  Often simply abbreviated by:

$$r_c^{(n)} \quad = \quad \frac{(2\pi\tilde{\sigma}_c^{old})^{-\frac{1}{2}} \exp(-\frac{1}{2\tilde{\sigma}_c^{old}}\left(x^{(n)} - \mu_c^{old}\right)^2)\pi_c^{old}}{\sum_{c'}(2\pi\tilde{\sigma}_{c'}^{old})^{-\frac{1}{2}} \exp(-\frac{1}{2\tilde{\sigma}_{c'}^{old}}\left(x^{(n)} - \mu_{c'}^{old}\right)^2)\pi_{c'}^{old}}$$

- compute the M-step:

$$\begin{aligned}
\pi_c^{new} &= \frac{1}{N}\sum_n q_{\text{new}}(c; x^{(n)}, \Theta^{old}) \\
\mu_c^{new} &= \frac{\sum_n q_{\text{new}}(c; x^{(n)}, \Theta^{old})x^{(n)}}{\sum_n q_{\text{new}}(c; x^{(n)}, \Theta^{old})} \\
\left(\sigma_c^{new}\right)^2 &= \frac{\sum_n q_{\text{new}}(c; x^{(n)}, \Theta^{old})(x^{(n)} - \mu_c^{new})^2}{\sum_n q_{\text{new}}(c; x^{(n)}, \Theta^{old})}
\end{aligned}$$

Or with abbreviation $r_c^{(n)}$

$$\begin{aligned}
\pi_c^{new} &= \frac{1}{N}\sum_n r_c^{(n)} \\
\mu_c^{new} &= \frac{\sum_n r_c^{(n)} x^{(n)}}{\sum_n r_c^{(n)}} \\
\big(\sigma_c^{new}\big)^2 &= \frac{\sum_n r_c^{(n)} (x^{(n)} - \mu_c^{new})^2}{\sum_n r_c^{(n)}}
\end{aligned}$$

- set $\Theta^{old} = \Theta^{new}$
- continue until converge

Remarks

- In Machine Learning such an algorithm is called a 'learning algorithm'.
- Rules to change parameters are often called 'learning rules' or 'parameter update rules'.
- Note that we can derive update rules (and thus algorithms) for much more general generative models.
- Models of the kind as above, i.e. models that take each data point to originate from one class and an associated distribution are called *mixtures models* (several individual distributions are mixed).
- A mixture model is a particular type of a hidden variable model.

Gaussian Mixture Models - The general case

Let us now consider the general case of Gaussian Mixture Models (GMMs). Considering the derivations above, we already notice that they apply for any number of clusters ('holes in the ceiling'). However, we need some non-trivial generalization if we go from scalar data x to higher dimensions $\vec{x} \in \mathbb{R}^D$. The scalar Gaussian we used as a noise model than has to be generalized to multiple-dimensions. Such a generalization is given by the multi-variate Gaussian density function:

$$\begin{aligned}
p(\vec{x}\,|\,c,\Theta) \;=\; \mathcal{N}(\vec{x};\vec{\mu}_c,\Sigma_c) &= \frac{1}{\sqrt{\det(2\pi\Sigma_c)}}\exp(-\frac{1}{2}(\vec{x}-\vec{\mu}_c)^T\Sigma_c^{-1}(\vec{x}-\vec{\mu}_c)) \\
&= \big(\det(2\pi\Sigma_c)\big)^{-\frac{1}{2}}\exp\big(-\frac{1}{2}(\vec{x}-\vec{\mu}_c)^T\Sigma_c^{-1}(\vec{x}-\vec{\mu}_c)\big) \qquad (108) \\
&\quad \text{where } \Sigma_c \text{ are symmetric matrices}
\end{aligned}$$

An example for $D = 2$ is:

$$\Sigma_c = \begin{pmatrix} \alpha_c & \beta_c \\ \beta_c & \gamma_c \end{pmatrix} \qquad (109)$$

Note that the shape of a cluster c corresponds to an ellipses in 2-D. If the matrix Σ_c is diagonal, the main axes of the ellipse are parallel to the coordinate axes. If the matrix Σ_c is proportional to the unit matrix, then the cluster has a circular shape. In the latter case (and similarly also in the diagonal case) we can also rewrite the multi-variate Gaussian as follows:

$$\Sigma_c = \begin{pmatrix} \sigma_c^2 & 0 \\ 0 & \sigma_c^2 \end{pmatrix} \qquad (110)$$

$$\Rightarrow \det\big(2\pi\Sigma_c\big) = (2\pi\sigma_c^2)\,(2\pi\sigma_c^2) \qquad (111)$$

$$\Rightarrow \big(\Sigma_c\big)^{-1} = \begin{pmatrix} \frac{1}{\sigma_c^2} & 0 \\ 0 & \frac{1}{\sigma_c^2} \end{pmatrix} \qquad (112)$$

$$
\begin{aligned}
p(\vec{x}\,|\,c,\Theta) &= \frac{1}{\sqrt{\det(2\pi\Sigma_c)}}\exp(-\frac{1}{2}(\vec{x}-\vec{\mu}_c)^T\Sigma_c^{-1}(\vec{x}-\vec{\mu}_c)) && (113)\\
&= \frac{1}{\sqrt{2\pi\sigma_c^2}\sqrt{2\pi\sigma_c^2}}\exp(-\sum_i\frac{1}{2}(x_i-\mu_{ci})\frac{1}{\sigma_c^2}(x_i-\mu_{ci})) && (114)\\
&= \frac{1}{\sqrt{2\pi\sigma_c^2}}\frac{1}{\sqrt{2\pi\sigma_c^2}}\prod_i\exp(-\frac{1}{2\sigma_c^2}(x_i-\mu_{ci})^2) && (115)\\
&= \prod_i\Big(\frac{1}{\sqrt{2\pi\sigma_c^2}}\exp\big(-\frac{1}{2\sigma_c^2}(x_i-\mu_{ci})^2\big)\Big) && (116)
\end{aligned}
$$

So this is just a product of two familiar scalar Gaussian densities – one for the x-coordinate and one for the y-coordinate. This special case stresses that the form (108) is a generalization of the standard Gaussian to higher dimensional spaces.

Having the tool of the multi-variate Gaussian density, we can now go back and model the data (e.g., blue-berries on a 2-D factory floor) using the following probabilistic generative model:

$$
\begin{aligned}
p(c\,|\,\theta) &= \pi_c \quad \text{where} \quad \sum_c \pi_c = 1\\
p(\vec{x}\,|\,c,\theta) &= \mathcal{N}(\vec{x};\vec{\mu}_c,\Sigma_c),
\end{aligned}
$$

where the latter Gaussian density is now the multi-variate Gaussian density function. For the higher-dimensional case, the E-step is (by applying Bayes' rule) performed in the same way as before but now using the multi-variate Gaussian (108):

$$
\begin{aligned}
r_c^{(n)} &= p(c\,|\,\vec{x}^{(n)},\Theta^{\text{old}})\\
&= \frac{\pi_c^{old}\mathcal{N}(\vec{x}^{(n)};\vec{\mu}_c^{old},\Sigma_c^{old})}{\sum_{c'}\pi_{c'}^{old}\mathcal{N}(\vec{x}^{(n)};\vec{\mu}_{c'}^{old},\Sigma_{c'}^{old})}\\
&= \frac{\pi_c^{old}\big(\det(2\pi\Sigma_c^{old})\big)^{-\frac{1}{2}}\exp\big(-\frac{1}{2}(\vec{x}-\vec{\mu}_c^{old})^T(\Sigma_c^{old})^{-1}(\vec{x}-\vec{\mu}_c^{old})\big)}{\sum_{c'}\pi_{c'}^{old}\big(\det(2\pi\Sigma_{c'}^{old})\big)^{-\frac{1}{2}}\exp\big(-\frac{1}{2}(\vec{x}-\vec{\mu}_{c'})^T(\Sigma_{c'}^{old})^{-1}(\vec{x}-\vec{\mu}_{c'}^{old})\big)}
\end{aligned} \qquad (117)
$$

The update equations for the parameters are derived in analogy to the one dimensional case (see exercises), and can be obtained as follows:

$$
\begin{aligned}
\pi_c^{new} &= \frac{1}{N}\sum_n r_c^{(n)}\\
\vec{\mu}_c^{new} &= \frac{\sum_n r_c^{(n)}\vec{x}^{(n)}}{\sum_n r_c^{(n)}}\\
\Sigma_c^{new} &= \frac{\sum_n r_c^{(n)}(\vec{x}^{(n)}-\vec{\mu}_c^{new})(\vec{x}^{(n)}-\vec{\mu}_c^{new})^T}{\sum_n r_c^{(n)}}
\end{aligned}
$$

Using an EM algorithm on the basis of these E- and M-steps realizes the classical GMM algorithm which is very widely used in Machine Learning, Statistics and beyond.

6 Multiple-Causes Models and Binary Sparse Coding

So far, we have considered what can be called *single cause* generative models, i.e., models with a latent variable whose value refers to one single cause. The example we used is a latent variable c which could take on one of two values ($c \in \{1, 2\}$). So a 'berry' would drop from either 'hole' $c = 1$ or $c = 2$. An example with, say, five 'holes' in the ceiling ($c \in \{1, \ldots, 5\}$) would, of course, also be a single cause model because for each 'berry' position there remains one single 'hole' that has generated the 'berry' position.
We consider again the following generative model:

$$\begin{aligned} p(c|\theta) &= \pi_c \\ p(\vec{y}\,|\,c, \theta) &= \mathcal{N}(\vec{y}; \vec{\mu}_c, \Sigma_c) \end{aligned}$$

Where $c \in 1, ..., C$ and $\sum_{c=1}^{C} \pi_c = 1$ and where:

$$p(\vec{y}\,|\,c, \theta) = \frac{1}{\sqrt{\det(2\pi\Sigma_c)}} \exp(-\frac{1}{2}(\vec{y} - \vec{\mu}_c)^T(\Sigma_c)^{-1}(\vec{y} - \vec{\mu}_c))$$

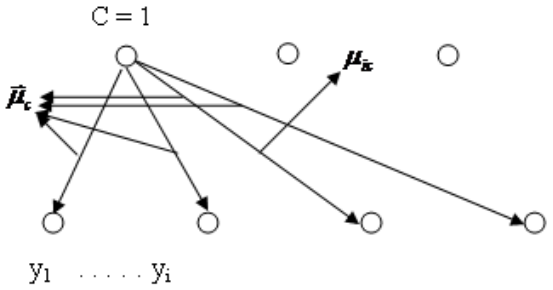


Figure 14: Graphical Model

Note that we can choose another noise model here such as Poisson, Beta or Bernoulli. In any case, the general form of a *graphical model* for the generative model above is depicted in figure 6. There is a hidden layer with one node for all the possible values c, and the observed layer below has one node per entry of the observable vector $\vec{y}$.
NOTE: ALL TEXT HAS TO BE REVISED HERE.

To understand why this generative model is associated with clustering, it is useful to take a look at the way in in which it generates data. Following the first equation above, we choose a c value from the set $\{1, ..., C\}$, according to the probabilities π_c, we call it c'. Afterwards, we draw a data point from the gaussian distribution with center $\mu_{c'}$ and variance $\Sigma_{c'}$. Following this procedure, we could generate at most C different clusters of data.

In this type of model, we have only one random variable asociated to it, this is why it is called single-cause model. What happens if we have more random variables?

- {Bars data is shown}.What would you (as humans) say about this data? What would a mixture model say?

6.1 Contrast: Multiple-causes model

Now, we consider as random variables a vector of H binary elements, for instance, each element only taking the values 0 or 1. We can assume for instance an independent Bernoulli distribution for each of the vector entries. We take for simplicity equal pi's $\pi_r = \pi$.

$$\begin{aligned} p(c \,|\, \Theta) &= \pi_c \\ p(\vec{y} \,|\, c, \Theta) &= \mathcal{N}(\vec{y}; \vec{\mu}_c, \sigma^2 \mathbf{1}) \end{aligned}$$

$$\begin{aligned} p(\vec{s} \,|\, \Theta) &= \vec{\pi}^T \vec{s} \qquad\qquad \vec{s} \in \{0,1\}^H, \quad |\vec{s}| = 1. \\ p(\vec{y} \,|\, \vec{s}, \Theta) &= \mathcal{N}(\vec{y}; W\vec{s}, \sigma^2 \mathbf{1}) \end{aligned}$$

or alternatively

$$\begin{aligned} p(\vec{s} \,|\, \Theta) &= \prod_{h=1}^{H} p(s_h \,|\, \Theta) = \prod_h \pi^{s_h} (1-\pi)^{1-s_h} \\ p(\vec{y} \,|\, \vec{s}, \Theta) &= \mathcal{N}(\vec{y}; W\vec{s}, \sigma^2 \mathbf{1}) \end{aligned}$$

$$\begin{aligned} p(\vec{s} \,|\, \Theta) &= \prod_{h=1}^{H} p(s_h \,|\, \Theta) = \prod_h \pi^{s_h} (1-\pi)^{1-s_r} \\ p(\vec{y} \,|\, \vec{s}, \Theta) &= \prod_{d=1}^{D} p(y_d \,|\, \vec{s}, \theta) = \prod_d \mathcal{N}(y_d; \underbrace{\sum_h W_{dh} s_h}_{(W\vec{s})_d}, \sigma^2) \end{aligned}$$

or alternatively

$$\begin{aligned} p(\vec{s} \,|\, \Theta) &= \prod_{h=1}^{H} p(s_h \,|\, \Theta) = \prod_h \pi^{s_h} (1-\pi)^{1-s_h} \\ p(\vec{y} \,|\, \vec{s}, \Theta) &= \mathcal{N}(\vec{y}; W\vec{s}, \sigma^2 \mathbf{1}) \end{aligned}$$

{Data of a multiple-causes model is shown}

Whenever we have data that is well described by the *binary sparse coding* model above, we can try to maximize the data likelihood under the this model.

We will go for the EM Method, as we saw in prevoius lectures.

E-step : We compute the posterior distribution over the hidden causes with the parameters values we have.

$$\begin{aligned} p(\vec{s}\,|\,\vec{y}^{(n)},\theta) &= \frac{p(\vec{y}^{(n)}\,|\,\vec{s},\theta)p(\vec{s}\,|\,\theta)}{\sum_{\vec{s}} p(\vec{y}^{(n)}\,|\,\vec{s}',\theta)p(\vec{s}'\,|\,\theta)} \\ &= \frac{\mathcal{N}(\vec{y}^{(n)};W\vec{s},\sigma^2\mathbf{1})(\prod_h \pi_h^{s_h}(1-\pi_h)^{1-s_h})}{\sum_{\vec{s}'}\mathcal{N}(\vec{y}^{(n)};W\vec{s}',\sigma^2\mathbf{1})(\prod_h \pi_h^{s'_h}(1-\pi_h)^{1-s'_h})} \end{aligned}$$

M-step : We update the parameter values that optimize the free energy function constructed with the distributions $q^{(n)}$ computed above. Note the double use of π. In the prefactor, π is the number *Pi*, whereas in the Bernoulli distribution it is a parameter in Θ.

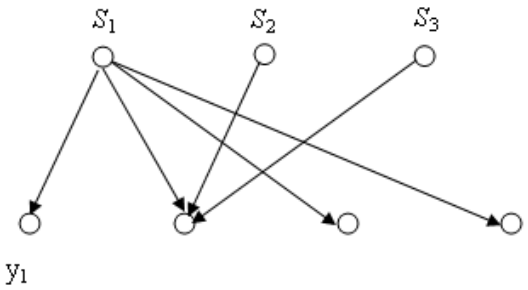


Figure 15: Graphical model for the multiple causes model mentioned in the text. Note that each s_i in the hidden layer refers to a random variable (each one taken the values 0 or 1)

$$\mathcal{F}(q,\theta) = \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) \Big(\log(p(\vec{y}^{(n)}|\vec{s},\theta)) + \log(p(\vec{s}\,|\,\theta)) \Big) - \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) \log\big(q^{(n)}(\vec{s})\big) \tag{118}$$

$$\begin{aligned}
\mathcal{F}(q,\theta) &= \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) \Big(-\frac{1}{2}\log(det(2\pi\sigma^2\mathbf{1})) - \frac{1}{2\sigma^2}(\vec{y}^{(n)} - W\vec{s})^T(\vec{y}^{(n)} - W\vec{s}) \\
&\quad + \sum_h (s_h \log(\pi) + (1-s_h)\log(1-\pi)) \Big) - \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) \log\big(q^{(n)}(\vec{s})\big) \\
\frac{d\mathcal{F}(q,\theta)}{d\pi} &= \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) \frac{d}{d\pi} \sum_h (s_h \log(\pi) + (1-s_h)\log(1-\pi)) \\
&= \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) \sum_h (s_h \frac{1}{\pi} - (1-s_h)\frac{1}{1-\pi}) \stackrel{!}{=} 0 \qquad \text{multiply by } \pi\,(1-\pi) \\
&\Rightarrow \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) \big(|\vec{s}|(1-\pi) - (H-|\vec{s}|)\pi\big) \stackrel{!}{=} 0 \qquad \text{abbreviate } |\vec{s}| = \textstyle\sum_h s_h \text{ and simplify} \\
&\Rightarrow \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) \big(|\vec{s}| - H\pi\big) \stackrel{!}{=} 0 \\
&\Rightarrow H\pi \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) = \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})|\vec{s}| \qquad \text{note that } \textstyle\sum_{\vec{s}} q^{(n)}(\vec{s}) = 1 \text{ for all } n \\
&\Rightarrow \pi = \frac{1}{NH} \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})|\vec{s}|
\end{aligned}$$

Or as update rule using expectation value notation:

$$\pi^{new} = \frac{1}{NH} \sum_n \langle |\vec{s}| \rangle_{q^{(n)}}$$

Note that in the same way we could have derived update equations for individual π_h values.

For the second parameter, the weight matrix W, we require some matrix derivation formulas:

$$\frac{d}{dW}(\vec{y}^T W \vec{s}) = \vec{y}\vec{s}^T; \frac{d}{dW}(\vec{s}^T W^T W \vec{s}) = W(\vec{s}\vec{s}^T + \vec{s}\vec{s}^T); \quad \frac{d}{dW}(b^T W^T W c) = W(bc^T + cb^T).$$

We can then take derivatives w.r.t. W:

(119)

$$
\begin{aligned}
\mathcal{F}(q,\theta) &= \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\big(-\frac{1}{2}\log(det(2\pi\sigma^2\mathbf{1})) - \frac{1}{2\sigma^2}(\vec{y}^{(n)} - W\vec{s})^T(\vec{y}^{(n)} - W\vec{s}) \\
&\quad + \sum_h (s_h \log(\pi) + (1-s_h)\log(1-\pi))\big) \; - \; \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) \log\left(q^{(n)}(\vec{s})\right) \\
\frac{d\mathcal{F}(q,\theta)}{dW} &= -\frac{1}{2\sigma^2}\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) \frac{d}{dW}((\vec{y}^{(n)} - W\vec{s})^T(\vec{y}^{(n)} - W\vec{s})) \\
&= -\frac{1}{2\sigma^2}\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) \frac{d}{dW}(\vec{y}^{(n)T}\vec{y}^{(n)} - 2\vec{y}^{(n)T}W\vec{s} + \vec{s}^T W^T W\vec{s}) \\
&= -\frac{1}{2\sigma^2}\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}))(-2\vec{y}^{(n)}\vec{s}^T + 2W(\vec{s}\vec{s}^T)) \\
&= -\frac{1}{2\sigma^2}\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})(\vec{y}^{(n)}\vec{s}^T - W(\vec{s}\vec{s}^T)) \stackrel{!}{=} 0 \\
&\Rightarrow \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\vec{y}^{(n)}\vec{s}^T - \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) W\vec{s}\vec{s}^T \stackrel{!}{=} 0 \\
&\Rightarrow \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\vec{y}^{(n)}\vec{s}^T = W(\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\vec{s}\vec{s}^T)
\end{aligned}
$$

$\sum_n \sum_{\vec{s}} p(\vec{s}|\vec{y}^{(n)}, \theta^{old})\vec{s}\vec{s}^T$ is squared, symmetric. Furthermore, there is at least one of the posteriors that is positive, (since they have to add up to one and are all greater or equal to zero) then the sum will give a nonzero matrix. We assume that the matrix is also invertible.

$$
\begin{aligned}
&\Rightarrow \quad W = (\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\vec{y}^{(n)}\vec{s}^T)(\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\; \vec{s}\vec{s}^T)^{-1} \\
&\Rightarrow \quad W = (\sum_n \vec{y}^{(n)} \sum_{\vec{s}} q^{(n)}(\vec{s})\vec{s}^T)(\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\vec{s}\vec{s}^T)^{-1} \\
&\textit{or} \\
&\qquad W^{new} = (\sum_n \vec{y}^{(n)}\langle\vec{s}\rangle_{q_n}^T)(\sum_n \langle\vec{s}\vec{s}^T\rangle_{q_n})^{-1} \\
&\qquad \texttt{with} \qquad \underbrace{\langle f(\vec{s})\rangle_{q_n}}_{\texttt{the expectation value}} = \sum_{\vec{s}} p(\vec{s}|\vec{y}^{(n)}, \theta^{old}) f(\vec{s})
\end{aligned}
$$

$\langle\vec{s}\rangle_{q_n}$ and $\langle\vec{s}\vec{s}^T\rangle_{q_n}$ i.e., the first and the second moments of the posterior distribution, is all we have to know for updating W. $\langle\vec{s}\rangle_{q_n}$ and $\langle\vec{s}\vec{s}^T\rangle_{q_n}$ are therefore called the sufficient statistics of this model.

Finally, the update for σ can be derived. We can do so in analogy to the derivation for σ_c of a GMM.

$$\begin{aligned}
\mathcal{F}(q,\theta) &= \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\big(-\frac{1}{2}\log(det(2\pi\sigma^2\mathbf{1})) - \frac{1}{2\sigma^2}(\vec{y}^{(n)} - W\vec{s})^T(\vec{y}^{(n)} - W\vec{s}) \\
&\quad + \sum_h (s_h \log(\pi) + (1-s_h)\log(1-\pi))\big) \; - \; \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) \log\big(q^{(n)}(\vec{s})\big) \\
\frac{\partial}{\partial\tilde{\sigma}}\mathcal{F}(q,\Theta) &= \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\big(-\frac{1}{2}\frac{\partial}{\partial\tilde{\sigma}}\log(det(2\pi\tilde{\sigma}\mathbf{1})) - \frac{1}{2}\frac{\partial}{\partial\tilde{\sigma}}\frac{1}{\tilde{\sigma}}(\vec{y}^{(n)} - W\vec{s})^T(\vec{y}^{(n)} - W\vec{s})\big) \\
&= \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\big(-\frac{1}{2}\frac{\partial}{\partial\tilde{\sigma}}\big(D\log(2\pi) + D\log(\tilde{\sigma})\big) \\
&\qquad\qquad - \frac{1}{2}\frac{\partial}{\partial\tilde{\sigma}}\frac{1}{\tilde{\sigma}}(\vec{y}^{(n)} - W\vec{s})^T(\vec{y}^{(n)} - W\vec{s})\big) \\
&= \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\big(-\frac{1}{2}D\frac{1}{\tilde{\sigma}} + \frac{1}{2}\frac{1}{\tilde{\sigma}^2}(\vec{y}^{(n)} - W\vec{s})^T(\vec{y}^{(n)} - W\vec{s})\big) \stackrel{!}{=} 0 \qquad \text{multiply by } -2\tilde{\sigma}^2 \\
&\Rightarrow \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\big(D\tilde{\sigma} - (\vec{y}^{(n)} - W\vec{s})^T(\vec{y}^{(n)} - W\vec{s})\big) \stackrel{!}{=} 0 \\
&\Rightarrow D\tilde{\sigma}\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s}) = \sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})(\vec{y}^{(n)} - W\vec{s})^T(\vec{y}^{(n)} - W\vec{s}) \qquad \text{with } \textstyle\sum_{\vec{s}} q^{(n)}(\vec{s}) = 1 \\
&\Rightarrow \tilde{\sigma} = \frac{1}{ND}\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})(\vec{y}^{(n)} - W\vec{s})^T(\vec{y}^{(n)} - W\vec{s}) \\
&\Rightarrow \sigma^2 = \frac{1}{ND}\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})(\vec{y}^{(n)} - W\vec{s})^T(\vec{y}^{(n)} - W\vec{s})
\end{aligned}$$

So we obtain (now using the elaborate W^{new} notation):

$$\sigma^2_{new} = \frac{1}{ND}\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})(\vec{y}^{(n)} - W^{new}\vec{s})^T(\vec{y}^{(n)} - W^{new}\vec{s})$$

Or in terms of expectation values:

$$\sigma^2_{new} = \frac{1}{ND}\sum_n \langle \|\vec{y}^{(n)} - W^{new}\vec{s})\|^2 \rangle_{q^{(n)}}$$

There is a possibility to rewrite the update rule for σ^2 slightly. We will use the following rule for traces to rewrite:

$$\vec{a}^T W \vec{b} \; = \; \mathrm{Tr}\big(\vec{a}^T W \vec{b}\big) \quad = \quad \mathrm{Tr}\big(W \vec{b}\,\vec{a}^T\big). \tag{120}$$

The first equality holds as the trace of a scalar (a one-by-one matrix) is the scalar itself. The second equation holds because of the permutation property of traces.

We can then rewrite:

$$\begin{aligned}
\sigma^2_{new} &= \frac{1}{ND}\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\big(\vec{y}^{(n)} - W_{new}\vec{s}\big)^T(\vec{y}^{(n)} - W_{new}\vec{s}\big) \\
&= \frac{1}{ND}\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\Big(\big(\vec{y}^{(n)}\big)^T\vec{y}^{(n)} \; + \; \vec{s}^T W_{new}^T W_{new}\,\vec{s} \; - \; 2\,\big(\vec{y}^{(n)}\big)^T W_{new}\,\vec{s}\Big) \\
&= \frac{1}{ND}\sum_n \sum_{\vec{s}} q^{(n)}(\vec{s})\Big(\big(\vec{y}^{(n)}\big)^T\vec{y}^{(n)} \; + \; \mathrm{Tr}\big(W_{new}^T W_{new}\,\vec{s}\vec{s}^T\big) \; - \; 2\,\big(\vec{y}^{(n)}\big)^T W_{new}\,\vec{s}\Big) \\
&= \frac{1}{ND}\sum_n \Big(\big(\vec{y}^{(n)}\big)^T\vec{y}^{(n)} \; + \; \mathrm{Tr}\big(W_{new}^T W_{new}\,\langle\vec{s}\vec{s}^T\rangle_{q^{(n)}}\big) \; - \; 2\,\big(\vec{y}^{(n)}\big)^T W_{new}\,\langle\vec{s}\rangle_{q^{(n)}}\Big)
\end{aligned}$$

{BSC simulations are shown here}

Note that in practice we have to do more, problems that may arise are: Local optima and computational complexity

- Local optima: Principle problem because EM is a gradient method
  Counter measure: annealing
- Computational complexity: Sums over all states $\vec{s}$ are large (it grows exponentially as 2^N).This exponential grow is a typical feature of multiple causes models
  Counter measure: Approximate schemes, i.e., approximate the sufficient statistics by approximations that be computed more efficiently.

Note1: also the likelihood computation becomes intractable
Note2: we can also use another prior than a Bernoulli prior

Applications of BSC

- - component extraction (what does the data consist of ?)
- - denoising (e.g. enhancement of image quality)
- - data compression (store W once and just store the $\vec{s}$ afterwards)

Similar Models: NMF (non-negative matrix factorization) : BSC with $W_{dr} \geq 0$ is a form of NMF (Lee and Seang, Science 1999)
NMF is a widely used component extraction method.

7 Probabilistic PCA

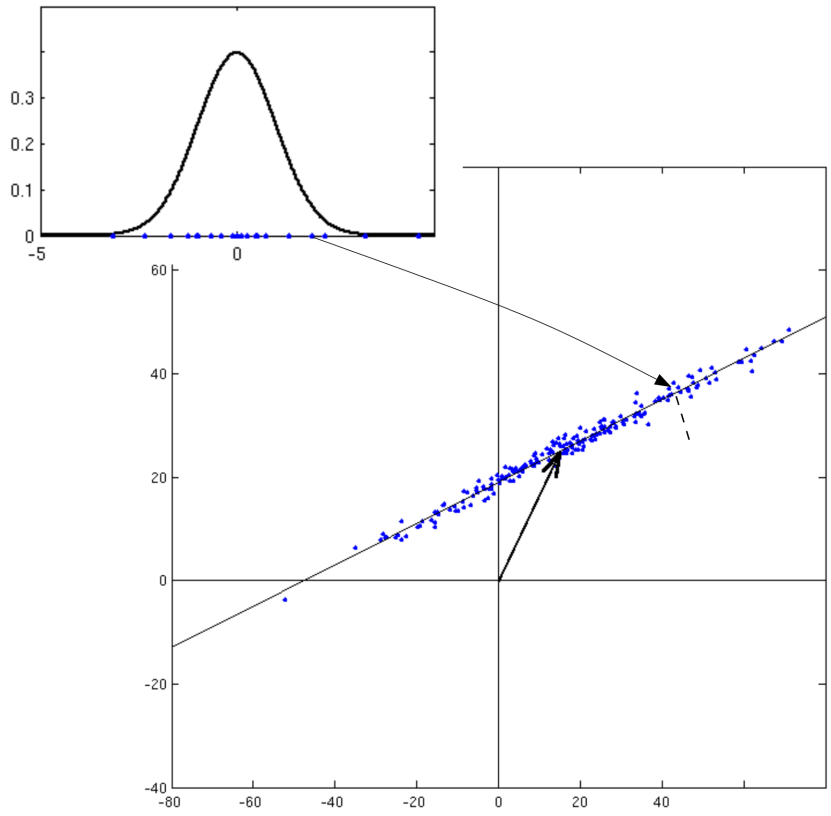


Figure 16: Illustration of probabilistic PCA with one latent dimension ($H = 1$) and two observed dimensions $D = 2$.

Let us be back in a factory. This time our goal is to produce blueberry jelly. We let the blueberries drop from somewhere above and they get stuck in a jelly that has been placed on the floor with some slope (see Fig. 16 for a 1-D factory floor). We now want to find the 'best' plane (or line in 1-D) through the jelly where most blueberries are. If we have such a plane, we could, e.g., cut along this line to get some nice jelly with dense and relatively equally distributed berries.

Again, we are just provided with the coordinates of all the berries. How do we find the best possible plane (or line in 1-D)?

Let us start with a 2-D plane in 3-D space (so indeed the situation which would occur in the blueberry factory). Following our previous strategy, we would seek to mathematically model the data distribution as we observe it. For the 2-D plane, we could describe a 2-D plane in a 3-D space as follows:

$$\vec{x} \quad = \quad \vec{w}_1\, z_1 \,+\, \vec{w}_2\, z_2 \,+\, \vec{\mu}, \quad \text{where} \quad \vec{w}_1, \vec{w}_2, \vec{\mu} \in \mathbb{R}^3. \tag{121}$$

For any values of the scalars z_1 and z_2, the vector $\vec{x}$ would lie on the same 2-D plane. $\vec{w}_1$ and $\vec{w}_2$ define the coordinate axes that are not necessarily orthogonal (but we do assume $\vec{w}_1$ and $\vec{w}_2$ to be linearly independent). The vector $\vec{\mu}$ defines the origin of the coordinate axes.

We have modeled a plane but we have not modeled a stochastic distribution of berries close to the plane (the berries never end up precisely on the plane). We first need a model for how the berries are distributed across the plane. We can define such a model letting the variables z_1 and z_2 which describe

the coordinates within the plane be chosen randomly according to a specific 2-D distribution:

$$\vec{x} \;=\; \vec{w}_1\, z_1 \,+\, \vec{w}_2\, z_2 \,+\, \vec{\mu}, \quad \text{where} \quad z_1, z_2 \sim \mathcal{N}(\vec{z}; 0, 1) \tag{122}$$

$$\Rightarrow \vec{x} \;=\; W\,\vec{z} \,+\, \vec{\mu}, \quad \text{where} \quad \vec{z} \sim \mathcal{N}(\vec{z}; \vec{0}, \mathbb{1}) \quad \text{with} \quad W = (\vec{w}_1, \vec{w}_2)\,. \tag{123}$$

The above data model would generate 'blueberry' positions that are *precisely* on the plane defined by $\vec{w}_1, \vec{w}_2, \vec{\mu}$.

In reality, the blueberries will only lie close to such a plane. Consequently, we do have to add some noise which describes how the berry positions divert from the positions on the plane. In the above notation, we can do so by simply adding additional, 3-D Gaussian noise:

$$\vec{x} \;=\; W\,\vec{z} \,+\, \vec{\mu} \,+\, \vec{\epsilon}, \quad \text{where} \quad \vec{z} \sim \mathcal{N}(\vec{z}; \vec{0}, \mathbb{1}) \quad \text{and} \quad \vec{\epsilon} \sim \mathcal{N}(\vec{0}, \sigma^2 \mathbb{1}). \tag{124}$$

If we now randomly generate data vectors $\vec{x}$ according to Eqn. 124, we indeed get a blueberry distribution in the form we would expect from occurring after the blueberries dropped down into the jelly. Hence, Eqn. 124 describes a generative model.

Considering Eqn. 124, the random vector $\vec{x}$ can actually be taken as following a Gaussian distribution given a value for variable $\vec{z}$. So we can rewrite the generative model to take a more familiar form:

$$p(\vec{z}\,|\,\Theta) \;=\; \mathcal{N}(\vec{z}; \vec{0}, \mathbb{1}) \tag{125}$$

$$p(\vec{x}\,|\,\vec{z}, \Theta) \;=\; \mathcal{N}(\vec{x}; W\vec{z} + \vec{\mu}, \sigma^2\,\mathbb{1}) \tag{126}$$

We will call this generative model the *PCA generative model* (for reasons that will only become clear later).

More generally, $\vec{z}$ can be any vector ($\vec{z}$) in a H dimensional (Euclidean) latent space, and $\vec{x}$ can be any vector in a D dimensional (Euclidean) observed space. For $H = 1$ and $D = 2$ we come back to the situation depicted in Fig.... In that case W consists of just one vector which parameterizes a line in 2-D. In the following derivations we will maintain the matrix notation for W.

Finding the 'best' plane now translates into finding the best parameters W and $\vec{\mu}$ of the generative model (125) and (126) for a given set of N blueberry positions $\{\vec{x}^{(1)}, \ldots, \vec{x}^{(N)}\}$. Indeed, the full set of parameters would also contain σ^2, i.e., $\Theta = (W, \vec{\mu}, \sigma^2)$.

In order to find these parameters, we can now follow the standard procedure for generative models which seeks to maximize the log-likelihood of the given data:

$$\mathcal{L}(\Theta) \;=\; \sum_{n=1}^{N} \log p(\vec{x}^{(n)}\,|\,\Theta) \tag{127}$$

7.1 EM for the PCA generative model: Abstract Procedure

Having a generative model for the described problem in a form we are familiar to, i.e. Eqns. 125 and 126, we can now apply the standard EM procedure again in order to find maximum likelihood solutions for the parameters $\Theta = (W, \sigma^2, \vec{\mu})$. The EM procedure increases the variational lower bound $\mathcal{F}(q, \Theta)$ (a.k.a., free energy or ELBO) which, if maximized, is equal to the likelihood $\mathcal{L}(\Theta)$.

The EM procedure works as follows:

(i) initialize model parameters Θ

(ii) maximize $\mathcal{F}(q, \Theta)$ w.r.t. the q (E-step)

(iii) maximize $\mathcal{F}(q, \Theta)$ w.r.t. the Θ (M-step)

(iv) go to (ii) unless the parameters Θ have converged

In Sec. 5 this general procedure was made more explicit and we obtained:

(i) initialize model parameters Θ

(ii) compute the posterior probabilities $p(c \,|\, \vec{x}^{(n)}, \Theta)$ (E-step)

(iii) choose the new parameters Θ such that $\frac{\partial}{\partial \Theta_i} \mathcal{F}(\Theta) = 0$ for all i (M-step)

(iv) go to (ii) unless the parameters have converged

Step (iii) gives us the *parameter update rules* or *learning rules* for a specific generative model. For the GMM generative model, the free energy was given by:

$$\mathcal{F}(q, \Theta) \;=\; \sum_n \sum_c \; q^{(n)}(c) \, \Big(\log \big(p(\vec{x}^{(n)} \,|\, c, \Theta) \big) + \log \big(p(c \,|\, \Theta) \big) \Big) \;-\; \sum_n \sum_c q^{(n)}(c) \log \big(q^{(n)}(c) \big) \,,$$

where c was a *discrete* hidden variable ($c = 1, 2, \ldots, C$). In the case of the PCA generative model we have a *continuous* hidden variable $\vec{z}$ instead of a finite integer c. The derivation of the EM optimization procedure can proceed along the same line, however, if the sums over c are replaced by integrals over $\vec{z}$. The corresponding free energy for a continuous latent variable $\vec{z}$ is consequently given by:

$$\mathcal{F}(q, \Theta) \;=\; \sum_n \int q^{(n)}(\vec{z}) \, \Big(\log \big(p(\vec{x}^{(n)} \,|\, \vec{z}, \Theta) \big) + \log \big(p(\vec{z} \,|\, \Theta) \big) \Big) d\vec{z} \;-\; \sum_n \int q^{(n)}(\vec{z}) \log \big(q^{(n)}(\vec{z}) \big) d\vec{z} \,.$$

7.2 Derivation of E- and M-steps

To derive the concrete EM algorithm for the PCA generative model we now proceed as for the mixture of Gaussian generative model: firstly, we derive formulas for the E-step and, secondly, we derive the M-step update rules.

The E-step. Using Bayes' rule the posterior probability $p(\vec{z} \,|\, \vec{x}^{(n)}, \Theta)$ is given by:

$$p(\vec{z} \,|\, \vec{x}^{(n)}, \Theta) \quad = \quad \frac{p(\vec{x}^{(n)} \,|\, \vec{z}, \Theta) \; p(\vec{z} \,|\, \Theta)}{\int p(\vec{x}^{(n)} \,|\, \vec{z}, \Theta) \; p(\vec{z}' \,|\, \Theta) \; d\vec{z}'} \tag{128}$$

$$= \quad \frac{\mathcal{N}(\vec{x}^{(n)} \,|\, W\vec{z} + \vec{\mu}, \sigma^2 \mathbb{1}) \; \mathcal{N}(\vec{z} \,|\, 0, 1)}{\int \mathcal{N}(\vec{x}^{(n)} \,|\, W\vec{z}' + \vec{\mu}, \sigma^2 \mathbb{1}) \; \mathcal{N}(\vec{z}' \,|\, 0, 1) \; d\vec{z}'} \tag{129}$$

Note that all parameters have to be thought of as being the 'old' parameters. We have for better readability dropped the superscript 'old', however. For the last step we simply plugged-in the PCA generative model given in (125) and (126). For the GMM generative model we were already done at this point. For the continuous PCA model we are, however, faced with an integral in the denominator that makes things more difficult. But if we have a closer look, we can observe that the integrand in the denominator can be rewritten as follows (see lecture/exercises):

$$\mathcal{N}(\vec{x}^{(n)} \,|\, W\vec{z}' + \vec{\mu}, \sigma^2 \mathbb{1}) \; \mathcal{N}(\vec{z}' \,|\, 0, 1) \quad = \quad \mathcal{N}(\vec{x}^{(n)} \,|\, \vec{\mu}, C) \; \mathcal{N}(\vec{z}' \,|\, \mu_z(\vec{x}^{(n)}, \Theta), \sigma_z(\vec{x}^{(n)}, \Theta)).$$

This might not seem like a great advancement but note that we have rewritten the integrand such that the first distribution on the right-hand-side does not depend on $\vec{z}'$ anymore. We can therefore move it out of the integral which becomes:

$$\int \mathcal{N}(\vec{x}^{(n)} \,|\, W\vec{z}' + \vec{\mu}, \sigma^2 \mathbb{1}) \; \mathcal{N}(\vec{z}' \,|\, 0, 1) \; d\vec{z}' \tag{130}$$

$$= \quad \int \mathcal{N}(\vec{x}^{(n)} \,|\, \vec{\mu}, C) \; \mathcal{N}(\vec{z} \,|\, \mu_z(\vec{x}^{(n)}), \sigma_z(\vec{x}^{(n)})) \; d\vec{z}' \tag{131}$$

$$= \quad \mathcal{N}(\vec{x}^{(n)} \,|\, \vec{\mu}, C) \; \int \mathcal{N}(\vec{z} \,|\, \mu_z(\vec{x}^{(n)}), \sigma_z(\vec{x}^{(n)})) \; d\vec{z}' \tag{132}$$

$$= \quad \mathcal{N}(\vec{x}^{(n)} \,|\, \vec{\mu}, C) \,, \tag{133}$$

where C is a function of the parameters Θ, $C = WW^T + \sigma^2 \mathbb{1}$ (see Exercises for corresponding Gaussian identity). Using the result above we get:

$$p(\vec{z}\,|\,\vec{x}^{(n)}, \Theta^{\text{old}}) = \frac{\mathcal{N}(\vec{x}^{(n)}\,|\,W\vec{z}+\vec{\mu}, \sigma^2\mathbb{1})\,\mathcal{N}(\vec{z}\,|\,\vec{0}, \mathbb{1})}{\int \mathcal{N}(\vec{x}^{(n)}\,|\,W\vec{z}'+\vec{\mu}, \sigma^2\mathbb{1})\,\mathcal{N}(\vec{z}'\,|\,0, 1)\,d\vec{z}'} \tag{134}$$

$$= \frac{\mathcal{N}(\vec{x}^{(n)}\,|\,W\vec{z}+\vec{\mu}, \sigma^2\mathbb{1})\,\mathcal{N}(\vec{z}\,|\,\vec{0}, \mathbb{1})}{\mathcal{N}(\vec{x}^{(n)}\,|\,\vec{\mu}, C)} \tag{135}$$

We can further simplify this expression by noticing that not only the product of Gaussian density functions is a Gaussian density function but also the division of to Gaussian densities (if we know that it vanishes for plus/minus infinity): The log of the right-hand-side has to be a polynomial of order two in x_d and z_h (with mixted x_d-z_h-terms with different d and h). The polynomial can therefore be written as a polynomial for z_h with prefactors depending on $\vec{x}$. Furthermore, the right-hand-side has to vanish for any z_h values going to plus or minus infinity because this applies for the left-hand-side (as $p(\vec{z}\,|\,\vec{x}^{(n)}, \Theta^{\text{old}})$ is a probability distribution). The polynomial therefore has negative prefactors for the quadratic terms. Exponentiating such a polynomial results in a density in the form of a Gaussian density. We can consequently conclude:

Thus we get:

$$p(\vec{z}\,|\,\vec{x}^{(n)}, \Theta^{\text{old}}) = \mathcal{N}(\vec{z}\,|\,\mu_z(\vec{x}^{(n)}), \Sigma_z^{(n)}). \tag{136}$$

The posterior probability distribution is thus a Gaussian distribution with mean $\vec{\mu}_z^{(n)}$ and covariance matrix $\Sigma_z^{(n)}$ (as a remark: in this case it turns out the actually $\Sigma_z^{(n)}$ does not depend on n).

Mean and covariance matrix give us directly the first two moments. The first moment $\langle\vec{z}\rangle_{q^{(n)}}$ is nothing else than the mean; the second moment $\left\langle\vec{z}\vec{z}^T\right\rangle_{q^{(n)}}$ is a function of the mean and the covariance (because $\text{Cov}(q^{(n)}(\vec{z})) = \left\langle\vec{z}\vec{z}^T\right\rangle_{q^{(n)}} - \langle\vec{z}\rangle_{q^{(n)}}\langle\vec{z}\rangle_{q^{(n)}}^T$). Using Gaussian identities to derive expressions for mean and variance (see Exercises), the moments are given as follows:

$$\langle\vec{z}\rangle_{q^{(n)}} = \mu_z(\vec{x}^{(n)}) = M^{-1}W^T(\vec{x}^{(n)} - \vec{\mu}) \tag{137}$$

$$\left\langle\vec{z}\vec{z}^T\right\rangle_{q^{(n)}} = \Sigma_z^{(n)} + (\mu_z(\vec{x}^{(n)}))^2 = \sigma^2 M^{-1} + \langle\vec{z}\rangle_{q^{(n)}}\langle\vec{z}\rangle_{q^{(n)}}^T \tag{138}$$

where $M = W^TW + \sigma^2\mathbb{1}$. Expressions (139) and (140) will be used later on.

$$\langle\vec{z}\rangle_{q^{(n)}} = M^{-1}W_{\text{old}}^T(\vec{x}^{(n)} - \vec{\mu}) \tag{139}$$

$$\left\langle\vec{z}\vec{z}^T\right\rangle_{q^{(n)}} = \sigma_{\text{old}}^2 M^{-1} + \langle\vec{z}\rangle_{q^{(n)}}\langle\vec{z}\rangle_{q^{(n)}}^T \tag{140}$$

$$W_{\text{old}} = W_{\text{new}}\ \sigma_{\text{old}}^2 = \sigma_{\text{new}}^2 \tag{141}$$

where $M = W_{\text{old}}^T W_{\text{old}} + \sigma_{\text{old}}^2\mathbb{1}$.

The M-step After we have simplified the computation of the posterior probabilities in the E-step let us now turn to the M-step. The M-step is about finding the parameters Θ which maximize the variational lower bound $\mathcal{F}(q, \Theta)$ while the distributions $q^{(n)}$ are held fixed. For the derivatives w.r.t. the parameters, let us first write down the free energy for the PCA generative model explicitly. Using the definition of the PCA generative model (125) and (126), and inserting into the general expression for the free energy

(180) we obtain:

$$\begin{aligned}
\mathcal{F}(q,\theta) &= \sum_n \int_{\vec{z}} q^{(n)}(\vec{z})\big(-\frac{1}{2}\log(det(2\pi\sigma^2\mathbf{1})) - \frac{1}{2\sigma^2}(\vec{x}^{(n)} - W\vec{z} - \vec{\mu})^T(\vec{x}^{(n)} - W\vec{z} - \vec{\mu}) \\
&\quad -\frac{H}{2}\log(2\pi) - \frac{1}{2}\vec{z}^T\vec{z}\big)d\vec{z} - \sum_n \int q^{(n)}(\vec{z})\log\big(q^{(n)}(\vec{z})\big)d\vec{z} \\
&= \sum_n \int_{\vec{z}} q^{(n)}(\vec{z})\big(-\frac{1}{2}\log(det(2\pi\sigma^2\mathbf{1})) - \frac{1}{2\sigma^2}(\tilde{\vec{x}}^{(n)} - W\vec{z})^T(\tilde{\vec{x}}^{(n)} - W\vec{z}) \\
&\quad -\frac{H}{2}\log(2\pi) - \frac{1}{2}\vec{z}^T\vec{z}\big)d\vec{z} - \sum_n \int q^{(n)}(\vec{z})\log\big(q^{(n)}(\vec{z})\big)d\vec{z}
\end{aligned}$$

where we have introduced $\tilde{\vec{x}}^{(n)} = \vec{x}^{(n)} - \vec{\mu}$ as an abbreviation.

Derivative w.r.t. W. Let us start investigating the maximum of $\mathcal{F}(q,\Theta)$ w.r.t. the matrix W. We derive the derivative of $\mathcal{F}(q,\Theta)$ w.r.t. W which has to be zero at an optimum:

$$\begin{aligned}
\frac{d\mathcal{F}(q,\theta)}{dW} &= -\frac{1}{2\sigma^2}\sum_n \int q^{(n)}(\vec{z})\frac{d}{dW}((\tilde{\vec{x}}^{(n)} - W\vec{z})^T(\tilde{\vec{x}}^{(n)} - W\vec{z}))\,d\vec{z} \\
&= -\frac{1}{2\sigma^2}\sum_n \int q^{(n)}(\vec{z})\frac{d}{dW}(\tilde{\vec{x}}^{(n)T}\tilde{\vec{x}}^{(n)} - 2\tilde{\vec{x}}^{(n)T}W\vec{z} + \vec{z}^TW^TW\vec{z})\,d\vec{z} \\
&= -\frac{1}{2\sigma^2}\sum_n \int q^{(n)}(\vec{z}))(-2\tilde{\vec{x}}^{(n)}\vec{z}^T + 2W(\vec{z}\vec{z}^T))\,d\vec{z} \\
&= -\frac{1}{2\sigma^2}\sum_n \int q^{(n)}(\vec{z})(\tilde{\vec{x}}^{(n)}\vec{z}^T - W(\vec{z}\vec{z}^T))\,d\vec{z} \stackrel{!}{=} 0 \\
&\Rightarrow \sum_n \int q^{(n)}(\vec{z})\tilde{\vec{x}}^{(n)}\vec{z}^T\,d\vec{z} - \sum_n \int q^{(n)}(\vec{z})W\vec{z}\vec{z}^T\,d\vec{z} \stackrel{!}{=} 0 \\
&\Rightarrow \sum_n \int q^{(n)}(\vec{z})\tilde{\vec{x}}^{(n)}\vec{z}^T\,d\vec{z} = W(\sum_n \int q^{(n)}(\vec{z})\vec{z}\vec{z}^T)\,d\vec{z}
\end{aligned}$$

$\sum_n \int p(\vec{z}\,|\,\tilde{\vec{x}}^{(n)},\theta^{old})\vec{z}\vec{z}^T$ is a squared and symmetric matrix which we will also assume to be invertible. We then obtain:

$$\begin{aligned}
&\Rightarrow \quad W = (\sum_n \int q^{(n)}(\vec{z})\tilde{\vec{x}}^{(n)}\vec{z}^T\,d\vec{z})(\sum_n \int q^{(n)}(\vec{z})\,\vec{z}\vec{z}^T\,d\vec{z})^{-1} \\
&\Rightarrow \quad W = (\sum_n \tilde{\vec{x}}^{(n)} \int q^{(n)}(\vec{z})\vec{z}^T\,d\vec{z})(\sum_n \int q^{(n)}(\vec{z})\vec{z}\vec{z}^T\,d\vec{z})^{-1} \\
&\textit{or} \\
&\quad W^{\text{new}} = (\sum_n \tilde{\vec{x}}^{(n)}\langle\vec{z}\rangle_{q_n}^T)\,(\sum_n \langle\vec{z}\vec{z}^T\rangle_{q_n})^{-1} \quad \text{with} \quad \underbrace{\langle f(\vec{z})\rangle_{q_n}}_{\text{expectation value}} = \int q^{(n)}(\vec{z})f(\vec{z})\,d\vec{z}
\end{aligned}$$

Reinserting $\tilde{\vec{x}}^{(n)} = \vec{x}^{(n)} - \vec{\mu}$ gives:

$$W^{\text{new}} = (\sum_n (\vec{x}^{(n)} - \vec{\mu})\langle\vec{z}\rangle_{q_n}^T)\,(\sum_n \langle\vec{z}\vec{z}^T\rangle_{q_n})^{-1} \tag{142}$$

$\langle\vec{z}\rangle_{q_n}$ and $\langle\vec{z}\vec{z}^T\rangle_{q_n}$, i.e. the first and the second moments of the posterior distribution, are all we have to know for updating W. $\langle\vec{z}\rangle_{q_n}$ and $\langle\vec{z}\vec{z}^T\rangle_{q_n}$ are therefore sometimes referred to as *sufficient statistics* of the model.

One may at first think that we now do have to do further analytical (or numerical steps) in order to compute the integrals $\langle \vec{z} \rangle_{q_n}$ and $\langle \vec{z}\vec{z}^T \rangle_{q_n}$ for the W update. After all, we computed only the distribution $q^{(n)}(\vec{z})$ in the E-step. However, as the distribution $q^{(n)}(\vec{z})$ turned out to be a Gaussian with mean and variances that we can derive, we have already seen that we can obtain closed-form formulas for the first and second moments of $q^{(n)}(\vec{z})$. Considering update (142), we realize, that the update rule is given only in terms of these two moments. We can thus use equations (**??**) and (**??**) in order to update W. As no further information than the first two moments is required, the $\langle \vec{z} \rangle_{q_n}$ and $\langle \vec{z}\vec{z}^T \rangle_{q_n}$ are sometimes referred to as *sufficient statistics* of the model.

Note that much of the derivation for W is very similar to the derivation w.r.t. W of the BSC model. The reason is the same Gaussian noise model. So for all Gaussian noise models, we will encounter similar derivations.

Derivative w.r.t. σ^2

With the same argumentation as above, the update rule for the variance parameter σ^2 is similar to the derivation (and the result) w.r.t. the same variance parameter for the BSC model. We therefore state without derivation the result, which is given by: W

$$\sigma^2_{\text{new}} \;=\; \frac{1}{ND}\sum_{n=1}^{N}\Big((\vec{x}^{(n)}-\vec{\mu})^T(\vec{x}^{(n)}-\vec{\mu}) \;+\; \text{Tr}\big(W^T_{\text{new}}W_{\text{new}}\,\langle \vec{z}\vec{z}^T\rangle_{q^{(n)}}\big) - 2\,(\vec{x}^{(n)}-\vec{\mu})^T\,W_{\text{new}}\,\langle \vec{z}\rangle_{q^{(n)}}\Big) \tag{143}$$

Derivative w.r.t. $\vec{\mu}$

The derivative w.r.t. turns out to be easier if we do not follow the standard procedure. That is, it is easier (in this case) to derive the maximum likelihood solution for $\vec{\mu}$ directly using the log-likelihood (not the free energy). It will turn out to be the mean value of the data points:

$$\vec{\mu} \;=\; \frac{1}{N}\sum_{n=1}^{N}\vec{x}^{(n)} \tag{144}$$

If we use this expression for $\vec{\mu}$ in the update equations for W and σ^2 we always increase the data likelihood under the generative model (125) and (126) (or we hold it constant). Note that we have derived the update equations for the condition $\frac{\partial}{\partial\Theta_i}Q(\Theta)=0$, which is a necessary condition for a local optimum of $Q(\Theta)$ (and thus $\mathcal{F}(q,\Theta)$). To determine whether the optima we find are maxima we would have to take higher derivatives or inspect the vicinity of the optimum. For brevity we will abstain from doing so, however.

7.3 EM for Probabilistic PCA: The concrete algorithm

Having derived E- and M-step solutions, we can now state an explicit EM-Algorithm for PCA. It can be given in the following form: **EM for PCA**

(i) initialize model parameters W and σ^2 and compute $\vec{\mu}$ using (144)

(ii) compute the moments (139) and (140) (E-step)

(iii) update the parameters W and σ^2 using update rules (142) and (181) (M-step)

(iv) go to (ii) unless the parameters have converged

In practice we find that the algorithm does increase the likelihood. In conclusion, we have found an algorithm that can give us the maximum likelihood solution we were looking for. The algorithm is usually referred to as *probabilistic PCA* (p-PCA).

7.4 Factor Analysis

A popular approach to data analysis which has been developed in parallel to conventional PCA is *Factor Analysis*. In the generative interpretation, factor analysis is simply a generalization of probabilistic PCA

in that we assume a more general noise model, i.e., a more general distribution in data space given a hidden variable. The generative model of Factor Analysis is given by:

$$p(\vec{z}\,|\,\Theta) = \mathcal{N}(\vec{z};\vec{0},\mathbb{1}) \tag{145}$$

$$p(\vec{x}\,|\,\Theta) = \mathcal{N}(\vec{x};W\vec{z}+\vec{\mu},\Phi) \quad \text{where} \quad \Phi = \begin{pmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_D^2 \end{pmatrix} \tag{146}$$

In contrast to probabilistic PCA we now have different variances along the axes of the data space. In other words, the Gaussian distribution which models the data noise is not isotropic anymore. To find the maximum likelihood parameters of Factor Analysis we can use EM and proceed analogously to the treatment of probabilistic PCA.

8 Deterministic Principal Component Analysis

The problem of finding the best hyperplane given a set of data points is a central and relatively old problem in data analysis. Probabilistic PCA is comparably recent (Roweis, 1998; Tipping and Bishop, 1999) and offers a way to solve the problem using a probabilistic interpretation and the EM algorithm. Conventionally, the PCA problem is solved differently, however. In this section, we will follow a very common way to derive a solution for the hyperplane. It will turn out that the PCA problem can be solved without relying on an iterative procedure.

8.1 One-dimensional PCA: Maximizing the Variance

Let us start with the problem of finding the 'best' one-dimensional subspace in a two-dimensional space. For this, consider the set of two dimensional data points shown in Fig. 16. As can be observed, the data points lie close to a straight line. So our goal will be to describe a method to find the parameters of the line. Later on, we will generalize to higher dimensions.

To define any straight line we need one point on the line (which we will denote $\vec{mu}$) and the direction on direction (which we will denote $\vec{n}$). In our case, we will take $\vec{\mu}$ to be the mean of the data points $\{\vec{x}^{(1)}, ..., \vec{x}^{(N)}\}$, given by:

$$\vec{\mu} \;=\; \frac{1}{N}\sum_i \vec{x}^{(i)} \tag{147}$$

For finding the direction $\vec{n}$ we can first assume without loss of generality a unit vector ($|\vec{n}| = 1$). We continue our construction by defining the variance of the data projected onto the line. To do so, we need to compute the projections for all the data points $\vec{x}^{(i)}$ into the line. To this end we define the projection operator $\vec{P}_{\vec{n}}$ onto the line given by the unit vector $\vec{n}$:

$$\vec{P}_{\vec{n}}(\vec{x}) \;=\; (\vec{x}^T\vec{n})\,\vec{n} \;=\; (\vec{n}^T\vec{x})\,\vec{n} \;=\; (\vec{n}\vec{n}^T)\,\vec{x}$$

Where $\vec{P}_{\vec{n}}$ is the projection operator onto the given straight line. As a projection operator, it satisfies the following property: $\vec{P}_{\vec{n}}(\vec{P}_{\vec{n}}(\vec{x}^{(i)})) = \vec{P}_{\vec{n}}(\vec{x}^{(i)})$;

$$\begin{aligned}
\vec{P}_{\vec{n}}(\vec{P}_{\vec{n}}(\vec{x}^{(i)})) &= (\vec{n}\vec{n}^T)\,\vec{P}_{\vec{n}}(\vec{x}^{(i)}) \\
&= (\vec{n}\vec{n}^T)\,(\vec{n}\vec{n}^T)\vec{x}^{(i)} \\
&= \vec{n}\,\underbrace{(\vec{n}^T\vec{n})}_{1}\,\vec{n}^T\vec{x}^{(i)} \\
&= (\vec{n}\vec{n}^T)\vec{x}^{(i)} \\
&= \vec{P}_{\vec{n}}(\vec{x}^{(i)})
\end{aligned} \tag{148}$$

(149)

Let us first consider the mean of all projected data points $\vec{P}_{\vec{n}}(\vec{x}^{(i)})$ which we will denote by $\vec{\mu}_{\vec{n}}$. It is given by:

$$\vec{\mu}_{\vec{n}} \;=\; \frac{1}{N}\sum_i \vec{P}_{\vec{n}}(\vec{x}^{(i)}) \;=\; \frac{1}{N}\sum_i (\vec{n}\vec{n}^T)\vec{x}^{(i)} \tag{150}$$

$$\;=\; (\vec{n}\vec{n}^T)\frac{1}{N}\sum_i \vec{x}^{(i)} \;=\; (\vec{n}^T\vec{\mu})\vec{n} \;=\; (\vec{n}\vec{n}^T)\vec{\mu} \;=\; \vec{P}_{\vec{n}}(\vec{\mu}) \tag{151}$$

Based on the previous equations 147-160 we can now also (for an arbitrary line!) compute the variance of the data points once they have been projected on the line. It is given by:

$$\begin{aligned}
Var_{\vec{n}} &= \frac{1}{N}\sum_i |\vec{P}_{\vec{n}}(\vec{x}^{(i)}) - \vec{\mu}_{\vec{n}}|^2 = \frac{1}{N}\sum_i |(\vec{n}\vec{n}^T)\vec{x}^{(i)} - (\vec{n}\vec{n}^T)\vec{\mu}|^2 \\
&= \frac{1}{N}\sum_i |(\vec{n}\vec{n}^T)\,(\vec{x}^{(i)} - \vec{\mu})|^2 = \frac{1}{N}\sum_i \big((\vec{n}\vec{n}^T)\,(\vec{x}^{(i)} - \vec{\mu})\big)^T\big((\vec{n}\vec{n}^T)\,(\vec{x}^{(i)} - \vec{\mu})\big) \\
&= \frac{1}{N}\sum_i (\vec{x}^{(i)} - \vec{\mu})^T\big(\vec{n}\vec{n}^T\big)^T(\vec{n}\vec{n}^T)\,(\vec{x}^{(i)} - \vec{\mu}) \\
&= \frac{1}{N}\sum_i (\vec{x}^{(i)} - \vec{\mu})^T\vec{n}\,\underbrace{(\vec{n}^T\,\vec{n})}_{1}\,\vec{n}^T\,(\vec{x}^{(i)} - \vec{\mu}) \\
&= \frac{1}{N}\sum_i (\vec{x}^{(i)} - \vec{\mu})^T\vec{n}\,\vec{n}^T\,(\vec{x}^{(i)} - \vec{\mu}) = \frac{1}{N}\sum_i \vec{n}^T(\vec{x}^{(i)} - \vec{\mu})\,(\vec{x}^{(i)} - \vec{\mu})^T\vec{n} \\
&= \vec{n}^T\underbrace{\frac{1}{N}\sum_i (\vec{x}^{(i)} - \vec{\mu})\,(\vec{x}^{(i)} - \vec{\mu})^T}_{S}\,\vec{n} \\
\Rightarrow \quad Var_{\vec{n}} &= \vec{n}^T S\,\vec{n} \quad \text{where} \quad S = \frac{1}{N}\sum_i (\vec{x}^{(i)} - \vec{\mu})\,(\vec{x}^{(i)} - \vec{\mu})^T\,. \qquad (152)
\end{aligned}$$

In the last line, S is the data covariance matrix. In the fifth line, we have used the fact that the expression $(\vec{x}^{(i)} - \vec{\mu})^T\vec{n})$ is a scalar, so it can be transposed.

The variance $Var_{\vec{n}}$ defines how much the projected data points 'stretch' along the line. Considering this stretch geometrically, it can be realized, that the a line with maximal stretch (i.e., maximal variance) is what we seek: it is the line that is maximally aligned with the spread direction of the original data points. So let us try to derive the value for $\vec{n}$ that maximizes the variance $Var_{\vec{n}}$.

As usual, we would consider the derivative of the function $Var_{\vec{n}}$ w.r.t. $\vec{n}$. However, the vector $\vec{n}$ was constrained to be of unit length $\vec{n}^T\vec{n} = 1$. A necessary condition for an optimum of $Var_{\vec{n}}$ is consequently:

$$\frac{\partial}{\partial\vec{n}}Var(\vec{P}_{\vec{n}}(\vec{x})) + \lambda\frac{\partial}{\partial\vec{n}}g(\vec{n}) = 0 \quad \text{where} \quad g(\vec{n}) = 1 - \vec{n}^T\vec{n} \qquad (153)$$

is a Lagrange multiplier that ensures $\vec{n}$ to be a unit vector. Using well-known formulas for the derivation w.r.t. vectors, we obtain:

$$\begin{aligned}
&\frac{\partial}{\partial\vec{n}}\,\vec{n}^T S\,\vec{n} + \lambda\frac{\partial}{\partial\vec{n}}(1 - \vec{n}^T\vec{n}) \stackrel{!}{=} 0 \qquad (154) \\
&\Rightarrow \underbrace{(S + S^T)}_{S=S^T}\vec{n} - 2\,\lambda\,\vec{n} \stackrel{!}{=} 0 \\
&\Rightarrow S\,\vec{n} \stackrel{!}{=} \lambda\,\vec{n} \quad \text{where} \quad \lambda = \vec{n}^T\,S\,\vec{n} \text{ and } \vec{n}^T\vec{n} = 1\,. \qquad (155)
\end{aligned}$$

Our task is to find a vector $\vec{n}$ which satisfies these equations. If we do, the vector $\vec{n}$ defines a line with an optimum (or saddle point) for the variance.

Considering (155) we recognize an eigenvector equation for a symmetric 2×2 matrix S. Assuming S is not degenerated, then we know from elementary matrix algebra that S has two (normed) eigenvectors with real eigenvalues, and the eigenvectors are orthogonal to each other. We will refer to the two eigenvectors as $\vec{u}_1$ and $\vec{u}_2$, and to the corresponding eigenvalues as η_1 and η_2. Let us try one of the eigenvectors, e.g. $\vec{n} = \vec{u}_1$, to see if it satisfies the equations. Indeed we get (starting with the middle

equations):

$$\lambda \;=\; \vec{n}^T\, S\, \vec{n} \;=\; \vec{u}_1^T\, S\, \vec{u}_1 \;=\; \vec{u}_1^T\, \eta_1\, \vec{u}_1 \;=\; \eta_1 \tag{156}$$

So the first equation is satisfied if $S\,\vec{n} \;=\; \eta_1 \vec{n}$ is fulfilled, which is by definition of $\vec{u}_1$ as an eigenvector of S the case. Naturally, also the third equation is satisfied. We have consequently found an optimum. However, with the same argumentation, also $\vec{n} = \vec{u}_2$ solves the equations and is an optimum. Therefore, we have to determine which of the two solutions corresponds to a maximum of the variance. We can do so by simply computing the variance $Var_{\vec{n}}$ for $\vec{n} = \vec{u}_1$ and $\vec{n} = \vec{u}_2$:

$$Var_{u_1} \;=\; \vec{u}_1^T S\, \vec{u}_1 = \eta_1 \quad \text{and} \quad Var_{u_2} \;=\; \vec{u}_2^T S\, \vec{u}_2 = \eta_2\,. \tag{157}$$

Hence, the eigenvector associated with the *largest* eigenvalue of S corresponds to the maximum of the variance $Var_{\vec{n}}$. If we introduce the convention that always $\vec{u}_1$ is the eigenvalue associated with the larger eigenvector of S, then the sought line is given by:

$$\vec{n} \;=\; \vec{u}_1 \quad \text{and} \quad \vec{\mu} \;=\; \frac{1}{N} \sum_i \vec{x}^{(i)}\,. \tag{158}$$

Note that the derivations above also apply for the problem of seeking the line with maximal variance in a D-dimensional data space with $D > 2$. In that case, the data covariance matrix is $D \times D$, and it has D orthonormal eigenvectors. Selecting the eigenvector that corresponds to the largest eigenvalue, again represents the maximum variance solution.

8.2 General Deterministic PCA: Maximizing the Variance

Now we will extend the formulation above to hyperplanes of dimension M in a D-dimensional data space. Obviously, $1 \leq M < D$. First, consider again how an M-dimensional hyperplane is parameterized: We require a point on the plane denoted by $\vec{\mu}$ and use a set of M orthonormal vectors $\vec{n}_1, \vec{n}_2, ..., \vec{n}_M$ which span the plane. In formulas, it applies for the orthonormal vectors by definition:

$$\vec{n}_m^T\, \vec{n}_{m'} \quad = \quad \delta_{mm'}\,. \tag{159}$$

In words, the scalar product of two vectors $\vec{n}_m$ and $\vec{n}_{m'}$ is zero (orthogonality) except for the case $m = m'$ in which case the scalar product is one (normed vectors).

Given a hyperplane define as above, the projection of a data point $\vec{x}^{(i)}$ onto the plane is given by:

$$\begin{aligned}
\vec{P}_p(\vec{x}^{(i)}) \quad &= \quad \vec{P}_{\vec{n}_1,\ldots,\vec{n}_M}(\vec{x}^{(i)}) \;=\; \sum_{m=1}^{M} \big((\vec{x}^{(i)})^T \vec{n}_m\big)\vec{n}_m \;=\; \sum_m \vec{n}_m \big(\vec{n}_m^T\, \vec{x}^{(i)}\big) \;=\; \big(\sum_m \vec{n}_m \vec{n}_m^T\big)\, \vec{x}^{(i)} \\
\vec{P}_p(\vec{x}^{(i)}) \quad &= \quad \big(\sum_m \vec{n}_m \vec{n}_m^T\big)\, \vec{x}^{(i)}
\end{aligned}$$

We abbreviated the projection onto the plane with $\vec{n}_1, \vec{n}_2, ..., \vec{n}_M$ as $\vec{P}_p$ for readibility.

The mean of the projected data points can again be shown to be equal to the projection of the mean:

$$\vec{\mu}_p \;=\; \vec{\mu}_{\vec{n}_1,\ldots,\vec{n}_M} \quad = \quad \frac{1}{N} \sum_i \vec{P}_p(\vec{x}^{(i)}) \;=\; \frac{1}{N} \sum_i \big(\sum_m \vec{n}_m \vec{n}_m^T\big)\vec{x}^{(i)} \tag{160}$$

$$= \quad \big(\sum_m \vec{n}_m \vec{n}_m^T\big) \frac{1}{N} \sum_i \vec{x}^{(i)} \;=\; \big(\sum_m \vec{n}_m \vec{n}_m^T\big)\vec{\mu} \;=\; \vec{P}_p(\vec{\mu}) \tag{161}$$

The variance of the projected data points, i.e. their mean squared deviation from the mean $\vec{\mu}_p$, can be computed in analogy to the one-dimensional case (like for the mean, we abbreviate with $Var_p =$

$Var_{\vec{n}_1,\ldots,\vec{n}_M}$:

$$
\begin{aligned}
Var_p &= \frac{1}{N}\sum_i |P_p(\vec{x}^{(i)}) - \vec{\mu}_{\vec{n}}|^2 = \frac{1}{N}\sum_i \big|\big(\sum_m \vec{n}_m\vec{n}_m^T\big)\vec{x}^{(i)} - \big(\sum_m \vec{n}_m\vec{n}_m^T\big)\vec{\mu}\big|^2 \\
&= \frac{1}{N}\sum_i \big|\big(\sum_m \vec{n}_m\vec{n}_m^T\big)\,(\vec{x}^{(i)} - \vec{\mu})\big|^2 \\
&= \frac{1}{N}\sum_i \Big(\big(\sum_m \vec{n}_m\vec{n}_m^T\big)\,(\vec{x}^{(i)} - \vec{\mu})\Big)^T\Big(\big(\sum_m \vec{n}_m\vec{n}_m^T\big)\,(\vec{x}^{(i)} - \vec{\mu})\Big) \\
&= \frac{1}{N}\sum_i (\vec{x}^{(i)} - \vec{\mu})^T\big(\sum_m \vec{n}_m\vec{n}_m^T\big)^T\big(\sum_{m'} \vec{n}_{m'}\vec{n}_{m'}^T\big)\,(\vec{x}^{(i)} - \vec{\mu}) \\
&= \frac{1}{N}\sum_i (\vec{x}^{(i)} - \vec{\mu})^T\sum_m \vec{n}_m\vec{n}_m^T\sum_{m'} \vec{n}_{m'}\vec{n}_{m'}^T\,(\vec{x}^{(i)} - \vec{\mu}) \\
&= \frac{1}{N}\sum_i (\vec{x}^{(i)} - \vec{\mu})^T\sum_m\sum_{m'} \vec{n}_m\,\underbrace{(\vec{n}_m^T\,\vec{n}_{m'})}_{\delta_{mm'}}\,\vec{n}_{m'}^T\,(\vec{x}^{(i)} - \vec{\mu}) \\
&= \frac{1}{N}\sum_i (\vec{x}^{(i)} - \vec{\mu})^T\sum_m\sum_{m'} \vec{n}_m\,\delta_{mm'}\,\vec{n}_{m'}^T\,(\vec{x}^{(i)} - \vec{\mu}) \\
&= \sum_m \frac{1}{N}\sum_i (\vec{x}^{(i)} - \vec{\mu})^T\vec{n}_m\,\vec{n}_m^T\,(\vec{x}^{(i)} - \vec{\mu}) \\
&= \sum_m \frac{1}{N}\sum_i \vec{n}_m^T(\vec{x}^{(i)} - \vec{\mu})\,(\vec{x}^{(i)} - \vec{\mu})^T\vec{n}_m \\
&= \sum_m \vec{n}_m^T\,\underbrace{\frac{1}{N}\sum_i (\vec{x}^{(i)} - \vec{\mu})\,(\vec{x}^{(i)} - \vec{\mu})^T}_{S}\,\vec{n}_m \\
\Rightarrow\ Var_p &= \sum_{m=1}^M \vec{n}_m^T S\,\vec{n}_m
\end{aligned}
\tag{162}
$$

We seek the maximum of Var_p w.r.t. to the M orthonormal vectors $\vec{n}_1,\ldots,\vec{n}_M$. A necessary condition for an extremum is given if for all $m \in \{1,...,M\}$ applies:

$$
\frac{\partial}{\partial\vec{n}_m}Var_p + \frac{\partial}{\partial\vec{n}_m}\sum_{m'=1}^M \lambda_{m'}g(\vec{n}_{m'}) = 0 \quad \text{where} \quad g(\vec{n}_{m'}) = 1 - \vec{n}_{m'}^T\vec{n}_{m'}\,. \tag{163}
$$

The functions $g(\vec{n}_{m'}$ represent the boundary conditions of all vectors having to be normalized, and the scalars $\lambda_1,...,\lambda_M$ are the corresponding Lagrange multipliers of the boundary conditions.

By inserting expression (162) for the variance Var_p we obtain:

$$
\begin{aligned}
\frac{\partial}{\partial\vec{n}_m}\sum_{m'}\vec{n}_{m'}^T S\vec{n}_{m'} + \sum_{m'=1}^M \lambda_{m'}\frac{\partial}{\partial\vec{n_m}}\,(1 - \vec{n}_{m'}^T\vec{n}_{m'}) &\stackrel{!}{=} 0 \\
\Rightarrow 2\,S\,\vec{n}_m - 2\,\lambda_m\,\vec{n}_m &\stackrel{!}{=} 0 \\
\Rightarrow S\,\vec{n}_m &= \lambda_m\,\vec{n}_m
\end{aligned}
\tag{164}
$$

By exploiting the boundary conditions, the value of each λ_m can be computed to be $\lambda_m = \vec{n}_m^T\,S\,\vec{n}_m$. So what we are seeking are now M vectors $\vec{n}_m$ that together satisfy all of the following equations:

$$
S\,\vec{n}_m = \lambda_m\,\vec{n}_m, \quad \text{with } \lambda_m = \vec{n}_m^T\,S\,\vec{n}_m, \quad \text{and} \quad \vec{n}_m^T\vec{n}_m = 1 \tag{165}
$$

Example: Suppose we have a $D = 7$ dimensional data space and seek a $M = 3$ dimensional hyperplane.

The matrix S is in this case a 7×7 matrix. As it is symmetric, S has seven orthonormal eigenvectors. As Eqn. 165(left) defines an eigenvector equation (and because of our experience with the one-dimensional case) the eigenvectors of (165) offer themselves as providing a solution. However, we seek just $M = 3$ vectors $\vec{n}_m$. Let us try three arbitrarily chosen eigenvectors $\vec{u}_i$ of S, say $\vec{n}_1 = \vec{u}_2$, $\vec{n}_2 = \vec{u}_5$ and $\vec{n}_3 = \vec{u}_6$ with corresponding eigenvalues η_2, η_5 and η_6. Let us test if Eqn. 165 is solved by this choice. Starting with the middle equation of Eqn. 165 we get:

$$\begin{aligned}
\lambda_1 &= \vec{n}_1^T\, S\, \vec{n}_1 = \vec{u}_2^T\, S\, \vec{u}_2 = \vec{u}_2^T\, \eta_2\, \vec{u}_2 = \eta_2 \\
\lambda_2 &= \vec{n}_2^T\, S\, \vec{n}_2 = \vec{u}_5^T\, S\, \vec{u}_5 = \vec{u}_5^T\, \eta_5\, \vec{u}_5 = \eta_5 \\
\lambda_3 &= \vec{n}_3^T\, S\, \vec{n}_3 = \vec{u}_6^T\, S\, \vec{u}_6 = \vec{u}_6^T\, \eta_6\, \vec{u}_6 = \eta_6
\end{aligned}$$

Correspondingly it follows:

$$\begin{aligned}
S\,\vec{n}_1 &= S\,\vec{u}_2 = \eta_2\,\vec{u}_2 = \lambda_1 \vec{n}_1 \\
S\,\vec{n}_2 &= S\,\vec{u}_5 = \eta_5\,\vec{u}_5 = \lambda_2 \vec{n}_2 \\
S\,\vec{n}_3 &= S\,\vec{u}_6 = \eta_6\,\vec{u}_6 = \lambda_3 \vec{n}_3
\end{aligned}$$

So the choice $\vec{n}_1 = \vec{u}_2$, $\vec{n}_2 = \vec{u}_5$ and $\vec{n}_3 = \vec{u}_6$ solves Eqn. 165, so the vectors $(\vec{u}_2, \vec{u}_5, \vec{u}_6)$ span a three-dimensional hyperplane which represents an extremum for the variance Var_p. However, also any other choice of three distinct eigenvectors of S will represent a solutions (as is obvious from how we showed this for the above three vectors.

The same also applies in general: any subset of M eigenvectors of S represents an extremum of the variance Var_p. But how do we find the combination that corresponds to the maximum? To answer the question, let us simply compute the value for variance Var_p for a given set of M vectors. For our example with $M = 3$ we get using (162):

$$\begin{aligned}
Var_p &= \vec{n}_1^T S\, \vec{n}_1 + \vec{n}_2^T S\, \vec{n}_2 + \vec{n}_3^T S\, \vec{n}_3 \\
&= \vec{u}_2^T S\, \vec{u}_2 + \vec{u}_5^T S\, \vec{u}_5 + \vec{u}_6^T S\, \vec{u}_6 \\
&= \vec{u}_2^T \eta_2\, \vec{u}_2 + \vec{u}_5^T \eta_5 \vec{u}_5 + \vec{u}_6^T \eta_6 \vec{u}_6 \\
&= \eta_2 + \eta_5 + \eta_6
\end{aligned} \tag{166}$$

So the variance is the sum of the eigenvalues that correspond to the eigenvectors used for $\vec{n}_1$, $\vec{n}_2$, and $\vec{n}_3$. Considering (166) the largest possible variance Var_p is achieved if we pick for $\vec{n}_1$, $\vec{n}_2$, and $\vec{n}_3$ the three eigenvectors of S that correspond to the largest three eigenvalues.

Of course, the result generalizes to any M with the variance being given by the sum of the M eigenvalues of the eigenvectors chosen. Accordingly, we obtain the general result:

The solution for the hyperplane is given by choosing for $\vec{n}_1$ to $\vec{n}_M$ the M eigenvectors $\vec{u}_m$ of S that correspond to the M largest eigenvalues η_m.

With the convention that the numbering of the eigenvalues of S is such that $\vec{u}_1$ corresponds to the eigenvector of S with the largest eigenvalue, $\vec{u}_2$ to the eigenvector with the second largest etc., the hyperplane is spanned by the vectors $\vec{u}_1$ to $\vec{u}_M$. Although there are different conventions in the literature, the vector $\vec{u}_1$ is often referred to as the first principal component, $\vec{u}_2$ as the second principal component etc. A common step in data analysis is, for instance, to plot the eigenvalues η_1 to η_M in descending order (note that by convention $\eta_1 \geq \eta_2 \geq \ldots \eta_M$. Imaging plotting the eigenvalues in a histogram-like plot. If data (like for our initial example) indeed lies close to a hyperplane than we will observe a drop from one eigenvalue to the next. The eigenvectors of the eigenvalues before the 'drop' are then considered to make up the hyperplane. Data compression is then one example application.

We have discussed here the maximum variance approach to derive deterministic PCA. Other approaches are the minimum squared error approach or singular value decomposition (SVD). Especially the minimum squared error approach may be familiar, e.g. from linear regression. The same hyperplane (and indeed the same algorithm is achieved for the minimum squared error approach).

To actually obtain the principal components, the eigenvectors of S actually have to be computed in some way. For such a computation one can use any of the many algorithms that have been suggested for eigenvector computation in the literature. The most popular ones are available in common software packages for data analysis. Depending on the data size and dimensionality, eigenvector computations can be computationally expensive though. So current research often tries to come up with approximate methods that can be estimate eigenvectors and eigenvalues more efficiently. For the PCA problem, methods that only compute the M eigenvectors of the largest M eigenvalues are of particular interest as usually, when PCA is applied, the hyperplane M is much smaller than the data dimensionality D.

PCA is of very central importance for data processing and one of the most intensively researched single algorithms. It is often applied as a first computational step to handle big data, for instance, and is the most elementary dimensionality reduction method or method for data compression etc (examples shown in lecture).

PCA is also a nice example how a given problem can be solved in different ways. We have here encountered EM for p-PCA as one approach, and the maximum variance approach of deterministic PCA. Both approaches can be shown to find the same hyperplane. Deterministic PCA (or simply called PCA) did not use the EM algorithm. Instead, it relies on eigenvalue computations. Both PCA and p-PCA have advantageous and disadvantageous. PCA allows for the computation of the hyperplane solution without iterations, i.e., we do neither have to specify starting values for the parameters nor does an algorithm have to determine when parameter updating has converged. Furthermore, deterministic PCA can directly profit from any results for efficient eigenvalue computation. One advantage of p-PCA is that we obtain probabilistic estimates of W as well as σ and a likelihood value which can be instructive about the quality of the PCA solution. Furthermore, and maybe counterintuitive, p-PCA can be faster. The reason is that eigenvalue computation for deterministic PCA is expensive and has be done for $D \times D$ matrices (although there are more efficiency variants). Probabilistic PCA in contrast has to invert a $M \times M$ matrix for the M-step (and M is usually much smaller than D). We require several M-steps but usually not so many such that the times saved by avoiding to operate in the $D \times D$ space outweigh the time required for additional iterations. Finally, EM for p-PCA can be generalized. For instance, the generalization to EM for Factor Analysis is straight-forward while closed-form solutions for Factor Analysis have not been suggested, so far.

8.3 PCA: Minimum Error Formulation (need revision and is not used in lecture)

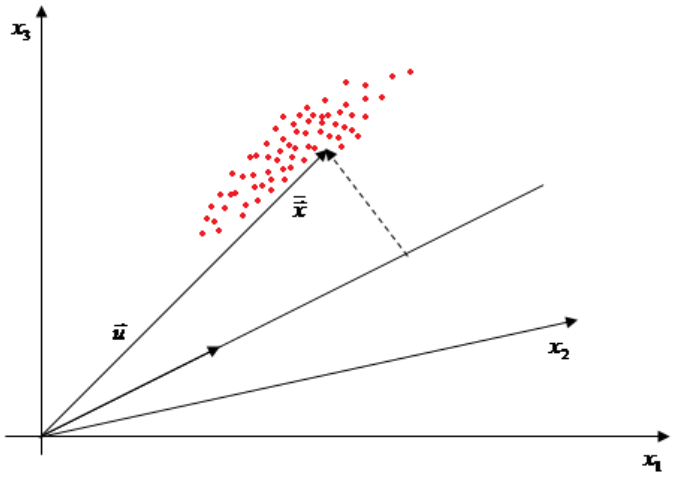


Figure 17: to be given

We use a hyperplane of $M < D$ and unit length vectors are $\vec{u}_m$

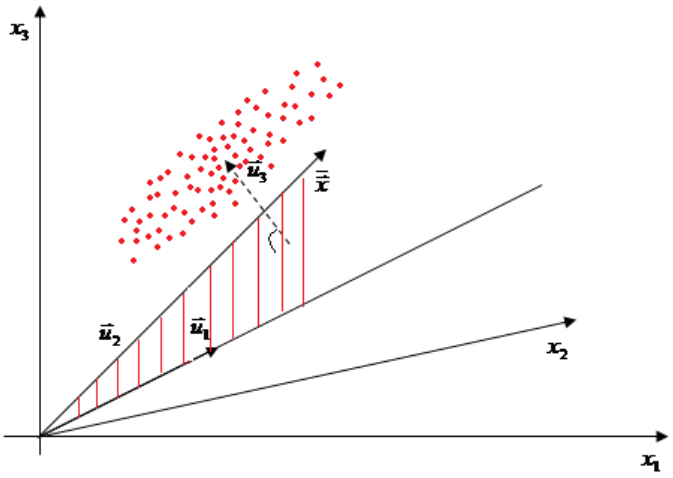


Figure 18: to be given

We can project all data points onto the hyperplane

$$P\vec{x}^n = \sum_{m=1}^{M} \vec{u}_m^T \vec{x}^n \vec{u}_m$$

To reconstruct a data point as well as possible we globally move the hyperplane within the space orthogonal to itself:

$$\tilde{\vec{x}}_n = P\vec{x}^n + \sum_{m=\mu+1}^{D} b_m \vec{u}_m$$

Our objective is now to minimise the reconstruction error given by

$$\begin{aligned} E &= \frac{1}{N}\sum_{n=1}^{N} \parallel \vec{x}^n - \tilde{\vec{x}}_n \parallel^2 \\ &= \frac{1}{N}\sum_{n}(\vec{x}^n - \tilde{\vec{x}}_n)^T(\vec{x}^n - \tilde{\vec{x}}_n) \\ &= \frac{1}{N}\sum_{n}(\vec{x}^n - \sum_{m=1}^{M} \vec{u}_m^T\vec{x}^n\vec{u}_m - \sum_{m=\mu+1}^{D} b_m\vec{u}_m)^T(...) \\ &= \frac{1}{N}\sum_{n}((\vec{x}^n)^T - \sum_{m} \vec{u}_m^T(\vec{x}^n)^T\vec{u}_m - \sum_{l} b_l\vec{u}_l^T)^T(...) \end{aligned}$$

then

$$\begin{aligned} \frac{\partial}{\partial\vec{u}_{m'}}E &= \frac{1}{N}\sum_{n}(-(\vec{x}^n)^T\vec{u}_{m'} - \vec{u}_{m'}^T\vec{x}^n)((\vec{x}^n)^T - \sum_{m} \vec{u}_m^T(\vec{x}^n)^T\vec{u}_m - \sum_{l} b_l\vec{u}_l^T) \\ &= -\frac{2}{N}\sum_{n} \vec{u}_{m'}^T\vec{x}^n((\vec{x}^n)^T - \sum_{m} \vec{u}_m^T(\vec{x}^n)^T\vec{u}_m - \sum_{l} b_l\vec{u}_l^T) \\ \frac{\partial}{\partial b_{m'}}E &= \frac{1}{N}\sum_{n} 2(-\vec{u}_{m'}^T)(\vec{x}^n - \sum_{m=1}^{M} \vec{u}_m^T\vec{x}^n\vec{u}_m - \sum_{m=\mu+1}^{D} b_m\vec{u}_m) \\ &= -\frac{2}{N}\sum_{n}(\vec{u}_{m'}^T\vec{x}^n - b_{m'}) \\ &= -2(\vec{u}_{m'}^T \underbrace{\frac{1}{N}\sum_{n}\vec{x}^n}_{\vec{\mu}} - b_{m'}) = 0 \\ \Rightarrow b_{m'} &= \vec{u}_{m'}^T.\vec{\mu} \end{aligned} \tag{167}$$

The $b_{m'}$ are simply the projections of the mean $\vec{\mu}$ onto the orthogonal space.

$$\begin{aligned} \Rightarrow E &= \frac{1}{N}\sum_{n}(\vec{x}^n - \sum_{m} \vec{u}_m^T\vec{x}^n\vec{u}_m - \sum_{l} \vec{u}_l^T\vec{\mu}\vec{u}_l)^T(...) \\ &= \frac{1}{N}\sum_{n}(\sum_{m=1}^{D} \vec{u}_m^T\vec{x}^n\vec{u}_m - \sum_{m=1}^{\mu} \vec{u}_m^T\vec{x}^n\vec{u}_m - \sum_{m=\mu+1}^{D} \vec{u}_m^T\vec{\mu}\vec{u}_m)^T(...) \\ &= \frac{1}{N}\sum_{n}(\sum_{m=\mu+1}^{D} \vec{u}_m^T(\vec{x}^n - \vec{\mu})\vec{u}_m)^T(...) \\ &= \frac{1}{N}\sum_{n}\sum_{m,m'=\mu+1}^{D} \vec{u}_m^T(\vec{x}^n - \vec{\mu})^T\vec{u}_m\vec{u}_{m'}^T(\vec{x}^n - \vec{\mu})\vec{u}_{m'} \\ &= \sum_{m=\mu+1}^{D} \vec{u}_m^T \underbrace{\frac{1}{N}\sum_{n}(\vec{x}^n - \vec{\mu})(\vec{x}^n - \vec{\mu})^T}_{S} \vec{u}_m \\ \Rightarrow E &= \sum_{m=\mu+1}^{D} \vec{u}_m^T S\vec{u}_m \end{aligned} \tag{168}$$

Using the same computations as for $Var(P\vec{x})$ above, the error E is minimized if the $\vec{u}_m$ are eigenvectors of S and when these eigenvectors are the ones with the $(D-M)$ smallest eigenvalues. The projection

plane is thus made up of the eigenvectors os S with the M largest eigenvalues.
Minimizing E and maximizing $Var(P\vec{x})$ are thus equivalent!

8.4 Application of PCA

Using the above formulas, we can write

$$\tilde{\vec{x}}_n = \bar{\vec{x}} + \sum_{m=1}^{M} \underbrace{((\vec{x}^n)^T \vec{u}_m - \bar{\vec{x}}^T \vec{u}_m)}_{M \text{ components per vector}} \quad \vec{u}_m$$

$\Rightarrow$ data compression.
Whitening (or PCA-Whitening):

$$S\vec{u} = \lambda\vec{u} \quad \text{or} \quad SU = UL$$

with

$$L = \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_D \end{pmatrix}$$

$$\vec{y}_n = L^{-\frac{1}{2}} U^T (\vec{x}^n - \bar{\vec{x}}^n)$$

It follows: $\sum_n \vec{y}_n = 0$

$$\begin{aligned} \frac{1}{N} \sum_n \vec{y}_n \vec{y}_n^T &= \frac{1}{N} \sum_n L^{-\frac{1}{2}} U^T (\vec{x}_n - \bar{\vec{x}}^n) L^{-\frac{1}{2}} U^T (\vec{x}_n - \bar{\vec{x}}_n)^T U \\ &= L^{-1} U^T S U = 1 \end{aligned}$$

9 Relating Deterministic and Probabilisitic PCA

It is also possible to find a direct maximum likelihood solution for W and σ^2 but derivations are more complex (see, e.g., Bishop and Tipping, 1999) and we will only give the main results.

The Direct Maximum Likelihood Solution

Instead of deriving parameter update rules for the PCA generative model we can also derive the maximum likelihood solution directly. Our generative model (125) and (126) still has a nice enough structure to allow for doing this.

The data likelihood under the PCA generative model is given by (247). Using (125) and (126) we get:

$$\begin{aligned} \mathcal{L}(\Theta) &= \sum_{n=1}^{N} \log \left(p(\vec{x}^{(n)} \,|\, \Theta) \right) \\ &= \sum_{n=1}^{N} \log \left(\int p(\vec{x}^{(n)} \,|\, z, \Theta^{\text{old}})\, p(z \,|\, \Theta^{\text{old}})\, dz \right) \\ &= \sum_{n=1}^{N} \log \left(\int \mathcal{N}(\vec{x}^{(n)} \,|\, Wz + \vec{\mu}, \sigma^2 \mathbb{1})\, \mathcal{N}(z \,|\, 0, 1)\, dz \right) \end{aligned} \tag{169}$$

Note that the argument of the logarithm is nothing else than the denominator in (129). Using the result in (133) we therefore find:

$$\begin{aligned}\mathcal{L}(\Theta) &= \sum_{n=1}^{N} \log\Big(\mathcal{N}(\vec{x}^{(n)} \,|\, \vec{\mu}, C)\Big) \\ &= \sum_{n=1}^{N} \Big(-\frac{1}{2}\log(|2\,\pi\,C|) \,-\, \frac{1}{2}(\vec{x}^{(n)} - \vec{\mu})^T\, C^{-1}\,(\vec{x}^{(n)} - \vec{\mu})\Big) \\ &= -\frac{N}{2}\log(|2\,\pi\,C|) \,-\, \frac{1}{2}\sum_{n=1}^{N}(\vec{x}^{(n)} - \vec{\mu})^T\, C^{-1}\,(\vec{x}^{(n)} - \vec{\mu}) \end{aligned} \tag{170}$$

If we now take the derivative w.r.t. $\vec{\mu}$ and set it to zero, we obtain:

$$\vec{\mu} \;=\; \frac{1}{N}\sum_{n=1}^{N}\vec{x}^{(n)} \tag{171}$$

This result tells us that we can directly compute $\vec{\mu}$ from the maximum likelihood solution using the data points. We therefore do not require an update rules for $\vec{\mu}$ but can compute $\vec{\mu}$ using (171) and keep it fixed for all EM iterations.

Now note, that C is a function of Θ given by $C = WW^T + \sigma^2\mathbb{1}$ and that the likelihood (170) can be rewritten as:

$$\mathcal{L}(\Theta) \;=\; -\frac{N}{2}\Big(D\log(2\,\pi) + \log(|C|) + \mathrm{Tr}(C^{-1}S)\Big), \tag{172}$$

where S is the data covariance matrix $S = \frac{1}{N}\sum_{n=1}^{N}(\vec{x}^{(n)} - \vec{\mu})(\vec{x}^{(n)} - \vec{\mu})^T$. Using (172) it can be shown that a necessary condition for a maxima of the likelihood is that W takes the form:

$$W_{ML} \;=\; U_{ML}\,(L_{ML} - \sigma^2\mathbb{1})^{\frac{1}{2}}\,R \tag{173}$$

where U_{ML} contains as columns m eigenvectors $\vec{u}_i$ of S and where L_{ML} is a diagonal matrix containing the eigenvalues λ_i associated with $\vec{u}_i$:

$$U_{ML} \,=\, \begin{pmatrix} \vec{u}_1 & \vec{u}_2 & \cdots & \vec{u}_m \end{pmatrix}\; L_{ML} \,=\, \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_m \end{pmatrix} \tag{174}$$

R is an arbitrary orthogonal matrix.

A necessary condition for the maximum likelihood solution for σ^2 is given by:

$$\sigma^2_{ML} \;=\; \frac{1}{D-m}\sum_{i=m+1}^{D}\lambda_i\,, \tag{175}$$

where λ_i are again the eigenvalues associated with $\vec{u}_i$.

Having computed the principle form of the solutions associated with the maximal likelihood solution, we can compute the solution(s) for the global maximum. It turns out (see, e.g., Bishop and Tipping, 1999) that for the global maximum likelihood solution U_{ML} consists of the eigenvectors (as columns) associated with the m largest eigenvalues of S. Note that this solution is not unique because of the arbitrary orthogonal matrix R in (173). This matrix reflects the fact that any rotation of the hidden space results in the same generative model and thus in the same likelihood.

Note that according to (**??**) the matrix W_{ML} contains as columns the vectors which span an m-dimensional subspace in the D-dimensional data space. Finding the subspace which 'best' fit the data was

the task of conventional PCA. In conventional PCA we found as solution that the subspace is spanned by the eigenvectors of the matrix S which are associated to the largest eigenvalues. In probabilistic PCA we find that the subspace is spanned by the columns of W_{ML}. By inspecting (173) we can see, however, that these column vectors are proportional to the eigenvectors of S. Thus, probabilistic PCA and conventional PCA give the same subspace as solution (for more details see, e.g., Bishop 2007).

Finally, note that our EM approach to probabilistic PCA in practice approaches the direct maximum likelihood solution (173) and (175). Thus, we know that EM for probabilistic PCA recovers the same subspace as conventional PCA.

Advantages of probabilistic PCA (via EM or direct) are interpretability, potential numerical advantages and generalizability (see, e.g., Bishop 2006 for a discussion).

10 Sparse Coding and Independent Components

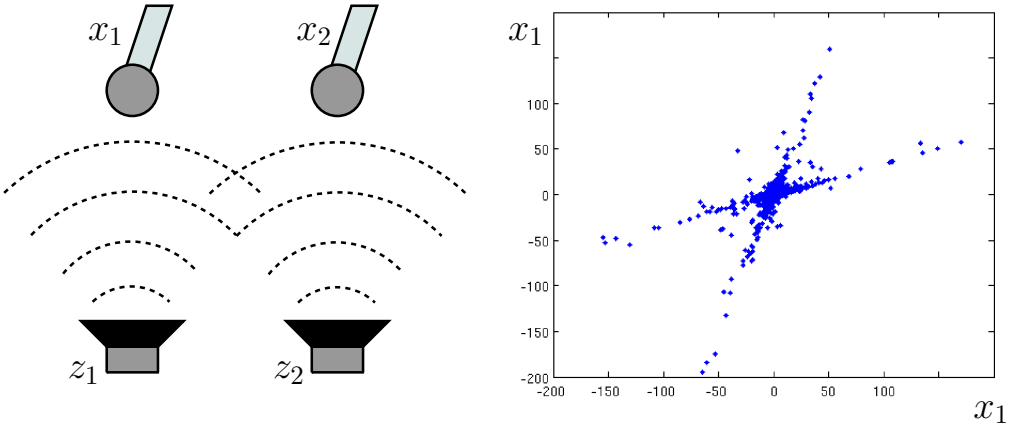


Figure 19: Two microphones record sound emitted by two loudspeakers (left). Data points from a mixture of two sparsely active hidden causes (right).

Consider the example of two loudspeakers and two microphones in Fig. 19. Suppose that each loudspeaker emits a pressure waves according to a non-Gaussian distribution. Also suppose that the amplitudes of the pressure waves the speakers emit are most of the time small and only occasionally very large. Note that this would be a reasonable assumption for spoken language. Now suppose we record the (mean) pressure at two different locations at each individual time interval n and approximate the velocity of sound waves to be infinite, i.e., instantly available at each microphone. Let us denote the pressure at location one by ${x^{(n)}}_1$ and at location two by ${x^{(n)}}_2$. If we record for N time intervals we obtain N data points $\vec{x}^{(n)}$. If we now disregard the time-dependency of the signals we can expect the data points $\vec{x}^{(n)}$ to be distributed as displayed in Fig. 19(right). Note that if only one loudspeaker were active all the time, we would expect just one 'branch' in the distribution. As a given microphone is closer to one of the loudspeaker and farer away form the other, one ${x^{(n)}}_i$ is larger than the other. (Note that in the case of just one branch we could apply some kind of PCA to find the subspace.)

Our task here is to find the sources or causes (here the loudspeaker signals) that have caused the data. If we follow a generative approach, we will first try to model the principle form of the data. Note that we cannot produce data distributions in Fig. 19 using the PCA generative model. It is, however, sufficient to just change the prior distribution (125) of the PCA generative model appropriately. If we change it from a Gaussian to a prior with so-called 'heavy tails', we can generate data as depicted in Fig. 19. If we

choose a Cauchy distribution as such a heavy tailed prior distribution, the generative model reads:

$$p(\vec{z}\,|\,\Theta) = \prod_{h=1}^{H} \mathcal{C}(z_h)\,, \quad \text{where} \quad \mathcal{C}(z_h) = \frac{1}{\pi\,(1+z_h^2)} \tag{176}$$

$$p(\vec{x}\,|\,\vec{z},\Theta) = \mathcal{N}(\vec{x};W\vec{z},\sigma^2\,\mathbb{1}) \tag{177}$$

Another prior that is actually much more frequently used is given by

$$p(\vec{z}\,|\,\Theta) = \prod_{h=1}^{H} \text{Laplace}(z_h)\,, \quad \text{where} \quad \text{Laplace}(z_h) = \frac{1}{2}\exp(-|z_h|) \tag{178}$$

$$p(\vec{x}\,|\,\vec{z},\Theta) = \mathcal{N}(\vec{x};W\vec{z},\sigma^2\,\mathbb{1}) \tag{179}$$

For a two-dimensional hidden space, values $\vec{z}$ generated according to the prior (176) look as depicted in Fig. 20(left). Note that in contrast to a product of Gaussians in PCA, the distribution in hidden space is not rotational symmetric anymore. Fig. 20(right) shows the difference between the Cauchy distribution and a Gaussian distribution in one-dimension. As can be observed, the Cauchy distribution has larger values far away from zero, and we therefore say it has heavy tails.

The prior distribution (176) models our expectation about the activity of the hidden causes. Note that for a number of hidden dimensions much larger than two, a prior as (176) results in hidden vectors $\vec{z}$ which have most entries z_i at values close to zero. Only some of the entries have values which are significantly larger. Large values are thus sparsely distributed within the vectors $\vec{z}$. A prior such as (176) is therefore often called a *sparse* prior.

The distribution of $\vec{x}$ given a generated data point $\vec{z}$ is the same as for the PCA generative model (with $\vec{\mu} = \vec{0}$ for simplicity), it is a linear projection from the hidden space to the data space. That is, we assume the interaction of the generated causes z_i to be linear. For the introductory example of two pressure waves this linear superposition assumption can be justified because of the physical properties of pressure waves.

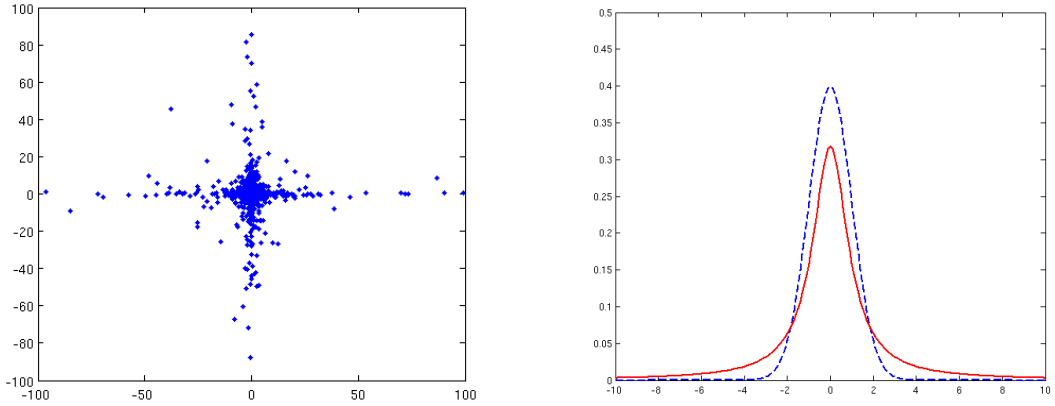


Figure 20: Left: Data points generated according to a Cauchy distribution. Right: Comparison between a one-dimensional Gaussian probability density function (dotted line) and a Cauchy probability density function (solid line).

10.1 EM for Sparse Coding

Having modeled the principle structure of the data, we now, as usual, want to find the parameters $\Theta = (W, \sigma^2)$ which maximize the likelihood of the data $\vec{x}_1, \ldots, \vec{x}_N$ under the generative model defined by (176) and (177). To maximize the likelihood $\mathcal{L}(\Theta)$ we use EM and proceed step-by-step as for the other generative models used.

First, we write down the free energy. As for p-PCA, we have continuous latent variables such that the free energy is given by:

$$\mathcal{F}(q,\Theta) \;=\; \sum_n \int q^{(n)}(\vec{z}) \,\Big(\log\big(p(\vec{x}^{(n)} \,|\, \vec{z},\Theta)\big) + \log\big(p(\vec{z}\,|\,\Theta)\big)\Big) d\vec{z} \;-\; \sum_n \int q^{(n)}(\vec{z}) \log\big(q^{(n)}(\vec{z})\big) d\vec{z}.$$

Inserting the generative model (176) and (177) we obtain:

$$\begin{aligned}\mathcal{F}(q,\theta) \;\;=\;\; & \sum_n \int_{\vec{z}} q^{(n)}(\vec{z}) \Big(-\frac{1}{2}\log\big(det(2\pi\sigma^2\mathbf{1})\big) - \frac{1}{2\sigma^2}(\vec{x}^{(n)} - W\vec{z})^T(\vec{x}^{(n)} - W\vec{z}) - \sum_h \log\big(\mathcal{C}(z_h)\big)\Big) d\vec{z} \\ & - \sum_n \int q^{(n)}(\vec{z}) \log\big(q^{(n)}(\vec{z})\big) d\vec{z}. \end{aligned} \tag{180}$$

The M-step

Let us start (other than for p-PCA) with the M-steps.

Derivative w.r.t. W. To determine the maximum of $\mathcal{F}(q,\Theta)$ w.r.t. the matrix W we consider the derivative of $\mathcal{F}(q,\Theta)$ w.r.t. W which has to be zero at an optimum. We quickly realize that the derivation is the same as for p-PCA (and very similar to the one for BSC):

$$\begin{aligned}\frac{d\mathcal{F}(q,\theta)}{dW} \;\;=\;\; & -\frac{1}{2\sigma^2}\sum_n \int q^{(n)}(\vec{z}) \frac{d}{dW}\big((\vec{x}^{(n)} - W\vec{z})^T(\vec{x}^{(n)} - W\vec{z})\big)\, d\vec{z} \\ =\;\; & -\frac{1}{2\sigma^2}\sum_n \int q^{(n)}(\vec{z}) \frac{d}{dW}(\vec{x}^{(n)T}\vec{x}^{(n)} - 2\vec{x}^{(n)T}W\vec{z} + \vec{z}^T W^T W\vec{z})\, d\vec{z} \\ =\;\; & -\frac{1}{2\sigma^2}\sum_n \int q^{(n)}(\vec{z}))(-2\vec{x}^{(n)}\vec{z}^T + 2W(\vec{z}\vec{z}^T))\, d\vec{z} \\ =\;\; & -\frac{1}{2\sigma^2}\sum_n \int q^{(n)}(\vec{z})(\vec{x}^{(n)}\vec{z}^T - W(\vec{z}\vec{z}^T))\, d\vec{z} \stackrel{!}{=} 0 \\ \Rightarrow\;\; & \sum_n \int q^{(n)}(\vec{z})\vec{x}^{(n)}\vec{z}^T\, d\vec{z} - \sum_n \int q^{(n)}(\vec{z}) W\vec{z}\vec{z}^T\, d\vec{z} \stackrel{!}{=} 0 \\ \Rightarrow\;\; & \sum_n \int q^{(n)}(\vec{z})\vec{x}^{(n)}\vec{z}^T\, d\vec{z} \;=\; W(\sum_n \int q^{(n)}(\vec{z})\vec{z}\vec{z}^T)\, d\vec{z} \end{aligned}$$

$\sum_n \int p(\vec{z}\,|\,\vec{x}^{(n)},\theta^{old})\vec{z}\vec{z}^T$ is a squared and symmetric matrix which we will also assume to be invertible. We then obtain:

$$\begin{aligned} \Rightarrow\;\; & W = (\sum_n \int q^{(n)}(\vec{z})\vec{x}^{(n)}\vec{z}^T\, d\vec{z})(\sum_n \int q^{(n)}(\vec{z})\;\vec{z}\vec{z}^T\, d\vec{z})^{-1} \\ \Rightarrow\;\; & W = (\sum_n \vec{x}^{(n)} \int q^{(n)}(\vec{z})\vec{z}^T\, d\vec{z})(\sum_n \int q^{(n)}(\vec{z})\vec{z}\vec{z}^T\, d\vec{z})^{-1} \\ or & \\ & W^{\rm new} = (\sum_n \vec{x}^{(n)}\langle\vec{z}\rangle_{q_n}^T)\,(\sum_n \langle\vec{z}\vec{z}^T\rangle_{q_n})^{-1} \end{aligned}$$

So basically the same update rule as for p-PCA and BSC. And again we have to know the first and the second moments of the posterior distribution: $\langle\vec{z}\rangle_{q_n}$ and $\langle\vec{z}\vec{z}^T\rangle_{q_n}$.

Derivative w.r.t. σ^2. With the same argumentation as above, the update rule for the variance parameter σ^2 is similar to the derivation (and the result) w.r.t. the same variance parameter for the p-PCA and the BSC model. We therefore state without derivation the result, which is given by:

$$\sigma^2_{\rm new} \;=\; \frac{1}{ND}\sum_{n=1}^{N}\Big((\vec{x}^{(n)})^T\,\vec{x}^{(n)} \;+\; {\rm Tr}\big(W^T_{\rm new}W_{\rm new}\,\langle\vec{z}\vec{z}^T\rangle_{q^{(n)}}\big) - 2\,(\vec{x}^{(n)})^T\,W_{\rm new}\,\langle\vec{z}\rangle_{q^{(n)}}\Big)\,, \tag{181}$$

which we have again expressed in terms of $\langle \vec{z} \rangle_{q_n}$ and $\langle \vec{z}\vec{z}^T \rangle_{q_n}$.

The E-step

The solution for the E-step is, as for all previous models, given by setting $q^{(n)}(\vec{z})$ equal to the posterior $p(\vec{z}\,|\,\vec{x}^{(n)}, \Theta)$ for all data points n. Using Bayes' rule to compute the posteriors we obtain (note that we again drop the superscript old for brevity):

$$p(\vec{z}\,|\,\vec{x}^{(n)}, \Theta) \;=\; \frac{p(\vec{x}^{(n)}\,|\,\vec{z}, \Theta)\; p(\vec{z}\,|\,\Theta)}{\int p(\vec{x}^{(n)}\,|\,\vec{z}', \Theta)\; p(\vec{z}'\,|\,\Theta)\, d\vec{z}'} \tag{182}$$

$$\;=\; \frac{\mathcal{N}(\vec{x}^{(n)}\,|\,W\vec{z}, \sigma^2 \mathbb{1})\; \prod_h \mathcal{C}(\vec{z}_h)}{\int \mathcal{N}(\vec{x}^{(n)}\,|\,W\vec{z}', \sigma^2 \mathbb{1})\; \prod_h \mathcal{C}(\vec{z}'_h)\, d\vec{z}'} \tag{183}$$

For the PCA generative model we could simplify firstly the denominator of the expression above and secondly the whole expression. We could finally derive the closed-form expressions (139) and (140) for the two moments. Unfortunately, the same is not possible for the SC generative model because the integral in the denominator cannot be reduced to a closed-form expression.

To proceed we have to come up with an approximation to the posterior, we do an *approximate E-step*. In sparse coding we simply replace the posterior distribution $p(z\,|\,\vec{x}^{(n)}, \Theta)$ by a Dirac delta defined using the maximum $\vec{z}_{\max}$ of this function:

$$p(\vec{z}\,|\,\vec{x}^{(n)}, \Theta) \;\approx\; \delta(\vec{z} - \vec{z}^{(n)}_{\max}) \;\text{ where }\; \vec{z}^{(n)}_{\max} \;=\; \mathrm{argmax}_{\vec{z}}\{p(\vec{z}\,|\,\vec{x}^{(n)}, \Theta)\} \tag{184}$$

Note that this approximations assumes a sharply peaked posterior distributions. Using this approximations (**??**) and (**??**) become:

$$\langle \vec{z} \rangle_{q_n} := \vec{z}^{(n)}_{\max}\,, \qquad \langle \vec{z}\vec{z}^T \rangle_{q_n} := \vec{z}^{(n)}_{\max}\, (\vec{z}^{(n)}_{\max})^T\,. \tag{185}$$

Unfortunately, this approximation results in parameters W that do not converge anymore. We have to counter-act for this with some explicit re-normalizations of W. If this is done appropriately (see, e.g., Olshausen/Field, 1996, for details) we recover the independent subspaces.

Note that the approximation in (184) is relatively crude. In fact there are more principled approaches to approximate the E-step when the exact solution cannot be found or if it is not computationally feasible. Common methods for E-step approximations are *variational Bayes* and *sampling* but there are many more. An E-step approximation is required for virtually any more complex multiple-cause generative model.

ICA. An algorithm familiar with Sparse Coding is the Independent Component Analysis (ICA) algorithm (Comon, 1994; Hyvärinen and Oja, 1997). It is typically applied in similar situations as Sparse Coding. ICA does in general seek directions in the data that are non-Gaussian (e.g., heavy tails), so it would find similar directions in data distributions as Sparse Coding. In particular, ICA also results in Gabor-like fields if trained on natural image patches. A relation between ICA and Sparse Coding is that one form of ICA is obtained if we let the noise assumption of ICA go to zero (Dayan and Abbott, 2001). So, without going into the details of the ICA algorithm, it can be thought of as a noiseless form of sparse coding.