A dog, a vegan flea, movie ratings, and the EM-algorithm

Carlos Cotrini
Department of Computer Science
ETH Zürich

March 25, 2019

1 Introduction

These notes motivate and present the EM (expectation-maximization) algorithm, an algorithm used for approximately computing parameter values for probability distributions in maximum-likelihood estimation. The reader is expected to have knowledge of undergraduate probability theory and to be familiar with maximum-likelihood estimation.

The presentation is divided in three parts. Section 2 presents an optimization problem regarding a vegan flea. We present an algorithm that finds an approximate solution and provides intuitions to understand why the EM-algorithm works. Section 3 presents the problem of building a simple movie recommendation system. We show that movie ratings can be understood as samples from a probabilistic model that is defined by a set of multivariate Bernoulli distributions. Estimating the parameters of these distributions via maximum-likelihood turns out to be very hard using analytical methods. In Section 4, we show that this maximum-likelihood estimation problem is an instance of the vegan-flea optimization problem and derive the EM-algorithm from the approximation algorithm presented in Section 2.

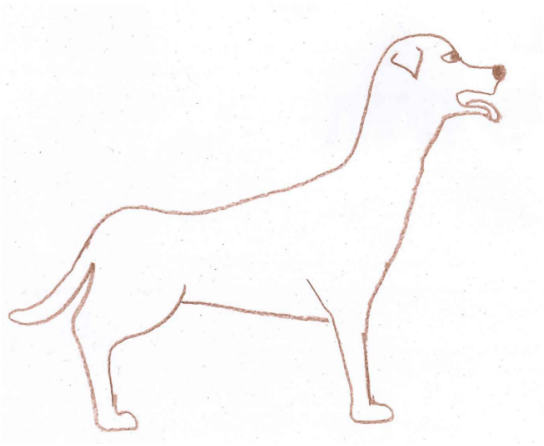
2 The vegan-flea optimization problem

2.1 The dog

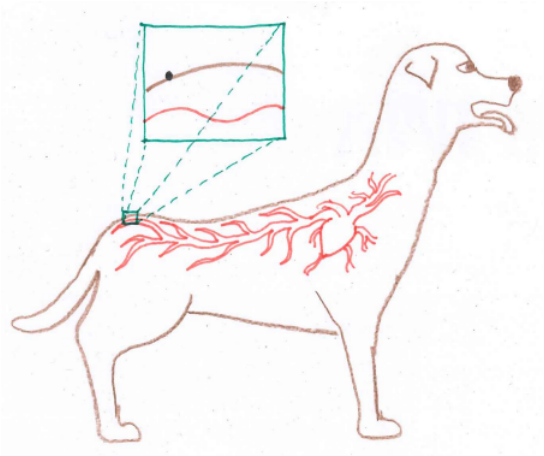
We introduce a two-dimensional dog, depicted in Figure 1a. Although, in practice, dogs are three-dimensional entities, a two-dimensional dog makes easier the presentation of some ideas later. Observe that this two-dimensional dog has only two legs, since two legs suffice to keep balance, and one eye, since there is no need for perspectives in a two-dimensional space.

Figure 1b shows some of the dog's cardiovascular system. Observe that a blood vessel of a two-dimensional being does not have the shape of a cilinder. They still naturally expand when a surge of blood flows from a heart's pump. For the rest of these notes we focus only on a small area of this figure; namely, the tiny green square shown in the figure.

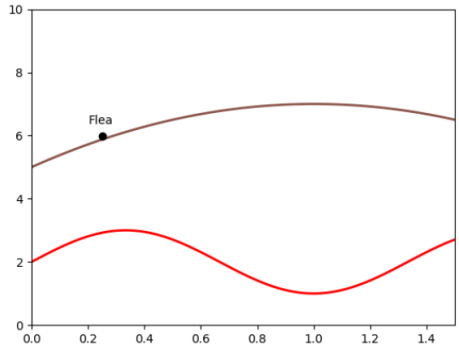
Figure 1c shows the area marked by the green square in detail. We have placed some Cartesian axes there for reference. There is a flea at coordinates (0.25, 6). The dog's



(a) A two-dimensional dog.



(b) A part of the dog's (two-dimensional) cardiovascular system.



(c) The flea, the skin, and the upper border of a blood vessel.

Figure 1

skin is the brown curve, and the upper border of a blood vessel is the red curve. Observe that, in a two-dimensional space, skins are lines instead of surfaces.

2.2 The skin and the blood vessel's upper border

We now mathematically model the skin and the blood vessel's upper border. Let $skin : [0,1] \times [0,\infty) \to \mathbb{R}$, and $vessel : [0,1] \times [0,\infty) \to \mathbb{R}$ be two functions. For $t \in [0,\infty)$, let $skin(\cdot,t) : [0,1] \to \mathbb{R}$ be the function that maps x to $skin(x,t)$. We define the function $vessel(\cdot,t)$ analogously. Intuitively, for $t \in [0,\infty)$, the functions $skin(\cdot,t)$ and $vessel(\cdot,t)$ describe the skin and the blood vessel's upper border at time t, as depicted in Figure 2. Hence, $skin(x,t) \geq vessel(x,t)$, for any $(x,t) \in [0,1] \times [0,\infty)$. Observe that $skin(\cdot,t)$ and $vessel(\cdot,t)$ vary with t. This is to model the fact that blood flows through the vessel and, consequently, makes the skin surface and the blood vessel's upper border vary with time. The animated .gif file attached with these notes illustrate the setting. We strongly encourage the reader to look the .gif file before proceeding.

Assumption 1. *We assume that for any $x \in [0,1]$ and any two time points $t_1, t_2 \in [0,\infty)$, $skin(x,t_1) - vessel(x,t_1) = skin(x,t_2) - vessel(x,t_2)$.*

This assumption states that the skin surface changes by the same amount that the blood vessel's upper border changes. This allows us to define a function d such that $d(x) = skin(x,t) - vessel(x,t)$, for any $x \in [0,1]$ and $t \in [0,\infty)$.

Blood flows periodically through the vessel and therefore makes the vessel shape change. Moreover, we assume the following:

Assumption 2. *For any $x \in [0,1]$ and any $t \in [0,\infty)$, there is $t' \geq t$ such that $vessel(x,t') = \max_{x'} vessel(x',t')$.*

If you observe the .gif animation, you can see that we defined a constant M. You can also see that, for any x and any t, $vessel(x,t) \leq M$ and that there is $t' \geq t$ such that $vessel(x,t') = M$. We could then make Assumption 2 stronger by stating that, for any $x \in [0,1]$ and any $t \in [0,\infty)$, $vessel(\cdot,t)$ is bounded by M and that there is $t' \geq t$ such that $vessel(x,t) = M$. However, Assumption 2 is enough for our purposes. Moreover, it is weaker and, hence, more general.

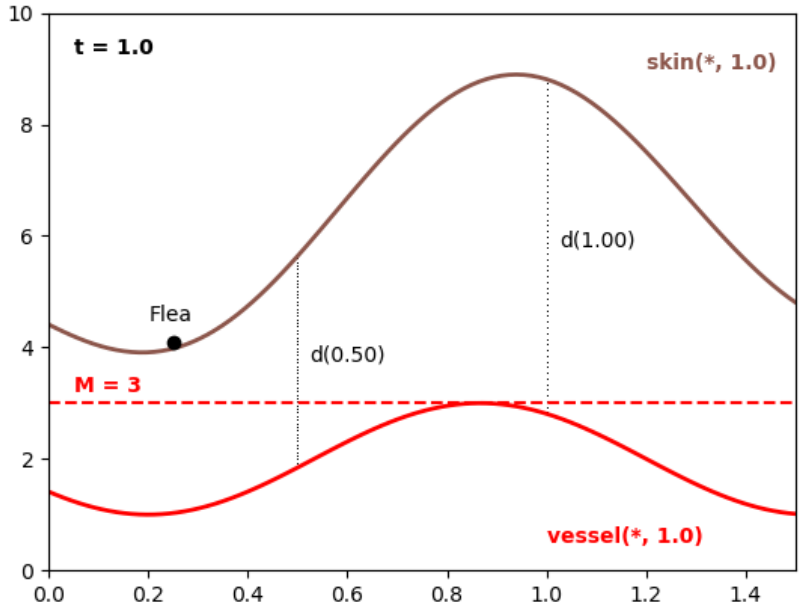
2.3 The vegan flea

Imagine now that there is a flea resting on the skin surface at $(x_0, skin(x_0,0))$, for some $x_0 \in [0,1]$. The flea has decided to become vegan and wishes to be as far away from the blood vessel as possible, to avoid the temptation of the blood. More precisely, the flea's goal is the following.

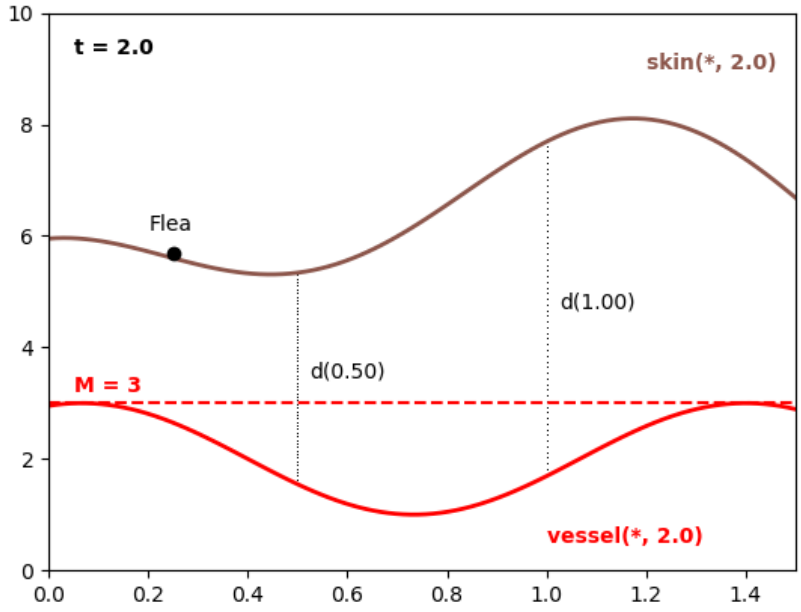
Objective 1. *Compute a value x^* that maximizes d.*

A look at the .gif file shows that $x^* = 1.0$. This is easy for us as we, three-dimensional creatures, have an omniscient view of the flea's universe. The flea, however, cannot see that as she knows nothing about *vessel*. In spite of this, we illustrate how the flea can partially achieve its objective.

We make two assumptions about the flea's computation abilities.



(a) Skin and blood vessel at $t = 1$



(b) Skin and blood vessel at $t = 2$

Figure 2: An illustration of the functions $skin(\cdot, t)$ and $vessel(\cdot, t)$, for $t \in \{1, 2\}$.

Assumption 3. *For any $t \in [0, \infty)$, the flea can efficiently compute*

$$x^* = \arg\max_{x'} \mathit{skin}(x, t).$$

This assumption bases on the idea that the flea can see the dog's skin and can therefore maximize $\mathit{skin}(\cdot, t)$. The next assumption states that the flea, located at $(x_0, (\mathit{skin}(x_0, 0)))$, can identify the moment t' when $\mathit{vessel}(x_0, t') = M$, the maximum of $\mathit{vessel}(\cdot, t)$.

Assumption 4. *For any $x \in [0, 1]$ and any $t \in [0, \infty)$, the flea can efficiently compute some $\hat{t} \geq t$ such that $\mathit{vessel}(x, \hat{t}) = \max_{x'} \mathit{vessel}(x', \hat{t})$.*

We give some justification for this assumption. Blood flows through the vessel in a periodic way and $\mathit{skin}(\cdot, t)$ changes in the same way as $\mathit{vessel}(\cdot, t)$ does. Hence, the flea can learn the blood pulse and then wait for a time $\hat{t}$ where $\mathit{vessel}(x_0, \hat{t}) = M$.

2.4 An approximate maximization algorithm

We describe a strategy by which the flea can compute a value x^* such that $d(x^*) \geq d(x_0)$, where $(x_0, \mathit{skin}(x_0, 0))$ is the flea's current position.

[**E-step**] The flea waits for a time $\hat{t}$ at which $\mathit{vessel}(x_0, \hat{t}) = \max_x \mathit{vessel}(x, \hat{t})$ (Figure 3a). This is possible by Assumption 4.

[**M-step**] At time $\hat{t}$, the flea computes a value x^* such that

$$x^* = \arg\max_{x} \mathit{skin}(x, \hat{t}).$$

(Figure 3b). This is possible by Assumption 3.

[**Move**] The flea moves to $(x^*, \mathit{skin}(x^*, \hat{t}))$ (Figure 3c).

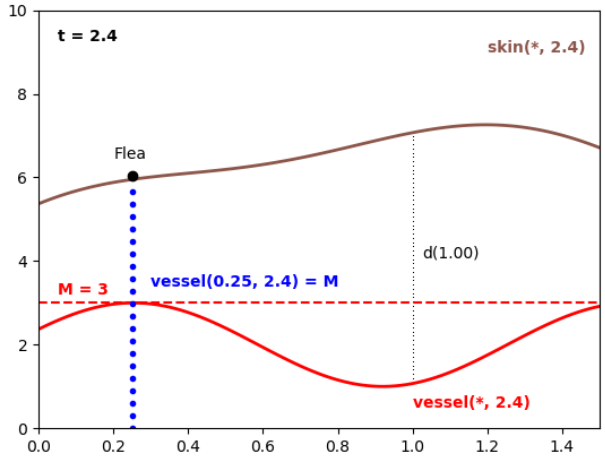
Figure 4 illustrates why $d(x^*) \geq d(x_0)$. Observe that $\mathit{skin}(x^*, \hat{t}) \geq \mathit{skin}(x_0, \hat{t})$, since x^* is a maximum of $\mathit{skin}(\cdot, \hat{t})$. Observe also that $\mathit{vessel}(x^*, \hat{t}) \leq \mathit{vessel}(x_0, \hat{t})$, since $\mathit{vessel}(x_0, \hat{t})$ is a maximum of $\mathit{vessel}(\cdot, \hat{t})$. Hence, $d(x^*) = \mathit{skin}(x^*, \hat{t}) - \mathit{vessel}(x^*, \hat{t}) \geq \mathit{skin}(x_0, \hat{t}) - \mathit{vessel}(x_0, \hat{t}) = d(x_0)$.

Notice that the flea can set $x_0 = x^*$ and repeat this procedure to find another value x^{**} such that $d(x^{**}) \geq d(x^*)$. The flea can repeat this process as long as the computed values increase d.

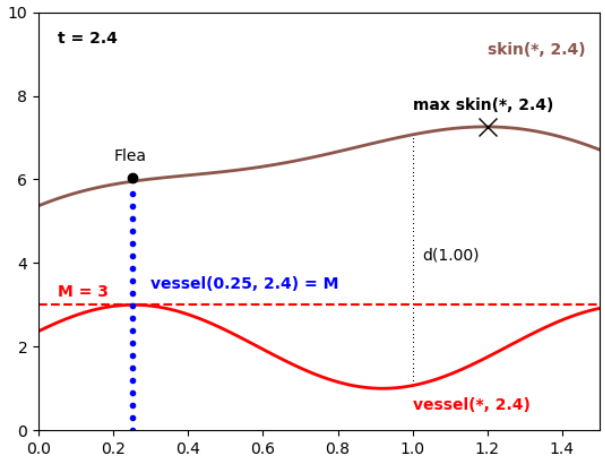
We summarize these insights into Algorithm 1, which computes a sequence of values $x_0, x_1, \ldots$ such that $d(x_0) \leq d(x_1) \leq \ldots$ Observe that this algorithm converges.

One can also relax the assumptions above so that the algorithm works even when *vessel* and *skin*'s domains are a product $\mathcal{X} \times \mathcal{T}$ of any two sets $\mathcal{X}$ and $\mathcal{T}$.

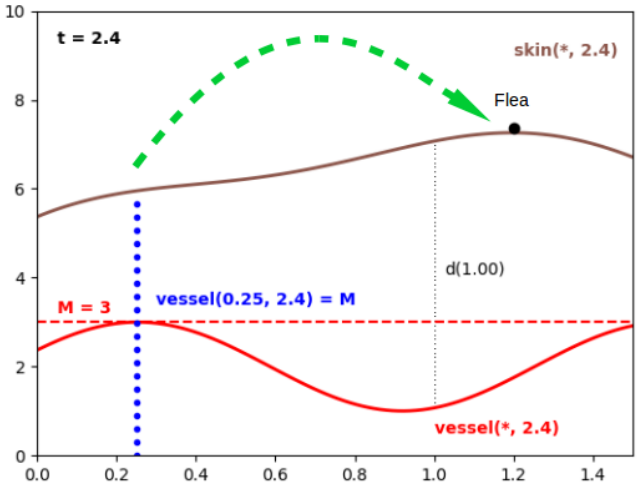
With an argumentation similar to the one above, one can prove that $d(x^*)$ never decreases between two iterations of Algorithm 1's loop.



(a) Step 1.



(b) Step 2.



(c) Step 3.

Figure 3

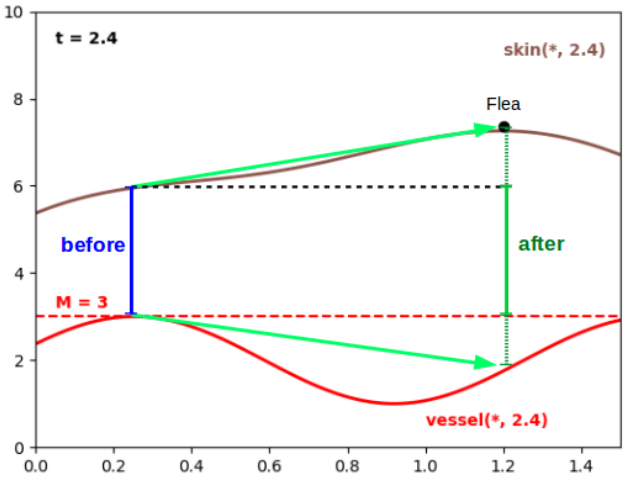


Figure 4

	Star Wars	Star Trek	Titanic	Pretty Woman	007	Mission Impossible
Alice	✓	✓	×	×	×	×
Bob	✓	✓	×	×	×	×
Carlos	×	×	✓	✓	×	×
David	×	×	✓	✓	×	×
Ellen	×	×	✓	×	×	✓
Fabian	×	×	✓	✓	×	×
Gabriel	✓	✓	×	×	✓	✓
Hector	✓	×	×	×	✓	✓
Ian	✓	✓	×	×	✓	✓
Zelya	✓	×	✓	×	×	✓
John	?	?	?	✓	?	?

Table 1: Ratings from 10 individuals for 6 movies. According to the table, everyone who likes Pretty Woman also liked Titanic. Therefore, it is likely that John would also like Titanic.

Algorithm 1

Require: $\mathcal{X}$ and $\mathcal{T}$ two sets and real functions d, *vessel*, and *skin* satisfying the following.

A1 $d(x) = skin(x,t) - vessel(x,t)$, for any $x \in \mathcal{X}$ and $t \in \mathcal{T}$.

A2 For any $x \in \mathcal{X}$, one can efficiently compute $\hat{t} \in \mathcal{T}$ such that $vessel(x,\hat{t}) = \max_{x'} vessel(x',\hat{t})$.

A3 For any $t \in \mathcal{T}$, one can efficiently compute $\arg\max_x skin(x,t)$.

function DISTANCEMAX($vessel : \mathcal{X} \times \mathcal{T} \to \mathbb{R}$, $skin : \mathcal{X} \times \mathcal{T} \to \mathbb{R}$)
  Choose any $x_0 \in \mathcal{X}$.
  for $i = 0, 1, \ldots$ **do**
    [**E-step**] Compute $\hat{t} \in \mathcal{T}$ s.t. $vessel(x_0,\hat{t}) = \max_x vessel(x,\hat{t})$.
    [**M-step**] Compute $x^* = \arg\max_x skin(x,\hat{t})$.
    Print x^* and $d(x^*)$.
    $x_0 \leftarrow x^*$.
  end for
end function

3 Building a movie-recommendation system with a mixture of multivariate Bernoulli distributions

Table 1 shows the ratings that 10 (fictitious) individuals gave to 6 popular movies. To keep it simple, we assume only binary ratings (good or bad). After a close look to the ratings, the reader can see that Alice and Bob have the exact same taste for movies: *sci-fi* movies. The next four people have a strong interest for *romantic* movies. The next three people like sci-fi and action movies. Zelya's tastes seem to be different from everyone else.

Consider now John, he likes Pretty Woman, but does not like Star trek. He has not seen any of the other movies. What movie could we recommend to him? Since he likes Pretty Woman and everyone who liked Pretty Woman also liked Titanic, we can recommend him to watch Titanic.

From all 2^6 combinations of ratings, the table contains only a few of them. Moreover, a large majority of the people in the table seem to belong to one of very few taste categories: sci-fi, romantic, or sci-fi+action. Real life is not so different: a large majority of people can partitioned into very few categories and people within a same category have very similar preferences. To recommend a movie to someone, we estimate the category where this person belongs and then search for a movie that people in this category liked.

3.1 Movie ratings as samples from probability distributions

We can view Table 1 as the result of a sampling process. Initially, the table was empty and then one person at random appeared (Alice, in our case) and filled the first row of the table. Then Bob appeared and so on. To sample the film ratings of one person, we first *sample the category* where this person belongs and then, for each movie, we *sample the rating* this person gave to that movie, conditioned on the person belonging to the sampled category. This sampling process is then defined by the following probability distributions:

- A distribution over categories.
- For each category and each movie, a distribution defining the probability that a person in the category likes the movie.

From these two distributions, we build a new probability distribution with which we can answer the following question: *if a person watched and liked movies m_1, m_2, ..., and m_k, how likely is that she will like a movie m' that she has not seen?* This probability distribution constitutes then our recommendation system. To decide which movie to recommend, we take the ratings the person has given to previously watched movies. Then, for each movie in the database she has not seen, we compute the probability that she likes that movie. Finally, we recommend the movie that she will most likely like.

We first formalize the two distributions mentioned above. Suppose we have K categories and D movies. We identify categories with the numbers 1, 2, ..., K and movies with the numbers 1, 2, ..., D. We can model the distribution of K categories using a discrete distribution with a set $\{\nu_1, \nu_2, \ldots, \nu_K\}$ of K parameters that add up to 1. For the k-th category and the movie j, we define a value μ_{kj} indicating what the probability is that a person in the k-th category likes movie j. We leave the values ν_k and μ_{kj}, for $k \leq K$ and $j \leq D$, undefined for the moment.

Having defined these distributions, we can now assign a probability to the ratings a person gave to all movies in the database. We model these ratings with a vector $x \in \{0,1\}^D$, where x_j, for $j \leq D$, indicates whether the movie was rated good ($x_j = 1$) or bad ($x_j = 0$). We leave as an exercise to show that the probability $p(x)$ of a vector $x \in \{0,1\}^D$ is as follows:

$$p(x) = \sum_{k \leq K} \left(\nu_k \prod_{j \leq D} \mu_{kj}^{x_j} \left(1 - \mu_{kj}\right)^{1 - x_j} \right). \tag{1}$$

We can now define probabilities for movie-rating databases. We model a movie-rating database with a matrix $X \in \{0,1\}^{N \times D}$. Each row X_i, for $i \leq N$, represents a person and each entry $X_{i,j}$, for $j \leq D$, represents the rating person i gave to movie j. Assuming that the ratings of two different people are independent, we can show that the probability $p(X)$ is given by the following.

$$p(X) = \prod_{i \leq N} p(X_i) = \prod_{i \leq N} \sum_{k \leq K} \left(\nu_k \prod_{j \leq D} \mu_{kj}^{X_{i,j}} \left(1 - \mu_{kj}\right)^{1 - X_{i,j}} \right). \tag{2}$$

3.2 Maximum-likelihood estimation

We now choose values for ν_k and μ_{kj}, for $k \leq K$ and $j \leq D$. Here, we use *maximum-likelihood estimation*, which argues that the best values for our parameters are those that maximize $p(X)$, for X the movie database we have. For computational reasons, one searches instead for the parameters that maximize $\log p(X)$. Using basic logarithm properties, we can show that

$$\log p(X) = \sum_{i \leq N} \log \left(\sum_{k \leq K} \left(\nu_k \prod_{j \leq D} \mu_{kj}^{x_j} \left(1 - \mu_{kj}\right)^{1 - x_j} \right) \right). \tag{3}$$

The value $\log p(X)$ is called *X's log likelihood*.

Finding the parameter values that maximize this log likelihood is difficult, even with approximation methods. Nonetheless, it is possible to find a set of parameter values that locally maximize the log likelihood, using Algorithm 1. To do this, we introduce a new set $\mathbf{Z} = \{\mathbf{Z}(i) \mid i \leq N\}$ of random variables. For $i \leq N$, $\mathbf{Z}(i)$ indicates to which category person i belongs. Assume for a moment that, in addition of X, we also know, for $i \leq N$, the category $Z(i)$ where person i belongs. One can show that (X, Z)'s log likelihood is given by the following.

$$\log p(X, Z) = \sum_{i \leq N} \begin{pmatrix} \log \nu_{Z(i)} + \\ x_{i,Z(i)} \log \mu_{Z(i),j} + \\ \left(1 - x_{i,Z(i)}\right) \log \left(1 - \mu_{Z(i),j}\right) \end{pmatrix}. \tag{4}$$

The log likelihood of (X, Z) is much easier to maximize with respect to the parameters than X's log likelihood; it can be maximized using standard calculus. However, observe that the movie database does not tell us to which category each person belongs, so we do not know Z and there is no clear way how to obtain it.

It is common that log likelihood maximization problems become easier when we introduce additional random variables to the probabilistic model. It is also common that the values that such random variables take are not available. For these situations, the EM-algorithm was proposed.

4 Derivation of the EM-algorithm

We generalize the problem we addressed in the previous section. Let $\mathbf{X}$ and $\mathbf{Z}$ be random variables and let X be an observed value for $\mathbf{X}$. Assume that the joint pdf $p(\cdot, \cdot \mid \theta)$ for $(\mathbf{X}, \mathbf{Z})$ is parameterized by θ, which can take values in Θ. Our goal is to compute

$$\arg\max_{\theta \in \Theta} \log p(X \mid \theta). \tag{5}$$

Assume now that it is preferrable to work with $\log p(X, Z)$ than with $\log p(X)$, for any Z in $\mathbf{Z}$'s range. If we knew the value Z that $\mathbf{Z}$ took when we obtained $\mathbf{X} = X$, then we could state that

$$\log p(X \mid \theta) = \log p(X, Z \mid \theta) - \log p(Z \mid X, \theta). \tag{6}$$

This identity follows from the definition of conditional pdfs. We can make Z irrelevant by computing expectations on both sides with respect to some pdf $\tilde{p}$ for $\mathbf{Z}$. We leave for later the problem of defining $\tilde{p}$.

$$\begin{aligned} \int \tilde{p}(Z) \log p(X \mid \theta) dZ &= \int \tilde{p}(Z) \log p(X, Z \mid \theta) dZ - \int \tilde{p}(Z) \log p(Z \mid X, \theta) dZ \\ &= \mathbb{E}_{\tilde{p}(\mathbf{Z})} \left[\log p(X, \mathbf{Z} \mid \theta)\right] - \mathbb{E}_{\tilde{p}(\mathbf{Z})} \left[\log p(\mathbf{Z} \mid X, \theta)\right]. \end{aligned}$$

Observe that $\log p(X \mid \theta)$ does not depend on Z or $\tilde{p}$. Therefore, the left-hand side equals $\log p(X \mid \theta)$. As a result,

$$\log p(X \mid \theta) = \mathbb{E}_{\tilde{p}(\mathbf{Z})} \left[\log p(X, \mathbf{Z} \mid \theta)\right] - \mathbb{E}_{\tilde{p}(\mathbf{Z})} \left[\log p(\mathbf{Z} \mid X, \theta)\right]. \tag{7}$$

We now show that the maximization problem in Equation 5 is an instance of the vegan-flea problem. Let $\mathcal{X} = \Theta$ and $\mathcal{T}$ be the set of all pdfs for $\mathbf{Z}$. For $\theta \in \Theta$ and $\tilde{p} \in \mathcal{T}$,

let

$$\begin{aligned} d(\theta) &= \log p(X \mid \theta) \\ skin(\theta, \tilde{p}) &= \mathbb{E}_{\tilde{p}(\mathbf{Z})} \left[\log p(X, \mathbf{Z} \mid \theta)\right] \\ vessel(\theta, \tilde{p}) &= \mathbb{E}_{\tilde{p}(\mathbf{Z})} \left[\log p(\mathbf{Z} \mid X, \theta)\right]. \end{aligned}$$

We now derive sufficient conditions for the assumptions **[A1]**, **[A2]**, and **[A3]** to hold.

A1 This assumption follows from Equation 7, so no condition is necessary.

A2 In our case, this assumption means the following: for any $\theta \in \Theta$, one can efficiently compute $\tilde{p} \in \mathcal{T}$ such that for any $\theta' \in \Theta$,

$$\mathbb{E}_{\tilde{p}(\mathbf{Z})} \left[\log p(\mathbf{Z} \mid X, \theta)\right] \geq \mathbb{E}_{\tilde{p}(\mathbf{Z})} \left[\log p(\mathbf{Z} \mid X, \theta')\right]. \tag{8}$$

We can fulfill this inequality by setting $\tilde{p}\,(\mathbf{Z}) = p(\mathbf{Z} \mid X, \theta)$. This follows from Gibbs's inequality, which states that for any two pdfs p and q for a random variable $\mathbf{Z}$,

$$\mathbb{E}_{p(\mathbf{Z})} \left[\log p(\mathbf{Z})\right] \geq \mathbb{E}_{p(\mathbf{Z})} \left[\log q(\mathbf{Z})\right]. \tag{9}$$

Hence, for **[A2]** to hold, we require the pdf $p(\mathbf{Z} \mid X, \theta)$ *to be efficiently computable.*

A3 This assumption requires that, for any $\tilde{p} \in \mathcal{T}$, we can efficiently compute

$$\arg\max_{\theta \in \Theta} \mathbb{E}_{\tilde{p}(\mathbf{Z})} \left[\log p(X, \mathbf{Z} \mid \theta)\right].$$

In summary, to apply Algorithm 1 to compute $\arg\max_\theta \log p\,(X \mid \theta)$, we require the following.

AE1 One can efficiently compute the pdf $p(\mathbf{Z} \mid X, \theta)$.

AE2 One can efficiently compute $\arg\max_{\theta \in \Theta} \mathbb{E}_{\tilde{p}(\mathbf{Z})} \left[\log p(X, \mathbf{Z} \mid \theta)\right]$, for any pdf $\tilde{p}$ for $\mathbf{Z}$.

Instantiating Algorithm 1 to our particular problem, we obtain *the EM algorithm.*

Algorithm 2

Require:

- Θ a set of parameters.
- A joint pdf $p(\mathbf{X}, \mathbf{Z} \mid \theta)$ over two random variables $\mathbf{X}$ and $\mathbf{Z}$, governed by a parameter θ that ranges over Θ.
- A value X in $\mathbf{X}$'s range.

AE1 One can efficiently compute the pdf $p(\mathbf{Z} \mid X, \theta)$.

AE2 One can efficiently compute $\arg\max_{\theta \in \Theta} \mathbb{E}_{\tilde{p}(\mathbf{Z})} \left[\log p(X, \mathbf{Z} \mid \theta)\right]$, for any pdf $\tilde{p}$ for $\mathbf{Z}$.

1: **function** EM(X, $p(\mathbf{X}, \mathbf{Z} \mid \theta)$, Θ)
2: Choose any $\theta_0 \in \Theta$.
3: **for** $i = 0, 1, \ldots$ **do**
4: **[E-step]** Compute $p(\mathbf{Z} \mid X, \theta_i)$.
5: **[M-step]** Compute $\theta_{i+1} = \arg\max_{\theta} \mathbb{E}_{p(\mathbf{Z} \mid X, \theta_i)} \left[\log p(X, \mathbf{Z} \mid \theta)\right]$.
6: Print θ_{i+1} and $\log p(X \mid \theta_{i+1})$.
7: **end for**
8: **end function**