# PGMs session 8 Learning: maximum likelihood estimation

We now have a fair idea of how to do *inference* in a graphical model. We have far from exhausted the topic, especially for cases where there are loops in the model, or the marginalisation integrals are just not feasible. However, we will now move on to *learning* in PGMs.

To do inference, we need a PGM and the corresponding parameters it depends on. For now, we will assume that the structure of the PGM is known. We therefore now turn to investigate methods to determine the parameters of the model. We first consider maximum likelihood (ML) estimation, which is probably the simplest approach we can take. This method sets the model parameters $\boldsymbol{\Theta}$ in such a way that the total likelihood of the data given the parameters is maximised; i.e., it finds

$$\boldsymbol{\Theta}^{\mathrm{ML}} = \underset{\boldsymbol{\Theta}}{\operatorname{argmax}}\ p(\mathcal{D}|\boldsymbol{\Theta}). \tag{1}$$

This is usually done by finding where the derivative of the likelihood function is equal to zero. One can obtain the same result by calculating

$$\boldsymbol{\Theta}^{\mathrm{ML}} = \underset{\boldsymbol{\Theta}}{\operatorname{argmax}}\ \log p(\mathcal{D}|\boldsymbol{\Theta}), \tag{2}$$

since $\log(\cdot)$ is a strictly monotonic function – this formulation is often more convenient to use. Note that $\boldsymbol{\Theta}$ (and not $\mathcal{D}$) is the argument to $p$, changing it from a probability distribution to a *likelihood* function. In spite of its simplicity, this technique is very useful and often forms a building block of more involved techniques.

## 1 Preparation: Watch the Koller videos on ML estimation

Watch the following videos in the Coursera course Probabilistic Graphical Models 1: Representation, Week 2:

- *Template Models: Plate Models (20:08)*

Watch the following videos in the Coursera course Probabilistic Graphical Models 3: Learning, Week 1:

- *Learning: Overview: Learning: Overview (15:35)*
- *ML-class Revision (Optional)* (by Andrew Ng) from *Regularization: The Problem of Overfitting (9:42)* to *Regularization and Bias Variance (11:20)*
- *Maximum Likelihood Parameter Estimation in BNs: Maximum Likelihood Estimation (14:59)*
- *Maximum Likelihood Parameter Estimation in BNs: Maximum Likelihood Estimation for Bayesian Networks (15:49)*

Watch the following videos in the Coursera course Probabilistic Graphical Models 3: Learning, Week 2:

- *Parameter Estimation in MNs: Maximum Likelihood for Log-Linear Models (28:47)*
- *Parameter Estimation in MNs: Maximum Likelihood for Conditional Random Fields (13:24)*

**Notes on these videos:**

- *Template Models: Plate Models (20:08)* introduces the concept of a "plate" in a graphical model, a mechanism used for repeating parts of the model. We use it here to model repeated data points.
- *Learning: Overview: Learning: Overview (15:35)* gives an overview of learning in PGMs. In this module, we will assume that the model *structure* is known, and we only try to learn the model *parameters*. We will initially focus on the complete data case (Week 4b and 5a) and end with the incomplete data case (Week 5b). The video discusses a number of machine learning concepts that should be familiar to you (training/validation/test data, overfitting, regularisation, model complexity), as well as the difference between learning in PGMs vs. other machine learning techniques.

- The *ML-class Revision (Optional)* videos cover machine learning concepts that should be familiar to you at this point: bias and variance of a model, regularisation, model selection, and training/validation/testing data. If you are familiar with these concepts, you may want to skip these videos. A note on regularisation: Quite often a training procedure can misuse the parameters of a model with many degrees of freedom, resulting in it specialising (overfitting) on the training data. Regularisation is a technique used to counter this. As discussed in these videos by Andrew Ng, it appears as a type of ad-hoc trick, but its origin (and much more flexibility in ways to use it) will become clear after we studied Bayesian estimation at a later stage.
- *Maximum Likelihood Parameter Estimation in BNs: Maximum Likelihood Estimation (14:59)* covers the basics of MLE, and *Maximum Likelihood Parameter Estimation in BNs: Maximum Likelihood Estimation for Bayesian Networks (15:49)* applies it to BNs. With fully observed data, the estimates decomposes nicely. For this practical assignment, we will apply MLE (only) to BNs – make sure you understand the concepts in these two videos well.
- *Parameter Estimation in MNs: Maximum Likelihood for Log-Linear Models (28:47)* and *Parameter Estimation in MNs: Maximum Likelihood for Conditional Random Fields (13:24)* apply MLE to undirected models, specifically MRFs and CRFs. Note that the presence of the normalizing partition function, $Z(\boldsymbol{\Theta})$, couples the parameters and thus confounds the nice decomposing property we saw with BNs. However, the log-likelihood function is concave and we can optimise it via gradient techniques. We will not do parameter estimation for undirected models in this module; however, it is important that you understand why it is more difficult to do than for BNs, and you should have an overview understanding of the learning process.

# 2 Preparation: Read sections of Barber Chapters 8 and 9 on maximum likelihood estimation

Read the following sections of Barber Chapter 8 about statistics for machine learning:

- Section 8.1: Take note of the different forms of data encoding, especially the 1-of-$m$ encoding in Subsection 8.1.1.
- Section 8.2: You should already be familiar with the concepts in the first half of this section. Take note of:
  - The empirical distribution describes the observed data and has use in parameter estimation. Note the use of Kronecker and Dirac delta functions to cover discrete and continous spaces respectively.
  - The Kullback-Leibler divergence measures the "difference" between two distributions. It plays an important role in some objective functions.
- Sections 8.3-8.5: These sections contain a mixture of familiar concepts and concepts that you do not need for this assignment.
- Section 8.6: Although this section contains both ML and Bayesian parameter estimation techniques (we'll get to the latter in Week 5a), the perspective of learning given by this section is important.
- Section 8.7 discusses the properties of ML estimation and is important to understand. ML estimation is equivalent to minimising the Kullback-Leibler divergence between the empirical distribution and the distribution that we want to fit to the data. We will later see that fitting one distribution to another via the Kullback-Leibler distance, also has other uses in estimation.
- Section 8.8 discusses learning the parameters of a Gaussian distribution. ML estimation of a Gaussian distribution (Subsection 8.8.1) is the important part to understand (for now).

Read the following sections of Barber Chapter 9 about learning as inference:

- Section 9.3: This section about ML parameter estimation for BNs is important – it is used in both questions of the practical assignment.
- Section 9.6 discusses ML parameter estimation for undirected models – you should take note of it, but we will not use it in this module.

# 3 Practical: MLE for the Hamming (7,4) error-correction code

For this question, we revisit the familiar Hamming (7,4) error-correction problem, but we now assume that the following parameters are unknown:

- The probability that any transmitted bit $b_i$ will "flip" to form the received bit $r_i$ is an unknown parameter, $\theta_e$. Note that we use the same parameter for all transmitted/received bits. The probability that any bit will not flip is therefore $1 - \theta_e$.
- The way in which the check bits, $b_4, \ldots, b_6$, are generated given the data bits, $b_0, \ldots, b_3$, is unknown. For example, the encoding scheme could use even parity or odd parity, but we do not know which. The parameters of the probability distributions, $p(b_4|b_0, b_1, b_2)$, $p(b_5|b_0, b_2, b_3)$ and $p(b_6|b_0, b_1, b_3)$ – i.e., $\boldsymbol{\theta}_{b_4|b_0,b_1,b_2}$, $\boldsymbol{\theta}_{b_5|b_0,b_2,b_3}$ and $\boldsymbol{\theta}_{b_6|b_0,b_1,b_3}$ respectively – are therefore unknown. Note that there are 8 parameters for each table, since e.g. $\theta_{b_4=1|b_0=0,b_1=0,b_2=0} = 1 - \theta_{b_4=0|b_0=0,b_1=0,b_2=0}$.

The objective of this question is to use maximum likelihood estimation to learn the parameter for of this model. Since we do not need `emdw`'s functionality for this question, you do not have to use C++ for coding.

(a) (Paper:) Draw the BN (not an MRF!) for the model with unknown parameters and for $M$ data points, using plate notation. Remember that the variable $\theta_e$ is shared among all the factors over the received bits.

(b) (Code:) Choose (actual) values for the parameters $\boldsymbol{\Theta} = (\theta_e, \boldsymbol{\theta}_{b_4|b_0,b_1,b_2}, \boldsymbol{\theta}_{b_5|b_0,b_2,b_3}, \boldsymbol{\theta}_{b_6|b_0,b_1,b_3})$. Now draw $M$ i.i.d. data points from this model. Note that a single data point consist of a combination of values for *all* the RVs in the model; i.e., $\mathbf{d}^{[m]} = (b_0^{[m]}, \ldots, b_6^{[m]}, r_0^{[m]}, \ldots, r_6^{[m]})$. Do the sampling in stages: first sample the RVs without parent – i.e., the data bits $b_0^{[m]}, \ldots, b_3^{[m]}$ – then sample the check bits, and lastly sample the received bits according to the conditional distribution $p(r_i|b_i)$. Gather the data points in an appropriate data structure, $\mathcal{D} = (\mathbf{d}^{[1]}, \ldots, \mathbf{d}^{[M]})$.

(c) (Paper:) Calculate the likelihood function, $p(\mathcal{D}|\boldsymbol{\Theta})$, decompose it into a number of local likelihood functions, and write each in terms of the unknown parameters and the number of each type of observations. For example, the local likelihood function for the received bits, $\prod_{m=1}^{M}\prod_{i=0}^{6} p(r_i|b_i : \theta_e)$, should only be written in terms of $\theta_e$, $\#(r_i \neq b_i)$ and $\#(r_i = b_i)$.

(d) (Paper:) Find the parameters that maximise the likelihood (in terms of the number of each type of observation) by solving for $\frac{\partial}{\partial \theta_k} \log p(\mathcal{D}|\boldsymbol{\Theta}) = 0$ for each parameter $\theta_k \in \boldsymbol{\Theta}$.

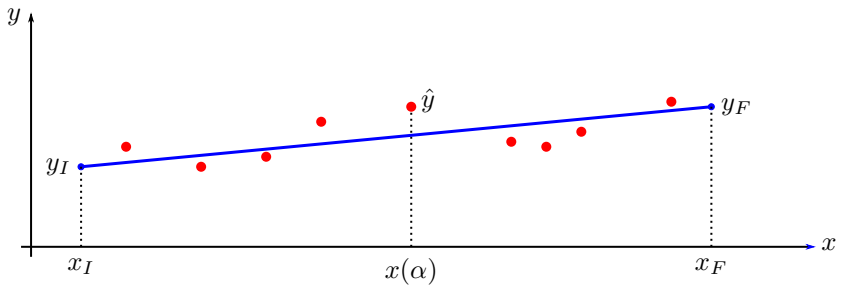
(e) (Code:) Now process your sampled data points to calculate the ML parameter estimates, $\boldsymbol{\Theta}^{\text{ML}}$. Compare the estimated parameters with the actual parameters.

(f) (Code:) Repeat the sampling of data points and parameter estimation, but vary the number of data points. What effect does the number of data points have on the accuracy of the parameter estimates? Under which conditions is it not possible to estimate certain parameters, and why does this happen?

# 4 Practical: MLE for a simple regression example

In this question, we'll use the Gaussian concepts of Week 4a to fit a line to a number of data points using maximum likelihood estimation. We'll also revisit this problem in Week 5a and Week 5b.

Consider the following setup, where a number of noisy measurements (red) are taken of points on a line (blue).




The two endpoints of the line are given by the coordinates $(x_I, y_I)$ and $(x_F, y_F)$. The $x$-coordinate of a measurement, $x$, is known, and it is parameterised by $\alpha \in [0, 1]$ as follows:

$$x = (1 - \alpha)\, x_I + \alpha x_F. \tag{3}$$

A measurement $\hat{y}$ is modelled as the $y$-coordinate of the line at $x$ plus zero-mean Gaussian noise, or

$$\hat{y} = (1-\alpha)y_I + \alpha y_F + v, \tag{4}$$

where $v \sim \mathcal{N}\left(0, \sigma_v^2\right)$. The $x$-coordinates of the endpoints, $x_I$ and $x_F$, are known. However, the $y$-coordinates of the endpoints, $y_I$ and $y_F$, are unknown and are the parameters we are trying to estimate; that is,

$$\boldsymbol{\Theta} = (y_I, y_F). \tag{5}$$

There are $M$ measurements, and the data is therefore given by

$$\mathcal{D} = \left(\hat{y}^{[1]}, \ldots, \hat{y}^{[M]}\right). \tag{6}$$

The objective of this question is to find the most likely line, $\boldsymbol{\Theta}$, given the measurements, $\mathcal{D}$. Since you again do not need `emdw`'s functionality for this question, you do not need to use C++.

(a) (Paper:) Draw the BN for this model with $M$ measurements using plate notation.

(b) (Paper:) Set up $p\left(\hat{y}^{[m]}|y_I, y_F\right)$, and then use it to construct $p\left(\mathcal{D}|\boldsymbol{\Theta}\right)$. Now set up $\log p\left(\mathcal{D}|\boldsymbol{\Theta}\right)$ and manipulate it until it is written as a sum of quadratic terms i.t.o. $\hat{y}^{[m]}$, $y_I$ and $y_F$ (and which also includes the known parameters $\sigma_v^2$ and $\alpha^{[m]}$).

(c) (Paper:) Derive $\boldsymbol{\Theta}^{\text{ML}} = \underset{\boldsymbol{\Theta}}{\text{argmax}} \log p(\mathcal{D}|\boldsymbol{\Theta})$ by using the following procedure: Calculate $\frac{\partial}{\partial y_I} \log p(\mathcal{D}|\boldsymbol{\Theta}) = 0$ and manipulate it until you can write it in the form

$$\mathbf{a}_1^T \begin{bmatrix} y_I \\ y_F \end{bmatrix} = b_1, \tag{7}$$

where $\mathbf{a}_1$ and $b_1$ are written in terms of known quantities. Similarly, calculate $\frac{\partial}{\partial y_F} \log p(\mathcal{D}|\boldsymbol{\Theta}) = 0$ and write it as

$$\mathbf{a}_2^T \begin{bmatrix} y_I \\ y_F \end{bmatrix} = b_2. \tag{8}$$

Now combine Equations 7 and 8 into a single matrix equation and solve for $\boldsymbol{\Theta}^{\text{ML}}$.

(d) (Code:) Generate a number of measurements by first uniformly sampling $\alpha^{[m]}$ from the interval $[0, 1]$ and then sampling $\hat{y}^{[m]}$ according to Equation 4 (choose an appropriate measurement noise variance $\sigma_v^2$). Plot both the measurements and the actual line.

(e) (Code:) Calculate $\boldsymbol{\Theta}^{\text{ML}}$ for the data generated in (d) and plot the estimated line on the same plot as the measurements and actual line.

(f) (Code:) Repeat the sampling of measurements and subsequent parameter estimation, but vary the number of measurements. What effect does the number of measurements have on the accuracy of the line fit? What is the lowest number of measurements you can use to fit a line?

(g) (Code:) Repeat the sampling of measurements and subsequent parameter estimation, but vary the measurement noise covariance $\sigma_v^2$ while keeping the number of measurements fixed. Does the measurement noise influence the parameter estimation calculations? How does the measurement noise influence the accuracy of the line fit?