PGMs session 10 Learning: latent variables

In our approach to learning the parameters of a model, we have assumed that the data is complete, which means that each data point contains a value for *every* RV in the model. However, in many scenarios, there are RVs that are never observed – these RVs are called *latent* RVs. For this section of the module, we look at how to learn the parameters of a model with incomplete data.

1 Preparation: Watch the Koller videos on learning with incomplete data

Watch the following videos in the Coursera course Probabilistic Graphical Models 3: Learning, Week 4:

- *Learning With Incomplete Data: Learning With Incomplete Data – Overview (21:34)*
- *Learning With Incomplete Data: Expectation Maximization – Intro (16:17)*
- *Learning With Incomplete Data: Analysis of EM Algorithm (11:34)*
- *Learning With Incomplete Data: EM in Practice (11:17)*
- *Learning With Incomplete Data: Latent Variables (22:00)*

Notes on these videos:

- The first situation leading to incomplete data is when some of our training data is missing. The missing-at-random (MAR) principle has to apply to enable valid parameter estimates. Although of great practical importance in semi-supervised systems, we will not concern us much with this.
- Quite often having latent states as part of our model, can make it much more powerful. Examples of this is the active basis PDF in a Gaussian mixture model (GMM, see Barber Chapter 20), the active state in a hidden Markov model (HMM), or in a temporal CRF. However, the presence of latent RVs couples our parameters and causes our parameter space to have multiple/local optima.
- We can estimate the parameters via the EM algorithm or directly using standard optimisation techniques. For this module, we focus on the EM algorithm.
- Koller discusses the EM algorithm in terms of parameter learning for discrete RVs. This perspective might help you to understand the concepts; however, Koller's description of the EM algorithm will not help you much to solve Question 4. You can find a general formulation of the EM algorithm in Section 3 below, which will be more helpful for Question 4.

2 Preparation: Read sections of Barber Chapters 11 on learning with hidden data

Read the following section from Barber Chapter 11:

- Section 11.1: This section discusses hidden (latent) variables and missing data, which covers the same ground as Koller, but is important to understand.
- Section 11.2: Barber presents a more general formulation of the EM algorithm, which is the same (except for notation) as the one given in Section 3 below. This section is important to understand.
- Section 11.3, 11.4 and 11.6 are useful concepts to know of, but we will not use them in this module.
- Section 11.5 discusses a more advanced topic – you can ignore it.

3 Information: Overview of the EM algorithm

To help you bring the concepts related to the EM algorithm together, here is an overview of the derivation and resulting algorithm.

We divide the variables in our model into three groups: $\mathcal{D}$ are the observed variables, $\mathcal{H}$ the latent/hidden variables and $\boldsymbol{\Theta}$ the parameters. We want to maximise:

$$p(\mathcal{D}|\boldsymbol{\Theta}) = \sum_{\mathcal{H}} p(\mathcal{D}, \mathcal{H}|\boldsymbol{\Theta}). \tag{1}$$

Let $q(\mathcal{H})$ be an arbitrary distribution over the latent variables. We can easily show that :

$$\log p(\mathcal{D}|\boldsymbol{\Theta}) = \mathcal{L}(q, \boldsymbol{\Theta}) + \mathrm{KL}(q \parallel p), \tag{2}$$

where

$$\begin{aligned} \mathcal{L}(q, \boldsymbol{\Theta}) &\triangleq \sum_{\mathcal{H}} q(\mathcal{H}) \log \left[\frac{p(\mathcal{D}, \mathcal{H}|\boldsymbol{\Theta})}{q(\mathcal{H})} \right], \\ \mathrm{KL}(q \parallel p) &\triangleq -\sum_{\mathcal{H}} q(\mathcal{H}) \log \left[\frac{p(\mathcal{H}|\mathcal{D}, \boldsymbol{\Theta})}{q(\mathcal{H})} \right]. \end{aligned} \tag{3}$$

(You can verify this by substituting and then simplifying while remembering that $q(\mathcal{H})$ is a distribution.) Note that $\mathcal{L}(q, \boldsymbol{\Theta})$ is a *functional* (i.e., it has an unspecified *function* q as parameter to optimise over) and $\mathrm{KL}(q \parallel p)$ is the Kullback-Leibler divergence between distributions q and p. $\mathrm{KL}(q \parallel p) \geq 0$ with equality only if the two distributions coincide. This implies that $\mathcal{L}(q, \boldsymbol{\Theta})$ is a lower bound for $\log p(\mathcal{D}|\boldsymbol{\Theta})$. We can now optimise by iterating over following two-step method until convergence is achieved:

1. In the E-step, we want to maximise this lower bound w.r.t. $q(\mathcal{H})$. We do so by annihilating the KL distance; i.e., by setting
$$q(\mathcal{H}) = p\left(\mathcal{H}|\mathcal{D}, \boldsymbol{\Theta}^{\text{old}}\right), \tag{4}$$
where $\boldsymbol{\Theta}^{\text{old}}$ is the parameter estimate of the previous iteration. This requires that we do inference to determine the distribution $q(\mathcal{H})$. In essence this step replaces the hidden data with a distribution for it.

2. In the M-step, we hold $q(\mathcal{H})$ fixed and maximise this lower bound $\mathcal{L}(q, \boldsymbol{\Theta})$ w.r.t. $\boldsymbol{\Theta}$. If not already at a maximum, this will cause the log-likelihood function $\log p(\mathcal{D}|\boldsymbol{\Theta})$ to rise. Let us simplify a bit more:
$$\sum_{\mathcal{H}} q(\mathcal{H}) \log \left[\frac{p(\mathcal{D}, \mathcal{H}|\boldsymbol{\Theta})}{q(\mathcal{H})} \right] = \sum_{\mathcal{H}} q(\mathcal{H}) \log p(\mathcal{D}, \mathcal{H}|\boldsymbol{\Theta}) - \sum_{\mathcal{H}} q(\mathcal{H}) \log q(\mathcal{H}). \tag{5}$$
The term on the right does not depend on $\boldsymbol{\Theta}$ and we can ignore it, leaving us with the auxiliary function
$$\mathcal{Q}(\boldsymbol{\Theta}) \triangleq \sum_{\mathcal{H}} q(\mathcal{H}) \log p(\mathcal{D}, \mathcal{H}|\boldsymbol{\Theta}) = \langle \log p(\mathcal{D}, \mathcal{H}|\boldsymbol{\Theta}) \rangle_{q(\mathcal{H})} \tag{6}$$
to optimise w.r.t. $\boldsymbol{\Theta}$. This is a particularly nice function to work with if our distribution is part of the exponential family. The $\log$ will cancel with the $\exp$ leaving us with a very clean function to optimise over.
We can also extend the M-step to do a MAP estimate instead of an ML estimate – for this we need to maximise $\mathcal{L}(q, \boldsymbol{\Theta}) + \log p(\boldsymbol{\Theta})$ w.r.t. $\boldsymbol{\Theta}$.

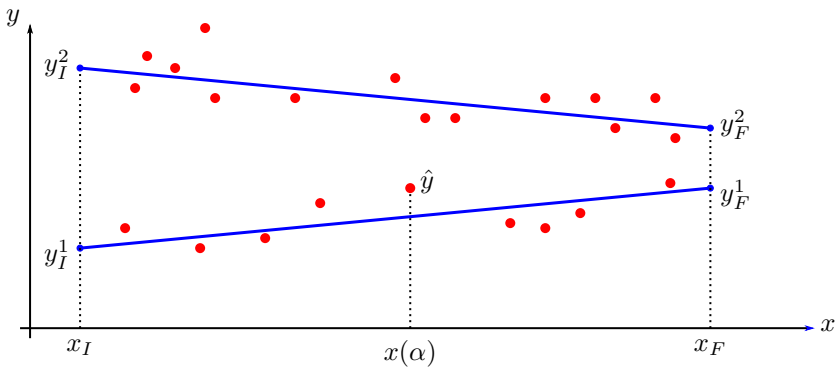
We can summarise the resulting iterative algorithm as shown below.

Algorithm 1 The expectation-maximisation (EM) algorithm

Initialise $\boldsymbol{\Theta}^{\text{old}}$
while not converged **do**
  $q(\mathcal{H}) \leftarrow p\left(\mathcal{H}|\mathcal{D}, \boldsymbol{\Theta}^{\text{old}}\right)$ ▷ E-step
  $\boldsymbol{\Theta}^{\text{new}} \leftarrow \underset{\boldsymbol{\Theta}}{\text{argmax}}\ \mathcal{Q}(\boldsymbol{\Theta}),$ where $\mathcal{Q}(\boldsymbol{\Theta}) = \sum_{\mathcal{H}} q(\mathcal{H}) \log p(\mathcal{D}, \mathcal{H}|\boldsymbol{\Theta})$ ▷ M-step
  $\boldsymbol{\Theta}^{\text{old}} \leftarrow \boldsymbol{\Theta}^{\text{new}}$
end while

4 Practical: EM for a regression example with multiple lines

For this question, we put a twist on the line-fitting problem of the past few assignments. In the previous assignments, the data points were noisy measurements of a line, and the y-coordinates of the endpoints of the line were unknown – these coordinates were the parameters to be estimated. In this assignment, we extend the problem to one where a data point is a noisy measurement of one of L different lines, and we try to estimate the y-coordinates of all the lines from the measurements. Consider the figure below with $L = 2$ lines, and with the y-coordinates of both lines shown.



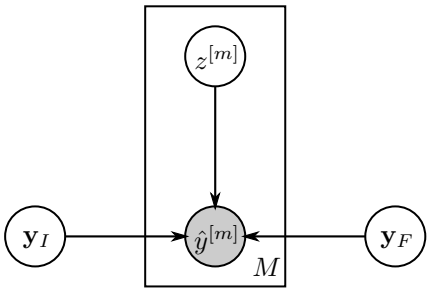
We know that there are two lines, but we do not know which of the two lines generated any specific measurement. We model this uncertain association between the lines and a measurement with a discrete RV z, where the value of z refers to the line that generated the measurement. For this two-line case, the measurement model of the previous assignments is now extended to become

$$\hat{y} = \begin{cases} (1-\alpha)y_I^1 + \alpha y_F^1 + v, & z = 1 \\ (1-\alpha)y_I^2 + \alpha y_F^2 + v, & z = 2 \end{cases}, \tag{7}$$

where the measurement noise for any point is given by $v \sim \mathcal{N}(0, \sigma_v^2)$. A measurement is therefore drawn from a different Gaussian distribution when line 1 generates the measurement than when line 2 generates the measurement.

In the general case where there are L lines, the y-coordinates of the left endpoints of the lines are given by $\mathbf{y}_I = \left(y_I^1, \ldots, y_I^L\right)$ and the y-coordinates of the right endpoints are given by $\mathbf{y}_F = \left(y_F^1, \ldots, y_F^L\right)$. There are M association variables – one for each measurement – and the mth association variable is a discrete RV that can take on L possible values, or $z^{[m]} \in \{1, \ldots, L\}$. The quantities of the general description of the EM algorithm of Section 3 are the following: the parameters are the endpoints of the lines, $\boldsymbol{\Theta} = (\mathbf{y}_I, \mathbf{y}_F)$, the hidden variables are the M association variables, $\mathcal{H} = \left(z^{[1]}, \ldots, z^{[M]}\right)$, and the data consists of the noisy measurements, $\mathcal{D} = \left(\hat{y}^{[1]}, \ldots, \hat{y}^{[M]}\right)$. The objective of this question is to estimate the paramters $\boldsymbol{\Theta}$ from the data $\mathcal{D}$ using the EM algorithm.

We can model this setup using a BN in plate notation as shown below.



From the BN, we can see that each measurement $\hat{y}^{[m]}$ has an association variable $z^{[m]}$ that functions as a "selector" of the lines parameterised by $\mathbf{y}_I$ and $\mathbf{y}_F$. We assume that each line is equally likely to be selected, which means that $p\left(z^{[m]}\right) = \frac{1}{L}$ (i.e., the prior distribution over $z^{[m]}$ is uninformative or flat).

Since working out the EM algorithm for this example is somewhat involved, we will work through most of the development together; you will only need to complete the last bit of the development and implement the algorithm in code by yourself.

E-step: For the E-step, we have to calculate the posterior distribution over the hidden variables $\mathcal{H}$ given the data $\mathcal{D}$ and a specific set of parameters $\boldsymbol{\Theta}$; i.e., we should calculate $p(\mathcal{H}|\mathcal{D}, \boldsymbol{\Theta})$. This means that we have to perform inference for this model.

Since the prior distributions over $z^{[m]}$ are uninformative, they have no effect on inference, and we can omit them. The only factors of interest are therefore the likelihood functions for the measurements; the likelihood function of the mth measurement is given in the table below.

$z^{[m]}$	$p\left(\hat{y}^{[m]}\vert\mathbf{y}_I,\mathbf{y}_F,z^{[m]}\right)$
1	$\mathcal{N}\left(\hat{y}^{[m]};\left(1-\alpha^{[m]}\right)y_I^1+\alpha^{[m]}y_F^1,\sigma_v^2\right)$
$\vdots$	$\vdots$
L	$\mathcal{N}\left(\hat{y}^{[m]};\left(1-\alpha^{[m]}\right)y_I^L+\alpha^{[m]}y_F^L,\sigma_v^2\right)$

As you can see, there is a different Gaussian likelihood function for each value of the association variable $z^{[m]}$. The combined likelihood function is the product of the likelihood functions for all the measurements, or

$$p\left(\mathcal{D}\vert\mathcal{H},\mathbf{\Theta}\right)=\prod_{m=1}^{M}p\left(\hat{y}^{[m]}\vert\mathbf{y}_I,\mathbf{y}_F,z^{[m]}\right). \tag{8}$$

The posterior distribution over the hidden variables is now simply given by

$$p\left(\mathcal{H}\vert\mathcal{D},\mathbf{\Theta}\right)\propto p\left(\mathcal{D}\vert\mathcal{H},\mathbf{\Theta}\right)p\left(\mathbf{\Theta}\right), \tag{9}$$

and since $p(\mathbf{\Theta})$ is uninformative, the posterior is directly proportional to the combined likelihood. The only remaining step in this inference process is to normalise the result. Since the association variables are independent of each other in the posterior distribution (you can clearly see this from the BN, or from the factorisation of Equation 9 as given by Equation 8), we can normalise the factor over each association variable $z^{[m]}$ separately, which results in the following table

$z^{[m]}$	$q\left(z^{[m]}\right)=p\left(z^{[m]}\vert\hat{y}^{[m]},\mathbf{y}_I,\mathbf{y}_F,\right)$
1	$\gamma_1^{[m]}$
$\vdots$	$\vdots$
L	$\gamma_L^{[m]}$

where

$$\gamma_l^{[m]}=\frac{\mathcal{N}\left(\hat{y}^{[m]};\left(1-\alpha^{[m]}\right)y_I^l+\alpha^{[m]}y_F^l,\sigma_v^2\right)}{\sum_{i=1}^{L}\mathcal{N}\left(\hat{y}^{[m]};\left(1-\alpha^{[m]}\right)y_I^i+\alpha^{[m]}y_F^i,\sigma_v^2\right)}. \tag{10}$$

The values $\gamma_1^{[m]},\ldots,\gamma_L^{[m]}$ are called the *responsibilities*, and they represent the soft assocation between data point m and each of the lines. Note that there are $M\times L$ responsibilities in total.

As specified in the E-step, we set $q(\mathcal{H})$ equal to this posterior; i.e.,

$$q(\mathcal{H})=\prod_{m=1}^{M}q\left(z^{[m]}\right)=\prod_{m=1}^{M}p\left(z^{[m]}\vert\hat{y}^{[m]},\mathbf{y}_I,\mathbf{y}_F,\right), \tag{11}$$

where $p\left(z^{[m]}\vert\hat{y}^{[m]},\mathbf{y}_I,\mathbf{y}_F,\right)$ is specified in the table above. The distribution $q(\mathcal{H})$ is therefore parameterised by the $M\times L$ responsibilities.

M-step: The aim of the M-step is to construct the function $\mathcal{Q}(\mathbf{\Theta})$ using distribution $q(\mathcal{H})$, and then to find the parameters $\mathbf{\Theta}$ that maximmise this function.

We start this step by writing down the joint distribution $p(\mathcal{D},\mathcal{H}\vert\mathbf{\Theta})$, which is simply the product of factors in the model,

$$p(\mathcal{D},\mathcal{H}\vert\mathbf{\Theta})=\prod_{m=1}^{M}p\left(\hat{y}^{[m]}\vert\mathbf{y}_I,\mathbf{y}_F,z^{[m]}\right)p\left(z^{[m]}\right). \tag{12}$$

Function $\mathcal{Q}(\mathbf{\Theta})$ is then given by

$$\begin{aligned}\mathcal{Q}(\mathbf{\Theta})&=\sum_{\mathcal{H}}q(\mathcal{H})\log p(\mathcal{D},\mathcal{H}\vert\mathbf{\Theta})\\&=\sum_{z^{[1]},\ldots,z^{[M]}}\left[\prod_{i=1}^{M}q\left(z^{[i]}\right)\right]\left[\sum_{m=1}^{M}\log p\left(\hat{y}^{[m]}\vert\mathbf{y}_I,\mathbf{y}_F,z^{[m]}\right)+\sum_{m=1}^{M}\log p\left(z^{[m]}\right)\right]\\&=\sum_{m=1}^{M}\left\{\sum_{z^{[1]},\ldots,z^{[M]}}\left[\prod_{i=1}^{M}q\left(z^{[i]}\right)\right]\log p\left(\hat{y}^{[m]}\vert\mathbf{y}_I,\mathbf{y}_F,z^{[m]}\right)\right\}+\sum_{z^{[1]},\ldots,z^{[M]}}\left[\prod_{i=1}^{M}q\left(z^{[i]}\right)\right]\left[M\log\frac{1}{L}\right],\end{aligned} \tag{13}$$

where we used Equations 11 and 12, moved the inner sum to the outside, and used the fact that the prior distribution $p(z^{[m]})$ is uninformative. We can manipulate the last term by moving each of the sums one by one into the product, or

$$\sum_{z^{[1]},\ldots,z^{[M]}} \left[\prod_{i=1}^{M} q\left(z^{[i]}\right)\right] = \left[\sum_{z^{[1]}} q\left(z^{[1]}\right)\right] \times \cdots \times \left[\sum_{z^{[M]}} q\left(z^{[M]}\right)\right] = 1, \tag{14}$$

which reduces to 1 since the marginalisation of distribution $q\left(z^{[m]}\right)$ over its variable $z^{[m]}$ is 1. We now continue manipulating $\mathcal{Q}$ by moving each of the sums one by one into the product, and using the same marginalisation property on all but one of the factors:

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\Theta}) = &\sum_{m=1}^{M}\left\{\left[\sum_{z^{[1]}} q\left(z^{[1]}\right)\right] \times \cdots \times \left[\sum_{z^{[m]}} q\left(z^{[m]}\right)\log p\left(\hat{y}^{[m]}|\mathbf{y}_I, \mathbf{y}_F, z^{[m]}\right)\right] \times \cdots \times \left[\sum_{z^{[M]}} q\left(z^{[M]}\right)\right]\right\} \\ &+ \text{const.} \\ = &\sum_{m=1}^{M}\left\{\sum_{z^{[m]}} q\left(z^{[m]}\right)\log p\left(\hat{y}^{[m]}|\mathbf{y}_I, \mathbf{y}_F, z^{[m]}\right)\right\} + \text{const.}\end{aligned} \tag{15}$$

The value of the constant terms are unimportant, since it will not effect the subsequent optimisation. We can now swop the sums around, and write things in terms of the responsibilities $\gamma_l^{[m]}$ and the definition of the likelihood function for each measurement (as defined in the development in the E-step) as follows:

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\Theta}) &= \sum_{z^{[m]}}\left\{\sum_{m=1}^{M} q\left(z^{[m]}\right)\log p\left(\hat{y}^{[m]}|\mathbf{y}_I, \mathbf{y}_F, z^{[m]}\right)\right\} + \text{const.} \\ &= \sum_{l=1}^{L}\left\{\sum_{m=1}^{M}\gamma_l^{[m]}\left(-\frac{1}{2\sigma_v^2}\left[\hat{y}^{[m]} - \left(1-\alpha^{[m]}\right)y_I^l - \alpha^{[m]}y_F^l\right]^2\right)\right\} + \text{const..}\end{aligned} \tag{16}$$

If you look closely at Equation 16, you will see that each term only depends on the parameters of a single line (or is constant). The part btween braces ($\{\cdots\}$) should also look familiar to you – it is virtually the same as the log-likelihood function used for ML estimation for a single line – only the responsibilities are new.

At this point, our journey together comes to an end and I leave you to forge ahead by yourself.

(a) Complete the development of the M-step; i.e., calculate the parameters $\boldsymbol{\Theta}^{\text{new}}$ that would maximise $\mathcal{Q}(\boldsymbol{\Theta})$, or

$$\boldsymbol{\Theta}^{\text{new}} = \underset{\boldsymbol{\Theta}}{\text{argmax}}\ \mathcal{Q}(\boldsymbol{\Theta}). \tag{17}$$

Note that you can do the optimisation for each line separately, since each term in $\mathcal{Q}(\boldsymbol{\Theta})$ only depends on the parameters of a single line. Use a similar optimisation procedure than what you previously used for MLE for a single line.

(b) Choose the number of lines L as well as the parameters for each line ($\mathbf{y}_I$ and $\mathbf{y}_F$). Now simulate a number of measurements by first drawing a value for $z^{[m]}$ and then drawing a value for $\hat{y}^{[m]}$ given the value of $z^{[m]}$ and the parameters of the associated line. Plot the actual lines and measurements.

(c) Implement the EM algorithm: Choose appropriate initial parameter values, and then iteratively perform the E-step an M-step until the parameters have converged. Plot the estimated lines, the actual lines and the data points on the same graph for some of the iterations.

(d) Investigate the effect of parameter initialisation, measurement noise level, number of measurements, as well as the number and arrangement of lines on the performance of the EM algorithm.

(Sjoe!)

5 Practical: EM for the Hamming (7,4) error-correction code (optional)

For this optional question (that you can complete for bonus marks), we'll revisit the familiar Hamming (7,4) error-correction code. In Question 3 of session 8, you used MLE to estimate the parameters of the check bit and measurement factors with the assumption that both the sent bits $b_0, \ldots, b_6$ and the received bits $r_0, \ldots, r_6$ are known. Now repeat this question, but with only the received bits observed; the sent bits are therefore latent/hidden RVs. Since we have latent RVs, you will have to use the EM algorithm for estimation. You can first assume that the parameters of the check bit factors are known (i.e., only the parameter of the measurement factors is unknown), and then extend it to include estimation of the check bit parameters too.