Assignment 2

Unsupervised & Probabilistic Learning

Maneesh Sahani

Due: Monday Nov 5, 2018

Note: Assignments are due at 11:00 AM (the **start** of lecture) on the date above. The usual College late assignments policy will apply.

You must make a credible attempt at the main questions to receive credit for the bonus ones.

1. **[65 marks + 5 BONUS] EM for Binary Data**.

   Consider the data set of binary (black and white) images used in the previous assignment. Each image is arranged into a vector of pixels by concatenating the columns of pixels in the image. The data set has N images $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ and each image has D pixels, where D is (number of rows × number of columns) in the image. For example, image $\mathbf{x}^{(n)}$ is a vector $(x_1^{(n)}, \ldots, x_D^{(n)})$ where $x_d^{(n)} \in \{0, 1\}$ for all $n \in \{1, \ldots, N\}$ and $d \in \{1, \ldots, D\}$.

   (a) Write down the likelihood for a model consisting of a mixture of K multivariate Bernoulli distributions. Use the parameters $\pi_1, \ldots, \pi_K$ to denote the mixing proportions $(0 \leq \pi_k \leq 1; \sum_k \pi_k = 1)$ and arrange the K Bernoulli parameter vectors into a matrix P with elements p_{kd} denoting the probability that pixel d takes value 1 under mixture component k. Assume the images are iid under the model, and that the pixels are independent of each other within each component distribution. [5 marks]

   Just as we can for a mixture of Gaussians, we can formulate this mixture as a latent variable model, introducing a discrete hidden variable $s^{(n)} \in \{1, \ldots, K\}$ where $P(s^{(n)} = k|\boldsymbol{\pi}) = \pi_k$.

   (b) Write down the expression for the responsibility of mixture component k for data vector $\mathbf{x}^{(n)}$, i.e. $r_{nk} \equiv P(s^{(n)} = k|\mathbf{x}^{(n)}, \boldsymbol{\pi}, \mathsf{P})$. This computation provides the E-step for an EM algorithm. [5 marks]

   (c) Find the maximizing parameters for the expected log-joint

   $$\underset{\boldsymbol{\pi},\mathsf{P}}{\operatorname{argmax}} \left\langle \sum_n \log P(\mathbf{x}^{(n)}, s^{(n)}|\boldsymbol{\pi}, \mathsf{P}) \right\rangle_{q(\{s^{(n)}\})}$$

   thus obtaining an iterative update for the parameters $\boldsymbol{\pi}$ and P in the M-step of EM. [10 marks]

   (d) Implement the EM algorithm for a mixture of K multivariate Bernoullis.

   Your code should take as input the number K, a matrix X containing the data set, and a maximum number of iterations to run. The algorithm should run for that number of iterations or until the log likelihood converges (does not increase by more than a very small amount).

   Hand in clearly commented code and a description of how the code works.

   Run your algorithm on the data set for values of K in $\{2, 3, 4, 7, 10\}$. Report the log likelihoods obtained and display the parameters found.

   [30 marks]

   [Hints: Although the implementation may seem simple enough given the equations, there are many numerical pitfalls (that are common in much of probabilistic learning). A few suggestions:

   - Likelihoods can be very small; it is often better to work with log-likelihoods.
   - You may still encounter numerical issues computing responsibilities. Consider scaling the numerator and denominator of the equation by a suitable constant while still in the log domain.

- It may also help to introduce (weak) priors on P and $\boldsymbol{\pi}$ and use EM to find the MAP estimates, rather than ML.
- If working in MATLAB, consider "vectorizing" your code, avoiding `for` loops where possible. This often [though not always] makes the code more readable and lets it run faster.
- It is always useful to plot the log-likelihood after every iteration. If it ever decreases (when implementing EM for ML) you have a bug! The same is true for the log-joint when implementing MAP.

(e) Express the log-likelihoods obtained in bits and relate these numbers to the length of the naive encoding of these binary data. How does your number compare to gzip (or another compression algorithm)? Why the difference? [5 marks]

(f) Run the algorithm a few times starting from randomly chosen initial conditions. Do you obtain the same solutions (up to permutation)? Does this depend on K?

Comment on how well the algorithm works, whether it finds good clusters (look at the cluster means and responsibilities and try to interpret them), and how you might improve the model. [10 marks]

(g) [BONUS] Consider the *total* cost of encoding both the model parameters and the data given the model. How does this total cost compare to gzip (or similar)? How does it depend on K? What might this tell you? [5 marks]

2. **[20 marks] Modelling Data**.

(a) Download the data file called `geyser.txt` from the course web site. This is a sequence of 295 consecutive measurements of two variables from Old Faithful geyser in Yellowstone National Park: the duration of the current eruption in minutes (rounded to the nearest second), and the waiting time since the last eruption in minutes (to the nearest minute).

Examine the data by plotting the variables within (`plot(geyser(:,1),geyser(:,2),'o');`) and between (e.g. `plot(geyser(1:end-n,1),geyser(n+1:end,1 or 2),'o');` for various `n`) time steps. Discuss and justify based on your observations what kind of model might be most appropriate for this data set. Consider each of the models we have encountered in the course through week 3: a multivariate normal, a mixture of Gaussians, a Markov chain, a hidden Markov model, an observed stochastic linear dynamical system and a linear-Gaussian state-space model. Can you guess how many discrete states or (continuous) latent dimensions the model might have? [10 marks]

(b) Consider a data set consisting of the following string of 160 symbols from the alphabet $\{\mathtt{A}, \mathtt{B}, \mathtt{C}\}$:

```
AABBBACABBBACAAAAAAAABBBACAAAAABACAAAAAABBBBACAAAAAAAAAAAAABACABACAABBACAAABBBBA
CAAABACAAAABACAABACAAABBACAAAABBBBACABBACAAAAAABACABACAAABACAABBBACAAAABACABBACA
```

Study this string and suggest the structure of an HMM model that may have generated it. Specify the number of hidden states in the HMM, the transition matrix with any constraints and estimates of the transition probabilities and the output or emission matrix probalities, and the intial state probabilities. You need to provide some description/justification for how you arrived at these numbers. We do **not** expect you to implement the Baum-Welch algorithm—you should be able to answer this question just by examining the sequence carefully. [10 marks]

3. **[15 marks] Zero-temperature EM**. In the automatic speech recognition community HMMs are sometimes trained by using the Viterbi algorithm in place of the forward–backward algorithm. In other words, in the E step of EM (Baum–Welch), instead of computing the expected sufficient statistics from the posterior distribution over hidden states: $p(\mathbf{s}_{1:T}|\mathbf{x}_{1:T}, \theta)$, the sufficient statistics are computed using the single *most probable* hidden state sequence: $\mathbf{s}^*_{1:T} = \arg\max_{\mathbf{s}_{1:T}} p(\mathbf{s}_{1:T}|\mathbf{x}_{1:T}, \theta)$.

(a) Is this algorithm guaranteed to converge (in the sense that the free-energy or some other reaches an asymptote)? To answer this you might want to consider the discussion of the real EM algorithm and what happens if we constrain $q(s)$ to put all its mass on one setting of the hidden variables. Support your arguments. [10 marks]

(b) If it converges, will it converge to a maximum of the likelihood? If it does not converge what will happen? Support your arguments. [5 marks]

(c) [Bonus (just for culture)] Why do you think this question is labelled “Zero-temperature EM” Hint: think about where temperature would appear in the the free-energy. [no marks]

BONUS QUESTION: attempt the questions above before anwering this one.

4. **[Bonus: 55 marks] BONUS question: LGSSMs, EM and SSID.**

Download the datafiles `ssm_spins.txt` and `ssm_spins_test.txt`. Both have been generated by an LGSSM:

$$\mathbf{y}_t \sim \mathcal{N}(A\mathbf{y}_{t-1}, Q) \quad [t = 2 \ldots T] \qquad\qquad \mathbf{y}_1 \sim \mathcal{N}(0, I)$$
$$\mathbf{x}_t \sim \mathcal{N}(C\mathbf{y}_t, R) \quad [t = 1 \ldots T]$$

using the parameters:

$$A = 0.99 \begin{pmatrix} \cos(\frac{2\pi}{180}) & -\sin(\frac{2\pi}{180}) & 0 & 0 \\ \sin(\frac{2\pi}{180}) & \cos(\frac{2\pi}{180}) & 0 & 0 \\ 0 & 0 & \cos(\frac{2\pi}{90}) & -\sin(\frac{2\pi}{90}) \\ 0 & 0 & \sin(\frac{2\pi}{90}) & \cos(\frac{2\pi}{90}) \end{pmatrix} \qquad Q = I - AA^{\mathsf{T}}$$

$$C = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0.5 & 0.5 & 0.5 & 0.5 \end{pmatrix} \qquad R = I$$

but different random seeds. We shall use the first as a training data set and the second as a test set.

(a) Run the function `ssm_kalman.m` or `ssm_kalman.py` that we have provided (or a re-implementation in your favourite language if you prefer) on the training data. Warning (for MATLAB version): the function expects data vectors in columns; you will need to transpose the loaded matrices! Make the following plots (or equivelent):

```
logdet = @(A)(2*sum(log(diag(chol(A)))));
[Y,V,~,L] = ssm_kalman(X',Y0,Q0,A,Q,C,R, 'filt');
plot(Y'):
plot(cellfun(logdet,V));

[Y,V,Vj,L] = ssm_kalman(X',Y0,Q0,A,Q,C,R, 'smooth');
plot(Y'):
plot(cellfun(logdet,V));
```

Explain the behaviour of `Y` and `V` in both cases (and the differences between them).
[5 marks]

(b) Write a function to learn the parameters A, Q, C and R using EM (we will assume that the distribution on the first state is known *a priori*). The M-step update equations for A and C were derived in lecture. You should show that the updates for R and Q can be written:

$$R_{\text{new}} = \frac{1}{T}\left[\sum_{t=1}^{T} \mathbf{x}_t\mathbf{x}_t^{\mathsf{T}} - \Big(\sum_{t=1}^{T} \mathbf{x}_t\langle\mathbf{y}_t^{\mathsf{T}}\rangle\Big)C_{\text{new}}^{\mathsf{T}}\right]$$
$$Q_{\text{new}} = \frac{1}{T-1}\left[\sum_{t=2}^{T} \langle\mathbf{y}_t\mathbf{y}_t^{\mathsf{T}}\rangle - \Big(\sum_{t=2}^{T} \langle\mathbf{y}_t\mathbf{y}_{t-1}^{\mathsf{T}}\rangle\Big)A_{\text{new}}^{\mathsf{T}}\right]$$

where C_{new} and A_{new} are as in the lecture notes. Store the log-likelihood at every step (easy to compute from the fourth value returned by `ssm_kalman`) and check that it increases.
[Hint: the matlab code

```
cellsum=@(C)(sum(cat(3,C{:}),3))
```

defines an inline function `cellsum()` that sums the entries of a cell array of matrices.]

Run at least 50 iterations of EM starting from a number of different initial conditions: (1) the generating parameters above (why does EM not terminate immediately?) and (2) 10 random choices.

Show how the likelihood increases over the EM iterations (hand in a plot showing likelihood vs iteration for each run, plotted in the same set of axes). Explain the features of this plot that you think are salient.

[If your code and/or computer is fast enough, try running more EM iterations. What happens?]

[25 marks]

(c) Write a function to learn A, Q, C and R using Ho-Kalman SSID. You should provide options to specify the maximum lag to include in the future-past Hankel matrix (H), and the latent dimensionality. You may like to plot the singular value spectrum of H to see how good a clue it provides to dimensionality.

[Hints: if `M` is a cell array of the lagged correlation matrices M_τ, you can build the Hankel matrix using:

```
   H = cell2mat(M(hankel([1:maxlag], [maxlag:2*maxlag-1])));
```

The matrices Ξ (`Xi`) and Υ (`Ups`) are found using SVD (assuming `K` latent dimensions):

```
  [Xi,SV,Ups] = svds(H, K);

  Xi = Xi*sqrt(SV);
  Ups = sqrt(SV)*Ups';
```

and these can be used to find $\widehat{C}$ and $\widehat{A}$ and $\widehat{\Pi}$ by regression (`/` or `\` in MATLAB), e.g.:

```
  Ahat = Xi(1:end-DD,:)\Xi(DD+1:end,:);
```

(The code for C and Π will take a little more thought!).

To find $\widehat{Q}$ and $\widehat{R}$ recall that, as Π is the stationary (prior) latent covariance:

$$A\Pi A^{\mathsf{T}} + Q = \Pi$$

and

$$C\Pi C^{\mathsf{T}} + R = \mathbb{E}_{t\to\infty}\left[\mathbf{x}_t\mathbf{x}_t^{\mathsf{T}}\right]$$

Run your code on the training data using a maximum lag of 10 and the true latent dimensionality. Evaluate the likelihood of this model using `ssm_kalman.m` and show it on the likelihood figure. Now use the SSID parameters as initial values for EM and show the resulting likelihood curve.

Comment on the advantages and disadvantanges of SSID as compared to EM.

(If you have time and inclination, you might like to explore how your results vary with different choices of lag and dimensionality)

[20 marks]

(d) Evaluate the likelihood of the test data under the true parameters, and all of the parameters found above (EM initialised at the true parameters, random parameters and the SSID parameters, as well as the SSID parameters without EM). Show these numbers on or next to the training data likelihoods plotted above. Comment on the results.

[5 marks]