Problem Set 3
Assigned: 03/31/2005
Due: 04/12/2005

Please submit an electronic copy of your writeup and code for each problem to `6869-submit@csail.mit.edu`.

Problem 1 *Probabilistic Graphical Model*

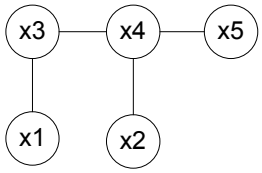

Figure 1: A probabilistic graphical model.

Consider the probabilistic graphical model in Figure 1.

(a) Assume that the 5 variables described by this model are jointly Gaussian. Write down the general form of the inverse covariance matrix (you can assume each of the depicted variables is a scalar, not a vector).

(b) Do the following in Matlab: First, select values for the inverse covariance matrix of the 5 variables. (*You should include this matrix in your write-up for comparison purposes.*) Then, assume the variables all have zero mean and draw 50 samples of the 5 jointly gaussian random variables. (this will give you a 50 by 5 dimensional matrix of observations). You can use `gausssamp.m` for the sampling.

   (i) Using the maximum likelihood method described in class, estimate the covariance of the 5 jointly gaussian random variables and compute the inverse covariance matrix. How does the result compare with the values implied by the true inverse covariance matrix?

   (ii) Now exploit the known graphical model structure: find the maximum likelihood estimate of the covariance matrix for these 4 pairs of jointly gaussian random variables: (x1, x3), (x3, x4), (x4, x2),

(x4, x5). Assuming these estimates are correct, form another estimate of the inverse covariance matrix of the 5 jointly gaussian random variables. How does it compare with your estimate in part (i), and with the true inverse covariance matrix?

Problem 2 *Iterated Proportional Fitting*

Consider 3 variables, a, b, and c, each of which can have 5 possible states. Form a 3-dimensional cube of (random) probability mass values between zero and one for each triplet of states, normalized to integrate to one over the whole cube of data.

As in class, you observe the marginal probabilities, $P_a(a)$, $P_b(b)$, $P_c(c)$. Make a random initial guess for the joint probabilities, $P(a, b, c)$, and iteratively modify it using Iterated Proportional Fitting for 5 or 10 iterations.

Plot the fit to the observed marginals for several iterations of IPF. Also plot the KL distance between the original and estimated joint probabilities as a function of IPF iteration. (For a definition of KL distance, see, for example, http://www-nlp.stanford.edu/fsnlp/mathfound/fsnlp-slides-kl.pdf).

Problem 3 *Markov Network*

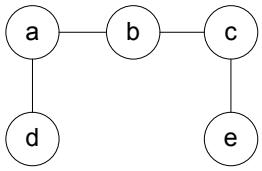

Figure 2: A Markov network.

Consider the Markov network in Figure 2. Each of the variables is binary and can be in state 0 or state 1.

d is observed to be in state 1. e is observed to be in state 0. The compatibility matrix Φ between a&d equals that between c&e and is

$$\Phi(a, d) = \begin{pmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{pmatrix}$$

The compatibility matrix Ψ between a&b equals that between b&c and is

$$\Psi(a, b) = \begin{pmatrix} \alpha & 1\text{-}\alpha \\ 1\text{-}\alpha & \alpha \end{pmatrix}$$

(a) For $\alpha = 0.99$, find $P(a)$, the marginal probabilities for the variable a being in either of its two possible states, 0 and 1.

(b) Do the same for $\alpha = 0.6$. Discuss why the result is different in these two cases.

For Problem 4 and 5, we will cover the material in lectures on April 5 and 7. If you want to start on these problems before then, the reading for this material is Chapt. 22, sections 1-3 of Forsyth and Ponce book

Problem 4 *Eigenfaces for Recognition*

In this problem we explore the use of eigenfaces for person recognition. Download the MATLAB data file `faces.mat`, which contains the cropped, normalized 51×43 images of 34 individuals. Each individual has been photographed with two facial expressions: "neutral" and "smiling". The pixels of each person's image have been stored as columns of the `neutralFaces` and `smileFaces` matrices. To convert back and forth between image and vector representations, use the `reshape` command.

(a) To (approximately) account for illumination variations, normalize each of the face vectors so that it has zero mean and unit variance.

(b) Assume that we have a database of neutral face images, and want to determine the identity of the smiling individuals. For each of the 34 smiling faces, measure the norm of the difference between their normalized appearance vector and each of the 34 neutral faces. Classify each smiling face according to its nearest neighbor. What percentage of the smiling faces are correctly recognized?

(c) Using the MATLAB `svd` command, determine the principle components of the set of normalized neutral faces (do *not* include the smiling faces). Be sure to subtract the mean face before performing your PCA. Plot the mean face and the first three principle components (the "eigenfaces").

Hint: To avoid excessive memory usage when calculating SVD, use MATLAB's "economy size" option.

(d) Determine the number of principle components required to model 90% of the total variance in the neutral face set. Project each of the neutral and smiling faces onto the corresponding eigenfaces. Use the coefficients of these projections to classify each smiling face. Compute the percentage of correctly recognized faces, and compare to part (b).

(e) Repeat part (d) using the number of principal components required to model 80%, 70%, 60%, and 50% of the total neutral face variance. Plot the corresponding recognition rates.

Problem 5 *Eigenfaces for Detection*

(a) Construct the principle components (eigenfaces) which best model the combined variations of the neutral and smiling faces from Problem 4. Be sure to normalize each face vector as in 2(a). Plot the mean face and first three eignefaces, and compare to 2(c).

(b) Load the test image `mad.png`. Use MATLAB's `ginput` command to (approximately) locate the center's of this image's two (human) faces.

(c) Repeat this part for each of the faces from part (b). For each pixel in an 80×80 pixel square centered around the face, determine the best reconstruction of the 51×43 patch centered at that point using the first 10 principal components from part (a). Each image patch should be normalized as in 2(a) prior to this reconstruction procedure. Plot the distance between the input and reconstructed (normalized) patches as a function of patch location. For the smallest error shift, plot the input and reconstructed image patches.

(d) Repeat part (c), but this time search over the entire image. Make the same pair of plots as before.

(e) Are the eigenfaces suitable for face detection in general scenes? Why or why not? If not, are there controlled situations where they would prove more useful?