CMPUT 652—Probability Models for Artificial Intelligence

Assignment 2

Fall 2003
Department of Computing Science
University of Alberta

Due: in class, *Thursday, November 27*
Worth: 25% of final grade
Instructor: Dale Schuurmans, Ath409, x2-4806, dale@cs.ualberta.ca

---

Note These questions require you to write Matlab programs which are to be submitted by email. When finished, please send a *single* tar file containing all of your .m files to dale@cs.ualberta.ca with a subject heading "A2 solutions".

Automated image modeling and stabilization

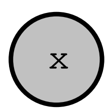
Figure 1: A bunch of images taken from a video of a person walking by and looking around

In this assignment you are going to estimate models of image structure from sample data like that shown in Figure 1. Each image can be represented by a vector of pixel intensities, and thus a set of sample images can be represented by a set of sample vectors. Note that in Matlab if an image, `img`, is an $n_1 \times n_2$ matrix, then `x = img(:)` creates an $n_1 n_2 \times 1$ vector representation of the matrix. Given such a vector representation `x`, the original image matrix can be recovered simply by `img = reshape(x,`n_1`,`n_2`)`.

You will model distributions over natural images by first postulating various generative probabilistic models of image formation, and then estimating the parameters of these models from sample data. Finally, you will use the models to stabilize test images. The image data is available on the course homepage `www.cs.ualberta.ca/~dale/cmput652` in the files "walkingby.mat" and "facesnow.mat". Each of these files contains a set of image data stored in an $n_1 \times n_2 \times t$ array `Data`, which you can retrieve using the `load` command in Matlab. (To amuse yourself you can view some of the images by typing in `imagesc(Data(:,:,`i`))` to display the ith image in the array.) Note that below, most of the operations require you to convert matrix images back and forth to their vector representation.

Question 1 (Gaussian model—3%)

The first model we consider is a simple multivariate Gaussian distribution over images, represented by the very simple graphical model:



$$p(x) \;=\; N(x; \mu, \Phi)$$

Here we assume that an image is represented by a vector x of pixel intensities. The model asserts that image vectors are generated by a Gaussian distribution with a mean vector μ and a covariance matrix Φ. We further assume that Φ is diagonal, and therefore only have to store the diagonal vector ϕ which is the same size as x and μ. This assumption asserts that each pixel has some variance in its values, but that the pixels are independent of one another.

Note that, in Matlab, the Gaussian model requires the representation of an $n_1 n_2 \times 1$ mean vector, `mu1`; and an $n_1 n_2 \times 1$ variance vector, `phi1`.

For this question, and the other questions below, assume you are given a set of observed image vectors $x_1, x_2, ..., x_t$ (stored in the array `Data`). From the set of observed image vectors you will then estimate the parameters of the given probability model. Below, we will use the notation

$$\left\langle f(x_i) \right\rangle \;=\; \frac{1}{t}\sum_{i=1}^{t} f(x_i)$$

which denotes the sample average of various functions f over observed image vectors $x_1, ..., x_t$.

(a) Write a Matlab function `[img] = generate1(mu1,phi1,n1,n2)` which generates a random `n1` $\times$ `n2` image matrix from the Gaussian model specified by mean vector `mu1` and variance vector `phi1`.

(b) Write a Matlab function `[mu1 phi1] = maxlike1(Data)` which returns the maximum likelihood parameters `mu1` and `phi1` for the Gaussian model, given the observed images in `Data`. Note that the formulas for the maximum likelihood estimates are given by

$$\begin{aligned} \mu &= \left\langle x_i \right\rangle \\ \phi &= \left\langle diag\left[(x_i - \mu)(x_i - \mu)^\top\right]\right\rangle \end{aligned}$$

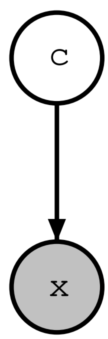
(c) Write a Matlab function `show1(mu1,phi1,n1,n2)` which displays an `n1` $\times$ `n2` matrix image of the mean vector `mu1` and the variance vector `phi1`.

(d) Generate a few random images from your estimated model, and tell me what you think.

Question 2 (Mixture of Gaussians—5%)

The first model, a simple Gaussian, turns out to be a rather poor model of natural images. The second, and subsequent models all attempt to enrich the representation to better capture certain phenomena of natural images.

The second model we consider is a *mixture* of m Gaussians:



$$\begin{aligned} p(c) &= M(c; \pi_1, \pi_2, ..., \pi_m) \\ p(x|c) &= N(x; \mu_c, \Phi_c) \end{aligned}$$

Here M denotes the multinomial distribution. The mixture component variable c can take on one of m values, $1, ..., m$; each of which determines a different Gaussian $N(x; \mu_c, \Phi_c)$ over the observed images. Again we assume the covariance matrices Φ_c are diagonal. Thus, this representation models the image distribution by a mixture of m Gaussians, each with their own probability π_c, mean vector μ_c, and variance vector ϕ_c. The complication in estimating these parameters from observed image vectors $x_1, x_2, ..., x_t$ is that the corresponding component variables $c_1, c_2, ..., c_t$ are unobserved. Therefore, we have to resort to the EM algorithm to estimate the parameters of this model. For this question assume $m = 5$.

Note that, in Matlab, this model requires the representation of an $m \times 1$ array of component probabilities, `pi2`; an $n_1 n_2 \times m$ array of mean vectors, `mu2` (one μ_c per column); and an $n_1 n_2 \times m$ array of variance vectors, `phi2` (one ϕ_c per column).

(a) Write a Matlab function `[img] = generate2(pi2,mu2,phi2,n1,n2)` which generates a random `n1` $\times$ `n2` image matrix from the mixture of Gaussian model specified by the component probabilities `pi2`, the mean vectors `mu2`, and variance vectors `phi2`.

(b) Write a Matlab function `[pi2 mu2 phi2] = em2(Data,maxiters,pi2in,mu2in,phi2in)` which uses EM to find a locally maximum likelihood set of parameters `pi2`, `mu2` and phi2 for the mixture of Gaussians model. Note that the formula for the E step is given by

$$p(c|x) = \frac{p(x|c)\,p(c)}{\sum_{c=1}^{m} p(x|c)\,p(c)} = \frac{N(x;\mu_c,\Phi_c)\,\pi_c}{\sum_{c=1}^{m} N(x;\mu_c,\Phi_c)\,\pi_c}$$

The formulas for the M step are given by

$$\pi_c = \Big\langle p(c|x_i) \Big\rangle$$

$$\mu_c = \frac{\Big\langle x_i\, p(c|x_i) \Big\rangle}{\Big\langle p(c|x_i) \Big\rangle}$$

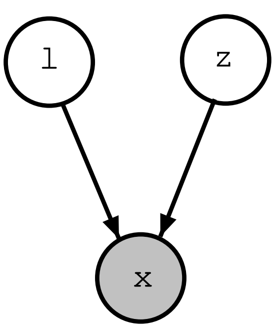
$$\phi_c = \frac{\Big\langle diag\left[(x_i-\mu_c)(x_i-\mu_c)^\top\right]\; p(c|x_i) \Big\rangle}{\Big\langle p(c|x_i) \Big\rangle}$$

(c) Write a Matlab function `show2(pi2,mu2,phi2,n1,n2)` which displays an `n1` $\times$ `n2` matrix image of the mean vectors `mu2` and the variance vectors `phi2`.

(d) Generate a few random images from your estimated model, and tell me what you think.

Question 3 (Transformed Gaussians—7%)

The Gaussian mixture model is still not very effective at modeling natural image sets (like image sequences from a video) because it does not capture the notion that objects moving in a scene undergo simple transformations in the image plane, such as translation, scaling, rotation, etc. Instead of a mixture model, one can attempt to model natural images by first generating a latent "ideal" image, z, and then transforming z according to one of a small number of simple transformations, G_l, for $l = 1, ..., k$. Thus, the image generation process can be characterized by first generating a latent image z, then picking a random transformation G_l, and finally generating the observed image x by transforming z according to $G_l z$ and adding zero mean Gaussian noise.



$$p(l) = M(l;\rho_1,\rho_2,...,\rho_k)$$
$$p(z) = N(z;\mu,\Phi)$$
$$p(x|l,z) = N(x;G_l z,\Psi)$$

What do the transformations look like? We will just consider simple horizontal and vertical shifts of the latent image. For example, given a latent image in matrix form, l, you can convert it to vector form z and then transform the vector by a multiplying it with a transformation matrix G constructed as follows.

$$
l = \begin{bmatrix} z_{11} & z_{12} & z_{13} \\ z_{21} & z_{22} & z_{23} \\ z_{31} & z_{32} & z_{33} \end{bmatrix} \qquad z = l(:) = \begin{bmatrix} z_{11} \\ z_{21} \\ z_{31} \\ z_{12} \\ z_{22} \\ z_{32} \\ z_{13} \\ z_{23} \\ z_{33} \end{bmatrix} \qquad G = \begin{bmatrix} & & & & & & 1 & 0 & 0 \\ & & & & & & 0 & 1 & 0 \\ & & & & & & 0 & 0 & 1 \\ 1 & 0 & 0 & & & & & & \\ 0 & 1 & 0 & & & & & & \\ 0 & 0 & 1 & & & & & & \\ & & & 1 & 0 & 0 & & & \\ & & & 0 & 1 & 0 & & & \\ & & & 0 & 0 & 1 & & & \end{bmatrix}
$$

Note that the product Gz gives the vector form of a new image, which is just the old image shifted one pixel to the right (with wrap-around), if you consider the corresponding matrix representation of the original image l. (The empty entries in G are zero.)

In this question, you will fix the transformations ahead of time. That is, the G_l matrices will not be estimated from data. Specifically, construct 17 transformation matrices corresponding to -4, -3, -2, -1, 0, 1, 2, 3, 4 pixel horizontal shifts and -4, -3, -2, -1, 1, 2, 3, 4 pixel vertical shifts respectively (with wrap-around). Thus, assume $k = 17$. (Since the G_l matrices are sparse, you may want to store them in a sparse format in Matlab.)

Note that, in Matlab, this model requires the representation of a $n_1 n_2 \times n_1 n_2 \times k$ sparse array of image transformation matrices `G3`, a $k \times 1$ array of transformation probabilities `rho3`, an $n_1 n_2 \times 1$ latent mean vector `mu3`, an $n_1 n_2 \times 1$ latent variance vector `phi3`, and an $n_1 n_2 \times 1$ observed variance vector `psi3`.

(a) Write a Matlab function `[img] = generate3(G3,rho3,mu3,phi3,psi3,n1,n2)` which generates a random `n1` $\times$ `n2` image matrix from the transformed Gaussian model specified by transformation matrices `G3`, transformation probabilities `rho3`, latent mean vector `mu3`, latent variance vector `phi3`, and observed variance vector `psi3`.

(b) Write a Matlab function `[rho3 mu3 phi3 psi3] = em3(Data,maxiters,G3,rho3in, mu3in,phi3in,psi3in)` which uses EM to find a locally maximum likelihood set of parameters `rho3`, `mu3`, `phi3` and `psi3` for the transformed Gaussians model. Note that the formulas for the E step are given by

$$
\begin{aligned}
p(l|x) &= \frac{p(x|l)\, p(l)}{\sum_{l=1}^{k} p(x|l)\, p(l)} = \frac{N\left(x;\ G_l \mu,\ G_l \Phi G_l^\top + \Psi\right)\ \rho_l}{\sum_{l=1}^{k} N\left(x;\ G_l \mu,\ G_l \Phi G_l^\top + \Psi\right)\ \rho_l} \\
p(z|l,x) &= N\Big(z;\ \mathrm{E}[z|l,x],\ \mathrm{Cov}[z|l,x]\Big)
\end{aligned}
$$

where

$$
\begin{aligned}
\mathrm{E}[z|l,x] &= \Omega_l G_l^\top \Psi^{-1} x + \Omega_l \Phi^{-1} \mu \\
\Omega_l \triangleq \mathrm{Cov}[z|l,x] &= (\Phi^{-1} + G_l^\top \Psi^{-1} G_l)^{-1}
\end{aligned}
$$

The formulas for the M step are given by

$$\rho_l = \Big\langle p(l|x_i) \Big\rangle$$

$$\mu = \left\langle \sum_{l=1}^{k} \mathrm{E}[z|l,x_i]\; p(l|x_i) \right\rangle$$

$$\phi = \left\langle \sum_{l=1}^{k} diag\Big[\mathrm{Cov}[z|l,x_i]\Big]\; p(l|x_i) \right\rangle$$

$$\Psi = \left\langle \sum_{l=1}^{k} diag\left[\Big(x_i - G_l\ \mathrm{E}[z|l,x_i]\Big)\Big(x_i - G_l\ \mathrm{E}[z|l,x_i]\Big)^\top\right]\; p(l|x_i) \right\rangle$$

(c) Write a Matlab function `show3(rho3,mu3,phi3,psi3,n1,n2)` which displays an `n1` $\times$ `n2` matrix image of the latent mean vector `mu3`, the latent variance vector `phi3`, and the observed variance vector `psi3`.

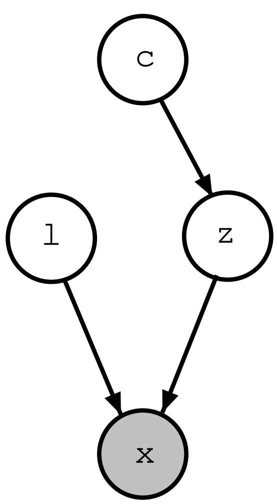
(d) Write a Matlab function `[img] = stabilize3(imgin,G3,rho3,mu3,phi3,psi3)` which calculates the expected value of the latent image given the observed image:

$$\mathrm{E}[z|x] = \sum_{l=1}^{k} \mathrm{E}[z|x,l]\; p(l|x)$$

(e) Generate a few random images from your estimated model. Stabilize a few of the natural images from your data. Tell me what you think.

Question 4 (Mixture of transformed Gaussians—8%)

Finally, we will consider a model that combines all of the previous models. Specifically, the fourth model characterizes natural images by a mixture of transformed Gaussians. Intuitively, the different components of the mixture capture different basic appearances of a latent object, whereas the transformations capture simple image translations as before.



$$\begin{aligned}
p(c) &= M(c;\pi_1,\pi_2,...,\pi_m) \\
p(l) &= M(l;\rho_1,\rho_2,...,\rho_k) \\
p(z|c) &= N(z;\mu_c,\Phi_c) \\
p(x|l,z) &= N(x;G_l z,\Psi)
\end{aligned}$$

In this case, set $m = 5$ mixture components, and set $k = 17$ transformations, using the same transformation matrices $G_1, ..., G_{17}$ as before.

Note that, in Matlab, this model requires the representation of an $m \times 1$ vector of component probabilities, `pi4`; a $n_1 n_2 \times n_1 n_2 \times k$ array of image transformation matrices, `G4`; a $k \times 1$ array of transformation probabilities, `rho4`; an $n_1 n_2 \times m$ array of latent mean vectors, `mu4` (one μ_c per column); an $n_1 n_2 \times m$ array of latent variance vectors, `phi4` (one ϕ_c per column); and an $n_1 n_2 \times 1$ observed variance vector, `psi4`.

(a) Write a Matlab function `[img] = generate4(pi4,G4,rho4,mu4,phi4,psi4,n1,n2)` which generates a random `n1` $\times$ `n2` image matrix from the mixture of transformed Gaussians model specified by component probabilities `pi4`, transformation matrices `G4`, transformation probabilities `rho4`, latent mean vectors `mu4`, latent variance vectors `phi4`, and observed variance vector `psi4`.

(b) Write a Matlab function `[pi4 rho4 mu4 phi4 psi4] = em4(Data,maxiters,G4, pi4in,rho4in,mu4in,phi4in,psi4in)` which uses EM to find a locally maximum likelihood set of parameters `pi4`, `rho4`, `mu4`, `phi4` and `psi4` for the mixture of transformed Gaussians model. Note that the formulas for the E step are given by

$$p(l,c|x) = \frac{p(x|l,c)\; p(l,c)}{\sum_{c=1}^{m}\sum_{l=1}^{k} p(x|l,c)\; p(l,c)} = \frac{N\left(x;\; G_l\mu_c,\; G_l\Phi_c G_l^\top + \Psi\right)\; \pi_c\; \rho_l}{\sum_{c=1}^{m}\sum_{l=1}^{k} N\left(x;\; G_l\mu_c,\; G_l\Phi_c G_l^\top + \Psi\right)\; \pi_c\; \rho_l}$$

$$p(z|l,c,x) = N\Big(z;\; \mathrm{E}[z|l,c,x],\; \mathrm{Cov}[z|l,c,x]\Big)$$

where

$$\mathrm{E}[z|l,c,x] = \Omega_{lc} G_l^\top \Psi^{-1} x + \Omega_{lc}\Phi_c^{-1}\mu_c$$

$$\Omega_{lc} \triangleq \mathrm{Cov}[z|l,c,x]) = (\Phi_c^{-1} + G_l^\top \Psi^{-1} G_l)^{-1}$$

The formulas for the M step are given by

$$\pi_c = \left\langle \sum_{l=1}^{k} p(l,c|x_i) \right\rangle$$

$$\rho_l = \left\langle \sum_{c=1}^{m} p(l,c|x_i) \right\rangle$$

$$\mu_c = \frac{\left\langle \sum_{l=1}^{k} \mathrm{E}[z|l,c,x_i]\; p(l,c|x_i) \right\rangle}{\left\langle \sum_{l=1}^{k} p(l,c|x_i) \right\rangle}$$

$$\phi_c = \frac{\left\langle \sum_{l=1}^{k} diag\Big[\mathrm{Cov}[z|l,c,x_i]\Big]\; p(l,c|x_i) \right\rangle}{\left\langle \sum_{l=1}^{k} p(l,c|x_i) \right\rangle}$$

$$\Psi = \left\langle \sum_{c=1}^{m}\sum_{l=1}^{k} diag\left[\Big(x_i - G_l\; \mathrm{E}[z|l,c,x_i]\Big)\Big(x_i - G_l\; \mathrm{E}[z|l,c,x_i]\Big)^\top\right]\; p(l,c|x_i) \right\rangle$$

(c) Write a Matlab function `show4(pi4,rho4,mu4,phi4,psi4,n1,n2)` which displays an `n1` $\times$ `n2` matrix image of the latent mean vectors in `mu4`, the latent variance vectors in `phi4`, and the observed variance vector `psi4`.

(d) Write a Matlab function `[img] = stabilize4(imgin,pi4,G4,rho4,mu4,phi4,psi4)` which calculates the expected value of the latent image given the observed image:

$$\mathrm{E}[z|x] \quad = \quad \sum_{c=1}^{m} \sum_{l=1}^{k} \mathrm{E}[z|x,c,l] \; p(c,l|x)$$

(e) Generate a few random images from your estimated model. Stabilize a few of the natural images from your data. Tell me what you think.

Question 5 (Discussion—2%)

(a) Informally compare the results of the 4 different models for modeling natural images.

(b) How would you objectively evaluate the performance of the 4 models?