UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
Department of Electrical and Computer Engineering

ECE 544NA Pattern Recognition

Solution 2

Fall 2013

Assigned: Thursday, September 5, 2013 Due: Thursday, September 12, 2013

Reading: NNPR Chapter 2, or PRML Chapters 2 and 9

Problem 2.1

(a) For general p,

$$\frac{\partial}{\partial \mu_{(i)}} \sum_{n=1}^{N} \ln p(\vec{x}^n|\mu_{(i)}) = -\frac{p}{\lambda_{(i)}^p} \sum_{n=1}^{N} |x_{(i)}^n - \mu_{(i)}|^{p-1} \text{sign}\left(x_{(i)}^n - \mu_{(i)}\right) = 0$$

For $p = 2$:

$$\mu_{(i)} = \frac{1}{N} \sum_{n=1}^{N} x_{(i)}^n$$

For $p = 1$,

$$\frac{\partial}{\partial \mu_{(i)}} \sum_{n=1}^{N} \ln p(\vec{x}^n|\mu_{(i)}) = -\frac{p}{\lambda_{(i)}^p} \sum_{n=1}^{N} \text{sign}\left(x_{(i)}^n - \mu_{(i)}\right)$$

which is solved by setting $\mu_{(i)} = \text{median}(x_{(i)}^n)$.

$$\frac{\partial}{\partial \lambda_{(i)}} \sum_{n=1}^{N} \ln p(\vec{x}^n|\mu_{(i)}) = \frac{p}{\lambda_{(i)}^{p+1}} \sum_{n=1}^{N} |x_{(i)}^n - \mu_{(i)}|^p - \frac{N}{\lambda_{(i)}}$$

$$\lambda_{(i)} = \left(\frac{p}{N} \sum_{n=1}^{N} |x_{(i)}^n - \mu_{(i)}|^p\right)^{1/p}$$

(b)

$$\frac{\partial}{\partial \mu_{(i)}} \left(\ln p(\mu_{(i)}) + \sum_{n=1}^{N} \ln p(\vec{x}^n|\mu_{(i)})\right) = \frac{p}{\alpha_{(i)}^p} |\mu_{(i)} - \nu_{(i)}|^{p-1} \text{sign}\left(\mu_{(i)} - \nu_{(i)}\right) - \frac{p}{\lambda_{(i)}^p} \sum_{n=1}^{N} |x_{(i)}^n - \mu_{(i)}|^{p-1} \text{sign}\left(x_{(i)}^n - \mu_{(i)}\right)$$

For $p = 2$,

$$\mu_{(i)} = \frac{\sum_{n=1}^{N} x_{(i)}^n + \left(\frac{\lambda_{(i)}}{\alpha_{(i)}}\right)^2 \nu_{(i)}}{N + \left(\frac{\lambda_{(i)}}{\alpha_{(i)}}\right)^2}$$

For $p = 1$,

$$\frac{\partial}{\partial \mu_{(i)}} \left(-\ln p(\mu_{(i)}) - \sum_{n=1}^{N} \ln p(\vec{x}^n|\mu_{(i)})\right) = \frac{p}{\lambda_{(i)}^p} \sum_{n=1}^{N} \text{sign}\left(x_{(i)}^n - \mu_{(i)}\right) + \frac{p}{\alpha_{(i)}^p} \text{sign}(\mu_{(i)} - \nu_{(i)})$$

Note: we do not have a closed form solution in this case. While $\mu_{(i)} = \text{median}(x^n_{(i)})$ sets the first part of the equation to 0, the overall derivative is either positive or negative depending on whether μ is greater than or less than ν. It is tempting to simply shift the value of $\mu_{(i)}$ by $\pm\frac{p}{\alpha^p_{(i)}}$, but that does not necessarily result in the optimum since the median is not a linear shift-invariant operation.

Matlab Exercises

Problem 2.2

Here's estep.m:

```
function gamma = estep(X,prior,mu,sigma)
% GAMMA = ESTEP(X,PRIOR,MU,SIGMA)
% GAMMA [KxN] - p(Ck|xn)
% X [DxN] - observation data
% prior [1xK] - priors of the K Gaussians
% mu [DxK] - means of the K Gaussians
% sigma [DxDxK] - covariances of the K Gaussians

K = length(prior);
[D,N] = size(X);

% Compute joint probabilities p(Ck,xn)
for k=1:K,
  lnpi(k) = log(prior(k)) - 0.5*log(det(2*pi*sigma(:,:,k)));
  prec = inv(sigma(:,:,k));
  for n=1:N,
    gamma(k,n)=exp(lnpi(k)-0.5*(X(:,n)-mu(:,k))'*prec*(X(:,n)-mu(:,k)));
  end
end
% Normalize
gamma = gamma ./ repmat(sum(gamma),[K,1]);
```

Here's mstep.m. This function needs some method to avoid singular sigmas; the method show here as "interpolate gamma with a uniform distribution" is one possibility:

```
function [prior,mu,sigma] = mstep(X,gamma)
% [PRIOR,MU,SIGMA] = MSTEP(X,GAMMA)
% prior [1xK] - priors of the K Gaussians
% mu [DxK] - means of the K Gaussians
% sigma [DxDxK] - covariances of the K Gaussians
% X [DxN] - observation data
% GAMMA [KxN] - p(Ck|xn)

[D,N]=size(X);
[K,N] = size(gamma);

% Get rid of NaNs
gamma(isnan(gamma)) = 0;
% Get rid of too-small columns
for n=find(sum(gamma)<1),
  gamma(:,n) = gamma(:,n) + (1-sum(gamma(:,n)))*ones(3,1)/3;
end
% If any prior is too small, interpolate with a uniform distribution
while any(sum(gamma') < N/200),
  disp(sprintf('Interpolating gamma because some are too small'));
```

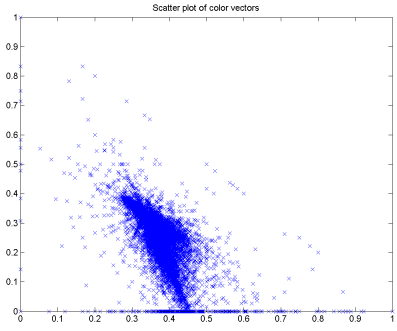
```
  disp(sum(gamma'));
  gamma = 0.9*gamma + 0.1/3;
end

nu = sum(gamma');
prior = nu/N;
scale = gamma./repmat(nu',[1 N]);
mu = X * gamma' ./ repmat(nu,[D,1]);
sigma = zeros(D,D,K);
for k=1:K, for n=1:N,
    sigma(:,:,k) = sigma(:,:,k) + ...
       gamma(k,n)*(X(:,n)-mu(:,k))*(X(:,n)-mu(:,k))'/nu(k);
end; end
```

Here's some wrapper code that calls those functions, and creates the figures:

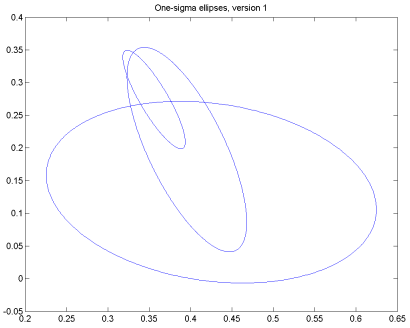
```
A = imread('everitt.jpg');
[NROWS,NCOLS,NPLANES]=size(A);
yr = double(A(:,:,1))./max(1,double(sum(A,3)));
yb = double(A(:,:,3))./max(1,double(sum(A,3)));
X = [yr(:)'; yb(:)'];
figure(1);
plot(X(1,:),X(2,:),'x');
title('Scatter plot of color vectors');
print -dpng fig1.png;

for version=1:2,
  % Initial parameter values
  oldprior = 0.33*ones(1,3);
  oldsigma = repmat(0.01*eye(2,2),[1 1 3]);
  if version==1,
    oldmu = [[0;0],[1;0],[0;1]];
  else,
    rand3 = rand(3,3);
    oldmu = rand3([1 3],:)./repmat(sum(rand3),[2,1]);
  end

  % Re-estimate until mu stops changing
  gamma = estep(X,oldprior,oldmu,oldsigma);
  [prior,mu,sigma] = mstep(X,gamma);
  initialdelta = sum(sum(abs(mu-oldmu)));
  t=0;
  while sum(sum(abs(mu-oldmu))) > 0.02*initialdelta,
    t=t+1;
    disp(sprintf('Iteration %d, delta is %g',t,sum(sum(abs(mu-oldmu)))));
    oldprior=prior; oldmu=mu; oldsigma=sigma;
    gamma = estep(X,oldprior,oldmu,oldsigma);
    [prior,mu,sigma] = mstep(X,gamma);
  end

  % One-sigma ellipses
```

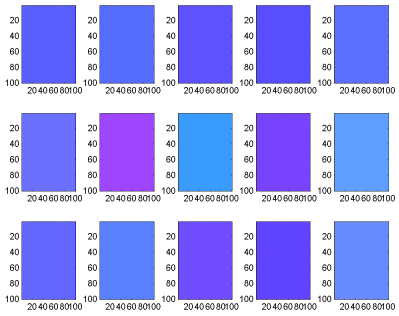
```
  figure(2);
  for k=1:3,
    [V(:,:,k),D(:,:,k)]=eig(sigma(:,:,k));
    phi = [0:0.01:(2*pi)];
    ell = repmat(mu(:,k),[1,length(phi)]) +...
      sqrt(D(1,1,k))*V(:,1,k)*cos(phi)+sqrt(D(2,2,k))*V(:,2,k)*sin(phi);
    plot(ell(1,:),ell(2,:),'-');
    hold on;
  end
  hold off;
  title(sprintf('One-sigma ellipses, version %d',version));
  print('-dpng',sprintf('fig2_%d.png',version));

  % Color centroids and standard deviations
  figure(3);
  Y2C = [1,0,0;-1,-1,1;0,0,1];
  for k=1:3,
    hues=repmat(mu(:,k),[1,5])+sqrt(D(:,:,k))*V(:,:,k)*[0,1,-1,0,0;0,0,0,1,-1];
    colors=[1,0,0;-1,-1,1;0,0,1]*[hues;ones(1,5)];
    for s=1:5,
      subplot(3,5,(k-1)*5+s);
      image(repmat(permute(colors(:,s),[3,2,1]),[100,100,1]));
    end
  end
  print('-dpng',sprintf('fig3_%d.png',version));
end
```

Here's the image, and its scatter plot:



Here are the ellipses and colors for the first starting point, with means set initially to the extremes of the triangle. The means don't wind up staying very extreme!





Here are the ellipses and colors for the second starting point, with randomly initialized means. It actually winds up pretty close to the other one: