1. (a) $I(X,Y) = \sum_y \sum_x p(x,y) \log \frac{p(x,y)}{p(x)\,p(y)}$

$= \sum_y \sum_x p(x,y) \left[\log \frac{p(x,y)}{p(x)} - \log p(y) \right]$

$= \sum_y \sum_x p(x,y) \left[\log p(y|x) - \log p(y) \right]$

$= \sum_y \sum_x p(x,y)(-\log p(y)) + \sum_y \sum_x p(x,y) \log p(y|x)$

$= -\sum_y \log p(y) \left(\sum_x p(x,y) \right) - \left[-\left(\sum_y \sum_x p(y|x)\, p(x) \log p(y|x) \right) \right]$

$= -\sum_y \log p(y) \cdot p(y) - \left[-\sum_x p(x) \sum_y p(y|x) \log p(y|x) \right]$

$= H(Y) - \sum_x p(x)\, H(Y|X=x) = H(Y) - H(Y|X)$

Since $I(X,Y)$ is symmetric for X, Y, Similarly we can get $I(X,Y) = H(X) - H(X|Y)$

(b)   From (a) we know, $I(X,Y) = H(X) - H(X|Y)$, =>

$I(X,Y) = H(X) + \sum_x p(x) \sum_y p(y|x) \log p(y|x)$. Since $x = f(y)$, then

$p(y|x) = 0$ or 1, for each case, $\sum_x p(x) \sum_y p(y|x) \log p(y|x) = 0$, then

$I(X,Y) = H(X) + 0 = H(X)$. Similarly, $I(X,Y) = H(Y)$

(c) $\min_\theta D_{KL}(\hat{p}\|q) = H(\hat{p}, q) - H(\hat{p}) \propto H(\hat{p}, q)$

(Noticing only q contains θ here)

$H(\hat{p}, q) = -\sum_x \hat{p}(x) \log q(x|\theta)$

$= -\sum_x \frac{1}{N} \sum_{i=1}^{N} 1_{\{x=x_i\}} \log q(x|\theta)$

$= -\frac{1}{N} \sum_{i=1}^{N} \log(q(x_i|\theta))$

Thus $\min_\theta D_{KL}(\hat{p}\|q) \iff \min_\theta -\frac{1}{N} \sum_{i=1}^{N} \log q(x_i|\theta)$

$\iff \max_\theta \sum_{i=1}^{N} \log q(x_i|\theta)$

$\iff \max_\theta \prod_{i=1}^{N} q(x_i|\theta) \longrightarrow$ likelyhood

Thus the minimum Kullback - Leibler divergence is obtained by the maximum likelihood.

(d) For continuous variable,

we have $H(x) = -\int p(x) \ln p(x)\, dx$.

From the question, it has three constraint:

s.t.
$$\begin{cases} \int p(x)\, dx = 1 \\ \int x\, p(x)\, dx = \mu \\ \int (x-\mu)^2 p(x)\, dx = \sigma^2 \end{cases}$$

Use Lagrange multiplies to max $H(x)$:

$$L = -\int p(x) \ln p(x)\, dx + \lambda_1 \left(\int p(x)\, dx - 1\right) + \lambda_2 \left(\int x\, p(x)\, dx - \mu\right) + \lambda_3 \left(\int (x-\mu)^2 p(x)\, dx - \sigma^2\right)$$

$$\frac{dL}{dp(x)} = -\ln p(x) - 1 + \lambda_1 + \lambda_2 x + \lambda_3 (x-\mu)^2 = 0$$

$$\Rightarrow p(x) = e^{-1 + \lambda_1 + \lambda_2 x + \lambda_3 (x-\mu)^2}$$

with
$$\begin{cases} \int e^{-1+\lambda_1+\lambda_2 x+\lambda_3(x-\mu)^2}\, dx = 1 \\ \int x\, e^{-1+\lambda_1+\lambda_2 x+\lambda_3(x-\mu)^2}\, dx = \mu \\ \int (x-\mu)^2 e^{-1+\lambda_1+\lambda_2 x+\lambda_3(x-\mu)^2}\, dx = \sigma^2 \end{cases} \Rightarrow p(x) = \frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$\therefore p = N(\mu, \sigma^2)$ is the solution of max $H(x)$

$\Leftrightarrow H(q) \le H(p)$ for q be any probability density

with mean μ and variance σ^2.

2.

(a) $\text{Dirichlet} (p|\alpha) = \dfrac{\Gamma\left(\sum\limits_{1}^{m}\alpha_k\right)}{\prod\limits_{k=1}^{m}\Gamma(\alpha_k)} \prod\limits_{k=1}^{m} p_k^{\alpha_k-1}$

$= \exp\left(\log \dfrac{\Gamma\left(\sum\limits_{1}^{m}\alpha_k\right)}{\prod\limits_{k=1}^{m}\Gamma(\alpha_k)} \prod\limits_{k=1}^{m} p_k^{\alpha_k-1}\right)$

$= \exp\left[\log\Gamma\left(\sum\limits_{1}^{m}\alpha_k\right) + \sum\limits_{1}^{m}(\alpha_k-1)\log p_k - \sum\limits_{1}^{m}\log\Gamma(\alpha_k)\right]$

$= \exp\left[\sum\limits_{1}^{m}(\alpha_k-1)\log p_k - \left(\sum\limits_{1}^{m}\log\Gamma(\alpha_k) - \log\Gamma\left(\sum\limits_{1}^{m}\alpha_k\right)\right)\right]$

Thus, set $\eta(\alpha) = \begin{pmatrix}\alpha_1-1\\\alpha_2-1\\\vdots\\\alpha_m-1\end{pmatrix}$, $T(p) = \begin{pmatrix}\log p_1\\\log p_2\\\vdots\\\log p_m\end{pmatrix}$,

$A(\alpha) = \sum\limits_{1}^{m}\log\Gamma(\alpha_k) - \log\Gamma\left(\sum\limits_{1}^{m}\alpha_k\right)$. Then $D(p|\alpha) = \exp\left[\eta(\alpha)^T T(p) - A(\alpha)\right]$

(b) The Dirichlet log-likehood function $F(\alpha) = \log p(D|\alpha) = \sum\limits_{j=1}^{N}\log D(p^j|\alpha)$

$\therefore F(\alpha) = \sum\limits_{j=1}^{N}\left[\sum\limits_{k=1}^{m}(\alpha_k-1)\log p_k^j + A(\alpha)\right]$

$= \sum\limits_{k=1}^{m}(\alpha_k-1)\sum\limits_{j=1}^{N}\log p_k^j + N A(\alpha)$

$= N\sum\limits_{k=1}^{m}(\alpha_k-1)\dfrac{1}{N}\sum\limits_{j=1}^{N}\log p_k^j + N\left[\log\Gamma\left(\sum\limits_{k=1}^{m}\alpha_k\right) - \sum\limits_{k=1}^{m}\log\Gamma(\alpha_k)\right]$

$= N\left[\sum\limits_{k=1}^{m}(\alpha_k-1)\hat{t}_k + \log\Gamma\left(\sum\limits_{k=1}^{m}\alpha_k\right) - \sum\limits_{k=1}^{m}\log\Gamma(\alpha_k)\right]$, $\hat{t}_k = \dfrac{1}{N}\sum\limits_{j=1}^{N}\log p_k^j$

(c) $\dfrac{\partial F}{\partial \alpha_k} = N\left(\hat{t}_k + \dfrac{\partial\log\Gamma\left(\sum\limits_{k=1}^{m}\alpha_k\right)}{\partial \sum\limits_{k=1}^{m}\alpha_k}\cdot 1 - \dfrac{\partial\log\Gamma(\alpha_k)}{\partial\alpha_k}\right)$

$= N\left(\hat{t}_k + \Psi\left(\sum\limits_{k=1}^{m}\alpha_k\right) - \Psi(\alpha_k)\right)$, $k=1,\cdots m$

$\therefore \dfrac{\partial F}{\partial \alpha} = \left(\dfrac{\partial F}{\partial\alpha_1}, \dfrac{\partial F}{\partial\alpha_2}, \cdots \dfrac{\partial F}{\partial\alpha_m}\right)$
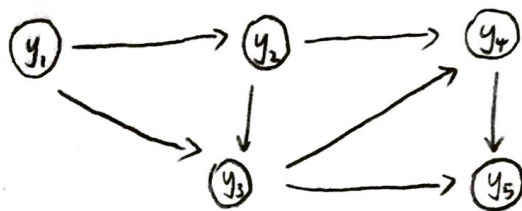
(d) $\dfrac{\partial^2 F}{\partial \alpha_k^2} = N\left(\Psi'\left(\sum\limits_{k=1}^{m}\alpha_k\right) - \Psi'(\alpha_k)\right)$, $\dfrac{\partial^2 F}{\partial\alpha_k\partial\alpha_j} = N\Psi'\left(\sum\limits_{k=1}^{m}\alpha_k\right)$

Thus set $q_{kk} = -N\Psi'(\alpha_k)$, $C = N\Psi'\left(\sum\limits_{k=1}^{m}\alpha_k\right)$.

then $\nabla_\alpha^2 F(\alpha) = \left(\dfrac{\partial^2 F}{\partial\alpha_k\partial\alpha_j}\right)_{k,j=1,\cdots m} = Q + C\,\mathbf{1}\mathbf{1}^T$

(e) $\alpha^{new} = \alpha^{old} - \left[H_F(\alpha^{old})\right]^{-1}\nabla F(\alpha^{old})$

$= \alpha^{old} - \left[Q + C\,\mathbf{1}\mathbf{1}^T\right]^{-1}\cdot\nabla F(\alpha^{old})$

$= \alpha^{old} - \left(Q^{-1} - \dfrac{Q^{-1}C\,\mathbf{1}\mathbf{1}^T Q^{-1}}{1 + \mathbf{1}^T Q^{-1} C\mathbf{1}}\right)\nabla F(\alpha^{old})$
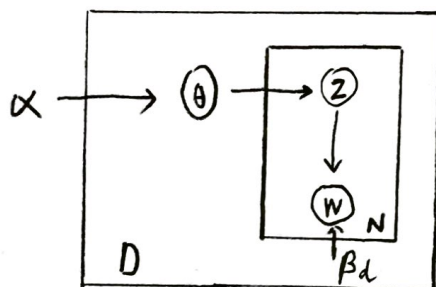
3. (a) (i) $P(y_1, y_2, y_3, y_4, y_4, y_5) = P(y_1)P(y_2) \prod_{k=3}^{5} P(y_k | y_{k-1}, y_{k-2})$

(ii) $P(x_1, \cdots x_N, y_1, \cdots y_N) = P(y_1) \prod_{k=2}^{N} P(y_k | y_{k-1}) \prod_{k=1}^{N} P(x_k | y_k)$

(b)
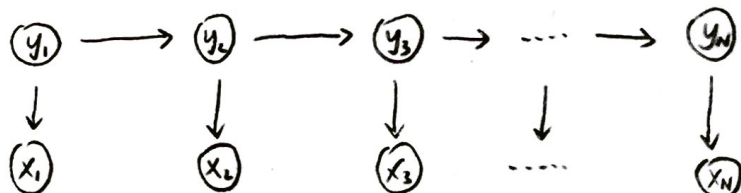
(c) ①: $P(r, \theta, \phi, z) = \prod_{i=1}^{M} \left[P(r_i) P(\theta_i | r_i) \prod_{j=1}^{N} P(\phi_{ij}) P(z_{ij} | \phi_{ij}) \right]$

②: $P(z, w) = \prod_{i=1}^{M} \left[P(z_i) \prod_{j=1}^{N} P(w_{ij} | z_i) \right]$

5. (a) $f(y|x;\theta) = \sum_{k=1}^{k} \pi_k \phi(y; w_k^T x + b_k, \sigma_k^2)$

$\Rightarrow f(y|x;\theta) = \prod_{i=1}^{N} \sum_{k=1}^{k} \pi_k \phi(y_i; w_k^T x_i + b_k, \sigma_k^2)$

$\therefore \log f(y|x;\theta) = \sum_{i=1}^{N} \log \sum_{k=1}^{k} \pi_k \phi(y_i; w_k^T x_i + b_k, \sigma_k^2)$

(b) $f(y|x, z=k;\theta) = \phi(y; w_k^T x + b_k, \sigma_k^2)$,

$f(y, z|x;\theta) = \pi_z \phi(y; w_z^T x + b_z, \sigma_z^2)$, Let $\delta_{ik} = 1_{\{z_i=k\}}$

Then $\log f(y, z|x;\theta) = \sum_{k=1}^{k} \sum_{i=1}^{N} 1_{\{z_i=k\}} \log \pi_{z_i} \phi(y_i; w_{z_i}^T x_i + b_{z_i}, \sigma_{z_i}^2)$

$= \sum_{k=1}^{k} \sum_{i=1}^{N} \delta_{ik} \log \pi_k \phi(y_i; w_k^T x_i + b_k, \sigma_k^2)$

(c) $Q(\theta, \theta^{old}) = E_z[\log f(y, z|x;\theta) | y, x; \theta^{old}]$

Let $\gamma_{ik} = p(z_i=k | y_i, x_i; \theta^{old})$

$\gamma_{ik} = \frac{f(z_i, y_i|x_i; \theta^{old})}{f(y_i|x_i; \theta^{old})} = \frac{\pi_k^{old} \phi(y_i, w_k^{T,old} x_i + b_k^{old}, \sigma_k^{old})}{\sum_{i=1}^{k} \pi_k^{old} \phi(y_i, w_k^{T,old} x_i + b_k^{old}, \sigma_k^{old})}$

$Q(\theta, \theta^{old}) = \sum_{i=1}^{N} \sum_{k=1}^{K} E_q[1_{\{z_i=k\}} \log \pi_k \phi(y, w_k^T x + b_k, \sigma_k^2)]$

$= \sum_{i=1}^{N} \sum_{k=1}^{K} \gamma_{ik} \log \pi_k \phi(y, w_k^T x + b_k, \sigma_k^2)$

$= \sum_{i=1}^{N} \sum_{k=1}^{K} \gamma_{ik} \log \pi_k + \sum_{i=1}^{N} \sum_{k=1}^{K} \gamma_{ik} \log \phi(y, w_k^T x + b_k, \sigma_k^2)$

(d) $\theta^{new} = \arg\max_\theta Q(\theta, \theta^{old})$, s.t. $\sum_{k=1}^{K} \pi_k = 1$, $\forall \pi_k > 0$ for $k = 1, \cdots K$

let $f = Q(\theta, \theta^{old}) + \lambda(\sum_{k=1}^{K} \pi_k - 1) - \sum_{k=1}^{K} \alpha_k \pi_k$, $\alpha_k = 0$, $\forall k = 1 \cdots K$

$\frac{\partial f}{\partial \pi_k} = \sum_{i=1}^{N} \frac{\gamma_{ik}}{\pi_k} + \lambda = 0 \Rightarrow \pi_k = \frac{\sum_{i=1}^{N} \gamma_{ik}}{-\lambda}$

$\frac{\partial f}{\partial \lambda} = \sum_{k=1}^{K} \pi_k - 1 = 0 \Rightarrow \sum_{k=1}^{K} \sum_{i=1}^{N} \gamma_{ik} / -\lambda - 1 = 0 \Rightarrow \lambda = -N \Rightarrow \pi_k = \frac{\sum_{i=1}^{N} \gamma_{ik}}{K}$

$\frac{\partial f}{\partial w_k} = \sum_{i=1}^{N} \gamma_{ik} \left(\frac{\log \phi(y, w_k^T x + b_k, \sigma_k^2)}{\partial w_k} \right) = \sum_{i=1}^{N} \gamma_{ik} \frac{\partial (\log \frac{1}{\sqrt{2\pi}\sigma_k} + \frac{y_i - [w_k^T x_i + b_k])^2}{2\sigma_k})}{\partial w_k}$

$= \sum_{i=1}^{N} \gamma_{ik} \frac{2(y_i - w_k^T x_i + b_k)}{2\sigma_k}(-x_i) = 0$

$\frac{\partial f}{\partial b_k} = \sum_{i=1}^{N} \gamma_{ik} \frac{\partial (\log \frac{1}{\sqrt{2\pi}\sigma_k} + \frac{(y_i - (w_k^T x_i + b_k)^2}{2\sigma_k})}{\partial b_k} = \sum_{i=1}^{N} \gamma_{ik} \frac{y_i - (w_k^T x_i + b_k)}{\sigma_k} \cdot (-1) = 0$

Set $\tilde{w}_k = \begin{bmatrix} w_k \\ b_k \end{bmatrix}$, $\tilde{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} = [x, 1]$, $y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}$

then we can combine $\frac{\partial f}{\partial w_k}$ & $\frac{\partial f}{\partial b_k}$ to get $\sum_{i=1}^{N} \gamma_{ik} (y_i - \tilde{w}_k^T \tilde{x}_i) \tilde{x}_i = 0$

Use closed form to represent the whole question, we can set the objective function as

$\sum_{i=1}^{N} \gamma_{ik} (y_i - \tilde{w}_k^T \tilde{x}_i)^2$ can be written as $(\tilde{x}\tilde{w}_k - y)^T R (\tilde{x}\tilde{w}_k - y)$

where $R = \text{diag}(\gamma_{1k}, \gamma_{2k}, \cdots \gamma_{Nk})$

$\therefore \sum_{i=1}^{N} \gamma_{ik} (y_i - \tilde{w}_k^T \tilde{x}_i) \tilde{x}_i = 2x^T R x w - 2x^T R y = 0$

$\Rightarrow \tilde{w}_k = (x^T R x)^{-1} x^T R y$

$\dfrac{\partial f}{\partial \sigma_k} = \dfrac{\partial \sum_{i=1}^{N} \gamma_{ik} \left(\log \frac{1}{\sqrt{2\pi \sigma_k^2}} + \left(-\frac{1}{2\sigma_k^2} (y_i - \tilde{w}_k^T \tilde{x}_i)^2 \right) \right)}{\partial \sigma_k}$
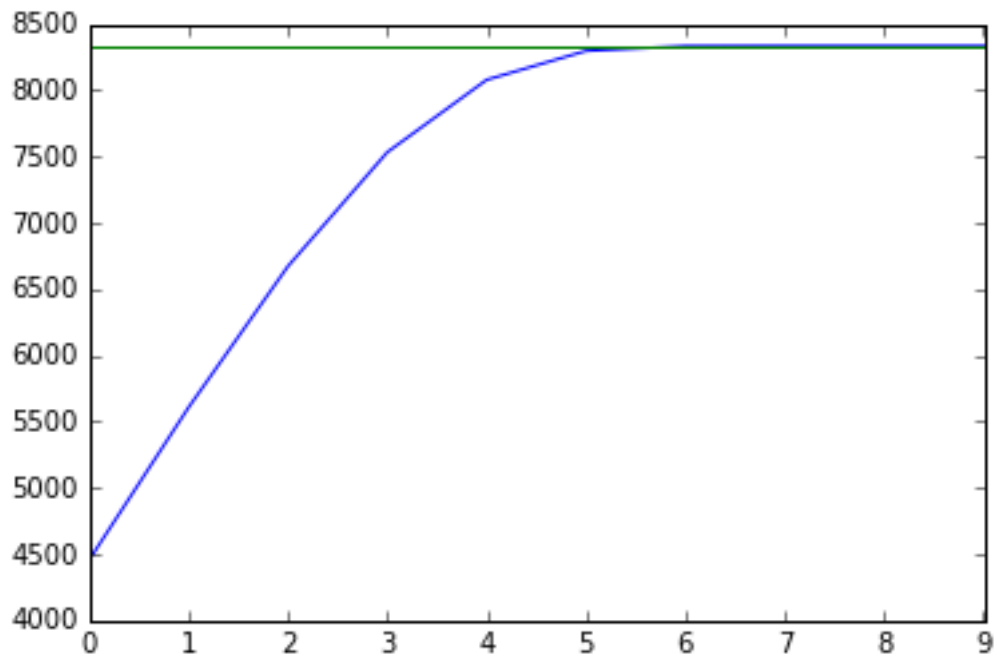
$= \sum_{i=1}^{N} \gamma_{ik} + \frac{1}{\sigma_k^2} \sum_{i=1}^{N} (y_i - \tilde{w}_k^T \tilde{x}_i)^2 \gamma_{ik} = 0$

$\Rightarrow \sigma_k^2 = \frac{1}{\sum_{i=1}^{N} \gamma_{ik}} \sum_{i=1}^{N} (y_i - \tilde{w}_k^T \tilde{x}_i)^2 \gamma_{ik}$

$= \frac{1}{\sum_{i=1}^{N} \gamma_{ik}} (\tilde{x}\tilde{w}_k - y)^T R (\tilde{x}\tilde{w}_k - y)$

2.(f)

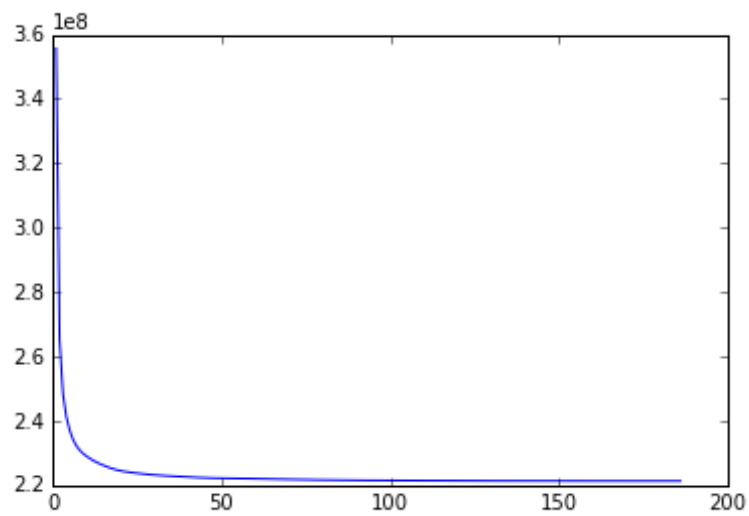The plot of the log-likelihood vs iteration is as follows:

The estimated model parameters are:

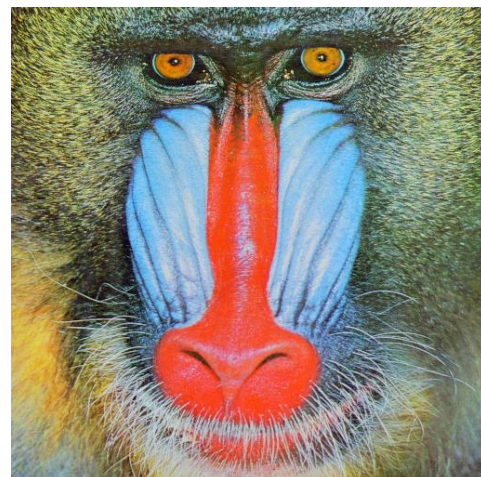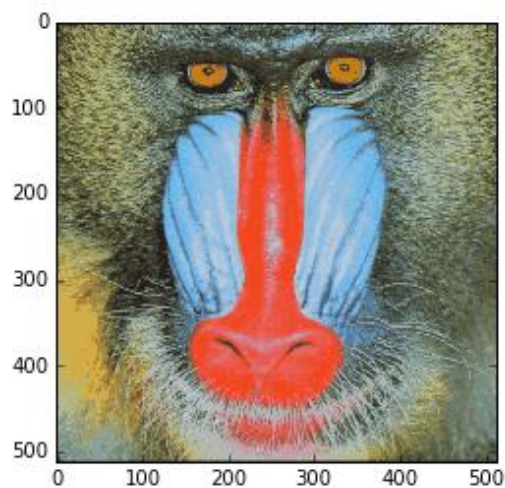9.96979256,   4.90985978,   14.86165346,   19.66495057,   48.97431779

4.(a)

Put the two pictures together:

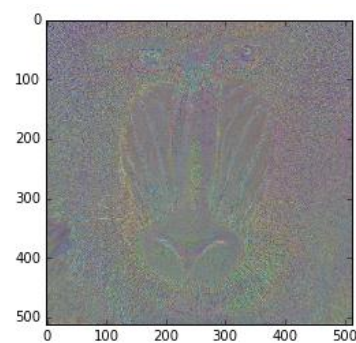| Compressed	Original

Comparing the compressed picture with the original one, we can find the large parts with single color are best preserved. For the border parts where the color changes from one to another, they are generally not preserved well.

The picture of the difference:

The compression ratio is :

$$r = \frac{24 * 2^2 * 64 + 216^2 * log_2 64}{24 * 512^2} = 6.3477\%$$

The relative mean absolute error of the compresses image is: 0.05
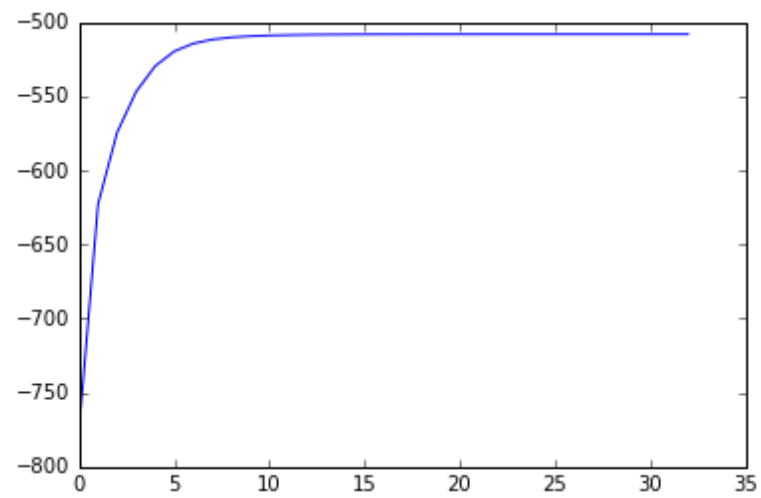
4.(b)
The number of 24 bits per pixel needed:

$$\text{bpp} = \frac{24 * M^2 * K + \left(\frac{N}{M}\right)^2 * log_2 K}{N^2}$$

the compress ratio is:

$$\text{ratio} = \text{bpp}/24 = \frac{24 * M^2 * K + \left(\frac{N}{M}\right)^2 * log_2 K}{24N^2}$$
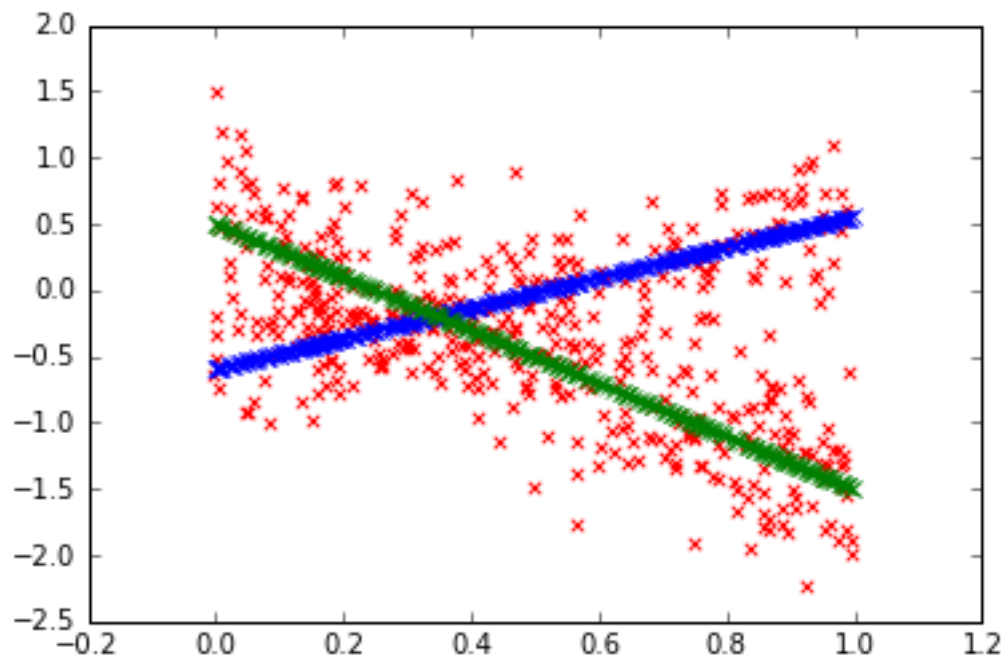
5.(e)

The plot of the log-likelihood is :

The estimated model parameters:

$$\pi = (0.293, 0.707), \quad (w, b) = \begin{pmatrix} 0.960 & -0.474 \\ -2.063 & 0.492 \end{pmatrix}, \quad \sigma^2 = 0.1778$$

The plot showing the data and estimated lines is :

Codes:

```
#2.f
import numpy as np
from matplotlib import pyplot as plt
from scipy.special import gammaln, polygamma
from __future__ import division
N = 1000
m = 5
alpha = np.array([10, 5, 15, 20, 50])
P = np.random.dirichlet(alpha, N)
t=np.sum(np.log(P),axis=0)/N
t=t.reshape((5,1))
U=np.ones((5,1))
alpha0=np.ones((5,1))
loglihd=np.zeros((1000,1))
for i in range(1000):
    dalpha1=N*(t+polygamma(0,np.sum(alpha0))*U-polygamma(0,alpha0));
    g1=-N*polygamma(1,alpha0)
    Q=np.diag(np.diag(np.ones((5,5))*g1))
    C=N*polygamma(1,np.sum(alpha0))
    Q1=np.linalg.inv(Q)
    alphan=alpha0-(Q1-
(Q1.dot(C*U.dot(U.T)).dot(Q1))/(1+C*U.T.dot(Q1).dot(U))).dot(dalpha1)
    loglihd[i]=N*((alphan-1).T.dot(t)+gammaln(np.sum(alphan))-
np.sum(gammaln(alphan)))
    if i==0:
        alpha0=alphan
    elif i > 0 and np.abs(loglihd[i]-loglihd[i-1])>0.0001:
        alpha0=alphan
    else:
        alphafnl=alphan
        break
lstd=N*((alpha-1).T.dot(t)+gammaln(np.sum(alpha))-np.sum(gammaln(alpha)))
x=np.linspace(0,9,10);y=loglihd[0:10];stdl=lstd*np.ones((10,1))
plt.plot(x,y);plt.plot(x,stdl)



#4.a
import numpy as np
from matplotlib import pyplot as plt
from __future__ import division
from scipy.ndimage import imread
mandrill = imread('mandrill.png', mode='RGB').astype(float)
N = int(mandrill.shape[0])
```

```python
M = 2;k = 64
X = np.zeros((N**2//M**2, 3*M**2))
for i in range(N//M):
    for j in range(N//M):
        X[i*N//M+j,:] = mandrill[i*M:(i+1)*M,j*M:(j+1)*M,:].reshape(3*M**2)
Jf=np.zeros((300,1))
#calculating tagfet value J
def calcJ(data,centers):
    diffsq=(centers[:,np.newaxis,:]-data)**2
    return np.sum(np.min(np.sum(diffsq,axis=2),axis=0))
#implement k means
def kmeans(data,k):
#initializing centers and list J
    centers=data[np.random.choice(range(data.shape[0]),k,replace=False),:]
    J=[];
#closest center for each sample
    for itera in range(300):
        sqdistances=np.sum((centers[:,np.newaxis,:]-data)**2,axis=2)
        closest=np.argmin(sqdistances,axis=0)
#calculate J and append to list
        J.append(calcJ(data,centers))
        Jf[itera]=calcJ(data,centers)
#update clusters
        for i in range(k):
            centers[i,:]=data[closest==i,:].mean(axis=0)
#decide whether stopping
        if itera>0 and np.abs(Jf[itera]-Jf[itera-1])==0:
            X=centers[closest,:]
            break
        else:
            continue
    J.append(calcJ(data,centers))
    return J,centers,closest

JF,centersf,closestf=kmeans(X,64)
Xnew=centersf[closestf,:]
mandrillnew=np.zeros(512*512*3)
mandrillnew=mandrillnew.reshape(512,512,3)
for i in range(256):
    for j in range(256):
        mandrillnew[i*2:(i+1)*2,j*2:(j+1)*2,:]=Xnew[i*256+j,:].reshape(2,2,3)
plt.imshow(mandrillnew/255)
plt.show()
mandrillgrey=mandrillnew-mandrill+128*np.ones(512*512*3).reshape(512,512,3)
```

```
plt.imshow(mandrillgrey/255)
plt.show()
Jf1=Jf[Jf>0]
x1=np.linspace(1,186,186)
plt.plot(x1,Jf1)
error=np.sum(np.abs(mandrillnew-mandrill))/(3*255*512**2)


#5.e
from __future__ import division
import numpy as np
from matplotlib import pyplot as plt
from scipy.stats import norm as norm
# Generate the data according to the specification in the homework description

N = 500
x = np.random.rand(N)

pi0 = np.array([0.7, 0.3])
w0 = np.array([-2, 1])
b0 = np.array([0.5, -0.5])
sigma0 = np.array([.4, .3])

y = np.zeros_like(x)
for i in range(N):
    k = 0 if np.random.rand() < pi0[0] else 1
    y[i] = w0[k]*x[i] + b0[k] + np.random.randn()*sigma0[k]

ccpiest = np.array([0.5, 0.5]);cwest = np.array([1, -1])
cbest = np.array([0, 0]);sigmaest = np.array([np.std(y), np.std(y)])
cwest2 = np.array([cwest,cbest]).T;x2 = np.array([x,np.ones(N)])
p1 = ccpiest[0]*norm(cwest2[0].dot(x2),sigmaest[0]).pdf(y)
p2 = ccpiest[1]*norm(cwest2[1].dot(x2),sigmaest[1]).pdf(y)
r1 = p1/(p1+p2);r2 = p2/(p1+p2);r = np.array([r1,r2])
Q1 = np.sum(r1*np.log(ccpiest[0]*norm(cwest2[0].dot(x2),sigmaest[0]).pdf(y)))
Q2 = np.sum(r2*np.log(ccpiest[1]*norm(cwest2[1].dot(x2),sigmaest[1]).pdf(y)))
Q = Q1+Q2;diff = 1;ll = [];ite = [];count = 0

for i in range(100):
    ll.append(Q);ite.append(count);tmp = Q
    ccpiest = np.array([np.mean(r1),np.mean(r2)])
    w1 = np.linalg.inv(x2.dot(np.diag(r1)).dot(x2.T)).dot(x2).dot(np.diag(r1)).dot(y)
    w2 = np.linalg.inv(x2.dot(np.diag(r2)).dot(x2.T)).dot(x2).dot(np.diag(r2)).dot(y)
    cwest2 = np.array([w1,w2])
```

```python
        sigmaest1 = ((np.sum(r1*(y-cwest2[0].dot(x2))**2))/np.sum(r1))**0.5
        sigmaest2 = ((np.sum(r2*(y-cwest2[1].dot(x2))**2))/np.sum(r2))**0.5
        sigmaest = np.array([sigmaest1,sigmaest2])
        p1 = ccpiest[0]*norm(cwest2[0].dot(x2),sigmaest[0]).pdf(y)
        p2 = ccpiest[1]*norm(cwest2[1].dot(x2),sigmaest[1]).pdf(y)
        r1 = p1/(p1+p2);r2 = p2/(p1+p2);r = np.array([r1,r2])
        Q1 = np.sum(r1*np.log(ccpiest[0]*norm(cwest2[0].dot(x2),sigmaest[0]).pdf(y)))
        Q2 = np.sum(r2*np.log(ccpiest[1]*norm(cwest2[1].dot(x2),sigmaest[1]).pdf(y)))
        Q = Q1+Q2;diff = Q-tmp;count = count+1
        if diff>=1e-4:
                continue
        else:
                break
plt.plot(x, cwest2[0].dot(x2), c='b', marker='x')
plt.plot(x, cwest2[1].dot(x2), c='g', marker='x')
plt.scatter(x, y, c='r', marker='x')
plt.plot(ite,ll)
```