

DD2434 - Machine Learning, Advanced Course
Assignment 1A

Tristan Perrot
tristanp@kth.se

Étienne Riguet
riguet@kth.se

November 2023



Contents

1	Exponential Family	3
2	Dependencies in a Directed Graphical Model	4
3	CAVI	9
A	Appendix	17
A.1	Question 3.12	17
A.2	Question 3.13	17
A.3	Question 3.14	17
A.4	Question 3.15	19

1 Exponential Family

Question 1.1

$$\begin{aligned}
 p(x|\theta) &= h(x) \exp(\eta(\theta) \cdot T(x) - A(\eta)) \\
 &= h(x) \exp(\eta(\lambda) \cdot T(x) - A(\eta(\lambda))) \\
 &= h(x) \exp(\log \lambda \cdot x - A(\log \lambda)) \\
 &= h(x) \exp(\log \lambda \cdot x - \lambda) \\
 &= h(x) \exp(\log \lambda \cdot x) \exp(-\lambda) \\
 &= e^{-\lambda} \frac{\lambda^x}{x!}
 \end{aligned} \tag{1}$$

We can see that the distribution correspond to a Poisson distribution of parameter λ .

Question 1.2

$$\begin{aligned}
 p(x|\theta) &= h(x) \exp(\eta(\theta) \cdot T(x) - A(\eta)) \\
 &= \exp(\eta([\alpha, \beta]) \cdot [\log x, x] - A(\alpha - 1, -\beta)) \\
 &= \exp([\alpha - 1, -\beta] \cdot [\log x, x] - \log \Gamma(\alpha) + \alpha \log(\beta)) \\
 &= \exp((\alpha - 1) \log x - \beta x - \log \Gamma(\alpha) + \alpha \log(\beta)) \\
 &= \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}
 \end{aligned} \tag{2}$$

We can see that the distribution correspond to a Gamma distribution of parameters α and β .

Question 1.3

$$\begin{aligned}
 p(x|\theta) &= h(x) \exp(\eta(\theta) \cdot T(x) - A(\eta)) \\
 &= \frac{\exp(\eta([\mu, \sigma^2]) \cdot [x, x^2] - A(\eta([\mu, \sigma^2])))}{\sqrt{2\pi}} \\
 &= \frac{\exp([\frac{\mu}{\sigma^2}, -\frac{1}{2\sigma^2}] \cdot [x, x^2] - A([\frac{\mu}{\sigma^2}, -\frac{1}{2\sigma^2}]))}{\sqrt{2\pi}} \\
 &= \frac{\exp(\frac{\mu x}{\sigma^2} - \frac{x^2}{2\sigma^2} - \frac{\mu^2}{2\sigma^2} - \log \sigma)}{\sqrt{2\pi}} \\
 &= \frac{\exp(-\frac{(x-\mu)^2}{2\sigma^2})}{\sigma\sqrt{2\pi}}
 \end{aligned} \tag{3}$$

We can see that the distribution correspond to a Normal distribution of parameters μ and σ^2 .

Question 1.4

$$\begin{aligned}
 p(x|\theta) &= h(x) \exp(\eta(\theta) \cdot T(x) - A(\eta)) \\
 &= 2 \exp(\eta(\lambda) \cdot x - A(\eta(\lambda))) \\
 &= 2 \exp(-\lambda x - A(-\lambda)) \\
 &= 2 \exp\left(-\lambda x + \log\left(\frac{\lambda}{2}\right)\right) \\
 &= \lambda e^{-\lambda x}
 \end{aligned} \tag{4}$$

We can see that the distribution correspond to a Exponential distribution of parameter λ .

Question 1.5

$$\begin{aligned}
 p(x|\theta) &= h(x) \exp(\eta(\theta) \cdot T(x) - A(\eta)) \\
 &= \exp(\eta([\psi_1, \psi_2]) \cdot [\log x, \log(1-x)] - A(\eta([\psi_1, \psi_2]))) \\
 &= \exp([\psi_1 - 1, \psi_2 - 1] \cdot [\log x, \log(1-x)] - A([\psi_1 - 1, \psi_2 - 1])) \\
 &= \exp((\psi_1 - 1) \log x + (\psi_2 - 1) \log(1-x) - \log \Gamma(\psi_1) - \log \Gamma(\psi_2) + \log \Gamma(\psi_1 + \psi_2)) \\
 &= \frac{\Gamma(\psi_1 + \psi_2)}{\Gamma(\psi_1) \Gamma(\psi_2)} x^{\psi_1 - 1} (1-x)^{\psi_2 - 1}
 \end{aligned} \tag{5}$$

We can see that the distribution correspond to a Beta distribution of parameters ψ_1 and ψ_2 .

2 Dependencies in a Directed Graphical Model

Question 2.6

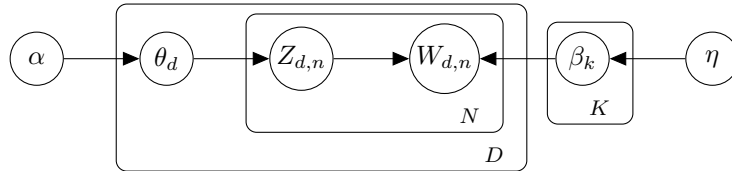
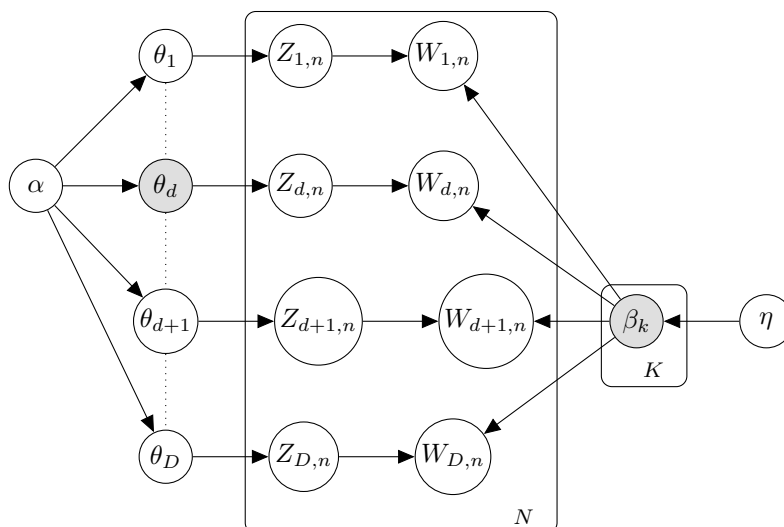
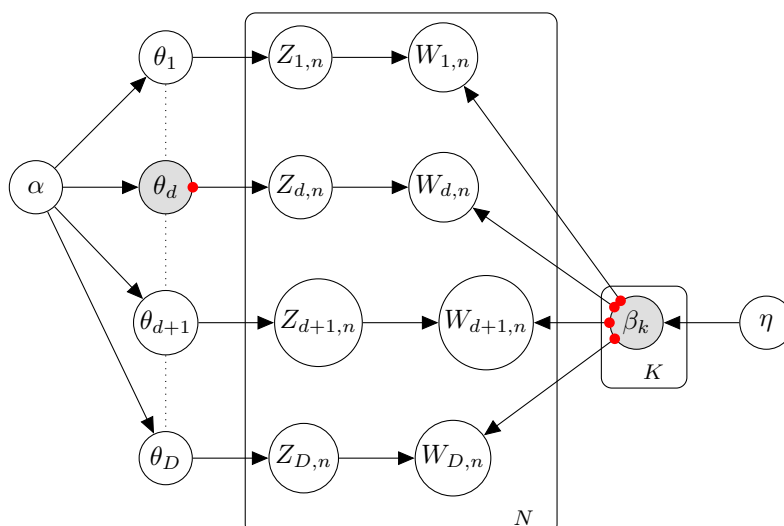


Figure 1: Graphical model of smooth LDA.

The Bayes net take this form :



Then, if we use the method using the d-separation, we obtain this :



Therefore, we can see that $W_{d,n} \perp W_{d,n+1} | \theta_d, \beta_{1:K}$ is true.

Question 2.7

The Bayes net take this form (with d-separation marks) :



Therefore, we can see that $\theta_d \perp \theta_{d+1} | Z_{d,1:N}$ is false.

Question 2.8

The Bayes net take this form (with d-separation marks) :



Therefore, we can see that $\theta_d \perp \theta_{d+1} | \alpha, Z_{1:D,1:N}$ is true.

Question 2.9



Figure 2: Graphical model of Labeled LDA.

The Bayes net take this form (with d-separation marks) :



Therefore, we can see that $W_{d,n} \perp W_{d,n+1} | \Lambda_d, \beta_{1:K}$ is false.

Question 2.10

The Bayes net take this form (with d-separation marks) :



Therefore, we can see that $\theta_d \perp \theta_{d+1} | Z_{d,1:N}, Z_{d+1,1:N}$ is false.

Question 2.11

The Bayes net take this form (with d-separation marks) :



Therefore, we can see that $\Lambda_d \perp \Lambda_{d+1} | \Phi, Z_{1:D,1:N}$ is false.

3 CAVI

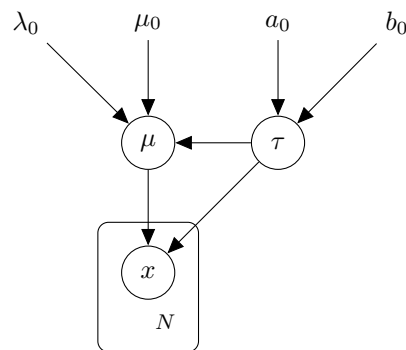


Figure 3: DGM

Question 3.12

In the bishop book, we can see that :

$$p(X|\mu, \tau) = \left(\frac{\tau}{2\pi}\right)^{N/2} \exp \left\{ -\frac{\tau}{2} \sum_{n=1}^N (x_n - \mu)^2 \right\} \quad (6)$$

$$p(\mu|\tau) = \mathcal{N}(\mu|\mu_0, (\lambda_0\tau)^{-1}) \quad (7)$$

$$p(\tau) = \text{Gam}(\tau|a_0, b_0) \quad (8)$$

Then, by using the code in appendix A.1, we obtain :

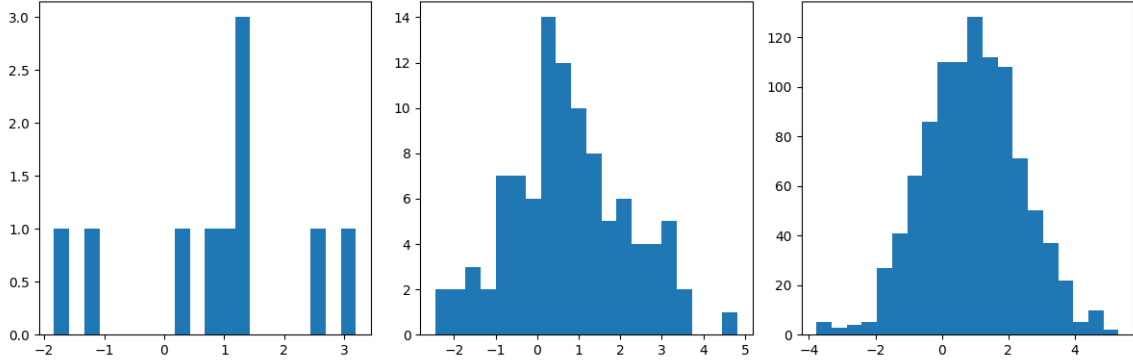


Figure 4: Generated Data

Question 3.13

Let's find the ML estimates of μ and τ . We know that $\log(q^*(\mu)) = \mathbb{E}_{-\mu}[\log p(X, \mu, \tau)]$. Then, we can write :

$$\begin{aligned} \log(q^*(\mu)) &= \mathbb{E}_{-\mu}[\log p(X, \mu, \tau)] \\ &\stackrel{+}{=} \mathbb{E}_{\tau}[\log p(X|\mu, \tau) + \log p(\mu|\tau)] \\ &= \mathbb{E}_{\tau} \left[\frac{N}{2} \log \left(\frac{\tau}{2\pi} \right) - \frac{\tau}{2} \sum_{n=1}^N (x_n - \mu)^2 + \frac{1}{2} \log \left(\frac{\lambda_0\tau}{2\pi} \right) - \frac{\lambda_0\tau}{2} (\mu - \mu_0)^2 \right] \\ &\stackrel{+}{=} -\frac{\mathbb{E}_{\tau}[\tau]}{2} \left(\lambda_0(\mu - \mu_0)^2 + \sum_{n=1}^N (x_n - \mu)^2 \right) \\ &= -\frac{\mathbb{E}_{\tau}[\tau]}{2} \left(\lambda_0\mu^2 - 2\lambda_0\mu\mu_0 + \lambda_0\mu_0^2 + \sum_{n=1}^N x_n^2 - 2\mu \sum_{n=1}^N x_n + N\mu^2 \right) \\ &= -\frac{\mathbb{E}_{\tau}[\tau]}{2} \left((\lambda_0 + N)\mu^2 - 2(\lambda_0\mu_0 + \sum_{n=1}^N x_n)\mu + \lambda_0\mu_0^2 + \sum_{n=1}^N x_n^2 \right) \\ &= -\frac{\mathbb{E}_{\tau}[\tau](\lambda_0 + N)}{2} \left(\mu^2 - 2\mu \frac{\lambda_0\mu_0 + \sum_{n=1}^N x_n}{\lambda_0 + N} + \frac{\lambda_0\mu_0^2 + \sum_{n=1}^N x_n^2}{\lambda_0 + N} \right) \end{aligned} \quad (9)$$

Therefore we can conclude that $q^*(\mu) = \mathcal{N}(\mu|\mu_N, \lambda_N^{-1})$ with :

$$\mu_N = \frac{\lambda_0 \mu_0 + \sum_{n=1}^N x_n}{\lambda_0 + N} \quad (10)$$

$$\lambda_N = (\lambda_0 + N) \mathbb{E}[\tau] \quad (11)$$

And for τ we have :

$$\begin{aligned} \log(q^*(\tau)) &= \mathbb{E}_{-\tau}[\log p(X, \mu, \tau)] \\ &\stackrel{\pm}{=} \mathbb{E}_{\mu}[\log p(X|\mu, \tau) + \log p(\mu|\tau)] + \log p(\tau) \\ &\stackrel{\pm}{=} (a_0 - 1) \log \tau - b_0 \tau + \frac{N+1}{2} \log \tau - \frac{\tau}{2} \mathbb{E}_{\mu} \left[\sum_{n=1}^N (x_n - \mu)^2 + \lambda_0 (\mu - \mu_0)^2 \right] \\ &= (a_0 + \frac{N+1}{2} - 1) \log \tau - \left(b_0 + \frac{1}{2} \mathbb{E}_{\mu} \left[\sum_{n=1}^N (x_n - \mu)^2 + \lambda_0 (\mu - \mu_0)^2 \right] \right) \tau \end{aligned} \quad (12)$$

Therefore we can conclude that $q^*(\tau) = \text{Gam}(\tau|a_N, b_N)$ with :

$$a_N = a_0 + \frac{N+1}{2} \quad (13)$$

$$\begin{aligned} b_N &= b_0 + \frac{1}{2} \mathbb{E}_{\mu} \left[\sum_{n=1}^N (x_n - \mu)^2 + \lambda_0 (\mu - \mu_0)^2 \right] \\ b_N &= b_0 + \frac{1}{2} \left(\sum_{n=1}^N x_n^2 + N \mathbb{E}_{\mu}[\mu^2] - 2 \mathbb{E}_{\mu}[\mu] \sum_{n=1}^N x_n + \lambda_0 (\mathbb{E}_{\mu}[\mu^2] + \mu_0^2 - 2 \mu_0 \mathbb{E}_{\mu}[\mu]) \right) \end{aligned} \quad (14)$$

With :

$$\begin{aligned} \mathbb{E}_{q(\mu)}[\mu] &= \mu_N \\ \mathbb{E}_{q(\mu)}[\mu^2] &= \frac{1}{\lambda_N} + \mu_N^2 \\ \mathbb{E}_{q(\tau)}[\tau] &= \frac{a_N}{b_N} \end{aligned} \quad (15)$$

If we take non-informative priors then $a_0 = b_0 = \mu_0 = \lambda_0 = 0$, then we have :

$$\begin{aligned} \mu_N &= \bar{x} \\ \lambda_N &= N \mathbb{E}[\tau] \\ a_N &= \frac{N+1}{2} \\ b_N &= \frac{1}{2} \mathbb{E}_{\mu} \left[\sum_{n=1}^N (x_n - \mu)^2 \right] \end{aligned} \quad (16)$$

And by using $\mathbb{E}[\tau] = \frac{a_N}{b_N}$ we obtain :

$$\begin{aligned}\frac{1}{\mathbb{E}[\tau]} &= \frac{b_N}{a_N} \\ \frac{1}{\mathbb{E}[\tau]} &= \frac{2}{2(N+1)} \mathbb{E}_\mu \left[\sum_{n=1}^N (x_n - \mu)^2 \right] \\ \frac{1}{\mathbb{E}[\tau]} &= \frac{N}{N+1} \left(\bar{x}^2 - 2\bar{x}\mathbb{E}[\mu] + \mathbb{E}[\mu^2] \right)\end{aligned}\tag{17}$$

And, with the fact that $\mathbb{E}[\mu] = \mu_N$ and $\mathbb{E}[\mu^2] = \frac{1}{\lambda_N} + \mu_N^2$, we obtain :

$$\begin{aligned}\mathbb{E}[\mu] &= \bar{x} \\ \mathbb{E}[\mu^2] &= \frac{1}{N\mathbb{E}[\tau]} + \bar{x}^2\end{aligned}\tag{18}$$

And therefore:

$$\begin{aligned}\frac{1}{\mathbb{E}[\tau]} &= \frac{N}{N+1} \left(\bar{x}^2 - 2\bar{x}^2 + \frac{1}{N\mathbb{E}[\tau]} + \bar{x}^2 \right) \Leftrightarrow \frac{1}{\mathbb{E}[\tau]} - \frac{1}{(N+1)\mathbb{E}[\tau]} = \frac{N}{N+1} (\bar{x}^2 - \bar{x}^2) \\ &\Leftrightarrow \frac{N+1-1}{(N+1)\mathbb{E}[\tau]} = \frac{N}{N+1} (\bar{x}^2 - \bar{x}^2) \\ &\Leftrightarrow \frac{1}{\mathbb{E}[\tau]} = (\bar{x}^2 - \bar{x}^2) \\ &\Leftrightarrow \frac{1}{\mathbb{E}[\tau]} = \frac{1}{N} \sum_{n=1}^N (x_n - \bar{x})^2\end{aligned}\tag{19}$$

Which define the ML estimates. The implementation is in the code in appendix A.2.

Question 3.14

The posterior is defined as $p(\mu, \tau|x)$. Then, we can write :

$$\begin{aligned}p(\mu, \tau|x) &= \frac{p(x|\mu, \tau)p(\mu, \tau)}{p(x)} \\ &\propto p(x|\mu, \tau)p(\mu, \tau)\end{aligned}\tag{20}$$

Where $x|\mu, \tau \sim \mathcal{N}(\mu|\mu, \tau^{-1})$ and $\mu, \tau \sim \text{NormalGamma}(\mu_0, \lambda_0, a_0, b_0)$. Therefore, as we saw in the question 1.3 in the Module 1 exercise, we have $\mu, \tau|x \sim \text{NormalGamma}(\mu', \lambda', a', b')$, where :

$$\begin{aligned}\mu' &= \frac{N\bar{x} + \mu_0\lambda_0}{N + \lambda_0} \\ \lambda' &= N + \lambda_0 \\ a' &= a_0 + \frac{N}{2} \\ b' &= b_0 + \frac{1}{2} \left(\sum_{n=1}^N x_n^2 + \lambda_0\mu_0^2 - \frac{(N\bar{x} + \mu_0\lambda_0)^2}{N + \lambda_0} \right)\end{aligned}\tag{21}$$

Therefore, if we plot the contour for each datasets we obtain :

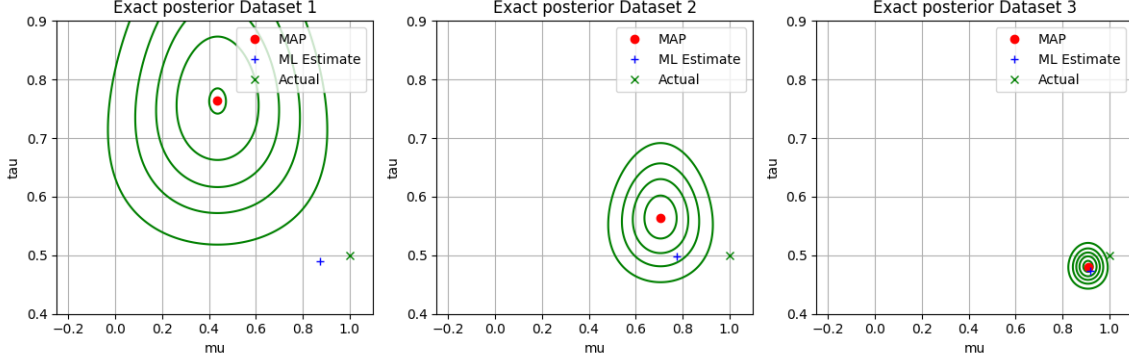


Figure 5: Contours of exact posteriors by datasets

The rest of the answer is in the code in appendix A.3.

Question 3.15

The equation (10.24) in the Bishop is the mean-field approximation which is :

$$q(\mu, \tau) = q(\mu)q(\tau) \quad (22)$$

This time, we take the result of the question 3.13 without setting the priors to 0. Then, we have :

$$\begin{aligned} q(\mu) &= \mathcal{N}(\mu | \mu_N, \lambda_N^{-1}) \\ q(\tau) &= \text{Gam}(\tau | a_N, b_N) \end{aligned} \quad (23)$$

with updates equations in the cavi algorithm described by :

$$\begin{aligned} \mu_N &= \frac{\lambda_0 \mu_0 + N \bar{x}}{\lambda_0 + N} \\ \lambda_N &= (\lambda_0 + N) \mathbb{E}[\tau] \\ a_N &= a_0 + \frac{N + 1}{2} \\ b_N &= b_0 + \frac{1}{2} \left(\sum_{n=1}^N x_n^2 + N \mathbb{E}_\mu[\mu^2] - 2 \mathbb{E}_\mu[\mu] \sum_{n=1}^N x_n + \lambda_0 (\mathbb{E}_\mu[\mu^2] + \mu_0^2 - 2 \mu_0 \mathbb{E}_\mu[\mu]) \right) \end{aligned} \quad (24)$$

and the expectations are the ones described in the equation (15).

Now, we need to find the ELBO formula :

$$\begin{aligned} \mathcal{L}(q) &= \mathbb{E}_{q(\mu), q(\tau)}[\log p(X, \mu, \tau)] - \mathbb{E}_{q(\mu), q(\tau)}[\log q(\mu, \tau)] \\ &= \mathbb{E}_{q(\mu), q(\tau)}[\log p(X | \mu, \tau) + \log p(\mu, \tau)] - \mathbb{E}_{q(\mu), q(\tau)}[\log q(\mu) + \log q(\tau)] \\ &= \mathbb{E}_{q(\mu), q(\tau)}[\log p(X | \mu, \tau)] + \mathbb{E}_{q(\mu), q(\tau)}[\log p(\mu, \tau)] - \mathbb{E}_{q(\mu)}[\log q(\mu)] - \mathbb{E}_{q(\tau)}[\log q(\tau)] \\ &= \mathbb{E}_{q(\mu), q(\tau)}[\log p(X | \mu, \tau)] + \mathbb{E}_{q(\mu), q(\tau)}[\log p(\mu, \tau)] + \mathbb{H}_q[\mu] + \mathbb{H}_q[\tau] \end{aligned} \quad (25)$$

If we compute the first term we have:

$$\begin{aligned}\mathbb{E}_{q(\mu),q(\tau)}[\log p(X|\mu, \tau)] &= \mathbb{E}_{q(\mu),q(\tau)} \left[\frac{N}{2} \log \left(\frac{\tau}{2\pi} \right) - \frac{\tau}{2} \sum_{n=1}^N (x_n - \mu)^2 \right] \\ &= \frac{N}{2} (\mathbb{E}_{q(\tau)}[\log \tau] - \log(2\pi)) \\ &\quad - \frac{\mathbb{E}_{q(\tau)}[\tau]}{2} \left(\sum_{n=1}^N x_n^2 - 2\mathbb{E}_{q(\mu)}[\mu]N\bar{x}_n + N\mathbb{E}_{q(\mu)}[\mu^2] \right)\end{aligned}\tag{26}$$

And the second one is:

$$\begin{aligned}\mathbb{E}_{q(\mu),q(\tau)}[\log p(\mu, \tau)] &= \mathbb{E}_{q(\mu),q(\tau)} \left[\log \left(\frac{b_0^{a_0} \sqrt{\lambda_0}}{\Gamma(a_0) \sqrt{2\pi}} \right) + (a_0 - \frac{1}{2}) \log \tau - b_0 \tau - \frac{\lambda_0 \tau (\mu - \mu_0)^2}{2} \right] \\ &= \log \left(\frac{b_0^{a_0} \sqrt{\lambda_0}}{\Gamma(a_0) \sqrt{2\pi}} \right) + (a_0 - \frac{1}{2}) \mathbb{E}_{q(\tau)}[\log \tau] - b_0 \mathbb{E}_{q(\tau)}[\tau] \\ &\quad - \frac{\lambda_0 \mathbb{E}_{q(\tau)}[\tau]}{2} (\mathbb{E}_{q(\mu)}[\mu^2] - 2\mu_0 \mathbb{E}_{q(\mu)}[\mu] + \mu_0^2)\end{aligned}\tag{27}$$

And we can compute all of that because entropies are known and we have the following expectations:

$$\begin{aligned}\mathbb{E}_{q(\mu)}[\mu] &= \mu_N \\ \mathbb{E}_{q(\mu)}[\mu^2] &= \frac{1}{\lambda_N} + \mu_N^2 \\ \mathbb{E}_{q(\tau)}[\tau] &= \frac{a_N}{b_N} \\ \mathbb{E}_{q(\tau)}[\log \tau] &= \psi(a_N) - \log b_N\end{aligned}\tag{28}$$

Then we obtain this result :

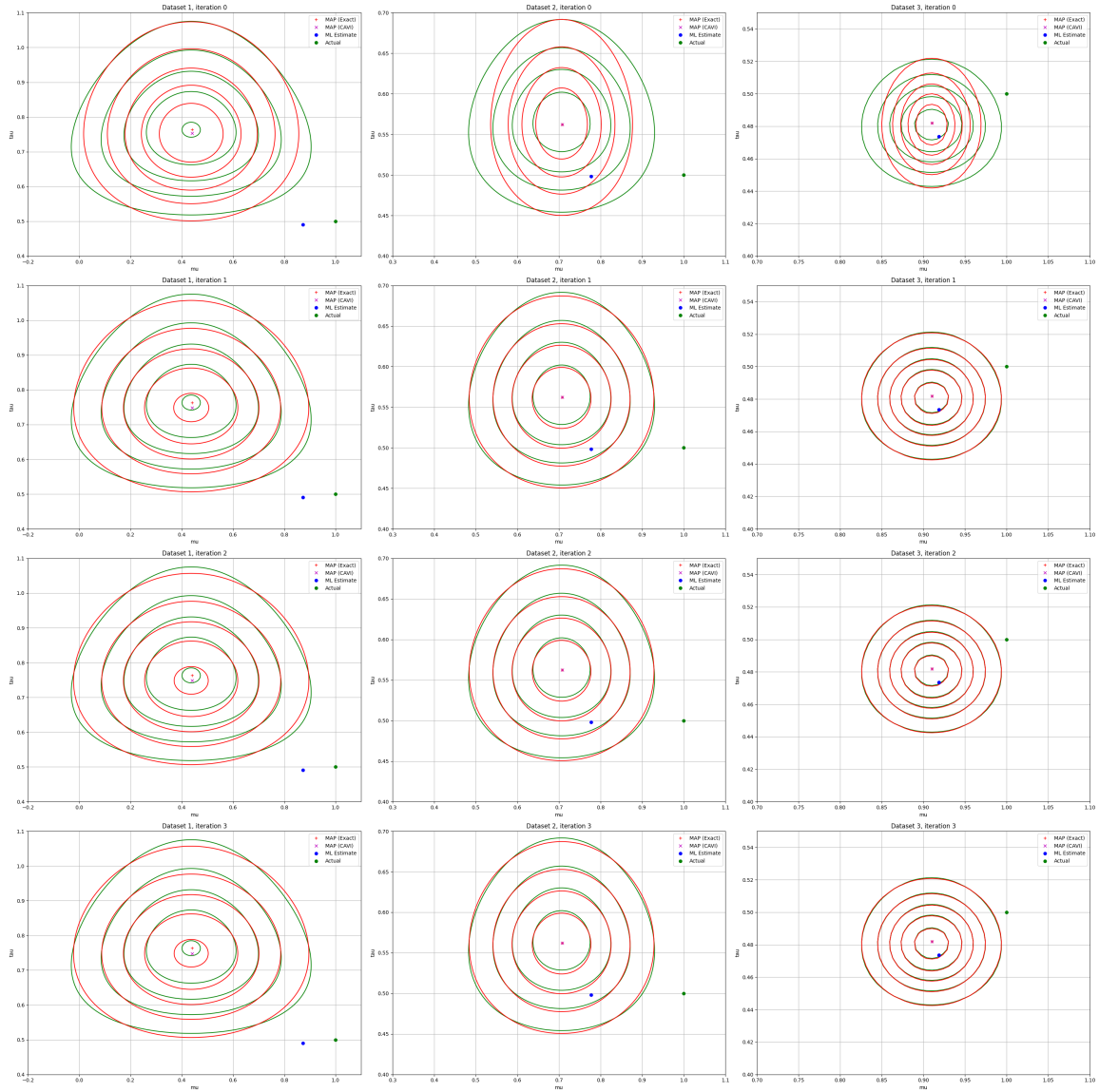


Figure 6: Contours of the approximations by VI and the exact posterior by datasets, by iterations

And we obtain an elbo plot :

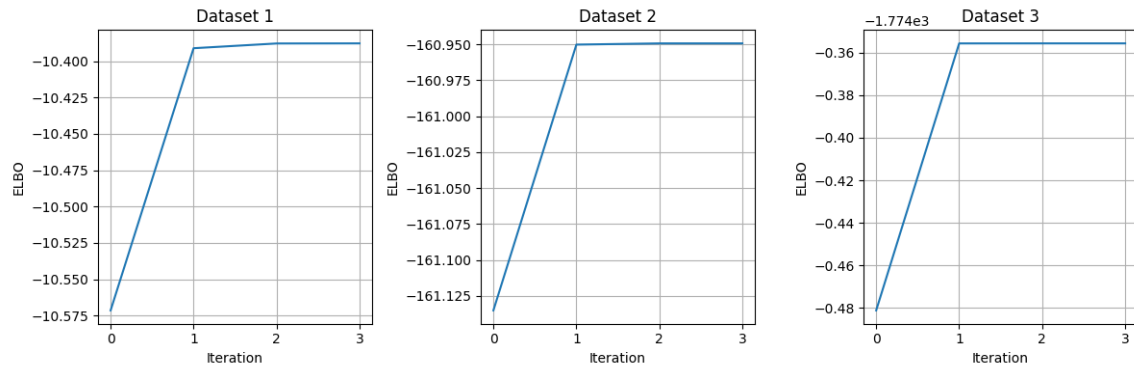


Figure 7: ELBO plot by datasets

The code is in appendix A.4.

A Appendix

A.1 Question 3.12

```
import numpy as np
from scipy.stats import gamma, norm
from scipy.special import psi
from scipy.special import gamma as gamma_func
np.random.seed(14)

def generate_data(mu, tau, N):
    # Insert your code here
    D = np.random.normal(mu, np.sqrt(1/tau), N)

    return D

MU = 1
TAU = 0.5

dataset_1 = generate_data(MU, TAU, 10)
dataset_2 = generate_data(MU, TAU, 100)
dataset_3 = generate_data(MU, TAU, 1000)

# Visualize the datasets via histograms
# Insert your code here
fig, axs = plt.subplots(1, 3, figsize=(12, 4))
axs[0].hist(dataset_1, bins=20)
axs[1].hist(dataset_2, bins=20)
axs[2].hist(dataset_3, bins=20)
plt.tight_layout()
plt.savefig('../images/12_data.png')
plt.show()
```

A.2 Question 3.13

```
def ML_est(data):
    # insert your code
    N = len(data)
    x_mean = np.mean(data)
    x_var = np.var(data)

    tau_ml = 1 / x_var
    mu_ml = x_mean

    return mu_ml, tau_ml
```

A.3 Question 3.14

```
def compute_exact_posterior(D, a_0, b_0, mu_0, lambda_0):
    # your implementation
    x_mean = np.mean(D)
    N = len(D)

    mu_prime = (lambda_0 * mu_0 + N * x_mean) / (lambda_0 + N)
    lambda_prime = lambda_0 + N
    a_prime = a_0 + N / 2
    b_prime = b_0 + 0.5 * (np.sum(D**2) +
                          lambda_0 * mu_0**2 - lambda_prime * mu_prime**2)

    exact_post_distribution = (a_prime, b_prime, mu_prime, lambda_prime)

    return exact_post_distribution

# prior parameters
mu_0 = 0
lambda_0 = 10
a_0 = 20
b_0 = 20

mus = np.linspace(-0.25, 1.1, 200)
taus = np.linspace(0.4, 0.9, 200)

fig, axs = plt.subplots(1, 3, figsize=(12, 4))
for i, dataset in enumerate([dataset_1, dataset_2, dataset_3]):
    mu_ml, tau_ml = ML_est(dataset)

    a_T, b_T, mu_T, lambda_T = compute_exact_posterior(
        dataset, a_0, b_0, mu_0, lambda_0)

    Z_exact = np.zeros((len(mus), len(taus)))
    pTau = gamma(a=a_T, loc=0, scale=1/b_T)
    for j, tau in enumerate(taus):
        pMu = norm(loc=mu_T, scale=1/np.sqrt(lambda_T*tau))
        Z_exact[:, j] = pMu.pdf(mus) * pTau.pdf(tau)
    # Finding the maximum of the exact posterior
    mu_max_exact = mus[np.argmax(np.max(Z_exact, axis=1))]
    tau_max_exact = taus[np.argmax(np.max(Z_exact, axis=0))]
    # Plotting the results
    axs[i].contour(*np.meshgrid(mus, taus), Z_exact.T,
                  levels=5, colors=['green'])
    axs[i].plot(mu_max_exact, tau_max_exact, 'ro', label='MAP')
    axs[i].plot(mu_ml, tau_ml, 'b+', label='ML Estimate')
    axs[i].plot(MU, TAU, 'gx', label='Actual')
    axs[i].legend()
    axs[i].grid()
    axs[i].set_xlabel('mu')
    axs[i].set_ylabel('tau')
    axs[i].set_title('Exact posterior Dataset {}'.format(i+1))
```

```
plt.tight_layout()
plt.savefig('../images/14_contours.png')
plt.show()
```

A.4 Question 3.15

```
def compute_E_tau(a_N, b_N):
    E_tau = a_N / b_N

    return E_tau

def compute_E_mu_2(mu_N, lambda_N):
    E_mu_2 = mu_N**2 + 1/lambda_N

    return E_mu_2

def compute_E_log_tau(a_N, b_N):
    E_log_tau = psi(a_N) - np.log(b_N)

    return E_log_tau

def compute_elbo(D, a_0, b_0, mu_0, lambda_0, a_N, b_N, mu_N, lambda_N):
    N = len(D)
    x_mean = np.mean(D)
    x_2_sum = np.sum(D**2)
    E_tau = compute_E_tau(a_N, b_N)
    E_mu_2 = compute_E_mu_2(mu_N, lambda_N)
    E_log_tau = compute_E_log_tau(a_N, b_N)

    # compute the elbo
    # E[log p(D|mu, tau)]
    E_log_p_D = N/2 * (E_log_tau - np.log(2*np.pi)) - 0.5 * \
        E_tau * (x_2_sum - 2*N*x_mean*mu_N + N*E_mu_2)

    # E[log p(mu, tau)]
    E_log_p_mu_tau = a_0*np.log(b_0) + 0.5*np.log(lambda_0) - np.log(
        gamma_func(a_0)) - 0.5*np.log(
            2*np.pi) - (a_0-0.5)*E_log_tau - b_0*E_tau - 0.5*lambda_0*E_tau*(
                E_mu_2 + mu_0**2 - 2*mu_0*mu_N)

    # Entropy of mu
    entropy_mu = norm.entropy(loc=mu_N, scale=1/np.sqrt(lambda_N))
    # Entropy of tau
    entropy_tau = gamma.entropy(a=a_N, scale=1/b_N)

    elbo = E_log_p_D + E_log_p_mu_tau + entropy_mu + entropy_tau

    return elbo
```

```
def CAVI(D, a_0, b_0, mu_0, lambda_0, iter=5):
    # make an initial guess for the expected value of tau
    E_tau = 1

    N = len(D)
    x_mean = np.mean(D)
    x_2_sum = np.sum(D**2)

    # Constants
    a_N = a_0 + (N+1) / 2
    mu_N = (lambda_0 * mu_0 + N * x_mean) / (lambda_0 + N)
    E_mu = mu_N

    # Variables
    b_Ns = []
    lambda_Ns = []

    # ELBO
    elbos = []

    # CAVI iterations ...
    for i in range(iter):
        # update the values for the variational parameters
        lambda_N = (lambda_0 + N) * E_tau

        E_mu_2 = compute_E_mu_2(mu_N, lambda_N)
        b_N = b_0 + 0.5 * (x_2_sum + N*E_mu_2 - 2*N*E_mu*x_mean +
                           lambda_0*(E_mu_2 - 2*E_mu*mu_0 + mu_0**2))

        E_tau = compute_E_tau(a_N, b_N)

        b_Ns.append(b_N)
        lambda_Ns.append(lambda_N)
        # save ELBO for each iteration, plot them afterwards to show
        # convergence
        elbos.append(compute_elbo(D, a_0, b_0, mu_0,
                                   lambda_0, a_N, b_N, mu_N, lambda_N))

    return a_N, b_N, mu_N, lambda_N, elbos, b_Ns, lambda_Ns

def compute_z_exact(mus, taus, a_, b_, mu_, lambda_):
    z = np.zeros((len(mus), len(taus)))
    pTau = gamma(a=a_, loc=0, scale=1/b_)
    for j, tau in enumerate(taus):
        pMu = norm(loc=mu_, scale=1/np.sqrt(lambda_*tau))
        z[:, j] = pMu.pdf(mus) * pTau.pdf(tau)

    return z
```

```
def compute_z_cavi(mus, taus, a_, b_, mu_, lambda_):
    pTau = gamma(a=a_, loc=0, scale=1/b_)
    pMu = norm(loc=mu_, scale=1/np.sqrt(lambda_))
    z = np.outer(pMu.pdf(mus), pTau.pdf(taus))
    return z

iter = 4 # number of iterations for CAVI
mus = np.linspace(-0.2, 1.1, 200)
taus = np.linspace(0.1, 1.1, 200)

xlims = [[-0.2, 1.1], [0.3, 1.1], [0.7, 1.1]]
ylims = [[0.4, 1.1], [0.4, 0.7], [0.4, 0.55]]

elbos_list = []

fig, axs = plt.subplots(iter, 3, figsize=(30, 30))
for i, dataset in enumerate([dataset_1, dataset_2, dataset_3]):
    mu_ml, tau_ml = ML_est(dataset)
    a_N, b_N, mu_N, lambda_N, elbos, b_Ns, lambda_Ns = CAVI(
        dataset, a_0, b_0, mu_0, lambda_0, iter=iter)
    a_T, b_T, mu_T, lambda_T = compute_exact_posterior(
        dataset, a_0, b_0, mu_0, lambda_0)

    elbos_list.append(elbos)

    for j in range(iter):
        Z_exact = compute_z_exact(mus, taus, a_T, b_T, mu_T, lambda_T)
        Z_cavi = compute_z_cavi(mus, taus, a_N, b_Ns[j], mu_N, lambda_Ns[j])
        # Finding the maximum of the exact posterior
        mu_max_exact = mus[np.argmax(np.max(Z_exact, axis=1))]
        tau_max_exact = taus[np.argmax(np.max(Z_exact, axis=0))]
        # Finding the maximum of the CAVI approximation
        mu_max_cavi = mus[np.argmax(np.max(Z_cavi, axis=1))]
        tau_max_cavi = taus[np.argmax(np.max(Z_cavi, axis=0))]
        # Plotting the results
        axs[j, i].contour(*np.meshgrid(mus, taus), Z_exact.T,
                           levels=5, colors=['green'])
        axs[j, i].contour(*np.meshgrid(mus, taus), Z_cavi.T,
                           levels=5, colors=['red'])
        axs[j, i].plot(mu_max_exact, tau_max_exact, 'r+', label='MAP (Exact)')
        axs[j, i].plot(mu_max_cavi, tau_max_cavi, 'mx', label='MAP (CAVI)')
        axs[j, i].plot(mu_ml, tau_ml, 'bo', label='ML Estimate')
        axs[j, i].plot(MU, TAU, 'go', label='Actual')
        axs[j, i].legend()
        axs[j, i].grid()
        axs[j, i].set_xlabel('mu')
        axs[j, i].set_ylabel('tau')
        axs[j, i].set_title(f'Dataset {i+1}, iteration {j}')
        axs[j, i].set_xlim(xlims[i])
```

```
        axs[j, i].set_ylim(ylims[i])
plt.tight_layout()
plt.savefig('../images/15_contours.png')
plt.show()

# Plot ELBOs
fig, axs = plt.subplots(1, 3, figsize=(12, 4))
for i in range(3):
    axs[i].plot(elbos_list[i])
    axs[i].set_xlabel('Iteration')
    axs[i].set_ylabel('ELBO')
    axs[i].set_title(f'Dataset {i+1}')
    axs[i].grid()
plt.tight_layout()
plt.savefig('../images/15_elbo.png')
plt.show()
```