DD2434, Machine Learning, Advance Course
Assignment 2, 31/1-18
Shadman Ahmed

I Graphical Models

2.1 Dependences in a Directed Graphical Model

Question 1: *Which pairs of variables, not including X, are dependent conditioned on X?*

The given model is not a fully connected graphical model where a node cannot track back to itself. The direct links between the nodes tells about the conditional distribution of nodes. Hence, for which pair of nodes that are dependent conditioned on X are:

E is dependent on C conditioned on X. F is dependent on D conditioned on X. But, A and B are just conditioned on X but no dependant on each other.

Question 2: *Which pairs of variables, not including X, are dependent, not conditioned on X*?

The node X is said to be head-to-tail, as we do not include X, still we consider the path between the nodes. Answer:

A is dependent on E, but not conditioned on X. So are:

A - F , B - E, B - F and C - E , D - F

2.2 The Sum-HMM

Due to changes of the grade system, I would like to replace task 2.2 with 2.6.

2.3 Simple VI

Question 8: *Implement the VI algorithm for the variational distribution in Equation (10.24) in Bishop.*

The implantation of factorized variational approximation of the posterior distribution by Gaussian- Gamma conjugate was done by first calculating the optimum factors:

$$q_\mu(\mu) = N\left(\mu\middle|\mu_N, \frac{1}{\lambda_N}\right) \text{ and } q_\tau(\tau) = Gamma(\tau|a_N, b_N).$$

The approximation of the posterior distribution is thus by first assuming that the factors for the mean μ and precision τ are independent, $q(\mu, \tau) = q_\mu(\mu)q_\tau(\tau)$.

Deriving the mean, precision and parameters from the factors was done by using the expressions according to [1], with $\{x_i\}_{i=1}^N$ data points:

$\mu_N = \frac{\lambda_0\mu_0+N\bar{x}}{\lambda_0+N}$, $\lambda_N = (\lambda_0 + N)\frac{a_N}{b_N}$, $a_N = a_0 + \frac{N+1}{2}$,

$b_N = b_0 + \frac{1}{2}[(\lambda_0 + N)\left(\frac{1}{\lambda_N} + \mu_N^2\right) - 2(\lambda_0\mu_0 + \sum_{n=1}^N x_n)\mu_N + \sum_{n=1}^N x_n^2 + \lambda_0\mu_0^2]$,

by first initializing λ_N to some arbitrary value and using that value along with the other parameters to obtain b_N. Later with the current value of b_N to compute λ_N. Repeat this procedure until convergence. This will let you obtain the hyperparameters of the approximation posterior.

Question 9: *Describe the exact posterior.*

The exact posterior distribution is derived by using Bayes' theorem.

$P(\mu, \tau|\, D) \propto P(D|\mu, \tau)P(\mu|\tau)P(\tau)$. The expression of the Gaussian- Gamma conjugate prior and likelihood distribution from p. 470 in Bishop [2] are as follows:

$$P(D|\mu, \tau) = \left(\frac{\tau}{2\pi}\right)^{N/2} e^{-\frac{\tau}{2}\sum_{n=1}^N (x_n-\mu)^2} \quad (1)$$

$$P(\mu|\tau) = N(\mu|\mu_0, (\lambda_0\tau))^{-1} = \left(\frac{1}{2\pi(\lambda_0\tau)^{-1}}\right)^{\frac{1}{2}} e^{-\frac{1}{2(\lambda_0\tau)^{-1}}(\mu-\mu_0)^2} \quad (2)$$

$$P(\tau) = Gamma(\tau|\, a_0, b_0) = \frac{b_0^{a_0}\tau^{a_0-1}e^{-b_0\tau}}{\Gamma(a_0)} \quad (3)$$

The calculation of the hyperparameter follows:

$$P(\mu, \tau|\, D) \propto P(D|\mu, \tau)P(\mu|\tau)P(\tau)$$
$$\propto \underbrace{\tau^{\frac{N}{2}+a_0-1}}_{(4)} \underbrace{e^{-\frac{\tau}{2}\sum_{n=1}^N x_n^2 - \frac{\lambda_0\mu_0^2\tau}{2} - \tau b_0}}_{(5)} \underbrace{\tau^{\frac{1}{2}} e^{\tau[\mu(\sum_{n=1}^N x_n + \lambda_0\mu_0) - \frac{1}{2}\mu^2(N+\lambda_0)]}}_{(6)}$$

Comparing (4) with (3) result with the parameter a: $a^* = \frac{N}{2} + a_0$.

(5) with (3) for parameter b : $b^* = \frac{1}{2}\sum_{n=1}^N x_n^2 + \frac{\lambda_0\mu_0^2}{2} + b_0$.

(6) $= \tau^{\frac{1}{2}} e^{\tau[\mu(\sum_{n=1}^N x_n + \lambda_0\mu_0) - \frac{1}{2}\mu^2(N+\lambda_0)]} = \tau^{\frac{1}{2}} e^{-\frac{\tau}{2}(N+\lambda_0)[\mu^2 - 2\mu\frac{\sum_{n=1}^N x_n + \lambda_0\mu_0}{N+\lambda_0}]} \propto$
$\tau^{\frac{1}{2}} e^{-\frac{\tau}{2}(N+\lambda_0)\left(\mu - \frac{(\sum_{n=1}^N x_n + \lambda_0\mu_0)}{N+\lambda_0}\right)^2}$, hence $\mu^* = \frac{\sum_{n=1}^N x_n + \lambda_0\mu_0}{N+\lambda_0}$ and $\lambda^* = N + \lambda_0$.

The resulting exact posterior is also a Gaussian – Gamma distribution:

$$P(\mu, \tau|\, D) \propto N(\mu|\mu^*, (\tau\lambda^*)^{-1})Gamma(\tau|a^*, b^*)$$

Question 10: *Compare the variational distribution with the exact posterior. Run the inference for a couple of interesting cases and describe the difference.*

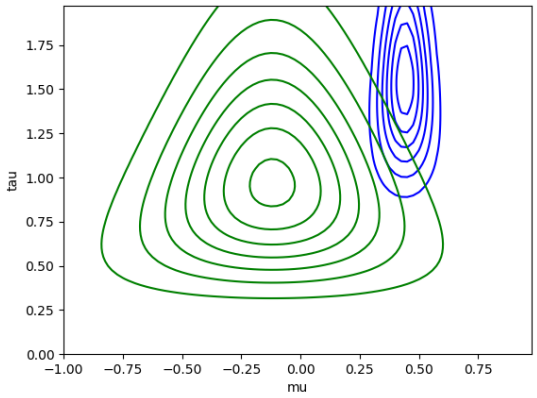

Fig 1. Mean and precision of variational inference, iterations: 2

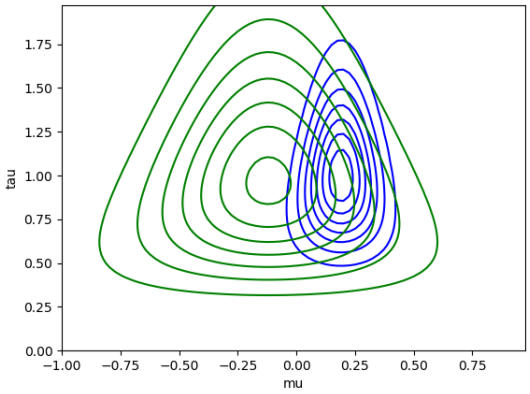

Fig 2. Mean and precision of variational inference, iterations: 5

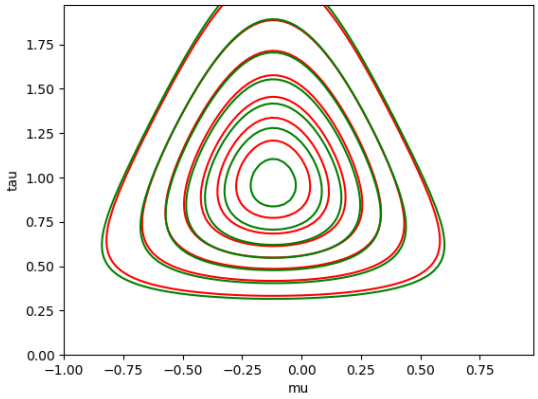

Fig 3. Mean and precision of variational inference, iterations: 7

The plots above represents variations of a univariate Gaussian distribution. The green contours stand for the exact posterior distribution regarding the derived hyperparameters mentioned in question 9. In fig 1 and fig 2, the blue contours show the re- estimated factors of the variational approximation to the posterior distribution. Especially in fig 2, one can see that the precision factor $q_\tau(\tau)$ of the approximation posterior starts to align corresponding to the tau-axis with the exact posterior. Lastly, fig 3 shows the convergence of the iterative procedure of the hyperparameters and result with the optimal factored approximation. The conclusion that can be drawn is that the factorized variational approximation of the posterior distribution $q(\mu, \tau) = q_\mu(\mu)q_\tau(\tau)$ can uphold similarities to the exact posterior when obtaining optimal hyperparameters mentioned earlier.

2.6 Variational Inference

Question 15: *Present the algorithm written down in a formal manner (using both text and mathematical notation, but not pseudo code).*

A Cartesian Matrix model is defined with rows r, $1 \le r \le R$ and columns c, $1 \le c \le C$. Each row has its distribution $N(\mu_r, \lambda_r^{-1})$ depending on each r, where the mean is $\mu_r \sim N(\mu, \lambda^{-1})$ and variance λ_r is known. Similar statement for each column which has the distribution $N(\xi_c, \tau_c^{-1})$ on each c and the mean is $\xi_c \sim N(\xi, \tau^{-1})$ also the variance τ_c^{-1} is known.

The matrix is generated such that $S_{rc} = X_r + Y_c$, where $X_r \sim N(\mu_r, \lambda_r^{-1})$ and $Y_c \sim N(\xi_c, \tau_c^{-1})$. By the assumption that X_r and Y_c are independent random variables from a normal distribution, hence the sum is also a normal distribution:

$$S_{rc} \sim N(\mu_r + \xi_c, \lambda_r^{-1} + \tau_c^{-1}).$$

For $S = [S_{11},,,S_{RC}]$, the posterior approximation can be written as:

$$P(\mu_1,,,\mu_R, \xi_1,,,\xi_C|S) = P(S|\mu_1,,,\mu_R, \xi_1,,,\xi_C)P(\mu_1,,,\mu_R)P(\xi_1,,,\xi_C)$$

Where,

$$P(S|\mu_1,,,\mu_R, \xi_1,,,\xi_C) = \prod_{r,c=1}^{R,C} N(S_{rc}|\, \mu_r + \xi_c, \lambda_r^{-1} + \tau_c^{-1}),$$

$P(\mu_1,,,\mu_R) = \prod_{r=1}^{R} N(\mu_r|\mu, \lambda^{-1})$ and $P(\xi_1,,,\xi_C) = \prod_{c=1}^{C} N(\xi_c|\xi, \tau^{-1})$.

The factorized approximation is, by assuming $q(\mu, \xi) = q(\mu)q(\xi)$ such that the posterior distribution factorizes into independent factors. By taking the logarithm, the optimal factor will then be:

$$\ln q^*(\mu) = \mathbb{E}_{\xi_c} \ln[P(\mu_1,,,\mu_R, \xi_1,,,\xi_C|S)] + const =$$

$$-\frac{1}{2}\mathbb{E}_{\xi_c}\left\{\sum_{r=1}^{R}\sum_{c=1}^{C}\frac{[S_{rc}-(\mu_r+\xi_c)]^2}{\lambda_r^{-1}+\tau_c^{-1}} + \lambda\sum_{r=1}^{R}(\mu_r-\mu)^2 + \tau\sum_{c=1}^{C}(\xi_c-\xi)^2\right\} + const.$$

to make it simple, doing for the first iteration in row one, $r = 1$,

$$\ln q^*(\mu_1) = -\frac{1}{2}\mathbb{E}_{\xi_c}\left\{\sum_{c=1}^{C}\frac{[S_{1c}-(\mu_1+\xi_c)]^2}{\lambda_1^{-1}+\tau_c^{-1}} + \lambda(\mu_1-\mu)^2 + \tau\sum_{c=1}^{C}(\xi_c-\xi)^2\right\} + const =$$

$$-\frac{1}{2}\mathbb{E}_{\xi_c}\left\{\sum_{c=1}^{C}\frac{-2S_{1c}\mu_1+\mu_1^2+2\mu_1\xi_c}{\lambda_1^{-1}+\tau_c^{-1}} + \lambda\mu_1^2 - 2\lambda\mu_1\mu + \lambda\mu^2 - \frac{S_{rc}^2+2S_{1c}\xi_c+\xi_c^2}{\lambda_1^{-1}+\tau_c^{-1}} + \tau\sum_{c=1}^{C}(\xi_c-\xi)^2\right\}$$
$+ const =$ {the fourth, fifth and sixth term are not dependent on μ_1, therefore are constant}=

$$-\frac{1}{2}\mu_1^2\left(\sum_{c=1}^{C}\frac{\lambda_1\tau_c}{\lambda_1+\tau_c} + \lambda\right) + \mu_1\frac{\left(\mathbb{E}_{\xi_c}\sum_{c=1}^{C}\frac{S_{1c}-\xi_c}{\lambda_1^{-1}+\tau_c^{-1}} + \lambda\mu\right)}{\left(\sum_{c=1}^{C}\frac{\lambda_1\tau_c}{\lambda_1+\tau_c} + \lambda\right)}\left(\sum_{c=1}^{C}\frac{\lambda_1\tau_c}{\lambda_1+\tau_c} + \lambda\right) + const =$$

$-\frac{1}{2}\left(\sum_{c=1}^{C}\frac{\lambda_1\tau_c}{\lambda_1+\tau_c}+\lambda\right)\left(\mu_1-\frac{\sum_{c=1}^{C}\frac{S_{1c}-\mathbb{E}[\xi_c]}{\lambda_1^{-1}+\tau_c^{-1}}+\lambda\mu}{\sum_{c=1}^{C}\frac{\lambda_1\tau_c}{\lambda_1+\tau_c}+\lambda}\right)^2+const$, hence the optimal factor is a Gaussian:

$q^*(\mu_1)\sim N(\mu_1|m_1,\gamma_1^{-1})$, where the mean is,

$$m_1=\frac{\sum_{c=1}^{C}\frac{S_{1c}-\mathbb{E}[\xi_c]}{\lambda_1^{-1}+\tau_c^{-1}}+\lambda\mu}{\sum_{c=1}^{C}\frac{\lambda_1\tau_c}{\lambda_1+\tau_c}+\lambda}$$

and the variance,

$$\gamma_1^{-1}=\left(\sum_{c=1}^{C}\frac{\lambda_1\tau_c}{\lambda_1+\tau_c}+\lambda\right)^{-1}.$$

And to go through each row you compute the iteration $i\in[1,R]$ for $q^*(\mu_i)$.

For the second factor follows similar mathematical procedure mentioned above, but going through each column j iterations $j\in[1,C]$.

$q^*(\xi_j)\sim N(\xi_j|\nu_j,\Lambda_j^{-1})$, where the mean is

$$\nu_j=\frac{\sum_{r=1}^{R}\frac{S_{rj}-\mathbb{E}[\mu_r]}{\lambda_r^{-1}+\tau_j^{-1}}+\xi\tau}{\sum_{r=1}^{R}\frac{\lambda_r\tau_j}{\lambda_r+\tau_j}+\tau}$$

and the variance,

$$\Lambda_j^{-1}=\left(\sum_{r=1}^{R}\frac{\lambda_r\tau_j}{\lambda_r+\tau_j}+\tau\right)^{-1}.$$

References:

[1] https://en.wikipedia.org/wiki/Variational_Bayesian_methods, last updated: 25/1-18

[2] Bishop, Cristopher M. Patter Recognition and Machine Learning, 2006-08-01

Appendix:

For question 8:

```
import numpy as np
from scipy.special import gamma
import math as mt

# Initial values
lambda_0, my_0, a_0, b_0 = 0, 0, 0, 0

# Data
data = np.random.normal(loc = 0.0, scale = 1.0, size = 100)
x = data[:,None]
numb = 100
my1 = np.linspace(-1, 1, numb)
my = my1[:,None]
tau1 = np.linspace(0,2, numb)
tau = tau1[:,None]

# Parametres
N = len(data)
x_sum = sum(data)
x_bar = x_sum*1/N
x_mean_sq = sum(data**2)
my_N = (lambda_0*my_0 + N*x_bar)/(lambda_0 + N)
a_N = a_0 + (N+1)/2

def trueParam(N, lambda_0, my_N, x_sum, x_mean_sq, my_0, a_N):
    minIt = 1e-6
    lambda_N = 1
    i = 1
    while(i < 1000):
        b_N = b_0 + 0.5 * ((lambda_0 + N) * (lambda_N ** -1 + my_N ** 2) -
2 * (lambda_0 * my_0 + x_sum) * my_N + x_mean_sq + lambda_0 * (my_0 ** 2))
        lambda_N = (lambda_0 + N)*a_N/b_N
        if lambda_N - lambda_0 < minIt:
            return(lambda_N, b_N)
        else:
            lambda_0 = lambda_N
            i = i + 1

def IVposterior(lambda_N, tau, my,my_N, b_N, a_N):
    q_my = np.sqrt((lambda_N) / (2 * np.pi)) * np.exp(np.dot((-1 *
(lambda_N) / 2), np.transpose((my - my_N) ** 2)))


    q_tau = (1 / mt.gamma(a_N)) * (b_N ** a_N) * np.exp(-1 * tau * b_N) *
(np.power(tau, -1 + a_N))

    IV = q_my * q_tau
    return (IV)

# Calling functions
lambda_true, b_true = trueParam(N, lambda_0, my_N, x_sum, x_mean_sq, my_0,
a_N)
```