**CS-E4820 Machine Learning: Advanced Probabilistic Methods**
Pekka Marttinen, Paul Blomstedt, Homayun Afrabandpey, Reza Ashrafi, Betül Güvenç, Tianyu Cui, Pedram Daee, Marko Järvenpää, Santosh Hiremath (Spring 2019)
Exercise problems, round 3, due on Tuesday, 12th February 2019, at 23:55
Please return your solutions in MyCourses as a single PDF file.

**Problem 1.** *"Poisson-Gamma."*

Suppose you have $N$ i.i.d. observations $\mathbf{x} = \{x_i\}_{i=1}^N$ from a Poisson$(\lambda)$ distribution with a rate parameter $\lambda$ that has a conjugate prior

$$\lambda \sim \text{Gamma}(a, b)$$

with the shape and rate hyperparameters $a$ and $b$. Derive the posterior distribution $\lambda|\mathbf{x}$.

**Problem 2.** *"Multivariate Gaussian."*

Suppose we have $N$ i.i.d. observations $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ from a multivariate Gaussian distribution

$$\mathbf{x}_i \mid \boldsymbol{\mu} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

with unknown mean parameter $\boldsymbol{\mu}$ and a known covariance matrix $\boldsymbol{\Sigma}$. As prior information on the mean parameter we have

$$\boldsymbol{\mu} \sim \mathcal{N}(\mathbf{m_0}, \mathbf{S_0}).$$

(a) Derive the posterior distribution $p(\boldsymbol{\mu}|\mathbf{X})$ of the mean parameter $\boldsymbol{\mu}$.

(b) Compare the Bayesian estimate (posterior mean) to the maximum likelihood estimate by generating $N = 10$ observations from the bivariate Gaussian

$$\mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right).$$

For this you can use the Python function *numpy.random.normal*[1], making use of the fact that the elements of the bivariate random vectors are independent. In the Bayesian case, use the prior with $\mathbf{m_0} = [0,0]^T$ and $\mathbf{S_0} = \left[\begin{smallmatrix}0.1 & 0\\0 & 0.1\end{smallmatrix}\right]$. Report both estimates. Is the Bayesian estimate closer to the true value $\boldsymbol{\mu} = [0,0]^T$?

**Problem 3.** *"Wishart distribution."*

Consider a multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1})$, parametrized using the precision matrix $\boldsymbol{\Lambda}$, which is simply the inverse of the covariance matrix $\boldsymbol{\Sigma}$. This parametrization is often convenient for conjugate Bayesian inference. The conjugate prior of the precision matrix of the above normal distribution is the Wishart distribution: $\boldsymbol{\Lambda} \sim \text{Wishart}(\mathbf{W}, \nu)$[2]. In Python, the function *scipy.stats.wishart.rvs* can be used to simulate samples from a Wishart distribution [3].

[1] https://docs.scipy.org/doc/numpy/reference/generated/numpy.random.normal.html.

[2] The scale matrix $\mathbf{W}$ is a $p \times p$ positive definite matrix and the degrees of freedom $\nu$ is a real value with $\nu > p - 1$.

[3] See documentation https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.wishart.html.

Suppose you wish to express as your prior belief that the precision matrix should be close to $\mathbf{A}$, given by

$$\mathbf{A} = \begin{bmatrix} 2 & 0.3 \\ 0.3 & 0.5 \end{bmatrix}.$$

(a) Familiarize yourself with the Wishart distribution, e.g., by reading from Wikipedia or any available book. What are the mean and variance of the Wishart distribution?

(b) Specify parameters of the Wishart distribution such that the expected value of the $\mathbf{\Lambda}$ matrix would be equal to $\mathbf{A}$. Use e.g. *scipy.stats.wishart.rvs(df, scale, size)* to simulate samples from the distribution you have specified, and confirm by averaging over the samples that the expectation indeed is equal to $\mathbf{A}$. Try 1, 10 and 1000 samples. (The average should converge to $\mathbf{A}$ as the number of samples increases.)

(c) How should one select the parameter values to make the matrices simulated from the Wishart distribution arbitrarily close to $\mathbf{A}$? Show a few examples of this by adjusting the parameters to get increasingly closer to $\mathbf{A}$.

Your solution should include code (with some comments) that accomplishes (b) and (c).