# CS689: Graphical Models - Fall 2018

## Final Exam

Dec 19, 2018 Name: Solutions

**Instructions:** Write your name on the exam sheet. The duration of this exam is **two hours**. No electronic devices may be used during the exam. You may consult your notes during the exam. Up to a two inch stack of notes is permitted. No other sources are permitted. Sharing of notes during the exam is strictly prohibited. Show your work and provide answers that are as detailed as possible. Attempt all problems. Partial credit may be given for incorrect or incomplete answers. If you need extra space for answers, write on the back of the preceding page. If you have questions at any time, raise your hand.

| Problem | Topic | Page | Points | Score |
|---|---|---|---|---|
| 1 | Optimization | 1 | 10 | |
| 2 | Maximum Likelihood | 2 | 10 | |
| 3 | Linear Regression | 3 | 10 | |
| 4 | Non-Linear Classification | 4 | 10 | |
| 5 | Mixture Models | 5 | 10 | |
| 6 | EM Algorithm | 6 | 10 | |
| 7 | Factor Analysis and Autoencoders | 7 | 10 | |
| 8 | Bayesian Inference | 8 | 10 | |
| 9 | Experiment Design | 9 | 10 | |
| 10 | Applications | 10 | 10 | |
| Total: | | | 100 | |

## 1. (*10 points*) Optimization

**(a)** (5 points) Consider the objective function $f(\mathbf{m}) = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{x}_n - \mathbf{m})^T\mathbf{A}(\mathbf{x}_n - \mathbf{m})$ where $\mathbf{m} \in \mathbb{R}^D$, $\mathbf{x}_n \in \mathbb{R}^D$, and $\mathbf{A}$ is a $D \times D$ strictly positive real diagonal matrix. Find the minimizer of $f(\mathbf{m})$ with respect to $\mathbf{m}$. Show your work.

$$\min_m f(m) = \min_m \frac{1}{2}\sum_{n=1}^{N}(x_n - m)^T A(x_n - m)$$

$$= \min_m \frac{1}{2}\sum_{n=1}^{N}\sum_{d=1}^{D} A_{dd}(x_{nd} - m_d)^2 \quad \text{... by } A \text{ is diagonal}$$

$$\frac{\partial}{\partial m_d} f(m) = \sum_{n=1}^{N} A_{dd}(x_{nd} - m_d) m_d = 0$$

$$\Rightarrow A_{dd} m_d \sum_n x_{nd} = A_{dd} \cdot m_d^2 \cdot N$$

$$\Rightarrow m_d = \frac{1}{N}\sum_{n=1}^{N} x_{nd} \Rightarrow m = \frac{1}{N}\sum_{n=1}^{N} x_n$$

**(b)** (5 points) Consider the objective function $f(\mathbf{x}) = \sum_{k=1}^{K} n_k \log(x_k)$ where $n_k$ are non-negative constants. Find the maximizer of this objective function subject to the constraints $x_k > 0$ for all $k$, and $\sum_{k=1} x_k = 1$. Show your work and explain your solution.

$$\max_x f(x) = \max_x \sum_{k=1}^{K} n_k \log x_k$$

$$\text{st } \sum_{k=1}^{K} x_k = 1 \qquad \text{st } \sum_{k=1}^{K} x_k = 1$$

Use Lagrange multiplier!

$$L(x, \lambda) = \sum_{k=1}^{K} n_k \log x_k - \lambda\left(\sum_{k=1}^{K} x_k - 1\right)$$

$$\frac{\partial L}{\partial x_k} = \frac{n_k}{x_k} - \lambda = 0 \Rightarrow \lambda x_k = n_k$$

$$\frac{\partial L}{\partial \lambda} = \sum_{k=1}^{K} x_k - 1 = 0 \Rightarrow \sum_k x_k = 1$$

$$\Rightarrow \sum_k \lambda x_k = \sum_k n_k$$

$$\lambda = \sum_k n_k$$

$$\text{So } \lambda x_k = n_k$$

$$\Rightarrow x_k = \frac{n_k}{\lambda} = \frac{n_k}{\sum_k n_k}$$

**2.** (*10 points*) **Maximum Likelihood:** Suppose we are interested in modeling the number of cars $y$ crossing a particular intersection as a function of a set of feature $\mathbf{x} \in \mathbb{R}^D$ including time of day, day of week, weather, etc. We decide to use a Poisson regression model as shown below where $g(\mathbf{x}, \mathbf{w}) = \exp(\mathbf{w}^T\mathbf{x})$.

$$P(Y = y|X = x, \mathbf{w}) = \frac{1}{y!}g(\mathbf{x}, \mathbf{w})^y \exp(-g(\mathbf{x}, \mathbf{w}))$$

(a) (5 points) What is the conditional log likelihood of this model given a data set $\mathcal{D} = \{(y_n, \mathbf{x}_n)|n = 1, ..., N\}$? Provide an expression that is as detailed as possible and explain your answer.

$$L(\mathcal{D}|w) = \sum_{n=1}^{N} \log P(Y = y_n | X = x_n, w)$$

$$= \sum_{n=1}^{N} \log \left( \frac{1}{y_n!} g(x_n, w)^{y_n} \exp\left(-g(x_n, w)\right) \right)$$

$$= \sum_{n=1}^{N} \left[ -\log(y_n!) + y_n \log\left(g(x_n, w)\right) - g(x_n, w) \right]$$

$$= \sum_{n=1}^{N} \left[ -\log(y_n!) + y_n \left(w^T x_n\right) - \exp\left(w^T x_n\right) \right]$$

(b) (5 points) Explain how you would find the optimal value of $\mathbf{w}$ using maximum conditional likelihood estimation. Provide supporting equations and/or sketch an algorithm.

* First, find the gradient of $L(\mathcal{D}|w)$ with respect to $w$.

* Next, either solve the gradient equation if possible, or supply the gradient to an optimizer.

* For a numerical solution, we could use basic gradient ascent:

$w \leftarrow w_0$
For $i = 1 : T$
  $w \leftarrow w + \alpha \nabla_w L(\mathcal{D}, w)$

**3.** (*10 points*) **Linear Regression:** Consider the linear regression model $y = \mathbf{w}^T\mathbf{x} + b$. Suppose we have access to code that fits the model by minimizing the objective

$$f(\mathbf{w}, b) = C \cdot \sum_{n=1}^{N} |y_n - (\mathbf{w}^T\mathbf{x}_n + b)| + \sum_{d=1}^{D} w_d^2$$

for a given constant $C$ and a data set $\mathcal{D} = \{(y_n, \mathbf{x}_n) | n = 1, ..., N\}$. What probabilistic model has a conditional log likelihood function corresponding to this objective function? Provide the conditional density $P(Y = y|\mathbf{X} = \mathbf{x})$ and the prior $P(\mathbf{W} = \mathbf{w})$ in terms of $\mathbf{w}$, $b$ and $C$. Show your work and explain your answer.

$$\min_{w,b} \quad C \cdot \sum_{n=1}^{N} |y_n - (w^T x_n + b)| + \sum_{d=1}^{D} w_d^2$$

$$= \max_{w,b} \quad - \sum_{n=1}^{N} |y_n - (w^T x_n + b)| - \frac{1}{C} \sum_{d=1}^{D} w_d^2$$

$$= \max_{w,b} \quad \prod_{n=1}^{N} \exp\left(-|y_n - (w^T x_n + b)|\right) \times \exp\left(-\frac{1}{2(C/2)} \|w\|_2^2\right)$$

We can see that $P(Y=y|X=x) \propto \exp\left(-|y_n - (w^T x_n + b)|\right)$, which is a Laplace or double exponential distribution.

We can see that $P(W=w) \propto \exp\left(-\frac{1}{2(C/2)} \|w\|_2^2\right) = \mathcal{N}\left(w; 0, \frac{C}{2} I\right)$, a normal distribution with mean 0 and covariance $\frac{C}{2} I$.

**4.** (*10 points*) **Non-Linear Classification:**

**(a)** (5 points) Explain two ways that a linear support vector classifier can be modified to produce non-linear decision boundaries. Provide equations to support your answer.

* We can use a basis expansion. Instead of the classification function being $w^T x + b \geq 0$, we use $\tilde{w}^T \varphi(x) + b \geq 0$ where $\varphi(x)$ is a function that maps from the initial feature space to an alternate space non-linearly.

* We can use a Kernel SVM. We supply a kernel function $K(x_i, x_j)$ that is non-linear. The SVM optimizes the dual using the kernel instead of $\langle \varphi(x_i), \varphi(x_j) \rangle$ explicitly.

**(b)** (5 points) Explain what the primary trade-offs are between using a non-linear support vector machine compared to a multi-layer neural network for learning a non-linear classifier.

① The SVM has a convex objective function, while the neural network has many local optima.

② The neural network can learn useful non-linear transformations of the data, while the SVM needs to be told what basis expansion or kernel to use.

**5.** (*10 points*) **Mixture Models** Suppose we have a data set $\mathcal{D}$ consisting of data vectors $\mathbf{x} = [\mathbf{x}_B, \mathbf{x}_R]$ where $\mathbf{x}_B$ is a block of $D_B$ binary variables and $\mathbf{x}_R$ is a block of $D_R$ real-valued variables.

**(a)** (5 points) Explain how you could model these data using a mixture model. Provide an expression for the joint distribution $P(\mathbf{X} = \mathbf{x}, Z = z)$ for your proposed model.

* For the block $x_B$, we can make all variables independant given $z$ with a Bernoulli distribution. $P(x_{Bi}=x \mid z=z) = \theta_{iz}^{x}(1-\theta_{iz})^{1-x}$

* For the block $x_R$, we can either make all variable independant or joint normal given $z$: $P(x_R = x \mid z=z) = \prod_{i=1}^{D_R} N(x_i \mid \mu_{iz}, \sigma_{iz}^2)$ ①

$$P(x_R = x \mid z=z) = N(x \mid \mu_z, \Sigma_z) \quad ②$$

* The joint model is then : $P(x_B = x_B \mid z=z) P(x_R = x_R \mid z=z) P(z=z)$

For $P(z=z)$, we can use $P(z=z) = \pi_z$, $\sum_z \pi_z = 1$, $\pi_z \geq 0, \forall z$.

**(b)** (5 points) Provide an expression for $P(Z = z|\mathbf{X} = \mathbf{x})$ for your proposed model. Show your work.

$$P(z=z \mid X=x) = \frac{P(z=z, X=x)}{P(X=x)}$$

using option ② for $P(x_R = x \mid z=z)$ we have :

$$= \frac{\left(\prod_{i=1}^{D_B} \theta_{iz}^{[x_{Bi}]} (1-\theta_{iz})^{(1-x_{Bi})}\right) \cdot N(x_R \mid \mu_z, \Sigma_z) \cdot \pi_z}{\sum_{z'=1}^{K} \left(\prod_{i=1}^{D_B} \theta_{iz'}^{x_{Bi}} (1-\theta_{iz'})^{1-x_{Bi}}\right) \cdot N(x_R \mid \mu_{z'}, \Sigma_{z'}) \cdot \pi_{z'}}$$

**6.** (*10 points*) **EM Algorithm** Suppose we have a data set $\mathcal{D} = \{x_n | n = 1, ..., N\}$ consisting of $N$ non-negative real numbers $x_n \in \mathbb{R}^{\geq 0}$ that we wish to model using a mixture of exponential distributions. The resulting Jensen's inequality lower bound on the log marginal likelihood is given below. Derive the M-Step update for the parameter $\lambda_k$. Show your work.

$$J(\mathcal{D}; \lambda, \phi_{1:N}) = \sum_{n=1}^{N} \sum_{k=1}^{K} \phi_{kn} (\log(\lambda_k) - \lambda_k x_n + \log \theta_k - \log \phi_{nk})$$

$$\frac{\partial J}{\partial \lambda_k} = \sum_{n=1}^{N} \left( \frac{\phi_{kn}}{\lambda_k} - \phi_{kn} x_n \right) = 0$$

$$\Rightarrow \quad \frac{1}{\lambda_k} \sum_{n=1}^{N} \phi_{kn} = \sum_{n=1}^{N} \phi_{kn} x_n$$

$$\Rightarrow \quad \lambda_k = \frac{\sum_{n=1}^{N} \phi_{kn}}{\sum_{n=1}^{N} \phi_{kn} x_n}$$

## 7. (*10 points*) Factor Analysis and Autoencoders

(a) (5 points) Factor analysis can be generalized by replacing the linear relationship between the mean of $\mathbf{x}$ and $z$ with a multi-layer neural network $g_{\mathbf{w}}(\mathbf{z})$ resulting in the likelihood $P(\mathbf{X} = \mathbf{x}|\mathbf{Z} = \mathbf{z}) = \mathcal{N}(\mathbf{x}; g_{\mathbf{w}}(\mathbf{z}), \Psi)$. Explain what the problem is with learning this model using maximum marginal likelihood estimation. Support your answer with equations as needed.

* The main problem is that the marginal likelihood can not be computed because it is not possible to integrate the latent variable z away analytically when $g_w(z)$ is a non-linear function. In other words, the integral below has no solution

$$\int \mathcal{N}(x;\, g_w(z),\, \Psi)\, \mathcal{N}(z;\, 0, I)\, dz = ?$$

(b) (5 points) Suppose we train a deep, non-linear autoencoder on images of digits. We then input noisy digit images into the autoencoder and find that the autoencoder produces outputs that exactly match the noisy inputs (i.e., the MSE between *noisy inputs* and reconstructions is 0). Explain what problem the learned network appears to have, why it might have this problem, and how we might be able to fix this problem.

* The learned network appears to be copying inputs to outputs, resulting in copying the noise.

* It may have this problem because it has too much capacity.

* We might be able to fix this problem by decreasing the capacity, or by training the model using a denoising objective.

**8.** (*10 points*) **Bayesian Inference:** Suppose that $x \in \mathbb{R}^{\geq 0}$ is an exponentially distributed random variable $P(X = x|\theta) = \theta \exp(-\theta x)$ where $\theta > 0$. The conjugate prior to $\theta$ is a Gamma distribution as defined below. Use this information to answer the following questions.

$$P(\theta) = \frac{\beta^\alpha}{\Gamma(\alpha)} \theta^{\alpha-1} \exp(-\beta \cdot \theta)$$

(a) (5 points) Given data $\mathcal{D} = \{x_n | n = 1, ..., N\}$, what is the likelihood of $\mathcal{D}$ given $\theta$? Explain your answer and simplify as much as possible.

* $L(D|\theta) = \prod_{n=1}^{N} P(x_n|\theta) = \prod_{n=1}^{N} \theta \exp(-\theta x_n) = \theta^N \exp\left(-\theta \sum_{n=1}^{N} x_n\right)$

* The likelihood function in Bayesian inference is a product over the data items of the likelihood $P(x_n|\theta)$.

(b) (5 points) Given data $\mathcal{D} = \{x_n | n = 1, ..., N\}$, what is the posterior distribution of $\theta$ given $\mathcal{D}$? Show your work and explain your answer.

* $P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)} \propto \left(\theta^N \exp\left(-\theta \sum_{n=1}^{N} x_n\right)\right)\left(\theta^{\alpha-1} \exp(-\beta \cdot \theta)\right)$

$$\propto \theta^{(\alpha+N)-1} \exp\left(-\left(\beta + \sum_{n=1}^{N} x_n\right)\theta\right)$$

$$\Rightarrow \text{Gamma}\left(\alpha + N, \beta + \sum_{n=1}^{N} x_n\right)$$

* The posterior is the probability of $\theta$ given D. We can define it using conditioning or Bayes Rule, but since the prior is conjugate, we can also use a short-cut method as shown above by identifying the posterior Gamma parameters.

**9.** (*10 points*) **Experiment Design:** We are attempting to select the latent dimension $k$ of a factor analysis model. Given a data set $\mathcal{D} = \{\mathbf{x}_n | n = 1, ..., N\}$, we train the model for different values of $k$ obtaining the maximum likelihood parameters $\theta_k^{MLE}$. We then evaluate the log likelihood of $\mathcal{D}$ using $\theta_k^{MLE}$, obtaining the value $L_k$. We then plot the values of $L_k$ and see that they are monotonically increasing with $k$, so we decide to increase the maximum value of $k$, and re-run the experiment. Use this information to answer the following questions.

**(a)** (5 points) What mistake have we made in this experimental design?

* We have used the same data set D to both find the MLE $\theta_k^{MLE}$ and to evaluate how well each model fit the same data D. This is equivalent to training on the test data.

**(b)** (5 points) Describe an alternative experimental design that will allow us to properly select the value of $k$.

* When we evaluate the learned models, we need to use a held out sample of data. We could use a train-validation split and pick the model with the best validation likelihood. We could also use cross-validation.

* Alternatively, we can use a penalized performance measure like AIC or BIC that penalizes the log likelihood based on the number of parameters in the model.

**10.** (*10 points*) **Applications:** Suppose we have a trained deep neural network classifier $f_{\mathbf{w}}(\mathbf{x})$ that maps complete data cases $\mathbf{x} \in \mathbb{R}^D$ to class probabilities. Assume there are $C$ classes. Unfortunately, some of our test data instances are incomplete (i.e., $\mathbf{x}_n = [\mathbf{x}_n^o, \mathbf{x}_n^m]$ where $\mathbf{x}_n^o$ are observed and $\mathbf{x}_n^m$ are missing). Propose an approach to use the existing neural network $f_{\mathbf{w}}(\mathbf{x})$ combined with a model of the features $P(\mathbf{X} = \mathbf{x})$ to make predictions for data cases with missing values. Explain the choices you make and provide pseudo code implementing your approach.

(1) given an incomplete data case $X = [x^o, x^m]$, first use $P(X = x)$ to compute $P(X^m = x^m \mid X^o = x^o)$

(2) Sample a set of values for the missing entries $x_m^s \sim P(X^m = x^m \mid X^o = x^o)$. If we can't sample from $P(X^m \mid X^o)$ directly, use an MCMC method or rejection sampling.

(3) For each sample, compute $P_s = f_w([x^o, x_s^m])$

(4) Return the average class probability vector:

$$\frac{1}{S}\sum_{s=1}^{S} P_s = \frac{1}{S}\sum_{s=1}^{S} f_w([x^o, x_s^m])$$