COGSCI 118B: Homework Assignment 2, due: Tues Oct 22 at the beginning of class Note this homework is 5 pages

1. (Matrix-vector calculus)
   Doing calculus with matrices and vectors is hugely useful for nearly everything we will do in this course. For reference, you may wish to consult one or more of the following sources, listed in order of increasing depth:

   - Bishop pages 697-8 (just a little bit of coverage).
   - Andrew Ng's Stanford CS 229 lecture notes 1, pages 8-9 (similar to the above):
     `http://see.stanford.edu/materials/aimlcs229/cs229-notes1.pdf`
   - Andrew Ng's Stanford CS 229 Linear Algebra Review and Reference, pages 15-20 (a bit more depth):
     `http://see.stanford.edu/materials/aimlcs229/cs229-linalg.pdf`
   - The Matrix Cookbook (extremely comprehensive listing):
     `www2.imm.dtu.dk/pubdb/views/edoc_download.php/3274/pdf/imm3274.pdf`

   The particular derivatives[1] that will be most immediately useful are, for matrix $\mathbf{S}$ and vectors $\mathbf{u}$ and $\mathbf{a}$, given by:

   - Equation (61) in the Matrix Cookbook:
   $$\frac{\partial \mathbf{a}^{\mathrm{T}}\mathbf{u}}{\partial \mathbf{u}} = \frac{\partial \mathbf{u}^{\mathrm{T}}\mathbf{a}}{\partial \mathbf{u}} = \mathbf{a}$$
   - Equation (73) in the Matrix Cookbook, the derivative of a quadratic form:
   $$\begin{aligned}\frac{\partial \mathbf{u}^{\mathrm{T}}\mathbf{S}\mathbf{u}}{\partial \mathbf{u}} &= (\mathbf{S}+\mathbf{S}^{\mathrm{T}})\mathbf{u} \qquad \text{(in general)}\\ &= 2\mathbf{S}\mathbf{u} \qquad \text{(if } \mathbf{S} \text{ is symmetric)}\end{aligned}$$

   [1 point] Using one or both of the above rules, for $J(\mathbf{u}) = \mathbf{u}^{\mathrm{T}}\mathbf{S}\mathbf{u} + \lambda(1-\mathbf{u}^{\mathrm{T}}\mathbf{u})$ determine the value of the derivative $\frac{\partial J(\mathbf{u})}{\partial \mathbf{u}}$.

2. [**2 points**] (Practice with matrix-vector calculus)
   Redo maximum likelihood estimation for the mean of a Gaussian random variable for the multidimensional case. This material is found on pages 93-94 of Bishop.
   Assume that you have a data set of observations $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N)}\}$, representing N observations of the vector variable $\mathbf{x} \in \mathbb{R}^D$. Each vector is normally distributed, so for each we have
   $$\begin{aligned}p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})\\ &= \frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}\end{aligned}$$
   We further assume that the data points are i.i.d. With these assumptions, we can write the probability of the entire data set, given $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, in the form
   $$\begin{aligned}p\left(\left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N)}\right\}|\boldsymbol{\mu}, \boldsymbol{\Sigma}\right) &= \prod_{n=1}^{N}\mathcal{N}(\mathbf{x}^{(n)}|\boldsymbol{\mu}, \boldsymbol{\Sigma})\\ &= \prod_{n=1}^{N}\frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}^{(n)}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}^{(n)}-\boldsymbol{\mu})\right\}\\ &= \frac{1}{(2\pi)^{ND/2}}\frac{1}{|\boldsymbol{\Sigma}|^{N/2}}\exp\left\{-\frac{1}{2}\sum_{n=1}^{N}(\mathbf{x}^{(n)}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}^{(n)}-\boldsymbol{\mu})\right\}\end{aligned}$$

[1] I am here using the term matrix derivative to make the parallel to the non-matrix case more apparent. As is seen in the Linear Algebra Review and Reference, the correct term is the *gradient.*

Focus on the ln of the above probability[2]. Expand the terms in the expression $(\mathbf{x}^{(n)}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}^{(n)}-\boldsymbol{\mu})$. Use the matrix-vector calculus material from the previous problem to take the derivative with respect to the mean vector $\boldsymbol{\mu}$. Set this derivative to zero to obtain the solution for the maximum likelihood estimate of the mean given by

$$\boldsymbol{\mu}_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}^{(n)}$$

3. **[8 points total]** (Matlab exercise: Bayesian sequential estimation and the effect of priors.) In this problem we will consider the problem of sequential Bayesian estimation of a coin's bias-weighting for heads. Beginning with a Beta[3] prior distribution of the bias μ, you will simulate observing successive coin flips, and plot the posterior distribution of the bias μ as determined by applying Bayes' Theorem[4]. This material is covered in Bishop, pages 71-74.

   To explore the effect that different initial prior distributions have, you will perform this sequential estimation process for three initial priors in parallel:$\{(a = 1, b = 1), (a = .5, b = .5), (a = 50, b = 50)\}$. You will simulate flipping a coin with a $\frac{1}{4}$ probability of coming up heads on each toss. Your end result will consist of two collections of subplots. The first collection of subplots will deal with the behavior of the three distributions of the bias parameter μ for the first few sample flips. The second collection of subplots will deal with the behavior of the three distributions of the bias parameter μ as we see several thousand flips.

   In each case you will begin by initializing the priors as specified above, along with a tally vector that records the heads, tails thus far observed. Using the included matlab m file **plotbetapdfs**, you will subplot these starting distributions. You will then begin a cycle:

   (a) Simulate a coin flip with bias $\mu = .25$ using the command **mnrnd**.
   (b) Update the head,tail tally vector with the flip result.
   (c) Update the prior parameter values with the flip result, yielding posteriors.
   (d) Plot these posteriors if the appropriate number of flips have been made (as specified below), and then begin the cycle again with these updated parameter values.

   [2 points] For the first collection of subplots, you will perform a total of 5 flips. You will have a 3-by-2 matrix of subplots, one plot each for the prior and for the state after each flip. The end result should resemble Figure 1.

   [2 points] For the second collection of subplots, you will perform a total of 2048 flips. You will have a 4-by-3 matrix of subplots. The first subplot will be of the prior state. Subsequent subplots will be made for numbers of flips that are the following powers of 2: $2^i, i = 1, \ldots, 11$. Your final results should resemble Figure 2.

   [2 points] What do the plots of posterior parameter distributions for the first 5 flips tell you about the difference between the Beta$(a = 50, b = 50)$ prior and the other two priors? When you are comparing the Beta$(a = 50, b = 50)$ case to the Beta$(a = 1, b = 1)$ case, couch your answer in terms of the interpretation of the prior parameters as "fake" data as per Bishop page 72.

   [2 points] What do the plots of the posterior parameter distributions after several thousand flips suggest about the behavior of Bayesian estimation in the limit of large numbers of observations? Your answer should touch on the commonalities in the behavior of the three different priors in the large data limit, relative to the "true" parameter value.

   Print out and hand in the plots specified above. Put all of the provided code and data files together with the m files that you have written into a folder. Zip up this folder and email the resulting zip file

[2]Recall that this expression, when viewed as a function of the parameter $\boldsymbol{\mu}$, is also referred to as the *log-likelihood*.
[3]Bishop equation (2.13)
[4]Bishop equation (1.43)

Figure 1: Short term behavior.

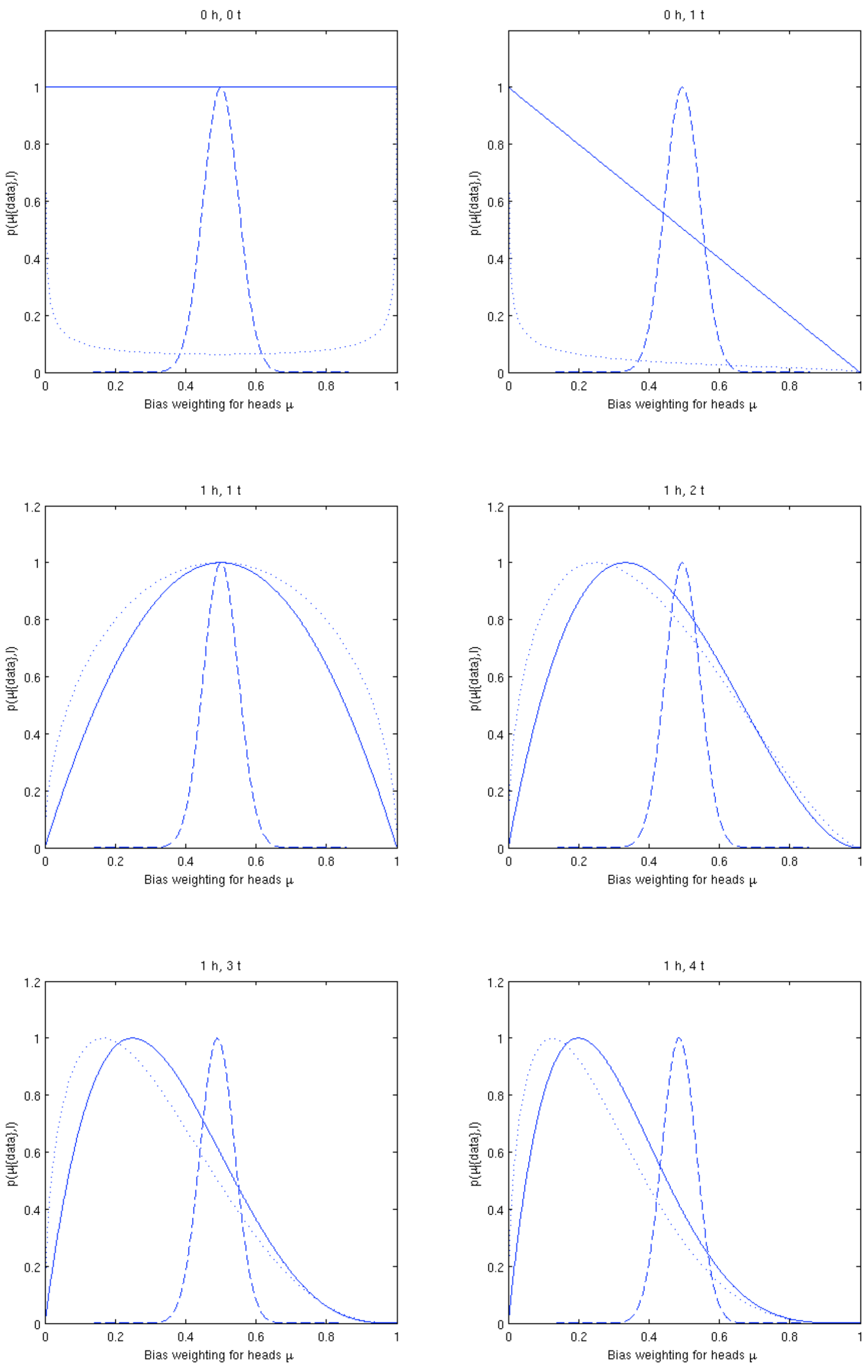
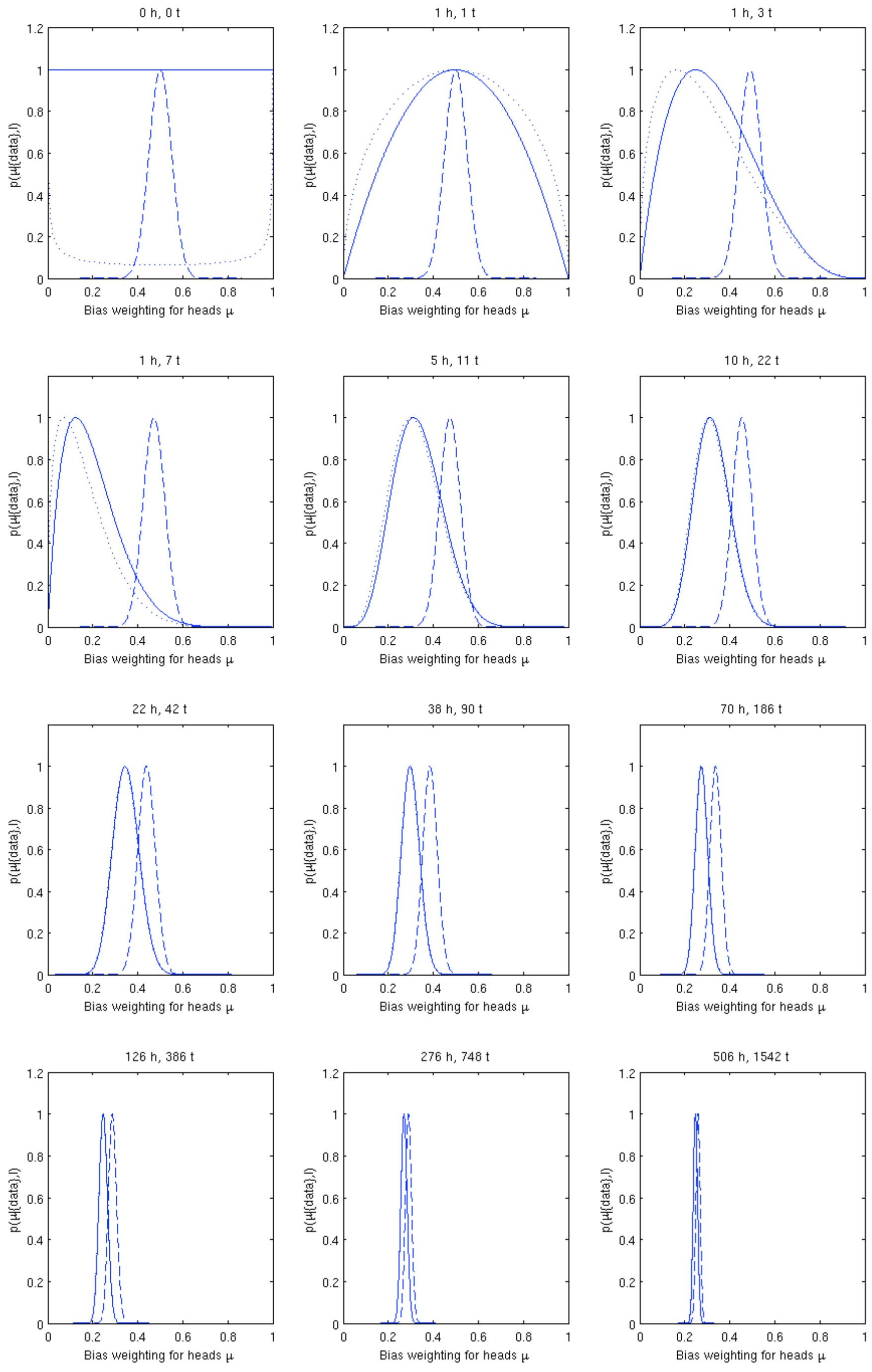


Figure 2: Long term behavior.



4. **[9 points]** (Matlab exercise: Implement K-means)
In this problem you will implement the K-means algorithm as covered in Bishop pages 424-425. You are given a main Matlab function file **runKMeans.m**, a plotting helper function **plotCurrent.m**, and the data files **faithful.txt** and **scaledfaithful.txt**. Your task is to provide code for the functions called by **runKMeans.m**:

   - **calcSqDistances.m** (Possibly helpful Matlab commands: **size, sum, repmat, .***)
   - **determineRnk.m** (Possibly helpful Matlab commands: **size, min, repmat, ==**)
   - **recalcMus.m** (Possibly helpful Matlab commands: **size, sum, repmat, .***)

   Note: If you find that you have written much more than 10 lines for any of the above functions, you should try to rethink your approach.

   Print out and hand in figures displaying the clustering results for $K = 2, 3, 4$ using the **scaledfaithful.txt** data file. Put all of the provided code and data files together with the m files that you have written into a folder. Zip up this folder and email the resulting zip file to the TA.

   Bonus:[3 points]
   One of the quirks of Matlab is that **for** loops are very slow when compared to "vectorized" versions performing the same calculations with matrices and vectors. To encourage you to learn how to vectorize your Matlab code, you can get up to 1 bonus point for each of the 3 functions, provided that you have implemented a correct vectorized version of the function.

(5) (6 points) **you should not integrate to answer these questions. Use the properties of the normal density function and your knowledge of Expected values.**

What is
$$\int_{-\infty}^{\infty} x e^{\frac{-(x-3)^2}{2}} dx =$$

What is
$$\int_{-\infty}^{\infty} (x-3)^2 e^{\frac{-(x-3)^2}{2}} dx =$$

What is
$$\int_{-\infty}^{\infty} x^2 e^{\frac{-(x-3)^2}{2}} dx =$$

(6) (6 points) Consider the scalar random variable X with mean m and variance σ^2. Consider constructing the following new random variables from X (a and b are constant numbers): $P = X + a$, $Q = bX$. Use the definition of mean and variance to derive expressions for the mean and variance of P and Q expressed as functions of m and σ^2.