The problem sets in this class will be done using Matlab. You may need to use the computers in the EECS labs to get access to Matlab. If you have problems accessing these computers, contact the GSIs at `gsi-cogsci131@lists.berkeley.edu`.

Collaboration Policy

Class policy is that collaboration is permitted when solving problems, but each student must: (1) write his or her own code; (2) write the document containing his or her answers independently; and (3) list the people he or she worked with on his or her problem set.

Submission Instructions

Problems will require you to either turn in Matlab code, written answers, or both. All of the files you need are provided for you – you just need to fill them in with your answers. **Make sure to fill in your name and collaborators for each part of the homework.**

All parts that include Matlab code come with a validation script called `validatePN.m` which you can run by calling `validatePN()` from Matlab (replace `N` with the problem number, e.g. for Problem 1, run `validateP1()`). Make sure you run these functions for each part of the homework before submitting your assignment. **If your code does not pass the validation script, you will NOT receive credit.** Note, however, that this is not a grading script; it is entirely possible for incorrect answers to pass the validation script.

You will be submitting your homework on bSpace. For each problem set, you will submit a single zip file containing your filled-in files from each problem. The directory structure within your submission should be the same as when the problem set was released. The zip file should be named:

```
YourLastName_YourFirstName_ProblemSetNumber.zip
```

For example, if your name is Jane Doe, your submitted file should be named `Doe_Jane_N.zip`, where `N` is the problem set number. You can update your submission until the due date. If you make any changes to your submission after the due date, we will regard your problem set as having been turned in at the last time it was changed.

If you use Matlab on the computers in Cory 105 or log in with remote desktop, please note that files saved to the Desktop are removed when you log out. Instead, save any files to your user directory, `U:/`. The `U:/` drive can also be accessed through a shortcut on the Desktop.

Late Problem Set Policy

This problem set is due on **April 17th, 2014 at 2:00pm**. You get **THREE LATE DAYS** total for the entire semester, to account for foreseeable minor crises. If you turn in a problem set late after using these late days, 20% of the total points on the problem set will be deducted for each 24 hours (or portion thereof) that it is late.

1 Bayes' Rule (5 points)

Suppose you pull a coin from your pocket and want to know whether it is fair or biased. Let θ denote the probability that the coin produces heads (`H`) each time you flip it, and assume that successive flips are independent. You have two hypotheses:

- h_0: the coin is fair, with $\theta = 0.5$
- h_1: the coin is biased, with $\theta = 0.95$.

A priori, you think your coin is more likely to be fair, although you know that a biased coin is a distinct possibility. To reflect this intuition, you choose priors $P(h_0) = 0.6$ and $P(h_1) = 0.4$.

Recall from class that you can use Bayes' rule to calculate the probability of a particular hypothesis being true given some data. Bayes' rule states that

$$P(h|d) = \frac{P(d|h)P(h)}{\sum_{h'} P(d|h')P(h')} \tag{1}$$

where the term $P(h|d)$ is the 'posterior' probability of a particular hypothesis h being true given some data, d, $P(d|h)$ is the 'likelihood' of seeing a d if h is true, and $P(h)$ is the 'prior' probability that a particular hypothesis h is true *before* observing the data d.

Part A

Imagine you flip your coin once and get a heads. Let d denote this data (i.e., the outcome of the coin flip). What is the probability of your data under h_0 and h_1 (i.e., what is $P(d|h_0)$ and $P(d|h_1)$)? Fill in the appropriate answers for the likelihood variables `pd_h0` and `pd_h1` in `writtenAnswersPartA.txt`. Use Bayes' rule to compute the posterior probability for h_0 and h_1 and save it in the variable `ph1_d`.

Hint: If you are stuck, take a look at your notes on Bernoulli variables. How do we calculate likelihoods for variables of this type?

Part B

Imagine you flipped your coin N times and produced a sequence of N_H heads and N_T tails (so that $N = N_H + N_T$). Complete the function template `probBiasedCoin.m`, so that it takes a $1 \times n$ binary vector of coin flips ($0 =$ `T`, $1 =$ `H`) as input and returns `postProbH1`, the posterior probability of the sequence under h_1. Ensure that your function works for any sequence of heads and tails. Example output:

```
>> probBiasedCoin([0 1 1 1 0 1])
ans =
    0.0799
```

```
>> probBiasedCoin([0 0])
ans =
    0.0066
```

Part C

Using the output from `probBiasedCoin`, complete the function `plotHeads` to graph the change in posterior probability for sequences of length $1, 2, \ldots,$ `numHeads` consisting solely of `H`'s. Run your function for `numHeads = 10` and save the resulting plot as as a JPEG named `postPlot.jpg` using the File → Save As menu in the figure window.

Part D

Using the plot from part (c), describe what happens to the posterior probability as the length of the sequence increases. Why should this be the case? Write your answers in the space provided in `writtenAnswersPartD.txt`.

2 Bias and Variance (5 points)

We wish to evaluate the performance of a learning algorithm that takes a set of (x, y) pairs as input and selects a function $g(x)$, which predicts the value of y for a given x (that is, for a given (x, y) pair, $g(x)$ should approximate y).

We will evaluate the 'fit' of the functions that our learning algorithm selects using the mean squared error (*MSE*) between $g(x)$ and y. For a set of n data points $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, the *MSE* associated with a function g is calculated as

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - g(x_i))^2 . \tag{2}$$

The first set of candidate functions that we will allow our algorithm to consider is the set of kth-order polynomials. This means that our hypothesis space will contain all functions of the form $g(x) = p_k x^k + p_{k-1} x^{k-1} + \ldots + p_1 x + p_0$. These functions are entirely characterized by their coefficient vector $\mathbf{p} = (p_k, p_{k-1}, \ldots, p_0)$, and become much more flexible as k increases (think about the difference between linear ($k = 1$), quadratic ($k = 2$), and cubic ($k = 3$) functions). If we are given a set of (x, y) pairs, it is straightforward to find the kth-order polynomial that minimizes the *MSE* between $g(x)$ and the observed y values. [1]

Load the file `xyData.mat` from `ps05/problem2/xyData.mat`. This file contains two matrices:

- `traindata`: A 10×2 coordinate matrix. The first column contains 10 x values between 0 and 1. The second column contains the approximate values of a function $y = f(x)$ at each x coordinate. The numbers in this column were generated by calculating $f(x)$ and then adding Gaussian (i.e., normally distributed) noise to each coordinate.
- `testdata`: A matrix similar to `traindata` except that it contains 100 (instead of 10) locations at which f is evaluated.

You will use this data to train and evaluate your learning algorithm.

A "learning algorithm" which finds a polynomial of given degree that minimizes the *MSE* on a set of (x, y) coordinates is implemented in Matlab with the `polyfit` command. You can use `p = polyfit(x,y,k)` to return the coefficient vector $\mathbf{p}$ for the kth-order polynomial g that best fits (i.e., has the smallest MSE on) the x and y coordinates in $\mathbf{x}$ and $\mathbf{y}$.

You can calculate the values of g at a set of x coordinates using the command `polyval`. This useful when making a plot. For example, if you wanted to plot the polynomial function g parameterized by coefficient vector $\mathbf{p}$ on the interval $[0, 1]$, you could write something like:

[1] For those who have done some statistics, the calculation for finding $\mathbf{p}$ is just a case of linear regression where the various powers of x are the predictors

```
plotx = 0:0.01:1;
ploty = polyval(p,plotx);
plot(plotx,ploty)
```

where the `0:0.01:1` on the first line gives the set of numbers between 0 and 1 which are divisible by 0.01. `polyval(p,plotx)` returns the values of $g(x)$ at each of these input coordinates.

Part A

Run the script `makePolynomialFitAndGraph.m`, which finds the best-fitting polynomials for $k = 0, \ldots, 8$, using the coordinates in `traindata` as its input. You should read the source code to see how this is accomplished.

Examine the figure `makePolynomialFitAndGraph` produces. Which polynomial do you think best captures the data? Why? Write your answer in the space provided in `writtenAnswersPartA.txt`.

Part B

Complete the function `findMSE.m` to compute the *MSE* for the polynomials identified above with $k = 0, \ldots, 8$, on both the training set (`traindata`) and the test set (`testdata`). This will entail using the `polyval` and `polyfit` functions in a manner similar to the way they were used in `makePolynomialFitAndGraph`.

The completed `findMSEAndGraph` function should return `MSE`, a 9×2 matrix containing the *MSE* values for a $k = 0, 1, \ldots, 8$th order polynomial (rows) on `traindata` (column 1) or `testdata` (column 2). For example, cell (5,1) would contain the MSE for a 4th-order polynomial on the points in `traindata`.

Next, complete the function template `plotMSE` to plot *MSE* versus k for both `traindata` and `testdata`. After completing the function, **save your plot** as as a JPEG named `msePlot.jpg` using the File $\rightarrow$ Save As menu in the figure. Be sure to include a proper legend, title, and axis labels! To find out how to set these programmatically, try `help legend`, `help title`, `help xlabel`, and/or `help ylabel`.

Part C

Look at the plot you generated in part (b). What happens to the *MSE* on the training set as the degree of the polynomial (i.e., the size of our hypothesis space) increases? What about on the test set? Should you always use a learning algorithm that gives the best results on the training set? Write your answers (with justification) in `writtenAnswersPartC.txt`.

Part D

The *MSE* on the test set is an approximation of the generalization error (*GE*) associated with a particular learning algorithm. In class we discussed how expected generalization error (*EGE*) is

composed of a bias and a variance term. Both the bias and the variance depend upon the true function, f, that we are estimating with our learning algorithm.

- An algorithm with high *bias* will systematically produce predictions that differ from the true function. This can happen in situations where the true function is more complex than the most complex function available to the algorithm.
- An algorithm with high *variance* will produce predictions that can vary wildly depending on the specifics of the dataset we are evaluating. This can happen in situations where the algorithm is able to return functions that are more complex than the true function.

In the space provided in `writtenAnswersPartD.txt` explain how the results from the *MSE* on `testset` could be explained in terms of the bias and variance of the learning algorithms involved.

3 The Coin of Infinite Hypotheses (4 points)

One day, you find yourself standing in a musty room in the back of an old magic shop. Heavy velvet curtains cover the dirty windows, reluctantly letting in a few narrow beams of light, which succeed only in illuminating layers of dust suspended in the stale air. You glance around cautiously, trying not to invite scrutiny from the watchful shopkeep. Your eyes dance over row upon row of leather-bound books, resting on the rich mahogany shelves lining each wall. Who knows how long they've been here, or what ancient secrets they conceal. They almost call out to you to be opened, to divulge their long-forgotten knowledge. But no... you've come here for another reason.

Before you can even turn around, the shopkeep anticipates your question. "You've come for a coin, haven't you?" His voice sounds strangely distant, and reminds you of some kind of large, creepy bird or something. You nod, swallowing. He totters past you, one of his legs struggling to keep up with the other. Reaching a black armoire in the corner of the room — how did you not notice it before? — he stops. The shopkeep slowly opens one of the drawers, revealing a beautiful, glimmering coin. It looks perfectly untarnished, yet, somehow, emanates the energy of centuries past.

Your father's deep, soulful voice echoes in your head. *"Fetch me a coin of Azeroth, child, but only if its probability of landing heads is between 0.55 and 0.75."* Is the coin in the armoire such a coin? How can we determine if this coin matches father's request?

We can formalize your predicament by letting the variable θ denote a coin of Azeroth's probability of landing heads on each toss, which is what you need to infer. We assume that each toss is independent of the others. In this learning problem, your hypothesis space is the set of all possible values of θ, which is all the real numbers from 0 to 1. We need to compare an infinite number of hypotheses! How can we do this?

Data: In the following questions, assume you tossed a coin of Azeroth 10 times and observed the specific sequence `HHTHTHHHHH`.

Prior: Assume we've observed some fictitious trials and have a prior belief that coins of Azeroth are slightly biased towards landing heads. Thus, let $V_H = 55$ and $V_T = 45$, denoting that in the past we've observed the coin of Azeroth flipped 100 times and saw 55 heads and 45 tails.

In parts A-C you will be completing the function `probAzeroth.m`. This function takes a single argument, `tossSequence`, a $1 \times n$ binary row vector representing an observed sequence of coin tosses where `1`'s represent `Heads` and `0`'s represent `Tails`. For parts A-C you should use `tossSequence = [1 1 0 1 0 1 1 1 1 1]`, corresponding to our observed sequence of `HHTHTHHHHH`.

Part A

Using the standard frequentist method of *maximum likelihood estimation*, find an estimate for the value of θ. Store this value in a variable named `prob3a` in the provided function `probAzeroth.m`.

Remember that *MLE* only relies on data that we've actually observed, so the prior should NOT factor into your answer.

Part B

Assuming a Bernoulli likelihood and Beta prior, compute the *maximum a posteriori* (MAP) estimate for θ. Store this value in a variable named `prob3b` in the provided function `probAzeroth.m`.

Part C

Assuming a Bernoulli likelihood and Beta prior, compute the posterior mean estimate for θ. Store this value in a variable named `prob3c` in the provided function `probAzeroth.m`.

Hint: The Beta and Bernoulli distributions are discussed in the lecture slides.

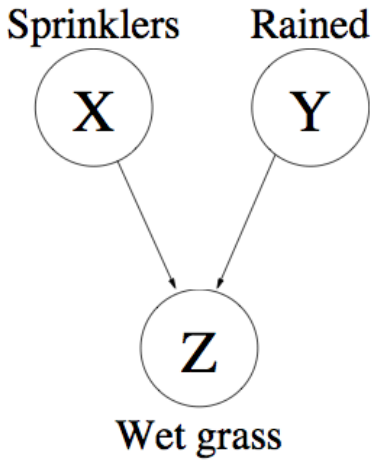
Part D

Given the father's request for a coin with probability of landing heads between 0.55 and 0.75 - do you think you should you take this coin? Why or why not? Please write your response in the provided `writtenAnswersPartD.txt`.

4 Graphical Models (6 points)

Bayesian networks provide an efficient way of representing a probability distribution. This made them very popular in artificial intelligence research, because they make it easy to handle uncertainty, as well as inferences both from cause to effect (making predictions) and effect to cause (forming explanations). These factors traditionally presented a challenge for AI approaches based on rules and symbols, such as production systems.

The figure below shows a simple graphical model relating three variables. X indicates whether the sprinklers were on last night, Y indicates whether it rained last night, and Z indicates whether the grass is wet. All three variables take on two values, with the value 1 being yes, and 0 being no.



The conditional probability distributions associated with this graphical model are as follows. The probability that $X = 1$ is 0.6. The probability that $Y = 1$ is 0.2. The probability of Z given X and Y is given in this table:

x	y	$P(Z = 1\|x, y)$
0	0	0.05
0	1	1.0
1	0	1.0
1	1	1.0

Thus, sprinklers being on are more probable than rain, and either the sprinklers being on or the presence of rain guarantees that the grass will be wet. There is also a small chance the grass could be wet as a result of some other cause.

Part A

Write down the factorization of the joint probability distribution on X, Y, and Z into conditional distributions that is specified by this Bayesian network. Use this factorization to compute the probability of each of the eight possible sets of values for the three variables. That is, fill in the numbers in the table below:

x	y	z	$P(x, y, z)$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Store the joint probability in a column vector named `jointProbTable` in the provided function `graphicalModel.m`. Remember that this vector should sum to 1, since this is a probability distribution.

Part B

Imagine that you observed the grass is wet in the morning ($Z = 1$). What happens to your beliefs about the sprinklers being on?

1. Use this joint distribution to work out the conditional distribution over X and Y when $Z = 1$, $P(X, Y|Z = 1)$. That is, fill in the numbers in the table below:

x	y	$P(x, y\|Z = 1)$
0	0	
0	1	
1	0	
1	1	

Store this conditional probability in a column vector named `condTable` in the provided function `graphicalModel.m`. Remember that this vector should sum to 1, since this is a probability distribution.

2. Now, sum over the values of Y to calculate the probability that the sprinklers were on given that the grass is wet, $P(X = 1|Z = 1)$. **Store this conditional probability in a variable** named `condXgivenZ` in the provided function `graphicalModel.m`.

3. How does $P(X = 1|Z = 1)$ compare to $P(X = 1)$? In the appropriate spot in `writtenAnswersPartB.txt`, give an interpretation for why these numbers are the same or different.

Part C

Imagine you got into your car, and heard on the radio that it rained last night. How does this affect your beliefs about the sprinklers being on?

1. Use the joint distribution to compute $P(X = 1|Y = 1, Z = 1)$. **Store this conditional probability in a variable** named `condXgivenYandZ` in the provided function `graphicalModel.m`.

2. How does $P(X = 1|Y = 1, Z = 1)$ compare to $P(X = 1|Z = 1)$? In the appropriate spot in `writtenAnswersPartC.txt`, give an interpretation for why these numbers are the same or different.

Part D

In the appropriate spot in `writtenAnswersPartD.txt`, summarize how the two pieces of evidence (wet grass, and hearing that it rained) affected beliefs about the sprinklers being on. This phenomenon is known as "explaining away." Compare the pattern of inferences produced by the Bayesian network with those that might result from using a production system of the kind explored in Problem Set 1. You don't need to focus on a specific production system or work out its predictions – just highlight the aspects of this inference that might be hard to capture using a production system.