EECS 16A Designing Information Devices and Systems I
Spring 2017 Babak Ayazifar, Vladimir Stojanovic Homework 13

This homework is due Thursday, May 4, 2017, at 23:59.
Self-grades are due Sunday, May 7, 2017, at 23:59.

Submission Format
Your homework submission should consist of **two** files.

- `hw13.pdf`: A single pdf file that contains all your answers (any handwritten answers should be scanned) as well as your IPython notebook saved as a pdf.

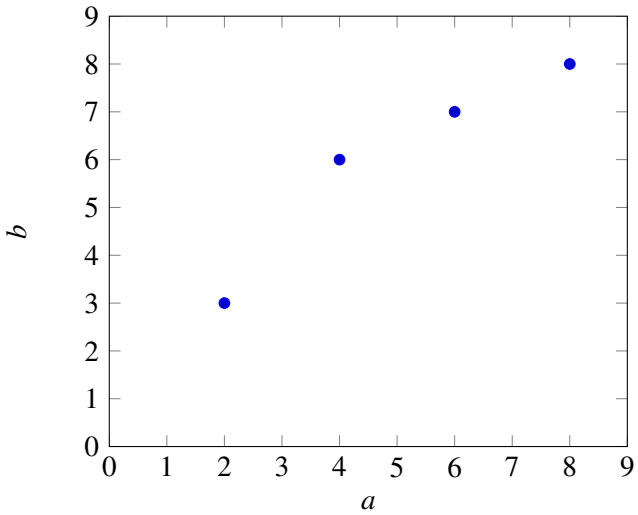
  If you do not attach a pdf of your IPython notebook, you will not receive credit for problems that involve coding. Make sure your results and plots are showing.

- `hw13.ipynb`: A single IPython notebook with all your code in it.

  In order to receive credit for your IPython notebook, you must submit both a "printout" and the code itself.

Submit each file to its respective assignment in Gradescope.

1. Mechanical: Linear Least Squares



a	2	4	6	8
b	3	6	7	8

(a) Consider the above data points. Find the linear model of the form

$$b = xa$$

that best fits the data, i.e. find the value of x that minimizes

$$\left\| \begin{bmatrix} b_1 \\ \vdots \\ b_4 \end{bmatrix} - \begin{bmatrix} a_1 \\ \vdots \\ a_4 \end{bmatrix} x \right\|^2 \tag{1}$$

Do not use IPython for this calculation and show your work. (A calculator is okay). Once you've computed x, compute the squared error between your model's prediction and the actual b values as shown in Equation (1). Plot the best fit line along with the data points to examine the quality of the fit. (If you're plotting by hand, it is okay if your plot of $b = xa$ is approximate.)

(b) You will notice from your graph that you can get a better fit by adding a b-intercept. That is we can get a better fit for the data by assuming a linear model of the form

$$b = x_1 a + x_2$$

In order to do this, we need to augment our $\mathbf{A}$ matrix for the least squares calculation with a column of 1's (do you see why?) so that it has the form

$$\mathbf{A} = \begin{bmatrix} a_1 & 1 \\ \vdots & \vdots \\ a_4 & 1 \end{bmatrix}$$

Find x_1 and x_2 that minimize

$$\left\| \begin{bmatrix} b_1 \\ \vdots \\ b_4 \end{bmatrix} - \begin{bmatrix} a_1 & 1 \\ \vdots & \vdots \\ a_4 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right\|^2 \tag{2}$$

Again, do not use IPython for this calculation and show your work. A calculator is okay, but take the inverse by hand using the formula for a 2×2 inverse.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

Compute the squared error between your model's prediction and the actual b values as shown in Equation (2). Plot your new linear model. Is it a better fit for the data?

(c) Let $\vec{\hat{x}}$ be the solution to a general linear least squares problem,

$$\vec{\hat{x}} = \underset{\vec{x}}{\operatorname{argmin}} \left\| \vec{b} - \mathbf{A}\vec{x} \right\|^2$$

Show that the error vector $\vec{b} - \mathbf{A}\vec{\hat{x}}$ is orthogonal to the columns of $\mathbf{A}$ by direct manipulation *(i.e. plug the formula for the linear least squares estimate into the error vector and then check if $\mathbf{A}^T$ times the vector is the zero vector.)*

2. Labelling Patients Using Gene Expression Data

Least squares techniques are useful for many different kinds of prediction problems. The core ideas we learned in class have been extensively further developed. These ideas are commonly used in machine learning for finance, healthcare, advertising, image processing, and many other fields. Here, we'll explore how least squares can be used for classification of data in a medical context.

Gene expression data of patients, along with other factors such as height, weight, age, family history, is often used to understand the likelihood that a patient might develop certain common diseases such as diabetes. Gene expression profiles can be read using DNA microarray technology, which uses tissue samples from a patient. This data, along with the patient specific characteristics above, can be combined into a vector to get a set of features that describe each patient.

Many scientific studies look at models in mice to understand how gene expression relates to diabetes. Previous studies have shown that the expression of the tomosin2 and ts1 genes are correlated to the onset of diabetes in mice. How can we predict whether or not a mouse will develop diabetes based on data about this expression as well as other factors of the mouse? We will use some (fake) data to explore this.

We are given information about the age and weight of the mouse and additionally have access to data about whether the genes tomosin2, ts1 and chn1 (a third gene) were expressed or not. The gene expression data is captured using vectors that are $+1$ if the gene is expressed and -1 if the gene is not expressed. Similarly, whether or not a mouse has diabetes is also captured using a $+1,-1$ vector, where $+1$ indicates that the mouse has diabetes. Using this data we would like to develop a linear model that predicts whether or not a mouse will have diabetes.

$$\alpha_1(\text{age}) + \alpha_2(\text{weight}) + \alpha_3(\text{tomosin2}) + \alpha_4(\text{ts1}) + \alpha_5(\text{chn1})$$

We would like the above expression to be positive if the mouse has diabetes and negative if the mouse does not have diabetes.

(a) In problems such as this, it is common to use some *training* data to generate a model. Turns out, a good heuristic for this can be developed using a least squares technique. Set up a linear model for the problem in a format we have used for least squares problems $\mathbf{A}\vec{x} = \vec{b}$. Here, $\vec{b}$ will be a vector with $+1,-1$ entries. The α_i's are your unknowns.

(b) Using the (fake) *training* data `diabetes_train.npy`, generate the linear model using the least squares technique, i.e. find the unknown model parameters for the given data set. Include the unknown parameter values in the writeup of your homework. Use the provided ipython notebook file.

(c) Now it is time to use the model you have developed to make some predictions! It is interesting to note here that we are not looking for a real number to model whether each mouse has diabetes or not, we are looking for a binary label. So we will use the *sign* of the expression above to assign a ± 1 value to each mouse. Predict whether each mouse with the characteristics in the *test* data set `diabetes_test.npy` will get diabetes. There are four mice in the test data set. Include the ± 1 vector that indicates whether or not they have diabetes in your writeup.

3. Image Analysis

Applications in medical imaging often require an analysis of images based on the pixels of the image. For instance, we might want to count the number of cells in a given sample. One way to do this is to "take a picture" of the cells and use the pixels to determine the locations and thus the number of cells. Alternatively, automatic detection of shape is useful in image classification as well (e.g. consider a robot trying to find out autonomously where a mug is in its field of vision).

Let us focus back on the medical imaging scenario. You are interested in finding the exact position and shape of a cell in an image, so you want to find the equation of the ellipse that bounds the cell relative to a given coordinate system that is represented by the image. Your collaborator uses edge detection techniques to find a bunch of points that are approximately along the edge of the cell. We assume that the origin is

in the center of the image with standard axes and collect the following points: $(0.3,-0.69)$, $(0.5,0.87)$, $(0.9,-0.86)$, $(1,0.88)$, $(1.2,-0.82)$, $(1.5,0.64)$, $(1.8,0)$.

Recall that a quadratic equation of the form

$$ax^2+bxy+cy^2+dx+ey=1$$

can be used to represent an ellipse (if $b^2-4ac<0$), and a quadratic equation of the form

$$a(x^2+y^2)+dx+ey=1$$

is a circle if $d^2+e^2-4a>0$. The circle has fewer parameters.

(a) How can you find the equation of a circle that surrounds the cell? First, provide a setup and formulate a set of matrix equations to do this, i.e. an equation of the form $\mathbf{A}\vec{x}=\vec{b}+\vec{e}$, where $\vec{b}$ represents your observations and $\vec{e}$ represents the unknown errors.

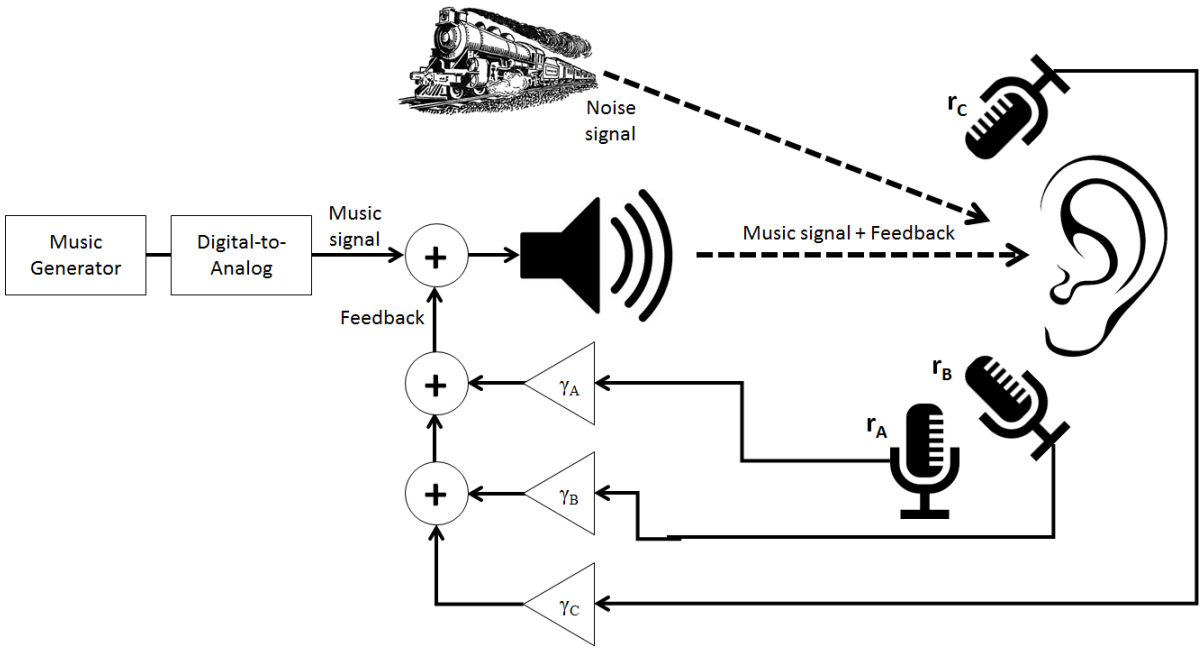
(b) How can you find the equation of an ellipse that surrounds the cell? Provide a setup and formulate a set of matrix equations to do this as above.

(c) In the IPython Notebook, write a short program to fit a circle using these points. If you model your system of equations as $\mathbf{A}\vec{x}=\vec{b}+\vec{e}$, where $\vec{e}$ is the error vector and the number of data points is N, what is $\frac{\|\vec{e}\|}{N}$? Plot your points and the best fit circle in IPython.

(d) Write a short program in IPython to fit an ellipse using these points. If you model your system of equations as $\mathbf{A}\vec{x}=\vec{b}+\vec{e}$, where $\vec{e}$ is the error vector and the number of data points is N, what is $\frac{\|\vec{e}\|}{N}$? Plot your points and the best fit ellipse in IPython. How does this error compare to the one in the previous subpart? Which technique is better?

4. Noise Cancelling Headphones

In this problem, we will explore a common design for noise cancellation, using noise-cancelling headphones as an example application. We will work with the model shown in the figure below.



A music signal is generated at a speaker and transmitted to the listener's ear. If there is noise in the environment (such as other people's voices, a train going by, or just any kind of noise), this noise signal will be superimposed on the music signal and the listener will hear both. In order to cancel the noise, we will try to record the noise and subtract it directly from the transmitted signal, with the hope that we can achieve perfect cancellation of everything but the music. Since our system is imperfect, we'll have to solve a least squares problem.

The gain blocks marked by γ (Greek "gamma") represent scalar multiplication, and we will assume that they can take on any real number, positive or negative.

(a) First, consider a noise signal noted by $\vec{n}$,

$$\vec{n} = \begin{bmatrix} n_1 \\ n_2 \\ n_3 \\ n_4 \\ n_5 \end{bmatrix}$$

We can use three microphones to record this signal, Mic A, Mic B, and Mic C. Each microphone records the noise, but they each have their own characteristics, so that they don't perfectly record the noise and that they are distinct recordings:

$$\vec{r}_A = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix}; \vec{r}_B = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix}; \vec{r}_C = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix}$$

For the system that is drawn in the figure above, and using matrix notation, write down the signal at the listener's ear. It should include the music signal, the noise signal, the recorded signals, and the microphone gains (γ_n). You can assume that the microphones do not pick up the music signal.

(b) Ideally, we would want to have a signal at the ear that matches the original music signal perfectly. In reality, this is not possible, so we will aim to minimize the effect of the noise. What quantity would we need to minimize to make sure this happens? Write your answer in terms of the matrix $\mathbf{R}$, the vector of mic gains $\vec{\gamma}$, and the noise vector $\vec{n}$. $\mathbf{R}$ and $\vec{\gamma}$ are given:

$$\mathbf{R} = \begin{bmatrix} \vec{r}_A & \vec{r}_B & \vec{r}_C \end{bmatrix} = \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \\ a_4 & b_4 & c_4 \\ a_5 & b_5 & c_5 \end{bmatrix}$$

$$\vec{\gamma} = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \end{bmatrix}$$

(c) We can solve minimization problems by the least squares method. In effect, if we have a problem, $\min_{\vec{x}} \left\| \mathbf{A}\vec{x} - \vec{b} \right\|$, then the $\vec{x}$ that solves this problem with the 2-norm can be found through: $\vec{x} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\vec{b}$. Implement this least squares method in the IPython Notebook helper function `doLeastSquares`.

(d) For the given $\vec{n}$ and the recordings, $\vec{r}_A$, $\vec{r}_B$, $\vec{r}_C$, below, report the γ's that minimize the effect of noise.

$$\vec{n} = \begin{bmatrix} 0.10 \\ 0.37 \\ -0.45 \\ 0.068 \\ 0.036 \end{bmatrix}; \vec{r}_A = \begin{bmatrix} 0 \\ 0.11 \\ -0.31 \\ -0.012 \\ -0.018 \end{bmatrix}; \vec{r}_B = \begin{bmatrix} 0 \\ 0.22 \\ -0.20 \\ 0.080 \\ 0.056 \end{bmatrix}; \vec{r}_C = \begin{bmatrix} 0 \\ 0.37 \\ -0.44 \\ 0.065 \\ 0.038 \end{bmatrix}$$

The next few questions can be answered in the IPython notebook by running the associated cells.

(e) We can use this least squares solution to train our algorithm for a given number of microphones and a training signal. Follow the instructions in the IPython notebook to load a music signal and some noise signals. Listen to the music signal and the two noise signals. Comment on what you hear in each audio clip.

(f) Use the IPython notebook to record the first noise signal using the `recordAmbientNoise` function and calculate a vector $\vec{\gamma}$. Create the noise cancellation signal by performing the multiplication $\mathbf{R}\vec{\gamma}$.

(g) Add the noise cancellation signal (with the correct sign) to the music signal to get the signal from the speaker and, finally, add the noise signal to the speaker signal. Play the noisy signal and the noise-cancelled signal. Can you hear a difference?

(h) Try adding the other noise signal to the music signal without re-calculating new values for $\vec{\gamma}$ (don't solve the least squares problem again). Add the noise-cancelling signal to your speaker signal and add the noise as well. Comment on the quality of the resulting noise-cancelled signal. Is it perfect or are there artifacts?

5. The Framingham Risk Score

For most of the parts of this problem, your work will be done in the appropriate section of the IPython notebook.

In Homework 1, we did a problem where we calculated the parameters of the Framingham risk score for predicting cardiovascular disease (CVD). In this problem, we will revisit the parameters of the Framingham risk score in a more realistic setting using the more sophisticated optimization tool of linear least squares. In the problem in Homework 1, we determined four parameters of Framingham risk score from the data from four patients – this amounts to solving four equations with four unknowns. This makes sense if we knew the correct parameters originally but then forgot them. Suppose, however, that we were trying to come up with the correct parameters for the Framingham risk score in the first place. How would we do it?

As a review, the Framingham risk score estimates the 10-year cardiovascular disease (CVD) risk of an individual. There are multiple factors (predictors) that weigh in the calculation of the score. In Homework 1, we simplified the score to only use four factors. Here we will look at more complex version of the score that takes into account six factors including age, total cholesterol, level of high-density lipoprotein (HDL) cholesterol, systolic blood pressure (SBP), whether or not the individual smokes, and whether or not the individual is diabetic.

Scores like this are determined empirically after tracking the characteristics of many medical patients. Once we have data from hundreds or thousands of test subjects, we want to find the parameters that best model the data we are seeing so that we can use our score to predict the probability of heart disease for a new patient. Of course there will be some variability in the probability of heart to disease for each individual but we want to design the parameters of our score so that it predicts their risk as closely as possible.

Linear least squares is a powerful tool for fitting these kind of models to minimize the error between the observed risk of heart disease for each individual and the predicted risk from the model. Linear least squares can even be a powerful tool in many cases when we expect our model to be nonlinear in the data. As long as we can transform the problem so that the model is **linear in the parameters** then we can use linear least squares. For example in the Framingham case, we have reason to believe (from medical modeling considerations) that the probability of the individual suffering from CVD in the next 10 years has the form

$$p = 1 - 0.95^{e^{(R-26.1931)}} \tag{3}$$

where the score R is calculated based on the values of age, total cholesterol (TC), HDL cholesterol, systolic blood pressure (SBP), whether or not the patient is diabetic (DIA), and whether or not the patient smokes (SMK) as follows

$$\begin{aligned} R = & x_1 \cdot \ln\Big(\text{AGE (years)}\Big) + x_2 \cdot \ln\Big(\text{TC (mg/dL)}\Big) + \\ & x_3 \cdot \ln\Big(\text{HDL (mg/dL)}\Big) + x_4 \cdot \ln\Big(\text{SBP (mm Hg)}\Big) + \\ & x_5 \cdot \Big(\text{DIA (binary)}\Big) + x_6 \cdot \Big(\text{SMK (binary)}\Big) \end{aligned} \tag{4}$$

DIA and SMK are binary variables that indicate whether or not the subject has diabetes and smokes respectively whereas AGE, TC, HDL, and SBP are all numeric values. Note also that AGE, TC, HDL, and SBP are passed through the natural log function $\ln(\cdot)$ where as DIA and SMK are not. For patient k, we will represent the probability as p^k and the score as R^k.

(a) We want to transform the probabilities and the input data (AGE, TC, HDL, SBP, DIA, SMK) for patient k into the form

$$b^k = x_1 A_1^k + x_2 A_2^k + x_3 A_3^k + x_4 A_4^k + x_5 A_5^k + x_6 A_6^k \tag{5}$$

in order to solve for the parameters $\vec{x} = [x_1, x_2, x_3, x_4, x_5, x_6]^T$. How can we transform the probabilities and the input data to express Equation (3) in the form of Equation (5), i.e. express b^k, A_1^k, A_2^k, A_3^k, A_4^k, A_5^k and A_6^k be in terms of p^k, AGE^k, TC^k, HDL^k, SBP^k, DIA^k, SMK^k. In the ipython notebook, load in the data file `CVDdata.mat` and apply these transformations to the appropriate variables.

Credit: The data was obtained from the Center for Disease Control and Prevention's (CDC) National Health and Nutrition Examination Survey (NHANES) dataset (October 2015) (https://www.cdc.gov/nchs/nhanes/index.htm).

(b) Now that we have transformed the problem into a linear problem, we want to use linear least squares to estimate the parameters $\vec{x}$. In order to do this we set up a system of equations in matrix form

$$\vec{b} = \mathbf{A}\vec{x}$$

where $\mathbf{A}$ is a tall matrix and $\vec{b}$ is a tall vector. What form should $\vec{b}$ and $\mathbf{A}$ have in terms of b^k, A_1^k, A_2^k, A_3^k, A_4^k, A_5^k, A_6^k? The data we loaded in python is for 91 patients. Construct $\vec{b}$ and $\mathbf{A}$ using the loaded data. What are the dimensions of $\vec{b}$ and A?

(c) We want to choose our estimate $\vec{\hat{x}}$ to minimize $\|\vec{b} - \mathbf{A}\vec{x}\|^2$. Use the linear least squares formula to find the best fit parameters $\vec{\hat{x}}$. What is the $\vec{\hat{x}}$ that you found?

(d) Now that we've found the best fit parameters $\vec{\hat{x}}$, write an expression for $\vec{\hat{b}}$, our model's prediction for the values of $\vec{b}$ given the data $\mathbf{A}$ and our estimate of the parameters $\vec{\hat{x}}$ and compute the squared error $\|\vec{b} - \vec{\hat{b}}\|^2$. What is the squared error that you computed?

(e) Since this problem has many parameters, it is difficult to visualize what is going on. One thing we can do to get a feel for the data and check that our fit is good is to plot it in a lower dimensional subspace. For example, we could plot b by A_1 or A_2 or A_3 individually. (A_1 is the vector of A_1^k's for all patients. It is the first column of $\mathbf{A}$. Similarly for A_2 and A_3.) In your IPython notebook, plot b by A_1, b by A_2, and b by A_3 individually using the plotting option `'ob'` to plot the data as blue dots. For each plot you should see a blue point cloud. What is actually happening here is that we're projecting the data onto the A_1-b plane, the A_2-b plane, and the A_3-b plane respectively. Now plot your models prediction $\hat{b}$ by A_1, A_2, and A_3 on the appropriate plots using the plotting option `'or'` to plot the predictions as red dots (refer to the IPython notebook for reference code). Does it look like your model is a good fit?

(f) To better visualize the linearity of the model, we will calculate the risk as a function of A_2 alone and plot it. The rest of the predictors will be fixed in this part. We will use the following values for the other parameters. Age=40 years, HDL=25 mg/dL, SBP=220 mm Hg, DIA=1 and SMK=1. In the IPython notebook, we have generated a block of code that you need to complete to make the calculation of predicted b values from the above parameters. Fill the code and plot the estimated b values vs. A_2 values. Is the plot linear?

Hint: Don't forget to apply the appropriate transformation to the different parameters.

(g) Try changing the parameters $\vec{x}$ slightly and re-plotting. Does it look like the fit is getting better or worse? Is the squared error increasing or decreasing?

(h) (Optional) Transform b and $\hat{b}$ back into the form of p and transform A_1, A_2, A_3 back into the form of AGE, TC, and HDL and re-plot. What do you see?

(i) (Optional) Use the values for b from part (f) to calculate p as a function of TC. Plot the curve p vs TC. Is the plot linear? What does this plot portray?

Note: Some of the values in the algorithm were modified from the original study values.

6. Mechanical Gram-Schmidt

(a) Use Gram-Schmidt to find a matrix $\mathbf{U}$ whose columns form an orthonormal basis for the column space of $\mathbf{V}$.

$$\mathbf{V} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

and show that you get the same resulting vector when you project $\vec{w} = \begin{bmatrix} 1 & -1 & 0 & -1 & 0 \end{bmatrix}^T$ onto $\mathbf{V}$ and onto $\mathbf{U}$, i.e. show that

$$\mathbf{V}(\mathbf{V}^T\mathbf{V})^{-1}\mathbf{V}^T\vec{w} = \mathbf{U}(\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T\vec{w}$$

(b) Use Gram-Schmidt to find a matrix $\mathbf{U}$ whose columns form an orthonormal basis for the column space of $\mathbf{V}$.

$$\mathbf{V} = \begin{bmatrix} 1 & 1 & -1 \\ 0 & 1 & -1 \\ 1 & 0 & 0 \\ 1 & -1 & 1 \\ 0 & -1 & 1 \end{bmatrix}$$

and show that you get the same resulting vector when you project $\vec{w} = \begin{bmatrix} 1 & -1 & 0 & -1 & 0 \end{bmatrix}^T$ onto $\mathbf{V}$ and onto $\mathbf{U}$.

(c) Compute the QR decomposition of the above matrices, that is, find an orthogonal matrix $\mathbf{Q}$ and an upper triangular matrix $\mathbf{R}$, such that $\mathbf{V} = \mathbf{QR}$.

7. Homework process and study group

Who else did you work with on this homework? List names and student ID's. (In case of hw party, you can also just describe the group.) How did you work on this homework?
Working in groups of 3-5 will earn credit for your participation grade.