EECS 16A Designing Information Devices and Systems I
Spring 2021 Homework 12

This homework is due April 23, 2021, at 23:59.
Self-grades are due April 26, 2021, at 23:59.

Submission Format
Your homework submission should consist of **one** file.

- `hw12.pdf`: A single PDF file that contains all of 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 "printout" of your IPython notebook, you will not receive credit for problems that involve coding. Make sure that your results and your plots are visible. Assign the IPython printout to the correct problem(s) on Gradescope.

Submit the file to the appropriate assignment on Gradescope.

1. Reading Assignment

For this homework, please review Note 22 (Trilateration and Correlation), and read Note 23 (Least Squares). You are always encouraged to read beyond this as well.

(a) In trilateration, the distances between the beacons and the unknown location $\vec{x}$ involve quadratic terms of $\vec{x}$. What trick can we use to get a system of linear equations in $\vec{x}$?

(b) Suppose the signal $x[n]$ is only defined for timesteps $0, 1, \cdots, 5$. For the purpose of computing linear cross-correlation, what value of $x[n]$ do we assume when n is a timestep out of the range: $0 \leq n \leq 5$ (e.g. $n = 6$ or $n = -1$)?

2. Mechanical Trilateration

Learning Goal: *The objective of this problem is to practice using trilateration to find the position based on the distance measurements and known beacon locations.*

Trilateration is the problem of finding one's coordinates given distances from known beacon locations. For each of the following trilateration problems, you are given 3 beacon locations ($\vec{s}_1$, $\vec{s}_2$, $\vec{s}_3$) and the corresponding distance (d_1, d_2, d_3) from each beacon to your location. Find your location or possible locations. If a solution does not exist, state that it does not.

(a) $\vec{s}_1 = \begin{bmatrix}4\\5\end{bmatrix}, d_1 = 5, \vec{s}_2 = \begin{bmatrix}1\\-1\end{bmatrix}, d_2 = 2\,, \vec{s}_3 = \begin{bmatrix}-11\\6\end{bmatrix}, d_3 = 13.$

(b) $\vec{s}_1 = \begin{bmatrix}0\\0\end{bmatrix}, d_1 = 5\sqrt{2}, \vec{s}_2 = \begin{bmatrix}10\\0\end{bmatrix}, d_2 = 5\sqrt{2}\,, \vec{s}_3 = \begin{bmatrix}5\\0\end{bmatrix}, d_3 = 5.$

(c) $\vec{s}_1 = \begin{bmatrix}3\\4\end{bmatrix}, d_1 = 5, \vec{s}_2 = \begin{bmatrix}0\\-2\end{bmatrix}, d_2 = 2\,, \vec{s}_3 = \begin{bmatrix}-12\\5\end{bmatrix}, d_3 = 12.$

3. Mechanical Projections

Learning Goal: *The objective of this problem is to practice calculating projection of a vector and the corresponding squared error.*

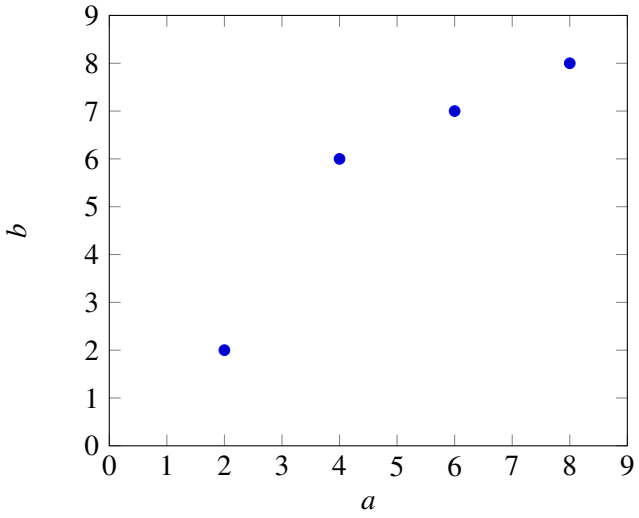
(a) Find the projection of $\vec{b} = \begin{bmatrix} 3 \\ 2 \\ -1 \end{bmatrix}$ onto $\vec{a} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$. What is the squared error between the projection and $\vec{b}$, i.e. $\|e\|^2 = \|\text{proj}_{\vec{a}}(\vec{b}) - \vec{b}\|^2$?

(b) **(OPTIONAL)** Find the projection of $\vec{b} = \begin{bmatrix} 1 \\ 4 \\ -5 \end{bmatrix}$ onto the column space of $\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$. What is the squared error between the projection and $\vec{b}$, i.e. $\|e\|^2 = \|\text{proj}_{\text{Col}(\mathbf{A})}(\vec{b}) - \vec{b}\|^2$?

4. Mechanical Least Squares

Learning Goal: *The goal of this problem is to use least squares to fit different models (i.e. equations) to a data set and find the squared error of each model.*

Depending on the least squares model's number of parameters, the model will fit the data better or worse. A better model results in a lower squared error than a worse one. In part (a), we consider a linear model that contains a single slope parameter and intercepts the vertical axis at zero. In part (b), we consider a linear model with a possibly non-zero vertical axis intercept parameter, also known as an affine model.



a	2	4	6	8
b	2	6	7	8

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

$$\vec{a}x = \vec{b}$$

that best fits the data, where x is a scalar that minimizes the squared error

$$\|\vec{e}\|^2 = \left\| \begin{bmatrix} a_1 \\ \vdots \\ a_4 \end{bmatrix} x - \begin{bmatrix} b_1 \\ \vdots \\ b_4 \end{bmatrix} \right\|^2 = \|\vec{a}x - \vec{b}\|^2.$$

Note that we can model this linear model as a generic system $\mathbf{A}\vec{x} = \vec{b}$ where $\mathbf{A} = [\vec{a}]$ and $\vec{x} = [x]$. Once you've computed the optimal solution $\hat{\vec{x}}$, compute the squared error between your model's prediction $\mathbf{A}\hat{\vec{x}}$ and the actual b values.

You may use a calculator but show your work. Do not directly plug your numbers into IPython.

Note: By using this linear model, we are implicitly forcing the line to go through the origin.

Optional but recommended: Plot the best fit line along with the data points to examine the quality of the fit. You may plot however you wish - one option is to use the helper code in the IPython notebook provided.

(b) Now, let us consider an affine model for the same data, i.e. one with a non-zero vertical b-intercept. We believe we can get a better fit for the data by assuming an affine model of the form

$$ax_1 + x_2 = b,$$

for each point. Note that x_1 is the slope and x_2 is the b-intercept here. We can write this equation jointly for all the points using the vector notation below:

$$\vec{a}x_1 + \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} x_2 = \vec{b}.$$

Make sure you understand why a column vector of 1's is required above. In order to do this, we need to augment our $\mathbf{A}$ matrix from the previous part for the least squares calculation with a column of 1's (do you see why?), so that it has the form

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

Set up a least squares problem to find the optimal x_1 and x_2 and compute the squared error between your model's prediction and the actual $\vec{b}$ values. Is it a better fit for the data? Provide a quantitative, numerical justification.

Optional: Plot your affine model and examine qualitatively how close the best fit line is to the data points compared to part (a). You may plot however you wish - one option is to use the helper code in the IPython notebook provided.

(c) **Prove the following theorem.**

Let $\mathbf{A} \in \mathbb{R}^{m \times n}$. If $\hat{\vec{x}}$ is the solution to the least squares problem

$$\min_{\vec{x}} \left\| \mathbf{A}\vec{x} - \vec{b} \right\|^2,$$

then the error vector $\mathbf{A}\hat{\vec{x}} - \vec{b}$ is orthogonal to the columns of $\mathbf{A}$, i.e. show that: $\mathbf{A}^T(\mathbf{A}\hat{\vec{x}} - \vec{b}) = \vec{0}$.

Note: Trying to show individually that $\langle \vec{a}_i, \mathbf{A}\hat{\vec{x}} - \vec{b} \rangle = \vec{a}_i^T(\mathbf{A}\hat{\vec{x}} - \vec{b}) = 0$ for $i = 0, \ldots, n$, where $\vec{a}_i$ is the ith column of $\mathbf{A}$ can be a bit tricky, however, stacking all of the $\vec{a}_i^T$ on top of each other and showing that $\mathbf{A}^T(\mathbf{A}\hat{\vec{x}} - \vec{b}) = \vec{0}$ is easier.

For this question, it is sufficient to prove that $\mathbf{A}^T(\mathbf{A}\hat{\vec{x}} - \vec{b}) = \vec{0}$.

Hint: Can you substitute $\hat{\vec{x}}$ using the least-squares formula?

5. GPS Receivers

Learning Goal: *This problem will help to understand how GPS satellites transmit encoded signals to GPS receivers and how a receiver decodes the received signals and calculates the distance to the satellites using the signal propagation delays. It also shows how the GPS is designed to be immune to noise.*

The Global Positioning System (GPS) is a space-based satellite navigation system that provides location and time information in all weather conditions, anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites. In this problem, we will understand how a receiver (e.g. your cellphone) can disambiguate signals from the different GPS satellites that are simultaneously received.

GPS satellites employ a special coding scheme where each GPS satellite uses a unique 1023 element long sequence as its "signature." These codes used by the satellites are called "Gold codes," and they have some special properties:

- The auto-correlation of a Gold code (cross-correlation with itself) is very **high** at the 0^{th} shift and very **low** at all other shifts.
- The cross-correlation between different Gold codes is very **low** at all shifts, i.e. different Gold codes are almost orthogonal to each other.

The important thing to know is that the Gold codes are 1023 element vectors where each element is either $+1$ or -1, and that any Gold code is "almost orthogonal" to any other Gold code.

A receiver listening for signature transmissions from a satellite has copies of all of the different GPS satellites' Gold codes. The receiver can determine how long it took for a particular GPS satellite's signal to reach it by **taking the cross-correlation of the received signal with a satellite's Gold code. (The Gold code is the signal which is shifted during cross-correlation — as discussed in lecture.)** The shift value (delay) that corresponds to distinct peaks (positive/negative) in the correlation determines the "propagation delay" between when the GPS satellite transmitted it's signal and when the receiver received it. This time delay can then be converted into a distance (*in the case of GPS, electromagnetic waves are used for transmissions, distance is equal to the speed of light multiplied by the time delay*).

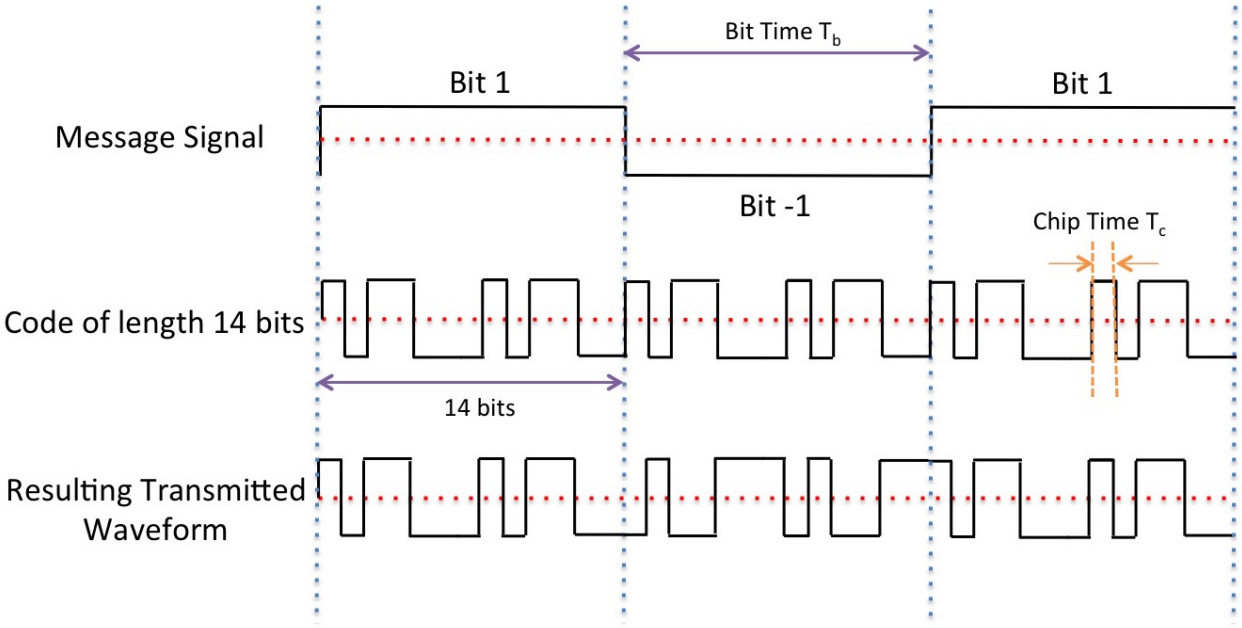
The GPS satellite is constantly transmitting its signature. In addition to identifying itself through its signature, it can also **"modulate" the signature** to communicate more information. Modulating a signature means multiplying the entire signature block by $+1$ or -1, as shown in the figure.

In the figure below, the signature is of length 14 as an example, i.e. it is made of 14 symbols (each one is either $+1$ or -1). T_c is the duration of one ± 1 symbol, and T_b is the duration of a whole signature. The figure shows 3 blocks of length 14 being transmitted. The message signal (made of $+1$ and -1 as well) multiplies the entire block of the signature, to give the resulting transformed waveform at the bottom of the figure. The message being transmitted in the figure is $\begin{bmatrix}1 & -1 & 1\end{bmatrix}$. So to send these three symbols of message, we need to send 14×3 symbols of the gold code.

Now, when a receiver receives a signal, in addition to finding the time delay between transmission and reception, the receiver will be able to decode the message by noting a very positive correlation if the message bit is equal to 1, and a very negative correlation if the message bit is equal to -1.

For the problem you will now do, $T_b = 1023T_c$.

You will use the ideas of linear correlation to figure out which of the satellites are transmitting.



For the purpose of this question we only consider 24 GPS satellites. Download the IPython notebook and the corresponding data files for the following questions.

Note: this code is calculation-heavy, and can take up to a few minutes to run for each code block. Be patient!

The iPython code required for part (a) is already given for you. You only need to write codes for parts (b)–(g).

(a) Auto-correlate (i.e. cross-correlate with itself) the Gold code of satellite 10 and plot it. What do you observe? Use the helper function `array_correlation` in the notebook to perform correlation for all sub-parts of this problem. Try to understand what the helper function `array_correlation` is doing.

(b) Cross-correlate the Gold code of satellite 10 with satellite 13 and plot it. What do you observe?

(c) Consider a random signal, i.e. a signal that is not generated due to a specific code but is a random ± 1 sequence. A helper function `integernoise_generator` in the notebook will generate this for you. Cross-correlate it with the Gold code of satellite 10. What do you observe? What does this mean about our ability to identify satellites in the presence of random ± 1 noise?

(d) The signal actually received by a receiver will be the satellites' transmissions plus additive noise, and this need not be just noise that takes values ± 1. Use the helper function `gaussiannoise_generator` in the notebook to generate a random noise sequence of length 1023, and compute the cross-correlation of this sequence with the Gold Code of satellite 10. What does this mean about our ability to identify satellites in the presence of real-valued noise?

For the next subparts of this problem, the received signals are corrupted by real-valued noise. Use the observation from this subpart for solving the rest of the question.

(e) The receiver may receive signals from multiple satellites simultaneously, in which case the signals will all be added together. In addition, noise might be added to the signal. What are the satellites present in the received signal from `data1.npy`?

Use helper function `find_peak` in your code to help you figure out whether peak correlation values of magnitude greater than a pre-specified threshold value are present. Note that the threshold is 800 for this problem.

(f) Let's assume that you can hear only one satellite, Satellite X, at the location you are in (though this never happens in reality). Let's also assume that this satellite is transmitting an unknown sequence of $+1$ and -1 of length 5 (after encoding it with the 1023 bit Gold code corresponding to Satellite X).

First, find out from `data2.npy` which satellite is transmitting using the same procedure that you used in part (e).

Next, find the 5 element sequence of ± 1's that is being transmitted. To do this, you can observe the cross-correlation of the received signal from `data2.npy` with the Gold code of Satellite X and then visually find the peaks (positive /negative) and use these to understand the message.

(g) **(OPTIONAL)**

Signals from different transmitters arrive at the receiver with different delays. We use these delays to figure out the distance between the satellite and receiver.

The signals from different satellites are superimposed on each other with different offsets at the start.

What satellites are you able to see in `data3.npy`? Assume that all satellites begin transmission at time 0. What are the delays of all the satellites that are present? Assume you are told that all the satellites have the same message signal given by $\begin{bmatrix}1 & 1 & -1 & -1 & -1\end{bmatrix}$.

6. Audio File Matching

Learning Goal: *This problem motivates the application of correlation for pattern matching applications such as Shazam.*

Many audio processing applications rely on representing audio files as vectors, referred to as audio *signals*. Every component of the vector determines the sound we hear at a given time. We can use inner products to determine if a particular audio clip is part of a longer song, similar to an application like *Shazam*.

Let us consider a very simplified model for an audio signal, $\vec{x}$. At each timestep k, the audio signal can be either $x[k] = -1$ or $x[k] = 1$.

(a) Say we want to compare two audio files of the same length N to decide how similar they are. First, consider two vectors that are exactly identical, namely $\vec{x}_1 = \begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}^T$ and $\vec{x}_2 = \begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}^T$. What is the inner product of these two vectors? What if $\vec{x}_1 = \begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}^T$ but $\vec{x}_2$ oscillates between 1 and -1? Assume that N, the length of the two vectors, is an even number.

Use this to suggest a method for comparing the similarity between a generic pair of length-N vectors.

(b) Next, suppose we want to find a short audio clip in a longer one. We might want to do this for an application like *Shazam*, which is able to identify a song from a short clip. Consider the vector of length 8, $\vec{x} = \begin{bmatrix}-1 & 1 & 1 & -1 & 1 & 1 & -1 & 1\end{bmatrix}^T$.

We want to find the short segment $\vec{y} := \begin{bmatrix}y[0] & y[1] & y[2]\end{bmatrix}^T = \begin{bmatrix}1 & 1 & -1\end{bmatrix}^T$ in the longer vector. To do this, perform the linear cross correlation between these two finite length sequences and identify at what shift(s) the linear cross correlation is maximized. Apply the same technique to identify what shift(s) gives the best match for $\vec{y} = \begin{bmatrix}1 & 1 & 1\end{bmatrix}^T$.

(If you wish, you may use iPython to do this part of the question, but you do not have to.)

(c) Now suppose our audio vector is represented using integers beyond simply just 1 and -1. Find the short audio clip $\vec{y} = \begin{bmatrix}1 & 2 & 3\end{bmatrix}^T$ in the song given by $\vec{x} = \begin{bmatrix}1 & 2 & 3 & 1 & 2 & 2 & 3 & 10\end{bmatrix}^T$. Where do you expect to see the peak in the correlation of the two signals? Is the peak where you want it to be, i.e. does it pull out the clip of the song that you intended? Why?

(If you wish, you may use iPython to do this part of the question, but you do not have to.)

(d) Let us think about how to get around the issue in the previous part. We applied cross-correlation to compare segments of $\vec{x}$ of length 3 (which is the length of $\vec{y}$) with $\vec{y}$. Instead of directly taking the cross correlation, we want to normalize each inner product computed at each shift by the magnitudes of both segments, i.e. we want to consider $\frac{<\vec{x}_k, \vec{y}>}{\|\vec{x}_k\|\|\vec{y}\|}$, where $\vec{x}_k$ is the length 3 segment starting from the k-th index of $\vec{x}$. This is referred to as normalized cross correlation. Using this procedure, now which segment matches the short audio clip best?

(e) We can use this on a more 'realistic' audio signal – refer to the IPython notebook, where we use normalized cross-correlation on a real song. Run the cells to listen to the song we are searching through, and add a simple comparison function `vector_compare` to find where in the song the clip comes from. Running this may take a couple minutes on your machine, but note that this computation can be highly optimized and run super fast in the real world! Also note that this is not exactly how Shazam works, but it draws heavily on some of these basic ideas.

7. Homework Process and Study Group

Who did you work with on this homework? List names and student ID's. (In case you met people at homework party or in office hours, you can also just describe the group.) How did you work on this homework? If you worked in your study group, explain what role each student played for the meetings this week.