CSM: EECS 16A **(Designing Information Devices and Systems I)**

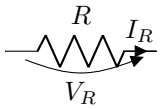
Worksheet #13 Metas

Term: **Fall 2019** *Name:*

Problem 1: And You Thought You Could Ignore Circuits Until Dead Week

Learning Goal: Understand when Least Squares is helpful for estimating values, and how to translate a word problem with given data points into a Least Squares set up.

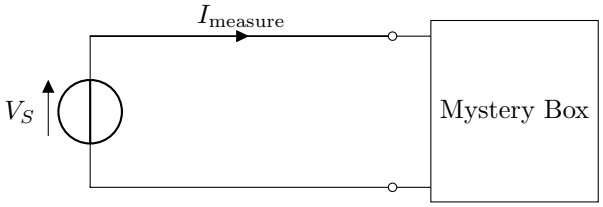
1. Write Ohm's Law for a resistor.



Solution: For a resistor, we have:

$$V_R = I_R R$$

2. You're given the following test setup and told to find R_{Th} between the two terminals of the mystery box. What is R_{Th} of the mystery box between the two terminals in terms of V_S and I_{measure}?



Solution:

$$R_{Th} = \frac{V_S}{I_{\text{measure}}}$$

$$R_{Th} = \frac{V_S}{I_{\text{measure}}}$$

3. You think you've figured out how to find R_{Th}! You've taken the following measurements:

Measurement #	I_{measure}	V_S
1	$1A$	$1.25kV$
2	$2A$	$1kV$
3	$3A$	$4kV$
4	$4A$	$3.5kV$

Using the information above, formulate a least squares problem whose answer provides an estimate of R_{Th}.

Meta: Since voltage and current are directly proportional, as seen in Ohm's Law, there is no additional input. Therefore, the corresponding matrix A is a 4 x 1 vector.

Solution: According to Ohm's Law, $V = IR$. We are estimating the resistance so R_{Th} corresponds to $\vec{x}$ in the $A\vec{x} = \vec{b}$ Least Squares equation. Additionally, I corresponds to A and V corresponds to $\vec{b}$.

$$A = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \quad \vec{b} = \begin{bmatrix} 1.25 \\ 1 \\ 4 \\ 3.5 \end{bmatrix}$$

By running Least Squares, $R_{Th} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\vec{b} = 975\,\text{k}\Omega$

Problem 2: The OP OMP Foundations

Learning Goal: Understand the motivation for OMP and how the algorithm works.

OMP is an algorithm that solves an equation of the form $A\vec{x} = \vec{b}$. This make look familiar, as we have discussed other ways of solving this equation, like Gaussian elimination, inverse matrices, or least squares. In this question, we will explore when and why we would want to use OMP instead of these other techniques.

Meta: This first half of the problem focuses on the motivation of and assumptions made in the OMP algorithm.

1. In the equation $A\vec{x} = \vec{b}$, what are A, $\vec{x}$, and $\vec{b}$? Specifically, what are their dimensions, and what do they mean in the context of a real-world problem like GPS trilateration?

   - What does A represent? What are its dimensions? What do A_i and $A_{i,j}$ represent?

     Solution: A is an $n \times m$ matrix, where m is the number of devices and n is the signal broadcasted by each device. Therefore, A_i is the signal from the ith device, and $A_{i,j}$ is the jth element of the signal.

   - What does $\vec{x}$ represent? What are its dimensions? What does x_i represent?

     Solution: $\vec{x}$ is an $m \times 1$ matrix, so we just say it is a vector of size m. x_i represents the scaling factor (and the strength) of the signal sent out by the ith device.

   - What does $\vec{b}$ represent? What are its dimensions?

     Solution: $\vec{b}$ is an $n \times 1$ matrix, so we just say it is a vector of size n. $\vec{b}$ represents the signal the signal received by the receiver.

2. To solve $A\vec{x} = \vec{b}$, what assumptions do we need to make in order to use each of the following techniques?

   Meta: Try to explain how each of these assumptions would be violated in the context of the GPS problem.

   - Gaussian elimination

     Solution: There must be no noise in our received signal $\vec{b}$.

   - Inverse matrices

     Solution: A must be square and its columns must be linearly independent. Note that GE can be used even when A is not square.

   - Least squares

     Solution: A^TA must be invertible, which in turn means A must have a non-trivial nullspace (see Note 23 for more details.) This also means A must have linearly independent columns.

3. What specific assumptions does the OMP algorithm make? *Hint: there's three main ones.*

   Solution:

   - There are many devices that could be broadcasting signals
   - The signals sent by each device are nearly orthogonal to each other
   - Only a few devices are actually sending out signals, i.e. there are few i for which x_i would be nonzero.

4. What do these assumptions imply about

- the dimensions of A, the matrix whose columns are all our possible beacons?
- the relationship between a_i and a_j, two of our possible beacon signals?
- the scaling factors in $\vec{x}$?

Solution:

- A should be short and wide, as we have many possible signals
- $< \vec{a_i}, \vec{a_j} > \approx 0$, as our signals are nearly orthogonal
- most x_i are 0, as $\vec{x}$ is sparse

Note that A here is the A in our original equation $A\vec{x} = \vec{b}$, not the same as the A constructed by the OMP algorithm. The A constructed by OMP simply omits all beacons that we believe to not be broadcasting, i.e. $\vec{a_i}$ for which x_i is 0.

OMP comes with one more twist: previously, we received a linear combination of our beacons signals, where

$$\vec{b} = x_1\vec{a_1} + \cdots + x_m\vec{a_m}$$

where $\vec{b}$ is the received signal, and each $\vec{a_i}$ is a beacon signal.

In OMP, instead of just receiving a linear combination of beacons, we receive a linear combination of *shifted versions* of our beacon signals. In other words,

$$\vec{b} = x_1\vec{a_1}^{(\tau_1)} + \cdots + x_m\vec{a_m}^{(\tau_m)}$$

where signal $\vec{a_i}^{(\tau_i)}$ denotes $\vec{a_i}$ shifted forward in time by τ_i.

5. How does receiving shifted versions of signals affect our assumptions?

Meta: It may help to draw on the board what it means to shift the signal *forward* in time, just to remind your students.

Solution: Instead of our signals being nearly orthogonal to each other, they must now be nearly orthogonal to all shifts of themselves, i.e.

$$< \vec{a_i}^{(\tau_i)}, \vec{a_j}^{(\tau_j)} >$$

where $i \neq j$ and $\tau_i \neq \tau_j$.

We now know what assumptions are necessary to apply OMP. Now let's talk about the algorithm! OMP as a whole revolves around a simple strategy. We keep track of how much of the received signal is still unaccounted for, find the strongest device within that signal, subtract out that component from the unaccounted for signal, and repeat until we can't find more beacon signals. At a high level, this is what the steps of the algorithm look like:

Meta: We now focus on the steps of the algorithm itself.

- We initialize A_{curr} to be our matrix of beacons we know to be a part of the received signal $\vec{b}$, and $\vec{x}_{curr}$ to be our guesses for the weights of these beacons within $\vec{b}$.
- We assume there to be at most k beacons that are broadcasting.
- While the magnitude of the difference between our guess for $\vec{b}$ and the actual value of $\vec{b}$ is above some threshhold AND we've not yet found k beacons:
  - Find the strongest beacon yet unaccounted for, and add it to A_{curr}.
  - Update your estimate for $\vec{x}_{curr}$.

6. How can we encode each of these steps in mathematical terms?

(a) Let $\vec{e}$ be the difference between our guess of $\vec{b}$ and the actual value of $\vec{b}$. How do we write an expression for $\vec{e}$.

Solution: $\vec{e}$ represents the amount of the original signal that's yet unaccounted for by our guess. Our guess is $A_{curr}\vec{x}_{curr}$, so our error vector is

$$\vec{e} = \vec{b} - A_{curr}\vec{x}_{curr}$$

(b) Given A_{curr} and $\vec{x}_{curr}$, how do we find the strongest remaining shifted signal $\vec{a_i}^{(\tau_i)}$? *Hint: How do we get an expression for how much of the signal is still unaccounted for by the vectors in A?*

Solution: $\vec{e}$, found in the previous problem, actually describes how much of the signal is not yet accounted for. To find the strongest beacon, we now just cross-correlate $\vec{e}$ with each $\vec{a}_i$. The correlation with the largest value tells us which $\vec{a}_i$ it is, and the index within the correlation vector is the amount this signal was shifted by.

(c) Given guessed signals A_{curr}, received signal $\vec{b}$, the signal we just found $\vec{a}$, and old estimate $\vec{x}_{curr,old}$, how do we get our new estimate for $\vec{x}_{curr,new}$?

Solution: We use least squares to approximate $\vec{x}$.

$$\vec{x}_{curr,new} = (A^T A)^{-1} A^T \vec{b}$$

Let's walk through the algorithm with some example values now.

Suppose we have 100 different beacons, each broadcasting a message $\vec{a}_i$. Let our maximum number of signals $k = 2$. Suppose we receive the signal $\vec{b} = x_{40}\vec{a_{40}}^{13} + x_{100}\vec{a_{100}}^{8}$, where $x_{40} = 100$ and $x_{100} = 10$

7. At the start of our algorithm, how much of the received signal is still unaccounted for?

8. Our next step is to find the strongest unaccounted for beacon within $\vec{e}$. How would we do this, and which vector would this give us?

Solution: If we were to cross-correlate $\vec{b}$ with each $\vec{a}_i$, we would find that $\vec{a_{40}}$ with a shift of 13 would have the highest correlation.

Meta: It may help to draw out what the shape of the correlation matrix looks like and what each term means, as it may be somewhat tedious to do this with actual numbers.

9. How do we estimate our new value of $\vec{x}_{curr}$? Approximately what would it be?

Solution: We would perform least squares with $A = \begin{bmatrix} | \\ \vec{a_{40}} \\ | \end{bmatrix}$ to give us $\vec{x}_{curr} \approx [100]$. The value won't exactly be 100, because there may be some noise in our received $\vec{b}$ and since the $\vec{a_i}$ aren't exactly orthogonal, some of the scaling of another signal might be erroneously accounted for here.

10. After the last step, how much of $\vec{b}$ remains unaccounted for?

Solution: $\vec{e} = \vec{b} - A\vec{x}_{curr}$

11. How would we find the next strongest unaccounted for beacon, and which vector would we get?

Solution: We cross-correlate as before, and we would find $\vec{a_{100}}$ with a shift of 8 to give the highest correlation.

12. How do we estimate our new value of $\vec{x}_{curr}$? Approximately what would it be?

    Solution: We do least squares with $A = \begin{bmatrix} | & | \\ \vec{a_{40}} & \vec{a_{100}} \\ | & | \end{bmatrix}$. This would yield $\vec{x}_{curr} \approx \begin{bmatrix} 100 \\ 10 \end{bmatrix}$.

    We can now terminate OMP, as we have performed 2 iterations of the algorithm, thereby finding $k = 2$ signals.

Problem 3: Revisiting the Acoustic Positioning System

Learning Goal: Understand how to apply properties of inner products, cross correlations, trilateration, least squares, and OMP to build an acoustic positioning system. Learn how to identify the pros/cons when applying different techniques in building the system.

In this question, we will revisit the **Acoustic Positioning System** (APS) and learn how to build it from the ground up using what we know about cross correlation, trilateration, least squares, and Orthogonal Matching Pursuit (OMP).

Recall that in an APS, we have a number of satellites (let's say there are m) transmitting gold codes, and you are a person standing at a location with the coordinate $\vec{x}$, with your phone as the receiver of the signals.

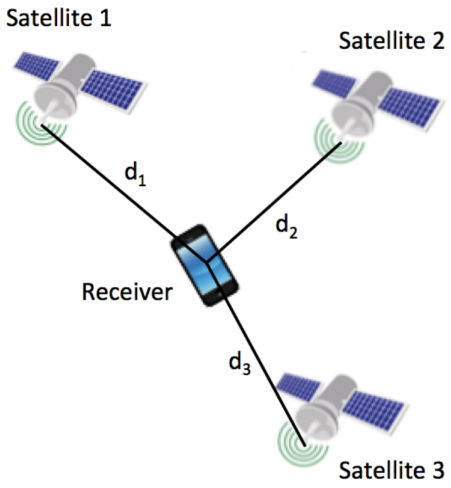
You receive a linear combination of these transmitted signals, each delayed by (τ_i):

$$\vec{r} = \alpha_1 \vec{s}_1^{\,(\tau_1)} + \alpha_2 \vec{s}_2^{\,(\tau_2)} + \ldots + \alpha_m \vec{s}_m^{\,(\tau_m)}$$

As shown in the expression above, each signal is scaled by a constant which is a *"message"* the satellite encodes into its signal while transmitting.

To solve for our current position, we can set up a system of equations based on our current position $\vec{x}$, the position of each satellite $\vec{p}_1, \vec{p}_2, \ldots, \vec{p}_m$, and the distance from our current position to each satellite d_i.

Here's an illustration of the APS (given 3 satellites):



1. Based on the provided information above, which of the following variables are known? Which are unknown (the ones we are trying to solve for)?

   - The position of each satellite $\vec{p}_i$
   - Our current position $\vec{x}$
   - The transmitted signals $\vec{s}_i^{\,(\tau_i)}$
   - The distances from our current position to each satellite d_i
   - The time delay of each signal $\tau_1, \ldots, \tau_m$
   - The messages $\alpha_1, \ldots, \alpha_m$

Solution:

- The position of each satellite $\vec{p}_i$: These are **known** variables since we as the engineer have placed these satellites out in space.
- Our current position $\vec{x}$: This is an **unknown** variable that we are trying to determine using the satellites and received signal.
- The transmitted signals $\vec{s}_i^{\,(\tau_i)}$: These are **known** variables that we as the engineer have programmed, hopefully using Gold Codes.
- The distances from our current position to each satellite d_i : These an **unknown** variables since we don't know our current position $\vec{x}$.
- The time delay of each signal $\tau_1, \ldots, \tau_m$. These are **unknown** variables as well, since we don't know the distance from our current position to the satellites. Note that d_i and τ_i are related by the physics equation $d = v \cdot t$ where v is the speed of light and t is the time delay in seconds.
- The messages $\alpha_1, \ldots, \alpha_m$: These are **unknown** variables that we need to solve for since we receive one signal $\vec{r}$ that is a linear combination of the satellite signals and α_i are the weights of this linear combination.

2. We will start by solving for the distances d_i and the time delays τ_i. How can we compute these quantities using the received signal $\vec{r}$ and the Gold-Codes for each satellite, $\vec{s}_i$?

   Meta: Note that τ_i is an integer which represents the time shift for the discrete signal $\vec{s}_i$. Therefore to compute the distance d_i in meters, the time delay τ_i must be converted into seconds. This is out of scope, but can be done based on the sampling rate of the receiver, and should only be addressed if students ask about it.

   Solution: We can solve for the time delay τ_i by computing the cross-correlation between $\vec{r}$ and $\vec{s}_i$.

$$\text{corr}_{\vec{r}}(\vec{s}_i)[k] = \sum_{n=-\infty}^{\infty} \vec{r}[n]\vec{s}_i[n-k]$$

   Remember that the cross-correlation between two vectors $\vec{r}, \vec{s}_i$ measures the similarity between $\vec{r}$ with all shifts of $\vec{s}_i$. Therefore, we pick the index τ_i as the index of greatest cross-correlation between $\vec{r}$ and $\vec{s}_i$. The distance d_i can be computed by multiplying the time delay by the

3. How can we express d_i in terms of $\vec{p}_i$ and $\vec{x}$? How many such equations can we set up in total?

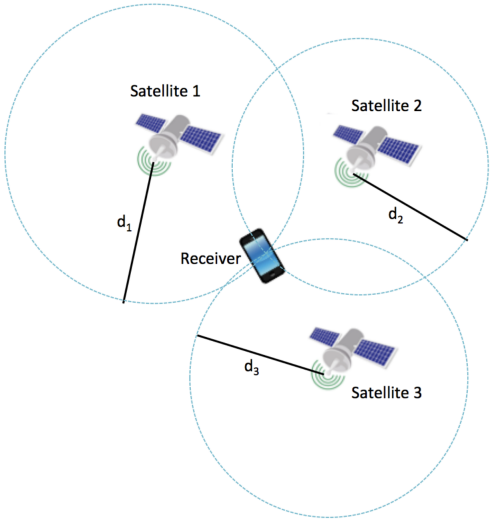
   Solution:

4. Recall that an APS can help us determine where we are. Which variable given in the question corresponds to our current location?

5. Based on the system of equations we set up in part 2, how can we solve for our current position?

6. If our current position can be represented by an n-dimensional vector, how many satellites do we need to be able to solve for our position?

7. As shown below geometrically, we can represent the area of coverage by each satellite as a circle with a radius of d_i. Explain why the radius of each circle is d_i, and how finding our current position is equivalent to finding the point of intersection among the circumferences of the circles.



8. Now that we have figured out where we are, it is time for us to decode the message! Recall that what our phone receives is a linear combination of these transmitted signals:

$$\vec{r} = \alpha_1 \vec{s}_1^{\,(\tau_1)} + \alpha_2 \vec{s}_2^{\,(\tau_2)} + \ldots + \alpha_m \vec{s}_m^{\,(\tau_m)}$$

From the expression above, which of the following variables are we trying to solve for?

- The scaling (attenuating) constant α_i
- The original signal sent by the satellite $\vec{s}_i$
- The delay in the transmission of the signal τ_i

9. To solve for the unknown variables from the previous part, we can use the **least squares** method. How can we reformulate the given expression and information above as a **least squares** problem? In other words, if we are to rewrite the problem in the form:

$$A\vec{v} \approx \vec{b},$$

what would A, $\vec{v}$, and $\vec{b}$ be equal to respectively?

10. What is the solution using the **least squares** method?

11. Does there exist a case where least squares would not work? Write down a sufficient condition where we have to resort to some other techniques to recover the original message.

12. If we are to solve this problem using Orthogonal Matching Pursuit (OMP) instead of least squares, what is one assumption made by this technique?

13. Describe what the stopping conditions for OMP are. In other words, how do we know when to stop running the OMP algorithm?