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

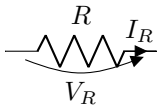
# Worksheet #13 Metas

*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.

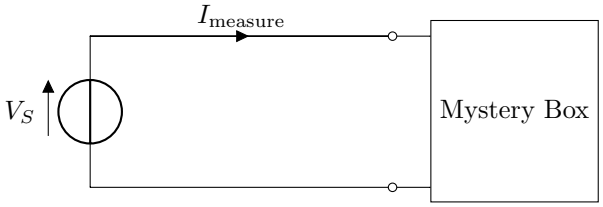
1. Write Ohm's Law for a resistor.




**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 $I_{\text{measure}}$?




**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 # | $I_{\text{measure}}$ | $V_S$ |
|---|---|---|
| 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, $R_{Th} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\vec{b} = 0.975\,\text{k}\Omega$

## 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 may 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 does the $i$th column of $A$ 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, the $i$th column of $A$ is the signal from the $i$th device.

   - 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 $i$th 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 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 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 $\vec{s}_i$ and $\vec{s}_j$, 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
- $< \vec{s}_i, \vec{s}_j > \approx 0$, 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. $\vec{s}_i$ for which $x_i$ is 0.

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

$$\vec{r} = \alpha_1 \vec{s}_1 + \cdots + \alpha_m \vec{s}_m$$

where $\vec{r}$ is the received signal, and each $\vec{s}_i$ is a beacon signal scaled by a factor of $\alpha_i$.

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,

$$\vec{r} = \alpha_1 \vec{s}_1^{(\tau_1)} + \cdots + \alpha_m \vec{s}_m^{(\tau_m)}$$

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

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

**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 shifted versions of themselves, i.e.

$$< \vec{s}_i^{(\tau_i)}, \vec{s}_j^{(\tau_j)} > \approx 0$$

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 $\vec{r}$, and $\vec{x}_{curr}$ to be our guesses for the weights of these beacons within $\vec{r}$.
- We assume there to be at most $k$ beacons that are broadcasting.
- While the magnitude of the difference between our guess for $\vec{r}$ and the actual value of $\vec{r}$ 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 $\vec{r}$ and the actual value of $\vec{r}$. 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

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

(b) Given $A_{curr}$ and $\vec{x}_{curr}$, how do we find the strongest remaining shifted signal $\vec{s}_i^{(\tau_i)}$? *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 $\vec{s}_i$. The correlation with the largest value tells us which $\vec{s}_i$ 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 $\vec{r}$, 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 $\vec{x}$.

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

Let's walk through the algorithm with some example values now. Suppose we have the following:

- $k = 2$ maximum number of signals
- beacon signals $\vec{s}_1 = \begin{bmatrix} -0.5 \\ 0.5 \\ 0.1 \end{bmatrix}$, $\vec{s}_2 = \begin{bmatrix} 0.8 \\ 0.6 \\ 0.7 \end{bmatrix}$, and $\vec{s}_3 = \begin{bmatrix} 0 \\ -0.6 \\ 0 \end{bmatrix}$
- received signal $\vec{r} = \begin{bmatrix} -1.8 \\ 1.5 \\ 0.3 \end{bmatrix}$

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

**Solution:** Initially, we've found 0 beacon vectors, so the entire received signal $\vec{r}$ is still unaccounted for.

8. Next, find the strongest unaccounted for beacon within $\vec{e}$.

**Solution:** Per the previous part, we initialize $\vec{e} = \vec{r}$. We examine the dot products of $\vec{e}$ with all shifted versions of our signal vectors.

- $< \vec{e}, \vec{s}_1^{(0)} > = 1.68$
- $< \vec{e}, \vec{s}_1^{(1)} > =$ -0.9
- $< \vec{e}, \vec{s}_1^{(2)} > =$ -0.78
- $< \vec{e}, \vec{s}_2^{(0)} > =$ -0.33
- $< \vec{e}, \vec{s}_2^{(1)} > = 0.21$
- $< \vec{e}, \vec{s}_2^{(2)} > = 0.12$
- $< \vec{e}, \vec{s}_3^{(0)} > =$ -0.9

- $< \vec{e}, \vec{s}_3^{(1)} > = 1.08$
- $< \vec{e}, \vec{s}_3^{(2)} > =$ -0.18

The largest magnitude is in the inner product between $\vec{e}$ and $\vec{s}_1^{(0)}$. This tells us that we $\vec{s}_1^{(0)}$ is the strongest beacon.

9. Estimate our new value of $\vec{x}_{curr}$.

**Solution:** We perform least squares with $A_{curr} = \begin{bmatrix} | \\ \vec{s}_1^{(0)} \\ | \end{bmatrix}$ to give us

$$\vec{x}_{curr} = (\begin{bmatrix} -0.5 & 0.5 & 0.1 \end{bmatrix} \begin{bmatrix} -0.5 \\ 0.5 \\ 0.1 \end{bmatrix})^{-1} \begin{bmatrix} -0.5 & 0.5 & 0.1 \end{bmatrix} \begin{bmatrix} -1.8 \\ 1.5 \\ 0.3 \end{bmatrix} \approx \begin{bmatrix} 3.294 \end{bmatrix}$$

We're slightly off the mark (since the actual value is 3), but this is pretty close, and will improve as we improve our estimate.

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

**Solution:**

$$\begin{aligned} \vec{e} &= \vec{r} - A_{curr}\vec{x}_{curr} \\ &= \begin{bmatrix} -1.8 \\ 1.5 \\ 0.3 \end{bmatrix} - \begin{bmatrix} -0.5 \\ 0.5 \\ 0.1 \end{bmatrix} \begin{bmatrix} 3.294 \end{bmatrix} \\ &\approx \begin{bmatrix} -0.152 \\ -0.147 \\ -0.029 \end{bmatrix} \end{aligned}$$

11. Find the next strongest unaccounted beacon.

**Solution:** We cross-correlate with the remaining vectors as follows.

- $< \vec{e}, \vec{s}_2^{(0)} > =$ -0.231
- $< \vec{e}, \vec{s}_2^{(1)} > =$ -0.218
- $< \vec{e}, \vec{s}_2^{(2)} > =$ -0.242
- $< \vec{e}, \vec{s}_3^{(0)} > = 0.088$
- $< \vec{e}, \vec{s}_3^{(1)} > = 0.092$
- $< \vec{e}, \vec{s}_3^{(2)} > = 0.018$

The inner product of $\vec{e}$ with $\vec{s}_2^{(2)}$ has the largest magnitude.

**Meta:** We take the inner product with the largest *magnitude* rather than the largest value to account for the possibility of negative scaling.

12. Once more, estimate our new value of $\vec{x}_{curr}$.

**Solution:** We do least squares again, this time with $A = \begin{bmatrix} | & | \\ \vec{s}_1^{(0)} & \vec{s}_3^{(2)} \\ | & | \end{bmatrix}$.

$$\vec{x}_{curr} = (\begin{bmatrix} -0.5 & 0.5 & 0.1 \\ 0.7 & 0.8 & 0.6 \end{bmatrix} \begin{bmatrix} -0.5 & 0.7 \\ 0.5 & 0.8 \\ 0.1 & 0.6 \end{bmatrix})^{-1} \begin{bmatrix} -0.5 & 0.5 & 0.1 \\ 0.7 & 0.8 & 0.6 \end{bmatrix} \begin{bmatrix} -1.8 \\ 1.5 \\ 0.3 \end{bmatrix} = \begin{bmatrix} 3.178 \\ 0.102 \end{bmatrix}$$

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

13. Express $\vec{r}$ as a linear combination of shifted versions of $\vec{s}_1$, $\vec{s}_2$, and $\vec{s}_3$. What is our final error vector $\vec{e}$?

**Solution:** The result of OMP tells us that $\vec{r} = 3.178\vec{s}_1^{(0)} + 0.102\vec{s}_2^{(2)}$. Our error vector $\vec{r} - A_{curr}\vec{x}_{curr}$ turns out to be $\begin{bmatrix} -0.282 \\ -0.170 \\ 0.556 \end{bmatrix}$.

**Meta:** If your students have some difficulty understanding this part, remind them of the fact that a matrix-vector multiplication is a linear combination of the columns of the matrix, i.e.

$$\begin{bmatrix} | & & | \\ \vec{a}_1 & \dots & \vec{a}_n \\ | & & | \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = x_1\vec{a}_1 + \dots x_n\vec{a}_n$$

**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.

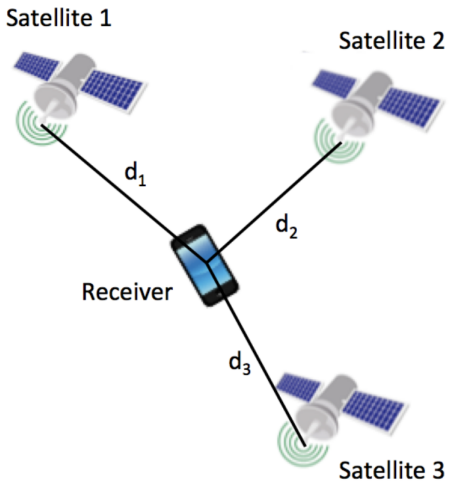
You receive a linear combination of these transmitted signals, each delayed by $(\tau_i)$:

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

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 $d_i$

   **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 $\vec{x}$.

2. We will start by solving for the distances $d_i$ and the time delays $\tau_i$. 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 $\tau_i$ 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 $\tau_i$ 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 $\tau_i$ 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 $\tau_i$ 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 speed of light.

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?

   **Meta:** Mentors, it might be helpful to draw out or point to the APS illustration to show why it is natural to label distances as the norms of the differences in position.

   **Solution:** It can be observed that the norm of the difference between $\vec{x}$ and $\vec{p}_i$ will indeed be the distance $d_i$ from $\vec{x}$ to $\vec{p}_i$. Therefore for each satellite, we can write out the equations:

$$\|\vec{x} - \vec{p}_1\| = d_1$$
$$\vdots$$
$$\|\vec{x} - \vec{p}_m\| = d_m$$

   There will be a total of $m$ equations since we get one equation for each satellite.

4. Recall that an APS can help us determine where we are. Which variable given in the question corresponds to our current location? Also, based on the system of equations we set up in part 3, how can we solve for our current position?

   **Solution:** The vector $\vec{x}$ corresponds to our current location. As seen in the first and second parts, we know the location of each satellite $\vec{p}_i$ and we have also solved for the distances $d_i$. Therefore it remains to solve the system of equations we created in the previous part.

   We will first take a look at satellite 1 and then generalize for the rest. First we can square both sides and expand out the norms:

$$\begin{aligned}\|\vec{x} - \vec{p}_1\|^2 &= \langle \vec{x} - \vec{p}_1, \vec{x} - \vec{p}_1 \rangle = (\vec{x} - \vec{p}_1)^T(\vec{x} - \vec{p}_1) \\ &= \vec{x}^T\vec{x} - \vec{p}_1^T\vec{x} - \vec{x}^T\vec{p}_1 + \vec{p}_1^T\vec{p}_1 = \vec{x}^T\vec{x} - 2\vec{p}_1^T\vec{x} + \vec{p}_1^T\vec{p}_1 = d_1^2\end{aligned}$$

   We can then do this for any satellite $i$ to say that

$$\|\vec{x} - \vec{p}_1\|^2 = \vec{x}^T\vec{x} - 2\vec{p}_1^T\vec{x} + d_1^2$$
$$\vdots$$
$$\|\vec{x} - \vec{p}_m\| = \vec{x}^T\vec{x} - 2\vec{p}_m^T\vec{x} + d_m^2$$

While we have a system of equations that we could try solving, the issue is the quadratic term $\vec{x}^T\vec{x}$. Therefore, we will get rid of this quadratic term by subtracting every equation from the first equation to get

$$2\vec{p_1}^T\vec{x} - 2\vec{p_2}^T\vec{x} = d_2^2 - d_1^2$$
$$\vdots$$
$$2\vec{p_1}^T\vec{x} - 2\vec{p_m}^T\vec{x} = d_m^2 - d_1^2$$

We can now set up the following matrix-vector equation:

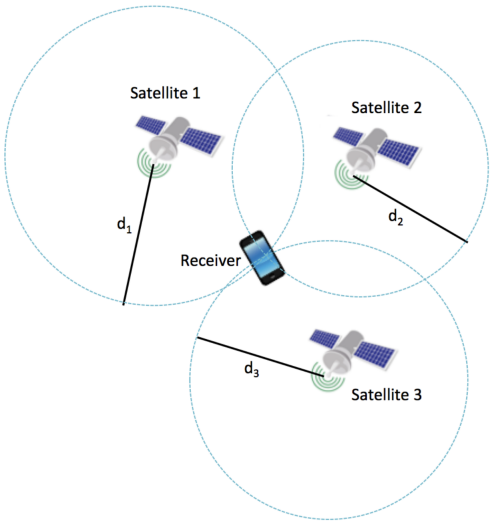
$$\begin{bmatrix} 2(\vec{p_1}^T - \vec{p_2}^T) \\ \vdots \\ 2(\vec{p_1}^T\vec{x} - \vec{p_m}^T) \end{bmatrix} \vec{x} = \begin{bmatrix} d_2^2 - d_1^2 \\ \vdots \\ d_m^2 - d_1^2 \end{bmatrix}$$

$\vec{x}$ can be solved either through Gaussian-Elimination, Inversion, or Least Squares depending on the shape of the matrix.

5. 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?

   **Solution:** If the position $\vec{x}$ is an $n$-dimensional vector, the system of equations will have $n$ unknowns. This means we need $n$ linear equations to solve this system uniquely. Each satellite will bring in one equation, but remember that these were nonlinear, meaning we had to subtract an equation out to get rid of the quadratic term. Therefore we will need $n+1$ equations total, or $n+1$ satellite measurements.

6. 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.




7. 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 $\alpha_i$
- The original signal sent by the satellite $\vec{s_i}$

- The delay in the transmission of the signal $\tau_i$

**Solution:**

- The scaling (attenuating) constant $\alpha_i$ : 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 $\alpha_i$ are the weights of this linear combination.
- The original signal sent by the satellite $\vec{s}_i$ Remember that these are **known** variables that we as the engineer have programmed, hopefully using Gold Codes. No need to solve for these!
- The delay in the transmission of the signal $\tau_i$ While these were initially **unknown** variables, we solved for these in the second part using cross-correlation, so need to solve for these either!

8. 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?

   **Solution:** The model for the received signal accounting for random noise $\vec{n}$ is:
$$\vec{r} = \alpha_1 \vec{s}_1^{\,(\tau_1)} + \alpha_2 \vec{s}_2^{\,(\tau_2)} + \ldots + \alpha_m \vec{s}_m^{\,(\tau_m)} + \vec{n}$$
   We can write this as the following matrix-vector equation:
$$\vec{r} = \begin{bmatrix} | & | & | \\ \vec{s}_1^{\,(\tau_1)} & \cdots & \vec{s}_m^{\,(\tau_m)} \\ | & | & | \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_m \end{bmatrix} + \vec{n}$$
   This can be formulated as a Least Squares problem with $A = \begin{bmatrix} | & | & | \\ \vec{s}_1^{\,(\tau_1)} & \cdots & \vec{s}_m^{\,(\tau_m)} \\ | & | & | \end{bmatrix}$, $\vec{v} = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_m \end{bmatrix}$, and $\vec{b} = \vec{r}$.
   Note that $\vec{b}$ will not be in the column span of $A$ due to the noise, so we cannot solve this matrix-vector equation using Gaussian-Elimination. We also cannot use matrix inversion, since we have made no assumptions of $A$ being square.

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

   **Solution:** The least-squares solution will be $\vec{\tilde{v}} = (A^T A)^{-1} A^T \vec{b}$.

10. 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.

    **Meta:** The null-space proof was shown in lecture, but if students ask about it, you may want to go over it one more time. Take a look at lecture, or look up the proof in case they ask for it.

    **Solution:** Least-Squares will fail when $A^T A$ is not an invertible matrix. It can in fact be shown that $\text{Nul}(A) = \text{Nul}(A^T A)$ meaning the two subspaces will have the same dimensions. If $A$ is a $p \times m$ matrix, then $A^T A$ is a $m \times m$ matrix. By the Rank-Nullity Theorem, dim Col $A^T A$ + dim Nul $A^T A = m$ so we want dim Nul $A^T A$ to be zero for $A^T A$ to be invertible. Therefore, since dim Col $A$ + dim Nul $A = p$, and Nul$A$ = Nul $A^T A$, it must be that dim Col $A = p$. We conclude by saying that the matrix $A$ must be full-rank in order for Least-Squares to have a unique solution.

11. Now let's consider the case in which we have a large number of satellites or ($m >> n$). Why can we no longer use Least-Squares anymore?

    **Solution:** If $m >> n$, then the $A$ matrix will have more columns than rows, and will no longer be full rank since it will have linearly dependent columns. Therefore the matrix $A$ can only be at more rank $p$ meaning $A^T A$ will also only have at most rank $p < m$ meaning it will not be invertible.

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

    **Meta:** If students ask about what to do when the sparsity assumption isn't made, tell them that it is out of scope for 16A, but they will learn how to handle this case in 16B.

    **Solution:** In Orthogonal Matching Pursuit, we make the crucial assumption that the vector $\vec{v}$ is sparse. This means that most of the satellites will not be transmitting and most of the $\alpha$ coefficients will be zero. For example, if we have a total of $10,000$ satellites in space, then we make the assumption that only a small number, $(ex: 50)$ are transmitting a message. This is why we use a greedy, iterative algorithm to solve for the messages.

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

    **Solution:** In each step of OMP, we try to explain the data using our selected satellites $\vec{s}_i$ with the message $\alpha_i$. We then create an estimate $\vec{\tilde{r}} = \sum_{l=1}^{k} \alpha_l \vec{s}_l^{\,(\tau_l)}$ using the information we have. The stopping condition of OMP will be when the residual $\vec{e} = \vec{r} - \vec{\tilde{r}}$ is smaller than some threshold for us to say that we have explained the received signal $\vec{r}$ to the best of our ability.