EECS 16A Designing Information Devices and Systems I
Spring 2018 Lecture Notes Note 24

24.1 Trilateration with multiple beacons

Now that you have learned about cross-correlation and least squares techniques, we can put these ideas together! Imagine you have a positioning system with a single receiver and many different beacons. The beacons (transmitters) could be things like temperature sensors, light-meters, humidity sensors, or something that measures how loudly your cat is purring. These all want to send information to the receiver, but how will the receiver distinguish between the different signals? Recall from the correlation note and locationing lab that these beacons work best when each one has a different *signature code* - let's call them 'songs' - and when these songs are (mostly) orthogonal to one another and (mostly) orthogonal to all the shifted versions of itself. In other words, they have high auto-correlation values and low cross-correlation values with other codes. The songs are streams of ± 1 and are known by the transmitters and the receiver. The songs are multiplied with a *message signal* to get the transmission signal. In the case of a temperature sensor, fro example, this message signal would be some real number corresponding to the temperature.

Now imagine we have 2,000 transmitters, each with it's own song: $\vec{S_0}, \vec{S_1}, ..., \vec{S_{1999}}$

Each song has a length of 400 samples, and therefore the circulant matrix for each song has 400 rows, each corresponding to a delay or shift of 0-399.

$$C_{\vec{S_1}} = \begin{bmatrix} \overrightarrow{S_1^{(0)T}} \\ \overrightarrow{S_1^{(1)T}} \\ \vdots \\ \overrightarrow{S_1^{(399)T}} \end{bmatrix}$$

Note: for the following notation $\overrightarrow{S_1^{(0)}}$, the subscript is the index of the song (transmitter), and the superscript in parentheses is the index of the shift.

Each message is a real number that scales the song: $a_0, a_1, ..., a_{1999}$, for each of the 2,000 sensors.

Each of the messages from the transmitters is delayed by a different amount, given by: $N0, N1, ..., N1999$.

Therefore- the signal received ($\vec{r}$) is a linear combination of shifted songs from all transmitting users given by:

$$\vec{r} = a_0\overrightarrow{S_0^{(N0)}} + a_1\overrightarrow{S_1^{(N1)}} + ... + a_{1999}\overrightarrow{S_{1999}^{(N1999)}}$$

In the simplest case, there will be only one sensor transmitting data to the receiver, for example. If sensor 130 is sending a message of value 1 with a delay of 10, the received signal will be:

$$\vec{r} = 1 * \overrightarrow{S_{130}^{(10)}}$$

Now if we perform cross-correlation of the received signal with the song from sensor 130, $\overrightarrow{S_{130}}$, the result will have a peak at shift index 10, with a normalized amplitude of 1. This is shown in Fig. 1.

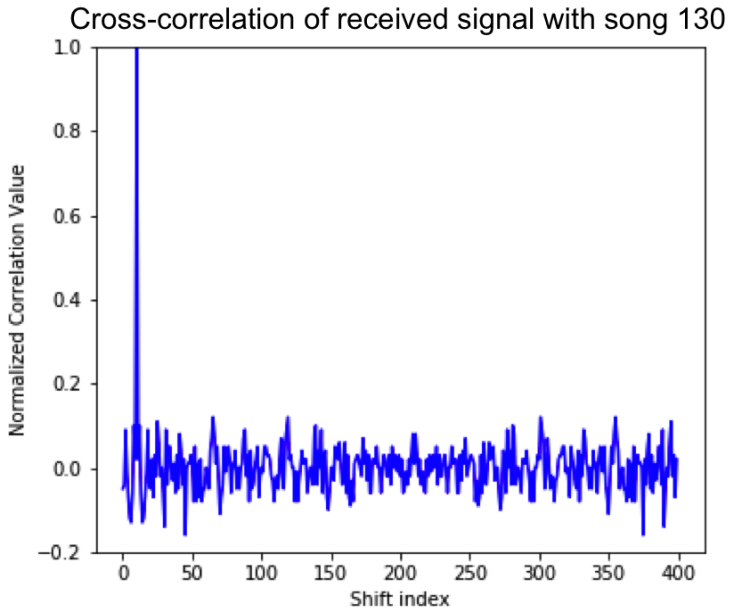


Figure 1: Correlation of received signal with circular shifts of $\vec{S_{130}}$, showing a peak at the true shift index of 10

If we correlate with all songs and plot the maximum from each one, we get the results in Fig. 2. there is a single peak at the sensor ID 130, telling us that sensor 130 was"on" and sending a value of 1.

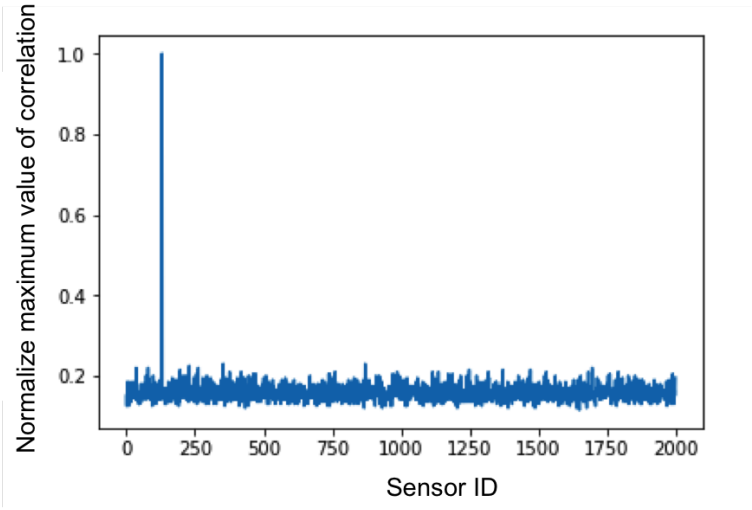


Figure 2: Correlation of received signal with circular shifts of all the songs. This is only on peak, at a shift of 10 for song 130.

What happens when multiple transmitters are "on"?

Imagine devices 40, 100, and 312 are simultaneously transmitting with delays of 13, 20, and 45 (respectively) and message values of 10, 10, and 8 (respectively). Our received signal is now:

$$\vec{r} = 10\overrightarrow{S_{40}^{(13)}} + 10\overrightarrow{S_{100}^{(20)}} + 8\overrightarrow{S_{312}^{(45)}}$$

If we cross-correlate the received signal with every song, find the maximum peak of each cross-correlation, and plot it for each sensor ID (normalizing by the length of the song), we get the plot in Fig. 3.

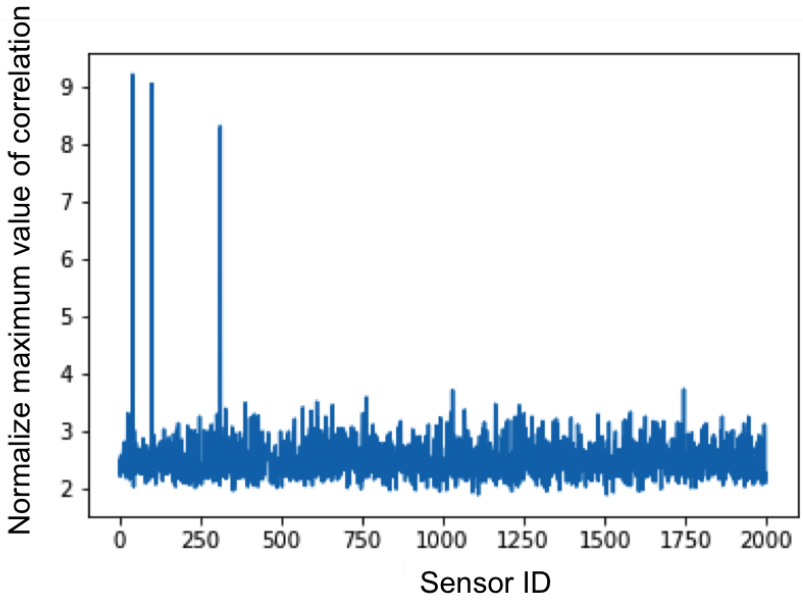


Figure 3: Maximum value of correlation of received signal with circular shifts of all the songs.

Let us look at one more example in which 4 users (user 40, 100, 312 and 350) transmit simultaneously and their corresponding messages are 100, 10, 8 and 0.02. The result of the correlation with different signatures is shown in Fig. 4a.

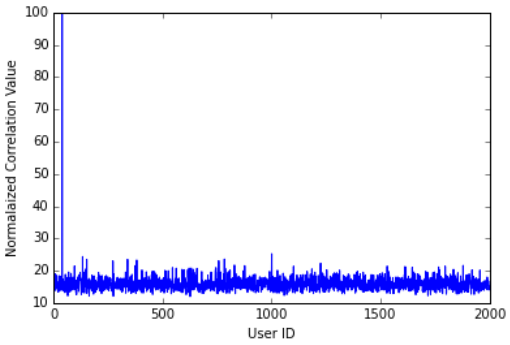
The peaks corresponding to sensor 100, 312 and 350 don't seem to appear at all! Even if we zoom in, we can't distinguish peaks corresponding to these sensors. What could possibly cause this? Perhaps the very high value of user 40's made it so that we cannot see the songs from users 100, 312 and 350.

Maybe if we are able to identify the dominant song and subtract it from the received signal and cross-correlate again, we will be able to see the other songs. We can see that the dominant song here is $\overrightarrow{S_{40}}$ with message value 100. We can remove the dominant song from our received signal $\vec{r}$:

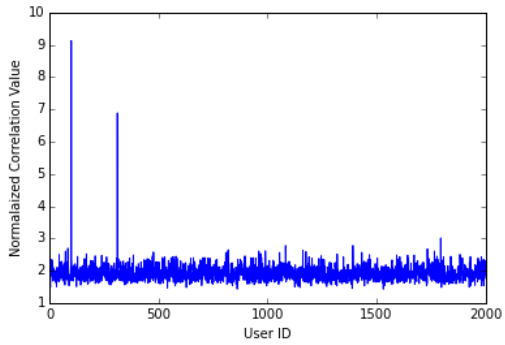
$$\vec{y} = \vec{r} - 100\overrightarrow{S_{40}^{(13)}}$$

Here, $\vec{y}$ is the *residual*, the signal that is left after removing the dominant song. To try to find the weaker signals, we calculate the correlation between the residual with all of the songs, shown in Fig. 4b

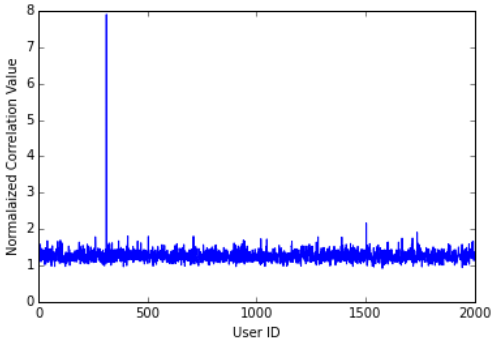
We now see the peaks corresponding to users 100 and 312, but not 350. If we repeat the same steps as before, removing the dominant song (that of sensor 100) and correlating again, we see the results of the subsequent steps in Fig. 4c. Then we do the same after removing sensor 312, and we show the result in Fig. 4d.



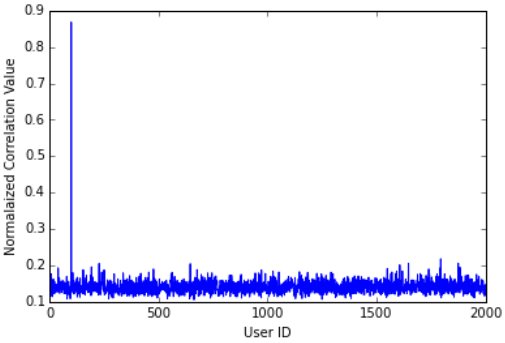
(a) Step 1: Correlation of signatures with the received signal



(b) Step 2: Correlation with residue after removal of sensor 40's song



(c) Step 3: Correlation with residue after removal of user 100's song



(d) Step 4: Correlation with residue after removal of user 312's song

Figure 4: Maximum correlation of received signal with circular shifts of all the signatures

In Fig. 4d we would have expected to find a peak at user 350 but surprisingly we don't. Instead, we see a peak at user 100. Why is that the case? Maybe, we didn't completely remove the other signals. We need a new technique!

24.2 Orthogonal Matching Pursuit

There is a better way to determine the values of all the songs in the received signal via an algorithm called Orthogonal Matching Pursuit (OMP).

Recall again that the received signal looks something like this:

$$\vec{r} = a_0\overrightarrow{S_0^{(N0)}} + a_1\overrightarrow{S_1^{(N1)}} + ... + a_{1999}\overrightarrow{S_{1999}^{(N1999)}}$$

and that cross-correlation is good at giving us the ID and shift of the songs we are hearing, but not the actual value, ie we can determine $\overrightarrow{S_1^{(N1)}}$ well, but not a_1 well.

Note that this technique works best for a sparse spectrum, meaning that the number of devices transmitting at any given time is much less than the total number of devices. In our case, we have 2000 devices, but let's operate under the assumption that at most $k = 10$ at a time are singing their songs.

What if we re-write this equation in a form that looks like $A\vec{x} = \vec{b}$ and use least squares to solve for the song values?

$$A \qquad\qquad \vec{x} \quad = \vec{b}$$

$$\begin{bmatrix} | & | & & | \\ \overrightarrow{S_0^{(N0)}} & \overrightarrow{S_1^{(N1)}} & \cdots & \overrightarrow{S_k^{(Nk)}} \\ | & | & & | \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_k \end{bmatrix} = \vec{r}$$

Recall that the least squares solution is:

$$\vec{x} = (A^TA)^{-1}A^T\vec{b}$$

We can use least squares to calculate the coefficients $a_0, a_1, ..., a_k$ that correspond to the songs and shift we find with cross-correlation. This is an improvement because least squares is guaranteed to give the best estimate of the coefficients, even with noise.

Here is a description of OMP with an example received signal of : $\vec{r} = a_{40}\overrightarrow{S_{40}^{(13)}} + a_{100}\overrightarrow{S_{100}^{(8)}}$ where $a_{40} = 100$ and $a_{100} = 10$.

1. Find the song with the highest correlation with $\vec{r}$, and find the shift that maximizes the correlation. We'll denote this shifted song as vector $\overrightarrow{S_*^{(N*)}}$, where $*$ indicates the song ID and shift that maximize the cross-correlation. What does it mean for a vector $\overrightarrow{S_*^{(N*)}}$ to have the highest correlation with $\vec{r}$? The highest correlation between two vectors means the error vector between them is the lowest. We find this vector via a cross-correlation with all the circular shifts of all the songs, the same first step we have always done. This gives us $\overrightarrow{S_{40}^{(13)}}$.

2. Use least squares to solve for $\vec{x}$ in the eqn $A\vec{x} = \vec{b}$ where $\vec{b}$ is the received signal $\vec{r}$ and A is $\begin{bmatrix} | \\ \overrightarrow{S_{40}^{(13)}} \\ | \end{bmatrix}$: $\vec{x} = (A^TA)^{-1}A^T\vec{r}$. In this case, $\vec{x} = 100 = a_{40}$. The 'orthogonal projection of the least squares solution onto the subspace spanned by A' (aka our 'ideal message') is given by: $A\vec{x}$.

3. Now find the residual of $\vec{r}$, denoted $\vec{y}$, left over by subtracting our ideal message from the received signal: $\vec{y} = \vec{r} - A\vec{x}$.

4. Repeat the above steps now using the residual $\vec{y}$ instead of the received signal $\vec{r}$. A new correlation between $\vec{y}$ and all of possible songs would find that the next strongest song is: $\overrightarrow{S_{100}^{(8)}}$. We then update our matrix A to be $\begin{bmatrix} | & | \\ \overrightarrow{S_{40}^{(13)}} & \overrightarrow{S_{100}^{(8)}} \\ | & | \end{bmatrix}$ Finally, we solve $\vec{x}$ via least squares. For this step, we use the received signal $\vec{r}$ (not the residual), $\vec{x} = (A^T A)^{-1} A^T \vec{r}$. We find that $\vec{x} = \begin{bmatrix} 100 \\ 10 \end{bmatrix}$. We have now recovered both of our message values!

5. We stop iterating when we have gone $k = 10$ steps, the known the sparsity of the signal, OR until the norm of the residual is below some threshold value, meaning the residual contains only noise.

The following section precisely describes the implementation of OMP.

Setup:
Let the number of possible unique songs (and users) be $m = 2000$ and the unique songs be represented by $\overrightarrow{S_i}$ (each of length $n = 400$). Each signature can potentially carry a message a_i along with it. Then the measurement at the receiver is

$$\vec{y} = a_0 \overrightarrow{S_0^{(N0)}} + a_1 \overrightarrow{S_1^{(N1)}} + ... + a_{1999} \overrightarrow{S_{1999}^{(N1999)}}$$

We will assume that the number of users talking at the same time is very small *i.e.,* there are at most $k = 10$ non-zero a_is. The OMP algorithm is described below.

Inputs:

- A set of m songs, each of length n: $\mathbf{S} = \{\overrightarrow{S_0}, \overrightarrow{S_1}, ..., \overrightarrow{S_{m-1}}\}$
- An n-dimensional received signal vector: $\vec{r}$
- The sparsity level k of the signal
- Some threshold, th. When the norm of the signal is below this value, the signal is assumed to contain only noise.

Outputs

- A set of songs that were identified, F, which will contain at most k elements.
- A vector, $\vec{x}$ containing song messages (a_1, etc.), which will be of length k or less.
- An n-dimensional residual $\vec{y}$

Procedure:

- Initialize the following values: $\vec{y} = \vec{r}$, $j = 1$, $k = 10$, $A = [\ ]$, $F = \{\emptyset\}$
- while (($j \leq k$) & ($\| \vec{r} \| \geq th$)) :
    1. Cross correlate $\vec{y}$ with the shifted versions of all songs. Find the song index, i, and the shifted version of the song, $\overrightarrow{S_i^{(Ni)}}$, with which the received signal has the highest correlation value.
    2. Add i to the set of song indices, F.
    3. Column concatenate matrix A with the correct shifted version of the song: $A = [A \mid \overrightarrow{S_i^{(Ni)}}]$
    4. Use least squares to obtain the message value: $\vec{x} = (A^T A)^{-1} A^T \vec{r}$
    5. Update the residual value $\vec{r}$ by subtracting: $\vec{y} = \vec{r} - A\vec{x}$
    6. Update the counter: $j = j + 1$