EECS 16A Designing Information Devices and Systems I
Summer 2020 Homework 6A

This homework is due Wednesday, August 5, 2020, at 23:59.
Self-grades are due Sunday August 9, 2020, at 23:59.

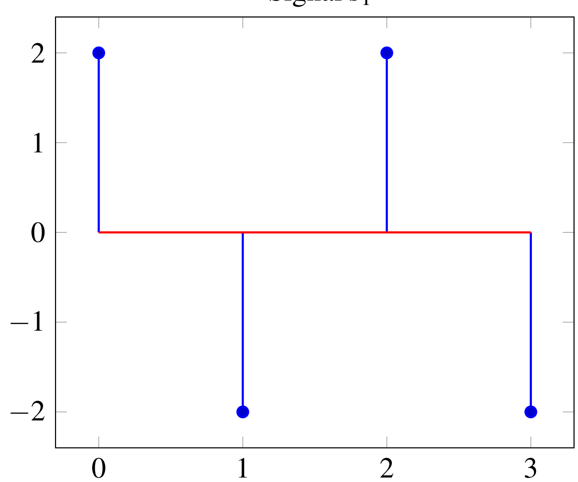
Submission Format
Your homework submission should consist of **one** file.

- `hw6A.pdf`: A single PDF file that contains all of your answers (any handwritten answers should be scanned) as well as your IPython notebook (if any) saved as a PDF.

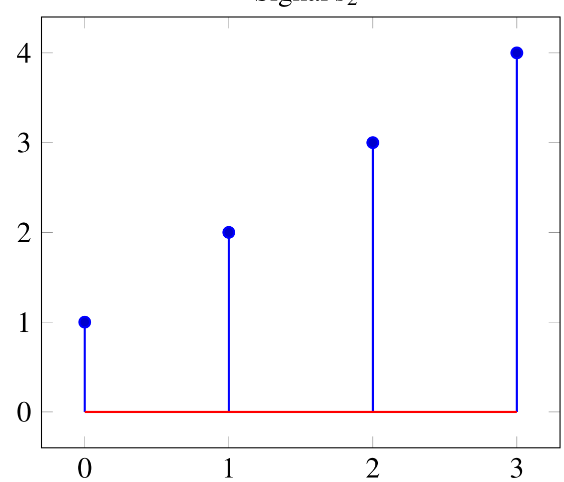
Homework Learning Goals: *The objectives of this homework is to familiarize you with the concept of cross-correlation.*

1. Mechanical Linear Correlation

Signal s_1



Signal s_2



Assume that both of the above signals extend to $\pm\infty$, and are 0 everywhere outside of the region shown in the above graphs. First, we will demonstrate the procedure for linear correlation by computing the linear correlation between signal s_1 with itself (*i.e.* $\text{corr}_{\vec{s}_1}(\vec{s}_1)[k]$). This is referred to as the linear autocorrelation. This can be computed by evaluating the inner product between the signal and the shifted version of the signal (outlined in the below tables). Here, we compute this quantity for shifts between -3 and 3. For all shifts outside this range, the inner product is zero. Finally, we plot the non-zero values of the linear autocorrelation.

$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0		
$\vec{s}_1[n+3]$	2		-2		2		-2		0		0		0		0		0		0		
$\langle \vec{s}_1[n], \vec{s}_1[n+3] \rangle$	0	+	0	+	0	+	-4	+	0	+	0	+	0	+	0	+	0	+	0		= -4

$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_1[n+2]$	0		2		-2		2		-2		0		0		0		0		0	
$\langle \vec{s}_1[n], \vec{s}_1[n+2] \rangle$	0	+	0	+	0	+	4	+	4	+	0	+	0	+	0	+	0	+	0	= 8

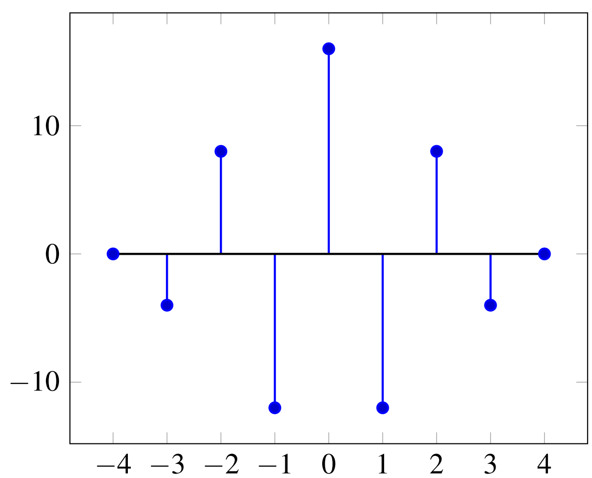
$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_1[n+1]$	0		0		2		-2		2		-2		0		0		0		0	
$\langle \vec{s}_1[n], \vec{s}_1[n+1] \rangle$	0	+	0	+	0	+	-4	+	-4	+	-4	+	0	+	0	+	0	+	0	= -12

$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_1[n+0]$	0		0		0		2		-2		2		-2		0		0		0	
$\langle \vec{s}_1[n], \vec{s}_1[n+0] \rangle$	0	+	0	+	0	+	4	+	4	+	4	+	4	+	0	+	0	+	0	= 16

$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_1[n-1]$	0		0		0		0		2		-2		2		-2		0		0	
$\langle \vec{s}_1[n], \vec{s}_1[n-1] \rangle$	0	+	0	+	0	+	0	+	-4	+	-4	+	-4	+	0	+	0	+	0	= -12

$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_1[n-2]$	0		0		0		0		0		2		-2		2		-2		0	
$\langle \vec{s}_1[n], \vec{s}_1[n-2] \rangle$	0	+	0	+	0	+	0	+	0	+	4	+	4	+	0	+	0	+	0	= 8

$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_1[n-3]$	0		0		0		0		0		0		2		-2		2		-2	
$\langle \vec{s}_1[n], \vec{s}_1[n-3] \rangle$	0	+	0	+	0	+	0	+	0	+	0	+	-4	+	0	+	0	+	0	= -4



(a) Using the procedure demonstrated above, compute $\text{corr}_{\vec{s}_1}(\vec{s}_2)[k]$, the linear cross-correlation of s_2 with s_1. Like the example, use tables like the one given below for $k = -3$ and plot the resulting correlation.

$\vec{s}_1[n]$	0	0	0	2	-2	2	-2	0	0	0
$\vec{s}_2[n+3]$	1	2	3	4	0	0	0	0	0	0
$\langle \vec{s}_1[n], \vec{s}_2[n+3] \rangle$										

Solution:

$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_2[n+3]$	1		2		3		4		0		0		0		0		0		0	
$\langle\vec{s}_1[n],\vec{s}_2[n+3]\rangle$	0	+	0	+	0	+	8	+	0	+	0	+	0	+	0	+	0	+	0	= 8

$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_2[n+2]$	0		1		2		3		4		0		0		0		0		0	
$\langle\vec{s}_1[n],\vec{s}_2[n+2]\rangle$	0	+	0	+	0	+	6	+	-8	+	0	+	0	+	0	+	0	+	0	= -2

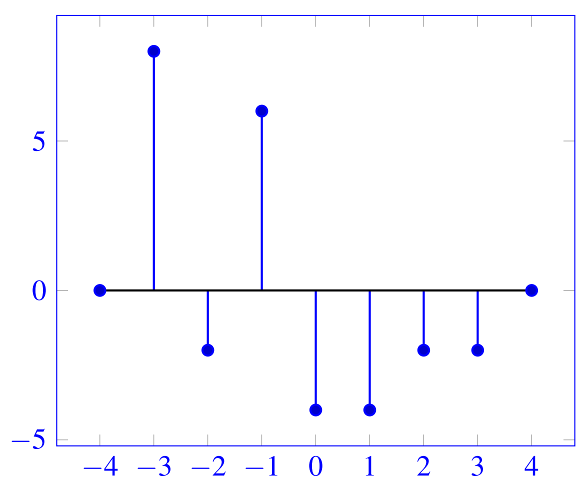
$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_2[n+1]$	0		0		1		2		3		4		0		0		0		0	
$\langle\vec{s}_1[n],\vec{s}_2[n+1]\rangle$	0	+	0	+	0	+	4	+	-6	+	8	+	0	+	0	+	0	+	0	= 6

$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_2[n+0]$	0		0		0		1		2		3		4		0		0		0	
$\langle\vec{s}_1[n],\vec{s}_2[n+0]\rangle$	0	+	0	+	0	+	2	+	-4	+	6	+	-8	+	0	+	0	+	0	= -4

$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_2[n-1]$	0		0		0		0		1		2		3		4		0		0	
$\langle\vec{s}_1[n],\vec{s}_2[n-1]\rangle$	0	+	0	+	0	+	0	+	-2	+	4	+	-6	+	0	+	0	+	0	= -4

$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_2[n-2]$	0		0		0		0		0		1		2		3		4		0	
$\langle\vec{s}_1[n],\vec{s}_2[n-2]\rangle$	0	+	0	+	0	+	0	+	0	+	2	+	-4	+	0	+	0	+	0	= -2

$\vec{s}_1[n]$	0		0		0		2		-2		2		-2		0		0		0	
$\vec{s}_2[n-2]$	0		0		0		0		0		0		1		2		3		4	
$\langle\vec{s}_1[n],\vec{s}_2[n-2]\rangle$	0	+	0	+	0	+	0	+	0	+	0	+	-2	+	0	+	0	+	0	= -2



(b) Will the linear cross-correlation of s_2 with s_1 ($\text{corr}_{\vec{s}_1}(\vec{s}_2)[k]$) be the same as the cross-correlation of s_1 with s_2 ($\text{corr}_{\vec{s}_2}(\vec{s}_1)[k]$)? You can use the iPython notebook to figure this out. How are they related to each other?

Solution: See sol6A.ipynb. They do not have the same result, but they are related: one is the reverse of the other. If you were able to observe this, give yourself full points.

You were not explicitly required to show why, but a sketch of why this is the case follows. Let us compare $\text{corr}_{\vec{s}_2}(\vec{s}_1)[k]$ and $\text{corr}_{\vec{s}_1}(\vec{s}_2)[k]$.
By definition:

$$\text{corr}_{\vec{s}_2}(\vec{s}_1)[k] = \sum_{n=-\infty}^{\infty} \vec{s}_2[n]\vec{s}_1[n-k]$$
$$\text{corr}_{\vec{s}_1}(\vec{s}_2)[k] = \sum_{n=-\infty}^{\infty} \vec{s}_1[n]\vec{s}_2[n-k]$$

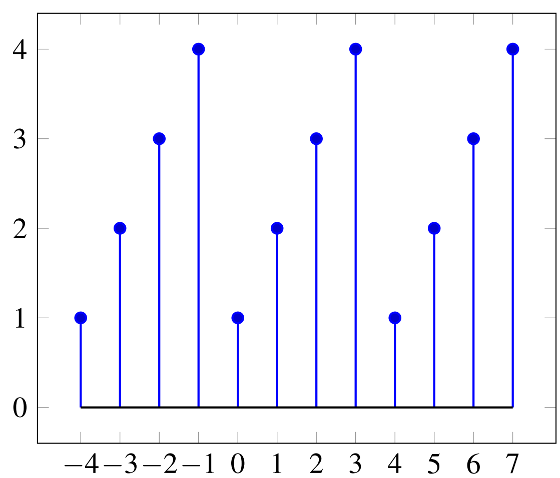
Using a substitution of index, $m = n-k$ we have:

$$\begin{aligned}\text{corr}_{\vec{s}_1}(\vec{s}_2)[k] &= \sum_{m=-\infty}^{\infty} \vec{s}_1[m+k]\vec{s}_2[m] \\ &= \sum_{m=-\infty}^{\infty} \vec{s}_2[m]\vec{s}_1[m-(-k)] \\ &= \text{corr}_{\vec{s}_2}(\vec{s}_1)[-k]\end{aligned}$$

So we can conclude that $\text{corr}_{\vec{s}_1}(\vec{s}_2)[k] = \text{corr}_{\vec{s}_2}(\vec{s}_1)[-k]$.

Now, we will review the procedure to perform linear cross-correlation between one signal that is periodic with a period of 4 and another that is finite length and extended with zeros as in the previous parts. As an example, we will compute the linear correlation $\text{corr}_{\vec{p}_2}(\vec{s}_1)[k]$ between the periodic signal $\vec{p}_2$ (with period 4), formed by repeating $\vec{s}_2$, and the finite length signal $\vec{s}_1$ extended with zeros. The result will be a periodic signal with period 4.

The periodic signal, $\vec{p}_2$, formed by repeating $\vec{s}_2$ is plotted below for indices -4 to 7. It is defined and non-zero for all indices from $-\infty$ to $+\infty$.



We compute one period of the result of the cross-correlation by starting at a shift of $k=-1$ and ending at a shift of $k=2$.

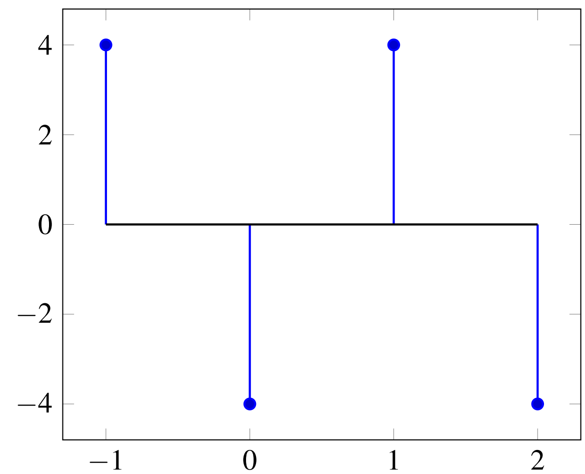
$\vec{p}_2[n]$	2		3		4		1		2		3		4		1		2		3	
$\vec{s}_1[n+1]$	0		0		2		-2		2		-2		0		0		0		0	
$\langle \vec{p}_2[n], \vec{s}_1[n+1] \rangle$	0	+	0	+	8	+	-2	+	4	+	-6	+	0	+	0	+	0	+	0	= 4

$\vec{p}_2[n]$	2		3		4		1		2		3		4		1		2		3	
$\vec{s}_1[n+0]$	0		0		0		2		-2		2		-2		0		0		0	
$\langle \vec{p}_2[n], \vec{s}_1[n+0] \rangle$	0	+	0	+	0	+	2	+	-4	+	6	+	-8	+	0	+	0	+	0	= -4

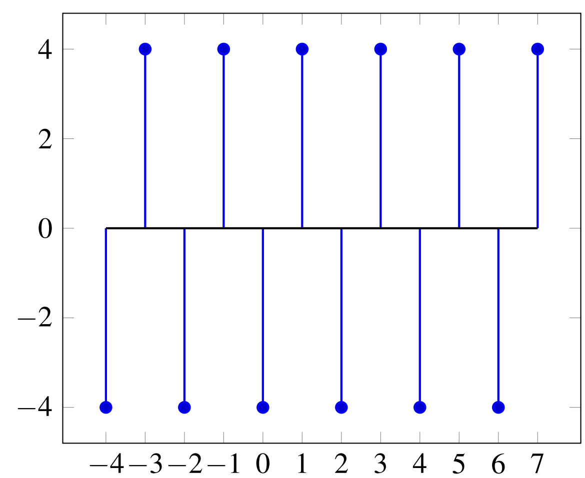
$\vec{p}_2[n]$	2		3		4		1		2		3		4		1		2		3	
$\vec{s}_1[n-1]$	0		0		0		0		2		-2		2		-2		0		0	
$\langle \vec{p}_2[n], \vec{s}_1[n-1] \rangle$	0	+	0	+	0	+	0	+	4	+	-6	+	8	+	-2	+	0	+	0	= 4

$\vec{p}_2[n]$	2		3		4		1		2		3		4		1		2		3	
$\vec{s}_1[n-2]$	0		0		0		0		0		2		-2		2		-2		0	
$\langle \vec{p}_2[n], \vec{s}_1[n-2] \rangle$	0	+	0	+	0	+	0	+	0	+	6	+	-8	+	2	+	-4	+	0	= -4

The computed single period of the resulting linear cross correlation is plotted below.



The resulting linear cross correlation for shifts from $k = -4$ to $k = 7$ is plotted below.



(c) Repeat the procedure described above to compute the correlation $\text{corr}_{\vec{p}_1}(\vec{s}_1)[k]$ between a periodic signal $\vec{p}_1$ (with period 4), formed by repeating s_1, and the finite-length signal s_1 extended with zeros. Like the example, evaluate tables like the one below for $k = -3$ for different shifts and plot a single period of the result.

$\vec{p}_1[n]$	-2	2	-2	2	-2	2	-2	2	-2	2
$\vec{s}_1[n+3]$	2	-2	2	-2	0	0	0	0	0	0
$\langle \vec{p}_1[n], \vec{s}_1[n+3] \rangle$										

Solution: We have computed below shifts from $k = -3$ to $k = 3$. However, so long as you have enough values for a single period, give yourself full credit.

$\vec{p}_1[n]$	-2	2	-2	2	-2	2	-2	2	-2	2	
$\vec{s}_1[n+3]$	2	-2	2	-2	0	0	0	0	0	0	
$\langle \vec{p}_1[n], \vec{s}_1[n+3] \rangle$	-4 +	-4 +	-4 +	-4 +	0 +	0 +	0 +	0 +	0 +	0	= -16

$\vec{p}_1[n]$	-2	2	-2	2	-2	2	-2	2	-2	2	
$\vec{s}_1[n+2]$	0	2	-2	2	-2	0	0	0	0	0	
$\langle \vec{p}_1[n], \vec{s}_1[n+2] \rangle$	0 +	4 +	4 +	4 +	4 +	0 +	0 +	0 +	0 +	0	= 16

$\vec{p}_1[n]$	-2	2	-2	2	-2	2	-2	2	-2	2	
$\vec{s}_1[n+1]$	0	0	2	-2	2	-2	0	0	0	0	
$\langle \vec{p}_1[n], \vec{s}_1[n+1] \rangle$	0 +	0 +	-4 +	-4 +	-4 +	-4 +	0 +	0 +	0 +	0	= -16

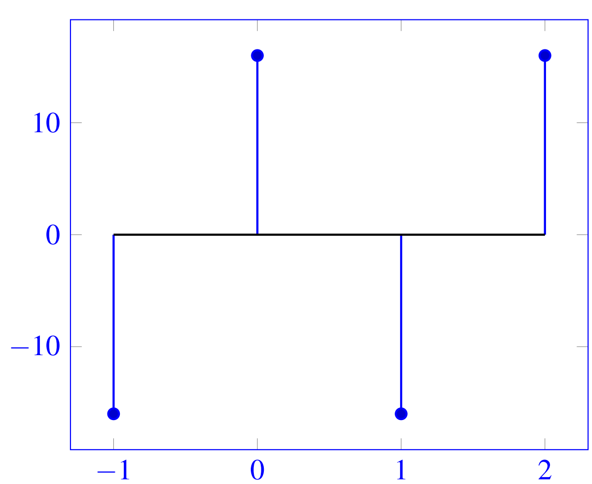
$\vec{p}_1[n]$	-2	2	-2	2	-2	2	-2	2	-2	2	
$\vec{s}_1[n+0]$	0	0	0	2	-2	2	-2	0	0	0	
$\langle \vec{p}_1[n], \vec{s}_1[n+0] \rangle$	0 +	0 +	0 +	4 +	4 +	4 +	4 +	0 +	0 +	0	= 16

$\vec{p}_1[n]$	-2	2	-2	2	-2	2	-2	2	-2	2	
$\vec{s}_1[n-1]$	0	0	0	0	2	-2	2	-2	0	0	
$\langle \vec{p}_1[n], \vec{s}_1[n-1] \rangle$	0 +	0 +	0 +	0 +	-4 +	-4 +	-4 +	-4 +	0 +	0	= -16

$\vec{p}_1[n]$	-2	2	-2	2	-2	2	-2	2	-2	2	
$\vec{s}_1[n-2]$	0	0	0	0	0	2	-2	2	-2	0	
$\langle \vec{p}_1[n], \vec{s}_1[n-2] \rangle$	0 +	0 +	0 +	0 +	0 +	4 +	4 +	4 +	4 +	0	= 16

$\vec{p}_1[n]$	-2	2	-2	2	-2	2	-2	2	-2	2	
$\vec{s}_1[n-3]$	0	0	0	0	0	0	2	-2	2	-2	
$\langle \vec{p}_1[n], \vec{s}_1[n-3] \rangle$	0 +	0 +	0 +	0 +	0 +	0 +	-4 +	-4 +	-4 +	-4	= -16

Like the example, the period was plotted from $k = -1$ to $k = 2$. Give yourself full credit if you plotted four consecutive values sufficient for a single period, i.e. your plot starts from a shift of $k = k_0$ and ends at $k = k_0 + 3$



2. Audio File Matching

Lots of different quantities we interact with every day can be expressed as vectors. For example, an audio clip can be thought of as a vector. Each element of the vector might be a sound pressure value or voltage recorded by a microphone, while the index of the element indicates the time at which this value was recorded. This simple series of numbers can completely capture the sound played by a speaker that we hear with our ears.

Many audio processing application rely on representing audio files as vectors. In this problem we explore how we could possibly use the idea of inner products to build an application like *Shazam*.

Audio files can naturally be represented as vectors. Every component of the vector determines the sound we hear at a given time. We'll also refer to these vectors as the audio *signal*. We will use inner products to determine if a particular audio clip was part of a longer song. The ideas here are similar to the themes of the Acoustic Positioning System in the lab.

For this problem we make the *simplification* that the magnitude of each vector determines the volume and the angle of each vector captures the tune.

Let us consider a very simplified model for an audio signal. At each time, the audio signal can be either -1 or $+1$. A vector of length N makes up the audio file.

(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.

Solution:

The inner product of $\vec{x}_1 = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T$ and $\vec{x}_2 = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T$ is $\vec{x}_1 \cdot \vec{x}_2 = N$. The inner product of $\vec{x}_1 = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T$ and $\vec{x}_2 = \begin{bmatrix} 1 & -1 & 1 & -1 & \cdots & 1 & -1 \end{bmatrix}^T$ is $\vec{x}_1 \cdot \vec{x}_2 = 0$ when the vector length is even. To compare two vectors of length N composed of 1 and -1, we take the inner product of the two vectors, a large inner product means the vectors have a similar direction.

In many circumstances, an inner product with a very large negative value would mean the vectors are very different, but it turns out that humans are unable to perceive the sign of sound, so two sounds vectors $\vec{x}$ and $-\vec{x}$ sound exactly the same. As a result, for this problem we are interested in is the **absolute value** of the dot product, but in many other problems, we will interpret a large negative dot product as very different vectors. Don't take off points in parts (a), (b), or (c) if you didn't mention the absolute value.

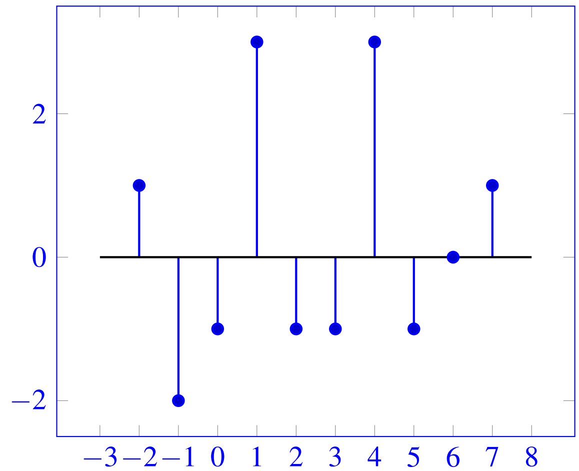
(b) Next, suppose we want to find a short audio clip in a longer one. We might want to do this for an application like *Shazam*, which is able to identify a song from a short clip. Consider the vector of length 8, $\vec{x} = \begin{bmatrix} -1 & 1 & 1 & -1 & 1 & 1 & -1 & 1 \end{bmatrix}^T$. Let us label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

$$\vec{x} := \begin{bmatrix} x[0] & x[1] & x[2] & x[3] & x[4] & x[5] & x[6] & x[7] \end{bmatrix}^T$$

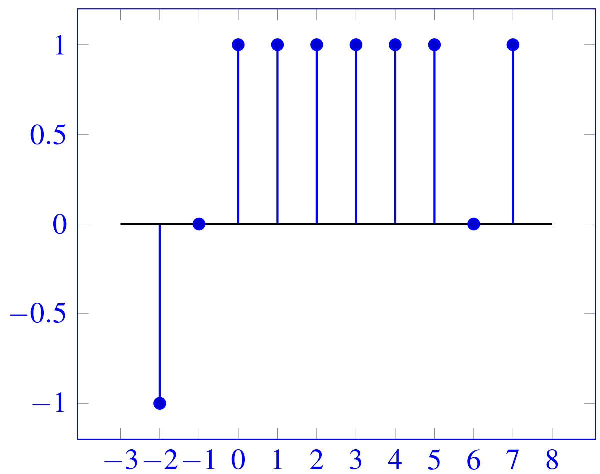
We want to find the short segment $\vec{y} := \begin{bmatrix} y[0] & y[1] & y[2] \end{bmatrix}^T = \begin{bmatrix} 1 & 1 & -1 \end{bmatrix}^T$ in the longer vector. To do this, perform the linear cross correlation between these two finite length sequences and identify at what shift(s) the linear cross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

Solution:



The above plot is $\text{corr}_{\vec{x}}(\vec{y})[k]$ where $\vec{y} = \begin{bmatrix}1 & 1 & -1\end{bmatrix}^T$. At shifts 1 and 4 the cross correlation is its *maximum possible value*, 3. These are both good matches.



The above plot is $\text{corr}_{\vec{x}}(\vec{y})[k]$ where $\vec{y} = \begin{bmatrix}1 & 1 & 1\end{bmatrix}^T$. At shifts 0 through 5 the cross correlation is only 1. There is not a really good match like before.

(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.)

Solution:

Applying the technique in part (b), we get the best match to be $\begin{bmatrix}2 & 3 & 10\end{bmatrix}^T$ as this has the largest dot product with $\vec{y} = \begin{bmatrix}1 & 2 & 3\end{bmatrix}^T$. This is not where we expect to see the peak, as we observe the short audio clip $\vec{y}$ appears at the beginning of the song.

This happens because the volume at the end of the song is louder than the beginning of the song. Despite the angle not matching as well, the louder volume causes the linear cross correlation to be larger.

(d) **(Optional:)** Let us think about how to get around the issue in the previous part. Cross-correlation compares 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?

Solution: Using the normalized cross correlation procedure, the best match for the short audio clip is at the 0^{th} shift and it perfectly matches the clip.

(e) **(Optional:)** We can use this on a more 'realistic' audio signal – see part 2e in the IPython notebook, where we use normalized 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.

Solution:

See `sol6A.ipynb`.

3. Homework Process and Study Group

Who else did you work with on this homework? List names and student ID's. (In case of homework party, you can also just describe the group.) How did you work on this homework?

Solution:

I worked on this homework with...

I first worked by myself for 2 hours, but got stuck on problem 5, so I went to office hours on...

Then I went to homework party for a few hours, where I finished the homework.