EECS 16A Designing Information Devices and Systems I
Spring 2017 Babak Ayazifar, Vladimir Stojanovic Homework 11

This homework is due Monday, April 17, 2017, at 23:59.
Self-grades are due Thursday, April 20, 2017, at 23:59.

Submission Format
Your homework submission should consist of **two** files.

- `hw11.pdf`: A single pdf file that contains all 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 of your IPython notebook, you will not receive credit for problems that involve coding. Make sure your results and plots are showing.

- `hw11.ipynb`: A single IPython notebook with all your code in it.

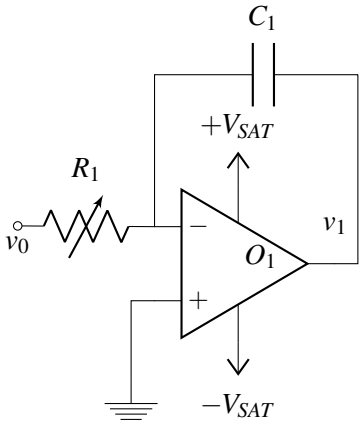
  In order to receive credit for your IPython notebook, you must submit both a "printout" and the code itself.

Submit each file to its respective assignment in Gradescope.

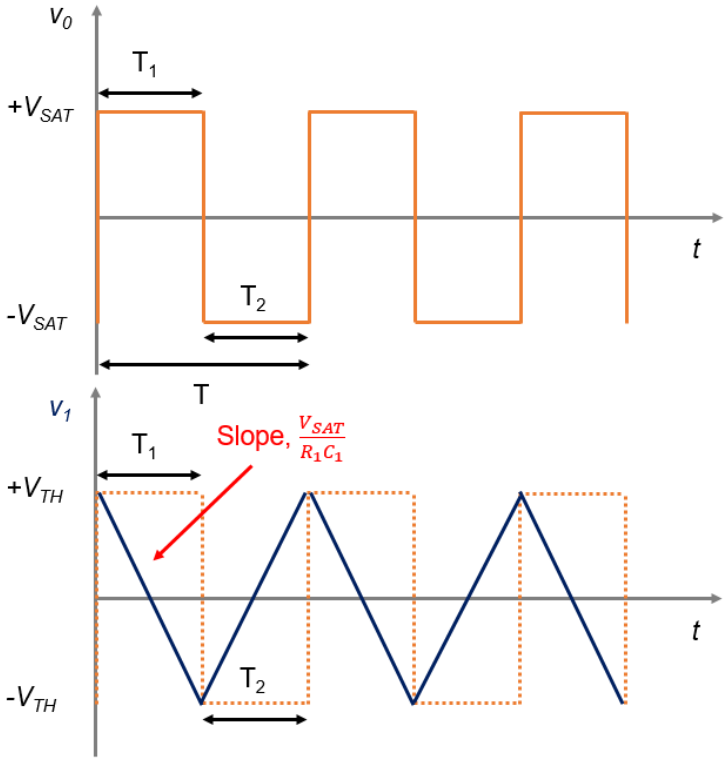
1. Jumpbot

In this problem you will be designing circuits allowing a robot named Jumpbot to execute a set of commands that will be described below. Specifically, the output voltages produced by your circuits are interpreted by Jumpbot as setting its vertical position in meters in free space (both positive and negative values will be used). You will be generating a oscillating triangular waveform with controllable time period.

(a) One of the circuit blocks you will use to generate the triangular waveform is the integrator. An integrator integrates the input signal. For the circuit given below express v_1 in terms of R_1, C_1, and v_0. Hint: you will have to apply KCL, and current flowing through a capacitor is given by $I = C\frac{dV}{dt}$.

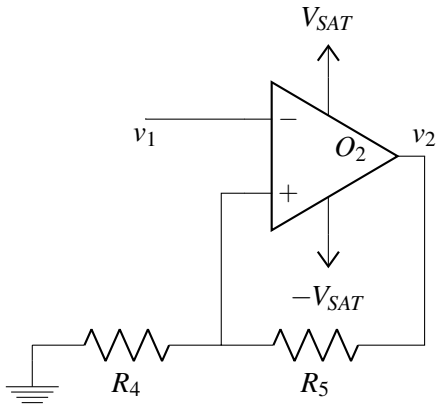


(b) The waveform for v_1 with input v_0 given is below. Here, T is the time period. Derive expressions for T_1 and T_2 as a function of R_1, C_1, V_{TH}, and V_{SAT}. Here, $+V_{TH}$ and $-V_{TH}$ are set using a comparator, we will discuss it in the next part.



(c) Now that we know how to generate a triangular waveform, we can use that for the jumpbot. However, we need to setup the initial signal (v_0) that helped us to create the triangular waveform (v_1). For the circuit below draw the waveform (v_2) if we use v_1 from part (b) as the input. Now, draw the waveform (v_2) if we use -v_1. Which v_2 (v_1 as input or -v_1 as input) matches v_0 from part (a)?

$$+V_{TH} = \frac{R_4}{R_4 + R_5} V_{SAT}$$
$$-V_{TH} = \frac{R_4}{R_4 + R_5} (-V_{SAT})$$

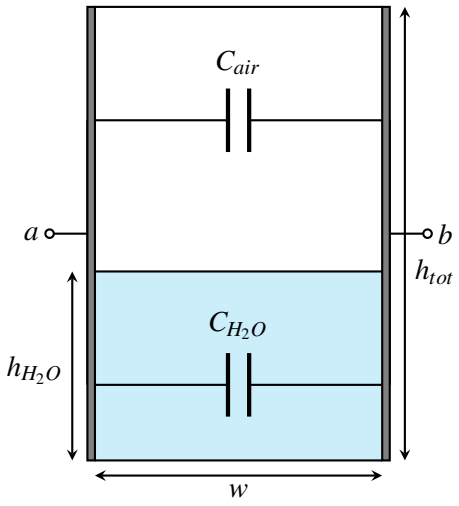


(d) Implement the circuits from part (a) and (c) for controlling the jumpbot using v_1. You may need to insert another circuit block in the middle to get the right polarity at the input terminal of the circuit in (c). Draw out your full circuit.

(e) In your circuit, if $\pm V_{SAT} = \pm 10\,\text{V}$, $C_1 = 0.01\,\mu\text{F}$, $R_4 = 10\,\text{k}\Omega$, find the values for R_1 and R_5, so that the jumpbot jumps with 10 V peak-to-peak amplitude ($\pm V_{TH} = \pm 5\,\text{V}$) with 1 kHz frequency (period = 1 / frequency).

2. Rain Sensor v2.0

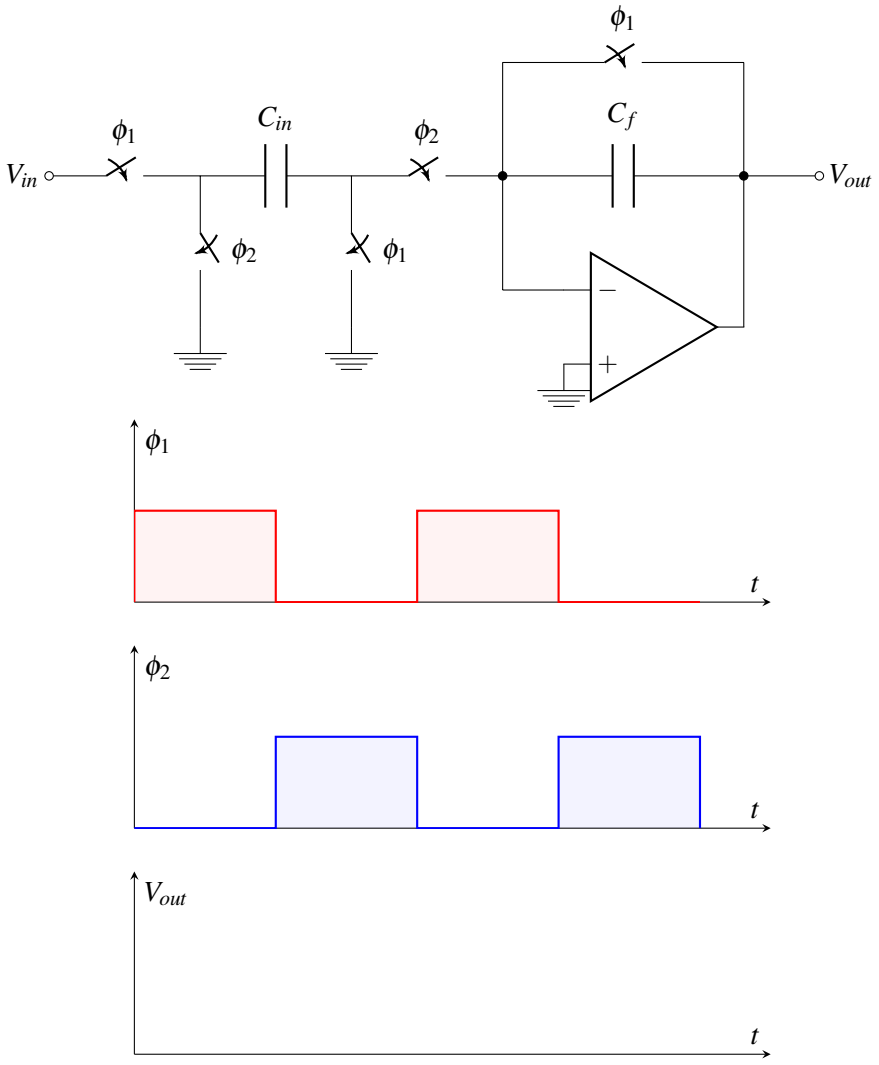
In homework 8, problem 4, we analyzed a rain sensor built by a lettuce farmer in the Salinas Valley. They used a rectangular tank outside and attached two metal plates to two opposite sides in an effort to make a capacitor whose capacitance varies with the amount of water inside. The width and length of the tank are both w (i.e. the base is square), and the height of the tank is h_{tot}.



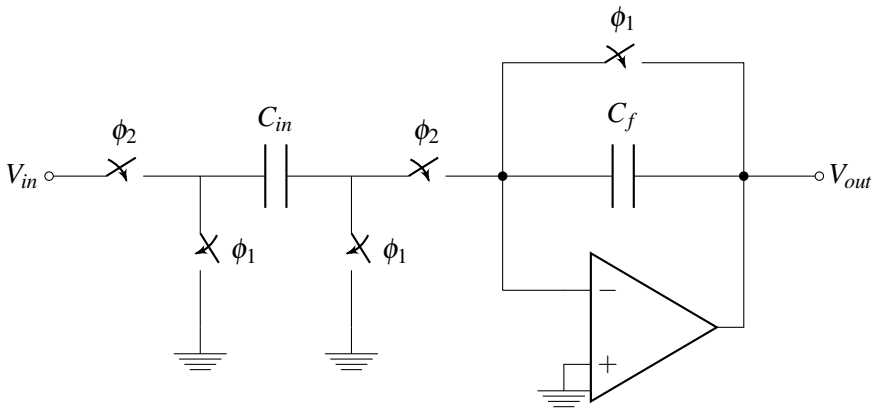
As your EE16A circuits toolkit is now complete with capacitors, op amps, and switches, we will revisit this problem to improve the readout electronics. The goal is to create a circuit block that will output voltage as a linear function of the water height, h_{H_2O}.

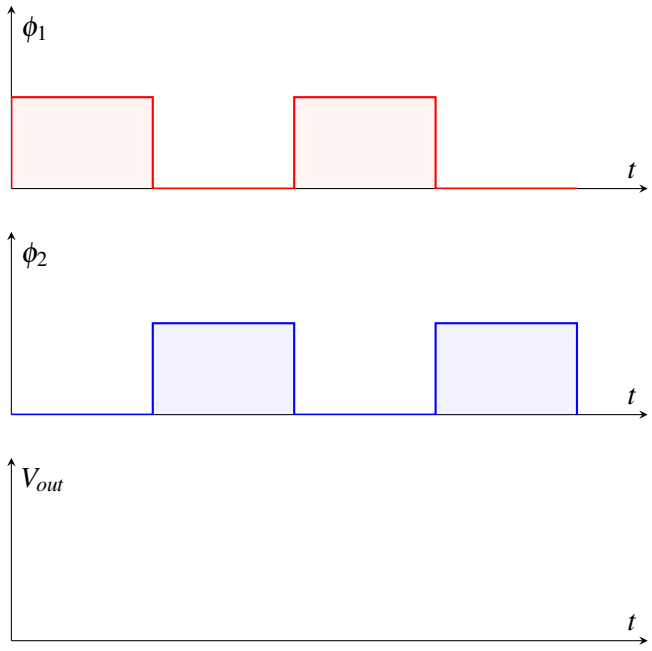
(a) What is the capacitance between terminals a and b when the tank is empty, C_{empty}? Again, the height of the water in the tank is h_{H_2O}. Modeling the tank as a pair of capacitors in parallel, find the total capacitance C_{tank} between the two plates. Can you write C_{tank} as a function of C_{empty}?
Note: the permittivity of air is ε, and the permittivity of rainwater is 81ε.

(b) One of the limitations of the previous readout circuit implementation was that it relied on capacitor charge sharing. Here we will analyze a circuit that transfers all charges for efficient readout. For the circuit below, draw the output waveform of V_{out} as a function of V_{in}, C_f, and C_{in}.

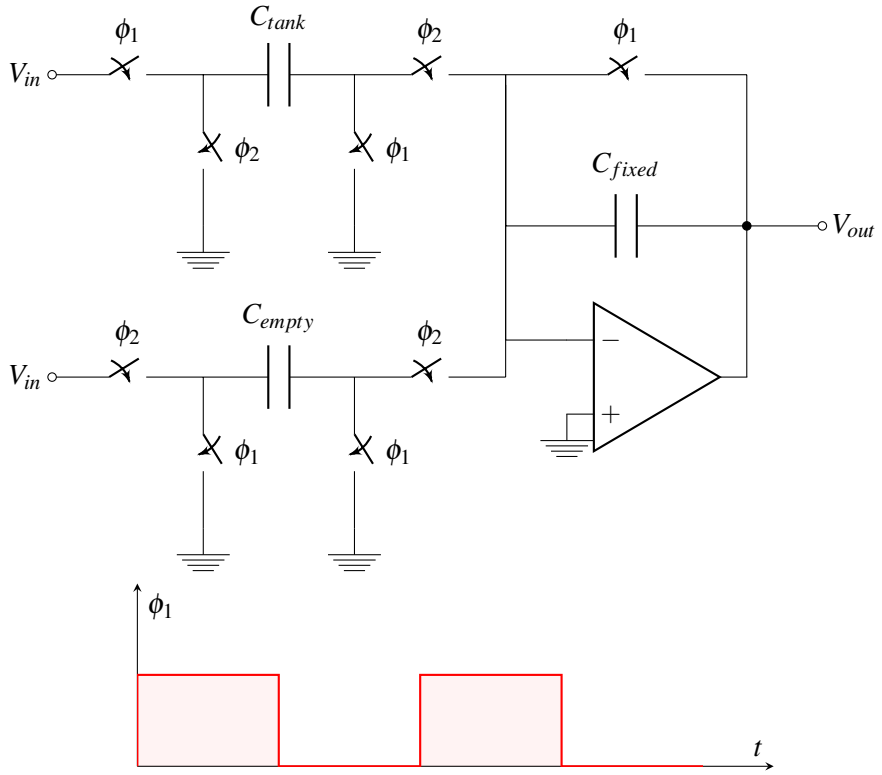


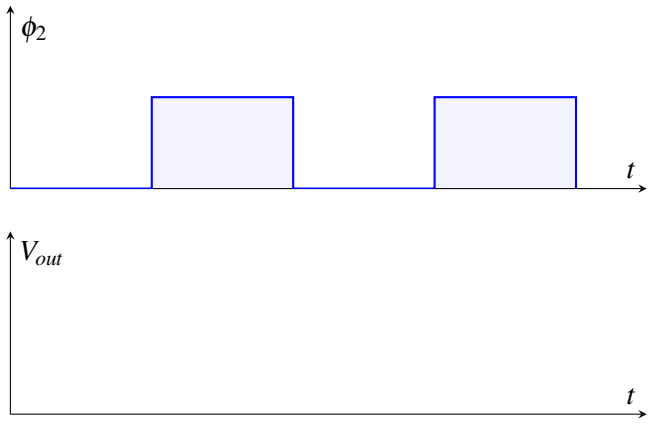
(c) The previous configuration is the non-inverting configuration. Now, we will look into the inverting configuration. For the circuit below, draw the output waveform of V_{out} as a function of V_{in}, C_f, and C_{in}.





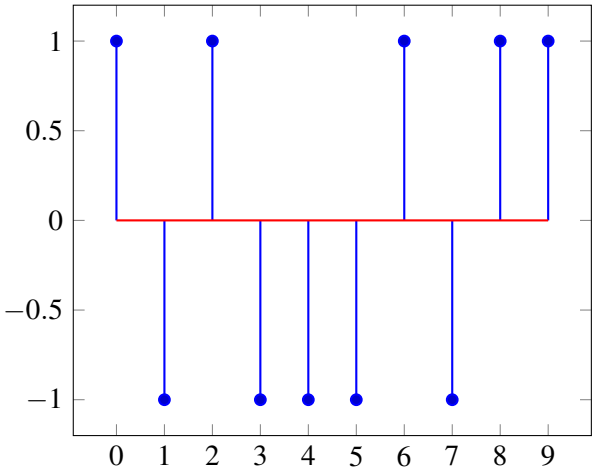
(d) With the help of the basic circuit blocks shown in parts (b) and (c), we will now implement a circuit that will output voltage as a linear function of the water height, h_{H_2O}. In addition to the rain-sensing capacitor, we will use two fixed value capacitors C_{fixed} and C_{empty}. Use the values obtained in part (a) for C_{tank} and C_{empty}. For the circuit below, draw the output waveform of V_{out} as a function of V_{in}, C_{fixed}, ε and h_{H_2O}.

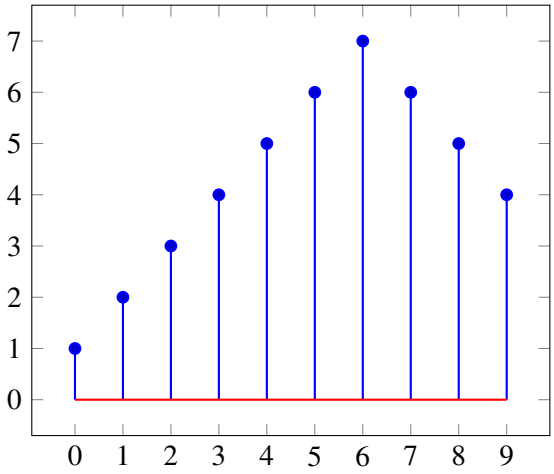




(e) Compare this version of the readout circuit to the previous version. Justify your choice.

3. Mechanical: Correlation





(a) Calculate and plot the **autocorrelation** (the inner products of one period of the signal with all the possible shifts of one period of the same signal) of each of the above signals. Each signal is periodic with a period of 10 (one period is shown).

(b) Calculate and plot the **cross-correlation** (the inner products of one period of the first signal with all possible shifts of one period of the second signal) of the two signals. Each signal is periodic with a period of 10 (one period is shown).

4. Finding Signals in Noise

Disclaimer: This problem looks long. However, almost all of the parts involve only running the provided IPython code and commenting on its output.

In this problem, we will explore how to use correlation and least squares to find signals, even in the presence of noise and other interfering signals.

(a) Suppose there is a transmitter sending a known signal $\vec{s}_1$, which is periodic with length $N = 1000$. Say $\vec{s}_1$ is chosen to be a random $\{+1, -1\}$ vector (for example, by tossing a coin N times and replacing

every heads with $+1$ and every tails with -1). For convenience, let us also normalize $\vec{s}_1$, so that $\vec{s}_1$ has a norm of 1. That is, the vector $\vec{s}_1$ looks like:

$$\vec{s}_1 = \frac{1}{\sqrt{n}} \begin{bmatrix} +1 & -1 & -1 & +1 & -1 & \cdots \end{bmatrix}^T$$

(where the ± 1 entries are chosen randomly).

We claim that such a vector $\vec{s}_1$ is "approximately orthogonal" to circular shifts of itself. That is, if $\vec{s}_1^{(j)}$ denotes circularly shifting $\vec{s}_1$ by j, then for all $j \neq 0$:

$$\left| \left\langle \vec{s}_1, \vec{s}_1^{(j)} \right\rangle \right| \lessapprox \varepsilon$$

for some small epsilon.

Run the provided IPython code to generate a random $\vec{s}_1$ and plot its autocorrelation (all inner products with shifted versions of itself). Run this a few times, for different random vectors $\vec{s}_1$. Around how small are the largest inner products $\left\langle \vec{s}_1, \vec{s}_1^{(j)} \right\rangle$, $j \neq 0$?

Recall that we normalized $\vec{s}_1$, such that $\langle \vec{s}_1, \vec{s}_1 \rangle = 1$.

(b) Suppose we receive a signal $\vec{y}$, which is $\vec{s}_1$ delayed by an unknown amount. That is,

$$\vec{y} = \vec{s}_1^{(j)}$$

for some unknown shift j. To find the delay, we can choose the shift j with the largest inner product $\left\langle \vec{s}_1^{(j)}, \vec{y} \right\rangle$.

Run the provided IPython code to plot the cross-correlation between $\vec{y}$ and $\vec{s}_1$ (i.e. all the shifted inner products). Can you identify the delay? Briefly comment on why this works using the findings from the previous part. *(Hint: What does $\left\langle \vec{s}_1^{(k)}, \vec{y} \right\rangle$ look like, for $k = j$? For $k \neq j$?)*

(c) Now suppose we receive a slightly noisy signal:

$$\vec{y} = \vec{s}_1^{(j)} + 0.1\vec{n}$$

where the "noise" source $\vec{n}$ is chosen to be a random normalized vector, just like $\vec{s}_1$.

Run the provided IPython code to compute $\langle \vec{s}_1, \vec{n} \rangle$. Run this a few times for different random choices of $\vec{s}_1, n$. Around how small is $\| \langle \vec{s}_1, \vec{n} \rangle \|$? How does this compare to $\left\langle \vec{s}_1, \vec{s}_1^{(j)} \right\rangle$ from your answer in part (a)?

(d) Can we identify the delay from this noisy reception? In this case, we do not know the noise $\vec{n}$, and we do not know the delay j. (But, as before, we know that the signal $\vec{s}_1$ being transmitted).

Run the provided IPython code to plot the cross-correlation between $\vec{y}$ and $\vec{s}_1$. Briefly comment on why this works to find the delay using the findings from the previous part.

(e) What if the noise is higher? For example:

$$\vec{y} = \vec{s}_1^{(j)} + \vec{n}$$

Does cross-correlation still work to find the delay? (Use the provided IPython notebook).

What about very high noise?

$$\vec{y} = \vec{s}_1^{(j)} + 10\vec{n}$$

Does cross-correlation still work to find the delay? If not, can you explain why? (use findings from previous parts).

(f) Now suppose there are two transmitters, sending known signals $\vec{s}_1$ and $\vec{s}_2$ at two unknown delays. That is, we receive

$$\vec{y} = \vec{s}_1^{(j)} + \vec{s}_2^{(k)}$$

for unknown shifts j,k. Both signals $\vec{s}_1$ and $\vec{s}_2$ are chosen as random normalized vectors, as before.

We can try to find the first signal delay by cross-correlating $\vec{y}$ with $\vec{s}_1$ (as in the previous parts). Similarly, we can try to find the second signal delay by cross-correlating $\vec{y}$ with $\vec{s}_2$. Run the provided IPython code to estimate the delays j,k. Does this method work to find both delays? Briefly comment on why or why not.

(g) Now, suppose the second transmitter is very weak, so we receive:

$$\vec{y} = \vec{s}_1^{(j)} + 0.1\vec{s}_2^{(k)}$$

Does the method of the previous part work reliably to find signal $\vec{s}_1$? What about $\vec{s}_2$? (Run the provided code a few times to test for different choices of random signals). Briefly justify why or why not.

(Hint: $\vec{s}_1$ looks like "noise" when we are trying to find $\vec{s}_2$. Based on the previous parts, would you expect to be able to find $\vec{s}_2$ under such high noise?)

(h) To address the problem of the previous part, suppose we use the following strategy: First, cross-correlate to find the delay j of the strongest signal (say, $\vec{s}_1$). Then, subtract this out from the received $\vec{y}$, to get a new signal $\vec{y}' = \vec{y} - \vec{s}_1^{(j)}$. Then cross-correlate to find the second signal in $\vec{y}'$.

Run the provided IPython code to test this strategy for the setup of the previous part (with a strong and weak transmitter). Does it work? Briefly comment on why or why not.

(i) Finally, suppose the amplitudes of the sent signals are also unknown. That is:

$$\vec{y} = \alpha_1\vec{s}_1^{(j)} + \alpha_2\vec{s}_2^{(k)}$$

for unknown amplitudes $\alpha_1 > 0$, $\alpha_2 > 0$, and unknown shifts j,k.

Can we use inner products (cross-correlation) to find the amplitudes as well? For example, suppose we find the correct shift j via cross-correlation. Then, briefly comment on why the following holds:

$$\left\langle \vec{s}_1^{(j)}, \vec{y} \right\rangle \approx \alpha_1$$

Run the provided IPython notebook to try this method of estimating the coefficients α_1, α_2. Roughly how close are the estimates to the actual amplitudes? (Run the code a few times to test different choices of random signals).

(j) Repeat the above for when there is some additional noise as well:

$$\vec{y} = \alpha_1\vec{s}_1^{(j)} + \alpha_2\vec{s}_2^{(k)} + 0.1\vec{n}$$

Roughly how close are the estimates to the actual amplitudes? (Run the code a few times to test different choices of random signals).

5. Homework process and study group

Who else did you work with on this homework? List names and student ID's. (In case of hw party, you can also just describe the group.) How did you work on this homework?
Working in groups of 3-5 will earn credit for your participation grade.