Spring 2011, Aaron Lanterman

ECE 6279: Spatial Array Processing
Homework 3

Due date: Friday 2/18/11 for on-campus students, Friday 2/25/11 for distance learning students. Homeworks are due at the *start* of class; homeworks turned in significantly later in the hour may be penalized at my discretion.

Late due date (30% penalty): Monday 2/21/11 for on-campus students; Monday 2/28/11 for distance learning students. (Again, if you need to use this late option, your homeworks are due at the *start* of class.)

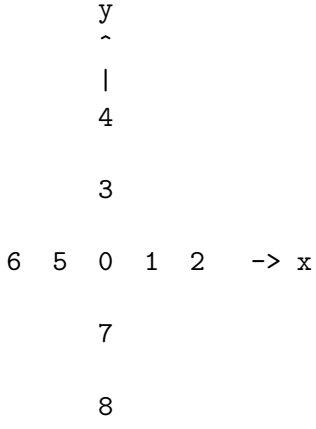
You are welcome to discuss approaches to the problems and solutions to difficulties you encounter with one another and with others outside the class. You can and should learn from each other as much as, and even more than, you learn from me. However, **your solutions should be your own work and should be written up by yourself**; feel free to discuss things, but **don't be looking at someone else's paper when you are writing your solution**. It's too easy to freeload that way and not learn anything. See the class website for more guidelines.

Looking at solutions to homeworks and quizzes from previous offerings of ECE6279 is expressly forbidden. Look, here I am expressing how forbidden it is. Forbidden! Forbidden!!!

A 30% penalty will be assessed on late homeworks (even homeworks turned in later the day it is due); I will distribute solutions to students in the different sections (on-campus or distance learning) shortly after class on the associated "late option" due date, so I will not accept solutions after that. If you cannot make a class, try to make arrangements to get your homework to me ahead of time.

1 Required Problems

The two problems in this homework will consider the array geometry for a 9-element array of omni-directional narrowband sensors, lying in the $\phi = \pi/2 = 90^\circ$ plane, centered at the origin, as shown below:



The numbers on the sensors indicate the order in which the sensor outputs appear in the observation vectors. The inter-sensor spacing along each axis is one-half wavelength. Suppose the sensors suffer from white Gaussian noise, each with the same noise power, uncorrelated from element to element.

In some of the parts below, you will use the "ideal" covariance matrix that you can directly calculate using the given signals and noise properties. In other parts, you will use an empircal covariance matrix created from a certain number of random snapshots. (Of course, in real life, you have to use the second method). If you're not familiar with making plots in MATLAB, type `help plot` at the MATLAB prompt for guidance.

To help jump-start you, I've provided several MATLAB files on the course website. You should download these and use them as a basis for your experiments. If you choose to customize the given code to suit your taste, please provide printouts of whatever routines you modify so I can follow what you're doing. Hand-write some annotations on your plots to tell me what the interesting features are.

Look over the code carefully, particulary in `steered_response_hw4_11.m`, and make sure you understand what I'm doing and how I'm doing it. That particular file is cut & pasted from a glob of code I was playing with while making up the homework; since I pulled out random bits it won't run as-is. You'll need to figure out the parts to fill in.

Below, we you will make a lot of plots of the power of the steered response of the conventional beamformer as a function of θ, for $0 \leq \theta < 360^\circ$, for a fixed $\phi = 60^\circ$.

For each problem, be sure to turn in listings of your code with your homework. Obviously, you don't need to turn in a separate listing for every parameter variation given in each problem; just one will suffice.

1. Let's start by exploring some resolution issues. Consider two signals, one with power 2 at $\theta_1^0 = 45^\circ$, $\phi_1^0 = 60^\circ$, and one with power 2 at $\phi_2^0 = 60^\circ$. We will vary θ_2^0. Use a noise power of 10.

   (a) Plot the power of the beamformer output as a function of θ using the **ideal** covariance matrix for two cases: $\theta_2^0 = 70^\circ$ and $\theta_2^0 = 90^\circ$. For the second case, you should clearly see two peaks corresponding for the two sources. For the first case, you will see that they have fused into one peak. Experiment to find the threshold between these two cases; i.e. if you start with $\theta_2^0 = 90$ and work you way down, find the θ^0 for which the two peaks just start to blend together in such a way that you cannot tell that two peaks are present. (This is a subjective decision.) Make a plot for this case. (Provide all three plots.)

   (b) Run the same experiment as in the previous part, for the cases $\theta_2^0 = 75^\circ$ and $\theta_2^0 = 83^\circ$ (those seemed to give me some interesting results), except this time use a "real" covariance matrix created by averaging 40 random snapshots. Run your experiment many times, looking at the plot each time. Count how many times in each case you are able to distinguish two peaks at more-or-less the correct locations. (Again, this is a subjective judgement). What percentage of the time are you able to distingush two peaks in each case? Run your experiment enough times that you feel like your results are vaguely statistically significant. Your percentage for the $\theta_2^0 = 83^\circ$ case should be higher than for

the $\theta_2^0 = 75^\circ$ case. For **each** θ_2^0 studied, provide **one** example plot showing a case where you can identify two peaks, and provide **second** example plot of a case where you can't. (Hence, you'll give me four plots in all on this subpart).

2. Now let's explore some accuracy issues. In this problem, we'll be considering a single source with power 2 at $\theta_1^0 = 150^\circ$, $\phi_1^0 = 60^\circ$. (You can handle the single-source case in my code easily by just setting one of the source powers to zero.) As in the previous part, we will assume the look angle ϕ is fixed at 60°.

   Do a Monte Carlo analysis in which you run the program hundreds of times (I recommend at least 1000 runs to get good histograms - this will tie up your machine for a bit, so go get a cup of coffee or something) via a loop, and store the θ_1 estimated by your beamformer for each of those runs. (I did this by using the same dense θ grid I used to make the plots in the previous problem followed by using MATLAB's "max" command. A grid spacing of one degree seems to work O.K., and makes the programming easy.)

   For each of the cases below, plot a histogram of the error (estimated θ minus true θ). Some of your histograms will look vaguely Gaussian; others will look like a mixture of a tight Gaussian with a wide, low, flat surface, giving it a "thumbtack" appearance. Also compute and give the square root of the mean squared error.

   (a) Noise power of 10, number of snapshot 40;

   (b) Noise power of 20, number of snapshots 40;

   (c) Noise power of 30, number of snapshots 40;

   (d) Noise power of 10, number of snapshots 10. (This case explores a situation where we have a low-quality covariance estimate).

   For each of the above cases, would you describe the histogram as a "Gaussian" or "thumbtack?" Provide your histogram plots as well as your square-root-of-mean-square-error values.

   (Note: It really doesn't make sense to compute a squared error for large angular errors, since angles take values on a circle and a squared error really makes sense for a Euclidian space. For small errors, it's not a big deal, but for big errors, it may not make a lot of sense, since, after all, the most error you can have is 180 degrees. Handling things properly requires worrying about whether measurements are "wrapping around." I just want you to get a rough feel for how the errors change with various parameters, so we won't worry about these issues here.)