The Cooper Union Department of Electrical Engineering
Prof. Fred L. Fontaine
ECE416 Adaptive Filters
Problem Set I: Linear Algebra
May 30, 2019

Note: Unless stated otherwise, consider only finite dimensional linear spaces. Also assume complex values.
Aside from basic MATLAB functions, in this homework you are allowed to use: *rank, orth, svd, svds, eig, eigs, pinv, chol.* Use *doc* to familiarize yourself with these functions. Given that these functions are available to you, I want you to use *efficient code.* Do NOT find more advanced or more sophisticated functions.
Notation: For a matrix A, $\mathcal{R}_A$ represents the range space and $\mathcal{N}_A$ the null-space. If $\mathcal{K}$ is a linear space, $P_{\mathcal{K}}$ is the orthogonal projection onto that space.
Matrix inversion lemma:

$$\left(B^{-1}+CD^{-1}C^H\right)^{-1}=B-BC\left(D+C^HBC\right)^{-1}C^HB$$

1. Let A be a square invertible matrix, α a scalar and u a vector. Use the matrix inversion lemma to simplify:

$$\left(A+\alpha u\ u^H\right)^{-1}$$

For certain values of α, this matrix may not be invertible; assume this is not the case. Now take the special case where $A=\delta I$. The answer reduces to something of the form $c_1 I + c_2 u\ u^H$. Find this expression. Also, for this case, determine the condition on α for this to be invertible.

2. Let $\{u_i\}_{i=1}^r$ be r linearly independent $M\times 1$ vectors, and $\{v_i\}_{i=1}^r$ be r linearly independent $N\times 1$ vectors. Show that the $M\times N$ matrix A given by:

$$A=\sum_{i=1}^r u_i v_i^H$$

has rank *exactly* r. *Hint:* Think about $\mathcal{R}_A$ and $\mathcal{N}_A^{\perp}$.

3. Let $\vec{x},\vec{u}$ be $M\times 1$ column vectors. We want to find optimal α such that $\vec{x}\approx\alpha\vec{u}$.

   (a) Find $u^{\#}$. Use your formula to show that:

$$\alpha=\frac{\langle x,u\rangle}{|u|^2}$$

   (b) Let's generalize this. First: if $D=diag\{d_i\}$, does DA or AD scale the columns of A by d_i's? So then what scales the rows?

   (c) Let U be an $M\times N$ matrix with *orthogonal* (not necessarily orthonormal) columns $\{u_i\}$. Find the SVD for U, from that find $U^{\#}$, and use this to solve the following problem: given x, find coefficients $\{\alpha_i\}$ such that:

$$x\approx\sum\alpha_i u_i$$

   Note: You should know the answer. Here we are using the pseudo-inverse to confirm it.

4. Let $\mathcal{U}$ be a linear subspace of $\mathbb{C}^M$, and P be the $M \times M$ projection matrix onto $\mathcal{U}$. It can be shown that $P = P^H$ and $P = P^2$ (I'm not asking you to prove this).

   (a) Prove the converse: if P is an $M \times M$ matrix that is Hermitian with $P = P^2$, then it is the projection matrix onto a subspace; this space in fact is the _____ space of P. *Hint:* First prove that, for any x, y, $Px \perp (I - P)\, y$.

   (b) Let P be the projection onto $\mathcal{U}$. Consider $H = I - 2P$. The expression Hx is called the *Householder transformation* of x. Consider the expansion $x = x_1 + x_2$, where $x_1 \in \mathcal{U}$, $x_2 \in \mathcal{U}^\perp$. Express Hx in terms of x_1 and x_2. **Note:** In the special case where $\mathcal{U}$ is 1-dimensional, say of the form $\{\alpha \mathbf{v} : \alpha \in \mathbb{C}\}$ for a unit vector $\mathbf{v}$, then the Householder transformation is the reflection across the hyperplane that is $\perp \mathbf{v}$.

5. **Sensor Array Signal Model**

   You will be implementing a number of algorithms for array processing in this course. To prepare for that, this problem has you write the simulation code to generate the signals received at the output of the sensor array, and then apply basic SVD analysis. Note we will assume ALL random signals are 0-mean.

   First some preliminary comments regarding Gaussian noise. If $n = n_I + jn_Q$ is scalar Gaussian, its variance $E\left(|n|^2\right) = E\left(|n_I|^2\right) + E\left(|n_Q|^2\right)$ (even if n_I, n_Q are correlated!). Thus, for example, the following will generate an $M \times N$ matrix, where every entry is UNIT variance complex Gaussian, and all entries are independent:

$$n = 1/sqrt(2) * (\mathrm{rand}\, n\,(M, N) + j * \mathrm{rand}\, n\,(M, N))$$

   Suppose we want to create an $M \times 1$ random Gaussian vector with covariance matrix C, and mean μ. If n is a unit variance white (complex) Gaussian vector, then:

$$x = C^{1/2} n + \mu$$

   will yield $x \sim N(\mu, C)$ where $C^{1/2}$ could either be the Cholesky factor or Hermitian square root of C.

   We consider multiple plane wave sources incident on a sensor array with M isotropic elements. Let $\{\vec{r}_i\}_{i=1}^M$ be the coordinates of the array. The parameter vector $\Theta = (\theta, \phi)$ is called the *angle of arrival (AOA)* or *direction of arrival (DOA)*, where we use spherical coordinates; θ is called the polar angle $(0 \le \theta \le \pi/2)$ and ϕ the azimuthal angle $(0 \le \phi \le 2\pi)$, and a unit vector in Θ direction is given by:

$$\hat{a}(\Theta) = \sin\theta \cos\phi \hat{a}_x + \sin\theta \sin\phi \hat{a}_y + \cos\theta \hat{a}_z$$

   The wavenumber vector then has the form:

$$\vec{k}(\Theta) = \frac{2\pi}{\lambda} \hat{a}(\Theta)$$

   where here we consider the narrowband case only, i.e. λ is a constant. Then the *steering vector* is:

$$s(\Theta) = \begin{bmatrix} \exp\left(-j\vec{k}(\Theta) \cdot \vec{r}_1\right) \\ \vdots \\ \exp\left(-j\vec{k}(\Theta) \cdot \vec{r}_M\right) \end{bmatrix}$$

Note this is not a unit vector, but $\frac{1}{\sqrt{M}}s(\Theta)$ is (i.e., the necessary scaling does not depend on Θ). Assume all coordinates $\vec{r}_i$ are expressed as multiples of an underlying spacing d; in this case, $s(\Theta)$ will depend on the factor d/λ; assume d/λ is given as an external parameter. Each snapshot (instant in time n) is an $M \times 1$ vector, comprised of L sources, plus noise:

$$u[n] = \frac{1}{\sqrt{M}} \sum_{i=1}^{L} \beta_i[n] s(\Theta_i) + v[n]$$

The Gaussian noise $v[n]$ is white across both space and time, i.e. $E\left(v[n] v^H[n]\right) = \sigma_v^2 I_{M\times M}$, and for $n \neq n'$, $E\left(v[n] v^H[n']\right) = 0_{M\times M}$. At each time, $\beta_i[n]$ for $i \leq n \leq L$, are scalars, representing the signals associated with each source. The relationships across space and time for $\beta_i[n]$ can be more complicated. Clearly it is not unreasonable to assume they are NOT white across time; more generally, each could be a WSS signal with some presumed correlation function. Also, the sources are not necessarily uncorrelated with each other; for example, in a multipath environment, reflections may cause the same underlying signal to arrive from multiple directions upon the array. However, at least for now, we are going to keep things simple. We will assume the source signals $\beta_i[n]$ are uncorrelated with each other, and are white Gaussian across time, with respective variances σ_i^2, $1 \leq i \leq L$. The normalization $\frac{1}{\sqrt{M}}$ is there so σ_i^2 and σ_v^2 can be properly viewed as signal and noise powers respectively. If S is the $M \times L$ matrix with *normalized* (unit length) steering vectors as columns, $b[n]$ the $L \times 1$ vector with entries $\{b_i[n]\}$, then each snapshot can be written as:

$$u[n] = Sb[n] + v[n]$$

To avoid confusion between the signal variances σ_i^2 and the singular values of A you will be computing later (which are NOT the same), let's use $\sigma_{S1}, \sigma_{S2}, \cdots$ to denote the singular values. (If we use s_i for singular values that looks too much like the steering vectors! Ugh)

(a) Write code that takes the given parameters as input, and outputs the steering matrix S, and the data matrix A, whose columns are $u[n]$ for $1 \leq n \leq N$. I expect EFFICIENCY and CLARITY in the code. The ONLY for loop you are allowed is one loop over the source index $1 \leq i \leq L$ to generate the S matrix. Once you have S, you must compute A with no for loops.

(b) Determine the formula for correlation matrix R of $u[n]$, and write code to compute it. Also, as specified, either $\frac{1}{K}A^H A$ or $\frac{1}{K}AA^H$ is an unbiased estimate of R, for some choice of K. Via ergodic arguments, this can be used as an estimate $\hat{R}$ of R. Determine the right answer, and write code to compute $\hat{R}$ as suggested.

(c) Take $d/\lambda = 0.5$, and consider an array in the form of a cross of two linear arrays aligned along the x- and y-axes, respectively. That is, take locations at $(md, 0, 0)$ and $(0, md, 0)$ where $-10 \leq m \leq 10$ (Don't count the origin twice!) Assume three sources, with normalization $\sigma_1^2 = 1$; the other two sources are $5dB$ and

$10dB$ below the primary source, and the noise is $20dB$ below the primary source. Assume AOAs as follows:

$$\begin{aligned}\theta_1 &= 10^\circ,\ \phi_1 = 20^\circ \\ \theta_2 &= 20^\circ, \phi_2 = -20^\circ \\ \theta_3 &= 30^\circ, \phi_3 = 150^\circ\end{aligned}$$

Take $N = 100$ snapshots and form the A matrix. Also compute R and $\hat{R}$ for this case.

1. Eigenanalysis of R can be used to determine "theoretical" values for the singular values and left singular vectors of A (obviously, since A is random, these are not what you would actually get from SVD of A; but we can interpret these as "ideal" representations for the signal and noise subspaces). Do the eigenanalysis to find the complete set of "theoretical" singular values and left singular vectors. Graph these computed singular values (as a stem plot), and confirm that there are three dominant singular values. Compute the projection matrices P_S, P_N onto the signal space and noise subspaces, respectively, from the singular vectors, and check that the singular vectors lie in the signal space by computing $|P_N s(\Theta_i)|$, which should be 0 but for numerical precision errors.
2. Consider the case where the white noise v is not present, i.e., $v[n] = 0$. This yields noise-free data matrix A_0 and correlation matrix R_0. Compare R_0 and R and comment what impact the noise has on the computed singular values and singular vectors.
3. Now perform SVD on A. Graph the singular values. Check that σ_{S4} appears much lower that σ_{S3}, which would be expected because we have a good SNR.
4. Let's compute SNR from the singular values. Based on your analysis above, you should expect $\sigma_{S4}^2 \approx \sigma_v^2$. What do you expect $\sum_{i=1}^{3} \sigma_{\mathrm{Si}}^2$ to be? Use this to compute a total SNR and compare to what it should be (i.e., add the signal powers- since they are uncorrelated this is reasonable- and divide by the noise power).
5. You have a theoretical value for the R matrix of u. On the other hand, you can estimate $\hat{R}$ from the data matrix A. Compute $\left\| R - \hat{R} \right\|$. In order to interpret how large this is, it would be reasonable to compare it to the noise power that is present. This is also equivalent to comparing it to a certain eigenvalue of R. Explain, and do this (i.e., compare). **Note:** Because computing $\hat{R}$ involves an averaging process, hopefully this error will be SMALLER than the intrinsic noise floor.
6. Using the ESTIMATED correlation matrix, compute the MVDR spectrum, and use the SVD of A to find the MUSIC spectrum. When computing the spectrum, use **unit length** steering vectors. Evaluate each on a grid (θ, ϕ) of 2°, and obtain both contour and surface plots of each.
7. Since the MUSIC spectrum and MVDR spectrum are computed for steering vectors, not "general" vectors, finding their theoretical minimum can be complicated. But we can at least get a lower bound: assuming we evaluate

these spectra for UNIT LENGTH steering vectors, obtain a lower bound for each. That is, if you are allowed to plug in ANY unit vector (not necessarily a steering vector), what would be the MINIMUM for each?

8. Plug in the three source steering vectors into the MVDR and MUSIC spectra, and compare to the theoretical lower bounds found above. You expect a peak at the source steering vectors: I want you see how tall the peak is relative to the theoretical minimum values.
9. Actually, since you are computing the MVDR and MUSIC spectra on a grid of (θ, ϕ) points, find the overal minimum values you computed, and compare to the theoretical lower bounds you found. Are they reasonably close?
10. You have $M = 21$ sensors in your array, and used $N = 100$ snapshots. Repeat the above with $N = 25$, and comment on what impact, if any, this reduction has on your results; in practice, people usually prefer to use $N > 2M$; here we have $N \approx M$ (though $N \geq M$ still holds). Repeat, again, if $N = 10$. It may be that something "fails" completely if $N < M$: if yes, report it. One could argue that, since $N > L$, we may still be able to identify the sources.
11. Reset back to $N = 100$. This time, however, make $\theta_2 = 10°$, $\phi_2 = 10°$. This brings the first two sources close together. Repeat your experiment and comment on any effect this may have had.