Machine Learning
Chapter 2: Latent linear models
Part 1: The PCA and MUSIC algorithms

1 Introduction

The purpose of this assignment is to learn basic PCA method and its probabilistic version using the EM algorithm.

The application example is the detection of angle of arrival in antenna array processing using the method known as MUltiple SIgnal Classification (MUSIC). The array signal model and the application, explained in the following section, are extracted from article [1] and they are the summary of the last given Latent Linear Variable lesson. The script for generating the signal is given along with this assignment.

2 Application example: Detection of angle of arrival

2.1 The MVDR algorithm

The minimum variance distortionless response (MVDR) method (see [2] for a full derivation) can be formalized as follows. Let $\mathbf{x}[n] = [x_1[n], \ldots, x_L[n]]^T$ be one of a set of snapshots taken from an array of D antennas which is illuminated by L narrowband signals of wavelength λ coming with different directions of arrival θ_m (see Figure 1). Each signal produces a spatial frequency

$$\omega_m = 2\pi \frac{d}{\lambda} sin(\theta_m),$$

where d is the element spacing of the array.

The power spectrum of process $\mathbf{x}[n]$ is estimated through a bank of filters $\{\mathbf{w}\}_k$, with length D, centred at frequencies ω_k. The output power of each

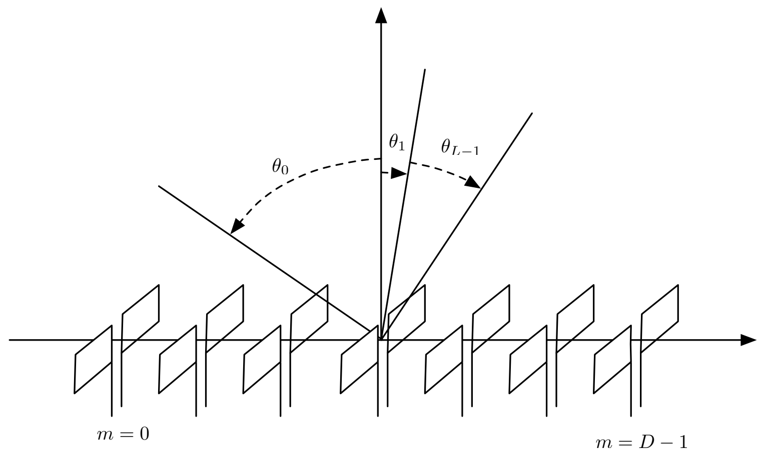

Fig. 1: Representation of an antenna array with D dipoles as array elements.

filter is

$$S_x(k) = \mathbb{E}[\mathbf{w}_k^H \mathbf{x}[n]\mathbf{x}^H[n]\mathbf{w}_k)] \tag{1}$$

where $\mathbb{E}[\cdot]$ is the expectation operator. A sample-based approximation to this power estimation is

$$S_x(k) \approx \frac{1}{N}\mathbf{w}_k^H \sum_n \mathbf{x}[n]\mathbf{x}^H[n]\mathbf{w}_k = \mathbf{w}_k^H \mathbf{R}\mathbf{w}_k \tag{2}$$

Let $\mathbf{X}$ be a matrix consisting of column vectors $\mathbf{x}[n]$, $1 \le n \le N$. Then, $\mathbf{R}$ has the expression

$$\mathbf{R} = \frac{1}{N}\mathbf{X}\mathbf{X}^H, \tag{3}$$

The criterion to design the filters is that their response to the input $\mathbf{x}[n]$ should present the minimum possible power, under the constraint of having unit response to a unit amplitude exponential signal with frequency ω_k, i.e.

$$\begin{aligned} &\min_{\mathbf{w}_k}\{\mathbf{w}_k^H \mathbf{R}\mathbf{w}_k\} \\ &\text{s.t} \qquad \mathbf{w}_k^H \mathbf{e}_k = 1 \end{aligned} \tag{4}$$

where $\mathbf{e}_k = [1, \exp(j\omega_k), \ldots, \exp(j(L-1)\omega_k)]^T$. Applying the method of Lagrange multipliers to the optimization problem in (4) leads to functional

$$L = \mathbf{w}_k^H \mathbf{R}\mathbf{w}_k + \lambda(1 - \mathbf{w}_k^H \mathbf{e}_k) \tag{5}$$

An optimization over all parameters of this functional gives the solution

$$\mathbf{w}_k = \frac{\mathbf{R}^{-1}\mathbf{e}_k}{\mathbf{e}_k^H \mathbf{R}^{-1}\mathbf{e}_k} \tag{6}$$

and then, the estimated power spectrum at frequency ω_k results

$$S_x^{MVDR}(k) = \frac{1}{\mathbf{e}_k^H \mathbf{R}^{-1}\mathbf{e}_k} \tag{7}$$

This is a well-known high resolution non-parametric spectrum estimation procedure [2].

2.2 A Relationship between MVDR and MUSIC

The autocorrelation matrix $\mathbf{R}$ can be decomposed in the form

$$\mathbf{R} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^H \tag{8}$$

where $\mathbf{V}$ is the matrix of eigenvectors of $\mathbf{R}$ and $\mathbf{\Lambda}$ is a diagonal matrix containing the corresponding eigenvalues. Let us define $\mathbf{V}_s$ and $\mathbf{V}_n$ as the L_s and L_n eigenvectors of the signal and noise subspaces, respectively. Matrices can be grouped in signal and noise elements:

$$\mathbf{V} = \begin{bmatrix} \mathbf{V}_s & \mathbf{V}_n \end{bmatrix} \tag{9}$$

$$\mathbf{\Lambda} = \begin{bmatrix} \mathbf{\Lambda}_s & \mathbf{0} \\ \mathbf{0} & \mathbf{\Lambda}_n \end{bmatrix} \tag{10}$$

By assuming orthonormality, $\mathbf{V}^{-1} = \mathbf{V}^H$, the inverse of $\mathbf{R}$ can be written as

$$\mathbf{R}^{-1} = \mathbf{V}\mathbf{\Lambda}^{-1}\mathbf{V}^H = \mathbf{V}_s\mathbf{\Lambda}_s^{-1}\mathbf{V}_s^H + \mathbf{V}_n\mathbf{\Lambda}_n^{-1}\mathbf{V}_n^H \tag{11}$$

Then, equation (7) can be rewritten as

$$S_x^{MVDR}(k) = \frac{1}{\mathbf{e}_k^H(\mathbf{V}_s\mathbf{\Lambda}_s^{-1}\mathbf{V}_s^H + \mathbf{V}_n\mathbf{\Lambda}_n^{-1}\mathbf{V}_n^H)\mathbf{e}_k} \tag{12}$$

It is worth noting that the original MUSIC algorithm [3, 4] can be viewed as a modification of this equation in which the matrices corresponding to the signal space are dropped, $\mathbf{V}_s\mathbf{\Lambda}_s^{-1}\mathbf{V}_s^H = \mathbf{0}$, and the noise eigenvalues are changed by ones, $\mathbf{\Lambda}_n = 1$, thus leading to

$$S_x^{MUSIC}(k) = \frac{1}{\mathbf{e}_k^H \mathbf{V}_n \mathbf{V}_n^H \mathbf{e}_k} \tag{13}$$

This form can be justified by deriving the MUSIC algorithm by means of a constrained minimization. Consider a bank of filters $\mathbf{w}_k$ where each one is tuned to the frequency ω_k. At the output of the filters, one wants to minimize the contribution of the signal subspace only, i.e., the filter must minimize

$$S_k^{MUSIC}(k) = \mathbf{w}_k^H \mathbf{V}_s \mathbf{V}_s^H \mathbf{w}_k, \tag{14}$$

where for a signal $\mathbf{e}_k$ of unit amplitude and frequency ω_k, the output of the filter must be $\mathbf{w}^H\mathbf{e}_k = 1$. Note that matrix $\mathbf{V}_s\mathbf{V}_s^H$ is not full rank, so this constrained minimization problem cannot be solved directly. Alternatively, one can minimize

$$\begin{aligned} &S_x^{MUSIC}(k) = \mathbf{w}_k^H \mathbf{V} \begin{bmatrix} \alpha\mathbf{I}_s & \mathbf{0} \\ \mathbf{0} & \beta\mathbf{I}_n \end{bmatrix} \mathbf{V}^H \mathbf{w}_k \\ &\text{s.t} \qquad \mathbf{w}_k^H \mathbf{e}_k = 1 \end{aligned} \tag{15}$$

where $\mathbf{I}_s$ and $\mathbf{I}_n$ are $L_s \times L_s$ and $L_n \times L_n$ identity matrices, and we assume that $\mathbf{V}$ is ordered so that all noise subspace vectors are grouped to the right and the signal vectors to the left. Also, $\alpha \gg \beta$ in order to have a full rank approximation to (14).

The optimization of functional (15) is identical to the optimization of (4) and it leads to solution

$$\mathbf{w}_k = \frac{\mathbf{V} \begin{bmatrix} \alpha^{-1}\mathbf{I}_s & \mathbf{0} \\ \mathbf{0} & \beta^{-1}\mathbf{I}_n \end{bmatrix} \mathbf{V}^H \mathbf{e}_k}{\mathbf{e}_k^H \mathbf{V} \begin{bmatrix} \alpha^{-1}\mathbf{I}_s & \mathbf{0} \\ \mathbf{0} & \beta^{-1}\mathbf{I}_n \end{bmatrix} \mathbf{V}^H \mathbf{e}_k}, \tag{16}$$

and, by plugging (16) into (15) and neglecting the contribution of α^{-1} (since $\alpha \gg \beta$), expression (13) is obtained.

3 Array data simulation

The following script produces the output of an antenna array with elements spaced $\lambda/2$ for several sources at directions ϕ_i. Just copy and paste in a Matlab editor to use it.

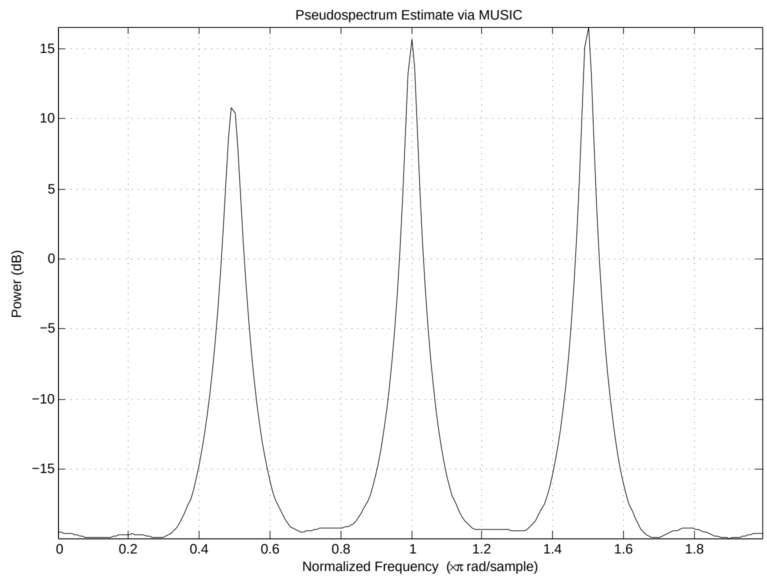

Fig. 2: Signal spectrum obtained using the standard PCA and the MUSIC function implemented in MatLab.

The script is to be used in a simulation with three sources with angles of arrival $\pi/2$, π and $3\pi/2z$ that simultaneously illuminate an array of 10 elements, with a noise standard deviation equal to 3. The goal is to compare the standard PCA and the EM based PCA in the MUSIC algorithm presented above. Figure 2 shows the expected result.

```
function [X,B]=generate_data(N,n_elements,phi,A,sigma)
%Inputs:
%    N: Number of samples
%    n_elements: Number of antennas (corresponds to dimension D)
%    phi: Vector of L sources
%    A: Vector of amplitudes for the L sources.
%    sigma: Noise standard deviation.
%
%Outputs
%    X: Matrix of snapshots.
%    B: Matrix of signals (not relevant in this assignment).
X=0;
```

```
B=[];
for i=1:length(A)
    %Produce a set of N complex baseband data fo direction i
    [x,b]=data(N,n_elements,phi(i));
    %Multiply it by its amplitude
    X=X+x*A(i);
    B=[B b];
end
%Add noise
X=X+sigma*(randn(size(X))+1j*randn(size(X)))/2;
function [x,b]=data(N,n_elements,phi)
%Simulate data in a N element array
x=zeros(n_elements,N); %Snapshots (row vectors)
b=sign(randn(N,1)) + 1j*sign(randn(N,1)); %Data
for i=1:N
    x(:,i)=exp(1j*phi*(0:n_elements-1)')*b(i);
end
```

4 Spectrum representation

The following function represents the spectrum, where W is the set of all eigenvectors, L is the number of signals. The script computes the FFT of the noise eigenvectors and plots its inverse in a log scale.

```
function draw_spectrum(W,L,text,fignum)
D=size(W,2);
figure(fignum)
w=1./sum(abs(fft((eye(D)-W(:,1:L)*W(:,1:L)'),1024)),2);
plot([0:1023]*2/1024,10*log10(w),'k')
title(text)
grid on
axis tight
end
```

5 Basic PCA

Implement the standard PCA algorithm and test it into the MUSIC algorithm. The steps of the procedure are:

1. Generate 100 samples of the data.

2. Compute the autocorrelation matrix of the data.

3. Compute all eigenvectors.

4. Compute the inverse of the power spectrum of the noise eigenvectors.

5. Use this expression to compute the MUSIC algorithm.

6. Represent the MUSIC spectrum in dB.

7. Compute the three first eigenvectors $\mathbf{V}_s$ of the autocorrelation, which are supposed to correspond to the signal eigenvectors. Since

$$\mathbf{V}\mathbf{V} = \mathbf{V}_s\mathbf{V}_s + \mathbf{V}_n\mathbf{V}_n = I$$

derive an expression that is equivalent to the inverse of the power spectrum of the noise eigenvalues.

8. Use this expression to compute the MUSIC algorithm.

9. Represent the MUSIC spectrum in dB and check that both graps are equal.

6 Probabilistic PCA

In this section, we will show that the EM algorithm produces a basis that is equivalent to the one spanned by the first eigenvectors, and that a modified version of the algorithm is able of actually produce all eigenvectors and their corresponding eigenvalues.

1. Using the EM algorithm presented in class, write a script that computes the three first factors $\mathbf{W}$ of data X and their corresponding latent variables $\mathbf{Z}$.

2. Compute the Fourier Transform of matrix $\mathbf{W}$ and represent it in a linear graph. You should be able to detect three frequency components, but with a worse resolution.

3. Compute the norm of these vectors and all dot products among them. What conclusion should you take from all results above?

The produced vectors are equivalent to the signal eigenvectors, but we must be able to produce the actual eigenvectors and eigenvalues. In order to do that, need to modify the algorithm to force the vectors in $\mathbf{W}$ to be orthonormal and also to be eigenvectors of the autocorrelation function. Prove that the following algorithm produces all the eigenvectors and eigenvalues if we choose $L = D$.

Repeat until convergence

1. $\mathbf{Z} = (\mathbf{W}^\top \mathbf{W})^{-1}\mathbf{W}^\top \mathbf{X}$
2. $\mathbf{W} = \mathbf{X}\mathbf{Z}^\top(\mathbf{Z}\mathbf{Z}^\top)^{-1}\mathbf{W}^\top \mathbf{R}\mathbf{W}$
3. $\mathbf{W} = gramschmidt(\mathbf{W})$

where $\mathbf{R} = \frac{1}{N}\mathbf{X}\mathbf{X}^\top$ is the sample estimation of the data autocorrelation matrix and $gramschmidt(\mathbf{W})$ is the Gram-Schmidt procedure, a function that outputs an orthonormal basis spanning the same space as $\mathbf{W}$.

The function is as follows:

1. For vectors $\mathbf{w}_j$, subtract their projections with the previous ones:

$$\tilde{\mathbf{w}}_i = \mathbf{w}_i - \sum_{j=1}^{i} \mathbf{v}_j^\top \mathbf{w}_i \mathbf{v}_j$$

   Obviously, for first vector, we omit this step.

2. Normalize the vector:

$$\mathbf{v}_i = \frac{\tilde{\mathbf{w}}_i}{|\tilde{\mathbf{w}}_i|}$$

Work out the following steps:

- Why line 2 of the PCA algorithm is different from what is seen in the theory?
- Using the script below, repeat the experiments of basic PCA but using the probabilistic PCA described above.

```
function V=gramsmith(V)
for i=1:size(V,2)
    V(:,i)=V(:,i)- V(:,1:i-1)*V(:,1:i-1)'*V(:,i);
    V(:,i)=V(:,i)/norm(V(:,i));
end
```

- Analyze the number of steps necessary to achieve a reasonable convergence.
- Represent all eigenvectors. Using the noise eigenvectors, compute the noise power.
- Is it necessary to sort the eigenvalues? Why?
- Change lines 1 and 2 of the algorithm by $\mathbf{W} = \mathbf{R}\mathbf{W}$. Does it work? Why?

7 Cross validation of the number of factors

Assume now that no knowledge of the number of data is provided. Generate 100 data for training and 100 more for test, and using PPCA and the computation of the negative log-likelihood, cross validate the right value for L. Represent the cross validation in a graph.

References

[1] A. El Gonnouni, M. Martinez-Ramon, J.L. Rojo-Alvarez, G. Camps-Valls, A.R. Figueiras-Vidal, and C.G. Christodoulou, "A support vector machine music algorithm," *Antennas and Propagation, IEEE Transactions on*, vol. 60, no. 10, pp. 4901–4910, Oct 2012.

[2] H. Van Trees, *Detection, Estimation, and Modulation Theory, Part IV, Optimum Array Processing*, Wiley, New York, NY, 2002.

[3] R. O. Schmidt, "Multiple emitter location and signal parameter estimation," in *RADC Spectral Estimation Workshop*, Rome, NY, 1979, pp. 243–258.

[4] G. Bienvenu and L. Kopp, "Optimality of high resolution array processing using the eigensystem approach," *IEEE Transactions on on Acoustics, Speech and Signal Processing*, vol. 31, no. 10, pp. 1234–1248, oct 1983.

[5] J.-H. Ahn and J.-H. Oh, "A constrained EM algorithm for principal component analysis," *Neural Computation*, vol. 15, pp. 57–65, 2003.