18-391 Noisy Signal Representation and Processing: Lab 4

Power Spectral Density Estimation

(weeks of 03/14/11 and 03/21/11)

Instructions: Refer to the course BlackBoard for Lab policies and procedures.

1 Introduction

An important attribute of random noise is its power spectral density (PSD). PSD characterizes the distribution in frequency of the power of $x[n]$, a discrete random process (a sequence of random variables defined for every integer n where n usually is the time index). In this Lab, you will learn about PSD and will explore several techniques for PSD estimation using MATLAB. The noisy signals you will investigate in this Lab include the noisy signal collected from a real application as well as the synthetic signals. Through this Lab, you can see how PSD estimation is used for real applications.

* * *

Let us first look at the fundamental principles regarding PSD. PSD is defined as a Fourier transform of the auto-correlation function $r_{xx}(m)$ of the signal, *i.e.*,

$$P_{xx}(\omega) = \sum_{m=-\infty}^{\infty} r_{xx}[m] \exp(-j\omega m) \tag{1}$$

where $r_{xx}(m) = E\{x[n]x[n+m]\}$ denotes the auto-correlation function of a wide sense stationary (WSS) discrete random process. Note that the PSD is periodic in ω with frequency 2π. Thus, we need to show PSD only over a range of $-\pi$ to $+\pi$. If $x[n]$ is real-valued, then the auto-correlation function is real and symmetric, *i.e.*, $r_{xx}(m) = r_{xx}(-m)$ and its Fourier transform (namely, the PSD) is also real and even symmetric, *i.e.*, $P_{xx}(\omega) = P_{xx}(-\omega)$. As a result, we show PSD only from 0 to π. Finally, it can be shown that PSD is non-negative. The goal is to estimate the PSD of a random signal from a sequence of time samples of a signal. Depending on prior information about the signal, estimation can be divided into two categories: non-parametric and parametric approaches. Non-parametric approaches explicitly estimate the auto-correlation function or the power spectral density of the process without any prior information. On the other hand, parametric approaches assume that the underlying random process has a certain structure, for example, an auto-regressive (AR) model, which can be described using a small number of parameters and estimate the parameters of the model. A common non-parametric estimation approach is the periodogram, which is based on fast Fourier transform (FT). A common parametric technique is maximum entropy spectral estimation (MESE), which involves fitting the observed signal to an AR model. You will additionally learn about pseudospectrum estimation such as multiple signal classification (MUSIC) method based on eigen-decomposition.

As a reference, MATLAB has a PSD estimation function, named `spectrum`, which covers most of the PSD estimation schemes, even pseudospectrum estimation, which are listed below.

- `periodogram` : Periodogram
- `welch` : Welch
- `burg` : Burg, or maximum entropy spectral estimation
- `music` : Multiple Signal Classification
- `cov` : Covariance
- `mcov` : Modified covariance
- `mtm` : Thompson multitaper
- `yulear` : Yule-Walker
- `eigenvector` : Eigenvector

You can explore various methods using `spectrum`.(estimation method), such as `spectrum.periodogram` for periodogram estimation. You can use `spectrum` to check your result and for debugging. In this Lab, periodogram, Welch and MUSIC methods are mainly discussed.

1.1 Periodogram

For a finite length real-valued signal $x[n]$, $n = 0,\ 1,\ ...,\ L-1$, an estimator for the autocorrelation sequence is

$$c_{xx}(m) = \frac{1}{L} \sum_{n=0}^{L-|m|-1} x[n]x[n+|m|] \tag{2}$$

where $|m| < L$. As an estimate of PSD, periodogram is the Fourier transform of the biased autocorrelation estimate, $c_{xx}(m)$ in Eq. (2), *i.e.*,

$$I_L(\omega) = \sum_{m=-(L-1)}^{L-1} c_{xx}(m)e^{-j\omega m} \tag{3}$$

By substituting $c_{xx}(m)$, the periodogram can be represented as below,

$$I_L(\omega) = \frac{1}{L}|X(e^{j\omega})|^2 \tag{4}$$

where, the Fourier transform of the real finite-length sequence $x[n]$, $0 \leq n \leq L-1$, is

$$X(e^{j\omega}) = \sum_{n=0}^{L-1} x[n]e^{-j\omega n} \tag{5}$$

There are two ways to compute the PSD: Estimate the autocorrelation function (ACF) of the signal and then take a Fourier transform of the ACF as shown in Eq. (2); Take a Fourier transform of the signal first and square its magnitude as shown in Eq. (4). These two are equivalent and will be investigated later. Here, we will use the one taking the Fourier transform of the signal, since it is easy to implement using MATLAB. Periodogram can be obtained by using the fast Fourier transform (FFT), an algorithm that enables a faster computation of discrete Fourier transform (DFT) for the sequences of length $n = 2^k$, where k is an integer. As an example, a signal of f_c Hz, embedded in additive white Gaussian noise (AWGN) of $\mathcal{N}(0, \sigma^2)$ is generated using MATLAB with parameters as follows: $f_c = 200$ Hz, $\sigma^2 = 10^{-2}$, $f_s = 1$ kHz (sampling frequency), and $L = 300$. Fig. 1 illustrates the signal, which can be represented as below,

$$x[n] = \cos(2\pi f_c n T_s) + r[n], \qquad n = 0, 1, ..., L-1 \tag{6}$$

where, $T_s = 1/f_s = 1$ms and $r[n]$ is the AWGN noise sequence, $\sim \mathcal{N}(0, \sigma^2)$. The periodogram of the signal can be obtained from Eq. (4), which is shown in Fig. 2. The periodogram is plotted in double-sided form, *i.e.*, from zero frequency to sampling frequency. There are two peaks at 200 Hz and 800 Hz (=-200 Hz) and other small values are contributed from the added white Gaussian noise. If there is no noise, the periodogram will exhibit two delta functions at 200 Hz and 800 Hz (=-200 Hz). An example of the MATLAB source code for generating the signal and periodogram is shown in the following.

```
%% Example 1 - periodogram ******************************
% 1.1) Signal generation
randn('state',0);         % initialize randn
f_c = 200;                % signal frequency (Hz)
sig_sq = 0.01;            % variance of random noise N(0,n_var)
Fs = 1000;                % sampling frequency, 1 kHz
```

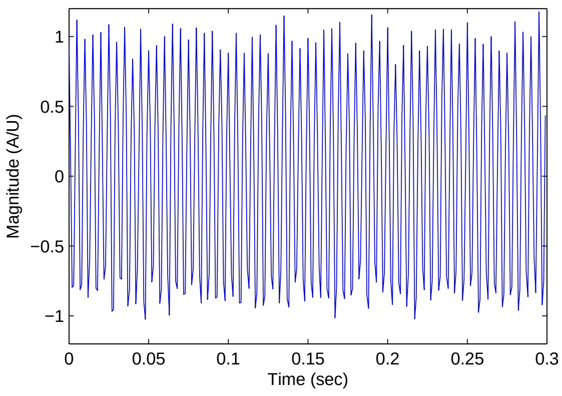

Figure 1: 200 Hz signal with additive noise, $\sim \mathcal{N}(0, 10^{-2})$

```
L=300;                    % no. of samples
t = 0:1/Fs:1/Fs*(L-1);  % time index, w/ sampling period Ts=1/Fs
x = cos((2*pi*f_c).*t)+sqrt(sig_sq).*randn(size(t));
            % f_c Hz signal with additive noise ~ N(0,sig_sq)
figure(1); plot(t,x);
axis([0 0.3 -1.2 1.2]);
xlabel('Time (sec)'); ylabel('Magnitude (A/U)');
title('200 Hz signal embedded in additive noise {\itN}(0,10^{-2})');

% 1.2) Periodogram using spectrum
n_t=numel(t);                     % length of sequence
Hs=spectrum.periodogram;          % set method to periodogram
psd_period=psd(Hs,x,'Fs',Fs,'NFFT',n_t,'SpectrumType','twosided');
% Hs : method (periodogram)
% Fs : sampling frequency
% NFFT : number of FFT
% SpectrumType : either 'twosided' (0~Fs) or 'onesided' (0~Fs/2)
figure(2); plot(psd_period);        % Display PSD
```

1.2 Averaging Modified Periodogram (Welch's Method)

The raw periodogram is not a statistically stable spectral estimate since there is not much averaging. The periodogram is computed from a finite-length observed sequence (length L in the previous description), that is sharply truncated at 0 and $L-1$, of the infinite length sequence, $-\infty \sim \infty$. This sharp truncation effectively spreads the original signal spectrum into other frequencies, which is called spectral leakage. The spectral leakage problem can be reduced by multiplying the finite sequence by a window function before FFT, which reduces the sequence values gradually rather than abruptly. Spectral leakage and window functions

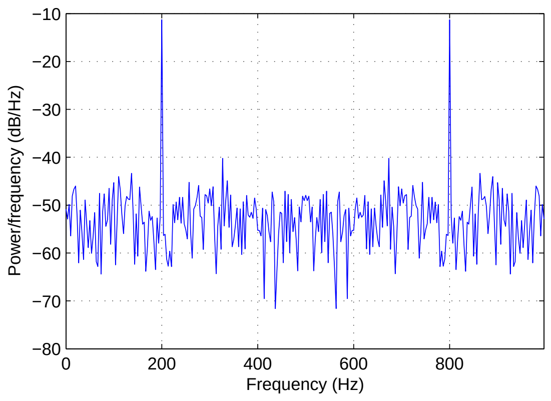

Figure 2: Periodogram PSD estimate

will be further investigated later. The variance can be reduced by averaging the periodogram. This modified algorithm is called Welch's method. A signal of length N is divided into K segments of length M, $K = N/M$, and K modified periodograms with window functions are averaged. This leads to a decrease in the variance of the estimate.

The modified periodogram for a segment of length M can be represented as below,

$$J_M^{(i)}(\omega) = \frac{1}{MU} \Big| \sum_{n=0}^{M-1} x^{(i)}[n] w[n] e^{-j\omega n} \Big|^2 \tag{7}$$

where $w[n]$ is the window function (to be discussed below in more detail) and

$$U = \frac{1}{M} \sum_{n=0}^{M-1} w^2[n]$$

The spectrum estimate is defined as below,

$$B_{xx}^{w}(\omega) = \frac{1}{K} \sum_{i=1}^{K} J_M^{(i)}(\omega) \tag{8}$$

In order to increase the number of segments being averaged in a finite-length sequence, the sequence can be segmented with overlap; for example, 50 % overlap can increase the number of segments of same length M from K to $2K - 1$. In this Lab, the effect of window functions, number of segmentations, and overlap percentages in the PSD estimate will be explored using MATLAB. MATLAB has various windows including, 'Bartlett', 'Bartlett-Hanning', 'Blackman', 'Blackman-Harris', 'Bohman', 'Chebyshev', 'Flat Top', 'Gaussian', 'Hamming', 'Hann', 'Kaiser', 'Nuttall', 'Parzen', 'Rectangular', 'Triangular', and 'Tukey'. The raw periodogram is a special case of Welch's method, with rectangular window, 0% overlap and no average ($K = N/M = 1$). One of the commonly used windows is a Kaiser window, which has a parameter that can change the properties of the window. In this Lab,

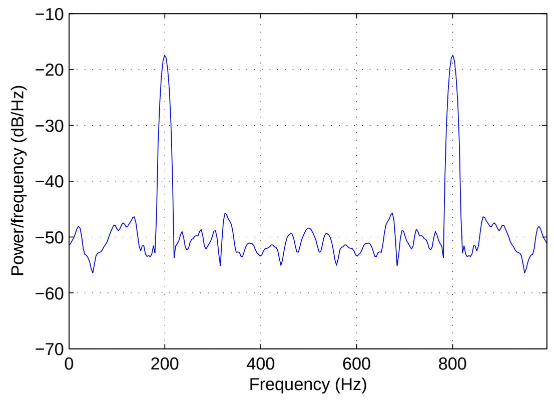

Figure 3: PSD estimate by Welch's method with the Hamming window, segment length of 100, and 50% overlap.

the Hamming window and Kaiser window will be investigated for PSD estimates. As an example of Welch's method, the PSD of the signal generated in the previous example is estimated with the Hamming window, segment length of 100, and 50 % overlap, and shown in Fig. 3 as below:

```
% Example 2 - Welch method (continued from example 1) *************
seg_lth=100;                    % segment length
ovl_per=50;                     % 50 % of overlap
Hs = spectrum.welch(‘Hamming’,seg_lth,ovl_per);
psd_welch=psd(Hs,x,‘Fs’,Fs,‘NFFT’,n_t,‘SpectrumType’,‘twosided’);
                                % Hs : method (Welch)
figure(11); plot(psd_welch);          % Display PSD
```

1.3 Multiple Signal Classification (MUSIC) Method

When a signal consists of a few tones and the differences between these frequencies are small, conventional PSD estimates may not be able to resolve these components. MUSIC is a pseudospectral estimate of the signal based on an eigenspace analysis, to improve the frequency resolution in PSD estimates. This method assumes that a signal, $x[n]$, consists of K complex exponentials in the presence of Gaussian white noise. Given an $M \times M$ autocorrelation matrix, $\mathbf{R}_{xx}$, if the eigenvalues are sorted in decreasing order, the eigenvectors corresponding to the K largest eigenvalues span the signal subspace. The other eigenvectors span the noise subspace, which is orthogonal to the signal subspace. A pseudospectrum can be obtained from the inverse of the Fourier transform of the noise subspace basis vectors. This provides higher resolution than the conventional PSD estimates and can be used to estimate the frequency components of the signal from the noisy samples. We will first introduce the

Pisarenko harmonic decomposition method, which is a basic frequency estimation method based on eigen-decomposition. The signal consists of K complex exponentials in the presence of Gaussian white noise and can be represented as below,

$$x[n] = \sum_{k=1}^{K} \alpha_k \exp(j2\pi f_k n) + r[n] \tag{9}$$

Here, f_s is assumed to be 1 Hz and $r[n]$ is a white noise sequence. The signal of current and future $M-1$ values, the time-window vector of a signal, $\mathbf{x}[n] = [x[n]\ x[n+1]\ x[n+2]\ ...\ x[n+M-1]]^T$, can be denoted as below,

$$\mathbf{x}[n] = \sum_{k=1}^{K} \alpha_k \mathbf{v}_{f_k} \exp(j2\pi f_k n) + \mathbf{r}[n] = \mathbf{s}[n] + \mathbf{r}[n] \tag{10}$$

Note that $\mathbf{v}_{f_k} = [1\ \exp(j2\pi f_k)\ \exp(j2\pi 2 f_k)\ ...\ \exp(j2\pi(M-1)f_k)]^T$, which corresponds to a length M DFT vector at frequency f_k. Assuming that the signal and noise components are uncorrelated, the autocorrelation matrix, $\mathbf{R}_{xx}$ can be represented as below,

$$\begin{aligned} \mathbf{R}_{xx} &= E(\mathbf{x}[n]\mathbf{x}[n]^H) = \mathbf{R}_{ss} + \mathbf{R}_{rr} \qquad (11) \\ &= \sum_{k=1}^{K} |\alpha_k|^2 \mathbf{v}_{f_k} \mathbf{v}_{f_k}^H + \sigma^2 \mathbf{I} \\ &= \mathbf{V}\mathbf{A}\mathbf{V}^H + \sigma^2 \mathbf{I} \end{aligned}$$

where H denotes a conjugate transpose, $\mathbf{V} = [\mathbf{v}_{f_1}\ \mathbf{v}_{f_2}\ ...\ \mathbf{v}_{f_K}]$ and $\mathbf{A}$ is a diagonal matrix with diagonal terms $|\alpha_1|^2$, $|\alpha_2|^2$, ..., $|\alpha_K|^2$. Here, $\mathbf{R}_{rr} = \sigma^2\mathbf{I}$ has rank M (*i.e.*, of full rank), while $\mathbf{R}_{ss}$ has rank K (*i.e.*, rank-deficient). Thus, the eigen-decomposition of $\mathbf{R}_{xx}$ will have K large eigen-values corresponding to the signal made up of complex exponentials, while the remaining eigenvalues are equal and correspond to the noise.

$$\begin{aligned} \mathbf{R}_{xx} &= \textstyle\sum_{m=1}^{K} \lambda_m \mathbf{q}_m \mathbf{q}_m^H + \sum_{m=K+1}^{M} \lambda_m \mathbf{q}_m \mathbf{q}_m^H \\ &= \mathbf{Q}_s \Lambda_s \mathbf{Q}_s^H + \mathbf{Q}_r \Lambda_r \mathbf{Q}_r^H \end{aligned} \tag{12}$$

where $\mathbf{Q}_s$ and $\mathbf{Q}_r$ are matrices of signal and noise eigenvectors, respectively. The projections to each subspace are

$$\mathbf{P}_s = \mathbf{Q}_s \mathbf{Q}_s^H, \quad \mathbf{P}_r = \mathbf{Q}_r \mathbf{Q}_r^H \tag{13}$$

where two subspaces are orthogonal as below,

$$\mathbf{P}_r \mathbf{Q}_s = \mathbf{0}, \quad \mathbf{P}_s \mathbf{Q}_r = \mathbf{0} \tag{14}$$

Pisarenko Harmonic Decomposition (PHD) uses the eigenvector associated with the smallest eigenvalue to estimate the frequencies of the complex exponentials. The autocorrelation matrix of $M \times M$ size, $M = K + 1$, is estimated from the noisy sequence as represented below,

$$\hat{\mathbf{R}}_{xx} = \frac{1}{M}\mathbf{X}^H\mathbf{X} \tag{15}$$

where $\mathbf{X}$ is a matrix defined as,

$$\mathbf{X} = \begin{pmatrix} x[0] & x[1] & ... & x[M-1] \\ x[1] & x[2] & ... & x[M] \\ ... & & & ... \\ x[N-1] & x[N] & ... & x[N+M-2] \end{pmatrix}$$

From the eigen-decomposition of $\hat{\mathbf{R}}_{xx}$, the noise subspace consists of a single eigenvector, $\mathbf{q}_M$, which corresponds to the minimum eigenvalue. Using this noise eigenvector, the pseudospectrum is estimated as below,

$$\hat{P}_{PHD}(f) = \frac{1}{|\mathbf{v}_f^H\mathbf{q}_M|^2} \tag{16}$$

where $\mathbf{v}_f = [1\ \exp(j2\pi f)\ \exp(j2\pi 2f)\ ...\ \exp(j2\pi(M-1)f)]^T$ and the denominator is the Fourier transform of the Mth eigenvector. Since the signal is orthogonal to the noise,

$$\mathbf{v}_{f_k}^H\mathbf{q}_M \sim 0, \qquad , k = 1,\ 2,\ ...,\ K \tag{17}$$

Thus, the resulting pseudospectrum shows peaks at the signal frequencies f_k. Because the method uses a single noise eigenvector, the result is very sensitive to any errors in the estimation of the noise eigenvector. MUSIC method uses more than one noise eigenvector by increasing M, $M > K + 1$ for pseudospectrum, as below.

$$\hat{P}_{MUSIC}(f) = \frac{1}{\sum_{m=K+1}^{M}|\mathbf{v}_f^H\mathbf{q}_m|^2} \tag{18}$$

Fig. 4 shows an example of pseudospectrum based on the MUSIC method. Here, the signal has two sinusoids at 200 Hz and 220 Hz and the pseudospectrum is plotted in a single sided form. The MUSIC estimate shows two components clearly, although the difference between two components is small. In the problems, we will see that MUSIC shows better frequency separation capability than other PSD estimates such as modified periodograms.

```
% Example 3 -MUSIC method ************************
f_c1=200;               % signal frequency (Hz)
f_c2=220;
x = cos((2*pi*f_c1).*t)+cos((2*pi*f_c2).*t)+sqrt(sig_sq).*randn(size(t));
Hs = spectrum.music(3,50);
psd_music=pseudospectrum(Hs,x,‘Fs’,Fs,‘NFFT’,n_t); %,‘SpectrumType’,‘twosided’);
                            % Hs : method (Welch)
figure(21); plot(psd_music);       % Display PSD
```

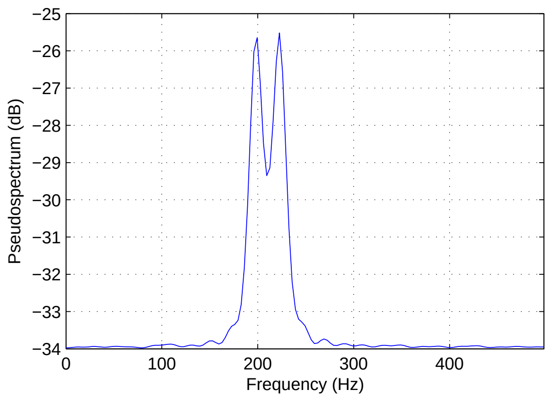

Figure 4: Pseudospectrum based on the multiple signal classification (MUSIC) method

2 Optional Reading

- A. V. Oppenheim and R. W. Schafer, Discrete-time Signal Processing, 2nd Edition, Chapter 10.6 $\sim$ 10.7, Prentice Hall
- S.M. Kay, S.L. Marple, Jr., "Spectrum analysis: A modern perspective," *Proc. IEEE*, vol. 69, no. 11, pp. 1380-1419, 1981.
- Wikipedia article on Power Spectral Density (PSD). http://en.wikipedia.org/wiki/ Spectral_density

3 Week 1 milestones [100 Points]

1. [Pre-Lab, 15 Points] The fast Fourier transform (FFT) of a signal provides the sampled Fourier transform of a signal. Generate a signal of 400 Hz cosine wave with additive noise $\mathcal{N}(0, \sigma^2 = 10^{-1})$ with a sampling frequency of 2 kHz and acquisition time of 0.2 sec, similar to the example in Section 1.1. Plot the magnitude of the FFT result of the generated signal, where the x-axis denotes the frequency (in Hz) and y axis denotes the FFT magnitude. You should see a spectral peak at 400 Hz. You can use the signal generation part of the example code and MATLAB FFT function, `fft`. Type `help fft` to learn how to use this function.
2. [10 Points] Write your own MATLAB function that computes a periodogram based on fast Fourier transform (FFT), as shown in Eq. (4). Use the signal generated from the previous problem for periodogram computation. Display the results by power/frequency (dB/Hz) as a function of frequency and compare the result to Fig. 2. Make sure that you use `fft` instead of `spectrum.periodogram` for this problem and the rest of the problems in this lab. Note, it is HIGHLY recommended that your

functionalize all of your periodogram code in this lab, as you will be using it several times both this week and next week. Writing functions will save you a lot of time.

3. [10 Points] Write your own MATLAB function that computes a modified periodogram with the Hamming window based on fast Fourier transform (FFT) from the signal you generated in the Problem 1. You can update the previous function by multiplying the window right before computing FFT. By applying the window, what changes do you observe in the power levels of background noise, compared to the raw periodogram (rectangular window)?

4. [10 Points] Write a new function where you change the Hamming window of the previous problem to a Kaiser window with the parameter $\beta = 2$. Use this function to compute a periodogram for the signal in Problem 1. What changes do you observe in the power levels of background noise, compared to the raw periodogram (rectangular window)? Using the window visualization tool, `wvtool`, compare the frequency domain response of these three windows - rectangular, Hamming, and Kaiser ($\beta = 2$). What are the possible reasons of the different power levels of background noise, based on the frequency domain responses? Note that applying a window to the signal in time domain is equivalent to convolving the frequency domain of the window to the frequency domain signal.

5. [10 Points] In order to reduce the variance of the estimate, PSD can be averaged from the segments, as illustrated in Eq. (7). Write a function to segment the Problem 1 sequence to length 100, and compute the averaged periodogram with Hamming window using your function.

6. [10 Points] Write a new function that will apply a 50% overlap to increase the number of segments being averaged for PSD estimation in a finite sequence. Display this averaged modified periodogram for the sequence in Problem 1. You can check the result from your program with that from spectrum referring to the example code in Section 1.2.

7. [15 Points] Compare the raw periodogram and Welch's method with regard to the resolution and variance of the estimate. What are the advantages and disadvantages of Welch's method by windowing and averaging compared to the raw periodogram? Note, it may be helpful to plot all of the spectral estimates on a single plot so that comparison is easier.

8. [10 Points] We now consider about a signal collected from a refrigerator magnet will be used for PSD estimates. A refrigerator magnet has alternating north and south poles on the same surface of the plane, which provides twice the magnetism on one side and is thus more effective at keeping the large planar magnet uniformly stuck onto the steel refrigerator. This arrangement is called the Halbach array, as shown in Fig. 5 (from Wikipedia, http://en.wikipedia.org/wiki/Halbach_array). Assume that the read sensor measures the strength of the magnetic field along the vertical direction on the surface of the magnet and it yields a voltage value ranging from $+1$ to -1, proportional

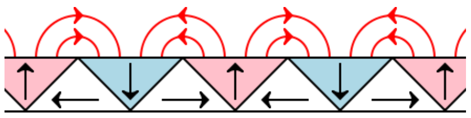

Figure 5: Halbach array (from Wikipedia)

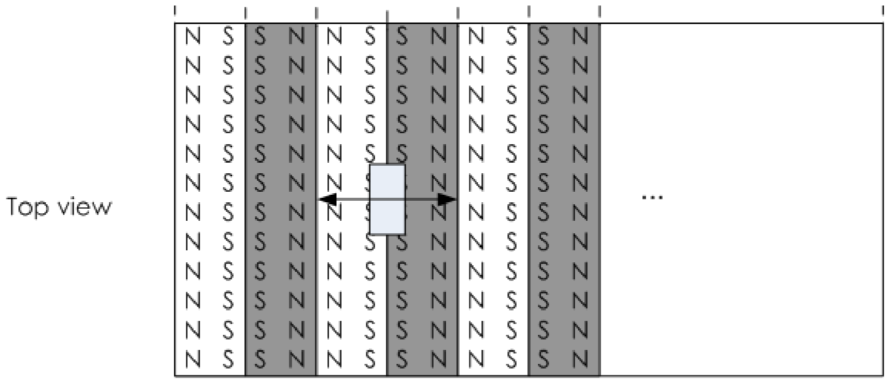

Figure 6: Refrigerator magnet scan

to the strength of the magnetic field.

A signal is collected by scanning the refrigerator magnet with the read sensor that measures the strength of the magnetic field, which is illustrated in Fig. 6. The measured signal reflects not only the periodic changes of the magnetic polarization of the magnet, but also random noise due to the surface roughness, scanning speed variation, sensor noise, etc. Load the signal `fridge_mag.mat` (on the course Blackboard), which contains t, time in seconds, and x, measured voltages for 1.6 seconds at a sampling frequency of 500 Hz. Plot the PSD estimate using the raw periodogram method described in Section 1.1.

9. [10 Points] Plot the PSD of the signal using Welch's method with a Hamming window, segments of 200 data points, and 50% overlap. Compare the results with the raw periodogram.

4 Week 2 milestones [100 Points]

1. [Pre-Lab, 15 Points] From the estimated PSD in the last problem of week 1, you can identify a dominant frequency in the signal, f_c, using the MATLAB command $[P_{max},\ i_{\hat{f}_c}]$ =`max`(P), where P is a PSD estimate and $i_{\hat{f}_c}$ is an index of corresponding

frequency. The measured signal can be represented by a sine wave corrupted by noise,

$$x[n] = A\sin(2\pi f_c(nT_s + \delta)) + r[n]$$

where A is an amplitude and $T_s = 1/f_s$(sampling frequency). δ represents the timing offset of the measured signal, $-T/2 < \delta \leq T/2$, where T is the period of the signal. For a segment of the signal of `fridge_mag.mat`, $t(51 : 251)$ and $x(51 : 251)$, find an offset δ that minimizes the mean square error between the measured signal $x[n]$ and an estimated signal $\hat{x}[n]$ for $A = 0.4$, as below,

$$\arg\min_{\delta} \Sigma_n (x[n] - \hat{x}[n])^2$$

where $\hat{x}[n] = A\sin(2\pi \hat{f}_c(nT_s + \delta))$ and $\hat{f}_c$ is a frequency corresponds to the $i_{\hat{f}_c}$ yielding maximum value in PSD estimate. You can try δ from $-T/2$ to $T/2$ in steps of 0.001, and select the one, $\hat{\delta}$, giving minimum mean square error between x and $\hat{x}$. Report $\hat{f}_c$ and $\hat{\delta}$ and plot $x[n]$ and $\hat{x}[n]$ together in a single plot.

2. [10 Points] Estimate the sample mean and variance of the noise in the measured signal, $r[n] = x[n]-\hat{x}[n]$. Use your estimate of δ for noiseless signal $\hat{x}[n]$. Plot the histogram of $r[n]$ for $-0.5 : 0.02 : 0.5$, by using `hist(r,-0.5:0.02:0.5)`. Report the sample mean and variance of r and compare the histogram to that of the AWGN with your sample mean and variance (plot your Gaussian curve on top of your histogram to compare the shape).

3. [15 Points] In this week, we will practice PSD estimates of the signal, which contains multiple frequency components, mainly focused on a signal having two frequency components (dual tone). A historical example of dual tone signal is a Dual Tone Multi-Frequency (DTMF) keypad in the telephone. DTMF is used for instructing a telephone switching system of the telephone number to dial, or to issue commands to switching systems or related telephony equipment. MATLAB has a demo for DTMF based on SIMULINK. You can type `dspdtmf` on your command window to launch the DTMF demo. This demo contains a DTMF tone generator, a channel, and a DTMF receiver. Review the help description of this demo, by clicking the $Info$ button on the lower left corner of the demo, to understand the fundamental principles of the DTMF keypad. Change the telephone numbers in a Tone Generator to '1-412-555-2345' and run a demo by clicking ▶ on the menu. You will see the detected number in the Display window with a spectrogram pop-up. Check the channel block (noise power and gain), and report them with a print screen image showing the DTMF SIMULINK model with the numbers you obtained.

4. [10 Points] Next, try to transmit keypad information under severe noisy conditions by increasing the channel noise power to 1 in a channel block (Gain 2) in the DTMF demo. The DTMF receiver fails to detect the numbers and displays all zeros. Try different channel noise power and find the one which gives several errors (not all errors). Report the channel noise power and the print screen image showing partial errors in the display window.

Table 1: The frequencies of the DTMF keypad

	1209 Hz	1336 Hz	1477 Hz
697 Hz	1	2	3
770 Hz	4	5	6
852 Hz	7	8	9
941 Hz	*	0	#

5. [10 Points] The DTMF keypad is laid out in a 4×3 matrix, with each row representing a low frequency, and each column representing a high frequency, as shown in Table 1. When any key is pressed, the tone of the column and the tone of the row are generated. As an example, pressing the '1' button generates the tones 697 Hz and 1209 Hz. The noisy signal of keypad '1' can be represented by the sum of the two tones; each can be represented by the cosine term of Eq. (6). Generate a signal when you press the '5' button with additive noise of $\mathcal{N}(0, 1)$. Use sampling frequency (f_s) of 10 kHz and acquisition time (T_a) of 0.02 sec and plot the noisy signal.

6. [10 Points] The spectrogram shown in the demo is a series of modified periodograms with a Hamming window and two averages per periodogram (Welch's method). Plot the PSD estimate of the previously generated signal of keypad '5' using Welch's method with a Hamming window, segment length of 50 and 50% of overlap. Make sure that the PSD shows two peaks (and their counterparts) at the designated frequencies of keypad '5'.

7. [10 Points] Using the MUSIC method, plot a pseudospectrum of the previously generated signal of keypad '5'. Use `spectrum.music` for this problem. Set the number of harmonics to 3 and segment length of 50 in your computation. Make sure that the frequencies of the peaks in the pseudospectrum correspond to those of '5' in the keypad.

8. [20 Points] Suppose that you are designing a modified DTMF keypad, where the frequency band for communication is reduced significantly compared to the conventional keypad. So, you are supposed to use the DTMF keypad with the frequencies listed in Table 2. Plot the signal of '5' button with the additive noise of same variance in Problem 5 (same sampling frequency and acquisition time as well).

   Plot PSD estimates of the new signal of '5' button using both the Welch's and MUSIC methods specified in the previous problems. Compare the results of the two methods and discuss the frequency resolution of these methods in the modified keypad example.

Table 2: The frequencies of the modified DTMF keypad

	1209 Hz	1241 Hz	1276 Hz
969 Hz	1	2	3
1006 Hz	4	5	6
1047 Hz	7	8	9
1091 Hz	*	0	#