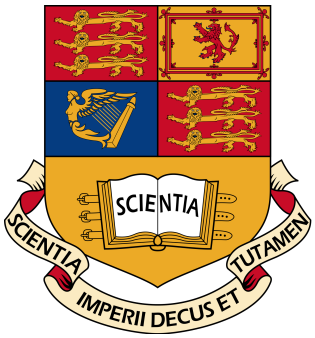
IMPERIAL COLLEGE LONDON

Advanced Signal Processing and Machine Intelligence

Author:
Pranav Malhotra
CID: 00823617
Email: pm1113@ic.ac.uk

Course Leader:
Prof. Danilo P. Mandic

March 24, 2017



Contents

1 Spectrum Estimation

1.1 Discrete Fourier Transform Basics

a. The ideal Fourier (magnitude) spectrum of a 20 Hz sine wave and the theoretical continuous frequency DTFT (magnitude) spectrum for a windowed sine wave at 20 Hz are shown below. Since a sine wave is a real periodic signal, we expect the ideal magnitude spectrum to be an even discrete function consisting of only dirac delta functions. The magnitude spectrum of the rectangular window used is a *sinc* function. The ideal magnitude spectrum is convoluted with the *sinc* function to produce the theoretical continuous magnitude spectrum; it is clear that the window has caused smearing and leaked the ideal spectrum.

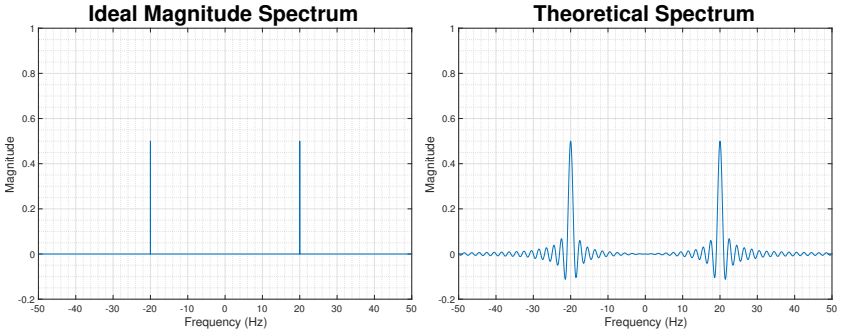


Figure 1: Ideal magnitude spectrum and Theoretical Continuous DTFT of 20 Hz sine wave

b. At first sight, it looks like the DFT spectrum on the left in Figure 2 is exactly the same as ideal spectrum of a 20 Hz sine wave. However, we know that this cannot be the case. The ideal spectrum is derived from a continuous infinite duration sine wave whereas the spectra presented below are both derived from finite duration discrete-time sine waves. In fact, both the spectra below are sampled versions of the theoretical continuous magnitude spectrum shown in Figure 1. The resemblance described above comes from the fact that **coherent sampling** was performed. The rectangular window used to limit is signal is 0.1 s long and thus the *sinc* function will have intercepts at integer interval of $f = \frac{1}{0.1} = 10Hz$. 100 samples are evenly divided across an axis that is 1000 Hz long and thus, each sample is separated by 10 Hz and exactly corresponds to frequencies at which the *sinc* function is 0, expect of course at ± 20 Hz. The spectral smearing and leakage artifacts that come about from time limiting the signal with a rectangular window become evident when more points from the continuous spectrum are sampled. This is clear by studying the graph on the right in Figure 2.

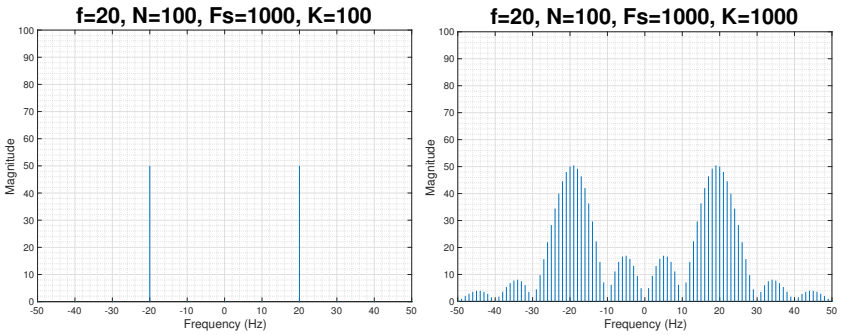


Figure 2: 100 and 100 Point DFT Spectra of 20 Hz Sine Wave

c. The graphs in Figure 3 show the K-point DFT spectra for $K = 100$ and $K = 2^{16}$; the graph on the right was generated using the `plot` function rather than the `stem` function since the theoretical spectra has been sampled so densely that it can be interpreted as effectively continuous. The graphs below illustrate the effects of sampling the continuous DTFT sparsely; the true peaks are not correctly identified. In fact, the peaks occur at bin $\frac{f_R}{f_s} = \frac{fxN}{f_s} = \frac{24*100}{1000} = 2.4$; since this bin does not exist in the 100-point DFT spectrum, the peak at 24 Hz cannot be identified. Increasing the value of K can be used to solve the incoherent sampling problem. This is easily obtained by zero-padding the signal.

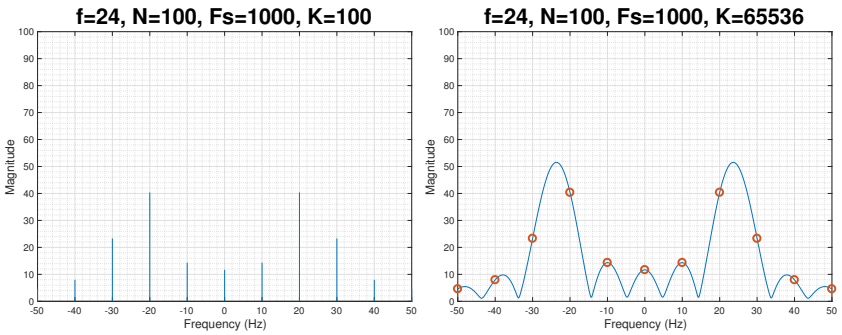


Figure 3: DFT Spectra Illustrating the Effects of Incoherent Sampling

1.2 Properties of Power Spectral Density (PSD)

Approximation in the definition of PSD

To show the equivalence of the two equations I shall start with (1).

$$\begin{aligned} P(\omega) &= \lim_{N\to\infty} E\left\{ \frac{1}{N} \left| \sum_{n=0}^{N-1} x(n)e^{-jn\omega} \right|^2 \right\} \\ &= \lim_{N\to\infty} E\left\{ \frac{1}{N} \sum_{m=0}^{N-1} x(m)e^{-jm\omega} \sum_{n=0}^{N-1} x^*(n)e^{jn\omega} \right\} \\ &= \lim_{N\to\infty} \frac{1}{N} \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} E\left\{ x(m)x^*(n) \right\} e^{-jm\omega}e^{jn\omega} \\ &= \lim_{N\to\infty} \frac{1}{N} \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} r_{xx}(m-n)e^{-j(m-n)\omega} \end{aligned} \tag{1}$$

Substituting $\tau = m - n$ and incorporating the double summation into one summation, we get:

$$\begin{aligned} P(\omega) &= \lim_{N\to\infty} \frac{1}{N} \sum_{\tau=-N+1}^{N-1} (N - |\tau|) r_{xx}(\tau) e^{-j\tau\omega} \\ &= \sum_{\tau=-\infty}^{\infty} r(\tau)e^{-j\tau\omega} - \lim_{N\to\infty} \frac{1}{N} \sum_{\tau=-N+1}^{N-1} |\tau| r(\tau) e^{j\tau\omega} \end{aligned}$$

And thus under the mild assumption that the covariance sequence $r(k)$ decays rapidly we have shown that (1) is equivalent to (2), which is simply the continuous DTFT of the Autocovariance Function (ACF).

$$P(\omega) = \sum_{\tau=-\infty}^{\infty} r(\tau)e^{-j\tau\omega} \tag{2}$$

a. The Fourier Transform (FT) of a real symmetrical signal will also be real and symmetrical; the symmetry in the FT is due to the signal being real whereas the fact that the imaginary component of the FT is zero is due to the symmetrical nature of the signal. Matlab's FFT algorithm assumes that the incoming signal is wrapped and thus the specific structure of $\mathbf{x}$, described in the coursework, is necessary. In contrast to the zero-padding performed in Section 1.1, here we pad zeros to the middle of the vector $\mathbf{x}$ to preserve symmetry. It is important to note that in this specific scenario, zero-padding is actually the same as extending the duration for which we observe the signal; this is because for $|k| \geq M$, the autocovariance function is 0. As such, zero-padding is increasing the frequency resolution of $P(\omega_k)$. In contrast, zero-padding is usually used to increase the number of points for the which the continuous DTFT is sampled and the frequency resolution depends solely on the size of the rectangular window used to restrict the signal from an infinite duration to a finite duration. The graphs below clearly illustrate this point. It is interesting to note that the autocovariance function described in the coursework is actually the ACF of the rectangular window. As M is increased from 10 to 128, the width of the mainlobe decreases; this is exactly what we expect. Smearing of the spectra is inversely proportional to the width of the rectangular window.

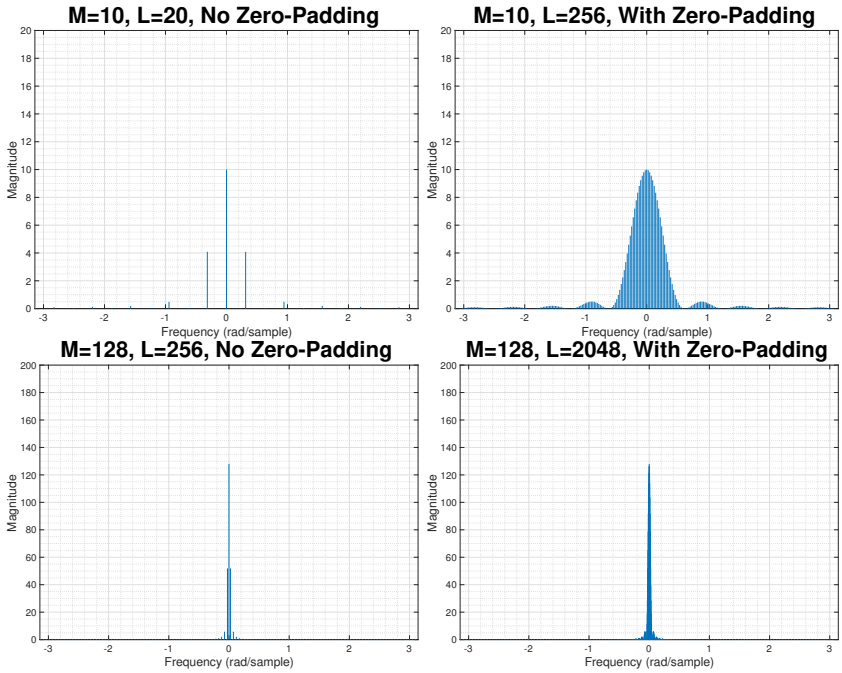


Figure 4: $P(\omega_k)$ of $r(k)$ Described in the Coursework with and without Zero-Padding for $M = 10$ and $M = 100$

b. Due to the symmetrical nature of the signal for which we are taking the FFT, we do not expect any imaginary components. The graph below shows that the imaginary components are extremely small and are about 15 orders of magnitude smaller than the real components. Above, the `abs` function was used to eliminate the imaginary components. Using the `real` function had no significant difference due to the extremely small values for the imaginary components.

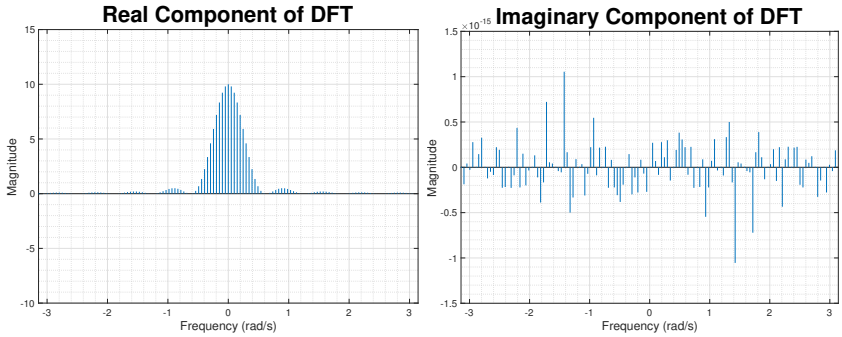


Figure 5: Comparing the magnitudes of the Real and Imaginary Components of $P(\omega_k)$

c. The graph below shows that in this case, the real and imaginary components of the DFT are of the same magnitude and thus using the `real` is not suitable.

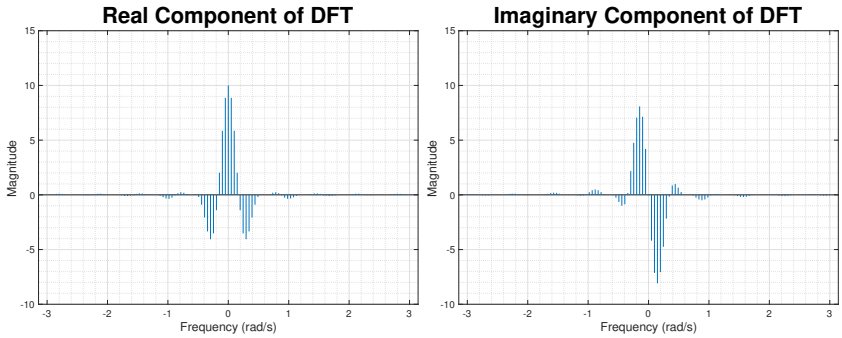


Figure 6: Magnitudes of the Real and Imaginary Components are of the Same Order of Magnitude

The correct Power Spectral Density can be obtained using the `abs` function in a similar fashion to part (a). This is shown in Figure 7.

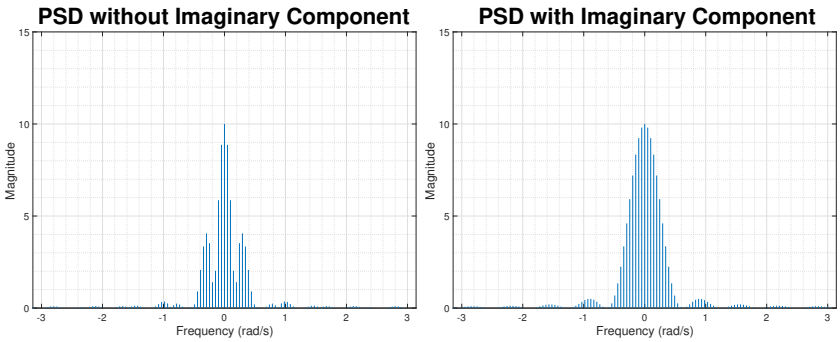


Figure 7: Erroneous Spectral Values because Imaginary and Real Components are Same Order of Magnitude

d. The vectors **w** and **n** do indeed depend on if the sequence is odd or even. The formula are described in the table below.

Sequence Type	Formula for w	Formula for n
Even	$-\pi : \frac{2\pi}{L} : \pi - \frac{2\pi}{L}$	$-\frac{L}{2} : 1 : \frac{L}{2} - 1$
Odd	$-\pi + \frac{\pi}{L} : \frac{2\pi}{L-1} : \pi - \frac{\pi}{L}$	$-\frac{L-1}{2} : 1 : \frac{L-1}{2}$

Table 1: Formule for vectors **w** and **n**

1.3 Resolution and Leakage of Periodogram-based Methods

Figure 8 shows the magnitude spectrum of the Bartlett Window for $N = 128$ and $N = 512$ on both linear and logarithmic scales. The red line is graphed to show the 3 dB frequency of each window.

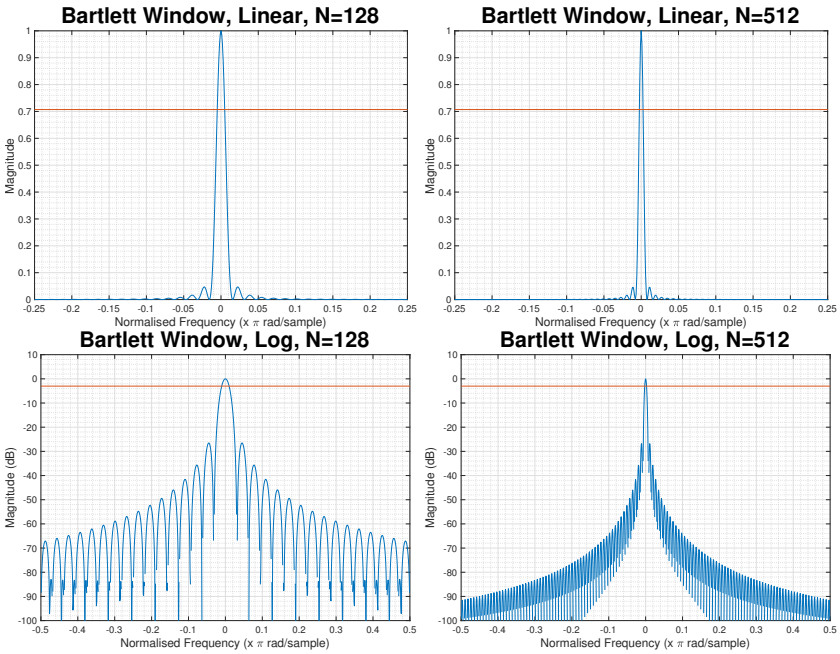


Figure 8: Bartlett Window with Linear and Log Scales for $N = 128$ and $N = 512$

To find the relationship between the width of the mainlobe and the length of the window, the 3 dB frequency is evaluated for a range of window lengths[1]. The figure below shows the relationship between the empirical 3 dB width of the main lobe of the Bartlett Window, ω_c, and both N and $\frac{1}{N}$.

[1]The intersection points were found using the function `InterX` found on matlab file exchange.

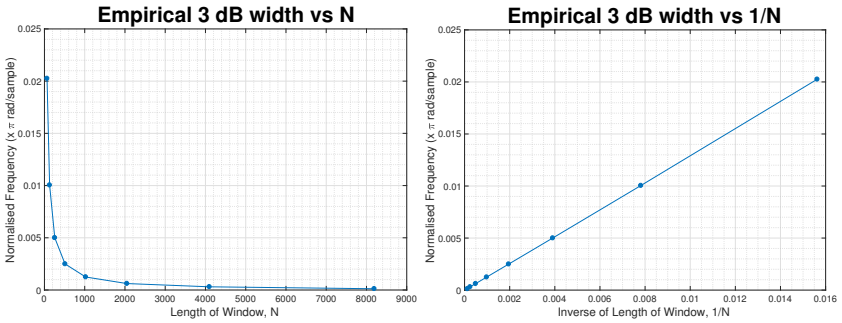


Figure 9: Empirical Relationship between 3 dB width of Mainlobe for Bartlett Window and Window Length

In matlab, a linear relationship between frequency and $1/N$ is obtained using `pinv`. The linear relationship is:

$$\omega_c = \frac{1.2965(2\pi)}{N}$$

The figure below shows the independence of the sidelobe peak to window length. The sidelobe peak is at a constant value of -26.5 dB relative to the height of the mainlobe.

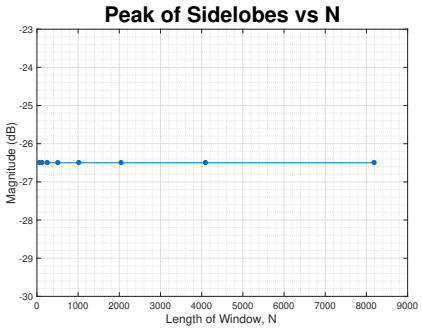


Figure 10: Empirical Relationship between Sidelobe Peak for Bartlett Window and Window Length

b. The signal $x(n)$ has been very carefully designed to study the frequency resolutions of different windows. This is because the frequency of one sine wave differs from the frequency of the other by a factor of $\frac{\alpha 2\pi}{N}$; The resolution of a spectral estimation method has the following general form described in (3). Thus, the signal $x(n)$ can be used to understand the scaling factor.

$$\text{Res}\left[\hat{P}_{per}\left(e^{j\omega}\right)\right] = c\frac{2\pi}{N} \tag{3}$$

The following graphs show the periodograms for $\alpha \in \{1, 0.65, 0.60\}$. It is clear that for values of $\alpha \leq 0.60$, the two peaks are not distinguishable. This value of alpha is significantly smaller than the value quoted in the lecture notes and in [1]. The value quoted in both pieces of literature take into account that the periodogram has some noise however we have removed noise and thus achieve greater resolution.

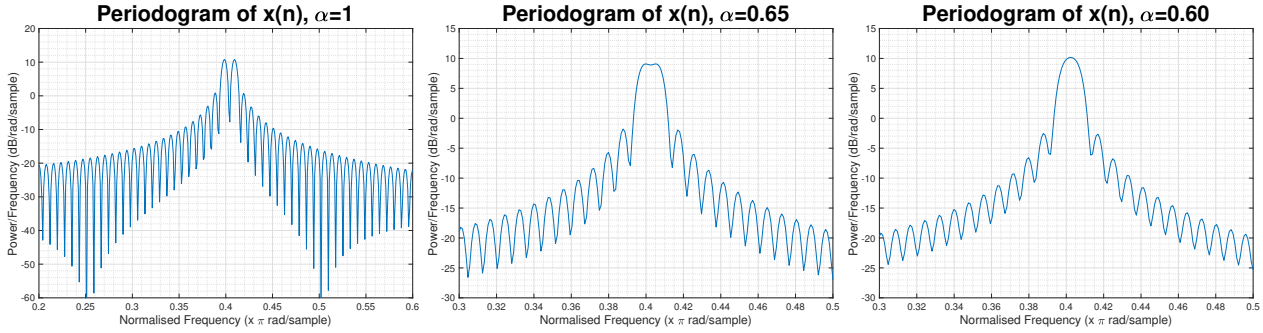


Figure 11: Periodogram of Signal $x(n)$ for Different Values of α

c. The hamming window has a wider mainlobe than the rectangular window. As such, the window smears the spectrum to a greater degree and thus the two frequencies are indistinguishable beyond $\alpha = 0.70$. Although hamming window has a wider mainlobe, its sidelobes are much lower and this is evident in the figure below.

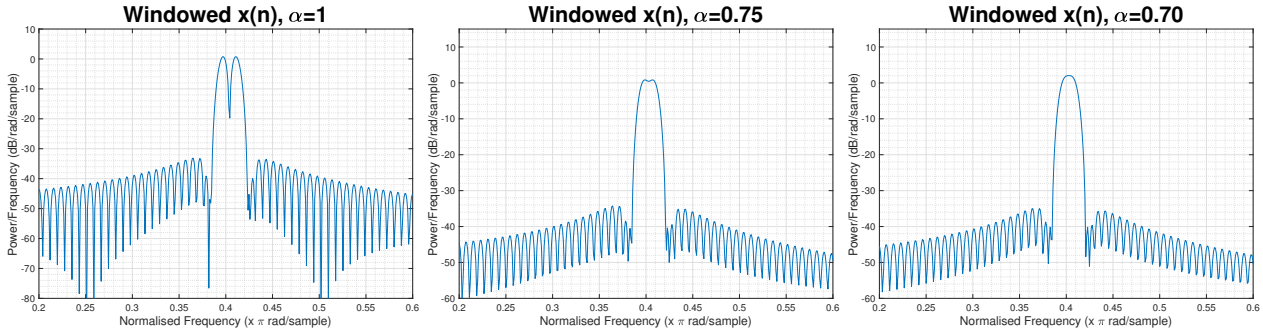


Figure 12: Periodogram of Signal $x(n)$, Windowed using a Hamming Window, for Different Values of α

d. Figure 13 shows the periodograms obtained using the rectangular window. All windows have to trade-off the width of the mainlobe and the relative heights of the sidelobes. The rectangular window lies at an extreme end of this spectrum in that it has the smallest mainlobe however it also has the highest sidelobes. This causes the least amount of smearing and bias but results in the most spectral leakage. As such, the ability to distinguish the second peak deteriorates significantly as a_2 decreases. The amplitude threshold identification of the second sinusoidal term was slightly easier for $\alpha = 12$ however it did not change significantly. The reason for this will be explained in the next part.

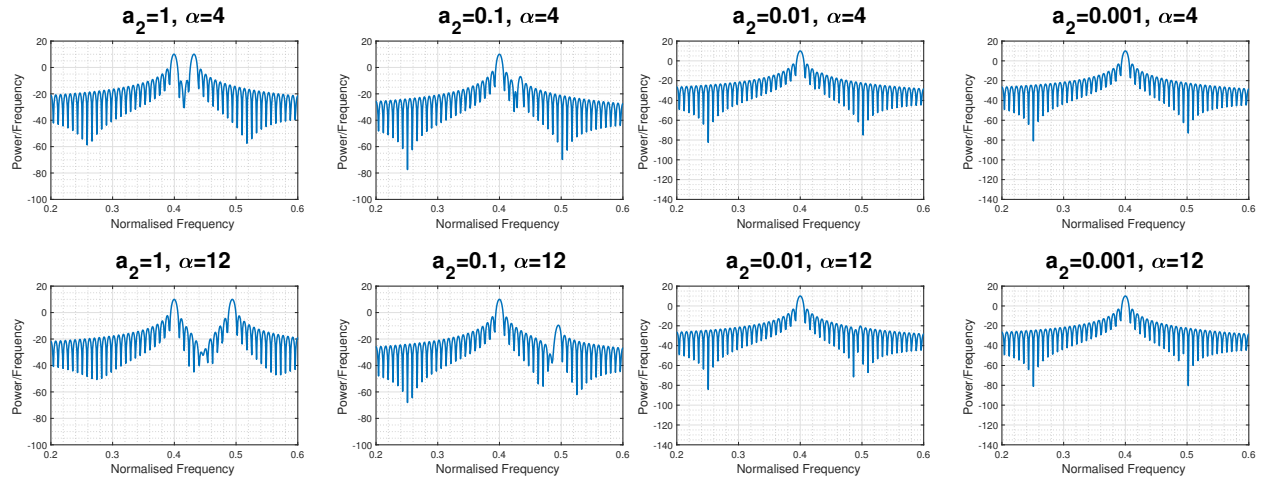


Figure 13: Identification of Sinusoids using Rectangular Window

e. A rectangular window with $N = 256$ can be thought of as a digital filter with 256 zeros; as such, it is able to fix the gain to be 0 at 256 points. At frequencies, $f = 4/N$ and $f = 12/N$ the gain of the filter is 0. However, the sidelobes of the rectangular window are not constant and they decrease as distance from the mainlobe increases. As such, the amplitude threshold identification is slightly easier at $\alpha = 12$ because the peak of sidelobes around $f = 12/N$ is slightly lower than the peak of sidelobes around $f = 4/N$.

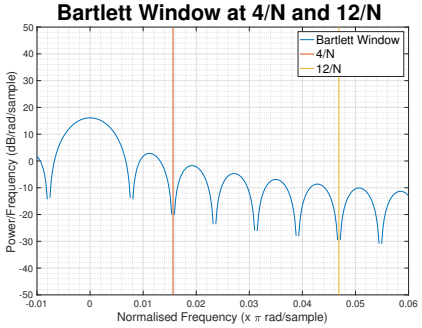


Figure 14: Power Spectral Density of the Bartlett Window at $f = \frac{4}{N}$ and $f = \frac{12}{N}$

f. As shown in the figure below, it is possible to resolve the two components even when $a_2 = 0.001$. Notice that when the Chebyshev window is used, the overlapping of the mainlobes causes trouble when trying to distinguish between two frequencies; this is clear when $\alpha = 4$. This was not the case when the rectangular window was used. Using the rectangular window, the height of sidelobes caused trouble when trying to distinguish between two frequencies. This represents the tradeoff that windows have to make between the width of the mainlobe as the height of the sidelobes. The rectangular window has a small mainlobe and thus the trouble in identification comes about because of the leakage effects whereas the Chebyshev window cause more smearing and less leakage.

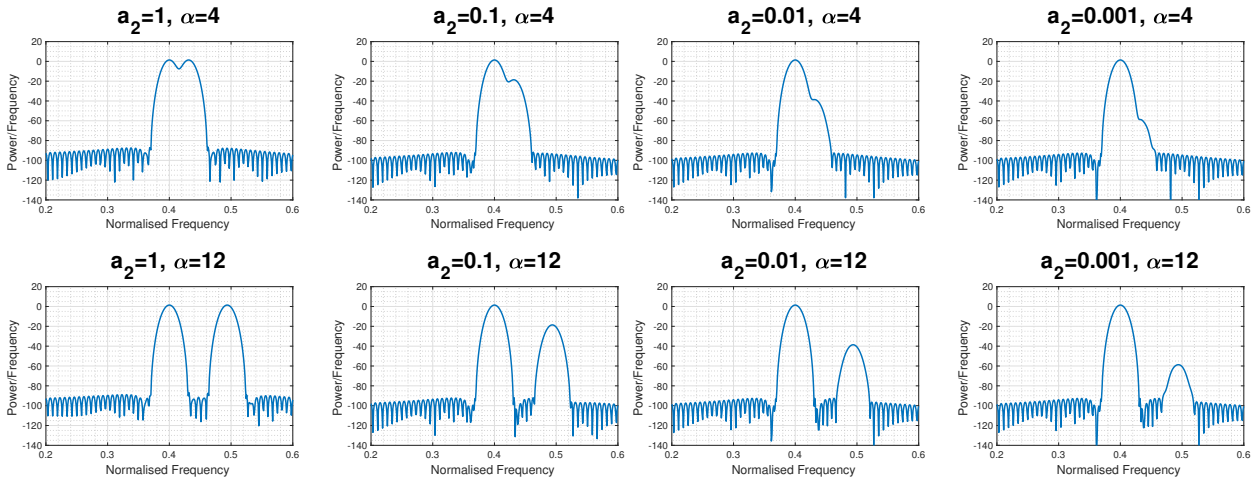


Figure 15: Identification of Sinusoids using Chebyshev Window

The Blackman-Tukey method was used to find the periodograms and the results are shown below. The autocovariance function is calculated but only 64 values are used; the tail end of the autocovariance is not reliable as they were calculated by averaging very few points and thus they are excluded when the Power Spectral Density is calculated. The results are not promising and the Blackman-Tukey method performs poorly compared to the Chebyshev window. However, this is completely expected. The Blackman-Tukey method is used to reduce the variance of the periodogram estimate by reducing the variance in the autocovariance estimate. A reduction is variance is traded-off for resolution; notice the number of troughs between the two peaks has reduced by a factor of 4 when the Blackman-Tukey estimates are compared to the periodograms in Figure 13. This is expected since we have used $\frac{1}{4}$ the number of points to calculate the DFT. In this illustrative example, we have removed noise and thus the periodograms are determinstic and using the Blackman-Tukey method does not actually remove any noise variance, it just reduces the resolution and thus depreciates our ability to distinguish between two frequencies.

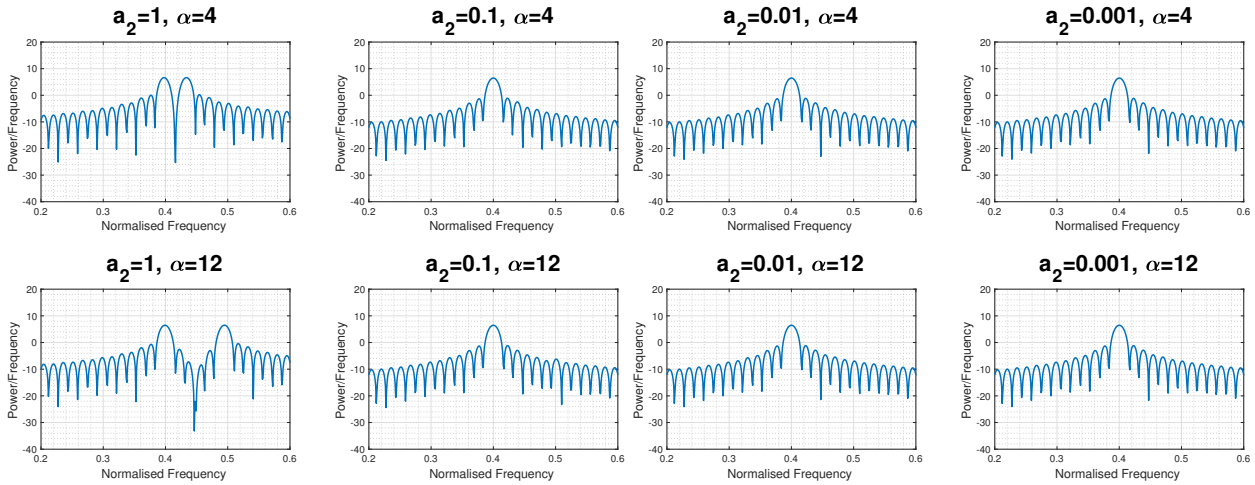


Figure 16: Blackman-Tukey Method for Identification of Two Sinusoidal Tones

1.4 Periodogram-based Methods Applied to Real-World Data

a. The figure below shows the periodogram, with and without pre-processing, for the sunspot time series. It is evident that the periodograms produced by removing the mean and removing the trend are very similar. These two periodograms are also extremely similar to the original periodogram beyond $f_n = 0.2$.

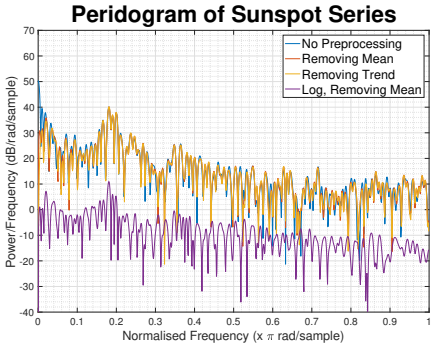


Figure 17: Standard Periodogram, with and without Preprocessing, for Sunspot Time Series

The original periodogram has a DC component that causes the periodogram to have a spike at $f = 0$ of approximately 50 dB. The removal of the mean removes the the spectral components close to $f = 0$. The mean can be visualized as a rectangular signal that has been added to a zero-mean signal. Through the linearity of the Fourier Transform and the fact that the spectrum of a rectangular signal is a *sinc*, the spectrum of the mean-removed signal only differs from the original signal at low frequencies. The `detrend` function removes linear trends. The Fourier Transform of linear functions is a spike at $f = 0$ and a magnitude proportional to $\frac{1}{f^2}$ for $f \neq 0$. As such, using `detrend` prior to formulating the periodogram also only alters low frequency components.

Note that before we are able to take logarithms of the time-series, a small offset of 0.001 was added to the entire series. This allowed the logarithms to be taken because the sunspot series has values of 0 at certain points. Since this offset is added to all the terms, it too was removed when the mean of the series was subtracted. The shape of the periodogram obtained is similar to the periodogram obtained solely by removing the mean and tread. The main difference is in the fact that spikes have been greatly accentuated.

b. The standard periodogram of the EEG obtained from an electrode located at the POz region of the head is graphed in the figure below. **The peaks in the spectrum corresponding to the SSVEP are clearly observed at $f = 13, 26, 39, 52$.** As expected, the fundamental frequency response peak is the largest whereas the height of the harmonics is monotonically decreasing.

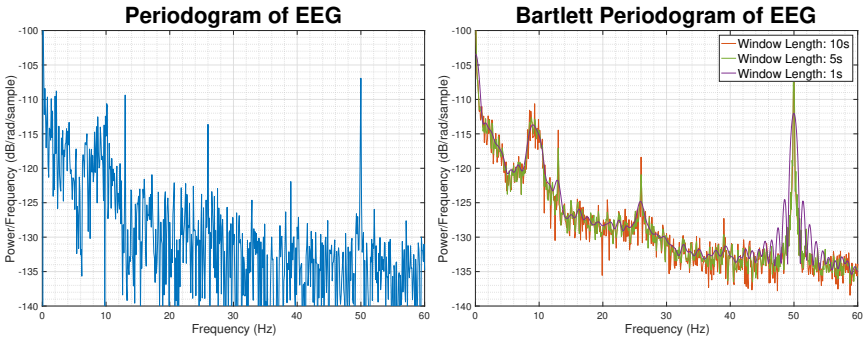


Figure 18: Standard Periodogram and Bartlett Periodogram with Hamming Window for EEG samples Obtained from POz Location on the Scalp

The figure also shows the Bartlett periodograms obtained by averaging the signal over different window lengths (10 s, 5 s, 1 s). As the duration of the observation window becomes smaller, the variance of the periodogram decreases. A smaller observation window means that the number of periodograms obtained increases; there are a greater number of periodograms to average over and thus the variance decreases. However, using a shorter observation window means that the frequency resolution of the periodogram has decreased. The damning effect of decreasing the frequency resolution is evident when the periodogram obtained from by averaging over 1 s windows is studied; the peridogram has been graphed again in Figure 19 to emphasize the difficulty in

identifying SSVEP peaks. The large alpha-rhythm present in the EEG have masked the distinct peak that occurs at 13 Hz. This makes identifying the fundamental frequency response peak much harder. Notice that the first harmonic at $f_n = 26$ Hz is still visible. This is due to the fact that there are no large frequency components around $f = 26$ Hz that can cause spectral leakage and thus mask the harmonic. Note that the extremely large frequency component at 50 Hz has completely masked the 3rd harmonic that is visible in the original, unaveraged periodogram. The component at 50 Hz is so strong that the shape of the power spectrum of rectangular window used is evident in Figure 19.

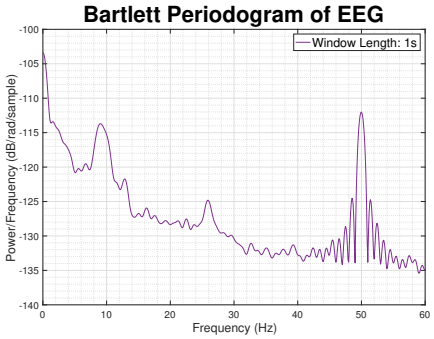


Figure 19: Bartlett Periodogram with Hamming Window averaged over 1 s intervals

This shows the clear trade off between frequency resolution and variance reduction that has to be made when averaging periodograms.

2 Parametric and Line Spectra

2.1 Correlation Estimate

a. Figure 20 shows the biased and unbiased estimates of the autocorrelation function (ACF) as well as the correlogram spectral estimates obtained for 3 signals: white gaussian noise, a sinewave and filtered white gaussian noise.

It is clear that the ACF estimates are similar at small lags; however the similarity fades as the lag increases. The biased estimator function imposes a Bartlett window on the ideal ACF whereas the unbiased estimator imposes a rectangular window on the ideal ACF. The choice of window is extremely important and the devastating effects of using a wrong window can be understood by studying the correlograms. The spectral estimate obtained from the unbiased ACF estimate returns values of power that are negative; this is obviously not possible.

This problem can also be understood in terms of the time-domain windowing that is performed prior to computing the spectral estimate. The biased estimator of ACF is obtained by windowing the time-domian signal using rectangular window. When we calculate the periodogram using the time-domain signal, the magnitude spectrum of the window is squared; a rectangular window ensures positive semidefiniteness of the resulting power spectrum. The same cannot be said about the window that is applied to the time-domain signal to obtain an unbiased estimate of ACF; as observed, the window does not preserve positive semidefiniteness of the spectral estimate.

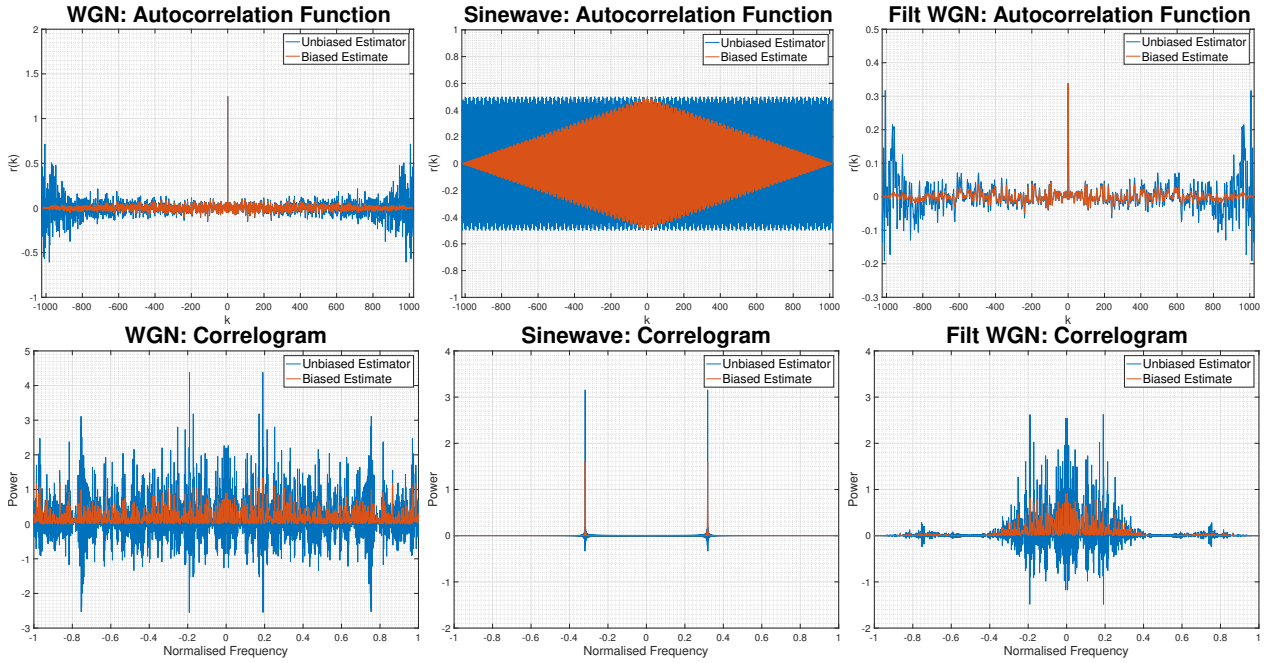
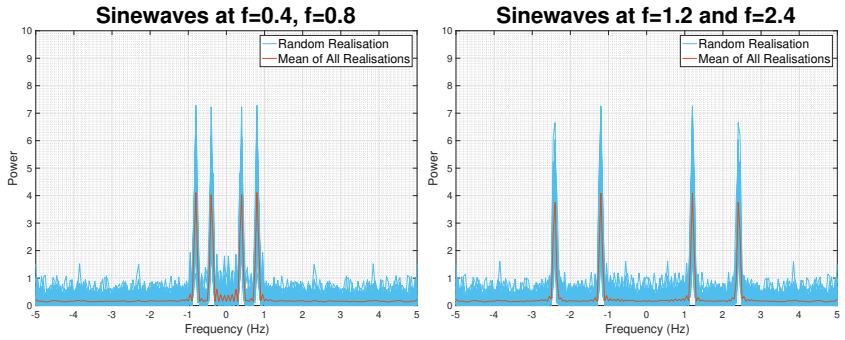


Figure 20: Biased and Unbiased Estimates of ACF and their Effect on the Correlogram

b. The figures below show 100 realizations of 2 unique random processes. The mean of the 100 realizations has been graphed in red. It is clear from observing the plots that the each random process consists of 2 sinewaves.



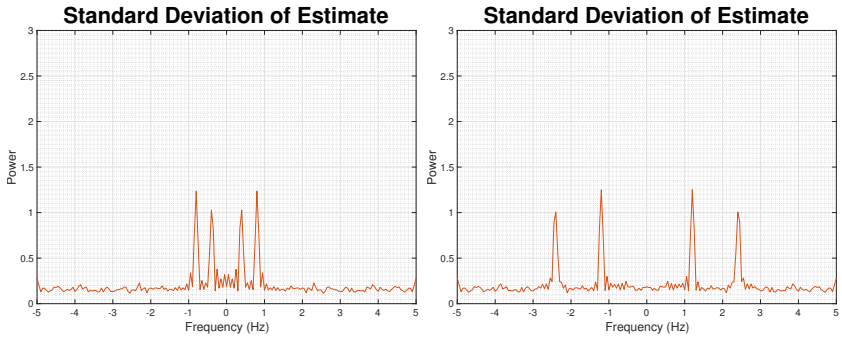


Figure 21: Variance and Standard Deviation of Correlogram Spectral Estimates

The figure above also includes the standard deviation of the 100 spectral estimates. Notice that there are peaks in the standard deviation at exactly the same frequencies as the peaks in the periodogram estimates. This is indicative of one of the short-comings of the periodogram spectral estimator. **The periodogram is not a consistent estimator.** For the periodogram to be consistent estimator, $\text{var}(P_{per}) \to 0$ as $N \to \infty$. The peridogram does not satisfy that condition but instead, $\text{var}(P_{per}) \to P_x^2(f)$ as $N \to \infty$. This is clearly depicted in the plots of standard deviation below; it spikes at $f = 0.4, 0.8$ Hz and $f = 1.2, 2.4$ Hz. Thus, simply collecting more samples and then calculating the periodogram will not reduce variance. Other spectral estimation techniques such as the Blackman-Tukey method must be considered if variance is to be reduced.

c. Notice that when the graphs are plotted in the dB, the effects described above are made even more obvious. It is interesting to note that the effects of windowing the sinewave are now also clear. Notice the $sinc^2$-like shape near the peaks at $f = 1.2, 2.4$ Hz.

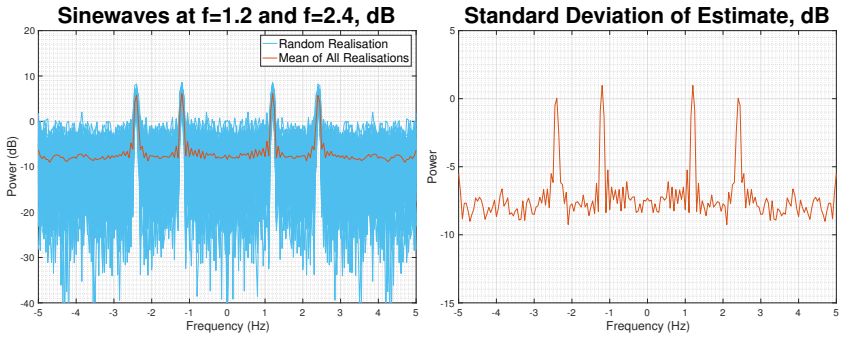


Figure 22: Effect of dB Scale on Spread of Variance and Standard Deviation of Correlogram Spectral Estimates

d. Since the frequency resolution of the periodogram is proportional to $\frac{1}{N}$, we expect to be able to distinguish 2 complex exponentials with frequencies of 0.3 Hz and 0.32 Hz with approximately 50 samples; with 50 samples, we will have a resolution of approximately 0.02 Hz. The figures below confirm this. As the number of samples increases, the ability to distinguish the peaks increases. Note that the periodograms are no longer symmetric. This is expected since the functions are complex. Due to their non-symmetric nature, the one-sided periodogram is plotted up to f_s instead of $\frac{f_s}{2}$.

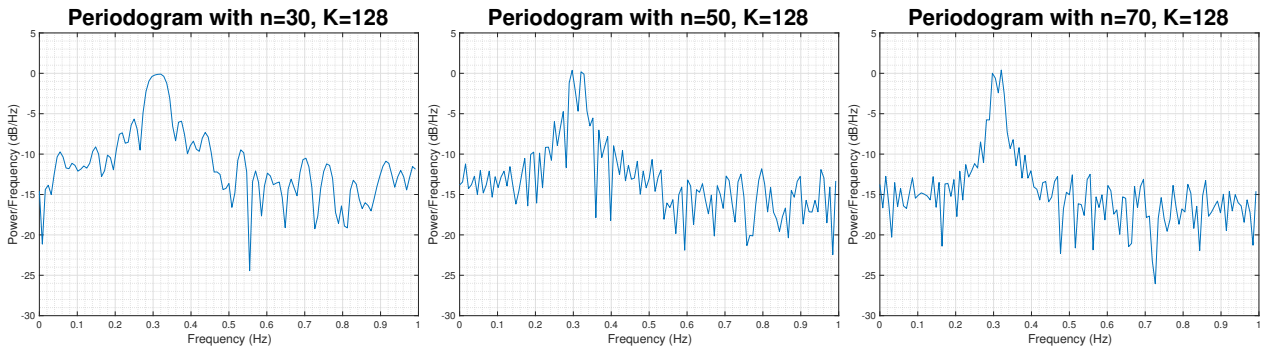


Figure 23: Effect of Increasing Number of Samples on the Resolution of Periodogram Spectral Estimates

e. The MUltiple SIgnal Classification method (MUSIC) is a frequency estimation technique that models the ACF matrix $\mathbf{R}_{xx}$ as the sum of 2 ACF matrices, namely the signal ACF matrix $\mathbf{R}_s$ and the noise ACF matrix $\mathbf{R}_n$. A specific scenario in which the signal only contains 1 complex exponential will have the following form:

$$x(n) = A_1 e^{jn\omega_1} + w(n) \tag{4}$$

Since the matrix $\mathbf{R}_{xx}$ is Hermitian, the eigenvectors of the matrix are orthogonal. As such, the following equation is true, where $\mathbf{e}_i$ is a eigenvector from matrix $\mathbf{R}_s$ whereas $\mathbf{v}$ is an eigenvector from $\mathbf{R}_n$ and $\mathbf{R}_{xx} \in \mathbb{R}^{M\times M}$ and there are p complex exponentials present in the signal:

$$\mathbf{e}_i^H \mathbf{v} = \sum_{k=0}^{M-1} v(k) e^{-jk\omega_i} = 0, \;\; i = 1, 2, \ldots, p \tag{5}$$

The power spectrum can be estimated using the following formula, where $\mathbf{e} = [1, e^{j\omega}, e^{j2\omega}, \ldots, e^{j(M-1)\omega}]$. Note that a discrete-time signal can contain only a finite set of unique complex exponentials that are all contained within the vector $\mathbf{e}$. This is due to the fact that the complex exponentials are periodic and this forms the basis of the Discrete Fourier Transform:

$$\hat{P}_{MU}(e^{j\omega}) = \frac{1}{\sum_{i=p+1}^{M} |\mathbf{e}^H v_i|^2} \tag{6}$$

Since the eigenvectors $\mathbf{e}$ and v_i are orthogonal, the inner product should have p roots which lie on the unit circle and correspond to the frequencies of the complex exponentials that are present in the signal. However, since $\mathbf{R}_{xx} \in \mathbb{R}^{M\times M}$ where $M > p+1$, then the inner product will have $M-p-1$ zeros which lie anywhere on the z-plane. In fact, the zeroes could lie very close to the unit circle and thus lead to suprious peaks in the power spectrum that could be mistaken as a complex exponentials that is present within the signal. To reduce the likelihood of this occuring, we average the estimate using all of the $M-p-1$ eigenvectors corresponding to the noise. The averaging effect will amplify the magnitude of the true peaks and thus makes identifying complex exponentials that are actually present within our signal easy.

Having understood how the algorithm works, it is clear that the first line gives the ACF matrix $\mathbf{R}_{xx} \in \mathbb{R}^{M\times M}$, where $M = 14$. This value of M is selected based on the fact that we can only trust that many ACF values. Larger values of M could be selected however, as the lag increases, the accuracy with which ACF estimation can be performed decreases; poor ACF estimation at large lags has been illustrated above. The second line of the code performs the spectral estimation using the MUSIC algorithm. The ACF matrix is provided as the first argument while the dimension of the signal subspace p, is given as the second argument. The third and fourth arguments are related to the length of the FFT and the sampling period while the last argument informs the function to treat the first argument as a correlation matrix rather than a data matrix.

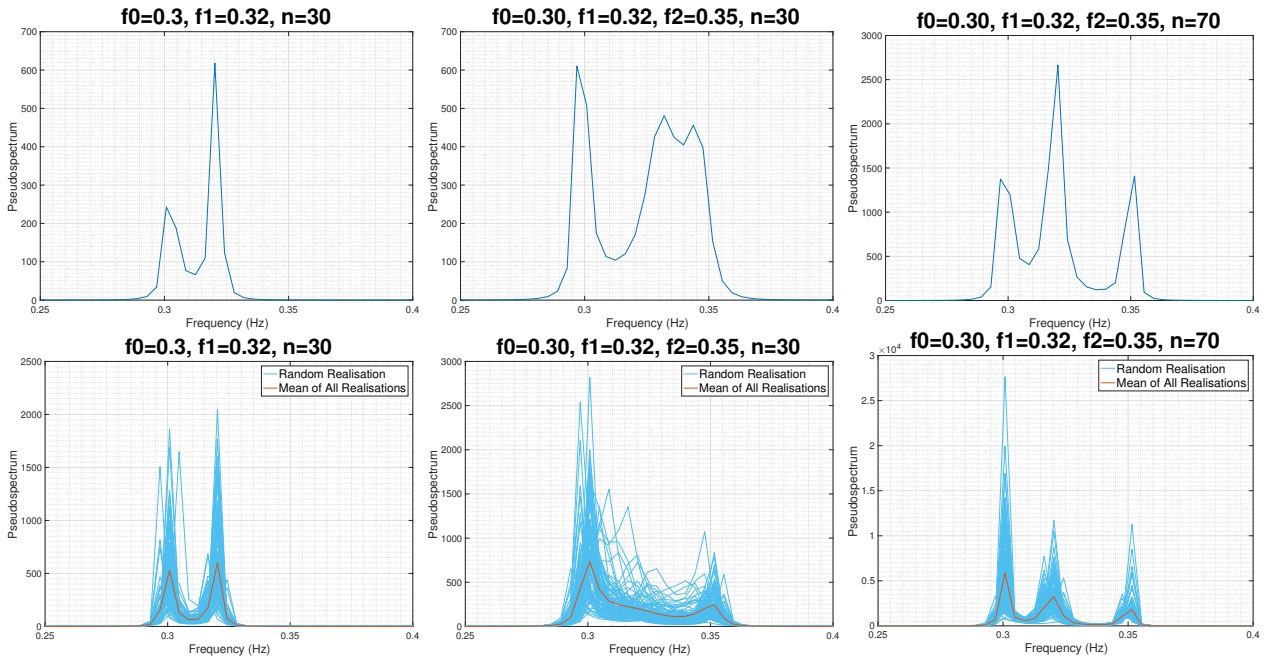


Figure 24: Detecting Sinewaves using MUSIC, and Effect of Increasing Number of Samples on the Resolution of Estimate

The figures above shows the results obtained. The first row shows just one realization of the random signal x, whereas the second row shows an overlay of 100 realizations. It is clear that the MUSIC algorithm, like the periodogram has a frequency resolution limit as it is not able to identify 3 sinewaves that are very closely spaced together. However increasing the number of samples, like for the periodogram, increases the frequency resolution and distinguishing the sinewaves is now possible.

The main advantage of the MUSIC algorithm is the ability to identify complex exponentials when very few samples are available. This is clear when comparing the results obtained in Figure 24 to those obtained in Figure 23. With just 30 samples, the MUSIC algorithm is able to correctly identify the location of the two peaks, while the periodogram is not.

The disadvantages are that the pseudospectrum does not provide information about the magnitude of the complex exponentials. Notice that the signal x was generated with equal magnitudes for each of the complex exponentials but the pseudospectrum did not highlight this. Secondly, the algorithm requires, as an input, the dimension of the signal subspace. Often, this may not be known. Lastly, the averaging that is performed to remove spurious peaks from the spectra have the adverse effect of removing information about the noise spectra; this information may be useful, especially if the noise is non-white.

2.2 Spectrum of Autoregrssive Processes

a. Autoregressive (AR) parameter estimation is performed using the following equation:

$$\mathbf{r}_{xx} = \mathbf{R}_{xx} a \implies a = \mathbf{R}_{xx}^{-1} \mathbf{r}_{xx}$$

It is crucial that the matrix $\mathbf{R}_{xx}$ be invertible. When the ACF is calculated using the biased estimator, $\mathbf{R}_{xx}$ is guaranteed to be positive definite and thus invertible, however this is not gauranteed when the unbiased estimator is utilized.

b. Figure 25 shows the spectrum obtained when the signal $x(n)$ is modelled as a 4th, 8th, 10th and 14th order AR process. The effects of increasing the model order from 4 to 8 are clear. The 4th order process is not able to model the 2 distinct spikes that are present in the ideal spectrum, whereas the 8th order process is. Arbitrarily increasing the model order beyond 8 allows more degrees of freedom since there is greater flexibility to place poles in the spectrum. However, the resulting spectrum is not more accurate. In fact, increasing the number of poles causes both the peaks to approach the same magnitude; this is obviously not correct. This highlights the fact that the 500 samples used for estimation may not contain enough information about the AR process for effective parameter estimation.

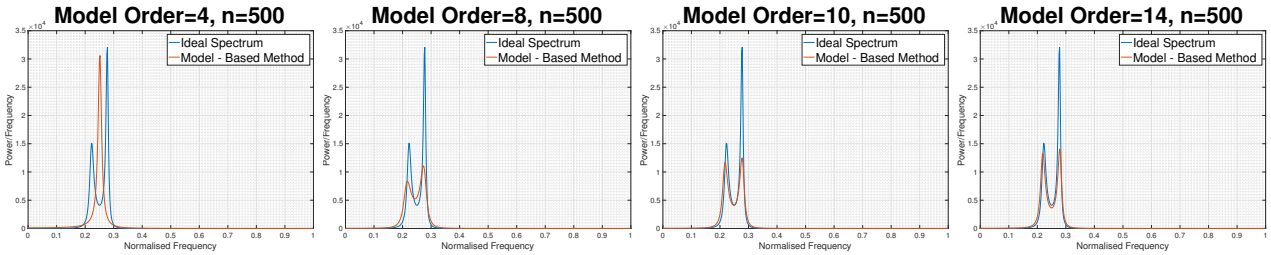


Figure 25: Power Spectrum Estimates Using All-Pole Autoregressive Model of Orders 4, 8, 10, 14

c. Figure 26 shows that by increasing the number of samples, we obtain a much more accurate spectrum estimate. The conjecture above about 500 samples not containing enough information about the AR process is correct. Notice that with 10000 samples, a 4th order estimate is able to replicate the two spikes observed in the data. Increasing the model order to 12 does not cause both peaks to approach the same magnitude.

However, by setting the model order to 12, overfitting of the training data starts to occur. The peaks of the estimated spectrum have overshot the peak of the ideal spectrum. The power of the signal has been compressed into a small frequency range and thus made the signal highly periodic. However, by increasing the model order, the estimation error has been reduced. To balance the trade-off between generalization error and estimation error, measures such as the Akaike Information Criterion (AIC) or the Minimum Description Length (MDL) can be used to simultaneously penalize high model orders and estimation errors.

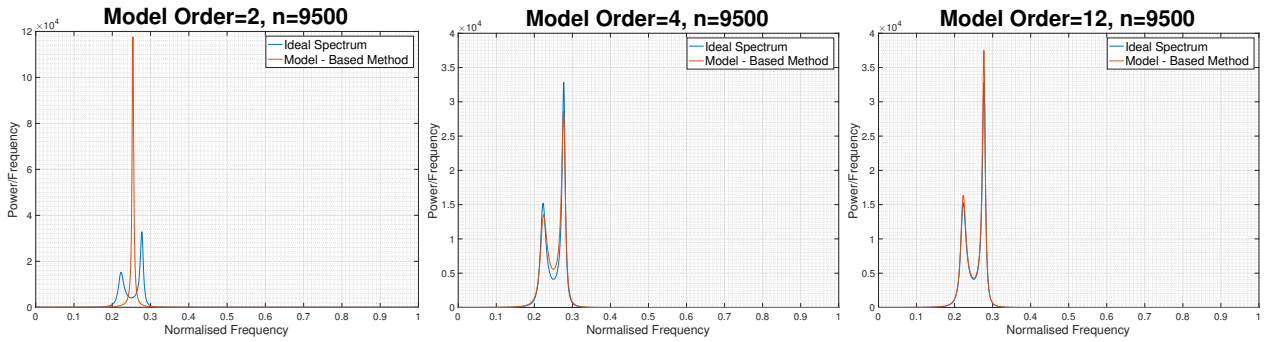


Figure 26: Effects of Underfitting and Overfitting Autoregressive Models

2.3 Principal Component Analysis

a. By studying the graphs below, it is clear that the training data $\mathbf{X}$ has a rank of 3; $\mathbf{X}$ only has 3 non-zero singular values. Notice that the matrix $\mathbf{X}_{\text{noise}}$ also only has 3 dominant singular values. The magnitude of the singular values that come about from the effects of noise have roughly half the magnitude of the original dominant singular values. Note that the noise has not only introduced additional non-zero singular values, but it has only increased the magnitude of the original singular values that are due to the signal.

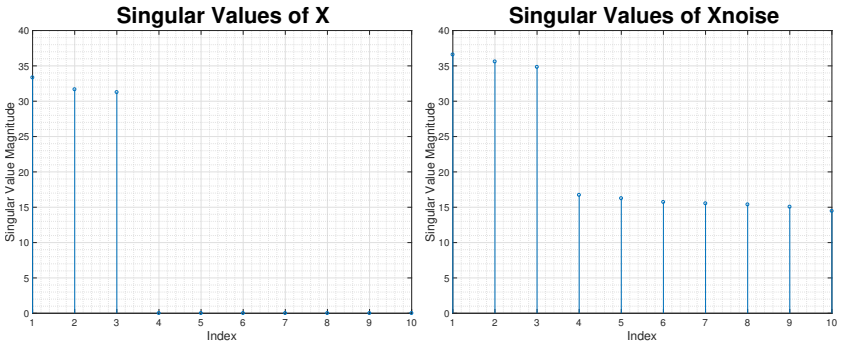


Figure 27: Singular Values of Matrices $\mathbf{X}$ and $\mathbf{X}_{\text{noise}}$

Next notice the squared error between each singular value of $\mathbf{X}$ and $\mathbf{X}_{\text{noise}}$. It is clear that the noise has slightly offset the original 3 singular values but the effects the noise has on the previously zero singular values is much more pronounced. It would be difficult to identify the rank of $\mathbf{X}_{\text{noise}}$ if the noise had increased the previously zero singular values to reach similar magnitudes as the original singular values.

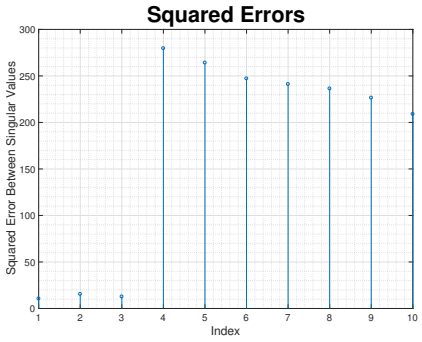


Figure 28: Squared Error between Each Singular Value of $\mathbf{X}$ and $\mathbf{X}_{\text{noise}}$

b. The basic low rank approximation $\tilde{\mathbf{X}}_{\text{noise}}$ problem can be formulated as the following optimization problem:

$$\begin{aligned} \underset{\tilde{\mathbf{X}}_{\text{noise}}}{\text{minimize}} \quad & ||\mathbf{X}_{\text{noise}} - \tilde{\mathbf{X}}_{\text{noise}}||_{\text{F}} \\ \text{subject to} \quad & \text{rank}\big(\tilde{\mathbf{X}}_{\text{noise}}\big) \leq r \end{aligned}$$

The above optimization problem admits as closed form solution in accordance with the Eckart-Young-Mirsky theorem; the theorem tells us to compute the approximation using only the r most significant principal components. The theorem also tells us that the Frobenius norm squared of $\mathbf{X}_{\text{noise}} - \tilde{\mathbf{X}}_{\text{noise},r}$ is equal to the squared sums of the singular values that we are "thrown" away while making the approximation.

$$||\mathbf{X}_{\text{noise}} - \tilde{\mathbf{X}}_{\text{noise},r}||_{\text{F}}^2 = \sum_{i=r+1}^{n} \sigma_i^2$$

The low-rank approximation is performed and the results obtained are presented in the table below:

	Frobenius Norm Squared
$\|\|\mathbf{X} - \mathbf{X}_{\text{noise}}\|\|_{\text{F}}^2$	2434.624
$\|\|\mathbf{X} - \tilde{\mathbf{X}}_{\text{noise}}\|\|_{\text{F}}^2$	732.789
$\|\|\mathbf{X}_{\text{noise}} - \tilde{\mathbf{X}}_{\text{noise}}\|\|_{\text{F}}^2$	1703.893

Note that the amount of energy that was "thrown" away during the low-rank approximation, 1703.893, is exactly equal to the improvement observed in $||\mathbf{X} - \tilde{\mathbf{X}}_{\text{noise}}||_{\text{F}}^2$.

c. $\hat{\mathbf{B}}_{\text{OLS}}$ and $\hat{\mathbf{B}}_{\text{PCR}}$ are calculated using the equations (7) and (8).

$$\hat{\mathbf{B}}_{\text{OLS}} = \left(\mathbf{X}_{\text{noise}}^{\text{T}}\mathbf{X}_{\text{noise}}\right)^{-1}\mathbf{X}_{\text{noise}}^{\text{T}}\mathbf{Y} \tag{7}$$

$$\hat{\mathbf{B}}_{\text{PC}} = \mathbf{V}_{1:r}\left(\mathbf{\Sigma}_{1:r}\right)^{-1}\mathbf{U}_{1:r}^{\text{T}}\mathbf{Y} \tag{8}$$

Note that it is critical that $\hat{\mathbf{B}}_{\text{OLS}}$ is calculated using $\mathbf{X}_{\text{noise}}$ and not $\mathbf{X}$. Matlab throws a warning if $\mathbf{X}$ is used; however this is expected because $\mathbf{X}$ is sub-rank. Next, using (9) and (10), estimation of $\mathbf{Y}$ is performed.

$$\hat{\mathbf{Y}}_{\text{OLS}} = \mathbf{X}_{\text{noise}}\hat{\mathbf{B}}_{\text{OLS}} \tag{9}$$

$$\hat{\mathbf{Y}}_{\text{PCR}} = \tilde{\mathbf{X}}_{\text{noise}}\hat{\mathbf{B}}_{\text{PCR}} \tag{10}$$

The estimation errors are presented in the table below. From this data alone, it is not clear that Principal Component Regression (PCR) scheme is any better than the Ordinary Least Squares (OLS) scheme. In fact, based purely on the training error obtained, one might conclude that OLS is a better estimation scheme.

	Euclidean Norm Squared
$\|\|\mathbf{Y} - \hat{\mathbf{Y}}_{\text{OLS}}\|\|_2^2$	3551
$\|\|\mathbf{Y} - \hat{\mathbf{Y}}_{\text{PCR}}\|\|_2^2$	3577

For a fair comparison of the PCR and OLS schemes, the estimates $\hat{\mathbf{B}}_{\text{OLS}}$ and $\hat{\mathbf{B}}_{\text{PCR}}$ must be evaluated on test data. Evaluation on test data was performed and the results obtained are presented in the table below. The results are telling of the fact that PCR regression has produced a better estimate $\hat{\mathbf{B}}_{\text{PCR}}$. While the OLS scheme is able to fit the training data well, when evaluated on test data, the scheme under performs the PCR scheme.

	Euclidean Norm Squared
$\|\|\mathbf{Y}_{\text{test}} - \hat{\mathbf{Y}}_{\text{test-OLS}}\|\|_2^2$	2374
$\|\|\mathbf{Y}_{\text{test}} - \hat{\mathbf{Y}}_{\text{test-PCR}}\|\|_2^2$	2354

d. Lastly, the estimates $\hat{\mathbf{B}}_{\text{OLS}}$ and $\hat{\mathbf{B}}_{\text{PCR}}$ are evaluated over 100 randomly generated testing datasets. The results obtained for presented in the table below:

	Euclidean Norm Squared
$\mathbb{E}\{\|\|\mathbf{Y} - \hat{\mathbf{Y}}_{\text{OLS}}\|\|_2^2\}$	2351
$\mathbb{E}\{\|\|\mathbf{Y} - \hat{\mathbf{Y}}_{\text{PCR}}\|\|_2^2\}$	2328

It is clear that $\hat{\mathbf{B}}_{\text{PCR}}$ is a much better estimate of $\mathbf{B}$. The results obtained by averaging over 100 randomly generated test datasets corroborate the results obtained in part c. The PCR scheme performs better on test data.

2.4 Real World Signals: Respiratory Sinus Arrhythmia from RR-Intervals

a. The figure below shows the standard periodogram obtained for each of the three unique sets of RRI data.

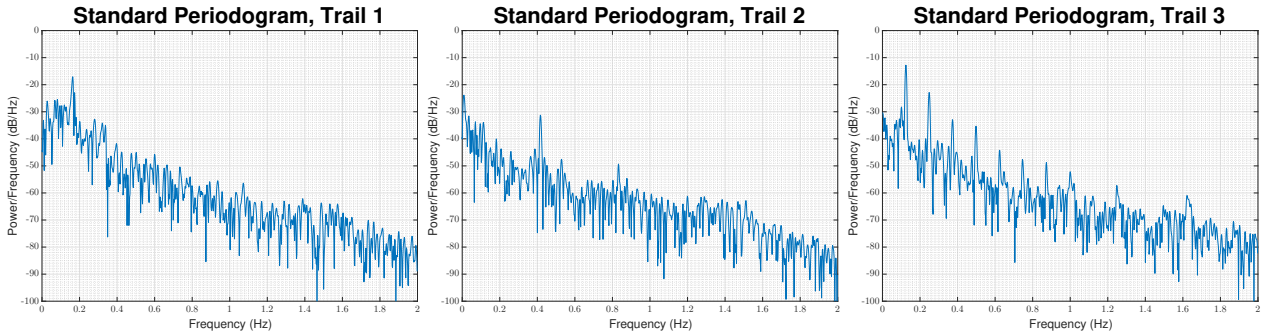


Figure 29: Standard Periodograms of RRI Data obtained from Three Separate Trials

The averaged periodogram with different window lengths (50 samples, 150 samples, 200 samples) for each unique trial are shown below. Note that the segments used to average the periodogram did not have any overlaps. A Hamming window was applied to each segment. The Hamming window was developed to minimize the height of the maximum (nearest) side lobe. This minimizes the amount of spectral leakage and produces a spectrum that allows for harmonics of the RSA to be identified in the presence of noise.

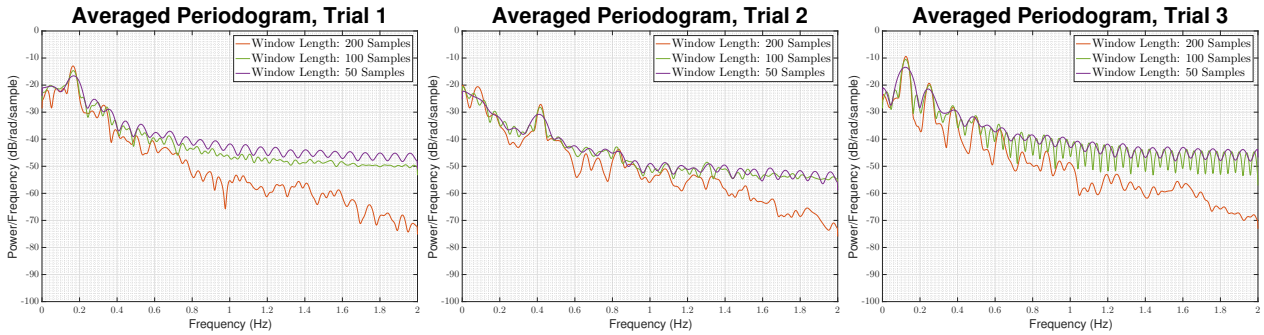


Figure 30: Averaged Periodograms, with Different Window Lengths, of RRI Data

b. Note that the three trials were conducted such that:

1. Trial 1: Unconstrained Breathing, Typical Ventilation Rate 20-45 Breaths Per Minute (BPM)
2. Trial 2: Fast Breathing, Ventilation Rate 50 BPM
3. Trial 3: Slow Breathing, Ventilation Rate 15 BPM

With this in mind, we study the standard periodograms and note that the first harmonics of the RSA for each trial are:

1. Trial 1: Peak at 0.1641 Hz, which implies a breathing rate of $2 \times 60 \times 0.1641 = 19.69 \approx 20$ BPM
2. Trial 2: Peak at 0.416 Hz, which implies a breathing rate of $2 \times 60 \times 0.416 = 49.92 \approx 50$ BPM
3. Trial 3: Peak at 0.125 Hz, which implies a breathing rate of $2 \times 60 \times 0.125 = 15$ BPM

We study the averaged periodograms and find that they produce smoother estimates because increasing the number of samples for which we average the periodogram means that the variance of the estimate decreases. Averaging over 200 samples, the 2nd harmonic in Trial 1 at $f = 0.332$ Hz is clear. This harmonic cannot be distinguished in the standard periodogram. In fact, when averaged over 50 and 100 samples, the peak of the 2nd harmonic does not occur at $f = 0.332$ Hz; the peaks actually occur at $f = 0.2773$ Hz and $f = 0.2949$ Hz for the 50 samples and 100 samples periodograms respectively. **This means that, in an attempt to remove noise, by averaging over smaller segments, we have lost the resolution and the ability to correctly identify the 2nd harmonic.** Similarly, for other trials, harmonics are much more evident in the averaged

periodogram with a small shift introduced when averaged over 50 and 100 samples.

c. The figure below shows the results obtained when the RRI data is modeled as an AR process. For Trial 1, it is impossible to fit an AR model that models any harmonic beyond the fundamental. This is because of the fact that in the original signal, the only significant periodic component occurs at $f = 0.1641$ Hz. As such, increasing the model order amplifies the peak at $f = 0.1641$ Hz instead of introducing more peaks. In contrast, the averaged periodogram, when averaged over 200 sample segments, is able to model the harmonics well. In addition, notice that beyond order 10, an artifact starts to occur at 1.5 Hz. This is not present in the averaged periodogram.

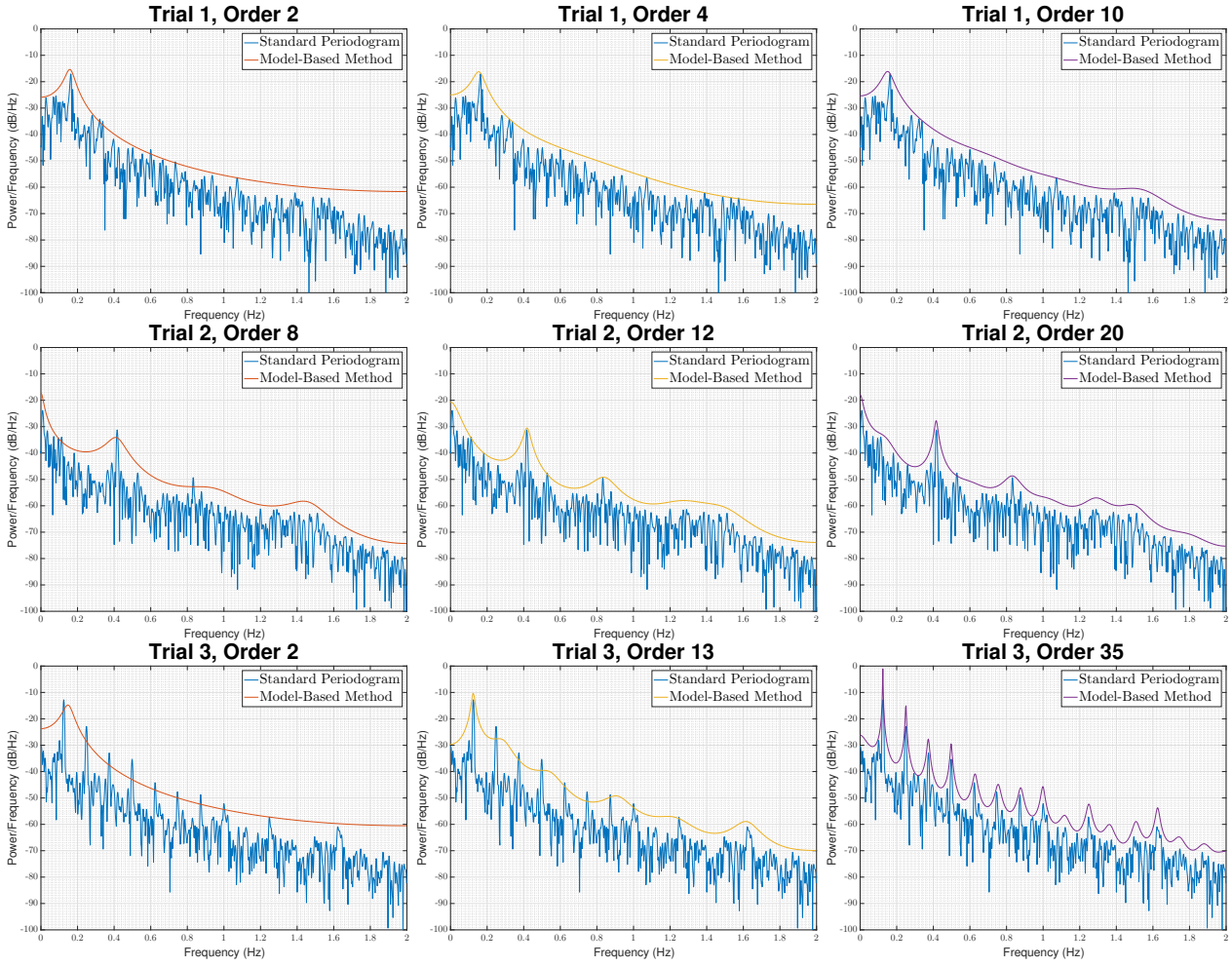


Figure 31: Autoregressive Spectrum Estimate for Trial 1,2,3 for Different Model Orders

For Trial 2, it is clear that increasing the model order provides greater degrees of freedom that are able to describe the 2nd and possibly 3rd harmonic well. For Trial 3, a similar trend is observed. Using an order 35 model, all the peaks can be accurately modeled. **For Trial 3, although it is possible to replicate the peaks observed in the standard periodogram, a very high model order is required and this is usually not a good idea as real-world process usually do not have that much memory.**

Note that, although the AR model based spectrum has its short-comings described above, in general the estimates are very smooth and only tiny offsets/artifacts are observed.

3 Adaptive Signal Processing

3.1 The Least Mean Square (LMS) Algorithm

a. The AR(2) process defined in the coursework has the following general form:

$$x(n) = a_1 x(n-1) + a_2 x(n-2) + \eta(n) \tag{11}$$

where $\eta(n) \sim \mathcal{N}(0, \sigma_\eta^2)$. Since the correlation matrix $\mathbf{R}_x = \mathbb{E}\{\mathbf{x}(n)\mathbf{x}(n)^T\}$, where $\mathbf{x}(n) = [x(n-1), x(n-2)]^T$, the entries of the matrix are:

$$\mathbf{R}_x = \begin{bmatrix} \mathbb{E}\{x(n-1)x(n-1)\} & \mathbb{E}\{x(n-1)x(n-2)\} \\ \mathbb{E}\{x(n-2)x(n-1)\} & \mathbb{E}\{x(n-2)x(n-2)\} \end{bmatrix}$$

In order to determine the entries, we multiply (11) with $x(n-k)$ and obtain:

$$x(n)x(n-l) = a_1 x(n-1)x(n-k) + a_2 x(n-2)x(n-l) + \eta(n)x(n-k)$$

By taking expectations, we get:

$$\gamma(k) = \mathbb{E}\{x(n)x(n-k)\} = a_1 \mathbb{E}\{x(n-1)x(n-k)\} + a_2 \mathbb{E}\{x(n-2)x(n-k)\} + \mathbb{E}\{\eta(n)x(n-k)\}$$

The above equation can be simplified and 3 simultaneous equations can be generated to obtain the entries of the correlation matrix:

$$\begin{aligned} \gamma(0) &= a_1\gamma(1) + a_2\gamma(2) + \sigma_\eta^2 \\ \gamma(1) &= a_1\gamma(0) + a_2\gamma(1) \\ \gamma(2) &= a_1\gamma(1) + a_2\gamma(0) \end{aligned}$$

Solving the simultaneous equations with $a_1 = 0.1$, $a_2 = 0.8$ and $\sigma_\eta^2 = 0.25$, we obtain the ACF matrix:

$$\mathbf{R}_x = \begin{bmatrix} \mathbb{E}\{x(n-1)x(n-1)\} & \mathbb{E}\{x(n-1)x(n-2)\} \\ \mathbb{E}\{x(n-2)x(n-1)\} & \mathbb{E}\{x(n-2)x(n-2)\} \end{bmatrix} = \begin{bmatrix} \gamma(0) & \gamma(1) \\ \gamma(1) & \gamma(0) \end{bmatrix} = \begin{bmatrix} \frac{25}{27} & \frac{25}{54} \\ \frac{25}{54} & \frac{25}{27} \end{bmatrix}$$

To find the range of values for μ that will allow the LMS algorithm to converge in mean, the eigenvalues of $\mathbf{R}_x$ have to be found and the inequality in (12) has to be satisfied.

$$0 \; < \; \mu \; < \; \frac{2}{\lambda_{max}} \tag{12}$$

If the inequality is satisfied, the LMS algorithm will converge to the Winer-Hopf solution. The maximum eigenvalue of $\mathbf{R}_x$ is 1.3889 and thus as long as $0 \; < \; \mu \; < \; 1.44$, the LMS algorithm will converge.

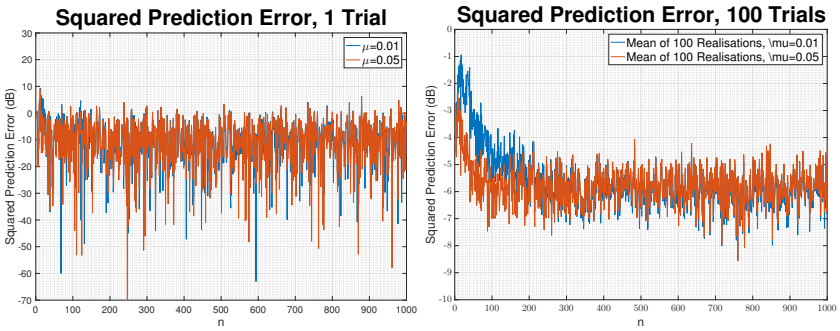


Figure 32: Analysis of Effects of Different Step Size on Convergence Speed of LMS Alogrithm

b. The LMS adaptive predictor using $N = 1000$ samples of $x(n)$ was implemented. Results of 1 realization and the average of 100 realizations are shown in the figure above for $\mu = 0.01$ and $\mu = 0.05$. Analyzing the squared

prediction error curves, it is clear that on average we obtain faster convergence when $\mu = 0.05$. This is exactly as expected because larger values of μ enable a steeper and thus quicker descent of the error surface.

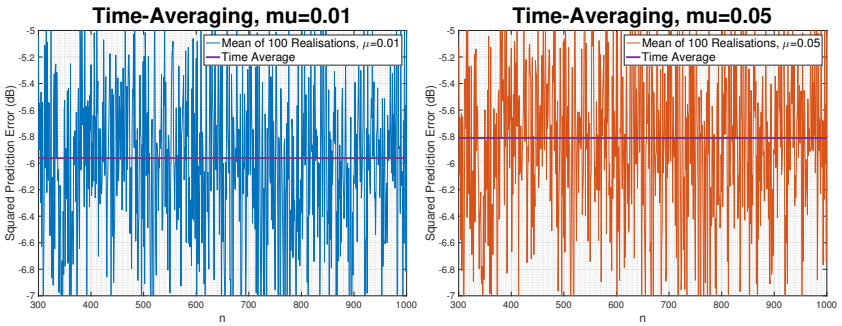


Figure 33: Time-Average of Steady-State Error for $\mu = 0.01$ and $\mu = 0.05$

c. The figure above shows the steady-state of the ensemble average learning curves using 100 independent trials of the experiment. Based on the 100 independent realizations, the excess mean square error (EMSE), misadjustment and theoretical misadjustment are calculated and the results are shown in the table below. The empirical misadjustments are slightly higher than the theoretical values. Regardless, it is abundantly clear that having a smaller step size introduces a much smaller EMSE and results in a smaller misadjustment value. However, as observed above, having a smaller step size causes the rate of convergence to be slow.

μ	EMSE	Empirical Misadjustment	Theoretical Misadjustment
0.01	0.0032	0.0128	0.0093
0.05	0.124	0.0496	0.0463

d. The graphs below show how the coefficients evolve over time. **For the rest of the section, most graphs will show the evolution of coefficients, averaged over 100 realizations, and thus it is good to get a feel of what the graphs look like for just 1 realization. Real world adaptive filters will often work with only 1 realization of a random signal.** Looking more specifically at the evolution of coefficients for $\mu = 0.05$ for 1 realization, it is clear that speed has been traded off for steady-state error. The amount of variation around the ideal values is significant.

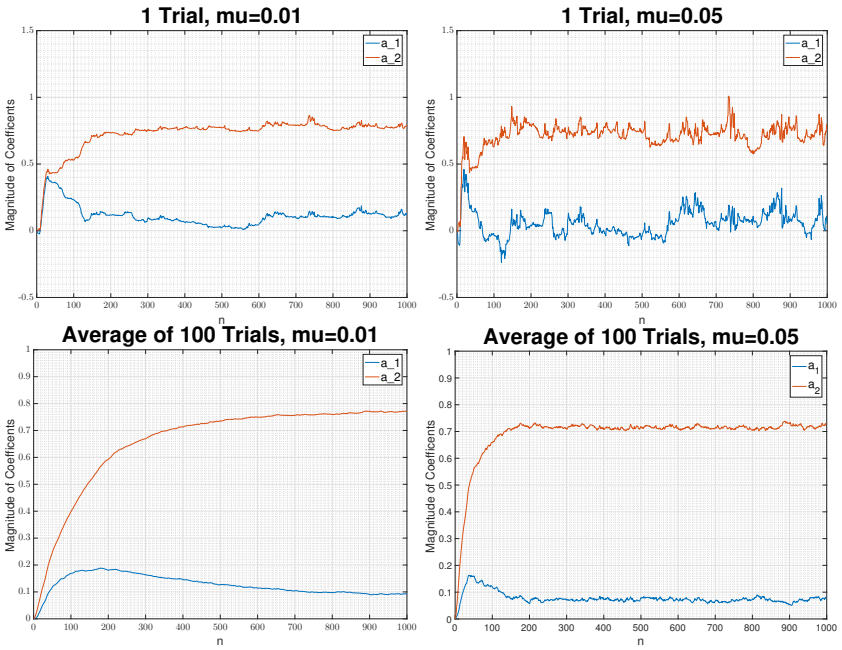


Figure 34: Evolution of Coefficients for 1 Trial and 100 Trial, for $\mu = 0.01$ and $\mu = 0.05$

Next, to answer the question directly, we study Figure 35. It is clear that for both coefficients, having a small value of μ results in the steady-state value that is much closer to the ideal values of $a_1 = 0.1$ and $a_2 = 0.8$. In addition, having a small value of μ results in a much smaller steady-state variance. However, looking at

Figure 34, we find that setting $\mu = 0.01$, we do not approach the steady-state value until the 900th iteration. In contrast, using $\mu = 0.05$, the algorithm converges in about 200 iterations.

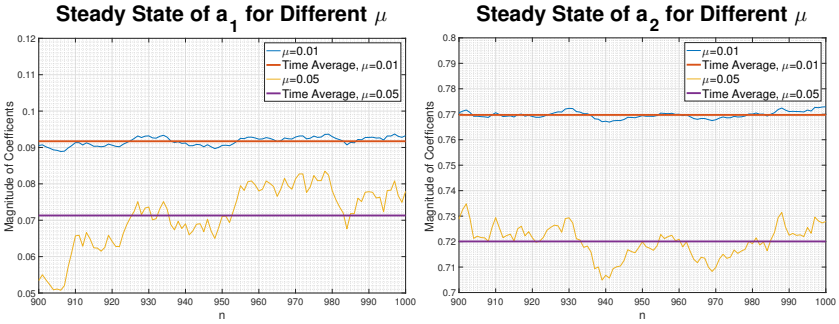


Figure 35: Steady-State Values of Coefficients a_1 and a_2, for $\mu = 0.01$ and $\mu = 0.05$

e. In order to minimize the cost function with respect to $\mathbf{w}$, we differentiate it with respect to $\mathbf{w}$ and set the derivative to 0. Mathematically, we obtain:

$$J(n) = \frac{1}{2}\left(e^2(n) + \gamma||\mathbf{w}(n)||_2^2\right) \tag{13}$$
$$\frac{\partial J}{\partial \mathbf{w}} = e(n)\frac{\partial e}{\partial \mathbf{w}} + \gamma \mathbf{w}(n) = 0$$

We can express the error as $e(n) = \mathbf{x}(n) - \mathbf{w}^T(n)\mathbf{x}(n)$. If we make the substitution and differentiate we get,

$$\frac{\partial e}{\partial \mathbf{w}} = \frac{\partial}{\partial \mathbf{w}}\left(\mathbf{x}(n) - \mathbf{w}^T(n)\mathbf{x}(n)\right) = -\mathbf{x}(n)$$

Then,

$$\frac{\partial J}{\partial \mathbf{w}} = -e(n)\mathbf{x}(n) + \gamma \mathbf{w}(n)$$

Now, using the gradient decent method to reach the optimal value of w, we have to implement the update equation presented in (14).

$$\mathbf{w}(n) = \mathbf{w}(n) - \mu \nabla_w J(n) \tag{14}$$

Substituting the derivative of the cost function into (14), we obtain:

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\left(-e(n)\mathbf{x}(n) + \gamma \mathbf{w}(n)\right)$$
$$\mathbf{w}(n+1) = (1 - \mu\gamma)\mathbf{w}(n) + \mu e(n)\mathbf{x}(n) \tag{15}$$

Therefore, we have shown that the leaky LMS equation quoted in the coursework and derived in (15) is equivalent to minimizing the cost function shown in (13).

f. The graphs shown in Figure 36 show evolution of coefficients for different values of μ and γ using the leaky LMS algorithm. The steady-state values do not converge to the true values of a_1 and a_2. In fact, the larger the value of γ, the greater the steady-state bias. The Weiner-Hopf solution was briefly mentioned above. It gives the optimal weights calculated using the autocorrelation matrix, $\mathbf{R}$, and the cross correlation vector, $\mathbf{p}$:

$$\mathbf{w}_{opt} = \mathbf{R}^{-1}\mathbf{p}$$

The LMS algorithm, under certain bounds on the value of μ, iteratively arrives at the Weiner-Hopf solution. The motivation of the algorithm is to prevent inverting the autocorrelation matrix. The leaky

LMS however does not converge to the Wiener-Hopf solution. The leaky LMS minimizes a cost function that is slightly different to the cost function that the LMS algorithm minimizes and thus it converges to:

$$\lim_{k\to\infty} \mathbb{E}[\mathbf{w}_k] = (\mathbf{R} + \gamma\mathbf{I})^{-1}\mathbf{p}$$

The additional $\gamma\mathbf{I}$ term causes a bias that means that the leaky LMS does not actually converge to the Wiener-Hopf solution. In fact, using the autocorrelation matrix $\mathbf{R}_x$ derived above and $\mathbf{p} = [\gamma(1), \gamma(2)]^T$, we can mathematically determine the values to which the Leaky-LMS algorithm converges:

γ	a_1	a_2
0.1	0.1542	0.3919
0.5	0.1626	0.4992
0.9	0.1319	0.7076

Notice that the graphs confirm the theoretical results. Different values of γ lead to the Leaky-LMS algorithm settling to different values. The leaky LMS displays the same properties, discussed above, with respect to μ.

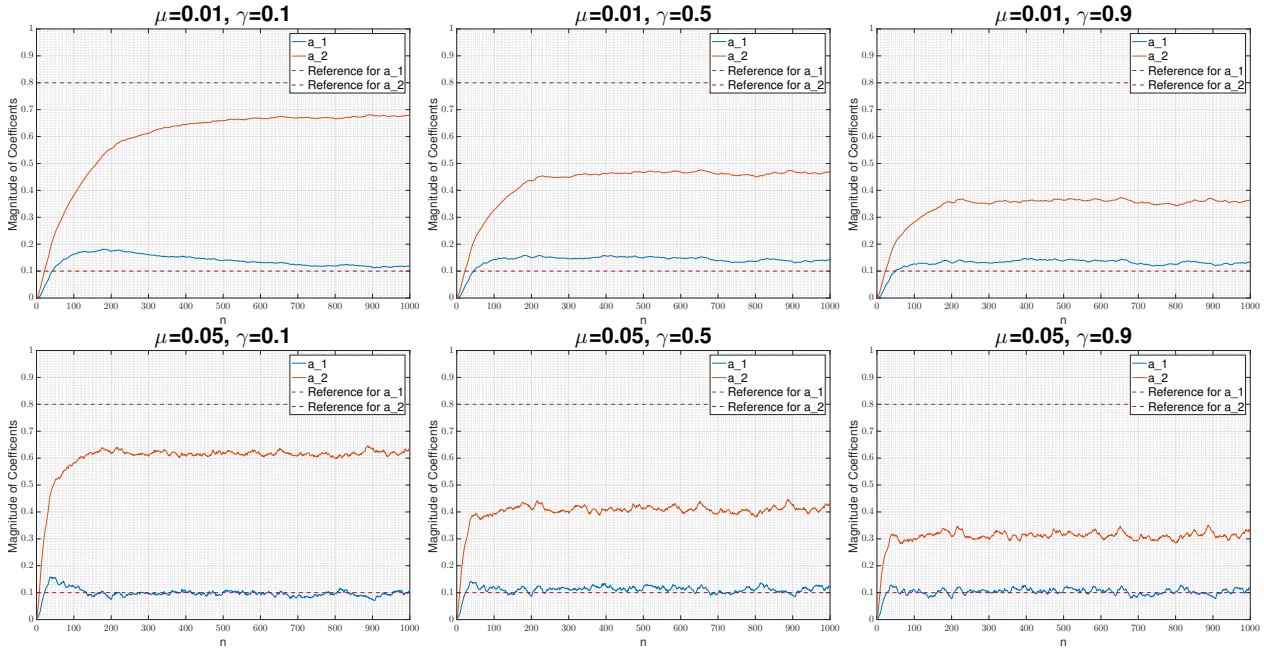


Figure 36: Effect of increasing γ on the Steady-State Values of Coefficients a_1 and a_2, for $\mu = 0.01$ and $\mu = 0.05$

3.2 Adaptive Step Sizes

a. Using the LMS algorithm, we have shown the effect that μ has on the convergence time as well as the misadjustment. Depending on the application, we can select a μ to satisfy constraints on convergence time or overshoot or steady-state error. Also, the non-varying nature of μ makes the LMS algorithm computationally cheap. However, keeping a constant μ has certain shortcomings. A major shortcoming is that certain choices μ can cause the algorithm to diverge instead of converging to a solution.

The Gradient Adaptive Step Size (GASS) algorithms allow μ to be time-varying. The algorithms are generally implemented such that the steady-state learning rates approach 0, thereby ensuring stability and convergence. The adaptive nature of the step sizes gives us the flexibilty to vary μ according to the error that is incurred. The three GASS algorithms specified in the coursework all modify the step size according to the following equation:

$$\mu(n+1) = \mu(n) + \rho e(n)\mathbf{x}^T(n)\boldsymbol{\psi}(n)$$

In all algorithms the value of ρ is a constant while $\boldsymbol{\psi}$ is time-varying. The value of $\boldsymbol{\psi}$ is altered using the following equations:

$$\begin{aligned}
\text{Benveniste} &: \boldsymbol{\psi}(n) = [\mathbf{I} - \mu(n-1)\mathbf{x}(n-1)\mathbf{x}^T(n-1)]\boldsymbol{\psi}(n-1) + e(n-1)\boldsymbol{x}(n-1) \\
\text{Ang \& Farhang} &: \boldsymbol{\psi}(n) = \alpha\boldsymbol{\psi}(n-1) + e(n-1)\boldsymbol{x}(n-1), \qquad 0 < \alpha < 1 \\
\text{Matthew \& Xie} &: \boldsymbol{\psi}(n) = e(n-1)\boldsymbol{x}(n-1)
\end{aligned}$$

Notice that in the Benveniste algorithm, the update equation for $\boldsymbol{\psi}$ is an adaptive low-pass filter with the instantaneous gradient $e(n-1)x(n-1)$ as the input. This algorithm has the ability to deal with noise in the gradient estimate. The Ang & Farhang algorithm modifies the Benveniste algorithm by turning the update equation for $\boldsymbol{\psi}$ from an adaptive to a non-adaptive low-pass filter. The algorithm trades computational complexity for slightly worse convergence properties. The Matthew & Xie algorithm goes a step further and removes the low-pass filtering that the other two algorithms perform on the instantaneous gradient. This makes the algorithm even faster, however once again convergence properties are sacrificed for speed.

The graphs below show the weight error curves for just 1 trial of the algorithms. It is clear that all of the GASS algorithms perform significantly better than the LMS algorithm which does not vary the value of μ.

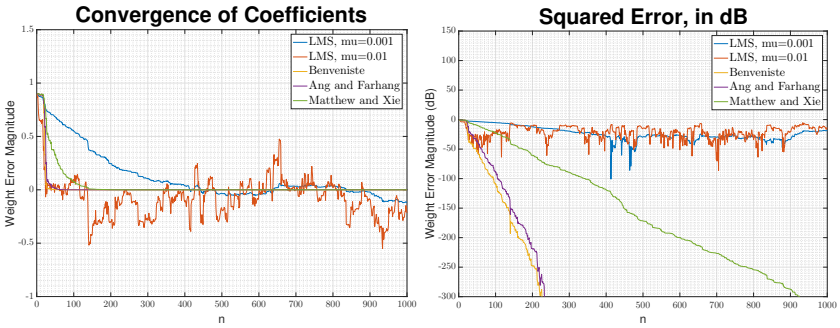


Figure 37: Comparison of Convergence Time using Adaptive Step Sizes and Standard LMS Algorithms

The averaged results of 100 trials are graphed below. The first thing to notice is that the LMS alogrithms with constant μ do not converge to the correct weights. The results obtained for the GASS algorithms are exactly as expected. The Benveniste algorithm, which is the most computationally expensive algorithm, is able to converge the most quickly. The next to follow is the Ang & Farhang algorithm. Notice that although the Matthew & Xie algorithm performs the worst amongst the GASS algorithms, it is still significantly better than the LMS algorithms without adaptive step-sizes. Notice also that all GASS algorithms do not have offsets and converge to the true weights.

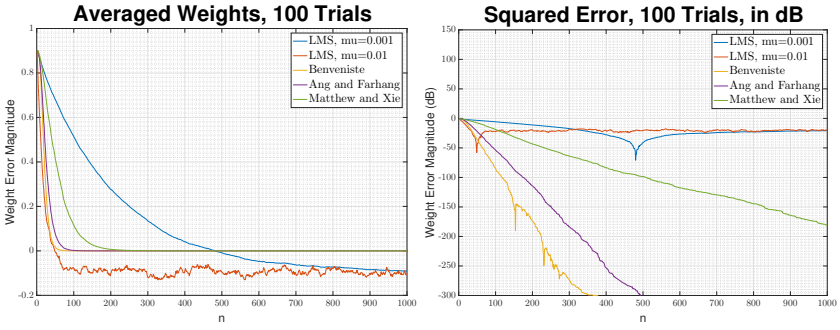


Figure 38: Studying Convergence Time by Averaging Weights over 100 Random Realizations

b. The normalised LMS (NLMS) algorithm has the following update equation:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \frac{\beta}{\epsilon + ||\mathbf{x}||^2} e(n)\mathbf{x}(n) \tag{16}$$

To show that the update equation in (17), based upon the *a posteriori* error $e_p(n) = d(n) - \mathbf{x}^T(n)\mathbf{w}(n+1)$, is equivalent to the NLMS algorithm, we first multiply both sides by $\mathbf{x}^T(n)$.

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu e_p(n)\mathbf{x}(n) \tag{17}$$
$$\mathbf{x}^T(n)\mathbf{w}(n+1) = \mathbf{x}^T(n)\mathbf{w}(n) + \mathbf{x}^T(n)\mu e_p(n)\mathbf{x}(n)$$

Next, we add $\mathbf{d}(n)$ to both sides and rearrange:

$$\mathbf{d}(n) + \mathbf{x}^T(n)\mathbf{w}(n+1) = \mathbf{d}(n) + \mathbf{x}^T(n)\mathbf{w}(n) + \mathbf{x}^T(n)\mu e_p(n)\mathbf{x}(n)$$
$$\mathbf{d}(n) - \mathbf{x}^T(n)\mathbf{w}(n) = \mathbf{d}(n) - \mathbf{x}^T(n)\mathbf{w}(n+1) + \mathbf{x}^T(n)\mu e_p(n)\mathbf{x}(n) \tag{18}$$

Next we simplify using the *a posteriori* error, $e_p(n)$, and the error, $e(n)$:

$$e(n) = e_p(n) + \mathbf{x}^T(n)\mu e_p(n)x(n)$$
$$e(n) = e_p(n)\Big(1 + \mu||\mathbf{x}||^2\Big)$$
$$e_p(n) = \frac{e(n)}{1 + \mu||\mathbf{x}||^2} \tag{19}$$

We can now substitute (19) into the original update equation (17) and obtain the NLMS algorithm defined in (16):

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu\frac{e(n)}{1 + \mu||\mathbf{x}||^2}\mathbf{x}(n)$$
$$= \mathbf{w}(n) + \frac{1}{\frac{1}{\mu} + ||\mathbf{x}||^2}e(n)\mathbf{x}(n) \tag{20}$$

Thereby proving that the update equation in (17) based on the *a posteriori* error with $\epsilon = \frac{1}{\mu}$ and $\beta = 1$ is equivalent to the NLMS algorithm defined in (16). Since ϵ is inversely proportional to the step size μ, the algorithm does not become unstable even if $||\mathbf{x}||^2$ is extremely small.

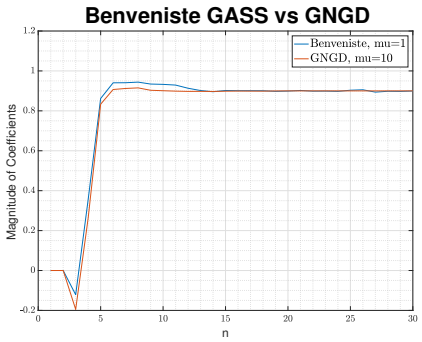


Figure 39: Comparing Convergence Speed of GNGD and Benveniste's GASS Algorithms

c. From the figure above, it is clear that the Generalized Normalized Gradient Descent (GNGD) algorithm converges slightly faster than the Benveniste algorithm. Note that prior testing was conducted to select an optimal value of μ for the Benveniste algorithm; the figure above is obtained when $\mu_{\text{Benveniste}} = 1$. While the Benveniste algorithm's performance changes significantly based on the initial value of μ, this is not the case for the GNGD algorithm. The GNGD algorithm performs well for a large range of μ values; the figure above is obtained when $\mu_{\text{GNGD}} = 10$.

In fact, both algorithms are similar because they modify the learning rate in an adaptive manner. The difference lies in the fact that the GNGD algorithm modifies the learning rate in a non-linear manner. The non-linear updates provide compensation for the assumptions made in the derivation of the NLMS. The non-linear updates make the algorithm more robust and the increased stability make it effective at processing non-linear and non-stationary signals [2].

In terms of the computational complexity of both algorithms, the GNGD algorithm also beats the Benveniste algorithm. The number of additions and multiplications for the Benveniste algorithm are presented in the Table

2. The variable M represents the model order. It is clear that the Benveniste algorithm scales quadratically with the model order. The computational complexity of the algorithm can be attributed to the outer product $\mathbf{x}(n-1)\mathbf{x}^T(n-1)$ that has to performed to update the value of $\boldsymbol{\psi}(n)$.

Step	Sub-Step	Multiplications	Additions
$\boldsymbol{\psi}(n)$	$\mu(n-1)\mathbf{x}(n-1)$	M	0
	$\mu(n-1)\mathbf{x}(n-1)\mathbf{x}^T(n-1)$	M^2	0
	$I-\mu(n-1)\mathbf{x}(n-1)\mathbf{x}^T(n-1)$	0	M
	$[I-\mu(n-1)\mathbf{x}(n-1)\mathbf{x}^T(n-1)]\boldsymbol{\psi}(n-1)$	M^2	M^2-M
	$e(n-1)\mathbf{x}(n-1)$	M	0
	$[I-\mu(n-1)\mathbf{x}(n-1)\mathbf{x}^T(n-1)]\boldsymbol{\psi}(n-1)+e(n-1)\mathbf{x}(n-1)$	0	M
$\mu(n+1)$	$\mathbf{x}^T(n)\boldsymbol{\psi}(n)$	M	$M-1$
	$\rho e(n)$	1	0
	$\rho e(n)\mathbf{x}^T(n)\boldsymbol{\psi}(n)$	1	0
	$\mu(n)+\rho e(n)\mathbf{x}^T(n)\boldsymbol{\psi}(n)$	0	1
$\mathbf{w}(n+1)$	$\mu(n)e(n)$	1	0
	$\mu e(n)\mathbf{x}(n)$	M	0
	$\mathbf{w}(n)+\mu e(n)\mathbf{x}(n)$	0	M
Total		$\mathcal{O}(M^2)$	$\mathcal{O}(M^2)$

Table 2: Computational Complexity of the Benveniste Algorithm

Table 3 shows the computational complexity of the GNGD algorithm. It is clear that the GNGD algorithm only scales linearly with the model order.

Step	Sub-Step	Multiplications	Additions
$\epsilon(n+1)$	$\mathbf{x}^T(n)\mathbf{x}(n-1)$	M	$M-1$
	$e(n)e(n-1)$	1	0
	$e(n)e(n-1)\mathbf{x}^T(n)\mathbf{x}(n-1)$	1	0
	$\|\|\mathbf{x}(n-1)\|\|^2$	M	$M-1$
	$(\epsilon(n-1)+\|\|\mathbf{x}(n-1)\|\|^2)^2$	1	1
	$\rho\mu\frac{e(n)e(n-1)\mathbf{x}^T(n)\mathbf{x}(n-1)}{(\epsilon(n-1)+\|\|\mathbf{x}(n-1)\|\|^2)^2}$	3	0
	$\epsilon(n)+\rho\mu\frac{e(n)e(n-1)\mathbf{x}^T(n)\mathbf{x}(n-1)}{(\epsilon(n-1)+\|\|\mathbf{x}(n-1)\|\|^2)^2}$	0	1
$\mathbf{w}(n+1)$	$\|\|\mathbf{x}(n)\|\|^2$	M	$M-1$
	$\frac{\beta}{\epsilon(n)+\|\|\mathbf{x}(n)\|\|^2}$	1	1
	$\frac{\beta}{\epsilon(n)+\|\|\mathbf{x}(n)\|\|^2}e(n)$	1	0
	$\frac{\beta}{\epsilon(n)+\|\|\mathbf{x}(n)\|\|^2}e(n)\mathbf{x}(n)$	M	$M-1$
	$\mathbf{w}(n)+\frac{\beta}{\epsilon(n)+\|\|\mathbf{x}(n)\|\|^2}e(n)\mathbf{x}(n)$	0	M
Total		$\mathcal{O}(M)$	$\mathcal{O}(M)$

Table 3: Computational Complexity of the GNGD Algorithm

3.3 Adaptive Noise Cancellation

a. By choosing an appropriate value of Δ, the Adaptive Line Enhancement (ALE) configuration can be set up such that the noise, $\eta(n)$, in the signal, $s(n)$, and the predicted input, $\mathbf{u}(n)$ are uncorrelated. If the noise $\eta(n)$ and the predicted input $\mathbf{u}(n)$ are uncorrelated, the noise can be suppressed and a clean signal can be obtained. To find an optimal value of Δ, we start with the mean-squared error:

$$\mathbb{E}\Big\{\Big(s(n)-\hat{x}(n)\Big)^2\Big\} = \mathbb{E}\Big\{\Big(x(n)+\eta(n)-\hat{x}(n)\Big)^2\Big\}$$
$$= \mathbb{E}\{\eta^2(n)\} + \mathbb{E}\Big\{\Big(x(n)-\hat{x}(n)\Big)^2\Big\} + 2\mathbb{E}\Big\{\eta(n)\Big(x(n)-\hat{x}(n)\Big)\Big\}$$

Expanding the mean-sqaured error shows that there are three terms that contribute to the error. Ideally, we would like the mean-squared error to be exactly equal to the noise-power. However, by changing the value of Δ, it is only possible to manipulate $2\mathbb{E}\Big\{\eta(n)\Big(x(n)-\hat{x}(n)\Big)\Big\}$. As such, Δ should be chosen so as to minimize $2\mathbb{E}\Big\{\eta(n)\Big(x(n)-\hat{x}(n)\Big)\Big\}$. It is important to note that since $x(n)$ and $\eta(n)$ are uncorrelated, the minimization problem reduces to minimizing $\mathbb{E}\{\eta(n)\hat{x}(n)\}$. Since $\eta(n) = v(n) + 0.5v(n-2)$ we get:

$$\mathbb{E}\{\eta(n)\hat{x}(n)\} = \mathbb{E}\Big\{\Big(v(n)+0.5v(n-2)\Big)\mathbf{w}^T\mathbf{u}(n)\Big\}$$
$$= \mathbb{E}\Big\{\Big(v(n)+0.5v(n-2)\Big)\Big(\sum_{i=0}^{M} w_i s(n-\Delta-i)\Big)\Big\}$$
$$= \mathbb{E}\Big\{\Big(v(n)+0.5v(n-2)\Big)\Big(\sum_{i=0}^{M} w_i\big(x(n-\Delta-i)+\eta(n-\Delta-i)\big)\Big)\Big\}$$

Again using the fact that $x(n)$ and $v(n)$ are uncorrelated:

$$\mathbb{E}\{\eta(n)\hat{x}(n)\} = \mathbb{E}\Big\{\Big(v(n)+0.5v(n-2)\Big)\Big(\sum_{i=0}^{M} w_i\eta(n-\Delta-i)\Big)\Big\}$$
$$= \mathbb{E}\Big\{\Big(v(n)+0.5v(n-2)\Big)\Big(\sum_{i=0}^{M} w_i\big(v(n-\Delta-i)+0.5v(n-\Delta-i-2)\big)\Big)\Big\}$$

Thus, it is clear that for a value of Δ greater than 2, the expectation reduces to adding the expectations of white noise samples at different time indexes, which are uncorrelated. The results presented in Figure 40 corroborate these finding. Note that the top row of the figure shows the averaged results from 100 realizations, whereas the bottom row shows the overlay of 100 realizations. The ideal sinewave has been superimposed for reference. All the figures conclusively prove that for $\Delta > 2$ the results obtained are significantly better. This is also corroborated by the large decrease in the mean-squared error from when Δ is increased from 2 to 3.

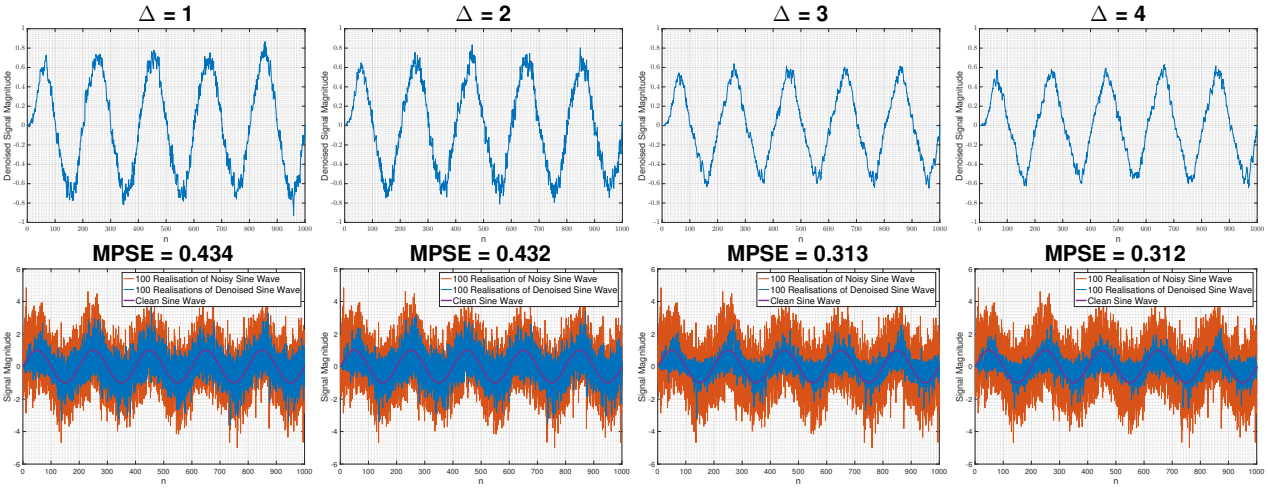


Figure 40: Determining Ideal Value for Δ to be used in the Adaptive Line Enhancement Algorithm

b. The results in Figure 41 further verify the results above. Choosing a value of Δ greater than 2 causes the mean-squared error to decrease for all model-orders. Note that arbitrarily increasing the value of Δ is not wise.

One of the terms which contribute to the mean-squared error, $\mathbb{E}\left\{\left(s(n)-\hat{x}(n)\right)^2\right\}$, is $\mathbb{E}\left\{\left(x(n)-\hat{x}(n)\right)^2\right\}$. Increasing the value of Δ will increase the delay between $x(n)$ and $\hat{x}(n)$ and subsequently increase the mean-squared error. This effect is visualized in Figure 41. Notice that the output sine-wave is shifted relatively to the ideal input sinewave when $\Delta = 25$. This shift causes $\mathbb{E}\left\{\left(x(n)-\hat{x}(n)\right)^2\right\}$ to be large, thereby increasing the mean-squared error.

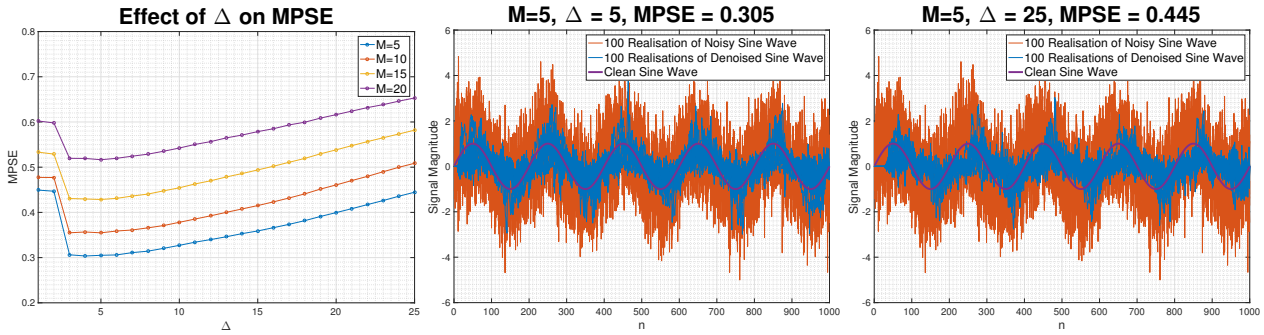


Figure 41: Studying the Effects of Increasing Delay on the MPSE

Theoretically, a higher model order should be able to describe the highly periodic process with greater accuracy. However, since we are deriving the solution numerically, factors such as convergence, stability and computational complexity should also be taken into account. Increasing the model order, while keeping μ constant may result in an increase in the mean-squared error. This effect is observed for values of $M > 6$. It is possible to keep μ small and increase the model order however this would cause the algorithm to converge slowly. Analyzing Figure 42, a pragmatic choice is to set $M = 6$ and set $\mu = 0.005$.

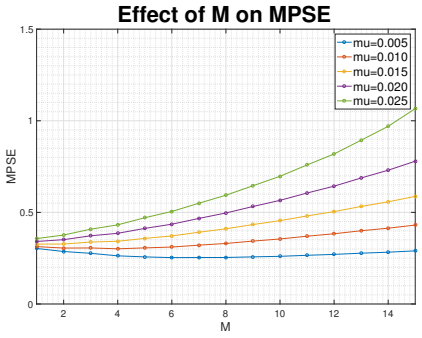


Figure 42: Studying the Effects of Increasing Model Order on the MPSE

c. The performance of the ALE configuration and the Adaptive Noise Cancellation (ANC) configuration are compared in the figure below. It is clear that the ANC configuration performs significantly better, once it has converged. However, notice that for small values of n, the ANC configuration incurs large errors.

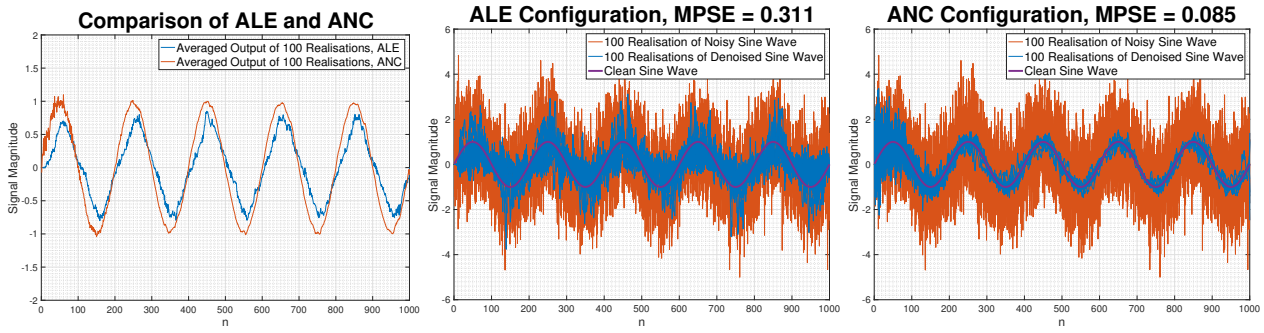


Figure 43: Comparison of ALE and ANC Configurations for Denoising Sinewave

d. The spectrogram in Figure 44 is graphed for reference. It shows the original EEG data with a very strong component at 50 Hz. The length of each segment is 4096, with an 80% overlap between segments. Also, each

segment is windowed with a Hamming window.

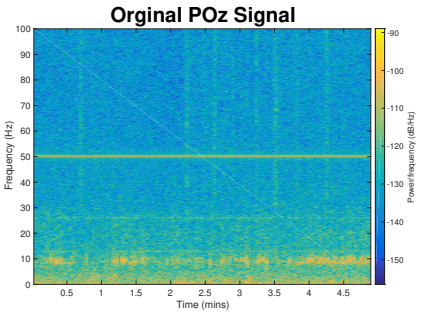


Figure 44: Original EEG Data collected from POz Location on the Scalp with Strong Component at 50 Hz

The first step is to determine the value of M that will remove the strong 50 Hz component. The figure below shows the effect of increasing the model order. Increasing the model order results in a filter that is more efficient at removing the noise. However, notice that certain artefacts have started to appear. With $M = 25$ and $\mu = 0.01$, the filter is not only removing the 50 Hz component but it is also suppressing components between 40 Hz and 60 Hz.

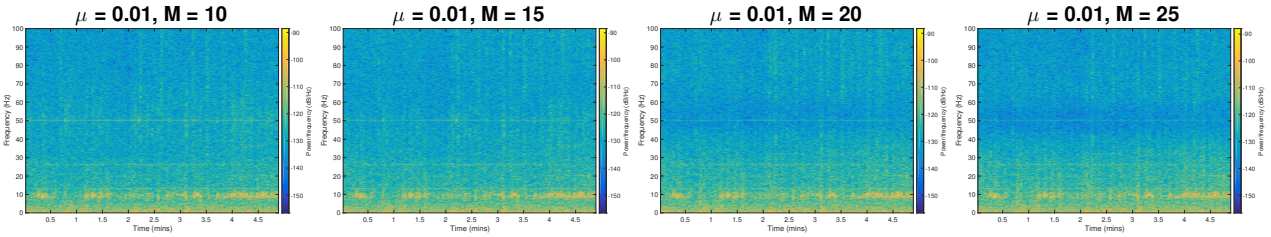


Figure 45: Effect of Increasing Model Order on the Spectrogram of EEG Data

To combat the large spread of frequencies over which the filter is acting, we vary μ. The figure below shows the effects of decreasing μ. Decreasing μ reduces the range of frequencies over which the ANC algorithm acts and leads to better performance. It is clear that the filter is now acting over a much smaller and targeted range of frequencies.

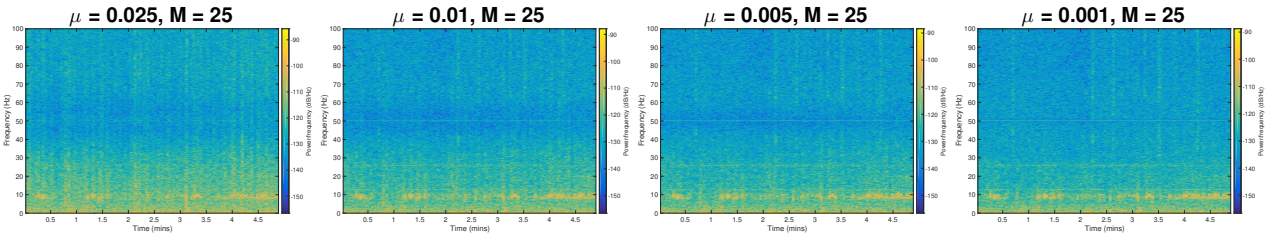


Figure 46: Effect of Varying μ on the artifacts observed in the Spectrogram of Denoised EEG Data

The periodogram and squared error plots corroborate the findings above. For large values of μ such as 0.025, large errors are incurred at low frequencies. This does not occur with $\mu = 0.001$.

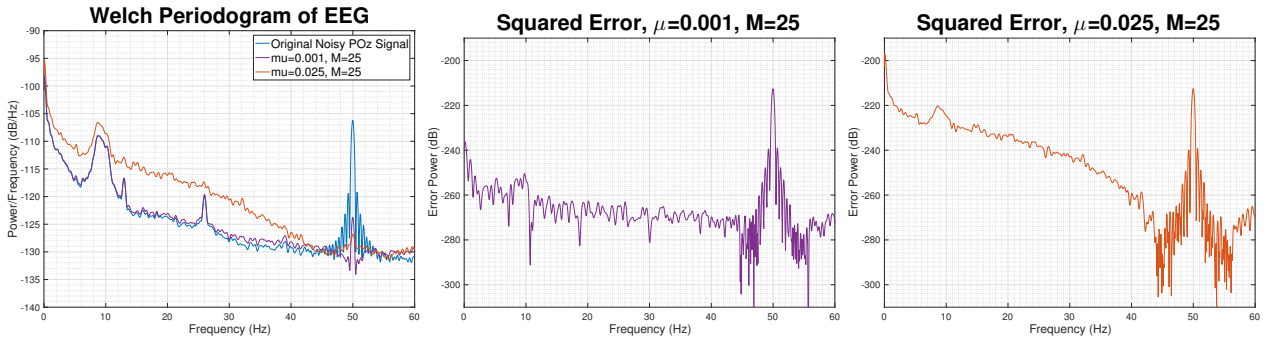


Figure 47: Welch Periodogram, averaged over 2 second intervals, and Squared Errors

4 Widely Linear Filtering and Adaptive Spectral Estimation

4.1 Complex LMS and Widely Linear Modelling

a. The Complex LMS (CLMS) and Augmented Complex LMS (ACLMS) algorithms are used to identify a first order Widely Linear Moving Average (WLMA(1)) process; it is clear that the WLMA(1) process is not circular. The learning curve graphed below shows that the CLMS algorithm is unable to capture all the degrees of freedom required to fully describe the non-circular process. In contrast, the additional weights $\mathbf{g}(n)$ afford the ACLMS algorithms additional degrees of freedom to fully describe the process. The ACLMS is also able to exploit the second order statistics of the data by utilizing both $x(n)$ and $x^*(n)$.

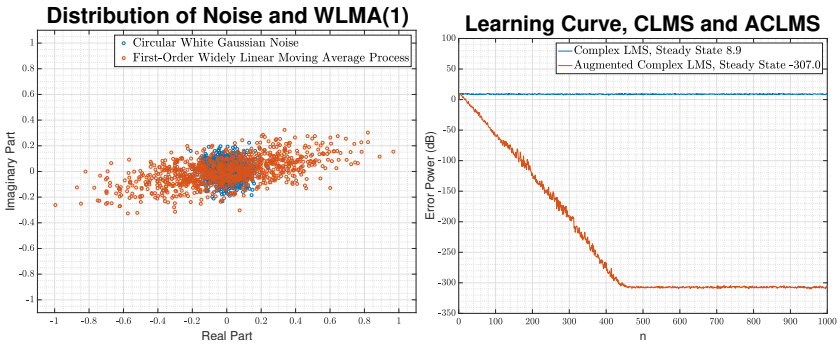


Figure 48: Comparison of Learning Curves using CLMS and ACLMS on Non-Circular WLMA(1) Process

b. Figure 49 shows the scatter plots for the three wind regimes. A complex-valued random variable is said to be circular if its probability distribution is not dependent on the angle, that is, the distribution is rotationally invariant. In other words the variable's probability distribution function should only depend on the Euclidean distance from the origin in the complex domain. The figure below also shows the circularity coefficient of each regime. **It is of great importance to note that the higher the value of the circularity quotient, the less circular the data is.**

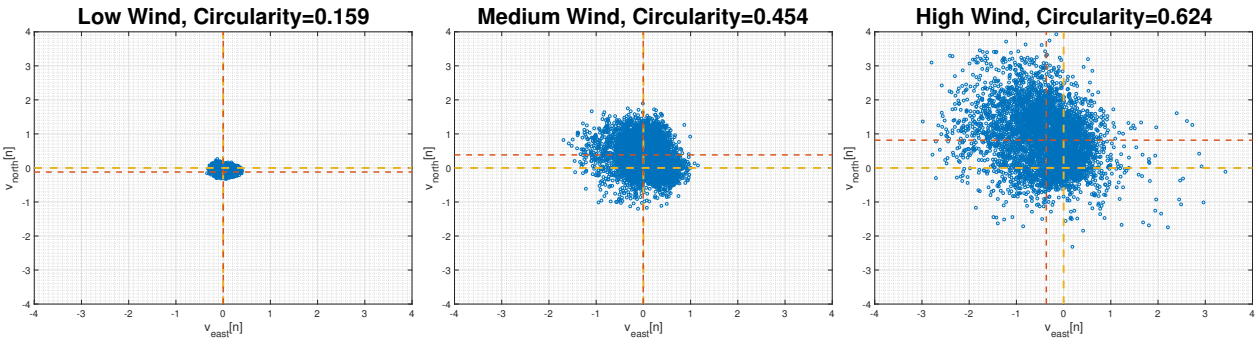


Figure 49: Circularity Plots for Different Wind Speeds

Looking at the graphs above without fully understanding the definition of circularity can lead to interpreting low wind as being the least circular. It may seem that the low wind regime is visually the least circular however upon closer inspection of the axis labels, it is clear that the high wind regime has a big offset in its mean. The red dotted lines show the average values along each axis while the yellow dotted lines show the reference to the origin. The scatter plot for the high wind regime is not centered around the origin and thus it is not rotationally invariant. In contrast, although the low wind regime does not appear circular and looks slightly elliptical, the fact that it is centered close to the origin means that it can be mathematically approximated to be circular. Each of the wind regimes was tested with the CLMS and the ACLMS algorithms and the results obtained are graphed in Figure 50.

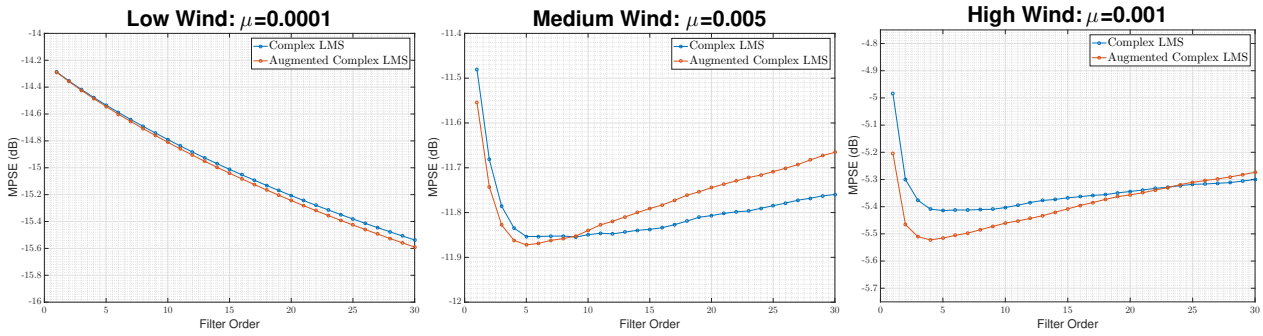


Figure 50: Scaling of MPSE with Increase in Filter Order

The results are in accordance with the circularity coefficients of each regime for small model orders. The high and medium wind regimes do not perform well when one-step prediction is done using CLMS. The algorithm is not able to capture the non-circularity in the data. The low wind regime performs equally well using both the CLMS and ACLMS algorithms.

For higher model orders, the performance of both algorithms starts to degrade. This is due to the fact that overfitting starts to take place. The ACLMS algorithm has a greater number of parameters to tune as compared to the CLMS algorithm. A larger number of parameters affords the ACLMS algorithm the ability to fit the training data extremely well. However, overfitting the training data has the disadvantage that the filter will no longer generalize well, or in this case, predict future values well. The CLMS algorithm also clearly overfits the data since the prediction error starts to rise as the model order increases but the rate of overfitting is slower; this is again due to the fact that the CLMS algorithm has fewer parameters to tune. **The above intuitive explanation of overfitting based on the number of free parameters or the degrees of freedom has been mathematically formalized. Vapnik-Chervonenkis Theory is the branch of mathematics that explains learning algorithms from a statistical point of view.**

c. The complex voltages in the three-phase power system can be expressed compactly using the Clarke Transform reproduced in (21).

$$v(n) = A(n)e^{j\left(2\pi\frac{f_0}{f_s}n+\phi\right)} + B(n)e^{-j\left(2\pi\frac{f_0}{f_s}n+\phi\right)} \tag{21}$$

where, $A(n)$ and $B(n)$ are as follows:

$$A(n) = \frac{\sqrt{6}}{6}\left(V_a(n) + V_b(n)e^{j\Delta_b} + V_c(n)e^{j\Delta_c}\right)$$
$$B(n) = \frac{\sqrt{6}}{6}\left(V_a(n) + V_b(n)e^{-j(\Delta_b+\frac{2\pi}{3})} + V_c(n)e^{-j(\Delta_c-\frac{2\pi}{3})}\right)$$

For the system to be balanced, $B(n)$ must be equal to 0. Firstly, all the voltages must have the same magnitudes, such that $V_a(n) = V_b(n) = V_c(n)$. Secondly, the relative phase difference between each of the three signals must not deviate from their prefixed values. This is achieved when $\Delta_b = \Delta_c = 0$. These two conditions thus give us 2 ways to unbalance the system. The figure below shows a balanced signal, variations of signals that have been unbalanced by changing the relative magnitudes of each component and also variations of signals that have been unbalanced by changing the phase difference between each component.

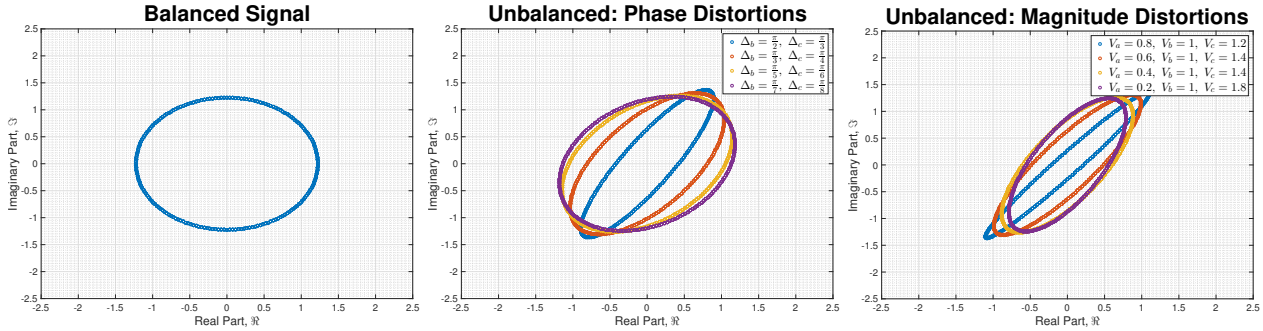


Figure 51: Balanced and Unbalanced Complex Voltages

It is clear that we can visually tell when a system is balanced and when it is unbalanced. A balanced system results in a circular plot, whereas an unbalanced system does not. In fact, patterns can also be drawn about the effects of changing the relative magnitudes or phase differences on the shape of the plot obtained.

d. A strictly linear first order autoregressive model has the form given in (22).

$$v(n+1) = h^*(n)v(n) \tag{22}$$

Next, the voltage of a balanced system can be expressed as follows:

$$v(n) = \sqrt{\frac{3}{2}} V e^{j\left(2\pi \frac{f_0}{f_s} n + \phi\right)}$$

We can rearrange (22) to give:

$$\frac{\sqrt{\frac{3}{2}} V e^{j\left(2\pi \frac{f_0}{f_s}(n+1) + \phi\right)}}{\sqrt{\frac{3}{2}} V e^{j\left(2\pi \frac{f_0}{f_s} n + \phi\right)}} = h^*(n)$$

Simplifying the left-hand side followed by replacing $h^*(n)$ with $h(n)$:

$$\begin{aligned} e^{j2\pi \frac{f_0}{f_s}} &= h^*(n) \\ &= \Re\{h(n)\} - \Im\{h(n)\} \\ &= |h(n)| e^{-j\left(\arctan\left(\frac{\Re\{h(n)\}}{\Im\{h(n)\}}\right)\right)} \end{aligned}$$

Next, equating only the phase of the left-hand and right-hand sides:

$$2\pi \frac{f_0}{f_s} = -\arctan\left(\frac{\Re\{h(n)\}}{\Im\{h(n)\}}\right)$$

Bringing the minus sign into the arctan by taking the reciprocal of the fraction, then rearranging, we get (23) which completes the proof.

$$f_0 = \frac{f_s}{2\pi} \arctan\left(\frac{\Im\{h(n)\}}{\Re\{h(n)\}}\right) \tag{23}$$

For the unbalanced system, we start with the widely linear first order autoregressive model which has the form given in (24).

$$v(n+1) = h^*(n)v(n) + g^*(n)v^*(n) \tag{24}$$

Next, we utilize the Clarke Transform to express the voltage of a unbalanced system:

$$v(n) = A(n) e^{j\left(2\pi \frac{f_0}{f_s} n + \phi\right)} + B(n) e^{-j\left(2\pi \frac{f_0}{f_s} n + \phi\right)}$$

We substitute the Clarke Transform and its conjugate in (24):

$$v(n+1) = h^*(n)A(n) e^{j\left(2\pi \frac{f_0}{f_s} n + \phi\right)} + h^*(n)B(n) e^{-j\left(2\pi \frac{f_0}{f_s} n + \phi\right)} + g^*(n)A^*(n) e^{-j\left(2\pi \frac{f_0}{f_s} n + \phi\right)} + g^*(n)B^*(n) e^{j\left(2\pi \frac{f_0}{f_s} n + \phi\right)}$$

Next, changing the index of the Clarke Transform we obtain:

$$v(n+1) = A(n+1) e^{j\left(2\pi \frac{f_0}{f_s}(n+1) + \phi\right)} + B(n+1) e^{-j\left(2\pi \frac{f_0}{f_s}(n+1) + \phi\right)}$$

We can now substitute $v(n+1)$ in (24). We then simplify and group exponential terms together and get the following equations:

$$A(n+1)e^{j\left(2\pi\frac{f_0}{f_s}(n+1)+\phi\right)} = \big(h^*(n)A(n)+g^*(n)B^*(n)\big)e^{j\left(2\pi\frac{f_0}{f_s}n+\phi\right)}$$
$$B(n+1)e^{-j\left(2\pi\frac{f_0}{f_s}(n+1)+\phi\right)} = \big(h^*(n)B(n)+g^*(n)A^*(n)\big)e^{-j\left(2\pi\frac{f_0}{f_s}n+\phi\right)}$$

Next, making the assumption that $A(n+1)\approx A(n)$ and $B(n+1)\approx B(n)$, we can simplify the above equations to:

$$e^{j2\pi\frac{f_0}{f_s}} = \frac{h^*(n)A(n)+g^*(n)B^*(n)}{A(n+1)} \approx h^*(n)+g^*(n)\frac{B^*(n)}{A(n)} \tag{25}$$
$$e^{-j2\pi\frac{f_0}{f_s}} = \frac{h^*(n)B(n)+g^*(n)A^*(n)}{B(n+1)} \approx h^*(n)+g^*(n)\frac{A^*(n)}{B(n)} \tag{26}$$

Taking the conjugate of (26) and equating it to (25), we obtain:

$$h^*(n)+g^*(n)\frac{B^*(n)}{A(n)} = h(n)+g(n)\frac{A(n)}{B^*(n)}$$

Next, performing a change of variable such that $Y=\frac{B^*(n)}{A(n)}$, we get a quadratic equation in Y:

$$g^*(n)Y^2+\big(h^*(n)-h(n)\big)Y+g(n)=0$$

Solving for Y, we get:

$$\begin{aligned}Y &= \frac{-\big(h^*(n)-h(n)\big)\pm\sqrt{\big(h^*(n)-h(n)\big)^2-4g^*(n)g(n)}}{2g^*(n)}\\ &= \frac{2\Im\{h(n)\}j\pm\sqrt{-4\Im^2\{h(n)\}+4|g(n)|^2}}{2g^*(n)}\\ &= \frac{\Im\{h(n)\}j\pm j\sqrt{\Im^2\{h(n)\}-|g(n)|^2}}{g^*(n)}\end{aligned}$$

Substituting the solution of Y into (25), we obtain:

$$\begin{aligned}e^{j2\pi\frac{f_0}{f_s}} &= h^*(n)+\Im\{h(n)\}j\pm j\sqrt{\Im^2\{h(n)\}-|g(n)|^2}\\ &= \Re\{h(n)\}\pm j\sqrt{\Im^2\{h(n)\}-|g(n)|^2}\end{aligned}$$

Since $f_s>f_0>0$, we can eliminate one solution. Note that M is an arbitrary constant.

$$\begin{aligned}e^{j2\pi\frac{f_0}{f_s}} &= \Re\{h(n)\}+j\sqrt{\Im^2\{h(n)\}-|g(n)|^2}\\ &= |M|e^{j\left(\arctan\left(\frac{\sqrt{\Im^2\{h(n)\}-|g(n)|^2}}{\Re\{h(n)\}}\right)\right)}\end{aligned}$$

Next, equating only the phase of the left-hand and right-hand sides:

$$2\pi\frac{f_0}{f_s} = \arctan\left(\frac{\sqrt{\Im^2\{h(n)\}-|g(n)|^2}}{\Re\{h(n)\}}\right)$$

Rearranging, we get (27) which completes the proof:

$$f_0 = \frac{f_s}{2\pi}\arctan\left(\frac{\sqrt{\Im^2\{h(n)\}-|g(n)|^2}}{\Re\{h(n)\}}\right) \tag{27}$$

e. The value of f_0 was set at 50 Hz to match the operating frequency of power transmission in the UK. The figure below shows how the two algorithms perform when tested on a balanced system. The CLMS algorithm converges much faster than the ACLMS algorithm. The ACLMS also displays a big overshoot during the first two iterations, something that the CLMS algorithm does not do. Note that both algorithms converge to the nominal frequency of 50 Hz without any bias.

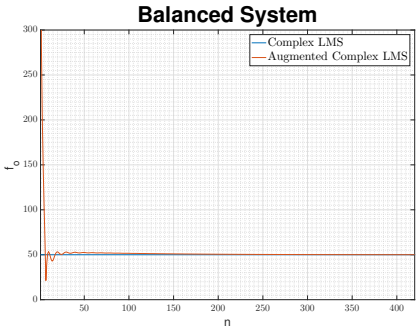


Figure 52: Comparison of CLMS and ACLMS for Balanced Complex Voltages

The figure below shows the results obtained when both algorithms are tested on an unbalanced system. Again, the CLMS algorithm converges much faster than the ACLMS algorithm. Also a large overshoot is once again observed in the ACLMS algorithm. However, the key difference is that the ACLMS algorithm converges to the nominal frequency of 50 Hz without a bias. The CLMS algorithm however does not converge to the nominal frequency. The CLMS algorithm is only able to describe circular data however we have seen above that the unbalanced system does not produce a circular plot. Using the CLMS algorithm to describe an unbalanced system is equivalent to attempting to describe an ellipse using a circle. While we only need the radius to describe a circle, an ellipse can only be fully describe when the length of both the major and minor axis are known.

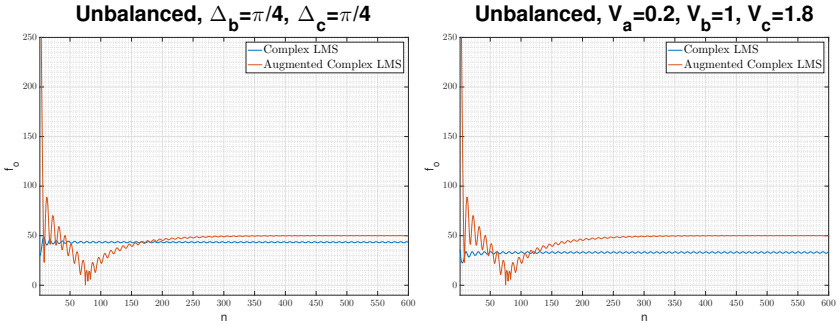


Figure 53: Comparison of CLMS and ACLMS for Unbalanced Complex Voltages

Below we see the circularity diagrams of the two unbalanced voltages that were tested using the CLMS and ACLMS algorithms. It is clear that the magnitude distorted complex voltage is less circular than the phase distorted complex voltage. As such, the bias introduced by the CLMS algorithm is smaller when estimating the frequency of the phase distorted complex voltage.

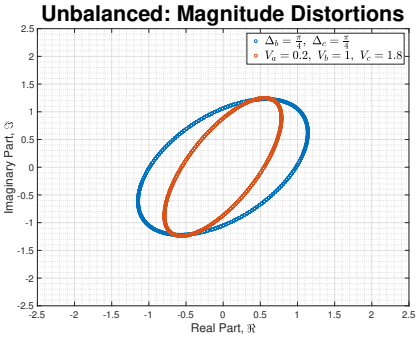


Figure 54: Circularity Plots for the two Complex Signals tested with the CLMS and ACLMS algorithms

4.2 Adaptive AR Model Based Time-Frequency Estimation

a. A frequency modulated signal is generated using circular white gaussian noise with zero mean and variance of 0.05. The figure below shows that the signal's frequency is not stationary. The figure also shows the frequency spectrum obtained if the entire signal is modeled as an AR process. Modeling the entire signal as an AR-process is not wise. The matlab function `aryule` assumes that the input signal is stationary. This is evidently not the case for this frequency modulated signal. The signal has 3 distinct stages in which the frequency is either constant, increasing linearly or increasing quadratically.

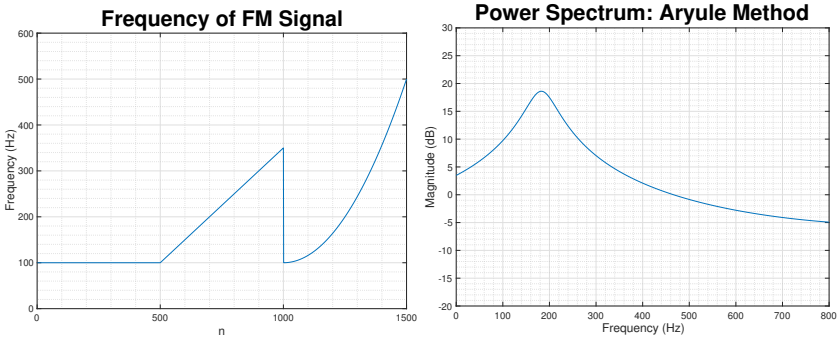


Figure 55: Original, Non-Stationary Frequency of FM Signal and Power Spectrum obtained using Block Based Estimate of AR Coefficients

It is possible to perform block based autoregressive modeling with a smaller block size. If three blocks each of 500 samples were generated and run through `aryule`, we would be able to model the first block which has a constant frequency using an AR(1) process. The non-stationarity present in the other 2 blocks would still not be captured by the block based AR process estimation procedure; especially since the questions asks us to model the frequency modulated signal as an AR(1) process. If we increase the model order, we will increase the number of poles that the frequency spectrum has but will still not obtain a good estimate since the linearly increasing and quadratically increasing frequencies do not exhibit stationarity. Below are power spectra obtained if blocked-based AR estimation with block sizes of 500 samples is performed for multiple model orders.

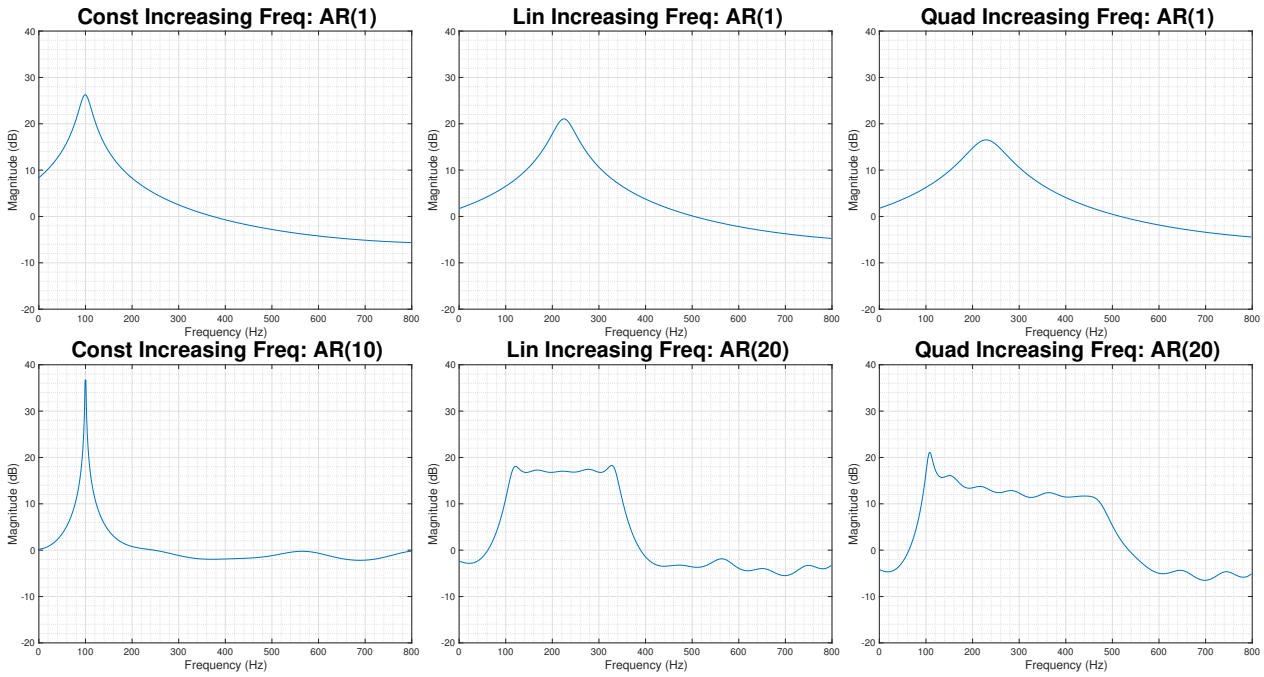


Figure 56: Block Based Estimate of AR Coefficients with Block Size of 500 Samples

The results are exactly as expected. With a smaller block size, the constant frequency segment of the frequency modulated signal can be modeled as an AR(1) process. The constant 100 Hz frequency is captured well. The non-stationary segments of the frequency modulated signal are still not well modeled. Moreover, increasing the model order does not result in any improvement.

b. Using the CLMS algorithm to infer the weights allows us to capture the non-stationarity in the data. We are now inferring the value of the AR coefficient in an adaptive manner rather than considering the entire dataset all at once. The figure below shows the results that are obtained. The impact that the choice of μ has is very clear. In the previous sections, the effect of μ on the rate of convergence was discussed at great length. The graphs below justify the need for fast convergence in a real-world application such as tracking the frequency of a signal. Notice that a large value of μ leads to a thicker yellow line. The thickness of the line indicates that there is large uncertainty in the exact frequency of the signal. Again this highlights the trade-off between convergence speed and steady-state error variation.

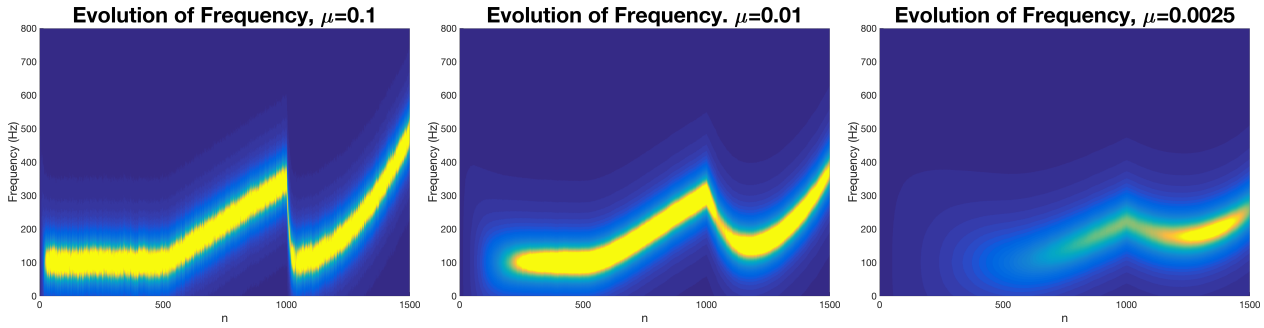


Figure 57: Time-Frequency Estimation using Complex LMS Algorithm with Different Values of μ

4.3 A Real Time Spectrum Analyser Using Least Mean Square

a. To solve the least squares problem, we start first with the cost function in (28)

$$\min_{\mathbf{w}} ||\mathbf{y} - \hat{\mathbf{y}}||^2 = \min_{\mathbf{w}} \sum_{n=0}^{N-1} |y(n) - \hat{y}(n)|^2 \tag{28}$$

We can start by expressing $||\mathbf{y} - \hat{\mathbf{y}}||^2$ as:

$$\begin{aligned} ||\mathbf{y} - \hat{\mathbf{y}}||^2 &= (\mathbf{y} - \hat{\mathbf{y}})^H (\mathbf{y} - \hat{\mathbf{y}}) \\ &= (\mathbf{y} - \mathbf{F}\mathbf{w})^H (\mathbf{y} - \mathbf{F}\mathbf{w}) \\ &= \mathbf{y}^H\mathbf{y} - \mathbf{w}^H\mathbf{F}^H\mathbf{y} - \mathbf{y}^H\mathbf{F}\mathbf{w} + \mathbf{w}^w\big(\mathbf{F}^H\mathbf{F}\big)\mathbf{w} \end{aligned}$$

To minimize with respect to $\mathbf{w}$, we differentiate the above equation with respect to $\mathbf{w}$:

$$0 - \mathbf{y}^H\mathbf{F} - \mathbf{y}^H\mathbf{F} + \mathbf{w}^H\big(\mathbf{F}^H\mathbf{F} + \mathbf{F}^H\mathbf{F}\big)$$

Equating the derivative to 0, cancelling out the constant multipliers and taking the transpose on each side, we get:

$$\begin{aligned} -\mathbf{y}^H\mathbf{F} - \mathbf{y}^H\mathbf{F} + \mathbf{w}\big(\mathbf{F}^H\mathbf{F} + \mathbf{F}^H\mathbf{F}\big) &= 0 \\ 2\mathbf{w}^H\mathbf{F}^H\mathbf{F} &= 2\mathbf{y}^H\mathbf{F}^H \\ \mathbf{F}^H\mathbf{F}\mathbf{w} &= \mathbf{F}\mathbf{y} \end{aligned}$$

Inverting the matrix $\mathbf{F}^H\mathbf{F}$ gives (29) which completes the proof:

$$\mathbf{w} = \big(\mathbf{F}^H\mathbf{F}\big)^{-1}\mathbf{F}\mathbf{y} \tag{29}$$

We know that the Inverse Discrete Fourier Transform (DFT), for a signal $\hat{x}(n)$, is given by the equation in (30):

$$\hat{x}(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(K)e^{j2\pi nk/N} \tag{30}$$

The formula in (30) can easily be expressed in the following matrix vector notation, where $W_{n,k} = e^{j2\pi nk/N}$:

$$\begin{bmatrix} \hat{x}(0) \\ \hat{x}(1) \\ \vdots \\ \hat{x}(N-1) \end{bmatrix} = \begin{bmatrix} W_{0,0} & W_{0,1} & \dots & W_{0,N-1} \\ W_{1,0} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ W_{N-1,0} & \dots & \dots & W_{N-1,N-1} \end{bmatrix} \begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix}$$

In compact notation,

$$\hat{\mathbf{x}} = \mathbf{W}\mathbf{X}$$

Using the matrix vector form of the Inverse DFT, it is clear that the Fourier Coefficients, $\mathbf{X}$, can be determined using the formula derived in (29), which is reproduced below:

$$\mathbf{X} = \left(\mathbf{W}^H\mathbf{W}\right)^{-1}\mathbf{W}\mathbf{x} \tag{31}$$

The Fourier coefficients, $\mathbf{X}$, that (31) returns would allow us to form a signal $\hat{\mathbf{x}}$ such that the squared error between $\mathbf{x}$ and $\hat{\mathbf{x}}$ is minimized.

b. Using the matrix vector form of the Inverse DFT, it is clear that $\hat{\mathbf{x}} = \mathbf{W}\mathbf{X}$ is an approximation of the signal $\mathbf{x}$. The approximation comes from the fact that $\mathbf{X}$ was formed by projecting $\mathbf{x}$ onto the column space of $\mathbf{W}$. Notice that the columns of $\mathbf{W}$ form an orthonormal basis consisting of complex exponentials. In contrast to the Continuous Time Fourier Transform, the DFT only requires a finite set of basis vectors, namely N for a signal $x(n)$ that has N non-zero components. The implicit assumption is that the signal $x(n)$ is periodic with period N.

In addition, since we are projecting onto an orthonormal basis, Parseval's theorem will hold. Parseval's theorem would not hold if the basis that the signal $x(n)$ is projected onto is not orthonormal.

c. The DFT-CLMS algorithm is applied to the Frequency Modulated signal and the results obtained are shown in the figure below. The rough shape of the spectrum observed somewhat matches the ideal frequency spectrum. However, it seems that frequency components are everlasting. Once the algorithm picks up a frequency component, that frequency component forever remains in the spectrum. In actual fact, this comes about from the fact that the weights are updated in an adaptive manner rather than being calculated at each time-instance. As such, for a frequency component to be removed from the spectrum, an error needs to be propagated back. The propagation of error takes some time and thus the frequency components seem to be everlasting.

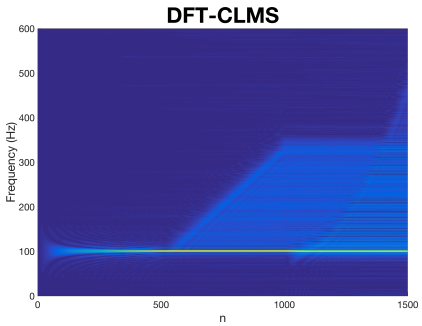


Figure 58: Time-Frequency Estimation using DFT-CLMS

To combat the slow trickle down effect that the back-propagation of the error has, the Leaky-LMS algorithm is used instead. This allows the weights to be "forgotten" and thus leads to a more accurate prediction of the weights. The value of γ was set at 0.05. The results obtained are significantly better. Notice that even such a small value of 0.05 was able have a tremendous effect on the accuracy of the weights. Increasing the value of γ beyond 0.05 did not prove useful. With $\gamma > 0.05$, the spectrum is effectively estimated using very few samples of the signal and the dominant frequency components are not correctly identified.

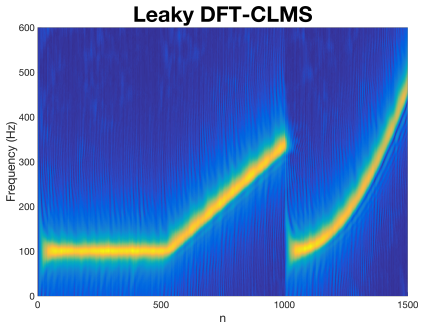


Figure 59: Time-Frequency Estimation using Leaky DFT-CLMS

d. The DFT-CLMS and Leaky DFT-CLMS algorithms are used to identify the spectrum of the EEG signal obtained from the POz location on the scalp. Both algorithms have identified clearly the strong component at 50 Hz. The first and second harmonics of the SSEVP have also been clearly identified at 13 Hz and 26 Hz. The third harmonic at 39 Hz is however indistinguishable.

Notice that the Leaky DFT-CLMS algorithm does not work as well as the DFT-CLMS algorithm in this scenario. The reason for this is due to the stationary nature of the EEG signal. The Leaky DFT-CLMS, to some extent, "forgets" weights that it learns. This amplifies the effect of placing a greater emphasis on recent samples rather than on samples that were observed a very long time ago. In so doing, it takes slightly longer to converge to a solution if the signal is stationary. The DFT-CLMS algorithm only 'forgets' weights using the backward error propagation mechanism, which is rather slow, and thus, in this case, is able to converge quickly. Not only does the DFT-CLMS algorithm converge more quickly, its ability to distinguish resonant components at 13, 26 and 50 Hz is also stronger; notice the brightness of the spectrogram at these frequencies.

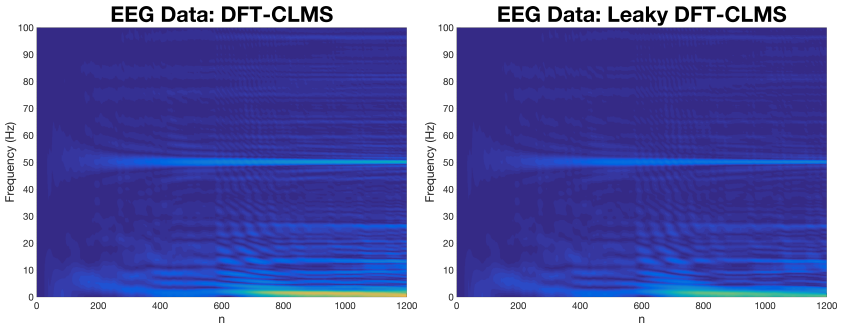


Figure 60: Time-Frequency Estimation of EEG Data

References

[1] Monson H Hayes. *Statistical digital signal processing and modeling*. John Wiley & Sons, 2009.

[2] Danilo P Mandic. A generalized normalized gradient descent algorithm. *IEEE signal processing letters*, 11(2):115–118, 2004.