Imperial College London

Adaptive Signal Processing and Machine Intelligence

Author: Zhuoda Han (zh3718)

CID: 01540269

Supervisor: Prof. Danilo Mandic

Department of Electrical and Electronic Engineering

February 26, 2019

Contents

Chapter 1

Classical and Modern Spectrum Estimation

1.1 Properties of Power Spectral Density (PSD)

One of the definitions of PSD is expressed in instruction and the power expression can be separated into conjugate product, which is

$$
\begin{aligned}
P(\omega) &= \lim_{N\to\infty} \mathbb{E}\left\{ \frac{1}{N} \left| \sum_{n=0}^{N-1} x(n) e^{-jn\omega} \right|^2 \right\} && (1.1)\\
&= \lim_{N\to\infty} \mathbb{E}\left\{ \frac{1}{N} \sum_{n=0}^{N-1} x(n) e^{-jn\omega} \sum_{m=0}^{N-1} x(m)^* e^{jm\omega} \right\} \\
&= \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} \mathbb{E}\Big\{ x(n) e^{-jn\omega} x^*(n) e^{jm\omega} \Big\} \\
&= \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} \mathbb{E}\Big\{ x(n) x^*(n) \Big\} e^{-j(n-m)\omega} && (1.2)
\end{aligned}
$$

In addition, the correlation of a complex signal can be expressed as

$$
\begin{aligned}
r_{xx}(n) &= \mathbb{E}\Big\{ x(k) x^*(k-n) \Big\} \\
r_{xx}(k-n) &= \mathbb{E}\Big\{ x(k) x^*(k-n) \Big\}
\end{aligned} \qquad (1.3)
$$

Therefore, the Equation 1.2 can be expressed as

$$
P(\omega) = \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} r_{xx}(n-m) e^{-j(n-m)\omega} = \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} g(n-m) \qquad (1.4)
$$

Then, the double summation of Equation 1.4 could be converted into single one based on $\sum_{n=-N}^{N} \sum_{m=-N}^{N} g(n-m) = \sum_{k=-2N}^{2N} (2N+1-|k|) g(k)$. Thus, Equation 1.4 is

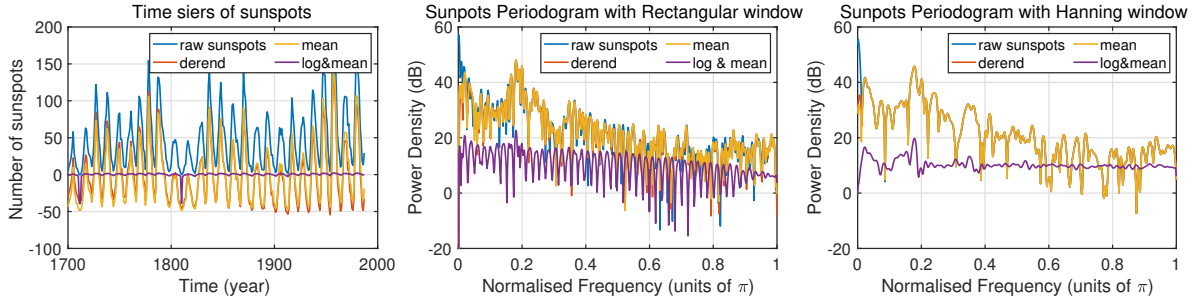
$$
\begin{aligned}
P(\omega) &= \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} g(n-m) \\
&= \lim_{N\to\infty} \frac{1}{N} \sum_{k=-(N-1)}^{N-1} (N-|k|) g(k) \\
&= \sum_{k=-\infty}^{\infty} r_{xx}(k) e^{-jk\omega} - \lim_{N\to\infty} \frac{1}{N} \sum_{k=-(N-1)}^{N-1} |k| r_{xx}(k) e^{-jk\omega} && (1.5)\\
&\approx \sum_{k=-\infty}^{\infty} r_{xx}(k) e^{-jk\omega} \ (under\ fast\ decays) && (1.6)
\end{aligned}
$$

With the assumption of rapidly decays, the latter part of Equation 1.5 tends to zero. Hence, the Equation 1.1 of PSD could be deduced to Equation 1.6.

1.2 Periodogram-based Methods Applied to Real-World Data

1.2.a The sunspot time series

As shown in Fig.1.1, the periodograms of raw sunspots data and its processed data are compared, including `detrend` in red line, `mean` in orange and `logarithm` in purple. In addition, the effects of applying rectangular window and Hanning window on spectral are illustrated as well.



(a): Time series **(b):** Rectangular window **(c):** Hanning window

Figure 1.1: Periodogram of processing sunspots data with rectangular and Hanning window

The results of using `detrend` and `mean` are similar. However, subtracting its mean leads to removing the DC component in zero frequency, while `detrend` results in eliminating the linear trend of a vector under low frequency region. Afterwards, the PSD curves of processing and raw data are same. As to centring after `log`, the curve is smoothly and the variance is reduced, as consequence of that the peaks are difficult to be detected.
Applying different windows will cause the changing of periodogram, due to the resolution of mainlobe width and sidelobe attenuation. Compared with Fig.1.1b and 1.1c, the curve is smoothly with less ripples when using the Hanning window. The reason is that the Hanning window has larger mainlobe width than rectangular window. Moreover, interest peaks can be effectively detected by applying `log`, since most of ripples are eliminated.

1.2.b The basis for brain computer interface (BCI)

Fig.1.2 illustrates periodagrams of the standard and Bartlett method. By applying Bartlett averaging method, the peaks of frequency are easily detected which are the range of $8-10$, 13, 26, 39 and 50 Hz. As stated in instruction, the strong response within $8-10$ Hz and 50 Hz are not the SSVEP[1]. Therefore, the fundamental frequency peak of the SSVEP is at $f = 13$ Hz with harmonics frequencies at $f = 26$ Hz and $f = 39$ Hz.

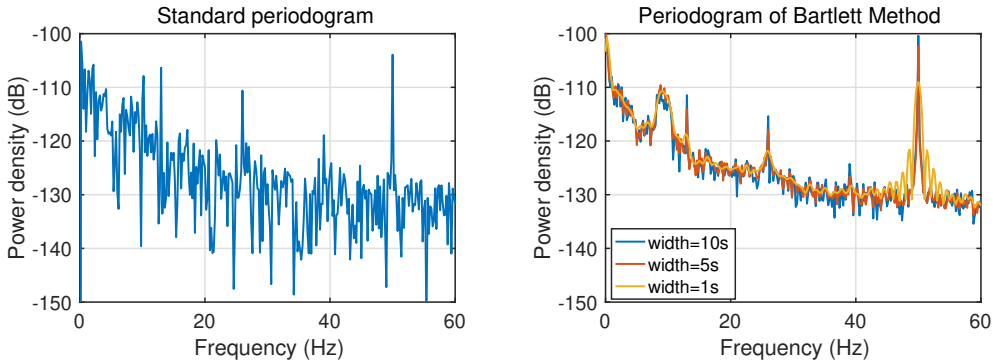


Figure 1.2: Standard and Bartlett Method of EEG Periodograms

With different length of windows, the total signal is divided into different length of segments. Fig.1.3 depicts the different periodograms with lengths of $\Delta t = 10s$ and $\Delta t = 1s$ windows. As to $\Delta t = 1s$, the periodogram is smooth with reduced variance. Nevertheless, the peak frequencies of interest are unapparent resulting in difficultly detecting the SSVEP[1]. Only alpha-rhythm peak at 8-10 Hz and interference at 50 Hz can be observed. When increasing the window length such as $\Delta t = 10s$, the number of segments decreases and each segment is longer. Hence, the variance of periodogram is reduced to a slight extent, resulting in distinct peak. However, the variance is sufficiently elimanitade compared with standard periodogram. Therefore, the trade-off between variance and precision need to be considered.

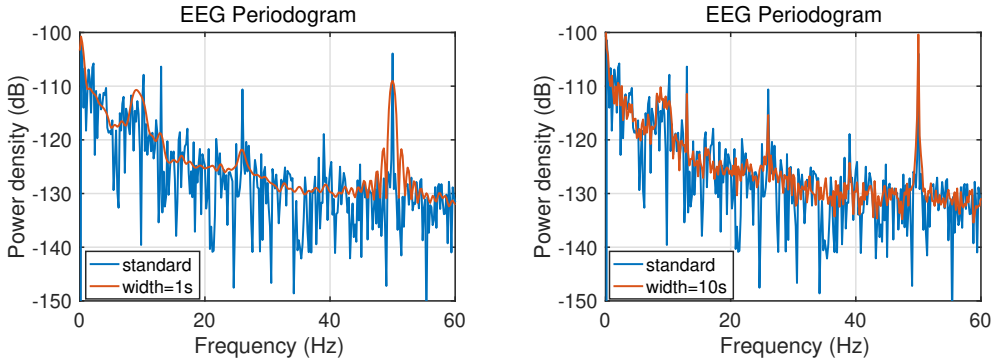


Figure 1.3: Bartlett method with different windows of EEG Periodograms

1.3 Correlation Estimation

1.3.a Biased and unbiased ACF and correlogram spectral

Fig.1.4 illustrates both unbiased and biased estimations of autocorrelation function (ACF) and correlogram spectral with WGN, filtered WGN and noisy sinusoidal signals. Observing the ACF diagrams, the biased and unbiased estimations are same when the lag k is approximately less than 200. As the lag k increases, the tendency is getting to separate in aspect of biased estimation increasing in value and the unbiased tending to zero, which verifies the Eq. 1.7 and1.8 based on the Eq. (12)-(13) in instruction.

$$\text{biased: } \mathbb{E}\left\{\hat{r}_{xx}(k)\right\} = \frac{1}{N}\sum_{n=k+1}^{N}\mathbb{E}\left\{x(n)x^*(n-k)\right\} = \frac{N-k}{N}\,r_{xx} \tag{1.7}$$

$$\text{unbiased: } \mathbb{E}\left\{\hat{r}_{xx}(k)\right\} = \frac{1}{N-k}\sum_{n=k+1}^{N}\mathbb{E}\left\{x(n)x^*(n-k)\right\} = r_{xx} \tag{1.8}$$

As to correlograms of these two ACF, the biased ACF guarantees the non-negative PSD due to the positive semi-definite of ACF. However, the unbiased ACF accords to true mean of PSD, resulting to highly erratic for large lags k. Thus, the ACF is not positive definite and causes negative values of the PSD, which are inappropriate with theory.

1.3.b PSD estimation with several realisations

The PSD estimations of the signal $x(n)$ in Eq.1.9 are plotted in Fig.1.5 with 100 realisitions.

$$x(n) = sin(2\pi 2n) + sin(2\pi 3n) + 1.5 * sin(2\pi 4n) + \omega(n) \quad \omega \sim \mathcal{N}(0,1) \tag{1.9}$$

The mean and standard deciation of the PSD are highlighted. The signal $x(n)$ has three frequency components with $f_1 = 2Hz$, $f_2 = 3Hz$ and $f_3 = 4Hz$ which are successfully detected in the PSD. It is obvious that

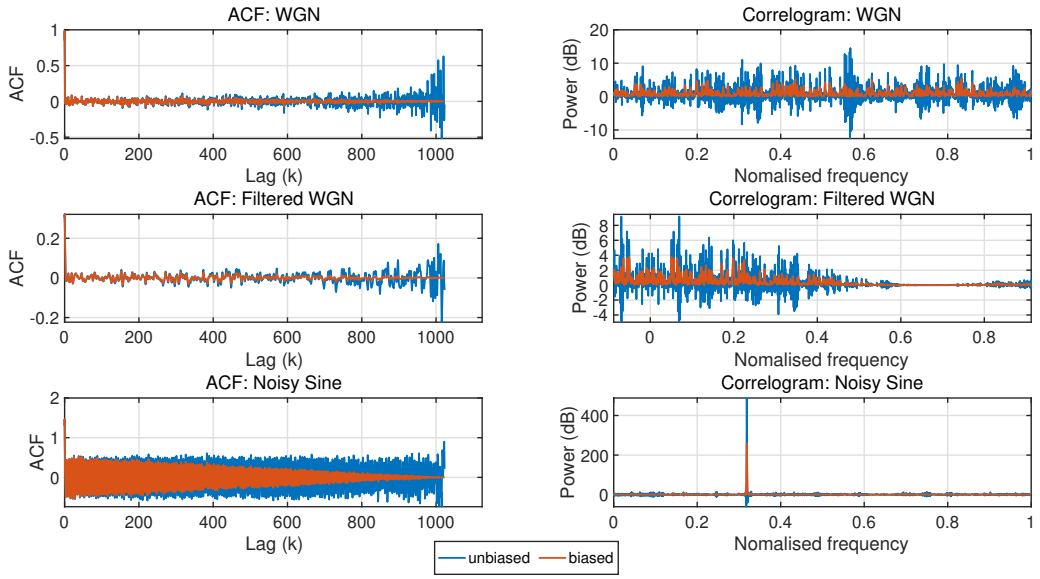


Figure 1.4: Standard and Bartlett Method of EEG Periodograms

the variance and noise are reduced by taking mean and standard deviation. Howerver, the peak value of the PSD is to sharp. Thus, the interval between realisitions is narrow and indistinct when the PSD estimations is in magnitude scale.

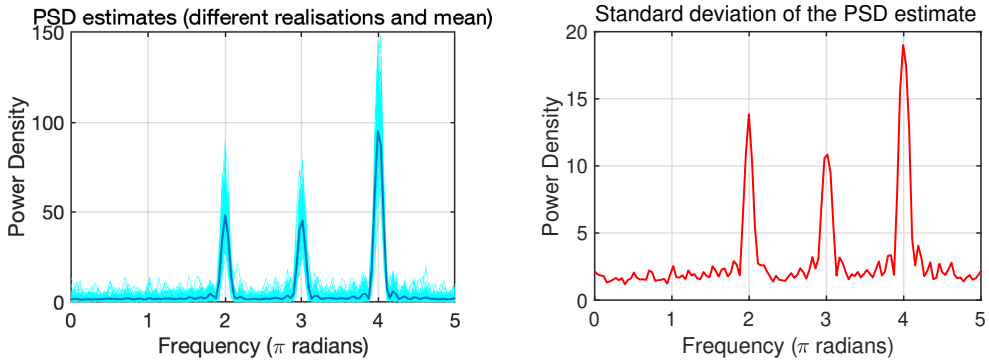


Figure 1.5: PSD estimations with mean and standard deviation

1.3.c PSD estimation in dB

Repeating the process in previous section, the PSD estimations are shown in Fig.1.6 in decibels scale. Observing the peak value of realisations, it was compressed to a large extent. As a consequence, the amplitudes of three sinusoid signals seem to be same. Meanwhile, the noise fluctuations are significantly amplified, which increases the variance of the PSD due to the logarithm. Overall, it is an admissible and advantageous presentations to plot the PSD estimations in dB since both large and small features are visible and distinct.

1.3.d Peak detection with window resolution

Fig.1.7 depicts the periodograms of peak detection of complex-valued signal by varying the number of sample. It is obvious that the two peaks are becoming with incremental of samples (N), since the frequency resolution

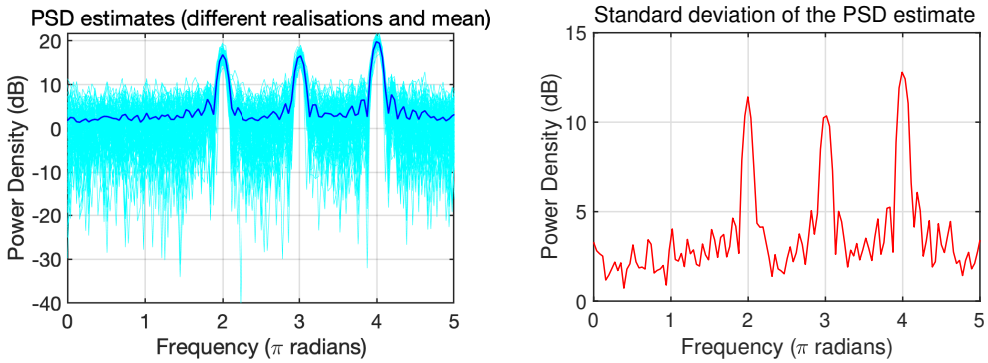


Figure 1.6: PSD estimations in dB

is proportional to $\frac{1}{N}$. In this experiment, the rectangular window was applied whose $3dB$ bandwidth is defined as $0.89(\frac{2\pi}{N})$. Therefore, with two frequencies in $0.3Hz$ and $0.32Hz$ in radian, the theoretical number of samples are $N = 0.89/(0.32 - 0.3) = 44.5$. Hence, the peaks can be successfully identified when the samples are larger than 45. However, when N is 40 as shown in Fig.1.7, peaks are still detected which is probably caused by the sidelobes of the window.

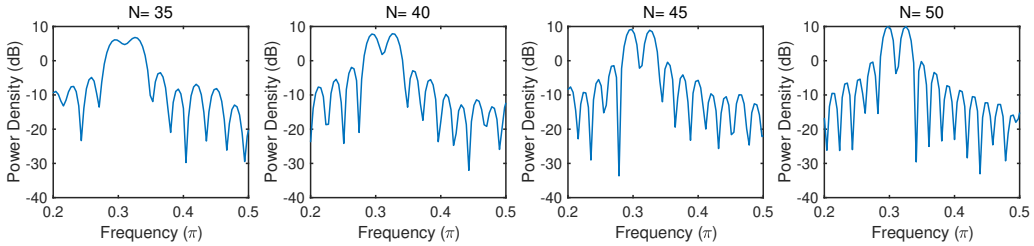


Figure 1.7: Periodogram: peak detection of varying samples

Fig.1.8 illustrates the peak detection with varying frequency interval from $0.32 \sim 0.35$. When fixing N to 30, the theoretical frequency interval $\Delta f = 0.89/30 \approx 0.3$. As shown in Fig.1.8, the results consist with the theory analyse where the peaks are detectable with $f2 \geq 0.33$.

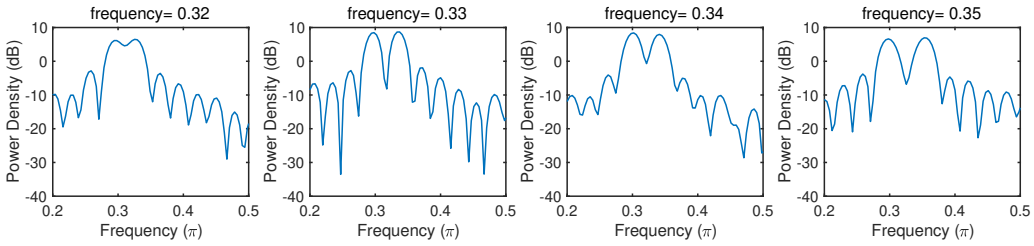


Figure 1.8: Periodogram: peak detection of varying frequencies

1.3.e Frequency estimation by MUSIC

The Multiple Signal (MUSIC) algorithm is an subspace method which focusing on the eigenvectors. A complex signal with AWGN can be expressed as $\mathbf{x}(n) = \mathbf{Ae} + \mathbf{w}$ on vector notation. Thus, its autocorrelation matrix is calculated and decomposed into the sum of signal subspace and noise subspace, as shown below.

$$\mathbb{E}(\mathbf{x}\mathbf{x}^H) = \mathbf{R}_{xx} = \mathbf{EDE^H} + \sigma^2\mathbf{I} = \mathbf{E_s D_s E_s^H} + \mathbf{E_n D_n E_n^H} \tag{1.10}$$

where $\mathbf{E_s} = [\mathbf{e_1}, ...\mathbf{e_p}]$ is the eigenvectors of signal, $\mathbf{E_s} = [\mathbf{v_{p+1}}, ...\mathbf{v_M}]$ is the eigenvectors of noise and $\mathbf{D} = diag[\mathbf{A_1}, ...\mathbf{A_p}, \sigma^2_{\mathbf{p+1}}...\sigma^2_{\mathbf{M}}]$ is the eigenvalues of subspace. Due to independence between signal and noise vectors, the signal subspace and noise subspace are orthogonal, expressing in $\mathbf{e_k^H v_i = 0}$. Therefore, the MUSIC algorithm is introduced by

$$\hat{P}_{MU}(\omega) = \frac{1}{\sum_{i=p+1}^{M} |\mathbf{e}^H \mathbf{v}_i|^2} \tag{1.11}$$

In this experiment, the function `corrmtx` is used to calculated the autocorrelation matrix $\mathbf{R}_{xx}$ in Eq.1.10 and `pmusic` is the MUSIC function to get the peak frequency in Eq.1.11. Meanwhile, the M is 14 and p is 2 which are defined as the total dimension of the subspace and the dimension of the signal subspace respectively.
Fig.1.9 depicts the estimation of the MUSIC function which is successfully detect the peaks. Comparing with the periodogram method, the MUSIC is using biased estimation and has less variance. Moreover, it can deal with less samples signal, while the periodogram method could only use long length signal due to the window resolution. However, the dimension of signal subspace p need to be determined in advance, which is not practical in general cases. On the contrast, the periodogram method does not require the information of the signal.

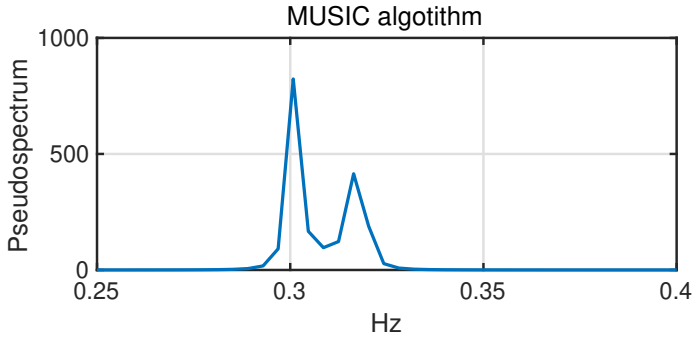


Figure 1.9: Periodogram: MUSIC algorithm

1.4 Spectrum of Autoregressive Processes

1.4.a Shortcomings of unbiased ACF

Based on the Yule-Walker equations, the autoregressive parameters can be obtained by the inversion of autocorrelation matrix, as expressed in Eq.1.12.

$$\mathbf{r}_{xx} = \mathbf{R}_{xx}\mathbf{a} \quad \Rightarrow \quad \mathbf{a} = \mathbf{R}_{xx}^{-1}\mathbf{r}_{xx} \tag{1.12}$$

Therefore, the ACF matrix $\mathbf{R}_{xx}$ is positive semi-definite for biased estimator which is invertible. As to unbiased ACF shown in previous section, the autocorrelation matrix may be singular which can not be inverted.

1.4.b AR Modelling with few samples

Given the autoregressive parameters $\mathbf{a}$ =[2.76 -3.81 2.65 -0.92] and the white noise power $\sigma^2 = 1$, the estimation of AR modelling process with the order p from 2 to 14 is shown in Fig.1.10. The AR model has a frustrated performance with low order which only detects one peak. With the order increasing, the estimation is approaching to the actual frequency response. However, the AR(4) model perform poorly although it consists with the true order of filter. This is probably caused by the small number of available samples with $N = 500$. Therefore, in order to obtain desirable response, either the order of AR model or the number of sample should increase.

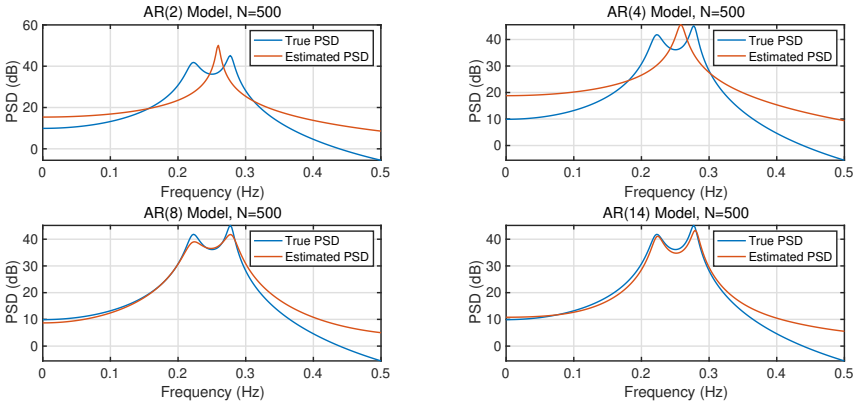


Figure 1.10: AR Model with varying order

1.4.c AR modelling with increasing samples

Repeat the process in previous section except applying $10k$ samples to model. When the AR order $p < p_{actrual=4}$, the autoregressive parameters can not be estimated. As shown in Fig.1.11, the AR(2) model only captures one peak frequency, which is an under-modelling estimation. However, the AR model matches the actual response with two peaks when the order p is larger than 4. Moreover, increasing order ($p > 5$) brought less improvement between actual response and estimation. Therefore, the optimal order for the large number of sample is 4^{th}.

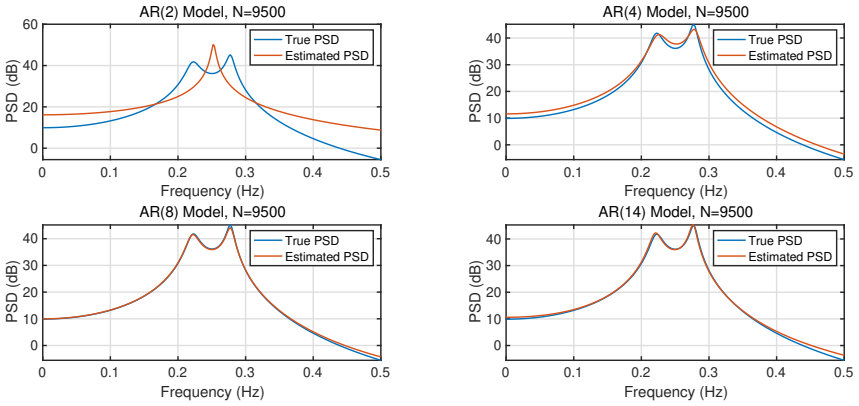


Figure 1.11: AR Modelling with $10k$ samples

1.5 Respiratory Sinus Arrhythmia from RR-Intervals

1.5.a PSD of RRI data

After processing the ECG data to RRI data in three trails, the standard and the Bartlett averange periodograms are plotted with rectangular window length $L \in [50s, 100s, 150s]$. Based on the analysis in section 1.2.2, the standard periodogram has relatively large variance and less leakage. With the decreasing length of window, the periodogram is tending to be smooth, resulting in both low variance and precision.

1.5.b Frequency of three trails

The experiment recorded the ECG data of three trails with normal, fast and slow breathing respectively. Therefore, the breaths per minute (BMP) for three trails will be different. A range of 12 $\sim$ 20 BMP is

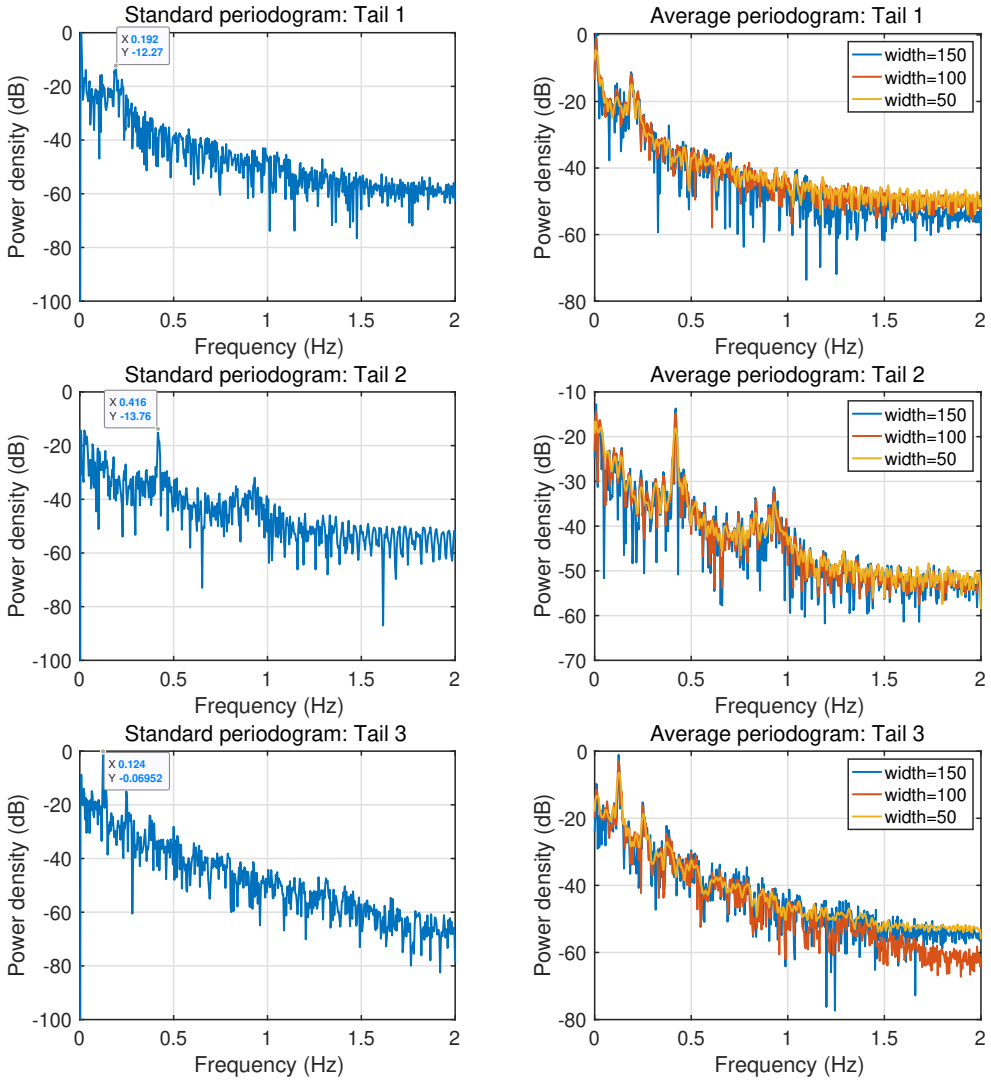


Figure 1.12: PSD of RRI data

considered as the reference range for the normal breath. The observed peaks for three trails are $0.192Hz$, $0.416Hz$ and $0.124Hz$, corresponding to 23.04, 49.92 and 14.88 in BMP respectively. However, the harmonics frequencies of Trail 1 are failed to be captured which is probably caused by no restriction of the breath experiment. Trail 2 illustrate one harmonic at $f = 0.832Hz$ and a large response at $0.928Hz$ at the same time, reasons of which may be the noise and the resolution of window. In addition, Trail 3 detects three harmonics frequencies at $0.248Hz$, $0.372Hz$ and $0.496Hz$.

1.5.c AR modelling of RRI data

Fig.1.13 depicts the AR model estimations with the incremental of order p. For Trail 1, the optimal order $p = 10$ which detects the approximate peak at $f = 0.192Hz$. The order of Trail 2 at $p = 6$ can only detect the fundamental peak while harmonics can be detected with higher order. As to Trail 3, only when the order $p \geq 2$, the peaks can be identified whereas the harmonics are difficult to be detected even if the order is high. Hence, the under-modelling causes the failed detection of peak and the over-modelling leads to capture harmonics and noise peaks. Compared with standard and averaging periodogram methods, the AR model performs powerfully in detecting interest peak and reducing variance as long as the order is determined.

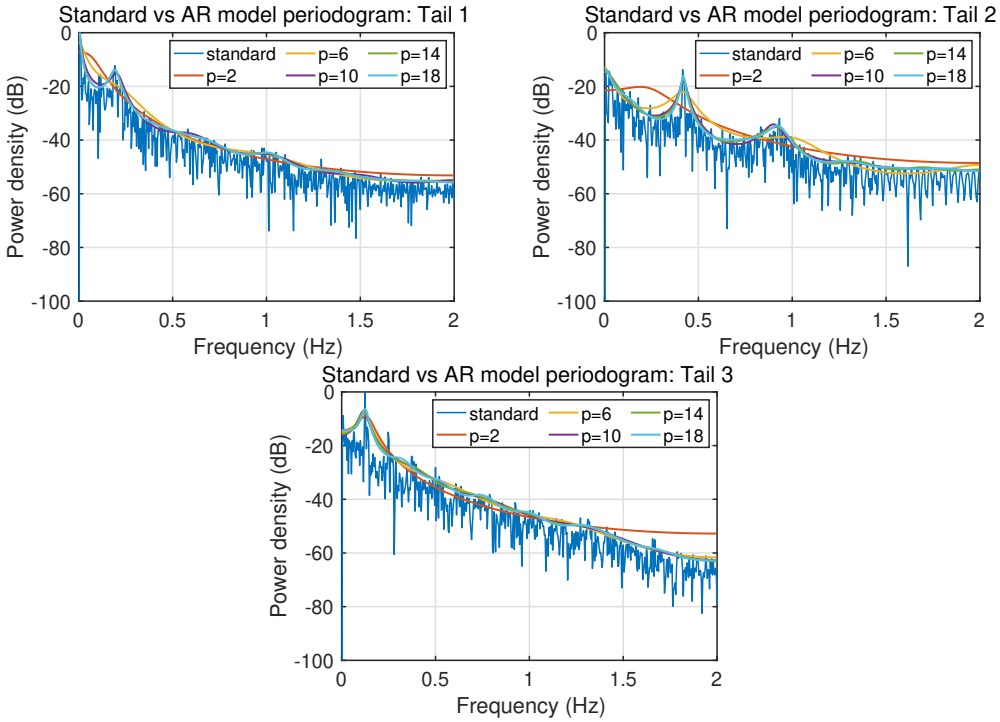


Figure 1.13: AR Model of RRI data

1.6 Robust Regression

1.6.a Singular values decomposition (SVD)

After applying the SVD to both $\mathbf{X}$ and $\mathbf{X}_{noise}$ data, the Fig.1.14 illustrates the singular values. For the original data $\mathbf{X}$, there are only three singular values which corresponding to its rank. Due to the randomness and independences of noise, the matrix $\mathbf{X}_{noise}$ is full rank which has ten singular values. However, there are three significant singular values which represents the dimension of signal subspace. The magnitude of other non-zero singular values are approximate half of the signal singular values, whose difference can be used to detect the signal and noise subspace. Nevertheless, if the noise power is large, it becomes hard to identify the rank of $\mathbf{X}_{noise}$.

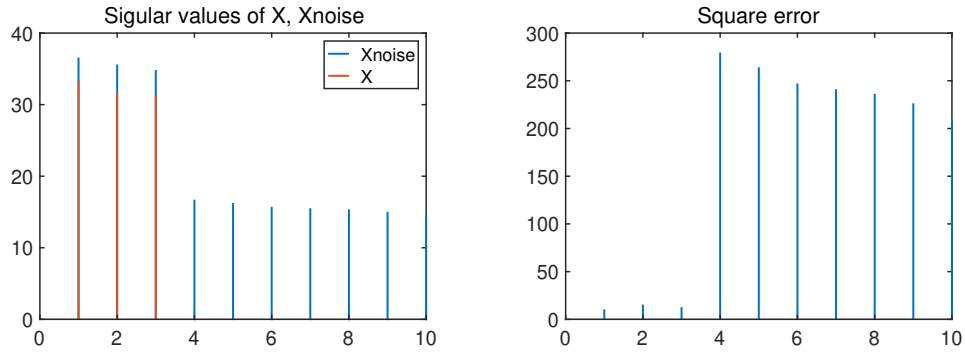


Figure 1.14: SVD of $\mathbf{X}$ and $\mathbf{X}_{noise}$

1.6.b Low rank approximation

The SVD algorithm could be used to recover the original matrix. Since the noise power is much less than the signal power, the recovered matrix is noiseless if only k significant values are concerned, where k is the rank of the original matrix. Fig.1.15 shows the error between the noise matrix and recovered noiseless matrix. As to the noiseless error curve, when the number of k is equal to the actual rank, the error reaches the bottom point. In this experiment, the minimum point at rank 3 is 27.07, while it is 49.34 of noise error over all rank. Therefore, the rank and the dimension of signal subspace are determined.

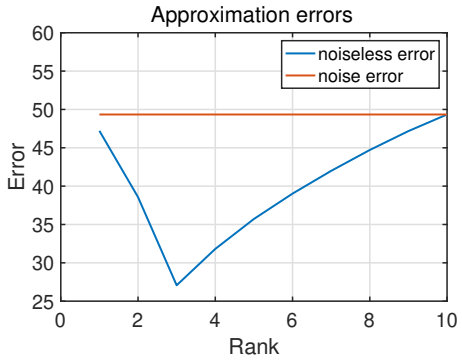


Figure 1.15: Error in low rank approximation

1.6.c OLS vs PCR

The parameter matrix $\mathbf{B}$ is calculated based on the OLS and PCR method. Meanwhile, the square error between the actual $\mathbf{Y}$ and estimated output on training and testing data are illustrated in Fig.1.16. When the number of significant components k is larger than 3, the PCR has the same performance with the OLS method. However, when testing the parameter matrix via $\mathbf{X}_{test}$, the estimated error of the PCR increases with $k < 3$, while the error of OLS decreases.

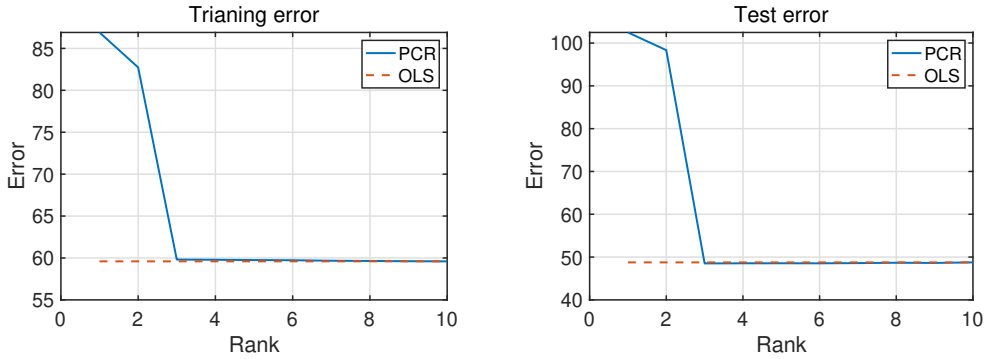


Figure 1.16: Training and testing error of OLS & PCR

1.6.d Realisations of OLS and PCR

Totally 50 realisations were simulated by OLS and PCR methods and its average error with test data are plotted in Fig.1.17. Compared with the testing error shown in Fig1.16, the average errors of both PCR and OLS algorithms are reduced to a large extent. The largest error of PCR is reduced from 85 to 65, while the

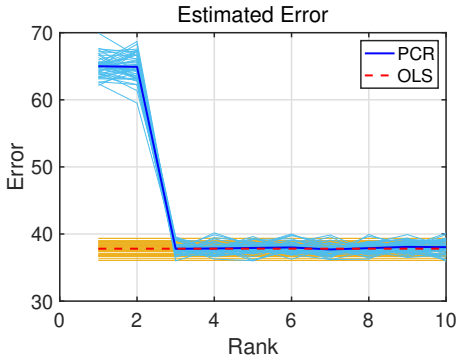


Figure 1.17: Estimated error of realisations

OLS decreases to 37.81.

Chapter 2

Adaptive signal processing

2.1 The Least Mean Square (LMS) Algorithm

2.1.a Correlation matrix

The autocorrelation matrix is defined [2] as $\mathbf{R}_{xx} = \mathbb{E}\{\mathbf{x}(n)\mathbf{x}^T(n)\}$ and $\mathbf{x}(n)$ is $[x(n-1), x(n-2)]^T$, thus $\mathbf{R}_{xx}$ is

$$\mathbf{R}_{xx} = \mathbb{E}\left\{\begin{bmatrix} x(n-1)x(n-1) & x(n-1)x(n-2) \\ x(n-2)x(n-1) & x(n-2)x(n-2) \end{bmatrix}\right\} = \begin{bmatrix} \mathbf{r}_{xx}(0) & \mathbf{r}_{xx}(1) \\ \mathbf{r}_{xx}(1) & \mathbf{r}_{xx}(0) \end{bmatrix} \tag{2.1}$$

Given by the $x(n)$ is a second-order auto-regressive process as

$$x(n) = a_1 x(n-1) + a_2 x(n-2) + \eta(n) \quad where \quad \eta(n) \sim \mathcal{N}(0, \sigma_\eta^2) \tag{2.2}$$

the $\mathbf{r}_{xx}(0)$ and $\mathbf{r}_{xx}(1)$ can be calculated by following steps due to the correlation matrix $\mathbf{r}_{xx}(k) = \mathbb{E}\{x(n)x(n-k)\}$

$$\begin{aligned} \mathbf{r}_{xx}(0) &= \mathbb{E}\{x(n)x(n)\} \\ &= \mathbb{E}\{a_1^2 x^2(n-1) + a_2^2 x^2(n-2) + 2a_1 a_2 x(n-1)x(n-2) \\ &\quad + \eta(n)[a_1 x(n-1) + a_2 x(n-2)]\} + \sigma_\eta^2 \\ &= a_1^2 \mathbf{r}_{xx}(0) + a_2^2 \mathbf{r}_{xx}(0) + 2a_1 a_2 \mathbf{r}_{xx}(1) + \sigma_\eta^2 \end{aligned} \tag{2.3}$$

Cause the noise subspace is orthogonal with signal subspace, the product term $\mathbb{E}\{\eta(n)[a_1 x(n-1)+a_2 x(n-2)]\}$ is zero. In the same way, the $\mathbf{r}_{xx}(1)$ can be calculated as well as shown in Eq.2.4.

$$\begin{aligned} \mathbf{r}_{xx}(1) &= \mathbb{E}\{x(n)x(n-1)\} \\ &= \mathbb{E}\{a_1 x^2(n-1) + a_2 x(n-1)x(n-2) + \eta(n)x(n-1)\} \\ &= a_1 \mathbf{r}_{xx}(0) + a_2 \mathbf{r}_{xx}(1) \end{aligned} \tag{2.4}$$

Therefore, substituting Eq.2.3 to Eq.2.4, the unique solutions of $\mathbf{r}_{xx}(0)$ and $\mathbf{r}_{xx}(1)$ are $\frac{25}{27}$ and $\frac{25}{54}$ respectively. The autocorrelation matrix $\mathbf{R}_{xx}$ is

$$\mathbf{R}_{xx} = \begin{bmatrix} \frac{25}{27} & \frac{25}{54} \\ \frac{25}{54} & \frac{25}{27} \end{bmatrix} \tag{2.5}$$

As to LMS algorithm, the convergence step μ is defined in the range of $0 < \mu < \frac{2}{\lambda_{max}}$., where λ_{max} is the maximum eigenvalue of $\mathbf{R}_{xx}$. Applying eigendecomposition to autocorrelation matrix, the eigenvalues are approximate 0.463 and 1.3889. Taking the large one into account, the range of step μ is

$$0 < \mu < \frac{2}{1.3889} = 1.44 \tag{2.6}$$

2.1.b LMS filter

The LMS adaptive filter is implemented to estimate AR model's weights using $N = 1000$ samples with 100 realizations. As a comparison of convergence, tow different steps $\mu = 0.01,\ 0.05$ are considered within the limited range in Eq.2.6. Fig.2.1 shows the estimated errors of single realization and mean error with two steps.

As to a single realization, the impact of descent step is not significant, since the white noise signal is randomly. However, taking account into mean error of 100 realizations, the convergence speed of large steps is faster than the smaller one. Specifically, the learning curve converges approximately after 100 samples at $\mu = 0.05$, while it is 250 samples at $\mu = 0.01$. Nevertheless, large step size causes significant fluctuation as well which should be trade-off.

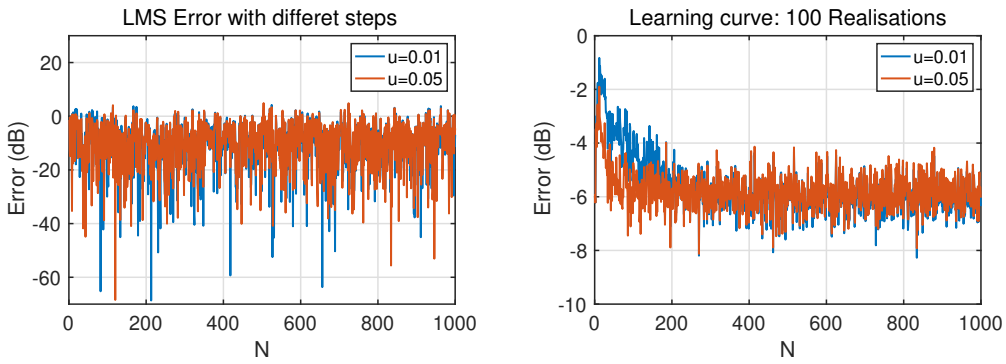


Figure 2.1: LMS estimated error with different realizations and steps

2.1.c Misadjustment of LMS

The theoretical misadjustent of the LMS is approximated as $\mathcal{M}_{LMS} \approx \frac{\mu}{2}\text{Tr}\{\mathbf{R}_{xx}\}$, where the autocorrelation matrix $\mathbf{R}_{xx}$ is calculated. However, the estimated misadjustment is the ratio of excess MSE and minimum MSE, which is $\mathcal{M} = \frac{MSE-\sigma^2}{\sigma^2}$. In addition, in order to guarantee the error calculated during steady state, the samples are selected after 400. Table.2.1 shows the values of misadjustment. As shown in table, the measured MSE and misadjustment is slightly larger than theoretical values. However, small step size introduces a small misadjustment, whereas the convergence speed is slow..

Table 2.1: Theoretical and Actual Misadjustment values for step sizes

μ	$\mathcal{M}_{LMS}$	$\mathcal{M}$
0.01	0.0093	0.0134
0.05	0.0463	0.0534

2.1.d LMS: estimated weights

Fig.2.2 depicts the estimated weights $a_1 = 0.1$ and $a_2 = 0.8$ with steps $\mu = 0.01$ and $\mu = 0.05$. Compared these estimated weights with true values, small step size provides an acceptable steady state error which is closer to actual weights. However, the convergence speed is relatively slow compared with large step size. On the contrast, when $\mu = 0.05$, the convergence speed is significantly improved, resulting in a large difference with actual values. Thus, there is a trade-off between steady state error and convergence speed.

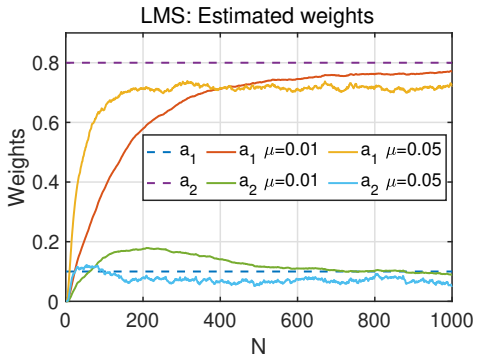


Figure 2.2: LMS estimated weights with different steps

2.1.e Leaky LMS

For the leaky LMS, the cost function is

$$\mathcal{J}_2(n) = \frac{1}{2}(e^2(n) + \gamma||\mathbf{w}(n)||_2^2) \tag{2.7}$$
$$where \quad e(n) = d(n) - \mathbf{w}_T(n)\mathbf{x}(n)$$

To get the minimum squared error, the gradient with weight $\mathbf{w}$ of Eq.2.7 is taken,

$$\begin{aligned}\nabla\mathcal{J}_2(n) &= \frac{1}{2}\partial(e^2(n) + \gamma||\mathbf{w}(n)||_2^2)/\partial\mathbf{w} \\ &= \frac{1}{2}(\frac{\partial(e^2(n))}{e(n)}\frac{\partial(e(n))}{\mathbf{w}} + 2\gamma||\mathbf{w}(n)||_2) \\ &= -e(n)\mathbf{x}(n) + \gamma\mathbf{w}(n)\end{aligned} \tag{2.8}$$

Therefore, the updated weight is

$$\begin{aligned}\mathbf{w}(n+1) &= \mathbf{w}(n) + \mu(-\nabla\mathcal{J}_2(n)) \\ &= \mathbf{w}(n) + \mu(e(n)\mathbf{x}(n) - \gamma\mathbf{w}(n)) \\ &= (1-\mu\gamma)\mathbf{w}(n) + \mu e(n)\mathbf{x}(n)\end{aligned} \tag{2.9}$$

2.1.f Leaky LMS: estimated weights

Fig.2.3 illustrated the estimated weights using leaky LMS filter with $\gamma = 0.2$, 0.4, 0.6 and $\mu = 0.01$, 0.05. However, the estimated values converges to a certain value with a large difference with actual coefficient. With the incremental of γ, the steady state error significantly rises up. Observing weights curves, increasing γ introduces a larger bias for lager weight than small weight. Thus, Leaky LMS causes large penalty for the large weight estimation.

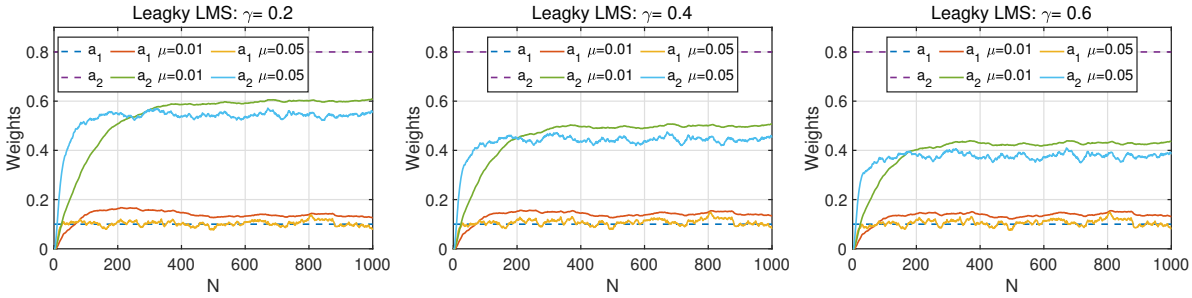


Figure 2.3: Leaky LMS estimated weights with different γ and μ

As to the general LMS algorithm, the optimal weight is

$$\mathbf{w}_{opt} = \mathbf{R}_{xx}^{-1}\mathbf{p} \quad \text{where} \quad \mathbf{p} = \mathbb{E}\{\mathbf{x}(n)d(n)\} \tag{2.10}$$

The autocorrelation matrix is semi-positive definite and can be invertible. However, the gradient vanishing problem is suffered when the eigenvalues of $\mathbf{R}_{xx}$ are zeros, resulting in divergence to expected values. By adding a small number γ of $\mathbf{R}_{xx}$, the weights can converge, which is the purpose of Leaky LMS. Thus, the optimal weight is

$$\mathbf{w}_{opt} = (\mathbf{R}_{xx} + \gamma\mathbf{I})^{-1}\mathbf{p} \tag{2.11}$$

Nevertheless, the eigenvalues of autocorrelation matrix are non-zeros for this experiment. Hence, adding large value of γ leads to an obvious bias and the estimated weights converge to incorrect coefficients.

2.2 Adaptive Step Size

2.2.a GASS

Based on the previous analysis, the step size should be large for fast convergence and getting small to reduce steady state errors. The gradient adaptive step-size (GASS) is implemented with time-varying step $\mu(n)$. The MA(1) process $x(n) = 0.9\eta(n-1)+\eta(n)$ with white noise $\eta \sim \mathcal{N}(0, 0.5)$ is simulated. For the GASS, the gradient step will be controlled by a constant ρ and $\psi(n)$. In addition, there are three algorithms which will update the $\psi(n)$. As introduced in guidelines, the Benveniste applies a time-varying adaptive filter, which provides low pass filtering of the instantaneous gradient [3]. Thus, the Benveniste's algorithm is robust to the noise and should be more accurate. The Ang & Farhang's algorithm replaces the low-pass filter term with a constant α. And the Matthews & Xie's algorithm simplifies the the algorithm by Ang & Farhang by setting α to zero, which only use the instantaneous gradient to update. Hence, the performance for this algorithm should be relative poor.

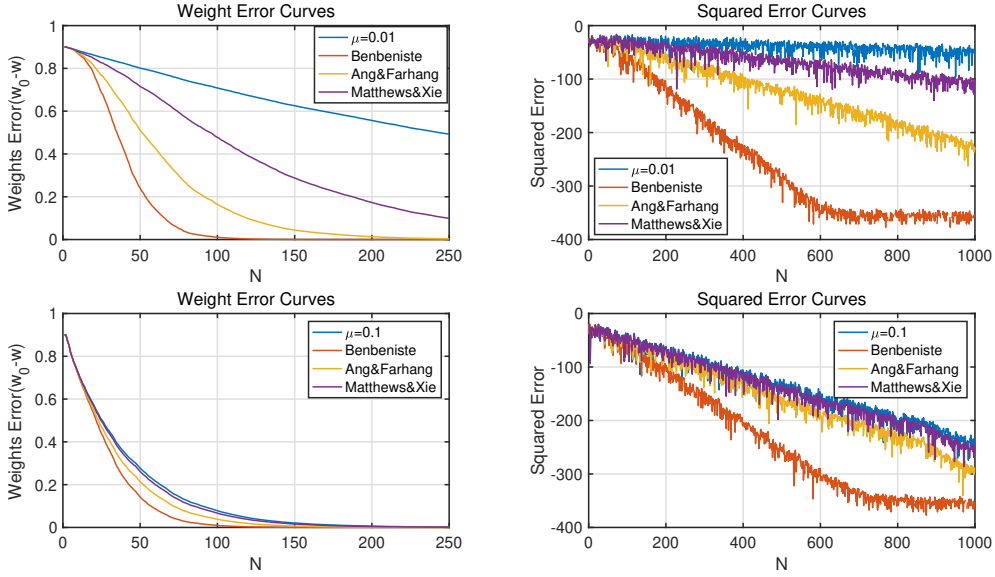


Figure 2.4: GASS estimated weights error and squared error

Fig.2.4 depicts the weight error and learning curves of three algorithms and standard LMS. When setting the initial μ =0.01, the standard LMS with fixed step has the slowest convergence speed and large steady state error. The Benveniste's algorithm converges before 100 samples, whereas the Ang & Farhang's converges at 200 samples and Matthews is after 250 samples. However, even if the Matthews's algorithm perform worst among GASS algorithm, it is still slightly better than the standard LMS. When increasing the initial step $\mu = 0.1$, all of algorithms converge rapidly with squared error lower than -300dB. However, the GASS algorithm requires more computational complexity than the standard LMS.

2.2.b NLMS

Given a the update weight $\mathbf{w}(n+1) = \mathbf{w}(n) + \mu e_p(n)\mathbf{x}(n)$, the relationship between posteriori error $e_p(n)$ and priori error $e(n)$can be derived by

$$\begin{aligned} e_p(n) &= d(n) - \mathbf{x}^T(n)\mathbf{w}(n+1) \\ &= d(n) - \mathbf{x}^T(n)\mathbf{w}(n) - \mu e_p(n)\mathbf{x}^T(n)\mathbf{x}(n) \\ &= e(n) - \mu e_p(n)\|\mathbf{x}(n)\|^2 \\ &= \frac{e(n)}{1+\mu\|\mathbf{x}(n)\|^2} \end{aligned} \tag{2.12}$$

Substituting Eq.2.12 into update equation,

$$\begin{aligned}\mathbf{w}(n+1) &= \mathbf{w}(n) + \mu e_p(n)\mathbf{x}(n) \\ &= \mathbf{w}(n) + \frac{\mu}{1+\mu\|\mathbf{x}(n)\|^2}e(n)\mathbf{x}(n) \\ &= \mathbf{w}(n) + \frac{1}{\frac{1}{\mu}+\|\mathbf{x}(n)\|^2}e(n)\mathbf{x}(n) \\ &= \mathbf{w}(n) + \frac{\beta}{\epsilon+\mathbf{x}^T(n)\mathbf{x}(n)}e(n)\mathbf{x}(n) \end{aligned} \tag{2.13}$$

Thus, the update equation based on a posteriori error is equivalent to the NLMS algorithm, where $\epsilon = \frac{1}{\mu}$ and $\beta = 1$.

2.2.c GNGD vs GASS

Based on the NLMS algorithm, the generalized normalized gradient descent (GNGD) applies a time-varying $\epsilon(n)$. Compared with the Benveniste's algorithm, the GNGD algorithm converges rapidly of weight estimation approximately at 40 samples when the initial step $\mu = 0.1$. The squared error for the GNGD is smaller as well. However, when increasing the initial step $\mu = 1$, the Benveniste's algorithm converges faster than the GNGD. Thus, the performance of the GASS algorithm is significantly affected by initial step μ, while it is not a problem for the GNGD algorithm.

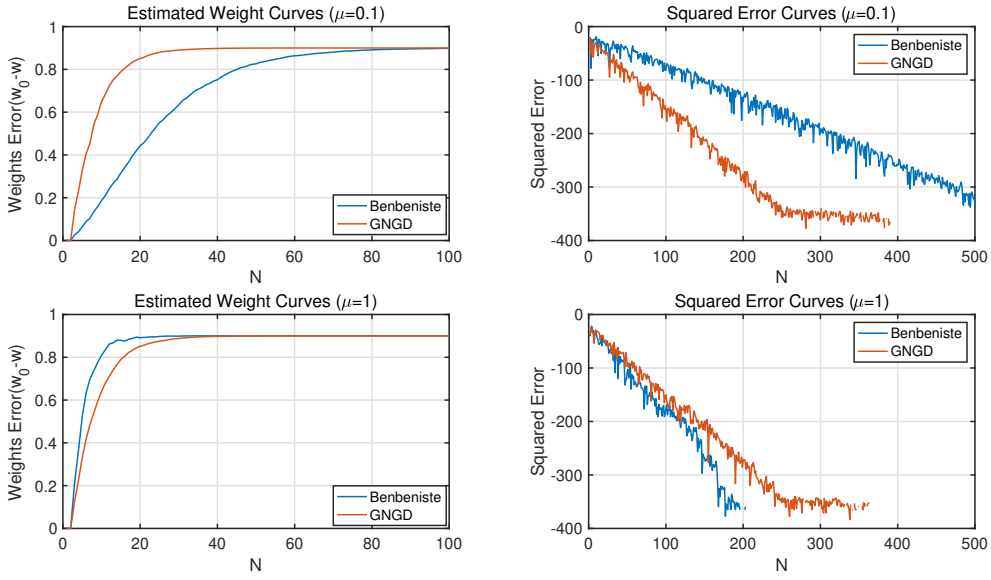


Figure 2.5: GAGN and Benveniste's GASS estimated weights and squared error

As to the complexity of these two algorithms, the computation complexity is caused by the matrix product in step update process. The Benveniste's algorithm, as shown below,

$$\psi(n) = [\mathbf{I} - \overbrace{\underbrace{\mu(n-1)\mathbf{x}(n-1)\mathbf{x}^T(n-1)}_{N^2}]\psi(n-1) + \underbrace{e(n-1)\mathbf{x}(n-1)}_{N}}^{2N} \tag{2.14}$$

has $O(N^2)$ multiplications, if the model order is N. The reason is that the Benveniste's algorithm applies matrix outer production which introduces a matrix result. Therefore, the computational complexity of the

Benveniste grows quadratically. For the GNGD, as shown below,

$$\epsilon(n+1) = \epsilon n - \rho\mu \frac{e(n)e(n-1)\overbrace{\mathbf{x}^T(n)\mathbf{x}(n-1)}^{N}}{(\epsilon(n-1) + \underbrace{\|\mathbf{x}(n-1)\|^2}_{N})^2} \tag{2.15}$$

It only has inner product of $\mathbf{x}(n)$, resulting in a numerical value. Hence, the computational complexity is reduced over order M which is $O(N)$.

2.3 Adaptive Noise Cancellation

2.3.a Delay of ALE

Due to the uncorrelation between interest signal and noise signal, the white noise can be eliminated by the delay version of the signal. The adaptive line enhancer (ALE) is used the delay of the noise-corrupted signal $s(n)$ to estimate the interest signal $\hat{x}(n)$. The optimal delay Δ can be calculated at the beginning of the Mean Squared Error.

$$\begin{aligned}\mathbb{E}\{(s(n) - \hat{x}(n))^2\} &= \mathbb{E}\{(x(n) + \eta(n) - \hat{x}(n))^2\} \\ &= \underbrace{\mathbb{E}\{\eta(n)^2\}}_{\eta^2} + \underbrace{\mathbb{E}\{(x(n) - \hat{x}(n))^2\}}_{\approx 0} + \underbrace{2\mathbb{E}\{(x(n) - \hat{x}(n))\eta(n)\}}_{minimize}\end{aligned} \tag{2.16}$$

Therefore, for the optimal estimation, the MSE should be equal to the noise power. That is, only the last term should minimize.

$$\begin{aligned}\min_{\Delta} \mathbb{E}\left\{(x(n) - \hat{x}(n))\eta(n)\right\}\} &\Rightarrow \min_{\Delta} \mathbb{E}\left\{\hat{x}(n)\eta(n)\right\}\} \\ &= \min_{\Delta} \mathbb{E}\left\{v(n) + 0.5v(n-2)\mathbf{w}^T(n)\mathbf{u}(n)\right\} \\ &= \min_{\Delta} \mathbb{E}\left\{(v(n) + 0.5v(n-2)) \sum_{i=0}^{M-1} \mathbf{w}^T(n)s(n-\Delta-i)\right\} \\ &= \min_{\Delta} \mathbb{E}\left\{(v(n) + 0.5v(n-2)) \sum_{i=0}^{M-1} \mathbf{w}^T(n)(x(n-\Delta-i) + \eta(n-\Delta-i))\right\}\end{aligned} \tag{2.17}$$

And due to uncorrelated of $x(n)$ and $v(n)$, equation above can be simplified to

$$\begin{aligned}&\min_{\Delta} \mathbb{E}\left\{(v(n) + 0.5v(n-2)) \sum_{i=0}^{M-1} \mathbf{w}^T(n)(x(n-\Delta-i) + \eta(n-\Delta-i))\right\} \\ =&\min_{\Delta} \mathbb{E}\left\{(v(n) + 0.5v(n-2)) \sum_{i=0}^{M-1} \mathbf{w}^T(n)\eta(n-\Delta-i)\right\} \\ =&\min_{\Delta} \mathbb{E}\left\{(v(n) + 0.5v(n-2)) \sum_{i=0}^{M-1} \mathbf{w}^T(n)(v(n-\Delta-i) + v(n-\Delta-2-i))\right\} \\ \approx&0 \rightarrow \Delta = 2\end{aligned} \tag{2.18}$$

Therefore, observing Eq.2.18, the error will tend to zero only if the delay Δ is larger than 2. Since the time indexes of signal $v(n)$ are non-overlapping, resulting in uncorrelated. Fig.2.6 depicts the effect of the delay $\Delta = 1 \sim 4$ on the estimated signal $\hat{x}(n)$ with the fixed filter length $M = 3$. The top row illustrated 100 realisations of $s(n)$, $\hat{x}(n)$, while the bottom row is the average signal of the estimation. When the delay $\Delta > 3$, the noise signal (in yellow) is suppressed a lot, leading to a small MSE. Thus, the results prove the previous analysis.

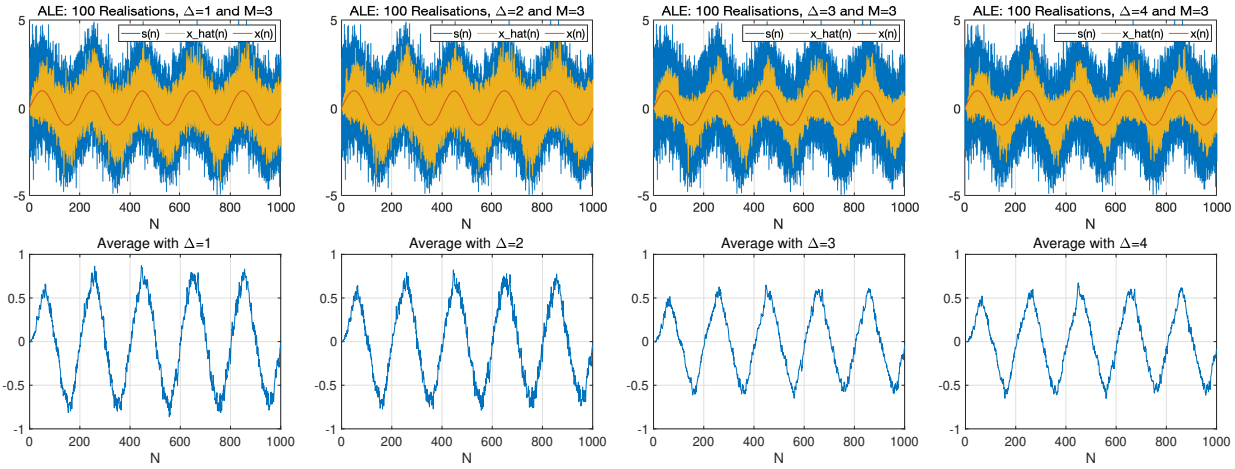


Figure 2.6: ALE: Effect of Δ with fixed $M = 3$

2.3.b Effects of M and delay on MSPE

In order to find the optimal delay and the filter length M, this experiment calculates the MPSE by varying Δ from 1 to 25 and M from 1 to 20. As shown in Fig.2.7, neglecting the inappropriate values of $\Delta \leq 2$ and $M = 1$, the MPSE curves both keep a increasing tendency with the growing of the delay and filter order. However, the MSPE remains approximately flat with a minimum error in range 3 to 6. When the filter order is large than 6, the over-modelling problem will occur, causing the growth of the MSPE. Notice that the performances of $\Delta = 3$ and $\Delta = 5$ are nearly same.

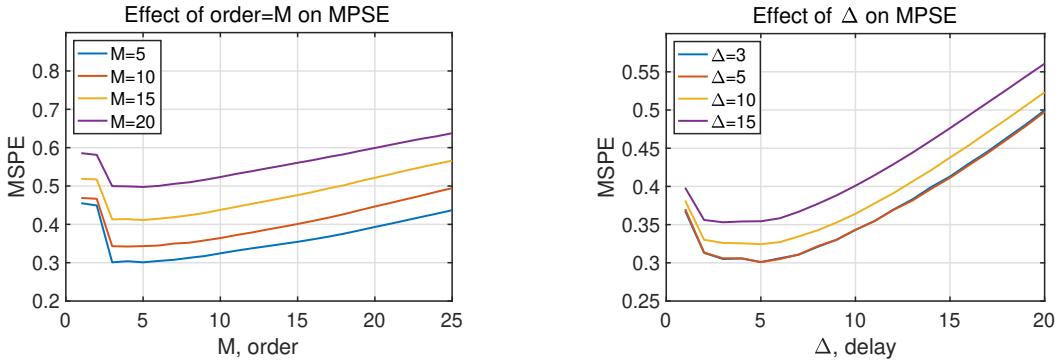


Figure 2.7: ALE: Effect of Δ and M on MSPE

Observing the effect on varying delay, the MSPE curves approximately keep same in the range of $\Delta \in [3, 6]$, which are similar as the ones of varying order M. Afterwards, the MSPE gradually rises up to the maximum when $\Delta = 25$. If the delay is set to large, there is a lag between the estimated signal $\hat{x}(n)$ and actual signal $x(n)$, resulting in the increasing MSPE. As shown in Fig.2.8, the realisations of optimal parameters ($M = 3$, $\Delta = 3$) and large delay ($M = 3$, $\Delta = 25$) are plotted. There is an obvious shift of estimated signal which proved the analysis. Due to the computational cost with increasing order, the relatively optimal parameters are $M = 3$ and $\Delta = 3$.

2.3.c ANC vs ALE

The adaptive noise cancellation (ANC) applied different input signal $\mathbf{u}(n)$ with correlated noise signal $\epsilon(n)$ whose aim is to estimate the noise $\eta(n)$. Thus, the desired signal $\hat{x}(n)$ is obtained by subtraction. The correlated secondary signal is assumed as $\epsilon(n) = 0.7\eta(n) + 0.01$. Fig.2.9 illustrated the performance of ANC and ALE. At the beginning of the estimation, the noise is quite large. With the increasing of time index, the

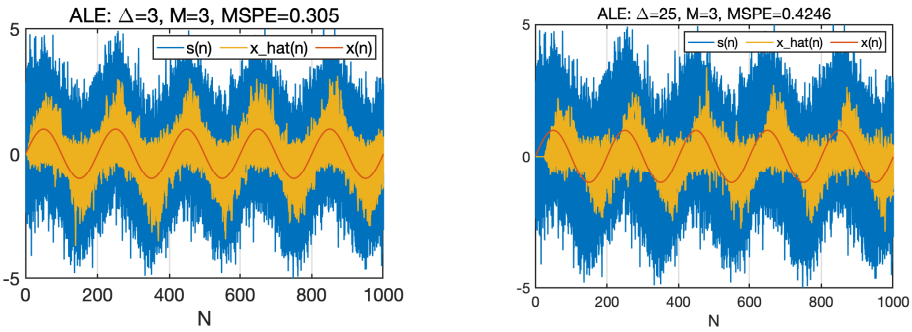


Figure 2.8: ALE: Realisations of increasing Δ with fixed M

noise is almost eliminated. Therefore, the MSPE of ANC is 0.1098 which is approximately one third of the ALE. Observing the average plot, the ANC estimation is equal to the actual sine wave after 200 samples. Thus, the ANC has a high performance than the ALE.

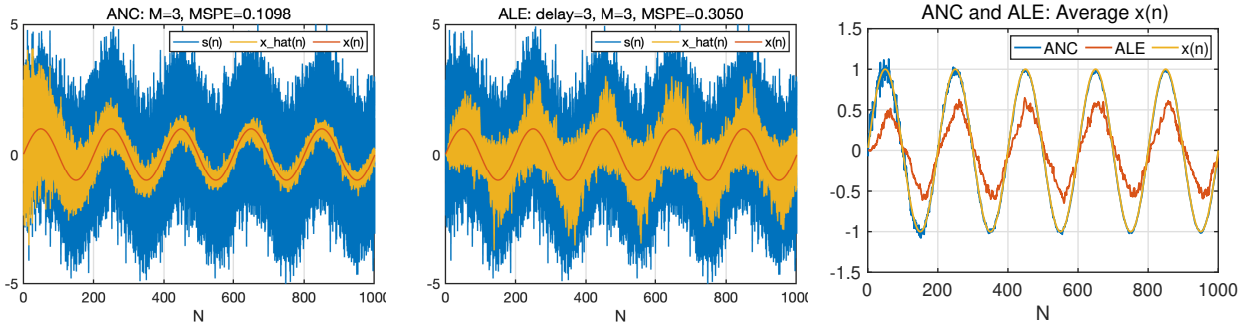


Figure 2.9: Performance of ANC and ALE

2.3.d ANC for EEG data

In order to remove the strong component at $50Hz$, a noisy sine wave $\epsilon(n)$ with corresponding frequency is synthesised. Fig.2.10 shows the spectrogram of the original `POz` data with the rectangular window length of 2^{12} and 0.5 overlapping. There is an distinct line in yellow at $50Hz$ which should be removed.

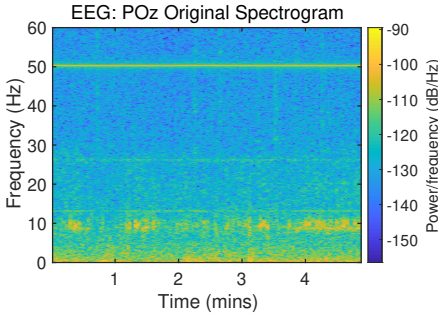


Figure 2.10: EEG: Original Spectrogram

Fig.2.11 shows the effects ANC by learning rate μ and M. When increasing the filter order M, the noisy component is getting to be removed more effectively. A under-modelling with small order causes residual of $50Hz$ component, while over-modelling results in excessively elimination. As to the μ, large learning rate will affect the component around $50Hz$, leading to the attenuation of interest signal.

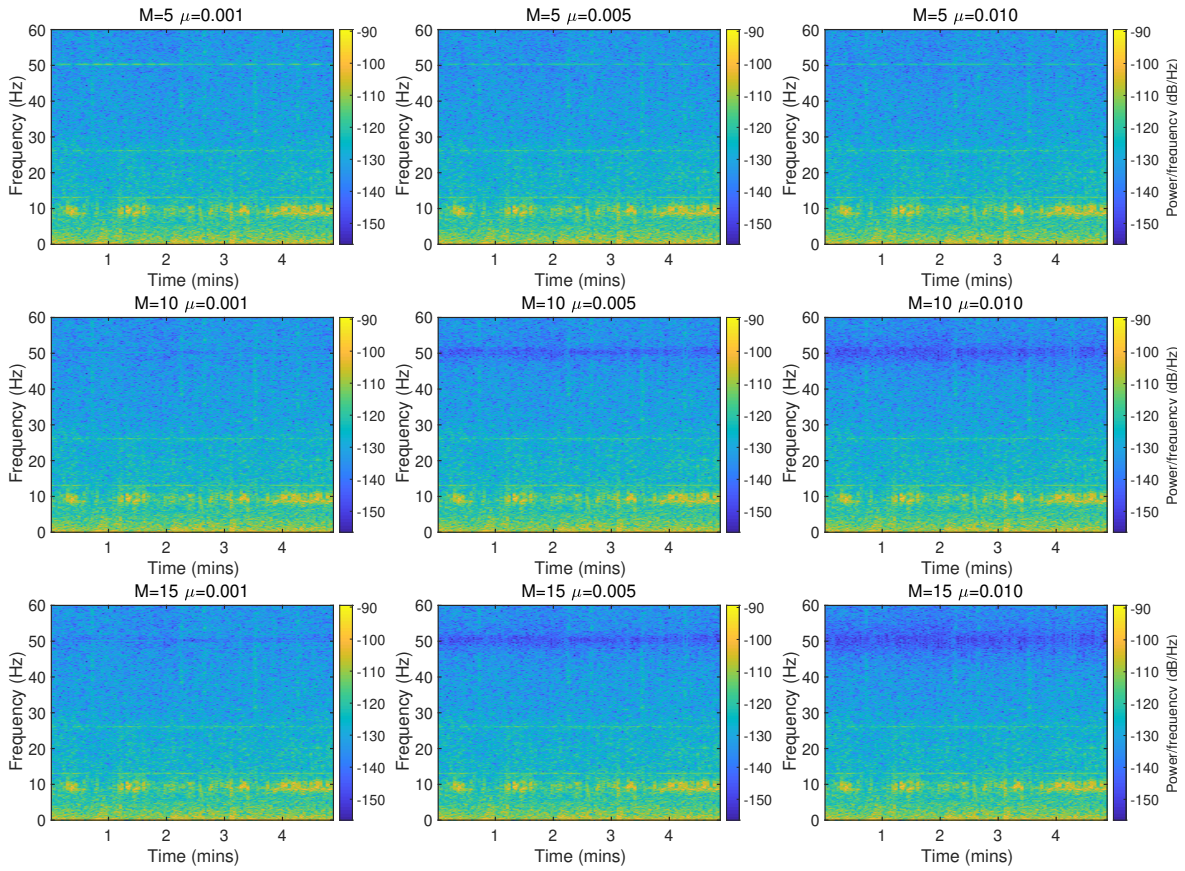


Figure 2.11: ANC POz Spectrogram: Effect of M and μ

Fig.2.12 shows the periodograms of original and ANC data at $\mu = 0.0001$ and $M = 10$. Only the component at $50Hz$ is suppressed and others are nearly same compared with original signal.

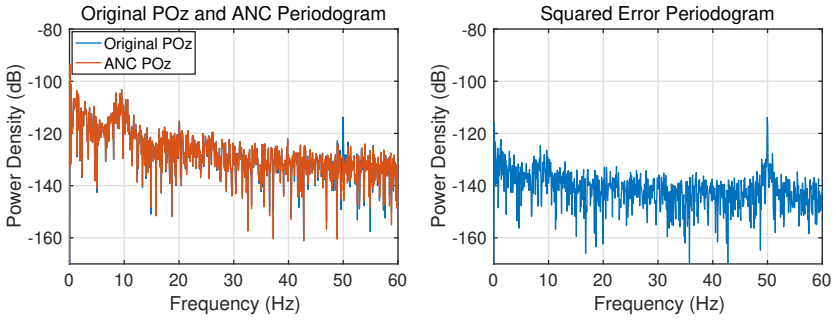


Figure 2.12: Periodograms and Squared error of Original EEG and de-noising EEG

Chapter 3

Widely Linear Filtering and Adaptive Spectrum Estimation

3.1 Complex LMS and Widely Linear Modelling

3.1.a CLMS vs ACLMS

The Wide Linear Moving Average (WLMA) process with first order is simulated. As shown in Fig.3.1, the white noise data and filtered data are plotted, which illustrates the circularity of the signal. The white noise is circularity and the filtered signal is not. As to the complex LMS (CLMS) algorithm, only standard strict linear model can be used to estimate the coefficients. Therefore, the learning curve of CLMS is a straight line without convergence. However, augmented CLMS (ACLMS) algorithm applies an additional weight $\mathbf{g}(n)$, which allows the ACLMS to capture the second-order statistical relationship [1]. Thus, the learning curve rapidly decreases and converges at $-300dB$.

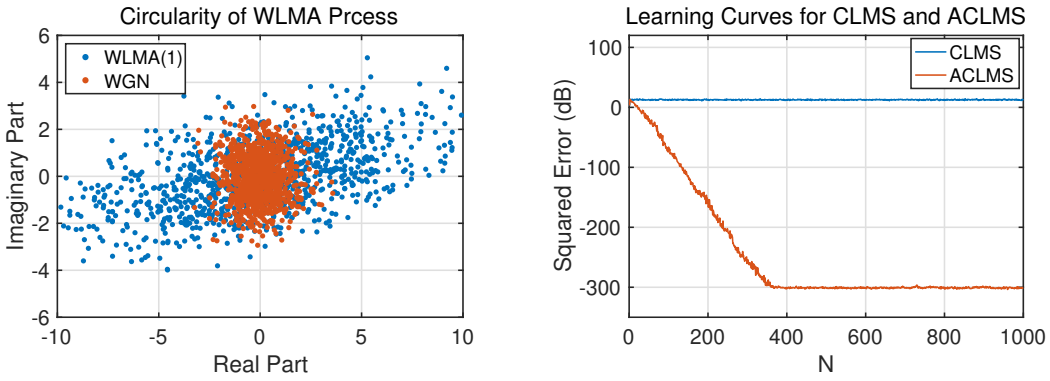


Figure 3.1: CLMS vs ACLMS: Circularity and learning curves

3.1.b Wind-speed data

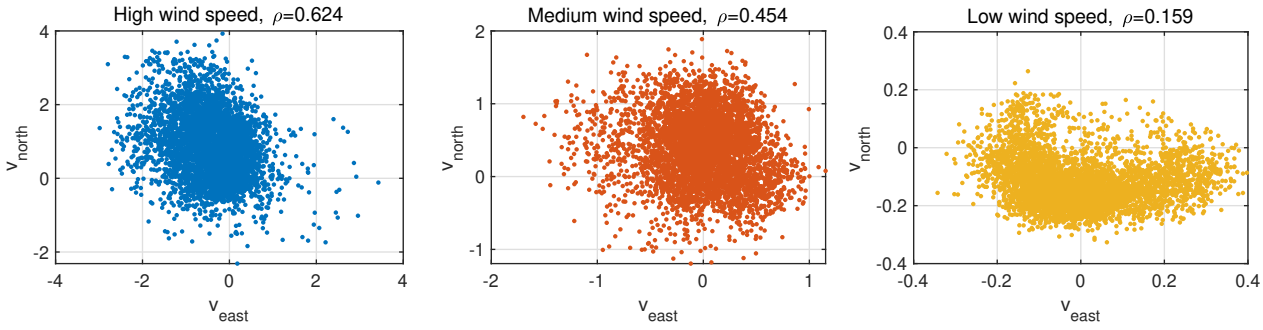


Figure 3.2: Circularity of Low, Medium and High speed wind data

The circularity describes the distribution of data. The circular complex random variables will not depend on the angle θ, whose the statistical relationship is remain invariant. As a consequence, the circular model will be

easily predicted. Fig.3.2 plots the scatter data for the three wind regimes with calculated circularity. The high wind regimes has largest circularity coefficient with $\rho = 0.624$, while the medium and the low wind data are $\rho = 0.454$ and $\rho = 0.159$ respectively. The low wind regime has relatively narrow range of distribution, which proved that the model has low degree of non-circularity with the smaller circularity coefficient. Applying

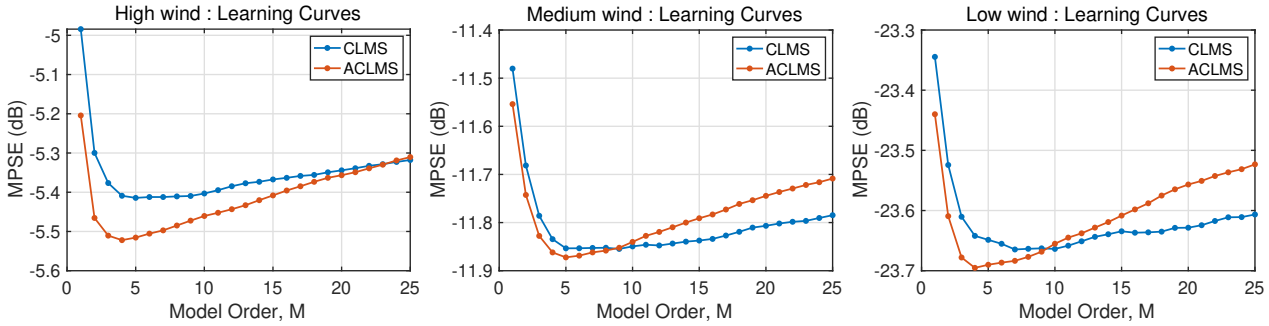


Figure 3.3: Learning curves of Low, Medium and High speed wind data

the CLMS and ACLMS algorithms to the three regimes data, the MSPE curves are illustrated by varying the model order $M \in [1, 25]$. It depicts that the wind data with small circularity coefficient has the small squared error. Moreover, the ACLMS algorithm perform has outstanding performance than the CLMS in the beginning range of model order. Afterwards, the minimum errors occur at $M = 4, 5$ and the model is getting to suffer over-fitting issues, resulting in growing error. In addition, the ACLMS has extra degrees of freedom, which leads to the advanced over-fitting than the CLMS. Therefore, the optimal model order for the wind regimes data are $M \in [3, 6]$.

3.1.c Balanced and Unbalanced System

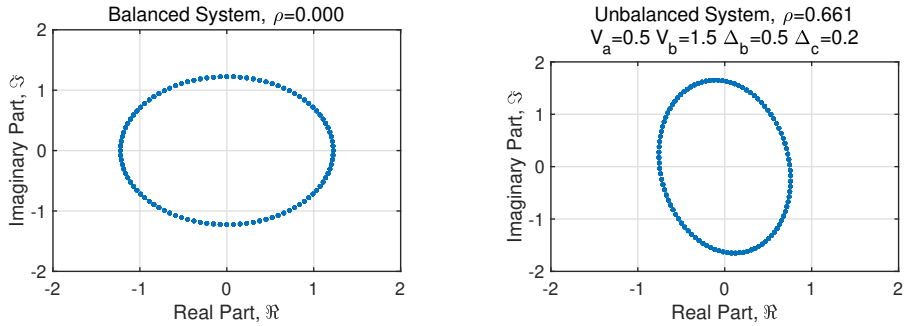


Figure 3.4: Examples of Balanced and Unbalanced System

According to the Clarke Transformation, the circularity example diagrams of balanced and unbalanced system are plotted as illustrated in Fig.3.4. The amplitude of three phase voltages are seleceted as $V_a = 0.5$, $V_b = 1.5$, $V_c = 1$ and the phase distortions are $\Delta_b = 0.5$ and $\Delta_c = 0.2$. Totally 2000 samples with sample frequency $f_s = 5000Hz$ are simulated. For the balanced system, the magnitude of phase voltage are equal $(V_a = V_b = V_c)$, meanwhile there is no phase distortion with $\Delta_b = \Delta_c = 0$. Therefore, changing the voltage magnitude and phase distortion causes a large circularity coefficient $\rho = 0.661$ shown in Fig.3.4(b). Fig.3.5 depicts the effect of magnitude and phase distortion on circularity.

3.1.d Derivation of nominal frequency

According to the zero-sequence voltage v_0 under balanced conditions given by

$$v(n) = \sqrt{\frac{3}{2}} V e^{j(2\pi \frac{f_o}{f_s} n + \phi)} \tag{3.1}$$

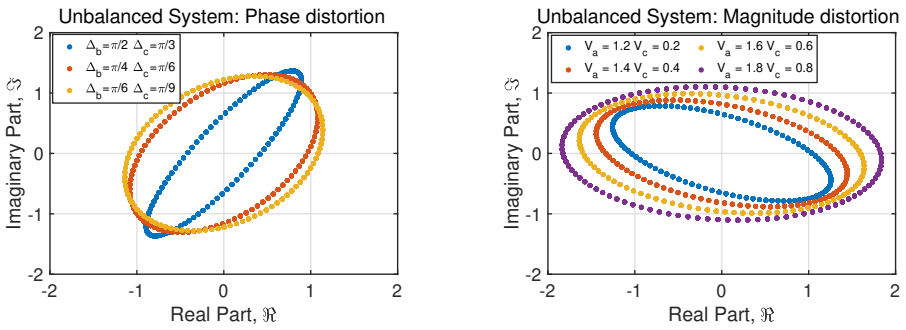


Figure 3.5: Unbalanced system with phase and magnitude distortion

Thus, the strict Linear model can be derived as

$$\begin{aligned} v(n+1) &= h^*(n)v(n) \\ \sqrt{\frac{3}{2}}Ve^{j(2\pi\frac{f_o}{f_s}(n+1)+\phi)} &= h^*(n)\sqrt{\frac{3}{2}}Ve^{j(2\pi\frac{f_o}{f_s}n+\phi)} \end{aligned} \tag{3.2}$$

Simplify both sides based on exponents rules,

$$\begin{aligned} e^{j2\pi\frac{f_o}{f_s}} &= h^*(n) \\ e^{j2\pi\frac{f_o}{f_s}} &= |h^*(n)|e^{j\angle\theta(h^*(n))} \end{aligned} \tag{3.3}$$

Observing Eq.3.3, the equation will only satisfy if and only if that the magnitude and the angle are equal on both sides. Due to the angle of complex number $h^*(n)$ defined as $\angle\theta(h^*(n)) = \tan^{-1}(\frac{\Im\{h^*(n)\}}{\Re\{h^*(n)\}})$, the equation below can be derived from Eq.3.3.

$$\begin{aligned} 2\pi\frac{f_o}{f_s} &= \tan^{-1}(\frac{\Im\{h^*(n)\}}{\Re\{h^*(n)\}}) \\ f_o(n) &= -\frac{f_s}{2\pi}\tan^{-1}(\frac{\Im\{h^(n)\}}{\Re\{h(n)\}}) \end{aligned} \tag{3.4}$$

However, the negative sign in the EQ.3.4 is used to guarantee the estimated nominal frequency f_o to be positive, since the imaginary part of weight $h(n)$ may be negative. As to the unbalanced system, the Clarke Transform is used instead of Eq.3.1

$$v(n) = A(n)e^{j(2\pi\frac{f_o}{f_s}n+\phi)} + B(n)e^{-j(2\pi\frac{f_o}{f_s}n+\phi)} \tag{3.5}$$

Hence, the widely linear model $v(n+1) = h^*(n)v(n) + g^*(n)v^*(n)$ can be expressed as

$$\begin{aligned} &A(n+1)e^{j(2\pi\frac{f_o}{f_s}(n+1)+\phi)} + B(n+1)e^{-j(2\pi\frac{f_o}{f_s}(n+1)+\phi)} \\ =&h^*(n)\left\{A(n)e^{j(2\pi\frac{f_o}{f_s}n+\phi)} + B(n)e^{-j(2\pi\frac{f_o}{f_s}n+\phi)}\right\} + g^*(n)\left\{A^*(n)e^{-j(2\pi\frac{f_o}{f_s}n+\phi)} + B^*(n)e^{j(2\pi\frac{f_o}{f_s}n+\phi)}\right\} \end{aligned} \tag{3.6}$$

Afterwards, the terms with identical exponent term are same, which are

$$A(n+1)e^{j(2\pi\frac{f_o}{f_s}(n+1)+\phi)} = [h^*(n)A(n) + g^*(n)B^*(n)]e^{j(2\pi\frac{f_o}{f_s}n+\phi)} \tag{3.7}$$

$$B(n+1)e^{-j(2\pi\frac{f_o}{f_s}(n+1)+\phi)} = [h^*(n)B(n) + g^*(n)A^*(n)]e^{-j(2\pi\frac{f_o}{f_s}n+\phi)} \tag{3.8}$$

For a time sequence unbalanced system, the term $A(n+1)$ and $B(n+1)$ can be assumed that $A(n+1) \approx A(n)$ and $B(n+1) \approx B(n)$ respectively. Thus, the Eq.3.7 and 3.8 can be simplified as

$$e^{j2\pi\frac{f_o}{f_s}} = \frac{h^*(n)A(n) + g^*(n)B^*(n)}{A(n+1)} \approx h^*(n) + g^*(n)\frac{B^*(n)}{A(n)} \tag{3.9}$$

$$e^{-j2\pi\frac{f_o}{f_s}} = \frac{h^*(n)B(n) + g^*(n)A^*(n)}{B(n+1)} \approx h^*(n) + g^*(n)\frac{A^*(n)}{B(n)} \tag{3.10}$$

In addition, the Eq.3.10 is the conjugate term of Eq.3.9. Thus,

$$\begin{aligned}\left\{h^*(n) + g^*(n)\frac{B^*(n)}{A(n)}\right\}^* &= h^*(n) + g^*(n)\frac{A^*(n)}{B(n)} \\ h(n) + g(n)\frac{B(n)}{A^*(n)} &= h^*(n) + g^*(n)\frac{A^*(n)}{B(n)}\end{aligned} \tag{3.11}$$

Let the term $\frac{B(n)}{A^*(n)} = X$ for simplicity. Multiply $\frac{B(n)}{A^*(n)}$ on both sides, a quadratic equation of X is formed.

$$\begin{aligned}h(n)\frac{B(n)}{A^*(n)} + g(n)\left|\frac{B(n)}{A^*(n)}\right|^2 &= h^*(n)\frac{B(n)}{A^*(n)} + g^*(n) \\ g(n)X^2 + [h(n) - h^*(n)]X + g^*(n) &= 0 \\ g(n)X^2 + 2\Im\{h(n)\}X + g^*(n) &= 0\end{aligned} \tag{3.12}$$

Therefore, the solution of term X is

$$\begin{aligned}\frac{B(n)}{A^*(n)} = X &= \frac{-2\Im\{h(n)\} \pm j\sqrt{4\Im^2\{h(n)\} - 4g^*(n)g(n)}}{2g(n)} \\ &= \frac{-\Im\{h(n)\} \pm j\sqrt{\Im^2\{h(n)\} - |g(n)|^2}}{g(n)}\end{aligned} \tag{3.13}$$

Substituting the solution into Eq.3.11 and 3.10 and keeping the corresponding sign, the equation below can be obtained.

$$\begin{aligned}e^{-j2\pi\frac{f_o}{f_s}} &\approx h^*(n) + g^*(n)\frac{A^*(n)}{B(n)} \\ &= h(n) + g(n)\frac{B(n)}{A^*(n)} \\ &= h(n) + g(n)\frac{-\Im\{h(n)\} - j\sqrt{\Im^2\{h(n)\} - |g(n)|^2}}{g(n)} \\ &= \Re\{h(n)\} - j\sqrt{\Im^2\{h(n)\} - |g(n)|^2}\end{aligned} \tag{3.14}$$

Thus, transform the complex number into exponential expression.

$$\begin{aligned}-2\pi\frac{f_o}{f_s} &= -\tan^{-1}\left\{\frac{\sqrt{\Im^2\{h(n)\} - |g(n)|^2}}{\Re\{h(n)\}}\right\} \\ f_o(n) &= \frac{f_s}{2\pi}\tan^{-1}\left\{\frac{\sqrt{\Im^2\{h(n)\} - |g(n)|^2}}{\Re\{h(n)\}}\right\}\end{aligned} \tag{3.15}$$

3.1.e ACLMS and CLMS on frequency estimation

As proofed on previous question, the nominal frequency can be estimated based on the ACLMS and CLMS algorithms. The theoretical nominal frequency is set to $50Hz$. Fig.3.6 illustrated the squared errors and estimation curves after applying these two algorithms. Both of algorithms can correctly estimate the $50Hz$ at stead-state without bias. The learning curve for the CLMS converges faster than the applying the ACLMS algorithm. Since there is an additional weight $\mathbf{g}(n)$ of the ACLMS needed to be considered, the frequency estimation has large bias at the beginning time index.

When dealing with the unbalanced system as shown in Fig.3.4(b), the CLMS algorithm can not capture the non-circularity data, resulting in non-update for learning. Thus, the frequency estimation has a bias and

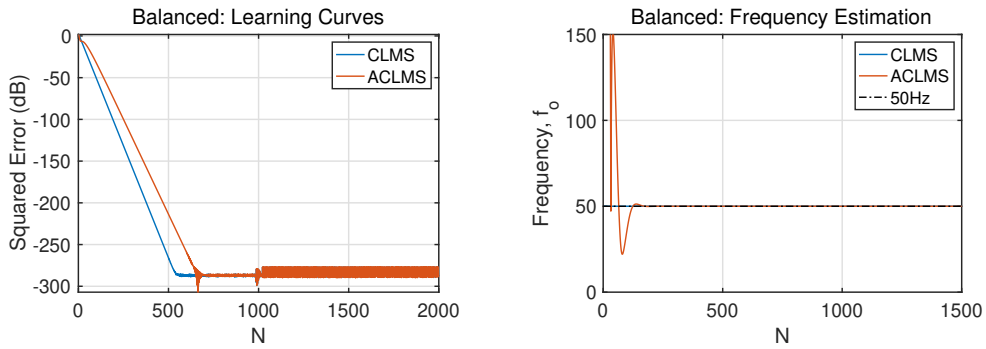


Figure 3.6: ACLMS vs CLMS: Learning curves and estimated frequency for balanced system

oscillation below $50Hz$. However, the convergence speed of the CLMS is much fast than the ACLMS. As to the ACLMS algorithm, the learning curve converges after 1800 samples with large ripples, which is caused by the distortion of the system. As to the frequency estimation, large oscillations occur in the beginning of 400 samples. Afterwards, the ACLMS converges to the true nominal frequency without bias and overshooting. In general, the ACLMS algorithm has sufficient performance that can replace the CLMS when dealing with both circularity and non-circularity data.

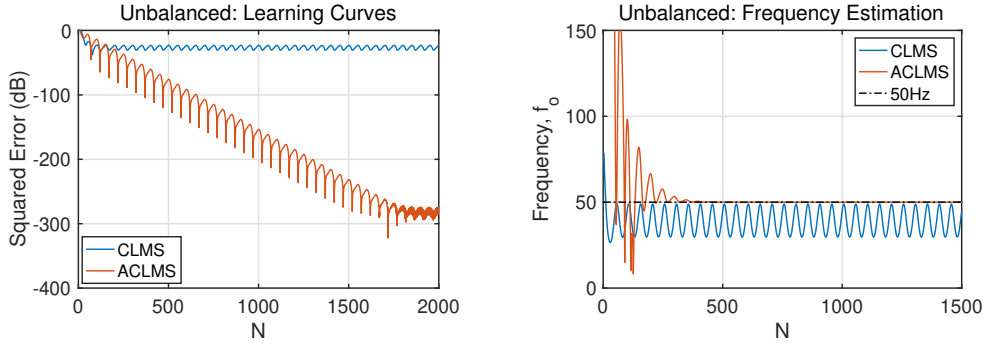


Figure 3.7: ACLMS vs CLMS: Learning curves and estimated frequency for unbalanced system

3.2 Adaptive AR Model Based Time-Frequency Estimation

3.2.a AR modelling of FM signal

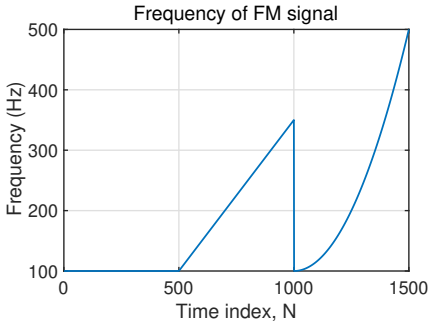


Figure 3.8: Time-variant frequency of FM signal

A FM signal with time-variant frequency as shown in Fig.3.8 is modulated with adding white noise in distribution of $\mathcal{N} \in (0,\ 0.05)$. The frequency is composed of three segments with constant, linear and quadratic parts, resulting in the non-stationary FM signal.

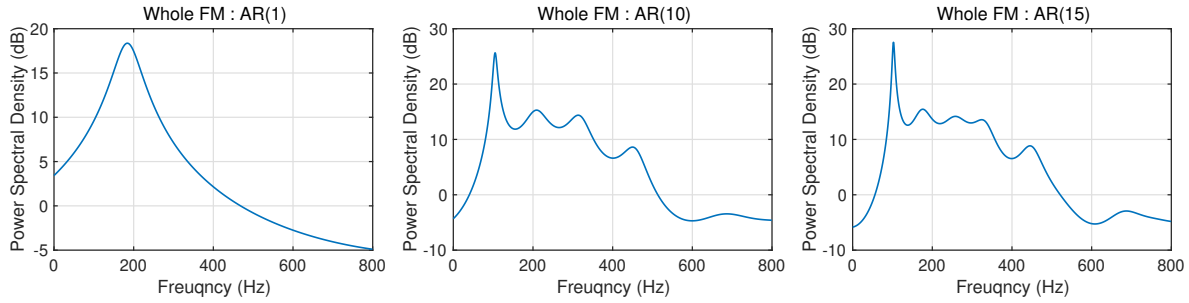


Figure 3.9: Whole FM AR estimation with order=1, 10 and 15

When using the MATLAB function `aryule` to estimate the coefficient for entire signal, Fig.3.9 depicts the performance of estimations with different orders. For AR(1) modelling, the estimated peak of frequency is inaccurate since the function is incapable to estimate non-stationary signal. With increasing the order of estimated model, only the constant frequency at $100Hz$ is successfully estimated. However, other segments frequencies are still not captured accurately. Thus, the previous method in Part 2.2 is not applicable for FM signal.

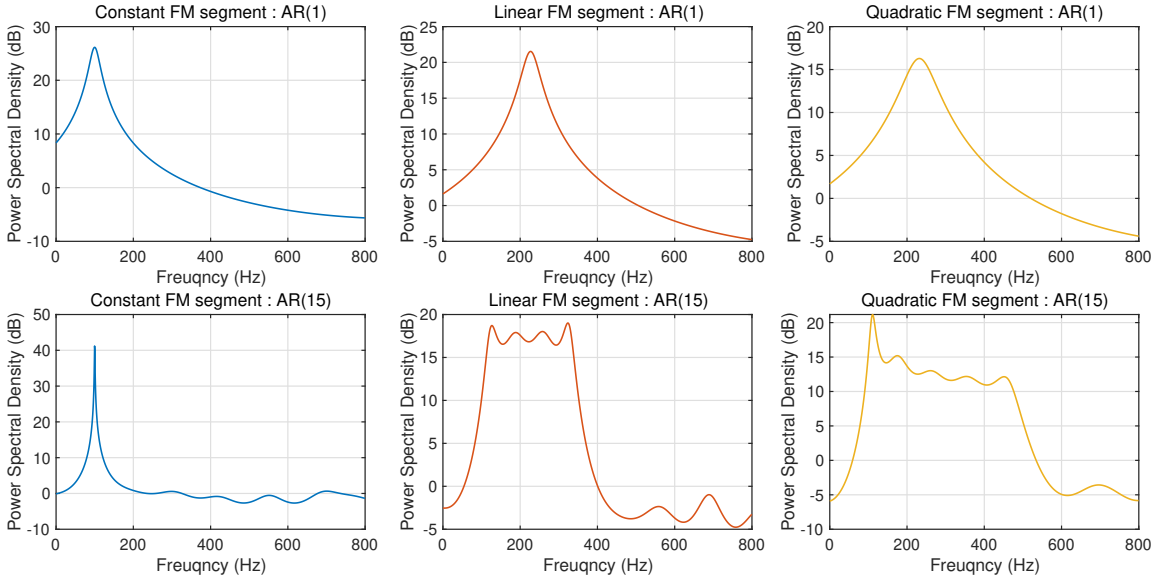


Figure 3.10: Block-based AR estimation with order=1 and 15

In addition, the frequency is time-varying function with constant, linear and quadratic segments. Thus, the block-based estimation is implemented with segments length $N = 500$ samples. Fig.3.10 illustrates the performance of estimated AR model with order 1 and 15. The constant frequency part can be successfully estimated with $p = 1$ presented in a peak value at $100Hz$. However, the linear and quadratic segments perform inferior with inaccurate frequencies. By increasing the capacity of estimated model, the frequency range of segment can be approximately estimated with acceptable uncertainty, as shown in second row of Fig.3.10. Nevertheless, the linear and quadratic relationship can not be observed based on the estimation since they are not satisfied with stationary.

3.2.b CLMS based estimated AR coefficient

In this section, the CLMS algorithm is applied to estimate non-stationary signal. Fig.3.11 shows the performance of estimating AR(1) coefficients with varying step-size μ. It is obviously that the CLMS can adaptively capture the time-variant frequencies. However, the different step-size μ also affects the performance of the CLMS estimation. With small step-size, the learning curve can not converge, leading to inadequate estimation. The performance at $\mu = 0.01$ is improved in spite of lacking beginning parts of the constant frequency. Setting $\mu = 0.05$ introduces an optimal estimation, while larger step causes oscillation in convergence. As a consequence, there are large variance and distortions of spectrum.

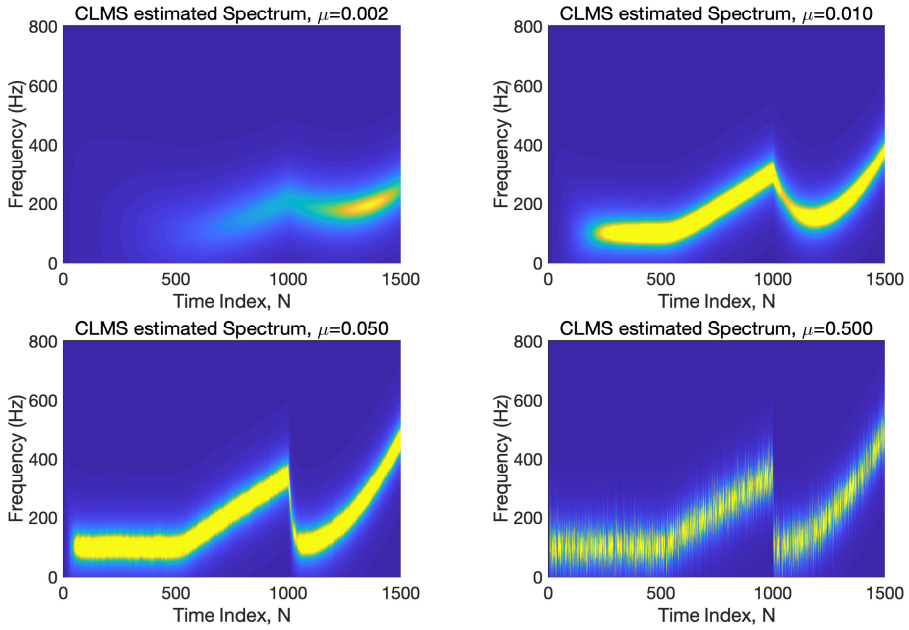


Figure 3.11: CLMS-based AR estimation with different step μ

3.3 A Real Time Spectrum Analyser Using LMS

3.3.a LS solution and relationship to DFT

Given the cost function $\mathcal{J}(\mathbf{w})$between estimated signal $\hat{\mathbf{y}}(n) = \mathbf{F}\mathbf{w}$ and true signal $\mathbf{y}(n)$,

$$\begin{aligned}\min_{\mathbf{w}} \|\mathbf{y} - \hat{\mathbf{y}}\|^2 &= \min_{\mathbf{w}} \|\mathbf{y} - \hat{\mathbf{y}}\|^H \|\mathbf{y} - \hat{\mathbf{y}}\| \\ &= \min_{\mathbf{w}} (\mathbf{y} - \mathbf{F}\mathbf{w})^H (\mathbf{y} - \mathbf{F}\mathbf{w}) \\ &= \min_{\mathbf{w}} (\mathbf{y}^H\mathbf{y} - \mathbf{y}^H\mathbf{F}\mathbf{w} - \mathbf{w}^H\mathbf{F}^H\mathbf{y} + \mathbf{w}^H\mathbf{F}^H\mathbf{F}\mathbf{w}) \end{aligned} \tag{3.16}$$

Thus, in order to minise the cost function, take the derivation with respect to $\mathbf{w}$ and equal to zero. The optimal weight can be obtained.

$$\begin{aligned}\frac{\partial \mathcal{J}}{\partial \mathbf{w}} &= -\mathbf{F}^H\mathbf{y} - \mathbf{F}^H\mathbf{y} + 2\mathbf{F}^H\mathbf{F}\mathbf{w} = 0 \\ \mathbf{w} &= (\mathbf{F}^H\mathbf{F})^{-1}\mathbf{F}^H\mathbf{y} \end{aligned} \tag{3.17}$$

The DFT of a sequence $x(n)$ is defined as

$$X_k = \sum_{n=0}^{N-1} \hat{x}_n e^{-j2\pi kn/N} = \sum_{n=0}^{N-1} \hat{x}_n W_N^{nk} \tag{3.18}$$

where $W_N = e^{-j2\pi/N}$. In vector expression of Eq.3.18, it is

$$\mathbf{X} = \mathbf{W}\hat{\mathbf{x}} \tag{3.19}$$

where transformation matrix $\mathbf{W}is$

$$\mathbf{W} = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & W_N & W_N^2 & \dots & W_N^{N-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & \dots & W_N^{(N-1)^2} \end{bmatrix} \tag{3.20}$$

Due to the transformation matrix $\mathbf{W}$ is orthogonal and symmetric, the DFT is unitray transform. Therefore, the IDFT is

$$\hat{\mathbf{x}} = \mathbf{W^{-1}X} \quad \text{with} \quad \mathbf{W^{-1}} = \frac{1}{N}\mathbf{W^H} \tag{3.21}$$

Which is corresponding to the estimated signal $\hat{\mathbf{y}} = \mathbf{Fw}$. Thus, the Eq.3.17 can be expressed as

$$\mathbf{w} = (\mathbf{F}^H\mathbf{F})^{-1}\mathbf{F}^H\mathbf{y} = (N\mathbf{F^{-1}F})^{-1}\mathbf{F}^H\mathbf{y} = N\mathbf{F}^H\mathbf{y} = \mathbf{F}^{-1}\mathbf{y} \tag{3.22}$$

where is corresponding to the DFT $\mathbf{X} = \mathbf{W}\hat{\mathbf{x}}$.

3.3.b Projection and baiss of DFT

Due to the transformation matrix of DFT is symmetric and orthogonal, the Fourier trainsfrom coefficients $\mathbf{w}$ is formed by projecting the signal $y(n)$ onto the transformation matrix $\mathbf{W}$ which is composed of harmonical sinusoids basises. In contrast, the IDFT is the inversely process to reconstruct the original signal by superposition of sinusoidal projections. However, the DFT and IDFT use finite length N to estimate either coefficients $\mathbf{w}$ or signal $\hat{\mathbf{y}}$, which makes errors occur comparing with true value.

3.3.c DFT-CLMS

The DFT-CLMS algorithm is implimented on the non-stationary FM signal. Fig.3.12 depicts the estimated time-varying frequencies. The trend of frequency is generally captured especially in perfectly estimation for constant segment. However, there exists an issue that the estimated frequency remains till the end of time index once it was obtained. Thus, the weights are not updated, which presents long smears in spectrum. The reason which causes this problem is that the gradient of back-propagation vanishes. Due to the gradient of the LMS algorithm is based on the eigenvalues of autocorrelation matrix. For the harmonically related sinusoids $\mathbf{x}(n)$, there are zero eigenvalues of $\mathbf{R}_{xx}$, resulting in hard back-propagation.

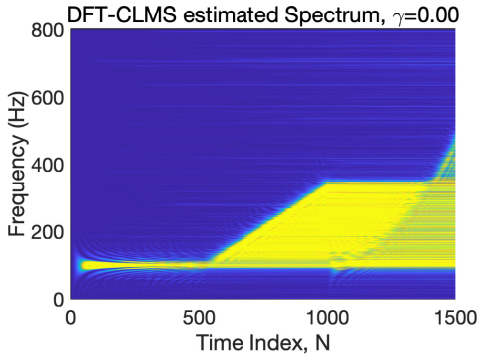


Figure 3.12: DFT-CLMS: estimated FM frequency

To solve this problem, the Leaky CLMS is applied on the weight update with leakage coefficient, in expression

of γ $\mathbf{w}(n+1) = (1-\gamma\mu)\mathbf{w}(n) + \mu e^*(n)\mathbf{x}(n)$. By changing previous value of weight, the weight will update along time. Fig.3.13 illustrated the performance affected by different γ. With small value of γ, most of smears are removed presented as relatively distinct trends. The optimal value of γ is 0.1 with acceptable bias. If the leakage coefficient is too large, the estimation is inaccurate due to the large bias added.

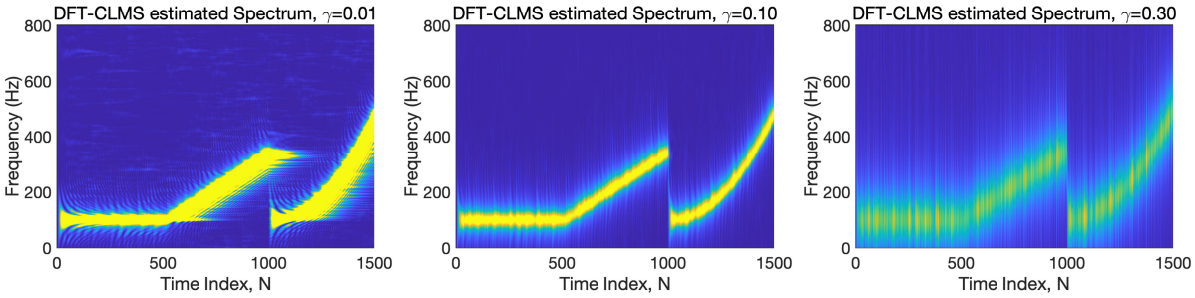


Figure 3.13: Leaky DFT-CLMS: estimated FM frequency with different γ

3.3.d DFT-CLMS estimated EEG signal

The DFT-CLMS algorithm can also be used to analyse EEG signal. As shown in Fig.3.14, the estimated spectrum agrees with the analysis in Part 1.2(b). A strong response at 8-10Hz called alpha-rhythm is clear shown. The SSVEP at 13Hz is detected as well, following its harmonic frequency at 26Hz. However, the harmonic frequency at 39Hz is hard to recognize. Moreover, the recording apparatus is strongly detected at 50Hz.
Nevertheless, the Leaky CLMS algorithm is not suitable for EEG data, since the EEG `POz` is stationary signal. Thus, adding a leakage coefficient results in adding bias on the correct estimations.

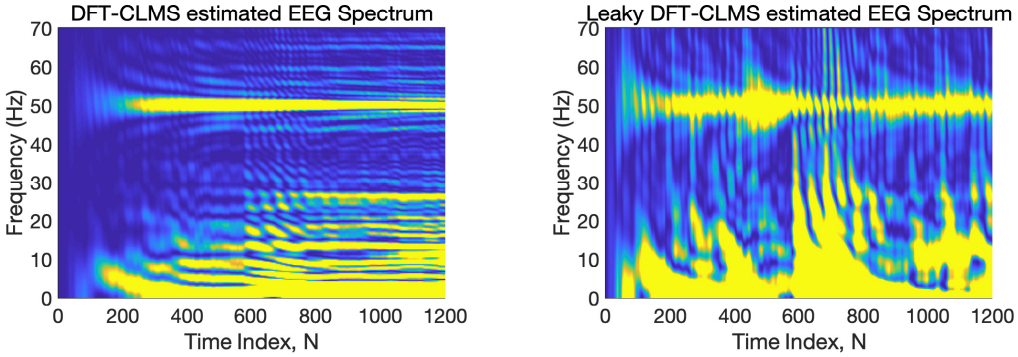


Figure 3.14: DFT-CLMS: estimated EEG frequency

Chapter 4

From LMS to Deep Learning

4.1 LMS of zero-mean time-series

The time-series signal is non-stationary data, which can be analysed by the LMS algorithm. The algorithm is used to predict one-step value based on the previous four values $y[n-4]$, $y[n-3]$, $y[n-2]$ and $y[n-1]$. As shown in Fig.4.1(a), the time-series is non-linear and zero mean. The performance of the basic LMS algorithm is illustrated in Fig.4.1(b). The predicted time-series is zero-mean as well. However, the predicted series do not capture perfectly of the original at the beginning part. After 400 time index, the predicted series converge with small difference with true series. In order to evaluate the performance appropriately, the metrics of mean squared error (MSE) and prediction gain (R_p) are measured. The MSE should be close to zero while the gain should be as large as possible. As to the LMS algorithm, the MSE is $16.032dB$ with $R_p = 5.196$, which performs inexpressively to some extent.

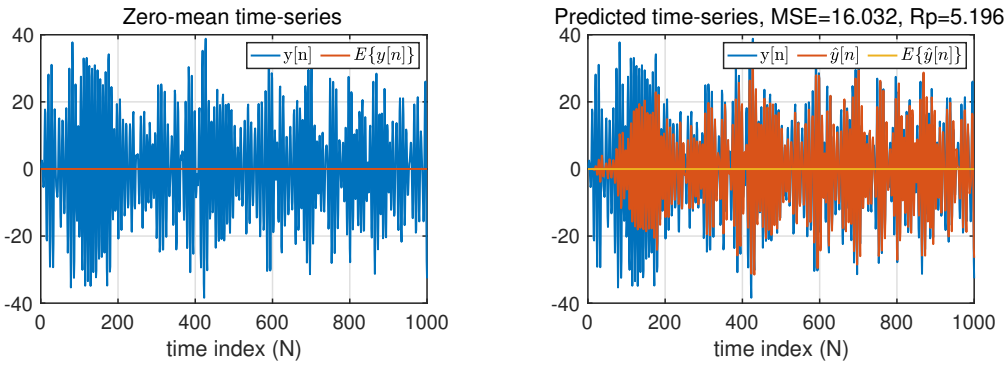


Figure 4.1: LMS: zero mean time-series one-step prediction

4.2 Activation function of predicted series

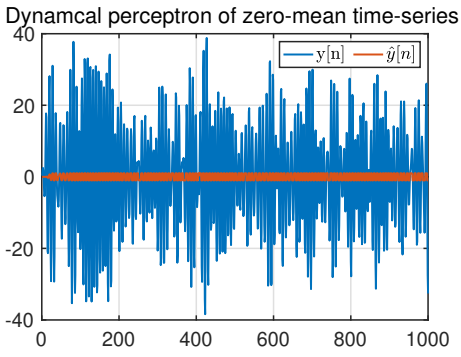


Figure 4.2: Dynamical perceptron: zero mean time-series one-step prediction

Due to the non-linearity of most of real-life data, the activation function `tanh` is applied to the each step of AR(4) process, which can be expressed as

$$\hat{y}[n] = tanh(\mathbf{w^T y}) \tag{4.1}$$

$$\text{where} \quad \mathbf{y} = [y[n-4], y[n-3], y[n-2], y[n-1]]$$

Fig.4.2 depicts the output using activation function against the original time-series. It is obvious that using `tanh` is inappropriate to predict the time-series data. The reason is that the range of `tanh` lies in $(-1, 1)$ whereas the zero-mean time-series is bounded in $(-40, 40)$. Therefore, in order to appropriately predict the series, the activation function need to be scaled.

4.3 Scaled activation function

As analysis previous, scaling the activation function expresses in Eq.4.1 by factor a. Fig.4.3 illustrates the performance of varying a with zero-mean data. For a small value of $a = 20$, the predicted $\hat{y}[n]$ is still lower than the range of the desired data. Thus, the MSE is extreme large than the standard LMS algorithm. However, the maximum range of prediction is restricted by activation function, leading to small variance of error. Thus, the prediction gain R_p is larger than the LMS. With incremental of a up to 80, the MSE is getting to decrease, corresponding to increasing predict gain. However, if the value of a is over 80, the prediction is overshooting to the true data, resulting in decreasing R_p and increasing MSE. In conclusion, the optimal range of a for predictinng zero mean data is $70 \sim 80$.

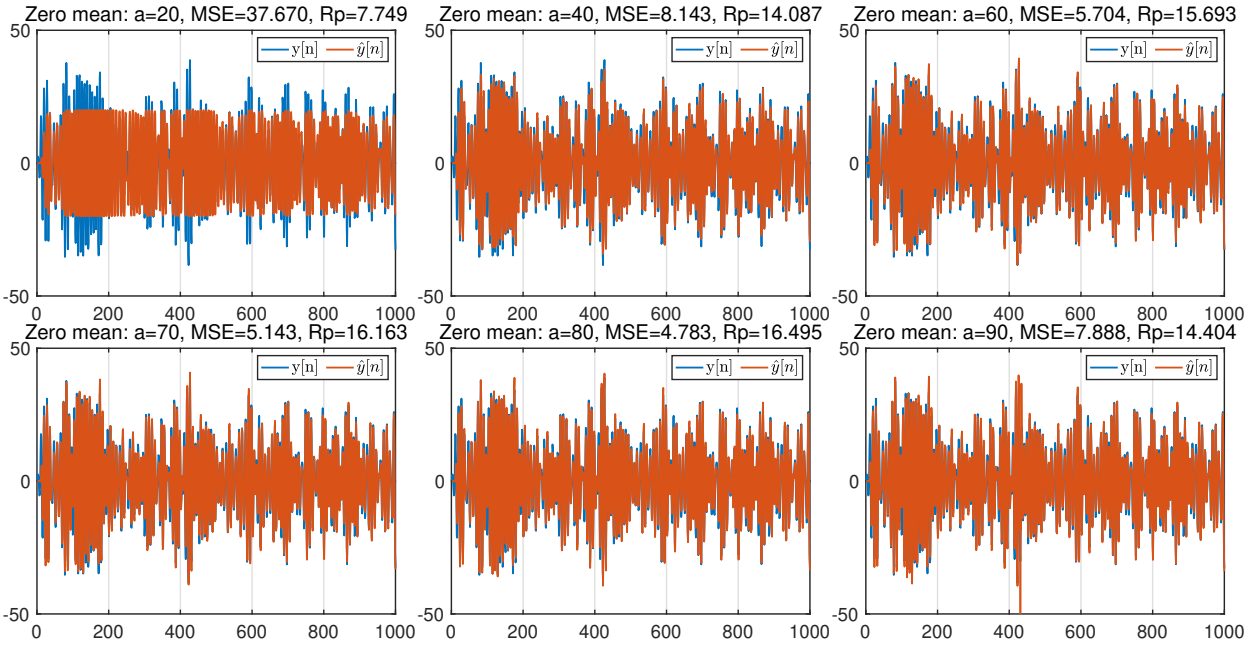


Figure 4.3: Scaled `tanh`: Prediction of zero mean data

In addition, it is harder to predict the none-zero mean data as shown in Fig.4.4, which presents in the larger MSE and smaller R_p. However, the optimal range of a for non-zero mean prediction is $40 \sim 50$.

4.4 None-linearity prediction with bias

Previous work is based on the zero mean time-series to make one-step ahead prediction. By adding a bias b for the activation function, expressed as $tanh(\mathbf{w}^T\mathbf{x} + b)$, the model can account for the mean automatically. Due to the small learning rate, the performances with bias are similar to the one without bias if only one epoch experiment is implemented. Thus, 100 number of epochs are used in order to continuously update weight. Fig.4.5a plots the MSE curves with or without bias for 100 epochs learning. Same performances are obtained at the beginning of training. Both of curves are rapidly plummet at first epochs and then slightly decrease up to convergence. Overall, the model with bias outperform the one without bias which converges

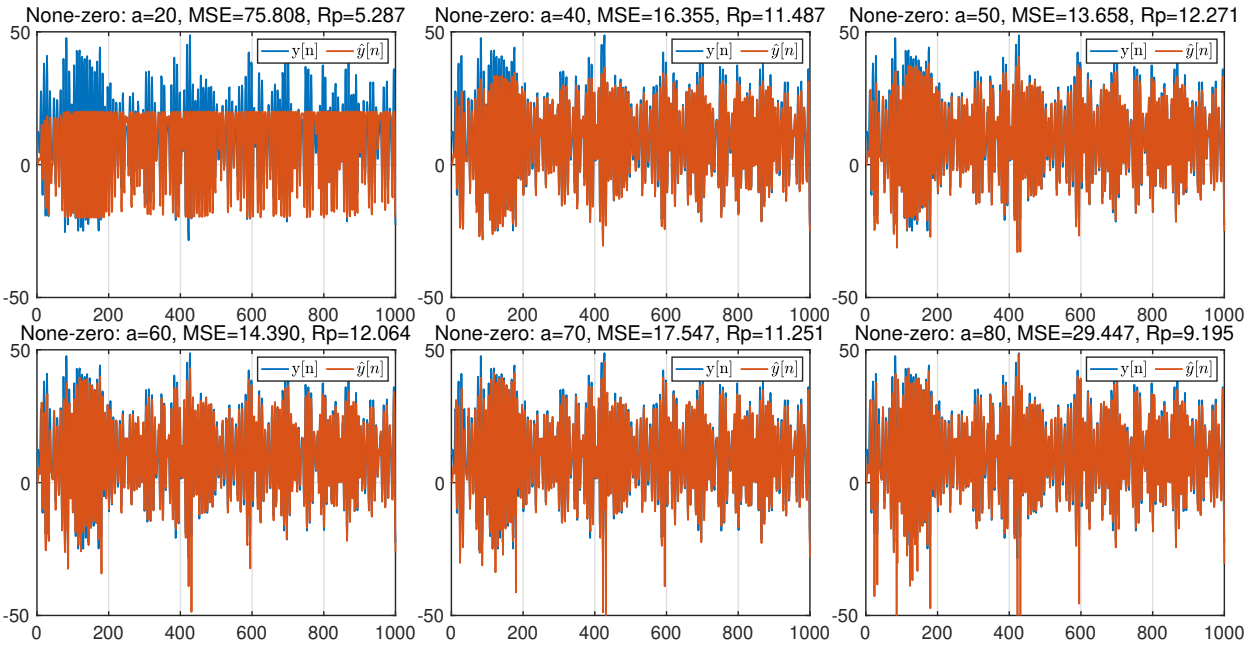
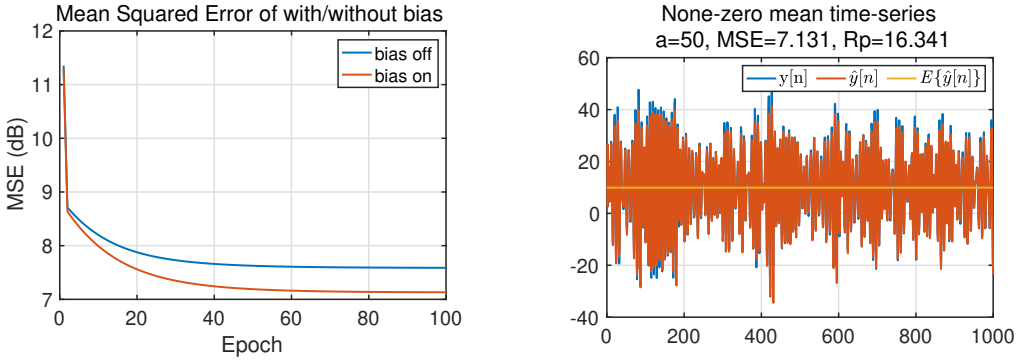


Figure 4.4: Scaled `tanh`: Prediction of none-zero mean data

after 60 epochs. Fig.4.5b shows the prediction of last epoch with bias against with the original non-zero mean series. Compared with the plots in Fig.4.4 with amplitude $a = 50$, the MSE reduces approximately in a half and prediction gain increases as well.



(a): Prediction MSE of 100 epochs

(b): The last epoch prediction

Figure 4.5: `tanh` with bias: None-zero mean time-series one-step prediction

4.5 Prediction with initialized weight

As to the LMS algorithm, the model cannot capture the original time-series at the beginning, which causes a long time to converge. Fig.4.6(a) illustrates the standard LMS prediction for non-zero mean which performs dissatisfactory with quite large error and insignificant prediction gain. The reason is that the initial weights are assumed to zero, which introduces difficulties to predict the non-zero mean data. Fig.4.6 depicts the performance of training initial weights. After training the first 20 data samples for 100 epochs, the initial weights are obtained which can speed up the learning process. With the pre-trained initial weights, the model perform better than training model after 100 epoch, with smaller MSE = 5.162 and slightly larger $R_p = 16.342$.

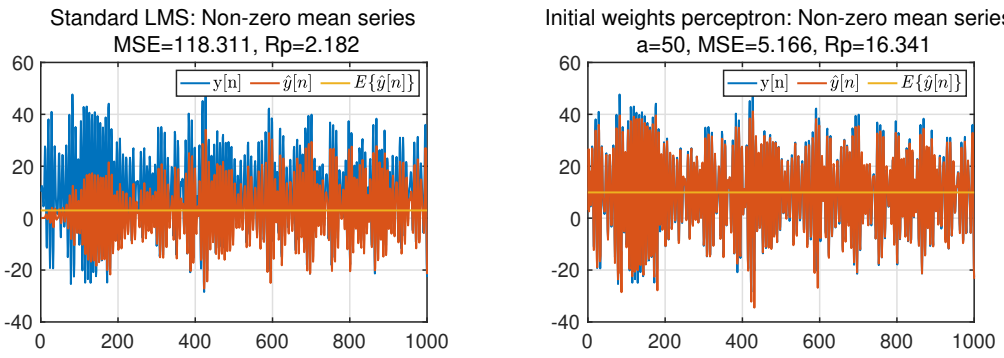


Figure 4.6: Standard LMS and Dynamical perceptron: zero mean time-series prediction

4.6 Back-propagation of Deep Network

For deep network, the neurons at each layers are fully connected with all inputs at last layer and outputs at next layer. And the weights can be expressed as:

$$\mathbf{w}^l \Rightarrow w_{ij}^{(l)} \begin{cases} 1 \leqslant l \leqslant L \text{ layers;} \\ 1 \leqslant i \leqslant d^{(l-1)} \text{ inputs;} \\ 1 \leqslant j \leqslant d^{(l)} \text{ outputs;} \end{cases} \tag{4.2}$$

d is neuron number at each layer

Thus, the weighted output on each neuron position is calculated by summing all weighted inputs with activation function.

$$z_j^l = \sigma(w_{ij}^l a_{ij}^{l-1} + w_{i0}^l) \tag{4.3}$$

$$\mathbf{z}^l = \sigma(\mathbf{w}^l \mathbf{a}^{l-1} + w_{i0}^l) \tag{4.4}$$

Therefore, set the weights randomly and the forward propagation process of the first iteration is finished. However, back-propagation of weights and bias need loss function, generally mean squared error, to calculated the gradient as shown below.

$$E = \mathbb{E}\{\|x[n] - \hat{x}[n]\|^2\} \tag{4.5}$$

Therefore, the error for the output δ_j^l can be expressed as

$$\begin{aligned} \delta_j^l &= \frac{\partial E}{\partial z_j^l} \\ &= \sum_i \frac{\partial z_i^{l+1}}{\partial z_j^l} \delta_i^{l+1} \\ &= \sum_i w_{ij}^{l+1} \delta_i^{l+1} \sigma'(z_j^l) \end{aligned} \tag{4.6}$$

In addition, the output error is corresponding to the previous weights. Thus, the gradient of weight is

$$\frac{\partial E}{\partial w_{ij}^l} = a_i^{l-1} \delta_j^l \tag{4.7}$$

Therefore, the weight can be updated by

$$w_{ij} = w_{ij} - \eta \frac{\partial E}{\partial w_{ij}^l} = w_{ij} - \eta a_i^{l-1} \delta_j^l \tag{4.8}$$

4.7 Deep Network

There are 10 sinusoidal waves with different frequency and amplitude as the linear inputs. The output $y[n]$ after applying activation function to $\mathbf{x}[n]$ is highly non-linear as shown in Fig.4.7. With the default parameters, three model, specifically in single neuron with linear and `tanh` function, deep network with `relu`, are used to train and test to evaluate their performance.

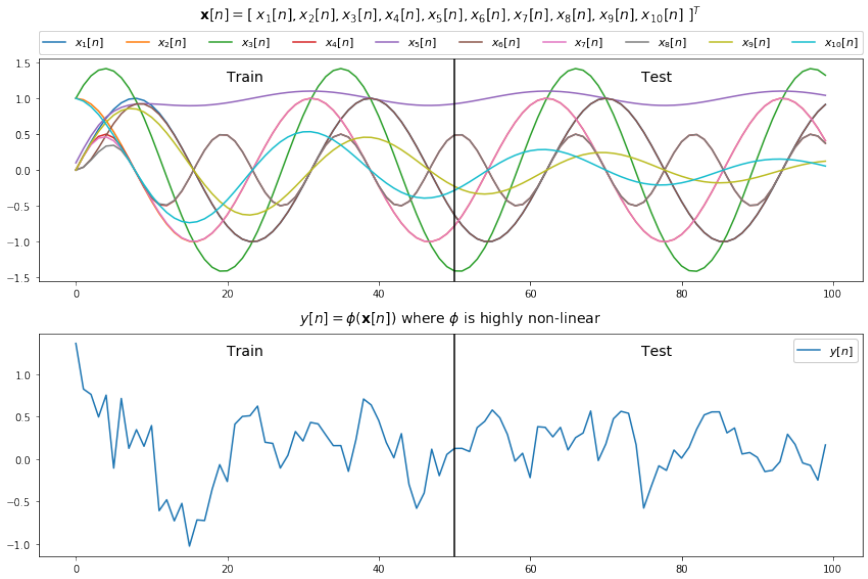


Figure 4.7: Harmonics sine waves and non-linear data with noise

Fig.4.8 shows the regression curves of these three models. The models use single neutron have similar performance, whose predictions are still linear. As to the deep network, this model can predict non-linear curve and the trend of the data is generally predicted. However, the performance is still insignificant which may be caused by the power of noise.

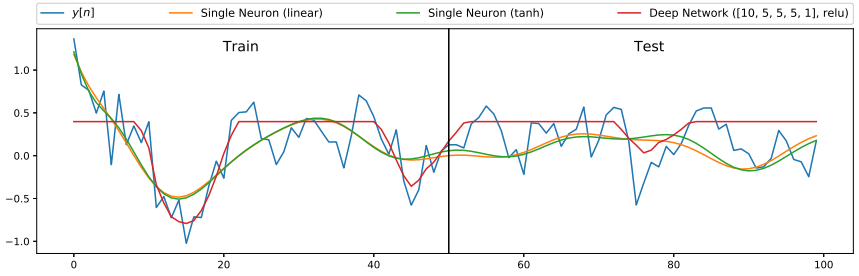


Figure 4.8: Performance of three models

Fig.4.9 shows the train error and test error for each model. The single neuron model learning curve rapidly decrease and then keep flat. While the test error of deep network starts at lower point and then converges to a slightly lower value. However, the single neuron models converge much more rapidly than the deep network, since their architectures are simple which are easily to be trained.

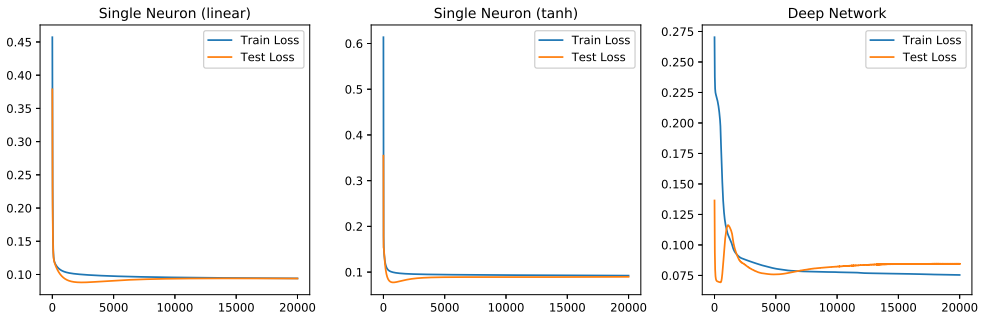


Figure 4.9: Learning curves of three models

4.8 Noise power and drawbacks of Deep Network

Experiments are repeated by changing different power of noise which are $\sigma^2 = 0$, 0.01, 0.2. Fig.4.10 shows the predictions and learning curves with $\sigma^2 = 0$. The model of deep network outperform during training and testing. The series is accurately captured in training and acceptable error in testing. However, the single neutron models can not predict non-linear series as well.

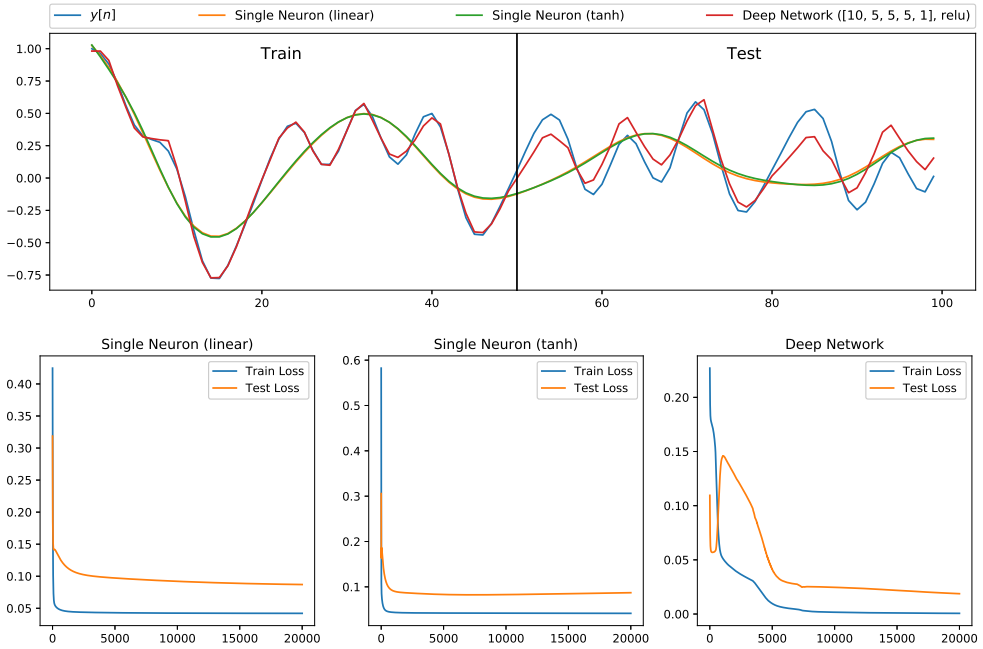


Figure 4.10: Performance of three models, $\sigma^2 = 0$

When adding a small power of noise $\sigma^2 = 0.01$, the performance of deep network model is getting worse. Parts of trends are inaccurately captured, resulting in a slightly increased error. Nevertheless, the deep network model still has expressive performance compared with single neuron models.
If the noise power is $\sigma^2 = 0.2$, the deep network model suffers the problem of over-fitting, which makes the model to predict the noise sufficiently in training. Thus, the testing error keeps increasing trend.

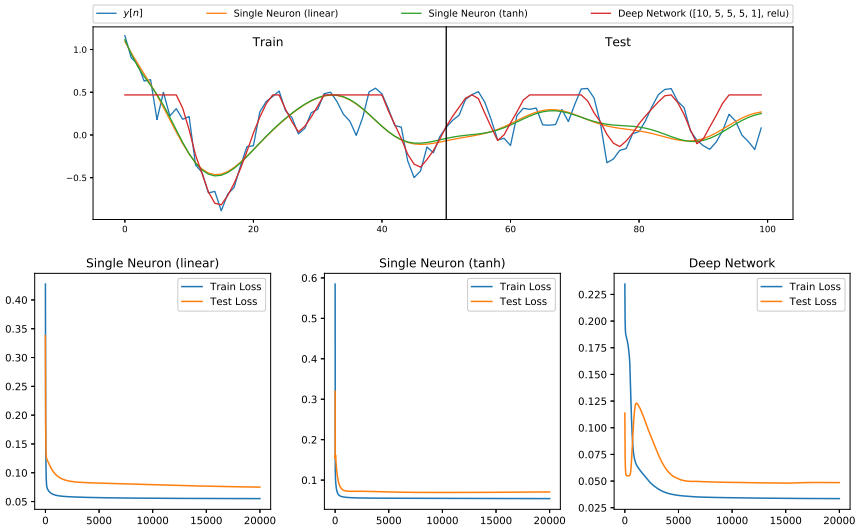


Figure 4.11: Performance of three models, $\sigma^2 = 0.01$

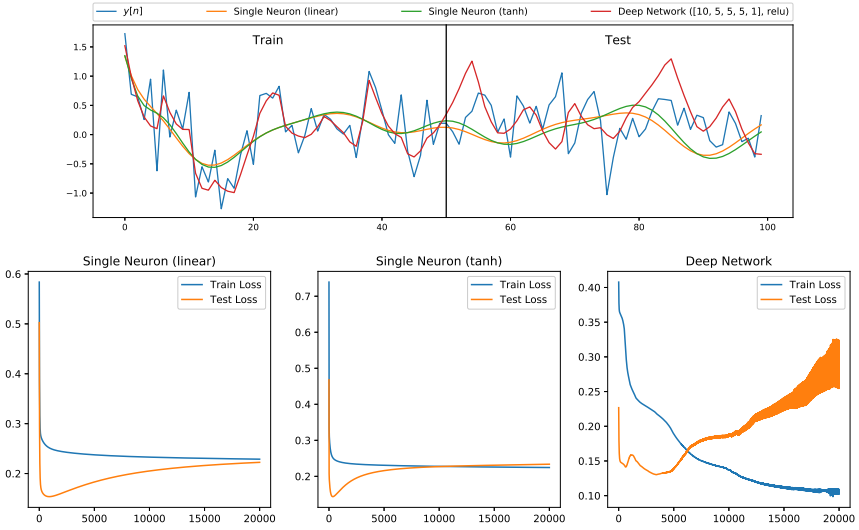


Figure 4.12: Performance of three models, $\sigma^2 = 0.2$

In summary, by varying the power of noise, the single neuron models perform much robust and stable with less computational cost as well, even if their performances need to be improved. As to the deep network model, it is easily affected by the noise power leading to a unstable performance. Moreover, due to fully-connected with all previous layer inputs and next layer outputs, the computational cost of deep network training is quite large.

References

[1] Danilo P Mandic. Spectrum Estimation and Adaptive Signal Processing Coursework. pages 1–23, 2019.

[2] Danilo P Mandic. Spectral Estimation and Adaptive Signal Processing, 2019.

[3] Danilo P Mandic and Vanessa Su Lee Goh. *Complex valued nonlinear adaptive filters: noncircularity, widely linear and neural models*, volume 59. John Wiley & Sons, 2009.