Imperial College London

Adaptive Signal Processing and Machine Intelligence Coursework

Yannan CHU – 02082872

April 12, 2022

Supervisor: Danilo Mandic

Department of Electrical and Electronic Engineering
Imperial College London, UK

Content:

1 Classical and Modern Spectrum Estimation

1.1 Properties of Power Spectral Density (PSD)

When covariance sequence decays rapidly, two definitions of PSD are equivalent, which can be proved as follows:

$$P(\omega) = \lim_{N\to\infty} E\left\{\frac{1}{N}\left|\sum_{n=0}^{N-1} x(n)e^{-jn\omega}\right|^2\right\} \tag{1.1}$$

$$= \lim_{N\to\infty} E\left\{\frac{1}{N}\left(\sum_{n=0}^{N-1} x(n)e^{-jn\omega}\right)\left(\sum_{m=0}^{N-1} x(m)e^{-jm\omega}\right)^*\right\}$$

$$= \lim_{N\to\infty} E\left\{\frac{1}{N}\left(\sum_{n=0}^{N-1} x(n)e^{-jn\omega}\right)\left(\sum_{m=0}^{N-1} x^*(m)e^{jm\omega}\right)\right\}$$

$$= \lim_{N\to\infty} \frac{1}{N}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} E\{x(n)x^*(m)\}e^{-j(n-m)\omega}$$

If $x(n)$ is a stationary signal, $r(n-m) = E\{x(n)x^*(m)\}$

$$P(\omega) = \lim_{N\to\infty} \frac{1}{N}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} r(n-m)e^{-j(n-m)\omega} \tag{1.2}$$

Set $k = n-m, n \in [0, N-1], m \in [0, N-1], k \in [-(N-1), N-1]$

$$P(\omega) = \lim_{N\to\infty} \frac{1}{N}\sum_{k=-(N-1)}^{N-1} (N-|k|)r(k)e^{-jk\omega} \tag{1.3}$$

$$= \lim_{N\to\infty} \sum_{k=-(N-1)}^{N-1} r(k)e^{-jk\omega} - \lim_{N\to\infty} \frac{1}{N}\sum_{k=-(N-1)}^{N-1} |k| \times r(k)e^{-jk\omega} \tag{1.4}$$

The second term of the above expression can be written as $Ae^{j\phi} = \lim_{N\to\infty} \frac{1}{N}\left(\sum_{k=-(N-1)}^{N-1}|k| \times r(k)e^{-jk\omega}\right)$ where A is the amplitude and ϕ is the phase. According to mild assumption, $A \le 0$.

$$A = \left|\lim_{N\to\infty} \frac{1}{N}\sum_{k=-(N-1)}^{N-1} |k| \times r(k)e^{-jk\omega}\right| \tag{1.5}$$

$$\le \left|\lim_{N\to\infty} \frac{1}{N}\sum_{k=-(N-1)}^{N-1} |k| \times |r(k)| \times \left|e^{-jk\omega}\right|\right|$$

$$= \left|\lim_{N\to\infty} \frac{1}{N}\sum_{k=-(N-1)}^{N-1} |k| \times |r(k)| \times 1\right| = 0$$

Since the amplitude should be non-negative ($A \ge 0$), the amplitude is zero ($A = 0$) regardless of phase.

$$Ae^{j\phi} = 0 \tag{1.6}$$

Only the first term of (1.4) is retained, and the second definition is converted to the first definition.

$$P(\omega) = \lim_{N\to\infty} \sum_{k=-(N-1)}^{N-1} r(k)e^{-jk\omega} = \sum_{k=-\infty}^{\infty} r(k)e^{-jk\omega} \tag{1.7}$$

The conditional equivalence between (1.1) and (1.7) is displayed in Figure 1. The auto-correlation function (ACF) of impulse signals only has one non-zero value and thus decays rapidly. Its PSD values computed by

two definitions are approximately the same. By contrast, the ACF of the sinusoidal signal decays slowly, and therefore its PSD values calculated by these two definitions are considerably different.

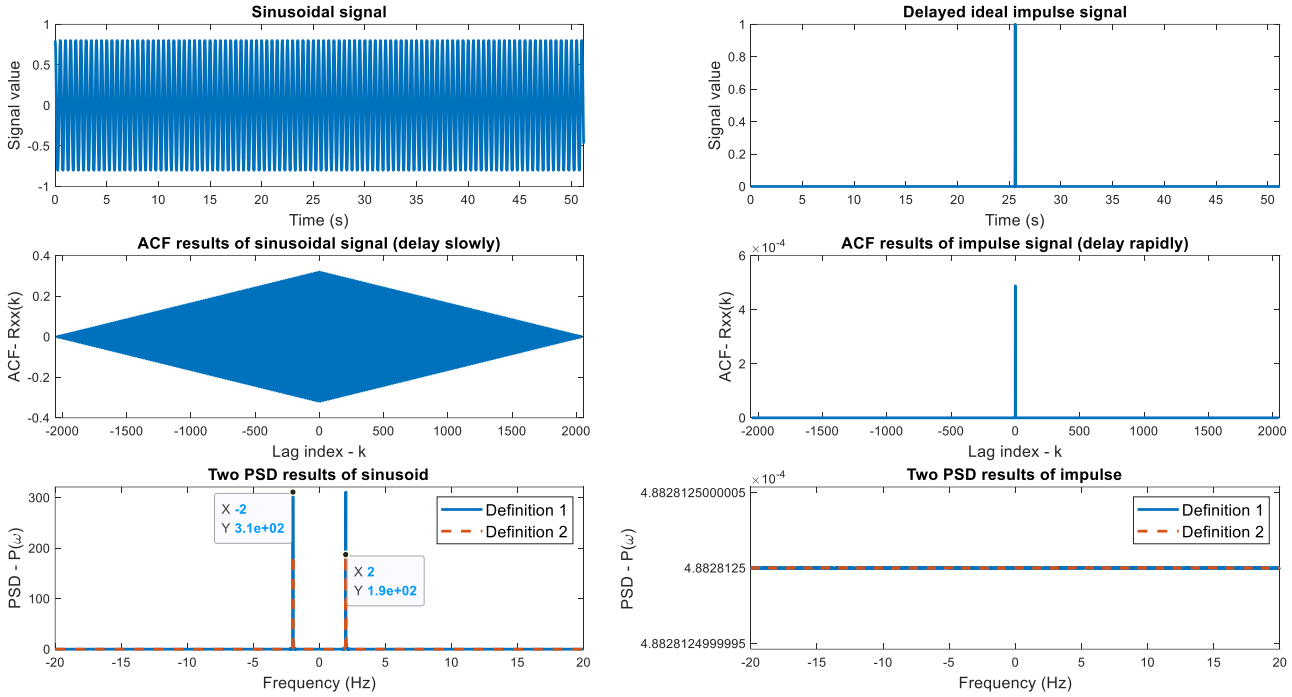

Figure 1. Two signals sampled at $40\ \mathrm{Hz}$ frequency and their ACF and PSD results (Definition 1 corresponds to (1.7) and Definition 2 corresponds to (1.1))

1.2 Periodogram-based Methods applied to Real-World Data

1.2.a The sunspot time series

The centering process removes the mean from the data, and the detrending process removes polynomial trend (i.e., linear trend in this task) from the data. For data that does not have an increasing or decreasing slope, the trend is stationary, and detrending can be approximated to mean removal. According to Figure 2, these two operations do not significantly change the data shape in time domain. When the hamming window is applied, the centered and detrended series and raw series provide approximately the same periodograms. However, since the DC component is removed, the peak in low-frequency range (i.e., $[0,\ 7\times10^{-3}]\ \mathrm{Hz}$) is lost. The pre-processing has a negligible impact on the periodograms in high-frequency range.

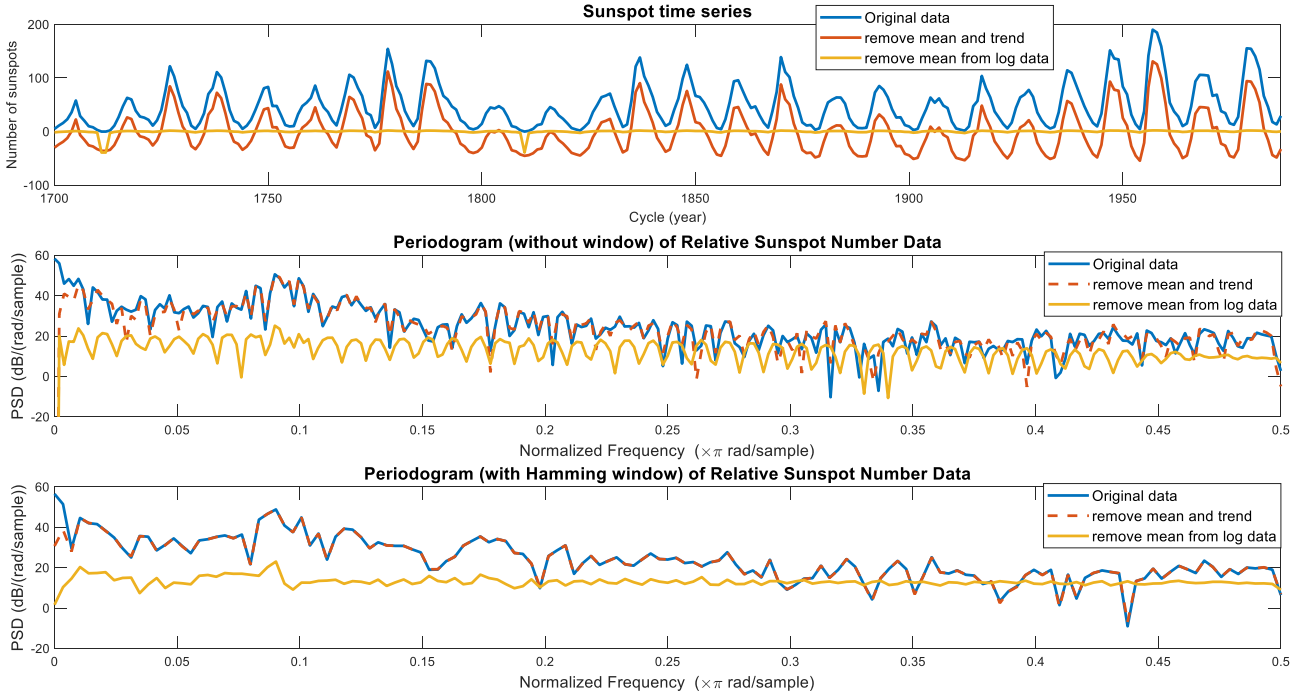

Figure 2. Original and processed sunspot time series as well as their periodogram results

The centered and log-transformed series produce a more stable and smoother periodogram especially when the hamming window is used. Generally, periodicities in original data only change slightly. Most local magnitude peaks at the same frequencies are observed. Some amplitude valleys and small undesired peaks near the high main peaks are removed, highlighting the dominating spectral peak. For example, the highest peak at the normalized frequency of $0.09\ \mathrm{Hz}$ can be easily identified because the height of the side peak at $0.1\ \mathrm{Hz}$ drops significantly after log-transform and mean removal.

1.2.b The electroencephalogram (EEG) experiment

In Figure 3, the averaged periodograms are generated by Bartlett's method without overlapping. Both standard and averaged periodograms with reasonably large window lengths can correctly demonstrate the spectral peaks of SSVEP. Since the frequency range of flashing stimulus is known ($[11\ \mathrm{Hz}, 20\ \mathrm{Hz}]$), it is easy to identify SSVEP peaks occurring at the fundamental frequency of $13\ \mathrm{Hz}$ and its harmonic frequencies (e.g., $26\ \mathrm{Hz}$ and $39\ \mathrm{Hz}$). The high peaks within $[8\ \mathrm{Hz}, 10\ \mathrm{Hz}]$ and at $50\ \mathrm{Hz}$ correspond to the alpha-rhythm and power-line interference instead of desired SSVEP peaks. However, harmonic peak amplitude drops as the frequency increases. When the harmonic frequency reaches $52\ \mathrm{Hz}$, the corresponding peak becomes hardly distinguishable from neighbouring undesired trivial peaks.

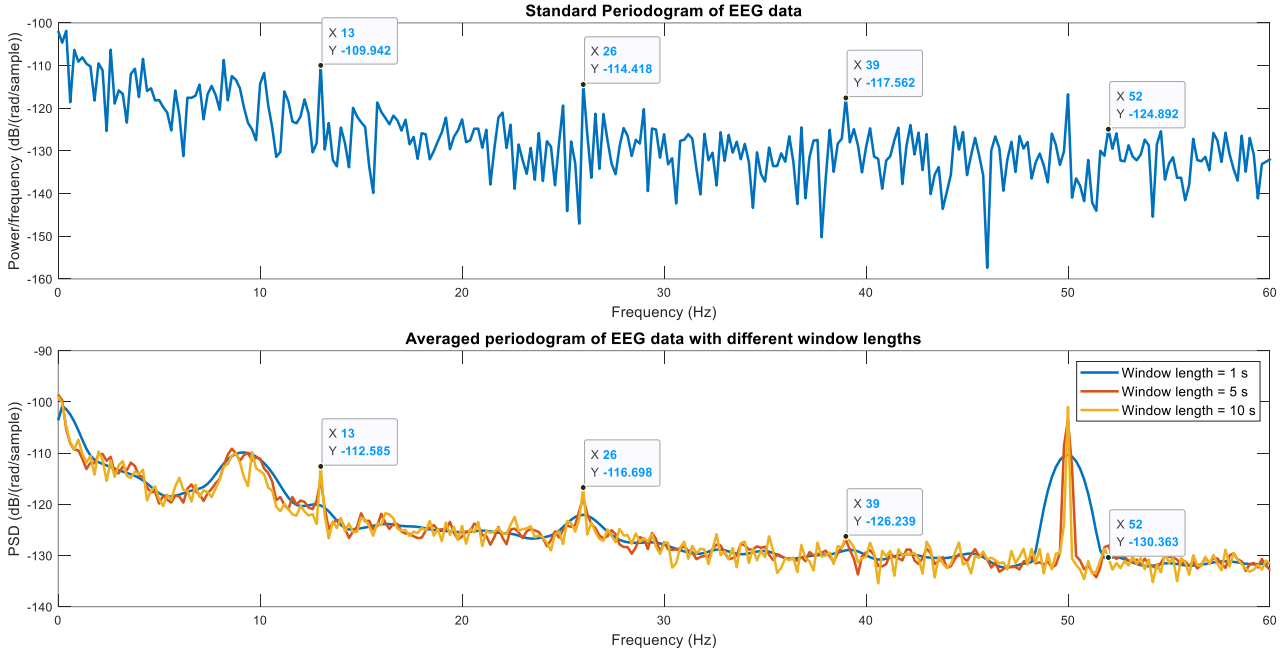

Figure 3. standard and averaged periodograms with three different window lengths (1s, 5s and 10) of the EEG recording signal (5 DFT samples per Hz)

Compared with the standard periodogram, averaged periodogram with window length 10s effectively improves the resolution at the cost of variance reduction. Generally, it has smoother spectral response, highlights desired peaks and attenuates their surrounding undesired interference. However, as the window shrinks, the frequency resolution drops, and the plotted curve becomes smoother. When the window length is 1s, it becomes difficult to accurately identify these broad peaks. Two peaks at fundamental and first harmonic frequency are still observable, while other SSVEP peaks cannot be identified. Therefore, 10s window length achieves a good balance between frequency resolution and variance.

1.3 Correlation Estimation

1.3.a Unbiased correlation estimation and preservation of non-negative spectra

For all three signals, when the lag index k is small (i.e., $0 \leq k \leq 200$), the biased and unbiased ACF estimates have high similarity. However, as k increases and gets close to N, the amplitudes of biased ACF estimates gradually reduce to small values (i.e., approximately zero) because fewer products of samples are summed and the coefficient $1/N$ is constant. By contrast, the amplitudes of unbiased ACF estimates become highly erratic and rise to large values due to its growing coefficient $1/(N-k)$.

According to Figure 4, the correlogram corresponding to biased ACF is non-negative and has a relatively small variance. However, the correlogram corresponding to unbiased ACF exhibits large variance and contains negative PSD values since the erratic unbiased ACF estimates are not positive-definite and reliable when few samples are available for PSD estimation (i.e., k becomes large). Therefore, the biased ACF is a better choice than the unbiased ACF.

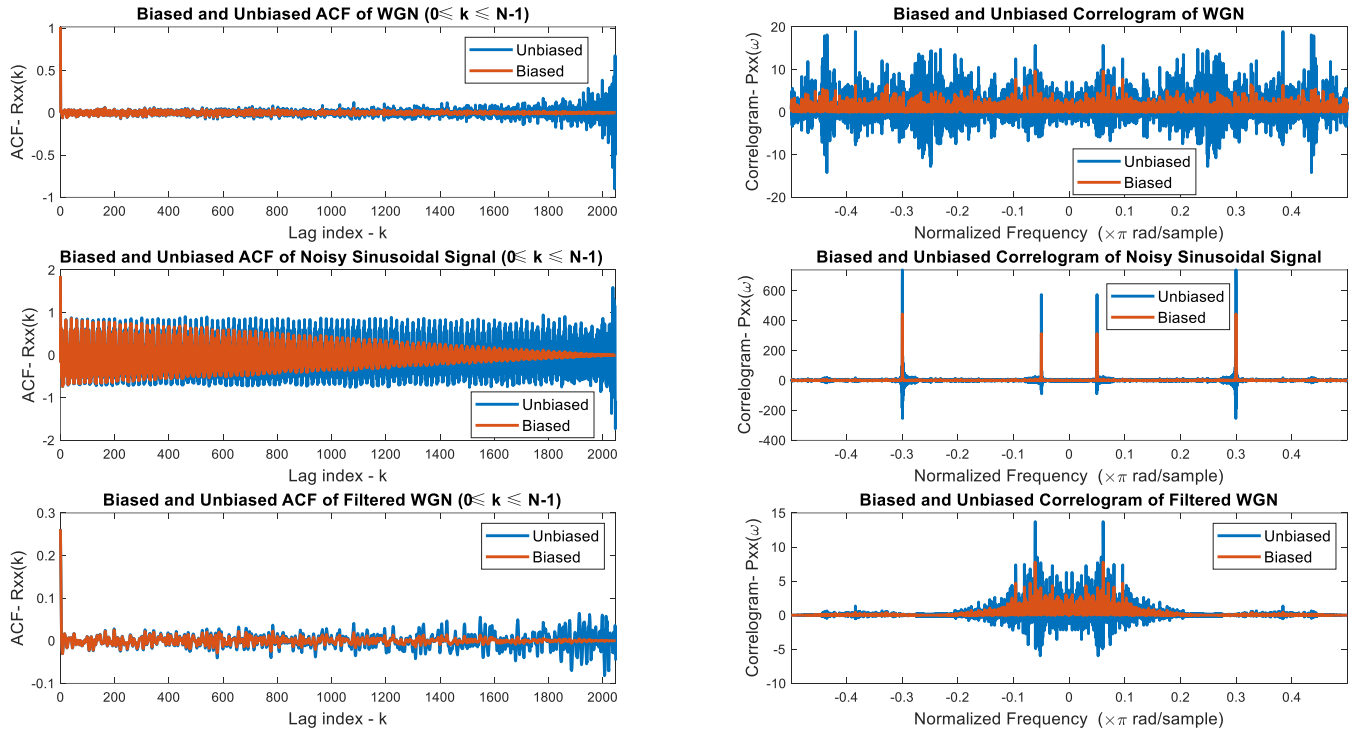

Figure 4. The biased and unbiased ACF estimates and corresponding correlograms of three signals (i.e., White Gaussian Noise, Noisy sinusoidal signals and WGN filtered by moving average filter). (Since the PSD should be a real function, only the real part of the Fourier transform of ACF estimates is retained, and log-transform is not applied to PSD results due to possible negative PSD values)

1.3.b Individual PSD estimates and their mean and standard deviation

500 signals composed of sinusoids corrupted by white Gaussian noise (WGN) were created, and their PSD estimates are plotted in Figure 5. Each individual signal has two main sinusoidal components with different amplitudes and frequencies.

$$x[n] = 0.4\sin(2\pi \times 12 \times n) + 0.7\sin(2\pi \times 24 \times n) + w[n], \quad w[n] \sim \mathcal{N}(0,1) \tag{1.8}$$

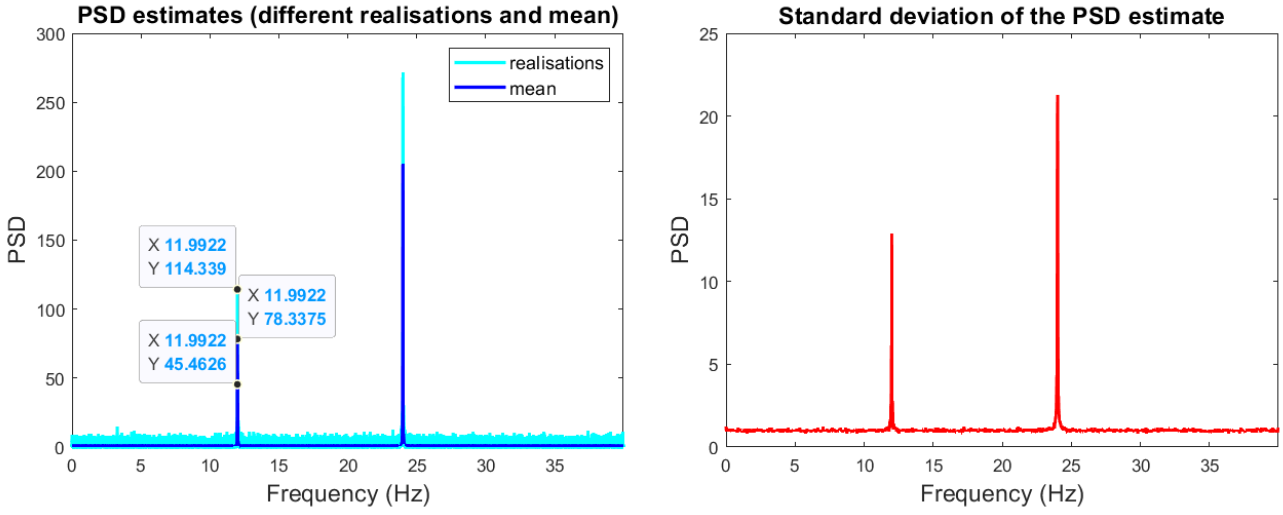

Figure 5. The 500 realisations of PSD estimate of a random process as well as their mean and standard deviation estimates

Due to the WGN, the signal power is spread over the whole frequency range. Despite some fluctuation, the mean noise power at each frequency is approximately 1 which matches the unity variance of noise. The two

sharp spectral peaks corresponding to two useful signal components have much higher amplitudes than noise components and can be easily identified. However, individual realisations sometimes result in considerably higher or lower spectral peaks than mean PSD peaks. This amplitude fluctuation is also caused by WGN. According to the bottom diagram of Figure 5, standard deviations of PSD estimates are proportional to the mean PSD estimates. Thus, large PSD values have wider confidence intervals, and the largest variance occurs at the frequency corresponding to the highest peak. At the undesired noise frequencies, different realisations tend to have similar PSD estimates. At the desired signal frequency, different realisations are dispersed over a wide range.

Since the PSD estimator is asymptotically unbiased, the estimated value ($\hat{P}_{per}(\omega_m)$) would approach the actual PSD value ($P_{xx}(\omega_m)$) when the number of samples (N) becomes infinity. Besides, the mainlobe width and variance can be effectively reduced by increasing N.

1.3.c Plotting the PSD in dB

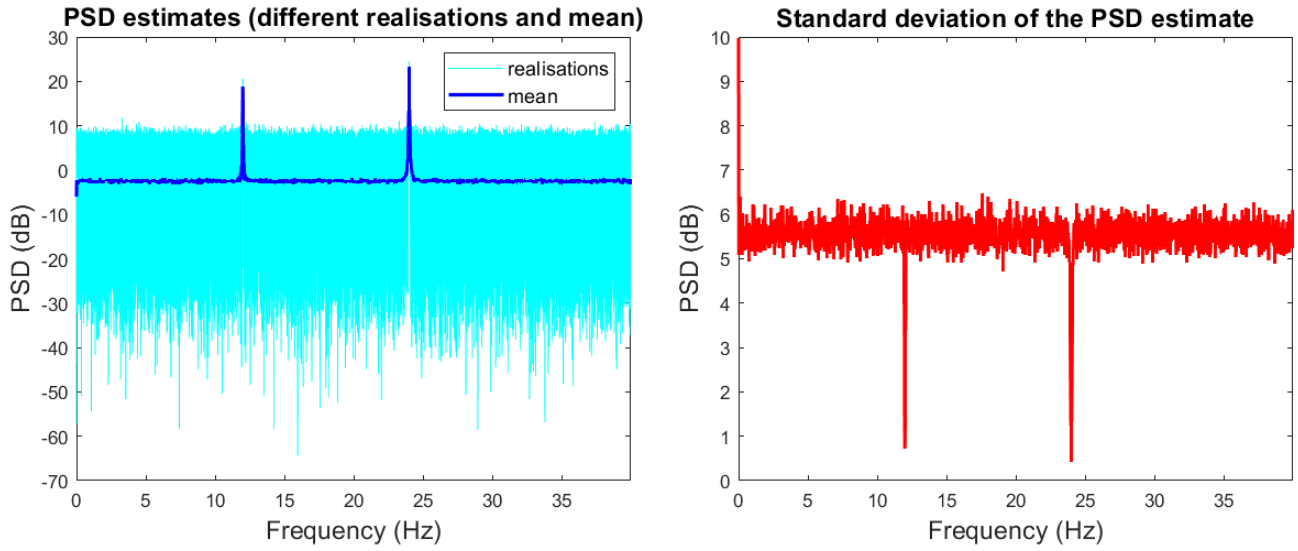

Figure 6. The PSD estimates in Figure 5 are displayed in the unit of dB

Generally, the log-transform highlights the small variance while ignoring the large variance. These small fluctuations of PSD values lower than 10 dB are visually amplified, but the fluctuations around two peaks become approximately invisible. Consequently, the large PSD value results in a small variance, and two peaks in the mean PSD curve correspond to two valleys on the standard deviation curve if the PSD value is expressed in the unit of dB. This representation is particularly useful for trivial signal details demonstration and analysis of noise effect on the useful signals with small amplitudes. Additionally, some subtle but desired signals can be found.

1.3.d Periodogram of complex exponential signals

The noisy complex exponential signal used to generate Figure 7 can be expressed as:

$$x[n] = e^{j\times 2\pi\times 0.3\times n} + e^{j\times 2\pi\times 0.32\times n} + w[n], \quad w[n]\sim\mathcal{N}(0, 0.2) \tag{1.9}$$

The frequency resolution of the periodogram is inversely proportional to the number of samples (N). When the value of N is small, two closely spaced spectral peaks are merged as one broad peak due to the spectral masking, and the desired signal frequencies cannot be correctly identified. To obtain two sharp peaks, more samples are required to improve frequency resolution. If the rectangular window is used, to achieve the resolution (i.e., $3\,\mathrm{dB}$ mainlobe width) of $0.02\,\mathrm{Hz}$ $(0.32\mathrm{Hz} - 0.3\mathrm{Hz})$, at least 45 samples are required.

$$N = \frac{0.89}{\Delta f} = \frac{0.89}{0.02} = 44.5 \approx 45 \tag{1.10}$$

However, in this task, 34 samples are sufficient to get two separated peaks despite the spectral leakage (i.e., two frequencies corresponding to two peak values are not exactly $0.3\mathrm{Hz}$ and $0.32\mathrm{Hz}$). The spectrum of the rectangular window is a sinc function containing mainlobe and sidelobes. It is these sidelobes that cause the

spectral leakage and transfer the power of correct frequency bins to their neighbouring bins which contain no signal power. To overcome this problem, N has to be larger.

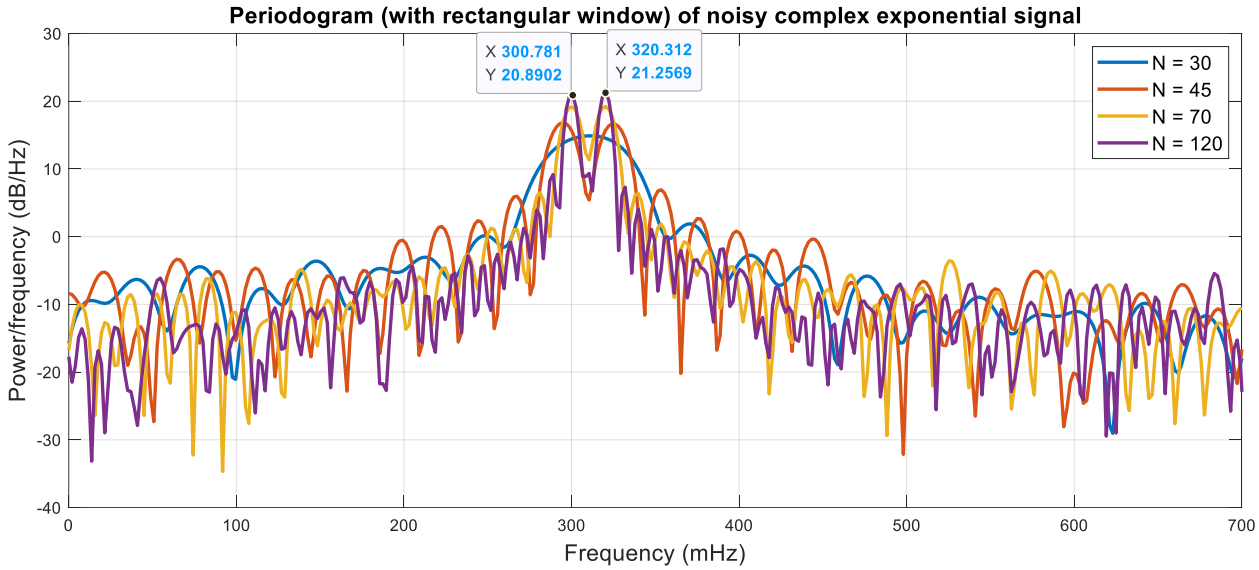

Figure 7. The periodogram of the noisy complex signal containing two exponentials with frequencies of 0.3 Hz and 0.32 Hz (512 frequency bins are used to smoothen the PSD curves)

1.3.e Frequency estimation by MUSIC (Multiple Signal Classification)

MUSIC can be classified as a subspace method and was implemented by MATLAB as:

[X,R] = corrmtx(x,14,'modified');
[S,F] = pmusic(R,2,[ ],1,'corr');
plot(F,S,'linewidth',2); set(gca,'xlim',[0.25 0.40]);

Firstly, the 'corrmtx()' function performs the autocorrelation matrix estimation for input signal 'x' with n samples. The input argument '14' determines the prediction model order (m). Since the matrix computation method is specified as 'modified', the first output 'X' is an $2(n-m)$-by-$(m+1)$ modified rectangular Toeplitz matrix and the second output 'R' is the $(m+1)$-by-$(m+1)$ biased estimate of the autocorrelation matrix for 'x'. R can be computed as $X^{\dagger}X$.

Secondly, the 'pmusic()' function implements the MUSIC algorithm to get the pseudospectrum estimate of the autocorrelation matrix R. Since the noisy signal only contains two exponentials, the signal subspace dimension 'p' is specified as 2. The empty matrix '[·]' means the default number of DFT points (256) which determines the number of spectrum estimates. The signal is sampled at the unit rate, and the sampling frequency is set as 1. The input argument 'corr' declares that the first argument R is the correlation matrix instead of the signal matrix. The output 'S' is the pseudospectrum estimates computed at the frequencies specified in 'F'. The length of 'S' and 'F' should be 256.

Finally, the relationship between pseudospectrum and normalized frequencies is displayed, and the frequency range is limited within $[0.25\text{ Hz}, 0.40\text{ Hz}]$.

When only 30 signal samples are available, the MUSIC algorithm provides more accurate details than the standard periodogram (Figure 7). Two sharp peaks can be observed. Although the spectral leakage problem cannot be perfectly avoided due to the randomness of noise, the leakage possibility is small and acceptable. Two candidate signal frequencies identified from the mean pseudospectrum are correct.

Essentially, MUSIC is a super-resolution method. It achieves high resolution and effectively separates closely spaced peaks. Thus, if the number of useful signal components (i.e., dimension of signal subspace - p) is known in advance or accurately estimated from the statistical properties (e.g., eigenvalues) of R, MUSIC method outperforms the standard periodogram method. One main disadvantage of MUSIC is that p is not

always available. If there are some useful components with small amplitudes and the p estimated is smaller than its actual value, the MUSIC results would be worse than the standard periodogram even if more signal samples are used. For instance, if p is changed to 1, two peaks in Figure 8 become one peak regardless of the value of N. When the noise level is significant and the estimated p is larger than its true value, some spurious peaks may appear.

Besides, the MUSIC estimates have a considerably large variance than the periodogram. The standard deviation values are still proportional to the corresponding PSD values, but the two peak values are greater than 300. Thus, the PSD peak amplitudes tend to have large variations between different realisations. Although it is easy to identify peak position and signal frequency, the pseudospectrum cannot reveal the relative strength comparison between useful signal components in each individual realisation. The bias and variance of estimation results are dependent on the noise level (or signal to noise ratio (SNR)). As the noise level increases, the variance drops while the bias rises, which means the peak identification accuracy decreases. Thus, to obtain a sufficiently accurate spectrum, a high SNR and reliable signal subspace dimension estimation algorithm are required.

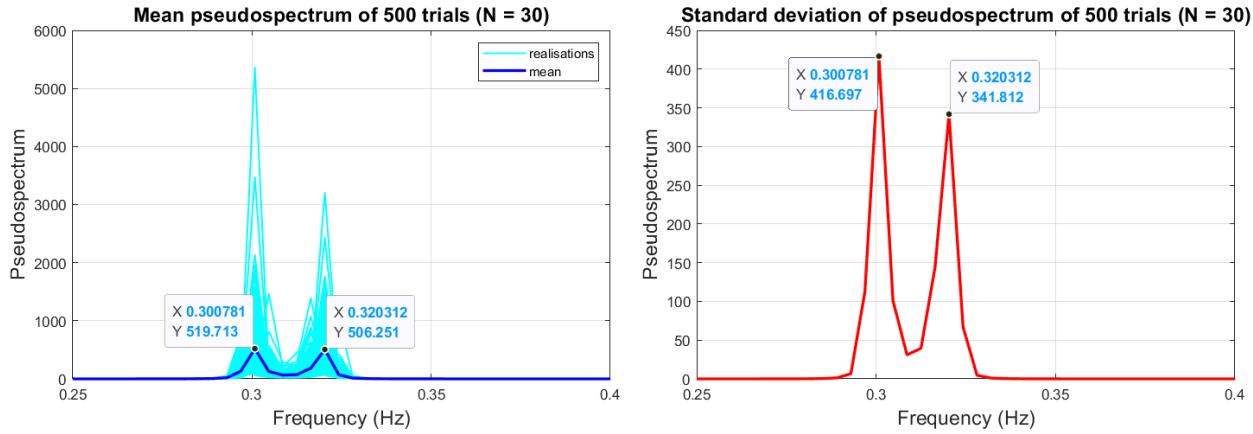

Figure 8. The pseudospectrum estimates of 500 different realizations of 2 complex exponentials in white noise and their mean and standard deviation estimates.

1.4 Spectrum of Autoregressive Processes

1.4.a The shortcoming of the unbiased ACF estimator

According to Yule-Walker (or called normal) equation (1.11), the set of autoregressive parameters ($\boldsymbol{r_x}(k)$) can be computed by equation (1.12).

$$\boldsymbol{r_x}(k) = a_1\boldsymbol{r_x}(k-1) + \cdots + a_k\boldsymbol{r_x}(0) + \cdots + a_p\boldsymbol{r_x}(p-k), \qquad \forall k \in [1,p] \tag{1.11}$$

$$\boldsymbol{r_x} = \boldsymbol{R_x}\boldsymbol{a} \rightarrow \boldsymbol{a} = \boldsymbol{R_x^{-1}}\boldsymbol{r_x} \tag{1.12}$$

Where $\boldsymbol{R_x}$ is the autocorrelation matrix, $\boldsymbol{r_x}(k)$ is the one autocorrelation estimate, $\boldsymbol{r_x}$ is the vector containing all these $\boldsymbol{r_x}(k)$ and all circularly shifted $\boldsymbol{r_x}$ form the $\boldsymbol{R_x}$.

Equation (1.12) requires the invertible $\boldsymbol{R_x}$. When the biased ACF estimator is used, the ACF matrix $\boldsymbol{R_{x,biased}}$ is positive definite and Toeplitz, which ensures the matrix inversion operation. However, if the unbiased ACF estimator is applied, the obtained ACF matrix may not be positive definite and the $\boldsymbol{R_{x,unbiased}}$ may be non-invertible. Consequently, the AR parameter set $\boldsymbol{a}$ cannot be obtained by equation (1.12).

1.4.b Effect of the order of underlying model

When the model order is small (e.g., $p = 4$), only one PSD peak can be observed, and the MSE (Mean Square Error) between the estimated and ground-truth PSD is large. As the model order increases to a relatively large value (e.g., $p = 6$), the single peak can be separated into two peaks, and the corresponding MSE error drops to a small level although both peaks are broad and shallow. If higher model order (e.g., $p = 12$) is applied, PSD peaks tend to have a similar shape to the ground truth PSD peaks, but the MSE error cannot be further significantly reduced due to the significant difference in the flat region (e.g., $[0,0.05\ \mathrm{Hz}] \cup$

$[0.2\ \text{Hz}, 0.5\ \text{Hz}]$) caused by the noise. If the model order is greater than a threshold value (e.g., 6 in this task), the MSE only has light variation.

If the model order becomes too large, the AR model would have an excessively high degree of freedom to fit the noisy data, and overfitting is much likely to occur, which may reduce the overall accuracy and increase the MSE. Besides, the high order results in large complexity and computation cost. Thus, higher-order does not always mean better fitting results. Although the model of order 6 results in the minimum MSE error, it has poor performance actually. Visually, the model of order 12 produces the best results.

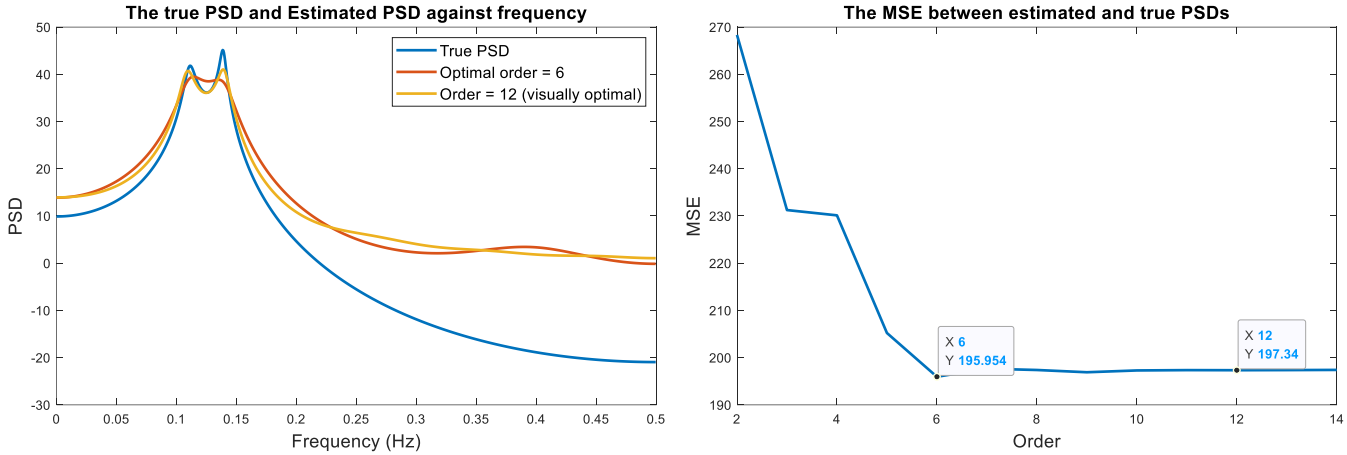

Figure 9. The true PSD and estimated PSD using different AR model as well as their MSE (500 signal samples are used)

1.4.c Effect of the order of underlying model when more samples are used

According to Figure 10, the large number of samples can effectively reduce the MSE between ground truth and estimated PSD and overcome the effect of random noise. Since more samples are available, the estimated model is closer to the correct model, and the bias becomes smaller. When the model order is lower than the correct order ($p = 4$), the underlying AR model does not have enough freedom degrees to fit these samples. Thus, the AR parameter and PSD estimation would be inaccurate, and peak identification would fail. When the order rises to 4, the minimum MSE can be achieved. If the order is higher than 4, the overfitting occurs. In the low-frequency range, the estimated PSD curve approximately overlaps the ground truth curve. But in the high-frequency range, the distance between two curves becomes large, which results in the slight increase in MSE error.

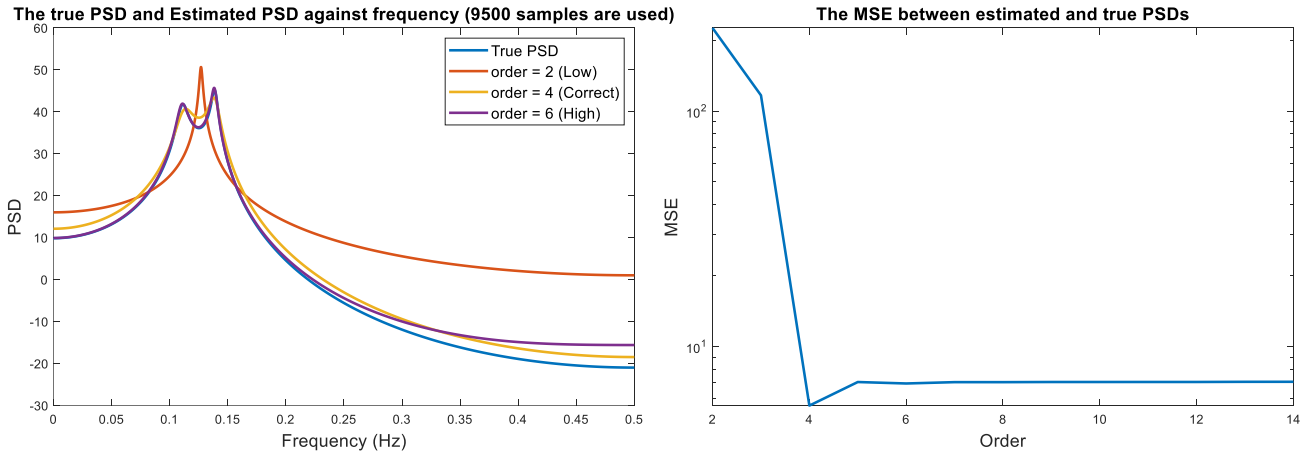

Figure 10. The true PSD and estimated PSD using different AR model as well as their MSE (9500 signal samples are used)

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

1.5.a Standard periodogram and averaged periodogram of RRI data

The PSDs of RRI data from three trials were estimated by the standard periodogram and averaged periodogram (i.e., Bartlett's method) with two window lengths (i.e., $50\ s$ and $150\ s$). Three trials correspond

to three breathing paces. The mean removal and detrending were applied to pre-process the raw data. Besides, the hamming window is applied, and there is no overlapping although 'pwelch()' function is selected.

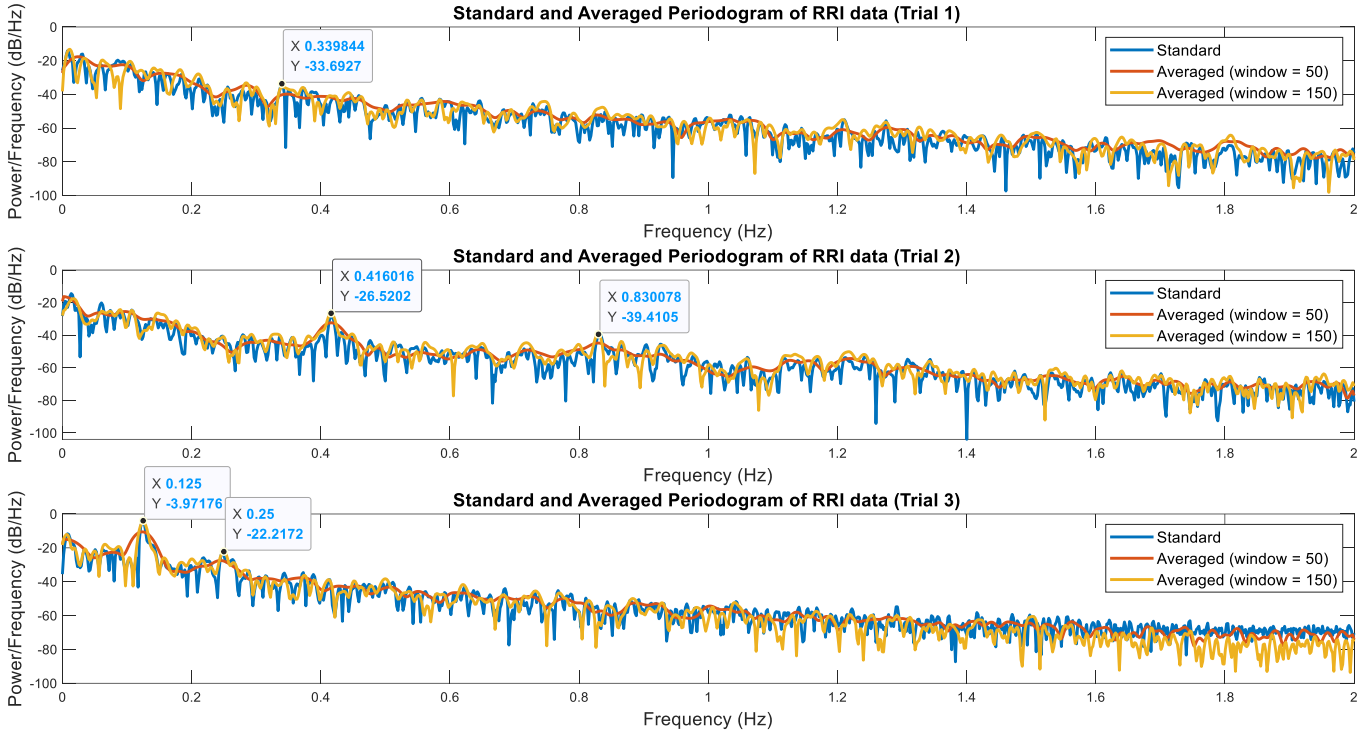

Figure 11. The standard periodograms and averaged periodograms for RRI signals for three trials

1.5.b Differences between the PSD estimates of the RRI data

Generally, the averaged periodogram with $50\ \mathrm{s}$ window length produces the smoothest PSD curve. The short window length effectively improves the frequency resolution at the cost of considerable variance reduction. The PSD curve estimated by the standard periodogram has considerably larger fluctuations than the other two curves, which may obscure true peaks. The averaged periodogram with $150\ \mathrm{s}$ window length results in moderate and appropriate fluctuation and may be most suitable for peak identification in this task. The unit conversion between Hertz and beats per minute (bpm) can be expressed as:

$$breathing\ rate = 2 \times 60 \times respiration\ frequency \tag{1.13}$$

Since the first trial data corresponds to unconstrained breathing, there is no primary frequency. Thus, no obvious sharp peak can be found. The broad peak at $0.340\ \mathrm{Hz}$ (i.e., $40.80\ \mathrm{bpm}$) may be a reasonable guess. The second and third trial corresponds to fast pace (i.e., $50\ \mathrm{bpm}$) breathing and slow pace (i.e., $15\ \mathrm{bpm}$) breathing respectively. Their peaks are obvious. The main peak of the second trial occurs at the frequency of $0.416\ \mathrm{Hz}$, representing $49.92\ \mathrm{bpm}$. The following small peak at the $0.832\ \mathrm{Hz}$ is the harmonic. The main peak of the third trial occurs at $0.125\ \mathrm{Hz}$ representing exactly $15.00\ \mathrm{bpm}$. The following harmonic at $0.25\ \mathrm{Hz}$ is also captured.

1.5.c AR spectrum estimate for the RRI signals

The optimal AR model orders of three trials are 24, 13 and 8 respectively. The spectral peak for the first trial is inconspicuous, and high model order must be used to highlight this peak at the cost over-modelling problem. The respiration frequencies estimated by the AR spectrum and periodogram are slightly different, but the largest difference is lower than 0.016 which is acceptable. When the PSD curve has obvious peaks, lower model order should be used to avoid over-fitting and spurious peaks. Therefore, the AR spectrum estimates are usually significantly smoother than the periodogram estimates and can only show the overall trend and prominent peaks. Many PSD variation details caused by noise and harmonics are ignored, which may simplify the peak identification. At the same time, due to the small variance, the peak magnitudes of the AR spectrum are relatively small, which is a source of peak identification error. Generally, if high accuracy is required, the standard periodogram or averaged periodogram with a reasonably long window size would perform better than the AR spectrum method.

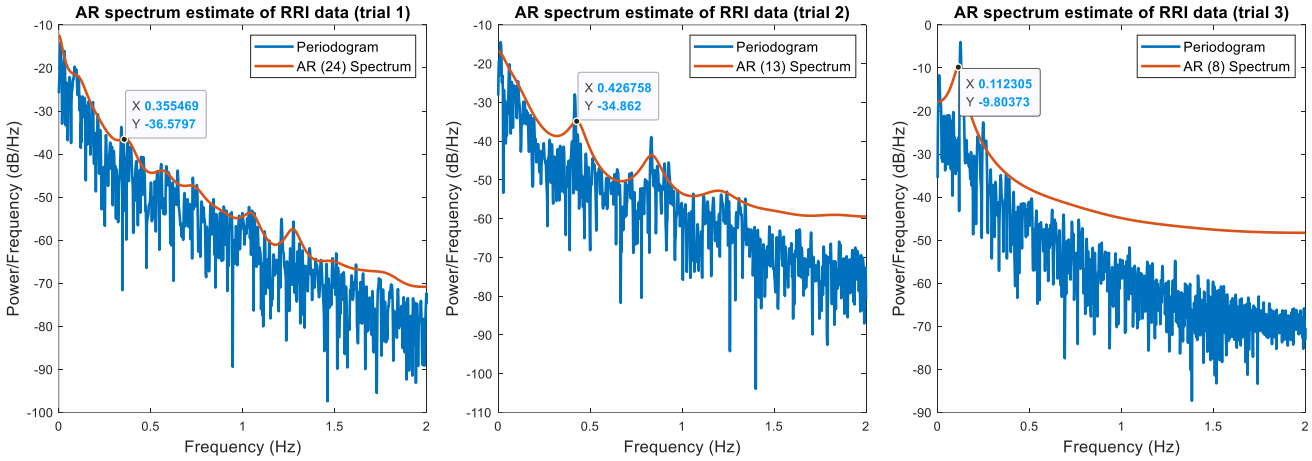

Figure 12. The AR spectrum estimates for RRI signals for three trials

1.6 Robust Regression

1.6.a Singular value and rank of a matrix

In Figure 13, when the index is greater than 3, the singular value of $\boldsymbol{X}$ becomes approximately zero. Thus, the rank of $\boldsymbol{X}$ can be estimated as 3. By contrast, because the white noise exists in three signal subspaces, the first three singular values of $\boldsymbol{X_{noise}}$ are greater those of $\boldsymbol{X}$ and all 10 singular values of $\boldsymbol{X_{noise}}$ are non-zero, which means that the rank of $\boldsymbol{X_{noise}}$ should be 10 in theory. However, since the SNR is relatively large, the first three dominant singular values of $\boldsymbol{X_{noise}}$ corresponding to three useful signal subspaces are considerably larger than the last seven trivial singular values corresponding to seven noise subspaces. $\boldsymbol{X_{noise}}$ can be approximately considered as a rank-3 matrix. In Figure 13, the signal and noise subspace dimensions can be easily identified. However, if the SNR drops to a small level, the noise components may have comparable magnitudes with the signal components, and the singular value difference becomes small. Thus, the signal and noise subspace may not be easily distinguished, and matrix rank identification becomes difficult.

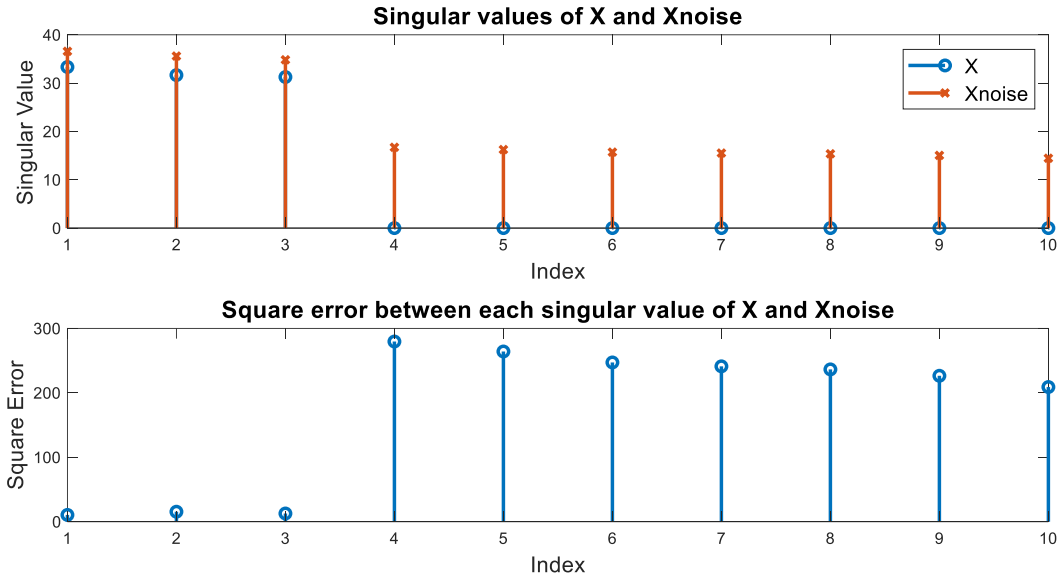

Figure 13. The singular values of $\boldsymbol{X}$ and $\boldsymbol{X_{noise}}$ as well as the square error of each singular value

1.6.b MSE error between clean signal and noisy signal and denoised signal

To reduce the noise, only three most significant principal components are retained. The denoised matrix $\widehat{\boldsymbol{X}}_{\boldsymbol{noise}}$ is the low-rank approximation of $\boldsymbol{X_{noise}}$. According to Figure 14, $\widehat{\boldsymbol{X}}_{\boldsymbol{noise}}$ has considerably smaller MSE error in each column than $\boldsymbol{X_{noise}}$ since the noise in seven noise subspaces are eliminated. Besides, since the noise is three signal subspaces are retained, the error cannot be reduced to exactly 0. The white noise results in similar MSE errors among columns of $\boldsymbol{X_{noise}}$, and different error reduction levels in different columns of $\widehat{\boldsymbol{X}}_{\boldsymbol{noise}}$ can be attributed to the randomness of noise. In practice, if the noisy matrix contains some trivial useful signals, these trivial signals may be wrongly recognized as noise and removed. But since

most information is stored within a few significant principal components, the overall signal loss is acceptable.

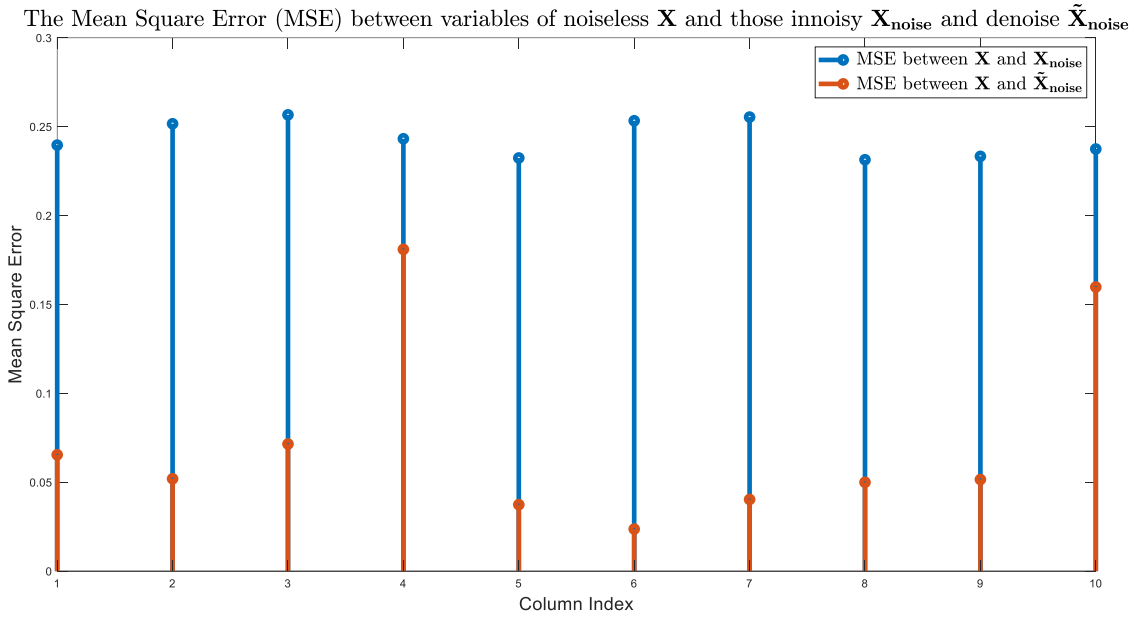

Figure 14. The MSE between clean signal $\boldsymbol{X}$, noisy signal $\boldsymbol{X_{noise}}$ and denoised signal $\widehat{\boldsymbol{X}}_{\boldsymbol{noise}}$

1.6.c OLS and PCR solutions and their errors

To solve the sub-rank problem of OLS, the pseudo matrix inversion was applied. The OLS and PCR methods result in similar MSE estimation errors for both test and training data sets. In Table 1, the $\widehat{\boldsymbol{Y}}_{\boldsymbol{OLS}}$ is slightly closer to the $\boldsymbol{Y}$ than $\widehat{\boldsymbol{Y}}_{\boldsymbol{PCR}}$ although its error improvement is smaller than 0.01. In this case, only the training data is used. The advantage of OLS based method cannot guarantee its high performance for other data due to the potential overfitting problem.

Table 1. Mean Square error between $\boldsymbol{Y}$ and $\widehat{\boldsymbol{Y}}_{\boldsymbol{OLS}}$ and $\widehat{\boldsymbol{Y}}_{\boldsymbol{PCR}}$ (Training set data)

Column Index	1	2	3	4	5	Overall
OLS	1.161	0.611	0.573	0.383	0.824	0.710
PCR	1.168	0.616	0.579	0.387	0.827	0.715

In Table 2, the test data was used. In this case, PCR solution results in smaller estimation errors, which validates that PCR outperforms the OLS solution. By comparing their overall MSE errors, it can be found that PCR error is 1.086% smaller than the OLS error. OLS has the overfitting problem.

Table 2. Mean Square error between $\boldsymbol{Y_{test}}$ and $\widehat{\boldsymbol{Y}}_{\boldsymbol{OLS}}$ and $\widehat{\boldsymbol{Y}}_{\boldsymbol{PCR}}$ (Test set data)

Column Index	1	2	3	4	5	Overall
OLS	0.964	0.390	0.343	0.132	0.549	0.475
PCR	0.960	0.385	0.334	0.129	0.543	0.470

1.6.d Effectiveness of the PCR compared to the OLS solution

In the last section, only one realisation of test data is used, and the computed error is not accurate enough. In this section, 5000 realisations were generated to produce 5000 groups of MSE errors. The mean value of 5000 individual errors of either PCR or OLS solution was computed to get more reliable results and support a more accurate comparison. Generally, Table 2 has high similarity with Table 3. PCR and OLS solutions result in similar estimation errors, but the PCR is a better choice since its overall error (i.e., 0.471) is 0.831% smaller than OLS error (i.e., 0.467). This slight advantage may become important when there are strict requirements on model accuracy.

Table 3. Mean value of 5000 MSEs between $\boldsymbol{Y_{test}}$ and $\widehat{\boldsymbol{Y}}_{\boldsymbol{OLS}}$ and $\widehat{\boldsymbol{Y}}_{\boldsymbol{PCR}}$ (Test set data)

Column Index	1	2	3	4	5	Overall
OLS	0.933	0.394	0.333	0.136	0.560	0.471
PCR	0.928	0.389	0.329	0.133	0.557	0.467

2 Adaptive signal processing

2.1 The Least Mean Square (LMS) Algorithm

2.1.a LMS stability bound

The correlation matrix of the input vector $\boldsymbol{x}(n) = [x(n-1),\ x(n-2)]^T$ can be expressed as:

$$R_{xx} = \mathbb{E}\{\boldsymbol{x}(n)\boldsymbol{x}^T(n)\} \tag{2.1}$$

$$= \mathbb{E}\left\{\begin{bmatrix} x(n-1) \\ x(n-2) \end{bmatrix} \times [x(n-1) \quad x(n-2)]\right\} = \mathbb{E}\left\{\begin{bmatrix} x^2(n-1) & x(n-1)x(n-2) \\ x(n-1)x(n-2) & x^2(n-2) \end{bmatrix}\right\}$$

$$= \begin{bmatrix} \mathbb{E}\{x^2(n-1)\} & \mathbb{E}\{x(n-1)x(n-2)\} \\ \mathbb{E}\{x(n-1)x(n-2)\} & \mathbb{E}\{x^2(n-2)\} \end{bmatrix} = \begin{bmatrix} r_{xx}(0) & r_{xx}(1) \\ r_{xx}(1) & r_{xx}(0) \end{bmatrix}$$

Where $r_{xx}(k)$ is the ACF of the real wide-sense stationary (WSS) signal $x(n)$. $r_{xx}(k)$ can be recursively computed from previous values.

$$r_{xx}(k) = \mathbb{E}\{x(n)x(n-k)\} = \mathbb{E}\{(a_1x(n-1) + a_2x(n-2) + \eta(n))x(n-k)\} \tag{2.2}$$

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

$$= a_1r_{xx}(k-1) + a_2r_{xx}(k-2) + \mathbb{E}\{\eta(n)x(n-k)\}$$

$$\mathbb{E}\{\eta(n)x(n-k)\} = a_1\mathbb{E}\{\eta(n)x(n-k-1)\} + a_2\mathbb{E}\{\eta(n)x(n-k-2)\} + \mathbb{E}\{\eta(n)\eta(n-k)\} \tag{2.3}$$

The noise $\eta(n)$ is the white Gaussian noise. Thus, $\eta(n)$ and signal $x(m)$ and noise at other time steps $\eta(n-k)\ (k \neq 0)$ are uncorrelated. The equation (2.3) can be simplified to equation (2.4).

$$\mathbb{E}\{\eta(n)x(n-k)\} = \mathbb{E}\{\eta(n)\eta(n-k)\} = \begin{cases} 0, & k \neq 0 \\ \sigma_\eta^2, & k = 0 \end{cases} \tag{2.4}$$

The equation (2.2) is updated as follows:

$$r_{xx}(k) = \begin{cases} a_1r_{xx}(k-1) + a_2r_{xx}(k-2), & k \neq 0 \\ a_1r_{xx}(k-1) + a_2r_{xx}(k-2) + \sigma_\eta^2, & k = 0 \end{cases} \tag{2.5}$$

To compute $r_{xx}(0)$ and $r_{xx}(1)$, three equations are listed and solved.

$$\begin{cases} r_{xx}(0) = a_1r_{xx}(1) + a_2r_{xx}(2) + \sigma_\eta^2 \\ r_{xx}(1) = a_1r_{xx}(0) + a_2r_{xx}(1) + 0 \\ r_{xx}(2) = a_1r_{xx}(1) + a_2r_{xx}(0) + 0 \end{cases} \rightarrow \begin{bmatrix} 1 & -a_1 & -a_2 \\ a_1 & a_2 - 1 & 0 \\ a_2 & a_1 & -1 \end{bmatrix} \begin{bmatrix} r(0) \\ r(1) \\ r(2) \end{bmatrix} = \begin{bmatrix} \sigma_\eta^2 \\ 0 \\ 0 \end{bmatrix} \tag{2.6}$$

The $r_{xx}(0) = 0.926$ and $r_{xx}(1) = 0.463$ can be computed, and the R_{xx} is:

$$R_{xx} = \begin{bmatrix} r_{xx}(0) & r_{xx}(1) \\ r_{xx}(1) & r_{xx}(0) \end{bmatrix} = \begin{bmatrix} 0.926 & 0.463 \\ 0.463 & 0.926 \end{bmatrix} \tag{2.7}$$

According to equation (2.8), the largest eigenvalues of R_{xx} can be obtained. ($\lambda_{max} = 1.389$)

$$\begin{bmatrix} 0.926 & 0.463 \\ 0.463 & 0.926 \end{bmatrix} \times \begin{bmatrix} -0.707 & 0.707 \\ 0.707 & 0.707 \end{bmatrix} = \begin{bmatrix} -0.707 & 0.707 \\ 0.707 & 0.707 \end{bmatrix} \begin{bmatrix} 0.463 & 0 \\ 0 & 1.389 \end{bmatrix} \tag{2.8}$$

To ensure the convergence of LMS, the absolute value of weight error should shrink in each iteration.

$$|1 - \mu\lambda_{max}| < 1 \rightarrow 0 < \mu < 2/\lambda_{max} = 1.44 \tag{2.9}$$

$$\mu \in (0, 1.44)$$

Thus, the LMS would converge to the mean when the step size ranges from 0 to 1.44.

2.1.b LMS adaptive predictor

The equation (2.10) was used as the weight update equation for LMS adaptive predictor implementation. Then, this predictor is applied to the AR(2) process $x(n)$, and the individual and mean squared prediction errors ($e^2(n)$) are plotted in Figure 15. When only one realisation of $x(n)$ is analysed, the effect of step size on LMS convergence cannot be observed due to the large variance of error. According to the learning curves averaged over 100 realisations, the larger step size ensures higher rate of convergence because it results in larger variation in the weight vector updating step. When the step size (μ) is set as 0.05, the LMS method converges within about 100 samples despite some fluctuation in steady-state. By comparison, when the step size is set as 0.01, the LMS converges after nearly 200 samples. Theoretically, larger step size is likely to result in a larger steady-state bias error. However, such bias error difference is slight and hardly

distinguishable due to the large variance error and the relatively small step size difference.

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu e(n)\mathbf{x}(n), \quad e(n) = x(n) - \hat{x}(n) = x(n) - \boldsymbol{w}(n)\mathbf{x}(n) \tag{2.10}$$

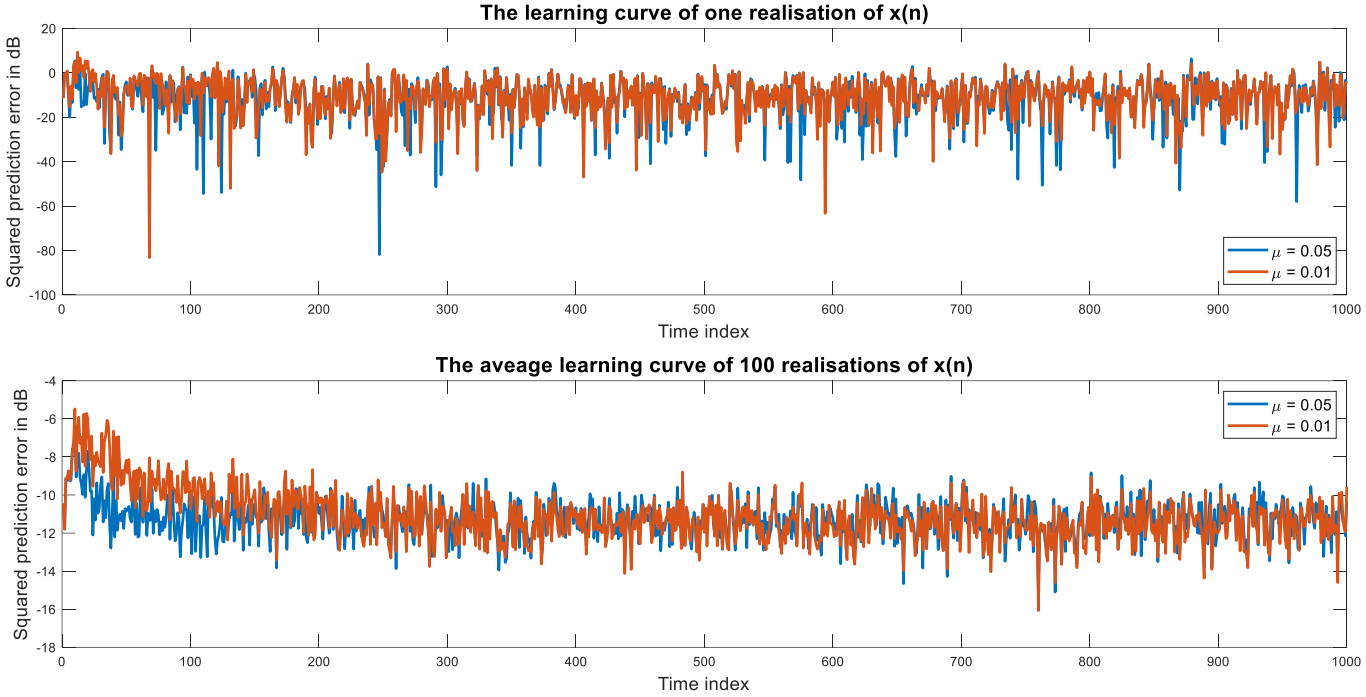

Figure 15. The individual and averaged learning curves of the LMS method with different step sizes

2.1.c The measured and approximated misadjustments of LMS

Table 4. The estimated and theoretical misadjustments of the LMS

Step Size μ	MSE	EMSE	$\mathcal{M}_{estimated}$	$\mathcal{M}_{\text{theoretical}}$
0.05	0.2622	0.0122	0.0488	0.0463
0.01	0.2518	0.0018	0.0074	0.0093

By time-averaging over the steady-state squared prediction errors (e.g., the last 500 errors when N=1000), the mean square errors (MSE_i) in 100 trials (Monte Carlo simulations) can be obtained. The mean value of all individual MSE_i can be approximated as the ideal MSE. Since the minimum attainable MSE of Wiener filter is exactly the noise variance that is known in advance, the excess mean square error (EMSE) can be computed.

$$EMSE = MSE - \sigma_\eta^2 = \lim_{n\to\infty} \mathbb{E}\{e^2(n)\} - \sigma_\eta^2 = \frac{1}{100}\sum_{i=1}^{100} MSE_i - \sigma_n^2 \tag{2.11}$$

$$\mathcal{M}_{estimated} = EMSE/\sigma_\eta^2 \qquad \mathcal{M}_{\text{theoretical}} \approx 0.5\mu\,\text{Tr}(R_{xx}) \tag{2.12}$$

Both step sizes used in this task are sufficiently small, and the theoretical misadjustment ($\mathcal{M}_{\text{theoretical}}$) approximation should be reliable. For $\mu = 0.05$, the difference between $\mathcal{M}_{estimated}$ and $\mathcal{M}_{\text{theoretical}}$ is 0.0025, and the estimation error is 5.39%. For $\mu = 0.01$, the difference between $\mathcal{M}_{estimated}$ and $\mathcal{M}_{\text{theoretical}}$ is 0.0019 and the estimation error is 20.27%. Generally, both difference values are sufficiently small ($|\mathcal{M}_{\text{theoretical}} - \mathcal{M}_{\text{estimated}}| < 0.003$), which proves the accuracy of both estimation and approximation methods of $\mathcal{M}$. The large error 20.27% can be attributed to the small $\mathcal{M}_{\text{theoretical}}$ and randomness of noise. If more steady-state MSE errors are available (e.g., N=10000), the computed error corresponding to $\mu = 0.01$ can be further reduced.

Besides, both estimated and theoretical $\mathcal{M}$ in Table 4 show the tendency that smaller step size results in smaller EMSE and consequently smaller misadjustment despite the slower convergence. If the step size is too large, the LMS cannot find the optimal weight vector estimate but fluctuate around the optimal solution with large variation. Therefore, if there is no strict requirement on computation cost and convergence rate, a reasonably small step size should be selected.

2.1.d Steady-state values of the adaptive filter coefficients

Table 5. The steady-state values of the adaptive filter coefficients

Step Size μ	Estimated $\hat{a}_1$	Error in $\hat{a}_1$	Estimated $\hat{a}_2$	Error in $\hat{a}_2$
0.05	0.077	23.260 %	0.728	8.989 %
0.01	0.093	6.930 %	0.773	3.427 %

In Figure 16, coefficients of 100 independent trials in each iteration are averaged to plot these mean coefficient updating curves. Compared with the squared error plotted in Figure 15, these curves have less variation. Thus, it becomes easier to detect when coefficients reach their steady states and when the algorithm converges. Besides, the steady-state coefficients estimated in the last iteration and their corresponding errors are listed in Table 5.

As mentioned in the previous two sections, the large step size ($\mu = 0.05$) enables the LMS to converge within 200 samples at the cost of the large coefficient estimation error. The true coefficient $a_1 = 0.1$ is wrongly estimated as 0.077, and the error reaches $23.260\ \%$ which is unacceptably large. Due to the large updating term $\mu e(n)\mathbf{x}(n)$ caused by large μ, the updated coefficient vector $\boldsymbol{w}(n+1)$ is much likely to overshoot the optimal solution and fluctuate around it with large excess error. When the step size is small ($\mu = 0.01$), the LMS requires about 700 samples to converge, but the estimated coefficients of the AR(2) model are close to the ground truth values. Both coefficient errors are controlled under 7%. The small updating term slows down convergence but reduces the excess error. Thus, the step size should be well adjusted to trade off coefficient error against the convergence rate.

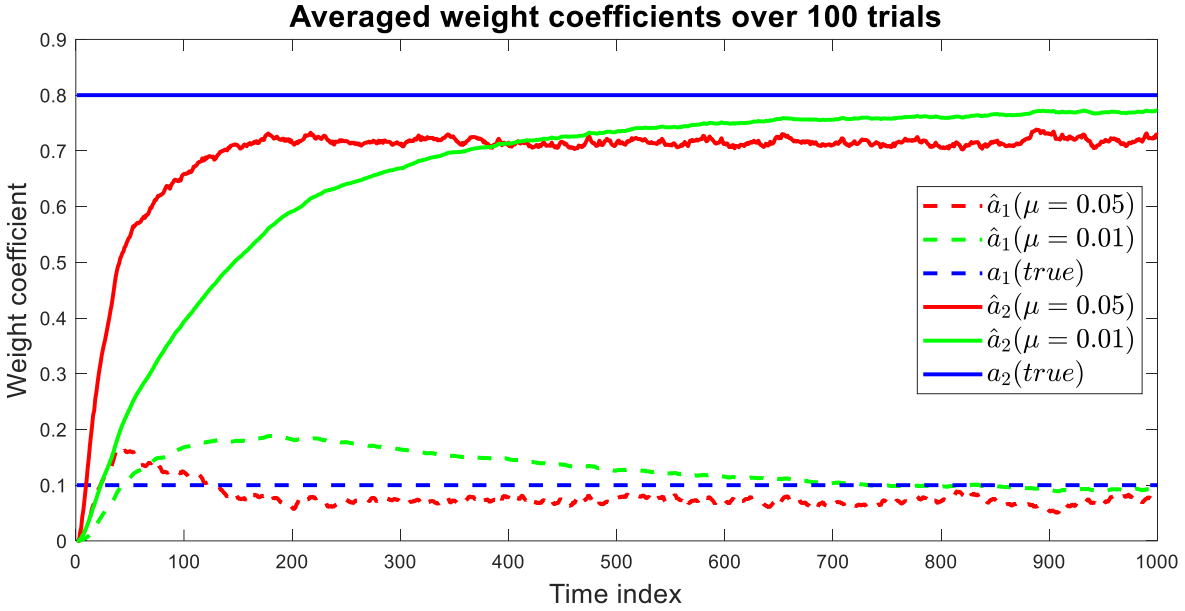

Figure 16. The values of the adaptive filter coefficients for two different step sizes

2.1.e Leaky LMS coefficient update equation

According to the new definition of the cost function (2.13), the gradient of $J_2(n)$ with respect to weight coefficient vector $\boldsymbol{w}(n)$ can be derived as (2.14).

$$J_2(n) = \frac{1}{2}(e^2(n) + \gamma\|\boldsymbol{w}(n)\|_2^2) = \frac{1}{2}e^2(n) + \frac{\gamma}{2}\|\boldsymbol{w}(n)\|_2^2 \tag{2.13}$$

$$\nabla_{\boldsymbol{w}} J_2(n) = \frac{\partial J_2(n)}{\partial \boldsymbol{w}(n)} = \frac{1}{2}\frac{\partial e^2(n)}{\partial e(n)}\frac{\partial e(n)}{\partial y(n)}\frac{\partial y(n)}{\partial \boldsymbol{w}(n)} + \frac{\gamma}{2}\frac{\partial\left(\boldsymbol{w}^T(n)\boldsymbol{w}(n)\right)}{\partial \boldsymbol{w}(n)} \tag{2.14}$$

$$= \tfrac{1}{2} \times 2e(n) \times (-1) \times \tfrac{\partial(\boldsymbol{w}(n)\mathbf{x}(n))}{\partial \boldsymbol{w}(n)} + \tfrac{\gamma}{2} \times 2\boldsymbol{w}(n) = -e(n)\mathbf{x}(n) + \gamma\boldsymbol{w}(n)$$

The steepest descent (SD) method can be used to get the general weight update equation (2.15). By substituting (2.14) to (2.15), the leaky LMS weight update equation can be obtained.

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) - \mu\nabla_{\boldsymbol{w}} J_2(n) \tag{2.15}$$

$$= \boldsymbol{w}(n) + \mu e(n)\mathbf{x}(n) - \mu\gamma\boldsymbol{w}(n) = (1 - \mu\gamma)\boldsymbol{w}(n) + \mu e(n)\mathbf{x}(n), \quad 0 < \gamma < 1 \tag{2.16}$$

2.1.f Leaky LMS implementation

For different step sizes μ and leakage coefficients γ, the estimated steady-state values of AR coefficients of the given signal are listed in Table 6. Similarly, 100 trials were done, and the coefficients generated in the final iterations were averaged to overcome the randomness problem.

The convergence can still be guaranteed by the leaky LMS. However, the weight coefficients converge to incorrect values and large bias errors are obtained especially when a large γ is selected. In ideal cases, the optimum weight vector $\boldsymbol{w}_{opt}$ can be obtained by the Wiener filter if the auto-correlation matrix $\boldsymbol{R}$ is invertible. $\mathbf{x}(n)$ is the filter input, and $\boldsymbol{d}(n)$ is the desired response.

$$\nabla_{\boldsymbol{w}} J(n) = -\boldsymbol{p} + \boldsymbol{R}\mathbf{w} = 0 \rightarrow \boldsymbol{w}_{opt} = \boldsymbol{R}^{-1}\boldsymbol{p}, \qquad \boldsymbol{R} = E\{\mathbf{x}(n)\mathbf{x}^{\boldsymbol{T}}(n)\}, \qquad \boldsymbol{p} = E\{\boldsymbol{d}(n)\mathbf{x}(n)\} \tag{2.17}$$

However, in practice, it is possible that $\boldsymbol{R}$ is a singular matrix or has some small eigenvalues. The standard LMS slowly converges in these directions with extremely small eigenvalues. This problem can be effectively solved by introducing the leaky coefficient. When leaky LMS is applied, the optimal solution $\boldsymbol{w}_{opt,leaky}$ which the algorithm converges to becomes:

$$\nabla_{\boldsymbol{w}} J_2(n) = -\boldsymbol{p} + \boldsymbol{R}\mathbf{w} + \gamma\mathbf{w} = 0 \rightarrow \boldsymbol{w}_{opt,leaky} = (\boldsymbol{R} + \gamma\boldsymbol{I})^{-1}\boldsymbol{p} \tag{2.18}$$

The additional full-rank term $\gamma\boldsymbol{I}$ in $\boldsymbol{w}_{opt,leaky}$ ensures the invertibility and large eigenvalues of the matrix $\boldsymbol{R} + \gamma\boldsymbol{I}$ but also results in the deviation from the Wiener-Hopf solution $\boldsymbol{w}_{opt}$. Thus, the γ is positively correlated to the steady-state error of coefficient estimates. When γ drops to zero, the leaky LMS becomes the standard LMS algorithm, and more accurate estimates are obtained. Besides, according to the positive μ in equation (2.16), the leaky LMS has a lower sensibility to current coefficient values $\boldsymbol{w}(n)$, and the updating term $\mu e(n)\mathbf{x}(n)$ makes more contribution to the new coefficients $\boldsymbol{w}(n+1)$, which speeds up the convergence. Generally, leaky LMS effectively overcomes slow convergence caused by the large eigenvalue spread. [1] But this algorithm is redundant and useless in this task.

Table 6. The coefficients estimated by the leaky LMS algorithm after 1000 iterations (samples) and their errors

μ	γ	Real a_1	Estimated $\hat{a}_1$	Error in $\hat{a}_1$	Real a_2	Estimated $\hat{a}_2$	Error in $\hat{a}_2$
0.05	0.1	0.1	0.1012	1.197%	0.8	0.6314	21.077%
0.05	0.3	0.1	0.1189	18.939%	0.8	0.5060	36.752%
0.05	0.5	0.1	0.1213	21.279%	0.8	0.4267	46.658%
0.05	0.7	0.1	0.1187	18.681%	0.8	0.3713	53.590%
0.05	0.9	0.1	0.1145	14.494%	0.8	0.3299	58.767%
0.01	0.1	0.1	0.1188	18.846%	0.8	0.6796	15.054%
0.01	0.3	0.1	0.1393	39.253%	0.8	0.5526	30.924%
0.01	0.5	0.1	0.1423	42.281%	0.8	0.4689	41.382%
0.01	0.7	0.1	0.1391	39.120%	0.8	0.4089	48.886%
0.01	0.9	0.1	0.1338	33.756%	0.8	0.3634	54.573%

2.2 Adaptive Step Sizes

2.2.a Three GASS algorithms

Compared with the standard LMS algorithm with a fixed step size, the Gradient Adaptive Step Size (GASS) has higher computational complexity due to the additional step size update process. But it can effectively solve the conflict between fast convergence and small steady-state error. Usually, the initial guess of weight coefficients tends to have large difference from the optimal weights. Thus, a reasonably large value can be used as the initial step size $\mu_{initial}$ to achieve the high rate of convergence at the beginning. As the weight estimates get close to the optimal weights, the prediction error drops and the step size becomes small to reduce the oscillation and ensure a small steady-state error.

In Figure 17 and Figure 18, two different initial steps are used to exhibit the difference between standard LMS and Benveniste, Ang & Farhang and Matthews & Xie GASS algorithms. When $\mu_{initial}$ in GASS is small (e.g.,

0.01), only Benveniste GASS algorithm shows the overwhelming advantage of convergence rate over LMS with a large step size (e.g., 0.1). It converges within about 60 iterations (samples). The standard LMS with $\mu = 0.1$ and Ang & Farhang GASS with $\alpha = 0.7$ have similar performance and converge within 100 iterations. The Matthews & Xie GASS requires 200 iterations to converge and is just better than LMS with $\mu = 0.01$ which requires 1000 iterations to converge. The steady-state weight errors are listed in Table 7. It can be observed that all algorithms achieve comparably small steady-state weight errors and squared prediction errors except the standard LMS with $\mu = 0.1$.

The Benveniste algorithm utilises the correct expression for gradient $\nabla_\mu E(n)$ where $E(n)$ is the squared prediction error to generate a new $\boldsymbol{\psi}(n)$, which results in the highest accuracy of gradient estimation and consequently best performance among all three algorithms. [2] However, due to the matrix multiplication, the Benveniste algorithm has the highest complexity of $O(N^2)$ (where N is the order of AM process). Ang & Farhang algorithm uses a low pass filter with a fix coefficient α. When a large α (e.g., 0.95) is used, the Ang & Farhang algorithm achieves comparable performance with Benveniste algorithm but has lower computational complexity $O(N)$ than Benveniste algorithm. When α drops to 0, the Ang & Farhang algorithm becomes exactly Matthews & Xie algorithm which only uses a noisy gradient and has the lowest complexity $O(N)$. But it also has the lowest accuracy of gradient estimation and worst adaption and performance.

The performances of GASS algorithms depend on the initial step size. When $\mu_{initial}$ becomes sufficiently large (e.g., 0.3), three GNGD algorithms achieve similar weight error curves and show significantly better performance than standard LMS algorithm. However, the range of $\mu_{initial}$ is also limited by equation (2.9).

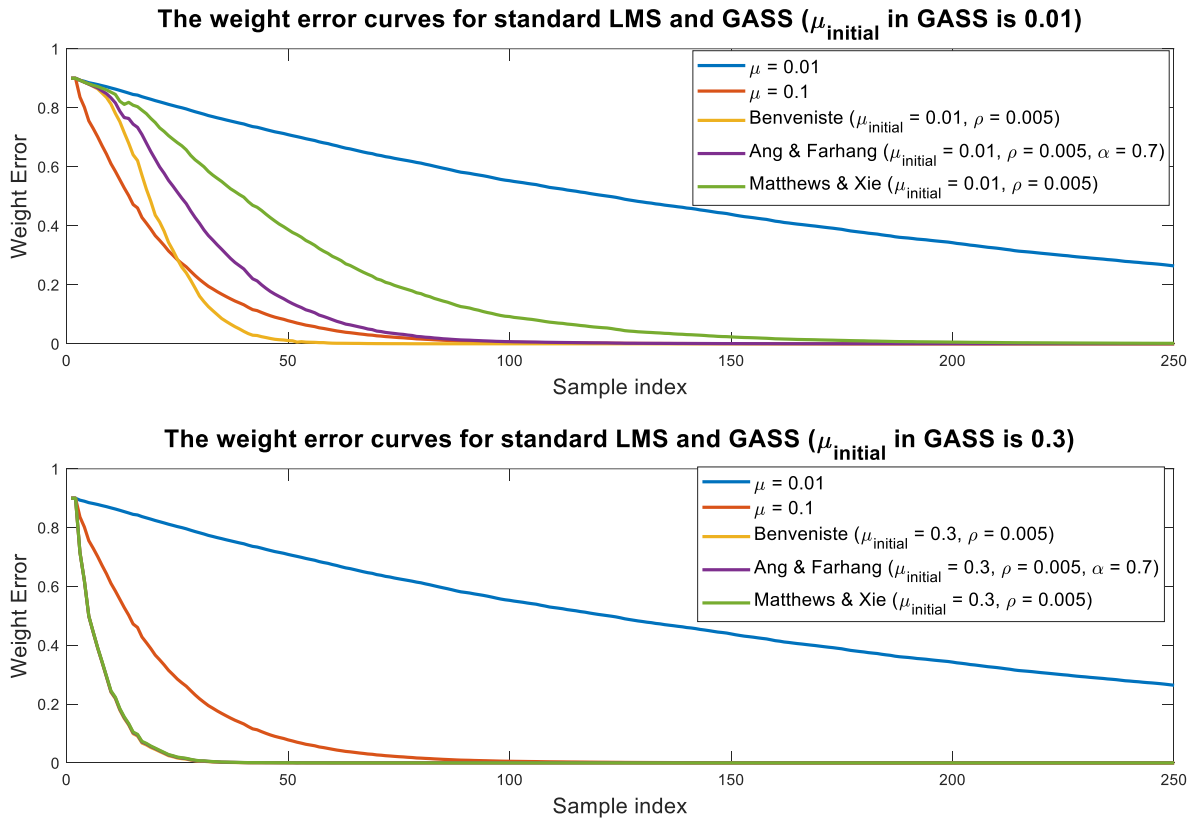

Figure 17. The weight error curves for standard LMS and three GASS algorithms

Table 7. Steady-state weight errors and squared prediction errors after 1000 iterations (samples)

	LMS $\mu = 0.01$	LMS $\mu = 0.1$	Benveniste	Ang & Farhang	Matthews & Xie
weight error	6.2×10^{-3}	6.7×10^{-18}	-3.3×10^{-18}	-3.3×10^{-18}	3.1×10^{-11}
squared prediction error	-43.4 dB	-321.1 dB	-320.7 dB	-322.5 dB	-191.9 dB

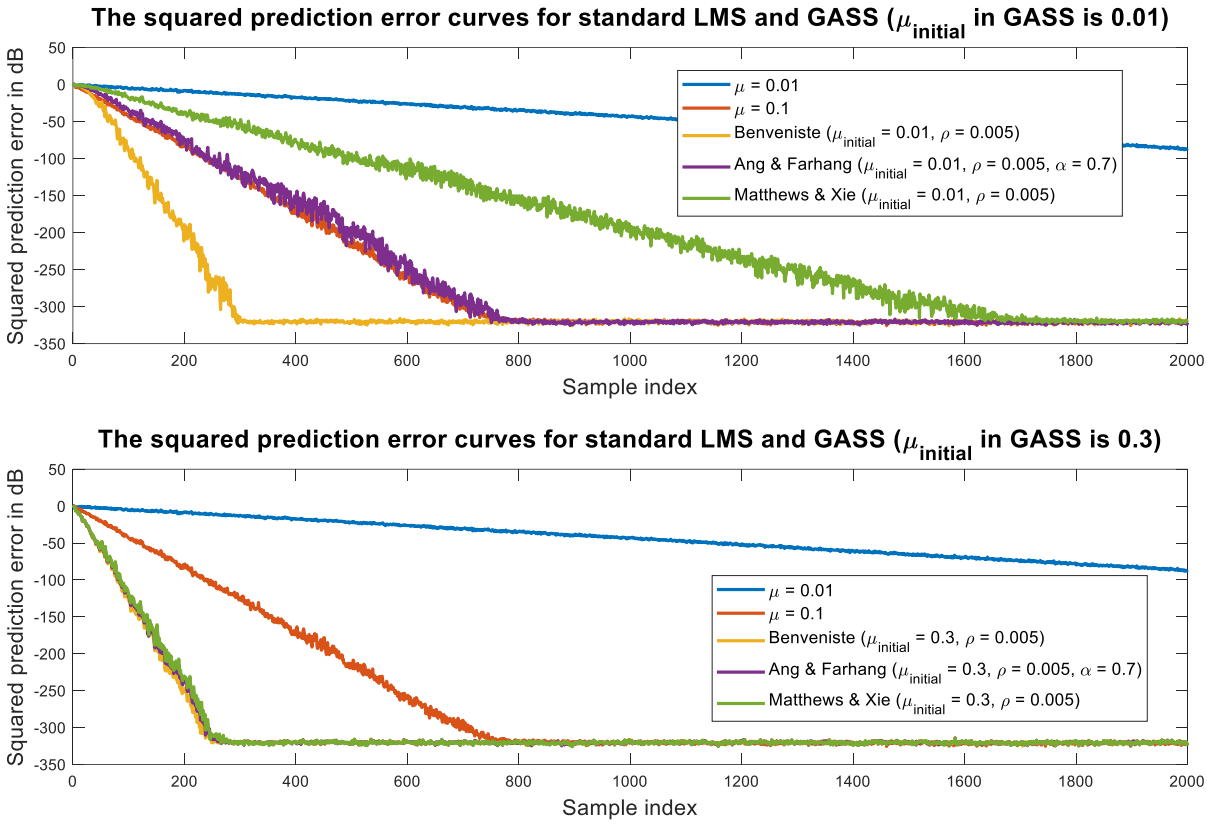

Figure 18. The squared prediction error curves in dB for standard LMS and three GASS algorithms

2.2.b Equivalence between posteriori error-based weight update equation and NLMS

The prior error $e(n)$ can be expressed as:

$$e(n) = d(n) - \boldsymbol{w}^T(n)\mathbf{x}(n) \tag{2.19}$$

The weight update equation is

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu e_p(n)\mathbf{x}(n) \tag{2.20}$$

Where $e_p(n)$ is the posteriori error and can be expressed as:

$$e_p(n) = d(n) - \boldsymbol{w}^T(n+1)\mathbf{x}(n) \tag{2.21}$$

The $\boldsymbol{w}(n+1)$ term in (2.21) can be substituted by (2.20), and the $e_p(n)$ expression becomes:

$$e_p(n) = d(n) - \Big(\boldsymbol{w}(n) + \mu e_p(n)\mathbf{x}(n)\Big)^T \mathbf{x}(n) = \big(d(n) - \boldsymbol{w}^T(n)\mathbf{x}(n)\big) - \mu e_p(n)\mathbf{x}^T(n)\mathbf{x}(n) \tag{2.22}$$

$$= e(n) - \mu e_p(n)\mathbf{x}^T(n)\mathbf{x}(n)$$

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

According to this new expression of $e_p(n)$, the new weight update equation can be re-written as:

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu\frac{e(n)}{1 + \mu\|\mathbf{x}(n)\|^2}\mathbf{x}(n) \tag{2.23}$$

The difference between current and updated weight vectors in terms of $e(n)$ now becomes:

$$\Delta\boldsymbol{w}(n) = \boldsymbol{w}(n+1) - \boldsymbol{w}(n) = \mu\frac{e(n)}{1 + \mu\|\mathbf{x}(n)\|^2}\mathbf{x}(n) = \frac{1}{\frac{1}{\mu} + \|\mathbf{x}(n)\|^2}e(n)\mathbf{x}(n) \tag{2.24}$$

In the NLMS (Normalised LMS) algorithm, $\Delta\boldsymbol{w}(n)$ expression is:

$$\Delta\boldsymbol{w}(n) = \boldsymbol{w}(n+1) - \boldsymbol{w}(n) = \frac{\beta}{\epsilon + \|\mathbf{x}(n)\|^2}e(n)\mathbf{x}(n) \tag{2.25}$$

By comparing (2.24) and (2.25), the relationship between β, ϵ and μ can be obtained. When $\beta = 1$ and $\epsilon = 1/\mu$, the update equation based on the a posteriori error is equivalent to the NLMS algorithm.

2.2.c GNGD algorithm

In addition to GASS, generalized normalized gradient descent (GNGD) is another adaptive learning rate strategy and adapts via a time varying regularization factor $\epsilon(n)$. Thus, GNGD can also be considered as

the adaptive version of regularized NLMS algorithm with higher robustness.

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \frac{\mu}{\epsilon(n) + \|\mathbf{x}(n)\|^2} e(n)\mathbf{x}(n) \tag{2.26}$$

$$\epsilon(n+1) = \epsilon(n) - \rho\mu \frac{e(n)e(n-1)\mathbf{x}^T(n)\mathbf{x}(n-1)}{(\epsilon(n-1) + \|\mathbf{x}(n-1)\|^2)^2} \tag{2.27}$$

In above two equations, ρ is set as 0.005 and $\epsilon(0)$ is set as $1/\mu$ in this task. In Figure 19, the performance of Benveniste GASS and GNGD are compared. If both the initial step size $\mu_{GASS}(1)$ of GASS and the fixed step size μ_{GNGD} in GNGD are set as a small value, GASS achieves a higher convergence rate and outperforms the GNGD. When $\mu_{GASS}(1) = \mu_{GNGD} = 0.1$, Benveniste GASS converges within 60 samples while GNGD converges after 900 samples. However, $\mu_{GASS}(n)$ should be within a limited range. If $\mu_{GASS}(1)$ is greater than 0.54, the GASS algorithm diverges. This instability problem can be effectively solved by the GNGD algorithm which supports a significantly wider range of μ_{GNGD}. A reasonably large value can be assigned to μ_{GNGD} to achieve fast convergence. When $\mu_{GNGD} = 2$, the GNGD algorithm converges faster than the GASS algorithm with $\mu_{GASS}(1) = 0.54$. The difference in stability between two algorithms can be attributed to the fact that the GASS utilises linear adaptive step size update while GNGD utilises non-linear regularisation factor update. Thus, GNGD is more robust than GASS.

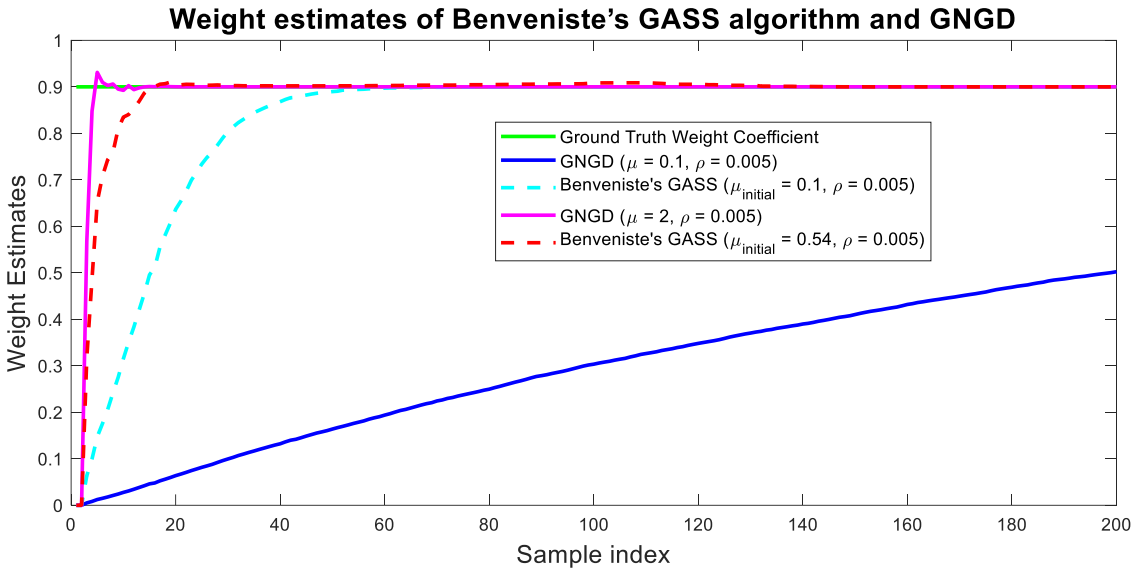

Figure 19. The weight estimation curves and weight error curves for GNGD and GASS algorithms

Table 8. Computational complexity analysis of GNGD algorithm

Step	Additions	Multiplications
$A = (\epsilon(n-1) + \|\mathbf{x}(n-1)\|^2)^2$	$N+1$	$N+1$
$B = \rho\mu e(n)e(n-1)\mathbf{x}^T(n)\mathbf{x}(n-1)$	$N-1$	$4+N$
$\epsilon(n+1) = \epsilon(n) - A/B$	1	1
$D = \epsilon(n) + \|\mathbf{x}(n)\|^2$	N	N
$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu e(n)\mathbf{x}(n)/D$	N	$N+2$
Total Number of operations	$8N+9$	
Overall Complexity	$O(N)$	

Table 9. Computational complexity analysis of GASS

Step	Additions	Multiplications
$e(n-1) = d(n) - \boldsymbol{w}^T(n)\mathbf{x}(n)$	N	N
$A = e(n-1)\mathbf{x}(n-1)$	0	N
$\boldsymbol{\psi}(n) = [\boldsymbol{I} - \mu(n-1)\mathbf{x}(n-1)\mathbf{x}^T(n-1)]\boldsymbol{\psi}(n-1) + A$	$2N^2$	$3N^2$
$\mu(n+1) = \mu(n) + \rho e(n)\mathbf{x}^T(n)\boldsymbol{\psi}(n)$	N	$N+1$
$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu(n)e(n)\mathbf{x}(n)$	N	$N+2$
Total Number of operations	$5N^2+7N+3$	
Overall Complexity	$O(N^2)$	

Furthermore, the Benveniste GASS has the complexity of $O(N^2)$ and GNGD has the complexity of $O(N)$. Low complexity means low requirement on hardware and less program operation time.

2.3 Adaptive Noise Cancellation

2.3.a Minimum value for the delay

The mean square error (MSE) between noise corrupted signal $s(n)$ and enhanced signal $\hat{x}(n)$ is:

$$\mathbb{E}\left\{\left(s(n)-\hat{x}(n)\right)^2\right\} = \mathbb{E}\left\{\left(x(n)+\eta(n)-\hat{x}(n)\right)^2\right\} \tag{2.28}$$

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

$$= \mathbb{E}\left\{\left(x(n)-\hat{x}(n)\right)^2\right\} + 2\mathbb{E}\{x(n)\eta(n)\} - 2\mathbb{E}\{\hat{x}(n)\eta(n)\} + \mathbb{E}\{\eta^2(n)\} \tag{2.29}$$

In ideal cases, the first term in (2.29) is exactly zero and is irrelevant to the noise. The fourth term is a fixed constant and determined by the statistical property of the coloured noise $\eta(n)$. It can be assumed that the clean signal $x(n)$ and zero-mean noise $\eta(n)$ are independent. Thus, the second term is also zero.

$$2\mathbb{E}\{x(n)\eta(n)\} = 2\mathbb{E}\{x(n)\}\mathbb{E}\{\eta(n)\} = 0 \tag{2.30}$$

Now, the minimization of MSE equation (2.28) can be simplified to the minimization of (2.32).

$$\hat{x}(n) = \boldsymbol{w}^T(n)\boldsymbol{u}(n) = \sum_{i=0}^{M-1} w_i(n)u_i(n) = \sum_{i=0}^{M-1} w_i(n)\left(x(n-\Delta-i)+\eta(n-\Delta-i)\right) \tag{2.31}$$

$$-2\mathbb{E}\{\hat{x}(n)\eta(n)\} = -2\mathbb{E}\left\{\sum_{i=0}^{M-1} w_i(n)\left(x(n-\Delta-i)+\eta(n-\Delta-i)\right)\eta(n)\right\}$$

$$= -2\sum_{i=0}^{M-1} w_i(n)\left(\mathbb{E}\{x(n-\Delta-i)\eta(n)\} + \mathbb{E}\{\eta(n-\Delta-i)\eta(n)\}\right) = -2\sum_{i=0}^{M-1} w_i(n)\mathbb{E}\{\eta(n-\Delta-i)\eta(n)\}$$

$$= -2\sum_{i=0}^{M-1} w_i(n)\mathbb{E}\left\{\left(v(n-\Delta-i)-0.5\times v(n-\Delta-i-2)\right)\left(v(n)-0.5\times v(n-2)\right)\right\} \tag{2.32}$$

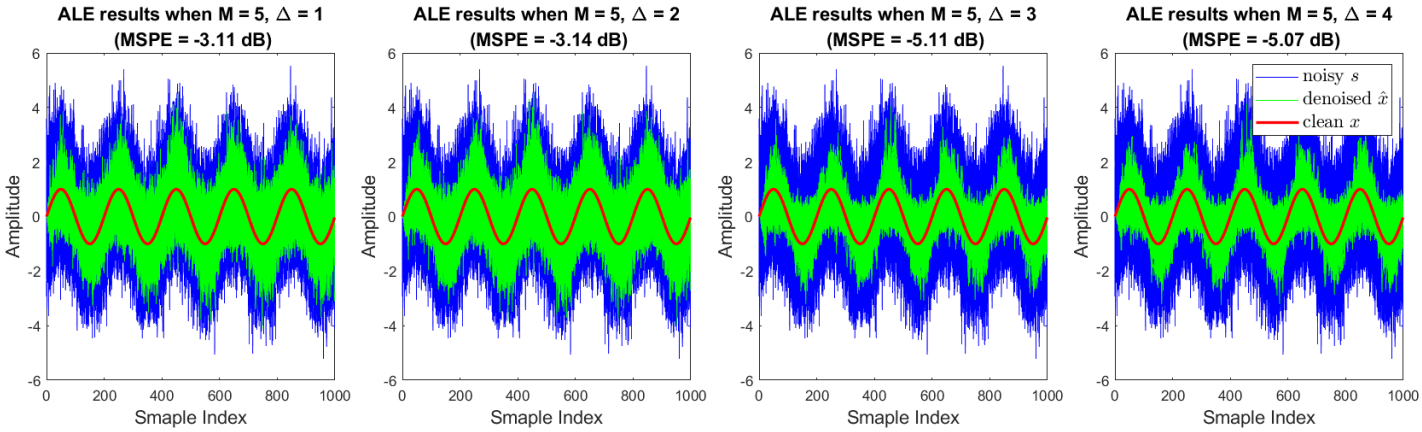

Figure 20. Performance comparison between ALE with different delays (step size $\mu = 0.01$ in LMS)

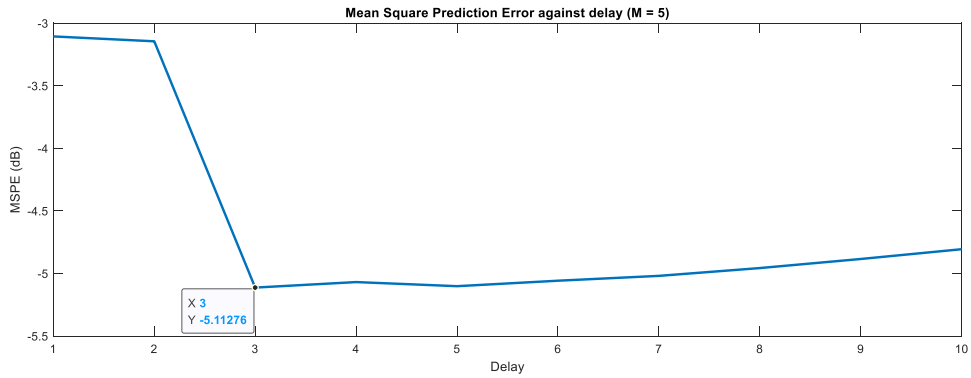

Figure 21. Relationship between MSPE and delay (step size $\mu = 0.01$ in LMS)

Since $v(n)$ is the white Gaussian noise with unit variance, $\mathbb{E}\{v(n)v(n-k)\} = 0, k \neq 0$. To ensure equation (2.32) to be zero, the delay Δ should be greater than 2 due to the $\mathbb{E}\{v(n-\Delta-i)v(n-2)\},\ i \in [0, M-1]$

term. Thus, the minimum delay should be 3 ($\Delta_{min}= 3$) to minimise MSE equation (2.28). In Figure 20, four delays are tested. $\Delta> 2$ can result in large MSPE reduction and the minimum MSPE is achieved by $\Delta= 3$, which matches the prediction.

2.3.b The effect of delay and filter order

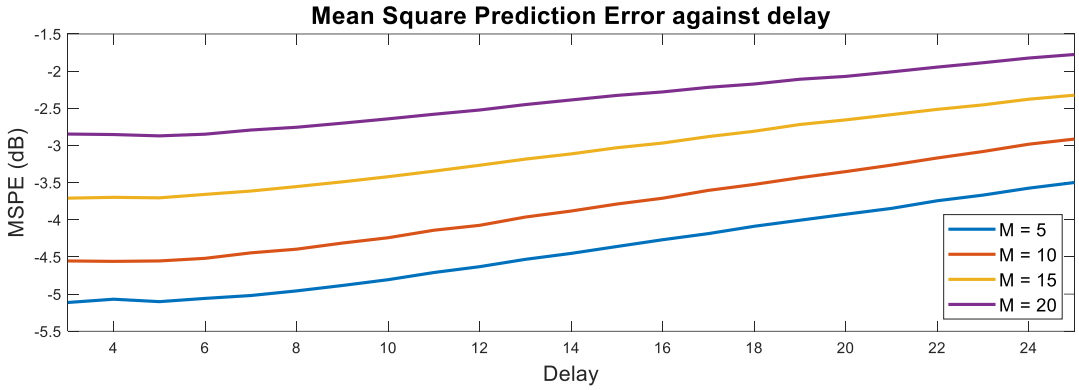

Figure 22. The dependence between delay and the MSPE (step size $\mu = 0.01$ in LMS)

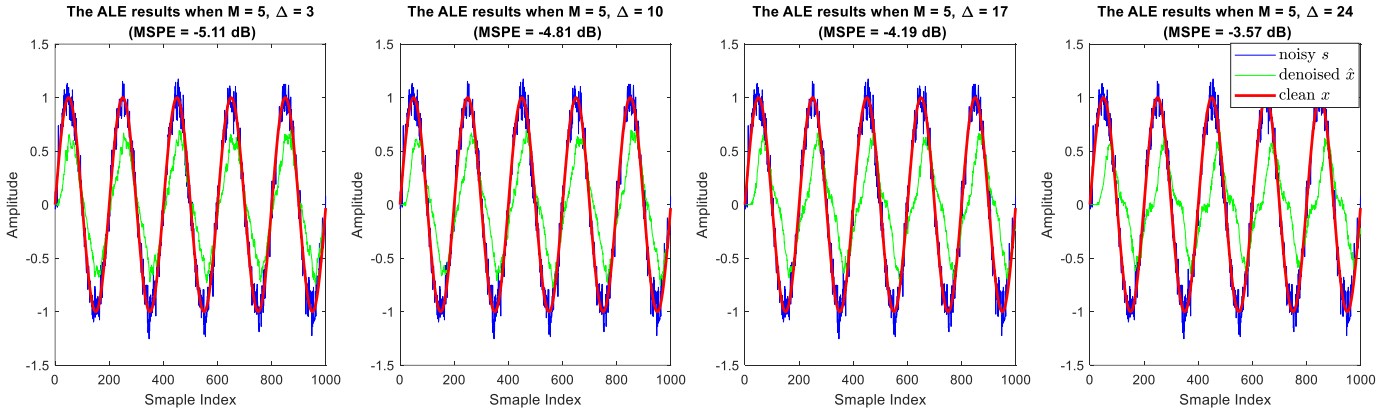

Figure 23. The mean denoised signal and noisy signal averaged over 100 independent trials

As shown in Figure 22, when the delay Δ is within the range of $[3, 6]$, the variation of MSPE caused by different delays is slight. Since these delays are greater than or equal to the low threshold value (i.e., 3), the uncorrelation between noise $\eta(n)$ and delayed noise $\eta(n-\Delta)$ can be ensured and the slight difference in phase shift introduced by the ALE system is ignorable due to the strength auto-correlation property of sinusoidal signal. However, when the delay is greater than 6, the slope of the MSPE curve becomes large for all four filter orders (M). A small increase in delay results in relatively large growth of MSPE. The excessively large delay causes considerable phase shift in denoised $\hat{x}(t)$ and large deviation from the clean signal $x(t)$. In last section, the first term in (2.29) is ignored and its main effect is displayed in Figure 23. Thus, to guarantee the high performance (i.e., small MSPE) of ALE system, the delay Δ has an upper limit (e.g., 6 in this case).

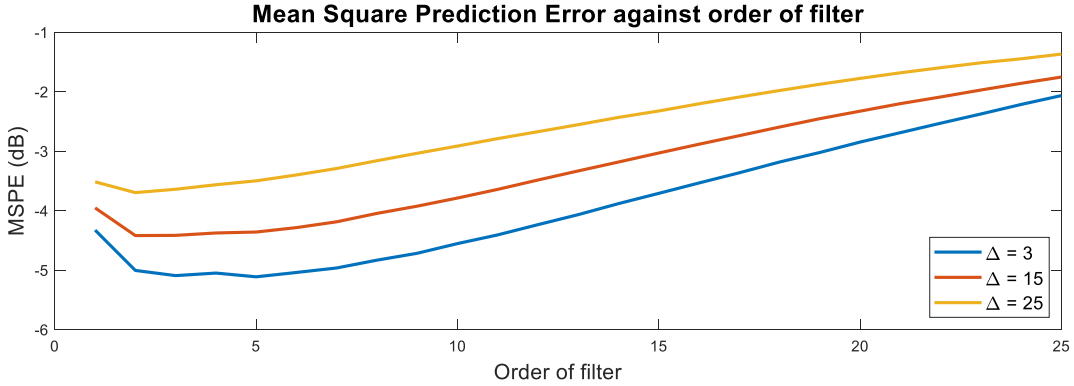

Figure 24. The effect of the filter order M on the MSPE (step size $\mu = 0.01$ in LMS)

In Figure 24, as the order of filter (M) increases from 1 to 25, the MSPE drops at first and then rises to a large value. When the minimum delay $\Delta_{min}= 3$ is used, $M = 5$ provides the global minimum MSPE. The initial

large MSPE values can be attributed to the under-fitting problem because the low-order filter does not have enough degrees of freedom to precisely track the true signal $x(t)$. By comparison, the following MSPE rise is caused by the over-fitting problem. Some noise components are wrongly modelled as the true signal components. Besides, the computational cost also increases with M. To achieve the trade-off between small MSPE and low complexity, $M = 5$ may be a good pragmatic choice. Furthermore, $\Delta = 3$ is the optimal delay.

2.3.c Adaptive noise cancellation configuration

Unlike the ALE system, the adaptive noise cancellation (ANC) system uses the secondary noise $\epsilon(n)$ as the reference input and noise corrupted signal $s(n) = x(n) + \eta(n)$ as the teaching signal to estimate the primary noise $\eta(n)$. The $\epsilon(n)$ is correlated with $\eta(n)$, and equation (2.33) is applied to simulate $\epsilon(n)$.

$$\epsilon(n) = 0.8 \times \eta(n) + 0.1 \tag{2.33}$$

According to Figure 25, the ALE system can reach the steady-state within dozens of samples but tends to have a large MSPE (-5.11 dB). The ANC system has poor performance in the initial stage and requires more than 200 samples to correctly model the pattern of colored noise but can result in small MSPE (-14.2 dB) in the steady-state. Thus, if adequate samples are available, ANC outperforms ALE and can effectively remove the noise from sinusoids. However, the ALE is a good choice when the number of samples is small, or the secondary noise signals cannot be obtained. Additionally, although the ANC avoids the selection of delay, it has high requirements on the correlation between $\eta(n)$ and $\epsilon(n)$.

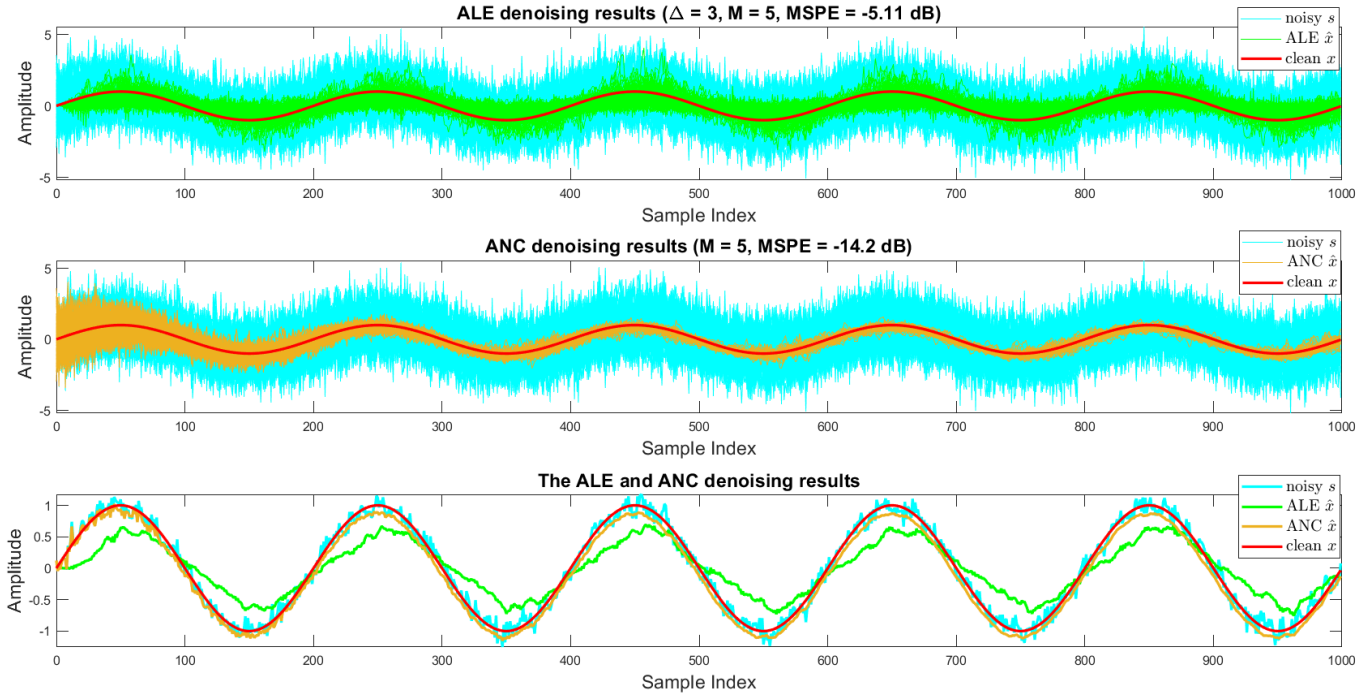

Figure 25. The ALE and ANC denoising results obtained in 100 trials and their averaged results when the filter order is set as 5, the step size is set as 0.01, and the delay in ALE is set as 3.

2.3.d Apply the ANC method to EEG data

The ANC system can effectively suppress the undesired component in the signal when the noisy sinusoid at a target frequency is created and used as the synthetic reference input to the system. In the spectrogram of noisy data (Figure 26), there is a strong 50 Hz component. Thus, the sinusoid at 50 Hz frequency with no phase shift and unit amplitude and the white Gaussian noise with 0.001 variance are combined and set as the reference which is correlated with this undesired component.

The denoising performance of ANC can be affected by the selected filter order M and step size μ in the LMS algorithm. In this task, different combinations were tried to find the optimal choice, and 12 representative results are displayed in Figure 27. For a fixed μ, excessively small M causes the under-fitting problem and the undesired component cannot be effectively removed. However, the large M may cause the over-fitting problem and parts of useful signals at neighbouring frequencies may also be wrongly removed. Theoretically, the LMS algorithm with smaller μ tends to achieve smaller steady-state error although it requires more samples to reach convergence. However, the number of EEG data is limited. If μ is too small (e.g., $\mu =$

0.0001), the LMS algorithm may stop before convergence or only noise in the last few data can be removed. However, LMS with large μ achieves fast convergence at the cost of large steady-state error. When μ rises to 0.01, many spectral components are suppressed together with the target noise, which reduces the denoised signal quality.

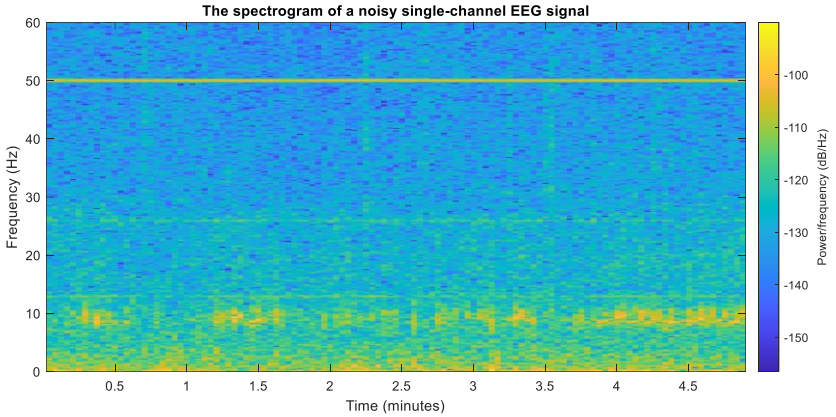

Figure 26. The spectrogram of noise corrupted EEG signal (Hamming window length = 6000 samples (5 s), the overlap ratio = 50%, 8192 sampling points are used to calculate DFT)

Thus, the medium M and μ should be selected to achieve undesired component removal and neighbouring frequency components preservation at the same time. After several trials, $M = 8$ and $\mu = 0.001$ enable the ANC system to achieve reasonably high performance and robust results.

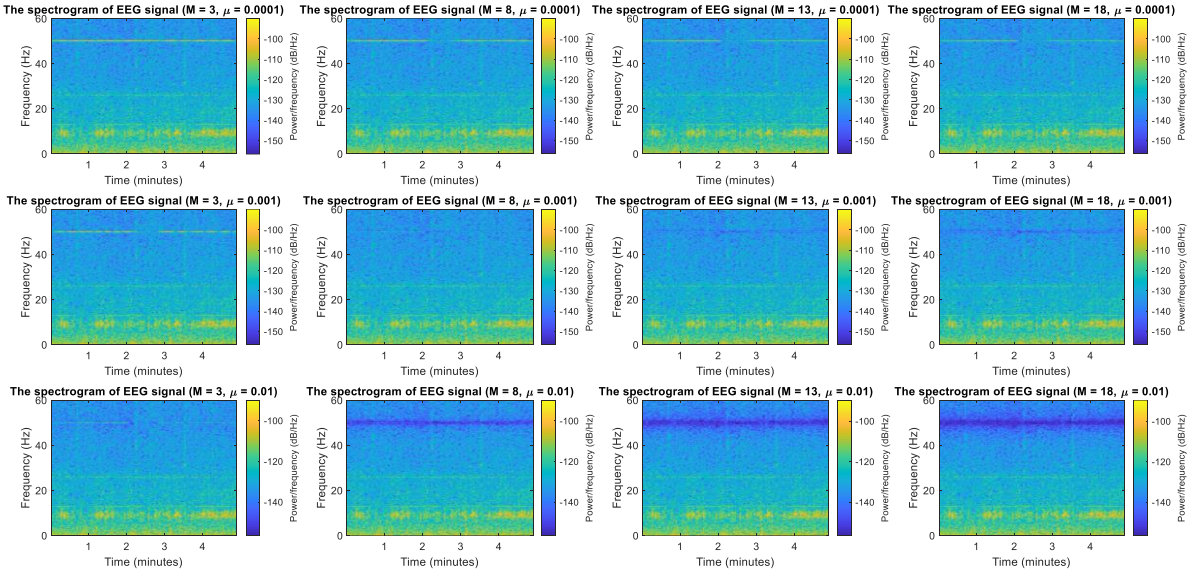

Figure 27. The spectrogram of the denoised EEG signal

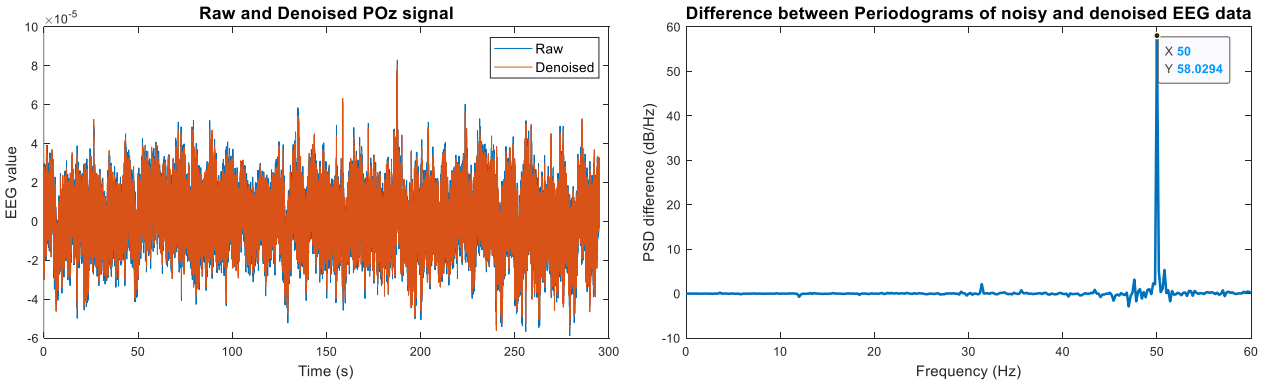

Figure 28. The raw and denoised EEG data, the difference between their standard periodograms

Furthermore, the raw and denoised EEG data and their periodogram difference diagram are displayed in Figure 28. Generally, both raw and denoised data have high similarity and their periodograms overlap at most frequencies except for some frequencies around 50 Hz. The spike at 50 Hz in the right diagram demonstrates the effectiveness of this ANC system with $M = 8$ and $\mu = 0.001$.

3 Widely Linear Filtering and Adaptive Spectrum Estimation

3.1 Complex LMS and Widely Linear Modelling

3.1.a ACLMS and CLMS

In the left diagram of Figure 29, the large circularity coefficient 0.855 of WLMA(1) process means the large non-circularity. However, the complex LMS (CLMS) is the generic strictly linear extension of real-valued estimators and is only applicable for circular random variables. CLMS may fail to model the WLMA(1). By comparison, the augmented CLMS (ACLMS) is a widely linear model. It utilises an augmented input vector to increases its degrees of freedom and can fully identify the second-order statistical features of the data. Thus, high performance (i.e., low steady-state error) of ACLMS can be expected.

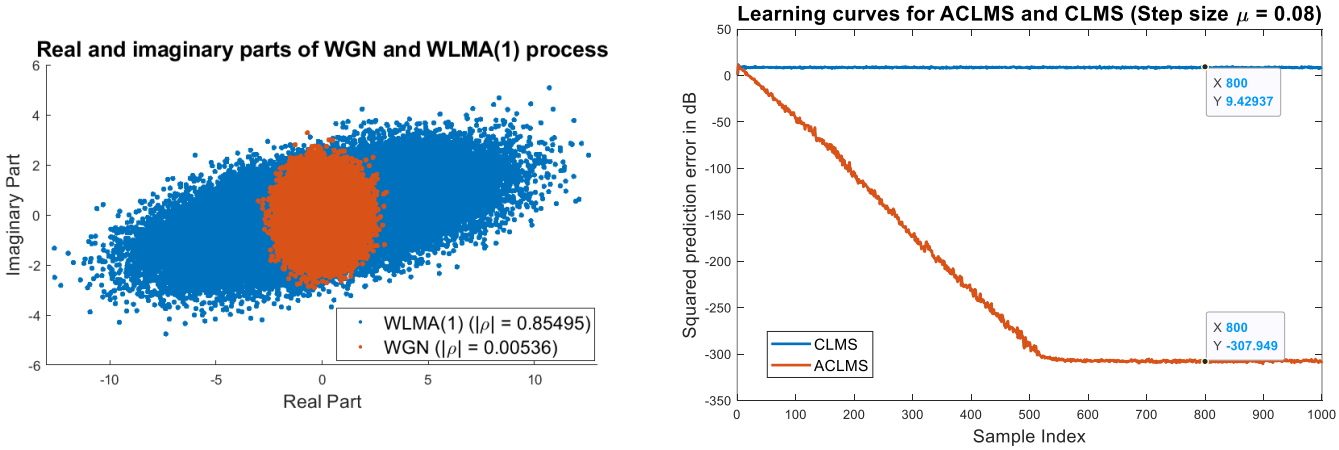

Figure 29. The circularity diagram of WGN process and WLMA(1) process; Ensemble average of the learning curve from 100 simulations for ACLMS and CLMS

The mean squared prediction error curves plotted in the right diagram of Figure 29 for both ACLMS and CLMS match the prediction. The steady state error of CLMS algorithm reaches $9.42\ \mathrm{dB}$ in the 800th sample. This error is excessively large and the identified weight coefficients have unacceptably large difference with their ground truth values. The CLMS only has one degree of freedom but the WLMA(1) has two weights that requires at least two degrees of freedom. This problem is solved by ACLMS. It considers both input $\boldsymbol{x}(n)$ and its conjugate $\boldsymbol{x}^*(n)$ to update two weight vectors ($\boldsymbol{h}(n)$ and $\boldsymbol{g}(n)$) which enables ACLMS to have one additional degrees of freedom to effectively identify the WLMA(1) model. When the step size is 0.08, ACLMS can converge after 540 samples and its steady state error is smaller than $-300\ \mathrm{dB}$. According to this extremely small error, the ACLMS has the large advantage on general complex data modeling over CLMS.

3.1.b Bivariate wind data

The circularity diagrams of wind speed data measured from three regimes are shown in Figure 30. The shapes formed by the medium and high wind speed data can be roughly considered as two ellipses which indicate relatively high non-circularity. The low wind speed data does form a regular circle, but its distribution still has high rotation invariance (high circularity). To quantitatively measure the degree of non-circularity of these data, their corresponding circularity coefficients (i.e., $|\rho|_{low} = 0.159$, $|\rho|_{medium} = 0.454$ and $|\rho|_{high} = 0.624$) are computed. As the dynamics increase, the circularity decreases, and wind speed data become more non-circular. Thus, the CLMS is applicable for low wind speed data, but the ACLMS is required for the data measured from medium and high dynamics regimes.

According to Figure 31, the MSPE errors of both algorithms tend to drop at first and then rise as the filter order M increases. The initial large error is due to the under-fitting problem caused by the lack of degrees of freedom. When M becomes large, the over-fitting problem also causes large MSPE. For ACLMS algorithm, the optimal value of M is 4 for three wind datasets. For CLMS algorithm, the optimal M is 7 for low-speed data and 5 for medium and high-speed data. By comparing the minimum values of both error curves for each data set, it can be concluded that ACLMS algorithm outperforms the CLMS algorithm for all three wind

regimes. Besides, the extra degree of freedom enables the ACLMS to overcome the under-fitting problem but worsens the over-fitting problem, which explains why the optimal M_{ACLMS} for ACLMS is smaller than optimal M_{CLMS} for CLMS. When M is smaller than a threshold (e.g., 10 for low-speed data, 4 for medium speed data, and 16 for high-speed data), ACLMS errors are always smaller than CLMS errors. As the growth of regime dynamics, the non-circularity of data increases and the advantage of ACLMS over CLMS is highlighted. Furthermore, the learning rate μ also requires careful adjustment. The large step can be applied to the data with small circularity, while these data with high circularity require small steps. Figure 32 displays the CLMS and ACLMS prediction results with optimal parameters. All predicted data follow the distribution pattern of the true data, but the advantage of ACLMS cannot be visually observed from Figure 32.

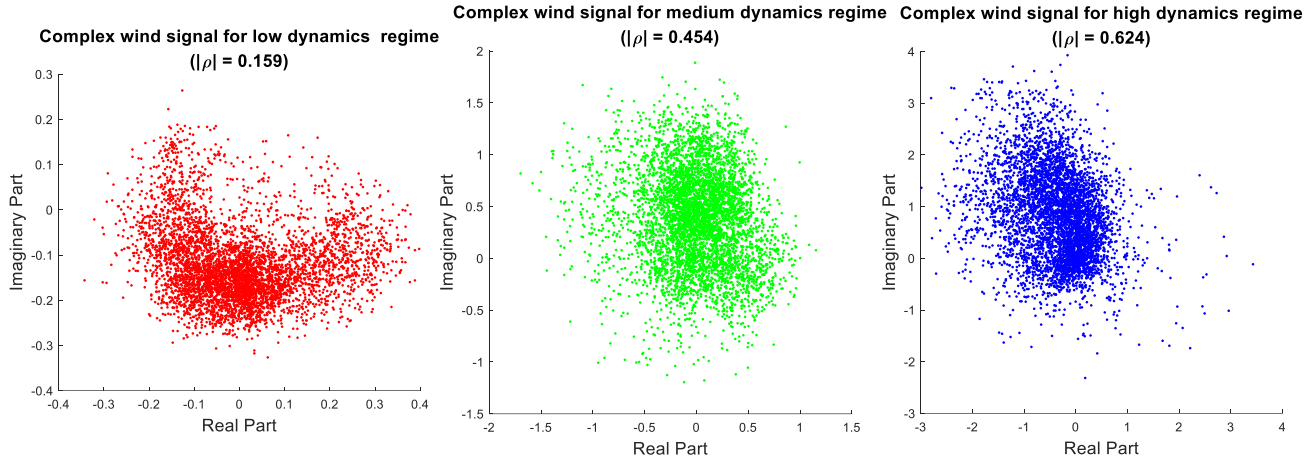

Figure 30. Circularity diagrams for wind data measured in low, medium and high dynamics regions

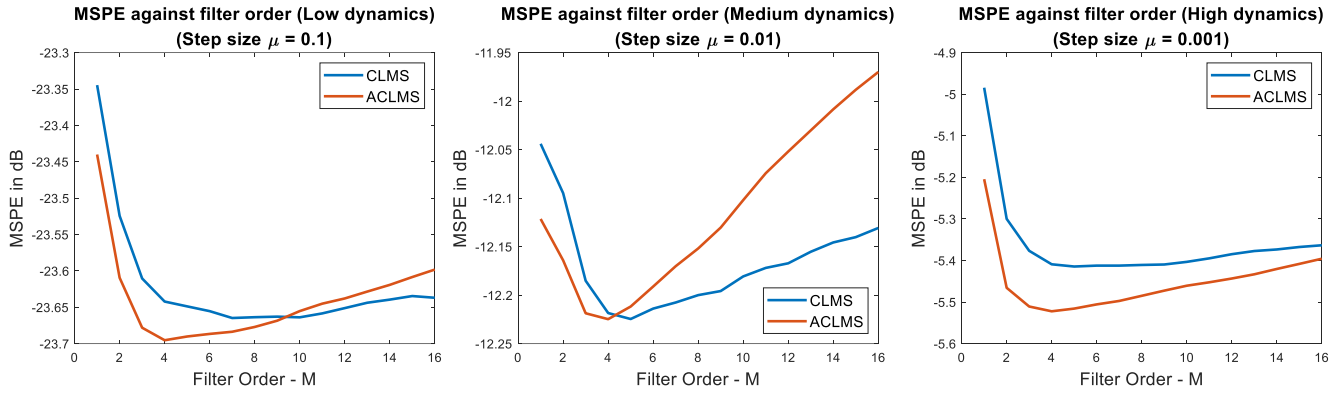

Figure 31. The relationship between filter order M and performance of CLMS and ACLMS

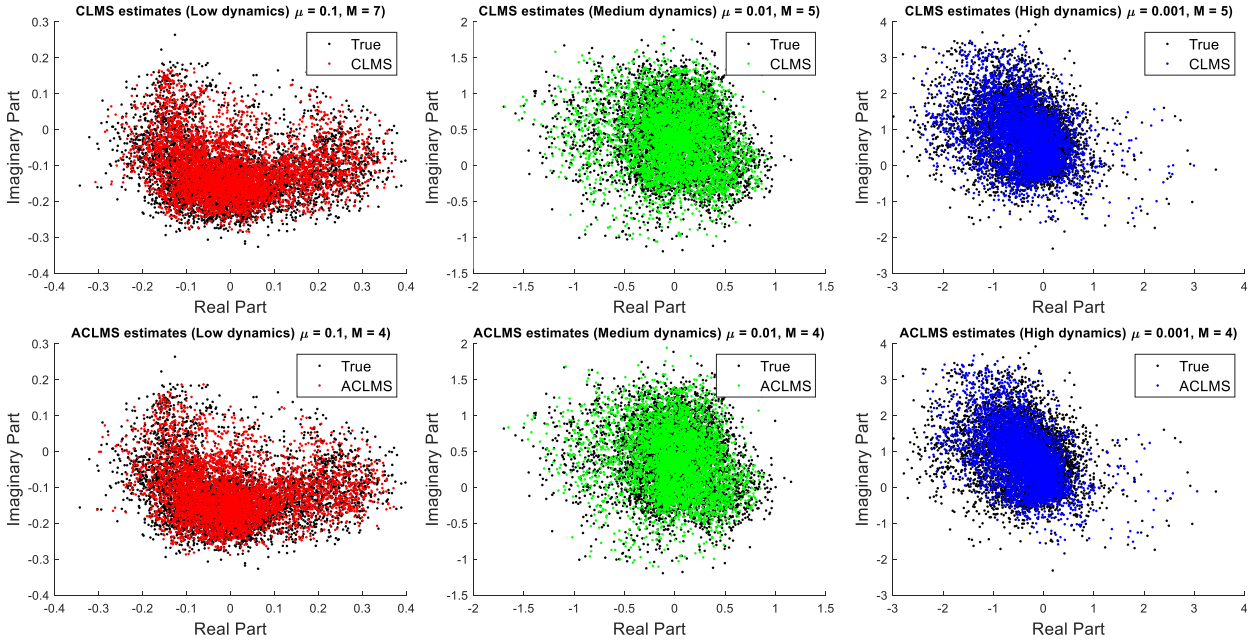

Figure 32. Circularity diagrams for true and estimated wind data (optimal one-step ahead prediction)

3.1.c Complex balanced and unbalanced system voltages

In the balanced system, the complex $\alpha - \beta$ voltage results in a circle in circularity diagram and its circularity constant $|\rho|$ is exactly zero, which means circular data. When the magnitude and / or phase of one or more phase voltages change, the system becomes unbalanced and generates an ellipse in circularity diagram with non-zero $|\rho|$. $|\rho|$ is proportional the eccentricity of this ellipse. Large eccentricity means large deviation from a circle and large distribution variation under rotation. Therefore, if the shape of the circularity diagram is not a perfect circle, the system would be unbalanced and contains some faults. The eccentricity can be used to measure the severity of these faults (e.g., the magnitude and / or phase distortion of certain phases).

In this task, $V_a(n) = 1, V_b(n) = 1.4$ and $V_c(n) = 0.6$ as well as $\Delta_b = \pi/3$ and $\Delta_c = \pi/4$ are used to simulate the unbalanced system complex voltages.

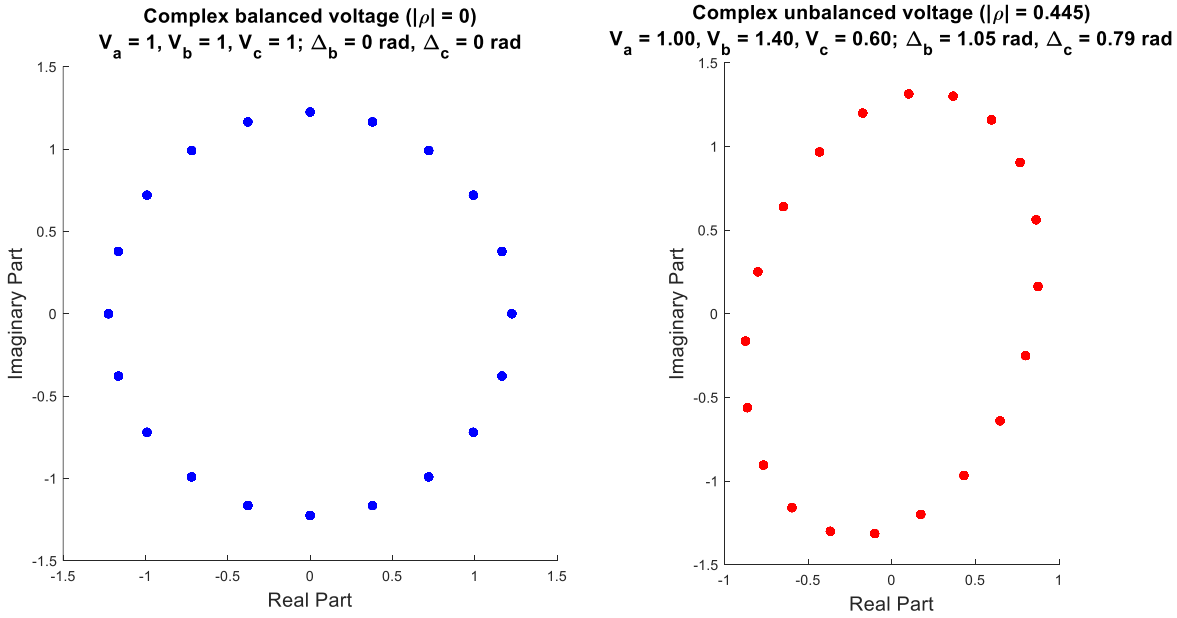

Figure 33. Circularity diagrams of balanced and unbalanced voltages ($f_o = 50$ Hz, $f_s = 1000$ Hz)

3.1.d Frequency estimate of the CLMS and ACLMS

Under balanced conditions, the complex voltage can be expressed as a complex exponent.

$$v(n) = \sqrt{\frac{3}{2}} V e^{j2\pi \frac{f_o}{f_s} n + \phi} \tag{3.1}$$

$$v(n+1) = \sqrt{\frac{3}{2}} V e^{j2\pi \frac{f_o}{f_s} n + \phi} e^{j2\pi \frac{f_o}{f_s}} = e^{j2\pi \frac{f_o}{f_s}} v(n) \tag{3.2}$$

In the strictly linear autoregressive models of order 1, the $v(n+1)$ is:

$$v(n+1) = h^*(n) v(n) \tag{3.3}$$

By comparing the equation (3.2) and (3.3), the $h(n)$ can be expressed as:

$$h^*(n) = e^{j2\pi \frac{f_o}{f_s}} \rightarrow h(n) = e^{-j2\pi \frac{f_o}{f_s}} = \cos\left(-2\pi \frac{f_o}{f_s}\right) + j \sin\left(-2\pi \frac{f_o}{f_s}\right) \tag{3.4}$$

$$\Re\big(h(n)\big) = \cos\left(-2\pi \frac{f_o}{f_s}\right), \qquad \Im\big(h(n)\big) = \sin\left(-2\pi \frac{f_o}{f_s}\right) \tag{3.5}$$

$$\frac{\Im\big(h(n)\big)}{\Re\big(h(n)\big)} = \tan\left(-2\pi \frac{f_o}{f_s}\right) \tag{3.6}$$

Thus, the frequency f_o of the complex balanced voltage can be computed by using estimated $h(n)$.

$$f_o = -\frac{f_s}{2\pi} \arctan\left(\frac{\Im\big(h(n)\big)}{\Re\big(h(n)\big)}\right) \tag{3.7}$$

The f_o should be a real value. The complex magnitude of $\frac{f_s}{2\pi}\arctan\left(\frac{\Im(h(n))}{\Re(h(n))}\right)$ should be used as f_o. The negative sign just changes the phase of f_o and does not affect the results.

$$f_o = \frac{f_s}{2\pi}\arctan\left(\frac{\Im\big(h(n)\big)}{\Re\big(h(n)\big)}\right) \tag{3.8}$$

According to Clarke Transform, the general complex voltage is the sum of two complex exponents.

$$v(n) = A(n)e^{j\left(2\pi\frac{f_o}{f_s}n+\phi\right)} + B(n)e^{-j\left(2\pi\frac{f_o}{f_s}n+\phi\right)} \tag{3.9}$$

$$v(n+1) = A(n+1)e^{j\left(2\pi\frac{f_o}{f_s}n+\phi\right)}e^{j\left(2\pi\frac{f_o}{f_s}\right)} + B(n+1)e^{-j\left(2\pi\frac{f_o}{f_s}n+\phi\right)}e^{-j\left(2\pi\frac{f_o}{f_s}\right)} \tag{3.10}$$

In widely linear autoregressive models of order 1, the $v(n+1)$ is:

$$v(n+1) = h^*(n)v(n) + g^*(n)v^*(n) \tag{3.11}$$

$$v(n+1) = \big(h^*(n)A(n) + g^*(n)B^*(n)\big)e^{j\left(2\pi\frac{f_o}{f_s}n+\phi\right)} + \big(g^*(n)A^*(n) + h^*(n)B(n)\big)e^{-j\left(2\pi\frac{f_o}{f_s}n+\phi\right)} \tag{3.12}$$

By comparing the equation (3.11) and (3.12), the following two equations can be obtained.

$$h^*(n)A(n) + g^*(n)B^*(n) = A(n+1)e^{j\left(2\pi\frac{f_o}{f_s}\right)} \tag{3.13}$$

$$g^*(n)A^*(n) + h^*(n)B(n) = B(n+1)e^{-j\left(2\pi\frac{f_o}{f_s}\right)} \tag{3.14}$$

$$A(n) = \sqrt{\frac{1}{6}}\big(V_a(n) + V_b(n)e^{j\Delta_b} + V_c(n)e^{j\Delta_c}\big), \tag{3.15}$$

$$B(n) = \sqrt{\frac{1}{6}}\left(V_a(n) + V_b(n)e^{-j\left(\Delta_b+\frac{2\pi}{3}\right)} + V_c(n)e^{-j\left(\Delta_c-\frac{2\pi}{3}\right)}\right) \tag{3.16}$$

In equation (3.15) and (3.16), the $V_a(n)$, $V_b(n)$ and $V_c(n)$ are the peak values of three phase voltages. If they are assumed as three constant values or their difference between two adjacent sample indexes is ignorable due to the large sampling frequency, $A(n) \approx A(n+1)$ and $B(n) \approx B(n+1)$. Equation (3.13) and (3.14) now can be simplified as follows:

$$e^{j\left(2\pi\frac{f_o}{f_s}\right)} = h^*(n) + \frac{g^*(n)B^*(n)}{A(n)}, \qquad e^{-j\left(2\pi\frac{f_o}{f_s}\right)} = h^*(n) + \frac{g^*(n)A^*(n)}{B(n)} \tag{3.17}$$

$$h^*(n) + \frac{g^*(n)B^*(n)}{A(n)} = h(n) + \frac{g(n)A(n)}{B^*(n)} \tag{3.18}$$

$$g^*(n)\left(\frac{B^*(n)}{A(n)}\right)^2 + \big(h^*(n) - h(n)\big)\left(\frac{B^*(n)}{A(n)}\right) - g(n) = 0 \tag{3.19}$$

Equation (3.19) can be treated as the quadratic equation with one unknown $B^*(n)/A(n)$.

$$\frac{B^*(n)}{A(n)} = \frac{h(n) - h^*(n) \pm \sqrt{\big(h^*(n) - h(n)\big)^2 + 4g^*(n)g(n)}}{2g^*(n)} \tag{3.20}$$

$$\frac{B^*(n)}{A(n)} = j\frac{\Im\big(h(n)\big) \pm \sqrt{\Im^2\big(h(n)\big) - |g(n)|^2}}{g^*(n)}$$

$$e^{j\left(2\pi\frac{f_o}{f_s}\right)} = \Re\big(h(n)\big) \pm j\sqrt{\Im^2\big(h(n)\big) - |g(n)|^2} = \cos\left(2\pi\frac{f_o}{f_s}\right) + j\sin\left(2\pi\frac{f_o}{f_s}\right) \tag{3.21}$$

Since $f_s \geq 2f_o \geq 0, \quad 0 < \frac{2\pi f_o}{f_s} < \pi, \quad \sin\left(2\pi\frac{f_o}{f_s}\right) > 0$. Thus, the $\sin\left(2\pi\frac{f_o}{f_s}\right) \neq -\sqrt{\Im^2\big(h(n)\big) - |g(n)|^2}$.

$$\cos\left(2\pi\frac{f_o}{f_s}\right)=\Re\big(h(n)\big),\sin\left(2\pi\frac{f_o}{f_s}\right)=\sqrt{\Im^2\big(h(n)\big)-|g(n)|^2} \tag{3.22}$$

$$2\pi\frac{f_o}{f_s}=\arctan\left(\frac{\sqrt{\Im^2\big(h(n)\big)-|g(n)|^2}}{\Re\big(h(n)\big)}\right)\rightarrow f_o=\frac{f_s}{2\pi}\arctan\left(\frac{\sqrt{\Im^2\big(h(n)\big)-|g(n)|^2}}{\Re\big(h(n)\big)}\right) \tag{3.23}$$

3.1.e Frequency estimates of the $\alpha-\beta$ voltages

For balanced system voltages, both CLMS and ACLMS can provide correct frequency estimates after a few iterations. Although both algorithms have similar learning curves, the CLMS outperforms ACLMS because its frequency estimate curve avoids the large fluctuation at the beginning and only requires two iterations to obtain the correct frequency estimate. Since the filter order is one and the sample data number is small initially, the over-fitting problem caused by excessive freedom degrees becomes obvious and results in a relatively large error in weight coefficient estimates. According to equation (3.23), the large f_s (i.e., $1000\ \mathrm{Hz}$) significantly amplifies such coefficient error and causes the large overshoot of frequency estimates. After sufficient training data is obtained, this over-fitting problem can be solved and the overshoot drops to zero.

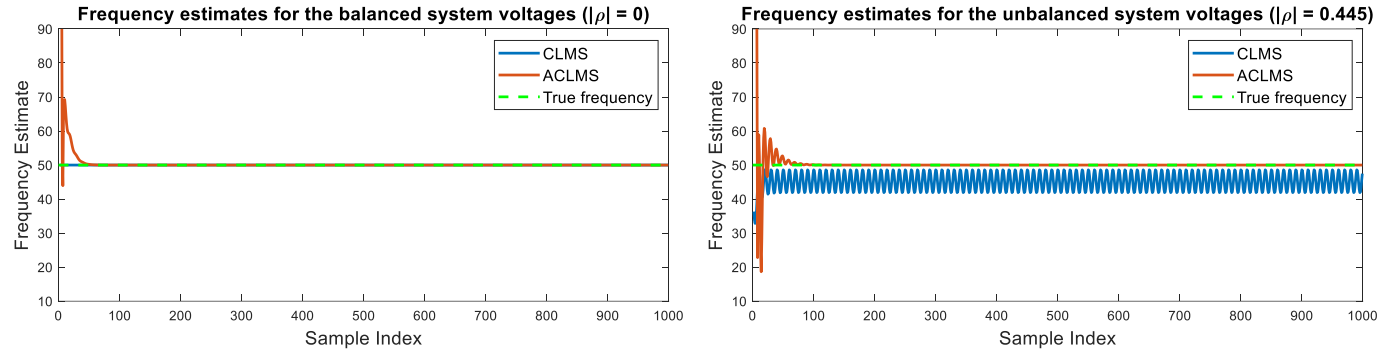

Figure 34. The frequency estimates of $\alpha-\beta$ voltages under balanced and unbalanced conditions

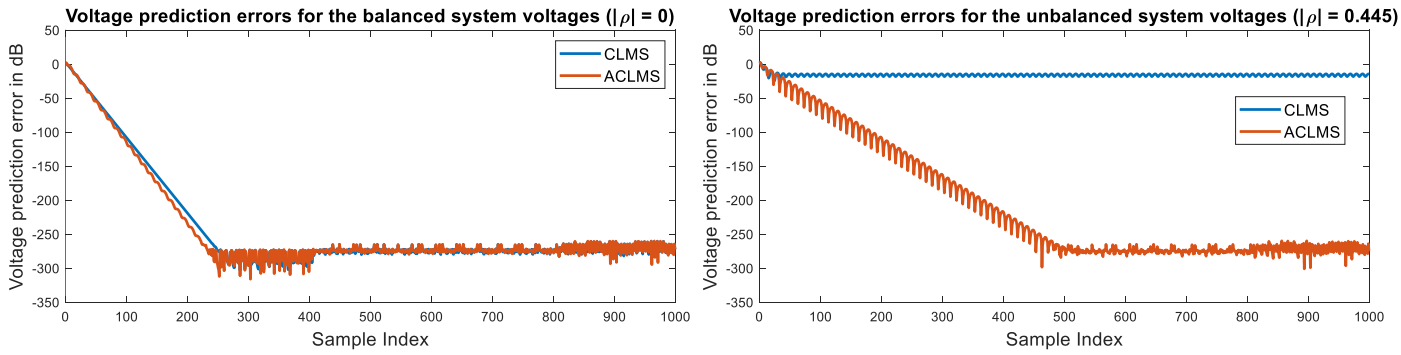

Figure 35. The learning curves of CLMS and ACLMS under balanced and unbalanced conditions ($\mu=0.08$)

For the unbalance system voltages, the CLMS fails to provide the correct frequency estimates. CLMS can converge within 40 iterations in this trial but results in large bias and variance errors. In the steady state, the frequency estimates are not constant but vary like a sinusoidal wave. The large CLMS voltage prediction errors shown in Figure 35 explain its low frequency estimation accuracy. As the degrees of phase and amplitude distortions increase, the circularity constant $|\rho|$ rises and the errors of CLMS rise. By comparison, the ACLMS requires 100 iterations to reach the steady state but can obtain accurate frequency estimation with ignorable bias and variance errors. Its convergence rate is still fairly high. Its augmented weight coefficient $g(n)$ provides additional degrees of freedom to fit the non-circular complex data. In Figure 35, in the first 500 iterations, the learning curve of ACLMS contains some fluctuations, but its overall trend is downward. After 500 iterations, the error becomes $-300\ \mathrm{dB}$ and the remaining fluctuations are negligible. Since both weight coefficients are correctly identified, the frequency estimates have high accuracy.

Generally, this task proves that CLMS is particularly suitable for a balanced system where complex voltages

are circular due to its advantage of fast convergence. Suppose the circularity of a general complex data is unknown in advance. In that case, ACLMS may be the optimal choice since it can successfully identify the model of both circular and non-circular complex data.

3.2 Adaptive AR Model Based Time-Frequency Estimation

3.2.a AR model based spectrum estimate

In this task, the sampling frequency is set as $1500\,\mathrm{Hz}$. The frequency $f(n)$ and phase $\phi(n)$ of the frequency-modulated (FM) signal are displayed in Figure 36. The $f(n)$ can be considered as a piece-wise function that consists of a constant function, a linear function and a quadratic function. Due to the varying frequency, the complete FM signal is non-stationary. The Yule-Walker method is employed to estimate the AR model parameter, and the model order is assumed to be 1. However, this power spectrum estimation method only supports stationary signals and performs poorly for non-stationary signals. In Figure 37, there is only one single peak at the approximately average frequency ($184\,\mathrm{Hz}$) of the complete signal in the spectrum when AR(1) model is used. No more details of $f(n)$ can be obtained. Thus, this method fails to capture the changes in frequency. This failure can be attributed to the fact that the block-based AR coefficient estimation cannot get accurate results when the signal is non-stationary.

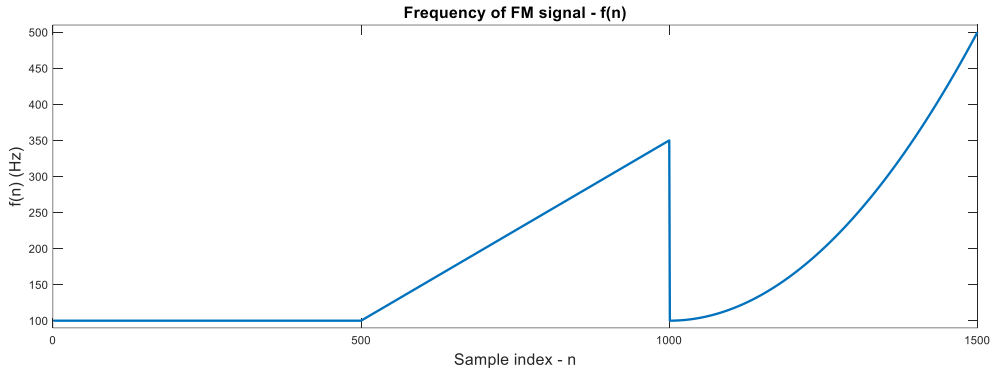

Figure 36. The frequency of FM signal

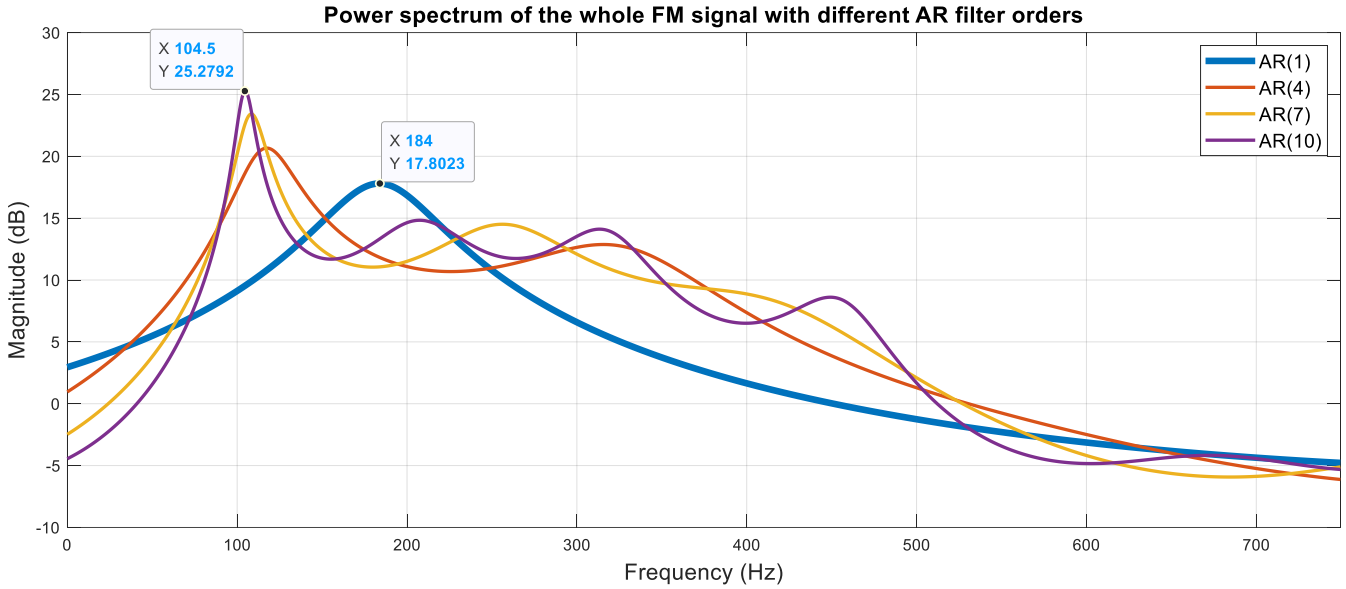

Figure 37. The power spectrum of the complete signal when different orders of AR model are used

As the order of the AR model increases, more sharp spectral peaks can be identified. For example, when the order rises up to 10, the peak around $100\,\mathrm{Hz}$ which corresponds to the frequency of the initial part of the signal appears. However, due to the lack of dimension, the following frequency variation with respect to time still cannot be revealed. To capture the variation details, the complete signal can be uniformly divided into several segments and each segment's spectrum can be estimated individually. If the length of a segment is sufficiently small, it can be roughly considered as a stationary signal with constant frequency. In this case,

the block-based AR coefficient estimation becomes applicable and one single sharp spectral peak can be expected. The frequencies corresponding to the highest peaks in all spectrums can be collected together to form a trajectory and model the frequency variation pattern with relatively high accuracy. But this method may have considerably high computational cost and low frequency resolution.

3.2.b CLMS based spectrum estimate

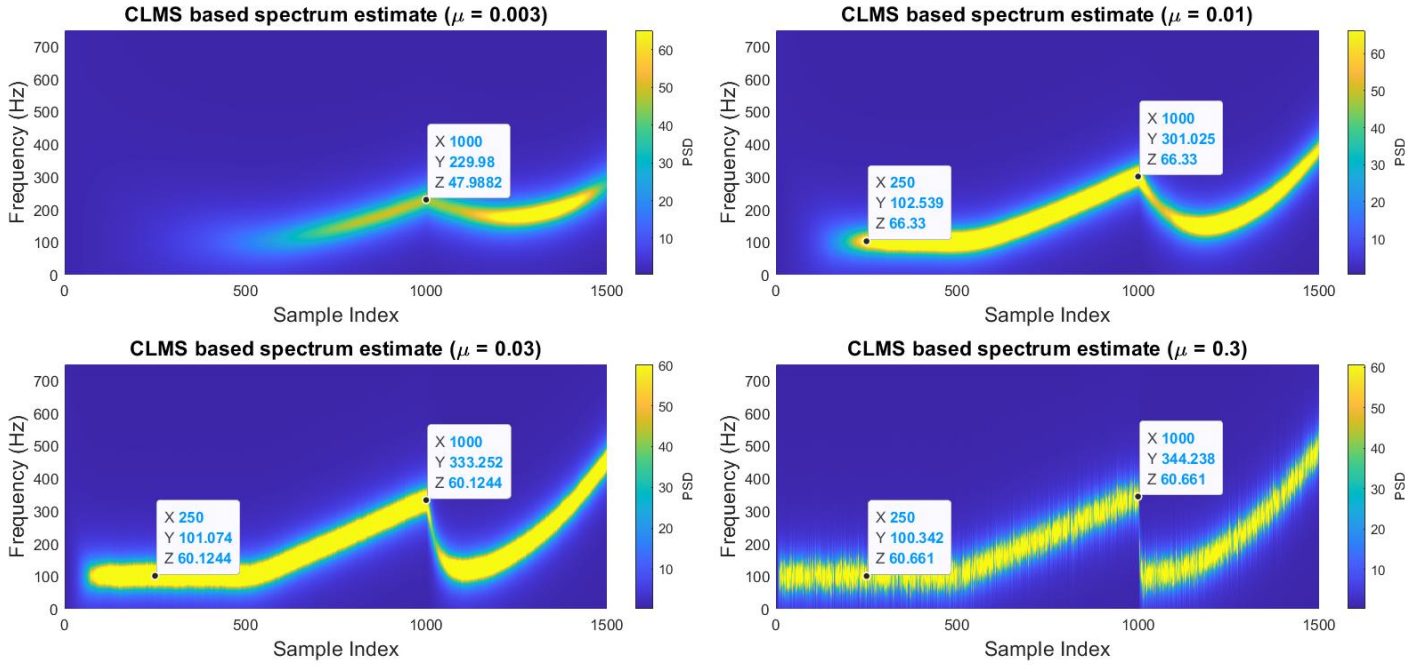

Figure 38. The CLMS based time-frequency spectrum estimates

Compared with the Yule-Walker method, the CLMS algorithm with a suitable step size μ can be applied to the non-stationary signal and estimate the AR model parameters with high accuracy. CLMS only relies on a few previous sample data instead of the complete data to iteratively update the AR model parameters. The CLMS based spectrum estimation effectively avoids the low accuracy problem in stationary AR spectrum and can dynamically capture the frequency variation with little latency time.

Four CLMS based spectrum estimates with different step sizes are listed in Figure 38 and show how the frequency changes over time. If the step size is small, the CLMS algorithm may stop before convergence especially when the number of previous samples is small. Thus, estimated non-optimal parameters contain large errors which result in spectral peak disappearance in a few initial iterations. For example, when $\mu = 0.003$, no dominating frequency can be identified from the time-frequency spectrum in the first 300 iterations. Besides, since the CLMS stops before convergence, there exists large bias error. For example, the estimated frequency at 1000$^{\text{th}}$ time step is $229.98\ \text{Hz}$ instead of the correct $350\ \text{Hz}$. But if the step size is excessively large (e.g., $\mu = 0.3$), the rate of convergence and bias error can be improved at the cost of large oscillation (variance error) in steady state. Thus, the trade-off should be achieved between convergence rate and steady-state variance error. In the above three results, $\mu = 0.03$ may be the optimal step size. Besides, the adaptive step size can also be employed to optimize this algorithm.

3.3 A Real-Time Spectrum Analyser Using Least Mean Square

3.3.a Least squares (LS) solution

The estimate of signal $y(n)$ can be considered as a linear combination of N harmonically related sinusoids.

$$\begin{bmatrix} \hat{y}(0) \\ \hat{y}(1) \\ \vdots \\ \hat{y}(N-1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & e^{j\frac{2\pi}{N}(1)(1)} & \cdots & e^{j\frac{2\pi}{N}(1)(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & e^{j\frac{2\pi}{N}(N-1)(1)} & \cdots & e^{j\frac{2\pi}{N}(N-1)(N-1)} \end{bmatrix} \begin{bmatrix} w(0) \\ w(1) \\ \vdots \\ w(N-1) \end{bmatrix} \tag{3.24}$$

$$\hat{\boldsymbol{y}} = \boldsymbol{F}\boldsymbol{w} \tag{3.25}$$

Where $\boldsymbol{F}$ represents the scaled inverse Fourier Transform matrix containing several sinusoids with different frequencies, $\boldsymbol{w}$ is the unknown weight vector and $\hat{\boldsymbol{y}}$ contains different estimates value $y(n)$.

The cost function $J(\boldsymbol{w})$ can be defined as the sum of squared errors between the estimated signal $\hat{y}(n)$ and true signal $y(n)$ (i.e., the L2 norm of prediction error vector).

$$J(\boldsymbol{w}) = \|\boldsymbol{y} - \hat{\boldsymbol{y}}\|^2 = (\boldsymbol{y} - \boldsymbol{F}\boldsymbol{w})^H(\boldsymbol{y} - \boldsymbol{F}\boldsymbol{w}) = \boldsymbol{y}^H\boldsymbol{y} - \boldsymbol{y}^H\boldsymbol{F}\boldsymbol{w} - \boldsymbol{w}^H\boldsymbol{F}^H\boldsymbol{y} + \boldsymbol{w}^H\boldsymbol{F}^H\boldsymbol{F}\boldsymbol{w} \tag{3.26}$$

The cost function $J(\boldsymbol{w})$ reaches its minimum value when its derivative becomes 0. According to this property, the optimal weight vector $\boldsymbol{w}_{opt}$ of the least square method can be obtained.

$$\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}} = 0 - \boldsymbol{F}^H\boldsymbol{y} - \boldsymbol{F}^H\boldsymbol{y} + 2\boldsymbol{F}^H\boldsymbol{F}\boldsymbol{w}_{opt} = \boldsymbol{0} \tag{3.27}$$

$$\boldsymbol{F}^H\boldsymbol{y} = \boldsymbol{F}^H\boldsymbol{F}\boldsymbol{w}_{opt} \tag{3.28}$$

The full rank square matrix $\boldsymbol{F}^H\boldsymbol{F}$ is invertible, and the optimal weight vector $\boldsymbol{w}_{opt}$ is expressed as:

$$\boldsymbol{w}_{opt} = (\boldsymbol{F}^H\boldsymbol{F})^{-1}\boldsymbol{F}^H\boldsymbol{y} \tag{3.29}$$

Since all column vectors of $\boldsymbol{F}$ are mutually orthogonal, the matrix $\boldsymbol{F}^H\boldsymbol{F} = N\boldsymbol{I}$, and equation (3.29) is simplified as $\boldsymbol{w}_{opt} = N\boldsymbol{F}^H\boldsymbol{y}$.

The discrete Fourier transform and its inverse transform can be expressed as:

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-\frac{j2\pi kn}{N}}, \quad x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{\frac{j2\pi kn}{N}} \tag{3.30}$$

$$\begin{bmatrix} \hat{x}(0) \\ \hat{x}(1) \\ \vdots \\ \hat{x}(N-1) \end{bmatrix} = \frac{1}{N}\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & e^{j\frac{2\pi}{N}(1)(1)} & \cdots & e^{j\frac{2\pi}{N}(1)(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & e^{j\frac{2\pi}{N}(N-1)(1)} & \cdots & e^{j\frac{2\pi}{N}(N-1)(N-1)} \end{bmatrix}\begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix} \rightarrow \hat{\boldsymbol{x}} = \boldsymbol{F}_{IDFT}\boldsymbol{X} \tag{3.31}$$

By comparing the equation (3.24) and (3.31), the weight vector $\boldsymbol{w}$ can be assumed to be the DFT coefficient vector $\boldsymbol{X}$ and the relationship between $\boldsymbol{F}$ and $\boldsymbol{F}_{IDFT}$ should be.

$$N\boldsymbol{F}_{IDFT} = \boldsymbol{F} \tag{3.32}$$

The inverse DFT matrix can be considered as the scaled Hermitian transpose of the DFT matrix.

$$\boldsymbol{F}_{IDFT} = \frac{1}{N}\boldsymbol{F}_{DFT}^H \tag{3.33}$$

Thus, the matrix $\boldsymbol{F}$ in equation (3.25) is exactly the Hermitian transpose of $\boldsymbol{F}_{DFT}$.

$$\boldsymbol{F} = \boldsymbol{F}_{DFT}^H \rightarrow \boldsymbol{w}_{opt} = N\boldsymbol{F}^H\boldsymbol{y} = N\boldsymbol{F}_{DFT}\boldsymbol{y} \tag{3.34}$$

According to (3.34), the optimal weights $\boldsymbol{w}_{opt}$ obtained by LS algorithm is the scaled discrete Fourier transform coefficient of the input signal $y(n)$.

3.3.b Least squares interpretation of the DFT

The column vectors of the DFT matrix are mutually orthogonal and they can be considered as linearly independent complex sinusoid bases which span an N-dimensional subspace. Each discrete signal $\boldsymbol{x}$ in time domain can be decomposed into the weighted sum of these orthogonal bases. The normalised Fourier transform matrix is a unitary matrix that purely rotates $\boldsymbol{x}$ and changes its coordinate system without amplification. Thus, $\boldsymbol{x}$ is just changed from one coordinate (e.g., discrete time domain) to another coordinate (e.g., frequency domain), which explains the term “change of basis”.

Typically, DCT linearly projects $\boldsymbol{x}$ onto a series of harmonically related sinusoids whose frequency is a multiple of f_s/N (f_s is the sampling frequency). The projection is achieved by computing the inner products of N samples of $\boldsymbol{x}$ and all N basis vectors (i.e., new coordinate vectors). In this projection, there is no information loss and power loss theoretically. In practice, due to the incoherent sampling problem, the square error between the original signal $\boldsymbol{x}$ and signal estimates $\hat{\boldsymbol{x}}$ cannot be exactly zero, and only the least square (LS) can be obtained. Thus, the number of samples (N) should be as large as possible.

3.3.c DFT-CLMS algorithm

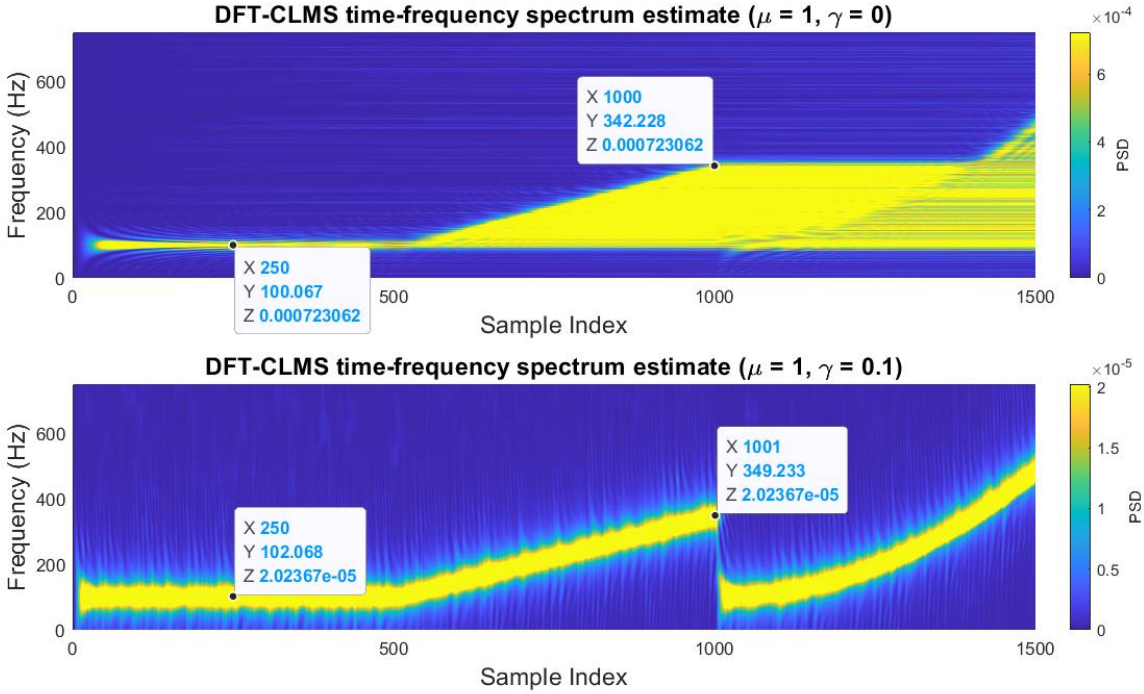

Figure 39. The power spectrum estimated by DFT-CLMS algorithm

Figure 39 demonstrates the time-frequency spectrum of the FM signal by using DFT-CLMS algorithm. Compared with the AR model-based spectrum estimation, the DFT-CLMS method achieves faster convergence. This first dominating frequency estimate (100 Hz) is computed after 40 iterations. However, if leakage coefficient $\gamma = 0$, only the frequency of the stationary segment of FM signal can be successfully identified from the spectrum with small steady-state error due to its narrow spectral peaks in the first 500 iterations. For the non-stationary segment, all frequencies of previous data are highlighted. In the last 1000 iterations, the spectral plateau instead of a single peak is obtained in each individual spectrum where many frequency components have similar PSD values although the FM signal has a single frequency at each time. When $\gamma = 0$, the current weight estimates are determined by all past weights and all past samples according to equation (3.35), and the current samples can only affect present errors and gradient descent directions. Thus, all past frequencies remain in the current spectrum which can be considered as the superimposition of all past spectrums. The 100 Hz should have the highest peak in theory. In this task, PSD values are small, and the peak at 100 Hz is not conspicuous. Thus, the time-frequency spectrum estimated by the standard DFT-CLMS algorithm fails to resemble the true power spectrum for non-stationary signals and has lower overall performance than the spectrum estimated by AR model-based method.

$$\boldsymbol{w}(n+1) = (1-\mu\gamma)\boldsymbol{w}(n) + \mu e^*(n)\mathbf{x}(n) \tag{3.35}$$

The standard CLMS can be replaced by the leaky CLMS algorithm to reduce the sensitivity to past weight estimates. If the γ is sufficiently large, the effect of previous weights can be ignored after a few iterations and current weights are dominated by the current sample and error. However, the excessively large γ may result in significant variance error. Thus, $\gamma = 0.1$ may be the optimal choice in this case. The DFT-leaky CLMS algorithm generates a comparable or even better time-frequency spectrum of FM signal than the AR model-based algorithm.

3.3.d DFT-CLMS for the EEG

The standard and leaky DFT-CLMS algorithms are applied to the EEG data segment of 1200 samples. The start sample index is 2000. A larger threshold is used to remove outliers in the PSD matrix. When $\gamma = 0$, the desired SSVEP fundamental frequency of 13 Hz can be detected after about 400 iterations and the first harmonic frequency (26 Hz) appears after 600 iterations. Besides, the alpha-rhythm with $[8\text{ Hz}, 10\text{ Hz}]$ and power-line interference at 50 Hz can also be obtained from the left time-frequency spectrum in Figure 40. Generally, the essential features of EEG data can be observed from this diagram. However, for EEG data, the leaky CLMS cannot improve the performance of spectrum estimation and generates inaccurate and

distorted spectrums due to the noise in signal. Thus, the EEG data may be stationary, or its frequency variation rate is low. The standard DFT-CLMS is more suitable for this EEG data. The low frequency resolution can be attributed to the small number of available sample data (i.e., small N).

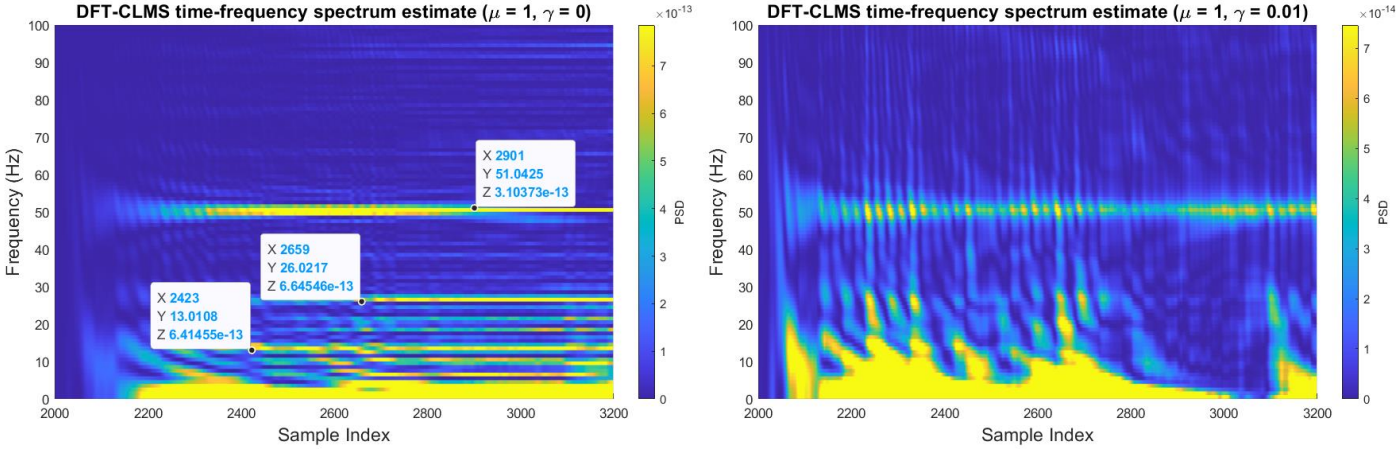

Figure 40. The DFT-CLMS spectrum estimate of EEG data

4 From LMS to Deep Learning

4.1 LMS based linear prediction for non-stationary signal

In this task, the standard LMS algorithm is implemented to achieve linear prediction for non-stationary time series. When the step size is set as $\mu = 10^{-5}$ and AR(4) model is used, the LMS converges after about 200 samples. Generally, the variation pattern of this zero-mean non-stationary time series is successfully captured despite some errors. The overall MSE and prediction gain (R_p) are $16.032\,\mathrm{dB}$ and $5.196\,\mathrm{dB}$ respectively. In ideal cases, the MSE should be 0 and the R_p is close to the infinity. In this case, the large MSE and small R_p values can be attributed to the large prediction errors in the first 200 steps where the local MSE and R_p are $21.270\,\mathrm{dB}$ and $-1.928\,\mathrm{dB}$. In the last 800 steps, the local MSE and R_p are $12.210\,\mathrm{dB}$ and $9.384\,\mathrm{dB}$ which are better than the global MSE and R_p. After convergence, the peak values of one-step prediction signals are slightly smaller than the original peak values, which is the primary source of remaining MSE error.

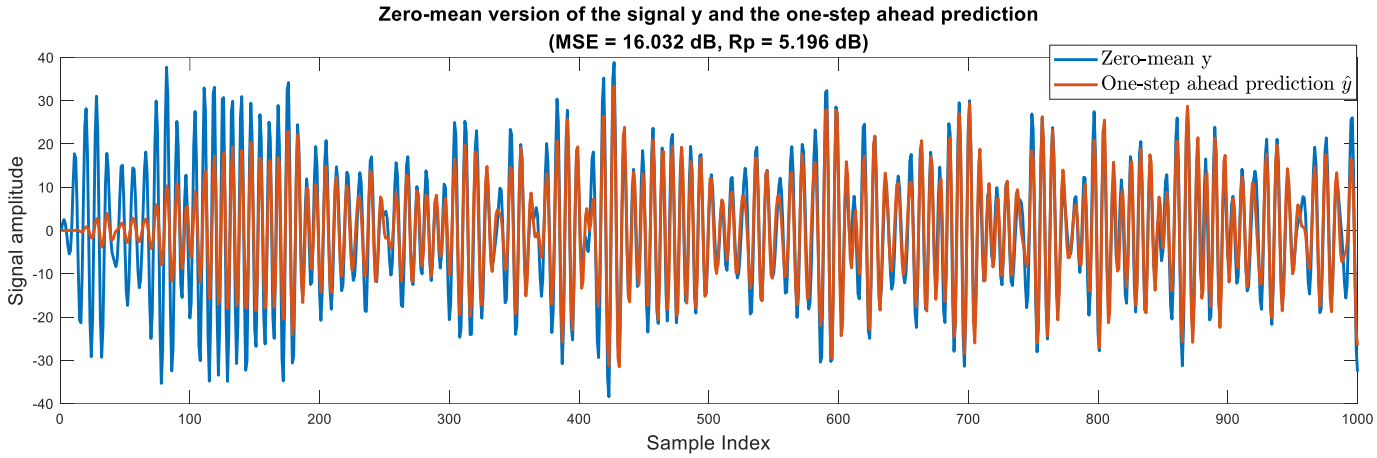

Figure 41. The zero-mean version of the original non-stationary time-series and linear prediction results

4.2 Dynamical perceptron for non-linear prediction

Since the time series is generated by a non-linear process, an activation function (e.g., hyperbolic tangent function – tanh in this task) can be applied to the output of the LMS to form a dynamical perceptron and get more accurate model estimates. The weight update equation now becomes:

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu\Phi'\big(\mathbf{x}^T(n)\boldsymbol{w}(n)\big)e(n)\mathbf{x}(n) = \boldsymbol{w}(n) + \mu\big(1-\tanh^2\big(\mathbf{x}^T(n)\boldsymbol{w}(n)\big)\big)e(n)\mathbf{x}(n) \qquad (4.1)$$

When $\mu = 10^{-5}$, the one-step prediction results are shown in Figure 42 where the amplitudes of all predicted signals range from -1 to 1. Tanh function is a sigmoidal activation function and non-linearly scales down

the input variable range of $(-\infty, \infty)$ to the output variable range of $(-1, 1)$. It achieves the Bounded Input Bounded Output (BIBO) stability. However, the amplitudes of most original signals are considerably greater than 1. The large amplitude difference causes the unacceptably large overall MSE (22.958 dB) and small R_p (-24.423 dB). Although the tanh function based LMS algorithm successfully identifies the frequency variation pattern of this time series, it has worse overall performance than the standard LMS used in the last section. Thus, tanh function with unit scaling factor is not appropriate in this case. However, tanh function may achieve good results if the amplitude range of zero-mean time series is normalised to $(-1, 1)$ in advance.

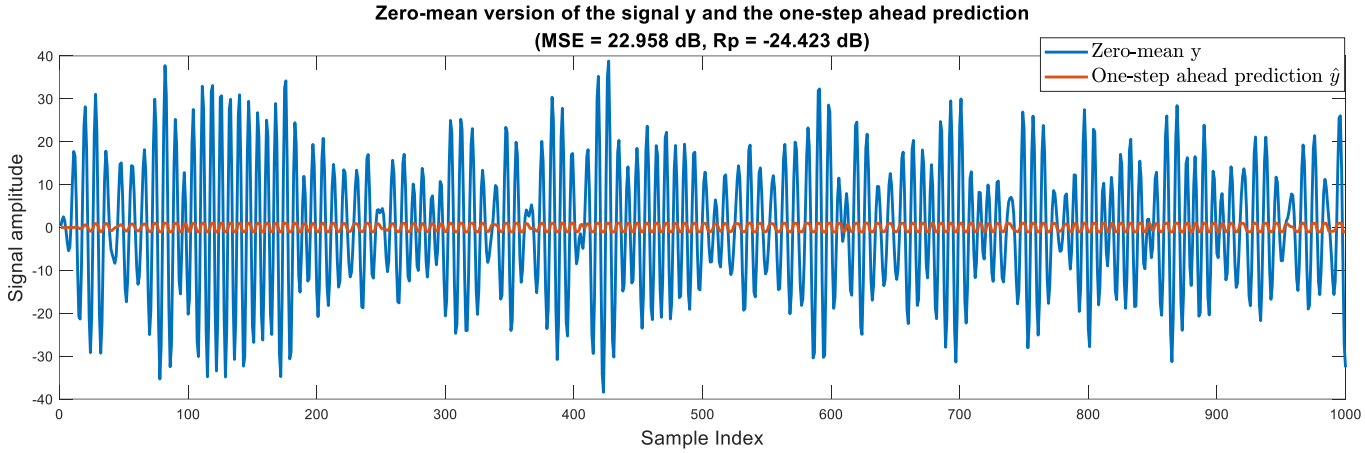

Figure 42. The zero-mean time-series and non-linear prediction results generated by LMS with scaled tanh

4.3 The effect of scaled activation function

Figure 43 demonstrates the effect of scaling factor a on the MSE and Rp of the tanh function based LMS algorithm. Since the amplitudes of zero-mean signal approximately range from -39 to 39, but the output of the scaled tanh function ranges from $-a$ to a, the scaling factor a must be greater than 39 to capture the peaks of time series and avoid clipping. However, a cannot be excessively large. Otherwise, the non-linearity of the scaled tanh function may to too large, and the output of LMS may be unstable. A slight change in signal amplitude or frequency caused by noise may be significantly amplified and result in a large overshoot.

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu a\left(1 - \tanh^2\left(\mathbf{x}^T(n)\boldsymbol{w}(n)\right)\right)e(n)\mathbf{x}(n) \tag{4.2}$$

According to equation (4.2), the learning rate of the modified LMS algorithm is the product of μ and a. If the same $\mu = 10^{-5}$ is used as in the last two tasks, the equivalent step size μa would be too large and the LMS prediction results may have a large steady state variance error. Thus, smaller μ should be used. After several trials, $\mu = 1.5 \times 10^5$ is an appropriate choice. According to Figure 43, $[40,90]$ is an appropriate range for the scaling factor a, and 78 is the optimal scaling factor which achieves the smallest MSE and largest Rp at the same time.

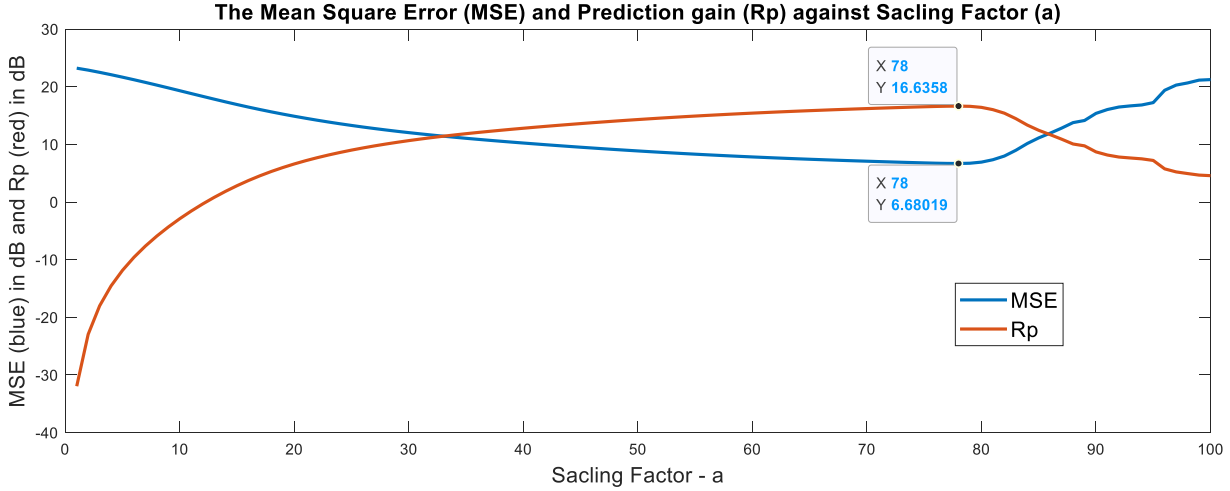

Figure 43. The MSE and Rp curves generated by the LMS with scaled tanh function when $\mu = 1.5 \times 10^{-7}$

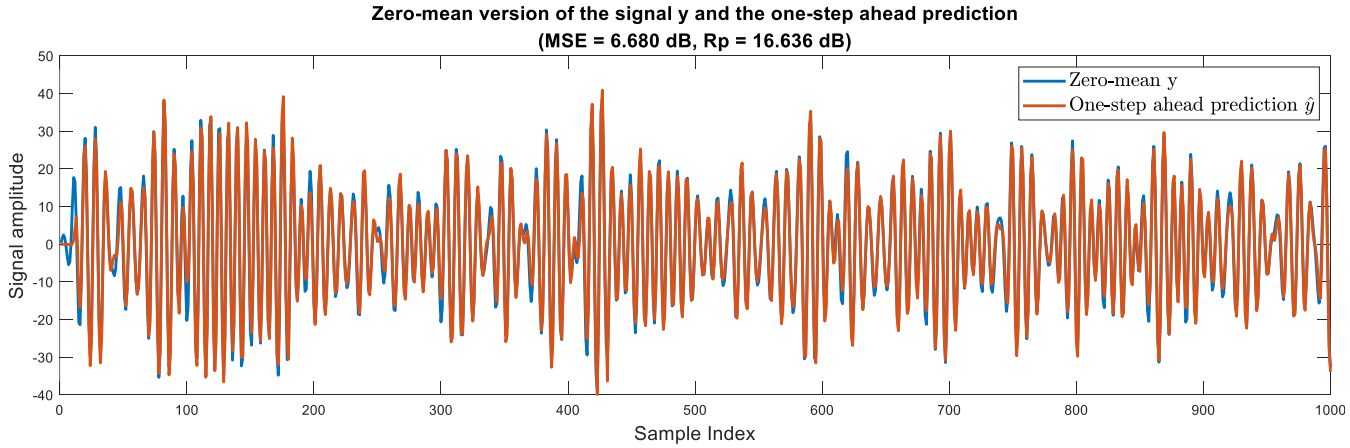

Figure 44. The zero-mean time-series and prediction results generated by LMS with scaled tanh function

When the optimal scaling factor ($a = 78$) is used, the one-step prediction results are demonstrated in Figure 44. Compared with the standard LMS algorithm, the LMS with scaled tanh function has a significantly smaller overall MSE (6.680 dB) and larger Rp (16.636 dB). Thus, it outperforms the standard LMS algorithm regarding non-stationary time series prediction and non-linear pattern estimation.

4.4 The effect of activation function with a bias input

The scaled tanh based LMS is only applicable for zero-mean signal. By adding a bias input to the model, the modified LMS can vertically move the predicted signal to fit the non-zero mean original signal and automatically account for the mean value. The bias can be implemented by adding one unit term to the original input vector to form the augmented input vector $[1, \mathbf{x}^T(n)]^T$ where $\mathbf{x}(n)$ is a column vector containing the last 4 sample data of time series. Due to the additional input, the one more coefficient w_0 should be added to the weight vector $\boldsymbol{w}(n)$. As shown in (4.3), the bias b is exactly the value of w_0 in each iteration.

$$\hat{y} = a \times \tanh(\mathbf{x}^T(n)\boldsymbol{w}(n) + b) = a \times \tanh([1, \mathbf{x}^T(n)][w_0, \boldsymbol{w}^T(n)]^T), \qquad b = w_0 \tag{4.3}$$

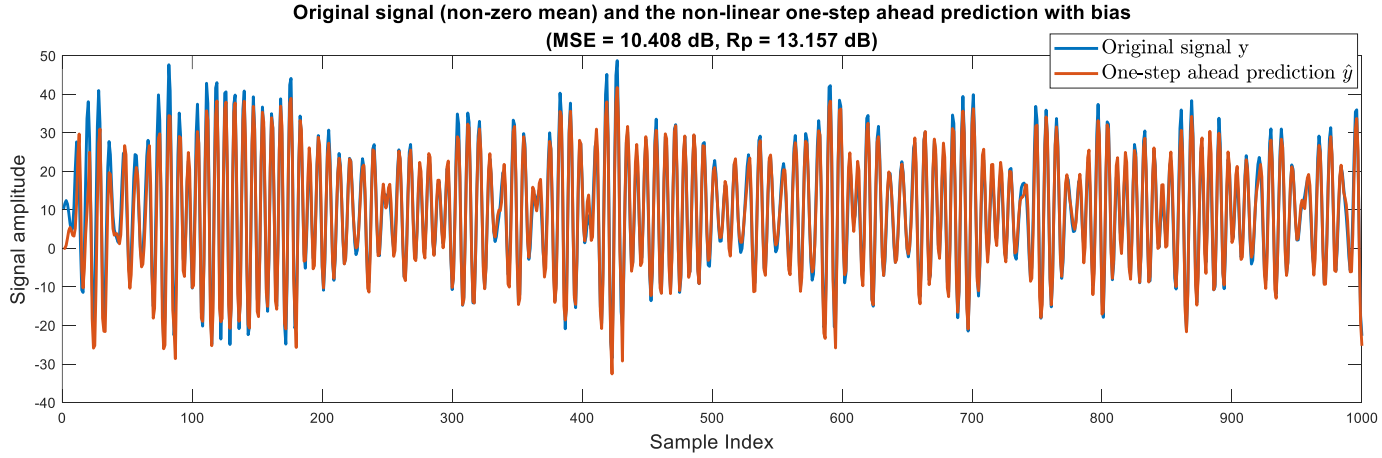

Figure 45. original signal (non-zero mean) and the non-linear one-step ahead prediction with bias

Since the LMS algorithm has been modified, the previous optimal scaling factor $a = 78$ becomes invalid. If a small scaling factor is selected, the LMS with the bias input may fail to model the mean value of original signal and move the predicted signal to this level. The large a results in the overshoot. If the MSE value is used as the metric, the new optimal a is 63 when $\mu = 1.5 \times 10^5$. Figure 45 illustrates the original signal and the prediction results. The measured MSE and prediction gain are 10.408 dB and 13.157 dB. Thus, the LMS with bias input tends to have larger MSE and smaller prediction gain than the LMS without bias input due to the additional mean value estimation error and some large prediction errors in the first few time steps caused

by the low convergence rate.

According to Figure 46, the LMS with bias input requires about 83 iterations to converge, while the LMS without bias only requires 13 iterations to converge. The LMS with unbiased input only needs to detect the zero-mean signal variation and update 4 weight coefficients. However, the LMS with biased input has to simultaneously identify the mean value and variation pattern of the original signal and update five weight coefficients. Thus, its relatively low convergence rate and large MSE are reasonable and predictable. The main advantage of this algorithm is the automatic mean value prediction, which reduces the pre-processing requirements on the time series to be analysed at the cost of slight accuracy loss.

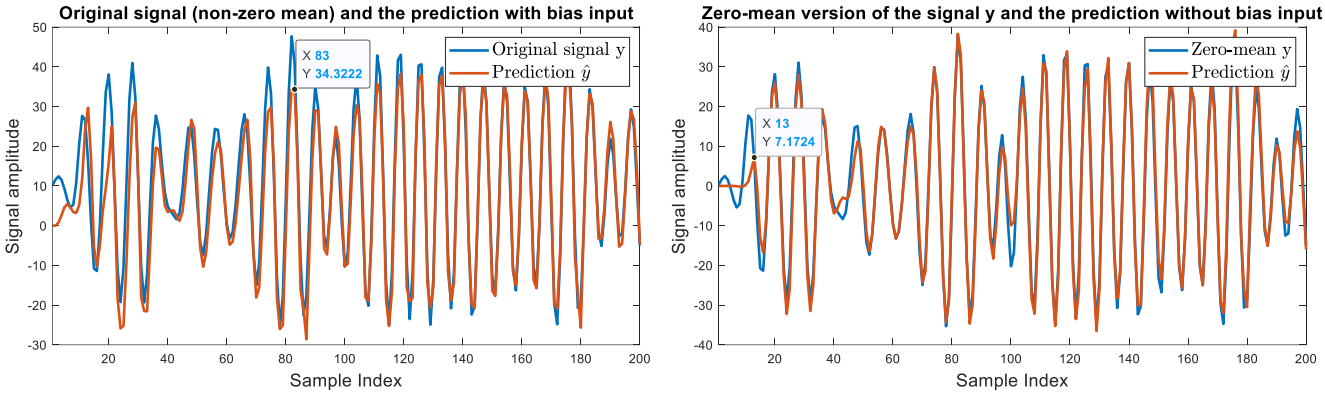

Figure 46. The convergence rate comparison between LMS with bias input and LMS without bias input

4.5 Convergence acceleration and pre-trained weights

Since the initial guess of weights usually differs significantly from the optimal weights, the LMS algorithm requires many samples to converge. To accelerate the convergence of LMS, 100 iterations (epochs) are used to pre-train the weights by over-fitting to the first 20 samples. Same scaling factor a and step size μ are used in pre-training and entire time-series prediction, which can avoid the potential amplitude attenuation or overshoot problems caused by parameter mismatch. Figure 47 illustrates the original signal and non-linear prediction results produced with bias ($a = 62$) and pre-trained weights when $\mu = 1.5 \times 10^{-7}$.

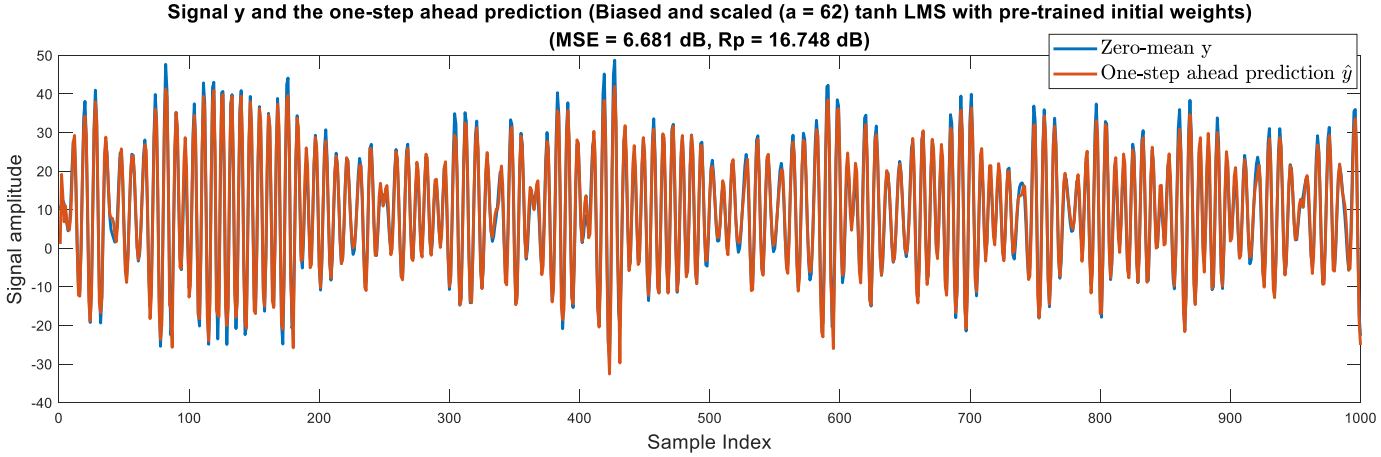

Figure 47. Original signal and the non-linear prediction produced with bias and pre-trained weights

The pre-training process yields more reasonable initial guesses of the weights and significantly improves the rate of convergence. The LMS with biased input and pretrained weights converges with only five samples. Besides, the MSE drops to $6.681\ \mathrm{dB}$, and the prediction gain rises to $16.748\ \mathrm{dB}$. They are $3.727\ \mathrm{dB}$ lower and $3.591\ \mathrm{dB}$ higher than the results in Figure 45. Generally, the MSE reduction can be attributed to the disappearance of the first few large prediction errors. After 100 samples, the advantage of pre-trained weights reduces to a negligible level and the LMS produces similar prediction curves as in Figure 45. Compared with the LMS with $\boldsymbol{w}(0) = \boldsymbol{0}$ used in last task (Figure 45), the LMS with pretrained weights ($\boldsymbol{w}(0) = \boldsymbol{w}_{init}$) has considerably higher performance in non-linear signal prediction. However, the additional pre-training process increases the computational complexity of the LMS algorithm.

4.6 Backpropagation

Several simple dynamical perceptron models (i.e., single neurons) can be combined to form a deep (neural) network that is sufficiently expressive to model highly nonlinear real-world signals. Typically, the deep network consists of multiple layers and each layer has several neurons. Each neuron has a simple activation function (e.g., tanh and ReLU) and utilises the weighted sum of outputs of previous layer as the input value to compute its own output (equation (4.4)) in the forward propagation. Although all neurons make contribution to the prediction error, neurons in hidden layers cannot get access to the teaching signal directly, which means direct gradient training is not impracticable.

$$z_j^{(l)} = \boldsymbol{w}^{(l-1)}\boldsymbol{a}^{(l-1)} = w_{0j}^{(l-1)} + \sum_{i=1} w_{ij}^{(l-1)}\, a_i^{(l-1)}, \qquad a_j^{(l)} = \sigma\left(z_j^{(l)}\right), \qquad l \in [1, L] \tag{4.4}$$

Where $z_j^{(l)}$ and $a_j^{(l)}$ are the input and output of the j^{th} neuron in the l^{th} layer, $\sigma(\cdot)$ is the specified activation function, $w_{ij}^{(l-1)}$ is the weight of the connection between $a_i^{(l-1)}$ and $a_j^{(l)}$, and $w_{0j}^{(l-1)}$ is the bias.

The backpropagation algorithm is based on the chain rule ((4.5)) and designed for supervised learning where the output of the deep network can be compared with the teaching signal, and the prediction error can be computed. It can repeatedly adjust the model parameters (e.g., weights and bias) to minimise the cost function (e.g., MSE in this coursework). The output error can be propagated backward to each neuron, and the delta error δ in each unit is defined in equation (4.6). Each error in current layer (l) is determined by all errors in the next layers ($l+1$) and is a portion of output error. Besides, δ can store the past computation results and effectively avoids repeat computation to speed up the algorithm.

$$\frac{\partial J(\boldsymbol{w})}{\partial w_{ij}^{(l-1)}} = \frac{\partial J(\boldsymbol{w})}{\partial z_j^{(l)}} \times \frac{\partial z_j^{(l)}}{\partial w_{ij}^{(l-1)}} = \frac{\partial J(\boldsymbol{w})}{\partial z_j^{(l)}} \times a_i^{(l-1)} = \delta_j^{(l)} \times a_i^{(l-1)} \tag{4.5}$$

$$\delta_j^{(l)} = \frac{\partial J(\boldsymbol{w})}{\partial z_j^{(l)}} = \sum_k \delta_k^{(l+1)} \frac{\partial z_k^{(l+1)}}{\partial a_j^{(l)}} \frac{\partial a_j^{(l)}}{\partial z_j^{(l)}} = \sum_k \delta_k^{(l+1)} w_{jk}^{(l+1)} \sigma'\left(z_j^{(l)}\right) \tag{4.6}$$

According to the general gradient descent algorithm, the degree of adjustment of each parameter in each iteration is determined by the partial derivatives of the cost function $J(\boldsymbol{w})$ with respect to this target weight (or bias). Equation (4.7) is the weight update equation in back-propagation.

$$w_{ij}^{(l-1)}(n+1) = w_{ij}^{(l-1)}(n) - \mu \frac{\partial J\left(\boldsymbol{w}(n)\right)}{\partial w_{ij}^{(l-1)}(n)} = w_{ij}^{(l-1)}(n) - \mu \delta_j^{(l)}(n) a_i^{(l-1)}(n) \tag{4.7}$$

$$\delta_j^{(l)} = \frac{\partial J(\boldsymbol{w})}{\partial a_j^{(l)}} \times \frac{\partial a_j^{(l)}}{\partial z_j^{(l)}} = \frac{\partial J(\boldsymbol{w})}{\partial a_j^{(l)}} \sigma'\left(z_j^{(l)}\right) \rightarrow w_{ij}^{(l-1)} = w_{ij}^{(l-1)} - \mu \frac{\partial J(\boldsymbol{w})}{\partial a_j^{(l)}} \sigma'\left(z_j^{(l)}\right) a_i^{(l-1)} \tag{4.8}$$

Generally, the whole backpropagation algorithm consists of five steps. Firstly, the appropriate initial weights are selected to ensure the preservation of the expected energy of the output. The selection of learning rate and momentum is also included in this step. Secondly, the new observation of input signal $\boldsymbol{z}^{(0)}(n)$ and teaching signal $y(n)$ should be observed. Thirdly, the feed forward algorithm should be implemented. Each neuron computes its non-linear output based on the inputs from the previous layer, as shown in equation (4.4). Fourthly, the output layer produces the non-linear prediction $\hat{y}(n)$ of the whole network, and $\hat{y}(n)$ is compared with the ground truth value $y(n)$ to get the prediction error $\delta^L = \hat{y}(n) - y(n)$ and compute cost function $J(\boldsymbol{w})$ with respect to all weights $\boldsymbol{w}(n)$ used in network. Finally, the overall error is propagated back to each neuron in each layer (equation (4.6)), and equation (4.7) is used to update all weights and biases.

4.7 Comparison between deep network and single dynamical perceptron

In Figure 48, 10 sinusoids with different frequencies and magnitudes are non-linearly combined with the additive white Gaussian noise to form the desired output signal $y[n]$. First 50% data are used as the training data set, and the remaining 50% data are the test data set. In order to identify the non-linear generating process of this data set, a dynamical perceptron (single neuron) with linear activation function, a dynamical perceptron with tanh activation function, and a deep network with four hidden layers and ReLU activation function were implemented and applied to the data set. The data fitting and prediction results are shown in Figure 49. It can be observed that none of these three models can perfectly identify the data variation pattern. However, the deep network successfully captures the general trend of data and outperforms the other two models because it has more degrees of freedom. Two simple dynamical perceptrons achieve similar results, and the advantage of the tanh function is slight. The high similarity can be attributed to the center range around the origin point where the tanh function can be approximated as a linear function. Two single neurons perform well in the first 10% data but fail to capture the remaining significant variations, such as two deep valleys at about 45th and 75th samples. Their data prediction curves are significantly smoother than the deep network's results. Both dynamical perceptrons may overfit to the first 10% data, and following training data fail to rectify this problem due to the lack of degrees of freedom.

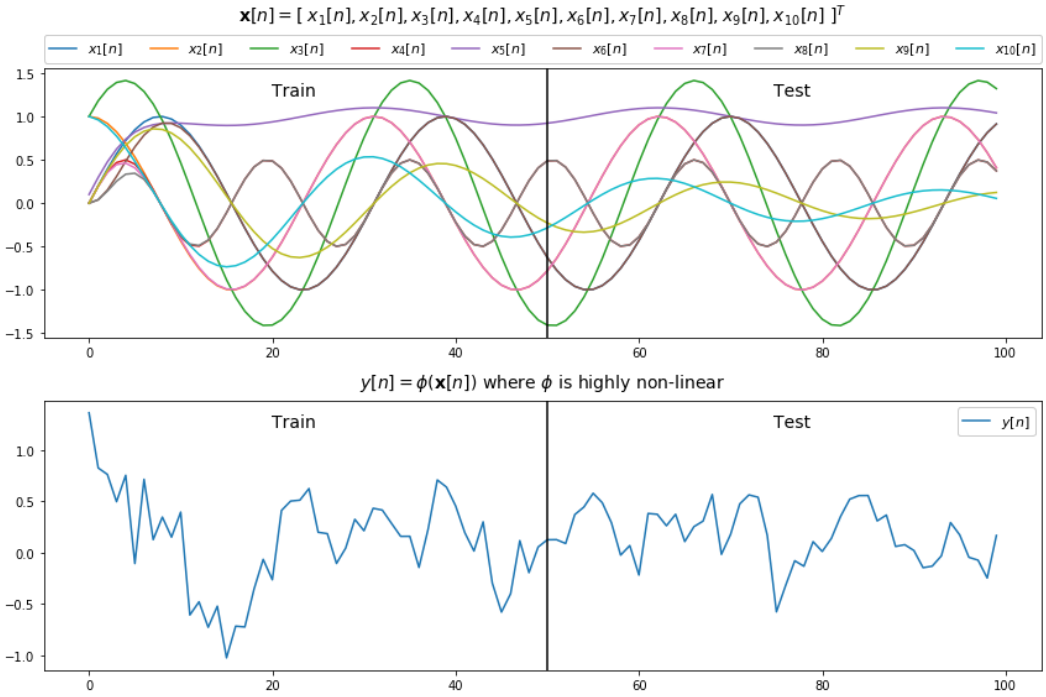

Figure 48. Input sinusoids and the non-linear output

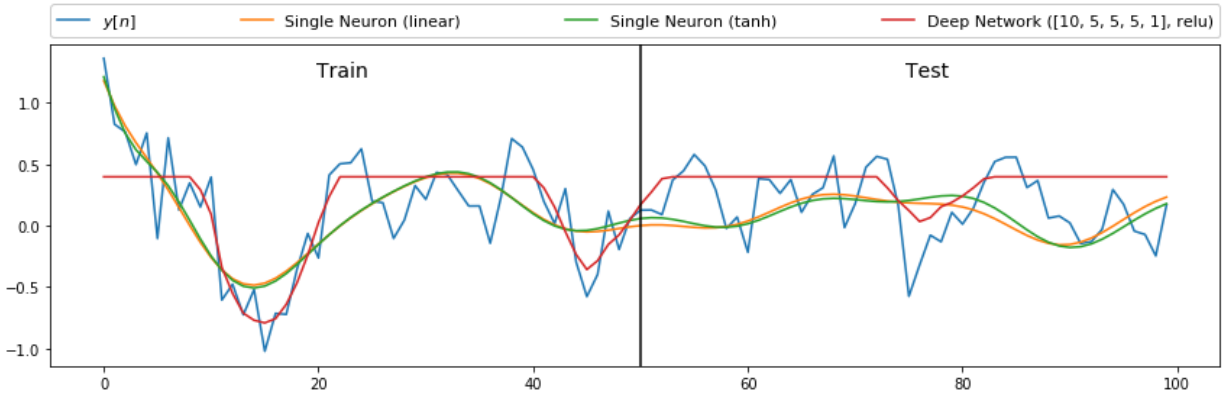

Figure 49. The ground truth signal and the prediction results of the three models

Figure 50 compares the train and test errors of three models. Two dynamical perceptrons converge within about 8000 epochs and yield relatively large test losses (about 0.1). Their learning curves are similar and have no fluctuation. By comparison, the deep network requires about 15000 epochs to converge but yields

a smaller test error (about 0.85). Abundant weights of deep network provide enough degrees of freedom to ensure small steady-state errors but require more epochs for optimization. The large initial fluctuation in test loss curve of the deep network may be attributed to the over-fitting problem caused by the complicated model.

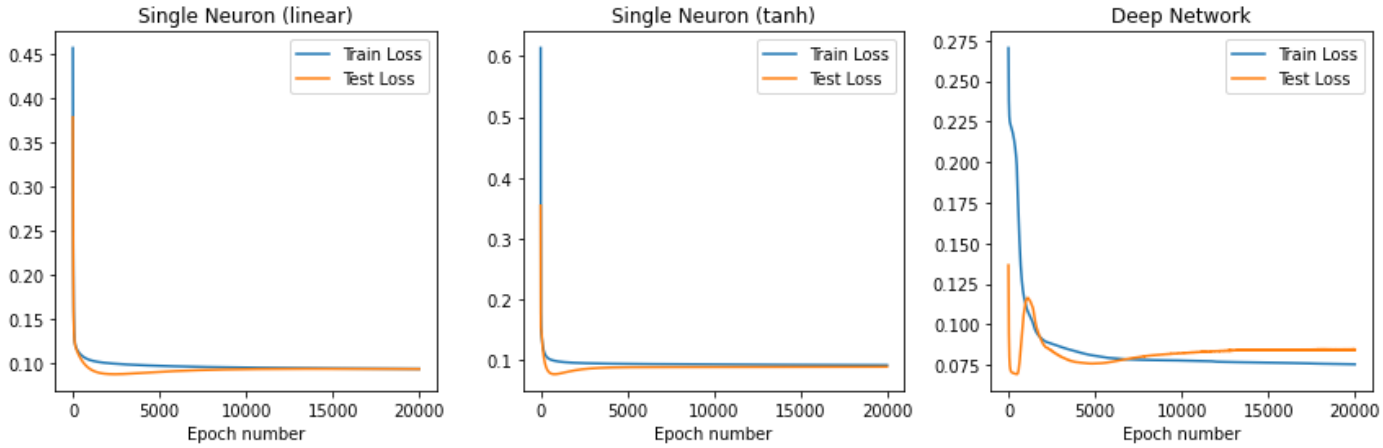

Figure 50. The learning curves of two single neurons and deep network (noise variance is 0.05)

4.8 Drawbacks of deep network

In this section, the performances of three models are compared when three different noise variances are used to generate the training and test data. When the noise variance is small, the deep network outperforms the dynamical perceptrons. When the signal is noiseless (Figure 51), the deep network almost perfectly predicts the training signal and achieves approximately zero training loss (Figure 52). Despite potential overfitting problems and some bias errors, it successfully captures all the significant test signal variation and achieves a reasonably small test loss. The dynamical perceptrons have a severe underfitting problem and can only roughly identify the general variation pattern. Many important details are ignored.

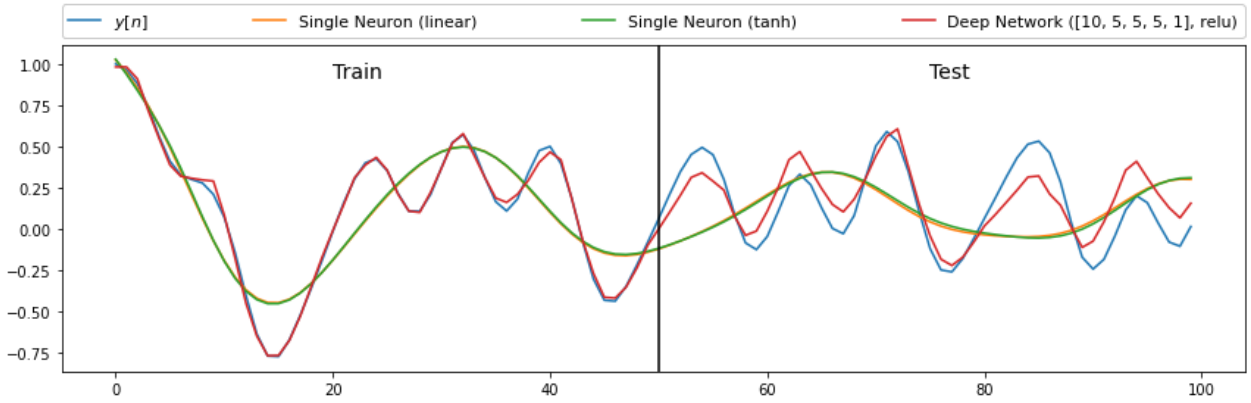

Figure 51. Ground truth signal and the prediction results of three models (noise variance is 0)

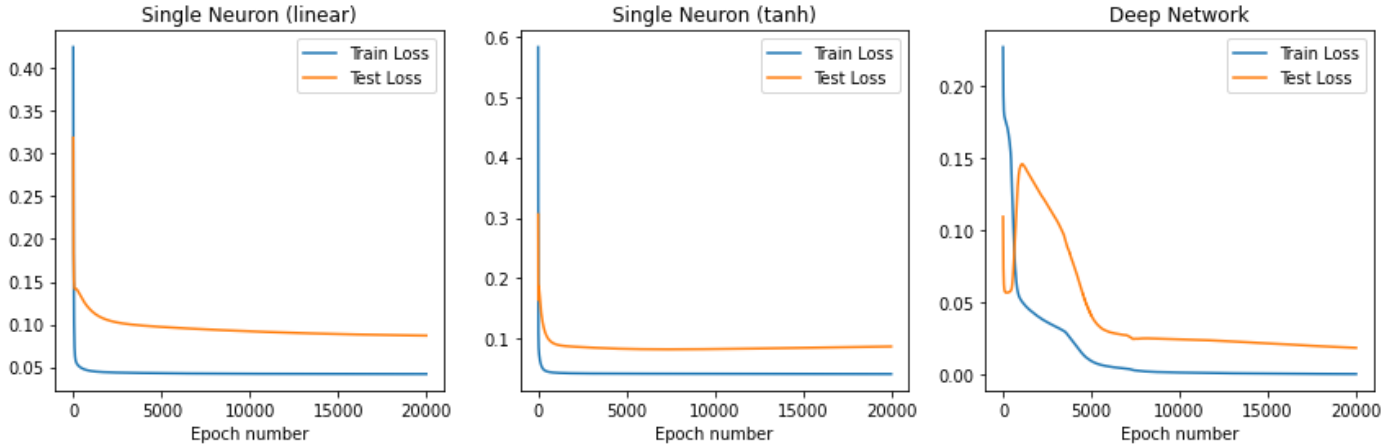

Figure 52. The learning curves of three models (noise variance is 0)

When noise variance rises from 0 to 0.03, the performance of the deep network degrades. It ignores some trivial variations caused by both noise and useful signals. Its steady-state train and test losses increase slightly to 0.05 and 0.06 which are still small and acceptable. For dynamical perceptrons, as the noise level increases, their steady-state train losses increase, but their test losses change slightly. Thus, the deep network still has the overwhelming advantage over single neurons.

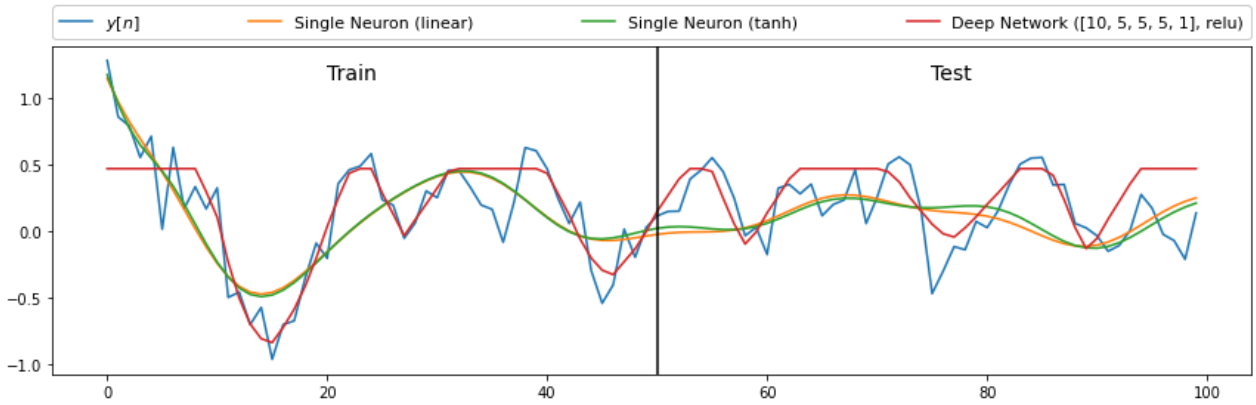

Figure 53. Ground truth signal and the prediction results of three models (noise variance is 0.03)

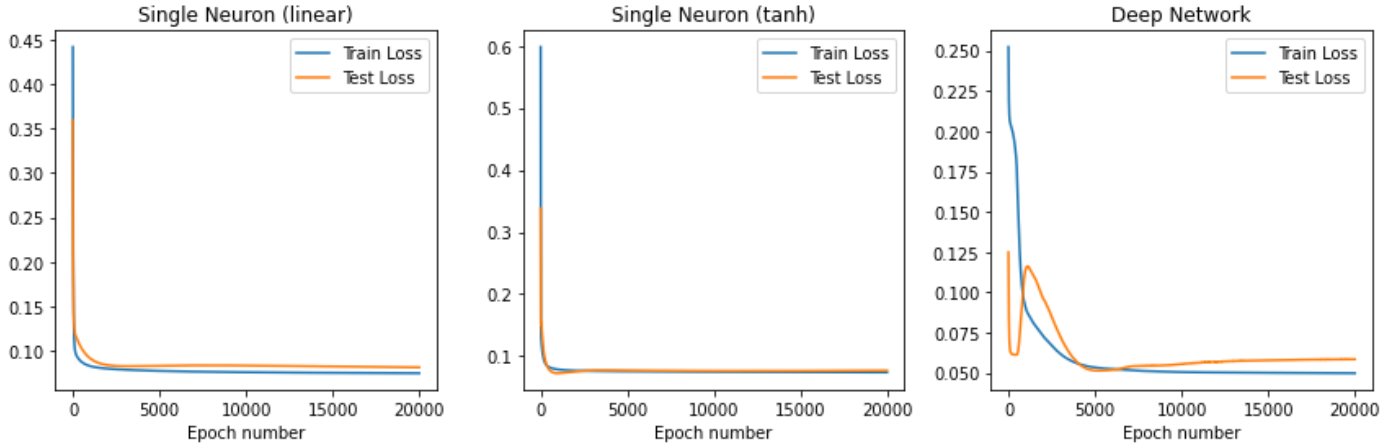

Figure 54. The learning curves of three models (noise variance is 0.03)

However, when the noise variance becomes large (e.g., 0.3), the deep network is likely to overfit the noise component in training data and has poor performance in test data prediction due to its large number of freedom degrees. According to Figure 56, after 10000 epochs, its train loss and test loss curves begin to oscillate with increasing amplitudes, which may indicate that the selected learning rate is excessively large. Besides, since the overall trend of the test loss is increasing, the deep network diverges in this trial. After 20000 epochs, the deep network has smaller train loss but significantly larger test loss than both dynamical perceptrons.

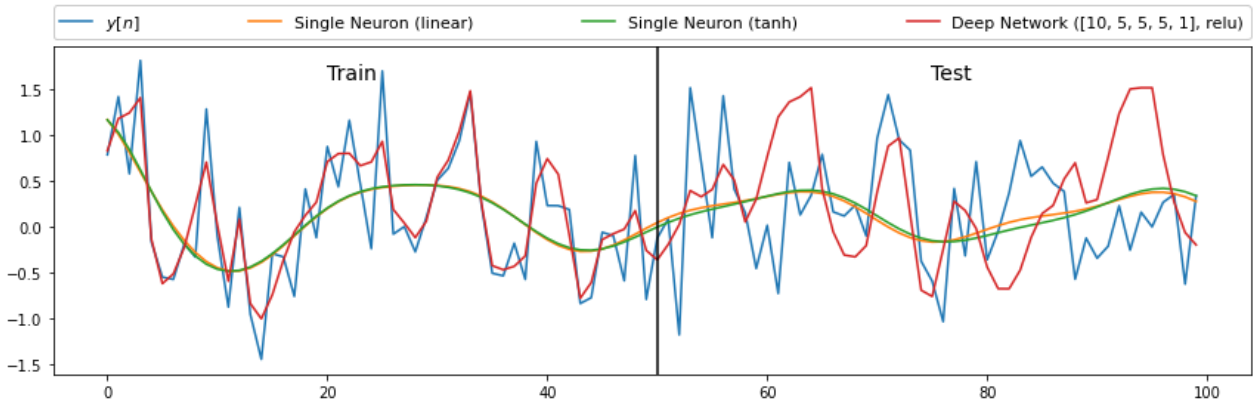

Figure 55. Ground truth signal and the prediction results of three models (noise variance is 0.3)

Generally, single neurons have under-fitting problems, and the noise variance variation only slightly impacts their performances. Six signal prediction results plotted in Figure 51, Figure 53 and Figure 55 have high similarity. By comparison, multiple neurons in hidden layers enable the deep network to have sufficient degrees of freedom to expressively model non-linear signals when the noise level is small. However, deep networks also have many shortcomings, as analysed previously. Firstly, most complicated models with many adjustable parameters (e.g., weights and biases) are prone to overfitting especially when the number of training data is small. Compared with a single neuron, the deep network is more likely to mistakenly overfit the noise component or other irrelevant features in the training datasets and become inapplicable to other unseen new data. Secondly, the indispensable backpropagation algorithm with high complexity has strict storage and computation requirements on the hardware. The deep network is computationally expensive and tends to have a low rate of convergence. Thirdly, its performance can be significantly affected by the noise. Once the noise variance is high, the deep network may have the problem of instability and divergence. Finally, numerous weights should be appropriately initialised to accelerate convergence, which also increases the implementation difficulty of a deep network.

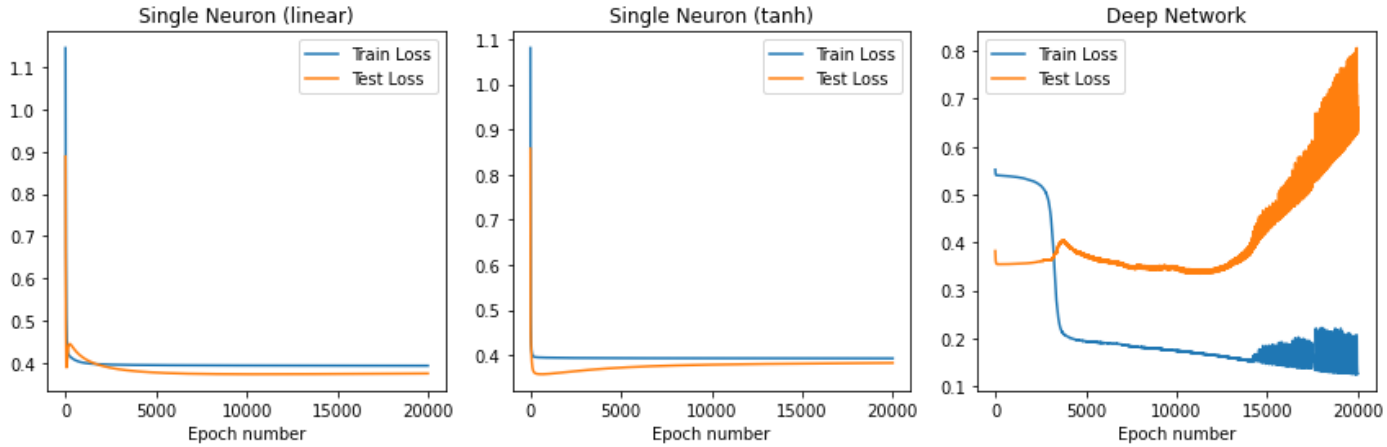

Figure 56. The learning curves of three models (noise variance is 0.3)

5 Tensor Decompositions for Big Data Applications

The results of this section are placed under the folder – “CW Part5 Code”.

Reference

[1] M. Kamenetsky and B. Widrow, "A variable leaky LMS adaptive algorithm," in *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers, 2004.*, 2004, vol. 1, pp. 125-128 Vol.1.

[2] Y. Jiao, R. Y. P. Cheung, W. W. Y. Chow, and M. P. C. Mok, "A novel gradient adaptive step size LMS algorithm with dual adaptive filters," in *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2013, pp. 4803-4806.