Content

1. Classical and Modern Spectrum Estimation

1.1 Properties of Power Spectral Density (PSD)

According to the definition 2 of PSD,

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

$$= \lim_{N\to\infty} E\left\{\frac{1}{N}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} x(n)x^*(m)e^{-j\omega(n-m)}\right\} \tag{2}$$

$$= \lim_{N\to\infty} \frac{1}{N}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} E\{x(n)x^*(m)\}e^{-j\omega(n-m)} \tag{3}$$

Let $k = n - m$,

$$P(\omega) = \lim_{N\to\infty} \frac{1}{N}\sum_{n=0}^{N-1}\sum_{k=n}^{n-(N-1)} E\{x(n)x^*(n-k)\}e^{-j\omega k} \tag{4}$$

$$= \lim_{N\to\infty} \frac{1}{N}\sum_{n=0}^{N-1}\sum_{k=n}^{n-(N-1)} r(k)e^{-j\omega k} \tag{5}$$

$$= \lim_{N\to\infty} \frac{1}{N}\left(\sum_{n=0}^{k+N-1}\sum_{k=-(N-1)}^{-1} r(k)e^{-j\omega k} + \sum_{n=k}^{N-1}\sum_{k=0}^{N-1} r(k)e^{-j\omega k}\right) \tag{6}$$

$$= \lim_{N\to\infty} \frac{1}{N}\left(\sum_{k=-(N-1)}^{-1} (N+k)r(k)e^{-j\omega k} + \sum_{k=0}^{N-1} (N-k)r(k)e^{-j\omega k}\right) \tag{7}$$

$$= \lim_{N\to\infty} \frac{1}{N}\sum_{k=-(N-1)}^{N-1} (N-|k|)r(k)e^{-j\omega k} \tag{8}$$

$$= \lim_{N\to\infty} \sum_{k=-(N-1)}^{N-1} \left(1-\frac{|k|}{N}\right)r(k)e^{-j\omega k} \tag{9}$$

$$= \sum_{k=-\infty}^{\infty} r(k)e^{-j\omega k} - \lim_{N\to\infty} \frac{1}{N}\sum_{k=-N+1}^{N-1} |k|r(k)e^{-j\omega k} \tag{10}$$

According to the given mild assumption,

$$\lim_{N\to\infty} \frac{1}{N}\sum_{k=-N+1}^{N-1} |k||r(k)| = 0 \tag{11}$$

and

$$0 < \lim_{N\to\infty} \frac{1}{N}\sum_{k=-N+1}^{N-1} |k|r(k)e^{-j\omega k} < \lim_{N\to\infty} \frac{1}{N}\sum_{k=-N+1}^{N-1} |k||r(k)| \tag{12}$$

Hence

$$P(\omega) = \sum_{k=-\infty}^{\infty} r(k)e^{-j\omega k} \quad \text{definition 1} \tag{13}$$

This proves that if (11) is satisfied, the autocorrelation function (ACF) decays rapidly, PSD is the Discrete Time Fourier Transform (DTFT) of the ACF.

The following two figures compare the PSD of different signals derived by the two different definitions.

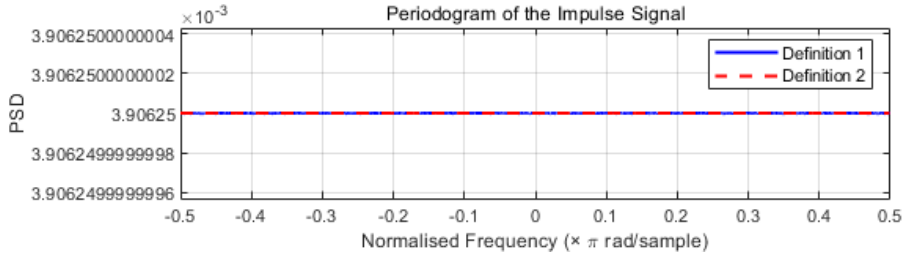

Fig. 1. Periodogram of an impulse signal.

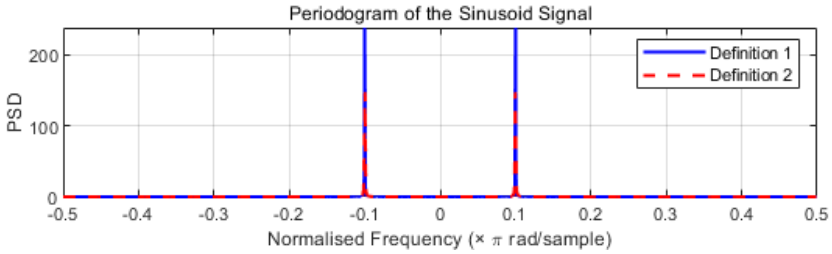

Fig. 2. Periodogram of a sinusoid signal.

In Fig. 1, the PSD of an impulse signal is found. The same results of the two definitions show that the two definitions are equivalent if ACF of the signal decays rapidly. In Fig. 2, the different PSDs prove that if the assumption is not satisfied, the equivalence of the two definitions does not hold.

1.2 Periodogram-based Methods Applied to Real-World Data

a) The sunspot data is pre-processed before the estimation. 0.1 is added to eliminate any zeros in the data series. This pre-processing allows the application of logarithm and does not affect the original sunspot data much. The reason for not choosing smaller values is that if a very small value is added (e.g. 1e-10), the mean of the logarithmic data can be significantly affected.

From the left figure in Fig. 3, the PSD of the pre-processed data has an extremely large value at around 0. The elimination of mean and trend from the data removes this peak and does not affect the result in other ranges. The estimation result of the centred and detrended data is shown alone in the middle figure and compared with that of the centred logarithmic data in the right figure. Despite the difference in magnitude, the shapes of the two estimation results are similar, while the logarithm removes the small peak at around 0.1. Also, only one harmonic is visible at around 0.17 in the middle figure, while two harmonics are shown in the right figure at around 0.17 and 0.26.

It can be concluded that the elimination of mean and trend removes the DC offset in the data series hence removes the large value around 0 in the periodogram. In addition, the precision of the periodicities of logarithmic data is higher than the centred and detrended data. Although there still exists a small peak at around 0.01, the interference peak at around 0.1 in the middle figure is removed and more harmonics are visible.

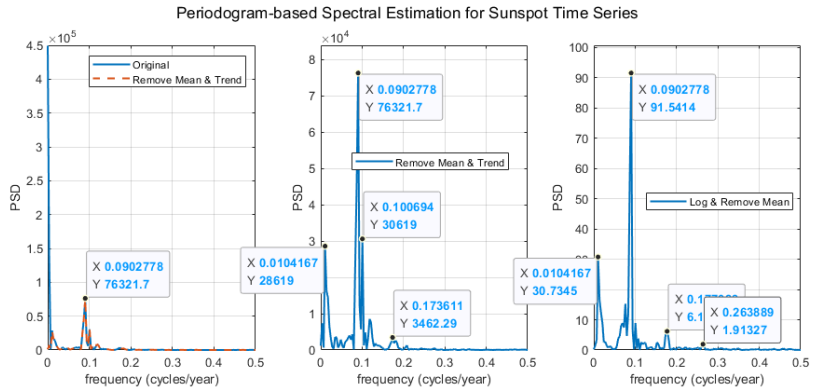

Fig. 3. Periodogram-based Spectral Estimation for Sunspot Time Series. (Left) Estimation results of the pre-processed data and the centred and detrended data. (Middle) Estimation result of the centred and detrended data alone. (Right) Estimation result of the centred logarithmic data.

b) The standard periodogram approach compared with the averaged periodogram of window length 10 s is shown in Fig. 4.

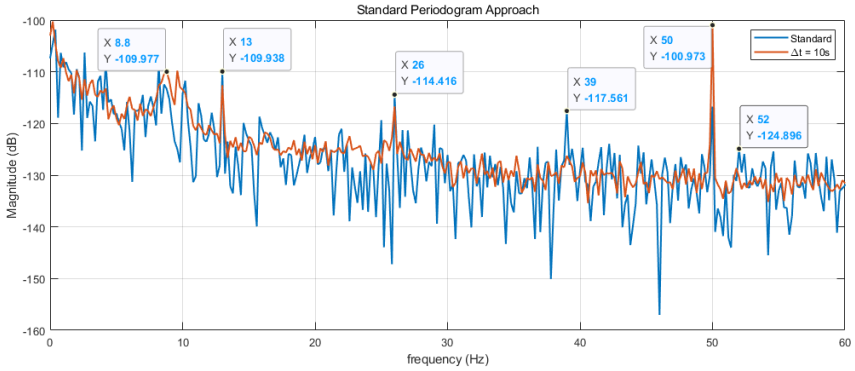

Fig. 4. Standard periodogram and averaged periodogram of window length 10 s.

The strong response within 8-10 Hz and the power-line interference at 50 Hz are visible from both approaches. The frequency of the flashing stimulus is 13 Hz, since three peaks exist at 13, 26, and 39 Hz. The peak at 13 Hz is the fundamental frequency response peak, and the other two peaks are the harmonics. There is also another harmonic at 52 Hz, which is only captured by the averaged periodogram.

The main difference between the two approaches is the magnitude of these peaks corresponding to SSVEP. Peaks derived by the standard periodogram approach are normally higher than that of the averaged periodogram of window length 10 s. This difference is obvious at 39 Hz, where the averaged periodogram approach gives a very low value. Also, the averaged periodogram fluctuates less than the standard periodogram does, which makes it easier to identify the estimated SSVEP peaks in the surrounding spectrum. This is because the averaged periodogram approach reduces the influence of noise hence reduces the variance of the result.

Theoretically, there should be another harmonic at 52 Hz. The peak at 52 Hz in the standard periodogram is hard to be determined as a harmonic since it has a similar value as the surrounding peaks. On the contrary, although the peak at 52 Hz in the averaged periodogram is very small, it is still identifiable.

Fig. 5 shows that as the decrease of window length, the result fluctuates less, while the peaks at 13 and 26 Hz become more difficult to be identified and the peaks at 39 and 52 Hz are not identifiable when the window length is reduced to 1s. Smaller window

length provides a smoother result making it easier to identify the estimated SSVEP peaks in the surrounding spectrum, while it also decreases the values of those peaks which makes the identification to be more difficult. The window length of 5 s is the most suitable one for this task, which clearly shows all the peaks and fluctuates less than the result using the window length of 10 s.

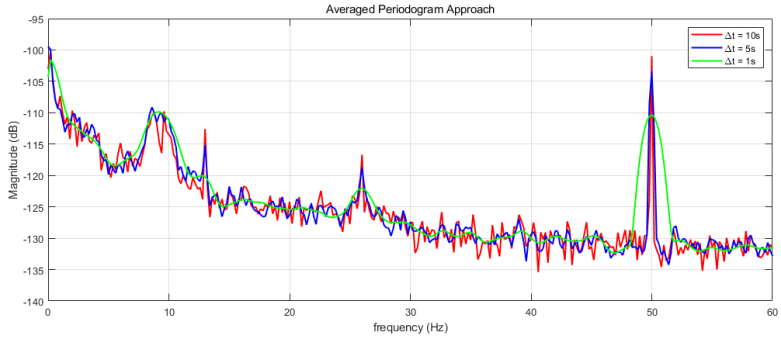

Fig. 5. Averaged periodogram of window length 10s, 5s, and 1s.

1.3 Correlation Estimation

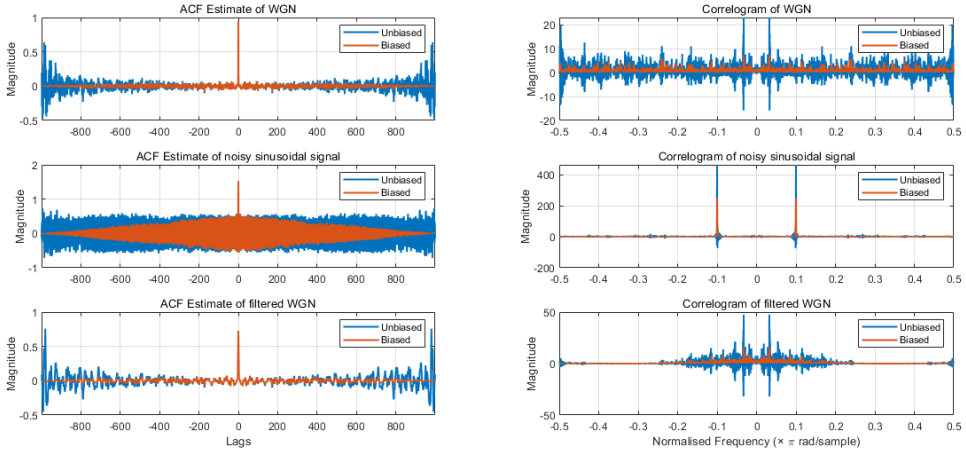

Fig. 6. ACF estimate and the correlogram for white Gaussian noise (WGN), noisy sinusoidal signal and filtered WGN.

a) The right figure in Fig. 6 shows the ACF computed by the unbiased and biased estimators for three kinds of signals. Results are similar for small lags (i.e. $|lags| < 200$). While if the absolute value of the lags is close to N (i.e. k is close to N), the biased ACF estimates approach 0, which shows large differences compared with the unbiased ACF estimates. This phenomenon exists in the processing of all three signals. Also, the unbiased ACF estimates are unstable when k approaches N, especially for the WGN and filtered WGN.

The influence of these differences is reflected in the computing of correlograms. The unbiased ACF estimates result in negative values of the estimated PSD, while the biased ACF estimates do not. The reason is the instability of the unbiased ACF estimates leads to fewer available samples used to estimate PSD when k is close to N. In this case, the positive definiteness of ACF estimates may lose hence resulting in negative PSD estimates.

b) The left figure in Fig. 7 shows the required 100 realisations with their mean. A sinusoidal signal is corrupted by i.i.d. WGN with a variance of 0.2. It can be found that the spread out of the 100 iterations is not clear in this figure. All the iterations surround the mean closely. The figure on the right illustrates the standard deviation of the 100 realisations. The standard deviation is proportional to the PSD with the same peaks at

the same frequencies. As mentioned in the handout, this leads to the increase in confidence interval where the PSD increases.

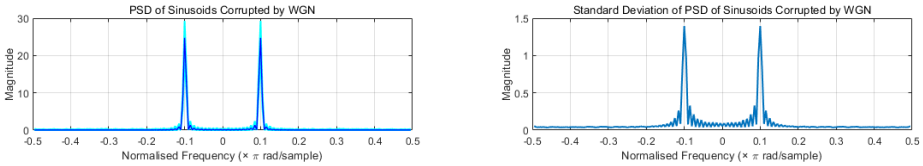

Fig. 7. (Left) 100 realisations of the PSD of a sinusoid corrupted by i.i.d. WGN and the mean. (Right) Standard deviation of the 100 realisations.

c) Fig. 8 is replotted by changing the unit of the magnitude of PSD in Fig. 7 into dB. The spread out of the original magnitudes that are larger than 1 is decreased and is increased for the original magnitudes that are smaller than 1. This representation keeps showing the peaks of the PSD representing the frequency of the sinusoidal signal, and meanwhile shows how the different WGN affects the PSD estimation in the other frequency band.

As for the standard deviation on the right, in Fig. 8 it shows an opposite trend as that shown in Fig. 7. By plotting the PSD in dB and calculating the standard deviation after that, the standard deviation is inversely proportional to the magnitude of PSD. In this case, the confidence interval is shrunk when the PSD increases.

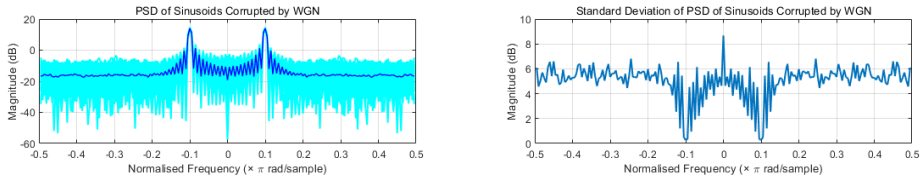

Fig. 8. (Left) 100 realisations of the PSD in dB of a sinusoid corrupted by i.i.d. WGN and the mean. (Right) Standard deviation in dB of the 100 realisations.

d) In Fig. 9, the periodogram of the given signal with different lengths is illustrated. The periodogram estimated using the signal with a length of 20 or 30 fails to distinguish the two frequencies in the spectrum, since the resolution of the periodogram is greater than the separation between the two frequencies. If the length of the signal is 40, the two frequencies are identifiable, and the result is better if it is increased to 50 or 60.

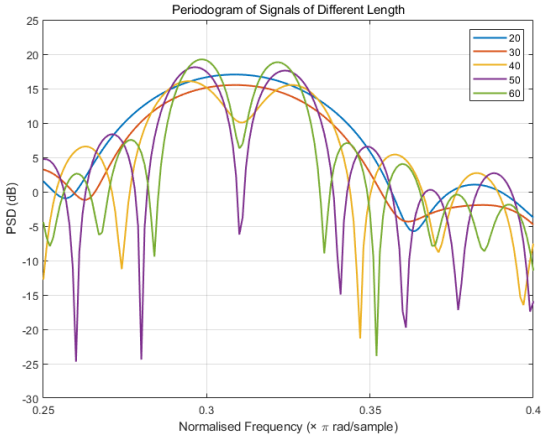

Fig. 9. Periodogram of two complex exponentials with frequencies of 0.3 and 0.32 Hz and additive complex WGN.

e) Function 'corrmtx()' returns a rectangular Toeplitz matrix **H** and the autocorrelation matrix estimate **R** of the input signal x. The second input $m = 14$ is the order of the prediction model used to derive **H**, and the third input "mod" chooses the method of computing **H** as "modified". This modified rectangular Toeplitz matrix **H** with a size

of $2(n-m)$-by-$(m+1)$ is derived by using the forward and backward prediction error estimates on the basis of the m^{th}-order prediction model. Then, the biased autocorrelation matrix **R** is computed as $\mathbf{H}^{\dagger}\mathbf{H}$, where $\mathbf{H}^{\dagger}$ is the conjugate transpose of **H**.

The autocorrelation matrix **R** is then inputted to the second function 'pmusic()' which returns the pseudospectrum of the original signal using the multiple signal classification (MUSIC) algorithm. $p = 2$ specifies the subspace dimension of the MUSIC algorithm used to compute the pseudospectrum. "[ ]" set the number of DFT points as the default 256. "1" is the sample rate, and "corr" tells the function to interpret the input **R** as a correlation matrix, otherwise **R** will be considered as a signal. The output **S**, the pseudospectrum, is computed based on the estimates of the eigenvectors of **R**, and **F** is the output normalized frequency.

As in Fig. 10, the third line of the code plots the computed pseudospectrum **S** versus the normalized frequency **F**. The width of line is set as 2 and only frequencies in the range of 0.25 to 0.4 are shown. The standard division of 100 realisations is also shown.

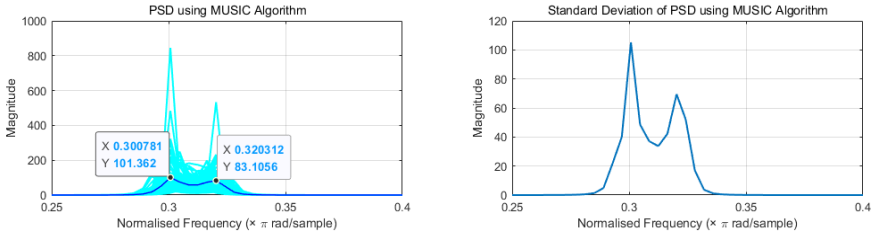

Fig. 10. (Left) 100 realisations of the spectrum estimated using MUSIC algorithm of two complex exponentials with frequencies of 0.3 and 0.32 Hz and additive complex WGN. The length of signal is 30. (Right) Standard deviation of the 100 realisations.

Different from the periodogram in Fig. 9, MUSIC algorithm successfully distinguishes the two frequencies when the length of the input signal is 30. This implies that the MUSIC algorithm has a higher resolution and lower bias comparing with the periodogram when the input signal is short in length. However, the standard division is proportional to the PSD and cannot be fixed by plotting in dB. The confidence interval is increased when the PSD increases. In addition, the MUSIC algorithm uses more computational resources than the periodogram does. The choice of its dimension is a trade-off between complexity and accuracy.

1.4 Spectrum of Autoregressive Processes

a) According to the Yule-Walker equations, the set of autoregressive (AR) parameters $\boldsymbol{a} = [a_1, \dots, a_p]^T$ can be computed as

$$\boldsymbol{r_{xx}} = \boldsymbol{R_{xx}}\boldsymbol{a} \quad \Rightarrow \quad \boldsymbol{a} = \boldsymbol{R_{xx}^{-1}}\boldsymbol{r_{xx}} \tag{14}$$

where $\boldsymbol{r_{xx}}$ can be computed using the autocorrelation estimate, $\boldsymbol{R_{xx}}$ is the ACF matrix.

As shown in (14), the ACF matrix $\boldsymbol{R_{xx}}$ should be invertible, hence it should be positive definite to ensure its invertibility. As discussed in section 1.3, the instability of the unbiased ACF estimates may influence the positive definiteness of the ACF estimates, hence $\boldsymbol{R_{xx}}$ may be not invertible. Therefore, it is not appropriate to use the unbiased ACF as the estimate in finding AR parameters.

b) AR parameters are estimated using the Yule-Walker Method. The smallest model order that can identify the two peaks in the spectrum is 6. The result is shown in the left figure of Fig. 11, compared with the true PSD of the signal. The right figure in Fig. 11 shows how the estimation error changes with the increase of model order, which also demonstrates that model order 6 is the best choice. If the order is lower than 6, the two

peaks cannot be identified. If the order is higher than 6, the complexity of the algorithm and the instability in the high frequency band are increased.

However, the order of the original signal is actually 4, not 6, while the model with order 4 fails to identify the two peaks in the spectrum. This might be the influence of randomness caused by the small number of samples (1000) used for estimation.

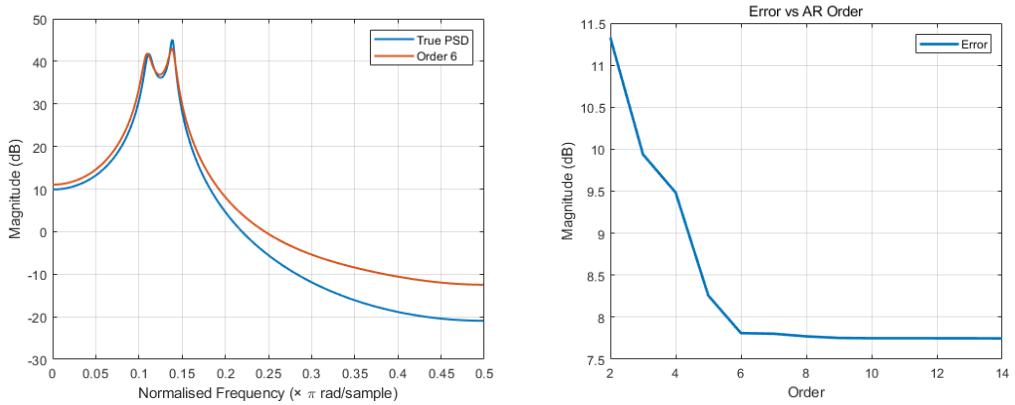

Fig. 11. (Left) Estimated PSD of the given signal (with a length of 1000) using AR model with order 6. (Right) The estimation error comparing with the true PSD versus the model order.

c) In order to eliminate the difference between the true model order and the best model order estimated in b), the length of signal used for estimation is increased to 10,000. The results are shown in Fig. 12.

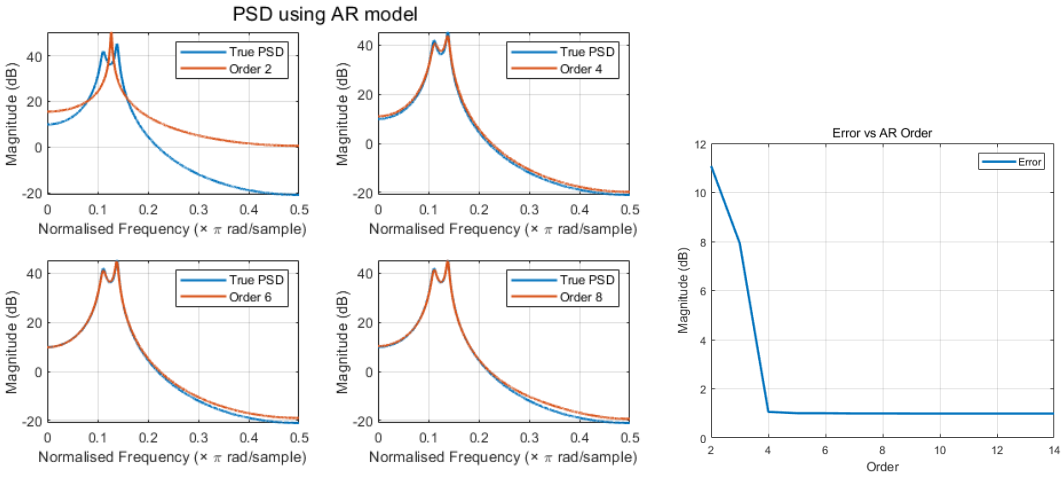

Fig. 12. (Left) Estimated PSD of the given signal (with a length of 10,000) using AR model with orders 2, 4, 6, and 8. (Right) The estimation error comparing with the true PSD versus the model order.

As shown in Fig. 12, the model with order 4 successfully estimates the PSD. Similar to b), if the model order is smaller than 4, the estimation fails. If the model order is higher than 4, the estimation error in high frequency band is increased.

During the experiment, the signal with a length of 10,000 still cannot fully overcome the influence of randomness. Generally, model order 4 cannot estimate PSD well. This can be solved by increasing the signal length further (e.g. 100,000).

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

a) The periodograms of the RRI data derived by using the standard and averaged methods are plotted in Fig. 13.

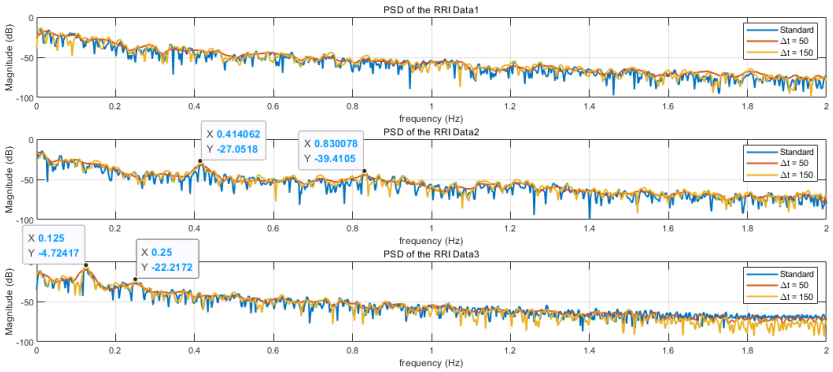

Fig. 13. Standard periodogram and the averaged periodogram with window lengths 50s and 150s for the RRI data.

b) In the figure for the RRI data 1, it is hard to identify the location of the peak. There does not exist a peak that can be distinguished in the surrounding environment. In the middle figure, a peak is identified at around 0.41, and a harmonic is found at around 0.83. In the last figure, a peak is identified at 0.125, and a harmonic is found at 0.25. According to the location of peaks, it can be concluded that the RRI data 2 is at a higher frequency than the RRI data 3.

The precision of using the standard periodogram and the averaged periodogram with different window lengths is similar to what has been discussed in section 1.2 b). The averaged periodogram gives a smoother result than the standard periodogram does. If the window length is too small, it becomes difficult to identify the peaks in the surrounding spectrum. As shown in the figure, the result derived by using window length 50s is worse than using window length 150s.

c) For the RRI data 1 in Fig. 14, the appropriate AR model order to fit the data is not found. Peaks cannot be found using small model orders while the model overfits the noise when high model orders are used.

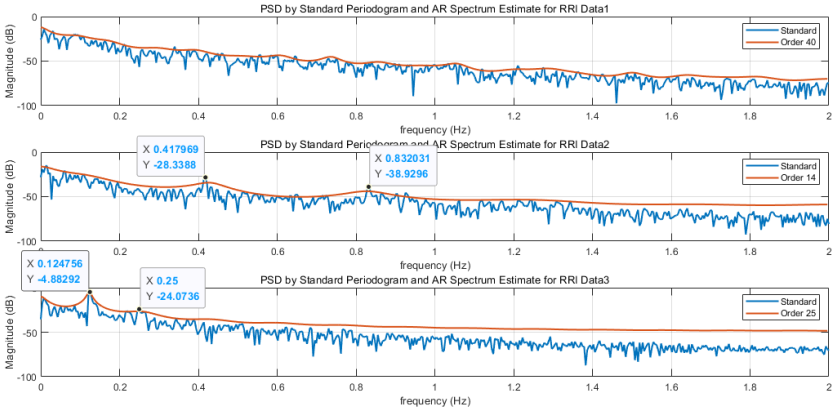

Fig. 14. Standard periodogram and the PSD estimated by AR model for the RRI data.

The AR modelling of the remaining two sets of RRI data is successful. With model order 14, the AR spectrum estimate for RRI data 2 shows two similar peaks as the standard periodogram at around 0.41 and 0.83. With model order 25, the AR spectrum estimate for RRI data 3 shows two similar peaks as the standard periodogram at around 0.124 and around 0.25.

The main difference between the AR spectrum and the periodogram estimate is the shape of the spectrum. The periodogram is influenced by noise, hence it is not easy to identify the peaks in the surrounding spectrum. The AR Spectrum is smooth and shows the location of peaks without the interference of the surrounding spectrum.

The problem of AR modelling is that it does not show the peaks as sharp as the periodogram does. In addition, the choice of model order for the AR modelling is important. If the order is too large, the model might overfit the noise, while if the order is too small, the AR estimation cannot identify peaks.

1.6 Robust Regression

a) The singular values of $\boldsymbol{X}$ and $\boldsymbol{X}_{noise}$ are plotted in Fig. 15. Using the function "rank" in Matlab, the rank of $\boldsymbol{X}$ is 3 and the rank of $\boldsymbol{X}_{noise}$ is 10. This difference in rank is reflected in the figure. The first three singular values of $\boldsymbol{X}$ are nonzero and those of $\boldsymbol{X}_{noise}$ are slightly larger. While different from the remaining zero-valued singular values of $\boldsymbol{X}$, the singular values of $\boldsymbol{X}_{noise}$ have about half of the magnitude of the three dominant singular values. These nonzero singular values are considered in the determination of rank, hence the rank of $\boldsymbol{X}_{noise}$ is 10.

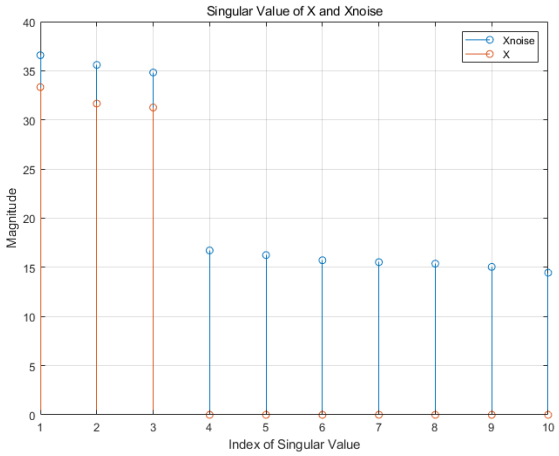

Fig. 15. Singular values of $\boldsymbol{X}$ and $\boldsymbol{X}_{noise}$.

b) As shown in Fig. 16, the square error between $\boldsymbol{X}$ and $\boldsymbol{X}_{noise}$ reflects the corruption of noise, and the square error between $\boldsymbol{X}$ and $\widetilde{\boldsymbol{X}}_{noise}$ shows the performance of the low-rank approximation of $\boldsymbol{X}_{noise}$. The choice of r used to estimate $\widetilde{\boldsymbol{X}}_{noise}$ is 3, since $\boldsymbol{X}$ has three principal components. The figure shows that the low-rank approximation successfully decreases the influence of noise, since only the first three elements have useful information while the remaining elements only contain noise.

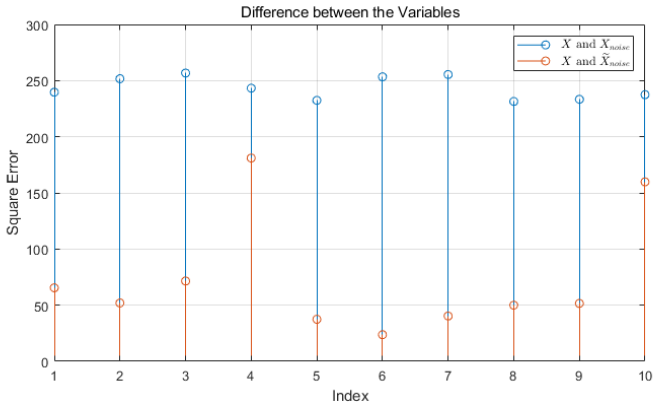

Fig. 16. Difference between $\boldsymbol{X}$ and $\boldsymbol{X}_{noise}$, and $\boldsymbol{X}$ and $\widetilde{\boldsymbol{X}}_{noise}$.

c) The parameter matrix $\boldsymbol{B}$ in ordinary least squares (OLS) is estimated using $\boldsymbol{X}_{noise}$ to get a full-rank matrix. The result is shown in Tab. 1. The principal component regression (PCR) solution has slightly higher errors than the OLS solution in all elements.

Tab. 1. Estimation error between $\boldsymbol{Y}$ and $\widehat{\boldsymbol{Y}}_{OLS}$ and $\widehat{\boldsymbol{Y}}_{PCR}$.

Index	1	2	3	4	5	Sum
OLS Error	1160.9	610.6	572.7	383.2	824.3	3551.7
PCR Error	1168.2	616.3	578.8	386.6	827.2	3577.1

By applying the regression coefficients computed from the training set to the estimation of the testing set, the result is shown in Tab. 2. PCR performs slightly better than OLS using the testing set, and both PCR and OLS perform better than using the training set. OLS error is decreased by 33.1%, and PCR error is decreased by 34.2%. These decreases in error show that the estimation of parameter matrix $\boldsymbol{B}$ is successful.

Tab. 2. Estimation error between $\boldsymbol{Y}_{test}$ and $\widehat{\boldsymbol{Y}}_{test-OLS}$ and $\widehat{\boldsymbol{Y}}_{test-PCR}$.

Index	1	2	3	4	5	Sum
OLS Error	963.8	390.1	343.2	131.6	548.8	2377.5
PCR Error	959.2	386.1	334.5	129.3	544.8	2353.9

d) Using the given function "regval", 1000 new realisations of the test data $\boldsymbol{Y}_{test}$ and its estimate $\widehat{\boldsymbol{Y}}_{test-OLS}$ and $\widehat{\boldsymbol{Y}}_{test-PCR}$ are used to compute the mean square error (MSE). The result is shown in Tab. 3. Compared with Tab. 2, OLS error and PCR error are decreased by 0.88% and 0.64%, respectively.

Tab. 3. MSE estimates for the PCR and OLS schemes based on $\boldsymbol{Y}_{test}$ and $\widehat{\boldsymbol{Y}}_{test-OLS}$ and $\widehat{\boldsymbol{Y}}_{test-PCR}$ provided by function "regval".

Index	1	2	3	4	5	Sum
OLS Error	931.9	395.1	333.0	135.3	561.2	2356.5
PCR Error	927.8	390.3	329.0	133.4	558.3	2338.8

2. Adaptive signal processing

2.1 The Least Mean Square (LMS) Algorithm

a) Given the input vector

$$\boldsymbol{x}(n) = [x(n-1), x(n-2)]^T \tag{15}$$

The correlation matrix $\boldsymbol{R}_{\boldsymbol{xx}}$ can be written as

$$\boldsymbol{R}_{\boldsymbol{xx}} = E\{\boldsymbol{x}(n)\boldsymbol{x}(n)^T\} \tag{16}$$

$$= E\begin{Bmatrix} x(n-1)^2 & x(n-1)x(n-2) \\ x(n-1)x(n-2) & x(n-2)^2 \end{Bmatrix} \tag{17}$$

$$= \begin{bmatrix} r_x(0) & r_x(1) \\ r_x(1) & r_x(0) \end{bmatrix} \tag{18}$$

where $r_x(k)$ is the k^{th} element of the ACF of $\boldsymbol{x}(n)$.

According to the property of ACF,

$$r_x(k) = E\{x(n)x(n-k)\} \tag{19}$$

$$= E\{[a_1x(n-1) + a_2x(n-2) + \eta(n)]x(n-k)\} \tag{20}$$

$$= a_1E\{x(n-1)x(n-k)\} + a_2E\{x(n-2)x(n-k)\} + E\{\eta(n)x(n-k)\} \tag{21}$$

Substitute $k = 0, 1, 2$ into (21),

$$\begin{cases} r_x(0) = a_1 r_x(1) + a_2 r_x(2) + {\sigma_\eta}^2 \\ r_x(1) = a_1 r_x(0) + a_2 r_x(1) \\ r_x(2) = a_1 r_x(1) + a_2 r_x(0) \end{cases} \tag{22}$$

Substitute $a_1 = 0.1, a_2 = 0.8, {\sigma_\eta}^2 = 0.25$ into (22), the solution of this equation set is

$$\begin{cases} r_x(0) = \dfrac{25}{27} \approx 0.926 \\ r_x(1) = \dfrac{25}{54} \approx 0.463 \\ r_x(2) = \dfrac{85}{108} \approx 0.787 \end{cases} \tag{23}$$

Substitute $r_x(0), r_x(1)$ into (18), the correlation matrix $\boldsymbol{R_{xx}}$ is

$$\boldsymbol{R_{xx}} = \begin{bmatrix} 0.926 & 0.463 \\ 0.463 & 0.926 \end{bmatrix} \tag{24}$$

Eigen vales of the matrix $\boldsymbol{R_{xx}}$ are $\lambda_1 = 0.463$ and $\lambda_2 = 1.389$, hence the maximum eigenvalue $\lambda_{max} = 1.389$.

According to the LMS stability bound,

$$0 < \mu < \frac{2}{\lambda_{max}} \tag{25}$$

Therefore, μ should be in $(0, 1.440)$ to allow the LMS to converge in the mean.

b) As shown in the top figure in Fig. 17, it is difficult to identify how the prediction error changes from only one realisation due to the randomness. By averaging 100 different realisations, the learning curve is observed in the bottom figure in Fig. 17. It is shown that the learning curve converges faster when a larger step size is used. The result of using $\mu = 0.05$ converges at around 80^{th} sample, while the result of using $\mu = 0.01$ converges at around 270^{th} sample. In addition, the learning curve using $\mu = 0.01$ has a slightly lower steady state error. This is reasonable since a larger step size not only increases the converging speed but the instability of the result.

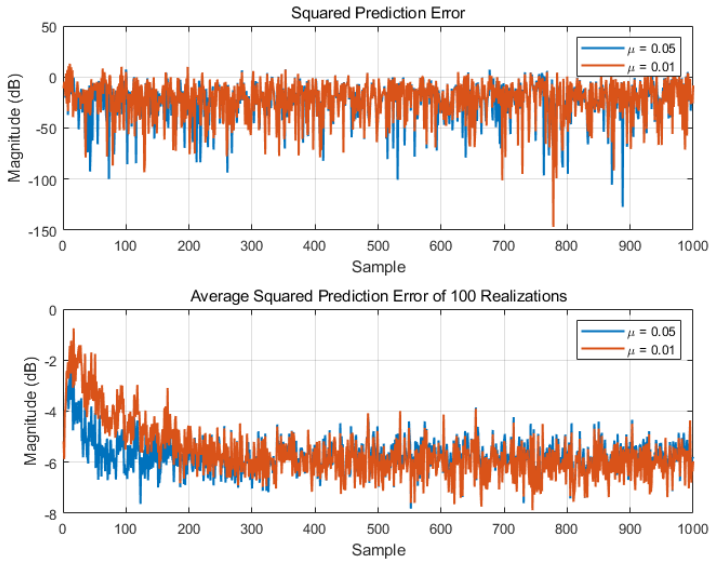

Fig. 17. (Top) Squared prediction error of the LMS adaptive predictor using different step sizes. (Bottom) Learning curve of the LMS adaptive predictor using different step sizes by averaging the squared prediction errors from 100 different realisations.

c) The estimated misadjustment and theoretical misadjustment are shown in Tab. 4. The estimated misadjustment is derived by taking the last 500 samples in the learning curve. There exist a 12.97% difference between the two misadjustments for step size 0.05, and a 9.71% difference for step size 0.01. These show that the estimation is successful.

Tab. 4. Estimated misadjustment and theoretical misadjustment.

Step Size	Estimated Misadjustment	Theoretical Misadjustment	Difference
0.05	0.0532	0.0463	12.97%
0.01	0.0103	0.0093	9.71%

d) By taking the average value of 100 realisations of the adaptive filter coefficients, the updating curves of the two coefficients are shown in Fig. 18.

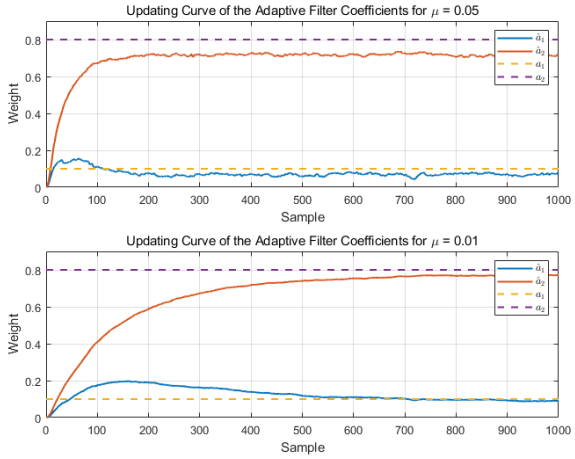

Fig. 18. The updating curve of the adaptive filter coefficients for step size 0.05 (Top) and step size 0.01 (Bottom).

Taking the average value of the last 200 samples of the coefficients, their steady state values are shown in Tab. 5. Obviously, although the converging speed of using step size 0.05 is faster than using step size 0.01 as shown in Fig. 18, the estimation precision of using a smaller step size is higher. The smaller step size provides a much smaller difference compared with the true value.

Tab. 5. Steady state values of the adaptive filter coefficients for step sizes of $\mu = 0.05$ and $\mu = 0.01$.

Step Size	a_1	Error	a_2	Error
0.05	0.0736	26.40%	0.7121	10.99%
0.01	0.0961	3.90%	0.7652	4.35%

e) The cost function is written as

$$J_2(n) = \frac{1}{2}\left(e^2(n) + \gamma\|\boldsymbol{w}(n)\|_2^2\right) \tag{26}$$

$$= \frac{1}{2}\left(e^2(n) + \gamma\boldsymbol{w}(n)^T\boldsymbol{w}(n)\right) \tag{27}$$

Taking the gradient of the cost function,

$$\nabla_{\boldsymbol{w}(n)}J_2(n) = \frac{1}{2}\underbrace{\frac{\partial e^2(n)}{\partial e(n)}}_{2e(n)}\underbrace{\frac{\partial e(n)}{\partial y(n)}}_{-1}\underbrace{\frac{\partial y(n)}{\partial \boldsymbol{w}(n)}}_{\boldsymbol{x}(n)} + \frac{\gamma}{2}\underbrace{\frac{\partial\left(\boldsymbol{w}(n)^T\boldsymbol{w}(n)\right)}{\partial \boldsymbol{w}(n)}}_{2\boldsymbol{w}(n)} \tag{28}$$

$$= -e(n)\boldsymbol{x}(n) + \gamma\boldsymbol{w}(n) \tag{29}$$

Substitute this result into the equation of the steepest descent algorithm,

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu\left[-\nabla J|_{\boldsymbol{w}=\boldsymbol{w}(n)}\right] \tag{30}$$

$$= \boldsymbol{w}(n) + \mu\left[-\nabla_{\boldsymbol{w}(n)} J_2(n)\right] \tag{31}$$

$$= \boldsymbol{w}(n) - \mu[-e(n)\boldsymbol{x}(n) + \gamma \boldsymbol{w}(n)] \tag{32}$$

$$= (1 - \mu\gamma)\boldsymbol{w}(n) + \mu e(n)\boldsymbol{x}(n) \tag{33}$$

The leaky LMS equation is shown in (33).

f) From Tab. 6, the estimation of the first adaptive filter coefficient a_1 using the step size of $\mu = 0.05$ is successful under all the leakage. On the contrary, the estimation of a_1 with $\mu = 0.01$ and the estimation of a_2 with any step size are significantly different from the true value. There also exists a trend in the estimation of a_2, the absolute error increases as the increase of the value of leakage.

Tab. 6. Steady state values of the adaptive filter coefficients estimated from leaky LMS algorithm with different γ using step sizes of $\mu = 0.05$ and $\mu = 0.01$.

Leakage	Step Size	a_1	Error (%)	a_2	Error (%)
0.1	0.05	0.0905	-9.5	0.6112	-23.6
	0.01	0.1180	18.0	0.6723	-16.0
0.3	0.05	0.1055	5.5	0.4820	-39.8
	0.01	0.1355	35.5	0.5436	-32.1
0.5	0.05	0.1063	6.3	0.4013	-49.8
	0.01	0.1365	36.5	0.4586	-42.7
0.7	0.05	0.1028	2.8	0.3457	-56.8
	0.01	0.1317	31.7	0.3977	-50.3
0.9	0.05	0.0982	-1.8	0.3047	-61.9
	0.01	0.1252	25.2	0.3517	-56.0

Generally speaking, the measurement error increases as the increase of leakage, especially for the larger coefficient a_2. The reason can be explained as follow.

The Wiener-Hopf solution of the optimal set of weights is

$$\boldsymbol{w}_{opt} = \boldsymbol{R}_{xx}^{-1}\boldsymbol{p} \tag{34}$$

where $\boldsymbol{R}_{\boldsymbol{xx}} = E\{\boldsymbol{x}(n)\boldsymbol{x}(n)^T\}$ and $\boldsymbol{p} = E\{d(n)\boldsymbol{x}(n)\}$.

When the leakage is introduced, the optimal set of weights becomes

$$\boldsymbol{w}_{leak-opt} = (\boldsymbol{R}_{\boldsymbol{xx}} + \gamma \boldsymbol{I})^{-1}\boldsymbol{p} \tag{35}$$

(35) guarantees that there are no zero-valued eigenvalues in $\boldsymbol{R}_{\boldsymbol{xx}}$, hence it stabilizes the LMS adaptive filter.

However, as shown in Tab. 6, the leaky LMS algorithm does not converge to the true value of the coefficients. The reason is that there exists a difference between the optimal weights derived by (34) and (35). This difference increases with the increase of γ. A more intuitive explanation is shown in (33). The value of γ influences the weight update.

2.2 Adaptive Step Sizes

a) The main difference between the gradient adaptive step-size (GASS) algorithm and the LMS algorithm is the step size. As analysed in the previous sections, it is difficult to choose a step size for the LMS algorithm to balance the steady state error and the

convergence speed. The advantage of GASS algorithm is the flexibility in step size. A larger step size is used at the start of the weight update to speed up the convergence, while a smaller step size is used to decrease the steady state error when the estimated parameters are close to the true value. The disadvantage of the GASS is the computational complexity. It requires updating the step size at every iteration during the estimation.

The weight error curves of the three GASS algorithms and the standard LMS algorithm are shown in Fig. 19. From the figure, GASS algorithms introduced by Benveniste, Ang & Farhang ($\alpha = 0.6$), and Matthews & Xie are ranked in the order of convergence speeds from fast to slow. Their convergence speeds are also determined by the step size ρ (the step size used for updating μ). They converge faster with a larger value of ρ.

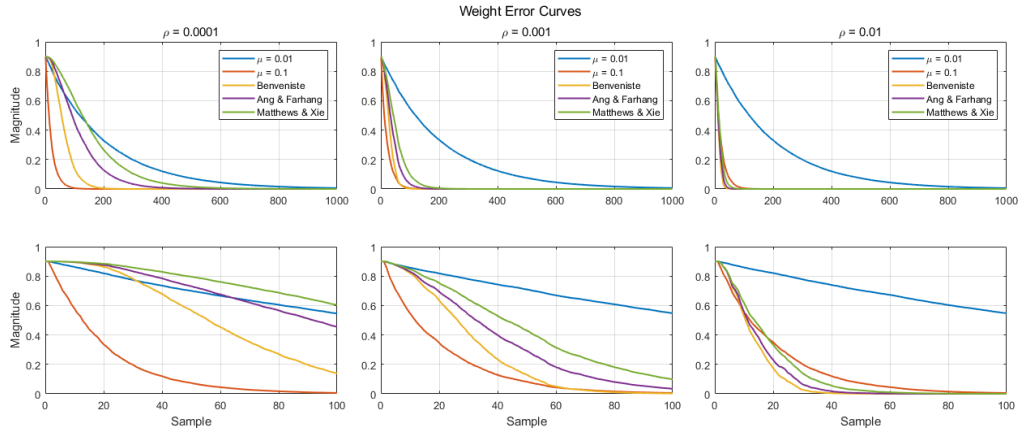

Fig. 19. Weight error curves of the three GASS algorithms and the standard LMS algorithm. (Top) Full 1000 samples. (Bottom) The first 100 samples.

Tab. 7 shows the steady state error of the estimation results in Fig. 19. Large value of ρ does not result instability in the estimation. The mean squared steady state error of three GASS algorithms converges to the same level as LMS with $\mu = 0.1$. The reason for the high error in LMS with $\mu = 0.01$ and three GASS algorithms with small ρ is that they have not converged to the optimal value after 800 iterations.

Tab. 7. Mean squared steady state error in dB of the last 200 samples of the weight estimated in Fig. 19.

ρ	Benveniste	Ang & Farhang	Matthews & Xie	LMS $\mu = 0.01$	LMS $\mu = 0.1$
0.01	-302.19	-302.15	-302.19	-39.56	-302.19
0.005	-302.19	-233.83	-203.24	-39.56	-302.19
0.001	-239.11	-77.6	-66.30	-39.56	-302.19

b) According to the update equation,

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu e_p(n)\boldsymbol{x}(n) \tag{36}$$

$$\boldsymbol{x}^T(n)\boldsymbol{w}(n+1) = \boldsymbol{x}^T(n)\boldsymbol{w}(n) + \mu e_p(n)\boldsymbol{x}^T(n)\boldsymbol{x}(n) \tag{37}$$

$$d(n) - \boldsymbol{x}^T(n)\boldsymbol{w}(n+1) = d(n) - \boldsymbol{x}^T(n)\boldsymbol{w}(n) - \mu e_p(n)\|\boldsymbol{x}(n)\|^2 \tag{38}$$

The a posteriori error and a priori error are expressed as

$$e_p(n) = d(n) - \boldsymbol{x}^T(n)\boldsymbol{w}(n+1) \tag{39}$$

$$e(n) = d(n) - \boldsymbol{x}^T(n)\boldsymbol{w}(n) \tag{40}$$

Substitute (39) and (40) into (38),

$$e_p(n) = e(n) - \mu e_p(n)\|\boldsymbol{x}(n)\|^2 \tag{41}$$

$$= \frac{e(n)}{1 + \mu\|\boldsymbol{x}(n)\|^2} \tag{42}$$

Substitute (42) into (36),

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu \frac{e(n)}{1 + \mu\|\boldsymbol{x}(n)\|^2}\boldsymbol{x}(n) \tag{43}$$

$$\triangle \boldsymbol{w}(n) = \frac{\mu e(n)}{1 + \mu\|\boldsymbol{x}(n)\|^2}\boldsymbol{x}(n) \tag{44}$$

$$= \frac{e(n)\boldsymbol{x}(n)}{\frac{1}{\mu} + \|\boldsymbol{x}(n)\|^2} \tag{45}$$

In comparison, the weight update of the normalized LMS (NLMS) is

$$\triangle \boldsymbol{w}(n) = \frac{\beta e(n)\boldsymbol{x}(n)}{\epsilon + \|\boldsymbol{x}(n)\|^2} \tag{46}$$

Therefore, if $\beta = 1$ and $\epsilon = \frac{1}{\mu}$, (45) and (46), the update equation based upon the a posteriori error $e_p(n)$ in (39) and the update equation of the NLMS algorithm, are equivalent.

c) The results of the weight estimates using the generalized normalized gradient descent (GNGD) algorithm and Benveniste's GASS algorithm are shown in Fig. 20. The initial learning rate and the step size of the GASS are 0 and 0.01, of the GNGD are 5 and 0.01.

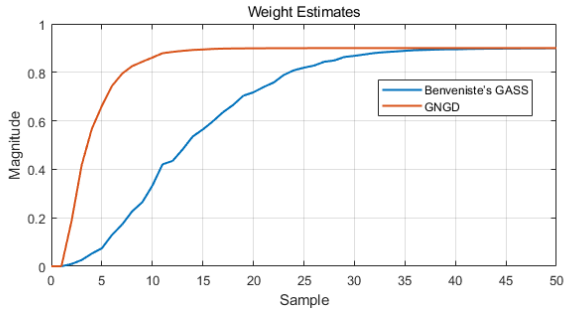

Fig. 20. Weight estimates using Benveniste's GASS algorithm and GNGD algorithm with different initial learning rates.

The advantage of GNGD is that it can use a large initial learning rate to speed up the convergence without being unstable. As shown in the figure, GNGD with an initial learning rate of 2 converges much faster than GASS with an initial learning rate of 0.

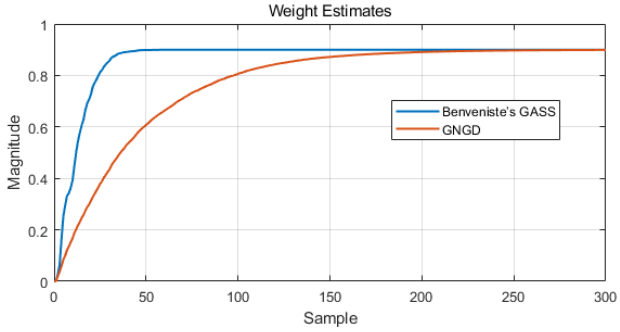

Fig. 21. Weight estimates using Benveniste's GASS algorithm and the GNGD algorithm with the same initial learning rates.

If the same learning rates are used ($\mu = 0.05$), as shown in Fig. 21, Benveniste's GASS algorithm converges faster than GNGD. However, if the initial learning rate is increased to get a higher convergence speed (e.g. $\mu = 5$ for GNGD in Fig. 20), Benveniste's GASS algorithm would be unstable. This difference is caused by the nonlinear updating of the learning rate in GNGD and the linear updating of the learning rate in GASS.

The computational complexity can be derived by computing the number of additions and multiplications. For the model order p, there exist $p + 1$ elements in the weight matrix. The measurement of computational complexity is shown in Tab. 8, which shows the computational complexity of GASS is $O(P^2)$, while that of GNGD is $O(P)$. GNGD not only performs better than GASS in the convergence speed but also in the saving of computational resources.

Tab. 8. Computational complexity (for one sample) of Benveniste's GASS algorithm and GNGD, where let $P = p + 1$.

Steps	GASS		GNGD	
	$\times, \div$	$+, -$	$\times, \div$	$+, -$
Predict the signal	P	$P-1$	P	$P-1$
Calculate the error	0	1	0	1
Update the weight	$P+1$	P	$2P+1$	$2P$
Calculate cost function	$3P^2+P$	$3P^2-P$	$2P+6$	$2P$
Update step-size	$P+2$	P	0	0
Sum	$3P^2+4P+3$	$3P^2+2P$	$5P+7$	$4P$

2.3 Adaptive Noise Cancellation

a) In Adaptive Line Enhancer (ALE), the MSE, noise-corrupted signal, coloured noise generated using the moving average filter, estimated clean signal, and predictor input with filter length M are expressed as

$$MSE = E\left\{\left(s(n) - \hat{x}(n)\right)^2\right\} \tag{47}$$

$$s(n) = x(n) + \eta(n) \tag{48}$$

$$\eta(n) = v(n) + 0.5v(n-2) \tag{49}$$

$$\hat{x}(n) = \boldsymbol{w}^T(n)\boldsymbol{u}(n) \tag{50}$$

$$u_i(n) = s(n - \Delta - i + 1), \;\; i = 1, 2, \dots, M \tag{51}$$

According to (47) and (48),

$$MSE = E\left\{\left(s(n) - \hat{x}(n)\right)^2\right\} \tag{52}$$

$$= E\left\{\left(x(n) + \eta(n) - \hat{x}(n)\right)^2\right\} \tag{53}$$

$$= E\left\{\left(x(n) - \hat{x}(n)\right)^2\right\} + 2E\left\{\left(x(n) - \hat{x}(n)\right)\eta(n)\right\} + E\{\eta^2(n)\} \tag{54}$$

$$= E\left\{\left(x(n) - \hat{x}(n)\right)^2\right\} + 2E\{x(n)\eta(n)\} - 2E\{\hat{x}(n)\eta(n)\} + E\{\eta^2(n)\} \tag{55}$$

In (55), the first term is irrelevant to the noise, the second term is 0 since the signal $x(n)$ is uncorrelated to the noise, and the last term is a constant which is the noise power.

Therefore, the problem of minimising MSE becomes the problem of minimising the absolute value of $E\{\hat{x}(n)\eta(n)\}$.

According to (48), (50), and (51),

$$E\{\hat{x}(n)\eta(n)\} = E\{\boldsymbol{w}^T(n)\boldsymbol{u}(n)\eta(n)\} \tag{56}$$

$$= E\left\{\eta(n)\sum_{i=1}^{M} w_i(n)u_i(n)\right\} \tag{57}$$

$$= E\left\{\eta(n)\sum_{i=1}^{M} w_i(n)s(n-\Delta-i+1)\right\} \tag{58}$$

$$= E\left\{\eta(n)\sum_{i=1}^{M} w_i(n)\big(x(n-\Delta-i+1)+\eta(n-\Delta-i+1)\big)\right\} \tag{59}$$

$$= E\left\{\eta(n)\sum_{i=1}^{M} w_i(n)x(n-\Delta-i+1)\right\} + E\left\{\eta(n)\sum_{i=1}^{M} w_i(n)\eta(n-\Delta-i+1)\right\} \tag{60}$$

Since $\eta(n)$ is not correlated to $x(n-\Delta-i+1)$, the first term of (60) is 0.

Then, substitute (49) into (60),

$$E\left\{\sum_{i=1}^{M} w_i(n)\big(v(n)+0.5v(n-2)\big)\big(v(n-\Delta-i+1)+0.5v(n-\Delta-i-1)\big)\right\} \tag{61}$$

In order to minimise the absolute value of (61), the inside signals should have no overlap with each other, hence

$$\begin{cases} n \neq n-\Delta-i+1 \\ n-2 \neq n-\Delta-i+1 \end{cases}, \quad \forall i = 1,2,\dots,M \tag{62}$$

$$\begin{cases} \Delta \neq 1-i \\ \Delta \neq 3-i \end{cases}, \quad \forall i = 1,2,\dots,M \tag{63}$$

Therefore, $\Delta \geq 3$.

In conclusion, the minimum value of the delay Δ that may be used in ALE is 3. The verification results using Matlab are shown in Fig. 22 and Fig. 23.

The filter order $M = 5$ is used in the simulation. As shown in Fig. 22, the mean square prediction error (MSPE) is decreased by 29.28% when the delay increases from 2 to 3. This change is also visible in Fig. 23. The spread out of the red estimated de-noised signal is less when the delay increases from 2 to 3.

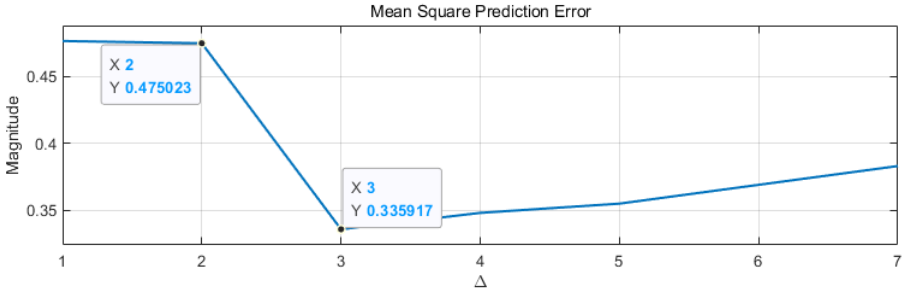

Fig. 22. MSPE between the estimated de-noised signal and the clean signal.

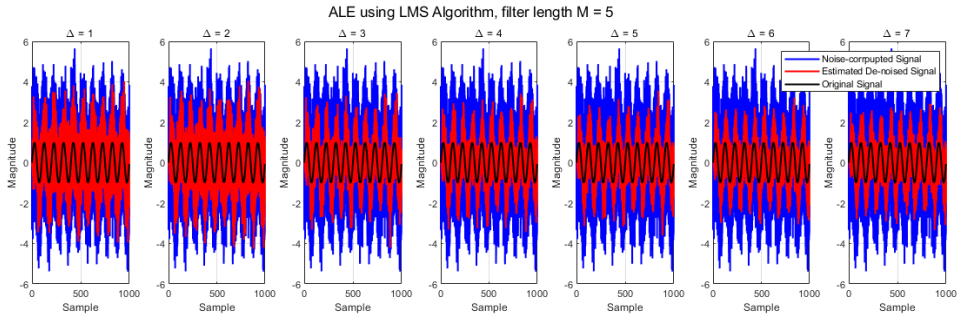

Fig. 23. ALE using LMS algorithm. The filter length of the model is 5.

b) The results are shown in Fig. 24.

From the left figure, the MSPE increases with the increase of delay. This is because the increase in delay reduces the level of correlation between the noise corrupted signal and its delayed version. Therefore, the MSPE is increased. However, for the large delays, the MSPE starts to decrease. This is caused by the periodicity in the signal. The period of the signal is 0.01π, and the signal contains 1000 samples, hence a period consists of 50 samples. Therefore, the 50-sample delayed version of the signal does not change the level of correlation. It can be considered that the result of MSPE versus delay has the same period as the signal.

From the right figure, the MSPE is also affected by the filter length. With a fixed delay $\Delta = 3$, the lowest MSPE is associated with the filter lengths 2 and 3. Despite the first MSPE at filter length 1, the MSPE increases as the increase of filter length. Therefore, the increase in filter length does not increase the prediction performance but increases the computational complexity. The best choice of filter length shown is the figure is 2.

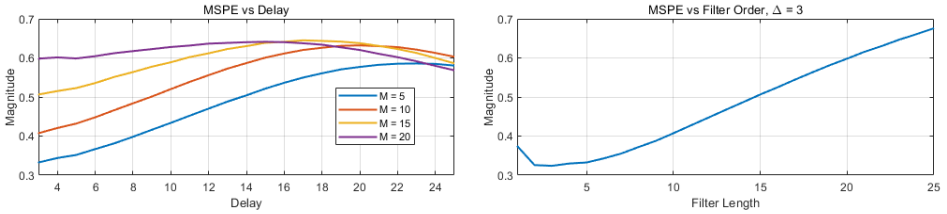

Fig. 24. (Left) MSPE against delay Δ. (Right) MSPE against filter length with delay $\Delta = 3$.

c) The result is shown in Fig. 25. Compared with ALE, adaptative noise cancellation (ANC) performs much better in noise cancellation. The MSPE of ALE is almost three times higher than that of ANC. In addition, the performance of ALE is relatively constant throughout the 1000 samples, while after the large fluctuation at the beginning, ANC provides a result with very low fluctuation in the steady state. This results in a much lower MSPE of 0.00645 in the last 500 samples.

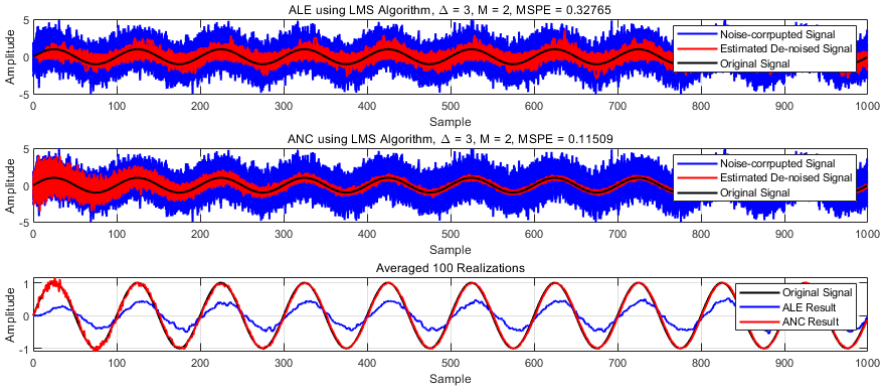

Fig. 25. (Top) ALE in de-noising the noisy sinusoid. (Middle) ANC in de-noising the noisy sinusoid, with 0.998 as the correlation coefficient between the noise and secondary noise. (Bottom) Averaged 100 realizations of the two methods comparing with the original sinusoid.

In Fig.25, the secondary noise is expressed as

$$\epsilon(n) = 0.7\eta(n) + 0.05n(n) \tag{64}$$

where $\eta(n)$ is the noise corrupts the original signal, and $n(n)$ is AWGN with unit power.

The correlation coefficient between $\epsilon(n)$ and $\eta(n)$ is $\rho = 0.998$. Due to this high-level correlation, the performance of ANC is good. However, if the correlation coefficient between $\epsilon(n)$ and $\eta(n)$ is decreased, for example,

$$\epsilon(n) = 0.7\eta(n) + 0.5n(n) \tag{65}$$

The correlation coefficient is 0.842 if $\epsilon(n)$ is derived using (65). In this case, the performance of ANC is shown in Fig. 26, where the MSPE is 0.4258, higher than that of ALE.

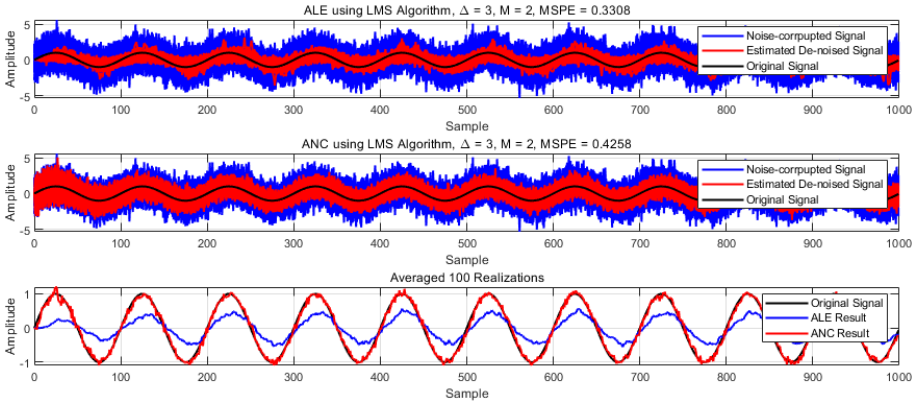

Fig. 26. (Top) ALE in de-noising the noisy sinusoid. (Middle) ANC in de-noising the noisy sinusoid, with 0.842 as the correlation coefficient between the noise and secondary noise. (Bottom) Averaged 100 realizations of the two methods comparing with the original sinusoid.

d) The spectrogram of the original POz data is shown in Fig. 27. The yellow horizontal line at 50 Hz is the component required to be removed without influencing other useful frequency components. These useful components generally have low frequency, at around 0 to 30 Hz.

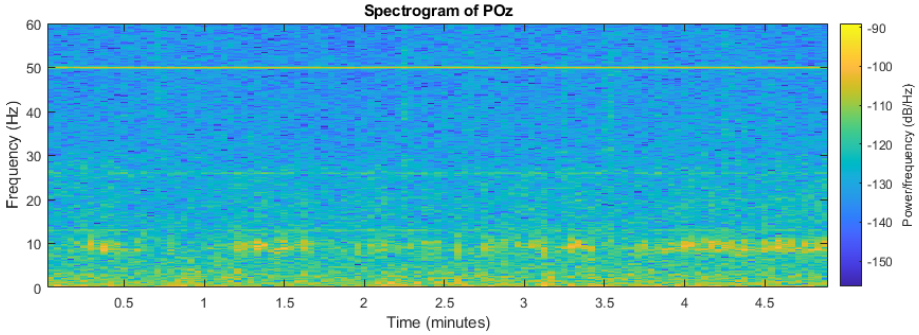

Fig. 27. Spectrogram of the POz data.

Theoretically, the choice of filter length M and step size μ should be under the consideration of the MSPE. However, simulation shows that although lower filter

length and lower step size results in lower MSPE, the performance of removing this 50 Hz component may not be good. In addition, although as shown in Fig. 28 that higher step sizes perform well in removing the 50 Hz component, they have significant influences on the other frequency components. It is also found that a higher filter length does not perform better compared with a medium one. Some simulation results are shown in Fig. 29, filter length $M = 10$ and step size $\mu = 0.01$ are regarded as the best choices. The periodogram of the ANC result derived using these two parameters is compared with the periodogram of POz data in Fig. 30. The 50 Hz peak in the periodogram of POz data is not visible in that of the ANC result. In the bottom figure in Fig. 30, except the initial fluctuation exists in ANC and the peak at 50 Hz, the squared error between the POz data and ANC result is at a low level.

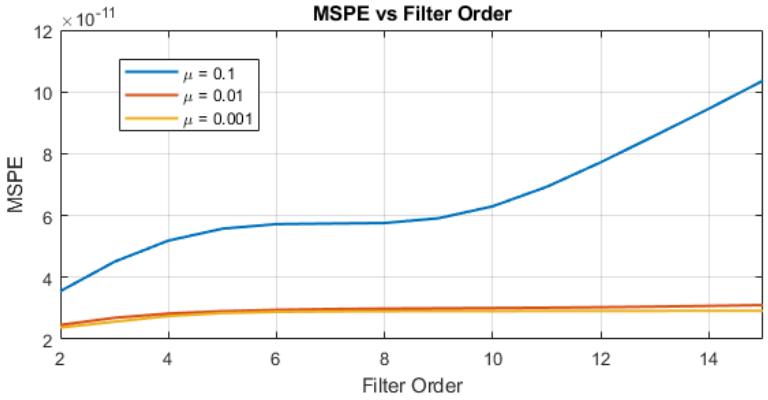

Fig. 28. MSPE versus the filter length.

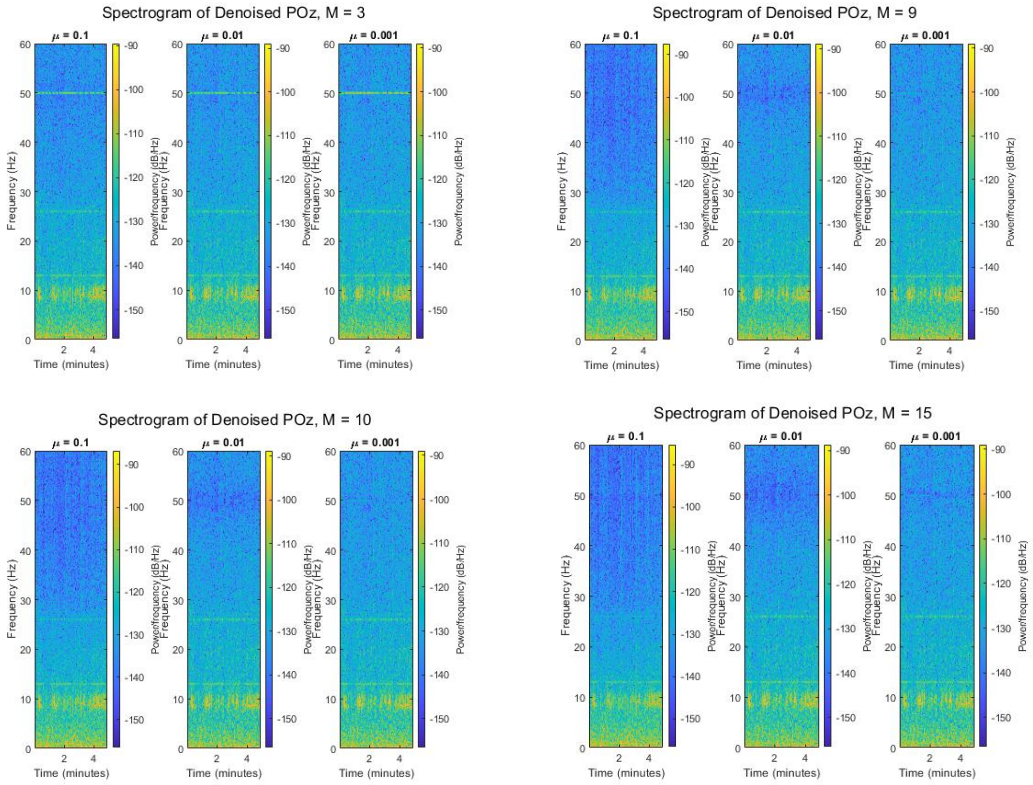

Fig. 29. Spectrogram of the de-noised POz data using ANC.

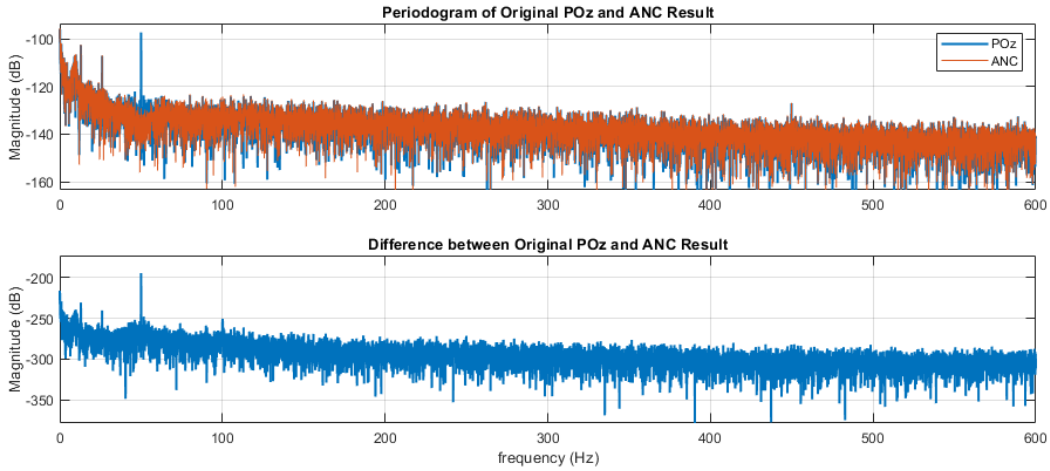

Fig. 30. (Top) Periodograms of the POz data and the de-noised POz data using ANC. (Bottom) Squared error between the POz data and ANC result.

3. Widely Linear Filtering and Adaptive Spectrum Estimation

3.1 Complex LMS (CLMS) and Widely Linear Modelling

a) The result is shown in Fig. 31.

In the left figure, the model of the first-order widely-linear-moving-average (WLMA) process is shown as a reference to be compared with the identification result using CLMS and Augmented CLMS (ACLMS). The result using ACLMS highly matches the WLMA(1) model, while the result using CLMS does not. This is also reflected in the learning curve of these two methods in the right figure. The learning curve for the ACLMS converges after about 430 samples reaching a low level of less than -300 dB. However, the learning curve for the CLMS does not change throughout the region with a high error.

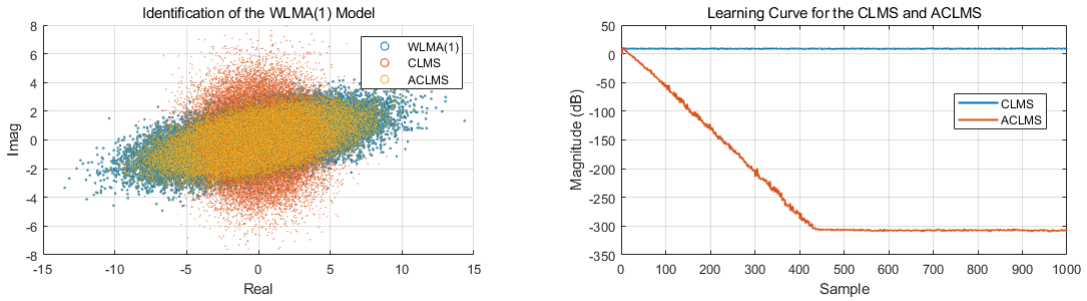

Fig. 31. (Left) WLMA(1) model and the identified result using CLMS and ACLMS with a learning rate $\mu = 0.1$. (Right) Squared error between the POz data and ANC result.

This phenomenon implies that CLMS is not suitable for estimating this WLMA(1) model. By analysing the circularity of the WLMA(1) model, the circularity coefficient $|\rho|$ is 0.8591. As suggested in the handout, CLMS is not valid for noncircular random variables. In this case, ACLMS should be used instead.

b) As shown in Fig. 32, low-wind data, medium-wind data, and high-wind data are associated with circularity coefficients $|\rho|$ of 0.159, 0.454, and 0.624, respectively. It can be concluded that the data with lower wind speed has smaller circularity coefficients, hence has a higher degree of circularity. This is also visible in the figure

at the bottom right corner in Fig. 32. The centre of the low-wind data is closer to the origin, and the data itself is more concentrated and circular than the other sets of data.

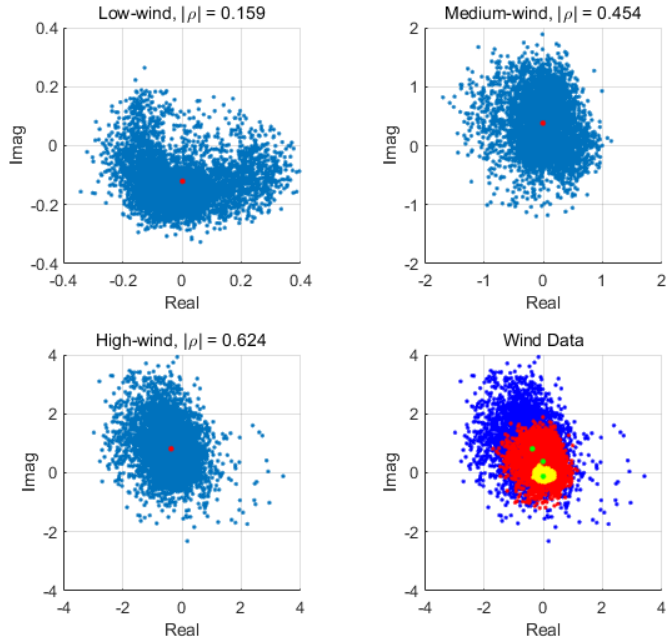

Fig. 32. The scatter diagram for the three wind regimes with their centre, and the comparison among them.

As shown in Fig. 33, CLMS and ACLMS are used to predict the three sets of complex wind data. For all sets of wind regimes, ACLMS performs better for lower filter lengths, and CLMS performs better for higher filter lengths. The particular filter length where the performance of CLMS exceeds ACLMS is different for different wind regimes. They are 9, 4, and 23 for low-wind, medium-wind, and high-wind data, respectively. In addition, the filter length where the ACLMS returns the lowest MSPE is generally lower than the filter length where the performance of CLMS exceeds ACLMS. Therefore, it can be concluded that ACLMS performs better than CLMS for all wind regimes, with lower filter length and MSPE.

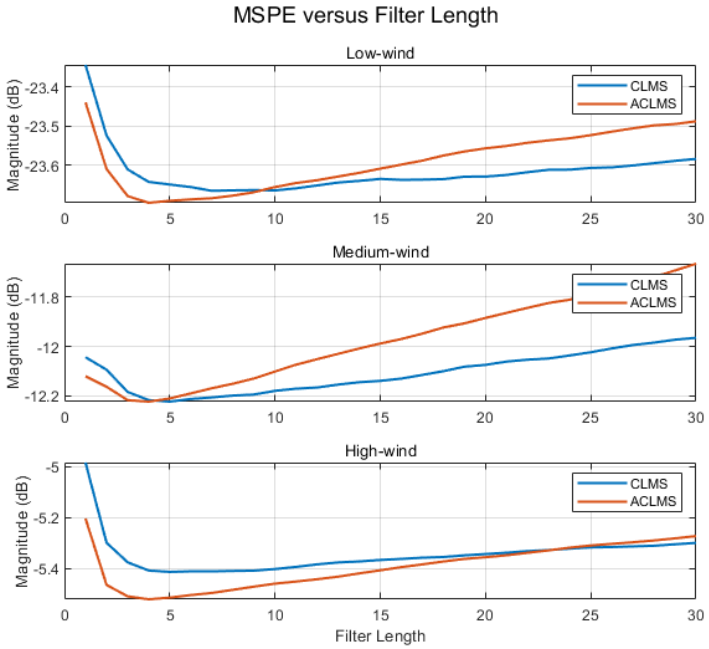

Fig. 33. The effect of the filter length on the MSPE of the CLMS and ACLMS.

c) The effect of changing the magnitude of voltages in the three-phase power system is shown in the left figure of Fig. 34. The balanced system is the blue circle with voltages $V_a = V_b = V_c = 1$ and circularity coefficient $|\rho| = 0$. By changing the voltages V_a, V_b, V_c to 0.5, 1, and 1.5, an ellipse is shown in the figure. In this case, the system is unbalanced with a circularity coefficient $|\rho| = 0.533$. Changing the order of 0.5, 1, and 1.5 does not change $|\rho|$ but the direction of the axis of the ellipse. If the gaps between the voltages are increased, the circularity coefficient $|\rho|$ increase simultaneously. This is shown in the figure as the green ellipse that is associated with V_a, V_b, V_c equal to 0.2, 1, and 1.8 and $|\rho| = 0.761$.

The effect of introducing the phase distortions to the three-phase system is shown in the right figure of Fig. 34. Similar to changing the magnitude, introducing nonzero phase distortions also leads to an unbalanced system. As shown in the figure, the original blue circle becomes ellipses with different nonzero circularity coefficients.

In order to identify a fault in the system, one efficient way is to check whether its circularity diagram is a circle. If it is a circle ($|\rho| = 0$), then the system is balanced, otherwise ($|\rho| \neq 0$) there exists a fault in the system.

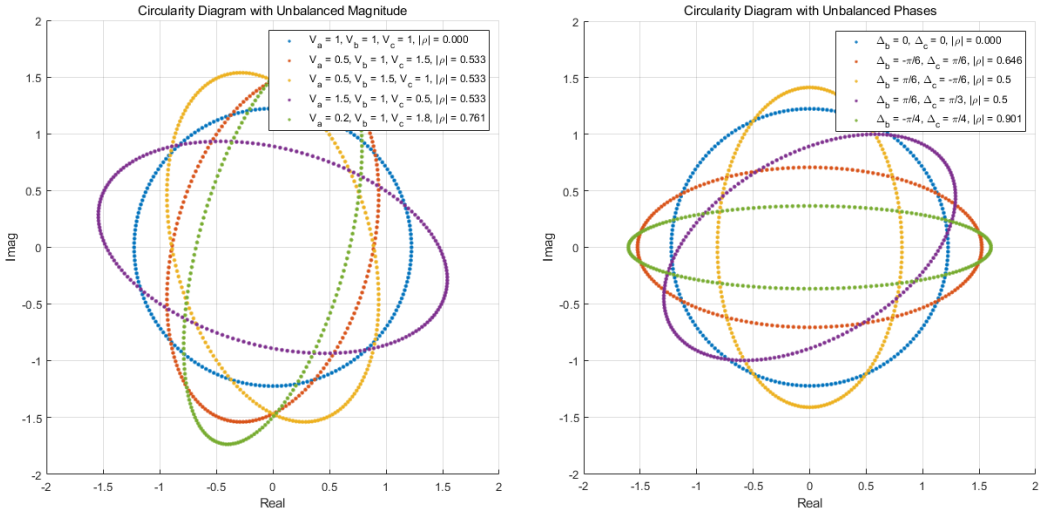

Fig. 34. (Left) Circularity diagram of the three-phase power system with unbalanced magnitude. (Right) Circularity diagram of the three-phase power system with unbalanced phase.

d) Firstly, consider the balanced system with the strictly linear autoregressive model of order 1,

$$v(n) = \sqrt{\frac{3}{2}} V e^{j(2\pi \frac{f_0}{f_s} n + \phi)} \tag{66}$$

$$v(n+1) = h^*(n) v(n) \tag{67}$$

Substitute (66) into (67),

$$\sqrt{\frac{3}{2}} V e^{j(2\pi \frac{f_0}{f_s}(n+1) + \phi)} = h^*(n) \sqrt{\frac{3}{2}} V e^{j(2\pi \frac{f_0}{f_s} n + \phi)} \tag{68}$$

$$h^*(n) = \frac{e^{j(2\pi \frac{f_0}{f_s}(n+1) + \phi)}}{e^{j(2\pi \frac{f_0}{f_s} n + \phi)}} \tag{69}$$

$$h^*(n) = e^{j2\pi\frac{f_0}{f_s}} \tag{70}$$

$$h(n) = e^{-j2\pi\frac{f_0}{f_s}} \tag{71}$$

Take the angle of $h(n)$,

$$angle\big(h(n)\big) = -2\pi\frac{f_0}{f_s} \tag{72}$$

$$\arctan\left\{\frac{\Im\{h(n)\}}{\Re\{h(n)\}}\right\} = -2\pi\frac{f_0}{f_s} \tag{73}$$

$$f_0 = -\frac{f_s}{2\pi}\arctan\left\{\frac{\Im\{h(n)\}}{\Re\{h(n)\}}\right\} \tag{74}$$

This expression is the same as the expression given in the handout. In this case, the $\arctan\left\{\frac{\Im\{h(n)\}}{\Re\{h(n)\}}\right\}$ should be negative to guarantee the frequency is positive. It also can be written as a positive definition as

$$f_0(n) = \frac{f_s}{2\pi}\arctan\left\{\frac{\Im\{h(n)\}}{\Re\{h(n)\}}\right\} \tag{75}$$

Then, consider the unbalanced system with the widely linear autoregressive model of order 1,

$$v(n) = A(n)e^{j\left(2\pi\frac{f_0}{f_s}n+\phi\right)} + B(n)e^{-j(2\pi\frac{f_0}{f_s}n+\phi)} \tag{76}$$

$$v(n+1) = h^*(n)v(n) + g^*(n)v^*(n) \tag{77}$$

where

$$A(n) = \frac{\sqrt{6}}{6}(V_a(n) + V_b(n)e^{j\Delta_b} + V_c(n)e^{j\Delta_c}) \tag{78}$$

$$B(n) = \frac{\sqrt{6}}{6}(V_a(n) + V_b(n)e^{-j(\Delta_b+2\pi/3)} + V_c(n)e^{-j(\Delta_c-2\pi/3)}) \tag{79}$$

Substitute (76) into (77),

$$\begin{aligned} &A(n+1)e^{j\left(2\pi\frac{f_0}{f_s}(n+1)+\phi\right)} + B(n+1)e^{-j\left(2\pi\frac{f_0}{f_s}(n+1)+\phi\right)} \\ &= h^*(n)\left(A(n)e^{j\left(2\pi\frac{f_0}{f_s}n+\phi\right)} + B(n)e^{-j\left(2\pi\frac{f_0}{f_s}n+\phi\right)}\right) \\ &+g^*(n)\left(A^*(n)e^{-j\left(2\pi\frac{f_0}{f_s}n+\phi\right)} + B^*(n)e^{j(2\pi\frac{f_0}{f_s}n+\phi)}\right) \end{aligned} \tag{80}$$

$$\begin{aligned} &A(n+1)e^{j2\pi\frac{f_0}{f_s}}e^{j\left(2\pi\frac{f_0}{f_s}n+\phi\right)} + B(n+1)e^{-j2\pi\frac{f_0}{f_s}}e^{-j\left(2\pi\frac{f_0}{f_s}n+\phi\right)} \\ &= \big(h^*(n)A(n) + g^*(n)B^*(n)\big)e^{j\left(2\pi\frac{f_0}{f_s}n+\phi\right)} \\ &+(g^*(n)A^*(n) + h^*(n)B(n))e^{-j\left(2\pi\frac{f_0}{f_s}n+\phi\right)} \end{aligned} \tag{81}$$

Since the equivalent is satisfied, the coefficients of $e^{j\left(2\pi\frac{f_0}{f_s}n+\phi\right)}$ and $e^{-j\left(2\pi\frac{f_0}{f_s}n+\phi\right)}$ are equal.

$$\begin{cases} A(n+1)e^{j2\pi\frac{f_0}{f_s}} = h^*(n)A(n) + g^*(n)B^*(n) \\ B(n+1)e^{-j2\pi\frac{f_0}{f_s}} = g^*(n)A^*(n) + h^*(n)B(n) \end{cases} \tag{82}$$

Assume that the coefficients $A(n)$ and $B(n)$ change slowly, that is, $A(n+1) \approx A(n)$ and $B(n+1) \approx B(n)$. Then, (82) is written as

$$\begin{cases} A(n)e^{j2\pi\frac{f_0}{f_s}} = h^*(n)A(n) + g^*(n)B^*(n) \\ B(n)e^{-j2\pi\frac{f_0}{f_s}} = g^*(n)A^*(n) + h^*(n)B(n) \end{cases} \tag{83}$$

$$\begin{cases} e^{j2\pi\frac{f_0}{f_s}} = h^*(n) + g^*(n)\dfrac{B^*(n)}{A(n)} \\ e^{-j2\pi\frac{f_0}{f_s}} = g^*(n)\dfrac{A^*(n)}{B(n)} + h^*(n) \end{cases} \tag{84}$$

$$\begin{cases} e^{j2\pi\frac{f_0}{f_s}} = h^*(n) + g^*(n)\dfrac{B^*(n)}{A(n)} \\ e^{j2\pi\frac{f_0}{f_s}} = g(n)\dfrac{A(n)}{B^*(n)} + h(n) \end{cases} \tag{85}$$

Therefore,

$$g(n)\frac{A(n)}{B^*(n)} + h(n) = h^*(n) + g^*(n)\frac{B^*(n)}{A(n)} \tag{86}$$

$$g^*(n)\frac{B^*(n)}{A(n)} - g(n)\frac{A(n)}{B^*(n)} + h^*(n) - h(n) = 0 \tag{87}$$

$$g^*(n)\left(\frac{B^*(n)}{A(n)}\right)^2 - j2\Im\{h(n)\}\frac{B^*(n)}{A(n)} - g(n) = 0 \tag{88}$$

Considering $\frac{B^*(n)}{A(n)}$ as the unknown,

$$\frac{B^*(n)}{A(n)} = \frac{j2\Im\{h(n)\} \pm \sqrt{(-j2\Im\{h(n)\})^2 + 4g^*(n)g(n)}}{2g^*(n)} \tag{89}$$

$$= j\frac{\Im\{h(n)\} \pm \sqrt{(\Im\{h(n)\})^2 - |g(n)|^2}}{g^*(n)} \tag{90}$$

Substitute (90) into the first equation in (84),

$$e^{j2\pi\frac{f_0}{f_s}} = h^*(n) + g^*(n)\left(j\frac{\Im\{h(n)\} \pm \sqrt{(\Im\{h(n)\})^2 - |g(n)|^2}}{g^*(n)}\right) \tag{91}$$

$$= h^*(n) + j\Im\{h(n)\} \pm j\sqrt{(\Im\{h(n)\})^2 - |g(n)|^2} \tag{92}$$

$$= \Re\{h(n)\} \pm j\sqrt{(\Im\{h(n)\})^2 - |g(n)|^2} \tag{93}$$

Take the angle of $e^{j2\pi\frac{f_0}{f_s}}$,

$$angle\left(e^{j2\pi\frac{f_0}{f_s}}\right) = \arctan\left\{\frac{\pm\sqrt{(\Im\{h(n)\})^2 - |g(n)|^2}}{\Re\{h(n)\}}\right\} \tag{94}$$

$$2\pi\frac{f_0}{f_s} = \arctan\left\{\frac{\pm\sqrt{(\Im\{h(n)\})^2 - |g(n)|^2}}{\Re\{h(n)\}}\right\} \tag{95}$$

$$f_0 = \pm\frac{f_s}{2\pi}\arctan\left\{\frac{\sqrt{(\Im\{h(n)\})^2 - |g(n)|^2}}{\Re\{h(n)\}}\right\} \tag{96}$$

This expression is the same as the expression given in the handout. It can also be written as a positive definition as

$$f_0(n) = \frac{f_s}{2\pi}\arctan\left\{\frac{\sqrt{(\Im\{h(n)\})^2 - |g(n)|^2}}{\Re\{h(n)\}}\right\} \tag{97}$$

e) From the top figure in Fig. 35, both CLMS and ACLMS successfully estimate the frequency of the balanced system with similar speeds of convergence. As for the frequency estimation for the unbalanced system shown in the remaining two figures in Fig. 35, CLMS fails. At the beginning, the curve for CLMS increases with that of ACLMS, while starts to oscillate after a few samples. It then fluctuates around a fixed value, lower than the frequency estimated by ACLMS which converges to the true value.

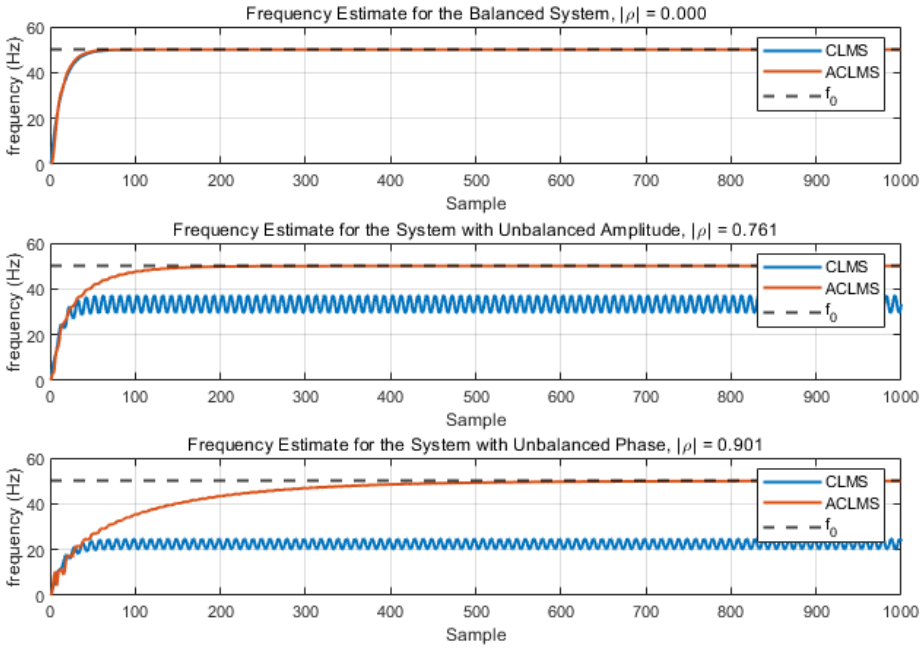

Fig. 35. Frequency estimation of the $\alpha - \beta$ voltages using CLMS and ACLMS for balanced and unbalanced systems.

The reason is that the strictly linear model yields biased estimation for unbalanced system. For balanced systems,

$$v(n) = A(n)e^{j\left(2\pi\frac{f_0}{f_s}n+\phi\right)} \tag{98}$$

(98) shows the circular trajectory. However, for unbalanced systems,

$$v(n) = A(n)e^{j\left(2\pi\frac{f_0}{f_s}n+\phi\right)} + B(n)e^{-j(2\pi\frac{f_0}{f_s}n+\phi)} \tag{99}$$

The existence of the conjugate component $e^{-j(2\pi\frac{f_0}{f_s}n+\phi)}$ does not satisfy the strictly linear model. Therefore, the CLMS does not give the correct frequency estimate. In other word, the degree of freedom of the CLMS is 1, which is less than the required degree of freedom of 2 for the ellipse.

The level of unbalance of the system also affects the frequency estimation. With a larger circularity coefficient $|\rho|$ which implies higher instability in the system, ACLMS converges slower, while CLMS fluctuates around a smaller fixed value.

3.2 Adaptive AR Model Based Time-Frequency Estimation

a) Fig. 36 shows the estimation result of the given frequency modulated (FM) signal using AR(1) model. In the top figure in Fig. 36, PSD peaks at 184 Hz. However, 184 Hz only occupies 3 samples in the 1000 samples of frequency $f(n)$.

If the FM signal is divided into three segments which are derived by the three stages of $f(n)$ respectively, the estimation result is plotted in the remaining figures in Fig. 36. It is shown that the frequency in the first segment is successfully captured,while is failed to be captured in the last two segments. This is because the frequency is fixed in the first segment while variate in the other two segments. Therefore, it can be concluded as the method used in this section cannot capture the changes in frequency.

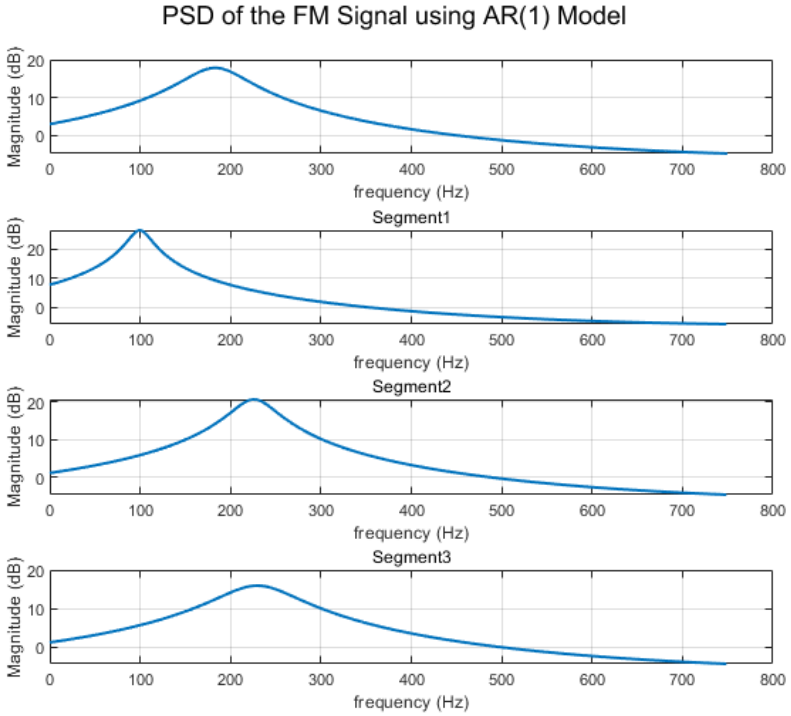

Fig. 36. (Top) PSD of the given FM signal estimated by AR(1) model. (Remaining figures) PSD of the 3 segments of the given FM signal estimated by AR(1) model.

b) The estimated time-frequency spectrum is shown in Fig. 37. The red line in the figure is the theoretical value as a reference. It is observed that the estimated frequency highly matches the theoretical value, except instant large decrease at the 1000th sample, and some losses at the beginning.

The FM signal can be processed by CLMS due to its low circularity coefficient $|\rho| = 0.0029$. The AR coefficients are the weights estimated by CLMS and are updated in every sample. The variate AR coefficients are able to capture the changes in frequency.

What have to be mentioned is that weights require time to converge for sudden changes. This is the reason for the inaccuracy at the beginning and around the 1000th sample.

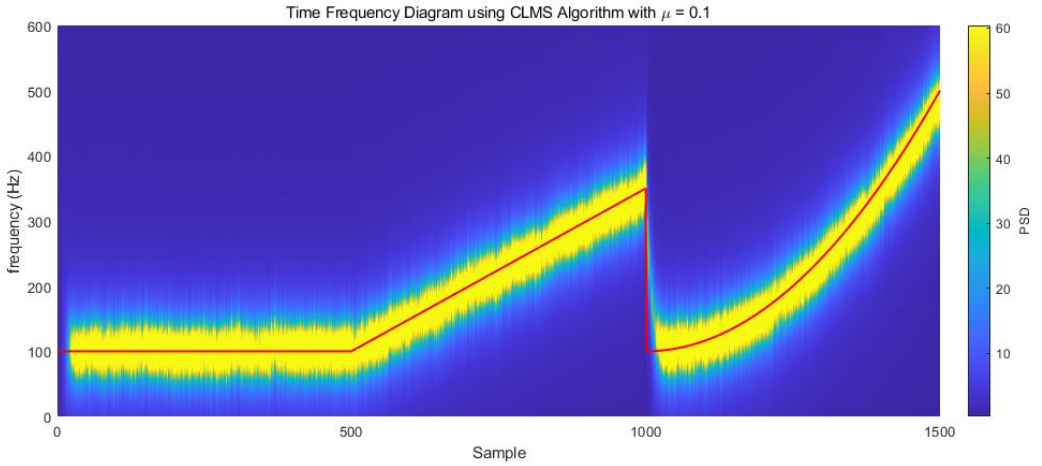

Fig. 37. Time-frequency spectrum of the FM signal based on CLMS.

3.3 A Real Time Spectrum Analyser Using Least Mean Square

a) The matrix expression of all N estimates of the signal $y(n)$ in the handout can be expressed as

$$\hat{\boldsymbol{y}} = \boldsymbol{F}\boldsymbol{w} \tag{100}$$

Given the cost function $J(\boldsymbol{w}) = \|\boldsymbol{y} - \hat{\boldsymbol{y}}\|^2$ and substitute (100) into it,

$$J(\boldsymbol{w}) = \|\boldsymbol{y} - \boldsymbol{F}\boldsymbol{w}\|^2 \tag{101}$$

$$= (\boldsymbol{y} - \boldsymbol{F}\boldsymbol{w})^H(\boldsymbol{y} - \boldsymbol{F}\boldsymbol{w}) \tag{102}$$

$$= (\boldsymbol{y}^H - \boldsymbol{w}^H\boldsymbol{F}^H)(\boldsymbol{y} - \boldsymbol{F}\boldsymbol{w}) \tag{103}$$

$$= \boldsymbol{y}^H\boldsymbol{y} - \boldsymbol{y}^H\boldsymbol{F}\boldsymbol{w} - \boldsymbol{w}^H\boldsymbol{F}^H\boldsymbol{y} + \boldsymbol{w}^H\boldsymbol{F}^H\boldsymbol{F}\boldsymbol{w} \tag{104}$$

In order to find the minimum value of $J(\boldsymbol{w})$, the first-order derivative is applied,

$$\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}} = \frac{\partial(\boldsymbol{y}^H\boldsymbol{y} - \boldsymbol{y}^H\boldsymbol{F}\boldsymbol{w} - \boldsymbol{w}^H\boldsymbol{F}^H\boldsymbol{y} + \boldsymbol{w}^H\boldsymbol{F}^H\boldsymbol{F}\boldsymbol{w})}{\partial \boldsymbol{w}} \tag{105}$$

$$= 2\boldsymbol{F}^H\boldsymbol{F}\boldsymbol{w} - 2\boldsymbol{F}^H\boldsymbol{y} \tag{106}$$

Let the first-order derivative to be 0,

$$\left.\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}}\right|_{\boldsymbol{w_{opt}}} = 0 \tag{107}$$

$$2\boldsymbol{F}^H\boldsymbol{F}\boldsymbol{w_{opt}} - 2\boldsymbol{F}^H\boldsymbol{y} = 0 \tag{108}$$

$$\boldsymbol{w_{opt}} = (\boldsymbol{F}^H\boldsymbol{F})^{-1}\boldsymbol{F}^H\boldsymbol{y} \tag{109}$$

(109) shows the least-squares (LS) solution for the minimisation of the cost function.

Consider the inverse discrete Fourier transform (IDFT),

$$y(n) = \frac{1}{N}\sum_{k=0}^{N} Y(k)e^{j\frac{2\pi kn}{N}} \tag{110}$$

(110) can be expressed as (100) using the same idea in the handout,

$$\hat{\boldsymbol{y}} = \frac{1}{N}\boldsymbol{F}\boldsymbol{Y} \tag{111}$$

Let $\boldsymbol{F}_{IDFT} = \frac{1}{N}\boldsymbol{F}$,

$$\hat{\boldsymbol{y}} = \boldsymbol{F}_{IDFT}\boldsymbol{Y} \tag{112}$$

Compare (112) with (100), they have the same meaning but belongs to different DFT and IDFT expression.

(112) satisfies the following DFT and IDFT expression,

$$Y(k) = \sum_{k=0}^{N} y(n)e^{-j\frac{2\pi kn}{N}} \quad y(n) = \frac{1}{N}\sum_{k=0}^{N} Y(k)e^{j\frac{2\pi kn}{N}} \tag{113}$$

(100) satisfies the following DFT and IDFT expression,

$$Y(k) = \frac{1}{N}\sum_{k=0}^{N} y(n)e^{-j\frac{2\pi kn}{N}} \quad y(n) = \sum_{k=0}^{N} Y(k)e^{j\frac{2\pi kn}{N}} \tag{114}$$

The difference is where to put the coefficient $\frac{1}{N}$.

b) Considering the IDFT matrix $\boldsymbol{F}$ given in the handout, $\boldsymbol{F}^H\boldsymbol{F} = N\boldsymbol{I}$. Therefore, the LS solution is expressed as

$$\boldsymbol{w} = \frac{1}{N}\boldsymbol{F}^H\boldsymbol{y} \tag{115}$$

(115) indicates the DFT of $y(n)$, and $\frac{1}{N}\boldsymbol{F}^H$ is the orthonormal DFT matrix. The DFT matrix projects N samples of the estimating signal $\boldsymbol{y}$ to the Fourier domain formed by the column vectors in the DFT matrix. The basis of $\boldsymbol{y}$ is changed from the time domain to the frequency domain spanned by N column vectors of the DFT matrix, each of them is a combination of N harmonically related sinusoids.

c) As shown in Fig. 38, the spectrum obtained from the weights of the DFT-CLMS does not converge to the true frequency shown as the red line in the domain where the frequency is not constant. The reason is that the weight is an accumulation starting from 0 when DFT is introduced to the CLMS algorithm. Therefore, for each sample of the FM signal, the weight is a vector consisting of 0 to its optimum value. In this case, all the weights starting from 0 to the optimum value are plotted in the figure.

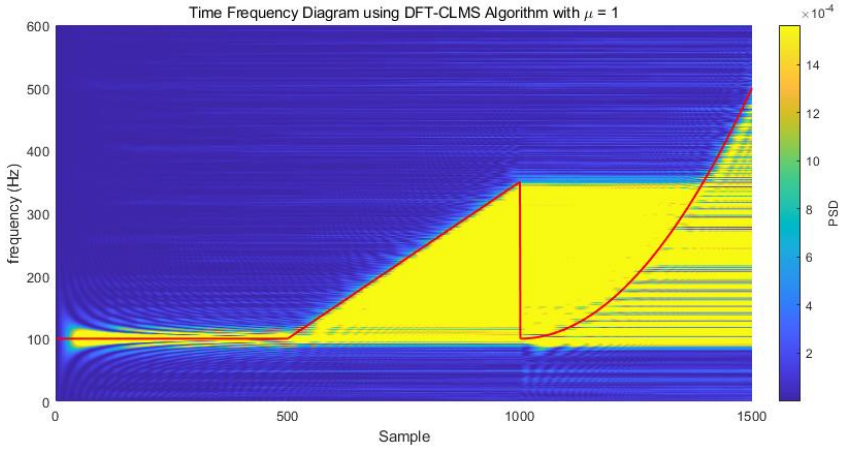

Fig. 38. Time-frequency spectrum of the FM signal based on DFT-CLMS.

This problem can be solved by using the leaky LMS in section 2.1 f). The result of using leaky DFT-CLMS is shown in Fig. 39. The leak decreases the dependence of the weight update to the previous weight, hence the accumulation of the weight from 0 becomes not significant. Although there still exists some vertical lines that indicate the accumulation of the weights, the performance of this algorithm is acceptable. Therefore, leaky LMS is a good method for the estimation of variant frequency when DFT-CLMS is used.

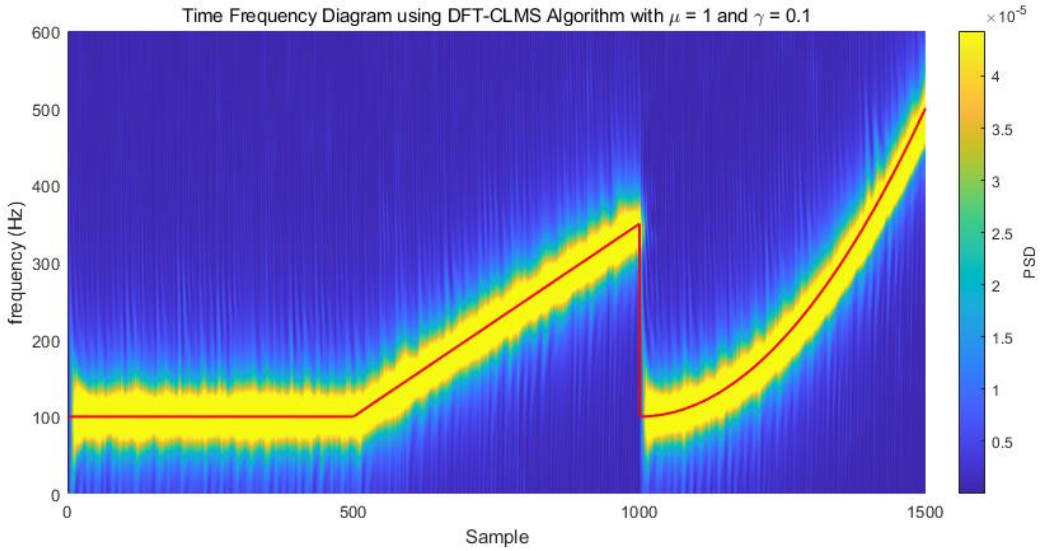

Fig. 39. Time-frequency spectrum of the FM signal based on leaky DFT-CLMS.

d) The result is shown in Fig. 40, where 1200 samples of POz data are taken, which starts from the 1000th sample. The strong response within 8-10 Hz, the power-line interference at 50 Hz, the fundamental frequency response at 13 Hz, and the harmonic at 26 Hz are visible in the figure. In order to observe the important frequency components clearly, the threshold of removing outliers is increased, hence the small harmonic at 39 Hz is not visible. There also exists a strong response in the low-frequency band, which is mainly caused by the noise.

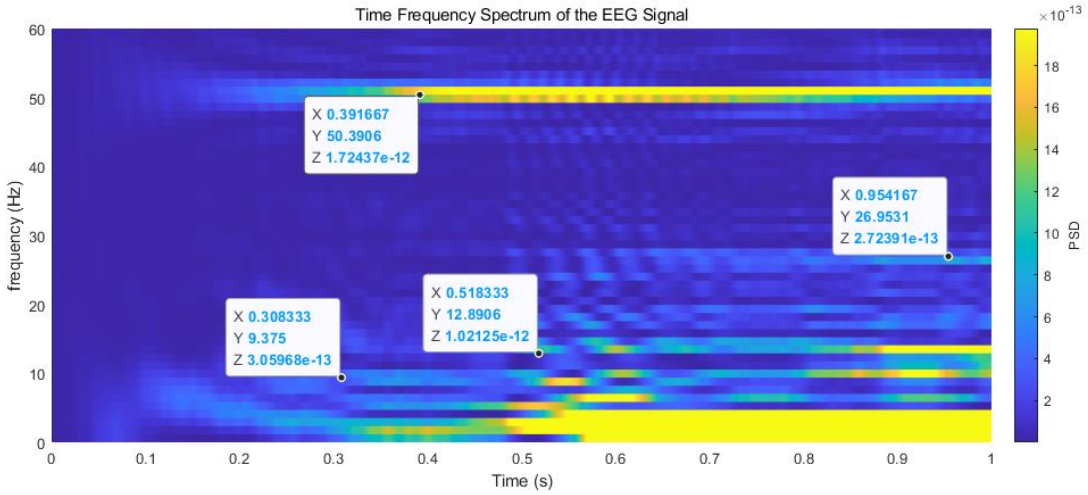

Fig. 40. Time-frequency spectrum of the EEG signal based on CLMS.

4. From LMS to Deep Learning

1)

As shown in Fig. 41, the prediction of the zero-mean version of y converges after about 200 samples. The performance of this prediction is quantified by computing the MSE and prediction gain R_P in the steady state (last 500 samples), which are 11.94 dB and 9.81 dB respectively. According to these two parameters, there still exist a large space for the improvement of the LMS algorithm.

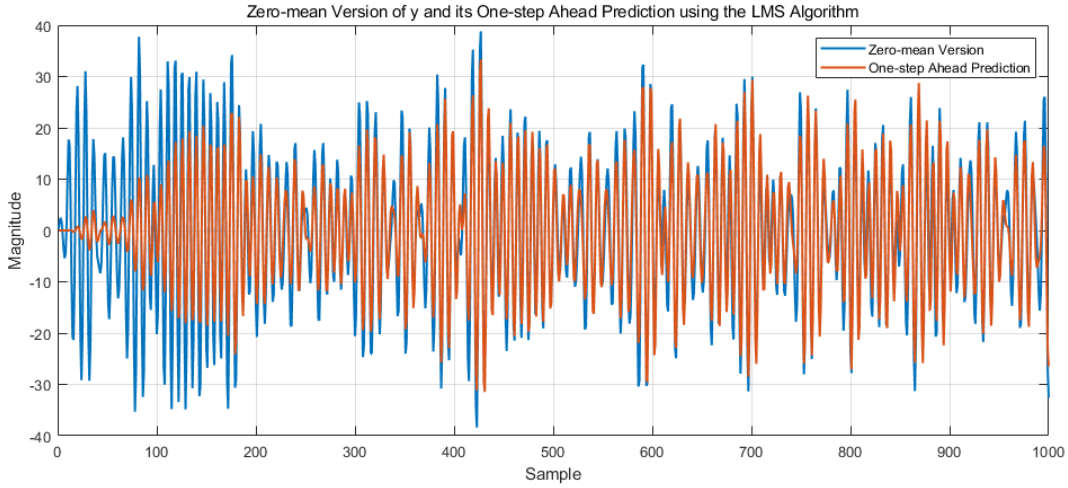

Fig. 41. Zero-mean version of the signal y and its one-step ahead prediction using LMS algorithm.

2)

To realise the dynamical perceptron, the expression of the signal estimate is changed.

$$\hat{y}(n) = \boldsymbol{w}^T \boldsymbol{x} \quad \Rightarrow \quad \hat{y}(n) = \tanh(\boldsymbol{w}^T \boldsymbol{x}) \tag{116}$$

As shown in Fig. 42, the prediction does not match the original signal. Although the frequency change of the signal is successfully captured, the amplitude is limited to a small range. The reason is that $\tanh()$ maps the region $(-\infty, \infty)$ to $(-1,1)$, hence the prediction $\hat{y}(n)$ is limited to the region $(-1,1)$.

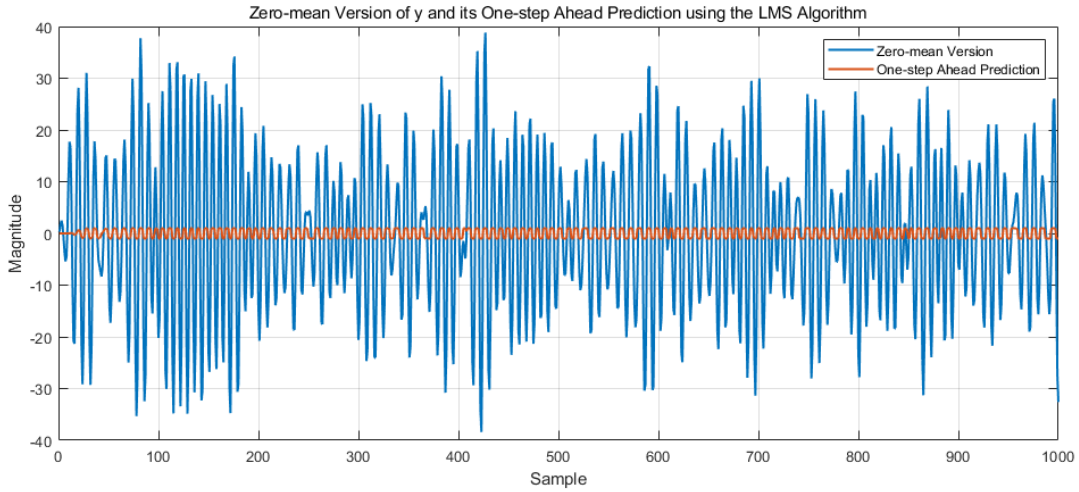

Fig. 42. Zero-mean version of the signal y and its one-step ahead prediction using LMS algorithm with activation function $\tanh()$.

3)

The problem in the last section can be solved by scaling the activation function by a. Then, the signal estimate is expressed as

$$\hat{y}(n) = a * \tanh(\boldsymbol{w}^T \boldsymbol{x}) \tag{117}$$

In order to find an appropriate scale a, MSPE and prediction gain R_P are computed against a increasing from 1 to 200, shown in Fig. 43.

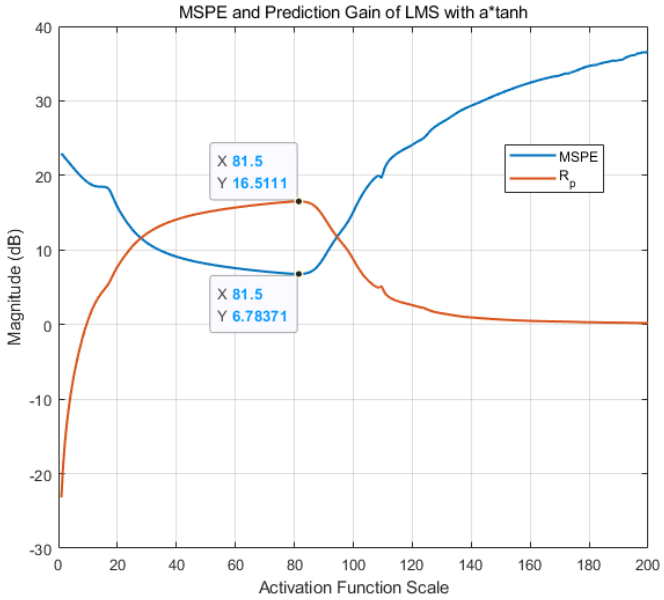

Fig. 43. MSPE and prediction gain versus activation function scale.

The minimum MSPE and maximum R_P exists at $a = 81.5$, which are 6.78 dB and 16.51 dB, respectively. Therefore, $a = 81.5$ is the best activation function scale. The prediction result using this scale is shown in Fig. 44, together with the results using $a = 20$ and $a = 100$ as references. It is observed that if a is too small, the problem in the last section appears again. If a is too large, the model becomes unstable, for example, the instant error at around the 430th sample in the bottom figure in Fig. 44.

Comparing with the standard LMS algorithm in section 4.1 (MSPE = 11.94 dB, R_P = 9.81 dB), MSPE is decreased by 5.16 dB and R_P is increased by 6.70 dB. Therefore, the performance of adding non-linearity to the output of LMS can significantly increase the performance.

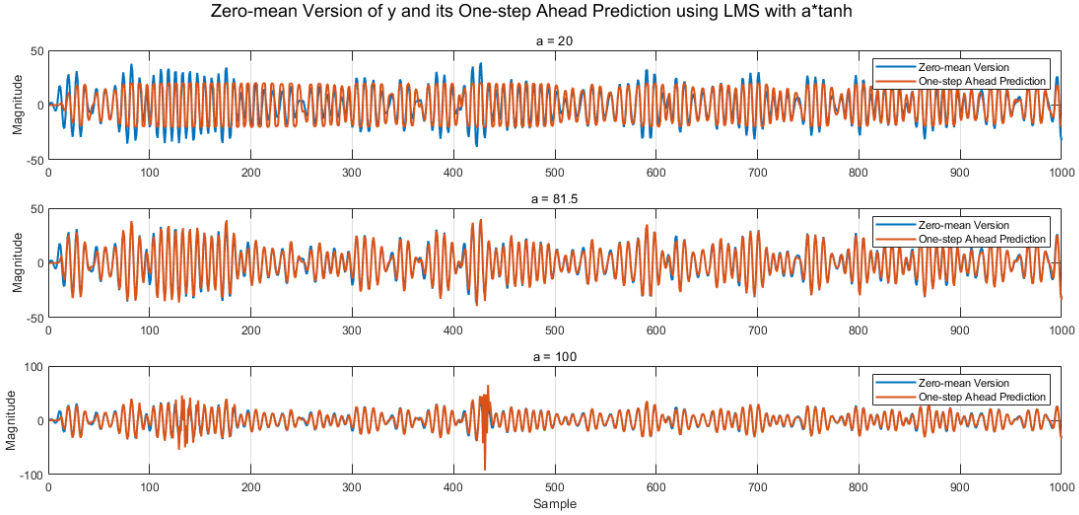

Fig. 44. Zero-mean version of the signal y and its one-step ahead prediction using LMS algorithm with activation function $a * \tanh()$.

4)

By introducing the bias, the signal estimate is expressed as

$$\hat{y}(n) = a * \tanh(\boldsymbol{w}^T \boldsymbol{x} + b) \tag{118}$$

This is realised in the code by using the "augmented" input mentioned in the handout.

Again, MSPE and prediction gain R_P are computed against a changing from 1 to 200. The optimum choice of a is 52.2 where MSPE and R_P are 11.27 dB and 12.34 dB respectively. The result is shown in Fig. 45, together with the results using $a = 20$ and $a = 100$ as references. Similar problems exist in the small and large choices of a. According to the values of MSPE and R_P, the performance of this algorithm is worse than that of the previous section, with an increase in MSPE by 4.49 dB and a decrease in R_P by 4.17 dB. However, this slight decline in performance is acceptable and can be improved by using a better choice of b.

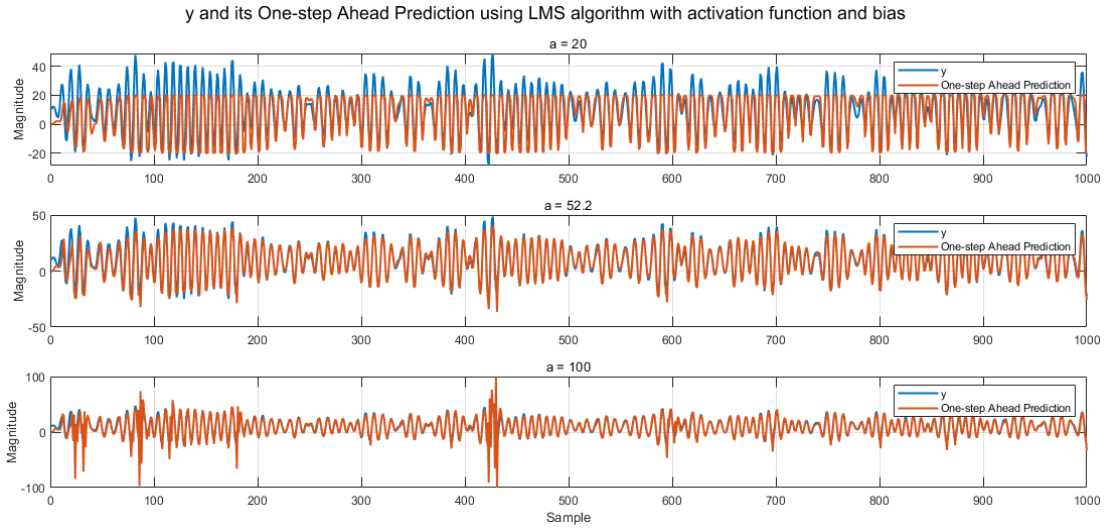

Fig. 45. y and its one-step ahead prediction using LMS algorithm with activation function and bias.

5)

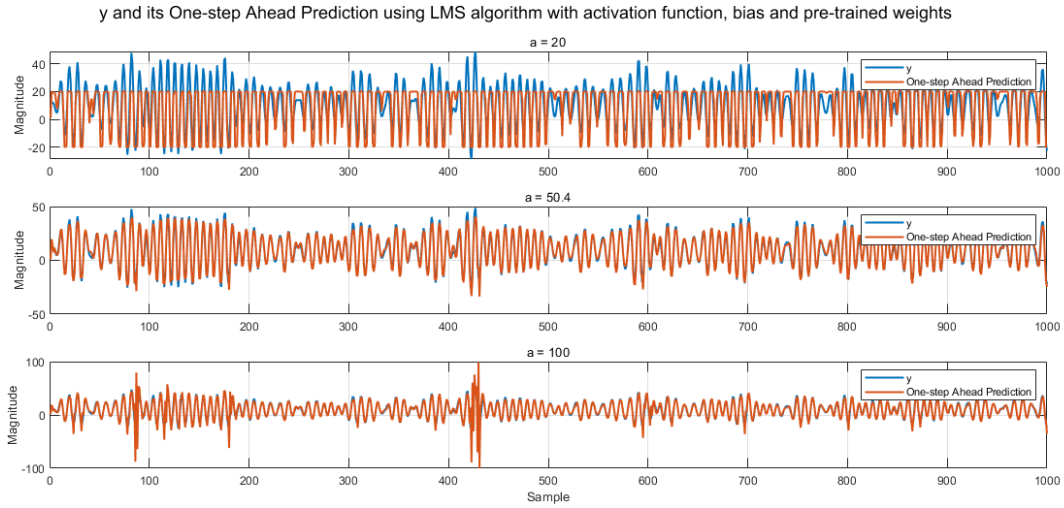

Fig. 46. Zero-mean version of the signal y and its one-step ahead prediction using LMS algorithm with activation function, bias, and pre-trained weights.

The non-linear one-step ahead prediction produced with activation function, bias, and pre-trained weights is shown in Fig. 46 with $a = 50.4$ and with $a = 20$ and $a = 100$ as references. The best choice of a is 50.4 where MSPE and R_P are 8.45 dB and 15.06 dB respectively. Comparing with MSPE = 11.27 dB, R_P = 12.34 dB in the previous section, pre-trained weights are able to improve the prediction performance.

The ability of increasing the convergence speed using pre-trained weights is quantified by measuring the MSPE and R_P in the first 200 samples of the prediction results in this and the previous section. As shown in Tab. 9, the differences between the MSPE and R_P measured by all the samples and by the first 200 samples becomes smaller if the pre-trained weights are used.

Tab. 9. MSPE and R_P of the prediction with and without pre-trained weights.

Measurement	MSPE		R_P	
	with $\boldsymbol{w}_{init}$	without $\boldsymbol{w}_{init}$	with $\boldsymbol{w}_{init}$	without $\boldsymbol{w}_{init}$
all the samples	8.45 dB	11.27 dB	15.06 dB	12.34 dB
first 200 samples	11.33 dB	16.37 dB	14.25 dB	9.77 dB
Difference	2.88 dB	5.1 dB	-0.81 dB	-2.57 dB

6)

The deep network in Figure 11 in the handout is taken as an example. At the beginning of the training of this feedforward deep network, the weights in each neuron of each layer are initialized, for example, starting with $w_0 = 0$. Then, the set of observations $x[n-1]$ to $x[n-4]$ flows into the network from left to right through the 3 hidden layers and generates a prediction of $\hat{x}[n]$ at the output layer. During this flow, the activation function in each neuron is computed based on those in the previous neurons. After that, in the output layer, the error is measured using the prediction $\hat{x}[n]$ and target x^*, and the weights are updated in the LMS manner. In order to update the weights belonging to each neuron in the hidden layers, as they do not have a target signal to evaluate their inaccuracy, the error information measured in the output layer is back-propagated to neurons in Layer 3. This error information is the local gradient of the cost function in the output layer. After receiving the error information, neurons in Layer 3 compute errors on their cost functions to compute errors on their weights. Then, the weights are updated, and the gradients of their cost functions are back-propagated to neurons in Layer 2. Then, the same procedure happens in Layer 2, and then in Layer 1.

This method is called backpropagation. Commonly used for training feedforward neural networks (e.g. the example given in the handout), backpropagation computes the gradient of the cost function with respect to weights in each neuron and sends it backwards for the computation of the gradient in upper neurons. This method is called gradient descent, which repeatedly takes steps in the opposite direction to the gradient. This is the direction of the steepest descent to the optimum. In addition, the backpropagation consisting of many simple "dynamical perceptron" is an example of dynamic programming, which computes gradients of cost functions in a layer, and iterates the gradients to the upper layers.

7)

Using the default settings, the result is shown as follows. The input signal is a combination of 10 different sinusoidal signals as shown in the top figure in Fig. 47. The desired signal $y[n]$ is the highly non-linear combination of the sinusoidal inputs and is corrupted by noise with a power of 0.05.

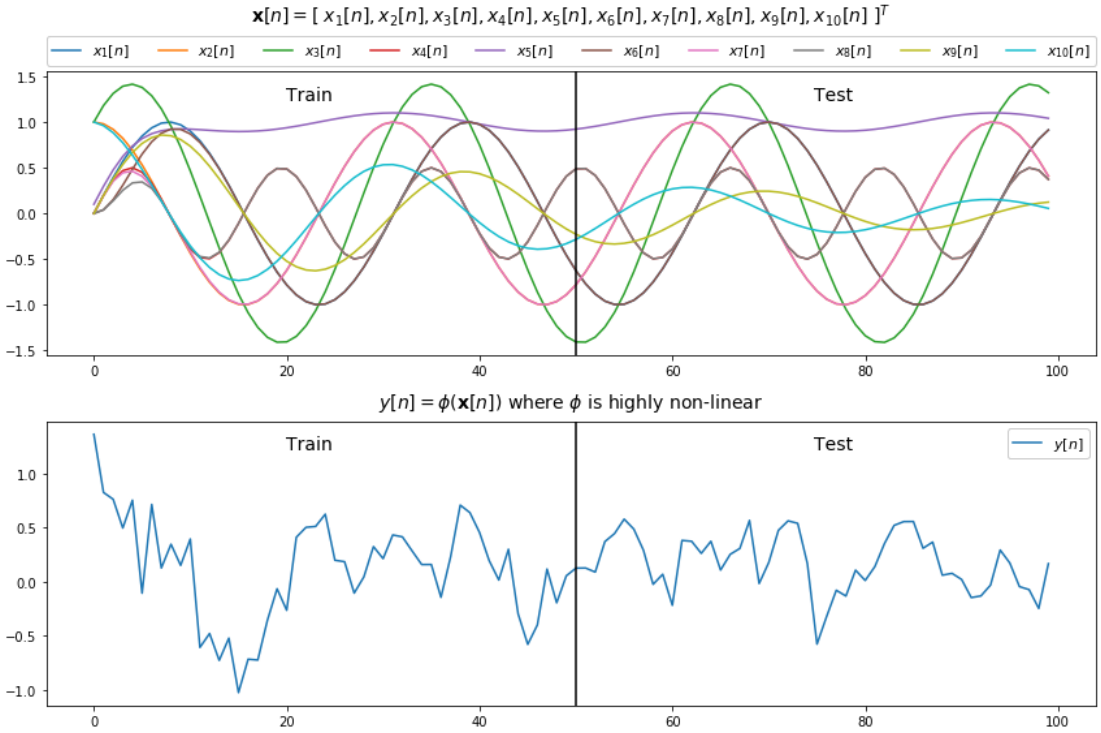

Fig. 47. (Top) Signals that form the desired signal. (Bottom) Desired signal, the non-linear combination of the input signals and corrupted by noise with a power of 0.05.

The prediction results are shown in Fig. 48.

The results of using single neuron with LMS and LMS using activation function tanh() have similar steady state errors, while the use of activation function increases the convergence speed slightly. The learning curve of the LMS using training data converges after about 500 epochs, while that of LMS using activation function converges after about 200 epochs. When the testing data is inputted into the trained model, there exists some difference in the learning curves of both methods, while the learning curves still converge to the same steady state error as the training data.

Different from the front two methods, the prediction using deep network converges much slower (after 10000 epochs). The reasons are the high order of the deep network and the influences of the error on the path of the gradient descent, hence it takes the network more epochs to reach the optimum result. However, learning curve shows that the deep network has the lowest steady state error among the three algorithms. When the input becomes the testing data, there exists a large fluctuation in the learning curve at the beginning, while the learning curve converges after about 10000 epochs with a slightly higher steady state error compared with the learning curve of the training data.

In conclusion, since the model is highly non-linear, the activation function does not improve the performance much. Both the LMS and LMS using tanh() are not able to predict the signal with a low error. In addition, although it seems inn the top figure in Fig. 48 that the performance of the prediction using deep network is even worse than

the other two methods, deep network results in lower prediction error. This is because the deep network has higher degrees of freedom than the other two methods hence processes this highly non-linear signal better.

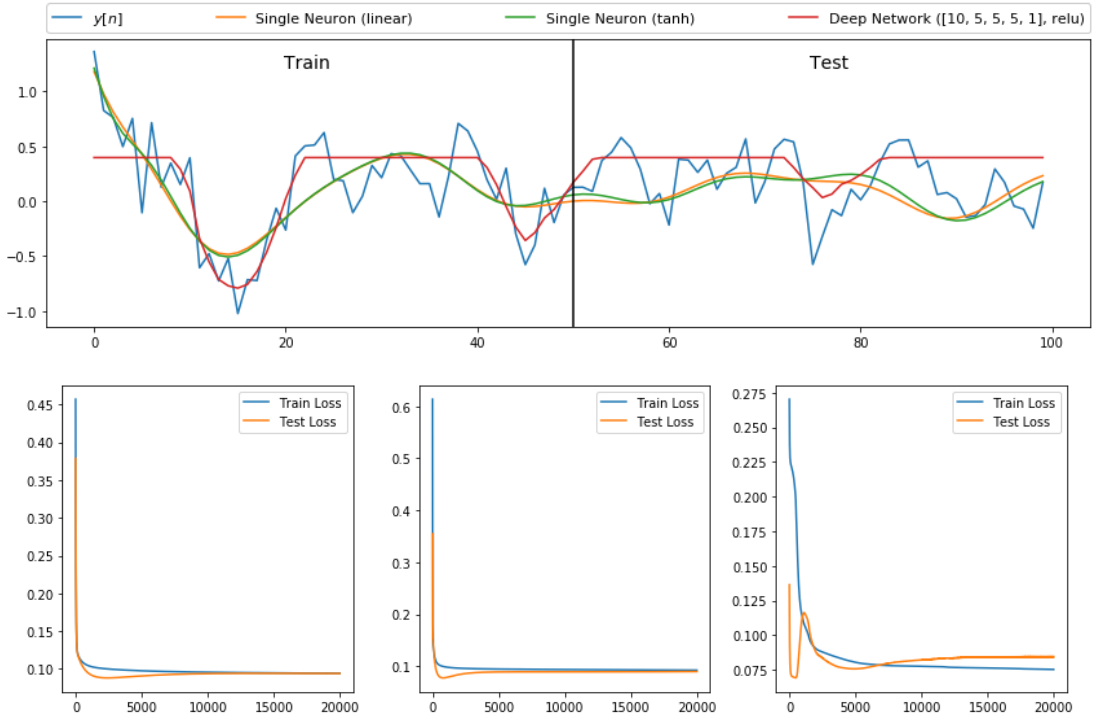

Fig. 48. (Top) Prediction result using single neuron with LMS, LMS with activation function tanh(), and the deep network. (Bottom) Learning curves. Noise power = 0.05.

8)

If the noise power is increased, the instability of the prediction result using deep network is increased. By increasing the noise power to 0.2, the prediction result and learning curves are shown in Fig. 49. By increasing the noise power from 0.5, the prediction result and learning curves are shown in Fig. 50.

From the following figures, although the learning curve of the training data using deep network can converge to a lower level than those of the LMS based algorithms, its performance is poor when the testing data is used. As shown in the figure, the learning curves of the testing data using deep network become unstable when the power of noise is increased. This phenomenon also happens to the learning curve of the training data, but not as significant.

The reason for this instability is overfitting. As the increase of the noise power, the deep network considers the noise corrupted signal as the clean signal that is required to be predicted. This can be observed from the top figures in Fig. 49 and Fig. 50, the predictions using deep network with the training signal match the change of the noise corrupted signals. However, the weights trained by overfitting the noise corrupted training signal do not fit the noise corrupted testing signal, hence the instability in the learning curves using the testing signal. The weights even do not fit some epochs of the training signal itself, which is visualised as the fluctuation in the learning curves using training signals.

In conclusion, the main drawback of the deep network is its poor performance in predicting signals that are corrupted by noise with high power, or in other word, signals

with a low signal to noise ratio (SNR). If the SNR is low, deep network tends to overfit the noise rather than fit the signal, which leads to unreliable trained weights. In addition, with many layers and neurons, the complexity of the prediction system is much higher than those algorithms that require one single neuron, for example, the LMS algorithm.

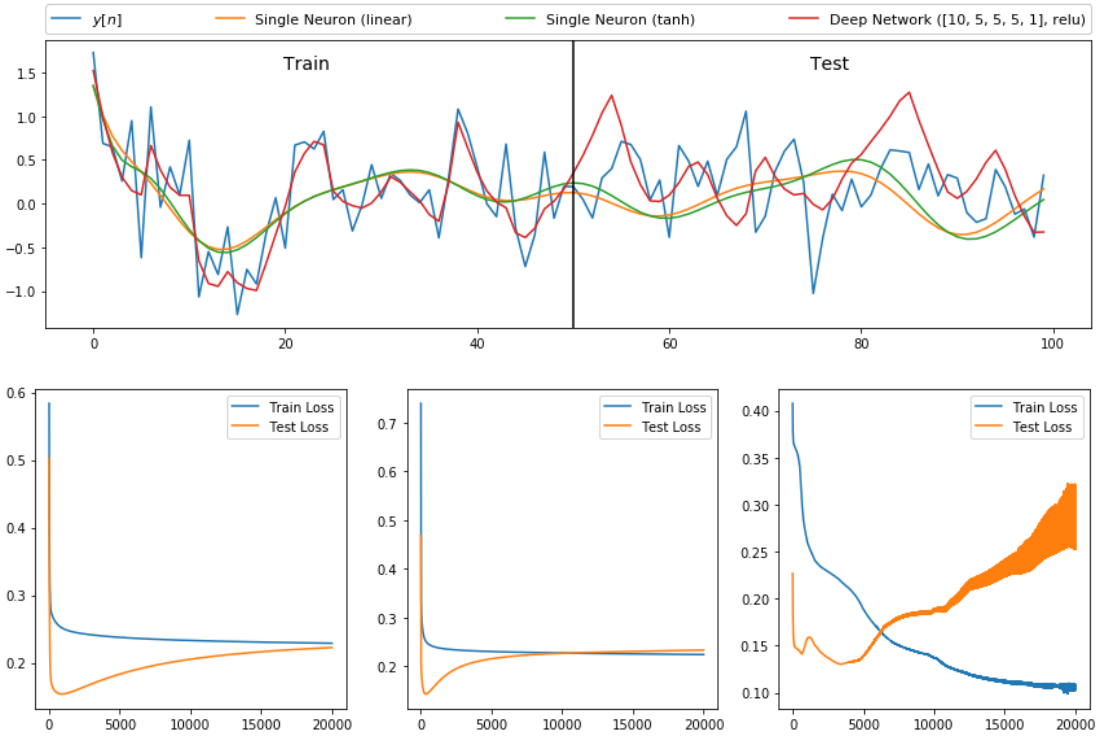

Fig. 49. (Top) Prediction result using single neuron with LMS, LMS with activation function tanh(), and deep network. (Bottom) Learning curves. Noise power = 0.2.

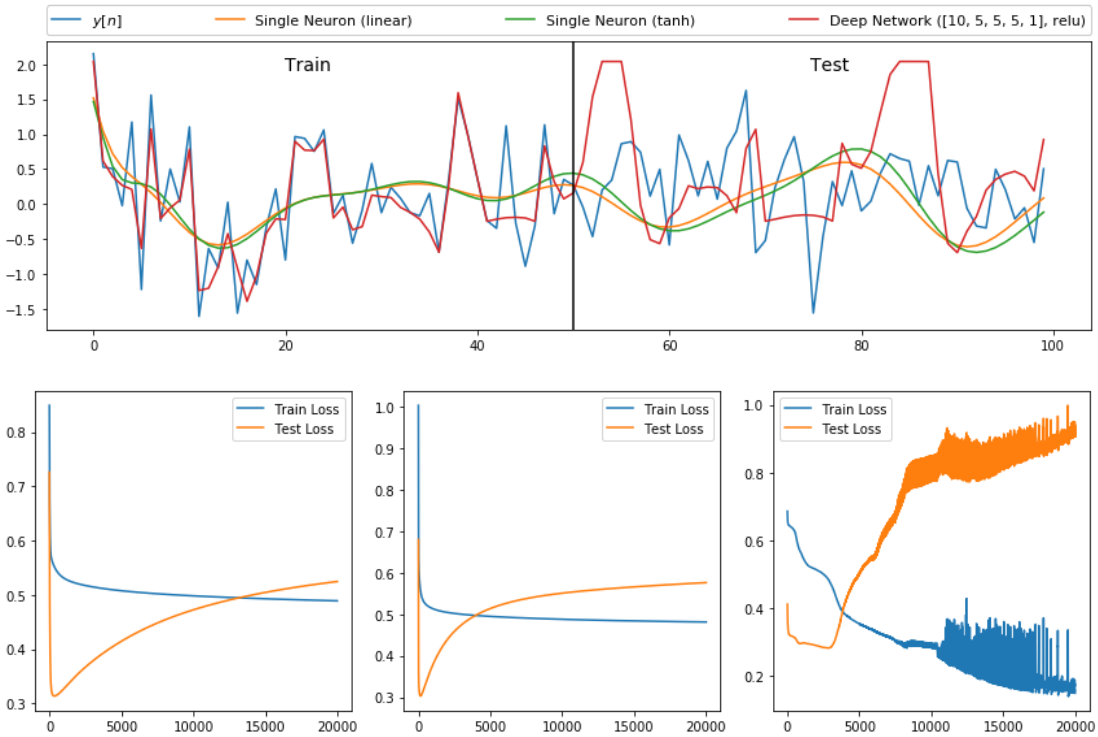

Fig. 50. (Top) Prediction result using single neuron with LMS, LMS with activation function tanh(), and deep network. (Bottom) Learning curves. Noise power = 0.5.