Statistical Signal Processing and Inference

Final Coursework

ELEC60002

(Trevi Yek) - CID02145518

Spring Term Coursework

Imperial College London

March 27, 2024

Contents

1 Random signals and stochastic processes

1.1 Statistical estimation

1.1.1 Calculate the expected value of X

The theoretical mean of a uniform distribution $X \sim U(a, b)$ is given by the formula derived below.

$$m = E\{X\} = \frac{1}{b-a}\int_a^b x dx = \frac{(b-a)(b+a)}{2} \cdot \frac{1}{b-a} = \frac{b+a}{2}$$

Hence with the uniform distribution $X \sim U(0, 1)$, m was calculated to be 0.5. Using the MATLAB *mean* function, the sample mean was calculated to be $\hat{m} = 0.494$ (3 s.f.) in the current realisation. This value varies slightly when different draws of random variables were used.

The accuracy of the sample mean as an estimator:

- Bias: It is an unbiased estimator.
- Accuracy: As sample size increases, the sample mean tends to converge to the population (true) mean. Also, the Central Limit Theorem states that the distribution of sample means approaches a normal distribution as sample sizes increase, although not as quickly in non-normal distributions.
- Outliers: Sensitive to outliers, but not relevant in this case due to no outliers in a uniform distribution.
- Efficiency: Does not have the minimum possible variance. So a wider distribution requires a larger sample size to achieve the same level of precision as a narrower distribution

1.1.2 Repeat the analysis for the standard deviation

The theoretical standard deviation for a uniform distribution is derived as such.

$$\begin{aligned}\sigma^2 &= E\{X - E\{X\}\}^2 = E\{X^2\} - (E\{X\}^2) = \frac{1}{b-a}\int_a^b x^2 dx - (\frac{b+a}{2})^2 \\ &= \frac{a^2+ab+b^2}{3} - \frac{a^2+2ab+b^2}{4} = \frac{(b-a)^2}{12} \\ \sigma &= \frac{(b-a)}{\sqrt{12}}\end{aligned}$$

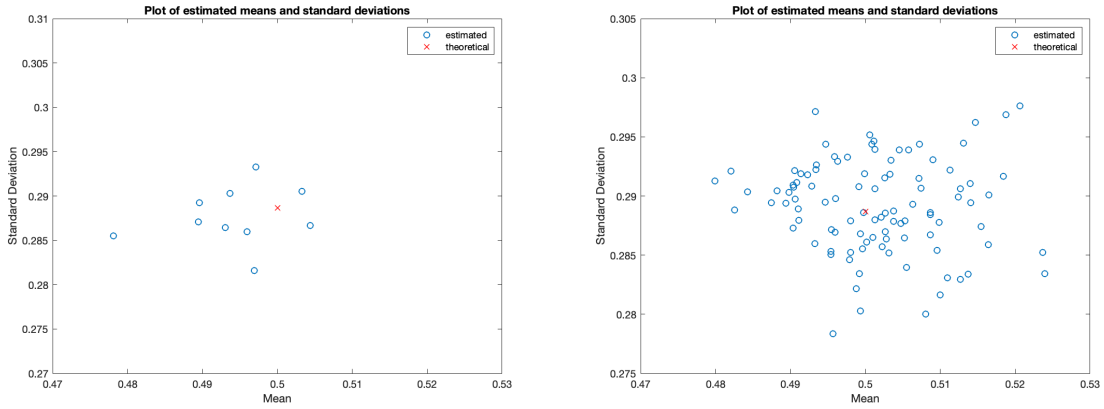
Hence, the theoretical standard deviation $\sigma = 0.289$ (3 s.f.). Using the MATLAB *std* function, the sample standard deviation was calculated to be $\hat{\sigma} = 0.291$ (3 s.f.) in the current execution. This value also varies slightly when different draws of random variables were used.

The accuracy of the sample standard deviation as an estimator:

- Bias: It is an unbiased estimator after the Bessel's correction is applied.
- Accuracy: As sample size increases, it converges to the population (true) standard deviation.
- Outlier: Same as in the mean
- Efficiency: Does not have the minimum possible variance, other estimators such as inter-quartile range may be more suitable.

1.1.3 How the estimates cluster about their theoretical values

Looking at the next figure (Figure 1.1), there was more positive biases in (a) for both means and standard deviations. With more realisations in (b), the spread becomes more even by optical observation. Also, the values of biases could be considered small.



(a) $\hat{m}_{1:10}$ against $\hat{\sigma}_{1:10}$ (b) $\hat{m}_{1:100}$ against $\hat{\sigma}_{1:100}$

Figure 1.1: Plot of $\hat{m}$ against $\hat{\sigma}$ (for 10 and 100 realisations)

1.1.4 Approximate the pdf of X

In Figure 1.2, histograms were generated using the MATLAB *histogram* function with the 'Normalization' parameter set to 'pdf'. We can deduce that the estimate converges to the theoretical PDF with an increase in number of generated samples. Increasing the number of bins produces a bigger observable variance.

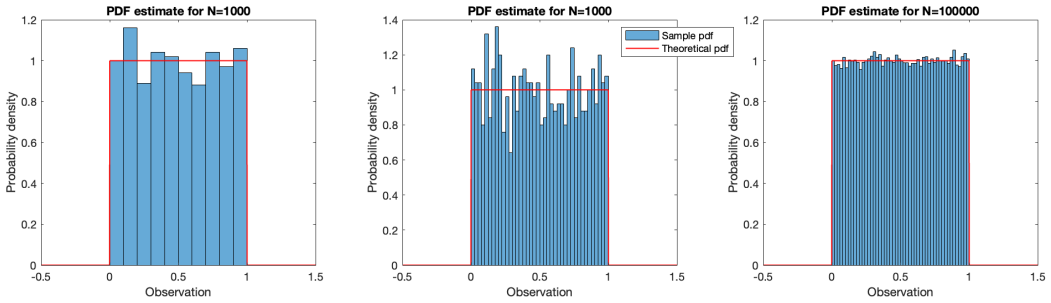


Figure 1.2: PDF plots for 1000-sample, with more bins and 100,000-sample signals

1.1.5 Repeat Part 1–Part 4 for zero-mean, unit std, Gaussian random variables

Zero-mean, unit standard deviation, Gaussian random variables is a phrase used to describe standard normal distribution random variables, $X \sim N(0,1)$. Hence, the theoretical mean is 0. Using the MATLAB *mean* function, the sample mean was calculated to be $\hat{m} = 0.0419$ (3 s.f.) in the current execution of the code. This value varies slightly when different draws of random variables were used.

The accuracy of the sample mean as an estimator:

- Bias: It is an unbiased estimator.
- Accuracy: As sample size increases, the sample mean tends to converge to the population (true) mean. The Central Limit Theorem applies very quickly.
- Outliers: Generally robust in the presence of outliers

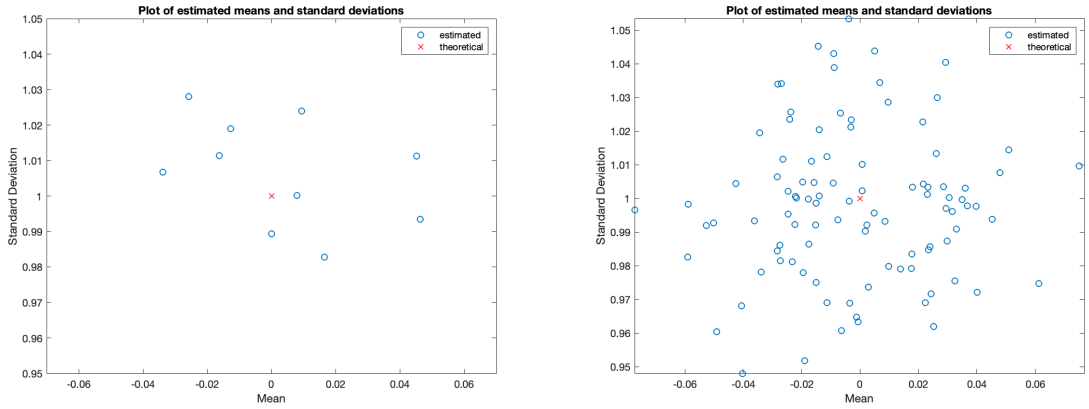
- Efficiency: For normal distribution, it is the most efficient measure out of all the unbiased estimators due to the minimum variance.

The theoretical standard deviation for a standard normal distribution is 1. Using the MATLAB *std* function, the sample standard deviation was calculated to be $\hat{\sigma} = 1.01$ (3 s.f.) in the current execution. This value also varies slightly when different draws of random variables were used.

The accuracy of the sample standard deviation as an estimator:

- Bias: It is an unbiased estimator after the Bessel's correction is applied.
- Accuracy: As sample size increases, it converges to the population (true) standard deviation.
- Outlier: Not robust against outliers
- Efficiency: Corrected standard deviation is more efficient.

Looking at the next figure (Figure 3), there was more negative biases in (a) for both means and standard deviations. With more realisations in (b), the spread becomes, visually, more even. Also, the values of biases could also be considered small.



(a) $\hat{m}_{1:10}$ against $\hat{\sigma}_{1:10}$ (b) $\hat{m}_{1:100}$ against $\hat{\sigma}_{1:100}$

Figure 1.3: Plot of $\hat{m}$ against $\hat{\sigma}$ (for 10 and 100 realisations)

In Figure 4, similar to the previous distribution, histograms were generated using the MATLAB *histogram* function with the 'Normalization' parameter set to 'pdf'. We can deduce that the estimate converges to the theoretical PDF with an increase in number of generated samples. Increasing the number of bins produces a bigger observable variance.

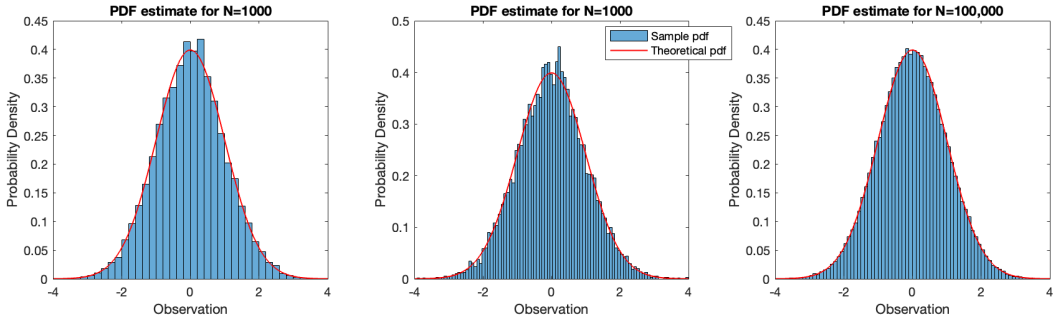


Figure 1.4: PDF plots for 1000-sample, with more bins and 100,000-sample signals

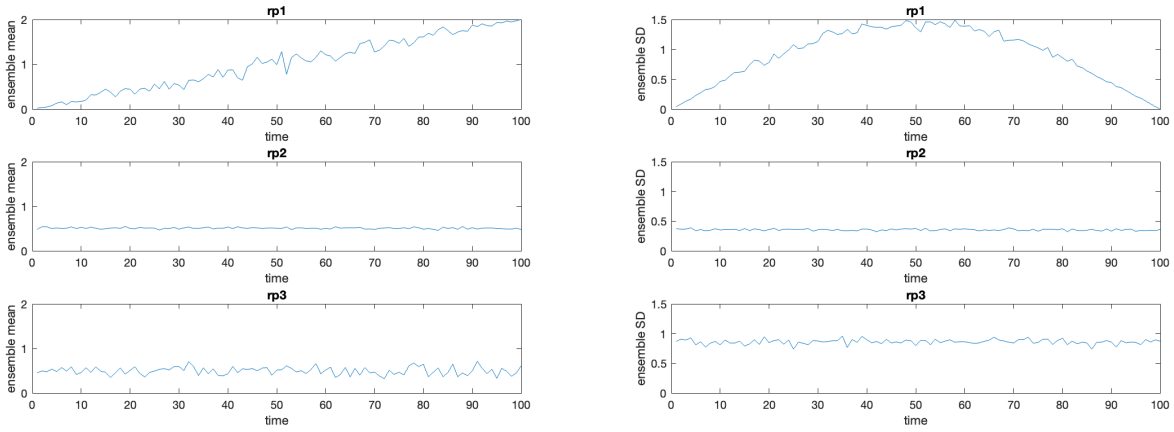
1.2 Stochastic processes

1.2.1 Compute the ensemble mean and standard deviation

(Averages in terms of mean and standard deviation)

- Stationarity: The ensemble average remains constant over time.
- Ergodic: The time average converges to the ensemble average as the observation time becomes sufficiently long.
- Note: A process can be stationary but not ergodic, while an ergodic process is always stationary

In Figure 5(a), we can see that rp1 has an approximately linear ensemble mean. The ensemble means of rp2 and rp3 are generally stable, at around 0.48 and 0.51 respectively, with the latter having bigger fluctuations.
In Figure 5(b), the ensemble standard deviation of rp1 concave downwards over time. The ensemble standard deviations of rp2 and rp3 are also stable, at around 0.33 and 0.87 respectively, also with the latter having bigger fluctuations. With these information, it is reasonable to conclude that rp2 and rp3 are stationary processes, where rp2 exhibits stronger characteristics of such process.



(a) ensemble mean

(b) ensemble standard deviation

Figure 1.5: Plot of averages as a function of time for M=100 and N=100

rp1 was concluded to be non-stationary, so it is definitely not ergodic.

1.2.2 Calculate the mean and std for each realisation

Tables 1 and 2 provides the time averages for the subsequent analysis. For rp2, while the mean values may be considered stable, the standard deviation is vastly different, varying hugely on a diffent execution. Hence rp2 is stationary but non-ergodic. For rp3, the mean and standard deviation values are quite stable, and they are approximately equal to the ensemble averages of 0.51 and 0.87 respectively. So rp3 is stationary and ergodic.

mean			
realisation	**rp1**	**rp2**	**rp3**
1st	10.0132	0.8064	0.4681
2nd	9.9944	0.6665	0.5098
3rd	10.0585	0.7286	0.5004
4th	10.0717	0.8561	0.4625

Table 1: Mean for each realisation, M=4 and N=1000

standard deviation			
realisation	**rp1**	**rp2**	**rp3**
1st	5.8729	0.0302	0.8793
2nd	5.8285	0.2278	0.8628
3rd	5.8370	0.0049	0.8791
4th	5.8593	0.1441	0.8854

Table 2: Standard deviation for each realisation, M=4 and N=1000

1.2.3 Mathematical descriptions

Note of clarification: the theoretical averages derived will be the same for each realisation unless stated otherwise.

For rp1:

Definition of rp1 is given by $v_1[n] = w[n] * 5sin(\frac{n\pi}{N}) + 0.02n$ where $w[n] \sim U(-0.5, 0.5)$ for each realisation. The theoretical averages are derived as such.

$$\begin{aligned} m &= E\{v_1[n]\} = E\{w[n] \cdot 5sin\left(\frac{n\pi}{N}\right) + 0.02n\} = E\{0.02n\} \\ &= \frac{1}{N}\int_{-\infty}^{+\infty} f(x)dx = \frac{0.02}{N}\int_0^N xdx = \frac{N}{100} \end{aligned}$$

$$\begin{aligned} \sigma^2 &= E\{(v_1[n] - m)^2\} = E\{(w[n] \cdot 5sin\left(\frac{n\pi}{N}\right))^2\} + E\{(0.02n - m)^2\} \\ &= E\{(w[n])^2\} \cdot E\{(5sin\left(\frac{n\pi}{N}\right))^2\} + E\{(0.02n)^2\} - m^2 \\ &= \frac{(1-0)}{12} \times \frac{5^2}{2} + \left(\frac{0.0004}{N}\int_0^N x^2dx - \left(\frac{N}{100}\right)^2\right) \\ &= \frac{25}{2(12)} + \left(\frac{N^2}{7500} - \frac{N^2}{10000}\right) = \frac{25}{24} + \frac{N^2}{30000} \qquad \sigma = \sqrt{\frac{25}{24} + \frac{N^2}{30000}} \end{aligned}$$

Taking the mean of the ensemble averages, $\hat{m} = 1.01$ and $\hat{\sigma} = 0.916$. Using the given value N=100, the theoretical values are $m = 1$ and $\sigma = 1.17$. Taking the mean of the time averages, $\hat{m} = 10.0$ and $\hat{\sigma} = 5.88$. Using the given value N=1000, the theoretical values are $m = 10$ and $\sigma = 5.86$. So the theoretical and sample averages agree with each other and the averages are not constant with the size of sample.

For rp2:

The signal is denoted by $v_2[n] = y_{1i}w[n] + y_{2i}$ where $w[n] \sim U(-0.5, 0.5)$ and (y_{1i}, y_{2i}) comes from $([y_1]^T, [y_2]^T) \sim U(0, 1)$ for each realisation i. So derivation will be made for many realisations.

$$\begin{aligned} m_{1:M} &= E\{([y_1]^T w[n] + [y_2]^T)\} = E\{[y_2]^T\} = \frac{(0+1)}{2} = 0.5 \\ \sigma^2_{1:M} &= E\{(([y_1]^T w[n] + [y_2]^T) - m_{1:M})^2\} = E\{([y_1]^T)^2\} \cdot E\{(w[n])^2\} + E\{([y_2]^T - E\{[y_2]^T\})^2\} \\ &= \frac{(0.5 - (-0.5))^2}{12} \cdot \frac{(1-0)^2}{12} + \frac{(1-0)^2}{12} = \frac{1}{144} + \frac{(1-0)}{12} = \frac{13}{144} \\ \sigma_{1:M} &= \frac{\sqrt{13}}{12} = 0.300 \end{aligned}$$

Taking the mean of the ensemble averages, $\hat{m} = 0.553$ and $\hat{\sigma} = 0.323$. The mean of the time averages are $\hat{m} = 0.398$ and $\hat{\sigma} = 0.109$. The theoretical values are $m = 0.5$ and $\sigma = 0.300$ which close to the ensemble averages. This makes sense because the mean and variance in each realisation will be different due to the constants (y_{1i}, y_{2i}).

For rp3:

The process can be defined as $v_3[n] = w[n] + 0.5$ where $w[n] \sim U(-1.5, 1.5)$. Hence the formulas for mean and standard deviation are derived as such.

$$m = E\{v_3[n]\} = E\{w[n] + 0.5\} = E\{0.5\} = 0.5$$

$$\sigma = E\{(v_3[n])^2\} = E\{(w[n] + 0.5)\} = \frac{(1.5 - (-1.5)}{\sqrt{12}} = 0.866$$

Taking the mean of the ensemble averages, $\hat{m} = 0.493$ and $\hat{\sigma} = 0.870$. The mean of the time averages are $\hat{m} = 0.499$ and $\hat{\sigma} = 0.853$. The theoretical values are $m = 0.5$ and $\sigma = 0.866$. So the theoretical and sample averages are close, so they also agree with each other.

1.3 Estimation of probability distributions

1.3.1 Test the function *my_pdf*

(Note: Although the instruction was to name the m-file 'pdf', it was named 'my_pdf' to avoid name clashes with another built in function by the same name.) The function was tested using a stationary process with a Gaussian pdf, shown in Figure 6.

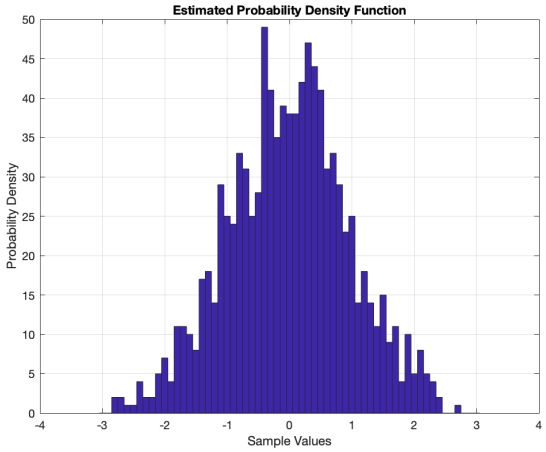


Figure 1.6: my_pdf tested with $v = randn(1, N = 1000)$

1.3.2 Stationary and ergodic process

From the previous part, rp3 is is the only one that is stationary and ergodic. So estimates of pdfs with increasing data length N was computed. As data length N increases, the pdf estimation approaches a uniform distribution of $X \sim U(-1, 2)$, i.e. the spread becomes more even.

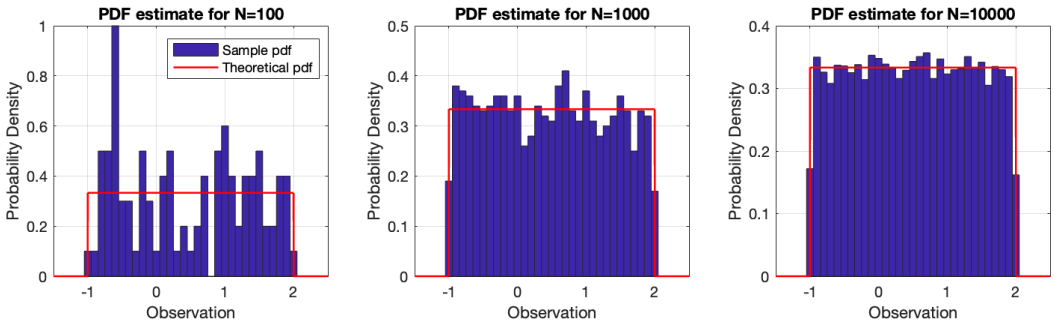


Figure 1.7: Estimated PDF for rp3 with N=100,1000,10000, and the theoretical PDF

1.3.3 Non-stationary process

my_pdf cannot be used to estimate the pdf of a non-stationary process, where the mean changes over time. Take a look at the case when the mean of a 1000-sample-long signal changes from 0 to 1, after sample point N = 500. The difficulty arises from the fact that a simple histogram will not effectively represent the change, as the transition introduces discontinuity.
One approach, specifically for this case, would be to seperately compute the pdf estimate with two windows (N = 1 to N = 500 to N = 501 to N = 1000). This produces 2 pdfs, one with mean 0 and the other with mean 1.
This method can only be utilised under the assumptions that the two pdfs are independent, each segment is stationary and the change in mean is instantaneous.

2 Linear stochastic modelling

2.1 ACF of uncorrelated and correlated sequences

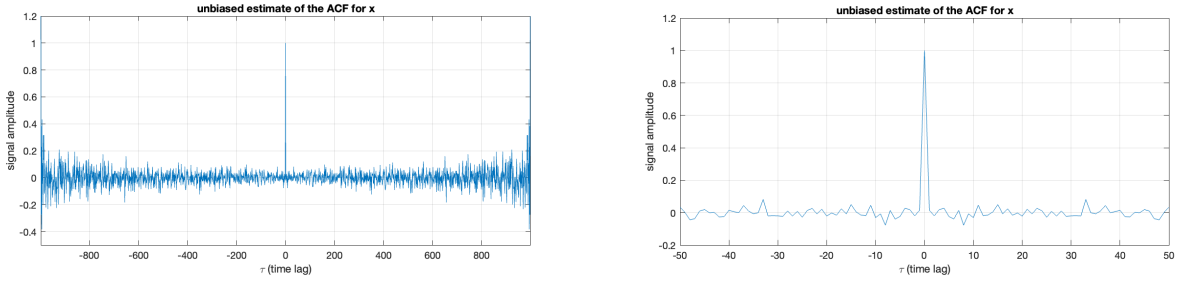
2.1.1 The unbiased estimate of the ACF for a 1000-sample realisation of WGN

For a White Gaussian Noise, a real stochastic process that is stationary and independent with zero-mean and unit-variance, the theoretical ACF is denoted by a discrete Dirac function.

$$R_{WGN}(\tau) = \begin{cases} E\{x^2[n]\} = \sigma^2 + \mu^2 = \sigma^2 = 1 & for\ \tau = 0 \\ E\{x[n]x[n+\tau]\} = \mu^2 = 0 & for\ \tau \neq 0 \end{cases}$$

The function $xcorr(x, 'unbiased')$ from MATLAB was used to calculate the unbiased estimate of the ACF for a 1000-sample realisation of WGN The plot is shown in Figure 8(a). The formula of autocorrelation function corrected for the bias is provided below.

$$\hat{R}_X(\tau) = \frac{1}{N - |\tau|} \sum_{n=0}^{N-|\tau|-1} x[n]x[n+\tau] \qquad where\ \tau = -N+1, ..., N-1$$



(a) Displayed for $\tau \in [-999 : 999]$

(b) Zoomed onto region $\tau < |50|$

Figure 2.1: Plot of the unbiased estimate of the ACF for a 1000-sample realisation of WGN

Due to the small or finite sample size,the ACF estimate deviates from the theoretical AFC. So, it should converge to the theoretical ACF when sample size approaches infinity. Also, the estimate is symmetrical around zero lag, $\tau = 0$, because $R_X(-\tau) = R_X(\tau)$.

2.1.2 Focus on region $\tau < |50|$

The "zoom" command was utilized to focus on the region $\tau < |50|$, shown in Figure 8(b). For smaller lag times, the ACF estimate has smaller fluctuations meaning that the value of a specific time is way less correlated to any past or future signal values. Hence it is closer to the theoretical ACF.
However, for the larger time lags, especially as τ approaches $N-1$ and $-N+1$, there seems to be very high correlation between signal values. This is due to the fact that the number of samples used to calculate the ACF estimate decreases as time lag increases. This leads to more discrepancy between the theoretical and estimated ACFs.

2.1.3 the effects of a large autocorrelation lag on the accuracy of ACF estimates

Looking at the formula corrected for the bias presented in a finite number of samples (displayed above), the time lag affects the factor $\frac{1}{N-|\tau|}$ and the number of summation terms $N-|\tau|$ in the equation. So the estimate becomes more different compared to the theoretical ACF (provided below) when $|\tau|$increases.

$$\hat{R}_X(\tau) = \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} x[n]x[n+\tau] \qquad where\ \tau = -N+1, ..., N-1$$

The variance of the ACF estimate is derived from the formula as shown, taking into consideration that $x[n]$ is zero-mean and unit-variance. It shows that variance increases if the number of samples N is decreased or when time lag is larger, leading to reduced accuracy. In other words, for a given number of samples, the ACF estimated at larger lag times (near N) are less statistically reliable.

$$\begin{aligned} Var(\hat{R}_X(\tau)) &= \frac{1}{(N-|\tau|)^2} \sum_{n=0}^{N-|\tau|-1} Var\left(x[n]x[n+\tau]\right) \\ &= \frac{(N-|\tau|)}{(N-|\tau|)^2} \left(E\{x^2[n]x^2[n+\tau]\} - (E\{x[n]x[n+\tau]\})^2\right) \\ &= \frac{1}{N-|\tau|} \left(Var(x[n])Var(x[n+\tau] + \mu_{X_n} Var(x[n+\tau]) + \mu_{X_{n+\tau}} Var(x[n])\right) \\ &= \frac{1}{N-|\tau|} \end{aligned}$$

The empirical rule, or sometimes called the three-sigma rule, is a statistical rule stating that for normally distributed data, almost all observed data will fall within three standard deviations of the mean of the data. It usually used to set the upper and lower limits in statistical analysis. So for the bound of $|\tau|$, we first set an upper and lower limit of 3 standard errors around the (zero) mean, giving a 99.7% confidence interval. Outliers will occur 0.3% of the time, so for 1000 samples, the empirical bound is obtained when the ACF value calculated goes beyond the limits for the third time.

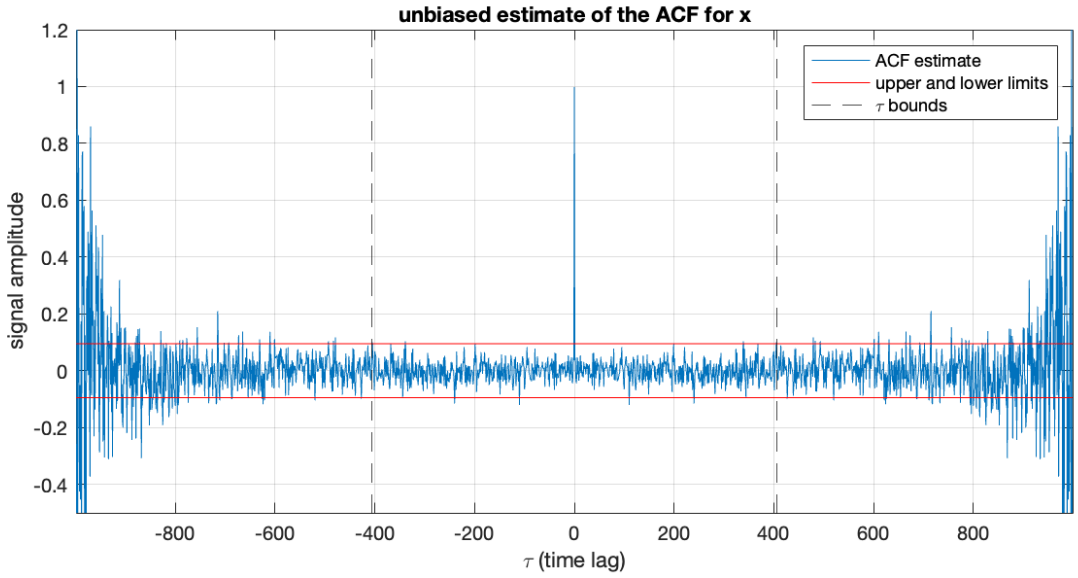


Figure 2.2: Plot showing the empirical bound for $|\tau|$

Illustrated by the plot above, the value of 3 standard errors is 0.949 and the value of the empirical bound of $|\tau|$ is found to be 406.

2.1.4 Filtering a 1000-sample WGN with Moving Average (MA) filter

A 1000-sample WGN is filtered by MA(9) filter with unit coefficients. The its ACF is plotted and displayed in Figure 2.3.
For lags greater than 8, the estimated ACF is similar to the unfiltered values in the previous sections where the ACF values are very small. However, for $|\tau| <= 8$, the output comprises of the sum of the current value and the previous 8 samples due to the MA(9) filter. So the significance of the correlation amplitude begins at a wider window of time lags around $\tau = 0$. Take a look at the formulaic representation of the filter $y = filter(ones(9,1),[1],x)$ provided below.

$$y[n] = (x[n] + x[n-1] + ... + x[n-(M-1)]) = \sum_{m=0}^{M-1} x[n-m] \qquad \textit{where } M \textit{ is the order}$$

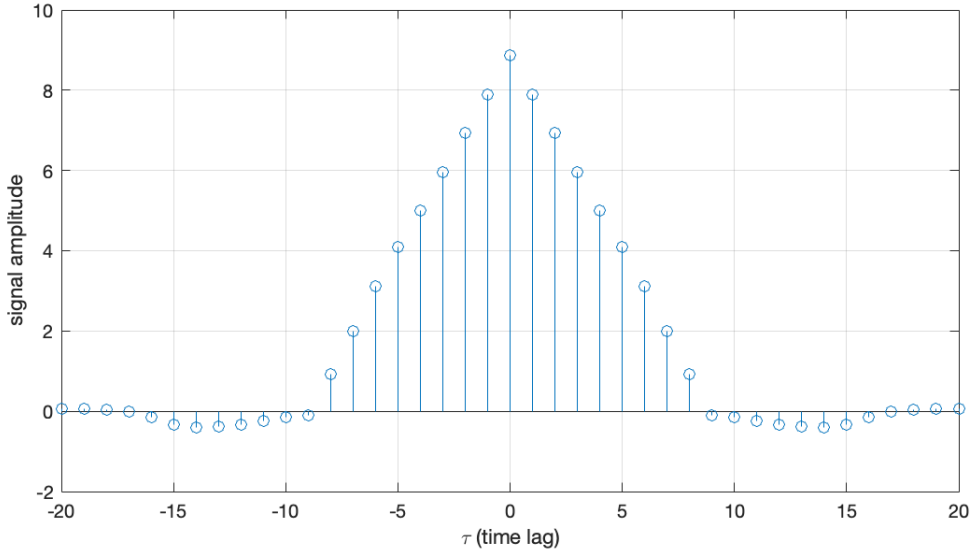


Figure 2.3: Plot of ACF estimate of the WGN filtered by MA filter for $|\tau| < 20$

The filter could also be represented by the convolution of $x[n]$ with $\sum_{m=0}^{M-1} \delta[n-m]$ (a rectangular window). Combined with the fact that ACF produces a symmetrical (time-inverted) copy which agrees with the observation that the shape of the central peak is triangular. The higher the order, or the bigger the value of M, the wider the base of the triangle. By going along with this logic, you can deduce that the sample mean can be calculated using an MA filter of the 1000th order (across the whole sample) divided by the order.

$$\hat{\mu} = \frac{1}{N} \sum_{m=0}^{N-1} x[n-m] \qquad \textit{where signal length N=M=1000 (the order)}$$

2.1.5 ACF of a filtered stochastic process

The stochastic process Y_n is the filtered version of the process X_n and the $R_Y(\tau)$ is a the ACF of a realisation of Y_n. Given that $R_Y(\tau)$ is the convolution of the ACF of the filter's impulse response ($R_h(\tau)$) and the ACF of X_n ($R_X(\tau)$), we get:

$$R_Y(\tau) = R_X(\tau) * R_h(\tau)$$

Assuming that the input X_n is independent, we can conclude that $R_X(\tau) = \sigma_X^2 \delta(\tau) + \mu_X^2$ because its ACF has the property of a Dirac Delta function. Hence by substituting $R_X(\tau)$ into $R_Y(\tau)$, we get:

$$R_Y(\tau) = (\sigma_X^2 \delta(\tau) + \mu_X^2) * R_h(\tau) = \sigma_X^2 (\delta(\tau) * R_h(\tau)) + \mu_X^2 * R_h(\tau)$$

$$= \sigma_X^2 R_h(\tau) + \mu_X^2 \sum_{\tau=-N}^{N} R_h(\tau) \qquad \textit{where N=length of } R_h(\tau)$$

So it is deduced that $R_Y(\tau)$ is the linearly scaled version of the filter's ACF, by the magnitude of the variance of X_n, that has an offset.

2.2 Cross-correlation function

2.2.1 CCF for x and y

Vectors x and y are the realisations of processes X_n and Y_n respectfully. The estimate of the CCF for sequences x and y is obtained using the MATLAB command $xcorr(x, y, \textit{'unbiased'})$ and presented in Figure 2.4.

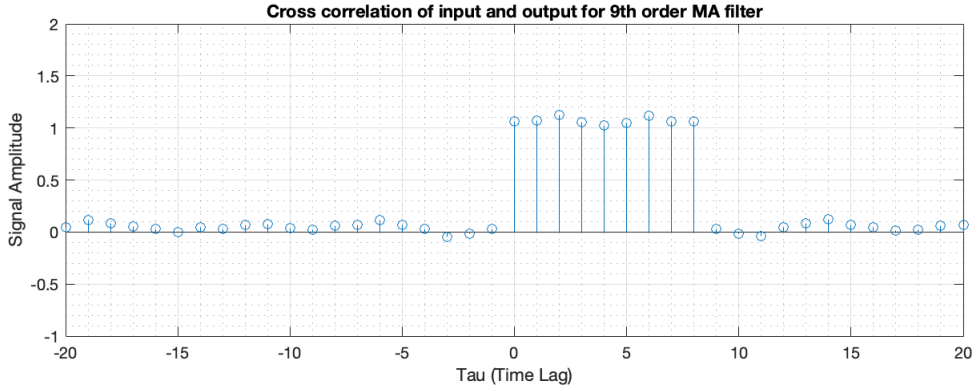


Figure 2.4: CCF estimate of x and y

There are significant correlations of approximately 1 for the time lags $0 \leq \tau \leq 8$, with the number of τs corresponding to the filter's order. The cross-correlation function is computed with the following equation, noting that $y[n] = h[n] * x[n]$:

$$R_{XY}(\tau) = E\{x[n]y[n-\tau]\} = E\left\{x[n] \sum_{k=-\infty}^{+\infty} h[k]x[n-\tau-k]\right\} = \sum_{k=-\infty}^{+\infty} h[k]E\{x[n]x[n-\tau-k]\}$$
$$= \sum_{k=-\infty}^{+\infty} h[k]R_X(-(\tau+k)) = h[\tau]R_X(\tau)$$

So if X_n is an uncorrelated stochastic process (or a WGN), its CCF is a Dirac delta function $R_X = \delta\tau$. So, convolving it with an MA filter of order 9 with unit coefficients will produce an discrete delta train for values $0 \leq \tau \leq 8$.

2.2.2 System identification

For an uncorrelated stochastic process X_n and its output from the filter Y_n, the CCF estimate, using the zero-mean input, can reveal the expected shape of the impulse response filter used. We can derive this using $h[\tau]$, the filter's impulse response with an assumption of the input having an offset.

$$R_Y(\tau) = h(-\tau) * R_X(\tau) = h(-\tau)(\sigma_X^2\delta(\tau) + \mu_X^2) = \sigma_X^2 h(-\tau) + \mu_X^2 \sum_{\tau=-N}^{+N} h(\tau)$$

Having a zero-mean input would remove the second term in the set of addition. This means that the filter order (or the MA order) can be determined by the number of peaks in the CCF estimate of X_n and Y_n, and the coefficients can be deduced from the height of each peak.

2.3 Autoregressive modelling

2.3.1 Stability of AR(2) process

A generated sample of 100 pairs, as 1000-sample AR(2) process coefficients, are plotted against each other in Figure 2.5, denoted by a_1 and a_2.
The conditions for stability, and proof for the triangular shape of the convergence region, is deduced from the equation for an AR(2) process $x[n] = a_1x[n-1]+a_2x[n-2]+w[n]$. The stability is determined by the pole, where z lies within unit circle:

$$\textit{Z-transform of x[n]:} \qquad X(z) = (a_1z^{-1} + a_2x^{-2})X(z) + W(z)$$

$$\frac{X(z)}{W(z)} = H(z) = \frac{1}{1 - a_1z^{-1} - a_2x^{-2}} = \frac{z^2}{z^2 - a_1z - a_2} \qquad \textit{With the root: } z = \frac{a_1 \pm \sqrt{a_1^2 + 4a_2}}{2}$$

$$\textit{Since } |z| < 1, \qquad -1 < \frac{a_1 \pm \sqrt{a_1^2 + 4a_2}}{2} < 1 \quad \Longrightarrow \quad -2 < a_1 \pm \sqrt{a_1^2 + 4a_2} < 2$$

When $\sqrt{b^2 - 4ac}$ is real:

$$a_1 \pm \sqrt{a_1^2 + 4a_2} < 2 \quad \Longrightarrow \quad a_1^2 + 4a_2 < (2 - a_1)^2 \quad \Longrightarrow \quad \cancel{a_1^2} + 4a_2 < 4 - 4a_1 + \cancel{a_1^2}$$

Hence, 1st condition: $a_1 + a_2 < 1$

$$a_1 \pm \sqrt{a_1^2 + 4a_2} < -2 \quad \Longrightarrow \quad a_1^2 + 4a_2 < (-2 - a_1)^2 \quad \Longrightarrow \quad \cancel{a_1^2} + 4a_2 < 4 + 4a_1 + \cancel{a_1^2}$$

Hence, 2nd condition: $a_2 - a_1 < 1$

When $\sqrt{b^2 - 4ac}$ is complex:

$$\textit{Taking: } z = \frac{a_1 \pm j\sqrt{-(a_1^2 + 4a_2)}}{2}, \qquad |z|^2 = \left(\frac{a_1}{2}\right)^2 + \left(\frac{\sqrt{-(a_1^2 + 4a_2)}}{2}\right)^2 = -a_2$$

Hence, 3rd condition: $-1 < a_1 < 1$ *(Because* $|z| < 1$*)*

From the derivation above, we can conclude that the stability conditions, demonstrated by the stability triangle, are $(a_1 + a_2 < 1)$, $(a_2 - a_1 < 1)$ and $(-1 < a_1 < 1)$.

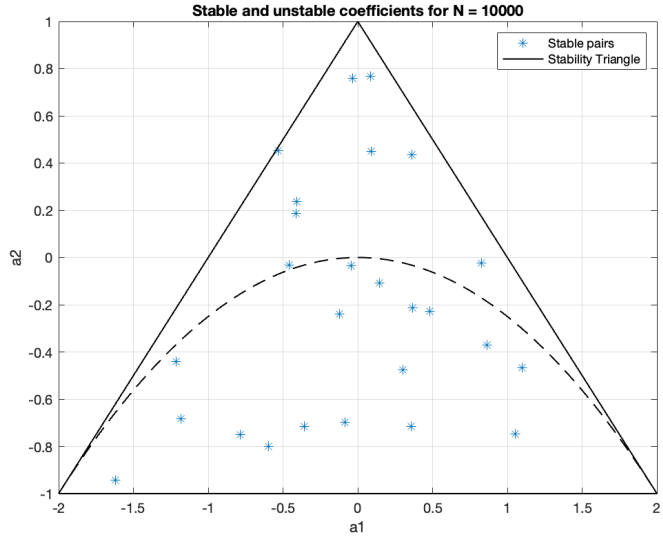


Figure 2.5: Plot of 100-sample stable pairs

As observed in Figure 2.5, the stable sets of coefficients lie within the stability triangle provided.

2.3.2 ACF of real world sunspot time series

The ACF of real world sunspot time series and its xero-mean version was plotted for data lengths $N = 5, 20, 250$ in Figure 2.6.
The original ACF estimate for $N = 5$ has an approximately triangular shape because the mean value is large in comparison to any meaningful variations in the data and it overwhelms the useful information. For the zero-mean ACF, you can deduce from the convex triangular shape that there is more correlation near xero time lag. As $N = 20$, the original ACF becomes sinusoidal where the correlation at zero lag is smaller than the peaks of the waves on either side, due to the interference form the mean. The zero-mean ACF shows the same sinusoidal behavior with reducing peak height further away from zero lag. However, when $N = 250$, the overall shape of the original ACF is the same as the zero-mean ACF, but with a significant offset due to the mean and gives the false notion of correlations.

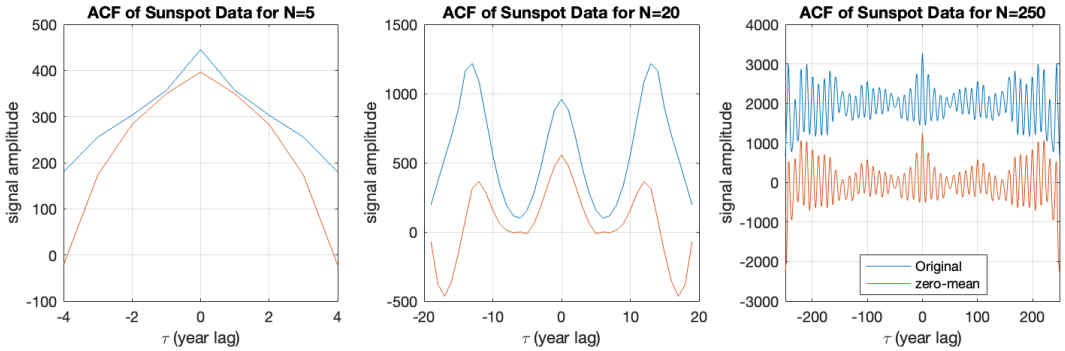


Figure 2.6: ACF estimates of the sunspot time series (original and zero-mean)

We know that a periodic ACF is derived from a periodic function, so there seems to be a pattern of similarity in sunspot values emerging every 12 years.

2.3.3 Yule-Walker equations

The Yule-Walker equations were used to calculate the partial correlation functions up until the model order $p = 10$. It can be concluded that the most likely model order of sunspot time series is 2, AR(2), where the trend of significant PACF values end.

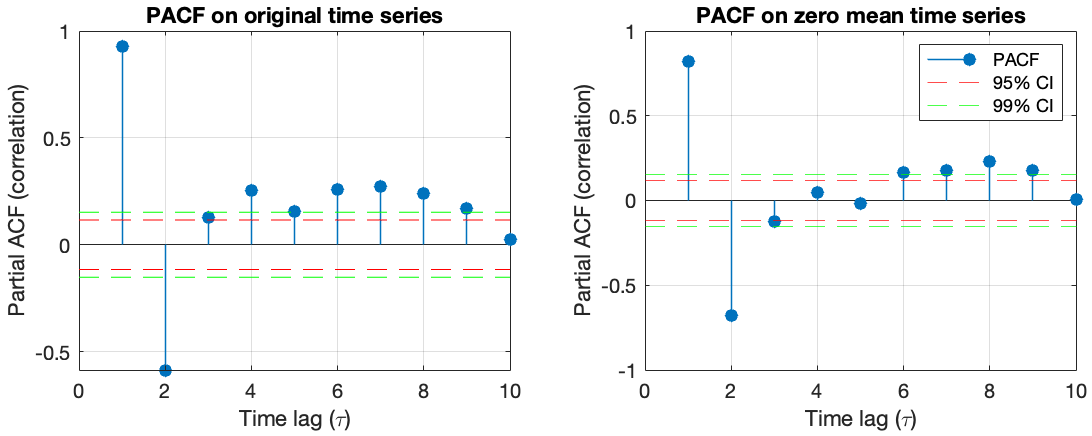


Figure 2.7: PACF

Then procedure was repeated for the series standardised to its Z-score values. You can observe the smaller PACF values after the proposed time lag of 2 days, meaning that the offset has to be taken into consideration in the non-centered data when modelling. So it is always a good idea to standardise the data before modelling.

2.3.4 Using ADL and MDL to determine the correct model order

A greater order for AR models would be more accurate, albeit more complex. So, in this case, the "correct" model order would one with the minimal complexity and demonstrating adequate accuracy. Which is why the three criterias, the minimum description length (MDL),the Akaike information criterion (AIC) and the corrected Akaike information criterion (AIC_c), are used to guage the optimal order p.
In Figure 2.8, the value of the optimal order is usually taken from the minimum value for the criterion, which is 9 for AIC and MDL. However, since the sample size of the data is small, AIC_c is better suited because it corrects for the possibility of overfitting. From this criteria, we get a minima at order 2. As a sidenote, even if we make a decision based on MCL and AIC, the decrease in the criterion magnitudes are not significant enough against the increase in model complexity to be desirable.

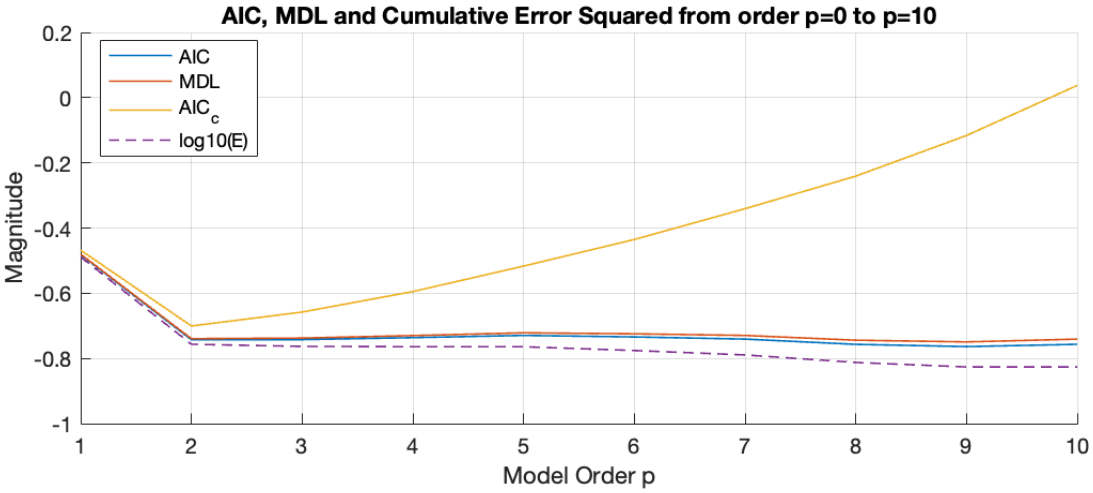


Figure 2.8: PACF

2.3.5 AR model to predict the sunspot time series

The AR(1), AR(2) and AR(10) models of the sunspots for the prediction horizons $m = 1, 2, 5, 10$ was plotted with the actual values.

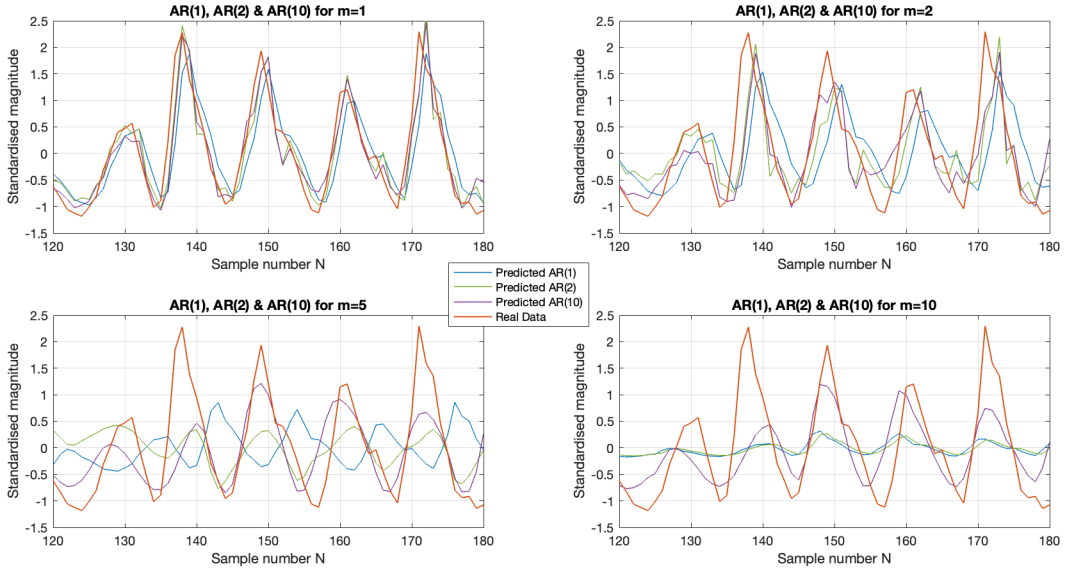


Figure 2.9: AR models of the sunspots for the prediction horizons $m = 1, 2, 5, 10$

Looking at Figure 2.9, we can see that the greater the prediction horizon, the greater the prediction

error and model order 10 has the least prediction error for all the prediction horizons. It is obvious that the error of AR(1) is the greatest while AR(2) and AR(10) have similar errors. This is because AR(1) does not present enough degrees of freedom to fully obtain the information in the data which is called under-fitting. So it predicts poorly. For AR(10), we an observe over-fitting where the information from the noise was also captured. The prediction error decreases with increasing model order, pointing to the trade off between model order and prediction error.

AR(2) is the correct model for this dataset because it produces reliable results at lower prediction horizons. At bigger prediction horizons, a higher order should be used.

2.4 Cramer-Rao Lower Bound

2.4.1 Sufficiency of AR(1)

The optimal model of AR model for the NASDAQ financial index data is determined by utilising both PACF and information theoretic criteria (AIC, MDL and AICc).

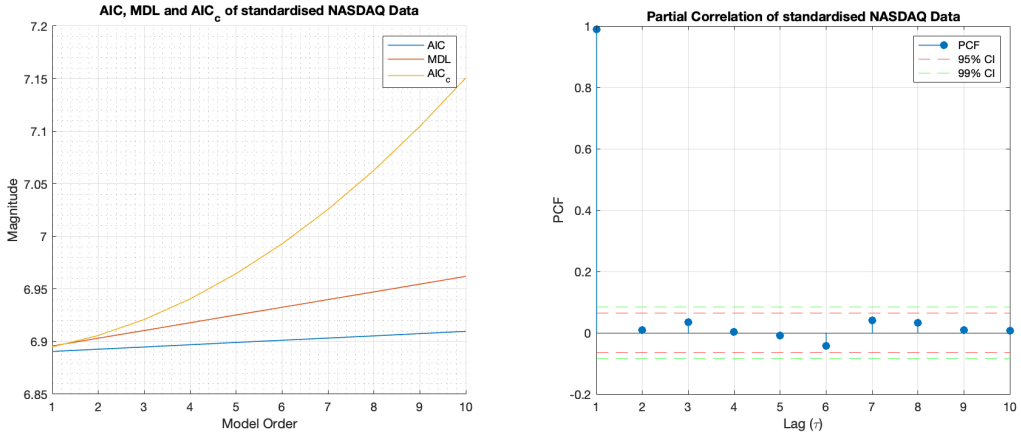


Figure 2.10: Cramer Rao

The PACF values for lags greater than 1 are not statistically significant and all the criterion show a minimum at model order 1. They all suggest that the correct model order is 1.

2.4.2 The Fisher's Information matrix

The Fisher's Information matrix is given by:

$$[I(\theta)]_{ij} = \frac{N}{2}\int_{-\frac{1}{2}}^{\frac{1}{2}} \frac{\partial ln[\hat{P}_X(f;\theta)]}{\partial\theta_i}\frac{\partial ln[\hat{P}_X(f;\theta)]}{\partial\theta_j}df$$

Assumptions: modelled by AR(1), $\theta = [\alpha_1, \sigma^2]$, $[I(\theta)]_{12} = [I(\theta)]_{21} = 0$ and $[I(\theta)]_{11=\frac{Nr_{xx}(0)}{\sigma^2}}$
So the value of $[I(\theta)]_{22}$ is derived as:

$$[I(\theta)]_{22} = \frac{N}{2}\int_{-\frac{1}{2}}^{\frac{1}{2}}\left(\frac{\partial ln[\hat{P}_X(f;\theta)]}{\partial\sigma^2}\right)^2 df = \frac{N}{2}\int_{-\frac{1}{2}}^{\frac{1}{2}}\left(\frac{1}{\sigma^2}\right)^2 df = \frac{N}{2\sigma^4}[f]_{-\frac{1}{2}}^{\frac{1}{2}} = \frac{N}{2\sigma^4}$$

$$\textit{Therefore:} \qquad [I(\theta)] = \begin{bmatrix} \frac{Nr_{xx}(0)}{\sigma^2} & 0 \\ 0 & \frac{N}{2\sigma^4} \end{bmatrix}$$

where $r_{xx}(0)$ is the autocorrelation of process at zero lag, σ^2 is the variance of input.

2.4.3 CRLB for both $\hat{\sigma}^2$ and $\hat{\alpha}_1$

The CRLB of an unbiased estimator is found to be the inverse of the Fisher's Information matrix.

$$Var(\hat{\theta}_i) \geq [I^{-1}(\theta)]_{ii}$$

The inverse for the Fisher's Information matrix is calculated to be:

$$I^{-1}(\theta) = \begin{bmatrix} \frac{\sigma^2}{Nr_{xx}(0)} & 0 \\ 0 & \frac{2\sigma^4}{N} \end{bmatrix} \qquad where: \quad Var(\hat{\sigma}^2) \geq \frac{2\sigma^4}{N} \ and \ Var(\hat{\alpha}_1) \geq \frac{\sigma^2}{Nr_{xx}(0)}$$

To find $r_{xx}(0)$, we use the equation for AR(1), given by $x[n] = \alpha_1 x[n-1] + w[n]$ where $w[n] \sim N(0, \sigma^2)$. Multiply with $x[n-\tau]$ to get the autocorrelation sequence:

$$\begin{aligned} x[n]x[n-\tau] &= \alpha_1 x[n-1]x[n-\tau] + w[n]x[n-\tau] \\ E\{x[n]x[n-\tau]\} &= E\{\alpha_1 x[n-1]x[n-\tau]\} + E\{w[n]x[n-\tau]\} \\ r_{xx}(\tau) &= \alpha_1 r_{xx}(\tau-1) + E\{w[n]x[n-\tau]\} \quad where: \ E\{w[n]x[n-\tau]\} = \begin{cases} \sigma^2, & if \ \tau = 0 \\ 0, & for \ |\tau| > 0 \end{cases} \\ substitute: \ \tau = 0, 1 \qquad & r_{xx}(0) = \alpha_1 r_{xx}(-1) + \sigma^2 \ and \ r_{xx}(1) = \alpha_1 r_{xx}(0) \\ r_{xx}(-1) = r_{xx}(1) \qquad & therefore: \ r_{xx}(0) = \alpha_1^2 r_{xx}(0) + \sigma^2 \alpha_1 \\ r_{xx}(0)(1-\alpha_1^2) = \sigma^2 \quad & hence: \ r_{xx}(0) = \frac{\sigma^2}{(1-\alpha_1^2)} \end{aligned}$$

So the CRLB for each estimator is:

- $Var(\hat{\sigma}^2) \geq \frac{2\sigma^4}{N}$
- $Var(\hat{\alpha}_1) \geq \frac{1}{N}(1-\alpha_1^2)$

The CRLB for both $\hat{\sigma}^2$ and $\hat{\alpha}_1$ was plotted for the number of data points, N, and the true variance of the driving noise, σ^2, both with ranges of $[1:50:1001]$ in MATLAB notation.
For both plots (Figure 2.11), the CRLB estimates have a minimum at $N = 1001$ and $\sigma^2 = 1$, and a maximum at $N = 1$ and $\sigma^2 = 1001$. This agrees with the theoretical derivation of the CRLB for both estimators, where the higher the driving noise variance, the higher the CRLB value. Due to the 10^6 order difference between the minimum and maximum values, the log_{10} values for the CRLB is displayed on the figures.

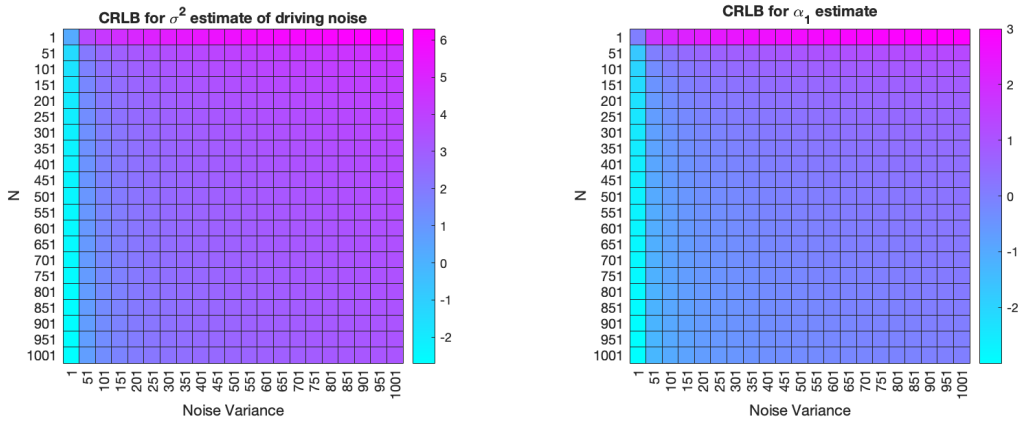


Figure 2.11: CRLB

The value of $\hat{\alpha}_1$ was determined by the AR(1) model of the financial dataset to be $\hat{\alpha}_1 = -0.9989$. So the CRLB for $\hat{\alpha}_1$ is $var(\hat{\alpha}_1) \geq \frac{1}{N}(1-\hat{\alpha}_1^2) = frac1924(1-(-0.9989)^2) = 2.3796 \times 10^{-6}$.
As $\hat{\alpha}_1$ approaches unity, then the $var(\hat{\alpha}_1)$ approaches zero. The filter's frequency response is

$$H(f) = \frac{1}{1-\alpha_1 e^{-j\omega}}$$

so its pole is equal to α_1. Which means that as α_1 approaches 1, its dependence on its previous values become stronger and the autocorrelation function decays much more slowly. Also, you could say that the model exhibits long memory characteristics, for example a financial shock would have longer-lasting effects on the end-of-day prices. This effect is displayed with the power spectral density of the AR(1) model,

$$P_X(f:\boldsymbol{\theta}) = |H(f)|^2 P_{WGN}(f) = \frac{\sigma^2}{|1-\alpha_1 e^{-j2\pi f}|^2}$$

where the peak becomes sharper (or higher) when α_1 approaches unity, meaning that the process parameters are more accurately estimated.

2.4.4 CRLB of PSD estimate

From the CRLB, it can be shown that

$$var(\hat{P}_X(f;\boldsymbol{\theta}))) \geq \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})^T}{\partial \boldsymbol{\theta}} \boldsymbol{I}^-(\boldsymbol{\theta}) \frac{\partial \hat{P}_X(f;\boldsymbol{\theta}))}{\partial \boldsymbol{\theta}} \quad where\ \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \left[\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \alpha_1}, \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \sigma^2}\right]^T$$

In terms of $A(f) = 1-\alpha_1 e^{-j2\pi f}$, we substitute it to get $P_X(f:\boldsymbol{\theta}) = \frac{\sigma^2}{|A(f)|^2}$. Then we can calculate the derivatives for α_1 and σ^2:

$$\begin{aligned}
\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \alpha_1} &= \frac{\partial}{\partial \alpha_1}\left(\frac{\sigma^2}{A(f)A^*(f)}\right) = -\frac{\sigma^2}{(A(f)A^*(f))^2}\left(A(f)\frac{\partial A^*(f)}{\partial \alpha_1} + A^*(f)\frac{\partial A(f)}{\partial \alpha_1}\right) \\
&= -\frac{\sigma^2}{|A(f)|^4}\left(A(f)e^{j2\pi f} + A^*(f)e^{-j2\pi f}\right) \\
\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \sigma^2} &= \frac{\partial}{\partial \sigma^2}\left(\frac{\sigma^2}{|A(f)|^2}\right) = \frac{1}{|A(f)|^2}
\end{aligned}$$

Hence the CRLB is computed as such:

$$\begin{aligned}
var(\hat{P}_X(f;\boldsymbol{\theta}))) &\geq \left[\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \alpha_1}, \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \sigma^2}\right] \begin{bmatrix} \frac{\sigma^2}{Nr_{xx}(0)} & 0 \\ 0 & \frac{2\sigma^4}{N} \end{bmatrix} \begin{bmatrix} \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \alpha_1} \\ \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \sigma^2} \end{bmatrix} \\
&\geq \frac{\sigma^2}{Nr_{xx}(0)}\left[\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \alpha_1}\right]^2 + \frac{2\sigma^4}{N}\left[\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \sigma^2}\right]^2 \\
&\geq \frac{\sigma^2}{Nr_{xx}(0)}\left[\frac{\sigma^2}{|A(f)|^4}\left(A(f)e^{j2\pi f} + A^*(f)e^{-j2\pi f}\right)\right]^2 + \frac{2\sigma^4}{N}\left[\frac{1}{|A(f)|^2}\right]^2 \\
&\geq \frac{\sigma^4}{N|A(f)|^4}\left[\frac{\sigma^2}{r_{xx}(0)|A(f)|^4}\left(A(f)e^{j2\pi f} + A^*(f)e^{-j2\pi f}\right)^2 + 2\right]
\end{aligned}$$

2.5 Real world signals: ECG from iAmp experiment

2.5.1 Heart rate probability density estimate (PDE)

The RRI signal $rr[n]$ was extracted from the ECG data for each trial using the provided function *ECG_to_RRI*. Using the segment for trial 1, the heart rate is obtained using the formula:

$$h[n] = \frac{60}{rr[n]}$$

And to obtain the averaged heart rates of every 10 samples by first using $\alpha = 1$ and then $\alpha = 0.6$ we use:

$$\hat{h}[1] = \frac{1}{10}\sum_{i=1}^{10} \alpha h[i],\ \ \hat{h}[2] = \frac{1}{10}\sum_{i=11}^{20} \alpha h[i],\ \ \ldots$$

The probability density estimate (PDE) of the original heart rates $h[n]$ and the averaged heart rates $\hat{h}[n]$ for $\alpha = 1$ and $\alpha = 0.6$ for trial 1 was plotted and displayed below (Figure 2.12).

The mean values of the heart rates are taken from the heart rate with the greatest probability (i.e. the maximum). By observation of the above plots, the original heart rate has approximately the same peak as the averaged heart rate with $\alpha = 1$, at about 85 beats/min and the averaged heart rate with $\alpha = 0.6$ has the peak at about 50 beats/min. The overall shapes of the original heart rate and the averaged heart rate with $\alpha = 1$ are quite similar, both with an approximately symmetric distribution where the heart rate averaged with $\alpha = 1$ having a slightly smaller standard deviation as an effect of averaging. On the other hand the overall shape of the heart rate averaged with $\alpha = 0.6$ has a higher peak and an even lower standard deviation, although the distribution is also approximately symmetrical.

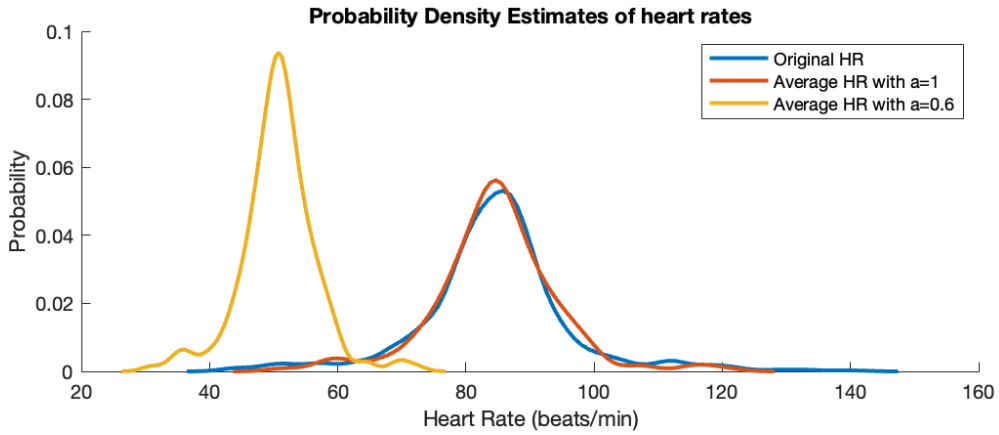


Figure 2.12: PDE for original and averaged heart rates for trial 1

This means that the value of α affects the shape of the probability density estimate in terms of its peak (the mean) and width (the standard deviation).

2.5.2 AR modelling of heart rate

The RRI data for the 3 trials were made to have zero mean (using MATLAB command *detrend*) and then used for producing their autocorrelation sequences. Then we use the shape of the autocorrelation sequence to infer whether the RRI data is an AR or an MA process.

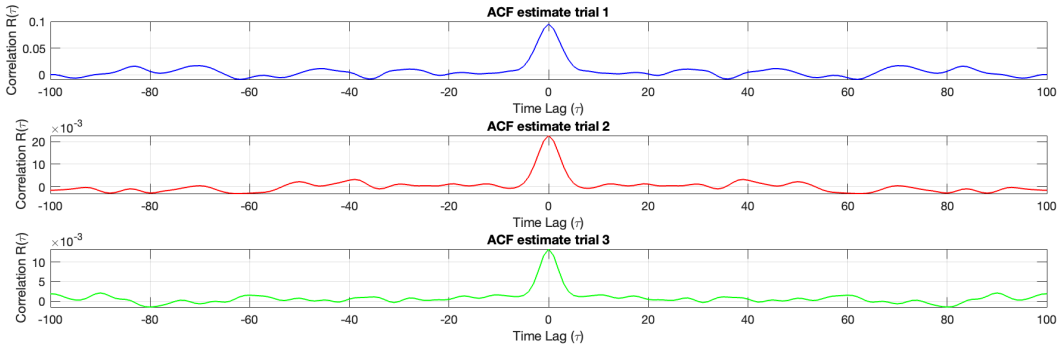


Figure 2.13: autocorrelation sequence for the RRI data for the three trials

The characteristics of the shape of the ACF plot for each process:

- **AR(p) -** Geometric decay, beyond model order
- **MA(q) -** Significant values until lag q

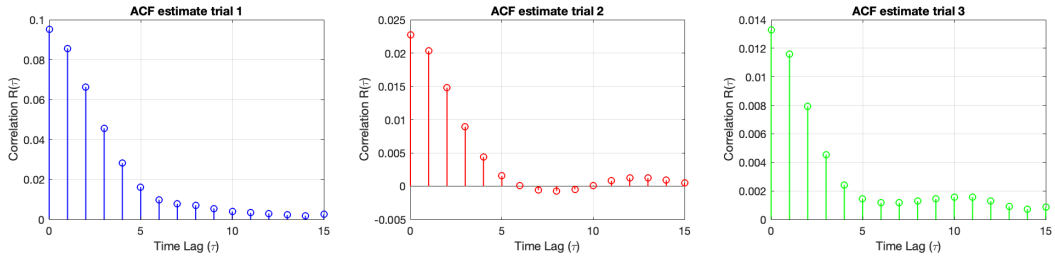
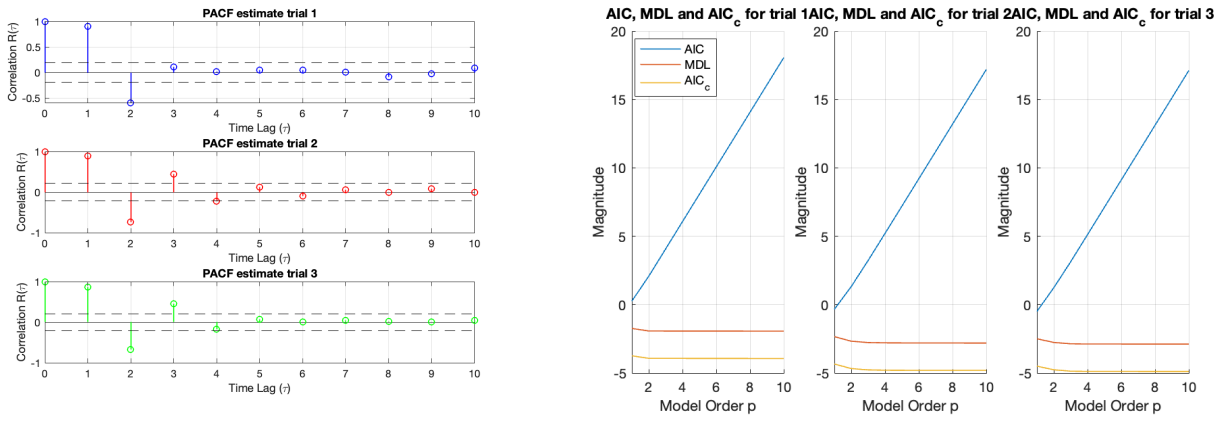


Figure 2.14: A closer look at the autocorrelation sequence

Looking at the figure above, the ACF for trial 1 is approximately an exponential decay, for trial 2 is approximately an exponential decay of a sinusoidal function and for trial 3 is closer to an exponential decay. However most of the values for ACF are not significant according to the 95% confidence interval bound. So based on the shape of the ACF, it is highly likely that the RRI data from all the trials are AR processes. Further investigations using the PACF and the information theoretic criteria (AIC, MDL and AICc) will be able to provide a confirmation of this observation.

(order)	**PACF**	**MDL**	**AIC**	**AICc**
Trial 1	2	2 (min=10)	1	2 (min=10)
Trial 2	3	2 (min=10)	1	2 (min=10)
Trial 3	3	2 (min=10)	1	2 (min=10)

Table 3: Predicted **optimal** order for AR models (visualised with Figure 2.15)



(a) PACF

(b) AIC, MDL and AICc

Figure 2.15: Plots of PACF and the information theoretic criteria (AIC, MDL and AICc)

For the PACFs, a bound of 95% confidence interval was applied and the significant values considered. In conclusion, trial 1 can be modelled as an AR(2), but it is between AR(2) or AR(3) for trails 2 and 3. If all trials have to be modelled with the same order, it would have to be AR(2).

3 Spectral estimation and modelling

The PSD of a stochastic process X_n is given by the absolute value of the Fourier transform of its autocorrelation function:

$$P_X(f) = \left| \sum_{\tau=-\infty}^{+\infty} R_X(\tau) e^{-j2\pi f\tau} \right|, \quad f \in [0,1]$$

where $R_X(.)$ is the ACF of the stochastic process X_n, and f is the normalised frequency. However, in real world applications, the method to estimate the PSD is based on the fast Fourier transform

(FFT), called the "periodogram", which is defined as:

$$P_X(f) = \frac{1}{N}\left|\sum_{n=0}^{N-1} x[n]e^{-j2\pi f\frac{n}{N}}\right|^2$$

This equation was written as an algorithm in a MATLAB script called *pgm* and tested with an N-sample realisation of WGN x, for N = 128, 256 and 512.
The result displayed in Figure 3.1 shows that the periodogram, with normalised frequency of 1 unit (which is 2π radians), is symmetric about the normalised frequency of 0.5, is periodic within the unit normalised frequency and has different magnitudes for different frequencies. Taking into consideration that the PSD is equivalent to the Fourier Transform of the ACF, and that the ACF of a WGN is $R_{WGN}(\tau) = \sigma^2\delta(\tau)$, the theoretical PSD of a unit-variance WGN is expected to to have a unit value for all frequencies (red plot).

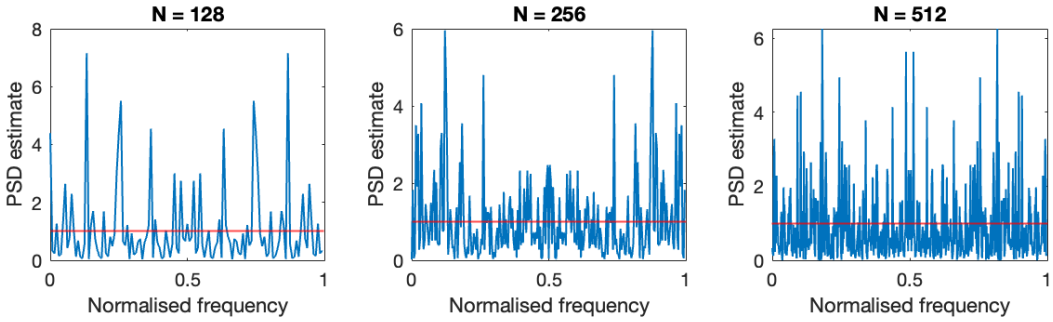


Figure 3.1: Result of test with an N-sample realisation of WGN x, for N = 128, 256 and 512.

However, this is not what was observed. This is mainly due to a finite length of samples generated WGN sample used. The PSD will approach a unit constant as the length of realisations approach infinity.

3.1 Averaged periodogram estimates

3.1.1 Zero-phase FIR filter

The PSD estimate computed can be smoothed by using a zero-phase FIR filter with impulse response given by $h[n] = 0.2\sum_{k=0}^{4}\delta[n-k]$. The smoothed PSD estimates for N=128, 256 and 512 are displayed in Figure 3.2. To gauge the improvement in the PSD estimate, the variance of the origonal and smoothed PSD estimates were calculated for each sample size N, and displayed in Figure 3.3.

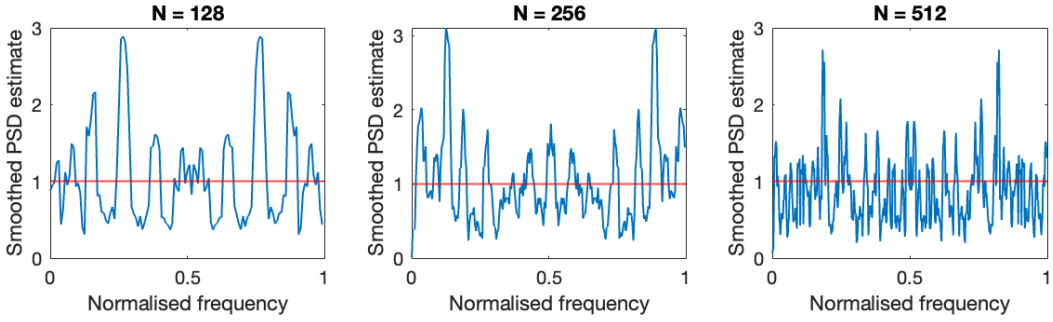


Figure 3.2: Smoothed PSD estimates using a zero-phase FIR, for N = 128, 256 and 512.

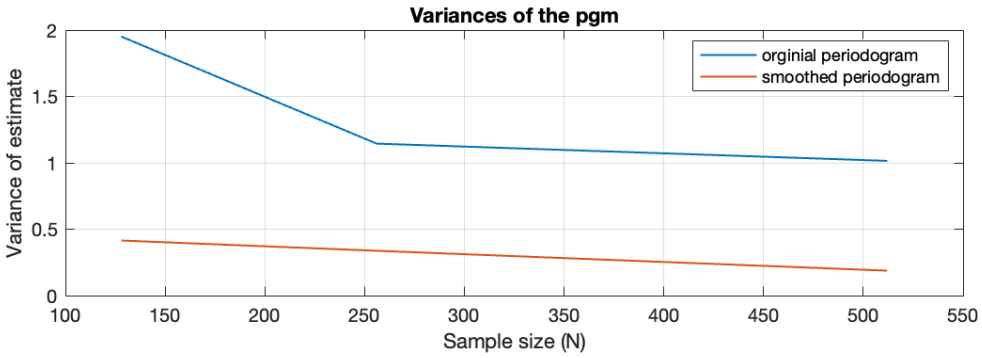


Figure 3.3: Variance comparison of original and smoothed PSD estimates

The smoothed PSD estimates still show symmetry about the normalised frequency of 0.5, is periodic within the unit normalised frequency and has different magnitudes for different frequencies. However, there is a visible reduction in the variability of the magnitudes and Figure 3.3 provides a closer look at the variances of such values.

There seemed to be a decrease in variance by approximately a factor of 5, which corresponds to the size of the filter's window length. This effect is demonstrated with a derivation using the smoothed periodogram $\tilde{P}_X(f) = 0.2\sum_{k=0}^{4}\hat{P}_X(f-k)$. Note that the periodogram estimates are independent.

$$Var(\tilde{P}_X(f)) = 0.04 \times Var\left(\sum_{k=0}^{4}\hat{P}_X(f-k)\right) = 0.04\sum_{k=0}^{4}Var(\hat{P}_X(f-k)) = 0.2 \times Var(\hat{P}_X(f))$$

The smoothing also does not introduce bias in the new estimator. This is evident when we derive the expected value of the smoothed PSD estimate:

$$E\left\{\tilde{P}_X(f)\right\} = 0.2 \times E\left\{\sum_{k=0}^{4}\hat{P}_X(f-k)\right\} = 0.2\sum_{k=0}^{4}E\left\{\hat{P}_X(f-k)\right\} = \hat{P}_X(f)$$

In conclusion, in this case where the input is a WGN, smoothing improves the PSD estimate.

3.1.2 Subdivide it into eight non-overlapping 128-sample segments

A 1024-sample sequence of WGN was generated and then subdivided into eight non-overlapping 128-sample segments. The PSD estimate for each segment was computed and displayed in Figure 3.4. The variance for the PSD estimate of each segment is displayed in Table 4.

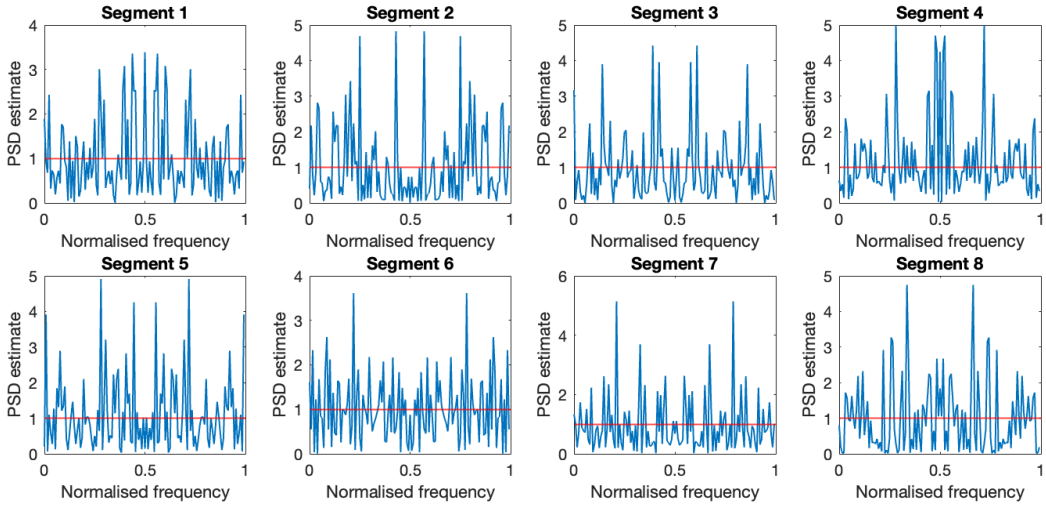


Figure 3.4: Plot for all 8 segments, red line is expected PSD

Segment	1	2	3	4	5	6	7	8	Average
Variance	1.2765	0.79305	0.45151	1.115	0.83907	0.82122	1.5395	0.64014	0.9345

Table 4: Sample Variance for each 128-sample segment

The variances for each segment can be considered approximately similar. This was explained in the previous section, where the variance of the PSD estimate is the same for the samples before and after segmenting.

3.1.3 Averaged Periodogram

The eight results were avraged to yield a new PSD estimator called the averaged periodogram, displayed in Figure 3.5.

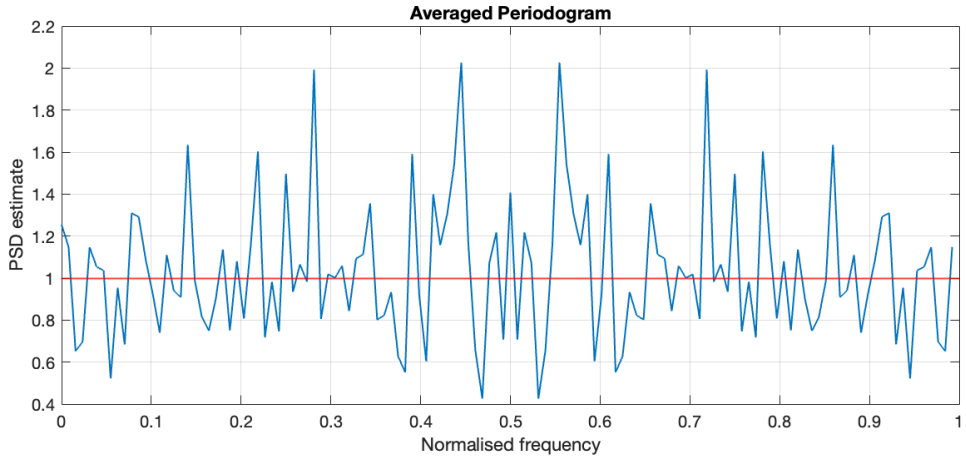


Figure 3.5: Periodogram averaged from the eight segments, with variance=0.10345

The variance of this averaged periodogram was calculated to be 0.10345. This is, as expected, much lower than the average variance for the 128-sample segments, 0.9345. To derive this effect we first present the mathematical description of the averaged periodogram, where **K** is the number of segments:

$$\bar{P}_X(f) = \frac{1}{K}\sum_{k=1}^{K}\hat{P}_k(f) \quad \textit{where } \hat{P}_k(f) \textit{ is a segment from } \hat{P}_X(f)$$

The variance of the averaged periodogram is given by:

$$Var(\bar{P}_X(f)) = \frac{1}{K^2}\sum_{k=1}^{K}Var(\hat{P}_k(f)) = \frac{1}{K}Var(\hat{P}_X(f))$$

The order of the decrease in variance is equal to the number of segments **K**.It can also be concluded that when number of segments approaches infinity, variance will approach zero (as $K \to \infty$, $Var(\bar{P}_X(f)) \to 0$). So there is a trade off between bias (due to small sample sizes) and the variance of the averaged periodogram when selecting for the optimal number of segments.

3.2 Spectrum of autoregressive processes

A 1064-sample WGN sequence x was filtered using a filter with the coefficients b = 1 and a = [1, 0.9], where y is filter's output sequence. This is an AR(1) filter.

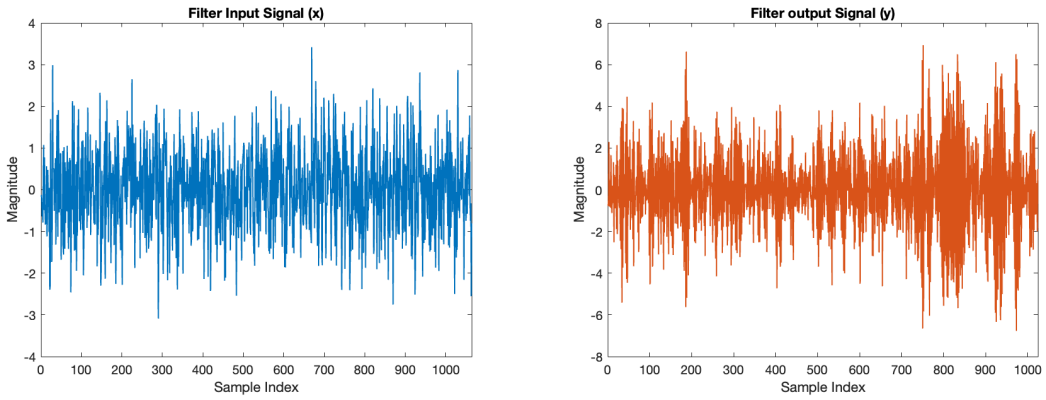


Figure 3.6: Plot of sequences x and y in the time domain

3.2.1 Exact PSD of the filtered signal

The exact PSD generated by the MATLAB code $plot(w/(2*pi), abs(h).^2)$ was plotted in Figure 3.7 **in blue**, given by the mathematical representation:

$$P_Y(f) = \frac{1}{|1 + 0.9e^{-j2\pi f}|^2}$$

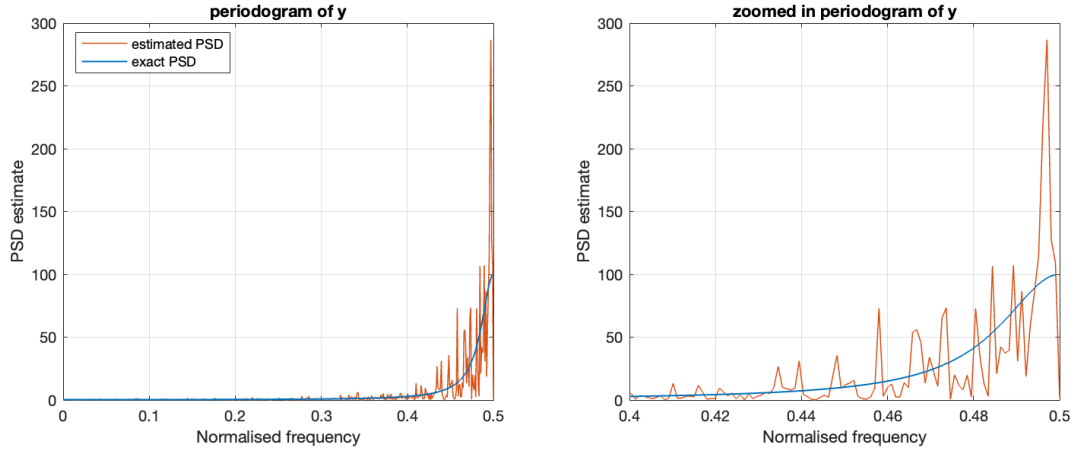


Figure 3.7: Plot of the exact and estimated PSD (normal and zoomed in)

The exact PSD of the AR(1) model shows that the PSD magnitude is close to zero for low frequencies and very large for high frequencies, indicating a high-pass filter. We can show this with the filter's transfer function:

$$H(z) = \frac{1}{1 + 0.9e^{-j2\pi f}} \qquad where\ f \to \infty, H(z) \to 1\ and\ f \to 0, H(z) \to \infty$$

The cut-off frequency is defined here as the frequency at which the amplitude falls -3dB below the passband gain. So we can calculate the cutoff-frequency for the gain of $|H(z)| = \frac{1}{\sqrt{2}}$ (corresponding to 3dB) as follows.

$$\frac{1}{\sqrt{2}} = \frac{1}{\sqrt{1.81 + 1.8\cos(2\pi f)}} \quad \Rightarrow \quad f = \frac{1}{2\pi}\cos^{-1}\left(\frac{2 - 1.81}{1.8}\right) = 0.2332$$

Since sampling frequency is 1000Hz by default in MATLAB, cutoff frequency in Hz is given by 233.2 Hz.

3.2.2 Periodogram of y

The periodogram for sequence y is calculated and then plotted in Figure 3.7 **in red**. Comparing both plots to each other, the PSD estimate has very low amplitude variations around the theoretical value for low normalised frequencies and this variance is increasing with increasing frequency values. The observation at the high frequencies is mainly due to the finite number of samples used. The lower the sample size the higher the inaccuracies, hence the bias of the estimate increases. As sample sizes approach infinity, the estimated PSD will converge to the theoretical value. AT low frequencies, there are little to no deviations from the theoretical values because the lower frequencies had been filtered out by the high-pass filter.

3.2.3 Zoom in

The plot of PSD values was zoomed for the interval of normalised frequencies between 0.4 and 0.5, shown in Figure 3.7. The results differ from each other due to the underlying rectangular windowing within the periodogram estimator. Let's say that our sample signal is the output of an infinite sample signal through a rectangular window ($w_R[n]$) of length N (sample size).

$$y_R[n] = y[n]w_R[n] \qquad where\ w_R[n] = \begin{cases} 1 & ,for\ 0 \leq n \leq N-1 \\ 0 & ,otherwise \end{cases}$$

So the expected value of the priodogram is given below, where the DFT of a rectangular window is similar to a sinc function.

$$E\left\{\hat{P}_Y(f)\right\} = \frac{1}{2\pi N}\hat{P}_Y(f) * |W_R(f)|^2 = \frac{1}{2\pi N}\hat{P}_Y(f) * \left(\frac{\sin(nfN)}{\sin(nf)}\right)^2$$

So as sample size N approaches very large values, the DFT of a rectangular window approaches a Dirac delta function. Which means that, with a finite sample size N, the results of the estimate and theoretical PSDs will be different. Also, with finite sample size, we can see that there is is a bias in estimated PSD, which was observed in Figure 3.7.

3.2.4 Model-based PSD for y

The *xcorr* function was used to estimate $\hat{a}_1 = -\hat{R}_Y(1)/\hat{R}_Y(0)$ and $\hat{\sigma}_X^2 = \hat{R}_Y(0) + \hat{a}_1\hat{R}_Y(1)$ for the sequence y. Then the model-based PSD was computed according to the formula

$$\hat{P}_Y(f) = \frac{\hat{\sigma}_X^2}{|1 + \hat{a}_1 e - j2\pi f|^2}$$

where $\hat{P}_Y(f)$ is again generated with $freqz$. The results are plotted in Figure 3.8.

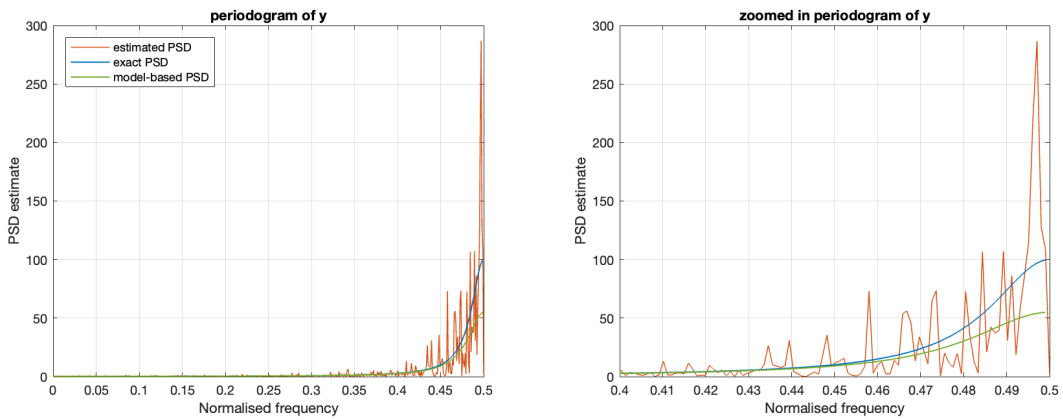


Figure 3.8: Plot of the exact, estimated and model-based PSD (normal and zoomed in)

We observe that the Model-based PSD estimate is a better estimate for signal's exact PSD compared to the periodogram, due to absence of spikes and closeness in value to the exact PSD for all frequencies. The model-based PSD estimates are less biased and possess a lower variance than periodogram. Deviations of model-based PSD from exact PSD is due to variance of the estimator. However, the downside is that this is only possible is the assumed model order is correct.

3.2.5 Model-based PSD for sunspot time series

The above procedure was repeated for the sunspot time series and its periodogram is compared with a model-based PSD. As shown in Figure 3.9, the analysis was done for a range of model orders, specifically AR(1), AR(2), AR(5), AR(10) and AR(50). As a note, we have concluded that model order 2 is the optimal order for this data in the earlier sections.

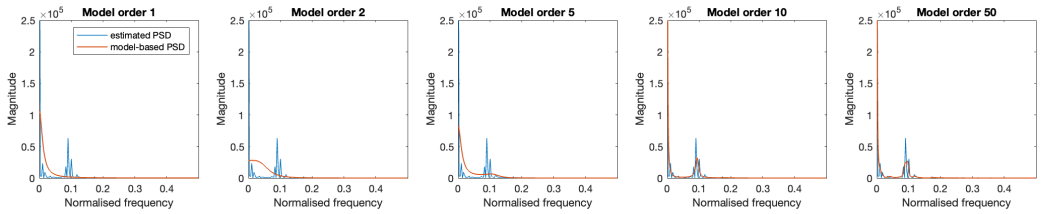


Figure 3.9: Plot of the estimated and model-based PSD for original sunspot time series

The AR(2) model-based estimate indicates that the for original series, theoretical PSD would have a maximum value at zero frequency and then decay to zero magnitude as frequency increases. The lack of peak identification is due to the DC offset in the original dataset. So a zero-mean version of this dataset was used instead and the results are displayed in Figure 3.10.

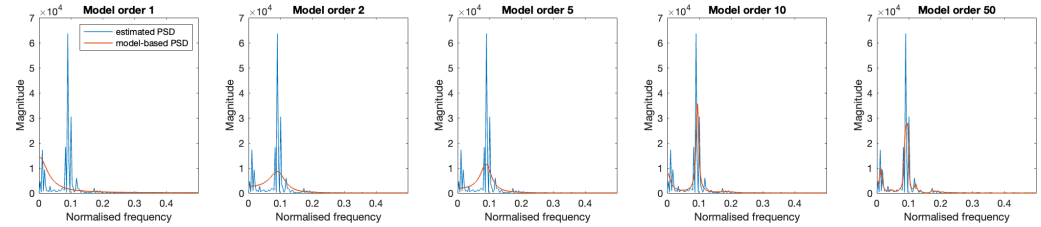


Figure 3.10: Plot of the estimated and model-based PSD for zero-meaned sunspot time series

Now we can see that, from the AR(2) model, the theoretical PSD would have a spectral peak at the normalised frequency of approximately 0.1, and a non-zero magnitude at 0 frequency. Under-modelling (with AR(1) model) have inadequate degrees of freedom, resulting in high variance and bias as well as the failure to recognise a peak at 0.1 normalised frequency. Over-modelling (with models orders greater than 2) gives estimates that are closer to the periodogram estimates as model order increases, making it unreliable. The spectral peak was captured, but its magnitude is higher than that in the AR(2) model. Although over-modelling provides too much drastic details in a zero-meaned dataset, it may perhaps be used for peak identification in datasets with an offset, as AR(2) in Figure 3.9 does not provide information on the peak.

3.3 The Least Squares Estimation (LSE) of AR Coefficients

3.3.1 Cost function matrix form

Then, the LS cost function for finding the unknown AR coefficients $a = [a_1, a_2, ..., a_p]^T$ is given by

$$J = \sum_{k=1}^{M} \left[\hat{r}_{xx}[k] - \sum_{i=1}^{p} a_i \hat{r}_{xx}[k-i] \right]^2 \qquad , for\ M \geq p$$

where M is data length and p is the model order. For ease of derivation, we will use $x[k] = \hat{r}_{xx}[k]$. So the LS cost function is rewritten as

$$J = \sum_{k=1}^{M} \left[x[k] - \sum_{i=1}^{p} a_i x[k-i] \right]^2 \qquad , \textit{for } M \geq p$$

We first define matrix **B** as

$$\boldsymbol{B} = \begin{bmatrix} x[1] - \sum_{i=1}^{p} a_i x[1-i] \\ \vdots \\ x[M] - \sum_{i=1}^{p} a_i x[M-i] \end{bmatrix} = \begin{bmatrix} x[1] \\ \vdots \\ x[M] \end{bmatrix} - \begin{bmatrix} \sum_{i=1}^{p} a_i x[1-i] \\ \vdots \\ \sum_{i=1}^{p} a_i x[M-i] \end{bmatrix} = \boldsymbol{x} - \boldsymbol{s}$$

Hence the cost function J can be expressed as $J = \boldsymbol{B}^T\boldsymbol{B}$ and x is the matrix representing $\sum_{k=1}^{M} \hat{r}_{xx}[k]$. Then we further break down **s** into the matrix **H** and **a**.

$$\boldsymbol{s} = \begin{bmatrix} x[1-1] & x[1-2] & \dots & x[1-p] \\ x[2-1] & x[2-2] & \dots & x[2-p] \\ \vdots & \vdots & \ddots & \vdots \\ x[M-1] & x[M-2] & \dots & x[M-p] \end{bmatrix} \boldsymbol{.a} = \begin{bmatrix} x[0] & 0 & \dots & 0 \\ x[1] & x[0] & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x[M-1] & x[M-2] & \dots & x[M-p] \end{bmatrix} \boldsymbol{.a} = \boldsymbol{H.a}$$

Where H is commonly called the observation matrix.
Using the above relationships $\boldsymbol{B} = \boldsymbol{x} - \boldsymbol{s}$ and $\boldsymbol{s} = \boldsymbol{H.a}$, we can derive the following relationship.

$$J = \boldsymbol{B}^T\boldsymbol{B} = (\boldsymbol{x} - \boldsymbol{H.a})^T(\boldsymbol{x} - \boldsymbol{H.a})$$

The Least Squares estimates for the unknown coefficients a is the derivative of the cost function with respect to the coefficients which equals to zero.

$$\begin{aligned} J &= (\boldsymbol{x} - \boldsymbol{Ha})^T(\boldsymbol{x} - \boldsymbol{Ha}) = \boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{Ha} - \boldsymbol{a}^T\boldsymbol{H}^T\boldsymbol{x} + \boldsymbol{a}^T\boldsymbol{H}^T\boldsymbol{Ha} \\ &= \boldsymbol{x}^T\boldsymbol{x} - 2\boldsymbol{x}^T\boldsymbol{Ha} + \boldsymbol{a}^T\boldsymbol{H}^T\boldsymbol{Ha} \\ \frac{\partial J}{\partial \boldsymbol{a}} &= -2\boldsymbol{H}^T\boldsymbol{x} + 2\boldsymbol{H}^T\boldsymbol{Ha} = -2\boldsymbol{H}^T(\boldsymbol{x} - \boldsymbol{Ha}) \end{aligned}$$

With zero gradient, we rearrange for **a** to yield the LSE.

$$\hat{\boldsymbol{a}} = (\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{x}$$

We can represent the derivative as shown and a zero gradient gives us a relationship of the LSE for unknown AR coefficients with the corresponding Yule-Walker estimates.

$$\frac{\partial J}{\partial \boldsymbol{a}} = -2\sum_{k=1}^{M} \left(x[k] - \sum_{i=1}^{p} a_i x[k-i] \right) \left(\sum_{i=1}^{p} x[k-i] \right) = 0$$

Hence,

$$x[k] = \sum_{i=1}^{p} a_i x[k-i] \qquad \textit{equivalent to } \hat{r}_{xx}[k] = \sum_{i=1}^{p} a_i \hat{r}_{xx}[k-i]$$

In the case of Yule-Walker method, AR coefficients are estimated according to the equations:

$$\hat{r}_{xx}[k] = \begin{cases} \sum_{i=1}^{p} a_i \hat{r}_{xx}[k-i] & , for k \geq 1 \\ \sum_{i=1}^{p} a_i \hat{r}_{xx}[k-i] + \sigma_w^2 & , for k = 0 \end{cases}$$

We can see that the LSE method uses the Yule-Walker equations with restriction of $k \geq 1$, and it is sometimes called the Least-square modified Yule-Walker equations

3.3.2 Close examination of the observation matrix H

Generally, the observation matrix **H** is considered to be deterministic but, in this case, it can be described as stochastic. This is because the elements in **H** are ACF estimates, which are considered to be stochastic due to its dependence of x[n], a smaple of WGN. The sample size is finite, so it should be noted that the values of the ACF estimates are not reliable at large time lags.

3.3.3 The LSE approach for the real world sunspot time series

The LSE method was used to calculate the AR coefficients for AR(p) models of order p=1,...,10. These values are compared to the coefficients obtained with the Yule-Walker method as shown in Figure 3.11.

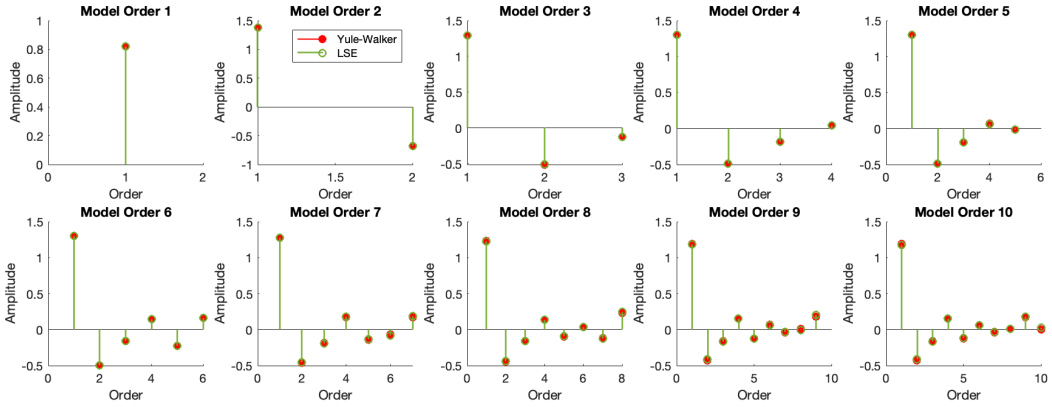


Figure 3.11: AR coefficients estimated using LSE method and Yule-Walker

3.3.4 Approximation error of the AR models

The AR models are compared by plotting their approximation error to the original data. Here, MSE is used as the prediction error.

In Figure 3.12, it shows very similar plots for the approximation error of both methods, where there is a steep decrease up to order 2 and then a slow decrease as the order increases. As a confirmation, the AICc criterion, which is suitable for a small sample size, indicates that the optimal model order is 2. So we can conclude that the optimal model order is 2 and this agrees with the result for the analysis conducted in Section 2.3.3.

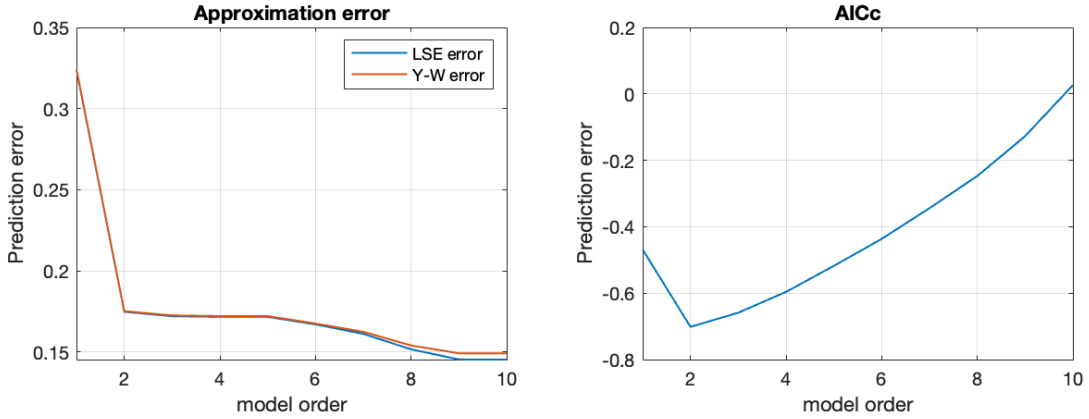


Figure 3.12: Plot of Approximation error for LSE and Yule-Walker method, and the AICc criterion

3.3.5 Power spectra associated with the AR(p) models

The power spectra associated with the AR(p) models of the sunspot time series are plotted in Figure 3.13, along with the PSD estimate with the periodogram for comparison.

The PSD estimates obtained are very similar to the ones obtained using the Yule-Walker method in section 3.2. We can also conclude that over-modelling increases the magnitude of the peak at normalised frequency of 0.1, and that the PSD estimates produces unreliable results by following the shape of periodogram more closely.

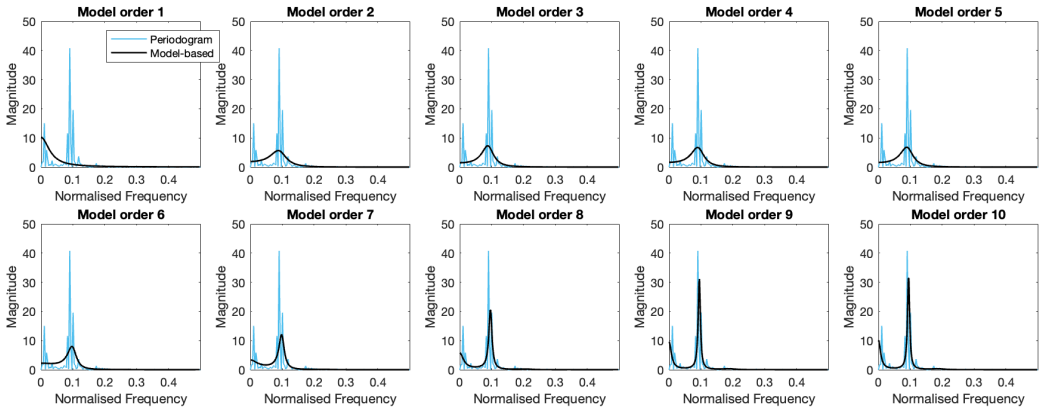


Figure 3.13: Model-based PSD of AR(p) for different model orders

3.3.6 Behaviour of MSE versus N

Using identified optimal model order 2, the approximation error was computed for data lengths $N \in [10 : 5 : 250]$ and displayed in Figure 3.14.
It can be observed that the MSE error reaches a relatively stable value of about 0.15, after data length of approximately 125. This means that N=125 is the optimal data length, with the minimum computation for the best results.

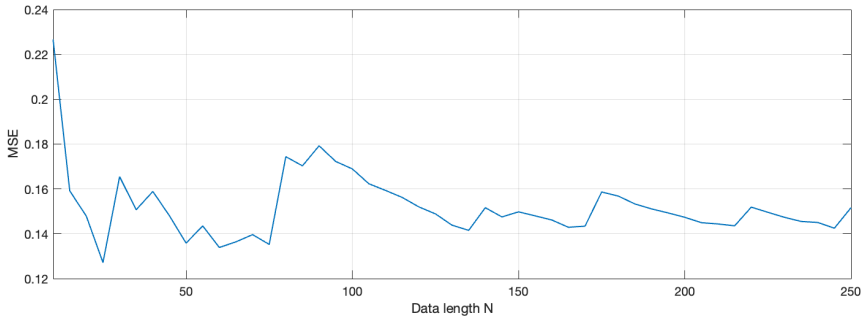


Figure 3.14: MSE versus Data length N for AR(2) model of sunspot time series

3.4 Spectrogram for time-frequency analysis: dial tone pad

3.4.1 Generate a random London landline number

A random London landline number was generated, i.e. a sequence of the form: 020 XXXX XXXX, where the last eight digits are drawn from the set $0, 1, 2, 3, 4, 5, 6, 7, 8, 9$ (with uniform probability). The Dual Tone Multi-frequency (DTMF) system specifies that a signal composed of two frequencies is assigned to each button of the keypad, so that the output y[n] is represented as such.

$$y[n] = \sin(2\pi f_1 n) + \sin(2\pi f_2 n)$$

where f_1 and f_2 = are frequencies in Hz, and n is the time index. Figure 3.15 provides the plots of signal y for keys 0, 2 and the idle sequence
According to the Dial Pad Frequencies provided, the maximum frequency we can get is 1477 Hz. According to Nyquist's Theorem, the sampling frequency mush be at least twice the frequency of

interest ($f_s \geq 2f_N$) to avoid aliasing, and this minimum frequency is coined the 'Nyquist Rate'. So the Nyquist Rate for sampling the dial pad frequency is 2954 Hz and the proposed sampling rate is much greater than this value. A much higher frequency also produces a smoother FFT due to higher resolutions. On a sidenote, FFT is optimized for power-of-2 sizes because they are more efficient in terms of memory access and arithmetic operations. So having the sampling frequency be a power of two ($2954 = 2^{15}$) is a plus.

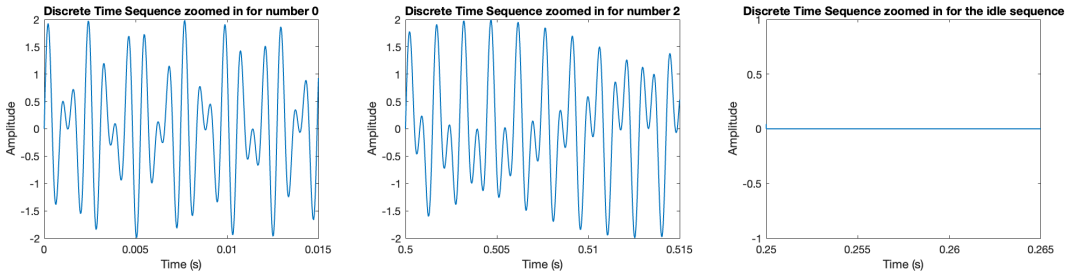


Figure 3.15: Plot of signal y for keys 0, 2 and the idle sequence

3.4.2 Analyse the spectral components of the sequence y

The spectral components of the sequence y were analysed using the MATLAB function *spectrogram* and the FFT was computed using non-overlapping segments and a Hanning window. The results are displayed in Figure 3.16.
The spectrogram is a plot of the signal frequency against time domain, with colours indicating the amplitude or power of each frequency at a certain time. In Figure 3.16, the maximum powers for the different frequencies are very clearly displayed in yellow and they correspond to the pairs of Dial Pad frequencies. So we can deduce from it the combinations of frequency as well as the time when they happen. In the same figure, the FFT for each segment is shown to contain two distinguishable peaks. However, the information can only be used to deduce the numbers, not the order they come in.

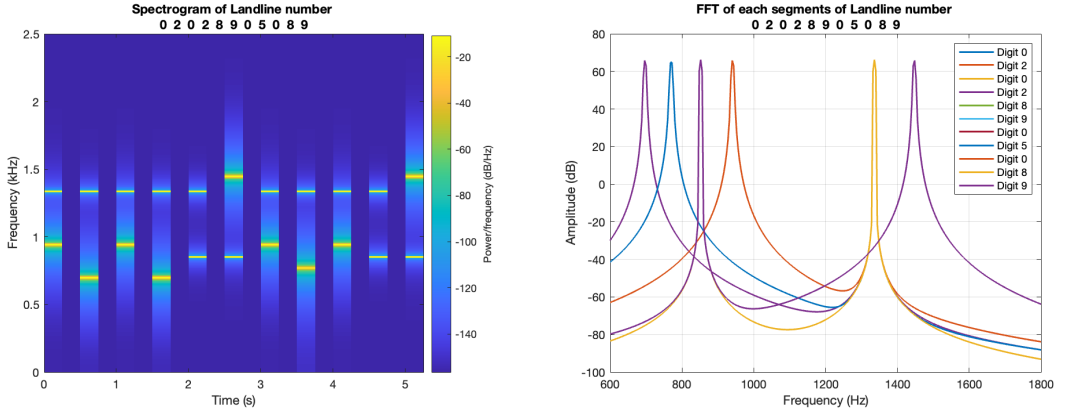


Figure 3.16: Spectrogram of sequence y and FFT of sequence segments

3.4.3 Identify the sequence generated by a key press

We can identify the sequence using the spectrogram by looking for the spectrum peaks for each time interval of 0.25s and ignoring the idle time. This can be done using the MATLAB command $[pks, locs] = findpeaks(data)$ to locate the normalised frequencies of the 2 maximum values. *locs* gives us the location of the maximum values, which are the normalised frequencies. Then the normalised frequencies are multiplied by the sampling frequency to convert them to Hz. Finally, matching the pairs of frequencies to the Dial Pad frequencies will produce the digit of the key pressed.

3.4.4 Signal corrupted by channel noise

White Gaussian noise was added to sequence y in three cases: Low-variance noise ($\sigma^2 = 0.001$), Medium-variance noise ($\sigma^2 = 0.1$) and High-variance noise ($\sigma^2 = 100$).

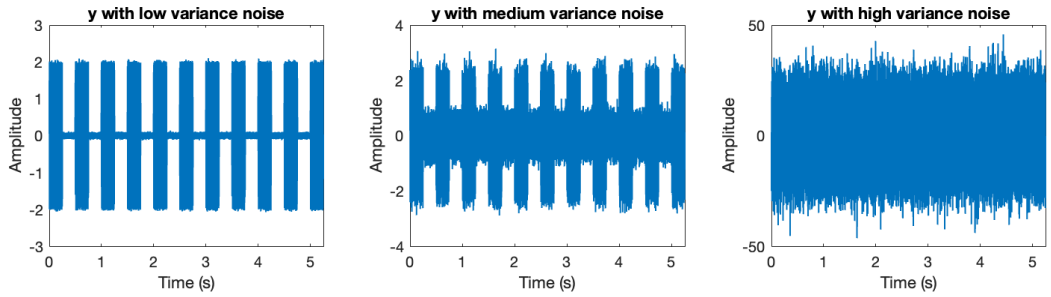


Figure 3.17: Plot of signal corrupted by the 3 levels of channel noise

As the noise variance increase, the desired signal becomes more immersed until it becomes completely immersed as shown in Figure 3.17. When it is completely immersed, it is expected that the frequencies for the key tones to not be identifiable, as demonstrated in Figure 3.18 and 3.19. However, the spectral peaks are still visible in the spectrogram and FFT for the cases of low and medium noise variances. The same method as mentioned above would be used to classify key presses for these cases, but it is impossible to extract any desired information from the case with a high noise variance.

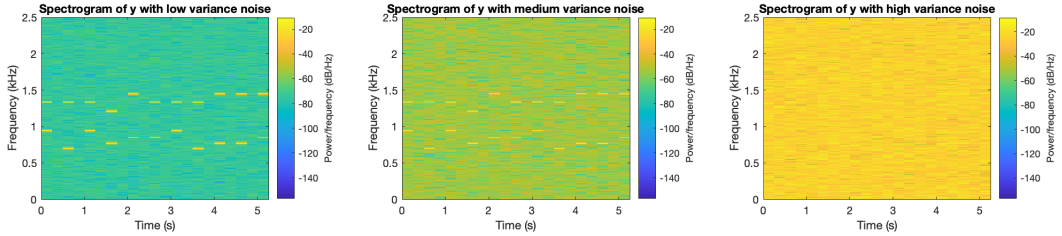


Figure 3.18: Spectrogram plots for 3 levels of noise variances

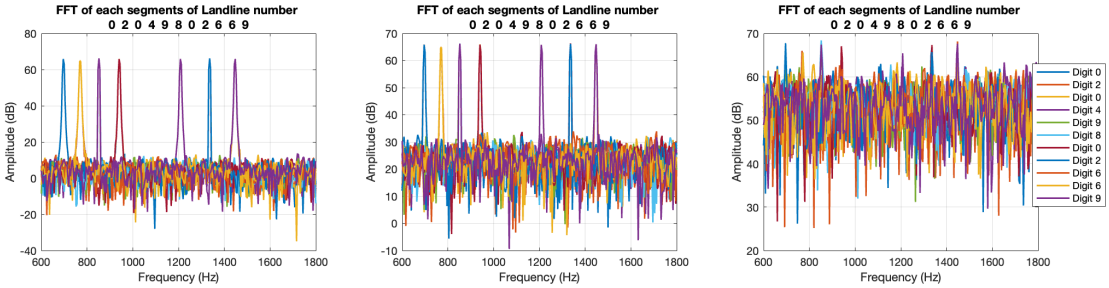


Figure 3.19: FFT of segments for 3 levels of noise variances

3.5 Real world signals: Respiratory sinus arrhythmia from RR-Intervals

The standard and averaged periodogram of window lengths 50 and 150 were used to obtain the PSD of the RRI data.

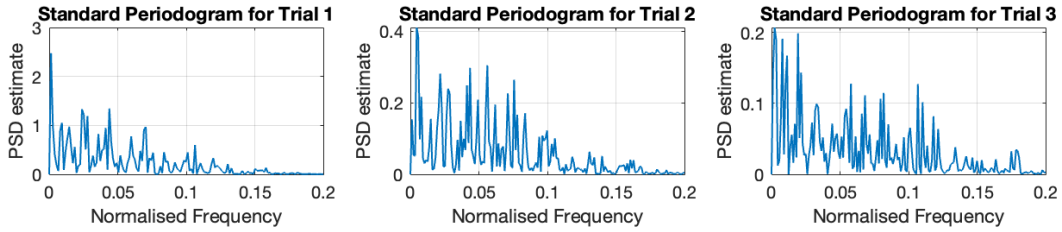


Figure 3.20: Standard Periodogram for the three trials

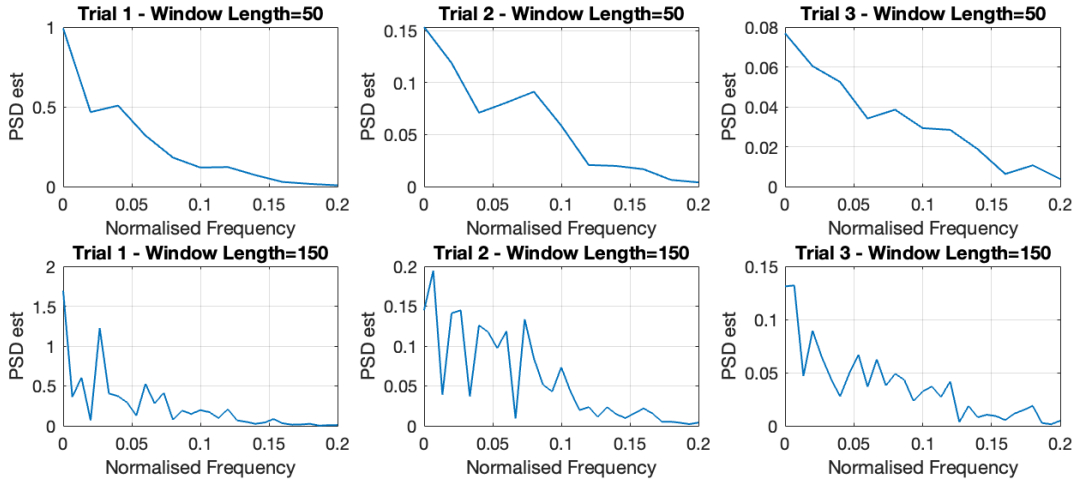


Figure 3.21: Averaged Periodogram for the three trials using window lengths 50 and 150

The PSD estimates of RRI data from the three trials contain multiple spectral peaks, most likely due to the poor quality of the recordings. Theoretically, there should be two spectral peaks, one at a lower frequency for the slow down of heart rate during expiration and the other at a higher frequency for the increase in heart rate due to inspiration. These are the effects of Respiratory Sinus Arrhythmia (RSA). However these peaks are not clear on both the standard and averaged periodogram, and the increased window size does not help either, possibly due to the faulty sensors or environmental interference.

4 Optimal filtering - fixed and adaptive

4.1 Wiener filter

A 1000-sample WGN sequence x was generated and present as an input to the unknown system described by a filter with coefficients b = [1, 2, 3, 2, 1], a = [1]. The unknown system output $y[n]$ was normalised to unit variance and a WGN $\eta[n]$ of 0.1 standard deviation was added to it to generate $z[n]$. The expected Signal-to- Noise Ratio (SNR) in dB was calculated to be:

$$SNR = 10log_{10}\left(\frac{P_{signal}}{P_{noise}}\right) = 10log_{10}\left(\frac{\sigma_y^2}{\sigma_\eta^2}\right) = 10log_{10}\left(\frac{1^2}{0.1^2}\right) = 20dB$$

Using the MATLAB function *snr*, this value was calculated to be 19.8627dB.

4.1.1 Find the optimal coefficients of the Wiener filter

The statistics $\boldsymbol{R_{xx}}$ and $\boldsymbol{p_{zx}}$ were calculated and the optimal coefficients of the Wiener filter are provided in Table 5. These estimates are shown to be very close to the theoretical values.

b values	1	2	3	2	1
$\boldsymbol{w_{opt}}$	0.97268	1.986	2.9785	1.9918	0.97978

Table 5: Wiener filter optimal coefficients

4.1.2 The effects of different noise powers and filter length

The experiment was repeated by varying the variance of the additive noise in the region $\sigma^2 \in [0.1, 10]$. Then the SNR and the optimal coefficients of the Wiener filter were calculated for each variance of noise, and displayed in Table 6

σ_η^2	**calculated SNR**	$\boldsymbol{w_{opt}}$				
0.1	9.6179	1.0128	1.9835	2.9957	2.0245	1.0032
0.5	2.9838	0.9225	1.9728	2.9484	1.9598	1.0397
1	-0.19931	0.81884	2.0652	3.0387	2.0998	1.3736
3	-4.6439	1.4027	1.7975	3.0545	1.7943	1.2424
6	-7.6243	1.0785	2.2394	3.1137	1.6487	1.2453
10	-10.0142	1.4169	1.6478	1.9864	1.7624	1.549

Table 6: SNR and the Wiener solution for different noise powers

The SNR for $z[n]$ decreases down to the negative values as the noise variance increases, as expected because the magnitude of the noise is greater than the signal. Also, the higher the variance of the noise, the less accurate the optimum Wiener filter solution (comparing it to the filter coefficients b). As noise variance increase, $z[n]$ will converge into $\eta[n]$, so any correlation introduced by filter is lost and it becomes harder to identify the system.
The effects on the Wiener solution for $N_w > 4$ were investigated by calculating the optimal Wiener filter solution for $N_w = 5, 6, 7$, as presented in Figure 7

N_w	$\boldsymbol{w_{opt}}$							
5	0.97286	1.9861	2.9789	1.9917	0.97996	0.0061837		
6	0.9734	1.9854	2.9783	1.99	0.98058	0.0054043	-0.02666	
7	0.97268	1.9859	2.9776	1.9894	0.97898	0.0059875	-0.027408	-0.026083

Table 7: Wiener solution for $N_w > 4$

By observation, only the first 5 coefficients have significant values that are very close to theoretical values, and all the following coefficients have values very close to zero. These deviations from zero are due to the effects of the additive noise $\eta[n]$ on the calculation of the statistic p_{zx}. So order 4 is concluded to be the correct filter order.

4.1.3 Estimate computational complexity

The number of multiplications and additions in the computation of the Wiener solution, with $N_w + 1$ number of coefficients, was estimated by considering the computation of the statistics $\boldsymbol{R_{xx}}$ and $\boldsymbol{p_{zx}}$. Taking N as the sample size and assuming that $N \gg N_w$, both the CCF and ACF estimates have $(N+1)$ multiplications followed by N additions. which are considered for $(2N_w+1)$ time-lags in ACF and (N_w+1) time-lags for CCF. The inversion of matrix $\boldsymbol{R_{xx}}$, with dimensions $(N_w+1) \times (N_w+1)$, has the number of operations of approximately $(N_w+1)^3$. The matrix multiplication of $\boldsymbol{R_{xx}^{-1}}$ and $\boldsymbol{p_{zx}}$ has (N_w+1) multiplications and N_w additions for each $\boldsymbol{w_{opt}}$ (with (N_w+1) elements). Hence the full calculation becomes:

$$\textit{Total Operations} \approx N(N+1)(3N_w+2) + (2N_w+1)(N_w+1)^2$$

So the number of operations is in the order of magnitude $\mathcal{O}(N^3)$.

4.2 The least mean square (LMS) algorithm

4.2.1 Apply the LMS algorithm

A MATLAB routine called *lms* was written and applied to the **x** and **z** signals used in Section 4.1. The results are displayed in Figure 4.1 using the learning rate $\mu = 0.004$.

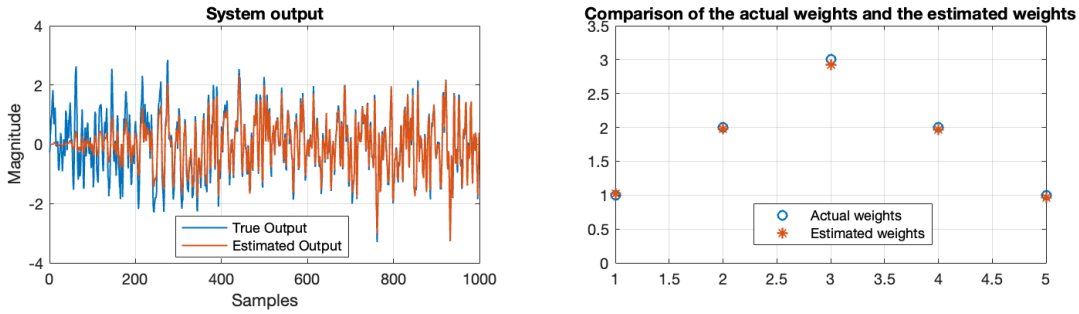


Figure 4.1: Plot of $\hat{y}$ and $\boldsymbol{z}$, and the comparison between the weight vectors

The selection of μ is done by ensuring that the mean and MSE converges to guarantee stability for the mean of LMS and the generation of filter coefficients. For the mean, the condition for stability is $0 < \mu < \frac{2}{\lambda_{max}} = 1.0412$, where λ_{max} is the largest eigenvector of $\boldsymbol{R}_{xx}$, and for the MSE it is $0 < \mu < \frac{2}{tr[\boldsymbol{R}_{xx}]} = \frac{2}{p\sigma_x^2} = 0.4001$, where p is the filter order. Hence μ was chosen to be 0.004.
The system output shows that there is substantial deviations in the initial iterations and the error decreases until the filter weights converge to their true values, and hence system output converges to the true output. This is due to the learning delay of the LMS algorithm, hence the finite rate of convergence.

4.2.2 The effect of increasing and decreasing μ

The adaptation gain μ was chosen to be 0.01. The plots of the time evolution of the coefficients and squared estimate error are provided in Figure 4.2.

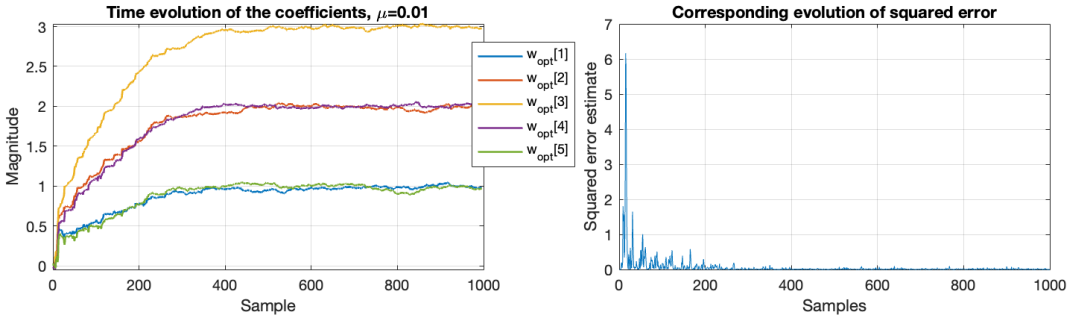


Figure 4.2: Plot of the coefficients time evolution and the squared estimate error

The algorithm utilises approximations in a recursive fashion, so the optimal filter coefficients (or weights) will never reach the exact true values, but they converge to an approximate value which can be averaged. Hence the convergence has to display a small variance in order to provide an accurate mean, and the corresponding squared error has to converge to zero. The figure above displays such convergence where the squared error decays exponentially with time. Because μ is very much less than the critical values, there is convergence in both the mean and MSE.

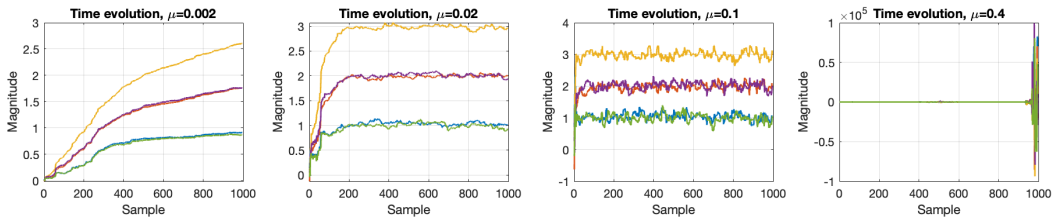


Figure 4.3: The effect of increasing μ on the coefficients

The effects of varying adaptation gain μ was investigated by plotting the evolution of the five coefficient estimates with varying adaptation gain in the interval [0.002,0.5], as shown in Figure 4.3.

We observe that as μ increases, the rate of convergence increases, but the evolution of the coefficients have a larger variance upon convergence (i.e. greater MSE). This happens because of the larger time steps taken in the algorithm, hence small adjustments are not possible. So there is a trade off between the rate of convergence and the error (or variance) of the converged values. Also, one thing to note is that with $\mu = 0.4$, there is no convergence in MSE even though it is within the bounds of the critical value because the sample ACF is used instead of the true ACF.
The time constant τ for the convergence indicates when the value has fallen by *1/e* of its initial value, is given by:

$$\tau_j = -\frac{1}{ln(1-\mu\lambda_j)} \approx \frac{1}{\mu\lambda_j} \qquad \textit{where } \lambda_j \textit{ is the jth eigenvalue of } \boldsymbol{R_{xx}}$$

Hence, there is a trade off when selecting the adaptation gain. The convergence requirements for the mean and MSE must be met, be small enough to provide acceptable steady state error, but large enough to ensure the convergence does not take too long.

4.2.3 The computational complexity of the LMS algorithm

The number of multiplications and additions in the computation of the LMS algorithm is provided by the table below.

Expressions	**additions**	**multiplications**
$\hat{y}[n] = \boldsymbol{w}^T(n)\boldsymbol{x}(n)$	$N_w + 1$	N_w
$e[n] = z[n] - \hat{y}[n]$	1	0
$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu e[n]\boldsymbol{x}[n]$	$N_w + 1$	$N_w + 2$
Total	$2N_w + 3$	$2N_w + 2$

Table 8: Calculation

So the total number of operations is $4N_w + 5$, in the order of magnitude $\mathcal{O}(N_w)$, which is way less computationally complex than the Wiener solution.

4.3 Gear shifting

A time-varying adaptation gain was incorporated into the MATLAB routine *lms*. This algorithm makes use of the advantages of both small and large μ values, where large values increase th rate of convergence and small values increase the accuracy in upon convergence. The formula for the time-varying μ is given by:

$$\mu(n+1) = \alpha\mu(n) + \beta e^2(n)$$

where the chose values for $\alpha = 0.99$ and $\beta = 0.001$ (suggested by Kwong, R. H., and Johnston, E. W. Variable step size LMS algorithm. IEEE Transactions on Signal Processing(1992)), where α is close to 1 and β is close to zero to ensure stability. When error is high, μ will increase due to the second term $\beta e^2(n)$ and bigger step sizes are used until a smaller error is achieved. This term diminishes to near zero when error is small, and the first term $\alpha\mu(n)$ will slowly bring the value of the adaptation gain down.
The algorithm was implemented and the behaviour of both the filter weights and the error was plotted in Figure 4.4.

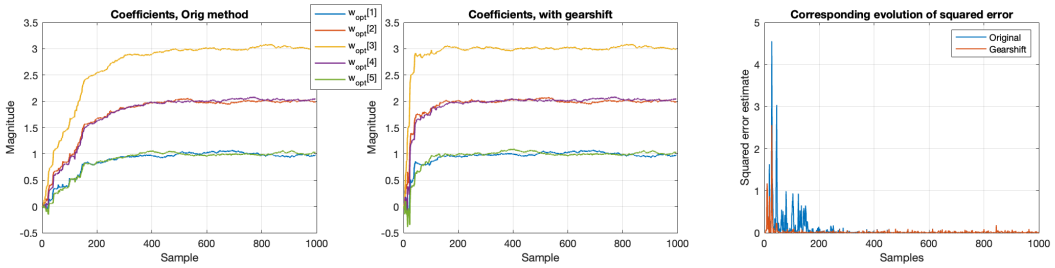


Figure 4.4: Plots of the the filter weights with and without gear shifting, and the error, for $\mu = 0.05$

The maximum permissible overshoot allowed for each coefficient is 20% of its true value and this was incorporated into the code.
The methods with and without gear shifting was compared in terms of the average rising times, which is the average time required for the filter coefficients to go from 10% to 90% of their converged values. This was calculated to be 257 steps for the original method and 105 steps for the method with gear shifting. The latter has a shorter rise time, hence it is the better method.

4.4 Identification of AR processes

4.4.1 Implementing the adaptive LMS algorithm

The adaptive LMS algorithm function in MATLAB was created (*adaptive_lms*) and implemented on an AR model with parameters $a = [1, 0.9, 0.2]$. The results are provided below in Figure 4.5.

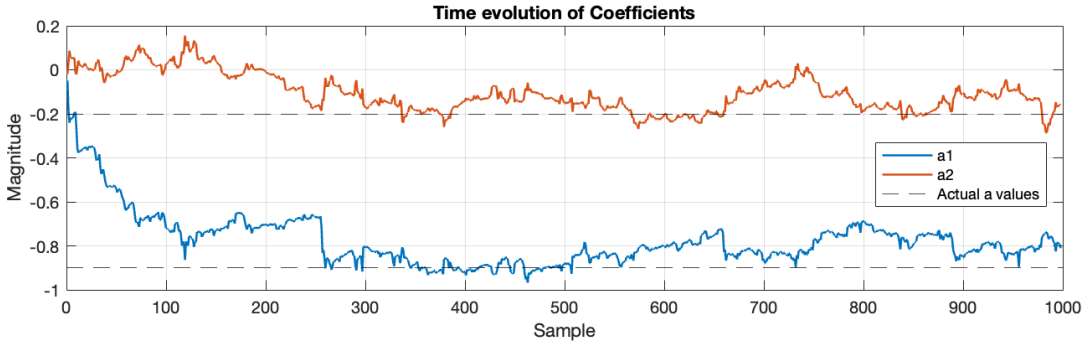


Figure 4.5: Time evolution of the coefficients using adaptive LMS algorithm

The output from the adaptation algorithm (estimate $\hat{x}[n]$) is given by:

$$\hat{x}[n] = \alpha_1 x[n-1] + \alpha_2 x[n-2]$$

Therefore the true value can be represented with:

$$x[n] = \hat{x}[n] - e[n] = \alpha_1 x[n-1] + \alpha_2 x[n-2] - e[n]$$

However, for the AR model with coefficients $\boldsymbol{a} = [a_0, a_1, a_2]$, the relationship of the input $\eta[n]$ and output $x[n]$ is given by:

$$x[n] = -\alpha_1 x[n-1] - \alpha_2 x[n-2] + \eta[n]$$

due to the nature of the algorithm.
In the convergence region, the error $e[n]$ is minimised and is approximately equal to the white noise $\eta[n]$. Hence the α_1 and α_2 that was plotted should converge to -0.2 and -0.9 respectively because the original signal is reconstructed. The adaptive filter works by deconstructing the AR model using a negative feedback loop.

4.4.2 Evolution of the coefficients for four different adaptation gains

The same algorithm is used to analyse how the coefficients evolve with different adaptive gains μ. Figure 4.6 shows the result for $\mu = 0.01, 0.05, 0.1, 0.5$.

The plots suggests that the coefficients converge earlier when the adaptive gain μ increases. However the trade of is, again, that the MSE increases with increasing μ. Integrating gear shifting for the adaptive gain reduces both the convergence rate and MSE.

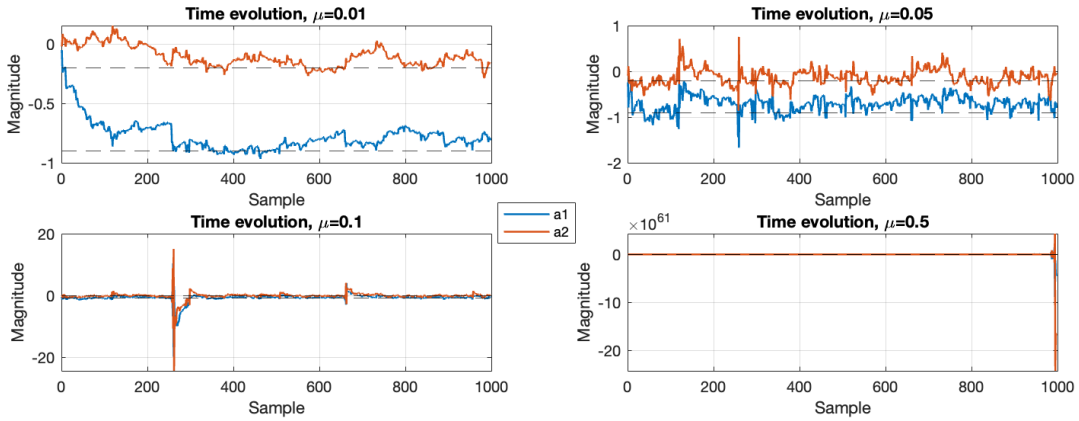


Figure 4.6: Time evolution of the coefficients for a range of μ

4.5 Speech recognition

4.5.1 Test the performance of the adaptive LMS algorithm on real-world audio signals

My own voice corresponding to the sounds "e", "a", "s", "t" and "x" were recorded with sampling frequency $f_s = 44100Hz$ and $N = 1000$ samples each were taken from the. The samples were then used as an input to the adaptive LMS predictor. The performance of the predictor was analysed using the SSE (Sum of Squared Errors, the summation of all $e^2[n]$ values) for a selection of the adaptation gains which were plotted against the order of the predictor, provided in Figure 4.7. Also the adaptive LMS with gear shifting was used and the SSE against model order plotted for the sound of each letter in Figure 4.7.

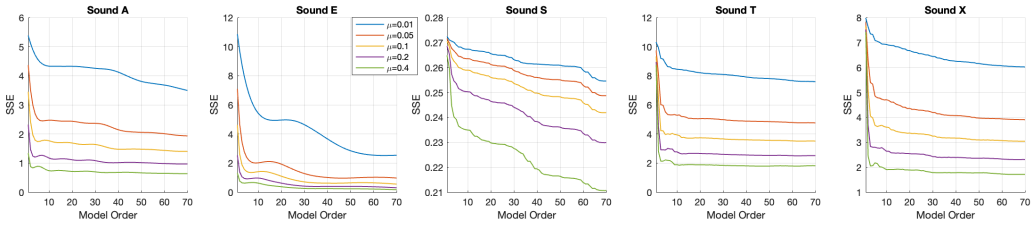


Figure 4.7: SSE against model order for the sound of letters "a,e,s,t,x" for a range of μ

Figure 4.7 indicates that the SSE values for all the sounds and model orders are higher for smaller adaptive gains μ. This is due to the slow rate of convergence as a result of small step sizes. The lower SSE values for higher μ suggests that the the MSE convergence bound has not been reached yet, so we can conclude that the optimal adaptive gain is $\mu = 0.4$ in this case.
Also, from the same figure, a sharp decrease in SSE is first observed which then gradually comes to a plateau as model order increases, with the exception of the sound from letter "s", suggesting that it may be optimally modelled with a higher order. The approach to a constant value is an indication of the optimal model order of the predictor used.

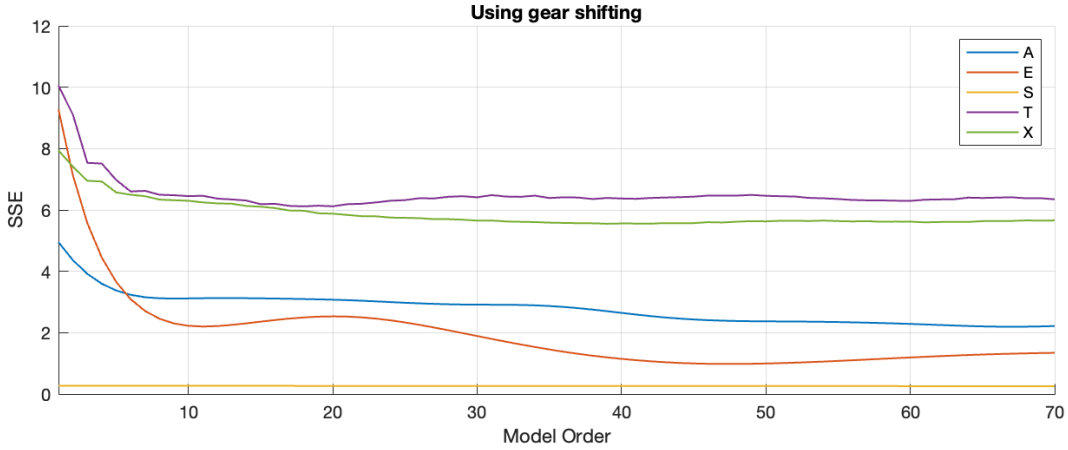


Figure 4.8: SSE against model order for the sound of letters "a,e,s,t,x", with gear shifting

Gear shifting for the adaptive gain was integrated into the adaptive LMS algorithm and, again, the SSE values were plotted against the model order, as displayed in Figure 4.8. We can see that the SSE plot for each sound converges to a different value at a range of model orders, suggesting that each sound has their own optimal model order.

4.5.2 Optimal filter length

The optimal filter length as be determined using the method in the section before, the adaptive LMS, where the order after the convergence of SSE is considered. An alternative way is to observe the evolution of the AR coefficients against time for different model orders. Also, because audio signals are analysed, the reconstructed audio signals that bears the closest resemblance to the original audio would have the corresponding optimal filter order. However, these suggestions are all more time consuming compared to the analytical approaches such as the MDL, AIC and AICc criteria, which were used in section 2. Another analytical approach, that will be introduced in the next section, is by using the prediction gain R_p, with the formula

$$R_p = 10 \times log_{10}(\frac{\sigma_x^2}{\sigma_e^2})$$

where σ_x^2 =variance of input and σ_e^2 =variance of error. And this is a good point to move on to the next section.

4.5.3 Assess the performance of the predictor for each audio recording

The prediction gain measures the performance of the predictors and the optimal filter length is obtained when R_p converges to a maximum value. For the predictors use in the previous sections (for both the 44.1 kHz 16 kHz sampled sounds) the prediction gains were computed and the plot is given by Figure 4.9.

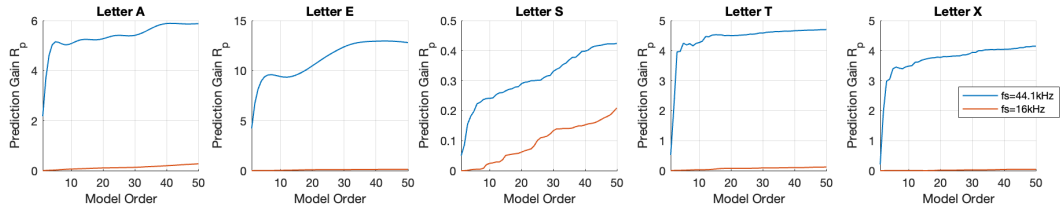


Figure 4.9: Prediction gains for the predictors of the sound of the letters, $\mu = 0.1$

All prediction gain for both cases converges to a maximum values, with the exception of the sound of the letter "s" (again). The point of convergence indicates the optimal filter length. Prediction gain increases up to a maximum value because error signal power decreases when filter order increases.

However, the difference between the prediction gain of both sampling rates is stark. $f_s = 44.1kHz$ has significantly higher prediction gains (for all sounds and model orders) than $f_s = 16kHz$. This is due to the better in resolution provided by the higher sampling rate, which reduces the error of signal and increases prediction gain. So $f_s = 16kHz$ can be concluded to be a terrible rate for prediction compared to $f_s = 44.1kHz$.

4.6 Dealing with computational complexity: sign algorithms

The simplified versions of the LMS algorithm, the class of the sign LMS algorithms, was created in MATLAB. They were simulated and compared to the performance of the basic LMS algorithm ($\mu = 0.01$), for AR parameter identification in Part 4.4, and for one speech file in Part 4.5. The prior is shown in Figure 4.10 and the latter (for the sound of "e") is displayed in Figure 4.11.

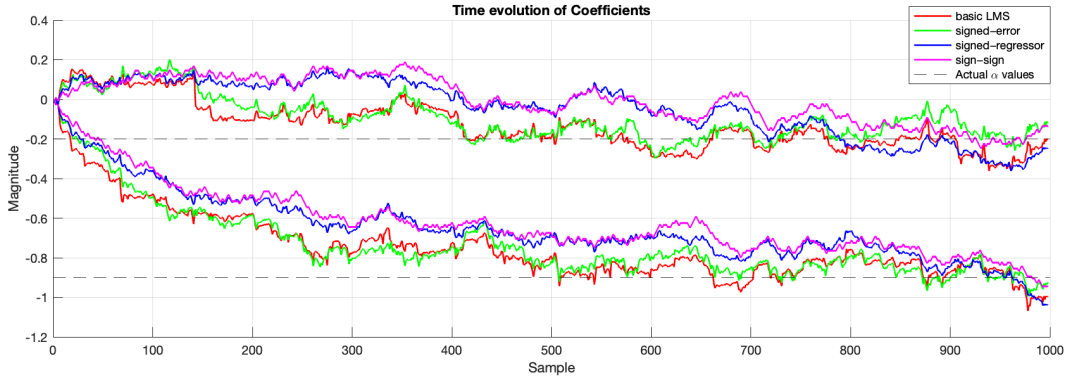


Figure 4.10: Time-Evolution of AR coefficients of process from section 4.4, $\mu = 0.01$

From the figure above, it is observed that the basic LMS algorithm has the highest rate of convergence. The sign-sign algorithm has the slowest convergence rate, which is expected because it is computationally the simplest. The sign-sign algorithm also has smaller deviations when using larger adaptive gains (tested with $\mu = 0.1$), whereas the basic LMS algorithm displayed abrupt divergence from the converged value.

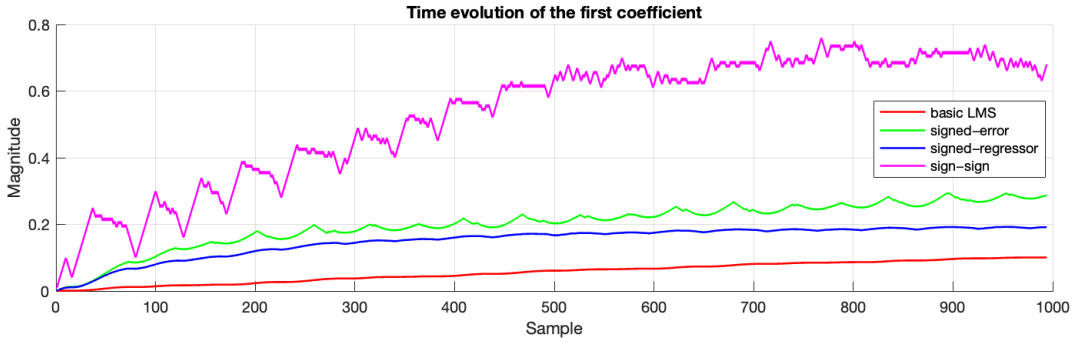


Figure 4.11: Time-Evolution of first coefficient of sound "e" from section 4.5

The time evolution of the first coefficient for the sound of "e" was used to make the comparisons in performance of the algorithms, as shown above. The same observations can be made where the basic LMS algorithm as the fastest rate of convergence and the sign-sign algorithm converges the slowest.

5 MLE for the Frequency of a Signal

The pdf of a real signal $\mathbf{x}$, parametrised by $\boldsymbol{\theta}[A, f_0, \phi]^T$ can be written as

$$p(\boldsymbol{x};\boldsymbol{\theta}) = \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp\left\{-\frac{1}{2\sigma^2}\sum_{n=0}^{N-1}(x[n] - A\cos(2\pi f_0 n + \phi))^2\right\}$$

where $A > 0$ and $0 < f_0 < 1/2$. From the pdf, the maximum-likelihood estimate (MLE), $\hat{f}_0$ (of the frequency f_0) is found by minimising the loss function:

$$J(\boldsymbol{\theta}) = \sum_{n=0}^{N-1}(x[n] - A\cos(2\pi f_0 n + \phi))^2$$

5.1 Remapping the matrix

First, we substitute $\alpha_1 = A\cos\phi$ and $\alpha_2 = -A\sin\phi$ into $J(\boldsymbol{\theta})$ to get:

$$\begin{aligned} J(\boldsymbol{\theta}) &= \sum_{n=0}^{N-1}(x[n] - A\cos(2\pi f_0 n + \phi))^2 = \sum_{n=0}^{N-1}(x[n] - A(\cos(2\pi f_0 n)\cos(\phi) - \sin(2\pi f_0 n)\sin(\phi))^2 \\ &= \sum_{n=0}^{N-1}[x[n] - \alpha_1\cos(2\pi f_0 n) - \alpha_2\sin(2\pi f_0 n)]^2 \end{aligned}$$

Then we can get $\boldsymbol{B}$ where $J(\boldsymbol{\theta}) = \boldsymbol{B}^T\boldsymbol{B}$.

$$\begin{aligned} \boldsymbol{B} &= \begin{bmatrix} x[0] - \alpha_1\cos(0) - \alpha_2\sin(0) \\ \vdots \\ x[N-1] - \alpha_1\cos(2\pi f_0(N-1)) - \alpha_2\sin(2\pi f_0(N-1)) \end{bmatrix} \\ &= \begin{bmatrix} x[0] \\ \vdots \\ x[N-1] \end{bmatrix} - \alpha_1\begin{bmatrix} \cos(0) \\ \vdots \\ \cos(2\pi f_0(N-1)) \end{bmatrix} - \alpha_2\begin{bmatrix} \sin(0) \\ \vdots \\ \sin(2\pi f_0(N-1)) \end{bmatrix} = \boldsymbol{x} - \alpha_1\boldsymbol{c} - \alpha_2\boldsymbol{s} \end{aligned}$$

where $\boldsymbol{c} = [1, \cos(2\pi f_0), \ldots, \cos(2\pi f_0(N-1))]^T$ and $\boldsymbol{s} = [1, \sin(2\pi f_0), \ldots, \sin(2\pi f_0(N-1))]^T$.
At this stage, we have mapped $J'(\alpha_1, \alpha_2, f_0) = (\boldsymbol{x} - \alpha_1\boldsymbol{c} - \alpha_2\boldsymbol{s})^T(\boldsymbol{x} - \alpha_1\boldsymbol{c} - \alpha_2\boldsymbol{s})$.
$\boldsymbol{B}$ can be further simplifies as such:

$$\begin{aligned} \boldsymbol{B} &= \begin{bmatrix} x[0] \\ \vdots \\ x[N-1] \end{bmatrix} - \alpha_1\begin{bmatrix} \alpha_1\cos(0) + \alpha_2\sin(0) \\ \vdots \\ \alpha_1\cos(2\pi f_0(N-1)) + \alpha_2\sin(2\pi f_0(N-1)) \end{bmatrix} \\ &= \begin{bmatrix} x[0] \\ \vdots \\ x[N-1] \end{bmatrix} - \begin{bmatrix} \cos(0) & \sin(0) \\ \vdots & \vdots \\ \cos(2\pi f_0(N-1)) & \sin(2\pi f_0(N-1)) \end{bmatrix}\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \boldsymbol{x} - \boldsymbol{H}\boldsymbol{\alpha} \end{aligned}$$

Finally, we get $J'(\alpha_1, \alpha_2, f_0) = (\boldsymbol{x} - \boldsymbol{H}\boldsymbol{\alpha})^T(\boldsymbol{x} - \boldsymbol{H}\boldsymbol{\alpha}) = J'(\boldsymbol{\alpha}, f_0)$, where $\boldsymbol{H} = [\boldsymbol{c}, \boldsymbol{s}]$ and $\boldsymbol{\alpha} = [\alpha_1, \alpha_2]^T$

5.2 Find the minimizing solution

First we get the derivative of $J'(\boldsymbol{\alpha}, f_0)$ w.r.t. α and set it to zero:

$$\begin{aligned} J'(\boldsymbol{\alpha}, f_0) &= (\boldsymbol{x} - \boldsymbol{H}\boldsymbol{\alpha})^T(\boldsymbol{x} - \boldsymbol{H}\boldsymbol{\alpha}) = \boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{H}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\boldsymbol{H}^T\boldsymbol{x} + \boldsymbol{\alpha}^T\boldsymbol{H}^T\boldsymbol{H}\boldsymbol{\alpha} \\ &= \boldsymbol{x}^T\boldsymbol{x} - 2\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\boldsymbol{H}^T\boldsymbol{H}\boldsymbol{\alpha} \\ \frac{\partial J'(\boldsymbol{\alpha}, f_0)}{\partial\boldsymbol{\alpha}} &= -2\boldsymbol{H}^T\boldsymbol{x} + 2\boldsymbol{H}^T\boldsymbol{H}\boldsymbol{\alpha} = 0 \end{aligned}$$

Hence, after rearranging, we get the the minimising solution $\hat{\boldsymbol{\alpha}}$:

$$\hat{\boldsymbol{\alpha}} = (\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{x}$$

Then we substitute it into the loss function to get the minimum value for the loss function.

$$\begin{aligned} J'(\hat{\boldsymbol{\alpha}}, f_0) &= (\boldsymbol{x} - \boldsymbol{H}\hat{\boldsymbol{\alpha}})^T(\boldsymbol{x} - \boldsymbol{H}\hat{\boldsymbol{\alpha}}) = \left[\boldsymbol{x} - \boldsymbol{H}(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{x}\right]^T \left[\boldsymbol{x} - \boldsymbol{H}(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{x}\right] \\ &= \boldsymbol{x}^T \left[\boldsymbol{I} - \boldsymbol{H}(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\right]^T \left[\boldsymbol{I} - \boldsymbol{H}(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\right]\boldsymbol{x} \end{aligned}$$

$\boldsymbol{I}$ is the $N \times N$ identity matrix and $\left[\boldsymbol{I} - \boldsymbol{H}(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\right]$ is $N \times N$ symmetric matrix which is idempotent, where a matrix equals its transpose and the square of matrix produces the same matrix. So $J'(\hat{\boldsymbol{\alpha}}, f_0)$ is further simplified into:

$$J'(\hat{\boldsymbol{\alpha}}, f_0) = \boldsymbol{x}^T \left[\boldsymbol{I} - \boldsymbol{H}(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\right]\boldsymbol{x} = \boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{H}(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{x}$$

Since $\boldsymbol{x}^T\boldsymbol{x}$ is independent from $\boldsymbol{\alpha}$, and in order to minimise $J'(\hat{\boldsymbol{\alpha}}, f_0)$, $\boldsymbol{x}^T\boldsymbol{H}(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{x}$ has to be maximised.

5.3 Show that the MLE of the frequency f_0 is the maximising value

For ease of derivation, we will have $\boldsymbol{x}^T\boldsymbol{H}(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{x} = (\boldsymbol{H}^T\boldsymbol{x})^T(\boldsymbol{H}^T\boldsymbol{H})^{-1}(\boldsymbol{H}^T\boldsymbol{x})$ where:

$$\boldsymbol{H}^T\boldsymbol{x} = \begin{bmatrix} \cos(0) & \dots & \cos(2\pi f_0(N-1)) \\ \sin(0) & \dots & \sin(2\pi f_0(N-1)) \end{bmatrix} \begin{bmatrix} x[0] \\ \vdots \\ x[N-1] \end{bmatrix} = \begin{bmatrix} \boldsymbol{c}^T \\ \boldsymbol{s}^T \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{c}^T\boldsymbol{x} \\ \boldsymbol{s}^T\boldsymbol{x} \end{bmatrix}$$

$$\begin{aligned} \boldsymbol{H}^T\boldsymbol{H} &= \begin{bmatrix} \cos(0) & \dots & \cos(2\pi f_0(N-1)) \\ \sin(0) & \dots & \sin(2\pi f_0(N-1)) \end{bmatrix} \begin{bmatrix} \cos(0) & \sin(0) \\ \vdots & \vdots \\ \cos(2\pi f_0(N-1)) & \sin(2\pi f_0(N-1)) \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{c}^T \\ \boldsymbol{s}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{c} & \boldsymbol{s} \end{bmatrix} = \begin{bmatrix} \boldsymbol{c}^T\boldsymbol{c} & \boldsymbol{c}^T\boldsymbol{s} \\ \boldsymbol{s}^T\boldsymbol{c} & \boldsymbol{s}^T\boldsymbol{s} \end{bmatrix} \end{aligned}$$

$$\boldsymbol{x}^T\boldsymbol{H}(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{x} = \begin{bmatrix} \boldsymbol{c}^T\boldsymbol{x} \\ \boldsymbol{s}^T\boldsymbol{x} \end{bmatrix}^T \begin{bmatrix} \boldsymbol{c}^T\boldsymbol{c} & \boldsymbol{c}^T\boldsymbol{s} \\ \boldsymbol{s}^T\boldsymbol{c} & \boldsymbol{s}^T\boldsymbol{s} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{c}^T\boldsymbol{x} \\ \boldsymbol{s}^T\boldsymbol{x} \end{bmatrix}$$

Hence the frequency f_0 is the value that maximizes $\boldsymbol{x}^T\boldsymbol{H}(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{x}$.

5.4 Justify why it is required for f_0 not to to be close to 0 or 1/2

When f_0 is not close to 0 or 1/2, the derivation above, derived from $f_0 \approx 0.25$ for example, can be approximated by

$$\begin{bmatrix} \boldsymbol{c}^T\boldsymbol{x} \\ \boldsymbol{s}^T\boldsymbol{x} \end{bmatrix}^T \begin{bmatrix} \frac{N}{2} & 0 \\ 0 & \frac{N}{2} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{c}^T\boldsymbol{x} \\ \boldsymbol{s}^T\boldsymbol{x} \end{bmatrix}$$

where the central matrix can be inverted and it makes calculations possible. The prove that $f_0 = 0$ and 0.5 gives a non-invertible central matrix, because they are singular matrices, is provided below.

For $f_0 = 0$:

$$\boldsymbol{H}^T\boldsymbol{H} = \begin{bmatrix} \boldsymbol{c}^T\boldsymbol{c} & \boldsymbol{c}^T\boldsymbol{s} \\ \boldsymbol{s}^T\boldsymbol{c} & \boldsymbol{s}^T\boldsymbol{s} \end{bmatrix} = \begin{bmatrix} 1 & \dots & 1 \\ 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \vdots & \vdots \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} N & 0 \\ 0 & 0 \end{bmatrix}$$

For $f_0 = 0.5$:

$$\boldsymbol{H}^T\boldsymbol{H} = \begin{bmatrix} \boldsymbol{c}^T\boldsymbol{c} & \boldsymbol{c}^T\boldsymbol{s} \\ \boldsymbol{s}^T\boldsymbol{c} & \boldsymbol{s}^T\boldsymbol{s} \end{bmatrix} = \begin{bmatrix} 1 & -1 & \dots & -1 & 1 \\ 0 & 0 & \dots & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ \vdots & \vdots \\ -1 & 0 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} N & 0 \\ 0 & 0 \end{bmatrix}$$

Therefore

$$\begin{aligned} \begin{bmatrix} \boldsymbol{c}^T\boldsymbol{x} \\ \boldsymbol{s}^T\boldsymbol{x} \end{bmatrix}^T \begin{bmatrix} \frac{2}{N} & 0 \\ 0 & \frac{2}{N} \end{bmatrix} \begin{bmatrix} \boldsymbol{c}^T\boldsymbol{x} \\ \boldsymbol{s}^T\boldsymbol{x} \end{bmatrix} &= \frac{2}{N}[(\boldsymbol{c}^T\boldsymbol{x})^2 + (\boldsymbol{s}^T\boldsymbol{x})^2] \\ &= \frac{2}{N}\left[\left(\sum_{n=0}^{N-1} x[n]\cos(2\pi f_0 n)\right)^2 + \left(\sum_{n=0}^{N-1} x[n]\sin(2\pi f_0 n)\right)^2\right] \\ &= \frac{2}{N}\left|\sum_{n=0}^{N-1} x[n]e^{-j2\pi f_0 \frac{n}{N}}\right|^2 = \hat{P}_X(f_0) \qquad \textit{Where } \hat{P}_X(f_0) \textit{ is the periodogram} \end{aligned}$$

Hence the MLE $\hat{f}_0$ is obtained by maximizing the periodogram.

5.5 The behaviour when f_0 approaches 0 or 1/2

Using the noiseless data $x[n] = \cos(2\pi f_0 n$ with $n = 1, 2, \dots, N-1$ for $N = 10$, The periodogram and the MLE estimate was plotted and displayed in Figure 5.1 below.

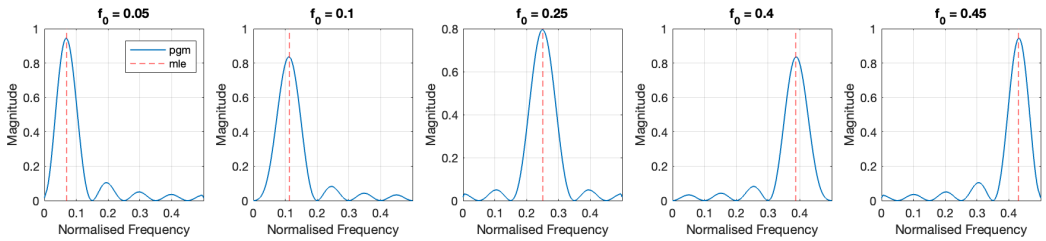


Figure 5.1: Periodogram and MLE estimates of noiseless data

Looking at the plots for each value of f_0, an observation is made that as f_0 approaches 0 or 0.5, the MLE deviates further away from its true value. And at $f_0 = 0.25$, the MLE is exactly its true value.