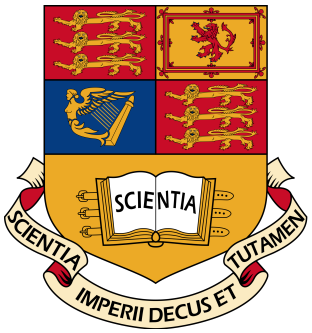

Imperial College London
Electrical & Electronic Engineering

Advanced Signal Processing Coursework

Author: Sushanth Kolluru *CID:* 01564794

Contents

1 Random signals and stochastic processes

1.1 Statistical Estimation

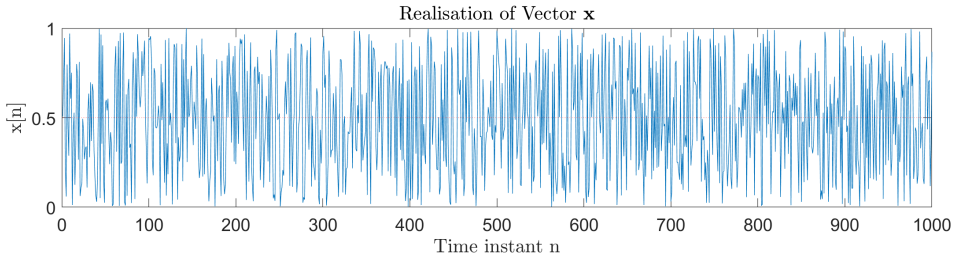

Figure 1: 1000-sample realisation of $\mathbf{x}$, where $x[n] \sim \mathcal{U}(0,1)$

Initially, a 1000 sample vector labelled $\mathbf{x}$ is created, where each sample $x[n]$ is a realisation at the time instant n of a uniform random variable $X \sim \mathcal{U}(0,1)$. The vector was constructed using the MATLAB function `rand`, and has been plotted in figure 1. The probability density function (pdf) for a uniform distribution has been defined in equation 1.

$$pdf_X(x) = \begin{cases} 1 & \text{if } 0 \leq x \leq 1. \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

1.1.1 Theoretical Mean and Theoretical Standard Deviation

The expected values for the random variables X and X^2 are defined below:

$$\mathbb{E}\{X\} = \int_{-\infty}^{\infty} x pdf_X(x)\, dx \qquad \mathbb{E}\{X^2\} = \int_{-\infty}^{\infty} x^2 pdf_X(x)\, dx \tag{2.1 , 2.2}$$

Using these definitions and the probability density function defined above, one can derive that $\mathbb{E}\{X\} = \frac{1}{2}$ and $\mathbb{E}\{X^2\} = \frac{1}{3}$. The theoretical mean and theoretical standard deviation were then calculated:

$$m = \mathbb{E}\{X\} = \frac{1}{2}$$
$$\sigma = \sqrt{\mathbb{E}\{X - \mathbb{E}\{X\}\}^2} = \sqrt{\mathbb{E}\{X^2\} - (\mathbb{E}\{X\})^2} = \frac{1}{\sqrt{12}}$$

1.1.2 Sample Mean and Sample Standard Deviation

The sample mean is given by the equation:

$$\hat{m} = \frac{1}{N} \sum_{n=1}^{N} x[n] \tag{2}$$

This is done with relative simplicity on MATLAB using the function `mean`. In one iteration, the sample mean obtained for $\mathbf{x}$ was equal to 0.5059, a percentage difference of just 1.18% when compared to the theoretical value. It is worth noting that if the number of samples of $\mathbf{x}$ is increased, the estimator converges to its theoretical counterpart. In other words, as the number of samples increases, there will be a resultant increase in accuracy and in turn a decrease in error.

The sample standard deviation is given by the equation:

$$\hat{\sigma} = \sqrt{\frac{1}{N-1} \sum_{n=1}^{N} (x[n] - \hat{m})^2} \tag{3}$$

Similar to the sample mean, applying this on MATLAB can be done easily using the function `std`. In the same iteration used earlier, the sample standard deviation was 0.2919; a the percentage error of only 1.1%. Like before, increasing the number of samples will result in an increase in accuracy and in turn a decrease in error.

1.1.3 Bias in Estimators

For this section, ten different realisations of $\mathbf{x}$ (each of size 1000) have been generated. From these, the bias within the estimators was calculated and assessed. The sample mean and sample standard deviation of each realisation have been plotted below, centred around their theoretical counterparts:

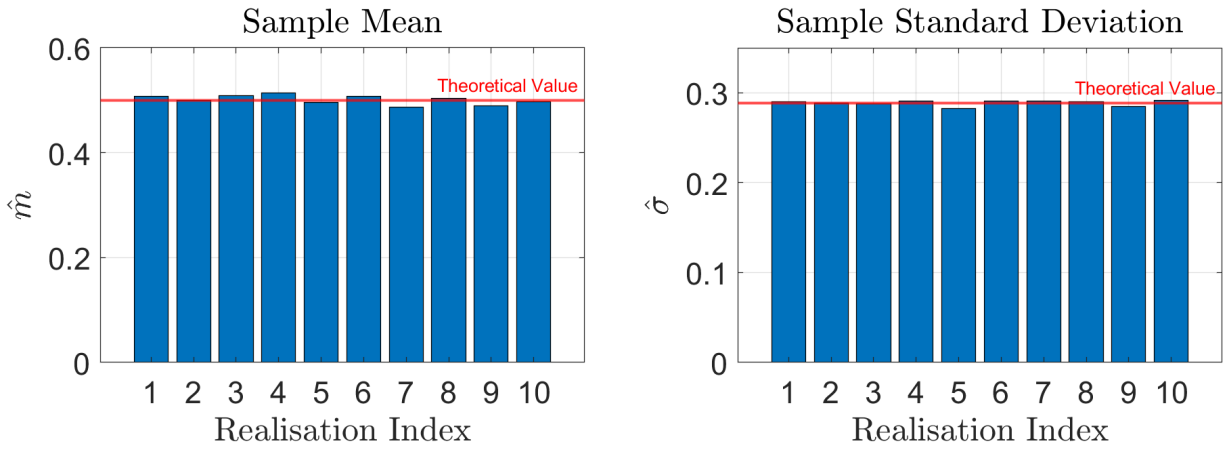

Figure 2: Sample means and sample standard deviations - Uniform Distribution

One can clearly see that the estimates vary from iteration to iteration, but the percentage error between the estimates and their theoretical counterparts remain low, never exceeding 3% in any realisation for either estimator. The biases for the sample mean and sample standard deviation were calculated to be 1.1×10^{-3} and 1.0132×10^{-4} respectively, with both estimators clustering closely around their theoretical counterparts. As a result, the biases within these estimators can be regarded as negligible. If the number of samples or the number of realisations were increased, the bias would reduce further in size.

1.1.4 Approximating the Probability Density Function

To obtain a complete statistical description it is necessary to examine the probability density function (pdf) from which the samples are drawn. This section will focus on how one can approximate the probability density function of X. The specification suggests that this should be done by making a generating a histogram with $\mathbf{x}$, and then normalising the y axis by the total number of samples.

The suggested normalisation is a poor approximation of the theoretical pdf, as the probability is defined by the y value of each bar, rather than the area of each bar; this behaviour is fundamentally different. This is easily mitigated and is will be applied in a section 1.3. To allow for a useful comparison of the outputs using different sample numbers and different bin counts within this task, the theoretical pdf will be scaled accordingly such that comments regarding accuracy can still be made. The resultant outputs, with ranging bin sizes and sample size are plotted in figure 3.

From figure 3, it can be concluded that as we increase the number of generated samples we tend towards the theoretical output. If the sample size is sufficient in size (as in the bottom right of figure 3), a greater number of bins results in a better and more detailed estimate. However, if the sample size is insufficient, the approximation loses its resemblance to the theoretical output. The impact of a reduced sample size is less significant if the bin width is greater. Ideally, we want a small bin width to improve the resolution, and as many samples as possible to improve the estimate's accuracy.

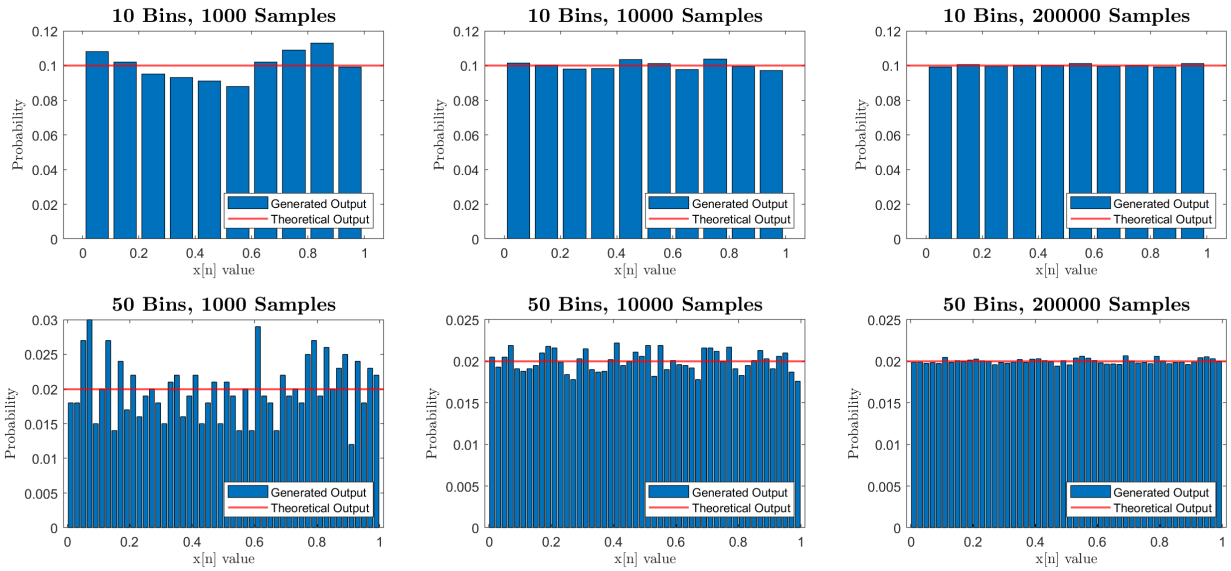

Figure 3: Theoretical and empirical Outputs with varying sample sizes and bin counts - Normal Distribution

1.1.5 Estimator Analysis on Gaussian Distribution

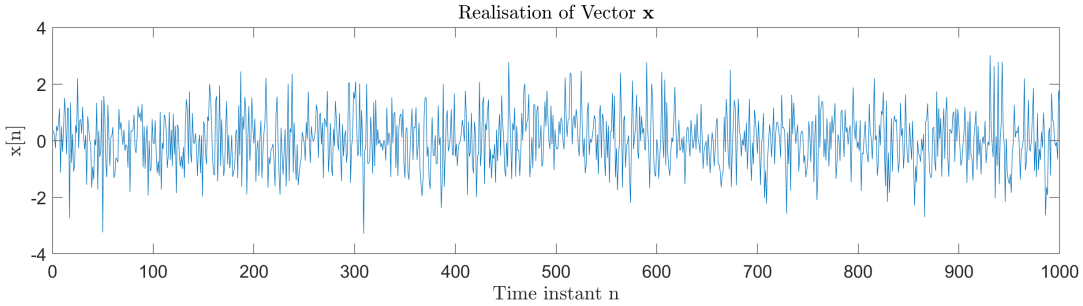

Figure 4: 1000-sample realisation of $\mathbf{x}$, where $x[n] \sim \mathcal{N}(0,1)$

The same analysis is to be repeated, where $\mathbf{x}$ is created and each sample $x[n]$ is a realisation at the time instant n of a random variable $X \sim \mathcal{N}(0,1)$. The probability density function is defined in equation 4.

$$pdf_X(x) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{x^2}{2}) \tag{4}$$

Using this pdf and equations 2.1 and 2.2 one can work out that $\mathbb{E}\{X\} = 0$ and $\mathbb{E}\{X^2\} = 1$ for a uniform distribution.The theoretical mean and theoretical standard deviation are then also 0 and 1 respectively. In one iteration, the sample mean was -0.0086, and the sample standard deviation was 0.9877; an absolute error of 0.0086 for the mean and 0.0123 for the standard deviation. If the number of samples is increased, these estimators will improve with respect to accuracy.

The sample means and standard deviations for ten different realisations are plotted in figure 5. Like before, the estimates vary from iteration to iteration, but the absolute difference between the estimates and their theoretical counterparts remain low, never exceeding 0.05 in any realisation for both estimates. The estimators cluster closely to the theoretical values. The biases for the sample mean and sample standard deviation were calculated to be 0.0029 and 0.0076 respectively. As a result, the biases within these estimators can be regarded as negligible. If the number of samples or the number of realisations were increased, the bias will

reduce further in size.

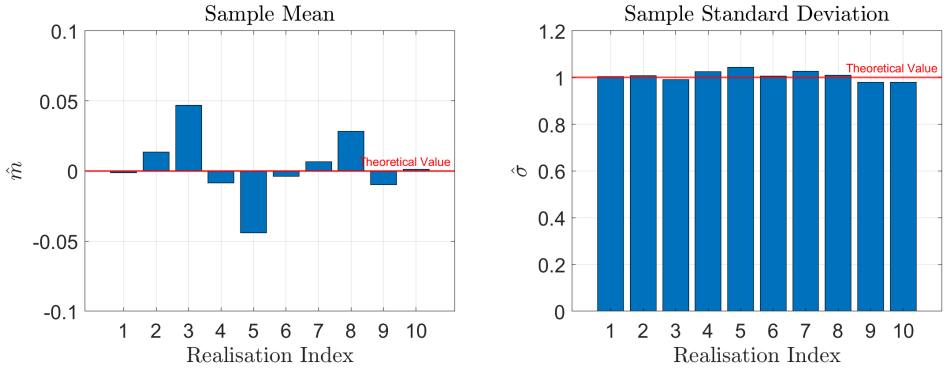

Figure 5: Sample means and sample standard deviations - Gaussian Distribution

Lastly, the pdf estimates are plotted below, with varying bins and sample sizes. The function `histogram` was employed in this subsection for the sake of convenience, with the same normalisation factor being applied as before. The conclusions that can be made from these results are the exact same as those made in Section 1.1.4: increasing the number of generated samples results in the output tending towards the theoretical output. If the sample size is sufficient (as in the bottom right of figure 3), a greater number of bins results in a better and more detailed estimate, but if the sample size is insufficient, the approximation loses its resemblance to the theoretical output. The impact of a reduced sample size is less significant if the bin width is greater. Ideally, we want a small bin width to improve the resolution, and as many samples as possible to improve the estimate's accuracy. The most accurate result was that in the bottom right of figure 6.

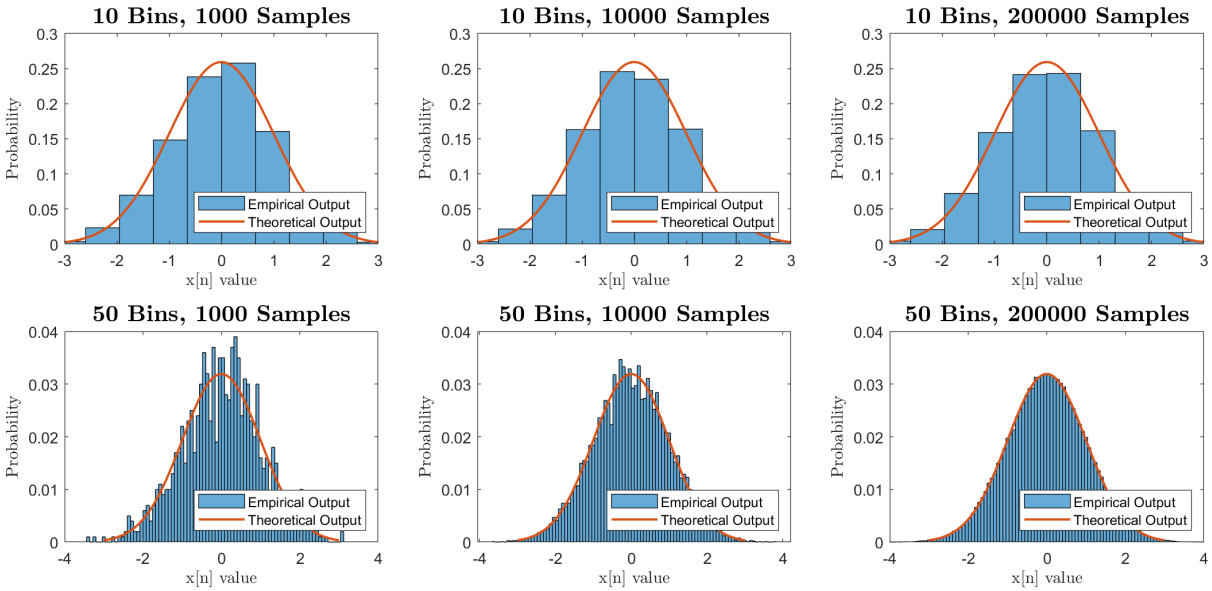

Figure 6: Theoretical and empirical outputs with varying sample sizes and bin counts - Gaussian Distribution

1.2 Stochastic Processes

This section will look at and analyse the behaviour of three different stochastic processes. More specifically, this section will study the three random processes and assess whether the processes are stationary and ergodic. For a random process to be considered stationary, the mean and standard deviation must remain constant with time. For a random process to be considered ergodic, the mean and standard deviation must remain constant across multiple realisations along with time. The final part of this section will derive

mathematical expressions for each random process, and obtain the theoretical mean and standard deviation.

1.2.1 Stationarity of rp1, rp2 and rp3

For each random process, 100 members of the ensemble were used, each of length 100. Using MATLAB, the ensemble mean and standard deviation have been plotted in figure 7. For rp1, the mean seems to vary linearly with time. Likewise, the standard deviation also seems to vary with time, with what is assumed to be some sort of sinusoidal relationship. As a result, **rp1 is considered to be a non-stationary process**. For rp2 and rp3, the ensemble mean and standard deviation seem to cluster around fixed values, with no clear time dependent relationship. Hence, **both rp2 and rp3 are considered to be stationary processes**. These empirical results are validated later in Section 1.2.3, where the theoretical results are calculated.

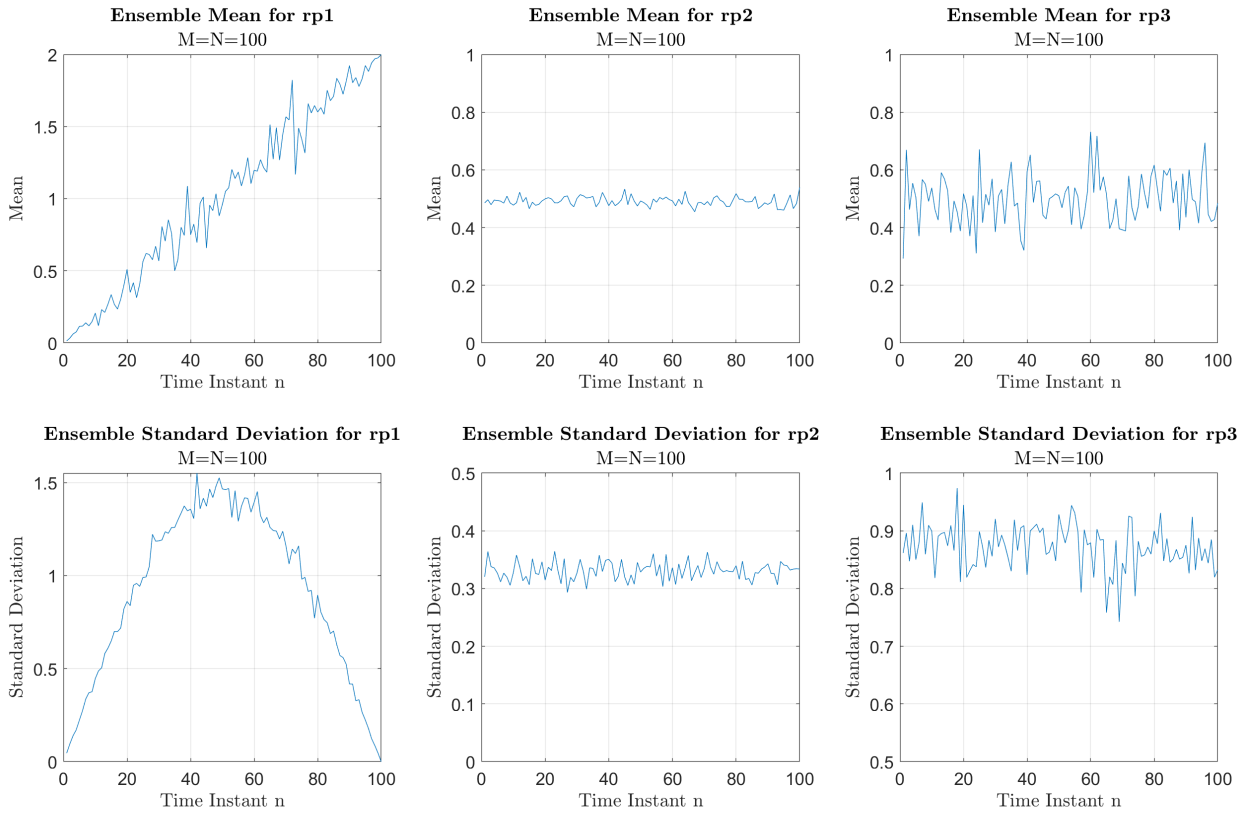

Figure 7: Ensemble Means and Standard Deviation of each random process

1.2.2 Ergodicity of rp1, rp2 and rp3

To study the ergodicity of each process, the mean and standard deviation for four independent realisations with 1000 samples were calculated. The results are shown in figure 8. It can immediately concluded that **rp1 is a non-ergodic process**, given that it is non stationary. This is the case even though the mean and standard deviation remain consistent from realisation to realisation. The results seem to indicate that **rp2 is also a non-ergodic process**, given that there are significant fluctuations in both the mean and standard deviation from realisation to realisation (both sample mean and sample std have ranges greater than the average sample mean and std). Lastly, the results indicate that **rp3 is an ergodic process**, with the sample means clustering around 0.53 (with a range of approximately 0.15), and the sample standard deviation clustering around 0.865 (with a range of 0.0434).

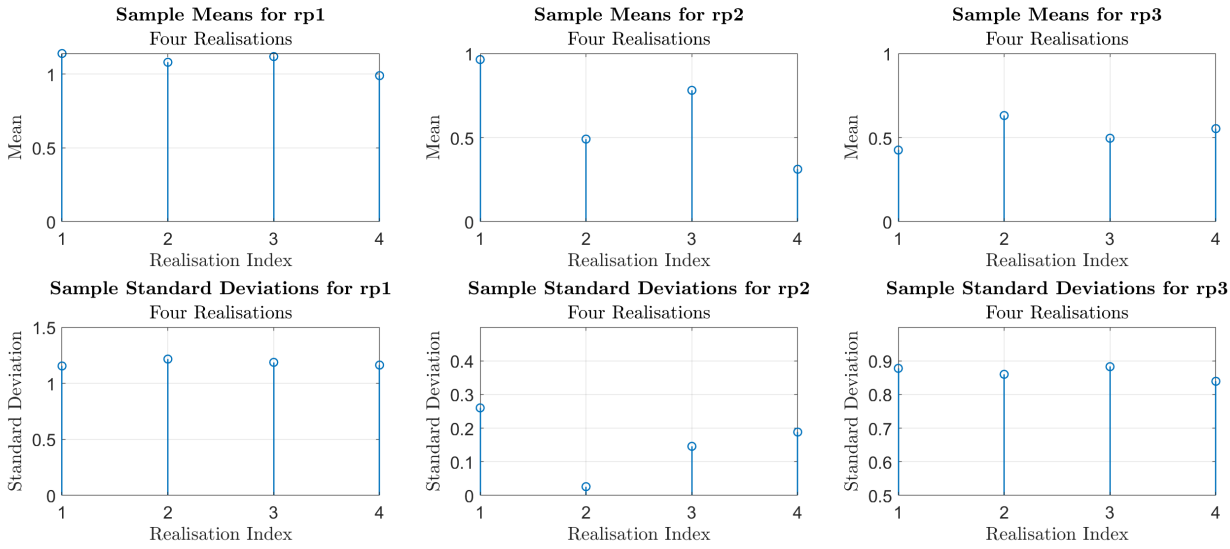

Figure 8: Sample Means and Standard Deviation for each random process across Four Realisations

1.2.3 Mathematical Descriptions of rp1, rp2 and rp3

Random Process 1: rp1 can be represented mathematically by the following equation:

$$rp1 = 5\left(U - \frac{1}{2}\right)\ sin\left(\frac{n\pi}{N}\right) + \frac{1}{50}n \tag{5}$$

where U is a uniformly distributed random variable, with $\mathbb{E}\{U\} = \frac{1}{2}$, and $\mathbb{E}\{U^2\} = \frac{1}{3}$.Using this, the mean and standard deviation are calculated:

$$m_{rp1} = \mathbb{E}\{rp1\} = \mathbb{E}\left\{5\left(U - \frac{1}{2}\right)\ \sin\left(\frac{n\pi}{N}\right) + \frac{1}{50}n\right\} = 5\sin\left(\frac{n\pi}{N}\right)\ \left(\mathbb{E}\{U\} - \frac{1}{2}\right) + \mathbb{E}\left\{\frac{1}{50}n\right\}$$
$$= \frac{1}{50}n$$
$$\sigma^2_{rp1} = \mathbb{E}\{(rp1 - m_{rp1})^2\} = \mathbb{E}\left\{5\left(U - \frac{1}{2}\right)\ \sin\left(\frac{n\pi}{N}\right) + \frac{1}{50}n - \frac{1}{50}n\right\} = 25\sin^2\left(\frac{n\pi}{N}\right) \times \mathbb{E}\left\{(U - \frac{1}{2})^2\right\}$$
$$= 25\sin^2\left(\frac{n\pi}{N}\right)$$
$$\implies \sigma_{rp1} = \frac{5}{\sqrt{12}}\ \sin\left(\frac{n\pi}{N}\right)$$

The theoretical mean and standard deviation are both dependent on time. This further highlights that rp1 is both non-stationary and non-ergodic. When comparing the obtained theoretical mean and standard deviation with the empirical results, it is evident that the empirical and theoretical results corroborate each other nicely. The linear behaviour in the theoretical mean can be seen clearly in the empirical data, and the sinusoidal shape implied by the theoretical standard deviation can be clearly seen in the empirical data.

Random Process 2: rp2 can be represented mathematically by the following equation:

$$rp2 = U_1 + U_2\ (U_3 - 0.5) \tag{6}$$

where U_1 and U_2 and U_3 are all different realisations of a uniform distribution. $\mathbb{E}\{U_1\} = \mathbb{E}\{U_2\} = \mathbb{E}\{U_3\} = \frac{1}{2}$,

and $\mathbb{E}\{U_1^2\} = \mathbb{E}\{U_2^2\} = \mathbb{E}\{U_3^2\} = \frac{1}{3}$ Using this, the mean and standard deviation are calculated:

$$m_{rp2} = \mathbb{E}\{rp2\} = \mathbb{E}\{U_1\} + \mathbb{E}\{U_2\}\ [\mathbb{E}\{U_3\} - 0.5]\ = \frac{1}{2}$$
$$\sigma_{rp2}^2 = \mathbb{E}\{rp2^2\} - m_{rp2}^2 = \mathbb{E}\{(U_1 + U_2\ (U_3 - 0.5))^2\} - \frac{1}{4}$$
$$= \mathbb{E}\{U_1^2\} + 2\mathbb{E}\{U_1\ U_2\ (U_3 - 0.5)\} + \ \mathbb{E}\{U_2^2\ (U_3 - 0.5)^2\} - \frac{1}{4}$$
$$= \frac{1}{3} + \frac{1}{3}\left(\frac{1}{3} - \frac{1}{2} + \frac{1}{4}\right) - \frac{1}{4} = \frac{1}{9}, \quad \implies \sigma_{rp2} = \frac{1}{3}$$

The theoretical mean and standard deviation are both time independent, corroborating. This further highlights that rp2 is stationary. Like rp1, the empirical and theoretical results corroborate each other nicely. The non-ergodic behaviour can also be predicted with the mathematical expression, given that V varying with both time and realisation. This non-ergodic behaviour is seen in the empirical data.

Random Process 3: rp3 can be represented mathematically by the following equation:

$$rp3 = 3\left(U - \frac{1}{2}\right) + \frac{1}{2} = 3U - \frac{3}{2} + \frac{1}{2} = 3U - 1 \tag{7}$$

where U has the same definition as in rp1. Using this, the mean and standard deviation are calculated:

$$m_{rp3} = \mathbb{E}\{rp3\} = \mathbb{E}\{3U - 1\} = 3\mathbb{E}\{U\} - 1 = 0.5$$
$$\sigma_{rp3}^2 = \mathbb{E}\{(rp3^2\} - m_{rp3}^2 = \mathbb{E}\{(3U - 1)^2\} - \frac{1}{4} = 9\ \mathbb{E}\{U^2\} - 6\ \mathbb{E}\{U\} + 1 - \frac{1}{4} = \frac{3}{4}$$
$$\implies \sigma_{rp3} = \frac{\sqrt{3}}{2}$$

The theoretical mean and standard deviation are both independent on time. This further highlights that rp3 is stationary. The theoretical and empirical results are consistent, and the process is ergodic. This means that the statistical properties can be obtained with one realisation of adequate duration.

1.3 Estimation of Probability Distributions

As discussed earlier, the probability density function is an essential element to providing a statistical description of a process. The function `pdf` (figure 9) was written to provide an estimate for the probability density function of a given collection of samples. This differs from the estimator used in section 1.1.4, as here the histogram is normalised further such that the probability of a given bin is provided by its area. The total area of the histogram is now equal to one.

```
1    function pdf(x, n_bins);
2        [f, x] = hist(x, n_bins);
3        f = f./trapz(x,f);
4        bar(x,f,1);
5    end
```

Figure 9: `pdf` function code

1.3.1 PDF of Gaussian Distribution

To verify whether the pdf function that was written is working as expected, the function was tested for a WGN signal of length N, where $N \in 100, 1000, 10000$. The plots are shown in figure 10.

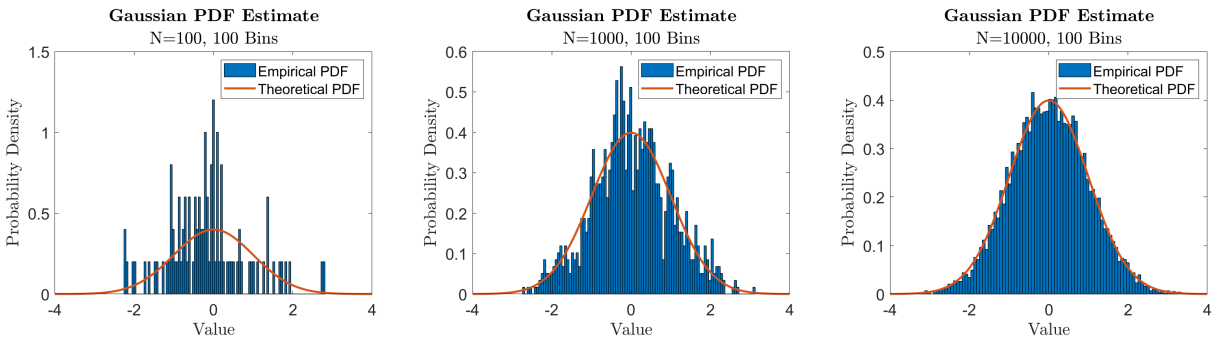

Figure 10: PDF Estimates of Gaussian Distribution

1.3.2 Estimating rp3's Probability Density Function

$$pdf_{rp3}(x) = \begin{cases} \frac{1}{3} & \text{if } -1 \leq x \leq 2. \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

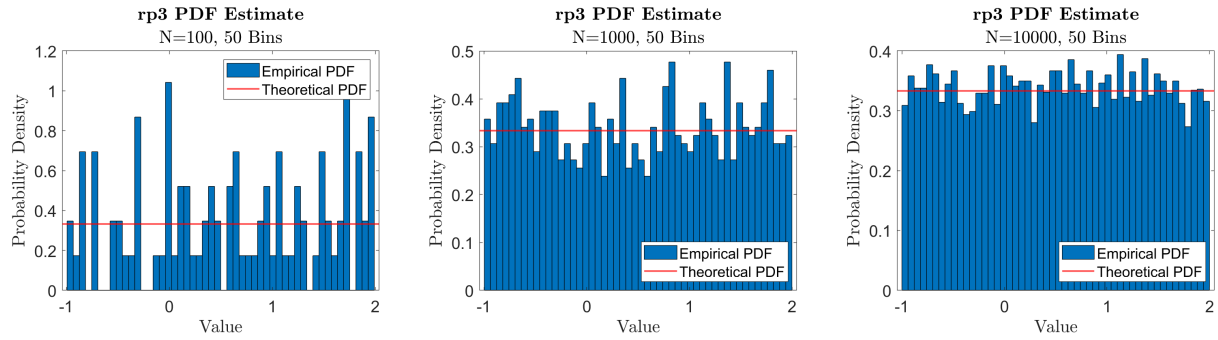

Figure 11: PDF Estimates of Gaussian Distribution

Of the three stochastic processes described in the previous section, only rp3 was ergodic and stationary, and so the pdf for rp3 has been estimated for a varying number of sample sizes. The theoretical pdf for the uniform distribution is defined in equation 8. The estimates are plotted in figure 11. It is clear from both figures 10 and 11 that a large N should be used to generate an accurate estimate of the pdf, with the empirical pdf converging to the theoretical pdf as the number of sample (N) increases. This is similar to the result seen in Section 1.1.4.

1.3.3 Estimating a Non-Stationary PDF

If we apply the `pdf` function to a non-stationary process, we do end up with a distribution that lacks information regarding how the mean and standard deviation vary across N due to the time averaging behaviour of this method (we will get the 'average' pdf). Hence, the `pdf` function cannot provide a good estimate of a stochastic process. A potential solution to this issue would be to break down a non-stationary process into two or more stationary processes. This will be exemplified with the example provided, named rp4, where $rp4 \sim \mathcal{U}(-0.5, 0.5)$ from n = 1:500, and $rp4 \sim \mathcal{U}(0.5, 1, 5)$ for n = 501:1000), following the behaviour characterised in the question. A plot of a realisation has been provided in figure 12.

With rp4, we can estimate the signal as two stationary pdfs: one for the first 500 time instants, and one for the last 500 time instants. This will provide us with a reasonably accurate estimation of the stochastic process (would be more accurate if more realisations/samples were available). The estimates have been plotted in figure 13. In reality, figuring out the solution would be challenging. Given that the statistical properties are unknown before, all options would have to explored, which would be time consuming and resource intensive,

and a more complex solution may not be attainable. A method for modelling a non-stationary process is introduced and discussed in Section 4.2.

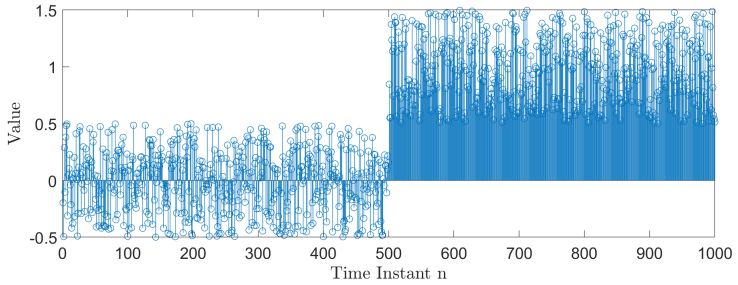

Figure 12: A realisation of rp4

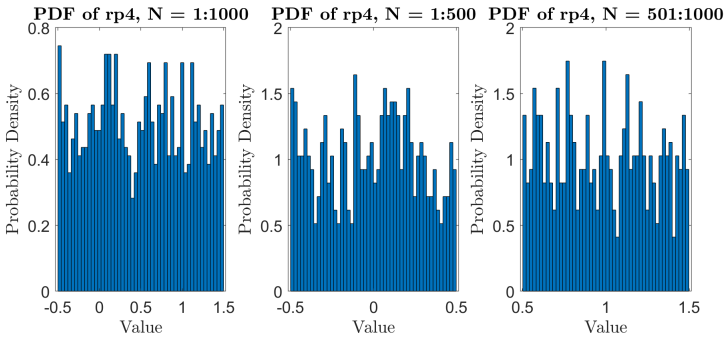

Figure 13: rp4 pdf estimations

Similar issues would arise if trying to accurately estimating rp1; the non-stationary pdf could estimated by modelling it as the combination of multiple pdfs. If multiple realisations are available, a pdf could be acquired for each time instant N. The issues mentioned before remain, and unlike before different realisations would be necessary to accurately estimate the pdf. Practically, this is not always an option.

2 Linear stochastic modelling

2.1 ACF of uncorrelated and correlated sequences

This section of the report will introduce the practical estimator of the autocorrelation function (ACF). The autocorrelation function for an ergodic stochastic process X_n is given by:

$$R_x(\tau) = \lim_{n\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} x[n]x[n+\tau], \quad \tau \in \mathbb{Z} \tag{9}$$

Given that in real-world applications, only a finite number of samples is available, we apply the unbiased estimate of the autocorrelation function, given by:

$$\hat{R}_x(\tau) = \frac{1}{N-|\tau|} \sum_{n=0}^{N-|\tau|-1} x[n]x[n+\tau], \quad \tau = -N+1, ..., N-1 \tag{10}$$

2.1.1 Unbiased Estimator of ACF - Gaussian Distribution

To explore the properties of the unbiased ACF, the matlab function `xcorr` will be used to generate an unbiased estimate for the ACF of a White Gaussian Noise (WGN) signal `x` (with zero mean and unit variance). The results are shown in figure 14.

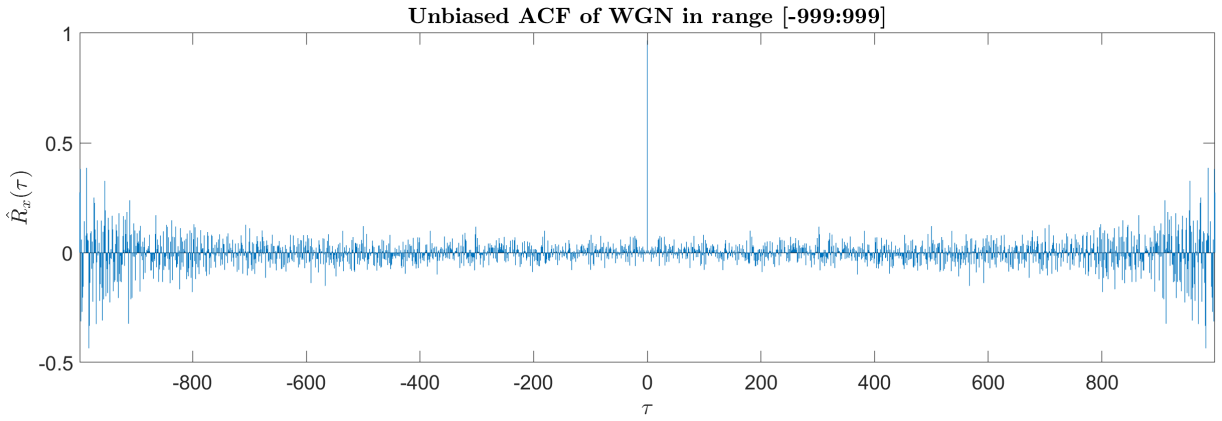

Figure 14: ACF of White Gaussian Noise

Theoretically, the ACF obtained from any real function should produce an even ACF (symmetric along the $\tau = 0$). This is seen within the generated unbiased estimate, which is an even function and which was produced with a purely real signal. Theoretically, a random white process should also have an autocorrelation function of zero at all values of τ with the exception of a distinct peak when $\tau = 0$, essentially forming a discrete Dirac function. A white random process is defined to have a fixed power spectral density (PSD) across all frequencies (hence the use of the word "white"). The relationship between the PSD and the ACF is defined as:

$$R_x(\tau) \overset{\mathcal{F}}{\Longrightarrow} P_x(f) \tag{11}$$

In turn, the ACF of a white process with unit variance should therefore be a Dirac function, ($\delta(t) \overset{\mathcal{F}}{\rightarrow} 1$). More intuitively, for every sample to be considered independent and random, no sample should have any correlation with any other value other than itself. With our estimate, there is a clearly defined peak at $\tau = 0$ (with a value very close to one), it can quite clearly be seen that the ACF function is not zero for non-zero values of τ, with the magnitude increasing with larger values of τ. This is due to the finite sample number, and will be explained in more detail in the following subsections.

2.1.2 Zooming into the generated ACF

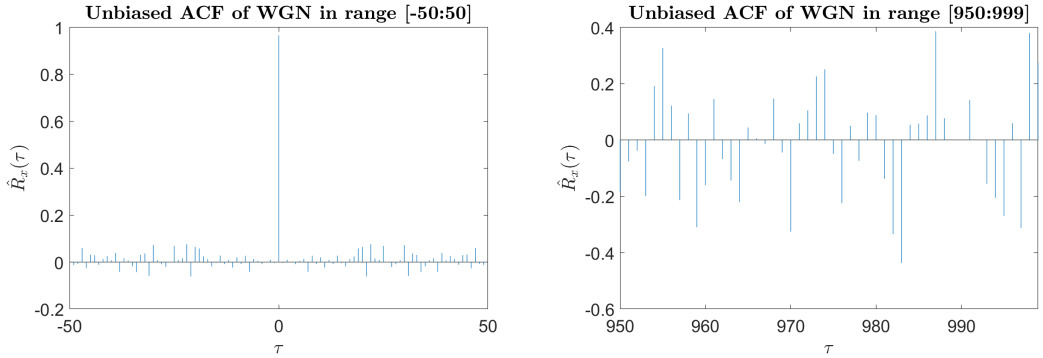

Figure 15: Zooming into ACF generated from WGN sample

Figure 15 displays a magnified version of the same unbiased ACF from section 2.1.1. When looking at the left graph (where $|\tau| < 50$), the unbiased ACF produces near ideal behaviour, with a very clear peak in the centre and small variations around it (values never exceed 0.1). For small τ, the unbiased ACF is a reliable estimator.

2.1.3 The Effects of a Large Autocorrerlation Lag

When looking at the right graph of figure 15, we can see that for large $\tau, \hat{R}_x(\tau) \neq 0$. This can be explained by the definition of the unbiased ACF, shown in equation 10. For larger τ, there are fewer iterations of the sum used to calculate the ACF. For example, for $\tau = 1$, there value of the ACF is summed over 998 instances. Howvever, when $\tau = 950$, the value of the ACF is computed with omly 49 instances. Additionally, the normalising term $\frac{1}{N-|\tau|}$ increases in magnitude as τ increases in value. These two factors couple together, and lead to significant error for larger τ. From figure 14, it seems as though the large error values become more frequent and severe from approximately N=300, and so a very conservative empirical bound for τ would be $|\tau| \leq \frac{N}{4}$.

2.1.4 ACF of a Moving Average Filter

A new realisation of the vector **x** was then passed through a MA filter of order 9. This was done using the MATLAB code `y=filter(ones(9,1),1,x)`, and the resultant ACF is shown in figure 16, along with the resulting ACFs of other MA filters with different orders.

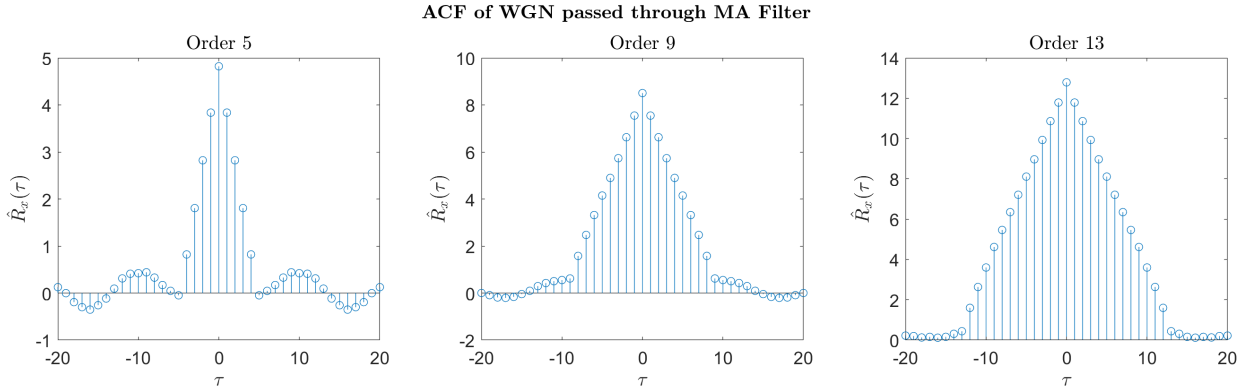

Figure 16: ACF of **x** passed through MA filter

Like before, the ACF is symmetric along $\tau = 0$. The ACF seen has a triangular shape centred around zero. From the other plotted filters (of order 5 and 13), we can see that the width and the height of the triangular function are proportional to the order (order 9 means a value of 9 at $\tau = 0$. We see a noticeable drop in the values of the ACF function after $|\tau| > N$, where N is the order of the filter. The FIRs of the coded MA filters are given by equation 12.

$$y[n] = \sum_{k=1}^{k=N} b_k \, x[n-k], \quad \text{where N is the order of the filter and all } b_k = 1 \tag{12}$$

The local sample mean can be obtained by simply changing all the coefficients equal to $\frac{1}{N}$ (rather than the unit coefficients that were initially applied). In doing this, the FIR response becomes $y[n] = \frac{1}{N} \sum_{k=1}^{k=N} x[n-k]$, which is equivalent to the local sample mean.

2.1.5 ACF of Filtered Stochastic Process

The ACF of Y_n, a filtered version of the process X_n, is given by the following:

$$R_Y(\tau) = R_X(\tau) * R_h(\tau) \tag{13}$$

where $R_x(\tau)$ is the ACF of X_n and $R_h(\tau)$ is the ACF of the impulse response. Given that X_n is an uncorrelated process, $R_x(\tau) = \delta(\tau)$.

$$\therefore R_Y(\tau) = \delta(\tau) * R_h(\tau) = R_h(\tau) \quad \textit{when } X_n \textit{ is uncorrelated}$$

2.2 Cross-correlation function

2.2.1 CCF Estimation

The practical unbiased estimate of the cross-correlation function (CCF) for an ergodic function is given by:

$$\hat{R}_{XY}(\tau) = \frac{1}{N-|\tau|} \sum_{n=0}^{N-|\tau|-1} x[n]y[n+\tau], \qquad \tau = -N+1, ..., N-1 \tag{14}$$

Using the commmand `xcorr` once again, the CCF for WGN signal **x** before the MA filter and after the MA filter was calculated. The CCF for varying orders N are shown in figure 17.

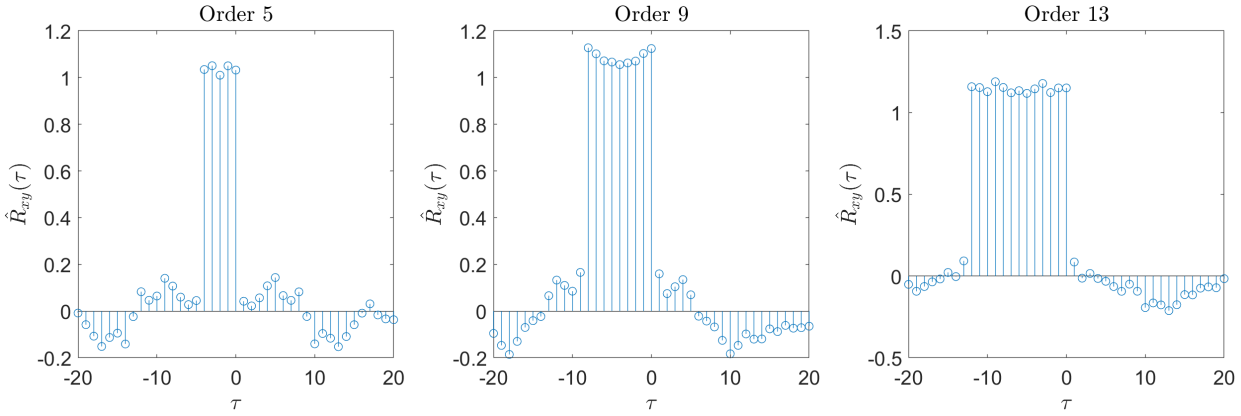

Figure 17: CCF of WGN signal **x** and filtered WGN signal **y** of varying filter orders

The CCFs resemble a rectangular function between a range of -N+1 to 0, where N is the filter order with a magnitude of approximately 1. This means that **y** at time n is equally correlated to the current value of **x**, as well as the 8 previous values of **x** This is in line with what is expected. Non-zero values in the unbiased CCF beyond this point are due to the finite nature of **x**. A more theoretical approach to understanding the CCF output will now be discussed. Given that X_n is an uncorrelated process we can show the following:

$$R_{XY}(\tau) = R_X(\tau) * h(\tau) = \delta(\tau) * h(\tau) = h(\tau)$$

From this, we can conclude that the CCF between the input and output of a filter provides the impulse response of the filter itself. This provides a mathematical justification for the rectangular functions seen in Figure 17.

2.2.2 System Identification

The result obtained in the previous section showed that taking the CCF of before and after a system provides the impulse response of the system, provided that the input is an uncorrelated process. With an ideal input, the order of the filter will be equivalent to the non-zero values present in the CCF. Given that the impulse response determines the behaviour of a system, this is a very effective method of system identification.

2.3 Autoregressive Modelling

2.3.1 AR(2) Processes and Process Stability

For this section, 100 samples of the uniformly distributed variables $a_1 \in [-2.5\ ,\ 2.5]$ and $a_2 \in [-1.5\ ,\ 1.5]$. Pairs were made from these and used as the coefficients of an AR(2) process described by equation 15.

$$x[n] = a_1\ x[n-1] + a_2\ x[n-2] + w[n], \quad w[n] \sim \mathcal{N}(0,1) \tag{15}$$

To determine the stability region of the AR(2) process,the Z transform of equation 15 must be can be taken and then rearranged to get:

$$H(z) = \frac{X(z)}{W(z)} = \frac{1}{1 - a_1\ z^{-1} - a_2\ z^{-2}} \tag{16}$$

An AR(2) process is considered stable so long as the poles remain within the unit circle. This means that the stability region can be obtained by solving its characteristic equation and ensuring that the roots remain within the unit circle:

$$C(z) = z^2 - a_1\ z - a_2 \implies z = \frac{a_1 \pm \sqrt{a_1^2 + 4a_2}}{2} \tag{17}$$

Solving for the positive root provides the stability inequality $a_1 + a_2 < 1$, and solving for the negative root provides the stability inequality $a_2 - a_1 < 1$. The last inequality is derived by rewriting the characteristic equation as:

$$C(z) = (z - \lambda_1)(z - \lambda_2) = z^2 - (\lambda_1 + \lambda_2)z + \lambda_1\lambda_2 \implies a_1 = \lambda_1 + \lambda_2\ ,\ a_2 = \lambda_1\lambda_2$$

Because both roots are less than one in magnitude, $|a_2| < 1$. These three inequalities form the stability triangle which can be seen in the plot from figure 18 indicating the stable region. To better illustrate the result, the task was repeated for N = 10000.

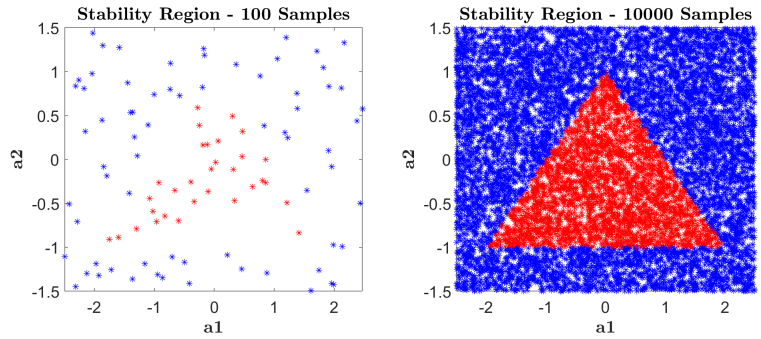

Figure 18: Stability Triangle for a1 and a2

2.3.2 ACF of Sunspot Series

This subsection will examine real-world data in the form of the sunspot data series. The dataset catalogues the number of sunspots seen over a period of nearly 300 years. The ACF of the data for both the original dataset and the zero-mean data set are plotted in figure 19.

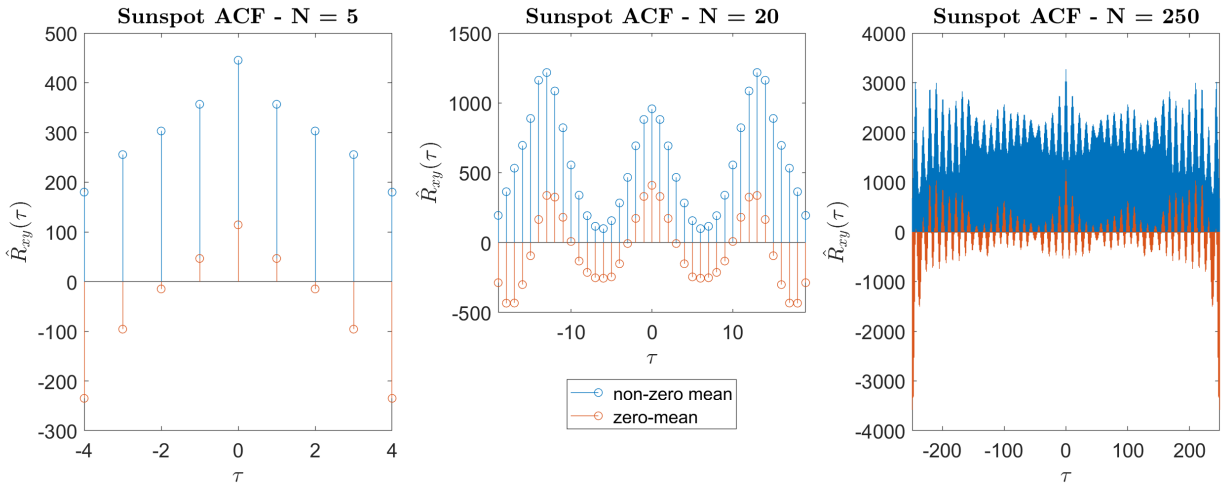

Figure 19: ACF for original sunspot data and zero-mean sunspot data

It can be immediately seen that there are noticeable differences between the original ACFs and the zero-mean ACFs. When the dataset mean $\neq 0$, the mean behaves like a DC component and biases the ACF terms. As a result, all ACF values from the original sunspot data are positive. The ACFs derived from the zero-mean data have their largest values at $\tau = 0$ which, as discussed previously, is consistent with the theoretical output. This not the case for the non-zero mean ACFs (due to the bias term). For N = 5, no repeated pattern is observed for either the original ACF or the zero-mean ACF. This is due to the limited sample size. For N=20, the both ACFs exhibits an oscillatory behaviour, with a period of 13 years. This implies that sunspot occurrence follows a 13 year cycle. For N = 250, the oscillations is made even more explicit, now with a periodicity of 11 years for small τ (large τ has not been considered due to the error in the unbiased estimator discussed previously). Note that the oscillatory behaviour is made more explicit when looking at the zero-mean ACF rather than the original ACF for N = 20 and 250.

2.3.3 PCF and Model Order

In this section, the partial autocorrelation function (PACF or PCF) was obtained (until the model order 10) using the Yule Walker equations do determine the "most likely" model order. The PCFs for both the original data and the normalised data were computed using the MATLAB function `aryule`. The results are plotted in figure 20. The most likely AR order will be determined based off of the normalised data (due to the removal of the bias term, which is what causes the variations between the original and normalised PCFs). It can be seen that the PCF amplitude noticeably drops from $\tau = 3$ onwards. As a result, the most likely AR model order is estimated to be 2.

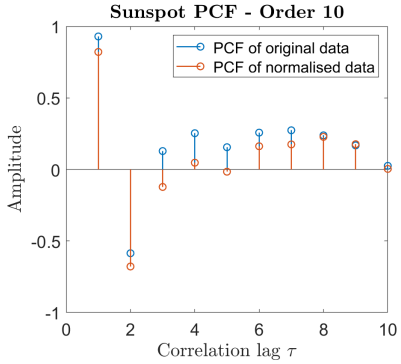

Figure 20: PCF of Sunspot Data

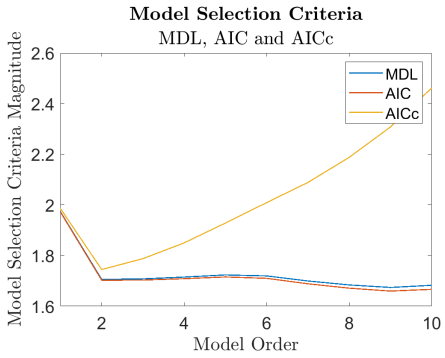

Figure 21: Model Order Selection Criteria

2.3.4 Model Order Selection

The correct model order will now be calculated using the minimum description length (MDL) and Akaike information criterion (AIC) and the corrected AIC (AICc). The cumulative square error was calculated by using the model coefficients to generate a predicted signal (stored as a vector on MATLAB). This was then compared to the original signal (also stored as a vector). The difference between the two vectors were obtained, and resultant vector's inner product was taken. Note that in other sections where the MSE was used, this number was then divided by N. The error was then used to obtain results for the selection criteria. These have been plotted in figure 21. The results for MDL and AIC indicate that $p = 2$ and $p = 9$ would be appropriate model order choices (local minima present at these values). The results for the AICc, which aims to reduce the probability of overfitting and penalises model complexity, indicate that $p = 2$ is the optimal model order choice. Given this, it is fair to state that $p = 2$ it is the optimal order choice.

2.3.5 AR Model Predictions

Figure 22 illustrates the predicted output of for the AR(1), AR(2) and AR(10) with differing prediction horizons for the normalised sunspot data. In general, the models perform better with a lower prediction horizon. For M=1, all three models perform quite well, estimating the true output with good accuracy. As the prediction horizon begins to increase, AR(1) and AR(2) begin to struggle, being unable to capture a sufficient degree of information to accurately predict ahead by 5 or 10 samples. Even the AR(10) model begins to struggle from M=5 onwards, but performs better than the other two. The models also begin to

showcase a 'lagging' behaviour when compared to the true output when the prediction horizon increases. All in all, the AR(10) model seemingly performs the best. This is unsurprising, given that the model was trained to best estimate this data. It is worth mentioning that the AR(10) model is more likely to overfit to unseen data, due to its increased complexity. This is factored into the AICc model selection criteria.

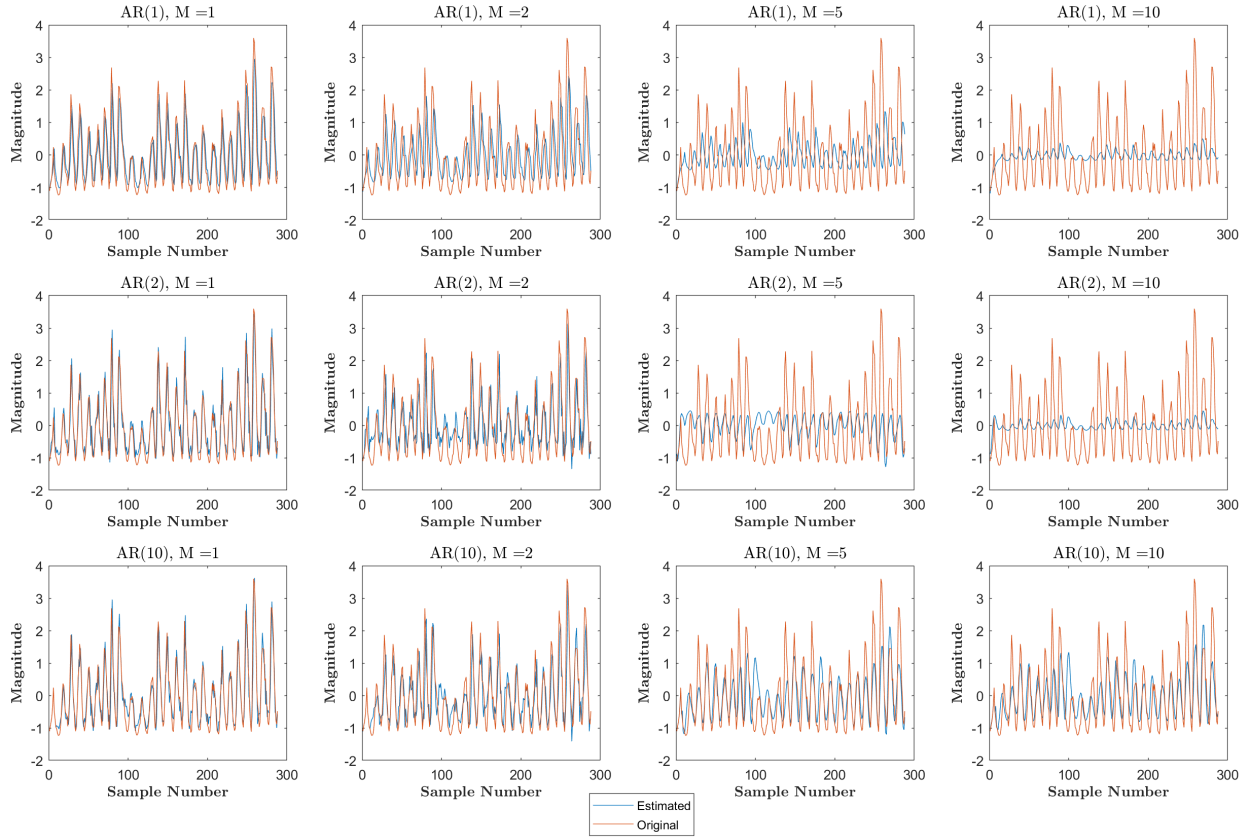

Figure 22: Time series plots for AR(1), AR(2) and AR(10) Models for varying prediction horizons

2.4 Cramer-Rao Lower Bound

The Cramer-Rao lower bound (CRLB) determines a lower bound on the variance of any unbiased estimator, and both allows us to establish whether an estimator is the minimum variance unbiased (MVU) estimator and provides a benchmark against which to compare the performance of any unbiased estimator. Consider an input signal $\mathbf{x} \in \mathbb{R}^n$ and a vector of unknown parameters $\boldsymbol{\theta}$ which we wish to estimate. The likelihood function is therefore denoted by $p(\mathbf{x}; \boldsymbol{\theta}$, , and for an unbiased estimator, the Cramer-Rao lower bound (CRLB) is given by:

$$var(\hat{\theta}_i) \geq [\mathbf{I}^{-1}(\boldsymbol{\theta})]_{ii}, \quad where \quad [\mathbf{I}(\boldsymbol{\theta})]_{ij} = -\mathbb{E}\left\{\frac{\partial^2 p(\mathbf{x};\boldsymbol{\theta})}{\partial\theta_i \partial\theta_j}\right\} \tag{18}$$

2.4.1 AR(1) Model for NASDAQ Financial Index

From figure 23, it can be seen via both the PCF results and the model order selection criteria and the PCF of the data that an order of 1 is the optimal choice to capture the daily returns from the NASDAQ data. After p = 1, all the selection criteria increase in magnitude, and the AICc increases considerably due to the complexity penalty.

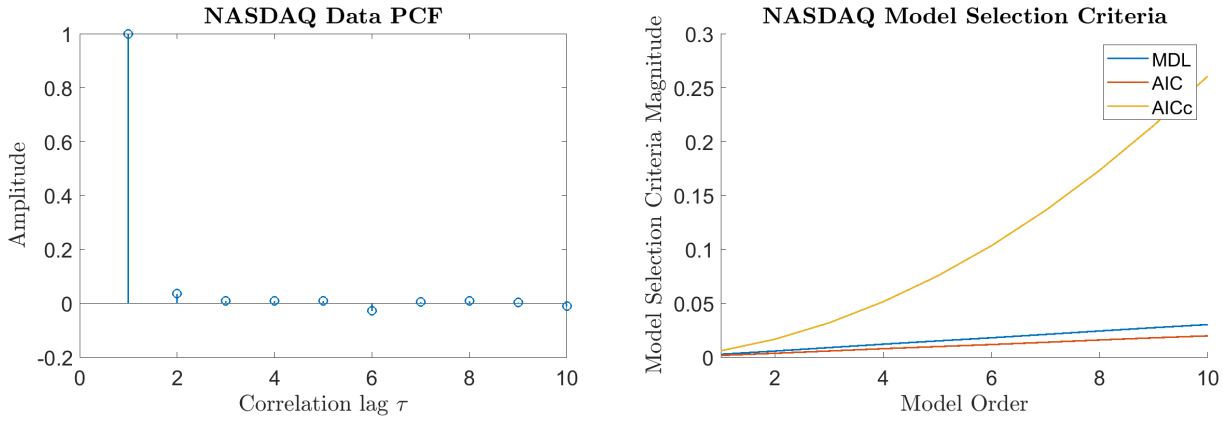

Figure 23: PCF and Model Order Selection Criteria for NASDAQ Daily Returns

2.4.2 The Fisher Information Matrix

The natural logarithm of an AR model's PSD is given by are given by equation 19. Here, $\hat{\sigma}^2$ is the estimated value of the driving noise variance.

$$\ln[\hat{P}_X(f;\boldsymbol{\theta})] = \ln[\hat{\sigma}^2] - \ln\left[1 - \sum_{m=1}^{p} \hat{a}_m e^{-j2\pi fm}\right] - \ln\left[1 - \sum_{m=1}^{p} \hat{a}_m e^{j2\pi fm}\right] \tag{19}$$

The standard log likelihood function is now be replaced with $\ln[\hat{P}_X(f;\boldsymbol{\theta})]$. From this, the elements of the Fisher information matrix are approximately given by equation xx.

$$[\mathbf{I}(\boldsymbol{\theta})]_{ij} = \frac{N}{2}\int_{\frac{-1}{2}}^{\frac{1}{2}} \frac{\partial\, ln[P_x(f;\boldsymbol{\theta})]}{\partial\theta_i}\frac{\partial\, ln[P_x(f;\boldsymbol{\theta})]}{\partial\theta_j} df \tag{20}$$

This approximation is considered to be quite accurate, even with a short data length. Given that $p = 1$ such that $\boldsymbol{\theta} = [a_1,\ \sigma^2]$, the Fisher information matrix is of dimension 2×2. We can then compute $\frac{\partial ln[P_x(f;\boldsymbol{\theta})]}{\partial\theta_2} = \frac{\partial\, ln[P_x(f;\boldsymbol{\theta})]}{\partial\sigma^2} = \frac{1}{\sigma^2}$. From this we can compute $[\mathbf{I}(\boldsymbol{\theta})]_{22}$:

$$[\mathbf{I}(\boldsymbol{\theta})]_{22} = \frac{N}{2}\int_{\frac{-1}{2}}^{\frac{1}{2}} \frac{\partial\, ln[P_x(f;\boldsymbol{\theta})]}{\partial\theta_2}\frac{\partial\, ln[P_x(f;\boldsymbol{\theta})]}{\partial\theta_2} df = \frac{N}{2}\int_{\frac{-1}{2}}^{\frac{1}{2}} \frac{1}{\sigma^2}\,\frac{1}{\sigma^2} = \frac{N}{2\sigma^4} \tag{21}$$

Given that $[\mathbf{I}(\boldsymbol{\theta})]_{12} = [\mathbf{I}(\boldsymbol{\theta})]_{21} = 0$, and $[\mathbf{I}(\boldsymbol{\theta})]_{11} = \frac{N\, r_{xx}(0)}{\sigma^2}$, the entire Fisher information matrix can be written out as:

$$\mathbf{I}(\boldsymbol{\theta}) = \begin{bmatrix} \frac{N\, r_{xx}(0)}{\sigma^2} & 0 \\ 0 & \frac{N}{2\sigma^4} \end{bmatrix} \tag{22}$$

2.4.3 Lower Bound of Parameter Variances

$$\mathbf{I}^{-1}(\boldsymbol{\theta}) = \frac{\sigma^6}{N^2\, r_{xx}(0)}\begin{bmatrix} \frac{N}{2\sigma^4} & 0 \\ 0 & \frac{N r_{xx}(0)}{\sigma^2} \end{bmatrix} = \begin{bmatrix} \frac{\sigma^2}{N r_{xx}(0)} & 0 \\ 0 & \frac{2\sigma^4}{N} \end{bmatrix} \tag{23}$$

Using the result above and the definition in equation 18 (where $\hat{\boldsymbol{\theta}} = [\hat{a}_1,\ \hat{\sigma}^2]$, we can deduce the minimum variance for our two estimators as:

$$var(\hat{a}_1) \geq \frac{\sigma^2}{N r_{xx}(0)}, \qquad var(\hat{\sigma}^2) \geq \frac{2\sigma^4}{N} \tag{24}$$

$$\text{Applying the result } r_{xx}(0) = \frac{\sigma^2}{1-a_1^2}, \quad var(\hat{a}_1) \geq \frac{1}{N}(1-a_1^2) = \frac{1}{924}(1 - 0.9989^2) = 2.3796 \times 10^{-6} \tag{25}$$

2.4.4 Parameter Heatmaps and their Behaviour Approaching Unity

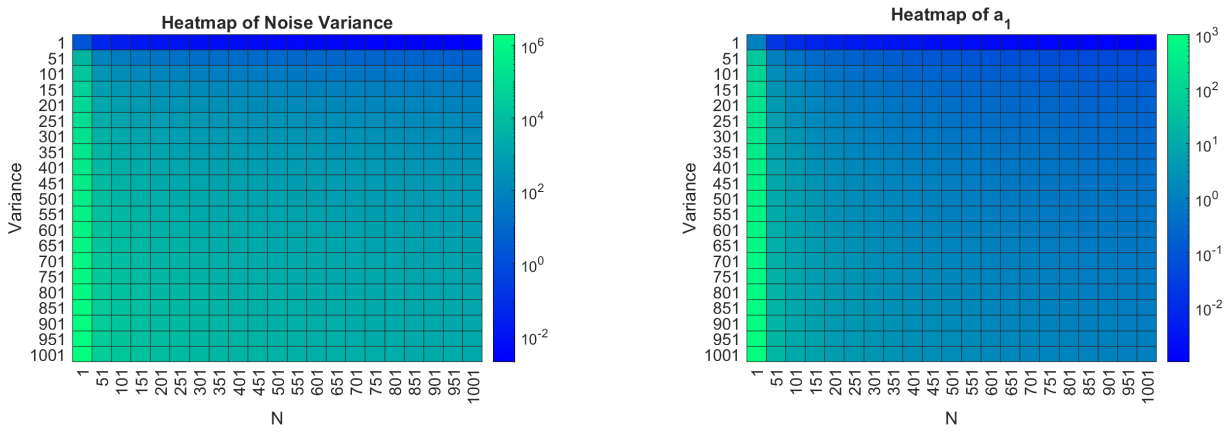

Figure 24: CRLB for estimators visualised with a heatmap

Figure 24 indicates that as a_1 approaches unity, the lower bound for the estimator variance decreases. As a_1 approaches zero, the variance lower bound increases. This is what is predicted by equation 25.

2.4.5 Computing Bound in terms of $A(f)$

By taking the PSD of an AR signal and setting p = 1, and then once again using $\boldsymbol{\theta} = [a_1,\ \sigma^2]$:

$$\hat{P}_X(f;\boldsymbol{\theta}) = \frac{\hat{\sigma}^2}{|1 - \sum_{m=1}^{p} \hat{a}_m e^{-j2\pi f m}|^2} = \frac{\hat{\sigma}^2}{|1 - \hat{a}_1 e^{-j2\pi f}|^2} = \frac{1}{|A(f)|^2}$$

$$\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \begin{bmatrix} \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial a_1} \\ \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \sigma^2} \end{bmatrix}$$

$$\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial a_1} = \sigma^2(1 - a_1 e^{-j2\pi f 1})^{-3}(-e^{-j2\pi f})(-2) = \frac{2\sigma^2(e^{-j2\pi f})}{|A(f)|^3}$$

$$\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \sigma^2} = \frac{1}{|A(f)|^2}$$

$$var(\hat{P}_X(f;\boldsymbol{\theta})) \geq \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}^T \mathbf{I}^{-1}(\boldsymbol{\theta}) \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$$

$$var(\hat{P}_X(f;\boldsymbol{\theta})) \geq \begin{bmatrix} \frac{2\sigma^2(e^{-j2\pi f})}{|A(f)|^3} & \frac{1}{|A(f)|^2} \end{bmatrix} \begin{bmatrix} \frac{1}{N}(1 - a_1)^2 & 0 \\ 0 & \frac{2\sigma^4}{N} \end{bmatrix} \begin{bmatrix} \frac{2\sigma^2(e^{-j2\pi f})}{|A(f)|^3} \\ \\ \frac{1}{|A(f)|^2} \end{bmatrix}$$

$$\implies var(\hat{P}_X(f;\boldsymbol{\theta})) \geq \frac{2\sigma^4}{N} \left[\frac{2(1 - a_1^2)e^{-j4\pi f}}{|A(f)|^6} + \frac{1}{|A(f)|^4} \right]$$

2.5 Real World Signal: ECG from iAmp experiment

2.5.1 Probability density estimate (PDE)

This section uses ECG data provided from the iAmp experiment. We must first assume that the data can be considered as a stationary signal, meaning that it can be modelled with a random variable with a given probability distribution. To obtain a smoother estimate of the heart rate, a moving average filter is applied

to the signal, with its FIR highlighted in 26. The estimated probability density functions (PDEs) for the original heart rate data, as well as the filtered heart rates for $\alpha = 1$ and $\alpha = 0.6$ are plotted in figure 25.

$$\hat{h}[k] = \frac{1}{10}\sum_{i=1}^{10} \alpha h[i] \tag{26}$$

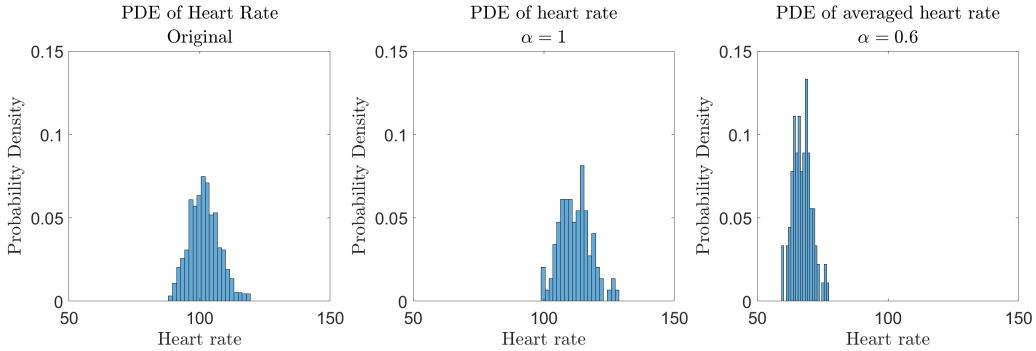

Figure 25: RRI Heart Rates

2.5.2 Comments

When averaging the heart rate signal we observe two significant differences: a reduction in variance, and a shift in mean with values of alpha other than 1We can see that the averaged signals have a different variance when compared to the original heart rate. The variances for $\alpha = 1$ and $\alpha = 0.6$ both decrease. This is unsurprising, due to the effect of scaling, where $var(ax) = a^2\, var(x)$. This is especially pronounced when $\alpha < 1$.

2.5.3 RRI Data Autocorrelation

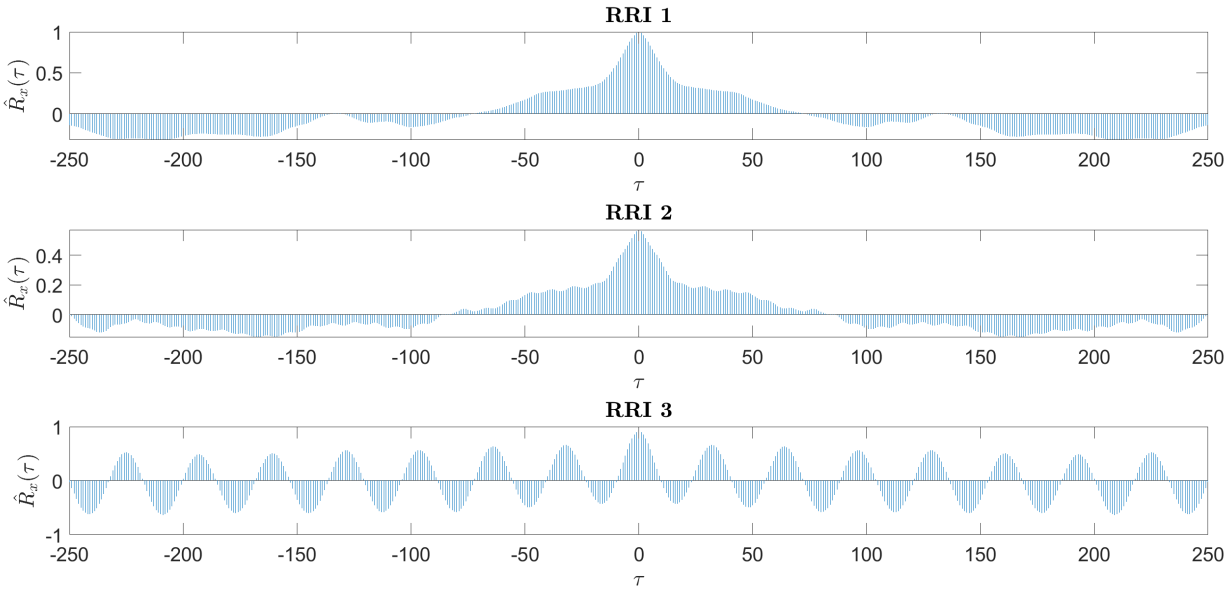

Figure 26: RRI Heart Rates

Figure 26 displays the autocorrelation sequence for the three RRI trials. The MATLAB function `detrend` was applied to ensure the data was zero-meaned. The plots have only been plotted $\pm\frac{N}{4}$ due to the inaccuracies present within the unbiased ACF. From these ACFs, we can conclude that the the heart rate can be modelled as an AR process. This is because of the clear sinusoidal nature present in all three ACF in figure 26, a common characteristic of many AR process.

2.5.4 Optimal AR Modelling of Heart Rate

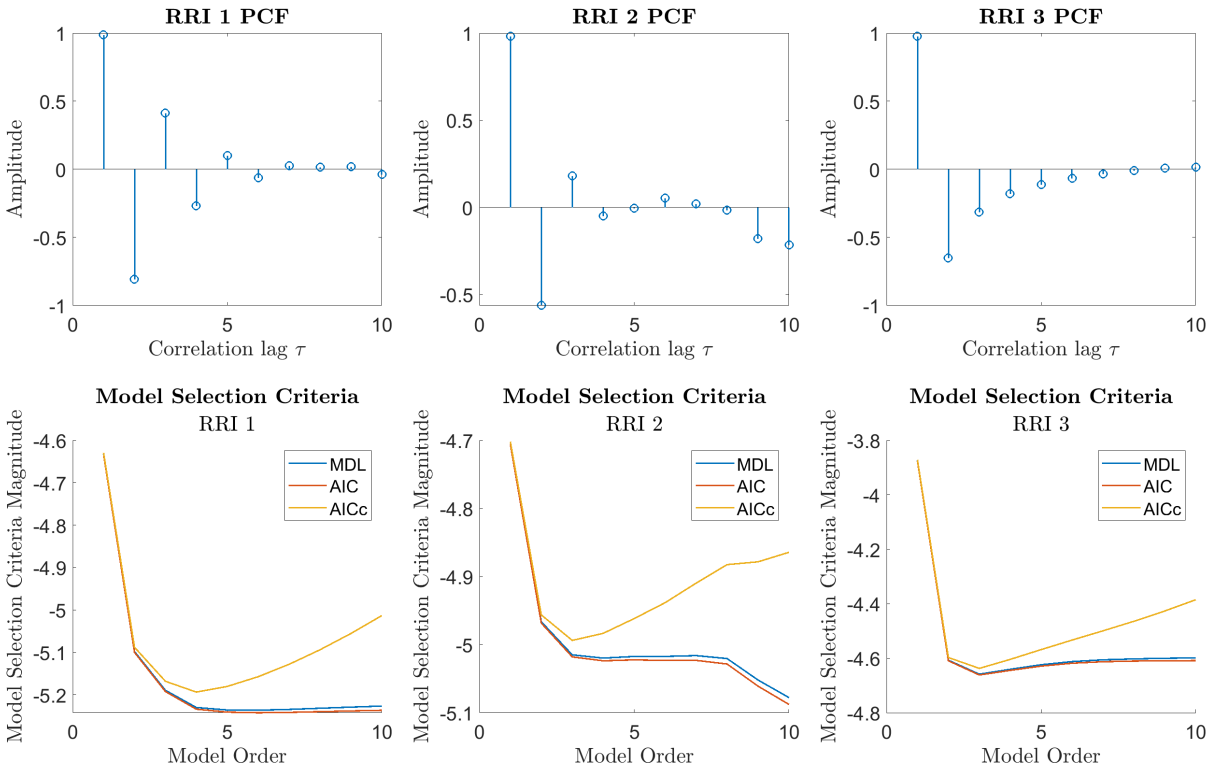

Figure 27: RRI Heart Rates

The MSE was used for this task to obtain the MDL, AIC and AICc. The results for trial 1 indicate that the ideal model order is 4. This is because of the common minimum seen for the MDL, AIC and AICc. This is also seen in the PCF results, where the amplitude at $\tau = 5$ is noticeably lower than at $\tau = 4$. The same justification is applied to the second and third trials. The ideal model order was obtained to be 3 for trial 2, and 3 for trial 3.

3 Spectral Estimation and Modelling

This section of the report will focus on the practical estimator of the Power Spectral Density (PSD). The PSD was defined earlier in equation 11 as the fourier transform of the autocorrelation function. A practical estimator of the PSD, also known as a periodogram, is defined with equation 27.

$$\hat{P}_X(f) = \frac{1}{N}\left|\sum_{n=0}^{N-1} x[n]e^{-j2\pi f\frac{n}{N}}\right|^2, \quad f \in (0,1) \tag{27}$$

```
function  [Px_f , f] = pgm(x)
    N = length(x);
    x_fft = fft(x);
    Px_f = (1/N)*(abs(x_fft).^2);
    f = (0:N-1)'/N;
end
```

Figure 28: `pdf` function code

The function `pgm` was written on MATLAB to generate a periodogram, shown in figure 28. To verify the code, the estimator was tested with N-sample realisations of WGN $\mathbf{x}$, for N = 128, 256 and 512. The results are shown in figure 29. The results show a function symmetric around $f = 0.5$, indicating a real input. The theoretical PSD of WGN generated should be equal to one at all frequencies (this was explained in Section 2.1). The obtained periodogram is far from this, displaying a high degree of error. There seems to be little to no correlation between the number of data points and the accuracy of the estimator. Whilst every periodogram has a mean of approximately one, the variance is significant (also having a value of approximately one), when ideally it should be zero. The variance does not reduce in magnitude when increasing N (or at least does not improve within the range of N that we have looked at). This error is likely caused the finite number of samples.

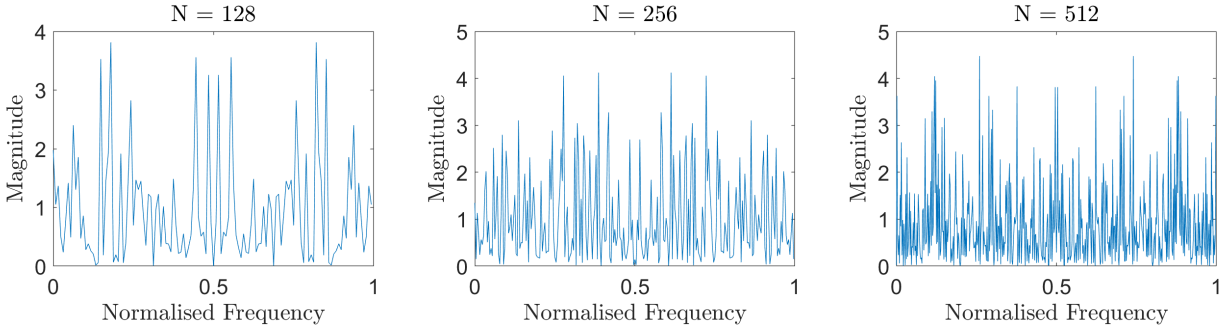

Figure 29: Periodograms for $\mathbf{x}$ of varying sizes N

3.1 Averaged Periodogram Estimates

3.1.1 Applying a Zero-Phase Filter

A zero-phase FIR filter with impulse response $0.2 \times [1\ 1\ 1\ 1\ 1]$ is applied to smooth the periodograms computed in the previous exercise. The results are shown in figure 30. The filtered periodograms are a marked improvement on the previous results. The mean of every periodogram is still approximately one, but the variance has reduced considerably in size (value of approximately 0.1 but unsurprisingly varies depending on the realisation of $\mathbf{x}$).

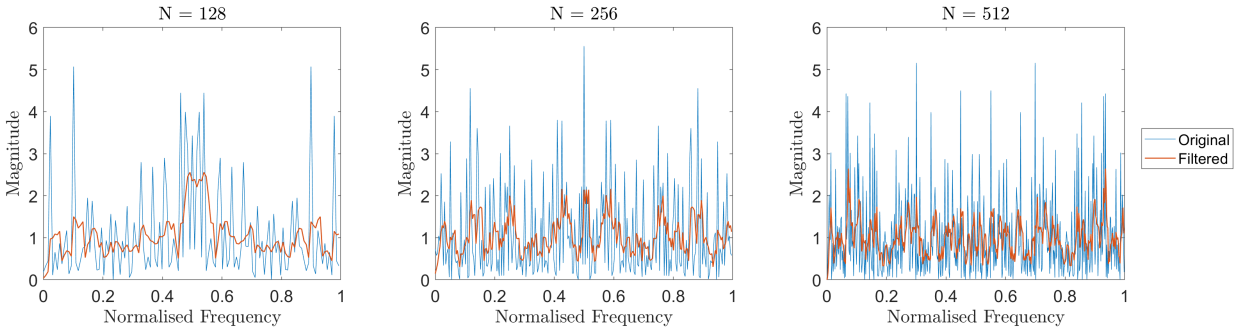

Figure 30: Filtered PSD

3.1.2 Segment Analysis of WGN

Figure 31 displays the estimated PSDs of eight equally sized, non-overlapping segments of a WGN signal of length 1024. Similar conclusions can be made as earlier: the variances of the periodograms are high, and each is on the whole a poor estimate of the PSD. Once again, the estimates are improved by the filter. It is worth noting that the peak amplitudes remain fairly consistent in magnitude from segment to segment, but occur at different frequencies. This will be taken advantage of in the following subsection.

3.1.3 Segment Averaging

Figure 32 displays the averaged periodogram formed by averaging the eight unfiltered periodograms obtained in the previous section. This yields a better result, with the mean equal to approximately 1, and the variance once again dropping to around 0.1. The filter could be applied here to further improve the estimate of WGN. It should be noted that both the filter and the averaging method reduce the applicability of our estimator; As observed, averaging results in less significant variations. However, if these variations are a result of the true PSD rather than the estimator itself, we may lose critical information about the true PSD.

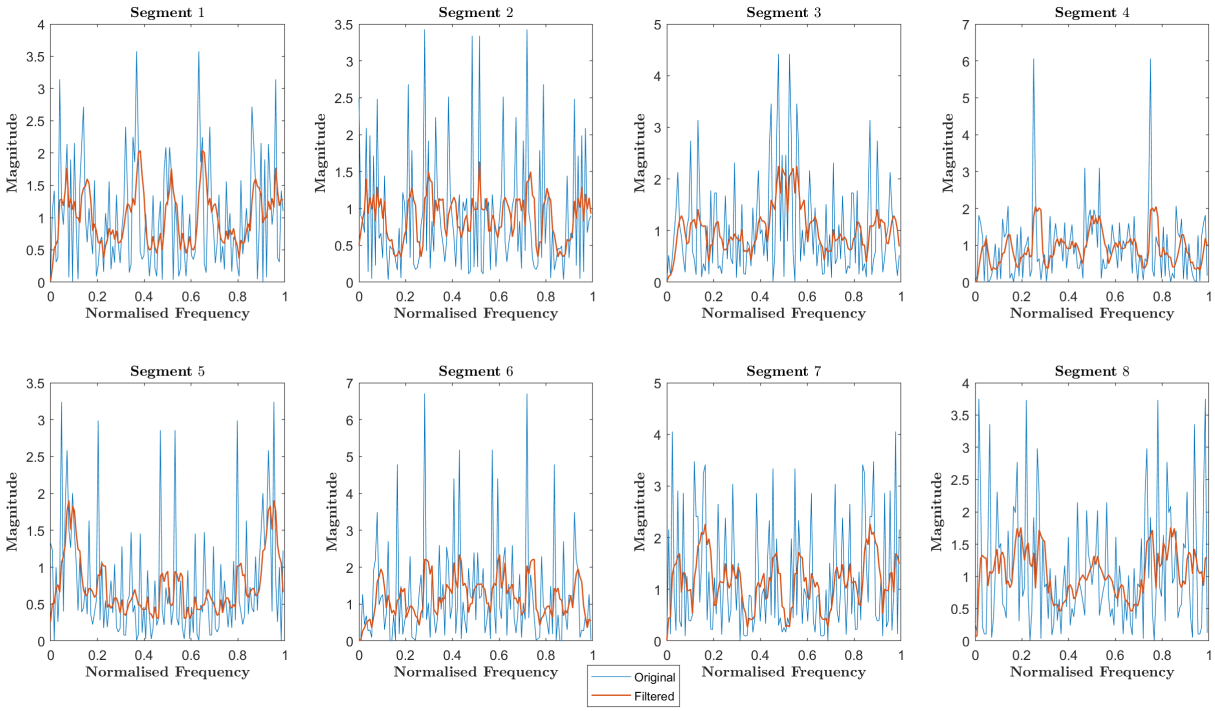

Figure 31: Periodograms of uniformly segmented WGN Signal of length 1024

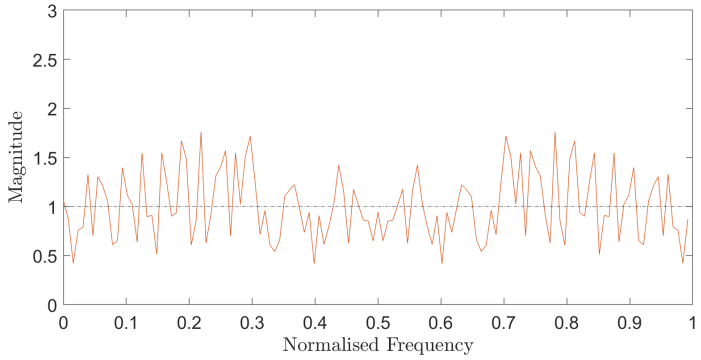

Figure 32: Averaged periodogram across eight segments

3.2 Spectrum of Autoregressive Processes

For this section, a filter with coefficients b = 1 and a = [1, 0.9] was applied to a 1064-sample WGN sequence **x** using the MATLAB function `filter`. The first 40 values of the resulting signal, **y** were then removed as these were affected by the transient effects of the filter. The theoretical PSD of the filtered signal **y** is given by equation 28.

$$P_Y(f) = \frac{1}{|1 + 0.9e^{-j\ 2\pi f}|^2} \tag{28}$$

3.2.1 Theoretical PSD

The theoretical PSD, given by equation 28, is plotted in figure 33 . The theoretical PSD indicates a normalised cut-off frequency of 0.3 Hz and the peak value of magnitude approximately 100 occurs at 0.5Hz. The PSD is focused on the higher end of the frequency spectrum, implying that the filter is a high-pass filter.

3.2.2 Adding the Periodogram

The periodogram has been plotted alongside the theoretical PSD. Like the previous section, the estimate seems to roughly follow the general shape, but has significant fluctuations. All in all, this periodogram can be considered inaccurate. Like before, this is down to the finite sample length.

3.2.3 Rectangular Windowing

When using a finite number of samples (as we do) within the time domain, its mathematical representation is equivalent to multiplying a signal of infinite length by a rectangular window. The corresponding effect within the frequency domain is equivalent to convoluting the PSD by a sinc function. This leads to the deviations between the estimate and the theoretical PSD that are made more clear when zooming in.

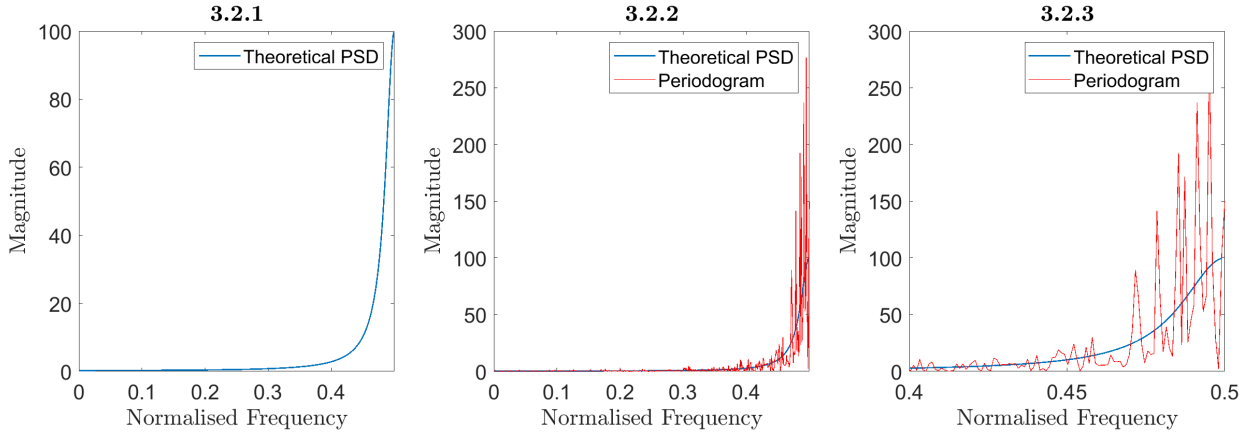

Figure 33: Theoretical PSD for given filter described by equation 28, along with periodogram estimate

3.2.4 Model Based PSD Estimate

Instead of approximating the PSD using the samples directly via the periodogram, we can assume that the sequence was generated by an AR model. Here, we require two parameters, which are estimated via the the unbiased ACF of **y**. The estimated model is given by equation 29.

$$\hat{P}_y(f) = \frac{\hat{\sigma}_X^2}{|1 + \hat{a}_1 e^{-j\ 2\pi f}|^2}, \quad \text{where } \hat{a}_1 = -\frac{\hat{R}_y(1)}{\hat{R}_y(0)}, \text{ and } \hat{\sigma}_X^2 = \hat{R}_y(0) + \hat{a}_1 \hat{R}_y(1) \tag{29}$$

Ideally, $a_1 = 0.9$ and $\sigma_X = 1$. We estimate these parameters as $\hat{a}_1 = 0.8789$, and $\hat{\sigma}_X^2 = 1.0524$. Using this, the model PSD has been plotted alongside the theoretical PSD and the periodogram in figure 34. It can

be clearly seen that the model-based PSD is much more accurate than the periodogram. The accuracy of the model based PSD is entirely dependent on the accuracy of the estimates of a_1 and σ_X. As a result, the length of $\mathbf{y}$ has a significant impact on the accuracy of the model-based PSD; the longer $\mathbf{y}$ is, the more accurate the model based PSD will be.

3.2.5 Sunspot Periodogram and Model-Based PSDs

Figure 34 displays the periodogram and the model-based PSDs for both the original sunspot data and the normalised equivalent. The most immediate difference between the two is clearly the dominant DC component present in the original data. This is unsurprising, due to the bias present within the data set. When setting the mean to zero, we see a considerably different, and more useful, result. We can see that AR10 captures the most similar result to the periodogram; this could be attributed to overfitting to the model. AR1, which can be considered as undermodelled, fails to effectively capture the larger peak. AR2 more effectively captures the larger peak, but struggles to capture the lower frequency peaks. However, for reasons discussed in previous sections, AR2 captures sufficient information to be considered a good model, it seems as though effectively acquiring the second larger peak is more important to get an effective model.

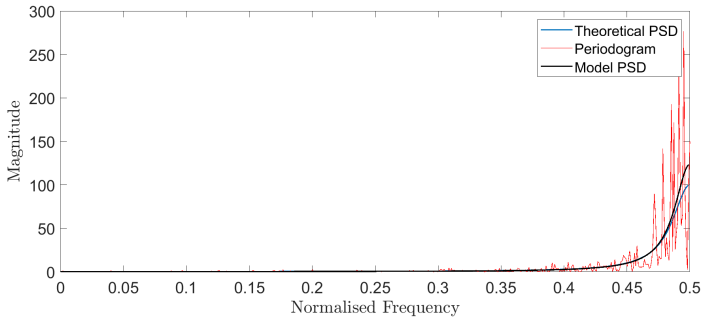

Figure 34: Model Based PSD for filter described by equation 28

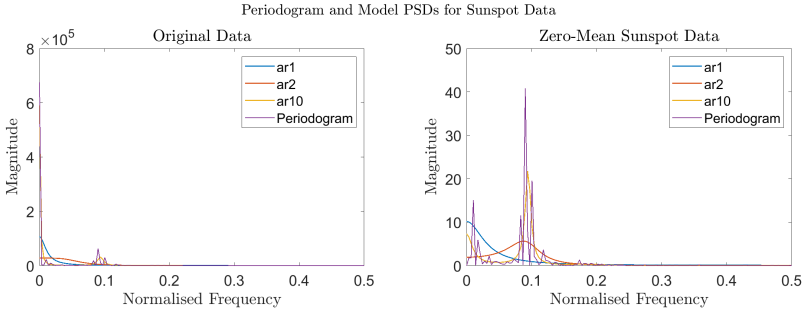

Figure 35: Periodogram for sunspot data and the model-based PSDs

3.3 The Least Squares Estimation (LSE) of AR Coefficients

3.3.1 Least Squares Estimate and its Relationship to Yule-Walker

The **biased** autocorrelation function is given by equation 30.

$$\hat{r}_{xx}[k] = \frac{1}{N} \sum_{n=0}^{N-1-k} x[n]x[n+k] \tag{30}$$

For an AR(p) process, equation 30 can be rewritten as:

$$\hat{r}_{xx}[k] = \sum_{i=1}^{p} a_i \hat{r}_{xx}[k-i] + \epsilon[k], \quad i \geq 1 \tag{31}$$

Next, the Least Squares (LS) cost function J , used to determine all the AR coefficients, is defined as:

$$J = \sum_{k=1}^{M} \left[\hat{r}_{xx}[k] - \left(\sum_{i=1}^{p} a[i] \hat{r}_{xx}[k-i] \right) \right]^2 , \tag{32}$$

$$J \text{ in matrix form:} \quad J = (\mathbf{x} - \mathbf{Ha})^T (\mathbf{x} - \mathbf{Ha}) = ||\mathbf{x} - \mathbf{Ha}||^2 \tag{33}$$

$$\text{where } \mathbf{x} = \begin{bmatrix} \hat{r}_{xx}[1] \\ \hat{r}_{xx}[2] \\ \vdots \\ \hat{r}_{xx}[M] \end{bmatrix}, \qquad \mathbf{a} = \begin{bmatrix} a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix}, \qquad \mathbf{H} = \begin{bmatrix} \hat{r}_{xx}[0] & \hat{r}_{xx}[-1] & \dots & \hat{r}_{xx}[1-p] \\ \hat{r}_{xx}[1] & \hat{r}_{xx}[0] & \dots & \hat{r}_{xx}[2-p] \\ \vdots & \vdots & \ddots & \vdots \\ \hat{r}_{xx}[M-1] & \hat{r}_{xx}[M-2] & \dots & \hat{r}_{xx}[M-p] \end{bmatrix} \tag{34}$$

$$\text{The estimated signal is: } \mathbf{S} = \mathbf{Ha} = \begin{bmatrix} \sum_{i=1}^{p} a[i] \hat{r}_{xx}[1-i] \\ \sum_{i=1}^{p} a[i] \hat{r}_{xx}[2-i] \\ \vdots \\ \sum_{i=1}^{p} a[i] \hat{r}_{xx}[M-i] \end{bmatrix} \tag{35}$$

To find $\arg\min_{\hat{\mathbf{a}}} J$, we set $\nabla J(\mathbf{a}) = 0$:

$$\begin{aligned} \nabla_{\mathbf{a}} J(\mathbf{a}) &= 2\mathbf{H}^T\mathbf{Ha} - 2\mathbf{H}^T\mathbf{x} = 0 \\ \mathbf{H}^T\mathbf{Ha} &= \mathbf{H}^T\mathbf{x} \\ \implies \hat{\mathbf{a}}_{ls} &= (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \end{aligned}$$

It can be noted that this solution is similar to that provided by the Yule Walker estimates for the optimal $\mathbf{a}$, which are derived like so:

$$\mathbf{r_{xx}} = \mathbf{R_{xx}a} \implies \mathbf{a_{yw}} = \mathbf{R_{xx}}^{-1}\mathbf{r_{xx}}$$

$$\text{where } \mathbf{r_{xx}} = \begin{bmatrix} \hat{r}_{xx}[1] \\ \hat{r}_{xx}[2] \\ \vdots \\ \hat{r}_{xx}[p] \end{bmatrix}, \quad \mathbf{R_{xx}} = \begin{bmatrix} \hat{r}_{xx}[0] & \hat{r}_{xx}[1] & \dots & \hat{r}_{xx}[p-1] \\ \hat{r}_{xx}[1] & \hat{r}_{xx}[0] & \dots & \hat{r}_{xx}[p-2] \\ \vdots & \vdots & \ddots & \vdots \\ \hat{r}_{xx}[p-1] & \hat{r}_{xx}[p-2] & \dots & \hat{r}_{xx}[0] \end{bmatrix}$$

It can be noted that the Least Squares solution is equivalent to the Yule Walker solution if $M = p$. When $M > p$, the LS solution can be regarded as as a modified Yule-Walker equations.

3.3.2 The Observation Matrix H

The matrix $\mathbf{H}$ is constructed using the autocorrelation values of $\mathbf{x}$. If $\mathbf{x}$ is a purely deterministic signal, then the matrix $\mathbf{H}$ will also be deterministic. If $\mathbf{x}$ is corrupted by noise or can be modelled as a stochastic process, then the matrix $\mathbf{H}$ will also be modelled as a stochastic process. Different realisations of $\mathbf{x}$ may yield different ACFs, and hence a different observation matrix $\mathbf{H}$.

3.3.3 Obtained AR Coefficients

Model Order:	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
AR(1):	0.7641									
AR(2):	1.5736	-0.8725								
AR(3):	2.2166	-1.9304	0.6007							
AR(4):	2.1201	-1.6673	0.3330	0.1044						
AR(5):	2.0891	-1.7843	0.7808	-0.3943	0.2021					
AR(6):	1.9868	-1.5905	0.3252	0.5478	-0.7659	0.3969				
AR(7):	1.8192	-1.2861	0.1387	0.3291	-0.1003	-0.2965	0.2918			
AR(8):	1.7282	-1.2178	0.2264	0.1868	-0.2314	0.2492	-0.3126	0.2638		
AR(9):	1.6999	-1.1806	0.1907	0.2261	-0.2597	0.1842	-0.1140	0.0530	0.0915	
AR(10):	1.6990	-1.1861	0.1994	0.2103	-0.2359	0.1668	-0.1440	0.1486	-0.0092	0.0427

Table 1: AR coefficient Obtained Using LS Method

The LSE approach is now applied to obtain values of $\mathbf{a}$ for different model orders. M = 75 was chosen as the length of $\mathbf{x}$. The resultant values of $\mathbf{a}$ are displayed in table 3.

3.3.4 Approximation error

The least squares estimate of $\mathbf{x}$ is given by $\mathbf{S} = \mathbf{H}\hat{\mathbf{a}}_{ls}$. The approximation error is then described using the MSE of this estimate, given by $\frac{1}{N}||\mathbf{x} - \mathbf{H}\hat{\mathbf{a}}_{ls}||^2$. The approximation error, E_p, has been plotted alongside the MDL, AIC and AICc. These indicate that the optimal model length is 3, given that there is a common minimum present for all model selection criteria.

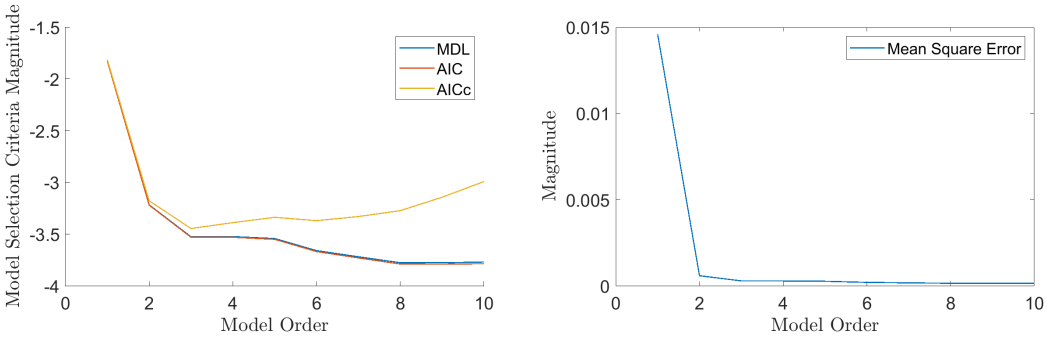

Figure 36: E_p, MDL, AIC and AICc for Least Squares solutions

3.3.5 Model-based PSD of LSE

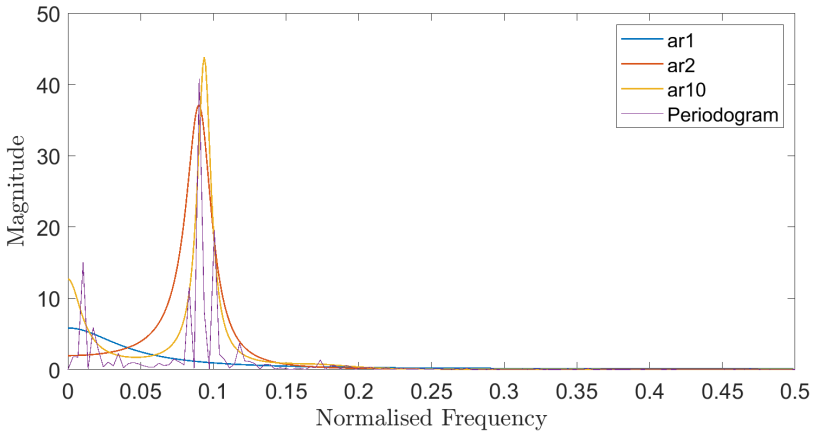

Figure 37: Model based PSDs obtained via Least Squares Method

The results from figure 37 indicate similar behaviour to that of Section 3.2.5. Like before, AR10 captures the most similar result to the periodogram; this could be attributed to overfitting to the model. AR2 also excellently captures the larger peak, but like before struggles to capture the lower frequency peaks. AR1 can once again be considered as undermodelled.

3.3.6 Approximation Error of Varying Data Lengths

This section will focus on how the approximation error will vary with varying the data lengths used to compute the ACF, and in turn $\mathbf{x}$ and $\mathbf{H}$. Within this subsection, M was kept constant at 9. This is to ensure that any trends/behaviour are solely due to N. The results, shown in figure 38 show a minimum found at $N = 20$, which implies that 20 is the optimal length. Longer data lengths seem to result in a greater MSE, and in turn poorer performance. This could be down to the biased ACF estimate used in this section; there is a bias present in every every ACF time lag estimate, and so the error caused by this bias will accumulate as nire values are factored into the computation.

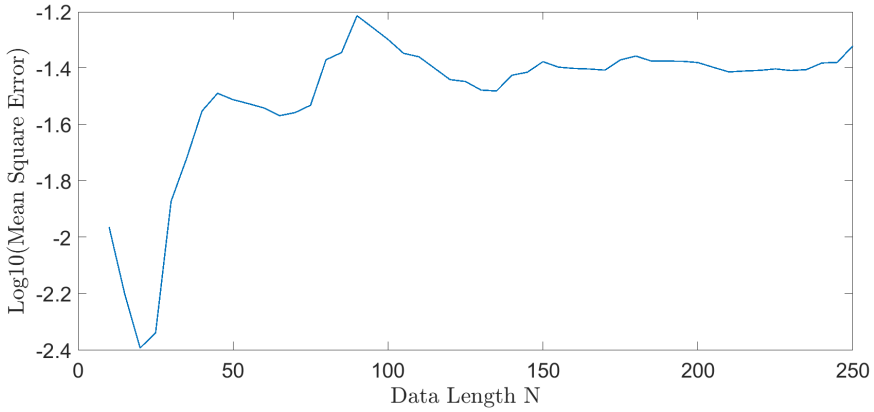

Figure 38: MSE with varying data length N

3.4 Spectrogram for time-frequency analysis: dial tone pad

In this section, we employ FFT to better illustrate the importance of time-frequency analysis, using the case study of touch-tone devices. The underlying concept of touch-tone telephone dialling is the Dual Tone Multi-Frequency (DTMF) system, which assigns a signal composed of two sinusoids to each button of the keypad, that is $y[n] = sin(2\pi f_1 n) + sin(2\pi f_2 n)$, where f_1 and f_2 are a unique combination of frequencies specific to each button. The sample rate (f_s) is equal to $32.768\,kHz$. This is a common frequency generated through crystal oscillators. In turn, the Nyquist frequency $\left(\frac{f_s}{2}\right)$ is equal to $16.384\,kHz$. This is well above the highest frequency employed by the system ($1477Hz$), and so the sampling rate is sufficient. Each digit is pressed for $0.25s$, followed by an idle time 0.25 seconds before the next digit, making y last 5.25 seconds in total.

3.4.1 Generating the London Landline Number

The first step was to generate a London landline number, which is in the form "020 XXXX XXXX", where the final eight digits were selected randomly from a discrete set ranging from (1:9) with uniform distribution. The generated landline number was "020695355424", and all further analysis will be conducted with this. Plots for the generate landline number are shown in figure 39. The behaviour described earlier can be clearly seen.

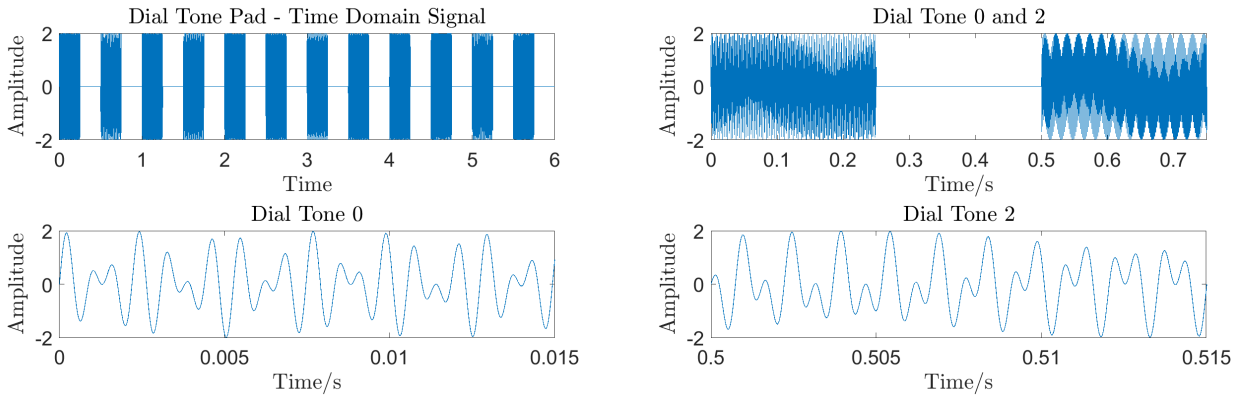

Figure 39: London Landline number sequence generation

3.4.2 Dial Tone Pad Spectrogram

Figure 40 displays the spectrograms of the randomly generated signal - one with a Hanning window and one without. When a digit is being sent, both graphs display show two clearly identifiable yellow peaks, which are congruous to the frequencies being transmitted for that digit. The Hanning window results in the tapering around the frequencies to reduce much quicker. However, without the presence of noise, this ultimately makes a negligible difference.

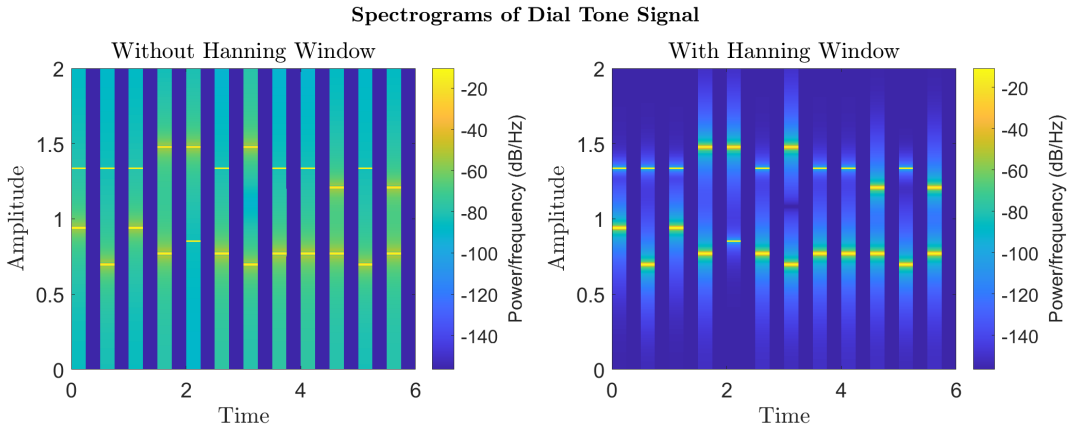

Figure 40: Spectrogram with and without Hanning Window

3.4.3 Sequence Identification

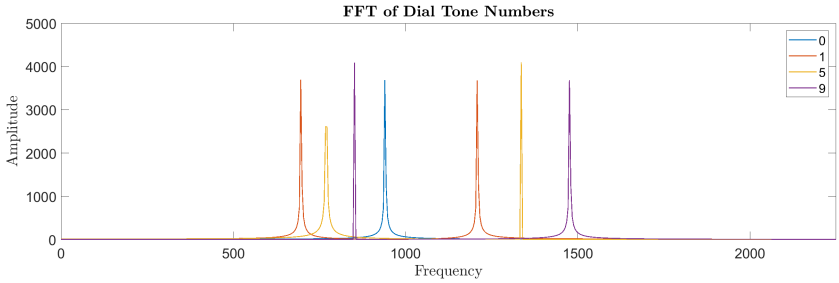

Figure 41: FFT of each applied digit

We can easily identify a key-press by taking the FFT of the signal within the 0.25 second time period where an individual digit is sent. This FFT will provide two distinct peaks, which will correspond to the unique combination of frequencies for that digit being transmitted. From this, the digit can be deciphered.

To illustrate this, four numbers have been plotted in figure 41, each with their own uniquely identifiable frequency responses.

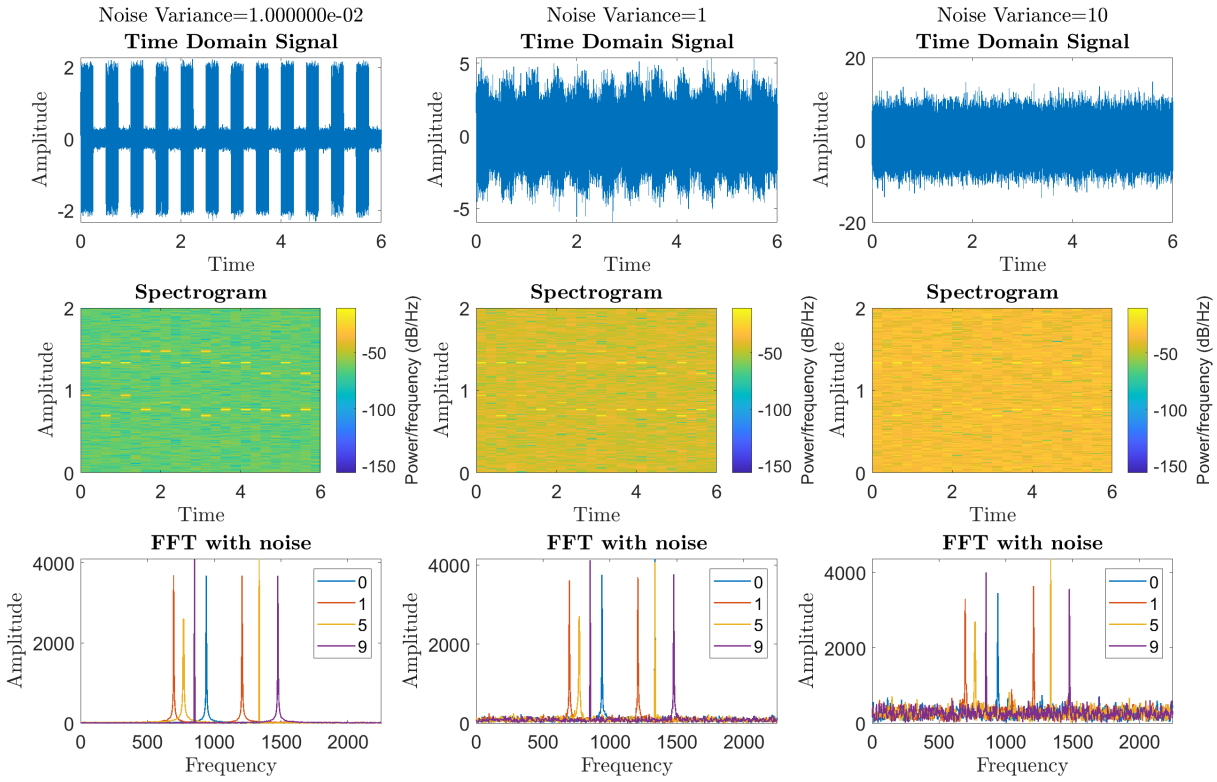

Figure 42: Analysis including varying noise power

3.4.4 Repeated Analysis With Channel Noise

Within this section, we will evaluate how the system's performance with AWGN, varying the noise power and seeing its effects. The same landline number has been used in this section. The noise power was varied from 0.01 to 10, and has been plotted in figure 42. As the noise power is increased, the signal becomes less and less noticeable, particularly in the time domain. The points in the spectrogram also get less distinct, even with the presence of the Hanning window. Here, the noise power is added all across the spectrum (due to the 'white' nature). However, when applying the FFT to the signals, we can still see distinct peaks. If the detection threshold is increased, the dial tone pad should not have any issues with detecting numbers.

3.5 Real World Signals: Respiratory sinus arrhythmia from RR-Intervals

3.5.1 PSD Estimates of RRI trials

This section uses previous RRI data to try to visualise Respiratory sinus arrhythmia. In trial 1, the breathing rate is unconstrained. In trial 2, the breathing rate is constrained to 25 breaths a minute, and in trial 3 the breathing rate is constrained to 7.5 breaths per minute. The data was sampled at 4Hz. The periodograms for the three RRI trials, with different averaging windows have been plotted in figure 43.

3.5.2 Differences in Periodograms between trials

Firstly, it is clear to see that the 50s averaging window produces a distinctly smoother PSD. the 150s averaging window does not provide the same degree of effectiveness, due to the fact that there are fewer segments to average. Looking at differing behaviour between trials, we can see that trials 1 and 2 have reasonably similar shapes. Both have distinct peaks at their DC values, with the majority of the power

clustering closely around it. There seems to be an small but noticeable peak present at $f = 0.1$ in the unaveraged trial 2 PSD, but given that this peak reduces in size for the 50s averaged trial, it can be assumed as noise. Constrained breathing to 50 breaths per minute does not seem to make a significant impact. Trial 3 produces a considerably different periodogram. An offset peak is seen at $f = 0.03Hz$, suggesting the presence of an additional signal, which is caused by the individual experiencing RSA. Trial 1 has a noticeable DC peak of amplitude 0.125.

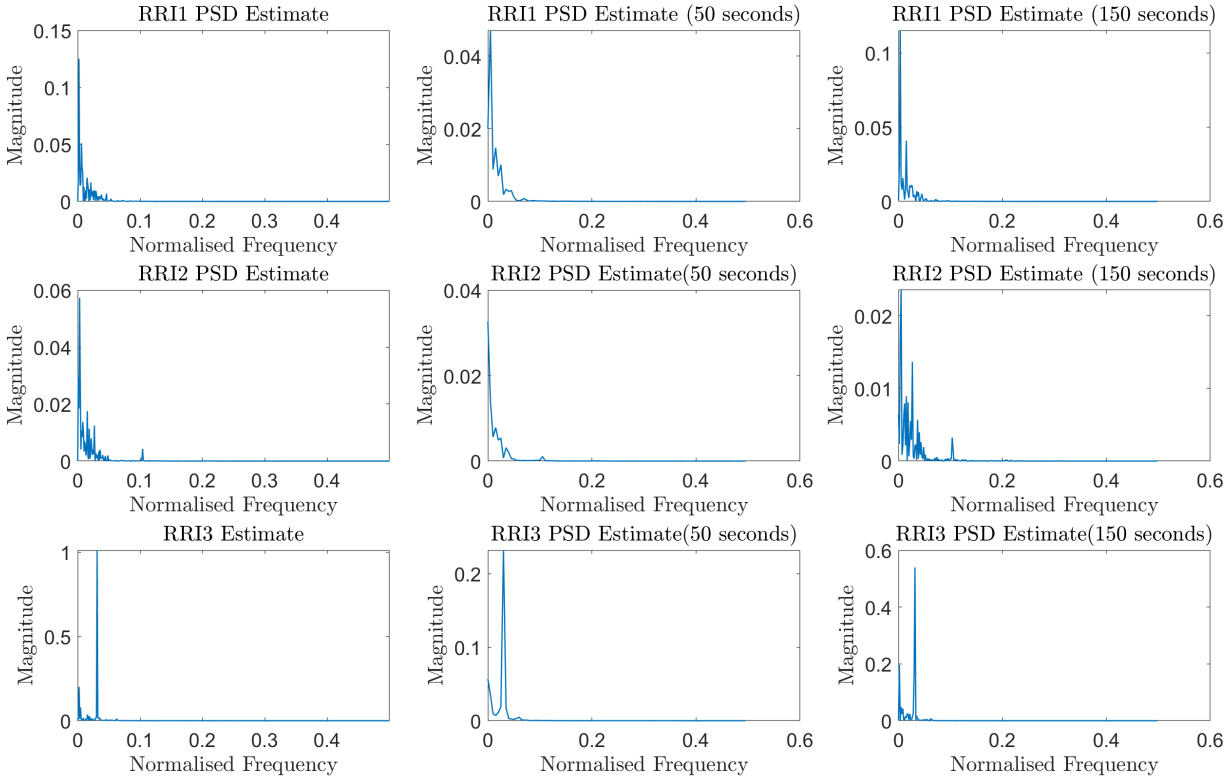

Figure 43: Periodograms for the three trials with varying averaging windows

4 Optimal filtering - Fixed and Adaptive

4.1 Wiener filter

A 1000 sample WGN signal $\mathbf{x}$ (of unit variance) is generated and passed through the filter with coefficients $\mathbf{b} = [1,2,3,2,1]^T$ and $\mathbf{a} = \mathbf{1}$ (MA filter). The output signal, $\mathbf{y}$, is then summed with $\eta[n]$, which is another WGN signal with variance σ_η^2. This is to try to model a real world system more effectively. We now assume that the filter behaviour is unknown, and try to obtain the filter coefficients from the summed signal, labelled $\mathbf{z}$ and $\mathbf{x}$. The optimal coefficients are provided by the Wiener solution are obtained with equation 36. Here, $\mathbf{R}_{xx}$ is the autocorrelation matrix of dimension $N_w \times N_w$, and $\mathbf{p}_{zx}$ is first N_w components of the cross correlation of $\mathbf{x}$ and $\mathbf{z}$.

$$\mathbf{w}_{opt} = \mathbf{R}_{xx}^{-1}\mathbf{p}_{zx} \tag{36}$$

The optimal weights for a realisation (and the corresponding matrices required to calculate these) of $\mathbf{x}$ for

when $\sigma_\eta = 0.1$ were obtained using MATLAB, and are shown below.

$$\mathbf{R}_{xx} = \begin{bmatrix} 0.967 & 0.036 & 0.015 & -0.016 & -0.029 \\ 0.036 & 0.967 & 0.036 & 0.015 & -0.016 \\ 0.015 & 0.036 & 0.967 & 0.036 & 0.015 \\ -0.016 & 0.015 & 0.036 & 0.967 & 0.036 \\ -0.029 & -0.016 & 0.015 & 0.036 & 0.967 \end{bmatrix}, \mathbf{p}_{zx} = \begin{bmatrix} 0.217 \\ 0.466 \\ 0.690 \\ 0.471 \\ 0.226 \end{bmatrix}, \mathbf{w}_{opt} = \begin{bmatrix} 0.2117 \\ 0.4454 \\ 0.6735 \\ 0.4505 \\ 0.2197 \end{bmatrix} = \frac{1}{5} \begin{bmatrix} 1.0583 \\ 2.1672 \\ 3.3677 \\ 2.2125 \\ 1.0984 \end{bmatrix}$$

4.1.1 Optimal Wiener Filter Coefficients

The optimal weights provide a good indication of the behaviour of the unknown system. The weights behave as scaled estimates of the true filter parameters. This is because of the normalisation that takes place at the output. The SNR is obtained calculated as $10\log\left(\frac{1}{\sigma_\eta^2}\right) = -10\log(\sigma_\eta^2)$, due to $\mathbf{y}$ being normalised and having unit variance. Within the realisation of $\mathbf{x}$ where $\sigma_\eta = 0.1$, the theoretical SNR is 20dB. An SNR of 19.8873dB is obtained. This is very close to the theoretical value, and the obtained SNR would converge to its real value with a larger $\eta[n]$.

4.1.2 Adjusting σ_η^2 and N_w

σ_η^2	SNR (dB)	$5w_1$	$5w_2$	$5w_3$	$5w_4$	$5w_5$
Ideal	N/A	1	2	3	2	1
0.1	9.8079	1.0428	1.9896	3.2695	2.2098	0.9867
1	-0.1920	1.1206	2.1706	3.3521	2.2371	1.1424
2.5	-4.4783	0.9344	2.2718	3.3758	2.8886	0.9699
5	-7.0499	1.1632	2.5287	2.8210	2.0083	1.2848
7.5	-8.7171	1.5729	2.5457	3.5055	2.1409	1.1325
10	-9.9967	0.8024	1.6196	4.2594	2.5572	0.7793

Table 2: SNR and Wiener solutions for varying noise powers - N.B. Scale factor of 5 has been applied to make the behaviour clearer.

Theoretically, as the noise variance is increased, the accuracy of the filter coefficient estimates decreases. This is because the SNR increases with reduced noise variance. This is what can be seen in table 2. For $N_w \geq 5$, we will get N_w weights. The filter only requires five variables to be described, and so any extra variables should theoretically be zero. In reality, these terms will not be zero due to the presence of noise, but they will be extremely small (permitting that $\mathbf{x}$ is of sufficient length). For example, taking $N_w = 5$, the Wiener solution for a realisation of $\mathbf{x}$ provides weights of $\mathbf{w} = [1.0376, 2.1375, 3.0112, 2.2117, 0.9566, 0.0023]^T$; the last coefficient is considerably smaller as hypothesised.

4.1.3 Computational Complexity of Wiener solution

Generating the autocorrelation for a single τ of signal $\mathbf{x}$ requires (N-1) multiplications and (N-1) summations. We need to compute N_w values of τ, resulting in an overall complexity of approximately $\mathcal{O}(N_w\, N^2)$. These are then placed into the autocorrelation matrix. This is assumed to have a constant complexity. Applying the same logic, obtaining $\mathbf{p}_{zx}$ will also have an overall complexity of $\mathcal{O}(N_w\, N^2)$. Obtaining the inverse of $\mathbf{R}_{xx}$ has a complexity of $\mathcal{O}(N_w^3)$ (this was given in the question). The last step, which is the matrix multiplication of $\mathbf{R}_{xx}^{-1}$ and $\mathbf{p}_{zx}$ requires $N_w * N_w$ element-wise multiplications, resulting in a complexity of $\mathcal{O}(N_w^2)$ for the product. The overall complexity is therefore $\mathcal{O}(N_w\, N^2)+\mathcal{O}(N_w^3)+\mathcal{O}(N_w\, N^2)+\mathcal{O}(N_w^2) \approx \mathcal{O}(N_w\, N^2)+\mathcal{O}(N_w^3)$. Given that $N >> N_w$, the complexity can be reduced further to $\mathcal{O}(N^2)$.

4.2 The Least Mean Square (LMS) algorithm

When trying to approximate a non-stationary function, the Wiener solution cannot be applied, as stationarity is assumed. Instead, we can approximate non-stationary signals by using an iterative approach (an adaptive filter). One such example is the Least Mean Squares algorithm, which is defined as so:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu\, e[n]\mathbf{x}(n), \qquad n = 0, 1 \ldots \tag{37}$$

Within 37, μ controls the 'learning rate' of the algorithm, $e[n]$ is computed as the difference between $z[n]$ and $\hat{y}[n]$ and $w[0] = 0$.

4.2.1 Coding LMS Function on MATLAB

```
function [y_hat , e , w] = lms(x, z, mu, lms_order)
    N = length(x);
    y_hat = zeros(N, 1);
    e = zeros(N, 1);
    w = zeros(lms_order, N);
    for i = lms_order:N
        y_hat(i) = w(:,i)' * x(i:-1:(i-lms_order+1));
        e(i) = z(i) - y_hat(i);
        w(:, i+1) = w(:,i) + mu * e(i)*x(i:-1:(i-lms_order+1));
    end
end
```

Figure 44: `lms` function code

The function `lms` was coded, shown in figure xx, taking in the inputs **x**, **z**, μ and the estimate order. This function was then applied to the same process as section 4.1.1. The results have been plotted in figure 45.

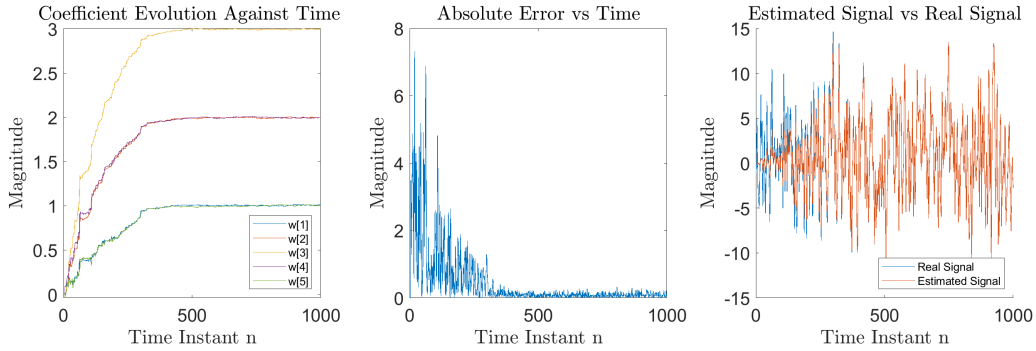

Figure 45: `lse` function outputs from filter described in 4.1.1

We can see that the weights provided by the LMS algorithm converge to accurate vlaues which ultimately lead to low errors. The initial error is quite high, before reducing significantly; this is due to the weights being initially set to zero and the prediction gain not allowing for quicker convergence. The real-time nature of the LMS algorithm provides a significant advantages over the Wiener solution, which cannot make real-time adjustments. The Wiener solution is also limited to stationary processes, whereas the LMS algorithm can be applied to be stationary and non-stationary process.

4.2.2 Determining Optimal adaptation gain

Adjusting the prediction gain of the LMS algorithm can significantly change the behaviour seen by figure 46. When $\mu = 0.002$, we can see that the rate of convergence is too slow, with the estimates yet to converge to their true values even after 1000 samples. This results in a reasonably high error throughout, but a smoothly changing weight with little/no fluctuations. When $\mu = 0.0125$, convergence occurs significantly faster, and the error remains low thereafter. We begin to see more fluctuations. but the system remains stable. When $\mu = 0.022$, we see a more unstable and unpredictable estimate. We see significant error throughout the estimate, even though the values jump to the right magnitude very quickly. A much more extreme version of this can be seen when $\mu = 0.5$, which does not provide us with useful results.

4.2.3 Computational Complexity of LMS algorithm

When looking at the LMS algorithm, the dominant computation within each iteration is the inner product between $\mathbf{x}$[n] and $\mathbf{w}$[n], which requires $N_w + 1$ multiplications and $N_w + 1$ additions. This results in a complexity of approximately $\mathcal{O}(N_w)$ for each iteration. This process is then repeated N times, resulting in an overall complexity of $\mathcal{O}(N\,N_w)$.

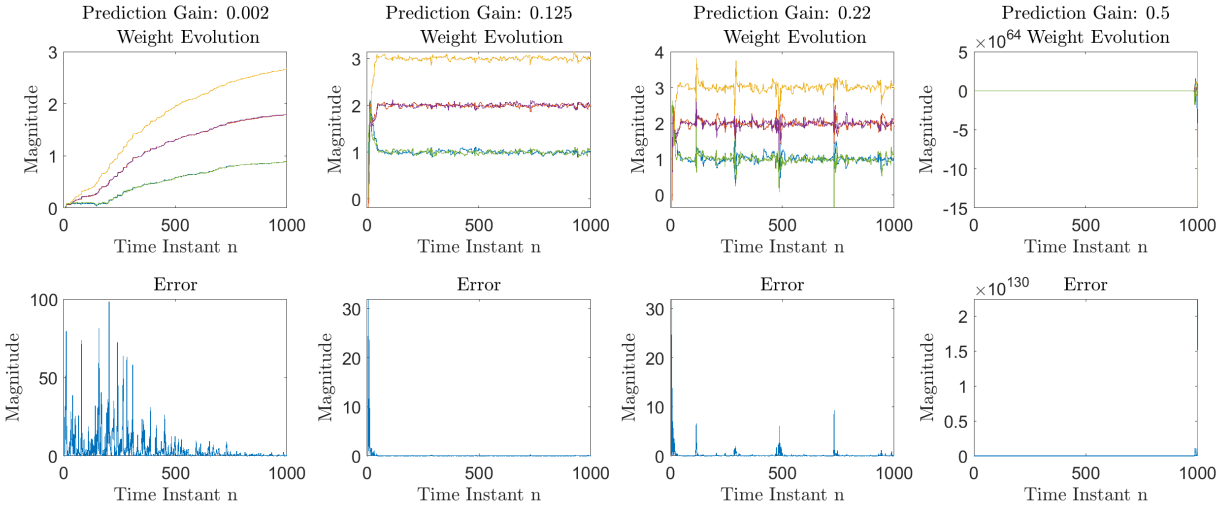

Figure 46: Adjusting Prediction Gain

4.3 Gear shifting

One way of improving the accuracy of the estimate is to vary the adaptation gain depending on the time instant n,

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu[n]e[n]\mathbf{x}(n) \tag{38}$$

We now have an extra degree of control to tune μ: by controlling the μ update equation, we can more effectively combat the issues that faced the previous section. The first is the initial overshoot, when the prediction gain is higher than it should be. The second issue are the fluctuations/instability within the weights once they are close to their ideal values. The equation for μ is shown below:

$$\mu[n+1] = u[n]\left(\mu(n) - 0.005e[n] - 0.001e[n-1]\right), \quad \text{where u[n] is the unit step function} \tag{39}$$

Both the overshoot and the oscillations are mitigated by the error terms that are subtracted. Decreasing the value of μ when the error term is large means that the update will be less severe, reducing the likelihood of overshooting. Including two error terms means that if the sign of the error is changing between iteration to iteration (which may lead to fluctuations), the impact on the update term is mitigated. The constant terms were derived heuristically. If given more time, these parameters would be better determined using some form of validation testing. The sign of μ is evaluated by the unit step function, and μ set to zero if a negative value is obtained. If the value is positive, ten This was initially done as sign variations in μ can result in instability; the sign of the weight updates should be solely dependent on the sign of error. It also has an added benefit in that it provides a 'buffer' period to ensure that μ will be set to zero until the error vector itself is negative. This allows the model to self-correct, resulting in faster convergence. A final condition was added, where if abs(e[n]) was less than 0.02, μ was scaled down. This was done to further reduce oscillations. The final result is plotted in figure 47, and highlights the greater stability and convergence of the gain.

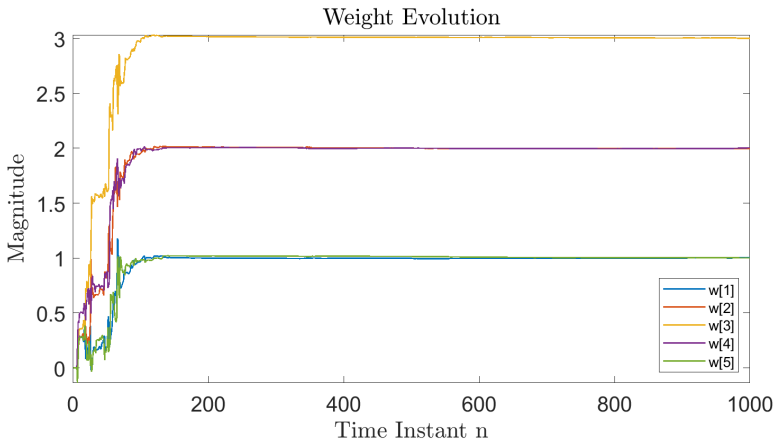

Figure 47: Performance Applying Gear Shifting

4.4 AR Process Identification

4.4.1 Estimating AR Coefficients using LMS Algorithm

In the previous sections, we focused on estimating the coefficients of an MA process iteratively. Now, we will try to do the same for an AR process. Like before, a LMS algorithm will be employed to update the model weights. These will be adjusted according tho the error value $e[n] = x[n] - \hat{x}[n]$, where $x[n]$ is the true current value of the signal, and $\hat{x}[n]$ is the estimated current value of the signal. The update equation is shown below:

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \mu\, e[n]\, \mathbf{x}[n-1], \text{ where } \mathbf{x}[n-1] = \begin{bmatrix} x[n-1] \\ x[n-2] \\ \vdots \\ x[n-p] \end{bmatrix}, \text{p = AR model Order} \tag{40}$$

To further investigate this behaviour, an AR(2) process is modelled with coefficients $\mathbf{a} = [1, 0.9, 0.2]$. The estimated value of the signal is therefore defined as:

$$\hat{x}[n] = a_1\, x[n-1] + a_2\, x[n-2] \tag{41}$$

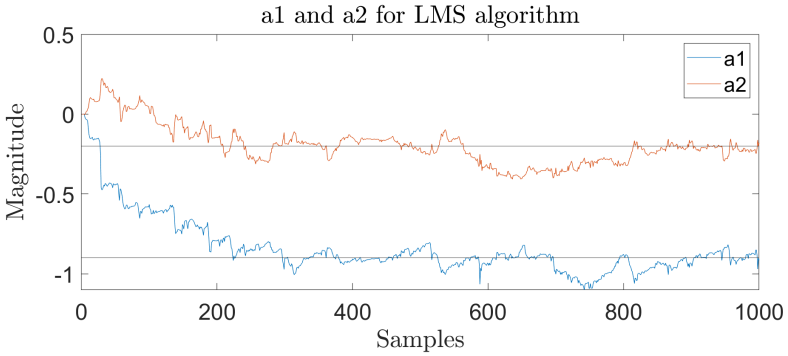

Figure 48: Weight Evolution for AR Model

The prediction gain is then set to 0.1. The evolution of the coefficients a_1 and a_2 has been plotted in figure 48, and we can see that the coefficients do converge to the correct values (note that these are negative due to the behaviour of the `filter` function), though there are a reasonable number of oscillations throughout, with

convergence being relatively slow. This can be adjusted by varying the adaptation gain, which is discussed in the next part.

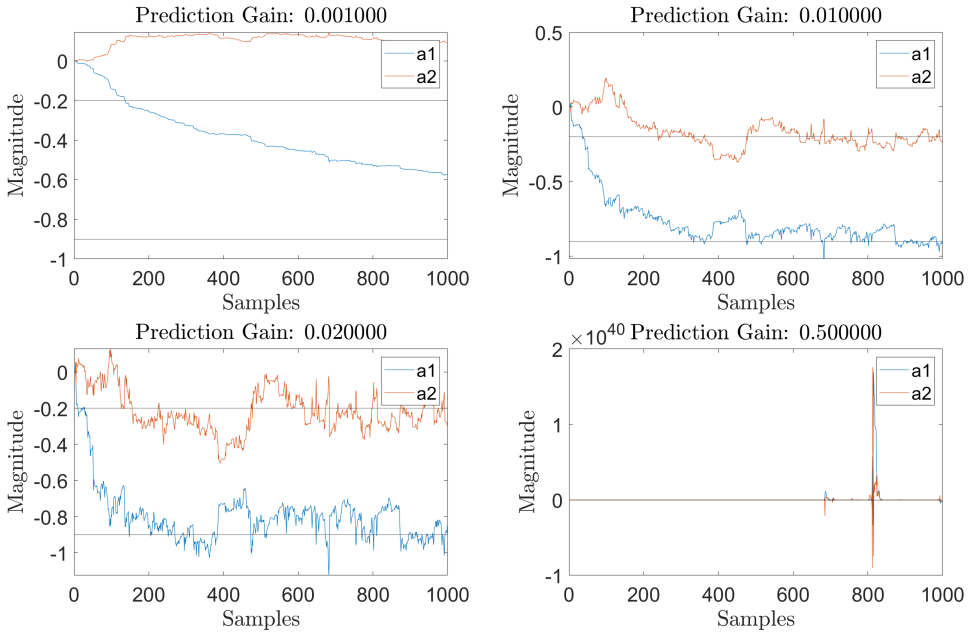

Figure 49: Adjusting the Prediction Gains - AR Models

4.4.2 Adjusting Adaptation Gains

From figure 49, we can come to same conclusions as in Section 4.2.3. When $\mu = 0.001$, convergence does not take place within 1000 samples. We can see that $\mu = 0.02$ results in quicker convergence when comparing it to $\mu = 0.01$, but greater oscillations thereafter. This is highlighted to a greater extreme when $\mu = 0.5$.

4.5 Speech Recognition

In this section, the LMS algorithm will be applied to vocal recordings. Speech is an example of a non-stationary signal, and so the analysis will be conducted over a very small time frame to ensure 'quasi-stationarity'. Here, N = 1000 samples are taken, with the $f_s = 44.1kHz$. The sounds "e", "a","s", "t" and "x" were recorded and inspected on MATLAB.

4.5.1 AR Predictor Performance

Figure 52 displays how the performance of the predictor varies with order, along with the weights and how they evolve with time for the sound for 'A'. On the whole, each order performs very well by itself, with an order of 6 looking very slightly better than the rest, though it is hard to tell. It is worth noting that some weights do not seem to converge to a fixed point, whilst still providing an accurate result. This may be down to the vocal recording being a non-stationary process, but still reasonably predictable. Gear shifting was considered, but was avoided due to the complexity of the task to tuning all of the associated parameters relevant to it, particularly when a constant μ worked effectively.

To understand the behaviour of adjusting the prediction gain, the MSE for a range of prediction gains has been plotted. We can see that initially, the error is high; this is due to the learning rate being too small, and so insufficient changes are made at every step. It then decreases, before increasing with a large predictor gains. The optimal predictor gain was determined to being approximated 0.735, as can be seen in figure 50. This analysis was repeated for all sounds.

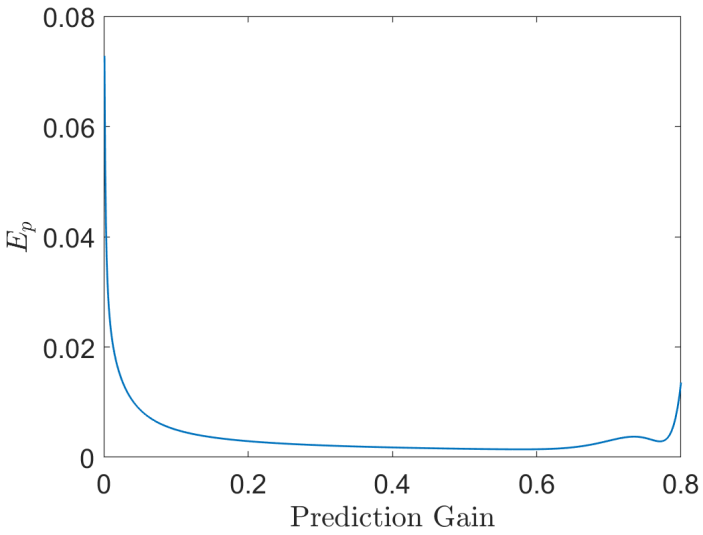

Figure 50: Impact of Prediction Gain

4.5.2 Optimal filter length

To determine the optimal order we need to first determine what sort of performance metric we wish to compare the models on. A more heuristic approach would be to adjust the model order and determine which order visually fits best to the graph. However, this introduces challenges when comparing similar results, and can introduce an element of human error. As a result, an analytical approach has been determined on.MDL, AIC and AICc were three good analytical methods employed to evaluate order performance. The results have been have been plotted in figure xx for the sounds 'a' and 'e'. We can see that for 'a', the optimal filter order length is p = 6, and for 'e' the optimal length is 51.

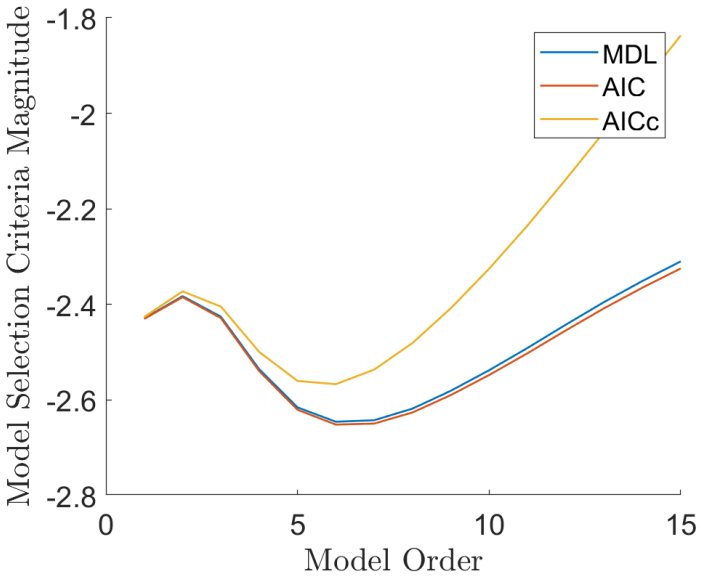

Figure 51: Impact of Prediction Gain

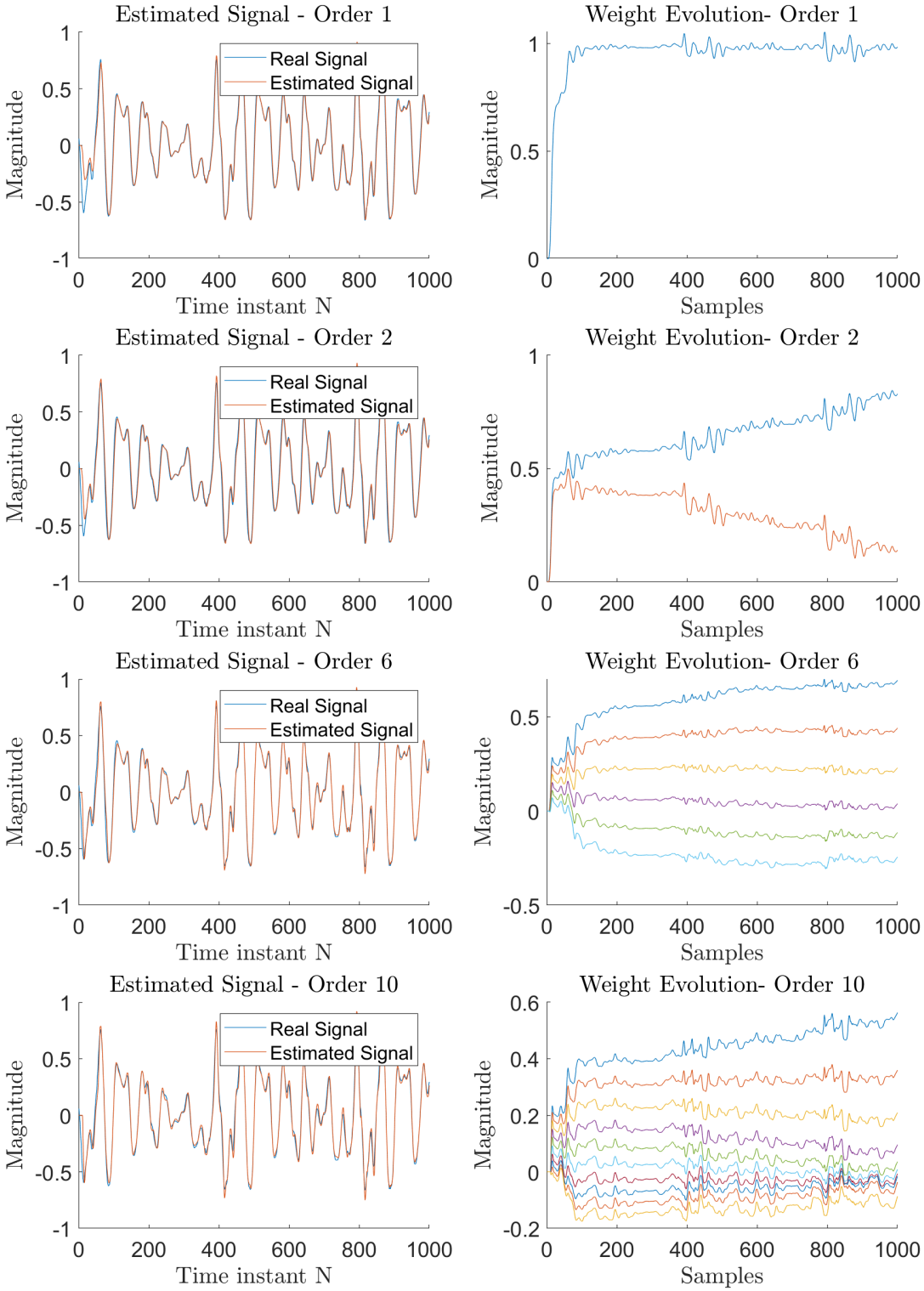

Figure 52: AR Predictor Performance of varying model orders of 'a'

4.5.3 Downsampling and its Effect on the Predictor Gain

Reducing the sample frequency of the signal results in the 'quasi-stationarity' assumption to be less likely to hold. The lower sample frequency was modelled using `resample`. This led to the resultant weights to consistently change. It also resulted in the optimal prediction gain decreasing for all letters by approximately a factor of 2. The optimal model orders also changed. These have been tabulated below:

Sound:	a	e	s	t	x
Original Model Order:	6	6	12	11	9
New Model Order:	11	9	15	11	15

Table 3: Comparing Optimal Model Order, up to $p = 15$

4.6 Dealing with Computational Complexity: Sign Algorithms

A simplified version of the LMS algorithm is the class of the sign LMS algorithms, given by:

- Signed Error: $\mathbf{w}(n+1) = \mathbf{w}(n) + \mu sign(e[n])\mathbf{x}(n)$
- Signed Regressor: $\mathbf{w}(n+1) = \mathbf{w}(n) + \mu e[n] sign(\mathbf{x}(n))$
- Sign Sign: $\mathbf{w}(n+1) = \mathbf{w}(n) + \mu sign(e[n]) sign(\mathbf{x}(n))$

These were applied to the same context as in 4.4, with a random process being modelled as a filter. The results are shown in figure 53. The functions perform quire well on the whole, though they do take longer to converge when compared to the original LMS algorithm. A trade-off clearly forms; the sign algorithms are computationally much more efficient, but do not converge as quickly, and hence can be considered less effective algorithms. Choosing which algorithm of the three is better is challenging, and would likely have to be determined through some validation method.

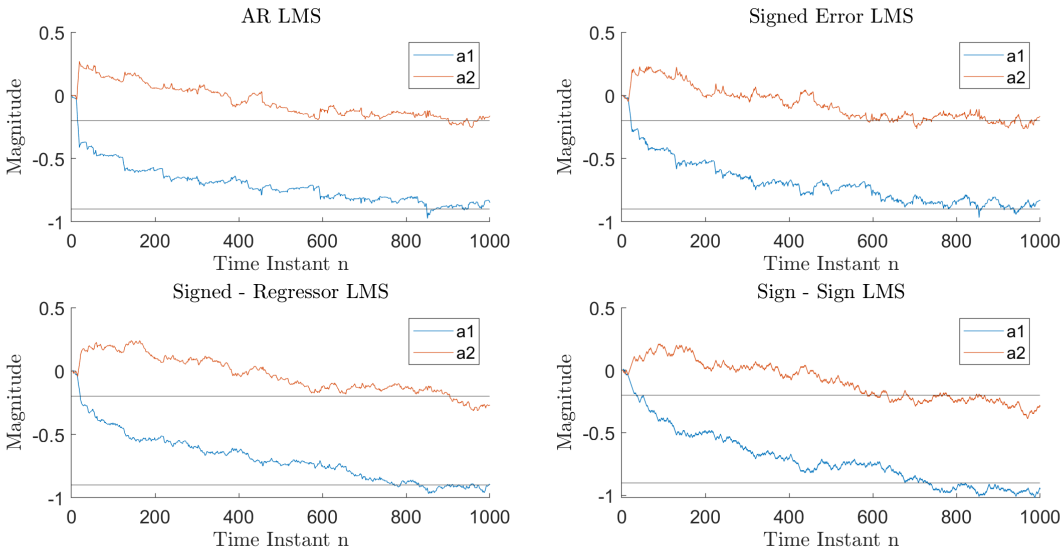

Figure 53: Sign LMS Algorithms

5 MLE for the Frequency of a Signal

5.0.1 Mapping $J(\boldsymbol{\theta})$ to $J'(\boldsymbol{\alpha}, f_0)$

$$J'(\boldsymbol{\alpha}, f_0) = J'(\alpha_1, \alpha_2, f_0) = (\mathbf{x} - \mathbf{H}\boldsymbol{\alpha})^T(\mathbf{x} - \mathbf{H}\boldsymbol{\alpha}) \tag{42}$$

Using the trigonometric identity: $\cos(p+q) = \cos(p)\cos(q) - \sin(p)\sin(q)$:

$$\mathbf{H}\boldsymbol{\alpha} = \begin{bmatrix} A\cos\phi \\ A\cos\phi\,\cos(2\pi f_0) - A\sin\phi\,\sin(2\pi f_0) \\ \vdots \\ A\cos\phi\,\cos\left(2\pi f_0\,(N-1)\right) - A\sin\phi\,\sin\left(2\pi f_0\,(N-1)\right) \end{bmatrix} = \begin{bmatrix} A\cos\phi \\ A\cos(2\pi f_0\,+\,\phi) \\ \vdots \\ A\cos\left(2\pi f_0\,(N-1) + \phi\right) \end{bmatrix} \tag{43}$$

Expanding equation 42:

$$J'(\alpha_1, \alpha_2, f_0) = \mathbf{x}^T\mathbf{x} - (\mathbf{H}\boldsymbol{\alpha})^T\mathbf{x} - \mathbf{x}^T(\mathbf{H}\boldsymbol{\alpha}) + (\mathbf{H}\boldsymbol{\alpha})^T(\mathbf{H}\boldsymbol{\alpha}) = \sum_{n=0}^{N-1}(x[n] - A\cos(2\pi f_0 n + \phi))^2 = J(\boldsymbol{\theta}) \tag{44}$$

5.0.2 Obtaining the Minimising solution

$$\nabla_{\boldsymbol{\alpha}} J'(\boldsymbol{\alpha}, f_0) = -2\mathbf{H}^T\mathbf{x} + 2\mathbf{H}^T\mathbf{H}\boldsymbol{\alpha} = 0 \tag{45}$$

$$2\mathbf{H}^T\mathbf{H}\hat{\boldsymbol{\alpha}} = 2\mathbf{H}^T\mathbf{x} \tag{46}$$

$$\hat{\boldsymbol{\alpha}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \tag{47}$$

Plugging equation 47 into 42:

$$J'(\alpha_1, \alpha_2, f_0) = \mathbf{x}^T\mathbf{x} - (\mathbf{H}^T\boldsymbol{\alpha})^T\mathbf{x} - \mathbf{x}^T\mathbf{H}\boldsymbol{\alpha} + \mathbf{H}^T\boldsymbol{\alpha}^T\mathbf{H}\boldsymbol{\alpha} \tag{48}$$

$$= \mathbf{x}^T\mathbf{x} - 2\mathbf{H}^T\left((\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x}\right)^T\mathbf{x} + \mathbf{H}^T\left((\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x}\right)^T\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \tag{49}$$

$$= \mathbf{x}^T\mathbf{x} - \mathbf{x}^T\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \tag{50}$$

$$\implies \text{minimizing } J'(\alpha_1, \alpha_2, f_0) \text{ is equivalent to maximising } \mathbf{x}^T\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \tag{51}$$

5.0.3 Obtaining MLE of f_0

Using $\mathbf{H} = [\mathbf{c}\ \mathbf{s}]$:

$$\mathbf{x}^T\mathbf{H} = (\mathbf{H}^T\mathbf{x})^T = \begin{bmatrix}\mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x}\end{bmatrix}^T \tag{52}$$

$$\mathbf{H}^T\mathbf{H} = \begin{bmatrix}\mathbf{c}^T \\ \mathbf{s}^T\end{bmatrix}[\mathbf{c}\ \mathbf{s}] = \begin{bmatrix}\mathbf{c}^T\mathbf{c} & \mathbf{c}^T\mathbf{s} \\ \mathbf{s}^T\mathbf{c} & \mathbf{s}^T\mathbf{s}\end{bmatrix} \tag{53}$$

$$\tag{54}$$

With these results, we can express the MLE for frequency f_0 as:

$$J(f_0) = \begin{bmatrix}\mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x}\end{bmatrix}^T \begin{bmatrix}\mathbf{c}^T\mathbf{c} & \mathbf{c}^T\mathbf{s} \\ \mathbf{s}^T\mathbf{c} & \mathbf{s}^T\mathbf{s}\end{bmatrix}^{-1} \begin{bmatrix}\mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x}\end{bmatrix} \approx \begin{bmatrix}\mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x}\end{bmatrix}^T \begin{bmatrix}\frac{N}{2} & 0 \\ 0 & \frac{N}{2}\end{bmatrix}^{-1} \begin{bmatrix}\mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x}\end{bmatrix} \tag{55}$$

Note that this approximation is only valid if $f_0 \neq 0, \frac{1}{2}$. If it is, the matrix $\mathbf{H}^T\mathbf{H}$ will have a determinant of zero, and hence will be singular. If invertible (which is a byproduct of the approximation made), the approximation can be rewritten as:

$$J(f_0) \approx \begin{bmatrix}\mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x}\end{bmatrix}^T \begin{bmatrix}\frac{2}{N} & 0 \\ 0 & \frac{2}{N}\end{bmatrix} \begin{bmatrix}\mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x}\end{bmatrix} \tag{56}$$

5.0.4 Periodogram Equivalent of MLE Result

Expanding upon equation 56:

$$J(f_0) = \frac{2}{N}\left[(\mathbf{c}^T\mathbf{x})^2 + (\mathbf{s}^T\mathbf{x})^2\right] \tag{57}$$

$$= \frac{2}{N}\left(\sum_{n=0}^{N-1}\left[x[n]\cos(2\pi f_0 n)\right)^2 + \left(\sum_{n=0}^{N-1} x[n]\sin(2\pi f_0 n)\right)^2\right] \tag{58}$$

$$= \frac{2}{N}\left|\sum_{n=0}^{N-1} x[n]\left(\cos(2\pi f_0 n) + j\sin(2\pi f_0 n)\right)\right|^2 \tag{59}$$

$$= \frac{2}{N}\left|\sum_{n=0}^{N-1} x[n]e^{-j2\pi f_0 \frac{n}{N}}\right|^2 \tag{60}$$

This is now identical to maximising periodogram, given by $\frac{1}{N}\left|\sum_{n=0}^{N-1} x[n]e^{-j2\pi f \frac{n}{N}}\right|^2$ where $f_0 = \frac{f}{N}$