Statistical Signal Processing and Inference Report

Savraj Sian

CID: 01847921

March 26, 2023

Contents

1 Random Signals and Stochastic Processes **2**
- 1.1 Statistical Estimation . . . 2
- 1.2 Stochastic Processes . . . 4
- 1.3 Estimation of Probability Distributions . . . 7

2 Linear Stochastic Modelling **8**
- 2.1 ACF of Uncorrelated and Correlated Sequences . . . 8
- 2.2 Cross-correlation Function . . . 9
- 2.3 Autoregressive Modelling . . . 10
- 2.4 Cramer-Rao Lower Bound . . . 14
- 2.5 ECG from iAmp Experiment . . . 17

3 Spectral Estimation and Modelling **18**
- 3.1 Averaged Periodogram Estimates . . . 19
- 3.2 Spectrum of Autoregressive Processes . . . 20
- 3.3 The Least Squares Estimation (LSE) of AR Coefficients . . . 23
- 3.4 Spectrogram for Time-Frequency Analysis: Dial Tone Pad . . . 25
- 3.5 Real World Signals: Respiratory Sinus Arrhythmia from RR-Intervals . . . 27

4 Optimal Filtering - Fixed and Adaptive **27**
- 4.1 Wiener Filter . . . 28
- 4.2 The Least Mean Square (LMS) Algorithm . . . 28
- 4.3 Gear Shifting . . . 30
- 4.4 Identification of AR processes . . . 30
- 4.5 Speech Prediction . . . 31
- 4.6 Dealing with Computational Complexity: Sign Algorithms . . . 33

5 MLE for the Frequency of a Signal **35**

1 Random Signals and Stochastic Processes

1.1 Statistical Estimation

A uniform distribution was used to generate a 1000-sample vector $\mathbf{x}$, with each sample $x[n]$ being a realisation of $X \sim \mathcal{U}(0,1)$ at time sample n.

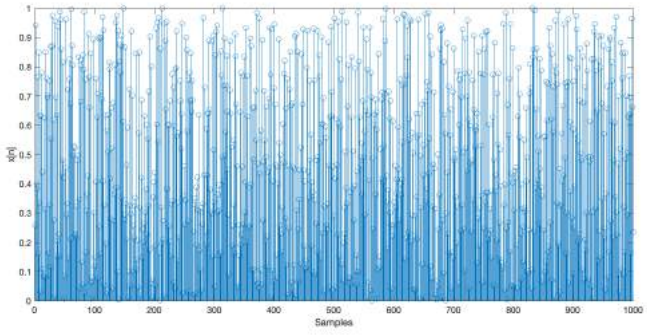


Figure 1: Realisation of the uniform distribution for 1000 samples

1.1.1

The theoretical mean is $m = \mathbb{E}[X]$ and the sample mean is calculated as $\hat{m} = \frac{1}{N}\sum_{n=1}^{N} x[n]$. The theoretical mean of $\mathbf{x}$ is $m = \mathbb{E}[X] = 0.5$ and the sample mean calculated is $\hat{m} = 0.49531$, an error of 0.938%.

1.1.2

The theoretical standard deviation is $\sigma = \sqrt{\mathbb{E}[X - \mathbb{E}[X]]}$ and the sample standard deviation is given by $\hat{\sigma} = \sqrt{\frac{1}{N-1}\sum_{n=1}^{N}(x[n] - \hat{m})^2}$. The theoretical standard deviation of $\mathbf{x}$ is $\sigma = 0.28868$ and the sample standard deviation calculated is $\hat{\sigma} = 0.28704$, an error of 0.568%. The accuracy of both of these estimators is high, given that a large sample number has been used; in general, the higher the number of samples, the closer the estimators will be to the theoretical value.

1.1.3

An ensemble of ten 1000-sample realisations of X were generated, and the sample means and standard deviations of each were calculated. Figures 2 and 3 depict these, with the red lines showing the theoretical mean and standard deviation respectively.

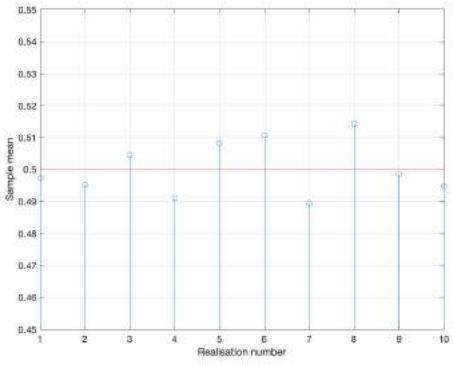


Figure 2: Sample mean of all realisations

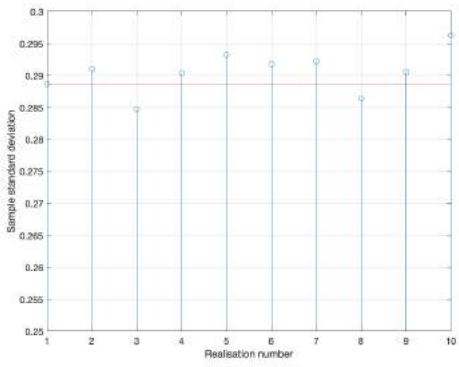


Figure 3: Sample standard deviation of all realisations

The sample mean lies within 0.303% of the theoretical mean and as shown by Figure 4, there is minimal bias, with the maximum absolute value being 0.014. The bias is given by $B = \mathbb{E}[X] - \hat{m}$. The sample standard deviation for the realisations lie within 2.536% of the theoretical standard deviation. Given the small bias and

that the sample estimations are close to the theoretical values, it can be concluded that the sample mean is an unbiased estimator. However, the same cannot be said of the sample standard deviation, whose error is higher.

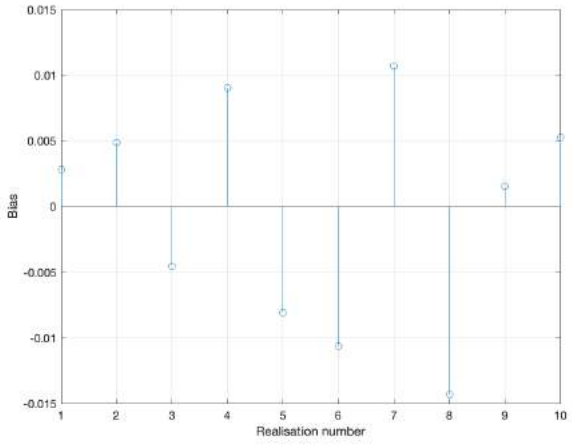
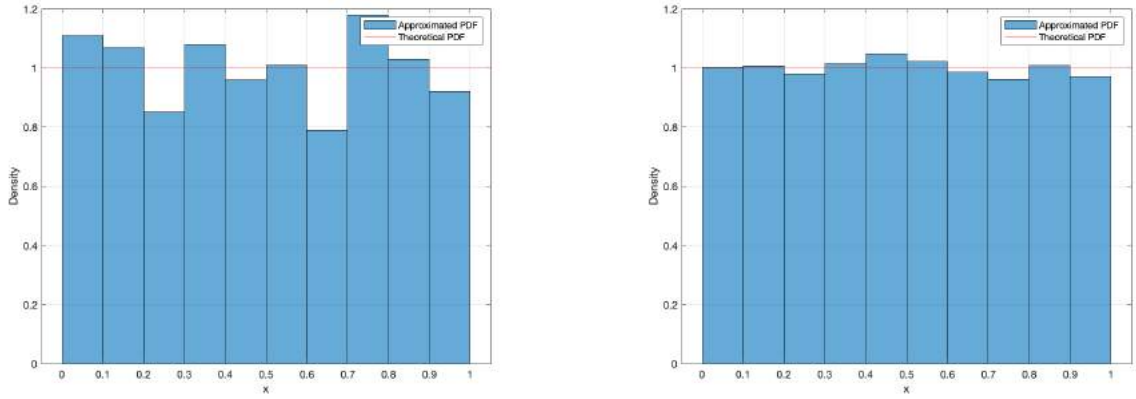


Figure 4: Bias of means for all realisations

1.1.4

Plotting a histogram using realisations allows for an approximation of the probability density function (PDF) of X. Figure 5 shows histograms generated using 1000 and 10000 samples with 10 bins. Increasing the number of samples results in a more accurate representation of the theoretical PDF, shown in red. Increasing the number of bins used resulted in a more varied histogram, and an increase in samples has the estimated PDF converge on the theoretical PDF.



(a) Generated using 1000 samples

(b) Generated using 10000 samples

Figure 5: Histograms showing the approximated and theoretical PDF of X

1.1.5

This analysis was repeated using a normal distribution to generate a 1000-sample vector $\mathbf{x}$, with each sample $x[n]$ being a realisation of $X \sim \mathcal{N}(0,1)$ at time sample n. By definition, the theoretical mean of x is 0 and the theoretical standard deviation is 1. The calculated sample mean of x is 0.03667 and the calculated sample standard deviation is 1.0341. Once again, these are accurate estimator with small errors to the theoretical values.

Ten 1000-sample realisations of X were generated; Figure 6 shows the sample means with the red line at the theoretical mean and Figure 7 shows the sample standard deviations with the red line at the theoretical standard deviation. The mean biases of the sample means and sample standard deviations were small at -0.0157 and 0.0138 respectively.

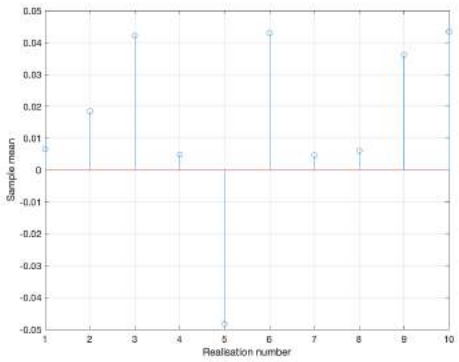


Figure 6: Sample mean of all realisations

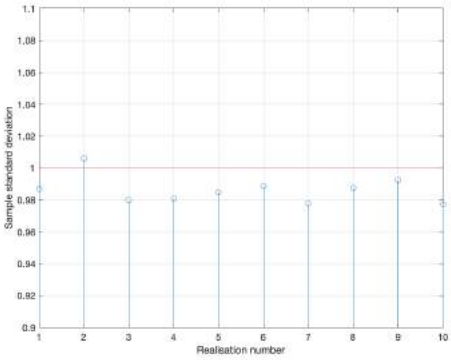
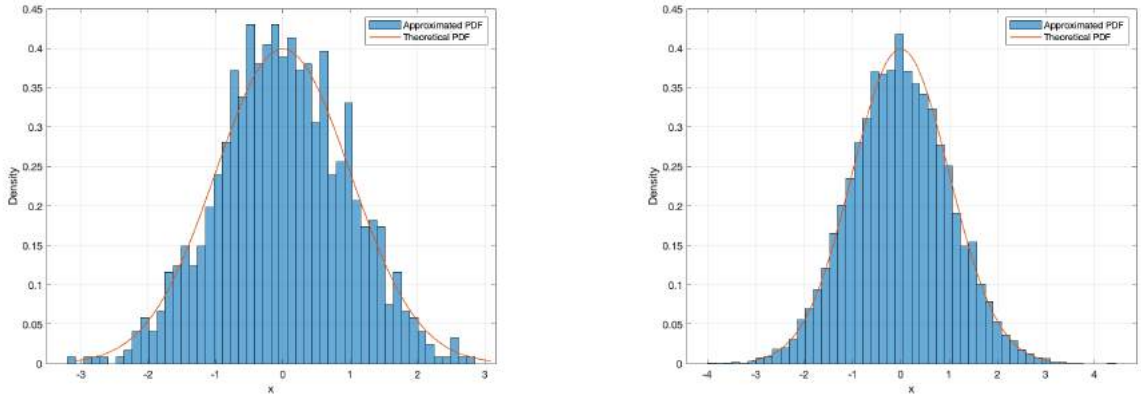


Figure 7: Sample standard deviation of all realisations

As before, approximating the PDF of X with a histogram using more samples results in a more accurate representation of the theoretical PDF. This time, 50 bins were used as opposed to the 10 bins used in the uniform distribution section, as the greater number of bins allows for more of a curve to be seen, whereas the theoretical uniform distribution PDF is a straight line so fewer bins were beneficial in showing a flatter approximation. Figure 8 contains histograms for 1000 and 10000 samples, with the red line showing the theoretical PDF.



(a) Generated using 1000 samples

(b) Generated using 10000 samples

Figure 8: Histograms showing the approximated and theoretical PDF of X

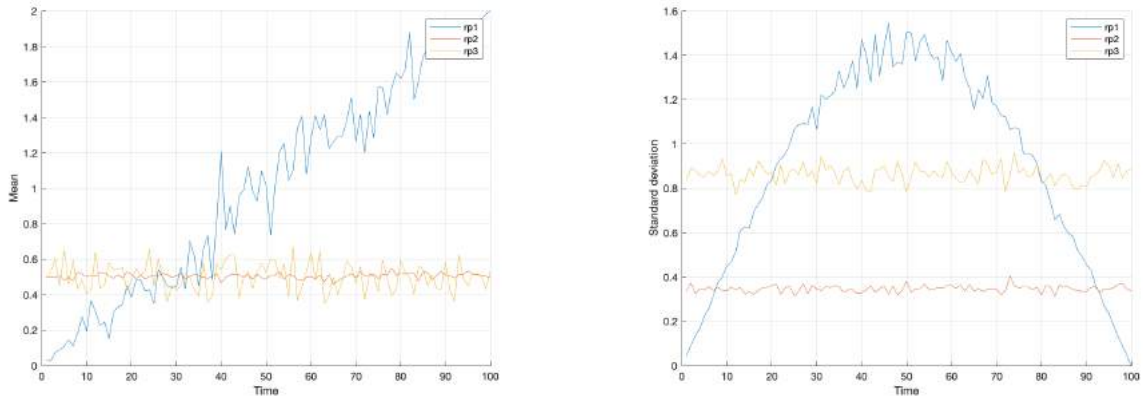
1.2 Stochastic Processes

This section uses three stochastic processes named $rp1$, $rp2$ and $rp3$, with each process giving an ensemble of M realisations of N samples.

1.2.1

The ensemble mean and standard deviation of each process were computed, with $M = 100$ and $N = 100$.

Figure 9 shows that $rp1$ is not a stationary process since its mean and standard deviation vary over time. The plots for $rp2$ and $rp3$ appear to fluctuate around a fixed value and therefore these two processes are stationary since they are time invariant.



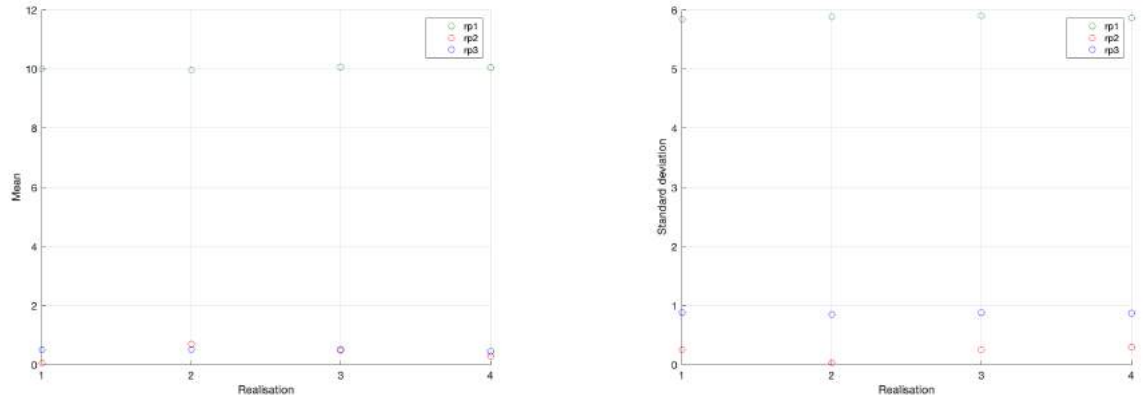
(a) Ensemble mean

(b) Ensemble standard deviation

Figure 9: Ensemble mean and standard deviation for $rp1$, $rp2$ and $rp3$

1.2.2

The mean and standard deviation for each realisation were calculated, with $M = 4$ and $N = 1000$.



(a) Realisation mean

(b) Realisation standard deviation

Figure 10: Mean and standard deviation for each realisation of $rp1$, $rp2$ and $rp3$

$rp1$ cannot be ergodic due to the findings in the previous section; the ensemble mean and standard deviation varied and it is therefore not possible to approximate the theoretical mean using time average since the time average cannot equal the ensemble average.

$rp2$ is not ergodic either as the mean of each realisation varies greatly, from 0.04 to 0.69, which is a far greater degree of fluctuation than is visible from the ensemble averages of the previous section; the ensemble and time averages do not align.

$rp3$ can be said to be ergodic since the mean and standard deviations found in the last section and this section roughly hover around the same values; the means around 0.5 and the standard deviations around 0.85. The time averages and ensemble averages appear to agree which indicates ergodicity.

1.2.3

rp1

The following equation describes this process:

$$v[n] = xb \cdot sin(\frac{n\pi}{N}) + an \tag{1}$$

where $a = 0.02$, $b = 5$, x is the value that the random variable X takes with $X \sim \mathcal{U}(-0.5, 0.5)$ and N is the total number of samples. $\mathbb{E}[X] = 0$ and $Var[X] = \frac{1}{12}$.

$$\mathbb{E}[v[n]] = \mathbb{E}[xb \cdot sin(\frac{n\pi}{N}) + an] = \mathbb{E}[xb \cdot sin(\frac{n\pi}{N})] + \mathbb{E}[an] = \mathbb{E}[X] \cdot sin(\frac{n\pi}{N})b + an = 0 + an = 0.02n \quad (2)$$

$$Var(v[n]) = \mathbb{E}[v[n]^2] - \mathbb{E}[v[n]]^2 = \mathbb{E}[x^2b^2 \cdot sin^2(\frac{n\pi}{N}) + 2xban \cdot sin(\frac{n\pi}{N}) + a^2n^2] - (an)^2 \quad (3)$$

$$= \mathbb{E}[x^2b^2 \cdot sin^2(\frac{n\pi}{N})] = \mathbb{E}[x^2] \cdot b^2 sin^2(\frac{n\pi}{N}) \quad (4)$$

We know that $Var[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2 = \frac{1}{12}$, so $\mathbb{E}[X^2] = \frac{1}{12} + 0^2 = \frac{1}{12}$

$$Var(v[n]) = \frac{1}{12} \cdot 5^2 sin^2(\frac{n\pi}{N}) = \frac{25}{12} sin^2(\frac{n\pi}{N}) \quad (5)$$

The linearly increasing mean and sinusoidal standard deviation can be seen in Figure 9.

rp2

This process is described by:

$$v[n] = X \cdot Y + Z \quad (6)$$

where $X \sim \mathcal{U}(-0.5, 0.5)$ and $Y, Z \sim \mathcal{U}(0, 1)$. $\mathbb{E}[X] = 0$, $\mathbb{E}[Y] = \mathbb{E}[Z] = 0.5$ and the variance of all three is $\frac{1}{12}$.

$$\mathbb{E}[v[n]] = \mathbb{E}[XY] + \mathbb{E}[Z] = \mathbb{E}[X]\mathbb{E}[Y] + \mathbb{E}[Z] = 0.5 \quad (7)$$

$$Var(v[n]) = \mathbb{E}[v[n]^2] - \mathbb{E}[v[n]]^2 = \mathbb{E}[X^2Y^2 + 2XYZ + Z^2] - 0.5^2 = \mathbb{E}[X^2]\mathbb{E}[Y^2] + \mathbb{E}[Z^2] - 0.25 \quad (8)$$

As with rp1, $E[X^2]$ can be found using $Var[X]$, giving $E[X^2] = Var[X] + E[X]^2 = \frac{1}{12} + 0^2 = \frac{1}{12}$. $E[Y^2]$ and $E[Z^2]$ are found in a similar way with them both equalling $\frac{1}{3}$.

$$Var[v[n]] = \frac{1}{12} \cdot \frac{1}{3} + \frac{1}{3} - 0.25 = \frac{1}{9} \quad (9)$$

$$\sigma_v = \sqrt{Var[v[n]]} = \sqrt{\frac{1}{9}} = \frac{1}{3} \quad (10)$$

The theoretical mean of 0.5 and theoretical standard deviation of $\frac{1}{3}$ match what is observed in Figure 9.

rp3

The equation for this process is:

$$v[n] = mX + a \quad (11)$$

where $X \sim \mathcal{U}(-0.5, 0.5)$, $m = 3$ and $a = 0.5$. $\mathbb{E}[X] = 0$ and $Var[X] = \frac{1}{12}$.

$$\mathbb{E}[v[n]] = \mathbb{E}[mX + a] = m\mathbb{E}[X] + a = 0 + a = 0.5 \quad (12)$$

$$Var(v[n]) = \mathbb{E}[v[n]^2] - \mathbb{E}[v[n]]^2 = \mathbb{E}[m^2X^2 + 2amX + a^2] - 0.5^2 = m^2\mathbb{E}[X^2] \quad (13)$$

As was the case with rp1 and rp2, $\mathbb{E}[X^2]$ is found using the variance.

$$Var[v[n]] = m^2 \cdot (Var[X] + \mathbb{E}[X]^2) = 3^2 \cdot \frac{1}{12} = \frac{3}{4} \quad (14)$$

$$\sigma_v = \sqrt{\frac{3}{4}} \approx 0.866 \quad (15)$$

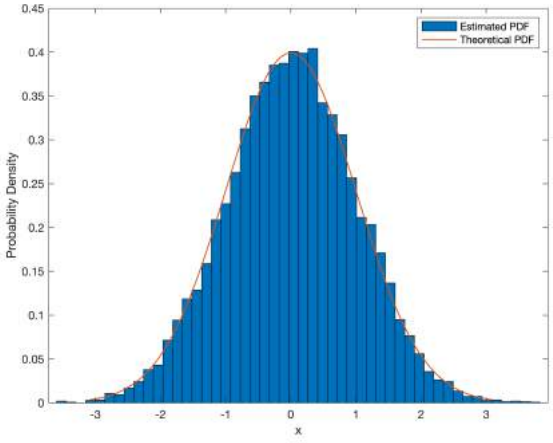
The theoretical mean and standard deviation line up with Figure 9.

1.3 Estimation of Probability Distributions

A PDF estimator file was created using the `hist` function, which has a normalisation option to generate a PDF using the input data; it ensures that the total area of the bars is 1. Figure 11 shows the code and a test of this file using 10000 data points.

```
[counts,centers]=hist(v,n);
N = length(v);
width = abs(centers(2) - centers(1));

figure
bar(centers,(counts/(N*width)),'BarWidth',1)
hold on
ylim([0 1])
legend('Estimated PDF', 'Theoretical PDF')
xlabel('x')
ylabel('Probability Density')
end
```

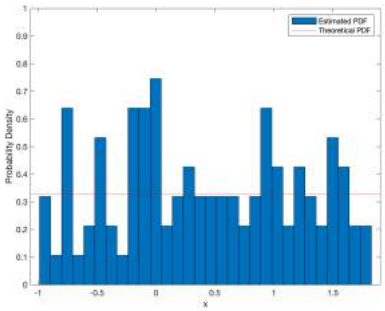
(a) Code for estimator



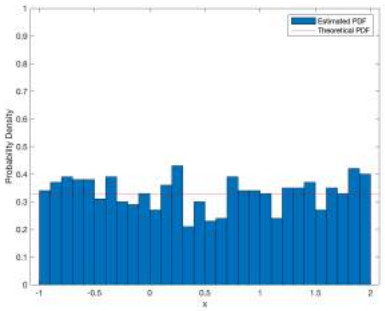
(b) Output with a Gaussian distribution as a test

Figure 11: PDF estimator code and output

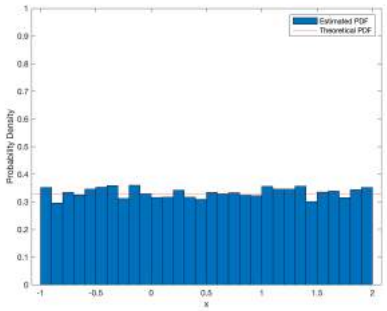
Process rp3 from section 1.2 is stationary and ergodic, so we can approximate its PDF using the file created and different numbers of data points (N). The results are shown in Figure 12.



(a) N = 100



(b) N = 1000



(c) N = 10000

Figure 12: Approximated and theoretical PDF for different $N \in \{100, 1000, 10000\}$

Increasing the number of data points used results in a more accurate approximation of the PDF of rp3 as there is less variation around the level at which the theoretical PDF lies.

Estimating the PDF of a non-stationary process in this way is not possible, because the number of data points would affect the values obtained. In the case of a 1000-sample signal whose mean changes from 0 to 1 after 500 data points, the PDF would change over time. Computing the PDF in this case would require splitting the signal where the mean changes, such that each new signal has a constant mean and is now stationary. The PDF of each can then be found using the previous method.

2 Linear Stochastic Modelling

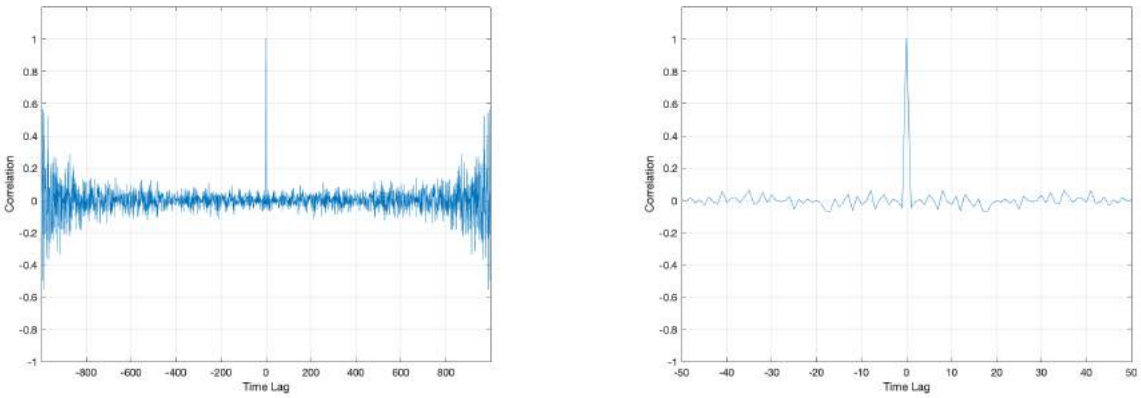
2.1 ACF of Uncorrelated and Correlated Sequences

2.1.1

The autocorrelation (ACF) estimate for a 1000-sample realisation of white Gaussian noise (WGN), $\mathbf{x}$, was calculated and is shown in Figure 13 for different τ values, where τ is time lag. Equation 16 is the formula used for the unbiased estimate of the autocorrelation function and equation 17 is the full autocorrelation function formula.

$$\hat{R}_x(\tau) = \frac{1}{N-|\tau|} \sum_{n=0}^{N-|\tau|-1} x[n]x[n+\tau], \qquad \tau = -N+1, ..., N-1 \tag{16}$$

$$R_x(\tau) = \mathbb{E}[x[n]x[n+\tau]] \tag{17}$$



(a) $\tau \in [-999 : 999]$ (b) $|\tau| < 50$

Figure 13: ACF estimates of $\mathbf{x}$ for different τ values

The ACF of $\mathbf{x}$ is symmetric around $\tau = 0$ as it is a real signal. Since the ACF calculates the correlation between a signal and itself at different time lags, the spike present at $\tau = 0$ is expected it shows the correlation between the signal and itself, in fact ideally the ACF of WGN is a Dirac Delta here and 0 elsewhere. This can be derived from the fact that WGN samples should be completely uncorrelated and random. Also, the power spectral density (PSD) of WGN is constant and since it is the Fourier Transform of the ACF, the ACF must be a Dirac Delta. The ideal behaviour of the ACF is largely present up to around $\tau = 500$, with a spike at $\tau = 0$ and the amplitude having small fluctuations around 0. Beyond this point, the fluctuations begin to increase, with the largest being at the edges. This is due to less of $\mathbf{x}$ being available for the calculation as the time lag increases, and the calculation becomes less reliable as a result.

2.1.2

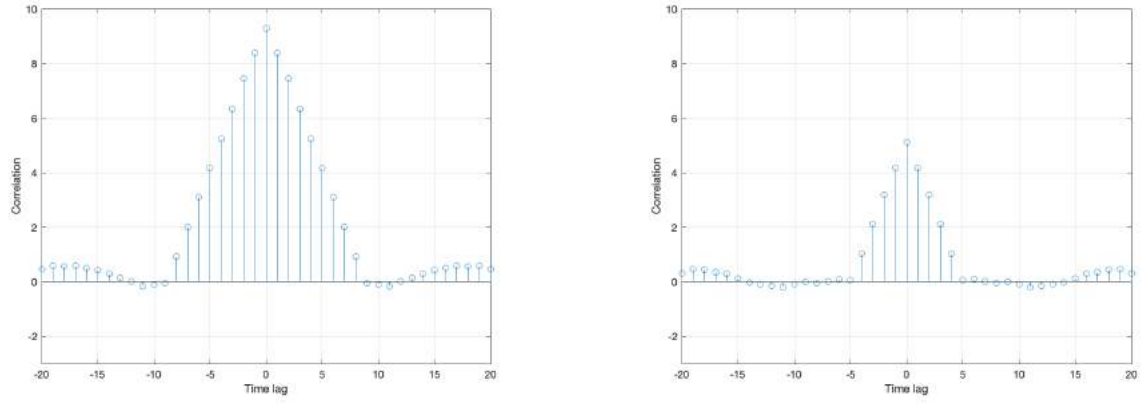
The zoomed in ACF for the range $|\tau| < 50$ in Figure 13b is closer to the ideal result, with the spike more closely resembling the Dirac Delta function at $\tau = 0$ and the oscillations elsewhere minimal; the amplitudes are within ± 0.06 of 0. This is due to at least 95% of $\mathbf{x}$ being available for this smaller range.

2.1.3

The effect of a large lag (τ) on the accuracy of the ACF estimate can be seen from equation 16. As mentioned earlier, a larger τ results in less summations since the upper bound of the sum decreases and therefore less samples are used in the calculation, resulting in greater variability in the estimate. An empirical bound on τ should strike a balance in the bias-variance trade-off; the choice of a maximum τ should balance the trade-off between capturing enough of the autocorrelation structure to reduce bias and not capturing too much of the autocorrelation structure to reduce variance. From Figure 13, a reasonable bound appears to be at $\tau = 400$, since the variance in the estimates remains small up to this point, but enough data points are used to properly capture the behaviour of the ACF estimate.

2.1.4

The results of applying a moving average (MA) filter to $\mathbf{x}$ before the ACF estimate can be seen in Figure 14.



(a) MA order 9

(b) MA order 5

Figure 14: ACF estimates of $\mathbf{x}$ after MA filters of different orders

A MA filter removes high frequency components of a signal and this results in a smoother ACF estimate, which is once again symmetric around $\tau = 0$. The ACF of a MA process should be zero after a lag equal to the order of the filter, but since this is an estimate, this is not the case. The shape of the estimate is expected to be triangular. As the input to the ACF has been MA filtered, $x[n]$ in equation 17 is of the form $a[n]+b_1a[n-1]+...+b_qa[n-q]$ and by extension $x[n+\tau]$ is of the form $a[n+\tau]+b_1a[n+\tau-1]+...+b_qa[n+\tau-q]$, where q is the order of the MA filter. After expanding and taking expectations, $R_x(\tau)$ is a sum that only depends on the coefficients explaining why ACF should be zero after $\tau > q$. The sum, being dependent on coefficients, creates the triangular shape. Intuitively, for a WGN signal, the ACF is a delta function at $\tau = 0$, which when convolved with the filter coefficients produces a triangular function. This can be seen in Figure 14 as well as the varying width as explained previously. Also, the height decreases for smaller order values. To obtain a sample mean from the ACF, we can take the value at $\tau = 0$ and divide by the order of the MA filter used.

2.1.5

Now we consider a stochastic process Y_n, which is a filtered version of the process X_n. The vector $\mathbf{y}$ is a realisation of Y_n. The ACF of Y_n, denoted by R_Y, is given by equation 18.

$$R_y(\tau) = R_x(\tau) * R_h(\tau) \tag{18}$$

where R_x is the autocorrelation of the input, R_h is the autocorrelation of the impulse response, and $*$ denotes the convolution operator.

Since X_n is stochastic, its ACF will only be non-zero at $\tau = 0$ where it will be equal to 1 and therefore R_y is the ACF of the impulse response as shown by equation 19.

$$R_y(\tau) = R_x(\tau) * R_h(\tau) = \sum_{k=-\infty}^{\infty} R_x(k)R_h(\tau - k) = R_x(0)R_h(\tau) = R_h(\tau) \tag{19}$$

2.2 Cross-correlation Function

2.2.1

Extending the idea of the ACF as shown in Section 2.1, the cross-correlation function (CCF) is defined as the expectation of delayed samples of two different stochastic processes as shown in equation 20. If the data is ergodic, an unbiased estimate for the CCF is given by equation 21.

$$R_{XY}(n, s) = \mathbb{E}[X_n Y_s] \tag{20}$$

$$\hat{R}_{XY}(\tau) = \frac{1}{N - |\tau|} \sum_{n=0}^{N-|\tau|-1} x[n]y[n+\tau], \qquad \tau = -N+1, ..., N-1 \tag{21}$$

The CCF for the sequences $\mathbf{x}$ and $\mathbf{y}$, as defined in Section 2.1, were generated and the result is shown in Figure 15.

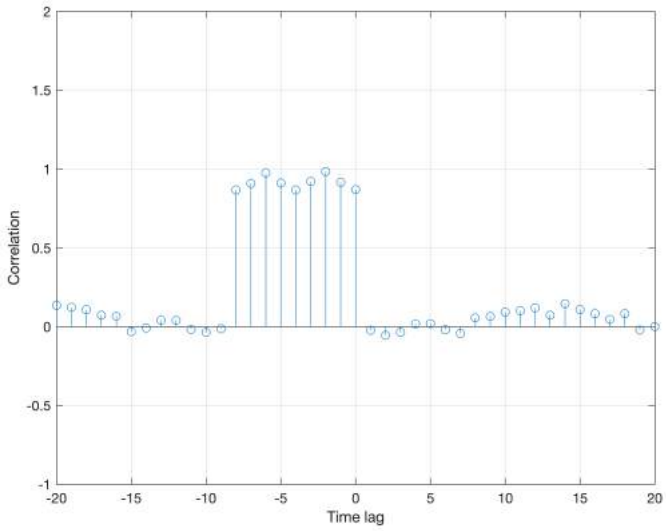


Figure 15: CCF of $\mathbf{x}$ and $\mathbf{y}$ in range $\tau \in [-20, 20]$

It can be seen that $\mathbf{x}$ is correlated to the last 8 samples of $\mathbf{y}$, which is consistent with $\mathbf{y}$ being an order 9 MA filtered version of $\mathbf{x}$. The CCF, R_{XY}, between the input and the output of a filter is given in equation 22, where $h(\tau)$ is the impulse response of the filter. If X_t is an uncorrelated stochastic process, R_{XY} can be found in a similar manner to equation 19 as $R_X(\tau) = 1$ at $\tau = 0$ and is 0 elsewhere; therefore $R_{XY}(\tau) = h(\tau)$.

$$R_{XY}(\tau) = h(\tau) * R_X(\tau) \tag{22}$$

2.2.2

This information can be used for system identification; if the input to a filter is uncorrelated and stochastic, the CCF gives the impulse response of the filter used. We can identify the order of the filter used by the number of peaks visible as the number increases or decreases with the order, for example if an order 5 MA filter was used to generate $\mathbf{y}$, Figure 15 would have only had 5 peaks. Since the impulse response completely characterises a linear time-invariant filter, this can be used to identify the system. The type of filter can also be inferred, for example if it is a high-pass, low-pass or band-pass filter.

2.3 Autoregressive Modelling

2.3.1

An autoregressive (AR) process of order 2 is given in equation 23.

$$x[n] = a_1 x[n-1] + a_2 x[n-2] + w[n], \qquad w[n] \sim \mathcal{N}(0, 1) \tag{23}$$

To show convergence of $x[n]$ depending on values of a_1 and a_2, 100 samples of each randomly distributed variable were generated, with $a_1 \in [-2.5, 2.5]$ and $a_2 \in [-1.5, 1.5]$. The result is shown in Figure 16 as a plot of a_1 and a_2 values to show what values give stability. A plot for 10000 samples is also shown to better illustrate the region of convergence of x[n].

Taking the z-transform of equation 23 gives equation 24 and rearranging to find the transfer function of the system yields equation 25.

$$X(z) = (a_1 z^{-1} + a_2 z^{-2})X(z) + W(z) \tag{24}$$

$$H(z) = \frac{X(z)}{W(z)} = \frac{1}{1 - a_1 z^{-1} - a_2 z^{-2}} = \frac{z^2}{z^2 - a_1 z - a_2} \tag{25}$$

The roots of the pole equation $1 - a_1 z^{-1} - a_2 z^{-2}$ must lie outside of the unit circle, or alternatively the roots of $z^2 - a_1 z - a_2$ must lie inside the unit circle. Taking the root of the latter using the quadratic formula:

$$|z| = \left| \frac{a_1 \pm \sqrt{a_1^2 + 4a_2}}{2} \right| < 1 \tag{26}$$

For a positive discriminant (real roots):

$$-1 < \frac{a_1 + \sqrt{a_1^2 + 4a_2}}{2} < 1 \qquad\qquad -1 < \frac{a_1 - \sqrt{a_1^2 + 4a_2}}{2} < 1 \tag{27}$$

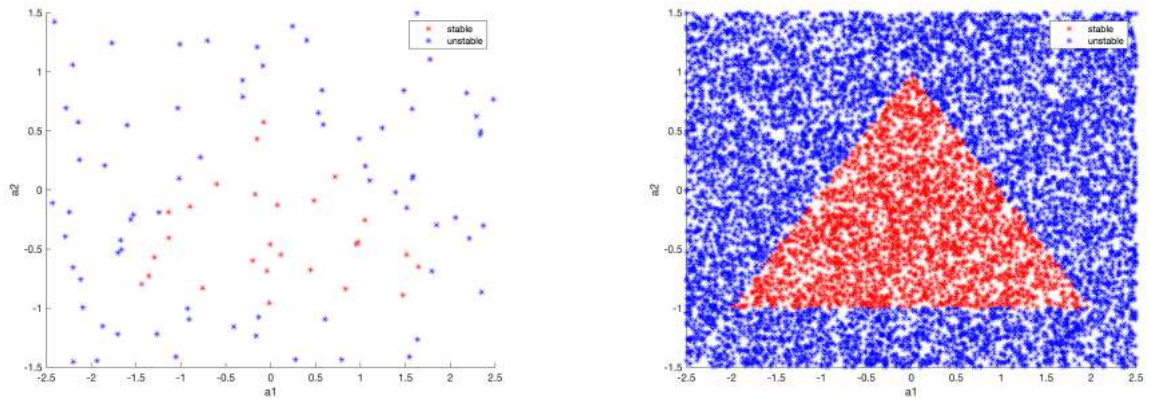
When the roots are real, the larger of the two must be less than 1, and the smaller root must exceed -1, therefore the inequalities reduce to:

$$\begin{aligned} \frac{a_1 + \sqrt{a_1^2 + 4a_2}}{2} &< 1 & \frac{a_1 - \sqrt{a_1^2 + 4a_2}}{2} &> -1 \\ \sqrt{a_1^2 + 4a_2} &< 2 - a_1 & \sqrt{a_1^2 + 4a_2} &< 2 + a_1 \\ a_1^2 + 4a_2 &< 4 - 4a_1 + a_1^2 & a_1^2 + 4a_2 &< 4 + 4a_1 + a_1^2 \\ a_1 + a_2 &< 1 & a_2 - a_1 &< 1 \end{aligned} \tag{28}$$

For a negative discriminant (complex roots) the inequality changes to:

$$\begin{aligned} \left| \frac{a_1 \pm i\sqrt{4a_2 - a_1^2}}{2} \right| &< 1 \\ \frac{a_1^2}{4} + \frac{-4a_2 - a_1^2}{4} &< 1 \\ -4a_2 &< 4 \\ a_2 &> -1 \end{aligned} \tag{29}$$

Therefore the three conditions for the convergence of $x[n]$ are: $a_1 + a_2 < 1$, $a_2 - a_1 < 1$ and $a_2 > -1$ as can be seen from Figure 16.



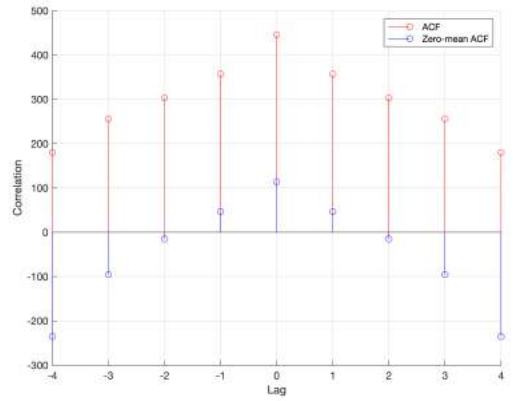
(a) N = 100

(b) N = 10000

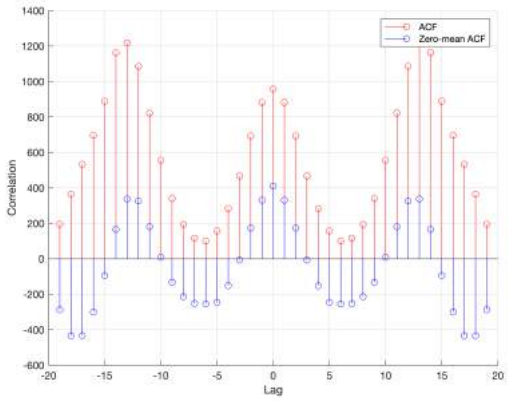
Figure 16: Stability region for AR(2) for different sample numbers (N)

2.3.2

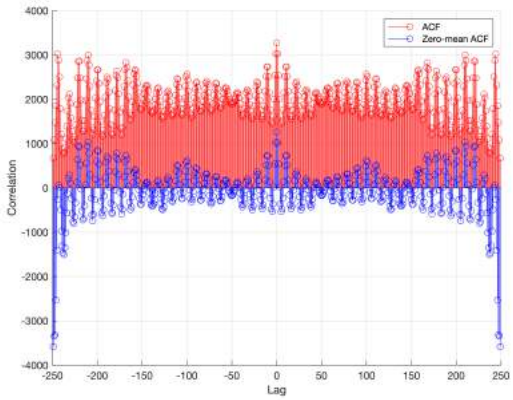
The MATLAB sunspot time series was used to generate ACFs for data lengths 5, 20 and 250 as well as the ACFs for these lengths a using zero-mean normalised time series. The results are shown in Figure 17. It is difficult to draw any meaningful conclusions about any underlying trends or patterns from the plot for N=5 since there is not enough data used. For N=20 and N=250, a periodicity is observed with a sinusoidal shape. The ACFs for the non-zero mean series are somewhat misleading, as useful information can get obscured under a DC offset, or in this case a non-zero mean. For example, there are no zero or negative values and the correlation for N=20 is shown to be lower at lag 0 than at lag 15 which does not make sense; a signal should have maximum correlation with a non-shifted version of itself. As discussed in Section 2.1, the ACF is less reliable as the lag approaches N which can be seen in the plot for N=250, so a zoomed in version is provided to avoid this. The zero-mean time series gives the true ACFs where we can see that these contain positive and negative values and the problem described earlier for N=20 is also solved.



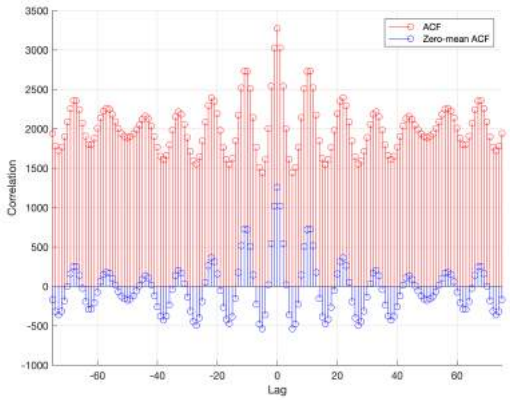
(a) N = 5



(b) N = 20



(c) N = 250



(d) N = 250 zoomed in

Figure 17: ACFs of normal and zero mean sunspot time series for different data lengths (N)

2.3.3

The Yule-Walker equations were used to generate the partial correlation function (PCF) of the sunspot time series and a zero-mean unit-variance version of this time series and the results are in Figure 18. A confidence interval of 95% is shown as green dotted lines and was calculated using $\pm\frac{1.96}{\sqrt{N}}$, where N is the sample size, in this case 288. The most likely order is 2, since the first two time lag values fall high above this interval, with the rest not as clearly outside and the value of the PCF for lags above 2 is significantly smaller. The standardised PCF is more accurate due to the removal of any offset, and it is clearer to see what the model order might be from these values.

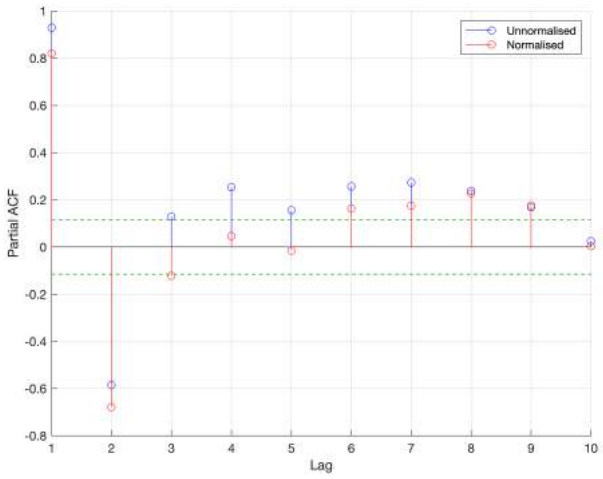


Figure 18: PCF of the sunspot time series

2.3.4

In order to correctly determine the model order, various criteria can be used such as the minimum description length (MDL), Akaike information criterion (AIC) and corrected AIC (AIC_c). These can be used to establish a trade-off between model complexity and accuracy and using the order suggested by the criteria can prevent overfitting to test/training data so that predictions made by the model can be as accurate as possible. It is worth noting that AIC_c is better suited for small sample sizes as it reduces overfitting through additional penalty term. Figure 19 is the graph of the criteria, from which we can see that a model order of 2 is ideal since it is the first minimum and a lower model order reduces overfitting.

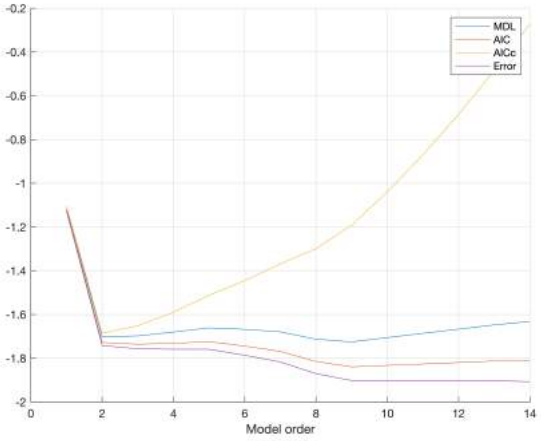
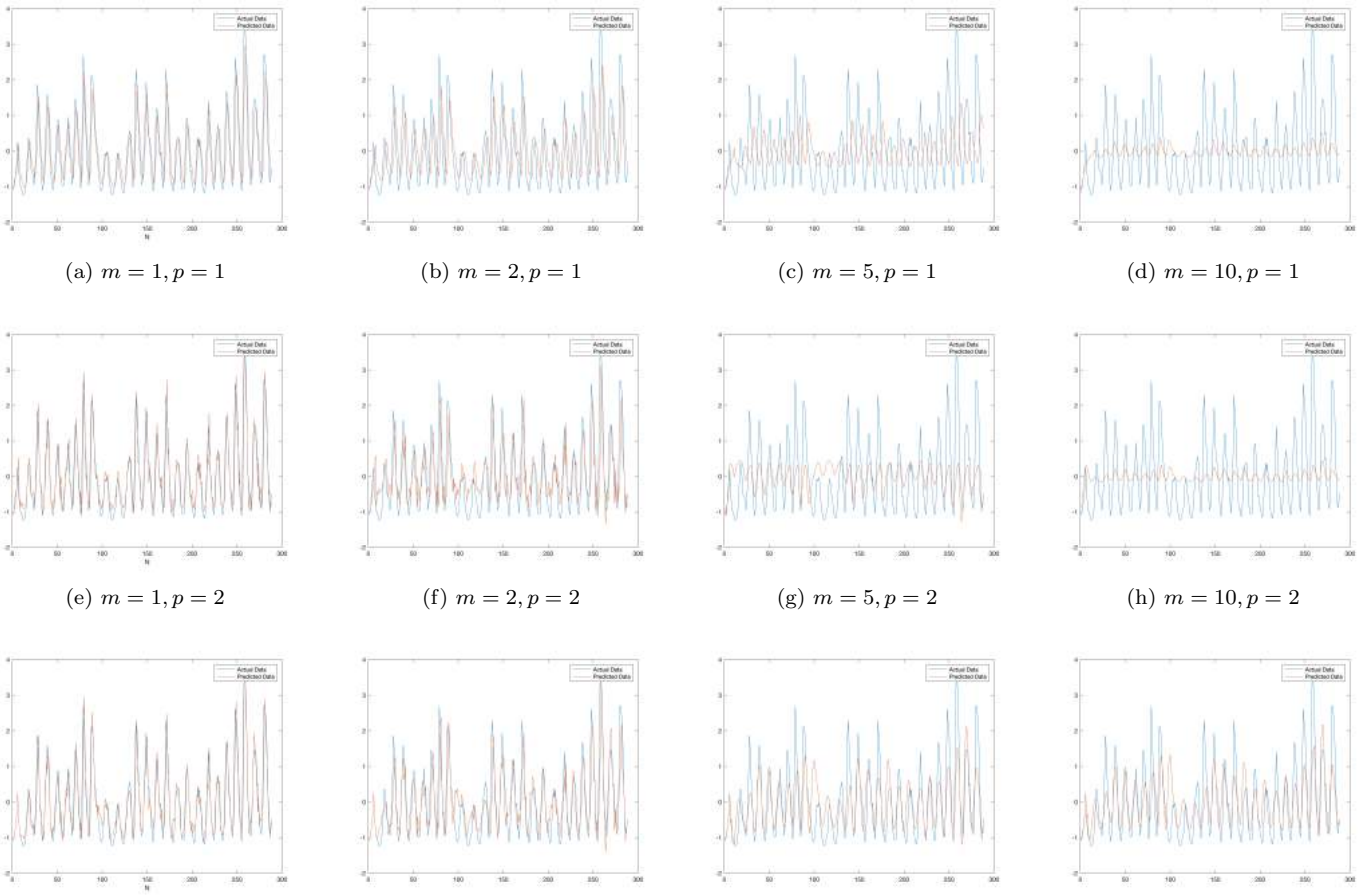


Figure 19: Criteria for choosing model order

2.3.5

Figure 20 shows a prediction of the sunspot time series for different prediction horizons, m, and for different AR orders, p with the actual data in blue and the prediction in orange. As m increases, the amplitude of prediction decreases and this is worse for the lower orders; predictions further into the future will be more accurate if the model order is higher. In general, it seems as though the horizons should roughly correspond with the model order in order to obtain an accurate prediction. A trade-off is needed between being able to predict values within the training (available) data and being able to predict future test (unknown) data. A model order that is too high will be fitted to the training data well and the model error will be low, however this usually results in degraded performance in extrapolating since it is over-parameterised and it may end up capturing noise in the data instead of the desired signal. In contrast, a model that is under-fitted (the order is too low in this case) will have poor performance in both interpolation and extrapolation as it is too simple to be able to model the data.



(a) $m = 1, p = 1$ (b) $m = 2, p = 1$ (c) $m = 5, p = 1$ (d) $m = 10, p = 1$

(e) $m = 1, p = 2$ (f) $m = 2, p = 2$ (g) $m = 5, p = 2$ (h) $m = 10, p = 2$

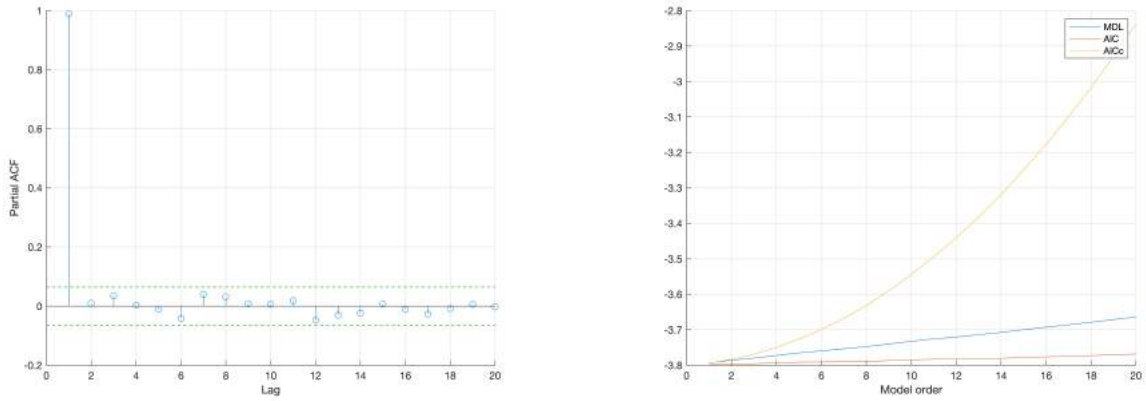
(i) $m = 1, p = 10$ (j) $m = 2, p = 10$ (k) $m = 5, p = 10$ (l) $m = 10, p = 10$

Figure 20: Prediction of sunspot time series m steps ahead and order p

2.4 Cramer-Rao Lower Bound

2.4.1

The NASDAQ financial index can be sufficiently modelled using an AR(1) model. To show this, the index from June 2003 to February 2007 was used to generate the PCF, MDL, AIC and AIC_c which were described previously and the findings are in Figure 21. In the PCF graph, the confidence interval of 95% is shown in green, and a lag of 1 is has the only value beyond this, which gives a first guess of model order 1. Moving over to the information criteria, the minimum for all three is at model order 1 and therefore we can conclude that a model order of one is sufficient for forecasting the NASDAQ.



(a) PCF for the NASDAQ

(b) MDL, AIC and AIC$_c$ for the NASDAQ

Figure 21: Various metrics used to determine a sufficient model order for the NASDAQ

2.4.2

The Cramer-Rao lower bound (CRLB) specifies a lower bound on the variance of an unbiased estimator and therefore is a target for the performance of an estimator. To compute the CRLB for the parameters of an AR(p) process, the inverse of its covariance matrix is needed which is an expensive and difficult operation. Therefore, the power spectrum of an AR(p) model is used to derive the asymptotic CRLB and the natural logarithm of this is given in equation 30 where $\boldsymbol{\theta}$ is the vector of parameters and $\hat{\sigma}^2$ is the estimated value of the noise variance.

$$ln[\hat{P}_X(f;\boldsymbol{\theta})] = ln[\hat{\sigma}^2] - ln[1 - \sum_{m=1}^{p} \hat{a}_m e^{-j2\pi f m}] - ln[1 - \sum_{m=1}^{p} \hat{a}_m e^{-j2\pi f m}] \tag{30}$$

The elements of the Fisher information matrix become:

$$[\mathbf{I}(\boldsymbol{\theta})]_{ij} = \frac{N}{2} \int_{-\frac{1}{2}}^{\frac{1}{2}} \frac{\partial ln[\hat{P}_X(f;\boldsymbol{\theta})]}{\partial \theta_i} \frac{\partial ln[\hat{P}_X(f;\boldsymbol{\theta})]}{\partial \theta_j} df \tag{31}$$

For an AR(1) process, i.e. p = 1 and $\boldsymbol{\theta} = [a_1, \sigma^2]$, $[\mathbf{I}(\boldsymbol{\theta})]_{22}$ (i=j=2) is found by doing the partial derivatives in equation 31 respect to σ^2:

$$\frac{\partial ln[\hat{P}_X(f;\boldsymbol{\theta})]}{\partial \sigma^2} = \frac{1}{\sigma^2} \tag{32}$$

$$[\mathbf{I}(\boldsymbol{\theta})]_{22} = \frac{N}{2} \int_{-\frac{1}{2}}^{\frac{1}{2}} \frac{1}{\sigma^2} \frac{1}{\sigma^2} df = \frac{N}{2\sigma^4} \tag{33}$$

Given $[\mathbf{I}(\boldsymbol{\theta})]_{11} = \frac{N r_{xx}(0)}{\sigma^2}$ and $[\mathbf{I}(\boldsymbol{\theta})]_{12} = [\mathbf{I}(\boldsymbol{\theta})]_{21} = 0$:

$$\mathbf{I}(\theta) = \begin{bmatrix} \frac{N r_{xx}(0)}{\sigma^2} & 0 \\ 0 & \frac{N}{2\sigma^4} \end{bmatrix} \tag{34}$$

2.4.3

The CRLB has the form $var(\hat{\theta}_i) \geq [\mathbf{I}^{-1}(\boldsymbol{\theta})]_{ii}$, so to get a lower bound of the variance of the parameters the inverse of $\mathbf{I}$ is needed. The inverse of a diagonal matrix is a matrix where the elements on the diagonal are replaced by their reciprocals. Therefore, $var(\hat{\sigma}^2) \geq [\mathbf{I}^{-1}(\boldsymbol{\theta})]_{22} = \frac{2\sigma^4}{N}$ and $var(\hat{a_1}) \geq [\mathbf{I}^{-1}(\boldsymbol{\theta})]_{11} = \frac{\sigma^2}{N r_{xx}(0)}$. The autocorrelation of the AR(1) process at lag 0 is:

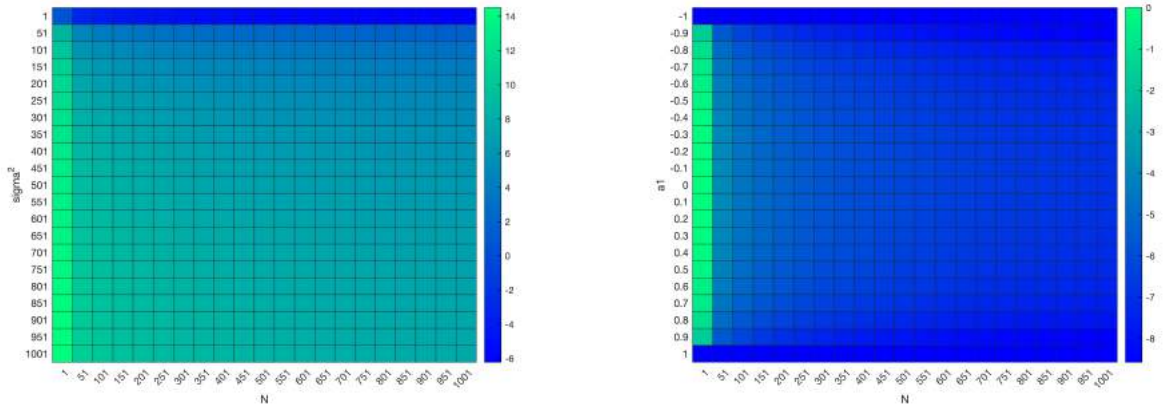
$$r_{xx}(0) = \mathbb{E}[x[n]x[n]] = \mathbb{E}[(a_1 x[n-1] + w[n])^2] = \mathbb{E}[a_1^2 x^2[n-1] + 2w[n]a_1 x[n-1] + w[n]^2] \tag{35}$$

$$= \mathbb{E}[a_1^2 x^2[n-1]] + \mathbb{E}[w[n]^2] = a_1^2 \mathbb{E}[x^2[n-1]] + \sigma^2 = a_1^2 \mathbb{E}[x^2[n]] + \sigma^2 = a_1^2 r_{xx}(0) + \sigma^2 \tag{36}$$

$$rxx(0)(1-a_1^2) = \sigma^2 \Rightarrow rxx(0) = \frac{\sigma^2}{1-a_1^2} \tag{37}$$

Since $r_{xx}(0) = \frac{\sigma^2}{1-a_1^2}$, $var(\hat{a_1}) \geq \frac{1}{N}(1-a_1^2)$. Note that for equation 36 the AR(1) process being stationary for $-1 < a_1 < 1$ is needed.

Heat maps for the CRLB $\hat{\sigma^2}$ and $\hat{a_1}$ were plotted, using a range of 1 to 1001 in increments of 50 for N and σ^2, and a range of -1 to 1 in increments of 0.1 for a_1. The variance of each parameter decreases as N increases, as σ^2 decreases so does the CRLB of $\hat{\sigma^2}$ and as the square of a_1 increases, the CRLB of $\hat{a_1}$ decreases.



(a) $\hat{\sigma^2}$ (b) $\hat{a_1}$

Figure 22: Heat maps for $\hat{\sigma^2}$ and $\hat{a_1}$

Calculating $r_{xx}[0]$ from the NASDAQ data using the Yule-Walker equations yields an a_1 value of 0.9887 and the time-series length is 924 and therefore $var(\hat{a_1}) \geq \frac{1-0.9887^2}{924} = 2.432 \times 10^{-5}$. As a value of a_1 approaches unity, the variance of $\hat{a_1}$ decreases, but if the value is or exceeds 1, the system becomes unstable. The transfer function for an AR(1) process is $\frac{1}{1-a_1 z^{-1}} = \frac{z}{z-a_1}$, and for stability in the z-domain, all poles must lie within the unit circle if the system is causal. Thus, when $a_1 \geq |1|$, there is instability.

The power spectrum of an AR(1) model is given by:

$$\hat{P}_X(f;\boldsymbol{\theta}) = \frac{\hat{\sigma^2}}{|1-\hat{a_1}e^{-j2\pi f}|^2} = \frac{\hat{\sigma^2}}{|A(f)|^2} \tag{38}$$

$$\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial a_1} = \frac{-\hat{\sigma}^2(2a_1 - e^{j2\pi f} - e^{-j2\pi f})}{|A(f)|^4} = \frac{\hat{\sigma^2}(A(f)e^{j2\pi f} + e^{-j2\pi f}A^*(f))}{|A(f)|^4} \tag{39}$$

$$\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \sigma^2} = \frac{1}{|A(f)|^2} \tag{40}$$

2.4.4

It can be shown that:

$$var(\hat{P}_X(f;\boldsymbol{\theta})) \geq \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}^T \mathbf{I}^{-1}(\boldsymbol{\theta}) \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}, \qquad \boldsymbol{\theta} = \left[\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \theta_1}, \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \theta_2}\right] \tag{41}$$

$$\therefore var(\hat{P}_X(f;\boldsymbol{\theta})) \geq \left[\frac{\hat{\sigma^2}(A(f)e^{j2\pi f} + e^{-j2\pi f}A^*(f))}{|A(f)|^4} \quad \frac{1}{|A(f)|^2}\right] \begin{bmatrix} \frac{1-a_1^2}{N} & 0 \\ 0 & \frac{2\sigma^4}{N} \end{bmatrix} \begin{bmatrix} \frac{\hat{\sigma^2}(A(f)e^{j2\pi f} + e^{-j2\pi f}A^*(f))}{|A(f)|^4} \\ \frac{1}{|A(f)|^2} \end{bmatrix} \tag{42}$$

$$var(\hat{P}_X(f;\boldsymbol{\theta})) \geq \left[\frac{\hat{\sigma^2}(1-a_1^2)(A(f)e^{j2\pi f} + e^{-j2\pi f}A^*(f))}{N|A(f)|^4} \quad \frac{2\sigma^4}{N|A(f)|^2}\right] \begin{bmatrix} \frac{\hat{\sigma^2}(A(f)e^{j2\pi f} + e^{-j2\pi f}A^*(f))}{|A(f)|^4} \\ \frac{1}{|A(f)|^2} \end{bmatrix} \tag{43}$$

$$var(\hat{P}_X(f;\boldsymbol{\theta})) \geq \frac{\hat{\sigma^4}(1-a_1^2)(A(f)e^{j2\pi f}+e^{-j2\pi f}A^*(f))^2}{N|A(f)|^8} + \frac{2\sigma^4}{N|A(f)|^4} \tag{44}$$

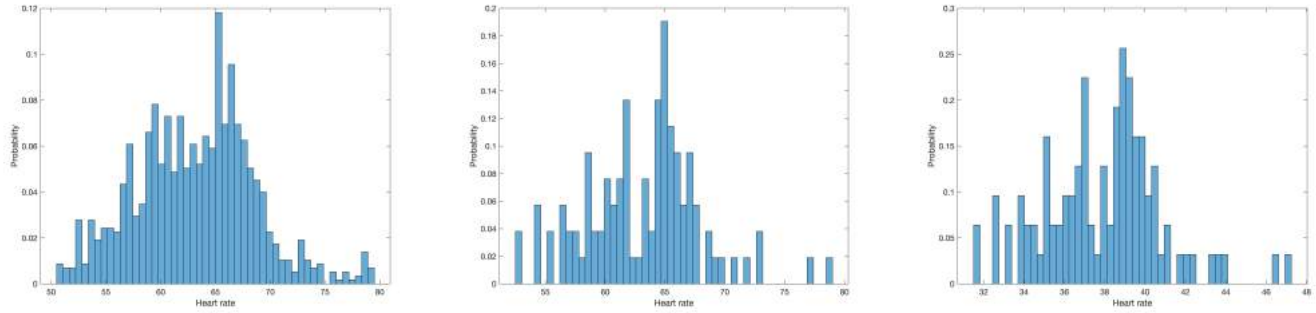
2.5 ECG from iAmp Experiment

2.5.1 Heart rate probability density estimate

The ECG signal obtained was converted into an RR interval (RRI) signal ($rr[n]$), from which the heart rate, $h[n]$, and averaged heart rate (averaging 10 consecutive samples), $\hat{h}[n]$, can be found as shown in equation 45 where α is a constant.

$$h[n] = \frac{60}{rr[n]} \qquad \hat{h}[n] = \frac{1}{10}\sum_{i=10n-9}^{10n} \alpha h[i] \tag{45}$$

Figure 23 shows the probability density estimate (PDE) for $h[n]$, and $\hat{h}[n]$ for $\alpha = 1$ and 0.6. The variance of the $\hat{h}[n]$ plots is lower since averaging the heart rate will help to mitigate some of the noise. If $var(h[n]) = \sigma^2$, then $var(\hat{h}[n]) = \frac{\alpha^2}{10^2} var(\sum_{i=10n-9}^{10n} h[i]) = \frac{\alpha^2\sigma^2}{10}$; the difference in variance due to α values can be seen in Figure 23(b) and (c) as the data spans less values for $\alpha = 0.6$. The heart rates for $\alpha = 0.6$ are shifted since this value introduces a bias to $\hat{h}[n]$ and its PDE.

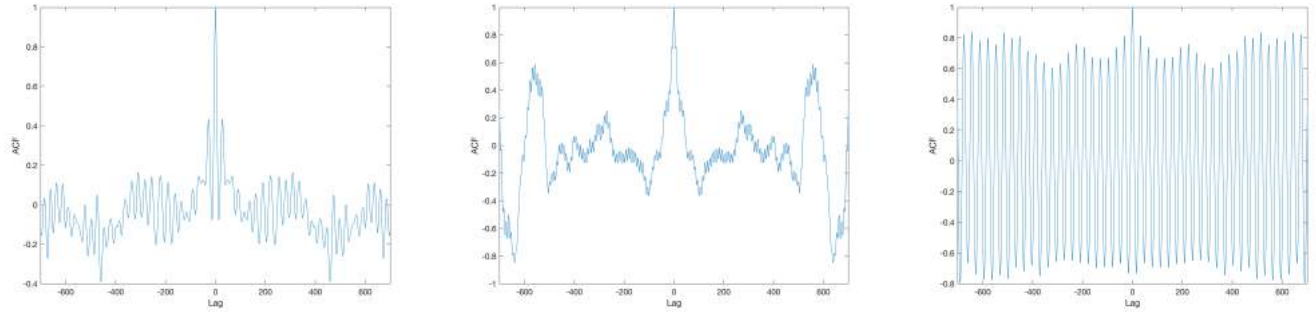


(a) $h[n]$ (b) $\hat{h}[n], \alpha = 1$ (c) $\hat{h}[n], \alpha = 0.6$

Figure 23: PDEs for $h[n]$ and $\hat{h}[n]$

2.5.2 AR modelling of heart rate

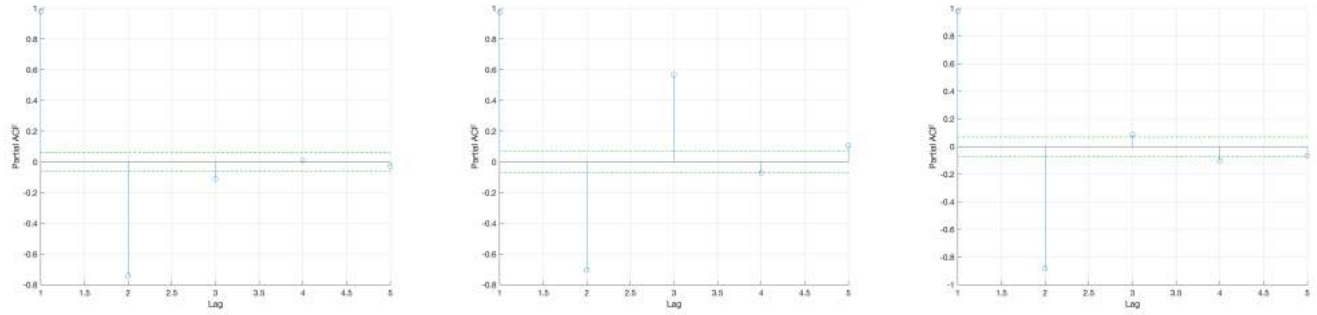
The RRI signal contains 3 trials for unconstrained breathing, constrained breathing at 50 beats per minute and constrained breathing at 15 beats per minute. To see whether they can be modelled by an AR or MA model, the ACF of each is shown in Figure 24. The RRI can be modelled by an AR process as the ACFs do not cut off after a particular lag p, which would happen with a MA(p) process, instead they display a decaying sinusoidal shape which is typical of an AR process. There is strange behaviour as the lag grows larger for the reasons explained in section 2.1 (the length of the RRI data is around 800).



(a) Trial 1 (b) Trial 2 (c) Trial 3

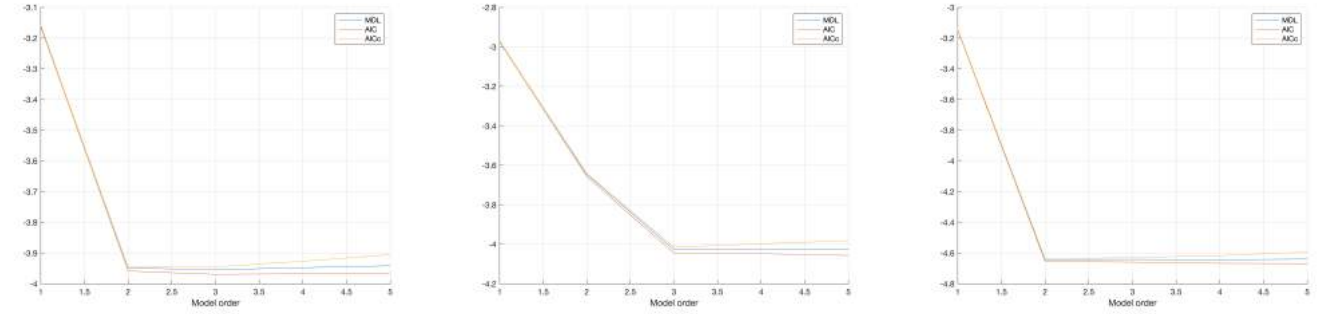
Figure 24: ACFs for the 3 trials

To find the correct AR model order, the normalised PACF (95% confidence interval in green), MDL, AIC and AICc are used:



(a) Trial 1 (b) Trial 2 (c) Trial 3

Figure 25: PACFs for the 3 trials



(a) Trial 1 (b) Trial 2 (c) Trial 3

Figure 26: Information criteria for the 3 trials

For trial 1, the minimum for the criteria is at order 2. Also the PACF sharply drops after 2, although 3 is not quite past the confidence region but due to the region being somewhat arbitrary and the criteria clearly showing 2 is the best order, an AR(2) model is most appropriate here. The PACF values and criteria minimums indicate that an AR(3) model would be best in trial 2. Similarly to trial 1, the PACF for trial 3 has values close to the confidence region lines, but the sharp drop after 2 and the criteria minimum being at order 2 means that this trial shows that an AR(2) model should be chosen. Taking all 3 trials into account, overall an AR(2) model would most likely be optimal for modelling the RRI series (if you do not know the breathing information that we have from the trials), otherwise each trial has shown the best model for that situation.

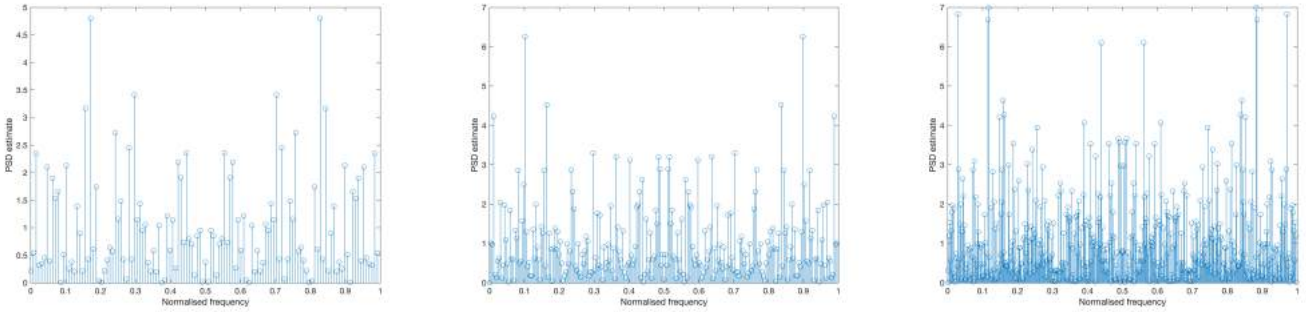
3 Spectral Estimation and Modelling

The power spectral density (PSD), $P_x(f)$, of an ergodic stochastic process X_n describes the distribution of power of the frequency components contained in the process. The Wiener–Khintchine allows the computation of the PSD of X_n using its autocorrelation function. In practise the ACF might not be known so an estimate of the PSD known as the periodogram (based on the Fast Fourier Transform) is used instead, defined in equation 46 where N is the number of discrete frequencies, f is normalised frequency and the sampling interval is assumed to be 1.

$$\hat{P}_x(f) = \frac{1}{N}\left|\sum_{n=0}^{N-1} x[n]e^{-j2\pi f\frac{n}{N}}\right|^2 \tag{46}$$

A function to calculate the periodogram of an input signal was created and tested using an N-sample realisation of WGN, denoted x for $N = \{128, 256, 512\}$. The results are shown in Figure 27. The ideal PSD for WGN is a straight line at a y-level of 1 since it should contain an equal amount of all frequencies, but since x is a finite

realisation of WGN, this is not the case in the plots. Also, as N increases, the variance does not drop indicating that this is not a consistent estimator, even though the periodogram is asymptotically unbiased.



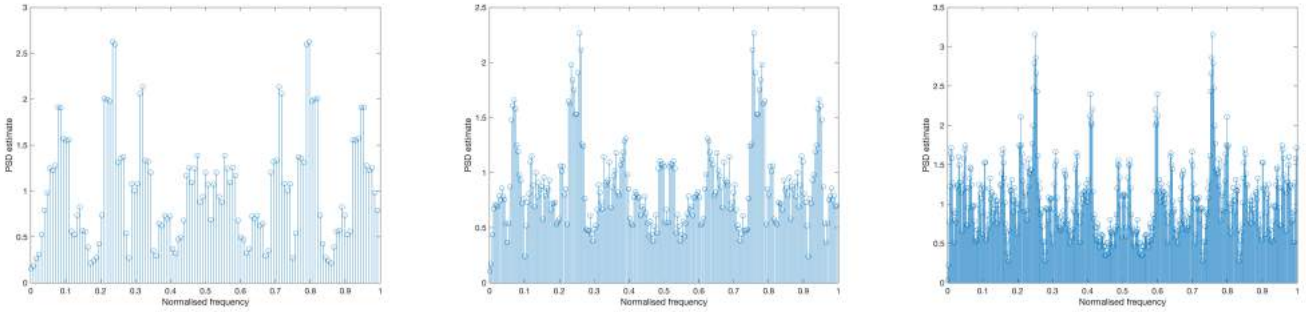
(a) $N = 128$ (b) $N = 256$ (c) $N = 512$

Figure 27: Periodograms of x for the N values

3.1 Averaged Periodogram Estimates

3.1.1

The periodogram can be smoothed using a zero-phase FIR filter with coefficients $[0.2, 0.2, 0.2, 0.2, 0.2]$, which will filter out high frequency components, and since it also is a moving average filter the variance drops. For $Var(\hat{P}_x[n]) = \sigma^2$, then $Var(\hat{P}_x^{smooth}[n]) = Var(0.2\sum_{i=1}^{5}\hat{P}_x[i]) = \frac{\sigma^2}{5}$; this reduced variance can be seen in Figure 28.



(a) $N = 128$ (b) $N = 256$ (c) $N = 512$

Figure 28: Smoothed periodograms for the N values

3.1.2

The periodograms of 8 non-overlapping, equally sized segments of a 1024-sample realisation of WGN were generated, and there was no particular pattern to the differences; there seemed to be random variation between them as can been seen from the mean and variance of each shown in Table 1, and Figure 29. The periodogram appears to be an unreliable estimate of the PSD.

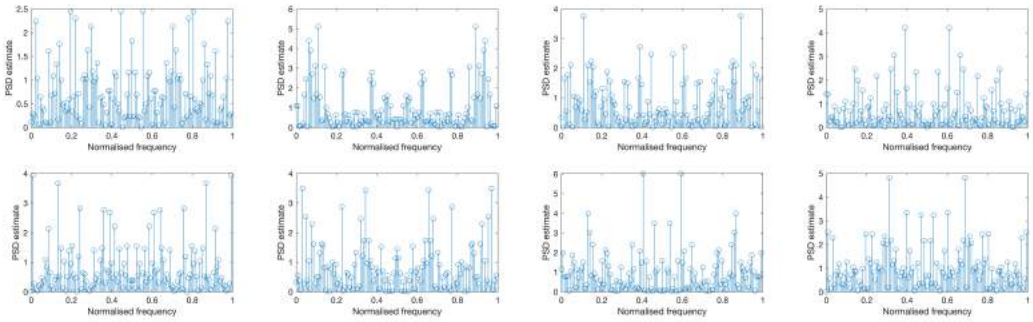


Figure 29: Periodogram of all 8 segments

Segment	1	2	3	4	5	6	7	8
Mean	0.7584	1.0455	0.8517	0.7562	0.8303	0.8455	1.0186	1.0769
Variance	0.4022	1.3400	0.6264	0.6725	0.7480	0.6657	1.1500	0.8433

Table 1: Mean and variance of each segment

3.1.3

Taking the average of the 8 periodograms results in a lower variance and a mean closer to the true value of 1, shown in Figure 30. Letting the individual periodograms be $\hat{P}^{(i)}_{x(ind)}(f)$, the average of the periodograms is $\hat{P}_{x(avg)}(f) = \frac{1}{L}\sum_{i=1}^{L}\hat{P}^{(i)}_{x(ind)}(f)$, then $var(\hat{P}_{x(avg)}(f)) = \frac{1}{L}var(\hat{P}^{(i)}_{x(ind)}(f)) = \frac{1}{L}\sigma^2$, where L is the number of segments. Thus the variance decreases with the number of segments which aligns with the fact that the variance of the averaged periodogram is 0.1281 which is a significant improvement from the values in Table 1 and it is a better estimate of the PSD. Since the variance now approaches 0 as $L \to \infty$, this is now a consistent estimator.

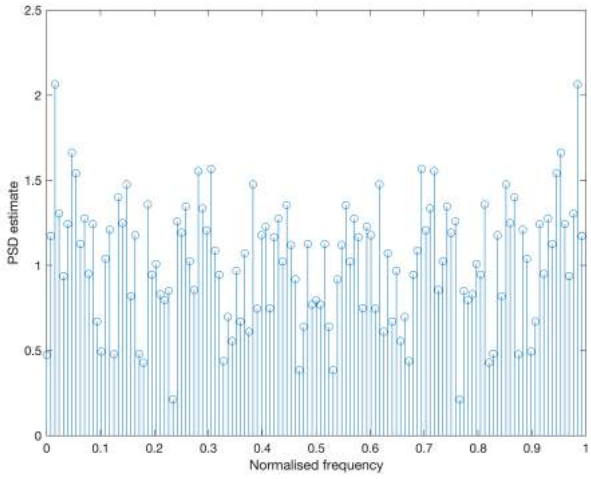
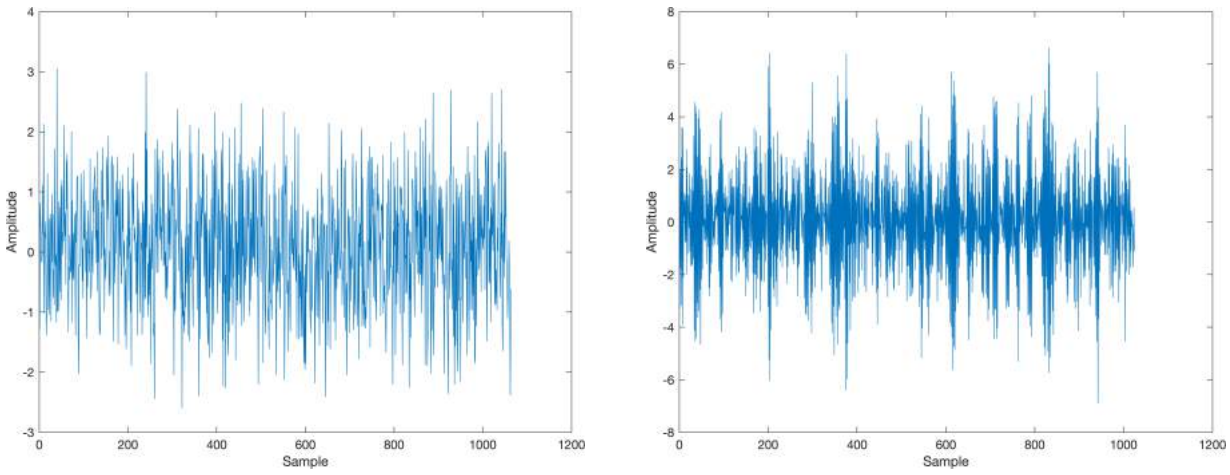


Figure 30: Averaged Periodogram

3.2 Spectrum of Autoregressive Processes

Figure 31 shows the time domain plots for a 1064-sample WGN sequence, x, and a filtered version y which is generated through an AR(1) model. The filter used was high pass and so the low frequency components of x have been attenuated to create y, as can be seen from the plots.

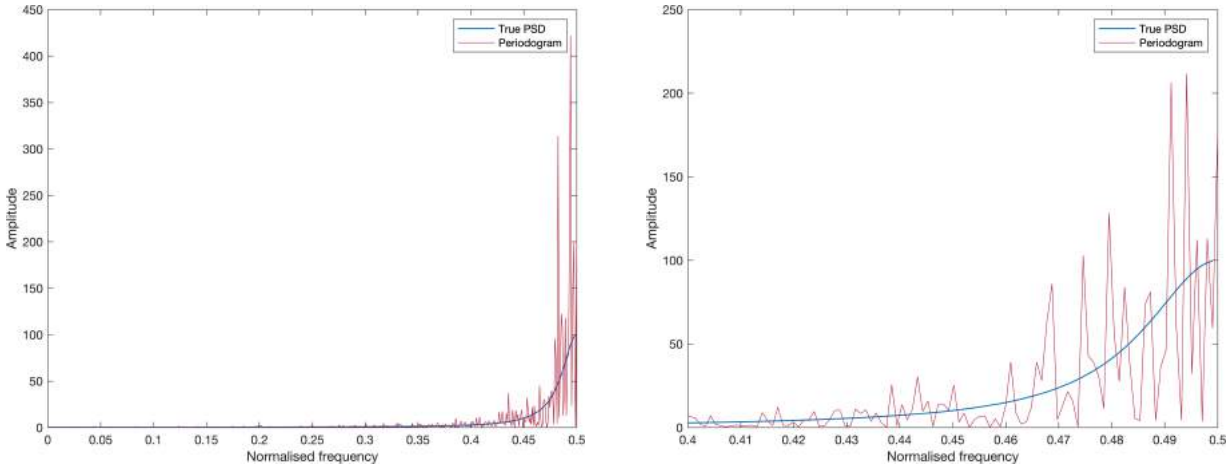


(a) x (b) y

Figure 31: Time domain plots for x and y

3.2.1 , 3.2.2 & 3.2.3

The PSD and periodogram of y are shown in Figure 32. The true PSD displays the behaviour expected due to y being high-pass filtered where there is a greater emphasis on the high frequencies, and the cutoff frequency is around 0.48Hz. The periodogram displays variance around the true values since it is an estimate of the PSD, with greater fluctuations around the high frequencies. There is underlying rectangular windowing in the periodogram estimator since a finite number of samples of WGN are used, so x is technically a windowed version of an infinite length process, i.e. $x(n) = x_{infinite}(n) \times w(n)$. Taking the Fourier transform of x results in a convolution between the infinite signal and window, which becomes the sinc function the frequency domain. The broad envelope of the side lobes of sinc spreads the power of the signal across nearby frequencies, causing spectral leakage, which can be seen more clearly in (b).



(a) Interval $f = [0, 0.5]$ (b) Zoomed in for interval $f = [0.4, 0.5]$

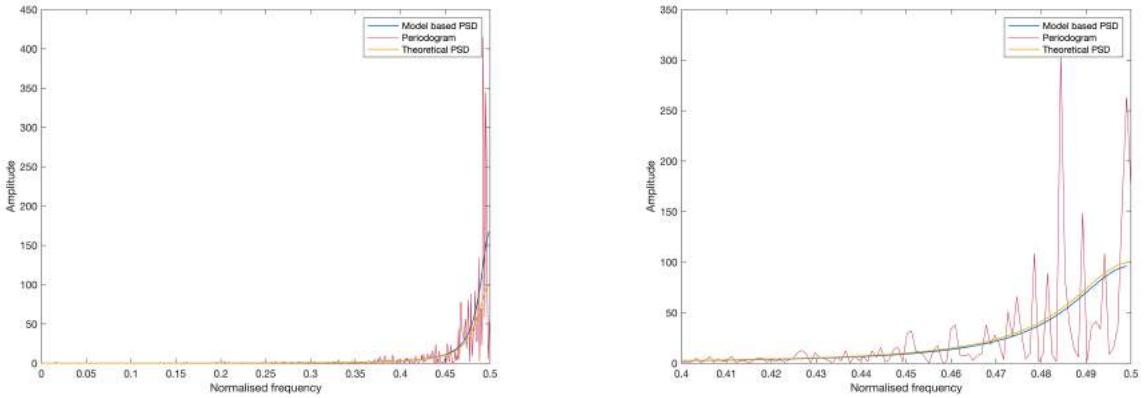
Figure 32: PSD and periodogram of y

3.2.4

An alternative method of finding the PSD is to use a model-based method, where the parameters of the model are estimated and the PSD calculated through those. For the AR(1) process used in this section, the two parameters are a_1 and σ_X^2. The estimates for these are found through the ACF of Y (equation 47) and the model based PDF estimate is found using equation 48. The results are shown in Figure 33.

$$\hat{a}_1 = -\frac{\hat{R}_Y(1)}{\hat{R}_Y(0)} \qquad \hat{\sigma}_X^2 = \hat{R}_Y(0) + \hat{a}_1 \hat{R}_Y(1) \tag{47}$$

$$\hat{P}_y(f) = \frac{\hat{\sigma}_X^2}{|1 + \hat{a}_1 e^{-j2\pi f}|^2} \tag{48}$$



(a) Interval $f = [0, 0.5]$

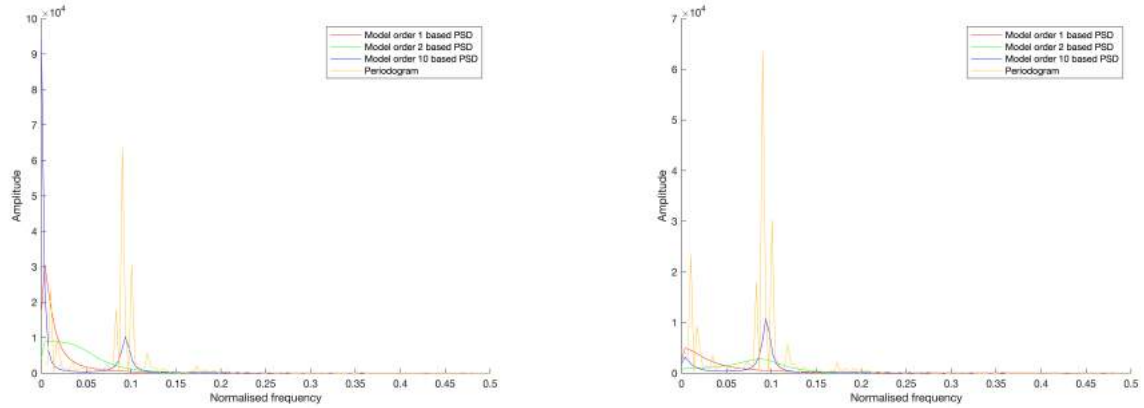
(b) Zoomed in for interval $f = [0.4, 0.5]$

Figure 33: Model based PSD, theoretical PSD and periodogram of y

The model based PSD is significantly closer to the theoretical PSD, compared to the periodogram. This is due to this method being able to exploit knowledge about the process behind the data to estimate parameters to calculate the PSD estimate, whereas the periodogram is a blanket method to estimate the PSD for all data. The specific parameters of a model based PSD result in a more accurate estimation of the PSD, which can be seen more clearly in Figure 33(b) where it does not randomly oscillate around the true value; instead it remains close to theoretical PSD.

3.2.5

Applying a similar method to the sunspot time series for AR orders of 1, 2 and 10 like in Section 2.3 yields the results in Figure 34. The optimal order found in that section from the information criteria was 2, and the same can be seen here, especially in the zero-mean time series plots in (b). Order 1 is too simple and not expressive enough to identify the dominant frequencies whereas orders 2 and 10 are, with 10 giving a larger peak and following the periodogram the closest. However, since a smaller model order is preferred for the reasons outlined previously, order 2 should be chosen over 10. Undermodelling will result in not all the information being displayed in the PSD, and overmodelling may try and reveal information that is not there, for example trying to model the noise, and whilst not visible in the graphs, spectral line splitting may occur. The periodogram identifies the dominant frequencies, albeit with a lot more variation, so the maximum values in spiking regions should be taken.



(a) Unmodified time series

(b) 0 mean time series

Figure 34: Model based PSDs and periodogram for the sunspot time series

3.3 The Least Squares Estimation (LSE) of AR Coefficients

3.3.1

The biased ACF of an AR(p) process can be calculated as:

$$\hat{r}_{xx}[k] = \frac{1}{N}\sum_{n=0}^{N-1-k} x[n]x[n+k] = \sum_{i=1}^{p} a_i \hat{r}_{xx}[k-i] + \epsilon[k], \qquad \text{for } i \geq 1 \tag{49}$$

where $\epsilon[n]$ is the error due to the effect of errors the ACF estimate. The least squares (LS) cost function for finding the AR coefficients, $\mathbf{a}$, is:

$$J = \sum_{k=1}^{M}\left[\hat{r}_{xx}[k] - \sum_{i=1}^{p} a_i \hat{r}_{xx}[k-i]\right]^2, \qquad \text{for } M \geq p \tag{50}$$

This cost function has general form $(\mathbf{x}-\mathbf{Ha})^T(\mathbf{x}-\mathbf{Ha})$, where $\mathbf{a} = [a_1, a_2, ..., a_p]$. A matrix or vector multiplied by its transpose is the equivalent of squaring it, so rewriting the contents of equation 50 in the form $(\mathbf{x}-\mathbf{Ha})$ yields:

$$\begin{bmatrix} \hat{r}_{xx}[1] \\ \hat{r}_{xx}[2] \\ \vdots \\ \hat{r}_{xx}[M] \end{bmatrix} - \begin{bmatrix} \hat{r}_{xx}[0] & \hat{r}_{xx}[-1] & \cdots & \hat{r}_{xx}[1-p] \\ \hat{r}_{xx}[1] & \hat{r}_{xx}[0] & \cdots & \hat{r}_{xx}[2-p] \\ \vdots & \vdots & \ddots & \vdots \\ \hat{r}_{xx}[M-1] & \hat{r}_{xx}[M-2] & \cdots & \hat{r}_{xx}[M-p] \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} \tag{51}$$

The signal model $\mathbf{s}$ is the multiplication of $\mathbf{H}$ and $\mathbf{a}$. To find the LS estimates for $\mathbf{a}$, the cost function needs to be minimised. Expanding it in terms of $(\mathbf{x}-\mathbf{Ha})$ gives $J = \mathbf{x}^T\mathbf{x} - 2\mathbf{x}^T\mathbf{Ha} + \mathbf{a}^T\mathbf{H}^T\mathbf{Ha}$. Setting the derivative to 0 and solving for a:

$$\frac{\partial J}{\partial \mathbf{a}} = -2\mathbf{H}^T\mathbf{x} + 2\mathbf{H}^T\mathbf{Ha} = 0 \tag{52}$$

$$\hat{\mathbf{a}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \tag{53}$$

The estimate for $\mathbf{a}$ in equation 53 is known as the least squared modified Yule-Walker equations. The actual Yule-Walker equation for the coefficients is $\mathbf{r}_{xx} = \mathbf{R}_{xx}\mathbf{a}$, and solving for $\mathbf{a}$: $\mathbf{a} = \mathbf{R}_{xx}^{-1}\mathbf{r}_{xx}$. $\mathbf{R}_{xx}$ is of similar form to $\mathbf{H}$ and so $\hat{\mathbf{a}}$ is a LS estimate of $\mathbf{a}$ where a LSE is used for the estimated ACF rather than the original data.

3.3.2

The observation matrix, H, is a stochastic matrix due to the fact that it is affected by the error; H uses $\hat{r}_{xx}$ which contains an error as shown in the biased ACF equation. This error could also be attributed to noise.

3.3.3

Applying the LSE approach to the sunspot series gave the coefficients in Table 2.

Model Order	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
1	-0.821	-	-	-	-	-	-	-	-	-
2	-1.378	0.678	-	-	-	-	-	-	-	-
3	-1.295	0.510	0.122	-	-	-	-	-	-	-
4	-1.301	0.486	0.184	-0.047	-	-	-	-	-	-
5	-1.302	0.488	0.191	-0.068	0.016	-	-	-	-	-
6	-1.304	0.499	0.160	-0.147	0.227	-0.162	-	-	-	-
7	-1.276	0.460	0.186	-0.175	0.139	0.066	-0.175	-	-	-
8	-1.236	0.445	0.154	-0.135	0.097	-0.039	0.115	-0.228	-	-
9	-1.196	0.424	0.161	-0.152	0.121	-0.066	0.037	-0.009	-0.177	-
10	-1.195	0.424	0.161	-0.152	0.121	-0.065	0.036	-0.011	-0.172	-0.004

Table 2: LSE estimated coefficients for different model orders

3.3.4

The mean squared error (MSE) between the model order and data is shown in Figure 35, where it can be seen that there is a sharp drop at model order 2. Comparing this to the PCF in Section 2.3 where the value drops after 2, the two graphs seem to corroborate that the model order should be around 2. The information criteria from that section also agrees that order 2 is best; in Figure 35, the MSE drops a small amount as the model order grows, but as seen earlier the increase in complexity is not worth the risk in overfitting. The LSE estimate reaches coefficient values that are close to the true values through minimising the loss function (error).

3.3.5

Figure 36 shows the power spectra of the AR(p) model from orders 1 to 5. Model order 1 is incapable of modelling the data, as shown by the fact that it does not have a peak to show a dominant frequency. Orders 2 to 5 do show the frequency peak in the data, but all have peaks high enough to sufficiently model the data. Therefore, this again shows that a model order of 2 is sufficient because it will be able to describe the underlying pattern in the data, without being needlessly complicated. The peaks are also in the same place as Figure 34, where the model based PSD is shown for the sunspot time series.

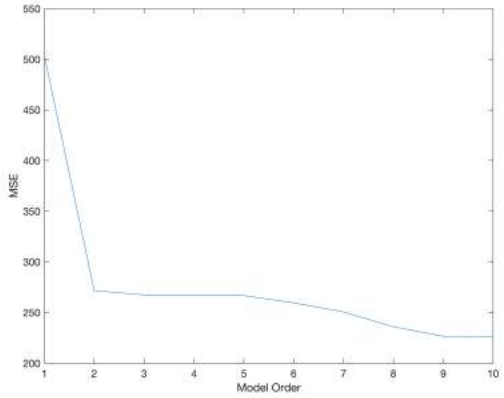


Figure 35: Approximation error of AR models

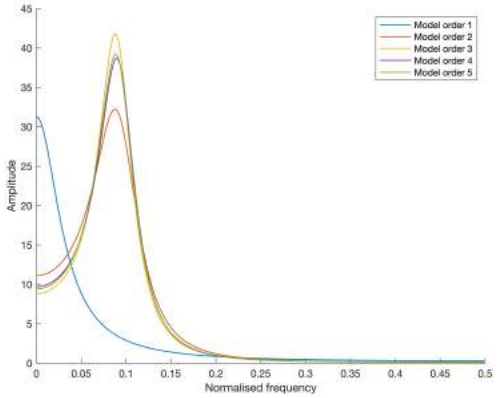


Figure 36: Power spectra of AR models

3.3.6

The MSE for different data lengths, N, is plotted in Figure 37 where it can been seen that the model performs best at a value of around 20. The decrease in performance as N increases may be down to overfitting, as well as the fact that the data may not be entirely stationary which would mean that the model would not hold after a certain point since an AR model assumes stationarity.

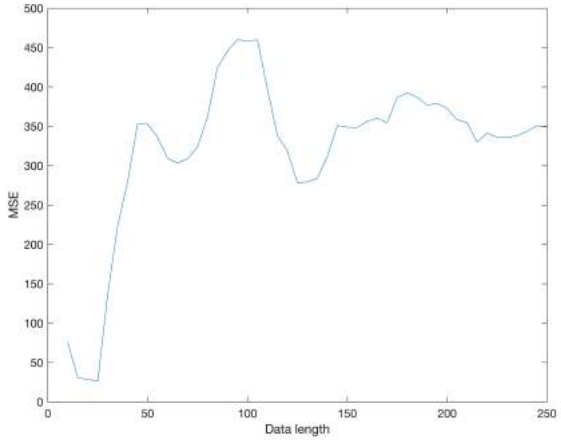
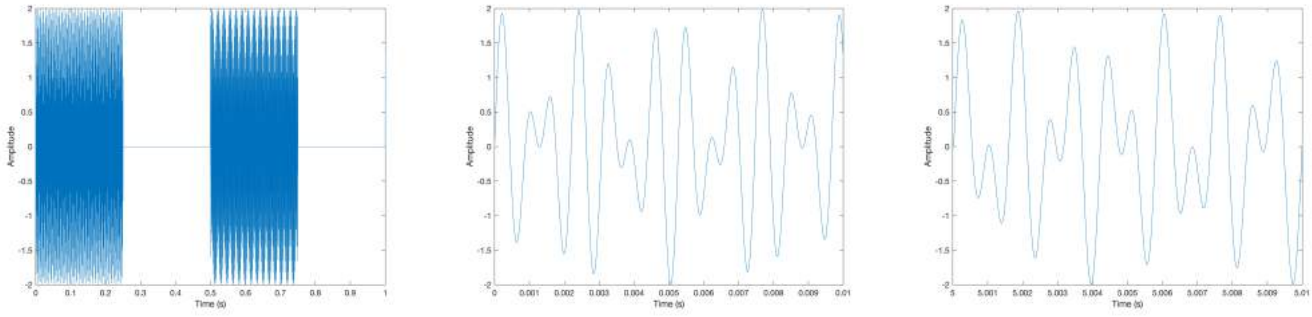


Figure 37: MSE for different N values

3.4 Spectrogram for Time-Frequency Analysis: Dial Tone Pad

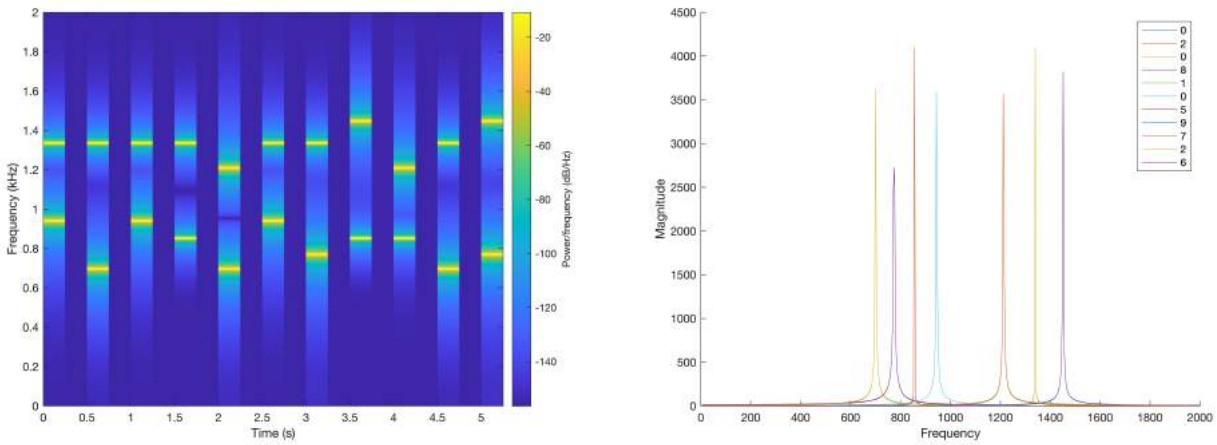
3.4.1

A London landline number in the format 020 XXXX XXXX was randomly generated, with the relevant frequencies of the tone presses for each digit. The tone is of the form $y[n] = sin(2\pi f_1 n) + sin(2\pi f_2 n)$, where f_1 and f_2 are the frequencies in Hz and n is the time index. The signal for keys 0 and 2 is in Figure 38, where the 0.25 second duration for the tone, followed by the 0.25 second gap can be seen. The maximum frequency of any of the tones 1477Hz, so the sampling frequency of 32768Hz is well above the Nyquist frequency (2954Hz), and the information will be accurately captured. Also, the FFT algorithm requires the length of the input signal is a power of two, and this sampling rate guarantees this as each tone is 0.25 seconds long, resulting in 8192 samples for the input.



(a) Signal for digits 0 and 2 (b) Zoomed in digit 0 (c) Zoomed in digit 2

Figure 38: Signal for the digits 0 and 2



(a) Spectrogram (b) FFT magnitude plot

Figure 39: Spectrogram and FFT magnitude plot of the signal

3.4.2

Figure 39 shows the spectrogram of the signal, which identifies the frequency components of a signal. The parameters used ensured that each 0.25 second segment of the signal had its own Fourier Transform computed, which allows us to read off the frequencies present in each key press as there are clearly 2 areas of high power at a particular frequency. In the idle time between presses, there are no frequencies as expected since there is nothing playing at these times. There is a spread in the power of frequencies due to the use of the Hanning window, whose side lobes cause spectral leakage. The FFT plot shows the precise peaks for the frequencies, but since there is an overlap between the frequencies used by the numbers, not all the numbers have their lines visible however the peaks correspond with what is seen in the spectrogram and the exact frequencies shown in Table 3.

3.4.3

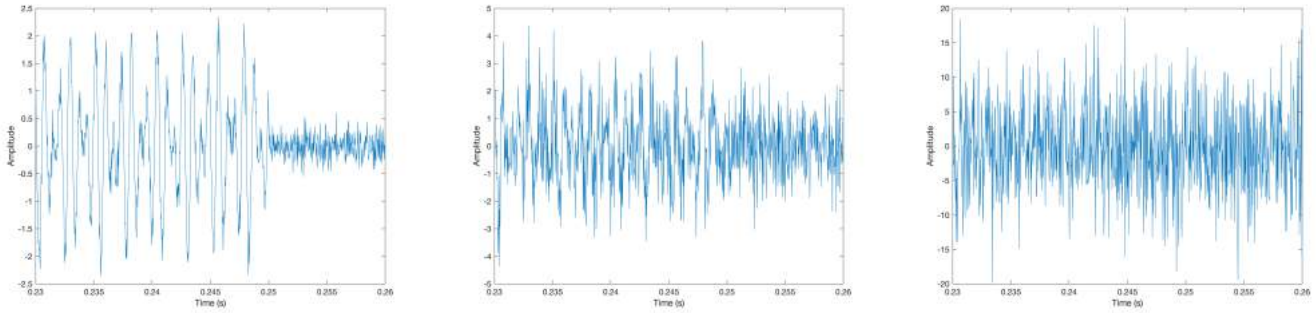
It is possible to identify the sequence of digits by iterating through each 0.25 second (8192 sample) window, taking the Fourier Transform like in Figure 39, finding the two highest power, distinct frequency values and reading off Table 3 to see the frequencies of each key press. Finding the two values could be done through looking for each known frequency value, within a small tolerance to make sure the two highest power values belong to separate digits; the highest values could describe the same digit due to spectral leakage so this method mitigates this effect. To The periods of silence between each key press should be ignored.

Number/Symbol	1	2	3	4	5	6	7	8	9	*	0	#
Low-Frequency (Hz)	697	697	697	770	770	770	852	852	852	941	941	941
High-Frequency (Hz)	1209	1336	1477	1209	1336	1477	1209	1336	1477	1209	1336	1477

Table 3: Dial pad frequencies

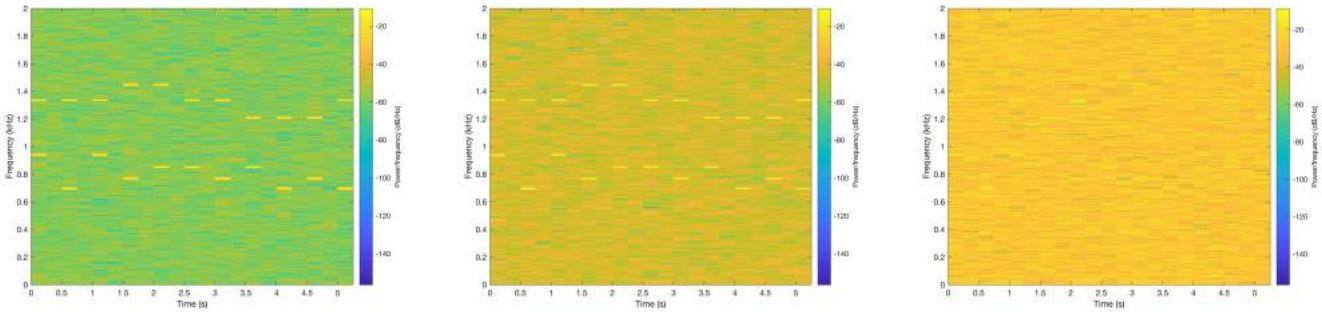
3.4.4

The dial tone signal had WGN added to it with three different levels of variance, until the signal appeared to be completely immersed in noise, as shown in Figure 40. The spectrograms for each variance are in Figure 41. The added noise obscures the sine waves in the signal more as σ increases; at $\sigma = 0.2$ the wave is still largely present and the spectrogram clearly shows the frequencies. To the eye, the signal plot for $\sigma = 1$ has been changed a lot by the noise and it is harder to locate the sine wave. Upon consulting its spectrogram however, the frequencies present are still able to be identified. At $\sigma = 6$, the power of the noise is too much as the frequencies cannot be seen in the spectrogram. The signal-to-noise ratio (SNR) gets worse (smaller) as σ increases since it is proportional to the power of WGN, and the noise will introduce additional frequency components, making it harder to find the true frequencies present.



(a) $\sigma = 0.2$ (b) $\sigma = 1$ (c) $\sigma = 6$

Figure 40: Dial tone signal with different noise variances

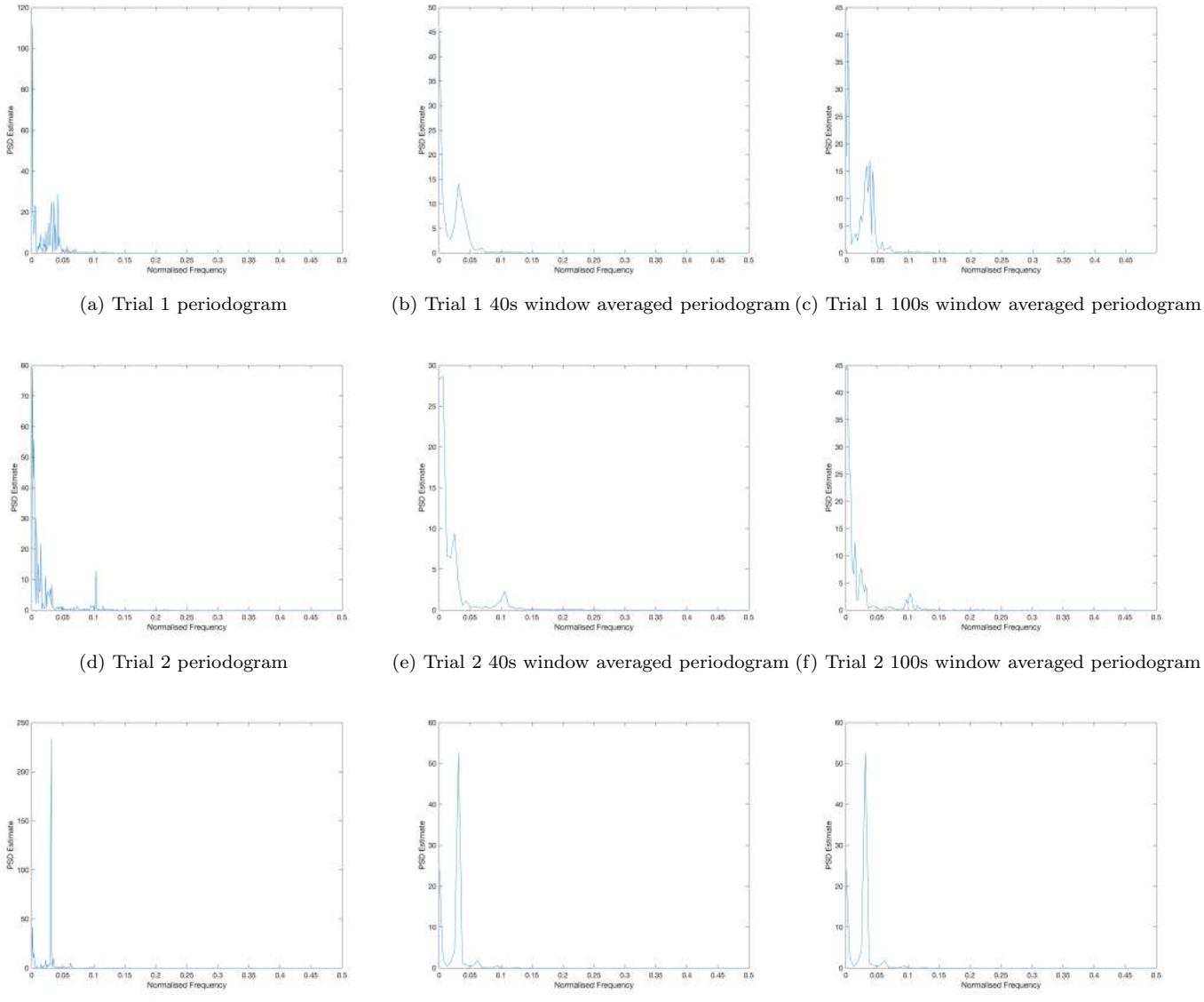


(a) $\sigma = 0.2$ (b) $\sigma = 1$ (c) $\sigma = 6$

Figure 41: Spectrogram from different noise variances

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

The RRI data from the ECG experiment was used to generate a periodogram and averaged periodograms using 2 different windows, 40 second and 100 second, for the 3 trials. The plots are in Figure 42, with the frequency zoomed in to make it easier to see the graphs (the graph is symmetrical about 0.5 on the x-axis). The PSD or RR data can show the variability of heart rate over time. Breathing in speeds up the heart rate and breathing out slows it down. The averaged periodogram that used a window of 100 seconds in general shows some more variability since there were less segments to average. The The peak for trial 1 is at a normalised frequency of 0.03, with a harmonic around 0.06. Trial 3 has the largest peak at 0.03, with a harmonic at 0.06. Trial 1 and trial 3 have fairly similar PSDs, indicating that the breathing rate ended up being similar despite the constrained restrictions in trial 3, although because trial 3 has the restriction, the peak is higher. The peak for trial 2 is a little harder to identify, with spikes at 0.025 and 0.11. The constrained breathing at 50 breaths per minute in trial 2 points to 0.11 being the true peak since the breathing rate is higher.



(a) Trial 1 periodogram (b) Trial 1 40s window averaged periodogram (c) Trial 1 100s window averaged periodogram

(d) Trial 2 periodogram (e) Trial 2 40s window averaged periodogram (f) Trial 2 100s window averaged periodogram

(g) Trial 3 periodogram (h) Trial 3 40s window averaged periodogram (i) Trial 3 100s window averaged periodogram

Figure 42: Periodogram and averaged periodograms for each trial

4 Optimal Filtering - Fixed and Adaptive

An unknown system and an identified system, $w[n]$, can be connected for system identification. Both systems are fed the same input, $x[n]$, the unknown system produces its output, $y[n]$ which has noise added to it to form $z[n]$. The error $z[n]$ and the identified system's output is used to update the identified systems parameters so that they match those of the unknown system.

4.1 Wiener Filter

4.1.1

The optimum Wiener filter involves minimising the the expectation of the squared error, and the the optimal weights $w_{opt} = R_{xx}^{-1} p_{zx}$. With a standard deviation of 0.1 for the added WGN noise, the coefficients obtained were [0.23,0.47,0.7,0.46,0.2306]. These do not reflect the value in the unknown system, which were set to be [1,2,3,2,1], but $y[n]$ was normalised which has affected the results. Undoing this normalisation gave coefficients of [1.01,2.00, 2.98, 1.98, 0.97] which are much closer to the actual coefficients. The measured SNR at $\sigma = 0.1$ for the noise was 19.89 and the theoretical SNR is $10log_{10}\frac{1}{0.1^2} = 20dB$.

4.1.2

Table 4 shows the measured SNR and coefficients $(w_1, w_2, w_3, w_4, w_5)$ for $\sigma^2 \in [0.1, 10]$. Increasing the noise power results in less accurate predictions by the Wiener filter since it is trying to match a signal that reflects the true output of the unknown system, but as this signal becomes noisier (as shown by the decreasing SNR), the minimisation of the error results in values that also minimise the error caused by the noise as well.

σ^2	SNR(dB)	w_1	w_2	w_3	w_4	w_5
0.25	6.9264	1.1290	2.0596	2.9935	1.8443	1.1521
1	3.1331	1.1190	2.0079	3.1123	2.0339	0.8823
2	1.9853	1.4836	1.8904	3.1280	2.0979	1.2534
4	0.9603	0.9141	2.3950	2.8725	1.9695	0.5402
7	0.6547	0.1433	1.8795	3.7514	1.8944	1.7029
10	0.3868	0.6141	1.7379	2.9468	2.6995	0.5270

Table 4: SNR and coefficients for different noise standard deviations

Increasing N_w, the number of coefficients in the identified system, yields a set of coefficients whose values beyond the 5th coefficient are very small since these are not needed.

4.1.3

The computational complexity of finding the Wiener solution is one of its drawbacks. The complexity of finding p_{zx} depends on the algorithm being used, for example MATLAB uses the FFT, although the need to only calculate up to lag N_w reduces this complexity. If we use a naive approach with the summation formula and say that each signal is of length K, then the complexity is $O(KN_w)$ as K multiplications and K-1 additions are needed per lag, and N_w lags are needed. It is the same complexity for R_{xx} as the operations are the same, but operating on the same input signal. R_{xx} has dimension $N_w + 1 \times N_w + 1$, so the complexity to invert it is $O({N_w}^3)$. Multiplying $R_{xx}^{-1} p_{zx}$ requires $O({N_w}^2)$ multiplications and additions, The dominant complexity is $O({N_w}^3)$, given N_w is large.

4.2 The Least Mean Square (LMS) Algorithm

4.2.1

The LMS algorithm takes a different approach to finding the weights; it recursively uses the error to make adjustments to the weights, after which they converge on the true value. Through this iterative method of using pieces of the data rather than all of it, there are no assumptions about stationarity, so this is more robust and versatile. When applying this algorithm to x and z from the previous section, a new set of weights are obtained: [1.0080, 2.0026, 2.9816, 1.9892, 0.9969]. These are extremely close to the true weights, however this algorithm is dependent on the adaptation gain, μ, which multiplies the error so that the changes to the weights are incremental for each iteration. Equation 54 shows the weight updates, equation 55 is the calculation of the output of the adaptive filter and equation 56 shows the calculation for the error.

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu e[n]\mathbf{x}(n) \tag{54}$$

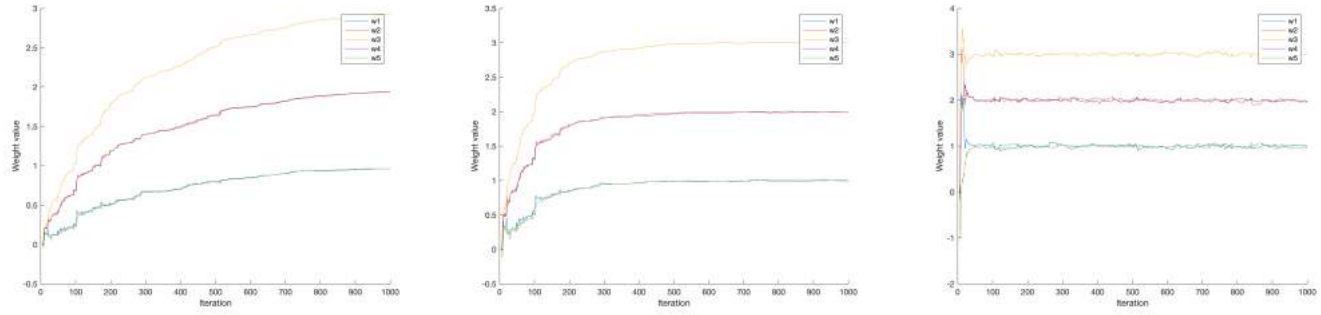
$$\hat{y}[n] = \mathbf{w}^T\mathbf{x}[n] = \sum_{m=0}^{N_w} w_m(n)x(n-m) \tag{55}$$

$$e[n] = z[n] - \hat{y}[n] \tag{56}$$

The code for my implementation of LMS is in Figure 43. The loop allows the grabbing of the relevant pieces of the input, x, to get the previous values, calculate $\hat{y}$ and $e[n]$ for that iteration, and update the value of the weights, storing them in rows of a matrix so the history can be seen.

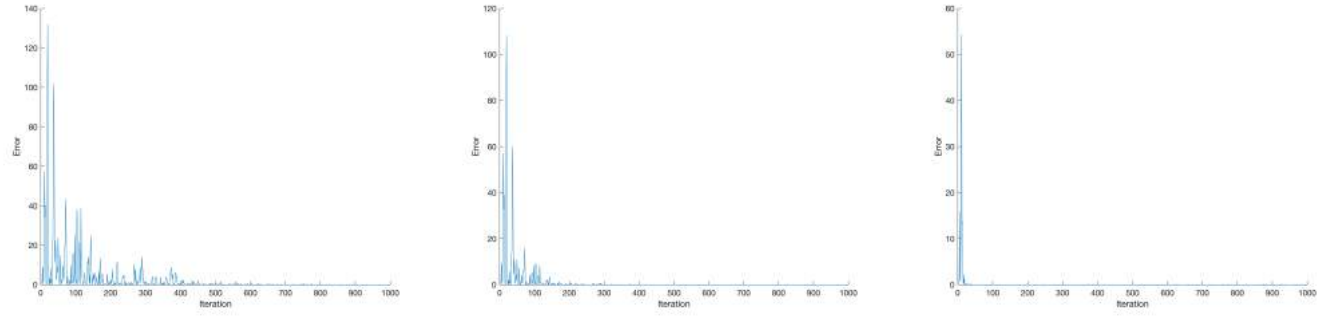
```
for i=order+1:N % need enough space to go backwards

    xtemp = x(i:-1:i-order); %get all previous values needed
    y(i) = w(:,i)' * xtemp;
    e(i) = z(i) - y(i);
    w(:,i+1) = w(:,i) + u * e(i) * xtemp;

end
```

Figure 43: LMS code



(a) $\mu = 0.004$ (b) $\mu = 0.01$ (c) $\mu = 0.1$

Figure 44: Evolution of weights for different μ values



(a) $\mu = 0.004$ (b) $\mu = 0.01$ (c) $\mu = 0.1$

Figure 45: Squared estimate error for different μ values

4.2.2

Figure 44 shows the effect of μ, with a lower value resulting in a slower convergence to the final value the weights take. Care should be taken with the gain, as a value that is too small will mean the weights do not converge on time, but a rate that is too high will make the weights take large steps on each iteration and therefore not properly converge, instead oscillating around what the value should be. From the three values used in the graphs, a value of around 0.006 would appear to be preferable when balancing these two factors, since for 0.004 it is difficult to tell if the weights did reach their final value, but for 0.01 the weights reached the final value quickly. A rate of 0.1 is too high, as this oscillation is visible. A rate of 0.006 would converge fast enough, whilst allowing enough granularity in the adjustments to reach an accurate value. This effect of μ is also clear when reading the squared error graphs in Figure 45, where the rate of error decrease gets higher as μ increases.

4.2.3

The computational complexity of the LMS algorithm is calculated: each update of the weights as per equation requires $N_w+1 = O(N_w)$ additions, each calculation of $\hat{y}[n]$ consists of N_w+1 multiplications and N_w additions so this is $O(N_w)$ and each calculation of the error is a single subtraction, $O(1)$. Combining all of these gives a complexity of $O(N_w)$ for one iteration. Given the algorithm runs M times (the length of the input), the overall complexity is $O(MN_w)$.

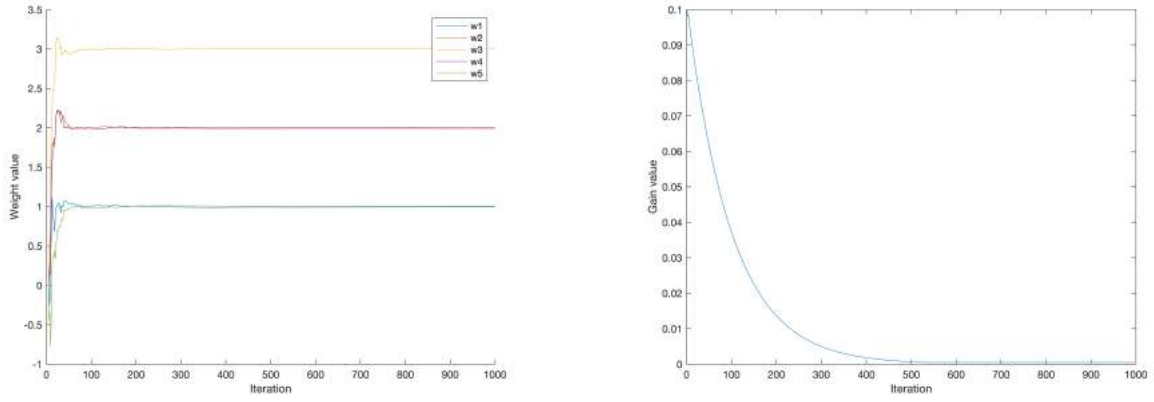
4.3 Gear Shifting

The idea of gear shifting is to reduce the gain over time to alleviate some of the issues discussed at the end of the last section. My implementation of this calculates the difference between the current and previous error and uses the gradient to adjust μ based on a hyper-parameter, beta. The code is shown in Figure 46 where the mechanism for altering μ can be seen in more detail.

```
% Difference
e_diff = e(i) - prev_e;
% Gradient
e_grad = e_diff / (i - order);
% Adjust adaptation gain based on error gradient
mu_adjust = beta * e_grad + (1-beta);
mu = mu * mu_adjust;
mu = max(mumin, mu);
w(:, i+1) = w(:, i) + mu * e(i) * xtemp;
% Store previous error for next iteration
prev_e = e(i);
trackgains = [trackgains mu];
```

Figure 46: Gear shifting code

A higher value of beta will give more weight to the gradient of the error signal, which means that the adaptation gain will be adjusted more aggressively in response to changes in the error signal. Whereas, a lower value of beta gives less weight to the gradient of the error signal, meaning the adaptation gain will be adjusted more slowly in response to changes in the error signal. This method reduces μ over time as the error becomes smaller, so finer adjustments to the weights can be made. This is shown in Figure 47, where the weights quickly reach a rough ballpark and then converge on the value, so the benefits of a small and large μ can be experienced. The value of the gain is also shown; it decreases over time. This was made possible by creating the 'trackgains' array to store all values. There is a minimum value of 0.0005 for μ so that it does not reach 0; this allows very small adjustments to make the final values as accurate as possible.



(a) Weights

(b) Gain

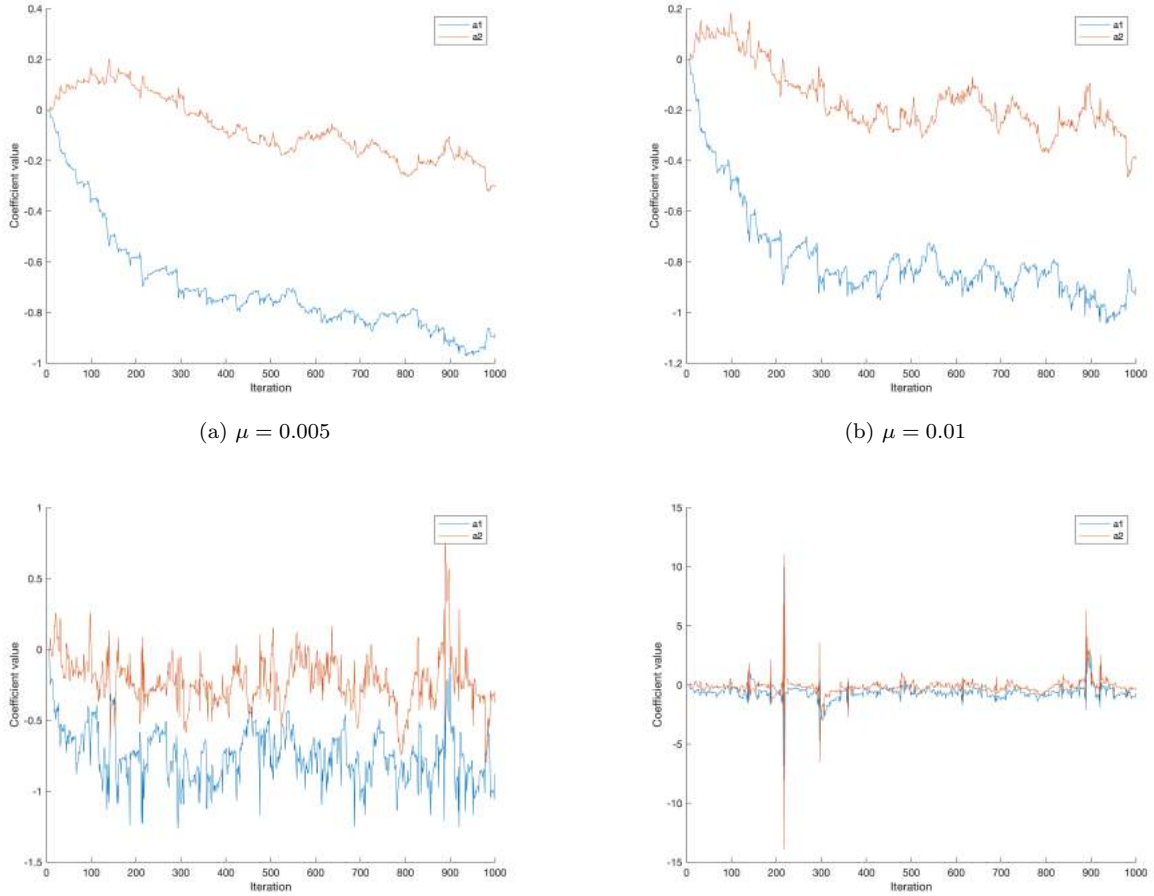
Figure 47: The evolution of the weights and gain over time

4.4 Identification of AR processes

A structure of a synthesis stage and an analysis stage was used to adaptively estimate an AR signal. The synthesis stage consists of noise passed into an AR model with parameters $\mathbf{a} = [10.90.2]$ (in MATLAB) which gives the input, $x[n]$ that feeds into the analysis stage. In this stage, $x[n-1]$ and $x[n-2]$ are multiplied with the coefficients that the adaptive filter alters, and then summed to form the output, $y[n]$. The error between

$y[n]$ and $x[n]$ is fed into the adaptive algorithm to alter the coefficients iteratively. With the filter working, the coefficients should converge to the values -0.9 and -0.2; the coefficients of the AR model (MATLAB reverses the sign).

The results from using the filter at different adaptation gains is shown in Figure 48. The smallest $\mu = 0.005$ was not able to converge quickly enough to the correct values, whereas for $\mu = 0.01$, the system reached the correct values but with a lot of variation; in this case an implementation of gear shifting would be useful to aid convergence. A gain of 0.05 is on the edge of usable since there is a significant level of variation, even though by the end the coefficients were close to the true values. Finally, a gain of 0.1 is too high; the adjustments made to the coefficients diverged in



(a) $\mu = 0.005$

(b) $\mu = 0.01$

(c) $\mu = 0.05$

(d) $\mu = 0.1$

Figure 48: The evolution of the weights and gain over time

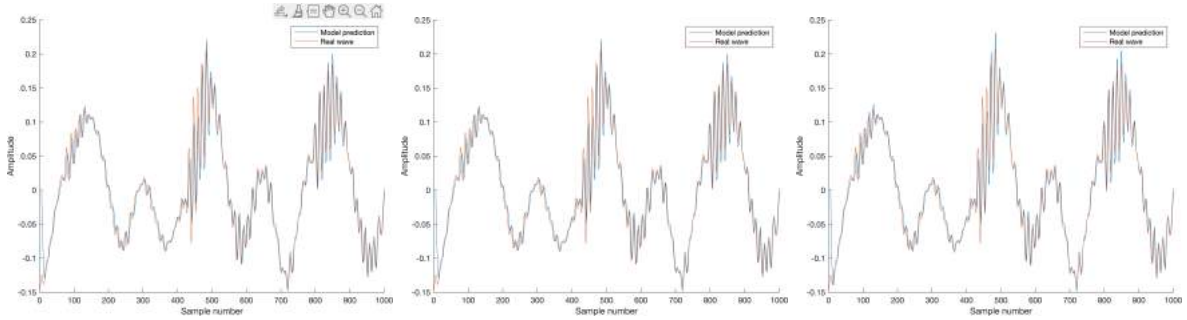
4.5 Speech Prediction

The analysis stage of the model used in the previous section can be used to predict sounds by inputting the sound and the model will find the weights (coefficients) needed to model it. I recorded me saying "e", "a", "s", "t", and "x" with a sampling frequency of 44100Hz, and used 1000 samples as an input to the predictor.

4.5.1

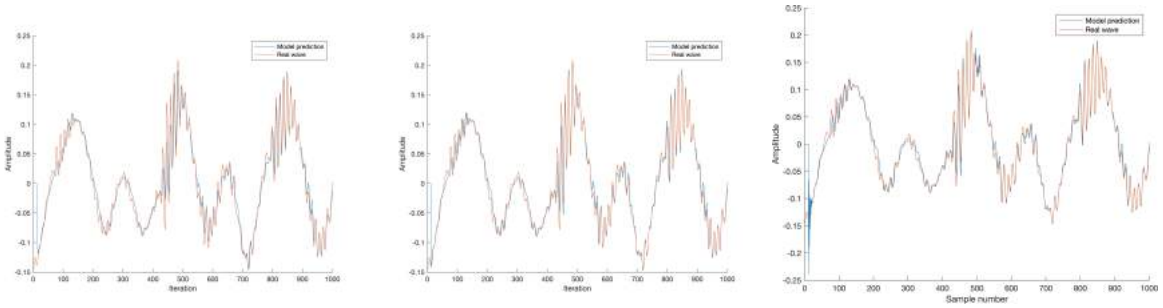
The value for the best predictive gain and order varied depending on the sound, and the values for gain are much higher than was needed in the AR prediction. It appeared that a higher order would require a comparatively smaller gain to avoid instability, and a lower order would need a comparatively higher gain in order for the weights to be able to change fast enough to make accurate predictions. For example, for "e", once the order was at 12, the gain had to be below 10 (since it was fixed). Figure 49 shows plots for "e", which as a spoken letter does not much variability as seen by the graphs, at order 5. The data is able to be modelled well, with the higher gains able to adapt to sudden changes well, but the overshoot increases. Figure 50 is "e" at order 10; this had more success at not overshooting the most quickly fluctuating parts of the input, but was not as

accurate at predicting the section from sample number 250 to 400, but the accuracy did generally improve as the gain got higher although the initial prediction for gain 10 was worse.



(a) gain = 1 (b) gain = 3 (c) gain = 5

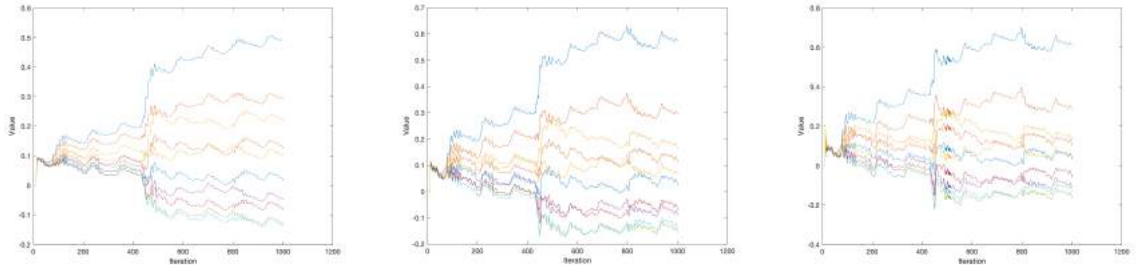
Figure 49: Prediction and actual signal for "e" at order 5



(a) gain = 2 (b) gain = 6 (c) gain = 10

Figure 50: Prediction and actual signal for "e" at order 10

The evolution of the weights for "e" with model order 10 is shown in Figure 51, where we can see that a gain of 2 does not allow some of the weights to each their proper value which explains the less accurate modelling, but the higher gains allows the weights to move more freely, with the weights for gain 10 moving the most aggressively, resulting in overshoots. There are also some weights which take very small values, perhaps indicating that an order this high is not necessary. Gear shifting was tested too, and it helped as the gain could adjust depending on any sudden changes due to the non-stationary nature of the input, raising it to counteract a large jump and change the weights quickly, and then lowering the gain once the error lowers for a more accurate prediction. Without this, the gain being at a fixed value meant a compromise had to be reached between accuracy and speed of convergence of the parameters. This was especially helpful for the more varying sounds like "x".



(a) gain = 2 (b) gain = 6 (c) gain = 10

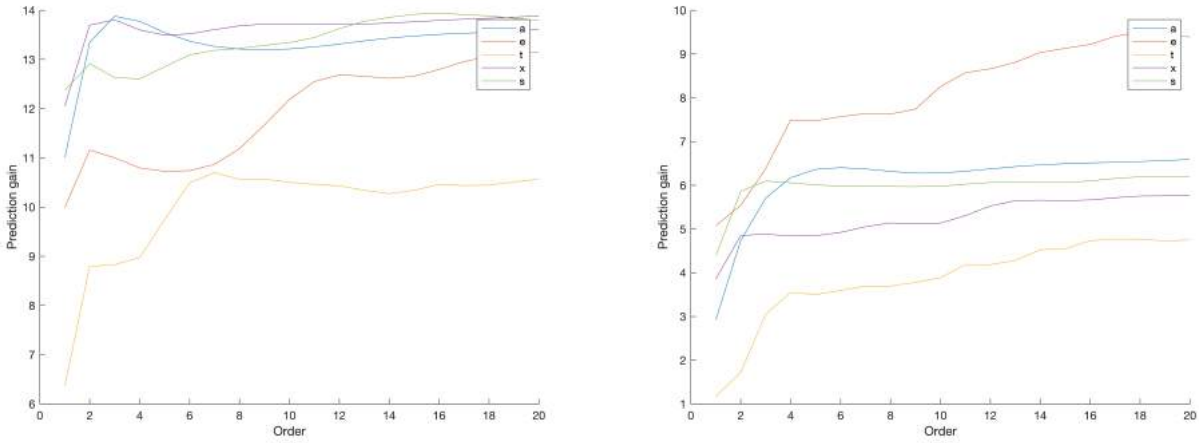
Figure 51: Weight changes for "e" at order 10

4.5.2

There are a number of methods which could be used to determine the best model length (order). The simplest method, but probably the worst is trial and error. Using the MDL and AIC criteria is a much better option

through looping over a range of orders, collating the cumulative squared error for each order and calculating the criteria. This would require the gain to be suitably set for each order, so an approach which loops over the gain for each order and takes the lowest error would solve this issue. A standard measure of the performance of an estimator is the prediction gain given in equation 57a, where σ_x^2 is the variance of the input and σ_e^2 is the variance of the output. The prediction gain was calculated for all letters and is shown in Figure 52. The value of the prediction gain was dependent on the adaptation gain (μ) used, so a variety of values for each letter was tried and the its line was plotted using the μ that gave an accurate prediction, but reduced instability as the order increased.

$$R_p = 10log_{10}\frac{\sigma_x^2}{\sigma_e^2} \tag{57}$$



(a) Sampling frequency = 44100Hz

(b) Sampling frequency = 16000Hz

Figure 52: Prediction gain for all letters for different sampling frequencies

4.5.3

It is worth noting that the section of the wave used is very small as 1000 samples at 44100Hz sampling frequency is only 0.02 seconds of audio and therefore the section used may be largely the same, but the results are dependent on which part of the wave was chosen. When using a sampling frequency of 16000Hz with 1000 samples, each sample captures a longer period of time (in total 0.0625 seconds) so less detail is captured due to quantization error, meaning the noise will also increase. Since the noise power will then be higher, the prediction gain will be lower for all letters, as shown in Figure 57b. Due to the nature of how consonants sound, they are in general non-stationary and do not have a clear quasi-stationary region. The capturing of a quasi-stationary region of vowel sounds depends on the sampling frequency; for a lower sampling frequency, less samples are needed to capture the region as the interval between samples is larger and the converse is true for a higher sampling rate. The convergence of learning curves follows a different pattern, where more samples are needed at lower sampling frequency in order to give the system enough information to alter the weights.

4.6 Dealing with Computational Complexity: Sign Algorithms

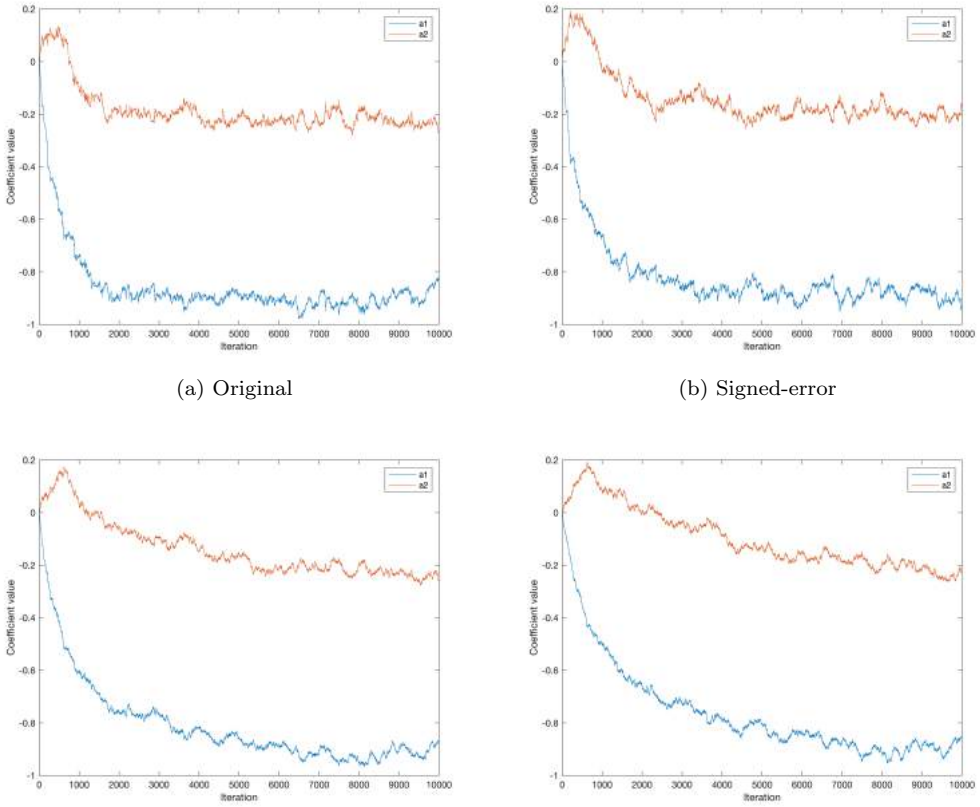
To reduce the computation required for the LMS algorithm, three simpler alternatives are proposed which implement the following weight updates:

$$\text{Signed-error:} \quad \mathbf{w}(n+1) = \mathbf{w}(n) + \mu \text{sign}(e[n])\mathbf{x}(n) \tag{58}$$

$$\text{Signed-regressor:} \quad \mathbf{w}(n+1) = \mathbf{w}(n) + \mu e[n]\text{sign}(\mathbf{x}(n)) \tag{59}$$

$$\text{Sign-sign:} \quad \mathbf{w}(n+1) = \mathbf{w}(n) + \mu \text{sign}(e[n])\text{sign}(\mathbf{x}(n)) \tag{60}$$

In order to evaluate the convergence speeds of these, they are applied to the AR identification problem from Section 4.4 and the convergence of the coefficients is shown in Figure 53. A small μ of 0.002 was used over 10000 iterations to convey the different speeds.



(a) Original

(b) Signed-error

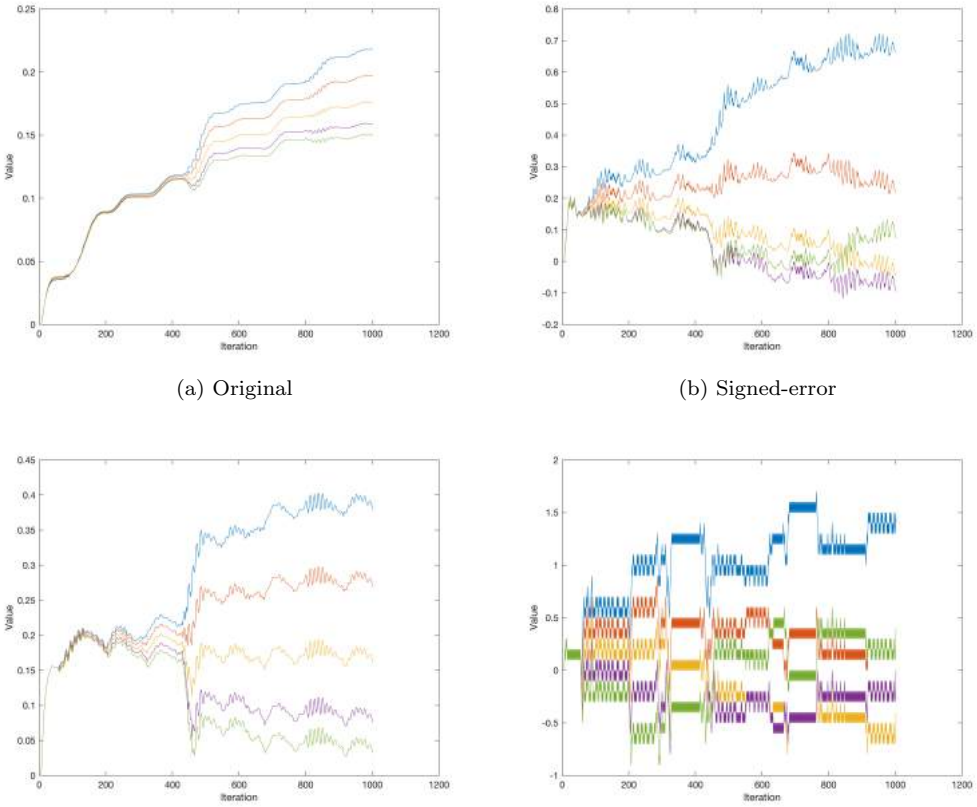
(c) Signed-regressor

(d) Sign-sign

Figure 53: Convergence of AR coefficients for the different LMS algorithms

The original version converged the quickest, signed-error 2nd quickest, signed-regressor 3rd and sign-sign was the slowest to converge. This is expected since the original is able to use all the error and input information to update the weights, whereas signed-error and signed-regressor reduce one of these parameters to just 1 or -1 (or 0 in very rare instances). Sign-sign can only use a 1 or -1 to update the weights which is why it converges so slowly. The fact that signed-error is faster converges faster than signed-regressor would indicate that the information in the error variable is more important than input $\mathbf{x}[n]$ which makes sense given that the magnitude of the error can be greater than 1. There does not appear to be a great difference in accuracy for this μ as all 4 variants were able to converge on the correct values of -0.9 and -0.2; the MSE of the final values is similar for all. However as μ increases, sign-sign has the highest MSE, the original has the lowest MSE with signed-error and signed-regressor close in the middle so as the algorithm simplifies, the accuracy can worsen if the conditions are not right.

In the case of speech prediction, the convergence speeds and accuracies are different as shown by Figure 54. A small μ was used again to portray the speeds. Here original algorithm is the slowest to converge (not converging in time), signed-regressor and signed-error come in 2nd and 3rd and sign-sign does not converge at all. It is worth pointing out that the signed algorithms are more sensitive to the μ value; the signed error algorithm tends to diverge more easily at a smaller μ compared to the original but the signed-regressor is a little more robust to this issue.



(a) Original

(b) Signed-error

(c) Signed-regressor

(d) Sign-sign

Figure 54: Convergence of speech coefficients for the different LMS algorithms

5 MLE for the Frequency of a Signal

5.1.1

The maximum likelihood estimator (MLE) of the frequency, f_0, for a real signal $\mathbf{x} = [x[0], x[1], ..., x[N-1]]^T$ that is parameterized by $\boldsymbol{\theta} = [A, f_0, \phi]^T$ is found by minimising:

$$J(\boldsymbol{\theta}) = \sum_{n=0}^{N-1} (x[n] - A\cos(2\pi f_0 n + \phi))^2 \tag{61}$$

Using the cosine identity cos(A+B) = cos(A)cos(B) - sin(A)sin(B):

$$J(\boldsymbol{\theta}) = \sum_{n=0}^{N-1} (x[n] - A\cos(2\pi f_0 n)\cos(\phi) + A\sin(2\pi f_0 n)\sin(\phi))^2 \tag{62}$$

Letting $\alpha_1 = A\cos(\phi)$ and $\alpha_2 = -A\sin(\phi)$:

$$J(\boldsymbol{\theta}) = \sum_{n=0}^{N-1} (x[n] - \alpha_1\cos(2\pi f_0 n) - \alpha_2\sin(2\pi f_0 n))^2 \tag{63}$$

Defining $\mathbf{c} = [1, \cos(2\pi f_0), ..., \cos(2\pi f_0(N-1)]^T$ and $\mathbf{s} = [0, \sin(2\pi f_0), ..., \sin(2\pi f_0(N-1)]^T$ allows the conversion to a vector format (the transpose of a vector multiplied by itself is equivalent to squaring):

$$J'(\alpha_1, \alpha_2, f_0) = (\mathbf{x} - \alpha_1\mathbf{c} - \alpha_2\mathbf{s})^T(\mathbf{x} - \alpha_1\mathbf{c} - \alpha_2\mathbf{s}) = (\mathbf{x} - \mathbf{H}\boldsymbol{\alpha})^T(\mathbf{x} - \mathbf{H}\boldsymbol{\alpha}) = J'(\boldsymbol{\alpha}, f_0) \tag{64}$$

where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2]^T$ and $\mathbf{H} = [\mathbf{c}, \mathbf{s}]$

5.1.2

The minimising solution to equation 64, $\hat{\boldsymbol{\alpha}}$, can be found by expanding, differentiating and setting to 0:

$$(\mathbf{x} - \mathbf{H}\boldsymbol{\alpha})^T(\mathbf{x} - \mathbf{H}\boldsymbol{\alpha}) = \mathbf{x}^T\mathbf{x} - 2\boldsymbol{\alpha}^T\mathbf{H}^T\mathbf{x} + \boldsymbol{\alpha}^T\mathbf{H}^T\mathbf{H}\boldsymbol{\alpha} \tag{65}$$

$$\frac{\partial J'}{\partial \boldsymbol{\alpha}} = -2\mathbf{H}^T\mathbf{x} + 2\mathbf{H}^T\mathbf{H}\boldsymbol{\alpha} = 0 \tag{66}$$

$$\hat{\boldsymbol{\alpha}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \tag{67}$$

which turns out to be the pseudo-inverse from the least-squares solution.

Substituting this back into equation 64:

$$J'(\hat{\alpha}_1, \hat{\alpha}_2, f_0) = (\mathbf{x} - \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x})^T(\mathbf{x} - \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x}) = \mathbf{x}^T\mathbf{x} - \mathbf{x}^T\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \tag{68}$$

Since $\mathbf{x}^T\mathbf{x}$ comes out to be a constant, minimising $J'(\hat{\alpha}_1, \hat{\alpha}_2, f_0)$ is equivalent to maximising $\mathbf{x}^T\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x}$.

5.1.3

Given the definition for H and that c and s are column vectors the same length as x, maximising can be transformed in the following way:

$$\mathbf{x}^T\mathbf{H} = (\mathbf{H}^T\mathbf{x})^T = \left(\begin{bmatrix} c_0 & c_1 & \dots & c_{N-1} \\ s_0 & s_1 & \dots & s_{N-1} \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ \dots \\ x[N-1] \end{bmatrix} \right)^T = \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix}^T \tag{69}$$

$$(\mathbf{H}^T\mathbf{H}) = \begin{bmatrix} \mathbf{c} \\ \mathbf{s} \end{bmatrix} [\mathbf{c} \quad \mathbf{s}] = \begin{bmatrix} c_0 & c_1 & \dots & c_{N-1} \\ s_0 & s_1 & \dots & s_{N-1} \end{bmatrix} \begin{bmatrix} c_0 & s_0 \\ c_1 & s_1 \\ \vdots & \vdots \\ c_{N-1} & s_{N-1} \end{bmatrix} = \begin{bmatrix} \mathbf{c}^T\mathbf{c} & \mathbf{c}^T\mathbf{s} \\ \mathbf{s}^T\mathbf{c} & \mathbf{s}^T\mathbf{s} \end{bmatrix} \tag{70}$$

Since $\mathbf{H}^T\mathbf{x} = (\mathbf{x}^T\mathbf{H})^T$, the maximisation becomes:

$$\begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix}^T \begin{bmatrix} \mathbf{c}^T\mathbf{c} & \mathbf{c}^T\mathbf{s} \\ \mathbf{s}^T\mathbf{c} & \mathbf{s}^T\mathbf{s} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix} \tag{71}$$

5.1.4

It is required that f_0 is not close to 0 or 0.5 so that the middle matrix is invertible; for these values, the sin terms in s all become 0 and so the matrix becomes singular. Under these conditions, can be shown that expression in 71 is approximately:

$$\begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix}^T \begin{bmatrix} \frac{N}{2} & 0 \\ 0 & \frac{N}{2} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix} \tag{72}$$

$$= \frac{4}{N^2} \begin{bmatrix} \frac{N}{2}\mathbf{c}^T\mathbf{x} & \frac{N}{2}\mathbf{s}^T\mathbf{x} \end{bmatrix} \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix} = \frac{2}{N}[(\mathbf{c}^T\mathbf{x})^2 + (\mathbf{s}^T\mathbf{x})^2] \tag{73}$$

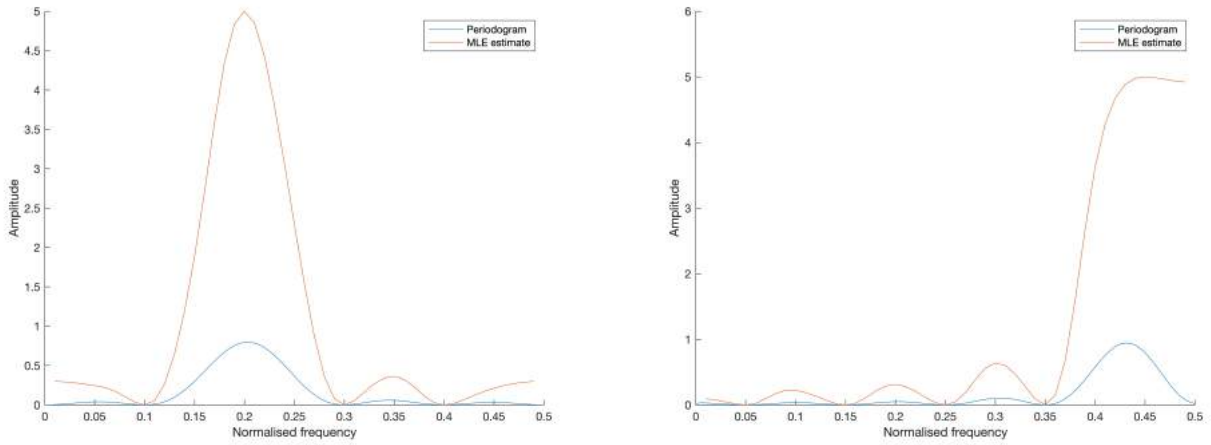
$$= \frac{2}{N}\left[(\sum_{n=0}^{N-1} x[n]\cos 2\pi f_0 n)^2 + (\sum_{n=0}^{N-1} x[n]\sin 2\pi f_0 n)^2 \right] \tag{74}$$

$$= \frac{2}{N}\left| (\sum_{n=0}^{N-1} x[n] e^{-j2\pi f \frac{n}{N}} \right|^2 = \hat{P}_X(f) \tag{75}$$

Therefore the MLE for $\hat{f}_0$ is obtained by maximising the periodogram given in equation 75.

5.1.5

The MLE estimate and periodogram of the noiseless signal $x = \cos(2\pi f_0 n)$ for $n = 0, 1, ..., N-1$ at different f_0 values and $N = 10$ are shown in Figure 55. The amplitude reached by the MLE estimate is roughly 6 times that of the the periodogram. At $f_0 = 0.2$, both methods are able to show the correct value but when f_0 is 0.45, the MLE estimate does not show a clear peak since it is approaching the unallowed value of 0.5; when this happens, the interaction of the sinusoids result in $(\mathbf{H}^T\mathbf{H})^{-1}$ becoming less reliable as the sin terms become very small. The periodogram becomes less accurate too as f_0 approaches 0 or 0.5. This is because a small number of samples are used, but since the periodogram is asymptotically unbiased a greater number of samples would solve this bias problem.



(a) $f_0 = 0.2$ (b) $f_0 = 0.45$

Figure 55: MLE and periodogram for different f_0 values