Imperial College London, Department of Electrical and Electronic Engineering

Advanced Signal Processing Coursework

Nima Karshenas

Contents

1 Random signals and stochastic processes

1.1 Statistical Estimation

1.1.1 Uniform Distribution: `rand`

Figure 1 shows the plot of a 1000 sample realisation of the MATLAB function `rand`, which corresponds to a uniform random variable, $X \sim \mathcal{U}(0,1)$, defined in equation 1. From this 1000 sample realisation, we calculate the sample mean ($\hat{\mu}_X$) and standard deviation($\hat{\sigma}_X$), via Equations 2 and 3, which are shown as well as their respective theoretical values (Equation 4 and 6) in Figure 1.

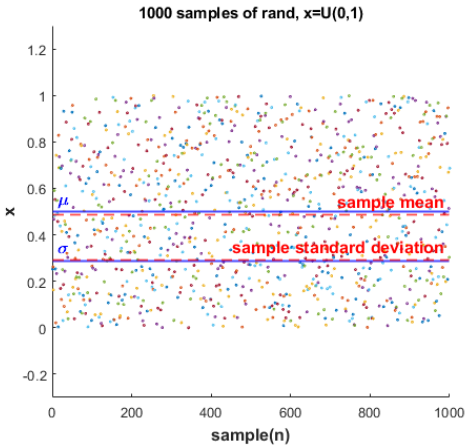


Figure 1: 1000 samples drawn from $\mathcal{U}(0,1)$

$$pdf(x) = \mathcal{U}(0,1) = \begin{cases} 1, & \text{if } 0 \le x \le 1 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

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

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

$$\mu_X = \mathbb{E}\{X\} = \int_{-\infty}^{\infty} x \cdot pdf(x)\,dx = \int_0^1 x\,dx = \tfrac{1}{2}x^2\Big|_0^1 = \tfrac{1}{2} \tag{4}$$

$$\sigma_X^2 = \mathbb{E}\{(X-\mathbb{E}\{X\})^2\} = \mathbb{E}\{X^2 - 2X\,\mathbb{E}\{X\} + (\mathbb{E}\{X\})^2\} = \mathbb{E}\{X^2\} - 2(\mathbb{E}\{X\})(\mathbb{E}\{\mathbb{E}\{X\}\}) + (\mathbb{E}\{X\})^2 \tag{5}$$

Note that since, $\mathbb{E}\{\mathbb{E}\{X\}\} = \mathbb{E}\{X\}$, then:

$$\sigma_X^2 = \mathbb{E}\{X^2\} - 2(\mathbb{E}\{X\})^2 + (\mathbb{E}\{X\})^2 = \mathbb{E}\{X^2\} - (\mathbb{E}\{X\})^2$$

$$= \int_{-\infty}^{\infty} x^2 \cdot pdf(x)\,dx - (\tfrac{1}{2})^2 = \tfrac{1}{3}x^3\Big|_0^1 - \tfrac{1}{4} = \tfrac{1}{12} \quad \therefore \quad \sigma_X = \sqrt{1/12} \approx 0.289 \tag{6}$$

The sample mean of the 1000-sample realisation of $\mathbf{x}$ is found using the MATLAB function `mean`, which yields a sample mean of 0.4945. From central limit theorem we expect sample estimations of the mean to be normally distributed with variance $\hat{\sigma}_N = \frac{\sigma}{\sqrt{N}}$, in this case, $\hat{\sigma}_N = \frac{0.289}{\sqrt{1000}} = 0.00914$ and so it is expected 95% of sample means within $\pm 0.0183 (\pm 2\hat{\sigma}_N)$ of the true value, meaning the estimate that was yielded from the MATLAB script, is a likely estimate. From a 1000-sample realisation, the standard deviation of X was given to be 0.2895.

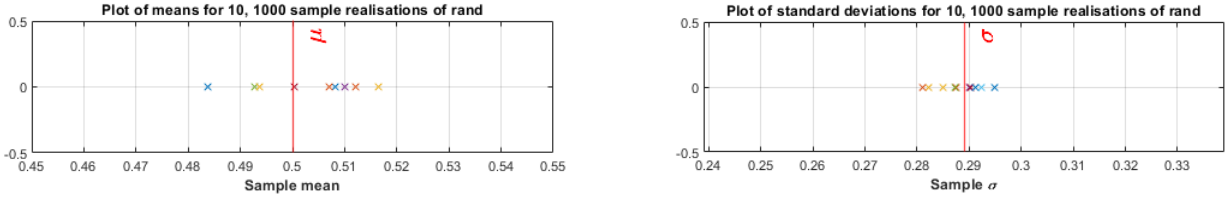


Figure 2: 10 sample means and standard deviations drawn from $\mathcal{U}(0,1)$ with a visible scope of ± 0.05 of their theoretical values.

Figure 2 shows the plots 10 sample means and standard deviations, it can be seen that the estimations for σ_X are more closely centred around its theoretical value, suggesting it has a smaller sample variance compared to the mean.

In Figure 3 we plot the histogram of samples drawn from the distribution as defined in this section,

along with the theoretical PDF. We can clearly see that as the number of samples increases, the estimated PDF converges to the theoretical PDF. It can also be seen that as the number of bins is increased, the deviation from the theoretical PDF is also increased and so it can be claimed that the greater the number of bins, the slower the convergence towards the theoretical PDF's shape.

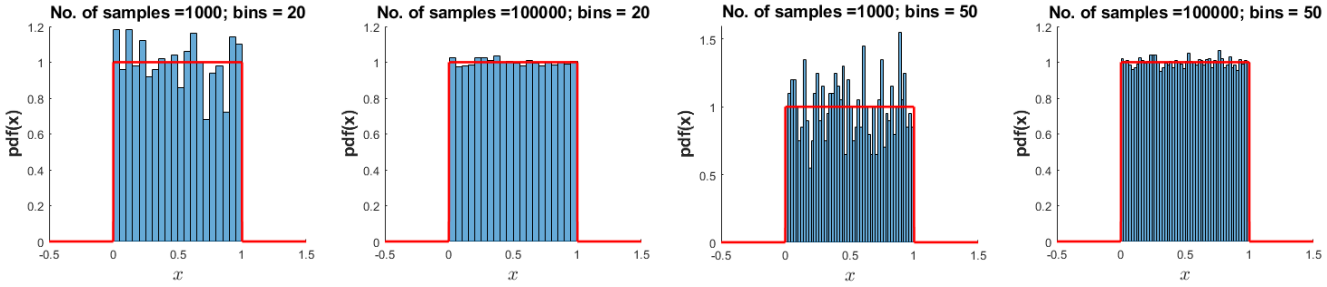


Figure 3: Histograms of samples drawn from $\mathcal{U}(0,1)$ for varying sample and bin sizes

1.1.2 Normal Distribution: `randn`

Figure 4 shows the plot of a 1000 sample realisation of the MATLAB fucntion `randn`, which now corresponds to the Gaussian random variable, $X \sim \mathcal{N}(0,1)$, of mean 0 and variance 1 (defined in Equation 7). As expected, samples appear to be clustered around the mean, with less and less points being realised as we move further from the mean. The sample mean and standard deviation are calculated via equations 2 and 3, and are shown alongside the theoretical values in Figure 4.

$$pdf(x) = \mathcal{N}(0,1) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} \tag{7}$$

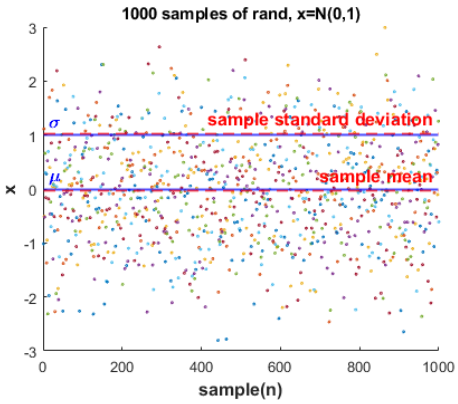


Figure 4: 1000 samples drawn from $\mathcal{N}(0,1)$

From the definition of `randn`, the theoretical mean is 0, it's 1000 sample estimate is given to be `-0.0135`. From central limit theorem we expect sample estimations of the mean to be normally distributed with standard deviation $\hat{\sigma}_N = \frac{\sigma}{\sqrt{N}}$, in this case $\hat{\sigma} = \frac{1}{\sqrt{1000}} = 0.0316$ and so we expect 95% of sample means within ± 0.0632 of the true value, meaning the estimate yielded from the MATLAB script is indeed a likely estimate. Again, by definition, the theoretical standard deviation of `randn` is 1, its 1000 sample estimate is `1.0315`.

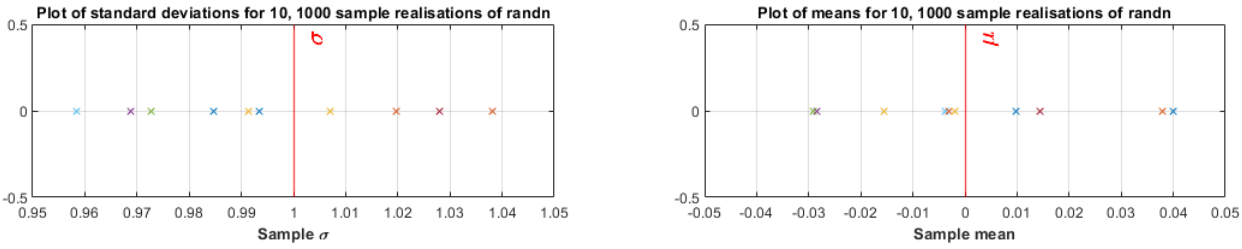


Figure 5: 10 sample means and standard deviations drawn from $\mathcal{N}(0,1)$ with a visible scope of ± 0.05 of their theoretical values.

Figure 5 shows 10 sample means and standard deviations drawn from $\mathcal{N}(0,1)$. Unlike in Section 1.1.1, estimates for σ_x and μ_X seem to be evenly distributed around their respective theoretical values.

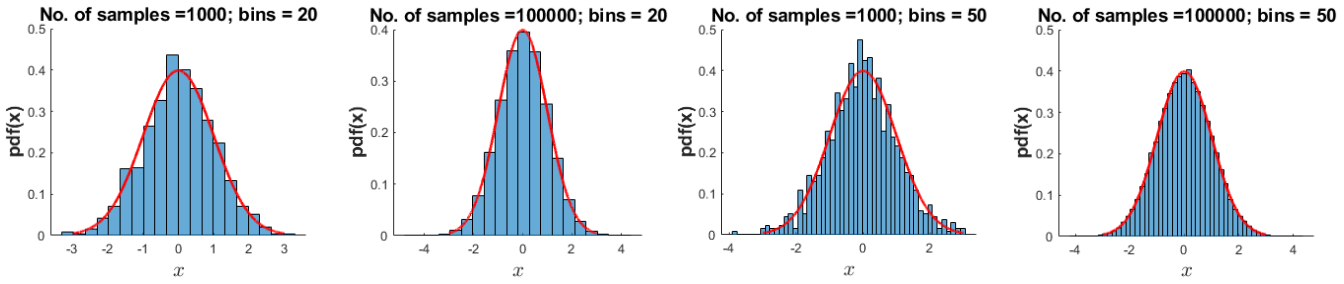


Figure 6: Histograms of samples drawn from $\mathcal{U}(0,1)$ for varying sample and bin sizes

Figure 6 shows the histograms of samples drawn from the normal distribution. As in Section 1.1.1, it can be seen that as the number of samples increases, the approximated PDF converges towards the theoretical PDF. Moreover, as the number of bins increases, the slower the convergence to the theoretical PDF. It can also be seen that when the number of bins is increased, given the sample size is sufficiently large, it allows the histogram to more tightly fit to the theoretical PDF, a feature which was not evident for `rand` due to the uniformality in its PDF.

1.2 Stochastic processes

1.2.1 Random Process 1; Questions 1, 2, & 3

Random Process 1 (`rp1`) is defined as the following MATLAB function:

```
function v = rp1(M,N)
    a=0.02;
    b=5;
    Mc=ones(M,1)*b*sin((1:N)*pi/N);  %half a period worth of N samples of sin of amplitude
    Ac=a*ones(M,1)*[1:N];
    v=(rand(M,N)-0.5).*Mc+Ac;
end
```

It can be seen that the output of the function is an $M \times N$ matrix, with M being the total number of realisations of the process and N being the number of samples used to construct each of the realisations. Figure 7 shows the Ensemble mean and standard deviation of `rp1`.

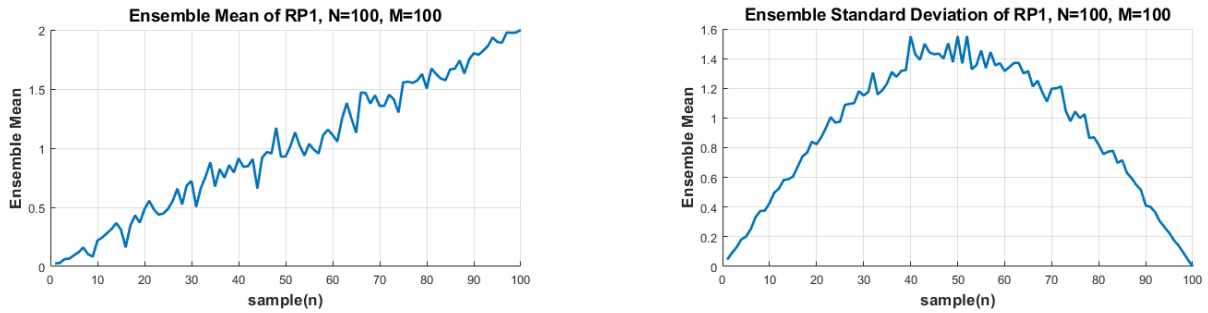


Figure 7: Ensemble mean and standard deviation for `rp1`

By looking at the script defining `rp1`, it can be seen that for large N, the output of the process is dominated by the incrementing bias, represented in code as the matrix `Ac`. What causes the deviation about the mean of course is the noise represented by `rand`, however this value is being multiplied by $sin(\frac{\pi n}{N})$, where n is the sample number and N is as in the function `rp1`. This means that for $n = 0$ and $n = N$, this product is equal to 0, hence why the value of the standard deviation is 0 at these points. Furthermore, as n tends to these values, the noise seen by the deterministic part of the signal tends to 0. Likewise, at $n = \frac{N}{2}$,

$sin(\frac{\pi n}{N}) = 1$, the noise seen by the deterministic part of the signal is now at its maximum and hence why we see a peak in the standard deviation.

A process is stationary if its probability distribution, and hence statistical properties such as mean and variance, do not vary with time. Immediately, we can clearly see that neither the mean nor the variance is stationary and so we cannot consider `rp1` as a stationary process.

	Realisation 1	**Realisation 2**	**Realisation 3**	**Realisation 4**
Mean	10.0037	9.9945	10.0034	9.9817
Standard Deviation	5.8631	5.8670	5.8512	5.8393

Table 1: Table showing sample means and standard deviations of 4, 1000 sample realisations of `rp1`

Table 1 shows the results of 4, 1000-sample realisations of `rp1`. For a stochastic process to be **Ergodic**, its statistical properties can be deduced from a sufficiently long sample of the process. Translating this to what is expected from this experiment, an ergodic process will show general agreement for the estimation of mean and standard deviation from each sample. Inspecting the results for `rp1`, we see that this is the case, however, as previously concluded, it is not stationary, if the value of `N` was changed, a different value for the sample mean and standard deviation would be calculated, hence, the process is **non-ergodic**.

We now attempt to derive the theoretical mean and standard deviation for the process, we begin by translating the function `rp1` into a mathematical expression (Equation 8)

$$x[n] = 5(w[n] - 0.5) \cdot sin(\tfrac{\pi n}{N}) + 0.02n \;=\; x[n] = 5g[n]sin(\tfrac{\pi n}{N}) + 0.02n \tag{8}$$

where, $x[n]$ is the nth sample taken from the process, and N is the total number samples taken, $w[n] \sim \mathcal{U}(0,1)$ and $g[n] \sim \mathcal{U}(-0.5, 0.5)$. For simplicity, the following values are established beforehand:
$\mathbb{E}\{w[n]\}$, $\mathbb{E}\{g[n]\}$, $\mathbb{E}\{w^2[n]\}$, $\mathbb{E}\{g^2[n]\}$.From part 1 we concluded that $\mathbb{E}\{w[n]\} = 0.5$ and by inspection we can see that $\mathbb{E}\{g[n]\} = 0$. Now:

$$\mathbb{E}\{w^2[n]\} \;=\; \int_0^1 n^2 \cdot dn \;=\; \tfrac{1}{3}x^3\Big|_0^1 = \tfrac{1}{3} \tag{9}$$

similarly,

$$\mathbb{E}\{g^2[n]\} \;=\; \int_{-0.5}^{0.5} n^2 \cdot dn \;=\; \tfrac{1}{3}x^3\Big|_{-0.5}^{0.5} = \tfrac{1}{12} \tag{10}$$

These values will be used to simplify analysis in the following sections as well. Now, we calculate the mean and standard deviation for this process, from equation 8, it follows that:

$$\mathbb{E}\{x[n]\} \;=\; \mathbb{E}\{5g[n]sin(\tfrac{\pi n}{N}) + 0.02n\}$$

From the linearity of the expectation operation and the independence between stochastic and deterministic processes we can write this as,

$$=\; 5\,\mathbb{E}\{g[n]\} \cdot \mathbb{E}\{sin(\tfrac{\pi n}{N})\} + 0.02\,\mathbb{E}\{n\} =$$

since $\mathbb{E}\{g[n]\} = 0$,

$$=\; 0.02\,\mathbb{E}\{n\} \;=\; \tfrac{0.02}{N}\sum_{n=1}^{N} n \;=\; \tfrac{0.02}{N} \cdot \tfrac{1}{2}N(N+1); \;=\; 0.01(N+1)$$

for $N = 1000$, $\mathbb{E}\{x[n]\} = 10.01$, which confirms the results seen in Table 1. Now for variance, from equation

8 it follows that,

$$\mathbb{E}\{x^2[n]\} \;=\; \mathbb{E}\{(5g[n]sin(\tfrac{\pi n}{N})+0.02n)^2\}$$

Expanding, using linearity and noticing independence, we have:

$$=\; 25\overbrace{\mathbb{E}\{g^2[n]\}}^{1/12}\cdot\mathbb{E}\{sin^2(\tfrac{\pi n}{N})\}+\overbrace{10\cdot 0.02\,\mathbb{E}\{g[n]\}\,\mathbb{E}\{sin(\tfrac{\pi n}{N})\cdot n\}}^{0}+(0.02)^2\,\mathbb{E}\{n^2\}$$

Now for $\mathbb{E}\{sin^2(\frac{\pi n}{N})\}$, $sin^2(\frac{\pi n}{N})$ is covering half a period no matter the value of N and so we can calculate its value from its continous-time form:

$$\mathbb{E}\{sin^2(x)\}\;\; for,\;\; 0\le x\le\pi \;=\; \frac{1}{\pi}\int_0^{\pi} sin^2(x)\cdot dx$$

$$\rightarrow \int_0^{\pi} sin^2(x)\cdot dx \;=\; \int_0^{\pi} \tfrac{1}{2}-\tfrac{1}{2}cos(2x)\;dx \;=\; \tfrac{1}{2}x-\tfrac{1}{4}sin(2x)\Big|_0^{\pi}=\tfrac{\pi}{2}$$

$$\therefore\;\; \mathbb{E}\{sin^2(x)\} \;=\; \mathbb{E}\{sin^2(\tfrac{\pi n}{N})\} \;=\; \tfrac{1}{\pi}\cdot\tfrac{\pi}{2} \;=\; \tfrac{1}{2}$$

Now,

$$\mathbb{E}\{x^2[n]\} \;=\; \tfrac{25}{24}+(0.02)^2\,\mathbb{E}\{n^2\} \;=\; \tfrac{25}{24}+\tfrac{(0.02)^2}{N}\sum_{n=1}^{N} n^2 \;=$$

$$=\; \tfrac{25}{24}+\tfrac{(0.02)^2}{N}\cdot\frac{1}{6}N(N+1)(2N+1) \;=\; \tfrac{25}{24}+\tfrac{(0.02)^2}{6}(N+1)(2N+1)$$

For $N=1000$,

$$\mathbb{E}\{x^2[n]\} \;=\; 134.58 \;\;\therefore\;\; \sigma_X^2=134.58-\overbrace{(\mathbb{E}\{x[n]\})^2}^{10.01^2} \;=\; 34.37 \tag{11}$$

1.2.2 Random Process 2: Questions 1, 2 & 3

The random process `rp2` is defined in the following MATLAB function:

```
function v=rp2(M,N)
Ar=rand(M,1)*ones(1,N);
Mr=rand(M,1)*ones(1,N);
v=(rand(M,N)-0.5).*Mr+Ar
end
```

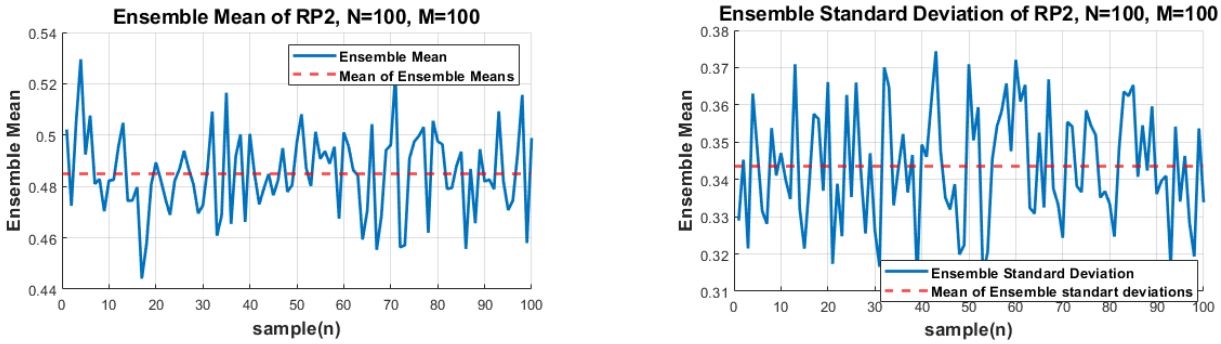


Figure 8: Ensemble mean and standard deviation for `rp2`

Using this function, the ensemble mean and standard deviation are plotted in Figure 8. This process shows only stochastic behaviour(with deterministic bias), which is as expected since it is purely composed of random variables in its function definition. Both mean and standard deviation deviate about a fixed

value, suggesting they do not vary through time, and thus may be considered as stationary moments. This is however insufficient evidence to claim that it is strict sense stationary since we have no information regarding estimations for higher order moments.

As done with `rp1`, The experiment is repeated for $M = 4$, $N = 1000$, the results being shown in Table 2. It can be seen that there is no agreement between sample estimates of the mean nor the standard deviation, and from the definition established in the previous section, `rp2` is **non-ergodic**.

	Realisation 1	**Realisation 2**	**Realisation 3**	**Realisation 4**
Mean	0.8750	0.2321	0.8477	0.4107
Standard Deviation	0.0836	0.1475	0.0384	0.0751

Table 2: Table showing sample means and standard deviations of 4, 1000 sample realisations of `rp2`

Finally, we attempt to derive the theoretical mean and standard deviation of the process, and begin by translating the MATLAB function into a mathematical expression as follows:

$$x[n] \;=\; (w[n]-0.5)\cdot w[n]+w[n] \;\;=\;\; g[n]\cdot w[n]+w[n] \tag{12}$$

Where $w[n], x[n], g[n]$ are as previously defined. Now calculating the mean and standard deviation for this process:

$$\mathbb{E}\{x[n]\} \;=\; \mathbb{E}\{g[n]\cdot w[n]+w[n]\} \;=\; \mathbb{E}\{g[n]\cdot w[n]\}+\mathbb{E}\{w[n]\}$$

$$\mathbb{E}\{x[n]\} = \overbrace{\mathbb{E}\{g[n]\}}^{0}\cdot\overbrace{\mathbb{E}\{w[n]\}}^{0.5}+\overbrace{\mathbb{E}\{w[n]\}}^{0.5} \;=\; 0.5 \tag{13}$$

$$\mathbb{E}\{x^2[n]\} \;=\; \mathbb{E}\{(g[n]\cdot w[n]+w[n])^2\} \;=\; \mathbb{E}\{g^2[n]w^2[n]\}+2\,\mathbb{E}\{g[n]w^2[n]\}+\mathbb{E}\{w^2[n]\}$$

$$\overbrace{\mathbb{E}\{g^2[n]\}}^{1/12}\cdot\overbrace{\mathbb{E}\{w^2[n]\}}^{1/3}+\overbrace{2\,\mathbb{E}\{g[n]\}\,\mathbb{E}\{w^2[n]\}}^{0}+\overbrace{\mathbb{E}\{w^2[n]\}}^{1/3} \;=\; \tfrac{13}{36} \;\;\Longrightarrow\;\; \sigma_X^2=\tfrac{13}{36}-(0.5)^2 \;=\; \tfrac{1}{9}. \tag{14}$$

1.2.3 Random Process 3: Questions 1, 2 & 3

```
function v=rp3(M,N)
a=0.5;
m=3;
v=(rand(M,N)-0.5)*m + a
end
```

The same set of experiments are carried out on `rp3`, beginning with the plots for the ensemble mean and standard deviation, shown in Figure 9.

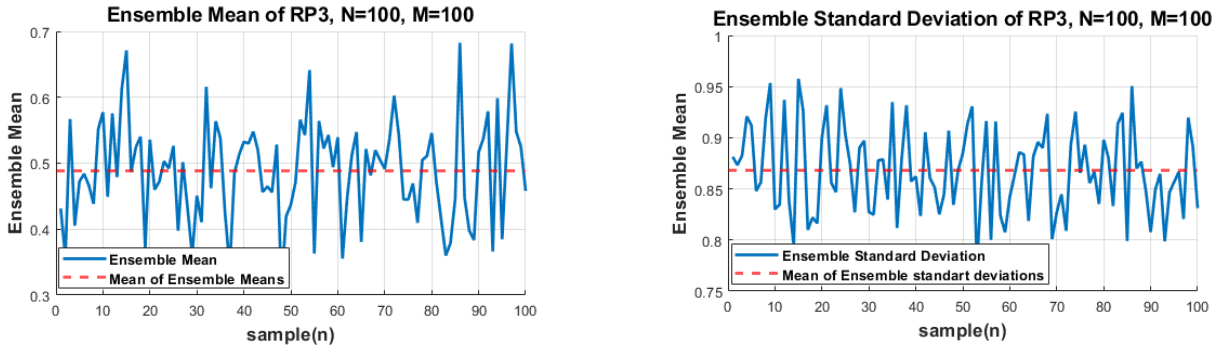


Figure 9: Ensemble mean standard deviation for `rp3`

Much like rp2, this process shows only stochastic behaviour about a bias, since it is also made up of only random variables and much like `rp2`, it also shows signs of stationarity in both its mean and variance, and thus we can approximate it as a stationary process. However, since yet again we have no information regarding higher moments, it cannot be concluded that it is Strict-Sense Stationary. To investigate the ergodicity of the process, Table 3 is inspected. There is clearly agreement between sample estimates, and since the process has already been concluded to be stationary, it satisfies the definition of ergodicity, and can be approximated as an ergodic process. Despite this, it cannot be **concluded** as ergodic, since yet again we do not have information regarding higher order moments.

	Realisation 1	**Realisation 2**	**Realisation 3**	**Realisation 4**
Mean	0.4743	0.4977	0.5285	0.4743
Standard Deviation	0.8789	0.8784	0.8728	0.8688

Table 3: Table showing sample means and standard deviations of 4, 1000 sample realisations of `rp3`

Finally, the theoretical mean and standard deviation for this process are attempted to be found, beginning with a mathematical definition of the process:

$$x[n] \;=\; 3(w[n]-0.5)+0.5 \;=\; 3g[n]+0.5 \tag{15}$$

Now calculating the mean and standard deviation for this process:

$$\mathbb{E}\{x[n]\} \;=\; 3\overbrace{\mathbb{E}\{g[n]\}}^{0}+\mathbb{E}\{0.5\} \;=\; 0.5 \tag{16}$$

For standard deviation,

$$\begin{aligned}\mathbb{E}\{x^2[n]\} \;=\; \mathbb{E}\{(3g[n]+0.5)^2\} \;&=\; 9\overbrace{\mathbb{E}\{g^2[n]\}}^{1/12}+3\overbrace{\mathbb{E}\{g[n]\}}^{0}+\mathbb{E}\{0.25\} \;=\; 1\\ \implies\quad \sigma_X^2 &= 1-(0.5)^2=\tfrac{3}{4}\end{aligned} \tag{17}$$

1.3 Estimation of probability distributions

1.3.1 `pdf` function definition: Question 1

The `pdf` function has been defined as follow:

```
function pdf(samples)
    size=length(samples);
    if size<1000
        bins=10;

    elseif size<5000
        bins =20;
    elseif size<50000
        bins = 50;
    else
        bins = 200;
    end
    [counts, edges] = histcounts(samples, bins);
    sum_counts = sum(counts);
    width = edges(2)-edges(1);
    area = sum_counts*width;
    y = counts/area;
    histogram('BinCounts', y, 'BinEdges', edges)
```

```
19      xlabel("x");
20      ylabel("pdf(x)");
21  end
```

It was then tested on the Normal Distrubution for varying sample sizes, shown in Figure 10 with the theoretical distribution superposed onto the plot. It shows good results.

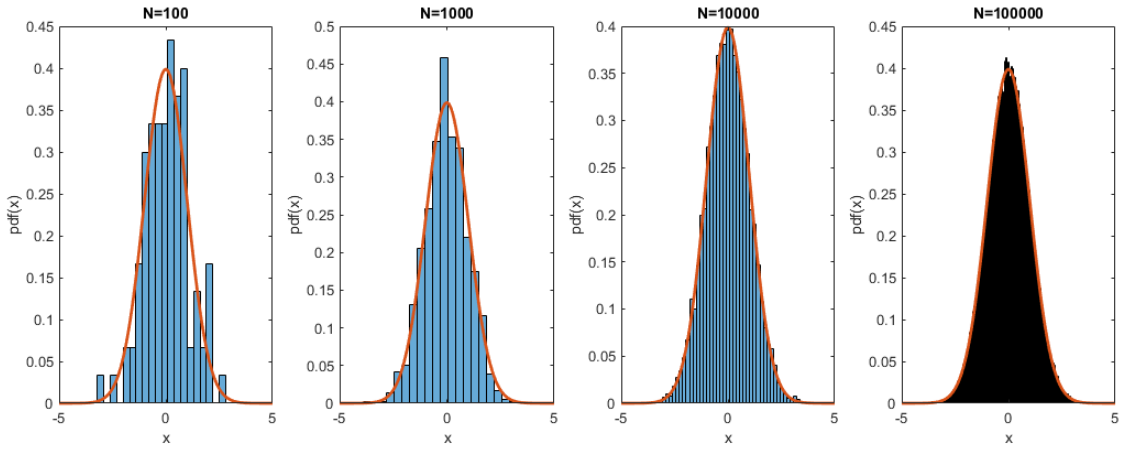


Figure 10: estimation of the PDF for `randn(1,N)`

1.3.2 Probability Density Function estimation of Stationary Ergodic processes: Question 2

Random Process 3 was the only process that satisfied the conditions of ergodicity, we test the function `pdf` on it. Clearly the estimate converges to the theoretical PDF as sample size increases. The theoretical pdf of `rp3` is formally defined in equation 18, and is superposed onto the the plot in Figure 11.

$$pdf(x) = \mathcal{U}(-1,2) = \begin{cases} 1, & \text{if } -1 \leq x \leq 2 \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

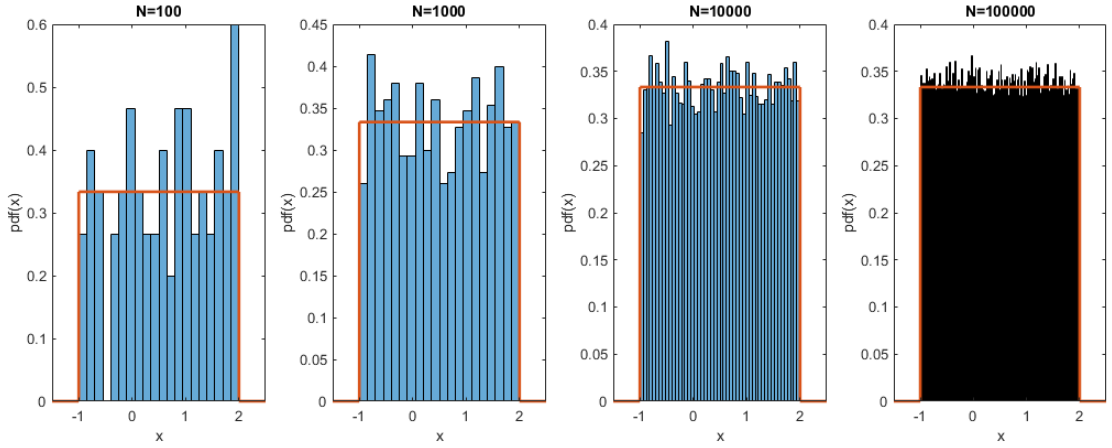


Figure 11: estimation of the PDF for `rp3`

1.3.3 Probability Density Function estimation of Non-Stationary processes: Question 3

The inherent difficulty with estimating the PDF of a non-stationary process comes from the dependence of the MATLAB functions `histcounts` on using time-averages to construct the histogram. Although `rp1`

is non-stationary, its PDF can be characterised as a function of time/sample size and so can be realised. It's PDF can be defined as:

$$pdf(x) \sim \mathcal{U}(0, 0.02N) \tag{19}$$

Where N is the number of samples taken, also assuming the samples have been taken from $t = 0$. This yields the following PDF estimations for varying N,

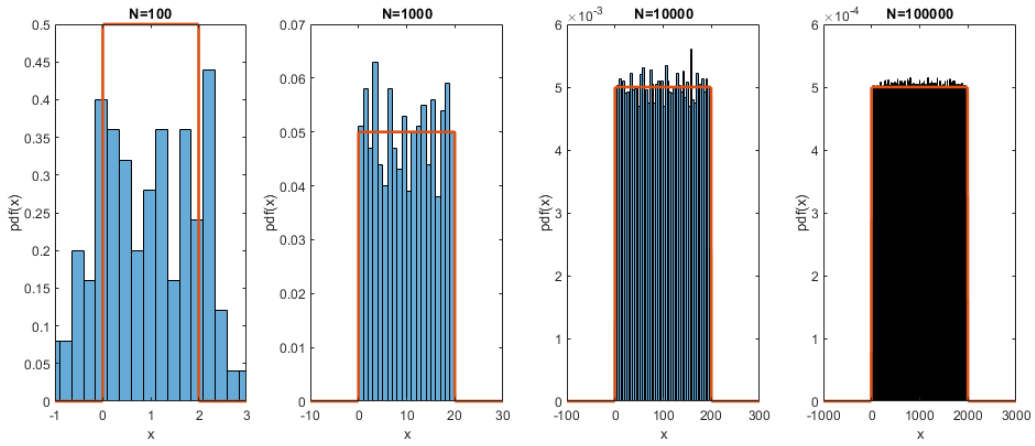


Figure 12: estimation of the PDF for `rp1`

We also test the ability of the function `pdf` to estimate the pdf with another non-stationary process. This process is defined by the vector `v`, where `v(1:500)` $\sim \mathcal{U}(-0.5, 0.5)$, and `v(501:1000)` $\sim \mathcal{U}(0.5, 1.5)$. The estimations are shown below,

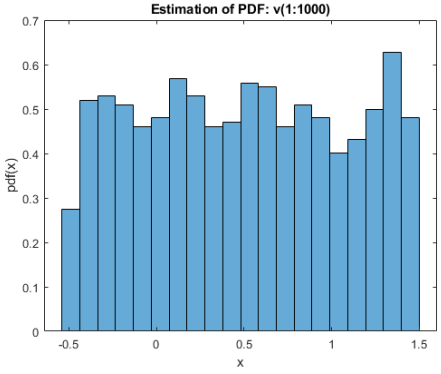


Figure 13: incorrect estimation of v(1:1000)

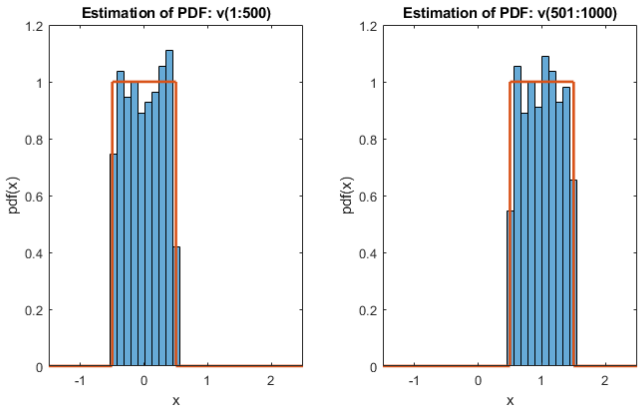


Figure 14: correct estimation of v(1:500), v(501:1000)

We can clearly see that if an estimate is made using the whole of the vector `v(1:1000)`, we obtain the incorrect result, but if this is split into two Stationary processes, then we obtain the correct estimations as seen in Figure 14. Thus, we can conclude that if we can obtain the decomposition of a process into separate Stationary processesm then the function `pdf` may be used to estimate the pdf of a non-Stationary process.

2 Linear Stochastic Modelling

2.1 ACF of uncorrelated and correlated sequences

2.1.1 Auto-correlation of White Gaussian noise: Questions 1, 2 & 3

The unbiased estimate for N samples of the autocorrelation function is given by:

$$\hat{R}_X(\tau) = \frac{1}{N-|\tau|} \sum_{n=0}^{N-|\tau|-1} x[n]x[n+\tau], \qquad \tau = -N+1, ..., N-1 \tag{20}$$

Since White Gaussian Noise(WGN) is a Wide-Sense Stationary(WSS) process, $R_X(\tau) = R_X(-\tau)$ and so it's autocorrelation should be symmetrical in the y-axis. Furthermore, its defined such that its power spectrum is, $P_X(f) = \sigma_X^2$, where σ_X^2 is equal to the noise power, which in this case is 1. We also note that the inverse Fourier transform of the power spectrum of a signal is equal to its autocorrelation function, such that:

$$\mathcal{F}^{-1}\{P_X(f)\} = R_X(\tau) \quad \therefore \quad R_X(\tau) = \mathcal{F}^{-1}\{1\} = 1 \cdot \delta(\tau) \tag{21}$$

For the autocorrelation of WGN we expect an impulse of height 1 at $\tau = 0$. The unbiased estimate for a 1000-sample realisation of WGN are shown in Figure 15.

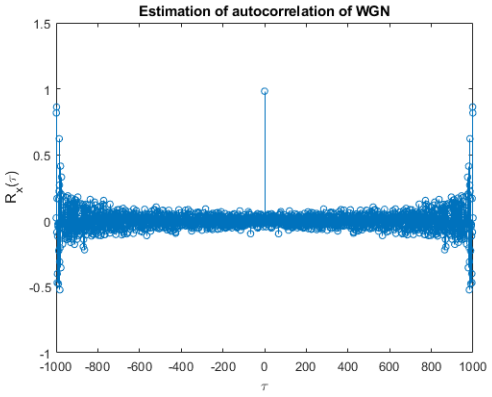


Figure 15: Autocorrelation for $|\tau| < 1000$

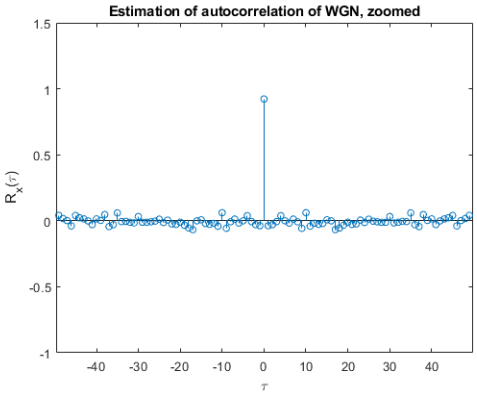


Figure 16: Autocorrelation for $|\tau| < 50$

It can clearly be seen that for smaller lags, the estimate of the autocorrelation is much closer to the theoretical behaviour that is expected. If we inspect equation 20, as $|\tau|$ approaches N, fewer and fewer data points are being used to calculate an estimate, meaning it's estimate will become unreliable, and its performance will begin to break down. The estimator shows good performance for lags between $\pm\frac{7N}{10}$.

2.1.2 Auto-correlation of Moving Average filter: Questions 4 & 5

The Moving Average(MA) filter makes use of the flat spectrum of white noise, $X(z)$ to model stochastic processes. An infinite order FIR/MA filter, $H(z)$, can always be obtained such that when white noise is filtered by it, the spectrum of the stochastic process, $Y(z)$, can be modelled. This means that:

$$Y(z) = H(z) \cdot X(z) \quad \text{where} \quad H(z) = \sum_{m=0}^{\infty} b_m z^{-m} \tag{22}$$

taking the inverse Z-transform, it follows that:

$$y[n] = \sum_{m=0}^{\infty} b_m x[n-m] \tag{23}$$

Note the autocorrelation of the output of this system $R_Y(\tau)$ is given by:

$$R_Y(\tau) = R_X(\tau) * R_h(\tau) \tag{24}$$

Since a Moving Average Filter is a Linear Time-invariant system, the following property holds:

$$P_h(f) = |H(f)|^2 \quad \text{where} \quad \mathcal{F}^{-1}\{P_h(f)\} = R_h(\tau)$$

$$P_h(f) = |H(f)|^2 = \left|\sum_{m=0}^{M} b_m e^{-j2\pi fm}\right|^2 \quad \Longrightarrow \quad R_Y(\tau) = \mathcal{F}^{-1}\left\{\left|\sum_{m=0}^{M} b_m e^{-j2\pi fm}\right|^2\right\} * R_X(\tau) \tag{25}$$

Where $P_h(f)$ is the power spectrum of the impulse response of the system, M is equal to the order of the MA filter minus 1. In the case of a filter of order 9 (using the definition of 'order' outlined in guidelines), P is equal to 8 and is denoted by MA(8), its response is the first graph in Figure 17.

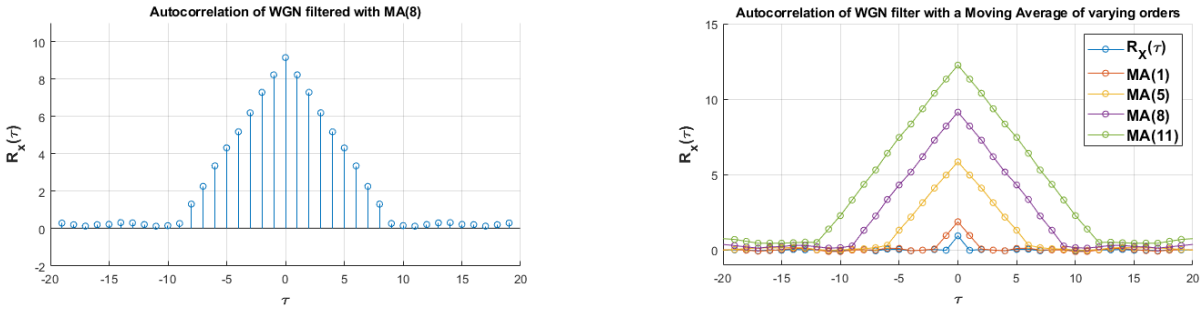


Figure 17: Autocorrelation of WGN filtered with varying orders of Moving Average filters

To explain the results seen in Figure 17, we consider equation 25 for an MA(1) filter, such that:

$$P_h(f) = \left|\sum_{m=0}^{1} b_m e^{-j2\pi fm}\right|^2 = \left|b_0 + b_1 e^{-j2\pi fm}\right|^2 = |b_0 + b_1 cos(2\pi f) - jb_1 sin(2\pi f)|^2$$

$$= (b_0 + b_1 cos(2\pi f))^2 + b_1^2 sin^2(2\pi f) = b_0^2 + 2b_0 b_1 cos(2\pi f) + b_1^2(cos^2(2\pi f) + sin^2(2\pi f))$$

$$= b_0^2 + b_1^2 + 2b_0 b_1 \frac{e^{2j\pi f} + e^{-2j\pi f}}{2} \quad \therefore \; R_h(\tau) = \mathcal{F}^{-1}\left\{b_0^2 + b_1^2 + b_0 b_1 e^{2j\pi f} + b_0 b_1 e^{-2j\pi f}\right\}$$

$$= (b_0^2 + b_1^2)\delta(\tau) + b_0 b_1 \delta(\tau - 1) + b_0 b_1 \delta(\tau + 1), \quad b_0, b_1 = 1 \quad \Longrightarrow \quad R_Y(\tau) = (2\delta(\tau) + \delta(\tau - 1) + \delta(\tau + 1)) * R_X(\tau)$$

$$R_Y(\tau) = 2R_X(\tau) + R_X(\tau + 1) + R_X(\tau - 1) \tag{26}$$

For WGN, $R_X(\tau) = \delta(\tau)$, which is the results that is seen in Figure 17, this can be extended for a general MA filter of order P+1, giving:

$$R_Y(\tau) = max\{0, (P + 1 - |\tau|)R_X(\tau)\} \tag{27}$$

2.2 Cross-correlation function

2.2.1 Cross-correlation of WGN and filtered WGN: Question 1

The cross-correlation function between the input and output of an LTI filter is given by:

$$R_{XY}(\tau) = h(\tau) * R_X(\tau) \tag{28}$$

For WGN, its autocorrellation function is equal to $\delta(\tau)$ as shown in Equation 21, thus:

$$R_{XY}(\tau) = h(\tau) * \delta(\tau) = h(\tau) \tag{29}$$

In the case of MA(8), $h(\tau)$ is given by:

$$h(\tau) = \sum_{m=0}^{8} b_m \delta(\tau - m) \quad b_m = 1 \quad \forall\, m \implies R_{XY}(\tau) = h(\tau) = \sum_{m=0}^{8} \delta(\tau - m) \tag{30}$$

This result is compared with the estimate in Figure 18, which confirms this result.

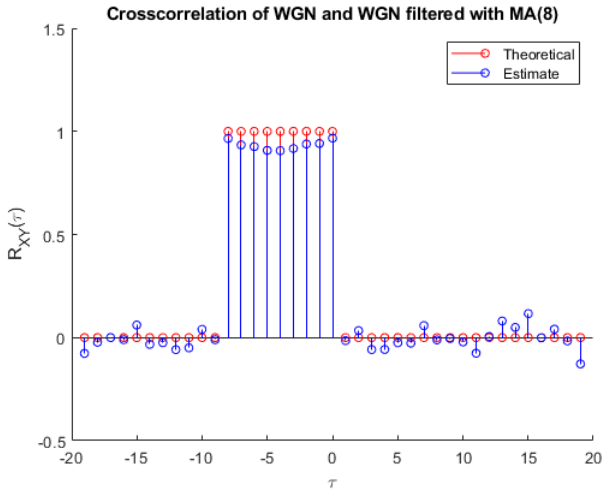


Figure 18: Cross-correlation of input and out in an LTI system

2.2.2 Cross-correlation for System Identification: Question 2

As seen in equation 29, if the input into an LTI system is WGN, the cross-correlation function is equal to the impulse response of the LTI system. The impulse response contains all the information regarding an LTI system and so the cross-correlation in this case is sufficient for system identification. Furthermore, as seen in equation 30, the number of non-zero impulses seen in the cross-correlation function is equal to the order of the filter, therefore, if the order increases so does the number of non-zero impulses that are seen.

2.3 Autoregressive Modelling

2.3.1 Stability of AR processes: Question 1

In the previous section, MA filters were explored, since MA filters are essentially FIR filters, all poles are located at the origin and thus the system is always stable. Autoregressive(AR) models are all pole filters and so poles can be located anywhere on the z-plane and so the stability of the system must be considered. For stability, all poles must be located within the unit circle. We explore the conditions on the AR(2)

coefficients to ensure stability of the process. An AR(2) process is given by:

$$x[n] = a_1 x[n-1] + a_2 x[n-2] + w[n], \quad w[n] \sim \mathcal{N}(0,1). \tag{31}$$

Taking the Z Transform:

$$X(z) = X(z) \cdot (a_1 z^{-1} + a_2 z^{-2}) + W(z) \quad \therefore \quad X(z) = \frac{W(z)}{1 - a_1 z^{-1} - a_2 z^{-2}} = \frac{W(z) \cdot z^2}{z^2 - a_1 z - a_2} \tag{32}$$

The poles are given by the roots of the polynomial in the denominator, they are denoted by p_1, p_2. Since for stability it is required at that the poles lie within the unit circle, the following inequality must hold:

$$p_1, p_2 = \left| \frac{a_1 \pm \sqrt{a_1^2 + 4a_2}}{2} \right| < 1 \tag{33}$$

Considering real poles, it follows that:

$$\frac{a_1 + \sqrt{a_1^2 + 4a_2}}{2} < 1, \quad \frac{a_1 - \sqrt{a_1^2 + 4a_2}}{2} > -1$$

Considering the first Inequality:

$$\sqrt{a_1^2 + 4a_2} < 2 - a_1 \implies a_1^2 + 4a_2 < (2 - a_1)^2 \implies a_2 < 1 - a_1 \tag{34}$$

Now for the second inequality:

$$a_1 + 2 > \sqrt{a_1^2 + 4a_2} \quad \implies ; \quad (a_1 + 2)^2 > a_1^2 + 4a_2 \quad \implies \quad a_2 < 1 + a_1 \tag{35}$$

We can write the transfer function of the process in terms of p_1 and p_2

$$X(z) = \frac{z^2}{(z - p_1)(z - p_2)} = \frac{z^2}{z^2 - (p_1 + p_2) + p_1 p_2}$$

Comparing this to the result seen in 32 it follows that:

$$p_1 p_2 = -a_2, \quad |p_1| < 1, \quad |p_2| < 1 \implies |a_2| < 1$$

Therefore the conditions for stability are summarised in the following 3 inequalities:

$$a_2 < 1 - a_1, \quad a_2 < 1 + a_1, \quad -1 < a_2 < 1$$

This is confirmed with the results shown in Figure 19.

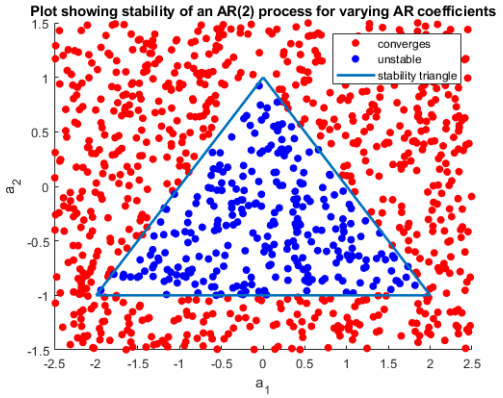


Figure 19: stability of AR(2) process based on coefficient values

2.3.2 ACF of Sunspot time series: Question 2

The autocorrelation of varying sample sizes, $N = 5, 20, 250$, from the Sunspot time series is taken, once for the raw data and once for the data with a zero-mean transformation. This is inspected in Figure 20:

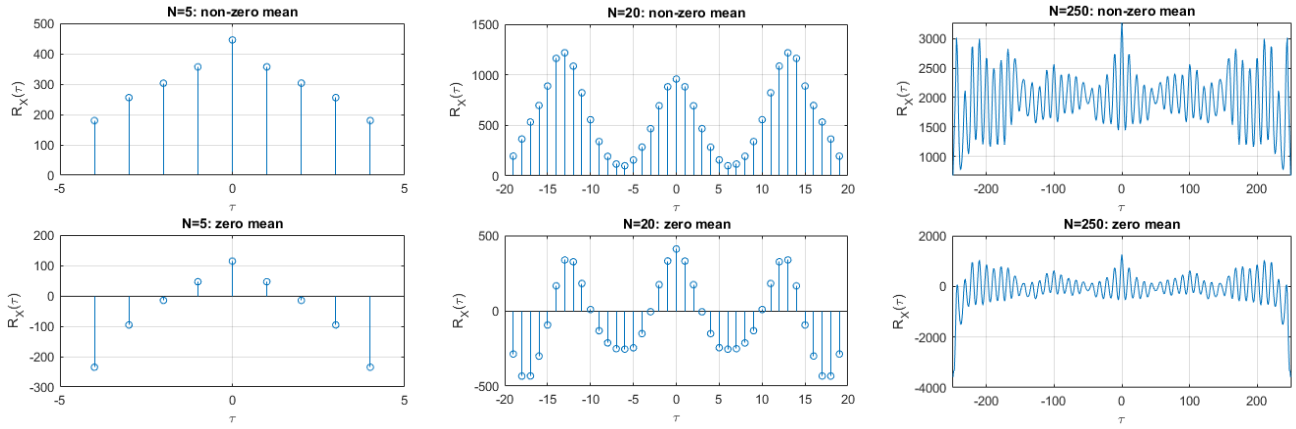


Figure 20: Auto correlation of the Sunspot series for varying sample sizes, and for non-zero and zero-mean forms of the data.

Removing the mean from the data produces an ACF that allows for a better understanding of the true similarity between the delayed samples of a the process. The same issue seen in Section 2.1 is of course still persistent in this case, as τ approaches each of the values of N, the estimation of the autocorrelation becomes more and more unreliable and unpredictable.

2.3.3 Yule-Walker equations: Question 3

Extending equation 31 to an AR(p) process, the difference equation is given as follows:

$$x[n] = a_1 x[n-1] + a_2 x[n-2] + \ \ldots \ a_p x[n-p] + w[n], \quad w[n] \sim \mathcal{N}(0,1). \tag{36}$$

We multiply both sides by $x[n-k]$ and take the expectation, such that:

$$\mathbb{E}\{x[n]x[n-k]\} = \mathbb{E}\{a_1 x[n-1]x[n-k] + a_2 x[n-2]x[n-k] + \ \ldots \ a_p x[n-p]x[n-k] + w[n]x[n-k]\} \tag{37}$$

For a stationary process $r_{xx}[|a-b|] = \mathbb{E}\{x[n-a]x[n-b]\}$, noting that $\mathbb{E}\{w[n]x[n-k]\} = 0, \quad k \neq 0$, it thus follows that for $k = 1, 2, \ \ldots p$:

$$\begin{aligned} r_{xx}[1] &= a_1 r_{xx}[0] + a_2 r_{xx}[1] + \ \ldots \ a_p r_{xx}[p-1] \\ r_{xx}[2] &= a_1 r_{xx}[1] + a_2 r_{xx}[0] + \ \ldots \ a_p r_{xx}[p-2] \\ &\vdots \\ r_{xx}[p] &= a_1 r_{xx}[p-1] + a_2 r_{xx}[p-2] + \ \ldots \ a_p r_{xx}[0] \end{aligned}$$

Expressing this is Matrix form, we obtain the vector of AR coefficients, $\mathbf{a}$ as follows:

$$\mathbf{r}_{xx} = \mathbf{R}_{xx}\mathbf{a} \implies \mathbf{a} = \mathbf{R}_{xx}^{-1}\mathbf{r}_{xx} \tag{38}$$

$$\mathbf{r}_{xx} = \begin{bmatrix} r_{xx}[1] \\ r_{xx}[2] \\ \vdots \\ r_{xx}[p] \end{bmatrix} \qquad \mathbf{R}_{xx} = \begin{bmatrix} r_{xx}[0] & r_{xx}[1] & \ldots & r_{xx}[p-1] \\ r_{xx}[1] & r_{xx}[0] & \ldots & r_{xx}[p-2] \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}[p-1] & r_{xx}[p-2] & \ldots & r_{xx}[0] \end{bmatrix}$$

The inbuilt MATLAB function `aryule` uses equation 38 to calculate the AR coefficients for a model order p. This will used henceforth in this section to obtain AR coefficients.

The Partial Correlation Function (PACF) of a model order k is found by first calculating the AR coefficients for the model order k using the Yule-Walker equations, denoted by the set $\mathbf{a}_k$. The k^{th}/final coefficient from this set is taken, denoted by a_{kk}, this is PACF for a model order k. The PACF is a better

indicator than the ACF of the model order, it is essentially measuring the significance of increasing the order to k. The PACF for the both the raw and normalised versions of the Sunspot time series are shown in Figure 21 for model orders up until $p = 10$.

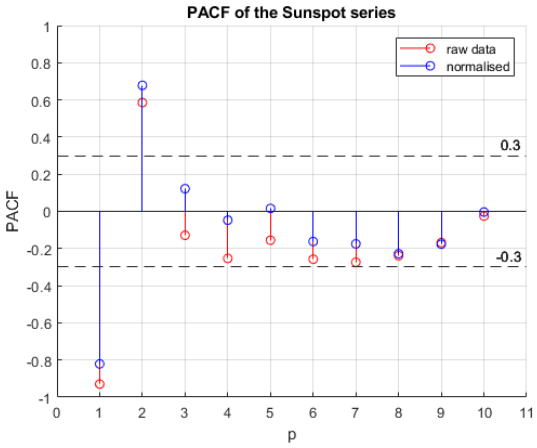


Figure 21: PACF of the Sunspot series for model orders up until $p = 10$

Both forms of the Sunspot series indicate clearly that the sunspot series can be well modelled by an AR(2) process. The main discrepancy between the two is that the normalised data, for orders greater than $p = 2$, the PACF is closer to 0, giving a stronger indication that the process is AR(2).

2.3.4 MDL, AIC, AIC$_c$: Question 4

For an AR(p) process, the predictions are made with the following difference equation, where $\hat{x}[n]$ is the prediction at time point n, and $x[n-k]$ is the true value of the time-series k samples ago:

$$\hat{x}[n] = a_1 x[n-1] + a_2 x[n-2] + \ \ldots \ + a_p x[n-p] \tag{39}$$

$$\hat{X}(z) = X(z)(a_1 z^{-1} + a_2 z^{-2} + \ \ldots \ + a_p z^{-p}) \tag{40}$$

The Minimum Description Length, Akaike information criterion(AIC) and the corrected AIC(AIC$_c$) are calculated for model orders up until $p = 10$ for the Sunspot time series, shown in Figure 22. These metrics are essentially error terms with added penalty for model order to account for overfitting and are used to indicate the order of the AR process. They are calculated as following, note that E_p is equal to the cumulative squared error between the prediction and the true value:

$$\text{MDL} = \log E_p + \tfrac{p \log N}{N} \tag{41}$$

$$\text{AIC} = \log E_p + \tfrac{2p}{N} \tag{42}$$

$$\text{AIC}_c = \text{AIC} + \tfrac{2p(p+1)}{N-p-1} \tag{43}$$

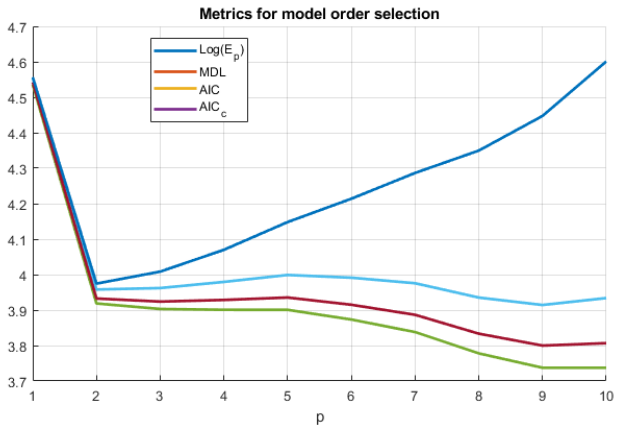


Figure 22: Model order indicators, MIC, AIC, AIC_c

For the MDL and AIC metrics, a clear minimum is shown at $p = 9$. The AIC_c is better suited for smaller datasets since it introduces an additional penalty term. The Sunspot time-series has only 288 data points, therefore it is a relatively small dataset, suggesting the AIC_c is a more suitable metric. Furthermore, it must be noted that the minimum shown on the AIC_c curve is far more substantial and clear than those seen with MDL and AIC, suggesting this is the best model order to select. The value of AIC_c at $p = 2$ is ≈ 3.98 and at $p = 9$ it is ≈ 4.45, a difference of ≈ 0.47, whereas for AIC the difference between its value at $p = 2$ and $p = 9$ is ≈ 0.06 and for MDL it is ≈ 0.13. To add to this, all metrics show their most significant 'elbow' at $p = 2$ further suggesting that this is the most suitable model order. Since the increase in performance of the AR(9) model (in terms of MDL, AIC) compared to AR(2) is insignificant, choosing an AR(9) model would suggest we are over-modelling/over-fitting to the training data and thus poor out-of-sample performance would be expected. Higher order modelling also comes at a far greater computational cost. It is thus concluded that an AR(2) model is most suitable to model the sunspot series.

2.3.5 Prediction Horizon: Question 5

AR modelling is an extremely useful tool for time series prediction, in the following task, 3 different AR models are generated, AR(1), AR(2), AR(10) and are each used to make predictions of the Sunspot time series data at 1, 2, 5, and 10 steps ahead. The results are shown in Figure 23.

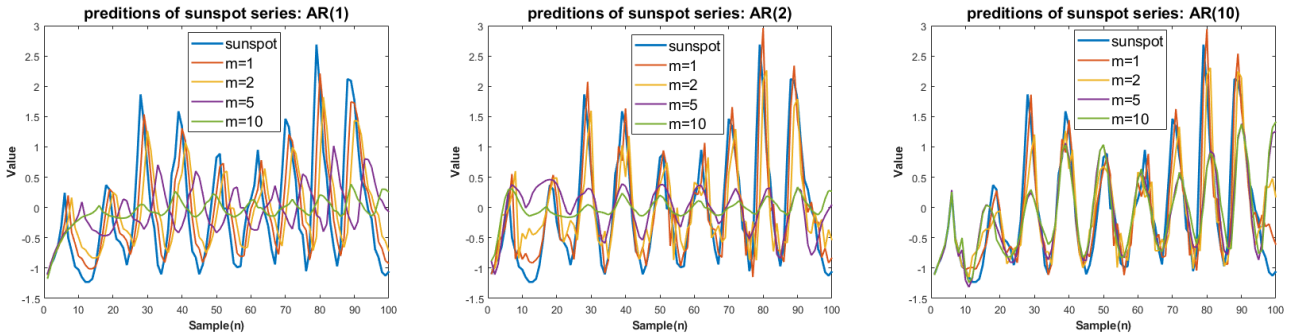


Figure 23: Predictions of the Sunspot series for different AR model orders

All 3 models perform well when predicting 1 and 2 steps ahead, however, for the AR(1) and AR(2) models, when predicting 5 and 10 steps ahead, the model breaks down. We examine how predictions are made in the case of AR(2), m=5(5 steps ahead) to understand why:

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

$$\vdots$$

$$\hat{x}[n+5] = a_1\hat{x}[n+4] + a_1\hat{x}[n+3]$$

The further and further that predictions are made, the less and less context that the model has of the original data, predictions are based on predictions and so these errors cascade, leading to a breakdown in performance. For the prediction horizons greater than the model order, eventually predictions are going to be based off of predictions alone. For the AR(10) model, the model performs better when making predictions both 5 and 10 steps ahead for the reasons just discussed. It can thus be suggested that if predictions are to be made with longer horizons, a higher order model should be opted for. This does however come at a greater computational cost.

2.4 Cramer-Rao Lower Bound

2.4.1 NASDAQ financial Index: Part a)

To understand what order AR model is most suitable to describe the NASDAQ financial Index, we use metrics seen in Section 2.3 such as the PACF, MDL, AIC, and AIC$_c$, shown in Figure 24 and Figure 25. After a model order of 1, the PACF drops to approximately 0, furthermore, all the metrics in Figure 25 show a clear minimum at p=1. The AR(1) is sufficient to describe the daily returns of the the index.

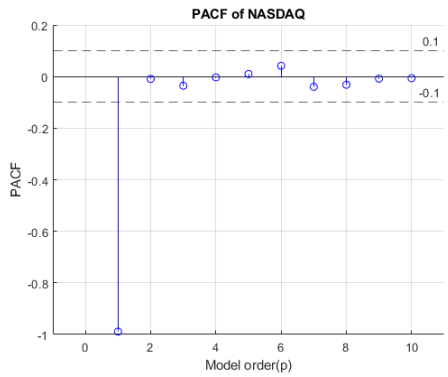


Figure 24: PACF of NASDAQ

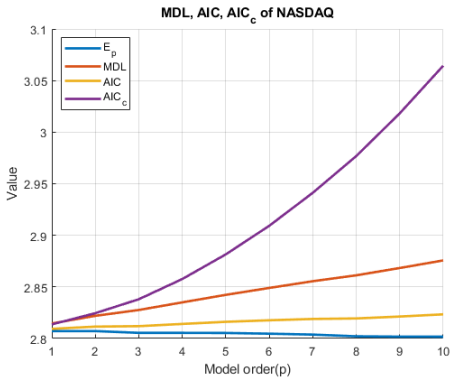


Figure 25: MDL, AIC, AIC$_c$ of NASDAQ

2.4.2 Fischer Information Matrix Estimation: Parts b) and c)

Consider an input signal $\mathbf{x} \in \mathbb{R}^N$ and a vector of unknown parameters $\boldsymbol{\theta} = [\theta_1, \theta_2, \ \ldots \ , \theta_p]^T$. If the likelihood function, $p(\mathbf{x};\boldsymbol{\theta})$, is known, and the estimator is ubiased such that, $\mathbb{E}\{\hat{\theta}\} = \boldsymbol{\theta}$, the Cramer-Rao lower bound(CRLB) has the form:

$$var(\hat{\theta}_i) \geq [\mathbf{I}^{-1}(\boldsymbol{\theta})]_{ii}, \qquad [\mathbf{I}(\boldsymbol{\theta})]_{ij} = -\mathbb{E}\left\{\frac{\partial^2 p(\mathbf{x};\boldsymbol{\theta})}{\partial\theta_i \partial\theta_j}\right\} \tag{44}$$

for $i, j = 1, 2, \ ..., p$ such that $\mathbf{I}(\theta) \in \mathcal{R}^{p\times p}$. We aim to find the CRLB of the unknown parameters of an AR(p) process(defined in equation 36), that is $\boldsymbol{\theta} = [a_1, a_2, \ \ldots \ a_p, \sigma^2]$. To find the true value of the CRLB, the covariance matrix needs to be inverted, this can prove to be a difficult task. Instead, the asymptotic CRLB can be used, which is derived from the power spectrum of the AR(p) model.

$$\hat{P}_X(f;\boldsymbol{\theta}) = \frac{\hat{\sigma}^2}{\left|1 - \sum_{m=1}^{p} \hat{a}_m e^{-j2\pi f m}\right|^2} \tag{45}$$

The log-likelihood can now be represented/approximated as:

$$\ln\left[\hat{P}_X(f;\boldsymbol{\theta})\right] = \ln[\hat{\sigma}^2] - 2\ln\left[\left|1 - \sum_{m=1}^{p} \hat{a}_m e^{-j2\pi fm}\right|\right] \tag{46}$$

and the elements of the Fischer matrix can be approximated as:

$$[\mathbf{I}(\boldsymbol{\theta})]_{ij} = \frac{N}{2}\int_{-\frac{1}{2}}^{\frac{1}{2}} \frac{\partial \ln\left[\hat{P}_X(f;\boldsymbol{\theta})\right]}{\partial\theta_i} \frac{\partial \ln\left[\hat{P}_X(f;\boldsymbol{\theta})\right]}{\partial\theta_j} df \tag{47}$$

As concluded in section 2.4.1 the NASDAQ financial index is best represented by a AR(1) process, therefore, $\boldsymbol{\theta} = [a_1, \sigma^2]$. It is given that $[\mathbf{I}(\boldsymbol{\theta})]_{11} = \frac{N r_{xx}(0)}{\sigma^2}$, $[\mathbf{I}(\boldsymbol{\theta})]_{21}$, $[\mathbf{I}(\boldsymbol{\theta})]_{12} = 0$, $[\mathbf{I}(\boldsymbol{\theta})]_{22}$ is calculated using equation 47.

$$[\mathbf{I}(\boldsymbol{\theta})]_{22} = \frac{N}{2}\int_{-\frac{1}{2}}^{\frac{1}{2}} \left(\frac{\ln[\hat{\sigma}^2] - 2\ln\left[\left|1 - \sum_{m=1}^{p} \hat{a}_m e^{-j2\pi fm}\right|\right]}{\partial\sigma^2}\right)^2 df$$

$$= \frac{N}{2}\int_{-\frac{1}{2}}^{\frac{1}{2}} \left(\frac{1}{\sigma^2}\right)^2 df = \frac{N}{2}\cdot\frac{f}{\sigma^4}\Big|_{-\frac{1}{2}}^{\frac{1}{2}} = \frac{N}{2\sigma^4}$$

Since $\mathbf{I}(\boldsymbol{\theta})$ is a diagonal matrix, it's inverse is given by taking the reciprocal of all the entries along the diagonal:

$$\mathbf{I}^{-1}(\boldsymbol{\theta}) = \begin{bmatrix} \frac{\sigma^2}{N r_{xx}(0)} & 0 \\ 0 & \frac{2\sigma^4}{N} \end{bmatrix} \tag{48}$$

And from equation 44, it follows that:

$$var(\hat{\sigma}^2) \geq [\mathbf{I}^{-1}(\boldsymbol{\theta})]_{22} = \frac{2\sigma^4}{N}, \qquad var(\hat{a}_1) \geq [\mathbf{I}^{-1}(\boldsymbol{\theta})]_{22} = \frac{\sigma^2}{N\mathbf{r}_{xx}(0)} \tag{49}$$

$$\text{since } \; r_{xx}(0) = \frac{\sigma^2}{(1-a_1^2)} \implies var(\hat{a}_1) \geq \frac{(1-a_1^2)}{N} \tag{50}$$

By looking at equation 49, it can be seen that CRLB$(\hat{\sigma}^2) \propto \sigma^4$ whereas CRLB$(\hat{a}_1) \propto \sigma^2$, it is thus expected that σ will have a more substantial impact on the CRLB of $\hat{\sigma}^2$ than it will for $\hat{a}_1$. It can also be seen that the CRLB of both parameters is inversely proportional to the number of data points N, hence for $\sigma = 1$ both will have the CRLB for all N. This is confirmed in Figure 26 and 27 where the CRLB is plotted as a heatmap with respect to N and σ.

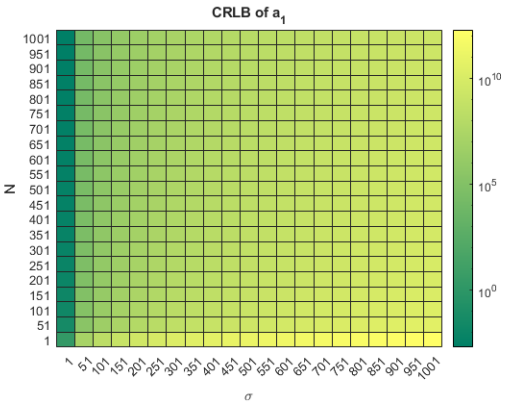


Figure 26: CRLB of a_1

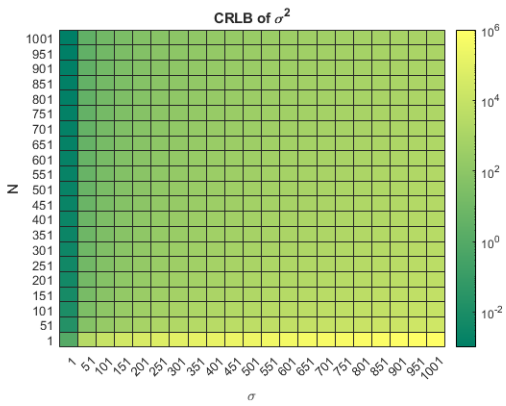


Figure 27: CRLB of σ^2

As aforementioned, the NASDAQ Financial index is best described by an AR(1) process, by solving the Yule-Walker equations, the following difference equation is obtained:

$$x[n] = -0.9887x[n-1] + w[n], \text{ where } a_1 = -0.9887 \tag{51}$$

Using equation 50, the CRLB $\hat{a}_1$ can easily be obtained, noting that for this dataset N=924:

$$var(\hat{a}_1) \geq \frac{(1-a_1^2)}{N} = \frac{(1-(-0.9887)^2)}{924} = 2.43\text{x}10^{-5}$$

By again looking at equation 50, it can be seen that as a_1 approaches unity, or as the pole(a_1) of the transfer function approaches the unit circle, the CRLB approaches 0, $|a_1| \to 1$, $\frac{(1-a_1^2)}{N} \to 0$. Furthermore, if $|a_1| > 1$, the CRLB becomes negative, this is because the pole of the transfer function now lies outside the unit circle and so the system becomes unstable and so the CRLB becomes redundant(negative variance has no semantic meaning).

2.4.3 CRLB of the Power Spectrum: Part d)

From the CRLB, the variance of the power spectrum can be bounded as:

$$var(\hat{P}_X(f;\boldsymbol{\theta})) \geq \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})^T}{\partial \boldsymbol{\theta}} \mathbf{I}^{-1}(\boldsymbol{\theta}) \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \tag{52}$$

where $\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \left[\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \theta_1}, \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \theta_2}\right]^T$, and $A(f) = 1 - a_1 e^{-j2\pi f}$. Using equation 45, we first calculate $\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$:

$$\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \theta_1} = \frac{\partial \left(\frac{\sigma^2}{|A(f)|^2}\right)}{\partial a_1} = \overbrace{\frac{\frac{\partial \sigma^2}{\partial a_1}|A(f)|^2 - \frac{\partial |A(f)|^2}{\partial a_1}\sigma^2}{|A(f)|^4}}^{\text{quotient rule}} = \frac{\frac{\partial (1-a_1 e^{-j2\pi f})(1-a_1 e^{j2\pi f})}{\partial a_1}\sigma^2}{|A(f)|^4}$$

$$= \frac{\sigma^2(e^{-j2\pi f} + e^{j2\pi f} - 2a_1)}{|A(f)|^4} = \frac{2\sigma^2(cos(2\pi f) - a_1)}{|A(f)|^4}$$

Now for θ_2:

$$\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \theta_2} = \frac{\partial \left(\frac{\sigma^2}{|A(f)|^2}\right)}{\partial \sigma^2} = \frac{1}{|A(f)|^2}$$

substituting these values into equation 52 we have:

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

$$\frac{2\sigma^4}{N}\left(\frac{2cos^2(2\pi f)(1-a_1)}{|A(f)|^8} + \frac{1}{|A(f)|^4}\right)$$

2.5 ECG from iAmp experiment: Parts a) and b)

2.5.1 PDF estimation

The RRI signal $rr[n]$ is converted to heart rate $h[n]$ from the following transformation:

$$h[n] = \frac{60}{rr[n]} \tag{53}$$

To obtain a smoother estimate of the heart rate, every 10 samples of $h[n]$ are averaged and scaled to obtain the following 2 new formulas:

$$\hat{h}_1[n] = \frac{1}{10}\sum_{i=1}^{10} 1 \cdot h[10n+i] \qquad \hat{h}_2[n] = \frac{1}{10}\sum_{i=1}^{10} 0.6 \cdot h[10n+i]$$

The mean and variance for the 3 signals are shown in Table 1

Signal	Mean	Variance
$h[1]$	101.7887	32.8653
$\hat{h}_1[n]$	101.7429	29.9180
$\hat{h}_2[n]$	61.0458	10.7705

Table 4: Table showing the Mean and Variance of h[1], $\hat{h}_1[n]$, $\hat{h}_2[n]$

Since each data point of $\hat{h_1}[n]$ is the average of 10 data points of h[n], the expected value of both sequences must be identical. The discrepancy seen between the 2 values is down to the fact that 6 data points were omitted from h[n] when constructing $\hat{h_1}[n]$. Furthermore, since there is some averaging/'smoothing' of h[n] when calculating $\hat{h_1}[n]$, it is expected to have a smaller variance, which is observed. The mean and variance of $\hat{h}_2[n]$ can be derived directly from $\hat{h}_1[n]$ since $\hat{h}_2[n] = 0.6\hat{h}_1[n]$:

$$\mathbb{E}\{\hat{h_2}\} = \mathbb{E}\{0.6\hat{h_1}\} = 0.6\,\mathbb{E}\{\hat{h_1}\} = 0.6 \cdot 101.7429 = 61.0458 \quad \text{as required.} \tag{54}$$

$$\text{Var}\{\hat{h_2}\} = \text{Var}\{0.6\hat{h_1}\} = 0.6^2\text{Var}\{\hat{h_1}\} = 0.36 \cdot 29.9180 = 10.7705 \quad \text{as required.} \tag{55}$$

Using a slightly modified version(adjusted bin numbers) of the `pdf` function defined in Section **??**, the PDFs of $h[n], \hat{h}_1[n],$ and $\hat{h}_2[n]$ are shown in Figure 28:

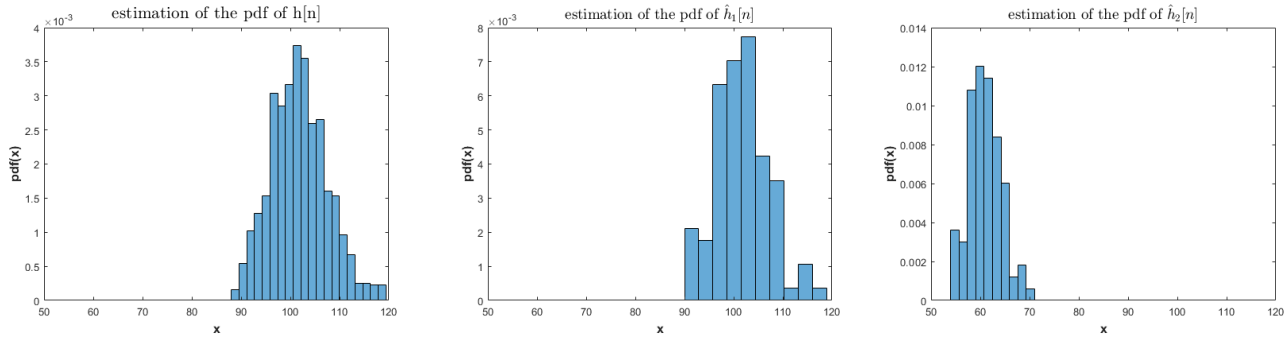


Figure 28: PDF estimation of $h[n]$, $\hat{h}_1[n]$, and $\hat{h}_2[n]$

2.5.2 AR modelling of heart rate: Parts c) and d)

The ACF of the RRI data for the 3 separate trials, RRI1, RRI2 and RRI3, are shown in Figure 29. For an MA(p) process as defined in Section 2.1.2 will exhibit an ACF that is finite and is 0 beyond a delay of p, this behaviour is not exhibited in any of the ACFs of the RRI data. AR processes will exhibit an ACF that is infinite in length(if $|\tau| \to$ inf is measured) and will consist of a mixture of damped exponentials and/or sine waves, this profile matches the behaviour of each of the ACFs and thus it can be concluded that they are AR processes. In Figure 30 the PACF, MDL, AIC and AIC$_c$ are plotted to help select the most suitable model order for each of the datasets.

- **RRI1:** Looking at the PACF, after p=4, coefficients fall very close to 0. The MDL, AIC, and AIC$_c$ all show minimums for a model order p, thus it can be concluded that RRI1 is most suitably

represented by an AR(4) process.

- **RRI2:** After p=2 the coefficients of the PACF fall close to 0. The coefficient at p=10 could however be considered significant, so we look towards the other metrics select the model order. As seen in Section 2.3.4, the MDL and AIC are minimum at p=10, however the minimum exhibited for the AIC_c at p=2 is more prominent, suggesting that if p=10 were selected, it would be overfitting to the given data.
- **RRI3:** The coefficients for the PACF of RRI3 fall gradually thus it is difficult to conclude from the PACF alone what model order to select. The MDL, AIC, and AIC_c all show mimimums at p=3, suggesting that this is the most suitable model order to select.

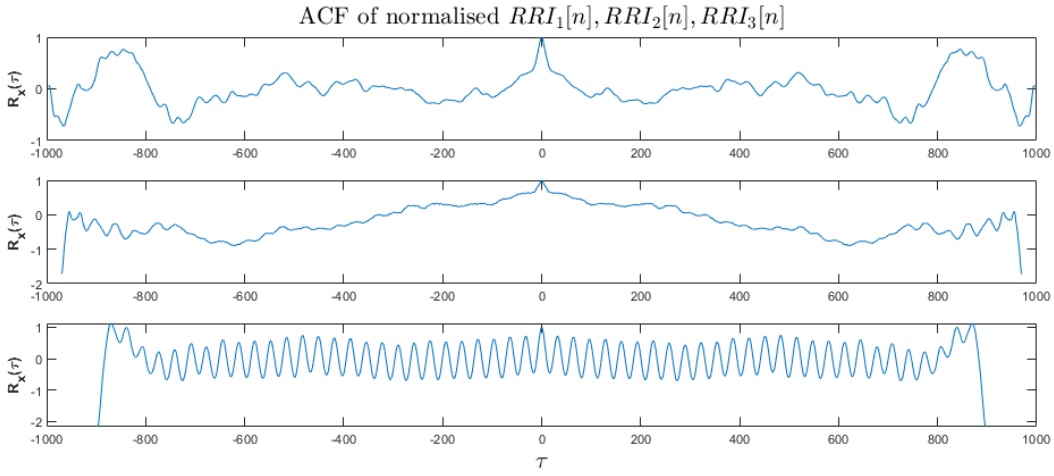


Figure 29: ACF of RRI_1, RRI_2 and RRI_3

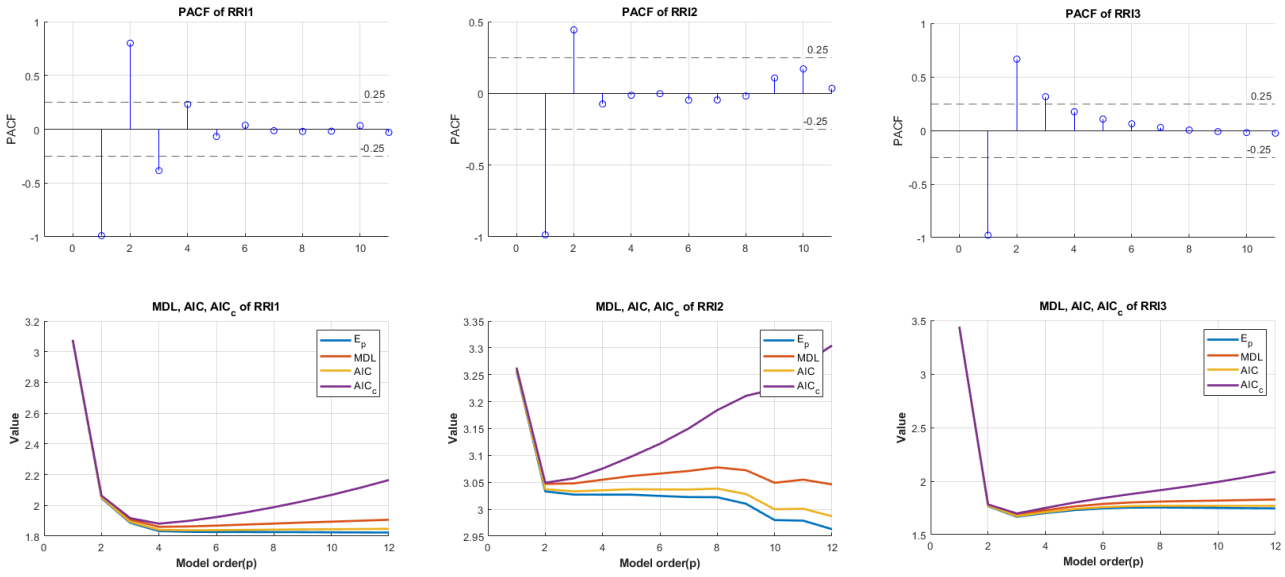


Figure 30: Plots that help indicate the AR model order of RRI1, RRI2, and RRI3

3 Spectral estimation and modelling

The Power Spectral Density(PSD) refers to the distribution of the power of an ergodic stochastic process X_n across all frequencies and is given by the absolute value of the Fourier transform of its autocorrelation function, such that:

$$P_X(f) = \left| \sum_{\tau=-\infty}^{\infty} R_X(\tau) e^{-j2\pi f\tau} \right|, \qquad f \in [0,1] \tag{56}$$

where $R_X(\tau)$ is the ACF of X_n and f is the normalised frequency. This calculation however is summed from $-\infty$ to ∞ and thus requires complete knowledge of the autocorrelation function of X_n. The PSD can instead be approximated by a finite number of samples using its periodogram, the squared absolute value of the Discrete-time Fourier Transform(DTFT), and is defined by:

$$\hat{P}_X(f) = \frac{1}{N} \left| \sum_{n=0}^{N-1} x[n] e^{-j2\pi f \frac{n}{N}} \right| \tag{57}$$

3.0.1 Periodogram estimate: Question 1

Equation 57 is translated directly into a self-defined MATLAB function `pgm`. It is tested with an 128, 256, and 512 sample realisations of WGN, the results are displayed in Figure 31, along with the averaged periodogram and the theoretical PSD, which will be discussed in the following section. It can be seen that the PSD is symmetrical about 0.5 in the normalised frequency domain. It is also observed that an N-sample sequence corresponds to an N point periodogram, from this it follows that the greater the length N, the more precise the periodogram becomes. Finally, it can be seen that the periodogram performs badly with respect to the theoretical PSD. This is because we are essentially windowing an infinitely long process, if the window is insufficiently long to capture the crucial properties of the process, the DTFT and thus the PSD will not resemble the theoretical PSD.

3.1 Averaged periodogram estimates

3.1.1 Periodogram filtered with Moving Average: Question 1

The PSD estimates in Section 3.0.1 are 'smoothed' using a moving average filter with coefficients `0.2*[1 1 1 1 1]`. The PSD estimation at each frequency point, f_n, is made up of an average of its PSD estimation and the PSD estimation of the 4 previous frequency points. In Figure 31, it appears that this method is improving the estimate of the PSD, however, this method would not work as well for a non-constant theoretical PSD. Since it is an average of itself and the 4 previous values, if, for example, there is a sharp change in the PSD, the filtered periodogram would be unable to capture this sharp change, due to the weighting of the previous 4 values.

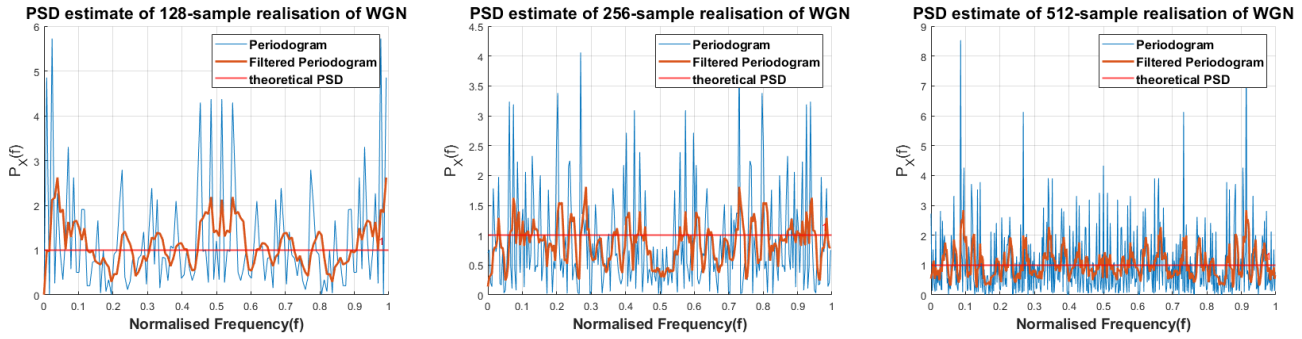


Figure 31: PSD estimation of WGN

3.1.2 Averaged Periodogram: Questions 2 & 3

A 1024-sample sequence of WGN is generated and divided into 8 non-overlapping 128-sample segments, the Periodogram of each of these segments is taken and shown in Figure 32.

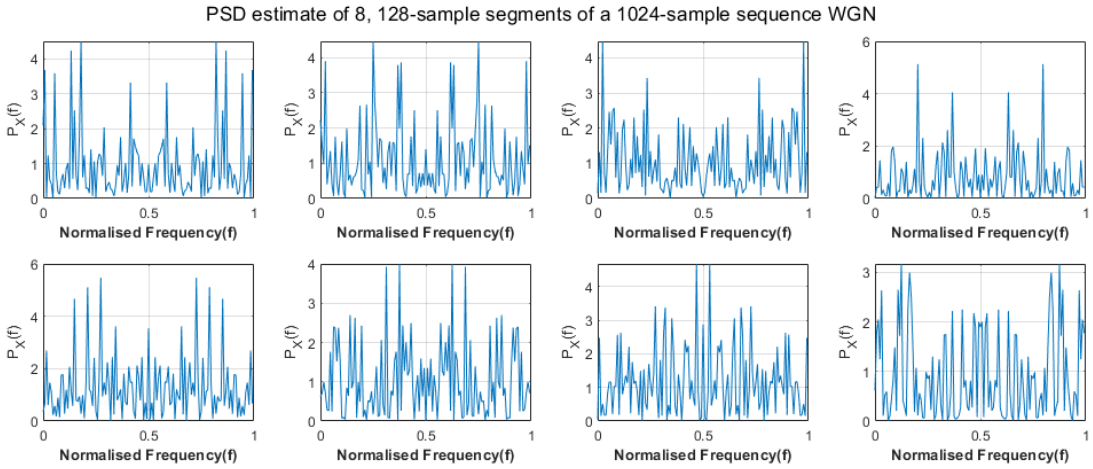


Figure 32: Periodogram of 8, 128-sample segments of a 1024-sample realisation of WGN

The PSD at each frequency point of the 8 realisations of WGN are averaged, producing the result shown in Figure 33. This method improves the estimation of the PSD of WGN, as did the method used in section 3.1.1. This method however is now suitable to be used for estimating non-constant PSDs. The estimation of the PSD at a certain frequency point is no longer dependent on the previous values. The downside of using the method is that now to produce an N point estimation of PSD, $8N$ points from the process are required, meaning that it is a far more data intensive method. Furthermore, this method assumes the ergodicity of the process, if the process is non-ergodic, then this method is not suitable in estimation the PSD.

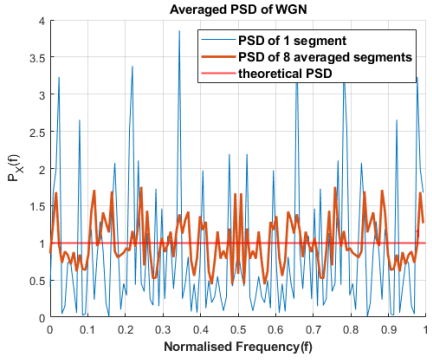


Figure 33: Periodogram of 8, 128-sample segments of a 1024-sample realisation of WGN

3.2 Spectrum of Autoregressive Processes

3.2.1 PSD of an AR(1) process: Questions 1, 2 & 3

The theoretical PSD of an AR(p) process has the form:

$$P_Y(f) = \frac{\sigma_X^2}{|1 + \sum_{k=1}^{p} a_k e^{-j2\pi f}|^2} \tag{58}$$

Where σ_X^2 is the variance of the input to the AR filter. We now consider an AR(1) model, where the AR coefficients are given by $\mathbf{a} = [1, 0.9]$, and the input to the filter is WGN, such that $\sigma_X^2 = 1$ and thus the theoretical PSD of the output is given as:

$$P_Y(f) = \frac{1}{|1 + 0.9e^{-j2\pi f}|^2} \tag{59}$$

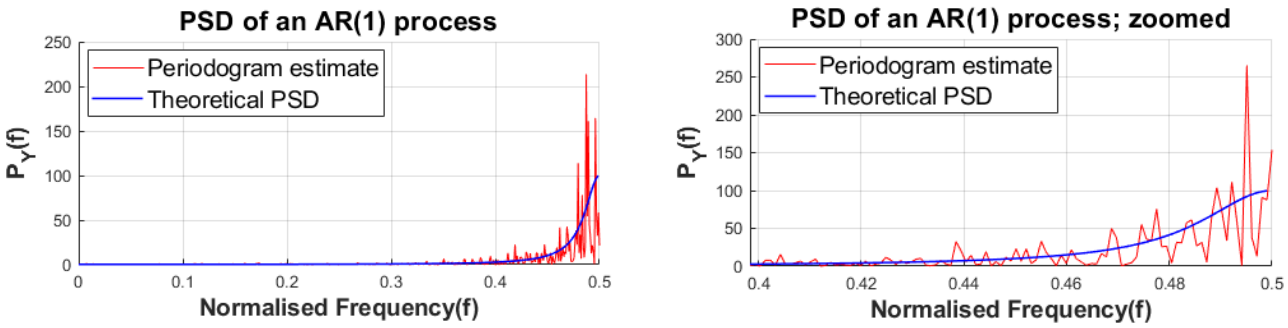


Figure 35: PSD estimate of an AR(1) process

To examine the practical nature of this process, a 1064-sample realisation of WGN is used as the input signal, and is filtered using the same filter as just defined to produce the output signal $\mathbf{y}$, the first 40 values of the resulting signal are removed to ignore the transient effects of the filter. The theoretical PSD, and the periodogram (using `pgm`) of $\mathbf{y}$ are shown in Figure 35. The pole-zero plot of this AR(1) process is shown in Figure 34. Using a geometric approach to find the point on the unit circle which is equidistant to the pole and zero, it can be seen that this point corresponds to 2.037 radians in normalised angular frequency and 0.324 in the normalised frequency domain. After this point, the pole begins dominating the frequency response and thus the PSD, with a maximum at π radians or 0.5 in normalised frequency; this is the point where the unit circle is geometrically closest to the pole and furthest from the zero. This response is exhibited in Figure 35.

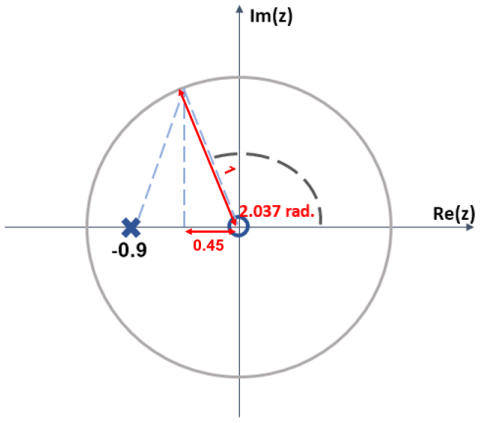


Figure 34: Pole-zero plot of an AR(1) process

When finding the periodogram of a finite length signal, this is seen in the frequency domain, as an infinite length signal windowed by the rectangular function. This corresponds to the frequency response (PSD also) being convolved with a sinc function. At low frequencies, the PSD is close to 0 and thus the apparent effects of this convolution are very little, as we approach higher frequencies, the PSD begins increasing in value and so the effect of the convolution gets 'enabled', its effects are more clearly seen.

3.2.2 Model based PSD estimation: Question 4

If the sequence is known, or is concluded from tests, to be an AR(p) model, then a model based estimation can be used for a more accurate estimation of the PSD. Looking at equation 58, it can be seen that there are $p+1$ unknown parameters, $\hat{\sigma}_X^2, \hat{a}_1, \hat{a}_2, \ldots\ \hat{a}_p$. These parameters can be estimated using the Yule-Walker equations(equation 38), and substituted into equation 58 to obtain the estimate $\hat{P}_Y(f)$. In the case of an AR(1) process there are two parameters that need to be estimated, which can be done using the ACF of the output signal, see eqauation 60, 61. The results of this are shown in Figure 36, it is clear that the model based approach yields far better results. Despite this, it is important to note that this method is only suitable when the type and order of the model is known or inferred.

$$\hat{a_1} = -\frac{\hat{R}_Y(1)}{\hat{R}_Y(0)} \qquad (60) \qquad\qquad \hat{\sigma}_X^2 = \hat{R}_Y(0) + \hat{a_1}\hat{R}_Y(1) \qquad (61)$$

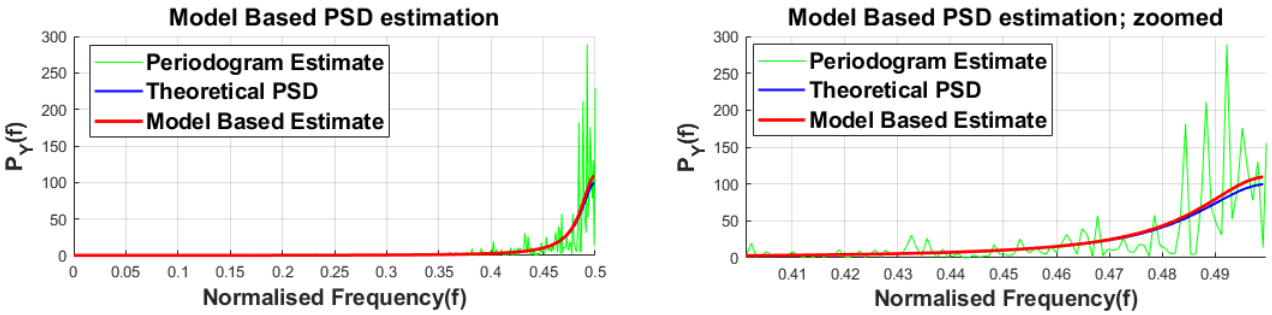


Figure 36: Model Based PSD estimation of an AR(1) process

3.2.3 PSD estimation of Sunspot time-series: Question 5

The above model based estimate is tested for un-normalised and normalised versions of the Sunspot time series dataset, with the results being shown in Figure 37. Since the sunspot series data has a large inherent DC offset, much of the power is concentrated at DC (see Figure 37). This makes it difficult to identify the higher order components of the PSD, it is thus more suitable to look at the PSD estimation of the normalised data.

AR(1) estimation correctly captures some spectral peaking near DC, however it completely bypasses the more significant peaking seen at around 0.1(normalised frequency), thus it is considered as unsuitable. The AR(10) model captures peaking at both these points, whereas AR(2) only captures the more significant peaking at 0.1(normalised frequency). This would naively suggest that the AR(10) model would be more suitable to model the sunspot series, however as discussed in 2.3.4, this would risk over-modelling and would come at a greater computational cost, these risks do not justify the slightly more accurate modelling at lower frequencies. In Figure 38, the risks of over-modelling are further exemplified, if the model order is too high, in this case AR(70), **spectral splitting** is observed, this is where the model incorrectly captures a 'split' of a single peak into two separate peaks.

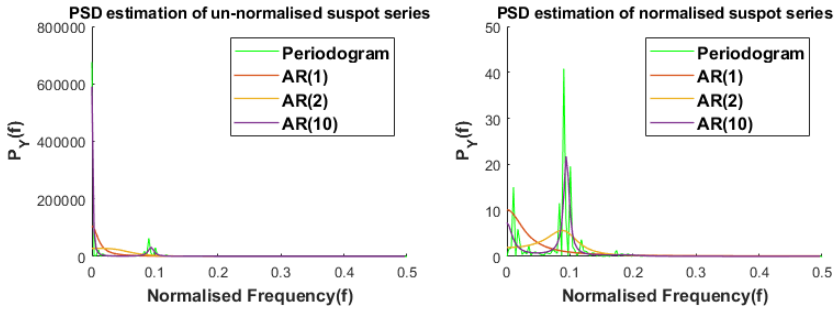


Figure 37: PSD estimation of sunspot series

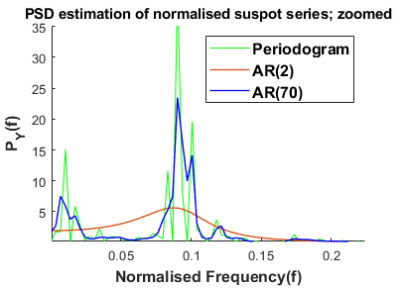


Figure 38: PSD estimation of normalised sunspot series; zoomed

3.3 The Least Squares Estimation(LSE) of AR coefficients

3.3.1 Derivation of the LSE of AR coefficients:L Questions 1 & 2

The LSE of AR coefficients is obtained via a minor modification to the **expanded** Yule Walker equations (using all ACF coefficients instead of p coefficients as in equation 38). We begin by examining the matrix form of the Yule-Walker equations, which has the same form as equation 38. The solution of this equation assumes that $\mathbf{r}_{xx}$ and $\mathbf{R}_{xx}$ are deterministic, where in fact we are only able to obtain estimations of these parameters. We adjust equation 38 to introduce an error term $\mathbf{e}$ which captures the stochastic error of the

parameter estimation.

$$\hat{\mathbf{r}}_{xx} = \hat{\mathbf{R}}_{xx}\mathbf{a} + \mathbf{e} \quad \implies \quad \mathbf{e} = \hat{\mathbf{r}}_{xx} - \hat{\mathbf{R}}_{xx}\mathbf{a} \tag{62}$$

We attempt to find the LSE of $\mathbf{a}$, which means we attempt to minimise the square of the magnitude of the error such that the cost function, J is:

$$J = |\mathbf{e}|^2 = |\hat{\mathbf{r}}_{xx} - \hat{\mathbf{R}}_{xx}\mathbf{a}|^2 = (\hat{\mathbf{r}}_{xx} - \hat{\mathbf{R}}_{xx}\mathbf{a})^T(\hat{\mathbf{r}}_{xx} - \hat{\mathbf{R}}_{xx}\mathbf{a}) \quad = \quad \sum_{k=1}^{M}\left[\hat{r}_{xx}[k] - \sum_{i=1}^{p} a_i r_{xx}[k-i]\right]^2 \tag{63}$$

The general form of the cost function of an LSE is given by:

$$J = (\mathbf{x} - \mathbf{H}\boldsymbol{\theta})^T(\mathbf{x} - \mathbf{H}\boldsymbol{\theta})$$

Where the assumed signal $\mathbf{s} = \mathbf{H}\boldsymbol{\theta}$. By comparison with equation 63, it can easily be seen that in fact:

$$\mathbf{x} = \hat{\mathbf{r}}_{xx} = \begin{bmatrix} \hat{r}_{xx}[1] \\ \hat{r}_{xx}[2] \\ \vdots \\ \hat{r}_{xx}[M] \end{bmatrix} \quad \mathbf{H} = \hat{\mathbf{R}}_{xx} = \begin{bmatrix} \hat{r}_{xx}[0] & \hat{r}_{xx}[-1] & \dots & \hat{r}_{xx}[1-p] \\ \hat{r}_{xx}[1] & \hat{r}_{xx}[0] & \dots & \hat{r}_{xx}[2-p] \\ \vdots & \vdots & \ddots & \vdots \\ \hat{r}_{xx}[M-1] & \hat{r}_{xx}[M-2] & \dots & \hat{r}_{xx}[M-p] \end{bmatrix} \quad \boldsymbol{\theta} = \mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix}$$

In the LSE of a set of parameters, $\boldsymbol{\theta}$, it is assumed that the values of the matrix $\mathbf{H}$ are known and thus are entirely deterministic. However, in the case of this problem, $\mathbf{H} = \hat{\mathbf{R}}_{xx}$, which is composed of estimations, and thus is stochastic, this means that the solution for $\mathbf{a}$ will not be the **exact** least-square solution. In the general sense, to find the set of parameters, $\boldsymbol{\theta}$ that minimise the cost function we solve as follows:

$$\frac{\partial J}{\partial \boldsymbol{\theta}} = \frac{\partial(\mathbf{x} - \mathbf{H}\boldsymbol{\theta})^T(\mathbf{x} - \mathbf{H}\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{\partial(\mathbf{x}\mathbf{x}^T - 2\mathbf{x}\mathbf{H}\boldsymbol{\theta} + \boldsymbol{\theta}^T\mathbf{H}^T\mathbf{H}\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = -2\mathbf{x}\mathbf{H} + 2\mathbf{H}^T\mathbf{H}\boldsymbol{\theta} = 0$$

We are estimating $\boldsymbol{\theta}$, thus its estimate is given by $\hat{\boldsymbol{\theta}}$, and so solving the equation just given, we have:

$$\implies \hat{\boldsymbol{\theta}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \tag{64}$$

3.3.2 AR model order selection using LSE

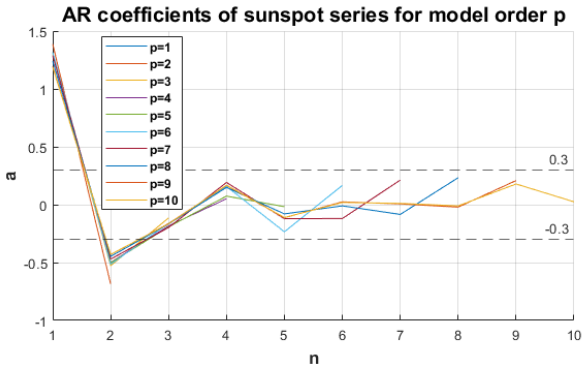


Figure 39: AR coefficients for model order p: Question 3 & 4

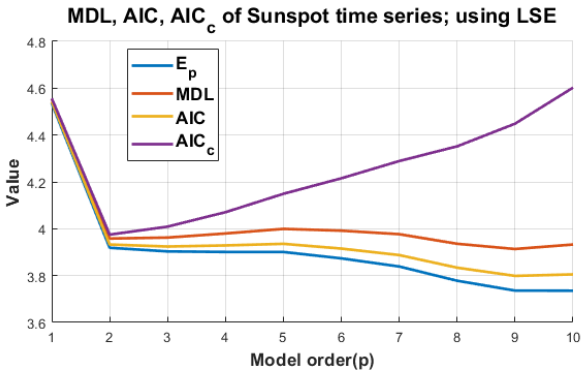


Figure 40: Criterion for model order selection

In Figure 39 we plot the AR coefficients of the sunspot time series using the LSE method along with the MDL, AIC, AIC$_c$(Figure 40) for model orders $p = 1, 2, ...10$. The plots of the different criterion are almost identical to that seen in Section 2.3.4, meaning the same inferences and thus conclusion still apply, a model

order of 2 is most suitable to represent the sunspot time series. Results are almost identical because both the Yule-Walker and LSE are inherently similar in their methods of obtaining estimates, they simply differ by the assumptions that they make to obtain estimates.

3.3.3 PSD of AR(2) model of the sunspot time series, LSE: Question 5

Just as done in section 3.2.3, the PSD of the normalised Sunspot sun series is estimated, but instead, it is done using the LSE of the AR coefficients rather than using the Yule-Walker estimations, results shown in Figure 41. The model-based estimation of the PSD is entirely dependent on the AR coefficients, since the estimated AR coefficients are almost identical for both methods, the PSD estimation is unsurprisingly also almost identical to that seen in Figure 37.

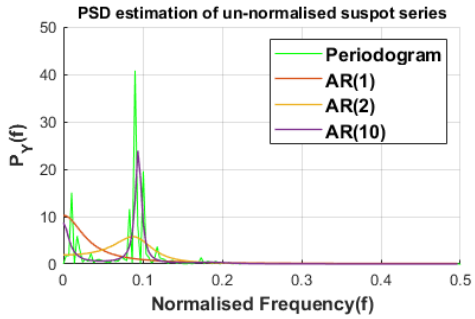


Figure 41: Pole-zero plot of an AR(1) process

3.3.4 Approximation error of AR(2) estimation of Sunspot series for changing data lengths: Question 6

Figure 42 shows the mean square error(MSE) of the AR(2) estimate of the sunspot time series for signal lengths $N \in [10 : 5 : 250]$. For data lengths less than $\simeq 25$, the model exhibits a high Mean Squared Error (MSE), after this point, performance appears to plateau. However, in fact, performance is expected to asymptotically approach the upper limit of the performance (lower limit of MSE) that can be achieved with an AR(2) model when modelling the sunspot time series.

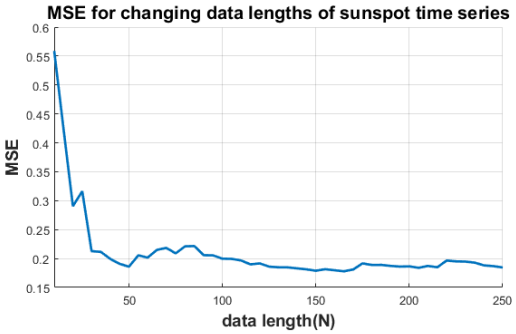


Figure 42: MSE estimating sunspot

3.4 Spectogram for time-frequency analysis: dial tone pad

3.4.1 Generating random London landline number: Question 1

Time-frequency analysis allows us to study the time evolution of the spectral contents of signals. In this section, time-frequency analysis is explored in the application of a Dual Tone Multi-Frequency(DTMF) system, which is the underlying concept of touch-tone telephone dialing. It assigns a signal composed of two sinusoids(Equation 65) with frequencies drawn from Table 5 for each button:

$$y[n] = sin(2\pi f_1 n) + sin(2\pi f_1 n) \quad (65)$$

$f_1 \backslash f_2$	1209 Hz	1336 Hz	1477 Hz
697 Hz	1	2	3
770 Hz	4	5	6
852 Hz	7	8	9
951 Hz	*	0	#

Table 5: Dial pad frequencies

A London landline number is generated, i.e. a sequence of the form: `020 XXXX XXXX`, where the last eight digits(0-9) are drawn randomly. A sampling frequency, $F_s = 32768$Hz is used. It is assumed that

the key is pressed for 0.25s, followed by an idle time of 0.25s. The following waveform is constructed using these assumptions for a London landline number, shown in Figure 43:

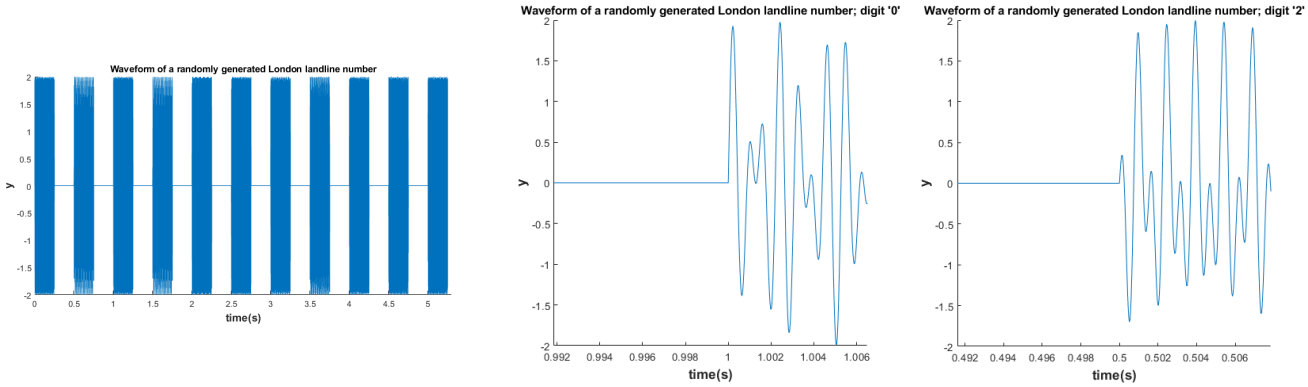


Figure 43: Waveform of a London landline number generated randomly using equation 65

The sampling rate is 32768 Hz, the maximum frequency that any waveform can have is $f_{max} = 1477$ Hz. According to Nyquist theorem, the sampling frequency, F_s, must satisfy the following inequality in order to avoid aliasing, $F_s \geq 2f_{max}$. The minimum sampling frequency to avoid aliasing is thus 2954 Hz. The sampling frequency chosen is over 11 times this value, this is because the greater the bandwidth of the sampled signal compared to the bandwidth of the original signal, the greater the signal-to-noise (SNR). This means that the signal can better handle noise compared to if it were to be sampled at Nyquist frequency.

3.4.2 Spectrogram of random London landline number: Questions 2 & 3

The spectrogram of a signal is constructed by utilising the Short-time Fourier Transform(STFT), which applies windowing to the input signal(isolate the signal between two time points) and then performs the Fast Fourier Transform(FFT). To best capture the time-frequency contents of the signal, the window size of the FFT must correspond to exactly 0.25 seconds worth of samples, such that each FFT window corresponds to the press of exactly one button, or to exactly one period of idle time. The Hann window is used to minimise spectral leakage effects, the spectrogram is shown in Figure 44. A threshold of -80dB is used to better highlight the main frequency components.

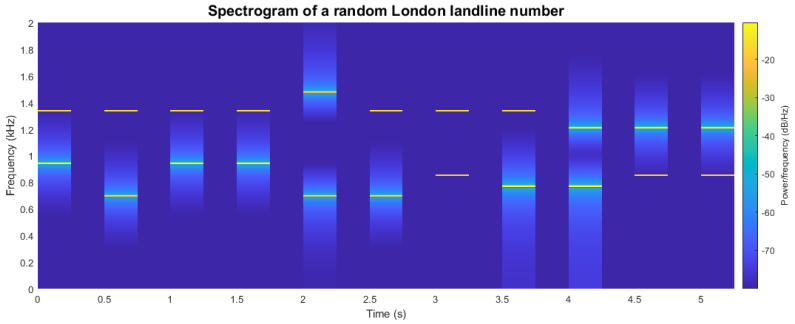


Figure 44: Spectrogram of London landline number

To identify a sequence of key presses, one can raise the power threshold on the sprectrogram such that for each 0.25s window, only the 2 main frequency components of the signal are visible. The values of these frequency components can be read off the graph, giving f_1 and f_2 from equation 65, and then the value of the button can be deduced from Table 5.

3.4.3 Spectrogram of random London landline number with noise

In real-world applications, signals become corrupted with white noise in the channel, in this section we inspect the spectrogram of the signal which is corrupted with varying levels of noise power (Figure 45). Note equation 65 now becomes:

$$y_n[n] = sin(2\pi f_1 n) + sin(2\pi f_1 n) + w[n], \qquad w[n] \sim \mathcal{N}(0, \sigma_N^2) \tag{66}$$

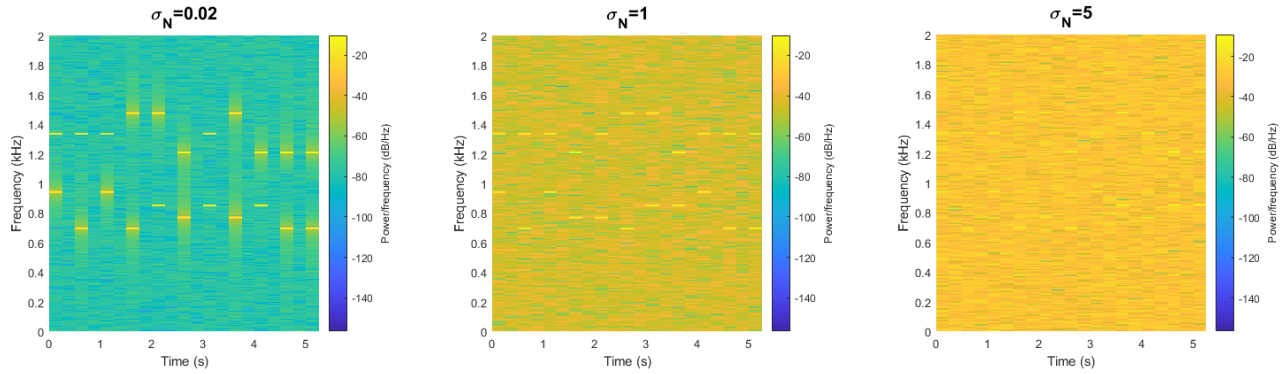


Figure 45: Spectrogram of London landline number with noise: Question 4

As expected, noise power is distributed evenly across the frequency spectrum(in the mean sense), as noise power approaches the signal power, f_1 and f_2 become less and less distinguishable from the WGN frequency components. For $\sigma_N = 0.02$, noise power is far smaller than the signal power, and thus f_1 and f_2 are still clearly distinguishable. For $\sigma_N = 1$, noise power is still smaller than signal power thus f_1 and f_2 are just about still distinguishable and a suitable threshold will be able to isolate them, however for $\sigma_N = 5$, noise power is far greater than signal power and thus the frequency content of the original signal is no longer recoverable.

3.5 Respiratory sinus arrhythmia from RR-Intervals

We revisit the RRI data in Section 2.5 and attempt to use their power spectra to deduce their breathing rates. RRI1 was recorded such that breathing is at approximately 15 times per minute (0.25Hz), RRI2 at roughly 25 times per minute(0.42Hz), and RRI3 at 7.5 times per minute (0.125Hz). The estimates of their respective PSDs are shown in Figure 46, calculated using the periodogram and the averaged periodogram of different window sizes.

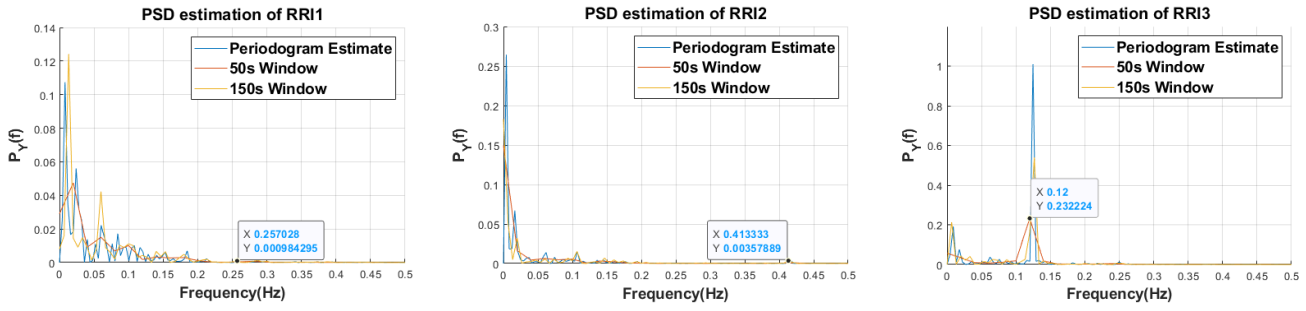


Figure 46: PSD estimate of RRI data

RRI3 shows a significant peak at the frequency corresponding to the breathing rate, whereas RRI1 and RRI2 exhibit very minor peaks at their respective breathing frequencies. If the PSD is observed with no prior context, it can be difficult to deduce the breathing rate in these cases, however, if a rough idea of

the breathing patterns of the patient are known, it may be sufficient to deduce the breathing rate from the PSD.

4 Optimal Filtering - fixed and adaptive

4.1 Wiener Filter

The optimum Wiener filter is used to obtain the closest estimate(in the square sense) of an unknown system's parameters. In order to obtain a suitable estimation, the order and nature of the system must be known, and this will be assumed when investigating the performance of the Wiener Filter. In this section we assume that the unkown system we are identifying is a Moving-Average of order 4. The optimum Wiener filter has a closed form solution. The coefficients of the unknown system, given by $\mathbf{w}$, are estimated by $\mathbf{w}_{opt}$, which is found via Equation 67, according to Equation 68:

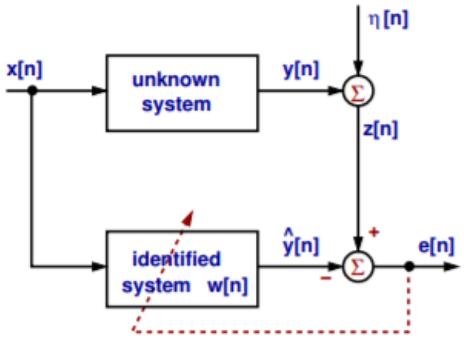


Figure 47: System Identification Diagram

$$\mathbf{w}_{opt} = \mathbf{R}_{xx}^{-1}\mathbf{p}_{zx} \quad (67) \qquad \mathbf{p}_{zx} = \begin{bmatrix} r_{zx}[0] \\ r_{zx}[-1] \\ \vdots \\ r_{zx}[-N_w] \end{bmatrix} \quad \mathbf{R}_{zx} = \begin{bmatrix} r_{zx}[0] & r_{zx}[-1] & \dots & r_{zx}[-N_w] \\ r_{zx}[1] & r_{zx}[0] & \dots & r_{zx}[-N_w+1] \\ \vdots & \vdots & \ddots & \vdots \\ r_{zx}[N_w] & r_{zx}[N_w-1] & \dots & r_{zx}[0] \end{bmatrix} \quad (68)$$

Where r_{zx} is the cross correlation between the input, x and the output, z, N_w is the order of the system. This method is tested with a 1000-sample realisation of WGN of power $\sigma_x^2 = 1$, and is input into the system which is described by an MA filter with coefficients, $\mathbf{b} = [1, 2, 3, 2, 1]$, whose output is given by $y[n]$ which is then normalised such that $\sigma_y^2 = 1$. This signal is corrupted by WGN, $\eta[n]$ with power $\sigma_\eta^2 = 0.01$, which yields the output, $z[n] = y[n] + \eta[n]$. The Signal-to-Noise Ratio(SNR) of the output is calculated in Equation 69.

$$SNR_z = \frac{P_y}{P_\eta} = \frac{\sigma_y^2}{\sigma_\eta^2} = 100 = 20dB \quad (69)$$

The above process is carried out experimentally, yielding a SNR of 20.333, which confirms the theoretical result, the discrepancy of course due to the fact that it is based off of finite-length data.

4.1.1 MA filter coefficient estimation: Question 1

The method just described is used to estimate the coefficients of the MA also just described, yielding the following estimates of coefficients, we note that we do not normalise y[n] as was done to calculate SNR as this leads to incorrect estimates:

$$\hat{\mathbf{b}} = [0.9105, 1.9599, 2.9939, 1.9862, 0.9916]$$

When the signal-to-noise ratio is large, the Wiener Filter produces estimated very close to its real values. Now the calculated statistics $\mathbf{p}_{zx}$ and $\mathbf{R}_{zx}$ that are used to calculate the estimate are given by:

$$\mathbf{p}_{zx} = \begin{bmatrix} 740.04 \\ 1607.9 \\ 2553.6 \\ 1610.6 \\ 737.12 \end{bmatrix}, \quad \mathbf{R}_{zx} = \begin{bmatrix} 908.7 & 20.20 & 42.50 & 39.92 & -81.20 \\ -20.20 & 908.7 & 20.20 & 42.50 & 39.92 \\ -42.50 & -20.20 & 908.7 & 20.20 & 42.50 \\ -39.92 & -42.50 & -20.20 & 908.7 & 20.20 \\ 81.20 & -39.92 & -42.50 & -20.20 & 908.7 \end{bmatrix} \quad (70)$$

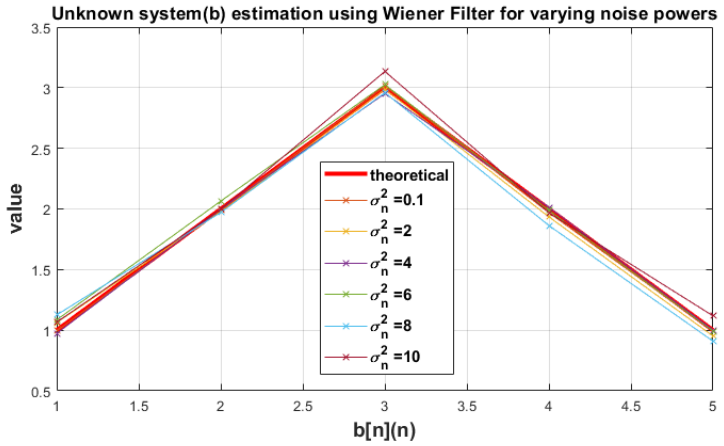
4.1.2 Varying noise power and assumed model order: Question 2

We repeat the same experiment but vary the noise power such that $\sigma_\eta^2 \in [0.1, 10]$. The estimations of the coefficients for these cases are shown in Figure 48, and the SNRs are shown in Table 6. As expected, SNR falls with increasing noise power. The model seems to be quite robust in its estimations, it performs well even when SNR falls below 0 dB. However, as expected, it can be seen that the accuracy of these estimations do fall with SNR.

The Experiment in 4.1.1 is repeated for a different assumed model order, $N_w = 8$, the prediction of the coefficient becomes:

$$\hat{\mathbf{b}} = [1.7158, 2.4317, 3.1201, 1.9440, 0.9553, 0.0148, 0.0462, 0.0462, 0.0152]$$

For coefficients past the order of the corresponding the system, estimations fall very close to 0, whilst the estimations of the system coefficients still remain quite accurate. This means that not only does the system order not **need** to be known, but it could potentially be inferred by selecting an arbitrarily large assumed model order and seeing where coefficients fall to 0, of course at a far greater computational cost.



Noise power σ_η^2	$SNR_z(dB)$
0.1	9.90
2	-3.17
4	-5.95
6	-8.03
8	-9.31
10	-10.2

Table 6: SNRs

Figure 48: Coefficient estimation of a MA(5) system using Wiener Filter

4.1.3 Complexity of optimum Wiener Filter: Question 3

We can break down the calculation involved with finding the optimum Wiener Solution to find its complexity:

1. Finding $\mathbf{R}_{xx}$ and $\mathbf{p}_{zx}$: Each requires $N \cdot (N_w + 1)$ calculations $\implies \mathcal{O}(N \cdot N_w)$.
2. Finding $\mathbf{R}_{xx}^{-1}$: Inverse of matrix of dimension: $(N_w + 1) \times (N_w + 1) \implies \mathcal{O}((N_w + 1)^3) \implies \mathcal{O}(N_w^3)$.
3. $\mathbf{R}_{xx}^{-1} \cdot \mathbf{p}_{zx}$: Equivalent to calculating $N_w + 1$ inner products, each inner product requires $N_w + 1$ multiplications and N_w additions, corresponding to $(Nw + 1) \cdot (2N_w + 1)$ calculations $\implies \mathcal{O}(N_w^2)$.
4. Combining these, the total complexity becomes: $\mathcal{O}(N \cdot N_w) + \mathcal{O}(N_w^3) + \mathcal{O}(N_w^2)$. If $N > N_w^2$, the complexity is $\mathcal{O}(N \cdot N_w)$, otherwise the complexity is $\mathcal{O}(N_w^3)$.

4.2 The Least Mean Square(LMS) algorithm

As just seen, the Wiener filter has its drawbacks, it can be a very computationally expensive calculation (not suitable for real-time) and it also hinges on the assumption that the process is stationary. Adaptive

filters are instead used to study these signals, the simplest being the LMS and is given by:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu e[n]\mathbf{x}(n), \quad n = 0, 1, ... \tag{71}$$

$$\hat{y}[n] = \mathbf{w}^T(n)\mathbf{x}(n) = \sum_{m=0}^{N_w} w_m(n)x(n-m) \tag{72}$$

$$e[n] = z[n] - \hat{y}[n] \tag{73}$$

Where $e[n], z[n], \hat{y}[n], x[n], \mathbf{w}(n)$ are as per Figure 47, and μ is the adaptation gain used to control the stability of the algorithm.

The same experiment as in 4.1.2 is repeated for the LMS algorithm with an adaptation gain $\mu = 0.01$. Coefficient estimates, $\hat{\mathbf{b}}$, are taken as the values according to the final iteration of the algorithm, the results being shown in Figure 49. It can be seen with the Wiener filter, estimates become less accurate as SNR decreases (noise power increases).

It can also be seen that for each SNR, the Wiener Filter is giving estimations closer to the true values compared to the LMS. This is largely due to the iterative nature of the LMS algorithm, the adaptation gain is relatively large and thus corrections to the estimates at each iteration tend to overshoot, leading to this variability over time.

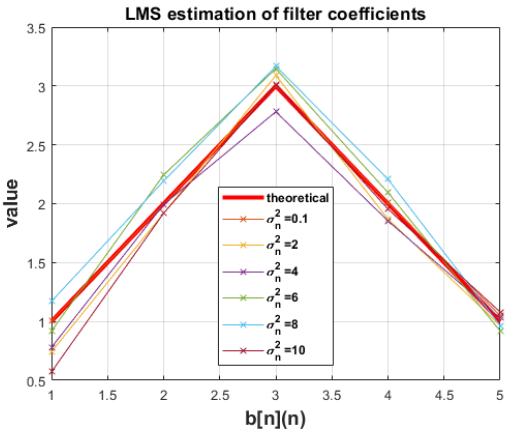


Figure 49: LMS estimate of $\mathbf{b}$

4.2.1 Adjusting adaptation gain: Question 2

The same experiment in Question 1 is repeated but using the LMS algorithm, and is tested for different values of adaptation gain, $\mu = \{0.01, 0.1, 0.5\}$. The evolution of coefficient estimates along with the squared error at each iteration and for each adaptation gain are shown in Figure 50:

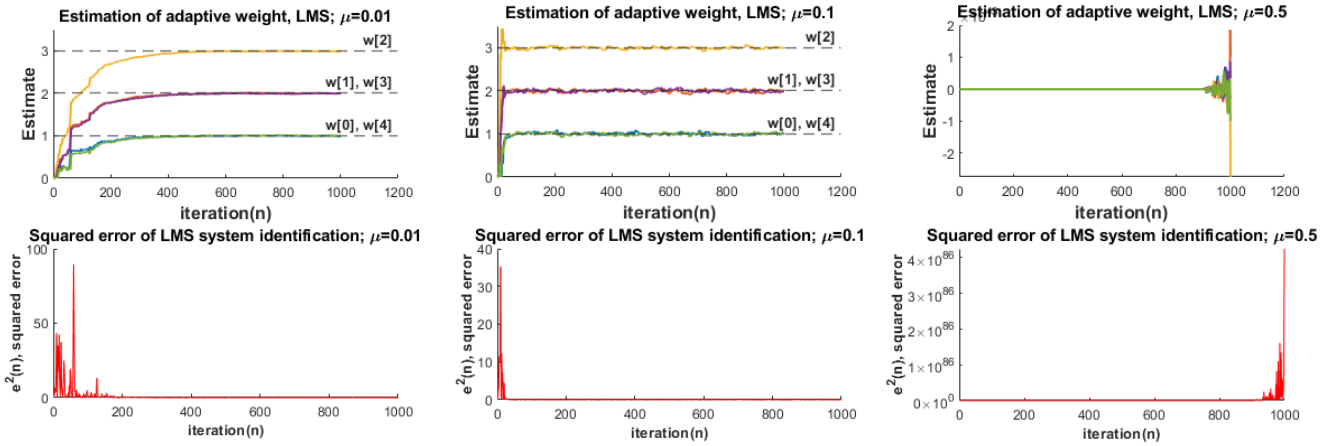


Figure 50: Evolution of coefficient estimation and squared error of the LMS algorithm

By inspecting equation 71, it can be seen that as μ is increased, the 'step' or correction at each time point is also increased, thus we expect a shorter rise time as μ is increased but also more variability in the estimate at each iteration. This is indeed observed for $\mu = 0.01$ and $\mu = 0.1$, however as μ is increased past a certain value, for example $\mu = 0.5$, this leads to divergent behaviour as the step size is too large

and we bypass the optimal solution. More formally, for the LMS to converge in the mean square sense, the adaptation gain is bounded as follows:

$$0 < \mu < \frac{2}{tr[\mathbf{R}_{xx}]} = \frac{2}{p\sigma_x^2} \tag{74}$$

Where $tr[\mathbf{R}_{xx}]$ is the trace of the autocorrelation matrix and $p\sigma_x^2$ is the total input power. Clearly, $\mu = 0.5$ does not satisfy this inequality, and thus has divergent behaviour.

4.2.2 Complexity of the LMS: Question 3

As done with the wiener filter, we break down the calculations to obtain the LMS to analyse its complexity, to do this we examine equations 71, 73, and 72.

1. Calculate $\hat{y[n]}$: Done by performing the inner product of $\mathbf{w}(n)$ and $\mathbf{x}(n)$, each with N_w+1 values. This corresponds to N_w+1 multiplications and N_w additions, which is $2N_w+1$ calculations, $\implies \mathcal{O}(N_w)$.

2. Calculating $e[n]$: $\mathcal{O}(1)$.

3. Updating $\mathbf{w}(n+1)$: (N_w+1) additions, $\implies \mathcal{O}(N_w)$ At each iteration the complexity of the algorithm is thus $\mathcal{O}(N_w) + \mathcal{O}(N_w) + \mathcal{O}(1) = \mathcal{O}(N_w)$. The total complexity however after N iterations is thus $\mathcal{O}(N \cdot N_w)$

4.3 Gear Shifting

For statistically stationary processes, Gear Shifting can be used to conditionally change the adaptation gain in time such that a balance is found in the rise time of the estimate and the accuracy after N iterations. From **??** it could be seen that for $\mu = 0.1$, a very short rise time is seen, and the estimates converge very quickly, it does however exhibit some overshooting so we choose, $\mu[0] = 0.08$ as the base gain for the model. The fractional error, $e_f = \left|\frac{e[n]}{z[n]}\right|$, was used as the main condition to change μ, It is a more consistent indicator of the current state of estimations compared to $e[n]$ as it removes sign and the magnitude dependency of the metric $(e[n])$. After comparing different policies for μ selection, the following yielded the best results:

$$\mu[n] = \begin{cases} \mu[0] \cdot \frac{10}{n}, & \text{if } e_f < 0.1 \textbf{ and } n > 150 \\ \mu[n-1], & \text{otherwise} \end{cases} \tag{75}$$

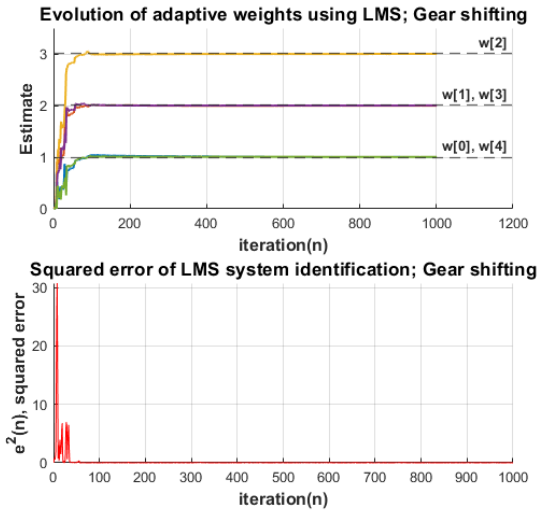


Figure 51: LMS estimate of **b**

The condition for $n > 150$ was chosen to allow the estimation to settle before gear shifting to minimise the risk of any divergent behaviour. The results of using this policy are shown in Figure 4.3, we can see it exhibits a fast rise time and very stable and accurate estimations once converged.

4.4 Identification of AR processes

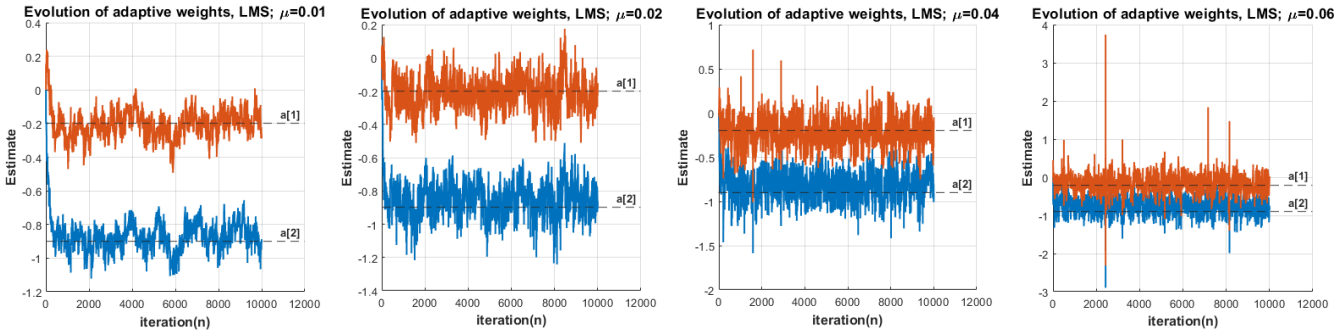


Figure 52: Evolution of coefficient estimation and squared error of the LMS algorithm

We synthesise an AR process by filtering a 10000-sample realisation of WGN, with an AR model with parameters $\mathbf{a} = [1, 0.9, 0.2]$. The output of the filter is given by:

$$X(z) = \frac{\overbrace{W(z)}^{1}}{1 + 0.9z^{-1} + 0.2z^{-2}} \implies x[n] = -0.9x[n-1] - 0.2x[n-2] = w_1 x[n-1] + w_2 x[n-2]$$

We attempt to estimate w_1 and w_2, hence why estimates converge to these values (Figure 52).

Figure 52 shows the estimations of the AR model parameters using the LMS algorithm for varying adaptation gains. As μ increases convergence gets quicker, but with far greater variance. None of the μ values appear to exceed the bounds in equation 74 (no divergent behaviour.).

4.5 Speech Recognition

In this section we aim to model speech signals of 1000 samples, sampled at 44.1 kHz and 16 kHz, with an AR model of order p such that the error between the predictions of the signal and the true signal is minimised. Such predictor will be tested on recordings of the letters , "e", "a", "s", "t", "x" being spoken.

4.5.1 Model order, adaptation gain, and Algorithm selection: Questions 1 & 2 (44.1kHz)

There are 2 main approaches to selecting the most suitable model order of an AR process:

1. **Analytical approach:** Use model order selection metrics and plots such as the PACF, MDL, AIC and AIC_c.

2. **Heuristic approach:** Choose an arbitrarily large model order (as mentioned in 4.1.2) and selecting the model order p as the final value where the coefficient estimate is sufficiently above 0.

Letter	**PACF**(order)	**MDL**(order)	**AIC**(order)	**AIC**$_c$(order)	**Selected Order**	Selected μ
e	4	5	11	11	5	0.08
a	6	6	9	9	6	0.05
s	3	2	2	2	2	0.08
t	5	7	7	7	7	0.05
x	2	6	6	6	6	0.08

Table 7: Table showing the suggested model order based on performance, and model indication metrics, along with optimal adaptation gain, μ: note these are not the minimums of each metric, simply what order they indicate from inspection.

We intend for an analytical approach to model order selection, using the PACF, MDL, AIC, AIC$_c$, Table 7 summarises what metric indicates what order, including the final selected model order for each letter. We note that the signals for each of the letters were normalised such that values for model order metrics, model adaptation gain etc. remain consistent.

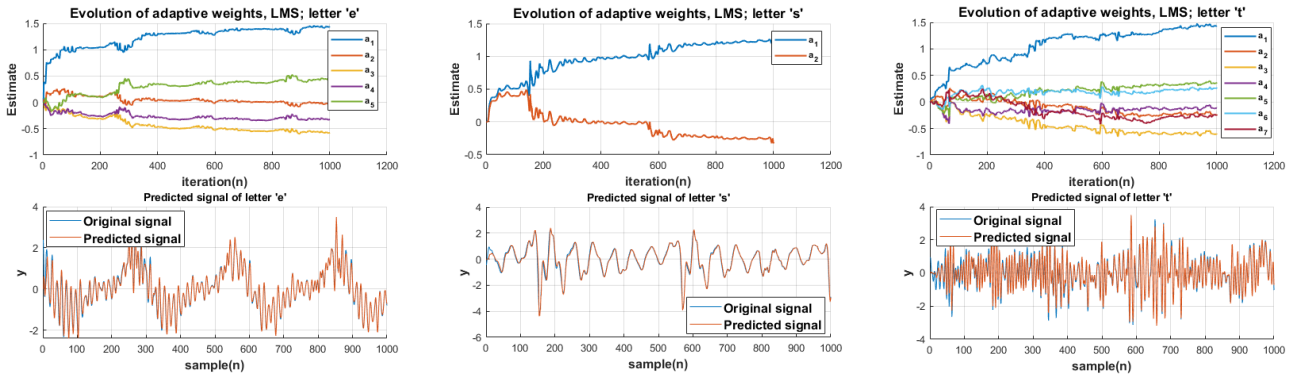


Figure 53: Estimation of speech signals(44.1kHz) using the LMS

Selection of adaptation gain was done experimentally for each letter, the highest gain with stable behaviour was opted to be chosen, this is because speech signals are non-stationary and μ has to be large enough to correctly track the changes in parameter values. The values chosen for each letter are included in table 7. Leading from this, the lack of stationarity in the signal means that gear shifting is not suitable as it hinges on the assumption of stationarity. Figure 53 shows the performance of the selected models on predicting the letters, "e", "s", "t". By looking at the predicted signal, it can be seen that the predictors perform well when signal variance is low ("e" and "s"), but when variance is large ("t"), the discrepancy between predictions and true values are more visually discernible.

4.5.2 Model order, adaptation gain, and Algorithm selection: Question 3 (16kHz)

We repeat the same process as the previous section but instead we sample the voice recordings at 16kHz. The model orders are updated for this new signal, shown in Table 8.

Letter	**PACF**(order)	**MDL**(order)	**AIC**(order)	**AIC**$_c$(order)	**Selected Order**	Selected μ
e	12	12	12	10	12	0.03
a	6	6	12	12	6	0.05
s	17	9	17	17	9	0.04
t	12	12	12	12	12	0.04
x	8	8	8	8	8	0.03

Table 8: Table showing the suggested model order based on performance, and model indication metrics, along with optimal adaptation gain, μ.

By inspecting Table 8, it can be seen that the suitable model order of almost all of the letters has been increased, this is mainly because there is lower density of data points over a certain period of time, and as a result of this, changes in value between each sample will also be greater, requiring a higher order model to appropriately capture this. Now, we can also see that where model order has increased, adaptation gain has had to be decreased to compensate. With reference to equation 74, the upper bound of the adaptation gain for convergence is inversely proportional to the total input power. In the case of adaptive AR model estimation, the input signal is given by equation 36, where the input signal is $x[n]$. Clearly, as model order increases, the input signal is composed of more samples and thus the input power also increases. For this reason, for convergence, the upper bound of μ decreases, hence the need to decrease the value of μ appropriately.

4.5.3 Evaluation of Performance

A standard measure for the peformance of a predictor is the prediction gain and is defined as such:

$$R_p = 10\log_{10}\left(\frac{\sigma_x^2}{\sigma_e^2}\right) \tag{76}$$

Where σ_x^2 is the variance for the input signal and σ_e^2 is the variance of the error signal. This is used to evaluate the performance of each predictor for the speech signals at both 16kHz and 44.1kHz. Results are shown in Table 9:

Letter	R_p (44.1kHz)	Model Order (44.1kHz)	R_p (16kHz)	Model Order (16kHz)
e	15.42dB	5	9.90dB	12
a	15.71dB	6	10.77dB	6
s	15.34dB	2	4.88dB	9
t	12.12dB	7	11.96dB	12
x	18.18dB	6	7.29dB	8

Table 9: Table showing the suggested model order based on performance, and model indication metrics, along with optimal adaptation gain, μ.

Of course the greater the value of R_p, the better the performance of the predictor as this corresponds to a greater ratio of input signal power to error power. We can thus conclude that better performance is seen for a sampling rate of 44.1 kHz since it has a higher prediction gain for every signal. This is because, as aformentioned, the temporal density of samples is greater at a higher sampling rate, thus changes between samples will be smaller in magnitude, which is far easier to capture from the model's standpoint.

4.6 Sign Algorithms

To improve the computational efficiency of estimates, we consider some simplified versions of the LMS algorithms, given by:

$$\text{signed - error:} \quad \mathbf{w}(n+1) = \mathbf{w}(n) + \mu\text{sign}(e[n])\mathbf{x}(n) \tag{77}$$

$$\text{signed - regressor:} \quad \mathbf{w}(n+1) = \mathbf{w}(n) + \mu e[n]\text{sign}(\mathbf{x}(n)) \tag{78}$$

$$\text{sign - sign:} \quad \mathbf{w}(n+1) = \mathbf{w}(n) + \mu\text{sign}(e[n])\text{sign}(\mathbf{x}(n)) \tag{79}$$

The experiment in Section 4.4 is repeated for these algorithms, their performances being compared against the original LMS algorithm in Figure 55:

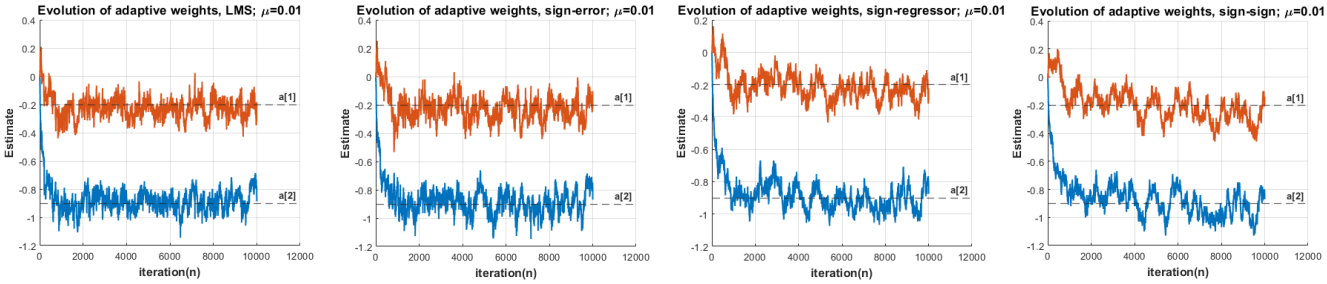


Figure 54: Comparison of LMS variant on the prediction of AR coefficients

All LMS variant algorithms perform well in this task, with the sign-error most resembling the LMS, and the sign-sign showing the most variance in its estimates. This is because the magnitude of the correction term is now entirely dependant on the value of μ and apart from the sign of the correction, it has no reference the current value of error or regressor.

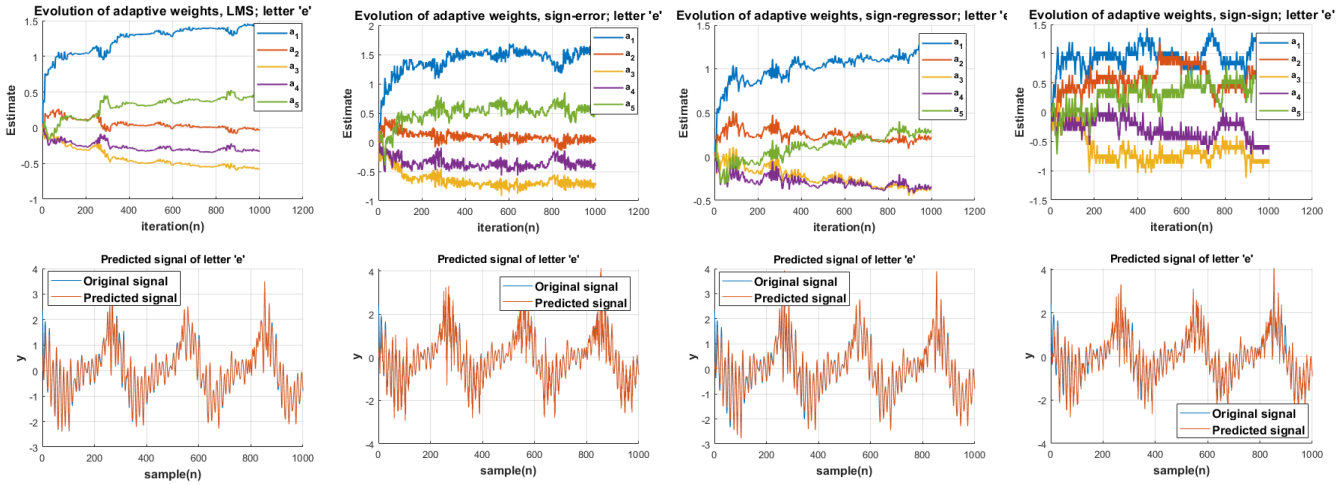


Figure 55: Comparison of LMS variant on the prediction of AR coefficients

We now repeat the experiment in Section 4.5 for the letter "e" using the proposed algorithms (Figure 55). Once again all predictors perform well, sign-error performs closest to the original LMS, and the sign-sign algorithm has the worst peformance out of the three.

5 MLE for the Frequency of a Signal

The pdf of a real signal $\mathbf{x} = [x[0], x[1], ..., x[N-1]]^T$, parametrised by $\boldsymbol{\theta} = [A, f_0, \phi]^T$ can be written as:

$$p(\mathbf{x}; \boldsymbol{\theta}) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left\{ -\frac{1}{2\sigma^2} \sum_{n=0}^{N-1} (x[n] - A\cos(2\pi f_0 n + \phi))^2 \right\} \tag{80}$$

where $A > 0, 0 < f_0 < 1/2$.

5.1 The Cost function

From the above pdf definition, the maximum-likelihood estimate(MLE), $\hat{f}_0$, of the frequency f_0 is found by minimizing the cost function,

$$J(\boldsymbol{\theta}) = \sum_{n=0}^{N-1} (x[n] - A\cos(2\pi f_0 n + \phi))^2 = \sum_{n=0}^{N-1} (x[n] - A\cos(\phi)\cos(2\pi f_0 n) + A\sin(\phi)\sin(2\pi f_0 n))^2 \tag{81}$$

Let $\alpha_1 = A\cos(\phi)$, and $\alpha_2 = A\sin(\phi)$, and then write in Matrix form:

$$J'(\boldsymbol{\theta}) = \sum_{n=0}^{N-1} (x[n] - \alpha_1\cos(2\pi f_0 n) - \alpha_2\sin(2\pi f_0 n))^2 \quad = \quad (\mathbf{x} - \alpha_1\mathbf{c} - \alpha_2\mathbf{s})^T(\mathbf{x} - \alpha_1\mathbf{c} - \alpha_2\mathbf{s}) \tag{82}$$

$$\mathbf{c} = \begin{bmatrix} 1 \\ \cos(2\pi f_0) \\ \cos(4\pi f_0) \\ \vdots \\ \cos(2\pi f_0(N-1)) \end{bmatrix} \quad \mathbf{s} = \begin{bmatrix} 1 \\ \sin(2\pi f_0) \\ \sin(4\pi f_0) \\ \vdots \\ \sin(2\pi f_0(N-1)) \end{bmatrix} \tag{83}$$

Let $\mathbf{H} = [\mathbf{c} \quad \mathbf{s}]$ and $\boldsymbol{\alpha} = [\alpha_1, \ \alpha_2]^T$, we can now rewrite $J'(\boldsymbol{\theta})$ as:

$$J'(\alpha_1, \alpha_2, f_0) = (\mathbf{x} - \alpha_1\mathbf{c} - \alpha_2\mathbf{s})^T(\mathbf{x} - \alpha_1\mathbf{c} - \alpha_2\mathbf{s}) = (\mathbf{x} - \mathbf{H}\boldsymbol{\alpha})^T(\mathbf{x} - \mathbf{H}\boldsymbol{\alpha}) \tag{84}$$

5.2 Minimising the Cost function

We now attempt to minimise the cost function with respect to $\boldsymbol{\alpha}$, this is done by setting the derivative of the cost function with respect to $\boldsymbol{\alpha}$ to 0, as follows:

$$\frac{\partial J'(\boldsymbol{\alpha}, f_0)}{\partial \boldsymbol{\alpha}} = \frac{\partial (\mathbf{x} - \mathbf{H}\boldsymbol{\alpha})^T(\mathbf{x} - \mathbf{H}\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}} = \frac{\partial}{\partial \boldsymbol{\alpha}} \left(\mathbf{x}^T\mathbf{x} - \boldsymbol{\alpha}^T\mathbf{H}^T\mathbf{x} - \mathbf{x}^T\mathbf{H}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\mathbf{H}^T\mathbf{H}\boldsymbol{\alpha} \right) = 0 - \mathbf{H}^T\mathbf{x} - (\mathbf{x}^T\mathbf{H})^T + 2\mathbf{H}^T\mathbf{H}\boldsymbol{\alpha}$$

$$= -2\mathbf{H}^T\mathbf{x} + 2\mathbf{H}^T\mathbf{H}\boldsymbol{\alpha} \implies -2\mathbf{H}^T\mathbf{x} + 2\mathbf{H}^T\mathbf{H}\boldsymbol{\alpha} = 0 \implies \hat{\boldsymbol{\alpha}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \tag{85}$$

If we substitute the value of $\hat{\boldsymbol{\alpha}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x}$ back into equation 84 as $\boldsymbol{\alpha}$, we have:

$$J'(\hat{\alpha_1}, \hat{\alpha_2}, f_0) = (\mathbf{x} - \mathbf{H}\hat{\boldsymbol{\alpha}})^T(\mathbf{x} - \mathbf{H}\hat{\boldsymbol{\alpha}}) = \mathbf{x}^T\mathbf{x} - \hat{\boldsymbol{\alpha}}^T\mathbf{H}^T\mathbf{x} - \mathbf{x}^T\mathbf{H}\hat{\boldsymbol{\alpha}} + \hat{\boldsymbol{\alpha}}^T\mathbf{H}^T\mathbf{H}\hat{\boldsymbol{\alpha}} = -2\hat{\boldsymbol{\alpha}}^T\mathbf{H}^T\mathbf{x} + \hat{\boldsymbol{\alpha}}^T\mathbf{H}^T\mathbf{H}\mathbf{x}$$

$$= \hat{\boldsymbol{\alpha}}^T(-2\mathbf{H}^T\mathbf{x} + \mathbf{H}^T\mathbf{H}\hat{\boldsymbol{\alpha}}) = ((\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x})^T(-2\mathbf{H}^T\mathbf{x} + \mathbf{H}^T\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x}) = \mathbf{x}^T\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}(-2\mathbf{H}^T\mathbf{x} + \mathbf{I}\mathbf{H}^T\mathbf{x})$$

$$\implies J'(\hat{\boldsymbol{\alpha}}, f_0) = -\mathbf{x}^T\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \tag{86}$$

Minimising $J'(\hat{\boldsymbol{\alpha}}, f_0)$ is equivalent to maximising $-J'(\hat{\boldsymbol{\alpha}}, f_0)$, where:

$$-J'(\hat{\boldsymbol{\alpha}}, f_0) = \mathbf{x}^T\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \quad \text{as required.} \tag{87}$$

5.3 Approximating the MLE

Note Equation 87, the cost function that is being maximised, can be rearranged such that:

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

Now substituting $\mathbf{H} = [\mathbf{c} \quad \mathbf{s}]$:

$$= \left(\begin{bmatrix} \mathbf{c}^T \\ \mathbf{s}^T \end{bmatrix} \mathbf{x} \right)^T \left(\begin{bmatrix} \mathbf{c}^T \\ \mathbf{s}^T \end{bmatrix} [\mathbf{c} \quad \mathbf{s}] \right)^{-1} \begin{bmatrix} \mathbf{c}^T \\ \mathbf{s}^T \end{bmatrix} \mathbf{x} = \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix}^T \begin{bmatrix} \mathbf{c}^T\mathbf{c} & \mathbf{c}^T\mathbf{s} \\ \mathbf{s}^T\mathbf{c} & \mathbf{s}^T\mathbf{s} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix} \tag{88}$$

Now it is observed that:

$$\mathbf{c}^T\mathbf{c} = \sum_{n=0}^{N-1} \cos^2(2\pi f_0 n) \approx \frac{N}{2}, \qquad \mathbf{s}^T\mathbf{s} = \sum_{n=0}^{N-1} \sin^2(2\pi f_0 n) \approx \frac{N}{2}$$

$$\mathbf{c}^T\mathbf{s} = \mathbf{s}^T\mathbf{c} = \sum_{n=0}^{N-1} \cos(2\pi f_0 n)\sin(2\pi f_0 n) \approx 0$$

Where $f_0 \neq 0, 1/2$, Equation 88 can thus be approximated as:

$$\begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix}^T \begin{bmatrix} \frac{N}{2} & 0 \\ 0 & \frac{N}{2} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix}, \quad \text{as required.} \tag{89}$$

Now,

$$\begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix}^T \begin{bmatrix} \frac{N}{2} & 0 \\ 0 & \frac{N}{2} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix} = [\mathbf{c}^T\mathbf{x} \quad \mathbf{s}^T\mathbf{x}] \begin{bmatrix} \frac{2}{N} & 0 \\ 0 & \frac{2}{N} \end{bmatrix} \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix} = \frac{2}{N} \left(\mathbf{c}^T\mathbf{x}\mathbf{c}^T\mathbf{x} + \mathbf{s}^T\mathbf{x}\mathbf{s}^T\mathbf{x} \right) = \frac{2}{N} \left((\mathbf{c}^T\mathbf{x})^2 + (\mathbf{s}^T\mathbf{x})^2 \right)$$

$$= \frac{2}{N} \left(\left(\sum_{n=0}^{N-1} x[n]\cos(2\pi f_0 n) \right)^2 + \left(\sum_{n=0}^{N-1} x[n]\sin(2\pi f_0 n) \right)^2 \right) = \frac{2}{N} \left| \sum_{n=0}^{N-1} x[n]\exp(-2j\pi f_0 n) \right|^2 \tag{90}$$

We note that $f_0 = \frac{f}{N}$ and maximising $2f(x)$ is the same as maximising $f(x)$, it thus follows that we are maximising the periodogram:

$$\frac{1}{N}\left|\sum_{n=0}^{N-1} x[n]\exp(-2j\pi f \tfrac{n}{N})\right|^2 \tag{91}$$

5.4 Experimental Results

Now, using the noiseless data

$$x[n] = \cos(2\pi f_0 n), \quad n = 0, 1, ...N-1 \tag{92}$$

The periodogram of this signal for the case of N=10 is shown in Figure 56. Clearly, peaks are exhibited at each of the corresponding fundamental frequencies, which is expected, as this is the only frequency component of the signal. Sharper peaks are observed for $f_0 = 0$ and $f_0 = 0.5$, this can be explained by substituting equation 92 into the periodogram equation (91), as follows:

$$\frac{1}{N}\left|\sum_{n=0}^{N-1}\cos(2\pi f_0 n)\exp(-2j\pi f_0 n)\right|^2 = \frac{1}{N}\left|\sum_{n=0}^{N-1}\cos^2(2\pi f_0 n) - j\sum_{n=0}^{N-1}\overbrace{\cos(2\pi f_0 n)\sin(2\pi f_0 n)}^{0}\right|^2$$

$$= \frac{1}{N}\left(\left(\sum_{n=0}^{N-1}\cos^2(2\pi f_0 n)\right)^2\right)$$

For $f_0 \neq 0, \frac{1}{2}$, $\cos^2(2\pi f_0 n) < 1$, whereas for $f_0 = 0, \frac{1}{2}$, $\cos^2(2\pi f_0 n) = 1$, thus the peaks exhibited at these frequency will clearly be larger in magnitude.

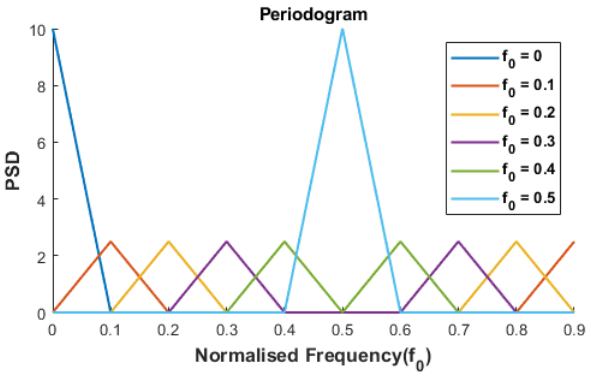


Figure 56: Periodogram of equation 92 for N=10

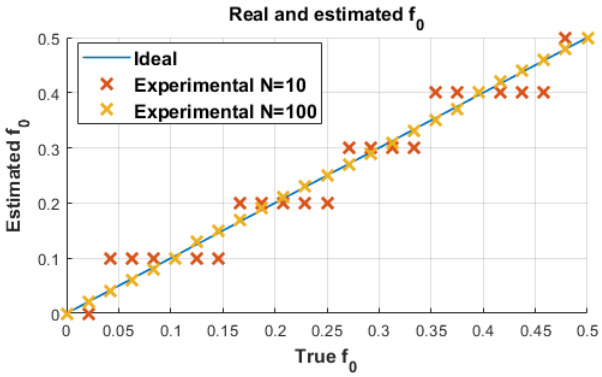


Figure 57: MLE of equation 92

Now, when looking at the MLE, it can be seen that for $N = 10$, the MLE can only take 5 unique values, which corresponds to $\frac{N}{2}$, and the number of points on the periodogram between $f_0 = 0$ and $f_0 = \frac{1}{2}$. This means that the number of data points, N, essentially corresponds to number of quantisation levels of the MLE. For $N = 100$, the number of values that the MLE of f_0 can take between 0 and 0.5 is 50, which clearly is sufficient to estimate the value of f_0 accurately.