Advanced Signal Processing

Vitiello Pietro

January 2020

Contents

1 Random Signals and Stochastic Processes

1.1 Statistical estimation

Using the MATLAB function `rand` we can generate a vector `x` of 1000 samples where each sample comes from a uniform distribution $U(0,1)$. This vector will represent one realization of a random process X_n. We can then plot the realization as seen in Figure 1.1.

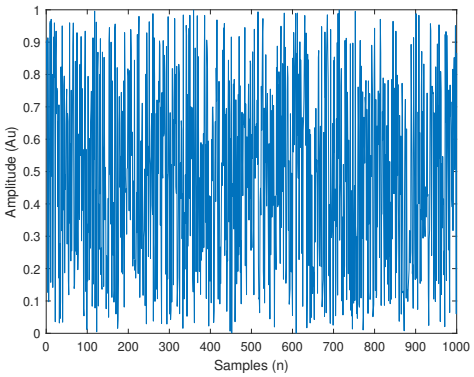


Figure 1.1: One realization of the process X_n

1.1.1

The theoretical expected value of X_n is $E\{X_n\} = E\{U\} = 0.5$ since U is a uniform distribution. However the vector `x` is a finite length realization therefore by taking the sample mean using `mean` we only get an estimate of the expected value. Based on the realization seen above we get a sample mean of approximately 0.5127 which corresponds to an error of 0.0127. The formula used for the sample mean is an unbiased estimator of the expected value of the process, therefore the more samples you consider the closer the sample mean should be to 0.5, meaning that the estimate will be more accurate as $n \to \infty$.

1.1.2

We can also easily find the theoretical standard deviation of X_n which coming from a uniform distribution would simply be $\sigma_{X_n} = \sigma_U = \frac{1}{\sqrt{12}}$. However, as we have said for the mean, `x` is a finite realization and hence by taking the sample standard deviation using `std` we only get an estimate of the true σ_{X_n}. For the realization shown in Figure 1.1 the sample standard deviation is approximately 0.2893 which corresponds to an error of circa -0.0006. The formula used to calculate this is also an unbiased estimator, therefore as $n \to \infty$ the accuracy of the estimate will increase until the true standard deviation is matched.

1.1.3

We can then generate ten 1000-sample realizations of X_n. It is possible to find the sample mean and sample standard deviation for each realization, what I have obtained is shown in the Table 1.1. We can also plot the results as in Figure 1.2 where we can see for each realization the value of the estimates and compare it with the theoretical value (the horizontal line).

By looking at the plots in Figure 1.2 we can see that the estimates cluster in a uniform way around the theoretical value, in the sense that there are approximately the same number of estimates above and below it. In fact we can make an estimate of the bias of the estimator, which for the mean resulted in a bias of approximately -0.0008 and for the standard deviation of approximately 0.0001. We can clearly see that already with only ten realizations the bias is very low, but as the number of realizations approached infinity the bias would go to zero, being the two estimators unbiased.

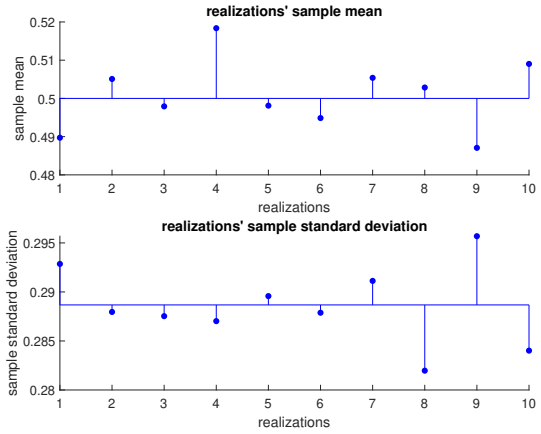


Figure 1.2: $\hat{m}$ and $\hat{\sigma}$ plotted for each realization with visible distance from theoretical value

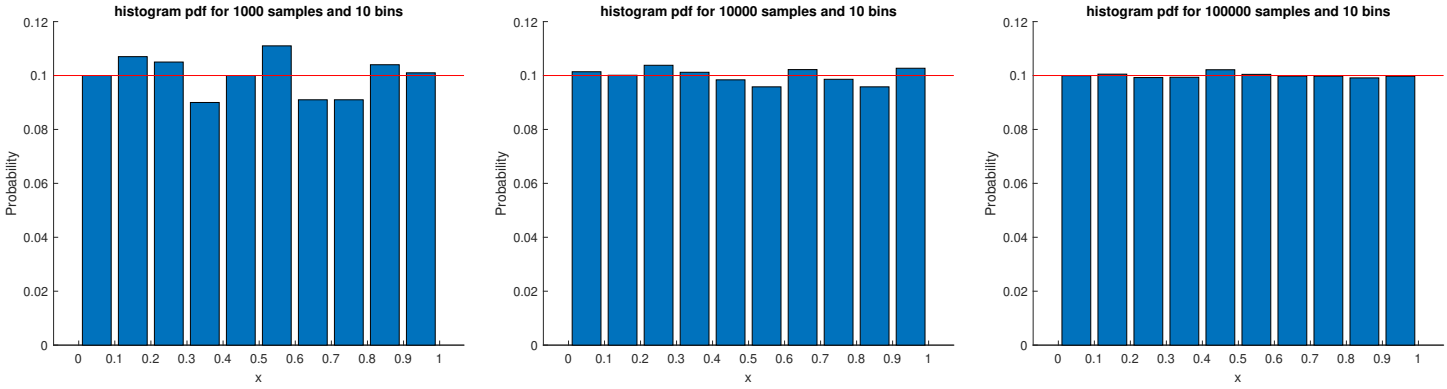
realization	$\hat{m}$	$\hat{\sigma}$
1	0.4897	0.2929
2	0.5051	0.2880
3	0.4979	0.2875
4	0.5184	0.2870
5	0.4981	0.2896
6	0.4949	0.2879
7	0.5054	0.2911
8	0.5029	0.2820
9	0.4871	0.2957
10	0.5090	0.2840

Table 1.1: Table representing $\hat{m}$ and $\hat{\sigma}$ for each realization

1.1.4

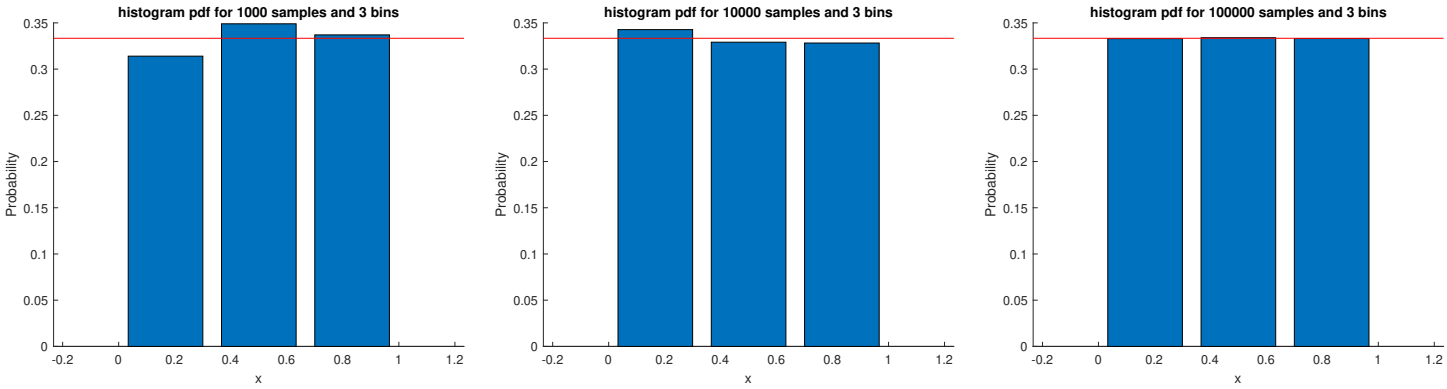
Using the function `hist` we can use a histogram to approximate the probability density function (pdf) of X_n. The results obtained have been plotted in Figure 1.3 and Figure 1.4 together with the theoretical pdf. The property of the pdf is that by summing all the bins the result should be one, hence the theoretical distribution changes with the number of bins, for instance it is uniform at 0.1 for ten bins and it is uniform at 0.33 for three bins.

The histograms have been represented with ten and three bins for different numbers of samples. Therefore we have a better idea of how the number of bins and of samples influence the pdf. Figure 1.3 shows how as the number of samples increases, keeping the number of bins constant, the approximated pdf better resembles the theoretical distribution (shown in red). Additionally from Figure 1.4 we can see the same plots of the previous figure, but with three bins instead of ten. We can clearly see how as the number of bins decreases the pdf gets more precisely approximated, which makes sense as each bin covers a wider range of values therefore resulting in a more uniform distribution of the values over each bin.



(a) 1,000 samples (b) 10,000 samples (c) 100,000 samples

Figure 1.3: Comparing pdf generated with different number of samples using 10 bins for the `hist` function



(a) 1,000 samples (b) 10,000 samples (c) 100,000 samples

Figure 1.4: Same pdf as the three above but using 3 bins

1.1.5

We can also repeat the same analysis but considering a process Y_n whose samples come from a Gaussian distribution generated by the function `randn` with mean 0 and σ of 1. Considering a 1000-sample vector **y** representing a realization of the process we get Figure 1.5.

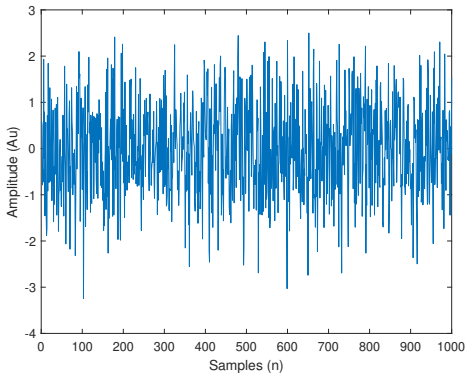


Figure 1.5: One realization of the process Y_n

We know that the function used in this exercise generates samples coming from a normal distribution of mean 0 and σ of 1. Therefore the theoretical values are $E\{Y_n\} = 0$ and $\sigma_{Y_n} = 1$. We can then use the same functions as in sections 1.1.1 and 1.1.2 to find the sample mean and sample standard deviation, which respectively turn out to be approximately 0.0663 (Error of 0.0663) and 1.0057 (error of 0.0057). As we have already said the estimators are unbiased, hence as the number of samples considered approaches infinity, the error will go to zero.

Furthermore we can consider ten realizations of Y_n and compute the $\hat{m}$ and $\hat{\sigma}$ for each one. The results can then be plotted compared to the theoretical value, as shown in Figure 1.6

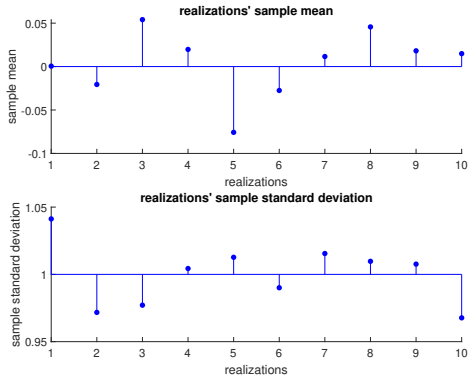
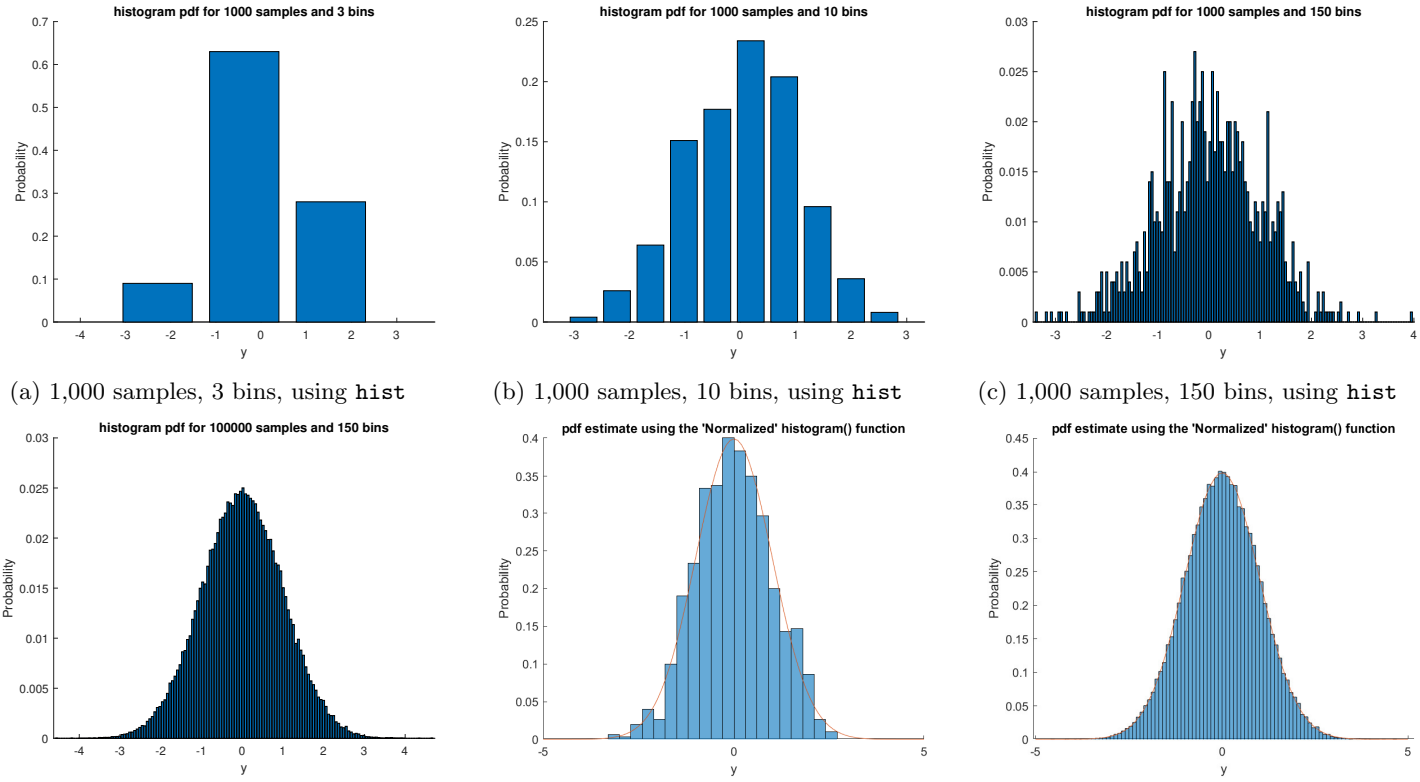


Figure 1.6: $\hat{m}$ and $\hat{\sigma}$ plotted for each realization with visible distance from theoretical value

From the data in Figure 1.6 we can calculate that the estimate of the bias for $\hat{m}$ is ~ 0.0011 while for $\hat{\sigma}$ is ~ 0.0088. If we compare these results with the one found in part 1.1.3 we can see that the Gaussian distribution has yielded higher bias estimates, this could be due to the fact that with the same number of samples the $\hat{m}$ and $\hat{\sigma}$ have better accuracy if coming from a uniform distribution.

Finally we can plot the estimated pdf as we have done for part 1.1.4. From Figure 1.7 we can see that the estimation can vary substantially. Unlike the uniform distribution, the Gaussian is not well represented neither by few bins (Figure 1.7a) nor by many (Figure 1.7c), the optimal solution is somewhere in between and depends on the number of samples; for 1000 samples I found that the pdf is represented better by 10 bins (Figure 1.7b). On the other hand as for the uniform distribution, as the number of samples increases the estimate better matches the theoretical pdf (Figure 1.7d). However the best estimate of the pdf is obtained using the function `histogram` instead of `hist` as this is able to normalize the data so that the distribution has total area of 1. The results obtained are shown in Figure 1.7f and Figure 1.7d, where the estimate is plotted together with the theoretical (shown in red).



(a) 1,000 samples, 3 bins, using `hist`

(b) 1,000 samples, 10 bins, using `hist`

(c) 1,000 samples, 150 bins, using `hist`

(d) 100,000 samples, 150 bins, using `hist`

(e) 1,000 samples, using `histogram`

(f) 100,000 samples, using `histogram`

Figure 1.7: Comparing pdf generated with different number of samples and bins

1.2 Stochastic Processes

In this section we are exploring the differences between time averages and ensemble averages. Important concepts to have in mind when looking at these topics are stationarity and ergodicity.

- Strict Sense Stationarity implies that the statistical properties of a random process, such as its mean, variance, standard deviation and autocorrelation, do not depend on time.
- Ergodicity, instead, is used to refer to processes where taking the time average yields the same result as taking the ensemble average, therefore having one infinite realization is like having infinite realizations.

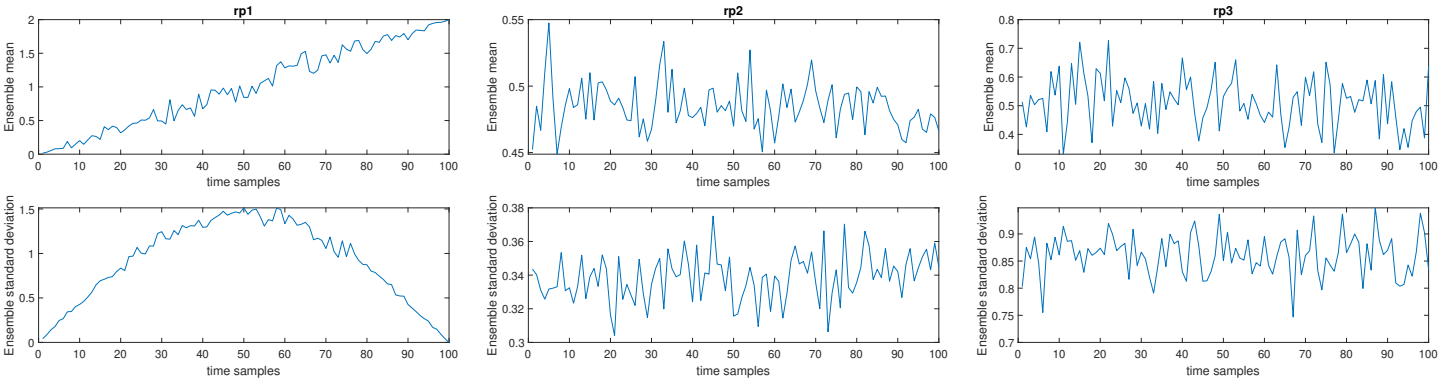
We have been provided with three MATLAB functions that generate M realizations of a random process made of N samples. To conduct the exercise it has been chosen to analyze 100 realizations of 100-samples long processes. The ensemble average of the process can be computed by taking the average of all the random variables of the realizations at each point in time. While an estimator of the expected value of the process is the time average of the ensemble. A crude way of understanding this is that the ensemble average is the vertical average across all realizations, while the time average is the horizontal average across time.

1.2.1

In Figure 1.8a we can see plotted the ensemble average and ensemble standard deviation of the process obtained using the *rp1* function. It is clear that the ensemble mean increases with time following what can be considered to be a line. On the other hand the ensemble standard deviation also changes with time, but follows a curve which increases to then decrease back, resembling what can be thought of half a period of a sine wave. The dependence on time of both ensemble mean and σ make this process non stationary.

In Figure 1.8b are plotted the ensembles for the process obtained from the *rp2* function. We can see that both the ensemble mean and σ cluster around a value, they never diverge completely away. The average ensemble mean for the plot is ~ 0.5039 while for σ is ~ 0.3428. We can therefore consider the process generated by *rp2* as being stationary. The reason why the mean does not stay constant is that there are only a finite number of ensembles, therefore the ensemble mean will only be an estimate of the real expected value of the process.

For the process generated by *rp3*, represented in Figure 1.8b we can follow the same reasoning as for the one generated by *rp2*. The average ensemble mean for the plot is ~ 0.5041 while for σ is ~ 0.8685. We can therefore consider it stationary.



(a) (b) (c)

Figure 1.8: Ensemble mean and standard deviation for the processes generated by the three functions

1.2.2

Additionally we can analyze the ergodicity of the processes generated by the three functions by considering four realizations of each process and calculating the sample mean and standard deviation for each realisation.

Obviously the process generated by *rp1* cannot be ergodic as it is not stationary. Moreover this conclusion can be reached also by thinking that since the ensemble mean increases with time, the number of samples will have an impact on the sample mean, hence the impossibility of this process being ergodic.

The process generated by *rp2* yielded the values found in Table 1.2. It is clear how there is no consistency in the realizations, each realization yields a drastically different sample mean. Therefore this suggests that taking one realization or averaging over many will produce different results, making this process not ergodic.

realization	sample mean	sample σ
1	0.8949	0.2578
2	0.4841	0.1113
3	0.1615	0.2258
4	0.1508	0.0336

Table 1.2: Each realization's sample mean and σ for *rp2* process

realization	sample mean	sample σ
1	0.5049	0.8633
2	0.4880	0.8761
3	0.5158	0.8603
4	0.5048	0.8681

Table 1.3: Each realization's sample mean and σ for *rp3* process

Finally the last process generates consistent values throughout all realizations, as shown from Table 1.3. Therefore we can consider the process generated by *rp3* to be ergodic, since averaging over many realizations yields approximately the same result as considering only one realization.

1.2.3

rp1

One realization of the process generated by `rp1()` is defined as:

$$x_n = a_n \cdot b \cdot \sin(\frac{\pi}{N}n) + cn \tag{1.1}$$

Where a_n is a process whose random variables come from $U(-0.5, 0.5)$, $b = 5$ and $c = 0.02$. We can easily find the expected value and variance of a. Being a uniform distribution $E(a_n) = 0$ and $Var(a_n) = \frac{1}{12}$ which corresponds to $\sigma_{a_n} = \frac{1}{\sqrt{12}}$.
We can then use this knowledge to find the expected value of the process:

$$E\{x_n\} = E\left\{a_n b \cdot \sin(\frac{\pi}{N}n) + cn\right\} = E\{a_n\} E\left\{b \cdot \sin(\frac{\pi}{N}n)\right\} + E\{cn\} = 0 + E\{cn\} = 0.02n \tag{1.2}$$

And the variance of the process:

$$\begin{aligned} Var\{x_n\} &= E\{x_n^2\} - E\{x_n\}^2 = E\left\{a_n^2 b^2 \sin^2(\frac{\pi}{N}n) + c^2n^2 + 2ab\sin(\frac{\pi}{N}n)cn\right\} - c^2n^2 = \\ &= E\left\{a_n^2 b^2 \sin^2(\frac{\pi}{N}n)\right\} + E\{c^2n^2\} + 2E\{a_n\} E\left\{b\sin(\frac{\pi}{N}n)cn\right\} - c^2n^2 = \\ &= E\{a_n^2\} E\left\{b^2 \sin^2(\frac{\pi}{N}n)\right\} + c^2n^2 + 0 - c^2n^2 = \\ &= Var\{a_n\} \cdot b^2 \sin^2(\frac{\pi}{N}n) = \frac{1}{12} \cdot b^2 \sin^2(\frac{\pi}{N}n) = 2.08 \cdot \sin^2(\frac{\pi}{N}n) \end{aligned} \tag{1.3}$$

From the variance we can get that the standard deviation of the process is:

$$\sigma_{x_n} = \sqrt{Var(x_n)} = 1.44 \cdot \sin(\frac{\pi}{N}n) \tag{1.4}$$

We can see how both the mean and the standard deviation depend on n, which proves that the process is not stationary as stated in Section 1.2.1

rp2

One realization of the process generated by rp2() is defined as:

$$x_n = a_n \cdot B + C \tag{1.5}$$

Where a_n is a process whose random variables come from $U(-0.5, 0.5)$ while B and C are random variables with distribution $U(0, 1)$ which remain constant in time but change for each realization. As we did for rp1, we can find the expected value and variance of a_n to be $E(a_n) = 0$ and $Var(a_n) = \frac{1}{12}$ which corresponds to $\sigma_{a_n} = \frac{1}{\sqrt{12}}$. On the

other hand B and C have as expected value $E(B) = E(C) = 0.5$ and share the same variance, and hence standard deviation, with a_n
We can then use this knowledge to find the expected value of the process:

$$E\{x_n\} = E\{a_n \cdot B + C\} = E\{a_n\} E\{B\} + E\{C\} = 0 + 0.5 = 0.5 \tag{1.6}$$

And the variance of the process:

$$\begin{aligned} Var\{x_n\} &= E\left\{x_n^2\right\} - E\{x_n\}^2 = E\left\{a_n^2 B^2 + C^2 + 2a_n BC\right\} - 0.25 = \\ &= E\left\{a_n^2\right\} E\left\{B^2\right\} + E\left\{C^2\right\} + 2E\{a_n\} E\{BC\} - 0.25 = \\ &= Var\{a_n\} (Var\{B\} + 0.25) + (Var\{C\} + 0.25) + 0 - 0.25 = \\ &= \frac{1}{12}\left(\frac{1}{12} + 0.25\right) + \left(\frac{1}{12} + 0.25\right) - 0.25 = \\ &= 0.08 \cdot 0.333 + 0.333 - 0.25 \approx 0.11 \end{aligned} \tag{1.7}$$

The standard deviation is then the square root of the variance, therefore $\sigma_{x_n} \approx 0.33$. We can see how both the mean and the standard deviation do not depend on n, which is a characteristic of stationary processes confirming our conclusion of 1.2.1. Moreover the theoretical values agree with the ones found in the ensemble in part 1.2.1, however being the signal not ergodic the sample mean an standard deviation for the single realizations found in part 1.2.2 do not perfectly match the theroetical values.

rp3

One realization of the process generated by rp3() is defined as:

$$x_n = a_n \cdot m + c \tag{1.8}$$

Where a_n is a process whose random variables come from $U(-0.5, 0.5)$ while $m = 3$ and $c = 0.5$. As we have done for the other processes, we can find the expected value and variance of a_n to be $E(a_n) = 0$ and $Var(a_n) = \frac{1}{12}$ which corresponds to $\sigma_{a_n} = \frac{1}{\sqrt{12}}$.
We can then use this knowledge to find the expected value of the process:

$$E\{x_n\} = E\{a_n \cdot m + c\} = E\{a_n\} E\{B\} + E\{C\} = 0 + 0.5 = 0.5 \tag{1.9}$$

And the variance of the process:

$$\begin{aligned} Var\{x_n\} &= E\left\{x_n^2\right\} - E\{x_n\}^2 = E\left\{a_n^2 m^2 + c^2 + 2a_n mc\right\} - 0.25 = \\ &= E\left\{a_n^2\right\} E\left\{m^2\right\} + E\left\{c^2\right\} + 2E\{a_n\} E\{mc\} - 0.25 = \\ &= Var\{a_n\} m^2 + c^2 + 0 - 0.25 = \\ &= \frac{1}{12} \cdot 9 + 0.25 - 0.25 = 0.75 \end{aligned} \tag{1.10}$$

The standard deviation is then the square root of the variance, therefore $\sigma_{x_n} \approx 0.87$ We can see how both the mean and the standard deviation do not depend on n, which is a characteristic of stationary processes. Moreover the theoretical values approximately match the ones found in part 1.2.1 for the ensemble, confirming our conclusion. Additionally since the process is ergodic, the theoretical values are also reflected in the sample mean and standard deviation of the single realizations in part 1.2.2.

1.3 Estimation of probability distributions

1.3.1

The code shown below represents the function *pdf*, written to estimate the pdf of a collection of samples.

```
function [frequencies, centers, probs] = pdf(x, n_bins)
    N = length(x); %length of the realization
    if (N >= 100)
        [counts, centers] = hist(x, n_bins);
        frequencies = counts/N; %the percentage of points of the realization that
                                %fall in the range represented by the bin
        bin_width = centers(2)-centers(1); %range represented by each bin
        probs = frequencies./bin_width; %normalization of each bin (pdf)
    else
        disp("Number of samples must be greater than 100")
    end
```

This function uses `hist` to generate histogram bins. The bins are then normalized by the number of samples in the realization `x`, received as an input, to obtain the percentage of the samples in each bin. To obtain a pdf it is however necessary to have an area of 1. Therefore the normalized bins are then divided by the width of the bin, scaling the area under the curve to 1.

The function has then been tested on a process with Gaussian pdf and the results are shown in Figure 1.6 .

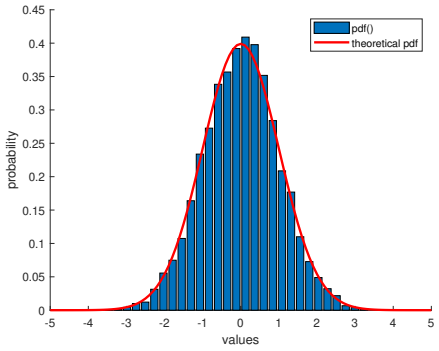


Figure 1.6: histogram pdf generated by `pdf()` using 10000 samples compared to the theoretical pdf in red

1.3.2

The only process that was both stationary and ergodic was that generated by `rp3()`. In Figure 1.7 we can see the estimated density function for realizations of the latter process. It is clear how as the number of samples in the sequence increases the estimate better fits the theoretical pdf shown in red.

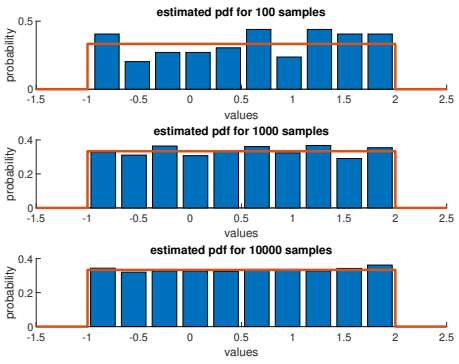
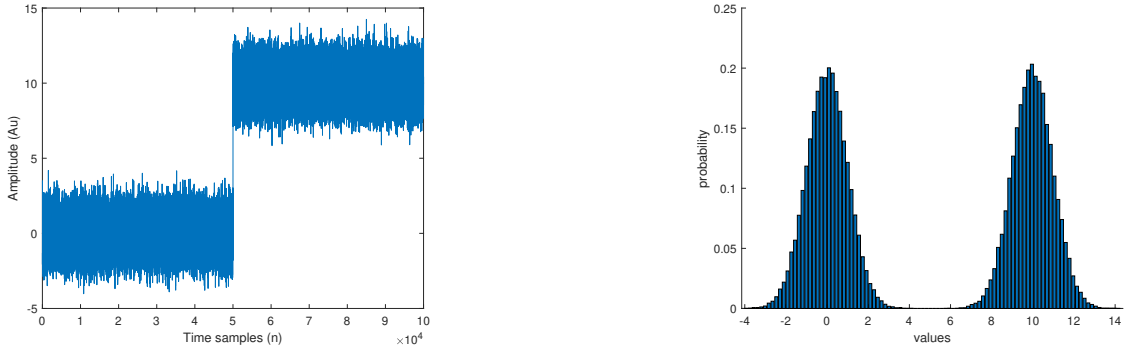


Figure 1.7: Comparison of the estimated and theoretical pdf as the number of samples increases

1.3.3



(a) (b)

Figure 1.8: (a) One realization of process changing mean from 0 to 10 after 5000 samples. (b) Estimated pdf using `pdf()` of the sequence plotted in Figure 1.8b

The function designed is however not ideal when analysing a non stationary process. For instance in Figure 1.8a we can see a sequence of 10000 samples drawn from a normal distribution of $\sigma = 1$ and mean that changes from 0 to 10 at sample 5000. As we can see from Figure 1.8b the pdf produced by `pdf` does not show how the a distribution changes over time. As it is represented it could be assumed that at each time instant the value 0 and 10 are approximately equally likely. However in reality the two halves of the sequence have two separate distributions. Therefore a way to truly represent the pdf would be to have more realizations and find the distribution for each sample by considering the ensemble density.

2 Linear stochastic modelling

2.1 ACF of uncorrelated and correlated sequences

2.1.1

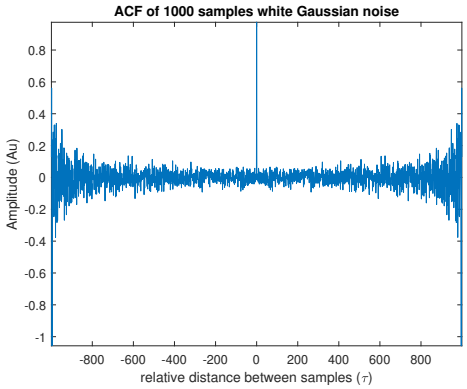


Figure 2.1: Unbiased autocorrelation function of a 1000-samples white Gaussian noise realization

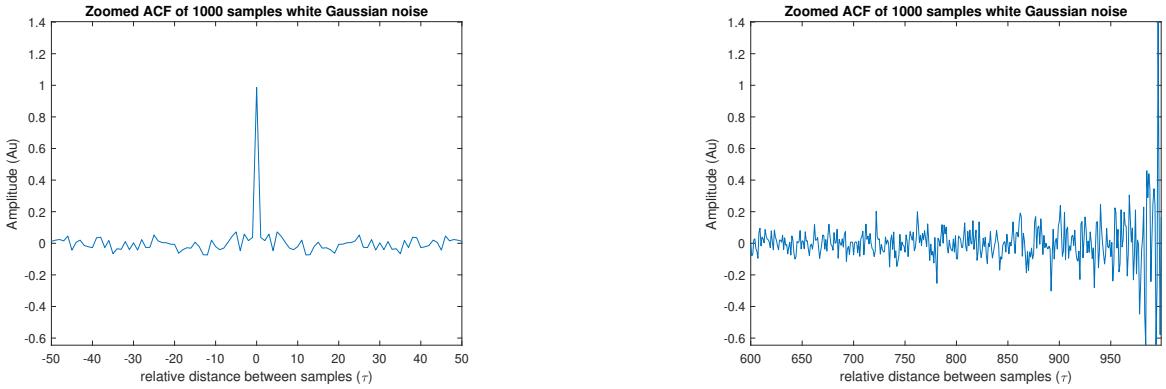
$$\widehat{R}_{xy}(m) = \begin{cases} \sum_{n=0}^{N-m-1} x_{n+m} y_n^*, & m \geq 0, \\ \widehat{R}_{yx}^*(-m), & m < 0. \end{cases}$$

Figure 2.2: `xcorr()` algorithm

Figure 2.1 shows one realization of the process X_n representing white Gaussian Noise. The result is symmetric with respect to $\tau = 0$ since, as shown in Figure 2.2, the algorithm used by *xcorr* looks at the conjugate of the ACF when τ is smaller that 0, however since the signal is real the result is a symmetric AC function. We can notice how the estimated autocorrelation is not 0 everywhere apart from $\tau = 0$ as it theoretically should be, but this is better discussed in part 2.1.3 . However it is important to notice that, being the estimator unbiased, as the number of samples increases the estimate becomes closer to the theoretical function.

2.1.2

Figure 2.3a shows the estimate of the autocorrelation function of section 2.1.1 for $\tau = |50|$. It can be therefore noticed that for small τ the estimated ACF well approximates the theoretical case. On the other hand from Figure 2.1 we can notice that for bigger absolute values of τ the estimate starts to get further away from the theoretical function as $|\tau|$ increases.



(a) (b)

Figure 2.3: The autocorrelation function represented in figure 2.1 shown for τ intervals between -50 and 50 (a) and between 600 and 999 (b)

2.1.3

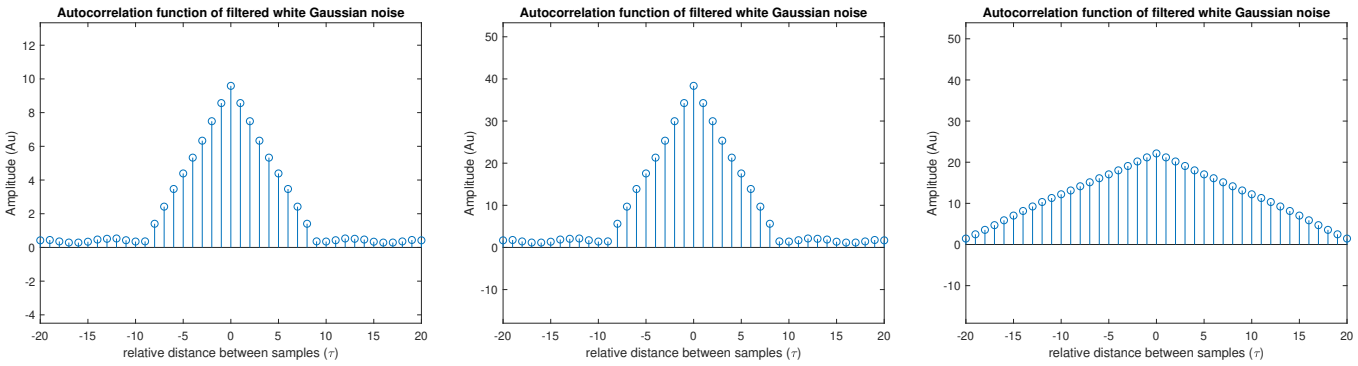
To further analyze the autocorrelation function for high magnitudes of τ, shown in Figure 2.3b, we can refer to Equation 2.1 which represents the formula of the unbiased estimator for the ACF.

$$\hat{R_X}(\tau) = \frac{1}{N - |\tau|} \sum_{n=0}^{N-|\tau|-1} x[n]x[n+\tau], \qquad \tau = -N+1, ..., N-1 \tag{2.1}$$

From this formula we can understand that as $|\tau|$ increases, the number of samples considered for the estimate decreases. Obviously it is more likely to find more values which are similar when considering less values, therefore as $|\tau|$ gets larger the estimate accuracy decreases and the estimated function gets further away from 0 (the theoretical value) as shown in Figure 2.3b. Additionally from Figure 2.1 we can see that the estimator is statistically reliable until approximately $|\tau| = 500$ therefore, leaving a margin, an empirical bound could be $|\tau| = 350$, which is approximately $N/3$.

2.1.4

In Figure 2.4 we can see the autocorrelation function of the noise used in the previous sections, filtered using a moving average filter. The result should theoretically be $R_y = N\Lambda\left(\frac{|\tau|}{N}\right)$ however it is not perfectly the case, especially for higher values of τ where the estimate gets further away from the ideal function.



(a) 9^{th} order with coefficients value of 1 (b) 9^{th} order with coefficients value of 2 (c) 20^{th} order with coefficients value of 1

Figure 2.4: Autocorrelation Fnction of 1000-samples white Gaussian noise using Moving Average filter

What we can analyze is the impact that the filter has on the ACF. As the order increases the Λ function gets wider and higher, as shown in Figures 2.4a and 2.4c, while the opposite is true when the order decreases. It is however important to keep in mind that once the order of the filter gets larger than the number of samples, there is no change in the ACF. Additionally also changing the value of each denominator coefficient, from 1 (Figure 2.4a) to 2 (Figures 2.4b), changes the scale of the function, making it higher as the coefficients get larger.

The MA filter of order equal to the length of the signal can also be used to calculate the sample mean. It can in fact be described by $y[n] = \sum_{k=0}^{N-1} b_k x[n+k]$ which when the coefficients are all $b = \frac{1}{N}$ at $y[0]$ can be compared to the sample mean, as it is the sum of all the values divided by the number of samples.

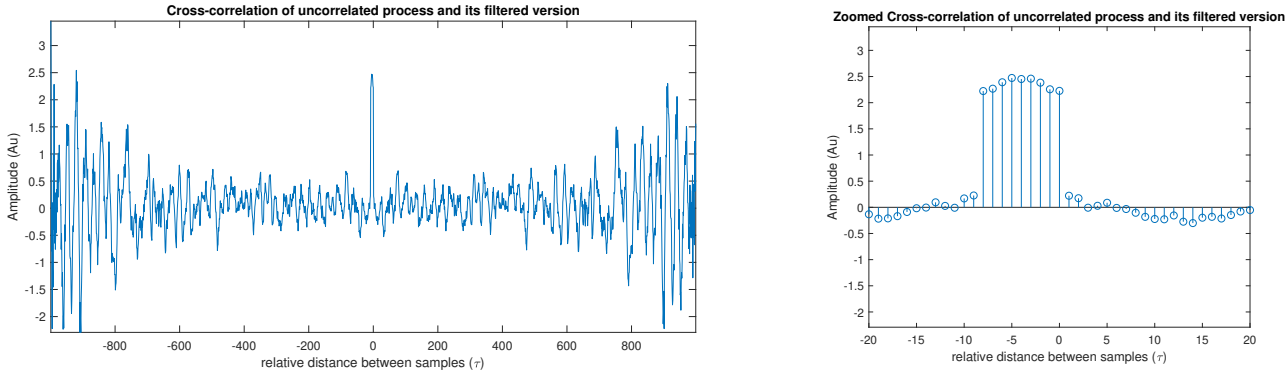
2.1.5

$$R_Y(\tau) = R_X(\tau) * R_h(\tau) \tag{2.2}$$

In Equation 2.2 we consider X_n to be an uncorrelated process and Y_n to be its filtered version. The autocorrelation function R_X of an uncorrelated process is simply a scaled delta function $\alpha\delta(\tau)$ where α corresponds to the squared standard deviation of the process. Therefore, as the convolution of a signal by a delta is the signal itself, the autocorrelation R_Y of the filtered process is the autocorrelation function of the filter's impulse response scaled by α, $R_Y = \alpha R_h$.

2.2 Cross-correlation function

2.2.1



(a) (b)

Figure 2.5: The cross-correlation function of white Gaussian noise and its filtered version shown for τ intervals between -999 and 999 (a) and between -20 and 20 (b)

From Figure 2.5 we can analyze the cross-correlation between the uncorrelated process X_n from section 2.1 and its filtered version. As seen in section 2.1.4 a moving average filter of order 9 generates a process that takes into

consideration the 9 successive samples of the process that is being filtered. Therefore all the samples of $y[n]$ from $n = 0$ to $n = 8$ contain information about $x[8]$. This characteristic is clearly visible in the cross-correlation of the two processes as shown in Figure 2.5b where we can see that the nth sample of x is correlated with the y samples between $n - 8$ and n.

This is also visible in the given formula of the cross-correlation $R_{XY}(\tau) = h(\tau) * R_X(\tau)$ where, when X_n is an uncorrelated process, R_{XY} is the impulse response of the filter scaled by the standard deviation. In this case the impulse response of the filter is a series of deltas from $\tau = -8$ to $\tau = 0$ therefore the cross-correlation keeps the same structure.

2.2.2

In the previous paragraph we have considered an LTI system and we have analyzed the formula of the cross-correlation of a signal with its filtered version. From that equation we can see that by computing the deconvolution of the ACF of the signal and the obtained cross-correlation we can find the impulse response of the filter (which is even easier in the Fourier domain). Moreover as we have previously stated, if the input signal is an uncorrelated process we can deduce the order of the filter by looking at the number of deltas and we can even estimate the coefficients by comparing the magnitude of the deltas of the output with the autocorrelation of the input.

2.3 Cross-correlation function

2.3.1

In this section we will consider an AR model of order 2, that is defined as:

$$x[n] = a_1 x[n-1] + a_2 x[n-2] + w[n], \qquad w[n] \sim \mathcal{N}(0, 1). \tag{2.3}$$

Figure 2.6 shows in red the pairs of the model's coefficients that preserve the stability of the input process. The model whose coefficients lie within the red triangle will produce a bounded output and therefore a WSS process, while the coefficient pairs that are found outside the stability triangle will yield an unstable non-stationary process.

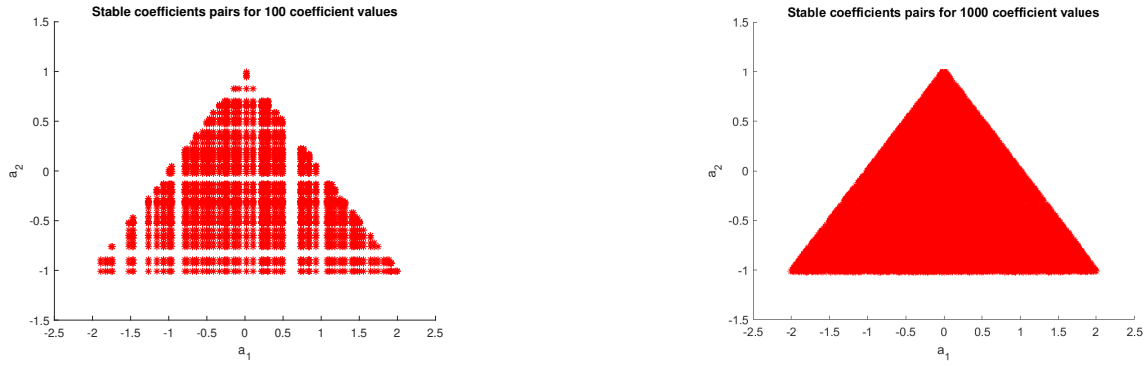


Figure 2.6: In red are the stable pairs of coefficients tested on 100 (left) and 1000 (right) values for each coefficient

The boundaries of the set of coefficients pairs that preserve the stability can be found by analysing the roots (Equation 2.5) of the characteristic equation (Equation 2.4).

$$C(z) = 1 - a_1 z^{-1} - a_2 z^{-2} \tag{2.4}$$

$$z_{1,2} = \frac{a_1 \pm \sqrt{a_1^2 + 4a_2}}{2} \tag{2.5}$$

In fact the magnitude of the roots has to be within the unit circle, hence by setting $|z| < 1$ we can obtain the three conditions shown below.

$$a_1 + a_2 < 1 \tag{2.6}$$

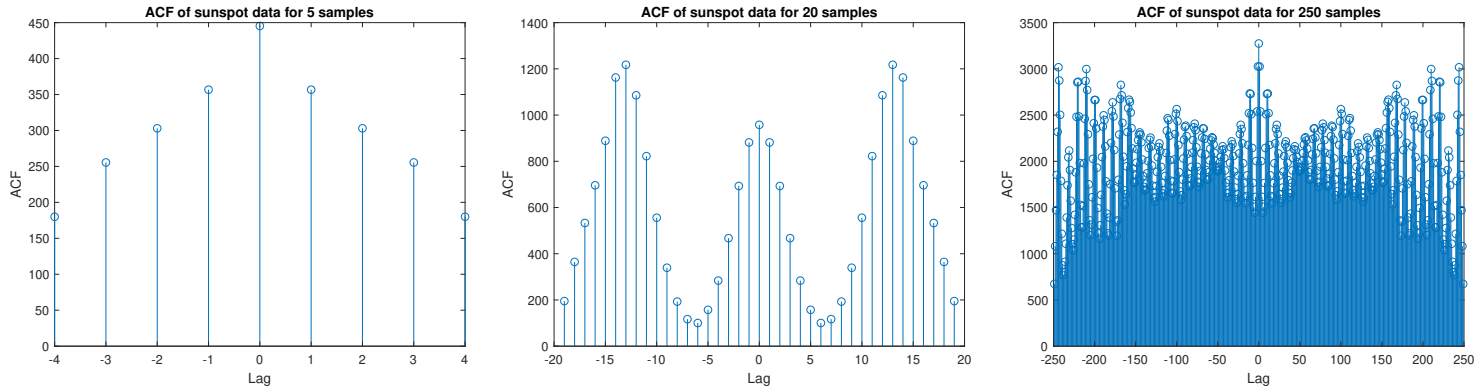
$$a_2 - a_1 < 1 \tag{2.7}$$

$$-1 < a_2 < 1 \tag{2.8}$$

2.3.2

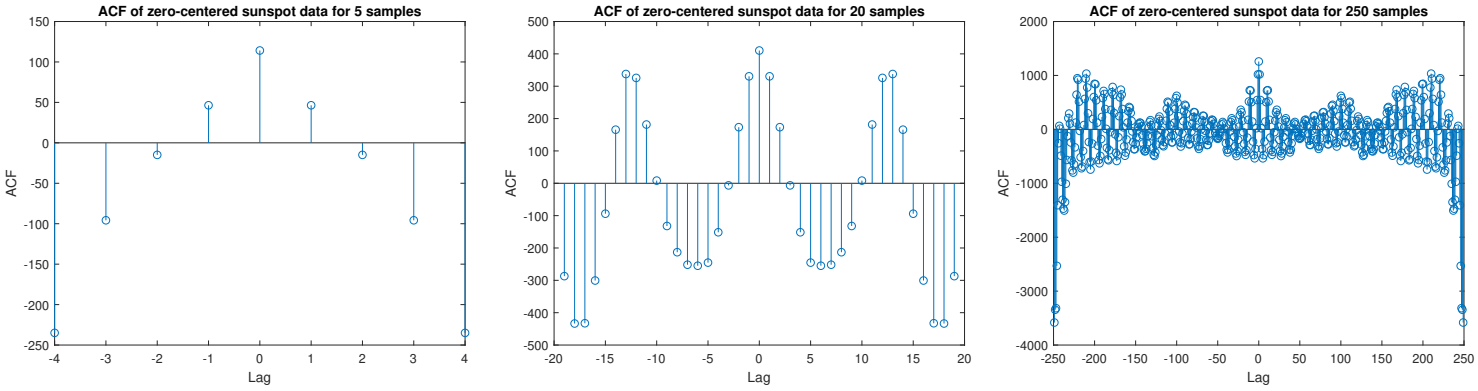
The sunspot time series can be modelled by a second order AR filter and is a good way of experimenting with the ACF. By looking at the evolution shown in Figures 2.7 and 2.8 we can see how considering different sample lengths of the same process affects the ACF. For instance in subfigures (a) we can see how there is no visible periodicity of the function. On the other hand we can observe ocillations in the 20 samples subfigures (b), which matches the expected plot as due to the position of the coefficient pair in the stability triangle we were expecting a decaying oscillation. When looking at 250 samples out of the 288 available, we can however notice an unexpected pattern. This can be explained by recalling section 2.1 where we discussed that the empirical bound could be approximately N/3, which would mean that for Figures 2.7c and 2.8c this bound has been exceeded, therefore resulting in a corrupted estimate.

Additionally we can analyse the effect of DC offset in the data. The sunspot data is in fact not centered in zero, therefore Figure 2.7 is influenced by the deterministic component, that has instead been removed for Figure 2.8 which only considers the stochastic nature of the data. The offset caused some abnormalities, such as the ACF in Figure 2.7b suggesting that a sample had a higher correlation with a shifted sample rather than with itself, which is incorrect and resolved when centering the sunspot data, which yielded Figure 2.7b. Moreover removing the offset also resulted in clearer oscillations which better fit the expected profile of the ACF.



(a) 5 samples (b) 20 samples (c) 250 samples

Figure 2.7: ACF of sunspot data



(a) 5 samples (b) 20 samples (c) 250 samples

Figure 2.8: ACF of zero-centered sunspot data

2.3.3

Shown in Figure 2.9 is the partial autocorrelation function for the empirical sunspot data, in blue and the standardised data in red. Ideally this function should be 0 for all lags above the model's order, however in reality this is not the case and therefore we use a threshold to better judge the importance of a coefficient. The thresholds are shown as a dashed black line and are set to $\pm 1.96/\sqrt{N}$, where N is the data sample length. We can clearly see how the standardised data respects this property more, having the higher orders closer to zero, compared to the empirical data. This being said,

by using the thresholds we can notice that from the third sample onward the magnitude of the PACF is significantly lower with respect to the first two samples, therefore we can assume the model order of the sunspot data to be AR(2).

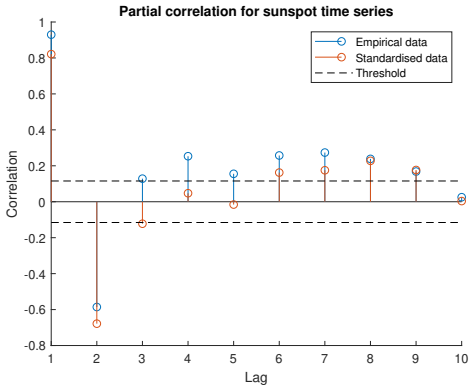
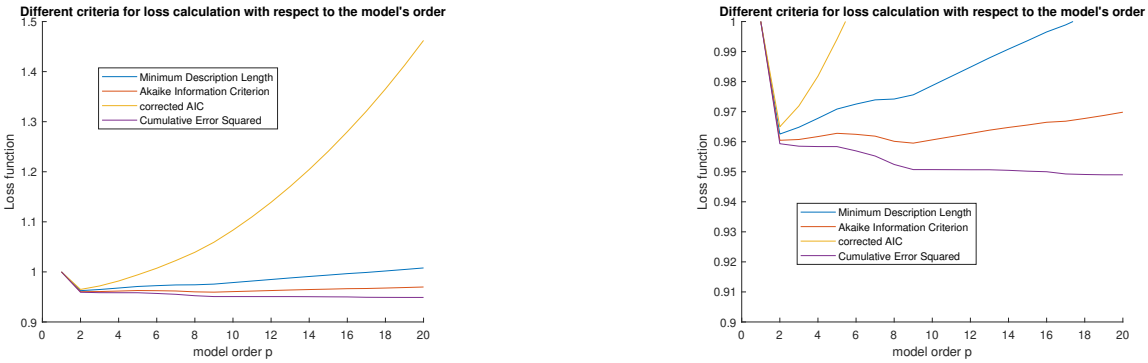


Figure 2.9: PACF for sunspot time series

2.3.4

An additional method to derive the order for a certain data is to look at the loss function of the model for various orders. In Figure 2.10a are shown four loss functions that follow different criteria. The cumulative error squared, shown in purple, only takes into account the error and therefore decreases as the order of the model increases. On the other hand the other models, especially the AIC_c, penalise higher orders as overfitting produces misleading properties.

From the simple error method we can see that there is a local minimum at order two and a global minimum at order nine. Since the difference in loss is only of approximately 0.008 we can consider the order to be the second, as we tend to prefer a lower order. However by looking at the functions that penalise higher orders we can clearly see that order two would be the best choice as it is the one with the lowest loss. Therefore, having analysed these functions, it would be safe to say that the correct model order for the standardised data would be AR(2).



(a) Zoomed out to better show the AIC_c criterion

(b) Zoomed in a smaller error range

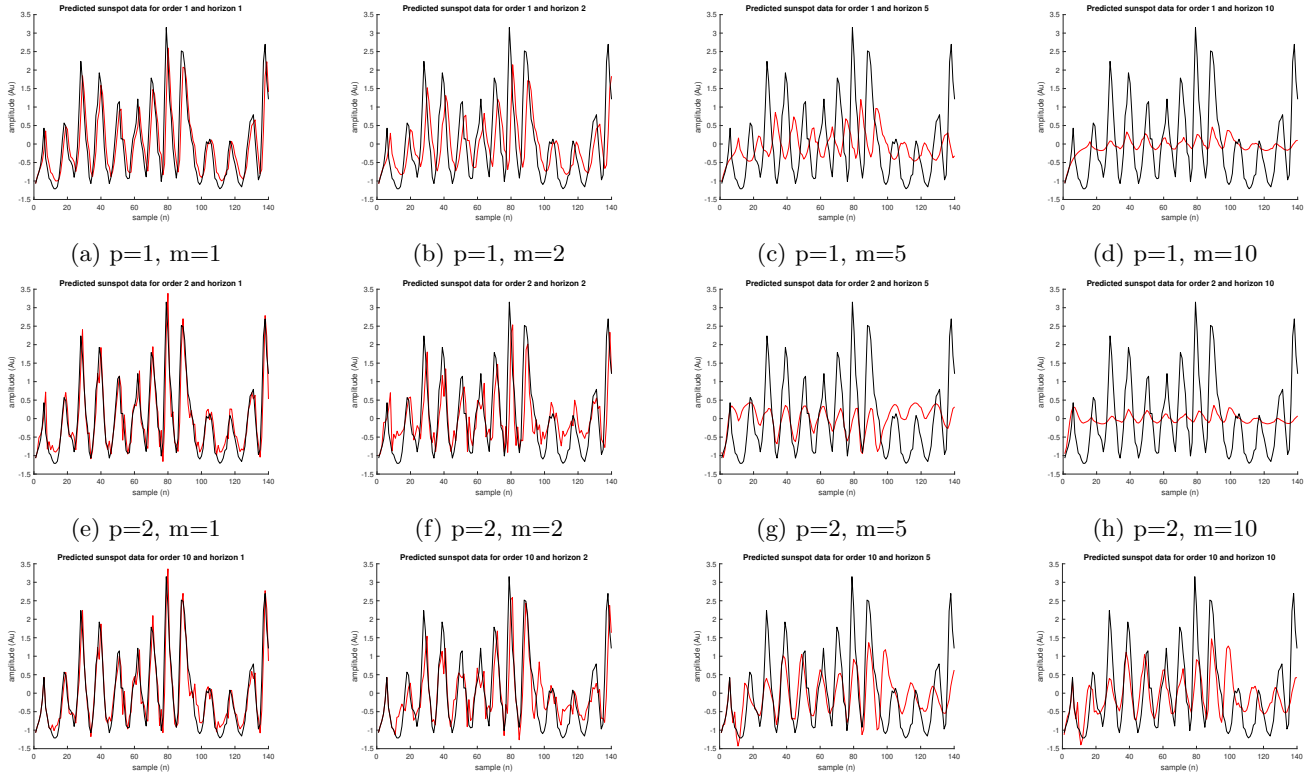
Figure 2.10: Error of the estimate according to different criteria

2.3.5

$$\hat{x}[n+m] = a_1 x[n-1] + ... + a_p x[n-p] \tag{2.9}$$

An AR model of order p can be used to predict the sunspot time series m steps ahead. Equation 2.9 shows the formula for the prediction, which has been used to produce the images of Figure 2.11. What can be noticed from these figures is that increasing the model order also increases the accuracy of the prediction, which makes sense since the larger the order the more points you consider for the prediction. Moreover all three orders display a decrease in amplitude for larger horizons, as it is harder to predict further in the future. Overall we can say that for $m = 1$ all orders produce similar results, however for larger horizons the lower order models produce inaccurate results and is therefore best to

use higher order models. It is, although, important to keep in mind that while higher order modules produce better predictions, they are more likely to overfit the model, there is hence a trade-off between having a better model or prediction.



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

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

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

Figure 2.11: In black the sunspot data and in red the predicted data for the horizon m using a model of order p

2.4 Cramer-Rao Lower Bound

2.4.1.a

We can analyse the NASDAQ data to understand which model order better represents the data. Figure 2.12 shows the PCF of the closing prices up to order 10. We can notice that the PCF magnitude at order one is much higher compared to the other orders, which suggests that the data is almost only correlated to the previous sample. Furthermore from Figure 2.13 we can see the loss functions for the data, which all have a global minimum at the first order. It is therefore possible to deduce that an AR(1) model would be sufficient to model the NASDAQ data provided.

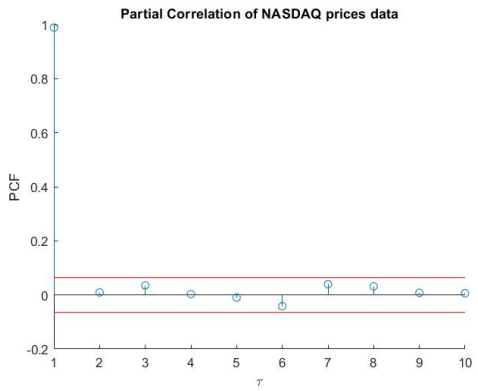


Figure 2.12: Partial autocorrelation function of the NASDAQ close prices

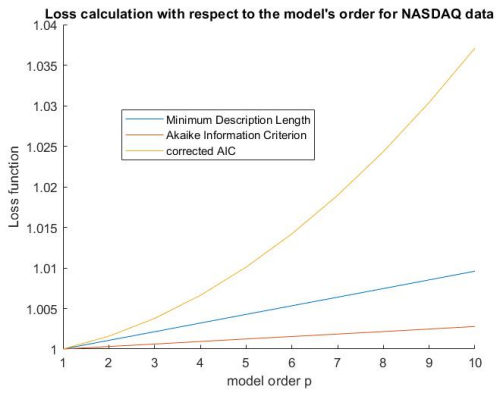


Figure 2.13: Loss function for the NASDAQ data estimates according to different criteria

2.4.1.b

With this data the CRLB for the unknown parameters proves to be a difficult to derive, it is therefore best to use the asymptotic CRLB, which is based on the power spectrum shown in Equation 17 of the coursework. From the expression of the power spectrum it can be proved that the asymptotic log likelihood function is that shown in Equation 2.10. We can use this to derive the elements of the Fisher information matrix following Equation 2.11.

$$\ln\left[\hat{P}_X(f;\boldsymbol{\theta})\right] = \ln\left[\hat{\sigma}^2\right] - \ln\left[1 - \sum_{m=1}^{p} \hat{a}_m e^{-\jmath 2\pi f m}\right] - \ln\left[1 - \sum_{m=1}^{p} \hat{a}_m e^{\jmath 2\pi f m}\right] \tag{2.10}$$

$$[\mathbf{I}(\boldsymbol{\theta})]_{ij} = \frac{N}{2}\int_{-\frac{1}{2}}^{\frac{1}{2}} \frac{\partial \ln\left[\hat{P}_X(f;\boldsymbol{\theta})\right]}{\partial\theta_i} \frac{\partial \ln\left[\hat{P}_X(f;\boldsymbol{\theta})\right]}{\partial\theta_j} df \tag{2.11}$$

To estimate the parameters for the NASDAQ data we can use the expressions above for $p = 1$ as this is the order found in the previous section. What we end up with is therefore the log likelihood function shown below.

$$\ln\left[\hat{P}_X(f;\boldsymbol{\theta})\right] = \ln\left[\hat{\sigma}^2\right] - \ln\left[1 - \hat{a}_1 e^{-\jmath 2\pi f m}\right] - \ln\left[1 - \hat{a}_1 e^{\jmath 2\pi f m}\right] \tag{2.12}$$

We can use Equation 2.12 to find that $[\mathbf{I}(\boldsymbol{\theta})]_{11} = \frac{N r_{xx}(0)}{\sigma^2}$, $[\mathbf{I}(\boldsymbol{\theta})]_{12} = [\mathbf{I}(\boldsymbol{\theta})]_{21} = 0$. Moreover we can take the derivative of the likelihood with respect to $\hat{\sigma}^2$ to find $[\mathbf{I}(\boldsymbol{\theta})]_{22}$ as shown in Equation 2.13.

$$[\mathbf{I}(\boldsymbol{\theta})]_{ij} = \frac{N}{2}\int_{-\frac{1}{2}}^{\frac{1}{2}} \frac{\partial \ln\left[\hat{P}_X(f;\boldsymbol{\theta})\right]}{\partial\hat{\sigma}^2} \frac{\partial \ln\left[\hat{P}_X(f;\boldsymbol{\theta})\right]}{\partial\hat{\sigma}^2} df = \frac{N}{2}\int_{-\frac{1}{2}}^{\frac{1}{2}} \frac{1}{\hat{\sigma}^4} = \frac{N}{2\hat{\sigma}^4} \tag{2.13}$$

Therefore the Fisher information matrix for the NASDAQ data turns out to be as shown here:

$$\mathbf{I}(\boldsymbol{\theta}) = \begin{bmatrix} \frac{N r_{xx}(0)}{\sigma^2} & 0 \\ 0^T & \frac{N}{2\sigma^4} \end{bmatrix} \tag{2.14}$$

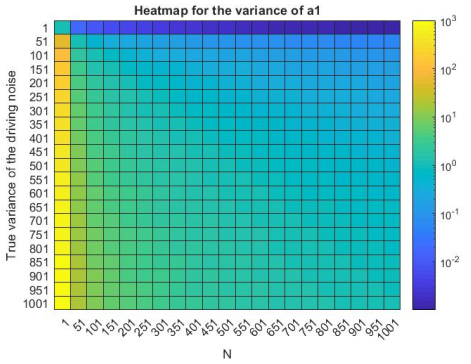
2.4.1.c

If we invert the matrix of Equation 2.14 we obtain a matrix whose diagonal elements are the minimum variance of each parameter, as they belong to the CRLB. The resulting matrix for the NASDAQ data shows that:

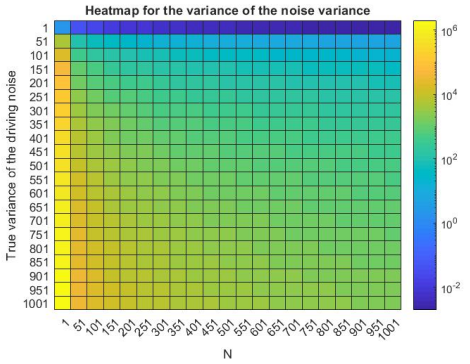
$$\text{var}_{minimum}\left(\hat{\sigma}^2\right) = [\mathbf{I}(\boldsymbol{\theta})]_{22}^{-1} = \frac{2\sigma^4}{N} \quad \therefore \quad \text{var}\left(\hat{\sigma}^2\right) \geq \frac{2\sigma^4}{N}$$

$$\text{var}_{minimum}\left(\hat{a}_1\right) = [\mathbf{I}(\boldsymbol{\theta})]_{11}^{-1} = \frac{\sigma^2}{N\mathbf{r}_{xx}[0]} = \frac{1}{N}\left(1 - a_1^2\right) \quad \therefore \quad \text{var}\left(\hat{a}_1\right) \geq \frac{\sigma^2}{N\mathbf{r}_{xx}[0]} = \frac{1}{N}\left(1 - a_1^2\right)$$

Where $\mathbf{r}_{xx}[0] = \frac{\sigma^2}{1-a_1^2}$. These variance bounds can be visualized using a heatmap as shown in Figure 2.14 where we can see that both variances decrease when the number of samples N increases. Moreover we can also notice the different



(a) Zoomed out to better show the AIC_c criterion



(b) Zoomed in a smaller error range

Figure 2.14: Error of the estimate according to different criteria

dependence on σ that the two variances have, with $\text{var}\left(\hat{\sigma}^2\right)$ being more influenced by it. From the CRLB of the parameter s we can see that as a_1 approaches the value of one, the variance decreases. This reflects on the estimated parameter, which turns out to be circa 0.989. Additionally we can notice that the pole of the filter is at $-a_1$, therefore if the parameter assumed the value of 1 the filter would be unstable. This however makes sense as the snippet of data provided contains a period of time in which the NASDAQ index had a positive trend. Therefore it is reasonable that the model would assume that the trend continued forever and hence the prices would diverge to infinity.

2.4.1.d

The power spectrum given in the coursework for an AR(1) model is shown in Equation 2.15.

$$\hat{P}_X(f;\boldsymbol{\theta}) = \frac{\hat{\sigma}^2}{|1-\hat{a}_1 e^{-j2\pi f}|^2} = \frac{\hat{\sigma}^2}{1+a_1^2-2a_1\cos(2\pi f)} \tag{2.15}$$

We can take the derivatives of $\hat{P}_X(f;\boldsymbol{\theta})$ with respect to both parameters, as shown below, where $A(f) = 1 - a_1 e^{-j2\pi f}$.

$$\begin{aligned}
\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial a_1} &= \frac{\partial}{\partial a_1}\left(\frac{\sigma^2}{1+a_1^2-2a_1\cos(2\pi f))}\right) = \frac{\partial}{\partial a_1}\left(\sigma^2\left(1+a_1^2-2a_1\cos(2\pi f)\right)^{-1}\right) \\
&= (-1)(\sigma^2)(2a_1-2\cos(2\pi f))(1+a_1^2-2a_1\cos(2\pi f))^{-2} \\
&= \frac{2\sigma^2[\cos(2\pi f)-a_1]}{[1+a_1^2-2a_1\cos(2\pi f)]^2} = \frac{2\sigma^2[\cos(2\pi f)-a_1]}{|A(f)|^4}
\end{aligned} \tag{2.16}$$

$$\frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \sigma^2} = \frac{\partial}{\partial \sigma^2}\left(\frac{\sigma^2}{1+a_1^2-2a_1\cos(2\pi f)}\right) = \frac{1}{|A(f)|^2} \tag{2.17}$$

Once we have found the derivatives, we can use those to compute the bound in terms of $A(f)$.

$$\begin{aligned}
\text{var}\left(\hat{P}_X(f;\boldsymbol{\theta})\right) &\geq \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})^T}{\partial \boldsymbol{\theta}} \mathbf{I}^{-1}(\boldsymbol{\theta}) \frac{\partial \hat{P}_X(f;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \\
&\geq \begin{bmatrix} \frac{2\sigma^2[\cos(2\pi f)-a_1]}{|A(f)|^4} & \frac{1}{|A(f)|^2} \end{bmatrix} \begin{bmatrix} \frac{1-a_1^2}{N} & 0 \\ 0 & \frac{2\sigma^4}{N} \end{bmatrix} \begin{bmatrix} \frac{2\sigma^2[\cos(2\pi f)-a_1]}{|A(f)|^4} \\ \frac{1}{|A(f)|^2} \end{bmatrix} \\
&\geq \begin{bmatrix} \frac{2\sigma^2[\cos(2\pi f)-a_1]\cdot\left(1-a_1^2\right)}{N|A(f)|^4} & \frac{2\sigma^4}{N|A(f)|^2} \end{bmatrix} \begin{bmatrix} \frac{2\sigma^2[\cos(2\pi f)-a_1]}{|A(f)|^4} \\ \frac{1}{|A(f)|^2} \end{bmatrix} \\
&\geq \left(\frac{\left(1-a_1^2\right)}{N}\cdot\frac{4\sigma^4[\cos(2\pi f)-a_1]^2}{|A(f)|^8}\right) + \left(\frac{2\sigma^4}{N}\cdot\frac{1}{|A(f)|^4}\right) \\
&\geq \text{Var}_{min}\left(\hat{\sigma}^2\right)\left[\frac{2\left(\cos(2\pi f)-a_1\right)^2\left(1-a_1^2\right)}{|A(f)|^8} + \frac{1}{|A(f)|^4}\right]
\end{aligned} \tag{2.18}$$

It is therefore interesting to notice that the variance of the power spectrum of an AR(1) model is directly proportional to the CRLB of the variance of the estimate of the driving noise's σ^2.

2.5 Real world signals: ECG from iAmp experiment

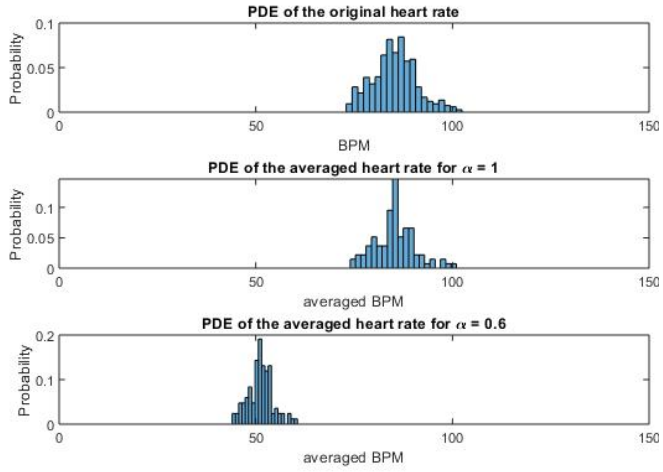


Figure 2.15: PDEs for the first trial

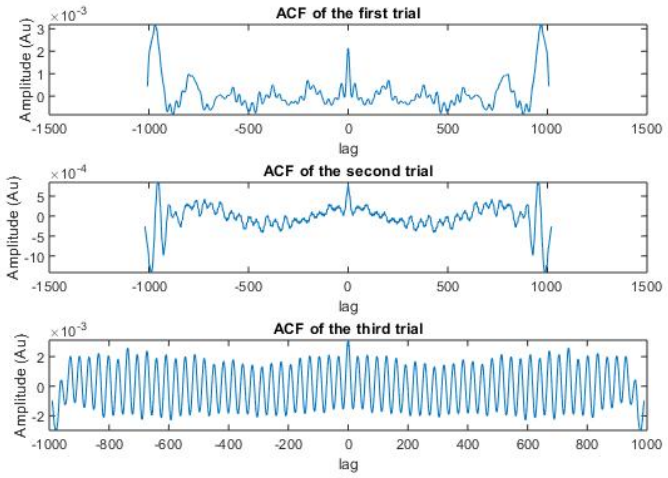


Figure 2.16: Zoomed in a smaller error range

2.5.1

Figure 2.15 shows the various probability density estimates of the heart rate for the first trial. The top histogram represents the original data measured, which has an average of around 85.3bpm. Instead the bottom histograms have been produced using an estimator of the heart rate, which averages the bpms over ten samples, applying a scaling factor α. The middle PDE has been obtained using an $\alpha = 1$ and has the same average as the original data, which is expected since the values have not been scaled. On the other hand the last histogram corresponds to an estimator with $\alpha = 0.6$ which therefore scales the values down by circa 40% and in fact has a lower mean of around 60bpm.

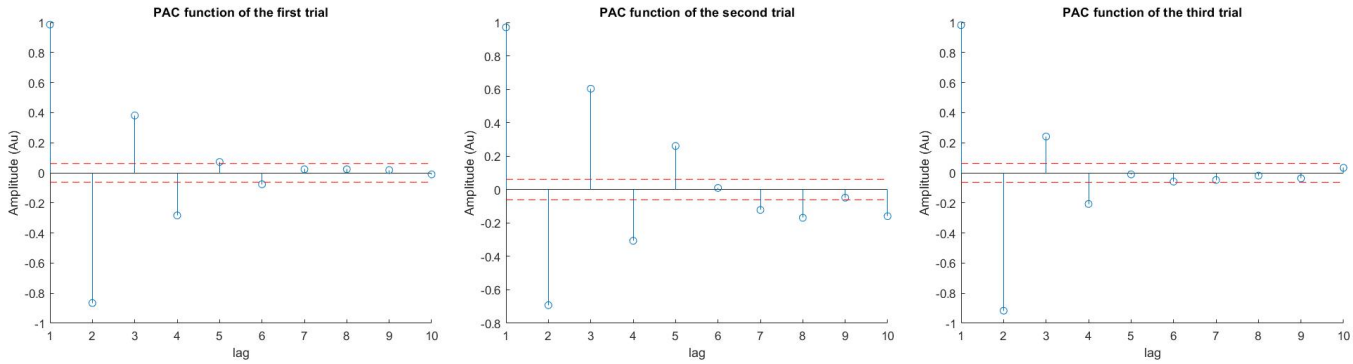
Furthermore we can notice that all three representations of the heart rate produce a PDE similar to a Gaussian distribution. Additionally it is important to say that the scaling factor α affects both the mean and the variance of the distribution. If the original data has mean μ_h and variance σ_h^2, the estimator will yield a mean of $\alpha\mu_h$ and a variance of $\alpha^2\sigma_h^2$. Hence an $\alpha = 0.6$ will not only shift the distribution but also decrease its variance to about a third of the original one (from 31.6 to 11.4) as the highest and lowest values get closer together.

2.5.2

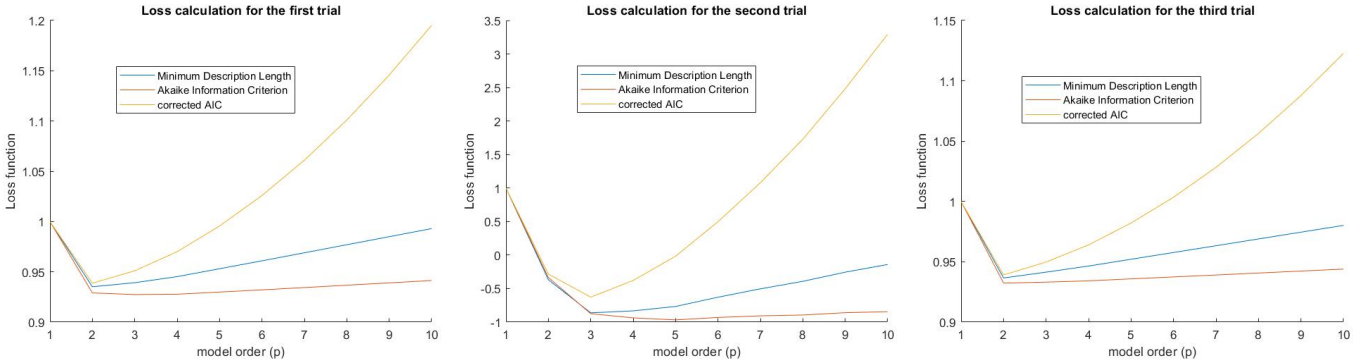
To understand how to model a process it is useful to analyse its ACF. In Figure 2.16 are shown the autocorrelation functions for all three trials. What we can immediately notice is that all three plots don't decay to zero, but instead oscillate, therefore suggesting that the heart rate is an AR process. Moreover the ACFs for infinitely long realizations of the data would likely turn out to be infinite, further pointing at an AR process, as MA processes have a finite ACF. Once we have identified that the process is autoregressive we can also use the PAC and the loss functions to deduce the model order for the three different trials, as shown in Figure 2.17.

From Figures 2.17a and 2.17d we can deduce that the first trial is an AR(2) process. In fact both the MDL and AIC_c have a clear global minimum at order two while the PAC function shows that the first two coefficients are much more significant than the others.

On the other hand the second trial produces a loss function, shown in Figure 2.17e, that has a global minimum in the third order both for the MDL and AIC_c criteria, while using the AIC produces similar error values for order three and four. Additionally by analysing its PAC in Figure 2.17b we can notice that from the fourth order onwards the coefficients get close to the threshold. Therefore the second trial could be approximated by an AR(3).



(a) PAC of the first trial (b) PAC of the second trial (c) PAC of the third trial



(d) Different loss functions for the first trial (e) Different loss functions for the second trial (f) Different loss functions for the third trial

Figure 2.17: Analyses for the AR modelling of the heart rate data. Top three figures show the PAC functions while the bottom ones the loss functions

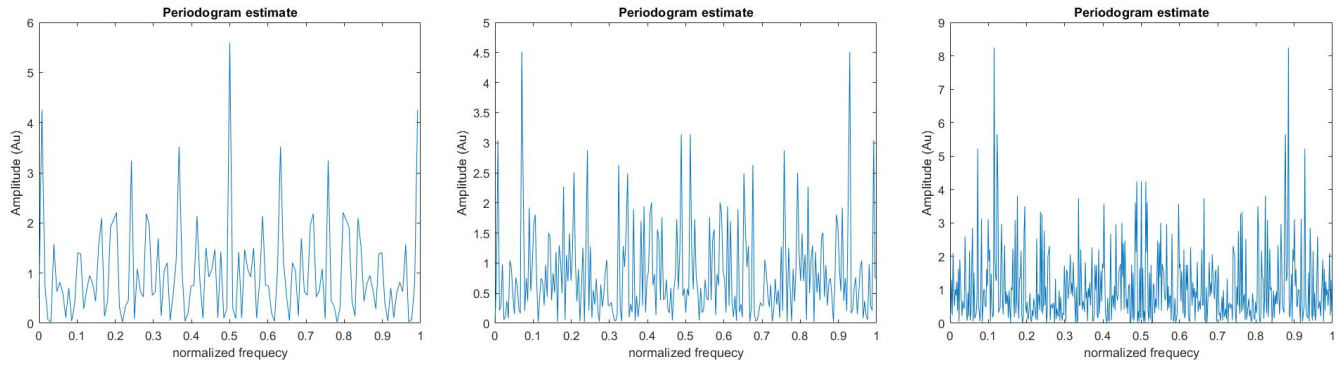
Finally the third trial produces similar results as he first one. Its loss functions have all global minimum in the second order as shown in Figure 2.17f. Moreover the PAC in Figure 2.17c has the first two coefficients that are dominant with respect to the others. Considering these analyses the third trial could be modelled by an AR(2).

3 Spectral estimation and modelling

3.1 Averaged periodogram estimates

Figure 3.1 shows the estimated power spectral density (PSD) against normalized frequency, which in this coursework is measured as $2\pi rad/sample$. The three plots have been generated for WGN realizations of different lengths and the PSD has been obtained by estimating the noise's periodogram using the `pgm` function that I have written. By looking at the resulting plots we can immediately notice that the resulting periodogram is symmetric, which is the case due to the input noise being a real signal. However the obtained results do not represent well the ideal WGN periodogram, which should be a constant σ^2 for all frequencies, hence for our example a constant unity. The difference between the generated PSD and the ideal one is mainly caused by the finite length of the realizations.

By comparing the three different plots we can see that increasing the number of samples makes the mean of the estimated periodogram tend to 1, as this estimator is asymptotically unbiased. Hence, the more samples in the periodogram, the lower the bias. On the other hand the variance is independent of sample length and in fact there is no noticeable decrease in variance when comparing the result obtained from 512 samples and that from 128 samples.

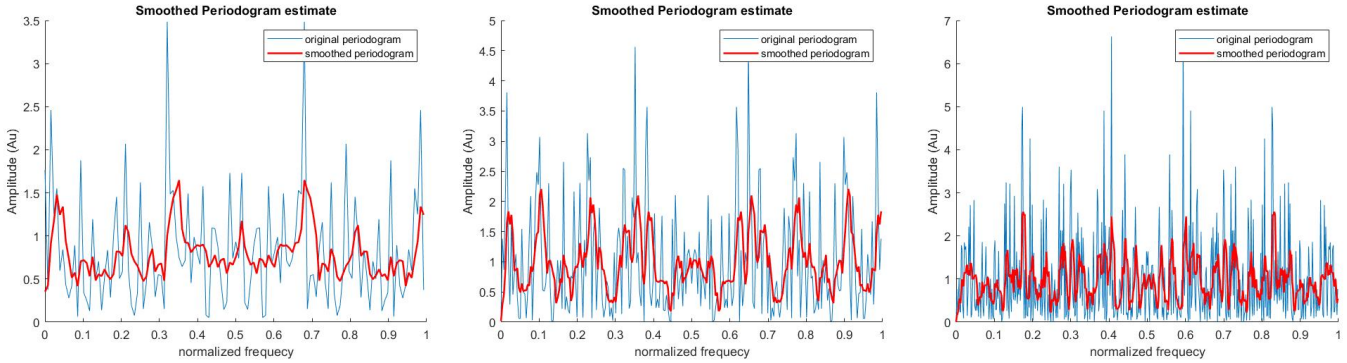


(a) 128 samples (b) 256 samples (c) 512 samples

Figure 3.1: Estimated periodogram for White Gaussian Noise realisations of different lengths

3.1.1

In order to have a better understanding of the shape of the WGN periodogram we can smooth the PSDs obtained above using the `filter` function. By using a zero-phase FIR filter with impulse response 0.2*[1 1 1 1 1] on estimated periodograms of WGN realizations we obtain the results shown in Figure 3.2. By analysing the plots we can understand that smoothing the signal keeps the periodogram's mean approximately unchanged, while it substantially reduces the variance, since the peaks are smoothed down. Therefore using a moving average filter does improve the apparent PSD estimate.



(a) 128 samples (b) 256 samples (c) 512 samples

Figure 3.2: In blue are the periodograms of WGN realizations of different lengths. In red are the filtered periodograms

3.1.2

The Bartlett's method is to divide a long signal realization into smaller segments and then average the segments' periodograms. In Figure 3.3 we can see the estimated PSDs for eight 128-samples realizations obtained by dividing one 1024-samples realization of WGN. We can see how each segment generates a different periodogram, which clearly shows how the spikes in the PSD are not actually dominant frequencies but are just generated by the random nature of the noise. Moreover it can be noticed that the variance remains approximately the same for all segments and is coherent with that found in Figure 3.1a. On the other hand the segments, as previously discussed, being shorter than the original realization, will produce a higher bias compared to the original signal.

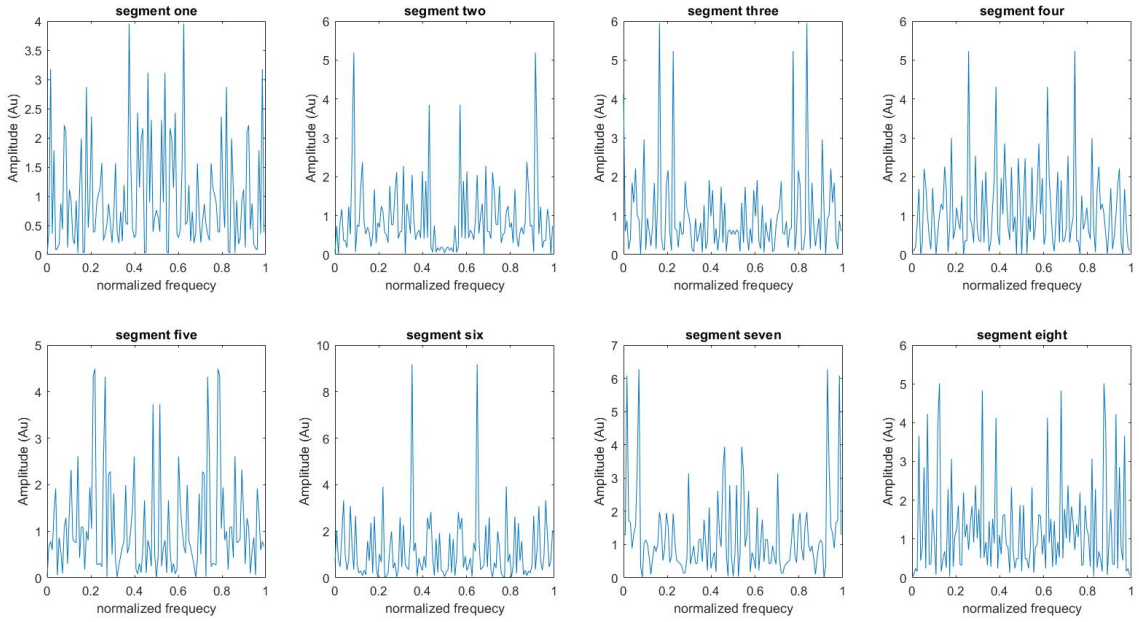
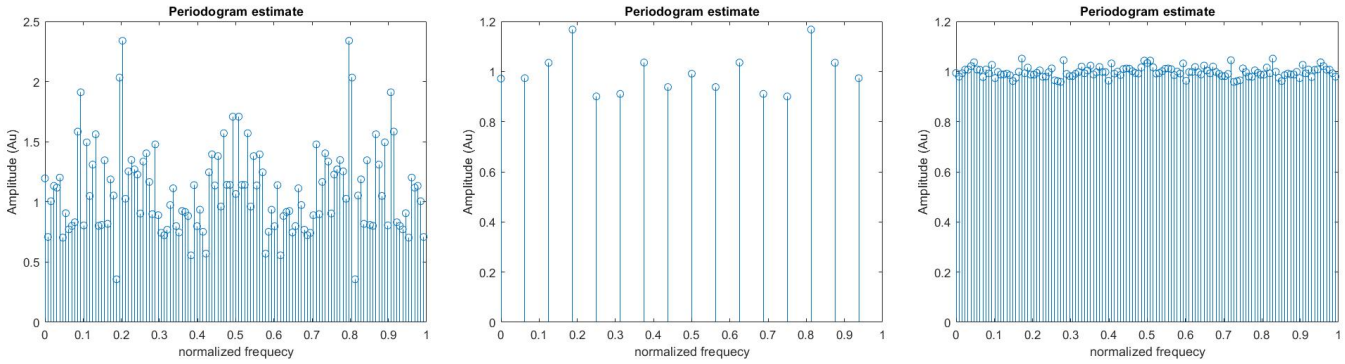


Figure 3.3: PSDs of eigth segments generated from a 1024-samples WGN realization

3.1.3

As stated in section 3.1.2, once you have generated a multitude of segments you can average the peridograms of each segment to produces a much better result. In Figure 3.4a we can see the averaged periodogram obtained from the eight PSDs found above. By averaging the most affected parameter is the variance, which should theoretically be $\frac{\sigma^2}{N}$ where σ^2 is the variance of one of the periodograms and N is the number of segments averaged. In fact we can notice how by increasing the number of segments averaged and keeping the number of samples the same we can greatly reduce the variance, as shown in Figure 3.4c. However you can't always record very long realizations therefore sometimes you are forced to reduce the number of samples per segment. This will indeed reduce the variance, as shown from Figure 3.4b, but it will also increase the bias, as discussed in section 3.1.1. Therefore using the Bartlett's method there is a trade-off between bias and variance. By dividing one realization into segments you will reduce the variance of the estimator but at the same time you will also increase its bias.



(a) 8 segments of 128 samples (b) 64 segments of 16 samples (c) 2000 segments of 128 samples

Figure 3.4: Averaged periodograms for different number of segments and segments' lengths

3.2 Spectrum of autoregressive processes

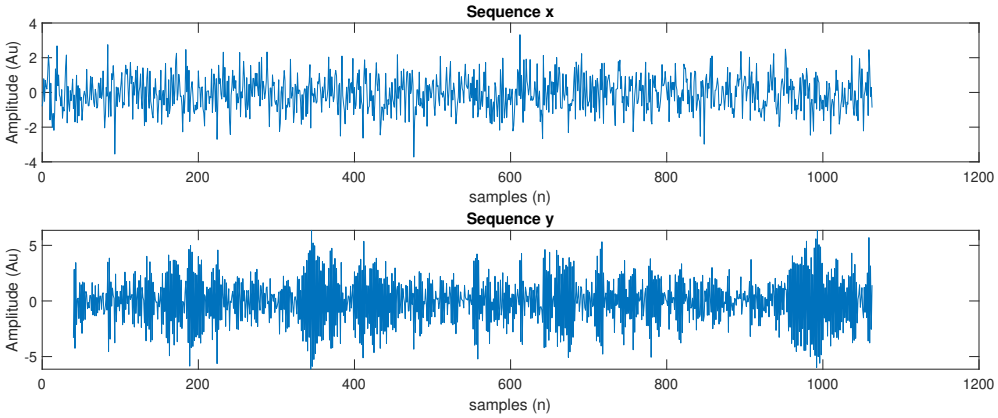
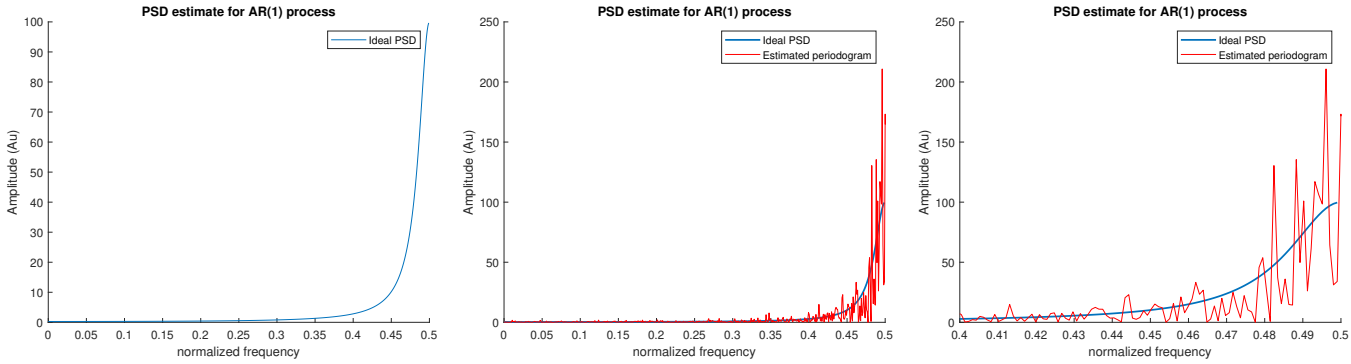


Figure 3.5: Sequence $\mathbf{x}$ and the correspondent filtered sequence $\mathbf{y}$ both in the time domain

Figure 3.5 shows the sequence $\mathbf{x}$, a 1064-samples long realization of WGN, and sequence $\mathbf{y}$, its filtered counterpart. From the plots we can visually see that the sequence $\mathbf{y}$ is very compact and has lost the overall oscillation. This signifies that the lowest frequencies have been filtered out, leaving only the fast oscillations, which cause the signal to seem more densely packed. Hence it can be deduced that the filter used is a high-pass.

3.2.1

To support our theory on the nature of the filter, Figure 3.6a has been generated, showing the ideal PSD of the filtered noise. From this plot it is clear that the filter is supposed to be a high pass, as the lower frequencies have all been attenuated. On the other hand the high frequencies have been hugely amplified. By measuring where there is an attenuation of -3dB, the cut-off normalized frequency has been found to be $f_c = 0.3096$. It is also wort noting that the plots have been cropped as they are symmetric with respect to the normalized frequency $f = 0.5$.



(a) Ideal PSD (b) Ideal and estimated PSD (c) Zooming in on $0.4 \leq f \leq 0.5$.

Figure 3.6: Comparison between the ideal PSD and the estimated periodogram

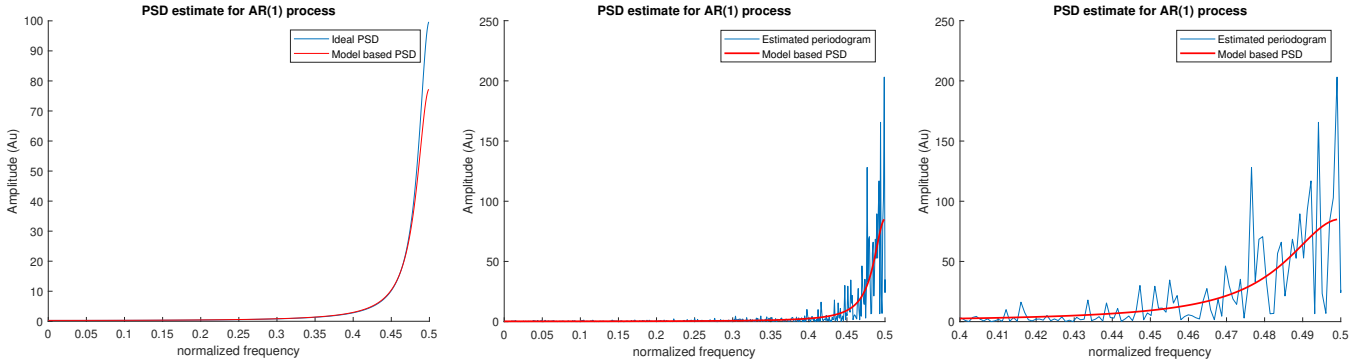
3.2.2 and 3.2.3

Additionally we can compare the ideal PSD to the estimated periodogram generated by `pgm`. The result is shown in Figure 3.6b. From this plot we can see that the estimated periodogram retains the high-pass properties, as it successfully suppresses lower frequencies while amplifying the high ones. On the other hand it can also be noticed how the estimated power is far noisier than the ideal one, which can be better visualized in Figure 3.6c that has been zoomed on the interval $0.4 \leq f \leq 0.5$.

The estimated periodogram is in fact based on a finite realization of noise and therefore is not able to fully mimic the ideal PSD. This is caused by the truncation of the process which is done by multiplying an infinite noise realization with a rectangular window. The truncation in the time domain is in fact equivalent to the convolution of the PSD

with a *sinc* wave in the frequency domain. However this convolution causes unwanted oscillations to appear in the resulting power spectrum. These oscillations are obviously more evident towards the higher frequencies (Figure 3.6c) as the lower frequencies have been attenuated.

3.2.4



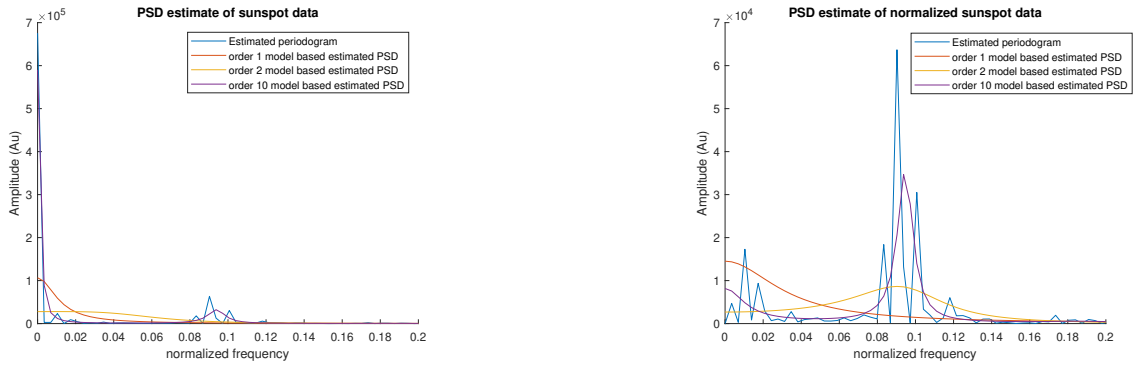
(a) Ideal and model based PSD (b) Model based PSD and periodogram (c) Zooming in on $0.4 \leq f \leq 0.5$.

Figure 3.7: Comparison of the model based estimated PSD with the ideal PSD and the estimated periodogram

Instead of directly estimating the periodogram, another way of estimating the power spectrum of a signal is to model the process itself and then find the PSD of the modeled process. The filtered noise considered can be modelled using an AR(2) with parameters a_1 and σ^2 which can be estimated using the equations 29 and 30 of the ASP coursework. The estimated parameters that have been found are $\hat{a_1} = 0.8963$ and $\hat{\sigma}^2 = 0.9571$. These estimates yield the PSD shown in red in Figure 3.7a. It can be noticed that the ideal PSD and that estimated from the model are not identical, this is because the model still uses estimated parameters, therefore introducing errors in the estimation of the whole process.

However when comparing the PSD generated by the model with the periodogram, as shown in Figures 3.7b and 3.7c we can definitely notice a difference. In fact the function `pgm` does not use prior knowledge about the process from which the signal is coming from, which is instead used for modelling. Consequently the fact that the realization is finite introduces errors due to the convolution with window. On the other hand, when modelling, the finite nature of the signal only affects the estimation of the parameters, however the model is then able to represent the process as a whole, which in turn does not introduce unwanted oscillations caused by the convolution. Therefore when there are enough samples to correctly model the process you obtain much better estimates by modelling rather than directly estimating the periodogram from the realization.

3.2.5



(a) PSD estimates for the original data (b) PSD estimates for the normalized data

Figure 3.8: Model based PSD estimate and periodogram for the sunspot data

Additionally we can repeat the above analysis using the sunspot data. Figure 3.8 shows the estimated PSD for the sunspot data considered in the normalized frequency range $0 \leq f \leq 0.2$, since the power stays approximately constant for higher frequencies. In Figure 3.8a we can compare the periodogram of the original sunspot data with the model based estimated PSD for different orders. As we have discussed in section 2.3 the sunspot data should be modelled by an AR(2). However the tenth order was the one that produced the most similar estimate to the periodogram. This does not mean that the process is better modelled by an order ten, as the periodogram is not the ideal power spectrum, but is simply an estimate. Moreover an AR(10) model would be impractical due to its computational complexity and would be overfitting the data. On the other hand the first order model generates an estimated power spectrum which is far from the periodogram. This result makes sense as the first order does not well represent the periodicity of the data due to the lack of degrees of freedom and hence causes undermodelling. Similarly the second order model does not very well match the periodogram estimate, even though it is the order best suited to represent this data. Nevertheless slightly different results can be seen when considering the mean-centered version of the data, where the estimate produced by the second order better reflects the periodogram when compared to the original data, as shown in Figure 3.8b. Instead the first order is still inappropriate as it completely ignores the $f \approx 0.9$ spike in the power spectrum. On the other hand the order ten remain the most similar to the periodogram.

3.3 The Least Squares Estimation (LSE) of AR Coefficients

3.3.1

In order to better analyse the autoregressive process considered in this question we can let $\mathbf{x} = [\hat{r}_{xx}(1), \hat{r}_{xx}(2), \cdots, \hat{r}_{xx}(M)]^T$, $\mathbf{a} = [a_1, a_2, \cdots, a_p]^T$ and $\mathbf{H}$ be as shown in Equation 3.1. This being said we can show that the cost function found in Equation 33 of the coursework can be expressed as in Equation 3.2. Using these expressions we can then define the model $\mathbf{s}$ as $\mathbf{s} = \mathbf{Ha}$.

$$H = \begin{bmatrix} \hat{r}_{xx}(0) & \hat{r}_{xx}(-1) & \dots & \hat{r}_{xx}(1-p) \\ \hat{r}_{xx}(1) & \hat{r}_{xx}(0) & \dots & \hat{r}_{xx}(2-p) \\ \vdots & \vdots & \vdots & \vdots \\ \hat{r}_{xx}(M-1) & \hat{r}_{xx}(M-2) & \dots & \hat{r}_{xx}(M-p) \end{bmatrix} \tag{3.1}$$

$$J = \sum_{k=1}^{M} \left[\hat{r}_{xx}[k] - \sum_{i=1}^{p} a_i \hat{r}_{xx}[k-i] \right]^2 = \|\mathbf{x} - \mathbf{Ha}\|_2^2 = (\mathbf{x} - \mathbf{Ha})^T (\mathbf{x} - \mathbf{Ha}) \tag{3.2}$$

By using the orthogonality condition on the last Equation we can reach to the conclusion that the Least Squares estimates for the unknown coefficients are found as shown in Equation 3.3. Furthermore we can recall from Section 2.3 that the coefficients can be also estimated using Yule-Walker estimation theory as in Equation 3.4. By comparing the two equations we can understand that $\left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T$ approximates $\mathbf{R}_{xx}^{-1}$.

$$\hat{\mathbf{a}}_{ls} = \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{x} \tag{3.3}$$

$$\hat{\mathbf{a}} = \mathbf{R}_{xx}^{-1}\mathbf{r}_{xx} \tag{3.4}$$

3.3.2

The observation matrix is usually deterministic. However as it can be seen from Equation 3.1, the matrix $\mathbf{H}$ depends on an estimate of the autocorrelation function, meaning that in this case the observation matrix is stochastic. In fact, by considering a different set of recorded data points, the estimated $\mathbf{r}_{xx}$ would change and with it also $\mathbf{H}$.

3.3.3

After having analysed the theory behind the LSE, this approach has been applied to the sunspot data. In this case the data vector $\mathbf{x}$ is the sunspot series itself and the objective is to find the estimated coefficients to model this sequence. The results for the estimates are very promising, as it can be seen from Table 3.1, the LS method produces very similar results to those yielded by the `aryule` function. In the latter table the example regards order four, but the same similarity between the two approaches has also also been confirmed for all orders below ten.

Estimation Approach	$\hat{\mathbf{a}}_1$	$\hat{\mathbf{a}}_2$	$\hat{\mathbf{a}}_3$	$\hat{\mathbf{a}}_4$
LSE	1.3126	-0.4989	-0.1833	0.0515
`aryule`	1.3011	-0.4856	-0.1836	0.0473

Table 3.1: Estimated coefficients for the sunspot model of order four using two different estimation theories

3.3.4

Using the coefficients obtained for question 3 it is possible to generate models of the sunspot series.We can compare the different order models by plotting their respective approximation error to the original signal. The results are shown in Figure 3.9, which takes into consideration various error criteria. From this plot it could be deduced that the order of the sunspot series is the second. In fact, even though both the cumulative and AIC approaches have a minimum at order nine, the error at that order is very similar to that of the AR(2). For this reason it is preferred to use the lowest order, since a higher order will increase the complexity of the model, which is worth doing only if the gain in error is high enough, which is not the case in Figure 3.9. Moreover, it is noticeable that the results obtained are very similar to those of section 2.3, which makes sense since, as shown in question 3, the LSE and `aryule` produce approximately the same coefficients.

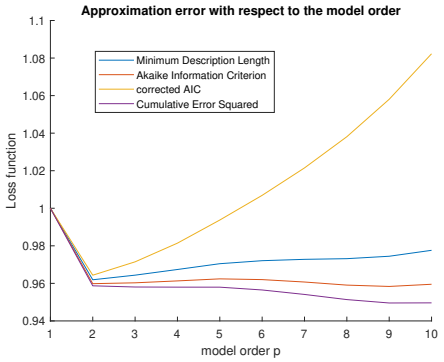


Figure 3.9: Approximation error with respect to the model's order using different criteria

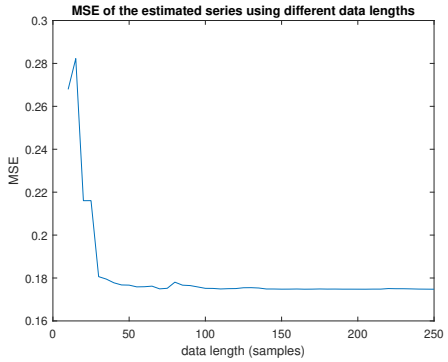


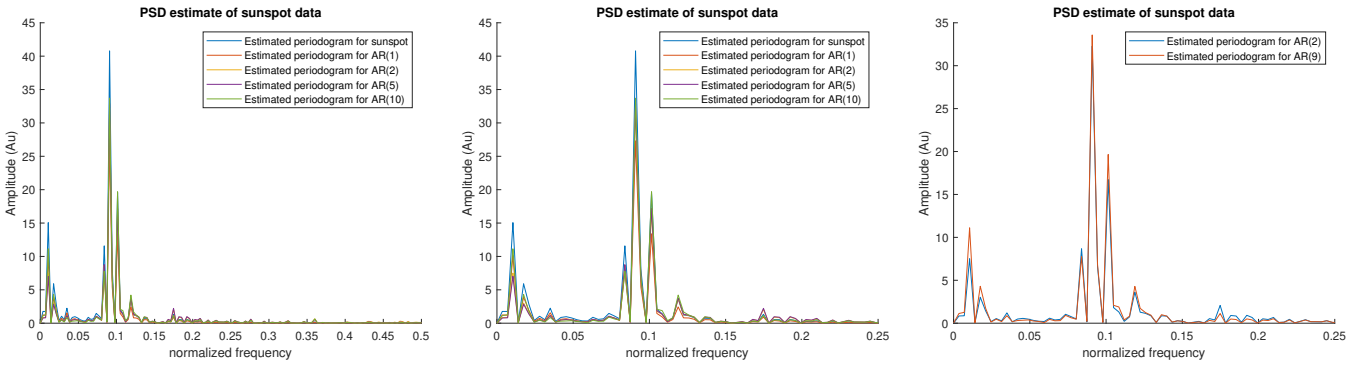
Figure 3.10: Approximation error for the second order model for different data lengths

3.3.5

Another way to determine how close the modelled series is to the original sunspot is to compare their PSD. For this reason the power spectra for orders 1, 2, 5 and 10 have been estimated using the `pgm()` function. From Figures 3.11a and 3.11b we can see that all the AR models are able to produce power spectra that closely approximate that of the original series, with the exception of the AR(1), which shows lower frequency peaks. However, having the four main peaks being all similar heights causes the main harmonics to have roughly the same power therefor. This is incorrect and is hence reflected in a higher error of the modelled AR(1) series, as it is shown in Figure 3.9. This being said, from Figure 3.11c it can also be noticed that the second and ninth order produce nearly identical power spectra. This further confirms what has been previously stated from Figure 3.9 that even though the AR(9) model produces a smaller error, the difference between order two and nine can be considered to be negligible.

3.3.6

From Figure 3.10 we can analyse the approximation error for AR(2) models obtained from different data lengths. In general it can be said that the LSE is a valid method to use for about any data size, but lower data lengths will produce a poor model when compared to longer signals, which is reasonable since the predicted coefficients are less accurate. Hence, as expected, we can see that as we take into consideration more samples, the approximation error of the estimate decreases. This obviously is understandable, since the more samples are used the clearer the properties of the series and hence the better model can be produced. However, it is also worth noting that, even though the minimum error is at 250 samples, the difference with the MSE obtained for 70 samples is in the order of $\text{x}10^{-4}$. Therefore, since a smaller data length will require less computational effort, considering that the error difference with 250 samples is very small it could be stated that the optimum data length for the AR modelling of sunspot time series is 70 samples.



(a) Estimated periodogram for orders 1,2,5 and 10

(b) Subfigure (a) for a smaller frequency interval

(c) Compared PSD of the estimated series using AR(2) and AR(9)

Figure 3.11: Estimated power spectra associated with the AR(p) models of the sunspot time series compared with the periodogram of the original data

3.4 Spectrogram for time-frequency analysis: dial tone pad

3.4.1

The landline number that has been generated for this section is 020 83916226. Using the information given, the **y** signal obtained from this number is shown in Figure 3.13a. By looking at only two digits it can be better noticed how different digits correspond to different signals, as shown in Figure 3.12. The signal has been recorded using a sampling frequency of 32768 *Hz*. This sampling rate allows for frequencies of up to 16384 *Hz* to be recorded without aliasing, which is a range that comfortably allows the recording of the dial tone signal, that should have a maximum frequency of 1477 *Hz*. However it is important to note that sampling at a higher frequency than required does not increase the frequency resolution. In fact, even though the number of samples increases, but also the number of frequencies that can be recorded does, hence the distance between frequencies remains the same.

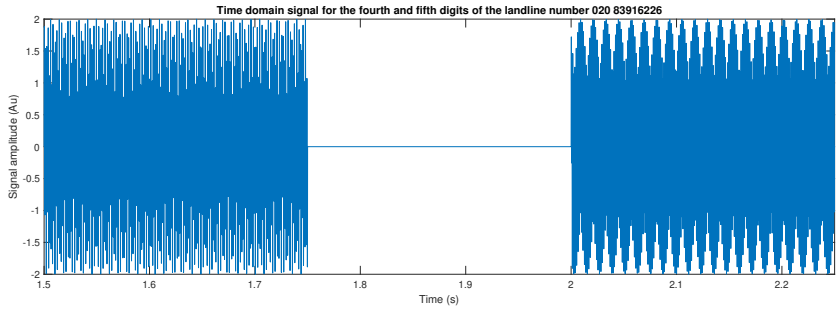
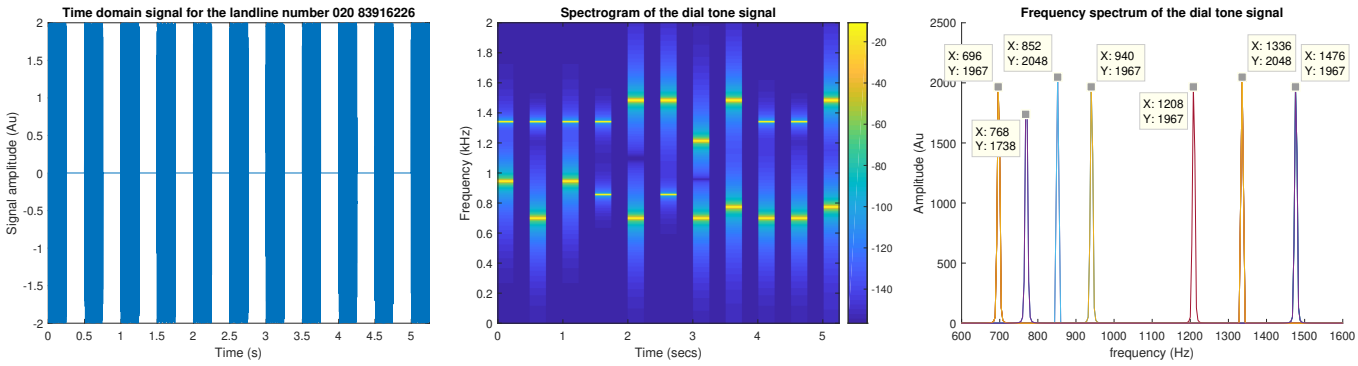


Figure 3.12: Time signal for the digits 8 and 3, separated by an idle period

3.4.2

From the time signal alone it is very hard to recognize which frequencies have been used to produce the waves. Therefore, it s very useful to use a spectrogram, as shown in Figure 3.13b. From the latter it is possible to easily recognize the distinct spectral components of each digit. Additionally we can also produce the full frequency spectrum, as in Figure 3.13c, where each colour represents a different digit. The ideal results would have been delta spikes at the frequencies of the sinusoids. However from both the width of the coloured bands in the spectrogram and the width of the spikes in the spectrum, we can see that the frequency components have been spread over a small range of frequencies. This behaviour was caused by the hanning window used to separate each segment, which being of a finite length introduced a slight error. Furthermore even though each segment is only 8192 samples long, it still needs to represent all the 32768 frequencies. Therefore by considering each segment independently there is a loss in frequency resolution and one sample in the frequency will actually represent four different frequencies. This loss of precision is reflected in the frequency spectrum, where we can notice that the peaks are not perfectly centered on the frequencies shown in the Table 1 of the coursework, but are very close to those, actually positioned on frequencies that are multiples of four, reflecting the frequency resolution discussed.



(a) Time signal (b) Spectrogram (c) Frequency spectrum

Figure 3.13: Analysis of the full untouched dial tone signal

3.4.3

By comparing the spectrogram with the time signal we can see that each band represents either a digit or an idle period. The digits will be composed of two frequency components, from the lowest it is possible to recognize the row on the dial pad and from the highest the column. On the other hand the idle periods will have no frequency components. The key thing in order to be able to understand the sequence of digits is therefore to recognize the peaks. First of all a filter could be used to discard all frequencies below 650 and above 1500 *Hz*, as we are not interested in those. Secondly a threshold could be applied in order to isolate the high peaks, which should represent the frequency components of the sinusoids, that for reasonable levels of noise should be the dominant frequencies.

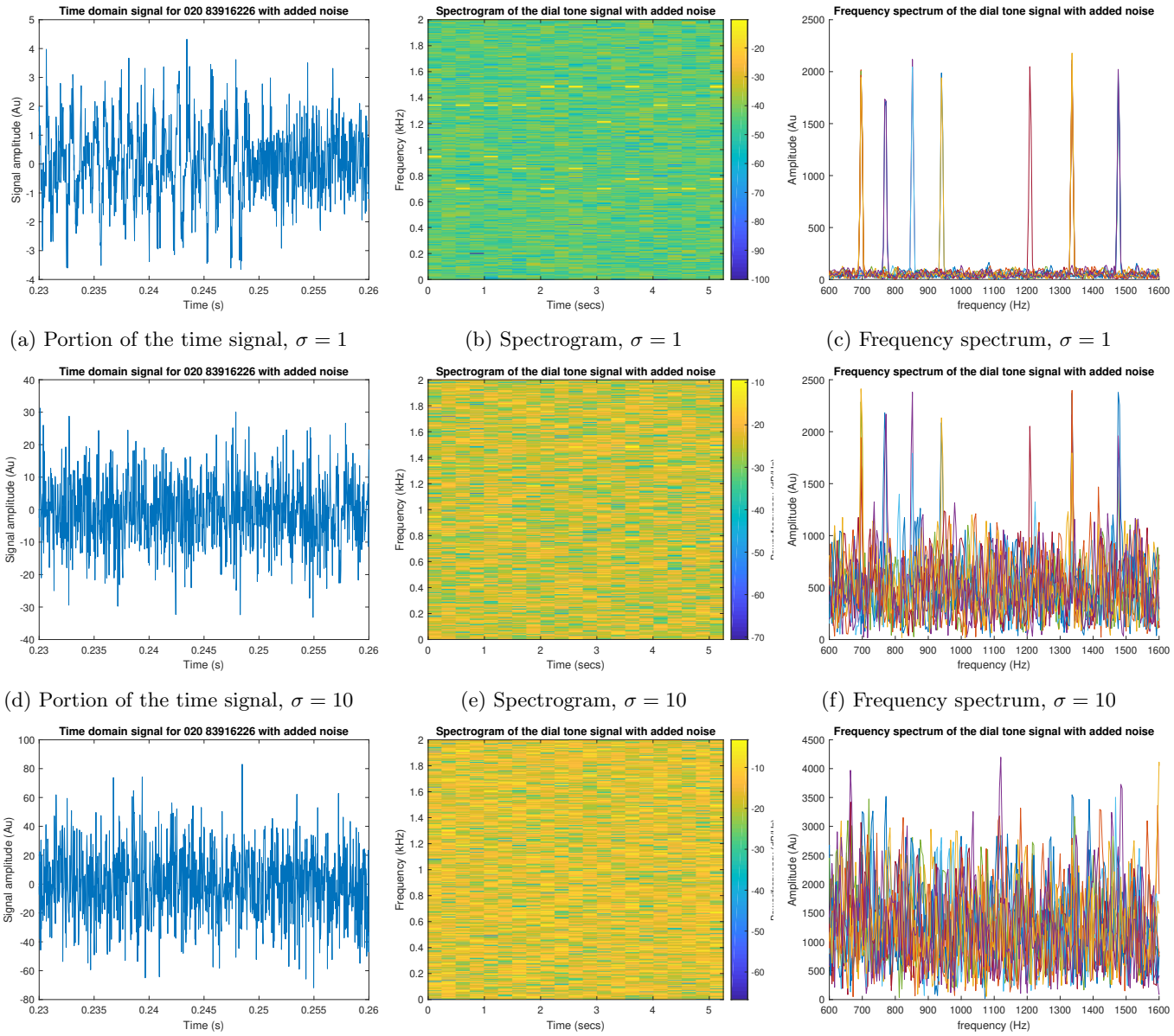
3.4.4

It is important to note that in real-world conditions the signal will get corrupted. The ability to recover the original signal depends on the strength of the noise. In Figure 3.14 are shown the results for the analysis of the same signal used in the previous questions but corrupted with WGN at three different standard deviations: 1, 10 and 25.

Corrupting with low-variance noise will make the time signal close to unrecognizable, however will not drastically affect the frequency domain. From the spectrogram in Figure 3.15b it can be seen how the yellow bands are still distinguishable from the surrounding noise. However the signal's frequency components appear even clearer when looking at the frequency spectrum in Figure 3.15c, where the signal's peaks are significantly higher compared to the noise's components.

On the other hand, when corrupting with $\sigma = 10$, the spectrogram becomes messier. In fact, only by closely examining Figure 3.15e it is possible to recognize the frequency bands at a higher power. Nevertheless, similarly to the previous case, the signal's frequencies can be better isolated from the frequency spectrum, as shown in Figure 3.15f, where the peaks are still clearly visible. However, following the reasoning from Section 3.4.3, it would be suggested to increase the threshold to better identify the peaks.

A different result can be instead found adding with high-variance noise. In this case the corruption's components engulf the signal both in the spectrogram and frequency spectrum. For instance, in Figure 3.15i it is possible to note that the highest peak is just above 1100 *Hz*, which is a frequency that does not belong to the landline signal. It is therefore very hard, without any prior knowledge on the number, to recover the original signal.



(a) Portion of the time signal, $\sigma = 1$ (b) Spectrogram, $\sigma = 1$ (c) Frequency spectrum, $\sigma = 1$

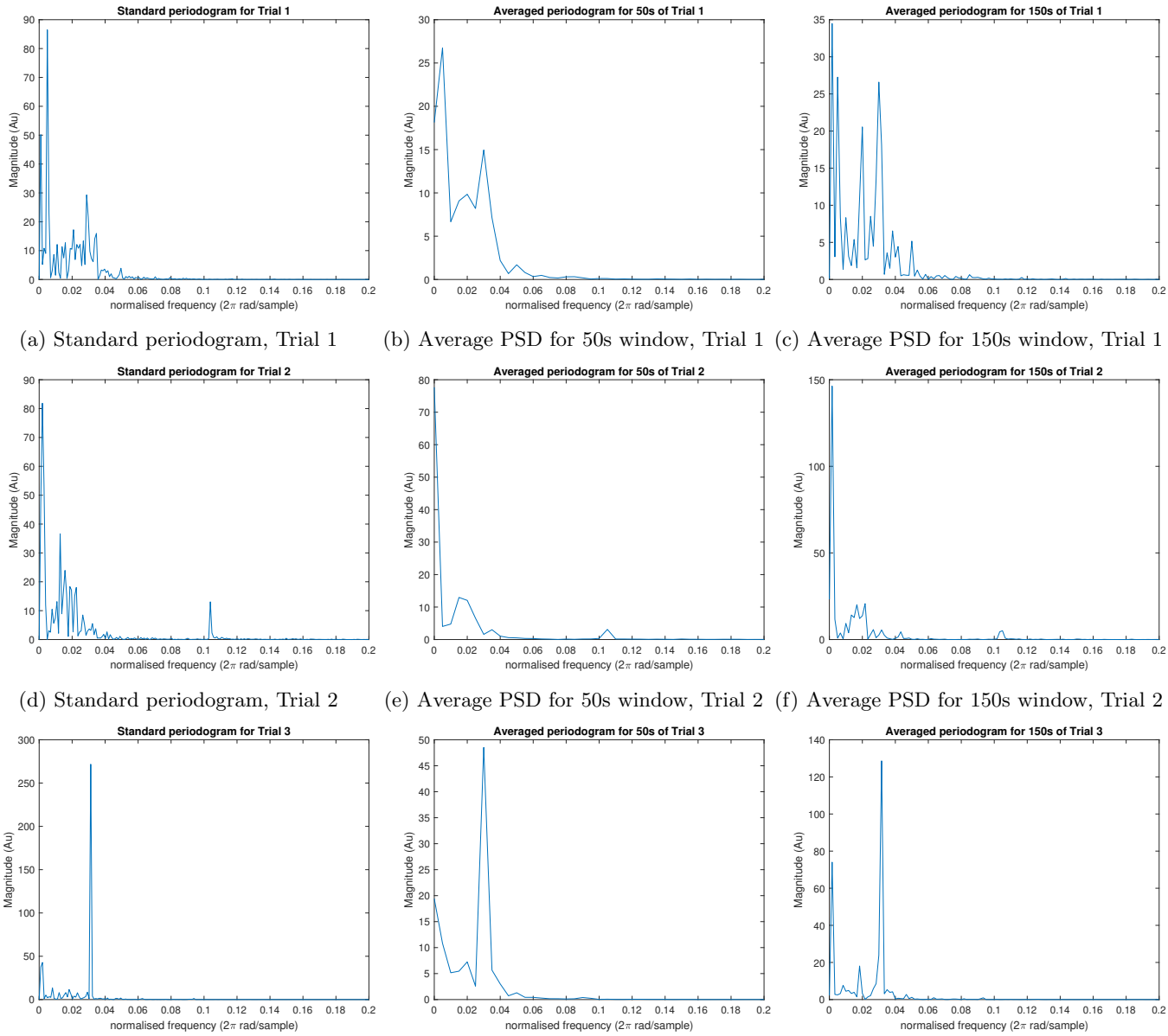
(d) Portion of the time signal, $\sigma = 10$ (e) Spectrogram, $\sigma = 10$ (f) Frequency spectrum, $\sigma = 10$

(g) Portion of the time signal, $\sigma = 25$ (h) Spectrogram, $\sigma = 25$ (i) Frequency spectrum, $\sigma = 25$

Figure 3.14: Analysis of the dial tone signal with added noise for three different variance cases

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

In Figure 3.15 are depicted the power spectral densities for the three trials discussed in Section 2.5. The dominant spectral components should represent the heart rate, therefore the lower the frequency of the peak, the lower the bpm, and vice versa for a high-frequency peak. In the first trial the subject was asked to breath normally, therefore we can use that as the baseline. In the second trial the airways were obstructed and we can see this reflected in the heart rate which is lower with respect to the first trial. On the other hand, in the third and final trial the subject was asked to breath more slowly and this is also reflected in a higher heart rate. Additionally we can see how in each trial there are two dominant harmonics. This could be explained by the fact that, according to the RSA phenomenon, the heart changes beating rate when inhaling and exhaling, therefore showing two different frequency peaks.



(a) Standard periodogram, Trial 1 (b) Average PSD for 50s window, Trial 1 (c) Average PSD for 150s window, Trial 1

(d) Standard periodogram, Trial 2 (e) Average PSD for 50s window, Trial 2 (f) Average PSD for 150s window, Trial 2

(g) Standard periodogram, Trial 3 (h) Average PSD for 50s window, Trial 3 (i) Average PSD for 150s window, Trial 3

Figure 3.15: Periodograms for the three RRI trials taken using the standard method and averaging for 50s and 150s

4 Optimal filtering - fixed and adaptive

4.1 Wiener filter

4.1.1

In this question we input WGN noise into a filter with given coefficients $b = [1\ 2\ 3\ 2\ 1]$. The resulting output, $\mathbf{y}$, is then normalized to have a variance of 1 and hence have the same power as the input. In order to simulate real-life conditions, noise is added to $\mathbf{y}$ with a variance of 0.1^2. Using this knowledge we can calculate the theoretical SNR as shown in Equation 4.1. This parameter can be also estimated experimentally, but its accuracy will depend on the number of samples used. For instance by using 1000-samples signals we get an estimated SNR of 19.9837 dB, but by using 10 million samples we get much closer value of 20.0013 dB.

$$SNR_{db} = \left(\frac{Power_y}{Power_\eta}\right)_{dB} = \left(\frac{\sigma_y^2}{\sigma_\eta^2}\right)_{dB} = 10\log_{10}\left(\frac{1}{0.1^2}\right) = 20\ dB \tag{4.1}$$

Using the statistics $\mathbf{R}_{xx}$ and $\mathbf{p}_{zx}$ we can find the optimum Wiener filter's coefficients to be those shown in Table 4.1. We can see that without normalizing $\mathbf{y}$, the optimisation yields coefficients very close to those of the unknown filter. Furthermore we can also see how by increasing the number of samples of the signal we will obtain a closer estimate. On the other hand normalizing the output of the filter does not allow for a good estimate of the coefficients. This is caused by the fact that the Wiener filter treats the normalized data as if it was the actual output of the unknown filter and hence optimises its coefficients accordingly, obtaining a scaled version of the ideal results.

Parameters	$\mathbf{w}_{opt1}$	$\mathbf{w}_{opt2}$	$\mathbf{w}_{opt3}$	$\mathbf{w}_{opt4}$	$\mathbf{w}_{opt5}$
Normalization	0.2217	0.4570	0.6839	0.4549	0.2259
No Normalization	0.9991	2.0054	3.0075	2.0103	1.0034
Normalization, 10^7 samples	1.0000	2.0000	3.0000	2.0000	1.0001

Table 4.1: Optimised Wiener coefficients for different parameters of the signal

4.1.2

Another parameter that affects the results is the variance of the added noise. From Table 4.2 we can notice how an increase in variance results in a decrease both in the accuracy of the estimates and in the SNR. This makes sense as increasing the variance of white noise ultimately increases its power and therefore its influence on the signal. The decrease in accuracy also gets reflected in the average error, which increases for higher variances. This behaviour is only caused by the fact that the realizations are finite. In fact if the signals were infinite, a change in variance would not affect the estimates, as the correlation of white noise is zero everywhere apart from at $lag = 0$.

Furthermore, increasing N_w will cause an overmodelling of the system. However since the unknown filter does not have additional coefficients, the estimated ones for the Wiener filter will be close to zero and hence their effect will be negligible. This being said, even though the error will roughly remain unchanged, overmodelling is to be avoided as it substantially increases the computational power needed and the time taken to obtain the results.

Noise Variance	$\mathbf{w}_{opt1}$	$\mathbf{w}_{opt2}$	$\mathbf{w}_{opt3}$	$\mathbf{w}_{opt4}$	$\mathbf{w}_{opt5}$	Average Error	SNR_{dB}
0.1	0.9982	2.0081	3.0081	1.9998	1.0010	0.0038	22.7921
2.08	1.0175	1.9591	3.0380	2.0131	0.9750	0.0269	10.0229
4.06	1.0711	1.8736	3.0849	2.0607	0.9740	0.0738	6.7594
6.04	0.8300	1.9484	3.0430	1.9756	1.1055	0.0789	5.3262
8.02	0.8553	1.9550	2.9366	1.9888	0.8556	0.0817	3.8286
10	1.0817	2.1021	3.1574	1.8972	0.8688	0.1151	2.4552

Table 4.2: Comparison of the effects of the noise's variance on the optimised Wiener coefficients

4.1.3

The disadvantage of this kind of optimisation is the computational power required. The $\mathbf{p}_{zx}$ matrix has $N_w + 1$ elements, where each one of those represents the expected value between $\mathbf{z}$ and $\mathbf{x}$ and hence requires N multiplications and additions. Therefore the latter matrix will require in total $N(N_w + 1)$ multiplications and just as many additions, which results in a complexity in the order of $\mathcal{O}\left(N(N_w + 1)\right)$. Moreover, even though $\mathbf{R}_{xx}$ is an $(N_w + 1)$x$(N_w + 1)$ matrix, thanks to the symmetry of the ACF with respect to $n = 0$, it will only require $N(N_w + 1)$ multiplications and additions, which results in the same complexity as $\mathbf{p}_{zx}$. However the $\mathbf{R}$ matrix has to be inverted and will therefore require a number of operations in the order of magnitude of $\mathcal{O}\left((N_w + 1)^3\right)$.

4.2 The least mean square (LMS) algorithm

4.2.1

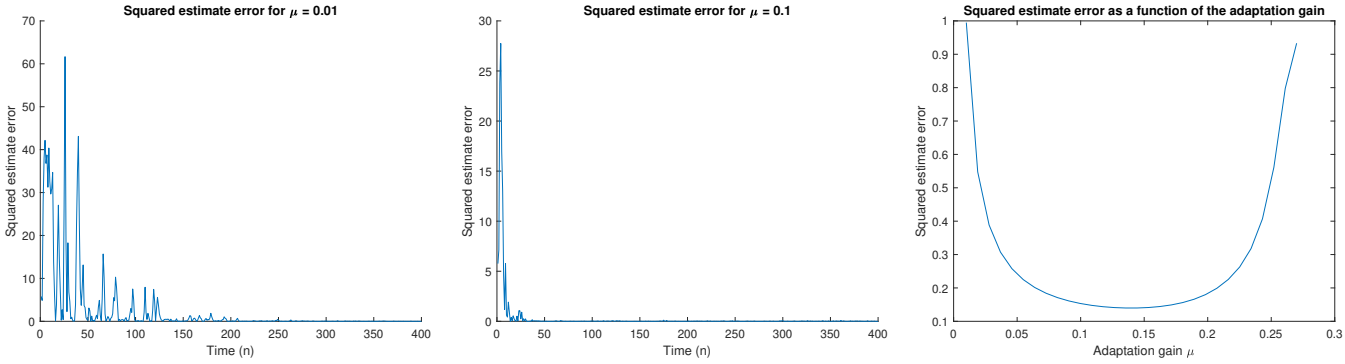
In this question I have used the LMS algorithm in order to find the optimised weights. The results are shown in Table 4.3, where the average error magnitude of the weights with respect to the actual coefficients can be compared between the LMS and Wiener optimisations. These two approaches are in fact very different. First of all, the signal has to be assumed stationary in order to use the Wiener filter, whereas the LMS method can be used in all circumstances. This is one of the strong advantages of the least mean square algorithm, since stationarity cannot always be guaranteed. However the LMS method is able to use only a small moving window of data for every iteration, while the Wiener filter takes advantage of the whole available data. For this reason, when it can be used, the Wiener filter yields more

accurate results, as it uses more data at the same time. This comparison between the two algorithms is reflected in the above mentioned Table, where the signal that has been processed was in fact stationary. Because of this, it can be noticed by the average errors that the Wiener filter registered a higher accuracy.

Noise Variance	LMS						Wiener
	$\mathbf{w}_{opt1}$	$\mathbf{w}_{opt2}$	$\mathbf{w}_{opt3}$	$\mathbf{w}_{opt4}$	$\mathbf{w}_{opt5}$	Average Error	Average Error
0.1	0.9794	1.9876	2.9836	1.9856	0.9955	0.0137	0.0038
4.06	0.9466	1.9168	2.9253	1.9415	0.8217	0.0896	0.0738
10	1.1935	1.7956	3.0761	2.3936	1.0618	0.1859	0.1151

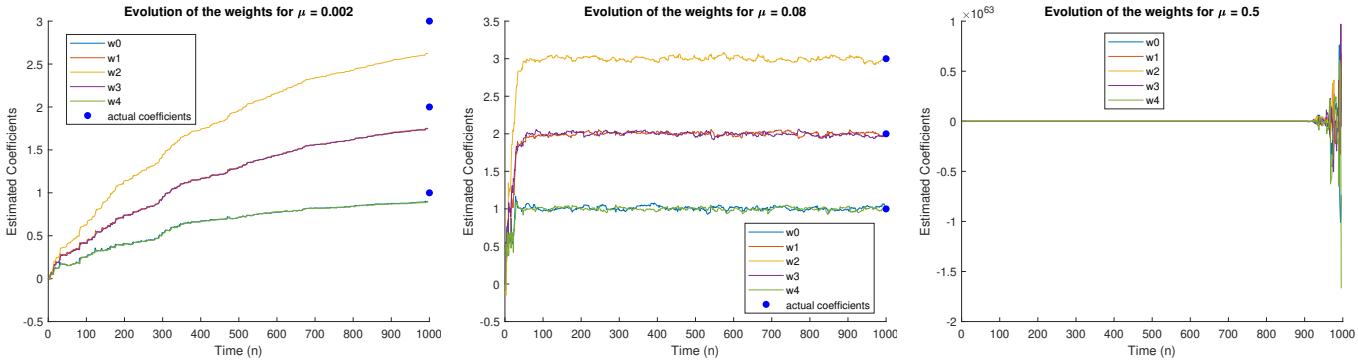
Table 4.3: Optimised LMS coefficients for different noise variances and the comparison between the errors for the LMS and Wiener algorithms

4.2.2



(a) (b) (c)

Figure 4.1: (a) Squared estimate error for $\mu = 0.01$. (b) Squared estimate error for $\mu = 0.1$. (c) Squared estimate error as a function of the adaptation gain μ.



(a) $\mu = 0.002$ (b) $\mu = 0.08$ (c) $\mu = 0.5$

Figure 4.2: Evolution of the optimal weights for different values of the adaptation gain μ

A very important parameter used in the LMS algorithm is the adaptation gain μ. The latter is also known as the learning step and represents by how much the update of the weights will be scaled. For instance, a small μ indicates that the weights will evolve in smaller steps, and hence more gradually, as shown in Figure 4.2a. Even though this may seem like a good strategy, as it is a controlled way of updating the coefficients, it may also cause the weights to never be able to converge with the amount of data points available, as it has happened in Figure 4.2a. A way around this obviously is to increase the gain, causing the weights to converge to the target values more quickly, as in Figure 4.2b. This can also be proven by comparing Figures 4.1a and 4.1b, which show the squared estimate error for two different gains. From these plots we can see that the larger learning step is able to reduce the error much faster than the lower one. However, increasing the learning step is not always beneficial, as it is shown in Figure 4.1c, where it is

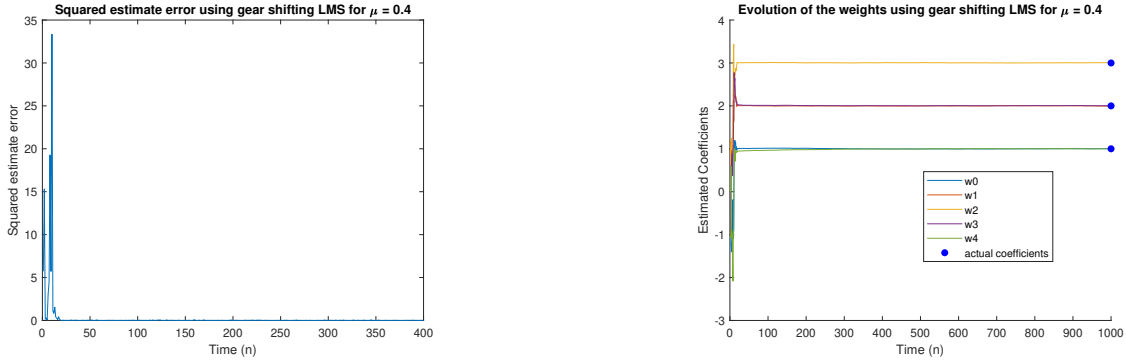
noticeable how increasing μ will decrease the error up t a certain point, but it will then start to increase it. In fact, an excessively large gain will cause very big movements in the weights' values, which could lead them to over-correct and overshoot the target. This ultimately means that, for very large μ, the weights will likely not converge, as shown in Figure 4.2c. Therefore, to sum up, smaller gains will achieve a gradual but slow convergence. Increasing the learning step will accelerate the updates, but when μ becomes too large it can cause to overshoot the target values and hence increase the error at steady state or even prevent convergence. There is therefore a critical value of μ that should be found in order to optimise the filter's adaptation.

4.2.3

The complexity of this algorithm can be determined by analysing one iteration. The first expression is $\mathbf{w}(n+1) = \mathbf{w}(n) + \mu e[n]\mathbf{x}(n)$ and it involves $N_w + 1$ additions, which is a number of operations in the order of $\mathcal{O}(N_w + 1)$. The second calculation estimates the signal as $\hat{y}[n] = \mathbf{w}^T(n)\mathbf{x}(n)$. It utilises $N_w + 1$ additions and multiplications, hence resulting in an order of complexity of $\mathcal{O}(2(N_w + 1)) \approx \mathcal{O}(N_w + 1)$. Lastly the calculation of the error is a single subtraction and it therefore does not impact the complexity of the LMS method. By considering these calculations together we find that the number of operations required for one iteration is roughly in the order of $\mathcal{O}(N_w + 1)$. However, it needs to be kept in mind that for an N-samples signal these calculations will be repeated for $N - N_w$ times, which is a factor to consider when working with very long recorded signals.

4.3 Gear shifting

The choice of adaptation gain, as we have discussed above, is crucial for a good optimisation. For this reason a common technique is gear shifting, where the learning step is progressively changed as the weights get updated. The goal, in this case, is to choose a high initial gain in order to have a fast convergence and then decrease it to have better precision at steady state. For this purpose I have changed the standard LMS algorithm to also calculate the mean squared error of the last five iterations. The first mse calculated is used as reference error. When the squared error becomes lower than a certain fraction of the reference error, the gain will be decreased by a certain percentage and the current mse will be used as the new reference. Both the fraction of the reference and the percentage decrease of the gain are parameters that can be changed. By choosing the right value of these parameters this algorithm is able to achieve very good results.

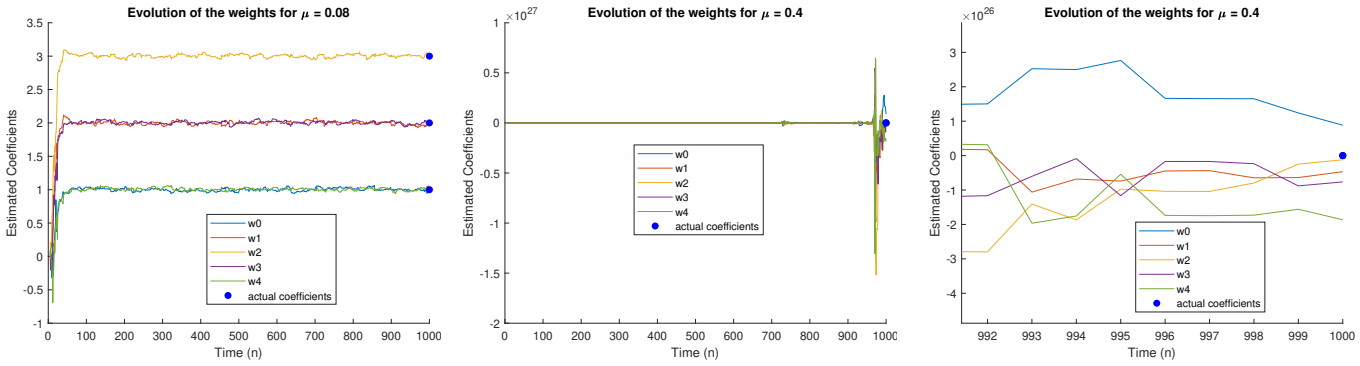


(a) Squared estimate error as a function of time

(b) Evolution of the optimal weights

Figure 4.3: Weights optimisation using the LMS algorithm with updating learning step for $\mu_{initial} = 0.4$

From Figure 4.4b and 4.4c we can see that the standard LMS algorithm did not converge for high gains. However, using gear shifting allows us to set a high initial gain and take advantage of its fast convergence, while exploiting the diminishing learning step to have higher accuracy in the steady state. This can be seen in Figure 4.3b where we can see that convergence happens very quickly, but the steady state has more gradual updates. By comparing this result with Figure 4.4a we can see how gear shifting produces better results both in the convergence and the steady state, clearly implying an adaptation of the gain itself. Additionally the improvement can also be noticed by looking at the error produced by the gear shifting LMS shown in Figure 4.3a, which converges to zero even before sample 25.



(a) $\mu = 0.08$ (b) $\mu = 0.4$ (c) zoomed in for $\mu = 0.4$

Figure 4.4: Evolution of the optimal weights using the standard LMS algorithm

4.4 Identification of AR processes

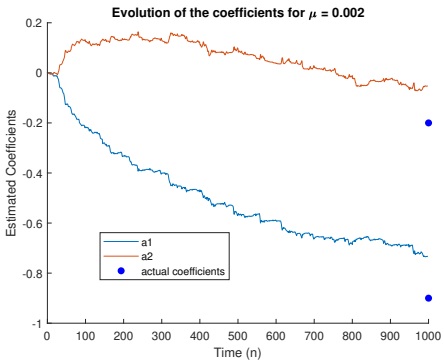
4.4.1

The equation of an AR model of order two is shown in Equation 4.2. However, `MATLAB` represents all 'a' coefficients being on the left hand side of the equation. For this reason, when modelling the coefficients of a filter with $a = [1\ 0.9\ 0.2]$ in `MATLAB` notation, we expect to estimate $a_1 = -0.9$ and $a_2 = -0.2$, as the adaptation algorithm follows Equation 4.2.

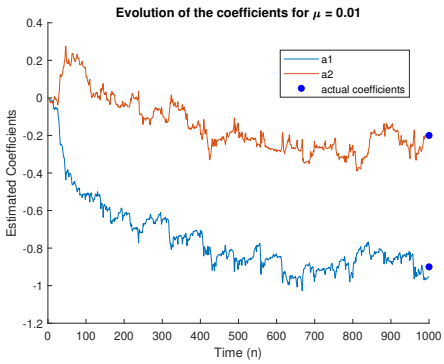
$$x[n] = a_1 x[n-1] + a_2 x[n-2] + w[n], \qquad w[n]\ \mathcal{N}(0,1) \tag{4.2}$$

4.4.2

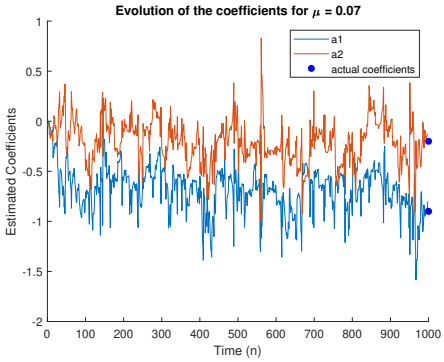
For this question I have implemented the structure in Figure 5 of the coursework. I have inputted WGN noise into a filter with coefficients $a_1 = -0.9$ and $a_2 = -0.2$ and then analysed the resulting output using the LMS algorithm to estimate the values of a_1 and a_2. Figure 4.5 shows the evolution of the estimated coefficients for different values of the adaptation gain. As we have seen for Section 4.2, there is a certain value of μ that yields the best results. For values of the gain that are too low, the update of the estimates is too slow and for a signal of 1000 samples it rarely converges, as shown in Figure 4.5a. On the other hand, when μ is too much above the optimal value, the learning step will be too large and cause the estimates to drastically overshoot, often preventing convergence, as shown in Figure 4.5d. Instead, by looking at Figure 4.5b, we can see how values of the gin close to he optimal one will indeed allow the estimates to converge to the desired values. Another interesting example is shown in Figure 4.5c, where we can see how values of μ that are slightly larger than the desired one will not diverge as seen for $\mu = 0.2$, but will still oscillate around the true value and will usually take more time to converge to a steady state.



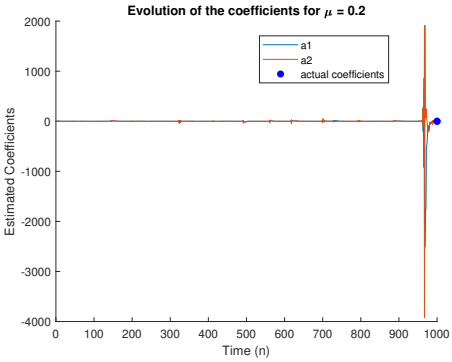
(a) $\mu = 0.002$



(b) $\mu = 0.01$



(c) $\mu = 0.07$



(d) $\mu = 0.2$

Figure 4.5: Evolution of the estimated coefficients

4.5 Identification of AR processes

4.5.1

For this section I have used the prediction model explored in the previous question. I recorded myself saying five letters, I then delayed the recordings and exploited the structure from Figure 5 of the coursework to find the coefficients to apply to past samples in order to predict the next one. I later used those found coefficients to predict how the original signal would turn out from the delayed version. In order to obtain the best results I investigated which adaptation gain and order would be most suitable for each letter's model. The results for the filter's length can be found in Table 4.4 and the optimal orders were selected using the MDL criterion. However the letter 'E' turned out to be best for order 33, which is unnecessarily high. For this reason I have decided to use the corrected AIC criterion for the latter letter. In fact, even though I am aware that the AIC_c is intended for short recordings only, also the MDL criterion, when the maximum order was set to 15, considered order one to be the best fit for the letter 'E'. The other important parameter to find is the adaptation gain. The results can be found in Table 4.5, where we can see how the different letters, having substantially different waveforms, necessitate of different learning rates.

It is also worth mentioning how short segments of speech recordings can be assumed to be stationary, as the statistics vary slowly. For this reason, gear shifting is a valid option to use in these cases, and should improve the performance of the predictors.

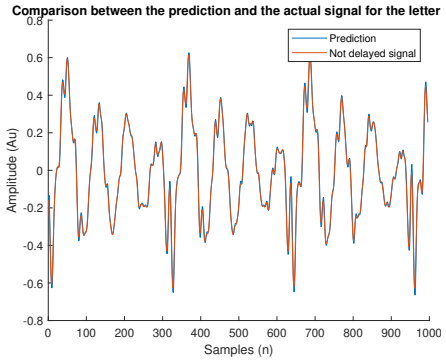
Loss Criterion	Letters				
	A	E	S	T	X
MDL	4	33	4	1	1
Corrected AIC	4	1	4	1	1
AIC	4	33	4	29	1

Table 4.4: Filter's order that yielded the lowest error for each letter following different criteria. The filtered signal has 1000 samples, taken at $44.1kHz$

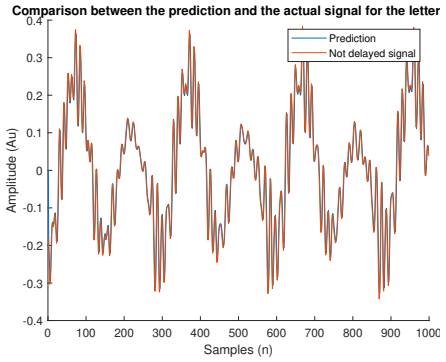
Adaptation Gain	Letters				
	A	E	S	T	X
0.2634	0.001134	0.000905	0.000931	0.000608	0.0003881
0.7074	0.000566	0.000916	0.000324	0.000116	0.0002594
0.8385	0.000582	0.000916	0.000290	0.000104	0.0002592
1	0.000637	0.000930	0.000267	0.000099	0.0002593

Table 4.5: Average squared error for different adaptation gains. In green are the μ values that yielded the best results when combined with the orders found in Table 4.4. The filtered signal has 1000 samples, taken at $44.1kHz$

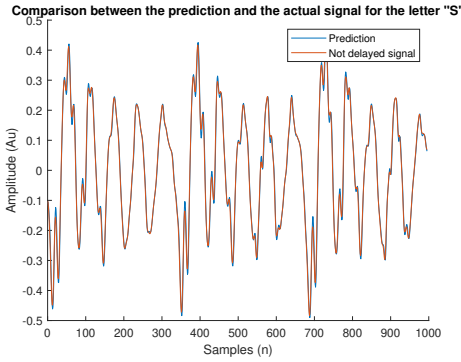
The optimal orders and gains reported in the tables above have been used in the adaptation filter. Some of the predictions obtained can be seen in Figure 4.6, where we are able to appreciate how, by using the right learning parameters, it is possible to get very promising predictions. Furthermore, from the plots for letters 'E' and 'T' it becomes understandable why the MDL and AIC resulted in higher optimal orders with respect to the other letters, as 'E' and 'T' appear to be more complex and have a higher number of harmonics.



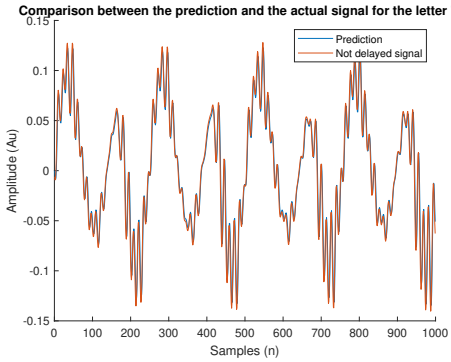
(a) letter 'A', order 4, gain 0.7074



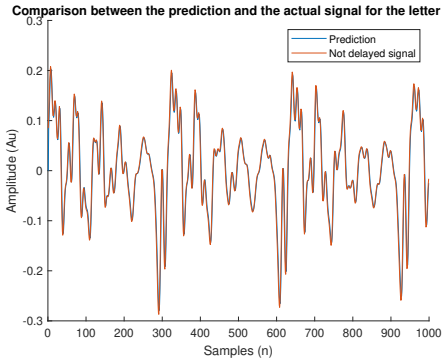
(b) letter 'E', order 1, gain 0.2634



(c) letter 'S', order 4, gain 1



(d) letter 'T', order 1, gain 1



(e) letter 'X', order 1, gain 0.8385

Figure 4.6: Predictions of the various letter's recordings against the original recordings

4.5.2

When dealing with a predictor the choice of filter's length is fundamental. As it has been done in section 4.5.1, the calculation of the cumulative squared error remains a valid option. The results can then be analysed using various criteria, with the most reliable possibly being the MDL. These criteria apply an extra factor of error as the order increases, penalising overmodelling. This last point is crucial when trying to achieve a fast and less computationally demanding algorithm.

Additionally, when dealing with a predictor, a good measure of its performance is the prediction gain R_p. It is then possible to compare this parameter for different orders to understand which one suits better the predictor. This process has been done for the five recordings and the results are shown in Table 4.6. From these R_p values we can see how the optimal orders match the ones found in Table 4.4, with he exception for the letter 'E' and 'T'. However the results for both these letters still make sense. The letter 'E' has in fact shown the highest performance for the order 33, which agrees with the MDL, while the letter 'T' performs best at order 29, as it was also proposed using the AIC

criteria. However since these orders require more computational power, if we discard the last two rows of Table 4.6 we can see that the orders with the best performances actually perfectly agree with those found in Table 4.4.

Order	Letters				
	A	E	S	T	X
1	17.2860	14.3541	17.8732	15.7795	14.7488
4	20.4751	12.6989	21.5453	11.5450	13.9007
10	17.3205	13.1208	16.2985	13.7387	13.3367
29	< 1	14.5715	< 1	15.8849	10.9908
33	< 1	16.8061	< 1	14.7683	9.7408

Table 4.6: Prediction gain R_p for different orders and letters. The filtered signal has 1000 samples, taken at $44.1kHz$ and the predictor used the optimal gains from Table 4.5

4.5.3

Further analysis regards changing the sampling frequency of the recordings from 44.1 *kHz* to 16 *kHz*. By doing so the signal's resolution decreases, since, for instance, at 44100 *Hz* 1000 samples represented roughly three periods, while at 16 *kHz* they represent approximately nine. This shows how lowering the sampling frequency compresses the signal, therefore the same 1000 samples represent more of the recording and for this reason will be less stationary. In fact, speech signals can be assumed stationary when considered for small intervals of time, and hence their properties are likely to remain unchanged. To sum up, a lower sampling frequency will decrease the resolution of the signal, and in turn also its stationarity. For these reasons the predictor using 16 *kHz* will be less trivial to train and will produce worse results, as it can be shown by the first two rows of Table 4.7. The higher complexity of the training and its lower performance are also visible in Table 4.8 where it is noticeable how the filters re longer compared to Table 4.4. This makes sense as the predictor will need more past samples in order to be able to predict the future signal. Furthermore, in the last row of Table 4.7 it is shown how increasing the number of samples considered can help compensate for the deceased signal resolution and will help the predictor's performance. Therefore, lowering the sampling frequency will facilitate the recording process, but will require more samples to achieve the same predictor performance of a higher sampling frequency. Hence, there is a trade off between a lower sampling frequency, but a higher computational effort in the training, and vice versa.

Sampling Frequency	Letters				
	A	E	S	T	X
44100 *Hz*, 1000 samples	20.4751	14.3541	21.5453	15.7795	14.7488
16000 *Hz*, 1000 samples	13.7280	16.3794	12.6148	12.1624	9.7750
16000 *Hz*, 6000 samples	16.1175	18.0542	Na	15.2637	Na

Table 4.7: Prediction gain of the various letters, filtered with their respective optimal gains and orders for different signal lengths and sampling frequencies

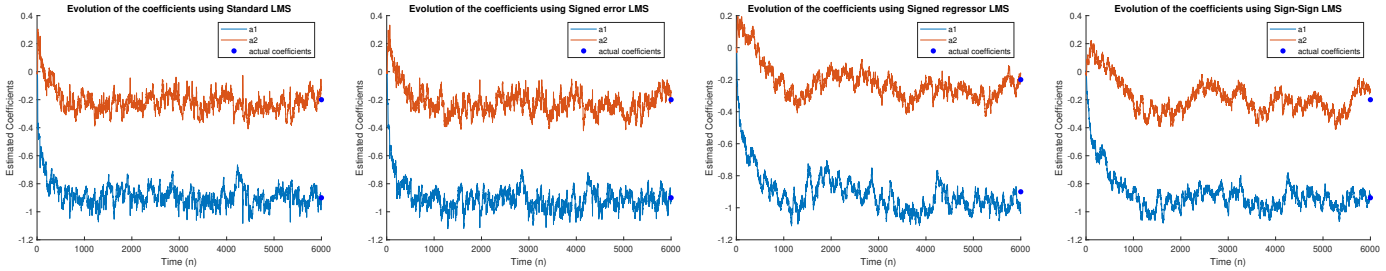
Loss Criterion	Letters				
	A	E	S	T	X
MDL	16	15	2	11	10
AIC	16	15	2	11	13

Table 4.8: Filter's order that yielded the lowest error for each letter following different criteria. The filtered signal has 1000 samples, taken at $16kHz$

4.6 Dealing with computational complexity: sign algorithms

A simplified version of the LMS algorithm is the class of the sign LMS algorithms. To better analyse this group, it has been decided to compare the various algorithms using the same signal family as in section 4.4. The results for the evolution of the estimated coefficients are shown in Figure 4.7. From here we can notice how the signed error is actually similar to the standard LMS. In the particular case shown in the figure it manages to converge slightly faster than the standard, but it is not always the case. On the other hand the other two signed algorithms show a slower convergence and a noticeably higher variance at steady state. Further analysing the steady state, we can understand that the simplified algorithms are substantially less accurate than the standard LMS. In fact, if we average the steady

state variance for 400 different noise realisations we can see that the results are 0.111, 0.1189, 0.1218 and 0.1253 for the standard, signed error, signed regressor and sign-sign LMS algorithms respectively. These results make sense as the simplified class depends on the sign of the error or/and the signal and not their absolute value. This means that, for instance, for the signed error algorithm an error of 0.0001 and an error of 100 will produce the same movement, determined by the adaptation gain. Therefore the coefficients will not fully converge unless an error of 0 is met. For this reason I assume that gear shifting combined with the signed algorithms would drastically improve the obtained results, but would go against their main goal of being a simplified version of the LMS.

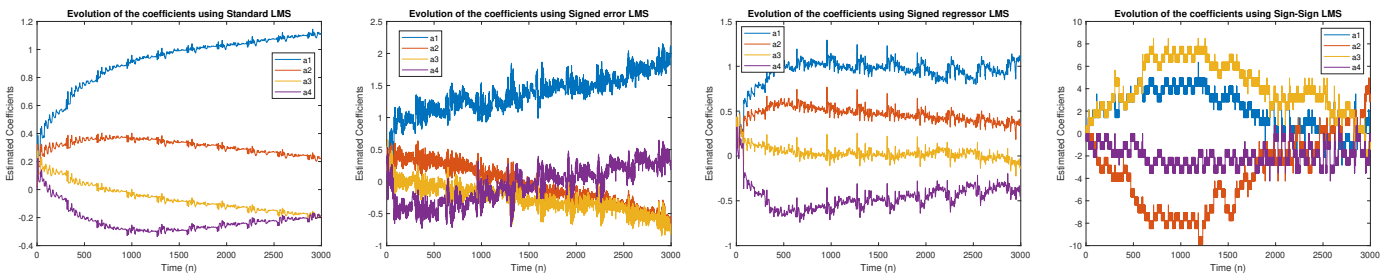


(a) Standard LMS (b) Signed error LMS (c) Signed regressor LMS (d) Sign-Sign LMS

Figure 4.7: Evolution of the estimated coefficients for different optimisation algorithms, using $\mu = 0.01$

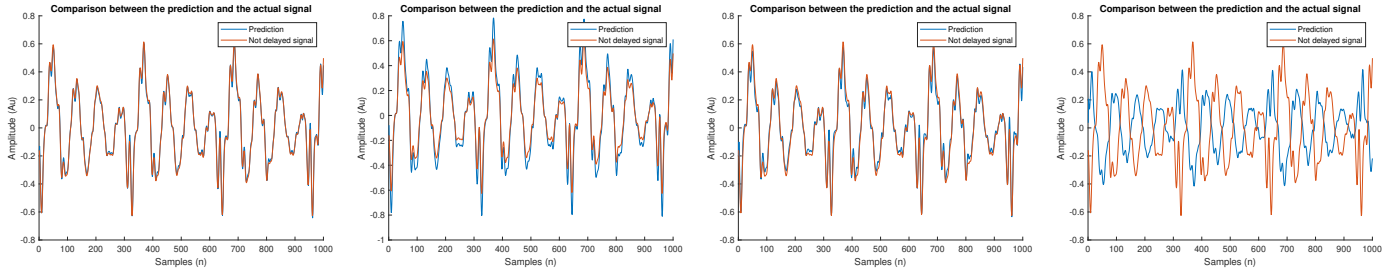
The signed algorithms have also been applied to the letter 'A' recording encountered in section 4.5. We can see the evolution of the estimated coefficients in Figure 4.8. From the results shown, we can see how, apart from the standard LMS, the only other algorithm that seems to converge is the signed regressor. The other two algorithms show a gradient in the evolution that does not appear to go to zero. This actually makes sense as the signed regressor is the only algorithm of the three that considers the error magnitude and hence should update less violently as the error goes to zero. On the other hand, the other two algorithms do not consider the magnitude of the error and will therefore evolve in the same way regardless of whether the error is very largo or close to zero. This behaviour increases the convergence time and could ultimately prevent the estimates to settle to their values. This can be seen especially for the sign-sign algorithm, where after approximately sample 1500 the estimates, where the coefficients had appeared to have converged, began to drastically change again, highlighting the fact that this algorithm is highly variable and inaccurate.

The effects of these evolution plots can be seen reflected in the resulting predictions, shown in Figure 4.9. It is clear how the best prediction is achieved by the standard LMS, with the signed regressor showing a similar result, as it could have been guessed by their resembling coefficient evolutions. Instead, the other two algorithms show a lower accuracy. These analyses are reflected in the average squared errors of the prediction for the different algorithms, which turned out to be 0.0005, 0.0043, 0.0012 and 0.1495 for the standard, signed error, signed regressor and sign-sign LMS algorithms respectively. The highest error was obtained using the sign-sign algorithm, which in fact was the only one to estimate the $a_1 < 0$. This is reflected in Figure 4.9d where we can see how the prediction is flipped with respect to the actual signal. Additionally, also the reason for the prediction being higher than the actual signal in Figure 4.9b can be explained from the coefficient evolution. In fact from Figure 4.8b we can see how the signed error estimated a_1 to be higher than two, which is roughly double the estimate of the standard LMS, resulting is a higher prediction. Therefore to sum up, as before the standard LMS produced the best results. However the signed regressor, even though it has a higher variance at steady state, still managed to produce a prediction very close to that of the standard, therefore showing that, for the letter 'A', it is a valid substitute algorithm.



(a) Standard LMS (b) Signed error LMS (c) Signed regressor LMS (d) Sign-Sign LMS

Figure 4.8: Evolution of the estimated coefficients for different optimisation algorithms, using $\mu = 0.01$



(a) Standard LMS (b) Signed error LMS (c) Signed regressor LMS (d) Sign-Sign LMS

Figure 4.9: The original signal for the letter 'A' against its prediction using the filter's coefficients shown in Figure 4.8

5 MLE for the Frequency of a Signal

5.1

The maximum-likelihood estimate (MLE) of the frequency f_0 is found by minimizing the cost function in Equation 5.1, where the difference between the signal $\mathbf{x}$ and the cosine has been described as the error.

$$\begin{aligned} J(\boldsymbol{\theta}) &= \sum_{n=0}^{N-1} \left(x[n] - A\cos\left(2\pi f_0 n + \phi\right)\right)^2 \\ &= \sum_{n=0}^{N-1} e[n]^2 \end{aligned} \tag{5.1}$$

In order to better understand the above expression it is necessary to simplify the error. For this purpose we let $\alpha_1 = A\cos\phi$ and $\alpha_2 = A\sin\phi$ and define $c = [1, \cos(2\pi f_0), \ldots, \cos(2\pi f_0(N-1))]^T$ and $s = [0, \sin(2\pi f_0), \ldots, \sin(2\pi f_0(N-1))]^T$. Additionally another useful property is the trigonometric identity $\cos(A+B) = \cos A\cos B - \sin A\sin B$.

$$\begin{aligned} \mathbf{e} &= \begin{bmatrix} x[0] - A\cos(\phi) \\ x[1] - A\cos(2\pi f_0 + \phi) \\ x[2] - A\cos(4\pi f_0 + \phi) \\ \vdots \\ x[N-1] - A\cos(2\pi f_0(N-1) + \phi) \end{bmatrix} = \begin{bmatrix} x[0] - A\cos(\phi) \\ x[1] - A\cos(\phi)\cos\left(2\pi f_0\right) - A\sin(\phi)\sin\left(2\pi f_0\right) \\ x[2] - A\cos(\phi)\cos\left(4\pi f_0\right) - A\sin(\phi)\sin\left(4\pi f_0\right) \\ \vdots \\ x[N-1] - A\cos(\phi)\cos\left(2\pi f_0(N-1)\right) - A\sin(\phi)\sin\left(2\pi f_0(N-1)\right) \end{bmatrix} \\ &= \mathbf{x} - \alpha_1\mathbf{c} - \alpha_2\mathbf{s} \\ &= \mathbf{x} - \mathbf{H}\boldsymbol{\alpha} \end{aligned} \tag{5.2}$$

Where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2]^T$ and $\mathbf{H} = [\mathbf{c}, \mathbf{s}]$. It is then possible to use Equation 5.2 to simplify 5.1 as shown below.

$$\begin{aligned} J(\boldsymbol{\theta}) &= \sum_{n=0}^{N-1} e[n]^2 \\ &= \mathbf{e}^T\mathbf{e} \\ &= \left(\mathbf{x} - \mathbf{H}\boldsymbol{\alpha}\right)^T\left(\mathbf{x} - \mathbf{H}\boldsymbol{\alpha}\right) \\ &= J'(\boldsymbol{\alpha}, \mathbf{f}_0) \end{aligned} \tag{5.3}$$

5.2

Equation 5.3 can be further solved to $J'(\boldsymbol{\alpha}, \mathbf{f}_0) = \mathbf{x}^T\mathbf{x} - 2\mathbf{x}^T\mathbf{H}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\mathbf{H}^T\mathbf{H}\boldsymbol{\alpha}$. In order to find the minimizing solution $\hat{\boldsymbol{\alpha}}$ we have to take the derivative of $J'(\boldsymbol{\alpha}, \mathbf{f}_0)$ with respect to $\boldsymbol{\alpha}$. Equations 5.4 and 5.5 will be used to help with the calculations.

$$\frac{\partial \mathbf{x}^T\mathbf{H}\boldsymbol{\theta}}{\partial \boldsymbol{\theta}} = \mathbf{H}^T\mathbf{x} \tag{5.4}$$

$$\frac{\partial \boldsymbol{\theta}^T\mathbf{H}^T\mathbf{H}\boldsymbol{\theta}}{\partial \boldsymbol{\theta}} = 2\mathbf{H}^T\mathbf{H}\boldsymbol{\theta} \tag{5.5}$$

Using these expressions we are able to determine the derivative, as shown in Equation 5.6. It will then be possible to set the latter equal to zero in order to find the value of $\boldsymbol{\alpha}$ that minimizes the cost function, which is found in Equation 5.7.

$$\begin{aligned}\frac{\partial J'(\boldsymbol{\alpha}, \mathbf{f}_0)}{\partial \boldsymbol{\alpha}} &= \frac{\partial}{\partial \boldsymbol{\alpha}} \left(\mathbf{x}^T\mathbf{x} - 2\mathbf{x}^T\mathbf{H}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\mathbf{H}^T\mathbf{H}\boldsymbol{\alpha} \right) \\ &= 2\mathbf{H}^T\mathbf{H}\boldsymbol{\alpha} - 2\mathbf{H}^T\mathbf{x} \end{aligned} \tag{5.6}$$

$$\begin{aligned} 2\mathbf{H}^T\mathbf{H}\hat{\boldsymbol{\alpha}} - 2\mathbf{H}^T\mathbf{x} &= 0 \\ \hat{\boldsymbol{\alpha}} &= (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \end{aligned} \tag{5.7}$$

We can then use the found minimizing value of $\boldsymbol{\alpha}$ to have $J'(\boldsymbol{\alpha}, \mathbf{f}_0)$ only dependent on $\mathbf{H}$ and $\mathbf{x}$. In order to do that we can plug in $\hat{\boldsymbol{\alpha}}$ in Equation 5.3 and take advantage of the fact that $\mathbf{I} - \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T$ is an idempotent matrix. The result is shown in Equation 5.8.

$$\begin{aligned} J'(\hat{\boldsymbol{\alpha}}, \mathbf{f}_0) &= (\mathbf{x} - \mathbf{H}\hat{\boldsymbol{\alpha}})^T (\mathbf{x} - \mathbf{H}\hat{\boldsymbol{\alpha}}) \\ &= \left(\mathbf{x} - \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x}\right)^T \left(\mathbf{x} - \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x}\right) \\ &= \mathbf{x}^T(\mathbf{I} - \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T)\mathbf{x} \end{aligned} \tag{5.8}$$

Now that we have the final expression for J', in order to find the estimated frequency $\hat{f}_0$ we need to minimize Equation 5.8 for f_0. However, from Equation 5.9 we can see that, since $\mathbf{x}$ is fixed, minimizing J' is equivalent to maximizing $\mathbf{x}^T\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x}$, which is the likelihood function.

$$\begin{aligned} J'(\hat{\boldsymbol{\alpha}}, \mathbf{f}_0) &= \mathbf{x}^T(\mathbf{I} - \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T)\mathbf{x} \\ &= \mathbf{x}^T\mathbf{x} - \mathbf{x}^T\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x} \end{aligned} \tag{5.9}$$

5.3

Using the definition of $\mathbf{H}$, as $\mathbf{H} = [\mathbf{c}, \mathbf{s}]$, we can further express $\mathbf{x}^T\mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{x}$ as shown below. Therefore, the MLE for the frequency can be found by maximizing Equation 5.10.

$$\begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix}^T \begin{bmatrix} \mathbf{c}^T\mathbf{c} & \mathbf{c}^T\mathbf{s} \\ \mathbf{s}^T\mathbf{c} & \mathbf{s}^T\mathbf{s} \end{bmatrix} \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix} \tag{5.10}$$

5.4

Under the condition that f_0 is not close to 0 or 0.5, we can approximate the terms of the matrix above as in Equations 5.11, 5.12 and 5.13.

$$\begin{aligned} \mathbf{c}^T\mathbf{s} = \mathbf{s}^T\mathbf{c} &= \sum_{n=0}^{N-1} \cos(2\pi f_0 n)\sin(2\pi f_0 n) \\ &= \frac{1}{2}\sum_{n=0}^{N-1} \sin(4\pi f_0 n) \\ &\approx 0 \end{aligned} \tag{5.11}$$

$$\begin{aligned} \mathbf{c}^T\mathbf{c} &= \sum_{n=0}^{N-1} \cos(2\pi f_0 n)\cos(2\pi f_0 n) \\ &= \sum_{n=0}^{N-1} \cos^2(2\pi f_0 n) \\ &\approx \frac{N}{2} \end{aligned} \tag{5.12}$$

$$\begin{aligned}\mathbf{s}^T\mathbf{s} &= \sum_{n=0}^{N-1} \sin(2\pi f_0 n)\sin(2\pi f_0 n) \\ &= \sum_{n=0}^{N-1} \sin^2(4\pi f_0 n) \\ &\approx \frac{N}{2} \end{aligned} \tag{5.13}$$

When f_0 is not near 0 or 0.5 we can hence use the above expressions to approximate Equation 5.10 to be Equation 5.14. The latter can be further simplified to be Equation 5.15.

$$\begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix}^T \begin{bmatrix} \mathbf{c}^T\mathbf{c} & \mathbf{c}^T\mathbf{s} \\ \mathbf{s}^T\mathbf{c} & \mathbf{s}^T\mathbf{s} \end{bmatrix} \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix} \approx \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix}^T \begin{bmatrix} \frac{N}{2} & 0 \\ 0 & \frac{N}{2} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{c}^T\mathbf{x} \\ \mathbf{s}^T\mathbf{x} \end{bmatrix} \tag{5.14}$$

$$\begin{aligned} &= \frac{2}{N}\left[\left(\mathbf{c}^T\mathbf{x}\right)^2 + \left(\mathbf{s}^T\mathbf{x}\right)^2\right] \\ &= \frac{2}{N}\left[\left(\sum_{n=0}^{N-1} x[n]\cos 2\pi f_0 n\right)^2 + \left(\sum_{n=0}^{N-1} x[n]\sin 2\pi f_0 n\right)^2\right] \\ &= \frac{2}{N}\left|\sum_{n=0}^{N-1} x[n]\exp\left(-j2\pi f_0 n\right)\right|^2 \end{aligned} \tag{5.15}$$

However, by comparing Equation 5.15 with the periodogram shown below, we can see that the two differ by simply a constant. For this reason the MLE for the frequency can be obtained by directly maximizing $\hat{P}_X(f)$.

$$\hat{P}_X(f) = \frac{1}{N}\left|\sum_{n=0}^{N-1} x[n]e^{-j2\pi f\frac{n}{N}}\right|^2 \tag{5.16}$$