Estimation of frequency and phase of sinusoidal signal with noise

Petter Ølberg & Paulus K.Hæreid

05.04.2022

Abstract

Estimations for the frequency ω and phase ϕ of a discrete complex sinusoidal signal afflicted with noise have been performed. The estimates, $\hat{\omega}$ and $\hat{\phi}$, are estimated numerous times using FFT and the maximum likelihood estimator for different values of Signal-to-noise ratio and FFT size. The errors between the estimate and the original value, as well as the variances of these errors, are calculated in order to compare the variances with the Cramer Raos Lower Bound to get a qualitative assessment of the performance of this estimator. Additionally, the frequency is estimated using another method which is based on fine tuning an estimate of a relatively short FFT size using a numerical search. This estimator is more feasible in practice, compared to using large FFT sizes for estimation.

Contents

1 Introduction

The sinusoidal wave is a mathematical function used in many technological areas. In cases where a signal, such like the sinusoidal wave, gets transmitted across a medium, the signal often gets distorted. This is often due to noise, which comes in many different forms. Equation 1 express a continuous analogue signal $x(t)$ consisting of a complex sinusoidal wave with added *White Gaussian Noise* [1].

$$x(t) = Ae^{i(\omega_0 t+\phi)} + w(t) \tag{1}$$

In 1, A is the amplitude, ω_0 is the frequency and ϕ is the displacement of the wave. Similarly with the sinusoidal wave, the added noise $w(t)$ is complex. The expression for the analogue WGN is given in (2) [1].

$$w(t) = w_{\mathrm{r}}(t) + iw_{\mathrm{i}}(t) \tag{2}$$

When transmitting a signal such as (1) to a receiver, the signal will get *sampled* to be expressed as a digital signal $x[n]$. The expression of $x[n]$ in (3) is derived from the analogue signal by substituting t with nT_S to divide the signal into sampling intervals. In (3), n represents the sample number and T_S is the sampling period.

$$\begin{aligned} x[n] &= x(nT_S) \\ &= Ae^{i(\omega_0 nT_S+\phi)} + w(nT_S) \\ &= Ae^{i(\omega_0 nT_S+\phi)} + w[n] \\ &= s[n] + w[n] \end{aligned} \tag{3}$$

In many cases, the parameters A, ω_0 and ϕ get manipulated because of distortion of the signal. In such cases the parameters need to be estimated. In this rapport, two methods will be used in the estimation process of the frequency estimator $\hat{\omega}$ and displacement estimator $\hat{\phi}$. For simplicity reasons, the amplitude of the sinusoidal wave, A, will be assumed known. Figure 1 illustrates a block scheme om the estimation process of $\hat{\omega}$ and $\hat{\phi}$.

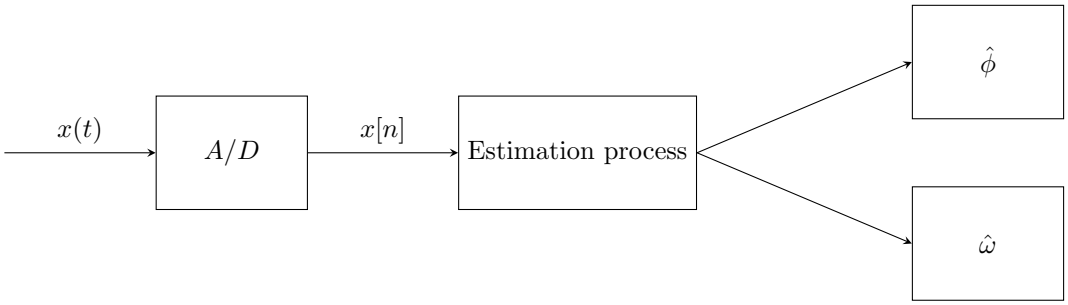

Figure 1: A block diagram of the estimation process of $\hat{\omega}$ and $\hat{\phi}$.

2 Theory

In this section, relevant theory for implementing the estimation process is given. Most of the theory is taken from the project description in [1], which has been further investigated through numerous sources on the internet. Some equations are taken from the courses in TTT4275, more specificly expression (12-15).

2.1 Fast Fourier Transform (FFT)

To perform frequency analysis on a discrete-time signal $x[n]$, we convert the time-domain sequence to an equivalent frequency-domain representation [2]. To do so we use the Discrete Fourier Transform on the sequence $x[n]$ as expressed in (4) to find $X(\omega)$[1].

$$X(\omega_0) = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-j\omega n T} \tag{4}$$

The Discrete Fourier Transform (DFT) is useful in many fields, but computing it directly from the definition is often too slow to be practical. An FFT rapidly computes such transformations by reducing the complexity of computing the DFT from $\mathrm{O}(N^2)$ to $\mathrm{O}(N \log_2 N)$, where N is the data size [3].

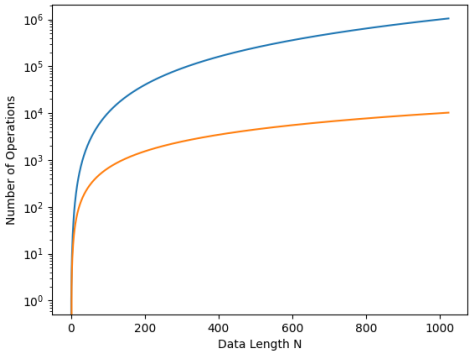

Figure 2: $\mathrm{O}(N^2)$ and $\mathrm{O}(N \log_2 N)$ represented in a logarithmic scale where $N \in [0, 2^{10}]$

The FFT will have a number of frequency bins M, which is not necessarily the same amount of bins as in the discrete input signal $x[n]$. Each bin represents the amount of energy that the signal has at that particular frequency. By looking at the frequency interval between every bin, one can determine the *frequency resolution* $\Delta\omega$ given in (5) where T_S is the sampling period [4].

$$\Delta\omega = 2\pi \frac{1}{MT_S} \tag{5}$$

If the frequency resolution is too low, there will be a chance of distorting the true frequency spectrum. For example, a frequency spectrum with a resolution of 31.25 Hz will not register the expected sharp peek of a sinusoidal wave of 110 Hz [4]. The bins closest to 110 Hz would be bin 4 and 5 (0 is considered as the first bin), respectively 93.75 Hz and 125 Hz, which has some deviation from the actual frequency. In order to get a continuous spectrum, an infinite small interval between every frequency bin is needed. The angular frequency ω corresponding to the m-th bin is given in (6).

$$\omega = 2\pi \frac{m}{MT_S} \tag{6}$$

2.2 Signal-to-Noise Ratio (SNR)

SNR compares the power of a desired signal with the power of background noise. SNR is most often expressed in decibel (dB), where a ratio above 0 dB indicates more signal than noise [5]. The SNR for the problem described in the introduction is defined as in (7) [1].

$$\mathrm{SNR}_{\mathrm{dB}} = \frac{A^2}{2\sigma^2} \tag{7}$$

In the above expression, A is the amplitude of the sinusoidal wave $s[n]$ and σ^2 is the variance of $w[n]$. The WGN has zero mean and follow a normal distribution as expressed in (8).

$$w[n] \sim N(0, \sigma^2) \tag{8}$$

The above equation can be explained from the distributions of $w_{\mathrm{r}}[n]$ and $w_{\mathrm{i}}[n]$ which have the properties shown in (9) and (10). These expressions yield that $w_{\mathrm{r}}[n]$ and $w_{\mathrm{i}}[n]$ are uncorrelated and that $w_{\mathrm{r}}[n]$ and $w_{\mathrm{i}}[n]$ follow two different, but identical, normal distributions where $\mathbb{E}\{w_{\mathrm{r}}[n]\} = \mathbb{E}\{w_{\mathrm{i}}[n]\} = 0$ and $\mathrm{Var}\{w_{\mathrm{r}}[n]\} = \mathrm{Var}\{w_{\mathrm{i}}[n]\} = \sigma^2$.

$$\mathbb{E}\{w_{\mathrm{r}}[n] \cdot w_{\mathrm{i}}[n]\} = 0 \tag{9}$$

$$w_{\mathrm{r}}[n] \sim N(0, \sigma^2) \qquad\qquad w_{\mathrm{i}}[n] \sim N(0, \sigma^2) \tag{10}$$

By rearranging (7), one can express σ^2 as shown in (11). When doing so, $\mathrm{SNR}_{\mathrm{dB}}$ has to be converted from the decibel to the linear scale.

$$\sigma^2 = \frac{A^2}{2 \cdot 10^{\frac{\mathrm{SNR}_{\mathrm{dB}}}{10}}} \tag{11}$$

2.3 Cramer Raos Lower Bound (CRLB)

The CRLB expresses the lower bound of the variance of an unbiased estimator which is considered unknown. An estimator that achieves the lower bound, is said to be *efficient* [6]. Considering a data set consisting of N elements as in (12), where each element comes from a known probability density function $p(x, \theta)$, the joined PDF can be expressed as given in (13).

$$\underline{\mathrm{x}} = \big[x[0], x[1], ..., x[N-1]\big]^T \tag{12}$$

$$p(\underline{\mathrm{x}}|\theta) = \prod_{n=0}^{N-1} p(x[n]|\theta) \tag{13}$$

In the case of (13), an unbiased estimator for θ will fulfill the expression given in (14). Furthermore, to express the lower bound of the variance of $\hat{\theta}$ we use the expression given in (15).

$$\mathbb{E}\left[\frac{\partial \log p(\underline{\mathrm{x}}|\theta)}{\partial\theta}\right] = 0 \tag{14}$$

$$\mathrm{Var}(\hat{\theta}) \geq \frac{1}{\mathbb{E}\left[\left(\frac{\partial}{\partial\theta}\log p(\underline{\mathrm{x}}|\theta)\right)^2\right]} \tag{15}$$

In (15), it is assumes there is only one unknown parameter in the PDF which is desired to be estimated. The signal of interest, $x[n]$, needs two estimators; $\hat{\omega}$ and $\hat{\phi}$. To express the CRLB of $\hat{\omega}$ and $\hat{\phi}$, we find the partial derivatives of the joined PDF with respect to these parameters. Doing so gives the *Fisher Information Matrix* which will provide the CRLB for $\hat{\omega}$ and $\hat{\phi}$. The Fisher information matrix is expressed in (16).

$$I(\underline{\theta}) = \begin{bmatrix} -\mathbb{E}\left[\frac{\partial^2 \log p(\underline{\mathrm{X}}|\theta)}{\partial\omega^2}\right] & -\mathbb{E}\left[\frac{\partial^2 \log p(\underline{\mathrm{X}}|\theta)}{\partial\omega\partial\phi}\right] \\ -\mathbb{E}\left[\frac{\partial^2 \log p(\underline{\mathrm{X}}|\theta)}{\partial\omega\partial\phi}\right] & -\mathbb{E}\left[\frac{\partial^2 \log p(\underline{\mathrm{X}}|\theta)}{\partial\phi^2}\right] \end{bmatrix} \tag{16}$$

$\underline{\theta}$ is a vector which is expressed as $\underline{\theta} = \big[\omega, \phi\big]^T$. The CRLB of the variances of $\hat{\omega}$ and $\hat{\phi}$ are given in (17) and (18) [1].

$$\mathrm{Var}(\hat{\omega}) \geq \frac{12\sigma^2}{A^2 T_S^2 N(N^2-1)} \tag{17}$$

$$\mathrm{Var}(\hat{\phi}) \geq \frac{12\sigma^2(n_0^2 N + 2n_0 P + Q)}{A^2 N^2(N^2-1)} \tag{18}$$

In the above expressions, σ^2 is the variance of the noise added to the the sinusoidal signal $s[n]$. Furthermore, N is the length of the data set accumulated by $x[n]$, A is still the amplitude of the sinusoidal signal, T_S is the sampling period and n_0 is the initial bin of the signal $x[n]$. P and Q are expressed in (19) and (20).

$$P = \frac{N(N-1)}{2} \tag{19}$$

$$Q = \frac{N(N-1)(2N-1)}{6} \tag{20}$$

Parameters N, T_S, A and n_0 are set constant, which means the only variable in the CRLB expressions is the variance σ^2.

2.4 Maximum Likelihood Estimator (MLE)

The expression in (13) is said to be the likelihood function of θ with respect to the set of samples given in (12). The MLE of θ is, by definition, the value $\hat{\theta}$ that maximizes $p(\underline{\mathrm{x}}|\theta)$. The definition for the MLE is given in (21)[7].

$$\hat{\theta} = \underset{\underline{\theta}}{\mathrm{argmax}}\ p(\underline{\mathrm{x}}|\theta) \tag{21}$$

The frequency, ω, with the most power in a frequency spectrum, $X(\omega)$, will be regarded as the dominant frequency of the discrete signal $x[n]$. By using the definition of the MLE, the following expression for the estimated dominant frequency $\hat{\omega}$ is given in (22)[1].

$$\hat{\omega} = \underset{\omega_0}{\mathrm{argmax}}\ \big|X(\omega_0)\big| \tag{22}$$

Here $\big|X(\omega_0)\big|$ is the absolute value of the Fourier transform of $x[n]$. The expression for $X(\omega_0)$ is given in (4). The MLE can also be used to find the bin, m^*, in the FFT corresponding to the dominant frequency. m^* is given in (23), where FFT_M denotes an M-point FFT, and $\{x\}$ is the signal vector [1].

$$m^* = \underset{\omega_0}{\mathrm{argmax}}\ FFT_M\{x\} \tag{23}$$

(23) can be substituted in (6) to find the maximum angular frequency $\hat{\omega}_{\mathrm{FFT}}$.

$$\hat{\omega}_{\mathrm{FFT}} = 2\pi \frac{m^*}{MT_S} \tag{24}$$

Given the frequency estimate, we can directly estimate the phase as in (25) [1].

$$\hat{\phi} = \angle\big\{e^{-i\hat{\omega}n_0 T_S} F(\hat{\omega})\big\} \tag{25}$$

3 Task and Method Descriptions

The main task for this project is to investigate the performance of the maximum likelihood estimator by estimating ω and ϕ from (3). In order to qualitatively consider the performance, the variances of the frequency and phase estimation errors will be calculated and compared to their respective CRLB. In addition to estimating the mentioned parameters by the MLE method, a second method which involves fine tuning an estimate with a relatively short FFT size, using a numerical search method, will also be investigated. The motivation behind this method is to lower the run time, as using large FFT sizes are not feasible in practice. Estimates for ω and ϕ in $x[n]$ will thus be computed by exploiting two different methods. These methods are referred to as **Method A)** and **Method B)**. This chapter will provide a description of the methods, which is based on the theory in chapter 2.

Method A

This method involves performing frequency and phase estimations by using the MLE. By performing MLE to accumulate a certain number of estimates, the variances of the *estimation errors* $\sigma^2_{\hat{\omega}}$ and $\sigma^2_{\hat{\phi}}$ are computed and compared to their respective CRLB. The expressions for the estimation error of $\hat{\omega}$ and $\hat{\phi}$ are given in (26) and (27). ω_0 and ϕ are the true values of the estimated parameters $\hat{\omega}$ and $\hat{\phi}$ respectively.

$$e_\omega = \omega_0 - \hat{\omega} \tag{26}$$

$$e_\phi = \phi - \hat{\phi} \tag{27}$$

Equation (28) and (29) shows that the the variance of the estimation errors is equivalent to the variance of each estimator. This is why it makes sense to compare the variance of the estimation errors to their respective CRLB.

$$\begin{aligned} \mathrm{Var}\{e_\omega\} &= \mathrm{Var}\{\omega_0 - \hat{\omega}\} \\ &= \mathrm{Var}\{\omega_0\} + \mathrm{Var}\{-\hat{\omega}\} \\ &= \mathrm{Var}\{\hat{\omega}\} \\ &= \sigma^2_{\hat{\omega}} \end{aligned} \tag{28}$$

$$\begin{aligned} \mathrm{Var}\{e_\phi\} &= \mathrm{Var}\{\phi - \hat{\phi}\} \\ &= \mathrm{Var}\{\phi\} + \mathrm{Var}\{-\hat{\phi}\} \\ &= \mathrm{Var}\{\hat{\phi}\} \\ &= \sigma^2_{\hat{\phi}} \end{aligned} \tag{29}$$

The MLE method will be performed with different levels of applied noise (SNR) and FFT sizes in order to investigate the robustness of the resulting estimators. The robustness is determined by observing the variance and mean of the estimators. The estimates will be performed with SNR values from -10 dB to 60 dB, in steps of 10 dB. The FFT size is defined as $M = 2^k$, where $k \in [10, 12, 14, 16, 18, 20]$. For clarity, the estimates for the different SNRs will be computed for all of the different FFT sizes.

Method B

Even though the FFT distinguishably decreases the complexity of the direct computation of the DFT, there are still a non-ideal amount of computations for large values of k. In this method we will estimate $\hat{\omega}$ and $\hat{\phi}$, without increasing k, but rather use a different approach involving a numerical search method; the *Nelder-Mead method*. The idea behind this numerical search method is to generate a new signal without noise and then search for the frequency which will provide the minimum *mean squared error* (MSE) between the original signal and the generated one.

Similarly with Method A, The Nelder-Mead method also involves computing the FFT of $x[n]$. To make the computation as cost efficient as possible, the length of the FFT is set to $k = 2^{10}$ and is kept constant. The SNR values are still the same and goes from -10 dB to 60 dB, in steps of 10 dB.

4 Implementation

This section contains the implementation of the methods described in section 3. *Python* is chosen as programming language. The implementation walk through will yield self defined functions as well the concepts behind the code. Signal $x[n]$, which was given in section 1, is defined in the code as (30).

$$\begin{aligned} x[n] &= Ae^{i(\omega(n+n_0)T_S+\phi)} + w[n] \\ &= s[n] + w[n] \end{aligned} \tag{30}$$

All variables for $x[n]$ are known, except evidently ω and ϕ which we want to estimate. The table below lists all constants used in the estimation process.

Constant	**Symbol**	**Value**
Sample period	T_S	10^{-6}
Sampling frequency	F_S	10^6
Actual phase	ϕ	$\frac{\pi}{8} = 22,5°$
Amplitude	A	1
Actual angular frequency	ω_0	$2\pi \cdot 10^5$ [Hz]
Actual frequency	f_0	10^5 [Hz]
Initial bin	n_0	-256
Sequence length	N	513

Table 1: Table of constants.

In both methods, a vector with accumulated values for $x[n]$ is required to estimate $\hat{\omega}$ and $\hat{\phi}$. This is done by making a function `generate_signal()`. The function returns a vector with N elements, where each element is complex. The sinusoidal wave, $s[n]$, and the WGN, $w[n]$, are made separately and then summed together to make $x[n]$.

Values for $w[n]$ are accumulated through $w_{\mathrm{r}}[n]$ and $w_{\mathrm{i}}[n]$. The values are put into a vector `gwn` with length N, where each element is evidently complex. Like $w[n]$, data from $s[n]$ is gathered in a vector `s_n` with same size as `gwn` to make the summation possible. It is noteworthy to underline that $s[n]$ is deterministic and $w[n]$ is not. Every time a signal is generated, `s_n` will contain the same values, while `gwn` will have different values due to the normal distribution.

Built-in functions in Python are used in both methods. The functions come the libraries `numpy`, `statistics` and `scipy.optimize`. The latter library is used in Method B for numerical search. To export result into an Excel sheet, *Pandas* is used.

Method A

A flow sheet of the implemented code of Method A is given in figure 3. The MLE method begins with generating a signal, $x[n]$, by using the function `generate_signal()`. The FFT is then performed on $x[n]$ to get the frequency spectrum. To find bin m^* of $X(\omega)$ that corresponds to the dominant frequency, expression (23) is used. Having a value for m^*, equation (24) is used to give an estimate of $\hat{\omega}$. This process is repeated for 100 iterations to make a vector contain different estimates of $\hat{\omega}$. The amount of iterations is set to 100 because it is a convenient number considering run time. The vector is then used to compute the mean and variance of the estimates.

To find the estimate for the angular frequency $\hat{\phi}$, equation (25) is used. We observe that the estimate of $\hat{\phi}$ is dependent of $\hat{\omega}$. This means that if $\hat{\omega}$ turns out not being a robust estimator, neither will $\hat{\phi}$. The estimate process of finding $\hat{\phi}$ is repeated 100 times to make a vector with purpose of computing the average and variance of resulting estimates.

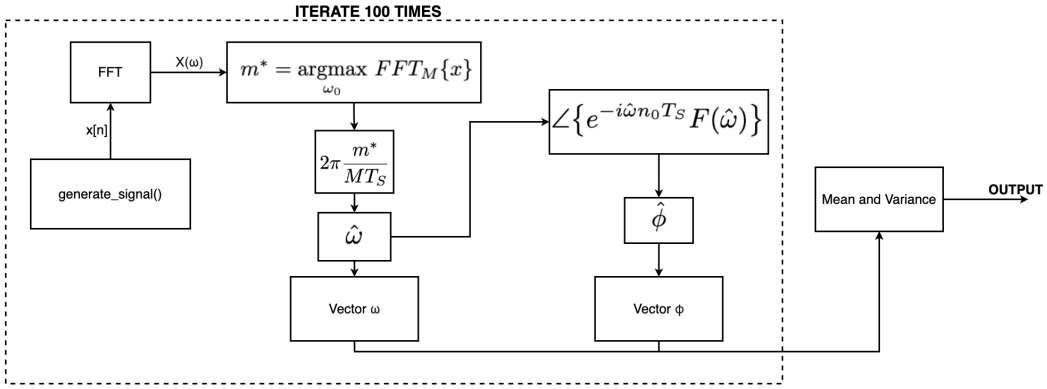

Figure 3: Flow sheet describing the code step by step.

Method B

A flow sheet of the implementation of Method B is given in figure 4. The code commences by generating two signals. The first signal is generated by using the function `generate_signal()`. The second signal has the same form as $s[n]$, but do not share the same frequency. When the code starts, $s[n]$ is set to an initial frequency. The FFT is then performed on the two signals, followed by computing the MSE of the absolute value of the two spectrums. The Nealder-Mead method is then used to perform a numerical search of the MSE. The numerical search is done by iterating through frequencies of $s[n]$ to minimize the MSE. Function `optimize.minimize()` from the `scipy.optimize` library is used to perform the numerical search. As in Method A, the code is iterated 100 times to make a vector consisting of 100 estimates for $\hat{\omega}$. Again the mean and variance is computed.

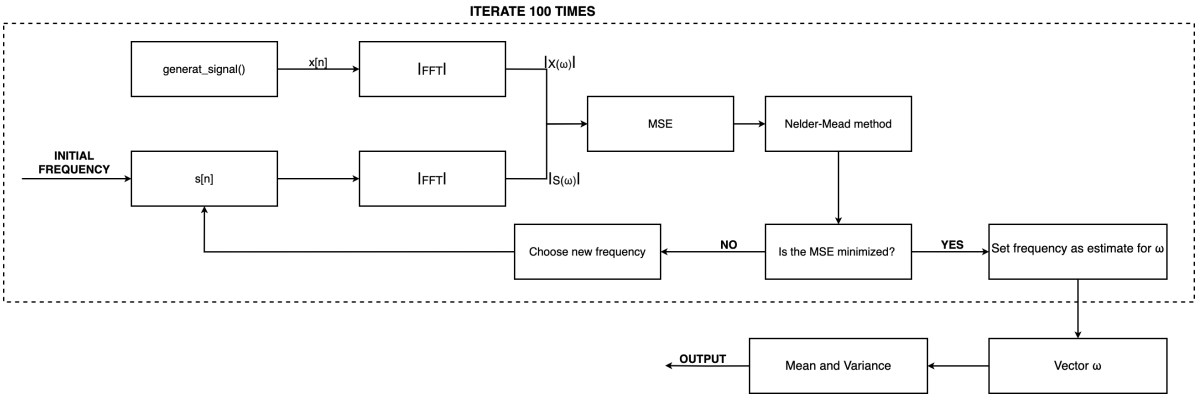

Figure 4: Flow chart describing the code step by step.

5 Results and Discussion

This section presents the results from the two methods. Tables 3 and 4, in the appendix, display the results from Method A. The estimates for $\hat{\omega}$ are presented as $\hat{f} = \frac{\hat{\omega}}{2\pi}$ because it was considered more intuitive to use this unit. The same applies for table 2 which yields results for Method B. The results for $\hat{\phi}$ are presented in degrees instead of radians for the same reason.

Method A

Tables in appendix A and B display the results of the mean, the variance and the error of 100 frequency and phase estimations for different values of SNR and FFT sizes. The CRLB for different values of SNR and FFT sizes are also listed. From theory we expect that higher values of SNR and FFT size will lead to a smaller error, $e_{\hat{\omega}}$, and $e_{\hat{\phi}}$, in other words; more accurate estimates of $\hat{\omega}$ and $\hat{\phi}$.

We begin looking at the results in figure 5 of the estimate error ,$e_{\hat{\omega}}$, displayed in the graph on the left. The graph shows the absolute value of $e_{\hat{\omega}}$ with different FFT sizes. The sensitivity of the $\hat{\omega}$ estimator increases as the FFT size becomes greater. This can be observed by looking at the blue and the green line, representing FFT sizes $M = 2^{10}$ and $M = 2^{20}$ respectively. As the SNR value becomes larger than 0 dB, meaning that the signal $s[n]$ becomes more dominant than the noise, the blue line becomes constant. This is due to the low frequency resolution that prevents an estimate of size $M = 2^{10}$ to distinguish the dominant frequency in the spectrum. As for the green line, the frequency resolution is higher and the estimate becomes more sensitive for changing SNR values. This can be shown by observing that the green line changes for different values of SNR. The lowest achieved value for $e_{\hat{\omega}}$ was $0,057$ Hz for an FFT size $M = 2^{20}$ and SNR 40 dB. The fact that green line increases after passing 40 dB is believed to be caused by pure coincidence due to the WGN.

Figure 5: On the left: absolute value of the estimation error $e_{\hat{\omega}}$ in a logarithmic scale. On the right: variance of the estimation error $\sigma^2_{\hat{\omega}}$ for an FFT size of $M = 2^{20}$ and the CRLB defined in 17.

The graph on the right in figure 5 displays the variance of the estimation error $\sigma^2_{\hat{\omega}}$ as well as the CRLB defined in (17). The graph implies that the variance of the estimation error will not always be larger than the CRLB. This is expected based on the stated theory, because the CRLB will only work for estimators that are unbiased. An estimator computed by the MLE method will be biased because of the limited frequency resolution. To achieve an unbiased estimator using this method, would require an infinite small distance $\Delta\omega$ between every frequency bin in the spectrum. As already mentioned, this is not feasible in real applications because of the many computations it requires. The graph on the left in figure 5 implies that the estimation errors for FFT sizes less than $M = 2^{20}$ become constant before reaching 60 dB. This is the reason why some values of $\sigma^2_{\hat{\omega}}$ become zero, and also the reason why only the variance of FFT size $M = 2^{20}$ is plotted. The 0 variances yields no information.

Results from the absolute value of the estimation errors $e_{\hat{\phi}}$ are displayed in the the left graph of figure 6. From section 4, we know that the estimate of $\hat{\phi}$ is dependent of the estimate of $\hat{\omega}$. Assuming that

the ideal FFT size for $\hat{\omega}$ is $M = 2^{20}$ to maximize the frequency resolution, one would expect the green line to begin below the other lines, but this is not the case. The estimation error is rather dependent on the SNR, where a higher SNR value results in a smaller estimation error. This is believed to be caused by less amount of noise in the signal due to the higher SNR value. Looking at (25), we see that the expression involves the spectrum of $x[n]$. When there is less noise in the signal, the spectrum will look more like $S(\omega)$ which will contribute to a more precise estimate of $\hat{\phi}$. Of the the six lines in the left graph of figure 6, the green line ended up giving the lowest estimate error of $5,58218 \cdot 10^{-5}$ degrees at 60 dB. This correlates well with the stated theory in section 2 and is an expected result. Looking at table 3 and 4, one would expect that $\hat{\phi}$ becomes constant when the variance of $\hat{\omega}$ is zero, but this is not the case. This can be explained through (25). Since every iteration of `generate_signal()` yields a slight variation of the signal for $x[n]$ due to the WGN, the spectrum $X(\omega)$ will change for every iteration. Even though $\hat{\omega}$ is the same, we will get a different value for $\hat{\phi}$ every time.

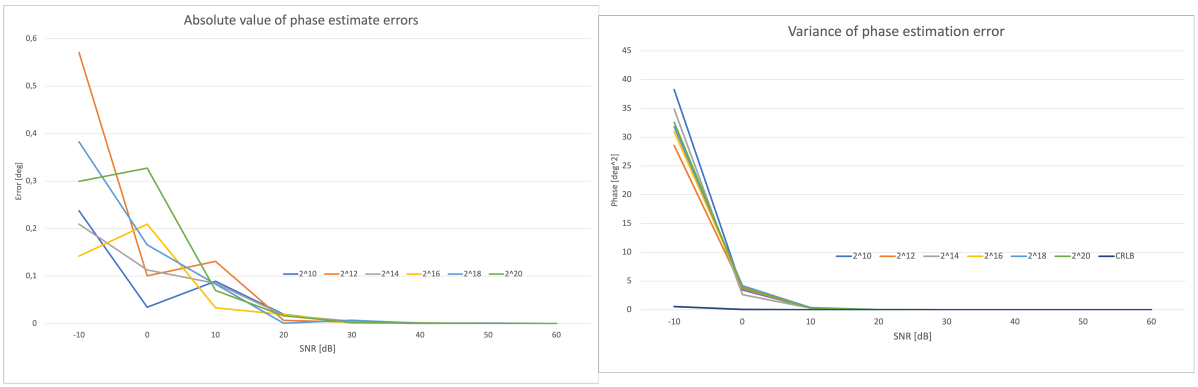

Figure 6: On the left: absolute value of the estimation error $e_{\hat{\phi}}$. On the right: variance of the estimation error $\sigma^2_{\hat{\phi}}$ for different FFT sizes and the CRLB defined in 18.

The graph on the right in figure (6) displays the variance of the estimation errors $\sigma^2_{\hat{\phi}}$ as well as the CRLB defined in 18. The CRLB is the dark blue line closest to the x-axis. Unlike $\sigma^2_{\hat{\omega}}$, the variance of the estimation error is always above the CRLB. This can be explained by the same reason as motioned above, that the spectrum $X(\omega)$ will change for every iteration.

Method B

Table 2 presents the the estimation error and the variance of the estimation error $e_{\hat{\omega}}$ from Method B. As the table implies, the Nelder-Mead method is almost unbias for a SNR value of 60 dB. All the variances of the estimation errors is above the CRLB which is correct according to theory. The table yields that the Nelder-Mead method gives more exact estimates for $\hat{\omega}$ for bigger values of SNR. This makes sense considering that the numerical search will output a MSE for a spectrum with less noise, in other words a spectrum more similar to $S(\omega)$.

Table 2: Results for $\hat{\omega}$ from 100 simulations with fine tuning.

SNR [dB]	Frequency average [Hz]	Error [Hz]	Variance of $e_{\hat{\omega}}$ [Hz^2]	CRLB$_f$ [Hz^2]
-10	100051,0329	51,03289817	883267,9432	11257,47701
0	100012,4098	12,40981312	12644,82161	1125,747701
10	100005,1228	5,122813883	1291,766297	112,5747701
20	100001,0612	1,061189708	116,2972797	11,25747701
30	99999,34299	0,657008726	13,28946517	1,125747701
40	99999,89765	0,102351958	1,034422745	0,11257477
50	100000,0036	0,00364185	0,101725986	0,011257477
60	100000,0048	0,004777312	0,010040365	0,001125748

Comparison of Methods

The graph in figure 7 compares the estimation errors found by the MLE method for FFT lengths $M = 2^{10}$ and $M = 2^{20}$ as well as the estimation error from the Nelder-Mead method (see table 2). As the graph implies, the Nelder-Mead method performs much better than the MLE method for an FFT length $M = 2^{10}$. From the tables and figure 7, it appears that the deviation from the original frequency of, $f_0 = 100000$Hz, is considerable decreased, from approximately 390Hz to 51Hz for SNR -10dB. When the FFT length is $M = 2^{20}$, we observe that the difference of estimation error from the two methods decreases. The Nelder-Mead method seems to perform better for larger values of SNR, where the difference between the two lines (orange and grey line) is approximatly $0,377$ Hz at 60 dB. Despite having small differences between the estimation errors for the two methods is small, one could argument that the Nelder-Mead is a more suitable method concerning computational complexity.

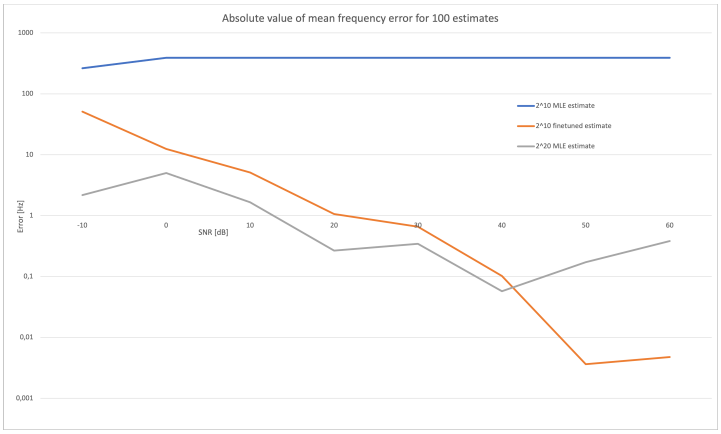

Figure 7: Absolute value of the estimation error $e_{\hat{\omega}}$ from MLE method for $M = 2^{10}$ and $M = 2^{20}$, in addition to the estimation errors from the Nelder-Mead method. The lines are plotted in a logarithmic scale.

6 Conclusion

In this rapport the MLE and Nelder-Mead method have been used to estimate the frequency $\hat{\omega}$ and phase $\hat{\phi}$ of a discrete complex sinusoidal wave with added noise. The MLE method was used by finding the bin in the frequency spectrum of $x[n]$ that corresponded to the most dominant frequency. With $\hat{\omega}$ found, one could also compute $\hat{\phi}$. To document the performance of the MLE, different values of SNR and FFT sizes were used. The Nelder-Mead method involves numerical search to minimize a function. The method was used by finding the frequency which minimized the MSE of the frequency spectrum of $x[n]$ and $s[n]$. The performance of the methods was documented for FFT size $M = 2^{10}$ and different SNR values.

The MLE method showed good results for FFT size $M = 2^{20}$. For this FFT size, the lowest estimate error of the frequency was $e_{\hat{\omega}} = 0,057$ Hz. The Nelder-Mead method showed also good results. The lowest estimate error was $e_{\hat{\omega}} = 0,004777$ Hz. Compared to the MLE for FFT size $M = 2^{10}$, the Nelder-Mead method performed much better taking into consideration that both methods used equal FFT size. Even for FFT size $M = 2^{20}$, the Nelder-Method performed well and computed an almost unbiased estimate of $\hat{\omega}$ for SNR value of 60 dB. Even though the MLE for $M = 2^{20}$ and the numerical method yield similar results, one could argument that Nelder-Method is more suitable to use in this case considering the computational complexity. The MLE gave a good result for the estimate error for the phase $e_{\hat{\phi}}$. The estimate was close to the original value $\phi_0 = 22,5°$ with a $5,58218 \cdot 10^{-5}$ deviation for SNR=60 dB and FFT size of $M = 2^{20}$.

The variance of estimated error for the frequency $\sigma^2_{\hat{\omega}}$ was found to not always stay above the CRLB. This is because the MLE is biased. To be considered unbiased, an infinitely small distance between every frequency bin is needed. Unlike $\sigma^2_{\hat{\omega}}$, $\sigma^2_{\hat{\phi}}$ was always above the CRLB.

References

[1] TTT4175: *TTT4175 Estimation, Detection and Classification Project descriptions: Estimation Theory*. Department of Electronic Systems, Spring 2022.

[2] John G. Proakis, Dimitris K. Monalikis: *Digital Signal Processing Fourth Edition*. Pearson Education Limited, 2014.

[3] *Fast fourier transform*. `https://en.wikipedia.org/wiki/Fast_Fourier_transform`.

[4] Marsar, J.: *Improving FFT Frequency Resolution*. EEL 4930: Real-Time Digital Signal Processing, Winter 2015.

[5] *Signal-to-noise ratio*. `https://en.wikipedia.org/wiki/Signal-to-noise_ratio`.

[6] *Cramér–rao bound*. `https://en.wikipedia.org/wiki/Cram%C3%A9r%E2%80%93Rao_bound`.

[7] Richard O. Duda, Peter E. Hart, David G. Stork: *Introduction to Statistical Pattern Recognition*. Academic Press, New York, 1990.

[8] TTT4280: *User Guide for the Laboratory*. Department of Electronic Systems, Spring 2022.

[9] Liaquat, M.U.;Munawar, H.S.; Rahman A.; Qadir Z.; Kouzani A.Z.; Mahmud M.A.P.: *Sound localization for ad-hoc microphone arrays*. `https://doi.org/10.3390/en14123446`.

[10] *Nelder–mead method*. `https://en.wikipedia.org/wiki/Nelder%E2%80%93Mead_method`.

A Results from Method A: 100 iterations of $\hat{\omega}$

Table 3: Results for $\hat{\omega}$ from 100 estimates with maximum likelihood estimator.

Length of FFT	SNR [dB]	Frequency average [Hz]	Error [Hz]	Variance of $e_{\hat{f}}$ [Hz^2]	CRLB [Hz^2]
2^10	-10	99736,32813	263,671875	108950,0658	11257,477
2^10	0	99609,375	390,625	0	1125,7477
2^10	10	99609,375	390,625	0	112,57477
2^10	20	99609,375	390,625	0	11,257477
2^10	30	99609,375	390,625	0	1,1257477
2^10	40	99609,375	390,625	0	0,11257477
2^10	50	99609,375	390,625	0	0,01125748
2^10	60	99609,375	390,625	0	0,00112575
2^12	-10	99980,46875	19,53125	16231,72953	11257,477
2^12	0	100036,6211	36,62109375	11288,75848	1125,7477
2^12	10	100095,2148	95,21484375	596,0464478	112,57477
2^12	20	100097,6563	97,65625	0	11,257477
2^12	30	100097,6563	97,65625	0	1,1257477
2^12	40	100097,6563	97,65625	0	0,11257477
2^12	50	100097,6563	97,65625	0	0,01125748
2^12	60	100097,6563	97,65625	0	0,00112575
2^14	-10	99993,28613	6,713867188	12816,1275	11257,477
2^14	0	100002,4414	2,44140625	1453,992092	1125,7477
2^14	10	99993,28613	6,713867188	774,7851237	112,57477
2^14	20	99977,41699	22,58300781	109,5009573	11,257477
2^14	30	99975,58594	24,4140625	0	1,1257477
2^14	40	99975,58594	24,4140625	0	0,11257477
2^14	50	99975,58594	24,4140625	0	0,01125748
2^14	60	99975,58594	24,4140625	0	0,00112575
2^16	-10	99981,84204	18,15795898	10179,14408	11257,477
2^16	0	99998,62671	1,373291016	1051,242115	1125,7477
2^16	10	99999,69482	0,305175781	109,0305923	112,57477
2^16	20	100001,0681	1,068115234	51,99884375	11,257477
2^16	30	100005,188	5,187988281	13,26429121	1,1257477
2^16	40	100006,1035	6,103515625	0	0,11257477
2^16	50	100006,1035	6,103515625	0	0,01125748
2^16	60	100006,1035	6,103515625	0	0,00112575
2^18	-10	99987,4115	12,58850098	9828,128363	11257,477
2^18	0	100005,3024	5,302429199	1253,903258	1125,7477
2^18	10	99999,16077	0,839233398	129,1680912	112,57477
2^18	20	100000,7629	0,762939453	11,46514533	11,257477
2^18	30	99999,54224	0,457763672	2,963299101	1,1257477
2^18	40	99998,74115	1,258850098	0,956898668	0,11257477
2^18	50	99998,47412	1,525878906	0	0,01125748
2^18	60	99998,47412	1,525878906	0	0,00112575
2^20	-10	99997,82562	2,174377441	9174,39239	11257,477
2^20	0	99995,01228	4,987716675	1293,736094	1125,7477
2^20	10	100001,6594	1,659393311	103,5027019	112,57477
2^20	20	100000,267	0,267028809	11,36004817	11,257477
2^20	30	99999,65668	0,343322754	1,049501765	1,1257477
2^20	40	100000,0572	0,057220459	0,206152132	0,11257477
2^20	50	100000,1717	0,171661377	0,157645748	0,01125748
2^20	60	100000,3815	0,381469727	0	0,00112575

B Results from Method A: 100 iterations of $\hat{\phi}$

Table 4: Results for $\hat{\phi}$ from 100 estimates based on $\hat{\omega}$

Length of FFT	SNR [dB]	Phase average [deg]	Error [deg]	Variance of $e_{\hat{\phi}}$ [deg^2]	CRLB$_{\hat{\phi}}$ [deg^2]
2^10	-10	22,73766011	0,237660114	38,26190839	0,558438397
2^10	0	22,46533133	0,034668669	3,465314832	0,05584384
2^10	10	22,58914156	0,089141558	0,324107266	0,005584384
2^10	20	22,51933447	0,019334467	0,040571527	0,000558438
2^10	30	22,50242344	0,00242344	0,003661082	5,58438E-05
2^10	40	22,49909297	0,00090703	0,000403282	5,58438E-06
2^10	50	22,50117323	0,001173228	3,36345E-05	5,58438E-07
2^10	60	22,49992046	7,95391E-05	3,41876E-06	5,58438E-08
2^12	-10	21,92906975	0,570930247	28,55713223	0,558438397
2^12	0	22,39931384	0,100686158	3,664451771	0,05584384
2^12	10	22,36908929	0,130910707	0,300110463	0,005584384
2^12	20	22,49322914	0,006770861	0,025994837	0,000558438
2^12	30	22,503154	0,003153997	0,003119206	5,58438E-05
2^12	40	22,50016563	0,000165634	0,000241547	5,58438E-06
2^12	50	22,50025485	0,000254849	2,86302E-05	5,58438E-07
2^12	60	22,49998159	1,84126E-05	2,8805E-06	5,58438E-08
2^14	-10	22,70937143	0,209371428	34,8441305	0,558438397
2^14	0	22,38690984	0,113090159	2,639406741	0,05584384
2^14	10	22,58495941	0,084959412	0,308245456	0,005584384
2^14	20	22,51770737	0,017707375	0,032305199	0,000558438
2^14	30	22,505174	0,005174005	0,003297363	5,58438E-05
2^14	40	22,50170459	0,001704592	0,000369616	5,58438E-06
2^14	50	22,49969062	0,000309375	4,37616E-05	5,58438E-07
2^14	60	22,49979631	0,000203692	3,30127E-06	5,58438E-08
2^16	-10	22,64213514	0,142135144	30,97979553	0,558438397
2^16	0	22,29055565	0,209444354	4,032546439	0,05584384
2^16	10	22,46672778	0,033272221	0,278273419	0,005584384
2^16	20	22,48080085	0,019199151	0,031150971	0,000558438
2^16	30	22,50102768	0,001027684	0,002586103	5,58438E-05
2^16	40	22,50103033	0,00103033	0,000291607	5,58438E-06
2^16	50	22,50022595	0,000225951	3,1424E-05	5,58438E-07
2^16	60	22,50008626	8,62592E-05	2,93617E-06	5,58438E-08
2^18	-10	22,11746853	0,382531472	31,82430358	0,558438397
2^18	0	22,66601768	0,166017677	4,179561025	0,05584384
2^18	10	22,41674793	0,083252071	0,391555783	0,005584384
2^18	20	22,50085636	0,000856357	0,029696288	0,000558438
2^18	30	22,50696851	0,006968513	0,00433623	5,58438E-05
2^18	40	22,50057222	0,000572222	0,000369401	5,58438E-06
2^18	50	22,49916222	0,000837777	3,36971E-05	5,58438E-07
2^18	60	22,4999355	6,44995E-05	3,41337E-06	5,58438E-08
2^20	-10	22,79940093	0,299400926	32,59419129	0,558438397
2^20	0	22,82722785	0,327227852	3,917634095	0,05584384
2^20	10	22,56965736	0,069657361	0,290745235	0,005584384
2^20	20	22,51643759	0,01643759	0,033881838	0,000558438
2^20	30	22,50248254	0,002482536	0,003481353	5,58438E-05
2^20	40	22,50065044	0,000650438	0,000354332	5,58438E-06
2^20	50	22,50044848	0,000448483	3,26988E-05	5,58438E-07
2^20	60	22,49994418	5,58218E-05	3,63408E-06	5,58438E-08