kapteyn astronomical
institute

Parameter estimation in signal with white Gaussian noise

Petar I. Penchev (s4683099)

Statistical Signal Processing [WMAS011-05] - Final Project

Abstract: This project investigates parameter estimation for a signal embedded in white Gaussian noise (WGN) using statistical signal processing techniques. A generative model for the signal is analyzed to estimate key parameters through Maximum Likelihood Estimation (MLE) implemented via the Newton-Raphson method. Monte Carlo simulations are employed to evaluate the means and variances of the estimators, while the Fisher Information Matrix serves to derive theoretical bound of said variance, known as Cramér-Rao Lower Bound (CRLB). Bayesian analysis is conducted by computing posterior distributions, further refined through Markov Chain Monte Carlo (MCMC) sampling using the Metropolis-Hastings algorithm. Results highlight the Gaussian nature of the estimates, the correlations between parameters, and discrepancies between observed variances and CRLB.

Contents

1 Introduction

Signal processing often involves extracting meaningful information from data contaminated by noise. In this project, we analyze a dataset of a signals in white Gaussian noise (WGN) and try to estimate the parameters of a mathematical model describing the underlying signal.

This report aims to:

1. Derive and numerically and evaluate the Maximum Likelihood Estimator (MLE) for the model describing our signal.
2. Assess the accuracy of our parameter estimators, using Monte Carlo methods, to compute their means and variances.
3. Analyze the theoretical bounds of said estimators using the Fisher Information Matrix and Cramér-Rao Lower Bound (CRLB).
4. Compute the likelihood function of the parameter grid and marginalize said function to compute the marginalized probability densities.
5. Explore the likelihood function and posterior distributions using numerical and sampling-based methods particularly belonging to the family of Markov Chain Monte Carlo (MCMC) algorithms.

This report emphasizes the role of statistical signal processing techniques in real-world applications. It provides an inside into the complexities and interplay between modeling, parameter estimation, the role of noise, and the accuracy of our estimates.

2 Problem Description

2.1 Problem Statement

As briefly mentioned in Section 1, we are provided with a dataset of a signal that is contaminated by White Gaussian Noise (WGN). WGN refers to a type of additive noise characterized by a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, where the mean μ is typically assumed to be zero, and the variance σ^2 quantifies the noise's strength in the input signal.

The primary objective of this project is to estimate three parameters of a generative model that characterizes the underlying signal. This involves deriving the Maximum Likelihood Estimator (MLE) for the model parameters and evaluating the performance of these estimates using Monte Carlo simulations. These simulations will provide insights into the mean and variance of the estimators. Additionally, the variances obtained from the simulations will be compared with the analytically derived Cramér-Rao Lower Bound (CRLB) to evaluate the efficiency of the estimators. Finally, the marginal Probability Density Functions (PDFs) for various parameter combinations will be computed and analyzed to gain a deeper understanding of the estimators' behavior.

2.2 Model

The observed signal in noise can be written in the following way,

$$\mathbf{x} = \mathbf{s} + \mathbf{w}, \tag{2.1}$$

where $\mathbf{x} = [x_0, x_1, ..., x_{N-1}]$ are the observed data points, $\mathbf{w} = [w1, w2, ..., wN]$ is WGN with $\mathcal{N}(\mu = 0, \sigma^2 = 0.0025)$, and $\mathbf{s}$ is our model. The signal is modeled in the following way,

$$s = A \times \left(\frac{\nu}{\nu_0}\right)^{\alpha} \left(1 + \frac{\nu}{\nu_o}\right)^{-4\alpha}, \tag{2.2}$$

where ν is the frequency and A, ν_0 and α are the parameters to be estimated. Combining Equation 2.2 and 2.1 we can write an expression for each observed data point,

$$x_i = A \times \left(\frac{\nu_i}{\nu_0}\right)^{\alpha} \left(1 + \frac{\nu_i}{\nu_o}\right)^{-4\alpha} + w_i \tag{2.3}$$

2.3 Dataset

The dataset consists of three columns, the first being the frequency ν, the second being the signal s, and the third the signal in noise s_n. There are 500 data points in this dataset, with a range of amplitude between 0.5 and 1.1, spread across a frequency range between 0 and 1. The dataset can be found on the Github repository (refer to A.1). A plot of the data can be seen bellow.

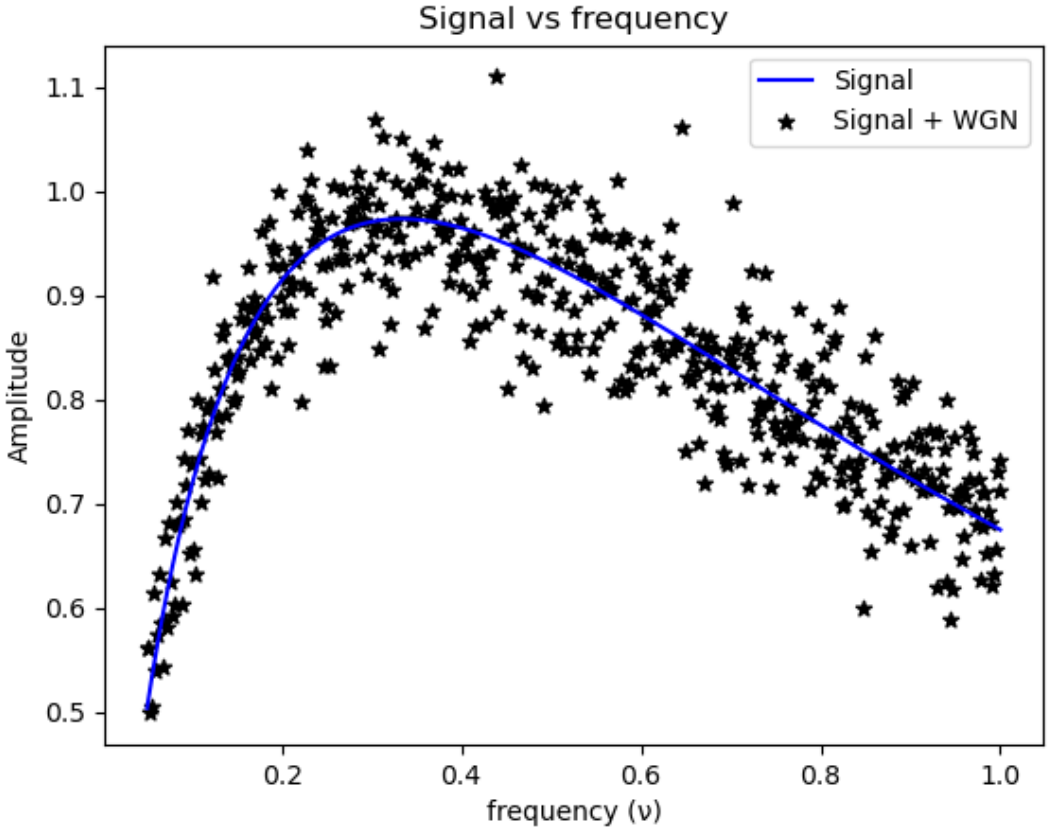


Figure 2.1: The figure on the top represents the perfect signal without noise (blue line) and the signal in noise (black stars).

3 Theory

3.1 Model Behavior

Examining the limits

In the limit where $\nu_0 \gg \nu$ we have that the terms involving these two quantities tend to go towards 0 as

$$\lim_{\nu_0 \gg \nu} \frac{\nu}{\nu_0} \approx 0 \tag{3.1}$$

Thus we have that our signal in noise in the limit behaves as,

$$\lim_{\nu_0 \gg \nu} x_i = \lim_{\nu_0 \gg \nu} A \times \left(\frac{\nu_i}{\nu_0}\right)^{\alpha} \left(1 + \frac{\nu_i}{\nu_o}\right)^{-4\alpha} + w_i \approx w_i, \tag{3.2}$$

because our signal goes to 0 ($s \to 0 \quad for\ \nu_0 \gg \nu$). If we flip the sign and look at the limit where $\nu \gg \nu_0$ we have that,

$$\lim_{\nu \gg \nu_0} \frac{\nu}{\nu_0} \approx \nu. \quad for\ \nu_0 \approx 1. \tag{3.3}$$

Our signal in noise then becomes dominated by the frequency

$$\lim_{\nu \gg \nu_0} x_i \approx A \times \nu_i^{\alpha} \left(1 + \nu_i\right)^{-4\alpha} + w_i \quad for\ \nu_0 \approx 1, \tag{3.4}$$

Probable values of the parameters

If we take the partial derivative of Equation 2.2 with respect to ν we have that

$$\frac{\partial s}{\partial \nu} = -\frac{A\alpha \left(\frac{\nu}{\nu_0}\right)^{\alpha} (3\nu - \nu_0) \left(\frac{\nu+\nu_0}{\nu_0}\right)^{-4\alpha}}{\nu\,(\nu + \nu_0)} \tag{3.5}$$

Assuming A and α to be greater than 0, we know that the root of this equation equals,

$$\nu_0 = 3\nu \quad , \tag{3.6}$$

for the non trivial solution. This root represents a peak of the signal, which occures at about $\nu \approx 0.33$. Thus we can have an educated guess for ν_0 of

$$\nu_0 \approx 1.0. \tag{3.7}$$

Using 3.7 and $\nu = 1$, Equation 2.2 becomes,

$$s = A \times 2^{-4\alpha} \tag{3.8}$$

reading out the signal value at that frequency from Figure 2.1, and solving for A we have that,

$$A = 0.67 \times 2^{4\alpha}. \tag{3.9}$$

We can repeat the same process for $\nu = 0.05$ and we get another equation for A,

$$A = \frac{0.5(1 + 0.5)^{4\alpha}}{0.5^{\alpha}}. \tag{3.10}$$

Solving this system of equations we find that

$$\alpha \approx 0.7 \tag{3.11}$$

$$A \approx 4.7 \tag{3.12}$$

Thus the probable values of those parameters are:

Parameter	Probable Value
A	4.7
ν_0	1.0
α	0.7

Table 3.1: Summary of probable parameter values.

3.2 The Likelihood Function

Since we know that our signal (as described by Equation 2.2) is embedded in Gaussian noise ($\mathcal{N}(0, \sigma)$), the probability of a single data points x_i can be modeled as follows

$$p(x_i \mid s) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i - s)^2}{2\sigma^2}\right), \tag{3.13}$$

where s is the model of our signal, and σ is the standart deviation of the noise. Assuming that each data point is independent of each other we can find the probability of the vector representation of the data $\mathbf{x} = [x_0, x_1, ..., x_{N-1}]$ as being the product of the probability of each individual data point,

$$p(\mathbf{x} \mid s) = \prod_{n=0}^{N-1} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x[n] - s)^2}{2\sigma^2}\right) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{n=0}^{N-1} (x[n] - s)^2\right) \tag{3.14}$$

In this case it is much more convinient do use the natural log of the probability since it has the nice proporty of being monotonic, all of our subsequent analysis wouldn't change. Thus we get,

$$\mathcal{L} = \ln p(\mathbf{x} \mid s) = -\frac{N}{2}(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{n=0}^{N-1} (x[n] - s)^2 \tag{3.15}$$

which is the log-likelihood of the data.

3.3 Maximum Likelihood Estimation (MLE)

The MLE is defined as maximizing the likelihood with respect to the parameter we want to estimate. We can write this as follows,

$$\theta_{\text{ML}} = \arg\max_{\theta} p(\mathbf{x} \mid \theta), \tag{3.16}$$

which can be found by solving,

$$\frac{\partial p(\mathbf{x} \mid \theta)}{\partial \theta}\Big|_{\text{ML}} = 0 \tag{3.17}$$

or in the more practical case,

$$\frac{\partial \ln p(\mathbf{x} \mid \theta)}{\partial \theta}\Big|_{\text{ML}} = 0 \tag{3.18}$$

The MLE has a number of theoretically desirable properties such as the fact that in the large enough samples it is unbiased, attains the CRLB, and is efficient.

In this paper we use the Newton-Raphson [Burden and Faires (2005)] in attempt to maximumize the log-likelihood function by finding the zero of its derivative, particularly Equation 3.18. If we let

$$g(\theta) = \frac{\partial \ln p(\mathbf{x} \mid \theta)}{\partial \theta}, \tag{3.19}$$

and expand around a first guess $\theta_{\mathbf{0}}$, Newton–Raphson then assume that $g(\theta)$ is linear (up to first order) around $\theta_{\mathbf{0}}$, so we write

$$g(\theta) \approx g(\theta_{\mathbf{0}}) + \frac{dg(\theta)}{d\theta}\Big|_{\theta=\theta_{\mathbf{0}}} (\theta - \theta_{\mathbf{0}}) \tag{3.20}$$

We can use that equation to iteratively converge to the next zero by rolling down the gradient and approaching a local minima. The iterative step in the Newton-Raphson algorithm is then given by,

$$\theta_{k+1} = \theta_k - \frac{g(\theta_k)}{\frac{dg(\theta)}{d\theta}\Big|_{\theta=\theta_k}}, \tag{3.21}$$

which only depends on the last θ_k. We can rewrite the step in terms of our log-likelihood function (given by Equation 3.15) as,

$$\theta_{k+1} = \theta_k - \alpha \left(\frac{\partial^2 \mathcal{L}}{\partial \theta^2} \right)^{-1} \frac{\partial \mathcal{L}}{\partial \theta}\Big|_{\theta=\theta_k}, \tag{3.22}$$

where α is a hyperparameter that controls the learning rate.

Vectorizing our problem and letting $\mathbf{\Theta} = \begin{bmatrix} A & \nu_0 & \alpha \end{bmatrix}^T$. The Newton-Raphson iterative step can be written in vector form as follows,

$$\mathbf{\Theta}_f = \mathbf{\Theta}_i - \gamma \mathbf{J}^{-1} \mathbf{F}(A_i, \nu_{0i}, \alpha_i), \tag{3.23}$$

where $\mathbf{J}$ is the Jacobian matrix (the inverse of which is sometimes called the Hessian) defined as,

$$\mathbf{J} = \begin{bmatrix} \frac{\partial g_1}{\partial A} & \frac{\partial g_1}{\partial \nu_0} & \frac{\partial g_1}{\partial \alpha} \\ \frac{\partial g_2}{\partial A} & \frac{\partial g_2}{\partial \nu_0} & \frac{\partial g_2}{\partial \alpha} \\ \frac{\partial g_3}{\partial A} & \frac{\partial g_3}{\partial \nu_0} & \frac{\partial g_3}{\partial \alpha} \end{bmatrix}, \tag{3.24}$$

and $\mathbf{F}(A, \nu_0, \alpha)$ is the gradient vector defined as,

$$\mathbf{F}(A, \nu_0, \alpha) = \begin{bmatrix} \frac{\partial \ln \mathcal{L}}{\partial A} \\ \frac{\partial \ln \mathcal{L}}{\partial \nu_0} \\ \frac{\partial \ln \mathcal{L}}{\partial \alpha} \end{bmatrix} = \begin{bmatrix} g_1(A, \nu_0, \alpha) \\ g_2(A, \nu_0, \alpha) \\ g_3(A, \nu_0, \alpha) \end{bmatrix} \tag{3.25}$$

3.4 Monte-Carlo simulations [Schervish (1996)]

Monte Carlo simulation is a computational technique used to estimate the possible outcomes of a process or system by generating random variables. Named after the Monte Carlo casino due to its reliance on randomness, the method is particularly useful for us as it deals well with problems having uncertain parameters or systems that are easier to solve numerically.

In general, Monte Carlo simulation involves modeling the system or in other words defining a mathematical model of the system. We use that model to generate random samples using a random number generators. After running the simulation multiple times in order to generate different realizations of the data, we can analyze the results and calculate probabilities, averages, or other relevant statistics that help us understand the problem.

3.5 Fisher information Matrix

The Fisher information matrix is a key concept in statistics and information theory, used to quantify the amount of information that a sample provides about unknown parameters of a statistical model. It is derived from the likelihood function of the model and plays a crucial role in the theory of estimation, particularly in maximum likelihood estimation (MLE) [Fisher (1925)]. The Fisher information matrix is defined as being square matrix where each element corresponds to the second derivative of the log-likelihood function with respect to the parameters.

$$\mathbf{I}(\boldsymbol{\theta})_{ij} = -\mathbb{E}\left[\frac{\partial^2 \ln p(\mathbf{x} \mid \boldsymbol{\theta})}{\partial\theta_i \partial\theta_j}\right] \tag{3.26}$$

It measures the sensitivity of the likelihood to a change in the parameters and it assess the precision of parameter estimates. In our case this formula can be represented as,

$$\mathbf{I}(\boldsymbol{\theta})_{ij} = -\mathbb{E}\left[\frac{\partial^2 \mathcal{L}}{\partial\theta_i \partial\theta_j}\right], \tag{3.27}$$

where $\mathcal{L}$ is defined in Equation 3.15. Since the log-likelihood has 3 parameters, given by $\boldsymbol{\Theta} = \begin{bmatrix} A & \nu_0 & \alpha \end{bmatrix}^T$, the Fisher information matrix is going to be a 3x3 matrix defined as,

$$\mathbf{I}(\boldsymbol{\theta}) = \begin{bmatrix} -\mathbb{E}\left[\frac{\partial^2 \mathcal{L}}{\partial A^2}\right] & -\mathbb{E}\left[\frac{\partial^2 \mathcal{L}}{\partial A \partial \nu_0}\right] & -\mathbb{E}\left[\frac{\partial^2 \mathcal{L}}{\partial A \partial \alpha}\right] \\ -\mathbb{E}\left[\frac{\partial^2 \mathcal{L}}{\partial \nu_0 \partial A}\right] & -\mathbb{E}\left[\frac{\partial^2 \mathcal{L}}{\partial \nu_0^2}\right] & -\mathbb{E}\left[\frac{\partial^2 \mathcal{L}}{\partial \nu_0 \partial \alpha}\right] \\ -\mathbb{E}\left[\frac{\partial^2 \mathcal{L}}{\partial \alpha \partial A}\right] & -\mathbb{E}\left[\frac{\partial^2 \mathcal{L}}{\partial \alpha \partial \nu_0}\right] & -\mathbb{E}\left[\frac{\partial^2 \mathcal{L}}{\partial^2 \alpha}\right] \end{bmatrix} \tag{3.28}$$

It is useful to calculate the second partial derivatives to see how the expectation values behave.

$$\begin{aligned} I_{i,i} &= -\mathbb{E}\left[\frac{\partial^2 \mathcal{L}}{\partial\theta^2}\right] \\ &= -\mathbb{E}\left[\frac{\partial^2}{\partial\theta^2}\left(-\frac{N}{2}(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{n=0}^{N-1}(x[n]-s)^2\right)\right] \\ &= -\mathbb{E}\left[\sum_{n=0}^{N-1}\frac{(x[n]-s)\frac{\partial^2 s}{\partial\theta^2} - \frac{\partial s}{\partial\theta}\frac{\partial s}{\partial\theta}}{\sigma^2}\right] \end{aligned} \tag{3.29}$$

From Equation 2.1 we can deduce that the expectation value of $\mathbf{x}$ is equal to $\mathbf{s}$ since that of the noise is 0,

$$\mathbb{E}[\mathbf{x}] = \mathbb{E}[\mathbf{s} + \mathbf{w}] = \mathbf{s} \tag{3.30}$$

This implies that the expectation value of $\mathbb{E}[(x[n] - s)]$ goes to 0,

$$\mathbb{E}[(x[n] - s)] = 0 \tag{3.31}$$

Meaning that the $I_{i,i}$ is equal to,

$$I_{i,i} = \frac{1}{\sigma^2}\sum_{n=0}^{N-1}\frac{\partial s}{\partial\theta}\frac{\partial s}{\partial\theta} \tag{3.32}$$

Thus the diagonal entries of the Fisher information matrix are given by the squared partial derivative w.r.t. the parameter θ. In our case, these are

$$I_{2,2} = \frac{1}{\sigma^2}\sum_{n=0}^{N-1}\frac{\partial s}{\partial A}\frac{\partial s}{\partial A} \tag{3.33}$$

$$I_{2,2} = \frac{1}{\sigma^2}\sum_{n=0}^{N-1}\frac{\partial s}{\partial \nu_0}\frac{\partial s}{\partial \nu_0} \tag{3.34}$$

$$I_{3,3} = \frac{1}{\sigma^2}\sum_{n=0}^{N-1}\frac{\partial s}{\partial \alpha}\frac{\partial s}{\partial \alpha} \tag{3.35}$$

Examining the expectation values of the mixed partials we have,

$$\begin{aligned} I_{i,j} &= -\mathbb{E}\left[\frac{\partial^2 \mathcal{L}}{\partial\theta_i \partial\theta_j}\right] \\ &= -\mathbb{E}\left[\frac{\partial^2}{\partial\theta_i \partial\theta_j}\left(-\frac{N}{2}(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{n=0}^{N-1}(x[n]-s)^2\right)\right] \\ &= -\mathbb{E}\left[\sum_{n=0}^{N-1}\frac{\frac{\partial^2 s}{\partial\theta_1^2}(x-s[n]) - \frac{\partial s}{\partial\theta_i}\frac{\partial s}{\partial\theta_j}}{\sigma^2}\right] \end{aligned} \tag{3.36}$$

Again using the fact that every expectation involving $\mathbb{E}\left[(x[n]-s)\right]$ goes to 0, we are left with:

$$I_{i,j} = \frac{1}{\sigma^2}\sum_{n=0}^{N-1}\frac{\partial s}{\partial\theta_i}\frac{\partial s}{\partial\theta_j} \tag{3.37}$$

Thus the offdiagonal entries of the Fisher information matrix are given by the product of the mixed partial derivative of our model. In our case, these are:

$$I_{1,2} = \frac{1}{\sigma^2}\sum_{n=0}^{N-1}\frac{\partial s}{\partial A}\frac{\partial s}{\partial \nu_0} \tag{3.38}$$

$$I_{1,3} = \frac{1}{\sigma^2}\sum_{n=0}^{N-1}\frac{\partial s}{\partial A}\frac{\partial s}{\partial \alpha} \tag{3.39}$$

$$I_{2,1} = \frac{1}{\sigma^2}\sum_{n=0}^{N-1}\frac{\partial s}{\partial \nu_0}\frac{\partial s}{\partial A} \tag{3.40}$$

$$I_{2,3} = \frac{1}{\sigma^2}\sum_{n=0}^{N-1}\frac{\partial s}{\partial \nu_0}\frac{\partial s}{\partial \alpha} \tag{3.41}$$

$$I_{3,1} = \frac{1}{\sigma^2}\sum_{n=0}^{N-1}\frac{\partial s}{\partial \alpha}\frac{\partial s}{\partial A} \tag{3.42}$$

$$I_{3,2} = \frac{1}{\sigma^2}\sum_{n=0}^{N-1}\frac{\partial s}{\partial \alpha}\frac{\partial s}{\partial \nu_0} \tag{3.43}$$

And the fisher information matrix is given by:

$$I = \frac{N}{\sigma^2}\begin{bmatrix} \frac{\partial s}{\partial A}\frac{\partial s}{\partial A} & \frac{\partial s}{\partial A}\frac{\partial s}{\partial \nu_0} & \frac{\partial s}{\partial A}\frac{\partial s}{\partial \alpha} \\ \frac{\partial s}{\partial \nu_0}\frac{\partial s}{\partial A} & \frac{\partial s}{\partial \nu_0}\frac{\partial s}{\partial \nu_0} & \frac{\partial s}{\partial A}\frac{\partial s}{\partial \alpha} \\ \frac{\partial s}{\partial \alpha}\frac{\partial s}{\partial A} & \frac{\partial s}{\partial \alpha}\frac{\partial s}{\partial \nu_0} & \frac{\partial s}{\partial \alpha}\frac{\partial s}{\partial \alpha} \end{bmatrix} \tag{3.44}$$

where the symbols have their usual meaning.

3.6 Cramér–Rao lower bound (CRLB) [Rao (1945)]

Using the vector parameter that we have defined before $\mathbf{\Theta} = \begin{bmatrix} A & \nu_0 & \alpha \end{bmatrix}^T$, and considering the fact that the MLE is asymptotically unbiased we can use the Fisher information matrix to get the CRLB of the estimator.

The Cramér–Rao lower bound is a theoretical lower bound on the variance of any unbiased estimator. It provides a way to evaluate the efficiency of an estimator. If an estimator reaches the CRLB, it is considered efficient, meaning it has the minimum possible variance among all unbiased estimators for that parameter and is in a sense the best estimator we can construct in the light of the data and the model.

For any vector transformation $\psi(\mathbf{\Theta})$ that is equal to $\mathbf{\Theta}$,

$$\psi(\mathbf{\Theta}) = \mathbf{\Theta}, \tag{3.45}$$

the CRLB is defined as being the diagonals of the inverse Fisher Information matrix,

$$[C(\mathbf{\Theta})] \geq [\mathbf{I}^{-1}(\mathbf{\Theta})]_{i,i} \tag{3.46}$$

3.7 Markov chain Monte Carlo (MCMC)

Markov Chain Monte Carlo (MCMC) is a family of sampling algorithms. They are used to sample from a usually complex probability distributions, especially when direct sampling is not possible. This class of algorithms are used across many different fields for its applicability.

Markov Chain

The Markov chain in MCMC is a stochastic process where a transition to a new state depends on knowledge only on the current state, this is the so called "Markov property". The sequence of states forms a chain that explores the target probability distribution. In the simplest case accept/reject methods are used to define transition probabilities between states.

Monte Carlo

Monte Carlo, named after the famous casino refers to the use of randomness to solve in these algorigms, usually involving a random number generator of some sorts. In MCMC, random sampling is used in attempt to approximate a target distribution.

Metropolis–Hastings

Now that we have discussed what the general MCMC algorithm has lets focus on the Metropolis–Hastings algorithm. This is an MCMC method that exploits the Markov property when proposing a new sample and then decides to accept or reject it based on the target distribution's value.

It works by using a so called proposal distribution, that defines how new samples are generated, often a Gaussian or uniform distribution. It utilizes an acceptance ratio that favors higher-density regions while occasionally exploring lower-density ones. It is very effective for high-dimensional distributions, avoiding the "curse of dimensionality" that plagues rejection sampling [Bellman (1961)]. It downside is its low effective sample size, that usually requires a Burn-in Period as initial samples may not represent the target distribution and need to be removed.

Algorithm 3.1 Metropolis-Hastings Algorithm (Pseudocode) [Hastings (1970)]

Input:
Target distribution $P(x)$ (proportional to $f(x)$)
Initial state x_0
Number of iterations N
Proposal distribution $g(x' \mid x)$
Output: Sequence of samples $\{x_t\}_{t=1}^{N}$
Initialize x_0
for $t = 1, 2, \ldots, N$ **do**
  Propose a new candidate $x' \sim g(x' \mid x_t)$
  Compute acceptance ratio:
  $\alpha = \min\left(1, \frac{f(x')}{f(x_t)} \cdot \frac{g(x_t|x')}{g(x'|x_t)}\right)$
  Draw $u \sim \text{Uniform}(0, 1)$
  if $u \leq \alpha$ **then**
    Accept $x_{t+1} = x'$
  else
    Reject $x_{t+1} = x_t$
  end if
end for=0

4 Results

4.1 Finding the MLE using Newton-Raphson

In Section 3.2, we defined the likelihood function, and in Section 3.3, we introduced the Maximum Likelihood Estimator (MLE). We are now fully equipped to perform our experiment. Using the Newton-

Raphson method, we aim to find a minimum in the space of possible estimates. The choice of the starting point is crucial, as the algorithm can only find a local minimum—whether it is global or not is not trivial to determine. For our search, we set the starting point as $\boldsymbol{\Theta} = \begin{bmatrix} 6 & 2 & 1 \end{bmatrix}^T$.

To better understand the optimization process, it is helpful to plot the gradient norm. This allows us to observe how the norm of the parameter vector changes at each step, giving us insight into the landscape of the space. Below is a graph illustrating this change.

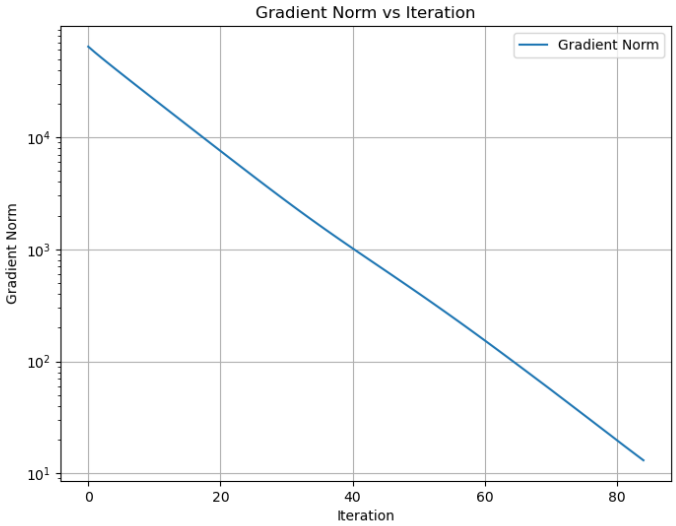


Figure 4.1: The L2 norm of the gradient vector at each step of the Newton-Raphson optimization algorithm.

We observe a steady decrease in the gradient norm, which indicates that we are approaching a minimum. The convergence is smooth, with a well-behaved gradient, and we reached the convergence threshold after about 80 iterations. The estimates for each step are shown in the following figure.

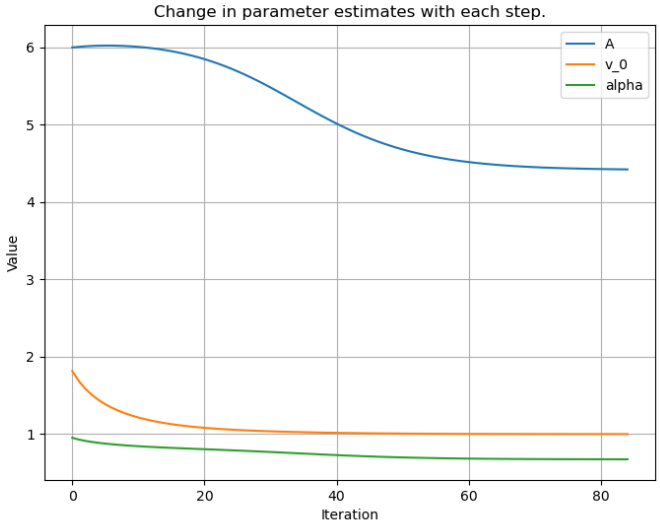


Figure 4.2: The values of the estimates at each step of the Newton-Raphson algorithm.

After approximately 80 iterations, the parameter estimates converge. These final values are presented in the table below, which we will use to fit the model to the data.

Parameter	Value
A	4.424
v_0	1.001
α	0.675

Table 4.1: Parameter estimates from the Newton-Raphson algorithm.

The following plot shows the fitted model using the parameter estimates from Table 4.1 alongside the data points used for estimation.

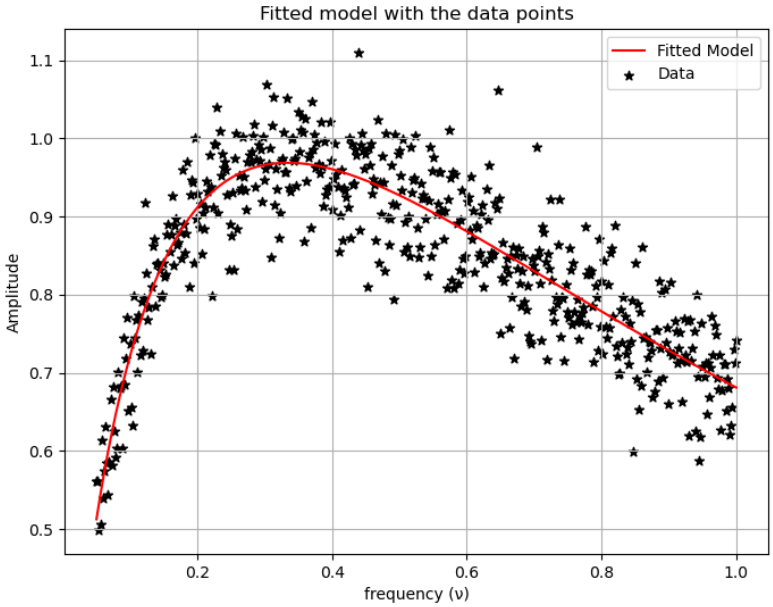


Figure 4.3: A plot of the fitted model using the parameter estimates from Table 4.1 and the data points.

As expected, the model does not perfectly fit every data point due to inherent noise (as the data are IID with noise), but the fit is quite good overall. To assess the noise distribution, we examine the residuals between the fitted model and the data points.

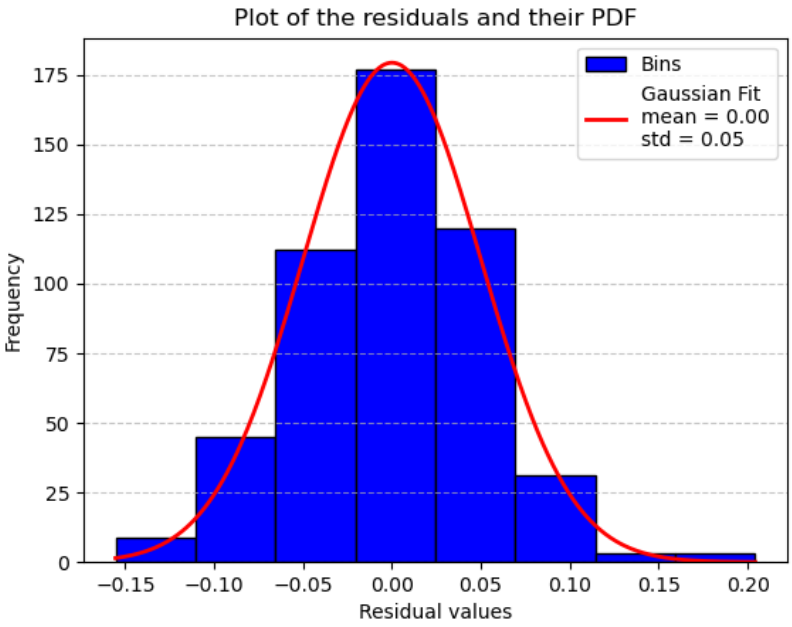


Figure 4.4: Histogram of the residuals between the fitted model and the data points, with a scaled Gaussian distribution overlaid in red.

The histogram above shows that the residuals are normally distributed around the fitted model, with a mean of $\mu = 0$ and a standard deviation of $\sigma = 0.05$. This confirms our expectations based on the assumptions outlined in Section 2.2.

4.2 Monte-Carlo (MC) simulations

We ran 10,000 Monte-Carlo simulations of the data by adding different realizations of noise, sampled from the normal distribution with 0 mean and 0.05 standard deviation $\mathcal{N}(\mu = 0, \sigma = 0.05)$ (a 3D representation of the first 5 realizations can be seen in the Appendix). We followed the same procedure as described in the section above on all different realizations and we logged the estimates during the optimization process, a plot of which can be seen bellow.

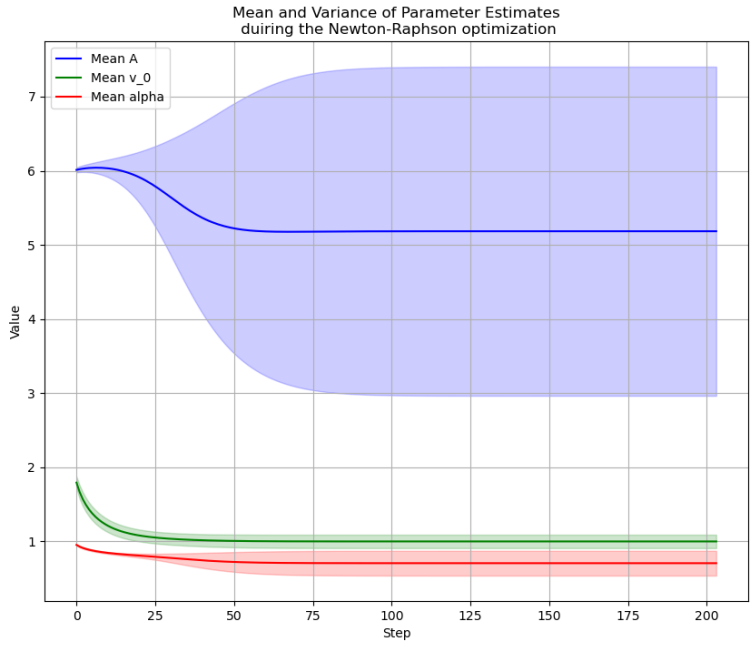
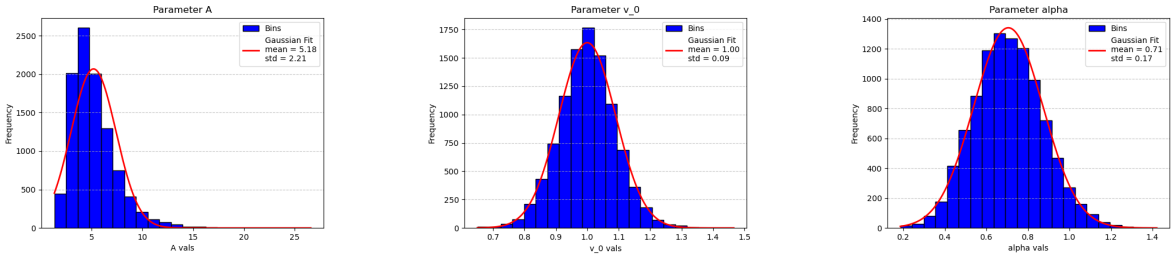


Figure 4.5: The top figure illustrates the estimated values at each step of the Newton-Raphson algorithm across all realizations. The solid line represents the mean estimate for the parameter, while the shaded region indicates the range of one standard deviation from the mean, calculated over all simulations of the data.

Different realizations of the data produced slightly varying estimates, which is expected due to the inherent randomness of the Monte Carlo (MC) simulations. This variability provides insight into the probability density functions (PDFs) of the parameters. By analyzing the frequency distribution of the estimates, we observed that a Gaussian distribution accurately described the parameters. Consequently, we fitted a Gaussian distribution to each parameter. The mean and the variance of the estimates can be seen in Table 4.2



(a) The PDF of A **(b) The PDF of ν_0** **(c) The PDF of α**

Figure 4.6: The PDFs for the estimated parameters.

Estimator	Mean	Variance
A	5.182	4.905
v_0	1.000	0.008
α	0.706	0.028

Table 4.2: Summary statistics for the estimators.

4.3 The Fisher information matrix and the CRLB

We derived the Fisher information matrix for our parameters in Section 3.5 and we have numerically calculated its value to be,

$$\mathbf{I}(\boldsymbol{\Theta}) = \begin{bmatrix} 6354.8536 & 5927.7900 & -79687.1140 \\ 5927.7900 & 23554.4935 & -77389.1547 \\ -79687.1140 & -77389.1547 & 1004023.3269 \end{bmatrix} \tag{4.1}$$

Using Equation 3.46 we can calculate the CRLB of our estimates and compare them to the ones calculated from the MC simulations. A summary of the result can be found in the table below.

Parameter	Variance (CRLB)	Variance (MC)
A	3.618×10^{-2}	490.521×10^{-2}
v_0	6.223×10^{-5}	820.852×10^{-5}
α	2.346×10^{-4}	279.893×10^{-4}

Table 4.3: Comparison of CRLB and calculated values for parameters.

4.4 Compute the likelihood function

Looking at the problem in the Bayesian way, we can find the probability distribution function of our parameters by evaluating the Likelihood function as defined in Section 3.2. Bayes' formula, perhaps one of the most important concepts in statistics, is given by the following:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)} \tag{4.2}$$

where $P(A \mid B)$ is the probability of the data A given the hypothesis B, $P(B \mid A)$ is the probability of the hypothesis given the data, $P(B)$ is the prior, and $P(A)$ is the evidence. In our case, this simplifies to:

$$P(\mathbf{x} \mid \boldsymbol{\Theta}) = \frac{P(\boldsymbol{\Theta} \mid \mathbf{x})P(\mathbf{x})}{P(\boldsymbol{\Theta})}. \tag{4.3}$$

Considering that the likelihood function represents the probability of the hypothesis given the data, we now have a relation between the PDF of the data, referred to as the posterior distribution, and the likelihood function:

$$P(\mathbf{x} \mid \boldsymbol{\Theta}) \propto \mathcal{L}, \tag{4.4}$$

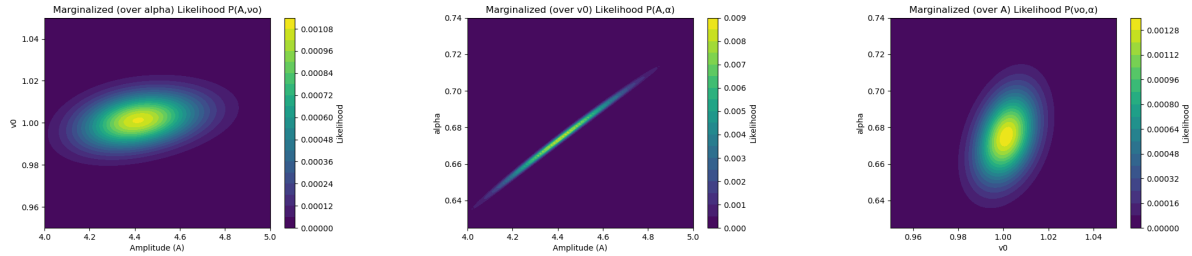
assuming a flat prior. We computed the log-likelihood function on a $100 \times 100 \times 100$ grid with the values for A, ν_0, and α generated within the following ranges:

$$4 \leq A \leq 5$$

$$0.95 \leq \nu_0 \leq 1.05$$

$$0.625 \leq \alpha \leq 0.740$$

Since our vector $\boldsymbol{\Theta}$ is three-dimensional, the expression above gives us a 3D posterior, which is not very easy to visualize. However, because probabilities must sum to one by definition, we can marginalize the PDF over each parameter to reduce the dimensionality. This allows for easier visualization. The marginalized 2D posterior distributions are shown in the contour plots below.

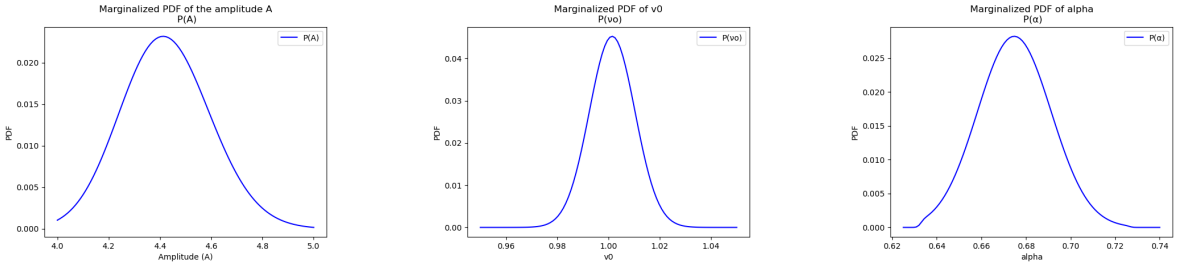


(a) The posterior of $P(A, \nu_0)$ **(b) The posterior of $P(A, \alpha)$** **(c) The posterior of $P(\nu_0, \alpha)$**

Figure 4.7: The posterior of each possible combination of paired parameters.

We can observe that the resulting 2D posterior resembles a 2D Gaussian, which further validates the results from the previous analysis. There is also correlation between each pair of parameters, as indicated by the tilt of the major and minor axes of the 2D Gaussian features.

We can further marginalize once more to obtain the marginal PDFs of each of the parameters of interest, which are shown in the figure below.



(a) The posterior of $P(A)$ **(b) The posterior of $P(\nu_0)$** **(c) The posterior of $P(\alpha)$**

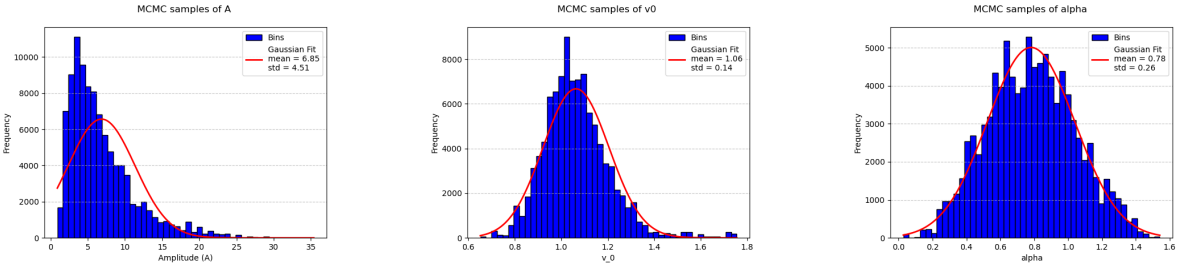
Figure 4.8: The marginalized posteriors of each parameter.

4.5 Metropolis Hastings (MCMC) samples

Earlier, we described the mechanics of the Metropolis-Hastings algorithm. Here, we apply it to sample values of our estimates. We use a 3D multivariate normal distribution as our proposal distribution, with a covariance matrix that has diagonal elements equal to three times the variance derived from the Monte Carlo (MC) simulations and off-diagonal elements set to zero. The mean of the proposal distribution is set to the last accepted sample.

From our previous analysis, we know that the distributions of these parameters are approximately Gaussian. Therefore, this proposal distribution is well-suited to cover the full range of possible samples. Over time, it is expected to converge to the target distribution.

The following results summarize the outcome of $100,000$ samples generated using the Metropolis-Hastings algorithm, along with an analysis of the sampling procedure.



(a) The sampled values of A **(b) The sampled values of ν_0** **(c) The sampled values of α**

Figure 4.9: The sampled values of each parameter by the Metropolis Hastings algorithm.

MCMC - Metropolis Hastings Analytics	
Number of accepted samples	3,018
Number of total samples	100,000
Acceptance rate	3.018%
Burn-In	1,000
Samples length	99,000

Table 4.4: MCMC - Metropolis Hastings Analytics Results

5 Discussion

The analysis conducted in this study highlights the effectiveness of statistical techniques for estimating parameters of a signal embedded in white Gaussian noise (WGN). Using the Maximum Likelihood Estimation (MLE) method, implemented through the Newton-Raphson algorithm, we successfully derived parameter estimates that demonstrated convergence within 80 iterations.

We used Monte Carlo (MC) simulations to assess the variability of these estimates. By generating 10,000 realizations of the data, we observed that the distributions of the parameters were approximately Gaussian.

We also examined the Fisher Information Matrix to compute the Cramér-Rao Lower Bound (CRLB), which offers a theoretical benchmark for the efficiency of our estimators. When comparing the variances obtained from MC simulations to the CRLB-derived variances, notable discrepancies were observed—particularly for the $\hat{A}$ estimator. These differences could be attributed to finite sample effects or potential biases introduced during the simulations. As the MLE is asymptotically efficient, achieving unbiased estimates for strongly correlated variables like A and α might require a substantially larger dataset.

Bayesian analysis, conducted by evaluating the log-likelihood function, provided a complementary perspective by generating posterior distributions of the parameters. The contour plots of the 2D posteriors revealed significant correlations—especially between A and α. This finding supports the hypothesis that the discrepancies between observed variances and the CRLB are due to the estimator's bias within the regime studied. Since the CRLB assumes unbiased estimator, this condition may not hold in our specific setup.

Finally, the Metropolis-Hastings algorithm, a Markov Chain Monte Carlo (MCMC) method, was used to sample parameter estimates. The MCMC results confirmed the findings from previous analyses.

6 Conclusion

The Newton-Raphson algorithm provided an estimates for the model parameters of,

$$\boldsymbol{\Theta} = \begin{bmatrix} A & \nu_0 & \alpha \end{bmatrix}^T = \begin{bmatrix} 4.424 & 1.001 & 0.675 \end{bmatrix}^T$$

after approximately 80 iterations. The convergence was smooth, indicating a robust optimization process.

Monte Carlo simulations confirmed the Gaussian nature of parameter distributions. The corresponding distributions of the estimators were as follows,

$$\hat{A} \sim \mathcal{N}(\mu = 5.182, \text{var} = 4.905)$$

$$\hat{\nu_0} \sim \mathcal{N}(\mu = 1.000, \text{var} = 0.008)$$

$$\hat{\alpha} \sim \mathcal{N}(\mu = 0.706, \text{var} = 0.028)$$

The Fisher Information Matrix and CRLB provided theoretical variances significantly smaller than those observed in simulations. This Fisher Information matrix was numerically computed to be,

$$\mathbf{I}(\boldsymbol{\Theta}) = \begin{bmatrix} 6354.8536 & 5927.7900 & -79687.1140 \\ 5927.7900 & 23554.4935 & -77389.1547 \\ -79687.1140 & -77389.1547 & 1004023.3269 \end{bmatrix}$$

By taking the inverse of this matrix we found the CRLB of our estimates to be,

$$\text{Var}(\hat{A}) \geq CRLB(\hat{A}) = 3.618 \times 10^{-2}$$

$$\text{Var}(\hat{\nu_0}) \geq CRLB(\hat{\nu_0}) = 6.223 \times 10^{-5}$$
$$\text{Var}(\alpha) \geq CRLB(\alpha) = 2.346 \times 10^{-4}$$

which gives us an inside into the theoretical efficiency of the MLE under ideal conditions ($N_{samples} \to \infty$) and the effect of the sample size on that variance.

Metroplis Hastings (MCMC) sampling produced posterior distributions that further reinforced the Gaussian nature of the parmeter estimates. With acceptance rate of 3.018% in the algorithm with 3,018 accepted samples from the 100,000 total.

The results collectively validate the findings with showing very consistently the same results.

Future research could explore extensions to more robust sampling techniques in order to improve efficiency, or alternative optimization strategies to atempt to reduce computational costs while maintaining accuracy.

References

Bellman, R. E. (1961). *Dynamic Programming*. Princeton University Press, Princeton, NJ. The concept of the "curse of dimensionality" was introduced in this work.

Burden, R. L. and Faires, J. D. (2005). *Numerical Analysis*. Brooks/Cole, Cengage Learning, 8th edition.

Fisher, R. A. (1925). *Theory of Statistical Estimation*, volume 22. Cambridge University Press, Cambridge.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109.

Rao, C. R. (1945). *Information and the Accuracy Attainable in the Estimation of Statistical Parameters*, volume 37.

Schervish, L. S. S. P. P. L. P. (1996). *Introduction to Monte Carlo Methods*. Springer, New York.

A Appendix

A.1 Github Repository

B Appendix

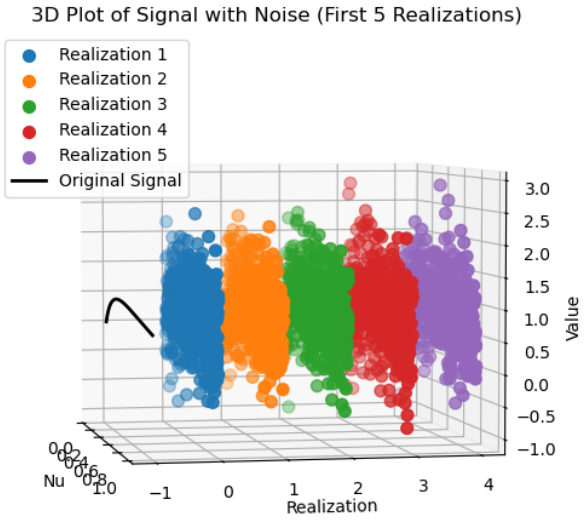


Figure B.1: A representation of the first 5 noise realiaztions from the MC simulations.