Estimating the Mean Parameter of a Normal Distribution and Constructing a Confidence Interval

Gbenga Agunbiade[1]

[1]Department of Physics and Astronomy, University of Kansas, Lawrence, KS 66045

gsagunbiade@ku.edu

Abstract. *This project simulates an experiment where data is generated from a normal distribution with a known mean ('true mean') and standard deviation ('true std'). It then estimates the mean parameter using maximum likelihood estimation based on the generated data. The histogram plot shows the frequency of different parameter estimates. It provides an overview of the estimated values and their distribution. The true distribution with the confidence interval visualizes how well the estimated parameter captures the true distribution. If the confidence interval includes the true mean, it suggests that the estimation procedure is effective in capturing the true parameter value. If we change the parameters such as 'true mean', 'true std', 'num samples', and 'num experiments', we can observe how the estimates and confidence intervals change.*

1. Introduction

In probability theory and statistics, the normal distribution is a continuous probability distribution that describes the probability distribution of a random variable that has a bell-shaped probability density function. It is also known as the Gaussian distribution or the bell curve. The normal distribution is a very important probability distribution in statistics because it is the most common distribution that arises in many natural phenomena, such as heights, weights, and test scores. The normal distribution is characterized by two parameters: the mean and the standard deviation. The mean determines the location of the center of the distribution, while the standard deviation determines the spread or the width of the distribution. The normal distribution has widespread applications in various fields, including statistics, finance, physics, and social sciences. It is often used as an approximation for real-world phenomena and provides a useful framework for modeling and analyzing data. Python is usually used to develop software and websites and analyze data, task automation, and data visualization. Since it's relatively easy to learn, Python has been adopted by many non-programmers such as accountants and scientists, for a variety of everyday tasks, like organizing finances. There are several ways programmers demonstrate how to use Python to simulate different experiments. This project provides Python code for normal distribution using libraries such as NumPy and SciPy. These libraries provide functions for generating random samples from the normal distribution, calculating PDF, and performing statistical operations related to the distribution.

2. Implementation, Results and Discussion

This code simulates an experiment to estimate the parameter (mean) of a normal distribution using maximum likelihood estimation and construct a confidence interval. The code begins with importing the necessary libraries: NumPy for numerical computations,

Matplotlib for plotting, and the norm class from SciPy's stats module to work with the normal distribution. The 'simulate experiment' function generates a sample of random values from a normal distribution with a specified mean ('true mean'), standard deviation ('true std'), and number of samples ('num samples'). It uses NumPy's 'random.normal' function to generate the samples. The 'likelihood' function calculates the likelihood of the mean given the observed data ('samples'). It uses SciPy's 'norm.pdf' function to compute the probability density function (PDF) of the samples, assuming a normal distribution with mean ('mean') and standard deviation ('std').

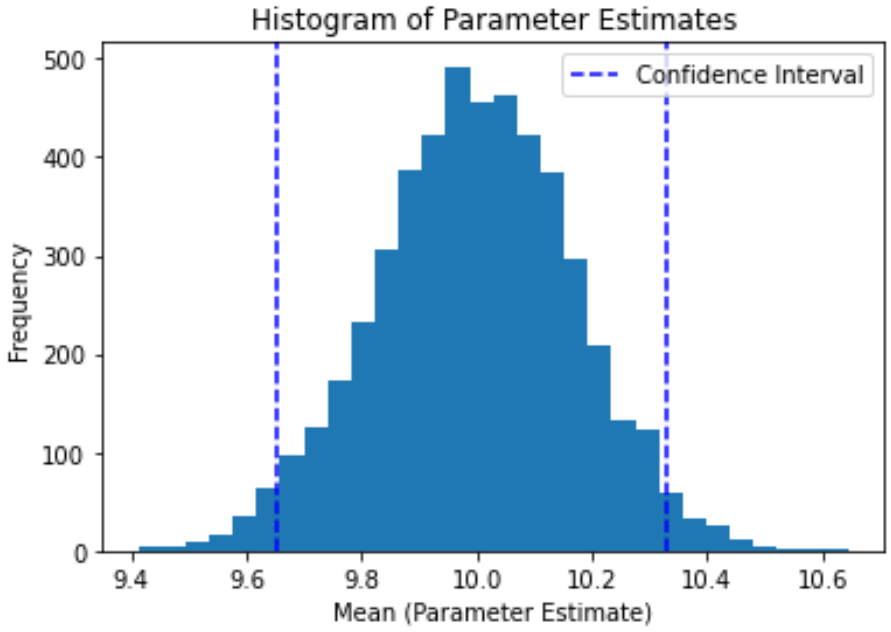

Figura 1. The histogram of the parameter estimates using 'plt.hist'. It also adds vertical dashed lines at the lower and upper bounds to indicate the confidence interval

The likelihood is calculated as the product of the PDF values for each sample. The 'estimate parameter' function estimates the parameter (mean) using the maximum likelihood estimation. It computes the mean of the given 'samples' using NumPy's 'mean' function and returns the estimated value. The code specifies the true values of the mean ('true mean'), standard deviation ('true std'), number of samples ('num samples'), and the number of experiments ('num experiments'). The code performs a loop for 'num experiments' iterations. In each iteration, it simulates an experiment by generating a sample of 'num samples' random values from a normal distribution with the true mean and standard deviation. It then estimates the mean using the 'estimate parameter' function and appends the estimate to the 'estimates' list. After the loop, the code calculates the confidence interval using the 'np.percentile' function on the 'estimates' list. The 'confidence level' variable specifies the desired confidence level (e.g., 0.95 confidence interval). The code plots a histogram of the parameter estimates ('estimates') using Matplotlib's 'hist' function as shown in Figure 1. It also adds vertical dashed lines to represent the lower and upper bounds of the confidence interval. The code then plots the true probability density function (PDF) of the distribution along with the confidence interval shaded in red as shown in Figure 2. It uses the 'np.linspace' function to generate a range of x-values, and

the ‘norm.pdf‘ function to compute the corresponding y-values.

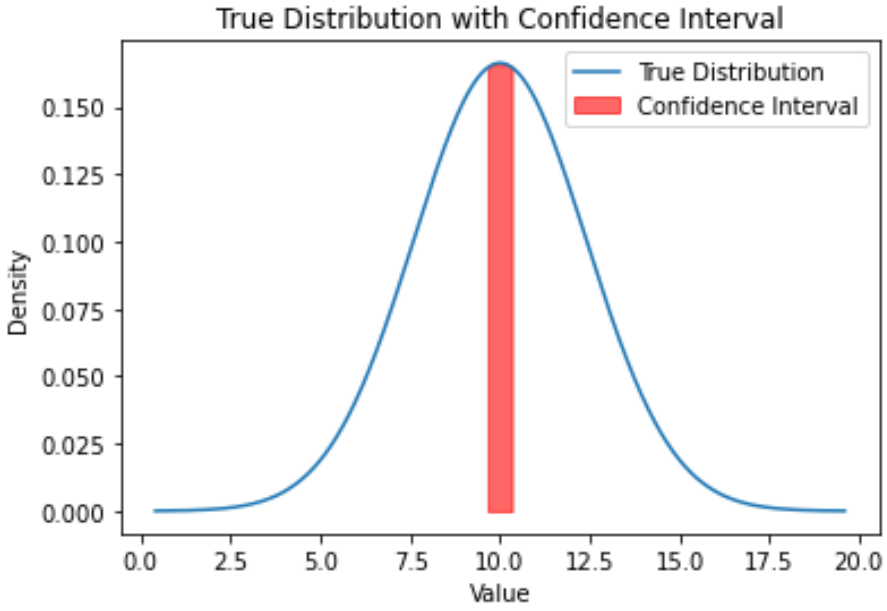

Figura 2. The true probability density function (PDF) of a normal distribution along with the confidence interval shaded in red

3. Interpretation

The histogram of parameter estimates provides an overview of the distribution of the estimated mean parameters obtained from the experiments. It shows the frequency or count of different estimated mean values. A peak or cluster around the true mean indicates that the estimation method is accurate. The width and shape of the histogram provide insights into the precision and uncertainty of the estimates. The vertical dashed lines in the histogram represent the confidence interval. They indicate the range of values within which the true mean is expected to fall with a certain level of confidence (in this case, 0.95). If the interval is narrow, it suggests a more precise estimate. The plot of the true distribution with the confidence interval provides a visual representation of the estimated mean parameter. The shaded region highlights the area where the true mean is likely to lie based on the data and the chosen confidence level. If the confidence interval is narrow, it suggests a more precise estimate. Finally, by varying the parameters such as ‘true mean‘, ‘true std‘, ‘num samples‘, and ‘num experiments‘, we can observe how the estimates and confidence intervals change. For example, increasing the number of samples generally leads to narrower confidence intervals and more accurate parameter estimation.

4. Conclusion

Overall, this code provides an example of how to simulate an experiment, estimate a parameter of interest, and calculate a confidence interval for the parameter estimate. It demonstrates the usefulness of simulation experiments in evaluating the performance of statistical methods.