2009-11-13

Parameter Estimation for the Lognormal Distribution

Brenda Faith Ginos
Brigham Young University - Provo

Parameter Estimation for the Lognormal

Distribution

Brenda F. Ginos

A project submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Scott D. Grimshaw, Chair
David A. Engler
G. Bruce Schaalje

Department of Statistics

Brigham Young University

December 2009

ABSTRACT

Parameter Estimation for the Lognormal Distribution

Brenda F. Ginos

Department of Statistics

Master of Science

The lognormal distribution is useful in modeling continuous random variables which are greater than or equal to zero. Example scenarios in which the lognormal distribution is used include, among many others: in medicine, latent periods of infectious diseases; in environmental science, the distribution of particles, chemicals, and organisms in the environment; in linguistics, the number of letters per word and the number of words per sentence; and in economics, age of marriage, farm size, and income.The lognormal distribution is also useful in modeling data which would be considered normally distributed except for the fact that it may be more or less skewed (Limpert, Stahel, and Abbt 2001). Appropriately estimating the parameters of the lognormal distribution is vital for the study of these and other subjects.

Depending on the values of its parameters, the lognormal distribution takes on various shapes, including a bell-curve similar to the normal distribution. This paper contains a simulation study concerning the effectiveness of various estimators for the parameters of the lognormal distribution. A comparison is made between such parameter estimators as Maximum Likelihood estimators, Method of Moments estimators, estimators by Serfling (2002), as well as estimators by Finney (1941). A simulation is conducted to determine which parameter estimators work better in various parameter combinations and sample sizes of the lognormal distribution. We find that the Maximum Likelihood and Finney estimators perform the best overall, with a preference given to Maximum Likelihood over the Finney estimators because of its vast simplicity. The Method of Moments estimators seem to perform best when σ is less than or equal to one, and the Serfling estimators are quite accurate in estimating μ but not σ in all regions studied. Finally, these parameter estimators are applied to a data set counting the number of words in each sentence for various documents, following which a review of each estimator's performance is conducted. Again, we find that the Maximum Likelihood estimators perform best for the given application, but that Serfling's estimators are preferred when outliers are present.

Keywords: Lognormal distribution, maximum likelihood, method of moments, robust estimation

ACKNOWLEDGEMENTS

Many thanks go to my wonderful husband, who kept me company while I burned the midnight oil on countless evenings during this journey. I would also like to thank my family and friends, for all of their love and support in all of my endeavors. Finally, I owe the BYU Statistics professors and faculty an immense amount of gratitude for their assistance to me during the brief but wonderful time I have spent in this department.

CONTENTS

CHAPTER

APPENDIX

TABLES

Table

FIGURES

Figure

1. THE LOGNORMAL DISTRIBUTION

1.1 Introduction

The lognormal distribution takes on both a two-parameter and three-parameter form. The density function for the two-parameter lognormal distribution is

$$f(X|\mu,\sigma^2) = \frac{1}{\sqrt{(2\pi\sigma^2)}X} \exp\left[-\frac{(\ln(X)-\mu)^2}{2\sigma^2}\right],$$

$$X > 0, -\infty < \mu < \infty, \sigma > 0. \qquad (1.1)$$

The density function for the three-parameter lognormal distribution, which is equivalent to the two-parameter lognormal distribution if X is replaced by $(X-\theta)$, is

$$f(X|\theta,\mu,\sigma^2) = \frac{1}{\sqrt{(2\pi\sigma^2)}(X-\theta)} \exp\left[-\frac{(\ln(X-\theta)-\mu)^2}{2\sigma^2}\right],$$

$$X > \theta, -\infty < \mu < \infty, \sigma > 0. \qquad (1.2)$$

Notice that, due to the nature of its contribution in the density function, θ is a location parameter which determines where to shift the three-parameter density function along the X-axis. Considering that θ's contribution to the shape of the density is null, it is not commonly used in data fitting, nor is it frequently mentioned in lognormal parameter estimation technique discussions. Thus, we will not discuss its estimation in this paper. Instead, our focus will be the two-parameter density function defined in Equation 1.1.

Due to a close relationship with the normal distribution in that $\ln(X)$ is normally distributed if X is lognormally distributed, the parameter μ from Equation 1.1 may be interpreted as the mean of the random variable's logarithm, while the parameter σ may be interpreted as the standard deviation of the random variable's logarithm. Additionally, μ is said to be a scale parameter, while σ is said to be a shape parameter of the lognormal density function. Figure 1.1 presents two plots which demonstrate the effect of changing μ

from 0 in the top panel to 1 in the bottom panel, as well as increasing σ gradually from 1/8 to 10 (Antle 1985).

The lognormal distribution is useful in modeling continuous random variables which are greater than or equal to zero. The lognormal distribution is also useful in modeling data which would be considered normally distributed except for the fact that it may be more or less skewed. Such skewness occurs frequently when means are low, variances are large, and values cannot be negative (Limpert, Stahel, and Abbt 2001). Broad areas of application of the lognormal distribution include agriculture and economics, while narrower applications include its frequent use as a model for income, wireless communications, and rainfall (Brezina 1963; Antle 1985). Appropriately estimating the parameters of the lognormal distribution is vital for the study of these and other subjects.

We present a simulation study to explore the precision and accuracy of several estimation methods for determining the parameters of lognormally distributed data. We then apply the discussed estimation methods to a data set counting the number of words in each sentence for various documents, following which we conduct a review of each estimator's performance.

1.2 Literature Review

The lognormal distribution finds its beginning in 1879. It was at this time that F. Galton noticed that if $X_1, X_2, ..., X_n$ are independent positive random variables such that

$$T_n = \prod_{i=1}^{n} X_i, \tag{1.3}$$

then the log of their product is equivalent to the sum of their logs,

$$\ln(T_n) = \sum_{i=1}^{n} \ln(X_i). \tag{1.4}$$

Due to this fact, Galton concluded that the standardized distribution of $\ln(T_n)$ would tend to a unit normal distribution as n goes to infinity, such that the limiting distribution of T_n

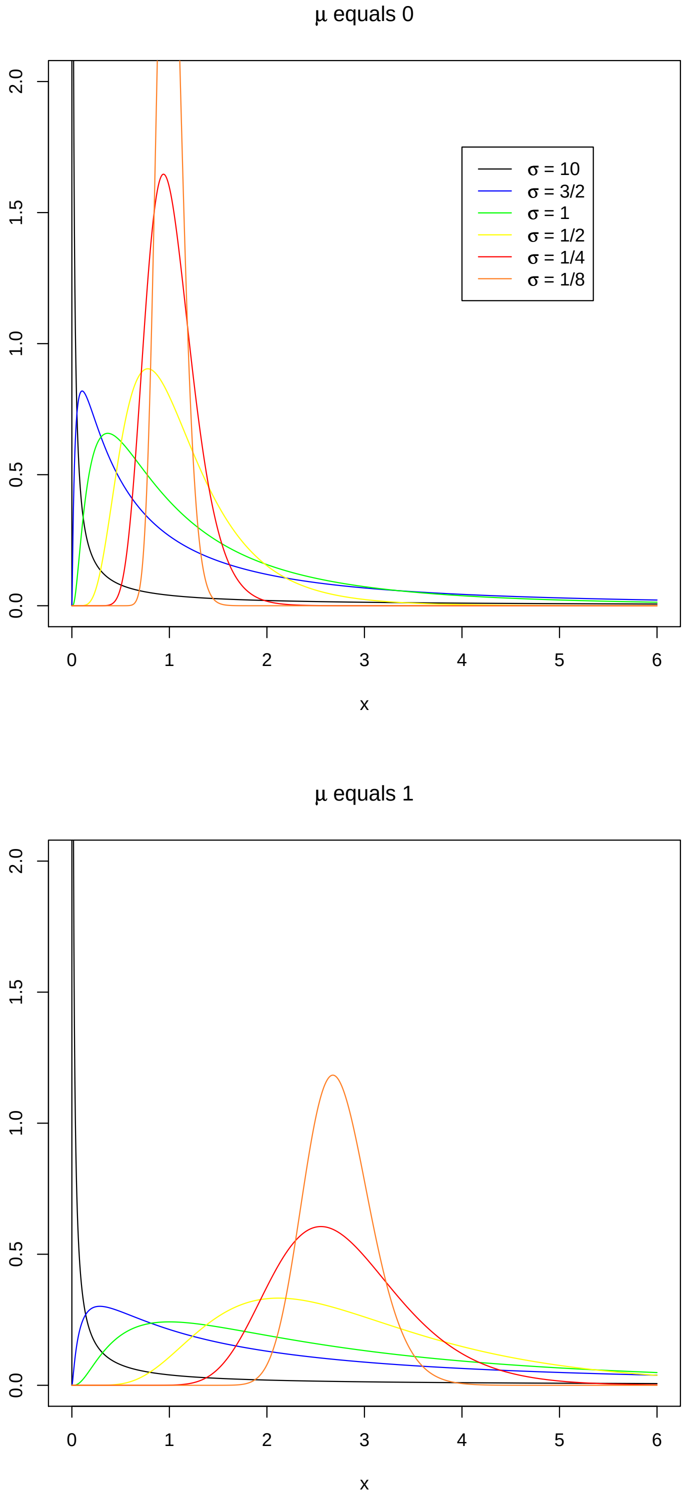

Figure 1.1: Some Lognormal Density Plots, $\mu = 0$ and $\mu = 1$.

would tend to a two-parameter lognormal, as defined in Equation 1.1. After Galton, these roots to the lognormal distribution remained virtually untouched until 1903, when Kapteyn derived the lognormal distribution as a special case of the transformed normal distribution. Note that the lognormal is sometimes called the anti-lognormal distribution, because it is not the distribution of the logarithm of a normal variable, but is instead the anti-log of a normal variable (Brezina 1963; Johnson and Kotz 1970).

1.3 Properties

An important property of the lognormal distribution is its multiplicative property. This property states that if two independent random variables, X_1 and X_2, are distributed respectively as $Lognormal(\mu_1, \sigma_1^2)$ and $Lognormal(\mu_2, \sigma_2^2)$, then the product of X_1 and X_2 is distributed as $Lognormal(\mu_1\mu_2, \sqrt{\sigma_1^2 + \sigma_2^2})$. This multiplicative property for independent lognormal random variables stems from the additive properties of normal random variables (Antle 1985).

Another important property of the lognormal distribution is the fact that for very small values of σ (e.g., less than 0.3), the lognormal is nearly indistinguishable from the normal distribution (Antle 1985). This also follows from its close ties to the normal distribution. A visual example of this property is shown in Figure 1.2.

However, unlike the normal distribution, the lognormal does not possess a moment generating function. Instead, its moments are given by the following equation defined by Casella and Berger (2002):

$$E(X^t) = \exp\left[t\mu + t^2\sigma^2/2\right]. \tag{1.5}$$

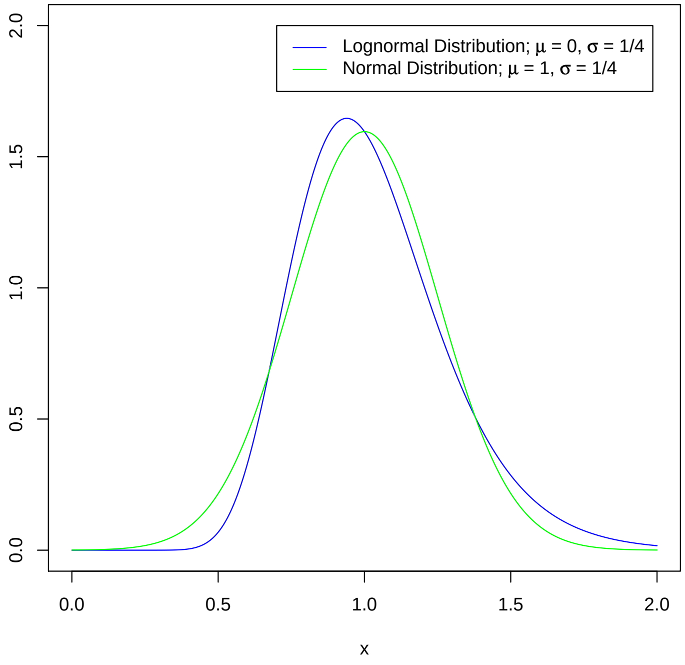

Figure 1.2: A Normal Distribution Overlaid on a Lognormal Distribution. This plot shows the similarities between the two distributions when σ is small.

2. PARAMETER ESTIMATION

The most frequent methods of parameter estimation for the lognormal distribution are Maximum Likelihood and Method of Moments. Both of these methods have convenient, closed-form solutions, which are derived in Sections 2.1 and 2.2. Other estimation techniques include those by Serfling (2002) as well as those by Finney (1941).

2.1 Maximum Likelihood Estimators

Maximum Likelihood is a popular estimation technique for many distributions because it picks the values of the distribution's parameters that make the data "more likely" than any other values of the parameters would make them. This is accomplished by maximizing the likelihood function of the parameters given the data. Some appealing features of Maximum Likelihood estimators include that they are asymptotically unbiased, in that the bias tends to zero as the sample size n increases; they are asymptotically efficient, in that they achieve the Cramer-Rao lower bound as n approaches ∞; and they are asymptotically normal.

To compute the Maximum Likelihood estimators, we start with the likelihood function. The likelihood function of the lognormal distribution for a series of X_is ($i = 1,\ 2,\ \ldots\ n$) is derived by taking the product of the probability densities of the individual X_is:

$$\begin{aligned} \mathrm{L}\left(\mu, \sigma^2 | X\right) &= \prod_{i=1}^{n} \left[f\left(X_i | \mu, \sigma^2\right)\right] \\ &= \prod_{i=1}^{n} \left(\left(2\pi\sigma^2\right)^{-1/2} X_i^{-1} \exp\left[\frac{-\left(\ln(X_i)-\mu\right)^2}{2\sigma^2} \right] \right) \\ &= \left(2\pi\sigma^2\right)^{-n/2} \prod_{i=1}^{n} X_i^{-1} \exp\left[\sum_{i=1}^{n} \frac{-\left(\ln(X_i)-\mu\right)^2}{2\sigma^2} \right]. \end{aligned} \tag{2.1}$$

The log-likelihood function of the lognormal for the series of X_is ($i = 1,\ 2,\ \ldots\ n$) is then derived by taking the natural log of the likelihood function:

$$\begin{aligned}
\mathcal{L}(\mu, \sigma^2|X) &= \ln\left(\left(2\pi\sigma^2\right)^{-n/2}\prod_{i=1}^{n} X_i^{-1}\exp\left[\sum_{i=1}^{n}\frac{-(\ln(X_i)-\mu)^2}{2\sigma^2}\right]\right)\\
&= -\frac{n}{2}\ln\left(2\pi\sigma^2\right) - \sum_{i=1}^{n}\ln(X_i) - \frac{\sum_{i=1}^{n}\left(\ln(X_i)-\mu\right)^2}{2\sigma^2}\\
&= -\frac{n}{2}\ln\left(2\pi\sigma^2\right) - \sum_{i=1}^{n}\ln(X_i) - \frac{\sum_{i=1}^{n}\left[\ln(X_i)^2 - 2\ln(X_i)\mu + \mu^2\right]}{2\sigma^2}\\
&= -\frac{n}{2}\ln\left(2\pi\sigma^2\right) - \sum_{i=1}^{n}\ln(X_i) - \frac{\sum_{i=1}^{n}\ln(X_i)^2}{2\sigma^2} + \frac{\sum_{i=1}^{n}2\ln(X_i)\mu}{2\sigma^2} - \frac{\sum_{i=1}^{n}\mu^2}{2\sigma^2}\\
&= -\frac{n}{2}\ln\left(2\pi\sigma^2\right) - \sum_{i=1}^{n}\ln(X_i) - \frac{\sum_{i=1}^{n}\ln(X_i)^2}{2\sigma^2} + \frac{\sum_{i=1}^{n}\ln(X_i)\mu}{\sigma^2} - \frac{n\mu^2}{2\sigma^2}. \qquad (2.2)
\end{aligned}$$

We now find $\hat{\mu}$ and $\hat{\sigma^2}$, which maximize $\mathcal{L}(\mu, \sigma^2|X)$. To do this, we take the gradient of $\mathcal{L}$ with respect to μ and σ^2 and set it equal to 0: with respect to μ,

$$\begin{aligned}
\frac{\delta\mathcal{L}}{\delta\mu} &= \frac{\sum_{i=1}^{n}\ln(X_i)}{\hat{\sigma}^2} - \frac{2n\hat{\mu}}{2\hat{\sigma}^2} = 0\\
&\implies \frac{n\hat{\mu}}{\hat{\sigma}^2} = \frac{\sum_{i=1}^{n}\ln(X_i)}{\hat{\sigma}^2}\\
&\implies n\hat{\mu} = \sum_{i=1}^{n}\ln(X_i)\\
&\implies \hat{\mu} = \frac{\sum_{i=1}^{n}\ln(X_i)}{n}; \qquad (2.3)
\end{aligned}$$

with respect to σ^2,

$$
\begin{aligned}
\frac{\delta\mathcal{L}}{\delta\sigma^2} &= -\frac{n}{2}\frac{1}{\hat{\sigma}^2} - \frac{\sum_{i=1}^{n}(\ln(X_i)-\hat{\mu})^2}{2}\left(-\hat{\sigma}^2\right)^{-2} \\
&= -\frac{n}{2\hat{\sigma}^2} + \frac{\sum_{i=1}^{n}(\ln(X_i)-\hat{\mu})^2}{2(\hat{\sigma}^2)^2} = 0 \\
&\Longrightarrow \frac{n}{2\hat{\sigma}^2} = \frac{\sum_{i=1}^{n}(\ln(X_i)-\hat{\mu})^2}{2\hat{\sigma}^4} \\
&\Longrightarrow n = \frac{\sum_{i=1}^{n}(\ln(X_i)-\hat{\mu})^2}{\hat{\sigma}^2} \\
&\Longrightarrow \hat{\sigma}^2 = \frac{\sum_{i=1}^{n}(\ln(X_i)-\hat{\mu})^2}{n} \\
&\Longrightarrow \hat{\sigma}^2 = \frac{\sum_{i=1}^{n}\left(\ln(X_i) - \frac{\sum_{i=1}^{n}\ln(X_i)}{n}\right)^2}{n}. \qquad (2.4)
\end{aligned}
$$

Thus, the maximum likelihood estimators are

$$
\begin{aligned}
\hat{\mu} &= \frac{\sum_{i=1}^{n}\ln(X_i)}{n} \text{ and} \\
\hat{\sigma}^2 &= \frac{\sum_{i=1}^{n}\left(\ln(X_i) - \frac{\sum_{i=1}^{n}\ln(X_i)}{n}\right)^2}{n}. \qquad (2.5)
\end{aligned}
$$

To verify that these estimators maximize the likelihood function L, it is equivalent to show that they maximize the log-likelihood function $\mathcal{L}$. To do this, we find the Hessian (second derivative matrix) of $\mathcal{L}$ and verify that it is a negative-definite matrix (Salas, Hille, and Etgen 1999):

$$\begin{aligned}\frac{\delta^2\mathcal{L}}{\delta\mu^2} &= \frac{\delta}{\delta\mu}\left[\frac{\sum_{i=1}^{n}\ln(X_i)}{\sigma^2} - \frac{2n\mu}{2\sigma^2}\right] \\ &= -\frac{n}{\hat{\sigma}^2}; \end{aligned} \tag{2.6}$$

$$\begin{aligned}\frac{\delta^2\mathcal{L}}{\delta(\sigma^2)^2} &= \frac{\delta}{\delta\sigma^2}\left[-\frac{n}{2\sigma^2} + \frac{\sum_{i=1}^{n}(\ln(X_i)-\mu)^2}{2(\sigma^2)^2}\right] \\ &= \frac{n}{2(\hat{\sigma}^2)^2} - 2\cdot\frac{\sum_{i=1}^{n}(\ln(X_i)-\hat{\mu})^2}{2(\hat{\sigma}^2)^3} \\ &= \frac{1}{2\cdot(\hat{\sigma}^2)^3}\left[n\hat{\sigma}^2 - 2\sum_{i=1}^{n}(\ln(X_i)-\hat{\mu})^2\right] \\ &= \frac{1}{2\cdot(\hat{\sigma}^2)^3}\left[\sum_{i=1}^{n}(\ln(X_i)-\hat{\mu})^2 - 2\sum_{i=1}^{n}(\ln(X_i)-\hat{\mu})^2\right] \\ &= \frac{1}{2\cdot(\hat{\sigma}^2)^3}\left[-\sum_{i=1}^{n}(\ln(X_i)-\hat{\mu})^2\right]; \end{aligned} \tag{2.7}$$

$$\begin{aligned}\frac{\delta^2\mathcal{L}}{\delta\sigma^2\cdot\delta\mu} &= \frac{\delta}{\delta\mu}\left[-\frac{n}{2\sigma^2} + \frac{\sum_{i=1}^{n}(\ln(X_i)-\mu)^2}{2(\sigma^2)^2}\right] \\ &= \frac{-2\cdot\sum_{i=1}^{n}(\ln(X_i)-\hat{\mu})}{2(\hat{\sigma}^2)^2} \\ &= \frac{n\hat{\mu} - \sum_{i=1}^{n}\ln(X_i)}{(\hat{\sigma}^2)^2} \\ &= \frac{n\frac{\sum_{i=1}^{n}\ln(X_i)}{n} - \sum_{i=1}^{n}\ln(X_i)}{(\hat{\sigma}^2)^2} \\ &= \frac{\sum_{i=1}^{n}\ln(X_i) - \sum_{i=1}^{n}\ln(X_i)}{(\hat{\sigma}^2)^2} = 0;\ \text{and} \end{aligned} \tag{2.8}$$

$$\begin{aligned}\frac{\delta^2\mathcal{L}}{\delta\mu\cdot\delta\sigma^2} &= \frac{\delta}{\delta\sigma^2}\left[\frac{\sum_{i=1}^{n}\ln(X_i)}{\sigma^2} - \frac{2n\mu}{2\sigma^2}\right] \\ &= \frac{-\sum_{i=1}^{n}\ln(X_i) + n\hat{\mu}}{(\hat{\sigma}^2)^2} \\ &= \frac{-\sum_{i=1}^{n}\ln(X_i) + n\frac{\sum_{i=1}^{n}\ln(X_i)}{n}}{(\hat{\sigma}^2)^2} \\ &= \frac{-\sum_{i=1}^{n}\ln(X_i) + \sum_{i=1}^{n}\ln(X_i)}{(\hat{\sigma}^2)^2} = 0. \end{aligned} \tag{2.9}$$

Therefore, the Hessian is given by

$$H = \begin{bmatrix} \frac{\delta^2 \mathcal{L}}{\delta \mu^2} & \frac{\delta^2 \mathcal{L}}{\delta \sigma^2 \cdot \delta \mu} \\ \frac{\delta^2 \mathcal{L}}{\delta \mu \cdot \delta \sigma^2} & \frac{\delta^2 \mathcal{L}}{\delta (\sigma^2)^2} \end{bmatrix} = \begin{bmatrix} -\frac{n}{\hat{\sigma}^2} & 0 \\ 0 & \frac{-\sum_{i=1}^{n} (\ln(X_i) - \hat{\mu})^2}{2 \cdot (\hat{\sigma}^2)^3} \end{bmatrix}, \tag{2.10}$$

which has a determinant greater than zero with $H_{(1,1)}$ less than zero. Thus, the Hessian is negative-definite, indicating a strict local maximum (Fitzpatrick 2006).

We additionally need to verify that the likelihoods of the boundaries of the parameters are less than the likelihoods of the derived Maximum Likelihood estimators for μ and σ^2; if so, then we know that the estimates are strict global maximums instead of simply local maximums, as determined by Equation 2.10. As stated in Equation 1.1, the parameter μ has finite magnitude with a range of all real numbers. Taking the limit as μ approaches ∞, the likelihood equation goes to $-\infty$; similarly, as μ approaches $-\infty$, the likelihood equation has a limit of $-\infty$:

$$\begin{aligned} \lim_{\mu \to \infty} \mathcal{L} &= \lim_{\mu \to \infty} \left\{ -\frac{n}{2} \ln\left(2\pi\sigma^2\right) - \sum_{i=1}^{n} \ln(X_i) - \frac{\sum_{i=1}^{n} \ln(X_i)^2}{2\sigma^2} + \frac{\sum_{i=1}^{n} \ln(X_i)\mu}{\sigma^2} - \frac{n\mu^2}{2\sigma^2} \right\} \\ &\longrightarrow -\frac{n}{2} \ln\left(2\pi\sigma^2\right) - \sum_{i=1}^{n} \ln(X_i) - \frac{\sum_{i=1}^{n} \ln(X_i)^2}{2\sigma^2} + \frac{\sum_{i=1}^{n} \ln(X_i)\infty}{\sigma^2} - \frac{n\infty^2}{2\sigma^2} \\ &\longrightarrow -\frac{n}{2} \ln\left(2\pi\sigma^2\right) - \sum_{i=1}^{n} \ln(X_i) - \frac{\sum_{i=1}^{n} \ln(X_i)^2}{2\sigma^2} + \infty - \infty^2 \\ &\longrightarrow \infty - \infty^2 \longrightarrow -\infty; \end{aligned}$$

$$\begin{aligned} \lim_{\mu \to -\infty} \mathcal{L} &= \lim_{\mu \to -\infty} \left\{ -\frac{n}{2} \ln\left(2\pi\sigma^2\right) - \sum_{i=1}^{n} \ln(X_i) - \frac{\sum_{i=1}^{n} \ln(X_i)^2}{2\sigma^2} + \frac{\sum_{i=1}^{n} \ln(X_i)\mu}{\sigma^2} - \frac{n\mu^2}{2\sigma^2} \right\} \\ &\longrightarrow -\frac{n}{2} \ln\left(2\pi\sigma^2\right) - \sum_{i=1}^{n} \ln(X_i) - \frac{\sum_{i=1}^{n} \ln(X_i)^2}{2\sigma^2} - \frac{\sum_{i=1}^{n} \ln(X_i)\infty}{\sigma^2} - \frac{n\infty^2}{2\sigma^2} \\ &\longrightarrow -\frac{n}{2} \ln\left(2\pi\sigma^2\right) - \sum_{i=1}^{n} \ln(X_i) - \frac{\sum_{i=1}^{n} \ln(X_i)^2}{2\sigma^2} - \infty - \infty^2 \\ &\longrightarrow -\infty - \infty^2 \longrightarrow -\infty. \end{aligned} \tag{2.11}$$

Also stated in Equation 1.1, the parameter σ^2 has finite magnitude with a range of all positive real numbers. Taking the limit as σ^2 approaches ∞, the likelihood equation goes to

$-\infty$; similarly, as σ^2 approaches 0, the likelihood equation has a limit of $-\infty$:

$$\begin{aligned}\lim_{\sigma^2\to\infty}\mathcal{L} &= \lim_{\sigma^2\to\infty}\left\{-\frac{n}{2}\ln\left(2\pi\sigma^2\right)-\sum_{i=1}^{n}\ln(X_i)-\frac{\sum_{i=1}^{n}\ln(X_i)^2}{2\sigma^2}+\frac{\sum_{i=1}^{n}\ln(X_i)\mu}{\sigma^2}-\frac{n\mu^2}{2\sigma^2}\right\}\\ &\longrightarrow -\frac{n}{2}\ln\left(2\pi\infty\right)-\sum_{i=1}^{n}\ln(X_i)-\frac{\sum_{i=1}^{n}\ln(X_i)^2}{2\infty}+\frac{\sum_{i=1}^{n}\ln(X_i)\mu}{\infty}-\frac{n\mu^2}{2\infty}\\ &\longrightarrow -\ln(\infty)-\sum_{i=1}^{n}\ln(X_i)-0+0-0\longrightarrow -\infty;\end{aligned}$$

$$\begin{aligned}\lim_{\sigma^2\to 0}\mathcal{L} &= \lim_{\sigma^2\to 0}\left\{-\frac{n}{2}\ln\left(2\pi\sigma^2\right)-\sum_{i=1}^{n}\ln(X_i)-\frac{\sum_{i=1}^{n}\ln(X_i)^2}{2\sigma^2}+\frac{\sum_{i=1}^{n}\ln(X_i)\mu}{\sigma^2}-\frac{n\mu^2}{2\sigma^2}\right\}\\ &\longrightarrow -\frac{n}{2}\ln\left(2\pi\varepsilon\right)-\sum_{i=1}^{n}\ln(X_i)-\frac{\sum_{i=1}^{n}\ln(X_i)^2}{2\varepsilon}+\frac{\sum_{i=1}^{n}\ln(X_i)\mu}{\varepsilon}-\frac{n\mu^2}{2\varepsilon}\\ &\longrightarrow -\ln(\varepsilon)-\sum_{i=1}^{n}\ln(X_i)-\infty+\infty-\infty\longrightarrow -\infty, \qquad (2.12)\end{aligned}$$

where ε is slightly greater than 0. Thus, the likelihoods of the boundaries of the parameters are less than the likelihoods of the derived Maximum Likelihood estimators for μ and σ^2.

2.2 Method of Moments Estimators

Another popular estimation technique, Method of Moments estimation equates sample moments with unobservable population moments, from which we can solve for the parameters to be estimated. In some cases, such as when estimating the parameters of an unknown probability distribution, moment-based estimates are preferred to Maximum Likelihood estimates.

To compute the Method of Moments estimators $\tilde{\mu}$ and $\tilde{\sigma}^2$, we first need to find $E(X)$ and $E(X^2)$ for $X \sim \mathit{Lognormal}(\mu, \sigma^2)$. We derive these using Casella and Berger's (2002) equation for the moments of the lognormal distribution found in Equation 1.5:

$$\begin{aligned} E(X^n) &= \exp\left[n\mu + n^2\sigma^2/2\right]; \\ \Longrightarrow E(X) &= \exp\left[\mu + \sigma^2/2\right], \\ \Longrightarrow E(X^2) &= \exp\left[2\mu + 2\sigma^2\right]. \end{aligned} \tag{2.13}$$

So, $E(X) = e^{\mu+(\sigma^2/2)}$ and $E(X^2) = e^{2(\mu+\sigma^2)}$. Now, we set $E(X)$ equal to the first sample moment m_1 and $E(X^2)$ equal to the second sample moment m_2, where

$$\begin{aligned} m_1 &= \frac{\sum_{i=1}^n X_i}{n}, \\ m_2 &= \frac{\sum_{i=1}^n X_i^2}{n}. \end{aligned} \tag{2.14}$$

Setting $E(X) = m_1$:

$$\begin{aligned} &\Longrightarrow e^{\tilde{\mu}+\tilde{\sigma}^2/2} = \frac{\sum_{i=1}^n X_i}{n} \\ &\Longrightarrow \tilde{\mu} + \frac{\tilde{\sigma}^2}{2} = \ln\left[\frac{\sum_{i=1}^n X_i}{n}\right] \\ &\Longrightarrow \tilde{\mu} + \frac{\tilde{\sigma}^2}{2} = \ln\left(\sum_{i=1}^n X_i\right) - \ln(n) \\ &\Longrightarrow \tilde{\mu} = \ln\left(\sum_{i=1}^n X_i\right) - \ln(n) - \frac{\tilde{\sigma}^2}{2}. \end{aligned} \tag{2.15}$$

Setting $E(X^2) = m_2$:

$$\begin{aligned} &\Longrightarrow e^{2(\tilde{\mu}+\tilde{\sigma}^2)} = \frac{\sum_{i=1}^n X_i^2}{n} \\ &\Longrightarrow 2\tilde{\mu} + 2\tilde{\sigma}^2 = \ln\left[\frac{\sum_{i=1}^n X_i^2}{n}\right] \\ &\Longrightarrow 2\tilde{\mu} + 2\tilde{\sigma}^2 = \ln\left(\sum_{i=1}^n X_i^2\right) - \ln(n) \\ &\Longrightarrow \tilde{\mu} = \left[\ln\left(\sum_{i=1}^n X_i^2\right) - \ln(n) - 2\tilde{\sigma}^2\right] * \frac{1}{2} \\ &\Longrightarrow \tilde{\mu} = \frac{\ln\left(\sum_{i=1}^n X_i^2\right)}{2} - \frac{\ln(n)}{2} - \tilde{\sigma}^2. \end{aligned} \tag{2.16}$$

Now, we set the two $\tilde{\mu}$s in Equations 2.15 and 2.16 equal to each other and solve for $\tilde{\sigma}^2$:

$$
\begin{aligned}
&\Longrightarrow \ln\left(\sum_{i=1}^{n} X_i\right) - \ln(n) - \frac{\tilde{\sigma}^2}{2} = \frac{\ln\left(\sum_{i=1}^{n} X_i^2\right)}{2} - \frac{\ln(n)}{2} - \tilde{\sigma}^2 \\
&\Longrightarrow 2\ln\left(\sum_{i=1}^{n} X_i\right) - 2\ln(n) - \tilde{\sigma}^2 = \ln\left(\sum_{i=1}^{n} X_i^2\right) - \ln(n) - 2\tilde{\sigma}^2 \\
&\Longrightarrow \tilde{\sigma}^2 = \ln\left(\sum_{i=1}^{n} X_i^2\right) - 2\ln\left(\sum_{i=1}^{n} X_i\right) + \ln(n). \qquad (2.17)
\end{aligned}
$$

Inserting the above value of $\tilde{\sigma}^2$ into either of the equations for $\tilde{\mu}$ yields

$$
\begin{aligned}
\tilde{\mu} &= \ln\left(\sum_{i=1}^{n} X_i\right) - \ln(n) - \frac{\tilde{\sigma}^2}{2} \\
&= \ln\left(\sum_{i=1}^{n} X_i\right) - \ln(n) - \frac{1}{2}\left[\ln\left(\sum_{i=1}^{n} X_i^2\right) - 2\ln\left(\sum_{i=1}^{n} X_i\right) + \ln(n)\right] \\
&= \ln\left(\sum_{i=1}^{n} X_i\right) - \ln(n) - \frac{\ln\left(\sum_{i=1}^{n} X_i^2\right)}{2} + \ln\left(\sum_{i=1}^{n} X_i\right) - \frac{\ln(n)}{2} \\
&= 2\ln\left(\sum_{i=1}^{n} X_i\right) - \frac{3}{2}\ln(n) - \frac{\ln\left(\sum_{i=1}^{n} X_i^2\right)}{2}. \qquad (2.18)
\end{aligned}
$$

Thus, the Method of Moments estimators are

$$
\begin{aligned}
\tilde{\mu} &= -\frac{\ln\left(\sum_{i=1}^{n} X_i^2\right)}{2} + 2\ln\left(\sum_{i=1}^{n} X_i\right) - \frac{3}{2}\ln(n) \text{ and} \\
\tilde{\sigma}^2 &= \ln\left(\sum_{i=1}^{n} X_i^2\right) - 2\ln\left(\sum_{i=1}^{n} X_i\right) + \ln(n). \qquad (2.19)
\end{aligned}
$$

2.3 Robust Estimators: Serfling

We will now examine an estimation method designed by Serfling (2002). To generalize, Serfling takes into account two different criteria when developing his estimators. The first, an efficiency criterion, is based on the asymptotic optimization in terms of the variance performance of the Maximum Likelihood estimation technique. As Serfling puts it, "for a competing estimator [to the Maximum Likelihood estimator], the asymptotic relative efficiency (ARE) is defined as the limiting ratio of sample sizes at which that estimator and the

Maximum Likelihood estimator perform 'equivalently' " (2002, p. 96). The second criterion employed by Serfling concerns robustness, which is broken down into the two measures of breakdown point and gross error sensitivity. "The breakdown point (BP) of an estimator is the greatest fraction of data values that may be corrupted without the estimator becoming uninformative about the target parameter. The gross error sensitivity (GES) approximately measures the maximum contribution to the estimation error that can be produced by a single outlying observation when the given estimator is used"(2002, p. 96). Serfling further mentions that, as the expected proportion of outliers increases, an estimator with a high BP is recommended. It is thus of greater importance that the chosen estimator have a low GES.

Thus, an optimal estimator will have a nonzero breakdown point while maintaining relatively high efficiency such that more data may be allowed to be corrupted without damaging the estimators too terribly, but with gross error sensitivity as small as possible such that the estimators are not too greatly influenced by any outliers in the data. Of course, a high asymptotic relative efficiency in comparison to the Maximum Likelihood estimators is also critical due to Maximum Likelihood's ideal asymptotic standards of efficiency. In general, Serfling outlines that, to obtain such an estimator, limits should be set which dictate a minimum acceptable BP and a maximum acceptable GES, after which ARE should be maximized subject to these contraints. It is within this framework that Serfling's estimators lie, and Serfling's estimators have made these improvements over the Maximum Likelihood estimators: despite the fact that $\hat{\mu}$ and $\hat{\sigma}^2$ possess desirable asymptotic qualities, they fail to be robust, having BP $= 0$ and GES $= \infty$, the worst case possible. The Maximum Likelihood estimation technique may attribute its sensitivity to outliers to these details. Serfling's estimators actually forfeit some efficiency (ARE) in return for a suitable amount of robustness (BP and GES).

Equation 2.20 gives the parameter estimates of μ and σ^2 for the lognormal distribution as developed by Serfling (2002):

$$\hat{\mu}_{S(k)} = \text{median}\left(\frac{\sum_{i=1}^{k} \ln X_{k(i)}}{k}\right) \text{ and}$$
$$\hat{\sigma}^2_{S(m)} = \text{median}\left(\frac{\sum_{i=1}^{m}\left(\ln X_{m(i)} - \frac{\sum_{j=1}^{m} \ln X_{m(j)}}{m}\right)^2}{m}\right), \tag{2.20}$$

where X_k and X_m are groups of k and m randomly selected values (without repetition) from a sample of size n lognormally distributed variables, taken $\binom{n}{k}$ and $\binom{n}{m}$ times, respectively. $X_{k(i)}$ or $X_{m(i)}$ indicate the i^{th} value of each group of the k or m selected Xs. Serfling notes that if $\binom{n}{k}$ and $\binom{n}{m}$ are greater than 10^7, then it is adequate to compute the estimator based on only 10^7 randomly selected groups. This is because using any more than 10^7 groups likely does not add any information that has not already been gathered about the data, but limiting the number of groups taken to 10^7 relieves a certain degree of computational burden. When simultaneously estimating μ and σ^2, Serfling suggests that $k = 9$ and $m = 9$ yield the best joint results with respect to values of BP, GES, and ARE (2002). These chosen values of k and m stem from evaluations conducted by Serfling.

It may be noted that taking the logarithm of the lognormally distributed values transforms them into normally distributed variables. If we also recall that the lognormal parameter μ is the mean of the log of the random variables, while the lognormal parameter σ is the variance of the log of the random variables, it is easier to see the flow of logic which Serfling utilized when developing these estimators. For instance, to estimate the mean of a sample of normally distributed variables, thereby finding the lognormal parameter μ, one sums their values and then divides by the sample size (note that this is actually the Maximum Likelihood estimator of μ derived in Section 2.1). By taking several smaller portions of the whole sample and finding the median of their means, Serfling eliminates almost any chance of his estimator for μ being affected by outliers. This detail is the Serfling estimators' advantage over both the Maximum Likelihood and Method of Moments estimation techniques, each of which is very susceptible to the influence of outliers found within the data. Similar results are found when examining Serfling's estimator for σ^2.

2.4 Efficient Adjusted Estimators for Large σ^2: Finney

As has been mentioned, the lognormal distribution is useful in modeling continuous random variables which are greater than or equal to zero, especially data which would be considered normally distributed except for the fact that it may be more or less skewed (Limpert et al. 2001). We can of course transform these variables such that they are normally distributed by taking their log. Although this technique has many advantages, Finney (1941) suggests that it is still important to be able to assess the sample mean and variance of the untransformed data. He notes that the result of back-transforming the mean and variance of the logarithms (the lognormal parameters μ and σ^2) gives the geometric mean of the original sample, which tends to inaccurately estimate the arithmetic mean of the population as a whole.

Finney also notes that the arithmetic mean of the sample provides a consistent estimate of the population mean, but it lacks efficiency. Finally, Finney declares that the variance of the untransformed population will not be efficiently estimated by the variance of the original sample. Therefore, the object of Finney's paper is to derive sufficient estimates of both the mean, M, and the variance, V, of the original, untransformed sample. We will thus use these estimators of M and V from Finney to retrieve the estimated lognormal parameters $\hat{\mu}_{F}$ and $\hat{\sigma}^2_{F}$ by back-transforming

$$\begin{aligned} E(X) &= M = e^{\mu+(\sigma^2/2)} \\ Var(X) &= V = e^{2(\mu+\sigma^2)} - e^{2\mu+\sigma^2}, \end{aligned} \tag{2.21}$$

(Finney 1941; Evans and Shaban 1974).

In Equations 2.28 through 2.31, we give the estimators from Finney (1941), using the notation of Johnson and Kotz (1970), for the mean and variance of the lognormal distribution, labeled M and V, respectively. In a fashion similar to the approach of Method of

Moments estimation, we can use Finney's estimators of the mean and variance to solve for estimates of the lognormal parameters μ and σ^2. Note that the following estimation procedure differs from Method of Moments estimation in that we set $E(X)$ and $E(X^2)$ equal to functions of the mean and variance as opposed to the sample moments, utilizing Finney's estimators for the mean and variance provided by Johnson and Kotz to derive the estimators for μ and σ^2.

To begin, we know from Equation 2.13 that

$$\begin{aligned} E(X) &= \exp\left[\mu + \sigma^2/2\right] \text{ and} \\ E(X^2) &= \exp\left[2\mu + 2\sigma^2\right]. \end{aligned} \tag{2.22}$$

Note that the mean of X is equivalent to the expected value of X, $E(X)$, while the variance of X is equivalent to the expected value of X^2 minus the square of the expected value of X, $E(X^2) - E(X)^2$. Therefore, we can set $E(X)$ and $E(X^2)$ as equivalent to functions of Finney's estimated mean and variance and back-solve for the parameters μ and σ^2:

$$\begin{aligned} M &= E(X) = \exp\left[\mu + \sigma^2/2\right] \\ &\Longrightarrow \ln(M) = \mu + \sigma^2/2 \\ &\Longrightarrow \hat{\mu}_{_F} = ln(\hat{M}_{_F}) - \hat{\sigma}^2_{_F}/2; \end{aligned} \tag{2.23}$$

$$\begin{aligned} V + M^2 &= E(X^2) = \exp\left[2\mu + 2\sigma^2\right] \\ &\Longrightarrow \ln(V + M^2) = 2\mu + 2\sigma^2 \\ &\Longrightarrow \hat{\mu}_{_F} = \frac{\ln(\hat{V}_{_F} + \hat{M}^2_{_F})}{2} - \hat{\sigma}^2_{_F}. \end{aligned} \tag{2.24}$$

Setting Equations 2.23 and 2.24 equal to each other, we can solve for $\hat{\sigma}^2_F$:

$$\begin{aligned}
\ln(\hat{M}_F) - \frac{\hat{\sigma}^2_F}{2} &= \frac{\ln(\hat{V}_F + \hat{M}^2_F)}{2} - \hat{\sigma}^2_F \\
\Longrightarrow \hat{\sigma}^2_F - \frac{\hat{\sigma}^2_F}{2} &= \frac{\ln(\hat{V}_F + \hat{M}^2_F)}{2} - \ln(\hat{M}_F) \\
\Longrightarrow \frac{\hat{\sigma}^2_F}{2} &= \frac{\ln(\hat{V}_F + \hat{M}^2_F)}{2} - \ln(\hat{M}_F) \\
\Longrightarrow \hat{\sigma}^2_F &= \ln(\hat{V}_F + \hat{M}^2_F) - 2\ln(\hat{M}_F). \qquad (2.25)
\end{aligned}$$

Finally, using $\hat{\sigma}^2_F$ to solve for $\hat{\mu}_F$, we obtain

$$\begin{aligned}
\hat{\mu}_F &= \ln(\hat{M}_F) - \frac{\hat{\sigma}^2_F}{2} \\
&= \ln(\hat{M}_F) - \frac{\ln(\hat{V}_F + \hat{M}^2_F) - 2\ln(\hat{M}_F)}{2} \\
&= 2\ln(\hat{M}_F) - \frac{\ln(\hat{V}_F + \hat{M}^2_F)}{2}. \qquad (2.26)
\end{aligned}$$

Thus, the Finney estimators for μ and σ^2 are

$$\begin{aligned}
\hat{\mu}_F &= 2\ln(\hat{M}_F) - \frac{\ln(\hat{V}_F + \hat{M}^2_F)}{2} \text{ and} \\
\hat{\sigma}^2_F &= \ln(\hat{V}_F + \hat{M}^2_F) - 2\ln(\hat{M}_F), \qquad (2.27)
\end{aligned}$$

where $\hat{M}_F$ and $\hat{V}_F$ are defined in Equations 2.28 through 2.31.

From Johnson and Kotz (1970), Finney's estimation of the mean, $E(X)$, and variance, $E(X^2) - E(X)^2$, for the lognormal distribution are given by

$$\begin{aligned}
\hat{M}_F &= \exp\left[\bar{Z}\right] \cdot g\left(\frac{S^2}{2}\right) \text{ and} \\
\hat{V}_F &= \exp\left[2\bar{Z}\right] \cdot \left[g\left(2S^2\right) - g\left(\frac{(n-2)S^2}{n-1}\right)\right], \qquad (2.28)
\end{aligned}$$

where

$$\begin{aligned}
Z_i &= \ln(X_i) \\
\Longrightarrow \bar{Z} &= \frac{\sum_{i=1}^{n} \ln(X_i)}{n}, \qquad (2.29)
\end{aligned}$$

$$\begin{aligned} S^2 &= \frac{\sum_{i=1}^{n}\left(Z_i - \bar{Z}\right)^2}{n-1} \\ &= \frac{\sum_{i=1}^{n}\Big(\ln\left(X_i\right) - \frac{1}{n}\sum_{j=1}^{n}\ln\left(X_j\right)\Big)^2}{n-1}, \end{aligned} \tag{2.30}$$

and $g(t)$ can be approximated as

$$g(t) = \exp[t] \cdot \left[1 - \frac{t(t+1)}{n} + \frac{t^2(3t^2+22t+21)}{6n^2}\right]. \tag{2.31}$$

It is worth mentioning that $\bar{Z}$ and S^2 from Equations 2.29 and 2.30 are equivalent to $\hat{\mu}$ and $\frac{n}{n-1}\hat{\sigma}^2$, respectively, where $\hat{\mu}$ and $\hat{\sigma}^2$ are the Maximum Likelihood estimators established earlier. Knowing this, we may rewrite Finney's estimators for the mean and variance of a lognormally distributed variable, $\hat{M}_{F}$ and $\hat{V}_{F}$, as functions of the Maximum Likelihood estimators $\hat{M}$ and $\hat{V}$:

$$\begin{aligned} \hat{M}_{F} &= \exp[\bar{Z}] \cdot g\left(\frac{S^2}{2}\right) \\ &= \exp[\hat{\mu}] \cdot g\left(\frac{\hat{\sigma}^2 n}{2(n-1)}\right) \\ &= \exp[\hat{\mu}] \cdot \exp\left[\frac{\hat{\sigma}^2 n}{2(n-1)}\right] \cdot \left[1 - \xi\left(\frac{\hat{\sigma}^2 n}{2(n-1)}\right)\right] \\ &= \exp\left[\hat{\mu} + \frac{\hat{\sigma}^2 n}{2(n-1)}\right] \cdot \left[1 - \xi\left(\frac{\hat{\sigma}^2 n}{2(n-1)}\right)\right] \\ &= \exp\left[\hat{\mu} + \frac{\hat{\sigma}^2}{2}\right] \cdot \left[1 - \xi\left(\frac{\hat{\sigma}^2}{2}\right)\right] \text{ as } n \to \infty \\ &= \hat{M} \cdot \left[1 - \xi\left(\frac{\hat{\sigma}^2}{2}\right)\right] = \hat{M} - \hat{M} \cdot \xi\left(\frac{\hat{\sigma}^2}{2}\right) \\ &> \hat{M}, \end{aligned} \tag{2.32}$$

because $\hat{M}$ is always positive and $\xi(t)$ is always negative except when n is sufficiently large;

$$\begin{aligned}
\hat{V}_{F} &= \exp\left[2\bar{Z}\right] \cdot \left[g\left(2S^2\right) - g\left(\frac{(n-2)S^2}{n-1}\right)\right] \\
&= \exp\left[2\hat{\mu}\right] \cdot \left[g\left(\frac{2n\hat{\sigma}^2}{n-1}\right) - g\left(\frac{n(n-2)\hat{\sigma}^2}{(n-1)^2}\right)\right] \\
&= \exp\left[2\hat{\mu}\right] \cdot \left(\exp\left[\frac{2n\hat{\sigma}^2}{n-1}\right] \cdot \left[1 - \xi\left(\frac{2n\hat{\sigma}^2}{n-1}\right)\right] - \exp\left[\frac{n(n-2)\hat{\sigma}^2}{(n-1)^2}\right] \cdot \left[1 - \xi\left(\frac{n(n-2)\hat{\sigma}^2}{(n-1)^2}\right)\right]\right) \\
&= \exp\left[2\hat{\mu} + \frac{2n\hat{\sigma}^2}{n-1}\right] \cdot \left[1 - \xi\left(\frac{2n\hat{\sigma}^2}{n-1}\right)\right] - \exp\left[2\hat{\mu} + \frac{n(n-2)\hat{\sigma}^2}{(n-1)^2}\right] \cdot \left[1 - \xi\left(\frac{n(n-2)\hat{\sigma}^2}{(n-1)^2}\right)\right] \\
&= \exp\left[2\hat{\mu} + 2\hat{\sigma}^2\right] \cdot \left[1 - \xi\left(2\hat{\sigma}^2\right)\right] - \exp\left[2\hat{\mu} + \hat{\sigma}^2\right] \cdot \left[1 - \xi\left(\hat{\sigma}^2\right)\right] \text{ as } n \to \infty \\
&= \exp\left[2\hat{\mu} + 2\hat{\sigma}^2\right] - \exp\left[2\hat{\mu} + \hat{\sigma}^2\right] - \exp\left[2\hat{\mu} + 2\hat{\sigma}^2\right] \cdot \xi\left(2\hat{\sigma}^2\right) + \exp\left[2\hat{\mu} + \hat{\sigma}^2\right] \cdot \xi\left(\hat{\sigma}^2\right) \\
&= \exp\left[2\hat{\mu} + 2\hat{\sigma}^2\right] - \hat{M}^2 - \exp\left[2\hat{\mu} + 2\hat{\sigma}^2\right] \cdot \xi\left(2\hat{\sigma}^2\right) + \exp\left[2\hat{\mu} + \hat{\sigma}^2\right] \cdot \xi\left(\hat{\sigma}^2\right) \\
&= \hat{V} - \exp\left[2\hat{\mu} + 2\hat{\sigma}^2\right] \cdot \xi\left(2\hat{\sigma}^2\right) + \exp\left[2\hat{\mu} + \hat{\sigma}^2\right] \cdot \xi\left(\hat{\sigma}^2\right) \\
&> \hat{V},
\end{aligned} \tag{2.33}$$

because $(\exp\left[2\hat{\mu} + 2\hat{\sigma}^2\right] \cdot \xi\left(2\hat{\sigma}^2\right)) < (\exp\left[2\hat{\mu} + \hat{\sigma}^2\right] \cdot \xi\left(\hat{\sigma}^2\right))$ except when n is sufficiently large and σ^2 is suffficiently small, where

$$\begin{aligned}
\xi(t) &= \frac{t(t+1)}{n} - \frac{t^2(3t^2 + 22t + 21)}{6n^2} \\
&= \frac{6nt(t+1) - t^2(3t^2 + 22t + 21)}{6n^2} \\
&= \frac{6nt^2 + 6nt - 3t^4 - 22t^3 - 21t^2}{6n^2}.
\end{aligned} \tag{2.34}$$

We note again the relationship between estimates of μ, σ^2, M, and V as

$$\begin{aligned}
\hat{\mu} &= 2\ln(\hat{M}) - \frac{\ln(\hat{V} + \hat{M}^2)}{2} \text{ and} \\
\hat{\sigma}^2 &= \ln(\hat{V} + \hat{M}^2) - 2\ln(\hat{M}).
\end{aligned} \tag{2.35}$$

Taking this relationship into consideration while simultaneously looking at its visual representation in Figures 2.1 and 2.2, we may notice that the magnitude of $\hat{M}$ has a greater effect on $\hat{\mu}$ and $\hat{\sigma}^2$ than does $\hat{V}$. This effect is such that the larger $\hat{M}$ gets, $\hat{\mu}$ becomes larger while $\hat{\sigma}^2$ becomes smaller, $\hat{V}$ having a near null effect. The fact that we mathematically should receive larger estimates of M from Finney than from the Maximum Likelihood estimators thus leads to larger estimates of μ and smaller estimates of σ^2 from Finney. This assumes that Finney's estimator of μ detects and corrects for a supposed negative bias from the Maximum Likelihood estimator of μ, and his estimator of σ^2 similarly detects and corrects a supposed positive bias from the Maximum Likelihood estimator of σ^2.

We additionally note that as the true value of the parameter σ^2 gets smaller and as n gets larger, the value of $g(t)$, t being a function of σ^2, has a limit of 1 (this is equivalent to the fact that $\xi(t)$ has a limit of 0). This means that, under these conditions, Finney's estimators $\hat{M}_F$ and $\hat{V}_F$ should become indistinguishable from the Maximum Likelihood estimators $\hat{M}$ and $\hat{V}$ as sample size increases, such that $\hat{\mu}_F$ and $\hat{\sigma}^2_F$ are also indistinguishable from $\hat{\mu}$ and $\hat{\sigma}^2$.

Finally, while Finney's estimators do compare to the Maximum Likelihood estimators in that they converge to the Maximum Likelihood estimators as σ decreases and n increases, Finney's estimators should nevertheless be emphasized as improvements on the Method of Moments estimators. In his paper, Finney (1941) states that his estimate of the mean is approximately as efficient as the arithmetic mean as σ^2 increases, and that his estimate of the variance is considerably more efficient than the arithmetic variance as σ^2 increases. Since μ and σ^2 can be written as functions of the mean M and variance V (refer to Equation 2.35), this efficiency over the moment estimates can be extended to the idea that Finney's estimates of μ and σ^2 are more efficient than the Method of Moments estimators of the lognormal distribution parameters. Whether Finney's estimators of μ and σ^2 accomplish these tasks will be discussed in Section 3.2.

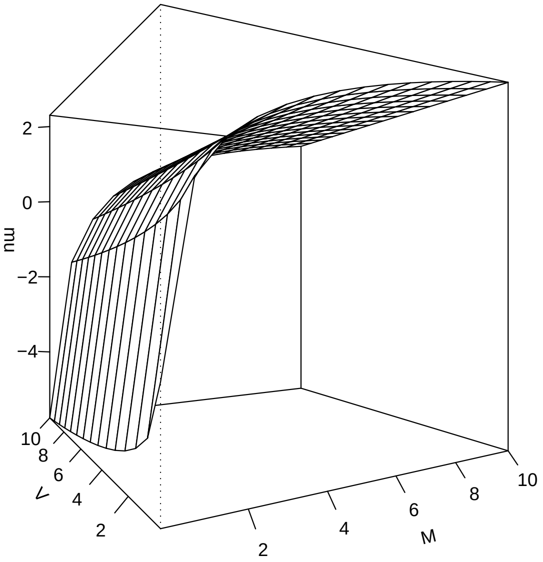

Figure 2.1: Visual Representation of the Influence of $\hat{M}$ and $\hat{V}$ on $\hat{\mu}$. $\hat{M}$ has greater influence on $\hat{\mu}$ than does $\hat{V}$, with $\hat{\mu}$ increasing as $\hat{M}$ increases.

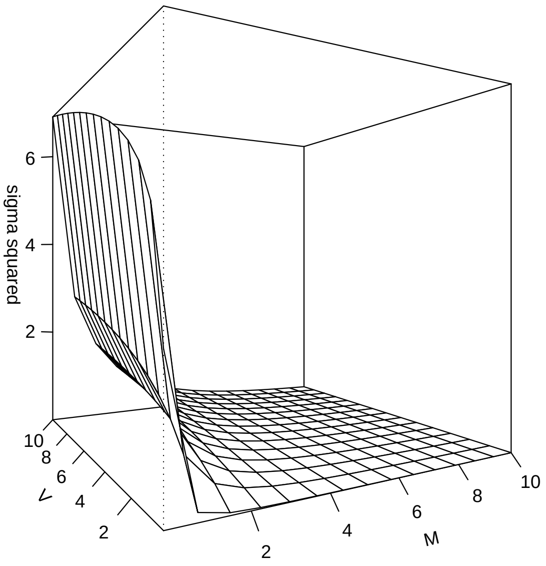

Figure 2.2: Visual Representation of the Influence of $\hat{M}$ and $\hat{V}$ on $\hat{\sigma}^2$. $\hat{M}$ has greater influence on $\hat{\sigma}^2$ than does $\hat{V}$, with $\hat{\sigma}^2$ decreasing as $\hat{M}$ increases.

3. SIMULATION STUDY

3.1 Simulation Procedure and Selected Parameter Combinations

Upon plotting various density functions, it may be found that different magnitudes of μ and σ provide varying shapes of the density in general. Figure 3.1 presents two plots of several densities overlaying each other to provide an idea of the different shapes which the lognormal can have; take note that changing the magnitude of μ appears to only change the stretch of the plots in the horizontal lengthwise direction.

To study parameter estimation of the lognormal distribution, brief preliminary parameter estimates for our application in Section 4 are conducted. This application deals with determining authorship of documents based on the distribution of sentence lengths, where a sentence is measured by the number of words it contains. More details follow in said Section 4. As depicted by Figure 3.1, the general density shapes for the lognormal distribution are mapped well by the shape parameters $\sigma = 10$, $3/2$, 1, $1/2$, and $1/4$. These σs are also relevant to our application, again based on the brief conduction of parameter estimates. It appears that any σ less than $1/4$ will continue the general trend of a bell curve, and so we will not be using the suggested shape parameter of Figure 3.1, $\sigma = 1/8$, in our simulation studies. For μ, as stated above, it appears that differing magnitudes generally only stretch the plots horizontally; because of this, we will limit our parameter estimation study to different μ values of 2.5, 3, and 3.5, which were particularly selected because of the preliminary estimates of μ for our application.

The chosen sample sizes for our simulations will be limited to $n = 10$, 25, 100, and 500. These values will allow us to look at small sample properties while confirming larger sample properties as well.

The number of simulations for each of the parameter and sample size combinations is 10,000. This value was selected based on the criterion that it is a sufficiently large number

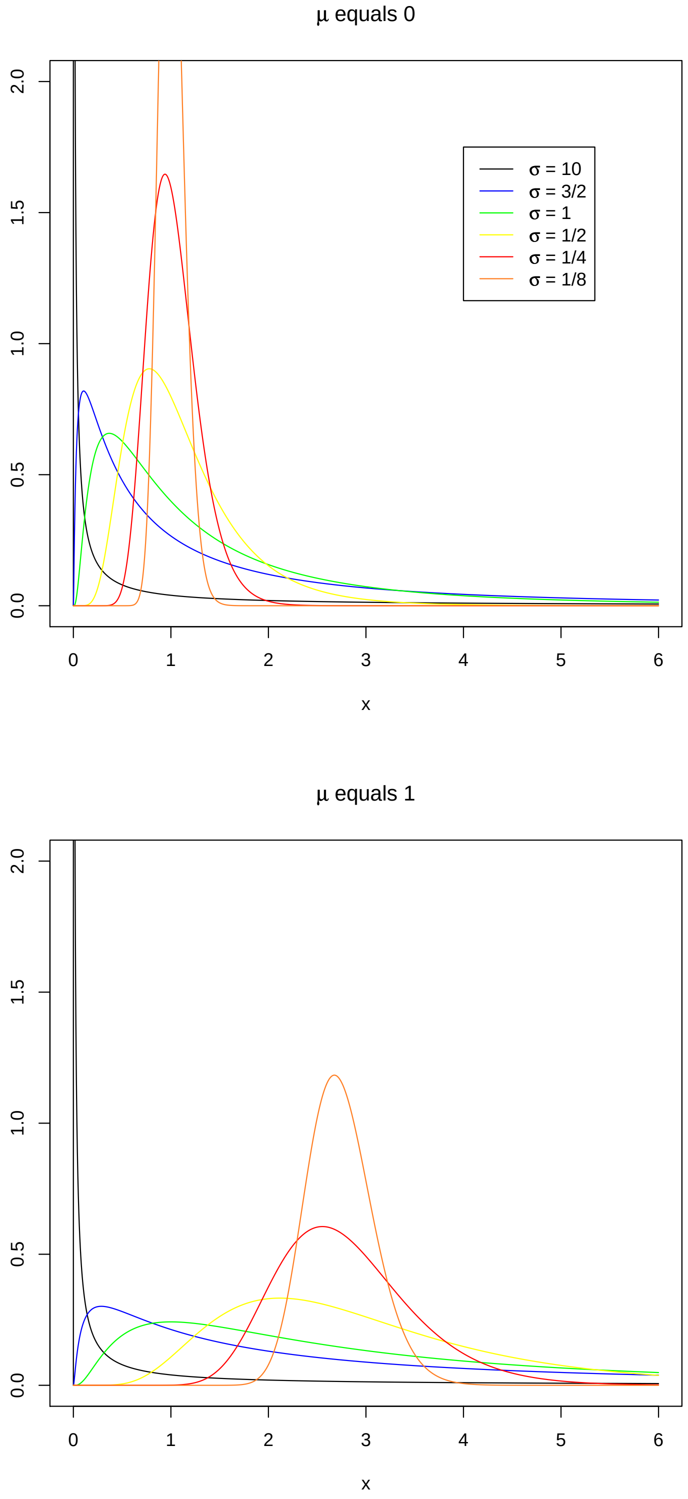

Figure 3.1: Some Lognormal Density Plots, $\mu = 0$ and $\mu = 1$.

to accurately approximate the bias and MSE of the discussed estimators.

3.2 Simulation Results

To generate the realizations of the lognormal distribution, Gnu Scientific Library functions were used in the coding language of C. In particular, the function *gsl_ran_lognormal* (*const gsl_rng* $*$ *r*, *double mu*, *double sigma*) generated individual realizations. This code is supplied in Appendix A. After running simulations under the specifications mentioned in Section 3.1, the estimates, biases, and mean squared errors were retrieved for each parameter and sample size combination. These results are summarized in Tables 3.1 through 3.6.

Table 3.1: Estimator Biases and MSEs of μ; $\mu = 2.5$.

		MLE		MOM		Serfling		Finney	
n:	σ	bias	MSE	bias	MSE	bias	MSE	bias	MSE
10	10	-0.036	9.842	12.178	180.548	-0.037	10.056	12.34	170.208
25	10	-0.019	4.043	15.081	251.128	-0.019	4.044	9.935	105.685
100	10	-0.025	1.000	18.591	362.103	-0.025	1.003	5.882	36.363
500	10	0.000	0.204	21.577	477.071	0.000	0.205	0.598	0.745
10	1.5	-0.002	0.222	0.403	0.46	-0.001	0.227	-0.959	1.540
25	1.5	-0.001	0.092	0.363	0.259	-0.001	0.092	-0.235	0.221
100	1.5	-0.001	0.022	0.253	0.104	-0.001	0.022	0.053	0.025
500	1.5	-0.001	0.004	0.138	0.046	0.000	0.004	0.014	0.005
10	1	0.000	0.099	0.119	0.125	0.000	0.102	-0.183	0.195
25	1	0.000	0.040	0.084	0.053	0.000	0.04	0.014	0.040
100	1	-0.001	0.010	0.041	0.018	-0.001	0.010	0.010	0.010
500	1	0.000	0.002	0.013	0.006	0.000	0.002	0.003	0.002
10	0.5	0.000	0.025	0.009	0.025	0.000	0.025	-0.005	0.025
25	0.5	-0.003	0.010	0.002	0.010	-0.003	0.010	-0.004	0.010
100	0.5	0.000	0.003	0.001	0.003	0.000	0.003	0.000	0.003
500	0.5	0.000	0.001	0.000	0.001	0.000	0.001	0.000	0.001
10	0.25	0.000	0.006	0.001	0.006	0.000	0.006	-0.002	0.006
25	0.25	-0.002	0.003	-0.001	0.003	-0.002	0.003	-0.003	0.003
100	0.25	0.000	0.001	0.000	0.001	0.000	0.001	0.000	0.001
500	0.25	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Table 3.2: Estimator Biases and MSEs of σ; $\mu = 2.5$.

		MLE		MOM		Serfling		Finney	
n:	σ	bias	MSE	bias	MSE	bias	MSE	bias	MSE
10	10	-0.757	5.419	-8.551	73.132	-0.611	5.436	-0.486	5.388
25	10	-0.331	2.097	-8.282	68.601	-0.857	2.559	-0.279	2.041
100	10	-0.071	0.512	-7.938	63.028	-0.934	1.308	-0.032	0.473
500	10	-0.018	0.100	-7.598	57.745	-0.963	1.012	0.164	0.114
10	1.5	-0.113	0.122	-0.525	0.316	-0.092	0.122	0.398	0.515
25	1.5	-0.044	0.047	-0.374	0.178	-0.124	0.057	0.063	0.117
100	1.5	-0.011	0.011	-0.219	0.079	-0.140	0.029	-0.056	0.013
500	1.5	-0.002	0.002	-0.109	0.034	-0.143	0.023	-0.013	0.002
10	1	-0.076	0.055	-0.236	0.088	-0.062	0.055	0.053	0.137
25	1	-0.031	0.020	-0.139	0.045	-0.084	0.025	-0.055	0.022
100	1	-0.007	0.005	-0.060	0.019	-0.093	0.013	-0.021	0.005
500	1	-0.002	0.001	-0.018	0.007	-0.096	0.010	-0.005	0.001
10	0.5	-0.038	0.014	-0.061	0.016	-0.031	0.014	-0.028	0.013
25	0.5	-0.016	0.005	-0.028	0.008	-0.043	0.006	-0.015	0.005
100	0.5	-0.004	0.001	-0.008	0.002	-0.047	0.003	-0.004	0.001
500	0.5	0.000	0.000	-0.001	0.000	-0.048	0.002	0.000	0.000
10	0.25	-0.020	0.004	-0.023	0.004	-0.017	0.004	-0.010	0.003
25	0.25	-0.008	0.001	-0.009	0.001	-0.021	0.002	-0.004	0.001
100	0.25	-0.002	0.000	-0.002	0.000	-0.023	0.001	-0.001	0.000
500	0.25	0.000	0.000	0.000	0.000	-0.024	0.001	0.000	0.000

Table 3.3: Estimator Biases and MSEs of μ; $\mu = 3$.

		MLE		MOM		Serfling		Finney	
n:	σ	bias	MSE	bias	MSE	bias	MSE	bias	MSE
10	10	0.005	9.936	12.233	182.703	0.010	10.146	12.398	172.223
25	10	0.002	3.999	15.192	254.520	0.002	4.001	10.023	107.379
100	10	0.001	1.010	18.531	359.679	0.002	1.012	5.899	36.542
500	10	0.002	0.201	21.587	477.523	0.002	0.202	0.600	0.741
10	1.5	0.001	0.225	0.410	0.476	0.000	0.229	-0.962	1.539
25	1.5	0.000	0.092	0.362	0.259	0.000	0.092	-0.231	0.216
100	1.5	0.000	0.022	0.255	0.106	0.000	0.023	0.054	0.026
500	1.5	0.000	0.005	0.139	0.046	-0.001	0.005	0.014	0.005
10	1	-0.002	0.102	0.115	0.126	-0.002	0.104	-0.185	0.204
25	1	-0.001	0.039	0.084	0.052	-0.001	0.039	0.014	0.039
100	1	0.001	0.010	0.042	0.017	0.001	0.010	0.012	0.010
500	1	0.000	0.002	0.014	0.006	0.000	0.002	0.003	0.002
10	0.5	-0.001	0.025	0.008	0.025	0.000	0.026	-0.006	0.025
25	0.5	-0.001	0.010	0.004	0.010	-0.001	0.010	-0.002	0.010
100	0.5	-0.001	0.002	0.001	0.003	-0.001	0.002	-0.001	0.002
500	0.5	0.000	0.000	0.001	0.001	0.000	0.000	0.000	0.000
10	0.25	0.001	0.006	0.002	0.006	0.001	0.007	-0.001	0.006
25	0.25	0.000	0.003	0.000	0.003	0.000	0.003	-0.001	0.003
100	0.25	0.000	0.001	0.000	0.001	0.000	0.001	0.000	0.001
500	0.25	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Table 3.4: Estimator Biases and MSEs of σ; $\mu = 3$.

		MLE		MOM		Serfling		Finney	
n:	σ	bias	MSE	bias	MSE	bias	MSE	bias	MSE
10	10	-0.743	5.435	-8.553	73.168	-0.597	5.476	-0.471	5.414
25	10	-0.275	2.064	-8.284	68.631	-0.806	2.473	-0.224	2.013
100	10	-0.080	0.502	-7.940	63.061	-0.940	1.313	-0.041	0.463
500	10	-0.018	0.097	-7.598	57.732	-0.963	1.011	0.164	0.112
10	1.5	-0.109	0.123	-0.524	0.316	-0.087	0.124	0.405	0.523
25	1.5	-0.045	0.046	-0.376	0.178	-0.124	0.056	0.060	0.114
100	1.5	-0.010	0.011	-0.218	0.079	-0.139	0.029	-0.055	0.013
500	1.5	-0.002	0.002	-0.108	0.034	-0.143	0.023	-0.013	0.002
10	1	-0.077	0.055	-0.234	0.087	-0.063	0.056	0.052	0.138
25	1	-0.032	0.020	-0.141	0.045	-0.085	0.025	-0.056	0.022
100	1	-0.009	0.005	-0.060	0.019	-0.095	0.013	-0.022	0.005
500	1	-0.002	0.001	-0.019	0.007	-0.096	0.010	-0.005	0.001
10	0.5	-0.039	0.014	-0.061	0.016	-0.032	0.014	-0.030	0.013
25	0.5	-0.015	0.005	-0.028	0.007	-0.042	0.006	-0.014	0.005
100	0.5	-0.003	0.001	-0.007	0.002	-0.046	0.003	-0.003	0.001
500	0.5	-0.001	0.000	-0.002	0.001	-0.048	0.003	-0.001	0.000
10	0.25	-0.020	0.003	-0.023	0.004	-0.017	0.003	-0.010	0.003
25	0.25	-0.008	0.001	-0.009	0.001	-0.021	0.002	-0.004	0.001
100	0.25	-0.002	0.000	-0.002	0.000	-0.023	0.001	-0.001	0.000
500	0.25	0.000	0.000	0.000	0.000	-0.024	0.001	0.000	0.000

Table 3.5: Estimator Biases and MSEs of μ; $\mu = 3.5$.

		MLE		MOM		Serfling		Finney	
n:	σ	bias	MSE	bias	MSE	bias	MSE	bias	MSE
10	10	-0.006	9.884	12.241	183.492	-0.006	10.121	12.384	171.569
25	10	-0.012	3.997	15.131	253.456	-0.012	3.998	9.990	106.913
100	10	-0.002	1.000	18.646	364.109	-0.003	1.002	5.901	36.565
500	10	0.000	0.202	21.518	474.399	0.000	0.203	0.597	0.753
10	1.5	-0.009	0.223	0.396	0.462	-0.009	0.228	-0.963	1.525
25	1.5	0.001	0.090	0.362	0.257	0.001	0.090	-0.226	0.213
100	1.5	0.002	0.022	0.255	0.106	0.002	0.022	0.056	0.026
500	1.5	0.001	0.005	0.139	0.047	0.001	0.005	0.015	0.005
10	1	-0.001	0.102	0.115	0.125	-0.002	0.104	-0.181	0.200
25	1	-0.001	0.040	0.083	0.053	-0.001	0.040	0.013	0.040
100	1	-0.001	0.010	0.042	0.017	-0.001	0.010	0.010	0.010
500	1	-0.001	0.002	0.012	0.006	-0.001	0.002	0.001	0.002
10	0.5	0.001	0.026	0.010	0.026	0.001	0.026	-0.005	0.026
25	0.5	0.000	0.010	0.005	0.010	0.000	0.010	-0.001	0.010
100	0.5	0.000	0.003	0.001	0.003	0.000	0.003	0.000	0.003
500	0.5	0.000	0.001	0.000	0.001	0.000	0.001	0.000	0.001
10	0.25	0.001	0.006	0.001	0.006	0.001	0.006	-0.002	0.006
25	0.25	0.000	0.002	0.000	0.002	0.000	0.002	-0.001	0.002
100	0.25	0.000	0.001	0.000	0.001	0.000	0.001	0.000	0.001
500	0.25	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Table 3.6: Estimator Biases and MSEs of σ; $\mu = 3.5$.

		MLE		MOM		Serfling		Finney	
n:	σ	bias	MSE	bias	MSE	bias	MSE	bias	MSE
10	10	-0.746	5.442	-8.554	73.179	-0.602	5.480	-0.474	5.424
25	10	-0.291	2.085	-8.284	68.639	-0.821	2.499	-0.239	2.034
100	10	-0.074	0.505	-7.939	63.031	-0.936	1.311	-0.035	0.466
500	10	-0.018	0.101	-7.600	57.763	-0.964	1.015	0.163	0.115
10	1.5	-0.113	0.121	-0.527	0.319	-0.093	0.122	0.397	0.510
25	1.5	-0.047	0.046	-0.375	0.179	-0.127	0.057	0.057	0.113
100	1.5	-0.011	0.011	-0.218	0.079	-0.141	0.030	-0.056	0.013
500	1.5	-0.002	0.002	-0.108	0.033	-0.144	0.023	-0.013	0.002
10	1	-0.080	0.055	-0.237	0.089	-0.066	0.055	0.047	0.135
25	1	-0.032	0.020	-0.140	0.045	-0.085	0.025	-0.055	0.022
100	1	-0.007	0.005	-0.061	0.019	-0.093	0.013	-0.020	0.005
500	1	-0.001	0.001	-0.018	0.007	-0.096	0.010	-0.004	0.001
10	0.5	-0.039	0.014	-0.062	0.016	-0.032	0.014	-0.029	0.013
25	0.5	-0.015	0.005	-0.027	0.007	-0.042	0.006	-0.014	0.005
100	0.5	-0.004	0.001	-0.007	0.002	-0.047	0.003	-0.003	0.001
500	0.5	-0.001	0.000	-0.002	0.000	-0.048	0.003	-0.001	0.000
10	0.25	-0.019	0.003	-0.022	0.004	-0.015	0.003	-0.009	0.003
25	0.25	-0.008	0.001	-0.009	0.001	-0.021	0.002	-0.004	0.001
100	0.25	-0.002	0.000	-0.002	0.000	-0.023	0.001	-0.001	0.000
500	0.25	0.000	0.000	0.000	0.000	-0.024	0.001	0.000	0.000

3.2.1 Maximum Likelihood Estimator Results

The Maximum Likelihood estimators performed very well in each parameter combination simulated; in most parameter combinations studied, the Maximum Likelihood estimators were among the most dependable estimators. In almost every case, both the biases and MSEs of the Maximum Likelihood estimators tend to zero as the sample size increases. Of course, this stems from the fact that Maximum Likelihood estimators are both asymptotically efficient (they achieve the Cramer-Rao lower bound) and unbiased (bias tends to zero as the sample size increases). Visual examples of these properties, as well as comparisons to the other estimators' results, may be seen in Figure 3.2.

3.2.2 Method of Moments Estimator Results

The efficiency and precision of the Method of Moments estimators is not as frequent as the Maximum Likelihood estimators. In particular, the Method of Moments estimators seem to improve as σ gets smaller; a rule for using a Method of Moments estimation on a lognormal distribution may be to restrict its use to $\sigma \leq 1$. These results are consistent across all values of μ studied. When σ is less than 1, the Method of Moments estimators are similar to the Maximum Likelihood estimators in that certain asymptotic properties are present, including the fact that biases and MSEs tend to zero as n increases in most cases.

When σ is as large as 10, however, the Method of Moments estimator biases for μ actually increase as n increases, and for both μ and σ the biases are very large in magnitude. This is mainly due to the fact that there are no pieces in Equation 2.19 for calculating the Method of Moments estimators of μ and σ which have a function of the data in the numerator with a function of the sample size in the denominator. Instead, estimating for μ relies on the idea that $-\frac{\ln\left(\sum_{i=1}^{n} X_i^2\right)}{2} + 2\ln\left(\sum_{i=1}^{n} X_i\right)$ will not grow too large such that $-\frac{3}{2}\ln(n)$ cannot compensate for it, and estimating for σ relies on the idea that $\ln\left(\sum_{i=1}^{n} X_i^2\right) - 2\ln\left(\sum_{i=1}^{n} X_i\right)$ will not grow too small such that it cannot compensate for the value of $\ln(n)$. Unfortunately, when σ (or the variance of the log of the random variables) is 10, the values of the random

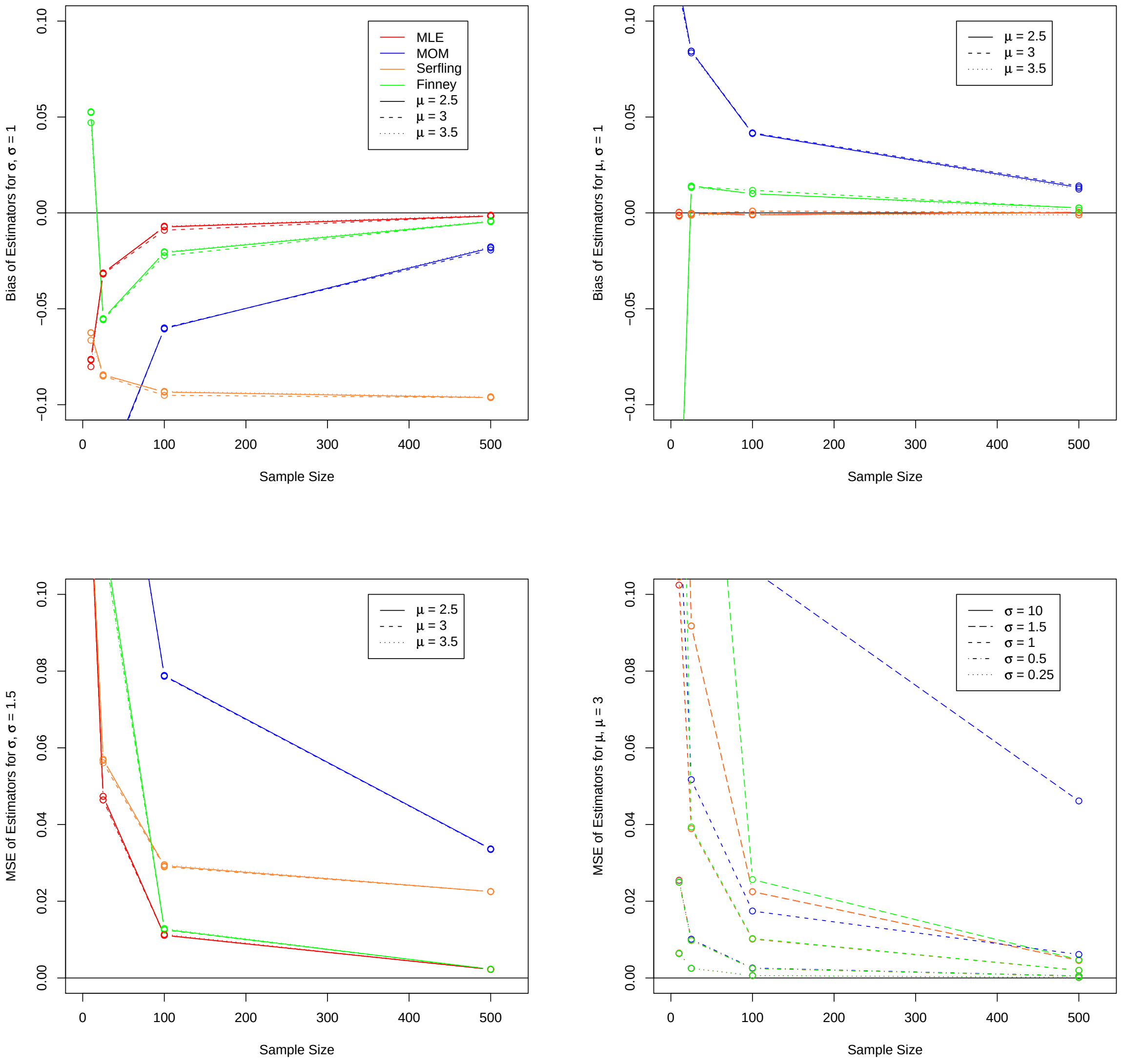

Figure 3.2: Plots of Maximum Likelihood Estimators' Performance Compared to Other Estimators. In almost every scenario, including those depicted above, the Maximum Likelihood estimators perform very well by claiming low biases and MSEs, especially as the sample size n increases.

variables greatly fluctuate, causing the estimates of μ and σ to be too high and too low, respectively, allowing for the large magnitude in the biases of each. A table of simulated values for when $\mu = 3$ and $\sigma = 10$ is given in Table 3.7. A contrasting table of simulated values for when $\mu = 3$ and $\sigma = 1$ is given in Table 3.8.

Table 3.7: Simulated Parts of the Method of Moments Estimators, $\mu = 3, \sigma = 10$.

	Estimating μ:			Estimating σ:		
n:	$\tilde{\mu}$	$-\frac{\ln\left(\sum_{i=1}^{n} X_i^2\right)}{2} + 2\ln\left(\sum_{i=1}^{n} X_i\right)$	$-\frac{3}{2}\ln(n)$	$\tilde{\sigma}$	$\ln\left(\sum_{i=1}^{n} X_i^2\right) - 2\ln\left(\sum_{i=1}^{n} X_i\right)$	$\ln(n)$
10	15.105	18.559	-3.454	1.446	-0.202	2.303
25	18.149	22.977	-4.828	1.717	-0.262	3.219
100	21.589	28.497	-6.908	2.061	-0.348	4.605
500	24.620	33.942	-9.322	2.403	-0.431	6.215

Table 3.8: Simulated Parts of the Method of Moments Estimators, $\mu = 3, \sigma = 1$.

	Estimating μ:			Estimating σ:		
n:	$\tilde{\mu}$	$-\frac{\ln\left(\sum_{i=1}^{n} X_i^2\right)}{2} + 2\ln\left(\sum_{i=1}^{n} X_i\right)$	$-\frac{3}{2}\ln(n)$	$\tilde{\sigma}$	$\ln\left(\sum_{i=1}^{n} X_i^2\right) - 2\ln\left(\sum_{i=1}^{n} X_i\right)$	$\ln(n)$
10	3.115	6.569	-3.454	0.765	-1.684	2.303
25	3.086	7.915	-4.828	0.860	-2.454	3.219
100	3.039	9.947	-6.908	0.942	-3.701	4.605
500	3.014	12.335	-9.322	0.981	-5.246	6.215

Despite the effectiveness of the Method of Moments estimators when σ is less than or equal to 1, they still tend to be inferior to the Maximum Likelihood estimators. This result, coupled with the results for large values of σ, makes the Method of Moments estimators less favorable. Figure 3.3 depicts these results visually.

3.2.3 Serfling Estimator Results

When considering the effectiveness of the Serfling estimators for our lognormal parameters, we find that they are dependable in most scenarios, but only if estimating μ. In many cases, they even contend well with the Maximum Likelihood estimators of μ, showing equally small biases and MSEs; this happens especially frequently as σ gets smaller. This is partly due to the similarities between calculating the Serfling estimators and the Maximum

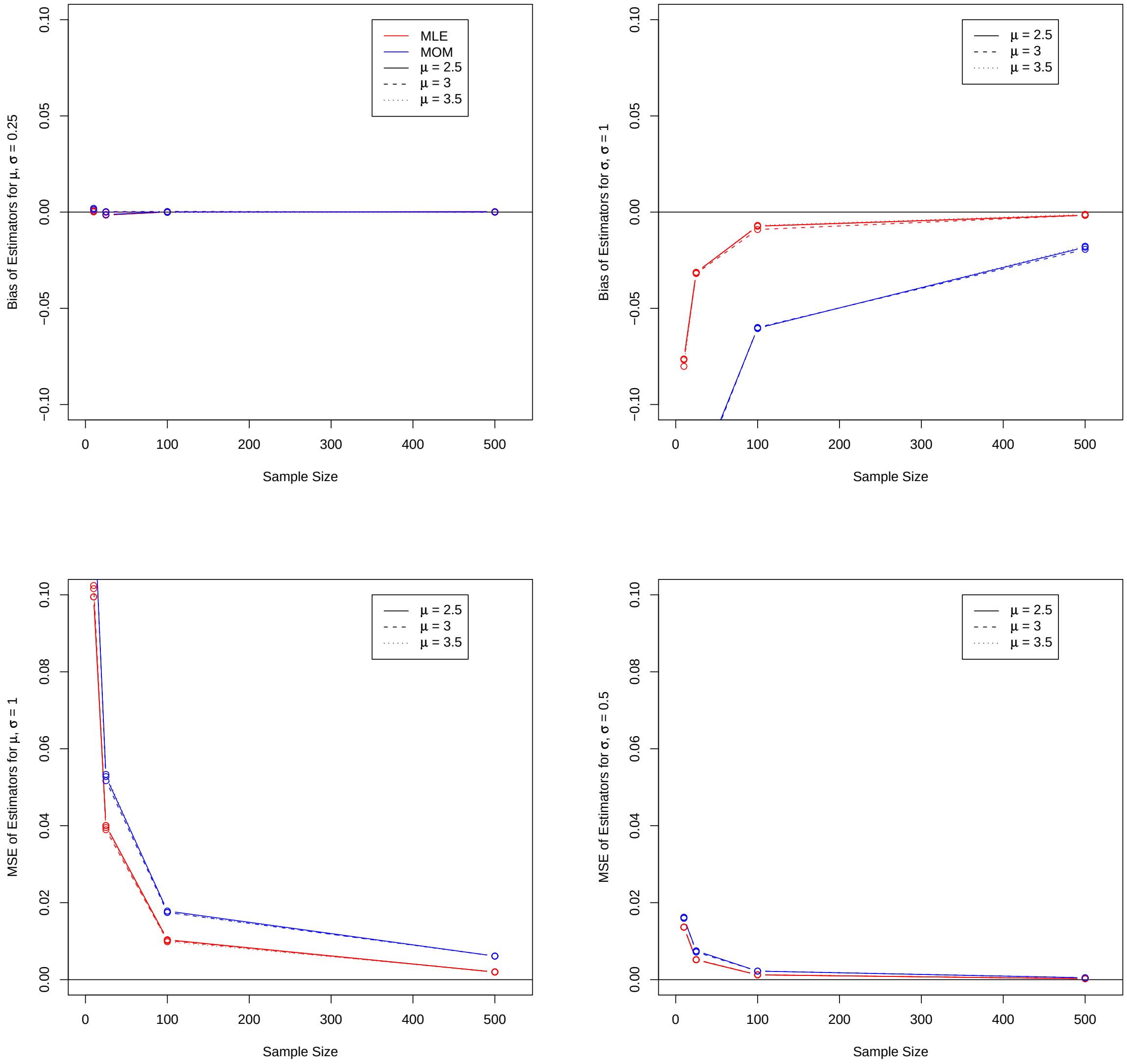

Figure 3.3: Plots of the Method of Moments Estimators' Performance Compared to the Maximum Likelihood Estimators. When $\sigma \leq 1$, the biases and MSEs of the Method of Moments estimators have small magnitudes and tend to zero as n increases, although the Method of Moments estimators are still inferior to the Maximum Likelihood estimators.

Likelihood estimators, which are noted in the second paragraph of Section 2.3. Areas where the Serfling estimator of μ proves to be superior to the Maximum Likelihood estimator of μ are likely due to any outliers in the simulated data, as the Serfling estimator is built to neglect such extremities.

On the flip side, Serfling's estimator of σ is not very accurate, having larger biases than any of the other three estimators in almost all cases. The reason for this becomes clear if we calculate the expected value of Serfling's estimator for σ^2:

$$\begin{aligned} E\left[\hat{\sigma}^2_{S(9)}\right] &= E\left[\frac{\sum_{i=1}^{9}\left(Y_i - \bar{Y}\right)^2}{9}\right] \\ &= \frac{1}{9}\cdot E\left[\sum_{i=1}^{9}\left(Y_i - \bar{Y}\right)^2\right], \end{aligned} \tag{3.1}$$

where $Y_i = \ln(X_i)$, X_i being lognormally distributed and thus making Y_i normally distributed. Note that

$$\begin{aligned} \sum_{i=1}^{9}\left(Y_i - \mu\right)^2 &= \sum_{i=1}^{9}\left[(Y_i - \bar{Y}) + (\bar{Y} - \mu)\right]^2 \\ &= \sum_{i=1}^{9}(Y_i - \bar{Y})^2 + \sum_{i=1}^{9} 2\cdot(Y_i - \bar{Y})\cdot(\bar{Y} - \mu) + \sum_{i=1}^{9}(\bar{Y} - \mu)^2 \\ &= \sum_{i=1}^{9}(Y_i - \bar{Y})^2 + 9\cdot(\bar{Y} - \mu)^2, \end{aligned} \tag{3.2}$$

where μ is the true mean of the normally distributed Y_is. Therefore,

$$\begin{aligned} \frac{1}{9}\cdot E\left[\sum_{i=1}^{9}\left(Y_i - \bar{Y}\right)^2\right] &= \frac{1}{9}\cdot E\left[\sum_{i=1}^{9}\left(Y_i - \mu\right)^2 - 9\cdot(\bar{Y} - \mu)^2\right] \\ &= \frac{1}{9}\cdot E\left[(Y_1 - \mu)^2 + (Y_2 - \mu)^2 + ... + (Y_9 - \mu)^2 - 9\cdot(\bar{Y} - \mu)^2\right] \\ &= \frac{1}{9}\cdot\left[9\sigma^2 - \frac{9\sigma^2}{9}\right] = \sigma^2 - \frac{\sigma^2}{9} = \frac{8}{9}\sigma^2. \end{aligned} \tag{3.3}$$

Thus, the bias of Serfling's estimator for σ^2 converges to $E\left[\hat{\sigma}^2_{S(9)}\right] - \sigma^2 = -\frac{1}{9}\sigma^2$ as the sample size increases. Visual comparisons between the Serfling and Maximum Likelihood estimators are depicted in Figure 3.4.

It should further be noted that another disadvantage of Serfling's estimators is the amount of time they take to compute; the simulation study as a whole took less than five minutes prior to incorporating Serfling's estimators, yet it took just short of 48 hours after incorporating them. The Serfling estimators are also memory intensive, as a limit of 10^5 combinations needed to be enforced (as opposed to 10^7, a limit suggested by Serfling) in order for the simulation to run without any segmentation faults.

3.2.4 Finney Estimator Results

Finally, analyzing the results pertaining to Finney's estimators, we can see that the hypotheses from the end of Section 2.4 which relate Finney's estimators as functions of the Maximum Likelihood estimators become reality. Especially for the hyposthesis regarding n as it gets larger and σ^2 as it gets smaller, we see that Finney's estimators $\hat{\mu}_F$ and $\hat{\sigma}_F$ truly do converge well to the Maximum Likelihood estimators $\hat{\mu}$ and $\hat{\sigma}$. It appears that the hypothesis that $\hat{\mu}_F$ should mathematically correct the negative bias of $\hat{\mu}$ and that $\hat{\sigma}_F$ should mathematically correct for the positive bias of $\hat{\sigma}$ is not true. Because of the results to these two hypotheses, there really is no advantage of using Finney's estimation technique over the Maximum Likelihood estimation technique, because rarely do Finney's estimators improve upon the Maximum Likelihood estimators. In addition to this, the Maximum Likelihood estimators are far less complicated, and thus easier to compute.

On the other hand, as was predicted in Section 2.4, we do see that Finney's estimators for μ and σ are more efficient than the Method of Moments estimators in many areas, especially as σ^2 increases. This is verified by subtracting the square of the estimator biases from their respective MSEs to retrieve the variance of that estimator for the particular parameter combination. For more details, see Tables 3.1 through 3.6. These results are also displayed graphically in Figure 3.5.

3.2.5 Summary of Simulation Results

In conclusion, the favored estimation technique of the four is the Maximum Likelihood, serving near precision in almost every scenario.

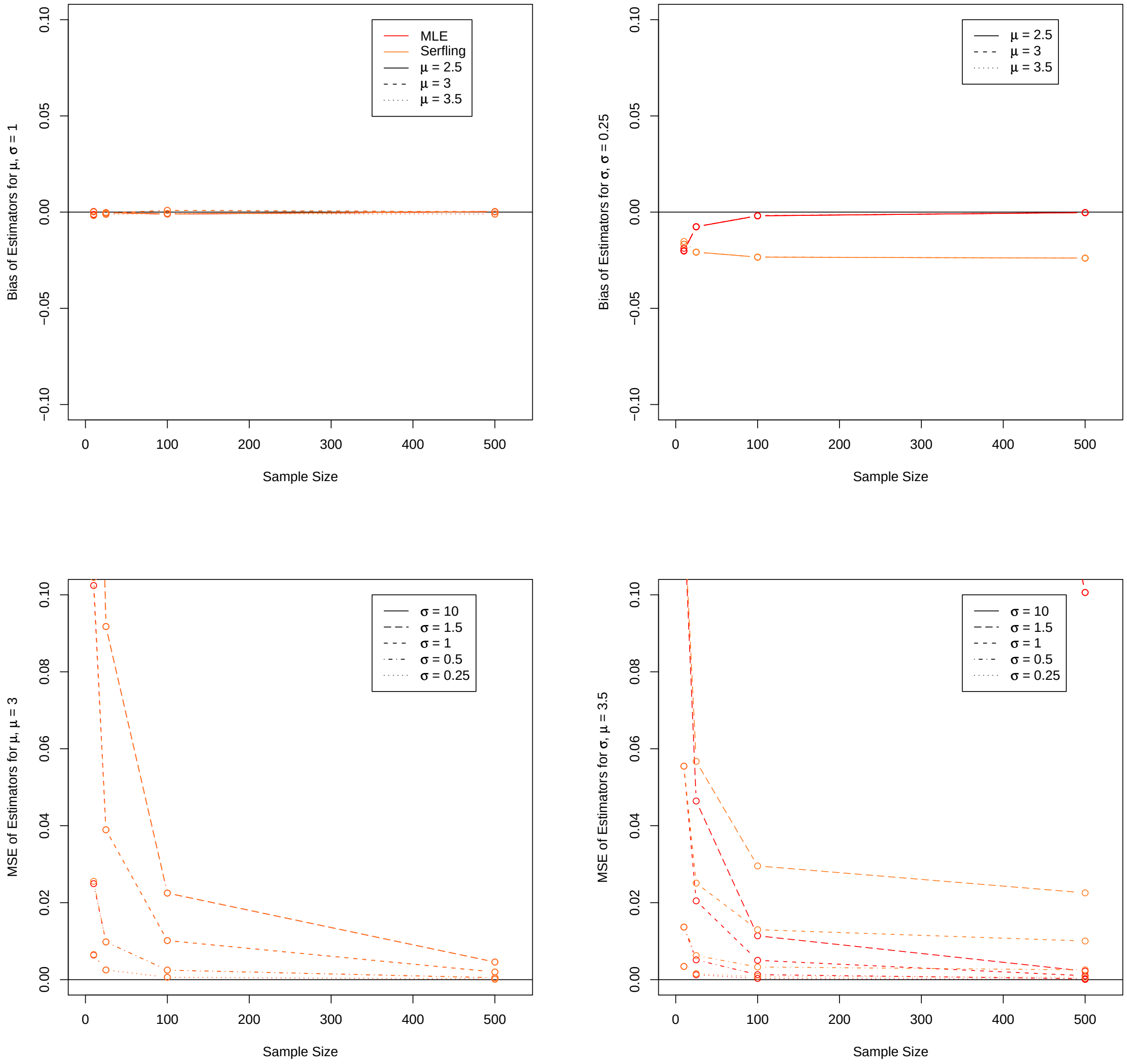

Figure 3.4: Plots of the Serfling Estimators' Performance Compared to the Maximum Likelihood Estimators. The Serfling estimators compare in effectiveness to the Maximum Likelihood estimators, especially when estimating μ and as σ gets smaller. The bias of $\hat{\sigma}_{S(9)}$ tends to converge to approximately $-\frac{\sigma}{9}$.

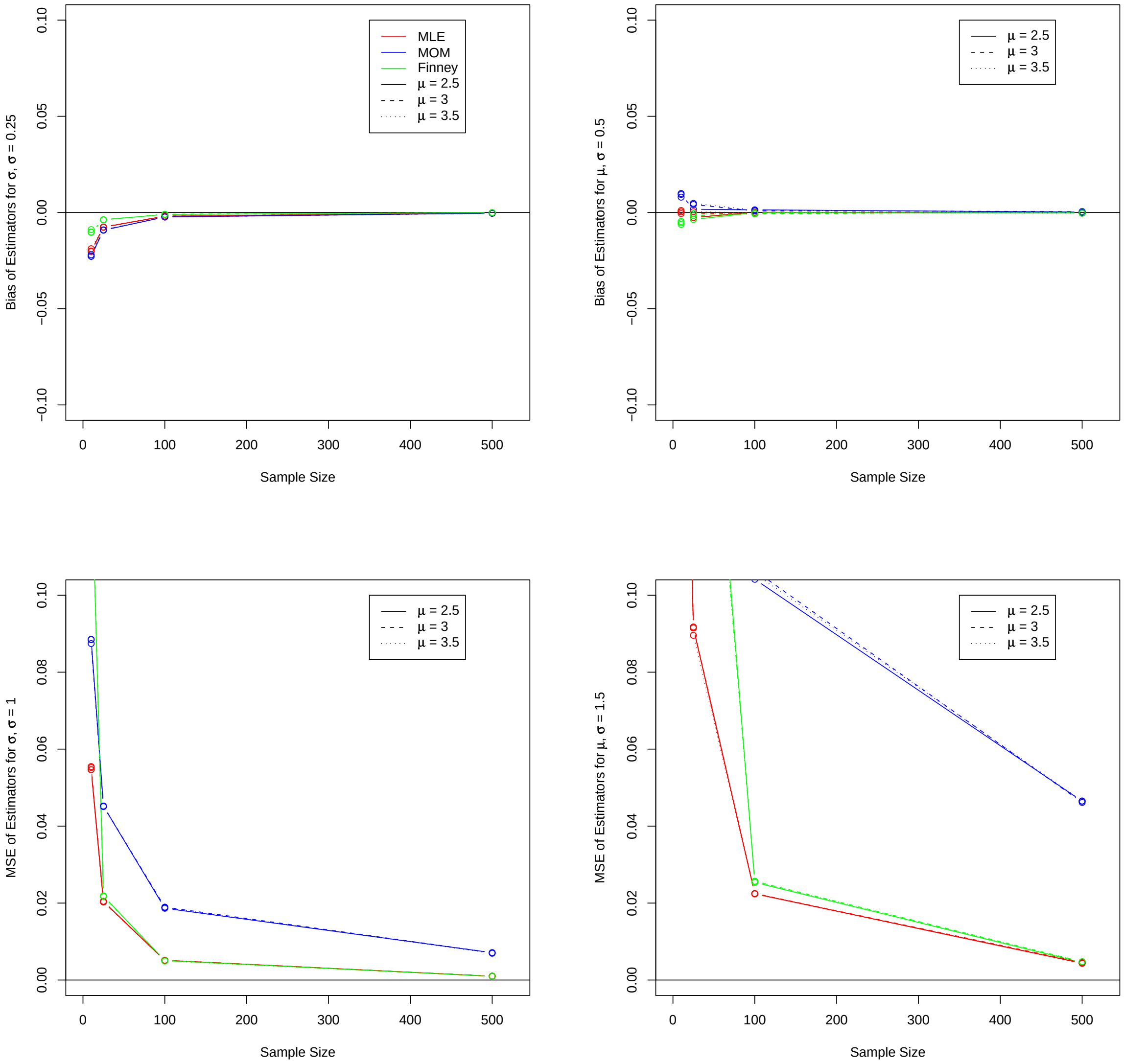

Figure 3.5: Plots of the Finney Estimators' Performance Compared to Other Estimators. Finney's estimators, while very accurate when $\sigma \leq 1$ and as n increases, rarely improve upon the Maximum Likelihood estimators. They do, however, have greater efficiency than the Method of Moments estimators, especially as σ^2 increases.

A very close contender to Maximum Likelihood estimation is Serfling's estimation technique, but only when estimating the lognormal parameter μ; when estimating σ, Serfling's technique is the worst of the four within the studied parameter combinations. Particularly, Serfling's technique should be favored in that it is intended to avoid any outliers, a capability which all three of the other estimators lack.

Finney's estimators, like the Maximum Likelihood estimators, are very accurate in estimating both μ and σ, but only as σ gets small and n grows large. Unfortunately, due to its complexity and its failure to make any major improvements over the Maximum Likelihood estimators, Finney's is not a favorable estimation method. Although they are not particularly accurate estimators as σ gets large, Finney's estimators do make improvements in efficiency over the Method of Moments estimators in this region.

Finally, Method of Moments is rather dependable when dealing with values of σ less than or equal to 1; however, this does not offer the flexibility afforded by the other three estimation techniques.

4. APPLICATION:
AUTHORSHIP ANALYSIS BY THE DISTRIBUTION OF SENTENCE LENGTHS

When analyzing the writing style of an author, qualities of interest may include the author's sentence structure. Some authors, for instance, employ short, strict sentences which are more to the point. Other authors, alternatively, may have a style consisting of long, sweeping, and thoughtful sentences. Yule declares that individual authors have a certain consistency with themselves, including within such statistics as mean and standard deviation of their sentence lengths over the course of several documents or writings. In addition to consistency with himself or herself, Yule also argues that a given author tends to differ from other authors within these statistics (1939). Combining this with Finney's assertion that the lengths of sentences (in words) are lognormally distributed (1941), what follows is an application of the estimators discussed in Sections 2 and 3, in which the application is based on authorship of a given set of texts and documents. The idea is that the differences of sentence style and length from author to author should be reflected in their individual lognormal parameters. In particular, we will examine the distribution of sentence lengths from several of the documents studied by Schaalje, Fields, Roper, and Snow (2009) as an addendum to their work concerning the authorship of the *Book of Mormon*.

Because of the results of the simulation study in Section 3.2, we will primarily use the Maximum Likelihood estimators in each of the following data sets, with the exception of those data sets which have outliers, in which cases we will use the Serfling estimators. In all plots and tables concerning the Serfling estimators in this section, it should be noted that only 10^4 combinations, as opposed to 10^5 as used in the simulation study of Section 3, have been used. This was done because the difference in the estimates of μ and σ between using 10^5 and 10^4 combinations is within a few thousandths, but the latter number of combinations takes significantly less time to compute.

4.1 *Federalist Papers* Authorship: Testing Yule's Theories

To begin, we look at the *Federalist Papers*, documents written and published during the 1780s in several New York newspapers. The intent of these papers was to persuade readers in the state of New York to ratify the proposed United States Constitution. Although the papers were written by a total of three authors, namely Alexander Hamilton, James Madison, and John Jay, they were all signed "Publius" so as to keep a certain degree of anonymity about them (FoundingFathers.info 2007). Most of the papers have since been organized into groups reflecting their particular authors.

To test Yule's theory that authors have consistency with themselves (1939), we may try dividing Hamilton's portion of the *Federalist Papers* into four groups, according to Table 4.1. Due to lack of outliers within each of the data sets, we use the Maximum Likelihood estimation technique on all four groups. Supporting Yule's suggestion, there does appear to be consistency in the density shape and parameter estimates across the writings of a single author, as depicted in Figure 4.1 and Table 4.2, with a slight exception to the first quarter of Hamilton's papers, which have a slightly higher-estimated peak than do the other three quarters.

Table 4.1: Grouping Hamilton's Portion of the *Federalist Papers* into Four Quarters.

Group	*Federalist Paper* number
1st Quarter	1, 6, 7, 8, 9, 11, 12, 13, 15, 16, 17, 18, 19, and 20
2nd Quarter	21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, and 34
3rd Quarter	35, 36, 59, 60, 61, 65, 66, 67, 68, 69, 70, 71, and 72
4th Quarter	73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, and 85

Table 4.2: Estimated Parameters for All Four Quarters of the Hamilton *Federalist Papers*.

Group	Estimation Method	$\hat{\mu}$	$\hat{\sigma}$
1st Quarter	MLE	3.272	0.579
2nd Quarter	MLE	3.458	0.550
3rd Quarter	MLE	3.443	0.584
4th Quarter	MLE	3.367	0.609

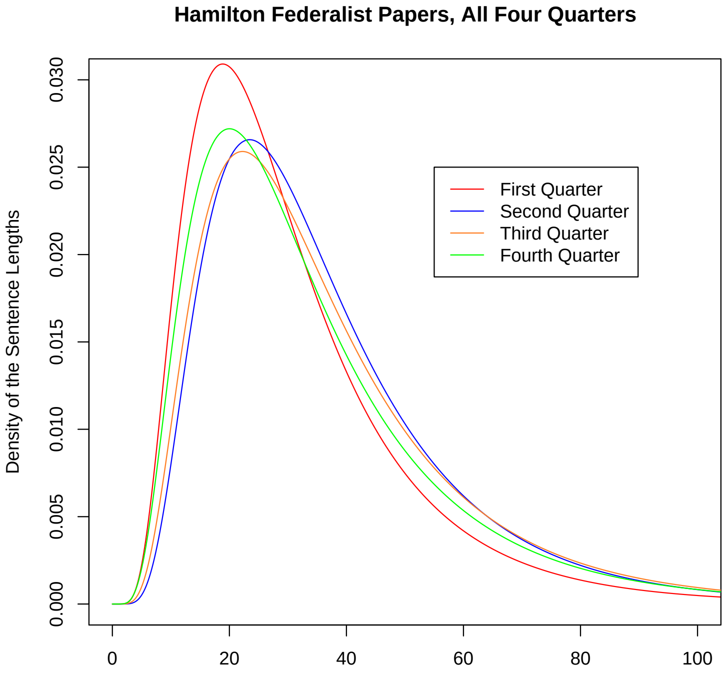

Figure 4.1: Hamilton *Federalist Papers*, All Four Quarters. When we group the *Federalist Papers* written by Hamilton into four quarters, we see some of the consistency proposed by Yule (1939).

4.2 *Federalist Papers* Authorship: Challenging Yule's Theories

To test Yule's theory that multiple authors have different lognormal parameters for the distribution of their sentence lengths (1939), we may compare the estimated distributions of the three separate authors of the *Federalist Papers.* Madison's documents have some outlying sentence lengths, and as a result we use Serfling's estimation technique to estimate the lognormal parameters here. The Maximum Likelihood estimation technique, on the other hand, is used to estimate both Hamilton's and Jay's parameters. The estimated parameters are given in Table 4.3.

Looking at the estimated density plots in Figure 4.2 and assuming Yule's theory is true, we can see that it seems as though the three authors are really a single author due to various qualities owned by each of the estimated distributions: a similar density shape is present across the writings of the three different authors, including the densities' general shape and width, as well as the lengths of their right tails. The only major difference which appears to be present among the densities is their varying heights. So much consistency between the three estimated curves may be considered as evidence against Yule's theory concerning unique lognormal parameters. Alternatively, however, the similarities in density shape may be attributed to Hamilton's, Madison's, and Jay's attempt to keep the papers anonymous with a single author instead of multiple authors. If an attempt was made, they seem to have adequately conformed to similar sentence lengths to accomplish their goal. If, however, no attempt was made, then Yule's declarations are lacking support in this case example.

Table 4.3: Estimated Parameters for All Three *Federalist Paper* Authors.

Author	Estimation Method	$\hat{\mu}$	$\hat{\sigma}$
Hamilton	MLE	3.379	0.588
Madison	Serfling	3.309	0.583
Jay	MLE	3.548	0.549

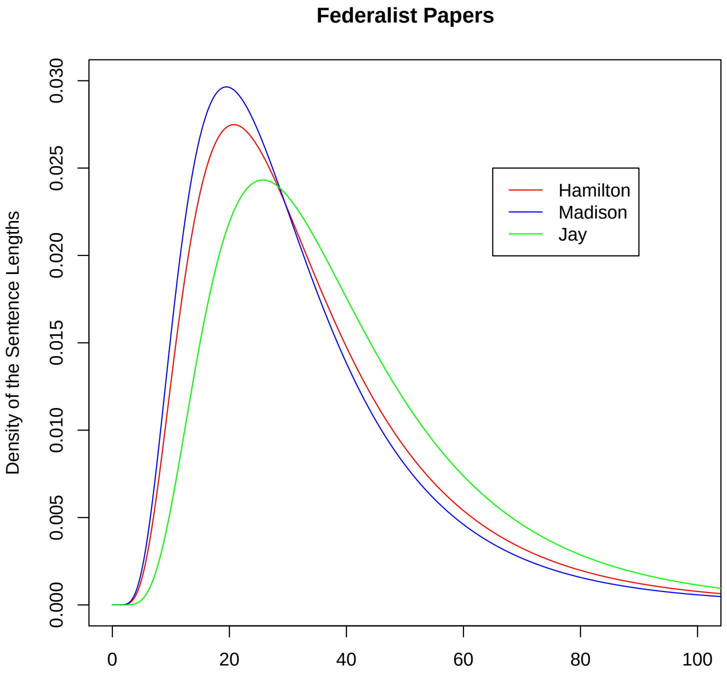

Figure 4.2: Comparing the Three Authors of the *Federalist Papers*. The similarities in the estimated sentence length densities suggest a single author, not three, for the *Federalist Papers*.

4.3 Conclusions Concerning Yule's Theories

In summary, the seemingly inconclusive findings concerning whether Yule's theories are true supports the idea that more than one method should be used simultaneously in determining authorship. One such method, for example, might be to examine the frequency of the use of noncontextual words within a document. Noncontextual words are those which act as the support of a sentence, providing structure and flow while connecting contextual words. They are frequently used in analyses to determine authorship because they are not biased or limited by the topic under discussion in a written document. Furthermore, it may be argued that frequency of such noncontextual words may be more distinguishable from author to author than sentence lengths.

4.4 The *Book of Mormon* and Sidney Rigdon

Turning to the object of the paper written by Schaalje et al. (2009), recent speculation has been made by Jockers, Witten, and Criddle (2008) that the majority of the chapters of the *Book of Mormon* were written either by Sidney Rigdon or Solomon Spalding. We can see the estimated density of sentence lengths for the 1830 version of the *Book of Mormon* text (with punctuation inserted by the printer, E.B. Grandin) compared to the estimated densities of both the compilation of letters written by Sidney Rigdon and the revelations of Sidney Rigdon in Figure 4.3; the parameter estimates may be found in Table 4.4. It may be noticed that the estimated densities of all three texts are very similar, suggesting similar authorship under Yule's theories; determining whether a significant difference is present, however, is beyond the scope of this paper.

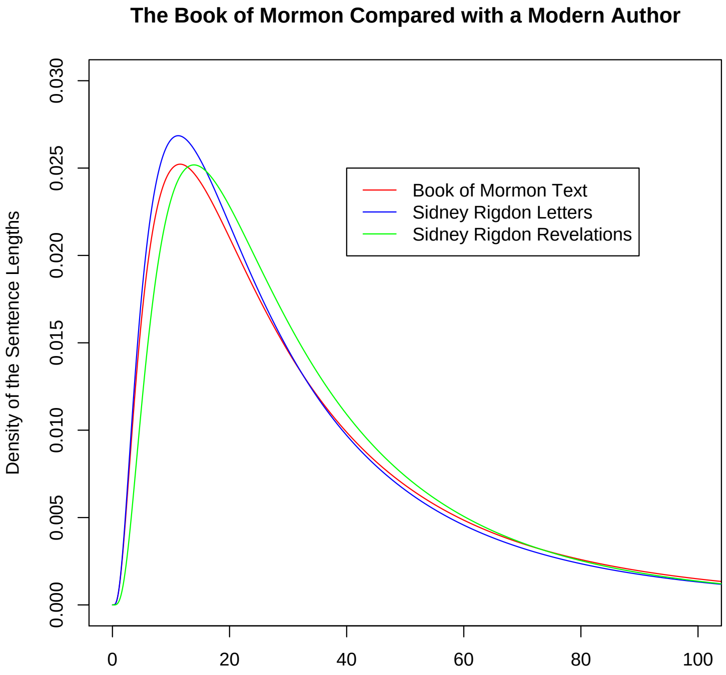

Figure 4.3: The *Book of Mormon* Compared with a Modern Author. The densities of the 1830 *Book of Mormon* text, the Sidney Rigdon letters, and the Sidney Rigdon revelations have very similar character traits.

Table 4.4: Estimated Parameters for the 1830 *Book of Mormon* Text, the Sidney Rigdon Letters, and the Sidney Rigdon Revelations.

Document	Estimation Method	$\hat{\mu}$	$\hat{\sigma}$
1830 *Book of Mormon* text	Serfling	3.270	0.907
Sidney Rigdon letters	MLE	3.211	0.890
Sidney Rigdon revelations	MLE	3.299	0.816

4.5 The *Book of Mormon* and Ancient Authors

Assuming that the *Book of Mormon* is scripture written by several ancient authors, an idea which is contrary to the declarations made by Jockers et al. (2008), a brief examination follows of the densities of sentence lengths of a few of these authors. First, we look at the writings of the prophet Nephi, found in the Books of First and Second Nephi, and compare them with those writings of the prophet Alma, found in the Book of Alma. In Figure 4.4 and Table 4.5, we find the density and parameter estimates for these two texts. A definite difference between the two density curves in Figure 4.4 may be seen, suggesting that two different authors truly are present and that the *Book of Mormon* is actually written by multiple authors rather than just one.

Table 4.5: Estimated Parameters for the Books of First and Second Nephi and the Book of Alma.

Document	Estimation Method	$\hat{\mu}$	$\hat{\sigma}$
First and Second Nephi text	Serfling	3.355	0.620
Alma text	MLE	3.465	0.789

Taking another example from the *Book of Mormon*, we look at the difference between the writings of the prophets Mormon and Moroni, found in the Book of Mormon, Words of Mormon, and Book of Moroni texts. In Figure 4.5, we may once again notice that two separate authors appear to be present. Parameter estimates of these texts are given in Table 4.6. Thus, although Figure 4.3 suggests that the *Book of Mormon* was written by Sidney Rigdon, there is alternative evidence suggested by Figures 4.4 and 4.5 and Tables 4.5 and

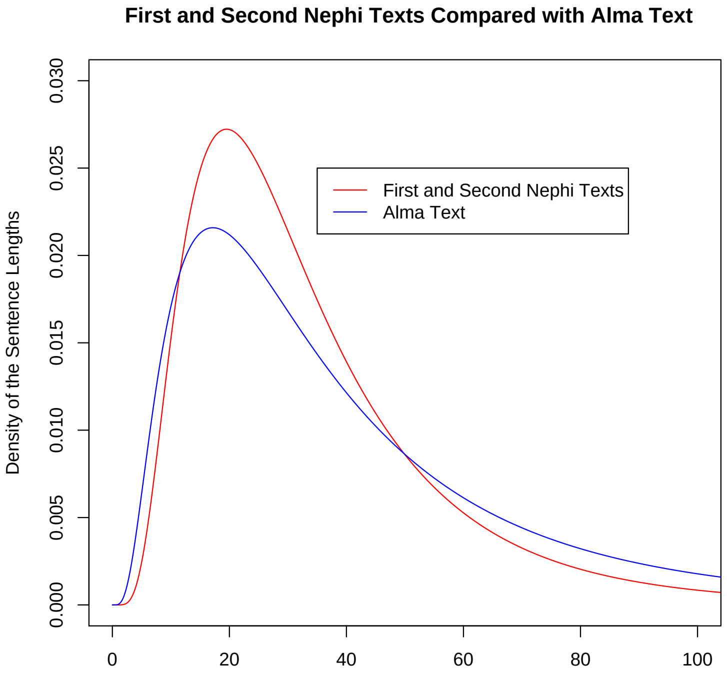

Figure 4.4: First and Second Nephi Texts Compared with Alma Text. There appears to be a difference between the densities and parameter estimates for the Books of First and Second Nephi and the Book of Alma, suggesting two separate authors.

4.6 that multiple authors are involved in the *Book of Mormon* text.

Table 4.6: Estimated Parameters for the Book of Mormon Combined with the Words of Mormon and the Book of Moroni.

Document	Estimation Method	$\hat{\mu}$	$\hat{\sigma}$
Mormon and Words of Mormon texts	MLE	3.385	0.665
Moroni text	MLE	3.380	0.852

4.6 Summary of Application Results

The densities of all the documents studied, as well as the superimposed estimated densities, may be found in Figures 4.6 through 4.8. A table of all the estimated parameters is given in Table 4.7. Overall, the best density parameter estimators appear to be the Maximum Likelihood and Finney estimators, which produce nearly identical results in every scenario. These two estimation techniques, while not perfect, more frequently capture the height, spread, and general shape of the datum's densities. Depending on the scenario, either the Method of Moments estimators or Serfling's estimators may be considered second best. Again, Serfling's estimators are beneficial because they are not so easily influenced by outliers found in the data.

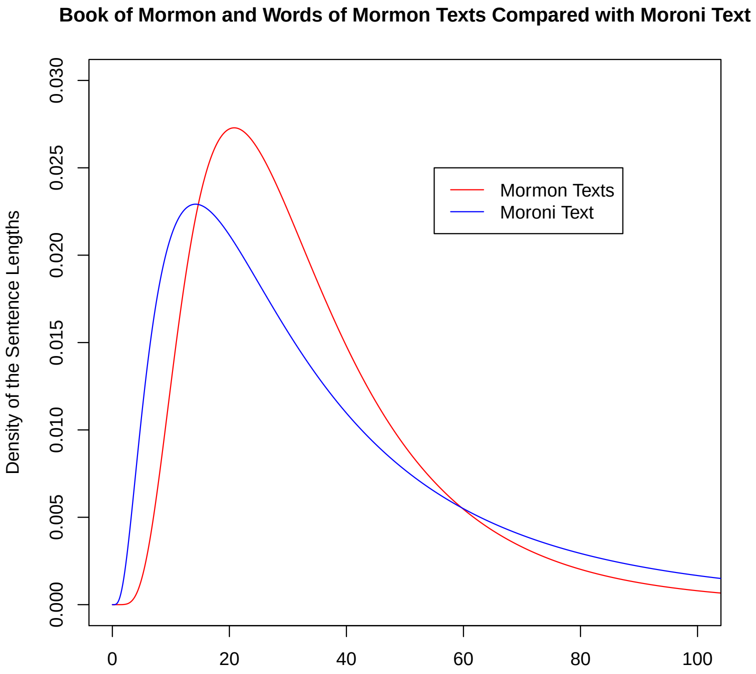

Figure 4.5: Book of Mormon and Words of Mormon Texts Compared with Moroni Text. There appears to be a difference between the densities and parameter estimates for the Book of Mormon and Words of Mormon compared to the Book of Moroni, suggesting two separate authors.

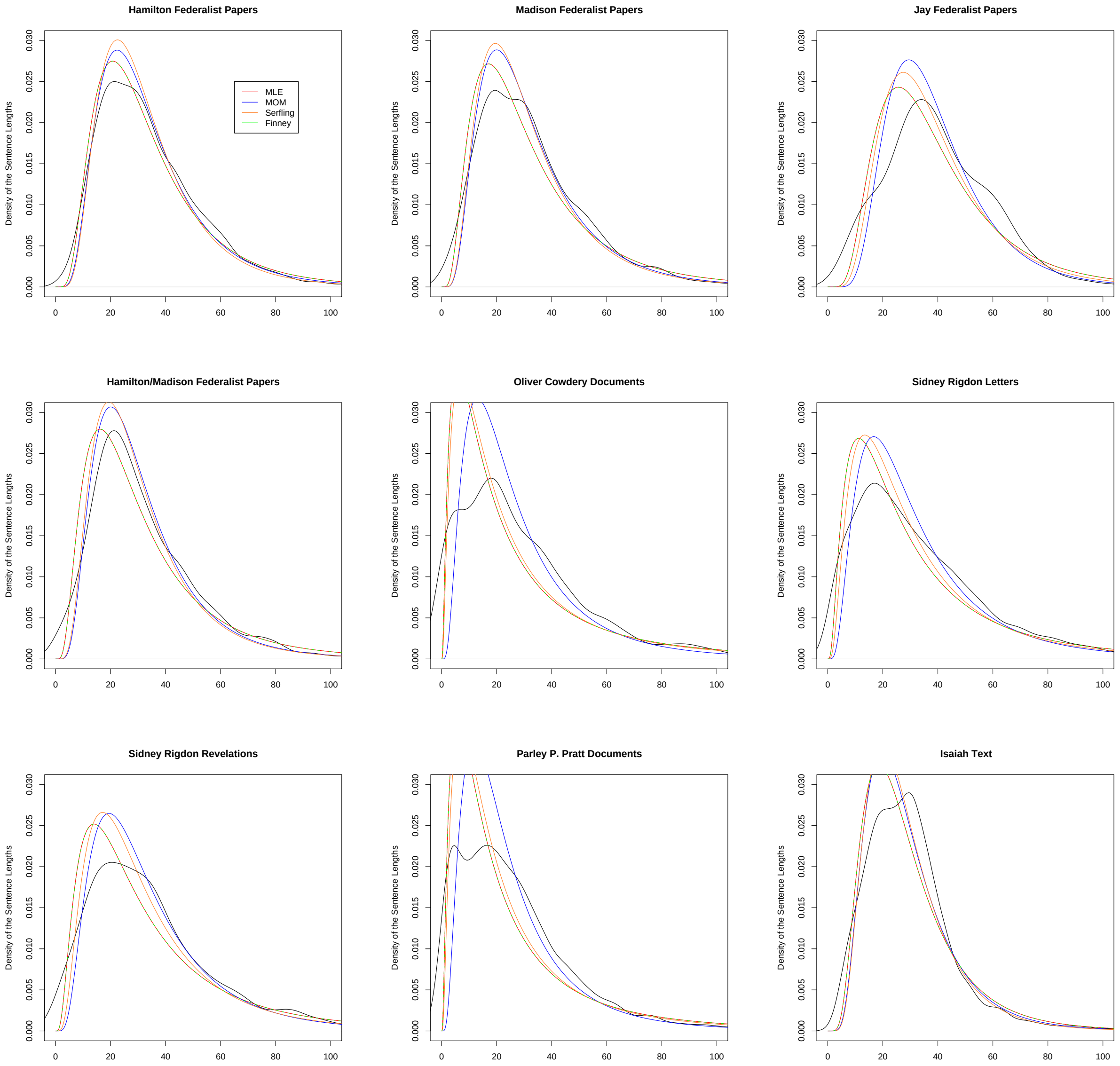

Figure 4.6: Estimated Sentence Length Densities. Densities of all the documents studied, overlaid by their estimated densities.

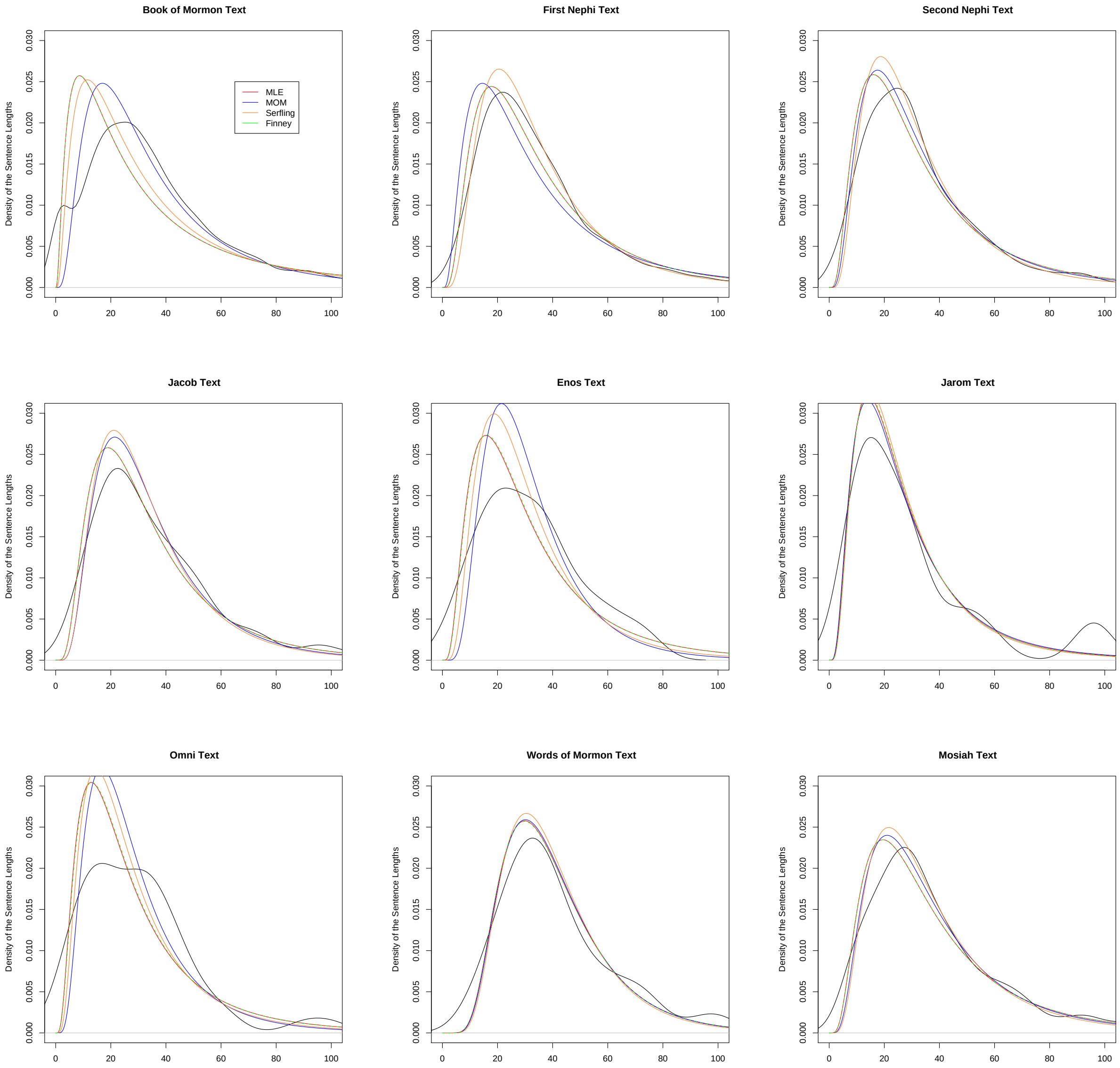

Figure 4.7: Estimated Sentence Length Densities. Densities of all the documents studied, overlaid by their estimated densities.

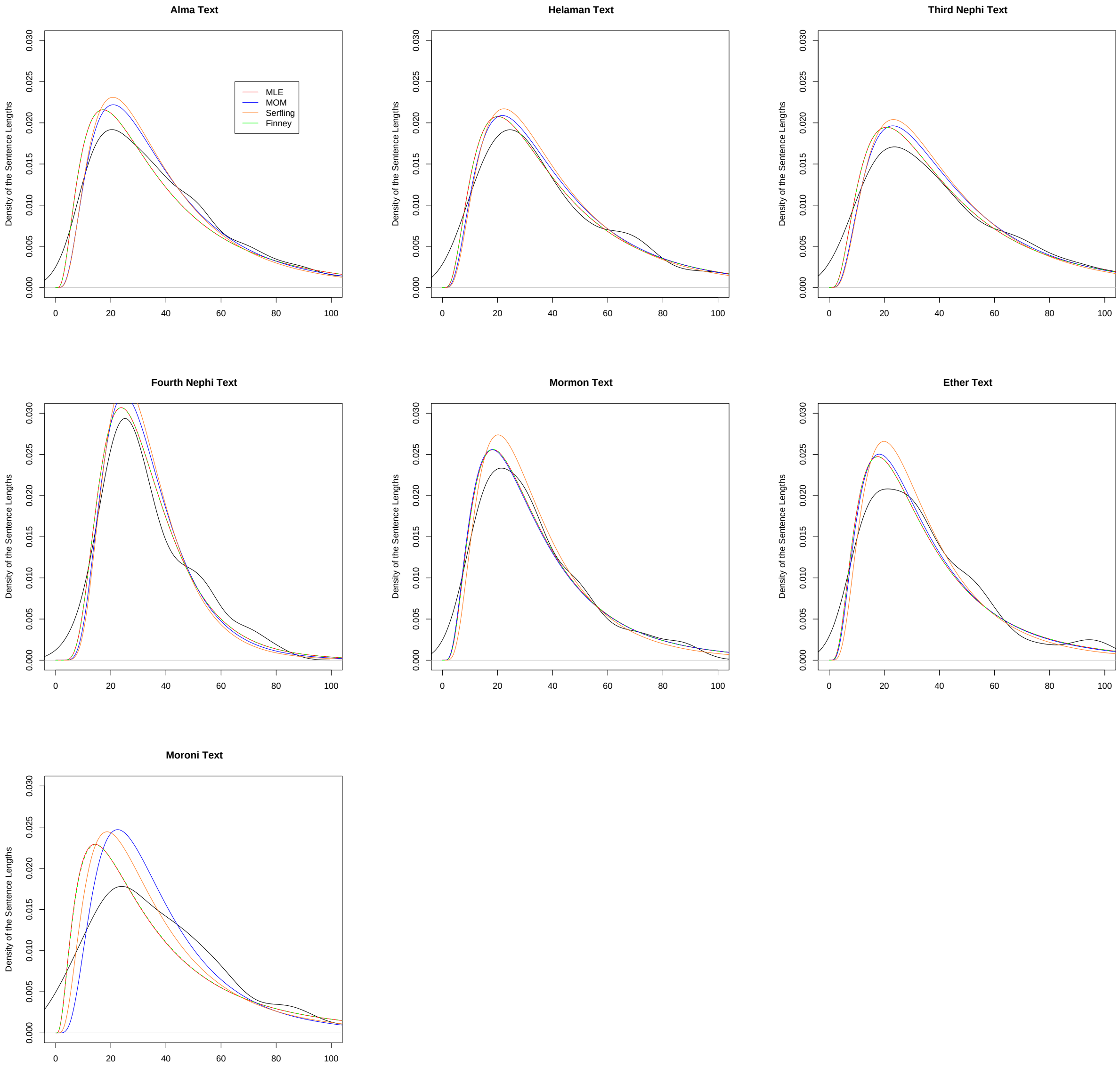

Figure 4.8: Estimated Sentence Length Densities. Densities of all the documents studied, overlaid by their estimated densities.

Table 4.7: Estimated Lognormal Parameters for All Documents Studied.

	MLE		MOM		Serfling		Finney	
Document	$\hat{\mu}$	$\hat{\sigma}$	$\hat{\mu}$	$\hat{\sigma}$	$\hat{\mu}$	$\hat{\sigma}$	$\hat{\mu}$	$\hat{\sigma}$
Hamilton *Federalist Papers*	3.379	0.588	3.395	0.536	3.380	0.516	3.379	0.588
Madison *Federalist Papers*	3.302	0.683	3.335	0.584	3.309	0.583	3.302	0.683
Jay *Federalist Papers*	3.548	0.549	3.580	0.444	3.555	0.493	3.549	0.549
Hamilton/Madison *Federalist Papers*	3.264	0.694	3.307	0.556	3.279	0.563	3.264	0.693
Oliver Cowdery documents	2.942	1.086	3.110	0.738	2.953	1.017	2.944	1.085
Sidney Rigdon letters	3.211	0.890	3.297	0.694	3.229	0.795	3.211	0.890
Sidney Rigdon revelations	3.299	0.816	3.368	0.635	3.316	0.691	3.300	0.815
Parley P. Pratt documents	2.860	1.049	3.010	0.734	2.879	0.965	2.860	1.049
Isaiah text	3.242	0.565	3.255	0.520	3.245	0.506	3.242	0.565
1830 *Book of Mormon* text	3.243	1.042	3.360	0.728	3.270	0.907	3.243	1.042
First Nephi text	3.386	0.713	3.319	0.804	3.392	0.609	3.386	0.712
Second Nephi text	3.315	0.734	3.330	0.683	3.319	0.626	3.315	0.733
Jacob text	3.373	0.658	3.401	0.581	3.379	0.574	3.374	0.657
Enos text	3.272	0.716	3.337	0.521	3.285	0.596	3.276	0.708
Jarom text	3.126	0.693	3.133	0.708	3.121	0.670	3.129	0.685
Omni text	3.136	0.761	3.193	0.615	3.142	0.678	3.141	0.752
Words of Mormon text	3.614	0.465	3.617	0.460	3.614	0.445	3.613	0.467
Mosiah text	3.444	0.688	3.459	0.643	3.449	0.613	3.444	0.688
Alma text	3.465	0.789	3.504	0.682	3.477	0.666	3.465	0.789
Helaman text	3.536	0.731	3.558	0.691	3.543	0.662	3.537	0.730
Third Nephi text	3.592	0.745	3.619	0.692	3.598	0.670	3.593	0.744
Fourth Nephi text	3.405	0.486	3.415	0.451	3.408	0.430	3.405	0.486
Mormon text	3.367	0.676	3.361	0.684	3.368	0.602	3.367	0.675
Ether text	3.374	0.712	3.377	0.691	3.377	0.622	3.375	0.712
Moroni text	3.380	0.852	3.473	0.600	3.402	0.690	3.384	0.848

5. SUMMARY

After reviewing the results from Sections 3 and 4, several generalized conclusions may be made. Firstly, the Maximum Likelihood estimators are the strongest estimators in most scenarios, proving to have low biases and MSEs in each parameter combination of the simulation study. Further, the Maximum Likelihood estimators show in the application study that they are capable of accurately estimating the density of nonsimulated data. They are the most versatile of the estimators, assuming outliers in the data are not present.

Similarly, it was found that Serfling's estimators are capable of accurate density estimation of nonsimulated data, being stronger than the other methods, especially when outliers are present. The flaw which may be attributed to Serfling's estimator of σ, however, is that it is negatively biased (see Section 3.2.3 for details). Aside from this, its performance is commendable in both the simulation and application studies.

As observed in both the simulation and application studies, Finney's estimators converge to the Maximum Likelihood estimators of μ and σ, especially when the true value of σ is small and as the sample size n gets large. Because they are computationally complex in relation to the Maximum Likelihood estimators, however, and since they rarely make improvements over the Maximum Likelihood estimators (measured in bias and MSE) which are worth such extra computation, Finney's estimators are seemingly worthless. If, however, the decision is between using Finney's estimators or the Method of Moments estimators, Finney's estimators were developed to be more efficient than the latter, especially as σ^2 increases. In this case, therefore, we find that Finney's estimators of μ and σ are superior.

Finally, the Method of Moments estimators, while relatively dependable in most situations when σ is less than or equal to 1, are the least desirable of the estimators. This is because they are usually outperformed by Finney's estimators as σ gets larger, they are susceptible to outliers, and in general they offer less flexibility than the other estimation techniques studied.

A. SIMULATION CODE

A.1 Overall Simulation

```
/**************************************************************
**
**           File name: generate.c
**
** Program Description: generates data and calculates estimator
**                      biases and MSEs for the Lognormal
**                      distribution.
**
**************************************************************/

//standard libraries
#include <stdio.h>
#include <math.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>
#include <gsl/gsl_statistics_double.h>


//prototypes
void generateSample(int sampleSize, double mu, double shapeParam);
void calcBias(int nSim, double mu, double sigma);
void calcMSE(int nSim, double mu, double sigma);
double g(double t, int n);


//variables
int nSim = 10000;
int numEst = 4; //MLE, MOM, Serfling, Finney(JK)
double muHat[4][10000]; //array holds location parameter estimate for each sim.
double sigmaHat[4][10000]; //array holds scale parameter estimate for each sim.
double muHatOverall[4]; //mean of muHat
double sigmaHatOverall[4]; //mean of sigmaHat

int index9[9]; //for the Serfling estimator

double sample[500]; //array holds each sample; made to hold longest sample n=500
double sampleSq[500]; //array holds each sample; made to hold longest sample n=500
double biasMu[4]; //bias of mu for each estimator (MLE, MOM, etc...)
double biasSigma[4]; //bias of sigma for each estimator (MLE, MOM, etc...)
double MSEMu[4]; //MSE of mu for each estimator (MLE, MOM, etc...)
                 //--> = var + bias^2
double MSESigma[4]; //MSE of sigma for each estimator (MLE, MOM, etc...)
                    //--> = var + bias^2

int seed = 31339;
int main (void)
{
  gsl_rng * r;
  r=gsl_rng_alloc(gsl_rng_mt19937);
  gsl_rng_default_seed=122; // set the random number generator seed in order
                            // to reproduce the data; note that the various
                            // parameter combinations must be kept constant
                            // when reproducing, due to the order in which
                            // simulations, etc... are generated.

  FILE *resultsMu;
  FILE *resultsSigma;
  resultsMu = fopen("resultsMu.txt","w"); //output file
  resultsSigma = fopen("resultsSigma.txt","w"); //output file

  int N = 4; //number of different sample sizes to loop through
  int n[4] = {10,25,100,500}; //Array that holds various sample sizes

  int p = 5; //Number of different sigmas to loop through
  double sigma[5] = {10.0,1.5,1.0,0.5,0.25}; //Array that holds various
                                             //values for sigma
  int V = 3; //Number of different mus to loop through
  double mu[3] = {2.5,3.0,3.5}; //Array that holds various values for mu

  fprintf(resultsMu, "n \tmu \t\tsigma
              \t\tMLE-mu \t\tMLE-bias(mu) \tMLE-MSE(mu)
              \tMOM-mu \t\tMOM-bias(mu) \tMOM-MSE(mu)
              \tSERF-mu \tSERF-bias(mu) \tSERF-MSE(mu)
              \tJK-mu \t\tJK-bias(mu) \tJK-MSE(mu)\n");
  fprintf(resultsSigma, "n \tmu \t\tsigma
              \t\tMLE-sig \tMLE-bias(sig) \tMLE-MSE(sig)
              \tMOM-sig \tMOM-bias(sig) \tMOM-MSE(sig)
              \tSERF-sig \tSERF-bias(sig) \tSERF-MSE(sig)
              \tJK-sig \tJK-bias(sig) \tJK-MSE(sig)\n");

  int v; //mu index
  int i; //sigma index
  int j; //sample index
  int m; //simulation index
  int q; //anywhere index

  for(v=0; v<V; v++) //this loop goes through the different mus
    {
      fprintf(resultsMu,"\n");
      fprintf(resultsSigma,"\n");
      for(i=0; i<p; i++) //this loop goes through the different sigmas
{
  for(j=0; j<N; j++) //this loop goes through the different sample sizes
    {
      for(m=0; m<nSim; m++) //this loop goes through each of the
                            //simulations for every parameter combo
                            //(around 10,000 simulations total)
{
  printf("mu: %lf; sigma: %lf; n: %d; Nsim: %d \n", mu[v],sigma[i],n[j],m);
  int sampSize = n[j];

       //generate a sample and store it in sample[]
       generateSample(sampSize, mu[v], sigma[i]);

  //GET ESTIMATORS:
  double fxSample[sampSize];

/////MLE
  //mu
  for(q=0; q<sampSize; q++)
    {
      fxSample[q] = log(sample[q]);
    }
  muHat[0][m] = gsl_stats_mean(fxSample,1,sampSize);
  //sigma
  for(q=0; q<sampSize; q++)
    {
      fxSample[q] = pow((log(sample[q])-muHat[0][m]),2);
    }
  sigmaHat[0][m] = pow(gsl_stats_mean(fxSample,1,sampSize),0.5);

/////MOM
  //mu
  double part1 = 0;
  double part2 = 0;
  for(q=0; q<sampSize; q++)
    {
      part1 = part1 + pow(sample[q],2);
      part2 = part2 + sample[q];
    }
  muHat[1][m] = (-log(part1)/2) + (2*log(part2)) - (3*log(sampSize)/2);
  //sigma
  sigmaHat[1][m] = pow(((log(part1)) - (2*log(part2)) + (log(sampSize))),0.5);

/////LITERATURE - Serfling
  double logX[sampSize];
  int iLog;

  for(iLog=0; iLog<=sampSize; iLog++)
    {
      logX[iLog] = log(sample[iLog]);
    }

          int combos = 10;
  if(sampSize == 10)
    combos = 10;
  else if(sampSize == 25)
    {
      //combos = 2042975;
      combos = 100000;
    }
  else if((sampSize == 100) || (sampSize == 500))
```

```
    {
      //combos = 10000000;
      combos = 100000;
    }

  double intermMu[combos]; //holds the intermediate Mu’s
  double intermSig[combos]; //holds the intermediate Sigma’s
  int combosMatrix[combos][9];

  int iSerf = 0;
  int count9 = 0;
  int indexMatrix1 = 0;
  int indexMatrix2 = 0;

  for(iSerf = 0; iSerf < combos; iSerf++)
    {
      for(count9 = 0; count9 < 9; count9++)
{
  int ranNum = floor(sampSize * gsl_ran_flat(r, 0.0, 1.0));

  if(count9==0)
    {
      index9[count9] = ranNum;
    }

  int ranNumIndex = 0;
  //loop through to make sure that we haven’t already used this index
  for(ranNumIndex = 0; ranNumIndex < count9; ranNumIndex++)
  {
    if(index9[ranNumIndex]==ranNum)
      {
ranNumIndex = count9;
count9 = count9-1; // we go back to find another number because this one
                   // has already been used.
      }
    else if(ranNumIndex==(count9-1))
      {
index9[count9] = ranNum;
      }
  } //end for-loop

  if(count9==8)
    {
      gsl_sort_int(index9,1,9);

      for(indexMatrix1=0; indexMatrix1 < iSerf; indexMatrix1++)
{
  for(indexMatrix2=0; indexMatrix2 < 9; indexMatrix2++)
    {
      if(combosMatrix[indexMatrix1][indexMatrix2] != index9[indexMatrix2])
{
  int iNew = 0;
  for(iNew = 0; iNew<9; iNew++)
    {
      combosMatrix[iSerf][iNew] = index9[iNew];
    }
  indexMatrix1 = iSerf+1;
  indexMatrix2 = 9;
}

      else if((indexMatrix1==8) && (indexMatrix2==(iSerf-1)))
// we haven't found a new combination
{
  count9 = 0; // causes us to find new indeces which we haven't already used...
}

    } // end for(indexMatrix2=0...)
} // end for(indexMatrix1=0...)
    } // end if(count9==8)
} //end for(count9=0...)

      // we only get to this point if we have found a new combination...:
      int aSerf = index9[0];
      int bSerf = index9[1];
      int cSerf = index9[2];
      int dSerf = index9[3];
      int eSerf = index9[4];
      int fSerf = index9[5];
      int gSerf = index9[6];
      int hSerf = index9[7];
      int jSerf = index9[8];

      double meanAll9 = (logX[aSerf] + logX[bSerf] + logX[cSerf] +
                         logX[dSerf] + logX[eSerf] + logX[fSerf] +
                         logX[gSerf] + logX[hSerf] + logX[jSerf])/9;

      intermMu[iSerf] = meanAll9;

      intermSig[iSerf] = (pow((logX[aSerf] - meanAll9),2) +
             pow((logX[bSerf] - meanAll9),2) + pow((logX[cSerf] - meanAll9),2) +
             pow((logX[dSerf] - meanAll9),2) + pow((logX[eSerf] - meanAll9),2) +
             pow((logX[fSerf] - meanAll9),2) + pow((logX[gSerf] - meanAll9),2) +
             pow((logX[hSerf] - meanAll9),2) + pow((logX[jSerf] - meanAll9),2))/9;
    }

  gsl_sort(intermMu,1,combos);
  gsl_sort(intermSig,1,combos);
  muHat[2][m] = gsl_stats_median_from_sorted_data(intermMu,1,combos);
  sigmaHat[2][m] = gsl_stats_median_from_sorted_data(intermSig,1,combos);
  sigmaHat[2][m] = pow(sigmaHat[2][m],0.5);


/////LITERATURE - Finney (JK)
  part1 = 0;
  part2 = 0;

  //mean
  double sumZ = 0.0;
  double sSq = 0.0;
  double zBar = 0.0;
  for(q=0; q<sampSize; q++)
    {
      sumZ = sumZ + log(sample[q]);
    }
  zBar = sumZ/sampSize;
  for(q=0;q<sampSize; q++)
    {
      sSq = sSq + pow((log(sample[q]) - zBar),2.0);
    }
  sSq = sSq/((double)sampSize - 1.0);
  if(sSq > 325)
    {
      sSq = 325.0;
    }

  double G = g(sSq/2.0, sampSize);
  double meanJK = pow(M_E,zBar)*G;

  //variance
  double varJK = pow(M_E, 2.0*zBar) * (g(2.0*sSq,sampSize) -
         g(((double)sampSize-2.0)*sSq/((double)sampSize-1.0), sampSize));

  // get mu and sigma for JK estimator:
  muHat[3][m] = 2*log(meanJK) - log(varJK + pow(meanJK,2))/2;

  sigmaHat[3][m] = log(varJK + pow(meanJK,2)) - 2*log(meanJK);
  sigmaHat[3][m] = pow(sigmaHat[3][m],0.5);
}


      // calculate the biases and MSEs for this particular
      // (sigma, sample size) pair
      calcBias(nSim, mu[v], sigma[i]);
      calcMSE(nSim, mu[v], sigma[i]);


      fprintf(resultsMu, "%d \t%lf \t%lf \t%lf \t%lf \t%lf %\t%lf \t%lf
                          \t%lf \t%lf \t%lf \t%lf \t%lf \t%lf %\t%lf\n",
                          n[j],mu[v],sigma[i],
                          muHatOverall[0],biasMu[0],MSEMu[0],
                          muHatOverall[1],biasMu[1],MSEMu[1],
                          muHatOverall[2],biasMu[2],MSEMu[2],
                          muHatOverall[3],biasMu[3],MSEMu[3]);
      fprintf(resultsSigma, "%d \t%lf \t%lf \t%lf \t%lf \t%lf %\t%lf \t%lf
                             \t%lf \t%lf \t%lf \t%lf \t%lf \t%lf %\t%lf\n",
                             n[j],mu[v],sigma[i],
                             sigmaHatOverall[0],biasSigma[0],MSESigma[0],
                             sigmaHatOverall[1],biasSigma[1],MSESigma[1],
                             sigmaHatOverall[2],biasSigma[2],MSESigma[2],
                             sigmaHatOverall[3],biasSigma[3],MSESigma[3]);

    }
```

```
}
    }

  fclose(resultsMu);
  fclose(resultsSigma);

  gsl_rng_free (r);

}//end main




//generates a sample of size n from the Lognormal distribution.
//void generateSample(int sampleSize, double mu, double shapeParam, int seed)
void generateSample(int sampleSize, double mu, double shapeParam)
{
  gsl_rng * r;
  r=gsl_rng_alloc(gsl_rng_mt19937);
  seed = seed + 1; // by incrementing our seed, we will get a new sample each time
  gsl_rng_default_seed = seed;
  int i;
  for(i=0; i<sampleSize; i++)
    {
      sample[i] = gsl_ran_lognormal(r,mu,shapeParam);
      sampleSq[i] = pow(sample[i],2);
    }

  gsl_rng_free (r);
}// end generateSample()


//the g() function used in calculating the estimators for the Lit estimates:
double g(double t, int n)
{
  double part1 = pow(M_E,t);
  double part2 = t*(t+1.0)/((double)n);
  double part3 = pow(t,2)*(3*pow(t,2) + 22*t + 21)/(6*(double)pow(n,2));

  double G = part1*(1 - part2 + part3);

  return G;
}


//given a sample of size n and a proposed value for mu and sigma,
//this function calculates the bias of the mu and sigma estimators.
void calcBias(int nSim, double mu, double sigma)
{
  double differenceMu[nSim];
  double differenceSigma[nSim];
  double tempMu[nSim];
  double tempSigma[nSim];
  int i;
```

```
//MLE
for(i=0; i<nSim; i++)
  {
    tempMu[i] = muHat[0][i];
    tempSigma[i] = sigmaHat[0][i];
    differenceMu[i] = muHat[0][i] - mu;
    differenceSigma[i] = sigmaHat[0][i] - sigma;
  }

muHatOverall[0] = gsl_stats_mean(tempMu,1,nSim);
sigmaHatOverall[0] = gsl_stats_mean(tempSigma,1,nSim);
biasMu[0] = gsl_stats_mean(differenceMu,1,nSim);
biasSigma[0] = gsl_stats_mean(differenceSigma,1,nSim);

//MOM
for(i=0; i<nSim; i++)
  {
    tempMu[i] = muHat[1][i];
    tempSigma[i] = sigmaHat[1][i];
    differenceMu[i] = muHat[1][i] - mu;
    differenceSigma[i] = sigmaHat[1][i] - sigma;
  }

muHatOverall[1] = gsl_stats_mean(tempMu,1,nSim);
sigmaHatOverall[1] = gsl_stats_mean(tempSigma,1,nSim);
biasMu[1] = gsl_stats_mean(differenceMu,1,nSim);
biasSigma[1] = gsl_stats_mean(differenceSigma,1,nSim);

//LIT - Serfling
for(i=0; i<nSim; i++)
  {
    tempMu[i] = muHat[2][i];
    tempSigma[i] = sigmaHat[2][i];
    differenceMu[i] = muHat[2][i] - mu;
    differenceSigma[i] = sigmaHat[2][i] - sigma;
  }

muHatOverall[2] = gsl_stats_mean(tempMu,1,nSim);
sigmaHatOverall[2] = gsl_stats_mean(tempSigma,1,nSim);
biasMu[2] = gsl_stats_mean(differenceMu,1,nSim);
biasSigma[2] = gsl_stats_mean(differenceSigma,1,nSim);

//LIT - Finney (JK)
for(i=0; i<nSim; i++)
  {
    tempMu[i] = muHat[3][i];
    tempSigma[i] = sigmaHat[3][i];
    differenceMu[i] = muHat[3][i] - mu;
    differenceSigma[i] = sigmaHat[3][i] - sigma;
  }

muHatOverall[3] = gsl_stats_mean(tempMu,1,nSim);
sigmaHatOverall[3] = gsl_stats_mean(tempSigma,1,nSim);
```

```
  biasMu[3] = gsl_stats_mean(differenceMu,1,nSim);
  biasSigma[3] = gsl_stats_mean(differenceSigma,1,nSim);

}// end calcBias()


//given a sample of size n and a proposed value for mu and sigma,
//this function calculates the MSE of the mu and sigma estimators.
void calcMSE(int nSim, double mu, double sigma)
{
  double differenceMu[nSim];
  double differenceSigma[nSim];
  double paramMuHat[nSim];
  double paramSigmaHat[nSim];
  double bMu, biasSqMu, varMu;
  double bSigma, biasSqSigma, varSigma;
  int i;


  //MLE
  for(i=0; i<nSim; i++)
    {
      differenceMu[i] = muHat[0][i] - mu;
      differenceSigma[i] = sigmaHat[0][i] - sigma;
      paramMuHat[i] = muHat[0][i];
      paramSigmaHat[i] = sigmaHat[0][i];
    }

  bMu = gsl_stats_mean(differenceMu,1,nSim); //the bias of mu
  biasSqMu = pow(bMu,2);
  varMu = gsl_stats_variance(paramMuHat,1,nSim);
  MSEMu[0] = varMu + biasSqMu; //the MSE of mu

  bSigma = gsl_stats_mean(differenceSigma,1,nSim); //the bias of sigma
  biasSqSigma = pow(bSigma,2);
  varSigma = gsl_stats_variance(paramSigmaHat,1,nSim);
  MSESigma[0] = varSigma + biasSqSigma; //the MSE of sigma

  //MOM
  for(i=0; i<nSim; i++)
    {
      differenceMu[i] = muHat[1][i] - mu;
      differenceSigma[i] = sigmaHat[1][i] - sigma;
      paramMuHat[i] = muHat[1][i];
      paramSigmaHat[i] = sigmaHat[1][i];
    }

  bMu = gsl_stats_mean(differenceMu,1,nSim); //the bias of mu
  biasSqMu = pow(bMu,2);
  varMu = gsl_stats_variance(paramMuHat,1,nSim);
  MSEMu[1] = varMu + biasSqMu; //the MSE of mu

  bSigma = gsl_stats_mean(differenceSigma,1,nSim); //the bias of sigma
  biasSqSigma = pow(bSigma,2);
  varSigma = gsl_stats_variance(paramSigmaHat,1,nSim);
  MSESigma[1] = varSigma + biasSqSigma; //the MSE of sigma

  //LIT - Serfling
  for(i=0; i<nSim; i++)
    {
      differenceMu[i] = muHat[2][i] - mu;
      differenceSigma[i] = sigmaHat[2][i] - sigma;
      paramMuHat[i] = muHat[2][i];
      paramSigmaHat[i] = sigmaHat[2][i];
    }

  bMu = gsl_stats_mean(differenceMu,1,nSim); //the bias of mu
  biasSqMu = pow(bMu,2);
  varMu = gsl_stats_variance(paramMuHat,1,nSim);
  MSEMu[2] = varMu + biasSqMu; //the MSE of mu

  bSigma = gsl_stats_mean(differenceSigma,1,nSim); //the bias of sigma
  biasSqSigma = pow(bSigma,2);
  varSigma = gsl_stats_variance(paramSigmaHat,1,nSim);
  MSESigma[2] = varSigma + biasSqSigma; //the MSE of sigma

  //LIT - Finney (JK)
  for(i=0; i<nSim; i++)
    {
      differenceMu[i] = muHat[3][i] - mu;
      differenceSigma[i] = sigmaHat[3][i] - sigma;
      paramMuHat[i] = muHat[3][i];
      paramSigmaHat[i] = sigmaHat[3][i];
    }

  bMu = gsl_stats_mean(differenceMu,1,nSim); //the bias of mu
  biasSqMu = pow(bMu,2);
  varMu = gsl_stats_variance(paramMuHat,1,nSim);
  MSEMu[3] = varMu + biasSqMu; //the MSE of mu

  bSigma = gsl_stats_mean(differenceSigma,1,nSim); //the bias of sigma
  biasSqSigma = pow(bSigma,2);
  varSigma = gsl_stats_variance(paramSigmaHat,1,nSim);
  MSESigma[3] = varSigma + biasSqSigma; //the MSE of sigma

}// end calcMSE()
```

A.2 Simulating Why the Method of Moments Estimator Biases Increase as n Increases when $\sigma = 10$

```
############################################################
###
###           File Name: simulateMOM.R
###
### Program Description: simulates the MOM estimator for
###                      varous lognormal data, and examines
###                      how the various parts are related
###                      when mu, sigma, and n are changed.
###
############################################################

set.seed(19)

nSim <- 10000
sampSizes <- c(10,25,100,500)

mu <- NULL
sigma <- NULL

muP1plusP2 <- NULL
neg1p5LogN <- NULL

sigP1plusP2 <- NULL
logN <- NULL

for(j in 1:4)
{
n <- sampSizes[j]
muPart1 <- NULL
muPart2 <- NULL
muPart3 <- NULL
sigPart1 <- NULL
sigPart2 <- NULL
sigPart3 <- NULL

muMOM <- NULL
muP2plusP1 <- NULL

sigmaMOM <- NULL

for(i in 1:nSim)
{
data <- rlnorm(n,3,10)

muPart1[i] <- -log(sum(data^2))/2
muPart2[i] <- 2*log(sum(data))
muPart3[i] <- -(3/2)*log(length(data))

muMOM[i] <- muPart1[i] + muPart2[i] + muPart3[i]
```

```
sigPart1[i] <- log(sum(data^2))
sigPart2[i] <- -2*log(sum(data))
sigPart3[i] <- log(length(data))

sigmaMOM[i] <- sqrt(sigPart1[i] + sigPart2[i] + sigPart3[i])

}

mu[j] <- mean(muMOM)
muP1plusP2[j] <- mean(muPart1 + muPart2)
neg1p5LogN[j] <- -1.5*log(n)

sigma[j] <- mean(sigmaMOM)
sigP1plusP2[j] <- mean(sigPart1 + sigPart2)
logN[j] <- log(n)

}

cbind(sampSizes,round(mu,3),round(muP1plusP2,3),round(neg1p5LogN,3),
                round(sigma,3),round(sigP1plusP2,3),round(logN,3))

####

mu <- NULL
sigma <- NULL

muP1plusP2 <- NULL
neg1p5LogN <- NULL

sigP1plusP2 <- NULL
logN <- NULL

for(j in 1:4)
{
n <- sampSizes[j]
muPart1 <- NULL
muPart2 <- NULL
muPart3 <- NULL
sigPart1 <- NULL
sigPart2 <- NULL
sigPart3 <- NULL

muMOM <- NULL
muP2plusP1 <- NULL

sigmaMOM <- NULL

for(i in 1:nSim)
{
data <- rlnorm(n,3,1)

muPart1[i] <- -log(sum(data^2))/2
muPart2[i] <- 2*log(sum(data))
```

```
muPart3[i] <- -(3/2)*log(length(data))

muMOM[i] <- muPart1[i] + muPart2[i] + muPart3[i]

sigPart1[i] <- log(sum(data^2))
sigPart2[i] <- -2*log(sum(data))
sigPart3[i] <- log(length(data))

sigmaMOM[i] <- sqrt(sigPart1[i] + sigPart2[i] + sigPart3[i])

}

mu[j] <- mean(muMOM)
muP1plusP2[j] <- mean(muPart1 + muPart2)
neg1p5LogN[j] <- -1.5*log(n)

sigma[j] <- mean(sigmaMOM)
sigP1plusP2[j] <- mean(sigPart1 + sigPart2)
logN[j] <- log(n)

}

cbind(sampSizes,round(mu,3),round(muP1plusP2,3),round(neg1p5LogN,3),
                round(sigma,3),round(sigP1plusP2,3),round(logN,3))
```

B. GRAPHICS CODE

B.1 Bias and MSE Plots

```
#############################################################
###
###           File Name: plotBiasMSE.R
###
### Program Description: plots the biases and MSEs retrieved
###                      from generate.c
###
#############################################################

mu <- read.table(’resultsMuCopy.txt’, header=T)
sigma <- read.table(’resultsSigmaCopy.txt’, header=T)

muBias.MLE <- mu[,5]
muBias.MOM <- mu[,8]
muBias.SERF <- mu[,11]
muBias.JK <- mu[,14]
muMSE.MLE <- mu[,6]
muMSE.MOM <- mu[,9]
muMSE.SERF <- mu[,12]
muMSE.JK <- mu[,15]

sigBias.MLE <- sigma[,5]
sigBias.MOM <- sigma[,8]
sigBias.SERF <- sigma[,11]
sigBias.JK <- sigma[,14]
sigMSE.MLE <- sigma[,6]
sigMSE.MOM <- sigma[,9]
sigMSE.SERF <- sigma[,12]
sigMSE.JK <- sigma[,15]

sampSize <- c(10,25,100,500)

minBias <- -0.1
maxBias <- 0.1
maxMSE <- 0.1



#######################
## group line plots: ##
#######################


plotAllMusMuBiasFx <- function(A,B,C,D,E,F,plotName,yLabel)
{
  pdf(paste(plotName,".pdf",sep=""),width=7,height=7)

  plot(sampSize, muBias.MLE[A:B], col='red',
    xlim=c(0,525), ylim=c(minBias,maxBias),
    xlab = "Sample Size",
    ylab = yLabel,
    type='b', lty=1)
  abline(h=0)
  points(sampSize, muBias.MLE[A:B], col='red', type='b', lty=1)
  points(sampSize, muBias.MOM[A:B], col='blue', type='b', lty=1)
  points(sampSize, muBias.SERF[A:B], col='chocolate1', type='b', lty=1)
  points(sampSize, muBias.JK[A:B], col='green', type='b', lty=1)

  points(sampSize, muBias.MLE[C:D], col='red', type='b', lty=2)
  points(sampSize, muBias.MOM[C:D], col='blue', type='b', lty=2)
  points(sampSize, muBias.SERF[C:D], col='chocolate1', type='b', lty=2)
  points(sampSize, muBias.JK[C:D], col='green', type='b', lty=2)

  points(sampSize, muBias.MLE[E:F], col='red', type='b', lty=3)
  points(sampSize, muBias.MOM[E:F], col='blue', type='b', lty=3)
  points(sampSize, muBias.SERF[E:F], col='chocolate1', type='b', lty=3)
  points(sampSize, muBias.JK[E:F], col='green', type='b', lty=3)

  legend(350,maxBias,c("MLE", "MOM", "Serfling", "Finney",
    expression(paste(mu, " = 2.5")), expression(paste(mu, " = 3")),
    expression(paste(mu, " = 3.5"))), lty=c(1,1,1,1,1,2,3),
    col=c("red", "blue", "chocolate1", "green", "black", "black", "black"),
    bg="white")

  dev.off()

  return
} # end plotAllMusMuBiasFx()

plotAllMusMuBiasFx(1,4,21,24,41,44,'plot_AllMus_MuBias_Sigma10',
  expression(paste("Bias of Estimators for ", mu, ", ", sigma, " = 10")))
plotAllMusMuBiasFx(5,8,25,28,45,48,'plot_AllMus_MuBias_Sigma1p5',
  expression(paste("Bias of Estimators for ", mu, ", ", sigma, " = 1.5")))
plotAllMusMuBiasFx(9,12,29,32,49,52,'plot_AllMus_MuBias_Sigma1',
  expression(paste("Bias of Estimators for ", mu, ", ", sigma, " = 1")))
plotAllMusMuBiasFx(13,16,33,36,53,56,'plot_AllMus_MuBias_SigmaP5',
  expression(paste("Bias of Estimators for ", mu, ", ", sigma, " = 0.5")))
plotAllMusMuBiasFx(17,20,37,40,57,60,'plot_AllMus_MuBias_SigmaP25',
  expression(paste("Bias of Estimators for ", mu, ", ", sigma, " = 0.25")))



plotAllMusMuMSEFx <- function(A,B,C,D,E,F,plotName,yLabel)
{

  pdf(paste(plotName,".pdf",sep=""),width=7,height=7)

  plot(sampSize, muMSE.MLE[A:B], col='red',
    xlim=c(0,525), ylim=c(0,maxMSE),
    xlab = "Sample Size",
    ylab = yLabel,
    type='b', lty=1)
  abline(h=0)
  points(sampSize, muMSE.MLE[A:B], col='red', type='b', lty=1)
  points(sampSize, muMSE.MOM[A:B], col='blue', type='b', lty=1)
  points(sampSize, muMSE.SERF[A:B], col='chocolate1', type='b', lty=1)
  points(sampSize, muMSE.JK[A:B], col='green', type='b', lty=1)

  points(sampSize, muMSE.MLE[C:D], col='red', type='b', lty=2)
  points(sampSize, muMSE.MOM[C:D], col='blue', type='b', lty=2)
  points(sampSize, muMSE.SERF[C:D], col='chocolate1', type='b', lty=2)
  points(sampSize, muMSE.JK[C:D], col='green', type='b', lty=2)

  points(sampSize, muMSE.MLE[E:F], col='red', type='b', lty=3)
  points(sampSize, muMSE.MOM[E:F], col='blue', type='b', lty=3)
  points(sampSize, muMSE.SERF[E:F], col='chocolate1', type='b', lty=3)
  points(sampSize, muMSE.JK[E:F], col='green', type='b', lty=3)

  legend(350,maxMSE,c("MLE", "MOM", "Serfling", "Finney",
    expression(paste(mu, " = 2.5")), expression(paste(mu, " = 3")),
    expression(paste(mu, " = 3.5"))), lty=c(1,1,1,1,1,2,3),
    col=c("red", "blue", "chocolate1", "green", "black", "black", "black"),
    bg="white")

  dev.off()

  return
} # end plotAllMusMuMSEFx()

plotAllMusMuMSEFx(1,4,21,24,41,44,'plot_AllMus_MuMSE_Sigma10',
  expression(paste("MSE of Estimators for ", mu, ", ", sigma, " = 10")))
plotAllMusMuMSEFx(5,8,25,28,45,48,'plot_AllMus_MuMSE_Sigma1p5',
  expression(paste("MSE of Estimators for ", mu, ", ", sigma, " = 1.5")))
plotAllMusMuMSEFx(9,12,29,32,49,52,'plot_AllMus_MuMSE_Sigma1',
  expression(paste("MSE of Estimators for ", mu, ", ", sigma, " = 1")))
plotAllMusMuMSEFx(13,16,33,36,53,56,'plot_AllMus_MuMSE_SigmaP5',
  expression(paste("MSE of Estimators for ", mu, ", ", sigma, " = 0.5")))
plotAllMusMuMSEFx(17,20,37,40,57,60,'plot_AllMus_MuMSE_SigmaP25',
  expression(paste("MSE of Estimators for ", mu, ", ", sigma, " = 0.25")))



plotAllMusSigmaBiasFx <- function(A,B,C,D,E,F,plotName,yLabel)
{
  pdf(paste(plotName,".pdf",sep=""),width=7,height=7)

  plot(sampSize, sigBias.MLE[A:B], col='red',
    xlim=c(0,525), ylim=c(minBias,maxBias),
    xlab = "Sample Size",
    ylab = yLabel,
    type='b', lty=1)
  abline(h=0)
  points(sampSize, sigBias.MLE[A:B], col='red', type='b', lty=1)
  points(sampSize, sigBias.MOM[A:B], col=’blue’, type=’b’, lty=1)
  points(sampSize, sigBias.SERF[A:B], col=’chocolate1’, type=’b’, lty=1)
  points(sampSize, sigBias.JK[A:B], col=’green’, type=’b’, lty=1)

  points(sampSize, sigBias.MLE[C:D], col=’red’, type=’b’, lty=2)
  points(sampSize, sigBias.MOM[C:D], col=’blue’, type=’b’, lty=2)
  points(sampSize, sigBias.SERF[C:D], col=’chocolate1’, type=’b’, lty=2)
  points(sampSize, sigBias.JK[C:D], col=’green’, type=’b’, lty=2)

  points(sampSize, sigBias.MLE[E:F], col=’red’, type=’b’, lty=3)
  points(sampSize, sigBias.MOM[E:F], col=’blue’, type=’b’, lty=3)
  points(sampSize, sigBias.SERF[E:F], col=’chocolate1’, type=’b’, lty=3)
  points(sampSize, sigBias.JK[E:F], col=’green’, type=’b’, lty=3)

  legend(350,maxBias,c("MLE", "MOM", "Serfling", "Finney",
    expression(paste(mu, " = 2.5")), expression(paste(mu, " = 3")),
    expression(paste(mu, " = 3.5"))), lty=c(1,1,1,1,1,2,3),
    col=c("red", "blue", "chocolate1", "green", "black", "black", "black"),
    bg="white")

  dev.off()

  return
} # end plotAllMusSigmaBiasFx()

plotAllMusSigmaBiasFx(1,4,21,24,41,44,’plot_AllMus_SigmaBias_Sigma10’,
  expression(paste("Bias of Estimators for ", sigma, ", ", sigma, " = 10")))
plotAllMusSigmaBiasFx(5,8,25,28,45,48,’plot_AllMus_SigmaBias_Sigma1p5’,
  expression(paste("Bias of Estimators for ", sigma, ", ", sigma, " = 1.5")))
plotAllMusSigmaBiasFx(9,12,29,32,49,52,’plot_AllMus_SigmaBias_Sigma1’,
  expression(paste("Bias of Estimators for ", sigma, ", ", sigma, " = 1")))
plotAllMusSigmaBiasFx(13,16,33,36,53,56,’plot_AllMus_SigmaBias_SigmaP5’,
  expression(paste("Bias of Estimators for ", sigma, ", ", sigma, " = 0.5")))
plotAllMusSigmaBiasFx(17,20,37,40,57,60,’plot_AllMus_SigmaBias_SigmaP25’,
  expression(paste("Bias of Estimators for ", sigma, ", ", sigma, " = 0.25")))



plotAllMusSigmaMSEFx <- function(A,B,C,D,E,F,plotName,yLabel)
{
  pdf(paste(plotName,".pdf",sep=""),width=7,height=7)

  plot(sampSize, sigMSE.MLE[A:B], col=’red’,
    xlim=c(0,525), ylim=c(0,maxMSE),
    xlab = "Sample Size",
    ylab = yLabel,
    type=’b’, lty=1)
  abline(h=0)
  points(sampSize, sigMSE.MLE[A:B], col=’red’, type=’b’, lty=1)
  points(sampSize, sigMSE.MOM[A:B], col=’blue’, type=’b’, lty=1)
  points(sampSize, sigMSE.SERF[A:B], col=’chocolate1’, type=’b’, lty=1)
  points(sampSize, sigMSE.JK[A:B], col=’green’, type=’b’, lty=1)

  points(sampSize, sigMSE.MLE[C:D], col=’red’, type=’b’, lty=2)
  points(sampSize, sigMSE.MOM[C:D], col='blue', type='b', lty=2)
  points(sampSize, sigMSE.SERF[C:D], col='chocolate1', type='b', lty=2)
  points(sampSize, sigMSE.JK[C:D], col='green', type='b', lty=2)

  points(sampSize, sigMSE.MLE[E:F], col='red', type='b', lty=3)
  points(sampSize, sigMSE.MOM[E:F], col='blue', type='b', lty=3)
  points(sampSize, sigMSE.SERF[E:F], col='chocolate1', type='b', lty=3)
  points(sampSize, sigMSE.JK[E:F], col='green', type='b', lty=3)

  legend(350,maxMSE,c("MLE", "MOM", "Serfling", "Finney",
    expression(paste(mu, " = 2.5")), expression(paste(mu, " = 3")),
    expression(paste(mu, " = 3.5"))), lty=c(1,1,1,1,1,2,3),
    col=c("red", "blue", "chocolate1", "green", "black", "black", "black"),
    bg="white")

  dev.off()

  return
} # end plotAllMusSigmaMSEFx()

plotAllMusSigmaMSEFx(1,4,21,24,41,44,'plot_AllMus_SigmaMSE_Sigma10',
  expression(paste("MSE of Estimators for ", sigma, ", ", sigma, " = 10")))
plotAllMusSigmaMSEFx(5,8,25,28,45,48,'plot_AllMus_SigmaMSE_Sigma1p5',
  expression(paste("MSE of Estimators for ", sigma, ", ", sigma, " = 1.5")))
plotAllMusSigmaMSEFx(9,12,29,32,49,52,'plot_AllMus_SigmaMSE_Sigma1',
  expression(paste("MSE of Estimators for ", sigma, ", ", sigma, " = 1")))
plotAllMusSigmaMSEFx(13,16,33,36,53,56,'plot_AllMus_SigmaMSE_SigmaP5',
  expression(paste("MSE of Estimators for ", sigma, ", ", sigma, " = 0.5")))
plotAllMusSigmaMSEFx(17,20,37,40,57,60,'plot_AllMus_SigmaMSE_SigmaP25',
  expression(paste("MSE of Estimators for ", sigma, ", ", sigma, " = 0.25")))



plotAllSigmasMuBiasFx <- function(A,B,C,D,E,F,G,H,I,J,plotName,yLabel)
{
  pdf(paste(plotName,".pdf",sep=""),width=7,height=7)

  plot(sampSize, muBias.MLE[A:B], col='red',
    xlim=c(0,525), ylim=c(minBias,maxBias),
    xlab = "Sample Size",
    ylab = yLabel,
    type='b', lty=1)
  abline(h=0)
  points(sampSize, muBias.MLE[A:B], col='red', type='b', lty=1)
  points(sampSize, muBias.MOM[A:B], col='blue', type='b', lty=1)
  points(sampSize, muBias.SERF[A:B], col='chocolate1', type='b', lty=1)
  points(sampSize, muBias.JK[A:B], col='green', type='b', lty=1)

  points(sampSize, muBias.MLE[C:D], col='red', type='b', lty=5)
  points(sampSize, muBias.MOM[C:D], col='blue', type='b', lty=5)
  points(sampSize, muBias.SERF[C:D], col='chocolate1', type='b', lty=5)
  points(sampSize, muBias.JK[C:D], col='green', type='b', lty=5)

  points(sampSize, muBias.MLE[E:F], col='red', type='b', lty=2)
  points(sampSize, muBias.MOM[E:F], col=’blue’, type=’b’, lty=2)
  points(sampSize, muBias.SERF[E:F], col=’chocolate1’, type=’b’, lty=2)
  points(sampSize, muBias.JK[E:F], col=’green’, type=’b’, lty=2)

  points(sampSize, muBias.MLE[G:H], col=’red’, type=’b’, lty=4)
  points(sampSize, muBias.MOM[G:H], col=’blue’, type=’b’, lty=4)
  points(sampSize, muBias.SERF[G:H], col=’chocolate1’, type=’b’, lty=4)
  points(sampSize, muBias.JK[G:H], col=’green’, type=’b’, lty=4)

  points(sampSize, muBias.MLE[I:J], col=’red’, type=’b’, lty=3)
  points(sampSize, muBias.MOM[I:J], col=’blue’, type=’b’, lty=3)
  points(sampSize, muBias.SERF[I:J], col=’chocolate1’, type=’b’, lty=3)
  points(sampSize, muBias.JK[I:J], col=’green’, type=’b’, lty=3)

  legend(350,maxBias,c("MLE", "MOM", "Serfling", "Finney",
    expression(paste(sigma, " = 10")), expression(paste(sigma, " = 1.5")),
    expression(paste(sigma, " = 1")), expression(paste(sigma, " = 0.5")),
    expression(paste(sigma, " = 0.25"))), lty=c(1,1,1,1,1,5,2,4,3),
    col=c("red", "blue", "chocolate1", "green",
    "black", "black", "black", "black", "black"), bg="white")

  dev.off()

  return
} # end plotAllSigmasMuBiasFx()

plotAllSigmasMuBiasFx(1,4,5,8,9,12,13,16,17,20,’plot_AllSigmas_MuBias_Mu2p5’,
  expression(paste("Bias of Estimators for ", mu, ", ", mu, " = 2.5")))
plotAllSigmasMuBiasFx(21,24,25,28,29,32,33,36,37,40,’plot_AllSigmas_MuBias_Mu3’,
  expression(paste("Bias of Estimators for ", mu, ", ", mu, " = 3")))
plotAllSigmasMuBiasFx(41,44,45,48,49,52,53,56,57,60,’plot_AllSigmas_MuBias_Mu3p5’,
  expression(paste("Bias of Estimators for ", mu, ", ", mu, " = 3.5")))



plotAllSigmasMuMSEFx <- function(A,B,C,D,E,F,G,H,I,J,plotName,yLabel)
{
  pdf(paste(plotName,".pdf",sep=""),width=7,height=7)

  plot(sampSize, muMSE.MLE[A:B], col=’red’,
    xlim=c(0,525), ylim=c(0,maxMSE),
    xlab = "Sample Size",
    ylab = yLabel,
    type=’b’, lty=1)
  abline(h=0)
  points(sampSize, muMSE.MLE[A:B], col=’red’, type=’b’, lty=1)
  points(sampSize, muMSE.MOM[A:B], col=’blue’, type=’b’, lty=1)
  points(sampSize, muMSE.SERF[A:B], col=’chocolate1’, type=’b’, lty=1)
  points(sampSize, muMSE.JK[A:B], col=’green’, type=’b’, lty=1)

  points(sampSize, muMSE.MLE[C:D], col=’red’, type=’b’, lty=5)
  points(sampSize, muMSE.MOM[C:D], col=’blue’, type=’b’, lty=5)
  points(sampSize, muMSE.SERF[C:D], col=’chocolate1’, type=’b’, lty=5)
  points(sampSize, muMSE.JK[C:D], col=’green’, type=’b’, lty=5)
  points(sampSize, muMSE.MLE[E:F], col=’red’, type=’b’, lty=2)
  points(sampSize, muMSE.MOM[E:F], col=’blue’, type=’b’, lty=2)
  points(sampSize, muMSE.SERF[E:F], col=’chocolate1’, type=’b’, lty=2)
  points(sampSize, muMSE.JK[E:F], col=’green’, type=’b’, lty=2)

  points(sampSize, muMSE.MLE[G:H], col=’red’, type=’b’, lty=4)
  points(sampSize, muMSE.MOM[G:H], col=’blue’, type=’b’, lty=4)
  points(sampSize, muMSE.SERF[G:H], col=’chocolate1’, type=’b’, lty=4)
  points(sampSize, muMSE.JK[G:H], col=’green’, type=’b’, lty=4)

  points(sampSize, muMSE.MLE[I:J], col=’red’, type=’b’, lty=3)
  points(sampSize, muMSE.MOM[I:J], col=’blue’, type=’b’, lty=3)
  points(sampSize, muMSE.SERF[I:J], col=’chocolate1’, type=’b’, lty=3)
  points(sampSize, muMSE.JK[I:J], col=’green’, type=’b’, lty=3)

  legend(350,maxMSE,c("MLE", "MOM", "Serfling", "Finney",
    expression(paste(sigma, " = 10")), expression(paste(sigma, " = 1.5")),
    expression(paste(sigma, " = 1")), expression(paste(sigma, " = 0.5")),
    expression(paste(sigma, " = 0.25"))), lty=c(1,1,1,1,1,5,2,4,3),
    col=c("red", "blue", "chocolate1", "green",
    "black", "black", "black", "black", "black"), bg="white")

  dev.off()

  return
} # end plotAllSigmasMuMSEFx()

plotAllSigmasMuMSEFx(1,4,5,8,9,12,13,16,17,20,’plot_AllSigmas_MuMSE_Mu2p5’,
  expression(paste("MSE of Estimators for ", mu, ", ", mu, " = 2.5")))
plotAllSigmasMuMSEFx(21,24,25,28,29,32,33,36,37,40,’plot_AllSigmas_MuMSE_Mu3’,
  expression(paste("MSE of Estimators for ", mu, ", ", mu, " = 3")))
plotAllSigmasMuMSEFx(41,44,45,48,49,52,53,56,57,60,’plot_AllSigmas_MuMSE_Mu3p5’,
  expression(paste("MSE of Estimators for ", mu, ", ", mu, " = 3.5")))



plotAllSigmasSigmaBiasFx <- function(A,B,C,D,E,F,G,H,I,J,plotName,yLabel)
{

  pdf(paste(plotName,".pdf",sep=""),width=7,height=7)

  plot(sampSize, sigBias.MLE[A:B], col=’red’,
    xlim=c(0,525), ylim=c(minBias,maxBias),
    xlab = "Sample Size",
    ylab = yLabel,
    type=’b’, lty=1)
  abline(h=0)
  points(sampSize, sigBias.MLE[A:B], col=’red’, type=’b’, lty=1)
  points(sampSize, sigBias.MOM[A:B], col=’blue’, type=’b’, lty=1)
  points(sampSize, sigBias.SERF[A:B], col=’chocolate1’, type=’b’, lty=1)
  points(sampSize, sigBias.JK[A:B], col=’green’, type=’b’, lty=1)

  points(sampSize, sigBias.MLE[C:D], col=’red’, type=’b’, lty=5)
  points(sampSize, sigBias.MOM[C:D], col=’blue’, type=’b’, lty=5)
  points(sampSize, sigBias.SERF[C:D], col=’chocolate1’, type=’b’, lty=5)
  points(sampSize, sigBias.JK[C:D], col=’green’, type=’b’, lty=5)

  points(sampSize, sigBias.MLE[E:F], col=’red’, type=’b’, lty=2)
  points(sampSize, sigBias.MOM[E:F], col=’blue’, type=’b’, lty=2)
  points(sampSize, sigBias.SERF[E:F], col=’chocolate1’, type=’b’, lty=2)
  points(sampSize, sigBias.JK[E:F], col=’green’, type=’b’, lty=2)

  points(sampSize, sigBias.MLE[G:H], col=’red’, type=’b’, lty=4)
  points(sampSize, sigBias.MOM[G:H], col=’blue’, type=’b’, lty=4)
  points(sampSize, sigBias.SERF[G:H], col=’chocolate1’, type=’b’, lty=4)
  points(sampSize, sigBias.JK[G:H], col=’green’, type=’b’, lty=4)

  points(sampSize, sigBias.MLE[I:J], col=’red’, type=’b’, lty=3)
  points(sampSize, sigBias.MOM[I:J], col=’blue’, type=’b’, lty=3)
  points(sampSize, sigBias.SERF[I:J], col=’chocolate1’, type=’b’, lty=3)
  points(sampSize, sigBias.JK[I:J], col=’green’, type=’b’, lty=3)

  legend(350,maxBias,c("MLE", "MOM", "Serfling", "Finney",
    expression(paste(sigma, " = 10")), expression(paste(sigma, " = 1.5")),
    expression(paste(sigma, " = 1")), expression(paste(sigma, " = 0.5")),
    expression(paste(sigma, " = 0.25"))), lty=c(1,1,1,1,1,5,2,4,3),
    col=c("red", "blue", "chocolate1", "green",
    "black", "black", "black", "black", "black"), bg="white")

  dev.off()

  return
} # end plotAllSigmasSigmaBiasFx()

plotAllSigmasSigmaBiasFx(1,4,5,8,9,12,13,16,17,20,
  ’plot_AllSigmas_SigmaBias_Mu2p5’,
  expression(paste("Bias of Estimators for ", sigma, ", ", mu, " = 2.5")))
plotAllSigmasSigmaBiasFx(21,24,25,28,29,32,33,36,37,40,
  ’plot_AllSigmas_SigmaBias_Mu3’,
  expression(paste("Bias of Estimators for ", sigma, ", ", mu, " = 3")))
plotAllSigmasSigmaBiasFx(41,44,45,48,49,52,53,56,57,60,
  ’plot_AllSigmas_SigmaBias_Mu3p5’,
  expression(paste("Bias of Estimators for ", sigma, ", ", mu, " = 3.5")))



plotAllSigmasSigmaMSEFx <- function(A,B,C,D,E,F,G,H,I,J,plotName,yLabel)
{

  pdf(paste(plotName,".pdf",sep=""),width=7,height=7)

  plot(sampSize, sigMSE.MLE[A:B], col=’red’,
    xlim=c(0,525), ylim=c(0,maxMSE),
    xlab = "Sample Size",
    ylab = yLabel,
    type=’b’, lty=1)
  abline(h=0)
  points(sampSize, sigMSE.MLE[A:B], col=’red’, type=’b’, lty=1)
  points(sampSize, sigMSE.MOM[A:B], col=’blue’, type=’b’, lty=1)
  points(sampSize, sigMSE.SERF[A:B], col=’chocolate1’, type=’b’, lty=1)
  points(sampSize, sigMSE.JK[A:B], col=’green’, type=’b’, lty=1)

  points(sampSize, sigMSE.MLE[C:D], col=’red’, type=’b’, lty=5)
  points(sampSize, sigMSE.MOM[C:D], col=’blue’, type=’b’, lty=5)
  points(sampSize, sigMSE.SERF[C:D], col=’chocolate1’, type=’b’, lty=5)
  points(sampSize, sigMSE.JK[C:D], col=’green’, type=’b’, lty=5)

  points(sampSize, sigMSE.MLE[E:F], col=’red’, type=’b’, lty=2)
  points(sampSize, sigMSE.MOM[E:F], col=’blue’, type=’b’, lty=2)
  points(sampSize, sigMSE.SERF[E:F], col=’chocolate1’, type=’b’, lty=2)
  points(sampSize, sigMSE.JK[E:F], col=’green’, type=’b’, lty=2)

  points(sampSize, sigMSE.MLE[G:H], col=’red’, type=’b’, lty=4)
  points(sampSize, sigMSE.MOM[G:H], col=’blue’, type=’b’, lty=4)
  points(sampSize, sigMSE.SERF[G:H], col=’chocolate1’, type=’b’, lty=4)
  points(sampSize, sigMSE.JK[G:H], col=’green’, type=’b’, lty=4)

  points(sampSize, sigMSE.MLE[I:J], col=’red’, type=’b’, lty=3)
  points(sampSize, sigMSE.MOM[I:J], col=’blue’, type=’b’, lty=3)
  points(sampSize, sigMSE.SERF[I:J], col=’chocolate1’, type=’b’, lty=3)
  points(sampSize, sigMSE.JK[I:J], col=’green’, type=’b’, lty=3)

  legend(350,maxMSE,c("MLE", "MOM", "Serfling", "Finney",
    expression(paste(sigma, " = 10")), expression(paste(sigma, " = 1.5")),
    expression(paste(sigma, " = 1")), expression(paste(sigma, " = 0.5")),
    expression(paste(sigma, " = 0.25"))), lty=c(1,1,1,1,1,5,2,4,3),
    col=c("red", "blue", "chocolate1", "green",
    "black", "black", "black", "black", "black"), bg="white")

  dev.off()

  return
} # end plotAllSigmasSigmaMSEFx()

plotAllSigmasSigmaMSEFx(1,4,5,8,9,12,13,16,17,20,’plot_AllSigmas_SigmaMSE_Mu2p5’,
  expression(paste("MSE of Estimators for ", sigma, ", ", mu, " = 2.5")))
plotAllSigmasSigmaMSEFx(21,24,25,28,29,32,33,36,37,40,’plot_AllSigmas_SigmaMSE_Mu3’,
  expression(paste("MSE of Estimators for ", sigma, ", ", mu, " = 3")))
plotAllSigmasSigmaMSEFx(41,44,45,48,49,52,53,56,57,60,’plot_AllSigmas_SigmaMSE_Mu3p5’,
  expression(paste("MSE of Estimators for ", sigma, ", ", mu, " = 3.5")))



#######################
## other plots used: ##
#######################

### MOM results

plotMleMomMusMuBiasFx <- function(A,B,C,D,E,F,plotName,yLabel)
{
  pdf(paste(plotName,"_MleMom.pdf",sep=""),width=7,height=7)

  plot(sampSize, muBias.MLE[A:B], col=’red’,
    xlim=c(0,525), ylim=c(minBias,maxBias),
    xlab = "Sample Size",
    ylab = yLabel,
    type=’b’, lty=1)
  abline(h=0)
  points(sampSize, muBias.MLE[A:B], col=’red’, type=’b’, lty=1)
  points(sampSize, muBias.MOM[A:B], col=’blue’, type=’b’, lty=1)

  points(sampSize, muBias.MLE[C:D], col=’red’, type=’b’, lty=2)
  points(sampSize, muBias.MOM[C:D], col=’blue’, type=’b’, lty=2)

  points(sampSize, muBias.MLE[E:F], col=’red’, type=’b’, lty=3)
  points(sampSize, muBias.MOM[E:F], col=’blue’, type=’b’, lty=3)

  legend(350,maxBias,c("MLE", "MOM", expression(paste(mu, " = 2.5")),
    expression(paste(mu, " = 3")), expression(paste(mu, " = 3.5"))),
    lty=c(1,1,1,2,3), col=c("red", "blue", "black", "black", "black"),
    bg="white")

  dev.off()

  return
} # end plotMleMomMusMuBiasFx()

plotMleMomMusMuBiasFx(9,12,29,32,49,52,’plot_AllMus_MuBias_Sigma1’,
  expression(paste("Bias of Estimators for ", mu, ", ", sigma, " = 1")))
plotMleMomMusMuBiasFx(17,20,37,40,57,60,’plot_AllMus_MuBias_SigmaP25’,
  expression(paste("Bias of Estimators for ", mu, ", ", sigma, " = 0.25")))


plotMleMomMusSigmaBiasNoLgndFx <- function(A,B,C,D,E,F,plotName,yLabel)
{
  pdf(paste(plotName,"_noLgnd_MleMom.pdf",sep=""),width=7,height=7)

  plot(sampSize, sigBias.MLE[A:B], col=’red’,
    xlim=c(0,525), ylim=c(minBias,maxBias),
    xlab = "Sample Size",
    ylab = yLabel,
    type=’b’, lty=1)
  abline(h=0)
  points(sampSize, sigBias.MLE[A:B], col=’red’, type=’b’, lty=1)
  points(sampSize, sigBias.MOM[A:B], col=’blue’, type=’b’, lty=1)

  points(sampSize, sigBias.MLE[C:D], col=’red’, type=’b’, lty=2)
  points(sampSize, sigBias.MOM[C:D], col=’blue’, type=’b’, lty=2)

  points(sampSize, sigBias.MLE[E:F], col=’red’, type=’b’, lty=3)
  points(sampSize, sigBias.MOM[E:F], col=’blue’, type=’b’, lty=3)

  legend(350,maxBias,c(expression(paste(mu, " = 2.5")),
    expression(paste(mu, " = 3")), expression(paste(mu, " = 3.5"))),
    lty=c(1,2,3), col=c("black", "black", "black"), bg="white")

  dev.off()

  return
} # end plotMleMomMusSigmaBiasNoLgndFx()

plotMleMomMusSigmaBiasNoLgndFx(9,12,29,32,49,52,'plot_AllMus_SigmaBias_Sigma1',
  expression(paste("Bias of Estimators for ", sigma, ", ", sigma, " = 1")))


plotMleMomMusMuMSENoLgndFx <- function(A,B,C,D,E,F,plotName,yLabel)
{

  pdf(paste(plotName,"_noLgnd_MleMom.pdf",sep=""),width=7,height=7)

  plot(sampSize, muMSE.MLE[A:B], col='red',
    xlim=c(0,525), ylim=c(0,maxMSE),
    xlab = "Sample Size",
    ylab = yLabel,
    type='b', lty=1)
  abline(h=0)
  points(sampSize, muMSE.MLE[A:B], col='red', type='b', lty=1)
  points(sampSize, muMSE.MOM[A:B], col='blue', type='b', lty=1)

  points(sampSize, muMSE.MLE[C:D], col='red', type='b', lty=2)
  points(sampSize, muMSE.MOM[C:D], col='blue', type='b', lty=2)

  points(sampSize, muMSE.MLE[E:F], col='red', type='b', lty=3)
  points(sampSize, muMSE.MOM[E:F], col='blue', type='b', lty=3)

  legend(350,maxBias,c(expression(paste(mu, " = 2.5")),
    expression(paste(mu, " = 3")), expression(paste(mu, " = 3.5"))),
    lty=c(1,2,3), col=c("black", "black", "black"), bg="white")

  dev.off()

  return
} # end plotMleMomMusMuMSENoLgndFx()

plotMleMomMusMuMSENoLgndFx(9,12,29,32,49,52,'plot_AllMus_MuMSE_Sigma1',
  expression(paste("MSE of Estimators for ", mu, ", ", sigma, " = 1")))
plotMleMomMusMuMSENoLgndFx(13,16,33,36,53,56,'plot_AllMus_MuMSE_SigmaP5',
  expression(paste("MSE of Estimators for ", mu, ", ", sigma, " = 0.5")))
plotMleMomMusMuMSENoLgndFx(17,20,37,40,57,60,'plot_AllMus_MuMSE_SigmaP25',
  expression(paste("MSE of Estimators for ", mu, ", ", sigma, " = 0.25")))


plotMleMomMusSigmaMSENoLgndFx <- function(A,B,C,D,E,F,plotName,yLabel)
{
  pdf(paste(plotName,"_noLgnd_MleMom.pdf",sep=""),width=7,height=7)

  plot(sampSize, sigMSE.MLE[A:B], col='red',
    xlim=c(0,525), ylim=c(0,maxMSE),
    xlab = "Sample Size",
    ylab = yLabel,
    type='b', lty=1)
  abline(h=0)
  points(sampSize, sigMSE.MLE[A:B], col='red', type='b', lty=1)
  points(sampSize, sigMSE.MOM[A:B], col='blue', type='b', lty=1)

  points(sampSize, sigMSE.MLE[C:D], col='red', type='b', lty=2)
  points(sampSize, sigMSE.MOM[C:D], col='blue', type='b', lty=2)

  points(sampSize, sigMSE.MLE[E:F], col='red', type='b', lty=3)
  points(sampSize, sigMSE.MOM[E:F], col='blue', type='b', lty=3)

  legend(350,maxBias,c(expression(paste(mu, " = 2.5")),
    expression(paste(mu, " = 3")), expression(paste(mu, " = 3.5"))),
    lty=c(1,2,3), col=c("black", "black", "black"), bg="white")

  dev.off()

  return
} # end plotMleMomMusSigmaMSENoLgndFx()

plotMleMomMusSigmaMSENoLgndFx(9,12,29,32,49,52,'plot_AllMus_SigmaMSE_Sigma1',
  expression(paste("MSE of Estimators for ", sigma, ", ", sigma, " = 1")))
plotMleMomMusSigmaMSENoLgndFx(13,16,33,36,53,56,'plot_AllMus_SigmaMSE_SigmaP5',
  expression(paste("MSE of Estimators for ", sigma, ", ", sigma, " = 0.5")))



### Serfling results

plotMleSerfMusMuBiasFx <- function(A,B,C,D,E,F,plotName,yLabel)
{

  pdf(paste(plotName,"_MleSerf.pdf",sep=""),width=7,height=7)

  plot(sampSize, muBias.MLE[A:B], col='red',
    xlim=c(0,525), ylim=c(minBias,maxBias),
    xlab = "Sample Size",
    ylab = yLabel,
    type='b', lty=1)
  abline(h=0)
  points(sampSize, muBias.MLE[A:B], col='red', type='b', lty=1)
  points(sampSize, muBias.SERF[A:B], col='chocolate1', type='b', lty=1)

  points(sampSize, muBias.MLE[C:D], col='red', type='b', lty=2)
  points(sampSize, muBias.SERF[C:D], col='chocolate1', type='b', lty=2)

  points(sampSize, muBias.MLE[E:F], col='red', type='b', lty=3)
  points(sampSize, muBias.SERF[E:F], col='chocolate1', type='b', lty=3)

  legend(350,maxBias,c("MLE", "Serfling", expression(paste(mu, " = 2.5")),
    expression(paste(mu, " = 3")), expression(paste(mu, " = 3.5"))),
    lty=c(1,1,1,2,3), col=c("red", "chocolate1", "black", "black", "black"),
    bg="white")

  dev.off()

  return
} # end plotMleSerfMusMuBiasFx()

plotMleSerfMusMuBiasFx(9,12,29,32,49,52,'plot_AllMus_MuBias_Sigma1',
  expression(paste("Bias of Estimators for ", mu, ", ", sigma, " = 1")))


plotMleSerfSigmasMuMSENoLgndFx <- function(A,B,C,D,E,F,G,H,I,J,plotName,yLabel)
{
  pdf(paste(plotName,"_noLgnd_MleSerf.pdf",sep=""),width=7,height=7)

  plot(sampSize, muMSE.MLE[A:B], col='red',
    xlim=c(0,525), ylim=c(0,maxMSE),
    xlab = "Sample Size",
    ylab = yLabel,
    type='b', lty=1)
  abline(h=0)
  points(sampSize, muMSE.MLE[A:B], col='red', type='b', lty=1)
  points(sampSize, muMSE.SERF[A:B], col='chocolate1', type='b', lty=1)

  points(sampSize, muMSE.MLE[C:D], col='red', type='b', lty=5)
  points(sampSize, muMSE.SERF[C:D], col='chocolate1', type='b', lty=5)

  points(sampSize, muMSE.MLE[E:F], col='red', type='b', lty=2)
  points(sampSize, muMSE.SERF[E:F], col='chocolate1', type='b', lty=2)

  points(sampSize, muMSE.MLE[G:H], col='red', type='b', lty=4)
  points(sampSize, muMSE.SERF[G:H], col='chocolate1', type='b', lty=4)

  points(sampSize, muMSE.MLE[I:J], col='red', type='b', lty=3)
  points(sampSize, muMSE.SERF[I:J], col='chocolate1', type='b', lty=3)

  legend(350,maxMSE,c(expression(paste(sigma, " = 10")),
    expression(paste(sigma, " = 1.5")), expression(paste(sigma, " = 1")),
    expression(paste(sigma, " = 0.5")), expression(paste(sigma, " = 0.25"))),
    lty=c(1,5,2,4,3), col=c("black", "black", "black", "black", "black"),
    bg="white")

  dev.off()

  return
} # end plotMleSerfSigmasMuMSENoLgndFx()

plotMleSerfSigmasMuMSENoLgndFx(21,24,25,28,29,32,33,36,37,40,'plot_AllSigmas_MuMSE_Mu3',
  expression(paste("MSE of Estimators for ", mu, ", ", mu, " = 3")))


plotMleSerfMusSigmaBiasNoLgndFx <- function(A,B,C,D,E,F,plotName,yLabel)
{
  pdf(paste(plotName,"_noLgnd_MleSerf.pdf",sep=""),width=7,height=7)

  plot(sampSize, sigBias.MLE[A:B], col='red',
    xlim=c(0,525), ylim=c(minBias,maxBias),
    xlab = "Sample Size",
    ylab = yLabel,
    type='b', lty=1)
  abline(h=0)
  points(sampSize, sigBias.MLE[A:B], col='red', type='b', lty=1)
  points(sampSize, sigBias.SERF[A:B], col='chocolate1', type='b', lty=1)

  points(sampSize, sigBias.MLE[C:D], col='red', type='b', lty=2)
  points(sampSize, sigBias.SERF[C:D], col='chocolate1', type='b', lty=2)

  points(sampSize, sigBias.MLE[E:F], col='red', type='b', lty=3)
  points(sampSize, sigBias.SERF[E:F], col='chocolate1', type='b', lty=3)

  legend(350,maxBias,c(expression(paste(mu, " = 2.5")),
    expression(paste(mu, " = 3")), expression(paste(mu, " = 3.5"))),
    lty=c(1,2,3), col=c("black", "black", "black"), bg="white")

  dev.off()

  return
} # end plotMleSerfMusSigmaBiasNoLgndFx()

plotMleSerfMusSigmaBiasNoLgndFx(17,20,37,40,57,60,'plot_AllMus_SigmaBias_SigmaP25',
  expression(paste("Bias of Estimators for ", sigma, ", ", sigma, " = 0.25")))


plotMleSerfSigmasSigmaMSENoLgndFx <- function(A,B,C,D,E,F,G,H,I,J,plotName,yLabel)
{

  pdf(paste(plotName,"_noLgnd_MleSerf.pdf",sep=""),width=7,height=7)

  plot(sampSize, sigMSE.MLE[A:B], col='red',
    xlim=c(0,525), ylim=c(0,maxMSE),
    xlab = "Sample Size",
    ylab = yLabel,
    type='b', lty=1)
  abline(h=0)
  points(sampSize, sigMSE.MLE[A:B], col='red', type='b', lty=1)
  points(sampSize, sigMSE.SERF[A:B], col='chocolate1', type='b', lty=1)

  points(sampSize, sigMSE.MLE[C:D], col='red', type='b', lty=5)
  points(sampSize, sigMSE.SERF[C:D], col='chocolate1', type='b', lty=5)

  points(sampSize, sigMSE.MLE[E:F], col='red', type='b', lty=2)
  points(sampSize, sigMSE.SERF[E:F], col='chocolate1', type='b', lty=2)

  points(sampSize, sigMSE.MLE[G:H], col='red', type='b', lty=4)
  points(sampSize, sigMSE.SERF[G:H], col='chocolate1', type='b', lty=4)

  points(sampSize, sigMSE.MLE[I:J], col='red', type='b', lty=3)
  points(sampSize, sigMSE.SERF[I:J], col=’chocolate1’, type=’b’, lty=3)

  legend(350,maxMSE,c(expression(paste(sigma, " = 10")),
    expression(paste(sigma, " = 1.5")), expression(paste(sigma, " = 1")),
    expression(paste(sigma, " = 0.5")), expression(paste(sigma, " = 0.25"))),
    lty=c(1,5,2,4,3), col=c("black", "black", "black", "black", "black"),
    bg="white")

  dev.off()

  return
} # end plotMleSerfSigmasSigmaMSENoLgndFx()

plotMleSerfSigmasSigmaMSENoLgndFx(41,44,45,48,49,52,53,56,57,60,
  ’plot_AllSigmas_SigmaMSE_Mu3p5’,
  expression(paste("MSE of Estimators for ", sigma, ", ", mu, " = 3.5")))



### Finney results

plotNoSerfMusSigmaBiasFx <- function(A,B,C,D,E,F,plotName,yLabel)
{
  pdf(paste(plotName,"_NoSerf.pdf",sep=""),width=7,height=7)

  plot(sampSize, sigBias.MLE[A:B], col=’red’,
    xlim=c(0,525), ylim=c(minBias,maxBias),
    xlab = "Sample Size",
    ylab = yLabel,
    type=’b’, lty=1)
  abline(h=0)
  points(sampSize, sigBias.MLE[A:B], col=’red’, type=’b’, lty=1)
  points(sampSize, sigBias.MOM[A:B], col=’blue’, type=’b’, lty=1)
  points(sampSize, sigBias.JK[A:B], col=’green’, type=’b’, lty=1)

  points(sampSize, sigBias.MLE[C:D], col=’red’, type=’b’, lty=2)
  points(sampSize, sigBias.MOM[C:D], col=’blue’, type=’b’, lty=2)
  points(sampSize, sigBias.JK[C:D], col=’green’, type=’b’, lty=2)

  points(sampSize, sigBias.MLE[E:F], col=’red’, type=’b’, lty=3)
  points(sampSize, sigBias.MOM[E:F], col=’blue’, type=’b’, lty=3)
  points(sampSize, sigBias.JK[E:F], col=’green’, type=’b’, lty=3)

  legend(350,maxBias,c("MLE", "MOM", "Finney", expression(paste(mu, " = 2.5")),
    expression(paste(mu, " = 3")), expression(paste(mu, " = 3.5"))),
    lty=c(1,1,1,1,2,3), col=c("red", "blue", "green", "black", "black", "black"),
    bg="white")

  dev.off()

  return
} # end plotNoSerfMusSigmaBiasFx()

plotNoSerfMusSigmaBiasFx(17,20,37,40,57,60,’plot_AllMus_SigmaBias_SigmaP25’,
  expression(paste("Bias of Estimators for ", sigma, ", ", sigma, " = 0.25")))


plotNoSerfMusMuBiasNoLgndFx <- function(A,B,C,D,E,F,plotName,yLabel)
{

  pdf(paste(plotName,"_noLgnd_NoSerf.pdf",sep=""),width=7,height=7)

  plot(sampSize, muBias.MLE[A:B], col=’red’,
    xlim=c(0,525), ylim=c(minBias,maxBias),
    xlab = "Sample Size",
    ylab = yLabel,
    type=’b’, lty=1)
  abline(h=0)
  points(sampSize, muBias.MLE[A:B], col=’red’, type=’b’, lty=1)
  points(sampSize, muBias.MOM[A:B], col=’blue’, type=’b’, lty=1)
  points(sampSize, muBias.JK[A:B], col=’green’, type=’b’, lty=1)

  points(sampSize, muBias.MLE[C:D], col=’red’, type=’b’, lty=2)
  points(sampSize, muBias.MOM[C:D], col=’blue’, type=’b’, lty=2)
  points(sampSize, muBias.JK[C:D], col=’green’, type=’b’, lty=2)

  points(sampSize, muBias.MLE[E:F], col=’red’, type=’b’, lty=3)
  points(sampSize, muBias.MOM[E:F], col=’blue’, type=’b’, lty=3)
  points(sampSize, muBias.JK[E:F], col=’green’, type=’b’, lty=3)

  legend(350,maxBias,c(expression(paste(mu, " = 2.5")),
    expression(paste(mu, " = 3")), expression(paste(mu, " = 3.5"))), lty=c(1,2,3),
    col=c("black", "black", "black"), bg="white")

  dev.off()

  return
} # end plotNoSerfMusMuBiasNoLgndFx()

plotNoSerfMusMuBiasNoLgndFx(13,16,33,36,53,56,’plot_AllMus_MuBias_SigmaP5’,
  expression(paste("Bias of Estimators for ", mu, ", ", sigma, " = 0.5")))


plotNoSerfMusSigmaMSENoLgndFx <- function(A,B,C,D,E,F,plotName,yLabel)
{
  pdf(paste(plotName,"_noLgnd_NoSerf.pdf",sep=""),width=7,height=7)

  plot(sampSize, sigMSE.MLE[A:B], col=’red’,
    xlim=c(0,525), ylim=c(0,maxMSE),
    xlab = "Sample Size",
    ylab = yLabel,
    type=’b’, lty=1)
  abline(h=0)
  points(sampSize, sigMSE.MLE[A:B], col=’red’, type=’b’, lty=1)
  points(sampSize, sigMSE.MOM[A:B], col=’blue’, type=’b’, lty=1)
  points(sampSize, sigMSE.JK[A:B], col=’green’, type=’b’, lty=1)

  points(sampSize, sigMSE.MLE[C:D], col=’red’, type=’b’, lty=2)
  points(sampSize, sigMSE.MOM[C:D], col=’blue’, type=’b’, lty=2)
  points(sampSize, sigMSE.JK[C:D], col=’green’, type=’b’, lty=2)

  points(sampSize, sigMSE.MLE[E:F], col=’red’, type=’b’, lty=3)
  points(sampSize, sigMSE.MOM[E:F], col=’blue’, type=’b’, lty=3)
  points(sampSize, sigMSE.JK[E:F], col=’green’, type=’b’, lty=3)

  legend(350,maxBias,c(expression(paste(mu, " = 2.5")),
    expression(paste(mu, " = 3")), expression(paste(mu, " = 3.5"))),
    lty=c(1,2,3), col=c("black", "black", "black"), bg="white")

  dev.off()

  return
} # end plotNoSerfMusSigmaMSENoLgndFx()

plotNoSerfMusSigmaMSENoLgndFx(9,12,29,32,49,52,’plot_AllMus_SigmaMSE_Sigma1’,
  expression(paste("MSE of Estimators for ", sigma, ", ", sigma, " = 1")))


plotNoSerfMusMuMSENoLgndFx <- function(A,B,C,D,E,F,plotName,yLabel)
{

  pdf(paste(plotName,"_noLgnd_NoSerf.pdf",sep=""),width=7,height=7)

  plot(sampSize, muMSE.MLE[A:B], col=’red’,
    xlim=c(0,525), ylim=c(0,maxMSE),
    xlab = "Sample Size",
    ylab = yLabel,
    type=’b’, lty=1)
  abline(h=0)
  points(sampSize, muMSE.MLE[A:B], col=’red’, type=’b’, lty=1)
  points(sampSize, muMSE.MOM[A:B], col=’blue’, type=’b’, lty=1)
  points(sampSize, muMSE.JK[A:B], col=’green’, type=’b’, lty=1)

  points(sampSize, muMSE.MLE[C:D], col=’red’, type=’b’, lty=2)
  points(sampSize, muMSE.MOM[C:D], col=’blue’, type=’b’, lty=2)
  points(sampSize, muMSE.JK[C:D], col=’green’, type=’b’, lty=2)

  points(sampSize, muMSE.MLE[E:F], col=’red’, type=’b’, lty=3)
  points(sampSize, muMSE.MOM[E:F], col=’blue’, type=’b’, lty=3)
  points(sampSize, muMSE.JK[E:F], col=’green’, type=’b’, lty=3)

  legend(350,maxBias,c(expression(paste(mu, " = 2.5")),
    expression(paste(mu, " = 3")), expression(paste(mu, " = 3.5"))),
    lty=c(1,2,3), col=c("black", "black", "black"), bg="white")

  dev.off()

  return
} # end plotNoSerfMusMuMSENoLgndFx()

plotNoSerfMusMuMSENoLgndFx(5,8,25,28,45,48,’plot_AllMus_MuMSE_Sigma1p5’,
  expression(paste("MSE of Estimators for ", mu, ", ", sigma, " = 1.5")))
```

B.2 Density Plots

```
################################################################
###
###           File Name: densityPlots.R
###
### Program Description: plots the various density graphics used
###
################################################################


#########################
## Generic Density Plots:

pdf("densityPlots0.pdf",width=7,height=7)

#plot the densities: mu = 0
x<- seq(0,6,length=6000)
y10 <- dlnorm(x,0,10)
y1.5 <- dlnorm(x,0,1.5)
y1 <- dlnorm(x,0,1)
y.5 <- dlnorm(x,0,.5)
y.25 <- dlnorm(x,0,.25)
y.125 <- dlnorm(x,0,.125)
plot(x,y10,type=’l’,ylim=c(0,2),
     main=c(expression(paste(mu, " equals 0"))),ylab="")
points(x,y1.5,type=’l’,col="blue")
points(x,y1,type=’l’,col="green")
points(x,y.5,type=’l’,col="yellow")
points(x,y.25,type=’l’,col="red")
points(x,y.125,type=’l’,col="chocolate1")
legend(4,1.75,c(expression(paste(sigma, " = 10")),
  expression(paste(sigma, " = 3/2")),expression(paste(sigma, " = 1")),
  expression(paste(sigma, " = 1/2")),expression(paste(sigma, " = 1/4")),
  expression(paste(sigma, " = 1/8"))),lty=1,
  col=c("black","blue","green","yellow","red","chocolate1"), bg="white")

dev.off()


pdf("densityPlots1.pdf",width=7,height=7)

#plot the densities: mu = 1
x<- seq(0,6,length=6000)
y10 <- dlnorm(x,1,10)
y1.5 <- dlnorm(x,1,1.5)
y1 <- dlnorm(x,1,1)
y.5 <- dlnorm(x,1,.5)
y.25 <- dlnorm(x,1,.25)
y.125 <- dlnorm(x,1,.125)
plot(x,y10,type=’l’,ylim=c(0,2),
     main=c(expression(paste(mu, " equals 1"))),ylab="")
points(x,y1.5,type=’l’,col="blue")
points(x,y1,type=’l’,col="green")
points(x,y.5,type=’l’,col="yellow")
points(x,y.25,type=’l’,col="red")
points(x,y.125,type=’l’,col="chocolate1")

dev.off()


#######################################################################
## Comparing the Densities of the Normal and Lognormal when sigma < 0.3

pdf("density_normLognorm.pdf",width=7,height=7)

x<- seq(0,2,length=3000)
yLogn <- dlnorm(x,0,.25)
yNorm <- dnorm(x,1,.25)

plot(x,yLogn,type=’l’,col="blue",ylim=c(0,2),main="",ylab="")
points(x,yNorm,type=’l’, col="green")

legend(0.7,2,
  c(expression(paste("Lognormal Distribution; ", mu, " = 0, ", sigma, " = 1/4")),
  expression(paste("Normal Distribution; ", mu, " = 1, ", sigma, " = 1/4"))),
  lty=1,col=c("blue","green"))

dev.off()
```

C. APPLICATION CODE

C.1 Count the Sentence Lengths of a Given Document

```
/**************************************************************
**
**           File Name: countWords.c
**
** Program Description: counts and records the number of words
**                      in the sentences of a given document.
**
**************************************************************/

//standard libraries
#include <stdio.h>
#include <math.h>
#include <string.h>

//variables
int numWords = 1; // number of words in a sentence
int numSentences = 0; // number of sentences in the document
int oldSpace = 0;
int oldPeriod = 0;
int oldExclamation = 0;
int oldQuestion = 0;

int main(void)
{
  FILE *myFile;
  FILE *wordCountBOM;
  myFile = fopen("1830 BOM.txt","r"); //input file
  wordCountBOM = fopen("WC 1830 BOM.txt","w"); //output file

  fprintf(wordCountBOM, "words: \n");

  int n = 0;
  int limit = 5000000;

  // Count the number of words in a sentence and then spit them out:
  // this loop goes through each of the sentences until it reaches
  // the end of a document.
  for(n=0; n<limit; n++)
  //while((n = fgetc(myFile)) != EOF)
  //while(!feof(myFile));
    {

      char newChar;
      fscanf(myFile, "%c", &newChar);

      int space = (newChar - ’ ’);
```

```
   int period = (newChar - '.');
   int exclamation = (newChar - '!');
   int question = (newChar - '?');

   if((space==0) & (oldSpace!=0))
{
 numWords++;
}
   else if((period==0) & (oldPeriod!=0))
{
 fprintf(wordCountBOM, "%d \n", numWords);
 numWords = 1;
 numSentences++;
}
   else if((exclamation==0) & (oldExclamation!=0))
{
 fprintf(wordCountBOM, "%d \n", numWords);
 numWords = 1;
 numSentences++;
}
   else if((question==0) & (oldQuestion!=0))
{
 fprintf(wordCountBOM, "%d \n", numWords);
 numWords = 1;
 numSentences++;
}


   if(n >= (limit-5))
{
 printf("%c %d %d %d %d %d %d %d %d \n", newChar, space, oldSpace,
     period, oldPeriod, exclamation, oldExclamation,
     question, oldQuestion);

 printf("Number of sentences in document: %d \n", numSentences);
}

   oldSpace = space;
   oldPeriod = period;
   oldExclamation = exclamation;
   oldQuestion = question;
  }

 fclose(wordCountBOM);
 fclose(myFile);

}//end main
```

C.2 Find the Lognormal Parameters and Graph the Densities of Sentence Lengths for a Given Document

```
#############################################################
###
###           File Name: estimateWCparam10000.R
###
### Program Description: estimates the lognormal parameters
###                      and plots the densities of the given
###                      word-count data using the four
###                      estimation techniques.
###
#############################################################



## This function reads in the sentence-lengths and
## then computes the parameter estimates:
estimateWCparam <- function(data.WordCount, plotName, mainTitle)
  {
print("mainTitle")
print(mainTitle)

set.seed(19)

data.WC <- data.WordCount[,1]

n.WC <- length(data.WC)

#### GET ESTIMATES OF MU AND SIGMA^2: ####

## MLE:
  muMLE <- sum(log(data.WC))/n.WC
  sigma2MLE <- sum((log(data.WC)-muMLE)^2)/n.WC

## MOM:
  muMOM <- -log(sum(data.WC^2))/2 + 2*log(sum(data.WC)) - (3/2)*log(n.WC)
  sigma2MOM <- log(sum(data.WC^2)) - 2*log(sum(data.WC)) + log(n.WC)

## SERFLING:
  combos <- choose(n.WC,9)
  combos <- min(combos,10000)
  logX <- log(data.WC)

  intermMu <- rep(0,combos)
  intermSig <- rep(0,combos)

  combosMatrix <- matrix(sort(sample(1:n.WC,9,replace=FALSE)),nrow=1)
  comboIndex1 <- 2

# Build our combos matrix:
  while(comboIndex1 <= combos)
  {
newCombo <- sort(sample(1:n.WC,9,replace=FALSE))
used = FALSE

comboIndex2 <- 1
while(comboIndex2 < comboIndex1)
{
if((newCombo[1] == combosMatrix[comboIndex2,1]) &&
   (newCombo[2] == combosMatrix[comboIndex2,2]) &&
   (newCombo[3] == combosMatrix[comboIndex2,3]) &&
   (newCombo[4] == combosMatrix[comboIndex2,4]) &&
   (newCombo[5] == combosMatrix[comboIndex2,5]) &&
   (newCombo[6] == combosMatrix[comboIndex2,6]) &&
   (newCombo[7] == combosMatrix[comboIndex2,7]) &&
   (newCombo[8] == combosMatrix[comboIndex2,8]) &&
   (newCombo[9] == combosMatrix[comboIndex2,9]))
{
comboIndex2 = comboIndex1   # gets us out of the while-loop
comboIndex1 = comboIndex1-1 # causes us to find a new sample because
                            #   this one has already been used.
used = TRUE
}
comboIndex2 <- comboIndex2+1
}

if(used == FALSE)
{
combosMatrix <- rbind(combosMatrix, newCombo)
}

comboIndex1 <- comboIndex1+1

  }

  for(iSerf in 1:combos)
  {

aSerf = combosMatrix[iSerf,1]
bSerf = combosMatrix[iSerf,2]
cSerf = combosMatrix[iSerf,3]
dSerf = combosMatrix[iSerf,4]
eSerf = combosMatrix[iSerf,5]
fSerf = combosMatrix[iSerf,6]
gSerf = combosMatrix[iSerf,7]
hSerf = combosMatrix[iSerf,8]
jSerf = combosMatrix[iSerf,9]

meanAll9 <- (logX[aSerf] + logX[bSerf] + logX[cSerf] +
         logX[dSerf] + logX[eSerf] + logX[fSerf] +
             logX[gSerf] + logX[hSerf] + logX[jSerf])/9

    intermMu[iSerf] <- meanAll9

    intermSig[iSerf] <- ((logX[aSerf] - meanAll9)^2 + (logX[bSerf] - meanAll9)^2 +
      (logX[cSerf] - meanAll9)^2 + (logX[dSerf] - meanAll9)^2 +
      (logX[eSerf] - meanAll9)^2 + (logX[fSerf] - meanAll9)^2 +
      (logX[gSerf] - meanAll9)^2 + (logX[hSerf] - meanAll9)^2 +
      (logX[jSerf] - meanAll9)^2)/9
  }

  muSERF <- median(intermMu)
  sigma2SERF <- median(intermSig)


## FINNEY (JK):
  g <- function(t,n)
    {
      exp(t) * (1 - (t*(t+1)/n) + (t^2*(3*t^2 + 22*t + 21)/(6*n^2)))
    }

  zBar <- mean(log(data.WC))
  sSq <- sum((log(data.WC) - zBar)^2) / (n.WC-1)

  meanJK <- exp(zBar) * g(.5 * sSq,n.WC)
  varJK <- exp(2*zBar) * (g(2*sSq,n.WC) - g((n.WC-2) * sSq / (n.WC-1),n.WC))

  muJK <- 2*log(meanJK) - log(varJK + meanJK^2)/2
  sigma2JK <- log(varJK + meanJK^2) - 2*log(meanJK)
  if(sigma2JK == 0)
    {
sigma2JK <- 0.0001
    }


## Writing all the values to a .txt file:
  WCtable <- cbind(round(muMLE,3), round(sqrt(sigma2MLE),3),
                 round(muMOM,3), round(sqrt(sigma2MOM),3),
                 round(muSERF,3), round(sqrt(sigma2SERF),3),
                 round(muJK,3), round(sqrt(sigma2JK),3))
  write.table(WCtable,file=paste(plotName,"10000.txt",sep=""),sep="&",
                row.names=FALSE)


  } # end estimateWCparam()



## This function reads in the parameter estimates and
## then graphs the corresponding estimated densities:
WCparamDensityPlots <- function(data.WordCount, plotName, mainTitle)
  {
print("mainTitle")
print(mainTitle)

set.seed(19)

data.WC <- data.WordCount[,1]
```

```
n.WC <- length(data.WC)

estimates <- read.table(file=paste(plotName,"10000.txt",sep=""),sep="&",header=T)

muMLE <- estimates[1,1]
      sigmaMLE <- estimates[1,2]
muMOM <- estimates[1,3]
      sigmaMOM <- estimates[1,4]
muSERF <- estimates[1,5]
      sigmaSERF <- estimates[1,6]
muJK <- estimates[1,7]
      sigmaJK <- estimates[1,8]

x <- seq(0,125,length=4000)
yMLE <- dlnorm(x,muMLE,sigmaMLE)
yMOM <- dlnorm(x,muMOM,sigmaMOM)
ySERF <- dlnorm(x,muSERF,sigmaSERF)
yJK <- dlnorm(x,muJK,sigmaJK)


#### OVERALL SUMMARY PLOT: ####
pdf(paste(plotName,"_ALL10000.pdf",sep=""),width=7,height=7)
plot(density(data.WC), xlab="", ylab="Density of the Sentence Lengths",
  main=mainTitle,  xlim=c(0,100),  ylim=c(0,.03))
points(x,yMLE,type=’l’,col="red",lty=1)
points(x,yMOM,type=’l’,col="blue",lty=1)
points(x,ySERF,type=’l’,col="chocolate1",lty=1)
points(x,yJK,type=’l’,col="green",lty=2)
legend(65,0.025,c("MLE","MOM","Serfling","Finney"),lty=c(1,1,1,1),
                 col=c("red","blue","chocolate1","green"))
dev.off()

#### OVERALL SUMMARY PLOT: (no legend) ####
pdf(paste(plotName,"_ALL_noLgnd10000.pdf",sep=""),width=7,height=7)
plot(density(data.WC), xlab="", ylab="Density of the Sentence Lengths",
  main=mainTitle,  xlim=c(0,100),  ylim=c(0,.03))
points(x,yMLE,type=’l’,col="red",lty=1)
points(x,yMOM,type=’l’,col="blue",lty=1)
points(x,ySERF,type=’l’,col="chocolate1",lty=1)
points(x,yJK,type=’l’,col="green",lty=2)
dev.off()

#### SUMMARY PLOT - MLE: ####
pdf(paste(plotName,"_MLE.pdf",sep=""),width=7,height=7)
plot(density(data.WC), xlab="", ylab="Density of the Sentence Lengths",
  main=mainTitle,  xlim=c(0,100),  ylim=c(0,.03))
points(x,yMLE,type=’l’,col="red",lty=1)
legend(125,0.025,c("MLE","MOM","Serfling","Finney"),lty=c(1,1,1,1),
                 col=c("red","blue","chocolate1","green"))
dev.off()

#### SUMMARY PLOT - MLE: (no legend) ####
pdf(paste(plotName,"_MLE_noLgnd.pdf",sep=""),width=7,height=7)
plot(density(data.WC), xlab="", ylab="Density of the Sentence Lengths",
  main=mainTitle,  xlim=c(0,100),  ylim=c(0,.03))
points(x,yMLE,type='l',col="red",lty=1)
dev.off()

#### SUMMARY PLOT - Serfling: ####
pdf(paste(plotName,"_SERF10000.pdf",sep=""),width=7,height=7)
plot(density(data.WC), xlab="", ylab="Density of the Sentence Lengths",
  main=mainTitle,  xlim=c(0,100),  ylim=c(0,.03))
points(x,ySERF,type='l',col="chocolate1",lty=1)
legend(125,0.025,c("MLE","MOM","Serfling","Finney"),lty=c(1,1,1,1),
                 col=c("red","blue","chocolate1","green"))
dev.off()

#### SUMMARY PLOT - Serfling: (no legend) ####
pdf(paste(plotName,"_SERF_noLgnd10000.pdf",sep=""),width=7,height=7)
plot(density(data.WC), xlab="", ylab="Density of the Sentence Lengths",
  main=mainTitle,  xlim=c(0,100),  ylim=c(0,.03))
points(x,ySERF,type='l',col="chocolate1",lty=1)
dev.off()

#### BOX PLOT ####
pdf(paste(plotName,"_BoxPlot.pdf",sep=""),width=7,height=7)
boxplot(data.WC, xlab="", ylab="Sentence Lengths", main=mainTitle)
dev.off()

  } # end WCparamDensityPlots()



## Getting the Parameter Estimates:

data.WordCount <- read.table("WC 1830 BOM.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_1830BOM',
     'Book of Mormon Text')
data.WordCount <- read.table("WC all12Nephi.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_all12Nephi',
     'First and Second Nephi Texts')
data.WordCount <- read.table("WC all34Nephi.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_all34Nephi',
     'Third and Fourth Nephi Texts')
data.WordCount <- read.table("WC allAlma.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allAlma',
     'Alma Text')
data.WordCount <- read.table("WC allEnos.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allEnos',
     'Enos Text')
data.WordCount <- read.table("WC allEther.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allEther',
     'Ether Text')
data.WordCount <- read.table("WC allFirstNephi.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allFirstNephi',
     'First Nephi Text')
data.WordCount <- read.table("WC allFourthNephi.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allFourthNephi',
     'Fourth Nephi Text')
data.WordCount <- read.table("WC allHelaman.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allHelaman',
     'Helaman Text')
data.WordCount <- read.table("WC allJacob.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allJacob',
     'Jacob Text')
data.WordCount <- read.table("WC allJarom.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allJarom',
     'Jarom Text')
data.WordCount <- read.table("WC allMormon.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allMormon',
     'Mormon Text')
data.WordCount <- read.table("WC allMoroni.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allMoroni',
     'Moroni Text')
data.WordCount <- read.table("WC allMosiah.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allMosiah',
     'Mosiah Text')
data.WordCount <- read.table("WC allOmni.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allOmni',
     'Omni Text')
data.WordCount <- read.table("WC allSecondNephi.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allSecondNephi',
     'Second Nephi Text')
data.WordCount <- read.table("WC allThirdNephi.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allThirdNephi',
     'Third Nephi Text')
data.WordCount <- read.table("WC allWoM.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allWoM',
     'Words of Mormon Text')
data.WordCount <- read.table("WC allMormonWoM.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allMormonWoM',
     'Book of Mormon and Words of Mormon Texts')

data.WordCount <- read.table("WC allIsaiah.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allIsaiah',
     'Isaiah Text')
data.WordCount <- read.table("WC allPratt.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allPratt',
     'Parley P. Pratt Documents')
data.WordCount <- read.table("WC allCowdery.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allCowdery',
     'Oliver Cowdery Documents')
data.WordCount <- read.table("WC allRigdonNOCORRECTIONS.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allRigdonNOCORRECTIONS',
     'Sidney Rigdon Letters')
data.WordCount <- read.table("WC allRigdonSections.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allRigdonSections',
     'Sidney Rigdon Revelations')
data.WordCount <- read.table("WC allHamMad.txt", header=T)
  estimateWCparam(data.WordCount, 'WCdensPlots_allHamMad',
     'Hamilton/Madison Federalist Papers')
```

```
data.WordCount <- read.table("WC allHamilton.txt", header=T)
  estimateWCparam(data.WordCount, ’WCdensPlots_allHamilton’,
     ’Hamilton Federalist Papers’)
data.WordCount <- read.table("WC allHamiltonQ1.txt", header=T)
  estimateWCparam(data.WordCount, ’WCdensPlots_allHamiltonQ1’,
     ’Hamilton Federalist Papers, First Quarter’)
data.WordCount <- read.table("WC allHamiltonQ2.txt", header=T)
  estimateWCparam(data.WordCount, ’WCdensPlots_allHamiltonQ2’,
     ’Hamilton Federalist Papers, Second Quarter’)
data.WordCount <- read.table("WC allHamiltonQ3.txt", header=T)
  estimateWCparam(data.WordCount, ’WCdensPlots_allHamiltonQ3’,
     ’Hamilton Federalist Papers, Third Quarter’)
data.WordCount <- read.table("WC allHamiltonQ4.txt", header=T)
  estimateWCparam(data.WordCount, ’WCdensPlots_allHamiltonQ4’,
     ’Hamilton Federalist Papers, Fourth Quarter’)
data.WordCount <- read.table("WC allMadison.txt", header=T)
  estimateWCparam(data.WordCount, ’WCdensPlots_allMadison’,
     ’Madison Federalist Papers’)
data.WordCount <- read.table("WC allJay.txt", header=T)
  estimateWCparam(data.WordCount, ’WCdensPlots_allJay’,
     ’Jay Federalist Papers’)


## Plotting the Estimated Densities:

data.WordCount <- read.table("WC 1830 BOM.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_1830BOM’,
     ’Book of Mormon Text’)
data.WordCount <- read.table("WC all12Nephi.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_all12Nephi’,
     ’First and Second Nephi Texts’)
data.WordCount <- read.table("WC all34Nephi.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_all34Nephi’,
     ’Third and Fourth Nephi Texts’)
data.WordCount <- read.table("WC allAlma.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allAlma’,
     ’Alma Text’)
data.WordCount <- read.table("WC allEnos.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allEnos’,
     ’Enos Text’)
data.WordCount <- read.table("WC allEther.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allEther’,
     ’Ether Text’)
data.WordCount <- read.table("WC allFirstNephi.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allFirstNephi’,
     ’First Nephi Text’)
data.WordCount <- read.table("WC allFourthNephi.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allFourthNephi’,
     ’Fourth Nephi Text’)
data.WordCount <- read.table("WC allHelaman.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allHelaman’,
     ’Helaman Text’)
data.WordCount <- read.table("WC allJacob.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allJacob’,
     ’Jacob Text’)
data.WordCount <- read.table("WC allJarom.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allJarom’,
     ’Jarom Text’)
data.WordCount <- read.table("WC allMormon.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allMormon’,
     ’Mormon Text’)
data.WordCount <- read.table("WC allMoroni.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allMoroni’,
     ’Moroni Text’)
data.WordCount <- read.table("WC allMosiah.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allMosiah’,
     ’Mosiah Text’)
data.WordCount <- read.table("WC allOmni.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allOmni’,
     ’Omni Text’)
data.WordCount <- read.table("WC allSecondNephi.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allSecondNephi’,
     ’Second Nephi Text’)
data.WordCount <- read.table("WC allThirdNephi.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allThirdNephi’,
     ’Third Nephi Text’)
data.WordCount <- read.table("WC allWoM.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allWoM’,
     ’Words of Mormon Text’)
data.WordCount <- read.table("WC allMormonWoM.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allMormonWoM’,
     ’Book of Mormon and Words of Mormon Texts’)

data.WordCount <- read.table("WC allIsaiah.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allIsaiah’,
     ’Isaiah Text’)
data.WordCount <- read.table("WC allPratt.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allPratt’,
     ’Parley P. Pratt Documents’)
data.WordCount <- read.table("WC allCowdery.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allCowdery’,
     ’Oliver Cowdery Documents’)
data.WordCount <- read.table("WC allRigdonNOCORRECTIONS.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allRigdonNOCORRECTIONS’,
     ’Sidney Rigdon Letters’)
data.WordCount <- read.table("WC allRigdonSections.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allRigdonSections’,
     ’Sidney Rigdon Revelations’)
data.WordCount <- read.table("WC allHamMad.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allHamMad’,
     ’Hamilton/Madison Federalist Papers’)

data.WordCount <- read.table("WC allHamilton.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allHamilton’,
     ’Hamilton Federalist Papers’)
data.WordCount <- read.table("WC allHamiltonQ1.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allHamiltonQ1’,
     ’Hamilton Federalist Papers, First Quarter’)
```

```
data.WordCount <- read.table("WC allHamiltonQ2.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allHamiltonQ2’,
     ’Hamilton Federalist Papers, Second Quarter’)
data.WordCount <- read.table("WC allHamiltonQ3.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allHamiltonQ3’,
     ’Hamilton Federalist Papers, Third Quarter’)
data.WordCount <- read.table("WC allHamiltonQ4.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allHamiltonQ4’,
     ’Hamilton Federalist Papers, Fourth Quarter’)
data.WordCount <- read.table("WC allMadison.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allMadison’,
     ’Madison Federalist Papers’)
data.WordCount <- read.table("WC allJay.txt", header=T)
  WCparamDensityPlots(data.WordCount, ’WCdensPlots_allJay’,
     ’Jay Federalist Papers’)



####################
### Other Plots Used

## Federalist Papers

data.Hamilton <- read.table(’WCdensPlots_allHamilton10000.txt’,
   header=TRUE, sep=’&’)
data.Madison <- read.table(’WCdensPlots_allMadison10000.txt’,
   header=TRUE, sep=’&’)
data.Jay <- read.table(’WCdensPlots_allJay10000.txt’,
   header=TRUE, sep=’&’)

mu.Hamilton <- data.Hamilton[1,1]
mu.Madison <- data.Madison[1,5]
mu.Jay <- data.Jay[1,1]

sigma.Hamilton <- data.Hamilton[1,2]
sigma.Madison <- data.Madison[1,6]
sigma.Jay <- data.Jay[1,2]

pdf("WC_FederalistPapers.pdf",width=7,height=7)
x <- seq(0,125,length=4000)
yHam <- dlnorm(x,mu.Hamilton,sigma.Hamilton)
yMad <- dlnorm(x,mu.Madison,sigma.Madison)
yJay <- dlnorm(x,mu.Jay,sigma.Jay)
plot(x,yHam, xlab="", ylab="Density of the Sentence Lengths",
  main=’Federalist Papers’,
  xlim=c(0,100),ylim=c(0,.03),type=’l’,lty=1,col="red")
points(x,yMad,type=’l’,lty=1,col="blue")
points(x,yJay,type=’l’,lty=1,col="green")
legend(65,0.025,c("Hamilton","Madison","Jay"),lty=c(1,1,1),
  col=c("red","blue","green"))
dev.off()


## Hamilton Federalist Papers in 4 groups
```

```
data.HamiltonQ1 <- read.table(’WCdensPlots_allHamiltonQ110000.txt’,
   header=TRUE, sep=’&’)
data.HamiltonQ2 <- read.table(’WCdensPlots_allHamiltonQ210000.txt’,
   header=TRUE, sep=’&’)
data.HamiltonQ3 <- read.table(’WCdensPlots_allHamiltonQ310000.txt’,
   header=TRUE, sep=’&’)
data.HamiltonQ4 <- read.table(’WCdensPlots_allHamiltonQ410000.txt’,
   header=TRUE, sep=’&’)

mu.Q1 <- data.HamiltonQ1[1,1]
mu.Q2 <- data.HamiltonQ2[1,1]
mu.Q3 <- data.HamiltonQ3[1,1]
mu.Q4 <- data.HamiltonQ4[1,1]

sigma.Q1 <- data.HamiltonQ1[1,2]
sigma.Q2 <- data.HamiltonQ2[1,2]
sigma.Q3 <- data.HamiltonQ3[1,2]
sigma.Q4 <- data.HamiltonQ4[1,2]

pdf("WC_HamiltonQuarters.pdf",width=7,height=7)
x <- seq(0,125,length=4000)
yQ1 <- dlnorm(x,mu.Q1,sigma.Q1)
yQ2 <- dlnorm(x,mu.Q2,sigma.Q2)
yQ3 <- dlnorm(x,mu.Q3,sigma.Q3)
yQ4 <- dlnorm(x,mu.Q4,sigma.Q4)
plot(x,yQ1, xlab="", ylab="Density of the Sentence Lengths",
  main=’Hamilton Federalist Papers, All Four Quarters’,
  xlim=c(0,100),ylim=c(0,.03),type=’l’,lty=1,col="red")
points(x,yQ2,type=’l’,lty=1,col="blue")
points(x,yQ3,type=’l’,lty=1,col="chocolate1")
points(x,yQ4,type=’l’,lty=1,col="green")
legend(55,0.025,c("First Quarter","Second Quarter","Third Quarter","Fourth Quarter"),
  lty=c(1,1,1,1),col=c("red","blue","chocolate1","green"))
dev.off()


## BOM and Modern Authors

data.BOM <- read.table(’WCdensPlots_1830BOM10000.txt’,
   header=TRUE, sep=’&’)
data.RigdonLetters <- read.table(’WCdensPlots_allRigdonNOCORRECTIONS10000.txt’,
   header=TRUE, sep=’&’)
data.RigdonRev <- read.table(’WCdensPlots_allRigdonSections10000.txt’,
   header=TRUE, sep=’&’)

mu.BOM <- data.BOM[1,5]
mu.RigdonLetters <- data.RigdonLetters[1,1]
mu.RigdonRev <- data.RigdonRev[1,1]

sigma.BOM <- data.BOM[1,6]
sigma.RigdonLetters <- data.RigdonLetters[1,2]
sigma.RigdonRev <- data.RigdonRev[1,2]
```

```
pdf("WC_BOMmodernAuthor.pdf",width=7,height=7)
x <- seq(0,125,length=4000)
yBOM <- dlnorm(x,mu.BOM,sigma.BOM)
yRL <- dlnorm(x,mu.RigdonLetters,sigma.RigdonLetters)
yRR <- dlnorm(x,mu.RigdonRev,sigma.RigdonRev)
plot(x,yBOM, xlab="", ylab="Density of the Sentence Lengths",
  main='The Book of Mormon Compared with a Modern Author',
  xlim=c(0,100),ylim=c(0,.03),type='l',lty=1,col="red")
points(x,yRL,type='l',lty=1,col="blue")
points(x,yRR,type='l',lty=1,col="green")
legend(40,0.025,c("Book of Mormon Text","Sidney Rigdon Letters",
  "Sidney Rigdon Revelations"),lty=c(1,1,1),col=c("red","blue","green"))
dev.off()


## 1+2 Nephi, Alma texts

data.12Ne <- read.table('WCdensPlots_all12Nephi10000.txt',
   header=TRUE, sep='&')
data.Alma <- read.table('WCdensPlots_allAlma10000.txt',
   header=TRUE, sep='&')

mu.12Ne <- data.12Ne[1,5]
mu.Alma <- data.Alma[1,1]

sigma.12Ne <- data.12Ne[1,6]
sigma.Alma <- data.Alma[1,2]

pdf("WC_12Ne_Alma.pdf",width=7,height=7)
x <- seq(0,125,length=4000)
y12Ne <- dlnorm(x,mu.12Ne,sigma.12Ne)
yAlma <- dlnorm(x,mu.Alma,sigma.Alma)
plot(x,y12Ne, xlab="", ylab="Density of the Sentence Lengths",
  main='First and Second Nephi Texts Compared with Alma Text',
  xlim=c(0,100),ylim=c(0,.03),type='l',lty=1,col="red")
points(x,yAlma,type='l',lty=1,col="blue")
legend(35,0.025,c("First and Second Nephi Texts","Alma Text"),
  lty=c(1,1),col=c("red","blue"))
dev.off()


## BOM+WoM, Moroni texts

data.BOMwom <- read.table('WCdensPlots_allMormonWoM10000.txt',
   header=TRUE, sep='&')
data.Moroni <- read.table('WCdensPlots_allMoroni10000.txt',
   header=TRUE, sep='&')

mu.BOMwom <- data.BOMwom[1,5]
mu.Moroni <- data.Moroni[1,1]

sigma.BOMwom <- data.BOMwom[1,6]
sigma.Moroni <- data.Moroni[1,2]
```

```
pdf("WC_BOMwom_Moroni.pdf",width=7,height=7)
x <- seq(0,125,length=4000)
yBOMwom <- dlnorm(x,mu.BOMwom,sigma.BOMwom)
yMoroni <- dlnorm(x,mu.Moroni,sigma.Moroni)
plot(x,yBOMwom, xlab="", ylab="Density of the Sentence Lengths",
  main='', xlim=c(0,100),ylim=c(0,.03),type='l',lty=1,col="red")
points(x,yMoroni,type='l',lty=1,col="blue")
legend(55,0.025,c("Mormon Texts","Moroni Text"),
  lty=c(1,1),col=c("red","blue"))
title("Book of Mormon and Words of Mormon Texts Compared with Moroni Text",
  cex.main=1.05,font.main=2)
dev.off()
```