BACHELOR'S DEGREE THESIS

Degree in Statistics

Title: Statistical significance of ratios on a biological environment

Author: Cristina Paré Estalella

Advisor: Sergi Civit and Mireia Vilardell

Department: Department of Genetics, Microbiology and Statistics, section of Statistics (University of Barcelona)

Academic year: 2019 - 2020

UNIVERSITAT DE BARCELONA

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat de Matemàtiques i Estadística

Abstract

Polymerase Chain Reaction (PCR) and its variants such as the qPCR, among many others, are molecular biology techniques highly used in biomedical research to compare the relative expression (RE) of a group against a control group. Even it's freqüent use, the statistical analysis of the data is still a reason for disagreement between labs and research groups. This is due to the ratiometric data that, in most cases, assuming an underlying probability distribution can be difficult to objectify due to the data's characteristics and the small sample size. In this project, we will present two statistical methodologies for the analysis of the RE obtained in qPCR, and they can be applied to other techniques that have as an output ratiometric data. These techniques are based on Bootstrap and Bayesian methodologies and we will perform a simulation study to identify how they perform under different circumstances and which one has a better result. We will arrive to the conclusion that, taking into account performance and computational cost, it's better to use the Bayesian methodology exclusively for small size samples where the performance difference between the two methodologies is notable, and opt for Bootstrap technique when working with medium and large size samples.

Key words

qPCR, Bootstrap, Bayesian, simulation

Abstract: Catalan Version

La reacció en cadena de la polimerasa i les seves variants com el qPCR, entre moltes altres, són tècniques de la biologia molecular molt utilitzades en la recerca biomèdica per a comparar l'expressió relativa (RE) d'un grup respecte al grup de control. Tot i l'alta freqüència d'ús d'aquestes tècniques, l'anàlisi estadístic de les dades és encara un motiu de desacord entre laboratoris i grups de recerca. Això es deu a la naturalesa de les dades que són ratis i, juntament amb la grandària mostral petita, provoca que en molts casos sigui difícil d'assumir que hi ha una distribució de probabilitat subjacent. En aquest projecte es presentaran dos mètodes estadístics per a l'anàlisi dels RE obtinguts del qPCR i que també poden ser utilitzats per a l'anàlisi de dades d'altres tècniques amb *outputs* similars. Aquests mètodes es basaran en la tècnica Bootstrap i la metodologia Bayesiana. Per últim, es durà a terme un estudi de simulació per identificar com rendeixen les tècniques sota diferents circumstàncies i avaluar amb quina s'obté un millor resultat. Arribarem a la conclusió que, tenint en compte el rendiment i el cost computacional, la millor opció és reservar la metodologia Bayesiana pels casos en els quals tenim una grandària mostral petita, ja que en aquest cas el rendiment de les dues metodologies difereix molt, i optar per la tècnica Bootstrap quan es treballa amb grandàries mostrals mitjanes o grans.

Paraules clau

qPCR, Bootstrap, Bayesià, simulació

Mathematics Subject Classification (MSC)

62F40 Bootstrap, jackknife and other resampling methods.

62F15 Bayesian inference.

92D20 Protein sequences, DNA sequences.

Acknowledgment

I would like to express my special thanks to Sergi Civit and Mireia Vilardell for their continued support, encouragement, and guidance on the project and, most importantly, their confidence and trust in me to carry through this assignment. They have helped me a lot to put these ideas into something worth it, challenging me when I needed it the most.

Besides that, I would also like to thank all the teachers I've met during the last 4 years for making me love statistics and for teaching me everything that I've learnt.

My completion of this project would not have been accomplished without the moral support of Clara, my best friend and faithful university fellow. Finally, I would like to thank my family: mum, for being so patient and always having good words; dad, for teaching me that hard work is the only path, to my brother for always helping me when I needed it, and Ignasi for making me realize that I'm capable of doing what I propose.

Contents

List of Figures

List of Tables

1 State of the art

1.1 Summary

During the last decade, a lot of new technologies have been developed allowing the quantification of biological material such as ADN, ARN or proteins, as many others. These techniques are named under the terminology of Omic Techniques. Even though they have become a revolution in biology and medicine, the results obtained with these techniques are not conclusive and need the validation of other procedures.

The advance on fast validation and reliable Omic Techniques such as the real-time polymerase chain reaction (qPCR), comes usually hand to hand with a change in the approach from the total quantification of biological material to the relative quantification. In other cases, it is necessary to complement the large-scale analysis with functional studies that allow to clarify the gain or loss of some variants associated with a specific phenotype. That way, the need for performing data analysis represented as quotients has increased during the last years. These quotients are known as RE, the acronym for relative expression. Even though there are a lot of cases where the classical methodology is being used, it is not always appropriate.

Boots-Ratio is an optimized and widely used software for the analysis of qPCR data combined with retrotranscription (RT-qPCR), but it has some limitations as for example, it is not that efficient for the infra expression as it is for the overexpression on the genes.

1.2 Current state of the topic and project motivation

With the improvement of Omic Technologies, the validation of functional studies has become more relevant, which means that a large number of genes of interest (GOI) must be re-analyzed using other methodologies. When it comes to genetic expression studies, the microarrays or RNA-seq tend to be validated with the qPCR method, acquiring relative expression of the GOI versus the reference gene. When other techniques such as Northern, Southern or Western blot are used, the final result is also a relation between two conditions. The same happens when functional studies based on reporter genes are performed – the result is a relation between two activities usually from luminescence.

Recently, Clèries et al has developed the application of BootstRatio that allows to evaluate the optimized RE coefficients for RT-qPCR on a website without having to assume any underlying probability distribution for the data. It is based on generating N "subsamples" of the RE with resampling methods, quantifying the proportion of subsamples in which the mean of the RE is higher than 1 or the mean or median of the RE in the reference sample. Since its publication, the use of BootstRatio has been increasing due to the necessity of an app that allows to analysis RE data and quotients. Nevertheless, it has limitations due to RE close to 0 and a small sample size, which was already pointed on the simulation study by the authors. [5]

This project will be based on the development of a methodology with the objective of facilitating the analysis of quotients, specifically for RT-qPCR results and the reporter gene. It will also contain the respective simulation study, to determine in which scenarios is better to use the different alternatives we will offer.

1.3 Quantification of the relative expression with the RT-qPCR

The retro-transcription quantitative polymerase chain reaction (RT-qPCR) is a technique widely used on the research and diagnosis as a method to detect, characterize, and quantify the nucleic acid.

It is a procedure easy to use and interpret the results as it allows scientists to take a very small sample of DNA and exponentially amplify it to a large enough amount to study in detail. It has become a fundamental genetic testing used in medical laboratory and clinical laboratory research for a large variety of applications. This technique presents other advantages such as the reproducibility of the results and the need for a lower amount of samples compared with other methods.

This technique is based on the conventional PCR which allows a targeted DNA sequence to be copied through a highly sensitive thermochemical cicle, repeated between 20 and 40 times, with a specific combination of reactors. This reaction has the potential to amplify the DNA molecule to become over 1 billion molecules in less than 2 hours. As we can see on Figure 1, the thermochemical cicle has 3 steps: denaturing, annealing and extension. During the first step, upon high temperatures (around 95ºC), the DNA template molecule will denaturate, meaning that the hydrogen bonds that link the double helix will break, pulling apart the two strands. After that, annealing starts using low temperatures (also called annealing temperatures around 55ºC - 65ºC) where the complementary sequences of single-stranded DNA will pair by hydrogen bonds with the DNA template. At optimal temperature (around 72ºC), extension occurs and DNA polymerase synthesizes in DNA.

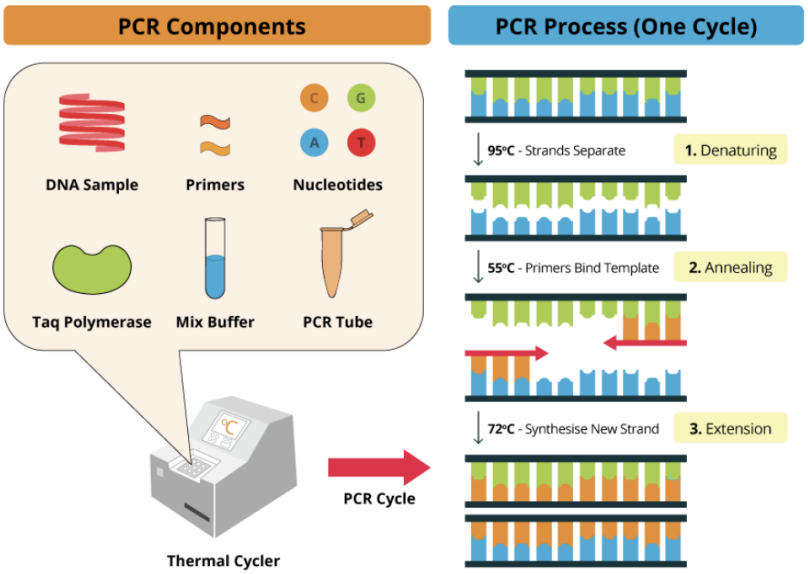

Figure 1: PCR termochemical cicle steps

The RT-qPCR, as an improvement to the basic technique, allows to use the RNA as a mold with an additional step to transcribe the complementary DNA (cDNA) using the retro-transcription. Another upgrade of the RT-qPCR is that there is a wider range of methods and chemical products available to quantify the DNA molecules as it is based on a dye, usually green, that reacts fluorescently to the double chain DNA union. At the end of every cycle, the thermocycler (the machine that carries out the process) measures the fluorescence in the sample to infer the amplicons concentrations. That' why it is called "real-time". The cycle where a certain DNA quantity is exceed is called C_t. [9]

The results are based on the relative expression (RE) that is a quotient of a target gene versus a reference gene or with an absolute quantification based on the internal or external calibration curves, it is computed as follows:

$$RE = \frac{(E_{objetivo})^{\Delta Ct_{objetivo}}}{(E_{referencia})^{\Delta Ct_{referencia}}}$$

or

$$RE = 2^{-\Delta\Delta Ct}$$

where C_t is the cycle in where the threshold is reached, ΔCt is the difference between C_t of the control and the treatment group and $\Delta\Delta Ct = \Delta C_{t_{referencia}} - \Delta C_{t_{objetivo}}$.

The RT-qPCR is widely used by researchers as it avoids the complications of having to generate calibration material and it is measured as a relation between the expression of the target gene versus the expression of the reference gene instead of having to standardize the expression values. The standardization is done during the pre-processing of the data and it ensures its independence and comparability.

1.4 Quantification of the function or relative expression through reporter gene tests

The gene reporter tests have also become one of the most important techniques of the molecular biology of the last decades. It allows us to evaluate the functionality or activity of the genes or some of the gene variants that have been found associated with determinate phenotypes, often using genotyping techniques on a large scale. In reporter gene assays, the activity of a reporter gene is measured. A reporter gene is joined to a target regulatory DNA sequence in an expression vector, which is then transfected into the cell type of choice. The reporter gene is transcribed and translated in the cells and its activity is measured to access the strength or function of the target regulatory DNA sequence or study effects of transcription factors or potential drugs etc.

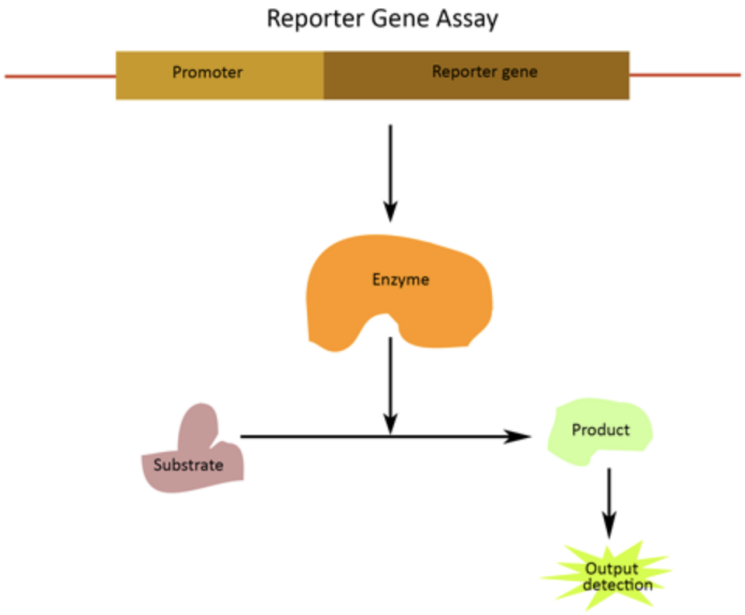

Figure 2: Reporter gene assay method

Even though the gene reporter test includes a large variety of protocols, the resultant data has similar statistical properties. Mostly the result is two paired expression measures of two internal genes. Frequently, the RE quotient of these genes is standardized with an external control gene that results in a final ratio of ratios.

1.5 Current problematic

A variety of methods have been developed for the analysis of RT-qPCR standardized data and for the reporter gene. In the majority of them, determine the statistical signification for the RE through the statistical modeling implies assuming an underlying probability distribution.

When geneticist biologists are working with this type of data, they want to know the probability of this Relative Expression being greater than 1 to have enough statistical evidence to ensure that both populations are different.

$$P(RE > 1)$$

Obtaining significance levels for these ratios through statistical modeling implies assuming an underlying probability distribution of the data obtained that, most of the time, may be difficult to evaluate, especially when the data has a small sample size on target and reference samples.

Also, the ratiometric nature of the data produced on some scientific studies requires careful statistical treatment, which is often lacking in the literature. It is known that these types of ratios are inherently sensitive to the propagation of error that each of the quotients is exposed to, resulting in a bigger error that can affect the quotients individually. It is important to avoid that this propagation of error derives to erroneous conclusions, especially in order to avoid false-positive and contribute to the reproducibility of the test.

Even though this type of assays is very popular among the scientific community, the reproducibility and reliability of the data depend on how the experiments are performed and interpreted as there is a considerable variation in the protocols. Different laboratories apply distinct data analysis techniques which lead to a lack of consistency on proper quality control steps throughout the assay. There is a need for a standardized process to analyze ratiometric data that leads to consistent and reproducible results, independently of the sample size.

1.6 Ratio estimator

As defined in mathematics, a ratio indicates how many times a number contains another. Ratios are commonly used to make comparisons between two things or two quantities in units of the same dimension, which means that ratios are unitless. That's the reason why ratios are widely used not only in mathematic and scientific fields but also on the quotidian scope.

The ratio estimator is a statistical parameter and it is defined as the division of the means of two random variables. This is a biased estimator, as shown below with Jensen's inequality assuming independence between x and y, and its distribution is asymmetrical.

$$E\left(\frac{y}{x}\right) = E\left(y\left(\frac{1}{x}\right)\right) = E(y)E\left(\frac{1}{x}\right) \geq \frac{E(y)}{E(x)} = E(y)\frac{1}{E(x)}$$

This properties, not only lead to a problem with symmetrical statistical tests such as the t-test, but it also affects on confidence intervals estimations created with variance and symmetrical distributions

as it tends to overestimate the lower side of the interval, at the same time it underestimates the upper side of the interval.

It is clear that working with ratios estimators is not an easy task and it requires corrections on the bias that are not easy to use for all individuals working with this type of data.

1.7 Data Presentation

To better understand the data problem and the solutions we propose, we will be working with an example of real data. This data incorporates the Relative Expression (RE) of 3 different genes. Each gene has 19 observations, which will allow us to work with a small-size sample before we get into the simulation study.

First of all, we will do a numeric analysis including the mean, standard deviation, median, minimum, maximum, range, skewness and kurtosis. All along with the project, we will be using these measures at some point.

Table 1: Descriptive analysis of the sample data

	n	mean	sd	median	min	max	range	skew	kurtosis
12s/MT-RNR1	19	0.905	0.371	0.841	0.261	1.705	1.444	0.260	-0.690
MT-ATP6	19	0.914	0.432	0.948	0.101	2.033	1.931	0.435	0.355
MT-CO2/COX2	19	0.900	0.485	0.932	0.060	2.230	2.170	0.721	0.945

Also, we will include a boxplot and a density plot to have a visual representation of the genes in our database.

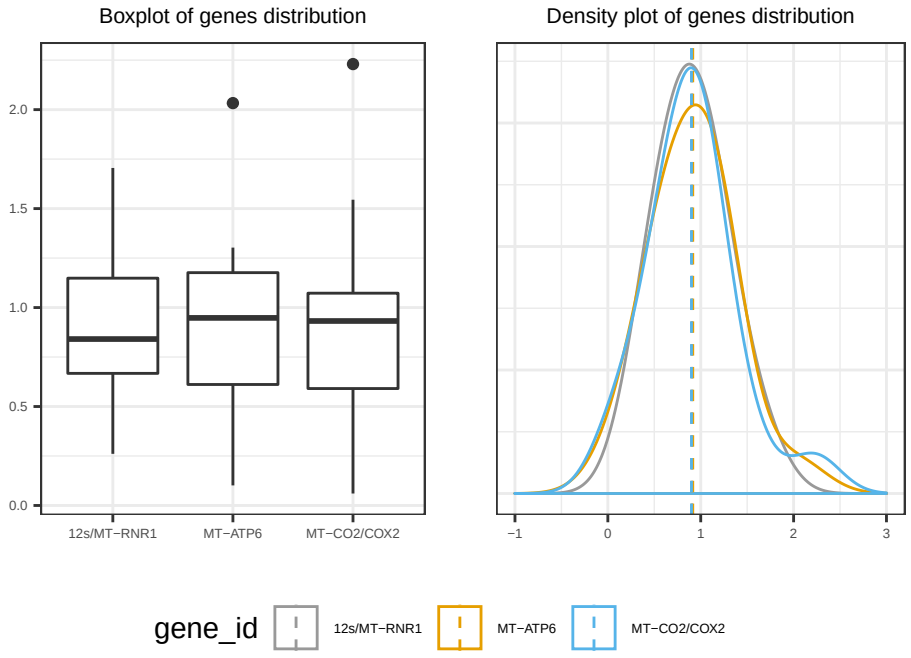

Figure 3: Boxplot and density plot for the sample data

If we take a look at the analysis, the three gene samples are distributed around the value 1 approximately. What the geneticist wants to know in these situations is the probability of these relative expression being greater than 1 to be able to determine from a statistical point of view, if there is enough evidence to ensure that the two populations being compared are different.

1.8 Statistical approach

To approach this problem from a statistical point of view, we will be calculating $P(R > 1)$ as $P(\overline{R} > 1)$ where $\overline{R}$ is the mean of the ratios. The reasoning behind this choice it's the attractiveness of the mean's properties. As we know, the mean is the geometrical or gravitational center of a data set as it equilibrates the deficient and extra values. This looks very tempting because, as seen in the previous section, we're working with nearly-symmetrical (slightly right-skewed) distributions where we don't typically come across outliers. We also want the methodology to be applied to any sample size so, using the mean allows us to avoid the median's bias when calculating it on a small sample size.

Another appealing characteristic from the mean statistic is its distribution, which is always more symmetric than the original variable's distribution. Having the general scenario where x is a random variable with mean μ and variance σ^2, then:

$$E[\overline{x}] = E[\tfrac{1}{n}\sum x_i] = \tfrac{1}{n}\sum E[x_i] = \tfrac{1}{n}\sum \mu = \mu$$

where all the variables x_i from a random sample have the population's distribution. The $\overline{x}$ variance's distribution is:

$$Var[\overline{x}] = \tfrac{1}{n^2}\sum Var[x_i] = \tfrac{\sigma^2}{n}$$

Therefore, we can conclude that drawing a n size sample from a variable with μ mean, σ^2 variance, and any distribution, the sampling distribution of the mean verifies:

$$E[\overline{x}] = \mu \text{ and } Var[\overline{x}] = \sigma^2/n$$

In terms of probability distributions, we choose the Gamma and the Log-Normal distribution for its resemblance to the original data's distribution (Figure 4).

The Gamma distribution has the following distribution function with the shape-rate parameterization: [6]

$$f(x;\alpha,\beta) = \tfrac{\beta^\alpha x^{\alpha-1} e^{-\beta x}}{\Gamma(\alpha)}$$

where

$$shape = \alpha = \tfrac{\mu^2}{\sigma^2} \text{ and } rate = \beta = \tfrac{\mu}{\sigma^2}$$

The Log-Normal distribution is tightly related with the Normal distribution, being $X \sim N(\mu, \sigma^2)$ the Normal distribution, then $exp(X) \sim Log - N(\mu, \sigma^2)$. The distribution function is defined as: [11]

$$f(x;\mu,\sigma) = \tfrac{1}{x\sigma\sqrt{2\pi}} e^{-(ln(x)-\mu)^2/2\sigma^2}$$

or it can also be defined with the Normal distribution function as:

$$f(x;\mu,\sigma) = \tfrac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

where

$$\mu = log(\frac{\overline{x}^2}{\sqrt{s^2+\overline{x}^2}})$$

and

$$\sigma = \sqrt{log(\frac{s^2}{\overline{x}} + 1)}$$

With the respective distribution functions, we've simulated 1000 samples with the gene's parameters in order to be able to see the resemblance with the Gamma and Log-Normal distribution.

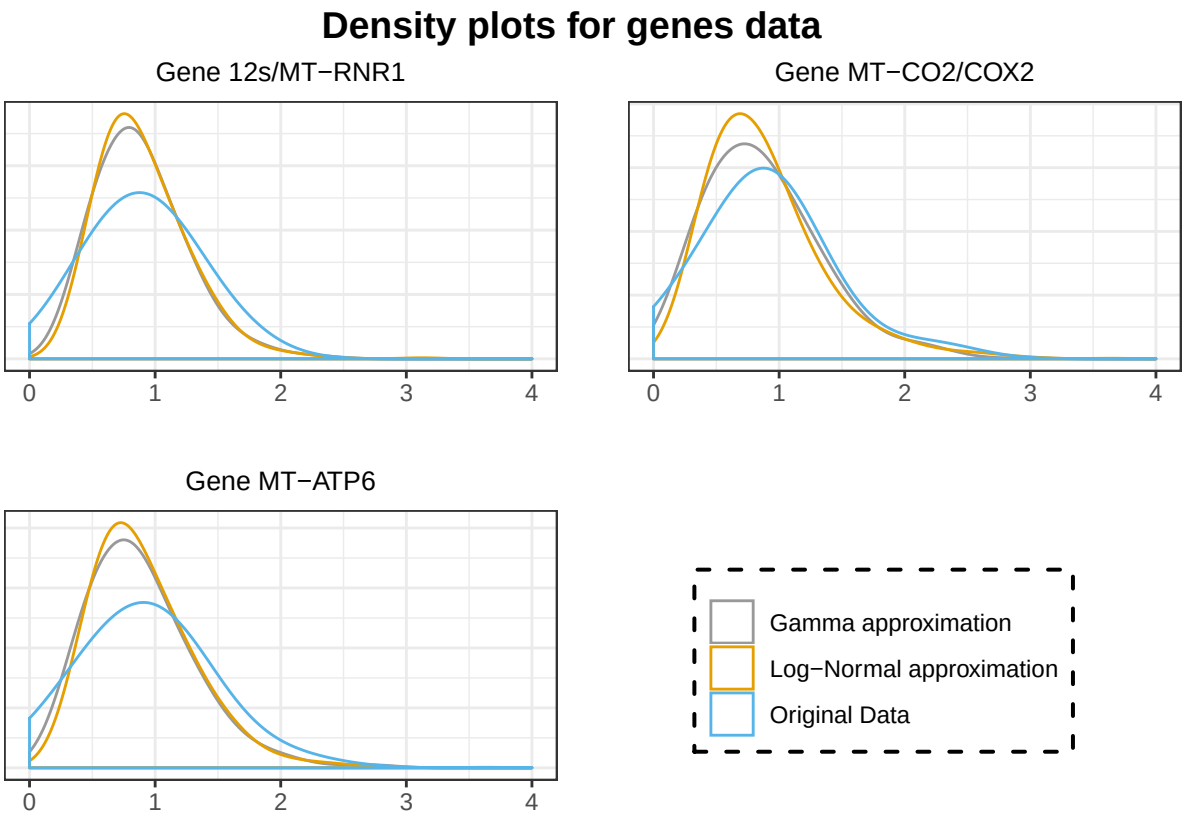

Figure 4: Density plots for the original data versus the fitted distributions

1.8.1 Relation between Log-Normal and Gamma distributions

The Log-Normal distribution and the Gamma distribution are commonly used to shape ratios and rates data because they're naturally bounded on the left by zero and positively skewed. Regarding the gamma distribution, it can represent many shapes by changing the α and β parameters. This versatility makes tha Gamma distribution an attractive candidate for representing this kind of data. In reference to the Log-Normal distribution, which is similar in appearance to the Gamma, it assumes that the logarithms of the data are normally distributed. [2] Modeling with the Log-Normal distributions allows the use of normal-theory statistics on a logarithmic scale where parameter fitting is simple and straightforward. Both distributions are frequently chosen because they can represent a wide variety of shapes from nearly symmetric to highly skewed. [1]

Some sources in the literature indicate that these two distributions are often interchangeable and may provide similar data fit. [4] The main difference of both distributions is in the shape around

the mean, which can be numerically represented by the coefficient of kurtosis. Following with the data example, we will calculate the coefficient of kurtosis for the Gamma and Log-Normal fitted distributions:

Table 2: Coefficient of Kurtosis for fitted distributions

	Gamma	Log-Normal
12s/MT-RNR1	1.108	3.393
MT-CO2/COX2	1.199	10.144
MT-ATP6	1.858	3.221

We can see that the Log-Normal distributions fitted for the three samples have a higher coefficient of kurtosis than the fitted Gamma distributions. In other words, the Log-Normal distribution has a higher concentration of data around the mean than the Gamma which can affect when doing the simulation study as the parametric mean overestimates the sample mean.

1.9 Packages being used

1.9.1 INLA Package

For the Bayesian approach, we will be using the INLA package, acronym for Integrated Nested Laplace Approximations, which is not a CRAN package. It includes a series of functions that allows the user to fit a model, providing different criterias to assess and compare Bayesian models. The two functions we will implement are:

- inla(): performs a full Bayesian analysis of additive models using Integrated Nested Laplace approximation. This function is used to fit the model, specifying the formula, the data and the family which indicates the likelihood family.
- inla.rmarginal(): generates random values from an inla model.

2 Material and methodology

2.1 Statistical methods

2.1.1 Non-parametric methodology

The first tool will be based on a non-parametric technique: the bootstrap approach. The main key of a non-parametric test is that it does not assume anything about the underlying distribution. The non-parametric techniques tend to be more accurate but they have worse statistical power. Those tests are meant to be used when the sample size of the data is small (considering that the sample correctly describes the population it comes from).

The bootstrap approach is a non-parametric technique that uses random sampling with replacement. The basic idea of bootstrapping is that inference can be made from a sample data, which becomes the population of the study (O), that is modeled by resampling the data, that becomes the sample ($\hat{O}$), and performing inference about the samples. When the sample data becomes the population of study, it allows to measure the quality of the sample from resampled data.

The bootstrap methodology can be used to obtain the approximate precision of an estimator without having to assume any underlying probability distribution. It is based on calculating the estimator variance considering the sample is the population and applying the Montecarlo technique to obtain sample replicas. Specifically, given a sample $(x_1, ..., x_n)$, the method proceeds as follows:

1. Considering the sample as the population of a variable that can be any possible value $(x_1, ..., x_n)$ with probability $1/n$. Draw a random sample of size n from the population with Montecarlo methods, equivalent to draw a random sample with replacement from the observed values. The generated sample won't coincide, in general terms, with the original sample, being $(y_1^{[1]}, ..., y_n^{[1]})$ the sample drawn.

2. Compute with the drawn sample the estimator $\hat{v}_1 = \hat{v}(y_1^{[1]}, ..., y_n^{[1]})$ which precision we want to estimate.

3. Repeat steps 1 and 2 B times. We will obtain a sequence of B values from the estimator $\hat{v}_1, ..., \hat{v}_B$ that we consider the distribution of the $\hat{v}$ values. It's average will be:

$$\hat{v}_m = \frac{1}{B} \sum \hat{v}_i$$

and it's variance:

$$Var(\hat{v}) = \frac{1}{B} \sum (\hat{v}_1 - \hat{v}_m)^2$$

It can be proven that, under general conditions, this method asymptotically estimates the estimator's variance and the percentile confidence interval with confidence level $1 - \alpha$ can be computed from the B values $\hat{v}_i$ distribution, calculating values $\hat{v}_{INF}$ and $\hat{v}_{SUP}$ such that:

$$P(\hat{v}_{INF} \leq \hat{v}_i \leq \hat{v}_{SUP}) = 1 - \alpha$$

$(\hat{v}_{INF}, \hat{v}_{SUP})$ are the limits of the percentile confidence interval with confidence level $1 - \alpha$. Those limits are calculated sorting the $\hat{v}_i$ values and taking $\hat{v}_{INF}$ and $\hat{v}_{SUP}$ as the values located on the $[B \times \alpha/2]$ and $[B \times (1 - \alpha/2)]$ positions.

The main advantage of the bootstrap technique is its simplicity as it is an unambiguous way to estimate the standard error and confidence intervals for complex distribution parameters or estimators, e.g. percentiles, proportions, odds ratios, and correlation coefficients. Another highlight is that this methodology allows to check the stability of the results. In terms of cost, using bootstrap

avoids having to collect large amounts of data while still getting reliable and unbiased results. On the other side, bootstrapping does not always provide guarantees because the small sample maybe not fully representative of the population. It can also be time-consuming when the number of iterations is large [14].

2.1.2 Bayesian methodology

The second set of procedures we will be using is based on Bayesian statistics. Bayesian methods reduce statistical inference problems in probability theory, minimizing the need for new concepts. It only requires the mathematics of probability theory and the interpretation of probability, expressing a degree of belief in an event. This degree of belief may be based on prior knowledge about the event, such as the results of previous experiments, or on personal beliefs about the event.

Bayesian analysis is based on Bayes' theorem which is stated mathematically as the following equation:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

where A and B are events, $P(B) \neq 0$, $P(A|B)$ is the likelihood of event A occurring given that B is true, $P(B|A)$ is the likelihood of event B occurring given that A is true and $P(A)$ and $P(B)$ are the probabilities of observing A and B respectively.

In the Bayesian paradigm, the process of learning from the data is systematically implemented by making use of Bayes' theorem to combine the available prior information with the information provided by the data to produce the required posterior distribution. There is an established process which involves the following steps:

1. Definition of the prior distribution that incorporates the subjective beliefs about a parameter.
2. Gather data.
3. Update the prior distribution with the data using Bayes' theorem to obtain a posterior distribution, which is a probability distribution that represents your updated beliefs about the parameter after having seen some data.
4. Analyze the posterior distribution and summarize it.

The advantages of the Bayesian methodology are that it allows you to combine data with prior information about a parameter and form, even once you have more observations, you can use the previous posterior distribution as a prior. It also provides exact inferences as it does not rely on asymptotic approximation, which also implies that small sample size inferences perform as well as large sample size. On the contrary, when using Bayesian methodologies you have to proceed with caution in order to avoid misleading results. As there is no correct way to choose a prior, Bayesian inferences require skills to translate subjective prior beliefs into mathematically formulated prior distribution, which can heavily influence posterior distributions. In terms of costs, Bayesian procedures come with a high computational cost, especially in models with a large number of parameters. [16]

2.1.2.1 INLA approach

INLA is a deterministic paradigm for Bayesian inference that relies on a combination of analytical approximations to achieve highly accurate deterministic approximations. The main advantage of using INLA instead of classical Bayesian techniques is that it is faster from a computational point of

view, even for complex models. Also, as INLA is a deterministic algorithm, it does not suffer from slow convergence.

INLA uses a Bayesian framework to fit generalized additive regression models that explain observed response variable through covariates and random effects:

$$y \sim \beta_0 + \beta_1 \times covar_1 + ... + \beta_m \times covar_m + \text{random effects}$$

where $\beta_0 + \beta_1 \times covar_1 + ... + \beta_m \times covar_m$ are the fixed effects and $\beta_0 + \beta_1 \times covar_1 + ... + \beta_m \times covar_m$ + random effects is the lineal predictor.

Even though INLA can fit a large variety of response distributions for y, we will be using the Gamma Distribution to fit the response variable.

2.2 Simulations

A simulation study has been carried out in order to assess the performance of the methods presented here. We will compare Bootstrap with the Bayesian approach under two situations: (A) Random samples of a Gamma distribution with different shapes and rates, and (B) Random samples of a log-Normal distribution, once again, with different parameters for the mu and sigma. With these different scenarios we will be able to assess which method performs better under different circumstances.

The simulations procedure consists in generating a sample of N=5, N=10, N=20 and N=30 observations from simulated distributions and asses the estimation of $P(\overline{X} > \mu)$ using Bootsratio and Bayesian method.

For the Bootstrap methodology, we will do 10000 simulations; for each iteration we will generate 1000 "re-samples" with replacement from the N-size generated sample and then compute the confidence interval for the mean. With the 10000 confidence intervals, we will then compute the average coverage and the length.

For the Bayesian methodology, we will also do 10000 simulations. For each one, we will draw a N-size sample from the popublation and we will consider it the sample. With it, we will approximate a Gamma distribution and generate 1000 random values from the computed distribution. Finally, we will calculate the confidence interval for that specific iteration with the generated values and evaluate the method with the coverage and length metrics.

2.3 Confidence Intervals

In order to assess the performance of the methodologies presented, we will be calculating the following confidence intervals for the measure of interest: the mean of the simulated population.

Percentile confidence interval: this is an approach of the ideal situation with infinite samples defined as:

$$[\hat{\theta}_{\%low}; \hat{\theta}_{\%up}] = [\hat{\theta}^*_{\frac{\alpha}{2}}; \hat{\theta}^*_{1-\frac{\alpha}{2}}]$$

As we have a finite number of samples B, we will define $\hat{\theta}^*_{B(\alpha)}$ as the α-th value of the $\hat{\theta}^*(b)$ estimator, in other word, the $B * \alpha$-th value on the ascending ordered list of the B replicas of $\hat{\theta}^*$. Following the same logic, we define $\hat{\theta}^*_{B(1-\alpha)}$ as the $1-\alpha$-th percentile of the values. Thereby, the $1-\alpha$ percentile confidence interval is traced as:

$$[\hat{\theta}_{\%low}; \hat{\theta}_{\%up}] \approx [\hat{\theta}^*_{B(\frac{\alpha}{2})}; \hat{\theta}^*_{B(1-\frac{\alpha}{2})})]$$

This interval can lead to errroneous estimations when the estimator is biased in respect to the parameter we are trying to aproximate. [14]

Bias Corrected and accelerated interval: this can be considered an improved version of the Percentile confidence interval. The main advantage is that it corrects for bias and skewness in the distribution of bootstrap estimates. To compute this interval we need two parameters:

- z_0: represents the proportion of bootstrap estimates that are less than the observed statistic. It is computed as:

$$\hat{z}_0 = \Phi^{-1}(\frac{\hat{\theta}^*(b)<\hat{\theta}}{B})$$

- a: represents the acceleration parameter that is proportional to the skewness and it corrects when the standard error $se(\hat{\theta})$ is not constant. It can be estimated with the jackknife method as follows:

$$\hat{a} = \frac{\sum_{i=1}^{n}(\hat{\theta}_{(\mathring{u})} - \hat{\theta}_i)^3}{6[\sum_{i=1}^{n}(\hat{\theta}_{(\mathring{u})} - \hat{\theta}_{(i)})^2]^{3/2}}$$

where $\hat{\theta}_{(i)}$ is the i-th jackknife replica of the estimator such as:

$$\hat{\theta}_{(i)} = s(x_{(i)})\ \hat{\theta}_{(\mathring{u})} = \frac{1}{n}\sum_{i=1}^{n}\hat{\theta}_{(i)}$$

With that information, we can construct whe BCa interval with the following formulation:

$$[\hat{\theta}^*_{\alpha_1}; \hat{\theta}^*_{\alpha_2}]$$

where

$$\alpha_1 = \Phi * \hat{z}_0 + \frac{\hat{z}_0 + z_{1-\frac{\alpha}{2}}}{1-\hat{a}(\hat{z}_0 + z_{1-\frac{\alpha}{2}})}$$

$$\alpha_2 = \Phi * \hat{z}_0 + \frac{\hat{z}_0 + z_{\frac{\alpha}{2}}}{1-\hat{a}(\hat{z}_0 + z_{\frac{\alpha}{2}})}$$

being $z_{\frac{\alpha}{2}}$ and $z_{1-\frac{\alpha}{2}}$ the quantiles of the N(0,1) distribution and Φ the distribution funcion of the N(0,1).

Note that when $\hat{z}_0 = \hat{a} = 0$ the BCa interval matches the percentile confidence interval:

$$\alpha_1 = \Phi(z_{1-\frac{\alpha}{2}}) = \frac{\alpha}{2}$$
$$\alpha_2 = \Phi(z_{\frac{\alpha}{2}}) = 1 - \frac{\alpha}{2}$$

2.4 Coverage and length as a performance measure

To asses the performance of the test, we will be using two main metrics based on confidence intervals: coverage and length. The coverage, also known as the empirical confidence level, is measured as the proportion of intervals that contain the measure of interest [3], in our simulation study it will be the mean of the distribution we are generating the samples with. Supposing X is the random variable of interest and θ is the parameter to be estimated, for each replicate indexed $j = 1, ..., m$:

(a) Generate j^{th} random samples.
(b) Compute the confidence interval C_j for the j^{th} sample.
(c) Compute $y_j = I(\theta \in C_j)$ for the j^{th} sample.

To compute the empirical confidence level after the m replicas:

$$\bar{y} = \frac{1}{m} \sum_{j=1}^{m} y_j$$

A good confidence interval should have coverage close to the nominal confidence level $1 - \alpha$ if all the assumptions used in deriving a confidence interval are met. If the coverage is greater than the $1 - \alpha$, the interval is termed "conservative" (also "wider" or "narrow") and, if the coverage is lower than the actual nominal confidence level, it is termed "permissive". [3]

The other metric will be the length, the typical measure for confidence intervals calculated as the subtraction of the upper side of the interval, and the lower side. The expected length is the shorter, the better. To calculate the length on a simulation study, supposing X is the random variable of interest, for each replicate indexed $j = 1, ..., m$: [3]

(a) Generate j^{th} random samples.
(b) Compute the confidence interval C_j for the j^{th} sample.
(c) Having $IC_\alpha = [x^-, x^+]$, compute the length $L = x^+ - x^-$ for the j^{th} sample.

To compute the length after the m replicas:

$$\bar{L} = \frac{1}{L} \sum_{j=1}^{m} L_j$$

As a summary, the ideal confidence interval IC_α should have a coverage or empirical confidence level as close as possible to the α and the sorter length, the better.

3 Simulation study

3.1 Functions applied

- simScenarioGamma: function to compute the simulations for the Gamma distribution. The output is the coverage and length of the confidence interval selected and its standard deviations. The definition of the arguments is the following:
  - shape: parameter shape for the Gamma
  - rate: parameter rate for the Gamma
  - n: sample size
  - nsims: number of simulations, by default 10000
  - conf.level: the desired confidence level, by default = 0.95
  - N: number of replicas for the confidence interval, by default 1000
  - z0: bias-correction factor, by default NULL
  - a: acceleration parameter, by default NULL
  - method: statistical method: "boot" for bootstrap methodology (by default), "bayesian" for Bayesian methodology

```
simScenarioGamma <- function(shape, rate, n, nsims = 10000,
    conf.level = 0.95, N = 1000, z0 = NULL, a = NULL,
    method = "boot") {
    if (sum(method == c("boot", "bayesian")) != 1) {
        return("Cannot compute this method, allowed methods are boot and bayesian")
    }
    mean.popul <- shape/rate
    if (method == "boot") {
        simOut <- replicate(nsims, ciPercentile(rgamma(n,
            shape = shape, rate = rate), conf.level,
            N, z0, a))
    } else {
        simOut <- replicate(nsims, ciPercentile_Bayesian(exp(inla.rmarginal(nsims,
            inla(RE ~ 1, family = "gamma", data = data.frame(RE = sample(rgamma(nsims,
                shape = shape, rate = rate), n)), control.predictor = list(compute = T),
                control.compute = list(dic = T, mlik = T))$marginals.fixed$`(Intercept)`)),
            conf.level))
    }
    true.coverage <- sum((simOut[1, ] <= mean.popul) &
        (simOut[2, ] >= mean.popul))/nsims
    lens <- simOut[2, ] - simOut[1, ]
    mean.length <- mean(lens)
    sqrt.nsims <- sqrt(nsims)
    z0.025 <- qnorm(0.025, lower.tail = FALSE)
    return(c(true.coverage = true.coverage, `?coverage` = z0.025 *
        sqrt(true.coverage * (1 - true.coverage))/sqrt.nsims,
        mean.length = mean.length, `?length` = z0.025 *
            sd(lens)/sqrt.nsims))
}
```

- simScenarioLogNorm: function to compute the simulations for the Log-Normal distribution. The output is the coverage and length of the confidence interval selected and its standard deviations. The definition of the arguments is the following:
  - meanlog: parameter mean for the Log-Normal
  - sdlog: parameter standard deviation for the Log-Normal
  - n: sample size
  - nsims: number of simulations, by default 1000
  - conf.level: the desired confidence level, by default = 0.95
  - N: number of replicas for the confidence interval, by default 1000
  - z0: bias-correction factor, by default NULL
  - a: acceleration parameter, by default NULL
  - method: statistical method: "boot" for bootstrap methodology (by default), "bayesian" for Bayesian methodology

```
simScenarioLogNorm <- function(mean, std, n, nsims = 1000,
    conf.level = 0.95, b = 1000, z0 = NULL, a = NULL,
    method = "boot") {
    if (sum(method == c("boot", "bayesian")) != 1) {
        return("Cannot compute this method, allowed methods are boot and bayesian")
    }
    mean.popul <- exp(mu + sd^2/2)
    if (method == "boot") {
        simOut <- replicate(nsims, ciPercentile(exp(rnorm(n,
            mean = mean, sd = std)), conf.level, b,
            z0, a))
    } else {
        simOut <- replicate(nsims, ciPercentile_Bayesian(exp(inla.rmarginal(nsims,
            inla(RE ~ 1, family = "gamma", data = data.frame(RE = sample(exp(rnorm(nsims,
                shape = shape, rate = rate)), n)),
                control.predictor = list(compute = T),
                control.compute = list(dic = T, mlik = T))$marginals.fixed$`(Intercept)`)),
            conf.level, N, z0, a))
    }
    true.coverage <- sum((simOut[1, ] <= mean.popul) &
        (simOut[2, ] >= mean.popul))/nsims
    lens <- simOut[2, ] - simOut[1, ]
    mean.length <- mean(lens)
    sqrt.nsims <- sqrt(nsims)
    z0.025 <- qnorm(0.025, lower.tail = FALSE)
    return(c(true.coverage = true.coverage, `?coverage` = z0.025 *
        sqrt(true.coverage * (1 - true.coverage))/sqrt.nsims,
        mean.length = mean.length, `?length` = z0.025 *
            sd(lens)/sqrt.nsims))
}
```

- ciPercentile: auxiliar function that calculates the confidence interval for the Bootstrap data introduced. With the parameters, we can choose to calculate the percentile confidence interval or the BCa interval. The definition of the arguments is the following:
  - x: data
  - conf.level: the desired confidence level, by default = 0.95
  - N: number of replicas, by default 1000
  - z0: bias-correction factor, by default NULL
  - a: acceleration parameter, by default NULL

```
ciPercentile <- function(x, conf.level = 0.95, N = 1000,
    z0 = NULL, a = NULL) {
    n <- length(x)
    inv.n <- 1/n
    mean.x <- sum(x) * inv.n
    zalpha <- -qnorm(0.5 * (1 - conf.level))
    mean.boot <- replicate(N, sum(sample(x, replace = TRUE)) *
        inv.n)
    if (is.null(a)) {
        inv.n_1 <- 1/(n - 1)
        mean_i <- numeric(n)
        for (i in 1:n) {
            mean_i[i] <- sum(x[-i]) * inv.n_1
        }
        mean_i <- sum(mean_i) * inv.n - mean_i
        a <- sum(mean_i^3)/(6 * sum(mean_i^2)^1.5)
    }
    if (is.null(z0)) {
        z0 <- qnorm(sum(mean.boot <= mean.x)/N)
    }
    ci <- quantile(mean.boot, probs = c(pnorm(z0 +
        (z0 - zalpha)/(1 - a * (z0 - zalpha))), pnorm(z0 +
        (z0 + zalpha)/(1 - a * (z0 + zalpha)))))
    names(ci) = NULL
    attr(ci, "conf.level") = conf.level
    attr(ci, "z0") <- z0
    attr(ci, "a") <- a
    return(ci)
}
```

- ciPercentile_Bayesian: auxiliar funcion that calculates the confidence interval for the Bayesian data introduced. In this case, the data introduced is a vector of N random values simulated with the Bayesian estimated distribution. The definition of the arguments is the following:
  - x: data
  - conf.level: the desired confidence level, by default = 0.95

```
ciPercentile_Bayesian <- function(x, conf.level = 0.95) {
    ci <- quantile(x, probs = c((1 - conf.level)/2,
        1 - ((1 - conf.level)/2)))
    return(ci)
}
```

3.2 Parameterization

In terms of parameters, we want to copy as much as possible the reality of the data. For that reason, we won't be specifying the parameters for the simulation, but instead we will present different scenarios varying the mean and the variation. As a reminder, we are working with data centered around 1, with small variance and right-skewed. You'll find the following scenarios:

Table 3: Parameters for the simulation study

Population Parameters		X ~ Gamma(shape,rate)		ln(X) ~ N(mu, sigma)	
Mean	Variance	Shape	Rate	Mu	Sigma
0.9	0.05	16.20	18.00	-0.135	0.245
0.9	0.10	8.10	9.00	-0.164	0.341
0.9	0.15	5.40	6.00	-0.190	0.412
0.9	0.20	4.05	4.50	-0.216	0.470
0.9	0.25	3.24	3.60	-0.240	0.519
0.9	0.30	2.70	3.00	-0.263	0.561
1.0	0.05	20.00	20.00	-0.024	0.221
1.0	0.10	10.00	10.00	-0.048	0.309
1.0	0.15	6.67	6.67	-0.070	0.374
1.0	0.20	5.00	5.00	-0.091	0.427
1.0	0.25	4.00	4.00	-0.112	0.472
1.0	0.30	3.33	3.33	-0.131	0.512
1.1	0.05	24.20	22.00	0.075	0.201
1.1	0.10	12.10	11.00	0.056	0.282
1.1	0.15	8.07	7.33	0.037	0.342
1.1	0.20	6.05	5.50	0.019	0.391
1.1	0.25	4.84	4.40	0.001	0.433
1.1	0.30	4.03	3.67	-0.015	0.471

3.3 Results

3.3.1 Gamma Distribution

3.3.1.1 Small Sample Size N = 5

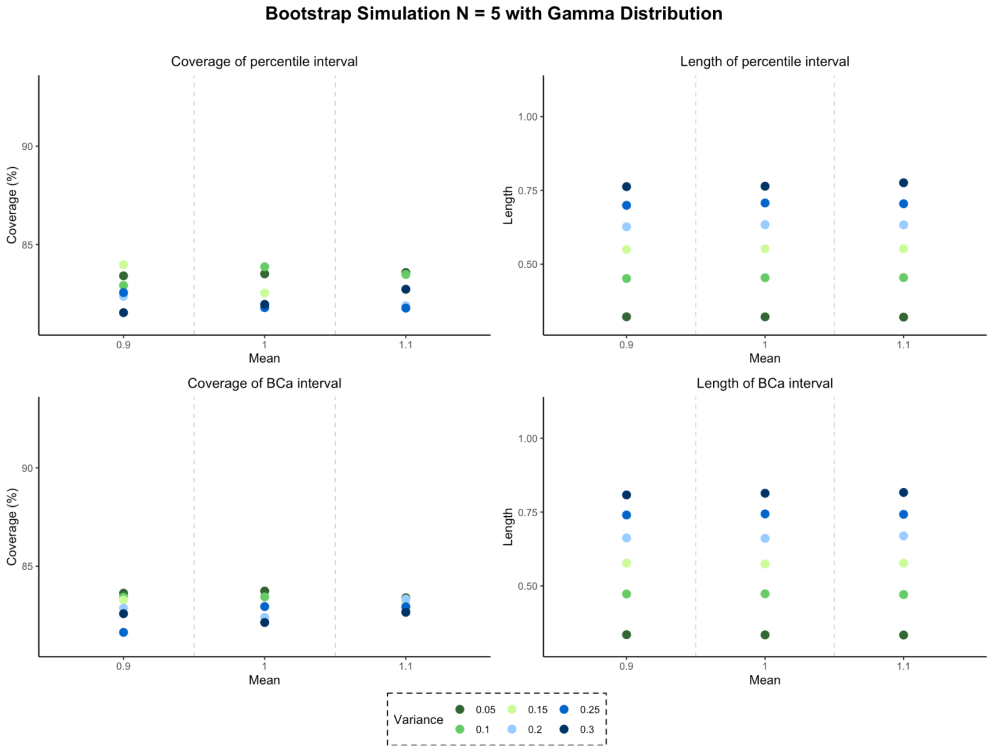

Figure 5: Coverage and length of the computed CI with Bootstrap N = 5 and Gamma Distribution

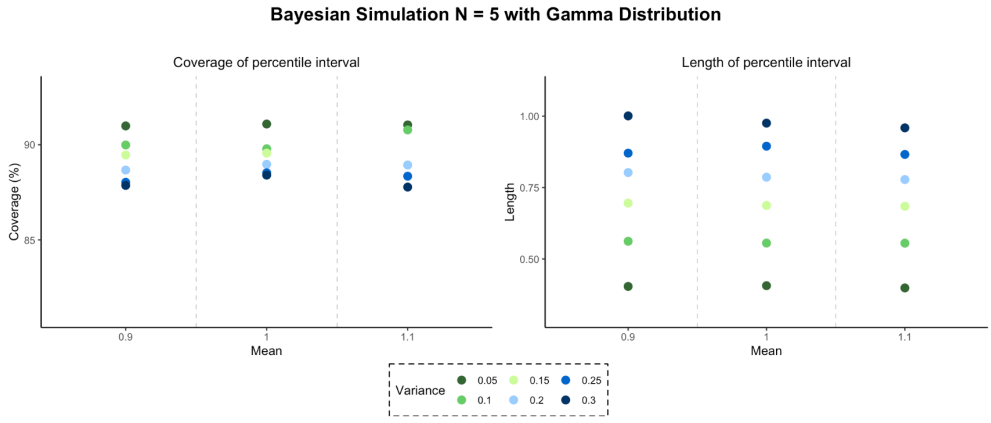

Figure 6: Coverage and length of the computed CI with Bayes N = 5 and Gamma Distribution

3.3.1.2 Small Sample Size N = 10

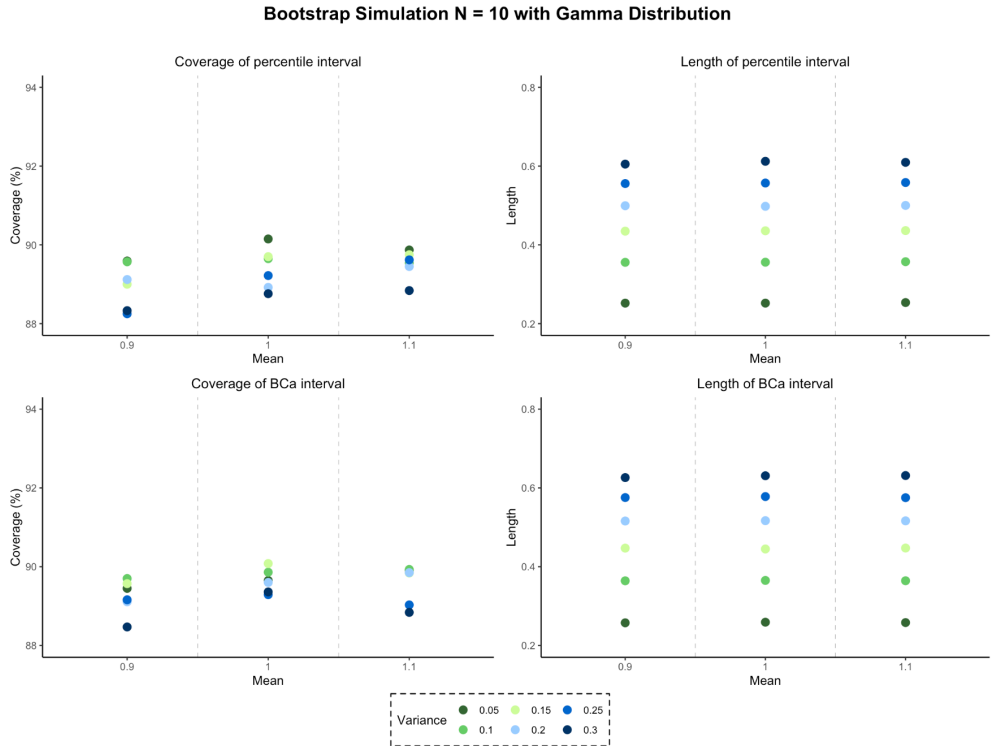

Figure 7: Coverage and length of the computed CI with Bootstrap N = 10 and Gamma Distribution

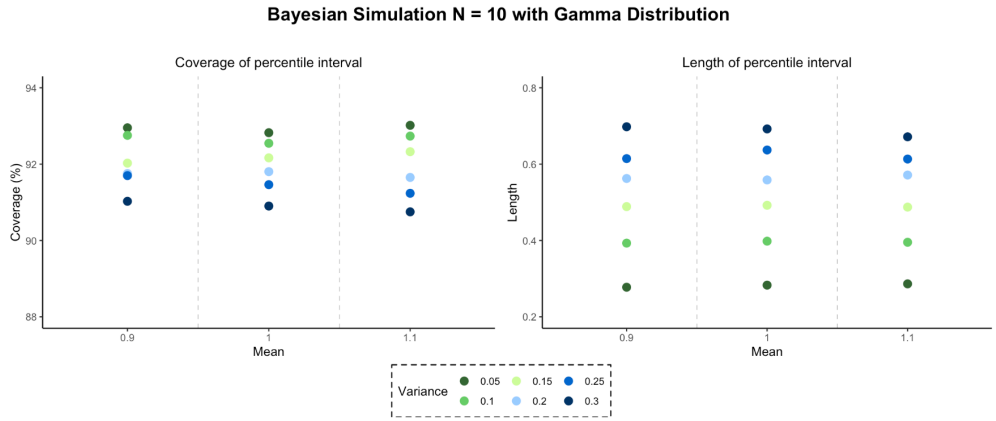

Figure 8: Coverage and length of the computed CI with Bayes N = 10 and Gamma Distribution

3.3.1.3 Medium Sample Size N = 20

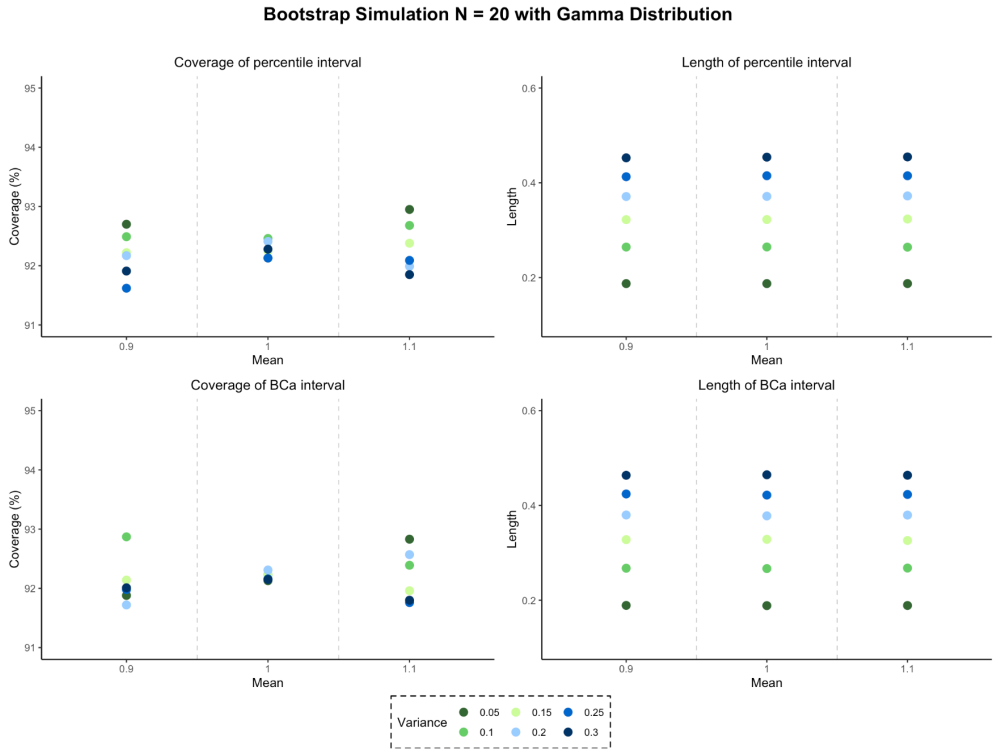

Figure 9: Coverage and length of the computed CI with Bootstrap N = 20 and Gamma Distribution

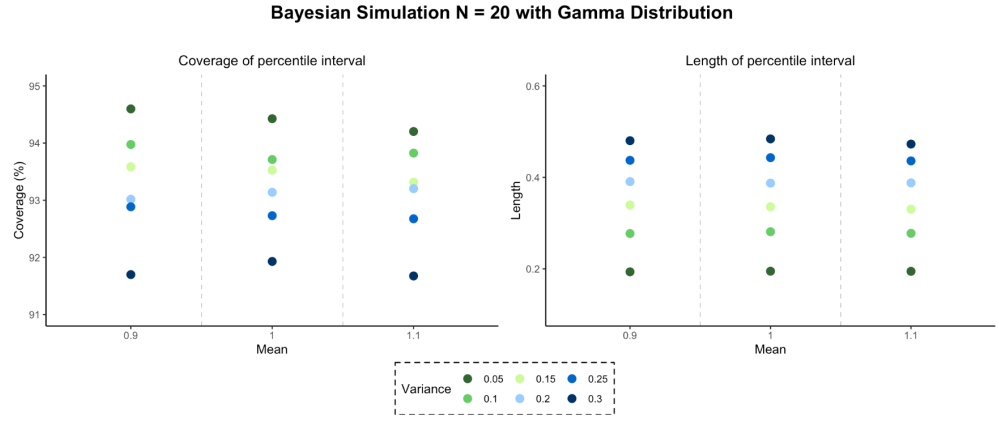

Figure 10: Coverage and length of the computed CI with Bayes N = 20 and Gamma Distribution

3.3.1.4 Large Sample Size N = 30

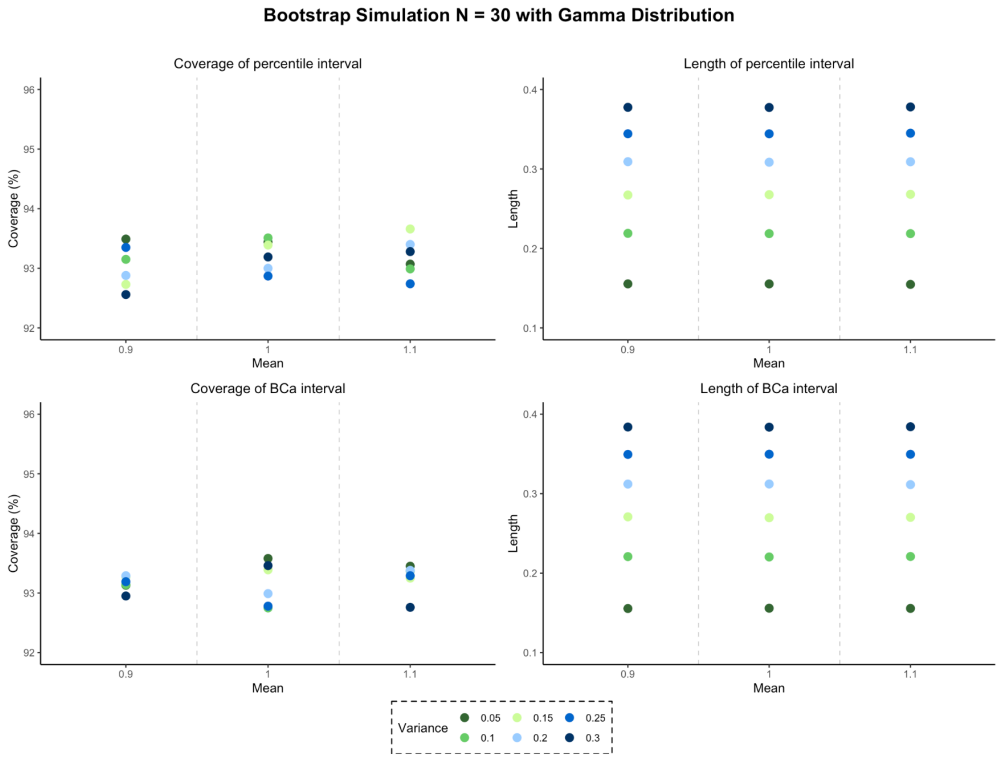

Figure 11: Coverage and length of the computed CI with Bootstrap N = 30 and Gamma Distribution

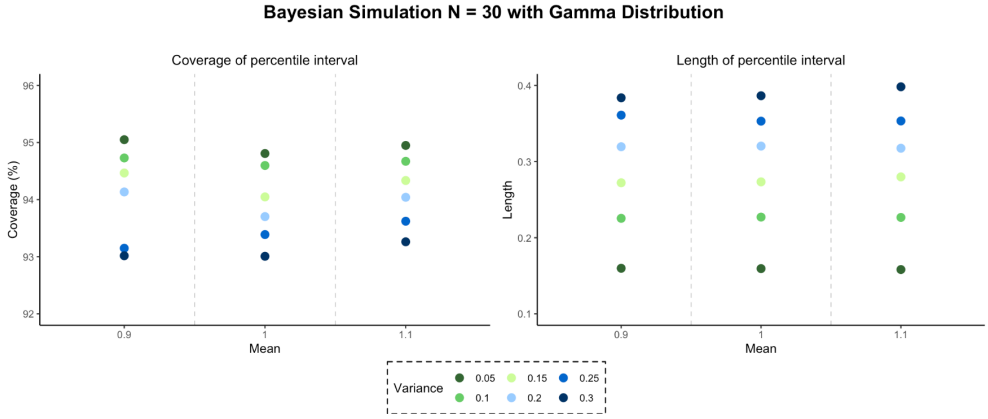

Figure 12: Coverage and length of the computed CI with Bayes N = 30 and Gamma Distribution

3.3.2 Log-Normal Distribution

3.3.2.1 Small Sample Size N = 5

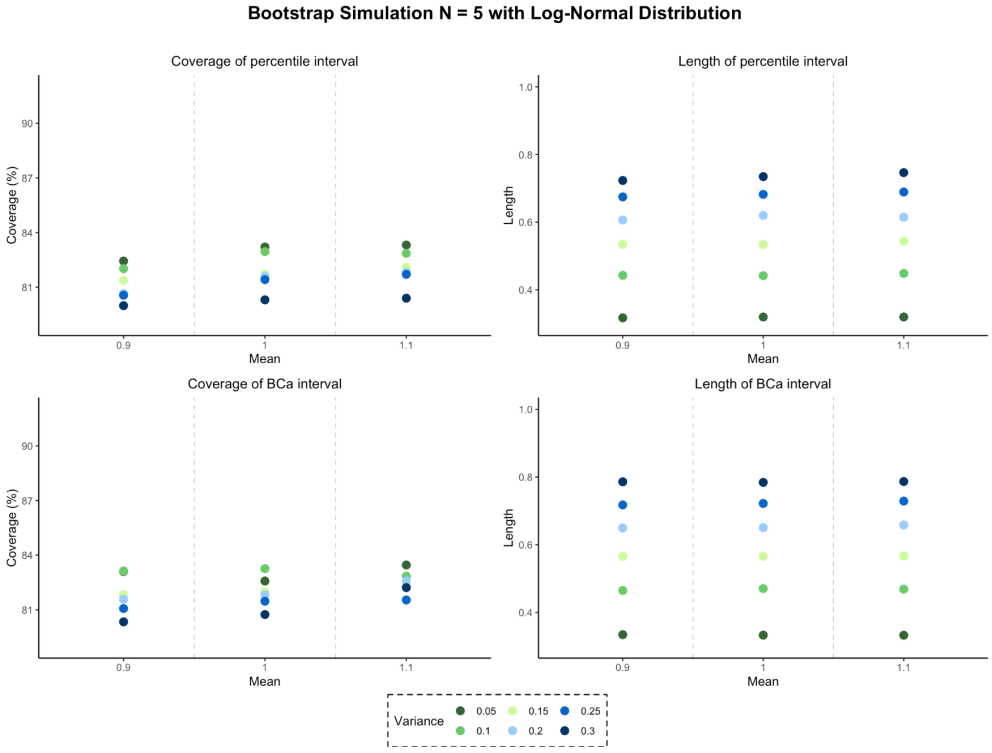

Figure 13: Coverage and length of the computed CI with Bootstrap N=5 and L-N Distribution

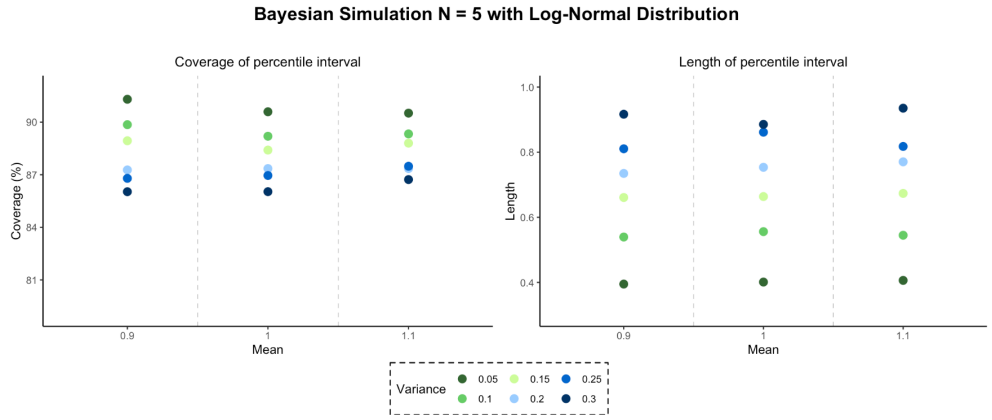

Figure 14: Coverage and length of the computed CI with Bayes N=5 and L-N Distribution

3.3.2.2 Small Sample Size N = 10

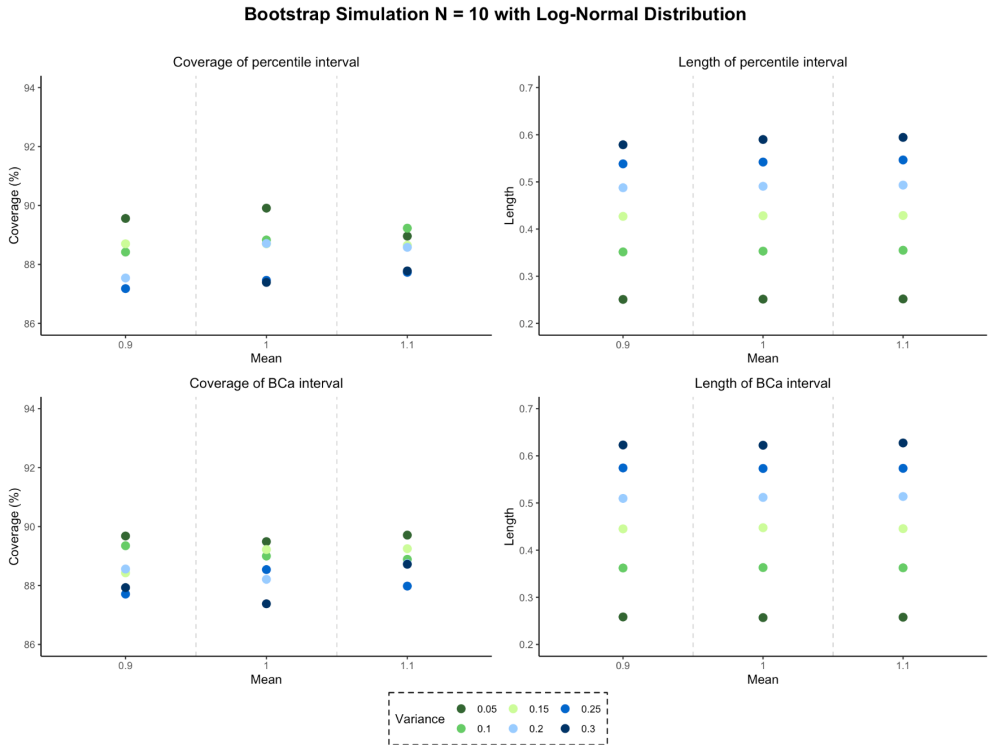

Figure 15: Coverage and length of the computed CI with Bootstrap N=10 and L-N Distribution

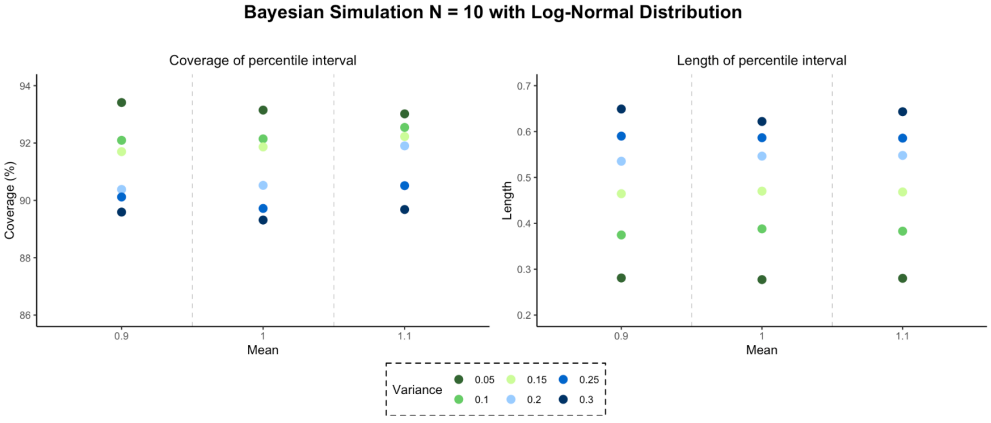

Figure 16: Coverage and length of the computed CI with Bayes N=10 and L-N Distribution

3.3.2.3 Medium Sample Size N = 20

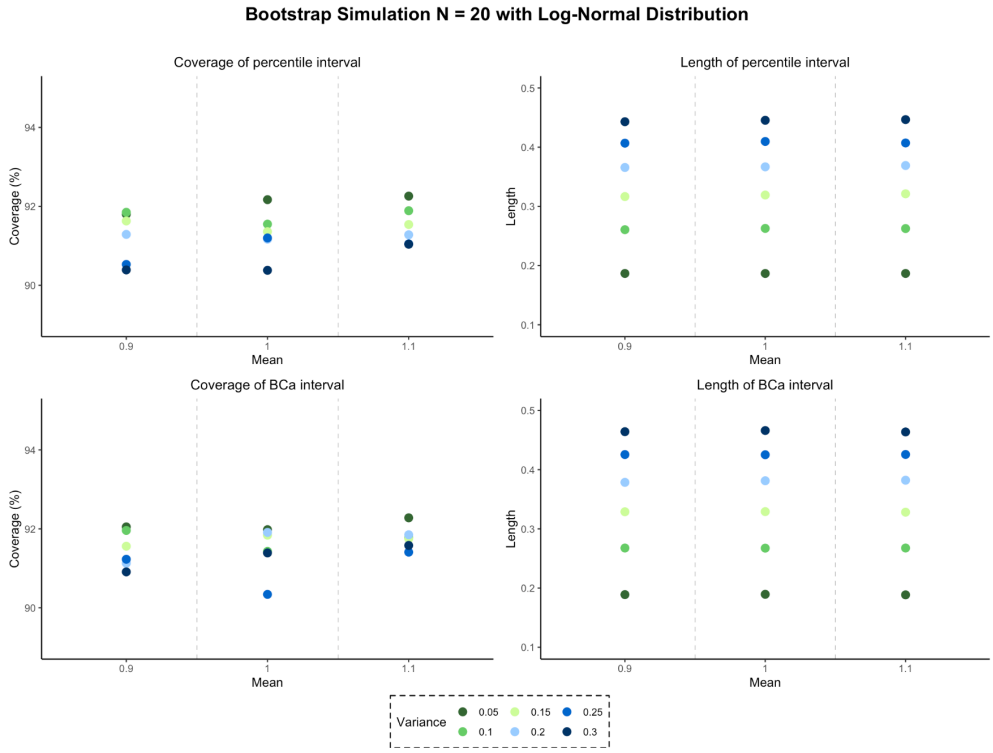

Figure 17: Coverage and length of the computed CI with Bootstrap N=20 and L-N Distribution

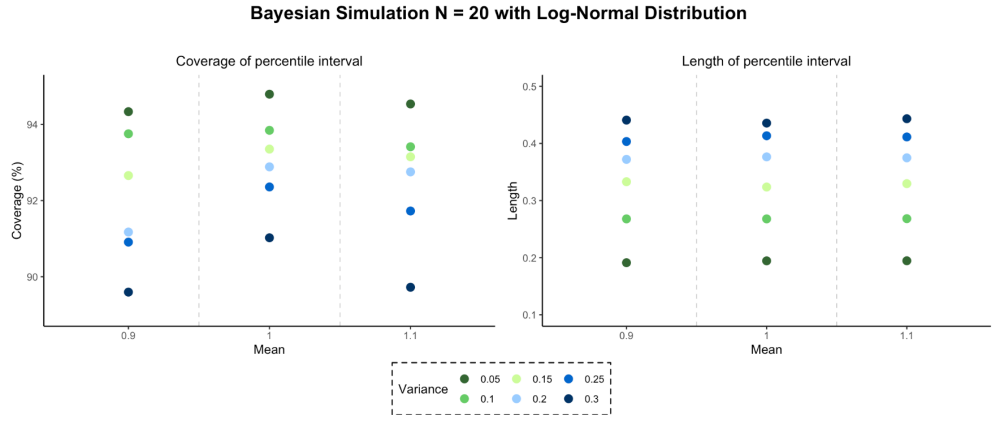

Figure 18: Coverage and length of the computed CI with Bayes N=20 and L-N Distribution

3.3.2.4 Large Sample Size N = 30

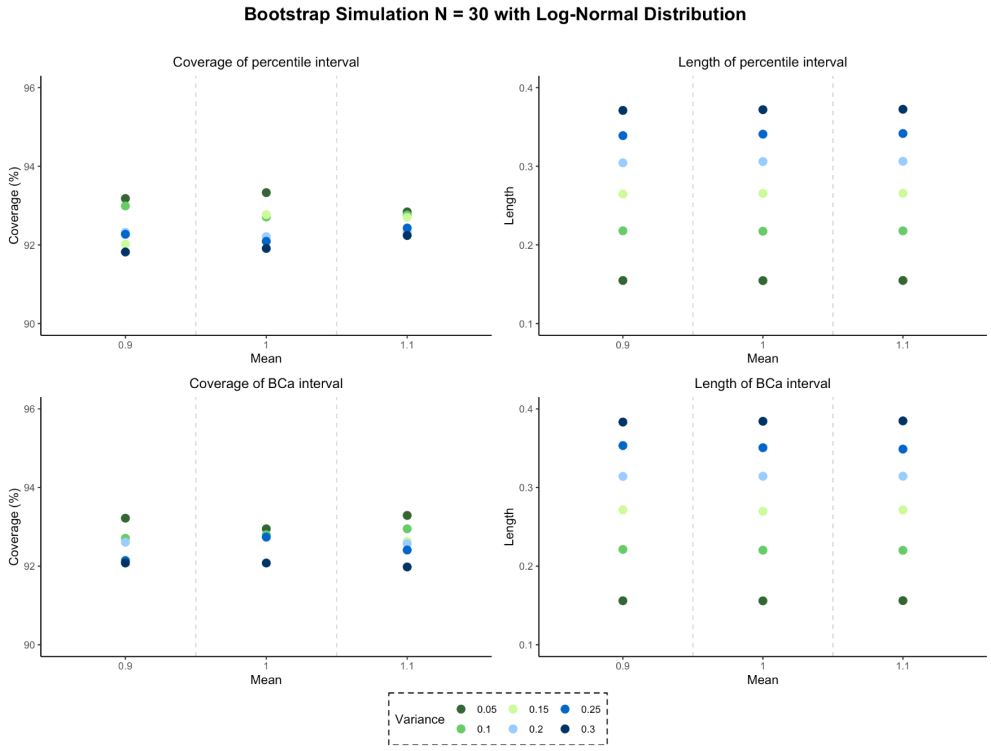

Figure 19: Coverage and length of the computed CI with Bootstrap N=30 and L-N Distribution

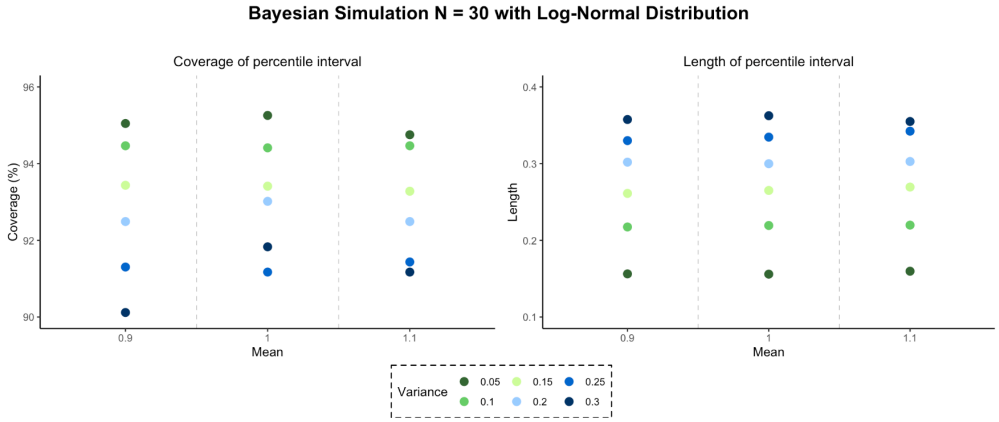

Figure 20: Coverage and length of the computed CI with Bayes N=30 and L-N Distribution

4 Discussion

The lack in statistical literature of a clear methodology to manipulate and analyze ratiometric data has lead to inconsistent conclusions when analyzing PCR data where researchers want to estime the probability of the ratio being greater that 1. Most of the times the techniques used in these scenarios need the assumption of an underlying probability distribution which is, oftenly, lacking because of the small sample size. This can lead to a discrepancy in the conclusions between laboratories or research groups. The approach we have taken is to use the Bootstrap and the Bayesian methodologies and the confidence interval so we will evaluate which can be more accurate to use in different circumstances.

The first assumption we did when performing the simulation study was to draw random data from Gamma and Log-Normal distributed populations as their characteristics resemble how the ratiometric data is distributed. We performed the same simulations for both distributions with different population parameters. To evaluate the performance of the Bootstrap and the Bayesian methods, we used the coverage of the confidence interval with 95% confidence level as a performance measure, obtaining the results that can be seen on Figure 21.

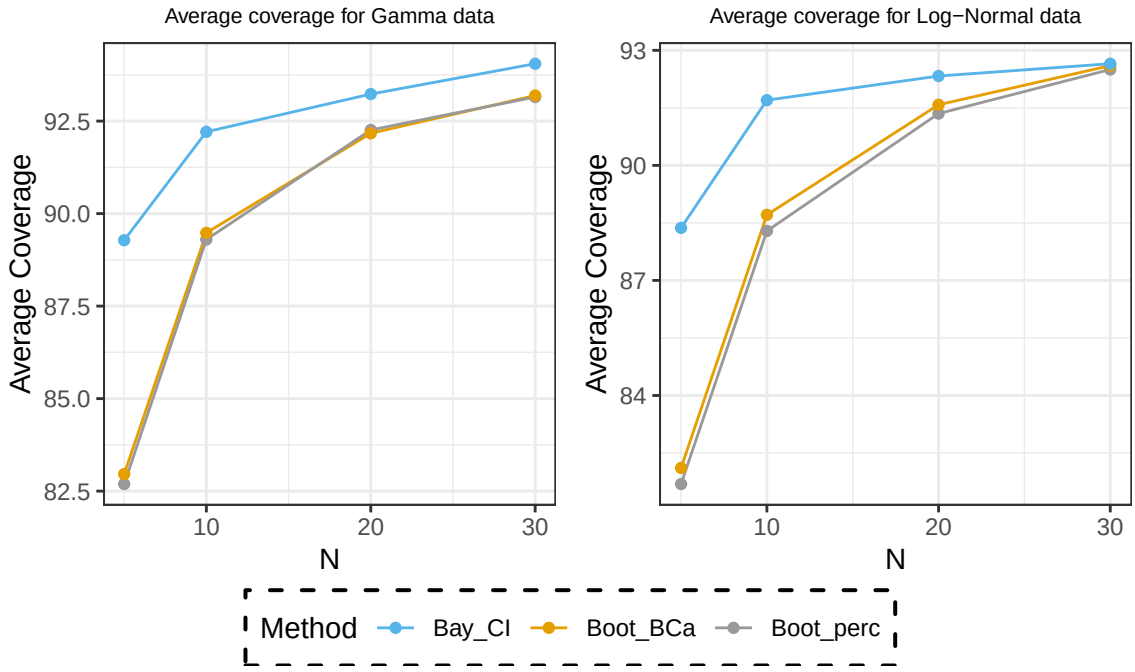

Figure 21: Average coverage for Gamma and Log-Normal by sample size and method

For both methodologies we can observe that the coverage asymptotically converges the confidence level of the interval, so there's evidence that, bigger the sample size, higher the accuracy. Regarding the Boostrap methogology, we wanted to compute not only the percentile interval, but also the BCa interval to automatically correct the bias. With that correction, the coverage of the confidence interval seems to be a slight bit higher, specially for the Log-Normal distribution data. Comparing Bayesian approach with Bootstrap, we can infer that Bayesian methodology has a higher accuracy, not only with small size samples but also with bigger ones.

Although we concluded that Bayesian estimates better, the high computational cost of this methodology leads us to recommend its use only for small sample sizes, where the difference between both approaches is higher. With the performed simulation study, we can guarantee that, under the same

study conditions including sample size, parameters and iterations, the Bayesian simulation takes 64 times longer than Bootstrap.

On the other side, we also wanted to measure the precision of both methodologies and, for that reason, we also measured the length of the confidence intervals. See results on Figure 22.

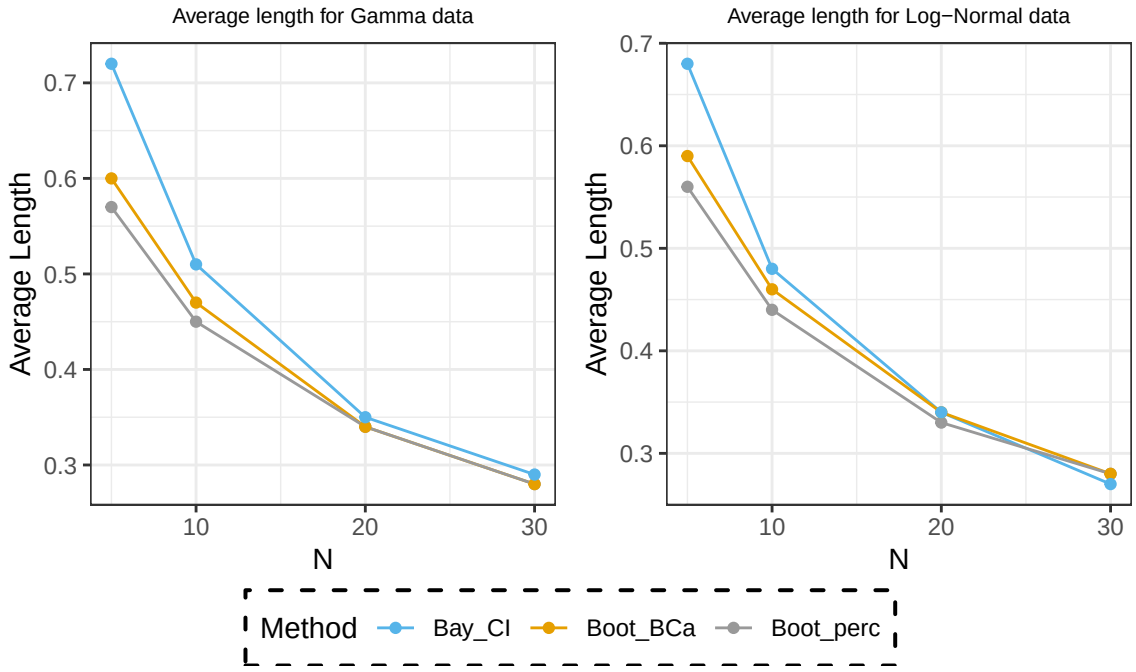

Figure 22: Average length for Gamma and Log-Normal by sample size and method

In terms of precision, the Bayesian methodology is better for small-medium sample sizes but, as seen on the plots, as the sample size gets bigger, both methodologies converge. Also, as expected, for small size samples, the confidence intervals include a wider range of values and, as the sample size gets bigger, the interval gains precision and so the length is smaller.

We mentioned previously that the Log-Normal distribution has a higher coefficient of kurtosis than the Gamma distribution, affecting on the concentration values around the mean. As expected, this affects on the simulation study as the parametric mean overestimates the sample mean and, as a consequence, the coverage is higher than for the Gamma distribution under the same population parameters, and the confidence interval's length is shorter. With that information, we can conclude that the less dispersion that has the sample data, the higher the accuracy will be.

To evaluate different population pictures, we added different variance parameters so we could analyze how the noise affects to the estimation. There's a clear pattern on the results independently of the method, the probability distribution and the sample size: as the variance gets bigger, the coverage decreases and the other way happens for the interval's length, as the variance gets bigger, the length increases.

As a conclusion, we can observe that, for both methodologies, the coverage estimator is under 95% (which is our confidence level) in almost every scenario we studied, so we have permissive confidence intervals. The coverage estimator asymptotically converges to the confidence level as the sample size gets bigger, at the same time the confidence interval's length gets shorter. Meanwhile the Bootstrap

technique converges slower, the computational cost is much lower compared with the Bayesian technique. For that reason, we recommend to use the Bayesian approach exclusively for small sample sizes, where the coverage estimator differs more between the Bootstrap and the Bayesian technique. We suggest to opt for Bootstrap approach when we are working with medium-large sample size as the coverage level is very close to the one obtained with Bayesian and the computational cost in terms of time is 64 times less. As a last punctuation, we saw that when working with Log-Normal distributed data, the estimations were more accurate obtaining higher coverage levels and shorter lengths. This indicates that we will obtain better results when working with data with little dispersion.

5 Glossary

DNA: acronym for Deoxyribonucleic acid, a molecule that contains the hereditary material in humans and almost all other organisms. It is a double helix formed by four chemical bases: (A) adenine, (G) guanine, (C) cytosine and (T) thymine, and they pair up with each other (A with T and C with G) to form units called base pairs. Human DNA consists of about 3 million bases and their sequence determines the information available for building and maintaining the organism.

RNA: acronym for Ribonucleic acid, a single-stranded molecule and consists of short chain of nucleotides and can be RNA molecule can form a double helix by complementary base pairing. The nitrogenous bases in RNA are (A) adenine, (G) guanine, (C) cytosine, and (U) uracil, which replaces thymine (T) in DNA.

Nucleic acid: large biomolecules essential to all forms of life, its the general term for DNA and RNA. The molecules are composed of nucleotides which are made of three components: a 5-carbon sugar, a phosphate group and a nitrogenous base.

Norther, Southern and Western blot techniques: techniques used to identify unique proteins and nucleic acid sequences.

Gene 12s/MT-RNR1: acronym for Mitochondrially Encoded 12S RRNA, is an RNA Gene, which means that is not translated into a protein and are involved in other cellular processes.

Gene MT-CO2/COX2: acronym for Mitochondrially Encoded Cytochrome C Oxidase II, is a Protein Coding gene.

Gene MT-ATP6: Mitochondrially Encoded ATP Synthase Membrane Subunit 6, is a Protein Coding gene.

Amplicons: it is a piece of DNA or RNA that is the source and/or product of amplification or replication events.

Skewness: it is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean. The skewness value can be positive, zero, negative, or undefined.

Kurtostis: it is a measure of the "tailedness" of the probability distribution of a real-valued random variable.

Deterministic algorithm: in computer science, a deterministic algorithm is an algorithm which, given a particular input, will always produce the same output, with the underlying machine always passing through the same sequence of states.

Confidence interval: it is a type of estimate computed from the statistics of the observed data. This proposes a range of plausible values for an unknown parameter.

Probability distribution: is the mathematical function that gives the probabilities of occurrence of different possible outcomes for an experiment.

6 References

1. Alzaid A., & Sultan K. S. (2009). Discriminating between gamma and lognormal distributions with applications. Journal of King Saud University - Science, 21(2), 99–108. https://doi.org/10.1016/j.jksus.2009.07.003

2. Cadigan N., & Myers R. (2011). A comparison of Gamma and Lognormal maximum likelihood estimators in a sequential population analysis. Canadian Journal of Fisheries and Aquatic Sciences. 58(3): 560-567, https://doi.org/10.1139/f01-003

3. Chakraborti S., & Li J. (2007). Confidence Interval Estimation of a Normal Percentile. The American Statistician. 61. 331-336. https://www.jstor.org/stable/27643933

4. Cho, H.-K., Bowman, K. P., & North, G. R. (2004). A Comparison of Gamma and Lognormal Distributions for Characterizing Satellite Rain Rates from the Tropical Rainfall Measuring Mission. Journal of Applied Meteorology, 43(11), 1586–1597. https://doi.org/10.1175/JAM2165.1

5. Clèries R., & Galvez J., & Espino M., & Ribes J., & Nunes V., López de Heredia M. (2012). BootstRatio: A web-based statistical analysis of fold-change in qPCR and RT-qPCR data using resampling methods. Computers in Biology and Medicine. 42. 438-445. https://doi.org/10.1016/j.compbiomed.2011.12.012

6. Gamma distribution - Analytica Wiki. (2017). Analytica. https://wiki.analytica.com/index.php?title=Gamma_distribution

7. Harger, J. W., & Dinman, J. D. (2003). An in vivo dual-luciferase assay system for studying translational recoding in the yeast Saccharomyces cerevisiae. RNA (New York, N.Y.), 9(8), 1019–1024. https://doi.org/10.1261/rna.5930803

8. Jacobs, J. L., & Dinman, J. D. (2004). Systematic analysis of bicistronic reporter assay data. Nucleic acids research, 32(20), e160. https://doi.org/10.1093/nar/gnh157

9. Kuang J, & Yan X, Genders AJ, & Granata C, & Bishop D (2018). An overview of technical considerations when using quantitative real-time PCR analysis of gene expression in human exercise research. PLoS ONE. 13(5):e0196438. https://doi.org/10.1371/journal.pone.0196438

10. Kvist, T. & Thyregod, P. (2005). Using Evolutionary Operation to Improve Yield in Biotechnological Processes. Qual. Reliab. Engng. Int., 21: 457-463. https://doi.org/10.1002/qre.733

11. Lognormal mean and variance - MATLAB lognstat. (2018). MathWorks. https://www.mathworks.com/help/stats/lognstat.html#:%7E:text=The%20lognormal%20distribution%20is%20a,(%20%CF%83%202%20)%20%E2%88%92%201%20)

12. Moore, R. (2004). On Log-Likelihood-Ratios and the Significance of Rare Events. EMNLP. 333-340. http://www.aclweb.org/anthology/W04-3243

13. Morris T., & White I., & Crowther M. (2017). Using simulation studies to evaluate statistical methods. Statistics in Medicine. https://doi.org/10.1002/sim.8086

14. Peña D. (2014). *Fundamentos de Estadística.* Alianza Editorial.

15. Scott AJ., & Wu CFJ. (1981). On the asymptotic distribution of ratio and regression estimators. JASA 76: 90-102. https://www.jstor.org/stable/2287051

16. Viertl, R. (1987). *Probability and Bayesian Statistics.* Springer.

Appendix

Appendix 1: Table with coverage results for Bootstrap methodology

Table 4: Coverage results from the Bootstrap simulations

				Percentile Interval		BCa Interval	
N	Distribution	Mean	Variance	Coverage	Coverage Std	Coverage	Coverage Std
5	Gamma	0.9	0.05	83.41	7.29e-03	83.63	7.25e-03
			0.10	82.92	7.38e-03	83.43	7.29e-03
			0.15	83.98	7.19e-03	83.29	7.31e-03
			0.20	82.37	7.47e-03	82.87	7.38e-03
			0.25	82.56	7.44e-03	81.64	7.59e-03
			0.30	81.54	7.60e-03	82.59	7.43e-03
		1.0	0.05	83.51	7.27e-03	83.74	7.23e-03
			0.10	83.87	7.21e-03	83.44	7.29e-03
			0.15	82.54	7.44e-03	82.95	7.37e-03
			0.20	81.88	7.55e-03	82.40	7.46e-03
			0.25	81.79	7.56e-03	82.95	7.37e-03
			0.30	81.96	7.54e-03	82.14	7.51e-03
		1.1	0.05	83.58	7.26e-03	83.40	7.29e-03
			0.10	83.48	7.28e-03	82.72	7.41e-03
			0.15	82.71	7.41e-03	83.12	7.34e-03
			0.20	81.89	7.55e-03	83.33	7.30e-03
			0.25	81.77	7.57e-03	82.94	7.37e-03
			0.30	82.73	7.41e-03	82.66	7.42e-03
	Log-Normal	0.9	0.05	82.44	7.46e-03	83.10	7.35e-03
			0.10	82.02	7.53e-03	83.13	7.34e-03
			0.15	81.38	7.63e-03	81.83	7.56e-03
			0.20	80.66	7.74e-03	81.58	7.60e-03
			0.25	80.56	7.76e-03	81.07	7.68e-03
			0.30	79.99	7.84e-03	80.34	7.79e-03
		1.0	0.05	83.21	7.33e-03	82.58	7.43e-03
			0.10	82.96	7.37e-03	83.26	7.32e-03
			0.15	81.73	7.61e-03	81.97	7.53e-03
			0.20	81.59	7.60e-03	81.82	7.56e-03
			0.25	81.42	7.62e-03	81.47	7.62e-03
			0.30	80.31	7.86e-03	80.74	7.73e-03
		1.1	0.05	83.32	7.31e-03	83.46	7.28e-03
			0.10	82.86	7.39e-03	82.85	7.39e-03
			0.15	82.11	7.53e-03	82.47	7.45e-03
			0.20	81.82	7.58e-03	82.59	7.43e-03
			0.25	81.71	7.53e-03	81.54	7.60e-03
			0.30	80.40	7.78e-03	82.22	7.49e-03
10	Gamma	0.9	0.05	89.59	5.99e-03	89.45	6.02e-03
			0.10	89.57	5.99e-03	89.70	5.96e-03
			0.15	89.00	6.13e-03	89.57	5.99e-03

Table 4: Coverage results from the Bootstrap simulations *(continued)*

N	Distribution	Mean	Variance	Coverage	Coverage Std	Coverage	Coverage Std
			0.20	89.12	6.10e-03	89.11	6.11e-03
			0.25	88.25	6.31e-03	89.16	6.09e-03
			0.30	88.33	6.29e-03	88.47	6.26e-03
		1.0	0.05	90.15	5.84e-03	89.65	5.97e-03
			0.10	89.65	5.97e-03	89.86	5.92e-03
			0.15	89.70	5.96e-03	90.08	5.86e-03
			0.20	88.92	6.15e-03	89.60	5.98e-03
			0.25	89.22	6.08e-03	89.29	6.06e-03
			0.30	88.76	6.19e-03	89.36	6.04e-03
		1.1	0.05	89.87	5.91e-03	89.91	5.90e-03
			0.10	89.54	6.00e-03	89.93	5.90e-03
			0.15	89.75	5.94e-03	89.84	5.92e-03
			0.20	89.45	6.02e-03	89.85	5.92e-03
			0.25	89.62	5.98e-03	89.03	6.13e-03
			0.30	88.84	6.17e-03	88.84	6.17e-03
	Log-Normal	0.9	0.05	89.56	5.99e-03	89.68	5.96e-03
			0.10	88.42	6.27e-03	89.35	6.05e-03
			0.15	88.70	6.21e-03	88.43	6.27e-03
			0.20	87.54	6.47e-03	88.56	6.24e-03
			0.25	87.18	6.55e-03	87.71	6.44e-03
			0.30	85.96	6.81e-03	87.93	6.39e-03
		1.0	0.05	89.91	5.90e-03	89.49	6.01e-03
			0.10	88.83	6.17e-03	89.00	6.13e-03
			0.15	88.70	6.21e-03	89.22	6.08e-03
			0.20	88.71	6.20e-03	88.21	6.32e-03
			0.25	87.46	6.49e-03	88.54	6.24e-03
			0.30	87.39	6.51e-03	87.38	6.51e-03
		1.1	0.05	88.96	6.14e-03	89.71	5.95e-03
			0.10	89.23	6.08e-03	88.89	6.16e-03
			0.15	88.65	6.22e-03	89.25	6.07e-03
			0.20	88.58	6.23e-03	88.74	6.20e-03
			0.25	87.73	6.43e-03	87.98	6.37e-03
			0.30	87.78	6.42e-03	88.72	6.20e-03
10	Gamma	0.9	0.05	92.70	5.10e-03	91.88	5.35e-03
			0.10	92.49	5.17e-03	92.87	5.04e-03
			0.15	92.22	5.25e-03	92.14	5.27e-03
			0.20	92.17	5.27e-03	91.72	5.40e-03
			0.25	91.62	5.43e-03	91.98	5.32e-03
			0.30	91.91	5.34e-03	92.01	5.31e-03
		1.0	0.05	92.13	5.28e-03	92.13	5.28e-03
			0.10	92.46	5.18e-03	92.23	5.25e-03
			0.15	92.25	5.24e-03	92.22	5.25e-03

Table 4: Coverage results from the Bootstrap simulations *(continued)*

N	Distribution	Mean	Variance	Coverage	Coverage Std	Coverage	Coverage Std
			0.20	92.41	5.19e-03	92.31	5.22e-03
			0.25	92.13	5.28e-03	92.16	5.27e-03
			0.30	92.28	5.23e-03	92.15	5.27e-03
		1.1	0.05	92.95	5.02e-03	92.83	5.06e-03
			0.10	92.68	5.11e-03	92.39	5.20e-03
			0.15	92.38	5.20e-03	91.96	5.33e-03
			0.20	91.99	5.32e-03	92.57	5.14e-03
			0.25	92.09	5.29e-03	91.76	5.39e-03
			0.30	91.85	5.36e-03	91.80	5.38e-03
	Log-Normal	0.9	0.05	91.80	5.38e-03	92.05	5.30e-03
			0.10	91.85	5.36e-03	91.96	5.33e-03
			0.15	91.63	5.43e-03	91.56	5.45e-03
			0.20	91.29	5.53e-03	91.13	5.57e-03
			0.25	90.53	5.74e-03	91.23	5.54e-03
			0.30	90.39	5.78e-03	90.91	5.63e-03
		1.0	0.05	92.17	5.27e-03	91.98	5.32e-03
			0.10	91.55	5.45e-03	91.43	5.49e-03
			0.15	91.36	5.51e-03	91.84	5.37e-03
			0.20	91.17	5.56e-03	91.91	5.34e-03
			0.25	91.20	5.55e-03	90.34	5.79e-03
			0.30	90.38	5.78e-03	91.39	5.50e-03
		1.1	0.05	92.26	5.24e-03	92.28	5.23e-03
			0.10	91.89	5.35e-03	91.78	5.38e-03
			0.15	91.54	5.45e-03	91.75	5.39e-03
			0.20	91.28	5.53e-03	91.85	5.36e-03
			0.25	91.05	5.59e-03	91.41	5.49e-03
			0.30	91.04	5.60e-03	91.58	5.44e-03
10	Gamma	0.9	0.05	93.49	4.84e-03	93.13	4.96e-03
			0.10	93.15	4.95e-03	93.14	4.95e-03
			0.15	92.73	5.09e-03	93.27	4.91e-03
			0.20	92.88	5.04e-03	93.29	4.90e-03
			0.25	93.35	4.88e-03	93.19	4.94e-03
			0.30	92.56	5.14e-03	92.95	5.02e-03
		1.0	0.05	93.44	4.85e-03	93.58	4.80e-03
			0.10	93.51	4.83e-03	92.75	5.08e-03
			0.15	93.39	4.87e-03	93.39	4.87e-03
			0.20	93.00	5.00e-03	92.99	5.00e-03
			0.25	92.87	5.04e-03	92.78	5.07e-03
			0.30	93.19	4.94e-03	93.46	4.85e-03
		1.1	0.05	93.07	4.98e-03	93.45	4.85e-03
			0.10	92.99	5.00e-03	93.37	4.88e-03
			0.15	93.66	4.78e-03	93.25	4.92e-03

Table 4: Coverage results from the Bootstrap simulations *(continued)*

N	Distribution	Mean	Variance	Coverage	Coverage Std	Coverage	Coverage Std
			0.20	93.40	4.87e-03	93.38	4.87e-03
			0.25	92.74	5.09e-03	93.29	4.90e-03
			0.30	93.28	4.91e-03	92.76	5.08e-03
	Log-Normal	0.9	0.05	93.18	4.94e-03	93.22	4.93e-03
			0.10	92.99	5.00e-03	92.71	5.10e-03
			0.15	92.02	5.31e-03	92.16	5.27e-03
			0.20	92.32	5.22e-03	92.61	5.13e-03
			0.25	92.27	5.23e-03	92.14	5.27e-03
			0.30	91.82	5.37e-03	92.08	5.29e-03
		1.0	0.05	93.33	4.89e-03	92.95	5.02e-03
			0.10	92.71	5.10e-03	92.80	5.07e-03
			0.15	92.77	5.08e-03	92.73	5.09e-03
			0.20	92.21	5.25e-03	92.74	5.09e-03
			0.25	92.09	5.29e-03	92.74	5.09e-03
			0.30	91.91	5.34e-03	92.08	5.29e-03
		1.1	0.05	92.84	5.05e-03	93.29	4.90e-03
			0.10	92.76	5.08e-03	92.95	5.02e-03
			0.15	92.70	5.10e-03	92.63	5.12e-03
			0.20	92.33	5.22e-03	92.57	5.14e-03
			0.25	92.43	5.18e-03	92.41	5.19e-03
			0.30	92.24	5.24e-03	91.98	5.32e-03

Appendix 2: Table with coverage results for Bayesian methodology

Table 5: Coverage results from the Bayesian simulations

				Confidence Interval	
N	Distribution	Mean	Variance	Coverage	Coverage Std
5	Gamma	0.9	0.05	90.99	2.14e-02
			0.10	89.99	2.25e-02
			0.15	89.46	2.29e-02
			0.20	88.67	2.29e-02
			0.25	88.03	2.18e-02
			0.30	87.86	2.27e-02
		1.0	0.05	91.09	1.84e-02
			0.10	89.79	2.08e-02
			0.15	89.56	2.17e-02
			0.20	88.97	2.38e-02
			0.25	88.53	2.11e-02
			0.30	88.40	2.36e-02
		1.1	0.05	91.04	2.03e-02
			0.10	90.78	2.06e-02
			0.15	88.93	1.90e-02
			0.20	88.93	2.23e-02
			0.25	88.35	2.33e-02
			0.30	87.78	2.36e-02
5	Log-Normal	0.9	0.05	91.30	2.17e-02
			0.10	89.86	2.27e-02
			0.15	88.94	2.29e-02
			0.20	87.27	2.15e-02
			0.25	86.79	2.29e-02
			0.30	86.03	2.42e-02
		1.0	0.05	90.59	2.00e-02
			0.10	89.20	2.21e-02
			0.15	88.41	2.28e-02
			0.20	87.35	2.36e-02
			0.25	86.96	2.47e-02
			0.30	86.03	2.40e-02
		1.1	0.05	90.51	2.08e-02
			0.10	89.33	2.20e-02
			0.15	88.80	2.24e-02
			0.20	87.35	2.36e-02
			0.25	87.48	2.35e-02
			0.30	86.72	2.35e-02
10	Gamma	0.9	0.05	91.57	1.98e-02
			0.10	92.75	1.84e-02
			0.15	92.23	1.90e-02
			0.20	92.75	1.84e-02
			0.25	92.23	1.90e-02

Table 5: Coverage results from the Bayesian simulations *(continued)*

N	Distribution	Mean	Variance	Coverage	Coverage Std
			0.30	91.70	1.96e-02
		1.0	0.05	92.75	1.84e-02
			0.10	92.75	1.84e-02
			0.15	93.02	1.81e-02
			0.20	91.57	1.98e-02
			0.25	91.04	2.03e-02
			0.30	92.49	1.87e-02
		1.1	0.05	93.02	1.81e-02
			0.10	93.41	1.76e-02
			0.15	91.44	1.99e-02
			0.20	93.41	1.76e-02
			0.25	90.25	2.11e-02
			0.30	91.44	1.99e-02
10	Log-Normal	0.9	0.05	93.41	1.76e-02
			0.10	92.09	2.10e-02
			0.15	91.70	1.96e-02
			0.20	90.38	1.92e-02
			0.25	90.12	2.17e-02
			0.30	89.59	2.12e-02
		1.0	0.05	93.15	1.80e-02
			0.10	92.14	1.99e-02
			0.15	91.86	1.80e-02
			0.20	90.52	1.92e-02
			0.25	89.72	2.21e-02
			0.30	89.32	1.92e-02
		1.1	0.05	93.02	1.81e-02
			0.10	92.54	1.75e-02
			0.15	92.23	1.90e-02
			0.20	91.90	1.87e-02
			0.25	90.51	2.08e-02
			0.30	89.68	2.10e-02
20	Gamma	0.9	0.05	94.60	1.73e-02
			0.10	93.98	1.83e-02
			0.15	93.59	1.61e-02
			0.20	93.02	1.81e-02
			0.25	92.89	1.96e-02
			0.30	91.70	1.83e-02
		1.0	0.05	94.43	1.90e-02
			0.10	93.71	1.76e-02
			0.15	93.53	1.73e-02
			0.20	93.14	1.70e-02
			0.25	92.73	1.59e-02

Table 5: Coverage results from the Bayesian simulations *(continued)*

N	Distribution	Mean	Variance	Coverage	Coverage Std
			0.30	91.93	1.59e-02
		1.1	0.05	94.20	1.66e-02
			0.10	93.82	1.92e-02
			0.15	93.32	1.81e-02
			0.20	93.20	1.66e-02
			0.25	92.68	1.73e-02
			0.30	91.68	1.73e-02
20	Log-Normal	0.9	0.05	94.33	1.64e-02
			0.10	93.75	2.02e-02
			0.15	92.66	2.05e-02
			0.20	91.17	1.84e-02
			0.25	90.91	2.21e-02
			0.30	89.60	2.02e-02
		1.0	0.05	94.79	1.55e-02
			0.10	93.84	1.75e-02
			0.15	93.35	1.84e-02
			0.20	92.89	1.86e-02
			0.25	92.36	1.83e-02
			0.30	91.02	1.86e-02
		1.1	0.05	94.54	1.84e-02
			0.10	93.41	1.76e-02
			0.15	93.15	1.80e-02
			0.20	92.75	1.84e-02
			0.25	91.73	2.02e-02
			0.30	89.72	2.16e-02
30	Gamma	0.9	0.05	95.05	1.45e-02
			0.10	94.73	1.59e-02
			0.15	94.47	1.81e-02
			0.20	94.13	1.80e-02
			0.25	93.15	1.64e-02
			0.30	93.02	1.63e-02
		1.0	0.05	94.81	1.71e-02
			0.10	94.60	1.61e-02
			0.15	94.05	1.39e-02
			0.20	93.70	1.66e-02
			0.25	93.39	1.49e-02
			0.30	93.01	1.71e-02
		1.1	0.05	94.95	1.80e-02
			0.10	94.67	1.68e-02
			0.15	94.33	1.64e-02
			0.20	94.04	2.03e-02
			0.25	93.62	1.86e-02

Table 5: Coverage results from the Bayesian simulations *(continued)*

N	Distribution	Mean	Variance	Coverage	Coverage Std
			0.30	93.26	1.57e-02
30	Log-Normal	0.9	0.05	95.05	1.66e-02
			0.10	94.47	1.63e-02
			0.15	93.44	1.99e-02
			0.20	92.49	2.00e-02
			0.25	91.30	1.87e-02
			0.30	90.12	2.12e-02
		1.0	0.05	95.26	1.51e-02
			0.10	94.41	1.81e-02
			0.15	93.41	1.76e-02
			0.20	93.02	1.76e-02
			0.25	91.17	2.02e-02
			0.30	91.83	1.95e-02
		1.1	0.05	94.75	1.84e-02
			0.10	94.47	1.63e-02
			0.15	93.28	1.78e-02
			0.20	92.49	1.87e-02
			0.25	91.44	1.99e-02
			0.30	91.17	2.02e-02

Appendix 3: Table with length results for Bootstrap methodology

Table 6: Length results from the Bootstrap simulations

				Percentile Interval		BCa Interval	
N	Distribution	Mean	Variance	Length	Length Std	Length	Length Std
5	Gamma	0.9	0.05	0.322	2.39e-03	0.335	2.57e-03
			0.10	0.452	3.50e-03	0.473	3.83e-03
			0.15	0.550	4.37e-03	0.577	4.90e-03
			0.20	0.627	5.21e-03	0.662	5.84e-03
			0.25	0.699	6.03e-03	0.740	6.85e-03
			0.30	0.763	6.74e-03	0.808	7.66e-03
		1.0	0.05	0.322	2.37e-03	0.334	2.52e-03
			0.10	0.454	3.45e-03	0.473	3.77e-03
			0.15	0.552	4.30e-03	0.575	4.74e-03
			0.20	0.634	5.16e-03	0.661	5.63e-03
			0.25	0.707	5.94e-03	0.744	6.55e-03
			0.30	0.764	6.50e-03	0.814	7.37e-03
		1.1	0.05	0.321	2.35e-03	0.334	2.54e-03
			0.10	0.455	3.39e-03	0.471	3.68e-03
			0.15	0.552	4.29e-03	0.577	4.70e-03
			0.20	0.633	5.02e-03	0.670	5.56e-03
			0.25	0.705	5.69e-03	0.742	6.32e-03
			0.30	0.776	6.38e-03	0.817	7.29e-03
5	Gamma	0.9	0.05	0.317	2.53e-03	0.334	2.79e-03
			0.10	0.443	3.82e-03	0.465	4.20e-03
			0.15	0.534	5.04e-03	0.565	5.58e-03
			0.20	0.606	5.88e-03	0.649	6.91e-03
			0.25	0.675	7.06e-03	0.718	8.16e-03
			0.30	0.723	7.84e-03	0.786	9.75e-03
		1.0	0.05	0.320	2.49e-03	0.333	2.70e-03
			0.10	0.442	3.67e-03	0.470	4.24e-03
			0.15	0.534	4.80e-03	0.566	5.42e-03
			0.20	0.620	5.84e-03	0.651	6.56e-03
			0.25	0.682	6.80e-03	0.722	7.77e-03
			0.30	0.735	7.60e-03	0.784	8.75e-03
		1.1	0.05	0.320	2.46e-03	0.332	2.62e-03
			0.10	0.449	3.63e-03	0.469	4.03e-03
			0.15	0.543	4.71e-03	0.567	5.28e-03
			0.20	0.615	5.53e-03	0.658	6.36e-03
			0.25	0.689	6.48e-03	0.729	7.57e-03
			0.30	0.747	7.38e-03	0.787	8.23e-03
5	Gamma	0.9	0.05	0.252	1.27e-03	0.258	1.33e-03
			0.10	0.356	1.87e-03	0.364	1.99e-03
			0.15	0.435	2.42e-03	0.447	2.64e-03
			0.20	0.499	2.89e-03	0.516	3.16e-03
			0.25	0.556	3.36e-03	0.576	3.74e-03

Table 6: Length results from the Bootstrap simulations *(continued)*

N	Distribution	Mean	Variance	Length	Length Std	Length	Length Std
			0.30	0.605	3.76e-03	0.626	4.20e-03
		1.0	0.05	0.252	1.25e-03	0.259	1.31e-03
			0.10	0.356	1.82e-03	0.365	1.96e-03
			0.15	0.436	2.32e-03	0.445	2.49e-03
			0.20	0.498	2.77e-03	0.517	3.05e-03
			0.25	0.557	3.20e-03	0.578	3.59e-03
			0.30	0.612	3.66e-03	0.631	4.07e-03
		1.1	0.05	0.254	1.24e-03	0.258	1.31e-03
			0.10	0.357	1.82e-03	0.364	1.91e-03
			0.15	0.436	2.29e-03	0.447	2.45e-03
			0.20	0.500	2.69e-03	0.516	2.98e-03
			0.25	0.558	3.12e-03	0.575	3.42e-03
			0.30	0.609	3.50e-03	0.631	3.89e-03
5	Gamma	0.9	0.05	0.251	1.37e-03	0.258	1.47e-03
			0.10	0.351	2.17e-03	0.362	2.37e-03
			0.15	0.427	2.80e-03	0.445	3.36e-03
			0.20	0.488	3.48e-03	0.510	4.14e-03
			0.25	0.538	4.15e-03	0.574	4.87e-03
			0.30	0.579	4.68e-03	0.623	5.75e-03
		1.0	0.05	0.251	1.33e-03	0.257	1.44e-03
			0.10	0.353	2.08e-03	0.363	2.27e-03
			0.15	0.428	2.68e-03	0.448	3.08e-03
			0.20	0.491	3.33e-03	0.512	3.86e-03
			0.25	0.542	3.86e-03	0.573	4.54e-03
			0.30	0.590	4.42e-03	0.622	5.20e-03
		1.1	0.05	0.252	1.33e-03	0.258	1.39e-03
			0.10	0.355	2.01e-03	0.363	2.18e-03
			0.15	0.429	2.58e-03	0.446	2.87e-03
			0.20	0.493	3.13e-03	0.514	3.63e-03
			0.25	0.547	3.74e-03	0.573	4.40e-03
			0.30	0.594	4.22e-03	0.627	5.00e-03
5	Gamma	0.9	0.05	0.187	6.52e-04	0.189	6.80e-04
			0.10	0.264	9.77e-04	0.267	1.02e-03
			0.15	0.322	1.26e-03	0.328	1.36e-03
			0.20	0.371	1.52e-03	0.380	1.67e-03
			0.25	0.413	1.76e-03	0.424	1.94e-03
			0.30	0.453	1.99e-03	0.464	2.20e-03
		1.0	0.05	0.187	6.48e-04	0.189	6.52e-04
			0.10	0.265	9.52e-04	0.267	1.01e-03
			0.15	0.322	1.21e-03	0.328	1.29e-03
			0.20	0.371	1.45e-03	0.378	1.59e-03
			0.25	0.415	1.68e-03	0.422	1.82e-03

Table 6: Length results from the Bootstrap simulations *(continued)*

N	Distribution	Mean	Variance	Length	Length Std	Length	Length Std
			0.30	0.454	1.89e-03	0.465	2.08e-03
		1.1	0.05	0.187	6.43e-04	0.189	6.53e-04
			0.10	0.264	9.31e-04	0.268	9.86e-04
			0.15	0.324	1.18e-03	0.326	1.27e-03
			0.20	0.372	1.41e-03	0.380	1.54e-03
			0.25	0.415	1.64e-03	0.423	1.77e-03
			0.30	0.455	1.83e-03	0.464	2.01e-03
5	Gamma	0.9	0.05	0.187	7.15e-04	0.189	7.66e-04
			0.10	0.261	1.12e-03	0.268	1.28e-03
			0.15	0.317	1.52e-03	0.329	1.76e-03
			0.20	0.366	1.90e-03	0.379	2.26e-03
			0.25	0.407	2.31e-03	0.426	2.74e-03
			0.30	0.443	2.69e-03	0.464	3.19e-03
		1.0	0.05	0.187	7.00e-04	0.189	7.43e-04
			0.10	0.263	1.10e-03	0.267	1.20e-03
			0.15	0.319	1.48e-03	0.329	1.65e-03
			0.20	0.367	1.82e-03	0.381	2.09e-03
			0.25	0.410	2.18e-03	0.425	2.53e-03
			0.30	0.445	2.57e-03	0.466	2.97e-03
		1.1	0.05	0.187	6.86e-04	0.189	7.08e-04
			0.10	0.263	1.06e-03	0.268	1.14e-03
			0.15	0.321	1.40e-03	0.328	1.57e-03
			0.20	0.369	1.73e-03	0.382	1.95e-03
			0.25	0.407	2.01e-03	0.426	2.37e-03
			0.30	0.447	2.32e-03	0.464	2.74e-03
5	Gamma	0.9	0.05	0.155	4.43e-04	0.156	4.47e-04
			0.10	0.219	6.55e-04	0.221	6.87e-04
			0.15	0.267	8.62e-04	0.271	9.02e-04
			0.20	0.309	1.04e-03	0.312	1.10e-03
			0.25	0.344	1.20e-03	0.349	1.31e-03
			0.30	0.378	1.39e-03	0.384	1.50e-03
		1.0	0.05	0.155	4.41e-04	0.156	4.49e-04
			0.10	0.219	6.55e-04	0.220	6.75e-04
			0.15	0.268	8.28e-04	0.270	8.63e-04
			0.20	0.308	9.93e-04	0.312	1.05e-03
			0.25	0.344	1.14e-03	0.350	1.25e-03
			0.30	0.377	1.31e-03	0.384	1.42e-03
		1.1	0.05	0.155	4.32e-04	0.156	4.39e-04
			0.10	0.219	6.23e-04	0.221	6.57e-04
			0.15	0.268	8.10e-04	0.270	8.40e-04
			0.20	0.309	9.70e-04	0.311	1.01e-03
			0.25	0.345	1.12e-03	0.350	1.19e-03

Table 6: Length results from the Bootstrap simulations *(continued)*

N	Distribution	Mean	Variance	Length	Length Std	Length	Length Std
			0.30	0.378	1.26e-03	0.384	1.36e-03
5	Gamma	0.9	0.05	0.155	4.85e-04	0.156	5.18e-04
			0.10	0.218	7.88e-04	0.221	8.55e-04
			0.15	0.265	1.06e-03	0.272	1.20e-03
			0.20	0.304	1.31e-03	0.314	1.55e-03
			0.25	0.339	1.60e-03	0.353	1.92e-03
			0.30	0.371	1.88e-03	0.383	2.15e-03
		1.0	0.05	0.155	4.78e-04	0.156	4.91e-04
			0.10	0.217	7.39e-04	0.220	7.85e-04
			0.15	0.266	1.00e-03	0.270	1.10e-03
			0.20	0.306	1.26e-03	0.314	1.45e-03
			0.25	0.341	1.51e-03	0.351	1.72e-03
			0.30	0.372	1.75e-03	0.384	2.05e-03
		1.1	0.05	0.155	4.57e-04	0.156	4.82e-04
			0.10	0.218	7.26e-04	0.220	7.53e-04
			0.15	0.266	9.62e-04	0.272	1.05e-03
			0.20	0.306	1.19e-03	0.314	1.35e-03
			0.25	0.342	1.42e-03	0.349	1.61e-03
			0.30	0.372	1.63e-03	0.385	1.89e-03

Appendix 4: Table with length results for Bayesian methodology

Table 7: Length results from the Bayesian simulations

				Confidence Interval	
N	Distribution	Mean	Variance	Length	Length Std
5	Gamma	0.9	0.05	0.404	1.01e-02
			0.10	0.562	1.61e-02
			0.15	0.696	2.09e-02
			0.20	0.803	2.50e-02
			0.25	0.871	2.92e-02
			0.30	1.001	3.48e-02
		1.0	0.05	0.406	9.96e-03
			0.10	0.556	1.45e-02
			0.15	0.687	1.93e-02
			0.20	0.786	2.32e-02
			0.25	0.895	2.68e-02
			0.30	0.976	3.29e-02
		1.1	0.05	0.398	9.18e-03
			0.10	0.555	1.45e-02
			0.15	0.684	1.91e-02
			0.20	0.778	2.27e-02
			0.25	0.866	2.58e-02
			0.30	0.959	3.13e-02
5	Log-Normal	0.9	0.05	0.395	1.03e-02
			0.10	0.539	1.63e-02
			0.15	0.661	2.10e-02
			0.20	0.735	2.54e-02
			0.25	0.810	2.80e-02
			0.30	0.917	3.67e-02
		1.0	0.05	0.401	9.79e-03
			0.10	0.556	1.67e-02
			0.15	0.664	2.10e-02
			0.20	0.754	2.54e-02
			0.25	0.861	3.12e-02
			0.30	0.885	3.15e-02
		1.1	0.05	0.406	9.90e-03
			0.10	0.545	1.58e-02
			0.15	0.673	1.99e-02
			0.20	0.770	2.52e-02
			0.25	0.818	2.78e-02
			0.30	0.935	3.44e-02
10	Gamma	0.9	0.05	0.278	4.93e-03
			0.10	0.393	7.14e-03
			0.15	0.489	9.84e-03
			0.20	0.562	1.12e-02
			0.25	0.615	1.28e-02

Table 7: Length results from the Bayesian simulations *(continued)*

N	Distribution	Mean	Variance	Length	Length Std
			0.30	0.698	1.59e-02
		1.0	0.05	0.283	4.77e-03
			0.10	0.398	6.90e-03
			0.15	0.492	9.51e-03
			0.20	0.559	1.16e-02
			0.25	0.637	1.31e-02
			0.30	0.692	1.46e-02
		1.1	0.05	0.286	4.67e-03
			0.10	0.395	7.40e-03
			0.15	0.487	9.24e-03
			0.20	0.571	1.15e-02
			0.25	0.613	1.30e-02
			0.30	0.672	1.40e-02
10	Log-Normal	0.9	0.05	0.281	5.03e-03
			0.10	0.375	7.59e-03
			0.15	0.465	1.03e-02
			0.20	0.535	1.22e-02
			0.25	0.590	1.55e-02
			0.30	0.649	1.72e-02
		1.0	0.05	0.277	4.96e-03
			0.10	0.388	7.64e-03
			0.15	0.470	1.01e-02
			0.20	0.546	1.25e-02
			0.25	0.587	1.33e-02
			0.30	0.622	1.57e-02
		1.1	0.05	0.280	4.52e-03
			0.10	0.383	7.38e-03
			0.15	0.468	9.25e-03
			0.20	0.548	1.11e-02
			0.25	0.586	1.30e-02
			0.30	0.643	1.45e-02
20	Gamma	0.9	0.05	0.194	2.41e-03
			0.10	0.277	3.55e-03
			0.15	0.340	4.62e-03
			0.20	0.391	5.66e-03
			0.25	0.437	6.79e-03
			0.30	0.480	7.58e-03
		1.0	0.05	0.195	2.41e-03
			0.10	0.281	3.48e-03
			0.15	0.336	4.51e-03
			0.20	0.387	5.13e-03
			0.25	0.443	5.95e-03

Table 7: Length results from the Bayesian simulations *(continued)*

N	Distribution	Mean	Variance	Length	Length Std
			0.30	0.484	6.73e-03
		1.1	0.05	0.195	2.26e-03
			0.10	0.278	3.63e-03
			0.15	0.331	4.10e-03
			0.20	0.388	5.39e-03
			0.25	0.436	5.99e-03
			0.30	0.473	6.38e-03
20	Log-Normal	0.9	0.05	0.191	2.43e-03
			0.10	0.268	3.77e-03
			0.15	0.333	5.12e-03
			0.20	0.372	5.88e-03
			0.25	0.403	7.10e-03
			0.30	0.441	7.76e-03
		1.0	0.05	0.194	2.28e-03
			0.10	0.268	3.82e-03
			0.15	0.324	4.59e-03
			0.20	0.376	5.79e-03
			0.25	0.413	6.60e-03
			0.30	0.436	6.73e-03
		1.1	0.05	0.195	2.43e-03
			0.10	0.268	3.69e-03
			0.15	0.330	4.66e-03
			0.20	0.375	5.64e-03
			0.25	0.411	6.36e-03
			0.30	0.443	7.40e-03
30	Gamma	0.9	0.05	0.160	1.68e-03
			0.10	0.225	2.45e-03
			0.15	0.272	2.99e-03
			0.20	0.319	3.64e-03
			0.25	0.361	4.34e-03
			0.30	0.384	4.52e-03
		1.0	0.05	0.159	1.60e-03
			0.10	0.227	2.50e-03
			0.15	0.273	2.86e-03
			0.20	0.320	3.42e-03
			0.25	0.353	3.92e-03
			0.30	0.386	4.80e-03
		1.1	0.05	0.158	1.54e-03
			0.10	0.227	2.45e-03
			0.15	0.280	2.90e-03
			0.20	0.317	3.48e-03
			0.25	0.353	3.89e-03

Table 7: Length results from the Bayesian simulations *(continued)*

N	Distribution	Mean	Variance	Length	Length Std
			0.30	0.398	4.56e-03
30	Log-Normal	0.9	0.05	0.156	1.62e-03
			0.10	0.217	2.51e-03
			0.15	0.261	3.20e-03
			0.20	0.302	4.15e-03
			0.25	0.330	4.46e-03
			0.30	0.358	5.03e-03
		1.0	0.05	0.156	1.55e-03
			0.10	0.219	2.38e-03
			0.15	0.265	3.02e-03
			0.20	0.300	3.57e-03
			0.25	0.335	4.35e-03
			0.30	0.362	5.05e-03
		1.1	0.05	0.160	1.68e-03
			0.10	0.220	2.46e-03
			0.15	0.269	3.17e-03
			0.20	0.303	3.63e-03
			0.25	0.342	4.23e-03
			0.30	0.355	4.66e-03

Appendix 5: Code used to run the simulation study

```
m <- c(0.9, 1, 1.1)
v <- c(0.05, 0.1, 0.15, 0.2, 0.25, 0.3)
```

Gamma Distribution

Small Sample Size $N = 5$

```
boot5_perc_cov_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_perc_cov_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_perc_leng_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_perc_leng_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_bca_cov_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_bca_cov_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_bca_leng_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_bca_leng_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        shape <- m[i]^2/v[j]
        rate <- m[i]/v[j]
        resultats_boot5_g_perc <- simScenarioGamma(shape,
            rate, 5, 10000, z0 = 0, a = 0, method = "boot")
        boot5_perc_cov_g <- rbind(boot5_perc_cov_g,
            list(value = resultats_boot5_g_perc[1],
                mean = m[i], variance = v[j]))
        boot5_perc_cov_sd_g <- rbind(boot5_perc_cov_sd_g,
            list(value = resultats_boot5_g_perc[2],
                mean = m[i], variance = v[j]))
        boot5_perc_leng_g <- rbind(boot5_perc_leng_g,
            list(value = resultats_boot5_g_perc[3],
                mean = m[i], variance = v[j]))
        boot5_perc_leng_sd_g <- rbind(boot5_perc_leng_sd_g,
            list(value = resultats_boot5_g_perc[4],
                mean = m[i], variance = v[j]))
        resultats_boot5_g_bca <- simScenarioGamma(shape,
            rate, 5, 10000, method = "boot")
        boot5_bca_cov_g <- rbind(boot5_bca_cov_g, list(value = resultats_boot5_g_bca[1],
            mean = m[i], variance = v[j]))
        boot5_bca_cov_sd_g <- rbind(boot5_bca_cov_sd_g,
            list(value = resultats_boot5_g_bca[2],
```

```
                mean = m[i], variance = v[j]))
        boot5_bca_leng_g <- rbind(boot5_bca_leng_g,
            list(value = resultats_boot5_g_bca[3],
                mean = m[i], variance = v[j]))
        boot5_bca_leng_sd_g <- rbind(boot5_bca_leng_sd_g,
            list(value = resultats_boot5_g_bca[4],
                mean = m[i], variance = v[j]))
    }
}

bay5_perc_cov_g <- data.frame(value = c(), mean = c(),
    variance = c())
bay5_perc_cov_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
bay5_perc_leng_g <- data.frame(value = c(), mean = c(),
    variance = c())
bay5_perc_leng_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        shape <- m[i]^2/v[j]
        rate <- m[i]/v[j]
        resultats_bay5_g_perc <- simScenarioGamma(shape,
            rate, n = 5, method = "bayesian")
        bay5_perc_cov_g <- rbind(bay5_perc_cov_g, list(value = resultats_bay5_g_perc[1],
            mean = m[i], variance = v[j]))
        bay5_perc_cov_sd_g <- rbind(bay5_perc_cov_sd_g,
            list(value = resultats_bay5_g_perc[2],
                mean = m[i], variance = v[j]))
        bay5_perc_leng_g <- rbind(bay5_perc_leng_g,
            list(value = resultats_bay5_g_perc[3],
                mean = m[i], variance = v[j]))
        bay5_perc_leng_sd_g <- rbind(bay5_perc_leng_sd_g,
            list(value = resultats_bay5_g_perc[4],
                mean = m[i], variance = v[j]))
    }
}
```

Small Sample Size $N = 10$

```
boot10_perc_cov_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot10_perc_cov_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot10_perc_leng_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot10_perc_leng_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
```

```
boot10_bca_cov_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot10_bca_cov_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot10_bca_leng_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot10_bca_leng_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        shape <- m[i]^2/v[j]
        rate <- m[i]/v[j]
        resultats_boot10_g_perc <- simScenarioGamma(shape,
            rate, 10, 10000, z0 = 0, a = 0, method = "boot")
        boot10_perc_cov_g <- rbind(boot10_perc_cov_g,
            list(value = resultats_boot10_g_perc[1],
                mean = m[i], variance = v[j]))
        boot10_perc_cov_sd_g <- rbind(boot10_perc_cov_sd_g,
            list(value = resultats_boot10_g_perc[2],
                mean = m[i], variance = v[j]))
        boot10_perc_leng_g <- rbind(boot10_perc_leng_g,
            list(value = resultats_boot10_g_perc[3],
                mean = m[i], variance = v[j]))
        boot10_perc_leng_sd_g <- rbind(boot10_perc_leng_sd_g,
            list(value = resultats_boot10_g_perc[4],
                mean = m[i], variance = v[j]))
        resultats_boot10_g_bca <- simScenarioGamma(shape,
            rate, 10, 10000, method = "boot")
        boot10_bca_cov_g <- rbind(boot10_bca_cov_g,
            list(value = resultats_boot10_g_bca[1],
                mean = m[i], variance = v[j]))
        boot10_bca_cov_sd_g <- rbind(boot10_bca_cov_sd_g,
            list(value = resultats_boot10_g_bca[2],
                mean = m[i], variance = v[j]))
        boot10_bca_leng_g <- rbind(boot10_bca_leng_g,
            list(value = resultats_boot10_g_bca[3],
                mean = m[i], variance = v[j]))
        boot10_bca_leng_sd_g <- rbind(boot10_bca_leng_sd_g,
            list(value = resultats_boot10_g_bca[4],
                mean = m[i], variance = v[j]))
    }
}

bay10_perc_cov_g <- data.frame(value = c(), mean = c(),
    variance = c())
bay10_perc_cov_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
```

```
bay10_perc_leng_g <- data.frame(value = c(), mean = c(),
    variance = c())
bay10_perc_leng_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        shape <- m[i]^2/v[j]
        rate <- m[i]/v[j]
        resultats_bay10_g_perc <- simScenarioGamma(shape,
            rate, n = 10, method = "bayesian")
        bay10_perc_cov_g <- rbind(bay10_perc_cov_g,
            list(value = resultats_bay10_g_perc[1],
                mean = m[i], variance = v[j]))
        bay10_perc_cov_sd_g <- rbind(bay10_perc_cov_sd_g,
            list(value = resultats_bay10_g_perc[2],
                mean = m[i], variance = v[j]))
        bay10_perc_leng_g <- rbind(bay10_perc_leng_g,
            list(value = resultats_bay10_g_perc[3],
                mean = m[i], variance = v[j]))
        bay10_perc_leng_sd_g <- rbind(bay10_perc_leng_sd_g,
            list(value = resultats_bay10_g_perc[4],
                mean = m[i], variance = v[j]))
    }
}
```

Medium Sample Size $N = 20$

```
boot20_perc_cov_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_perc_cov_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_perc_leng_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_perc_leng_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_bca_cov_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_bca_cov_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_bca_leng_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_bca_leng_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        shape <- m[i]^2/v[j]
        rate <- m[i]/v[j]
```

```
        resultats_boot20_g_perc <- simScenarioGamma(shape,
            rate, 20, 10000, z0 = 0, a = 0, method = "boot")
        boot20_perc_cov_g <- rbind(boot20_perc_cov_g,
            list(value = resultats_boot20_g_perc[1],
                mean = m[i], variance = v[j]))
        boot20_perc_cov_sd_g <- rbind(boot20_perc_cov_sd_g,
            list(value = resultats_boot20_g_perc[2],
                mean = m[i], variance = v[j]))
        boot20_perc_leng_g <- rbind(boot20_perc_leng_g,
            list(value = resultats_boot20_g_perc[3],
                mean = m[i], variance = v[j]))
        boot20_perc_leng_sd_g <- rbind(boot20_perc_leng_sd_g,
            list(value = resultats_boot20_g_perc[4],
                mean = m[i], variance = v[j]))
        resultats_boot20_g_bca <- simScenarioGamma(shape,
            rate, 20, 10000, method = "boot")
        boot20_bca_cov_g <- rbind(boot20_bca_cov_g,
            list(value = resultats_boot20_g_bca[1],
                mean = m[i], variance = v[j]))
        boot20_bca_cov_sd_g <- rbind(boot20_bca_cov_sd_g,
            list(value = resultats_boot20_g_bca[2],
                mean = m[i], variance = v[j]))
        boot20_bca_leng_g <- rbind(boot20_bca_leng_g,
            list(value = resultats_boot20_g_bca[3],
                mean = m[i], variance = v[j]))
        boot20_bca_leng_sd_g <- rbind(boot20_bca_leng_sd_g,
            list(value = resultats_boot20_g_bca[4],
                mean = m[i], variance = v[j]))
    }
}

bay20_perc_cov_g <- data.frame(value = c(), mean = c(),
    variance = c())
bay20_perc_cov_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
bay20_perc_leng_g <- data.frame(value = c(), mean = c(),
    variance = c())
bay20_perc_leng_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        shape <- m[i]^2/v[j]
        rate <- m[i]/v[j]
        resultats_bay20_g_perc <- simScenarioGamma(shape,
            rate, n = 20, method = "bayesian")
        bay20_perc_cov_g <- rbind(bay20_perc_cov_g,
            list(value = resultats_bay20_g_perc[1],
```

```
                mean = m[i], variance = v[j]))
        bay20_perc_cov_sd_g <- rbind(bay20_perc_cov_sd_g,
            list(value = resultats_bay20_g_perc[2],
                mean = m[i], variance = v[j]))
        bay20_perc_leng_g <- rbind(bay20_perc_leng_g,
            list(value = resultats_bay20_g_perc[3],
                mean = m[i], variance = v[j]))
        bay20_perc_leng_sd_g <- rbind(bay20_perc_leng_sd_g,
            list(value = resultats_bay20_g_perc[4],
                mean = m[i], variance = v[j]))
    }
}
```

Large Sample Size $N = 30$

```
boot30_perc_cov_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_perc_cov_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_perc_leng_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_perc_leng_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_bca_cov_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_bca_cov_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_bca_leng_g <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_bca_leng_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        shape <- m[i]^2/v[j]
        rate <- m[i]/v[j]
        resultats_boot30_g_perc <- simScenarioGamma(shape,
            rate, 30, 10000, z0 = 0, a = 0, method = "boot")
        boot30_perc_cov_g <- rbind(boot30_perc_cov_g,
            list(value = resultats_boot30_g_perc[1],
                mean = m[i], variance = v[j]))
        boot30_perc_cov_sd_g <- rbind(boot30_perc_cov_sd_g,
            list(value = resultats_boot30_g_perc[2],
                mean = m[i], variance = v[j]))
        boot30_perc_leng_g <- rbind(boot30_perc_leng_g,
            list(value = resultats_boot30_g_perc[3],
                mean = m[i], variance = v[j]))
        boot30_perc_leng_sd_g <- rbind(boot30_perc_leng_sd_g,
            list(value = resultats_boot30_g_perc[4],
```

```
                mean = m[i], variance = v[j]))
        resultats_boot30_g_bca <- simScenarioGamma(shape,
            rate, 30, 10000, method = "boot")
        boot30_bca_cov_g <- rbind(boot30_bca_cov_g,
            list(value = resultats_boot30_g_bca[1],
                mean = m[i], variance = v[j]))
        boot30_bca_cov_sd_g <- rbind(boot30_bca_cov_sd_g,
            list(value = resultats_boot30_g_bca[2],
                mean = m[i], variance = v[j]))
        boot30_bca_leng_g <- rbind(boot30_bca_leng_g,
            list(value = resultats_boot30_g_bca[3],
                mean = m[i], variance = v[j]))
        boot30_bca_leng_sd_g <- rbind(boot30_bca_leng_sd_g,
            list(value = resultats_boot30_g_bca[4],
                mean = m[i], variance = v[j]))
    }
}

bay30_perc_cov_g <- data.frame(value = c(), mean = c(),
    variance = c())
bay30_perc_cov_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())
bay30_perc_leng_g <- data.frame(value = c(), mean = c(),
    variance = c())
bay30_perc_leng_sd_g <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        shape <- m[i]^2/v[j]
        rate <- m[i]/v[j]
        resultats_bay30_g_perc <- simScenarioGamma(shape,
            rate, n = 30, method = "bayesian")
        bay30_perc_cov_g <- rbind(bay30_perc_cov_g,
            list(value = resultats_bay30_g_perc[1],
                mean = m[i], variance = v[j]))
        bay30_perc_cov_sd_g <- rbind(bay30_perc_cov_sd_g,
            list(value = resultats_bay30_g_perc[2],
                mean = m[i], variance = v[j]))
        bay30_perc_leng_g <- rbind(bay30_perc_leng_g,
            list(value = resultats_bay30_g_perc[3],
                mean = m[i], variance = v[j]))
        bay30_perc_leng_sd_g <- rbind(bay30_perc_leng_sd_g,
            list(value = resultats_bay30_g_perc[4],
                mean = m[i], variance = v[j]))
    }
}
```

Log-Normal Distribution

Small Sample Size $N = 5$

```
boot5_perc_cov_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_perc_cov_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_perc_leng_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_perc_leng_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_bca_cov_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_bca_cov_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_bca_leng_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot5_bca_leng_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        mu <- log(m[i]^2/sqrt(v[j] + m[i]^2))
        sd <- sqrt(log(v[j]/m[i]^2 + 1))
        resultats_boot5_l_perc <- simScenarioLogNorm(mu,
            sd, 5, 10000, z0 = 0, a = 0, method = "boot")
        boot5_perc_cov_l <- rbind(boot5_perc_cov_l,
            list(value = resultats_boot5_l_perc[1],
                mean = m[i], variance = v[j]))
        boot5_perc_cov_sd_l <- rbind(boot5_perc_cov_sd_l,
            list(value = resultats_boot5_l_perc[2],
                mean = m[i], variance = v[j]))
        boot5_perc_leng_l <- rbind(boot5_perc_leng_l,
            list(value = resultats_boot5_l_perc[3],
                mean = m[i], variance = v[j]))
        boot5_perc_leng_sd_l <- rbind(boot5_perc_leng_sd_l,
            list(value = resultats_boot5_l_perc[4],
                mean = m[i], variance = v[j]))
        resultats_boot5_l_bca <- simScenarioLogNorm(mu,
            sd, 5, 10000, method = "boot")
        boot5_bca_cov_l <- rbind(boot5_bca_cov_l, list(value = resultats_boot5_l_bca[1],
            mean = m[i], variance = v[j]))
        boot5_bca_cov_sd_l <- rbind(boot5_bca_cov_sd_l,
            list(value = resultats_boot5_l_bca[2],
                mean = m[i], variance = v[j]))
        boot5_bca_leng_l <- rbind(boot5_bca_leng_l,
            list(value = resultats_boot5_l_bca[3],
                mean = m[i], variance = v[j]))
        boot5_bca_leng_sd_l <- rbind(boot5_bca_leng_sd_l,
```

```
            list(value = resultats_boot5_l_bca[4],
                mean = m[i], variance = v[j]))
    }
}

bay5_perc_cov_l <- data.frame(value = c(), mean = c(),
    variance = c())
bay5_perc_cov_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
bay5_perc_leng_l <- data.frame(value = c(), mean = c(),
    variance = c())
bay5_perc_leng_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        mu <- log(m[i]^2/sqrt(v[j] + m[i]^2))
        sd <- sqrt(log(v[j]/m[i]^2 + 1))
        resultats_bay5_l_perc <- simScenarioLogNorm(mean = mu,
            std = sd, n = 5, method = "bayesian")
        bay5_perc_cov_l <- rbind(bay5_perc_cov_l, list(value = resultats_bay5_l_perc[1],
            mean = m[i], variance = v[j]))
        bay5_perc_cov_sd_l <- rbind(bay5_perc_cov_sd_l,
            list(value = resultats_bay5_l_perc[2],
                mean = m[i], variance = v[j]))
        bay5_perc_leng_l <- rbind(bay5_perc_leng_l,
            list(value = resultats_bay5_l_perc[3],
                mean = m[i], variance = v[j]))
        bay5_perc_leng_sd_l <- rbind(bay5_perc_leng_sd_l,
            list(value = resultats_bay5_l_perc[4],
                mean = m[i], variance = v[j]))
    }
}
```

Small Sample Size $N = 10$

```
boot10_perc_cov_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot10_perc_cov_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot10_perc_leng_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot10_perc_leng_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot10_bca_cov_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot10_bca_cov_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot10_bca_leng_l <- data.frame(value = c(), mean = c(),
```

```
    variance = c())
boot10_bca_leng_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        mu <- log(m[i]^2/sqrt(v[j] + m[i]^2))
        sd <- sqrt(log(v[j]/m[i]^2 + 1))
        resultats_boot10_l_perc <- simScenarioLogNorm(mu,
            sd, 10, 10000, z0 = 0, a = 0, method = "boot")
        boot10_perc_cov_l <- rbind(boot10_perc_cov_l,
            list(value = resultats_boot10_l_perc[1],
                mean = m[i], variance = v[j]))
        boot10_perc_cov_sd_l <- rbind(boot10_perc_cov_sd_l,
            list(value = resultats_boot10_l_perc[2],
                mean = m[i], variance = v[j]))
        boot10_perc_leng_l <- rbind(boot10_perc_leng_l,
            list(value = resultats_boot10_l_perc[3],
                mean = m[i], variance = v[j]))
        boot10_perc_leng_sd_l <- rbind(boot10_perc_leng_sd_l,
            list(value = resultats_boot10_l_perc[4],
                mean = m[i], variance = v[j]))
        resultats_boot10_l_bca <- simScenarioLogNorm(mu,
            sd, 10, 10000, method = "boot")
        boot10_bca_cov_l <- rbind(boot10_bca_cov_l,
            list(value = resultats_boot10_l_bca[1],
                mean = m[i], variance = v[j]))
        boot10_bca_cov_sd_l <- rbind(boot10_bca_cov_sd_l,
            list(value = resultats_boot10_l_bca[2],
                mean = m[i], variance = v[j]))
        boot10_bca_leng_l <- rbind(boot10_bca_leng_l,
            list(value = resultats_boot10_l_bca[3],
                mean = m[i], variance = v[j]))
        boot10_bca_leng_sd_l <- rbind(boot10_bca_leng_sd_l,
            list(value = resultats_boot10_l_bca[4],
                mean = m[i], variance = v[j]))
    }
}

bay10_perc_cov_l <- data.frame(value = c(), mean = c(),
    variance = c())
bay10_perc_cov_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
bay10_perc_leng_l <- data.frame(value = c(), mean = c(),
    variance = c())
bay10_perc_leng_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
```

```
for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        mu <- log(m[i]^2/sqrt(v[j] + m[i]^2))
        sd <- sqrt(log(v[j]/m[i]^2 + 1))
        resultats_bay10_l_perc <- simScenarioLogNorm(mean = mu,
            std = sd, n = 10, method = "bayesian")
        bay10_perc_cov_l <- rbind(bay10_perc_cov_l,
            list(value = resultats_bay10_l_perc[1],
                mean = m[i], variance = v[j]))
        bay10_perc_cov_sd_l <- rbind(bay10_perc_cov_sd_l,
            list(value = resultats_bay10_l_perc[2],
                mean = m[i], variance = v[j]))
        bay10_perc_leng_l <- rbind(bay10_perc_leng_l,
            list(value = resultats_bay10_l_perc[3],
                mean = m[i], variance = v[j]))
        bay10_perc_leng_sd_l <- rbind(bay10_perc_leng_sd_l,
            list(value = resultats_bay10_l_perc[4],
                mean = m[i], variance = v[j]))
    }
}
```

Medium Sample Size $N = 20$

```
boot20_perc_cov_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_perc_cov_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_perc_leng_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_perc_leng_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_bca_cov_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_bca_cov_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_bca_leng_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot20_bca_leng_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        mu <- log(m[i]^2/sqrt(v[j] + m[i]^2))
        sd <- sqrt(log(v[j]/m[i]^2 + 1))
        resultats_boot20_l_perc <- simScenarioLogNorm(mu,
            sd, 20, 10000, z0 = 0, a = 0, method = "boot")
        boot20_perc_cov_l <- rbind(boot20_perc_cov_l,
            list(value = resultats_boot20_l_perc[1],
                mean = m[i], variance = v[j]))
```

```
        boot20_perc_cov_sd_l <- rbind(boot20_perc_cov_sd_l,
            list(value = resultats_boot20_l_perc[2],
                mean = m[i], variance = v[j]))
        boot20_perc_leng_l <- rbind(boot20_perc_leng_l,
            list(value = resultats_boot20_l_perc[3],
                mean = m[i], variance = v[j]))
        boot20_perc_leng_sd_l <- rbind(boot20_perc_leng_sd_l,
            list(value = resultats_boot20_l_perc[4],
                mean = m[i], variance = v[j]))
        resultats_boot20_l_bca <- simScenarioLogNorm(mu,
            sd, 20, 10000, method = "boot")
        boot20_bca_cov_l <- rbind(boot20_bca_cov_l,
            list(value = resultats_boot20_l_bca[1],
                mean = m[i], variance = v[j]))
        boot20_bca_cov_sd_l <- rbind(boot20_bca_cov_sd_l,
            list(value = resultats_boot20_l_bca[2],
                mean = m[i], variance = v[j]))
        boot20_bca_leng_l <- rbind(boot20_bca_leng_l,
            list(value = resultats_boot20_l_bca[3],
                mean = m[i], variance = v[j]))
        boot20_bca_leng_sd_l <- rbind(boot20_bca_leng_sd_l,
            list(value = resultats_boot20_l_bca[4],
                mean = m[i], variance = v[j]))
    }
}

bay20_perc_cov_l <- data.frame(value = c(), mean = c(),
    variance = c())
bay20_perc_cov_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
bay20_perc_leng_l <- data.frame(value = c(), mean = c(),
    variance = c())
bay20_perc_leng_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        mu <- log(m[i]^2/sqrt(v[j] + m[i]^2))
        sd <- sqrt(log(v[j]/m[i]^2 + 1))
        resultats_bay20_l_perc <- simScenarioLogNorm(mean = mu,
            std = sd, n = 20, method = "bayesian")
        bay20_perc_cov_l <- rbind(bay20_perc_cov_l,
            list(value = resultats_bay20_l_perc[1],
                mean = m[i], variance = v[j]))
        bay20_perc_cov_sd_l <- rbind(bay20_perc_cov_sd_l,
            list(value = resultats_bay20_l_perc[2],
                mean = m[i], variance = v[j]))
        bay20_perc_leng_l <- rbind(bay20_perc_leng_l,
```

```
            list(value = resultats_bay20_l_perc[3],
                mean = m[i], variance = v[j]))
        bay20_perc_leng_sd_l <- rbind(bay20_perc_leng_sd_l,
            list(value = resultats_bay20_l_perc[4],
                mean = m[i], variance = v[j]))
    }
}
```

Large Sample Size $N = 30$

```
boot30_perc_cov_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_perc_cov_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_perc_leng_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_perc_leng_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_bca_cov_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_bca_cov_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_bca_leng_l <- data.frame(value = c(), mean = c(),
    variance = c())
boot30_bca_leng_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        mu <- log(m[i]^2/sqrt(v[j] + m[i]^2))
        sd <- sqrt(log(v[j]/m[i]^2 + 1))
        resultats_boot30_l_perc <- simScenarioLogNorm(mu,
            sd, 30, 10000, z0 = 0, a = 0, method = "boot")
        boot30_perc_cov_l <- rbind(boot30_perc_cov_l,
            list(value = resultats_boot30_l_perc[1],
                mean = m[i], variance = v[j]))
        boot30_perc_cov_sd_l <- rbind(boot30_perc_cov_sd_l,
            list(value = resultats_boot30_l_perc[2],
                mean = m[i], variance = v[j]))
        boot30_perc_leng_l <- rbind(boot30_perc_leng_l,
            list(value = resultats_boot30_l_perc[3],
                mean = m[i], variance = v[j]))
        boot30_perc_leng_sd_l <- rbind(boot30_perc_leng_sd_l,
            list(value = resultats_boot30_l_perc[4],
                mean = m[i], variance = v[j]))
        resultats_boot30_l_bca <- simScenarioLogNorm(mu,
            sd, 30, 10000, method = "boot")
        boot30_bca_cov_l <- rbind(boot30_bca_cov_l,
            list(value = resultats_boot30_l_bca[1],
```

```
                mean = m[i], variance = v[j]))
        boot30_bca_cov_sd_l <- rbind(boot30_bca_cov_sd_l,
            list(value = resultats_boot30_l_bca[2],
                mean = m[i], variance = v[j]))
        boot30_bca_leng_l <- rbind(boot30_bca_leng_l,
            list(value = resultats_boot30_l_bca[3],
                mean = m[i], variance = v[j]))
        boot30_bca_leng_sd_l <- rbind(boot30_bca_leng_sd_l,
            list(value = resultats_boot30_l_bca[4],
                mean = m[i], variance = v[j]))
    }
}

bay30_perc_cov_l <- data.frame(value = c(), mean = c(),
    variance = c())
bay30_perc_cov_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())
bay30_perc_leng_l <- data.frame(value = c(), mean = c(),
    variance = c())
bay30_perc_leng_sd_l <- data.frame(value = c(), mean = c(),
    variance = c())

for (i in 1:length(m)) {
    for (j in 1:length(v)) {
        mu <- log(m[i]^2/sqrt(v[j] + m[i]^2))
        sd <- sqrt(log(v[j]/m[i]^2 + 1))
        resultats_bay30_l_perc <- simScenarioLogNorm(mean = mu,
            std = sd, n = 30, method = "bayesian")
        bay30_perc_cov_l <- rbind(bay30_perc_cov_l,
            list(value = resultats_bay30_l_perc[1],
                mean = m[i], variance = v[j]))
        bay30_perc_cov_sd_l <- rbind(bay30_perc_cov_sd_l,
            list(value = resultats_bay30_l_perc[2],
                mean = m[i], variance = v[j]))
        bay30_perc_leng_l <- rbind(bay30_perc_leng_l,
            list(value = resultats_bay30_l_perc[3],
                mean = m[i], variance = v[j]))
        bay30_perc_leng_sd_l <- rbind(bay30_perc_leng_sd_l,
            list(value = resultats_bay30_l_perc[4],
                mean = m[i], variance = v[j]))
    }
}
```