My amazing title

Tony Ni
April DD, 20YY

Submitted to the Department of
Mathematics and Statistics
of Amherst College in partial fulfillment
of the requirements for the degree of
Bachelor of Arts with honors.

Advisor:
Brittney Bailey

Abstract

The abstract should be a short summary of your thesis work. A paragraph is usually sufficient here.

Missing data is ubiquitious in statistics: from nonresponse in survey studies to issues with data collection in field studies, it is often unavoidable to encounter missing data in practice. Our thesis begins with an overview of missingness, specifically with left-censored data, and introduces four different methods: substitution, maxmimum likelihood, kaplan-meier, and regression on order statistics to combat this. We explore a simulation study to compare and contrast the effectiveness of these methods when considering for the distribution of the data, censoring rates, and the size of the dataset in order to validate several claims made in several papers regarding these methods. Finally, we explore left-censored data in the form of coal ash contamination in groundwater wells, and apply our findings to this case study. Our results from our simulation study show that BLAH.

Acknowledgments

Use this space to thank those who have helped you in the thesis process (professors, staff, friends, family, etc.). If you had special funding to conduct your thesis work, that should be acknowledged here as well.

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction {intro}

The concept of missing data is ubiquitous across academic disciplines and often complicates real-world studies. Most studies utilize data collected through surveys, questionnaires, and/or field research which is why missing data is often unavoidable. Missing data can hinder one's ability to work with and analyze the phenomena at hand, giving rise to inaccurate or even misleading analyses.

Barnard & Meng (1999) outline several significant issues when conducting analysis on missing data. Firstly, missing data can introduce bias in regards to parameter estimation. It can also lead to a reduction in statistical power, which can affect the conclusions one makes during studies involving hypothesis testing. Finally, missing data can introduce complications with statistical software and lead to functions not working as intended, if they have not accounted for the possibility of the data containing missingness.

This thesis will go into a more specific instance of missing data known as censoring, which is *the condition when one has only partial information regarding the values of a measurement within a dataset*. In this chapter, we will introduce and define the three types of censored data, discuss the challenges with the reporting of censored data, and explore common statistical approaches to handling censored data.

1.1 Censored Data

As discussed previously, censored data is a specific type of missingness where one has only partial information regarding the values of a measurement in a dataset. There are three types of censoring which can occur: right censoring, interval censoring, and left censoring.

1.1.1 Right Censoring

Right censoring is a specific instance in which we only know that the true value of a data point lies above a certain threshold, but it is unknown by how much. Suppose a study on income and mortality is conducted with the variable of interest, T, being the time measured from the start of the study to the death of the participant. The study has a duration of 5 years, in which participants are expected to submit a form regarding their annual income. The value for the participant would be considered to be right-censored if at any point during the study, they failed to follow-up, or if the participant was still alive at the conclusion of the 5 year study. In this design study, several possibilities can occur, illustrated in Figure 1.1.

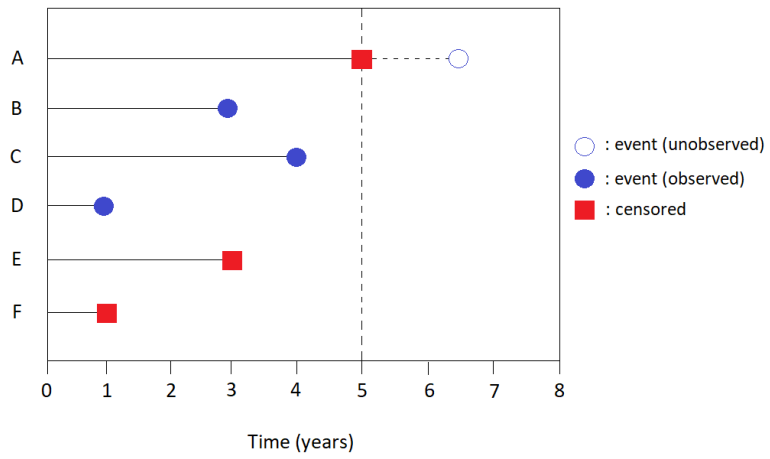

Figure 1.1: Right Censoring Example

As illustrated by individual A in Figure 1.1, this individual lives on until the termination of the study. We don't know at what point they passed away exactly, since they didn't pass away during the time constraints of the study. As such, the only information we have is $T > 5$.

If an individual does pass away at some point, t_i, *during* the study, then $T = t_i$. This can be illustrated within Figure 1.1 by individuals B, C, and D for which $T = 3$, $T = 4$, and $T = 1$.

There is a final possibility for individuals who choose to censor themselves. Illustrated in Figure 1.1 by individuals E and F, we can see that they are marked as censored at $T = 3$ and $T = 1$, respectively. These individuals may have chosen to stop submitting information to the study or drop out of the study entirely without warning. As we have no information about whether or not if they died or simply did not submit their form, all we know is that the individual died/will die at some point after the point at which they were censored.

Right censoring is the most common type of censoring and can often be found in clinical trial studies, mortality studies, and other forms of surival analyses.

1.1.2 Left Censoring

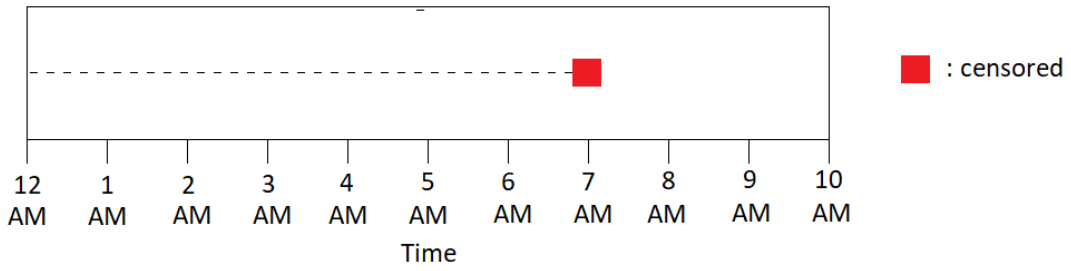

Figure 1.2: Left Censoring Example

In contrast with right censoring, left censoring is a specific instance of censoring in which we only know that the true value of a data point falls below a certain threshold which we call the *limit of detection* (LOD).

To understand this concept better, consider the following example. Imagine a scenario in which you are attempting to estimate the time at which the sun rises each morning. You plan to wake up every morning far before the sun rises, but on the first day of the study, you oversleep and wake up at 7:00 A.M. with the sun already out. We now have an instance of left-censored data. We want to know the time at which the sun rose, but all we have is an upper limit (7:00 A.M.).

Left censoring is commonly found in environmental, water quality, and chemical-related research where the focus is on the concentration of an analyte. Due to limitations on measuring instruments, left censored data are commonly found in

these types of studies. The most pressing issue of left-censored data mostly lie in the difficulty of distinguishing between extremely low values and statistical noise (Hall, Perry, & Anderson, 2020).

1.1.3 Interval Censoring

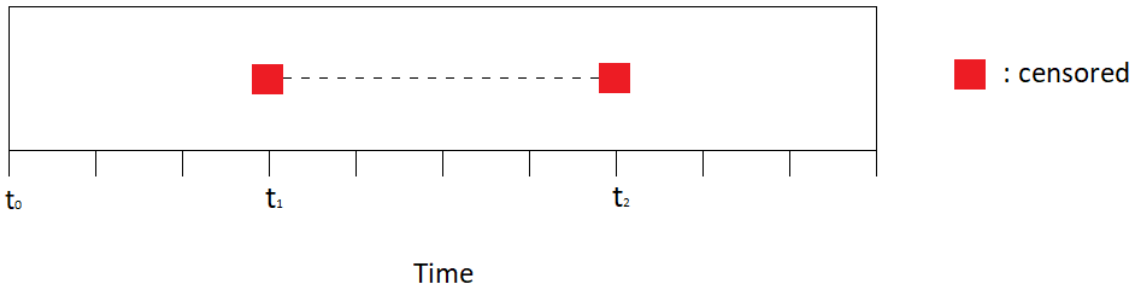

Figure 1.3: Interval Censoring Example

Interval censoring is another form of censoring in which the random variable of interest is known to be between an interval of two values. Considering a random variable T, which denotes the survival time of interest, if interval censoring is at hand, we can denote the interval containing T to be $I = [t_1, t_2]$, with t_1 being the beginning of the interval and t_2 being the end of the interval. Left and right censoring are special cases of interval censoring. In the case of left censoring, $t_1 = 0$; and conversely in the case of right censoring, $t_2 = \infty$.

To conceptualize interval censoring, we can consider a example study on virus testing in which participants get their blood drawn in order to detect whether or not they test positive for a virus or not. The random variable in question is T, which represents the exact timepoint at which the subject contracted the virus. If an individual was first tested at time t_1 and tested negative, but was tested again at a later time t_2 and tested positive, the specific time t at which the subject contracted

the virus is unknown. All we know is that it lies somewhere between the interval, $I = [t_1, t_2]$, but not the exact time at which they contracted it.

The focus of my thesis deals specifically with the challenges of reporting and working with left-censored data.

1.2 Challenges of Reporting Censored Data

There is no universal reporting practice for values below the LOD which can lead to confusion amongst researchers. The lack of standardization makes it difficult to distinguish values below the LOD and uncensored values. This can lead to values below the LOD unintentionally being overlooked, causing faulty analysis or conclusions which are heavily flawed.

In a study involving the precision of lead measurements near concentrations of the limit of detection, Berthouex (1993) discusses the disparity among chemists regarding practices involving the recording values below the LOD. He enumerates the following list.

1. Reporting the letters ND, "not detected"
2. Reporting the numeric value of the LOD
3. Reporting "< LOD", where LOD is the numeric value of the LOD
4. Reporting some value between 0 and the LOD, such as one-half the LOD
5. Reporting the actual measured concentration, even if it falls below the LOD
6. Reporting the actual measured concentration, followed by "(LOD)"
7. Reporting the actual measured concentration with a precision ($\pm$) statement

According to Gilbert (1987), the latter three methods are the best procedures to follow, especially from a practical and statistical point of view. He argues that assuming the small concentration values are not from some sort of measurement error during data collection, then the measured concentration holds value. As such, recording a measurement as "below LOD" without any sort of accompanying value would be discarding useful information which could have been used in practice and analysis.

Berthouex (1993) discusses the prevalence in regards to the practice of censoring data by reporting only values which are above the detection limit and discarding those which fail to yield quantifiable results. In the study he conducted, five laboratories were assigned tasks to measure samples of a certain solution. The laboratories were not given information regarding the intent of the study, but a general statement that the concentrations being measured were of "low" concentrations. All but one laboratory recorded the actual measured concentrations even though they fell below the LOD. Fortunately, the original measurements for the laboratory that did not report values for all samples were maintained and able to be recovered. Berthouex (1993) stresses the importance of standardization in reporting practices for laboratories and suggested reporting all measurements accompanied with some precision statement, so that data is not lost.

Further supporting the stance of keeping all concentration measurements rather than only those above the detection limit, Monte-Carlo experiments were conducted by Gllllom, Kirsch, Gilroy, & Survey (1984) to investigate trend-detection for water-quality data. Trend detection is the practice of determining whether the values of a random variable generally increase or decrease over a period of time. They found a general relationship of decreasing trend detection percentages with increased censoring levels, attributing this to the limited availability of information in censored data.

[START HERE]

1.3 Approaches

It is important to note that the values below the LOD still contain information, specifically that the values is between the lower bound value (if it exists) and the LOD (Chen et al., 2011). As such, there are a variety of statistical treatments to handle censored data which have been popularized in the statistical literature which will be discussed within this section.

Omission involves the deletion of data points which are deemed to be invalid as a result of left-censoring or any other deficiencies in the data. This is also more commonly known as *available-case analysis*, in which statistical analysis is conducted while only considering the observations which have no missing data on the variables of interest, and excluding the observations with missing values (May, 2012). May argues against this approach and claims that the loss of information from discarding data and the inflation of standard errors of estimates (when discussing missingness in a regression context) will invariably be inflated as a result of the decreased sample size. The advantages of omission lies in its ease of implementation.

Apart from available-case analysis, over the past century, a myriad of methods to deal with censoring have been developed to counter this issue – some more statistically sound than others. We will review some of the most common methods to estimate descriptive statistics involving censored data, which include: substitution, maximum likelihood estimation, Kaplan-Meier, and regression on order statistics (Lafleur et al., 2011).

1.3.1 Substitution Method

Often condemned in papers as a statistically unsound method to handle censored data, substitution methods are ubiquitous in the chemical and environmental sciences

as an appropriate and recommended method to work with left-censored chemical concentration data (Canales, 2018).

The substitution method simply involves imputing in a replacement value in lieu of the censored data point. The lack of a global, standardized replacement value to substitute is one of the most pronounced downside of this method. The replacement value used may differ between studies but common values include: $\frac{LOD}{2}$, $\frac{LOD}{\sqrt{2}}$, or LOD (Lee & Helsel, 2005). Different disciplines have their own suggested "best" replacement value to use, an example being $\frac{3}{4}$ times the LOD being a common replacement value in geochemistry (Crovelli, 1993). However, it must be recognized that the substitution method is a statistically unsound technique which is often used in non-rigorous statistical settings due to them being quite easy to implement (Chen et al., 2011). As such, there have been several studies in order to investigate the effectiveness of the method.

Proponents of the substitution method claim that the replacement value $\frac{LOD}{2}$ is useful for data sets in which the majority of the data are below the LOD or when the distribution of the data is highly skewed; the definition of "highly skewed" being any distribution with a geometric standard deviation (a measure of spread commonly used in tandem with log-normal distributions) of 3 or more (Hornung & Reed, 1989). They also suggest using $\frac{LOD}{\sqrt{2}}$ when there are only a few data points below the LOD or when the data is not highly skewed.

Substitution methods are flawed as they can often introduce a "signal" which was not originally present within the data, or even obstruct an actual signal which was present in the original data (Lee & Helsel, 2005). Numerous authors have advised against the usage of substitution methods for being statistically inappropriate to use. Glass and Gray (2001) found that both introduce large errors and biases in descriptive statistics of interest. Thompson and Nelson (2001) conducted a study in

which they found similar results, in that it often led to biased parameter estimates and "artificially small standard error estimates." Hewett and Ganser (2007) also found in their simulation study that the substitution method yielded the lowest average bias and root mean squared error values (comparison metrics to measure accuracy) in their estimation of the mean. Overall, the overall consensus seems to advise against the practice of these substitution techniques.

[paragraph talking about proponents of the substitution methods??]

1.3.2 Maximum Likelihood Estimation Method

Maximum likelihood (ML) estimation is a parametric technique which allows us to estimate the parameters of a distribution or model when the data is from a multivariate normal distribution.

To give a brief introduction to the mechanisms of ML estimation, let $f(x|\theta)$ denote the probability density function (PDF) which specifies the probability of observing the random variable x given the parameter θ.

Given a random, independently and identically distributed (*i.i.d.*) set of random variables $X_1, X_2, ..., X_n$ from $f(x|\theta)$, we know that each individual observation x_i's are statistically independent from one another, which allows us to express the PDF as the product of all individual densities. For every observed random sample $x_1, ..., x_n$, we can define the joint density function to be:

$$f(x_1, ..., x_n|\theta) = f(x_1|\theta)...f(x_n|\theta) = \prod_{i=1}^{n} f(x_i|\theta)$$

In most real-life scenarios, the actual (observed) data is already given, and our goal is to find the PDF which is most likely to generate our observed values. In order to solve this inverse problem, we introduce the likelihood function, which is defined as

the joint density of the observed data as a function of the parameter (with the data held as a fixed constant).

In mathematical notation, upon observing the given data, $f(x_1, ..., x_n|\Theta)$ becomes a function of θ alone, so we obtain a likelihood of:

$$lik(\theta) = f(x_1, ..., x_n|\theta)$$

It is important to recognize the difference which separates the likelihood function and the PDF. The PDF is a function of the observed data given a parameter(s). It gives information regarding the probability of a particular data value for a fixed parameter.

On the other hand, the likelihood function is a function of the parameter, given a set of observed data. It tells us the likelihood of observing a particular parameter value for a fixed set of data.

Our goal is to obtain the ML estimate of our parameter which maximizes the likelihood function, $lik(\theta)$, in other words, to obtain a θ which makes our observed data the most probable.

As we previously declared our random variables $X_1, X_2, ..., X_n$ to be i.i.d, we can rewrite the likelihood to be a product of the marginal densities:

$$lik(\theta) = \prod_{i=1}^{n} f(x_i|\theta)$$

in which we can then maximize the likelihood to find the best mle of θ to best capture our observed data.

Yavuz et al. (2017) discuss the usage of MLE method, when missing data is present, and note that it is only appropriate to use for non-negative probability distributions such as: exponential, log-normal, normal, and Weibull.

When left censoring is present, the likelihood function changes in order to account

for both the censored observations and the uncensored observations and becomes:

$$lik(\theta) = \prod_{i=1}^{n} f(x_i|\theta)^{\delta_i} \times F(x_i|\theta)^{1-\delta_i}$$

in which δ_i is an indicator, representing whether or not if the ith observation is censored or not:

$$\delta_i = \begin{cases} 0, & \text{if censored} \\ 1, & \text{if uncensored} \end{cases}$$

From this updated definition of the likelihood function which must be used in the presence of left censored data, it is then possible to follow typical procedures to find the estimator, θ, which maximizes the likelihood, also known as the *maximum likelihood estimator*. With this knowledge, the descriptive statistics of interest (mean, variance, etc.) relating to the specified distribution can be calculated.

Canales (2018) outlines a imputation technique which involves replacing censored observations with values from the estimated parameterized distribution. However, is not mentioned explicitly how this imputation method is conducted.

The code for the MLE method will be handled with the `cenmle` function in the `NADA` package, which allows the user to specify censored and uncensored data, and uses the LOD as the placeholder. As this method is not an imputation technique, values are not replaced. This method allows us to calculate the summary statistics for the entire data set – including the censored values. (remove and write own code??)

As a technique which heavily relies upon knowing a distribution which best models the data, MLE is one of the most well-known parametric approaches to handling values below the LOD. Many studies use the MLE as a sort of baseline method of handling censored values, to which they compare their new techniques upon (Ganser

& Hewett, 2010). However, it must be known that regardless of the prevalence of the MLE method, it is not free from its own downfalls. Canales (2018) found that the MLE method seems to underperform when the data in question was highly skewed, in which overinflated mean squared errors were often obtained. Being a technique which is so heavily dependent upon distributional assumptions, an incorrect specification of the distribution of the censored data will inevitably lead to misleading results (Bolks, DeWire, & Harcum, 2014).

1.3.3 Kaplan-Meier Method

As a phenomenon, censoring is most often discussed in survival analysis, which concerns itself with techniques to analyze a time to an *event variable*. As its name suggests, these variables measure the time which passes until some sort of event occurs. This can be as innocuous as the time until device breaks, time until birds migrate away from their homes, time until a person passes away, etc. Regardless of which, all these scenarios share a common problem in terms of the possibility of the data being "censored."

The Kaplan-Meier (KM) method is a common nonparametric technique used to deal with censored data. Nonparametric methods do not utilize any information regarding the parameters for a specified distribution, like the mean and standard deviation for the normal distribution. The KM method was originally developed to handle right-censored survival analysis data. The advantages of the KM method lie in its robustness as a nonparametric method, it performs well without having to depend upon distributional assumptions. Many recommend its usage for when there are cases of severe censoring, instances where $> 90\%$ of the data is censored (Canales, 2018).

To introduce the concept of the KM-estimator, it is helpful to take a look into its usages in survival analysis studies where the focus is often on a type of data known as

"time to event" data. These types of studies often involve events such time to death, time to failure, and so forth.

The KM-estimator is a statistic used to estimate the survival curve from the empirical data while accounting for the possibilities of certain values being censored. It does this by assuming that censoring is independent from the event of interest and that survival probabilities remain the same in observations found early in the study and those recruited later in the study.

The KM-estimator when performing an empirical estimation of the survival curve at time t can be represented by the following equation:

$$\hat{S}(t) = \prod_{t_i \leq\ t} \left(1 - \frac{d_i}{n_i}\right)$$

where t_i is the distinct event time, d_i is the number of event occurrences at time t_i, and n_i is the number of followup times (t_i) that are $\geq t_i$ (how many observations in sample survived at least/or past the time t_i) (Klein & Moeschberge, 2003).

Typically, the KM-estimator can only be used to estimate the distribution function of right-censored data, in which a data point is above a certain threshold, but it is unknown by how much. A simple tweak to the typical KM-method, allows for the estimation of the survival curve with left-censored values.

Helsel (2005, as cited in Yavuz et al., 2017) provides a detailed explanation on how to apply the KM method when left censoring is present. Firstly, it is essential to reverse the left-censored data through a transformation algorithm before using the KM method to change them into right-censored data.

Let $x_i \ldots x_n$ be the values for the observations $i = 1, 2, ..., n$. Arrange all the left-censored values in descending order and then subtract them by M, a constant bigger than the biggest value of the dataset, in order to get the transformed, right-censored

value, $M - x_i$. All values are then arranged in ascending order to be used to estimate the survival function through the Kaplan-Meier estimator.

It must be known that the KM-method is not an imputation procedure, but instead an estimation technique that allows for the calculation of descriptive statistics for left-censored datasets. Nian (1997 gives the expressions to calculate the estimated mean, median, and variance below:

NOTE TO SELF::: USE "backslash" displaystyle to do pretty bottom/above indexes

$\hat{\mu} = \int_0^\infty \hat{S}(t)\ dt\ \hat{M} = \hat{S}^{-1}\left(\frac{1}{2}\right)\ Var(\hat{\mu}) = \sum_{i=1}^{r}\left(\int_{t_i}^\infty \hat{S}(t)\ dt\right)^2 \frac{d_i}{n_i(n_i-d_i)}$

1.3.4 Regression on Order Statistics

Lastly, regression on order statistics (ROS) combines both the parametric nature of the MLE approach and nonparametric nature of the KM method. ROS is a semi-parametric method which assumes an underlying normal or lognormal distribution for the censored measurements but makes no assumption towards the distribution of uncensored measurements.

(Environmental Protection Agency, 2009) provides a a more detailed explanation to the methodology of ROS, but the basic procedures will be outlined in this thesis.

ROS begins with the estimation of the cumulative probability associated with each distinct LOD. This cumulative probability is distributed equally between the censored values with a common LOD (see (Environmental Protection Agency, 2009), for more details). A regression model is fit between the uncensored values and the distributional quantiles. The slope and intercept of the regression line from this model is then used to estimate the mean and standard deviation of the distributional model which are then used to generate imputed values for the censored observations.

In order for ROS to be utilized, there needs to be at least 5 known values and

more than half the values within the censored variables must be known. As regression is utilized in this method, the response variable must also be a linear function of the explanatory variable (quantiles). Additionally, the errors should have constant variance (Lee & Helsel, 2005).

The `NADA` package contains the function `ros` which provides an implementation of regression on order statistics which allows us to calculate descriptive statistics for left censored values.

[INSERT PARAGRAPH TO TRANSITION TO CHAPTER 3 (?)]

Chapter 2 Simulations

Having discussed the various methods to handle left-censored data in the previous chapter, we now turn to a simulation study in order to evaluate the strengths and weaknesses of each method in cases of differing censoring rate, sample size, and distribution. We will also discuss the implementation of the methods, data generating mechanisms, and specific evaluation metrics to assess the performance of each method.

2.1 Aims

The question of which method is the best to use is frequently discussed topic within the field. Many studies have been conducted over the years to evaluate the performance of these methods to handle left-censored data, with the results being widely varied and largely inconclusive. First and foremost, a large issue comes in that every conducted study widely differs in the methods being investigated and the scope of the study. As an example of the broad differences between studies which can make comparisons difficult, (Antweiler, 2015) evaluates the effectiveness of 11 different methods with several censoring rates and distributional assumptions, using the median absolute deviation (MAD) as their performance metric of choice. Meanwhile, (Hall, Perry, & Anderson, 2020) focuses instead on the applications of such methods from a water-quality focused context and investigates the performance of the four methods used in this thesis while disregarding distributional assumptions and censoring rates.

As each of these studies are concerned with their own goals – specifics of their study will inevitably be different. Studies that are more focused on a general, broad audience, with no assumptions as to what sort of data the individual is working with – may find more use with the conclusion and results that investigators like Antweiler come up with. There may also be individuals who are more focused on the performance of such methods in a specific context, as in the study conducted by Hall. There is no common ground between statisticians on the optimality of methods, prompting our own foray into this topic. I wish to incorporate the detailed specifications of a simulation study, in essence, taking into consideration distributional parameters of the data-generating mechanism, censoring rates, and sample sizes for each run, while also keeping it applicable to our case-study specific data.

With our simulation study, we wish to identify settings where a method can be effective but also those in which the methods may not be able to perform quite as well. Several investigators in this field have found issues with certain methods underperforming under certain conditions, and brings up the possibility of particular methods being more equipped than others to deal with different rates of censoring. To provide an example of an instance when investigators have found a certain method to perform better than others, we can take a look at Canales (2018)'s study on methods to handle left-censored microbial risk assessment. They found that the substitution method seemed to work much better than expected while other methods, such as the MLE method, seemed to have trouble when applied to highly skewed data. Meanwhile Antweiler (2015)'s report suggest that regardless of the method or sample size, obtaining reliable estimates from datasets where censoring was greater than 40% was unfeasible.

Claims regarding the effectiveness of methods with regards to censoring rates, distribution of data, and sample size are all highly contentious. In order to get a

better idea of sense of how these claims hold up, our goal is to evaluate the validity of those claims by conducting a simulation study of our own to put these methods into practice. We expect the results of our simulation study to largely depend on how we specify our data generating mechanism, which will be discussed shortly.

2.2 Data-Generating Mechanisms

The data generated for use in our study will be obtained by using parametric draws from user-specified distributions (log-normal, exponential, and Weibull), as the methods utilized can only be used with non-negative distributions (Yavuz, Tekindal, & Dog, 2017). Our data-generating mechanism also alters criterion such as the sample size, $n_{obs} = \{10, 100, 1000\}$ and censoring rate, $R = \{0.10, 0.30, 0.50\}$. In the following sections, we will often interchangeably use "small, medium, large" and "low, medium, and high" to differentiate between the different values for sample size and censoring rate, respectively. For example, "small sample size" is equivalent to $n_{obs} = 10$ and "high censoring rate" is equivalent to $R = 0.50$.

We begin by generating draws from a specified distribution of size n_obs, and then determine our censored values by arranging the uncensored observations in ascending order, and then setting those in the lowest $100R$ % to be censored. If the censoring rate were to be $R = 0.10$, the lowest 10% of the observations will be marked as censored while the rest remain uncensored.

2.3 Estimands

Each of the four methods discussed in the previous chapter are designed for usage in obtaining summary statistics for left censored data (Shoari, 2018). In our simulation study, we will be using the sample mean as an estimator for our estimand, the population mean, μ.

2.4 Performance Measures

Morris, White, & Crowther (2019) define performance measures as numeric metrics used to assess the performance of the method in question. The criterion we will use to assess the performance of each of our four methods will consist of: bias, variance, and mean squared error (MSE).

2.4.1 Variance

Prior to defining variance, we must discuss the concept of *precision*. Precision simply refers to how far away estimates from different samples are from one another. Low precision indicates that the estimates from each sample are far from one another in value, while high precision indicates the opposite. Knowing this, *variance* is a metric which informs us on the precision of an estimator. It is defined as the average squared deviation of the estimator from its average:

$$Variance = E[(\hat{\mu} - E(\hat{\mu}))^2]$$

Estimators with low variances generally remain close in value throughout all samples, while those with high variance may wildly differ between samples. It is generally preferable to have an estimator with low variance. Precision measurements,

such as the variance, are not a sole indicator of an estimator's performance (Walther & Moore, 2005). While useful for assessing how close values are to one another, it is just as important to obtain the estimator's *bias*, a measure of how close the obtained estimate is to the true value.

2.4.2 Bias

The next performance metric which we will use is bias, which is defined as the difference between an estimator's expected value and the true value of the parameter. In our case, we are using the estimator $\hat{\mu}$ to estimate the true population mean, μ, in each of our samples. The formal definition of bias is as follows:

$$Bias = E(\hat{\mu}) - \mu$$

Bias informs us on the difference of the estimator from the true parameter. A natural question to ask is often whether if an estimator is any "good." One possible measures of this idea "good," naturally comes in the idea of an unbiased estimator. If the bias of an estimator were to be equal to zero, we would define the estimator to be *unbiased*, meaning that the estimator produces parameter estimates which are on average, equal to the true value.

An estimator being unbiased does not necessarily equate to it being ideal. An unbiased estimator could have high variance, which would mean that the estimator in each sample would be significantly different from one another, but on average – they equal the true population estimand.

On the opposite hand, it would also not be very useful if an estimator had low variance but high bias. This would mean that each sample would consistently produce similar estimates which are very far from the true population estimand.

2.4.3 Mean Squared Error (MSE)

Evidently we do not want our estimator to be too biased nor too variant. This conflict is known as the *bias-variance tradeoff*, a dilemma in which we can never simultaneously minimize the bias and variance of our estimator.

While we generally would like estimators which have low bias and low variance, it can be difficult to achieve both at once. As such, it is common to instead turn to a quantity known as the *mean squared error* (MSE), which is a quantitative measurement used to assess the accuracy of an estimator. The MSE measures how far away, on average, an estimator is from its true value and makes use of both bias and variance in its calculations. The formal definition of MSE is:

$$MSE = E[(\hat{\mu} - \mu)^2] = Var(\hat{\mu}) + [Bias(\hat{\mu})]^2$$

We can show that the MSE of estimator can be rewritten in terms of its variance and bias:

$$E[(\hat{\mu} - \mu)^2] = E(\hat{\mu}^2) + \mu^2 - 2E(\hat{\mu})\mu$$

From $Bias = E(\hat{\mu}) - \mu$, it follows that:

$$Bias^2 = E^2(\hat{\mu}) + \mu^2 - 2E(\hat{\mu})\theta$$

From $Variance = E[(\hat{\mu} - E(\hat{\mu}))^2] = E(\hat{\mu}^2) - E^2(\hat{\mu})$, we can combine the square of the bias with variance, which yields:

$$Bias^2 + Var = [E^2(\hat{\mu}) + \mu^2 - 2E(\hat{\mu})\theta] + [E(\hat{\mu}^2) - E^2(\hat{\mu})]$$

The $E^2(\hat{\mu})$ terms cancel out, and we are left with:

$$E^2(\hat{\mu}) + \mu^2 - 2E(\hat{\mu})\theta = E[(\hat{\mu} - \mu)^2] = Bias$$

As the MSE is always positive, MSE values closer to zero are more desirable – as it is an indicator that the estimator is accurate.

2.5 Results

The results of our simulation study are presented in the following tables below. From the results of our simulation study, we can see that with the data generated from the log-normal distribution, in the case of low censoring (0.10), when dealing with sample sizes of 10 and 100, the methods are largely comparable to one another. Substitution and KM do not perform quite as well as ROS and MLE, both displaying an increase in absolute bias and MSE when compared to the latter two. Substitution performs significant worse than KM. MLE and ROS perform rather equally well in all sample sizes for low-censoring.

When considering medium censoring (0.30), much of the same observations still hold true. Substitution performs the worse, followed by KM. MLE and ROS both perform well. However, ROS has a slight edge over MLE, especially as sample sizes increase, attaining lower MSE values than the latter.

All four methods begin to perform worse when the censoring rate is increased to 0.5, which is to be expected. As more and more missingness is introduced within the dataset, it becomes more difficult to obtain accurate estimates for all methods. Once again, substitution and KM attain high absolute bias and MSE values. However, it is now KM which performs worse than substitution in the setting of high censoring. Similarly to before, albeit being more noticeable now, ROS performs better than MLE with all sample sizes.

Table 2.1: Performance metrics of our our 4 methods with data derived from the log-normal distribution with mean = 1 and SD = 0.5.

	Sample Size	Avg. Mean	Bias	Variance	MSE
Censoring Rate = 0.1					
km	10	1.009	0.00924	0.02289	0.02295
mle	10	0.997	-0.00262	0.02276	0.02275
ros	10	0.991	-0.00923	0.02190	0.02197
substitution	10	0.981	-0.01901	0.02201	0.02235
km	100	1.008	0.00766	0.00263	0.00268
mle	100	0.998	-0.00178	0.00263	0.00263
ros	100	0.999	-0.00142	0.00258	0.00258
substitution	100	0.983	-0.01707	0.00255	0.00284
km	1000	1.009	0.00913	0.00024	0.00033
mle	1000	1.000	-0.00004	0.00024	0.00024
ros	1000	1.001	0.00127	0.00024	0.00024
substitution	1000	0.985	-0.01536	0.00024	0.00047
Censoring Rate = 0.3					
km	10	1.050	0.05015	0.02754	0.03002
mle	10	0.968	-0.03235	0.02431	0.02534
ros	10	0.976	-0.02381	0.02374	0.02429
substitution	10	0.937	-0.06286	0.02298	0.02691
km	100	1.049	0.04904	0.00288	0.00528
mle	100	0.974	-0.02628	0.00252	0.00321
ros	100	1.000	-0.00001	0.00259	0.00259
substitution	100	0.944	-0.05642	0.00241	0.00559
km	1000	1.049	0.04939	0.00026	0.00270
mle	1000	0.974	-0.02554	0.00024	0.00089
ros	1000	1.004	0.00377	0.00024	0.00025
substitution	1000	0.945	-0.05540	0.00022	0.00329
Censoring Rate = 0.5					
km	10	1.148	0.14806	0.03718	0.05907
mle	10	0.908	-0.09188	0.02236	0.03078
ros	10	0.967	-0.03257	0.02810	0.02914
substitution	10	0.908	-0.09229	0.02368	0.03217
km	100	1.129	0.12867	0.00372	0.02027
mle	100	0.904	-0.09649	0.00240	0.01171
ros	100	0.996	-0.00399	0.00316	0.00317
substitution	100	0.904	-0.09614	0.00250	0.01174
km	1000	1.129	0.12885	0.00035	0.01695
mle	1000	0.905	-0.09535	0.00023	0.00932
ros	1000	1.005	0.00515	0.00029	0.00032
substitution	1000	0.905	-0.09484	0.00023	0.00923

Table 2.2: Performance metrics of our our 3 methods (MLE method absent) with data derived from the exponential distribution with a shape parameter = 1.

	Sample Size	Avg. Mean	Bias	Variance	MSE
Censoring Rate = 0.1					
km	10	1.012	0.01191	0.06330	0.06341
mle	10				
ros	10	0.997	-0.00265	0.06173	0.06170
substitution	10	0.992	-0.00761	0.06186	0.06189
km	100	1.010	0.00974	0.00646	0.00656
mle	100				
ros	100	1.005	0.00469	0.00647	0.00648
substitution	100	0.994	-0.00553	0.00649	0.00652
km	1000	1.007	0.00693	0.00067	0.00072
mle	1000				
ros	1000	1.003	0.00308	0.00066	0.00067
substitution	1000	0.992	-0.00798	0.00071	0.00077
Censoring Rate = 0.3					
km	10	1.063	0.06291	0.07325	0.07717
mle	10				
ros	10	0.991	-0.00916	0.06389	0.06394
substitution	10	0.970	-0.02978	0.06372	0.06457
km	100	1.053	0.05270	0.00719	0.00996
mle	100				
ros	100	1.011	0.01140	0.00685	0.00698
substitution	100	0.972	-0.02764	0.00717	0.00793
km	1000	1.054	0.05368	0.00073	0.00361
mle	1000				
ros	1000	1.017	0.01683	0.00085	0.00113
substitution	1000	0.974	-0.02557	0.00152	0.00217
Censoring Rate = 0.5					
km	10	1.193	0.19309	0.10925	0.14648
mle	10				
ros	10	0.992	-0.00789	0.07636	0.07638
substitution	10	0.968	-0.03151	0.07419	0.07514
km	100	1.162	0.16154	0.00992	0.03601
mle	100				
ros	100	1.023	0.02269	0.00792	0.00844
substitution	100	0.961	-0.03925	0.00946	0.01100
km	1000	1.163	0.16301	0.00201	0.02858
mle	1000				
ros	1000	1.036	0.03598	0.00164	0.00293
substitution	1000	0.964	-0.03594	0.00405	0.00534

Table 2.3: Performance metrics of our our 3 methods (MLE method absent) with data derived from the Weibull distribution with a shape parameter = 1 and scale parameter = 1.

	Sample Size	Avg. Mean	Bias	Variance	MSE
Censoring Rate = 0.1					
km	10	1.010	0.00977	0.07703	0.07710
mle	10				
ros	10	0.997	-0.00339	0.07518	0.07516
substitution	10	0.993	-0.00678	0.07525	0.07527
km	100	1.010	0.00998	0.00768	0.00778
mle	100				
ros	100	1.006	0.00626	0.00768	0.00772
substitution	100	0.998	-0.00215	0.00768	0.00768
km	1000	1.006	0.00625	0.00077	0.00081
mle	1000				
ros	1000	1.004	0.00374	0.00077	0.00079
substitution	1000	0.995	-0.00548	0.00081	0.00084
Censoring Rate = 0.3					
km	10	1.067	0.06660	0.08638	0.09078
mle	10				
ros	10	0.996	-0.00440	0.07540	0.07540
substitution	10	0.981	-0.01896	0.07526	0.07559
km	100	1.054	0.05434	0.00845	0.01140
mle	100				
ros	100	1.016	0.01566	0.00804	0.00829
substitution	100	0.982	-0.01761	0.00825	0.00856
km	1000	1.055	0.05451	0.00085	0.00382
mle	1000				
ros	1000	1.021	0.02059	0.00094	0.00136
substitution	1000	0.984	-0.01618	0.00152	0.00178
Censoring Rate = 0.5					
km	10	1.221	0.22073	0.12826	0.17694
mle	10				
ros	10	1.009	0.00898	0.08864	0.08869
substitution	10	0.998	-0.00169	0.08642	0.08639
km	100	1.173	0.17290	0.01172	0.04162
mle	100				
ros	100	1.033	0.03267	0.00934	0.01040
substitution	100	0.980	-0.01968	0.01051	0.01090
km	1000	1.173	0.17294	0.00201	0.03192
mle	1000				
ros	1000	1.045	0.04497	0.00165	0.00367
substitution	1000	0.982	-0.01751	0.00372	0.00402

We can see with the exponential and Weibull cases, all three methods perform equally well in the case of low (0.10) censoring with all sample sizes, obtaining similar bias and MSE values across all sample sizes for both the exponential and Weibull datasets. KM consistently performs the worst with with medium (0.30) and high (0.50) censoring when compared to the other methods across all sample sizes. In these censoring settings, it is also the case that ROS performs the best with substitution not far behind.

In summary, regardless of distributional assumptions all of the methods perform well when censoring is low, with very minute differences in performance metrics. KM does not perform well in the lognormal distribution with high censoring rates and struggles in the exponential and Weibull cases with medium and high censoring rates. While MLE was only used in the case of the lognormal data, it also performs quite well, although not quite as well as ROS. ROS performed the best in the case of medium and high censoring across all 3 distributions.

2.6 Discussion

It important to note that while there are a large number of papers which discuss the ideal method or strategy to handle left-censored data,these studies have a large number of differences in censoring rates, distribution used, methods used, and other aspects of design setups which make comparisons regarding the results obtained from the studies quite difficult. As such, descriptions of specifics regarding the study design in the following studies will be omitted as necessary.

Several results from our simulation studies agree with previous findings conducted from other investigators in the field. Gilliom & Helsel (1986) claims that with the lognormal distribution, the ROS method was superior. This claim is furthered with our

own results: we find that the ROS method is rather robust, even with censoring and produces an accurate and precise estimate of the mean in all cases in our simulation study.

Another investigation by Kroll & Stedinger (1996) found that with regards to a lognormal distribution only, ROS and MLE worked extremely well, with MLE outperforming the other methods especially in highly censored cases. While the MLE method did perform rather well in most cases in our simulation study, it did not outperform the ROS method, which in fact obtained much better estimates of the mean in highly censored settings.

There are of course also studies which offer differing results from the ones we obtained in our simulation study.

Schmoyer, Beauchamp, Brandt, & Hoffman (1996) compared only MLE and KM and found that the KM method performed nearly as well as the MLE in the case of the lognormal distribution. However, this was not the case in our simulation study. While the KM and MLE methods were able to perform adequately in the case of low (0.1) and medium (0.3) censoring, they performed the worst out of all four methods when dealing with highly censored cases. The censoring rates used in their study consisted of 25%, 50%, and 70% – which far exceeded the censoring values used in this thesis.

She (1997) conducted a study investigating censored water quality data with the intent of investigating how well the KM method performs with regards to the same methods we utilize in this thesis. The results from She's study showed that KM outperformed all other methods, which contradicts the findings in the simulation study conducted in this thesis. Upon further investigation, the size of the dataset utilized by She consisted of 56 observations from water monitoring stations, in which around 11 were censored (20% of the dataset). While the KM method is not ideal for highly

censored cases from the results of our simulation study, it certainly is able to be used for smaller sample sizes – which is further supported with She's findings. There may be a difference in how well the methods perform with actual data as compared to simulated data – which we will investigate in the next chapter.

2.6.1 Limitations

Shortcomings in the results presented in this study may come from the fact that we generated data with known distributional parameters. It could be the case that the effectiveness of our methods were only due to having such artificial data. Alterations in our study to instead generate data from methods such as randomized pulls from an a real-world dataset of interest via. methods such as bootstrapping could provide different insights. As we discussed previously with the results from She (1997), methods may perform differently when utilized with artificial datasets as compared to real world, left censored data. We do not claim our findings in the simulation study to be representative for all cases of left censored data.

2.7 Study on Real Data

[can write this section after some preliminary exploration with coal groundwater data]

Chapter 3 Case Study

3.1 Background

Coal is one of the most prevalent combustible fuels being utilized all over the world, as it is one of the easiest methods of obtaining energy due to the abundance of the substance. Generally, coal plants produce electricity by burning coal, which produces coal ash as a byproduct. Over 100 million tons of coal ash are produced every year at these plants, which are then disposed through landfills and waste ponds at these plants. The main concern of ecologists regarding this matter is that the coal ash produced by these plants can often contaminate the local groundwater, leading to toxic contaminants being found in local water sources. Coal ash is dangerous due to its composition, which contains a long list of dangerous chemicals including – but not limited to: arsenic, radium, boron, and other contaminants which have been found to be toxic to humans and animals alike (Kelderman et al., 2019).

Only recently has there been an increase in the frequency of complaints and concerns regarding the disposing practices of coal plants. This is due to disturbances at coal plants, such as the 2010 Kingston Fossil Plant coal ash incident in Tennessee. This area has become an attractive location in which many sites of ecological studies have been conducted the years following the incident. Leaching experiments conducted by Ruhl, Vengosh, Dwyer, Hsu-Kim, & Deonarine (2010) has revealed significant levels of dissolved arsenic, boron, strontium, and barium in the water which has been

in contact with the coal ash, which they specifically note to be threat to aquatic life in the surrounding area. Prompted by environmental organizations, groups, and individuals alike, an onslaught of pressure was put on the Environmental Protection Agency, which resulted in the Coal Ash Rule being put into effect in 2015 (Kelderman et al., 2019).

This rule has forced over 265 coal plants – about 3/4 of all coal plants in the US – to make data regarding chemical concentrations publicly available to the general population. In their analysis using this data, the Environmental Integrity Project (2020), a non-profit organization dedicated to environmental justice issues, have discussed the prevalence of groundwater contamination for in the wells located in these coal related facilities.

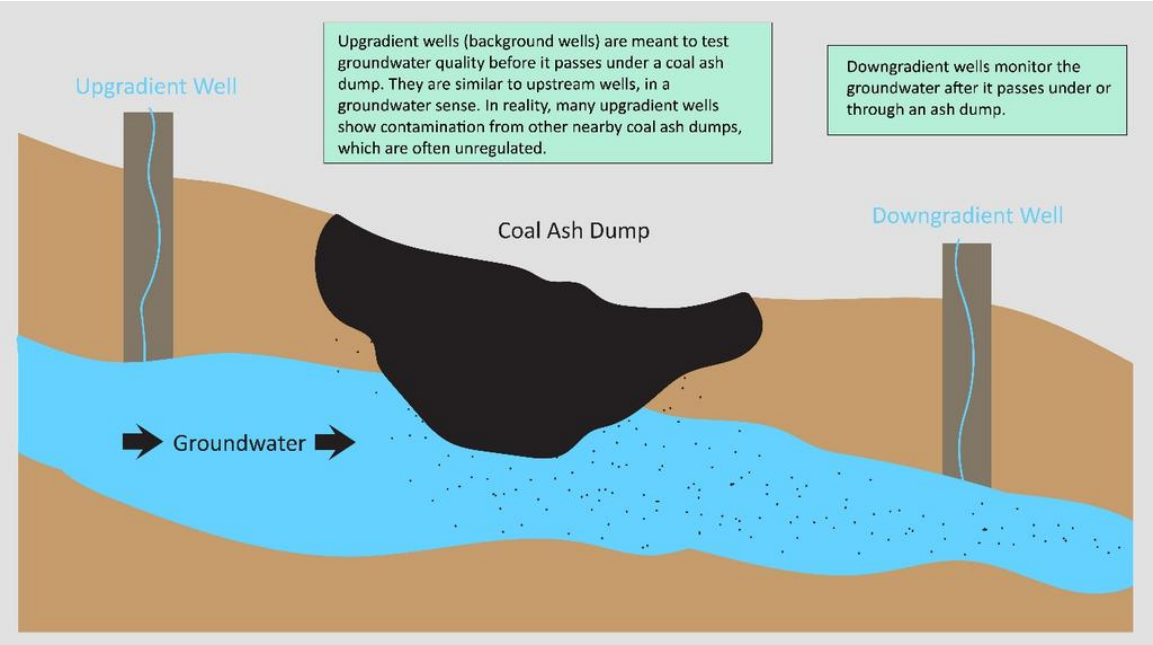

Figure 3.1: Difference Between Upgradient and Downgradient Wells

Typically in a coal ash plant, there exists two types of wells: upgradient wells and downgradient wells. These wells are essential to measure the amount of contamination being caused by coal ash. Upgradient wells, also known as background wells, measures

the concentrations of chemicals in groundwater before it passes through an coal ash dump. Conversely, downgradient wells measure the concentrations of chemicals in groundwater after it passes through a coal ash dump. Figure 3.1 is a useful visualization detailing the specifics regarding the differences between the two types of wells. While both types of well are susceptible to contamination through coal ash related means, it is more frequently the case that we focus on the concentrations of downgradient wells as they are good indicators of possible contamination in the water being accessed by the general public.

The goal of the study conducted by Kelderman et al. (2019) was to identify the percentage of coal plants which have unsafe levels of contamination. High concentrations of toxic chemicals are what classify a well as being contaminated or not. More specifically, we obtain the mean concentrations of the contaminant in question for all the wells measuring a specific chemical in a site. If the mean contamination of a contaminant in question, say, arsenic, was above the health-based thresholds set by the EPA, then that well would be marked as an "exceedance," and deemed to be contaminated.

Kelderman et al. (2019) notes the possibility of contamination being caused by an external factor, unrelated to the coal ash, and provides a stipulation as how to account for this. In the process of calculating mean concentrations, they excluded wells in which the mean downgradient values were lower than the mean upgradient values, as this would mean that the contamination was not caused by the coal plant itself. They also exclude upgradient wells in this calculation, as the focus is on possible contamination in downgradient well in the coal plant.

3.2 Data

It is important to note where the data used originated from, before we delve into the details of our case study. As such, a brief history regarding the Coal Ash Rule and its origin will be explored, alongside details regarding our coal ash dataset.

3.2.1 Coal Ash Rule

A large coal ash spill at the Tennessee Valley Authority which occurred on December 22, 2008 in Kingston, TN – prompted the Environmental Protection Agency (EPA) to propose a set of standardized regulations and procedures to address the concerns regarding coal ash plants nationwide in the US. This was known as the Coal Ash Rule, which passed legislation on December 19, 2014 (Environmental Protection Agency, 2020). Over the years, several changes were made to the Coal Ash Rule in the form of amendments. One of these amendments (published on the April 15, 2015) stated that coal plants would be required to publish data regarding the concentrations of contaminants in the wells and other facility information to the general public.

3.2.2 Source of Data

The data used in the study are from a collection of results published by each coal plant in their "Annual Groundwater Monitoring and Corrective Action Reports." These reports are coal-specific, in PDF format, and can often be up to thousands of pages long, which makes it difficult for individuals to parse through data in a meaningful way. Due to this inaccessibility, the Environmental Integrity Project (2020) began a long project to parse and wrangle through these reports to compile them into a more accessible machine-readable format. This compilation contains information from over 443 annual groundwater monitoring reports posted by 265 coal ash plants, and is

downloadable from the EIP's website. This dataset compiled by the Environmental Integrity Project are what we will utilize in our case study.

3.2.3 Variables

The coal dataset contains information regarding chemical concentrations at coal plants. A coal plant consists of multiple disposal areas for the coal ash that it produces. At each disposal area, there are specific locations that groundwater is being measured, known as wells, which represent an observation in the dataset. There are over 265 coal plants, also known as "sites," in this dataset. A single site is divided into multiple subsections, known as "disposal areas." Each well can be associated with a disposal area and subsequently, a site.

Specifics regarding the variables in the coal dataset can be viewed in Table 3.1. Each observation in the dataset represents a well which measures the concentration of the contaminant in question. Most of the variables are explanatory, such as the state, site, and disposal area in which the well is located in. However, there are several variables specific to groundwater data collection which are important to note.

There are four different types that each well can be classified as, which is represented in the `type` variable. These consist of: `L`, `M`, `SI`, and `U` which stand for "landfill," "mixed," "surface-impacted," and "unknown." We have already mentioned downgradient and upgradient wells during our discussion of the coal ash rule, but we will provide more detail now. In a groundwater monitoring system it is common to have designated wells for specific purposes. A common approach in a coal site is to have separate wells, upgradient wells, whose purpose is to measure "natural" water conditions and downgradient wells, which measures water conditions after it passes through a coal ash disposal area.

Coal plants may also follow different reporting protocols, which necessitates the

"measurement unit" column. While some contaminants such as radium, are measured only in one unit (pCi/L) – most others are measured differently across sites. One site may measure arsenic with using milligrams/L while another site uses micrograms/L.

The remaining variables in our dataset are mostly self explanatory, containing information regarding the date when sample was collected, unique ID of the well, and whether if the measurement is below the limit of detection. A data dictionary of all variables in the dataset can be viewed in Table 3.1.

3.2.4 Plan of Action

The investigation conducted by Kelderman et al. (2019) mentions certain restrictions within the data that we believe may have caused their analysis to potentially be inaccurate. The limit of detection problem arises when measuring devices used to measure chemical concentrations are unable to detect below a certain threshold, causing large numbers of observations to be considered "below detection." These values are often encoded as NA or even mistakenly marked as 0.

Our end goal remains the same as the original research question proposed by Kelderman et al. (2019), which is to identify (the top ten most) contaminated coal plants. Around 2/3 of all wells in the dataset have concentrations found to be below the detection limit. This is a significant portion of the data being censored, which we believe may have significant consequences in the results obtained during analysis. Kelderman et al. (2019) handled these censored values by assuming that their concentration was one half of the detection limit. In essence, they employed the substitution method we discussed previously, with a replacement value of $\frac{1}{2}$ LOD for the values below the detection limit.

The goal of our case study is to employ the techniques we introduced back in chapter 2 to see if they would result in potential differently conclusions. Specifically,

Table 3.1: Data dictionary for the coal dataset.

Variable	Variable Name	Description
State	state	The state where the site is located.
Site	site	The name of the site as it is presented in its groundwater monitoring report.
Disposal Area	disposal.area	The name of the disposal area(s) as they are presented in the groundwater monitoring report. Note: some wells monitor groundwater from more than one disposal unit.
Type of Well	type	The type of disposal unit. SI = surface impoundment, L= landfill, M = mixed multi-unit (landfill and surface impoundment), and U = unknown.
ID of Well	well.id	The identifier given to each monitoring well in the groundwater monitoring report.
Gradient Type	gradient	The location of the groundwater monitoring well relative to the regulated ash disposal unit it monitors.
Sample Date	samp.date	The date the well was sampled.
Contaminant Name	contaminant	The contaminant name. These have been standardized to allow for analyses across plants.
Measurement Unit	measurement.unit	The concentration units. These include mg/l, ug/l, pCi/l, and standard units (SU) for pH.
Below Detection	below.detection	LOD status for the concentration, '<' indicates that the concentration was below the LOD.
Concentration	concentration	The concentration of the contaminant.

we wish to check if the proportion of wells in the U.S. in which contamination is present would be altered if we used our mean estimation techniques to calculate the average concentrations of the contaminants. We would also like to implement a baseline (control) method in which we calculate mean estimates while disregarding the censoring status of the observations and see if our methods truly offer any different conclusions than the claims made by Kelderman et al. (2019) regarding the top 10 most contaminated wells in the U.S.

3.3 Application

3.3.1 Background (tentative title)

As we can see from Figure 3.2, out of the 265 sites in our dataset, most are concentrated heavily in the mid-western and southern areas of the United States. The report written by Kelderman et al. (2019) pointed out that 91% of these sites (242 sites, to be precise) had groundwater wells with contaminants at an unsafe level determined by the health-based threshold put out by the EPA.

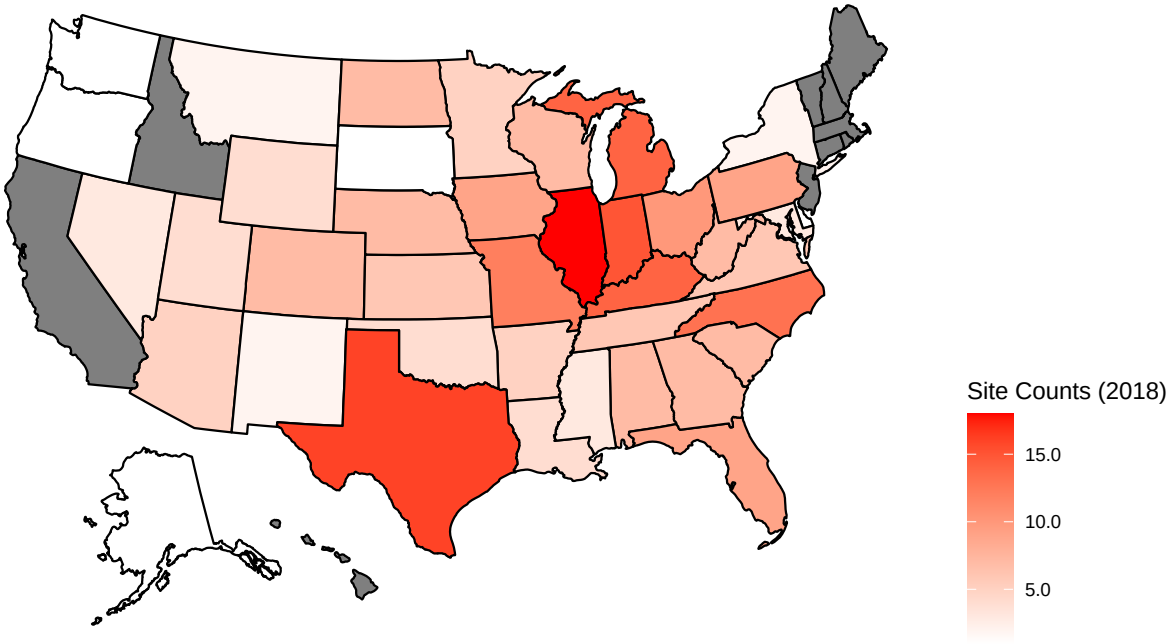

Figure 3.2: Counts of Coal Sites in the United States (gray indicates no sites for that state).

3.3.2 Kelderman's Top 10 Sites

As stated previously, Kelderman et al. (2019) compiled a list of the top ten most contaminated sites across the U.S., following the EPA's health-based threshold for specific contaminants. We will briefly follow with a discussion on their methodology in determining these top ten most contaminated sites.

First, they calculated the average mean concentration for all contaminants in each well across all dates. To ensure that all observations used in this calculation were attributable to the coal site in question, there are several guidelines that were followed.

After these values are obtained, any wells with average downgradient concentrations

Table 3.2: Health-based thresholds set by EPA.

Contaminant	Exceedance Limit	Unit
Antimony	6	ug/L
Arsenic	10	ug/L
Barium	2	mg/L
Beryllium	4	ug/L
Boron	3	mg/L
Cadmium	5	ug/L
Chromium	100	ug/L
Cobalt	6	ug/L
Fluoride	4	mg/L
Lead	15	ug/L
Lithium	40	ug/L
Mercury	2	ug/L
Molybdenum	40	ug/L
Radium	5	pCi/L
Selenium	50	ug/L
Sulfate	500	mg/L
Thallium	2	ug/L

lower than the highest average upgradient concentration for that specific contaminant and disposal area were removed. It is also important to note that since upgradient wells measure the quality of the water *before* it passes through a coal site, they are also removed.

These two exclusion principles are followed to ensure that we did not use any wells that potentially had contamination from an external source, apart from the coal site. The remaining average mean concentrations of the contaminants in question for each site are then compared to the health-based thresholds set by the EPA, these thresholds can be viewed in Table 3.2. For each site, the wells with the highest

average concentration of each contaminant were identified and compared to their respective health-based thresholds. A ratio is then calculated for each of these highest average contaminants to the health-based thresholds. These contaminant-specific ratios are then summed together in order to get a cumulative "contamination score" for each site. Contamination scores of a higher magnitude indicates more "severe" contamination, with the top ten highest contamination scores comprising the top ten most contaminated sites.

While the contamination scores are not explicitly published in Kelderman et al. (2019)'s report, the site names of the top ten most contaminated sites in the country according to their analysis which in descending order severity are as follows:

> 1. San Miguel Plant,
> 2. Allen Steam Station
> 3. Jim Bridger Power Plant
> 4. Naughton Power Plant
> 5. New Castle Generating Station
> 6. Allen Fossil Plant
> 7. Brandywine Ash Management Facility
> 8. Hunter Power Plant
> 9. R.D. Morrow, Sr. Generating Station
> 10. Ghent Generating Station

We will follow their criterion(s) for determining if a well is contaminated, with the only difference being how we obtain the average concentrations of the contaminants in the wells. We will use our several left-censored estimation techniques to obtain mean estimates of the concentrations while accounting for the values below the limit of detection and see if our results significantly differ from the top ten list of most contaminated sites obtained by Kelderman et al. (2019). We will also have a baseline implementation of the methods described by Kelderman et al. (2019), without any attempts to account for the censored data. This implementation will serve as our control in which we will compare our left-censored estimation techniques towards.

Table 3.3: Top 10 most contaminated sites for our baseline (control) implementation.

Site	Composite Score
San Miguel Plant	940
Allen Steam Station	565
New Castle Generating Station	441
Brandywine Ash Management Facility	422
R.D. Morrow, Sr. Generating Station	402
Allen Fossil Plant	368
Hunter Power Plant	345
Naughton Power Plant	273
Jim Bridger Power Plant	266
Sebree Generating Station	266

3.3.3 (Our) Top Ten Most Contaminated Sites

The results from our baseline implementation which calculates mean estimates without any attempt to account for censoring, obtains the top ten sites presented in Table 3.3. When comparing this to the original substitution method using 1/2(LOD) implemented by Kelderman et al. (2019), 9 out of the 10 sites are shared, with the only major difference being the order in which the sites are presented. As the composite scores of the sites were not originally published in Kelderman et al. (2019)'s report, we are unable to delve into more of the specifics regarding the magnitude of difference between the two implementations. As such, we found it prudent to perform their analysis using the substitution method, the results of which are presented in Table 3.4.

Something unexpected to note is that although we followed the procedures outlined by Kelderman et al. (2019) in their implementation of the substitution method, our top ten sites presented in Table 3.5 somewhat differ from theirs. Reproducible code

Table 3.4: Top 10 most contaminated sites for our substitution method implementation.

Site	Composite Score
San Miguel Plant	939
Allen Steam Station	565
New Castle Generating Station	441
Brandywine Ash Management Facility	419
R.D. Morrow, Sr. Generating Station	403
Allen Fossil Plant	367
Hunter Power Plant	345
Naughton Power Plant	272
Jim Bridger Power Plant	265
Sebree Generating Station	264

Table 3.5: Top 10 most contaminated sites for our KM method implementation.

Site	Composite Score
San Miguel Plant	941
Allen Steam Station	565
New Castle Generating Station	441
Brandywine Ash Management Facility	422
R.D. Morrow, Sr. Generating Station	404
Allen Fossil Plant	368
Ghent Generating Station	363
Hunter Power Plant	345
Naughton Power Plant	274
Jim Bridger Power Plant	265

was not provided in their report and attempts to completely reproduce their top ten list would not be conducive to the goal of this report. As such, from now on, we will proceed with the analysis with *our* top ten list produced by the substitution method as the norm, for the sake of comparability with our other methods.

It is also important to note that the MLE method and the ROS method are absent from this case-study. It can be recalled that in our previous discussion regarding the MLE method, we discussed noted a limitation of the MLE method being how it can obtain overinflated estimates when the data is highly skewed. In our attempt to use the MLE method, we found that the estimates we obtained did not provide very useful information. To exemplify this, we had one well with an abnormally large concentration, a sign that it was most likely contaminated, and hundreds of smaller wells in which the concentration is 0. The MLE method gave us mean estimates nearing infinity, which threw our analysis into disarray. As such, we refrained from utilizing the MLE method in this specific case study.

The ROS method was absent for a similar reason, but with the limitations more-so being on the capabilities of the method itself. As we discussed previously, the ROS method can only be used in settings when more than half the data is uncensored. Unfortunately with our data, censoring far exceeds 50%, and as such, the ROS method is unable to be implemented (Environmental Protection Agency, 2009).

The top ten sites obtained using our substitution implementation are the same sites as the ones obtained from our baseline implementation, with the same ordering but slight differences in the composite scores. With our KM method, the Ghent Generating Station replaces the Sebree Generating Station as one of the top ten most contaminated sites, with some slight differences in the ordering of the sites in the latter half of the list.

It is not a surprise that our left-censoring mean estimation techniques did not

provide much of a different result than what we obtained with our baseline implementation. Left-censoring techniques are not as useful with this case-study, where we are focused on finding the extremities of the dataset, i.e. wells with the highest concentration of contaminants. If the research question was instead, identifying wells that were potentially contaminated, the distinction between the results obtained by these methods would be more visible. Kelderman et al. (2019)'s implementation of the substitution method as a way to account for left-censoring is justifiable due to its ease of implementation and the results we just discussed, regarding the lack of differences in top ten sites when comparing all three implementations we used.

Chapter 4 Conclusion

[write a few paragraphs to wrap up entire thesis]

Appendix A Main Appendix

This first appendix includes all of the R chunks of code that were hidden throughout the document (using the `include = FALSE` chunk tag) to help with readibility and/or setup.

A.1 In the main file ??:

A.2 In Chapter ??:

A.3 In Chapter 3:

```
#breaking apart into different datasets for each region
northeast <- import_df %>%
  filter(state %in% c("ME", "NH", "VT", "NY", "PA", "NJ", "MD",
                      "MA", "DE", "RI", "CT")) %>%
  mutate(region = "northeast")

midwest <- import_df %>%
  filter(state %in% c("OH", "IN", "MI", "IL", "WI", "MN", "IA",
                      "MO", "ND", "SD", "NE", "KS"))%>%
  mutate(region = "midwest")

west <- import_df %>%
  filter(state %in% c("WA", "MT", "OR", "ID", "WY", "CA", "NV",
                      "UT", "CO", "AZ", "NM", "AK", "HI")) %>%
  mutate(region = "west")
```

```r
south <- import_df %>%
  filter(state %in% c("WV", "VA", "KY", "TN", "NC", "SC", "GA",
                      "FL", "MS", "AL", "LA", "AR", "OK", "TX", "PR")) %>%
  mutate(region = "south")

#rejoin them back together for future ref. if needed
full <- list(northeast, midwest, west, south) %>%
  reduce(full_join) %>%
  select(-c("qualifier", "link"))

#getting # of sites in each state

midwest_n <- midwest %>%
  group_by(state) %>%
  distinct(site) %>%
  summarize(n = n())

northeast_n <- northeast %>%
  group_by(state) %>%
  distinct(site) %>%
  summarize(n = n())

south_n <- south %>%
  group_by(state) %>%
  distinct(site) %>%
  summarize(n = n())

west_n <- west %>%
  group_by(state) %>%
  distinct(site) %>%
  summarize(n = n())

states_n <- rbind(midwest_n, northeast_n, south_n, west_n)

state_name <- state.name
state_abb <- state.abb
states_map <- map_data("state")
```

Appendix B Simulation and Case Study Appendix

The second appendix, Appendix B, contains all necessary code required to run the simulation study and for our case study.

B.1 Simulation Study

B.1.1 Libraries

```
library(tidyverse)
library(Metrics) #package to help calculate mse
library(NADA) #package with implementation of many methods
library(survival)
library(kableExtra)
```

B.1.2 Generating Data

```
#function will generate a vector of numbers from the lognormal
#distribution and censor them at the given rate
#function will take in arguments for 1) samplesize, 2) logmean, 3)logsd
#4) censoring rate

generateLN <- function(sampsize, m, s, censrate){
  true.value <- rlnorm(sampsize,
                meanlog=log(m^2 / sqrt(s^2 + m^2)),
```

```
                sdlog=sqrt(log(1 + (s^2 / m^2))))

  uncensored_df <- as.data.frame(true.value) %>%
    arrange(true.value)

  censored_df <- uncensored_df %>% #take the head(%) of data to be censored
    slice_head(n=nrow(uncensored_df)*censrate) %>%
    mutate(censored = TRUE)

  #full join original df and sliced df
  return_df <- full_join(uncensored_df, censored_df, by = "true.value")

  #replace NAs with FALSE
  return_df$censored <- replace_na(return_df$censored, replace = FALSE)

  return(return_df)
}
#function will generate a vector of numbers from the exponential
#distribution and censor them at the given rate
#function will take in arguments for 1) samplesize, 2) rate,
#3) censoring rate

generateEXP <- function(sampsize, r, censrate){
  true.value <- rexp(sampsize, rate = r)

  uncensored_df <- as.data.frame(true.value) %>%
    arrange(true.value)

  censored_df <- uncensored_df %>% #take the head(%) of data to be censored
    slice_head(n=nrow(uncensored_df)*censrate) %>%
    mutate(censored = TRUE)

  #full join original df and sliced df
  return_df <- full_join(uncensored_df, censored_df, by = "true.value")

  #replace NAs with FALSE
  return_df$censored <- replace_na(return_df$censored, replace = FALSE)

  return(return_df)
}
#function will generate a vector of numbers from the Weibull
#distribution and censor them at the given rate
```

```r
#function will take in arguments for 1) samplesize, 2) rate,
#3) censoring rate

generateW <- function(sampsize, sh, sc, censrate){
  true.value <- rweibull(sampsize, shape = sh, scale = sc)

  uncensored_df <- as.data.frame(true.value) %>%
    arrange(true.value)

  censored_df <- uncensored_df %>% #take the head(%) of data to be censored
    slice_head(n=nrow(uncensored_df)*censrate) %>%
    mutate(censored = TRUE)

  #full join original df and sliced df
  return_df <- full_join(uncensored_df, censored_df, by = "true.value")

  #replace NAs with FALSE
  return_df$censored <- replace_na(return_df$censored, replace = FALSE)

  return(return_df)
}
```

B.1.3 Setup

```r
iterations <- 1000 #number of iterations
censvalues <- c(0.10, 0.30, 0.50)
sampsizes <- c(10, 100, 1000)

df.tall <- data.frame(prop_cens = numeric(),
                      samplesize = numeric(),
                      iteration = numeric(),
                      method = character(),
                      true_mean = numeric(),
                      mean_complete = numeric(),
                      mean_method = numeric(),
                      true_sd = numeric(),
                      SE_complete = numeric(),
                      SE_method = numeric())
```

B.1.4 Lognormal

```
#LOGNORMAL
options(scipen=999) #prevent scientific notation
set.seed(7271999)

for(i in censvalues){
  for(j in sampsizes){
    for(k in 1:iterations){
      m <- 1
      s <- 0.5
      df <- generateLN(sampsize = j, m = 1, s = 0.5, censrate = i)

      #substitution
      #define LOD to be smallest, uncensored value
      LOD <- min(df$true.value[df$censored == FALSE])
      df <- df %>%
        mutate(impSubValue = if_else(censored == TRUE, LOD/2, true.value))

      df.tall <- df.tall %>%
        add_row(prop_cens = i,
                samplesize = j,
                iteration = k,
                method = "substitution",
                true_mean = m,
                mean_complete = mean(df$true.value),
                mean_method = mean(df$impSubValue),
                true_sd = s,
                SE_complete =
                  sd(df$true.value)/sqrt((length(df$true.value))),
                SE_method =
                  sd(df$impSubValue)/sqrt((length(df$impSubValue))))

      #mle
      mle_res = cenmle(df$true.value, df$censored)

      df.tall <- df.tall %>%
        add_row(prop_cens = i,
                samplesize = j,
                iteration = k,
                method = "mle",
```

```
                true_mean = m,
                mean_complete = mean(df$true.value),
                mean_method = mean(mle_res)[1],
                true_sd = s,
                SE_complete =
                  sd(df$true.value)/sqrt((length(df$true.value))),
                SE_method = mean(mle_res)[2])

    #km
    km_res = cenfit(df$true.value, df$censored)

    df.tall <- df.tall %>%
      add_row(prop_cens = i,
                samplesize = j,
                iteration = k,
                method = "km",
                true_mean = m,
                mean_complete = mean(df$true.value),
                mean_method = mean(km_res)[[1]],
                true_sd = s,
                SE_complete =
                  sd(df$true.value)/sqrt((length(df$true.value))),
                SE_method = mean(km_res)[[2]])

    #ros
    ros_res = ros(df$true.value, df$censored)

    df.tall <- df.tall %>%
      add_row(prop_cens = i,
                samplesize = j,
                iteration = k,
                method = "ros",
                true_mean = m,
                mean_complete = mean(df$true.value),
                mean_method = mean(ros_res),
                true_sd = s,
                SE_complete =
                  sd(df$true.value)/sqrt((length(df$true.value))),
                SE_method =
                  sd(ros_res)/sqrt((length(df$true.value))))
  }
  #end of # iterations
```

```
  }
}

#aggregating performance criteria

df.ln <- df.tall %>%
  group_by(prop_cens, samplesize, method) %>%
  summarize(Avg_Mean = mean(mean_method),
            Bias = (mean(mean_method) - true_mean),
            Variance = var(mean_method),
            MSE = mse(true_mean, mean_method)
            ) %>%
  distinct() %>%
  ungroup()
```

B.1.5 Exponential

```
#EXPONENTIAL
options(scipen=999) #prevent scientific notation
set.seed(7271999)

for(i in censvalues){
  for(j in sampsizes){
    for(k in 1:iterations){
      r = 1
      df <- generateEXP(sampsize = j, r = r, censrate = i)

      #substitution
      #define LOD to be smallest, uncensored value
      LOD <- min(df$true.value[df$censored == FALSE])
      df <- df %>%
        mutate(impSubValue = if_else(censored == TRUE, LOD/2, true.value))

      df.tall <- df.tall %>%
        add_row(prop_cens = i,
                samplesize = j,
                iteration = k,
                method = "substitution",
                true_mean = 1/r,
                mean_complete = mean(df$true.value),
```

```
            mean_method = mean(df$impSubValue),
            true_sd = 1/r,
            SE_complete =
              sd(df$true.value)/sqrt((length(df$true.value))),
            SE_method =
              sd(df$impSubValue)/sqrt((length(df$impSubValue))))

#mle
# mle_res = cenmle(df$true.value, df$censored)
#
# df.tall <- df.tall %>%
#   add_row(prop_cens = i,
#           samplesize = j,
#           iteration = k,
#           method = "mle",
#           true_mean = 1/r,
#           mean_complete = mean(df$true.value),
#           mean_method = mean(mle_res)[1],
#           true_sd = 1/r,
#           SE_complete =
#             sd(df$true.value)/sqrt((length(df$true.value))),
#           SE_method = mean(mle_res)[2])

df.tall <- df.tall %>%
  add_row(prop_cens = i,
          samplesize = j,
          iteration = k,
          method = "mle",
          true_mean = NA,
          mean_complete = NA,
          mean_method = NA,
          true_sd = NA,
          SE_complete = NA,
          SE_method = NA)

#km
km_res = cenfit(df$true.value, df$censored)

df.tall <- df.tall %>%
  add_row(prop_cens = i,
          samplesize = j,
          iteration = k,
```

```
                method = "km",
                true_mean = 1/r,
                mean_complete = mean(df$true.value),
                mean_method = mean(km_res)[[1]],
                true_sd = 1/r,
                SE_complete =
                  sd(df$true.value)/sqrt((length(df$true.value))),
                SE_method = mean(km_res)[[2]])

      #ros
      ros_res = ros(df$true.value, df$censored)

      df.tall <- df.tall %>%
        add_row(prop_cens = i,
                samplesize = j,
                iteration = k,
                method = "ros",
                true_mean = 1/r,
                mean_complete = mean(df$true.value),
                mean_method = mean(ros_res),
                true_sd = 1/r,
                SE_complete =
                  sd(df$true.value)/sqrt((length(df$true.value))),
                SE_method =
                  sd(ros_res)/sqrt((length(df$true.value))))
    }
    #end of # iterations
  }
}

#aggregating performance criteria

df.exp <- df.tall %>%
  group_by(prop_cens, samplesize, method) %>%
  summarize(Avg_Mean = mean(mean_method),
            Bias = (mean(mean_method) - true_mean),
            Variance = var(mean_method),
            MSE = mse(true_mean, mean_method)
            ) %>%
  distinct() %>%
  ungroup()
```

B.1.6 Weibull

```
#WEIBULL
options(scipen=999) #prevent scientific notation
set.seed(7271999)

for(i in censvalues){
  for(j in sampsizes){
    for(k in 1:iterations){
      sh = 1
      sc = 1
      df <- generateW(sampsize = j, sh = sh, sc = sc, censrate = i)

      #substitution
      #define LOD to be smallest, uncensored value
      LOD <- min(df$true.value[df$censored == FALSE])
      df <- df %>%
        mutate(impSubValue = if_else(censored == TRUE, LOD/2, true.value))

      df.tall <- df.tall %>%
        add_row(prop_cens = i,
                samplesize = j,
                iteration = k,
                method = "substitution",
                true_mean = sc*gamma(1+(1/sh)),
                mean_complete = mean(df$true.value),
                mean_method = mean(df$impSubValue),
                true_sd = sqrt((sc^2)*(gamma(1+(2/sh)) -
                                         (gamma(1+(1/sh)))^2)),
                SE_complete =
                  sd(df$true.value)/sqrt((length(df$true.value))),
                SE_method =
                  sd(df$impSubValue)/sqrt((length(df$impSubValue))))

      #mle
      # mle_res = cenmle(df$true.value, df$censored)
      #
      # df.tall <- df.tall %>%
      #   add_row(prop_cens = i,
      #           samplesize = j,
      #           iteration = k,
```

```
#             method = "mle",
#             true_mean = 1/r,
#             mean_complete = mean(df$true.value),
#             mean_method = mean(mle_res)[1],
#             true_sd = 1/r,
#             SE_complete =
#               sd(df$true.value)/sqrt((length(df$true.value))),
#             SE_method = mean(mle_res)[2])

df.tall <- df.tall %>%
  add_row(prop_cens = i,
          samplesize = j,
          iteration = k,
          method = "mle",
          true_mean = NA,
          mean_complete = NA,
          mean_method = NA,
          true_sd = NA,
          SE_complete = NA,
          SE_method = NA)

#km
km_res = cenfit(df$true.value, df$censored)

df.tall <- df.tall %>%
  add_row(prop_cens = i,
          samplesize = j,
          iteration = k,
          method = "km",
          true_mean = sc*gamma(1+(1/sh)),
          mean_complete = mean(df$true.value),
          mean_method = mean(km_res)[[1]],
          true_sd = sqrt((sc^2)*(gamma(1+(2/sh)) -
                                   (gamma(1+(1/sh)))^2)),
          SE_complete =
            sd(df$true.value)/sqrt((length(df$true.value))),
          SE_method = mean(km_res)[[2]])

#ros
ros_res = ros(df$true.value, df$censored)

df.tall <- df.tall %>%
```

```
        add_row(prop_cens = i,
                samplesize = j,
                iteration = k,
                method = "ros",
                true_mean = sc*gamma(1+(1/sh)),
                mean_complete = mean(df$true.value),
                mean_method = mean(ros_res),
                true_sd = sqrt((sc^2)*(gamma(1+(2/sh)) -
                                       (gamma(1+(1/sh)))^2)),
                SE_complete =
                  sd(df$true.value)/sqrt((length(df$true.value))),
                SE_method =
                  sd(ros_res)/sqrt((length(df$true.value))))
    }
    #end of # iterations
  }
}

#aggregating performance criteria

df.w <- df.tall %>%
  group_by(prop_cens, samplesize, method) %>%
  summarize(Avg_Mean = mean(mean_method),
            Bias = (mean(mean_method) - true_mean),
            Variance = var(mean_method),
            MSE = mse(true_mean, mean_method)
            ) %>%
  distinct() %>%
  ungroup()
```

B.2 Case Study

B.2.1 Preliminary

```
#breaking apart into different datasets for each region
northeast <- import_df %>%
  filter(state %in% c("ME", "NH", "VT", "NY", "PA", "NJ", "MD",
                      "MA", "DE", "RI", "CT")) %>%
  mutate(region = "northeast")
```

```
midwest <- import_df %>%
  filter(state %in% c("OH", "IN", "MI", "IL", "WI", "MN", "IA",
                      "MO", "ND", "SD", "NE", "KS"))%>%
  mutate(region = "midwest")

west <- import_df %>%
  filter(state %in% c("WA", "MT", "OR", "ID", "WY", "CA", "NV",
                      "UT", "CO", "AZ", "NM", "AK", "HI")) %>%
  mutate(region = "west")

south <- import_df %>%
  filter(state %in% c("WV", "VA", "KY", "TN", "NC", "SC", "GA",
                      "FL", "MS", "AL", "LA", "AR", "OK", "TX", "PR")) %>%
  mutate(region = "south")

#rejoin them back together for future ref. if needed
full <- list(northeast, midwest, west, south) %>%
  reduce(full_join) %>%
  select(-c("qualifier", "link"))

#getting # of sites in each state

midwest_n <- midwest %>%
  group_by(state) %>%
  distinct(site) %>%
  summarize(n = n())

northeast_n <- northeast %>%
  group_by(state) %>%
  distinct(site) %>%
  summarize(n = n())

south_n <- south %>%
  group_by(state) %>%
  distinct(site) %>%
  summarize(n = n())

west_n <- west %>%
  group_by(state) %>%
  distinct(site) %>%
  summarize(n = n())
```

```
states_n <- rbind(midwest_n, northeast_n, south_n, west_n)

state_name <- state.name
state_abb <- state.abb
states_map <- map_data("state")
```

```
#we need to standardize values, some wells report concentration of contaminants

#template to find different units
template <- full %>%
  filter(grepl("Antimony", contaminant)) %>% #select all rows containing __
  group_by(measurement.unit) %>%
  summarize(n = n())

#antimony has units mg/L and ug/L, we need to change all to ug/L
antimony <- full %>%
  filter(grepl("Antimony", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration*1000,
             measurement.unit %in% "ug/l" ~ concentration),
         measurement.unit = "ug/l")

#arsenic has units mg/L and ug/L, we need to change all to ug/L
arsenic <- full %>%
  filter(grepl("Arsenic", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration*1000,
             measurement.unit %in% "ug/l" ~ concentration),
         measurement.unit = "ug/l")

#barium has units  mg/l and ug/l, we need to change all to mg
barium <- full %>%
  filter(grepl("Barium", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration,
             measurement.unit %in% "ug/l" ~ concentration/1000),
         measurement.unit = "mg/l")
```

```
#beryllium has units mg/l and ug/l, we need to change all to ug
beryllium <- full %>%
  filter(grepl("Beryllium", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration*1000,
             measurement.unit %in% "ug/l" ~ concentration),
         measurement.unit = "ug/l")

#boron has units mg/l and ug/l, we need to change all to mg
boron <- full %>%
  filter(grepl("Boron", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration,
             measurement.unit %in% "ug/l" ~ concentration/1000),
         measurement.unit = "mg/l")

#cadmium has units mg/l and ug/l, we need to change all to ug
cadmium <- full %>%
  filter(grepl("Cadmium", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration*1000,
             measurement.unit %in% "ug/l" ~ concentration),
         measurement.unit = "ug/l")

#chromium has units mg/l and ug/l, we need to change all to ug
chromium <- full %>%
  filter(grepl("Chromium", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration*1000,
             measurement.unit %in% "ug/l" ~ concentration),
         measurement.unit = "ug/l")

#cobalt has units mg/l and ug/l, we need to change all to ug
cobalt <- full %>%
  filter(grepl("Cobalt", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration*1000,
```

```
            measurement.unit %in% "ug/l" ~ concentration),
          measurement.unit = "ug/l")

#fluoride has units mg/l and mg/l, we need to change all to mg
fluoride <- full %>%
  filter(grepl("Fluoride", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration,
             measurement.unit %in% "ug/l" ~ concentration/1000),
          measurement.unit = "mg/l")

#lead has units mg/l and ug/l, we need to change all to ug
lead <- full %>%
  filter(grepl("Lead", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration*1000,
             measurement.unit %in% "ug/l" ~ concentration),
          measurement.unit = "ug/l")

#lithium has units mg/l and ug/l, we need to change all to ug
lithium <- full %>%
  filter(grepl("Lithium", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration*1000,
             measurement.unit %in% "ug/l" ~ concentration),
          measurement.unit = "ug/l")


#mercury has units mg/l and ug/l, we need to change all to ug
mercury <- full %>%
  filter(grepl("Mercury", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration*1000,
             measurement.unit %in% "ug/l" ~ concentration),
          measurement.unit = "ug/l")


#molybdenum has units mg/l and ug/l, we need to change all to ug
```

```
molybdenum <- full %>%
  filter(grepl("Molybdenum", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration*1000,
             measurement.unit %in% "ug/l" ~ concentration),
         measurement.unit = "ug/l")


#radium has units mg/l (14), pCi/l (30k+), ug/l (1), we need to change all to pCi
radium <- full %>%
  filter(grepl("Radium", contaminant)) %>%
  filter(measurement.unit %in% "pCi/l")

#selenium has units mg/l (23k+), pCi/l (8), ug/l (14k+), we need to change all to ug
selenium <- full %>%
  filter(grepl("Radium", contaminant)) %>%
  filter(measurement.unit %in% c("mg/l, ug/l")) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration*1000,
             measurement.unit %in% "ug/l" ~ concentration),
         measurement.unit = "ug/l")

#sulfate has units mg/l and ug/l, we need to change all to mg
sulfate <- full %>%
  filter(grepl("Sulfate", contaminant)) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration,
             measurement.unit %in% "ug/l" ~ concentration/1000),
         measurement.unit = "mg/l")

#thallium has units mg/l (23k+), pCi/l (16), ug/l (13k+), we need to change all to ug
thallium <- full %>%
  filter(grepl("Thallium", contaminant)) %>%
  filter(measurement.unit %in% c("mg/l, ug/l")) %>%
  mutate(concentration =
           case_when(
             measurement.unit %in% "mg/l" ~ concentration*1000,
             measurement.unit %in% "ug/l" ~ concentration),
         measurement.unit = "ug/l")
```

```
#recombine back into new standardized df
df_std <- rbind(antimony, arsenic, barium, beryllium, boron, cadmium,
                chromium, cobalt, fluoride, lead, lithium, mercury,
                molybdenum, radium, selenium, sulfate, thallium)
```

```
#calculation of avg mean concentration for all contaminants in each well across

#get average mean conc for all contaminants in each well
avg_control <- df_std %>%
  group_by(well.id, contaminant, measurement.unit, gradient, site) %>%
  summarize(avg_conc = mean(concentration)) %>%
  ungroup()

#get highest avg. upgradient concentrations for each contaminant/site
upgrad_control <- avg_control %>%
  filter(gradient %in% "Upgradient") %>%
  group_by(contaminant, site) %>%
  summarize(upgrad_conc = min(avg_conc)) %>%
  ungroup()

#remove upgradient wells
downgrad_control <- avg_control %>%
  filter(gradient %in% "Downgradient") %>%
  select(c(well.id, contaminant, site, avg_conc))

#remove any wells with average downgradient concentrations lower than the highes

downgrad_control2 <- left_join(downgrad_control, upgrad_control) %>%
  filter(avg_conc > upgrad_conc) #remove downgradient mean concentrations that we


#for each site, identify the well(s) with the highest mean concentration of each
highest_control <- downgrad_control2 %>%
  group_by(site, contaminant) %>%
  slice(which.max(avg_conc)) %>%
  mutate(contaminant = recode(contaminant,
                              "Antimony, total" = "Antimony",
                              "Arsenic, total" = "Arsenic",
                              "Barium, total" = "Barium",
                              "Beryllium, total" = "Beryllium",
                              "Boron, total" = "Boron",
```

```
                                "Cadmium, total" = "Cadmium",
                                "Chromium, total" = "Chromium",
                                "Cobalt, total" = "Cobalt",
                                "Fluoride, total" = "Fluoride",
                                "Lead, total" = "Lead",
                                "Lithium, total" = "Lithium",
                                "Mercury, total" = "Mercury",
                                "Molybdenum, total" = "Molybdenum",
                                "Radium 226+228" = "Radium",
                                "Selenium, total" = "Selenium",
                                "Thallium, total" = "Thallium"))

hbt_case <- hbt %>% #case sensitivity
  rename("contaminant" = "Contaminant",
         "limit" = "Exceedance Limit",
         "unit" = "Unit")

top10_control <- left_join(highest_control, hbt_case, by = "contaminant") %>%
  mutate(ratio = avg_conc/as.numeric(limit)) %>% #calculate the ratios -> 'highest aver
  group_by(site) %>%
  summarize(composite_score = sum(ratio)) %>% #sum together all contaminants for a site
  arrange(desc(composite_score)) %>%
  slice(1:10)
```

```
#calculation of avg mean concentration for all contaminants in each well across all da

#get average mean conc for all contaminants in each well
avg_substitution <- df_std %>%
  mutate(concentration_new = if_else(below.detection %in% "<",
                                     concentration/2, concentration))%>%
  group_by(well.id, contaminant, measurement.unit, gradient, site) %>%
  summarize(avg_conc = mean(concentration_new)) %>%
  ungroup()

#get highest avg. upgradient concentrations for each contaminant/site
upgrad_substitution <- avg_substitution %>%
  filter(gradient %in% "Upgradient") %>%
  group_by(contaminant, site) %>%
  summarize(upgrad_conc = min(avg_conc)) %>%
  ungroup()
```

```r
#remove upgradient wells
downgrad_substitution <- avg_substitution %>%
  filter(gradient %in% "Downgradient") %>%
  select(c(well.id, contaminant, site, avg_conc))

#remove any wells with average downgradient concentrations lower than the highes
downgrad_substitution2 <- left_join(downgrad_substitution, upgrad_substitution) %>
  filter(avg_conc >= upgrad_conc) #remove downgradient mean concentrations that w


#for each site, identify the well(s) with the highest mean concentration of each
highest_substitution <- downgrad_substitution2 %>%
  group_by(site, contaminant) %>%
  slice(which.max(avg_conc)) %>%
  mutate(contaminant = recode(contaminant,
                              "Antimony, total" = "Antimony",
                              "Arsenic, total" = "Arsenic",
                              "Barium, total" = "Barium",
                              "Beryllium, total" = "Beryllium",
                              "Boron, total" = "Boron",
                              "Cadmium, total" = "Cadmium",
                              "Chromium, total" = "Chromium",
                              "Cobalt, total" = "Cobalt",
                              "Fluoride, total" = "Fluoride",
                              "Lead, total" = "Lead",
                              "Lithium, total" = "Lithium",
                              "Mercury, total" = "Mercury",
                              "Molybdenum, total" = "Molybdenum",
                              "Radium 226+228" = "Radium",
                              "Selenium, total" = "Selenium",
                              "Thallium, total" = "Thallium"))

top10_substitution <- left_join(highest_substitution, hbt_case, by = "contaminant"
  mutate(ratio = avg_conc/as.numeric(limit)) %>% #calculate the ratios -> 'highes
  group_by(site) %>%
  summarize(composite_score = sum(ratio)) %>% #sum together all contaminants for
  arrange(desc(composite_score)) %>%
  slice(1:10)
```

```r
get_km_mean <- function(values, censored){
  km = cenfit(values, censored)
  mean(km)[[1]]
}

#calculation of avg mean concentration for all contaminants in each well across all da

#get average mean conc for all contaminants in each well
avg_km <- df_std %>%
  mutate(censored = if_else(below.detection %in% "<",
                                    TRUE, FALSE)) %>%
  group_by(well.id, contaminant, measurement.unit, gradient, site) %>%
  do(summarize(., avg_conc = get_km_mean(.$concentration, .$censored))) %>%
  ungroup()

#get highest avg. upgradient concentrations for each contaminant/site
upgrad_km <- avg_km %>%
  filter(gradient %in% "Upgradient") %>%
  group_by(contaminant, site) %>%
  summarize(upgrad_conc = min(avg_conc)) %>%
  ungroup()

#remove upgradient wells
downgrad_km <- avg_km %>%
  filter(gradient %in% "Downgradient") %>%
  select(c(well.id, contaminant, site, avg_conc))

#remove any wells with average downgradient concentrations lower than the highest aver
downgrad_km2 <- left_join(downgrad_km, upgrad_km) %>%
  filter(avg_conc >= upgrad_conc) #remove downgradient mean concentrations that were lo


#for each site, identify the well(s) with the highest mean concentration of each conta
highest_km <- downgrad_km2 %>%
  group_by(site, contaminant) %>%
  slice(which.max(avg_conc)) %>%
  mutate(contaminant = recode(contaminant,
                              "Antimony, total" = "Antimony",
                              "Arsenic, total" = "Arsenic",
                              "Barium, total" = "Barium",
                              "Beryllium, total" = "Beryllium",
                              "Boron, total" = "Boron",
```

```
                                "Cadmium, total" = "Cadmium",
                                "Chromium, total" = "Chromium",
                                "Cobalt, total" = "Cobalt",
                                "Fluoride, total" = "Fluoride",
                                "Lead, total" = "Lead",
                                "Lithium, total" = "Lithium",
                                "Mercury, total" = "Mercury",
                                "Molybdenum, total" = "Molybdenum",
                                "Radium 226+228" = "Radium",
                                "Selenium, total" = "Selenium",
                                "Thallium, total" = "Thallium"))

top10_km <- left_join(highest_km, hbt_case, by = "contaminant") %>%
  mutate(ratio = avg_conc/as.numeric(limit)) %>% #calculate the ratios -> 'highes
  group_by(site) %>%
  summarize(composite_score = sum(ratio)) %>% #sum together all contaminants for
  arrange(desc(composite_score)) %>%
  slice(1:10)
```

Corrections

A list of corrections after submission to department.

Corrections may be made to the body of the thesis, but every such correction will be acknowledged in a list under the heading "Corrections," along with the statement "When originally submitted, this honors thesis contained some errors which have been corrected in the current version. Here is a list of the errors that were corrected." This list will be given on a sheet or sheets to be appended to the thesis. Corrections to spelling, grammar, or typography may be acknowledged by a general statement such as "30 spellings were corrected in various places in the thesis, and the notation for definite integral was changed in approximately 10 places." However, any correction that affects the meaning of a sentence or paragraph should be described in careful detail. The files samplethesis.tex and samplethesis.pdf show what the "Corrections" section should look like. Questions about what should appear in the "Corrections" should be directed to the Chair.

References

10 Antweiler, R. C. (2015). Evaluation of Statistical Treatments of Left-Censored Environmental Data Using Coincident Uncensored Data Sets. II. Group Comparisons. http://doi.org/10.1021/acs.est.5b02385

Barnard, J., & Meng, X. L. (1999). Applications of multiple imputation in medical studies: From AIDS to NHANES. *Statistical Methods in Medical Research*, *8*(1), 17–36. http://doi.org/10.1191/096228099666230705

Berthouex, P. (1993). A Study of the Precision of Lead Measurements at Concentrations Near the Method Limit of Detection, *65*(5), 620–629.

Bolks, A., DeWire, A., & Harcum, J. B. (2014). Baseline Assessment of Left-Censored Environmental Data Using R. *Technotes*, *9*(2), 153–172. Retrieved from https://www.epa.gov/sites/production/files/2016-05/documents/tech_notes_10_jun2014_r.pdf

Canales, R. (2018). Methods for Handling Left-Censored Data in Quantitative Microbial Risk Assessment, *84*(20), 1–10.

Chen, H., Quandt, S. A., Grzywacz, J. G., Arcury, T. A., Environmental, S., Perspectives, H., ... Arcury, T. A. (2011). A Distribution-Based Multiple Imputation Method for Handling Bivariate Pesticide Data with Values below the Limit of Detection, *119*(3), 351–356. http://doi.org/10.1289/ehp.l002124

Crovelli, R. A. (1993). An Objective Replacement Method for Censored Geochemical Data, *25*(1), 59–80.

Environmental Integrity Project. (2020). Coal Ash Groundwater Contamination: Documenting Coal Ash Pollution. Retrieved from https://environmentalintegrity.org/coal-ash-groundwater-contamination/

Environmental Protection Agency. (2009). *STATISTICAL ANALYSIS OF GROUNDWATER MONITORING DATA AT RCRA FACILITIES UNIFIED GUIDANCE*.

Environmental Protection Agency. (2020). Disposal of Coal Combustion Residuals from Electric Utilities Rulemakings. Retrieved from https://www.epa.gov/coalash/coal-ash-rule

Ganser, G. H., & Hewett, P. (2010). An Accurate Substitution Method for Analyzing Censored Data, (April), 233–244. http://doi.org/10.1080/15459621003609713

Gilbert, R. O. (1987). *Statistical Methods for Environmental Pollution Monitoring*. New York: Van Nostrand Reinhold Co.

Gilliom, R. J., & Helsel, D. R. (1986). Estimation of Distributional Parameters for Censored Trace Level Water Quality. *Water Resources Research*, *22*(2), 135–146. http://doi.org/10.1029/WR022i002p00135

Gllllom, R. J., Kirsch, R. M., Gilroy, E. J., & Survey, U. S. G. (1984). Effect of Censoring Trace-Level Water-Quality Data Capability Trend-Detection, (1), 530–535. http://doi.org/10.1021/es00125a009

Hall, L. W., Perry, E., & Anderson, R. D. (2020). A Comparison of Different Statistical Methods for Addressing Censored Left Data in Temporal Trends Analysis of Pyrethroids in a California Stream. *Archives of Environmental Contamination*

and Toxicology, *79*(4), 508–523. http://doi.org/10.1007/s00244-020-00769-0

Hornung, R., & Reed, L. (1989). Estimation of Average Concentration in the Presence of Nondetectable Values. *Applied Occupational and Environmental Hygiene*, *5*(1), 46–51.

Kelderman, K., Kunstman, B., Roy, H., Sivakumar, N., Mccormick, S., & Bernhardt, C. (2019). Coal's Poisonous Legacy: Groundwater Contaminated by Coal Ash Across the U.S.

Klein, J. P., & Moeschberge, M. L. (2003). *SURVIVAL ANALYSIS: Techniques for Censored and Truncated Data* (2nd ed., pp. 92–104). New York: Springer-Verlag.

Kroll, C. N., & Stedinger, J. R. (1996). Estimation of moments and quantiles using censored data. *Water Resources Research*, *32*(4), 1005–1012. http://doi.org/10.1029/95WR03294

Lafleur, B., Lee, W., Billhiemer, D., Lockhart, C., Liu, J., & Merchant, N. (2011). Statistical methods for assays with limits of detection: Serum bile acid as a differentiator between patients with normal colons, adenomas, and colorectal cancer. *Journal of Carcinogenesis*, *10*, 1–8. http://doi.org/10.4103/1477-3163.79681

Lee, L., & Helsel, D. (2005). Statistical analysis of water-quality data containing multiple detection limits : S-language software for regression on order statistics $, *31*, 1241–1248. http://doi.org/10.1016/j.cageo.2005.03.012

May, R. C. (2012). Estimation Methods for Data Subject to Detection Limits, 82.

Morris, T. P., White, I. R., & Crowther, M. J. (2019). Using simulation studies to evaluate statistical methods. *Statistics in Medicine*, *38*(11), 2074–2102. http://doi.org/10.1002/sim.8086

Ruhl, L., Vengosh, A., Dwyer, G., Hsu-Kim, H., & Deonarine, A. (2010). Environmental Impacts of the Coal Ash Spill in Kingston , Tennessee: An 18-Month Survey, *44*(24), 9272–9278.

Schmoyer, R. L., Beauchamp, J. J., Brandt, C. C., & Hoffman, F. O. (1996). Difficulties with the lognormal model in mean estimation and testing. *Environmental and Ecological Statistics*, *3*(1), 81–97. http://doi.org/10.1007/bf00577325

She, N. (1997). ANALYZING CENSORED WATER QUALITY DATA USING A NON-PARAMETRIC APPROACh, *33*(3).

Shoari, N. (2018). Toward Improved Analysis of Concentration Data : Embracing Nondetects, *37*(3), 643–656. http://doi.org/10.1002/etc.4046

Walther, B. A., & Moore, J. L. (2005). The concepts of bias , precision and accuracy , and their use in testing the performance of species richness estimators , with a literature review of estimator performance, *6*(July).

Yavuz, Y., Tekindal, M. A., & Dog, B. (2017). Evaluating Left-Censored Data Through Substitution , Parametric , Semi-parametric , and Nonparametric Methods : A Simulation Study, 153–172. http://doi.org/10.1007/s12539-015-0132-9