Evaluation of Parameter Estimation Methods to Handle Left-Censored Missingness

Tony Ni

April 28, 2021

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

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 common methods: substitution, maxmimum likelihood, kaplan-meier, and regression on order statistics, which are commonly used to handle left-censored missingness. We follow this discussion by performing a simulation study, designed to compare and contrast the effectiveness of these methods, while taking into account the distribution of the simulated data, censoring rates, and the size of the dataset. Finally, we explore left-censored data in a case-study focusing on coal ash contamination in groundwater wells and use our four methods to obtain mean estimates of contaminant concentrations to identify the top ten most contaminated wells across the U.S.

Acknowledgments

Throughout the writing of this thesis, I have garnered an immense amount of guidance and support from my advisors, professors, and friends.

I would first like to thank my thesis advisor, Brittney Bailey, for her unwavering belief in me and my skills during the writing of this thesis. Her expertise and knowledge in the field has aided me in all stages of my thesis writing process: from the formulation of a research topic, literature search, and the writing of the actual thesis itself. I would not have been able to write my thesis in this period of tumultuous events if it were not for her steadfast support and encouragement, and for that, I cannot express my utmost gratitude.

I would also like to thank all of my professors who have guided me thus far in my academic journey in statistics. I don't think that I would be here writing a thesis, without their teachings, guidance, and support over my four years at Amherst.

I also need to thank my friends who have been alongside me throughout our statistical journey at Amherst: Breanna Richards, Jasper Flint, and Sabir Meah. Our early morning, mid-afternoon, and even late night homework and study sessions, especially after the pandemic, helped keep me sane and motivated in a time where I had trouble keeping hope. Perhaps also just for the good conversation too.

Thank you also to all the people in the Greenway Group over my four years at Amherst who have motivated me in my times of darkness, lifted me up in times of doubt, and cheered me on in all my pursuits.

This has been a wonderful endeavor to pursue and I'm very grateful for all my supporters who have enabled me to write this thesis.

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

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 Three Types of Censoring

Right censoring is a specific instance of censoring 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. This is the most common type of censoring and can often be found in clinical trial studies, mortality studies, and other forms of survival analyses.

In contrast with right censoring, in the case of left censoring, we only know that the true value of a data point falls *below* a certain threshold which we call the *limit of detection* (LOD), and similarly to right censoring, it is unknown by how much. Left censoring is commonly found in environmental, water quality, and chemical-related research where the focus is on the concentration of an analyte.

Right and left-censoring are both special cases of interval censoring. Interval censoring is when 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 present, 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. In the case of left censoring, $t_1 = 0$; and conversely in the case of right censoring, $t_2 = \infty$.

1.1.2 Left Censoring

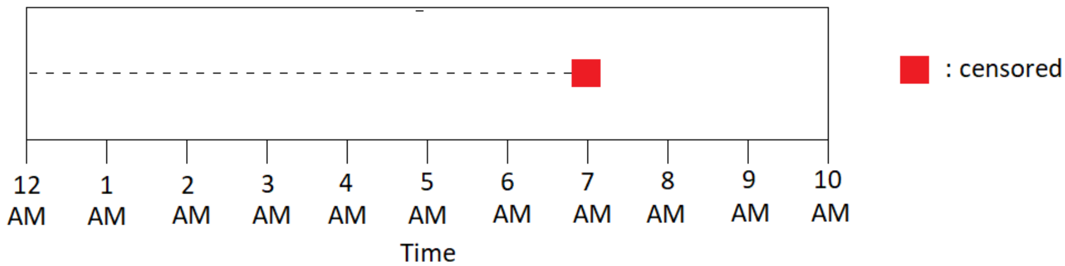

Figure 1.1: Left Censoring Example

To clarify the concept of left-censoring, consider the following example illustrated with Figure 1.1. 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.). As the focus of my thesis, left-censoring poses many challenges in regards to how individuals should report and work with left-censored data.

1.2 Challenges of Reporting Censored Data

The most pressing issue in regards to the reporting practices for left-censored data is the lack of a universal standard as to how to report values below the LOD, which can lead to confusion amongst researchers. This lack of standardization makes it difficult to distinguish values below the LOD and uncensored values, which 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.

1.3 Parameter Estimation for Left-Censored Data

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, Quandt, Grzywacz, & Arcury, 2011). As such, there are a variety of statistical treatments to handle censored data which have been popularized in the statistical literature.

Before we discuss the techniques which have been popularized through literature and studies regarding parameter estimation with left-censored data, it important to discuss *omission*, the deletion of data points which are deemed to be invalid as a result of left-censoring or any other deficiencies in the data. As a result of being simple to comprehend and implement, omission is a common technique used in lieu of specialized techniques designed to handle missing data.

One type of omission is 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.

Over the past century, a myriad of methods to deal with censoring have been developed to counter this issue of discarding data with omission-based techniques – some more statistically sound than others. We will review some of the more common methods to estimate descriptive statistics involving censored data, which include: substitution, maximum likelihood estimation, Kaplan-Meier, and regression on order statistics.

1.3.1 Substitution

The first technique we will discuss is the substitution method, which involves imputing in a replacement value in lieu of the censored data point. As a method commonly condemned in papers as a statistically unsound method to handle censored data, substitution methods are ubiquitous in the chemical and environmental sciences (Canales, 2018).

Substitution techniques are easy to understand and to implement, akin to the omission techniques we discussed previously. Observations for which measurements fall below the LOD are replaced with a replacement value, which is non-specific and can vary between studies. A non-exhaustive list of common replacement values include: $\frac{LOD}{2}$, $\frac{LOD}{\sqrt{2}}$, and the LOD itself (Lee & Helsel, 2005).

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.

Regardless of which replacement value is used, once a dataset is generated containing these replacement values, analysis continues as is, utilizing this new dataset.

As mentioned previously, contention between whether this method is statistically sound or not remains to the present day. The lack of a global, standardized replacement value to substitute is one of the most pronounced downsides of this method. 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). Due to the lack of standardization, many regard substitution techniques as a non-rigorous, statistically unsound method for handling left-censored data (Chen, Quandt, Grzywacz, & Arcury, 2011).

Lee & Helsel (2005) also provides a critique on substitution methods and claim that they can often introduce a "signal" which was not originally present within the data, or even obstruct an actual signal which was originally present in the data – leading to misleading and/or inaccurate results. Supporting Lee & Helsel (2005)'s claim, Glass and Gray (2001) found the substitution method to introduce large errors and biases when calculating descriptive statistics of interest with left-censored data. 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.

1.3.2 Maximum Likelihood Estimation

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

Let the random variables $X_i, \cdots, X_n$ be independent and identically distributed with probability function $f(x_i|\theta)$, where θ is the parameter we are interested in estimating.

For every observed random sample $x_1, \cdots, x_n$, the joint density function is:

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

Our goal is to find the value of θ which is most likely to generate our observed data. In order to solve this inverse problem, we introduce the likelihood function, which is defined as a function of the parameter given the observed data:

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

Our *maximum likelihood estimate* of our parameter θ, then, is the value $\hat{\theta}$ that maximizes the likelihood function, $L(\theta)$.

When left censoring is present, the likelihood function changes in order to account for both the censored observations and the uncensored observations. We define $F(x_i|\theta)$

to the cumulative distribution function for our RVs conditioned on θ. Our new likelihood function when left censoring is present is:

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

where δ_i indicates whether or not the ith observation is censored:

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

It is then possible to follow typical procedures to find the maximum likelihood estimates of the parameteres of interest (mean, variance, etc.) from our censored data.

Yavuz, Tekindal, & Dog (2017) discusses the usage of ML estimation when missing data is present, and notes it is only appropriate for non-negative probability distributions such as the exponential, log-normal, and Weibull models. This is one limitation of ML estimation, it cannot be applied for data which do not fit a specified model – and is very limited in scope.

ML estimation is one of the most well-known parametric approaches to handling left-censored data. Many studies use ML estimation as a baseline method of handling censored values, to which they compare their new techniques (Ganser & Hewett, 2010). Despite its prevalence, ML estimation has its weaknesses. Canales (2018) found that the ML estimation seems to underperform when the data in question was highly skewed, producing overinflated mean squared errors. Additionally, because ML estimation 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). Regardless of these limitations, it is a definite staple in the field of parameter estimation with regards to censored data.

1.3.3 Kaplan-Meier

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 event of interest occurs. This can be as innocuous as the time until a device breaks, time until birds migrate away from their homes, time until a person passes away, etc. In all cases, there is a 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 make assumptions about the underlying distribution of the data. The KM method was originally developed to handle right-censored survival analysis data (Hall, Perry, & Anderson, 2020). The advantages of the KM method lies in its robustness as a nonparametric method – it performs well without having to depend upon distributional assumptions. Many recommend its usage in cases of severe censoring, instances where more than 90% of the data is censored (Canales, 2018).

The KM-estimator is a statistic used to estimate the survival curve from the data while accounting for censoring. 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 (Gillespie et al., 2010).

The KM-estimator of the survival curve at time t is:

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

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 greater than or equal to t_i (how many observations in sample survived until at least time t_i) (Klein & Moeschberge, 2003).

Typically, the KM-estimator is used to estimate the distribution function of right-censored data. Helsel (2005, as cited in Yavuz et al., 2017) provided a simple modification of the KM-estimator to allow for the estimation of the survival curve with left-censored values. In his implementation, he ran the left-censored data through a transformation algorithm before using the KM method to change them into right-censored data.

The transformation algorithm works as follows: First, all the left-censored values in the dataset are arranged in descending order of magnitude. Then these left-censored values are subtracted from M, to get $M - x_i$, the newly transformed, right-censored value (M is a constant bigger than the maximum value in the dataset). Finally, the non-censored values and the newly transformed values are then arranged in ascending order to be used to estimate the survival function through the Kaplan-Meier estimator.

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. She (1997) gives the expressions to calculate the estimated mean, median, and variance:

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

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 distribution (usually lognormal) 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.

A brief explanation as to how ROS works is as follows. 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. Once the censored values are ranked, a linear regression model is fit between the uncensored values and the distributional z-scores from the censored probability plot. The parameters of the regression model (slope and intercept of the regression line) 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.

Delving into the mathematics behind this procedure, we need to first define several variables.

We define k be the number of distinct LOD values in the data, A_i as the number of uncensored values between the ith and $(i+1)$th LOD for $i = 1$ to $k-1$, and A_k as the number of uncensored values above the highest LOD, A_0 as the number of uncensored values below the lowest LOD.

We also define B_i as the total number of observations (both uncensored and censored) with values below the ith LOD and $B_0 = 0$.

We can then write the number of uncensored values below the ith LOD (C_i) as:

$$\text{Eq. 1.1: } C_i = B_i - B_{i-1} - A_{i-1}$$

for $i = 1$ to k.

We can also calculate what is known as the *exceedance probability* (pe_i) for each of our k LOD values, which represents the proportion of the data greater than or equal to each distinct LOD:

$$\text{Eq. 1.2: } pe_i = pe_{i+1} + \frac{A_i}{A_i + B_i}(1 - pe_{i+1})$$

where pe_i is the proportion of the data which exceeds the ith LOD.

Once we obtain exceedance probabilities for each of our distinct LOD values, we compute the plotting positions, aka. the cumulative probabilities, for each of our uncensored values between the ith and $(i+1)$th LOD values with the following equation:

$$\text{Eq. 1.3: } pu_{ij} = (1 - pe_i) + (\frac{j}{A_i + 1}) * (pe_i - pe_{i+1})$$

for $j = 1$ to A_i and for $i = 0$ to k.

We also similarly compute the plotting positions for our censored values with:

$$\text{Eq. 1.4: } pc_{ij} = (\frac{j}{C_i + 1}) * (1 - pe_i)$$

for $j = 1$ to C_i and for $i = 1$ to k.

Having defined the relevant variables and equations, we can detail a step-by-step algorithmic procedure on how the ROS method works.

1. Given a dataset with n observations, identify and sort the k distinct LOD values.

2. Compute A_0, A_k, A_i's for $i = 1$ to $k - 1$, and B_i for $i = 1$ to k.

3. Use Eq. 1.1 to compute C_i for $i = 1$ to k.

4. Let $pe_0 = 1$ and $pe_{k+1} = 0$. Use Eq. 1.2 to compute pe_i for $i = 1$ to k.

5. Sort all pe_i from $i = 1$ to k. Use Eq. 1.3 to compute pu_{ij}.

6. Compute normal quantiles (z-scores) by computing $z^d_{ij} = \Phi^{-1}(pd_{ij})$, where Φ^{-1} is the inverse standard normal CDF.

7. Construct censored probability plots using the z-scores from step 6.

8. Plot z^d_{ij} against a transformation, $f(\cdot)$, (ex: log, square root, inverse, etc.) of the uncensored values x^d_{ij}.

9. Compute correlation coefficient between the z^d_{ij} and $f(x^d_{ij})$ pairs. The transformation which yields the highest correlation coefficient is the one which most optimizes the left-censored values.

10. Compute a linear regression of the transformed values, $f(x^d_{ij})$ against the z-scores, z^d_{ij}.

11. Use Eq. 1.4 to compute pc_{ij} and repeat steps 6 through 8, to obtain z-scores for the censored values.

12. Obtain the transformed imputed values using the slope and intercept from the regression model in step 10 and the censored z-scores from step 11 using the following expression:

$$f(\hat{x}^c_{ij}) = Intercept\ + Slope\ * z^c_{ij}$$

13. Combine the transformed imputed values for the censored values with the transformed uncensored values to obtain estimates for the population mean and standard deviation as normal.

There are of course, also limitations to the ROS method. 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).

We have now covered the techniques that are commonly used to handle left-censored data. These methods can be classified as either imputation-based techniques, in the case of substitution and ROS, or parameter-estimation techniques, in the case of MLE and KM. How then, can we compare the effectiveness of each of these methods?

Several studies have already been conducted over the years to evaluate the performance of different methods for handling left-censored data, with the results being widely varied and largely inconclusive. However, each study varies wildly in the methods being investigated and the scope of the study – some leaning more towards seeking theoretical conclusions while others focus on applications to real-world data, such as water-quality data. For example, Antweiler (2015) evaluates the effectiveness of 11 different methods with several censoring rates and distributional assumptions, using the median absolute deviation as their performance metric of choice. Meanwhile, Hall, Perry, & Anderson (2020) investigated the performance of the four methods used in this thesis while disregarding distributional assumptions and censoring rates, using a water-quality dataset.

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.

Regarding the simulation study that we will conduct in the following chapter, 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, comparing and contrasting the effectiveness of our four techniques with regards to censoring rate and sample size.

Chapter 2 Simulations

Having discussed four common 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 after varying the censoring rate, sample size, and the underlying distribution of the data. We will also discuss the implementation of the methods, data generating mechanisms for the simulation study, and our findings.

2.1 Data-Generating Mechanisms

To reflect the typical distributions of left-censored data, we generated data for use in our simulation study with parametric draws from the log-normal, exponential, and Weibull distributions. We also varied sample sizes, $n = \{10, 100, 1000\}$, and censoring rates, $R = \{0.10, 0.30, 0.50\}$. The sample sizes are chosen as such to reflect realistic sizes for water quality datasets, which we plan to investigate in the following chapter with a case-study on coal ash contamination in water wells. Regarding the values we chose for R, if we recall from the previous chapter, the ROS method is unable to be implemented when more than half the data is censored, which is why we chose our censoring rates as such.

In the following sections of this chapter, we will interchangeably use "small, medium, large" and "low, medium, and high" to differentiate between the values for sample size and censoring rate, respectively. For example, "small sample size" is equivalent to

$n = 10$ and "high censoring rate" is equivalent to $R = 0.50$.

To artificially induce censoring in our simulated data, we arranged the uncensored observations in ascending order and set those in the lowest $100R\%$ to be censored, and the remaining observations as uncensored. As an example, if the censoring rate was $R = 0.10$, the lowest 10% of the observations would be marked as censored while the rest remained uncensored.

2.2 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 preparation for our case study in the following chapter, we will be utilizing the four methods to estimate the mean of the censored variable of interest in our simulation study.

2.3 Implementation in R

All our code will be written using R. For our substitution method, we begin by defining the LOD as the minimum uncensored value for the variable of interest. We will choose LOD/2 as our replacement value, calculating our estimated mean using this newly defined dataset.

For the remaining three methods (KM, MLE, and ROS), we will be using specialized functions from the `NADA` package (Lee, 2020). The code for the MLE method will be handled with the `cenmle` function, which allows the user to specify censored and uncensored data, and uses the LOD as the placeholder. This method allows us to calculate the summary statistics for the entire data set – including the censored values. The `cenfit` function allows us to implement KM. This function "computes an estimate of an empirical cumulative distribution function for censored data using the

Kaplan-Meier method," from which we can then obtain summary statistics of interest. Similarly, the `ros` function implements ROS and outputs a dataframe containing the original uncensored values and the estimates for the censored values, from which we can then use to also compute our summary statistics of interest.

2.4 Performance Measures

The criteria we will use to assess the performance of each of our four methods will consist of: bias, variance, and mean squared error, which measure the difference between our mean estimate and the true mean, precision, and accuracy – respectively.

Bias is calculated by obtaining the difference between the expectation of $\hat{\mu}$ and the true value of μ: $Bias = E(\hat{\mu}) - \mu$. Variance is calculated by obtaining the average squared deviation of the estimator, μ, from its average: $Variance = E[(\hat{\mu} - E(\hat{\mu}))^2]$. The MSE is calculated by: $MSE = E[(\hat{\mu} - \mu)^2] = Var(\hat{\mu}) + [Bias(\hat{\mu})]^2$.

We will run our simulation with 1000 iterations and calculate our aforementioned performance measures using the results from the runs.

2.5 Results

The results of our simulation study are presented in Tables 2.1, 2.2, and 2.3 and Figures 2.1, 2.2, and 2.3.

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

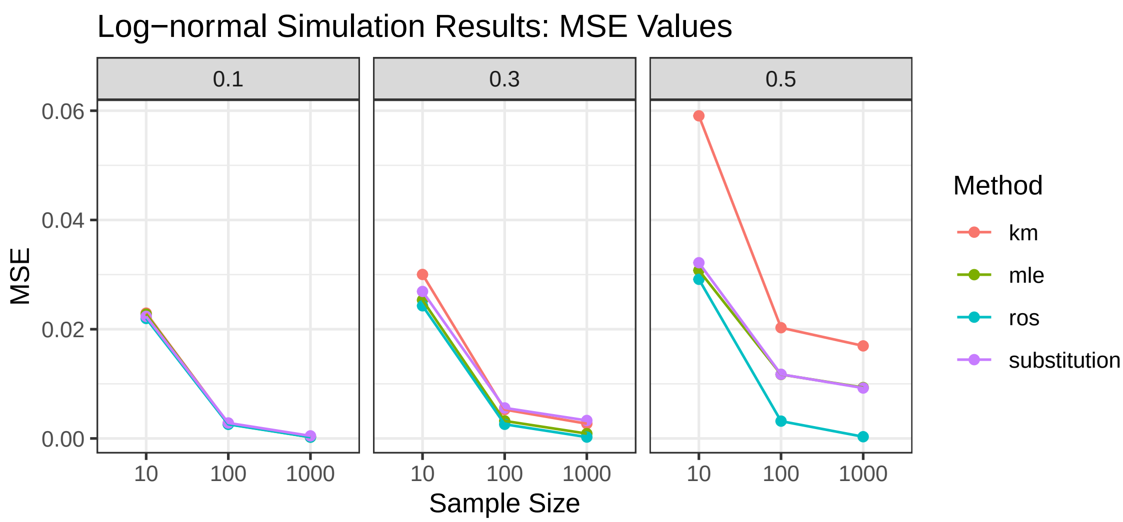

Figure 2.1: MSE values of lognormal simulations, faceted by censoring rates.

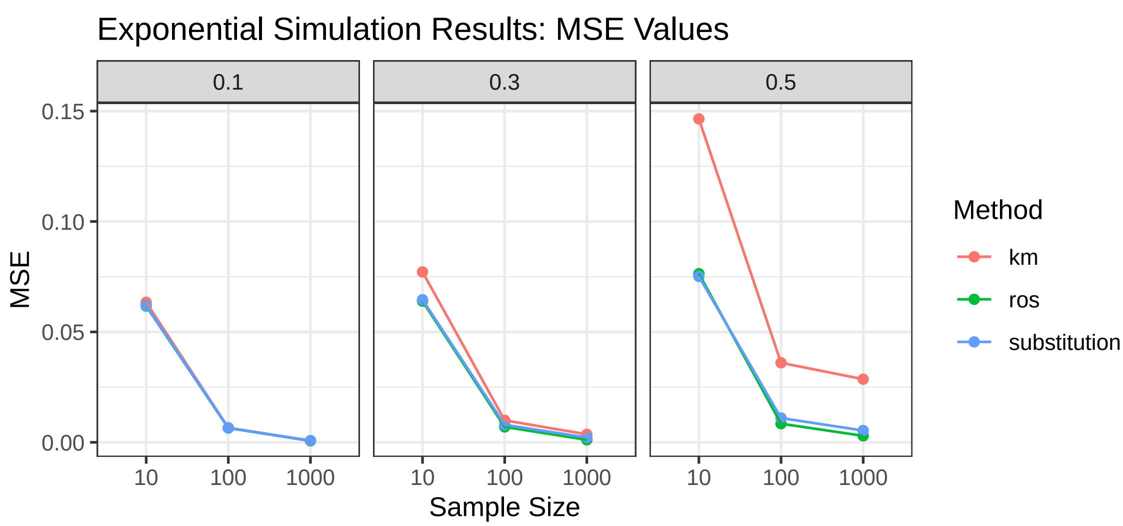

Figure 2.2: MSE values of exponential simulations, faceted by censoring rates.

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

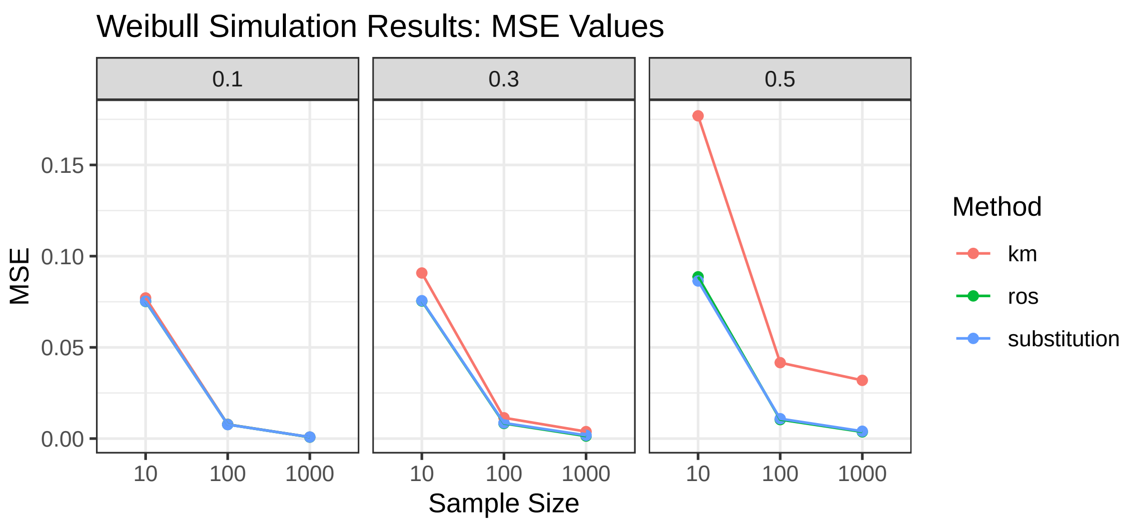

Figure 2.3: MSE values of Weibull simulations, faceted by censoring rates.

2.5.1 Log-normal

Table 2.1 shows our simulation results for the log-normal distribution. In the case of low censoring (0.10), when dealing with sample sizes of 10 and 100, the difference in the performances of the methods are not very pronounced, but they exist. 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 significantly worse than KM, while 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 worst, 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.

In the case of the log-normally distributed data, 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.

2.5.2 Exponential and Weibull

The results of the simulation study for the exponential and Weibull-distributed data can be viewed in Tables 2.2 and 2.3. It must be noted that results for MLE are absent due to the inability of NADA's `cenmle` function to work with non-lognormally distributed data. In future works, we may want to write a custom function to apply MLE for such cases.

We can see for both the exponential and weibull distributed data, all three methods, substitution, KM, and ROS, 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. Unfortunately, in the case of medium (0.30) and high (0.50) censoring, KM consistently performs the worst when compared to the other methods across all sample sizes. In contrast to the bad performance by the KM method, ROS on the other hand, 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 with log-normally distributed data 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 log-normal data, it performs quite well, although not quite as well as ROS. ROS performed the best in the case of medium and high censoring across all three 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 log-normal 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 log-normal 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 log-normal 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 eleven observations were censored (around 20% of the dataset). While the KM method is not ideal for highly censored cases from the results of our simulation study, She (1997)'s results suggest that it is able to be used for smaller sample sizes.

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)'s study, methods may perform differently when utilized with artificial datasets as compared to real world, left censored data. As such, we do not claim our findings in the simulation study to be representative for all cases of left censored data.

Chapter 3 Case Study

3.1 Background

In a report detailing the extensive dangers and risks posed by coal plants and the waste they produce, Kelderman et al. (2019) explains how 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.

Kelderman et al. (2019) disseminates the long list of dangers posed by coal ash by detailing the chemical composition in coal, 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.

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 3/5 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, has discussed the prevalence of groundwater contamination in the wells located in these coal related facilities.

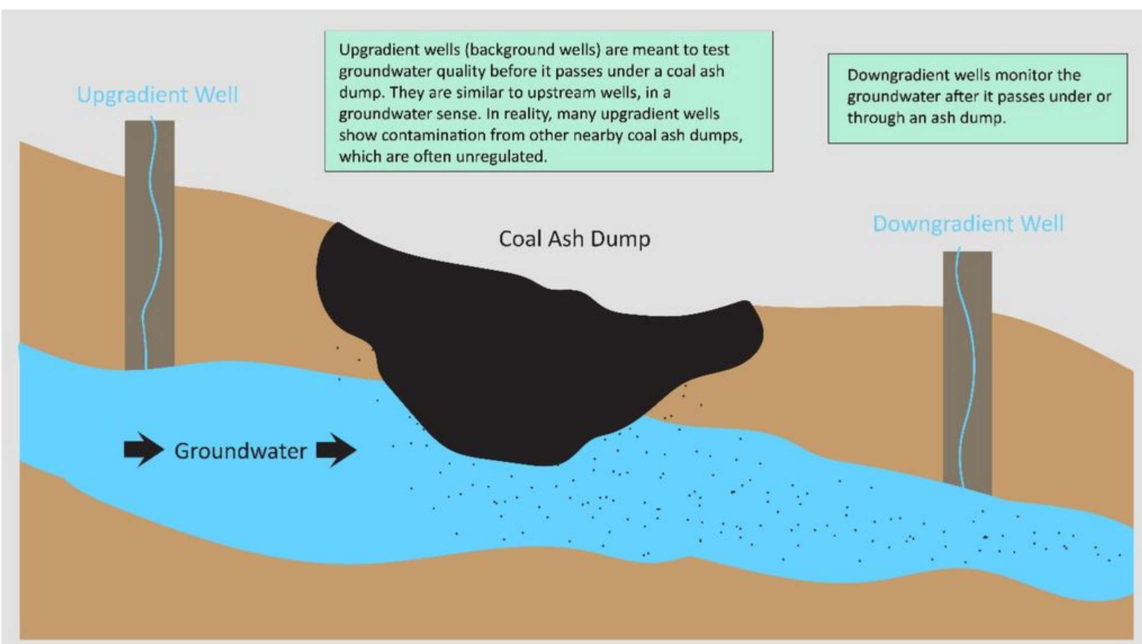

Figure 3.1: Difference Between Upgradient and Downgradient Wells

Typically in a coal ash plant, there exist 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, measure 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: "landfill," "mixed," and "surface-impacted" wells. 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

Table 3.1: Data dictionary for the coal dataset.

Variable	Description
State	The state where the site is located.
Site	The name of the site, as presented in the groundwater monitoring report.
Disposal Area	The name of the disposal area(s), as presented in the groundwater monitoring report.
Type of Well	The type of disposal unit: surface impoundment, landfill, and mixed multi-unit.
ID of Well	The identifier given to each monitoring well in the groundwater monitoring report.
Gradient Type	The location of the groundwater monitoring well: upgradient or downgradient.
Sample Date	The date the well was sampled.
Contaminant Name	The contaminant name.
Measurement Unit	The concentration units. These include mg/l, ug/l, and pCi/l.
Below Detection	LOD status: '<' indicates the concentration is below the LOD.
Concentration	The concentration of the contaminant.

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 an indicator on whether the measurement is below the limit of detection or not. 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. Recalling the discussion regarding the statistical feasibility of the substitution method from Chapter 1, their usage of this method may pose some questions regarding the viability of their results.

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, 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

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.

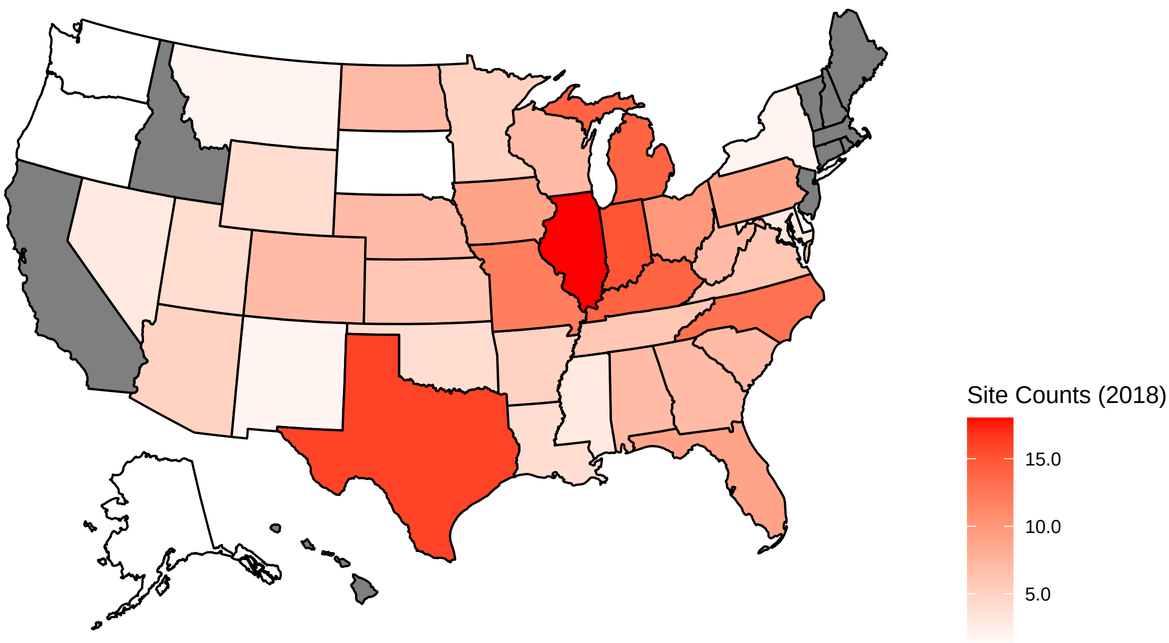

Figure 3.2: Counts of Coal Sites in the United States (gray indicates no sites for that state).

3.3.1 Kelderman's Top Ten 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 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 procedures which will be detailed next.

After these values were obtained, any wells with average downgradient concentrations 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 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.

```
'summarise()' ungrouping output (override with '.groups' argument)
```

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:

Table 3.2: Health-based thresholds set by EPA.

Contaminant	Exceedance Limit	Unit	Range
Antimony	6	ug/L	(0, 50)
Arsenic	10	ug/L	(0, 1890)
Barium	2	mg/L	(0.00010, 429000)
Beryllium	4	ug/L	(0, 57)
Boron	3	mg/L	(0, 62248)
Cadmium	5	ug/L	(0, 197)
Chromium	100	ug/L	(0, 1300)
Cobalt	6	ug/L	(0, 4750)
Fluoride	4	mg/L	(0, 100000)
Lead	15	ug/L	(0, 475)
Lithium	40	ug/L	(0.00007, 10050)
Mercury	2	ug/L	(0, 5.84)
Molybdenum	40	ug/L	(0, 8300)
Radium	5	pCi/L	(-63.335, 302)
Selenium	50	ug/L	(0, 642)
Sulfate	500	mg/L	(0.01, 1340000)
Thallium	2	ug/L	(0, 474)

```
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 criteria 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 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.

3.3.2 (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

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

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

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 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 noted a limitation of the MLE method is that 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. We do not observe any significant differences in top ten sites when comparing all three implementations.

Chapter 4 Conclusion

Missing data is is an important problem in the field of statistics which will most likely continue to plague data in the next few decades. As an issue for which there is no one-size-fits-all solution designed to effectively remedy it, it is important for practiced statisticians to understand and carefully consider what type of method to use for the situation at hand. In this thesis, we provided an overview regarding the concept of missingness, specifically with censoring, and the ramifications that it can have on statistical analyses. We discussed methods designed to work with left-censored data: substitution, ML estimation, KM, and ROS, which are designed to obtain parameter estimates for left-censored datasets. After our exploration on the theoretical foundations of these methods, we carefully designed a simulation study to investigate our hypothesis the effectiveness of each of our methods would differ when for different proportions, sizes, and underlying distribution for the dataset. We then introduced the problem of coal ash contamination across groundwater wells in the U.S. and introduced the Kelderman et al. (2019)'s report which contained information regarding the top ten most contaminated coal sites across the U.S. Due to a lack of discussion in Kelderman et al. (2019)'s report regarding how they accounted for left-censoring in their analysis, we believed that conclusions may have differed if different methods of handling the left-censored data were used. Motivated by this report, we conducted a case study to explore using a baseline method, substitution, and KM to identify our own top ten most contaminated sites, and compared our results to those obtained by Kelderman et al. (2019).

Even among the many peer-reviewed papers and reports which compared and contrasted several of these methods via different scopes of field of study, a common shared idea was the impossibility of pinning one method down as being the "best." Even from our own case study, we had trouble implementing the MLE and ROS methods for our water quality data, due to issues such as the percentage of censoring being too high or our data being too highly skewed for the methods to be used. Future work involving finding a way to account for these issues, perhaps by investigating the efficacy of our methods with transformed data, would be a conducive path to follow.

The motivation for our thesis was largely centered on left-censored data, which is commonly found in environmental sciences with water quality datasets. Due to recent innovations in data storage and data provenance in general, data is becoming increasingly easier to access for the general public. As such, it would be expected that many of these datasets will inevitably contain missing data as well. Although we have delved into several methods of handing left-censored missingness within this thesis, missingness in data will most likely continue to be an inevitable problem even in the near future.

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 readability and/or setup.

A.1 In Chapter 1: Introduction

No noteworthy code for this chapter.

A.2 In Chapter 2: Simulations

Code for the reorganization of dataframes in order to create tables:

```
ln.table <- df.ln %>%
  select(-"prop_cens") %>%
  select(c(2, 1, 3:6)) %>%
  rename(" " = "method", "Sample Size" = "samplesize",
         "Avg. Mean" = "Avg_Mean")

exp.table <- df.exp %>%
  select(-"prop_cens") %>%
  select(c(2, 1, 3:6)) %>%
  rename(" " = "method", "Sample Size" = "samplesize",
         "Avg. Mean" = "Avg_Mean")

w.table <- df.w %>%
  select(-"prop_cens") %>%
  select(c(2, 1, 3:6)) %>%
```

```r
  rename(" " = "method", "Sample Size" = "samplesize",
         "Avg. Mean" = "Avg_Mean")
```

```r
knitr::kable(ln.table, caption = "Performance metrics of our our
             4 methods with data derived from the log-normal
             distribution with mean = 1 and SD = 0.5.",
             digits = 5, booktabs = "T") %>%
  kable_styling(font_size = 11.5) %>%
  pack_rows("Censoring Rate = 0.1", 1, 12) %>%
  pack_rows("Censoring Rate = 0.3", 13, 24, latex_gap_space = "1em") %>%
  pack_rows("Censoring Rate = 0.5", 25, 36, latex_gap_space = "1em")
```

```r
knitr::kable(exp.table, caption = "Performance metrics of our our 3
             methods (MLE method absent) with data derived from the
             exponential distribution with a shape parameter = 1.",
             digits = 5, booktabs = "T") %>%
  kable_styling(font_size = 11.5) %>%
  pack_rows("Censoring Rate = 0.1", 1, 12) %>%
  pack_rows("Censoring Rate = 0.3", 13, 24, latex_gap_space = "1em") %>%
  pack_rows("Censoring Rate = 0.5", 25, 36, latex_gap_space = "1em")
```

```r
knitr::kable(w.table, caption = "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.",
             digits = 5, booktabs = "T") %>%
  kable_styling(font_size = 11.5) %>%
  pack_rows("Censoring Rate = 0.1", 1, 12) %>%
  pack_rows("Censoring Rate = 0.3", 13, 24, latex_gap_space = "1em") %>%
  pack_rows("Censoring Rate = 0.5", 25, 36, latex_gap_space = "1em")
```

Code for the creation of figures/plots:

```r
ggplot(data = df.ln, aes(x = samplesize, y = MSE, group = method, color = method)) +
  geom_point() +
  facet_wrap(~prop_cens) +
  geom_line() +
  theme_bw() +
```

```r
  ggtitle("Log-normal Simulation Results: MSE Values") +
  scale_x_discrete(name = "Sample Size") +
  labs(color = "Method")
```

```r
df.exp <- df.exp %>%
  filter(method != "mle")

ggplot(data = df.exp, aes(x = samplesize, y = MSE, group = method, color = method)
  geom_point() +
  facet_wrap(~prop_cens) +
  geom_line() +
  theme_bw() +
  ggtitle("Exponential Simulation Results: MSE Values") +
  scale_x_discrete(name = "Sample Size") +
  labs(color = "Method")
```

```r
df.w <- df.w %>%
  filter(method != "mle")

ggplot(data = df.w, aes(x = samplesize, y = MSE, group = method, color = method))
  geom_point() +
  facet_wrap(~prop_cens) +
  geom_line() +
  theme_bw() +
  ggtitle("Weibull Simulation Results: MSE Values") +
  scale_x_discrete(name = "Sample Size") +
  labs(color = "Method")
```

A.3 In Chapter 3: Case Study

A.3.1 Data Dictionary

Code for the creation of the data dictionary:

```r
dd_var <- c("State", "Site", "Disposal Area", "Type of Well",
            "ID of Well", "Gradient Type",
            "Sample Date",
            "Contaminant Name", "Measurement Unit",
            "Below Detection", "Concentration")

dd_desc <- c("The state where the site is located.",
             "The name of the site, as presented in the
             groundwater monitoring report.",
             "The name of the disposal area(s), as presented
             in the groundwater monitoring report.",
             "The type of disposal unit: surface impoundment,
             landfill, and mixed multi-unit.",
             "The identifier given to each monitoring well in the
             groundwater monitoring report.",
             "The location of the groundwater monitoring well:
             upgradient or downgradient.",
             "The date the well was sampled.",
             "The contaminant name.",
             "The concentration units. These include mg/l, ug/l,
             and pCi/l.",
             "LOD status: '<' indicates
             the concentration is below the LOD.",
             "The concentration of the contaminant.")

dd <- as.data.frame(cbind(dd_var, dd_desc)) %>%
  rename("Variable" = "dd_var",
         "Description" = "dd_desc")

knitr::kable(dd, caption = "Data dictionary for the coal dataset.",
             booktabs = "T", linesep = "\\addlinespace") %>%
             kable_styling(latex_options="scale_down")
```

A.3.2 Map

Code for the creation of the US map visualization:

```r
plot_usmap(data = states_n, values = "n", regions = "states") +
  scale_fill_continuous(low = "white", high = "red",
                        name = "Site Counts (2018)",
                        label = scales::comma) +
  theme(legend.position = "right",
        panel.background = element_rect(color = "white",
                                        fill = "white"),
        plot.background = element_rect(color = "white"))
```

A.3.3 Health-based Thresholds

Code for the calculating of range of values in dataset:

```r
#code to obtain ranges for each contaminant

ranges <- full %>%
  group_by(contaminant) %>%
  summarize(min = min(concentration),
            max = max(concentration))
```

Code for the creation of the health-based thresholds dataframe:

```r
#CODE TO MAKE HEALTH BASED THRESHOLDS TABLE

hbt_names <- c("Antimony", "Arsenic", "Barium", "Beryllium",
               "Boron", "Cadmium", "Chromium", "Cobalt", "Fluoride",
               "Lead", "Lithium", "Mercury", "Molybdenum",
               "Radium", "Selenium", "Sulfate", "Thallium")

hbt_values <- c(6, 10, 2, 4,
                3, 5, 100, 6, 4,
                15, 40, 2, 40,
                5, 50, 500, 2)

hbt_units <- c("ug/L", "ug/L", "mg/L", "ug/L",
               "mg/L", "ug/L", "ug/L", "ug/L", "mg/L",
               "ug/L", "ug/L", "ug/L","ug/L",
               "pCi/L", "ug/L", "mg/L", "ug/L")
```

```
hbt_ranges <- c("(0, 50)", "(0, 1890)", "(0.00010, 429000)", "(0, 57)",
               "(0, 62248)", "(0, 197)", "(0, 1300)", "(0, 4750)",
               "(0, 100000)", "(0, 475)", "(0.00007, 10050)",
               "(0, 5.84)", "(0, 8300)", "(-63.335, 302)",
               "(0, 642)", "(0.01, 1340000)", "(0, 474)")

hbt <- as.data.frame(cbind(hbt_names, hbt_values, hbt_units, hbt_ranges)) %>%
  rename("Contaminant" = "hbt_names",
         "Exceedance Limit" = "hbt_values",
         "Unit" = "hbt_units",
         "Range" = "hbt_ranges")
```

A.3.4 Top 10 Tables

Code for the creation of our tables for the top 10 sites, according to a specified method:

```
options(scipen=999) #prevent scientific notation

top10_control.table <- top10_control %>%
  rename("Site" = "site",
         "Composite Score" = "composite_score")

knitr::kable(top10_control.table, caption = "Top 10 most contaminated sites
             for our baseline (control) implementation.",
             digits = 5, booktabs = "T", linesep = "\\addlinespace") %>%
  kable_styling(font_size = 11.5)
```

```
top10_substitution.table <- top10_substitution %>%
  rename("Site" = "site",
         "Composite Score" = "composite_score")

knitr::kable(top10_substitution.table, caption = "Top 10 most contaminated sites
             for our substitution method implementation.",
             digits = 5, booktabs = "T", linesep = "\\addlinespace") %>%
  kable_styling(font_size = 11.5)
```

```
top10_km.table <- top10_km %>%
  rename("Site" = "site",
         "Composite Score" = "composite_score")

knitr::kable(top10_km.table, caption = "Top 10 most contaminated sites
             for our KM method implementation.",
             digits = 5, booktabs = "T", linesep = "\\addlinespace") %>%
  kable_styling(font_size = 11.5)
```

A.4 In Chapter 4: Conclusion

No noteworthy code for this chapter.

Appendix B Simulation/CS 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 log-normal
#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) %>%
```

```r
    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
#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

```
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

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() %>%
  mutate(prop_cens = as.factor(prop_cens),
         samplesize = as.factor(samplesize))
```

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
```

```r
    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() %>%
  mutate(prop_cens = as.factor(prop_cens),
         samplesize = as.factor(samplesize))
```

B.2 Case Study

B.2.1 Libraries

```
library(tidyverse)
library(kableExtra)
library(mosaic)
library(usmap)
```

B.2.2 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")) %>%
```

```r
  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")
```

B.2.3 Standardization of Units

```
#we need to standardize values, some wells report concentration of
#contaminants in different units, we will standardize all units to
#the units provided by the health-based threshold set by the EPA

#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
```

```r
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")
```

```r
#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)
```

B.2.4 Baseline/Control

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

#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 highest average upgradient concentration for that specific
#contaminant/site

downgrad_control2 <- left_join(downgrad_control, upgrad_control) %>%
  #remove downgradient mean concentrations that were greater than "background" levels
  filter(avg_conc > upgrad_conc)


#for each site, identify the well(s) with the highest
#mean concentration of each contaminant
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") %>%
  #calculate the ratios -> 'highest average' concentrations/hbt
  mutate(ratio = avg_conc/as.numeric(limit)) %>%
  group_by(site) %>%
  #sum together all contaminants for a site to get a composite score
```

```
  summarize(composite_score = sum(ratio)) %>%
  arrange(desc(composite_score)) %>%
  slice(1:10)
```

B.2.5 Substitution

```
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()

upgrad_substitution <- avg_substitution %>%
  filter(gradient %in% "Upgradient") %>%
  group_by(contaminant, site) %>%
  summarize(upgrad_conc = min(avg_conc)) %>%
  ungroup()

downgrad_substitution <- avg_substitution %>%
  filter(gradient %in% "Downgradient") %>%
  select(c(well.id, contaminant, site, avg_conc))

downgrad_substitution2 <- left_join(downgrad_substitution,
                                    upgrad_substitution) %>%
  filter(avg_conc >= upgrad_conc)

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)) %>%
  group_by(site) %>%
  summarize(composite_score = sum(ratio)) %>%
  arrange(desc(composite_score)) %>%
  slice(1:10)
```

B.2.6 KM

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

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()

upgrad_km <- avg_km %>%
  filter(gradient %in% "Upgradient") %>%
  group_by(contaminant, site) %>%
  summarize(upgrad_conc = min(avg_conc)) %>%
  ungroup()

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

```r
downgrad_km2 <- left_join(downgrad_km, upgrad_km) %>%
  filter(avg_conc >= upgrad_conc)

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)) %>%
  group_by(site) %>%
  summarize(composite_score = sum(ratio)) %>%
  arrange(desc(composite_score)) %>%
  slice(1:10)
```

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. (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.1002124

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.*

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.

Gillespie, B. W., Chen, Q., Reichert, H., Franzblau, A., Lepkowski, J., Adriaens, P., . . . Garabrant, D. H. (2010). Estimating Population Distributions When Some Data Are Below a Limit of Detection by Using a Reverse Kaplan-Meier Estimator. *Epidemiology*, *21*. http://doi.org/10.1097/EDE.0b013e3181ce9fD8

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

Lee, L. (2020). *NADA: Nondetects and data analysis for environmental data.* Retrieved from `https://CRAN.R-project.org/package=NADA`

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.

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

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