

Estimating Non-Point Source Load and Measuring Uncertainty

Prepared for Metropolitan Council Environmental Services
January 2006

St. Olaf Mathematics Practicum Team
Haley Hedlin, Jostein Reiners, Phillip Schulte,
Allan Trapp II, Molly Tuerk, and Stacey Wood

Abstract

As both industrial and residential development continue to increase, upholding federal and state water standards becomes progressively difficult. Knowledge of the current level of water quality can help to ease public policy decisions concerning this development. Methods of stream monitoring and water quality assessment are of most importance in gaining this information. Obtaining accurate and precise estimates of annual pollutant load are essential. Software packages including FLUX and Estimator currently employ methods of estimation, but how accurate are these estimates? We investigate both common ratio and regression methods for pollutant load, and propose a new estimation model incorporating additional factors affecting pollutant concentrations. We then examine and compare estimates produced by these methods. After establishing the accuracy of an estimate, how can we determine uncertainty surrounding it? Currently, the level of uncertainty in estimates is determined through confidence intervals calculated using an empirical jackknifing procedure. This jackknife method is compared with a bootstrap method. We found that the estimation methods produced varying results, and none of the methods distinctly out-performed one another. For the most accurate estimates, models should be fine-tuned to the dynamics of an individual river. In a simulation of a true river, we determined the quantile bootstrap procedure generates narrower intervals of uncertainty than the jackknife procedure with a comparable level of accuracy. We recommend implementation of the quantile bootstrap approach.

Introduction

Stream monitoring attempts to determine the amount of non-point source pollutant loading. The data obtained can then be used toward improving water quality without over-investing in wastewater treatment, upholding federal and state water standards and making decisions concerning industrial and residential development. These public policy decisions have important implications, therefore, it is imperative to obtain accurate and precise annual pollutant load estimates. However, lack of funding, time and technology prevent daily the collection of precise daily pollutant load estimates. The Metropolitan Council Environmental Services (MCES) invited us to investigate both the accuracy and the precision of their current methods of load estimation.

Throughout the past month, we have focused our attention on two main questions of interest. First, what is the best model for determining pollutant loads and how should the current approach be adjusted? Second, what is the level of uncertainty in non-point source load estimates based upon the current monitoring approach?

Before we assess different models and methods for determining pollutant loads, it is essential to understand basic concepts of load and load estimation. Load is the mass of pollutant that passes a sampling station over a specific period of time. Flux is concentration multiplied by flow, which is the loading rate with units given in mass per unit time.

When expressing flux as a function of time, the area under the function represents load. Hence, in order to obtain the load, we integrate the function of flux over time:

$$Load = \int Flux(t)dt \quad t = time$$

However, this method is unrealistic since we are not equipped with continuous concentration data. Due to the cost of testing stream samples for pollutants, concentration samples are limited to approximately thirty per year. Therefore, a summation of the sample concentrations multiplied by the corresponding flow is a more realistic approach to calculate annual loads(Richards).

Δt_i = interval between observations

q_i = sample flow

$$Load = k \sum_{i=1}^n c_i \cdot q_i \cdot \Delta t_i$$

c_i = sample concentration

k = conversion factor

With over three hundred missing daily pollutant concentration samples, obtaining an accurate annual load estimate is difficult. Hence, pollutant loading errors of 10%-40% are not unusual (Richards). Another factor that creates a lot of uncertainty in our load

estimates, is the rapid change in flow rate seen as the result of a storm event or sudden snow melt.

Given only a month of time, we based our statistical analysis on only three rivers. Nine Mile Creek, Credit River and Bevens Creek were thought to be the most representative of the Minnesota streams. These streams also gave us a combination of urban (Nine Mile), semi-urban (Credit) and rural (Bevens) environments. The pollutants that we considered were nitrates, total phosphorus and total suspended solids. We focused on the years 1996, 1997, 1998 and 2004 for our analysis, as they contained the most complete data sets. Also, note that we obtained our data sets from the MCES Environmental Information Management System (EIMS) online.

This paper will go into depth on a number of topics concerning non-point source load estimation. First, we will discuss and compare the methods of estimating annual loads in FLUX and Estimator. We will then build on the concepts used in the methods of FLUX and Estimator, and propose our own model of estimation. Next, we will evaluate the measures of uncertainty by explaining the statistical methods of jackknifing and bootstrapping and compare how they perform in FLUX, Estimator and our model. Lastly, we will explain the benefits and results of the river simulation we conducted to explore these methods of estimation and assessment of uncertainty.

Models for Load for Estimation

Currently the Metropolitan Council Environmental Services (MCES) uses a software package entitled FLUX to produce estimates of annual pollutant load (Walker 1996). This software provides six separate methods for load calculation. These six methods can be broken down into two main categories: ratio and regression estimates. We considered one example from each category. Method 2, a ratio estimate, adjusts this mean based on the ratio of average daily continuous flow to the average sampled daily flow. This ratio takes into account the fact that a greater number of samples are taken during periods of high flow. FLUX Method 2 is given below:

$$W_2 = \text{Mean}(w) \cdot \left[\frac{\text{Mean}(Q)}{\text{Mean}(q)} \right]$$

where W is the estimated total annual load in kilograms per year, w is a daily sample load in kilograms per year, Q is an average continuous daily flow in cubic feet per second, and q is a daily sample flow in cubic feet per second. This method assumes that flow is the main factor affecting pollutant load, and therefore should not be used if a distinct relationship between flow and concentration exists.

Method 4, regression estimate, continues to build on Method 2 by accounting for the relationship between flow and concentration. FLUX Method 4 is given below.

$$W_2 = \text{Mean}(w) \cdot \left[\frac{\text{Mean}(Q)}{\text{Mean}(q)} \right]^{b+1}$$

where b is the slope of ln(concentration) vs. ln(flow). This method assumes that ln(concentration) is linearly related to ln(flow), and therefore should only be used if there is evidence of this relationship.

Estimator, currently used by the United States Geological Survey, is another software package used to produce annual load estimates (Cohn and Gilroy 2000). The method behind Estimator is a seven-parameter linear regression model predicting ln(concentration).

$$\begin{aligned} \ln(\text{concentration}) = & \beta_0 + \beta_1 \ln\left[\frac{Q}{Q_0}\right] + \beta_2 \left(\ln\left[\frac{Q}{Q_0}\right] \right)^2 + \beta_3 [T - T_0] + \beta_4 [T - T_0]^2 \\ & + \beta_5 \sin[2\pi T] + \beta_6 \cos[2\pi T] \end{aligned}$$

where Q is flow in cubic feet per second, Q₀ is a flow centering constant, T is time in years, and T₀ is a time centering constant. This equation reflects the default option in Estimator. The user has the ability to choose from several variables in order to best predict concentration. Many regression models, such as those utilized by Estimator, predict average daily pollutant ln(concentration) rather than estimate annual load directly.

In this case, $\ln(\text{concentration})$ needs to be transformed back into concentration through exponentiation. During this back-transformation, bias is introduced into the load estimate. This simply means that now any load estimates will deviate systematically from their respective true loads. This systematic deviation, or bias, needs to be corrected. The Estimator program does this automatically, thus producing unbiased estimates (Cohn, DeLong et al. 1989). As we have previously shown, once we have obtained a daily concentration we can then multiply by the corresponding daily flow to calculate a daily flux. Because flux is a rate, we multiply flux by the necessary time factor to convert daily flux into daily load. This process is repeated for each day of the year and the sum of these daily loads gives us an unbiased annual load estimate.

On the other hand, FLUX estimates load directly, so there is no back-transformation bias. Also, FLUX assumes flow is the most important factor in estimating load. Estimator takes a similar approach, considering the effect of both flow and seasonality. While these two factors are essential, we felt that additional factors may be vital to load estimation. For instance, predictors such as precipitation and temperature could supplement the effects of seasonality and flow. High precipitations could indicate the start of a storm event, which may result in higher concentrations. Additionally, temperature could signify other events like snow-melt, which cause pollutants to be deposited into a stream. We also thought it was imperative to consider possible relationships between predictors, such as the relationship between seasonality and flow. In order to explore these additional factors, we developed our own regression model for load estimation loosely based on the methodology behind FLUX and Estimator. The additional data we incorporated for temperature and precipitation reflects the weather conditions at the Minneapolis-St. Paul International Airport for the years 1995-2004 and was obtained through the Climatology Working Group at the University of Minnesota. This data ultimately came from the National Weather Service(<http://climate.umn.edu/HIDradius/radius.asp>).

We began with a graphical exploration of relationships between variables. Using data from all three rivers and all three pollutants between the years of 1993-2004, we validated the assumption of a log-linear relationship between flow and concentration (see

Figures 1,2). We also verified the need for a quadratic $\ln(\text{flow})$ predictor (Figure 3) and a $\ln(\text{flow}) \cdot \text{highflow}$ term (Figure 4,5) to account for curvature and stratification, respectively. Stratification allows us to separate samples taken at times of high flow from those taken at low and medium flows. This allows us to account for differing relationships between concentration and flow at differing flow levels. FLUX employs a flow stratification option, and we wanted to further investigate this.

To combine these predictors into a model, we began with a simple model, $\ln(\text{conc}) = \ln(\text{flow})$. Next we considered additional factors we were most important, such as season and a high flow term for stratification. From there, we added factors such as precipitation, temperature, quadratic terms, and interactions between variables, one at a time to examine their effect on the model. Throughout this process we followed good model building practices, meaning when interaction terms or squared terms were present in the model lower order terms were always included. For example, the introduction of $\ln(\text{flow})^2$ requires that $\ln(\text{flow})$ must be in our model.

Once several adequate models were created, we considered four model diagnostics in order to establish our best model. The deviance of residuals, statistical significance of each of the predictors in our model, R-squared (R^2) value, and residual analysis each played a key role in determining our final model. The deviance of residuals describes the difference between the predicted concentrations and the known concentrations in the sample. Low p-values imply the statistical significance of individual predictor coefficients, meaning that the values of these coefficients can be considered to differ markedly from zero. This reflects the amount of information the predictor provides to the overall model. The R-squared value can be thought of as the percentage of variability among the sampled observations that can be explained by our model. Each of these diagnostics can be seen in the output for each model in the appendix (see table 1). Finally, we examined a plot of the residuals vs. the $\ln(\text{conc})$ values predicted by our model along with a histogram of the residuals to verify that our model satisfied assumptions of normality at all concentration levels (see figure 6,7).

After considering the aforementioned diagnostics, we settled on a nine-parameter model predicting $\ln(\text{concentration})$ as the overall best fit.

$$\ln(\text{conc}) = \beta_0 + \beta_1 \cdot \ln(\text{flow}) + \beta_2 \cdot \ln(\text{flow})^2 + \beta_3 \cdot \text{temp} + \beta_4 \cdot \text{precip} + \beta_5 \cdot \text{spring} \\ + \beta_6 \cdot \text{summer} + \beta_7 \cdot \text{fall} + \beta_8 \cdot \text{highflow} + \beta_9 \cdot \ln(\text{flow}) * \text{highflow}$$

where $\ln(\text{flow})$ and $[\ln(\text{flow})]^2$ are in cubic feet per second; spring, summer, and fall are seasonal indicator variables using winter as a reference point; temp is air temperature in degrees Fahrenheit; and precip is precipitation in inches. Highflow is an indicator variable that takes on the value 1 if the flow is above the mean flow, and 0 if the flow is below the mean flow. $\ln(\text{flow}) * \text{highflow}$ is an interaction term that allows for differing effects on concentration for both high flows and low flows; essentially, this interaction term stratifies based on flow. This model indicates that time of year, temperature, precipitation, and a stratification of flow are important factors in determining $\ln(\text{concentration})$.

This nine-parameter model performed well for all combinations of rivers and pollutants, however, it should be noted that this model is not always the absolute best model for each individual river and pollutant (see appendix). Several predictors improved our model for certain rivers and pollutants. Interaction terms such as $\ln(\text{flow}) * \text{precipitation}$, $\ln(\text{flow}) * \text{temperature}$, and $\text{highflow} * \text{precipitation}$ occasionally had significant effects. We also found that in some cases sine and cosine terms representing seasonal harmonic oscillation, similar to the Estimator terms $\sin(2\pi T)$ and $\cos(2\pi T)$, were superior to season indicators. However, these terms increased the complexity of the model and did not perform well across all rivers and pollutants, so we did not include them in our general model. For optimal results, a custom model should be created to fit each river following the described techniques.

Throughout the process of exploratory data analysis and model building, we used the statistical software package Stata; however, our model could be replicated with any basic statistical package. Once we were satisfied with the performance of our model, we used

it to predict daily ln(concentrations) for our three rivers, three pollutants, and four years. We then transformed these daily ln(concentrations) into annual load estimates using the procedure outlined above with Estimator. Similarly, back-transformation bias was an issue with our model predictions, so we adjusted our load estimates according to Cohn 1989 (Cohn, DeLong et al. 1989). In order to perform this adjustment, we needed an estimate of the standard error of our predicted load. To obtain this standard error, we used a bootstrapping procedure which will be outlined in a later section.

Now that we have discussed a number of estimation methods—Flux Method 2, Flux Method 4, Estimator, and our model—it seems appropriate to compare the various results. The annual load estimates obtained from these four methods do not seem to follow any fundamental trend(see below tables). We did note that FLUX Method 2 consistently gave higher estimates than FLUX Method 4, and that Estimator and our model often produced annual loads between the two FLUX estimates.

| | Estimate |
|----------------------|-----------------|
| FLUX Method 2 | 5300 |
| FLUX Method 4 | 2800 |
| Estimator | 4500 |
| Our Model | 3700 |

A comparison of annual load estimates from the four methods for total phosphorus in Credit River for the year 1996

| | Estimate |
|----------------------|-----------------|
| FLUX Method 2 | 590,000 |
| FLUX Method 4 | 490,000 |
| Estimator | 660,000 |
| Our Model | 620,000 |

A comparison of annual load estimates from the four methods for total phosphorus in Credit River for the year 1996

As you can see from the broad range of these estimates, not much information can be gleaned from looking only at point estimates. We need to create confidence intervals to capture the associated uncertainty.

Uncertainty of Load Estimates

As noted above, estimates of annual loads generated using FLUX, Estimator, and our model are quite varied. Therefore it is crucial to determine how close each estimate is to

the true load. To address this question of uncertainty, a discussion of confidence intervals follows.

Confidence Intervals

A confidence interval is a bound around an estimated population parameter value. The purpose of a confidence interval is to assert an amount of certainty (i.e., 95 % confidence) that the true population parameter value is contained within some calculated bounds. Confidence intervals are generated in hopes of capturing the true population parameter.

There are various methods of calculating confidence intervals. Traditionally, one might use the following equation:

$$\textit{Confidence Interval} = \textit{Mean} \pm 1.96 * \textit{Standard Error}$$

The above mean and standard error are calculated from a given sample. In this example, the mean is the parameter of interest. Although this method for calculating confidence intervals is relatively simple, assumptions for when it should be implemented are rarely met. Such assumptions of normal or t-shaped distributions (both of which are symmetric around the mean) must be made in order to use a traditional confidence interval.

Furthermore, an adequate sample size must be available to obtain valid confidence intervals. In examining the data for Bevens, Credit, and Nine Mile streams, we determined that this direct approach to computing confidence intervals was inappropriate and so we explored other options.

Currently, many statisticians are exploring the concept of deriving confidence intervals through a process of resampling a given sample. With this method, the amount of variation within a sample is more accurately approximated than when relying solely on the standard deviation of the sample. This approximation is intuitively closer to the true population variation than the sample standard error since multiple samples are used to calculate standard error. Unlike traditional confidence intervals, sample variance and sample mean are not of importance in the resampling process. Additionally, the sample distribution need not satisfy any assumptions. However, a representative sample of a

population must be available (similar to the traditional direct method of computing confidence intervals). Random sampling strategies are implemented to best obtain this representation

MCES currently uses the jackknife resampling method. We decided to investigate this resampling technique given that MCES's sampling strategy is somewhat randomized and the flexibility in assumptions about sample distributions. Additionally, we decided to explore another resampling method called the bootstrap. Resampling processes are composed of multiple steps. First, some resampling algorithm must be defined. Suppose we have a sample of size ten, and we resample this sample in a "leave one observation out method." In other words, the first new sample is generated by removing the first observation in the original dataset, the second new sample is generated by removing the second observation, and so on until every observation in the original sample has been removed one and only one time to create ten different new samples. This is jackknife resampling (Efron and Tibshirani 1993). Another resampling strategy could be to randomly sample with replacement a given sample 50 times. This is a bootstrap resampling. Each of these resampling strategies is explained further within this paper.

The second step is to compute a single parameter of interest from each new sample. In our study, the parameter was the annual pollutant load of a particular river. Thus, applying FLUX Method 2 to the ten "leave one observation out" samples, we would obtain ten different estimates of load. Employing FLUX Method 2 to the 50 random samples would yield 50 different estimates of load.

At this point, the procedure for calculating confidence intervals using the estimates from "leave one observation out" and random sampling of the given sample diverge. The random resampling procedure generally produces very sound and shapely distributions from which traditional confidence intervals may be applied. The "leave one observation out" distribution of estimates is not useful since randomness was not included in the process. Consequently, estimates of error, either a standard error called jackknife

standard error or a coefficient of variation using jackknife standard error, must be used in conjunction with traditional confidence interval computations.

Notice that the resampling technique adds another layer to the computation of a confidence interval. Furthermore, the traditional approach of calculating confidence bounds is used in both resampling methods; however, the values put in the place of mean, 1.96, and standard error are different.

The Jackknife

Now that we have methods of calculating annual load from a set of sample data and a method for presenting uncertainty, we want to pursue methods of obtaining uncertainty estimates. In our case uncertainty will be presented in the form of confidence intervals, as previously discussed. We present 95% confidence intervals such that we expect the interval, based on the given method, to contain the true value of load 95% of the time. The jackknife is one method for calculating a confidence interval.

A simple example with theory

Starting with a very basic example, suppose one has a population of numbers, for which they want to say something about the population mean. They don't know all of the numbers in the population, but instead only sampled four values from the population: 2, 5, 7, and 20. In more mathematical terms, we write

$$x = (x_1, x_2, \dots, x_n)$$

where x is the vector of n values we know from the population. We can obtain an estimator of this sample:

$$\hat{\theta} = s(x)$$

where $s(x)$ is a function of the vector of our sampled data used to calculate the estimate of interest. In our case, this function calculates the mean of our sample. Instead of just claiming the population has a mean of 8.5 (as this sample does), we will create a confidence interval for the population. First, we re-sample the sample of four values. Each re-sample involves removing one of the observed values in the sample just once.

Hence, in a sample of size n from the population, we should have n re-samples, each of size $n-1$:

$$x_{(i)} = (x_1, x_2, \dots, x_{i-1}, x_{i+1}, \dots, x_n), \text{ where } i = 1, 2, \dots, n$$

This is referred to as a jackknife replication. So in our example, we have four re-samples, which we will call J(1), J(2), J(3), and J(4). J(1) = (5,7,20), J(2) = (2,7,20), J(3) = (2,5,20), J(4) = (2,5,7), such that J(i) is the re-sample with the i^{th} value removed. We can then calculate the i^{th} jackknife replication of $\hat{\theta}$,

$$\hat{\theta}_{(i)} = s(x_{(i)})$$

which is equivalent to calculating the average of each jackknifed re-sample in this

example. Here, the average of J(1) is $\frac{5+7+20}{3} = 10.67$, J(2) is 9.67, J(3) is 9, and J(4)

is 4.67.

The next step in the process of deriving a confidence interval based upon jackknifing is to find the jackknife estimate of standard error.

$$se_{jack} = \left[\frac{n-1}{n} \cdot \sum (\hat{\theta}_{(i)} - \hat{\theta}_{(\cdot)})^2 \right]^{1/2}, \text{ where } \hat{\theta}_{(\cdot)} = \sum_{i=1}^n \hat{\theta}_{(i)} / n$$

In our case, the jackknife estimate of standard error is 3.97. Finally, we can put together a confidence interval, using the traditional method of $mean \pm 1.96 \cdot se_{jack}$. This results in the following interval

$$8.5 \pm 1.96 \cdot 3.97 = (.72, 16.28)$$

While this interval appears to be a wide interval, we say that based on our methods, we expect the mean of the population to be between .72 and 16.28 about 95% of the time. Small sample size is a contributing factor to the large length of the interval.

Application to load estimation

Of course, we wish to calculate annual load for a given stream instead of the mean of a few values, but we can apply the same principles. First, we note that the vector x is now a matrix, where the i^{th} line corresponds to the i^{th} sample of the year. The number of

samples in a year, which is the number of values in the matrix x , varies based on year, stream, and pollutant of interest. Let us consider measurements of total phosphorus for Bevens Creek in 1996. This year involved 29 samples including both grab samples and composite samples. We calculate load $\hat{\theta} = s(x)$ by defining the function s as a ‘load-calculating’ function. As was already mentioned, there are multiple methods of calculating loads, so let us focus on FLUX Method 2. We allow computer software such as FLUX or R (statistical computing package) to calculate load for each $x_{(i)}$ such that we have 29 jackknife replications. From R, we also obtain the jackknife estimate of standard error. Given these, we define the coefficient of variation (cv) as

$$cv = se_{jack} / mean .$$

R provides both the jackknifed estimate of standard error and the mean of the jackknife replications. Specifically, $se_{jack} = 3020$ and $mean = 27,400$ kg/year, resulting in a coefficient of variation $cv = .11$. With this, we replicate the confidence interval method currently used by Steve Kloiber of the Metropolitan Council Environmental Services (MCES). While FLUX will provide a user with the mean and cv , R requires just a few additional calculations. Our code used to attain these results in R can be found in the appendices. This confidence interval is calculated as follows:

$$95\%CI = mean \cdot e^{(\pm 2 \cdot cv)}$$

In our case, we obtain an approximate interval of 22,000 to 34,100 kg in that year. This method for calculating intervals is used by MCES for a variety of streams and rivers, but it is not the only method of generating confidence intervals.

Pseudo-load based interval

An additional method for computing a confidence interval via the jackknife procedure is based upon pseudo-loads for each jackknife sample. We define a pseudo-load for the i^{th} jackknife sample as

$$\tilde{\theta}_i = n\hat{\theta} - (n-1)\hat{\theta}_{(i)}$$

Additionally, we average all of these to obtain a new estimator of annual load, $\tilde{\theta}$. From here, there are two methods to pursue. First, we could calculate an interval such as

$\tilde{\theta} \pm t_{n-1}^{(1-\alpha)} \cdot se_{jack}$ where the coefficient of the jackknife standard error is derived from a t -distribution with $n-1$ degrees of freedom. However, this interval does not perform well and better methods are available (Efron and Tibshirani 1993).

Another method using pseudo-loads is similar to the method used by MCES. Instead of using the mean jackknifed load, we use the mean pseudo-load to obtain a confidence interval. With this method, we calculated a confidence interval for the total phosphorus levels in Bevens Creek during 1997 to be 22,200 to 34,600 kg. Upon further inspection, we found that this interval is also an ineffective method of conveying uncertainty. Overall, there is no evidence that pseudo-values are a useful method of analyzing the jackknife (Efron and Tibshirani 1993).

Bootstrapping

Bootstrapping is the random resampling technique as outlined under the confidence interval section. Due to its highly computational nature, it has only recently made a debut in statistics due to advanced computer technology. As one might imagine, it is a relatively new statistical tool used to generate confidence intervals.

To explain how bootstrapping works, we turn to an example. Suppose that we have a river that has 30 concentration samples in a year. To perform a single bootstrap of this sample, we randomly draw, with replacement, 30 concentrations from the sample. Consequently, each time we draw an observation, there is a $1/30$ probability of grabbing the same observation from a previous draw since the drawn observation is replaced. This means there may be zero, double, triple, or more occurrences of observations from the original sample in the new sample. Next, we calculate a parameter of interest in each new sample. From here, we can compare observed sample parameters, provide an estimate for the population, and derive confidence intervals. Using this distribution is justified since we are treating the sample as if it were the true population. Hence, if the original sample is somewhat representative of the population where hints of the population variability are present, it is sufficient to bootstrap. Using a large number of

simple random samples of the original sample will give a distribution where traditional confidence intervals may be applied.

Types of Bootstrap confidence intervals

The bootstrap histogram distribution of sample estimates per stream and year will occasionally have the form needed to use traditional confidence interval derivations. However, unlike traditional confidence intervals, there are instances where it is not necessary to make distributional assumptions.

For symmetric bootstrap distributions, we can make one of two distributional assumptions. If the shape of the histogram appeared normal, we assumed a normal distribution and used the following formula

$$95\%CI = estimate \pm 1.96 * SE$$

This estimate is the mean of the bootstrap distribution of the load estimates. *SE* is the standard error of the bootstrap distribution.

If the shape of the histogram appeared to be a t-distribution, which is similar to a normal distribution but with smaller mean and longer tails, we assumed this distribution and used the following formula

$$95\%CI = estimate \pm t_{df}^{97.5} * SE$$

The $t_{df}^{97.5}$ is a coefficient generated from the 97.5 percentile of a t-distribution with degrees of freedom (df) equal to the original sample size minus one. Estimate and *SE* are the same as defined in normality assumptions.

The next two methods are the quantile methods of calculating confidence intervals make no distributional assumptions. The basic premise of this method is to order the values of interest for each bootstrap sample, choose the values of the 2.5% and 97.5% observations, and report these values as the lower and upper bounds, respectively. Bias correction and accelerated is also a quantile method that instead shifts the interval a small amount. It attempts to correct for bias within the bootstrap process that results from

outlying observations. We refer to Efron and Tibshirani, 1993, for further discussion of the bias correction and accelerated (BCa) method.

A simple example with theory

Suppose we have a population of values, for which we want to say something about their mean. We observe a simple random sample of size $n=4$ from the population. We use x to denote the vector of values from our simple random sample, $x = (2,5,7,20)$. As we do the bootstrap procedure, we obtain multiple new vectors x_i , $i = 1,2,\dots,m$, each of size $n=4$. Also, in comparison to jackknife procedures, we can create as many of the new vectors x_i as we wish. According to Efron and Tibshirani, 1993, a good bootstrap procedure requires at least 200 iterations, creating 200 vectors. In this example, we iterate only five times for simplicity. The result is the following randomly derived vectors

$$\begin{aligned}x_1 &= (2,7,7,20) \\x_2 &= (5,7,20,20) \\x_3 &= (2,2,5,7) \\x_4 &= (5,5,7,20) \\x_5 &= (2,5,5,20)\end{aligned}$$

From here, we can calculate the mean of each vector, such that $\bar{x}_1 = \text{mean}(x_1) = 9$, $\bar{x}_2 = 13$, $\bar{x}_3 = 4$, $\bar{x}_4 = 9.25$, $\bar{x}_5 = 8$. The mean of these provides us with an estimate of the population mean, namely 8.65. The standard error of these means is $se_{boot} = 3.22$. Hence, we can use a traditional technique to obtain a confidence interval $95\%CI = \text{estimate} \pm 1.96 \cdot se_{boot}$. In this simple example, the result is an interval (2.34, 14.96). Thus, we can say we expect 95% of the time that the true mean of the population will be between 2.34 and 14.96.

Application to load estimation

We can use this same process with data related to annual load. Suppose we examine the 29 samples of phosphorus concentration from Bevens Creek in 1996. From these 29

observations, we create 2000 re-samples, $x_1, x_2, \dots, x_{2000}$ to calculate the annual load. Hence, we can use a load calculation method such as FLUX Method 4 on each of the 2000 vectors, leaving us with 2000 estimates of annual load. We can then obtain the distribution of the 2000 estimates by plotting the values in a histogram. In this instance, the total loads are fairly symmetric and normally distributed. We can use the quantiles method above to create an interval. Hence, we use the 50th and 1950th smallest values for our lower bound and upper bound, respectively. Thus, the result our bootstrap process gives us is the interval 9800 to 16,000 kg for the year. Based on our chosen method, we are 95% confident that the true annual load of phosphorus is between 9800 and 16,000 kg in 1996 for Bevens Creek.

River Simulation

In order to assess the validity of our confidence intervals for annual load, we sought to determine how frequently they captured the true annual load. Performing such an evaluation requires definitive knowledge of the true annual load. However, since the rivers we studied had limited concentration sampling data, we could only obtain *estimates* of the total annual load. Therefore, we performed a stream simulation in which we created continuous records of both concentration and flow to calculate the true annual load for our simulated stream. We then assessed the validity of the confidence intervals our methods produced for the simulated stream.

To make our simulated river realistic, we based it on flow and nitrate concentration from the Credit River. Since the Credit River watershed exhibits a mix of urban and rural land usage, it served as a “median” model on which to base our simulation. A graphical exploration of 1996 Credit River nitrate concentration data revealed that concentrations during this year reflected other yearly concentration trends. Therefore, based on flow and concentration data from the Credit River in 1996, we designed a simulated river to reflect the general characteristics of the three rivers we studied.

Since we had access to daily flow measurements for the Credit River in 1996, we simply adopted these flows for our new simulated river. Only twenty-two nitrate concentration

samples were taken during 1996; the simulation exercise therefore required us to “fill in” the missing concentrations such that we could obtain a complete concentration record, an essential in computing true annual load. A *loess smoother* alleviates the difficulty of fitting a regression curve to the plotted concentrations from 1996. This statistical technique of local regression plots a smooth curve among the sampled concentrations, allowing each sample to affect the final curve according to a rate dependent on its extremeness. This curve does not connect all the sampled values, but rather lies among the points, reflecting the scaled importance of each. Spanning the year 1996, the loess curve gave concentration predictions for the days lacking sample data. In order to reflect the variability of concentrations, we added slight random noise to the smooth curve.

Multiplying each simulated daily nitrate concentration by its corresponding daily flow rate and a scaling factor yielded a collection of daily loads, which we summed to calculate the true annual load of our simulated river. Having obtained the true annual load, we desired to test the validity of our confidence intervals by determining how frequently they captured the true load in 1000 samplings.

Our construction of confidence intervals for annual load began by sampling concentrations from our simulated river in a manner similar to the way in which MCES samples pollutant concentrations throughout the year. MCES records show that a typical sampling station collects a total of approximately thirty yearly grab and composite samples. We collected a discrete number from the simulated river by choosing thirty of the daily concentration-flow pairs. In order to reflect the sampling strategy implemented by MCES, we took a higher proportion of our samples at high flow rates. This mimics the composite samples taken automatically at high flows. We also included the possibility of sampling at low flows to mimic the grab samples taken throughout the year.

Having collected a set of sample concentrations from our simulated river, we sought to create 95% confidence intervals for annual load. This process mirrors the real-world load estimation procedure: beginning with a finite set of concentration samplings and a

continuous flow record, we sought to generate a confidence interval for annual load of the simulated river.

For comparison purposes, we generated confidence intervals from our thirty simulated river concentration samples using the four bootstrapping methods and the two jackknifing methods outlined earlier. Using the bootstrapping method, we first iterated the resampling procedure inherent to bootstrapping 2000 times on our sample of thirty concentrations from the simulated river. This process yielded 2000 bootstrapped samples, each containing thirty concentrations randomly chosen from our original sample of thirty. For each of these new samples, we implemented the equations used by FLUX Methods 2 and 4 to obtain 2000 estimates of yearly load for both FLUX calculation methods. From the distribution of these load estimates, we generated confidence intervals according to each of the four methods under bootstrapping, namely normality, t-confidence, quantiles, and bias corrected and accelerated.

We also created confidence intervals from the jackknifing procedure in order to compare them with those created by bootstrapping. We generated these intervals by calculating a coefficient of variation and utilizing the following formula:

$95\%CI = (load_estimate) \cdot e^{(\pm 2 \cdot cv)}$ (see justification in the jackknife section). This represents the current method of obtaining confidence intervals when using FLUX as the only tool for load estimation. We also calculated a confidence interval based on pseudo-load (a slight variation of the standard method) from the jackknifing procedure as such: $95\%CI = (average_pseudo_load) \cdot e^{(\pm 2 \cdot cv)}$. Because the average pseudo-load is also an estimate of annual load, this method seems to produce a similar confidence interval.

After generating the confidence intervals from the various methods of bootstrapping and jackknifing, we compared the validity of each of the intervals by examining how frequently each captured the true load. This demanded repeating the procedure of creating confidence intervals by starting over with a new sample of thirty flow-concentration pairs taken from our simulated river. According to proper simulation technique, we drew our new sample of pollutant concentrations from a loess smooth

curve with different random noise added. With each repetition of the procedure, we employed bootstrapping to create a new confidence interval. After 1000 iterations of the entire process, we judged the various intervals by the frequency at which they captured the true load of the simulated stream.

Statistical theory states that upon repeated trials, we expect a 95% confidence interval to capture the true yearly load 95% of the time. Using FLUX Method 2 to calculate loads associated with the bootstrapped samples, the 95% confidence intervals captured the true load approximately 98% of the time in all bootstrap confidence interval methods (see table 2). The traditional jackknife-inspired confidence intervals performed similarly. Using FLUX Method 4 to calculate loads associated with the bootstrapped samples, the 95% confidence intervals captured the true load approximately 92% of the time. The intervals created after jackknifing also performed similarly in this situation. The slight over-coverage produced with FLUX Method 2 and the slight under-coverage produced with FLUX Method 4 simply result from the respective variation in load estimation associated with each method; both produced intervals sufficiently close to the desired 95% coverage. When calculated according to the pseudo-load method, the jackknifed confidence intervals exhibited a low rate of coverage and were therefore dropped from further investigation.

Although all intervals seemed to capture the true load close to 95% of the time, we suggest another criterion that clearly favors bootstrap-generated confidence to those created with jackknifing. The confidence intervals created with bootstrapping were approximately 16% smaller than the intervals created with jackknifing. Among intervals with equal coverage rates (for a given load estimation method, both the bootstrap and jackknife intervals captured the true load equally often), shorter intervals yield a better estimation of load by incorporating less variability. Since short confidence intervals incorporate less uncertainty than long intervals with the same coverage rate, the bootstrapped confidence intervals are preferred, particularly the quantiles method of calculation.

Conclusion

Although FLUX provides adequate models for pollutant load estimation, these models frequently produce starkly contrasting load estimates; one must judiciously decide which model suits a given situation. In creating our model, we carefully examined our data and chose regression predictors that improved the model within our situational context. The model we created fits well across all three rivers and pollutants we examined. However, we found that tailoring the model to a particular river and/or pollutant further improved the model's performance. The general model we suggested performs comparably to FLUX under the appropriate method.

Our simulation revealed that bootstrapping generates tighter confidence intervals with the same coverage rate as jackknifing. Particularly, the quantiles interval calculation creates tight confidence intervals that are suitable in the widest variety of situations. Because these intervals are smaller than those obtained using jackknifed coefficients of variation, they incorporate less uncertainty. Thus, creating bootstrapped confidence intervals for annual pollutant loads reduces the overall level of uncertainty.

Future

In addition to the optional predictors we discussed in the Models for Load Estimation section, a possible relationship could exist between the first derivative of flow and concentration. Since concentration does not peak at the same time as the concentration peaks, then the rate of flow increase would describe concentration levels in the water after storms. With flow measurements taken every 15 minutes, this relationship could easily be determined.

Further river simulations could increase the robustness of our preference for bootstrapped confidence intervals. One could base simulated concentration and flow data on samples from other rivers and streams, for example. Alternatively, one could use different schemes for adding noise to the loess curve of predicted concentrations. While our simulation mimicked conditions from the Credit River in 1996, simulating additional

streams during different years with various pollutants would increase the validity of our assertions.

The St. Olaf Mathematics Practicum Team would like to thank the Metropolitan Council Environmental Services for providing us with this significant research project. We thank Steve Kloiber and Kent Johnson for collaborating with us and providing helpful information and suggestions. We hope that our suggestions provide insight into measuring the uncertainty in pollutant load estimates.

(Cochran 1977; Mosteller and Tukey 1978; Efron 1982; Bodo and Umy 1983; Gilroy, Hirsch et al. 1990; Cohn, Calder et al. 1992; Efron and Tibshirani 1993; Polta, Jacobson et al. 1999)

References

- Bodo, B. and T. B. Umy (1983). "Sampling strategies for mass-discharge estimation." Journal of the Engineering Division, American Society of Civil Engineers **198**(4): 812-829.
- Cochran, W. G. (1977). Sampling techniques. New York, John Wiley and Sons.
- Cohn, T. A., D. L. Caulder, et al. (1992). "The validity of a simple statistical model for estimating fluvial constituent loads: an empirical study involving nutrient loads entering Chesapeake Bay." Water Resources Research **28**(9): 2353-2363.
- Cohn, T. A., L. L. DeLong, et al. (1989). "Estimating constituent loads." Water Resources Research **25**(5): 937-942.
- Cohn, T. A. and E. J. Gilroy (2000). "Instructions for using the estimator software."
- Efron, B. (1982). The jackknife, the bootstrap, and other resampling plans. Philadelphia, PA, Society for Industrial and Applied Mathematics.
- Efron, B. and J. Tibshirani (1993). An introduction to the bootstrap. New York, Chapman and Hall.
- Gilroy, E. J., R. M. Hirsch, et al. (1990). "Mean square error of regression-based constituent transport estimates." Water Resources Research **26**(9): 2069-2077.
- Mosteller, F. and J. W. Tukey (1978). Data analysis and regression -A second course in statistics. Reading, MA, Addison-Wesley.
- Polta, R., R. Jacobson, et al. (1999). NPS loadings--can you trust them? EPE Report 99-471, Metropolitan Council Environmental Services, Research and Development Section, St. Paul, MN: 37.
- Richards, R. P. Estimation of pollutant loads in rivers and streams: A guidance document for NPS programs.
- Walker, W. W. (1996). Simplified procedures for eutrophication assessment and prediction: User Manual. Instruction Report W-96-2. U. S. A. E. W. E. Station.

Appendix

Model for Estimating Pollutant Loads

We have chosen phosphorus and Nine Mile Creek to exhibit trends common to most pollutants and rivers in the graphical exploration section of the appendix.

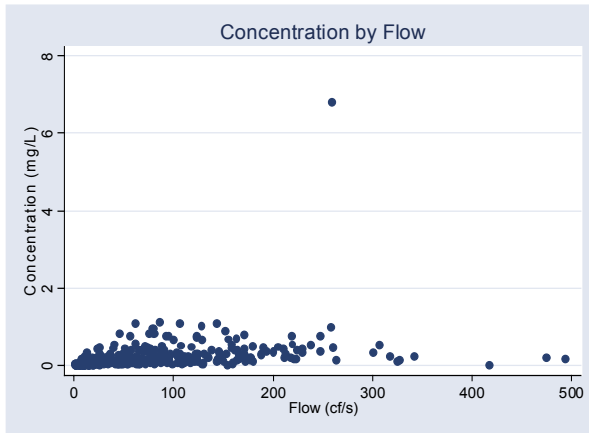


Figure 1: Concentration by flow for total phosphorus in Nine Mile Creek

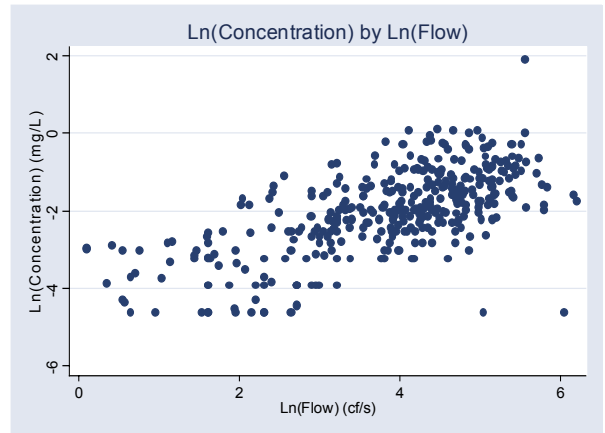


Figure 2: Ln(concentration) by ln(flow) for total phosphorus in Nine Mile Creek

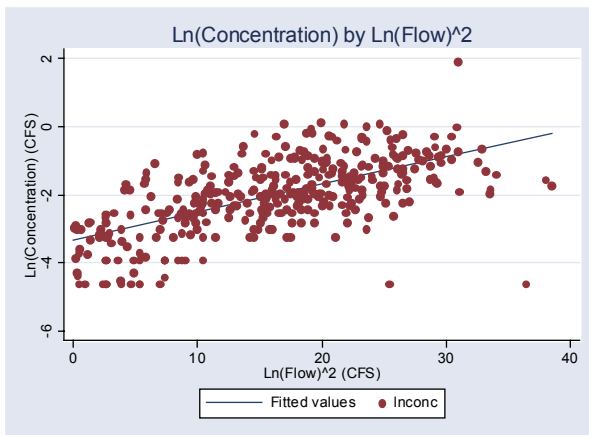


Figure 3: Ln(concentration) by $[\ln(\text{flow})]^2$ for total phosphorus in Nine Mile Creek (compare to fig2?)

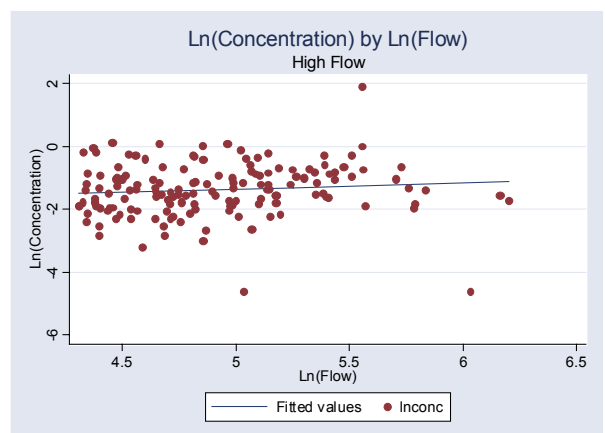
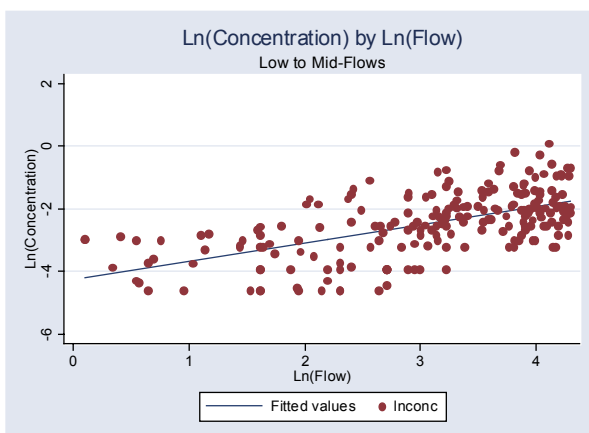


Figure 4: Ln(concentration) by ln(flow) for total phosphorus at low and mid-flow* times in Nine Mile Creek

Figure 5: Ln(concentration) by ln(flow) for total phosphorus at high flow* times in Mile Creek

*We defined a high flow as any flow above the mean (determined from all flows between the years 1993 and 2004) and low and mid-flows as any flow below the mean.

| Source | SS | df | MS | | | |
|----------|------------|-----|------------|-----------------|--------|--|
| Model | 221.339874 | 9 | 24.5933194 | Number of obs = | 349 | |
| Residual | 202.349277 | 339 | .596900522 | F(9, 339) = | 41.20 | |
| Total | 423.689151 | 348 | 1.21749756 | Prob > F = | 0.0000 | |
| | | | | R-squared = | 0.5224 | |
| | | | | Adj R-squared = | 0.5097 | |
| | | | | Root MSE = | .77259 | |

| lnconc | Coef. | Std. Err. | t | P> t | [95% Conf. Interval] | |
|--------------|-----------|-----------|--------|-------|----------------------|-----------|
| lnflow | .0429734 | .2380258 | 0.18 | 0.857 | -.42522 | .5111668 |
| lnflow2 | .0791393 | .0450354 | 1.76 | 0.080 | -.0094448 | .1677234 |
| spring | .5285192 | .1820339 | 2.90 | 0.004 | .1704609 | .8865774 |
| summer | .2997691 | .2191953 | 1.37 | 0.172 | -.1313851 | .7309233 |
| fall | .0740877 | .1942195 | 0.38 | 0.703 | -.3079395 | .4561148 |
| temp | .0127285 | .0033565 | 3.79 | 0.000 | .0061263 | .0193307 |
| precip | .2729996 | .0744209 | 3.67 | 0.000 | .1266148 | .4193844 |
| highflow | 4.04846 | 1.203126 | 3.36 | 0.001 | 1.681927 | 6.414993 |
| lnflowXhig~w | -.9031118 | .2800664 | -3.22 | 0.001 | -1.453999 | -.3522251 |
| _cons | -4.547751 | .3310143 | -13.74 | 0.000 | -5.198852 | -3.89665 |

Table 1: Output from our regression model using data from Nine Mile Creek total phosphorus

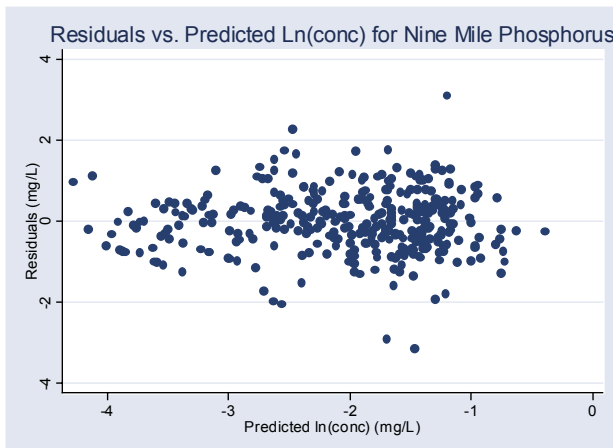


Figure 6: Residuals by predicted ln(concentration) for total phosphorus in Nine Mile Creek

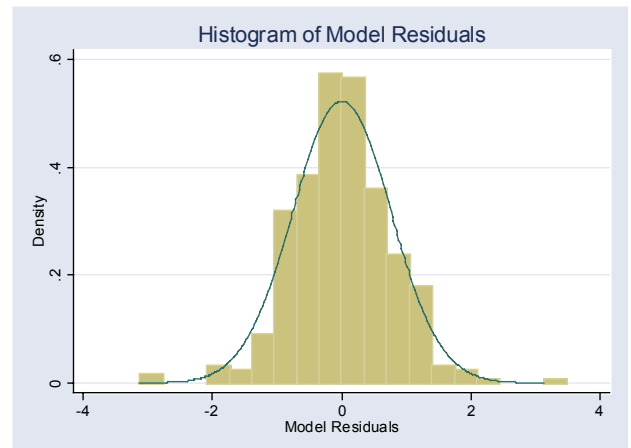


Figure 7: A normal curve superimposed over a histogram of residuals for total phosphorus in Nine Mile Creek

Uncertainty of Load Estimates

Comparison of Confidence Intervals

FLUX Method 2

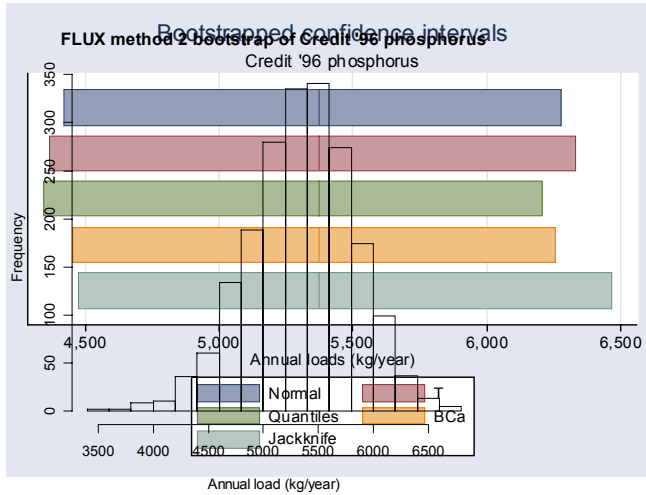


Figure 7: A histogram of the bootstrap distribution as calculated by FLUX Method 2 for Credit River Total Phosphorus in 1996.

Figure 8: The bootstrapped confidence intervals for Credit River Total Phosphorus in 1996 as calculated from the distribution in Figure 7.

FLUX Method 4

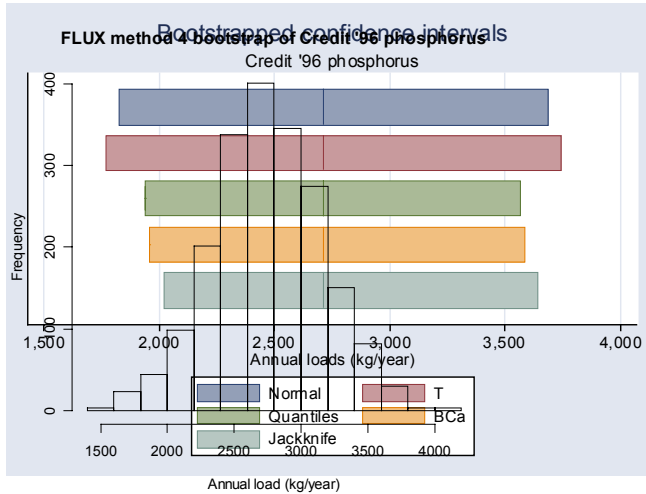


Figure 9: A histogram of the bootstrap distribution as calculated by FLUX Method 4 for Credit River Total Phosphorus in 1996.

Figure 10: The bootstrapped confidence intervals for Credit River Total Phosphorus in 1996 as calculated from the distribution in Figure 9.

Estimator

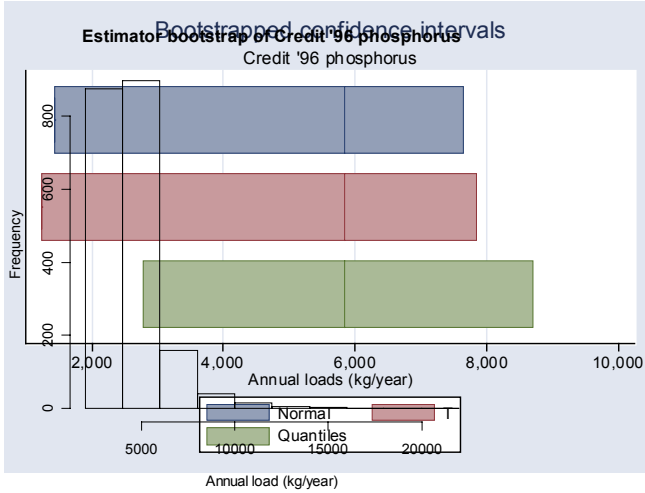


Figure 11: A histogram of the bootstrap distribution as calculated by Estimator for Credit River Total Phosphorus in 1996.

Figure 12: The bootstrapped confidence intervals for Credit River Total Phosphorus in 1996 as calculated from the distribution in Figure 11.

Our Model

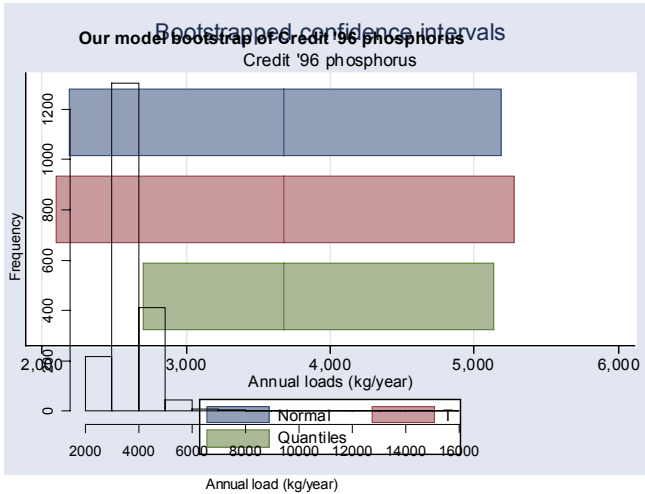


Figure 13: A histogram of the bootstrap distribution as calculated by our model for Credit River Total Phosphorus in 1996.

Figure 14: The bootstrapped confidence intervals for Credit River Total Phosphorus in 1996 as calculated from the distribution in Figure 13.

Quantiles and Jackknifed intervals, Credit River Total Phosphorus 1996

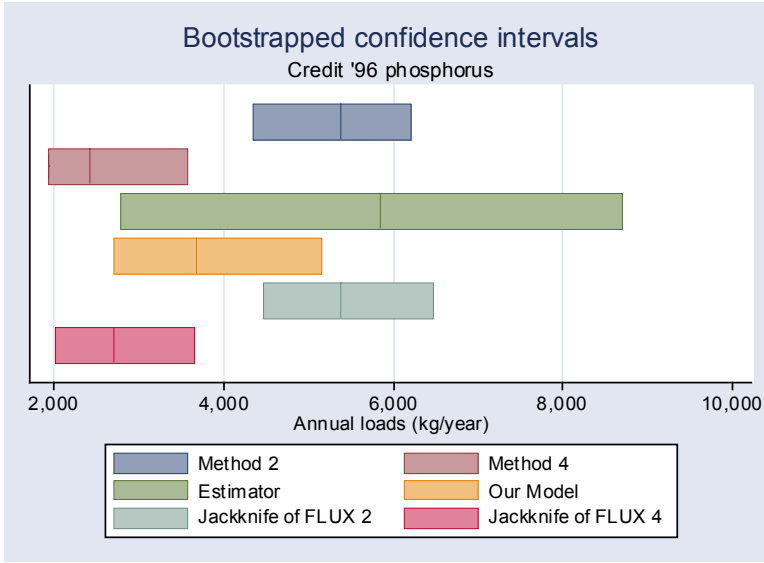


Figure 15: Comparison of methods of load calculations for Credit River Phosphorus in 1996 using the bootstrapping quantile and jackknifing methods of uncertainty calculation.

Quantiles and Jackknifed intervals, Bevens Creek Nitrate 2004

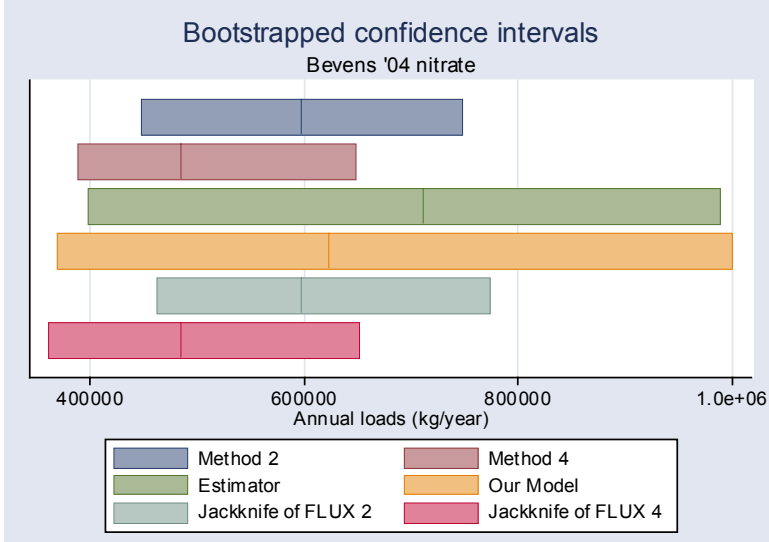


Figure 16: Comparison of methods of load calculations for Bevens Creek Nitrate in 2004 using the bootstrapping quantile and jackknifing methods

River Simulation

| | Normal | T-Dist | Quantile | BCa | Jackknife |
|-------|--------|--------|----------|------|-----------|
| FLUX2 | 98.0 | 98.4 | 98.1 | 98.0 | 98.8 |
| FLUX4 | 92.1 | 93.2 | 93.0 | 88.8 | 93.2 |

Table 2: Percentage of 95% confidence intervals that capture the true load over repeated iterations for various calculation methods.

R-Code for Calculation of Confidence Intervals

R is a powerful freeware program used in the field of Statistics. Sample data should be arranged as follows for the entire script to be utilized

Col1 - DATE (any format)
Col2 - Flow (CFS)
Col3 - concentration (mg/L)
Col4 - $\ln(\text{FlowCFS (col2)} * .8937)$
Col5 - $\ln(\text{concentration (col3)} * 1000)$
Col6 – $\log_{\text{qr}}(\text{the log of the ratio of Q to the centering Q used in Estimator})$
Col7 - \log_{qr}^2 (Col 6 squared)
Col8 – tdiff (Difference between T and the centering T used in Estimator)
Col9 - tdiff^2 (Col 8 squared)
Col10 – $\sin(2\pi T)$
Col11 – $\cos(2\pi T)$
Col12 - $\ln(\text{concentration}(\text{col3}))$
Col13 - Spring
Col14 - Summer
Col15 - Fall
Col16 - Precipitation
Col17 - Temperature
Col18 - \log_{flow}
Col19 - \log_{flow}^2
Col20 - highflow
Col21 – $\log_{\text{flow}} \times \text{highflow}$

Continuous flow data is also necessary and should contain the same data as above, with the following exceptions:

Eliminate columns 3 and 5
Shift column 4 to column 3 and columns 6 through 21 down by two after elimination of 3 and 5.

With data formatted correctly, the only inputs needed for the script on the following page are for the sample data file, continuous flow data file, and start and end dates. All data

files should be saved in comma delimited format (.csv) for importing into R. The code will compile a table of interval results upon finishing all calculations.

```
#Loading Bootstrap package
#Go to Packages, Install Packages, Bootstrap
#Go to Packages, Load Package, Bootstrap
#Bootstrapping process
#Choose the proper directory and data file
xdata<-read.csv(file("bevensp.csv"))
ydata<-read.csv(file("qbevenc.csv"))
#m and n will need to be changed to the number of observations within a
sample
#Choose the proper daily flow average (39.2338 in this example)
#remember to remove one row from an excel sheet that uses the first row
as column headings
m<-1
n<-29
#j and k chosen for year of continuous flow data corresponding to m,n
above
#again, remember to remove one from each if row one of spreadsheet is
column headings
j<-1
k<-366
yflow<-matrix(ydata[j:k,2], nrow=(k-j+1),ncol=1)
flow<-mean(yflow)
flow

#Method 2
set.seed(1234)
theta<-function(x,xdata)
{mean(xdata[x,2]*xdata[x,3]*.8937*1000)*(flow/mean(xdata[x,2]))}
results<-bootstrap(m:n,2000,theta,xdata)
#Bootstrapped Flux via Method 2
y<-c(results $thetastar)
mu<-mean(y)
mu

#Method 2 of Flux
mua<-
mean(xdata[m:n,2]*xdata[m:n,3]*.8937*1000)*(flow/mean(xdata[m:n,2]))
mua

#Histogram of Bootstrapped Values
hist(y)

#Deriving the 95 % Confidence Interval for Bootstrapped Method 2
#Assuming normality and low bias we can use the following:
se<-sd(results$thetastar)
lbound<-mu-1.96*se
ubound<-mu+1.96*se
lbound
ubound

#Bootstrapped t-confidence interval
lbounda<-mu-qt(.975,n-m-1)*se
ubounda<-mu+qt(.975,n-m-1)*se
```

```

lbounda
ubounda

#Using quantiles
per97.5 <-quantile(y,.975)
per2.5 <- quantile(y,.025)
per2.5
per97.5

#using bias correction acceleration
set.seed(1234)
resultsa<-bcanon(m:n,2000,theta,xdata)
resultsa$confpoints
#95%interval from confpoints
lboundb<-resultsa$confpoints[1,2]
uboundb<-resultsa$confpoints[8,2]
lboundb
uboundb

#Method 4
set.seed(1234)
theta1<-function(x,xdata)
{mean(xdata[x,2]*xdata[x,3]*.8937*1000)*((flow/mean(xdata[x,2]))^
(array(lsfite((xdata[x,4]),(xdata[x,5]))$coefficient,dim=c(1,2))[1
,2]+1))}
resultsl<-bootstrap(m:n,2000,theta1,xdata)

#Bootstrapped Flux via Method 4
y1<-c(resultsl $thetastar)
mul<-mean(y1)
mul

#Method 4 of Flux
mula<-
mean(xdata[m:n,2]*xdata[m:n,3]*.8937*1000)*((flow/mean(xdata[m:n,2]))^
(array(lsfite((xdata[m:n,4]),(xdata[m:n,5]))$coefficient,dim=c(1,2
)))[1,2]+1)
mula

#Histogram of Bootstrapped Values in Method 4
hist(y1)

#Deriving the 95 % Confidence Interval for Bootstrapped Method 4
#Assuming normality and low bias we can use the following:
se1<-sd(resultsa$thetastar)
lboundl<-mul-1.96*se1
uboundl<-mul+1.96*se1
lboundl
uboundl

#Bootstrapped t-confidence interval
lboundla<-mul-qt(.975,n-m-1)*se1
uboundla<-mul+qt(.975,n-m-1)*se1
lboundla
uboundla

#Using quantiles

```

```

per97.5a <-quantile(y1,.975)
per2.5a <- quantile(y1,.025)
per2.5a
per97.5a

#using bias correction acceleration
set.seed(1234)
results1a<-bcanon(m:n,2000,theta1,xdata)
results1a$confpoints
#95%interval from confpoints
lbound1b<-results1a$confpoints[1,2]
ubound1b<-results1a$confpoints[8,2]
lbound1b
ubound1b

#Jackknife of Method 2 and Method 4
jresults<-jackknife(m:n,theta,xdata)
jresults1<-jackknife(m:n,theta1,xdata)

#Coefficient of Variance (CV)
cv<-jresults$jack.se / mua
cv
cv1<-jresults1$jack.se / mula
cv1

#confidence intervals using jackknifing
lbjack<-mua*exp(-2*cv)
ubjack<-mua*exp(2*cv)
lbjack
ubjack
lbjack1<-mula*exp(-2*cv1)
ubjack1<-mula*exp(2*cv1)
lbjack1
ubjack1

#pseudo values
jackarray<-array(jresults$jack.values,dim=c(n-m+1,1))
jpseudo<-(n-m+1)*mua - (n-m)*jackarray
mub<-mean(jpseudo)
mub
jackarray1<-array(jresults1$jack.values,dim=c(n-m+1,1))
jpseudo1<-(n-m+1)*mula - (n-m)*jackarray1
mulb<-mean(jpseudo1)
mulb

#confidence interval around jackknifed estimate and jackknifed CV
lbjackb<-mub*exp(-2*cv)
ubjackb<-mub*exp(2*cv)
lbjackb
ubjackb
lbjack1b<-mulb*exp(-2*cv1)
ubjack1b<-mulb*exp(2*cv1)
lbjack1b
ubjack1b

#Estimator calculations

```

```

#linear regression model
estfunc<-
lm(xdata[,12]~xdata[,6]+xdata[,7]+xdata[,8]+xdata[,9]+xdata[,10]+xdata[
,11],xdata,m:n)
residuals<-array(estfunc$residuals,dim=c(n-m+1,1))

#bootstrap of residuals calculated from linear regression
theta2<-function(r,residuals){r}
set.seed(1234)
results2<-bootstrap(residuals,2000,theta2)
residuals1<-results2$thetastar
estimator<-array(estfunc$coefficients,dim=c(7,1))
estimators<-estimator[1,1]+estimator[2,1]*xdata[m:n,6]+
  estimator[3,1]*xdata[m:n,7]+estimator[4,1]*xdata[m:n,8]+
  estimator[5,1]*xdata[m:n,9]+estimator[6,1]*xdata[m:n,10]+
  estimator[7,1]*xdata[m:n,11]

#estimated values from regression model for each sample (ie...new
ln(c))
estimators1<-array(estimators,dim=c(n-m+1,2000))

#adding bootstrapped residuals to estimated values
values<-matrix(residuals1+estimators1,nrow=n-m+1,ncol=2000)

#create one matrix with values and predictors used in regression model
pred<-array(values,dim=c(n-m+1,2000))
xdata6<-array(xdata[m:n,6],dim=c(n-m+1,1))
xdata7<-array(xdata[m:n,7],dim=c(n-m+1,1))
xdata8<-array(xdata[m:n,8],dim=c(n-m+1,1))
xdata9<-array(xdata[m:n,9],dim=c(n-m+1,1))
xdata10<-array(xdata[m:n,10],dim=c(n-m+1,1))
xdata11<-array(xdata[m:n,11],dim=c(n-m+1,1))
mat.q<-
matrix(c(pred,xdata6,xdata7,xdata8,xdata9,xdata10,xdata11),nrow=n-
m+1,ncol=2006)
#change matrix to data.frame
dat.q<-data.frame(mat.q)

#create matrix that load values will be stored in
mat.f<-matrix(c(0),nrow=2000,ncol=1)

#iteration procedure for calculation of loads with each of the 2000
sets of bootstrapped values

#each result from an iteration saved in mat.f
for(i in 1:2000){
load<-rep(0,2000)
estfun<-
array(lm(dat.q[,i]~dat.q[,2001]+dat.q[,2002]+dat.q[,2003]+dat.q[,2004]+
dat.q[,2005]+dat.q[,2006],dat.q)$coefficients,dim=c(7,1))
variance<-
var(lm(dat.q[,i]~dat.q[,2001]+dat.q[,2002]+dat.q[,2003]+dat.q[,2004]+da
t.q[,2005]+dat.q[,2006],dat.q)$residuals)
lnc<-
estfun[1,1]+estfun[2,1]*ydata[j:k,4]+estfun[3,1]*ydata[j:k,5]+estfun[4,
1]*ydata[j:k,6]+estfun[5,1]*ydata[j:k,7]+estfun[6,1]*ydata[j:k,8]+estfu
n[7,1]*ydata[j:k,9]+(variance/2)

```

```

elnc<-exp(lnc)
load[i]<-sum(2.44658*elnc*ydata[j:k,2])
mat.f[i,1]<-load[i]
}
#mean and confidence intervals
y2<-mat.f
mu2<-mean(y2)
hist(y2)

#Assuming normality and low bias we can use the following:
se2<-sd(mat.f)
lbound2<-mu2-1.96*se2
ubound2<-mu2+1.96*se2
lbound2
ubound2

#Bootstrapped t-confidence interval
lbound2a<-mu2-qt(.975,n-m-1)*se2
ubound2a<-mu2+qt(.975,n-m-1)*se2
lbound2a
ubound2a

#Using quantiles
per97.5b <-quantile(y2,.975)
per2.5b <- quantile(y2,.025)
per2.5b
per97.5b

#NEW MODEL
#linear regression model
estfunc1<-
lm(xdata[,12]~xdata[,13]+xdata[,14]+xdata[,15]+xdata[,16]+xdata[,17]+xdata[,18]+xdata[,19]+xdata[,20]+xdata[,21],xdata,m:n)
residualsa<-array(estfunc1$residuals,dim=c(n-m+1,1))
#bootstrap of residuals calculated from linear regression
theta3<-function(r,residualsa){r}
set.seed(1234)
results2a<-bootstrap(residualsa,2000,theta3)
residualsla<-results2a$thetastar
estimator1<-array(estfunc1$coefficients,dim=c(10,1))
estimatorsa<-estimator1[1,1]+estimator1[2,1]*xdata[m:n,13]+
  estimator1[3,1]*xdata[m:n,14]+estimator1[4,1]*xdata[m:n,15]+
  estimator1[5,1]*xdata[m:n,16]+estimator1[6,1]*xdata[m:n,17]+
  estimator1[7,1]*xdata[m:n,18]+estimator1[8,1]*xdata[m:n,19]+
  estimator1[9,1]*xdata[m:n,20]+estimator1[10,1]*xdata[m:n,21]
#estimated values from regression model for each sample (ie...new
ln(c))
estimators1a<-array(estimatorsa,dim=c(n-m+1,2000))
#adding bootstrapped residuals to estimated values
values1<-matrix(residualsla+estimators1a,nrow=n-m+1,ncol=2000)

#create one matrix with values and predictors used in regression model
pred1<-array(values1,dim=c(n-m+1,2000))
xdata13<-array(xdata[m:n,13],dim=c(n-m+1,1))
xdata14<-array(xdata[m:n,14],dim=c(n-m+1,1))
xdata15<-array(xdata[m:n,15],dim=c(n-m+1,1))
xdata16<-array(xdata[m:n,16],dim=c(n-m+1,1))

```

```

xdata17<-array(xdata[m:n,17],dim=c(n-m+1,1))
xdata18<-array(xdata[m:n,18],dim=c(n-m+1,1))
xdata19<-array(xdata[m:n,19],dim=c(n-m+1,1))
xdata20<-array(xdata[m:n,20],dim=c(n-m+1,1))
xdata21<-array(xdata[m:n,21],dim=c(n-m+1,1))
mat.qa<-
matrix(c(pred1,xdata13,xdata14,xdata15,xdata16,xdata17,xdata18,xdata19,
xdata20,xdata21),nrow=n-m+1,ncol=2009)
#change matrix to data.frame
dat.qa<-data.frame(mat.qa)

#create matrix that load values will be stored in
mat.fa<-matrix(c(0),nrow=2000,ncol=1)

#iteration procedure for calculation of loads with each of the 2000
sets of bootstrapped values

#each result from an iteration saved in mat.fa
for(i in 1:2000){
load<-rep(0,2000)
estfun<-
array(lm(dat.qa[,i]~dat.qa[,2001]+dat.qa[,2002]+dat.qa[,2003]+dat.qa[,2004]+dat.qa[,2005]+dat.qa[,2006]+dat.qa[,2007]+dat.qa[,2008]+dat.qa[,2009],dat.qa)$coefficients,dim=c(10,1))
variance<-
var(lm(dat.qa[,i]~dat.qa[,2001]+dat.qa[,2002]+dat.qa[,2003]+dat.qa[,2004]+dat.qa[,2005]+dat.qa[,2006]+dat.qa[,2007]+dat.qa[,2008]+dat.qa[,2009],dat.qa)$residuals)
lnc<-
estfun[1,1]+estfun[2,1]*ydata[j:k,10]+estfun[3,1]*ydata[j:k,11]+estfun[4,1]*ydata[j:k,12]+estfun[5,1]*ydata[j:k,13]+estfun[6,1]*ydata[j:k,14]+estfun[7,1]*ydata[j:k,15]+estfun[8,1]*ydata[j:k,16]+estfun[9,1]*ydata[j:k,17]+estfun[10,1]*ydata[j:k,18]+(variance/2)
elnc<-exp(lnc)
load[i]<-sum(2.44658*elnc*ydata[j:k,2])
mat.fa[i,1]<-load[i]
}

#mean and confidence intervals
y3<-mat.fa
mu3<-mean(y3)
hist(y3)

#Assuming normality and low bias we can use the following:
se3<-sd(mat.fa)
lbound3<-mu3-1.96*se3
ubound3<-mu3+1.96*se3
lbound3
ubound3

#Bootstrapped t-confidence interval
lbound3a<-mu3-qt(.975,n-m-1)*se3
ubound3a<-mu3+qt(.975,n-m-1)*se3
lbound3a
ubound3a

#Using quantiles

```

```

per97.5c <-quantile(y3,.975)
per2.5c <- quantile(y3,.025)
per2.5c
per97.5c
#matrix representation of Confidence Intervals and estimates

#Method 2 calculations
intervals<-matrix(c(mua,lbjack,ubjack,ubjack-lbjack,(ubjack-mua)/(mua-
lbjack),mub,lbjackb,ubjackb,ubjackb-lbjackb,(ubjackb-mub)/(mub-
lbjackb),mu,
      lbound,ubound,ubound-lbound,(ubound-mu)/(mu-
lbound),mu,lbounda,ubounda,ubounda-lbounda,(ubounda-mu)/(mu-lbounda),
      mu,per2.5,per97.5,per97.5-per2.5,(per97.5-mu)/(mu-
per2.5),mu,lboundb,uboundb,uboundb-lboundb,(uboundb-mu)/(mu-lboundb)),
      nrow=5,ncol=6,dimnames=list(c("mu","lb","ub","length","shape"),
      c("jackknife1","jackknife2","normality","t-
confidence","quantiles","a-bias_correction")))
intervals

#Method 4 calculations
intervals1<-matrix(c(mula,lbjack1,ubjack1,ubjack1-lbjack1,(ubjack1-
mula)/(mula-lbjack1),mulb,lbjack1b,ubjack1b,ubjack1b-
lbjack1b,(ubjack1b-mulb)/(mulb-lbjack1b),mul1,
      lbound1,ubound1,ubound1-lbound1,(ubound1-mu1)/(mu1-
lbound1),mul1,lbound1a,ubound1a,ubound1a-lbound1a,(ubound1a-mu1)/(mu1-
lbound1a),mul1,per2.5a,per97.5a,per97.5a-per2.5a,
      (per97.5a-mu1)/(mu1-per2.5a),mul1,lbound1b,ubound1b,ubound1b-
lbound1b,(ubound1b-mu1)/(mu1-lbound1b)),
      nrow=5,ncol=6,dimnames=list(c("mu","lb","ub","length","shape"),
      c("jackknife1","jackknife2","normality","t-
confidence","quantiles",
      "a-bias_correction")))
intervals1

#Estimator calculations
intervals2<-matrix(c(mu2,lbound2,ubound2,ubound2-lbound2,(ubound2-
mu2)/(mu2-lbound2),
      mu2,lbound2a,ubound2a,ubound2a-lbound2a,(ubound2a-mu2)/(mu2-
lbound2a),mu2,per2.5b,per97.5b,per97.5b-per2.5b,
      (per97.5b-mu2)/(mu2-per2.5b)),
      nrow=5,ncol=3,dimnames=list(c("mu","lb","ub","length","shape"),
      c("normality","t-confidence","quantiles")))
intervals2

#New Model calculations
intervals3<-matrix(c(mu3,lbound3,ubound3,ubound3-lbound3,(ubound3-
mu3)/(mu3-lbound3),
      mu3,lbound3a,ubound3a,ubound3a-lbound3a,(ubound3a-mu3)/(mu3-
lbound3a),mu3,per2.5c,per97.5c,per97.5c-per2.5c,
      (per97.5c-mu3)/(mu3-per2.5c)),
      nrow=5,ncol=3,dimnames=list(c("mu","lb","ub","length","shape"),
      c("normality","t-confidence","quantiles")))
intervals3

#Estimator load estimate
hat<-estfunc$coefficients

```

```

estim<-
hat[1]+hat[2]*ydata[j:k,4]+hat[3]*ydata[j:k,5]+hat[4]*ydata[j:k,6]+hat[
5]*ydata[j:k,7]+hat[6]*ydata[j:k,8]+hat[7]*ydata[j:k,9]
var11<-var(estim)
estimatorest<-
2.44658*sum(ydata[j:k,2]*exp(hat[1]+hat[2]*ydata[j:k,4]+hat[3]*ydata[j:
k,5]+hat[4]*ydata[j:k,6]+hat[5]*ydata[j:k,7]+hat[6]*ydata[j:k,8]+hat[7]
*ydata[j:k,9]+var11/2))
estimatorest

#Our model load estimate
estim1<-estfunc1$coefficients
new<-estim1[1]+
estim1[2]*ydata[j:k,10]+estim1[3]*ydata[j:k,11]+estim1[4]*ydata[j:k,12]
+estim1[5]*ydata[j:k,13]+estim1[6]*ydata[j:k,14]+estim1[7]*ydata[j:k,15
]+estim1[8]*ydata[j:k,16]+estim1[9]*ydata[j:k,17]+estim1[10]*ydata[j:k,
18]
var12<-var(new)
news<-2.44657*sum(ydata[j:k,2]*exp(estim1[1]+
estim1[2]*ydata[j:k,10]+estim1[3]*ydata[j:k,11]+estim1[4]*ydata[j:k,12]
+estim1[5]*ydata[j:k,13]+estim1[6]*ydata[j:k,14]+estim1[7]*ydata[j:k,15
]+estim1[8]*ydata[j:k,16]+estim1[9]*ydata[j:k,17]+estim1[10]*ydata[j:k,
18]
+var12/2))

```