

ESTIMATING DISTRIBUTION FUNCTIONS FROM SURVEY DATA  
USING NONPARAMETRIC REGRESSION

Submitted by  
Alicia Johnson  
Statistics

In partial fulfillment of the requirements  
for the Degree of Master of Science  
Colorado State University  
Fort Collins, Colorado  
Spring 2003

# ESTIMATING DISTRIBUTION FUNCTIONS FROM SURVEY DATA USING NONPARAMETRIC REGRESSION

Alicia Johnson, Colorado State University

May 1, 2003

## **Abstract**

Survey sampling often supplies information about a study variable only for sampled elements. However, auxiliary information is often available for the entire population. The relationship of the auxiliary information with the study variable across the sample allows inferences about the nonsampled portion of the population. Thus, auxiliary information can be used to improve upon survey estimation. In particular, finite population distribution function estimation can be improved. Existing parametric estimators incorporate auxiliary information by assuming it to have a linear relationship with the study variable, which is often unreasonable or unverifiable in survey sampling. A model-assisted nonparametric estimator based on local polynomial regression is introduced which removes these parametric restrictions by working under a more generic context. A Monte Carlo comparison of this estimator with parametric estimators demonstrates its superior efficiency for estimation of the distribution function and quantiles in most cases in which the parametric methods misspecify the relationship between the auxiliary and study variables. Finally, a semiparametric estimator is introduced which allows for the incorporation of parametric fixed effects, including categorical variables. When applied to a population of Northeastern lakes, this estimator is more efficient on average than a more conventional estimator that does not incorporate any auxiliary information.

## **Acknowledgements**

The work reported here was developed under the STAR Research Assistance Agreement CR-829095 awarded by the U.S. Environmental Protection Agency (EPA) to Colorado State University. This presentation has not been formally reviewed by EPA. The views expressed here are solely those of authors and STARMAP. EPA does not endorse any products or commercial services mentioned in this report.

# Contents

<b>1</b>	<b>Introduction</b>	<b>6</b>
<b>2</b>	<b>Nonparametric Regression and Extensions</b>	<b>11</b>
2.1	Overview . . . . .	11
2.2	Local Polynomial Regression . . . . .	12
2.2.1	General local polynomial regression . . . . .	12
2.2.2	Application to estimators in survey sampling . . . . .	13
2.2.3	Application to finite population cdf estimation . . . . .	14
2.3	Semiparametric Estimation . . . . .	14
2.3.1	General semiparametric estimation . . . . .	14
2.3.2	Application to estimators in survey sampling . . . . .	16
2.3.3	Application to finite population cdf estimation . . . . .	16
<b>3</b>	<b>Simulations</b>	<b>17</b>
3.1	CDF estimation . . . . .	17
3.2	Quantile Estimation . . . . .	25
<b>4</b>	<b>Empirical Results</b>	<b>26</b>
<b>5</b>	<b>Conclusion</b>	<b>34</b>
	<b>References</b>	<b>36</b>
	<b>Appendix I: graphs and tables</b>	<b>37</b>
	<b>Appendix II: SPlus code</b>	<b>46</b>
	CDF estimation simulation . . . . .	47
	Quantile estimation simulation . . . . .	54
	Semiparametric cdf estimation . . . . .	63

## List of Tables

1	<b>Summary of MSE ratios</b> . . . . .	20
2	MSE ratios for CDF estimation calculated at the median of HT, CD0, CD1, RKM0, RKM1, and DORF to local polynomial regression estimator (LPR). . . . .	40
3	Percent relative biases for cdf estimators HT, CD0, CD1, RKM0, RKM1, LPR, and DORF calculated at the median . . . . .	41
4	MSE ratios for CDF estimation calculated at the first quartile of HT, CD0, CD1, RKM0, RKM1, and DORF to local polynomial regression estimator (LPR). . . . .	42
5	Percent relative biases for cdf estimators HT, CD0, CD1, RKM0, RKM1, LPR, and DORF calculated at the first quartile . . . . .	43
6	MSE ratios for estimation of the median of estimators HT, CD0, CD1, RKM0, RKM1, and DORF to local polynomial regression estimator (LPR). . . . .	44
7	Percent relative biases for quantile estimators HT, CD0, CD1, RKM0, RKM1, LPR, and DORF calculated at the median . . . . .	45

# 1 Introduction

In finite population sampling, we are often concerned with the proportion of values  $y_i$  in the finite population,  $U = \{1, \dots, i, \dots, N\}$ , that are bounded by a given constant. For example, we may be interested in the proportion of lakes in acid-sensitive regions of the Northeast United States with acid neutralizing capacity less than zero. Such a proportion is one particular value of the cumulative distribution function (cdf) for the finite population,  $F(t) = N^{-1} \sum_{i \in U} I_{\{y_i \leq t\}}$ . This is just the average of the indicator function  $I_{\{y_i \leq t\}}$  over all elements of the finite population where  $I_{\{y \leq t\}} = 1$  for  $y \leq t$  and  $I_{\{y \leq t\}} = 0$  for all  $y > t$ . Often in survey sampling, we can only measure the study variable for those items in some sample,  $A \subset U$ . Thus, standard estimators of the distribution function depend solely on the selection of the sampling design and the sampled portion of the population.

Although it is often the case in survey sampling that measurements of the study variable are unavailable for the nonsampled portion of the population, we may have auxiliary information available for the entire population. Thus, by incorporating this auxiliary information we can improve our estimation of the finite population distribution function. Parametric methods do exist which utilize the available auxiliary information by assuming some superpopulation meant to represent the relationship between the auxiliary data and the study variable. Therefore, these parametric methods are able to incorporate inferences about the nonsampled portion of the population into the estimation of the finite population distribution function. In most cases, parametric methods place restrictions on the relationship between the auxiliary data and the study variable by assuming the superpopulation to be built upon a linear process. However, the assignment of such a relationship in survey sampling is often inappropriate or unverifiable. Also, we may not have the resources to perform diagnostics for multiple study variables thus increasing the susceptibility of misspecifying the superpopulation. To address the situation in which parametric methods misspecify the superpopulation, we will introduce a model-assisted nonparametric regression approach to the estimation of the distribution function, which does not place any restrictions on the relationship between the auxiliary data and the study variable.

Let  $y_i$  be our study variable for finite population  $U$  from which we take a sample  $A \subset U$  of size  $n$ . The chosen sample design determines inclusion probability  $\pi_i$ , the probability that element  $i$  is included in the sample, and second-order inclusion probability  $\pi_{ij}$ , the probability that elements  $i$  and  $j$

are both members of the sample. We assume that both  $\pi_i$  and  $\pi_{ij}$  are  $> 0$ . Let  $U - A$  represent the portion of the population not included in sample  $A$ . Suppose  $y_i$  is only available for  $i \in A$  and we are interested in estimating the finite population cdf,  $F(t)$ .

A conventional estimator of  $F(t)$  is the Horvitz-Thompson estimator

$$\hat{F}_{HT}(t) = \frac{1}{N} \sum_{i \in A} \frac{I_{\{y_i \leq t\}}}{\pi_i} \quad (1)$$

(Horvitz and Thompson (1952)), which is design unbiased and does not depend on any model. Rather, it only depends on the selection of the sampling design which in turn determines the values of inclusion probabilities  $\pi_i$ . Since  $y_i$  is unknown for  $i \in U - A$ , the Horvitz-Thompson estimator is built solely upon information supplied by the sample.

We can improve upon the conventional Horvitz-Thompson estimator by incorporating auxiliary information  $x_i$  available for all  $i \in U$ . We assume a superpopulation for  $y$  built on some mean function of  $x$  and allowing for heteroskedastic error terms:

$$y_i = m(x_i) + v^{1/2}(x_i)\epsilon_i. \quad (2)$$

The  $\epsilon_i$  are independent and identically distributed for some distribution function  $G$  with  $E(\epsilon_i) = 0$  and  $\text{Var}(\epsilon_i) = \sigma^2$ . This assumed relationship between our study variable,  $y_i$ , and auxiliary information,  $x_i$ , allows us to make inferences about  $y_i$ ,  $i \in U - A$ , since  $x_i$  is available for the entire population. Incorporating these inferences into our estimation of  $F(t)$  will provide estimators that integrate information from both sampled and nonsampled portions of the population.

Existing methods that utilize such superpopulations and the incorporation of auxiliary information in finite population cdf estimation are based on parametric models. These methods assume superpopulation (2) has linear mean function and also that  $v^{1/2}(x_i)$  is known and strictly positive, i.e.

$$y_i = \beta_0 + \beta_1 x_i + v^{1/2}(x_i)\epsilon_i \quad (3)$$

where estimates of  $\beta_0$  and  $\beta_1$  are calculated using the method of weighted least squares, resulting in an estimate of the mean function which can be written as  $\hat{m}(x_i) = \hat{\beta}_0 + \hat{\beta}_1 x_i$ .

Under the assumption of linear superpopulation (3), the model-based Chambers and Dunstan (1986) estimator is

$$\hat{F}_{CD}(t) = \frac{1}{N} \left[ \sum_{j \in A} I_{\{y_j \leq t\}} + \sum_{i \in U-A} \hat{G}_i \right] \quad (4)$$

where  $\hat{G}_i = n^{-1} \sum_{j \in A} I_{\{\hat{r}_j(y_j) \leq \hat{r}_i(t)\}}$  for  $\hat{r}_i(t) = (t - \hat{m}(x_i))v^{-1/2}(x_i)$ , an estimate of  $r_i(t) = (t - m(x_i))v^{-1/2}(x_i)$ . This  $\hat{G}_i$  estimates  $G_i$ , the model expectation of  $I_{\{y_i \leq t\}}$ , which can be written as  $G_i(r_i(t)) = E_m I_{y_i \leq t} = P(y_i \leq t) = P(\epsilon_i \leq r_i(t))$ . Under the appropriateness of a linear mean function and correctly specified variance term, the Chambers and Dunstan (1986) estimator is asymptotically unbiased. However, since this estimator operates under a strict context by placing such emphasis on these conditions being met, any deviation from these specific requirements cause the model-based Chambers and Dunstan (1986) estimator to become biased.

Under the restrictions of linear superpopulation (3), Rao, Kovar, and Mantel (1990) propose a model-assisted, parametric approach to finite population cdf estimation:

$$\hat{F}_{RKM}(t) = \frac{1}{N} \sum_{i \in U} \tilde{G}_i + \frac{1}{N} \sum_{i \in A} \frac{I_{\{y_i \leq t\}} - \tilde{G}_{ic}}{\pi_i} \quad (5)$$

where

$$\begin{aligned} \tilde{G}_i &= \left( \sum_{j \in A} \frac{1}{\pi_j} \right)^{-1} \left[ \sum_{j \in A} \frac{1}{\pi_j} I_{\{r_j(y_j) \leq r_i(t)\}} \right] \\ \tilde{G}_{ic} &= \left( \sum_{j \in A} \frac{\pi_i}{\pi_{ij}} \right)^{-1} \left[ \sum_{j \in A} \frac{\pi_i}{\pi_{ij}} I_{\{r_j(y_j) \leq r_i(t)\}} \right]. \end{aligned}$$

It can be seen that  $\tilde{G}_{ic}$  is just  $\tilde{G}_i$  weighted with conditional probabilities  $\pi_{ij}/\pi_i = Pr\{i, j \in A | i \in A\}$ . This model-assisted estimator (5) consists of a model-based prediction plus a design-bias adjustment and therefore does not operate under as strict conditions as the Chambers and Dunstan (1986) estimator. Rao, Kovar, and Mantel (1990) showed this estimator to be both asymptotically design and model unbiased under all degrees of variance misspecification. Yet, despite the asymptotic design unbiasedness,  $\hat{F}_{RKM}$  remains under the tight restrictions of a parametric model and when the true mean function is nonlinear,  $\hat{F}_{RKM}$  will be an inefficient estimator of the finite population distribution function.

Parametric methods are a reasonable alternative to the Horvitz-Thompson estimator under the specific context and appropriateness of a linear mean

function and known variance term. However, if the true mean function,  $m(x_i)$ , is nonlinear or the variance term is misspecified, parametric restrictions will cause the model-based Chambers and Dunstan estimator to be biased and the model-assisted Rao, Kovar, and Mantel estimator to be inefficient. The susceptibility of parametric methods to mean function and variance misspecification bias provides motivation to explore alternative methods that do not place such restrictions on superpopulation model (2). Fitting (2) using nonparametric methods removes this parametric straitjacket and can therefore be applied to a more generic context in which no model is assumed to be correct or defensible. In fact, the only assumption placed on the superpopulation model is that mean function  $m(x_i)$  is some smooth function of the auxiliary data  $x_i$  and that  $v(x_i)$  is smooth and strictly positive.

One nonparametric approach is to adapt the model-based nonparametric finite population total estimator proposed by Dorfman (1992) to the case of cdf estimation by replacing  $y_i$  with  $I_{\{y_i \leq t\}}$ . The general form of this estimator is

$$\hat{F}_{DORF}(t) = \frac{1}{N} \sum_{i \in A} I_{\{y_i \leq t\}} + \frac{1}{N} \sum_{j \in U-A} \hat{g}(x_j) \quad (6)$$

where  $\hat{g}(x_j)$  is a model-based nonparametric estimator of  $g(x_j)$ , the kernel smooth estimator of  $G \left\{ (t - m(x_i))v^{-1/2}(x_i) \right\}$ .

Although this nonparametric model-based estimator removes the restrictions imposed by parametric estimation, we can go one step further. Section 2.1 includes a brief overview of model-assisted nonparametric regression estimation which not only removes the restrictions of parametric estimation, it does not even make the assumption that the nonparametric model always holds and is therefore appropriate for the most generic context in which no model is assumed to be correct or defensible. Section 2.2 extends nonparametric regression to local polynomial regression and its application to estimators in survey sampling. This, in turn, may be adapted to form an estimator of finite population distribution functions which is of the form

$$\hat{F}_{LPR}(t) = \frac{1}{N} \sum_{i \in U} \hat{g}_i(t) + \frac{1}{N} \sum_{i \in A} \frac{I_{\{y_i \leq t\}} - \hat{g}_i(t)}{\pi_i} \quad (7)$$

where  $\hat{g}(x_j)$  is a model-assisted nonparametric estimator of  $g(x_j)$  as before. Section 3.1 presents the results of a Monte Carlo comparison of the local polynomial regression approach to cdf estimation introduced in section 2.2.3 to the existing parametric methods as well as the model-based nonparametric

Dorfman estimator. It will be shown that the efficiency of the local polynomial regression estimator is essentially competitive with or outperforms the existing parametric estimators for the cases in which parametric estimators misspecify the mean function. However,  $\hat{F}_{LPR}$  is slightly less efficient for the cases in which a linear mean function is appropriate.

Another topic of interest closely related to the estimation of the finite population cdf is that of quantile estimation. For  $\theta(\alpha) = \min\{t : F(t) \geq \alpha\}$ , the  $\alpha$ -quantile of  $y$ , we can write an estimate  $\hat{\theta}(\alpha)$  as

$$\hat{\theta}(\alpha) = \min\{t : \hat{F}(t) \geq \alpha\} \quad (8)$$

where  $\hat{F}(t)$  is based on some finite population cdf estimator. The estimation of  $\theta(\alpha)$  weighs heavily on  $\hat{F}(t)$ , thus choosing a parametric method for  $\hat{F}(t)$  causes  $\hat{\theta}(\alpha)$  to be vulnerable to the same misspecification biases as previously discussed in the case of the parametric approach to finite population cdf estimation. A Monte Carlo comparison of the different approaches to quantile estimation is included in section 3.2 with similar results to the Monte Carlo comparison of cdf estimators in section 3.1.

The discussed cdf estimators have only addressed situations in which one continuous auxiliary variable is available. In many surveys however, more than one auxiliary variable may be available for the population. In the case that these additional auxiliary variables are continuous, the local polynomial regression estimator can be extended to include multiple variables with little difficulty. However, it is often the case that the available auxiliary variables may either be categorical in nature or else the covariate space contains a hole. To address this situation, section 2.3 extends nonparametric regression estimation to semiparametric estimation which combines both parametric and nonparametric methodology. Superpopulation (2) can be rewritten to address the situation in which one auxiliary variable is treated nonparametrically and the others are treated parametrically:

$$\begin{aligned} y_i &= g(x_i, \mathbf{z}_i) + v^{1/2}(x_i)\epsilon_i \\ &= m(x_i) + \mathbf{z}_i\boldsymbol{\beta} + v^{1/2}(x_i)\epsilon_i \end{aligned} \quad (9)$$

where  $\epsilon_i$  and  $v(x_i)$  are as before. Section 2.3 introduces Breidt and Opsomer's (working paper) semiparametric approach to survey sampling estimation and also its extension to finite population distribution function estimation which is of the form

$$\hat{F}_{SEMI}(t) = \frac{1}{N} \sum_{i \in U} \hat{g}(x_i, \mathbf{z}_i) + \frac{1}{N} \sum_{i \in A} \frac{I_{\{y_i \leq t\}} - \hat{g}(x_i, \mathbf{z}_i)}{\pi_i} \quad (10)$$

where  $\hat{g}(x_i, \mathbf{z}_i)$  is a model-assisted nonparametric estimator of  $g(x_i, \mathbf{z}_i)$ , the kernel smooth estimator of  $G\{(t - m(x_i) - \mathbf{z}_i\boldsymbol{\beta})v^{-1/2}(x_i)\}$ .

Empirical results of applying the semiparametric estimator,  $\hat{F}_{SEMI}$ , and the more conventional Horvitz-Thompson estimator,  $\hat{F}_{HT}$ , to an actual population of Northeastern Lakes will be discussed in section 4. The estimated proportion of acidic Northeastern lakes is calculated by both estimators and it will be shown that the semiparametric estimator is more efficient, on average, than the Horvitz-Thompson estimator which does not incorporate any auxiliary information.

## 2 Nonparametric Regression and Extensions

### 2.1 Overview

Assume superpopulation (2) holds and we would like to estimate mean function,  $m(x)$ . As discussed, parametric methods base estimation on an assumed linear mean function between  $y$  and auxiliary information,  $x$ , that is fit to the entire population based on the given auxiliary information and  $y_i, i \in A$ . These restrictions on the true form of the mean function leave parametric estimators vulnerable to mean function misspecification bias.

Instead of assuming a linear mean function and estimating  $m(x)$  based on the entire population of  $x_i, i \in U$ , nonparametric methods approach the same problem by locally approximating  $m(x)$ , thereby removing the parametric straitjacket. Local approximation of the mean function is achieved by placing more weight on the  $y_i$ 's corresponding to  $x_i$ 's which are closer to  $x$ . These weights are assigned according to a scaled kernel function centered at  $x$ . The kernel function is scaled by a specified bandwidth which determines the spread of the kernel. In general, if the kernel function is scaled by a large bandwidth, this increases its spread and therefore increases the influence of  $x_i$  further away from  $x$ . For bandwidths that are very large, say as wide as the range of the auxiliary variable, nonparametric regression fits will become equivalent to those made by parametric methods specifying a linear mean function. Thus, choosing a large bandwidth may result in oversmoothing of the mean function. On the other hand, if the bandwidth is small, the kernel is based on few observations near  $x$  and may result in undersmoothing of the mean function.

In order to obtain a nonparametric regression estimation of mean func-

tion,  $m(x)$ , evaluating local approximations of  $m(x)$  for each point  $x \in [x_{min}, x_{max}]$  will result in a solid curve. Another, quicker option more easily applied to survey sampling situations is to create a grid of  $x_i$ 's on which to evaluate local approximations of  $m(x_i)$ , then to linear interpolate between grid points. Another alternative to linear interpolation is to use a piecewise constant which assigns the same approximation to  $m(x)$  for all  $x_i$  which are no more than one-half the grid spacing away from  $x$ .

Section 2.2 extends nonparametric regression to local polynomial regression in which a polynomial of degree,  $q$ , is fit for each  $x_i$  on the grid instead of the more conventional approach in which a line is fit to  $x$  and  $y$  at each grid point. It also introduces an application of local polynomial regression estimation to finite population distribution functions as an extension of survey sampling estimation. Section 2.3 follows with an extension to semiparametric estimation and its application to survey sampling estimation. More specifically, it introduces a semiparametric approach to finite population cdf estimation incorporating both local polynomial regression and parametric estimation.

## 2.2 Local Polynomial Regression

### 2.2.1 General local polynomial regression

Consider the case in which we have  $K$ , some kernel function of degree  $q$ , with bandwidth  $h$ . We then define the  $N \times (q + 1)$  population design matrix

$$\mathbf{X}_{U_i} = \left[ \begin{array}{cccc} 1 & x_j - x_i & \cdots & (x_j - x_i)^q \end{array} \right]_{j \in U},$$

and the  $N \times N$  population weighting matrix

$$\mathbf{W}_{U_i} = \text{diag} \left\{ \frac{1}{h} K \left( \frac{x_j - x_i}{h} \right) \right\}_{j \in U}$$

where  $x_i$  comes from vector  $\mathbf{x}_U = [x_i]_{i \in U}$ , the vector of  $x_i$ 's for  $i \in U$ . We can now write the local polynomial regression population smoother vector at observation  $x_i$  as

$$\mathbf{s}'_{U_i} = \mathbf{e}'_1 (\mathbf{X}'_{U_i} \mathbf{W}_{U_i} \mathbf{X}_{U_i})^{-1} \mathbf{X}'_{U_i} \mathbf{W}_{U_i}$$

where  $\mathbf{e}_r$  equals the  $r$ th unit vector which places a 1 in the  $r$ th position and 0 elsewhere. Applying  $\mathbf{s}'_{U_i}$  to  $\mathbf{y}_U = [y_i]_{i \in U}$  results in a finite population kernel smooth regression fit at  $x_i$ .

The calculation of  $\mathbf{s}'_{U_i}$  requires the knowledge of  $y_i$  for the entire population yet  $y_i$  is only available for  $i \in A$ . Therefore we must find an estimate  $\mathbf{s}'_{A_i}$  that only incorporates information from the sampled portion of the population. For  $\mathbf{x}_A = [x_i]_{i \in A}$ , the vector of sampled  $x_i$ , we have the corresponding vector  $\mathbf{y}_A = [y_i]_{i \in A}$ . To restrict  $\mathbf{X}_{U_i}$  and  $\mathbf{W}_{U_i}$  to corresponding elements in  $A$ , define the  $n \times (q + 1)$  sample design matrix

$$\mathbf{X}_{A_i} = \left[ \begin{array}{cccc} 1 & x_j - x_i & \cdots & (x_j - x_i)^q \end{array} \right]_{j \in A},$$

and the  $n \times n$  sample weighting matrix

$$\mathbf{W}_{A_i} = \text{diag} \left\{ \frac{1}{h} K \left( \frac{x_j - x_i}{h} \right) \frac{1}{\pi_j} \right\}_{j \in A}$$

for  $i \in A$ . We can now write a sample smoother vector at  $x_i$ ,

$$\mathbf{s}'_{A_i} = \mathbf{e}'_1 (\mathbf{X}'_{A_i} \mathbf{W}_{A_i} \mathbf{X}_{A_i})^{-1} \mathbf{X}'_{A_i} \mathbf{W}_{A_i} \mathbf{y}_A,$$

which eliminates any dependence on  $i \notin A$  and when applied to  $\mathbf{y}_A$  provides a sample estimate of the population kernel smooth regression fit at  $x_i$ .

### 2.2.2 Application to estimators in survey sampling

A local polynomial regression approach to population total estimation is introduced by Breidt and Opsomer (2000). Denote the finite population total as  $T_y = \sum_{i \in U} y_i$ . Then the local polynomial regression estimator of  $T_y$  can be written as

$$\hat{T}_{LPR} = \sum_{i \in U} \hat{m}_i + \sum_{i \in A} \frac{y_i - \hat{m}_i}{\pi_i} \quad (11)$$

where  $\hat{m}_i = \mathbf{s}'_{A_i} \mathbf{y}_A$ , the sample local polynomial regression fit at observation  $i$ . It can be shown that  $\hat{T}_{LPR}$  has estimated variance

$$\widehat{\text{Var}}(\hat{T}_{LPR}) = \sum_{i, j \in A} (y_i - \hat{m}_i)(y_j - \hat{m}_j) \frac{\pi_{ij} - \pi_i \pi_j}{\pi_i \pi_j} \frac{1}{\pi_{ij}}$$

and is also design consistent and asymptotically design unbiased.

### 2.2.3 Application to finite population cdf estimation

The Breidt and Opsomer finite population total estimator, (11), can be adapted to the estimation of the finite population cdf by replacing  $y_i$  with  $I_{\{y_i \leq t\}}$  and multiplying by a factor of  $1/N$ . Using notation introduced in section 2.2.1 we write a sample estimate of the  $q$ th degree finite population kernel smooth of  $G\{(t - m(x_i))v^{-1/2}(x_i)\}$  as

$$\hat{m}_i = \mathbf{s}'_{Ai} \mathbf{I}_A$$

for  $\mathbf{I}_A = [I_{\{y_i \leq t\}}]$  corresponding to  $i \in A$ . This sample estimator  $\hat{m}_i$  can be shown to be design consistent for  $m_i = \mathbf{s}'_{Ui} \mathbf{I}_U$ . Now we can write the model-assisted local polynomial regression estimator (LPR) of the finite population cdf as

$$\hat{F}_{LPR}(t) = \frac{1}{N} \sum_{i \in U} \hat{m}_i + \frac{1}{N} \sum_{i \in A} \frac{I_{\{y_i \leq t\}} - \hat{m}_i}{\pi_i}. \quad (12)$$

It can be seen that this model-assisted estimator is built upon a model-based nonparametric prediction with a design-bias adjustment and can also be written in a weighted form,

$$\hat{F}_{LPR}(t) = \sum_{i \in s} w_{is} I_{\{y_i \leq t\}}.$$

Since weights,  $w_{is}$ , are calculated independently of study variable  $y_i$ , they allow for generic inference in that they are easily applied to any variable of interest.

We can similarly adjust the estimated variance of the finite population total. Thus,  $\hat{F}_{LPR}(t)$  has estimated variance

$$\widehat{\text{Var}}(\hat{F}_{LPR}(t)) = \frac{1}{N^2} \sum_{i,j \in A} (I_{\{y_i \leq t\}} - \hat{m}_i)(I_{\{y_j \leq t\}} - \hat{m}_j) \frac{\pi_{ij} - \pi_i \pi_j}{\pi_i \pi_j} \frac{1}{\pi_{ij}}.$$

The design properties of  $\hat{T}_{LPR}$ , design consistency and asymptotic design unbiasedness, hold for this cdf estimator,  $\hat{F}_{LPR}$ , from the results in Breidt and Opsomer (2000).

## 2.3 Semiparametric Estimation

### 2.3.1 General semiparametric estimation

To address situations in which multiple auxiliary variables are available of which some may be categorical, redefine superpopulation (2) to include a

term that may incorporate these additional variables. Recall the adjusted superpopulation (9) which addresses these additional auxiliary variables:

$$\begin{aligned} y_i &= g(x_i, \mathbf{z}_i) + v^{1/2}(x_i)\epsilon_i \\ &= m(x_i) + \mathbf{z}_i\boldsymbol{\beta} + v^{1/2}(x_i)\epsilon_i \end{aligned}$$

where  $\epsilon_i$  and  $v(x_i)$  are as before. Here,  $x_i$  represents a single continuous auxiliary variable and  $\boldsymbol{\beta}$  is a parameter vector corresponding to  $\mathbf{z}_i = (1, z_{1i}, z_{2i}, \dots, z_{Di})$ , a vector of  $D + 1$  auxiliary variables. We can organize these  $D + 1$  auxiliary variables into a matrix  $\mathbf{Z}_U = (\mathbf{1}', \mathbf{z}'_1, \dots, \mathbf{z}'_N)'$ .

We are able to extend local polynomial regression to include multiple auxiliary variables when all of the variables are continuous. However, local polynomial regression estimation can not handle categorical variables and may result in the potential loss of information when the covariate space contains holes. It is also possible to extend the parametric estimators. However, survey sampling does not always allow for diagnostics to be performed on all variables, thus parametric estimators are susceptible to misspecifying the relationships between some of the auxiliary variables with the study variable. Therefore it is beneficial to be able to model one of these variables nonparametrically in order to maintain some of the flexibility of nonparametric model specification. The superpopulation is then nonparametric in  $x_i$ , where  $x_i$  is continuous, and parametric in  $\mathbf{z}_i$ . Estimation based on this merging of parametric and nonparametric methodology is called semiparametric estimation.

Assume superpopulation (9) holds and define  $\mathbf{X}_{U_i}$ ,  $\mathbf{W}_{U_i}$ , and population smoother vector  $\mathbf{s}'_{U_i}$  as in Section 2.2.1. Define the smoother matrix,  $\mathbf{S}_U$ , as  $\mathbf{S}_U = [\mathbf{s}'_{U_i} : i \in U]$ . From section 2.2.1 we know that applying  $\mathbf{s}'_{U_i}$  to  $\mathbf{y}_U$  results in a kernel smooth regression fit at  $x_i$ , thus applying smoother matrix  $\mathbf{S}_U$  to  $\mathbf{y}_U$  results in a vector of population kernel smooth fits at all observation points  $x_1, x_2, \dots, x_N$ . Also define the centered smoother matrix,  $\mathbf{S}_U^* = (\mathbf{I} - \mathbf{1}\mathbf{1}'/N)\mathbf{S}_U$ , which subtracts the mean of the fitted values from each fitted value and therefore prevents the intercept from being estimated twice. Based on these smoothing matrices and for any given observation  $x_i$ ,  $i \in U$ , we can now write population estimates of the parameter vector,  $\boldsymbol{\beta}$ , and nonparametric component,  $m(x_i)$ :

$$\begin{aligned} \mathbf{B} &= (\mathbf{Z}'_U(\mathbf{I} - \mathbf{S}_U^*)\mathbf{Z}_U)^{-1}\mathbf{Z}'_U(\mathbf{I} - \mathbf{S}_U^*)\mathbf{y}_U \\ m_i &= \mathbf{s}'_{U_i}(\mathbf{y}_U - \mathbf{Z}_U\mathbf{B}). \end{aligned}$$

It can be seen that the parametric residual vector, as opposed to  $\mathbf{y}_U$ , is smoothed with respect to  $x$  in order to account for the parametric component of the  $y_i$ .

In order to calculate  $\mathbf{B}$  and  $m_i$ ,  $y_i$  must be known for all  $i \in U$ . However,  $y_i$  is only available for  $i \in A$  so we must adjust these estimates accordingly. From section 2.2.1 we have  $\mathbf{X}_{Ai}$ ,  $\mathbf{W}_{Ai}$ , and sample smoother vector,  $\mathbf{s}_{Ai}$  which restrict  $\mathbf{X}_{Ui}$ ,  $\mathbf{W}_{Ui}$ , and  $\mathbf{s}_{Ui}$  respectively to items in the sample. Similarly, restrict  $\mathbf{Z}_U$  and smoother matrices  $\mathbf{S}_U$  and  $\mathbf{S}_U^*$  to corresponding sampled elements. This results in the sample smoother matrix and sample centered smoother matrix,

$$\mathbf{S}_A = [\mathbf{s}'_{Ai} : i \in A] \quad \text{and} \quad \mathbf{S}_A^* = (\mathbf{I} - \mathbf{1}\mathbf{1}'\mathbf{\Pi}_A^{-1}/N)\mathbf{S}_A,$$

where  $\mathbf{\Pi}_A = \text{diag}\{\pi_i : i \in A\}$ . Utilizing these sample restricted matrices, design-weighted estimators of  $\mathbf{B}$  and  $m_i$  are as follows:

$$\begin{aligned} \hat{\mathbf{B}} &= (\mathbf{Z}'_A \mathbf{\Pi}_A^{-1} (\mathbf{I} - \mathbf{S}_A^*) \mathbf{Z}_A)^{-1} \mathbf{Z}'_A \mathbf{\Pi}_A^{-1} (\mathbf{I} - \mathbf{S}_A^*) \mathbf{y}_A \\ \hat{m}_i &= \mathbf{s}'_{Ai} (\mathbf{y}_A - \mathbf{Z}_A \hat{\mathbf{B}}). \end{aligned} \quad (13)$$

### 2.3.2 Application to estimators in survey sampling

Breidt and Opsomer (working paper) apply this semiparametric approach to survey sampling estimation and introduce a semiparametric estimator of the finite population total. This estimator can be written as

$$\hat{T}_{SEMI} = \sum_{i \in U} \hat{g}(x_i, \mathbf{z}_i) + \sum_{i \in A} \frac{y_i - \hat{g}(x_i, \mathbf{z}_i)}{\pi_i} \quad (14)$$

where  $\hat{g}(x_i, \mathbf{z}_i) = \hat{m}_i + \mathbf{z}_i \hat{\mathbf{B}}$ . The semiparametric estimator of the finite population total can be shown to be asymptotically design unbiased and design consistent. Its estimated variance is

$$\widehat{\text{Var}}(\hat{T}_{SEMI}) = \sum_{i, j \in A} \frac{\pi_{ij} - \pi_i \pi_j}{\pi_{ij}} \frac{y_i - \hat{g}_i}{\pi_i} \frac{y_j - \hat{g}_j}{\pi_j}.$$

### 2.3.3 Application to finite population cdf estimation

As previously discussed, it is beneficial to approach finite population distribution function estimation nonparametrically. Yet, if we have a set of auxiliary variables which includes at least one categorical variable or the co-variate space includes an empty hole, we may choose to estimate the cdf semiparametrically. This situation can be represented with superpopulation

(9). The Breidt and Opsomer semiparametric approach to population total estimation, (14), can be adapted to the semiparametric estimation of the finite population cdf by replacing  $y_i$  with  $I_{\{y_i \leq t\}}$  as we did for the local polynomial regression cdf estimator in section 2.2. Thus,

$$\begin{aligned}\hat{\mathbf{B}} &= (\mathbf{Z}'_A \mathbf{\Pi}_A^{-1} (\mathbf{I} - \mathbf{S}_A^*) \mathbf{Z}_A)^{-1} \mathbf{Z}'_A \mathbf{\Pi}_A^{-1} (\mathbf{I} - \mathbf{S}_A^*) \mathbf{I}_A \\ \hat{m}_i &= \mathbf{s}'_{Ai} (\mathbf{I}_A - \mathbf{Z}_A \hat{\mathbf{B}}).\end{aligned}\quad (15)$$

Following notation in section 2.3.2 and based on the adjusted estimators,  $\hat{\mathbf{B}}$  and  $\hat{m}_i$ , the semiparametric model-assisted estimator (SEMI) of the finite population distribution function can be written as

$$\hat{F}_{SEMI}(t) = \frac{1}{N} \sum_{i \in U} \hat{g}(x_i, \mathbf{z}_i) + \frac{1}{N} \sum_{i \in A} \frac{I_{\{y_i \leq t\}} - \hat{g}(x_i, \mathbf{z}_i)}{\pi_i} \quad (16)$$

where  $\hat{g}(x_i, \mathbf{z}_i) = \hat{m}_i + \mathbf{z}_i \hat{\mathbf{B}}$ . The estimated variance of this estimator is

$$\widehat{\text{Var}}(\hat{F}_{SEMI}(t)) = \frac{1}{N^2} \sum_{i, j \in A} \frac{\pi_{ij} - \pi_i \pi_j}{\pi_{ij}} \frac{I_{\{y_i \leq t\}} - \hat{g}_i}{\pi_i} \frac{I_{\{y_j \leq t\}} - \hat{g}_j}{\pi_j}. \quad (17)$$

From the results in Breidt and Opsomer (working paper), the design properties of  $\hat{T}_{SEMI}$  hold for  $\hat{F}_{SEMI}$ . Thus,  $\hat{F}_{SEMI}$  is both asymptotically design unbiased and design consistent.

## 3 Simulations

### 3.1 CDF estimation

In order to evaluate the efficiency of finite population cdf estimators that incorporate auxiliary information compared to more conventional estimators, such as the Horvitz-Thompson estimator, we include a Monte Carlo comparison of seven different estimators:

- HT: Horvitz-Thompson (1)
- CD0: Chambers and Dunstan (4) with  $\beta_0 = 0$ ,  $v^{1/2}(x) = x^{1/2}$
- CD1: Chambers and Dunstan (4) with  $v^{1/2}(x) = 1$
- RKM0: Rao, Kovar, Mantel (5) with  $\beta_0 = 0$ ,  $v^{1/2}(x) = x^{1/2}$
- RKM1: Rao, Kovar, Mantel (5) with  $v^{1/2}(x) = 1$
- DORF: Dorfman (1992)
- LPR: Local Polynomial Regression,  $q = 1$  (12)

In particular, we are interested in the comparison between the model-assisted, local polynomial regression estimator,  $\hat{F}_{LPR}(t)$ , and the parametric estimators to evaluate whether or not the removal of the restrictions placed on superpopulation (2) by parametric methods results in efficiency gains. Thus, we want to determine whether or not modeling in a generic context will improve upon estimators that operate under a more specific context. We are also interested in comparing model-assisted nonparametric  $\hat{F}_{LPR}(t)$  with the model-based nonparametric  $\hat{F}_{DORF}(t)$ . Both of these nonparametric estimators are carried out with the Epanechnikov kernel function

$$K(x) = \frac{3}{4}(1 - x^2)I_{\{|x| \leq 1\}}$$

and evaluated at bandwidth,  $h$ . Two bandwidths are chosen, the first according to the ad hoc rule of one quarter of the range of the auxiliary variable,  $h = 0.25$  for this simulation. A smaller bandwidth,  $h = 0.1$ , is also included for comparison. In order to reduce computation time, we evaluate both nonparametric estimators at one hundred grid points and interpolate using a piecewise constant function.

To investigate the efficiency of the different estimators under a variety of true mean functions and variance terms, we generate seven different superpopulations of the form  $y_i = m(x_i) + v^{1/2}(x_i)\epsilon_i$  from  $x_i \sim \text{Unif}(0, 1)$  and  $\epsilon_i \sim N(0, \sigma^2)$ :

Ratio:	$0.5x + x^{1/2}\epsilon$
Linear:	$1 + 2(x - 0.5) + \epsilon$
Expo:	$e^{-8x} + \epsilon$
Bump:	$1 + 2(x - 0.5) + e^{-200(x-0.5)^2} + \epsilon$
Jump:	$(0.35 + 2(x - 0.5))I_{\{x \leq 0.65\}} + \epsilon$
Quad:	$1 + 2(x - 0.5)^2 + \epsilon$
Cycle:	$2 + \sin(2\pi x) + \epsilon$

Plots of these populations and their respective cdf's are included in Appendix I. The selection of these populations allows us to compare the efficiencies of the different estimators under an assortment of correct and incorrect specifications of mean function  $m(x_i)$  and heteroskedastic variance term  $v^{1/2}(x_i)$ . Since it does not depend on any model, the Horvitz-Thompson estimator does not misspecify the mean function or variance term for any of the pop-

ulations. The nonparametric local polynomial regression and Dorfman estimators only assume a smooth mean function, therefore they only misspecify the mean function for the jump population since it has a point of discontinuity at  $x = 0.65$ . On the other hand, parametric estimators CD0 and RKM0 misspecify the mean function in every case except for the ratio population and CD1 and RKM1 misspecify the mean function in every case except for the ratio and linear populations. The selected populations also allow for a variety of variance misspecifications among the estimators. These cases have been studied but will not be emphasized here.

We perform the simulation for a population of size  $N = 1000$  from which samples of size  $n = 100$  are chosen based on simple random sampling (SI). It should be noted that  $\pi_i = n/N$  under our SI design, thus  $\tilde{G}_i$  of (5) is equal to  $\hat{G}_i$  of (4). The simulation is run for 1000 replications over fixed populations. At each replication a new sample is chosen and all seven estimators of the finite population cdf  $F(t)$  are calculated for each population at  $t$ , where  $t$  is the median of each respective population ( $F(t) = 0.5$ ). This procedure is repeated over different combinations of levels of bandwidth,  $h$ , and error variance,  $\sigma^2$ .

In order to compare the efficiencies of the the different estimators with the nonparametric local polynomial regression estimator, we calculate MSE ratios

$$\frac{MSE(\hat{F}_*(t))}{MSE(\hat{F}_{LPR}(t))}.$$

Thus an MSE ratio greater than 1 for a certain population implies that the local polynomial regression estimator is more efficient than the estimator with which it is being compared, whereas the opposite is true for MSE ratios less than 1. MSE ratios close to 1 indicate that the local polynomial regression estimator results in competitive estimates of  $F(t)$  in comparison to the other estimator.

Table 2 displays the MSE ratios for all cdf estimators to the local polynomial regression cdf estimator at the median of each population and over the different combinations of bandwidth,  $h$ , and error variance,  $\sigma^2$ . To summarize Table 2 we can look at the range and quartiles of the MSE ratios for each estimator with LPR. In particular, we can look at a summary of MSE ratios for cdf estimation calculated at the medians of all seven populations and across all combinations of  $h$  and  $\sigma$ :

Table 1: Summary of MSE ratios

	Min	Q1	Med	Q3	Max
<b>HT</b>	0.97	1.26	2.08	2.79	10.25
<b>CD0*</b>	0.48	0.76	1.63	6.58	19.47
<b>CD1*</b>	0.79	0.97	2.12	3.13	17.51
<b>RKM0*</b>	1.07	1.22	1.47	2.62	16.55
<b>RKM1*</b>	0.94	1.06	1.33	1.90	3.37
<b>DORF</b>	1.08	1.22	1.41	1.56	3.29

For Table 1, the summary statistics included for CD\* and RKM\* are calculated using MSE ratios only for those populations in which the CD and RKM estimators misspecify the mean function, all but the “ratio” and “linear” populations. We can see that the local polynomial regression estimator is essentially more efficient for all populations and across all combinations of  $h$  and  $\sigma$  than HT, which does not incorporate any auxiliary data into its estimation. In most cases it is much more efficient. In general, this indicates an increase in efficiency from the incorporation of auxiliary data into the estimation of distribution functions. LPR is also more efficient than the other nonparametric estimator, DORF. This gain in efficiency is intuitive since being model-based, DORF will display more bias than a model-assisted approach, which includes a design-bias adjustment.

We can break down the results of the Monte Carlo comparison between the local polynomial regression estimator and the parametric estimators into three separate categories with varying degrees of model misspecification:

- CASE 1: CD & RKM misspecify the mean function
- CASE 2: CD & RKM correctly specify the mean function, misspecify the variance term
- CASE 3: CD & RKM correctly specify both the mean function and variance term

Case 1 includes every population except for the “ratio” and “linear” populations for the CD and RKM estimators and is summarized in Figure 1. Here it can be seen that LPR essentially results in estimates that are either more efficient or competitive with the efficiencies of the CD and RKM estimators under mean function misspecification with a few exceptions. We can anticipate these results since the parametric CD and RKM estimators are based on a linear mean function for all populations, no matter how inappropriate, whereas LPR works under a more generic context. This misspecification re-

sults in loss of efficiency for the parametric estimators in all cases with one exception. The CD0 estimator, although misspecified for the “Quad” population, is more efficient than the local polynomial regression estimator. Recall that CD0 bases its estimates on a mean function that is a line fit through the origin via weighted least squares. If this were the true mean function, its cdf evaluated at the median of the “Quad” population will result in a value near 0.5. This coincidence is very favorable for CD0 estimation at the median of the “Quad” population, but CD0 does not perform well away from the median.

Case 2 includes CD0 and RKM0 estimation for the “linear” population and CD1 and RKM1 estimation for the “ratio” population. Case 3 includes CD0 and RKM0 estimation for the “ratio” population and CD1 and RKM1 estimation for the “linear” population. In order to explore these cases, we can look at their MSE ratios for cdf estimation calculated at the medians of these populations:

<b>Estimator</b>	<b>Case</b>	<b>Population</b>	$\sigma = 0.1$	$\sigma = 0.4$
CD0	2	Linear	0.55, 0.67	2.74, 2.86
CD1	2	Ratio	0.27, 0.27	1.85, 1.94
RKM0	2	Linear	0.69, 0.85	0.93, 0.97
RKM1	2	Ratio	0.94, 0.96	0.92, 0.97
CD0	3	Ratio	0.39, 0.39	0.67, 0.71
CD1	3	Linear	0.19, 0.23	0.54, 0.56
RKM0	3	Ratio	0.92, 0.94	0.91, 0.95
RKM1	3	Linear	0.69, 0.84	0.93, 0.97

We can see that for Case 2, when the CD estimators correctly specify the mean function but misspecify the variance term, LPR is more efficient than CD0 and CD1 for high noise,  $\sigma = 0.4$ , but less efficient for low noise,  $\sigma = 0.1$ . Since CD operates under the specific context of a linear mean function and known variance term, it is highly dependent on these conditions being met. Therefore, when the CD estimators misspecify the variance term, they will become increasingly inefficient as the noise level,  $\sigma$ , increases and more emphasis is placed on the variance misspecification. On the other hand, LPR is competitive with the RKM estimators with MSE ratios near one for both noise levels for Case 2. Since they are model-assisted as opposed to model-based, the RKM estimators work under a slightly less restricted context in

comparison with the CD estimators and are therefore not as influenced by the variance misspecification.

When the CD estimators correctly specify both the mean function and variance term, Case 3, LPR is less efficient across both noise levels. These results can be anticipated since CD, being parametric as well as model-based, works under the most specific or restricted context in comparison to the other estimators. Therefore, when the conditions of a linear mean function and correctly specified variance term are met, the CD estimators outperform more generic estimators that do place such emphasis on these assumptions being correct. However, when the CD estimators are applied to populations which do not fit into the context of a linear mean function or correctly specified variance, LPR gains in efficiency since it does not assume any model to be correct or defensible. Conversely, with MSE efficiency ratios slightly less than one, the LPR estimates are competitive with the RKM estimators for Case 3. Again, since the RKM estimators are model-assisted they work under a slightly more general context than the CD estimators. Thus, even when they correctly specify both the mean function and variance term, they are not rewarded with a much larger efficiency as the CD estimators are.

So far we have not looked at the effect of bandwidth on these MSE ratios in Table 2. Studying this table, it is unclear how our choice of bandwidth affects the MSE ratios since there is no obvious pattern. It needs to be emphasized that whereas the parametric estimators are based on modeling  $y_i$  vs.  $x_i$ , the nonparametric estimators are based on smoothing indicators  $I_{\{y_i \leq t\}}$  versus  $x_i$ . Recall that  $E_m I_{\{y_i \leq t\}} = P\{\epsilon_i \leq r_i(t)\}$ . Thus, for this simulation we can plot  $E_m I_{\{y_i \leq t\}}$  vs.  $x_i$  where  $E_m I_{\{y_i \leq t\}} = \Phi\{r_i(t)\}$  since  $\epsilon_i \sim N(0, \sigma^2)$ . Since  $\Phi\{r_i(t)\} = \Phi\{(t - m(x_i))v^{-1/2}(x_i)\}$ , we can see that the shape of  $\Phi(\cdot)$  depends on and varies with the different populations and selection of error variance,  $\sigma$ , and  $t$ . Therefore, in order to explore the effect of bandwidth selection on the MSE ratios, we must look at the shape of  $\Phi(\cdot)$  for each combination of population, error variance, and  $t$  for which we are estimating the finite population cdf.

To illustrate this idea we can look at how the different estimators approach the estimation of the cdf for the “jump” population of size 200. This population is illustrated in Figure 1 along with its cdf. The plot of Jump vs.  $x$  in Figure 1 illustrates the nonlinearity of the mean function for the “jump” population. Based on these plots however, we cannot determine a bandwidth that will maximize the efficiency of LPR.

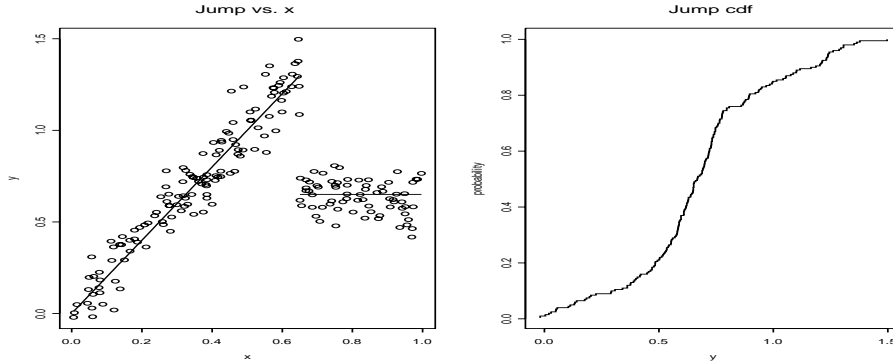


Figure 1: Jump vs.  $x$  with model mean (left plot) and cdf for the “jump” population (right plot) where  $\sigma = 0.1$

From Table 2, we have the following MSE ratios for the “jump” population for cdf estimation at its median:

#### MSE ratios for the “jump” population

$h$	$\sigma$	HT	CD0	CD1	RKM0	RKM1	DORF
0.10	0.1	1.99	0.77	2.38	1.51	1.90	1.39
0.10	0.4	1.20	1.19	0.90	1.07	1.11	1.13
0.25	0.1	1.87	0.72	2.24	1.42	1.79	1.49
0.25	0.4	1.26	1.26	0.95	1.13	1.18	1.24

From this table we can see that for low noise,  $\sigma = 0.1$ , LPR calculated using the smaller bandwidth of  $h = 0.10$  produces larger MSE ratios, and thus is more efficient, in comparison to using the larger bandwidth,  $h = 0.25$ . The opposite is true for high noise,  $\sigma = 0.4$ .

These results are not obvious by looking at Figure 1. In order to explain the effect that bandwidth has on LPR for this example we must look at what it is smoothing. Hence, we need to look at the plot of  $E_m I_{\{y_i \leq t\}}$  vs.  $x_i$ , Figure 2, where  $t$  is the median of the “jump” population. Figure 2 is constructed based on the fact that  $E_m I_{y_i \leq t} = \Phi\{r_i(t)\}$  and that for the “jump” population,  $r_i(t) = t - \{(0.35 + 2(x - 0.5))I_{\{x \leq 0.65\}}\}$ .

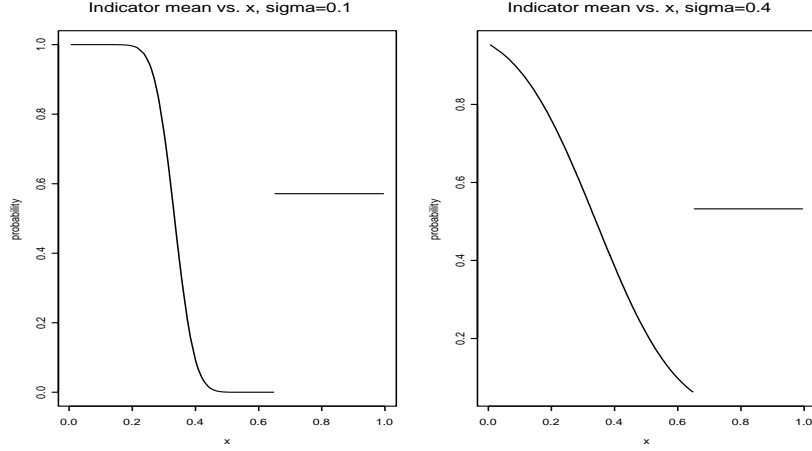


Figure 2:  $\Phi\{r_i(t)\}$  vs.  $x_i$  for the “jump” population where  $t$  is the median of the “jump” population and  $\sigma = 0.1$  (left plot),  $\sigma = 0.4$  (right plot)

Recall that for  $\sigma = 0.1$ , LPR calculated using the smaller bandwidth is more efficient than when using the larger bandwidth. When looking at the left hand plot where  $\Phi(\cdot)$  is calculated using  $\sigma = 0.1$  and focusing on the curve left of the discontinuity, we can see that the larger bandwidth will not capture the nonlinearity of this relationship, thus causing an increase in the MSE for LPR. On the other hand, the right hand plot where  $\Phi(\cdot)$  is calculated using  $\sigma = 0.4$  illustrates a more linear relationship in the curve left of the discontinuity in comparison to when we use  $\sigma = 0.1$ . Therefore, LPR calculated using a larger bandwidth will be more efficient than with a smaller bandwidth, which is supported by our table of MSE ratios.

We also include a table (Table 3) of percent relative biases

$$\left( \frac{\overline{\hat{F}(t)} - \alpha}{\alpha} \right) 100\% \quad (18)$$

for all estimators of each population and also across all combinations of bandwidth,  $h$ , and error variance,  $\sigma^2$ . For the case in which we are evaluating  $\hat{F}(t)$  at the median of each population, the true value of  $F(t)$ ,  $\alpha$ , is equal to 0.5 and  $\overline{\hat{F}(t)}$  is the average of  $\hat{F}(t)$  over the number of replications. We can see from this table that the model-assisted LPR and RKM estimators essentially have relative biases less than or near one percent for all populations whereas

the model-based CD and DORF estimators have up to twenty percent relative bias. Again, this is intuitive since the model-assisted estimators include a design-bias adjustment term.

This Monte Carlo comparison is repeated for the estimators evaluated at the first quartiles of the different populations. MSE ratios evaluated at the first quartile can be seen in Table 4 and Table 5 displays the percent relative biases of each estimator across the different populations and specifications of bandwidth,  $h$ , and error variance,  $\sigma^2$ .

Results of the cdf estimators evaluated at the first quartile are qualitatively similar to those found for the comparison between the estimators evaluated at the population medians with a few significant differences. In comparing the MSE ratios for the Monte Carlo comparison at the median and the Monte Carlo comparison at the first quartile, we can see the greatest disparity in the MSE ratios between the CD0 and LPR estimators. When evaluated at the first quartile, we see a great jump in efficiency of the LPR estimator over the CD0 estimator for the “jump”, “quad”, and “cycle” populations in comparison to the MSE ratios evaluated at the median. Recall that CD0 bases its estimation on a line fit through the origin. If a cdf for the population following this mean function is evaluated at the medians of these three populations, its value will be closer to 0.5 than its value when evaluated at the first quartiles of these three populations will be to 0.25. Just the opposite is true for the MSE ratio between the two estimators for the “bump” population.

### 3.2 Quantile Estimation

The same Monte Carlo comparison structure used in section 3.1 to compare the efficiency of several finite population cdf estimators can also be used to compare the efficiency of these same estimators when applied to quantile estimation. Estimators of the median for each population will be built by substituting each of the seven cdf estimators for  $\hat{F}(t)$  in  $\hat{\theta}(0.5) = \min\{t : \hat{F}(t) \geq 0.5\}$ . For example, the local polynomial regression estimator of the median is written as  $\hat{\theta}_{LPR}(0.5) = \min\{t : \hat{F}_{LPR}(t) \geq 0.5\}$ .

Table 6 summarizes the MSE ratios for all quantile estimators to the local polynomial regression quantile estimator of the median and Table 7 displays the percent relative biases of each estimator across the different populations and specifications of bandwidth,  $h$ , and error variance,  $\sigma^2$ . Both tables display results that are very similar to those of the Monte Carlo comparison

of cdf estimators at the median. The LPR approach to quantile estimation is essentially more efficient or competitive with the efficiencies of the CD and RKM approaches when the parametric methods misspecify the mean function. Also, for the situations in which the parametric methods do not misspecify the mean function, the LPR quantile estimator results in slightly less efficient or competitive estimates. Again, it is still the case that the LPR quantile estimator is essentially more efficient than both the Horvitz-Thompson and Dorfman quantile estimators across all populations and combinations of bandwidth,  $h$ , and error variance,  $\sigma^2$ . The relative biases for the different quantile estimators are similar to those of the cdf estimators in that model-assisted estimators incorporating LPR and RKM essentially have relative biases less than or near one percent for all populations whereas the model-based CD and DORF estimators have up to 64 percent relative bias. It makes sense that the results are so similar to the Monte Carlo comparison of the different cdf estimators since using parametric cdf estimators for  $\hat{F}(t)$  that are subject to mean function and variance misspecification bias will cause  $\hat{\theta}(\alpha)$  to be vulnerable to the same biases.

The largest disparity between the results of the Monte Carlo comparison of the cdf estimators and quantile estimators evaluated at the median is the MSE ratio results between CD0 and LPR at the “quad” population. These MSE ratios are much larger for the Monte Carlo comparison of the LPR and CD0 quantile estimators. The CD0 quantile estimator involves estimating the cdf in a neighborhood of values. It is possible that CD0 got lucky with its cdf estimation at the median, yet not over the whole neighborhood thereby increasing its MSE in the quantile estimation case.

There are very few discrepancies between the results of the Monte Carlo comparison of cdf estimators at the median and the Monte Carlo comparison of median estimators. Therefore, it is anticipated that the results of the Monte Carlo comparison of estimators of the first quartile would be similar to those of the Monte Carlo comparison of cdf estimators at the first quartile, hence a simulation of quartile estimators has not been performed.

## 4 Empirical Results

The National Surface Water Survey (NSWS) sponsored by the the Environmental Protection Agency (EPA) between the years of 1984 and 1986 estimated 4.2 percent of the lakes in the northeastern region of the United

States to be acidic (Stoddard et al. 2002). These acid-sensitive Northeastern lakes were among the concerns addressed by the Clean Air Act Amendment (CAAA) of 1990 which placed restrictions on industrial sulfur and nitrogen emissions in an effort to reduce the acidity of these waters.

A common measurement of acidity is acid neutralizing capacity (ANC), measured in  $\mu\text{eq/L}$ , which is defined as a water's ability to buffer acid. An ANC value less than zero indicates the presence of acidity in the water. Between the years of 1991 and 1996, the Environmental Monitoring and Assessment Program (EMAP) of the EPA studied the acid-sensitive regions of the Northeast. We can study this data in order to determine the effect that restrictions put in place by the CAAA had on the ANC of these waters. The survey is based on a population of 21384 lakes from which 557 water samples were taken. These samples were taken using a complex sampling design commonly employed by EMAP based on a hexagonal grid frame (Larsen et al. 1993).

First, a grid of hexagons with area  $635\text{km}^2$  is placed over the region to be studied. Centered within each of these hexagons is a smaller hexagon of area  $40\text{km}^2$ . The sampling frame consists of the lakes within these smaller hexagons. The lakes are sorted into size classes from which their inclusion probabilities are determined. The  $40\text{km}^2$  hexagons are spatially grouped into clusters and these clusters, hexagons within the clusters, and lakes within the hexagons are then randomly ordered. A list is made of these randomly ordered lakes and each lake in the list is assigned a length proportional to its inclusion probability. This list is then broken up into increments at which point lakes falling at these increments are sampled. The 557 samples taken by applying this design to the 21384 lakes in the survey come from 338 individual lakes, thus many lakes were sampled multiple times. In order to balance the sample, we average the multiple measurements of a given study variable over a lake in order to obtain one measurement per lake sampled.

We can determine the proportion of Northeastern lakes with ANC less than zero (the proportion of acidic lakes) by evaluating the finite population distribution function,  $F(t)$ , at  $t = 0$ . However, ANC was only measured for lakes in the sample; measurements are not available for the entire population. Therefore, we can only base conclusions on estimates of the distribution function. One option is to estimate the ANC cdf using the conventional Horvitz-Thompson estimator (1). However, many auxiliary variables are available for the population and the results from the simulations in section 3 suggest that the incorporation of this auxiliary information in our estimate

may improve upon the efficiency of the Horvitz-Thompson estimator. Let  $y_i$  represent the ANC value of the  $i$ th sampled lake and label the auxiliary variables as

$$\begin{aligned} x_i &= \text{longitude} \\ z_{ji} &= \text{eco-region } j\text{-indicator} \\ z_{11,i} &= \text{latitude} \\ z_{12,i} &= \text{elevation} \end{aligned}$$

where  $z_{ji}$  is the indicator variable assigning a 1 for lakes in eco-region  $j$  and 0 elsewhere. There are eleven different eco-regions included in the sample, thus dummy variables  $z_{ji}$  are constructed for  $j = 1, \dots, 10$ . The following table includes summary statistics for the non-categorical auxiliary variables:

**Summary Statistics for Auxiliary Variables**

	Minimum	Median	Mean	Maximum	Standard Deviation
Longitude	-79.0	-72.9	-72.8	-67.3	2.42
Latitude	39.5	43.5	43.5	47.2	1.43
Elevation (m)	4.0	313.0	320.2	807.0	196.8

The covariate space for this problem includes empty holes. Also, eco-region is a categorical variable. Given the choice of auxiliary variables to be incorporated into our estimate, we cannot apply the nonparametric estimator  $\hat{F}_{LPR}$  in (12) without potential loss of information. However, the semiparametric approach to cdf estimation,  $\hat{F}_{SEMI}$  given by (16), is able to incorporate each of the continuous and categorical auxiliary variables while still maintaining some of the flexibility in model specification of nonparametric methodology. In this example, the superpopulation model is nonparametric in longitude,  $x_i$ , and parametric in  $\mathbf{z}_i = (1, z_{1i}, z_{2i}, \dots, z_{12i})$ .

In order to determine the effect of incorporating auxiliary information into our estimate of the cdf evaluated at zero, we compare  $\hat{F}_{SEMI}$ , (16), to the conventional Horvitz-Thompson estimator (1). Because of the way the sampling design determines the structure of the inclusion probabilities,  $\hat{F}_{HT}$  must be modified as

$$\hat{F}_{HT*}(t) = \left( \sum_{j \in A} \frac{1}{\pi_j} \right)^{-1} \sum_{i \in A} \frac{I_{\{y_i \leq t\}}}{\pi_i}, \quad (19)$$

which is sometimes called the Hájek estimator. It can be seen that for a simple random sampling design,  $\pi_j = \frac{n}{N}$ , that this form of the Horvitz-Thompson estimator reduces to the conventional form given by (1).

In order to determine the efficiency of estimators  $\hat{F}_{SEMI}(t)$ , (16), and  $\hat{F}_{HT*}(t)$ , (19), we must compute the variance estimates of each. However, second order inclusion probabilities,  $\pi_{ij}$ , are not available, thus  $\widehat{\text{Var}}(\hat{F}_{SEMI}(t))$  given by (17) cannot be evaluated. In order to come up with appropriate variance estimates, we treat the complex sampling design as a stratified sample taken with replacement. Let  $H$  be the number of strata,  $n_h$  the number of observations within stratum  $h$ , and  $A_h$  the set of sampled elements that fall in stratum  $h$ . Define  $p_i = n_h^{-1}\pi_i$ . Using this notation and the assumption of a stratified sample with replacement, we can write the semiparametric estimator as

$$\hat{F}_{SEMI}(t) = \frac{1}{N} \sum_{i \in U} \hat{g}(x_i, \mathbf{z}_i) + \frac{1}{N} \sum_{h \in H} \frac{1}{n_h} \sum_{i \in A_h} \frac{I_{\{y_i \leq t\}} - \hat{g}(x_i, \mathbf{z}_i)}{p_i}$$

and the variance estimate as

$$\widehat{\text{Var}}(\hat{F}_{SEMI}(t)) = \frac{1}{N^2} \sum_{h \in H} \frac{1}{n_h} S_h^2(t)$$

where  $S_h^2(t)$  is the estimated within-stratum weighted residual variance for stratum  $h$ . Assuming the strata are sampled with replacement, Särndal, Swensson, and Wretman (1992, 421-422) suggest  $S_h^2(t)$  can be calculated as

$$S_h^2(t) = \frac{1}{n_h(n_h - 1)} \sum_{i \in A_h} \left( \frac{I_{\{y_i \leq t\}} - \hat{g}(x_i, \mathbf{z}_i)}{p_i} - \sum_{j \in A_h} \frac{I_{\{y_j \leq t\}} - \hat{g}(x_j, \mathbf{z}_j)}{\pi_j} \right)^2.$$

Similarly, we can represent  $\widehat{\text{Var}}(\hat{F}_{HT*}(t))$  as

$$\widehat{\text{Var}}(\hat{F}_{HT*}(t)) = \frac{1}{N^2} \sum_{h \in H} \frac{1}{n_h} V_h^2(t)$$

where

$$V_h^2(t) = \frac{1}{n_h(n_h - 1)} \sum_{i \in A_h} \left( \frac{I_{\{y_i \leq t\}} - \hat{F}_{HT*}(t)}{p_i} - \sum_{j \in A_h} \frac{I_{\{y_j \leq t\}} - \hat{F}_{HT*}(t)}{\pi_j} \right)^2.$$

We can now calculate estimates of the proportion of Northeastern lakes with ANC less than zero along with their respective variance estimates:

$$\begin{array}{lcl} \hat{F}_{SEMI}(0) & = & 0.044 \quad \widehat{\text{Var}}(\hat{F}_{SEMI}(0)) & = & 0.00031 \\ \hat{F}_{HT^*}(0) & = & 0.059 \quad \widehat{\text{Var}}(\hat{F}_{HT^*}(0)) & = & 0.00046. \end{array}$$

Confidence intervals for  $F(0)$ , the proportion of Northeastern lakes with ANC less than zero, based on both estimators can be calculated from this information as follows:

95% CI for  $F(0)$  based on  $\hat{F}_{SEMI}(0)$ :

$$(0.044 \pm 1.96\sqrt{0.00031}) = (0.0095, 0.0785)$$

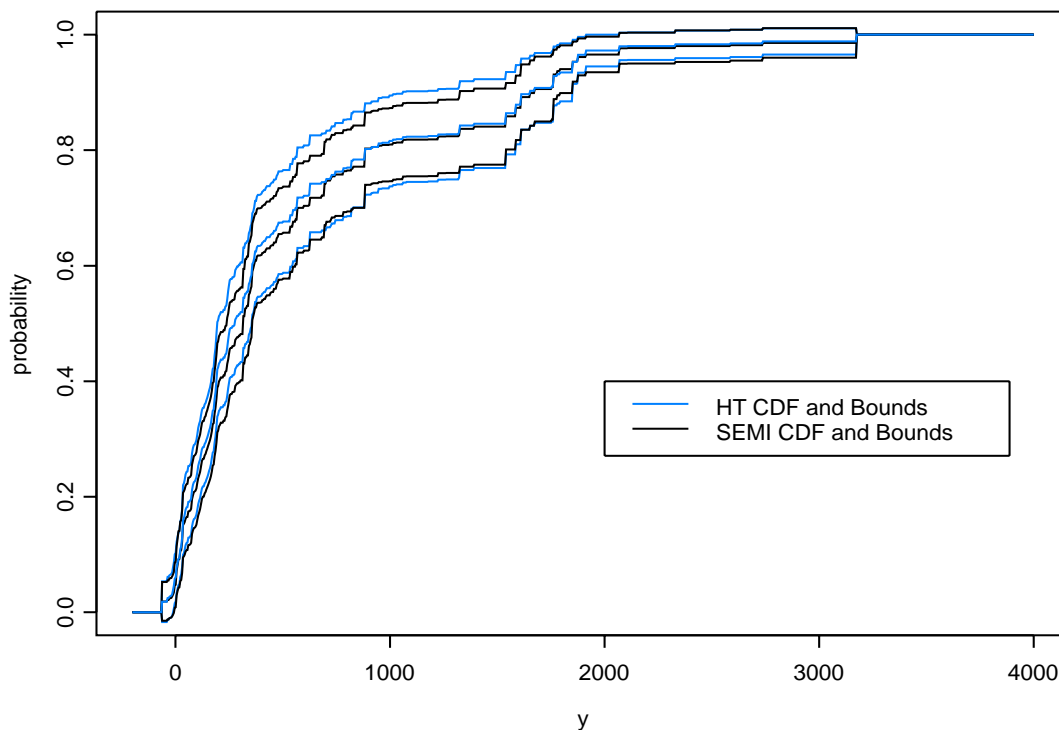
95% CI for  $F(0)$  based on  $\hat{F}_{HT^*}(0)$ :

$$(0.059 \pm 1.96\sqrt{0.00046}) = (0.0170, 0.1010)$$

The confidence interval constructed using the Hájek estimator is about 22 percent wider than that constructed using the semiparametric estimator. This suggests that  $\hat{F}_{HT^*}(t)$  produces estimates that are more conservative and less efficient than those given by  $\hat{F}_{SEMI}(t)$  at  $t = 0$ .

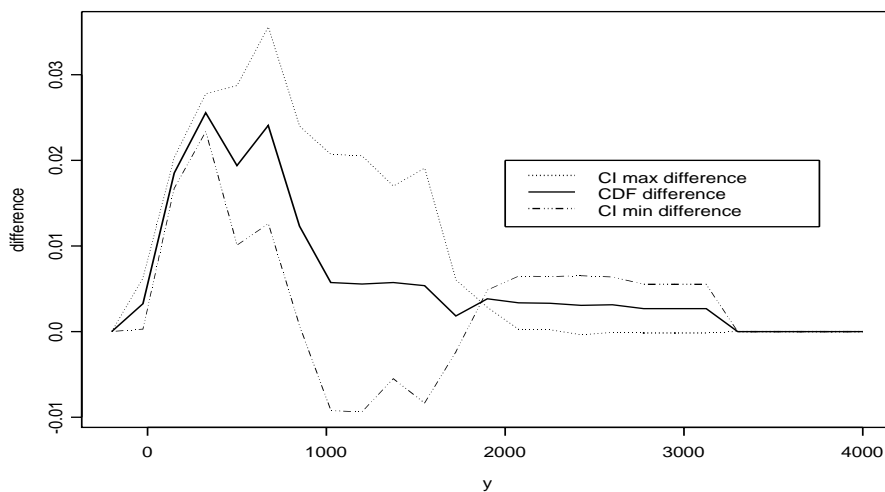
The following graph includes estimates of the ANC cdf produced by  $\hat{F}_{HT^*}(t)$  and  $\hat{F}_{SEMI}(t)$  evaluated on a grid of 1000 equally spaced values for  $t$ . Included are their respective confidence intervals which are also calculated at each grid point.

### ANC CDF Estimation



From this graph we can see that both the semiparametric and conventional Hájek approaches result in very similar cdf estimates which is to be expected since  $\hat{F}_{HT^*}(t)$  is design unbiased and  $\hat{F}_{SEMI}(t)$  is asymptotically design unbiased. However,  $\hat{F}_{SEMI}$  has narrower confidence intervals on average than  $\hat{F}_{HT^*}$ , 0.077 versus 0.084. In other words, confidence intervals based on  $\hat{F}_{HT^*}$  are about nine percent wider on average. This discrepancy is especially apparent at grid values away from the tails. The divergence between the confidence intervals based on the two different estimators are more apparent when looking at a plot of the differences between their upper and lower bounds as well as the difference between their estimates of  $F(t)$  for a grid of 1000 equally spaced values for  $t$ :

CI Comparison for ANC CDF Estimation



For Figure 4, the “differences” are defined as follows:

$$\begin{aligned} \text{CI max difference} &= \hat{F}_{HT^*}(t) \text{ upper bound} - \hat{F}_{SEMI}(t) \text{ upper bound} \\ \text{CI min difference} &= \hat{F}_{HT^*}(t) \text{ lower bound} - \hat{F}_{SEMI}(t) \text{ lower bound} \\ \text{CDF difference} &= \hat{F}_{HT^*}(t) - \hat{F}_{SEMI}(t) \end{aligned}$$

Having proved that the semiparametric estimator is more efficient than the Hájek estimator, we may question whether or not it would be just as efficient to model all the auxiliary variables parametrically. Figure 3 displays a plot of residuals from a parametric fit,  $I_{\{y_i \leq 0\}} - \mathbf{z}'_i \hat{\mathbf{B}}$ , versus longitude, the nonparametrically modeled auxiliary variable. This figure also includes superimposed nonparametric smooths,  $\hat{m}_i$ , of these residuals. The nonlinear behavior of these smooths indicates that parametrically modeling longitude would have been a misspecification of its true behavior.

Based on the above results we can evaluate the effect that emissions restrictions put in place by the CAAA of 1990 have had on levels of acidity in the lakes of the northeastern region of the United States. Recall that the NSWIS determined 4.2 percent of Northeastern Lakes to be acidic in 1986. Based on EMAP survey data taken between the years of 1991 and 1996,  $\hat{F}_{SEMI}$  estimated 4.4 percent of Northeastern lakes to be acidic and  $\hat{F}_{HT^*}$  estimated 5.9 percent to be acidic. These latter estimates show no evidence that the CAAA of 1990 has yet been successful in significantly reducing acid levels of waters in the Northeastern region of the United States.

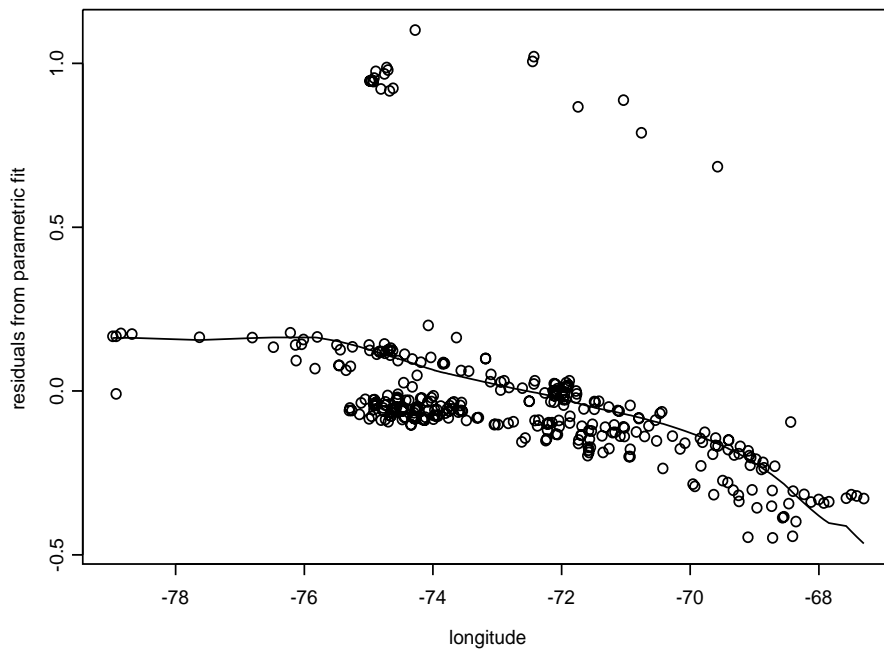


Figure 3: Residuals from a parametric fit versus longitude with superimposed nonparametric smooth based on a bandwidth equal to 5

However, there are many reasons why such a strict conclusion may not be reasonable. For instance, the ANC of Northeastern lakes may have a delayed response to the emission restrictions and take many more years to become recognizable or significant (Stoddard et al., 2002). It is also possible that other atmospheric conditions play a role in offsetting positive effects of the emissions restrictions. Also, it is possible that the emissions restrictions may have stopped or reversed a trend toward increasing acidity. In any case, it has been shown that incorporating auxiliary information such as latitude, longitude, elevation, and eco-region improve, on average, the efficiency of our estimates of the proportion of acidic Northeastern lakes in comparison to a more conventional estimator that does not take advantage of the additional variables.

The incorporation of additional auxiliary variables into semiparametric survey estimators is handled with ease. Also, we can write  $\hat{F}_{SEMI}$  as  $\hat{F}_{SEMI}(t) = \sum_{i \in A} w_{si}^* I_{\{y_i \leq t\}}$  where  $w_{si}^*$  are weights that are calculated independently of the study variable,  $y$ . Since these weights are calculated independently of  $y$ , they are easily applied to a variety of study variables. Along with ANC, the EMAP survey of Northeastern lakes measured multiple chemistry variables including sulfate ( $\text{SO}_4$ ), magnesium (Mg), and chloride (Cl). Not only can we estimate the cdf's of these variables semiparametrically, we can easily extend  $\hat{F}_{SEMI}$ , (16), to quantile estimators  $\hat{\theta}_{SEMI}(\alpha) = \min\{t : \hat{F}_{SEMI}(t) \geq \alpha\}$  of these additional chemistry variables. The following table displays semiparametric estimates of the first, second, and third quartiles of sulfate, magnesium, and chloride measured in (ueq/L):

### Quantile Estimates of Chemistry Variables

$\alpha$	Sulfate	Magnesium	Chloride
0.25	73.0	66.1	25.1
0.50	105.0	123.8	177.0
0.75	194.9	238.3	495.4

These estimates are easily obtained by modeling and incorporating the same auxiliary variables in exactly the same way as in the semiparametric cdf estimation for ANC.

## 5 Conclusion

It has been shown via Monte Carlo comparisons that finite population distribution function estimators which incorporate auxiliary information improve upon more conventional estimators, such as Horvitz-Thompson, that do not. Moreover, we have introduced a model-assisted nonparametric estimator (12) based on local polynomial regression which essentially improves upon the efficiency of parametric methods under misspecification of the relationship between the auxiliary and study variables. It is also more efficient than a model-based nonparametric estimator (6). These results extend to the incorporation of these estimators into quantile estimation.

In order to address survey situations in which multiple auxiliary variables are available, of which some may be categorical in nature, but in

which holes in the covariate space are possible, we have also introduced a semiparametric estimator which merges both nonparametric and parametric methods. As well as being an extension of the introduced local polynomial regression estimator, this estimator allows for the incorporation of a wider variety of auxiliary variables while still maintaining some of the flexibility of nonparametric model specification. Applying this estimator to a survey of Northeastern lakes in order to assess the proportion of acidic lakes resulted in estimates more efficient, on average, than those determined from the Horvitz-Thompson approach, which does not incorporate any auxiliary information.

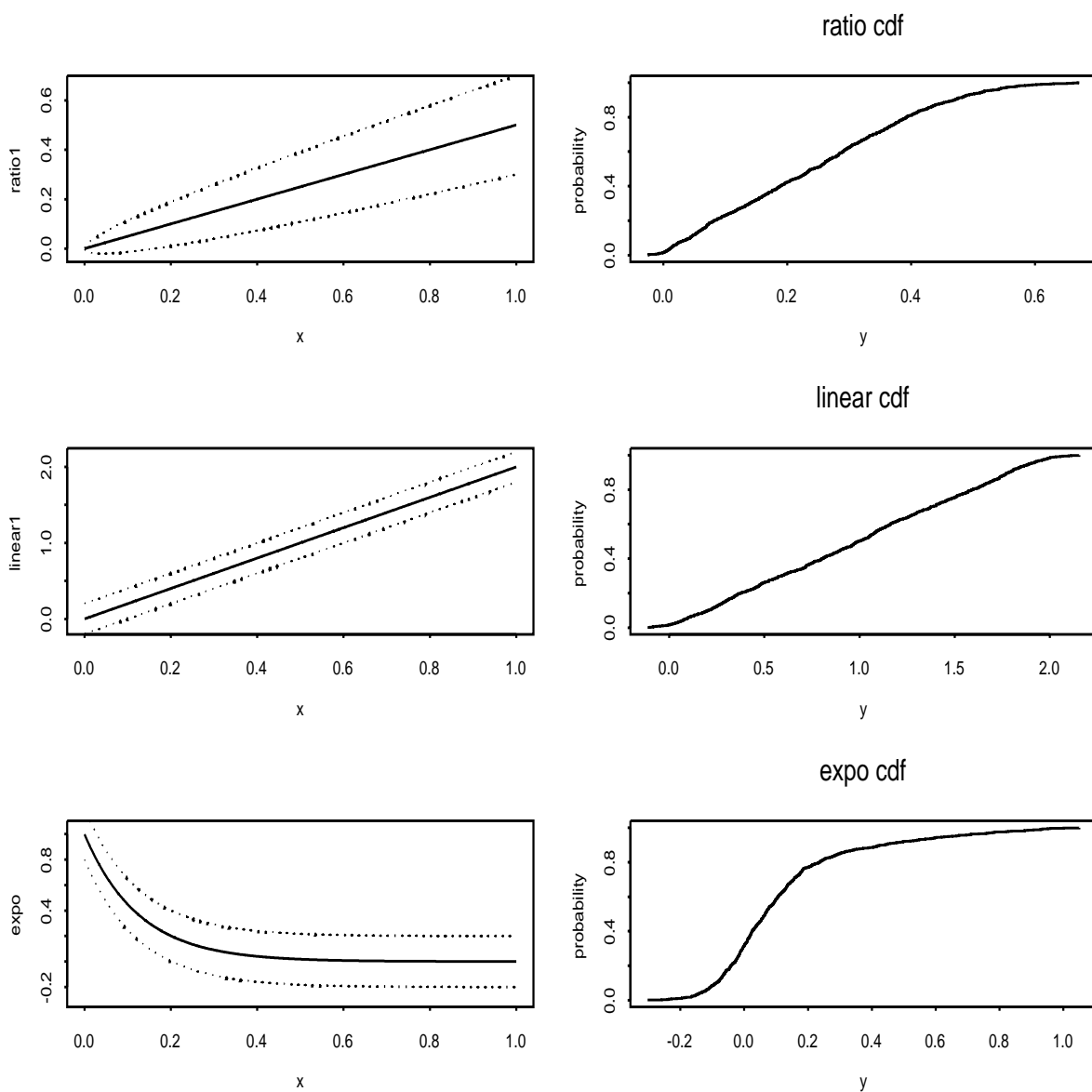
In conclusion, the results included in this paper have demonstrated that the incorporation of auxiliary information improves the efficiency of survey estimation. Also, nonparametric methods remove the parametric strait-jacket, and therefore work under a generic context in which no model is assumed to be correct or defensible. In comparison to parametric estimators and model-based nonparametric estimators which operate under stricter assumptions, this estimator is essentially more efficient under misspecification of the relationship between the auxiliary and study variables. Therefore, for sampling situations in which we do not have the resources to investigate the behavior of every variable, it is advantageous to apply the model-assisted local polynomial regression estimator (12).

## References

- Breidt, F.J. and J.D. Opsomer (2000). Local polynomial regression estimators in survey sampling. *Ann. Statist.*, **28**, 1026–1053.
- Breidt, F.J. and J.D. Opsomer (2002). Design Properties of Semiparametric Model-assisted Estimators. Working paper. Iowa State University.
- Chambers, R.L., A.H. Dorfman, and P. Hall (1992). Properties of estimators of the finite population distribution function. *Biometrika* **79**, 577–82.
- Chambers, R.L. and R. Dunstan (1986). Estimating distribution functions from survey data. *Biometrika* **73**, 597–604.
- Dorfman, A.H. (1992). Nonparametric regression for estimating totals in finite populations. *Proceedings of the Section on Survey Research Methods*, American Statistical Association, 622–625.
- Larsen, D.P., K.W. Thornton, N.S. Urquhart, and S.G. Paulsen (1993). Overview of Survey Design and Lake Selection. *EMAP - Surface Waters 1991 Pilot Report*, edited by Larsen, D.P. and Christie, S.J. EPA/620/R-93/003.
- Rao, J.N.K., J.G. Kovar, and H.J. Mantel (1990). On estimating distribution functions and quantiles from survey data using auxiliary information. *Biometrika* **77**, 365–75.
- Särndal, C.-E., B. Swensson, and J. Wretman (1992). *Model Assisted Survey Sampling*. New York: Springer-Verlag.
- Stoddard, J.L., J.S. Kahl, F.A. Deviney, D.R. DeWalle, C.T. Driscoll, A.T. Herlihy, J.H. Kellogg, P.S. Murdoch, J.R. Webb, and K.E. Webster (2002). Response of surface water chemistry to the Clean Air Act Amendments of 1990.

## Appendix I: graphs and tables

Left column: Superpopulation model means and standard deviation bounds. Right column: cdf's for finite populations of size 1000 simulated from corresponding left-column superpopulations.



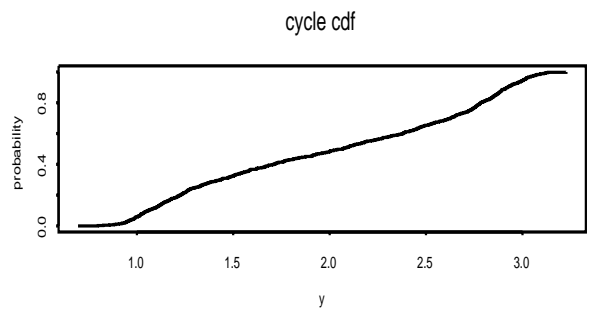
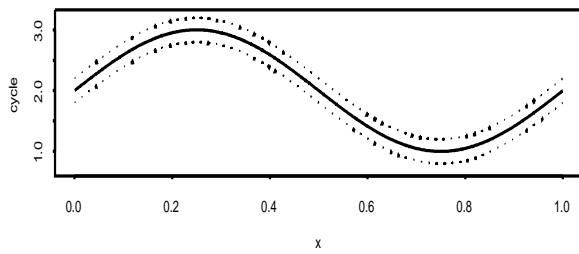
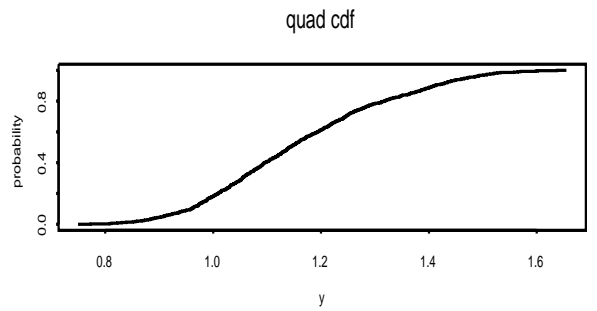
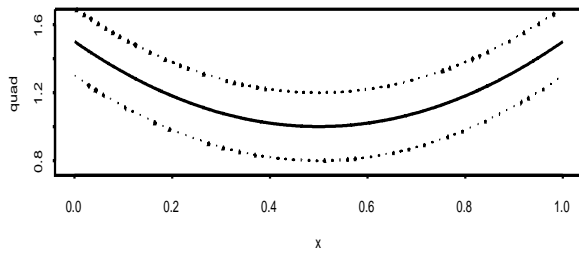
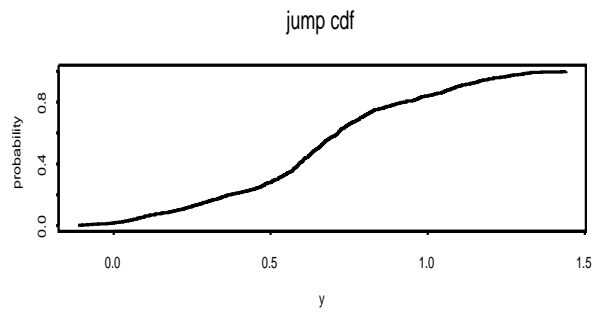
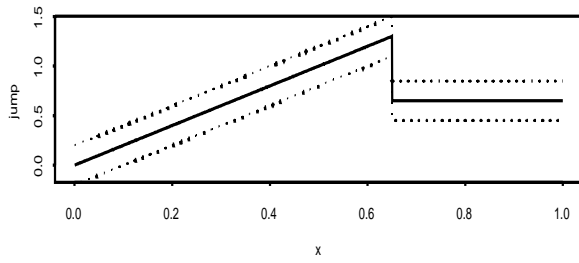
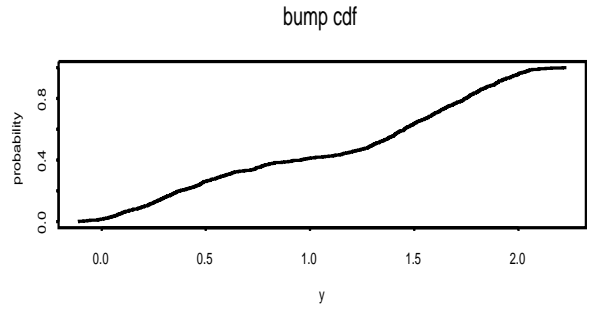
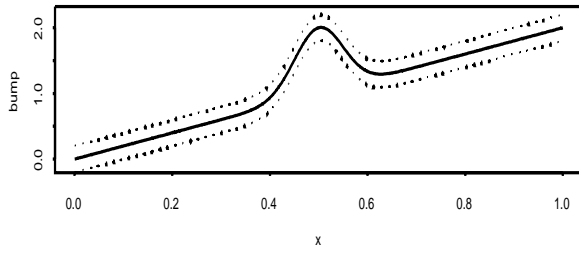


Table 2: MSE ratios for CDF estimation calculated at the median of HT, CD0, CD1, RKM0, RKM1, and DORF to local polynomial regression estimator (LPR).

Population	h	$\sigma$	HT	CD0	CD1	RKM0	RKM1	DORF
Ratio	0.10	0.1	2.72	0.39	0.27	0.92	0.94	1.51
	0.10	0.4	1.18	0.67	1.85	0.91	0.92	1.15
	0.25	0.1	2.77	0.39	0.27	0.94	0.96	1.56
	0.25	0.4	1.24	0.71	1.94	0.95	0.97	1.22
Linear	0.10	0.1	7.02	0.67	0.23	0.85	0.84	2.48
	0.10	0.4	2.07	2.74	0.54	0.93	0.93	1.32
	0.25	0.1	5.73	0.55	0.19	0.69	0.69	2.32
	0.25	0.4	2.16	2.86	0.56	0.97	0.97	1.40
Expo	0.10	0.1	1.35	2.00	3.59	1.80	1.01	1.11
	0.10	0.4	1.00	0.96	0.79	1.14	0.94	1.08
	0.25	0.1	1.42	2.10	3.77	1.90	1.06	1.23
	0.25	0.4	1.06	1.02	0.83	1.20	0.99	1.17
Bump	0.10	0.1	4.85	19.47	17.51	2.15	2.37	1.96
	0.10	0.4	2.58	7.24	2.98	1.23	1.29	1.41
	0.25	0.1	3.05	12.26	11.02	1.35	1.49	1.49
	0.25	0.4	2.26	6.36	2.62	1.08	1.14	1.39
Jump	0.10	0.1	1.99	0.77	2.38	1.51	1.90	1.39
	0.10	0.4	1.20	1.19	0.90	1.07	1.11	1.13
	0.25	0.1	1.87	0.72	2.24	1.42	1.79	1.49
	0.25	0.4	1.26	1.26	0.95	1.13	1.18	1.24
Quad	0.10	0.1	2.09	0.70	2.00	2.67	2.11	1.41
	0.10	0.4	0.97	0.48	0.91	1.26	0.98	1.09
	0.25	0.1	2.03	0.68	1.95	2.60	2.06	1.89
	0.25	0.4	1.04	0.51	0.97	1.34	1.05	1.20
Cycle	0.10	0.1	10.25	13.67	4.66	16.55	3.37	3.29
	0.10	0.4	2.79	3.11	1.18	4.29	1.37	1.47
	0.25	0.1	5.79	7.73	2.63	9.35	1.90	2.42
	0.25	0.4	2.79	3.11	1.19	4.29	1.38	1.57

Table 3: Percent relative biases for cdf estimators HT, CD0, CD1, RKM0, RKM1, LPR, and DORF calculated at the median

Population	h	$\sigma$	HT	CD0	CD1	RKM0	RKM1	LPR	DORF
Ratio	0.10	0.1	0.29	2.13	-1.45	0.13	0.11	-0.96	-0.90
	0.10	0.4	0.42	2.12	-9.53	0.33	0.22	-0.55	-0.64
	0.25	0.1	0.29	2.13	-1.45	0.13	0.11	-0.89	-0.64
	0.25	0.4	0.42	2.12	-9.53	0.33	0.22	-0.48	-1.01
Linear	0.10	0.1	0.16	2.61	1.48	0.04	0.03	-1.04	-0.96
	0.10	0.4	0.35	10.01	3.27	0.26	0.22	-0.85	-0.80
	0.25	0.1	0.16	2.61	1.48	0.04	0.03	-1.02	-0.85
	0.25	0.4	0.35	10.01	3.27	0.26	0.22	-0.83	-0.62
Expo	0.10	0.1	-0.25	8.01	-14.56	-0.35	-0.22	0.49	0.56
	0.10	0.4	-0.05	4.05	-2.98	-0.15	0.07	0.40	0.72
	0.25	0.1	-0.25	8.01	-14.56	-0.35	-0.22	0.56	1.07
	0.25	0.4	-0.05	4.05	-2.98	-0.15	0.07	0.51	1.54
Bump	0.10	0.1	0.45	18.86	17.84	0.53	0.52	-0.79	-0.59
	0.10	0.4	0.28	15.02	9.11	0.24	0.17	-1.04	-0.95
	0.25	0.1	0.45	18.86	17.84	0.53	0.52	-0.80	-0.55
	0.25	0.4	0.28	15.02	9.11	0.24	0.17	-0.97	-0.49
Jump	0.10	0.1	0.18	2.80	6.88	0.01	-0.18	-0.66	-0.84
	0.10	0.4	0.29	6.87	1.56	0.19	-0.04	-0.42	-0.60
	0.25	0.1	0.18	2.80	6.88	0.01	-0.18	-0.45	-0.84
	0.25	0.4	0.29	6.87	1.56	0.19	-0.04	-0.22	-1.06
Quad	0.10	0.1	0.27	0.21	-0.10	0.12	0.74	0.36	0.35
	0.10	0.4	0.40	2.49	-0.35	0.29	0.58	0.32	0.86
	0.25	0.1	0.27	0.21	-0.10	0.12	0.74	0.08	2.91
	0.25	0.4	0.40	2.49	-0.35	0.29	0.58	0.41	1.90
Cycle	0.10	0.1	-0.01	7.34	4.49	-0.07	0.21	0.65	0.53
	0.10	0.4	-0.23	6.15	2.95	-0.30	-0.07	-0.01	0.24
	0.25	0.1	-0.01	7.34	4.49	-0.07	0.21	0.91	0.44
	0.25	0.4	-0.23	6.15	2.95	-0.30	-0.07	0.24	0.15

Table 4: MSE ratios for CDF estimation calculated at the first quartile of HT, CD0, CD1, RKM0, RKM1, and DORF to local polynomial regression estimator (LPR).

<b>Population</b>	<b>h</b>	<b><math>\sigma</math></b>	<b>HT</b>	<b>CD0</b>	<b>CD1</b>	<b>RKM0</b>	<b>RKM1</b>	<b>DORF</b>
Ratio	0.10	0.1	3.23	0.14	0.75	0.92	0.96	1.39
	0.10	0.4	1.00	0.56	0.71	0.89	0.90	1.00
	0.25	0.1	3.22	0.14	0.74	0.92	0.96	1.76
	0.25	0.4	1.09	0.60	0.77	0.97	0.98	1.07
Linear	0.10	0.1	5.07	0.08	0.19	0.93	0.89	1.69
	0.10	0.4	1.91	1.15	0.58	0.94	0.93	1.15
	0.25	0.1	4.48	0.07	0.17	0.82	0.79	2.14
	0.25	0.4	1.99	1.21	0.60	0.98	0.98	1.31
Expo	0.10	0.1	1.01	2.12	0.83	1.17	0.98	0.98
	0.10	0.4	0.91	0.93	0.64	0.89	0.89	0.98
	0.25	0.1	1.10	2.31	0.90	1.27	1.07	1.05
	0.25	0.4	0.98	0.99	0.69	0.95	0.95	1.03
Bump	0.10	0.1	5.25	1.05	6.56	0.92	1.01	1.71
	0.10	0.4	2.12	1.10	1.33	0.95	0.98	1.21
	0.25	0.1	4.59	0.92	5.73	0.81	0.88	2.16
	0.25	0.4	2.21	1.15	1.39	0.99	1.03	1.39
Jump	0.10	0.1	3.89	37.43	10.82	1.30	2.20	1.44
	0.10	0.4	1.20	3.92	0.74	0.96	1.03	0.99
	0.25	0.1	3.65	35.19	10.17	1.22	2.07	1.92
	0.25	0.4	1.31	4.26	0.81	1.04	1.12	1.11
Quad	0.10	0.1	1.31	28.12	1.20	1.92	1.35	1.05
	0.10	0.4	0.98	14.59	0.91	1.39	0.99	1.01
	0.25	0.1	1.32	28.36	1.21	1.93	1.36	1.18
	0.25	0.4	1.04	15.37	0.95	1.46	1.04	1.12
Cycle	0.10	0.1	5.87	84.39	18.36	9.21	4.65	2.24
	0.10	0.4	1.82	28.09	1.56	2.94	1.37	1.22
	0.25	0.1	3.09	44.49	9.68	4.85	2.45	2.63
	0.25	0.4	1.74	26.83	1.49	2.81	1.30	1.36

Table 5: Percent relative biases for cdf estimators HT, CD0, CD1, RKM0, RKM1, LPR, and DORF calculated at the first quartile

Population	h	$\sigma$	HT	CD0	CD1	RKM0	RKM1	LPR	DORF
Ratio	0.10	0.1	1.18	-0.66	-6.28	0.79	0.74	-1.15	-1.14
	0.10	0.4	0.88	-1.27	-9.39	0.72	0.62	-0.65	-1.02
	0.25	0.1	1.18	-0.66	-6.28	0.79	0.74	-1.11	-3.21
	0.25	0.4	0.88	-1.27	-9.39	0.72	0.62	-0.32	-1.30
Linear	0.10	0.1	0.69	0.17	-1.87	0.32	0.25	-1.68	-1.45
	0.10	0.4	1.09	10.54	-4.29	0.87	0.69	-1.22	-1.40
	0.25	0.1	0.69	0.17	-1.87	0.32	0.25	-1.59	-3.52
	0.25	0.4	1.09	10.54	-4.29	0.87	0.69	-1.09	-2.46
Expo	0.10	0.1	-0.39	19.81	11.34	-0.87	0.06	0.33	0.32
	0.10	0.4	-0.47	-7.82	0.84	-0.60	-0.20	-0.08	0.11
	0.25	0.1	-0.39	19.81	11.34	-0.87	0.06	0.61	0.62
	0.25	0.4	-0.47	-7.82	0.84	-0.60	-0.20	0.35	0.55
Bump	0.10	0.1	0.70	-6.82	-17.36	0.28	-0.01	-1.69	-1.46
	0.10	0.4	1.11	9.56	-10.15	0.87	0.52	-1.21	-1.43
	0.25	0.1	0.70	-6.82	-17.36	0.28	-0.01	-1.59	-3.49
	0.25	0.4	1.11	9.56	-10.15	0.87	0.52	-1.09	-2.42
Jump	0.10	0.1	0.82	49.25	24.20	0.98	0.50	-1.51	-1.29
	0.10	0.4	1.02	27.92	-2.43	1.00	0.46	-0.61	-1.01
	0.25	0.1	0.82	49.25	24.20	0.98	0.50	-1.38	-3.63
	0.25	0.4	1.02	27.92	-2.43	1.00	0.46	-0.22	-2.45
Quad	0.10	0.1	-0.01	75.05	0.07	0.51	1.01	0.20	0.08
	0.10	0.4	-0.03	61.25	0.54	0.26	0.32	-0.09	0.45
	0.25	0.1	-0.01	75.05	0.07	0.51	1.01	-0.06	1.73
	0.25	0.4	-0.03	61.25	0.54	0.26	0.32	-0.08	2.13
Cycle	0.10	0.1	0.23	58.37	-26.63	0.46	0.34	-0.17	0.53
	0.10	0.4	-0.28	62.72	-11.36	0.05	-0.06	-0.74	1.05
	0.25	0.1	0.23	58.37	-26.63	0.46	0.34	-0.85	6.10
	0.25	0.4	-0.28	62.72	-11.36	0.05	-0.06	-0.17	3.43

Table 6: MSE ratios for estimation of the median of estimators HT, CD0, CD1, RKM0, RKM1, and DORF to local polynomial regression estimator (LPR).

Population	h	$\sigma$	HT	CD0	CD1	RKM0	RKM1	DORF
Ratio	0.10	0.1	3.14	0.37	0.31	0.97	1.00	1.25
	0.10	0.4	1.15	0.58	1.75	0.89	0.91	0.95
	0.25	0.1	3.09	0.37	0.31	0.96	0.98	1.26
	0.25	0.4	1.26	0.64	1.90	0.97	0.99	1.03
Linear	0.10	0.1	8.14	0.88	0.32	0.85	0.84	1.78
	0.10	0.4	2.49	3.64	0.65	1.05	1.04	1.13
	0.25	0.1	6.55	0.71	0.26	0.69	0.68	1.85
	0.25	0.4	2.57	3.77	0.61	1.08	1.08	1.18
Expo	0.10	0.1	1.32	2.34	7.68	1.80	0.99	1.01
	0.10	0.4	1.00	0.88	0.92	1.13	0.95	0.96
	0.25	0.1	1.40	2.49	8.16	1.92	1.05	1.10
	0.25	0.4	1.06	0.94	0.97	1.21	1.01	1.02
Bump	0.10	0.1	11.70	35.91	26.10	2.53	2.84	1.72
	0.10	0.4	2.66	6.97	2.23	1.26	1.31	1.18
	0.25	0.1	7.01	21.49	15.63	1.51	1.70	1.42
	0.25	0.4	2.37	6.22	1.99	1.12	1.17	1.16
Jump	0.10	0.1	2.01	3.46	3.98	1.46	1.81	1.10
	0.10	0.4	1.18	1.74	0.82	1.07	1.10	0.99
	0.25	0.1	1.88	3.24	3.74	1.37	1.70	1.18
	0.25	0.4	1.26	1.85	0.88	1.14	1.18	1.07
Quad	0.10	0.1	1.98	28.08	1.91	2.66	1.99	1.17
	0.10	0.4	0.96	2.55	0.87	1.25	0.96	0.97
	0.25	0.1	1.98	28.04	1.90	2.65	1.99	1.44
	0.25	0.4	1.02	2.71	0.92	1.33	1.02	1.02
Cycle	0.10	0.1	7.07	29.93	1.04	10.65	2.46	2.06
	0.10	0.4	3.89	18.47	0.86	6.10	1.67	1.30
	0.25	0.1	4.32	18.29	0.63	6.51	1.51	1.83
	0.25	0.4	3.52	16.68	0.78	5.51	1.51	1.55

Table 7: Percent relative biases for quantile estimators HT, CD0, CD1, RKM0, RKM1, LPR, and DORF calculated at the median

Population	h	$\sigma$	HT	CD0	CD1	RKM0	RKM1	LPR	DORF
Ratio	0.10	0.1	-0.89	-2.06	1.62	-0.64	-0.59	0.47	0.40
	0.10	0.4	-1.16	-3.18	18.15	-1.25	-0.91	0.61	0.92
	0.25	0.1	-0.89	-2.06	1.62	-0.64	-0.59	0.50	0.17
	0.25	0.4	-1.16	-3.18	18.15	-1.25	-0.91	0.32	1.53
Linear	0.10	0.1	-0.61	-2.60	-1.50	-0.27	-0.27	0.72	0.57
	0.10	0.4	-1.67	-10.62	-3.21	-1.68	-1.67	-0.65	-0.66
	0.25	0.1	-0.61	-2.60	-1.50	-0.27	-0.27	0.74	0.48
	0.25	0.4	-1.67	-10.62	-3.21	-1.68	-1.67	-0.67	-0.79
Expo	0.10	0.1	-1.54	-24.88	63.67	-0.95	-1.85	-4.42	-4.29
	0.10	0.4	0.44	-20.88	19.98	1.79	-0.20	-1.58	-3.29
	0.25	0.1	-1.54	-24.88	63.67	-0.95	-1.85	-4.61	-5.71
	0.25	0.4	0.44	-20.88	19.98	1.79	-0.20	-2.79	-8.36
Bump	0.10	0.1	-2.25	-16.61	-14.09	-0.81	-0.89	0.40	0.08
	0.10	0.4	-1.95	-15.32	-8.11	-1.54	-1.44	-0.28	-0.43
	0.25	0.1	-2.25	-16.61	-14.09	-0.81	-0.89	0.48	0.03
	0.25	0.4	-1.95	-15.32	-8.11	-1.54	-1.44	-0.31	-0.86
Jump	0.10	0.1	-0.73	-2.80	-4.22	-0.51	-0.50	-0.26	-0.24
	0.10	0.4	-0.82	-9.18	-1.24	-0.83	-0.49	0.22	0.27
	0.25	0.1	-0.73	-2.80	-4.22	-0.51	-0.50	-0.43	-0.27
	0.25	0.4	-0.82	-9.18	-1.24	-0.83	-0.49	-0.19	0.58
Quad	0.10	0.1	0.01	-0.28	0.17	0.01	-1.11	-0.11	-0.08
	0.10	0.4	0.03	-2.38	0.46	-0.03	-0.05	0.00	-0.22
	0.25	0.1	0.01	-0.28	0.17	0.01	-0.11	-0.05	-0.56
	0.25	0.4	0.03	-2.38	0.46	-0.03	-0.05	-0.05	-0.71
Cycle	0.10	0.1	-1.80	-11.29	-2.25	-1.88	-1.59	-1.96	-1.93
	0.10	0.4	-1.02	-9.26	-1.49	-1.26	-0.54	-0.50	-0.74
	0.25	0.1	-1.80	-11.29	-2.25	-1.88	-1.59	-2.18	-1.98
	0.25	0.4	-1.02	-9.26	-1.49	-1.26	-0.54	-0.74	-0.79

## Appendix II: SPlus code

## CDF estimation simulation

```
"cdfum2"<-
function(N = 1000, n = 100, nreps = 1000, h = 0.1, iseed = 1965, sigma = 0.1,
grid = 100, pi.s = -9, no.decimals = 2, alpha = 0.25)
{
set.seed(iseed)
x <- runif(N)
  one <- rep(1, length(x))
xint <- cbind(one, x) #epsilon iid N(0, sigma^2)
epsilon <- sigma * rnorm(N) ###define populations!!!
ratio <- 0.5 * x + x^0.5 * epsilon
linear <- 1 + 2 * (x - 0.5) + epsilon
expo <- exp(-8 * x) + epsilon
bump <- 1 + 2 * (x - 0.5) + exp(-200 * (x - 0.5)^2) + epsilon
mx <- 1 + 2 * (x - 0.5)
mx[x > 0.65] <- 0.65
jump <- mx + epsilon
quad <- 1 + 2 * (x - 0.5)^2 + epsilon
  cycle <- 2 + sin(2 * 3.141592654 * x) + epsilon
y <- cbind(ratio, linear, expo, bump, jump, quad, cycle)
quan <- apply(y, MAR = 2, FUN = "quantile", probs = alpha)
ny <- 7 ###TRUE CDF (correct model)
empiricalcdf <- function(z, quan)
{
e <- rep(0, length(z))
e[z <= quan] <- 1
cdf <- mean(e)
return(cdf)
}
pi.s <- n/N #HORVITZ-THOMPSON
###CHAMBERS & DUNSTAN CDF==cd1 includes intercept
cd0 <- function(z, x, s, quan)
{
vstar <- x^0.5
b <- (sum((z[s] * x[s])/vstar[s]^2))/(sum((x[s] * x[s])/vstar[s]^2))
U <- (z[s] - b * x[s])/vstar[s]
```

```

e <- rep(0, length(z[s]))
e[z[s] <= quan] <- 1
f <- (quan - b * x[ - s])/vstar[ - s]
M <- outer(f, U, "-")
M <- c(M)
indic <- rep(0, length(M))
indic[M >= 0] <- 1
cdf <- (1/length(x)) * (sum(e) + (1/length(x[s])) * sum(indic))
return(cdf)
}
cd1 <- function(n, N, z, xint, s, quan)
{
vstar <- rep(1, N)
vstar.s <- vstar[s]
W <- diag(1/vstar.s^2)
b <- solve(t(xint[s, ]) %*% W %*% xint[s, ]) %*% t(xint[s, ]
) %*% W %*% z[s]
U <- (z[s] - xint[s, ] %*% b)/vstar[s]
e <- rep(0, n)
e[z[s] <= quan] <- 1
f <- (quan - xint[ - s, ] %*% b)/vstar[ - s]
M <- outer(f, U, "-")
M <- c(M)
indic <- rep(0, length(M))
indic[M >= 0] <- 1
cdf <- (1/N) * (sum(e) + (1/n) * sum(indic))
return(cdf)
}
###RAO, KOVAR, & MANTEL CDF (v not = x^0.5)==rkm1 includes intercept
rkm0 <- function(n, N, z, x, s, quan)
{
e <- rep(0, n)
e[z[s] <= quan] <- 1
vstar <- x^0.5
b <- (sum((z[s] * x[s])/vstar[s]^2))/(sum((x[s] * x[s])/vstar[s
]^2))
U <- (z[s] - b * x[s])/vstar[s]
f <- (quan - b * x)/vstar

```

```

M <- outer(f, U, "-")
M <- c(M)
indic1 <- rep(0, length(M))
indic1[M >= 0] <- 1
g <- (quan - b * x[s])/vstar[s]
L <- outer(g, U, "-")
L <- c(L)
indic2 <- rep(0, length(L))
indic2[L >= 0] <- 1
cdf <- (1/N) * ((N/n) * sum(e) + (1/n) * sum(indic1) - (N/n^2) *
sum(indic2))
return(cdf)
}
rkml <- function(n, N, xint, z, quan, s)
{
e <- rep(0, n)
e[z[s] <= quan] <- 1
vstar <- rep(1, N)
W <- diag(1/vstar[s]^2)
b <- solve(t(xint[s, ]) %*% W %*% xint[s, ]) %*% t(xint[s, ]
) %*% W %*% z[s]
u <- (z[s] - xint[s, ] %*% b)/vstar[s]
f <- (quan - xint %*% b)/vstar
M <- outer(f, u, "-")
M <- c(M)
indic1 <- rep(0, length(M))
indic1[M >= 0] <- 1
g <- (quan - xint[s, ] %*% b)/vstar[s]
L <- outer(g, u, "-")
L <- c(L)
indic2 <- rep(0, length(L))
indic2[L >= 0] <- 1
cdf <- (1/N) * ((N/n) * sum(e) + (1/n) * sum(indic1) - (N/n^2) *
sum(indic2))
return(cdf)
return(W)
}
#BRING IN NONPARAMETRICS

```

```

#
# Define Epanechnikov kernel function.
#
kern <- function(x)
{
k <- rep(0, length(x))
k[abs(x) <= 1] <- 0.75 * (1 - x[abs(x) <= 1]^2)
return(k)
}
if(grid > 0) {
xmax <- max(x)
xmin <- min(x)
xstar <- xmin + ((xmax - xmin) * (0:grid))/grid
}
else {
xstar <- x
}
if(pi.s < 0) {
pi.s <- rep(n/N, n)
tmp <- (n * (n - 1))/(N * (N - 1))
pi2.s <- diag(pi.s - rep(tmp, n)) + matrix(tmp, n, n)
}
#end if-then clause
grid <- length(xstar) - 1
F.ht <- matrix(0, nreps, ny)
F.cd0 <- matrix(0, nreps, ny)
F.cd1 <- matrix(0, nreps, ny)
F.rkm0 <- matrix(0, nreps, ny)
F.rkm1 <- matrix(0, nreps, ny)
F.lpr <- matrix(0, nreps, ny)
F.dorf <- matrix(0, nreps, ny)
for(rr in 1:nreps) {
s <- sample(1:N, size = n, replace = F)
x.s <- x[s]
#Compute what's on grid of x-values, then interpolate
#using piecewise constant.
wstar <- matrix(0, n, grid + 1)
wstar0 <- wstar

```

```

wgt <- 1/pi.s
for(i in 1:(grid + 1)) {
W <- (kern((x.s - xstar[i])/h) * wgt)/h
D <- sum(W) * sum(W * (x.s - xstar[i])^2) - sum(W * (
x.s - xstar[i]))^2
wstar[, i] <- (W * (sum(W * (x.s - xstar[i])^2) - (x.s -
xstar[i]) * sum(W * (x.s - xstar[i]))))/D
wstar0[, i] <- W/sum(W)
}
freq <- table(cut(x, c((xmin - 0.01), xstar)))
what <- t(t(wstar) * c(freq))
what0 <- t(t(wstar0) * c(freq))
ww <- matrix(0, n, n)
ww0 <- ww
for(i in 1:n) {
W <- (kern((x.s - x.s[i])/h) * wgt)/h
D <- sum(W) * sum(W * (x.s - x.s[i])^2) - sum(W * (x.s -
x.s[i]))^2
ww[, i] <- (W * (sum(W * (x.s - x.s[i])^2) - (x.s - x.s[
i]) * sum(W * (x.s - x.s[i]))))/D
ww0[, i] <- W/sum(W)
}
lpr.wt <- 1/pi.s + c(apply(what, MAR = 1, FUN = "sum")) - c(
apply(ww, MAR = 1, FUN = "sum")) * (1/pi.s)
dorf.wt <- 1 + c(apply(what0[, - s], MAR = 1, FUN = "sum"))
for(i in 1:ny) {
F.ht[rr, i] <- empiricalcdf(y[s, i], quan[i])
F.cd0[rr, i] <- cd0(y[, i], x, s, quan[i])
F.cd1[rr, i] <- cd1(n, N, y[, i], xint, s, quan[i])
F.rkm0[rr, i] <- rkm0(n, N, y[, i], x, s, quan[i])
F.rkm1[rr, i] <- rkm1(n, N, xint, y[, i], quan[i], s)
indic.s <- rep(0, n)
indic.s[y[s, i] <= quan[i]] <- 1
F.lpr[rr, i] <- rbind(lpr.wt/N) %*% indic.s
F.dorf[rr, i] <- rbind(dorf.wt/N) %*% indic.s
}
}
##END LOOP ON rr

```

```

true <- rep(alpha, ny)
ht.mean <- apply(F.ht, MAR = 2, FUN = "mean")
ht.bias <- ht.mean - true
ht.var <- apply(F.ht, MAR = 2, FUN = "var")
ht.mse <- ht.var + ht.bias^2
cd0.mean <- apply(F.cd0, MAR = 2, FUN = "mean")
cd0.bias <- cd0.mean - true
cd0.var <- apply(F.cd0, MAR = 2, FUN = "var")
cd0.mse <- cd0.var + cd0.bias^2
cd1.mean <- apply(F.cd1, MAR = 2, FUN = "mean")
cd1.bias <- cd1.mean - true
cd1.var <- apply(F.cd1, MAR = 2, FUN = "var")
cd1.mse <- cd1.var + cd1.bias^2
rkm0.mean <- apply(F.rkm0, MAR = 2, FUN = "mean")
rkm0.bias <- rkm0.mean - true
rkm0.var <- apply(F.rkm0, MAR = 2, FUN = "var")
rkm0.mse <- rkm0.var + rkm0.bias^2
rkm1.mean <- apply(F.rkm1, MAR = 2, FUN = "mean")
rkm1.bias <- rkm1.mean - true
rkm1.var <- apply(F.rkm1, MAR = 2, FUN = "var")
rkm1.mse <- rkm1.var + rkm1.bias^2
lpr.mean <- apply(F.lpr, MAR = 2, FUN = "mean")
lpr.bias <- lpr.mean - true
lpr.var <- apply(F.lpr, MAR = 2, FUN = "var")
lpr.mse <- lpr.var + lpr.bias^2
dorf.mean <- apply(F.dorf, MAR = 2, FUN = "mean")
dorf.bias <- dorf.mean - true
dorf.var <- apply(F.dorf, MAR = 2, FUN = "var")
dorf.mse <- dorf.var + dorf.bias^2 #CALCULATE RELATIVE BIAS
ht.relbias <- ht.bias/true
cd0.relbias <- cd0.bias/true
cd1.relbias <- cd1.bias/true
rkm0.relbias <- rkm0.bias/true
rkm1.relbias <- rkm1.bias/true
lpr.relbias <- lpr.bias/true
dorf.relbias <- dorf.bias/true
table1 <- rbind(true, ht.bias, ht.var, ht.mse, cd0.bias, cd0.var,
cd0.mse, cd1.bias, cd1.var, cd1.mse, rkm0.bias, rkm0.var,

```

```

rkm0.mse, rkm1.bias, rkm1.var, rkm1.mse, lpr.bias, lpr.var,
lpr.mse, dorf.bias, dorf.var, dorf.mse)
table0 <- rbind(ht.relbias, cd0.relbias, cd1.relbias, rkm0.relbias,
rkm1.relbias, lpr.relbias, dorf.relbias)
percentbias <- (((table0 * 10^(no.decimals + 2)) %/% 1)/(10^(
no.decimals + 2))) * 100
#CALCULATE RELATIVE EFFICIENCY (COMPARED TO LPR)
hteфф <- ht.mse/lpr.mse
cd0eff <- cd0.mse/lpr.mse
cd1eff <- cd1.mse/lpr.mse
rkm0eff <- rkm0.mse/lpr.mse
rkm1eff <- rkm1.mse/lpr.mse
dorfeфф <- dorf.mse/lpr.mse
table2 <- rbind(hteфф, cd0eff, cd1eff, rkm0eff, rkm1eff, dorfeфф)
mse.efficiency <- ((table2 * 10^no.decimals) %/% 1)/(10^no.decimals)
}

```

## Quantile estimation simulation

```
"quant"<-
function(N = 1000, n = 100, nreps = 300, h = 0.1, iseed = 7965, sigma = 0.1,
ygrid = 20, grid = 100, pi.s = -9, no.decimals = 2, alpha = 0.5)
{
set.seed(1965)
x <- runif(N) #epsilon iid N(0, sigma^2)
epsilon <- sigma * rnorm(N)
#
###define populations
ratio <- 0.5 * x + x^0.5 * epsilon
linear <- 1 + 2 * (x - 0.5) + epsilon
expo <- exp(-8 * x) + epsilon
bump <- 1 + 2 * (x - 0.5) + exp(-200 * (x - 0.5)^2) + epsilon
mx <- 1 + 2 * (x - 0.5)
mx[x > 0.65] <- 0.65
jump <- mx + epsilon
quad <- 1 + 2 * (x - 0.5)^2 + epsilon
cycle <- 2 + sin(2 * pi * x) + epsilon
y <- cbind(ratio, linear, expo, bump, jump, quad, cycle)
ny <- 7
quan <- apply(y, MAR = 2, FUN = "quantile", probs = alpha)
set.seed(iseed)
##estis give 1's to each sample value >= alpha for the 9 different
##populations according to specified alpha, change b's for each
##estimator to its quant
esti <- function(n, e, b, alpha)
{
#m's have 1;s where cdf estimator is > alpha
m <- rep(0, n)
m[b >= alpha] <- 1
#if cdf est is > alpha, a has corresponding y value, if not it's very large
a <- m * e
a[a == 0] <- 1000000000000000
#smallest value in a is smallest value s.t. cdf >=alpha
#dd will give upper bound for cdf straddling alpha & d will give
#lower bound, high and low will give corresponding cdf
```

```

#estimates at those values
dd <- e[order(a)[1]]
d <- e[order(a)[1] - 1]
high <- b[order(a)[1]]
low <- b[order(a)[1] - 1]
return(d, dd, high, low)
}
empiricalcdf <- function(z, u)
{
e <- rep(0, length(z))
e[z <= u] <- 1
cdf <- mean(e)
return(cdf)
}
cd0 <- function(z, x, s, quan)
{
vstar <- x^0.5
b <- (sum((z[s] * x[s])/vstar[s]^2))/(sum((x[s] * x[s])/vstar[s]^2))
U <- (z[s] - b * x[s])/vstar[s]
e <- rep(0, length(z[s]))
e[z[s] <= quan] <- 1
f <- (quan - b * x[ - s])/vstar[ - s]
M <- outer(f, U, "-")
M <- c(M)
indic <- rep(0, length(M))
indic[M >= 0] <- 1
cdf <- (1/length(x)) * (sum(e) + (1/length(x[s])) * sum(indic))
return(cdf)
}
cd1 <- function(N, n, z, a, s, u)
{
one <- rep(1, N)
xint <- cbind(one, a)
vstar <- rep(1, N)
vstar.s <- vstar[s]
W <- diag(1/vstar.s^2)
b <- solve(t(xint[s, ]) %*% W %*% xint[s, ]) %*% t(xint[s, ])

```

```

) %*% W %*% z[s]
U <- (z[s] - xint[s, ] %*% b)/vstar[s]
e <- rep(0, n)
e[z[s] <= u] <- 1
f <- (u - xint[ - s, ] %*% b)/vstar[ - s]
M <- outer(f, U, "-")
M <- c(M)
indic <- rep(0, length(M))
indic[M >= 0] <- 1
cdf <- (1/N) * (sum(e) + (1/n) * sum(indic))
return(cdf)
}

####RAO, KOVAR, & MANTEL CDF (v not = x^.5)==rkml includes intercept
rkml0 <- function(N, n, z, a, s, u)
{
e <- rep(0, n)
e[z[s] <= u] <- 1
vstar <- a^0.5
b <- (sum((z[s] * a[s])/vstar[s]^2))/(sum((a[s] * a[s])/vstar[s]^2))
U <- (z[s] - b * a[s])/vstar[s]
f <- (u - b * a)/vstar
M <- outer(f, U, "-")
M <- c(M)
indic1 <- rep(0, length(M))
indic1[M >= 0] <- 1
g <- (u - b * a[s])/vstar[s]
L <- outer(g, U, "-")
L <- c(L)
indic2 <- rep(0, length(L))
indic2[L >= 0] <- 1
cdf <- (1/N) * ((N/n) * sum(e) + (1/n) * sum(indic1) - (N/n^2) *
sum(indic2))
return(cdf)
}

rkml <- function(N, n, z, a, s, u)
{
one <- rep(1, N)

```

```

xint <- cbind(one, a)
e <- rep(0, n)
e[z[s] <= u] <- 1
vstar <- rep(1, N)
W <- diag(1/vstar[s]^2)
b <- solve(t(xint[s, ]) %*% W %*% xint[s, ]) %*% t(xint[s, ]
) %*% W %*% z[s]
U <- (z[s] - xint[s, ] %*% b)/vstar[s]
f <- (u - xint %*% b)/vstar
M <- outer(f, U, "-")
M <- c(M)
indic1 <- rep(0, length(M))
indic1[M >= 0] <- 1
g <- (u - xint[s, ] %*% b)/vstar[s]
L <- outer(g, U, "-")
L <- c(L)
indic2 <- rep(0, length(L))
indic2[L >= 0] <- 1
cdf <- (1/N) * ((N/n) * sum(e) + (1/n) * sum(indic1) - (N/n^2) *
sum(indic2))
return(cdf)
return(W)
}
#BRING IN NONPARAMETRICS
# Define Epanechnikov kernel function.
kern <- function(x)
{
k <- rep(0, length(x))
k[abs(x) <= 1] <- 0.75 * (1 - x[abs(x) <= 1]^2)
return(k)
}
if(grid > 0) {
xmax <- max(x)
xmin <- min(x)
xstar <- xmin + ((xmax - xmin) * (0:grid))/grid
}
else {
xstar <- x

```

```

}
if(pi.s < 0) {
pi.s <- rep(n/N, n)
tmp <- (n * (n - 1))/(N * (N - 1))
pi2.s <- diag(pi.s - rep(tmp, n)) + matrix(tmp, n, n)
}
#end if-then clause
grid <- length(xstar) - 1
esti.ht <- matrix(0, nreps, ny)
esti.cd0 <- matrix(0, nreps, ny)
esti.cd1 <- matrix(0, nreps, ny)
esti.rkm0 <- matrix(0, nreps, ny)
esti.rkm1 <- matrix(0, nreps, ny)
esti.lpr <- matrix(0, nreps, ny)
esti.dorf <- matrix(0, nreps, ny)
for(rr in 1:nreps) {
s <- sample(1:N, size = n, replace = F)
x.s <- x[s]
y.s <- y[s, ]
yy.s <- matrix(0, n, ny)
for(i in 1:ny) {
yy.s[, i] <- sort(y.s[, i])
}
pi.s <- n/N
nlow <- n * alpha - 0.3 * n
nhigh <- n * alpha + 0.3 * n
diff <- nhigh - nlow + 1
yy.ss <- matrix(0, diff, ny)
for(i in 1:ny) {
yy.ss[, i] <- yy.s[nlow:nhigh, i]
}
#Compute mhat's on grid of x-values, then piecewise constant.
wstar <- matrix(0, n, grid + 1)
wstar0 <- wstar
wgt <- 1/pi.s
for(i in 1:(grid + 1)) {
W <- (kern((x.s - xstar[i])/h) * wgt)/h
D <- sum(W) * sum(W * (x.s - xstar[i])^2) - sum(W * (

```

```

x.s - xstar[i]))^2
wstar[, i] <- (W * (sum(W * (x.s - xstar[i])^2) - (x.s -
xstar[i]) * sum(W * (x.s - xstar[i]))))/D
wstar0[, i] <- W/sum(W)
}
freq <- table(cut(x, c((xmin - 0.01), xstar)))
what <- t(t(wstar) * c(freq))
what0 <- t(t(wstar0) * c(freq))
ww <- matrix(0, n, n)
ww0 <- ww
for(i in 1:n) {
W <- (kern((x.s - x.s[i])/h) * wgt)/h
D <- sum(W) * sum(W * (x.s - x.s[i])^2) - sum(W * (x.s -
x.s[i])^2)
ww[, i] <- (W * (sum(W * (x.s - x.s[i])^2) - (x.s - x.s[
i]) * sum(W * (x.s - x.s[i]))))/D
ww0[, i] <- W/sum(W)
}
lpr.wt <- 1/pi.s + c(apply(what, MAR = 1, FUN = "sum")) - c(
apply(ww, MAR = 1, FUN = "sum")) * (1/pi.s)
###correct weights where there may be neg weights!!!
lpr.wt[lpr.wt < 0] <- 0
a <- sum(lpr.wt)
lpr.wt <- (lpr.wt * N)/a
dorf.wt <- 1 + c(apply(what0[, - s], MAR = 1, FUN = "sum"))
dorf.wt[dorf.wt < 0] <- 0
b <- sum(dorf.wt)
dorf.wt <- (dorf.wt * N)/b
#quants evaluate cdf's for each pop at each sample value of y b.t.
#nlow and nhigh (in order)for parametric case and assign 0
#in all other sampled y places
#nonparametric case: lprquant and dorfquant are cumulative weights
#sorted by y, evaluate cdf's for all values, not just between nlow and
#nhigh...otherwise it leads to NA's when weights between nlow and nhigh
#don't straddle alpha
empquant <- matrix(0, n, ny)
for(i in 1:ny) {
for(j in nlow:nhigh) {

```

```

empquant[j, i] <- empiricalcdf(y.s[, i], yy.s[j,
  i])
}
}
cd0quant <- matrix(0, n, ny)
cd0interp <- rep(0, ny)
for(i in 1:ny) {
for(j in nlow:nhigh) {
cd0quant[j, i] <- cd0(y[, i], x, s, yy.s[j, i])
}
z <- esti(n, yy.s[, i], cd0quant[, i], alpha)
maxvaluey <- z$dd
minvaluey <- z$d
cdfhigh <- z$high
cdflow <- z$low
cd0interp[i] <- minvaluey + ((maxvaluey - minvaluey)/(
cdfhigh - cdflow)) * (alpha - cdflow)
}
cd1quant <- matrix(0, n, ny)
cd1interp <- rep(0, ny)
for(i in 1:ny) {
for(j in nlow:nhigh) {
cd1quant[j, i] <- cd1(N, n, y[, i], x, s, yy.s[
  j, i])
}
z <- esti(n, yy.s[, i], cd1quant[, i], alpha)
maxvaluey <- z$dd
minvaluey <- z$d
cdfhigh <- z$high
cdflow <- z$low
cd1interp[i] <- minvaluey + ((maxvaluey - minvaluey)/(
cdfhigh - cdflow)) * (alpha - cdflow)
}
rkm0quant <- matrix(0, n, ny)
rkm0interp <- rep(0, ny)
for(i in 1:ny) {
for(j in nlow:nhigh) {
rkm0quant[j, i] <- rkm0(N, n, y[, i], x, s,

```

```

    yy.s[j, i])
  }
  z <- esti(n, yy.s[, i], rkm0quant[, i], alpha)
  maxvaluey <- z$dd
  minvaluey <- z$d
  cdfhigh <- z$high
  cdflow <- z$low
  rkm0interp[i] <- minvaluey + ((maxvaluey - minvaluey)/(
cdfhigh - cdflow)) * (alpha - cdflow)
}
rkm1quant <- matrix(0, n, ny)
rkm1interp <- rep(0, ny)
for(i in 1:ny) {
  for(j in nlow:nhigh) {
    rkm1quant[j, i] <- rkm1(N, n, y[, i], x, s,
      yy.s[j, i])
  }
  z <- esti(n, yy.s[, i], rkm1quant[, i], alpha)
  maxvaluey <- z$dd
  minvaluey <- z$d
  cdfhigh <- z$high
  cdflow <- z$low
  rkm1interp[i] <- minvaluey + ((maxvaluey - minvaluey)/(
cdfhigh - cdflow)) * (alpha - cdflow)
}
#nonparametric quants are based on sorted y's, so sort y sample for the grid
sortlpr.wt <- matrix(0, n, ny)
for(i in 1:ny) {
  sortlpr.wt[, i] <- lpr.wt[sort.list(y.s[, i])]
}
sortdorf.wt <- matrix(0, n, ny)
for(i in 1:ny) {
  sortdorf.wt[, i] <- dorf.wt[sort.list(y.s[, i])]
}
lprquant <- cbind(lpr.wt, lpr.wt, lpr.wt, lpr.wt, lpr.wt,
  lpr.wt, lpr.wt)
lprinterp <- rep(0, ny)
for(i in 1:ny) {

```

```

lprquant[, i] <- cumsum(sortlpr.wt[, i])/N
z <- esti(n, yy.s[, i], lprquant[, i], alpha)
maxvaluey <- z$dd
minvaluey <- z$d
cdfhigh <- z$high
cdflow <- z$low
lprinterp[i] <- minvaluey + ((maxvaluey - minvaluey)/(
cdfhigh - cdflow)) * (alpha - cdflow)
}
dorfquant <- cbind(dorf.wt, dorf.wt, dorf.wt, dorf.wt, dorf.wt,
dorf.wt, dorf.wt)
dorfinterp <- rep(0, ny)
for(i in 1:ny) {
dorfquant[, i] <- cumsum(sortdorf.wt[, i])/N
z <- esti(n, yy.s[, i], dorfquant[, i], alpha)
maxvaluey <- z$dd
minvaluey <- z$d
cdfhigh <- z$high
cdflow <- z$low
dorfinterp[i] <- minvaluey + ((maxvaluey - minvaluey)/(
cdfhigh - cdflow)) * (alpha - cdflow)
}
for(i in 1:ny) {
esti.ht[rr, i] <- esti(n, yy.s[, i], empquant[, i],
alpha)$dd
esti.cd0[rr, i] <- cd0interp[i]
esti.cd1[rr, i] <- cd1interp[i]
esti.rkm0[rr, i] <- rkm0interp[i]
esti.rkm1[rr, i] <- rkm1interp[i]
esti.lpr[rr, i] <- lprinterp[i]
esti.dorf[rr, i] <- dorfinterp[i]
}
}
true <- quan # ht.mean <- apply(esti.ht, MAR = 2, FUN = "mean")
}

```

## Semiparametric cdf estimation

```
"newsemitoo"<-  
function(grid = 100, grid2 = 999, iseed=1965, clust=NewUse$LAKE.ID)  
{ set.seed(iseed)  
  N_length(NELakes.Frame[,1])  
  x_NELakes.Frame[,16]  
  h_0.25*(max(x)-min(x))  
  z1_rep(1,N)  
  z2_NELakes.Frame[,17]  
  z3_NELakes.Frame[,18]  
  #create dummy variables for eco code  
  #To see the different values of eco code:  
  #> unique(NELakes.Frame[,22])  
  ##[1] 82 0 58 83 59 84 60 62 67 61 64 63  
  ##eco code "67" not included in sample frame, leave that out  
  eco1_rep(0,length(NELakes.Frame[,22]))  
  eco2_eco1  
  eco3_eco1  
  eco4_eco1  
  eco5_eco1  
  eco6_eco1  
  eco7_eco1  
  eco8_eco1  
  eco9_eco1  
  eco10_eco1  
  eco1[NELakes.Frame[,22]==82]_1  
  eco2[NELakes.Frame[,22]==0]_1  
  eco3[NELakes.Frame[,22]==58]_1  
  eco4[NELakes.Frame[,22]==83]_1  
  eco5[NELakes.Frame[,22]==59]_1  
  eco6[NELakes.Frame[,22]==84]_1  
  eco7[NELakes.Frame[,22]==60]_1  
  eco8[NELakes.Frame[,22]==62]_1  
  eco9[NELakes.Frame[,22]==63]_1  
  eco10[NELakes.Frame[,22]==61]_1  
  zz_cbind(z1,z2,z3,eco1,eco2,eco3,eco4,eco5,eco6,eco7,eco8,eco9,eco10)  
  #use tapply to average over observations on the same lake
```

```

x.s_NewUse[,16]
x.s_tapply(x.s,clust,mean)
y.s_NewUse[,56]
y.s_tapply(y.s,clust,mean)
n_length(y.s)
z1.s_rep(1,n)
z2.s_NewUse[,17]
z2.s_tapply(z2.s,clust,mean)
z3.s_NewUse[,18]
z3.s_tapply(z3.s,clust,mean)
eco1.s_rep(0,length(chemplus[,20]))
eco2.s_eco1.s
eco3.s_eco1.s
eco4.s_eco1.s
eco5.s_eco1.s
eco6.s_eco1.s
eco7.s_eco1.s
eco8.s_eco1.s
eco9.s_eco1.s
eco10.s_eco1.s
eco11.s_eco1.s
eco1.s[chemplus[,20]==82]_1
eco1.s_tapply(eco1.s,clust,mean)
eco2.s[chemplus[,20]==0]_1
eco2.s_tapply(eco2.s,clust,mean)
eco3.s[chemplus[,20]==58]_1
eco3.s_tapply(eco3.s,clust,mean)
eco4.s[chemplus[,20]==83]_1
eco4.s_tapply(eco4.s,clust,mean)
eco5.s[chemplus[,20]==59]_1
eco5.s_tapply(eco5.s,clust,mean)
eco6.s[chemplus[,20]==84]_1
eco6.s_tapply(eco6.s,clust,mean)
eco7.s[chemplus[,20]==60]_1
eco7.s_tapply(eco7.s,clust,mean)
eco8.s[chemplus[,20]==62]_1
eco8.s_tapply(eco8.s,clust,mean)
eco9.s[chemplus[,20]==63]_1

```

```

eco9.s_tapply(eco9.s,clust,mean)
eco10.s[chemplus[,20]==61]_1
eco10.s_tapply(eco10.s,clust,mean)
zz.s_cbind(z1.s,z2.s,z3.s,eco1.s,eco2.s,eco3.s,eco4.s,eco5.s,eco6.s,
eco7.s,eco8.s,eco9.s,eco10.s)
#Define Epanechnikov kernel function
kern_function(x)
{
k_rep(0, length(x))
k[abs(x) <= 1]_0.75*(1 - x[abs(x) <= 1]^2)
return(k)
}
xmax_max(x)
xmin_min(x)
xstar_sort(x.s)
#weights are inverses of inclusion probabilities
wgt.1_NewUse[,140]
wgt.1_tapply(wgt.1,clust,mean)
wgt_wgt.1/2
j_rep(1,n)
ident_matrix(0,n,n)
ident_diag(j)
pia_matrix(0,n,n)
pivector_1/wgt
pia_diag(pivector)
e_c(1,0)
Sa_matrix(0,n,n)
W_matrix(0,n,n)
for(i in 1:n){
xmat1_rep(1,n)
xmat2_x.s - x.s[i]
Xmatrix_cbind(xmat1,xmat2)
Wvec(kern((x.s - x.s[i])/h)*wgt)/h
W_diag(Wvec)
#sample smoother matrix:
Sa[i,]_t(e)%*%solve(t(Xmatrix)%*%W%*%Xmatrix)%*%t(Xmatrix)%*%W
}
###Sstar is centered sample smoother matrix

```

```

Sstar_(ident - j**t(j)**solve(pia)/N)**Sa
empiricalcdf_function(z,s,wgt,N)
{  e_rep(0,length(z))
e[z<=s]_1
u_t(wgt)**e
g_sum(wgt)
cdf_(1/g)*u
return(cdf)
}
Bhat_solve(t(zz.s)**solve(pia)**(ident - Sstar)**zz.s)
      **t(zz.s)**solve(pia)**(ident - Sstar)
ypart_t(j)**solve(pia)
zpart_(c(apply(zz,MAR=2,FUN="sum"))) - t(j)**solve(pia)**zz.s)**Bhat
##compute mhat's on sampled x-values
mhat_rep(0,n)
for(i in 1:n) {
mhat[i]_Sa[i,]**(ident - zz.s)**Bhat)
}
freq_table(cut(x,c((xmin - 0.01), xstar)))
mhat2_mhat[order(x.s)]
Umhat_mhat2*c(freq)
mpart_Umhat - mhat
lpr.wt_ypart + mpart + zpart
lpr.wt[lpr.wt < 0] _ 0
a_sum(lpr.wt)
lpr.wt_(lpr.wt*N)/a
varclust_tapply(NewUse$CLUSTER.1,clust,mean)
#n.var is the number per cluster in the sample
n.var_tapply(y.s,varclust,length)
# 1 2 3 4 5 6 7 8 9 10 11 12 13 14
# 29 19 25 27 53 34 38 57 41 30 27 122 30 21
#n.h assigns length of cluster i to all y.s in cluster i
n.h_rep(0,n)
for(i in 1:14){
n.h[varclust==i]_n.var[i]
}
#Create grid of values on which to calculate cdf estimates
y_-200 + ((4200)*(0:grid2))/grid2

```

```

ylength_length(y)
F.lpr <- rep(0, ylength)
F.emp <- rep(0, ylength)
v.ht_rep(0,ylength)
v.lpr_v.ht
cv.ht_v.ht
cv.lpr_v.ht
ci.min.lpr_rep(0,ylength)
ci.min.ht_ci.min.lpr
ci.max.lpr_ci.min.lpr
ci.max.ht_ci.min.lpr
ci.length.lpr_ci.min.lpr
ci.length.ht_ci.min.lpr
for(i in 1:ylength){
  indic.s <- rep(0, n)
  indic.s[y.s <= y[i]] <- 1
  Bhat2_solve(t(zz.s)%%solve(pia)%%(ident - Sstar)%%zz.s)
    %%t(zz.s)%%solve(pia)%%(ident - Sstar)%%indic.s
  ##compute mhat's on sampled x-values
  mhat2_rep(0,n)
  for(i in 1:n) {
  mhat2[i]_Sa[i,]%%(indic.s - zz.s%%Bhat2)
  }
  ##resid is weighted resid
  resid_rep(0,n)
  for(i in 1:n){
  resid[i]_(indic.s[i] - mhat2[i] - zz.s[i,]%%Bhat2)*lpr.wt[i]
  }
  F.lpr[i] <- rbind(lpr.wt/N) %% indic.s
  F.emp[i]_empiricalcdf(y.s,y[i],wgt,N)
  ##VARIANCE ESTIMATES...
  #l.var.'s are vectors of within cluster variances of z.ht,z.lpr
  z.lpr_resid*n.h/N
  l.var.lpr_tapply(z.lpr,varclust,var)
  v.lpr[i]_sum(l.var.lpr/n.var)
  cv.lpr[i]_v.lpr[i]^0.5/F.lpr[i]*100
  ci.min.lpr[i]_F.lpr[i] - 1.96*(v.lpr[i]^0.5)
  ci.max.lpr[i]_F.lpr[i] + 1.96*(v.lpr[i]^0.5)

```

```

ci.length.lpr[i]_ci.max.lpr[i]-ci.min.lpr[i]
ci.lpr_cbind(ci.min.lpr,ci.max.lpr,ci.length.lpr)
totalwidth.lpr_sum(ci.length.lpr)
avewidth.lpr_totalwidth.lpr/ylength
gg_sum(wgt)
z.ht_(indic.s - F.emp[i])*wgt*n.h/gg
l.var.ht_tapply(z.ht,varclust,var)
v.ht[i]_sum(l.var.ht/n.var)
cv.ht[i]_v.ht[i]^0.5/F.emp[i]*100
ci.min.ht[i]_F.emp[i] - 1.96*(v.ht[i]^0.5)
ci.max.ht[i]_F.emp[i] + 1.96*(v.ht[i]^0.5)
ci.length.ht[i]_ci.max.ht[i]-ci.min.ht[i]
ci.ht_cbind(ci.min.ht,ci.max.ht,ci.length.ht)
totalwidth.ht_sum(ci.length.ht)
avewidth.ht_totalwidth.ht/ylength
}
###FOR ONE POINT ANC=0
indic.s.0 <- rep(0, n)
indic.s.0[y.s <= 0] <- 1
Bhat2.0_solve(t(zz.s)%solve(pia)%%(ident - Sstar)%zz.s)
      %t(zz.s)%solve(pia)%%(ident - Sstar)%indic.s.0
##compute mhat's on sampled x-values
mhat2.0_rep(0,n)
for(i in 1:n) {
mhat2.0[i]_Sa[i,]%%(indic.s.0 - zz.s)%Bhat2.0)
}
##resid.0 is weighted resid
resid.0_rep(0,n)
for(i in 1:n){
resid.0[i]_(indic.s.0[i] - mhat2.0[i] - zz.s[i,]%Bhat2.0)*lpr.wt[i]
}
F.lpr.0 <- rbind(lpr.wt/N) %indic.s.0
F.emp.0_empiricalcdf(y.s,0,wgt,N)
z.lpr.0_resid.0*n.h/N
l.var.lpr.0_tapply(z.lpr.0,varclust,var)
v.lpr.0_sum(l.var.lpr.0/n.var)
cv.lpr.0_v.lpr.0^0.5/F.lpr.0*100
ci.min.lpr.0_F.lpr.0 - 1.96*(v.lpr.0^0.5)

```

```

ci.max.lpr.0_F.lpr.0 + 1.96*(v.lpr.0^0.5)
ci.length.lpr.0_ci.max.lpr.0-ci.min.lpr.0
ci.lpr.0_c(ci.min.lpr.0,ci.max.lpr.0,ci.length.lpr.0)
gg_sum(wgt)
z.ht.0_(indic.s.0 - F.emp.0)*wgt*n.h/gg
l.var.ht.0_tapply(z.ht.0,varclust,var)
v.ht.0_sum(l.var.ht.0/n.var)
cv.ht.0_v.ht.0^0.5/F.emp.0*100
ci.min.ht.0_F.emp.0 - 1.96*(v.ht.0^0.5)
ci.max.ht.0_F.emp.0 + 1.96*(v.ht.0^0.5)
ci.length.ht.0_ci.max.ht.0-ci.min.ht.0
ci.ht.0_c(ci.min.ht.0,ci.max.ht.0,ci.length.ht.0)
rr_c(1:length(v.lpr))
plot(rr,v.lpr)
plot(rr,v.ht)
plot(y,F.lpr)
plot(y, F.emp)
plot(c(y,y),c(F.emp,F.lpr),type="n",xlab="y",ylab="prob")
lines(y,F.lpr)
points(y,F.emp)
plot(y,F.lpr,type="n",xlab="y",ylab="probability")
title("ANC CDF Estimation")
lines(y,ci.min.ht, lty=7,col=5)
lines(y,ci.max.ht,lty=7,col=5)
lines(y,ci.min.lpr,lty=5,col=1)
lines(y,ci.max.lpr,lty=5,col=1)
lines(y,F.lpr,lty=1,col=1)
lines(y,F.emp,lty=1,col=5)
legend(2000,0.4,legend=c("HT CDF","HT Bounds","SEMI CDF","SEMI Bounds"),
      lty=c(1,7,1,5), col=c(5,5,1,1))
}

```