

MASTER'S PROJECT

NONPARAMETRIC SURVEY REGRESSION ESTIMATION IN TWO-STAGE
SPATIAL SAMPLING

Submitted by
Siobhan Everson-Stewart
Department of Statistics

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

ABSTRACT OF MASTER'S PROJECT
NONPARAMETRIC SURVEY REGRESSION ESTIMATION IN TWO-STAGE
SPATIAL SAMPLING

A nonparametric model-assisted survey estimator based on local polynomial regression is extended to incorporate spatial auxiliary information and two-stage sampling designs. Under mild assumptions, this estimator is asymptotically design-unbiased and consistent. Simulation studies show that the nonparametric regression estimator is competitive with standard parametric techniques when the parametric specification is correct, and outperforms those techniques when the parametric specification is incorrect. The methodology is applied to water chemistry data from the EMAP Northeastern Lakes Survey.

Siobhan Everson-Stewart
Statistics Department
Colorado State University
Fort Collins, CO 80523
Spring 2003

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

1 Introduction

1.1 Motivation

Many surveys contain auxiliary information at the population level in addition to sample data. This auxiliary information can come from satellite images, GPS data, aerial photographs, or other sources. In a common survey situation, a statistical agency, often a federal, state, or tribal, collects data and auxiliary information. A data set is then created and released to the users. This data set reflects knowledge of both the design and auxiliary information. The end users are then responsible for estimating the status of many study variables. The U.S. EPA's EMAP (Environmental Monitoring and Assessment Program) Northeastern Lakes survey is a good example of this. The EPA had some information about all the lakes in the region of interest, such as longitude, latitude, and elevation. A sample of lakes was then taken. Each of the sampled lakes was visited, and additional measurements were taken. The final data set includes both this sample information and the auxiliary information previously available.

In this situation, it is of interest to develop a model relating sample and auxiliary information so that inferences can be made about non-sampled population values. However, in this modeling environment, time and other resources are often limited. The agency using the data may need quick answers to questions about population status. Their statistical resources may also be limited. Additionally, there may be controversy among the end users. The preferred model should not be influenced by user biases. This ideal model should be capable of handling many study variables without requiring modeling efforts for each. It should also be efficient if the model is correct, but not fail if the model is wrong. In this situation, that means the estimator should have small variance compared to other available estimators when the model is specified correctly, but still give reasonable answers if this specification is incorrect. That is, the estimator should be approximately unbiased and consistent.

Breidt and Opsomer (2000) developed a local polynomial regression estimator for direct element sampling with scalar auxiliary information available for every item in the population. This estimator is asymptotically unbiased, and more efficient than regression models when the model regression function is not correctly specified, while being competitive with the regression estimator when the regression model is correct. In this paper, the method developed by Breidt and Opsomer is extended to two-stage sampling designs with bivariate auxiliary information. This extension has numerous practical applications, as many surveys have complicated study designs, often including two stages of sampling. Also, many data sets include more than one auxiliary variable. Frequently, for example, the spatial location of every population element is known. For concreteness, here we consider the auxiliary vector to be spatial coordinates of the first stage sampling unit, though any other auxiliary vector would be handled the same way.

1.2 Overview of Paper

In this paper, we first define some of the notation used for two stages of sampling. Additionally, the population parameters of interest are identified. Then we will review model assisted estimation, in both the parametric and nonparametric cases. The Horvitz-Thompson estimator and the regression estimator are examined, in addition to nonparametric estimators such as local polynomial regression. Kernel smoothing is explained, and then the nonparametric regression estimator is presented. Variance estimation for this estimator is described. The nonparametric regression technique is examined through a simulation study, and then applied to data from the EMAP Northeastern Lakes study in the empirical example. The paper concludes with a discussion of results and further topics of research. Finally, appendices containing the S-Plus code for the simulation studies and the empirical example are included.

2 Methods

2.1 Notation

Consider a finite population of elements $U = \{1, \dots, k, \dots, N\}$ partitioned into M clusters, $U_1, \dots, U_i, \dots, U_M$. The population of clusters is represented as $C = \{1, \dots, i, \dots, M\}$. The number of elements in the i th cluster U_i is denoted N_i . We have $U = \cup_{i \in C} U_i$ and $N = \sum_{i \in C} N_i$. For all clusters $i \in C$, an auxiliary vector $\mathbf{x}_i = (x_i, y_i)'$ of spatial coordinates is available.

At stage one, a probability sample s of clusters is drawn from C according to a fixed size design $p_I(\cdot)$, where $p_I(s)$ is the probability of drawing the sample s from C . Let m be the size of s . The cluster inclusion probabilities $\pi_i = \Pr\{i \in s\} = \sum_{s: i \in s} p_I(s)$ and $\pi_{ij} = \Pr\{i, j \in s\} = \sum_{s: i, j \in s} p_I(s)$ are assumed to be strictly positive, where p_I refers to first-stage design.

For every sampled cluster $i \in s$, a probability sample s_i of elements is drawn from U_i according to a fixed size design $p_i(\cdot)$ with inclusion probabilities $\pi_{k|i}$ and $\pi_{kl|i}$. That is, $p_i(s_i)$ is the probability of drawing s_i from U_i given that the i th cluster is chosen at stage one. The size of s_i is denoted n_i . Assume that $\pi_{k|i} = \Pr\{k \in s_i | s \ni i\} = \sum_{s_i: k \in s_i} p_i(s_i)$ and $\pi_{kl|i} = \Pr\{k, l \in s_i | s \ni i\} = \sum_{s_i: k, l \in s_i} p_i(s_i)$ are strictly positive. As is customary for two-stage sampling, we assume invariance and independence of the second-stage design. Invariance of the second-stage design means that for every i , and for every $s \in i$, $p_i(\cdot | s) = p_i(\cdot)$. That is, the same within-cluster design is used whenever the i th cluster is selected, regardless of what other clusters are selected. Independence of the second-stage design means that sub-sampling in a given cluster is independent of sub-sampling in any other cluster.

The whole sample of elements and its size are $\cup_{i \in s} s_i$ and $\sum_{i \in s} n_i$, respectively. The study variable z_k is observed for $k \in \cup_{i \in s} s_i$. The parameter of interest is the average cluster mean, $\theta_z = M^{-1} \sum_{i \in C} \sum_{j \in U_i} z_j / N_i$, or the population total $t_z = \sum_{k \in U} z_k = \sum_{i \in C} t_i$, where $t_i = \sum_{k \in U_i} z_k$ is the i th cluster total.

Let $I_i = 1$ if $i \in s$ and $I_i = 0$ otherwise. Note that $E_p [I_i] = E_I [E_{II}[I_i]] = E_I [I_i] = \pi_i$, where $E_p[\cdot]$ denotes expectation with respect to the sampling design, $E_I[\cdot]$ denotes expectation with respect to stage one, and $E_{II}[\cdot]$ denotes conditional expectation with respect to stage two given s . Also, $V_I(\cdot)$ and $V_{II}(\cdot)$ denote variances with respect to stage one and two, respectively. Using this notation, an estimator \hat{t} of t is said to be design-unbiased if $E_p [\hat{t}] = t$.

2.2 Model-Assisted Estimation

Here, we are assuming that we have auxiliary information, \mathbf{x}_i for each cluster, $i \in C$. The response for the i th cluster (either t_i or $\bar{z}_i = N_i^{-1} \sum_{j \in U_i} z_j$) is assumed to be a smooth function of \mathbf{x} plus random error, i.e.

$$\begin{aligned} t_i &= g_t(\mathbf{x}_i) + \epsilon_i \\ \bar{z}_i &= g_z(\mathbf{x}_i) + \epsilon_i \end{aligned}$$

where $g_t(\mathbf{x}_i)$, $g_z(\mathbf{x}_i)$ are smooth functions of \mathbf{x}_i , ϵ_i are independent random variables with zero mean and variance $v(\mathbf{x}_i)$, a smooth, strictly positive function of \mathbf{x}_i .

Model-assisted estimators are formed from a model-based prediction, plus a design bias adjustment:

$$\hat{\theta}_z = \frac{1}{M} \left\{ \sum_{i \in C} \hat{\mu}_i + \sum_{i \in s} \frac{\hat{z}_i - \hat{\mu}_i}{\pi_i} \right\}$$

where $\hat{\mu}_i$ is the fitted model mean for the i th cluster and

$$\hat{z}_i = \left(\sum_{j \in s_i} \frac{z_{ij}}{\pi_{j|i}} \right) \left(\sum_{j \in s_i} \frac{1}{\pi_{j|i}} \right)^{-1},$$

the estimate of the i th cluster mean based on the sampled observations. Note that in the case of equal sampling weights, \hat{z}_i becomes the empirical cluster mean.

These estimators are approximately design-unbiased, with small variance if the model is correct. For the popular Horvitz-Thompson estimator, $\hat{\mu}_i \equiv 0$ since no auxiliary information is used. In generalized regression, $\hat{\mu}_i = \mathbf{x}_i' \hat{\beta}$, while $\hat{\mu}_i$ comes from a kernel smooth in local polynomial regression estimation.

2.2.1 Parametric Model-Assisted Estimation

In parametric model-assisted estimation, a parametric model is assumed for the relationship between any auxiliary information and the response. The Horvitz-Thompson and regression estimators are two of the most common parametric model-assisted estimators.

The Horvitz-Thompson (1952) estimator of t_z in two-stage element sampling is given by

$$\hat{t}_z = \sum_{i \in s} \frac{\hat{t}_i}{\pi_i} = \sum_{i \in C} \frac{\hat{t}_i I_i}{\pi_i}, \quad (1)$$

where

$$\hat{t}_i = \sum_{k \in s_i} \frac{z_k}{\pi_{k|i}}$$

is the Horvitz-Thompson estimator of t_i with respect to stage two. Since \hat{t}_i is design-unbiased for t_i , the Horvitz-Thompson estimator \hat{t}_z is design-unbiased for t_z . Note that \hat{t}_z does not depend on the auxiliary information. The variance of the Horvitz-Thompson estimator \hat{t}_z can be written as the sum of two components,

$$\begin{aligned} \text{Var}_p(\hat{t}_z) &= V_I(\mathbf{E}_{II}[\hat{t}_z]) + \mathbf{E}_I[V_{II}(\hat{t}_z)] \\ &= \sum_{i,j \in C} (\pi_{ij} - \pi_i \pi_j) \frac{t_i}{\pi_i} \frac{t_j}{\pi_j} + \sum_{i \in C} \frac{V_i}{\pi_i}, \end{aligned} \quad (2)$$

where

$$\begin{aligned} V_i &= V_{II}(\hat{t}_i) \\ &= \sum_{k,l \in U_i} (\pi_{kl|i} - \pi_{k|i} \pi_{l|i}) \frac{z_k}{\pi_{k|i}} \frac{z_l}{\pi_{l|i}} \end{aligned}$$

is the variance of \hat{t}_i with respect to stage two. A design-unbiased estimator of V_i is given by

$$\hat{V}_i = \sum_{k,l \in s_i} \frac{\pi_{kl|i} - \pi_{k|i} \pi_{l|i}}{\pi_{kl|i}} \frac{z_k}{\pi_{k|i}} \frac{z_l}{\pi_{l|i}}.$$

Note that V_i is non-random due to invariance. Also, the result for single-stage cluster sampling, in which all elements in each selected cluster are observed, is obtained if we set $\hat{t}_i = t_i$ and $V_i = \hat{V}_i = 0$ for all $i \in C$. See Särndal, Swensson, and Wretman (1992, Result 4.3.1).

The analogous estimator of θ_z is

$$\begin{aligned} \hat{\theta}_z &= \frac{1}{M} \sum_{i \in s} \frac{\hat{z}_i}{\pi_i} \\ &= \frac{1}{M} \sum_{i \in s} \left(\left(\frac{1}{\pi_i} \right) \left(\sum_{j \in s_i} \frac{z_j}{\pi_{j|i}} \right) \left(\sum_{j \in s_i} \frac{1}{\pi_{j|i}} \right)^{-1} \right). \end{aligned}$$

The variance of this estimator is

$$\text{Var}[\hat{\theta}_z] = \frac{1}{M^2} \left[\sum_{i \in C} \sum_{j \in C} (\pi_{ij} - \pi_i \pi_j) \frac{t_i}{N_i \pi_i} \frac{t_j}{N_j \pi_j} + \sum_{i \in C} \frac{V_i}{\pi_i} \right]$$

where

$$V_i = \frac{1}{N_i^2} \sum_{k \in U_i} \sum_{l \in U_i} (\pi_{kl|i} - \pi_{k|i} \pi_{l|i}) \left(\frac{z_k - \bar{z}_i}{\pi_{k|i}} \right) \left(\frac{z_l - \bar{z}_i}{\pi_{l|i}} \right)$$

and

$$\bar{z}_i = \frac{1}{N_i} \sum_{j \in U_i} z_j.$$

The regression estimator of t_z in two-stage element sampling, as given by Särndal, Swensson, and Wretman (1992), takes the form

$$\hat{t}_{zr} = \sum_{i \in C} \hat{t}_{ir} + \sum_{i \in s} \frac{\hat{t}_i - \hat{t}_{ir}}{\pi_i}$$

where \hat{t}_i is the Horvitz-Thompson estimator of the i th cluster total, as given above. The \hat{t}_{ir} are the regression model estimates of the i th cluster totals,

$$\hat{t}_{ir} = \mathbf{x}'_{ir} \hat{\mathbf{B}}$$

where $\mathbf{x}'_{ir} = [1 \ x_i \ y_i]$ and $\hat{\mathbf{B}}$ is found from

$$\hat{\mathbf{B}} = \left(\sum_{i \in s} \frac{\mathbf{x}_{ir} \mathbf{x}'_{ir}}{\pi_i} \right)^{-1} \sum_{i \in s} \frac{\mathbf{x}_{ir} \hat{t}_i}{\pi_i}.$$

If t_{ir} is defined as the regression estimate of the i th cluster total when the response is known for the entire population (i.e. $t_{ir} = \mathbf{x}'_{ir} \mathbf{B}$, $\mathbf{B} = (\sum_{i \in C} \mathbf{x}_{ir} \mathbf{x}'_{ir})^{-1} \sum_{i \in C} \mathbf{x}_{ir} t_i$), then define $D_i = t_i - t_{ir}$, the residual of the i th cluster. Also define $\Delta_{kl} = \pi_{kl} - \pi_k \pi_l$. Then the approximate variance of the regression estimator is (Särndal, Swensson, and Wretman, p.311):

$$\text{AV}(\hat{t}_{zr}) = \text{AV}_{PSU} + \text{AV}_{SSU}$$

where

$$\text{AV}_{PSU} = \sum_{i \in C} \sum_{j \in C} \Delta_{ij} \frac{D_i D_j}{\pi_i \pi_j}$$

and

$$\text{AV}_{SSU} = \sum_{i \in C} \frac{V_i}{\pi_i}.$$

Here,

$$V_i = \sum_{k \in U_i} \sum_{l \in U_i} \Delta_{ij|i} \frac{z_i}{\pi_{k|i}} \frac{z_j}{\pi_{l|i}}.$$

Regression estimation of θ_z has a similar form:

$$\hat{\theta}_{zr} = \frac{1}{M} \left\{ \sum_{i \in C} \hat{\mu}_{ir} + \sum_{i \in s} \frac{\hat{z}_i - \hat{\mu}_{ir}}{\pi_i} \right\}.$$

The $\hat{\mu}_{ir}$ are the regression estimates of the i th cluster average:

$$\hat{\mu}_{ir} = \mathbf{x}'_{ir} \hat{\mathbf{B}}.$$

In this case,

$$\hat{\mathbf{B}} = \left(\sum_{i \in s} \frac{\mathbf{x}_{ir} \mathbf{x}'_{ir}}{\pi_i} \right)^{-1} \sum_{i \in s} \frac{\mathbf{x}_{ir} \hat{z}_i}{\pi_i}.$$

Define $\bar{z}_i = \frac{1}{N_i} \sum_{m \in U_i} z_m$ and \bar{z}_{ir} as the regression estimate of the i th cluster mean when the entire population is known, $\bar{z}_{ir} = \mathbf{x}'_{ir} \mathbf{B}$, where $\mathbf{B} = (\sum_{i \in C} \mathbf{x}_{ir} \mathbf{x}'_{ir})^{-1} \sum_{i \in C} \mathbf{x}_{ir} \bar{z}_i$. With $D_i = \bar{z}_i - \hat{z}_{ir}$, the asymptotic variance of this estimator is:

$$\frac{1}{M^2} \left[\sum_{i \in C} \sum_{j \in C} \Delta_{ij} \frac{D_i D_j}{\pi_i \pi_j} + \sum_{i \in C} \frac{V_i}{\pi_i} \right],$$

where

$$\Delta_{ij} = \pi_{ij} - \pi_i \pi_j$$

and

$$V_i = \frac{1}{N_i^2} \sum_{k \in U_i} \sum_{l \in U_i} (\pi_{kl|i} - \pi_{k|i} \pi_{l|i}) \left(\frac{z_k - \bar{z}_i}{\pi_{k|i}} \right) \left(\frac{z_l - \bar{z}_i}{\pi_{l|i}} \right).$$

2.2.2 Nonparametric Model-Assisted Estimation

In some ways, nonparametric model-assisted estimation is very similar to parametric model-assisted estimation. The basic form of the estimator is the same: a model based prediction, plus a design bias adjustment. In this case, however, the model based prediction comes from a nonparametric model rather than a parametric one. The remainder of the paper is spent developing and evaluating a nonparametric model-assisted estimator based on nonparametric regression.

2.3 Smoothing

With local polynomial regression, locally weighted regression is used to estimate μ_i . The weights for this local polynomial regression come from a kernel function. To use this smoothing technique, one must make a few assumptions as explained in Wand and Jones (1995). First,

$$z_i = \mu(\mathbf{x}_i) + v^{\frac{1}{2}}(\mathbf{x}_i) \epsilon_i$$

where $\mu(\mathbf{x}_i)$ is a smooth function of \mathbf{x}_i and $v(\mathbf{x}_i)$, the variance of z_i , can vary with \mathbf{x}_i . Also, $E(\epsilon_i) = 0$, $\text{Var}(\epsilon_i) = 1$, and ϵ_i, ϵ_j are assumed to be independent for $i \neq j$.

To find the value of the smooth at a point \mathbf{x}_i , one performs weighed least squares regression and reads off the local intercept. Figure 1 depicts this for the case of scalar x_i . The kernel function provides the weights for the weighted least squares regression. Often, K , the kernel function, is a density, such as the normal. Another common kernel is the Epanechnikov function, which is $3/4(1 - u^2)I_{\{|u| < 1\}}$ in the one-dimensional case (See Figure 2).

In one dimension, $h^{-1}K((x_j - x_i)/h)$ gives the weights for the local regression at the point x_i for each data point, x_j . The bandwidth, h , is a smoothing parameter. The larger

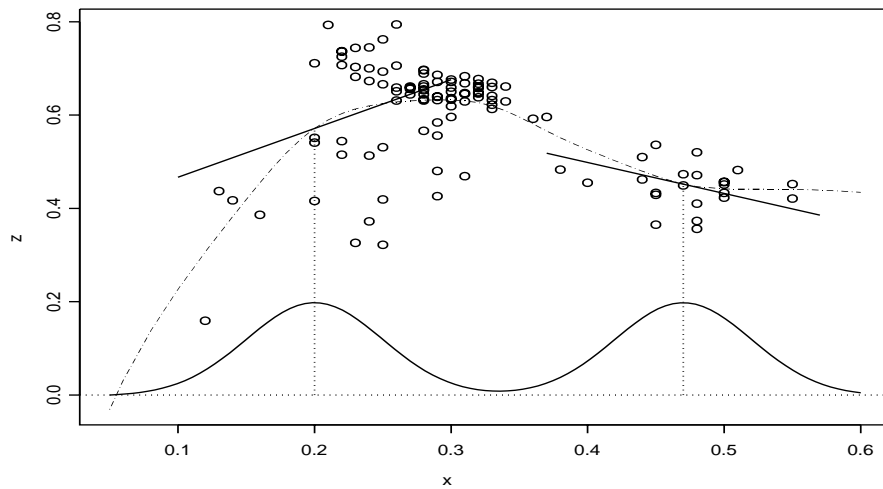


Figure 1: Illustration of local linear regression. Curves at the bottom of the graph show kernel weights; solid line represents the locally weighted least squares regression line at a point; dashed line is the fitted curve.

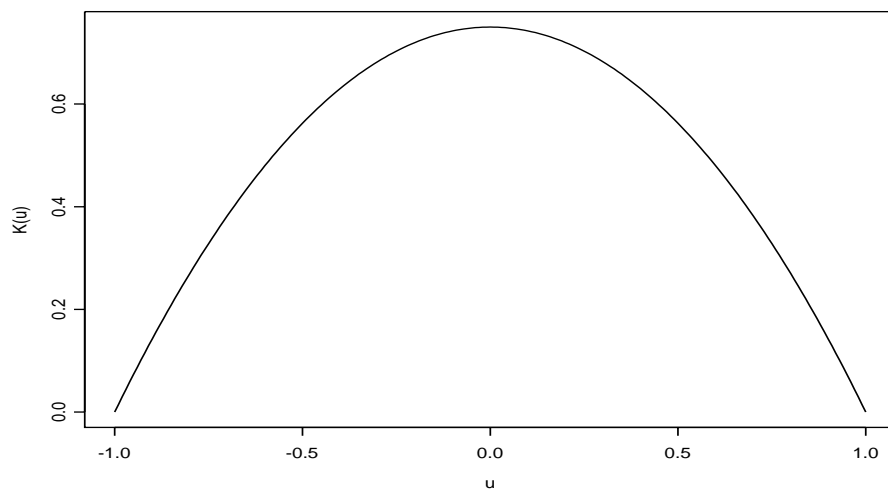


Figure 2: Epanechnikov kernel function, $\frac{3}{4}(1 - u^2)I_{\{|u|<1\}}$.

h is, the smoother the fitted curve. For very large h , kernel regression approaches standard least squares regression. With the Epanechnikov kernel, only points within $\pm h$ of x_i get any weight. In d dimensions, weights still come from a kernel function, but here the kernel is a d -variate function. This kernel may be the product of d one-dimensional kernels, such as the Epanechnikov. The multivariate standard normal is another popular choice. In this case, the scalar bandwidth, h , is replaced by a symmetric, positive definite matrix, \mathbf{H} . When \mathbf{H} is a diagonal matrix, h_i is the bandwidth in the i th direction. Non-diagonal matrices change the orientation of the smoothing weights.

Smoothing in more than one dimension is subject to the curse of dimensionality. In higher dimensions, the data cloud may become sparse. This makes accurate smoothing very difficult. For this reason, the focus here will be on $d = 2$. This case is of special interest because of its direct application to spatial sampling.

2.4 Two-Stage Estimator for Spatial Sampling

2.4.1 Mean

Using local polynomial regression, the form of the estimator of a population mean for the two dimensional case is the basic model assisted estimator, where the model based prediction of $\hat{\mu}_i$ is

$$\hat{\mu}_i = \mathbf{e}'_1 (\mathbf{X}'_{si} \mathbf{W}_{si} \mathbf{X}_{si})^{-1} \mathbf{X}'_{si} \mathbf{W}_{si} \hat{\mathbf{z}}_s = \mathbf{w}'_{si} \hat{\mathbf{z}}_s$$

Here, $\hat{\mathbf{z}}_s$ is the vector of estimated cluster means. The local design matrix, \mathbf{X}_{si} , is defined as

$$\mathbf{X}_{si} = \left[\begin{array}{ccccccc} 1 & x_j - x_i & \cdots & (x_j - x_i)^p & y_j - y_i & \cdots & (y_j - y_i)^q \end{array} \right]_{j \in s},$$

where p and q are the degrees of the local polynomials being fitted in the x and y directions, respectively. The local weighting matrix is

$$\mathbf{W}_{si} = \text{Diag} \left\{ \frac{1}{\pi_j h_x h_y} K \left(\frac{x_j - x_i}{h_x}, \frac{y_j - y_i}{h_y} \right) \right\}_{j \in s}.$$

Here, K is a two-dimensional kernel function. The bandwidth in the x direction is h_x , while h_y is the bandwidth in the y direction. This form assumes a diagonal matrix, \mathbf{H} . If \mathbf{H} is not diagonal, then the expression above becomes

$$\mathbf{W}_{si} = \text{Diag} \left\{ \frac{1}{\pi_j |\mathbf{H}|} K \left(\mathbf{H}^{-1} \mathbf{x}_{ij} \right) \right\}_{j \in s},$$

where

$$\mathbf{x}_{ij} = \begin{bmatrix} x_j - x_i \\ y_j - y_i \end{bmatrix}.$$

The estimator of θ_z can now be expressed as

$$\hat{\theta}_z = \frac{1}{M} \left\{ \sum_{i \in C} \hat{\mu}_i + \sum_{i \in s} \frac{\hat{z}_i - \hat{\mu}_i}{\pi_i} \right\}.$$

2.4.2 Total

The form of the estimator of a population total is similar to that of the population mean:

$$\tilde{t}_i = \mathbf{e}'_1 (\mathbf{X}'_{si} \mathbf{W}_{si} \mathbf{X}_{si})^{-1} \mathbf{X}'_{si} \mathbf{W}_{si} \hat{\mathbf{t}}_s = \mathbf{w}'_{si} \hat{\mathbf{t}}_s$$

Here, $\hat{\mathbf{t}}_s$ is the vector of estimated cluster totals, obtained using the Horvitz-Thompson estimator. The estimator of the population total then becomes

$$\tilde{t}_z = \sum_{i \in C} \tilde{t}_i + \sum_{i \in s} \frac{\hat{t}_i - \tilde{t}_i}{\pi_i}.$$

2.4.3 Weights

The nonparametric regression estimator can be expressed as a linear combination of the study variables, with weights which do not depend on the study variables. These weights are extremely useful in practice. From earlier, note that

$$\begin{aligned} \tilde{t}_z &= \sum_{i \in s} \left\{ \frac{1}{\pi_i} + \sum_{j \in C} \left(1 - \frac{I_j}{\pi_j} \right) \mathbf{w}'_{sj} \mathbf{e}_i \right\} \hat{t}_i \\ &= \sum_{i \in s} \omega_{is} \hat{t}_i \\ &= \sum_{i \in s} \sum_{k \in s_i} \frac{\omega_{is}}{\pi_{k|i}} z_k. \end{aligned}$$

Thus, \tilde{t}_y is a linear combination of the \hat{t}_i 's in s , with cluster weights $\{\omega_{is}\}$ that are the sampling weights of clusters, suitably modified to reflect auxiliary information $[\mathbf{x}_i]_{i \in C}$. Alternatively, \tilde{t}_y is a linear combination of the z_k 's in $\cup_{i \in s} s_i$, with element weights $\{\omega_{is} \pi_{k|i}^{-1}\}$ which reflect both the design and the auxiliary information. Because both sets of weights are independent of the study variables, they can be applied to any study variable of interest.

The same is true for $\hat{\theta}_z$. In this case,

$$\begin{aligned} \hat{\theta}_z &= \frac{1}{M} \sum_{i \in s} \left\{ \frac{1}{\pi_i} + \sum_{j \in C} \left(1 - \frac{I_j}{\pi_j} \right) \mathbf{w}'_{sj} \mathbf{e}_i \right\} \bar{z}_{si} \\ &= \frac{1}{M} \sum_{i \in s} \omega_{is} \bar{z}_{si} \\ &= \frac{1}{M} \sum_{i \in s} \sum_{k \in s_i} \frac{\omega_{is}}{\pi_{k|i}} z_k. \end{aligned}$$

Again, these ω weights can be applied to any variable of interest.

2.4.4 Bandwidth Selection

Bandwidth selection is important in implementing nonparametric regression estimation. If the bandwidth is too large, the nonparametric regression estimate will approach the

regression estimate, and all efficiency gains will be lost for non-linear populations. If the bandwidth is too small, the smooth will fluctuate from point to point and vary too much from one sample to the next. This under-smoothing can result in variance estimates that have a large negative bias. To implement nonparametric regression estimation effectively, one should select a bandwidth that gives an appropriate amount of smoothing. Often, one of the best ways to do this is by eye. Vary the bandwidth until a smooth curve is obtained which follows the general trends of the data, without jumping from point to point. Another rule of thumb is to select a bandwidth that covers 1/4 of the range of the auxiliary variables.

2.5 Variance Estimation

Variance approximations for a finite second stage population follow. Define γ_i as the smoothed estimate of the i th cluster total when the entire population is known. Then $D_i = t_i - \gamma_i$, the residual of the i th cluster for this population level smooth. Also define $\Delta_{kl} = \pi_{kl} - \pi_k \pi_l$. Then the asymptotic variance of the local polynomial regression estimator of a population total is

$$AV(\hat{t}) = AV_{PSU} + AV_{SSU}$$

where

$$AV_{PSU} = \sum_{i \in C} \sum_{j \in C} \Delta_{ij} \frac{D_i D_j}{\pi_i \pi_j}$$

and

$$AV_{SSU} = \sum_{i \in C} \frac{V_i}{\pi_i}.$$

Here,

$$V_i = \sum_{k \in U_i} \sum_{l \in U_i} \Delta_{kl|i} \frac{z_k}{\pi_{k|i}} \frac{z_l}{\pi_{l|i}}.$$

The estimated variance of \hat{t} is then

$$\hat{V}(\hat{t}) = \hat{V}_{PSU} + \hat{V}_{SSU}$$

where

$$\hat{V}_{PSU} = \sum_{i \in s} \sum_{j \in s} \frac{\Delta_{ij}}{\pi_{ij}} \omega_i d_i \omega_j d_j - \sum_{i \in s} (1 - \pi_i) \omega_i^2 \hat{V}_i$$

and

$$\hat{V}_{SSU} = \sum_{i \in s} \omega_i^2 \hat{V}_i.$$

Also, define

$$d_i = \hat{t}_i - \tilde{t}_i$$

and

$$\hat{V}_i = \sum_{k \in s_i} \sum_{l \in s_i} \frac{\Delta_{kl|i}}{\pi_{kl|i}} \frac{z_k}{\pi_{k|i}} \frac{z_l}{\pi_{l|i}}.$$

Recall that ω_i is the weight for the i th sampled cluster in the weighted form of \hat{t}_z .

The calculations are similar when estimating θ_z . First, define $\bar{z}_i = \frac{1}{N_i} \sum_{m \in U_i} z_m$ and μ_i as the local polynomial regression estimate of the i th cluster mean based on the entire population. With $D_i = \bar{z}_i - \mu_i$, the asymptotic variance of this estimator is

$$\text{AV}(\hat{\theta}_z) = \frac{1}{M^2} \left[\sum_{i \in C} \sum_{j \in C} \Delta_{ij} \frac{D_i D_j}{\pi_i \pi_j} + \sum_{i \in C} \frac{V_i}{\pi_i} \right],$$

where

$$V_i = \frac{1}{N_i^2} \sum_{k \in U_i} \sum_{l \in U_i} \Delta_{kl|i} \left(\frac{z_k - \bar{z}_i}{\pi_{k|i}} \right) \left(\frac{z_l - \bar{z}_i}{\pi_{l|i}} \right).$$

Here, define $d_i = \hat{z}_i - \hat{\mu}_i$. Then, the estimator of the variance is

$$\hat{\text{AV}}(\hat{\theta}_z) = \frac{1}{M^2} \left[\sum_{i \in s} \sum_{j \in s} \frac{\Delta_{ij}}{\pi_{ij}} \omega_i d_i \omega_j d_j - \sum_{i \in s} (1 - \pi_i) \omega_i^2 \hat{V}_i + \sum_{i \in s} \omega_i^2 \hat{V}_i \right]$$

where

$$\hat{V}_i = \frac{1}{\hat{N}_i} \sum_{k \in s_i} \sum_{l \in s_i} \frac{\Delta_{kl|i}}{\pi_{kl|i}} \frac{(z_k - \hat{z}_i)}{\pi_{k|i}} \frac{(z_l - \hat{z}_i)}{\pi_{l|i}}$$

and $\hat{N}_i = \sum_{i \in s_i} \pi_i^{-1}$.

2.5.1 Variance Estimation with an Infinite Second Stage Population

Variance estimation is slightly more complicated when the second stage population is infinite. Here, the variance of the nonparametric regression estimator is approximated for that situation. Define

$$\begin{aligned} \hat{\xi}_i &= \frac{\hat{z}_i - \hat{\mu}_i}{\pi_i/m} \\ \text{E}[\hat{\xi}_i | s] &= \frac{\theta_i - \mu_i}{\pi_i/m} \\ &= \xi_i \\ \text{Var}[\hat{\xi}_i | s] &= \frac{\sigma_i^2/n_i}{(\pi_i/m)^2} \\ \hat{\bar{\xi}}_i &= \frac{1}{m} \sum_{i \in s} \frac{\hat{z}_i - \hat{\mu}_i}{\pi_i/m} \\ \text{E}[\hat{\bar{\xi}}_i | s] &= \frac{1}{m} \sum_{i \in s} \xi_i \\ &= \bar{\xi} \\ \text{Var}[\hat{\bar{\xi}}_i | s] &= \frac{1}{m^2} \sum_{i \in s} \frac{\sigma_i^2/n_i}{(\pi_i/m)^2} \end{aligned}$$

$$\begin{aligned}
E \left[\sum_{i \in s} (\hat{\xi}_i - \hat{\xi})^2 | s \right] &= E \left[\sum_{i \in s} \hat{\xi}_i^2 - m \hat{\xi}^2 | s \right] \\
&= \sum_{i \in s} \left(\xi_i^2 + \frac{\sigma_i^2/n_i}{(\pi_i/m)^2} \right) - m \left(\bar{\xi}^2 + \frac{1}{m^2} \sum_{i \in s} \frac{\sigma_i^2/n_i}{(\pi_i/m)^2} \right) \\
&= \sum_{i \in s} \xi_i^2 - m \bar{\xi}^2 + \sum_{i \in s} \frac{\sigma_i^2/n_i}{(\pi_i/m)^2} \left(1 - \frac{1}{m} \right).
\end{aligned}$$

Note that μ_i are the smoothed estimates of the i th cluster mean when the entire population is known and σ_i^2 is the true variance within the i th cluster.

Assume ξ_i are independent and identically distributed. Then

$$\begin{aligned}
P \left[\xi_i = \frac{\theta_k - \mu_k}{\pi_k/m} \right] &= \frac{\pi_k}{m} \\
E[\xi_i] &= \sum_{k \in C} \frac{\theta_k - \mu_k}{\pi_k/m} \frac{\pi_k}{m} \\
&= \sum_{k \in C} \theta_k - \mu_k \\
&= \delta \\
\text{Var}[\xi_i] &= \sum_{k \in C} \left(\frac{\theta_k - \mu_k}{\pi_k/m} - \delta \right)^2 \frac{\pi_k}{m} \\
&= \tau^2.
\end{aligned}$$

An unbiased estimator of the variance of ξ_i would be

$$\frac{\sum_{i \in s} (\xi_i - \bar{\xi})^2}{m-1}.$$

This implies that

$$\begin{aligned}
E[\hat{\theta}_z | s] &= E \left[\frac{1}{M} \left(\sum_{i \in C} \hat{\mu}_i + \frac{1}{m} \sum_{i \in s} \frac{\hat{z}_i - \hat{\mu}_i}{\pi_i/m} \right) | s \right] \\
&= \frac{1}{M} \sum_{i \in C} \mu_i + \frac{1}{M} \sum_{i \in s} \frac{\xi_i}{m} \\
\text{Var} \left(E[\hat{\theta}_z | s] \right) &= \frac{\tau^2}{M^2 m} \\
\text{Var} \left(\hat{\theta}_z | s \right) &= \frac{1}{m^2 M^2} \sum_{i \in s} \frac{\sigma_i^2/n_i}{(\pi_i/m)^2} \\
E[\text{Var} \left(\hat{\theta}_z | s \right)] &= \frac{1}{m M^2} \sum_{i \in C} \frac{\sigma_i^2/n_i}{\pi_i/m}
\end{aligned}$$

$$\begin{aligned}
\text{Var}(\hat{\theta}_z) &= \text{Var}(\mathbb{E}[\hat{\theta}_z|s]) + \mathbb{E}[\text{Var}(\hat{\theta}_z|s)] \\
&= \frac{\tau^2}{M^2m} + \frac{1}{M^2} \sum_{i \in C} \frac{\sigma_i^2/n_i}{\pi_i}.
\end{aligned}$$

Assuming with replacement sampling, an unbiased estimator of $\text{Var}(\hat{\theta}_z)$ is:

$$\hat{V}_1 = \frac{1}{M^2m} \sum_{i \in s} \frac{(\hat{\xi}_i - \hat{\xi})^2}{m-1} + \frac{m-1}{M^2m^2} \sum_{i \in s} \frac{s_i^2/n_i}{(\pi_i/m)^2}.$$

Note that s_i^2 is the sample variance of the i th cluster.

A simpler estimator of $\text{Var}(\hat{\theta}_z)$ is:

$$\begin{aligned}
\hat{V}_0 &= \frac{1}{M^2m} \sum_{i \in s} \frac{(\hat{\xi}_i - \hat{\xi})^2}{m-1}. \\
\mathbb{E}[\hat{V}_0|s] &= \frac{1}{M^2m(m-1)} \left(\sum_{i \in s} (\xi_i - \bar{\xi})^2 + \sum_{i \in s} \frac{\sigma_i^2/n_i}{(\pi_i/m)^2} \left(1 - \frac{1}{m}\right) \right) \\
\mathbb{E}[\hat{V}_0] &= \frac{\tau^2}{M^2m} + \frac{1}{M^2m} \sum_{i \in C} \frac{\sigma_i^2/n_i}{\pi_i}
\end{aligned}$$

This estimator is negatively biased.

When estimating the population total, the notation is slightly different. Here, define γ_i as the smoothed estimate of the i th cluster total based on the entire population. Also,

$$\begin{aligned}
\hat{\xi}_{ti} &= \frac{\hat{t}_i - \tilde{t}_i}{\pi_i/m} \\
\mathbb{E}[\hat{\xi}_{ti}|s] &= \frac{t_i - \gamma_i}{\pi_i/m} \\
&= \xi_{ti} \\
\mathbb{E}[\hat{\xi}_{ti}] &= \sum_{k \in C} t_k - \gamma_k \\
&= \delta_t \\
\text{Var}[\hat{\xi}_{ti}|s] &= \frac{n_i \sigma_i^2}{(\pi_i/m)^2} \\
\text{Var}[\hat{\xi}_{ti}] &= \sum_{k \in C} \left(\frac{t_k - \gamma_k}{\pi_k/m} - \delta_t \right)^2 \frac{\pi_k}{m} \\
&= \tau_t^2
\end{aligned}$$

Calculations analogous to those in the preceding section yield

$$\text{Var}(\hat{t}) = \frac{\tau_t^2}{m} + \sum_{i \in C} \frac{n_i \sigma_i^2}{\pi_i}$$

and estimators

$$\begin{aligned}\hat{V}_{t1} &= \frac{1}{m} \sum_{i \in s} \frac{(\hat{\xi}_{ti} - \hat{\xi}_t)^2}{m-1} + (m-1) \sum_{i \in s} \frac{s_i^2 n_i}{(\pi_i/m)^2} \\ \hat{V}_{t0} &= \frac{1}{m} \sum_{i \in s} \frac{(\hat{\xi}_{ti} - \hat{\xi}_t)^2}{m-1}.\end{aligned}$$

The variance estimator V_{t1} is unbiased, while V_{t0} is negatively biased.

3 Simulation

We performed simulations to examine the performance of the local polynomial regression estimator, and to compare it to that of other estimators. The estimators used are the Horvitz-Thompson (HT), linear regression (REG) as in Särndal, Swensson, and Wretman (1992), and local polynomial regression, with $p = q = 0$ (LPR0) and $p = q = 1$ (LPR).

In this simulation, we sample from a continuous, spatially correlated surface, or from a noisy version of this surface. The surface is considered fixed, as we are looking at properties of estimators under repeated sampling of a population. Note that it is not possible to compute and store a continuous surface in a simulation study without first parameterizing it. In order to create the continuous surface, we laid down an 11×11 grid of points over the unit square. The response at each of these grid points is a deterministic trend plus a spatially correlated stochastic process:

$$g(x_i, y_i) = \mu(x_i, y_i) + G(x_i, y_i),$$

where

$$\text{Cov}(G(x_i, y_i), G(x_j, y_j)) = \sigma^2 \exp \left\{ \log(\rho) \left((x_i - x_j)^2 + (y_i - y_j)^2 \right)^{\frac{1}{2}} \right\}.$$

and the deterministic trend used was the plane

$$\mu(i) = \beta_0 + \beta_x x_i + \beta_y y_i.$$

Between the grid points, an interpolating thin plate spline was used to create a continuous surface. Aldworth and Cressie did a similar study where they created a grid of points with a planar trend and spatially correlated stochastic process (1999). However, in their study, rather than using a spline to create a continuum, they used the grid points as the population of interest. Thin plate splines, the extension of natural cubic splines to two dimensions, use roughness penalties to create a smooth, twice-continuously differentiable surface (Green and Silverman, 1994). The interpolating spline takes the form

$$g(\mathbf{t}_i) = \sum_{j=1}^n \delta_j \eta(\|\mathbf{t}_i - \mathbf{t}_j\|) + \sum_{k=1}^3 a_k \phi_k(\mathbf{t}_i)$$

where

$$\begin{aligned}\eta(\|\mathbf{t}_i - \mathbf{t}_j\|) &= \frac{1}{16\pi} \|\mathbf{t}_i - \mathbf{t}_j\|^2 \log \|\mathbf{t}_i - \mathbf{t}_j\|^2 \\ \phi_1(x, y) &= 1 \\ \phi_2(x, y) &= x \\ \phi_3(x, y) &= y.\end{aligned}$$

The $3 \times n$ matrix \mathbf{T} is defined $\mathbf{T}_{ij} = \phi_i(\mathbf{t}_j)$, i.e.

$$\mathbf{T} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \mathbf{t}_1 & \mathbf{t}_2 & \cdots & \mathbf{t}_n \end{bmatrix}.$$

The vector $\mathbf{t}_i = (x_i, y_i)'$, and the matrix \mathbf{E} is a $n \times n$ matrix with the ij th entry

$$E_{ij} = \eta(\|\mathbf{t}_i - \mathbf{t}_j\|).$$

The values of δ and \mathbf{a} are found by solving

$$\begin{bmatrix} \mathbf{E} & \mathbf{T}' \\ \mathbf{T} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \delta \\ \mathbf{a} \end{pmatrix} = \begin{pmatrix} \mathbf{z} \\ \mathbf{0} \end{pmatrix}$$

where \mathbf{z} is the vector of responses on the 11×11 grid.

The simulation was run for 48 different parameter settings. The sample size was either $m = 100$ or $m = 200$. Two different deterministic trends were considered: $\beta_0 = 0.5, \beta_x = \beta_y = 1$ and $\beta_0 = 0.5, \beta_x = \beta_y = 0$. We also varied the correlation between the points, ρ . Runs were completed with values of $\rho = 0.3487, 9.7656e - 4$, and 0, which gave correlations of 0.9, 0.5, and 0, respectively, between points that were one grid unit apart (a distance of 0.1). A correlation of zero adds random noise to the grid points, while those of 0.5 and 0.9 create stochastic trends of varying smoothness. The observed z values were then

$$z(x_i, y_i) = \beta_0 + \beta_x x_i + \beta_y y_i + G(x_i, y_i) + \epsilon_i$$

where ϵ_i are independent, identically distributed normal random variables with $\text{Var}(\epsilon_i) = \nu^2$, and with $G(x_i, y_i)$ the spatially correlated stochastic process defined above. Both the case where $\nu^2 = 0$ and that when $\nu^2 > 0$ were considered. We chose different values of σ^2 so that the spatially correlated stochastic process accounted for 5% and 15% of the total variation in the response values at the grid points in the case with planar trend. That is, for $\text{Var}(G(x_i, y_i)) = \sigma^2$,

$$\frac{\sigma^2}{\sigma^2 + \int_0^1 \int_0^1 (\mu(x, y))^2 dx dy - \left(\int_0^1 \int_0^1 \mu(x, y) dx dy \right)^2 + \nu^2} = 0.05 \text{ or } 0.15$$

The calculations of total variability were done for the planar response, and the same values of the variance parameters were used for the flat response. These two levels of variance

were chosen to create an overall smooth that is nearly deterministic, and one that has a more substantial stochastic component. In the case of $\nu^2 = 0$, $\sigma^2 = 0.00877$ for the 5% case and 0.02632 for 15%. With $\nu^2 > 0$ and 5% of the total variation coming from the noise, these values of σ^2 become 0.008795 and 0.03214. The noise variance, ν^2 , is 0.008795 with $\sigma^2 = .008795$ and $\nu^2 = 0.01033$ when $\sigma^2 = .03214$.

For one run of the simulation at a particular setting, the 11×11 grid of points was generated. The spline was created, and the volume under the spline calculated by numerical integration. This value was later used in bias calculations. Then, for each replication, we took a simple random sample over the unit square. The interpolating spline was evaluated at each sampled point.

For each sample, the four estimates and their estimated variances were calculated. A sample was taken 100 times for each setting. Then, for each estimator, the average estimate, the average estimated variance, the variance of the estimates, the percent bias of the estimator, the relative efficiency of the estimator, as compared to the LPR estimator, and the percent bias of the variance estimate were calculated. For the local polynomial regression estimators, we used the product Epanechnikov kernel function with $h = 0.35$ in both directions. This corresponds to approximately 1/8 of the total surface being included in each smooth. Because the response surface is continuous, an estimate of the smooth could not be made for every population element. Rather, the smooth was evaluated over a 15×15 grid, and then a piecewise constant interpolation was used to extend the smooth over the entire sampling space.

From the simulation results given in Tables 18, it can be seen that the local polynomial regression estimator performs well, especially in the cases where the parametric models are misspecified. Tables 1, 2, 5, and 6 show the results for data with a planar trend. In these cases, the regression estimator dominates all others. It should be noted that, for these parameter settings, the LPR estimator does best in the case with a correlation of 0.5. This makes sense, since when $\rho = 0$, the truth is close to a plane with random noise, and for the highest value of ρ , the high correlation forces a flatter response surface. The 0.5 correlation creates a locally smooth surface with peaks and valleys. Additionally, the LPR's performance relative to the regression estimator improves with the introduction of random error. Tables 3, 4, 7, and 8 give the results for the simulation runs with no plane. Here, the LPR0 estimator outperforms all other estimators except when there is no correlation in the data, and, even then, is competitive and is sometimes the most efficient estimator. Here, the response surface is a smoothed spatially correlated stochastic process, a situation in which local polynomial regression would be expected to perform well. And because the underlying structure is flat, not planar, regression of degree zero not surprisingly outperforms local planar regression. It should also be noted that in limited simulations with higher levels of variance in the stochastic trend, the LPR estimator dominated the other estimator for all values of the trend and ρ . The relative efficiencies of the HT and REG estimators as compared to the LPR estimator are plotted in figures 3 and 4 for a sample size of $m = 100$.

From these simulation results, it is apparent that there is a bias problem with the variance estimation of the LPR estimators. For almost every parameter setting, both

Settings	Estimator	Estimate	Var(Est)	Est.Var.	%Bias	Efficiency	%Bias Var.
$\rho = 0$	HT	1.5064	0.000686	0.000859	0.546	16.092	5.281
PSSV=5%	REG	1.4987	0.000021	0.000020	0.032	0.493	-3.810
Noise=0%	LPR	1.4992	0.000043	0.000016	0.065	1.000	-61.502
True=1.4982	LPR0	1.5004	0.000067	0.000051	0.147	1.577	-24.851
$\rho = 0.5$	HT	1.5083	0.000729	0.000895	0.549	19.703	22.771
PSSV=5%	REG	1.5005	1.58E-05	1.75E-05	0.025	0.427	10.759
Noise=0%	LPR	1.4995	0.000043	9.98E-06	-0.041	1.000	-73.015
True=1.4982	LPR0	1.50128	6.51E-05	4.39E-05	0.079	1.759	-32.606
$\rho = 0.9$	HT	1.4708	8.28E-04	9.94E-04	0.585	26.938	19.995
PSSV=5%	REG	1.4623	3.90E-06	4.47E-06	0.009	0.127	14.662
Noise=0%	LPR	1.4624	3.08E-05	2.04E-06	0.012	1.000	-93.362
True=1.4622	LPR0	1.4640	6.22E-05	3.59E-05	0.125	2.022	-42.185
$\rho = 0$	HT	1.5053	6.96E-04	8.98E-04	0.565	8.423	29.121
PSSV=15%	REG	1.4977	6.30E-05	6.05E-05	0.055	0.7627	-4.030
Noise=0%	LPR	1.4982	8.26E-05	4.91E-05	0.088	1.000	-40.533
True=1.4969	LPR0	1.4991	1.02E-04	8.86E-05	0.149	1.237	-13.251
$\rho = 0.5$	HT	1.5087	7.66E-04	9.60E-04	0.570	12.543	25.348
PSSV=15%	REG	1.5008	4.73E-05	5.26E-05	0.044	0.774	11.409
Noise=0%	LPR	1.4987	6.11E-05	3.00E-05	-0.097	1.000	-50.938
True=1.5002	LPR0	1.5006	8.84E-05	6.95E-05	0.030	1.448	-21.423
$\rho = 0.9$	HT	1.4436	9.28E-04	1.12E-03	0.634	24.692	20.698
PSSV=15%	REG	1.4348	1.17E-05	1.34E-05	0.015	0.311	14.665
Noise=0%	LPR	1.4345	3.76E-05	6.12E-06	-0.006	1.000	-83.711
True=1.4345	LPR0	1.4361	7.05E-05	4.38E-05	0.108	1.877	-37.869

Table 1: For all settings, 100 replications were run with a 11×11 population grid, and a 15×15 grid for the interpolation within the LPR estimator. PSSV is the proportion of small-scale variation. This table presents the results for $m = 200$, $\mu_i = .5 + x_i + y_i$, and no noise.

Settings	Estimator	Estimate	Var(Est)	Est.Var.	%Bias	Efficiency	%Bias Var.
$\rho = 0$	HT	1.5011	7.67E-04	9.02E-04	0.197	9.209	17.687
PSSV=5%	REG	1.4989	5.78E-05	6.35E-05	0.0492	0.694	9.913
Noise=5%	LPR	1.4988	8.32E-05	5.71E-05	0.041	1.000	-31.407
True=1.4982	LPR0	1.4988	1.00E-04	9.29E-05	0.038	1.204	-7.354
$\rho = 0.5$	HT	1.5030	7.96E-04	9.37E-04	0.195	9.424	17.773
PSSV=5%	REG	1.5008	6.54E-05	6.07E-05	0.045	0.774	-7.067
Noise=5%	LPR	1.4991	8.45E-05	5.05E-05	-0.069	1.000	-40.223
True=1.5000	LPR0	1.4996	1.01E-04	8.61E-05	-0.030	1.196	-14.814
$\rho = 0.9$	HT	1.4650	8.95E-04	1.04E-03	0.196	11.736	15.896
PSSV=5%	REG	1.4626	4.78E-05	4.76E-05	0.029	0.627	-0.547
Noise=5%	LPR	1.4620	7.62E-05	4.25E-05	-0.010	1.000	-44.196
True=1.4622	LPR0	1.4623	9.43E-05	7.81E-05	0.008	1.237	-17.213
$\rho = 0$	HT	1.5000	8.03E-04	9.62E-04	0.227	6.091	19.766
PSSV=15%	REG	1.4978	1.13E-04	1.23E-04	0.083	0.853	9.490
Noise=5%	LPR	1.4975	1.32E-04	1.07E-04	0.061	1.000	-19.180
True=1.4966	LPR0	1.4972	1.48E-04	1.49E-04	0.042	1.119	1.157
$\rho = 0.5$	HT	1.5035	8.64E-04	1.03E-03	0.2234	6.893	18.791
PSSV=15%	REG	1.503	1.31E-04	1.13E-04	0.075	1.043	-13.262
Noise=5%	LPR	1.4980	1.25E-04	8.31E-05	-0.1478	1.000	-33.693
True=1.5002	LPR0	1.4989	1.44E-04	1.26E-04	-0.086	1.145	-12.126
$\rho = 0.9$	HT	1.4319	1.03E-03	1.20E-03	0.228	10.705	15.954
PSSV=15%	REG	1.4293	7.35E-05	6.65E-05	0.046	0.761	-9.423
Noise=5%	LPR	1.4282	9.66E-05	5.48E-05	-0.037	1.000	-43.280
True=1.4287	LPR0	1.4285	1.16E-04	9.53E-05	-0.015	1.200	-17.805

Table 2: For all settings, 100 replications were run with a 11×11 population grid, and a 15×15 grid for the interpolation within the LPR estimator. PSSV is the proportion of small-scale variation. This table presents the results for $m = 200$, $\mu_i = .5 + x_i + y_i$, and 5% noise.

Settings	Estimator	Estimate	Var(Est)	Est.Var.	%Bias	Efficiency	%Bias Var.
$\rho = 0$	HT	0.4986	2.03E-05	2.09E-05	0.076	0.935	2.678
PSSV=5%	REG	0.4987	2.10E-05	2.02E-05	0.096	0.967	-4.030
Noise=0%	LPR	0.4987	2.17E-05	1.64E-05	0.091	1.000	-24.627
True=0.4982	LPR0	0.4982	2.08E-05	1.82E-05	0.003	0.955	-12.385
$\rho = 0.5$	HT	0.5005	1.56E-05	2.04E-05	0.087	1.268	30.702
PSSV=5%	REG	0.5005	1.58E-05	1.76E-05	0.077	1.283	11.410
Noise=0%	LPR	0.4989	1.23E-05	9.99E-06	-0.230	1.000	-18.659
True=0.5000	LPR0	0.4991	1.27E-05	1.21E-05	-0.203	1.036	-4.722
$\rho = 0.9$	HT	0.4629	8.43E-06	1.11E-05	0.161	3.553	31.839
PSSV=5%	REG	0.4623	3.90E-06	4.48E-06	0.027	1.647	14.667
Noise=0%	LPR	0.4618	2.37E-06	2.04E-06	-0.0768	1.000	-13.859
True=0.4622	LPR0	0.4618	2.67E-06	2.73E-06	-0.080	1.126	2.280
$\rho = 0$	HT	0.4975	6.09E-05	6.26E-05	0.131	0.935	2.678
PSSV=15%	REG	0.4977	6.30E-05	6.05E-05	0.166	0.967	-4.030
Noise=0%	LPR	0.4977	6.52E-05	4.91E-05	0.158	1.000	-24.628
True=0.4969	LPR0	0.4969	6.23E-05	5.46E-05	0.006	0.955	-12.385
$\rho = 0.5$	HT	0.501	4.67E-05	6.10E-05	0.163	1.268	30.702
PSSV=15%	REG	0.501	4.73E-05	5.26E-05	0.132	1.283	11.410
Noise=0%	LPR	0.498	3.68E-05	3.00E-05	-0.399	1.000	-18.659
True=0.5002	LPR0	0.498	3.82E-05	3.64E-05	-0.351	1.036	-4.722
$\rho = 0.9$	HT	0.43585	2.53E-05	3.33E-05	0.296	3.553	31.839
PSSV=15%	REG	0.4348	1.17E-05	1.34E-05	0.050	1.647	14.665
Noise=0%	LPR	0.4339	7.11E-06	6.12E-06	-0.141	1.000	-13.859
True=0.4345	LPR0	0.4339	8.00E-06	8.19E-06	-0.148	1.1264	2.281

Table 3: For all settings, 100 replications were run with a 11×11 population grid, and a 15×15 grid for the interpolation within the LPR estimator. PSSV is the proportion of small-scale variation. This table presents the results for $m = 200$, $\mu_i = .5$, and no noise.

Settings	Estimator	Estimate	Var(Est)	Est.Var.	%Bias	Efficiency	%Bias Var.
$\rho = 0$	HT	0.4989	5.57E-05	6.45E-05	0.148	0.922	15.780
PSSV=5%	REG	0.4989	5.78E-05	6.35E-05	0.148	0.958	9.912
Noise=5%	LPR	0.4988	6.04E-05	5.71E-05	0.117	1.000	-5.395
True=0.4982	LPR0	0.4984	5.58E-05	6.02E-05	0.033	0.924	8.025
$\rho = 0.5$	HT	0.5008	6.54E-05	6.37E-05	0.143	1.104	-2.680
PSSV=5%	REG	0.5008	6.54E-05	6.07E-05	0.135	1.102	-7.067
Noise=5%	LPR	0.4990	5.93E-05	5.05E-05	-0.215	1.000	-14.851
True=0.5001	LPR0	0.4992	5.57E-05	5.40E-05	-0.171	0.939	-3.076
$\rho = 0.9$	HT	0.4628	5.04E-05	5.46E-05	0.144	1.020	8.328
PSSV=5%	REG	0.4626	4.78E-05	4.76E-05	0.093	0.966	-0.547
Noise=5%	LPR	0.4620	4.95E-05	4.25E-05	-0.038	1.000	-14.001
True=0.4622	LPR0	0.4619	4.42E-05	4.45E-05	-0.061	0.893	0.842
$\rho = 0$	HT	0.4978	1.07E-04	1.26E-04	0.239	0.9834	17.521
PSSV=15%	REG	0.4978	1.13E-04	1.23E-04	0.250	1.034	9.488
Noise=5%	LPR	0.4975	1.09E-04	1.07E-04	0.176	1.000	-2.068
True=0.4966	LPR0	0.4968	1.04E-04	1.14E-04	0.045	0.955	10.182
$\rho = 0.5$	HT	0.5013	1.31E-04	1.23E-04	0.229	1.341	-6.447
PSSV=15%	REG	0.5013	1.31E-04	1.13E-04	0.225	1.334	-13.262
Noise=5%	LPR	0.4979	9.81E-05	8.31E-05	-0.449	1.000	-15.216
True=0.5002	LPR0	0.4985	9.83E-05	9.24E-05	-0.339	1.002	-5.987
$\rho = 0.9$	HT	0.4297	8.68E-05	9.06E-05	0.245	1.310	4.444
PSSV=15%	REG	0.4293	7.35E-05	6.65E-05	0.154	1.109	-9.423
Noise=15%	LPR	0.4281	6.62E-05	5.48E-05	-0.129	1.000	-17.298
True=0.4287	LPR0	0.4281	6.16E-05	5.88E-05	-0.143	0.930	-4.547

Table 4: For all settings, 100 replications were run with a 11×11 population grid, and a 15×15 grid for the interpolation within the LPR estimator. PSSV is the proportion of small-scale variation. This table presents the results for $m = 200$, $\mu_i = .5$, with 5% noise.

Settings	Estimator	Estimate	Var(Est)	Est.Var.	%Bias	Efficiency	%Bias Var.
$\rho = 0$	HT	1.505548	0.001504	0.001679	0.490	10.748	11.607
PSSV=5%	REG	1.49763	3.98E-05	4.09E-05	-0.038	0.284	2.781
Noise=0%	LPR	1.498212	0.000140	3.13E-05	0.001	1.000	-77.644
True=1.4982	LPR0	1.499236	0.000189	0.000100	0.069	1.349	-46.986
$\rho = 0.5$	HT	1.5082	0.001540	1.75E-03	0.538	11.686	13.551
PSSV=5%	REG	1.5002	3.43E-05	3.57E-05	0.006	0.260	3.976
Noise=0%	LPR	1.4989	0.000132	1.91E-05	-0.080	1.000	-85.470
True=1.5001	LPR0	1.5010	0.000181	8.75E-05	0.058	1.377	-51.771
$\rho = 0.9$	HT	1.4708	1.70E-03	1.95E-03	0.592	14.157	14.335
PSSV=5%	REG	1.4623	8.01E-06	9.13E-06	0.007	0.067	13.987
Noise=0%	LPR	1.4625	1.20E-04	3.93E-06	0.018	1.000	-96.739
True=1.4622	LPR0	1.4640	1.74E-04	7.33E-05	0.120	1.444	-57.854
$\rho = 0$	HT	1.5038	1.61E-03	1.75E-03	0.461	7.146	8.844
PSSV=15%	REG	1.4959	1.19E-04	1.23E-04	-0.066	0.530	2.781
Noise=0%	LPR	1.4961	2.25E-04	9.38E-05	-0.054	1.000	-58.294
True=1.4969	LPR0	1.4971	2.67E-03	1.74E-03	0.013	11.850	-34.764
$\rho = 0.5$	HT	1.5083	1.67E-03	1.87E-03	0.544	8.707	12.108
PSSV=15%	REG	1.5003	1.03E-04	1.07E-04	0.010	0.538	3.976
Noise=0%	LPR	1.4973	1.91E-04	5.74E-05	-0.193	1.000	-70.014
True=1.5002	LPR0	1.5001	2.39E-04	1.37E-04	-0.006	1.247	-42.574
$\rho = 0.9$	HT	1.4437	1.93E-03	2.19E-03	0.639	13.638	13.569
PSSV=15%	REG	1.4347	2.40E-05	2.74E-05	0.013	0.170	13.987
Noise=0%	LPR	1.4342	1.41E-04	1.18E-05	-0.026	1.000	-91.677
True=1.4345	LPR0	1.4360	2.02E-04	8.89E-05	0.101	1.425	-55.902

Table 5: For all settings, 100 replications were run with a 11×11 population grid, and a 15×15 grid for the interpolation within the LPR estimator. PSSV is the proportion of small-scale variation. This table presents the results for $m = 100$, $\mu_i = .5 + x_i + y_i$, and no noise.

Settings	Estimator	Estimate	Var(Est)	Est.Var.	%Bias	Efficiency	%Bias Var.
$\rho = 0$	HT	1.5023	1.93E-03	1.75E-03	0.274	7.770	-9.301
PSSV=5%	REG	1.5007	1.32E-04	1.27E-04	0.166	0.531	-3.531
Noise=5%	LPR	1.4987	2.48E-04	1.07E-04	0.0364	1.000	-56.932
True=1.4982	LPR0	1.4997	3.42E-04	1.84E-04	0.103	1.377	-46.237
$\rho = 0.5$	HT	1.5044	2.00E-03	1.82E-03	0.284	8.119	-9.086
PSSV=5%	REG	1.5028	1.37E-04	1.21E-04	0.184	0.557	-11.845
Noise=5%	LPR	1.4992	2.47E-04	9.49E-05	-0.058	1.000	-61.505
True=1.5001	LPR0	1.5011	3.41E-04	1.71E-04	0.066	1.381	-49.897
$\rho = 0.9$	HT	1.4662	2.18E-03	2.01E-03	0.279	9.387	-7.432
PSSV=5%	REG	1.4646	1.04E-04	9.46E-05	0.164	0.449	-9.085
Noise=5%	LPR	1.4621	2.32E-04	7.98E-05	-0.005	1.000	-65.591
True=1.4622	LPR0	1.4635	3.23E-04	1.56E-04	0.089	1.392	-51.679
$\rho = 0$	HT	1.5012	2.09E-03	1.86E-03	0.305	5.824	-10.937
PSSV=15%	REG	1.4995	2.38E-04	2.46E-04	0.193	0.662	3.470
Noise=5%	LPR	1.4972	3.59E-04	1.98E-04	0.042	1.000	-44.854
True=1.4966	LPR0	1.4982	4.65E-04	2.92E-04	0.108	1.293	-37.141
$\rho = 0.5$	HT	1.5059	2.35E-03	2.12E-03	0.330	7.269	-9.830
PSSV=15%	REG	1.5044	2.28E-04	2.08E-04	0.231	0.704	-8.501
Noise=5%	LPR	1.4983	3.23E-04	1.40E-04	-0.173	1.000	-56.582
True=1.5009	LPR0	1.5014	4.33E-04	2.31E-04	0.030	1.339	-46.645
$\rho = 0.9$	HT	1.4332	2.54E-03	2.33E-03	0.315	9.065	-8.371
PSSV=15%	REG	1.4314	1.45E-04	1.32E-04	0.192	0.516	-8.521
Noise=5%	LPR	1.4281	2.80E-04	1.03E-04	-0.038	1.000	-63.271
True=1.4287	LPR0	1.4298	3.87E-04	1.90E-04	0.08	1.381	-50.910

Table 6: For all settings, 100 replications were run with a 11×11 population grid, and a 15×15 grid for the interpolation within the LPR estimator. PSSV is the proportion of small-scale variation. This table presents the results for $m = 100, \mu_i = .5 + x_i + y_i$, and 5% noise.

Settings	Estimator	Estimate	Var(Est)	Est.Var.	%Bias	Efficiency	%Bias Var.
$\rho = 0$	HT	0.4976	3.97E-05	4.27E-05	-0.121	0.881	7.385
PSSV=5%	REG	0.4976	3.97E-05	4.09E-05	-0.114	0.881	2.781
Noise=0%	LPR	0.4971	4.51E-05	3.13E-05	-0.225	1.000	-30.694
True=0.4982	LPR0	0.4970	3.66E-05	3.58E-05	-0.232	0.810	-2.053
$\rho = 0.5$	HT	0.5002	3.79E-05	4.13E-05	0.024	1.2685	9.102
PSSV=5%	REG	0.5002	3.43E-05	3.57E-05	0.018	1.148	3.975
Noise=0%	LPR	0.4978	2.99E-05	1.91E-05	-0.466	1.000	-35.945
True=0.5001	LPR0	0.4988	2.46E-05	2.39E-05	-0.264	0.824	-2.737
$\rho = 0.9$	HT	0.4630	1.96E-05	2.15E-05	0.002	3.295	9.944
PSSV=5%	REG	0.4623	8.01E-06	9.13E-06	0.000	1.349	13.986
Noise=0%	LPR	0.4613	5.94E-06	3.92E-06	-0.002	1.000	-33.908
True=0.4622	LPR0	0.4618	5.76E-06	5.41E-06	-0.001	0.970	-6.015
$\rho = 0$	HT	0.4958	1.19E-04	1.28E-04	-0.209	0.881	7.385
PSSV=15%	REG	0.4959	1.19E-04	1.23E-04	-0.199	0.881	2.781
Noise=0%	LPR	0.4949	1.35E-04	9.38E-05	-0.3909	1.000	-30.694
True=0.4969	LPR0	0.4949	1.10E-04	1.07E-04	-0.402	0.810	-2.053
$\rho = 0.5$	HT	0.5004	1.14E-04	1.24E-04	0.042	1.268	9.102
PSSV=15%	REG	0.5003	1.03E-04	1.07E-04	0.031	1.148	3.976
Noise=0%	LPR	0.4961	8.96E-05	5.74E-05	-0.806	1.000	-35.945
True=0.5002	LPR0	0.4979	7.39E-05	7.18E-05	-0.457	0.824	-2.737
$\rho = 0.9$	HT	0.4358	5.87E-05	6.45E-05	0.280	3.295	9.944
PSSV=15%	REG	0.43478	2.40E-05	2.74E-05	0.041	1.349	13.987
Noise=0%	LPR	0.43308	1.78E-05	1.18E-05	-0.345	1.000	-33.908
True=0.4345	LPR0	0.4338	1.73E-05	1.62E-05	-0.171	0.970	-6.015

Table 7: For all settings, 100 replications were run with a 11×11 population grid, and a 15×15 grid for the interpolation within the LPR estimator. PSSV is the proportion of small-scale variation. This table presents the results for $m = 100$, $\mu_i = .5$, and no noise.

Settings	Estimator	Estimate	Var(Est)	Est.Var.	%Bias	Efficiency	%Bias Var.
$\rho = 0$	HT	0.5006	1.31E-04	1.30E-04	0.479	0.981	-0.759
PSSV=5%	REG	0.5007	1.32E-04	1.27E-04	0.499	0.985	-3.531
Noise=5%	LPR	0.5004	1.34E-04	1.07E-04	0.439	1.000	-20.149
True=0.4982	LPR0	0.5003	1.30E-04	1.17E-04	0.424	0.968	-9.8292
$\rho = 0.5$	HT	0.5026	1.40E-04	1.28E-04	0.509	1.090	-8.237
PSSV=5%	REG	0.5028	1.37E-04	1.21E-04	0.551	1.072	-11.845
Noise=5%	LPR	0.5009	1.28E-04	9.49E-05	0.154	1.000	-25.955
True=0.5001	LPR0	0.5016	1.29E-04	1.05E-04	0.3112	1.009	-19.007
$\rho = 0.9$	HT	0.4645	1.16E-04	1.09E-04	0.509	1.098	-6.554
PSSV=5%	REG	0.4646	1.04E-04	9.46E-05	0.520	0.981	-9.085
Noise=5%	LPR	0.4637	1.06E-04	7.98E-05	0.3391	1.000	-24.787
True=0.4622	LPR0	0.4640	1.05E-04	8.62E-05	0.403	0.990	-17.817
$\rho = 0$	HT	0.4994	2.42E-04	2.54E-04	0.571	1.007	5.111
PSSV=15%	REG	0.4995	2.38E-04	2.46E-04	0.582	0.991	3.470
Noise=5%	LPR	0.4989	2.40E-04	1.98E-04	0.457	1.000	-17.519
True=0.4966	LPR0	0.4988	2.32E-04	2.21E-04	0.440	0.963	-4.442
$\rho = 0.5$	HT	0.5041	2.57E-04	2.39E-04	0.644	1.330	-7.099
PSSV=15%	REG	0.5044	2.28E-04	2.08E-04	0.692	1.179	-8.501
Noise=5%	LPR	0.5000	1.93E-04	1.40E-04	-0.190	1.000	-27.301
True=0.5009	LPR0	0.5019	2.01E-04	1.61E-05	0.203	1.039	-91.963
$\rho = 0.9$	HT	0.4315	1.97E-04	1.80E-04	0.647	1.408	-8.782
PSSV=15%	REG	0.4314	1.45E-04	1.32E-04	0.638	1.034	-8.521
Noise=5%	LPR	0.4298	1.40E-04	1.03E-04	0.257	1.000	-26.339
True=0.4287	LPR0	0.4304	1.44E-04	1.14E-04	0.400	1.027	-20.625

Table 8: For all settings, 100 replications were run with a 11×11 population grid, and a 15×15 grid for the interpolation within the LPR estimator. PSSV is the proportion of small-scale variation. This table presents the results for $m = 100, \mu_i = .5$, and 5% noise.

Efficiency of LPR Relative to HT and REG

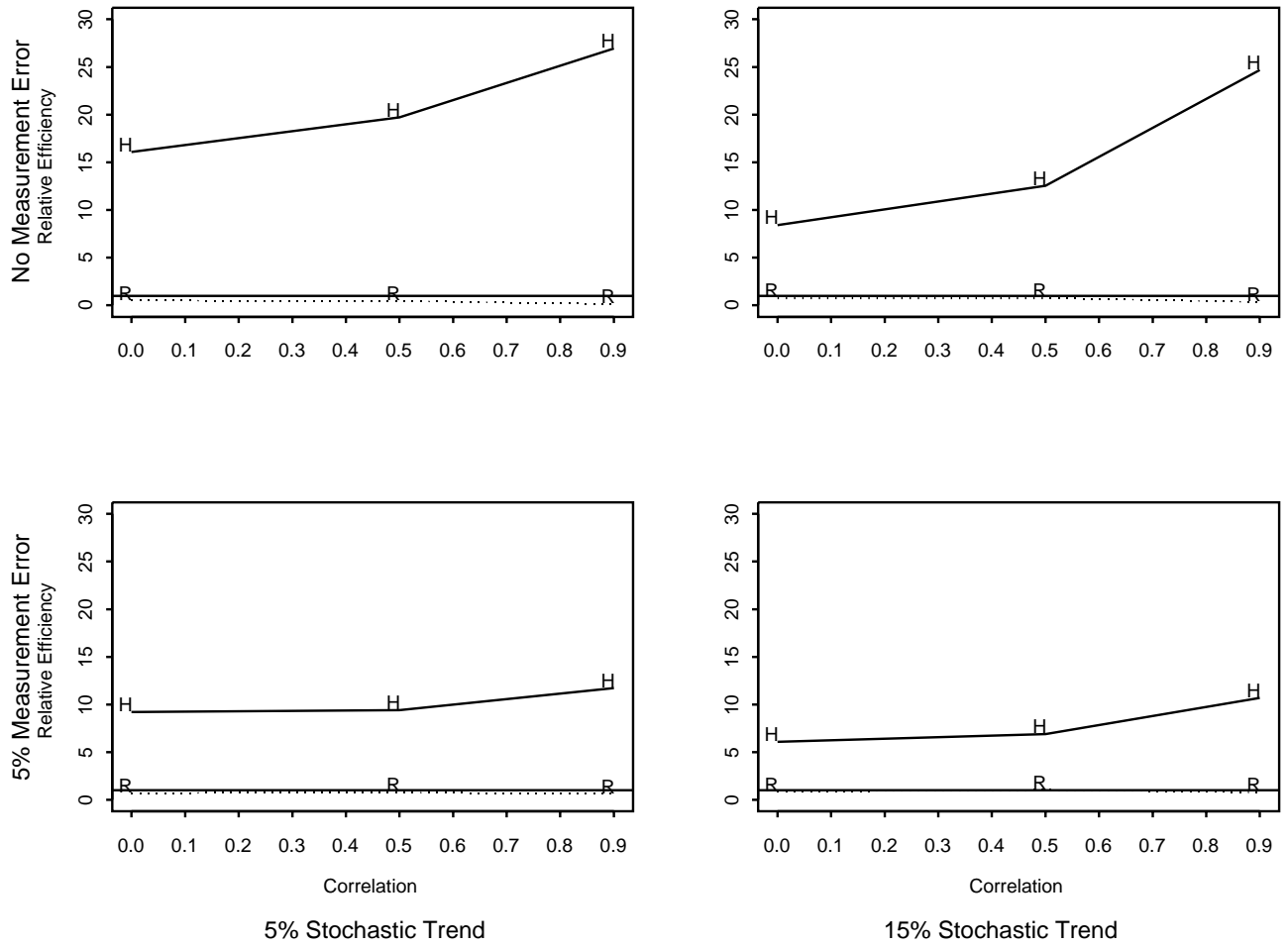


Figure 3: Simulation results for data with a planar trend.

Efficiency of LPR Relative to HT and REG

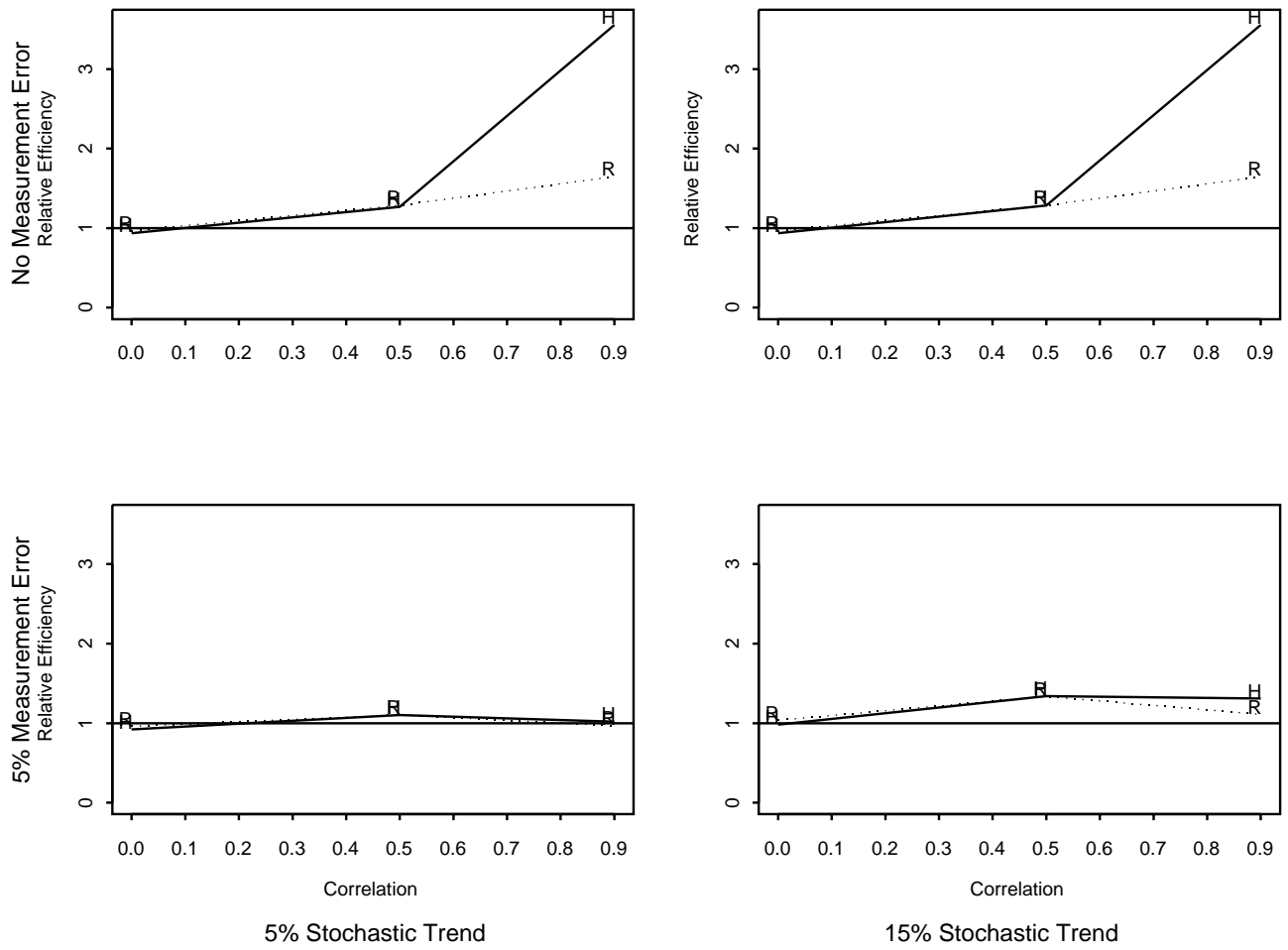


Figure 4: Simulation results for data with no planar trend.

Settings	Estimator	Estimate	Var(Est)	Est.Var.	%Bias	Efficiency	%Bias Var.
$\rho = 0$	HT	1.5055	1.50E-03	1.68E-03	0.490	11.024	11.608
PSSV=5%	REG	1.4976	3.97E-04	4.08E-05	-0.038	0.291	2.781
Noise=0%	LPR	1.4988	1.36E-04	4.07E-05	0.367	1.000	-70.206
True=1.4982	LPR0	1.5053	1.39E-03	1.54E-03	0.471	10.199	10.439
$\rho = 0.5$	HT	0.5033	2.66E-04	2.48E-04	0.628	1.081	-6.818
PSSV=15%	REG	0.5036	2.48E-04	2.26E-04	0.679	1.007	-8.792
Noise=5%	LPR	0.5034	2.46E-04	2.24E-04	0.642	1.000	-8.832
True=0.5002	LPR0	0.5033	2.64E-04	2.46E-04	0.631	1.074	-6.901
$\rho = 0.9$	HT	0.4358	5.87E-05	6.45E-05	0.280	2.31	9.944
PSSV=15%	REG	0.4347	2.40E-05	2.74E-05	0.041	0.946	13.987
Noise=0%	LPR	0.4348	2.54E-05	2.71E-05	0.059	1.000	6.917
True=0.4345	LPR0	0.4357	5.60E-05	6.13E-05	0.272	2.206	9.473
$\rho = 0$	HT	1.5023	1.93E-03	1.75E-03	0.274	7.930	-9.301
PSSV=5%	REG	1.5007	1.32E-04	1.27E-04	0.166	0.542	-3.531
Noise=5%	LPR	1.4990	2.43E-04	1.27E-04	0.054	1.000	-47.929
True=1.4982	LPR0	1.5022	1.80E-03	1.61E-03	0.265	7.386	-10.339
$\rho = 0.5$	HT	1.5094	1.76E-03	1.98E-03	0.568	9.144	12.436
PSSV=15%	REG	1.5011	8.54E-05	9.24E-05	0.018	0.443	8.172
Noise=0%	LPR	1.5021	1.93E-04	9.11E-05	0.084	1.000	-52.743
True=1.5008	LPR0	1.5091	1.63E-03	1.82E-03	0.548	8.473	11.331
$\rho = 0.9$	HT	0.4315	1.97E-04	1.80E-04	0.647	1.346	-8.782
PSSV=15%	REG	0.4314	1.45E-04	1.32E-04	0.638	0.989	-8.521
Noise=5%	LPR	0.4311	1.46E-04	1.31E-04	0.560	1.000	-9.983
True=0.4287	LPR0	0.4314	1.93E-04	1.76E-04	0.644	1.319	-9.023

Table 9: For all settings, 100 replications were run with a 11×11 population grid, and a 15×15 grid for the interpolation within the LPR estimator. PSSV is the proportion of small-scale variation. This table presents the results for a discrete population, $m = 100$ and $h=.35$.

variances estimates are negatively biased, often greatly so. We suspect much of this problem can be attributed to under-smoothing. If the bandwidth is too small, the sampled kernel smooth will follow the data points more closely than does the population level smooth. This results in small residuals and underestimation of variance.

A few additional runs of the simulation program were made to see if increasing the bandwidth would decrease the bias of the estimated variance of the local polynomial regression estimator. The results can be seen in table 9. The over-smoothing done in these runs greatly improves the bias problem, though there are still a few setting for which the variance is greatly underestimated.

As previously mentioned, these simulations were done on a continuous population. In the empirical example to follow, the data is discrete, made up of individual lakes. To verify

that these simulation results are applicable to a discrete population, a short additional simulation study was performed. In this study, the 11×11 grid of points was set up, and the spline calculated the same way as in the original study. Then, 10,000 random points over the unit square were selected. The interpolating spline was used to evaluate the response surface at each of these points. These 10,000 points then served as the population of interest. For each repetition of the simulation, a simple random sample of these points was selected. The estimators were evaluated as before, and summary statistics were computed at the end of each run. The results, in table 10, are very similar to those for the continuous population. This suggests that the earlier simulation results are applicable to both continuous and discrete resources.

4 Empirical Example

We applied the two-stage, two-dimensional technique to the EMAP Northeastern lakes survey. The sampling frame for this study consisted of over 20,000 lakes in 8 northeastern states. Over the course of 5 years, from 1991 to 1996, over 330 individual lakes were visited, each from one to six times. Each individual lake was treated as a cluster, and water samples within lakes (obtained on repeat visits) were treated as secondary sampling units. During a visit, many variables of interest were measured and recorded. For the purpose of this example, we examine the lake chemistry measures. Auxiliary information is available for each lake. For this example, we worked with Universal Transverse Mercator (UTM) coordinates, a form of spatial coordinates. For UTM coordinates, the world is divided into 60 zones. Each zone has an origin point. The UTM coordinates of a spatial location within the zone are then the meters to the north and east of this origin. The range of the UTM easting coordinates in the Northeastern lakes data set is 108095 to 1127690, and that of the UTM northing coordinates is 4311620 to 5263630. These location measures can serve as a proxy for other spatial covariates, or account for spatial correlation. We calculated nonparametric regression estimates of the average lake mean for several lake chemistry measures.

Table 11 displays the average mean for four chemistry measures calculated using the Horvitz-Thompson (HT), linear regression (REG), local linear regression (LPR), and local constant (LPR0) estimators. The estimated coefficient of variation is also given. The model assisted procedures are estimated to be much more efficient than the Horvitz-Thompson estimator. Additionally, the local regression estimators are estimated to make small gains over the linear regression estimator in each case. However, it was apparent from the simulation study that the LPR variance estimator is negatively biased. Based on those results, it is reasonable to conclude, for this example, that the LPR estimator improves upon the Horvitz-Thompson estimator and is competitive with the regression estimator. It should be noted that the trends in the variables over space are noisy, with weak linear trends.

To minimize numerical errors, all UTM coordinates were divided by 1,000,000. For both LPR and LPR0, a bandwidth of $h = 0.22$ was used in both the north-south (y) and east-

Settings	Estimator	Estimate	Var(Est)	Est.Var.	%Bias	Efficiency	%Bias Var.
$\rho = 0$	HT	1.5006	1.40E-03	1.69E-03	-0.187	11.707	20.616
PSSV=5%	REG	1.4985	3.06E-04	4.11E-05	-0.324	0.255	34.472
Noise=0%	LPR	1.4968	1.20E-04	3.07E-05	-0.441	1.000	-74.323
True=1.5034	LPR0	1.4974	1.59E-04	1.03E-04	-0.400	1.330	-35.295
$\rho = 0.9$	HT	0.4293	7.12E-05	7.88E-05	-0.033	3.505	10.704
PSSV=15%	REG	0.4288	2.94E-05	3.16E-05	-0.152	1.445	7.489
Noise=5%	LPR	0.4272	2.03E-05	1.39E-05	-0.526	1.000	-31.467
True=0.4295	LPR0	0.4278	1.92E-05	1.92E-05	-0.390	1.072	-11.875
$\rho = 0.5$	HT	0.5011	9.52E-05	1.23E-04	0.204	1.293	29.316
PSSV=15%	REG	0.5008	9.03E-05	1.06E-04	0.147	1.228	17.849
Noise=0%	LPR	0.4973	7.36E-05	5.72E-05	-0.562	1.000	-22.267
True=0.5001	LPR0	0.4990	6.70E-05	7.18E-05	-0.229	0.911	7.175
$\rho = 0$	HT	1.506	1.40E-03	1.69E-03	-0.187	11.698	20.619
PSSV=5%	REG	1.4985	3.07E-05	4.12E-05	-0.324	0.256	34.472
Noise=5%	LPR	1.4968	1.20E-04	3.08E-05	-0.441	1.000	-74.277
True=1.5034	LPR0	1.4974	1.59E-04	1.03E-04	-0.400	1.330	-35.267
$\rho = 0.5$	HT	0.5012	1.13E-04	1.46E-04	0.222	1.293	29.316
PSSV=15%	REG	0.5009	1.07E-04	1.26E-04	0.160	1.228	17.849
Noise=5%	LPR	0.4971	8.73E-05	6.79E-05	-0.612	1.000	-22.267
True=0.5001	LPR0	0.4989	7.95E-05	8.52E-05	-0.249	0.911	7.175
$\rho = 0.9$	HT	1.4594	1.61E-03	1.96E-03	-0.007	13.229	21.271
PSSV=5%	REG	1.4630	8.11E-06	9.36E-06	0.237	0.066	15.411
Noise=5%	LPR	1.4593	1.22E-04	3.83E-06	-0.015	1.000	-96.864
True=1.4595	LPR0	1.4606	1.67E-04	7.77E-05	0.073	1.367	-53.426
$\rho = 0.5$	HT	0.5005	3.67E-05	4.32E-05	0.328	1.371	17.648
PSSV=5%	REG	0.5015	3.16E-05	3.74E-05	0.529	1.178	18.438
Noise=0%	LPR	0.4989	2.68E-05	1.87E-05	0.011	1.000	-30.284
True=0.4988	LPR0	0.4996	2.33E-05	2.48E-05	0.153	0.869	5.895
$\rho = 0$	HT	1.4941	1.43E-03	1.82E-03	0.108	6.399	27.374
PSSV=15%	REG	1.4976	1.11E-04	1.31E-04	0.345	0.498	17.731
Noise=0%	LPR	1.4940	2.23E-04	9.46E-05	0.104	1.000	-57.596
True=1.4925	LPR0	1.4947	2.33E-04	1.90E-04	0.146	1.045	-18.566

Table 10: For all settings, 100 replications were run with a 11×11 population grid, and a 15×15 grid for the interpolation within the LPR estimator. PSSV is the proportion of small-scale variation. This table presents the results for $m = 100$ and $h=2$.

Chemistry Measure	HT	REG	LPR	LPR0
Log K	2.845 (6.88%)	2.916 (1.88%)	2.921 (1.80%)	2.890 (1.82%)
Log SO_4	4.828 (6.93%)	4.785 (0.93%)	4.791 (0.88%)	4.779 (0.94%)
Log Ca	5.835 (6.75%)	5.723 (1.49%)	5.742 (1.20%)	5.760 (1.37%)
Log Cl	4.531 (6.95%)	4.754 (2.20%)	4.772 (1.84%)	4.690 (2.03%)

Table 11: Estimated average mean of lake chemistry measures using Horvitz-Thompson (HT), linear regression (REG), local linear regression (LPR) and local linear regression fitting a local constant (LPR0). Numbers in parentheses are estimated coefficients of variation.

west (x) directions. This bandwidth is approximately 1/4 of the range of the standardized UTM coordinates in each direction.

Variance estimation was done using a stratified with replacement approximation, where the lakes were stratified by a clustering variable used in the original sampling process. The original sampling plan contained built-in spatial control to ensure a well-dispersed sample in space, with some similarity to systematic sampling (Larsen, et.al. 1993). The stratification employed here is analogous to the standard collapsed-stratum variance estimator for systematic sampling (Särndal, Swensson, and Wretman, p.423). The original sampling frame consisted of all lakes with their centroids in a grid of hexagons. These hexagons were spatially clustered. All clusters of hexagons were listed in random order, with the hexagons within a cluster randomly ordered and the lakes within a hexagon also listed randomly. A systematic sample was then taken (Larsen et al. 1993). This design forced spatial control in the sample.

To form the strata for the variance estimation, these spatial clusters were grouped together into 14 larger clusters. These clusters were created by combining the old clusters with their nearest neighbors. Each of the new clusters consists of between one and seven of the original clusters, with each of the new clusters containing at least 19 but not more than 122 lakes.

We did all these variance calculations at the first stage of sampling. For many of the lakes, only one sample was taken. This made it impossible to estimate the variability of the second stage of sampling. Because this second-stage variance is the same for all estimators, ignoring it here does not affect the comparisons of techniques.

5 Discussion and Conclusions

In this paper, we have introduced the local polynomial regression estimator in the context of two-stage spatial sampling. This estimator incorporates known population auxiliary

information to improve status estimates. Here, the estimator was described in the situation where the spatial location of every cluster in the population is known, in addition to the sample information collected. In order to motivate the introduction of this estimator, kernel smoothing and model-assisted estimation were discussed. The estimator was given, then variance estimation was discussed. To investigate the performance of the local polynomial regression estimator, a simulation study was performed. When the parametric model was incorrectly specified, the LPR estimator outperformed standard parametric techniques. The local polynomial regression technique was also applied to data from the EMAP Northeastern lakes data set. The estimator also performed well in this setting, giving comparable estimates, but smaller estimated standard errors than the other techniques. It should be noted, however, that the simulation study showed that the variance estimator for local polynomial regression is negatively biased, greatly so if the data is under-smoothed.

The local polynomial regression estimator was applied to both continuous and discrete data. The simulation was performed on a continuum, which was meant to mimic a continuous resource, such as an estuary. The data used for the empirical example is discrete; each lake is a unit. The technique appears valid for both types of resource.

Based on the results of the simulation study and the other work done here, the local polynomial regression estimator is a reasonable modeling choice in spatial sampling. Unless a population is known to have a planar trend, the local polynomial regression estimator should perform better than other estimators, especially if the variable of interest is spatially correlated or is highly variable. The local polynomial regression estimator also has the advantage of requiring little modeling efforts. Because the estimate can be expressed in terms of a weighted sum of sampled observations, with the weights remaining constant from one variable to the next, the technique can easily and quickly be applied to many study variables. In sampling with auxiliary information available for the entire population, the local polynomial regression estimator should improve the precision of estimation in populations with spatial correlation or other non-linear trends.

References

- Aldworth, J. and Cressie, N. (1999). Sampling Designs and Prediction Methods for Gaussian Spatial Processes. *Multivariate Analysis, Design of Experiments, and Survey Sampling*, edited by S. Ghosh. Dekker, New York, 1-54.
- Breidt, F.J. and Opsomer, J.D. (2000). Local polynomial regression estimation in survey sampling. *Annals of Statistics* **28**, 1026–1053.
- Horvitz, D.G. and D.J. Thompson. (1952). A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association* **47**, 663–685.
- Green, P.J., and Silverman, B.W.. (1994). *Nonparametric Regression and Generalized Linear Models: A Roughness Penalty Approach*, Chapman and Hall, London.
- Larsen, D.P., Thornton, K.W., Urquhart, N.S., Paulsen, S.G. (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.
- Särndal, C.-E., Swensson, B., and Wretman, J. (1992). *Model Assisted Survey Sampling*, Springer, New York.
- Wand, M.P. and Jones, M.C. (1995). *Kernel Smoothing*, Chapman and Hall, London.

Appendix I: Simulation Code for Continuous Population

```
"sim2d"<-
function(nreps=50, h = 0.35, n = 200, sigma = 0.041667, b0=.5, b1=1, b2=1,
grid= 14, rho=.00097656, noise=.01, izeed = 1965,seed2=2)
{
set.seed(izeed)

# 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)
}

# Define 2D kernel
kern2 <- function(x,y)
{
r <- rep(0, length(x))
r[abs(x) <= 1] <- 0.75 * (1 - x[abs(x) <= 1]^2)
t <- rep(0, length(y))
t[abs(y) <= 1] <- 0.75 * (1 - y[abs(y) <= 1]^2)
k <- r*t
return(k)
}

results <- matrix(0,nreps,8)

#Defining Z on a grid:
x_seq(0,1,.1)
X_rep(x,11)
Y_kronecker(x,rep(1,11))

if(rho > 0){
```

```

bigsig <- matrix(0,121,121)
for (i in 1:121){
for(j in 1:121){
bigsig[i,j] <- sigma*exp(log(rho)*((X[i]-X[j])^2+(Y[i]-Y[j])^2)^.5)
}
}
sig <- chol(bigsig)
Z <- b0+b1*X+b2*Y+t(sig)%*%rnorm(121)
}

if(rho == 0){
Z <- b0+b1*X+b2*Y+rnorm(121,,sigma^.5)
}
z <- matrix(Z,11,11)

eta <- function(x1,x2,y1,y2)
{
k <- ((x1-x2)^2+(y1-y2)^2)*log((x1-x2)^2+(y1-y2)^2)/(16*pi)
return(k)
}

#Fitting the Spline:
E <- matrix(0,121,121)
for(i in 1:121){
for(j in 1:121){
E[i,j] <- ifelse(i==j,0,eta(X[i],X[j],Y[i],Y[j]))
}
}
J <- rep(1,121)
zero <- matrix(0,3,3)
tmat <- cbind(J,X,Y)
tween <- cbind(E,tmat)
tween2 <- cbind(t(tmat),zero)
tween3 <- rbind(tween,tween2)
resp <- c(Z,rep(0,3))
ans <- solve(tween3)%*%resp
deltaspline <- ans[1:121]
aspline <- ans[122:124]

#Calculating the true volume:
intvect <- rep(0,121)
intvect[1] <- -0.1254142

```

```
intvect[2] <- -.1777895234
intvect[3] <- -.2115919090
intvect[4] <- -.2320571435
intvect[5] <- -.2428749331
intvect[6] <- -.2462383622
intvect[7] <- intvect[5]
intvect[8] <- intvect[4]
intvect[9] <- intvect[3]
intvect[10] <- intvect[2]
intvect[11] <- intvect[1]
intvect[12] <- intvect[2]
intvect[13] <- -.2180381735
intvect[14] <- -.2418674027
intvect[15] <- -.2549746967
intvect[16] <- -.2612957192
intvect[17] <- -.2631475665
intvect[18] <- intvect[16]
intvect[19] <- intvect[15]
intvect[20] <- intvect[14]
intvect[21] <- intvect[13]
intvect[22] <- intvect[2]
intvect[23] <- intvect[3]
intvect[24] <- intvect[14]
intvect[25] <- -.2574224860
intvect[26] <- -.2643900790
intvect[27] <- -.2669481000
intvect[28] <- -.2675335709
intvect[29] <- intvect[27]
intvect[30] <- intvect[26]
intvect[31] <- intvect[25]
intvect[32] <- intvect[14]
intvect[33] <- intvect[3]
intvect[34] <- intvect[4]
intvect[35] <- intvect[15]
intvect[36] <- intvect[26]
intvect[37] <- -.2667738701
intvect[38] <- -.2665113491
intvect[39] <- -.2661458855
intvect[40] <- intvect[38]
intvect[41] <- intvect[37]
intvect[42] <- intvect[26]
intvect[43] <- intvect[15]
```

```
intvect[44] <- intvect[4]
intvect[45] <- intvect[5]
intvect[46] <- intvect[16]
intvect[47] <- intvect[27]
intvect[48] <- intvect[38]
intvect[49] <- -.2645079219
intvect[50] <- -.2635546026
intvect[51] <- intvect[49]
intvect[52] <- intvect[38]
intvect[53] <- intvect[27]
intvect[54] <- intvect[16]
intvect[55] <- intvect[5]
intvect[56] <- intvect[6]
intvect[57] <- intvect[17]
intvect[58] <- intvect[28]
intvect[59] <- intvect[39]
intvect[60] <- intvect[50]
intvect[61] <- -.2624026140
intvect[62] <- intvect[50]
intvect[63] <- intvect[39]
intvect[64] <- intvect[28]
intvect[65] <- intvect[17]
intvect[66] <- intvect[6]
intvect[67] <- intvect[5]
intvect[68] <- intvect[16]
intvect[69] <- intvect[27]
intvect[70] <- intvect[38]
intvect[71] <- intvect[49]
intvect[72] <- intvect[50]
intvect[73] <- intvect[49]
intvect[74] <- intvect[38]
intvect[75] <- intvect[27]
intvect[76] <- intvect[16]
intvect[77] <- intvect[5]
intvect[78] <- intvect[4]
intvect[79] <- intvect[15]
intvect[80] <- intvect[26]
intvect[81] <- intvect[37]
intvect[82] <- intvect[38]
intvect[83] <- intvect[39]
intvect[84] <- intvect[38]
intvect[85] <- intvect[37]
```

```
intvect[86] <- intvect[26]
intvect[87] <- intvect[15]
intvect[88] <- intvect[4]
intvect[89] <- intvect[3]
intvect[90] <- intvect[14]
intvect[91] <- intvect[25]
intvect[92] <- intvect[26]
intvect[93] <- intvect[27]
intvect[94] <- intvect[28]
intvect[95] <- intvect[27]
intvect[96] <- intvect[26]
intvect[97] <- intvect[25]
intvect[98] <- intvect[14]
intvect[99] <- intvect[3]
intvect[100] <- intvect[2]
intvect[101] <- intvect[13]
intvect[102] <- intvect[14]
intvect[103] <- intvect[15]
intvect[104] <- intvect[16]
intvect[105] <- intvect[17]
intvect[106] <- intvect[16]
intvect[107] <- intvect[15]
intvect[108] <- intvect[14]
intvect[109] <- intvect[13]
intvect[110] <- intvect[2]
intvect[111] <- intvect[1]
intvect[112] <- intvect[2]
intvect[113] <- intvect[3]
intvect[114] <- intvect[4]
intvect[115] <- intvect[5]
intvect[116] <- intvect[6]
intvect[117] <- intvect[5]
intvect[118] <- intvect[4]
intvect[119] <- intvect[3]
intvect[120] <- intvect[2]
intvect[121] <- intvect[1]
```

```
true <- aspline[1]+.5*(aspline[2]+aspline[3])+deltaspline%*%intvect/(16*pi)
set.seed(seed2)
```

```
for(rr in 1:nreps){
```

```

# Taking a sample; using the spline to obtain z vales.
x.s <- runif(n)
y.s <- runif(n)
z.s <- rep(0,n)

E.s <- matrix(0,n,121)
for(i in 1:n){
for(j in 1:121){
if(x.s[i]!=X[j] || y.s[i]!=Y[j]){
E.s[i,j] <- eta(x.s[i],X[j],y.s[i],Y[j])
}
}
}
J2 <- rep(1,n)
tmat.s <- cbind(J2,x.s,y.s)
z.s <- E.s%%deltaspline+tmat.s%%aspline
if(noise>0){
z.s <- z.s+rnorm(n,sd=(noise^.5))
}

t.hat1 <- z.s
x.clust <- x.s
y.clust <- y.s

wgt <- rep(1/n,n)
ht1 <- mean(z.s)

#Regression estimator:
J <- rep(1,n)
Xr <- cbind(J,x.clust,y.clust)
W <- diag(wgt)
B1 <- solve(t(Xr)%*%W%*%Xr)%*%(t(Xr)%*%W%*%t.hat1)
e.1 <- t.hat1-Xr%*%B1
t.reg1 <- B1[1]+.5*B1[2]+.5*B1[3]+ sum(e.1*(wgt))

xmax <- max(x.s)
xmin <- min(x.s)
xstar <- xmin + ((xmax - xmin) * (0:grid))/grid
xstar <- rep(xstar,grid+1)
ymax <- max(y.s)

```

```

ymin <- min(y.s)
ystar <- ymin + ((ymax - ymin) * (0:grid))/grid
J <- rep(1,grid+1)
ystar <- kronecker(ystar, J, fun="*")
size <- length(xstar)

##Local Polynomial Estimator

wstar <- matrix(0,n,size)
wstar0 <- wstar
for(i in 1:size) {
W <- kern2((x.s - xstar[i])/h,(y.s - ystar[i])/h)*wgt/h^2
a1 <- sum(W)
b1 <- sum(W*(x.s-xstar[i]))
c1 <- sum(W*(y.s-ystar[i]))
d1 <- sum(W*(x.s-xstar[i])^2)
e1 <- sum(W*(x.s-xstar[i])*(y.s-ystar[i]))
f1 <- sum(W*(y.s-ystar[i])^2)
D <- a1*d1*f1-a1*e1^2+2*b1*e1*c1-b1^2*f1-d1*c1^2
wstar[,i] <- ((d1*f1-e1^2)*W+(-b1*f1+c1*e1)*W*(x.s-xstar[i])+
(b1*e1-c1*d1)*W*(y.s-ystar[i]))/D
wstar0[,i] <- W/a1
}

freq <- rep(1/size,size)
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)*kern((y.s - y.s[i])/h)*wgt/h^2
a1 <- sum(W)
b1 <- sum(W*(x.s-x.s[i]))
c1 <- sum(W*(y.s-y.s[i]))
d1 <- sum(W*(x.s-x.s[i])^2)
e1 <- sum(W*(x.s-x.s[i])*(y.s-y.s[i]))
f1 <- sum(W*(y.s-y.s[i])^2)
D <- a1*d1*f1-a1*e1^2+2*b1*e1*c1-b1^2*f1-d1*c1^2

```

```

ww[,i] <- ((d1*f1-e1^2)*W+(-b1*f1+c1*e1)*W*(x.s-x.s[i])+
(b1*e1-c1*d1)*W*(y.s-y.s[i]))/D
ww0[,i] <- W/a1
}

lpr.wt <- wgt + c(apply(what, MAR = 1, FUN = "sum")) - c(
apply(t(wgt*t(ww)), MAR = 1, FUN = "sum"))
lpr0.wt <- wgt + c(apply(what0, MAR = 1, FUN = "sum")) - c(
apply(t(wgt*t(ww0)), MAR = 1, FUN = "sum"))

#LPR totals:
t1.lpr <- (rbind(lpr.wt) %*% z.s)
t1.lpr0 <- (rbind(lpr0.wt) %*% z.s)
results[rr,1] <- ht1
results[rr,3] <- t.reg1
results[rr,5] <- t1.lpr
results[rr,7] <- t1.lpr0

#Variance calculations:
#Population 1
z1.ht <- t.hat1
z1.reg <- e.1
z1.lpr <- (t.hat1-t(ww))%*%t.hat1
z1.lpr0 <- (t.hat1-t(ww0))%*%t.hat1

v1.ht <- var(z1.ht)/n
v1.reg <- var(z1.reg)/n
v1.lpr <- var(z1.lpr)/n
v1.lpr0 <- var(z1.lpr0)/n
results[rr,2] <- v1.ht
results[rr,4] <- v1.reg
results[rr,6] <- v1.lpr
results[rr,8] <- v1.lpr0

}

output <- matrix(0,4,6)

for(i in 1:4){
output[i,1] <- mean(results[,2*i-1])

```

```

output[i,2] <- var(results[,2*i-1])
output[i,3] <- mean(results[,2*i])
}

for(i in 1:4){
output[i,4] <- (output[i,1]-true)*100/true
output[i,5] <- output[i,2]/output[3,2]
output[i,6] <- (output[i,3]-output[i,2])*100/output[i,2]
}

return(nreps, h, n, sigma, grid, true, output, results)
}

```

Appendix II: Simulation Code for Discrete Population

```

"dsim2d"<-
function(nreps=50, h = 0.35, n = 200, npop=1000, sigma = 0.041667, b0=.5, b1=1,
b2=1, grid= 14, rho=.00097656, noise=.01, iseed = 1965)
{
set.seed(iseed)

# 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)
}

# Define 2D kernel
kern2 <- function(x,y)
{
r <- rep(0, length(x))
r[abs(x) <= 1] <- 0.75 * (1 - x[abs(x) <= 1]^2)
t <- rep(0, length(y))
t[abs(y) <= 1] <- 0.75 * (1 - y[abs(y) <= 1]^2)
k <- r*t
return(k)
}

results <- matrix(0,nreps,8)

```

```

#Defining Z on a grid:
x_seq(0,1,.1)
X_rep(x,11)
Y_kronecker(x,rep(1,11))

if(rho > 0){
bigsig <- matrix(0,121,121)
for (i in 1:121){
for(j in 1:121){
bigsig[i,j] <- sigma*exp(log(rho)*((X[i]-X[j])^2+(Y[i]-Y[j])^2)^.5)
}
}
sig <- chol(bigsig)
Z <- b0+b1*X+b2*Y+t(sig)%*%rnorm(121)
}

if(rho == 0){
Z <- b0+b1*X+b2*Y+rnorm(121,.,sigma^.5)
}
z <- matrix(Z,11,11)

eta <- function(x1,x2,y1,y2)
{
k <- ((x1-x2)^2+(y1-y2)^2)*log((x1-x2)^2+(y1-y2)^2)/(16*pi)
return(k)
}

#Fitting the Spline:
E <- matrix(0,121,121)
for(i in 1:121){
for(j in 1:121){
E[i,j] <- ifelse(i==j,0,eta(X[i],X[j],Y[i],Y[j]))
}
}
J <- rep(1,121)
zero <- matrix(0,3,3)
tmat <- cbind(J,X,Y)
tween <- cbind(E,tmat)
tween2 <- cbind(t(tmat),zero)
tween3 <- rbind(tween,tween2)
resp <- c(Z,rep(0,3))

```

```

ans <- solve(tween3)%*%resp
deltaspline <- ans[1:121]
aspline <- ans[122:124]

#Generating a population:
x.p <- runif(npop)
y.p <- runif(npop)
z.p <- rep(0,npop)

E.p <- matrix(0,npop,121)
for(i in 1:npop){
for(j in 1:121){
if(x.p[i]!=X[j] || y.p[i]!=Y[j]){
E.p[i,j] <- eta(x.p[i],X[j],y.p[i],Y[j])
}
}
}

J2 <- rep(1,n)
tmat.p <- cbind(J2,x.p,y.p)
z.p <- E.p%*%deltaspline+tmat.p%*%aspline

true <- mean(z.p)

for(rr in 1:nreps){

# Taking a sample
s <- sample(1:npop,size=n,replace=F)
x.s <- x.p[s]
y.s <- y.p[s]
z.s <- z.p[s]

t.hat1 <- z.s
x.clust <- x.s
y.clust <- y.s

wgt <- rep(1/n,n)
ht1 <- mean(z.s)

#Regression estimator:
J <- rep(1,n)
Xr <- cbind(J,x.clust,y.clust)

```

```

W <- diag(wgt)
B1 <- solve(t(Xr)%*%W%*%Xr)%*%(t(Xr)%*%W%*%t.hat1)
e.1 <- t.hat1-Xr%*%B1
t.reg1 <- B1[1]+.5*B1[2]+.5*B1[3]+ sum(e.1*(wgt))

xmax <- max(x.s)
xmin <- min(x.s)
xstar <- xmin + ((xmax - xmin) * (0:grid))/grid
xstar <- rep(xstar,grid+1)
ymax <- max(y.s)
ymin <- min(y.s)
ystar <- ymin + ((ymax - ymin) * (0:grid))/grid
J <- rep(1,grid+1)
ystar <- kronecker(ystar, J, fun="*")
size <- length(xstar)

##Local Polynomial Estimator

wstar <- matrix(0,n,size)
wstar0 <- wstar
for(i in 1:size) {
W <- kern2((x.s - xstar[i])/h,(y.s - ystar[i])/h)*wgt/h^2
a1 <- sum(W)
b1 <- sum(W*(x.s-xstar[i]))
c1 <- sum(W*(y.s-ystar[i]))
d1 <- sum(W*(x.s-xstar[i])^2)
e1 <- sum(W*(x.s-xstar[i])*(y.s-ystar[i]))
f1 <- sum(W*(y.s-ystar[i])^2)
D <- a1*d1*f1-a1*e1^2+2*b1*e1*c1-b1^2*f1-d1*c1^2
wstar[,i] <- ((d1*f1-e1^2)*W+(-b1*f1+c1*e1)*W*(x.s-xstar[i])+
(b1*e1-c1*d1)*W*(y.s-ystar[i]))/D
wstar0[,i] <- W/a1
}

freq <- rep(1/size,size)
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)*kern((y.s - y.s[i])/h)*wgt/h^2
a1 <- sum(W)
b1 <- sum(W*(x.s-x.s[i]))
c1 <- sum(W*(y.s-y.s[i]))
d1 <- sum(W*(x.s-x.s[i])^2)
e1 <- sum(W*(x.s-x.s[i])*(y.s-y.s[i]))
f1 <- sum(W*(y.s-y.s[i])^2)
D <- a1*d1*f1-a1*e1^2+2*b1*e1*c1-b1^2*f1-d1*c1^2
ww[,i] <- ((d1*f1-e1^2)*W+(-b1*f1+c1*e1)*W*(x.s-x.s[i])+
(b1*e1-c1*d1)*W*(y.s-y.s[i]))/D
ww0[,i] <- W/a1
}

```

```

lpr.wt <- wgt + c(apply(what, MAR = 1, FUN = "sum")) - c(
apply(t(wgt*t(ww)), MAR = 1, FUN = "sum"))
lpr0.wt <- wgt + c(apply(what0, MAR = 1, FUN = "sum")) - c(
apply(t(wgt*t(ww0)), MAR = 1, FUN = "sum"))

```

#LPR totals:

```

t1.lpr <- (rbind(lpr.wt) %*% z.s)
t1.lpr0 <- (rbind(lpr0.wt) %*% z.s)
results[rr,1] <- ht1
results[rr,3] <- t.reg1
results[rr,5] <- t1.lpr
results[rr,7] <- t1.lpr0

```

#Variance calculations:

#Population 1

```

z1.ht <- t.hat1
z1.reg <- e.1
z1.lpr <- (t.hat1-t(ww))%*%t.hat1)
z1.lpr0 <- (t.hat1-t(ww0))%*%t.hat1)

```

```

v1.ht <- var(z1.ht)/n
v1.reg <- var(z1.reg)/n
v1.lpr <- var(z1.lpr)/n
v1.lpr0 <- var(z1.lpr0)/n
results[rr,2] <- v1.ht

```

```

results[rr,4] <- v1.reg
results[rr,6] <- v1.lpr
results[rr,8] <- v1.lpr0

}

output <- matrix(0,4,6)

for(i in 1:4){
output[i,1] <- mean(results[,2*i-1])
output[i,2] <- var(results[,2*i-1])
output[i,3] <- mean(results[,2*i])
}

for(i in 1:4){
output[i,4] <- (output[i,1]-true)*100/true
output[i,5] <- output[i,2]/output[3,2]
output[i,6] <- (output[i,3]-output[i,2])*100/output[i,2]
}

return(nreps, h, n, sigma, grid, true, output)

```

Appendix III: Empirical Example Code

```

"estimate4"<-
function(x1=NewUse$UTMX/1000000, y1=NewUse$UTMY/1000000, z=NewUse$LogK,
clust=NewUse$LAKE.ID, grid=50, h = .22, weight = NewUse$WGT.3X,
x=NELAKES2$UTMXeasting/1000000, y=NELAKES2$UTMYnorthing/1000000)
{
t.hat <- tapply(z,clust,mean)
x.clust <- tapply(x1,clust,mean)
y.clust <- tapply(y1,clust,mean)
n <- length(t.hat)
N <- length(x)
wgt <- .5 * tapply(weight,clust,mean)
ht <- t(wgt)%*%t.hat/N
unweightedtotal <- mean(t.hat)*N

##Regression estimator:
J <- rep(1,n)

```

```

X <- cbind(J,x.clust,y.clust)
W <- diag(wgt)
B <- solve(t(X)%*%W%*%X)%*%(t(X)%*%W%*%t.hat)
e <- t.hat-X%*%B
X2 <- cbind(rep(1,N),x,y)
y.hat <- sum(X2%*%B)
t.reg <- (y.hat + sum(e*(wgt)))/N

# 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)
}

# Define 2D kernel
kern2 <- function(x,y)
{
r <- rep(0, length(x))
r[abs(x) <= 1] <- 0.75 * (1 - x[abs(x) <= 1]^2)
t <- rep(0, length(y))
t[abs(y) <= 1] <- 0.75 * (1 - y[abs(y) <= 1]^2)
k <- r*t
return(k)
}

#Define Grid for Interpolation
# if(grid > 0) {
# xmax <- max(x)
# xmin <- min(x)
# xstar <- xmin + ((xmax - xmin) * (0:grid))/grid
# ymax <- max(y)
# ymin <- min(y)
# ystar <- ymin + ((ymax - ymin) * (0:grid))/grid
# }
# else {
# xstar <- x
# ystar <- y
# }
# grid <- length(xstar) - 1

```

```

binned <- hexbin(x,y,grid)
grid <- length(binned$count)
xstar <- binned$xcenter
ystar <- binned$ycenter

##Local Polynomial Estimator
z.s <- t.hat
x.s <- x.clust
y.s <- y.clust

wstar <- matrix(0,n,grid)
wstar0 <- wstar
for(i in 1:grid) {
W <- kern2((x.s - xstar[i])/h,(y.s - ystar[i])/h)*wgt/h^2
a1 <- sum(W)
b1 <- sum(W*(x.s-xstar[i]))
c1 <- sum(W*(y.s-ystar[i]))
d1 <- sum(W*(x.s-xstar[i])^2)
e1 <- sum(W*(x.s-xstar[i])*(y.s-ystar[i]))
f1 <- sum(W*(y.s-ystar[i])^2)
D <- a1*d1*f1-a1*e1^2+2*b1*e1*c1-b1^2*f1-d1*c1^2
wstar[,i] <- ((d1*f1-e1^2)*W+(-b1*f1+c1*e1)*W*(x.s-xstar[i]))+
(b1*e1-c1*d1)*W*(y.s-ystar[i])/D
wstar0[,i] <- W/a1
}

freq <- binned$count
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)*kern((y.s - y.s[i])/h)*wgt/h^2
a1 <- sum(W)
b1 <- sum(W*(x.s-x.s[i]))
c1 <- sum(W*(y.s-y.s[i]))
d1 <- sum(W*(x.s-x.s[i])^2)
e1 <- sum(W*(x.s-x.s[i])*(y.s-y.s[i]))
}

```

```

f1 <- sum(W*(y.s-y.s[i])^2)
D <- a1*d1*f1-a1*e1^2+2*b1*e1*c1-b1^2*f1-d1*c1^2
ww[,i] <- ((d1*f1-e1^2)*W+(-b1*f1+c1*e1)*W*(x.s-x.s[i])+
(b1*e1-c1*d1)*W*(y.s-y.s[i]))/D
ww0[,i] <- W/a1
}

lpr.wt <- wgt + c(apply(what, MAR = 1, FUN = "sum")) - c(
apply(t(wgt*t(ww)), MAR = 1, FUN = "sum"))
lpr0.wt <- wgt + c(apply(what0, MAR = 1, FUN = "sum")) - c(
apply(t(wgt*t(ww0)), MAR = 1, FUN = "sum"))
t.lpr <- (rbind(lpr.wt) %*% z.s)/N
t.lpr0 <- (rbind(lpr0.wt) %*% z.s)/N

varclust <- tapply(NewUse$CLUSTER.1,clust,mean)
n.var <- tapply(t.hat,varclust,length)
n.h <- rep(0,n)
index <- 1
for (i in 1:length(n.var)){
n.h[index:(index+n.var[i]-1)] <- rep(n.var[i],n.var[i])
index <- index + n.var[i]
}

newclust <- sort(varclust)
z.ht <- unlist(tapply(t.hat,varclust,sort))*wgt*n.h/N
z.reg <- unlist(tapply(e,varclust,sort))*wgt*n.h/N
z.lpr <- unlist(tapply((t.hat-t(ww))%*%t.hat),varclust,sort))*wgt*n.h/N
z.lpr0 <- unlist(tapply((t.hat-t(ww0))%*%t.hat),varclust,sort))*wgt*n.h/N

l.var.ht <- tapply(z.ht,newclust,var)
l.var.reg <-tapply(z.reg,newclust,var)
l.var.lpr <-tapply(z.lpr,newclust,var)
l.var.lpr0 <-tapply(z.lpr0,newclust,var)

v.ht <- sum(l.var.ht/n.var)
v.reg <- sum(l.var.reg/n.var)
v.lpr <- sum(l.var.lpr/n.var)
v.lpr0 <- sum(l.var.lpr0/n.var)

cv.ht <- v.ht^.5/ht*100
cv.reg <- v.reg^.5/t.reg*100

```

```
cv.lpr <- v.lpr^.5/t.lpr*100  
cv.lpr0 <- v.lpr0^.5/t.lpr0*100
```

```
return(h,unweightedtotal,ht,v.ht,cv.ht,t.reg,v.reg,cv.reg,t.lpr,  
v.lpr,cv.lpr,t.lpr0,v.lpr0,cv.lpr0)
```