SEMI-PARAMETRIC ESTIMATION IN TOBIT REGRESSION MODELS

by

CHUNXIA CHEN

B.S., College of Medicine- Southeast University, China, 1999

A REPORT

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Statistics
College of Arts and Sciences

KANSAS STATE UNIVERSITY
Manhattan, Kansas
2013

Approved by:

Major Professor
Weixing Song

Abstract

In the classical Tobit regression model, the regression error term is often assumed to have a zero mean normal distribution with unknown variance, and the regression function is assumed to be linear. If the normality assumption is violated, then the commonly used maximum likelihood estimate becomes inconsistent. Moreover, the likelihood function will be very complicated if the regression function is nonlinear even the error density is normal, which makes the maximum likelihood estimation procedure hard to implement. In the full nonparametric setup when both the regression function and the distribution of the error term ε are unknown, some nonparametric estimators for the regression function has been proposed. Although the assumption of knowing the distribution is strict, it is a widely adopted assumption in Tobit regression literature, and is also confirmed by many empirical studies conducted in the econometric research. In fact, a majority of the relevant research assumes that ε possesses a normal distribution with mean 0 and unknown standard deviation. In this report, we will try to develop a semi-parametric estimation procedure for the regression function by assuming that the error term follows a distribution from a class of 0-mean symmetric location and scale family. A minimum distance estimation procedure for estimating the parameters in the regression function when it has a specified parametric form is also constructed. Compare with the existing semiparametric and nonparametric methods in the literature, our method would be more efficient in that more information, in particular the knowledge of the distribution of ε, is used. Moreover, the computation is relative inexpensive. Given lots of application does assume that ε has normal or other known distribution, the current work no doubt provides some more practical tools for statistical inference in Tobit regression model.

Table of Contents

Table of Contents **iv**

List of Figures **v**

List of Tables **vi**

Acknowledgements **vii**

Dedication **viii**

1 Introduction **1**

1.1 Tobit Regression Model 1

1.2 The Research Objective and Literature Review 3

2 Semi-Parametric Regression Procedure **7**

3 Simulation Studies **13**

3.1 Semi-Parametric Estimation of $m(x)$ 14

3.2 Biases and MSEs of the MDEs 17

A R-Programs **24**

List of Figures

3.1 Kernel Smoothing, $y_0 = 0.5$ 17

3.2 Kernel Smoothing, $y_0 = 1$ 17

3.3 Local Linear Smoothing, $y_0 = 0.5$ 17

3.4 Local Linear Smoothing, $y_0 = 1$ 17

List of Tables

3.1 Kernel, dpill bandwidth . . . 15
3.2 Local Linear, dpill bandwidth . . . 15
3.3 Kernel, $h = 0.5n^{-1/5}$ . . . 16
3.4 Kernel, default bandwidth . . . 16
3.5 Kernel, Linear, $y_0 = 0.5$ . . . 18
3.6 Kernel, Linear, $y_0 = 1$ . . . 18
3.7 Kernel, Quadratic, $y_0 = 0.5$ . . . 18
3.8 Kernel, Quadratic, $y_0 = 1$ . . . 19
3.9 Local Linear, Linear, $y_0 = 0.5$ . . . 19
3.10 Local Linear, Linear, $y_0 = 1$ . . . 19
3.11 Local Linear, Quadratic, $y_0 = 0.5$ . . . 20
3.12 Local Linear, Quadratic, $y_0 = 1$ . . . 20

Acknowledgments

First of all, I am heartily thankful to my major professor, Dr. Weixing Song, whose encouragement, guidance and support from the initial to the final level enabled me to develop an understanding of the subject. Without his guidance and persistent help this dissertation would not have been possible.

Then I would like to thank Dr. Gary L. Gadbury and Dr. Juan Du, for being the Committee members for this report, and making comments, and sharing ideas.

I would like to thank Dr. Guihua Bai for his great support and help during my graduate study. I thank Dr. Amy Bernardo for her patient guidance in the KSU DNA Sequencing and Genotyping Facility.

Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of the report.And I thank all the people in the Department of Statistics at Kansas State University. They are the nicest people, and together, we made the department!

Dedication

This work is dedicated to my grandma, my parents, my husband Hao Yan and my adorable son Daniel Yan, for their love, support, and understanding.

Chapter 1

Introduction

1.1 Tobit Regression Model

Censored and truncated data are very common data types studied in the areas such as econometrics, biometrics, agricultural study, engineering and family study. Variety statistical models have been constructed to fit these data and to make further statistical inferences. Among all the statistical models developed so far, Tobit regression models no doubt are the most frequently used modeling procedures.

When studying the relationship between household expenditures on durable goods and household incomes, Tobin (1958) noted that although a large portion of the data follows a linear pattern, yet an important feature of the data is that there are few observations flatted at zero. Therefore, imposing the linearity assumption on the whole data set is clearly inappropriate. To find a proper statistical model to fit his data, Tobin (1958) first developed a utility model to explain the phenomenon discovered in the study, and eventually formulated the so called Tobit regression model

$$Y^* = m(X) + \varepsilon, \quad Y = \max\{Y^*, y_0\}, \tag{1.1}$$

where X is the explanatory vector of dimension p and its value can be observed directly, Y^* is the response variable and can only be observed if $Y^* \geq y_0$ for some pre-specified threshold y_0, $m(x)$ denotes the regression function $E(Y^*|X)$. ε denotes the random errors and may be interpreted as the collection of all the unobservable variables which affect the response variable. The name, Tobit model, was coined by Goldberger(1964) inspired by its similarity to Probit model. See Maddala (1983) and Amemiya (1984) for a comprehensive discussion on Tobit models and its variants, together with some important estimation procedures. Recent application of Tobit regression modeling includes McConnel and Zetzman (1993)'s study on the differences between urban and rural elderly persons in the use of hospital, nursing home, and physician services, McConnel and Zetzman (1997)'s study on the relationship between land use and NO3-N concentrations in drinking water wells. Although it is the 90's that witnessed the wide application of Tobit regression model, its appeal doesn't fade with the elapse of time. On the contrary, Tobit regression model has its unique advantage in dealing with biased and inconsistent parameter estimates caused by the inappropriate use of standard ordinary least squares, and is being paid with more and more attention.

In its the early development, the regression function $m(x)$ was assumed to be linear $m(x) = x'\beta$ and the random error ε to be normally distributed with mean 0 and a possibly unknown variance σ^2, where β is unkown regression coefficients. The existing work on this standard Tobit regression model mainly focuses on the estimation of $\theta = (\beta', \sigma^2)'$. Under the normality assumption of the error term ε, Amemiya (1973) and Heckman (1976,1979) proposed consistent estimators for θ, but these estimators lose their consistency if the normality assumption is violated. A robust estimator of θ was proposed by Powell (1984) based on the least absolute deviations and was shown to be consistent and asymptotically normal without assuming the normality.

Generally speaking, assuming that $m(x)$ has a linear or other parametric form is either based on some empirical evidence or simply for the sake of mathematical convenience. Misidentification of the regression function often results in misleading conclusions. For example, it is well known that violation of the linearity assumption can produce inconsistent estimators of the parameters and biased prediction of the survival time in censored regression models. See Horowitz and Neumann (1989) for a detailed discussion on this issue. Therefore, from both theoretical and practical points of view, it is necessary to develop certain semiparametric or nonparametric estimation procedures in the Tobit regression models, without assuming a rigid parametric form for the regression functions.

Because of its flexibility in exploring the data structures, nonparametric modeling has enjoyed a long lasting popularity among researchers and practitioners, and extensive research has been done in the literature. Complete nonparametric estimation procedures were already been tried for Tobit regression models by Lewbel and Linton (2002), and Zhou (2007) without even assuming the knowledge of the distribution of ε.

1.2 The Research Objective and Literature Review

Abundant research in the literature was conducted on how to estimate the regression coefficients β and σ^2 when the regression function in model (1.1) is linear, and the error term ε has a normal distribution. Amemiya (1984)'s survey paper provided a panoramic view on the early development of various estimation methods. The probit MLE can only consistently estimate the ratio of the slop and the standard deviation of the error term. This loss of efficiency is not beyond our expectation since the estimation procedure only used the truncation information from the data, and totally ignored its numerical value even when it is observed. The probit MLE is often served as the initial values in the iteration algorithms of other estimation procedures. The nonlinear least square and weighted least square estimation, Heckman's two step estimation were all based on the following two key observations:

$$E(Y|Y>0,X) = X'\beta + \frac{\sigma\phi(X'\beta/\sigma)}{\Phi(X'\beta/\sigma)}, \quad E(Y|X) = X'\beta\Phi(X'\beta/\sigma) + \sigma\phi(X'\beta/\sigma). \tag{1.2}$$

The above expressions provide two heteroscedastic regression models,

$$Y = X'\beta + \frac{\sigma\phi(X'\beta/\sigma)}{\Phi(X'\beta/\sigma)} + \xi, \text{ for } Y>0; \quad Y = X'\beta\Phi(X'\beta/\sigma) + \sigma\phi(X'\beta/\sigma) + \eta. \tag{1.3}$$

The nonlinear least square estimators and the weighted nonlinear least square estimators are shown to be asymptotically normal. Intuitively, one may think that the weighted nonlinear least square estimator would perform better than the nonlinear least square estimator, but a definite comparison between these procedures is not possible due to the fact that the asymptotic covariance matrices from both procedures are hard to compare. A simulation study by Wales and Woodland (1980) based on only one replication with sample sizes of 1000 and 5000 showed that the nonlinear least square procedures are distinctly inferior to the MLE procedures. This is also confirmed by our simulation studies conducted in Antoneitte (2012). A computationally efficient estimation procedure is provided by Heckman (1979), known as Heckman's two-step estimator. In the first step, an initial value for β/σ is obtained from Probit MLE procedure, then in the second step, this initial value is inserted into equation (2.2), then a linear regression of Y against ether $(X, \phi(X'\beta/\alpha)/\Phi(X'\beta/\sigma))$, or $(X\Phi(X'\beta/\sigma), \phi(X'\beta/\alpha))$ will provide the estimates for β and σ. Again, one may think the estimators obtained based on the second model in (2.2) would perform better than the ones obtained from the first model in (2.2), but the actual performance of these estimators depends on the true parameter values, and the direct comparison is not possible. The most efficient estimation procedure in this parametric setup is of course the MLE. The usual MLE procedure by equating the derivative of the log-likelihood function with respect to β, σ is not applicable, since the Tobit likelihood function is not globally concave with respect to the original parameters β and σ, as showed in Amemiya (1973). However, Olsen (1978) proved that the log-likelihood function is globally concave in the transformed parameters $\alpha = \beta/\sigma$ and $h = 1/\sigma$, so a standard iterative method such as Newton-Raphson or Fisher Scoring method will always provide estimators converging to the global maximum of the log-likelihood function. Empirical studies showed that a good initial value for the iterations in MLE procedure can greatly speed up the convergence. The EM algorithm for searching the MLE was proposed by Amemiya (1984). Under some regularity conditions, Amemiya (1984) proved the convergence of the EM algorithm. The regularity conditions do not generally hold for the Tobit model, however, if the sample size is sufficiently large then they do hold, and if the iteration of the EM algorithm is started from a good initial value, then a rapid convergence can be achieved.

The above mentioned procedures are developed under the normality assumption of the error term ε. If this assumption is violated, then generally these estimators become inconsistent. Some robust estimation procedures are proposed to accommodate the non-normal ε. For example, Powell (1984) proposed an estimator which is a generalization of least absolute deviations estimation for the standard linear model, and, unlike estimation methods based on the assumption of normally distributed error terms, the estimator is consistent and asymptotically normal for a wide class of error distributions, and is also robust to heteroscedasticity under some regularity conditions. The most recent work on this area is Guardiola (2012)'s work on the robust tobit regression when errors are from the so called epsilon skew exponential power distribution.

Most development on the parametric and semi-parametric estimators on Tobit regression model or other more general censored regression model includes Buckley and James (1979), Koul, Suslara, and Van Ryzin (1981), Powell (1986a, 1986b), Powell, Stock and Stoker (1989), Honore and Powell (1994), Zhou (2007) and the references therein. A review on the recent developments on this area can be found in Yvette et al. (2011).

In the full nonparametric setup when both the regression function and the distribution of the error term ε are unknown, Lewbel and Linton (2002) provides consistent estimators of $m(x)$ and its derivatives. They showed that the convergence rate is the same as for an uncensored nonparametric regression and its derivatives. A $\sqrt{n}$-consistent estimates of weighted average derivatives of $m(x)$ is also derived, which enables us to estimate the coefficients in linear or partly linear specification for $m(x)$ with parametric convergence rate. Their work also allows an extension to the heteroscedasticity case. Based on a location relationship about the conditional survival function of the censored dependent variable, Zhou (2007) constructs a nonparametric estimator for the regression function, which is the minimizer of an integrated least-squares type sample objective function in which the conditional survival function is estimated by kernel method. Under some regularity conditions, the nonparametric estimator is shown to be consistent and asymptotically normal. Although simulation studies show that Zhou (2007)'s estimator sometimes performs better than Lewbel and Linton (2002)'s estimator, but the superiority is not obvious.

Although the assumption of knowing the distribution is strict, it is a widely adopted assumption in Tobit regression literature, and is also confirmed by many empirical studies conducted in the econometric research. In fact, a majority of the relevant research assumes that ε possesses a normal distribution with mean 0 and unknown standard deviation. In this report, we will try to develop a semi-parametric estimation procedure for the regression function by assuming that the error term follows a distribution from a class of 0-mean symmetric location and scale family. A minimum distance estimation procedure for estimating the parameters in the regression function when it has a specified parametric form is also constructed. Compare with the existing semiparametric and nonparametric methods in the literature, our method would be more efficient in that more information, in particular the knowledge of the distribution of ε, is used. Given lots of application does assume that ε has normal or other known distribution, the current work no doubt provides some more practical tools for statistical inference in Tobit regression model.

Chapter 2

Semi-Parametric Regression Procedure

In this chapter we will develop an estimation procedure for the regression function under the assumption that the error term has a known distribution. To be specific, consider the following semi-parametric Tobit regression model:

$$Y^* = m(X) + \varepsilon; \quad Y = \max\{Y^*, y_0\}, \tag{2.1}$$

where y_0 is a known threshold. It is often assumed to be 0, simply because an unknown y_0 ban be absorbed into $m(x)$. In current report, we will keep y_0 as it is, and the algorithm we developed surely can be applied to $y_0 = 0$ case. The following regularity condition on ε will be adopted in the report.

(C). The density function of ε is symmetric around 0 and is a member of a scale family $\{f(\cdot/\sigma)/\sigma : \sigma > 0\}$; The CDF of f is strictly increasing.

(C) is not a strict condition, since commonly used distribution in the literature, such as Normal, Laplace, t distributions all satisfy this condition. The following three questions will be addressed in the current report.

(1). How to estimate $m(x)$ nonparametrically?

(2). How to estimate σ^2?

(3). How to estimate the regression parameters in $m(x)$ if $m(x)$ has a parametric form?

By assumption (C), denote the density function of ε as $f(\cdot/\sigma)/\sigma$, where f is symmetric around 0. For convenience, denote $Q_j(x) = \int_x^\infty u^j f(u)du$, $j = 0, 1$, therefore,

$$\int_x^\infty \frac{u^j}{\sigma} f\left(\frac{u}{\sigma}\right) du = \sigma^j \int_{x/\sigma}^\infty u^j f(u)du = \sigma^j Q_j(x/\sigma), \quad j = 0, 1.$$

Let $g_1(x) = E[I(Y = y_0)|X = x]$, $g_2(x) = E(Y|X = x)$, then

$$g_1(x) = 1 - Q_0\left(\frac{y_0 - m(x)}{\sigma}\right), \tag{2.2}$$

$$g_2(x) = y_0 - \sigma\left[\left(\frac{y_0 - m(x)}{\sigma}\right) Q_0\left(\frac{y_0 - m(x)}{\sigma}\right) - Q_1\left(\frac{y_0 - m(x)}{\sigma}\right)\right]. \tag{2.3}$$

By assumption (C), one can show that, for any fixed y_0 and σ, $1-Q_0(x)$ and $y_0-\sigma(xQ_0(x)-Q_1(x))$, as functions of x, is strictly monotone. In fact, note that

$$\frac{\partial Q_0(x)}{\partial x} = -f(x), \quad \frac{\partial Q_1(x)}{\partial x} = -xf(x),$$

Hence

$$\frac{\partial[1 - Q_0(x)]}{\partial x} = f(x) > 0,$$
$$\frac{\partial[y_0 - \sigma(xQ_0(x) - Q_1(x))]}{\partial x} = -\sigma Q_0(x) < 0,$$

for any x in the support of ε. This implies, as functions of $(y_0 - m(x))/\sigma$, $g_1(x)$ and $g_2(x)$ are strictly monotone. On the other hand, since we have full observations on (X, Y), so nonparametric estimators for $g_1(x)$ and $g_2(x)$ can be easily constructed. These important observations motivate us to develop a three-step procedure to estimate m and σ^2 which is described below.

Algorithm 1:

Step 1: Estimate $g_1(x)$ nonparametrically, denote it as $\hat{g}_1(x)$; then for each X_i in the sample, estimate $(y_0 - m(X_i))/\sigma$ by $F^{-1}(\hat{g}_1(X_i))$ based on (2.2) and calculate

$$Z_i = -F^{-1}(\hat{g}_1(X_i)) \cdot Q_0\left(F^{-1}(\hat{g}_1(X_i))\right) + Q_1\left(F^{-1}(\hat{g}_1(X_i))\right).$$

Step 2: Estimate $g_2(x)$ nonparametrically, denote it as $\hat{g}_2(x)$. For each X_i in the sample, calculate $\hat{g}_2(X_i)$; Conduct a regression analysis without intercept of $\{\hat{g}_2(X_i) - y_0\}_{i=1}^n$ against $\{Z_i\}_{i=1}^n$ from Step 1, then estimate σ by the slope of this regression. Denote the estimator as $\hat{\sigma}$. This step is based on (2.3).

$$\widehat{\sigma} = \frac{\sum_{i=1}^n (W_i - \overline{W})(Z_i - \overline{Z})}{\sum_{i=1}^n (Z_i - \overline{Z})^2} \tag{2.4}$$

Step 3: Estimate $m(x)$ either by

$$\hat{m}(x) = y_0 - \hat{\sigma} F^{-1}(\hat{g}_1(x)) \tag{2.5}$$

based on inverting (2.2), or

$$\hat{m}(x) = y_0 - \hat{\sigma} H^{-1}(\hat{g}_2(x)) \tag{2.6}$$

based on inverting (2.3), where $H(x) = xQ_0(x) - Q_1(x)$.

There are many nonparametric smoothing procedures to estimate a regression function. Among which, the most popular one is the Nadaraya-Watson kernel estimate due to its simplicity; then it comes to the local linear estimator. The superiority of the latter to the former lies in the fact that the local linear does not suffer from the boundary effect. In our simulation study, we will use both to evaluate the finite sample performance of the proposed estimation procedure.

Sometimes, $m(x)$ is assumed to have a parametric form $m(x; \theta)$. In addition to many methods developed in the literature for this scenario, we provide another alternative method based the nonparametric estimator obtained from the above algorithm.

Algorithm 2:

Step 1: Estimate $m(x)$ using Algorithm 1.

Step 2: Estimate θ by minimizing some proper distance between the nonparametric estimator $\hat{m}(x)$ and the parametric regression function $m(x; \theta)$.

For example, in standard Tobit regression model, $m(x) = \alpha + \beta x$. We can estimate α and β by the intercept and slope, respectively, from the simple linear regression of $\{\hat{m}(X_i)\}_{i=1}^n$ against $\{X_i\}_{i=1}^n$. If $m(x, \theta)$ has a complicated nonlinear form, then one can estimate θ by the following minimum distance or empirical minimum distance procedures:

$$\hat{\theta}_n = \text{argmin}_\theta \int [\hat{m}(x) - m(x, \theta)] dW(x), \quad \hat{\theta}_n = \text{argmin}_\theta \sum_{i=1}^n [\hat{m}(X_i) - m(X_i, \theta)]^2.$$

The weight function $W(x)$ can be chosen to minimize the asymptotic variance of estimator $\hat{\theta}_n$. However, to do this, we need to develop some asymptotic theories of $\hat{\theta}$, which is beyond the scope the current report. So, for the sake of convenience, we will use the empirical minimum distance procedure to estimate the unknown parameter θ.

It is well known that the MLEs are generally most efficient among all the estimation procedures. Given the parametric form of the density function of ε and the regression function $m(x, \theta)$, the MLE should be available. The main advantage for Algorithm 2 really comes from its relatively simple computation. To obtain the MLE, we have to resort to some iteration algorithms, such as New-Raphson, EM etc. One also has to select initial values to start the iteration. But in our proposed method, after getting the nonparametric estimates of g_1, g_2, we only have to invert these estimates using F^{-1}, the inverse function of the CDF of ε, then applying a empirical linear or nonlinear LSE procedure.

In fact, the above idea can also be extended to the case in which the threshold value is unknown and thus needs to be estimated. Consider the following semi-parametric Tobit regression model:

$$Y^* = m(X) + \varepsilon; \quad Y = Y^* I(Y^* \geq \gamma) + y_0 I(Y^* < \gamma). \tag{2.7}$$

Except for assuming γ is known, other conditions stay the same as in model (2.1). Define g_1 and g_2 as before, we can obtain and $g_1(x) = E[I(Y = y_0)|X = x]$, $g_2(x) = E(YI(Y \neq$

$y_0)|X = x)$, then

$$g_1(x) = 1 - Q_0\left(\frac{\gamma - m(x)}{\sigma}\right), \qquad (2.8)$$
$$g_2(x) = y_0 - (y_0 - m(x))Q_0\left(\frac{\gamma - m(x)}{\sigma}\right) + \sigma Q_1\left(\frac{\gamma - m(x)}{\sigma}\right).$$

Rewrite the second equation as

$$g_2(x) = y_0 - \sigma\left[\frac{(y_0-\gamma)+(\gamma-m(x))}{\sigma}Q_0\left(\frac{\gamma - m(x)}{\sigma}\right) - Q_1\left(\frac{\gamma - m(x)}{\sigma}\right)\right]. \qquad (2.9)$$

By assumption (C), one can show that, for any fixed y_0, σ and γ, $1-Q_0(x)$ and $y_0-\sigma((c+x)Q_0(x)-Q_1(x))$, as functions of x, is strictly monotone if $y_0 \leq \gamma$, where $c = (y_0-\gamma)/\sigma < 0$ (which is intuitively reasonable, one should not assign a bigger value to y if it is a smaller value). In fact, note that

$$\frac{\partial Q_0(x)}{\partial x} = -f(x), \quad \frac{\partial Q_1(x)}{\partial x} = -xf(x),$$

Hence

$$\frac{\partial[1-Q_0(x)]}{\partial x} = f(x) > 0,$$
$$\frac{\partial[y_0 - \sigma((c+x)Q_0(x) - Q_1(x))]}{\partial x} = -\sigma[Q_0(x) - cf(x)] < 0.$$

This implies, as functions of $(y_0 - m(x))/\sigma$, $g_1(x)$ and $g_2(x)$ are strictly monotone. This important observation motivate us to develop a three-step procedure to estimate m, y_0 and σ^2. In fact, the following algorithm does not need the strict monotonicity of $g_2(x)$.

Algorithm 3:

Step 1: Estimate $g_1(x)$ nonparametrically, denote it as $\hat{g}_1(x)$; then for each X_i in the sample, estimate $(\gamma - m(X_i))/\sigma$ by $F^{-1}(\hat{g}_1(X_i))$ based on (2.8) and calculate

$$Z_{1i} = Q_0(F^{-1}(\hat{g}_1(X_i))),$$
$$Z_{2i} = -F^{-1}(\hat{g}_1(X_i)) \cdot Q_0\left(F^{-1}(\hat{g}_1(X_i))\right) + Q_1\left(F^{-1}(\hat{g}_1(X_i))\right).$$

Step 2: Estimate $g_2(x)$ nonparametrically, denote it as $\hat{g}_2(x)$. For each X_i in the sample, calculate $\hat{g}_2(X_i)$; Conduct a regression analysis without intercept of $\{\hat{g}_2(X_i) - y_0\hat{g}_1(X_i)\}_{i=1}^n$ against $\{Z_{1i}\}_{i=1}^n$ and $\{Z_{2i}\}_{i=1}^n$ from Step 1, then estimate γ by the slope of Z_1 and σ by the slope of Z_2 in this regression. Denote the estimators as $\hat{\gamma}$ and $\hat{\sigma}$. This step is based on (2.9).

Step 3: Estimate $m(x)$ by $\hat{m}(x) = \hat{\gamma} - \hat{\sigma}F^{-1}(\hat{g}_1(x))$.

We can use Carson and Sun's (2007) estimator $\hat{\gamma} = \min\{Y_i : Y_i \neq y_0\}$ to estimate γ, which has a faster convergence rate than the above estimator. Just modify the above algorithm appropriately to estimate σ. After obtaining the nonparametric estimate for $m(x)$, then we can estimate the parameters in a projected parametric regression function $m(x, \theta)$ using the similar methods as in the known y_0 case:

Algorithm 4:

Step 1: Estimate $m(x)$ by Algorithm 1.

Step 2: Estimate θ by conducting a regression analysis in which $\{X_i, \hat{m}(X_i)\}_{i=1}^n$ are observations, and $\{m(x, \theta)\}$ is the regression function.

Similar evaluation criteria can be used for checking the finite sample performance of the proposed estimates.

When the error term ε has a normal distribution and the regression function is linear, some existing procedures and programs can be used to implement the maximum likelihood estimation. The significance of the proposed methods in this report lies in the fact that when ε possesses other than normal distributions, and the regression function is nonlinear, then the proposed algorithm would provide a computationally effective way to obtain the estimation.

Chapter 3

Simulation Studies

Numerical Simulation studies will be conducted in this section to evaluate the finite sample performance of the proposed estimation procedure. The following setup will be used in the simulation. The data from the following two regression functions

$$m(x) = \alpha + \beta x, \quad m(x) = \alpha + \beta x + \gamma x^2, \tag{3.1}$$

where the true values of α, β and γ are chosen to be 1. Two threshold values are selected to be $y_0 = 0.5$ and $y_0 = 1$. The random error ε follows $N(0,1)$ or t-distribution with degrees of freedom 3, and the design variables X is chosen to have a normal $N(0,1)$ and uniform distribution. That is, we have 16 scenarios in total. Based on the true distributions of X, ε and the threshold value y_0, we can figure out the true truncation rate in each case. The following table presents the truncation rates for each scenario via simulation.

We shall use both Nadaraya-Watson kernel estimator and local linear estimator to estimate the regression function $g_1(x)$ and $g_2(x)$, the ksmooth and locpoly functions in R-package KernSmooth are used to implement the nonparametric estimation, with bandwidth

		$m(x) = 1 + x$		$m(x) = 1 + x + x^2$	
		$X \sim N(0,1)$	$X \sim U[-1,1]$	$X \sim N(0,1)$	$X \sim U[-1,1]$
$\varepsilon \sim N(0,1)$	$y_0 = 0.5$	36.18%	33.37%	19.56%	24.74%
	$y_0 = 1$	50.00%	50.00%	32.38%	40.22%
$\varepsilon \sim t(3)$	$y_0 = 0.5$	37.58%	35.20%	22.02%	27.19%
	$y_0 = 1$	50.00%	50.00%	33.75%	41.36%

chosen by default for Nadaraya-Watson estimator and the direct plugin one for local linear. The kernel function is chosen to be uniform. We also tried the normal kernel, and the simulation results are similar. So for the sake of brevity, we only report the simulation results from uniform kernel. The sample sizes are chosen to be $n = 100, 200$ and 500, and each simulation is replicated 200 times. For the nonparametric estimation for the regression function $m(x)$, we will illustrated the performance of the proposed estimation procedure by some fitting plots and the MSE calculated at observed X-values. As for the regression parameters, we will report the biases and MSEs of the minimum distance estimates (MDE).

3.1 Semi-Parametric Estimation of $m(x)$

For the sake of brevity, we only report the simulation results when $m(x) = 1 + x + x^2$, and $x \sim U(-1, 1)$, $\varepsilon \sim N(0, 1)$. In addition to the empirical MSE calculated with

$$MSE_1 = \frac{1}{n}\sum_{i=1}^{n}[\hat{m}(X_i) - m(X_i)]^2,$$

we also report the the empirical MSE obtained from

$$MSE_2 = \frac{1}{n}\sum_{i=1}^{n}[\hat{\alpha} - 1 + (\hat{\beta} - 1)X_i - (\hat{\gamma} - 1)X_i^2]^2.$$

Table 3.1 reports the simulation results when using kernel smoothing to estimate the function g_1 and g_2, and the bandwidth is selected by the dpill function from R-package KernSmooth. As we expected, when sample sizes increase, the MSEs are generally decreasing; the estimation based on (2.6) is better than the one based on (2.5). It might be interesting to notice that when truncation rate gets bigger, the performance of both estimators for all scenarios gets better! This seemingly confusing phenomenon indeed can be explained by the following observation: one has to rely on both (2.2) and (2.3) to obtain the final estimation, but estimating $(y_0 - m(x))/\sigma$ from (2.2) one need to calculate $F^{-1}(\hat{g}_1(x))$, but we conjecture that the asymptotic variance of this estimator will become very large if the truncation rate is too small or too large. Of course, the exact dependence of the asymptotic variance on the truncation rate should be investigated.

		(2.5)		(2.6)	
Sample Size	y_0	MSE_1	MSE_2	MSE_1	MSE_2
$n = 100$	0.5	0.1742	0.0892	0.1249	0.0718
	1	0.0842	0.0531	0.0607	0.0300
$n = 200$	0.5	0.1674	0.0457	0.0490	0.0353
	1	0.1081	0.0123	0.1014	0.0880
$n = 500$	0.5	0.1355	0.0080	0.0570	0.0312
	1	0.0513	0.0193	0.0417	0.0075

Table 3.1: Kernel, dpill bandwidth

Table 3.2 reports the simulation results when using local linear smoothing to estimate the function g_1 and g_2, and the bandwidth is selected by the dpill function from R-package KernSmooth. We can the similar patterns as the one shown in Table 3.2, but the simulation results show that using local linear smoothing is generally much better than using kernel smoothing, which is very well within our expectation due to the superiority of local smoothing to the kernel smoothing. Also, we see that using the fitted parametric form $\hat{\alpha} + \hat{\beta}x + \hat{\gamma}x^2$ is much better than using the direct nonparametric fit. To check the effect of bandwidth on the performance of the estimation procedure, we also conduced some simulation studies by choosing different bandwidth when applying kernel smoothing. Table 3.3 uses $h = 0.5n^{-1/5}$, where $n^{-1/5}$ is the optimal order for Nadaraya-Watson estimator under the MSE sense, and the choice 0.5 is somehow arbitrary and no particular theoretical or practical reason. Table 3.4 uses the default bandwidth value in function ksmooth. It is easy to see that the bandwidth section does have some effect on the estimation, but similar patterns as in Table 3.1

		(2.5)		(2.6)	
Sample Size	y_0	MSE_1	MSE_2	MSE_1	MSE_2
$n = 100$	0.5	0.1184	0.0807	0.0518	0.0264
	1	0.0479	0.0471	0.0377	0.0364
$n = 200$	0.5	0.0514	0.0137	0.0074	0.0033
	1	0.0178	0.0071	0.0258	0.0153
$n = 500$	0.5	0.0170	0.0117	0.0120	0.0084
	1	0.0269	0.0215	0.0164	0.0088

Table 3.2: Local Linear, dpill bandwidth

and 3.2 are kept.

		(2.5)		(2.6)	
Sample Size	y_0	MSE_1	MSE_2	MSE_1	MSE_2
$n = 100$	0.5	0.1978	0.0791	0.1218	0.0682
	1	0.1413	0.0667	0.0671	0.0398
$n = 200$	0.5	0.1756	0.0456	0.0519	0.0350
	1	0.1090	0.0122	0.0120	0.1050
$n = 500$	0.5	0.1475	0.0070	0.0610	0.0347
	1	0.0544	0.0205	0.0428	0.0076

Table 3.3: Kernel, $h = 0.5n^{-1/5}$

		(2.5)		(2.6)	
Sample Size	y_0	MSE_1	MSE_2	MSE_1	MSE_2
$n = 100$	0.5	0.1805	0.1118	0.0542	0.0496
	1	0.0528	0.0428	0.0427	0.0330
$n = 200$	0.5	0.1532	0.0359	0.0363	0.0319
	1	0.0149	0.0029	0.0201	0.0124
$n = 500$	0.5	0.0219	0.0165	0.0163	0.0123
	1	0.0320	0.0267	0.0159	0.0114

Table 3.4: Kernel, default bandwidth

For the sake of completeness, we also plots the fitted curves against the true quadratic curves using both kernel and local linear smoothing, and direct plug-in bandwidth. In the plots, the red curve is the true quadratic regression function; the black curve denotes the fitted curve using (2.2) and $\hat{m}(x)$; the cyan curve denotes the fitted curve using (2.2) and $\hat{\alpha} + \hat{\beta}x + \hat{\gamma}x^2$; the green curve denotes the fitted curve using (2.3) and $\hat{m}(x)$; the blue curve denotes the fitted curve using (2.3) and $\hat{\alpha}+\hat{\beta}x+\hat{\gamma}x^2$. From the left to the right, the plots are corresponding to sample sizes $n = 100, 200$ and 500. From the plots, we can see that local linear fitting provides us smoother estimates and smaller variability using the direct plug in bandwidth than the kernel fitting procedure. The effect of bandwidth on the performance of estimates is significant when the sample size is small, while improved performance can be seen for larger sample sizes.

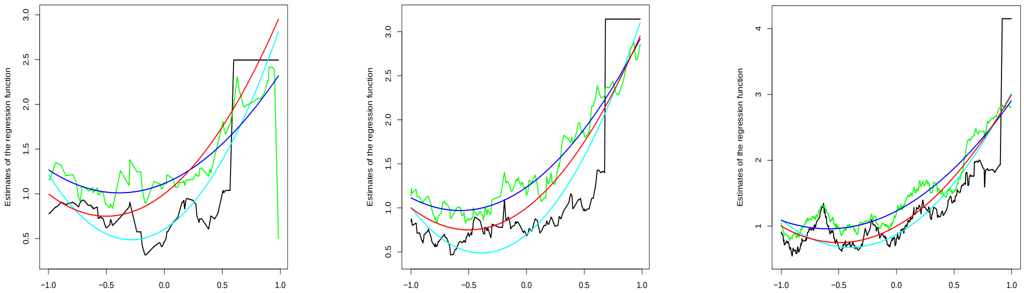

Figure 3.1: Kernel Smoothing, $y_0 = 0.5$

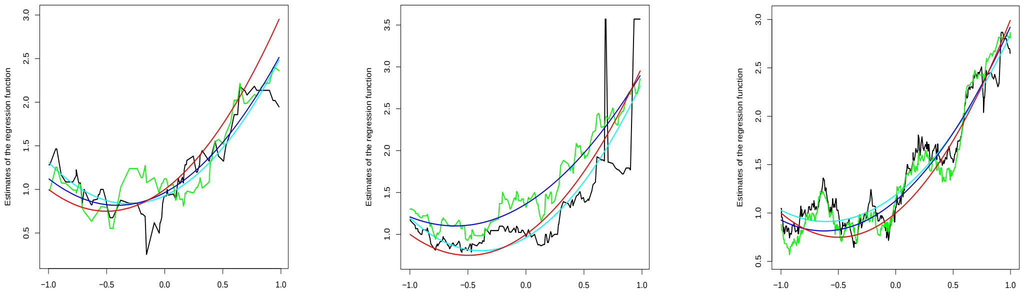

Figure 3.2: Kernel Smoothing, $y_0 = 1$

3.2 Biases and MSEs of the MDEs

This section reports the simulation results for the parameter estimations when the regression function is taking linear and quadratic forms. Of course, as we indicated earlier, the minimum distance estimating procedure also applies to other nonlinear function, while the computation will becomes complicated, since one has to solve a nonlinear minimization problem.

Table 3.5 and 3.6 reports the MSE and bias (the number in the parentheses) when the regression function is assumed to be linear $m(x) = \alpha + \beta x$ in which true values of α and β are all chosen to be 1, and Nadaraya-Watson kernel estimates are used for estimating g_1 and g_2. For all setups, the MLE performs best, as we expected. The simulation study does not show very much difference in general using either (2.5) or (2.6) to estimate the regression function. Similar pattern appears in other setup.

Table 3.7 and 3.8 reports the MSE and bias (the number in the parentheses) when the

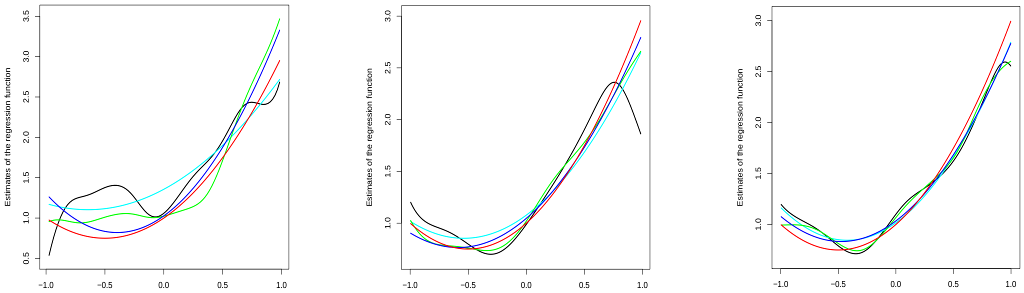

Figure 3.3: Local Linear Smoothing, $y_0 = 0.5$

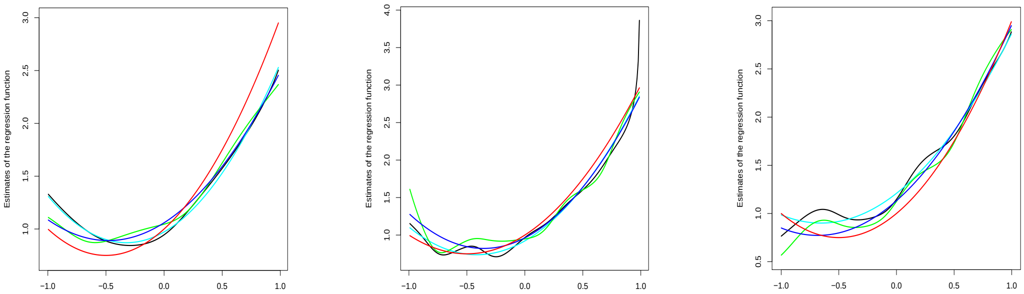

Figure 3.4: Local Linear Smoothing, $y_0 = 1$

		$n = 100$	$n = 200$	$n = 300$
α	(2.5)	0.0144(-0.0293)	0.0061(-0.0162)	0.0023(-0.0074)
	(2.6)	0.0254(0.0576)	0.0277(-0.0825)	0.0028(0.0009)
	MLE	0.0131(0.0126)	0.0055(-0.0015)	0.0021(-0.0041)
β	(2.5)	0.0501(-0.1106)	0.0092(0.0202)	0.0100(-0.0448)
	(2.6)	0.0566(-0.0803)	0.0267(-0.0732)	0.0097(-0.0452)
	MLE	0.0460(0.0083)	0.0185(-0.0117)	0.0066(0.0021)
σ	(2.4)	0.0756(-0.1354)	0.0228(-0.0543)	0.0037(-0.0113)
	MLE	0.0086(-0.0134)	0.0045(-0.0059)	0.0015(-0.0078)

Table 3.5: Kernel, Linear, $y_0 = 0.5$

regression function is assumed to be quadratic $m(x) = \alpha + \beta x + \gamma x^2$ in which true values of α, β and γ are all chosen to be 1, and Nadaraya-Watson kernel estimates are used for estimating g_1 and g_2.

Table 3.9 and 3.10 reports the MSE and bias (the number in the parentheses) when the regression function is assumed to be quadratic $m(x) = \alpha + \beta x$ in which true values of α, β and γ are all chosen to be 1, and local linear estimates are used for estimating g_1 and g_2.

		$n = 100$	$n = 200$	$n = 300$
α	(2.5)	0.0143(0.0173)	0.0093(0.0013)	0.0036(-0.0069)
	(2.6)	0.0343(0.0540)	0.0139(0.0167)	0.0051(-0.0039)
	MLE	0.0141(0.0195)	0.0086(0.0031)	0.0033(-0.0042)
β	(2.5)	0.0541(-0.0951)	0.0226(-0.0347)	0.0137(-0.0531)
	(2.6)	0.0588(-0.0672)	0.0301(-0.0309)	0.0139(-0.0451)
	MLE	0.0372(-0.0036)	0.0224(0.0198)	0.0099(-0.0090)
σ	(2.4)	0.0472(-0.0893)	0.0130(-0.0222)	0.0041(-0.0015)
	MLE	0.0119(-0.0207)	0.0059(-0.0016)	0.0023(-0.0052)

Table 3.6: Kernel, Linear, $y_0 = 1$

		$n = 100$	$n = 200$	$n = 300$
α	(2.5)	0.0792(-0.1671)	0.0581(-0.1113)	0.0200(-0.0269)
	(2.6)	0.0514(0.1658)	0.0285(0.1120)	0.0133(0.0700)
	MLE	0.0258(0.0175)	0.0121(-0.0057)	0.0045(-0.0061)
β	(2.5)	0.0425(-0.0606)	0.0250(-0.0677)	0.0128(-0.0630)
	(2.6)	0.0504(-0.1189)	0.0255(-0.0836)	0.0114(-0.0579)
	MLE	0.0405(0.0062)	0.0149(0.0099)	0.0057(0.0016)
γ	(2.5)	0.2312(0.0829)	0.1597(-0.0021)	0.0692(-0.0914)
	(2.6)	0.1276(-0.2293)	0.0854(-0.2182)	0.0501(-0.1784)
	MLE	0.1240(-0.0199)	0.0550(-0.0087)	0.0217(0.0072)
σ	(2.4)	0.1447(-0.3094)	0.0873(-0.1900)	0.0268(-0.0732)
	MLE	0.0071(-0.0168)	0.0041(-0.0071)	0.0014(-0.0094)

Table 3.7: Kernel, Quadratic, $y_0 = 0.5$

		$n = 100$	$n = 200$	$n = 300$
α	(2.5)	0.0263(0.0029)	0.0191(0.0472)	0.0085(0.0405)
	(2.6)	0.0741(0.1510)	0.0393(0.0847)	0.0125(0.0454)
	MLE	0.0228(0.0072)	0.0148(0.0141)	0.0059(0.0049)
β	(2.5)	0.0554(-0.1326)	0.0302(-0.0857)	0.0127(-0.0637)
	(2.6)	0.0589(-0.0796)	0.0252(-0.0405)	0.0110(-0.0345)
	MLE	0.0373(0.0171)	0.0162(0.0091)	0.0079(-0.0094)
γ	(2.5)	0.1502(-0.1242)	0.1132(-0.1843)	0.0618(-0.1777)
	(2.6)	0.1253(-0.2207)	0.0991(-0.1837)	0.0601(-0.1909)
	MLE	0.1046(0.0020)	0.0731(-0.0013)	0.0254(-0.0203)
σ	(2.4)	0.1139(-0.1849)	0.0434(-0.0490)	0.0085(0.0090)
	MLE	0.0099(-0.0153)	0.0061(-0.0062)	0.0021(-0.0079)

Table 3.8: Kernel, Quadratic, $y_0 = 1$

		$n = 100$	$n = 200$	$n = 300$
α	(2.5)	0.0136(-0.0423)	0.0066(-0.0301)	0.0030(-0.0111)
	(2.6)	0.0304(0.0862)	0.0135(0.0404)	0.0048(0.0206)
	MLE	0.0128(0.0182)	0.0055(-0.0005)	0.0026(0.0013)
β	(2.5)	0.0536(-0.1004)	0.0245(-0.0544)	0.0086(-0.0182)
	(2.6)	0.0557(-0.0788)	0.0300(-0.0519)	0.0087(-0.0220)
	MLE	0.0444(-0.0069)	0.0209(-0.0160)	0.0067(-0.0043)
σ	(2.4)	0.0936(-0.1841)	0.0427(-0.0977)	0.0107(-0.0393)
	MLE	0.0090(-0.0140)	0.0039(-0.0041)	0.0018(-0.0065)

Table 3.9: Local Linear, Linear, $y_0 = 0.5$

		$n = 100$	$n = 200$	$n = 300$
α	(2.5)	0.0143(0.0284)	0.0089(0.0134)	0.0033(0.0031)
	(2.6)	0.0512(0.1081)	0.0291(0.0610)	0.0067(0.0124)
	MLE	0.0148(0.0221)	0.0077(0.0094)	0.0030(0.0036)
β	(2.5)	0.0628(-0.1340)	0.0358(-0.0678)	0.0102(-0.0064)
	(2.6)	0.0680(-0.1094)	0.0392(-0.0470)	0.0117(0.0083)
	MLE	0.0389(-0.0297)	0.0212(-0.0112)	0.0090(0.0040)
σ	(2.4)	0.0818(-0.1509)	0.0428(-0.0886)	0.0081(-0.0118)
	MLE	0.0113(-0.0175)	0.0065(-0.0132)	0.0026(-0.0044)

Table 3.10: Local Linear, Linear, $y_0 = 1$

		$n = 100$	$n = 200$	$n = 300$
α	(2.5)	0.0762(-0.1821)	0.0659(-0.1616)	0.0346(-0.0937)
	(2.6)	0.0490(0.1596)	0.0256(0.1068)	0.0142(0.0771)
	MLE	0.0258(0.0220)	0.0121(-0.0063)	0.0054(0.0000)
β	(2.5)	0.0497(-0.0216)	0.0294(-0.0305)	0.0124(-0.0322)
	(2.6)	0.0458(-0.0591)	0.0199(-0.0444)	0.0092(-0.0401)
	MLE	0.0405(0.0108)	0.0147(0.0104)	0.0065(-0.0052)
γ	(2.5)	0.2616(0.2446)	0.2018(0.1902)	0.1034(0.1120)
	(2.6)	0.1293(-0.1308)	0.0644(-0.0931)	0.0282(-0.0606)
	MLE	0.1236(-0.0203)	0.0555(-0.0072)	0.0256(-0.0025)
σ	(2.4)	0.1071(-0.2757)	0.0768(-0.2111)	0.0388(-0.1239)
	MLE	0.0068(-0.0156)	0.0041(-0.0068)	0.0015(-0.0060)

Table 3.11: Local Linear, Quadratic, $y_0 = 0.5$

Table 3.11 and 3.12 reports the MSE and bias (the number in the parentheses) when the regression function is assumed to be quadratic $m(x) = \alpha + \beta x + \gamma x^2$ in which true values of α, β and γ are all chosen to be 1, and local linear estimates are used for estimating

g_1 and g_2. It is interesting to notice that for estimating the parameters in the regression function, local linear estimate does not show overall superiority over the Nadaraya-Watson kernel estimate.

Based on the simulation studies, we can see that if the error term ε has a normal distribution, then the MLE procedure is the most efficient one. The merit of the proposed methodology is its computational effectiveness when the ε has distributions other than normal or the regression function is nonlinear, since in such cases, the likelihood function would be very complicated.

		$n = 100$	$n = 200$	$n = 300$
α	(2.5)	0.0256(-0.0321)	0.0172(-0.0084)	0.0089(-0.0135)
	(2.6)	0.0902(0.2132)	0.0554(0.1575)	0.0294(0.1007)
	MLE	0.0235(0.0076)	0.0144(0.0108)	0.0062(-0.0017)
β	(2.5)	0.0540(-0.1343)	0.0312(-0.0979)	0.0152(-0.0684)
	(2.6)	0.0545(-0.0880)	0.0267(-0.0542)	0.0147(-0.0389)
	MLE	0.0339(0.0104)	0.0166(0.0132)	0.0067(0.0021)
γ	(2.5)	0.1449(0.0320)	0.0929(-0.0037)	0.0410(0.0096)
	(2.6)	0.1295(-0.1419)	0.0847(-0.1072)	0.0363(-0.0690)
	MLE	0.0958(0.0100)	0.0730(-0.0033)	0.0300(0.0059)
σ	(2.4)	0.1405(-0.2808)	0.0807(-0.1813)	0.0444(-0.1112)
	MLE	0.0103(-0.0174)	0.0059(-0.0067)	0.0019(-0.0042)

Table 3.12: Local Linear, Quadratic, $y_0 = 1$

Bibliography

[1] Amemiya, T. (1973). Regression analysis when the dependent variable is truncated normal. Econometrica 41 997-1016.

[2] Amemiya, T. (1984). Tobit models: a survey. J. of Econometrics 24(1-2) 3-61.

[3] Buckley, J. and James, I. (1979). Linear regression with censored data. Biometrics, 66, 429-436.

[4] Carson, R. T. and Sun, Y. (2007). The Tobit model with a non-sero threshold. Econometrics Journal 10, 488-502

[5] Goldberger, A. S.(1964), Econometric theory, PP. 251-255.

[6] Guardiola(2012), A Robust Tobit Regression Model When Errors Are from the Epsilon Skew Exponential Power Family.

[7] Heckman, J. (1976). The common structure of statistical models of truncation, sample selection, and limited dependent variables and a simple estimator for such models. Ann. Econom. Social Meas. 5 475-492.

[8] Heckman, J. (1979). Sample bias as a specification error. Econometrica 47 153-162.

[9] Honore, B. E. and Powell, J.L. (1994). Pairwise Difference Estimators for Censored and Truncated Regression Models. Semiparametric Censored Regression Models. Journal of Econometrics. 64(1-2), 24178.

[10] Horowitz, J.L., Neumann, G.R. (1989). Specification testing in censored regression models: parametric and semi-parametric methods. J. Appl. Econometrics 4 61-86.

[11] Koul, H., Susarla, V. and Ryzin, J.V. (1981). Regression Analysis with Randomly Right-Censored Data. Ann. Statist. 9(6), 1276-1288.

[12] Lewbel, A., & Linton, O. B. (2002). Nonparametric censored and truncated regression. Econometrica, 70, 765-779.

[13] Maddala, G. S. (1983). Limited-dependent and qualitative variables in econometrics. Cambridge University Press.

[14] McConnel, C. E., Zetzman, M. R. (1993). Urban/rural differences in health service utilization by elderly persons in the United States. J. Rural Health, 9(4), 270-280

[15] Olsen, R.J. (1978). note on the uniqueness of maximum likelihood estimation for the tobit model . Econometrica 46, 1211-1215

[16] Powell, J.L. (1984). Least absolute deviations estimation for the censored regression model. J. Econometrics 25 303-325.

[17] Powell, J.L. (1986a). Censored Regression Quantiles. Journal of Econometrics. 32(1), 14355.

[18] Powell, J.L. (1986b). Symmetrically Trimmed Least Squares Estimation for Tobit Models. Econometrica, 54, 1435-1460.

[19] Powell, J.L., J.H. Stock and T.M. Stoker (1989). Semiparametric estimation of weighted average derivatives. Econometrica 57, 1403-1430.

[20] Tobin, J. (1958). Estimation of relationships for limited dependent variables. Econometrica, 26(1), 24-36.

[21] Wales, T.J. and Woodland, A.D. (1980). Sample selectivity and the esitmation of labor supply functions. International Economic Review. 21, 437-468.

[22] Wand, M.P., & Jones M.C. (1995). Kernel Smoothing Chapman Hall

[23] Yvette Y. Z., Li, Q. and Li, D. (2011). Recent Developments in Semi-/Non-Parametric Estimation of Censoring, Sample Selection, Missing Data, and Measurement Error in Panel Data. http://agecon2.tamu.edu/people/faculty/zhang-yvette/AIE0510.pdf.

[24] Zhou, X. B. (2007). Semi-parametric and Nonparametric Estimation of Toibt Models. PhD thesis, Department of Economics, Hong Kong University of Science and Technology.

Appendix A

R-Programs

This section includes the R programs used in the simulation studies.

Table 3.5, **3.6**,**3.7**, **3.8** verbatim

```
# Nadaraya-Watson Estimate

 library("KernSmooth")
 library("censReg")
 set.seed(987654)
 result=array(0,dim=c(6,8,2));
 total=200;
 kk=1;
 for(y0 in c(0.5,1))
  {
   jj=1;
   for(n in c(100,200,500))
     {
     sig.est=aid.est=bid.est=ay.est=by.est=amle=bmle=smle=rep(0,total)
     for(i in seq(total))
       {
```

```
repeat
{
 x=runif(n,-1,1); # Uniform Design
 #x=rnorm(n,0,1); # Normal Design
 ystar=1+x+rnorm(n); # Linear Regression
 ystar=1+x+x^2+rnorm(n,0,1); # Quadratic Regression
 y=pmax(ystar,y0)
 yid=(y==y0);
# First Step: Estimate [y0-m(x)]/sig
  kest.id=ksmooth(x,yid,kernel="box",x.points=x);
  kest.yid=kest.id$y;
  temp=kest.yid+10^(-6);
  temp1=qnorm(temp);  # for uniform design
# Second Step: Estimate sigma
  xtemp=-(1-kest.yid)*temp1+dnorm(temp1)
  kest=ksmooth(x,y,kernel="box",x.points=x);
  kest.y=kest$y;
  kest.y0=kest.y-y0;
  regid=lm(kest.y0~xtemp-1)
  sig.est[i]=coef(regid)[1];
# Third Step: Estimate m(x), using E[I(Y=y0)|X], equation 2.1
  mx=y0-sig.est[i]*temp1;
# Fourth Step: Estimate a, b  using L2 minimum distance
  regab=lm(mx~sort(x))
  aid.est[i]=coef(regab)[1]
  bid.est[i]=coef(regab)[2]
# Third Step: Estimate m(x), using E(Y|X), equation 2.2
```

```
   resy=(y0-kest.y)/sig.est[i]
   lv=-5+5*pnorm(-5)-dnorm(-5)-resy
   rv=5-5*pnorm(5)-dnorm(5)-resy
   if(all(lv*rv<0)) break;
}
   mest=rep(0,length(x));
   for(k in seq(n))
     {
       g=function(z){z-z*pnorm(z)-dnorm(z)-resy[k]}
       mest[k]=uniroot(g,c(-5,5))$root
     }
   mx=y0-sig.est[i]*mest;
 # Fourth Step: Estimate a, b
   regab=lm(mx~sort(x))
   ay.est[i]=coef(regab)[1]
   by.est[i]=coef(regab)[2]
 # Maximum Likelihood Estimate Based on Normal Error
   myreg=censReg(y~x,left=y0, right=Inf)
     amle[i]=coef(myreg)[1]
     bmle[i]=coef(myreg)[2]
     smle[i]=exp(coef(myreg)[3])
   }
Mu=c(mean(aid.est-1), mean(bid.est-1),mean(ay.est-1),
     mean(by.est-1),mean(sig.est-1),
     mean(amle-1),mean(bmle-1),mean(smle-1))
Ms=c(mean((aid.est-1)^2),mean((bid.est-1)^2),mean((ay.est-1)^2),
     mean((by.est-1)^2), mean((sig.est-1)^2),mean((amle-1)^2),
           mean((bmle-1)^2),mean((smle-1)^2))
       result[c((jj-1)*2+1,jj*2),1:8,kk]=rbind(Mu,Ms)
       jj=jj+1;
      }
      kk=kk+1
    }
 dimnames(result)=list(c("n=100, bias","n=100, mse","n=200, bias",
          "n=200, mse","n=500, bias","n=500, mse"),
          c("alpha, id","beta, id","alpha, py","beta, py",
          "sig","alpha, MLE","beta, MLE","sig, MLE"),c("y0=0.5","y0=1"))
 result
```

Table 3.9, 3.10, 3.11 and 3.12

```
library("KernSmooth")
library("censReg") set.seed(987654)
result=array(0,dim=c(6,8,2));
total=200;
kk=1;
for(y0 in c(0.5,1))
 {
  jj=1;
  for(n in c(100,200,500))
   {
    sig.est=aid.est=bid.est=ay.est=by.est=amle=bmle=smle=rep(0,total)
    for(i in seq(total))
     {
      repeat{
```

```
x=runif(n,-1,1);
ystar=1+x+rnorm(n,0,1); # Linear Regression
#ystar=1+x++x^2+rnorm(n,0,1); # Quadratic Regression
y=pmax(ystar,y0)
yid=(y==y0);   # Indicator of y=y0;
# First Step: Estimate [y0-m(x)]/sig
h=dpill(x, y)
lest=locpoly(x,yid,bandwidth=h,gridsize=n)
fitfn=approxfun(lest$x, lest$y)
lest.yid=fitfn(x)
# Estimate of [y0-m(x)]/sig temp=lest.yid+10^(-6);
temp[temp<0]=10^(-6);
temp1=qnorm((1-10^(-6))*(temp>=1)+temp*(temp<1)); # for normal design
#temp1=qnorm(temp);  # for uniform design
#if(any(is.nan(temp1))){break}
# Second Step: Estimate y0 and sig xtemp=-(1-lest.yid)*temp1+dnorm(temp1)
lest=locpoly(x,y,bandwidth=h,gridsize=n)
fitfn=approxfun(lest$x, lest$y)
lest.y=fitfn(x) lest.y0=lest.y-y0;
regid=lm(lest.y0~xtemp-1)
sig.est[i]=coef(regid)[1];
# Third Step: Estimate m(x), using E[I(Y=y0)|X]
mx=y0-sig.est[i]*temp1;
# Fourth Step: Estimate a, b
regab=lm(mx~x)
aid.est[i]=coef(regab)[1]
bid.est[i]=coef(regab)[2]
```

```
 ## Third Step: Estimate m(x), using E(Y|X)
 resy=(y0-lest.y)/sig.est[i]
 lv=-5+5*pnorm(-5)-dnorm(-5)-resy
 rv=5-5*pnorm(5)-dnorm(5)-resy
 if(all(lv*rv<0)) break;
}
 mest=rep(0,length(x)); for(k in seq(n))
   {
     g=function(z){z-z*pnorm(z)-dnorm(z)-resy[k]}
     mest[k]=uniroot(g,c(-5,5))$root
     k=k+1
   }
 mx=y0-sig.est[i]*mest;
 # Fourth Step: Estimate a, b
 regab=lm(mx~x)
 ay.est[i]=coef(regab)[1]
 by.est[i]=coef(regab)[2]
 # Maximum Likelihood Estimate Based on Normal Error
 myreg=censReg(y~x,left=y0, right=Inf)
 amle[i]=coef(myreg)[1]
 bmle[i]=coef(myreg)[2]
 smle[i]=exp(coef(myreg)[3])
  }
 Mu=c(mean(aid.est-1), mean(bid.est-1),mean(ay.est-1),mean(by.est-1),
    mean(sig.est-1),mean(amle-1),mean(bmle-1),mean(smle-1))
 Ms=c(mean((aid.est-1)^2),mean((bid.est-1)^2),mean((ay.est-1)^2),
    mean((by.est-1)^2),mean((sig.est-1)^2),mean((amle-1)^2),
        mean((bmle-1)^2),mean((smle-1)^2))
     result[c((jj-1)*2+1,jj*2),1:8,kk]=rbind(Mu,Ms)
     jj=jj+1;
    }
     kk=kk+1
  } dimnames(result)=list(c("n=100, bias","n=100, mse","n=200, bias",
                          "n=200, mse","n=500, bias","n=500, mse"),
                        c("alpha, id","beta, id","alpha, py","beta, py",
          "sig","alpha, MLE","beta, MLE","sig, MLE"),c("y0=0.5","y0=1"))
  round(result,4)
```

Table 3.1, Table 3.2, Figure 3.1 and Figure 3.2

```
library("KernSmooth")
set.seed(987654)
# Generate Sample
# Threshold y0=1; # Standard Deviation
  sig=1; # Sample Size
  n=500; # Sample Gegeration
   x=runif(n,-1,1);
   # x=rnorm(n);
   ystar=1+x+x^2+rnorm(n,0,sig);
   y=pmax(ystar,y0)
   yid=(y==y0);
   # First Step: Estimate [y0-m(x)]/sig
     #h=0.5*n^(-1/5)
     #h=dpill(x, y)
     #kest.yid=ksmooth(x,yid,kernel="box",bandwidth=h,x.points=x)$y;
     kest.yid=ksmooth(x,yid,kernel="box",x.points=x)$y;
```

```
# Estimate of [y0-m(x)]/sig temp=kest.yid+10^(-6);
  #temp1=qnorm((1-10^(-6))*(temp>=1)+temp*(temp<1)); # for normal design
  temp1=qnorm(temp);  # for uniform design
# Second Step: Estimate y0 and sig
  xtemp=-(1-kest.yid)*temp1+dnorm(temp1)
  #kest.y=ksmooth(x,y,kernel="box",bandwidth=h,x.points=x)$y;
           kest.y=ksmooth(x,y,kernel="box",x.points=x)$y;
  kest.y0=kest.y-y0;
  regid=lm(kest.y0~xtemp-1)
  sig.est=coef(regid)[1];
# Third Step: Estimate m(x), using E[I(Y=y0)|X]
  mx=y0-sig.est*temp1; mxid=mx;
  # Fourth Step: Estimate a, b, c x=sort(x);
  x2=x^2;
  regab=lm(mx~x+x2)
  aid=coef(regab)[1]
  bid=coef(regab)[2]
  cid=coef(regab)[3]
  mxidp=predict(regab)
# Third Step: Estimate m(x), using E(Y|X)
  resy=(y0-kest.y)/sig.est
  mest=rep(0,length(x));
  for(k in seq(n))
     {
      g=function(z){z-z*pnorm(z)-dnorm(z)-resy[k]}
      mest[k]=uniroot(g,c(-5,5))$root
     }
```

```
 mx=y0-sig.est*mest;
 mxy=mx
 # Fourth Step: Estimate a, b
 regab=lm(mx~x+x2)
 ay=coef(regab)[1]
 by=coef(regab)[2]
 cy=coef(regab)[3]
 mxyp=predict(regab)
 fv=1+x+x^2;  # Ture regression function
 lend=min(cbind(mxid,mxidp,mxy,mxyp,fv))-0.05
 rend=max(cbind(mxid,mxidp,mxy,mxyp,fv))+0.05
plot(x,mxid,type="l",lwd=2,ylim=c(lend,rend),xlab="",
      ylab="Estimates of the regression function")
     # Estimate of m(x) from (eq2)
 lines(x,mxidp,col="cyan",lwd=2)
     # Estimate of m(x) by linear regression with (eq2) as response
 lines(x,mxy,col="green",lwd=2)
     #   Estimate of m(x) from (eq3)
 lines(x,mxyp,col="blue",lwd=2)
     # Estimate of m(x) by linear regression with (eq3) as response
 lines(x,fv,col="red",lwd=2) # True regression
 ms.mxid=mean((mxid-fv)^2)
 ms.mxidp=mean((mxidp-fv)^2)
 ms.mxy=mean((mxy-fv)^2)
 ms.mxyp=mean((mxyp-fv)^2)
 cbind(ms.mxid,ms.mxidp,ms.mxy,ms.mxyp)
```

Table 3.3, Table 3.4, Figure 3.3 and Figure 3.4

```
set.seed(987654)
# Generate Sample
repeat{
  repeat{
# Threshold
  y0=1;
# Standard Deviation
  sig=1;
# Sample Size
  n=5000;
# Sample Gegeration
  x=runif(n,-1,1);
  ystar=1+x+x^2+rnorm(n,0,sig);
  y=pmax(ystar,y0)
  yid=(y==y0);   # Indicator of y=y0;
# First Step: Estimate [y0-m(x)]/sig
  # Local Linear Smoothing of Indicator versus X
    h=dpill(x, y)
    lest.xid=locpoly(x,yid,bandwidth=h,gridsize=n)$x
    lest.yid=locpoly(x,yid,bandwidth=h,gridsize=n)$y
  # Estimate of [y0-m(x)]/sig
    temp1=qnorm(lest.yid)
    if(all(temp1!="NaN")) break;
    }
  # Second Step: Estimate y0 and sig
    xtemp=-(1-lest.yid)*temp1+dnorm(temp1)
    lest.x=locpoly(x,y,bandwidth=h,gridsize=n)$x
```

```
   lest.y=locpoly(x,y,bandwidth=h,gridsize=n)$y
   lest.y0=lest.y-y0;
   regid=lm(lest.y0~xtemp-1)
   sig=coef(regid)[1];
 # Third Step: Estimate m(x), using E[I(Y=y0)|X]
   mx=y0-sig*temp1;
   mxid=mx;
 # Fourth Step: Estimate a, b
   lest.xid2=lest.xid^2;
   regab=lm(mx~lest.xid+lest.xid2)
   mxidp=predict(regab);
 # Third Step: Estimate m(x), using E(Y|X)
  resy=(y0-lest.y)/sig
  lv=-5+5*pnorm(-5)-dnorm(-5)-resy
  rv=5-5*pnorm(5)-dnorm(5)-resy
  if(all(lv*rv<0)) break;
  }
  mest=rep(0,length(x));
  for(k in seq(n))
    {
     g=function(z){z-z*pnorm(z)-dnorm(z)-resy[k]}
     mest[k]=uniroot(g,c(-5,5))$root
     k=k+1
    }
   mx=y0-sig*mest;
   mxy=mx;
# Fourth Step: Estimate a, b
```

```
lest.x2=lest.x^2
regab=lm(mx~lest.x+lest.x2)
mxyp=predict(regab)
  # Ture regression function
fv=1+lest.x+lest.x^2
lend=min(cbind(mxid,mxidp,mxy,mxyp,fv))-0.05
rend=max(cbind(mxid,mxidp,mxy,mxyp,fv))+0.05
plot(lest.x, mxid,type="l",lwd=2,ylim=c(lend,rend),xlab=""
     ,ylab="Estimates of the regression function")
     # Estimate of m(x) from (eq2)
lines(lest.x, mxidp,col="cyan",lwd=2)
  # Estimate of m(x) by linear regression with (eq2) as response
lines(lest.x, mxy,col="green",lwd=2)
  #   Estimate of m(x) from (eq3)
lines(lest.x,mxyp,col="blue",lwd=2)
 # Estimate of m(x) by linear regression with (eq3) as response
lines(lest.x,fv,col="red",lwd=2) # True regression
ms.mxid=mean((mxid-fv)^2)
ms.mxidp=mean((mxidp-fv)^2)
ms.mxy=mean((mxy-fv)^2)
ms.mxyp=mean((mxyp-fv)^2)
cbind(ms.mxid,ms.mxidp,ms.mxy,ms.mxyp)
```