A COMPARISON STUDY ON THE ESTIMATION IN TOBIT REGRESSION MODELS

by

ANTOINETTE LEIKER

B.S., Fort Hays State University, 2009

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

2012

Approved by:

Major Professor
Dr. Weixing Song

Abstract

The goal of this report is to compare various estimation procedures on regression models in which the dependent variable has a restricted range. These models, called Tobit models, are seeing an increase in use among economists and market researchers, specifically. Only the standard Tobit regression model is discussed in the report.

First we will examine the five estimation methods discussed in Amemiya (1984) for standard Tobit model. These methods include Probit maximum likelihood, least squares, Heckman's two-step, Tobit maximum likelihood, and the EM algorithm. We will examine the algorithm utilized in each method's estimation process.

We will then conduct simulation studies using these estimation procedures. Twelve scenarios have been considered consisting of three different truncation threshold on the response variable, two distributions of
covariates, and the error variance known and unknown. The results are reported and a discussion of the goodness of each method follows.

The study shows that the best method for estimating Tobit regression models is indeed the Tobit maximum likelihood estimation. Heckman's two-step method and the EM algorithm also estimate these models well when the truncation rate is low and the sample size is large. The simulation results show that the Least squares estimation procedure is far less efficient than other estimation procedures.

Table of Contents

List of Tables

List of Figures

Acknowledgements

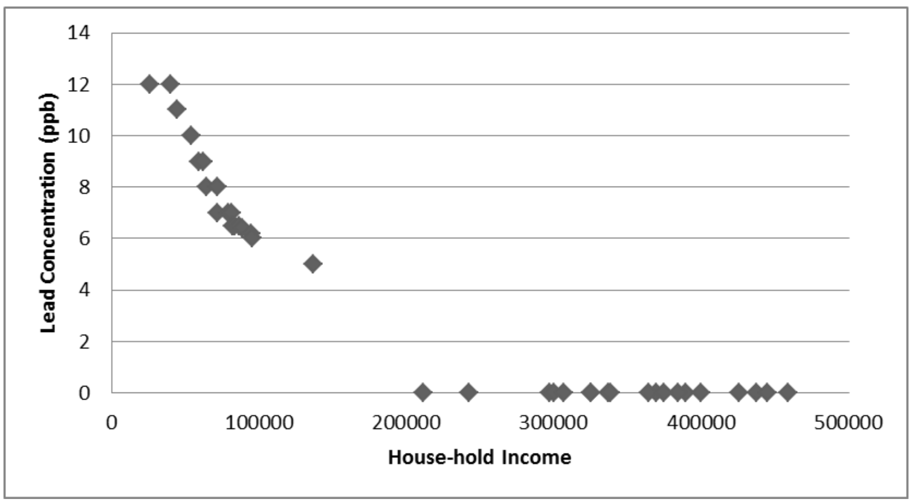
I would like to thank my major advisor, Dr. Weixing Song, for his time and effort over the last year. He has had tremendous patience as I have been working professionally and writing my paper simultaneously. The many emails he answered throughout the process must be in the hundreds.

I would also like to thank my committee, Dr. Juan Du and Dr. Gary Gadbury, for their efforts, opinions and time. My time at K-State has been one of learning, maturation, and professional growth. I have the entire faculty and staff of the Statistics department to thank for this.

Chapter 1 – Introduction

Consider a research project in which the level of lead in drinking water is being analyzed as a function of house-hold income. Most lead-testing kits have a threshold on minimum detectable concentration levels, say five parts per billion. Thus, any value below 5 ppb will read as a 0. In figure 1.1 one can see that the observations where the lead concentration is greater than 5 ppb could easily be modeled linearly, but the observations below the 5 ppb threshold are unusable. This is an example of left-censoring, or censoring from below.

Figure 1.1



Tobin (1958) noted the relationship between household expenditures on a durable good and household income are similarly distributed and cannot be simply modeled as a linear regression due to the characteristic that several observations on expenditure are zeros. He developed a model to adjust for this censoring. In a 1964 paper, Goldberg names Tobin's model the Tobit model because of its similarities to Probit models.

Consider the previous examples. To be specific, assume that a response variable y^* and a predictor X, possibly multidimensional, can be modeled as $y^* = m(X) + \epsilon$, where m is the regression function $E(y^*|X)$, and ϵ is the random error. In Tobit regression model, y^* can only be observed if its value is above a threshold y_0, which is often assumed to be known, or one can observe $Y =$

$\max\{y^*, y_0\}$. The classical Tobit regression model assumes that $m(x) = \beta_0 + X_i'\beta_1$, and the random error ϵ follows a normal distribution $N(0, \sigma^2)$.

Since the 1960s, the applications of Tobit regression models have increased dramatically. The value of these models has led to various research areas such as economics, biometrics, agriculture, psychology, sociology and medicine to incorporate Tobit regression in their respective fields. Shishko and Rostker (1980) utilized tobit regression in labor studies—determining the probability a full-time employee moonlights (works a second job) as well as estimating the number of hours worked at the second job. Delva and associates (2006) employed tobit regression in an analysis of the association of youth alcoholism with depression and parental factors in Korea. This study examines the extent to which depressive symptoms, parental alcoholism and parental attention predict or explain adolescent drinking behaviors. Tobit regression is an appropriate method due to the large number of adolescents who didn't exhibit issues with alcohol and thus creating a cluster of "zero" observations.

Other examples can be found in Ekstrand and Carpenter (1998), Smith and Brame (2003), Holden (2004), Wang (2007), Caudill and Mixon (2009), Solon (2010), and the references therein.

In the classic Tobit regression model, the statistical inference mainly focuses on the estimation of the regression parameter β and the variance σ^2. Assuming that ϵ follows a normal distribution $N(0, \sigma^2)$, one can use the Probit maximum likelihood to find a consistent and asymptotic normally distributed estimate for β_0/σ and β_1/σ. However, one cannot estimate the regression parameters and standard deviation separately; naive least square estimation by simply regressing y linearly on x produces biased estimates, but the bias can be corrected by nonlinear regression. The log-likelihood function of the Tobit regression model is not globally concave with respect to the original parameters β and σ, see Amemiya (1973). After certain reparametrization, Olsen (1978) showed that the log-likelihood function in a reparameterized Tobit model is globally concave, which implies that a standard iterative method such as the Newton-Raphson or Fisher scoring always converges to the global maximum of the log-likelihood function. Extensive computations are involved when implementing the nonlinear least squares and Tobit maximum likelihood procedures. Heckman's two step estimator can significantly reduce the computation load by combining a probit maximum likelihood procedure and a simple linear regression procedure. Treating the Tobit regression as a missing data structure, one can apply EM algorithm to estimate the unknown parameters. The computation cost is even less than Heckman's two-step estimate, since only simple linear regressions are needed in the procedure.

Simulation studies show that the Tobit maximum likelihood estimation is not robust to nonnormality and heteroscedasticity. This characteristic may be shared by other procedures since they all rely on the normal assumption of the error term ϵ. To overcome this disadvantage some nonparametric and semiparametric estimation procedures are constructed in literature. One such estimator is the least absolute deviation (LAD), proposed by Powell (1984). However, the merit of Powell's LAD estimator as being semi-parametric and robust to non-normality and heterscedasticity are diminished by the computational difficulty and the limitation that the regression function form must be linear. See Berg (1998) for more discussion. Lewbel and Linton (2002) and Zhou (2007) proposed several nonparametric estimation procedures for the regression function. Both of these estimators involve some integrals whose computation in turn uses numerical approximation, and more importantly, their estimators are not consistent unless some strict conditions are imposed on the tails of the distribution of ϵ.

Although the estimation procedures developed for the classical Tobit regression models are subject to some disadvantages, they still enjoy a great popularity among statisticians and econometricians because of the following reasons: (i) the real data generated from various applications may not be exactly normal, but are not far from normal, and after some data transformation, the homoscedasticity assumption holds. (ii) The computational difficulty is much less than their nonparametric and semi-parametric counterparts. And finally, (iii) the methodology developed for Tobit regression models with normal errors can be extended to Tobit regression models with non-normal errors.

This report will compare five different estimation procedures for Tobit regression models through simulation studies: Probit maximum likelihood estimator, least squares estimator, Heckman's two-step estimator, Tobit maximum likelihood estimator and the estimator based EM algorithm. To make the comparison, an empirical relative efficiency of an estimator to maximum likelihood estimator is employed. This relative efficiency is defined as the ratio of the empirical mean squares of errors from both estimation procedures. For each simulation setup, the efficiency is calculated. An estimation procedure is deemed to be good if the relative efficiency is close to 1.

The report is organized as follows. In Chapter 2, we will briefly review the basic ideas for each estimation procedures and the algorithms will be given. Any modifications to these methods are also discussed there. Simulation studies will be conducted in Chapter 3, together with some comparison results and our recommendations. For the sake of completeness, R codes for each estimation procedure are included in the Appendix.

Chapter 2 – Estimation Methods

For the sake of brevity, throughout the report, we shall assume that the predictor x is univariate. The extensions of the developed algorithms to multidimensional cases would be straightforward.

Probit Maximum Likelihood Estimators

The Probit model is a popular model in econometrics and statistics. The response variable, y, is binary while the independent variables can be continuous or categorical. The Probit model, along with the logistic model, is one of the most popular models for dichotomous data.

The Tobit likelihood function can be trivially rewritten as

$$L = \prod_0 \left[1 - \Phi\left(\frac{\alpha + X_i\beta}{\sigma}\right)\right] \prod_1 \Phi\left(\frac{\alpha + X_i\beta}{\sigma}\right) \prod_1 \Phi\left(\frac{\alpha + X_i\beta}{\sigma}\right)^{-1} \sigma^{-1} \varphi\left(\frac{Y_i - (\alpha + X_i\beta)}{\sigma}\right). \quad (2.1)$$

Then, the likelihood function of the Probit model is simply

$$L = \prod_0 \left[1 - \Phi\left(\frac{\alpha + X_i\beta}{\sigma}\right)\right] \prod_1 \Phi\left(\frac{\alpha + X_i\beta}{\sigma}\right). \quad (2.2)$$

The Probit maximum likelihood estimator of $\frac{\alpha}{\sigma}$ and $\frac{\beta}{\sigma}$, denoted $\frac{\hat{\alpha}}{\sigma}$ and $\frac{\hat{\beta}}{\sigma}$, is found by maximizing the likelihood function (2.2). In this study we utilize the R function glm with a probit link function to maximize. It is quickly obvious that one cannot estimate α, β and σ separately, but must estimate the ratios $\frac{\alpha}{\sigma}$ and $\frac{\beta}{\sigma}$ instead. This results in a loss of efficiency and for this study the Probit maximum likelihood estimator is examined only when σ is known.

Least Squares Estimators

Simple calculation shows that

$$E(y_i|y_i > 0) = (\alpha + X_i\beta) + \sigma\lambda((\alpha + X_i\beta)/\sigma) \quad (2.3)$$

and

$$Ey_i = \Phi\left(\alpha + X_i\beta/\sigma\right)[X_i\beta + \sigma\lambda(\alpha + X_i\beta/\sigma)] \quad (2.4)$$

where $\lambda\left((\alpha + X_i\beta)/\sigma\right) = \phi((\alpha + X_i\beta)/\sigma)/\Phi((\alpha + X_i\beta)/\sigma)$ is the reciprocal of Mill's ratio. These relationships imply that simply regressing y on x will ignore some factors in the regression function (2.3) and (2.4), hence results in biased estimates. A consistent and asymptotically normally distributed estimate can be obtained by considering the following nonlinear regression models

$$y_i = (\alpha + X_i\beta) + \sigma\lambda\left(\alpha + X_i\beta/\sigma\right) + \xi_i, \quad y_i > 0 \quad (2.5)$$

or

$$y_i = \Phi\left(\alpha + X_i\beta\right) \; + \sigma\lambda(\alpha + X_i\beta/\sigma)] + \eta_i. \quad (2.6)$$

In the following section, we will develop the algorithm to implement the nonlinear least squares procedures.

σ is unknown

First let's consider the nonlinear least squares estimation based on model (2.5). For convenience, let $z_i = (\alpha + X_i\beta)/\sigma$ and $h(z) = z + \lambda(z)$. The MLEs of α, β and σ using only positive observations are defined as

$$\left(\hat{\alpha}, \hat{\beta}, \hat{\sigma}\right) = \operatorname{argmin}_{\alpha,\beta,\sigma} L_n(\alpha, \beta, \sigma) = \operatorname{argmin}_{\alpha,\beta,\sigma} \sum_{i=1}^{n} [y_i - \sigma h(z_i)]^2.$$

With basic calculus, it's easy to see that

$$\lambda'(z) = -z\lambda(z) - \lambda^2(z), \; \lambda''(z) = (z^2 - 1)\lambda(z) + 3z\lambda^2(z) + 2\lambda^3(z). \quad (2.7)$$

Then

$$\frac{\partial z}{\partial \alpha} = \frac{1}{\sigma}, \; \frac{\partial z}{\partial \beta} = \frac{x}{\sigma}, \; \frac{\partial z}{\partial \sigma} = -\frac{z}{\sigma}, \; h'(z) = 1 + \lambda'(z), \; h''(z) = \lambda''(z), \quad (2.8)$$

and

$$\frac{\partial[y-\sigma(z+\lambda(z))]}{\partial\alpha} = -h'(z),$$
$$\frac{\partial[y-\sigma(z+\lambda(z))]}{\partial\beta} = -xh'(z), \tag{2.9}$$
$$\frac{\partial[y-\sigma(z+\lambda(z))]}{\partial\sigma} = -h(z)+zh'(z).$$

To use Newton-Rhaphson algorithm, we have to calculate the first and second order derivatives of $L_n(\alpha,\beta,\sigma)$ with respect to α, β and σ. Using (2.7), (2.8) and (2.9), the first order derivatives are

$$\frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha} = -2\sum_{i=1}^{n}[y_i-\sigma h(z_i)]h'(z_i),$$
$$\frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\beta} = -2\sum_{i=1}^{n}[y_i-\sigma h(z_i)]h'(z_i)X_i,$$
$$\frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\sigma} = -2\sum_{i=1}^{n}[y_i-\sigma h(z_i)][h(z_i)-z_ih'(z_i)].$$

The second derivatives are

$$\frac{\partial^2 L_n(\alpha,\beta,\sigma)}{\partial\alpha^2} = 2\sum_{i=1}^{n}\left[h'(z_i)\right]^2 - 2\sum_{i=1}^{n}\frac{\left[y_i-\sigma h(z_i)\right]h''(z_i)}{\sigma},$$
$$\frac{\partial^2 L_n(\alpha,\beta,\sigma)}{\partial\alpha\partial\beta} = 2\sum_{i=1}^{n}\left[h'(z_i)\right]^2 - 2\sum_{i=1}^{n}\frac{\left[y_i-\sigma h(z_i)\right]h''(z_i)X_i}{\sigma},$$
$$\frac{\partial^2 L_n(\alpha,\beta,\sigma)}{\partial\alpha\partial\sigma} = 2\sum_{i=1}^{n}\left[h(z_i)-Z_ih'(z_i)\right]h'(z_i)+2\sum_{i=1}^{n}\frac{\left[y_i-\sigma h(z_i)\right]h''(z_i)z_i}{\sigma},$$
$$\frac{\partial^2 L_n(\alpha,\beta,\sigma)}{\partial\beta^2} = 2\sum_{i=1}^{n}X_i^2\left[h'(z_i)\right]^2 - 2\sum_{i=1}^{n}\frac{\left[y_i-\sigma h(z_i)\right]h''(z_i)X_i^2}{\sigma},$$
$$\frac{\partial^2 L_n(\alpha,\beta,\sigma)}{\partial\beta\partial\sigma} = 2\sum_{i=1}^{n}\left[h(z_i)-z_ih'(z_i)\right]h'(Z_i)X_i+2\sum_{i=1}^{n}\frac{\left[y_i-\sigma h(z_i)\right]h''(z_i)X_iz_i}{\sigma},$$
$$\frac{\partial^2 L_n(\alpha,\beta,\sigma)}{\partial\sigma^2} = 2\sum_{i=1}^{n}\left[h(z_i)-z_ih'(z_i)\right]^2 - 2\sum_{i=1}^{n}\frac{\left[y_i-\sigma h(z_i)\right]Z_i^2h''(z_i)}{\sigma}.$$

Then we can use the following Newton-Rhaphson algorithm to find out the MLEs of α, β and σ.

Algorithm

(1) Select α_0, β_0 and σ_0 be the initial values;

(2) Iterate the following equation:

$$\begin{pmatrix}\hat{\alpha}\\ \hat{\beta}\\ \hat{\sigma}\end{pmatrix}=\begin{pmatrix}\alpha_0\\ \beta_0\\ \sigma_0\end{pmatrix}-\left.\begin{pmatrix}\frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha^2} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha\partial\beta} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha\partial\sigma}\\ \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha\partial\beta} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\beta^2} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\beta\partial\sigma}\\ \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha\partial\sigma} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\beta\partial\sigma} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\sigma^2}\end{pmatrix}^{-1}\begin{pmatrix}\frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha}\\ \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\beta}\\ \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\sigma}\end{pmatrix}\right|_{\substack{\alpha=\alpha_0\\ \beta=\beta_0\\ \sigma=\sigma_0}} \tag{2.10}$$

until it converges.

Now let's consider the nonlinear least squares estimation based on model (2.6) which uses all the available data on y including 0s. Denote $\lambda(z)=z\Phi(z)+\phi(z)$ and $z=\alpha+\beta x/\sigma$, then we have $\lambda'(z)=\Phi(z)$, $\lambda''(z)=\phi(z)$. In this case, we have to minimize the following quantity

$$L_n(\alpha,\beta,\sigma)=\sum_{i=1}^{n}[y_i-\sigma\lambda(z_i)]^2.$$

Note that

$$\frac{\partial[y-\sigma\lambda(z)]}{\partial\alpha}=-\lambda'(z),$$
$$\frac{\partial[y-\sigma\lambda(z)]}{\partial\beta}=-x\lambda'(z),$$
$$\frac{\partial[y-\sigma\lambda(z)]}{\partial\sigma}=-\lambda(z)+z\lambda'(z).$$

Then we can obtain the first order derivatives

$$\frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha}=-2\sum_{i=1}^{n}[Y_i-\sigma\lambda(Z_i)]\lambda'(Z_i),$$
$$\frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\beta}=-2\sum_{i=1}^{n}[Y_i-\sigma\lambda(Z_i)]\lambda'(Z_i)X_i,$$
$$\frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\sigma}=-2\sum_{i=1}^{n}[Y_i-\sigma\lambda(Z_i)][\lambda(Z_i)-Z_i\lambda'(Z_i)].$$

The second order derivatives are

$$\frac{\partial^2 L_n(\alpha,\beta,\sigma)}{\partial\alpha^2} = 2\sum_{i=1}^{n}[\lambda'(Z_i)]^2 - 2\sum_{i=1}^{n}\frac{[Y_i-\sigma\lambda(Z_i)]\lambda''(Z_i)}{\sigma},$$

$$\frac{\partial^2 L_n(\alpha,\beta,\sigma)}{\partial\alpha\partial\beta} = 2\sum_{i=1}^{n}[\lambda'(Z_i)]^2 - 2\sum_{i=1}^{n}\frac{[Y_i-\sigma\lambda(Z_i)]\lambda''(Z_i)X_i}{\sigma},$$

$$\frac{\partial^2 L_n(\alpha,\beta,\sigma)}{\partial\alpha\partial\sigma} = 2\sum_{i=1}^{n}[\lambda(Z_i)-Z_i\lambda'(Z_i)]\lambda'(Z_i) + 2\sum_{i=1}^{n}\frac{[Y_i-\sigma\lambda(Z_i)]\lambda''(Z_i)Z_i}{\sigma},$$

$$\frac{\partial^2 L_n(\alpha,\beta,\sigma)}{\partial\beta^2} = 2\sum_{i=1}^{n}X_i^{\,2}[\lambda'(Z_i)]^2 - 2\sum_{i=1}^{n}\frac{[Y_i-\sigma\lambda(Z_i)]\lambda''(Z_i)X_i^{\,2}}{\sigma},$$

$$\frac{\partial^2 L_n(\alpha,\beta,\sigma)}{\partial\beta\partial\sigma} = 2\sum_{i=1}^{n}[(\lambda(Z_i)-Z\lambda'(Z_i))\lambda'(Z_i)X_i] + 2\sum_{i=1}^{n}\frac{[Y_i-\sigma\lambda(Z_i)]\lambda''(Z_i)X_iZ_i}{\sigma},$$

$$\frac{\partial^2 L_n(\alpha,\beta,\sigma)}{\partial\sigma^2} = 2\sum_{i=1}^{n}[\lambda(Z_i)-Z_i\lambda'(Z_i)]^2 - 2\sum_{i=1}^{n}\frac{[Y_i-\sigma\lambda(Z_i)]Z_i^{\,2}\lambda''(Z_i)}{\sigma}.$$

Then we can use the following Newton-Rhaphson algorithm to find out the MSEs of α, β and σ.

Algorithm

(1) Select α_0, β_0 and σ_0 be the initial values;

(2) Iterate the following equation:

$$\begin{pmatrix}\hat{\alpha}\\ \hat{\beta}\\ \hat{\sigma}\end{pmatrix} = \begin{pmatrix}\alpha_0\\ \beta_0\\ \sigma_0\end{pmatrix} - \left.\begin{pmatrix}\frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha^2} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha\partial\beta} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha\partial\sigma}\\ \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha\partial\beta} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\beta^2} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\beta\partial\sigma}\\ \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha\partial\sigma} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\beta\partial\sigma} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\sigma^2}\end{pmatrix}^{-1}\begin{pmatrix}\frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\alpha}\\ \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\beta}\\ \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial\sigma}\end{pmatrix}\right|_{\substack{\alpha=\alpha_0\\ \beta=\beta_0\\ \sigma=\sigma_0}} \tag{2.11}$$

σ is known

Sometimes, the standard deviation σ is known. In this case, we only have to estimate α and β. The equation iterated in the Newton-Rhaphson algorithm becomes more simple:

$$\begin{pmatrix} \hat{\alpha} \\ \hat{\beta} \end{pmatrix} = \begin{pmatrix} \alpha_0 \\ \beta_0 \end{pmatrix} - \begin{pmatrix} \frac{\partial L_n(\alpha,\beta)}{\partial \alpha^2} & \frac{\partial L_n(\alpha,\beta)}{\partial \alpha \partial \beta} \\ \frac{\partial L_n(\alpha,\beta)}{\partial \alpha \partial \beta} & \frac{\partial L_n(\alpha,\beta)}{\partial \beta^2} \end{pmatrix}^{-1} \begin{pmatrix} \frac{\partial L_n(\alpha,\beta)}{\partial \alpha} \\ \frac{\partial L_n(\alpha,\beta)}{\partial \beta} \end{pmatrix} \Bigg|_{\substack{\alpha=\alpha_0 \\ \beta=\beta_0}} .$$

The first and second derivatives are identically defined as before, except σ is a known value.

Remark on the Least Squares Estimators: σ is unknown

Hartley (1976) and Amemiya (1981) showed that the nonlinear least squares estimators are asymptotically normal and consistency is then a natural consequence. When σ is unknown, the MSEs of α, β and σ are obtained by iterating the Newton-Rhaphson equations (2.10) or (2.11). However, the potential singularity of the matrix

$$\begin{pmatrix} \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial \alpha^2} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial \alpha \partial \beta} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial \alpha \partial \sigma} \\ \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial \alpha \partial \beta} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial \beta^2} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial \beta \partial \sigma} \\ \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial \alpha \partial \sigma} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial \beta \partial \sigma} & \frac{\partial L_n(\alpha,\beta,\sigma)}{\partial \sigma^2} \end{pmatrix}$$

presents some serious computation challenges. A possible way to avoid the singularity is to re-parameterize the model. For example, one can define $a = \alpha / \sigma$, $b = \beta / \sigma$, $\sigma = \sigma$ and apply the Newton-Rhaphson algorithm directly to a, b and σ.

Another way to avoid the calculation of the second order derivative matrix is to use the fixed-point algorithm. Suppose σ is unknown, the nonlinear least squares estimation procedure based on all data is to solve the following equations:

$$\sum_{i=1}^{n} [y_i - \alpha\Phi(z_i) - \beta x_i \Phi(z_i) - \sigma\phi(z_i)]\Phi(z_i) = 0$$

$$\sum_{i=1}^{n} [y_i - \alpha\Phi(z_i) - \beta x_i \Phi(z_i) - \sigma\phi(z_i)]\mathrm{x_i}\Phi(z_i) = 0$$

$$\sum_{i=1}^{n} [y_i - \alpha\Phi(z_i) - \beta x_i \Phi(z_i) - \sigma\phi(z_i)]\phi(z_i) = 0$$

where $z = (\alpha + X_i\beta)/\sigma$. It should be noted that, similarly, one can construct a fixed point algorithm for other cases.

Tobit Maximum Likelihood Estimators

The log-likelihood function of the Tobit regression models is given by

$$\log L = \sum_0 \log\left[1 - \Phi\left(\frac{\alpha + X_i\beta}{\sigma}\right)\right] - \frac{n_1}{2}\log\sigma^2 - \frac{1}{2\sigma^2}\sum_1 \left(y_i - \alpha - X_i\beta\right)^2$$

where n_1 is the number of non-zero observations. Again we will consider situations in which σ is known and unknown. Here, the Tobit MLE is consistent and asymptotically normal, as shown in Amemiya (1973). In the simulation the R function VGLM is utilized to obtain these estimates. Below is a description of the algorithm utilized by this function.

σ is unknown

Following Olsen (1978)'s suggestion, we use the transformed parameters $a = \dfrac{\alpha}{\sigma}, b = \dfrac{\beta}{\sigma}$ and $h = \dfrac{1}{\sigma}$. Hence, the log-likelihood function in terms of the new parameters can be written as

$$\log L = \sum_0 \log\left[1 - \Phi\left(a + X_i b\right)\right] - n_1 \log h - \frac{1}{2}\sum_1 \left(y_i - \alpha - X_i b\right)^2$$

where $\lambda_i = \dfrac{\phi(a + X_i b)}{[1 - \Phi(a + X_i b)]}$.

The first order derivatives of log L with respect to a, b, and h are

$$\frac{\partial \log L}{\partial a} = -\sum_0 \lambda_i + \sum_1 (hy_i - a - X_i b),$$

$$\frac{\partial \log L}{\partial b} = -\sum_0 X_i\lambda_i + \sum_1 X_i(hy_i - a - X_i b),$$

$$\frac{\partial \log L}{\partial h} = \frac{n_1}{h}\sum_1 (hy_i - a - X_i b)y_i.$$

The second order derivatives of log L with respect to a, b, and h are

$$\frac{\partial^2 \log L}{\partial a^2} = \sum_0 \lambda_i (a + X_i b - \lambda_i) - n_1,$$

$$\frac{\partial^2 \log L}{\partial a \partial b} = \sum_0 \lambda_i X_i (a + X_i b - \lambda_i) - \sum_1 X_i,$$

$$\frac{\partial^2 \log L}{\partial a \partial h} = \sum_1 y_i,$$

$$\frac{\partial^2 \log L}{\partial b^2} = \sum_0 \lambda_i X_i^{\,2} (a + X_i b - \lambda_i) - \sum_1 X_i^{\,2},$$

$$\frac{\partial^2 \log L}{\partial b \partial h} = \sum_1 X_i y_i,$$

$$\frac{\partial^2 \log L}{\partial h^2} = -\frac{n_1}{h^2} - \sum_1 y_i^{\,2}.$$

Then the Newton-Rhaphson algorithm of finding MLEs of a, b and h is to iterate the following equation:

$$\begin{pmatrix} \hat{a} \\ \hat{b} \\ \hat{h} \end{pmatrix} = \begin{pmatrix} a_0 \\ b_0 \\ h_0 \end{pmatrix} - \left. \begin{pmatrix} \frac{\partial L_n(a,b,h)}{\partial a^2} & \frac{\partial L_n(a,b,h)}{\partial a \partial b} & \frac{\partial L_n(a,b,h)}{\partial a \partial h} \\ \frac{\partial L_n(a,b,h)}{\partial a \partial b} & \frac{\partial L_n(a,b,h)}{\partial b^2} & \frac{\partial L_n(a,b,h)}{\partial b \partial h} \\ \frac{\partial L_n(a,b,h)}{\partial a \partial h} & \frac{\partial L_n(a,b,h)}{\partial b \partial h} & \frac{\partial L_n(a,b,h)}{\partial h^2} \end{pmatrix}^{-1} \begin{pmatrix} \frac{\partial L_n(a,b,h)}{\partial a} \\ \frac{\partial L_n(a,b,h)}{\partial b} \\ \frac{\partial L_n(a,b,h)}{\partial h} \end{pmatrix} \right|_{\substack{a = a_0 \\ b = b_0 \\ h = h_0}} \qquad (2.12)$$

The MSEs of α, β and σ will be obtained by

$$\hat{\alpha} = \frac{\hat{a}}{\hat{h}}, \quad \hat{\beta} = \frac{\hat{b}}{\hat{h}}, \quad \hat{\sigma} = \frac{1}{\hat{h}}.$$

σ is known

If σ is known, we use the transformed parameters $a = \frac{\alpha}{\sigma}$ and $b = \frac{\beta}{\sigma}$. Again denote $h = \frac{1}{\sigma}$ and $\lambda_i = \frac{\phi(a + X_i b)}{[1 - \Phi(a + X_i b)]}$. The first order derivative of log L with respect to a and b are

$$\frac{\partial \log L}{\partial a} = -\sum_0 \lambda_i + \sum_1 (hy_i - a - X_i b),$$
$$\frac{\partial \log L}{\partial b} = -\sum_0 X_i \lambda_i + \sum_1 X_i (hy_i - a - X_i b).$$

The second derivatives of log L with respect to a and b are

$$\frac{\partial^2 \log L}{\partial a^2} = \sum_0 \lambda_i (a + X_i b - \lambda_i) - n_1,$$
$$\frac{\partial^2 \log L}{\partial a \partial b} = \sum_0 \lambda_i X_i (a + X_i b - \lambda_i) - \sum_1 X_i,$$
$$\frac{\partial^2 \log L}{\partial b^2} = \sum_0 \lambda_i X_i^2 (a + X_i b - \lambda_i) - \sum_1 X_i^2.$$

The Newton-Rhaphson algorithm of finding MSEs of a and b is to iterate the following equation:

$$\begin{pmatrix} \hat{a} \\ \hat{b} \end{pmatrix} = \begin{pmatrix} a_0 \\ b_0 \end{pmatrix} - \left. \begin{pmatrix} \dfrac{\partial L_n(a,b,h)}{\partial a^2} & \dfrac{\partial L_n(a,b,h)}{\partial a \partial b} \\ \dfrac{\partial L_n(a,b,h)}{\partial a \partial b} & \dfrac{\partial L_n(a,b,h)}{\partial b^2} \end{pmatrix}^{-1} \begin{pmatrix} \dfrac{\partial L_n(a,b,h)}{\partial a} \\ \dfrac{\partial L_n(a,b,h)}{\partial b} \end{pmatrix} \right|_{\substack{a = a_0 \\ b = b_0}} \qquad (2.13)$$

The MSEs of α and β will be obtained by

$$\hat{\alpha} = \sigma \hat{a}, \quad \hat{\beta} = \sigma \hat{b}.$$

Tobit Maximum likelihood estimators are strongly consistent and are asymptotically normal. Unfortunately, due to the non-linearality of the equations they must be solved iteratively and do take some computation time.

Heckman's two-step estimator

Heckman's two-step estimation procedure, also known as λ-correction or Heckit method, was originally designed for the Type 3 Tobit model. It turns out this methodology also applies to the standard Tobit regression model after a minor adjustment.

The estimation procedure relies on one of the following equations, which also appear in the section on the Least squares estimation procedure.

$$y_i = (\alpha + X_i\beta) + \sigma\lambda\left(\frac{\alpha + X_i\beta}{\sigma}\right) + \xi_i, \text{ for } i \text{ such that } y_i > 0 \tag{2.14}$$

$$y_i = \Phi\left(\frac{\alpha + X_i\beta}{\sigma}\right)\left[(\alpha + X_i\beta) + \sigma\lambda\left(\frac{\alpha + X_i\beta}{\sigma}\right)\right] + \delta_i \tag{2.15}$$

First we assume that σ^2 is unknown. The following is the steps to implement the Heckman's two-step estimation method.

Step 1: Estimate $\left(\frac{\alpha + X_i\beta}{\sigma}\right)$ by the Probit MLE defined earlier or other applicable procedures. Denote the estimate as $\left(\widehat{\frac{\alpha + X_i\beta}{\sigma}}\right)$.

Step 2: If (2.14) is used, then regress y_i on $(\alpha + X_i\beta)$ and $\lambda\left(\widehat{\frac{\alpha + X_i\beta}{\sigma}}\right)$ by least squares using only the positive observations on y_i. The coefficient of $(\alpha + X_i\beta)$ will be the estimator of β, and the coefficient of $\lambda\left(\widehat{\frac{\alpha + X_i\beta}{\sigma}}\right)$ will be the estimator of σ. If (2.15) is used, then regress y_i on $\Phi\left(\widehat{\frac{\alpha + X_i\beta}{\sigma}}\right)X_i$ and $\phi\left(\widehat{\frac{\alpha + X_i\beta}{\sigma}}\right)$ without intercept by least squares using all the data y_i. The coefficient of $\Phi\left(\widehat{\frac{\alpha + X_i\beta}{\sigma}}\right)X_i$ will be the estimator of β and the coefficient of $\phi\left(\widehat{\frac{\alpha + X_i\beta}{\sigma}}\right)$ will be the estimator of σ.

The Heckman's two-step procedure for known σ^2 follows the similar steps as above, except the response variable in the regression analysis in Step 2 becomes $y - \sigma\lambda\left(\widehat{\frac{\alpha + X_i\beta}{\sigma}}\right)$ for (3.1) and $y_i - \sigma\phi\left(\widehat{\frac{\alpha + X_i\beta}{\sigma}}\right)$ for (3.2), and β is still estimated by the coefficient of $\lambda\left(\widehat{\frac{\alpha + X_i\beta}{\sigma}}\right)$ or the coefficient of $\Phi\left(\widehat{\frac{\alpha + X_i\beta}{\sigma}}\right)X_i$.

The regression models in (2.14) and (2.15) are heteroscedastic. An efficient estimation procedure should take the variances into account, for example, one can use the weighted least squares in Step 2. However, using the weighted least squares procedure requires one to consistently estimate the asymptotic covariance matrix, which in turn needs initial estimates for the regression parameters.

Large sample results, such as the weak consistency, the asymptotically normality of Heckman's two-step estimators can be found in Amemiya (1984) and Heckman (1979).

The EM algorithm

The EM algorithm is a generic device that provides an iterative procedure for computing MLEs in situations where, but for the absence of some additional data, MLE would be straightforward. We start this section with a brief introduction of EM algorithm in a general setup.

Let Y be the random vector with density function $f(y;\theta)$, where $\theta \epsilon \Theta$, X be the vector containing the complete data which include some additional data, referred to as the unobservable or missing data. Let $f_c(x;\theta)$ denote the density function of the random vector X corresponding to the complete data vector x. Then the complete data log-likelihood function is given by

$$\log L_c(\theta) = log f_c(x;\theta).$$

Let $\theta^{(0)}$ be some initial value of θ. Then on the first iteration of EM algorithm, the E-step requires the calculation of

$$Q\left(\theta;\theta^{(0)}\right) = E_{\theta^{(0)}}[log L_c(\theta)|Y = y].$$

The M-step requires the maximization of $Q(\theta;\theta^{(0)})$ with respect to θ over the parameter space Θ. That is, we choose $\theta^{(1)}$ such that

$$Q(\theta^{(1)};\theta^{(0)}) \geq Q(\theta;\theta^{(0)}).$$

For all $\theta \epsilon \Theta$. The E- and M-steps are then carried out again, but this time with $\theta^{(0)}$ replaced by $\theta^{(1)}$. On the j-th iteration, the E- and M-steps are defined as follows:

E-Step: Calculate $Q\left(\theta;\theta^{(j-1)}\right) = E_{\theta^{(j-1)}}[log L_c(\theta)|Y = y]$.

M-Step: Choose $\theta^{(j)}$ such that $Q(\theta^{(j)};\theta^{(j-1)}) \geq Q(\theta;\theta^{(j-1)})$ for all $\theta \epsilon \Theta$.

The E- and M-steps are iterated repeatedly until some convergence criteria is met. For example, one can stop the iteration whenever the difference $L(\theta^k) - L(\theta^{k-1})$, or $|\theta^{(k)} - \theta^{(k-1)}|$, changes by a very small amount.

The formulation of the idea behind the EM algorithm can be traced back to the late 19th century. But, it was the paper by Dempster, Laird and Rubin (1977) that the ideas in the earlier literature were synthesized, a general formulation and a theory developed, and a variety of applications indicated.

The EM algorithm is especially suited for censored regression models such as Tobit models. Now, we consider the application of the EM algorithm to the Tobit model. Define $\theta = (\alpha, \beta, \sigma^2)'$. Suppose all the values of Y^* are observable, then we have

$$L_c(\theta) = -\frac{n}{2} log\sigma^2 - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i^* - \alpha - X_i\beta)^2 .$$

For an initial value of $\theta^{(0)} = (\alpha_0, \beta_0, {\sigma^2}_0)$, let W denote a random variable indicating whether y is truncated. Then, the EM algorithm for iteration is as follows.

E-Step:

$$E\left[L_c(\theta)\middle|Y = y, W = w, \theta^{(0)}\right] = -\frac{n}{2} log\sigma^2 - \frac{1}{2\sigma^2}\sum_{i:w_i=0}(y_i - \alpha - X_i\beta)^2$$

$$-\frac{1}{2\sigma^2}\sum_{i:w_i=0}[E(y_i^*|W_i = 0, \theta^{(0)}) - \alpha - X_i\beta]^2 + \frac{1}{2\sigma^2}\sum_{i:w_i=0} Var(y_i^*|W_i = 0, \theta^{(0)}).$$

Where

$$E\left[y^*|W_i = 0, \theta^{(0)} = \alpha_0 + X_i\beta_0 - \frac{\sigma_0\phi_0}{1-\Phi_0}\right],$$

$$Var(y_i^*|W_i = 0, \theta^{(0)}) = \sigma_0^2 + \frac{\sigma_0\phi_0(\alpha_0 + X_i\beta_0)}{1-\Phi_0} - \left(\frac{\sigma_0\phi_0}{1-\Phi_0}\right)^2,$$

and $\phi_0 = \phi\left(\frac{\alpha_0 + X_i\beta_0}{\sigma_0}\right)$ and $\Phi_0 = \Phi\left(\frac{\alpha_0 + X_i'\beta_0}{\sigma_0}\right)$.

M-Step: Without loss of generality, we assume that the first n_1 observations of Y_i are positive, denoted by $Y^{(1)}$, and an n-n_1-vector with elements $E(Y^*|W_i = 0, \theta^{(1)})$ is denoted by $Y^{(2)}$. Arrange the matrix X accordingly. Then, by maximizing $E\left[L_c(\theta)\middle|Y = y, W = w, \theta^{(0)}\right]$ with respect to θ, we have

$$\binom{\alpha_1}{\beta_1} = (X'X)^{-1}X'\binom{Y^{(1)}}{Y^{(2)}},$$

$$\sigma_1^2 = \frac{1}{n}\left[\sum_{i=1}^{n_1}(Y_i^{(1)} - \alpha_1 - X_i\beta_1)^2 + \sum_{i=n_i+1}^{n}\left[(Y_i^{(1)} - \alpha_1 - X_i\beta_1)^2 + Var(y_i^*|W_i = 0, \theta^{(0)})\right]\right].$$

Anemiya (1984) showed that when *n* is large enough, and if the iteration is started from a point close to the MLE, the above estimate obtained by EM algorithm converges to the MLE.

Chapter 3 – Simulation Study

This section is a summarization of the simulation studies. These simulations are performed under various scenarios. The goal of the simulation study is to examine which estimation algorithm does best at estimating α, β and sometimes σ.

The simulation will be ran 500 times for sample sizes of 100, 200, 300, 400, 500, 800 and 1000. Each scenario consists of specifically chosen settings for the distribution of X, the distribution of ε, y_o and whether σ^2 is known or unknown. For all scenarios $\alpha=\beta=\sigma^2=1$. The parameter estimates and the respective MSEs are listed in tables located in Appendix A. Because we will examine various lower limits, y_o, for the response variable, I will also list the truncation percentage for each simulation. This is calculated by the percentage of y* observations that are less than the designated y_o (denoted low).

The simulation study will be conducted in 12 scenarios. The first six scenarios will use all 5 estimation methods. The last six scenarios will consist only of the least squares estimator, Heckman's two-step estimator, Tobit maximum likelihood estimator, and the EM algorithm. The scenarios will be conducted with the following parameter settings:

1. $\varepsilon \sim N(0,\sigma^2)$, $X \sim N(0,1)$, $y_o = -0.8$, σ^2 known
2. $\varepsilon \sim N(0,\sigma^2)$, $X \sim N(0,1)$, $y_o = 0$, σ^2 known
3. $\varepsilon \sim N(0,\sigma^2)$, $X \sim N(0,1)$, $y_o = 1$, σ^2 known
4. $\varepsilon \sim N(0,\sigma^2)$, $X \sim \text{Uniform}(-\sqrt{3}, \sqrt{3})$, $y_o = -0.8$, σ^2 known
5. $\varepsilon \sim N(0,\sigma^2)$, $X \sim \text{Uniform}(-\sqrt{3}, \sqrt{3})$, $y_o = 0$, σ^2 known
6. $\varepsilon \sim N(0,\sigma^2)$, $X \sim \text{Uniform}(-\sqrt{3}, \sqrt{3})$, $y_o = 1$, σ^2 known
7. $\varepsilon \sim N(0,\sigma^2)$, $X \sim N(0,1)$, $y_o = -0.8$, σ^2 unknown
8. $\varepsilon \sim N(0,\sigma^2)$, $X \sim N(0,1)$, $y_o = 0$, σ^2 unknown
9. $\varepsilon \sim N(0,\sigma^2)$, $X \sim N(0,1)$, $y_o = 1$, σ^2 unknown
10. $\varepsilon \sim N(0,\sigma^2)$, $X \sim \text{Uniform}(-\sqrt{3}, \sqrt{3})$, $y_o = -0.8$, σ^2 unknown
11. $\varepsilon \sim N(0,\sigma^2)$, $X \sim \text{Uniform}(-\sqrt{3}, \sqrt{3})$, $y_o = 0$, σ^2 unknown
12. $\varepsilon \sim N(0,\sigma^2)$, $X \sim \text{Uniform}(-\sqrt{3}, \sqrt{3})$, $y_o = 1$, σ^2 unknown

Results of Simulation Study

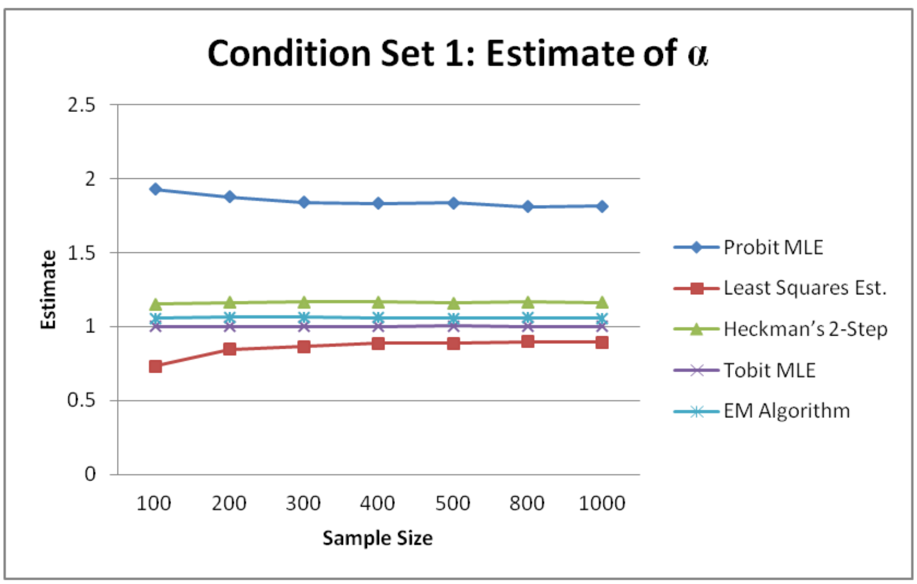
The simulation results are displayed in Table 1 – Table 12 in Appendix A. Some graphs are displayed with discussion, all others can be found in Appendix C. The simulation study will be discussed by evaluating each condition set individually. Condition Sets 1 through 6 estimate α and β only and Condition Sets 7 through 12 estimate α, β and σ. The results are compiled in 12 tables.

Results when σ^2 is known

As mentioned in the previous section, the scenarios were conducted under the assumption that σ^2 was known and unknown. I am first considering the cases where σ^2 is known. Three threshold or cutoff values were applied to two distributions.

Condition Set 1

Figure 3.1

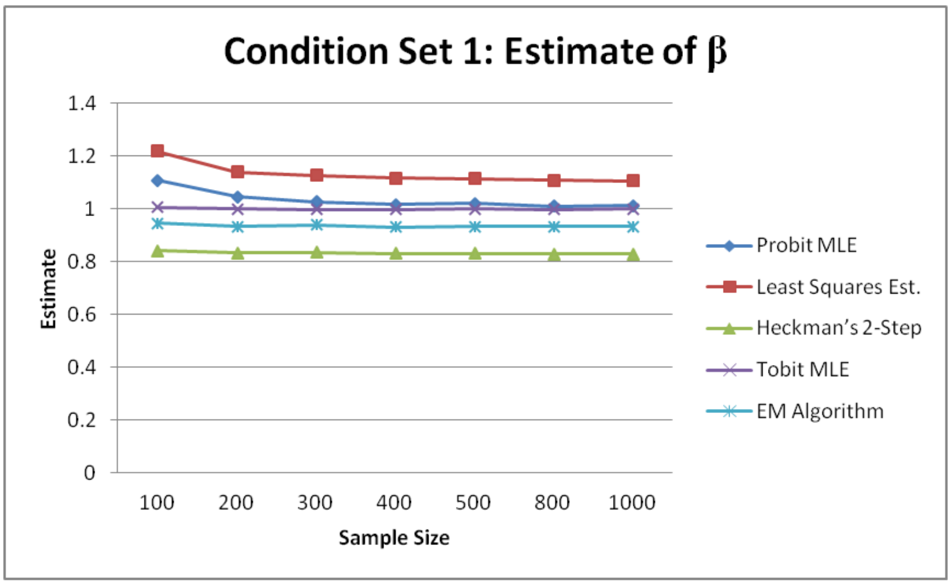


Under these conditions, all methods were able to produce estimates for cases when σ was known. Estimates of α and β are displayed in Figure 3.1 and Figure 3.2. As you can see, the Probit MLE does not do a good job of estimating α, but produces a fairly good estimation of β.

When estimating β, Heckman's Two-step and the Least Squares methods are not particularly effective. As sample size increases the Least Squares method becomes better than Heckman's. This is evident in both Figure 3.2 and table 1, where the MSE(b) for Least Squares becomes smaller than Heckman's for larger sample sizes.

Overall, accuracy becomes better as sample size increases. Estimates for all five methods grow closer to the actual values for both parameters. The best method for estimating under condition set 1 is Tobit maximum likelihood estimation because the estimates are very accurate and the MLEs for both α and β are small. As a second option, the EM algorithm does produce the next best estimates of α and β.

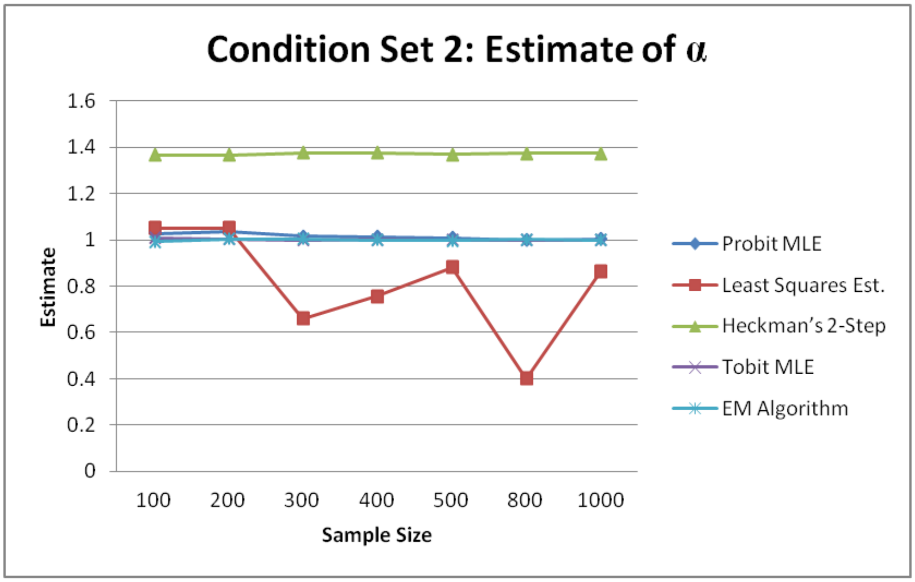
Figure 3.2



Condition Set 2

Again, all methods were able to produce estimates of both parameters. Under condition set 2, the truncation rate was nearly 25%. This causes the estimates of α and β to not be as good as in condition set 1.

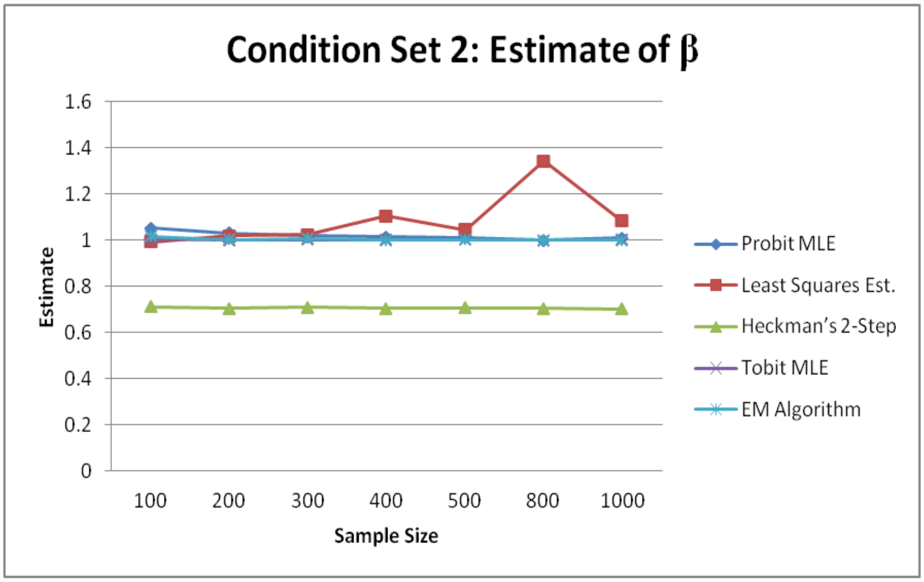
Figure 3.3



As seen in Figure 3.3, the Tobit MLE, EM Algorithm and Probit MLE methods have almost identical estimates for α. The Least squares method results in inconsistent estimates. Heckman's Two-step method produce over-estimates, which do not improve in accuracy as sample size increases.

Figure 3.4 displays estimates for β. Again, the Tobit MLE, EM algorithm and the Probit MLE methods have very similar and close estimates. Much like the estimates for α, the estimates produced by the Least Squares method are inconsistent as sample size increases. Heckman's two-step produces under-estimates, which do not improve in accuracy as sample size increases.

Figure 3.4

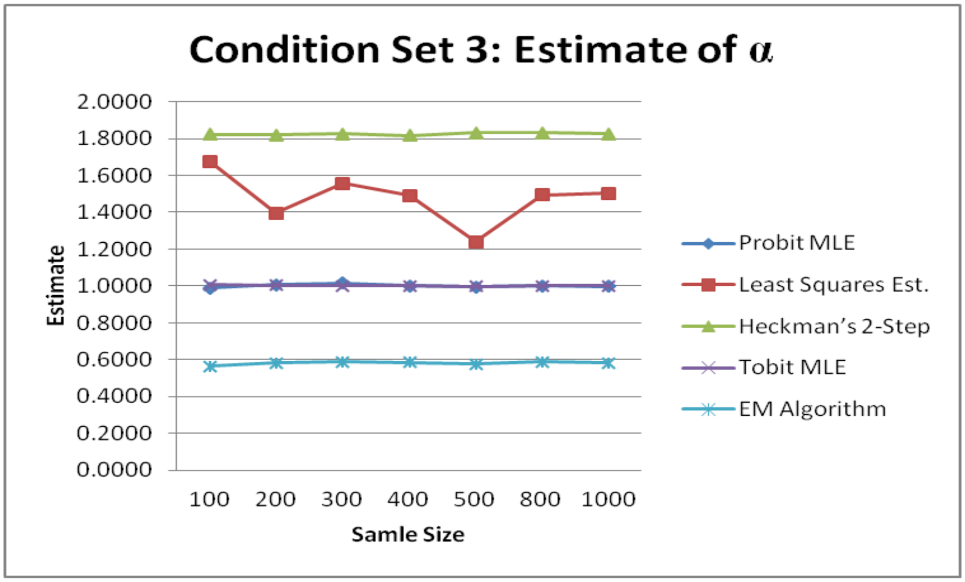


Condition set 2 is a very likely scenario. A threshold of zero is common in many economics and econometrics models as well as sociology, psychology, biology and others. There are three good methods one could use when analyzing these data. Displayed in Figure C.1 and Figure C.2 of Appendix C are the Tobit MLE, EM Algorithm and Probit MLE method estimates. From these figures, one can see that Tobit MLE does the best at estimating α and β. However, If ease of calculation was a concern, the Probit maximum likelihood estimation method is suitable under large sample sizes.

Condition Set 3

Condition set 3 has a threshold value of $Y_0 = 1$. This causes a truncation rate of approximately 50%, which is not desirable. Because of this, the estimates of α and β are occasionally very poor under some methods.

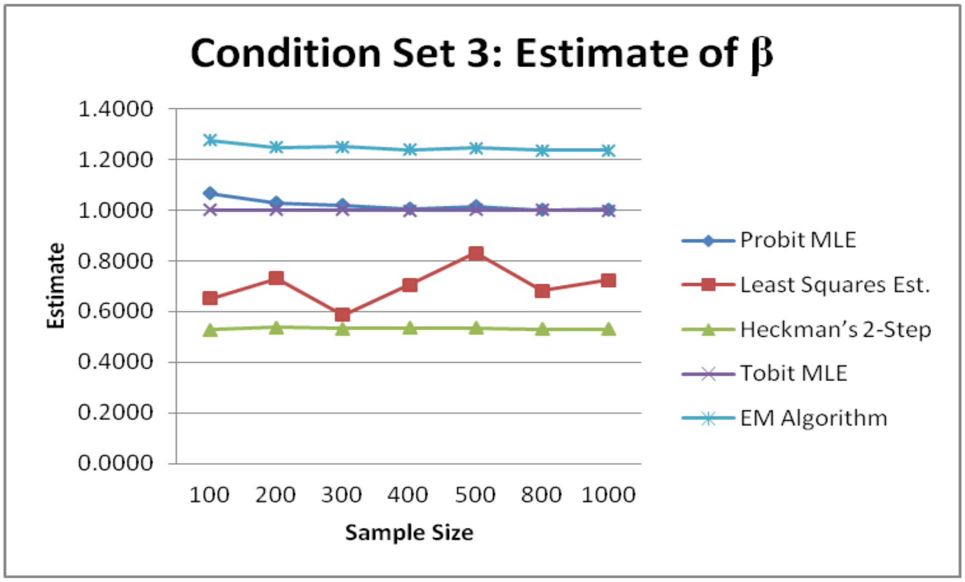
Figure 3.5



Estimates for α are inconsistent, over-estimates using the Least Squares method. The resulting mean squared errors are also extremely large, as seen in Table 3. The EM algorithm produces consistent under-estimates with large mean squared errors. Heckman's Two-step doesn't improve accuracy as sample size increases and yields the worst estimates of α of the four methods. Both the Probit and Tobit maximum likelihood methods estimate α well. However, the Probit method does so with relatively large errors. The Tobit estimation method not only estimates better but it also results in very small MSEs.

The estimation of β is not improved over α. Again Least squares, Heckman's two-step and the EM Algorithm produce poor estimates with unwanted, large MSEs. The Probit method improves estimation as sample size increases and yields desirable estimates of β with small errors when n is greater than 400. The most favorable option is Tobit maximum likelihood. This method estimates well at all sample sizes and results in small errors. However, I do believe that either Probit or Tobit methods would be satisfactory under these conditions.

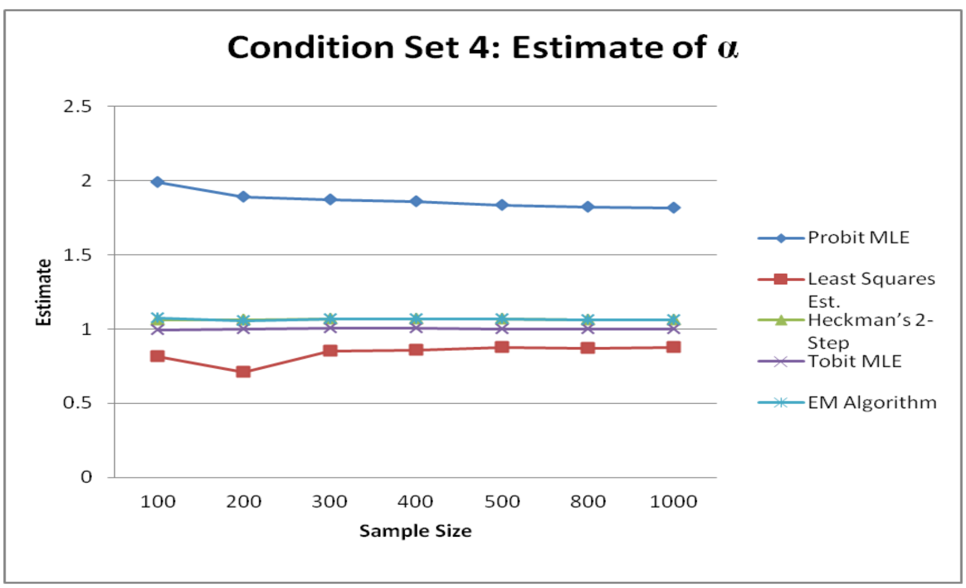
Figure 3.6



Condition Set 4

Condition set 4 has a Uniformly distributed X. The threshold for this data is Y_0 = -0.8, yielding a truncation rate of 10%.

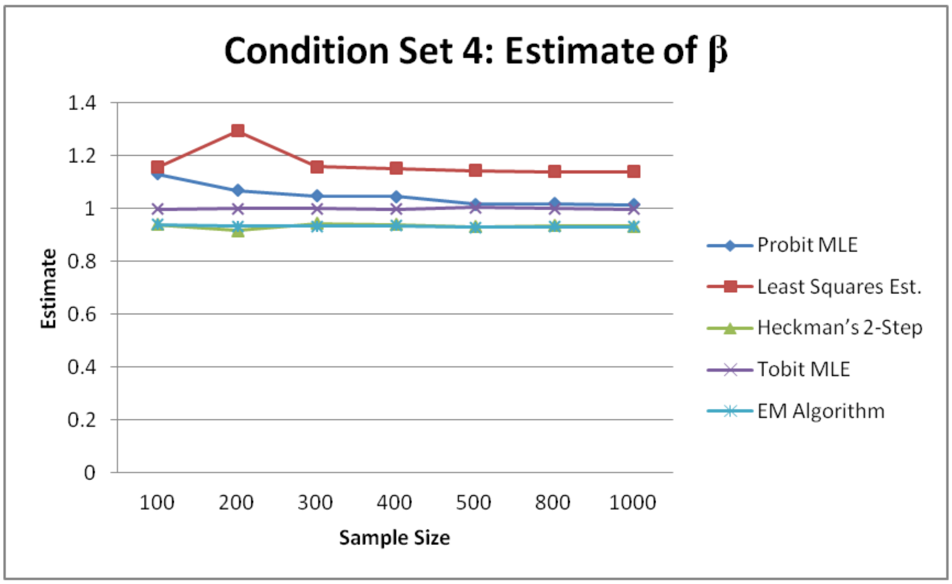
Figure 3.7



As you can see from Figure 3.7, all five methods converge as sample size increases. It is evident that Probit maximum likelihood estimation does not do a good job of estimating α. However, the mean squared errors for the Probit estimates are not as big as the MSEs for LSE estimates, when sample size is 100 or 200. This can be seen in Table 4.

Figure C.3 shows that Tobit maximum likelihood estimation is indeed the best estimator for α. This method also had the smallest mean squared errors. The EM and Heckman's two-step are very close in the estimates for α at all sample sizes. However, the squared errors for the EM algorithm estimates are smaller. A second good option for estimating α would be the EM algorithm, even though the calculations are somewhat cumbersome.

Figure 3.8



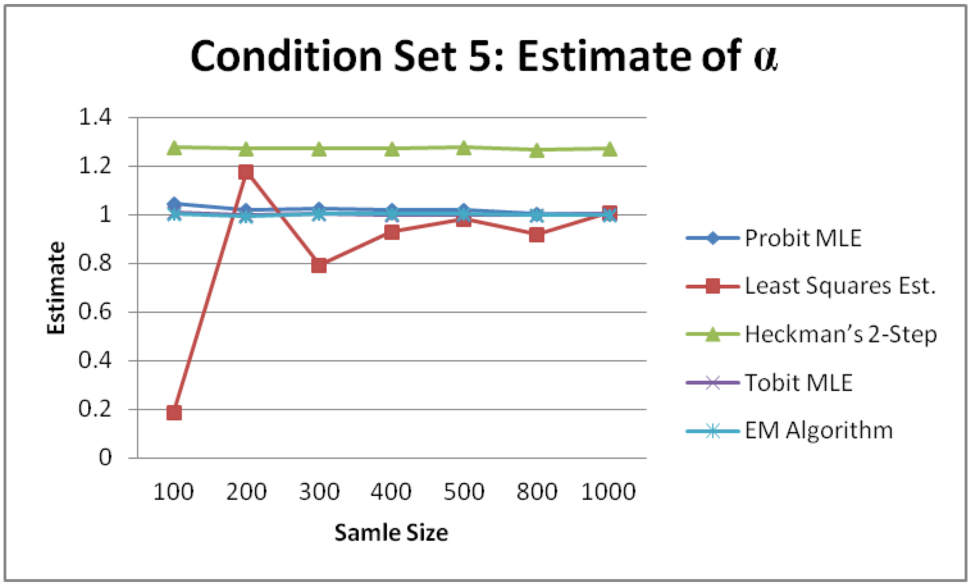
The estimates of β for all five methods are displayed in Figure 3.8. The Least Squares estimation method does not produce good estimates and, as shown in Table 4, the mean squared errors are large as well. The EM algorithm and Heckman's two-step method generate similar estimates for all sample sizes. The best estimates of β are a result of the Probit MLE and Tobit MLE methods. Probit MLE is best when sample size is very large.

Even though Probit maximum likelihood estimation estimates β well, it is at the cost of poor estimates of α. Under condition set 4, the best estimation method is Tobit maximum likelihood. It estimates both parameters well and with small mean squared errors.

Condition Set 5

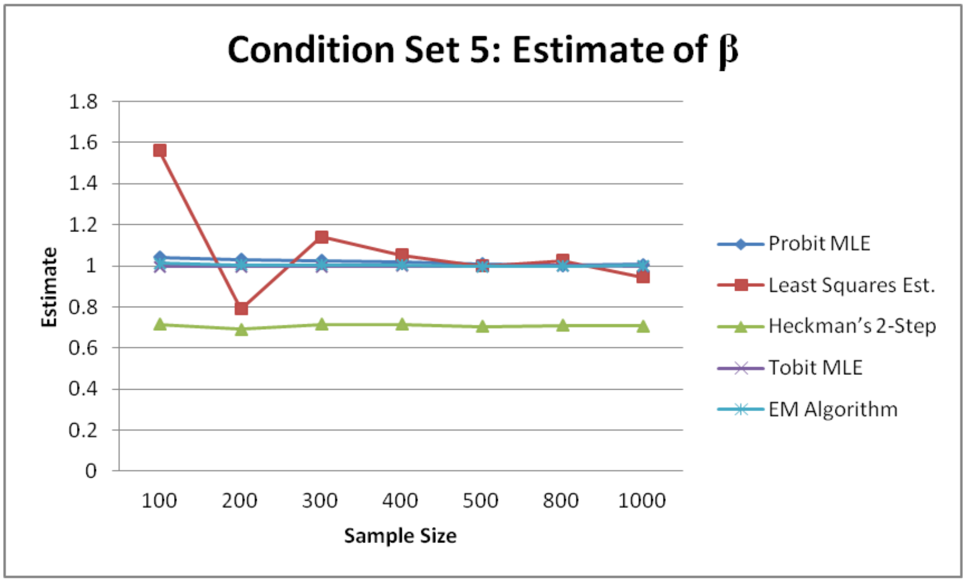
Under condition set 5, the threshold value is zero, which results in a truncation rate of about 25%.

Figure 3.9



There are three methods that produce good estimates of α. The first is Probit maximum likelihood. It is hard to see from Figure 3.9, but it is clear from Figure C.4 and Table 5 that the Probit MLE increases in the strength of estimation as sample size increases. When sample size is large, the estimates are very good and have very small mean squared errors. This method is an excellent choice when sample sizes are large. Tobit maximum likelihood estimation produces estimates that are close to α. The squared errors are small as well. However, the EM Algorithm also does a superb job at estimating α. The estimate oscillates around one until it converges to a value *very* close to one. The mean squared errors are very small. Because of this, I think this method is the best option for estimating α when X is Uniformly distributed and the cutoff value is equal to one.

Figure 3.10

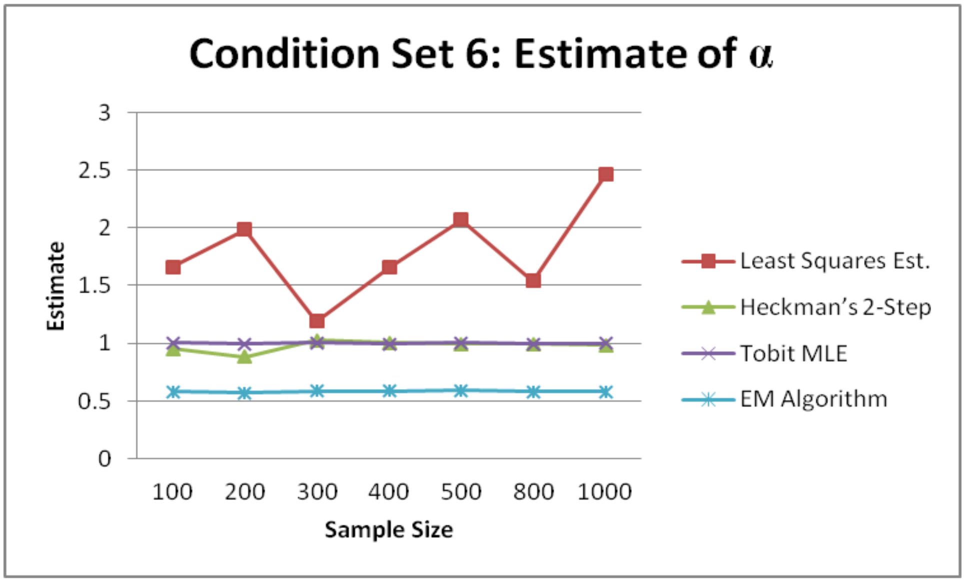


Like in estimating α, Heckman's Two-step and the Least Squares estimation methods do not have high quality estimates of β. The Least squares estimates have large MSEs and Heckman's two-step estimates do not converge to a value near one. Using figure C.5 I conclude that the best estimates of β are yielded by the EM algorithm and Tobit MLE. Either method is a good choice, but again I believe that the EM algorithm does the best.

Condition Set 6

Similarly to condition set 3, under condition set 6 Probit maximum likelihood estimation cannot be used. The other four methods are displayed in Table 6 and Figures 3.11 and 3.12. Note that the truncation rate is about 50% for all methods.

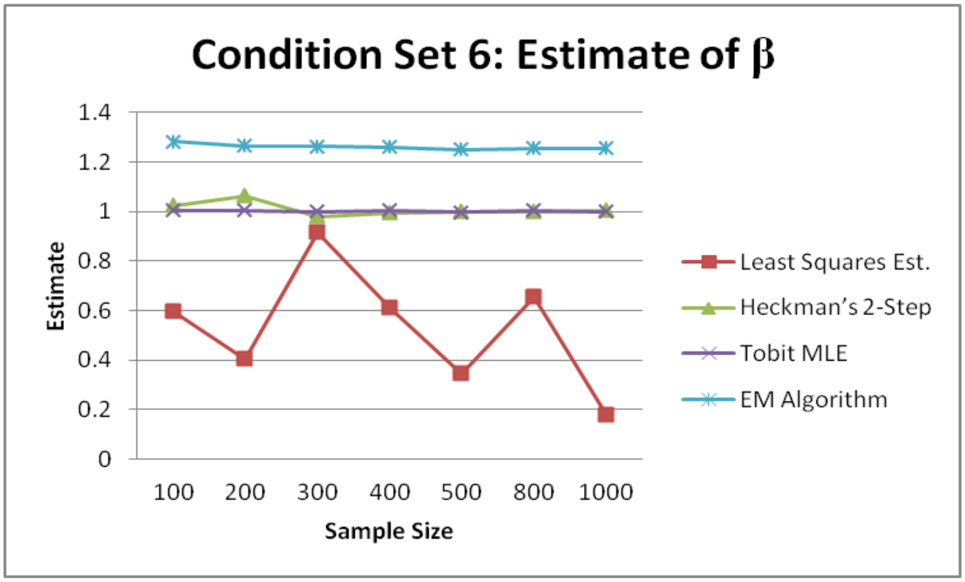
Figure 3.11



The estimates of α generated by both the EM algorithm and Least Squares estimation are poor. The estimates from Heckman's two-step and Tobit maximum likelihood estimation are much better. Both methods have good accuracy but Tobit MLEs do result in smaller mean squared errors. The mean squared errors from Heckman's estimates are large, especially when sample size is small. Even at n=1000, the mean squared errors are comparatively very large.

Similarly, the estimates of β are best from Tobit MLE and Heckman's two-step and are not desirable with the Least squares estimation or the EM algorithm methods. The mean squared errors for Heckman's estimate are much improved over the errors for the estimates of α. Although, this is not enough to make this method better than the Tobit method. Because of the very small MSEs I believe Tobit maximum likelihood estimation is the best option for Uniformly distributed data with a truncation rate greater than zero.

Figure 3.12



Results when σ^2 is known

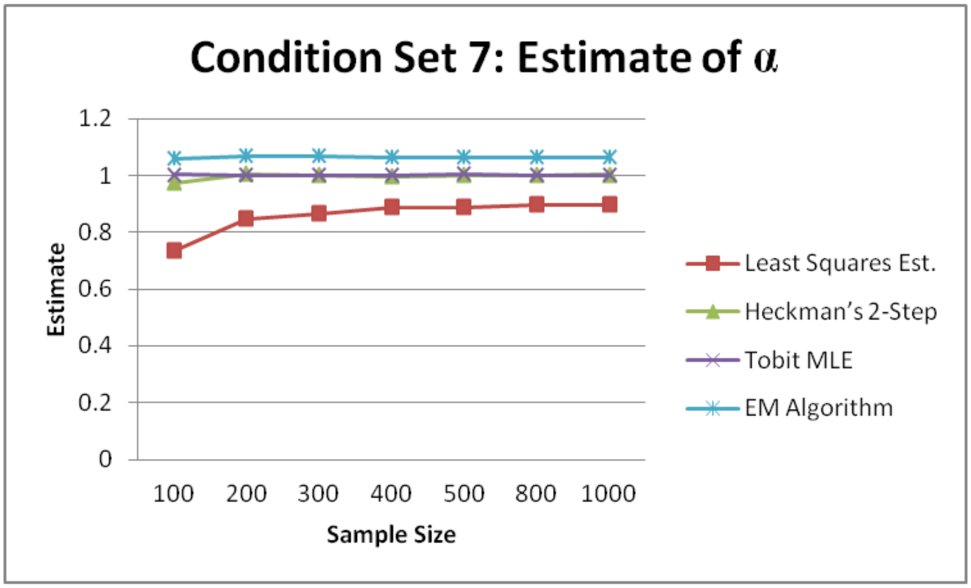
Condition Sets 7 through 12 include estimates for σ. As previously mentioned, Probit maximum likelihood estimation cannot produce estimates for σ. Therefore, only the remaining four methods are used, except when the truncation value Y_0 is zero. When the threshold value is zero, the EM Algorithm can fail to converge and thus was left out of the simulation study.

When performing the simulation study on the next six condition sets, I wanted to answer the following question: does estimating a third parameter alter the effectiveness of the estimation methods? That is, do the estimation methods that work well when σ^2 is known continue to produce good estimates when estimating σ is required?

Condition Set 7

Using a Normally distributed X and a cutoff value of Y_0 = -0.8, I looked at the estimates of α, β and σ. Does adding another unknown parameter affect the truncation rate or the ability for the estimation methods to produce quality estimates? Under the conditions of Condition set 7, I do not believe so. The truncation rates are identical to those in Condition set 1.

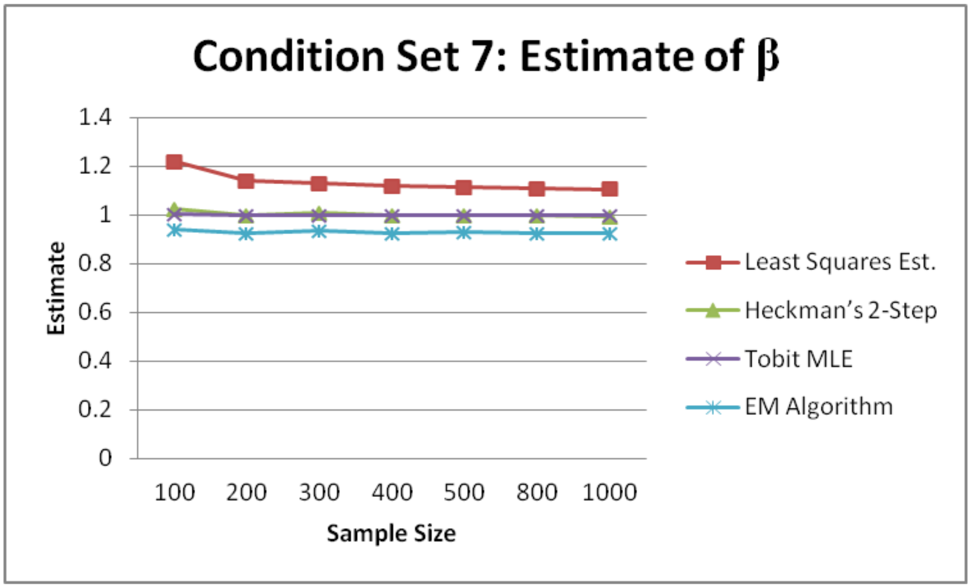
Figure 3.13



As show in figure 3.13 the estimating ability of Heckman's two-step and Tobit MLE are still good. Even the Least squares estimation method is a little more stable than under previous conditions. The EM algorithm produces estimates that fail to be better than either Heckman's or Tobit maximum likelihood estimates. When comparing the two best methods, the Tobit MLE results in the smallest mean squared errors. Heckman's two-step can easily be used as a good estimator of α, however.

Not much changes when looking at the estimates of β. The same two methods are better producers of accurate estimates. Though, the estimates of β and the mean squared errors resulting from these estimates are much improved with Least squares estimates. However, they are still not desirable. As before, the Tobit maximum likelihood estimates are superior to Heckman's by only a little. The smaller MSEs do make this method the best option for estimating β.

Figure 3.14



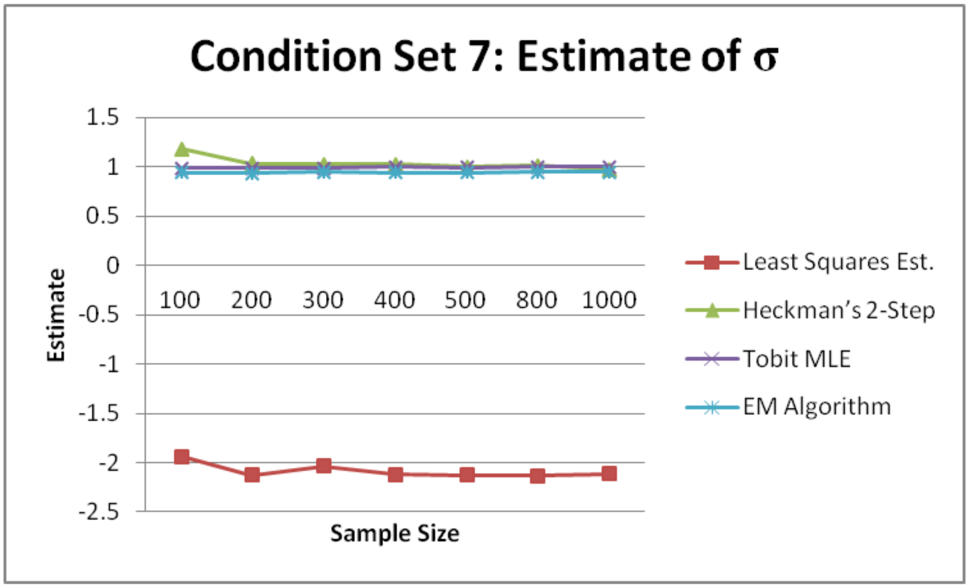
In the estimates for α and β, the precision increases as sample size increases for all methods. This is not true for estimates of σ. In Figure 3.15 one can see that the Least squares method produces under-estimates that do not improve as sample size increases. This immediately makes me believe this method is not a good candidate for estimating under these conditions.

The remaining three methods are still possible options. Using Figure C.8 and Table 7, one can see that the EM algorithm and Heckman's two-step methods do not produce as useful of estimates as the estimates made by Tobit maximum likelihood estimation. While Heckman's estimates are closer to the true value of β, the EM algorithm makes estimates with smaller mean squared errors. Across all sample sizes, Tobit MLE has the most favorable estimates as well as the smallest errors.

Most evident from Figure 3.16 is the inability for the Least squares estimation method to produce favorable estimates of σ. Using Figure C.9 it becomes easier to see that Tobit MLE does indeed have the best estimates even at small sample sizes. Heckman's two-step fails to produce estimates without comparatively large mean squared errors, especially when sample size small.

When considering the necessity to accurately estimate all parameters simultaneously, it's my opinion the Tobit maximum likelihood estimates are the best option. Even at small sample sizes this method can be used.

Figure 3.15



Condition Set 8

Because of the poor estimating ability of the Least squares method, I will refer to tables C.10 through C.12 for this section. This condition set, like Condition set 2 is very important. This is a common scenario for researchers. With a 25% truncation rate, good estimates of α, β and σ could be hard to achieve.

In Figure 3.16 the strange inconsistent nature of the Least squares method is evident. This has been seen under other conditions, but it is very prominent when σ is unknown and the threshold value is zero. As seen in Figure C.10 Heckman's two-step and Tobit maximum likelihood estimation are the best two methods while the EM algorithm fails to improve in accuracy as sample size increases. Heckman's estimates are best at larger sample sizes. However, Tobit's method produces the best estimates of α with the smallest errors.

Figure 3.16

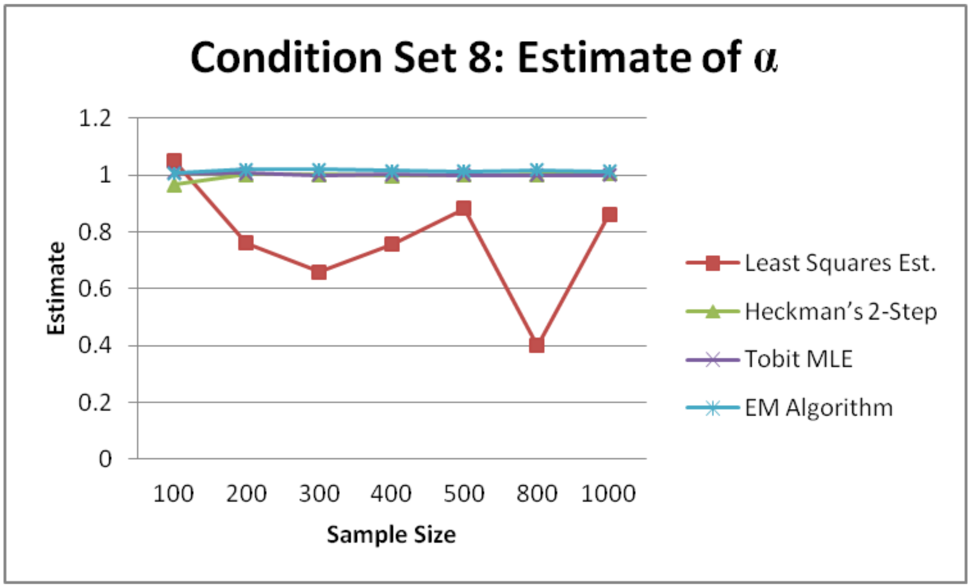
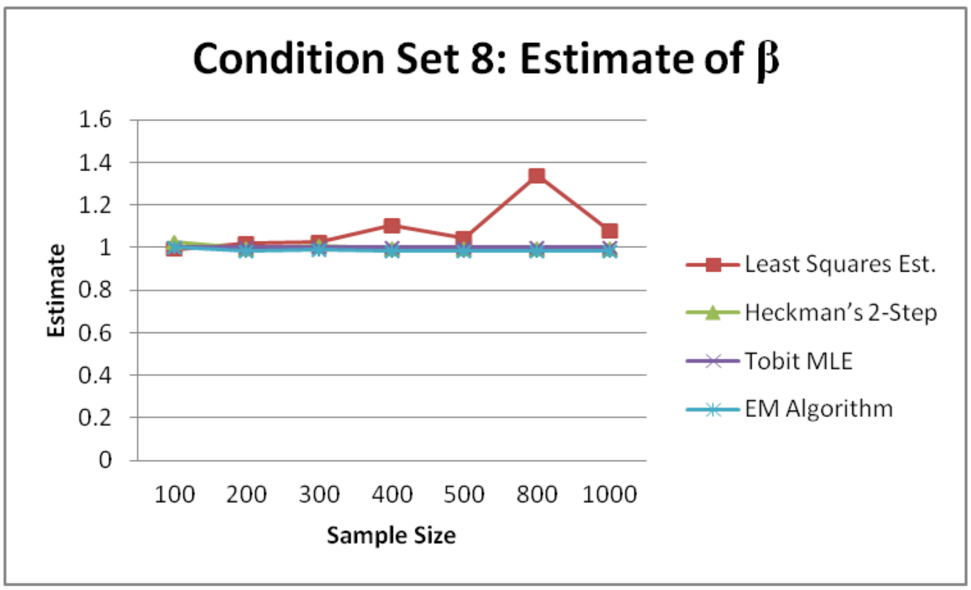
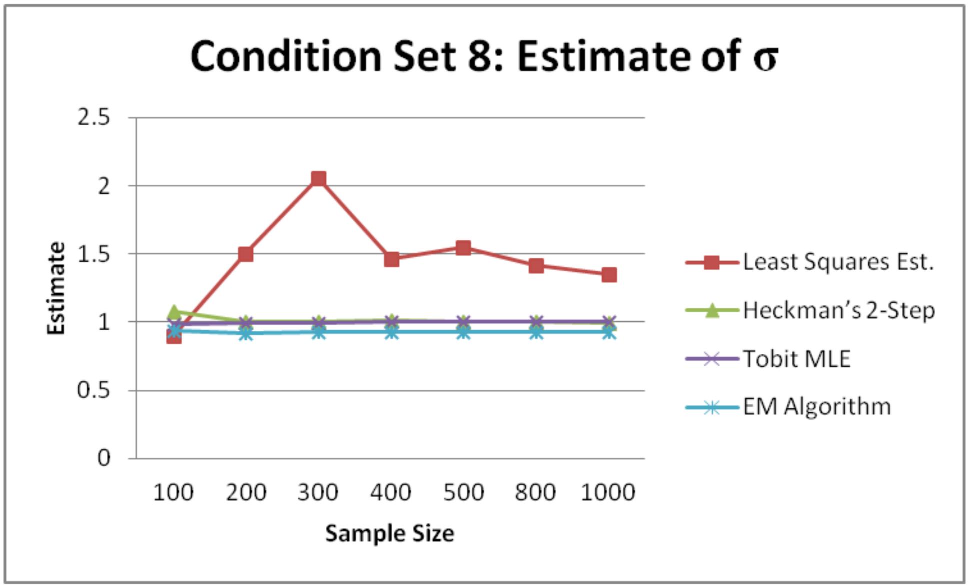


Figure 3.17



The estimates of β are better than estimates of α from the Least squares method. This does not mean they are desirable. Looking at Figure C.11 the best two methods are again the Tobit MLE and Heckman's two-step. And once more, Tobit maximum likelihood estimation is the best overall method for estimating β. To comment on the EM algorithm, the estimations do not improve as sample size increases. This method seams to converge to an estimate close to one, however, it never reaches the true value of β.

Figure 3.18



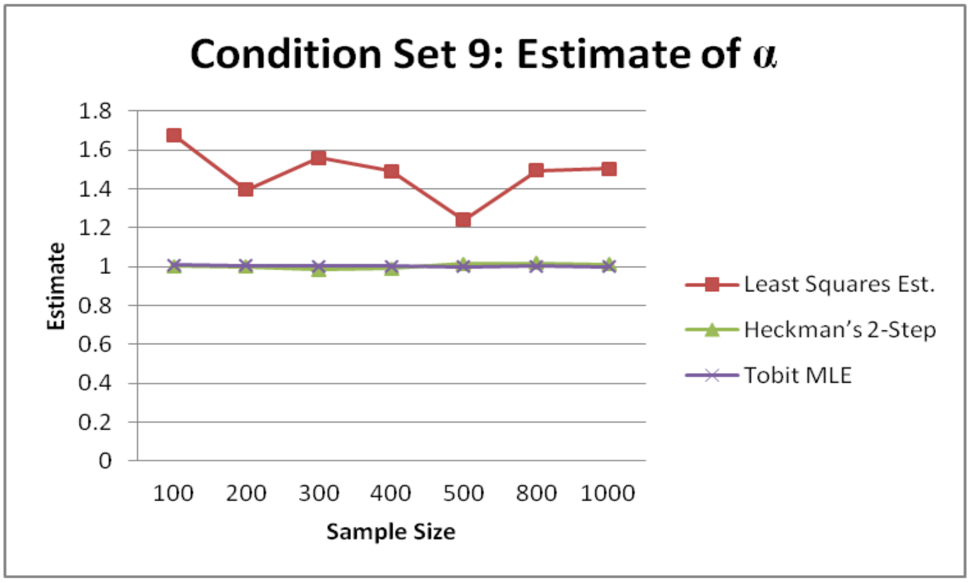
Without commenting on the Least squares method, I immediately refer to Figure C.12. It's immediately evident that the Tobit maximum likelihood method quickly converges to σ = 1. Even at small sample sizes the estimates are very good with small errors. The other methods are not ideal. The EM algorithm stops improving its estimating at sample size n=400. Heckman's two-step under-estimates and then over-estimates σ at the larger sample sizes of n=400, 500, 800 and 1000.

In considering the best method for this condition set, I looked at the estimates as well as the error. Clearly Tobit maximum likelihood estimation is the top method. It produces accurate results, even at small sample sizes. One does not have to trade off inaccurate estimates of one parameter for accurate estimates of another. It's consistently good at estimating the unknowns.

Condition Set 9

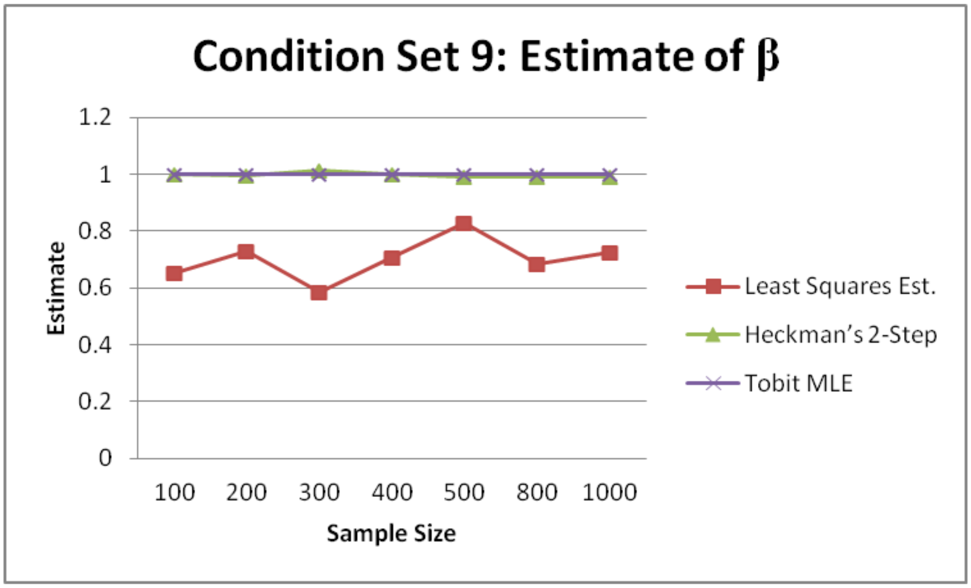
Under condition Set 9 the threshold value is $Y_0 = 1$. This positive cutoff leads to a 50% truncation rate. This high rate does not seem to affect the ability for the Tobit maximum likelihood estimation and Heckman's two-step methods to estimate the parameters. The EM algorithm cannot consistently be invertible, leading to errors. Because of this, I have omitted this method from consideration under the given conditions.

Figure 3.19



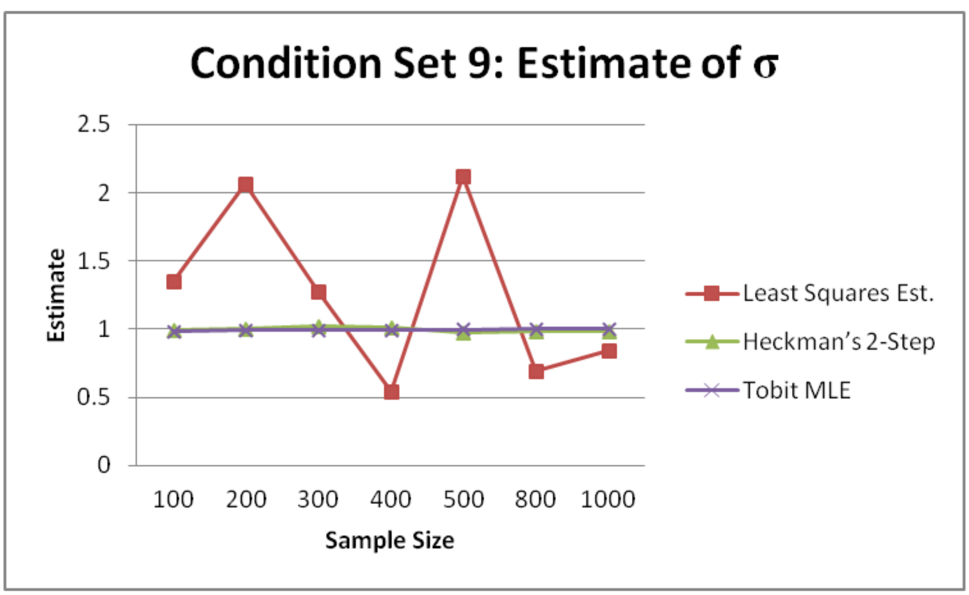
As seen in Figure 3.19 the Least squares estimation method is unreliable—over-estimating α at all sample sizes. Referring to figure C.13 a clearer picture is painted of the behavior of the estimation methods as sample size increases. Tobit maximum likelihood estimation is good even at the smallest sample size of 100. The squared error is also very small at sample size n=200 and greater. This almost immediately shows that when the threshold value is greater than 0, the best method is Tobit maximum likelihood. Heckman's two-step can't seem to converge to a single estimate of α even at large sample sizes. This, accompanied with the large mean squared errors from Table 9, indicate that this method is not the best option.

Figure 3.20



The true behavior of the estimation methods is hard to see from Figure 3.20. Looking at figure C.14, it's easier to understand the methods' effectiveness. Heckman's two-step estimates of β exhibit a similar behavior as the estimates for α. The squared errors are improved over the errors from estimating α, but they remain larger than the Tobit maximum likelihood estimate mean squared errors. The estimates produced by the Tobit MLE method are very close to one, fluctuation only slightly as sample size increases. Again, the squared errors are small.

Figure 3.21

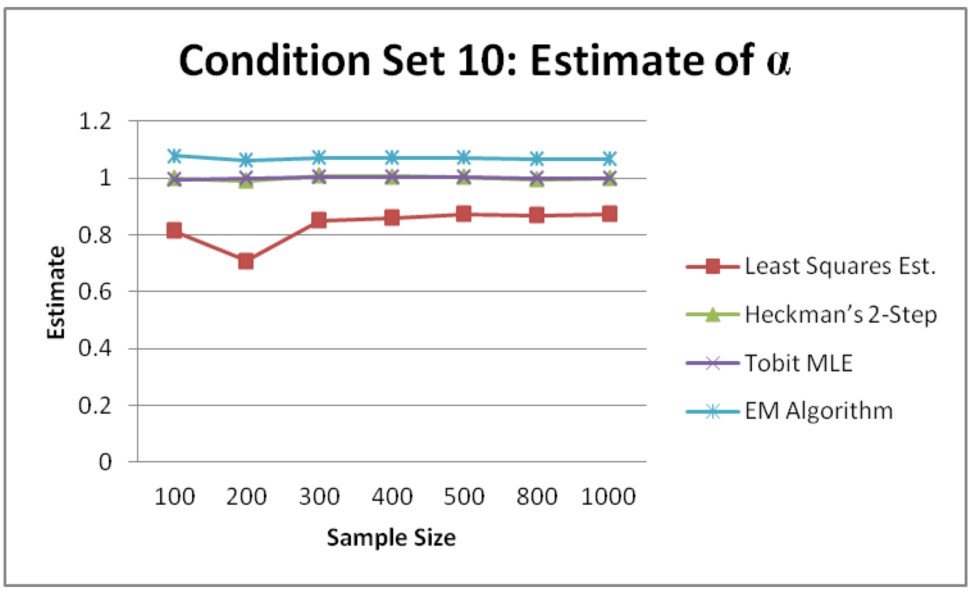


When estimating σ, the Tobit maximum likelihood estimation method approaches one from the left and quickly converges to the true value of one. It does this with the smallest errors of all estimated parameters. Heckman's two-step still produces fine estimates, but they are not superior to those produced by Tobit MLE. The ideal method for estimating all three unknown parameters simultaneously is Tobit maximum likelihood.

Condition Set 10

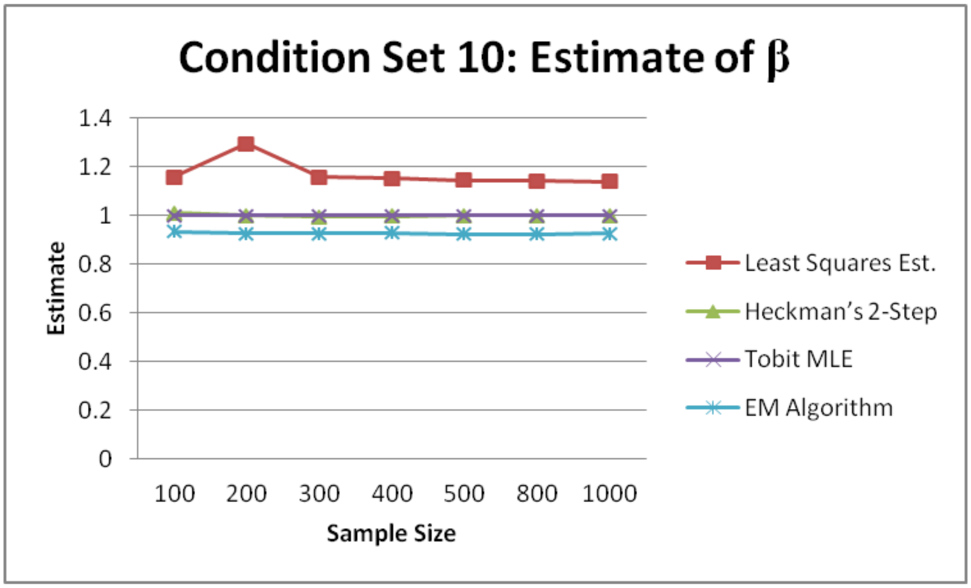
We are again considering condition sets where X is Uniformly distributed. The threshold value is Y_0 = -0.8 which yields a truncation rate of about 10%. As under previous conditions, the Least squares estimates are the least effective at estimating all three unknowns.

Figure 3.22



The estimates of α produced by all four methods converge to a single value as sample size increases. The Least squares estimates are not good compared to the other methods. The errors from these estimates are large at small sample sizes, and are still large in comparison at sample sizes greater than 500. In Figure C.16 I consider only Heckman's two-step and Tobit maximum likelihood estimation methods. From this figure one sees that both methods are good estimators of α. In Table 10, it is evident that between these two, Tobit MLE is the better option only because of the mean squared errors, though I believe either method would do a proficient job at estimating α.

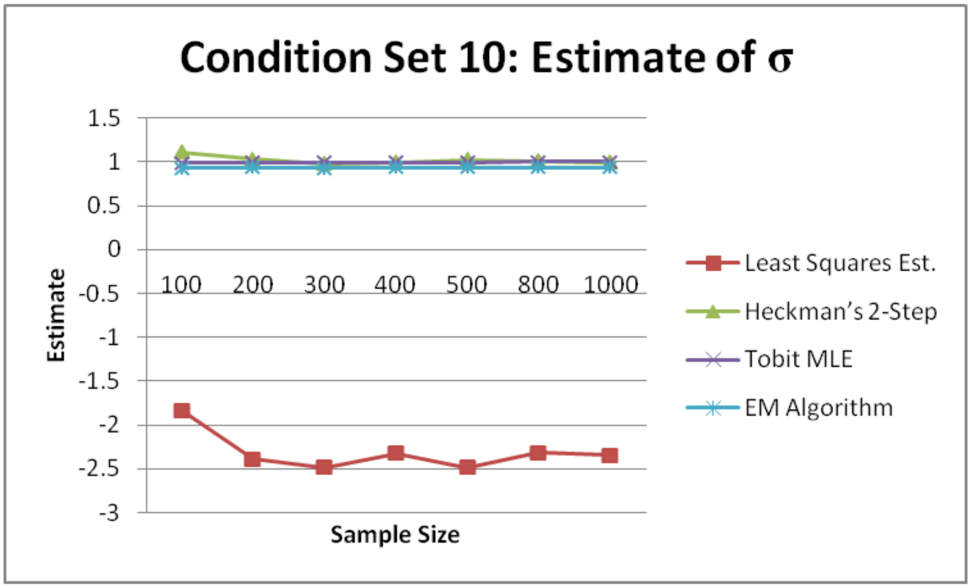
Figure 3.23



Estimates of β follow the patterns for the estimates of α. The Least squares estimation method does not improve, while Heckman's two-step and Tobit maximum likelihood are the best two options. Using Figure C.17, in which only Heckman's and Tobit MLE are pictured, the convergence of the Tobit MLE is apparent. Heckman's two-step is again an adequate estimator and I believe either would be an appropriate choice for estimating β.

The Least squares estimates of σ are very goofy under the given conditions. As shown in Table 10, the mean squared errors are huge at all sample sizes. Figure 3.24 shows just how strange the Least squares estimation method behaves. Though not easily discerned, the EM algorithm produces under-estimates of σ that are less accurate than those produced by Tobit maximum likelihood and Heckman's two-step methods. I do not believe that Heckman's two-step method performs as well when estimating σ compared to the estimates of α and β. Because of this, I think that Tobit MLE is the top choice for estimating σ as well as the other two parameters under these conditions.

Figure 3.24

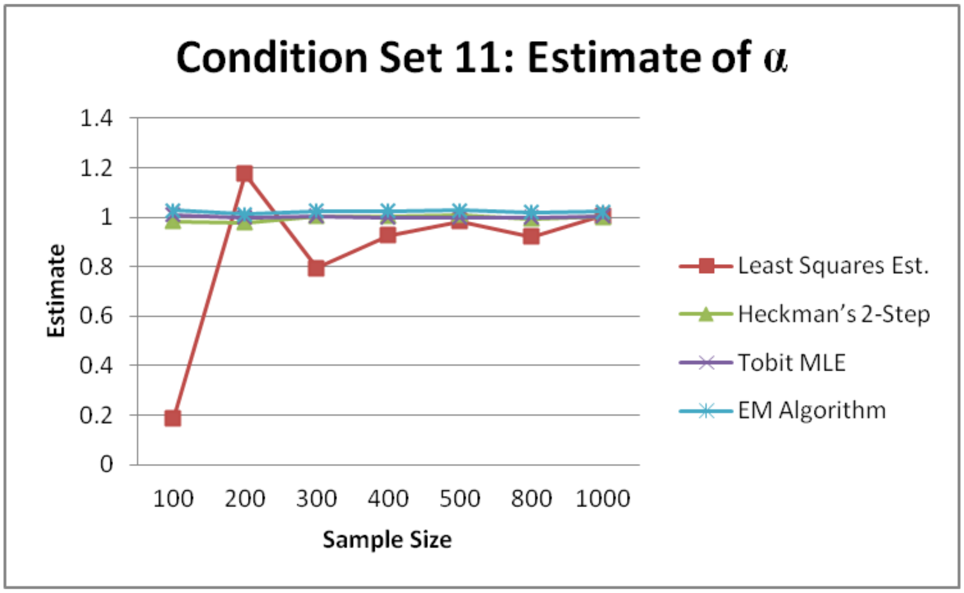


Condition Set 11

Looking at condition set 11, I was not surprised that the Least squares estimates are not desirable compared to all others. As shown in Table 11, the mean squared errors for this method are incredibly large and are an indication that this method cannot be used under these conditions. The usefulness of this method in general will be discussed later. For now, I focus on the other three methods. Here, $Y_0 = 0$ is studied and the consequential truncation rate is approximately 25%.

Figures 3.25 and 3.26 are useful for observing the irregularity of the Least Squares method but do not serve another purpose in this discussion. Figure C.19 displays the estimation performance of the remaining methods in a clearer nature. The Tobit maximum likelihood estimates are again superior, but I first want to discuss the EM algorithm and Heckman's two-step method. At the small sample sizes of n=100, 200 EM algorithm estimates slightly more accurately with smaller errors. At the larger sample sizes, Heckman's method produces better estimates, however, the errors are still larger than the EM algorithm. Neither method produces undesirable estimates, but Tobit maximum likelihood estimates are more accurate and generate smaller errors. Tobit maximum likelihood does a good job of estimating α even at small sample sizes.

Figure 3.25



Relying on the previous statements regarding Least squares estimates as the only discussion necessary, I move on to the remaining three methods. As seen in Table 11 and Figure C.20 the EM algorithm cannot estimate β as well as Heckman's two-step and Tobit maximum likelihood estimation can. However, the mean squared errors resulting from the use of the EM algorithm are relatively smaller than those of Heckman's. Either method would be an adequate second option behind the Tobit maximum likelihood estimator.

Figure 3.26

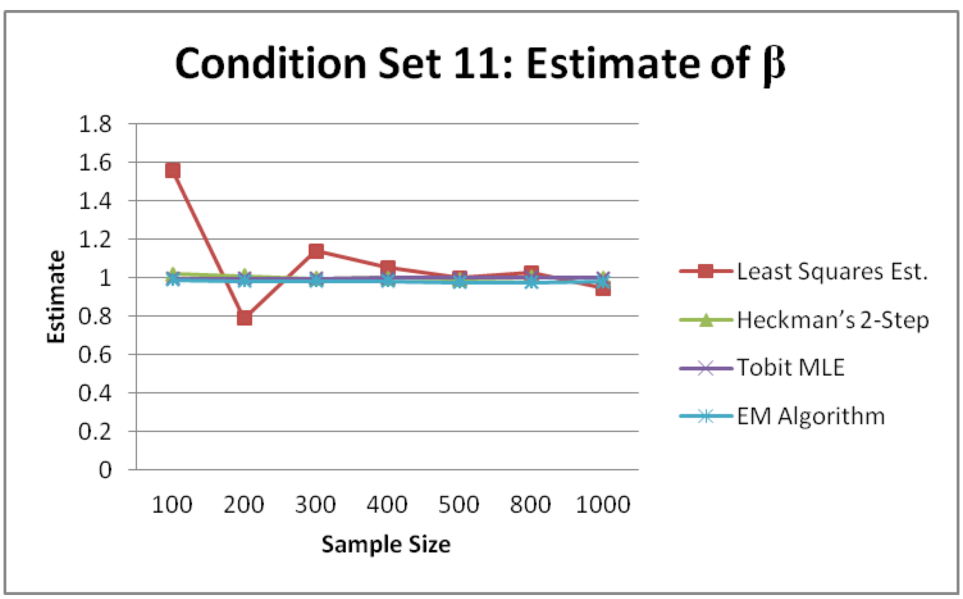
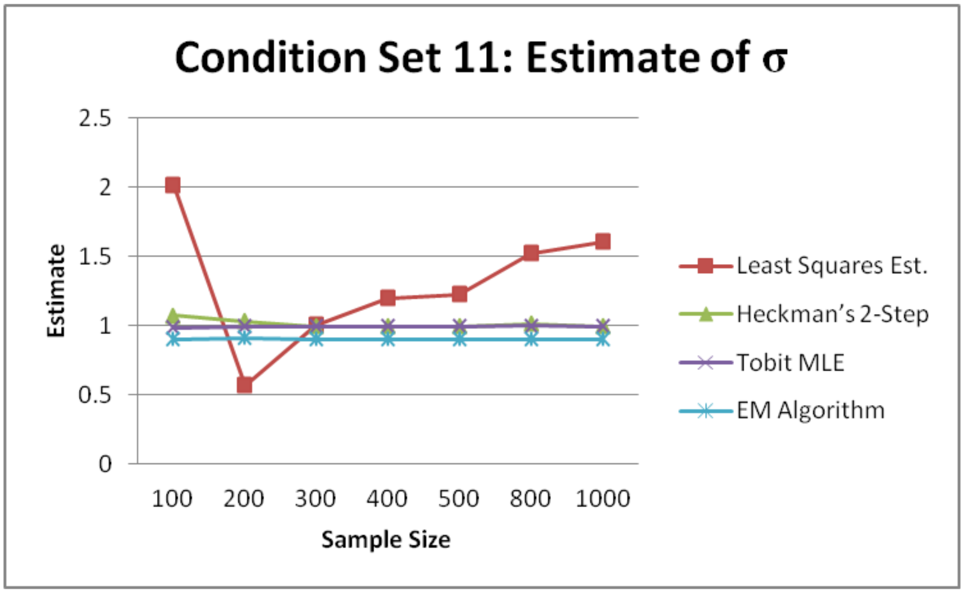


Figure 3.27

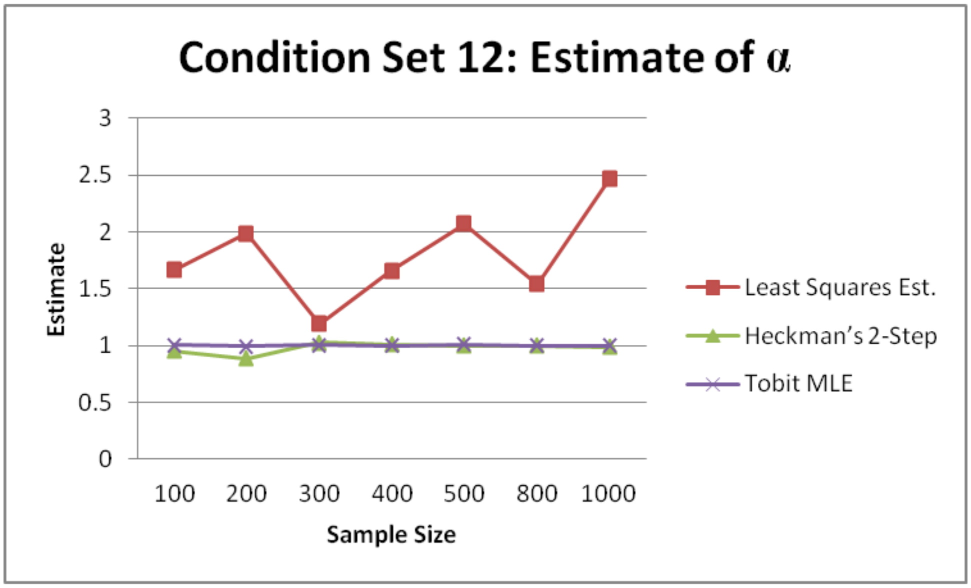


Estimating σ proves to be the strength of Tobit maximum likelihood estimators. The estimates are the most accurate and have the smallest errors. Neither Heckman's two-step, nor the EM algorithm improves their estimates of σ over estimates of α and β. I do not think one can use the EM algorithm or Heckman's two-step to estimate σ when Tobit maximum likelihood estimation is available. Overall, the best method for estimating α, β and σ under the given conditions is Tobit MLE. I do not think there is much of an argument for the other methods when looking at the ability to simultaneously estimate all unknowns well.

Condition Set 12

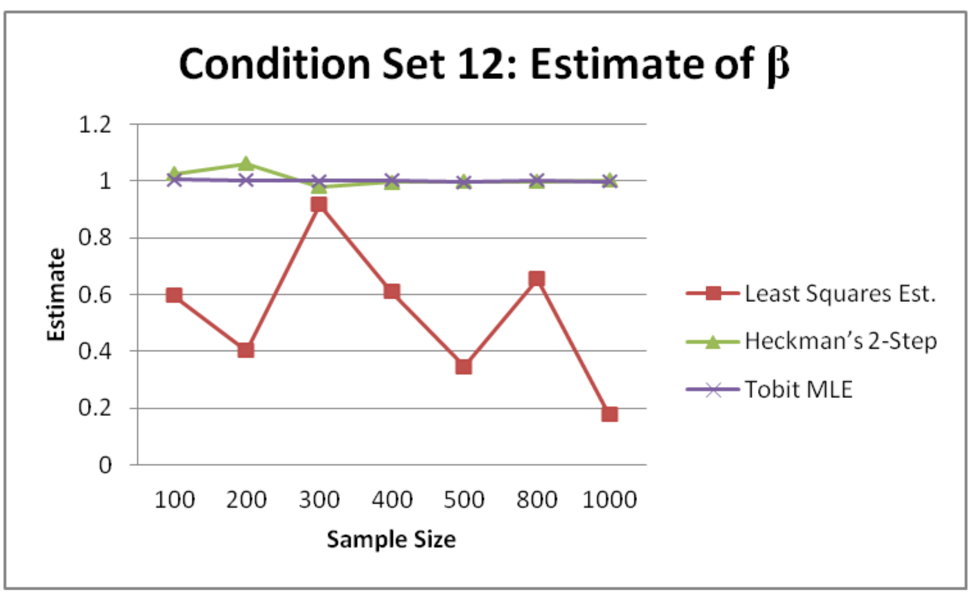
The last set of conditions considered was a Uniformly distributed X with a threshold of $Y_0 = 1$. This positive cutoff leads to a 50% truncation rate amongst the estimation methods. Like Condition set 9, the EM algorithm can become singular, making the calculations impossible. I did not use the EM algorithm under the given conditions. Like much of the results, Least squares estimation does not provide useful estimates of any unknown. Some of the largest mean squared errors seen in the simulation study were exhibited by this method. Because of this, it will not be discussed in the following. The other two methods are discussed next.

Figure 3.28



Though hard to tell from Figure 2.28 both Heckman's 2-step and Tobit maximum likelihood estimation provide good estimates of α. Like under other conditions, Tobit provides slightly better estimates accompanied with smaller squared errors. Both methods reach good estimates of α with sample sizes of n=300. The mean squared errors of Heckman's two-step do improve as sample size gets large. However, the errors are smaller for estimates of β.

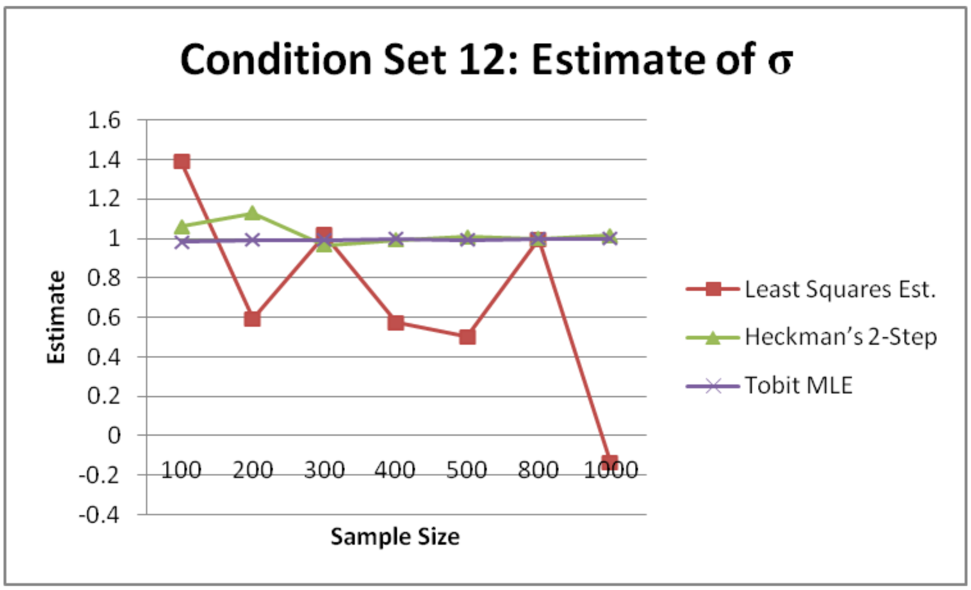
Figure 3.29



Looking at the estimating power of the two methods for β, one sees that both are adequate. However, the errors are again smaller for Tobit maximum likelihood estimates. For the sake of observation, the squared errors produced by Heckman's two-step method are smallest when estimating β at large sample sizes. At small sample sizes, the mean squared errors are too large to make this method useful.

Figure C.23 shows that Heckman's two-step can estimate σ well at large sample sizes, but compared to Tobit maximum likelihood estimation, it doesn't do a good job of producing accurate estimates at sample sizes smaller than 400. The mean squared errors are relatively large for Heckman's method as well.

Figure 3.30



Ranking the estimating capability of the methods by the ability to accurately estimate α, β and σ with small MSEs leads to the conclusion that the best method under Condition set 12 is Tobit maximum likelihood estimation. Heckman's two step can be used with large sample sizes if needed.

Recommendations

Many scenarios were considered in the simulation study. For ease of discussion, I will consider each estimation method individually and then make a few general recommendations.

The Least squares estimation method is not a useful estimation procedure. I make this conclusion because of the complexity of calculations as well as the very large mean squared errors. Throughout the simulation study, the results presented problems. For a smaller sample size, the matrix could become singular and at large sample sizes the matrix was non-singular, but the results were very unstable. In general, when estimates were produced, they were not good. I do not see a need to use least squares estimation when better options are available.

The EM algorithm has its advantages. When σ^2 is known and the threshold value is $Y_0 = 0$, the EM algorithm produces good estimates of α and β with very small mean squared errors. Under Condition set 5 the EM algorithm performed the best of all methods. The major drawback of this method is its computational complexity. That being said, the EM algorithm is best with large sample sizes, but even with small n, the estimates and errors are reasonable.

Heckman's two-step has some good qualities. It is computationally easier than Tobit maximum likelihood estimation. And, under certain conditions, it can produce good estimates with small MSEs. When the threshold value is not positive and sample size is large, Heckman's method can be used with little hesitation.

Probit maximum likelihood estimation was only used when σ^2 was known; except for under Condition set 6. Of the five scenarios it estimated best when the cutoff value was not negative. This method performed well under Condition set 3, where X was Normally distributed and the cut off value was 1. The drawback of Probit maximum likelihood estimation is the inability to be fully efficient. Because it only uses the sign of y_i^* and not the numeric value, this method cannot compete with the other methods.

Lastly, there is Tobit maximum likelihood estimation. This method is clearly ideal for estimating under the conditions of this study. In 11 of the 12 scenarios, Tobit MLEs were the best and had the smallest errors. I do not believe there is a better method for estimating censored and truncated data under these conditions. The numerous applications of Tobit regression across a spectrum of research fields support this opinion.

References

Amemiya, T. (1984). Tobit models: a survey. Journal of Econometrics, (24), 3-61.

Berg, G. D. (1998). Extending powell's semiparametric censored estimator to include non-linear functional forms and extending buchinsk'ys estimation technique (Working Paper No. 98-27). Retrieved from University of Colorado Department of Economics website: http://www.colorado.edu/Economics/papers/papers98/wp98-27.pdf

Caudill, S. B., & Mixon, F. G. (2009). More on testing the normality assumptionin the tobit model. *Journal of Applied Statistics*, *36*(12), 1345-1352.

Delva J, Kaylor AG, Steinhoff E, Shin DE, Siefert K. Using tobit regression analysis to further understand the association of youth alcohol problems with depression and parental factors among Korean adolescent females. J Prev Med Public Health 2007;40:145–149.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm . *Journal of the Royal Statistical Society*, *39*(B), 1-38.

Ekstrand, C., & Carpenter, T. E. (1998). Using a tobit regression model to analyse risk factors for foot-pad dermatitis in commercially grown broilers. *Preventive Veterinary Medicine*, *37*(1), 219-228.

Hartley, M.J. (1976). Estimation of the Tobit model by nonlinear least squares methods (Discussion paper No. 373).

Heckman, J. J. (1979). Sample selection bias as a specification error. *Econometrica*, *47*, 153-161.

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

McDonald, John F., and Robert A. Moffitt. "The uses of Tobit analysis." *Review of Economics and Statistics* 62 (1980): 318+. *Academic OneFile*. Web. 25 Aug. 2011.

Powell, J. L. (1984). Least absolute deviations estimation for the censored regression model. *Journal of Econometrics*, *25*, 303-325.

Shishko, R., Rostker, B., & , (1976). The economics of multiple job holding. The American Economic Review, 66(3), 298-308.

Smith, D. A., & Brame, R. (2003). Tobit models in social science research: Some limitations and a more general alternative. *Sociological Methods and Research*, *31*(1), 364-388.

Solon, G. (2010). A simple microeconomic foundation for a tobit model of consumer demand. *Economics Letters*, *106*, 131-132. Retrieved from www.elsevier.com/locate/ecolet

Stat Data Analysis Examples Tobit Analysis. UCLA: Academic Technology Services, Statistical Consulting Group. Retrieved from http://www.ats.ucla.edu/stat/stata/dae/tobit.htm (accessed January 25, 2012).

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

Wang, L. (2007). A simple nonparametric test for diagnosing nonlinearity in tobit median regression model. *Statistics & Probability Letters*, *77*, 1034–1042.

Zhou, X. (2007). *Semiparametric and nonparametric estimation of tobit models*. (Doctoral dissertation, Hong Kong University of Science and Technology, Hong Kong), Available from ProQuest. (3434086).

Appendix A – Tables

Table A.1

Condition Set 1					
Sample Size n=100					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.9288	0.7323	1.1512	1.0021	1.0542
MSE (a)	1.0744	3.7745	0.0418	0.0092	0.0124
Mean (b)	1.1081	1.2187	0.8424	1.0060	0.9454
MSE (b)	0.1668	1.6576	0.0372	0.0103	0.0134
Trunc. %	0.1017	0.1400	0.1017	0.1018	0.1017
Sample Size n=200					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.8760	0.8475	1.1629	1.0010	1.0627
MSE (a)	0.8268	0.0814	0.0347	0.0055	0.0084
Mean (b)	1.0461	1.1393	0.8322	1.0002	0.9333
MSE (b)	0.0484	0.0501	0.0337	0.0054	0.0092
Trunc. %	0.0993	0.0850	0.0993	0.0997	0.0993
Sample Size n=300					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.8390	0.8656	1.1659	1.0014	1.0628
MSE (a)	0.7456	0.0440	0.0333	0.0039	0.0071
Mean (b)	1.0272	1.1275	0.8360	0.9985	0.9391
MSE (b)	0.0361	0.0323	0.0310	0.0036	0.0070
Trunc. %	0.1013	0.0733	0.1013	0.1006	0.1013
Sample Size n=400					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.8328	0.8855	1.1650	0.9989	1.0595
MSE (a)	0.7199	0.0274	0.0314	0.0027	0.0058
Mean (b)	1.0160	1.1173	0.8308	0.9984	0.9311
MSE (b)	0.0206	0.0236	0.0315	0.0026	0.0072
Trunc. %	0.1002	0.1225	0.1002	0.1015	0.1002
Sample Size n=500					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.8346	0.8863	1.1602	1.0041	1.0570
MSE (a)	0.7155	0.0223	0.0288	0.0019	0.0051
Mean (b)	1.0212	1.1140	0.8314	1.0002	0.9335
MSE (b)	0.0187	0.0201	0.0307	0.0021	0.0063
Trunc. %	0.1006	0.0800	0.1006	0.1007	0.1016
Sample Size n=800					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.8104	0.8974	1.1673	1.0013	1.0601
MSE (a)	0.6685	0.0168	0.0299	0.0013	0.0047
Mean (b)	1.0096	1.1072	0.8297	0.9971	0.9324
MSE (b)	0.0117	0.0157	0.0306	0.0015	0.0059
Trunc. %	0.1014	0.1000	0.1014	0.1015	0.1014
Sample Size n=1000					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.8161	0.8952	1.1625	1.0006	1.0573
MSE (a)	0.6762	0.0155	0.0279	0.0010	0.0041
Mean (b)	1.0119	1.1061	0.8297	1.0001	0.9323
MSE (b)	0.0084	0.0144	0.0302	0.0012	0.0056
Trunc. %	0.1017	0.0920	0.1017	0.1010	0.1017

Table A.2

Condition Set 2					
Sample Size n=100					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0264	1.0509	1.3680	1.0077	0.9920
MSE (a)	0.0417	17.5457	0.1616	0.0101	0.0119
Mean (b)	1.0511	0.9934	0.7127	1.0035	1.0173
MSE (b)	0.0686	11.5432	0.1008	0.0139	0.0140
Trunc. %	0.2407	0.2400	0.2407	0.2372	0.2407
Sample Size n=200					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0347	1.0509	1.3658	1.0052	1.0032
MSE (a)	0.0208	12.3629	0.1453	0.0060	0.0055
Mean (b)	1.0306	1.0189	0.7041	1.0002	1.0020
MSE (b)	0.0277	0.3551	0.0961	0.0067	0.0062
Trunc. %	0.2368	0.2500	0.2368	0.2370	0.2368
Sample Size n=300					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0156	0.6583	1.3774	0.9977	1.0033
MSE (a)	0.0126	12.2833	0.1514	0.0037	0.0039
Mean (b)	1.0196	1.0241	0.7105	1.0033	1.0076
MSE (b)	0.0160	4.6802	0.0895	0.0046	0.0044
Trunc. %	0.2395	0.2467	0.2395	0.2390	0.2395
Sample Size n=400					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0131	0.7560	1.3746	1.0006	0.9996
MSE (a)	0.0097	5.0732	0.1463	0.0031	0.0029
Mean (b)	1.0149	1.1047	0.7042	1.0015	0.9997
MSE (b)	0.0128	0.6163	0.0915	0.0040	0.0032
Trunc. %	0.2386	0.2525	0.2386	0.2390	0.2386
Sample Size n=500					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0073	0.8815	1.3688	0.9991	0.9968
MSE (a)	0.0072	4.1731	0.1410	0.0024	0.0024
Mean (b)	1.0117	1.0455	0.7069	1.0014	1.0022
MSE (b)	0.0108	1.3514	0.0893	0.0025	0.0025
Trunc. %	0.2402	0.2300	0.2402	0.2403	0.2402
Sample Size n=800					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0004	0.3990	1.3737	0.9989	1.0013
MSE (a)	0.0043	82.1690	0.1426	0.0016	0.0014
Mean (b)	0.9994	1.3416	0.7053	0.9998	0.9992
MSE (b)	0.0057	26.2138	0.0891	0.0018	0.0017
Trunc. %	0.2393	0.2288	0.2393	0.2405	0.2393
Sample Size n=1000					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0025	0.8607	1.3737	0.9990	0.9977
MSE (a)	0.0036	0.4431	0.1419	0.0010	0.0010
Mean (b)	1.0088	1.0834	0.7029	1.0020	1.0006
MSE (b)	0.0044	0.2107	0.0900	0.0013	0.0013
Trunc. %	0.2407	0.2440	0.2407	0.2407	0.2407

Table A.3

Condition Set 3					
Sample Size n=100					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.9905	1.6765	1.8227	1.0079	0.5650
MSE (a)	0.0237	1.7185	0.7487	0.0194	0.2159
Mean (b)	1.0663	0.6510	0.5285	1.0000	1.2768
MSE (b)	0.0559	0.5139	0.2699	0.0210	0.1061
Trunc. %	0.4991	0.4500	0.4991	0.4954	0.5009
Sample Size n=200					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0057	1.3947	1.8192	1.0051	0.5832
MSE (a)	0.0108	31.1831	0.7097	0.0095	0.1857
Mean (b)	1.0287	0.7311	0.5364	1.0020	1.2475
MSE (b)	0.0211	8.9383	0.2340	0.0094	0.0734
Trunc. %	0.5012	0.4850	0.5012	0.4986	0.4988
Sample Size n=300					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0156	1.5586	1.8271	1.0029	0.5893
MSE (a)	0.0126	10.7067	0.7045	0.0061	0.1775
Mean (b)	1.0196	0.5870	0.5318	1.0019	1.2494
MSE (b)	0.0160	3.2934	0.2312	0.0071	0.0701
Trunc. %	0.2395	0.4800	0.5013	0.4994	0.4986
Sample Size n=400					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0020	1.4900	1.8161	1.0024	0.5850
MSE (a)	0.0053	16.4612	0.6841	0.0048	0.1784
Mean (b)	1.0055	0.7055	0.5347	0.9993	1.2379
MSE (b)	0.0089	4.8979	0.2265	0.0048	0.0625
Trunc. %	0.5016	0.5250	0.5016	0.4987	0.4984
Sample Size n=500					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.9947	1.2412	1.8323	0.9991	0.5765
MSE (a)	0.0044	37.3577	0.7063	0.0039	0.1848
Mean (b)	1.0154	0.8301	0.5353	1.0015	1.2465
MSE (b)	0.0081	11.2221	0.2237	0.0035	0.0662
Trunc. %	0.4984	0.4920	0.4984	0.4996	0.5016
Sample Size n=800					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0010	1.4948	1.8324	1.0003	0.5880
MSE (a)	0.0026	6.2415	0.7011	0.0023	0.1727
Mean (b)	1.0010	0.6827	0.5311	1.0008	1.2370
MSE (b)	0.0044	5.5079	0.2248	0.0025	0.0593
Trunc. %	0.2393	0.4663	0.5008	0.5000	0.4992
Sample Size n=1000					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.9990	1.5014	1.8264	0.9994	0.5844
MSE (a)	0.0019	7.7428	0.6897	0.0017	0.1749
Mean (b)	1.0028	0.7232	0.5308	0.9981	1.2375
MSE (b)	0.0038	4.8093	0.2242	0.0022	0.0588
Trunc. %	0.4995	0.4960	0.4995	0.5000	0.5005

Table A.4

Condition Set 4					
Sample Size n=100					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.9888	0.8130	1.0619	0.9944	1.0722
MSE (a)	1.2208	2.8999	0.0395	0.0098	0.0138
Mean (b)	1.1302	1.1560	0.9397	0.9983	0.9399
MSE (b)	0.1868	3.2213	0.0671	0.0097	0.0118
Trunc. %	0.1054	0.1200	0.1054	0.1073	0.1054
Sample Size n=200					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.8904	0.7081	1.0601	1.0013	1.0577
MSE (a)	0.8794	8.6356	0.0177	0.0047	0.0077
Mean (b)	1.0684	1.2940	0.9174	0.9993	0.9328
MSE (b)	0.0717	9.0833	0.0339	0.0053	0.0087
Trunc. %	0.1060	0.1000	0.1060	0.1052	0.1060
Sample Size n=300					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.8722	0.8524	1.0650	1.0054	1.0664
MSE (a)	0.8147	0.0474	0.0144	0.0033	0.0075
Mean (b)	1.0478	1.1572	0.9436	0.9994	0.9337
MSE (b)	0.0455	0.0426	0.0215	0.0036	0.0071
Trunc. %	0.1045	0.1000	0.1045	0.1051	0.1045
Sample Size n=400					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.8569	0.8585	1.0664	1.0040	1.0665
MSE (a)	0.7714	0.0501	0.0114	0.0028	0.0065
Mean (b)	1.0455	1.1513	0.9398	0.9985	0.9345
MSE (b)	0.0342	0.0415	0.0169	0.0026	0.0065
Trunc. %	0.1056	0.0975	0.1056	0.1060	0.1056
Sample Size n=500					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.8341	0.8749	1.0659	1.0016	1.0677
MSE (a)	0.7250	0.0271	0.0096	0.0021	0.0064
Mean (b)	1.0171	1.1441	0.9312	1.0029	0.9292
MSE (b)	0.0241	0.0293	0.0162	0.0020	0.0069
Trunc. %	0.1047	0.1140	0.1047	0.1059	0.1047
Sample Size n=800					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.8243	0.8690	1.0603	1.0011	1.0617
MSE (a)	0.6971	0.0256	0.0072	0.0013	0.0049
Mean (b)	1.0189	1.1402	0.9350	0.9991	0.9305
MSE (b)	0.0142	0.0250	0.0111	0.0014	0.0059
Trunc. %	0.1057	0.1163	0.1057	0.1056	0.1057
Sample Size n=1000					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.8168	0.8746	1.0635	1.0009	1.0628
MSE (a)	0.6799	0.0212	0.0069	0.0011	0.0048
Mean (b)	1.0137	1.1391	0.9341	0.9986	0.9318
MSE (b)	0.0107	0.0230	0.0093	0.0010	0.0056
Trunc. %	0.1058	0.0940	0.1058	0.1061	0.1058

Table A.5

Condition Set 5					
Sample Size n=100					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0432	0.1864	1.2776	1.0074	1.0053
MSE (a)	0.0541	241.1589	0.1112	0.0125	0.0107
Mean (b)	1.0431	1.5603	0.7154	0.9959	1.0115
MSE (b)	0.0559	144.7041	0.2112	0.0116	0.0110
Trunc. %	0.2515	0.2500	0.2515	0.2475	0.2515
Sample Size n=200					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0185	1.1760	1.2701	1.0002	0.9904
MSE (a)	0.0217	278.5386	0.0857	0.0059	0.0056
Mean (b)	1.0321	0.7895	0.6928	0.9976	1.0056
MSE (b)	0.0242	104.0293	0.1558	0.0059	0.0056
Trunc. %	0.2525	0.2450	0.2525	0.2507	0.2525
Sample Size n=300					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0219	0.7937	1.2723	1.0031	1.0014
MSE (a)	0.0143	2.3255	0.0830	0.0039	0.0038
Mean (b)	1.0259	1.1405	0.7162	0.9964	1.0046
MSE (b)	0.0170	1.8829	0.1179	0.0046	0.0036
Trunc. %	0.2502	0.2133	0.2502	0.2501	0.2502
Sample Size n=400					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0212	0.9278	1.2713	1.0001	1.0023
MSE (a)	0.0118	2.5133	0.0795	0.0028	0.0026
Mean (b)	1.0214	1.0517	0.7149	0.9993	1.0041
MSE (b)	0.0120	1.1279	0.1072	0.0031	0.0029
Trunc. %	0.2495	0.2350	0.2495	0.2494	0.2495
Sample Size n=500					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.01689	0.9839	1.2777	0.9998	1.0046
MSE (a)	0.00794	2.0925	0.0819	0.0026	0.0023
Mean (b)	1.01075	0.9989	0.7041	0.9983	0.9976
MSE (b)	0.00745	1.6558	0.1096	0.0023	0.0024
Trunc. %	0.2487	0.2260	0.2487	0.2498	0.2487
Sample Size n=800					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.00391	0.9192	1.2681	0.9971	0.9982
MSE (a)	0.00483	0.2830	0.0751	0.0015	0.0013
Mean (b)	1.00175	1.0263	0.7099	1.0024	0.9990
MSE (b)	0.00542	0.2019	0.0991	0.0015	0.0014
Trunc. %	0.2500	0.2363	0.2500	0.2510	0.2500
Sample Size n=1000					
	Probit MLE	**Least Squares Est.**	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.00575	1.0066	1.2737	1.0020	0.9992
MSE (a)	0.00401	4.0095	0.0776	0.0012	0.0011
Mean (b)	1.00808	0.9451	0.7077	0.9978	1.0010
MSE (b)	0.00475	4.4093	0.0969	0.0012	0.0012
Trunc. %	0.2504	0.2530	0.2504	0.2495	0.2504

Table A.6

Condition Set 6				
Sample Size n=100				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.6647	0.9547	1.0040	0.5821
MSE (a)	10.9099	1.6872	0.0177	0.2000
Mean (b)	0.5970	1.0255	1.0045	1.2820
MSE (b)	4.8580	0.5802	0.0189	0.1081
Trunc. %	0.4900	0.5026	0.4977	0.4975
Sample Size n=200				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.9864	0.8844	0.9923	0.5685
MSE (a)	811.9849	0.7441	0.0098	0.1980
Mean (b)	0.4029	1.0622	1.0037	1.2656
MSE (b)	355.4275	0.2804	0.0102	0.0823
Trunc. %	0.5050	0.4973	0.5009	0.5027
Sample Size n=300				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.1901	1.0270	1.0051	0.5869
MSE (a)	92.7502	0.4001	0.0065	0.1790
Mean (b)	0.9166	0.9804	1.0007	1.2626
MSE (b)	41.1026	0.1590	0.0064	0.0759
Trunc. %	0.4767	0.5022	0.4986	0.4978
Sample Size n=400				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.6577	1.0072	0.9980	0.5865
MSE (a)	1.6882	0.3231	0.0044	0.1766
Mean (b)	0.6113	0.9963	1.0033	1.2612
MSE (b)	1.0576	0.1262	0.0048	0.0744
Trunc. %	0.4825	0.5009	0.4998	0.4991
Sample Size n=500				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	2.0711	0.9978	1.0054	0.5928
MSE (a)	94.8792	0.2328	0.0040	0.1705
Mean (b)	0.3461	0.9981	0.9965	1.2520
MSE (b)	39.2993	0.0942	0.0039	0.0682
Trunc. %	0.5140	0.5029	0.4982	0.4971
Sample Size n=800				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.5408	0.9969	0.9972	0.5817
MSE (a)	11.8077	0.1610	0.0024	0.1781
Mean (b)	0.6561	0.9978	1.0027	1.2561
MSE (b)	4.4319	0.0640	0.0026	0.0688
Trunc. %	0.4825	0.4993	0.5005	0.5007
Sample Size n=1000				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	2.4617	0.9876	1.0018	0.5848
MSE (a)	298.1035	0.1229	0.0020	0.1748
Mean (b)	0.1771	1.0038	0.9993	1.2567
MSE (b)	114.3189	0.0477	0.0019	0.0684
Trunc. %	0.5040	0.5002	0.4995	0.4998

Table A.7

Condition Set 7				
Sample Size n=100				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.7323	0.9734	1.0021	1.0586
MSE (a)	3.7745	0.0504	0.0092	0.0127
Mean (b)	1.2187	1.0262	1.0060	0.9396
MSE (b)	1.6576	0.0479	0.0103	0.0140
Mean (sig)	-1.9352	1.1874	0.9912	0.9432
MSE (sig)	21.7195	1.5938	0.0058	0.0080
Trunc. %	0.1400	0.1017	0.1018	0.1017
Sample Size n=200				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8475	1.0023	1.0010	1.0677
MSE (a)	0.0814	0.0181	0.0055	0.0089
Mean (b)	1.1393	0.9971	1.0002	0.9267
MSE (b)	0.0501	0.0203	0.0054	0.0101
Mean (sig)	-2.1236	1.0343	0.9912	0.9381
MSE (sig)	18.9400	0.5189	0.0030	0.0060
Trunc. %	0.0850	0.0993	0.0997	0.0993
Sample Size n=300				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8656	1.0012	1.0014	1.0675
MSE (a)	0.0440	0.0107	0.0039	0.0076
Mean (b)	1.1275	1.0068	0.9985	0.9332
MSE (b)	0.0323	0.0129	0.0036	0.0078
Mean (sig)	-2.0332	1.0253	0.9942	0.9443
MSE (sig)	10.9246	0.2602	0.0021	0.0046
Trunc. %	0.0733	0.1013	0.1006	0.1013
Sample Size n=400				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8855	0.9965	0.9989	1.0641
MSE (a)	0.0274	0.0093	0.0027	0.0064
Mean (b)	1.1173	1.0002	0.9984	0.9252
MSE (b)	0.0236	0.0106	0.0026	0.0080
Mean (sig)	-2.1166	1.0335	0.9972	0.9435
MSE (sig)	11.0884	0.2265	0.0015	0.0043
Trunc. %	0.1225	0.1002	0.1015	0.1002
Sample Size n=500				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8863	0.9984	1.0041	1.0617
MSE (a)	0.0223	0.0065	0.0019	0.0056
Mean (b)	1.1140	0.9987	1.0002	0.9276
MSE (b)	0.0201	0.0075	0.0021	0.0071
Mean (sig)	-2.1242	1.0039	0.9956	0.9440
MSE (sig)	10.2163	0.1550	0.0012	0.0040
Trunc. %	0.0800	0.1006	0.1007	0.1006
Sample Size n=800				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8974	0.9995	1.0013	1.0643
MSE (a)	0.0168	0.0042	0.0013	0.0052
Mean (b)	1.1072	0.9998	0.9971	0.9271
MSE (b)	0.0157	0.0051	0.0015	0.0066
Mean (sig)	-2.1267	1.0149	0.9989	0.9503
MSE (sig)	10.0923	0.0989	0.0007	0.0030
Trunc. %	0.1000	0.1014	0.1015	0.1014
Sample Size n=1000				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8952	1.0017	1.0006	1.0617
MSE (a)	0.0155	0.0030	0.0010	0.0046
Mean (b)	1.1061	0.9954	1.0001	0.9267
MSE (b)	0.0144	0.0037	0.0012	0.0064
Mean (sig)	-2.1112	0.9817	0.9978	0.9475
MSE (sig)	9.9008	0.0747	0.0006	0.0032
Trunc. %	0.0920	0.1017	0.1010	0.1017

Table A.8

Condition Set 8				
Sample Size n=100				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0509	0.9656	1.0077	1.0054
MSE (a)	17.5457	0.1533	0.0101	0.0122
Mean (b)	0.9934	1.0278	1.0035	1.0034
MSE (b)	11.5432	0.1002	0.0139	0.0152
Mean (sig)	0.8962	1.0785	0.9820	0.9331
MSE (sig)	82.5837	0.9276	0.0069	0.0112
Trunc. %	1.0509	0.2407	0.2372	0.2407
Sample Size n=200				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.7608	1.0015	1.0052	1.0196
MSE (a)	12.3629	0.0572	0.0060	0.0058
Mean (b)	1.0189	0.9967	1.0002	0.9850
MSE (b)	0.3551	0.0413	0.0067	0.0072
Mean (sig)	1.5011	1.0063	0.9914	0.9215
MSE (sig)	24.5965	0.3561	0.0032	0.0090
Trunc. %	0.7608	0.2368	0.2370	0.2368
Sample Size n=300				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.6583	1.0021	0.9977	1.0190
MSE (a)	12.2833	0.0343	0.0037	0.0041
Mean (b)	1.0241	1.0061	1.0033	0.9915
MSE (b)	4.6802	0.0268	0.0046	0.0047
Mean (sig)	2.0576	1.0069	0.9934	0.9269
MSE (sig)	436.9739	0.2155	0.0025	0.0073
Trunc. %	0.6583	0.2395	0.2390	0.2395
Sample Size n=400				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.7560	0.9973	1.0006	1.0156
MSE (a)	5.0732	0.0300	0.0031	0.0032
Mean (b)	1.1047	0.9981	1.0015	0.9833
MSE (b)	0.6163	0.0223	0.0040	0.0038
Mean (sig)	1.4622	1.0079	0.9993	0.9258
MSE (sig)	17.2744	0.1782	0.0017	0.0070
Trunc. %	0.7560	0.2386	0.2390	0.2386
Sample Size n=500				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8815	0.9981	0.9991	1.0127
MSE (a)	4.1731	0.0216	0.0024	0.0025
Mean (b)	1.0455	0.9994	1.0014	0.9860
MSE (b)	1.3514	0.0160	0.0025	0.0029
Mean (sig)	1.5469	0.9984	0.9986	0.9273
MSE (sig)	9.4372	0.1284	0.0014	0.0064
Trunc. %	0.8815	0.2402	0.2403	0.2402
Sample Size n=800				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.3990	1.0015	0.9989	1.0165
MSE (a)	82.1690	0.0136	0.0016	0.0017
Mean (b)	1.3416	0.9982	0.9998	0.9837
MSE (b)	26.2138	0.0102	0.0018	0.0021
Mean (sig)	1.4164	1.0012	0.9991	0.9302
MSE (sig)	174.3011	0.0800	0.0009	0.0056
Trunc. %	0.3990	0.2393	0.2405	0.2393
Sample Size n=1000				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8607	1.0030	0.9990	1.0132
MSE (a)	0.4431	0.0106	0.0010	0.0012
Mean (b)	1.0834	0.9949	1.0020	0.9848
MSE (b)	0.2107	0.0080	0.0013	0.0016
Mean (sig)	1.3545	0.9889	0.9990	0.9298
MSE (sig)	23.1927	0.0646	0.0007	0.0055
Trunc. %	0.8607	0.2407	0.2407	0.2407

Table A.9

Condition Set 9			
Sample Size n=100			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	1.6765	1.0008	1.0079
MSE (a)	1.7185	0.9525	0.0194
Mean (b)	0.6510	1.0004	1.0000
MSE (b)	0.5139	0.3285	0.0210
Mean (sig)	1.3490	0.9890	0.9804
MSE (sig)	59.3216	1.2465	0.0100
Trunc. %	0.4500	0.4991	0.4954
Sample Size n=200			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	1.3947	0.9990	1.0051
MSE (a)	31.1831	0.4430	0.0095
Mean (b)	0.7311	0.9961	1.0020
MSE (b)	8.9383	0.1536	0.0094
Mean (sig)	2.0660	1.0017	0.9919
MSE (sig)	135.8685	0.5991	0.0055
Trunc. %	0.4850	0.5012	0.4986
Sample Size n=300			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	1.5586	0.9831	1.0029
MSE (a)	10.7067	0.2455	0.0061
Mean (b)	0.5870	1.0154	1.0019
MSE (b)	3.2934	0.0904	0.0071
Mean (sig)	1.2737	1.0250	0.9971
MSE (sig)	198.4167	0.3370	0.0037
Trunc. %	0.4800	0.5013	0.4994
Sample Size n=400			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	1.4900	0.9890	1.0024
MSE (a)	16.4612	0.2013	0.0048
Mean (b)	0.7055	1.0010	0.9993
MSE (b)	4.8979	0.0740	0.0048
Mean (sig)	0.5378	1.0102	0.9964
MSE (sig)	128.5356	0.2713	0.0029
Trunc. %	0.5250	0.5016	0.4987
Sample Size n=500			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	1.2412	1.0137	0.9991
MSE (a)	37.3577	0.1414	0.0039
Mean (b)	0.8301	0.9918	1.0015
MSE (b)	11.2221	0.0531	0.0035
Mean (sig)	2.1237	0.9767	0.9978
MSE (sig)	197.2982	0.1890	0.0023
Trunc. %	0.4920	0.4984	0.4996
Sample Size n=800			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	1.4948	1.0174	1.0003
MSE (a)	6.2415	0.0915	0.0023
Mean (b)	0.6827	0.9891	1.0008
MSE (b)	5.5079	0.0334	0.0025
Mean (sig)	0.6902	0.9829	1.0004
MSE (sig)	121.9921	0.1235	0.0013
Trunc. %	0.4663	0.5008	0.5000
Sample Size n=1000			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	1.5014	1.0107	0.9994
MSE (a)	7.7428	0.0823	0.0017
Mean (b)	0.7232	0.9900	0.9981
MSE (b)	4.8093	0.0300	0.0022
Mean (sig)	0.8410	0.9862	0.9990
MSE (sig)	82.1504	0.1116	0.0011
Trunc. %	0.4960	0.4995	0.5000

Table A.10

Condition Set 10				
Sample Size n=100				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8130	0.9981	0.9944	1.0785
MSE (a)	2.8999	0.0635	0.0098	0.0145
Mean (b)	1.1560	1.0103	0.9983	0.9324
MSE (b)	3.2213	0.0549	0.0097	0.0125
Mean (sig)	-1.8363	1.1130	0.9863	0.9276
MSE (sig)	23.5156	1.6579	0.0057	0.0100
Trunc. %	0.1200	0.1054	0.1073	0.1054
Sample Size n=200				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.7081	0.9920	1.0013	1.0637
MSE (a)	8.6356	0.0270	0.0047	0.0083
Mean (b)	1.2940	0.9990	0.9993	0.9257
MSE (b)	9.0833	0.0280	0.0053	0.0096
Mean (sig)	-2.3901	1.0325	0.9922	0.9324
MSE (sig)	16.3463	0.7349	0.0030	0.0068
Trunc. %	0.1000	0.1060	0.1052	0.1060
Sample Size n=300				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8524	1.0078	1.0054	1.0725
MSE (a)	0.0474	0.0200	0.0033	0.0082
Mean (b)	1.1572	0.9947	0.9994	0.9266
MSE (b)	0.0426	0.0186	0.0036	0.0080
Mean (sig)	-2.4844	0.9822	0.9942	0.9306
MSE (sig)	14.4509	0.4749	0.0019	0.0061
Trunc. %	0.1000	0.1045	0.1051	0.1045
Sample Size n=400				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8585	1.0063	1.0040	1.0724
MSE (a)	0.0501	0.0131	0.0028	0.0072
Mean (b)	1.1513	0.9972	0.9985	0.9276
MSE (b)	0.0415	0.0128	0.0026	0.0074
Mean (sig)	-2.3248	0.9908	0.9968	0.9338
MSE (sig)	12.1332	0.3178	0.0016	0.0055
Trunc. %	0.0975	0.1056	0.1060	0.1056
Sample Size n=500				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8749	1.0025	1.0040	1.0735
MSE (a)	0.0271	0.0099	0.0028	0.0072
Mean (b)	1.1441	0.9980	0.9985	0.9225
MSE (b)	0.0293	0.0116	0.0026	0.0078
Mean (sig)	-2.4860	1.0221	0.9968	0.9345
MSE (sig)	13.1248	0.2669	0.0016	0.0052
Trunc. %	0.1140	0.1047	0.1060	0.1047
Sample Size n=800				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8690	0.9964	1.0011	1.0674
MSE (a)	0.0256	0.0074	0.0013	0.0056
Mean (b)	1.1402	0.9997	0.9991	0.9240
MSE (b)	0.0250	0.0073	0.0014	0.0068
Mean (sig)	-2.3176	1.0126	1.0004	0.9368
MSE (sig)	11.3288	0.1790	0.0007	0.0045
Trunc. %	0.1163	0.1057	0.1056	0.1057
Sample Size n=1000				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.8746	1.0006	1.0009	1.0686
MSE (a)	0.0212	0.0055	0.0011	0.0055
Mean (b)	1.1391	0.9982	0.9986	0.9251
MSE (b)	0.0230	0.0053	0.0010	0.0065
Mean (sig)	-2.3466	0.9961	1.0000	0.9359
MSE (sig)	11.4365	0.1248	0.0005	0.0046
Trunc. %	0.0940	0.1058	0.1061	0.1058

Table A.11

Condition Set 11				
Sample Size n=100				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.1864	0.9813	1.0074	1.0268
MSE (a)	241.1589	0.2047	0.0125	0.0117
Mean (b)	1.5603	1.0197	0.9959	0.9897
MSE (b)	144.7041	0.1287	0.0116	0.0115
Mean (sig)	2.0188	1.0741	0.9848	0.9047
MSE (sig)	125.2262	1.0800	0.0071	0.0148
Trunc. %	0.2500	0.2515	0.2475	0.2515
Sample Size n=200				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.1760	0.9789	1.0002	1.0119
MSE (a)	278.5386	0.0864	0.0059	0.0057
Mean (b)	0.7895	1.0084	0.9976	0.9839
MSE (b)	104.0293	0.0610	0.0059	0.0062
Mean (sig)	0.5697	1.0324	0.9957	0.9084
MSE (sig)	192.8885	0.4726	0.0038	0.0112
Trunc. %	0.2450	0.2525	0.2507	0.2525
Sample Size n=300				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.7937	1.0044	1.0031	1.0242
MSE (a)	2.3255	0.0577	0.0039	0.0044
Mean (b)	1.1405	0.9968	0.9964	0.9816
MSE (b)	1.8829	0.0395	0.0046	0.0042
Mean (sig)	1.0062	0.9951	0.9978	0.9024
MSE (sig)	63.0037	0.3094	0.0022	0.0111
Trunc. %	0.2133	0.2502	0.2501	0.2502
Sample Size n=400				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.9278	1.0031	1.0001	1.0244
MSE (a)	2.5133	0.0373	0.0028	0.0031
Mean (b)	1.0517	1.0000	0.9993	0.9816
MSE (b)	1.1279	0.0275	0.0031	0.0034
Mean (sig)	1.1982	0.9973	0.9950	0.9052
MSE (sig)	35.2919	0.2025	0.0020	0.0104
Trunc. %	0.2350	0.2495	0.2494	0.2495
Sample Size n=500				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.9839	1.0086	0.9998	1.0272
MSE (a)	2.0925	0.0279	0.0026	0.0029
Mean (b)	0.9989	0.9920	0.9983	0.9748
MSE (b)	1.6558	0.0224	0.0023	0.0032
Mean (sig)	1.2263	0.9930	0.9952	0.9029
MSE (sig)	51.0299	0.1669	0.0014	0.0105
Trunc. %	0.2260	0.2487	0.2498	0.2487
Sample Size n=800				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	0.9192	0.9927	0.9971	1.0203
MSE (a)	0.2830	0.0232	0.0015	0.0017
Mean (b)	1.0263	1.0023	1.0024	0.9768
MSE (b)	0.2019	0.0162	0.0015	0.0020
Mean (sig)	1.5203	1.0131	0.9997	0.9061
MSE (sig)	6.0143	0.1291	0.0010	0.0095
Trunc. %	0.2363	0.2500	0.2510	0.2500
Sample Size n=1000				
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**	**EM Algorithm**
Mean (a)	1.0066	1.0010	1.0020	1.0216
MSE (a)	4.0095	0.0164	0.0012	0.0015
Mean (b)	0.9451	0.9979	0.9978	0.9783
MSE (b)	4.4093	0.0115	0.0012	0.0017
Mean (sig)	1.6104	0.9960	0.9981	0.9048
MSE (sig)	12.0106	0.0863	0.0006	0.0096
Trunc. %	0.2530	0.2504	0.2495	0.2504

Table A.12

Condition Set 12			
Sample Size n=100			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	1.6647	0.9547	1.0040
MSE (a)	10.9099	1.6872	0.0177
Mean (b)	0.5970	1.0255	1.0045
MSE (b)	4.8580	0.5802	0.0189
Mean (sig)	1.3880	1.0609	0.9827
MSE (sig)	23.0105	2.1549	0.0112
Trunc. %	0.4900	0.4974	0.4977
Sample Size n=200			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	1.9864	0.8844	0.9923
MSE (a)	811.9849	0.7441	0.0098
Mean (b)	0.4029	1.0622	1.0037
MSE (b)	355.4275	0.2804	0.0102
Mean (sig)	0.5917	1.1270	0.9922
MSE (sig)	737.4882	0.9193	0.0055
Trunc. %	0.5050	0.5027	0.5009
Sample Size n=300			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	1.1901	1.0270	1.0051
MSE (a)	92.7502	0.4001	0.0065
Mean (b)	0.9166	0.9804	1.0007
MSE (b)	41.1026	0.1590	0.0064
Mean (sig)	1.0162	0.9675	0.9910
MSE (sig)	31.2024	0.5167	0.0036
Trunc. %	0.4767	0.4978	0.4986
Sample Size n=400			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	1.6577	1.0072	0.9980
MSE (a)	1.6882	0.3231	0.0044
Mean (b)	0.6113	0.9963	1.0033
MSE (b)	1.0576	0.1262	0.0048
Mean (sig)	0.5726	0.9934	0.9992
MSE (sig)	50.6474	0.4101	0.0031
Trunc. %	0.4825	0.4991	0.4998
Sample Size n=500			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	2.0711	0.9978	1.0054
MSE (a)	94.8792	0.2328	0.0040
Mean (b)	0.3461	0.9981	0.9965
MSE (b)	39.2993	0.0942	0.0039
Mean (sig)	0.5025	1.0072	0.9917
MSE (sig)	174.4657	0.3105	0.0025
Trunc. %	0.5140	0.4971	0.4982
Sample Size n=800			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	1.5408	0.9969	0.9972
MSE (a)	11.8077	0.1610	0.0024
Mean (b)	0.6561	0.9978	1.0027
MSE (b)	4.4319	0.0640	0.0026
Mean (sig)	0.9939	0.9981	0.9972
MSE (sig)	308.8809	0.2039	0.0014
Trunc. %	0.4825	0.5007	0.5005
Sample Size n=1000			
	Least Squares Est.	**Heckman's 2-Step**	**Tobit MLE**
Mean (a)	2.4617	0.9876	1.0018
MSE (a)	298.1035	0.1229	0.0020
Mean (b)	0.1771	1.0038	0.9993
MSE (b)	114.3189	0.0477	0.0019
Mean (sig)	-0.1347	1.0142	0.9988
MSE (sig)	1011.6760	0.1552	0.0011
Trunc. %	0.5040	0.4998	0.4995

Appendix B – R Program

Condition Set 2

```
#Tobit MLE
set.seed(987654)
library(VGAM)
low=0;
Ma=Mb=Msig=Lpe=rep(0,7)
MSa=MSb=MSsig=rep(0,7)
j=1;
for(n in c(100,200,300,400,500,800,1000))
{
        a=b=sig=lowp=rep(0,500)
        for(k in seq(500))
        {
                x=rnorm(n,0,1)
                ystar=1+x+rnorm(n)
                y=pmax(ystar,low)
                fit=vglm(y~x, tobit(Lower=low))
                lowp[k]=sum(y==low)/n
                table(fit@extra$censoredL)
                a[k]=coef(fit,matrix=TRUE)[1,1]
                b[k]=coef(fit,matrix=TRUE)[2,1]
                sig[k]=1

        }

        Ma[j]=mean(a)
        Mb[j]=mean(b)
        Msig[j]=mean(sig)
        MSa[j]=mean((a-1)^2)
        MSb[j]=mean((b-1)^2)
        MSsig[j]=mean((sig-1)^2)
        Lpe[j]=mean(lowp)
        j=j+1
}
results=rbind(Ma,MSa, Mb, MSb, Msig, MSsig, Lpe)
dimnames(results)=list(c("Mean(a)", "MSE(a)", "Mean(b)", "MSE(b)", "Mean(sig)", "MSE(sig)", "Trunc.Pt"),
c(100, 200, 300, 400, 500, 800, 1000))

results
```

#Probit MLE

```
set.seed(987654)
low=0;
Ma=Mb=Msig=Lpe=rep(0,7)
MSa=MSb=MSs=rep(0,7)
j=1;
for(n in c(100,200,300,400,500,800,1000))
{
	a=b=sig=lowp=rep(0,500)
	for(k in seq(500))
	{
		x=rnorm(n,0,1)
		ystar=1+x+rnorm(n)
		y=(ystar>=low)
		fit=glm(y~x, family=binomial(link="probit"))
		lowp[k]=sum(y==low)/n
		a[k]=coef(fit)[1]
		b[k]=coef(fit)[2]

	}

	Ma[j]=mean(a)
	Mb[j]=mean(b)
	MSa[j]=mean((a-1)^2)
	MSb[j]=mean((b-1)^2)
	Lpe[j]=mean(lowp)
	j=j+1;
}
results=rbind(Ma,MSa, Mb, MSb, Lpe)
dimnames(results)=list(c("Mean(a)", "MSE(a)", "Mean(b)", "MSE(b)", "Trunc.Pt"), c(100, 200, 300, 400, 500,
800, 1000))

results

#Theoretical Truncation Rate
f=function(x){pnorm(-1-x)};
integrate(f, lower=-1, upper=1)$value/2
```

 # LSE Positive

```
 set.seed(987654)
 total=500
 ma=msa=mb=msb=Lpe=rep(0,7)
 j=1;
 for( n in c(100, 200, 300, 400, 500, 800, 1000))
{
        alp=bet=rep(0,total)
        for(i in seq(total))
        {
         repeat{
                        lowp=0;
                        low=0;
                        se=1;
                        x=rnorm(n,0,1)
                        ystar=1+x+rnorm(n,0,se)
                        y=ystar[ystar>=low]
                        x=x[ystar>=low];
                        lowp=length(ystar[ystar<low])/n
                        length(y);
                        length(x);
                        a=0.95;
                        b=0.95;

                        for(k in seq(10))
                        {
                                z=(a+b*x)/1;
                                laz=dnorm(z)/pnorm(z);
                                laz1=-z*laz-laz^2;
                                 laz2=(z^2-1)*laz+3*z*laz^2+2*laz^3;
                                a1=y-1*(z+laz)
                                a2=1+laz1
                                a3=laz-z*laz1;

                                B1=-sum(a1*a2)
                                B2=-sum(a1*a2*x)
                                B3=-sum(a1*a3)

                                A11=sum(a2^2-a1*laz2/1)
                                A12=sum(x*a2^2-a1*laz2*x/1)
                                A13=sum(a3*a2+a1*laz2*z/1)
                                A22=sum(x^2*a2^2-a1*laz2*x^2/1)
                                A23=sum(a3*a2*x+a1*laz2*x*z/1)
                                A33=sum(a3^2-a1*z^2*laz2/1)

                                A=matrix(c(A11,A12,A13,A12,A22,A23,A13,A23,A33),nrow=3)
                                cond=rcond(A);
                                flag=0;
                                if(abs(cond)<10^(-6))
                                {
                                        flag=1;
                                         break;
                                }
                                B=matrix(c(B1,B2,B3),nrow=3)
                                AiB=solve(A)%*%B
                                a=a-AiB[1]
                                b=b-AiB[2]
                        }
                        if(flag==0) break;
                }
                alp[i]=a;
```

```
            bet[i]=b;
        }
        ma[j]=mean(alp)
        msa[j]=mean((alp-1)^2)
        mb[j]=mean(bet)
        msb[j]=mean((bet-1)^2)
        Lpe[j]=mean(lowp)
        j=j+1;
}
results=rbind(ma,msa, mb, msb, Lpe)
dimnames(results)=list(c("Mean(a)", "MSE(a)", "Mean(b)", "MSE(b)", "Trunc.Pt"), c(100, 200, 300, 400, 500,
                        800, 1000))

results
```

#Heckman 2-step

```
set.seed(987654)
low=0;
Ma=Mb=Msig=Lpe=Maa=Mbb=rep(0,7)
MSa=MSb=MSs=MSaa=MSbb=rep(0,7)
j=1;
for(n in c(100,200,300,400,500,800,1000))
{
	a=b=sig=lowp=aa=bb=rep(0,500)
	for(k in seq(500))
	{
		x=rnorm(n,0,1)
		ystar=1+x+rnorm(n)
		y=(ystar>=low)
		fit=glm(y~x, family=binomial(link="probit"))
		lowp[k]=sum(y==low)/n
		a[k]=coef(fit)[1]
		b[k]=coef(fit)[2]
		x=x[ystar>low]
		lamda=dnorm(a[k]+b[k]*x)/pnorm(a[k]+b[k]*x)
		y=ystar[ystar>low]
		lamda=lamda[ystar>low]
		fit2=lm(y~x+lamda)
		aa[k]=coef(fit2)[1]
		bb[k]=coef(fit2)[2]
		sig[k]=1
	}

	Ma[j]=mean(a)
	Mb[j]=mean(b)
	MSa[j]=mean((a-1)^2)
	MSb[j]=mean((b-1)^2)
	Maa[j]=mean(aa)
	Mbb[j]=mean(bb)
	MSaa[j]=mean((aa-1)^2)
	MSbb[j]=mean((bb-1)^2)
	Msig[j]=mean(sig)
	MSs[j]=mean((sig-1)^2)
	Lpe[j]=mean(lowp)
	j=j+1
}
results=rbind(Maa, MSaa, Mbb, MSbb, Msig,MSs,Lpe)
dimnames(results)=list(c("Mean(aa)", "MSE(aa)", "Mean(bb)", "MSE(bb)", "Mean(sig)","MSE(sig)", "Trunc.Pt"),
c(100, 200, 300, 400, 500, 800, 1000))

results

#Theoretical Truncation Rate
f=function(x){pnorm(-1-x)};
integrate(f, lower=-1, upper=1)$value/2
```

#EM Algorithm

```
set.seed(987654)
options(decimal=3)
total=500
low=0;
se=1;
ma=mb=ms=msa=msb=mss=Lpe=rep(0,7)
k=1;
for(n in c(100,200,300,400,500,800,1000))
	{
		alp=bet=sigm=Lowp=rep(0,total)
		 for(i in seq(total))
		{
			x=rnorm(n,0,1)
			ystar=1+x+rnorm(n,0,se)
			y=pmax(ystar,low);
			x0=x[y==low];
			xp=x[y>low];
			 y0=y[y==low];
			yp=y[y>low];
			a=0.95;
			b=0.95;
			sig=1;
			X=cbind(rep(1,n),c(xp,x0));
			a1=b1=10;
			s1=1;
			 repeat
			{
				z0=(a+b*x0)/sig;
				 p1=dnorm(z0);
				P1=pnorm(z0);
				y0new=a+b*x0-sig*p1/(1-P1);
				vy0=sig^2+a+b*x0*(sig*p1/(1-P1))-(sig*p1/(1-P1))^2;
				B=solve(t(X)%*%X)%*%t(X)%*%(c(yp,y0new))
				a=B[1];
				b=B[2];
				sig=1;
				if((abs(a1-a)<10^(-6))&(abs(b1-b)<10^(-6))&(abs(s1-sig)<10^(-6))){break;}
				a1=a;
				b1=b;
				s1=sig;
			}
			alp[i]=a;
			bet[i]=b;
			sigm[i]=sig;
			Lowp[i]=sum(y==low)/n
		 }
		ma[k]=mean(alp)
		msa[k]=mean((alp-1)^2)
		mb[k]=mean(bet)
		msb[k]=mean((bet-1)^2)
		ms[k]=mean(sigm)
		mss[k]=mean((sigm-1)^2)
		Lpe[k]=mean(Lowp)
		k=k+1;
	}
result=rbind(ma,msa,mb,msb,ms,mss, Lpe)
dimnames(result)=list(c("Mean(a)","MSE(a)","Mean(b)","MSE(b)","Mean(sigma)","MSE(sigma)",
			"Trunc.Pt"),c(100,200,300,400,500,800,1000))

result
```

Condition Set 5

```
#Tobit MLE
set.seed(987654)
library(VGAM)
low=0;
Ma=Mb=Msig=Lpe=rep(0,7)
MSa=MSb=MSsig=rep(0,7)
j=1;
for(n in c(100,200,300,400,500,800,1000))
{
        a=b=sig=lowp=rep(0,500)
        for(k in seq(500))
        {
                x=seq(-sqrt(3), sqrt(3), len=n)
                ystar=1+x+rnorm(n)
                y=pmax(ystar,low)
                fit=vglm(y~x, tobit(Lower=low))
                lowp[k]=sum(y==low)/n
                table(fit@extra$censoredL)
                a[k]=coef(fit,matrix=TRUE)[1,1]
                b[k]=coef(fit,matrix=TRUE)[2,1]
                sig[k]=1

        }

        Ma[j]=mean(a)
        Mb[j]=mean(b)
        Msig[j]=mean(sig)
        MSa[j]=mean((a-1)^2)
        MSb[j]=mean((b-1)^2)
        MSsig[j]=mean((sig-1)^2)
        Lpe[j]=mean(lowp)
        j=j+1
}
results=rbind(Ma,MSa, Mb, MSb, Msig, MSsig, Lpe)
dimnames(results)=list(c("Mean(a)", "MSE(a)", "Mean(b)", "MSE(b)", "Mean(sig)", "MSE(sig)", "Trunc.Pt"),
c(100, 200, 300, 400, 500, 800, 1000))

results
```

```
#Probit MLE
set.seed(987654)
low=0;
Ma=Mb=Msig=Lpe=rep(0,7)
MSa=MSb=MSs=rep(0,7)
j=1;
for(n in c(100,200,300,400,500,800,1000))
{
        a=b=sig=lowp=rep(0,500)
        for(k in seq(500))
        {
                x=seq(-sqrt(3), sqrt(3),len=n)
                ystar=1+x+rnorm(n)
                y=(ystar>=low)
                fit=glm(y~x, family=binomial(link="probit"))
                lowp[k]=sum(y==low)/n
                a[k]=coef(fit)[1]
                b[k]=coef(fit)[2]

        }

        Ma[j]=mean(a)
        Mb[j]=mean(b)
        MSa[j]=mean((a-1)^2)
        MSb[j]=mean((b-1)^2)
        Lpe[j]=mean(lowp)
        j=j+1;
}
results=rbind(Ma,MSa, Mb, MSb, Lpe)
dimnames(results)=list(c("Mean(a)", "MSE(a)", "Mean(b)", "MSE(b)", "Trunc.Pt"), c(100, 200, 300, 400, 500,
800, 1000))

results

#Theoretical Truncation Rate
f=function(x){pnorm(-1-x)};
integrate(f, lower=-1, upper=1)$value/2
```

LSE Positive

```
 set.seed(987654)
 total=500
 ma=msa=mb=msb=Lpe=rep(0,7)
 j=1;
 for( n in c(100, 200, 300, 400, 500, 800, 1000))
{
  alp=bet=rep(0,total)
  for(i in seq(total))
  {
  repeat{
                lowp=0;
               low=0;
               se=1;
               x=seq(-sqrt(3), sqrt(3), len=n)
               ystar=1+x+rnorm(n,0,se)
               y=ystar[ystar>=low]
               x=x[ystar>=low];
               lowp=length(ystar[ystar<low])/n
                length(y);
               length(x);

               a=0.95;
               b=0.95;

                for(k in seq(10))
                        {
                                z=(a+b*x)/1;
                                laz=dnorm(z)/pnorm(z);
                                laz1=-z*laz-laz^2;
                                laz2=(z^2-1)*laz+3*z*laz^2+2*laz^3;
                                a1=y-1*(z+laz)
                                a2=1+laz1
                                a3=laz-z*laz1;

                                B1=-sum(a1*a2)
                                B2=-sum(a1*a2*x)
                                B3=-sum(a1*a3)

                                 A11=sum(a2^2-a1*laz2/1)
                                A12=sum(x*a2^2-a1*laz2*x/1)
                                A13=sum(a3*a2+a1*laz2*z/1)
                                A22=sum(x^2*a2^2-a1*laz2*x^2/1)
                                A23=sum(a3*a2*x+a1*laz2*x*z/1)
                                A33=sum(a3^2-a1*z^2*laz2/1)

                                A=matrix(c(A11,A12,A13,A12,A22,A23,A13,A23,A33),nrow=3)
                                cond=rcond(A);
                                flag=0;
                                if(abs(cond)<10^(-6))
                                        {
                                                flag=1;
                                                break;
                                         }
                                B=matrix(c(B1,B2,B3),nrow=3)
                                AiB=solve(A)%*%B

                                a=a-AiB[1]
                                b=b-AiB[2]

                        }
```

```
            if(flag==0) break;
        }

        alp[i]=a;
        bet[i]=b;

  }


  ma[j]=mean(alp)
  msa[j]=mean((alp-1)^2)
  mb[j]=mean(bet)
  msb[j]=mean((bet-1)^2)
  Lpe[j]=mean(lowp)
  j=j+1;

}

results=rbind(ma,msa, mb, msb, Lpe)
dimnames(results)=list(c("Mean(a)", "MSE(a)", "Mean(b)", "MSE(b)", "Trunc.Pt"), c(100, 200, 300, 400, 500,
                                800, 1000))
results
```

#Heckman 2-Step

```
set.seed(987654)
low=0;
Ma=Mb=Msig=Lpe=Maa=Mbb=rep(0,7)
MSa=MSb=MSs=MSaa=MSbb=rep(0,7)
j=1;
for(n in c(100,200,300,400,500,800,1000))
{
	a=b=sig=lowp=aa=bb=rep(0,500)
	for(k in seq(500))
	{
		x=seq(-sqrt(3), sqrt(3), len=n)
		ystar=1+x+rnorm(n)
		y=(ystar>=low)
		fit=glm(y~x, family=binomial(link="probit"))
		lowp[k]=sum(y==low)/n
		a[k]=coef(fit)[1]
		b[k]=coef(fit)[2]
		x=x[ystar>low]
		lamda=dnorm(a[k]+b[k]*x)/pnorm(a[k]+b[k]*x)
		y=ystar[ystar>low]
		lamda=lamda[ystar>low]
		fit2=lm(y~x+lamda)
		aa[k]=coef(fit2)[1]
		bb[k]=coef(fit2)[2]
		sig[k]=1
	}

	Ma[j]=mean(a)
	Mb[j]=mean(b)
	MSa[j]=mean((a-1)^2)
	MSb[j]=mean((b-1)^2)
	Maa[j]=mean(aa)
	Mbb[j]=mean(bb)
	MSaa[j]=mean((aa-1)^2)
	MSbb[j]=mean((bb-1)^2)
	Msig[j]=mean(sig)
	MSs[j]=mean((sig-1)^2)
	Lpe[j]=mean(lowp)
	j=j+1
}
results=rbind(Maa, MSaa, Mbb, MSbb, Msig,MSs,Lpe)
dimnames(results)=list(c( "Mean(aa)", "MSE(aa)", "Mean(bb)", "MSE(bb)", "Mean(sig)","MSE(sig)", "Trunc.Pt"),
			c(100, 200, 300, 400, 500, 800, 1000))

results
```

#EM Algorithm

```
set.seed(987654)
options(decimal=3)
total=500
low=0;
se=1;
ma=mb=ms=msa=msb=mss=Lpe=rep(0,7)
k=1;
for(n in c(100,200,300,400,500,800,1000))
 {
	alp=bet=sigm=Lowp=rep(0,total)
	for(i in seq(total))
	{
		x=seq(-sqrt(3), sqrt(3),len=n)
		ystar=1+x+rnorm(n,0,se)
		 y=pmax(ystar,low);
		x0=x[y==low];
		xp=x[y>low];
		y0=y[y==low];
		yp=y[y>low];
		a=0.95;
		b=0.95;
		sig=1;
		X=cbind(rep(1,n),c(xp,x0));
		a1=b1=10;
		s1=1;
		repeat
		{
			z0=(a+b*x0)/sig;
			p1=dnorm(z0);
			P1=pnorm(z0);
			y0new=a+b*x0-sig*p1/(1-P1);
			vy0=sig^2+a+b*x0*(sig*p1/(1-P1))-(sig*p1/(1-P1))^2;
			B=solve(t(X)%*%X)%*%t(X)%*%(c(yp,y0new))
			a=B[1];
			b=B[2];
			sig=1;
			if((abs(a1-a)<10^(-6))&(abs(b1-b)<10^(-6))&(abs(s1-sig)<10^(-6))){break;}
			a1=a;
			b1=b;
			s1=sig;
		}
		alp[i]=a;
		bet[i]=b;
		sigm[i]=sig;
		Lowp[i]=sum(y==low)/n
	}
	ma[k]=mean(alp)
	msa[k]=mean((alp-1)^2)
	mb[k]=mean(bet)
	msb[k]=mean((bet-1)^2)
	ms[k]=mean(sigm)
	mss[k]=mean((sigm-1)^2)
	Lpe[k]=mean(Lowp)
	k=k+1;
 }
result=rbind(ma,msa,mb,msb,ms,mss, Lpe)
dimnames(result)=list(c("Mean(a)","MSE(a)","Mean(b)","MSE(b)","Mean(sigma)","MSE(sigma)",
				"Trunc.Pt"),c(100,200,300,400,500,800,1000))

result
```

Condition Set 8

```
#Tobit MLE
set.seed(987654)
library(VGAM)
low=0;
Ma=Mb=Msig=Lpe=rep(0,7)
MSa=MSb=MSsig=rep(0,7)
j=1;
for(n in c(100,200,300,400,500,800,1000))
{
        a=b=sig=lowp=rep(0,500)
        for(k in seq(500))
        {
                x=rnorm(n,0,1)
                ystar=1+x+rnorm(n)
                y=pmax(ystar,low)
                fit=vglm(y~x, tobit(Lower=low))
                lowp[k]=sum(y==low)/n
                table(fit@extra$censoredL)
                a[k]=coef(fit,matrix=TRUE)[1,1]
                b[k]=coef(fit,matrix=TRUE)[2,1]
                sig[k]=exp(coef(fit,matrix=TRUE)[1,2])

        }

        Ma[j]=mean(a)
        Mb[j]=mean(b)
        Msig[j]=mean(sig)
        MSa[j]=mean((a-1)^2)
        MSb[j]=mean((b-1)^2)
        MSsig[j]=mean((sig-1)^2)
        Lpe[j]=mean(lowp)
        j=j+1
}
results=rbind(Ma,MSa, Mb, MSb, Msig, MSsig, Lpe)
dimnames(results)=list(c("Mean(a)", "MSE(a)", "Mean(b)", "MSE(b)", "Mean(sig)", "MSE(sig)", "Trunc.Pt"),
                c(100, 200, 300, 400, 500, 800, 1000))

results
```

 # LSE Positive

```
 set.seed(987654)
 total=500
 ma=msa=mb=msb=ms=mss=Lpe=rep(0,7)
 j=1;
 for( n in c(100, 200, 300, 400, 500, 800, 1000))
{
        alp=bet=sigm=rep(0,total)
        for(i in seq(total))
        {
              repeat{
                           lowp=0;
                           low=0;
                          se=1;
                          x=rnorm(n,0,1)
                          ystar=1+x+rnorm(n,0,se)
                          y=ystar[ystar>=low]
                          x=x[ystar>=low];
                          lowp=length(ystar[ystar<low])/n
                           length(y);
                           length(x);
                           a=0.95;
                          b=0.95;
                          sig=0.95;


                          for(k in seq(10))
                          {
                                      z=(a+b*x)/1;
                                      laz=dnorm(z)/pnorm(z);
                                      laz1=-z*laz-laz^2;
                                      laz2=(z^2-1)*laz+3*z*laz^2+2*laz^3;
                                      a1=y-1*(z+laz)
                                      a2=1+laz1
                                      a3=laz-z*laz1;

                                      B1=-sum(a1*a2)
                                      B2=-sum(a1*a2*x)
                                      B3=-sum(a1*a3)

                                      A11=sum(a2^2-a1*laz2/1)
                                      A12=sum(x*a2^2-a1*laz2*x/1)
                                      A13=sum(a3*a2+a1*laz2*z/1)
                                      A22=sum(x^2*a2^2-a1*laz2*x^2/1)
                                      A23=sum(a3*a2*x+a1*laz2*x*z/1)
                                      A33=sum(a3^2-a1*z^2*laz2/1)

                                      A=matrix(c(A11,A12,A13,A12,A22,A23,A13,A23,A33),nrow=3)
                                      cond=rcond(A);
                                      flag=0;
                                      if(abs(cond)<10^(-6))
                                      {
                                                  flag=1;
                                                  break;
                                      }
                                      B=matrix(c(B1,B2,B3),nrow=3)
                                      AiB=solve(A)%*%B

                                      a=a-AiB[1]
                                      b=b-AiB[2]
                                      sig=sig-AiB[3]
```

```
                }
                if(flag==0) break;
            }
            alp[i]=a;
            bet[i]=b;
            sigm[i]=sig;
        }
        ma[j]=mean(alp)
        msa[j]=mean((alp-1)^2)
         mb[j]=mean(bet)
        msb[j]=mean((bet-1)^2)
        ms[j]=mean(sigm)
         mss[j]=mean((sigm-1)^2)
        Lpe[j]=mean(lowp)
        j=j+1;
}

results=rbind(ma,msa, mb, msb, ms, mss, Lpe)
dimnames(results)=list(c("Mean(a)", "MSE(a)", "Mean(b)", "MSE(b)", "Mean(sig)", "MSE(sig)", "Trunc.Pt"),
                    c(100, 200, 300, 400, 500, 800, 1000))
results
```

#Heckman 2-Step

```
set.seed(987654)
low=0;
se=1;

Ma=Mb=Msig=Lpe=Maa=Mbb=rep(0,7)
MSa=MSb=MSs=MSaa=MSbb=rep(0,7)
j=1;
for(n in c(100,200,300,400,500,800,1000))
{
        a=b=sig=lowp=aa=bb=rep(0,500)
        for(k in seq(500))
        {
                 x=rnorm(n,0,1)
                ystar=1+x+rnorm(n,0,se)
                y=(ystar>=low)
                fit=glm(y~x, family=binomial(link="probit"))
                lowp[k]=sum(y==low)/n
                a[k]=coef(fit)[1]
                b[k]=coef(fit)[2]
                lamda=dnorm(a[k]+b[k]*x)/pnorm(a[k]+b[k]*x)
                y=ystar[ystar>low]
                x=x[ystar>low]
                lamda=lamda[ystar>low]
                fit2=lm(y~x+lamda)
                aa[k]=coef(fit2)[1]
                bb[k]=coef(fit2)[2]
                sig[k]=coef(fit2)[3]
        }

        Ma[j]=mean(a)
        Mb[j]=mean(b)
        MSa[j]=mean((a-1)^2)
        MSb[j]=mean((b-1)^2)
        Maa[j]=mean(aa)
        Mbb[j]=mean(bb)
        MSaa[j]=mean((aa-1)^2)
        MSbb[j]=mean((bb-1)^2)
        Msig[j]=mean(sig)
        MSs[j]=mean((sig-se)^2)
        Lpe[j]=mean(lowp)
        j=j+1
 }
 results=rbind(Ma, MSa, Mb, MSb, Maa, MSaa, Mbb, MSbb, Msig,MSs,Lpe)
 dimnames(results)=list(c("Mean(a)", "MSE(a)", "Mean(b)", "MSE(b)", "Mean(aa)", "MSE(aa)", "Mean(bb)",
                        "MSE(bb)", "Mean(sig)","MSE(sig)", "Trunc.Pt"), c(100, 200, 300, 400, 500, 800,
                        1000))

 results
```

#EM Algorithm

```
 set.seed(987654)
 total=500
 low=0;
 se=1;
 ma=mb=ms=msa=msb=mss=Lpe=rep(0,7)
 k=1;
 for(n in c(100,200,300,400, 500,800,1000))
 {
        alp=bet=sigm=Lowp=rep(0,total)
         for(i in seq(total))
        {
                x=rnorm(n,0,1)
                ystar=1+x+rnorm(n,0,se)
                y=pmax(ystar,low);
                x0=x[y==low];
                xp=x[y>low];
                y0=y[y==low];
                yp=y[y>low];
                a=0.95;
                b=0.95;
                sig=0.95;
                X=cbind(rep(1,n),c(xp,x0));
                a1=b1=s1=10;
                repeat
                {
                         z0=(a+b*x0)/sig;
                        p1=dnorm(z0);
                        P1=pnorm(z0);
                        y0new=a+b*x0-sig*p1/(1-P1);
                        vy0=sig^2+a+b*x0*(sig*p1/(1-P1))-(sig*p1/(1-P1))^2;
                        B=solve(t(X)%*%X)%*%t(X)%*%(c(yp,y0new))
                        a=B[1];
                        b=B[2];
                        sig=sqrt((sum((yp-a-b*xp)^2)+sum((y0-a-b*x0)^2)+
                        sum(sig^2+(a+b*x0)*(sig*p1/(1-P1))-(sig*p1/(1-P1))^2))/n);
                        if((abs(a1-a)<10^(-6))&(abs(b1-b)<10^(-6))&(abs(s1-sig)<10^(-6))){break;}
                        a1=a;
                        b1=b;
                        s1=sig;
                }
                alp[i]=a;
                bet[i]=b;
                sigm[i]=sig;
                Lowp[i]=sum(y==low)/n
        }
        ma[k]=mean(alp)
        msa[k]=mean((alp-1)^2)
        mb[k]=mean(bet)
        msb[k]=mean((bet-1)^2)
        ms[k]=mean(sigm)
        mss[k]=mean((sigm-1)^2)
        Lpe[k]=mean(Lowp)
        k=k+1;
 }

result=rbind(ma,msa,mb,msb,ms,mss, Lpe)
dimnames(result)=list(c("Mean(a)","MSE(a)","Mean(b)","MSE(b)","Mean(sigma)","MSE(sigma)","Trunc.Pt"),
                                c(100,200,300,400, 500,800,1000))
result
```

Condition Set 11

```
#Tobit MLE
set.seed(987654)
library(VGAM)
low=0;
Ma=Mb=Msig=Lpe=rep(0,7)
MSa=MSb=MSsig=rep(0,7)
j=1;
for(n in c(100,200,300,400,500,800,1000))
{
	a=b=sig=lowp=rep(0,500)
	for(k in seq(500))
	{
		x=seq(-sqrt(3), sqrt(3), len=n)
		ystar=1+x+rnorm(n)
		y=pmax(ystar,low)
		fit=vglm(y~x, tobit(Lower=low))
		lowp[k]=sum(y==low)/n
		table(fit@extra$censoredL)
		a[k]=coef(fit,matrix=TRUE)[1,1]
		b[k]=coef(fit,matrix=TRUE)[2,1]
		sig[k]=exp(coef(fit,matrix=TRUE)[1,2])

	}

	Ma[j]=mean(a)
	Mb[j]=mean(b)
	Msig[j]=mean(sig)
	MSa[j]=mean((a-1)^2)
	MSb[j]=mean((b-1)^2)
	MSsig[j]=mean((sig-1)^2)
	Lpe[j]=mean(lowp)
	j=j+1
}
results=rbind(Ma,MSa, Mb, MSb, Msig, MSsig, Lpe)
dimnames(results)=list(c("Mean(a)", "MSE(a)", "Mean(b)", "MSE(b)", "Mean(sig)", "MSE(sig)", "Trunc.Pt"),
			c(100, 200, 300, 400, 500, 800, 1000))

results
```

 # LSE Positive

```
set.seed(987654)
total=500
ma=msa=mb=msb=ms=mss=Lpe=rep(0,7)
 j=1;
for( n in c(100, 200, 300, 400, 500, 800, 1000))
{
     alp=bet=sigm=rep(0,total)
     for(i in seq(total))
     {
              repeat{
                           lowp=0;
                           low=0;
                           se=1;
                           x=seq(-sqrt(3), sqrt(3), len=n)
                           ystar=1+x+rnorm(n,0,se)
                           y=ystar[ystar>=low]
                           x=x[ystar>=low];
                           lowp=length(ystar[ystar<low])/n
                           length(y);
                            length(x);
                           a=0.95;
                           b=0.95;
                           sig=0.95;

                           for(k in seq(10))
                           {
                                        z=(a+b*x)/1;
                                        laz=dnorm(z)/pnorm(z);
                                        laz1=-z*laz-laz^2;
                                        laz2=(z^2-1)*laz+3*z*laz^2+2*laz^3;
                                        a1=y-1*(z+laz)
                                        a2=1+laz1
                                        a3=laz-z*laz1;

                                         B1=-sum(a1*a2)
                                        B2=-sum(a1*a2*x)
                                        B3=-sum(a1*a3)

                                        A11=sum(a2^2-a1*laz2/1)
                                        A12=sum(x*a2^2-a1*laz2*x/1)
                                        A13=sum(a3*a2+a1*laz2*z/1)
                                        A22=sum(x^2*a2^2-a1*laz2*x^2/1)
                                        A23=sum(a3*a2*x+a1*laz2*x*z/1)
                                        A33=sum(a3^2-a1*z^2*laz2/1)

                                         A=matrix(c(A11,A12,A13,A12,A22,A23,A13,A23,A33),nrow=3)
                                         cond=rcond(A);
                                        flag=0;
                                        if(abs(cond)<10^(-6))
                                        {
                                                     flag=1;
                                                     break;
                                        }
                                        B=matrix(c(B1,B2,B3),nrow=3)
                                        AiB=solve(A)%*%B

                                        a=a-AiB[1]
                                        b=b-AiB[2]
                                        sig=sig-AiB[3]
                           }
```

```
                if(flag==0) break;
            }
            alp[i]=a;
            bet[i]=b;
            sigm[i]=sig;
        }
        ma[j]=mean(alp)
        msa[j]=mean((alp-1)^2)
        mb[j]=mean(bet)
        msb[j]=mean((bet-1)^2)
        ms[j]=mean(sigm)
        mss[j]=mean((sigm-1)^2)
        Lpe[j]=mean(lowp)
        j=j+1;
}

results=rbind(ma,msa, mb, msb, ms, mss, Lpe)
dimnames(results)=list(c("Mean(a)", "MSE(a)", "Mean(b)", "MSE(b)", "Mean(sig)", "MSE(sig)", "Trunc.Pt"),
                    c(100, 200, 300, 400, 500, 800, 1000))
results
```

#Heckman 2-Step

```
set.seed(987654)
low=0;
se=1;
Ma=Mb=Msig=Lpe=Maa=Mbb=rep(0,7)
MSa=MSb=MSs=MSaa=MSbb=rep(0,7)
j=1;
for(n in c(100,200,300,400,500,800,1000))
 {
        a=b=sig=lowp=aa=bb=rep(0,500)
        for(k in seq(500))
        {
                x=seq(-sqrt(3), sqrt(3), len=n)
                ystar=1+x+rnorm(n,0,se)
                y=(ystar>=low)
                fit=glm(y~x, family=binomial(link="probit"))
                lowp[k]=sum(y==0)/n
                a[k]=coef(fit)[1]
                b[k]=coef(fit)[2]
                lamda=dnorm(a[k]+b[k]*x)/pnorm(a[k]+b[k]*x)
                y=ystar[ystar>low]
                x=x[ystar>low]
                lamda=lamda[ystar>low]
                fit2=lm(y~x+lamda)
                aa[k]=coef(fit2)[1]
                bb[k]=coef(fit2)[2]
                sig[k]=coef(fit2)[3]
        }
        Ma[j]=mean(a)
        Mb[j]=mean(b)
        MSa[j]=mean((a-1)^2)
        MSb[j]=mean((b-1)^2)
        Maa[j]=mean(aa)
        Mbb[j]=mean(bb)
        MSaa[j]=mean((aa-1)^2)
        MSbb[j]=mean((bb-1)^2)
        Msig[j]=mean(sig)
         MSs[j]=mean((sig-se)^2)
        Lpe[j]=mean(lowp)
        j=j+1
 }
results=rbind(Ma, MSa, Mb, MSb, Maa, MSaa, Mbb, MSbb, Msig,MSs,Lpe)
dimnames(results)=list(c("Mean(a)", "MSE(a)", "Mean(b)", "MSE(b)", "Mean(aa)", "MSE(aa)", "Mean(bb)",
                    "MSE(bb)", "Mean(sig)","MSE(sig)", "Trunc.Pt"), c(100, 200, 300, 400, 500, 800,
                    1000))

results
```

#EM Algorithm

```
set.seed(987654)
total=500
low=0;
 se=1;
ma=mb=ms=msa=msb=mss=Lpe=rep(0,7)
k=1;
for(n in c(100,200,300,400, 500,800,1000))
{
	alp=bet=sigm=Lowp=rep(0,total)
	for(i in seq(total))
	{
		x=seq(-sqrt(3), sqrt(3),len=n)
		ystar=1+x+rnorm(n,0,se)
		y=pmax(ystar,low);
		x0=x[y==low];
		xp=x[y>low];
		y0=y[y==low];
		yp=y[y>low];
		a=0.95;
		b=0.95;
		sig=0.95;
		X=cbind(rep(1,n),c(xp,x0));
		a1=b1=s1=10;
		repeat
		{
			z0=(a+b*x0)/sig;
			p1=dnorm(z0);
			P1=pnorm(z0);
			y0new=a+b*x0-sig*p1/(1-P1);
			vy0=sig^2+a+b*x0*(sig*p1/(1-P1))-(sig*p1/(1-P1))^2;
			B=solve(t(X)%*%X)%*%t(X)%*%(c(yp,y0new))
			a=B[1];
			b=B[2];
			sig=sqrt((sum((yp-a-b*xp)^2)+sum((y0-a-b*x0)^2)+
			sum(sig^2+(a+b*x0)*(sig*p1/(1-P1))-(sig*p1/(1-P1))^2))/n);
			if((abs(a1-a)<10^(-6))&(abs(b1-b)<10^(-6))&(abs(s1-sig)<10^(-6))){break;}
			a1=a;
			b1=b;
			s1=sig;
		}
		alp[i]=a;
		bet[i]=b;
		sigm[i]=sig;
		Lowp[i]=sum(y==low)/n
	}
	ma[k]=mean(alp)
	msa[k]=mean((alp-1)^2)
	mb[k]=mean(bet)
	msb[k]=mean((bet-1)^2)
	ms[k]=mean(sigm)
	mss[k]=mean((sigm-1)^2)
	Lpe[k]=mean(Lowp)
	k=k+1;
 }
 result=rbind(ma,msa,mb,msb,ms,mss, Lpe)

dimnames(result)=list(c("Mean(a)","MSE(a)","Mean(b)","MSE(b)","Mean(sigma)","MSE(sigma)","Trunc.Pt"),
                                c(100,200,300,400, 500,800,1000))
 result
```

Appendix C – Additional Figures

Figure C.1

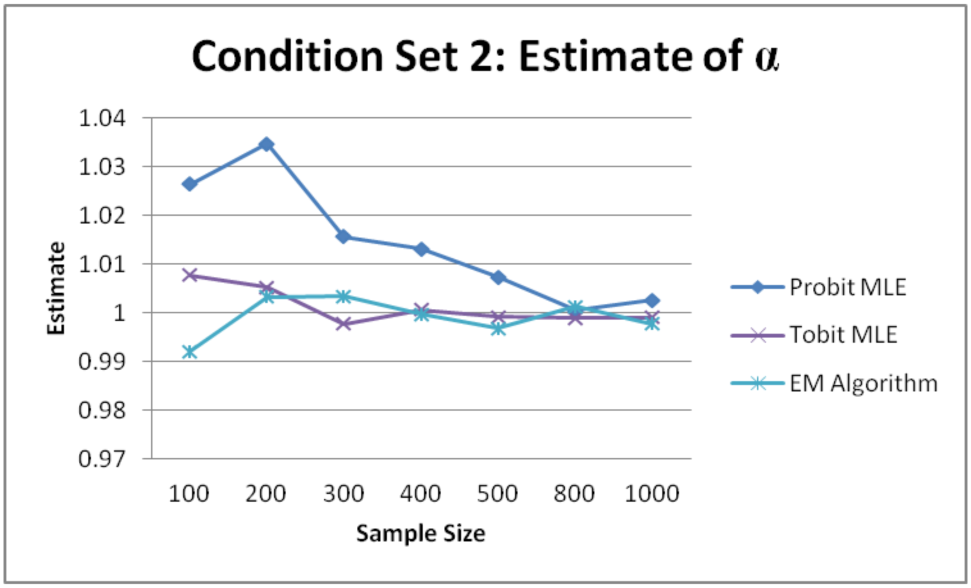


Figure C.2

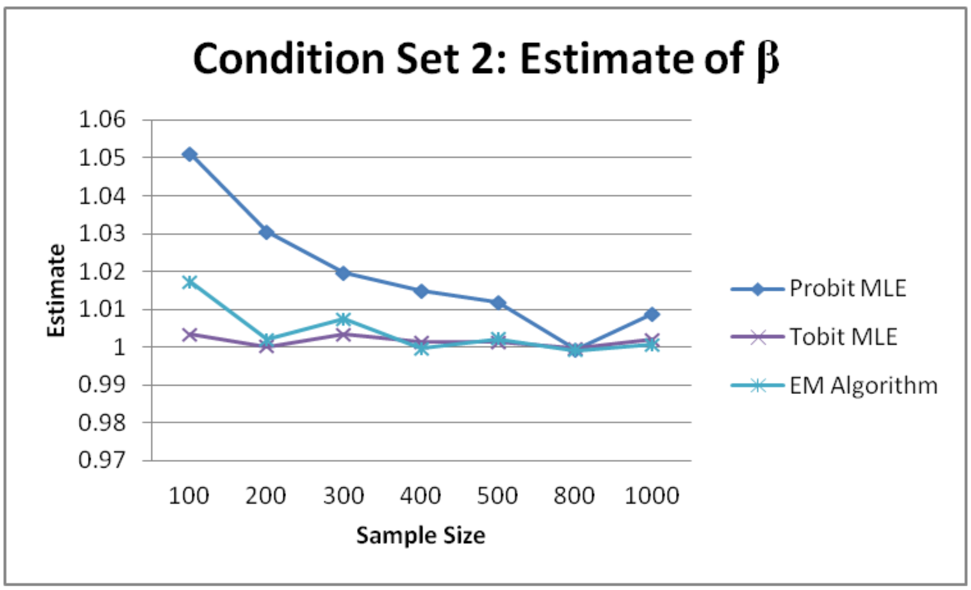


Figure C.3

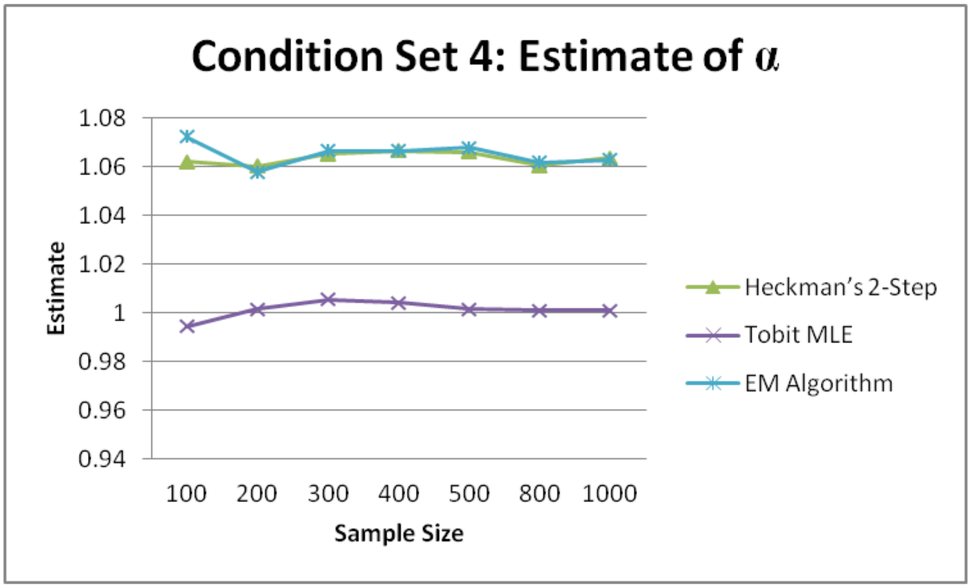


Figure C.4

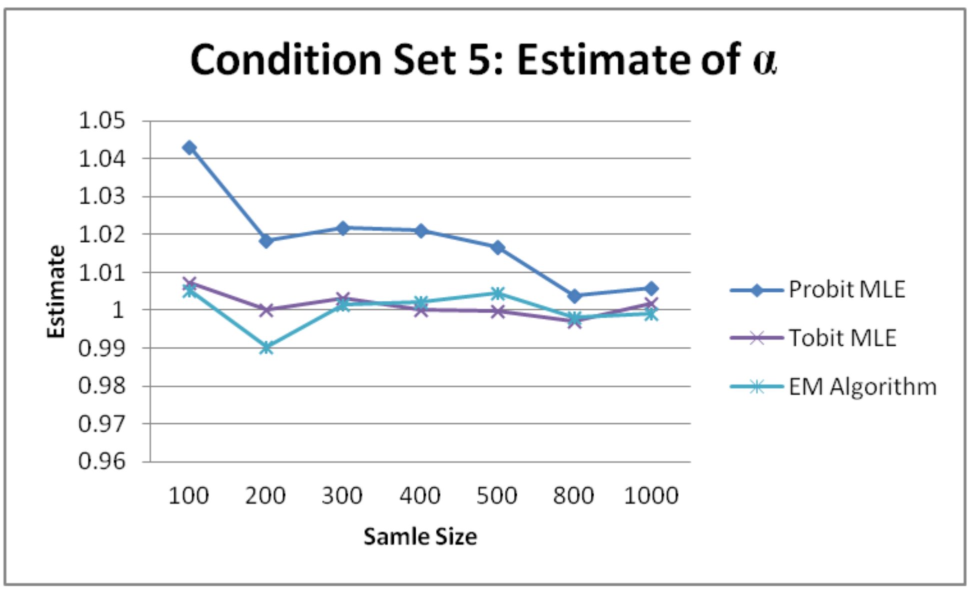


Figure C.5

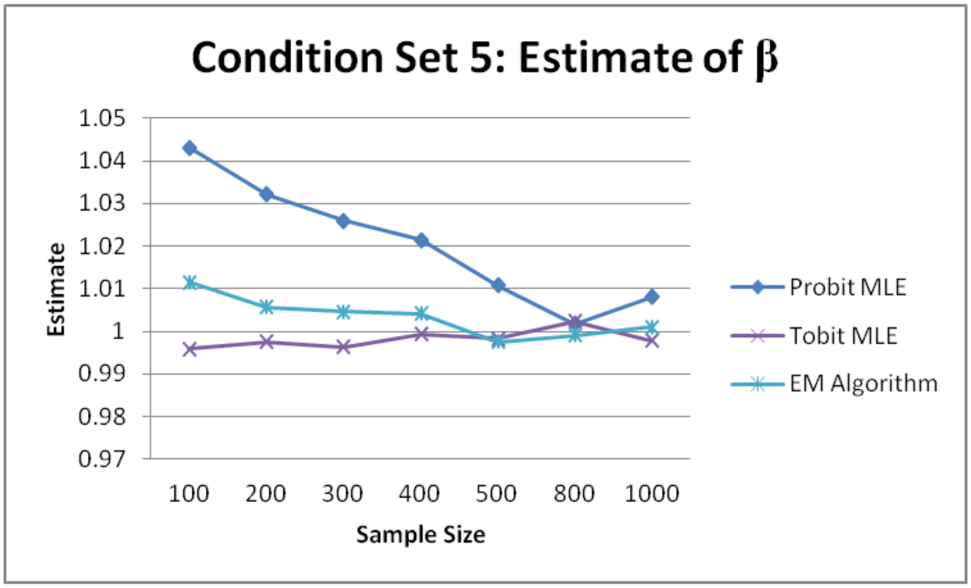


Figure C.6

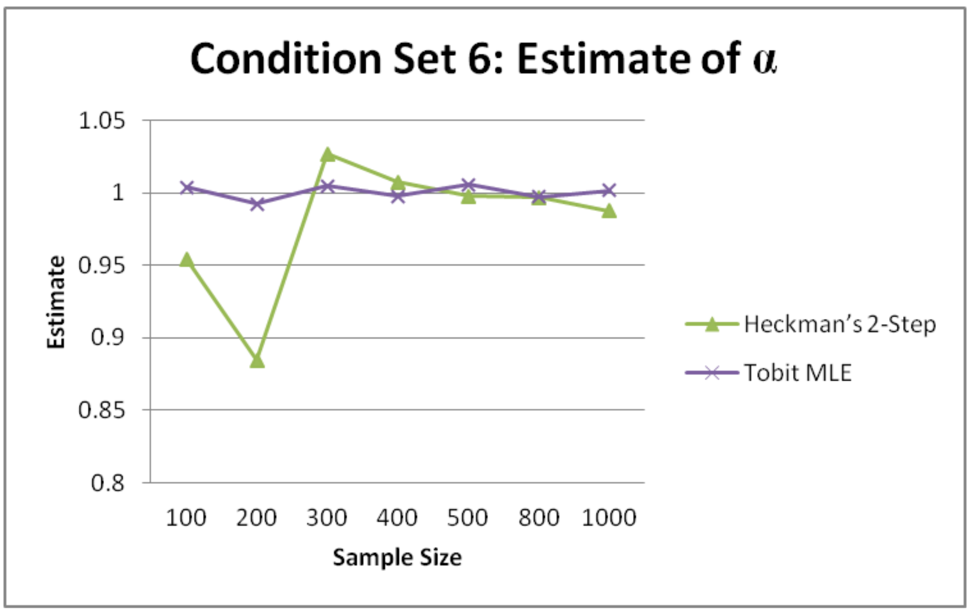


Figure C.7

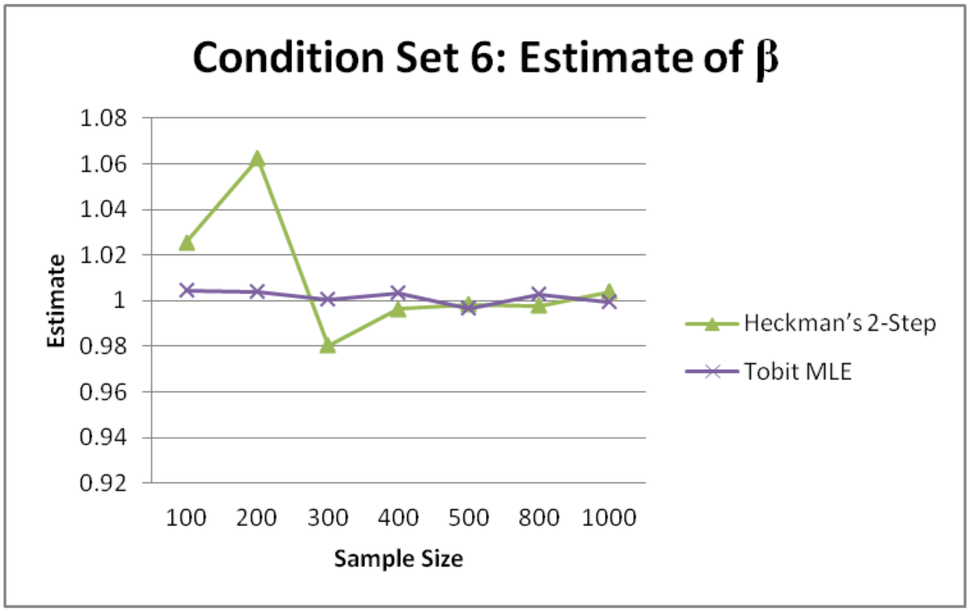


Figure C.8

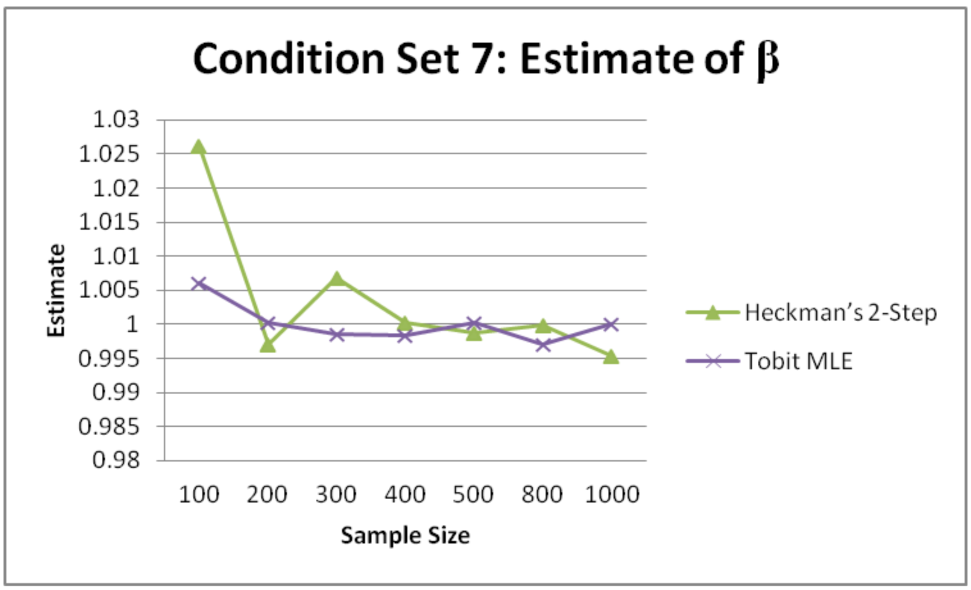


Figure C.9

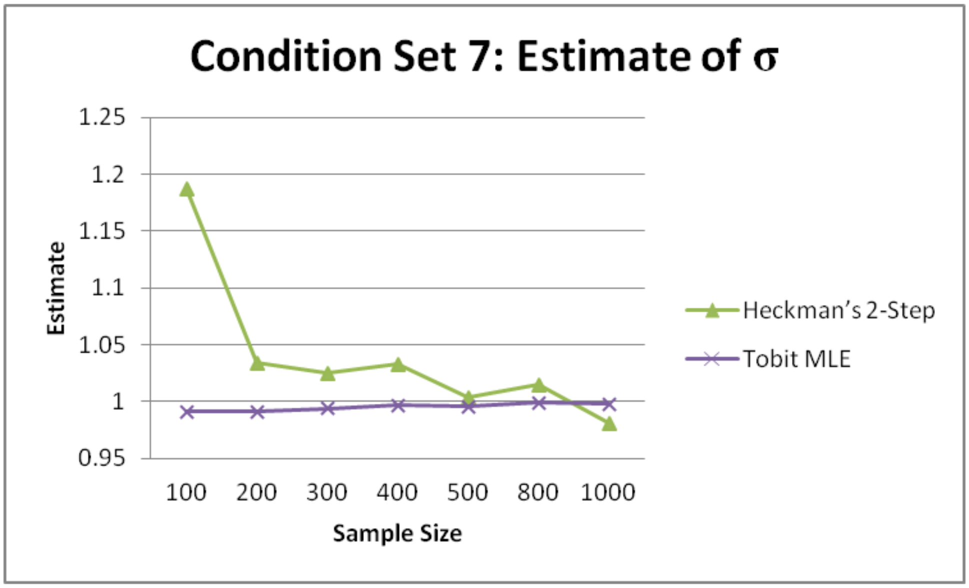


Figure C.10

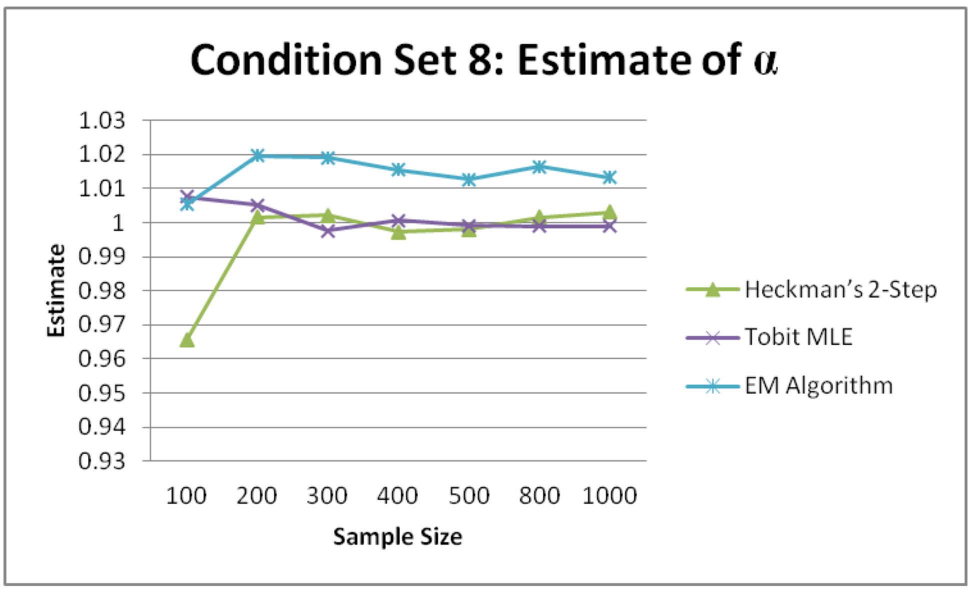


Figure C.11

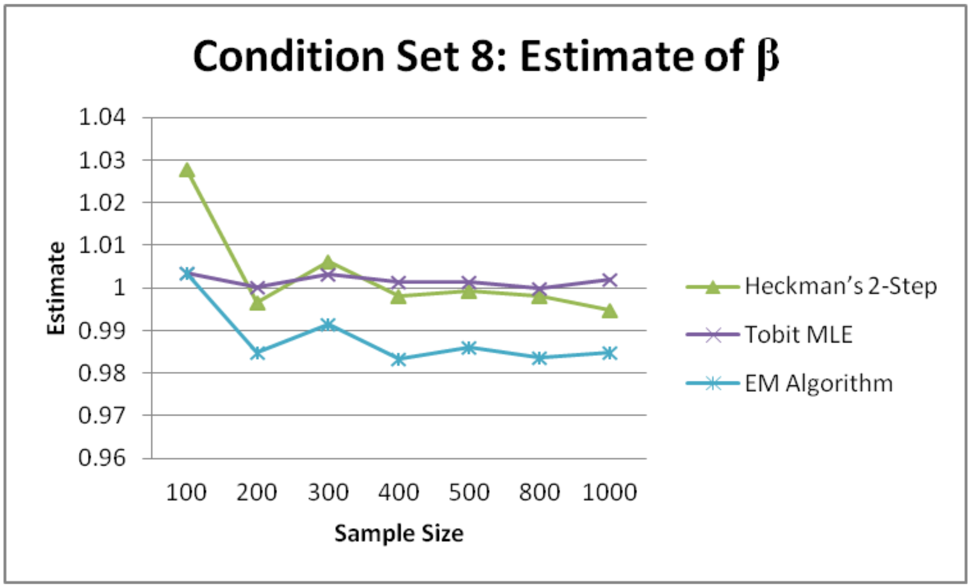


Figure C.12

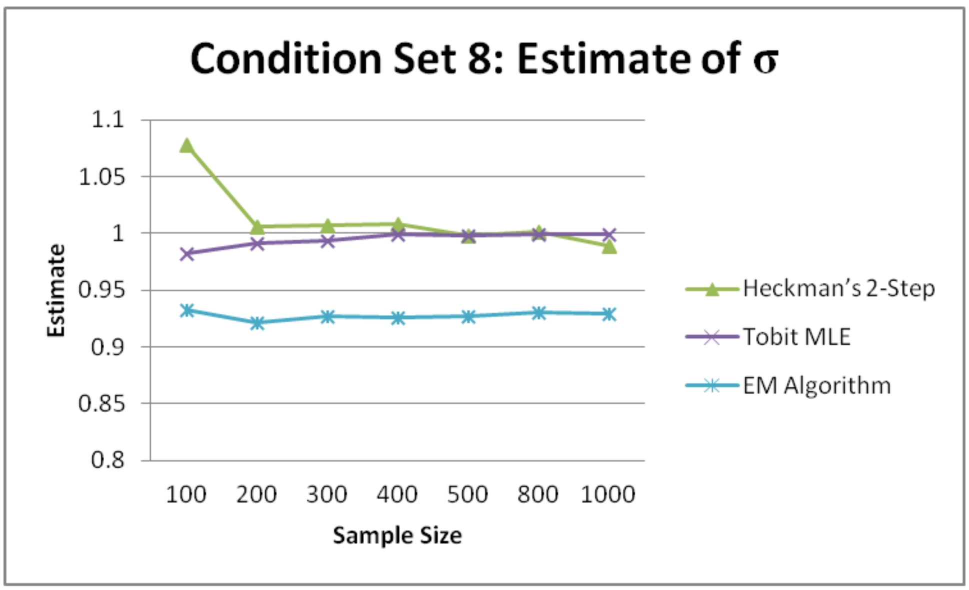


Figure C.13

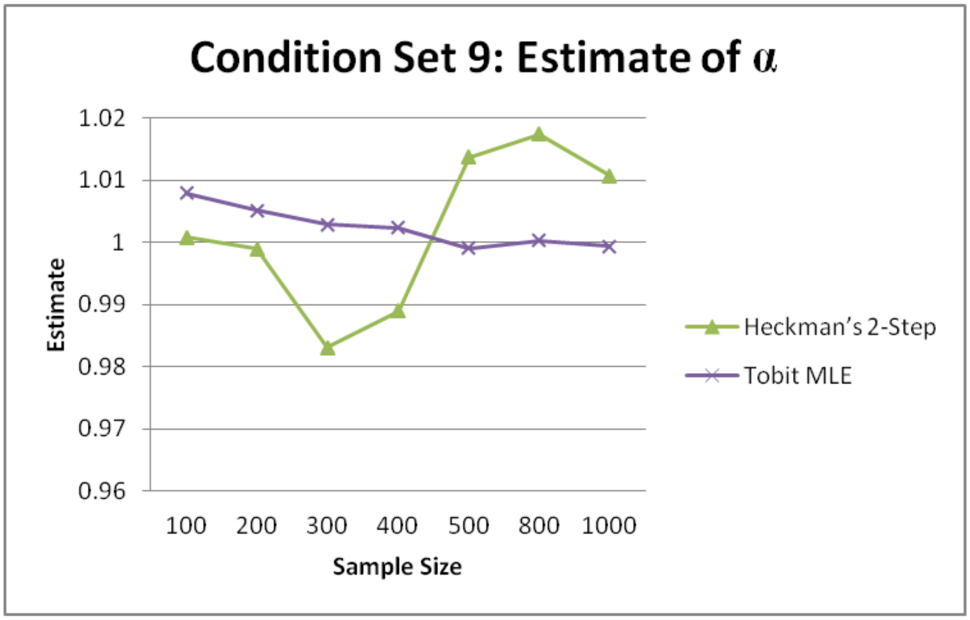


Figure C.14

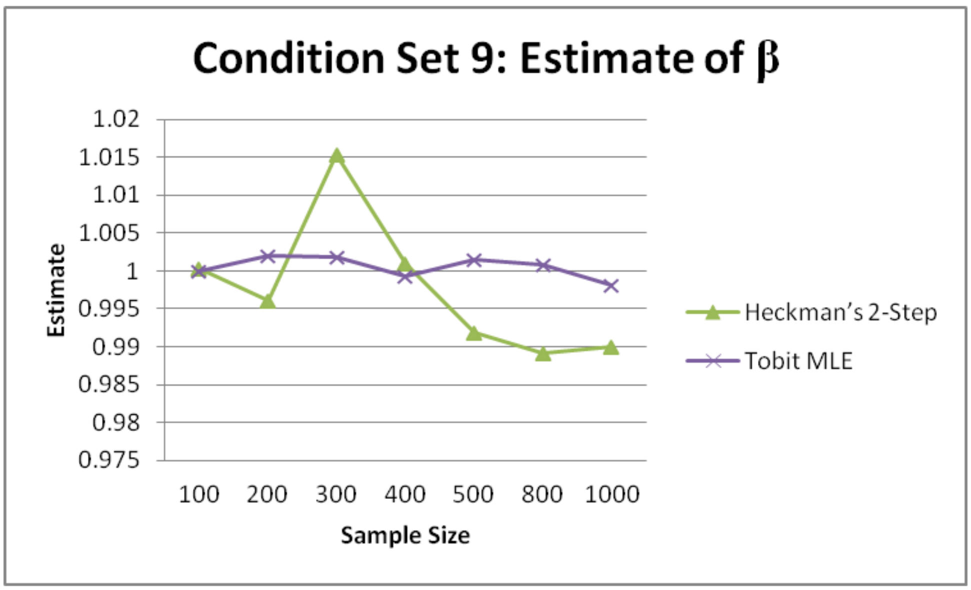


Figure C.15

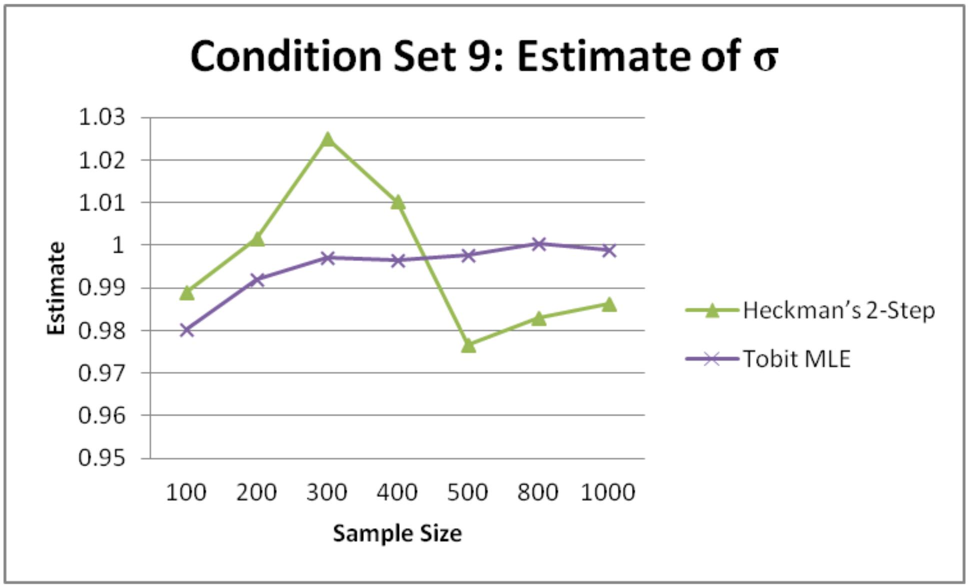


Figure C.16

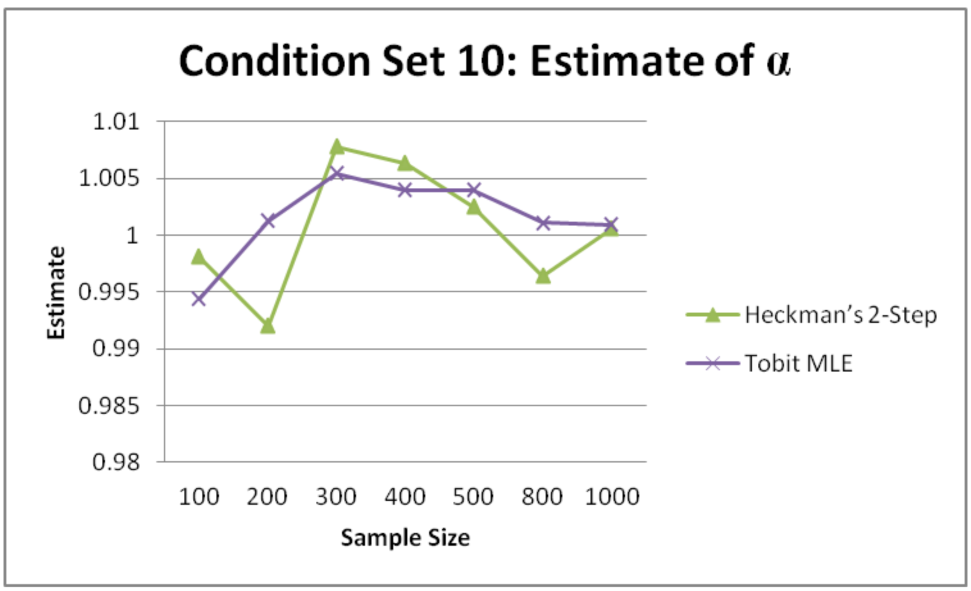


Figure C.17

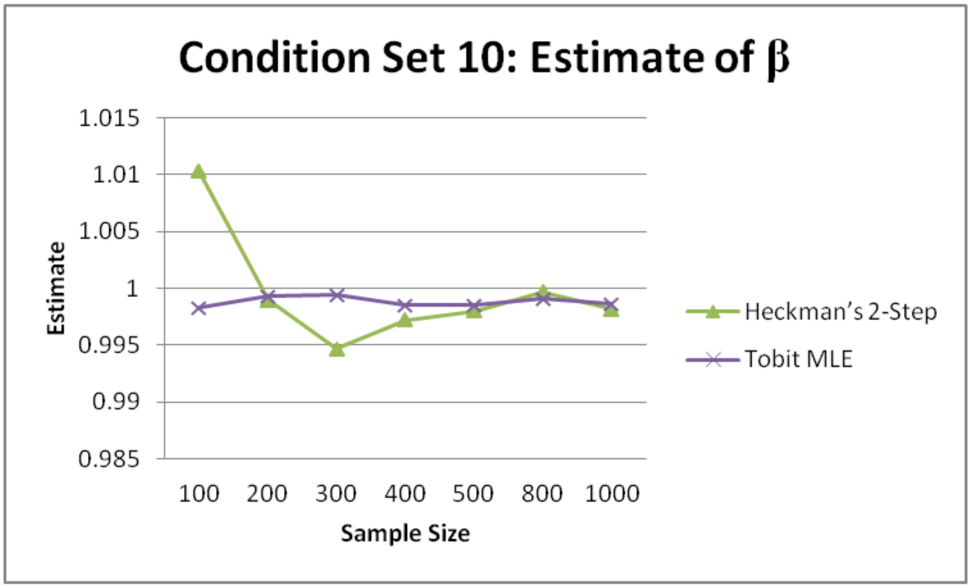


Figure C.18

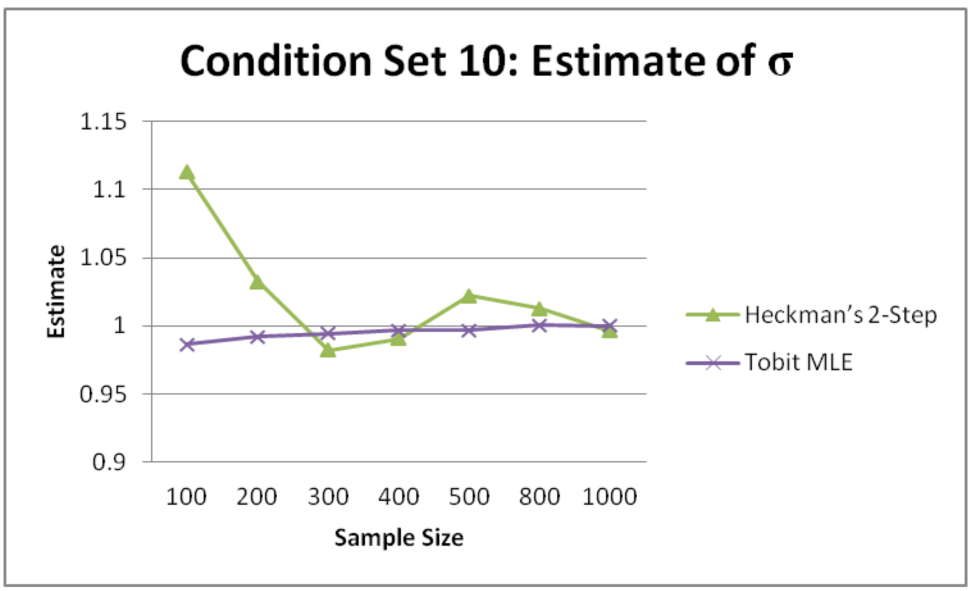


Figure C.19

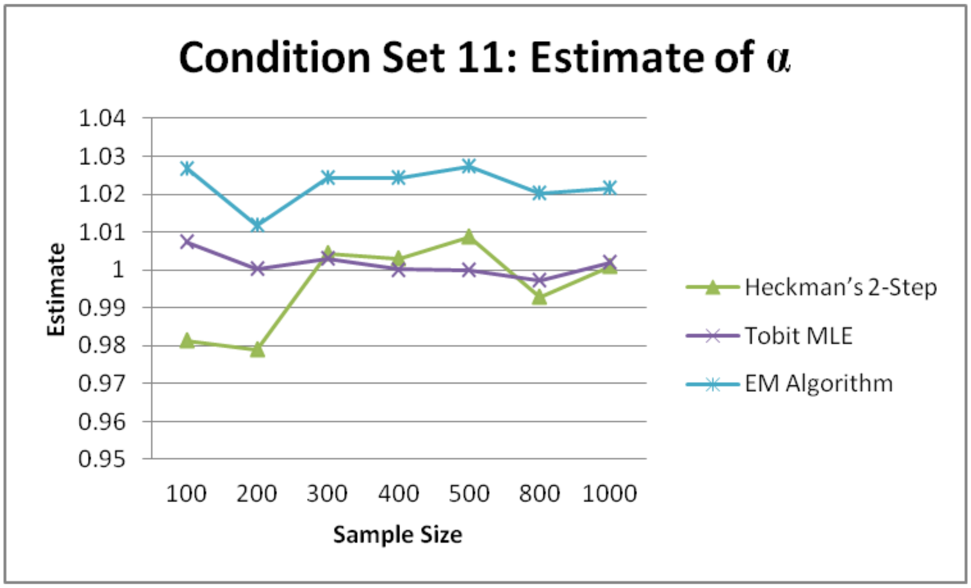


Figure C.20

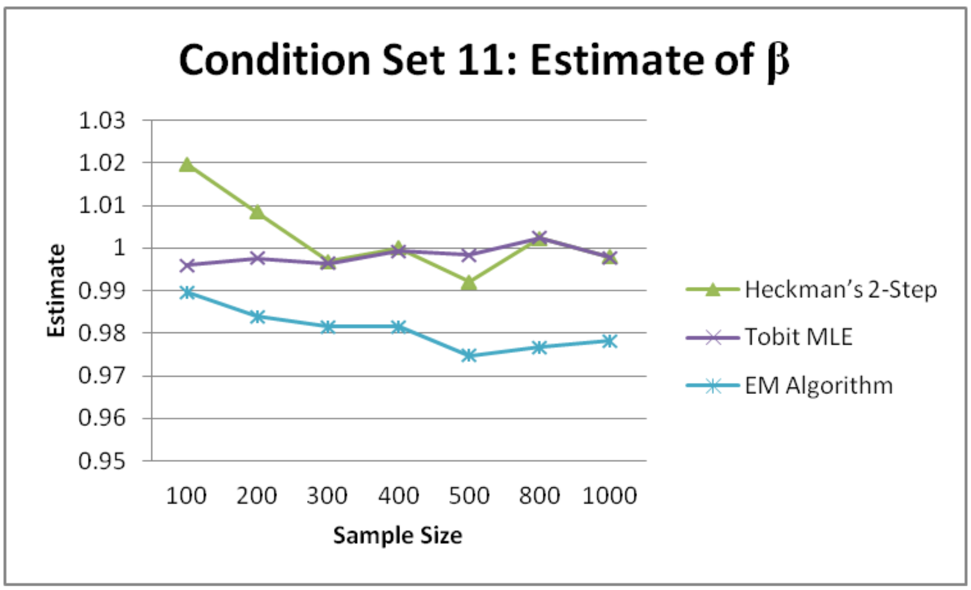


Figure C.21

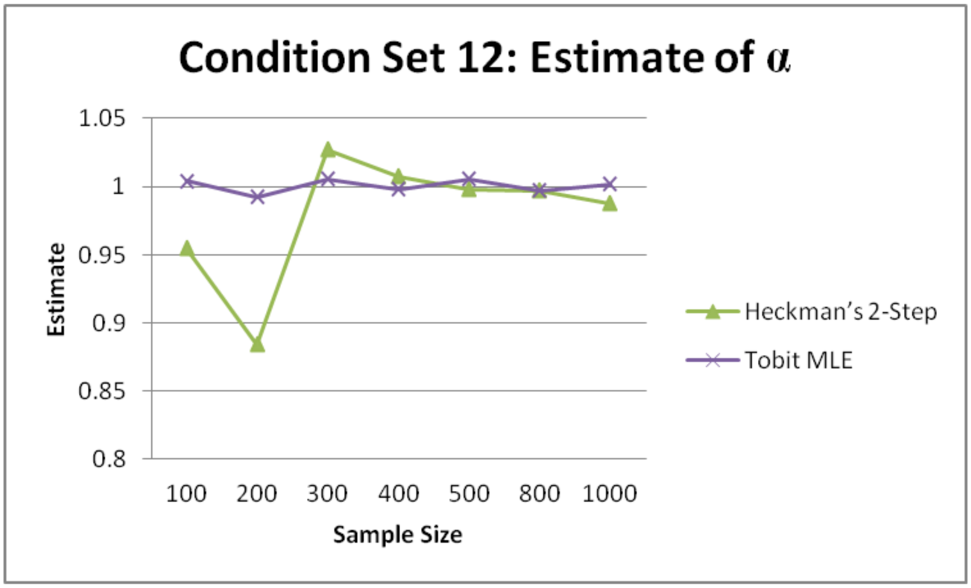


Figure C.22

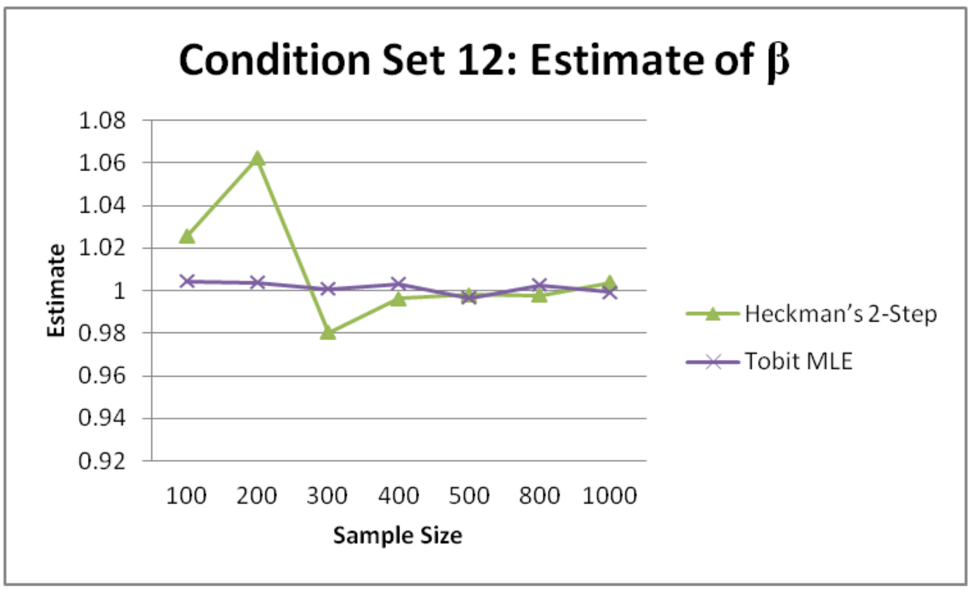
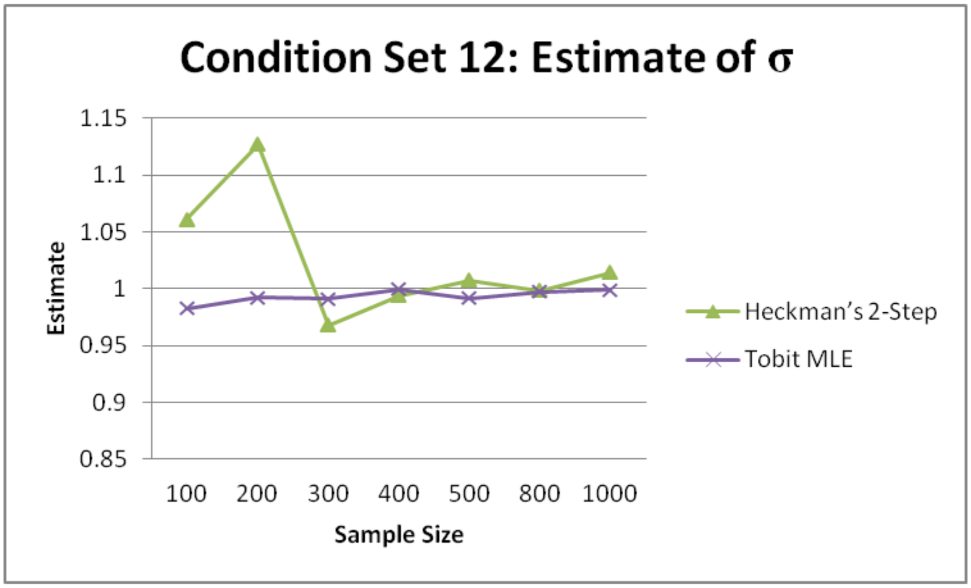


Figure C.23



Condition Set 12: Estimate of σ
1.15
1.1
1.05
1
0.95
0.9
0.85
Estimate
100
200
300
400
500
800
1000
Sample Size
Heckman's 2-Step
Tobit MLE