MI

93-24

Staff Paper

A GAUSS PROGRAM FOR ESTIMATING
DOUBLE-BOUNDED CENSORED LOGISTIC REGRESSION

Lih-Chyun Sun and John P. Hoehn

May, 1993 93-24

Department of Agricultural Economics
MICHIGAN STATE UNIVERSITY
East Lansing, Michigan

A GAUSS Program for Estimating Double-Bounded Censored Logistic Regression

Lih-Chyun Sun

and

John P. Hoehn[1]

[1] Sun is a graduate student and Hoehn is an associate professor in the Department of Agricultural Economics, Michigan State University, East Lansing, MI 48824.

A GAUSS Program for Estimating

Double-Bounded Censored Logistic Regression

1. Introduction

In a recent contingent valuation (CV) study by Hoehn and Loomis (1992), double-bounded censored logistic regression (DBCLR) was developed to estimate a willingness-to-pay (WTP) equation. From an econometric point of view, a DBCLR estimator combines two ML estimators suggested by two earlier studies. The first one is the censored logistic regression (Cameron, 1988), and the second one is the double-bounded dichotomous choice model (Hanemann, et al., 1991). Compared to a conventional dichotomous choice model (e.g. a logit, or a probit model), DBCLR estimates are more efficient and easier to interpret. In addition, DBCLR estimator is able to identify β and σ^2 (the variance of the conditional distribution of WTP) separately where a conventional dichotomous choice model can only identify β and σ^2 jointly as β/σ. However, unlike a conventional dichotomous choice model that can be estimated easily by most of the econometric packages, to our knowledge, none of the existing econometric packages has a convenient routine to implement the DBCLR estimator, this creates difficulties for some researchers in conducting econometric analysis. The purpose of this paper is to provide a computer program which can be used specifically for implementing the DBCLR estimator. The program, DBCLR.G, is written based on GAUSS (version 3.0) with the application module OPTMUM (version 2.01).

This paper contains 4 sections. In section 2, a DBCLR model and the corresponding log-likelihood function are specified. Section 3 describes the structure and user manual for DBCLR.G. A concluding summary is given as Section 4. In addition, source code of DBCLR.G is listed in

Appendix A. A sample program, DBCLR.E, is demonstrated in Appendix B along with the output. Appendix C gives the data set that are used in DBCLR.E.

2. A DBCLR Model and Its Log-Likelihood Function

Suppose that closed-ended questionnaires are used in a CV study to elicit respondents' WTP's for some environmental programs. For a respondent i (i = 1, 2, ..., N), WTP_i is specified as:

$$WTP_i^* = x_i'\beta + \mu_i, \ (\mu_i|x_i) \sim \text{i.i.d. } G(0, \sigma_\mu^2). \tag{1}$$

In equation (1), WTP_i^* is a latent dependent variable, x_i is a mx1 vector of independent variables, β is a mx1 vector of parameters to be estimated, and $G(0, \sigma_\mu^2)$ represents a logistic distribution with mean 0 and variance σ_μ^2. In addition, let $\Phi(.)$ be a logistic cumulative distribution function, and

$$\Phi(z_i) = \frac{\exp(z_i)}{1 + \exp(z_i)}.$$

According to equation (1), WTP_i^* is unobservable (only YES or NO response is observed). Given a referendum price, t_i, the individual i will answer NO (YES) if $WTP_i^* < t_i$ ($WTP_i^* \geq t_i$). It is also possible that the WTP_i^* is double-bounded, bounded above by t_i^h and below by t_i^l (i.e. $t_i^l \leq WTP_i^* < t_i^h$). Thus, the log-likelihood function of the WTP equation is[2]

[2] Equation (2) is exactly the same as equation (12) in Hoehn and Loomis's article. The third term in equation (2) here is a combination of the second and forth terms from Hoehn and Loomis' equation (12).

$$
\begin{aligned}
L = \sum_{i \in NO} \ln \left[\Phi\left(\frac{t_i - x_i\beta}{k}\right) \right] + \\
\sum_{i \in YES} \ln \left[1 - \Phi\left(\frac{t_i - x_i\beta}{k}\right) \right] + \qquad (2) \\
\sum_{i \in D-B} \ln \left[\Phi\left(\frac{t_i^h - x_i\beta}{k}\right) - \Phi\left(\frac{t_i^l - x_i\beta}{k}\right) \right].
\end{aligned}
$$

In equation (2), NO (YES) indicates the group of respondents who answered NO (YES) to the referendum price, D-B represents the group of respondents whose WTP's are double-bounded, and k is a dispersion parameter of logistic distribution. Since a standard logistic distribution has variance $\pi^2/3$, k can be converted to the standard deviation of logistic distribution by $\sigma_\mu = k*(\pi/\sqrt{3})$. Consistent and efficient estimates of β and k are acquired by maximizing L. Since the numerical optimization algorithms were developed for minimizing an objective function, in practice, it is the negative log-likelihood function (-L) that will be minimized.

3. Structure and User Manual for DBCLR.G

DBCLR.G uses GAUSS and its application module OPTMUM to minimize the negative log-likelihood function. There are three sections contained in DBCLR.G, a brief description for each section is listed below.

1. User input section: in this section, a user is required to

   A. specify an output file name;

   B. input data from a text file;

   C. specify variables; and

   D. specify starting values.

2. Data sorting, optimization, and output section: in this section, based on the user input, DBCLR.G performs the following routines:

A. sorting data into three groups (Double-Bounded, answered Yes, and answered No);

B. specifying variables for each group;

C. minimizing the negative log-likelihood function;

D. calculating covariance matrix; and

E. writing output file.

3. Procedures section: this section contains two procedures that specify the negative log-likelihood function (-L), and analytical gradient ($\partial(-L)/\partial\theta'$) respectively, where $\theta = [\beta' \; k]$.

To use DBCLR.G, a user is required to fill out the user input section, and in order to do that, some basic knowledge about GAUSS programming language is needed. For users who are short of GAUSS programming knowledge, GAUSS System and Graphics Manual, Goldberger (1991), and Hill (1989) can provide valuable help.

To fill out the user input section, DBCLR.G needs the following input from a user:

A. Specify an output file name

An output file name has to be specified. To specify an output file name, the general format is

output file = [drive:][\path\]filename[.ext];

where everything inside braces is an option. For example,

output file = out;

and

output file = c:\dbclr\dbclr.out;

are both valid specification. If the user specified output file does not exist, DBCLR.G will create the file; if the user specified output file has already exist, DBCLR.G will sent the output file to replace the original file. For a detail description about specifying output file, a user can consult both the GAUSS System and Graphics manual, and the GAUSS Applications Manual (Chapters 12 and 13).

B. Input data from a text file

Data can be contained in DBCLR.G as part of the program, however, a preferred way is to have the data contained in a separate file. The data file has to be a text file contains only numbers, no labels and empty cells are allowed. A convenient way to arrange the data file is to have a text data file in free format with no labels for variables, and all empty cells (e.g. missing values) are filled by a number (e.g. -999). After the data have been loaded, it is suggested to form the data into a big matrix with rows represent observations and columns represent variables. For example, for a text data file in drive a named data.asc that contains a 250x12 matrix, the command

load d[250,12] = a:data.asc;

will read the 250x12 data matrix from a:data.asc, and this 250x12 matrix will be named d in the DBCLR.G. About how GAUSS read data from a text file, consult GAUSS System and Graphic manual for detail description. (Note: as long as all the variables described below can be specified in the required format, it is actually not very important how a user read data into DBCLR.G.)

C. Variables Specification

Suppose that a user has already read a data file into DBCLR.G, after removing invalid observations and appropriate data manipulation, the valid data set becomes a NxP matrix, where N is the number of valid observations and P is the number of variables. In DBCLR.G, a user has to specify the following variables in exactly the same format as described.

TI: A Nx1 vector contains only 1s and 0s. An element in TI has a value 1, if the corresponding observation is double-bounded; 0, if the corresponding observation is not double-bounded. (Note: other than 1 and 0, no other value is allowed in TI vector.)

Y: A Nx1 vector contains 1s and 0s. An element in Y has a value 1, if the corresponding observation answered YES to the referendum price; 0, if the corresponding observation answered NO to the referendum price. If the corresponding observation is double-bounded, the element can have any value.

TL: A Nx1 vector. Each element in TL contains the lower bound of the corresponding observation's WTP. If the corresponding observation's WTP is not double-bounded, the element can have any value.

TH: A Nx1 vector. Each element in TH contains the upper bound of the corresponding observation's WTP. If the corresponding observation's WTP is not double-bounded, the element can have any value.

T: A Nx1 vector. Each element in T contains the referendum price for the corresponding observation. If the corresponding observation's WTP is double-bounded, the element can have any value.

X: A NxM matrix of independent variables, where M is the number of independent variables. (Note: do not include constant term in X matrix.)

C: A value, 1 or 0. 1, if a constant term is needed as the first independent variable; 0, if a constant term is not needed.

The following example provides an explicit way to explain the variables specification. Suppose that a valid 10x8 data matrix named d is

1	0	1	0	0	80	12	3
2	0	1	0	0	55	23	2
3	0	1	0	0	70	14	5
4	1	1	20	70	50	22	2
5	1	0	35	65	65	14	6
6	1	0	15	30	30	56	3
7	1	1	25	55	25	23	4
8	0	0	0	0	65	47	1
9	0	0	0	0	45	33	3
10	0	0	0	0	70	25	4

Where each row represents one observation, and each column represents a variable. Suppose that column 1 is the observation numbers, columns 2 through 6 are TI, Y, TL, TH, and T respectively, and columns 7 and 8 are two independent variables, X1 and X2. Clearly, matrix d has the following structure:

ID	TI	Y	TL	TH	T	X1	X2
1	0	1	0	0	80	12	3
2	0	1	0	0	55	23	2
3	0	1	0	0	70	14	5
4	1	1	20	70	50	22	2
5	1	0	35	65	65	14	6
6	1	0	15	30	30	56	3
7	1	1	25	55	25	23	4
8	0	0	0	0	65	47	1
9	0	0	0	0	45	33	3
10	0	0	0	0	70	25	4

If a user specifies variables as

```
TI = d[.,2];   @- TI is the second column of d -@
Y  = d[.,3];   @- Y is the third column of d -@
TL = d[.,4];   @- TL is the forth column of d -@
TH = d[.,5];   @- TH is the fifth column of d -@
T  = d[.,6];   @- T is the sixth column of d -@
X  = d[.,7 8]; @- X is the seventh and eighth column of d -@
```

Then, in DBCLR.G data sorting routine, it first separates data into two groups, double-bounded (obs. 4 5 6 7) and non-double-bounded (obs. 1 2 3 8 9 10) , based on the variable TI; secondly, it separates those non-double-bounded observations (obs. 1 2 3 8 9 10) into another two groups, answered YES (obs. 1 2 3) and answered NO (obs. 8 9 10), based on the variable Y. After data sorting, the original data set becomes 3 groups (double-bounded, answered YES, and answered NO). In the double-bounded group (obs. 4 5 6 7), since Y and T are irrelevant, they will be dropped. In both the answered YES (obs. 1 2 3) and NO (obs 8 9 10) groups, TI, TL, and TH are irrelevant and will all be ignored.

D. Specify starting values

The starting values for the parameters to be estimated are specified by a column vector named **B0**. Elements in B0 should be arranged as [constant β_1 β_2 ... β_M k]' (if no constant term is specified, ignore the constant). For example, the GAUSS command

let B0 = 2 4 3.5 3 1;

is a valid specification for specifying a 5x1 column vector as starting values.

E. Run DBCLR.G

After a user has finished the user input section, save DBCLR.G as a text file. To run DBCLR.G, at DOS prompt, type

gauss [drive:][\path\]dbclr.g

F. Optimization algorithms

Optimization algorithms used in DBCLR.G are the default algorithms as set by GAUSS application module OPTMUM, to change the algorithms, consult GAUSS Applications Modules Manual, Chapters 12 and 13. Also, a file READ.P04 under GAUSS sub-directory contains some valuable information too.

G. Output

An output text file is contained under the name specified by the user. It should cause no problem for a user to read the output, however, some explanation is provided below.

a) One shortcoming of the output design is that it does not give labels to the parameter estimates. In the output file, under "Estimation of Parameters", starting from the first row, the estimates appear in the order of β_0 (constant, if specified to have one), β_1, β_2, ..., β_M.

b) For convenience, both the dispersion parameter of logistic distribution (k), and the standard deviation of logistic distribution σ_μ ($= k * \pi/\sqrt{3}$) are reported in the output file. However, a user should keep in mind that k is one of the parameters estimates from the optimization routine, and σ_μ is calculated by $\sigma_\mu = k * \pi/\sqrt{3}$.

c) The variance-covariance matrix is calculated as the numerical hessian evaluated at the estimated θ, where $\theta = [\beta' \; k]$. In the output file, under "Variance-Covariance Matrix", each block represents one row in the covariance matrix, and the order is the same as the estimated parameters. In addition, the last block is the covariance between k and β, and the last element is the variance of k.

4. Summary

Double-bounded censored logistic regression has many advantages over the conventional dichotomous choice model, and may become a useful analytical tool for researchers to analyze CV data. In this paper, a GAUSS program, DBCLR.G, programmed specifically for estimating DBCLR is made available to researchers. With help provided by the user manual and the example given in Appendix B, a user with minimum GAUSS programming knowledge should be able to use DBCLR.G and estimate a DBCLR model.

DBCLR.G is an uncompiled GAUSS program, and it is programmed in a straight forward way, no complicated structure is involved. Most of the routines in the source code have notes attached to them indicating the purposes of the routines, this should help a user to understand the program. Due to its simple structure, DBCLR.G can be modified easily by a user to satisfy his/her own purposes.

Before any commercial econometric package that offers a very friendly and easy way to implement DBCLR, DBCLR.G should fulfill a researcher's need in estimating a DBCLR.

All the information contained in this paper are stored in a floppy disk too. The disk consists the following files:

READ.ME: A text file for short instruction.

DBCLR.WP: This document, in WordPerfect 5.1 format.

DBCLR.G: A text file, the source code of DBCLR.G.

DBCLR.E: A text file, the sample program for DBCLR.G.

DBCLR.ASC: A text data file used by DBCLR.E.

DBCLR.OUT: An output text file from DBCLR.E.

References

Cameron, T. A. (1988), "A New Paradigm for Valuing Non-market Goods Using Referendum Data: Maximum Likelihood Estimation by Censored Logistic Regression," Journal of Environmental Economics and Management, 15(3): 355-379.

GAUSS System and Graphic Manual, Version 2.2, Maple Valley, WA: Aptech Systems, Inc.

GAUSS Applications Manual, Version 2.01, Maple Valley, WA: Aptech System, Inc.

Goldberger, A. S. (1991), A Course in Econometrics, Cambridge, MA: Harvard University Press.

Hanemann, M., Loomis, J., and B. Kanninen (1991), "Statistical Efficiency of Double-Bounded Dichotomous Choice Contingent Valuation," American Journal of Agricultural Economics, 73(4): 1255-1263.

Hill, R. C. (1989), Learning Econometrics Using GAUSS: A Computer Handbook to Accompany Judge/Hill/Griffiths/Lutkepohl/Lee Introduction to the Theory and Practice of Econometrics, 2nd ed., New York, NY: John Wiley & Sons.

Hoehn, J. P., and J. B. Loomis (1992), "Substitution Effects in the Valuation of Multiple Environmental Programs," East Lansing, MI: Michigan State University, Department of Agricultural Economics Staff Paper No. 92-17.

Appendix A: Source Code of DBCLR.G

```
/**************************************************************/
/*                     DOUBLE-BOUNDED                         */
/*              CENSORED LOGISTIC REGRESSION                  */
/*                                                            */
/*                     Lih-Chyun Sun                          */
/*                     John P. Hoehn                          */
/*                                                            */
/*          Department of Agricultural Economics              */
/*               Michigan State University                    */
/**************************************************************/

/**************************************************************/
/* To use this program:                                       */
/* 1.  Specify an output file name                            */
/* 2.  Load data, variables arranged in columns               */
/* 3.  Manage data matrix                                     */
/* 4.  Specify variables TI, TL, TH, Y, T, X, and C           */
/* 5.  Specify starting values, B0                            */
/**************************************************************/

@-- Load OPTMUM Routine into RAM, Reset OPTMUM Parameters --@

library optmum;
#include optmum.ext;
optset;


/**************************************************************
1.  User Input Section
**************************************************************/

@-- Specify an Output File Name --@

output file = ;


@-- Input Data from a Text File --@

load ;


@-- Manipulating Data, if Necessary --@
```

```
@-- Specify Indicator Variables --@

TI = ;

Y = ;


@-- Specify Referendum Prices --@

TL = ;

TH = ;

T = ;


@-- Specify Independent Variables --@

X = ;


@-- Specify Constant Term --@

C = ;


@-- Specify Starting Values --@

let B0 = ;


/****************************************************************
TI: a Nx1 column vector contains only 1s and 0s.
    (1: if the corresponding obs. is double-bounded;
     0: if the corresponding obs. is not double-bounded.)
    (N: number of observations.)
Y: a Nx1 column vector contains 1s and 0s.
   (1: if the corresponding obs. answered YES;
    0: if the corresponding obs. answered NO;
    any number: if the corresponding obs. is double-bounded.)
TL: a Nx1 column vector, each element contains the lower bound
    of the corresponding obs.; if the corresponding obs. is not
    double-bounded, it can be any number.
TH: a Nx1 column vector, each element contains the upper bound
    of the corresponding obs.; if the corresponding obs. is not
    double-bounded, it can be any number.
T: a Nx1 column vector, each element contains the referendum
   price for the corresponding obs.; if the corresponding obs.
```

```
   is double-bounded, it can be any number.
X: a NxM matrix contains independent variables other than
   constant term.  (M: number of independent variables.)
C: a value, 1 or 0.
   (1: if a constant term as the first independent variable
       is needed;
    0: if a constant term is not needed.)
B0: a column vector contains starting values for parameters
    estimates.  [constant b1 b2 ... bM K]'
(See Document for detail)
****************************************************************/


/****************************************************************
             !!! User Can Usually Stop Here !!!
****************************************************************/


/****************************************************************
2.  Data Sorting, Estimation, and Output Section
****************************************************************/

@-- Adding Constant Term --@

if c == 1;
        x=ones(rows(x),1)~x;
elseif c == 0;
        x=x;
endif;


@-- Data Sorting --@

m=cols(x);
a=ti~y~t~tl~th~x;
clear ti, y, t, tl, th, x;
nob=rows(a);

@-- Observations whose WTPs are Double-Bounded --@

v1=selif (a, a[.,1] .eq 1);
nv1=rows(v1);


@-- Observations whose WTPs are not Double-Bounded --@

v2=selif (a, a[.,1] .eq 0);
a=0;
```

```
@-- Observations that Answered YES --@

a1=selif (v2, v2[.,2] .eq 1);
nyes=rows(a1);


@-- Observations that Answered NO --@

a2=selif (v2, v2[.,2] .eq 0);
nno=rows(a2);

v2=0;


@-- Variables for YES Observations --@

t1=a1[.,3];
x1=a1[.,6:(m+5)];
a1=0;


@-- Variables for NO Observations --@

t2=a2[.,3];
x2=a2[.,6:(m+5)];
a2=0;


@-- Variables for Double-Bounded Observations --@

tl=v1[.,4];
th=v1[.,5];
x3=v1[.,6:(m+5)];
v1=0;

x=x1|x2;
t=t1|t2;

x1=0;
t1=0;


@--- Pointers to Analytical Gradient ---@

_opgdprc = &foc;
```

```
@-- Call OPTMUM to Minimize -(Log-Likelihood Function) --@

{b,f,g,retcode} = optmum(&ofn,b0);


@-- Calculates Covariance Matrix from Numerical Hessian --@

h=_opfhess;


@-- Calculate Standard Errors and Z-Values --@

se=sqrt(diag(h));
tr=b./se;

bf=b~se~tr;
beta=bf[1:m,.];
k=bf[m+1,.];
ksd=k[1,1]*pi/(3^0.5);


@-- Output Section --@

output reset;
print "          ===========================================";
print "          ==           DOUBLE-BOUNDED              ==";
print "          ==     CENSORED LOGISTIC REGRESSION      ==";
print "          ===========================================";
print " ";

format /rd 6,0;
print "  Number of Observations: " nob;
print " ";
print "  Number of Observations With Double Bounds: " nv1;
print " ";
print "  Number of Observations Who Answered YES:   " nyes;
print " ";
print "  Number of Observations Who Answered NO:    " nno;
print " ";

format 14,4;
print "  Max. Log-Likelihood = " -f;
print " ";

format 15,5;
print "  Estimation of Parameters: ";
print "      Beta            Std.Err.        Z-Value";;
beta;
```

```
print " ";

print "  Estimation of K: ";
print "        K                  Std.Err.        Z-Value";
k;
print " ";

print "  Standard Deviation of Logistic Distribution ";
print "             [ K x Pi / (3^0.5) ]             = " ksd;
print " ";

format 12,5;

print "  Variance-Covariance Matrix:";;

h;

output off;


/****************************************************************
3.  Procedures Section
****************************************************************/

@-- Procedure Specifies -(Log-Likelihood Function) --@

proc ofn(b);
  local bb, k;
  bb=b[1:m,.];
  k=b[m+1,.];
  retp(-sumc((t2-x2*bb)/k)
       +sumc(ln(1+exp((t-x*bb)/k)))
       -sumc( ln( exp((th-x3*bb)/k)-exp((tl-x3*bb)/k) ) )
       +sumc(ln(1+exp((th-x3*bb)/k)))
       +sumc(ln(1+exp((tl-x3*bb)/k))));
endp;
```

```
@-- Procedure Specifies Analytical Gradient --@

proc foc(b);
  local bb,k,k2,fa,fb,fc,fd,fe,ka,kb,kc,kd,ke,fbeta,fk;
  bb=b[1:m,.];
  k=b[m+1,.];
  k2=k*k;
  fa = sumc( x2/k );
  fb = -sumc( ( exp( (t-x*bb)/k ) .* (x/k) )
          ./ ( 1 + exp( (t-x*bb)/k ) ) );
  fc = sumc( x3/k );
  fd = -sumc( ( exp( (th-x3*bb)/k ) .* (x3/k) )
          ./ ( 1 + exp( (th-x3*bb)/k ) ) );
  fe = -sumc( ( exp( (tl-x3*bb)/k ) .* (x3/k) )
          ./ ( 1 + exp( (tl-x3*bb)/k ) ) );
  ka = sumc( (t2-x2*bb)/k2 );
  kb = -sumc( exp( (t-x*bb)/k ) .* ( (t-x*bb)/k2 )
          ./ ( 1 + exp( (t-x*bb)/k ) ) );
  kc = sumc( ( exp( (th-x3*bb)/k ) .* ( (th-x3*bb)/k2 )
          - exp( (tl-x3*bb)/k ) .* ( (tl-x3*bb)/k2 ) )
          ./ ( exp( (th-x3*bb)/k ) - exp( (tl-x3*bb)/k ) ) );
  kd = -sumc( ( exp( (th-x3*bb)/k ) .* ( (th-x3*bb)/k2 ) )
          ./ ( 1 + exp( (th-x3*bb)/k ) ) );
  ke = -sumc( ( exp( (tl-x3*bb)/k ) .* ( (tl-x3*bb)/k2 ) )
          ./ ( 1 + exp( (tl-x3*bb)/k ) ) );
  fbeta = ( fa+fb+fc+fd+fe )';
  fk = ( ka+kb+kc+kd+ke );
  retp( fbeta~fk );
endp;

system;

/*************************************************************
                         End of Program
*************************************************************/
```

Appendix B: Sample Program, DBCLR.E

This appendix consists two sections, the first section contains the user input section of a sample program, DBCLR.E; the second section is the output from DBCLR.E.

1. DBCLR.E, User Input Section

```
library optmum;
#include optmum.ext;
optset;

output file = c:dbclr.out;

load d[237,12] = c:dbclr.asc;

TI = d[.,2];

Y = d[.,3];

TL = d[.,4];

TH = d[.,5];

T = d[.,6];

X = d[., 7:12];

C= 1;

let B0 = 0 0 0 0 0 0 0 1;
```

(Note: Everything beyond this point is exactly the same as in DBCLR.G, Appendix A.)

In this example, the user input specified the following information.

output file = c:dbclr.out;
The output file dbclr.out will be sent to drive c.

```
load d[237,12] = c:dbclr.asc;
```

This command causes GAUSS to load a 237x12 matrix from c:dbclr.asc (see Appendix C), and this 237x12 matrix is named d in this program.

```
TI = d[.,2];
```

TI is the second column of the d matrix

```
X = d[., 7:12];
```

The independent variables are the 7^{th}, 8^{th}, ..., 12^{th} columns of the d matrix. To be explicit, other than constant term, the 7^{th} column of d is the first independent variable (X1), the 8^{th} column is the second independent variable (X2), ..., and the 12^{th} column of d is the sixth independent variable (X6).

```
C= 1;
```

A constant term is needed as the first independent variable.

```
let B0 = 0 0 0 0 0 0 0 1;
```

A 8x1 column vector is specified as starting values. Since there are a constant term, six independent variables, and a k, eight parameters will be estimated. The starting values for constant and βs are all 0, the starting value for k is 1.

To implement DBCLR.E, on a 386-33 IBM compatible PC with GAUSS version 3.0, it takes about half a minute to finish DBCLR.E.

2. DBCLR.OUT, Output File

```
==============================================
==             DOUBLE-BOUNDED               ==
==     CENSORED LOGISTIC REGRESSION         ==
==============================================

Number of Observations:     237

Number of Observations With Double Bounds:       69

Number of Observations Who Answered YES:         87

Number of Observations Who Answered NO:          81

Max. Log-Likelihood =       -218.4293

Estimation of Parameters:
      Beta             Std.Err.          Z-Value
    -0.50747           0.37324          -1.35964
     0.43059           0.18667           2.30676
     0.52301           0.19734           2.65032
     0.94402           0.19543           4.83046
    -0.45593           0.21691          -2.10190
     0.29135           0.15455           1.88518
     0.65523           0.23485           2.78999

Estimation of K:
      K                Std.Err.          Z-Value
     0.57271           0.05284          10.83917

Standard Deviation of Logistic Distribution
            [ K x Pi / (3^0.5) ]              =           1.03878

Variance-Covariance Matrix:
   0.13931     -0.03065      0.00373     -0.02174     -0.00340
  -0.01247     -0.07650     -0.00301
  -0.03065      0.03484     -0.01882      0.01324     -0.00199
   0.00411      0.00596      0.00096
   0.00373     -0.01882      0.03894     -0.00870      0.00443
  -0.00054     -0.00049      0.00025
  -0.02174      0.01324     -0.00870      0.03819     -0.02128
   0.00397      0.00101      0.00134
  -0.00340     -0.00199      0.00443     -0.02128      0.04705
   0.00429      0.00109     -0.00190
```

-0.01247	0.00411	-0.00054	0.00397	0.00429
0.02389	-0.00124	0.00007		
-0.07650	0.00596	-0.00049	0.00101	0.00109
-0.00124	0.05516	0.00222		
-0.00301	0.00096	0.00025	0.00134	-0.00190
0.00007	0.00222	0.00279		

In the output file, under "Estimation of Parameters:", the 1st row is for constant term, the 2nd through 7th rows are for X1 through X6 respectively.

The first block of variance-covariance matrix contains [Var(constant) Cov(constant, β_1) Cov(constant, β_2) ... Cov(constant, β_6) Cov(constant, k)], and the last block is [Cov(constant, k) Cov(β_1, k) Cov(β_2, k) ... Cov(β_6, k) Var(k)].

Appendix C: Sample Data Set

This appendix contains a sample data set, DBCLR.ASC, used by DBCLR.E. DBCLR.ASC consists a 237x12 matrix (i.e. 237 observations and 12 variables) with no missing value and labels. In DBCLR.ASC, the 1st column is an index number for observations, 2nd through 6th columns are TI, Y, TL, TH, and T respectively, and 7th through 12th columns are independent variables X1, X2, ..., and X6. Following is the DBCLR.ASC:

1	1	1	0.35	1	0.35	0	0	1	0	0	1.6
2	0	0	0	0	0.35	0	0	1	0	0	1.8
3	1	1	0.3	0.6	0.3	0	0	0	0	1	1
4	1	1	0.35	1.1	0.35	0	0	1	0	0	1
5	1	1	0.35	0.5	0.35	0	0	1	0	0	1.6
6	0	0	0	0	0.5	1	0	0	0	0	1
7	0	0	0	0	1.8	0	0	1	1	0	1.6
8	0	1	0	0	2.4	1	1	0	0	1	1.6
9	0	0	0	0	1.8	1	1	1	0	0	0.8
10	0	0	0	0	2.4	1	0	1	1	1	0.8
11	0	0	0	0	1.25	1	1	0	0	0	0.8
12	0	0	0	0	1.3	0	0	1	1	0	0.8
13	0	0	0	0	1.8	1	0	1	1	0	0.8
14	0	0	0	0	0.65	1	0	0	0	0	0.8
15	0	1	0	0	0.35	0	0	1	0	0	1.4
16	0	1	0	0	0.7	1	1	1	0	0	1.4
17	0	1	0	0	0.3	0	0	0	0	1	1.2
18	0	1	0	0	0.9	1	0	1	0	1	1.2
19	0	1	0	0	0.9	1	1	0	0	0	1.2
20	1	1	0.9	1.1	0.9	1	1	0	0	0	1.4
21	1	0	0.65	1.25	1.25	1	0	0	0	1	1.8
22	0	1	0	0	0.8	1	0	0	0	0	1.4
23	1	1	1.2	1.8	1.2	0	0	0	0	1	1.4
24	1	1	1.5	2.25	1.5	0	0	0	0	1	1.2
25	1	1	0.6	1.8	0.6	0	0	1	0	0	1.6
26	0	1	0	0	2	1	1	1	0	1	1.4
27	0	1	0	0	0.9	1	1	1	0	0	1.2
28	0	0	0	0	1.4	1	0	0	0	1	1.8
29	0	0	0	0	2.75	1	1	1	0	0	1.8
30	0	1	0	0	4	1	1	1	0	0	1.4
31	0	1	0	0	2.5	0	0	1	0	1	1.6
32	0	1	0	0	0.55	1	1	0	0	0	1.6
33	1	1	0.8	1.25	0.8	0	0	1	0	0	1.6
34	1	1	1	2.25	1	0	0	1	1	0	1.6
35	0	0	0	0	1	0	0	1	1	0	0.6

36	0	0	0	0	1.5	1	0	1	1	1	0.6
37	0	0	0	0	0.8	1	0	0	0	1	0.6
38	0	0	0	0	1.2	1	0	1	1	0	0.6
39	0	0	0	0	0.8	1	1	0	0	0	0.6
40	0	0	0	0	0.4	1	0	0	0	0	0.6
41	0	1	0	0	0.5	1	0	0	0	0	1.2
42	0	1	0	0	1.75	1	1	1	0	0	1.2
43	1	0	1.3	4.5	4.5	1	0	1	1	1	1.4
44	1	0	1.3	2.25	2.25	0	0	1	0	1	1.4
45	1	1	1.3	2.25	1.3	0	0	1	0	0	1.4
46	0	0	0	0	0.6	0	0	1	0	0	1
47	0	0	0	0	1.3	0	0	1	1	0	1
48	1	0	1.4	4.5	2.1	1	1	0	0	0	1.8
49	0	1	0	0	2.25	0	0	1	1	0	1.4
50	0	1	0	0	0.45	0	0	0	0	1	1.4
51	1	0	0.3	0.6	0.6	1	0	0	0	1	1.4
52	0	1	0	0	0.7	0	0	0	0	1	0.8
53	0	1	0	0	0.65	1	0	0	0	0	0.8
54	1	1	0.65	1.25	0.65	1	0	0	0	0	1.6
55	0	1	0	0	2	1	0	1	1	0	1.4
56	0	0	0	0	3.25	1	1	0	0	1	0.6
57	0	1	0	0	0.5	1	0	0	0	0	1.2
58	0	0	0	0	2.5	1	0	1	0	0	1.6
59	0	1	0	0	0.55	1	1	0	0	0	1.6
60	0	1	0	0	0.65	1	0	0	0	0	1.2
61	0	1	0	0	0.8	0	0	1	0	0	1.4
62	0	1	0	0	2	1	1	1	0	1	1.4
63	1	1	1.3	3.5	1.3	0	0	1	0	0	1.6
64	0	1	0	0	4.5	1	1	1	0	1	1.6
65	1	0	1.2	4.5	4.5	1	1	1	1	1	1.4
66	1	1	0.8	2.1	0.8	1	0	0	0	0	1.4
67	1	1	1.2	3	1.2	0	0	0	0	1	1.4
68	0	1	0	0	1	1	0	0	0	1	1.2
69	0	1	0	0	2.4	1	1	1	0	0	1.4
70	0	1	0	0	1.8	0	0	1	1	0	1.4
71	1	1	0.8	1.8	0.8	1	0	0	0	0	1.4
72	1	0	0.8	3.75	3.75	1	0	1	1	1	1.4
73	1	0	0.4	0.7	0.7	1	0	0	0	1	1.4
74	1	1	1.8	3.25	1.8	0	0	1	0	0	1.6
75	0	0	0	0	3.25	1	1	0	0	1	1.6
76	0	0	0	0	4	1	1	1	1	0	1.6
77	1	1	0.5	1.4	0.5	1	0	0	0	0	1
78	1	1	1.25	2.25	1.25	1	1	0	0	0	1
79	0	1	0	0	0.7	1	0	1	1	0	1
80	0	1	0	0	0.9	1	1	1	0	0	1.4
81	1	0	0.45	1.25	1.25	1	0	1	1	1	1.4
82	0	1	0	0	0.35	0	0	1	0	0	1.4
83	0	1	0	0	0.7	1	1	1	0	0	1.4

84	0	1	0	0	1.25	1	1	1	0	1	1.4
85	0	1	0	0	1	1	1	1	1	0	1.4
86	0	1	0	0	1.75	1	1	1	0	0	1.4
87	0	1	0	0	2.25	1	1	1	0	1	1.4
88	1	1	1.3	2.25	1.3	0	0	1	0	0	1.4
89	0	1	0	0	0.9	1	0	1	0	1	1.4
90	1	0	1.8	9	5	0	0	1	1	0	1.8
91	0	1	0	0	1.4	1	1	0	0	0	1.6
92	1	0	0.9	1.25	1.25	0	0	1	0	1	1.4
93	1	1	0.9	1.25	0.9	0	0	0	0	1	1.4
94	1	1	0.6	1	0.6	0	0	1	1	0	1.6
95	1	1	0.3	0.8	0.3	0	0	0	0	1	1.6
96	1	1	0.9	1.1	0.9	1	1	0	0	0	1.4
97	1	0	0.3	1.1	1.1	1	1	0	0	1	1.4
98	1	0	0.45	0.7	0.7	1	0	0	0	1	1.4
99	0	1	0	0	1.6	1	1	1	0	1	1.4
100	0	1	0	0	0.5	1	0	0	0	0	1.4
101	1	1	0.65	1.1	0.65	1	1	0	0	0	1.4
102	1	1	0.45	0.7	0.45	0	0	0	0	1	1.2
103	0	0	0	0	0.45	0	0	1	0	0	1.8
104	0	0	0	0	0.6	0	0	0	0	1	1
105	0	0	0	0	1.75	1	1	1	1	1	1
106	0	1	0	0	1.75	1	1	1	0	1	1.2
107	0	0	0	0	0.7	0	0	0	0	1	1.8
108	0	0	0	0	2.25	1	1	1	1	1	1.8
109	1	1	2.1	4.5	2.1	1	1	0	0	0	1.2
110	0	1	0	0	2.4	1	1	1	0	0	1.2
111	0	1	0	0	2.4	1	0	1	1	0	1.2
112	0	1	0	0	2	1	1	1	0	1	1.2
113	0	0	0	0	1	0	0	1	1	0	1.2
114	1	0	0.65	1	1	1	0	0	0	1	1.8
115	0	0	0	0	1.8	1	1	0	0	1	1.8
116	0	0	0	0	2.25	1	1	1	1	0	1.8
117	0	1	0	0	1.5	1	0	0	0	1	1.6
118	1	0	1	3.75	2.25	0	0	1	1	0	1.4
119	1	0	0.3	0.65	0.65	1	1	0	0	1	1.8
120	1	0	0.3	0.65	0.45	1	1	0	0	0	1.8
121	0	1	0	0	1.25	1	1	0	0	0	1.6
122	1	0	0.6	2.75	1.8	0	0	1	1	0	1.8
123	0	0	0	0	2	0	0	0	0	1	1.8
124	0	0	0	0	0.6	0	0	1	1	0	1.8
125	0	0	0	0	0.9	1	1	0	0	1	1.8
126	0	0	0	0	0.7	1	0	0	0	1	1.8
127	0	0	0	0	0.45	0	0	1	0	0	1.8
128	0	0	0	0	1.25	1	1	1	0	1	1.8
129	0	0	0	0	0.55	1	1	0	0	0	1.8
130	0	0	0	0	1	1	1	1	1	0	1.8
131	0	0	0	0	1	0	0	1	1	0	1.8

132	0	1	0	0	0.65	1	0	0	0	0	1.8
133	0	1	0	0	0.8	1	0	0	0	1	1.2
134	0	1	0	0	1.2	1	0	1	1	0	1.2
135	1	0	1.2	3.75	3.75	1	1	1	1	1	1.4
136	1	0	1.3	2	2	0	0	1	0	1	1.4
137	0	1	0	0	0.25	0	0	0	0	1	1.2
138	0	1	0	0	1.5	1	1	1	0	1	1.2
139	0	1	0	0	1.2	1	1	0	0	1	1.2
140	1	1	0.6	1.8	0.6	0	0	1	0	0	1.6
141	0	0	0	0	1.4	1	0	0	0	1	1.8
142	0	0	0	0	0.9	0	0	0	0	1	1.8
143	0	0	0	0	4	1	1	1	1	0	1.8
144	0	0	0	0	0.65	1	0	0	0	0	1.8
145	0	0	0	0	2.25	0	0	1	1	0	1.8
146	0	1	0	0	2.5	1	1	0	0	0	1.8
147	0	1	0	0	1.8	0	0	1	0	1	1.2
148	0	1	0	0	2.4	1	1	1	0	0	1.4
149	1	0	1.3	2.75	2.25	0	0	1	1	0	1.4
150	1	1	1.3	2.25	1.3	0	0	1	0	0	1.4
151	1	1	1.1	2	1.1	1	0	0	0	0	1.4
152	0	1	0	0	0.4	1	0	0	0	0	1.8
153	1	0	0.45	0.8	0.8	1	0	1	0	0	1.4
154	0	1	0	0	1	1	1	1	0	0	1.4
155	0	1	0	0	0.75	1	1	0	0	0	1.4
156	0	1	0	0	0.7	1	1	1	0	0	1.4
157	0	0	0	0	1.4	1	1	1	1	0	1.8
158	1	1	0.5	1.25	0.5	1	0	0	0	0	1.6
159	1	0	1	3.75	2.25	0	0	1	1	0	1.4
160	1	1	0.8	2.1	0.8	1	0	0	0	0	1.4
161	0	0	0	0	1.1	1	0	0	0	0	1
162	0	0	0	0	2.5	1	1	0	0	0	1
163	0	0	0	0	3.25	1	1	1	0	0	0.6
164	0	0	0	0	2.5	1	0	0	0	1	0.6
165	0	0	0	0	2.25	0	0	1	1	0	0.6
166	0	0	0	0	1.4	1	0	0	0	0	0.6
167	0	0	0	0	0.6	0	0	1	1	0	0.8
168	0	0	0	0	0.8	1	1	1	0	0	0.8
169	0	0	0	0	0.3	1	0	0	0	0	0.8
170	0	0	0	0	0.55	1	1	0	0	0	0.8
171	0	0	0	0	0.3	0	0	0	0	1	0.8
172	0	0	0	0	0.3	1	0	0	0	0	0.8
173	0	1	0	0	0.4	1	0	0	0	0	1.8
174	0	0	0	0	5	0	0	1	1	0	1.6
175	0	1	0	0	6.5	1	1	1	0	1	1.6
176	1	0	0.8	3.75	2.1	1	1	0	0	0	1.8
177	0	1	0	0	2	1	1	1	0	1	1.4
178	0	1	0	0	0.3	1	0	0	0	0	1.2
179	0	1	0	0	1	1	0	1	0	0	1.2

180	0	1	0	0	0.55	1	1	0	0	0	1.2
181	0	1	0	0	0.6	0	0	1	1	0	1.2
182	0	1	0	0	0.6	0	0	1	0	1	1.2
183	0	1	0	0	0.3	1	0	0	0	0	1.2
184	0	0	0	0	1.1	1	0	1	1	0	1
185	1	1	0.8	1.1	0.8	0	0	1	1	0	1
186	1	1	0.75	1.1	0.75	1	1	0	0	0	1
187	0	0	0	0	1.1	1	1	1	0	0	1
188	0	1	0	0	0.4	1	0	0	0	0	1.8
189	0	0	0	0	0.7	0	0	0	0	1	0.8
190	0	0	0	0	0.5	1	0	0	0	0	0.8
191	0	0	0	0	2.25	1	1	1	1	0	0.8
192	0	0	0	0	1.75	1	1	1	0	0	0.8
193	0	0	0	0	1.3	0	0	1	1	0	0.8
194	0	0	0	0	1.75	1	1	0	0	1	0.8
195	0	0	0	0	1	1	0	0	0	1	0.8
196	1	1	0.8	1.4	0.8	0	0	1	0	0	1.4
197	1	0	0.65	3	1.7	1	1	0	0	0	1.4
198	1	0	0.8	2	2	1	0	1	0	1	1.4
199	1	0	1	1.8	1.8	0	0	1	0	1	1.4
200	1	0	0.8	3.75	3.75	1	0	1	1	1	1.4
201	0	1	0	0	1.4	1	1	0	0	0	1.6
202	0	0	0	0	1.8	0	0	1	0	0	1.8
203	0	0	0	0	3.75	1	1	0	0	0	1.8
204	0	1	0	0	1.4	1	1	1	0	1	1.2
205	0	1	0	0	1.8	0	0	1	1	0	1.4
206	0	1	0	0	0.65	1	0	0	0	0	1.4
207	0	0	0	0	1.8	0	0	1	1	0	1
208	0	0	0	0	2.4	1	1	1	0	0	1
209	0	1	0	0	0.8	1	0	0	0	0	1.4
210	0	0	0	0	0.3	1	0	0	0	0	1.6
211	1	1	0.25	0.5	0.25	0	0	0	0	1	1.6
212	1	1	0.35	0.5	0.35	0	0	1	0	0	1.6
213	0	1	0	0	0.6	0	0	1	0	1	1.4
214	0	1	0	0	1.4	1	1	1	0	1	1.4
215	1	1	0.6	0.9	0.6	0	0	1	1	0	1.4
216	1	1	0.45	0.7	0.45	0	0	0	0	1	1.4
217	0	0	0	0	0.8	0	0	1	1	0	1.2
218	1	1	0.45	0.75	0.45	0	0	0	0	1	1.6
219	0	1	0	0	0.6	0	0	1	0	0	1.4
220	0	0	0	0	2.25	0	0	1	1	0	1.8
221	0	0	0	0	3.75	1	1	1	0	1	0.8
222	0	0	0	0	3	1	1	0	0	1	0.8
223	0	0	0	0	2.1	1	1	0	0	0	0.8
224	0	0	0	0	2.25	0	0	1	1	0	0.8
225	1	0	1.5	3.5	3.5	1	1	0	0	1	1.8
226	0	1	0	0	1.1	1	0	0	0	0	1.8
227	0	0	0	0	1	0	0	1	1	0	1.8

228	0	0	0	0	0.8	1	0	1	0	0	1.8
229	0	1	0	0	1.75	1	1	1	0	1	1.2
230	0	1	0	0	0.5	1	0	0	0	0	1.2
231	0	1	0	0	1	0	0	1	1	0	1.2
232	0	1	0	0	3.25	1	1	1	0	1	1.4
233	0	1	0	0	2	1	0	1	0	0	1.4
234	0	1	0	0	0.3	1	0	0	0	0	1.2
235	0	1	0	0	1.1	1	1	1	0	1	1.4
236	0	1	0	0	0.6	0	0	0	0	1	1.4
237	0	1	0	0	0.6	0	0	1	0	0	1.4