[5] GAUSS PROBIT PROGRAM

PROBIT. PRG:

```
/*
** Probit MLE Estimation
*/

new ;

@ LOADING DATA @

load dat[1962,25] = mwpsid82.db ; @ Data on married women from PSID @

@ OPEN OUTPUT FILE @

output file = probit.out reset ;

@ DEFINE VARIABLES @
  nlf    = dat[.,1]  ; @ dummy variable for non-labor-force  @
  emp    = dat[.,2]  ; @ dummy varrable for employed workers @
  wrate  = dat[.,3]  ; @ hourly wage rate @
  lrate  = dat[.,4]  ; @ log of (hourly wage rate ($)+1) @
  edu    = dat[.,5]  ; @ years of schooling @
  urb    = dat[.,6]  ; @ dummy variable for urban resident @
  minor  = dat[.,7]  ; @ dummy variable for minority race @
  age    = dat[.,8]  ; @ age @
  tenure = dat[.,9]  ; @ # of months under current employer @
  expp   = dat[.,10] ; @ years of experience since age 18 @
  regs   = dat[.,11] ; @ dummy variable for South @
  occw   = dat[.,12] ; @ dummy variable for white collar @
  occb   = dat[.,13] ; @ dummy variable for blue collar @
  indumg = dat[.,14] ; @ dummy variable for manufacturing industry @
  indumn = dat[.,15] ; @ dummy variable for non-manu. industry @
  unionn  = dat[.,16] ; @ dummy variable for union membership @
  unempr  = dat[.,17] ; @ local unemployment rate @
  ofinc   = dat[.,18] ; @ other family income in 1980 ($) @
  lofinc  = dat[.,19] ; @ log of (other family income + 1) @
```

```
  kids    = dat[.,20] ; @ # of children of age <= 17 @
  wune80  = dat[.,21] ; @ UNKNOWN @
  hwork   = dat[.,22] ; @ hours of house work per week @
  uspell  = dat[.,23] ; @ # of unemployed weeks in 1980h @
  search  = dat[.,24] ; @ # of weeks looking for job in 1980 @
  kids5   = dat[.,25] ; @ # of children of age <= 5 @

@ DEFINE # of OBSERVATIONS @

n = rows(dat) ;

@  DEFINE DEPENDENT VARIABLE AND REGRESSORS @

   @ Dependent variable @

   yy = emp ;
   vny = {"emp"};

   @ Regressors @

   xx = ones(n,1)~edu~urb~minor~age~expp~kids~kids5~lofinc ;
   vnx = {"cons", "edu", "urb", "minor", "age", "expp",
           "kids", "kids5", "lofinc" };

/*
**  DO NOT CHANGE FROM HERE
*/

@   DEFINE INITIAL VALUE @

    bb = invpd(xx'xx)*(xx'yy) ;

library optmum;
#include optmum.ext;
optset ;

proc  f(b) ; local ltt   ;
   ltt = yy.*ln( cdfn(xx*b) ) + (1-yy).*ln( 1-cdfn(xx*b) ) ;
RETP(  -sumc(ltt)  ) ; ENDP  ;
```

```
b0 = bb ;
__title = "Probit MLE ";
_opgtol = 1e-4;
_opstmth = "newton, helf";
__output = 0 ;

{b,func,grad,retcode} = optprt(optmum(&f,b0)) ;

@ value of likelihood function @

   logl = -func ;

@ likelihood ratio test @

   rlogl = sumc(yy)*ln(sumc(yy)/n) + (n-sumc(yy))*ln(1-sumc(yy)/n) ;
   lrt   = 2*(logl-rlogl);
   lrdf  = cols(xx)-1 ;

@ Covariance matrix by Hessian @

   covhess = invpd(hessp(&f,b));

@ Covariance matrix by BHHH @

   proc  ff(b) ; local ltt   ;
      ltt = yy.*ln(cdfn(xx*b)) + (1-yy).*ln(1-cdfn(xx*b));
   RETP(  ltt  ) ; ENDP  ;

   gtt = gradp(&ff,b);
   covbhhh = invpd(gtt'gtt) ;

@ Robust covarince matrix @

   covrob = covhess*invpd(covbhhh)*covhess ;

@ Computing Standard errors @

   sehess = sqrt(diag(covhess));
```

```
   sebhhh = sqrt(diag(covbhhh));
   serob  = sqrt(diag(covrob)) ;
stathess = b~sehess~(b./sehess);
statbhhh = b~sebhhh~(b./sebhhh);
statrob  = b~serob~(b./serob);
yyhess = vnx~stathess;
yybhhh = vnx~statbhhh;
yyrob  = vnx~statrob;

let mask[1,4] = 0 1 1 1;
let fmt[4,3] =
    "-*.*s" 8 8
    "*.*lf" 10 4
    "*.*lf" 10 4
    "*.*lf" 10 4;

format /rd 10,4 ;
"" ;
"Probit Estimation Result" ;
"------------------------" ;
" dependent variable: " $vny ;
"" ;
" log likelihood:     " logl ;
"" ;
" Pseudo R-square:    " (1-logl/rlogl) ;
"" ;
" LR test, df, p-val: " lrt lrdf cdfchic(lrt,lrdf) ;
"" ;
" Using Hessian ";
"" ;
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(yyhess,mask,fmt);
"" ;
" Using BHHH ";
"" ;
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(yybhhh,mask,fmt);
"" ;
" Using Robust ";
```

```
"" ;
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(yyrob,mask,fmt);
output off ;
```

[OUTPUT]

```
Probit Estimation Result
------------------------
 dependent variable:        emp
 log likelihood:     -1167.9154
 Pseudo R-square:        0.1390
 LR test, df, p-val:   377.2165     8.0000     0.0000

Using Hessian
variable     coeff.  std. err.   t-st
cons         2.2010    0.5087    4.3269
edu          0.0846    0.0153    5.5340
urb          0.1553    0.0667    2.3296
minor        0.0480    0.0671    0.7153
age         -0.0344    0.0040   -8.6319
expp         0.0743    0.0059   12.6220
kids         0.0593    0.0264    2.2417
kids5       -0.5566    0.0513  -10.8400
lofinc      -0.2595    0.0556   -4.6661

Using BHHH
variable     coeff.  std. err.   t-st
cons         2.2010    0.5226    4.2115
edu          0.0846    0.0151    5.6031
urb          0.1553    0.0679    2.2886
minor        0.0480    0.0736    0.6522
age         -0.0344    0.0041   -8.4538
expp         0.0743    0.0059   12.4907
kids         0.0593    0.0259    2.2850
kids5       -0.5566    0.0470  -11.8383
lofinc      -0.2595    0.0566   -4.5859

Using Robust
variable     coeff.  std. err.   t-st
cons         2.2010    0.4975    4.4237
edu          0.0846    0.0156    5.4338
urb          0.1553    0.0658    2.3591
minor        0.0480    0.0614    0.7811
age         -0.0344    0.0039   -8.7092
expp         0.0743    0.0059   12.5926
kids         0.0593    0.0273    2.1742
kids5       -0.5566    0.0576   -9.6608
lofinc      -0.2595    0.0548   -4.7324
```

[6] GAUSS TOBIT PROGRAM

TOBIT.PRG:

```
/*
** Tobit MLE Estimation
*/

new ;

@ LOADING DATA @

    load dat[1962,25] = mwpsid82.db ;
     @ Data on married women from PSID 82 @

@ OPEN OUTPUT FILE @

    output file = tobit.out reset ;

@ DEFINE VARIABLES @

    dat     = delif(dat,dat[.,2] .== 0); @Delete nonemployed people@
    nlf     = dat[.,1]  ; @ dummy variable for non-labor-force  @
    emp     = dat[.,2]  ; @ dummy varrable for employed workers @
    wrate   = dat[.,3]  ; @ hourly wage rate @
    lrate   = dat[.,4]  ; @ log of (hourly wage rate ($)+1) @
    edu     = dat[.,5]  ; @ years of schooling @
    urb     = dat[.,6]  ; @ dummy variable for urban resident @
    minor   = dat[.,7]  ; @ dummy variable for minority race @
    age     = dat[.,8]  ; @ age @
    tenure  = dat[.,9]  ; @ # of months under current employer @
    expp    = dat[.,10] ; @ years of experience since age 18 @
    regs    = dat[.,11] ; @ dummy variable for South @
    occw    = dat[.,12] ; @ dummy variable for white collar @
    occb    = dat[.,13] ; @ dummy variable for blue collar @
    indumg  = dat[.,14] ; @ dummy var. for manufac. industry @
    indumn  = dat[.,15] ; @ dummy var. for non-manufac. industry @
    unionn  = dat[.,16] ; @ dummy variable for union membership @
    unempr  = dat[.,17] ; @ local unemployment rate @
```

```
    ofinc   = dat[.,18] ; @ other family income in 1980 ($) @
   lofinc  = dat[.,19] ; @ log of (other family income + 1) @
   kids    = dat[.,20] ; @ # of children of age <= 17 @
   wune80  = dat[.,21] ; @ UNKNOWN @
   hwork   = dat[.,22] ; @ hours of house work per week @
   uspell  = dat[.,23] ; @ # of unemployed weeks in 1980 @
   search  = dat[.,24] ; @ # of weeks looking for job in 1980 @
   kids5   = dat[.,25] ; @ # of children of age <= 5 @

@ DEFINE # of OBSERVATIONS @

    n = rows(dat) ;

@  DEFINE DEPENDENT VARIABLE AND REGRESSORS @

   @ Dependent variable @

   yy = uspell ;
   vny = {"uspell"};

   @ Regressors @

   xx = ones(n,1)~kids5~kids~edu~lofinc~age~expp~wrate~occw~occb  ;
   vnx = {"cons", "kids5", "kids", "edu", "lofinc", "age",
           "expp", "wrate", "occw", "occb" };

/*
**  DO NOT CHANGE FROM HERE
*/

@   DEFINE INITIAL VALUE @

   bb = invpd(xx'xx)*(xx'yy) ;
   ss = sqrt((yy-xx*bb)'(yy-xx*bb)/n) ;
   bb = bb|ss ;

@  dummy variable for yy @

   v  = 0 ;
```

```
   yd = dummy(yy,v);
   yd = yd[.,2] ;

   vnx1 = {"sigma"};
   vnx = vnx|vnx1 ;

library optmum;
#include optmum.ext;
optset ;

proc  f(b) ;
local ltt, q ;

   q   = rows(b);
   ltt = yd.*( -.5*ln(2*pi) -.5*ln(b[q]^2)
                -(.5/b[q]^2).*(yy-xx*b[1:q-1])^2 )
         + (1-yd).*ln( 1- cdfn( xx*b[1:q-1]/b[q]) ) ;

retp(  -sumc(ltt)  ) ;
endp ;

proc gradien(b) ;
local q, i, gge, g1, g2, gap, gac, poi,e ;

   q  = rows(b)    ;
   gge= zeros(q,1) ;

i = 1; do while i <= n;

   gap  = pdfn(xx[i,.]*b[1:q-1]/b[q]) ;
   gac  = cdfn(xx[i,.]*b[1:q-1]/b[q]) ;
   poi  = xx[i,.]*b[1:q-1]/b[q]       ;
   e    = yy[i] -xx[i,.]*b[1:q-1]     ;
   g1 = yd[i]*e/b[q]^2*xx[i,.]'
              + (1-yd[i])*( -gap/b[q]/(1-gac) )*xx[i,.]' ;

   g2 = yd[i]*( -1/b[q] + 1/b[q]^3*e^2  )
          + (1-yd[i])*( poi*gap/b[q]/(1-gac) ) ;
```

```
   gge = gge + (g1|g2) ;

i = i + 1 ; endo ;

retp( -gge' ) ; endp ;

proc hessi(b) ;
local q, i, he, h11, h12, h21, h22, gap, gac, poi ;
   q  = rows(b)    ;
   he = zeros(q,q) ;

i = 1; do while i <= n;
   gap  = pdfn(xx[i,.]*b[1:q-1]/b[q]) ;
   gac  = cdfn(xx[i,.]*b[1:q-1]/b[q]) ;
   poi  = xx[i,.]*b[1:q-1]/b[q]       ;

   h11 = yd[i]*(-1/b[q]^2)*xx[i,.]'xx[i,.]
            + (1-yd[i])*( poi*gap/(1-gac)/b[q]^2
                        - gap^2/(1-gac)^2/b[q]^2 )*xx[i,.]'xx[i,.] ;

   h12 = yd[i]*( -2/b[q]^3*(yy[i]-xx[i,.]*b[1:q-1])*xx[i,.]' )
            + (1-yd[i])*( (-poi^2*gap+gap)/b[q]^2/(1-gac)
                         + gap^2*poi/b[q]^2/(1-gac)^2 )*xx[i,.]'   ;
   h21 = h12' ;
   h22 = yd[i]*( 1/b[q]^2 - 3/b[q]^4*(yy[i]-xx[i,.]*b[1:q-1])^2 )
          + (1-yd[i])*( (-2*poi*gap+poi^3*gap)/b[q]^2/(1-gac)
                  - poi^2*gac^2/b[q]^2/(1-gac)^2 ) ;

   he = he + ((h11~h12)|(h21~h22)) ;

i = i + 1 ; endo ;

retp( -he ) ; endp ;

b0 = bb ;
__title = "Tobit MLE ";
_opgtol = 1e-6;
_opstmth = "bfgs,half";
__output = 0 ;
```

```
_opgdprc = &gradien;
_ophsprc = &hessi ;

{b,func,grad,retcode} = optmum(&f,b0) ;

@ value of likelihood function @

   logl = -func ;

@ Covariance matrix by Hessian @

covhess = invpd(hessi(b)) ;

@ Covariance matrix by BHHH @

   proc  ff(b) ;
   local ltt, q;

   q   = rows(b);
   ltt = yd.*( -.5*ln(2*pi) -.5*ln(b[q]^2)
                  -(.5/b[q]^2).*(yy-xx*b[1:q-1])^2 )
         + (1-yd).*ln( 1- cdfn( xx*b[1:q-1])/b[q] ) ;

   retp(  ltt  ) ;
   endp  ;

   gtt = gradp(&ff,b);
   covbhhh = invpd(gtt'gtt) ;

@ Robust covariance matrix @

   covrob = covhess*invpd(covbhhh)*covhess ;

@ Computing Standard errors @

   sehess = sqrt(diag(covhess));
   sebhhh = sqrt(diag(covbhhh));
   serob  = sqrt(diag(covrob)) ;
```

```
stathess = b~sehess~(b./sehess);
statbhhh = b~sebhhh~(b./sebhhh);
statrob  = b~serob~(b./serob);
yyhess = vnx~stathess;
yybhhh = vnx~statbhhh;
yyrob  = vnx~statrob;

let mask[1,4] = 0 1 1 1;
let fmt[4,3] =
    "-*.*s" 8 8
    "*.*lf" 10 4
    "*.*lf" 10 4
    "*.*lf" 10 4;

format /rd 10,4 ;
"" ;
"Tobit Estimation Result" ;
"------------------------" ;
" dependent variable: " $vny ;
"" ;
" log likelihood:     " logl ;
"" ;
" Using Hessian ";
"" ;
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(yyhess,mask,fmt);
" " ;
" Using BHHH ";
"" ;
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(yybhhh,mask,fmt);
"" ;
" Using Robust ";
"" ;
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(yyrob,mask,fmt);

output off ;
```

[OUTPUT]

```
Tobit Estimation Result
------------------------
 dependent variable:     uspell
 log likelihood:      -933.4950

 Using Hessian
variable     coeff.  std. err.   t-st
cons       -4.8340   22.3049   -0.2167
kids5       3.5827    2.4637    1.4542
kids       -0.1792    1.2566   -0.1426
edu        -0.2718    0.8127   -0.3345
lofinc      0.8886    2.2498    0.3950
age        -0.5739    0.2198   -2.6107
expp       -0.0550    0.3019   -0.1823
wrate      -1.8647    0.6981   -2.6710
occw       -2.1017    3.9879   -0.5270
occb       16.7014    4.4333    3.7672
sigma      28.0552    2.0858   13.4509

 Using BHHH
variable     coeff.  std. err.   t-st
cons       -4.8340   31.1688   -0.1551
kids5       3.5827    2.6498    1.3521
kids       -0.1792    1.4861   -0.1206
edu        -0.2718    1.0740   -0.2531
lofinc      0.8886    3.1981    0.2779
age        -0.5739    0.2621   -2.1895
expp       -0.0550    0.3643   -0.1511
wrate      -1.8647    0.5911   -3.1545
occw       -2.1017    4.3457   -0.4836
occb       16.7014    5.9349    2.8141
sigma      28.0552    2.6810   10.4644
```

Using Robust

variable	coeff.	std. err.	t-st
cons	-4.8340	17.7222	-0.2728
kids5	3.5827	2.3581	1.5193
kids	-0.1792	1.0989	-0.1631
edu	-0.2718	0.6642	-0.4092
lofinc	0.8886	1.7398	0.5108
age	-0.5739	0.1902	-3.0169
expp	-0.0550	0.2686	-0.2049
wrate	-1.8647	0.9460	-1.9712
occw	-2.1017	3.8091	-0.5518
occb	16.7014	3.5704	4.6778
sigma	28.0552	1.7858	15.7098

[7] GAUSS TRUNCATION MLE PROGRAM

TRUNC.PRG:

```
/*
** Truncation MLE Estimation
*/

new ;

@ LOADING DATA @

   load dat[1962,25] = mwpsid82.db ;
        @ Data on married women from PSID 82 @

@ OPEN OUTPUT FILE @

    output file = turnc.out reset ;

@ DEFINE VARIABLES @

    dat     = delif(dat,dat[.,2] .== 0) ; @Delete nonemployed people@
    nlf     = dat[.,1]  ; @ dummy variable for non-labor-force  @
    emp     = dat[.,2]  ; @ dummy varrable for employed workers @
    wrate   = dat[.,3]  ; @ hourly wage rate @
    lrate   = dat[.,4]  ; @ log of (hourly wage rate ($)+1) @
    edu     = dat[.,5]  ; @ years of schooling @
    urb     = dat[.,6]  ; @ dummy variable for urban resident @
    minor   = dat[.,7]  ; @ dummy variable for minority race @
    age     = dat[.,8]  ; @ age @
    tenure  = dat[.,9]  ; @ # of months under current employer @
    expp    = dat[.,10] ; @ years of experience since age 18 @
    regs    = dat[.,11] ; @ dummy variable for South @
    occw    = dat[.,12] ; @ dummy variable for white collar @
    occb    = dat[.,13] ; @ dummy variable for blue collar @
    indumg  = dat[.,14] ; @ dummy variable for manufac. industry @
    indumn  = dat[.,15] ; @ dummy variable for non-manufac. industry @
    unionn  = dat[.,16] ; @ dummy variable for union membership @
    unempr  = dat[.,17] ; @ local unemployment rate @
```

```
    ofinc   = dat[.,18] ; @ other family income in 1980 ($) @
   lofinc  = dat[.,19] ; @ log of (other family income + 1) @
   kids    = dat[.,20] ; @ # of children of age <= 17 @
   wune80  = dat[.,21] ; @ UNKNOWN @
   hwork   = dat[.,22] ; @ hours of house work per week @
   uspell  = dat[.,23] ; @ # of unemployed weeks in 1980 @
   search  = dat[.,24] ; @ # of weeks looking for job in 1980 @
   kids5   = dat[.,25] ; @ # of children of age <= 5 @

@ DEFINE # of OBSERVATIONS @

    n = rows(dat) ;

@  DEFINE DEPENDENT VARIABLE AND REGRESSORS @

   @ Dependent variable @

   yy = uspell ;
   vny = {"uspell"};

   @ Regressors @

   xx = ones(n,1)~kids5~kids~edu~lofinc~age~expp~wrate~occw~occb  ;
   vnx = {"cons", "kids5", "kids", "edu", "lofinc", "age",
           "expp", "wrate", "occw", "occb" };

/*
**  DO NOT CHANGE FROM HERE
*/

   vnx1 = {"sigma"};
   vnx = vnx|vnx1 ;

@  Deleting observations with y = 0  @

   zz = yy~xx ;
   zz = delif(zz,zz[.,1] .== 0);
   yy = zz[.,1] ;
   xx = zz[.,2:cols(zz)];
```

```
@   DEFINE INITIAL VALUE @

   bb = invpd(xx'xx)*(xx'yy) ;
   ss = sqrt((yy-xx*bb)'(yy-xx*bb)/n) ;
   bb = bb|ss ;

library optmum;
#include optmum.ext;
optset ;

proc  f(b) ;
local ltt, q ;

   q   = rows(b);
   ltt = (-.5*ln(2*pi)-.5*ln(b[q]^2)-(.5/b[q]^2).*(yy-xx*b[1:q-1])^2 )
            -ln( cdfn( xx*b[1:q-1]/abs(b[q]) ) ) ;

retp(  -sumc(ltt)  ) ;
endp ;

b0 = bb ;
__title = "Tobit MLE ";
_opgtol = 1e-6;
_opstmth = "bfgs,half";
__output = 1 ;

{b,func,grad,retcode} = optmum(&f,b0) ;

@ value of likelihood function @

   logl = -func ;

@ Covariance matrix by Hessian @

covhess = invpd(hessp(&f,b)) ;

@ Covariance matrix by BHHH @
```

```
   proc  ff(b) ;
   local ltt, q;
   q   = rows(b);
   ltt = (-.5*ln(2*pi)-.5*ln(b[q]^2)-(.5/b[q]^2).*(yy-xx*b[1:q-1])^2 )
         - ln( cdfn( xx*b[1:q-1]/abs(b[q]) ) ) ;

   retp(  ltt  ) ;
   endp  ;

   gtt = gradp(&ff,b);
   covbhhh = invpd(gtt'gtt) ;

@ Robust covariance matrix @

   covrob = covhess*invpd(covbhhh)*covhess ;

@ Computing Standard errors @

   sehess = sqrt(diag(covhess));
   sebhhh = sqrt(diag(covbhhh));
   serob  = sqrt(diag(covrob)) ;

stathess = b~sehess~(b./sehess);
statbhhh = b~sebhhh~(b./sebhhh);
statrob  = b~serob~(b./serob);
yyhess = vnx~stathess;
yybhhh = vnx~statbhhh;
yyrob  = vnx~statrob;

let mask[1,4] = 0 1 1 1;
let fmt[4,3] =
    "-*.*s" 8 8
    "*.*lf" 10 4
    "*.*lf" 10 4
    "*.*lf" 10 4;

format /rd 10,4 ;
"" ;
"Tobit Estimation Result" ;
```

```
"------------------------" ;
" dependent variable: " $vny ;
"" ;
" # of obs:           " rows(yy);
"";
" log likelihood:     " logl ;
"" ;
" Using Hessian ";
"" ;
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(yyhess,mask,fmt);
" " ;
" Using BHHH ";
"" ;
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(yybhhh,mask,fmt);
"" ;
" Using Robust ";
"" ;
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(yyrob,mask,fmt);

output off ;
```

[OUTPUT]

```
Truncation MLE Result
------------------------
 dependent variable:     uspell
 # of obs:              149.0000
 log likelihood:       -540.9255

 Using Hessian
variable     coeff.  std. err.   t-st
cons      -73.8799   45.9070   -1.6093
kids5       9.7156    3.5880    2.7078
kids       -2.6155    1.9766   -1.3232
edu        -3.0710    1.3927   -2.2051
lofinc     14.4962    4.8355    2.9979
age        -0.1635    0.3557   -0.4598
expp       -0.6647    0.5370   -1.2378
wrate      -2.7669    1.3399   -2.0650
occw        2.2922    5.4263    0.4224
occb       -3.9403    6.0062   -0.6560
sigma      16.6620    2.0786    8.0161

 Using BHHH
variable     coeff.  std. err.   t-st
cons      -73.8799   50.2120   -1.4714
kids5       9.7156    4.3665    2.2251
kids       -2.6155    2.3516   -1.1122
edu        -3.0710    1.7460   -1.7588
lofinc     14.4962    5.4724    2.6490
age        -0.1635    0.3813   -0.4289
expp       -0.6647    0.5639   -1.1789
wrate      -2.7669    1.4744   -1.8766
occw        2.2922    6.2740    0.3653
occb       -3.9403    6.3253   -0.6229
sigma      16.6620    2.9164    5.7132
```

Using Robust

variable	coeff.	std. err.	t-st
cons	-73.8799	46.3758	-1.5931
kids5	9.7156	3.3284	2.9190
kids	-2.6155	1.7147	-1.5253
edu	-3.0710	1.2146	-2.5284
lofinc	14.4962	4.6603	3.1106
age	-0.1635	0.3469	-0.4715
expp	-0.6647	0.5411	-1.2285
wrate	-2.7669	1.3726	-2.0158
occw	2.2922	4.9224	0.4657
occb	-3.9403	5.8350	-0.6753
sigma	16.6620	1.6353	10.1887

[8] GAUSS PROGRAM FOR HECKMAN'S TWO-STEP ESTIMATOR

HECKMAN1.PRG:

```
/*
** Heckman's two-step estimation
*/
new ;

@ LOADING DATA @
    load dat[1962,25] = mwpsid82.db ;
     @ Data on married women from PSID 82 @

@ OPEN OUTPUT FILE @
    output file = heckman1.out reset ;

@ DEFINE VARIABLES @
    nlf     = dat[.,1]  ; @ dummy variable for non-labor-force  @
    emp     = dat[.,2]  ; @ dummy varrable for employed workers @
    wrate   = dat[.,3]  ; @ hourly wage rate @
    lrate   = dat[.,4]  ; @ log of (hourly wage rate ($)+1) @
    edu     = dat[.,5]  ; @ years of schooling @
    urb     = dat[.,6]  ; @ dummy variable for urban resident @
    minor   = dat[.,7]  ; @ dummy variable for minority race @
    age     = dat[.,8]  ; @ age @
    tenure  = dat[.,9]  ; @ # of months under current employer @
    expp    = dat[.,10] ; @ years of experience since age 18 @
    regs    = dat[.,11] ; @ dummy variable for South @
    occw    = dat[.,12] ; @ dummy variable for white collar @
    occb    = dat[.,13] ; @ dummy variable for blue collar @
    indumg  = dat[.,14] ; @ dummy variable for manufac. industry @
    indumn  = dat[.,15] ; @ dummy variable for non-manufac. industry @
    unionn  = dat[.,16] ; @ dummy variable for union membership @
    unempr  = dat[.,17] ; @ local unemployment rate @
    ofinc   = dat[.,18] ; @ other family income in 1980 ($) @
    lofinc  = dat[.,19] ; @ log of (other family income + 1) @
    kids    = dat[.,20] ; @ # of children of age <= 17 @
    wune80  = dat[.,21] ; @ UNKNOWN @
    hwork   = dat[.,22] ; @ hours of house work per week @
    uspell  = dat[.,23] ; @ # of unemployed weeks in 1980 @
    search  = dat[.,24] ; @ # of weeks looking for job in 1980 @
    kids5   = dat[.,25] ; @ # of children of age <= 5 @

@ DEFINE # of OBSERVATIONS @
    n = rows(dat) ;

@  DEFINE DEPENDENT VARIABLE AND REGRESSORS @
```

```
   @ Dependent variable @
   yy1  = lrate   ;
   vny1 = {"lrate"} ;

   yy2  = emp     ;
   vny2 = {"emp"} ;

   @ Regressors @
   xx1  = ones(n,1)~edu~urb~minor~age~regs~unempr~expp~occw~occb;
   xx1  = xx1~unionn~tenure ;
   vnx1 = {"cons",   "edu",  "urb",  "minor", "age",    "regs",
           "unempr", "expp", "occw", "occb",  "unionn", "tenure" };

   xx2  = ones(n,1)~edu~urb~minor~age~regs~unempr~expp~lofinc~kids5 ;
   vnx2 = {"cons",   "edu",  "urb",    "minor", "age", "regs",
           "unempr", "expp", "lofinc", "kids5" };

/*
**  DO NOT CHANGE FROM HERE
*/

/* Begin the first Step */

@   DEFINE INITIAL VALUE @
   bb = invpd(xx2'xx2)*(xx2'yy2) ;

library optmum;
#include optmum.ext;
optset ;

proc  f(b) ; local ltt   ;

   ltt = yy2.*ln( cdfn(xx2*b) ) + (1-yy2).*ln( 1-cdfn(xx2*b) ) ;

RETP(  -sumc(ltt)  ) ; ENDP  ;

b0 = bb ;
__title = "Heckman's Two-Step Estimator";
_opgtol = 1e-4;
_opstmth = "bfgs, half";
__output = 0 ;
{b,func,grad,retcode} = optmum(&f,b0) ;

@ value of likelihood function @
   logl = -func ;

@ likelihood ratio test @
   rlogl = sumc(yy2)*ln(sumc(yy2)/n)
             + (n-sumc(yy2))*ln(1-sumc(yy2)/n) ;
```

```
   lrt   = 2*(logl-rlogl);
   lrdf  = cols(xx2)-1 ;

@ Covariance matrix by BHHH @

   proc  ff(b) ; local ltt   ;
      ltt = yy2.*ln(cdfn(xx2*b)) + (1-yy2).*ln(1-cdfn(xx2*b));
   RETP(  ltt  ) ; ENDP  ;

   gtt = gradp(&ff,b);
   covbhhh = invpd(gtt'gtt) ;

@ Computing Standard errors @

   sebhhh = sqrt(diag(covbhhh));
   statbhhh = b~sebhhh~(b./sebhhh);
   yybhhh = vnx2~statbhhh;

let mask[1,4] = 0 1 1 1;
let fmt[4,3] =
    "-*.*s" 8 8
    "*.*lf" 10 4
    "*.*lf" 10 4
    "*.*lf" 10 4;

format /rd 10,4 ;
"" ;
"Probit Estimation Result" ;
"------------------------" ;
" dependent variable: " $vny2 ;
"" ;
" # of observations:  "   rows(yy2) ;
"" ;
" log likelihood:     " logl ;
"" ;
" Pseudo R-square:    " (1-logl/rlogl) ;
"" ;
" LR test, df, p-val: " lrt lrdf cdfchic(lrt,lrdf) ;
"" ;
" Using BHHH ";
"" ;
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(yybhhh,mask,fmt);
"" ;

/* Begin the second Step */

@ Data selection @
yyy  = yy2 ;
```

```
zzz1 = yy1~xx1;
zzz1 = delif(zzz1,yyy .== 0);
yy1 = zzz1[.,1];
xx1 = zzz1[.,2:cols(zzz1)];

zzz2 = yy2~xx2;
zzz2 = delif(zzz2,yyy .== 0);
yy2 = zzz2[.,1];
xx2 = zzz2[.,2:cols(zzz2)];

@ Generating the selectivity regressor @
xb2  = xx2*b ;
lam  = pdfn(xb2)./cdfn(xb2) ;

@ Define the list of regressors at the second step @
zz    = xx1~lam ;

@ OLS regression with the selectivity regressor @
gam  = invpd(zz'zz)*(zz'yy1) ;
sele = yy1 - zz*gam;
rsq  = 1 - (sele'sele)/( yy1'yy1 - rows(yy1)*meanc(yy1)^2 ) ;

@  Residual from two-step  @
vv  = yy1 - zz*gam ;

@ Calculating corrected variance of errors in eq. 1 @
@ Define sig12 = c @
sig12 = gam[rows(gam)] ;
hh  = sig12*xx2.*(-xb2.*lam - lam^2 );
xi  = sig12^2*(xb2.*lam + lam^2) ;
vv2 = vv^2 ;

@ Corrected variance of error in the first equation @
sig11   = meanc(xi) + meanc(vv2) ;
pai     = sig11 - xi ;

@ Covariance Matrices @
cov0 = (vv'vv/(rows(zz)-cols(zz)))*invpd(zz'zz) ;
cov1 = invpd(zz'zz)*(zz'hh)*covbhhh*(hh'zz)*invpd(zz'zz)
       + invpd(zz'zz)*( zz'(zz.*pai) )*invpd(zz'zz) ;
cov2 = invpd(zz'zz)*(zz'hh)*covbhhh*(hh'zz)*invpd(zz'zz)
       + invpd(zz'zz)*( zz'(zz.*vv2) )*invpd(zz'zz) ;

@ Standard errors @
se0 = sqrt(diag(cov0)) ;
se1 = sqrt(diag(cov1)) ;
se2 = sqrt(diag(cov2)) ;

@ Statistics @
```

```
res0 = gam~se0~(gam./se0);
res1 = gam~se1~(gam./se1);
res2 = gam~se2~(gam./se2);
vnxx1 = {"sigma12"} ;
vnx1  = vnx1|vnxx1;
res0  = vnx1~res0 ;
res1  = vnx1~res1 ;
res2  = vnx1~res2 ;

"" ;
"Two-Step Estimation Result" ;
"------------------------" ;
" dependent variable:      " $vny1 ;
"" ;
" # of observations :      " rows(yy1) ;
"" ;
" R-Square:                " rsq ;
"" ;
" sigma_11                 " sig11 ;
"" ;
"Estimation results by usual OLS";
"";
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(res0,mask,fmt);
"" ;
"Estimation results by Greene's methods";
"";
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(res1,mask,fmt);
"" ;
"Estimation results by Ahn's methods";
"" ;
"variable     coeff.  std. err.   t-st " ;
yyprin = printfm(res2,mask,fmt);
"" ;

output off ;
```

[OUTPUT]

```
Probit Estimation Result
------------------------
 dependent variable:         emp
 # of observations:    1962.0000
 log likelihood:     -1164.0459
 Pseudo R-square:        0.1419
 LR test, df, p-val:   384.9554     9.0000     0.0000

 Using BHHH
variable     coeff.  std. err.   t-st
cons         2.2660    0.5531    4.0971
edu          0.0788    0.0148    5.3300
urb          0.1759    0.0690    2.5477
minor        0.0411    0.0772    0.5330
age         -0.0340    0.0041   -8.3703
regs         0.1167    0.0726    1.6070
unempr      -3.6906    1.3345   -2.7656
expp         0.0735    0.0059   12.4090
lofinc      -0.2316    0.0566   -4.0911
kids5       -0.5069    0.0379  -13.3739

Two-Step Estimation Result
--------------------------
 dependent variable:           lrate
 # of observations :        923.0000
 R-Square:                   0.3892
 sigma_11                    0.1097
```

Estimation results by usual OLS

variable	coeff.	std. err.	t-st
cons	0.5169	0.1123	4.6012
edu	0.0665	0.0059	11.2205
urb	0.1774	0.0246	7.2141
minor	-0.0916	0.0270	-3.3935
age	-0.0041	0.0016	-2.4726
regs	-0.0392	0.0255	-1.5365
unempr	-0.8525	0.4630	-1.8413
expp	0.0111	0.0031	3.5868
occw	0.1619	0.0296	5.4675
occb	0.1268	0.0357	3.5490
unionn	0.1509	0.0285	5.3018
tenure	0.0171	0.0029	5.9230
sigma12	0.1448	0.0476	3.0407

Estimation results by Greene's methods

variable	coeff.	std. err.	t-st
cons	0.5169	0.1142	4.5257
edu	0.0665	0.0060	11.0533
urb	0.1774	0.0251	7.0738
minor	-0.0916	0.0275	-3.3264
age	-0.0041	0.0017	-2.4486
regs	-0.0392	0.0260	-1.5060
unempr	-0.8525	0.4730	-1.8023
expp	0.0111	0.0031	3.5503
occw	0.1619	0.0293	5.5208
occb	0.1268	0.0354	3.5804
unionn	0.1509	0.0283	5.3357
tenure	0.0171	0.0029	5.9062
sigma12	0.1448	0.0478	3.0277

Estimation results by Ahn's methods

variable	coeff.	std. err.	t-st
cons	0.5169	0.1312	3.9410
edu	0.0665	0.0074	9.0180
urb	0.1774	0.0256	6.9189
minor	-0.0916	0.0257	-3.5691
age	-0.0041	0.0017	-2.4543
regs	-0.0392	0.0254	-1.5423
unempr	-0.8525	0.4947	-1.7232
expp	0.0111	0.0031	3.6119
occw	0.1619	0.0293	5.5339
occb	0.1268	0.0345	3.6745
unionn	0.1509	0.0279	5.4153
tenure	0.0171	0.0030	5.6775
sigma12	0.1448	0.0429	3.3737