Recursively Simulating Multinomial Multiperiod Probit Probabilities

Geweke, John and Keane, Michael and Runkle, David

1994

RECURSIVELY SIMULATING MULTINOMIAL MULTIPERIOD PROBIT PROBABILITIES

John Geweke, Michael Keane, David Runkle, University of Minnesota and Federal Reserve Bank of Minneapolis
David Runkle, Federal Reserve Bank of Minneapolis, P.O. Box 291, Minneapolis, MN 55480-0291

Keywords: Simulation, Probit

I. Introduction

This paper describes how to recursively simulate individual choice probabilities in a multiperiod multinomial probit model. Section II presents the model. Section III presents the simulator. The Appendix presents GAUSS code for simulating probabilities when the errors in the multinomial probit model contain both an AR 1 error and a random individual effect. For evidence on the sampling performance of the simulator in conjunction with the method of simulated moments and simulated maximum likelihood see Geweke, Keane, and Runkle (1994b).

II. The Model

Assume that agents choose among a set of J mutually exclusive alternatives in each of T time periods. If individual i chooses alternative j at time t, he/she derives utility

$$U_{ijt} = X_{ijt}'\beta_j + \epsilon_{ijt}$$

$$(j = 1,\ldots,J;\ t = 1,\ldots,T),$$

where X_{ijt} is a $p \times 1$ vector of exogenous variables, β_j is a $p \times 1$ vector of corresponding coefficients, and ϵ_{ijt} is a random shock to utility that is known to the agent but unknown to the econometrician. Choice j is made at time t if $U_{ijt} > U_{ikt}$ for all $k \neq j$. The econometrician observes the choice

$$d_{ijt} = \begin{cases} 1 \text{ if i chooses j at time t} \\ 0 \text{ otherwise,} \end{cases}$$

but not the utility of any choice. The probit model is obtained by assuming

$$\epsilon_i \equiv (\epsilon_{i11},\ldots,\epsilon_{iJ1},\ldots,\epsilon_{i1T},\ldots,\epsilon_{iJT})'$$

$$\sim \text{IIDN}(0,\Sigma),\quad \Sigma = [\sigma_{jk}].$$

Since choices only depend on utility differences, it is conventional to measure utility *relative* to alternative J. Since the scale of utilities is indeterminate, it is also conventional to normalize by setting the variance of the error term corresponding to the first alternative in the transformed model equal to one. Thus, we define

$$(2.1)\quad U_{ijt}^* = (U_{ijt} - U_{iJt})(\sigma_{11} + \sigma_{JJ} - 2\sigma_{1J})^{-1/2}$$

$$= [(X_{ijt}'\beta_j - X_{iJt}'\beta_J) + (\epsilon_{ijt} - \epsilon_{iJt})](\sigma_{11} + \sigma_{JJ} - 2\sigma_{1J})^{-1/2}$$

$$= X_{ijt}^{*\prime}\beta_j^* + \epsilon_{ijt}^* \quad (j = 1,J;\ t = 1,T),$$

where $X_{ijt}^*(j = 1,\ldots,J)$ is the appropriate transformation of $X_{ijt}(j = 1,\ldots,J)$ and $\beta_j^*(j = 1,\ldots,J)$ is the appropriate transformation of $\beta_j(j = 1,\ldots,J)$. (Notice that $U_{iJt}^* = 0$ and $\epsilon_{iJt}^* = 0$.) We further define

$$(2.2)\quad \epsilon_i^* = (\epsilon_{i11}^*,\ldots,\epsilon_{i,J-1,1}^*,\ldots,\epsilon_{i1T}^*,\ldots,\epsilon_{i,J-1,T}^*)'$$

$$\epsilon_i^* \sim \text{IIDN}(0,\Sigma^*),\quad \Sigma^* = [\sigma_{jk}^*],$$

where Σ^* is the corresponding appropriate transformation of Σ; by construction, $\sigma_{11}^* = 1$.

In the notation of the transformed model, choice j is made at time t if

$$(2.3)\quad U_{ijt}^* > U_{ikt}^* \text{ for all } k \neq j\ (j = 1,\ldots,J-1).$$

In order to have a compact notation for the sequence of choices observed for person i, define

$$(2.4)\quad d_{it} = (d_{i1t},\ldots,d_{iJt}),\quad d_i = (d_{i1},\ldots,d_{iT}),$$

and

$$j_{it} = \{j \mid d_{ijt} = 1\}.$$

If $P(d_i)$ denotes the probability that i chooses the sequence d_i,

$$P(d_i) = P(U_{i,j_{it},t}^* > U_{ikt}^*\ \forall\ k \neq j_{it},\ t = 1,\ldots,T)$$

$$= P[\epsilon_{i,j_{it},t}^* - \epsilon_{ikt}^* > X_{ikt}^{*\prime}\beta_k^* - X_{i,j_{it},t}^{*\prime}\beta_{j_{it}}^*\ \forall\ k \neq j_{it},\ (t = 1,\ldots,T)].$$

If the ϵ^*_{ijt} are serially independent, then this is the product of T integrals each of dimension J − 1. However, if the ϵ^*_{ijt} are serially correlated, this is in general a T · (J−1) variate integral. As T and/or J grow, inference requiring exact evaluation of such integrals rapidly becomes infeasible. Much of the earlier work on the MMP model sought to avoid this problem by imposing low order factor structures on Σ^*. For example, if a random effects structure is imposed, the order of integration is reduced to 2 · (J−1). The goal of simulation based inference is to allow a richer covariance structure to be used.

III. Simulation of Choice Sequence Probabilities

Classical approaches to inference in the MMP model rely on Monte-Carlo simulation of the choice sequence probabilities $P(d_i)$ and substitution of these simulated probabilities into likelihood functions or moment conditions. In an extensive study of alternative methods for simulation of multinomial orthant probabilities, Hajivassiliou, McFadden, and Ruud (1993) conclude that the GHK probability simulator, due to Keane (1990), Geweke (1991), and Hajivassiliou and McFadden (1994), is the most accurate of all methods considered. Geweke, Keane, and Runkle (1994a), in a Monte-Carlo study of alternative approaches to simulation based inference in the single period multinomial probit model, concluded that classical methods based on GHK substantially outperformed classical methods based on kernel smoothed probability simulators. Here we provide a description of the GHK simulator applied to the simulation of choice probabilities on the MMP model just described. For a proof of the unbiasedness of the GHK simulator in general, see Börsch-Supan and Hajivassiliou (1993).

To describe the GHK simulator it is useful to define some additional notation. Let

$$\tilde{U}^j_{ikt} = U^*_{ikt} - U^*_{ijt} (j = 1,\ldots,J;\ t = 1,\ldots,T),$$

$$\tilde{\epsilon}^j_{ikt} = \epsilon^*_{ikt} - \epsilon^*_{ijt}.$$

(Notice that $\tilde{U}^j_{ijt} = 0$ and $\tilde{\epsilon}^j_{ijt} = 0$.) Choice j is made at t if the J − 1 constraints $\tilde{U}^j_{ikt} < 0$ for all $k \neq j$ are satisfied. Further let

$$\tilde{\epsilon}_{it}(-j) = (\tilde{\epsilon}^j_{i1t},\ldots,\tilde{\epsilon}^j_{i,j-1,t},\tilde{\epsilon}^j_{i,j+1,t},\ldots,\tilde{\epsilon}^j_{iJt})'$$

and

$$\tilde{\epsilon}(d_i) = (\tilde{\epsilon}_{i1}(-j_{i1}),\ldots,\tilde{\epsilon}_{iT}(-j_{iT}))'$$

where d_i is the choice vector defined in (2.4). Thus $\tilde{\epsilon}(d_i) \sim \text{IIDN}(0,\tilde{\Sigma}(d_i))$, where $\tilde{\Sigma}(d_i)$ is the appropriate transformation of Σ^*.

Let $\tilde{A}(d_i)$ be the unique lower triangular Cholesky decomposition $\tilde{\Sigma}(d_i) = \tilde{A}(d_i)\tilde{A}(d_i)'$. Then $\tilde{\epsilon}(d_i) = \tilde{A}(d_i)\tilde{\eta}(d_i)$, where (suppressing the i subscript) $\tilde{\eta}_t(-j_t) = (\eta_{1t},\ldots,\eta_{j_t-1,t},\eta_{j_t+1,t},\ldots,\eta_{Jt})'$, $\tilde{\eta}(d_i) = (\tilde{\eta}_1(-j_1),\ldots,\tilde{\eta}_T(-j_T))'$, and $\eta_{ijt} \sim \text{IIDN}(0,1)$ for all i, j, t.

Finally, define $\tilde{U}^j_{ikt}(\tilde{\eta}^\ell_{11},\ldots,\tilde{\eta}^\ell_{j,t-1},\tilde{\eta}^\ell_{1t},\ldots,\tilde{\eta}^\ell_{pt})$ as the value of $\tilde{u}^j_{ikt}$ when the random variables $(\tilde{\eta}_{11},\ldots,\tilde{\eta}_{J,t-1},\tilde{\eta}_{1t},\ldots,\tilde{\eta}_{pt})$ are fixed at the *draw* $(\tilde{\eta}^\ell_{11},\ldots,\tilde{\eta}^\ell_{J,t-1},\tilde{\eta}^\ell_{1t},\ldots,\tilde{\eta}^\ell_{pt})$. Note that for p = k this is a number, and for p < k this is a random variable. Then, the GHK simulator for the probability of the choice sequence $(d_{i1},\ldots,d_{it})$, or equivalently $(j_{i1},\ldots,j_{it})$, is constructed as follows (suppressing the i subscript):

Period 1:
Step:

(1) Draw η^ℓ_{11}

s.t. $\tilde{U}^{j_1}_{11}(\eta^\ell_{11}) < 0$

⋮

(j_1−1) Draw $\eta^\ell_{j_1-1,1}$

s.t. $\tilde{U}^{j_1}_{j_1-1,1}(\eta^\ell_{11},\ldots,\eta^\ell_{j_1-1,1}) < 0$

(j_1) Skip $\eta^\ell_{j_1,1}$

(j_1+1) Draw $\eta^\ell_{j_1+1,1}$

s.t. $\tilde{U}^{j_1}_{j_1+1,1}(\eta^\ell_{11},\ldots,\eta^\ell_{j_1-1,1},\eta^\ell_{j_1+1,1}) < 0$

⋮

(J) Draw η^ℓ_{J1}

s.t. $\tilde{U}^{j_1}_{J1}(\eta^\ell_{11},\ldots,\eta^\ell_{j_1-1,1},\eta^\ell_{j_1+1,1},\ldots,\eta^\ell_{J1}) < 0$

⋮

Period t:
Step:

(1) Draw η^ℓ_{1t}

s.t. $\tilde{U}^{j_t}_{1t}(\tilde{\eta}^\ell_{11},\ldots,\tilde{\eta}^\ell_{J,t-1},\eta^\ell_{1t}) < 0$

⋮

(j_t−1) Draw $\eta^\ell_{j_t-1,t}$

s.t. $\tilde{U}^{j_t}_{j_t-1,t}(\tilde{\eta}^\ell_{11},\ldots,\tilde{\eta}^\ell_{J,t-1},\eta^\ell_{1t},\ldots,\eta^\ell_{j_t-1,t}) < 0$

(j_t) Skip $\eta^\ell_{j_t t}$

(j_t+1) Draw $\eta^t_{j_t+1,t}$

s.t. $\tilde{U}^{j_t}_{j_t+1,t}(\tilde{\eta}^t_{11},\ldots,\tilde{\eta}^t_{J_1,t-1},\eta^t_{1t},\ldots,\eta^t_{j_t-1,t},\eta^t_{j_t+1,t}) < 0$

$\vdots$

(J) Draw η^t_{Jt}

s.t. $\tilde{U}^{j_t}_{Jt}(\tilde{\eta}^t_{11},\ldots,\tilde{\eta}^t_{J_1,t-1},\eta^t_{1t},\ldots,\eta^t_{j_t-1,t},\eta^t_{j_t+1,t},\ldots,\eta^t_{Jt}) < 0$

and finally, construct:

$$\hat{P}_{GHK}(d_1,\ldots,d_t|\beta^*,\Sigma^*,X^*)$$

$$= \frac{1}{M}\sum_{t=1}^{M} P(\tilde{U}^{j_1}_{11} < 0)\prod_{k=2}^{j_1-1} P\left[\tilde{U}^{j_1}_{k1}(\eta^t_{11},\ldots,\eta^t_{k-1,1}) < 0\right]$$

$$\cdot P\left[\tilde{U}^{j_1}_{j_1+1,1}(\eta^t_{11},\ldots,\eta^t_{j_1-1,1}) < 0\right]\prod_{k=j_1+2}^{J}$$

$$P\left[\tilde{U}^{j_1}_{k1}(\eta^t_{11},\ldots,\eta^t_{j_1-1,1},\eta^t_{j_1+1,1},\ldots,\eta^t_{k-1,1}) < 0\right]$$

$$\cdot \ldots \cdot P\left(\tilde{U}^{j_t}_{1t}(\tilde{\eta}^t_{11},\ldots,\tilde{\eta}^t_{Jt-1}) < 0\right)\prod_{k=2}^{j_t-1}$$

$$P\left[\tilde{U}^{j_t}_{kt}(\tilde{\eta}^t_{11},\ldots,\tilde{\eta}^t_{Jt-1},\eta^t_{1t},\ldots,\eta^t_{k-1,t}) < 0\right]$$

$$P\left[\tilde{U}^{j_t}_{j_t+1,t}(\tilde{\eta}^t_{11},\ldots,\tilde{\eta}^t_{J_1,t-1},\tilde{\eta}^t_{1t},\ldots,\tilde{\eta}^t_{j_t-1,t}) < 0\right]\prod_{k=j_t+2}^{J}$$

$$P\left[\tilde{U}^{j_t}_{kt}(\tilde{\eta}^t_{11},\ldots,\tilde{\eta}^t_{Jt-1},\eta^t_{1t},\ldots,\eta^t_{j_t-1,t},\eta^t_{j_t+1,t},\ldots,\eta^t_{k-1,t}) < 0\right].$$

References

Börsch-Supan, A., and Hajivassiliou, V. 1993. Smooth unbiased multivariate probability simulators for maximum likelihood estimation of limited dependent variable models. *Journal of Econometrics* 58: 347–68.

Geweke, J. 1991. Efficient simulation from the multivariate normal and student-t distributions subject to linear constraints. *Computer Science and Statistics: Proceedings of the Twenty-Third Symposium on the Interface*, pp. 571–78. Alexandria, VA: American Statistical Association.

Geweke, J., Keane, M., and Runkle, D. 1994a. Alternative computational approaches to statistical inference in the multinomial probit model. Forthcoming in *Review of Economics and Statistics.*

Geweke, J., Keane, M., and Runkle, D. 1994b. Statistical inference in the multinomial multiperiod probit model. Research Department Staff Report 177. Federal Reserve Bank of Minneapolis.

Hajivassiliou, V., McFadden, D.; and Rudd, P. 1993. Simulation of multivariate normal orthan probabilities: Methods and programs. Cowles Foundation Disucssion Paper 1021. Yale University.

Hajivassiliou, V., and McFadden, D. 1994. A method of simulated scores for the estimation of LDV models with an application to external debt crises. Discussion Paper #967. Cowles Foundation.

Keane, M. 1990. Four essays in empirical macro and labor economics. Ph.D. dissertation. Brown University.

Appendix

This code is available via anonymous ftp access at res.mpls.frb.fed.us in the /pub/msm subdirectory.

```
proc
ghkprob
(persdata,drawdat,param,sigstar,capj,capt,nk,np,nl);
/*John F. Geweke, Michael P. Keane, David E.
Runkle. These routines can be used and distributed
without any royalty except that users must cite:

Michael P. Keane (1994), "A Computationally Practical
Simulation Estimation for Panel Data," Econometrica
62 (1), 95-116.

John F. Geweke, Michael P. Keane, David E. Runkle
(1994), "Statistical Inference in the Multinomial Multi-
period Probit Model," Federal Reserve Bank of Min-
neapolis Research Department Staff Report 177.*/

/* this routine creates the capj x capt matrix of transi-
   tion
   probabilities for a single person using the ghk simu-
   lator
   Inputs are:
   persdata : matrix of data for the person
              1 column-choice for person n in period
              t.
              This is a number from 1 to capj.
              nk columns-individual data
              np*capj columns-choice specific data

   drawdat  : the set of uniform draws for the person
              length = nl*capj*capj*capt
              (note: these MUST BE kept constant for
              a given person throughout estimation)

   param    : the parameter vector with following ele-
              ments in order
```

```
   number of elments     description
   1                     @variance of ind effect@
   capj-1                @elements of rho vector@
   ((capj)*(capj-1)/2)-1 @elements of phi chol
                         decomp@
   nk*(capj-1)           @individual coefficients@
   np*capj;              @alternative coefficients@

   capj  : the number of choices in each time period
   capt  : the number of time periods
   nk    : the number of individual-specific variables
   np    : the number of alternative-specific variables
   nl    : the number of draws */

local    amat,choicevc,drawmat,drawno,drawprba,dra
         wprob,etadrawa,etadraws,j,persprob,proba,si
         m-prob,t,tmat,truncp,utwchoic,xbvec,xbtwid;

choicevc  =   persdata[.,1];
etadraws  =   zeros(nl,(capj-1)*capt); @stored eta-
              draw matrix for current round@
etadrawa  =   zeros(nl,capj-1); @etadraw for actual
              choice in current period@
persprob  =   zeros(capj,capt); @transition prob for
              each person@
proba     =   zeros(capt,1); @probability of actual
              sequence cumulative @
drawprob  =   zeros(nl,(capj-1)*capt);
drawprba  =   zeros(nl,(capj-1));

drawmat   =   reshape(drawdat,nl,cols(drawdat)/nl);
drawno    =   1;
t         =   1;
do until t > capt;
   xbvec = xbveccr(persdata,param,capj,capt,nk,np,t);
j=1;
do until j > capj;
   @create tranformed data and chol decomp@
   tmat=transmat(capt,capj,t,j,choicevc);
   amat=atwch(tmat,sigstar);
   xbtwid=tmat*xbvec;

/* evaluate probability of choice j by simulating
   probability of negative eta twiddles for capj-1
   errors*/

@choice index for prob eval is utwchoic@
utwchoic=1;
do until utwchoic > capj-1;
if ( (t == 1) and ( utwchoic == 1 ) );
   truncp =  -(xbtwid[(t-1)*(capj-1)+utwchoic,.])./
             amat[(t-1)*(capj-1)+utwchoic,(t-1)
             *(capj-1) +utwchoic]. *ones(nl,1);
else;
   truncp =  -(xbtwid[(t-1)*(capj-1)+utwchoic,.]
             +sumc((amat[(t-1)*(capj-1)
             +utwchoic,1:(t-1)*(capj-1)+utwchoic-1]
             .*etadraws[.,1:(t-1)*(capj-1)
             +utwchoic-1])')).amat[(t-1)*(capj-1)
             +utwchoic,(t-1)*(capj-1)+utwchoic]
             .*ones(nl,1);
endif;
drawprob[.,(t-1)*(capj-1)+utwchoic] = cdfn(truncp);
etadraws[.,(t-1)*(capj-1)+utwchoic] = gauinv
   (cdfn(truncp).*drawmat[.,drawno]);
drawno   =  drawno+1;
utwchoic =  utwchoic+1;
endo;
@now do cleanup before cycling on j@
   simprob = sumc(exp(sumc(ln(drawprob
             [.,1:(capj-1)*t])')))/nl;
if t == 1;
   persprob[j,t]=simprob;
else;
   persprob[j,t]=simprob/proba[t-1,1];
endif;

if (choicevc[t,.] == j );
   etadrawa = etadraws[.,(t-1)*(capj-1)+1:t*(capj-1)];
   proba[t,1] = simprob;
   drawprba=drawprob[.,(t-1)*(capj-1)+1:t*(capj-1)];

endif;
   j=j+1;
   endo;
@now do cleanup before cycling on t@
@use draws corresponding to actual choices in this
 period@
   etadraws[.,(t-1)*(capj-1)+1:t*(capj-1)] = etadrawa;
   drawprob[.,(t-1)*(capj-1)+1:t*(capj-1)]=drawprba;
   t=t+1;
   endo;
retp(persprob);
endp;

proc  blderrcv(capt,capj,param);
   local bmat,crosscov,i,j,ind,phi,phichfl,phichol,phiin,
   phivec,rhoin,rhomat,trick,strt,covmat;
/* this is a function to take portions of the parameter
   vector and turn them into the elements of the covari-
   ance matrix for the ar-1 model with individual ef-
   fects for MSM */

format 6,4;
   @capj   = number of choices per period@
   @capt   = number of periods @
   covmat  = zeros((capj)*capt,(capj)*capt); @output
             covariance matrix@
```

```
   rhomat  = covmat; @dummy matrix to be used for

           autocorrelations@
   ind     = param[1,1]; @variance of individual

effect@
   "ind ";;ind;
   rhoin   = param[2:capj,.]; @ error autoregressive

           parameters@
   "rhoin ";;rhoin;
   strt=capj+1;
   phiin   =  param[strt:strt+((capj)*(capj-1)/2)-2,.];

           @elements of phi chol decomp@
   "phiin ";;phiin;

   /* vec of cholesky decomp of transformed system

   without 1,1 element) */
   phivec= ones(1,1)|phiin;
   /* the following is a trick to create the lower-
      triangular version of phichol */
   trick = (rndn(capj-1,capj-1));
   phichol = xpnd(phivec).*(chol(trick'trick)'.ne 0);
             clear trick;
   /* create expanded version of phi to be used in GHK
      transformations */
   phichfl = (phichol~(zeros(capj-1,1)))|
             zeros(1,capj);
   phi = phichfl*phichfl';
   /* create basic matrix using phi and cross covari-
      ances in errors */
   crosscov=1/(1-(rhoin.*rhoin'));
   bmat = phi.*((crosscov~(zeros(capj-1,1)))|
          zeros(1,capj));
   /* now create matrix to take account of autocorrela-
      tions */
   i=1;
   do until i > capt;
   j=1;
   do until j > capt;
rhomat[(i-1)*(capj)+1:i*(capj),(j-1)*(capj)+1:j*(capj)]
   =ones(capj-1,capj)|zeros(1,capj)).*((rhoin^abs(i-j))
   |(0))'*( i < j) +(((ones(capj-1,capj)|zeros
   (1,capj)).*((rhoin^abs(i-j))|(0))')' *(1-(i <j)));

   j=j+1;
   endo;
   i=i+1;
   endo;
      @put everything together into the covariance
      matrix@
   covmat =  (rhomat.*(ones(capt,capt).*.bmat))+ind*
             (rhomat .> 0);
   retp(covmat);
endp;

proc selmat(capj,j);
   /* this procedure creates the one-period selection
      matrix */

   @j      = choice selected@
   @capj   = maximum number of choices@
   @transj = one period selection matrix@

   local m1,m2,transj;
   m1   = -1*ones(capj-1,1);
   m2   = eye(capj-1);
   if j == 1;
      transj = m1~m2;
   elseif j == capj;
      transj = m2~m1;
   else;
      transj = m2[.,1:j-1]~m1~m2[.,j:capj-1];
   endif;
   retp(transj);
endp;

proc transmat(capt,capj,t,j,choicevc);
   /* this procedure creates the appropriate selection
      matrix for choice j in period t.
      note that this proceedure conditions on actual
      choices in periods t-1 and before */

   @ capt      = maximum number of periods @
   @ capj      = maximum number of choices @
   @ t         = current time period @
   @ j         = current choice @
   @ choicevc = vector of choices made by person n@
   @ trans     = selection matrix for choice j in
                 period t@
   local jj,tt,trans;
   if t == 1;
   trans=selmat(capj,j)~zeros(capj-1,(capj)*(capt-1));
   else;
      tt  = 1;
      jj     = choicevc[tt,1];
      trans = selmat(capj,jj)~zeros(capj-1,(capj)*
              (capt-1));
      tt     = 2;
      do until tt> t;
         if tt < t;
            jj=choicevc[tt,1];
         else;
            jj=j;
         endif;
```

```
        if tt < capt;
      trans = trans|((zeros(capj-1,capj*(tt-1)))~

      selmat(capj,jj)~(zeros(capj-1,capj*capt

-tt))));
        else;
      trans = trans|((zeros(capj-1,capj*(tt-1)))~

          selmat(capj,jj));
        endif;

        tt=tt+1;
      endo;
   endif;
retp(trans);
endp;

proc gauinv(p);
   /* vector inverse gaussian distribution from kennedy
      and gentle page 95*/
   /* constants */
   local p0,p1,p2,p3,p4,q0,q1,q2,q3,q4,opy,xp,lim;

   lim = 10e-20; p0=-0.322232431088; p1=-1.0;

   p2 = -0.342242088547;
   p3 = -0.0204231210245;
   p4 = -0.453642210148*10e-05;
   q0 =  0.0993484626060;
   q1 =  0.588581570495;
   q2 =  0.531103462366;
   q3 =  0.103537752850;
   q4 =  0.38560700634*10E-03;
   op =  p;
   p  =  (p .> .5).*(1-p) + (1-(p .> .5)).*p;
   y  =  sqrt(ln(1/(p.^2)));
   xp =  (y+((((y*p4 + p3).*y + p2).*y +p1).*y
   '        +p0)./((((y*q4+q3).*y+q2).*y+q1).*y+q
         0)).*(1-( p .< lim));xp =   (op .<.5).
         *(-xp) + (1-(op .< .5)).*(xp);
   retp(xp);
endp;

proc xbveccr(persdata,param,capj,capt,nk,np,t);
   local betastrt,gammstrt,gam,x,z,xbvec,tt,i,beta;
   /* subroutine to create xb +zg to determine trunc
      points*/
   betastr t= 1+ capj-1+ ((capj)*(capj-1)/2);
   beta = reshape(param[betastrt:betastrt+nk*
         (capj-1)-1,.],nk,capj-1);
   gammstrt = betastrt+nk*(capj-1);
   gam = reshape(param[gammstrt:gammstrt+
         np*(capj)-1,.],np,capj);
   x = persdata[.,2:2+nk-1];
   z = persdata[.,2+nk:2+nk+np*capj-1];
   xbvec = zeros(capj*capt,1);
   tt = 1;
   do until tt > t;
      i=1;
      do until i> capj-1; xbvec[(tt-1)*capj+i,.]=
                  x[tt,.]*beta[.,i]
                  +z[tt,(i-1)*np+1:i*np]*gam[.,i]
                  +z[tt,(capj-1)*np+1:capj*np]
                  *gam[.,capj];i=i+1;
      endo;
      tt=tt+1;
   endo;
   retp(xbvec);
endp;

proc atwch(tmat,sigstar);
   /* this proceedure creates the cholesky matrix of
      sigtwid using sigstar and the transformation
      matrix */
   local atwmat,sigtwid;
   sigtwid = tmat*sigstar*tmat';
   atwmat = chol(sigtwid)';
   retp(atwmat);
endp;
```