Dynamic Discrete Choice Panel Data Model for "Measuring Individual Competitiveness and its Impact on Sporting Success"*

Julene Palacios-Saracho† Ander Palacios-Saracho‡

TECHNICAL APPENDIX
September 2023

We describe here the semi–parametric, random effects, discrete choice model with predetermined variables that is implemented. The model is based on Arellano and Carrasco (2003) and controls for the effects of unobservable heterogeneity and for state dependence. As is well known, in the presence of lagged endogenous variables in linear models with additive effects, the standard procedure in the econometrics literature is to consider instrumental-variables estimates that exploit the lack of correlation between lagged values of the variables and future errors in first differences. In non-linear settings such as ours, however, few results are available.[1] The basic idea of the Arellano-Carrasco model is to define conditional probabilities for every possible sequence of realizations of the state variables. Then, the estimator computes the probability of a given outcome along every possible path of past realizations of the endogenous regressors. This means that the panel data structure allows the identification of the effects of individual unobserved heterogeneity since outcomes can be different even when individuals share the same history of realizations of the state variables.

Formally, consider two discrete outcomes denoted as $y_{it} = \{0, 1\}$. These could be discrete measures of, for example, performance or success such as reaching professionalism or not. The probability of each of them depends on the specific sequence of past outcomes and the "state process" of the individual. Since outcomes can be different, different experiences change the information set and the expected realizations of future outcomes. To be specific, the probability of a given outcome may depend on certain intrinsic characteristics of the individual, as well as on his expectation on the realization of the final outcome:

$$y_{it} = \mathbf{1}\left\{\gamma + \beta z_{it} + E\left(\eta_i \mid w_i^t\right) + \varepsilon_{it} \geq 0\right\}, \qquad \varepsilon_{it} \mid w_i^t \sim N\left(0, \sigma_t^2\right),$$

*We are grateful to Eugenio Miravete for invaluable help and for his generosity with the Gauss program.

†Universidad de Deusto, Spain (https://www.deusto.es/en/home) and Athletic Club de Bilbao (https://www.athletic-club.eus/en). Email: jpalaciossaracho@gmail.com.

‡Universidad de Deusto, Spain (https://www.deusto.es/en/home). Email: ander.palacios@gmail.com.

[1] For fixed effects the few available methods are case-specific (logit and Poisson) and, in practice, lead to estimators that do not converge at the usual $\sqrt{n}$-rate. In the case of random effects, the main difficulty is the so-called initial conditions problem: if one begins to observe subjects after the "process" in question is already in progress, it is necessary to isolate the effect of the first lagged dependent variable from the individual-specific effect and the distribution of the explanatory variables prior to the sample.

The constant γ captures the effect of all time-invariant determinants of the individuals ("inertia" for short).[2] The set of predetermined variables z_{it} includes the past realization of other endogenous variables x_{it} and previous outcomes $y_{i(t-1)}$, so that together they define the particular realization of the "state" for each individual i, that is $w_{it} = \left\{x_{it}, y_{i(t-1)}\right\}$. Thus, the estimates of β identify the effect of state dependence separately from inertia as z_{it} includes time-varying regressors that are only predetermined, that is not directly correlated with the current or future values of the error ε_{it} (although lagged values of errors ε_{it} might be correlated with z_{it}). The third ingredient is the following. As time elapses, individuals may take different decisions and become increasingly different. These decisions are summarized by $w_i^t = \{w_{i1}, \ldots, w_{it}\}$, which is the history of past $w_{it} = \left\{x_{it}, y_{i(t-1)}\right\}$. Finally, the last element of the model is η_i, an individual effect whose forecast is revised each period t as the information summarized by the history w_i^t accumulates. This value is not known to individuals and, hence, only its expectation enters the decision rule. In other words, y_{it} is not only affected by inertia (γ) and state dependence (β), but also by the learning effect $E\left(\eta_i \mid w_i^t\right)$ after controlling for individual heterogeneity.

Summing up, the model defines conditional probabilities for every possible sequence of realizations of state variables in order to deal with regressors that are endogenously predetermined but not exogenous. Then the panel data structure allows us to identify the effect of individual unobserved heterogeneity since at each time individuals may have different outcomes even if they have shared the same history of realizations of state variables.

Finally, the conditional distribution of the sequence of expectations $E\left(\eta_i \mid w_i^t\right)$ is left unrestricted, and so the process of updating expectations as information accumulates is not explicitly modeled. This is the only aspect that makes the model semi-parametric. While the assumption of normality of the distribution of errors is not essential, the assumption that the errors ε_{it} are not correlated over time is necessary for the estimation. Since errors are assumed to be normally distributed, conditional on the history of past decisions, the probability of a given outcome at time t for any given history w_i^t can be written as:

$$\text{Prob}\left(y_{it} = 1 \mid w_i^t\right) = \Phi\left[\frac{\gamma + \beta z_{it} + E\left(\eta_i \mid w_i^t\right)}{\sigma_t}\right] .$$

In terms of econometric implementation, since all our regressors are dichotomous variables, their support is a lattice with J points. The vector w_{it} has a support defined by $2J$ nodes $\{\phi_1, ..., \phi_{2J}\}$. The $t \times 1$–vector of regressors $z_i^t = \{z_{i1}, ..., z_{it}\}$ has a multinomial distribution and may take up to J^t different values. Similarly, the vector w_i^t is defined on $(2J)^t$ values, for $j = 1, ..., (2J)^t$. Given that the model has discrete support, any individual history can be summarized by a cluster of nodes representing the sequences for each individual in the sample. Thus, the conditional probability can be rewritten as:

$$p_{jt} = \text{Prob}\left(y_{it} = 1 \mid w_i^t = \phi_j^t\right) \equiv h_t\left(w_i^t = \phi_j^t\right) , \qquad j = 1, \ldots, (2J)^t .$$

In order to account for unobserved individual effects we compute the proportion of individuals with identical history up to time t that have outcome M at each time t. We then repeat this procedure for every available history in the data. For each history we

[2] The specification of Arellano and Carrasco (2003) is more general in the sense that it also includes a time-varying component common to all individuals γ_t. In our case, all our available demographics are time-invariant.

compute the percentage of individuals as this proportion provides a simple estimate of the unrestricted probability $\hat{p}_{tj}$ for each history present in the sample. Then, by taking first differences of the inverse of the equation above we get:

$$\sigma_t \Phi^{-1}\left[h_t\left(w_i^t\right)\right] - \sigma_{t-1}\Phi^{-1}\left[h_{t-1}\left(w_i^{t-1}\right)\right] - \beta\left(z_{it} - z_{i(t-1)}\right) = \xi_{it}\,,$$

and, by the law of iterated expectations, we have:

$$E\left[\xi_{it} \mid w_i^{t-1}\right] = E\left[E\left(\eta_i \mid w_i^t\right) - E\left(\eta_i \mid w_i^{t-1}\right)\middle| w_i^{t-1}\right] = 0\,.$$

This conditional moment condition serves as the basis of the GMM estimation of parameters β after normalizing $\sigma_1 = 1$. To identify the effect of inertia (time-invariant determinants) we use:

$$E\left[E(\eta_i \mid w_i^{t-1})\right] = E\left[\Phi^{-1}\left[h_t\left(w_i^{t-1}\right)\right] - \gamma - \beta z_{it}\right] = 0\,.$$

Arellano and Carrasco (2003) show that there is no efficiency loss in estimating these parameters by a two–step GMM method where in the first step the conditional probabilities p_{tj} are replaced by unrestricted estimates $\hat{p}_{tj}$. Then:

$$\hat{h}_t\left(w_i^t\right) = \sum_{j=1}^{(2J)^t} \mathbf{1}\left\{w_i^t = \phi_j^t\right\} \cdot \hat{p}_{tj}\,,$$

which is used to define the sample orthogonality conditions of the GMM estimator:[3]

$$\frac{1}{N}\sum_{i=1}^{N}\left\{\sigma_t\Phi^{-1}\left[\hat{h}_t\left(w_i^{t-1}\right)\right] - \gamma - \beta z_{it}\right\} = 0\,, \quad t = 2,\ldots,T\,,$$

and

$$\frac{1}{N}\sum_{i=1}^{N} d_{it}\left\{\sigma_t\Phi^{-1}\left[\hat{h}_t\left(w_i^t\right)\right] - \sigma_{t-1}\Phi^{-1}\left[\hat{h}_{t-1}\left(w_i^{t-1}\right)\right] - \beta\left(x_{it} - x_{i(t-1)}\right)\right\} = 0\,, \quad t = 3,\ldots,T\,,$$

and where d_{it} is a vector containing the indicators $\mathbf{1}\left\{w_i^t = \phi_j^t\right\}$ for $j = 1, ..., (2J)^{t-1}$.

With respect to the magnitude of the effects, the marginal effects associated with the transition among different states can be computed as follows. Arellano and Carrasco (2003) show that the probability of a given outcome when we compare two states $z_{it} = z^0$ and $z_{it} = z^1$ changes by the proportion:

$$\hat{\triangle}_t = \frac{1}{N}\sum_{i=1}^{N}\left\{\Phi\left(\hat{\sigma}_t^{-1}\hat{\beta}\left(z^1 - z_{it}\right) + \Phi^{-1}\left[\hat{h}_t\left(w_i^t\right)\right]\right) - \Phi\left(\hat{\sigma}_t^{-1}\hat{\beta}\left(z^0 - z_{it}\right) + \Phi^{-1}\left[\hat{h}_t\left(w_i^t\right)\right]\right)\right\}\,.$$

Since the evaluation depends on the history of past choices ω_i^t, these marginal effects are different for each "partial score" in the sample and for each individual

[3] We use the orthogonal deviations suggested by Arellano and Bover (1995) rather than first differences among past values of the state variables.

Finally, as in other settings where this method has been implemented, a reason why the Arellano-Carrasco is preferred in our setting is that alternative fixed-effect approaches such as Honoré and Lewbel (2002) and Honoré and Kyriazidou (2000) require the exogenous regressors to vary over time, something that does not occur in our data since we have standard demographic variables. The alternative in Honoré and Kyriazidou (2000) includes one lagged dependent variable but requires that the remaining explanatory variables should be strictly exogenous, thus excluding the possibility of a lagged dependent regressor (e.g., past successes, past productivities). Further, their estimator does not converge at the usual $\sqrt{n}$–rate. Honoré and Lewbel (2002) allow for additional predetermined variables but at the cost of requiring a continuous, strictly exogenous, explanatory variable that is independent of the individual effects. Fernandez-Val (2009) offers a useful characterization of the source of biases of fixed-effect estimators in non-linear panel data models.

References

Arellano, Manuel, and Olympia Bover. 1995. "Another Look at the Instrumental Variable Estimation of Error–Components Models." *Journal of Econometrics*, 68(1): 29–51.

Arellano, Manuel, and Raquel Carrasco. 2003. "Binary Choice Panel Data Models with Predetermined Variables." *Journal of Econometrics,* 115(1): 125–157.

Fernandez-Val, Ivan. 2009. "Fixed Eects Estimation of Structural Parameters and Marginal Eects in Panel Probit Models." *Journal of Econometrics*, 150(1): 71–85.

Honoré, Bo, and Arthur Lewbel. 2002. "Semiparametric Binary Choice Panel Data Models without Strictly Exogenous Regressors." *Econometrica,* 70(5): 2053–2063.

Honoré, Bo, and Ekaterini Kyriazidou. 2000. "Panel Data Discrete Choice Models with Lagged Dependent Variables." *Econometrica*, 68(4): 839–874.

```
 /*
  FILE: C:\FILES\EUGENIO\PAPERS\PENN12\*.DEF
  THESE PROGRAMS COMPUTE:
  + A STANDARD PROBIT MODEL WITH LAGGED ENDOGENOUS VARIABLES.
  + A GENERIC ARELLANO-CARRASCO DISCRETE CHOICE MODEL FOR THE CASE OF A
CONSTANT, TWO LAGGED ENDOGENOUS VARIABLES AND A GIVEN CHOICE OF SUBSAMPLE.
  + MARGINAL EFFECTS IN EACH CASE.
*/

new;
library maxlik;
time0=time;

output file =c:\gausswin\Penn12\*.OUT reset;

load exg = exogenas;
load dnames;

/*
** dnames="CONSTANT"~ /* 01 Constant */
**        "NAME OF VARIABLE #1 HERE"~ /* 02 Predetermined variable 1*/
**        "NAME OF VARIABLE #2 HERE"~ /* 03 Predetermined variable 2*/
**        "NAME OF VARIABLE #3 HERE"~ /* 04 Exogenous variable 1*/
**           …
**        "NAME OF VARIABLE #32 HERE"~ /* 33 Exogenous variable 30*/
**        "RHO     "; /* 34 Rho */
** dnames=dnames';
*/

/* ======================== PROCEDURES ======================== */

/* Descriptive Statistics */
proc(0) = Dsa(dsanames,dsaxvar);
local aa,bb,cc,dd,ee,aux,mask,fmt;
   aa = meanc(dsaxvar);
   bb = stdc(dsaxvar);
   cc = minc(dsaxvar);
   dd = maxc(dsaxvar);
   ee = rows(dsaxvar)*ones(rows(dsanames),1);
   aux = dsanames~aa~bb~cc~dd~ee;
   print "-----------------------------------------------------------------------";
   print "Variable         MEAN         STDC          MIN          MAX
Obs.";
   print "-----------------------------------------------------------------------";
   mask = {0 1 1 1 1 1};
   fmt = { "-*.*s "  8 8,
           "*.*lf " 12 4,
           "*.*lf " 12 4,
           "*.*lf " 12 4,
           "*.*lf " 12 4,
           "*.*lf " 8 0};
   call printfm (aux, mask, fmt);
   print "-----------------------------------------------------------------------";
   print "\r\l ";
endp;


tt = 1; r = 1; t = 1;
/* DELTA METHOD */
proc (2) = DMethod(knames, b, cov, f);
local s, p, mask, fmt, oldtrap;
local f:proc, jac, vr;
    r = f (b);
    jac = gradp (&f, b);
    vr = jac*cov*jac';
    oldtrap = trapchk (1);
    trap 1, 1;
```

```
    print "-------------------------------------------------------------";
    print "Variable       Coeff.    Std. Err.      t-Stat.    Prob.";
    print "-------------------------------------------------------------";
    s = Sqrt (Diag (vr)); t = abs(r./s); p = 2*Cdfnc (Abs (t));
    mask = {0 1 1 1 1};
    fmt = { "-*.*s "   8 8,
            "*.*lf " 12 4,
            "*.*lf " 12 4,
            "*.*lf " 12 2,
            "*.*lf "  8 3};
    call printfm (knames~r~s~t~p, mask, fmt);
    print "\r\l-------------------------------------------------------------";
    trap oldtrap, 1;
    tt = t;
    retp (r, vr);
endp;

xxx = 1;
/* Marginal Effects for Probit at Mean */
proc(1) = mef (b);
local f;
   f = b*pdfn(meanc(xxx)'b);
   retp (f);
endp;

w = 1;
/* Probit Likelihood Function */
proc LProbit(beta, x);
local k, y, xb;
    k = cols(x);
    xb =  beta[1]+x[.,2:k]*beta[2:k];
    y = 2*x[.,1]-1;
    retp(w.*ln(cdfn(y.*xb)));
endp;

/* Probit Gradient */
proc GProbit(beta, x);
local k, s, y, l, xb;
    k = cols(x);
    xb = beta[1] + x[.,2:k]*beta[2:k];
    y = 2*x[.,1]-1;
    l = pdfn(y.*xb)./cdfn(y.*xb);
    retp(w.*(y.*l).*(ones(rows(x), 1)~x[.,2:k]));
endp;

/* Probit Hessian Expectation */
proc HProbit(beta, x);
local k, s, y, l, xb;
    k = cols(x);
    xb = beta[1]+x[.,2:k]*beta[2:k];
    l = sqrt(w.*(pdfn(xb)^2)./(cdfn(xb).*(1-cdfn(xb))));
    y = l.*(ones(rows(x), 1)~x[.,2:k]);
    retp(-y'y);
endp;


/* Computing Orthogonal Deviations */
proc adev1(x);
   local xdev,xm,no,t,a,l;
   t=rows(x); l=ones(t-1,1);
   a=ones(1,cols(x)); no=seqa(t-1,-1,t-1)*a;
   if t>3;
   xm=recserar(-x[1:t-1,.],l'x[2:t,.],a);
   elseif t==3;
   xm=(x[2,.]+x[3,.])|x[3,.];
   elseif t==2;
   xm=x[2,.];
   endif;
```

```
   xdev=x[1:t-1,.]-xm./no; xdev=sqrt(no./(no+1)).*xdev;
retp(xdev);endp;


/* =================== END OF PROCEDURES ==================== */


print "\r\l\l _____________ TABLE 7&8: ARELLANO AND CARRASCO _____________ ";

sav = exg[.,35:37];
cutoffs = 0;
i1 = dummy(sav[.,1],cutoffs);
i2 = dummy(sav[.,2],cutoffs);
i3 = dummy(sav[.,3],cutoffs);
savi = i1[.,2]~i2[.,2]~i3[.,2];
exg[.,26:28] = usgi;

bkpexg = exg~savi;
case = 0;
do while case .<= 32;
   if case .== 1;
      exg = selif(bkpexg[.,.], bkpexg[.,1] .== 1);
      print "\r\l CASE 1: NAME OF VARIABLE1 = 1 ";
   elseif case .== 2;
      exg = selif(bkpexg[.,.], 1-bkpexg[.,1]-bkpexg[.,2] .== 1);
      print "\r\l CASE 2: NAME OF VARIABLE2 = 1 ";
   elseif case .== 3;
      exg = selif(bkpexg[.,.], bkpexg[.,2] .== 1);
      print "\r\l CASE 3: NAME OF VARIABLE3 = 1 ";

/* DO FOR AS MANY VARIABLES AS DESIRED */

   else;
      exg = bkpexg;
      print "\r\l CASE 0: COMPLETE SAMPLE ";
   endif;

/* EXAMPLES FOR DIFFERENT SETS OF VARIABLES */

savi = exg[.,cols(bkpexg)-2:cols(bkpexg)];
exg = exg[.,1:cols(bkpexg)-3];

datos1 = exg[.,29 20:22 26:28];
datos2 = exg[.,32 23:25 29:31];
xm = exg[.,1:19];

dsai = 1;
xxx1 = (exg[.,20]|exg[.,21]|exg[.,22])~
       (exg[.,26]|exg[.,27]|exg[.,28])~
       (exg[.,29]|exg[.,30]|exg[.,31]);
xxx2 = (exg[.,23]|exg[.,24]|exg[.,25])~
       (exg[.,29]|exg[.,30]|exg[.,31])~
       (exg[.,32]|exg[.,33]|exg[.,34]);

qexg = { 3,7,8,10,11,12,13,14,15,16,17,19 };
qq = rows(qexg);
do while qq .<= rows(qexg);

exg = xm[.,1:qexg[qq]];
cexg = cols(exg);

table = 7;
do while table .<= 8;

/* Extract matrix of predetermined and exogenous variables */
if table .== 7;
   datos = datos1[.,5:7 1:4];
   xxx = xxx1;
```

```
else;
   datos = datos2[.,5:7 1:4];
   xxx = xxx2;
endif;
cdts = cols(datos)-1;

/* Number of support points */
nc = 0;
l = 1;
do while l .<= cdts;
   nc = nc+2^l;
   l = l+1;
endo;

/* Routine to compute the number of observations per cell */
datost = datos[.,1 4 2 5 3 6];

dt = zeros(rows(datost),nc);
dt[.,1] = datost[.,1];
dt[.,2] = 1-datost[.,1];

ac=0;
l = 2;
do while l .<= cols(datost);
   bc = ac;
   ac = ac+2^(l-1);
   dt[.,ac+1:ac+2^(l-1)] = dt[.,bc+1:bc+2^(l-1)].*(datost[.,l]);
   dt[.,ac+2^(l-1)+1:ac+2^l] = dt[.,bc+1:bc+2^(l-1)].*(1-datost[.,l]);
   l = l+1;
endo;

/* Matrix D selects relevant columns of DT */
q = {4,16,64};
d = zeros(rows(dt),sumc(q));
d[.,1:4] = dt[.,3:6];
d[.,5:20] = dt[.,15:30];
d[.,21:84] = dt[.,63:126];
clear dt;

/* This routine computes the number of individual observations */
let rg = {};
lk = 1;
s = sumc(d);
i = 1;
do while lk .<= 3;
   j = 1;
   v1 = zeros(q[lk],1);
   do while i .<= sumc(q[1:lk]);
      if s[i] .> 4; v1[j] = i;
         j = j+1;endif;
      i=i+1;
   endo;
   v1 = selif(v1, v1[.,1] .> 0);
   df=submat(d,0,v1);
   g1=sumc(df');
   g1=selif(g1, g1[.,1] .> 0);
   rg=rg~rows(g1);
   lk = lk+1;
endo;
clear i,j,lk,v1,df,g1;

/* Relative Frequency, P */
s=sumc(d);
num=zeros(rows(d),cols(d));
num[.,1:4] = datos[.,5].*d[.,1:4];
num[.,5:20] = datos[.,6].*d[.,5:20];
num[.,21:84] = datos[.,7].*d[.,21:84];
n=sumc(num);
```

```
clear num;
p=zeros(rows(s),1);
j=1;
do while j .<= rows(s);
   if s[j] .== 0;
      p[j] = 0;
   else;
      p[j] = n[j]/s[j];
   endif;
   j=j+1;
endo;

/* Matrix H */
h=zeros(rows(d),3);
j = 1;
do while j .<= 84;
   if j .<= 4;
      h[.,1] = h[.,1]+p[j]*d[.,j];
   else;
      if j .<= 20;
         h[.,2] = h[.,2]+p[j]*d[.,j];
      else;
         h[.,3] = h[.,3]+p[j]*d[.,j];
      endif;
   endif;
   j=j+1;
endo;

/* Matrix V */
v = zeros(rows(d),3);
j=1;
do while j .<= 84;
   if j .<= 4;
      if s[j] .== 0;
         v[.,1] = v[.,1];
      else;
         v[.,1] = v[.,1]+(2*s[j])^(-1)*d[.,j];
      endif;
   else;
      if j .<= 20;
         if s[j] == 0;
            v[.,2] = v[.,2];
         else;
            v[.,2] = v[.,2]+(2*s[j])^(-1)*d[.,j];
         endif;
      else;
         if s[j] == 0;
            v[.,3] = v[.,3];
         else;
            v[.,3] = v[.,3]+(2*s[j])^(-1)*d[.,j];
         endif;
      endif;
   endif;
   j=j+1;
endo;

/* Matrix W */
w = ln((h+v)./(1-h+v));

/* Orthogonal deviations of predetermined variables */
wst = adev1(w');
wst = wst';

k12 = datos[.,1:3];
k12st = adev1(k12');
k12st = k12st';

if table .== 6;
```

```
   ytm1 = datos[.,4:6];
else;
   ytm1 = savi;
endif;
ytm1st = adev1(ytm1');
ytm1st = ytm1st';

/* Sum of columns of K12, YTM1, W */
c3 = zeros(rows(d),cols(d));
c5 = zeros(rows(d),cols(d));
b2 = zeros(rows(d),cols(d));
j = 1;
do while j .<= 84;
   if j .<= 4;
      c3[.,j] = d[.,j].*k12[.,1];
      c5[.,j] = d[.,j].*ytm1[.,1];
      b2[.,j] = d[.,j].*w[.,1];
   else;
      if j .<= 20;
         c3[.,j] = d[.,j].*k12[.,2];
         c5[.,j] = d[.,j].*ytm1[.,2];
         b2[.,j] = d[.,j].*w[.,2];
      else;
         c3[.,j] = d[.,j].*k12[.,3];
         c5[.,j] = d[.,j].*ytm1[.,3];
         b2[.,j] = d[.,j].*w[.,3];
      endif;
   endif;
   j=j+1;
endo;


f1 = zeros(4,1);
f2 = zeros(16,1);
f3 = zeros(64,1);
j = 1;jj = 1;jjj = 1;
i = 1;
do while i .<= 84;
   if i .<= 4;
      if s[i] .> 4;
         f1[j] = i;
         j=j+1;
      else;endif;
   else;
      if i .<= 20;
         if s[i] .> 4;
            f2[jj] = i;
            jj=jj+1;
         else;endif;
      else;
         if s[i] .> 4;
            f3[jjj] = i;
            jjj=jjj+1;
         else;endif;
      endif;
   endif;
   i=i+1;
endo;

f1=selif(f1, f1[.,1] .> 0);
f2=selif(f2, f2[.,1] .> 0);
f3=selif(f3, f3[.,1] .> 0);
c31=submat(c3,0,f1);
c31=sumc(c31');
c32=submat(c3,0,f2);
c32=sumc(c32');
c33=submat(c3,0,f3);
c33=sumc(c33');
```

```
c3=(sumc(c31)|sumc(c32)|sumc(c33));
clear c31,c32,c33;

c51=submat(c5,0,f1);
c51=sumc(c51');
c52=submat(c5,0,f2);
c52=sumc(c52');
c53=submat(c5,0,f3);
c53=sumc(c53');
c5=(sumc(c51)|sumc(c52)|sumc(c53));
clear c51,c52,c53;

b21=submat(b2,0,f1);
b21=sumc(b21');
b22=submat(b2,0,f2);
b22=sumc(b22');
b23=submat(b2,0,f3);
b23=sumc(b23');
b2=(sumc(b21)|sumc(b22)|sumc(b23));
clear b21,b22,b23;

let cxx = {};
i = 1;
do while i .<= cexg;
   j = 1;
   do while j .<= 84;
      cx = zeros(rows(d),cols(d));
      cx[.,j] = d[.,j].*exg[.,i];
      j = j+1;
   endo;
   cx1 = sumc((submat(cx,0,f1))');
   cx2 = sumc((submat(cx,0,f2))');
   cx3 = sumc((submat(cx,0,f3))');
   cxx = cxx~(sumc(cx1)|sumc(cx2)|sumc(cx3));
   i = i+1;
endo;
clear cx1,cx2,cx3,cx;

/* Computing matrices C1, C22, B1 */
c1 = zeros(rows(d),4+16);
c22 = zeros(rows(d),4+16);
b1 = zeros(rows(d),4+16);
j=1;
do while j .<= 20;
   if j .<= 4;
      c1[.,j] = d[.,j].*k12st[.,1];
      c22[.,j] = d[.,j].*ytm1st[.,1];
      b1[.,j] = d[.,j].*wst[.,1];
   else;
      c1[.,j] = d[.,j].*k12st[.,2];
      c22[.,j] = d[.,j].*ytm1st[.,2];
      b1[.,j] = d[.,j].*wst[.,2];
   endif;
   j=j+1;
endo;

/* Sum of columns of C1, C22 and B1 */
sv=s[1:(4+16),.];
i = 1;
j = 1;
v1 = zeros(4+16,1);
do while i .<= 4+16;
   if sv[i] .> 4;
      v1[j] = i;
      j=j+1;
   else;endif;
   i=i+1;
endo;
```

```
v1 = selif(v1, v1[.,1] .> 0);
cv1=submat(c1,0,v1);
cv1=sumc(cv1);
bv1=submat(b1,0,v1);
bv1=sumc(bv1);
cv22=submat(c22,0,v1);
cv22=sumc(cv22);

/* Definition of matrix C and vector B */
ca = cv1'~c3';
ccc = rg;
cc = zeros(1,rows(cv1))~ccc;
cx = zeros(cexg,rows(cv1))~cxx';
cd = cv22'~c5';
c = ca|cd|cc|cx;
c = ca|cd|cc;
b = bv1'~b2';
clear ca,ccc,cc,bv1,b2;

/* Computing matrices A1, A, and its inverse Ai*/
a1 = zeros(20,20);
a1[1:4,1:4] = d[.,1:4]'*d[.,1:4];
a1[5:20,5:20] = d[.,5:20]'*d[.,5:20];
i = 1;
j = 1;
v1 = zeros(20,1);
do while i .<= 20;
   if sv[i] .> 4;
      v1[j] = i;
      j = j+1;
   else;endif;
   i=i+1;
endo;
v1 = selif(v1, v1[.,1] .> 0);
a1 = submat(a1,v1,v1);

af = rg';
af = diagrv(eye(3),af);
a = (a1~zeros(rows(a1),3))|(zeros(3,rows(a1))~af);

ai = inv(a);
clear a1,a;

/* Consistent estimator. First stage */
/* -------------------------------- */
k=pinv(c*ai*c')*(c*ai*b');
k=inv(c*ai*c')*(c*ai*b');
_b=b';

/* Covariance of the joint optimal estimator */
e1 = wst-k[1]*k12st[.,1:2]-k[2]*ytm1st[.,1:2];
@ e2 = w-k[1]*k12[.,1:3]-k[2]*ytm1[.,1:3]-k[3]-exg[.,1:cexg]*k[4:cexg+3]; @
e2 = w-k[1]*k12[.,1:3]-k[2]*ytm1[.,1:3]-k[3];

m0 = zeros(rows(d),20);
m02 = zeros(rows(d),84);
j=1;
do while j .<= 84;
   if j .<= 4;
      m0[.,j] = d[.,j].*e1[.,1];
      m02[.,j] = d[.,j].*e2[.,1];
   elseif j .> 16;
      m02[.,j] = d[.,j].*e2[.,3];
   else;
      m0[.,j] = d[.,j].*e1[.,2];
      m02[.,j] = d[.,j].*e2[.,2];
   endif;
```

```
   j = j+1;
endo;
m0 = submat(m0,0,v1);
m021 = submat(m02,0,f1);
m021 = sumc(m021');
m022 = submat(m02,0,f2);
m022 = sumc(m022');
m023 = submat(m02,0,f3);
m023 = sumc(m023');
m02 = m021~m022~m023;
m1 = m0~m02;
clear m0,m02,m021,m022,m023;
fi1 = m1'*m1;

e3 = zeros(rows(datos),84);
e3[.,1:4] = datos[.,5] - p[1:4]';
e3[.,5:20] = datos[.,6] - p[5:20]';
e3[.,21:84] = datos[.,7] - p[21:84]';

m2 = d.*e3;

i = 1;
j = 1;
v2 = zeros(rows(s),1);
do while i .<= rows(s);
   if s[i] .> 4;
      v2[j] = i;
      j = j+1;
   else;endif;
   i=i+1;
endo;
v2 = selif(v2, v2[.,1] .> 0); /* More than 4 observations per cell indicator
*/
m2 = submat(m2,0,v2);
fi2 = m2'*m2;
fi3 = m1'*m2;
fi4 = fi3';
fi = (fi1~fi3)|(fi4~fi2);
clear fi1,fi2,fi3,fi4,m1,m2;

/* ---> Derivatives of fi2 w.r.t. p */
s2 = submat(s,v2,0);
cd2 = diagrv(eye(rows(s2)),s2);
clear s2;

/* ---> Derivatives of fi1(1-20) w.r.t. p1-p4 */
cpt1 = zeros(4,20);
f = 1;
j = 1;
do while j .<= 4;
   cp1 = zeros(rows(d),20);
   ac = 0;
   l = 1;
   do while l .<= 4;
      ac = ac+1;
      rt1 = sqrt(2/3);
      cp1[.,ac] = (d[.,ac].*d[.,j])*rt1;
      cp1[.,ac] = cp1[.,ac].*(1+2*v[.,1])
                  ./((h[.,1]+v[.,1]).*(1-h[.,1]+v[.,1]));
      l = l+1;
   endo;
   cp1 = sumc(cp1);
   cpt1[.,f] = cp1[1:4];
   f = f+1;
   j = j+1;
endo;
clear cp1;
```

```
/* ---> Derivatives of fi1(1-20) w.r.t. p5-p20 */
cpt2 = zeros(16,20);
f = 1;
j = 5;
do while j .<= 20;
   cp2 = zeros(rows(d),20);
   ac = 0;
   l = 1;
   do while l .<= 20;
      ac = ac+1;
      rt2 = sqrt(1/2);
      if ac .<= 4;
         cp2[.,ac] = (d[.,ac].*d[.,j])*(-rt1)*(1/2);
         cp2[.,ac] = cp2[.,ac].*(1+2*v[.,2])
                     ./((h[.,2]+v[.,2]).*(1-h[.,2]+v[.,2]));
      else;
         cp2[.,ac] = (d[.,ac].*d[.,j])*rt2;
         cp2[.,ac] = cp2[.,ac].*(1+2*v[.,2])
                     ./((h[.,2]+v[.,2]).*(1-h[.,2]+v[.,2]));
      endif;
      l = l+1;
   endo;
   cp2 = sumc(cp2);
   cpt2[f,.] = cp2';
   f=f+1;
   j=j+1;
endo;
clear cp2;

/* ---> Derivatives of fi1(1-20) w.r.t. p21-p84 */
cpt3 = zeros(84,20);
f = 1;
j = 21;
do while j .<= 84;
   cp3 = zeros(rows(d),20);
   ac = 0;
   l = 1;
   do while l .<= 20;
      ac = ac+1;
      rt2 = sqrt(1/2);
      if ac .<= 4;
         cp3[.,ac] = (d[.,ac].*d[.,j])*(-rt1)*(1/2);
         cp3[.,ac] = cp3[.,ac].*(1+2*v[.,3])
                     ./((h[.,3]+v[.,3]).*(1-h[.,3]+v[.,3]));
      else;
         cp3[.,ac] = (d[.,ac].*d[.,j])*rt2;
         cp3[.,ac] = cp3[.,ac].*(1+2*v[.,3])
                     ./((h[.,3]+v[.,3]).*(1-h[.,3]+v[.,3]));
      endif;
      l = l+1;
   endo;
   cp3 = sumc(cp3);
   cpt3[f,.] = cp3';
   f=f+1;
   j=j+1;
endo;
clear cp3;

/* ---> Derivatives of fi1(1-20) w.r.t. p1-p84 */
c61 = cpt1|cpt2|cpt3;
clear cpt1,cpt2,cpt3;

/* ---> Derivatives of fi1(85-87) w.r.t. p1-p84 */
cpt11=zeros(84,3);
f = 1;
r = 0;
j = 1;
do while j .<= 84;
```

```
   if j .<= 4;
      r = 1;
   elseif j .> 4 and j.<= 20;
      r = 2;
   else;
      r = 3;
   endif;
   cp11 = zeros(rows(d),3);
   cp11[.,r] = d[.,j].*(1+2*v[.,r])
               ./((h[.,r]+v[.,r]).*(1-h[.,r]+v[.,r]));
   cp11 = sumc(cp11);
   cpt11[f,.] = cp11';
   f = f+1;
   j = j+1;
endo;
clear cp11;

c61 = submat(c61,v2,v1);
c62 = submat(cpt11,v2,0);
c6 = c61~c62;
clear c61,c62;

/* Covariance matrix for the non-optimal non-joint estimator */
vf1 = (eye(rows(c6'))~(c6'*pinv(cd2)))*fi*(eye(rows(c6'))|(pinv(cd2))*c6);
vano = pinv((-c)*ai*(-c'))*(-c)*ai*vf1*ai*(-c')*pinv((-c)*ai*(-c'));

/* Covariance matrix of the optimal non-joint estimator */
/* vao = inv((-c)*inv(vf1)*(-c')); */

if table .== 7;
   gmmnames = dnames[1 7:cexg+6 2:3];
   gmmnames = dnames[1 2:3];
print; "GMM Estimation with predetermined variables and orthogonal
deviations";
print; "Endogenous Variable: SCORE";
endif;
colk = rows(gmmnames);

bb = k[3:colk 1:2,.];
varb = sqrt(diag(vano));
varb = varb[3:colk 1:2];
tb = abs(bb./sqrt(diag(varb)));
aux = gmmnames~bb~tb;
auxn = aux[1:colk,.];
print "\r\l     VAR       EST.       t-STAT";
print "-------------------------------------";
mask={ 0 1 1 };
fmt={ "*.*s " 8 8, "*.*lf " 10 4, "*.*lf " 10 2 };
call printfm(auxn,mask,fmt);
ort = round(4032*rows(xxx)/11850)*(_b-c'k)'*pinv(vf1)*(_b-c'k);
ort = round(4032*rows(xxx)/11850)*(_b-c'k)'*ai*(_b-c'k);
print "Overidentifying Restriction Test = " ort;
print "p-value ....................... = " cdfchic(ort,(rows(vf1)-rows(k)));
print "Number of Moments ............. = " rows(vf1);
print "DG = Degrees of Freedom.........= " (rows(vf1)-rows(k));
print "Chi-Sq(0.90,DG) = " cdfchii(0.90,(rows(vf1)-rows(k)));
print "Chi-Sq(0.95,DG) = " cdfchii(0.95,(rows(vf1)-rows(k)));
print "Chi-Sq(0.99,DG) = " cdfchii(0.99,(rows(vf1)-rows(k)));

if case .== 0;
fromm = "From ("|"From ("|"From ("|"From ("|"From ("|"From ("|"From
(";
tto = ") to ("|") to ("|") to ("|") to ("|") to ("|") to ("|") to ("|") to (";
prntss = ") "|") "|") "|") "|") "|") "|") "|") ";
deltax =  {0 0, 0 0, 0 1, 0 1, 1 0, 1 0, 1 1, 1 1};
deltaxx = {0 1, 1 0, 0 0, 1 1, 1 1, 0 0, 1 0, 0 1};
xit = k12~ytm1;
inphi = ln((h+0.0000001)./(1.0000001-h[.,.]));
```

```
let acmef = {};
ixx = 1;
do while ixx .<= 8;
   bdxx = ((deltaxx[ixx,.]-xit[.,1 4])*bb[2:3])~
          ((deltaxx[ixx,.]-xit[.,2 5])*bb[2:3])~
          ((deltaxx[ixx,.]-xit[.,3 6])*bb[2:3]);
   bdxx = bdxx+inphi;
   bdxx = exp(bdxx)./(1+exp(bdxx));
   bdx = ((deltax[ixx,.]-xit[.,1 4])*bb[2:3])~
         ((deltax[ixx,.]-xit[.,2 5])*bb[2:3])~
         ((deltax[ixx,.]-xit[.,3 6])*bb[2:3]);
   bdx = bdx+inphi;
   bdx = exp(bdx)./(1+exp(bdx));
   ds = meanc(bdxx-bdx);
   ds = (ds|meanc(ds))';
   acmef = acmef|ds;
   ixx = ixx+1;
endo;
acmef = acmef*100;
aux = fromm~deltax~tto~deltaxx~prntss~acmef;
print "\r\l Marginal Effects Team1:   Round2  Round3 Round4  Round 5";
print "-----------------------------------------------------------------------";
mask = { 0 1 1 0 1 1 0 1 1 1 1};
fmt={ "*.*s " 6 6, "*.*lf " 1 0, "*.*lf " 1 0,
      "*.*s " 6 6, "*.*lf " 1 0, "*.*lf " 1 0,
      "*.*s " 2 2, "*.*lf " 8 2, "*.*lf " 10 2, "*.*lf " 10 2, "*.*lf " 10 2
};
call printfm(aux,mask,fmt);
else;endif;

w = ones(rows(xxx),1);
yxxx = xxx[.,1];
zxxx = w~xxx[.,2:3];
pbeta1 = inv(zxxx'zxxx)*(zxxx'yxxx);
MaxSet;
_max_ParNames = gmmnames;
_max_GradProc = &GProbit;
_max_HessProc = &HProbit;
_max_GradTol = 1e-3;
_max_Options = { NEWTON ONE SCREEN HETCON };
{ b,f,g,cov,ret } = maxlik(xxx,0,&LProbit,pbeta1);
{ b,f,g,cov,ret } = maxprt(b,f,g,cov,ret);
print "-lnL = " -rows(xxx)*f;
print "Actual Sample = " round(4032*rows(xxx)/11850);

if case .== 0;
print "\r\l\l _________ ESTIMATES OF MARGINAL EFFECTS AT MEAN __________ ";
call DMethod(gmmnames,b,cov,&mef);

if dsai .< 100;
   print "\r\l ";
   dsa(dnames[7:25],exg[.,1:19]);
else;endif;
dsai = -1*dsai;
else;endif;


table = table+1;
endo;

qq = qq+1;
endo;

case = case+1;
endo;

print "\r\l\l Computation Time:";
dtime=time-time0;
```

```
dtime=3600*dtime[1]+60*dtime[2]+dtime[3]+dtime[4]/100;dtime;
output off;
end;
```