The Stata Journal (2003)
3, Number 3, pp. 278–294

Multivariate probit regression using simulated maximum likelihood

Lorenzo Cappellari
Università del Piemonte Orientale and
University of Essex

Stephen P. Jenkins
University of Essex

Abstract. We discuss the application of the GHK simulation method for maximum likelihood estimation of the multivariate probit regression model and describe and illustrate a Stata program `mvprobit` for this purpose.

Keywords: st0045, maximum likelihood estimation, multivariate probit regression model, GHK, mvprobit, mvppred

1 Introduction

Evaluation of probit-model likelihood functions requires calculation of normal probability distribution functions. Algorithms exist that provide accurate calculations for univariate and bivariate normal pdfs, and these are used by functions incorporated in many software packages. (See, for example, `norm` and `binorm` in Stata 8.) Accurate functions for the evaluation of trivariate and higher-dimensional normal distributions do not exist in Stata, however. Moreover, in these multivariate normal cases, computations based on standard linear numerical approximations, such as those based on the Newton–Raphson method, are relatively inefficient and may provide poor approximations (Hajivassiliou and Ruud 1994). Researchers have turned instead to simulation-based methods that have much better properties. See Stern (1997) and Gourieroux and Monfont (1996) for extensive discussion of simulation estimation techniques and their applications in a number of contexts. Greene (2003, 931–933) provides a brief textbook exposition.

In this article, we discuss the application of a simulation method to maximum likelihood estimation of the multivariate probit regression model and describe a Stata program `mvprobit` for this purpose. In section 2, we describe the model and review the principles underlying estimation by simulated maximum likelihood using the so-called GHK simulator. Our `mvprobit` program is explained in section 3, and it is illustrated in section 4. Section 5 discusses issues such as the choice of number of replications. Antoine Terracol's program `triprobit`, available from the SSC-IDEAS archive, fits trivariate probit regression models using simulated likelihood estimation. In `mvprobit`, written independently, a more general algorithm is used, the number of model equations is unlimited in principle, there are more options, and there is also a companion post-estimation prediction program (`mvppred`).

2 The model and the method of simulated maximum likelihood

Consider the M-equation multivariate probit model:

$$y_{im}{}^{*} = \beta_m{}'X_{im} + \epsilon_{im},\ m = 1, ..., M$$

$$y_{im} = 1 \text{ if } y_{im}{}^{*} > 0 \text{ and } 0 \text{ otherwise}$$

ϵ_{im}, $m = 1, \ldots, M$ are error terms distributed as multivariate normal, each with a mean of zero, and variance–covariance matrix V, where V has values of 1 on the leading diagonal and correlations $\rho_{jk} = \rho_{kj}$ as off-diagonal elements.

The model has a structure similar to that of a seemingly unrelated regression (SUR) model, except that the dependent variables are binary indicators. As for the SUR case (`sureg`), the equations need not include exactly the same set of explanatory variables. The familiar univariate and bivariate probit models correspond to the cases when $M = 1$ and 2 (estimable using `probit` and `biprobit`).

The y_{im} might represent outcomes for M different choices at the same point in time, for example, whether an individual owns each of M different consumer durables. Alternatively, the y_{im} might represent M outcomes on the same choice at M different points in time. That is, the multivariate probit model can be used to fit a univariate probit model for panel (cross-sectional time-series) data allowing for a free correlation structure over time.

To facilitate exposition of the method of estimation by simulated maximum likelihood, let us focus on the case in which $M = 3$. In the trivariate probit case, the log-likelihood function for a sample of N independent observations is given by

$$L = \sum_{i=1}^{N} w_i \log\Phi_3(\mu_i; \Omega)$$

where w_i is an optional weight for observation $i = 1, \ldots, N$, and $\Phi_3(.)$ is the trivariate standard normal distribution with arguments μ_i and Ω, where

$$\mu_i = (K_{i1}\beta_1{}'X_{i1}, K_{i2}\beta_2{}'X_{i2}, K_{i3}\beta_3{}'X_{i3})$$

with $K_{ik} = 2y_{ik} - 1$, for each i, $k = 1, \ldots, 3$. Matrix Ω has constituent elements Ω_{jk}, where

$$\Omega_{jj} = 1 \text{ for } j = 1, \ldots, 3$$
$$\Omega_{21} = \Omega_{12} = K_{i1}K_{i2}\rho_{21}$$
$$\Omega_{31} = \Omega_{13} = K_{i3}K_{i1}\rho_{31}$$
$$\Omega_{32} = \Omega_{23} = K_{i3}K_{i2}\rho_{32}$$

Clearly the log-likelihood function depends on the trivariate standard normal distribution function $\Phi_3(.)$.

The most popular simulation method for evaluating multivariate normal distribution functions is the Geweke–Hajivassiliou–Keane (GHK) smooth recursive conditioning simulator. See Börsch-Supan et al. (1992), Börsch-Supan and Hajivassiliou (1993), Keane (1994), and Hajivassiliou and Ruud (1994). Greene's textbook (2003, 931–933) provides a useful brief review. The GHK simulator exploits the fact that a multivariate normal distribution function can be expressed as the product of sequentially conditioned univariate normal distribution functions, which can be easily and accurately evaluated.

In the trivariate case, there are eight joint probabilities corresponding to the eight possible combinations of successes ($y_{im} = 1$) and failures ($y_{im} = 0$). Let us focus on the probability that every outcome is a success. This is given by

$$\begin{aligned}
&\Pr(y_1 = 1, y_2 = 1, y_3 = 1) \\
&\quad = \Pr(\epsilon_1 \leq {\beta_1}' X_1, \epsilon_2 \leq {\beta_2}' X_2, \epsilon_3 \leq {\beta_3}' X_3) \\
&\quad = \Pr(\epsilon_3 \leq {\beta_3}' X_3 \mid \epsilon_2 < {\beta_2}' X_2, \epsilon_1 < {\beta_1}' X_1) \times \Pr(\epsilon_2 < {\beta_2}' X_2 \mid \epsilon_1 < {\beta_1}' X_1) \\
&\qquad \times \Pr(\epsilon_1 < {\beta_1}' X_1)
\end{aligned}$$

where observation subscript i has been dropped for convenience. This expression involves conditioning upon unobservable variables (that are correlated with each other). The expressions for each of the joint probabilities of each of the seven other outcome combinations involve similar conditioning. However, if a good approximation for these conditional distributions can be found, then the likelihood function only requires evaluation of univariate integrals. How may the approximations be derived?

Consider the Cholesky decomposition of the covariance matrix for the errors

$$E(\epsilon\epsilon') \equiv V = Cee'C$$

where C is the lower triangular Cholesky matrix corresponding to V and $e \sim \Phi_3(0, I_3)$, where I_3 is the 3×3 identity matrix (i.e., the e are three uncorrelated standard normal variates). It follows that

$$\epsilon_1 = C_{11}e_1$$
$$\epsilon_2 = C_{21}e_1 + C_{22}e_2$$
$$\epsilon_3 = C_{31}e_1 + C_{32}e_2 + C_{33}e_3$$

and C_{jk} is the jkth element of matrix C. We can, therefore, rewrite the decomposition of the trivariate normal probability of three successes as

$$\begin{aligned}
&\Pr(\epsilon_1 \leq {\beta_1}'X_1, \epsilon_2 \leq {\beta_2}'X_2, \epsilon_3 \leq {\beta_3}'X_3) \\
&\quad = \Pr[e_3 \leq ({\beta_3}'X_3 - C_{32}e_2 - C_{31}e_1)/C_{33} \mid e_2 \leq ({\beta_2}'X_2 - C_{21}e_1)/C_{22}, e_1 \\
&\qquad \leq {\beta_1}'X_1/C_{11}] \\
&\quad \times \Pr[e_2 \leq ({\beta_2}'X_2 - C_{21}e_1)/C_{22} \mid e_1 \leq {\beta_1}'X_1/C_{11}] \times \Pr[e_1 \leq {\beta_1}'X_1/C_{11}]
\end{aligned}$$

The standard normal variates, e, that now appear in the decomposition are uncorrelated with each other (by construction). The first two conditional probabilities can be further rewritten as unconditional probabilities defined in terms of truncated standard normal variates. That is,

$$\begin{aligned}
&\Pr(\epsilon_1 \leq {\beta_1}'X_1, \epsilon_2 \leq {\beta_2}'X_2, \epsilon_3 \leq {\beta_3}'X_3) \\
&\quad = \Pr[\epsilon_3 \leq ({\beta_3}'X_3 - C_{32}{e_2}^* - C_{31}{e_1}^*)/C_{33}] \\
&\qquad \times \Pr[\epsilon_2 \leq ({\beta_2}'X_2 - C_{21}{e_1}^*)/C_{22}] \times \Pr[\epsilon_1 \leq {\beta_1}'X_1/C_{11}] \\
&\quad = Q_3 \times Q_2 \times Q_1, \text{ say,}
\end{aligned}$$

where ${e_1}^*$ and ${e_2}^*$ are truncated univariate standard normal variates with upper truncation points at $\beta_1\mathbf{X}_1/C_{11}$ and $({\beta_2}'\mathbf{X}_2 - C_{21}{e_1}^*)/C_{22}$, respectively. Computation of Q_1 is straightforward, and if one had some specific values for ${e_1}^*$ and ${e_2}^*$, then one could also compute Q_2 and Q_3 and hence the overall multivariate probability.

The GHK simulator derives values for ${e_1}^*$ and ${e_2}^*$ by taking random draws from upper-truncated standard normal distributions with truncation points as given above and then recursively computes a multivariate probability value from the Qs. (The procedure generalizes straightforwardly to the case where $M > 3$; there are as many Q terms as there are equations.) The process is replicated R times, and the simulated probability—the value that is included in the log-likelihood function at each iteration—is the arithmetic mean of the values of the simulated probabilities from each replication.

The drawing of random variables from upper-truncated normal distributions is done using a random-number generator combined with the inversion formula given by, among others, Stern (1997). Recall that a univariate standard normal variate is generated by applying the inverse of the normal probability function to random numbers drawn from a uniform distribution over the unit interval (`invnorm(uniform())`). Draws from upper-truncated standard normal distributions can be obtained similarly. The truncated standard normal probability is given by $p \equiv \Pr(x|x < b) = \Phi(x)/\Phi(b)$, where b is the upper truncation point. The desired random variate is calculated using the formula $\Phi^{-1}\{p\Phi(b)\}$.

The GHK simulator has a number of desirable properties in the context of multivariate normal limited dependent variable models (Börsch-Supan and Hajivassiliou 1993): the simulated probabilities are unbiased, they are bounded within the (0,1) interval, and the simulator is a continuous and differentiable function of the model's parameters. The GHK simulator is also more efficient (in terms of the variance of the estimators of probabilities) than other simulators such as the acceptance–rejection or Stern simulators.

Under standard conditions, the simulated maximum likelihood (SML) estimator is consistent as the number of draws and the number of observations tend to infinity. Thus, the desirable properties of the SML estimator are asymptotic, as they are for all ML estimators. And, intuitively, the sample size required to reduce the finite sample bias to some acceptable level will increase with the number of equations.

Simulation bias is reduced to negligible levels when the number of draws is raised with the sample size. Ensuring that the ratio of the number of draws to the square root of sample size is sufficiently large ensures this (Hajivassiliou and Ruud 1994, 2416–2419). Thus, other things equal, the more draws there are, the more accurate the results. In practice, however, it has been observed that a relatively small number of draws may work well for 'smooth' likelihoods. We illustrate the impact of changing the number of draws in section 4.

By its very nature, estimation using SML is numerically intensive, and convergence may be very slow, particularly if the number of draws is large, or especially if the number of equations is large. By contrast, changing the number of explanatory variables does not affect convergence speed very much.

3 The mvprobit program

The `mvprobit` program fits multivariate probit models using the method of SML described in the previous section. The number of equations in the model is unlimited in principle, though subject to speed and capacity constraints discussed later. The program is written for Stata 7, using `ml model lf`. In the next subsections, we describe the syntax and options for `mvprobit` and for the companion program for post-estimation prediction, `mvppred`.

3.1 Syntax for mvprobit

The syntax for `mvprobit` is very similar to that for the seemingly unrelated bivariate probit model syntax of `biprobit`:

`mvprobit` *equation1 equation2* `...` *equationM* [*weight*] [`if` *exp*] [`in` *range*]
[, `draws(`#`)` `seed(`#`)` `beta0` `atrho0(`*matrix_name*`)` `robust`
`cluster(`*varname*`)` `constraints(`*numlist*`)` `level(`#`)` *maximize_options*]

where each equation is specified as

([*eqname*`:`] *depvar* [`=`] [`varlist`] [`, noconstant`])

`by ...:` may be used with `mvprobit`. `pweights`, `fweights`, and `iweights` are allowed. `mvprobit` shares the features of all estimation commands, including access to estimated results, and `mvprobit` typed without arguments redisplays the last estimates.

Predictions based on `mvprobit` estimates, including predicted joint and marginal probabilities, can be derived using `mvppred`, discussed below.

Restrictions on the structure of the correlation matrix may be imposed using the `constraint` option.

3.2 Options for mvprobit

`draws(`#`)` specifies the number of random variates drawn when calculating the simulated likelihood. The default is 5.

`seed(`#`)` specifies the initial value of the (pseudo) random-number seed used by the `uniform()` function in the simulation process. The value should be an integer (the default value is 123456789).

`beta0` specifies that the estimates of the marginal probit regressions (used to provide starting values) are reported.

`atrho0(`*matrix_name*`)` allows users to specify starting values for the off-diagonal elements of the correlation matrix V that differ from the default starting values, which are all zero. More precisely, the matrix *matrix_name* contains values of the incidental parameter in each `/atrhojk` equation (see section 4); i.e., $\text{atanh}\rho_{jk} = 0.5 * \log\{(1+\rho_{jk})/(1-\rho_{jk})\}$. Matrix *matrix_name* must have properly named column names. For example, if a starting value in `/atrho21` is being set, one would first use the command matrix *matrix_name* = (value), followed by `matrix colnames` *matrix_name* = `atrho21:_cons`. Between 1 and $M(M-1)/2$ `/atrhojk` starting values may be specified, where $j = 2, \ldots, M$ and $k < j$.

The remaining options are the same as the corresponding ones for `biprobit`.

3.3 Syntax for the prediction program mvppred

`mvppred` *newvarname_prefix* [`if` *exp*] [`in` *range*] [`,` *statistic*]

where *statistic* is one of

`xb`	the linear prediction for each equation; the default.
`stdp`	the standard error of the linear predictions for each equation.
`pmarg`	the marginal success probability for each equation.
`pall`	the joint probabilities: (i) $\Pr(y_{im} = 1$, for all $m = 1, \ldots, M)$, and (ii) $\Pr(y_{im} = 0$, for all $m = 1, \ldots, M)$.

Only one statistic may be chosen at a time. For statistics `xb`, `stdp`, and `pmarg`, results are stored in the variables *newvarname_prefixi*, for equations $i = 1, \ldots, M$. For the `pall` statistics, results are stored in the variables *newvarname_prefix1s* for predicted probability (i) and *newvarname_prefix0s* for predicted probability (ii). The options for predicting joint probabilities are restricted to the 'all successes' and 'all failures' cases because these were the only two cases that could be programmed without the number of equations being fixed. (The number of joint probabilities corresponding to the different combinations of successes and failures is 2^M.)

4 Illustrations

We use two datasets to illustrate `mvprobit`. First, we take the `school` dataset from Pindyck and Rubinfeld (1998, 332), which is used to illustrate `biprobit` in the *Stata 7 Reference Manual*. This example has two purposes: it shows the command syntax, including options (such as for prediction), in action, and it helps demonstrate the accuracy of the simulated maximum estimator `mvprobit` relative to the maximum likelihood estimator. (Because the sample size, 95, is 'small', both the SML and ML estimators may be subject to finite sample bias.) Second, we fit a four-equation model using data generated from a model with prespecified parameters. (The sample size in this case, 5,000, is relatively large, and so finite sample bias is less of an issue.)

4.1 Illustration using the school data

The `school` dataset contains 95 observations on whether children attend a private school (`private`), the number of years that the family has been at the present residence (`years`), the logarithm of property tax (`logptax`), the logarithm of income (`loginc`), and whether the household head had voted for an increase in the property tax (`vote`). We model the bivariate outcomes of whether children attend private school and whether the household head voted for an increase in property tax as functions of log property tax, log income, and residential tenure. The `biprobit` estimates of this model are as follows:

```
. use http://www.stata-press.com/data/r7/school, clear
. biprobit (private=years logptax loginc) (vote=years logptax loginc), nolog
Seemingly unrelated bivariate probit              Number of obs   =         95
                                                  Wald chi2(6)    =       9.59
Log likelihood = -89.254028                       Prob > chi2     =     0.1431
```

	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
private						
years	-.0118884	.0256778	-0.46	0.643	-.0622159	.0384391
logptax	-.1066962	.6669782	-0.16	0.873	-1.413949	1.200557
loginc	.3762037	.5306484	0.71	0.478	-.663848	1.416255
_cons	-4.184694	4.837817	-0.86	0.387	-13.66664	5.297253
vote						
years	-.0168561	.0147834	-1.14	0.254	-.0458309	.0121188
logptax	-1.288707	.5752266	-2.24	0.025	-2.416131	-.1612839
loginc	.998286	.4403565	2.27	0.023	.1352031	1.861369
_cons	-.5360573	4.068509	-0.13	0.895	-8.510188	7.438073
/athrho	-.2764525	.2412099	-1.15	0.252	-.7492153	.1963102
rho	-.2696186	.2236753			-.6346806	.1938267

```
Likelihood-ratio test of rho=0:     chi2(1) =  1.38444    Prob > chi2 = 0.2393
```

We also generate predictions of several joint and marginal probabilities and linear predictions and their standard errors, for comparisons with their SML counterparts below:

```
. predict p11, p11
. predict p00, p00
. predict xbb1, xb1
. predict xbb2, xb2
. predict stdpb1, stdp1
. predict stdpb2, stdp2
. predict pmargb1, pmarg1
. predict pmargb2, pmarg2
```

The `mvprobit` estimates of the same model, with the number of random draws set at the default 5, are given by

(*Continued on next page*)

```
. mvprobit (private = years logptax loginc) (vote=years logptax loginc), nolog
Multivariate probit (SML, # draws = 5)          Number of obs   =         95
                                                Wald chi2(6)    =       9.24
Log likelihood = -89.773212                     Prob > chi2     =     0.1608
```

	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
private						
years	-.0088706	.0239362	-0.37	0.711	-.0557847	.0380436
logptax	-.1232144	.6665783	-0.18	0.853	-1.429684	1.183255
loginc	.3991725	.5389061	0.74	0.459	-.6570639	1.455409
_cons	-4.321043	4.880546	-0.89	0.376	-13.88674	5.244652
vote						
years	-.0177467	.0148995	-1.19	0.234	-.0469492	.0114557
logptax	-1.275053	.5703058	-2.24	0.025	-2.392832	-.1572747
loginc	.9799365	.4399119	2.23	0.026	.117725	1.842148
_cons	-.438055	4.050873	-0.11	0.914	-8.37762	7.50151
/atrho21	-.1161373	.1997462	-0.58	0.561	-.5076327	.2753582
rho21	-.1156179	.1970761	-0.59	0.557	-.4680986	.2686035

```
Likelihood ratio test of  rho21 = 0:
             chi2(1) =  .346069   Prob > chi2 = 0.5563
```

The SML estimates of the coefficients and their statistical significance are very close to the ML estimates, even though the number of draws is relatively small. There is a sharp contrast in the estimates of the correlation between the equation error terms, however: -0.116 compared with -0.270. Raising the number of random draws used by the SML estimator brings the estimates much closer together, however. With $R = 100$, we have

(*Continued on next page*)

```
. mvprobit (private = years logptax loginc) (vote=years logptax loginc), nolog
> draws(100)
Multivariate probit (SML, # draws = 100)          Number of obs   =         95
                                                  Wald chi2(6)    =       9.64
Log likelihood = -89.220805                       Prob > chi2     =     0.1405
```

	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
private						
years	-.0118233	.0256205	-0.46	0.644	-.0620386	.0383921
logptax	-.1033056	.6672673	-0.15	0.877	-1.411126	1.204514
loginc	.3695001	.5303571	0.70	0.486	-.6699806	1.408981
_cons	-4.140149	4.837923	-0.86	0.392	-13.6223	5.342006
vote						
years	-.0172153	.0148029	-1.16	0.245	-.0462285	.011798
logptax	-1.280732	.5725493	-2.24	0.025	-2.402908	-.1585563
loginc	.9956743	.437904	2.27	0.023	.1373982	1.85395
_cons	-.5627991	4.055359	-0.14	0.890	-8.511157	7.385558
/atrho21	-.2811165	.2396506	-1.17	0.241	-.7508231	.18859
rho21	-.2739382	.2216667	-1.24	0.217	-.6356397	.1863855

```
Likelihood ratio test of  rho21 = 0:
             chi2(1) =  1.45088   Prob > chi2 = 0.2284
```

The SML estimate of ρ_{21} is now -0.274 ($z = -1.24$) compared with the ML estimate of -0.270 ($z = -1.21$).

Predictions following `mvprobit` are obtained using the command `mvppred`. The predictions are based on the parameter estimates from the last `mvprobit` model fitted and use the same number of random draws (and seed) to generate the joint probabilities. For example,

```
. mvppred pall, pall
(Pr(all zeros), Pr(all ones) will be stored in variables pall0s, pall1s)
. mvppred xbm, xb
(xb will be stored in variables xbmi, i = 1,...,#eqs)
. mvppred stdpm, stdp
(stdp will be stored in variables stdpmi, i = 1,...,#eqs)
. mvppred pmargm, pmarg
(pmarg will be stored in variables pmargmi, i = 1,...,#eqs)
```

The SML predictions are very similar to their ML counterparts. For example, the mean `mvprobit` prediction of $\Pr(\texttt{private} = 1, \texttt{vote} = 1)$ is 0.0514, compared with the mean `biprobit` prediction of 0.0515:

```
. summarize pall1s p11 pall0s p00 xbm1 xbb1 xbm2 xbb2 /*
>         */ stdpm1 stdpb1 stdpm2 stdpb2 pmargm1 pmargb1 pmargm2 pmargb2
```

Variable	Obs	Mean	Std. Dev.	Min	Max
pall1s	95	.0513848	.0293697	.0006823	.1675037
p11	95	.0514965	.0295284	.0006783	.1691212
pall0s	95	.3252403	.1496049	.0397381	.8772917
p00	95	.3241522	.1485598	.040815	.882799
xbm1	95	-1.273431	.2017621	-1.930628	-.8744448
xbb1	95	-1.275218	.2041617	-1.937996	-.8695227
xbm2	95	.3308476	.4381363	-1.365069	1.519954
xbb2	95	.3313479	.4383708	-1.37215	1.52224
stdpm1	95	.3404043	.1805338	.185824	1.046698
stdpb1	95	.3405953	.1807651	.1859334	1.049172
stdpm2	95	.2541217	.1181758	.141525	.7976922
stdpb2	95	.2546185	.1186652	.1415696	.8023056
pmargm1	95	.1057509	.0322633	.0267645	.190938
pmargb1	95	.1055308	.032576	.0263119	.1922807
pmargm2	95	.6216642	.1502225	.0861157	.9357387
pmargb2	95	.6218135	.1500823	.0850083	.9360256

In sum, the results suggest that `mvprobit`'s SML estimates are similar to those of the corresponding ML estimator—conditional on the number of random draws used in the former being sufficiently large. We return to the issue of the choice of R later.

4.2 Illustration using generated data

We now extend the illustrations of `mvprobit` to the four-equation case. To benchmark the results, we generate a dataset with 5,000 observations from a multivariate probit model with known parameters. The data were created using methods similar to those discussed in *Stata 7 Reference Manual* volume 2, page 36.

```
. set seed 12309
. set obs 5000
obs was 0, now 5000
. matrix R=(1, .25, .5, .75 \ .25, 1, .75, .5 \ .5, .75, 1, .75 \ .75, .5, .75, 1)
. drawnorm u1 u2 u3 u4, corr(R)
. correlate u*
(obs=5000)
```

	u1	u2	u3	u4
u1	1.0000			
u2	0.2587	1.0000		
u3	0.5077	0.7483	1.0000	
u4	0.7523	0.5093	0.7589	1.0000

```
. generate x1 = uniform()-.5
. generate x2 = uniform() + 1/3
. generate x3 = 2*uniform() + .5
. generate x4 = .5*uniform() - 1/3
```

```
. *Equations
. ge y1s = .5 + 4*x1 + u1
. ge y2s = 3 + .5*x1 - 3*x2 + u2
. ge y3s = 1 - 2*x1 + .4*x2 - .75*x3 + u3
. ge y4s = -6 + 1*x1 - .3*x2 + 3*x3 - .4*x4 + u4
. ge y1 = y1s>0
. ge y2 = y2s>0
. ge y3 = y3s>0
. ge y4 = y4s>0
```

The equations for `y1s`, `y2s`, `y3s`, and `y4s` correspond to the equations for $y_{im}{}^*$, $i = 1, \ldots, M$ given at the beginning of section 2, and those for `y1`, `y2`, `y3`, and `y4` correspond to those for y_{im}. The correlations between the error terms (the elements of the matrix V) are shown in the output from the `correlate` command.

`mvprobit` estimates of this four-equation model are set out below for the case in which the number of random draws $R = 75$ (i.e., slightly larger than the square root of the sample size).

```
. mvprobit (y1=x1) (y2=x1 x2) (y3 = x1 x2 x3) (y4=x1 x2 x3 x4), dr(75)
Iteration 0:   log likelihood = -8681.8526
Warning: cannot do Cholesky factorization of rho matrix
Iteration 1:   log likelihood =  -7922.415
Iteration 2:   log likelihood = -7749.5236
Iteration 3:   log likelihood = -7746.5769
Iteration 4:   log likelihood = -7746.5734
Iteration 5:   log likelihood = -7746.5734
Multivariate probit (SML, # draws = 75)           Number of obs   =       5000
                                                  Wald chi2(10)   =    5561.10
Log likelihood = -7746.5734                       Prob > chi2     =     0.0000
```

	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
y1						
x1	3.991634	.0962586	41.47	0.000	3.80297	4.180297
_cons	.5078066	.0233257	21.77	0.000	.462089	.5535241
y2						
x1	.5413448	.0704179	7.69	0.000	.4033283	.6793614
x2	-2.84842	.0781794	-36.43	0.000	-3.001648	-2.695191
_cons	2.867846	.0734467	39.05	0.000	2.723894	3.011799
y3						
x1	-2.011164	.0708565	-28.38	0.000	-2.15004	-1.872288
x2	.5341271	.0642228	8.32	0.000	.4082527	.6600015
x3	-.7438451	.0311735	-23.86	0.000	-.8049441	-.6827462
_cons	.9133876	.0725944	12.58	0.000	.7711051	1.05567
y4						
x1	1.125271	.0891551	12.62	0.000	.9505303	1.300012
x2	-.3030878	.0826195	-3.67	0.000	-.4650191	-.1411565
x3	2.898115	.0779738	37.17	0.000	2.745289	3.050941
x4	-.4352364	.1598866	-2.72	0.006	-.7486084	-.1218644
_cons	-5.751499	.1685475	-34.12	0.000	-6.081846	-5.421152

/atrho21	.2621884	.0317031	8.27	0.000	.2000516	.3243253
/atrho31	.5646645	.0345949	16.32	0.000	.4968597	.6324692
/atrho41	.9396437	.0571792	16.43	0.000	.8275746	1.051713
/atrho32	.9737443	.0406156	23.97	0.000	.8941391	1.053349
/atrho42	.5670195	.0424459	13.36	0.000	.483827	.6502119
/atrho43	1.007126	.0537097	18.75	0.000	.9018571	1.112395
rho21	.2563413	.0296198	8.65	0.000	.1974249	.3134126
rho31	.5114301	.0255462	20.02	0.000	.4596439	.5597501
rho41	.7350585	.0262846	27.97	0.000	.6791715	.7824714
rho32	.7503451	.0177483	42.28	0.000	.7134321	.7831052
rho42	.513167	.0312682	16.41	0.000	.4493033	.5718126
rho43	.7645708	.0223127	34.27	0.000	.7172009	.8049075

```
Likelihood ratio test of  rho21 = rho31 = rho41 = rho32 = rho42 = rho43 = 0:
             chi2(6) =  1870.56   Prob > chi2 = 0.0000
```

Clearly, `mvprobit` provides good estimates of the underlying model—not only of the regression coefficients, but also of the correlation matrix. The warning about Cholesky factorization that appears between the first and second iterations is not a matter of concern, as the model subsequently converged satisfactorily. The GHK simulator relies on a Cholesky factorization, and in order to do this, the estimate of the correlation matrix V at each iteration has to be positive definite. Occasionally this is not so, in which case `mvprobit` traps the error and uses instead the most recent estimate of V (which is guaranteed to be positive definite).

5 Using mvprobit: further remarks

The number of replications, R, used by the GHK simulator is a key choice for `mvprobit` users. Increasing R increases accuracy but at the cost of lengthening run time. There is also a choice to be made about the seed—different seeds lead to different sets of random numbers being used to calculate the simulated probabilities and hence potentially different parameter estimates. In order to investigate these issues, we re-estimated the models discussed in section 4 for several alternative seed values (including the default as before), in each case varying the number of replications between 1 and 150.

Our first experiments were based on the generated dataset and the two-equation model for `y1s` and `y2s`. Our discussion focuses, for brevity's sake, on the estimates of correlation ρ_{21} only. Figure 1 shows the estimates generated using the default seed (left-hand graph) and a seed of 999 (right-hand graph). The horizontal line in each graph

is a benchmark estimate, 0.266729, derived from `biprobit`. Both graphs suggest that the SML estimate approaches the ML estimate relatively rapidly as the number of draws increases. Changing the seed can make some difference, however. For a seed of 999, a larger number of random draws was required before the SML estimate settled down ($R \approx 50$, rather than $R \approx 25$, for the default seed), and convergence, as R increased, was to a value that was not as close to the ML estimate. We also repeated the exercise for a number of other seeds (31, 11111111, 33333333, 55555555, 77777777, 99999999). Taken together, the results suggest that the SML estimator provided a good estimate of ρ_{21} (and its standard error), regardless of the choice of seed value, when the number of replications was at least as large as the square root of the sample size (71 in this case).

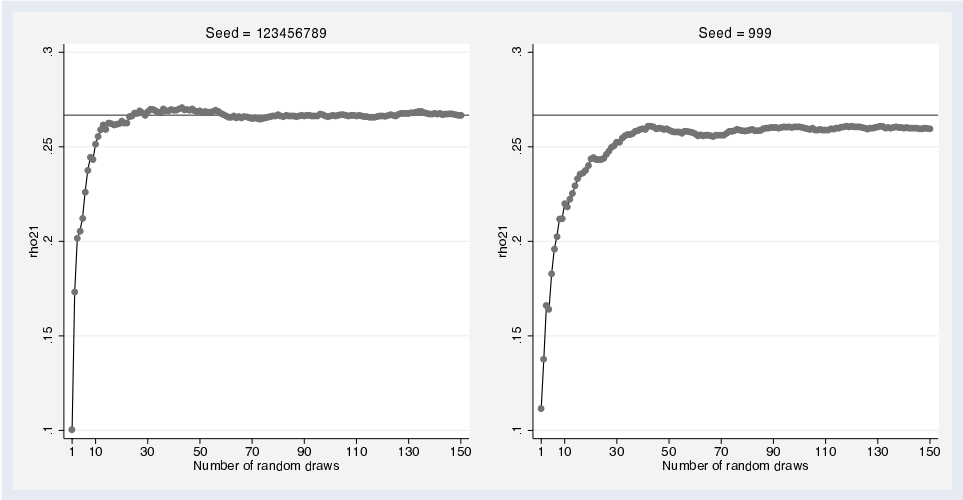

Figure 1: Variation in SML estimate of correlation ρ_{21} with number of replications, for `seed` = 123456789 and `seed` = 999 (two-equation model, generated data)

Figure 2 shows that conclusions such as these need to be treated with some caution when the sample size is relatively small. The figure is based on the `school` dataset used in section 4 and shows the SML estimate of ρ_{21} as R was varied from 1 to 150, for each of two different seeds (the default and 999). Again the ML estimate was taken as the benchmark. In this case, convergence of the estimate was much slower, nonmonotonic, and did not occur until R was at least as large as the sample size.

(*Continued on next page*)

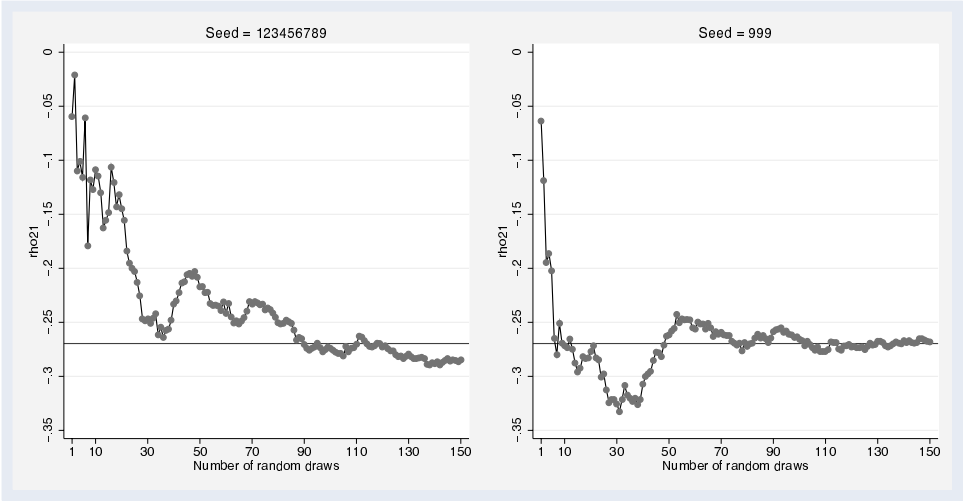

Figure 2: Variation in SML estimate of correlation ρ_{21} with number of replications, for `seed` = 123456789 and `seed` = 999 (two-equation model, `school` dataset)

We advise `mvprobit` users to choose a relatively large number for R, though a small value of R would suffice when checking program syntax statements. For sample sizes common in social surveys (of the order of several thousands), setting R equal to an integer approximately equal to the square root of the sample size may suffice, and the estimates are likely to be insensitive to the choice of seed. Conversely, users should be warned that for small sample sizes, estimates may be sensitive to the choice of seed value unless R is large. (There would remain the issue of finite sample bias, of course.)

For a given sample size and number of equations, run time increases roughly linearly in the number of replications (Hajivassiliou 1997). The four-equation model based on simulated data that was discussed in section 4 took about 2.25 hours to run when estimated using Stata/SE 7 for Windows running on a Pentium P4/1.4 Ghz PC. The same model estimated using networked Intercooled Stata 7 running on a Sun Solaris computer took about 5.25 hours.

Use of the `atrho0()` option may provide some scope for reduction in run time. It allows users to specify starting values for off-diagonal elements of the correlation matrix V that differ from the default starting values (which are all zero). (Values are specified in the atanh metric as described in section 3.2.) A natural source of a nondefault starting value for element jk is the `/athrho` parameter estimated from the `biprobit` model corresponding to equations j and k of a multi-equation `mvprobit` model. Our experiments with the option used in this way suggest that there are gains in speed, but they are not large. This is because, even with the default starting values, relatively good estimates of V are derived within just one or two iterations. An alternative use of the option would be when run times were expected to be long (for example, because a large R was required). One might run the model with a smaller R initially and use the estimated `/athrho` parameters from this model as starting values for the main model.

Run time varies substantially with the number of equations in the model. For datasets with several thousand observations (and appropriate R), our experiments suggest that estimation may take days or even a few weeks when the number of equations is above six or seven. Thus, although the number of equations that `mvprobit` can handle is unlimited in principle, there are practical constraints that are likely to affect most users.

Other potential constraints concern the matsize and memory available. The simulation procedure creates $R \times M$ temporary variables. Users need to ensure that sufficient memory is available for these (`set memory`). For large models (many explanatory variables in each of many equations), users may also need to increase matsize (`set matsize`).

6 Acknowledgments

Nick Cox, Weihua Guan, Mark Stewart, Nick Winter, and an anonymous referee offered helpful comments and suggestions. This research was funded by a Nuffield Foundation New Career Development Fellowship and core funding to ISER from the UK Economic and Social Research Council and the University of Essex.

7 References

Börsch-Supan, A. and V. Hajivassiliou. 1993. Smooth unbiased multivariate probability simulators for maximum likelihood estimation of limited dependent variable models. *Journal of Econometrics* 58: 347–368.

Börsch-Supan, A., V. Hajivassiliou, L. Kotlikoff, and L. Morris. 1992. Health, children, and elderly living arrangements: A multiperiod-multinomial probit model with unobserved heterogeneity and autocorrelated errors. In *Topics in the Economics of Aging*, ed. D. Wise, 79–104. Chicago: The University of Chicago Press.

Gourieroux, C. and A. Monfont. 1996. *Simulation-Based Econometric Methods*. Oxford: Oxford University Press.

Greene, W. H. 2003. *Econometric Analysis*. 5th ed. Upper Saddle River, NJ: Prentice–Hall.

Hajivassiliou, V. 1997. Some practical issues in maximum simulated likelihood. STICERD Discussion Paper no. EM/97/340. London: London School of Economics. Downloadable from *http://econ.lse.ac.uk/staff/vassilis/index_own.html*.

Hajivassiliou, V. and P. Ruud. 1994. Classical estimation methods for LDV models using simulation. In *Handbook of Econometrics*, eds. R. Engle and D. McFadden, vol. IV, 2383–2441. Amsterdam: North-Holland.

Keane, M. P. 1994. A computationally practical simulation estimator for panel data. *Econometrica* 62: 95–116.

Pindyck, R. S. and D. L. Rubinfeld. 1998. *Econometric Models and Economic Forecasts*. 4th ed. New York: McGraw–Hill.

Stern, S. 1997. Simulation-based estimation. *Journal of Economic Literature* 35: 2006–2039.

About the Authors

Lorenzo Cappellari is an assistant professor at the Università del Piemonte Orientale (Novara, Italy) and a visiting researcher at the Institute for Social and Economic Research (ISER), at the University of Essex, Colchester, UK.

Stephen Jenkins is a professor at ISER and an associate editor of the *Stata Journal*.