Estimating Fully Observed Recursive Mixed-Process Models with `cmp`

David Roodman

Abstract

At the heart of many econometric models is a linear function and a normal error. Examples include the classical small-sample linear regression model and the probit, ordered probit, multinomial probit, Tobit, interval regression, and truncated-distribution regression models. Because the normal distribution has a natural multidimensional generalization, such models can be combined into multi-equation systems in which the errors share a multivariate normal distribution. The literature has historically focused on multi-stage procedures for estimating mixed models, which are more efficiently computationally, if less so statistically, than maximum likelihood (ML). But faster computers and simulated likelihood methods such as the Geweke, Hajivassiliou, and Keane (GHK) algorithm for estimating higher-dimensional cumulative normal distributions have made direct ML estimation practical. ML also facilitates a generalization to switching, selection, and other models in which the number and types of equations vary by observation. The Stata module `cmp` fits Seemingly Unrelated Regressions (SUR) models of this broad family. Its estimator is also consistent for recursive systems in which all endogenous variables appear on the right-hand-sides as observed. If all the equations are structural, then estimation is full-information maximum likelihood (FIML). If only the final stage or stages are, then it is limited-information maximum likelihood (LIML). `cmp` can mimic a dozen built-in Stata commands and several user-written ones. It is also appropriate for a panoply of models previously hard to estimate. Heteroskedasticity, however, can render it inconsistent. This paper explains the theory and implementation of `cmp` and of a related Mata function, `ghk2()`, that implements the GHK algorithm.

Estimating fully observed recursive mixed-process models with $\mathtt{cmp}$[1]

David Roodman

April 2009

Abstract

At the heart of many econometric models are a linear function and a normal error. Examples include the classical small-sample linear regression model and the probit, ordered probit, multinomial probit, Tobit, interval regression, and truncated-distribution regression models. Because the normal distribution has a natural multidimensional generalization, such models can be combined into multi-equation systems in which the errors share a multivariate normal distribution. The literature has historically focussed on multi-stage procedures for estimating mixed models, which are more efficiently computationally, if less so statistically, than maximum likelihood (ML). But faster computers and simulated likelihood methods such as the Geweke, Hajivassiliou, and Keane (GHK) algorithm for estimating higher-dimensional cumulative normal distributions have made direct ML estimation practical. ML also facilitates a generalization to switching, selection, and other models in which the number and types of equations vary by observation. The Stata module `cmp` fits Seemingly Unrelated Regressions (SUR) models of this broad family. Its estimator is also consistent for recursive systems in which all endogenous variables appear on the right-hand-sides *as observed*. If all the equations are structural, then estimation is full-information maximum likelihood (FIML). If only the final stage or stages are, then it is limited-information maximum likelihood (LIML). `cmp` can mimic a dozen built-in Stata commands and several user-written ones. It is also appropriate for a panoply of models previously hard to estimate. Heteroskedasticity, however, can render it inconsistent. This paper explains the theory and implementation of `cmp` and of a related Mata function, `ghk2()`, that implements the GHK algorithm.

1 Introduction

Econometrics is most straightforward when dealing with variables whose domains are continuous and unbounded. But economists are often confronted with data that do not come directly from such variables. Sometimes this complication reflects reality: women are either pregnant or not; people do not work for negative numbers of hours. Sometimes it reflects the structure of data collection instruments that, for example, ask yes/no questions and solicit 5-point ratings. A common approach to modeling such *limited dependent variables* is to assume that the data generating processes is classically linear and unbounded at its heart, with a normally distributed error term. *Link functions* of chosen form translate these *latent variables* into the observed ones. Examples include the probit, ordered probit, rank-ordered probit, multinomial probit, and Tobit models, as well as those for interval data and truncated distributions.

Also common are situations in which it is desirable to model or instrument several such variables at once, whether in a Seemingly Unrelated Regressions (SUR) set-up, in which the dependent variables are generated by processes that are independent except for correlated errors, or in the broader simultaneous equations framework, in which endogenous variables influence each other. A poor household's decision about how much microcredit to borrow—a variable censored from the left at 0—might influence the binary variable of a child's enrollment in school, and vice versa. General tools for estimating parameters in such multi-equation systems are rare, perhaps because the likelihoods can be complicated and fitting them computationally demanding.[1] Until recently, official and user-written Stata commands have filled small parts of this space piecemeal. `ivtobit`, for example, implements estimators for Tobit models when some variables on the RHS are endogenous. `heckprob` brings Heckman selection modeling to probit models, making a two-equation system. `cmp` is the first general Stata tool for this class of models, and even it could be extended much further. At this writing, `cmp` implements an estimator for all the model types above except rank-ordered probit; and it allows mixing of these models in multi-equation systems. `cmp` is written as an SUR estimator. Yet it works for a substantially larger class of simultaneous-equation systems, namely, ones having two properties:

1. *Recursivity*, meaning that the equations can be arranged so that the matrix of coefficients of the endogenous variables in each other's equations is triangular. Recursive models have clearly defined stages, with one ore more equations in each stage.

---

[1] The aML package by Lee Lillard and Constantijn Panis, now available at applied-ml.com, showed the practicality of a general tool, and indeed is substantially broader than `cmp`. It allows full simultaneity in systems of equations, random effects at various clustering levels, and more model types.

2. What I call *full observability*, meaning that endogenous variables appear on the right-sides of equations only as *observed*. A dummy endogenous variable, for example, can be included in an equation, but the hypothesized linear variable latent within it cannot.

Given this mathematical scope, `cmp` is appropriate for two broad types of estimation situations: 1) those in which a truly recursive data-generating process is posited and fully modeled; and 2) those in which there is simultaneity but instruments allow the construction of a recursive set of equations, as in two-stage least squares (2SLS). In the first case, `cmp` is a full-information maximum likelihood (FIML) estimator, all estimated parameters being structural. In the latter, it is a limited-information (LIML) estimator, and only the final stage's (or stages') parameters are structural, the rest being reduced-form.

`cmp` is flexible in another way: models can vary by observation. In other words, they can be conditioned on the data. "cmp" stands for Conditional Mixed Process. Thus, within the `cmp` universe is the Heckman selection model, in which sample selection, represented by a dummy variable, is modeled in parallel with a dependent variable of interest. Sample selection is modeled for the full data set and the dependent variable for the subset with complete observations. The framework also embraces switching regressions in which the model used for a given variable depends on the data. And it allows suppression of equations that do not apply for particular observations. Pitt and Khandker (1998), in the example that inspired `cmp`, study the effects of male and female microcredit borrowing on household outcomes such as consumption and school enrollment in Bangladesh. Male and female credit are instrumented, but their equations are dropped from the model for households in villages with no program offering credit to their sex. (Notice the mix of processes too: log consumption is continuous and unbounded, enrollment is binary, and credit is censored from the left.)

One measure of the flexibility of the `cmp` framework is the list of Stata commands it can emulate more or less fully: `probit`, `ivprobit`, `treatreg`, `biprobit`, `oprobit`, `mprobit`, `asmprobit`, `tobit`, `ivtobit`, `cnreg`, `intreg`, `truncreg`, `heckman`, `heckprob`, in principle even `regress`, and `sureg`, as well as the user-written `triprobit`, `mvprobit` (Cappellari and Jenkins 2003), `bitobit`, `mvtobit`, `oheckman` (Chiburis and Lokshin 2007), and `bioprobit` (in its "non-endogenous" mode; Sajaia 2006). Of course, the purpose of `cmp` is not to replicate capabilities already available, but to make practical a wide array of new ones.

Section 2 of this paper explains the mathematics of estimating fully observed recursive mixed-process models that are conditioned on the data. Section 3 discusses some practicalities of implementation in Stata. Section 4 details how to use `cmp`, with examples and tips.

2 Fully observed recursive mixed-process models

2.1 The building blocks

We start the exposition by briefly stating the individual models available in `cmp`. All are are built on linear models and the Gaussian distribution, and so can be seen as specific instances of a larger family of models. All but the multinomial probit model have just one equation. All but classical linear regression and truncated-distribution regression involve censoring. One purpose of this review is to express them all within a unified, formal structure, in order to prepare for combining them in mixed models.

2.1.1 Classical linear regression

The model is

$$y^* = \theta + \varepsilon$$

$$\theta = \mathbf{x}'\boldsymbol{\beta}$$

$$\mathrm{E}[\varepsilon|\mathbf{x}] = 0$$

$$\varepsilon \sim \mathcal{N}\left(0, \sigma^2\right)$$

where y and ε are random variables, $\mathbf{x} = (x_1, \ldots, x_k)'$ is a column vector of K predetermined variables, and $\boldsymbol{\beta}$ is a vector of coefficients. For the sake of maximum likelihood (ML) estimation, we assume that the errors are normally distributed, even though they need not be for large-sample Ordinary Least Squares (OLS).

Representing the zero-centered normal distribution by $\phi\left(u; \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{u^2}{2\sigma^2}}$, the likelihood for observation i is

$$L_i\left(\boldsymbol{\beta}, \sigma^2; y_i | \mathbf{x}_i\right) = \phi\left(y_i - \theta_i; \sigma^2\right). \tag{1}$$

To express this model and likelihood more universally, we define the probability distribution function for ε as $f_\varepsilon(u) = \phi\left(u; \sigma^2\right)$; the *link function* (trivial in this case) as $g(y^*) = y$; and an "error link function" to connect the error process to the outcome:

$$h(\varepsilon) = g(\theta + \varepsilon). \tag{2}$$

3

In terms of these functions, the likelihood is (rather pedantically)

$$L_i\left(\boldsymbol{\beta}, \sigma^2; y_i | \mathbf{x}_i\right) = \int_{h^{-1}(y_i)} f_\varepsilon\left(\varepsilon\right) d\varepsilon \tag{3}$$

where the domain of integration is the single point

$$h^{-1}\left(y_i\right) = \{y_i - \theta_i\} \tag{4}$$

and the integral of a probability density over such a singleton is interpreted as the density at that point.

2.1.2 Truncated regression

In the truncated linear regression model, the dependent variable is confined to some range. An example is studying income determinants among low-income people. The model posits lower and upper truncation points, $\underline{\tau}_i$ and $\bar{\tau}_i$, that can vary by observation. For generality, they can take the value $-\infty$ or ∞ respectively. Within the sample, the model for y is the same as above. But the likelihood must be normalized by the total probability over the observable range:

$$L_i\left(\boldsymbol{\beta}, \sigma^2, \underline{\tau}_i, \bar{\tau}_i; y_i | \mathbf{x}_i\right) = \frac{\phi\left(y_i - \theta_i; \sigma^2\right)}{\Phi\left(\bar{\tau}_i - \theta_i; \sigma^2\right) - \Phi\left(\underline{\tau}_i - \theta_i; \sigma^2\right)}$$

where $\Phi()$ is the cumulative normal distribution. In more abstract terms, it is

$$L_i\left(\boldsymbol{\beta}, \sigma^2, \underline{\tau}_i, \bar{\tau}_i; y_i | \mathbf{x}_i\right) = \frac{\displaystyle\int_{h^{-1}(y_i)} f_\varepsilon\left(\varepsilon\right) d\varepsilon}{\displaystyle\int_T f_\varepsilon\left(\varepsilon\right) d\varepsilon}. \tag{5}$$

where T is the region $[\underline{\tau}_i - \theta_i, \bar{\tau}_i - \theta_i]$. If $\underline{\tau}_i = -\infty$ and $\bar{\tau}_i = \infty$, the denominator is 1, and this formulation reduces to (3).

2.1.3 Censored (Tobit) regression

Where truncation excludes observations with the dependent variable outside some range, censoring retains observations while confining the variable to a range. So the link function is now

$$
y = g\left(y^*\right) = \begin{cases} \underline{c} & \text{if} \quad y^* \leq \underline{c} \\ y^* & \text{if} \quad \underline{c} < y^* < \bar{c} \\ \bar{c} & \text{if} \quad y^* \geq \bar{c} \end{cases} \tag{6}
$$

where $\underline{c}$ and $\bar{c}$ are censoring instead of truncation threshholds. The definition of $h()$ relative to $g()$ (in (2)) does not change. The likelihood is

$$
L_i\left(\boldsymbol{\beta}, \sigma^2, \underline{c}, \bar{c}; y_i | \mathbf{x}_i\right) = \left. \begin{cases} \Phi\left(\underline{c} - \theta_i; \sigma^2\right) & \text{if} \quad y_i \leq \underline{c} \\ \phi\left(y_i - \theta_i; \sigma^2\right) & \text{if} \quad \underline{c} < y_i < \bar{c} \\ 1 - \Phi\left(\bar{c} - \theta_i; \sigma^2\right) & \text{if} \quad y_i \geq \bar{c} \end{cases} \right\} = \int_{h^{-1}(y_i)} f_\varepsilon\left(\varepsilon\right) d\varepsilon \tag{7}
$$

where

$$
h^{-1}\left(y_i\right) = \begin{cases} \left(-\infty, \underline{c} - \theta_i\right] & \text{if} \quad y_i \leq \underline{c} \\ \left\{y_i - \theta_i\right\} & \text{if} \quad \underline{c} < y_i < \bar{c} \\ \left[\bar{c} - \theta_i, \infty\right) & \text{if} \quad y_i \geq \bar{c} \end{cases} \quad . \tag{8}
$$

If $\underline{c} = -\infty$ and $\bar{c} = \infty$, this formula for $h^{-1}()$ also reduces to (3), so it generalizes (3) in a different way than (5) does. We will save the grand unification for section 2.2.1.

The Tobit model is so commonplace that its mathematical peculiarity is often overlooked. It mixes cumulative probabilities integrated over one-dimensional ranges with probability densities computed at zero-dimensional points. The overall likelihood is the product of probabilities of both types. It is not obvious that maximizing such a mixed-probability likelihood is consistent. Fifteen years passed between when James Tobin (1958) explored estimators of this type (and he was not the first) and when Takeshi Amemiya (1973) proved their consistency. Later in this paper, we take advantage of this fact in writing down integrals whose domains of integration are all embedded in an error space of fixed dimension, yet whose own dimensions vary by observation along with the number of equations whose realizations are censored. For observations in which none of the equations is censored, the domain of integration will be zero-dimensional, just as for uncensored observations in the one-equation Tobit model. Defining the integrals above to signify probability

densities as well as cumulative probabilities paves the way for expressing likelihoods economically.[2]

A more practical complication that is sometimes missed is the way heteroskedasticity can compromise the consistency of ML estimatation of Tobit and other censored models. We defer this issue to section 2.2.5.

2.1.4   Probit

The model is changed from the previous section in that

$$y = g\left(y^*\right) = \begin{cases} 0 & \text{if} \quad y^* \leq 0 \\ 1 & \text{if} \quad y^* > 0 \end{cases} \tag{9}$$

This model involves two normalizations. It normalizes *location* by setting 0 as the cut point, which costs nothing in generality if $\mathbf{x}$ contains a constant. And because it is no longer possible to determine the *scale* of y^*, the model normalizes to $\sigma^2 = 1$. But for consistency with the notation of other models, we still include σ^2 in the equations. The probit link function gives rise to the likelihood

$$L_i\left(\boldsymbol{\beta}, \sigma^2; y_i | \mathbf{x}_i\right) = \begin{cases} \Phi\left(-\theta_i; \sigma^2\right) & \text{if} \quad y_i = 0 \\ 1 - \Phi\left(-\theta_i; \sigma^2\right) & \text{if} \quad y_i = 1 \end{cases} = \int_{h^{-1}(y_i)} f_\varepsilon\left(\varepsilon\right) d\varepsilon$$

where $h^{-1}\left(0\right) = \left(-\infty, -\theta_i\right]$ and $h^{-1}\left(1\right) = \left(-\theta_i, \infty\right)$.

2.1.5   Ordered probit

The ordered probit model is for variables with ordered, discrete values. It generalizes the probit model by slicing the continuum into a finite set of ranges, each corresponding to one possible outcome. Unlike in the Tobit and probit models, the cut points are unknown parameters for estimation. If we wanted to maximize consistency with the definition of the probit model above, we would fix one of the cut points at 0. But we will follow the convention set by Stata's built-in `oprobit` command, which is to make all the cut points free parameters and remove the constant term from $\mathbf{x}$.

Assume y can achieve J outcomes, $O_1, \ldots, O_J$. Use the ascending sequence of cut points $c_1, \ldots, c_{J-1}$ to

---

[2]A more rigorous statement of the observation-level likelihoods discussed in this section is that they are probability functions (p.f.'s) of $y_i | \mathbf{x_i}$ induced by the maps $h\left(\right)$ into error space. These p.f.'s are in general *mixed distributions*, containing both mass points and ranges of continuous distribution. See the discussion of the Stieltjes integral in Ruud (2000, pp. 875–76.).

define the regions into which y^* might fall, and define $c_0 = -\infty$ and $c_J = \infty$. Then the link function is:

$$
y = g(y^*) = \begin{cases} O_1 & \text{if} \quad c_0 < y^* \le c_1 \\ \vdots \\ O_j & \text{if} \quad c_{j-1} < y^* \le c_j \\ \vdots \\ O_J & \text{if} \quad c_{J-1} < y^* < c_J \end{cases} .
$$

Again, we normalize to $\sigma^2 = 1$. For the case of $y_i = O_j$, the likelihood is

$$
L_i\left(\boldsymbol{\beta}, \sigma^2, c_1, \ldots, c_{J-1}; y_i | \mathbf{x}_i\right) = \Phi\left(c_j - \theta_i; \sigma^2\right) - \Phi\left(c_{j-1} - \theta_i; \sigma^2\right) = \int_{h^{-1}(y_i)} f_\varepsilon\left(\varepsilon\right) d\varepsilon
$$

where the region of integration is

$$
h^{-1}(y_i) = (c_{j-1} - \theta_i, c_j - \theta_i]. \tag{10}
$$

2.1.6 Interval regression

The interval regression model is identical to that for ordered probit, except that cut points are known. An agricultural census, for example, might report farm landholdings by bracket: less than 1 hectare, 1–10 hectares, 10–100 hectares, etc. So the likelihood when $y_i = O_j$ differs only in having a shorter parameter list on the left:

$$
L_i\left(\boldsymbol{\beta}, \sigma^2; y_i | \mathbf{x}_i\right) = \Phi\left(c_j - \theta_i; \sigma^2\right) - \Phi\left(c_{j-1} - \theta_i; \sigma^2\right) = \int_{h^{-1}(y_i)} f_\varepsilon\left(\varepsilon\right) d\varepsilon.
$$

2.1.7 Multinomial probit

The multinomial probit model applies to situations in which an agent chooses from alternatives that are not inherently ordered, such the brand of car to buy or whether to fly or drive to a destination. All that is observed is the chosen alternative. Observations are often called *cases*, and the agent in each case chooses from a discrete set of *alternatives*. The model is more complicated than any of the foregoing ones because it involves multiple equations. (See Long and Freese (2006, ch. 7) and Train (2003, ch. 5) for more.) It posits

7

one "utility" equation for each alternative, indexed by $j = 1, \ldots, J$:

$$y_j^* = \theta_j + \varepsilon_j$$

$$\theta_j = \mathbf{x}_j' \boldsymbol{\beta}_j.$$

The $\mathbf{x}_j$ can be distinct variable sets, but overlap. A regressor that appears in every equation, such as buyer's income in a car choice model, is *case-specific*. The opposite, an *alternative-specific* variable, such a car's fuel economy, can be thought of as a single variable that varies across alternatives. But the structure used here treats it as a set of variables, one for each equation: Ford fuel economy, Volkswagen fuel economy, etc. The ε_j can be correlated, according to $\boldsymbol{\varepsilon} = (\varepsilon_1, \ldots, \varepsilon_J)' \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is a positive-definite symmetric matrix. The alternative with the highest utility is chosen.

Let $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_J)'$ and $\mathbf{y}^* = (y_1^*, \ldots, y_J^*)'$. A convenient way to express the outcome is to model a vector of dummy variables $\mathbf{y} = (y_1, \ldots, y_J)'$, only one of which can be 1 for any particular case. The vector-valued link function is then

$$\mathbf{g}(\mathbf{y}^*) = \left[\mathbf{1}\left\{ k = \arg\max_j y_j^* \right\}; k = 1, \ldots, J \right]'$$

where $\mathbf{1}\{\}$ is a dummy variable indicating the truth of the bracketed equality.

Because of the nature of choice we can only study determinants of relative, not absolute, desirability. In other words, the equations above are underidentified. For example, if buyer's income is a determinant for every car model, then variation in that variable will only reveal its influence on the *relative* attractiveness of various models, not its *absolute* impact on each one. To make the model identified, we choose a "base alternative" and exclude from its equation regressors that appear in the utility equations of all other alternatives. In particular, the constant term is normally excluded for the base alternative.

To derive the likelihood for some case i, suppose the agent chooses alternative k. The probability that this will happen is the probability that y_{ik}^* is greater than all the other y_{ij}^*. To state that precisely, define $\mathbf{M}_k$ as the $(J-1) \times J$ matrix made by inserting a column of -1's as the new k^{th} column in the $(J-1)$

8

identity matrix. E.g., if there are 4 alternatives and $k = 3$,

$$
\mathbf{M}_k = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}.
\tag{11}
$$

Left-multiplying a J-vector of variables by this matrix subtracts the kth entry from the others and then deletes it. So let $\tilde{\mathbf{y}}^* = \mathbf{M}_k \mathbf{y}^*$, $\tilde{\boldsymbol{\theta}} = \mathbf{M}_k \boldsymbol{\theta}$, and $\tilde{\boldsymbol{\varepsilon}} = \mathbf{M}_k \boldsymbol{\varepsilon}$. The choice of alternative k implies that the utilities of all other alternatives are negative relative to k's: $\tilde{\mathbf{y}}^* = \tilde{\boldsymbol{\theta}} + \tilde{\boldsymbol{\varepsilon}} < \mathbf{0}$. That is, $\tilde{\boldsymbol{\varepsilon}} < -\tilde{\boldsymbol{\theta}}$. Finally, define

$$
\tilde{\boldsymbol{\Sigma}}_i \equiv \operatorname{Var}\left[\tilde{\boldsymbol{\varepsilon}}\right] = \operatorname{Var}\left[\mathbf{M}_k \boldsymbol{\varepsilon}\right] = \mathbf{M}_k \operatorname{Var}\left[\boldsymbol{\varepsilon}\right] \mathbf{M}_k' = \mathbf{M}_k \boldsymbol{\Sigma} \mathbf{M}_k'.
\tag{12}
$$

$\tilde{\boldsymbol{\Sigma}}$ is indexed by i because it depends on which alternative is chosen in case i. The likelihood is then

$$
L_i\left(\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_J, \boldsymbol{\Sigma}; \mathbf{y}_i | \mathbf{x}_i\right) = \Pr\left(\tilde{\boldsymbol{\varepsilon}}_i < -\tilde{\boldsymbol{\theta}}_i\right) = \Phi\left(-\tilde{\boldsymbol{\theta}}; \tilde{\boldsymbol{\Sigma}}_i\right)
\tag{13}
$$

where $\Phi()$ is the multidimensional cumulative normal distribution.

To write this in the more abstract terms of link and distributions functions, let $\tilde{\mathbf{g}}_i\,()$ be the implied link function from transformed latent variables $\mathbf{y}^*$ to observed outcomes. Its domain is the set of possible utilities of alternatives other than k relative to alternative k; it maps these to vectors of dummies that are 1 only for the chosen, highest-utility alternative. Then define

$$
\tilde{f}_{\tilde{\boldsymbol{\varepsilon}}_i}\left(\mathbf{u}\right) = \phi\left(\mathbf{u}; \tilde{\boldsymbol{\Sigma}}\right)
$$
$$
\tilde{\mathbf{h}}_i\left(\tilde{\boldsymbol{\varepsilon}}_i\right) = \tilde{\mathbf{g}}_i\left(\tilde{\boldsymbol{\theta}}_i + \tilde{\boldsymbol{\varepsilon}}_i\right).
$$

The likelihood can now be expressed in a general form with a multidimensional integral:

$$
L_i\left(\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_J, \boldsymbol{\Sigma}; \mathbf{y}_i | \mathbf{x}_i\right) = \int_{\tilde{\mathbf{h}}_i^{-1}(\mathbf{y}_i)} f_{\tilde{\boldsymbol{\varepsilon}}_i}\left(\tilde{\boldsymbol{\varepsilon}}\right) d\tilde{\boldsymbol{\varepsilon}}
$$

where, recall, $\mathbf{y}_i$ is a vector that is all 0's except for a 1 in the position of the chosen alternative and

$$
\tilde{\mathbf{h}}_i^{-1}\left(\mathbf{y}_i\right) = \left\{\tilde{\boldsymbol{\varepsilon}} \,\middle|\, \tilde{\boldsymbol{\varepsilon}} < -\tilde{\boldsymbol{\theta}}\right\}.
\tag{14}
$$

This formula for the likelihood generalizes the earlier ones in pre-transforming the error space (by $\mathbf{M}_k$) before integrating.

As written, this model still has several excess degrees of freedom. While this consideration is important in applying the multinomial probit model, it does not change the mathematical form of the likelihood in (13), which is our primary interest here. So this discussion of the identification issues is brief.

One issue is that, just as in the probit model, scale needs to be normalized. The analog of setting $\sigma^2 = 1$ in the probit model is to place a constraint on $\mathbf{\Sigma}$ or $\tilde{\mathbf{\Sigma}}_i$. But here we run into a more complicated problem. One consequence of the relative nature of utility is that the likelihood in (13) depends on the $(J-1) \times (J-1)$ matrix $\tilde{\mathbf{\Sigma}}_i$, not the $J \times J$ matrix $\mathbf{\Sigma}$. Thus not all of the elements of $\mathbf{\Sigma}$ can be identified. To fit the model, one can impose structure on $\mathbf{\Sigma}$: J constraints to account for the symmetric $\mathbf{\Sigma}$ having J more independent elements, plus one more constraint to normalize scale.

Alternatively, one can parameterize using a covariance matrix for relative-differenced errors—relative, that is, to an alternative that is fixed across cases. We take this to be the base alternative, which we assume is alternative 1, and label this covariance matrix $\hat{\mathbf{\Sigma}}$ to distinguish it from $\tilde{\mathbf{\Sigma}}_i$, the covariance of the errors relative to the *chosen* alternative, which varies by case. Given trial values for $\hat{\mathbf{\Sigma}}$, the implied values for $\tilde{\mathbf{\Sigma}}_i$ are readily computed. To see this, return again to the example of $J = 4$ and $k = 3$. The vector of errors relative to the base alternative is related to the vector relative to the chosen alternative by the transformation

$$\tilde{\varepsilon}_i = \begin{bmatrix} \varepsilon_{i1} - \varepsilon_{i3} \\ \varepsilon_{i2} - \varepsilon_{i3} \\ \varepsilon_{i4} - \varepsilon_{i3} \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & -1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} \varepsilon_{i2} - \varepsilon_{i1} \\ \varepsilon_{i3} - \varepsilon_{i1} \\ \varepsilon_{i4} - \varepsilon_{i1} \end{bmatrix} \equiv \mathbf{N}_3 \hat{\varepsilon}.$$

So

$$\tilde{\mathbf{\Sigma}}_i = \mathrm{Var}\left[\varepsilon_i\right] = \mathrm{Var}\left[\mathbf{N}_3 \hat{\varepsilon}\right] = \mathbf{N}_3 \mathrm{Var}\left[\hat{\varepsilon}\right] \mathbf{N}_3' = \mathbf{N}_3 \hat{\mathbf{\Sigma}} \mathbf{N}_3'.$$

In general, N_k is M_k without its first column. With this parameterization, just one constraint, such as $\hat{\mathbf{\Sigma}}_{11} = 1$, is needed, in order to normalize scale.

Unfortunately, both $\mathbf{\Sigma}$ and $\hat{\mathbf{\Sigma}}$ have disadvantages as bases for parameterization. Maximizing likelihood with respect to $\hat{\mathbf{\Sigma}}$ is mathematically sound, but yields estimates of quantities such as $\mathrm{Cov}\left[\varepsilon_3 - \varepsilon_1, \varepsilon_2 - \varepsilon_1\right]$ (an element of $\hat{\mathbf{\Sigma}}$) that are hard to interpret. Fitting with respect to $\mathbf{\Sigma}$ is more intuitive. But it turns out to be surprisingly difficult to impose the needed $J + 1$ constraints while guaranteeing that the mapping

$\boldsymbol{\Sigma} \mapsto \hat{\boldsymbol{\Sigma}} = \mathbf{N}_1 \boldsymbol{\Sigma} \mathbf{N}_1'$ is *onto*. That is, even when the constraints are minimally arbitrary and meant to remove excess degrees of freedom—not to restrict the model—there may be positive-definite $(J-1) \times (J-1)$ matrices that are valid candidates for $\hat{\boldsymbol{\Sigma}}$ yet which are not compatible with the constraints on $\boldsymbol{\Sigma}$ (Bunch 1991). This can prevent the model fit from reaching the true optimum. Freese and Long (2006, pp. 327–29) provide an example.

A final, separate complication is that $\hat{\boldsymbol{\Sigma}}$ is not identified unless there are alternative-specific regressors, such as fuel economy (Keane 1992). This is why the Stata command `mprobit`, which allows only case-specific variables, makes the assumption of Independence of Irrelevant Alternatives (IIA), i.e., that $\boldsymbol{\Sigma}$ is an identity matrix. By (12), under the IIA,

$$\hat{\boldsymbol{\Sigma}}_i = \mathbf{M}_1 \mathbf{M}_1' = \begin{bmatrix} 2 & 1 & \cdots & 1 \\ 1 & 2 & \ddots & 1 \\ \vdots & \ddots & \ddots & \vdots \\ 1 & 1 & \cdots & 2 \end{bmatrix}. \tag{15}$$

The newer `asmprobit` allows alternative-specific variables and frees up the structure of $\boldsymbol{\Sigma}$.

2.2   Multi-equation mixed models

2.2.1   The SUR model

Because all of these models are built on the classical linear regression model with normally distributed errors, they combine naturally into systems of "seemingly unrelated" equations. Equations in an SUR system seem unrelated in the sense that no endogenous (LHS) variables appear on the right side of other equations. But their errors can be correlated, sharing a multidimensional distribution. Parameters in SUR systems can be consistently estimated equation-by-equation, but simultaneous estimation that takes into account the full covariance structure is in general more efficient.

The SUR model is

$$\mathbf{y}^{*\prime}_{\substack{1 \times J}} = \boldsymbol{\theta}'_{\substack{1 \times J}} + \boldsymbol{\varepsilon}'_{\substack{1 \times J}}$$

$$\boldsymbol{\theta}'_{\substack{1 \times J}} = \mathbf{x}'_{\substack{1 \times K}} \mathbf{B}_{\substack{K \times J}}$$

$$\mathrm{E}[\boldsymbol{\varepsilon}|\mathbf{x}] = \mathbf{0}$$

$$\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$$

$$\mathbf{y} = \mathbf{g}(\mathbf{y}^*) = (g_1(\mathbf{y}^*), \ldots, g_J(\mathbf{y}^*))'.$$

where $\boldsymbol{B}$ is a matrix of coefficients, $\mathbf{y}$ and $\boldsymbol{\varepsilon}$ are random vectors, and $\mathbf{x} = (x_1, \ldots, x_k)'$ is a vector of predetermined random variables. In general, constraints can be imposed on $\mathbf{B}$ and $\boldsymbol{\Sigma}$—for example, not all elements of $\mathbf{x}$ need enter every equation—but we will ignore this complexity for the sake of exposition.

Variables of most types require just one equation within the system. But a multinomial probit variable requires one for each alternative. As before, the observed dependent variables for multinomial probit equations are dummies collectively indicating which alternative is chosen. Since the multinomial probit link functions depend on several equations—an outcome depends on the utilities of all alternatives—each $g_j()$ is allowed to depend on all of $\mathbf{y}^*$, not just y_j^*.

To state the likelihood for observation i in this model in a way that embraces all the individual models stated in 2.1, we need a mathematical form that encompasses the likelihoods displayed earlier. It must allow truncation, probability functions that mix continuous and discrete components, and linear pre-transformation of the data (as in multinomial probits). For economy of presentation, gather unknown censoring points for ordered probits into a single vector $\mathbf{c}$. To express truncation, let $\underline{\tau}_{ij}$ and $\bar{\tau}_{ij}$ be the lower and upper truncation bounds for y_j; they take infinite values if there is no truncation (truncation makes no sense in probit and multinomial probit models, for instance). The overall truncation range, the region of possible values of $\boldsymbol{\varepsilon}$ that could generate observable values for $\mathbf{y}$ given $\mathbf{x}$, is the Cartesian product

$$T_i = [\underline{\tau}_{i1} - \theta_{i1}, \bar{\tau}_{i1} - \theta_{i1}] \times \ldots \times [\underline{\tau}_{iJ} - \theta_{iJ}, \bar{\tau}_{iJ} - \theta_{iJ}]. \tag{16}$$

In addition, let m be the number of multinomial probit variables (thus multinomial equation groups) in the model, $\tilde{J}$ be $J - m$, and $\mathbf{M}_i$ be the $\tilde{J} \times J$ matrix that when left-multiplied with $\mathbf{y}^*$ and $\mathbf{x}$ subtracts the data for the multinomial probit equations for chosen alternatives from the corresponding rejected ones while leaving

data for non-multinomial-probit equations untouched. $\mathbf{M}_i$ is block-diagonal, with blocks analogous to that in (11) for multinomial probit equation groups and "blocks" that are simply 1 for equations of other types. As in the multinomial probit discussion, we define $\tilde{\mathbf{y}}_i^* = \mathbf{M_i y}^*$, $\tilde{\boldsymbol{\theta}}_i = \mathbf{M_i}\boldsymbol{\theta}$, $\tilde{\boldsymbol{\varepsilon}}_i = \mathbf{M_i}\boldsymbol{\varepsilon}$, $\tilde{\boldsymbol{\Sigma}}_i = \mathrm{Var}\left[\tilde{\boldsymbol{\varepsilon}}_i\right] = \mathbf{M}_i\boldsymbol{\Sigma}\mathbf{M}_i'$, $\tilde{T}_i = \mathbf{M}_i T_i$, and $\tilde{\mathbf{g}}()$ as the link function implied by $\mathbf{g}()$ from the space of $\mathbf{M}_i$-transformed errors to outcomes. Then the general observation-level likelihood is given by

$$f_{\tilde{\boldsymbol{\varepsilon}}_i}(\mathbf{u}) = \phi\left(\mathbf{u}; \tilde{\boldsymbol{\Sigma}}_i\right)$$

$$\tilde{\mathbf{h}}_i\left(\tilde{\boldsymbol{\varepsilon}}_i\right) = \tilde{\mathbf{g}}_i\left(\tilde{\boldsymbol{\theta}}_i + \tilde{\varepsilon}_i\right)$$

$$L_i\left(\mathbf{B}, \boldsymbol{\Sigma}, \mathbf{c}; \mathbf{y}_i | \mathbf{x}_i\right) = \frac{\displaystyle\int\limits_{\tilde{\mathbf{h}}_i^{-1}(\mathbf{y}_i)} f_{\tilde{\boldsymbol{\varepsilon}}_i}(\tilde{\boldsymbol{\varepsilon}})\, d\tilde{\varepsilon}}{\displaystyle\int\limits_{\tilde{T}_i} f_{\tilde{\boldsymbol{\varepsilon}}_i}(\tilde{\boldsymbol{\varepsilon}})\, d\tilde{\varepsilon}}. \tag{17}$$

The observation-level likelihood is the ratio of two integrals over certain regions of the distribution $f_{\tilde{\boldsymbol{\varepsilon}}_i}$, which is known given $\boldsymbol{\Sigma}$ and the outcomes of any multinomial choices. Both regions of integration have fairly simple forms: Cartesian products of line segments, rays, and lines ("Cartesian" for short). Since T is unbounded in the dimensions corresponding to multinomial probit equations, transforming the space by $\mathbf{M}_i$ to produce $\tilde{T}$ merely deletes a few unbounded dimensions of T—one for each multinomial probit variable's chosen alternative in case i. So $\tilde{T}$ is the Cartesian product in (16), except deleting the components that correspond to these alternatives. Importantly, $\tilde{\mathbf{h}}_i^{-1}(\mathbf{y}_i)$ too is rectilinear, because the realization of one equation's error term for a given observation, ε_{ij} does not affect the feasible range for another equation's, ε_{ij}. Thus $\tilde{\mathbf{h}}_i^{-1}(\mathbf{y}_i)$ is a Cartesian product of the types of domains defined in (4), (8), (10), and (14):

$$\tilde{\mathbf{h}}_i^{-1}(\mathbf{y}_i) = \left[\underline{c}_{i1} - \tilde{\theta}_{i1}, \bar{c}_{i1} - \tilde{\theta}_{i1}\right] \times \ldots \times \left[\underline{c}_{i\tilde{J}} - \tilde{\theta}_{i\tilde{J}}, \bar{c}_{i\tilde{J}} - \tilde{\theta}_{i\tilde{J}}\right]. \tag{18}$$

If y_{ij} is uncensored, then $\underline{c}_{ij} = \bar{c}_{ij}$; in the censored case, either bound can be infinite.

2.2.2 SUR examples

The general likelihood above looks a bit formidable, but it works out intuitively in elementary examples.

Example 1. *Bivariate probit*

13

The model is

$$y_1^* = \theta_1 + \varepsilon_1$$

$$y_2^* = \theta_2 + \varepsilon_2$$

$$\theta_1 = \beta_1 x$$

$$\theta_2 = \beta_2 x$$

$$\mathbf{y} = \mathbf{g}\left(\mathbf{y}^*\right) = \left(\mathbf{1}\left\{y_1^* > 0\right\}, \mathbf{1}\left\{y_2^* > 0\right\}\right)'$$

$$\boldsymbol{\varepsilon} = \left(\varepsilon_1, \varepsilon_2\right)' \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Sigma}\right)$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}.$$

The diagonal entries of $\boldsymbol{\Sigma}$ are 1 to normalize scale for both equations. With reference to the definitions in the previous section, since there are no multinomial probit equations, $\mathbf{M}$ is the identity matrix, and we can dispense with the $\sim$ hats. Since the model does not vary by observation, we can drop some of the i subscripts. And because there is no truncation, the denominator of (17) is 1. We have:

$$f_{\boldsymbol{\varepsilon}}\left(\mathbf{u}\right) = \phi\left(\mathbf{u}; \boldsymbol{\Sigma}\right)$$

$$\mathbf{h}\left(\boldsymbol{\varepsilon}\right) = \mathbf{g}\left(\boldsymbol{\theta} + \boldsymbol{\varepsilon}\right)$$

$$L_i\left(\beta_1, \beta_2, \rho; \mathbf{y}_i | \mathbf{x}_i\right) = \int_{\mathbf{h}^{-1}(\mathbf{y}_i)} f_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\varepsilon}\right) d\boldsymbol{\varepsilon}.$$

Suppose that we observe $y_{i1} = y_{i2} = 0$. Then the space of possible values of the latent variables $\mathbf{y}_i^*$ is the quarter plane $\mathbf{g}^{-1}\left(\mathbf{y}_i\right) = (-\infty, 0] \times (-\infty, 0]$ and the space of possible values for the errors $\boldsymbol{\varepsilon}_i$ is the quarter plane $\mathbf{h}^{-1}\left(\mathbf{y}_i\right) = (-\infty, -\theta_{i1}] \times (-\infty, -\theta_{i2}]$. Integrating the probability distribution over this Cartesian range gives us the likelihood for this particular pair of outcomes:

$$L_i\left(\beta_1, \beta_2, \rho; \mathbf{y}_i | \mathbf{x}_i\right) = \int_{-\infty}^{-\theta_{i1}} \int_{-\infty}^{-\theta_{i2}} \phi\left(\left(\varepsilon_1, \varepsilon_2\right)'; \boldsymbol{\Sigma}\right) d\varepsilon_2 d\varepsilon_1 = \Phi\left(\left(-\theta_{i1}, -\theta_{i2}\right)'; \boldsymbol{\Sigma}\right). \tag{19}$$

Similarly, if $\mathbf{y}_i = (1, 1)'$, the likelihood works out to $\int_{-\theta_1}^{\infty} \int_{-\theta_2}^{\infty} \phi\left(\left(\varepsilon_1, \varepsilon_2\right)'; \boldsymbol{\Sigma}\right) d\varepsilon_2 d\varepsilon_1$, which by the symmetry

14

of the normal distribution is $\Phi\left((\theta_{i1}, \theta_{i2})'; \Sigma\right)$ In general, it works out that

$$L_i(\beta_1, \beta_2, \rho; \mathbf{y}_i | \mathbf{x}_i) = \Phi\left((q_1\theta_{i1}, q_2\theta_{i2})'; \Sigma\right), \text{ where } q_j = 2y_{ij} - 1. \tag{20}$$

Example 2. *A mixed probit-uncensored model*

We modify the previous example to illustrate how the likelihood works out in a model that mixes censored and uncensored variables. Equation 1 is classically linear while equation 2 is probit:

$$y_1^* = \theta_1 + \varepsilon_1$$

$$y_2^* = \theta_2 + \varepsilon_2$$

$$\theta_1 = \beta_1 x$$

$$\theta_2 = \beta_2 x$$

$$\mathbf{y} = \mathbf{g}(\mathbf{y}^*) = (y_1^*, \mathbf{1}\{y_2^* > 0\})'$$

$$\varepsilon = (\varepsilon_1, \varepsilon_2)' \sim \mathcal{N}(\mathbf{0}, \Sigma)$$

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{12} & 1 \end{bmatrix}.$$

Suppose that we observe some $\mathbf{y}_i = (y_{i1}, 0)'$. Then the space over which to integrate the probability distribution for the errors is $\mathbf{h}^{-1}(\mathbf{y}_i) = \{y_{i1}\} \times (-\infty, -\theta_2]$, which is a one-dimensional ray within the plane. Integrating over this set,

$$L_i(\beta_1, \beta_2, \Sigma; \mathbf{y}_i | \mathbf{x}_i) = \int_{-\infty}^{-\theta_2} \phi\left((y_{i1} - \theta_{i1}, \varepsilon_2)'; \Sigma\right) d\varepsilon_2. \tag{21}$$

This formula is accurate, but impossible to compute directly using standard functions available in statistical software. To make it practical—and to illustrate how `cmp` computes such mixed likelihoods—we need to factor $\phi(\cdot; \Sigma)$ into probability distribution functions for ε_1 and $\varepsilon_2 | \varepsilon_1$. Fortunately, the reproductive rules for the normal distribution generally boil down to linear algebra in mean vectors and covariance matrices. The rule we need is:

Lemma 1. *(Conditional distribution of a multivariate normal distribution.) Let* $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, \Sigma)$ *be a random*

vector, $\mathbf{u} = (\mathbf{u}_1', \mathbf{u}_2')'$ *be a partitioning of* $\mathbf{u}$, *and*

$$\left[\begin{array}{cc} \mathbf{\Sigma}_{11} & \mathbf{\Sigma}_{12} \\ \mathbf{\Sigma}_{21} & \mathbf{\Sigma}_{22} \end{array} \right]$$

be a conformable partitioning of $\mathbf{\Sigma}$. *Then* $\mathbf{u}_1 \sim \mathcal{N}\left(\mathbf{0}, \mathbf{\Sigma}_{11}\right)$ *and*

$$\mathbf{u}_2 \mid \mathbf{u}_1 \sim \mathcal{N}\left(\boldsymbol{\mu}_{2|1}, \mathbf{\Sigma}_{2|1}\right).$$

where

$$\boldsymbol{\mu}_{2|1} = \mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1}\mathbf{u}_1 \tag{22}$$

$$\mathbf{\Sigma}_{2|1} = \mathbf{\Sigma}_{22} - \mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1}\mathbf{\Sigma}_{12}.$$

Proof. The math here is that of linear projection, akin to OLS. $\boldsymbol{\mu}_{2|1}$, the expectation of $\mathbf{u}_2$ given $\mathbf{u}_1$, is

$$\boldsymbol{\mu}_{2|1} = \mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1}\mathbf{u}_1 = \mathrm{Cov}\left[\mathbf{u}_2, \mathbf{u}_1\right] \cdot \mathrm{Cov}\left[\mathbf{u}_1, \mathbf{u}_1\right]^{-1}\mathbf{u}_1;$$

this is the orthogonal projection of $\mathbf{u}_1$ into $\mathbf{u}_2$ space.[3] To see this, we check that the projection and $\mathbf{u}_2$'s deviation from it are orthogonal, having 0 covariance:

$$\mathrm{Cov}\left[\boldsymbol{\mu}_{2|1}, \mathbf{u}_2 - \boldsymbol{\mu}_{2|1}\right] = \mathrm{Cov}\left[\mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1}\mathbf{u}_1, \mathbf{u}_2 - \mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1}\mathbf{u}_1\right] = \mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1}\mathrm{Cov}\left[\mathbf{u}_1, \mathbf{u}_2\right] - \mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1}\mathrm{Var}\left[\mathbf{u}_1\right]\left(\mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1}\right)'$$

$$= \mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1}\mathbf{\Sigma}_{12} - \mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1}\mathbf{\Sigma}_{11}\mathbf{\Sigma}_{11}^{-1}\mathbf{\Sigma}_{12} = \mathbf{0}.$$

And we check that $\mathbf{\Sigma}_{2|1}$ is the variance of the deviation of $\mathbf{u}_2$ around its conditional expectation:

$$\mathrm{Var}\left[\mathbf{u}_2 - \boldsymbol{\mu}_{2|1}\right] = \mathrm{Cov}\left[\mathbf{u}_2 - \boldsymbol{\mu}_{2|1}, \mathbf{u}_2 - \boldsymbol{\mu}_{2|1}\right] = \mathrm{Cov}\left[\mathbf{u}_2, \mathbf{u}_2 - \boldsymbol{\mu}_{2|1}\right] - \mathrm{Cov}\left[\boldsymbol{\mu}_{2|1}, \mathbf{u}_2 - \boldsymbol{\mu}_{2|1}\right]$$

$$= \mathrm{Cov}\left[\mathbf{u}_2, \mathbf{u}_2 - \boldsymbol{\mu}_{2|1}\right] - 0 = \mathrm{Var}\left[\mathbf{u}_2\right] - \mathrm{Cov}\left[\mathbf{u}_2, \mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1}\mathbf{u}_1\right] = \mathbf{\Sigma}_{22} - \mathbf{\Sigma}_{21}\mathbf{\Sigma}_{11}^{-1}\mathbf{\Sigma}_{12} = \mathbf{\Sigma}_{2|1}.$$

This proof is incomplete in that it does not demonstrate that $\mathbf{u}_1$ and $\mathbf{u}_2 \mid \mathbf{u}_1$ are normal. See Greene (2000, p. 87) for a fuller proof that works directly with the formula for the normal distribution. $\square$

---

[3] Compare to the OLS projection $\hat{\mathbf{Y}} = \mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{Y}$ or, transposing, $\hat{\mathbf{Y}}' = \mathbf{y}'X\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'$.

For the model at hand, this formula for the conditional distribution of a normal distribution leads to the factoring

$$\phi\left(\left(\varepsilon_1, \varepsilon_2\right); \mathbf{\Sigma}\right) = \phi\left(\varepsilon_1; \sigma_{11}\right) \phi\left(\varepsilon_2 - \frac{\sigma_{21}}{\sigma_{11}}\varepsilon_1; 1 - \frac{\sigma_{21}\sigma_{12}}{\sigma_{11}}\right),$$

where $\sigma_{21} = \sigma_{12}$.

Plugging this into (21),

$$
\begin{aligned}
L_i\left(\beta_1, \beta_2, \mathbf{\Sigma}; \mathbf{y}_i | \mathbf{x}_i\right) &= \int_{-\infty}^{-\theta_2} \phi\left(y_{i1} - \theta_{i1}; \sigma_{11}\right) \phi\left(\varepsilon_2 - \frac{\sigma_{21}}{\sigma_{11}}\left(y_{i1} - \theta_{i1}\right); 1 - \frac{\sigma_{21}\sigma_{12}}{\sigma_{11}}\right) d\varepsilon_2 \\
&= \phi\left(y_{i1} - \theta_{i1}; \sigma_{11}\right) \int_{-\infty}^{-\theta_2} \phi\left(\varepsilon_2 - \frac{\sigma_{21}}{\sigma_{11}}\left(y_{i1} - \theta_{i1}\right); 1 - \frac{\sigma_{21}\sigma_{12}}{\sigma_{11}}\right) d\varepsilon_2 \\
&= \phi\left(y_{i1} - \theta_{i1}; \sigma_{11}\right) \Phi\left(-\theta_2 - \frac{\sigma_{21}}{\sigma_{11}}\left(y_{i1} - \theta_{i1}\right); 1 - \frac{\sigma_{21}\sigma_{12}}{\sigma_{11}}\right).
\end{aligned}
$$

This product of a one-dimensional normal probability density and a one-dimensional cumulative normal density can be computed with standard functions in statistical software.

2.2.3 Recursive systems

SUR systems are a special case of simultaneous-equation systems. In the larger class, endogenous variables can figure in each other's equations. Estimation in the broader framework is more complex, especially where there is censoring: a censored endogenous variable could influence other endogenous variables in either its latent or observed realization. cmp does *not* include features to handle these complexities. cmp is fundamentally an SUR estimation program. But it turns out that the ML SUR can consistently estimate parameters in an important subclass of mixed-process simultaneous systems: ones that are *recursive*, with clearly defined stages; and that are *fully observed*, meaning that endogenous variables appear on the right-hand side only as observed.

Recursive equation systems arise in two major ways. In the *full-information* case, the structural model is itself recursive and fully articulated (omitting no variables), and leads directly to a recursive set of equations that are the basis for ML estimation. In the more common *limited-information* case, only the final stage(s) are fully specified. Equations for earlier stages include instruments to address endogeneity or, more generally, omit influential variables. In this case, if the dependent variable(s) in the final stage are continuous and unbounded then simpler techniques such as 2SLS are consistent (Kelejian 1971). In addition, 2SLS and

17

related linear methods are consistent in the presence of heteroskedasticity whereas ML estimation that explicitly models the censoring may not be, as discussed in section 2.2.5 below. On the other hand, if the errors can be assumed to be identically (if not independently) distributed, a model that uses the information about limited nature of the earlier-stage dependent variables should be more accurate and efficient.

To be clear, the ML SUR framework works for fully observed recursive equation systems in the sense that simply inserting the observed endogenous variables into the vector of the predetermined variables, $\mathbf{x}$, in (17) yields likelihoods whose maximization generates consistent parameter estimates. This fact is not widely understood, except as it applies to the classical example of all dependent variables being linear.[4]

To fix ideas, we change the SUR model to

$$
\underset{1 \times J}{\mathbf{y}^{*\prime}} = \underset{1 \times J}{\boldsymbol{\theta}^{\prime}} + \underset{1 \times J}{\boldsymbol{\varepsilon}^{\prime}} \tag{23}
$$

$$
\underset{1 \times J}{\boldsymbol{\theta}^{\prime}} = \underset{1 \times J}{\mathbf{y}^{\prime}} \underset{J \times J}{\boldsymbol{\Delta}} + \underset{1 \times K}{\mathbf{x}^{\prime}} \underset{K \times J}{\mathbf{B}}
$$

$$
\mathrm{E}[\boldsymbol{\varepsilon}|\mathbf{x}] = \mathbf{0}
$$

$$
\boldsymbol{\varepsilon} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Sigma}\right)
$$

$$
\mathbf{y} = \mathbf{g}\left(\mathbf{y}^{*}\right) = \left[g_{1}\left(\mathbf{y}^{*}\right) \ldots g_{J}\left(\mathbf{y}^{*}\right)\right]^{\prime}.
$$

where $\boldsymbol{\Delta}$ is strictly upper triangular, meaning that the diagonal and the lower triangle are all 0's.

Just as in the SUR case, the likelihood for some observation i is an integral over a region of the probability distribution of $\boldsymbol{\varepsilon}_i$, potentially divided by a second integral to account for truncation, as in (17). For a seemingly inappropriate ML SUR estimator to compute this probability correctly despite treating $\mathbf{y}$ like $\mathbf{x}$ on the right, it must integrate the right distribution over the right regions. To demonstrate that it does, we want to factor the distribution at the heart of (17) in a way that conforms with the sequential structure of the model. Perhaps the best way to see this is with an example.

Example 3. *Two-stage probit*

---

[4]E.g., Greene (1998) writes "surprisingly" and "seem not to be widely known" in discussing the two-stage probit model.

We modify the bivariate probit model in Example 1 in one way, adding y_1 to the y_2^* equation:

$$y_1^* = \theta_1 + \varepsilon_1$$

$$y_2^* = \theta_2 + \varepsilon_2$$

$$\theta_1 = \beta_1 x$$

$$\theta_2 = \beta_2 x + \delta y_1$$

$$\mathbf{y} = \mathbf{g}(\mathbf{y}^*) = (\mathbf{1}\{y_1^* > 0\}, \mathbf{1}\{y_2^* > 0\})'$$

$$\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2)' \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}.$$

Note that ρ measures the endogeneity of y_1 in the y_2^* equation. For if ε_1 is uncorrelated with ε_2, so is y_1.

Begin again by supposing that for some observation i, we observe $\mathbf{y}_i = (0,0)'$. As in Example 1, the feasible range for ε_{i1} is $(-\infty, -\theta_{i1}]$. The distribution over this region is $f_{\varepsilon_1}(\varepsilon_1) = \phi(\varepsilon_1; \Sigma_{11})$ where $\Sigma_{11} = 1$. Conditioning on this realization of ε_{i1}, and given the resulting value of y_{i1}, the feasible range for ε_{i2} is $(-\infty, -\theta_{i2})$ and its distribution is $\mathcal{N}(\mu_{2|1}, \Sigma_{2|1})$, using the definitions in Lemma 1. So

$$L_i(\beta_1, \beta_2, \delta, \rho; \mathbf{y}_i | \mathbf{x}_i) = \int_{-\infty}^{-\theta_{i1}} f_{\varepsilon_1}(\varepsilon_1) \int_{-\infty}^{-\theta_{i2}} f_{\varepsilon_2 | \varepsilon_1}(\varepsilon_2) \, d\varepsilon_2 d\varepsilon_1.$$

Because the bounds for the inner integral do not depend on the variable for the outer one—y_{i1} is constant (at 0) over $(-\infty, -\theta_{i1}]$, so $\theta_{i2} = \beta_2 x_i + \delta y_{i1}$ is too—the domain of integration for the full, double integral is Cartesian, and we can write

$$L_i(\beta_1, \beta_2, \delta, \rho; \mathbf{y}_i | \mathbf{x}_i) = \int_{-\infty}^{-\theta_{i1}} \int_{-\infty}^{-\theta_{i2}} f_{\varepsilon_1}(\varepsilon_{i1}) f_{\varepsilon_2 | \varepsilon_1}(\varepsilon_{i2}) \, d\varepsilon_2 d\varepsilon_1 = \int_{-\infty}^{-\theta_{i1}} \int_{-\infty}^{-\theta_{i2}} f_{\boldsymbol{\varepsilon}}(\boldsymbol{\varepsilon}_i) \, d\varepsilon_2 d\varepsilon_1$$

$$= \Phi\left((-\theta_{i1}, -\theta_{i2})'; \boldsymbol{\Sigma}\right).$$

This likelihood equals that in (19), except that here θ_{i2} is a linear combination of y_{i1} as well as x_{i2}. Thus, adapting the SUR likelihood by treating y_1 as an ordinary, predetermined regressor produces the correct LIML or FIML likelihood (Maddala and Lee (1976), p. 526; Maddala (1983), pp. 122–23). ML estimation with the likelihood is consistent in both cases and efficient in the latter.

This example extends straightforwardly to the general fully observed recursive model:

$$L_i\left(\mathbf{B}, \boldsymbol{\Sigma}, \boldsymbol{\Delta}, \mathbf{c}; \mathbf{y}_i | \mathbf{x}_i\right) = \frac{\displaystyle\int_{\tilde{\mathbf{h}}_i^{-1}(\mathbf{y}_i)} f_{\tilde{\boldsymbol{\varepsilon}}_i}\left(\tilde{\boldsymbol{\varepsilon}}\right) d\tilde{\boldsymbol{\varepsilon}}}{\displaystyle\int_{\tilde{T}_i} f_{\tilde{\boldsymbol{\varepsilon}}_i}\left(\tilde{\boldsymbol{\varepsilon}}\right) d\tilde{\boldsymbol{\varepsilon}}}$$

$$= \frac{\displaystyle\int_{\underline{c}_1-\tilde{\theta}_{i1}}^{\bar{c}_1-\tilde{\theta}_{i1}} f_{\tilde{\varepsilon}_1}\left(\tilde{\varepsilon}_1\right) \int_{\underline{c}_2-\tilde{\theta}_{i2}}^{\bar{c}_2-\tilde{\theta}_{i2}} f_{\tilde{\varepsilon}_2|\tilde{\varepsilon}_1}\left(\tilde{\varepsilon}_2\right) \dots \int_{\underline{c}_{\bar{J}}-\tilde{\theta}_{i\bar{J}}}^{\bar{c}_{\bar{J}}-\tilde{\theta}_{i\bar{J}}} f_{\tilde{\varepsilon}_{\bar{J}}|\tilde{\varepsilon}_1,\dots,\tilde{\varepsilon}_{\bar{J}-1}}\left(\tilde{\varepsilon}_{\bar{J}}\right) d\tilde{\varepsilon}_{\bar{J}} \dots d\tilde{\varepsilon}_1}{\displaystyle\int_{\underline{\tau}_{i1}-\tilde{\theta}_{i1}}^{\bar{\tau}_{i1}-\tilde{\theta}_{i1}} f_{\tilde{\varepsilon}_1}\left(\tilde{\varepsilon}_1\right) \int_{\underline{\tau}_{i2}-\tilde{\theta}_{i2}}^{\bar{\tau}_{i2}-\tilde{\theta}_{i2}} f_{\tilde{\varepsilon}_2|\tilde{\varepsilon}_1}\left(\tilde{\varepsilon}_2\right) \dots \int_{\underline{\tau}_{i\bar{J}}-\tilde{\theta}_{i\bar{J}}}^{\bar{\tau}_{i\bar{J}}-\tilde{\theta}_{i\bar{J}}} f_{\tilde{\varepsilon}_{\bar{J}}|\tilde{\varepsilon}_1,\dots,\tilde{\varepsilon}_{\bar{J}-1}}\left(\tilde{\varepsilon}_{\bar{J}}\right) d\tilde{\varepsilon}_{\bar{J}} \dots d\tilde{\varepsilon}_1}$$

$$= \frac{\displaystyle\int_{\underline{c}_1-\tilde{\theta}_{i1}}^{\bar{c}_1-\tilde{\theta}_{i1}} \int_{\underline{c}_2-\tilde{\theta}_{i2}}^{\bar{c}_2-\tilde{\theta}_{i2}} \dots \int_{\underline{c}_{\bar{J}}-\tilde{\theta}_{i\bar{J}}}^{\bar{c}_{\bar{J}}-\tilde{\theta}_{i\bar{J}}} f_{\tilde{\boldsymbol{\varepsilon}}}\left(\tilde{\boldsymbol{\varepsilon}}\right) d\tilde{\boldsymbol{\varepsilon}}}{\displaystyle\int_{\underline{\tau}_{i1}-\tilde{\theta}_{i1}}^{\bar{\tau}_{i1}-\tilde{\theta}_{i1}} \int_{\underline{\tau}_{i2}-\tilde{\theta}_{i2}}^{\bar{\tau}_{i2}-\tilde{\theta}_{i2}} \dots \int_{\underline{\tau}_{i\bar{J}}-\tilde{\theta}_{i\bar{J}}}^{\bar{\tau}_{i\bar{J}}-\tilde{\theta}_{i\bar{J}}} f_{\tilde{\boldsymbol{\varepsilon}}}\left(\tilde{\boldsymbol{\varepsilon}}\right) d\tilde{\boldsymbol{\varepsilon}}} \tag{24}$$

where $f_{\tilde{\boldsymbol{\varepsilon}}}\left(\tilde{\boldsymbol{\varepsilon}}\right) = \phi\left(\tilde{\boldsymbol{\varepsilon}}; \tilde{\boldsymbol{\Sigma}}\right)$. Again, this matches the SUR likelihood with $\boldsymbol{\theta}$ redefined to treat $\mathbf{y}$ on the right as if it is predetermined.

Two more examples illustrate why an ML SUR estimator is inconsistent when using equation systems that are not fully observed or not recursive.

Example 4. *Two-stage probit with a latent determinant*

We make a small typographic, but important econometric, change to the previous example: we add a star to y_1 in the y_2 equation:

$$y_1^* = \theta_1 + \varepsilon_1$$

$$y_2^* = \theta_2 + \varepsilon_2$$

$$\theta_1 = \beta_1 x$$

$$\theta_2 = \beta_2 x + \delta y_1^*.$$

Even though y_1^* cannot be observed, the model can be fitted. Given observations of y_1, y_2, x_1, x_2, and the parameters, only values for ε_1 and ε_2 in a certain region are possible, so the likelihood is the total probability over that region. The region, however, is not Cartesian.

Suppose once again that for some i, we observe $\mathbf{y}_i = (0,0)'$. As before, the feasible range for ε_{i1} is

20

$(-\infty, -\theta_{i1}]$. Given a realization of ε_{i1} in this region, and given that $y_{i2} = 0$, the model tells us that

$$\beta_2 x_i + \delta y_{i1}^* + \varepsilon_{2i} \leq 0$$

$$\Rightarrow \beta_2 x_i + \delta (\beta_1 x_i + \varepsilon_{i1}) + \varepsilon_{i2} \leq 0$$

$$\Rightarrow \varepsilon_{i2} \in (-\infty, -\beta_2 x - \delta (\beta_1 x_i + \varepsilon_{i1})]$$

The term involving δ is new relative to Example 3, and it makes the feasible range for ε_{i2} depend on ε_{i1}, unlike before. Geometrically, it sheers the quarter-plane of integration for the likelihood. As a result, the argument of the previous example does not go through. A somewhat different mathematical structure is needed to represent this likelihood and fit the model.

In fact, the standard approach to simultaneous equations can extend the techniques already developed to estimate Example 4. Write the general recursive model (not fully observed) as

$$\underset{1 \times J}{\mathbf{y}^{*\prime}} \underset{J \times J}{\mathbf{\Gamma}} = \underset{1 \times J}{\boldsymbol{\theta}'} + \underset{1 \times J}{\boldsymbol{\varepsilon}'}$$

$$\underset{1 \times J}{\boldsymbol{\theta}'} = \underset{1 \times J}{\mathbf{y}'} \underset{J \times J}{\boldsymbol{\Delta}} + \underset{1 \times K}{\mathbf{x}'} \underset{K \times J}{\mathbf{B}}$$

$$\mathrm{E}[\varepsilon | \mathbf{x}] = \mathbf{0}$$

$$\varepsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$$

$$\mathbf{y} = \mathbf{g}(\mathbf{y}^*) = [\mathbf{g_1}(\mathbf{y}^*) \dots \mathbf{g_J}(\mathbf{y}^*)]'. \tag{25}$$

where $\mathbf{\Gamma}$ is upper triangular with, for normalization, 1's on the diagonal. Latent and observed variables appear separately here: only truly censored variables go in $\mathbf{y}$. Solving for $\mathbf{y}^*$ in the first equation,

$$\mathbf{y}^{*\prime} = \mathbf{y}' \boldsymbol{\Delta} \mathbf{\Gamma}^{-1} + \mathbf{x}' \mathbf{B} \mathbf{\Gamma}^{-1} + \varepsilon' \mathbf{\Gamma}^{-1}$$

$$\equiv \mathbf{y}' \mathbf{\Pi} + \mathbf{x}' \mathbf{C} + \mathbf{v}' \tag{26}$$

where $\mathbf{\Pi}$, $\mathbf{C}$, and $\mathbf{v}$ are defined as implied, meaning, in particular, that $\mathbf{v} \sim \mathcal{N}(\mathbf{0}; \boldsymbol{\Omega})$ where $\boldsymbol{\Omega} = \mathbf{\Gamma}^{-1\prime} \boldsymbol{\Sigma} \mathbf{\Gamma}^{-1}$. Because $\boldsymbol{\Delta}$ is strictly upper triangular and $\mathbf{\Gamma}^{-1}$ is upper triangular, $\mathbf{\Pi} = \boldsymbol{\Delta}\mathbf{\Gamma}^{-1}$ is strictly upper triangular too. Thus the last line *does* have the form of a fully observed recursive system, and can be estimated consistently with ML SUR. To compute the likelihood, the distribution function $f_{\mathbf{v}}()$ would be integrated

over some Cartesian region C whose bounds would depend on $\mathbf{\Gamma}$, $\mathbf{\Delta}$, and $\mathbf{B}$.[5]

Example 5. *Classical simultaneous equations*

This model features true simultaneous causation, but no censoring:

$$y_1^* = \theta_1 + \varepsilon_1$$

$$y_2^* = \theta_2 + \varepsilon_2$$

$$\theta_1 = \beta_1 x + \gamma_1 y_2$$

$$\theta_2 = \beta_2 x + \gamma_2 y_1$$

$$\mathbf{y} = \mathbf{g}\left(\mathbf{y}^*\right) = (y_1^*, y_2^*)'$$

In this case the equation-by-equation approach used in the previous examples to state the likelihood breaks down. If we start with equation 1, we need to write down the likelihood of a realization ε_1 given the observed values of both x and y_2: $\Pr\left(\varepsilon_{i1} \,|x_i, y_{i2}\right)$. However, because y_2 depends on ε_2, which is in general correlated with ε_1, conditioning on y_{i2} complicates the probability. It is not the same as $\Pr\left(\varepsilon_{i1} \,|x_i\right)$. Much as in the previous example, standard simultaneous equation techniques are needed, which solve out for y_1 and y_2 on the left, incorporate exclusion restrictions, and apply LIML or some other method to the resulting recursive set of equations.

2.2.4 Conditional modeling

The setting for almost all the theoretical discussion so far has been a single observation. This focus is deliberate, for it leaves open the possibility that the model can vary by observation, i.e., depend on the data. A model that is conditional on the data can seem strange to minds accustomed to the rigidity of OLS, 2SLS, and other GMM-class estimators. But it is theoretically possible in ML, and practically useful. *Parameters* cannot vary so freely (or they might not be identified), but *a priori* choices of model structure, such as the number of equations, the form of their link functions, and the location of known truncation and censoring points, can. For example, in an evaluation of a worker retraining program, an equation for the determinants of uptake could be dropped for observations from cities where the program was not offered. The consistency

---

[5]Because multiplication by $\mathbf{\Gamma}$ is a transformation whose Jacobian has determinant 1 ($|\mathbf{\Gamma}| = 1$), $f_{\mathbf{v}}\left(\mathbf{v}\right) = f_{\mathbf{v}}\left(\boldsymbol{\varepsilon}'\mathbf{\Gamma}^{-1}\right) = f_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\varepsilon}\right)$ (Ruud 2000, p. 726). So the likelihood is also the integral of $f_{\boldsymbol{\varepsilon}}\left(\right)$ over $\mathbf{\Gamma}C$. But this integral is impractical since $\mathbf{\Gamma}C$ is not Cartesian, just as in Example 4.

of SUR likelihoods for fully observed recursive systems is unaffected by this generalization.

Two examples of conditional modeling deserve special mention. One is the switching regression; it can incorporate two or more models for the same dependent variable, with the data determining which one applies to which observations. It can be viewed as a system of equations whose samples do not overlap. The other example is selection modeling. The classical Heckman selection model is like that of Example 2, except that y_1, the variable of interest, is only modeled when y_2, the dummy indicating whether the observation is complete, is 1. For complete observations, the likelihood is (21). For incomplete observations, the likelihood is just that for a one-equation probit model. The overall likelihood is the product of the observation-level ones.

These two ideas can be combined: a variable that drives a switching process can itself be modeled with a "selection" equation. An ordered categorical variable, for example, can be modeled as ordered probit while determining which of several models for a variable of interest apply, as in the `oheckman` command (Chiburis and Lokshin 2007).

All of these examples can be seen as instances of a singel, general modeling framework. Interestingly, Heckman's (1976) seminal paper on selection modeling is entitled "The common structure of statistical models of truncation, sample selection, and limited dependent variables and a simple estimator for such models."

2.2.5 Heteroskedasticity and consistency

One virtue of linear methods such as OLS and 2SLS is that heteroskedasticity harms only their efficiency, not their consistency. Heteroskedasticity does render the classical formulas for the standard errors of these estimators inconsistent, but several methods are available for correcting that problem, including bootstrapping and "robust" sandwich-type formulas.

Heteroskedasticity is a more serious threat to the limited dependent variable models considered here. The problem can be explained both graphically and analytically (Deaton 1997, pp. 85–89). Suppose we model hours worked outside the home, and that for those who do work it is in expectation a linear function of education with slope 1.Suppose that errors are normal so that the Tobit model in (6) is correct—except that σ^2, the variance of the error term, is not constant, and is instead convexly, positively related to education. So high is the variance for highly educated people, we assume, that they are particularly likely to have extreme employment propensities (y^*), whether positive and negative. The disproportionate censoring of the *negative* values for highly educated people will increase their apparent tendency to work and bias upward the estimated

slope of the relationship for uncensored observations. Figure 1 illustrates. The values of y^* are plotted in blue and censored values observations of y in purple. The blue line segments show the true regression model, $\mathrm{E}\left[y\left|x\right.\right] = x \cdot \mathbf{1}\left\{x > 0\right\}$. But high variance in the errors on the right end of the graphed range (according to $\sigma = 1 + \frac{(x+25)^{1.8}}{50}$) generates a cluster of large, negative values for y^*. These are censored upward, to 0, while the high positive values of y^* in the same region are not symmetrically censored downward. As a result, their presence steepens the best-fit line for uncensored observations (purple line), making the Tobit fit inconsistent.

More formally, it can be checked that the classical linear regression likelihood (1) has the property that when its first derivatives are 0 (at an optimum) the second-order cross derivative between $\boldsymbol{\beta}$ and σ^2 is too. So, at an optimum, an infinitesimal change in the best-fit value for σ^2 does not perturb the first derivative of the likelihood with respect to $\boldsymbol{\beta}$ from its value of 0. The first-order condition for $\boldsymbol{\beta}$ being an optimal value remains satisfied. In this sense, the ML estimate of $\boldsymbol{\beta}$ and that of σ^2 are independent; and the first remains consistent even when the latter, under heteroskedasticity, is not. In contrast, the Tobit likelihood in (7), with its novel term involving a cumulative normal density, lacks this property.

Technically, heteroskedasticity can also afflict more-completely censored models, such as probit. But here the problem is best thought of differently, and is perhaps less of a practical concern. Consider, as Deaton suggests, a probit model like that in (9) except with heteroskedasticity that happens to take the peculiar form $\sigma = \mathbf{x}'\boldsymbol{\beta}/\mathbf{x}'\boldsymbol{\gamma}$, where $\boldsymbol{\gamma}$ is a coefficient vector. The likelihood for an observation with $y_i = 1$ would then be

$$\Phi\left(\mathbf{x}_i'\boldsymbol{\beta}; \sigma^2\right) = \Phi\left(\frac{\mathbf{x}_i'\boldsymbol{\beta}}{\sigma}; 1\right) = \Phi\left(\frac{\mathbf{x}_i'\boldsymbol{\beta}}{\mathbf{x}_i'\boldsymbol{\beta}/\mathbf{x}_i'\boldsymbol{\gamma}}; 1\right) = \Phi\left(\mathbf{x}_i'\boldsymbol{\gamma}; 1\right)$$

Probit estimates of $\boldsymbol{\beta}$ would be consistent for $\boldsymbol{\gamma}$ instead! This again shows the inseparability of the location and dispersion parameters ($\boldsymbol{\beta}$ and σ^2) in limited dependent variable models.

The problem needs to be viewed in larger perspective, however (Wooldridge 2002, pp. 479–80). The model for y^* is a mathematical convenience, a hypothetical equation for a hypothetical variable. Consider that if the true process for y^* is that implied by the logit model then probit is technically inconsistent, and vice versa. Probably in most cases, neither model is perfectly correct, yet both are useful. From this point of view, heteroskedasticity is just one of possible deviations of the probit model from reality, one more potential imperfection in a chosen functional form. Meanwhile, because censoring in the probit model is more symmetric than in the censored-from-below Tobit model, the potential for systematic bias may be smaller.

Note two points the foregoing does *not* imply. First, correlations in errors across observations do not cause the same trouble. For consistency, errors need to be identically, but not independently, distributed. Second, moving to the multi-equation context, heteroskedasticity in one equation does not necessarily render coefficient estimates for other equations inconsistent. For example, if the heteroskedasticity is confined to the reduced-form equations in a LIML estimation set-up, the ones not required to be structurally correct, this may not harm the consistency of the parameter estimates for the structural equations. Rather, modeling a reduced-form heteroskedastic error term as homoskedastic would be one more example of "limited information" about the true model.[6] This cross-equation "non-interference" of heteroskedasticty would only be useful, though, when a researcher believed that the error process for one equation was heteroskedastic while a (correlated) error process for another was not. This assumption might appear arbitrary, making it hard to defend.

### 2.2.6   Logical consistency and identification

The conditions for the consistency of ML SUR for simultaneous equations—recursivity and full observability—are less strict than they appear, in the sense that many models that one could write down that violate one or both restrictions are in fact logically impossible (Maddala and Lee 1976; Heckman 1978). For example, a fully observed multivariate probit model *must* be recursive to be logically consistent (Schmidt 1981). A simple example of an impossible model is:

$$y_1^* = \gamma_1 y_2 + \varepsilon_1$$

$$y_2 = \gamma_2 y_1 + \varepsilon_2$$

$$y_1 = \mathbf{1}\left\{y_1^* > 0\right\}$$

$$\mathrm{E}[\boldsymbol{\varepsilon}|\mathbf{x}] = \mathbf{0}$$

$$\boldsymbol{\varepsilon} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Sigma}\right)$$

---

[6]I have not worked out, nor seen worked out, the precise conditions under which heteroskedasticity in one equation affects the consistency of coefficient estimates for another.

Figure 1: Example of heteroskedasticity-induced inconsistency in Tobit model
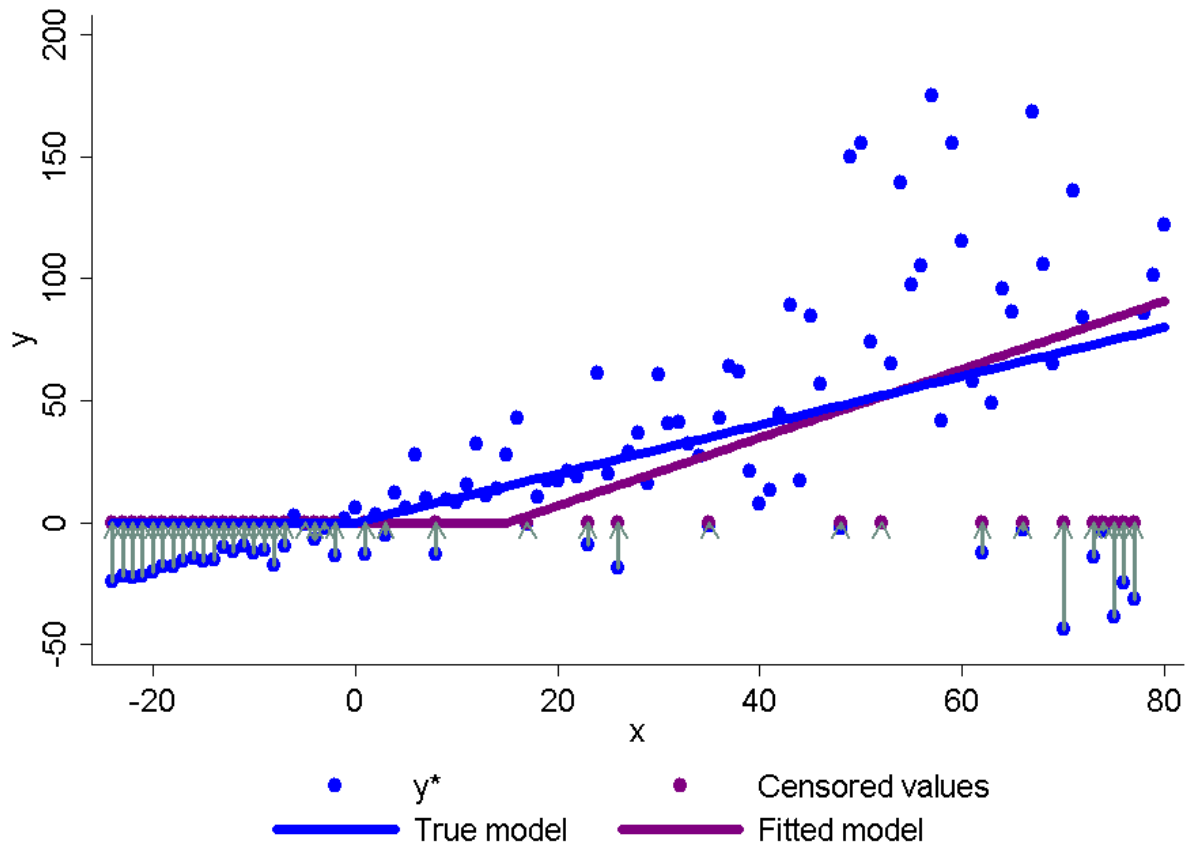
This mixed-process example is fully observed, but not recursive. Substituting for y_2 in the y_1^* equation gives $y_1^* = \gamma_1\gamma_2 y_1 + \gamma_1\varepsilon_2 + \varepsilon_1$. Combining this with the definition of y_1,

$$y_1 = 0 \text{ when } \gamma_1\varepsilon_2 + \varepsilon_1 \leq -\gamma_1\gamma_2 y_1 = 0$$

$$y_1 = 1 \text{ when } \gamma_1\varepsilon_2 + \varepsilon_1 > -\gamma_1\gamma_2 y_1 = -\gamma_1\gamma_2.$$

Thus, seemingly, depending upon the sign of $\gamma_1\gamma_2$, if $\gamma_1\varepsilon_2 + \varepsilon_1$ happens to be between 0 and $-\gamma_1\gamma_2$, then y_1 would equal both 0 and 1—or neither. The model is logically consistent only if $-\gamma_1\gamma_2 = 0$, i.e., if it is recursive.

The positive flip-side of the nonlinearity at work here is that multi-equation limited dependent variable models that are logically consistent often require fewer assumptions for formal identification than classical linear ones. For classical systems to be identified, a rank condition must be met. A common, though technically not quite sufficient, rule is the order condition: in each equation, at least one predetermined variable must be excluded for every endogenous one included (Greene 2000, p. 670). Surprisingly, such rules become less necessary as censoring introduces nonlinearities. For example, a fully observed multivariate probit model with unrestricted correlation error structure (which we just saw must be recursive) is in general identified without any further exclusion restrictions (Wilde 2000). Consider the two-stage probit model in Example 3. Notice that the two equations share a single predetermined regressor, x. There is no "instrument." The first-stage equation, a standard one-equation probit, is clearly identified. Wilde points out that the concern with regard to identification of the second is that some non-trivial linear combination of the two latent variables, $\lambda_1 y_1^* + \lambda_2 y_2^*, \lambda_1 \neq 0$, contains the same set of variables as the structural y_2^* equation—that would be a sign of underidentification in an uncensored system. By the definition of the model, this linear combination is

$$\lambda_1 y_1^* + \lambda_2 y_2^* = \lambda_2\gamma y_1 + \lambda_1\beta_1 x + \lambda_2\beta_2 x + \lambda_1\varepsilon_1 + \lambda_2\varepsilon_2.$$

Thus,

$$y_2^* = \gamma y_1 - \frac{\lambda_1}{\lambda_2} y_1^* + \frac{\lambda_1}{\lambda_2}\beta_1 x + \beta_2 x + \frac{\lambda_1}{\lambda_2}\varepsilon_1 + \varepsilon_2.$$

The first right-hand term is the same as in the structural equation, but it is followed by a new term containing y_1^*, so the equation as a whole is not redundant with the structural equation.

27

Wilde (2000) shows that a general (recursive) multi-equation probit model is identified as long as each equation contains one varying predetermined variable. This is akin the result cited in section 2.1.7 that the unrestricted multinomial probit model is identified as long as one variable varies across alternatives. And here too, despite the theoretical results, identification might still be more robust if exclusion restrictions are imposed—if the classical order condition is met.

3   Estimation

The econometric literature on mixed-process models historically focused on multi-stage estimation procedures that are less computationally demanding than ML, if less efficient (e.g., Amemiya 1974; Heckman 1976; Maddala 1983, chs. 7–8; Smith and Blundell 1986; Rivers and Vuong 1988). This is one reason I have not found an encompassing discussion of ML estimation like that here. But faster computers have made direct ML fitting more practical. In particular, Monte Carlo-type simulated likelihood methods now facilitate estimation of integrals of the multivariate normal distributions of dimension 3 and higher (Train 2003).

Given a general likelihood-maximizing tool such as Stata's `ml`, the business of estimating parameters in the models described in this paper boils down to writing a program to compute the log of the likelihood (17) for each observation (and, optionally for speed, its first and even second derivatives). In general, an observation of **y** might not be censored in all dimensions, as in Example 2, so that the dimensionality of the integrals may be lower than the number of equations. These likelihoods are most practically calculated just as in that example, by factoring the uncensored dimensions out of the overall distribution. In particular, if we order equations to put the uncensored observations before the censored ones and partition $\tilde{\boldsymbol{\varepsilon}}$ and $\tilde{\boldsymbol{\Sigma}}$ accordingly, the numerator of the likelihood (17) can be calculated as

$$\phi\left(\tilde{\boldsymbol{\varepsilon}}_1; \tilde{\boldsymbol{\Sigma}}_{11}\right) \int\limits_C \phi\left(\tilde{\boldsymbol{\varepsilon}}_2 - \tilde{\boldsymbol{\Sigma}}_{21}\tilde{\boldsymbol{\Sigma}}_{11}^{-1}\tilde{\boldsymbol{\varepsilon}}_1; \tilde{\boldsymbol{\Sigma}}_{22} - \tilde{\boldsymbol{\Sigma}}_{21}\tilde{\boldsymbol{\Sigma}}_{11}^{-1}\boldsymbol{\Sigma}_{12}\right) d\tilde{\boldsymbol{\varepsilon}}_2.$$

where C is the Cartesian region of feasible values for $\tilde{\boldsymbol{\varepsilon}}_2$. Appendix A sets forth formulas for practical computation of the log likelihood and its first derivatives.

`cmp` works like most ML estimation programs in Stata (Gould, Pitblado, and Sribney 2006). A front end processes the command line and prepares for estimation by Stata's built-in `ml` command. To choose a promising starting point for the search, it first estimates each equation separately. It also performs several specification checks to improve the odds of convergence. For example, it drops collinear regressors, and

detects whether any equations have non-overlapping samples, which should force the correlation parameter between their errors out of the model.

A separate program, also written in Stata's ado language and called repeatedly by `ml`, computes the likelihood associated with a provided set of trial parameter values. This evaluator in turn calls a Mata program to perform most of the computations. To make results easier to interpret, `cmp` represents Σ in "sigma-rho" form, that is, with a standard deviation (σ) parameter for each error and a correlation coefficient (ρ) for each pair. Since these parameters are bounded, they are transformed onto an unbounded scale by using the logarithm of the σ's and the arc-hyperbolic tangents (inverse S-curve transforms) of the ρ's. Fitting with "lnsig" and "atanhrho" parameters eliminates the possibility that in the course of its search, `ml` will submit impossible trial values for the parameters, such as a negative value for a σ.

Two aspects of `cmp`'s workings are novel enough to warrant discussion: the implementation of the Geweke, Hajivassiliou, and Keane (GHK) algorithm to compute higher-dimensional cumulative normal distributions, and the form of the likelihood evaluator. Both are designed to speed up `cmp`. Because ML estimation is computationally expensive, speed increases practicality.

3.1   ghk2()

For models in which three or more equations are censored at once for some observations, cumulative normal densities of dimension 3 or higher must be estimated. This is not a trivial problem. For explanations of the dominant approach to this problem, the GHK algorithm (Geweke 1989; Hajivassiliou and McFadden 1998; Keane 1994), see Greene (2000, pp. 183–85), Cappellari and Jenkins (2003), and Gates (2006). The GHK algorithm estimates the cumulative probability with a Monte Carlo technique, taking a number of draws from the unit interval. The draws can come from a pseudorandom sequence, which is designed to minimize correlation between successive entries. Or they can come from Halton or Hammersley sequences; these are designed to maximize uniformity of coverage over the unit interval (Drukker and Gates 2006). ("Generalized Halton" sequences can be seen as Halton sequences with pseudorandom starting points.) Stata 9 shipped with the Mata function `ghk()`, which computes the necessary draws every time it is called. Stata 10's `ghkfast()` allows the user to pre-compute the draws once, for speed.

For implementation of `cmp`, these built-in functions have several disadvantages. The Mata function `ghk2()`, which `cmp` requires, addresses the limitations (though not without introducing some disadvantages of its own). The major differences between `ghk2()` and the built-in functions are:

29

1. `ghk2()` accepts lower as well as upper bounds for integration. This allows efficient estimation of cumulative probabilities over bounded rectilinear regions such as $[\underline{c}_1, \bar{c}_1] \times [\underline{c}_2, \bar{c}_2]$, which arise in multi-equation ordered probit models. Without this ability, the routine would need to be called 2^d times, where d is the dimension of the integral. For example, an integral of $\phi\left(\cdot; \boldsymbol{\Sigma}\right)$ over the rectangle above would have to be computed as $\Phi\left((\bar{c}_1, \bar{c}_2)'; \boldsymbol{\Sigma}\right) - \Phi\left((\bar{c}_1, \underline{c}_2)'; \boldsymbol{\Sigma}\right) - \Phi\left((\underline{c}_1, \bar{c}_2)'; \boldsymbol{\Sigma}\right) + \Phi\left((\underline{c}_1, \underline{c}_2)'; \boldsymbol{\Sigma}\right)$.[7]

2. `ghk2()` does not "pivot" the bounds of integration. On the recommendation of Genz (1992), `ghk()` and `ghkfast()`, at least by default, reorder each vector of bounds to put the larger entries toward the end, which turns out to reduce the variability of the simulated probability. However, pivoting has the disadvantage of creating discontinuities in results. Small changes in the bounds that shuffle their rank ordering—e.g., from $(.999, 1.000)'$ to $(1.001, 1.000)'$—can produce relatively large changes in return values. Especially when the number of draws is low and the approximations coarse, these discontinuities can stymie a maximum likelihood search algorithm. Thus `ghk2()` behaves smoothly even at low draw counts, at the expense of some precision. After `ghk2()` was released, Stata Corporation added an option to `ghk()` and `ghkfast()` to turn off pivoting, in Stata 10.1.

3. `ghk2()` is optimized for contexts with a large number of observations relative to draws per observation. In extreme cases, such as 10,000 observations and 10 draws per observation, it can perform an order of magnitude faster than `ghkfast()`. (At least in single-processor versions of Stata: since `ghk2()` is written Mata it does not benefit from multiple processors in Stata/MP.) But at the opposite extreme, with, say, 100 observations and 1,000 draws, it can run half as fast.

Taking lower as well as upper bounds complicates the computation of the simulated probability as well as its derivatives with respect to the parameters—which `ghk2()`, like the built-in functions, optionally provides. Appendix B lays out the relevant formulas.

3.2 A new `ml` evaluator type

As mentioned, Stata's `ml` performs the maximum likelihood search for `cmp`, calling on a `cmp` subprogram to calculate the log likelihood for each trial parameter vector. `ml` accepts several types of likelihood evaluation routines. The simplest from the programmer's point of view is the `lf` method, so-called because it is appropriate for likelihoods, like those here, that satisfy the linear form restriction: the overall log likelihood

---

[7]Actually, the built-in `biprobit()` function would compute these four two-dimensional integrals much faster than a GHK implementaiton. The real utility is for higher-dimensional integrals.

is the sum of observation-level log likelihoods, ones that can be computed using only a given observation's data. (As a counterexample, random effects models do not have linear form because their formulas irreducibly involve clusters of observations.) `ml` provides an `lf` evaluator with trial values for $\theta_j = \mathbf{x}_j{}'\boldsymbol{\beta}_j$ for each linear component of a model, as well as for ancillary parameters such as elements of $\boldsymbol{\Sigma}$. The `lf` evaluator calculates only the log likelihood; `ml` computes the first and second derivatives numerically as needed for the search, via repeated calls to the evaluator. Numerical calculation of deriviates is usually slower than analytical calculation. But `ml` at least partly compensates for this computational inefficiency by exploiting the linear nature of the θ_j parameters. Since

$$\frac{\partial \ln L_i}{\partial \boldsymbol{\beta}_j} = \frac{\partial \ln L_i}{\partial \theta_j} \frac{\partial \theta_j}{\partial \boldsymbol{\beta}_j} = \frac{\partial \ln L_i}{\partial \theta_j} \mathbf{x}_j',$$

to the vector $\dfrac{\partial \ln L_i}{\partial \boldsymbol{\beta}_j}$, `ml` need only use the evaluator to calculate the scalar $\dfrac{\partial \ln L_i}{\partial \theta_j}$, which typically requires two calls to it (at the trial point and at that point plus or minus some small h along the equation's dimension). That is, `ml` calls the evaluator twice for each $\boldsymbol{\beta}_j$ rather than twice for each element of each $\boldsymbol{\beta}_j$. Similarly, the number of calls to compute the Hessian of the likelihood is quadratic in the number of linear components (plus ancillary parameters) rather than in the full number of parameters (Gould, Pitblado, and Sribney 2006, pp. 63–64). (Confusingly, this is *not* why the method is called "linear form.")

`ml` also accepts `d0`-, `d1`-, and `d2`-method evaluators. These do not assume linear form and they do not perform the trick just described to economize on numerical computation of derivatives. They do, however, allow the evaluator to compute these derivatives analytically. `d1` evaluators calculate first derivatives while `d2` evaluators do those and the Hessian. The odd thing about this arrangement is that moving from `lf` to `d1` imposes two independent changes on the programmer: ability to provide analytical first derivatives; and loss of the clever method of computing the Hessian that minimizes calls to the evaluator. Yet since these two changes are independent, the trade-off they create is theoretically unnecessary for linear-form likelihoods and could be avoided by a new evaluator type that blends the advantages of `lf` and `d1`. The trade-off affects the performance of `cmp` because its likelihoods are in linear form and it computes first, but not second, derivatives analytically.

Richard Gates and Jeffrey Pitblado of Stata Corporation suggested a work-around: write a "pseudo-`d2`" evaluator that takes full control of the process for computing first and second derivatives, and thus can avoid the trade-off. A pseudo-`d2` evaluator computes first derivatives analytically, and second derivatives numerically, but with two calls to the evaluation code per linear component, rather than per parameter. That

is how `cmp` works by default. Tests with one real-world example, a replication of the headline regression in Pitt and Khandker (1998), showed "pseudo-`d2`" with `ml`'s Newton-Raphson search method running twice as fast as the next-best alternative, `d1` with a Davidon-Fletcher-Powell search.

4 Using `cmp`

4.1 Syntax for `cmp`

`cmp` runs in Stata 9.2 and later. The syntax is

`cmp` *eq* $\big[$ *eq* ... $\big]$ $\big[$ *if* $\big]$ $\big[$ *in* $\big]$ $\big[$ *weight* $\big]$ `indicators(`*exp* $\big[$ *exp* ... $\big]$`)` $\big[$`lf nolrtest quietly`
  `ghkdraws(#) ghktype(`*string*`) ghkanti nodrop level(#)` *ml_opts* `svy` *svy_opts* `interactive`
  `init(`*vector*`) structural psampling(# #)`$\big]$

Each *eq* is an equation to be estimated, defined according to the `ml model` *eq* syntax. That is, the equation is enclosed in parentheses, optionally prefixed with a name for the equation:

`(`$\big[$ *eqname*`:` $\big]$ *varname_y* $\big[$ *varname_y2* $\big]$ `=` $\big[$ *indepvars* $\big]$ $\big[$`,` `noconstant offset(`*varname_o)*
  `exposure(`*varname_e*`) truncpoints(`*exp exp*`)` $\big]$`)`

varname_y2 is included only for interval-censored data, in a syntax analogous to that of `intreg`: *varname_y* and *varname_y2* hold the lower and upper bounds of each interval. `truncpoints(`*exp exp*`)` is included only for truncated regressions, and specifies any lower and upper truncation points as constants, variable names, or expressions. Missing values in these variables or expressions (".") are interpreted as $-\infty$ or ∞ as appropriate.

Since `cmp` is built on `ml`, it accepts most of the options that `ml model` accepts in its non-interactive mode. See [R] **ml**. It accepts the `svy` prefix, all weight types, `constraints()`, the search `technique()` option, and the full variety of standard error types (`robust`, `cluster()`, `vce(bootstrap)`, etc.) Note that if the errors are not believed to be i.i.d. (and yet the estimation is constistent, under circumstances discussed at the end of section 2.2.5), then the `cmp` likelihood is not fully accurate. The likelihood maximized is then a *pseudolikelihood*, and is labeled as such.

For estimates requiring the GHK algorithm, if a pseudorandom (`ghktype(random)`) or generalized Halton sequence (`ghktye(ghalton)`) is requested, the starting state of the Stata random number generator influences the values returned by `ghk2()`, thus those returned by `cmp`. For exact reproducibility of results with these sequences, initialize the seed to some chosen value with the `set seed` command before running `cmp`.

The required `indicators()` option is central to the use of `cmp`. Each *exp* in the option is an expression that evaluates to a `cmp` *indicator variable*, which communicates observation-level information about the

dependent variable(s). The option must include one indicator variable for each equation in the model, and each can be a constant, a variable name, or a mathematical expression. Expressions can contain spaces or parentheses if they are double-quoted. For each observation, each *exp* must evaluate to one of the following codes, with the meanings shown:

0 = observation is not in this equation's sample; i.e., equation does not apply to this observation
1 = observation is uncensored
2 = observation is left-censored
3 = observation is right-censored
4 = equation is probit for this observation
5 = equation is ordered probit for this observation
6 = equation is multinomial probit for this observation
7 = equation is interval-censored for this observation
8 = equation is truncated on the left and/or right for this observation

Notice that for a Tobit-modeled variable, it is the user's responsibility to determine and indicate in which observations the variable has been censored. The censoring point(s) can vary by observation. Also, as currently written, `cmp` treats truncated regression as a distinct model type, so truncation cannot be combined with censoring. One complication in the syntax relates to the specification of multinomial probit models; see section 4.5 below.

For clarity, users can execute the `cmp setup` subcommand, which defines global macros that can then be used in defining `cmp` indicators:

```
$cmp_out = 0
$cmp_cont = 1
$cmp_left = 2
$cmp_right = 3
$cmp_probit = 4
$cmp_oprobit = 5
$cmp_mprobit = 6
$cmp_int = 7
$cmp_trunc = 8
```

`cmp` leaves behind other global macros with names beginning with "cmp_", as well as some variables whose names begin with "_cmp" and some Mata globals that begin with "_". It must leave them in order to serve any subsequent requests for scores, such as are needed to compute a linearized covariance matrix in `svy` estimation. To remove these macros and variables, type `cmp clear`.

4.2   Options for `cmp`

<u>indicators</u>(*exp* [*exp ...*]) is required, as just described.
<u>level</u>(#) specifies the confidence level, in percent, for confidence intervals of the coefficients. The default is 95.

33

<u>nolrtest</u> suppresses calculation and reporting of the likelihood ratio (LR) test of overall model fit, relative to a constant(s)-only model. This has no effect if data are `pweight`ed or errors are `robust` or `cluster`ed. In those cases, the likelihood function does not reflect the non-sphericity of the errors, and so is a pseudolikelihood. The LR test is then invalid and not run anyway.

`lf` makes `cmp` use its `lf`-method evaluator instead of its pseudo-`d2` one. This is rarely needed.

<u>quietly</u> suppresses much of the output: the results from any single-equation initial fits and the iteration log during the full model fit.

<u>init</u>(*vector*) passes a row vector of user-chosen starting values for the model fit, in the manner of the `ml init, copy` command. The vector must contain exactly one element for each parameter `cmp` will estimate, and in the same order as `cmp` reports the parameter estimates in the output (excluding the displayed "sig" and "rho" results, which are merely transformed versions of the "lnsig" and "atanhrho" ones). The names of the row and columns of the vector do not matter.

<u>interactive</u> makes `cmp` fit the model in `ml`'s interactive mode. This allows the user to interrupt the model fit by hitting Ctrl-Break or its equivalent, then view and adjust the trial solution with such commands as `ml plot`, then restart optimization by typing `ml max`. `cmp` runs more slowly in interactive mode.

<u>ghkdraws</u>(#) sets the length of the sequence to draw for each observation in the GHK simulation of higher-dimensional cumulative multivariate normal distributions. The default is twice the square root of the number of observations for which the simulation is needed. (Cappellari and Jenkins (2003) suggest the square root.)

<u>ghktype</u>(*string*) specifies the type of sequence in the GHK simulation. Choices are "halton" (the default), "hammersley", "random", and "ghalton". See Drukker and Gates 2006; [M-5] **halton( )**.

<u>ghkanti</u> requests antithetical draws, which doubles the number of draws. See Drukker and Gates 2006; [M-5] **halton( )**.

<u>structural</u> requests the structural covariance parameterization for all multinomial equation groups, rather than the default differenced parameterization. See section 2.1.7.

<u>nodrop</u>: `cmp` starts by fitting each equation separately in order to obtain a good starting point for the full model fit. Sometimes in this preparatory step, convergence difficulties make a reported variance matrix singular, with missing standard errors for some regressors. Or variables can be found to be collinear. In order to maximize the chance of convergence, `cmp` ordinarily drops such regressors from the equations in which they cause trouble. <u>nodrop</u> prevents this behavior.

<u>psampling</u>(# #) makes `cmp` perform what I call "progressive sampling," which can speed estimation on large data sets. First it estimates on a small subsample, then a larger one, etc., until reaching the full sample. Each iteration uses the previous one's estimates as a starting point. The first argument in the option sets the initial sample size, either in absolute terms (if it is at least 1) or as a fraction of the full sample (if it is less than 1). The second argument is the factor by which the sample should grow in each iteration.

ml_opts: `cmp` accepts the following standard `ml` options: <u>trace</u>, <u>gradient</u>, <u>hessian</u>, <u>score</u>(*newvarlist*|*stub**), <u>technique</u>(*algorithm_specs*), <u>vce</u>(oim|opg|robust|cluster), <u>iterate</u>(#), <u>constraints</u>(clist), <u>tolerance</u>(#), <u>ltolerance</u>(#), <u>gtolerance</u>(#), <u>nrtolerance</u>(#), <u>nonrtolerance</u>, <u>shownrtolerance</u>, <u>showstep</u>, and <u>difficult</u>. See [R] **ml**.

<u>svy</u> indicates that `ml` is to pick up the `svy` settings set by `svyset` and use the robust variance estimator. This option requires the data to be `svyset`. `svy` may not be specified with `vce()` or weights. See [SVY] **svy: estat**.

svy_opts: Along with `svy`, users may also specify any of these `ml` options, which affect how the `svy`-based variance is estimated: <u>nosvyadjust</u>, <u>subpop</u>(*subpop_spec*), and <u>srssubpop</u>. And users may specify any of these `ml` options, which affect output display: <u>deff</u>, <u>deft</u>, <u>meff</u>, <u>meft</u>, <u>eform</u>, <u>prob</u>, and `ci`. See [SVY] **svy: estat**.

4.3 On `predict` and `mfx` after `cmp`

The syntax of `predict` following cmp is

`predict` $\big[\,type\,\big]$ `{`*newvarname*|*stub**|*newvarlist*`}` $\big[\,if\,\big]$ $\big[\,in\,\big]$ $\big[$, *statistic* <u>eq</u>`uation(`*eqno* $\big[$, *eqno*$\big]$`)`
    <u>o</u>`utcome(`*outcome*`)` <u>noo</u>`ffset`$\big]$

where *statistic* is `xb`, `pr`, `stdp`, `stddp`, <u>`scores`</u>, <u>`residuals`</u>, `e(`*a b*`)`, or `ystar(`*a b*`)`; and *a* and *b* may be numbers or variables, with any missing values interpret as infinite lower or upper "bounds." These options mean:

`xb`	linear prediction
`stdp`	standard error of linear prediction
`stddp`	standard error of difference in linear predictions
<u>`scores`</u>	derivative of the log likelihood with respect to a θ_j or an ancillary parameter
<u>`residuals`</u>	residuals
<u>`pr`</u>	probability of a positive outcome (meant for probit and ordered probit equations)
`e(# #)`	censored expected value (see [R] **regress postestimation**)
`ystar(# #)`	truncated expected value (see [R] **regress postestimation**)
<u>`eq`</u>`uation(`*eqno* $\big[$, *eqno*$\big]$`)`	specify equation(s)
<u>`o`</u>`utcome(`*outcome*`)`	specify outcome(s), for ordered probit only
`nooffset`	ignore any offset() or exposure() variable in the model

eqno can be an equation name. (If not set explicitly, an equation's name is that of its dependent variable.) Or it can be an equation number preceded by a `#`. Usually `predict` will default to reporting statistics just for equation `#1`. However, it will generate statistics for all equations if the provided variable list has one entry for each equation, or takes the form *stub**, either of which requests predictions for all equations, with names as given or as automatically generated beginning with *stub*.

In addition, for ordered probit equations, if `pr` is specified, `predict` will by default compute probability variables for all outcomes. The names for these variables will be automatically generated using a provided name as a stub. This stub may be directly provided in the command line—in which case it should *not* include a ***—or may itself be automatically generated by a cross-equation *stub**. Thus it is possible to generate probabilities for all outcomes in all ordered probit equations with a single, terse command. Alternatively, the `outcome(`*outcome*`)` option can be used to request probabilities for just one outcome. *outcome* can be a value for the dependent variable, or a category number preceded by a `#`. For example, if the categorical dependent variable takes the values 0, 3, and 4, then `outcome(4)` and `outcome(#3)` are synonyms. (`outcome()` also implies `pr`.)

In explaining the multi-equation and multi-outcome behavior of `predict` after `cmp`, examples are worth a thousand words: see section 4.4.3 below.

The flexibility of `cmp` affects the use of `predict` and `mfx` (for marginal effects) after estimation. Because the censoring type (probit, Tobit, etc.) can vary by observation, the default statistic for predict is always

xb, linear fitted values. So to obtain probabilities predicted by (ordered) probit equations, the user must include the `pr` option in the predict command line or `predict(pr)` in the `mfx` command line. (For ordered probit equations, `outcome()` option implies `pr`.) And sometimes `mfx`'s `nonlinear` and `force` options are required after `cmp`.

4.4 Examples

4.4.1 Replicating standard commands

The purpose of `cmp` is not to replicate existing commands, but to fit models that were previously much harder to estimate. Nevertheless, mimicking the familiar is a good way to illustrate how `cmp` works:

Set-up:

```
cmp setup

webuse laborsup, clear
replace fem_inc = fem_inc - 10^8
```

OLS:

```
reg kids fem_inc male_educ
cmp (kids = fem_inc male_educ), ind($cmp_cont) quietly
```

Iterated SUR for a linear system (see Pagan (1979) on the equivalence of linear iterated SUR and ML SUR):

```
sureg (kids = fem_inc male_educ) (fem_work = male_educ), isure
cmp (kids = fem_inc male_educ) (fem_work = male_educ), ind($cmp_cont $cmp_cont) quietly

mvreg fem_educ male_educ = kids other_inc fem_inc
cmp (fem_educ = kids other_inc fem_inc) (male_educ = kids other_inc fem_inc), ind($cmp_cont $cmp_cont) qui
```

Exactly identified linear two-stage (2SLS and LIML agree):

```
ivreg fem_work fem_inc (kids = male_educ), first

ivregress liml fem_work fem_inc (kids = male_educ), first
cmp (fem_work = kids fem_inc) (kids = fem_inc male_educ), ind($cmp_cont $cmp_cont) qui
```

Overidentified linear two-stage (2SLS and LIML differ):

```
ivreg fem_work fem_inc (kids = male_educ other_inc), first

ivregress liml fem_work fem_inc (kids = male_educ other_inc), first
cmp (fem_work = kids fem_inc) (kids = fem_inc male_educ other_inc), ind($cmp_cont $cmp_cont) qui
```

Probit:

```
probit kids fem_inc male_educ
predict p
mfx
cmp (kids = fem_inc male_educ), ind($cmp_probit) qui
predict p2, pr
```

---

[8]This line makes the censoring level 0 for `fem_inc`. This is needed to match pre-October 2007 versions of `ivtobit`, which, because of a bug, only work when censoring is at 0.

```
    mfx, predict(pr) nonlinear
```

Ordered probit:

```
    oprobit kids fem_inc male_educ
    mfx, predict(outcome(#2))
    cmp (kids = fem_inc male_educ), ind($cmp_oprobit) qui
    mfx, predict(pr outcome(#2)) nonlinear
```

Bivariate SUR probit:

```
    biprobit (kids = fem_inc male_educ) (fem_work = male_educ)
    cmp (kids = fem_inc male_educ) (fem_work = male_educ), ind($cmp_probit $cmp_probit)
```

"IV-probit": first stage uncensored, second stage probit:

```
    ivprobit fem_work fem_educ kids (other_inc = male_educ), first
    mfx, force predict(pr)
    cmp (other_inc = fem_educ kids male_educ) (fem_work = other_inc fem_educ kids), ind($cmp_cont $cmp_probit)
    mfx, force predict(pr)
```

Treatment effects model with endogenous treatment: first stage probit, second stage uncensored:

```
    treatreg other_inc fem_educ kids, treat(fem_work  = male_educ)
    cmp (fem_work  = male_educ) (other_inc = fem_educ kids fem_work), ind($cmp_probit $cmp_cont) qui
```

Tobit:

```
    tobit fem_inc kids male_educ, ll
    cmp (fem_inc = kids male_educ), ind("cond(fem_inc, $cmp_cont, $cmp_left)") qui
```

"IV-Tobit": first stage uncensored, second stage Tobit:

```
    ivtobit fem_inc kids (male_educ = other_inc), ll first
    cmp (male_educ=kids other_inc) (fem_inc=kids male_educ), ind($cmp_cont "cond(fem_inc,$cmp_cont,$cmp_left)")
```

Interval regression:

```
    webuse intregxmpl, clear
    intreg wage1 wage2 age age2 nev_mar rural school tenure
    cmp (wage1 wage2 = age age2 nev_mar rural school tenure), ind($cmp_int) qui
```

Truncated regression:

```
    webuse laborsub, clear
    truncreg whrs kl6 k618 wa we, ll(0)
    cmp (whrs = kl6 k618 wa we, trunc(0 .)), ind($cmp_trunc) qui
```

Heckman selection model:

```
    webuse womenwk, clear
    heckman wage education age, select(married children education age)
    gen byte selectvar = wage<.
    cmp (wage = education age) (selectvar = married children education age), ind(selectvar $cmp_probit) nolr qui
```

Probit with Heckman selection:

```
    gen byte wage2 = wage > 20 if wage < .
    heckprob wage2 education age, select(married children education age)
    cmp (wage2 = education age) (selectvar = married children education age), ind(selectvar*$cmp_probit $cmp_probit) qui
```

4.4.2   Going beyond standard commands

```
webuse laborsup, clear
```

Regress an unbounded, continuous variable on an instrumented, binary one. 2SLS is consistent but less efficient:

```
cmp (other_inc = fem_work) (fem_work = kids), ind($cmp_cont $cmp_probit) qui robust
ivreg other_inc (fem_work = kids), robust
```

Now regress on a left-censored variable, female income, which is only modeled for observations in which the woman works:

```
gen byte ind2 = cond(fem_work, cond(fem_inc, $cmp_cont, $cmp_left), $cmp_out)
cmp (other_inc=fem_inc kids) (fem_inc=fem_edu), ind($cmp_cont ind2)
```

"IV-oprobit":

```
cmp (fem_educ = fem_work) (kids = fem_educ), ind($cmp_cont $cmp_oprobit) nolr
```

Ordered probit with Heckman selection modeling:

```
webuse womenwk, clear
gen selectvar = wage<.
gen wage3 = (wage > 10)+(wage > 30) if wage < .
cmp (wage3 = education age) (selectvar = married children education age), ind(selectvar*$cmp_oprobit $cmp_probit) qui
```

4.4.3   `predict` after `cmp`

Set-up:

```
webuse laborsup, clear
```

Bivariate seemingly unrelated ordered probit:

```
gen byte kids2 = kids + int(uniform()*3)
cmp (kids=fem_educ) (kids2=fem_educ), ind($cmp_oprobit $cmp_oprobit) nolr tech(dfp) qui
```

Predict fitted values. Fitted values are always the default, as is equation #1:

```
predict xbA
```

Two ways to predict fitted values for all equations:

```
predict xbB*
predict xbC xbD
```

Predict scores for all equations and parameters:

```
predict sc*, score
```

Two ways to predict probability of `kids`=0, using (default) first equation:

```
predict prA, pr outcome(0)
predict prB, outcome(#1)
```

Predict `kids2`=4, using second equation:

```
predict prC, outcome(4) eq(kids2)
```

Predict all outcomes, all equations:

```
    predict prD*, pr
```

Same, but resulting variable names for the two equations start with `prE` and `prF` respectively:

```
    predict prE prF, pr
```

Predict all outcomes, equation 2. Generates variables `prG_#` where `#` is outcome number (not outcome value):

```
    predict prG, eq(#2) pr
```

4.5   On estimation with multinomial probit equations

The multinomial probit model is most complicated in theory and in implementation in `cmp`. Uniquely, it can be specified in `cmp` with two different command line syntaxes, roughly corresponding to the Stata commands `mprobit` and `asmprobit`. In the first syntax, the user lists a single equation, just as for other dependent variable types, and puts a 6 ($cmp_mprobit) in the `indicators()` list. The unordered categorical dependent variable holds the choice made in each case. Like `mprobit`, `cmp` then treats all regressors as case-specific, meaning that they are determinants of the attractiveness of all alternatives. More precisely, in order to estimate, `cmp` expands the specified equation into a group with one equation for each possible choice. All equations in this group include all regressors, except for the first, the designated base alternative (see section 2.1.7) which includes none. This alternative corresponds to the lowest value of the dependent variable. The next alternative, corresponding to the second-lowest value, is the "scale alternative," meaning that to normalize results, the variance of its error term is fixed. The value it is fixed at depends on whether the `structural` option is invoked, on which see below. Unlike `mprobit`, `cmp` does not automatically make the Independence of Irrelevant Alternatives assumption. I.e., it allows a general covariance structure rather than assuming the errors are i.i.d. IIA can be imposed with `cmp` through constraints.

As discussed in section 2.1.7 a general non-IIA multinomial probit model is technically identified as long as one variable varies across alternatives. However, this model is often slippery to fit, and excluding certain regressors from certain equations, or imposing cross-equation constraints, may produce more reliable convergence. Such exclusion restrictions can be impose in two ways with `cmp`. The first is to use the standard `constraints()` option.

The second is to use `cmp`'s other multinomial probit syntax. In this "alternative-specific" syntax, the user lists one equation in the `cmp` command line for each alternative, including the base alternative. Different equations may include different regressors. Unlike `asmprobit`, `cmp` does not force regressors that appear in more than one equation to have the same coefficient across alternatives, although again this restriction

can be imposed through constraints. When using the alternative-specific syntax, the dependent variables listed should be a set of dummies indicating which alternatives are chosen in each case, as can be generated with `xi, noomit` from the underlying polychotomous choice variable. The first equation is always treated as the base alternative, so the user can control which alternative is the base by reordering the equations. In general, regressors that appear in all other equations should be excluded from the base alternative. (`cmp` automatically excludes the constant.) Variables that are specific to the base alternative, however, or to a strict subset of alternatives, can be included in the base alternative equation.

To specify an alternative-specific multinomial probit group, the user includes expressions in the `indicators()` option that evaluate to 0 or 6 ($cmp_out or $cmp_mprobit) for each equation in the group (0 indicating that the choice is unavailable for given observations) and encloses the whole list in an additional set of parentheses. Unlike with `asmprobit`, there should be one row in the data set per case, not per case and alternative.

Section 2.1.7 explains the trade-off between two ways of parameterizing the covariance matrix of the errors. By default, `cmp` interprets the `lnsigma` and `atanhrho` parameters as characterizing these errors after differencing with respect to the base alternative. To eliminate an excessive degree of scaling freedom, it constrains the error variance of the second alternative's equation (the "scaling alternative") to 2, which it would be under the IIA, as in (15). If the `structural` option is invoked, the parameters are interpreted as describing the error covariances before differencing. In this case, to remove the excessive degrees of freedom, `cmp` constrains the base alternative error to have variance 1 and no correlation with the other errors; and constrains the error for the scaling alternative to also have variance 1.

Two examples illustrate. The first is of a multinomial probit with case-specific variables only and IIA assumed, run with both the differenced and structural parameterizations. In the structural parameterization, the covariance matrix is the identity matrix. In the differenced parameterization, it is as in (15):

```
webuse sysdsn3, clear
mprobit insure age male nonwhite site2 site3
```

Replicate first with structural parameterization: σ^2's are 1 and ρ's are 0, so lnsig's and atanhrho's are 0:

```
constraint 1 [lnsig_3]_cons
constraint 2 [atanhrho_23]_cons
cmp (insure = age male nonwhite site2 site3), nolr ind($cmp_mprobit) constraint(1 2) structural qui
```

Now with the differenced parameterization. IIA puts 2's on the diagonal and 1's off the diagonal, meaning

$\sigma_{11} = \sigma_{22} = 2$ and $\rho_{12} = \frac{\sigma_{12}}{\sqrt{\sigma_{11}\sigma_{22}}} = \frac{1}{2}$:

```
constraint 3 [atanhrho_23]_cons = '=atanh(1/2)'
constraint 4 [lnsig_3]_cons = '=ln(sqrt(2))'
cmp (insure = age male nonwhite site2 site3), nolr ind($cmp_mprobit) constraint(3 4) qui
```

The second example is of an alternative-specific multinomial probit with unconstrained covariance structure:

```
webuse travel, clear
asmprobit choice travelcost termtime, casevars(income) case(id) alternatives(mode) struct
```

To replicate, we reshape the data set to have one observation per case and impose cross-equation equality on alternative-specific variables:

```
drop invehiclecost traveltime partysize
reshape wide choice termtime travelcost, i(id) j(mode)
constraint 1 [air]termtime1 = [train]termtime2
constraint 2 [train]termtime2 = [bus]termtime3
constraint 3 [bus]termtime3 = [car]termtime4
constraint 4 [air]travelcost1 = [train]travelcost2
constraint 5 [train]travelcost2 = [bus]travelcost3
constraint 6 [bus]travelcost3 = [car]travelcost4
cmp (air:choice1=t*1) (train:choice2 = income t*2) (bus:choice3 = income t*3) (car:choice4 = income t*4),
        ind((6 6 6 6)) ghkanti ghkdraws(200) ghktype(hammersley) constraints(1/6) nodrop structural
        tech(dfp nr) nrtol(1e-4)
```

4.5.1 Tips for achieving and speeding convergence

These techniques can help `cmp` converge:

1. Changing the search method using the `technique()` option. The default Newton-Raphson (NR) method usually works well once `ml` has found a concave region. The Davidon-Fletcher-Powell (DFP) algorithm (`tech(dfp)`) often works better before then, and the two sometimes work very well in combination, as with `tech(dfp nr)`, which specifies that ml should switch between the two methods every five steps. See [R] **ml**.

41

2. Switching from the default pseudo-d2 evaluator to the `lf` evaluator, with the `lf` option, occasionally helps.

3. If the estimation problem requires the GHK algorithm, changing the number of draws per observation in the simulation sequence using the `ghkdraws()` and/or `ghkanti` options. Raising simulation accuracy by increasing the number of draws is sometimes necessary for convergence and can even speed it by improving search precision. On the other hand, especially when the number of observations is high, convergence can be achieved, at some loss in precision, with remarkably few draws per observations—as few as 5 when the sample size is 10,000 (Cappellari and Jenkins 2003). And taking more draws slows execution.

4. adding a `nrtolerance(#)` or `nonrtolerance` option to the command line if the search appears to be converging in likelihood—if the reported log likelihood is hardly changing in each iteration—and yet convergence is not declared. These are `ml` options. By default, `ml` declares convergence when the log likelihood is changing very little with successive iterations (within tolerances adjustable with the `tolerance(#)` and `ltolerance(#)` options) *and* when the calculated gradient vector is close enough to 0. In some difficult problems, such as ones with nearly collinear regressors, the imperfect precision of floating point numbers prevents `ml` from quite satisfying the second criterion. It can be loosened by using `nrtolerance(#)` to set the scaled gradient tolerance to a value larger than its default of 10^{-5}, or eliminated altogether with `nonrtolerance`.

5. Exploring with `cmp`'s interactive mode, described in section 4.2.

5   Conclusion

`cmp`'s estimation framework could be further developed. The requirement of full observability could be dropped, along the lines laid out in section 2.2.3. Random effects and random coefficients could be added, as could other model types; `gllamm` offers these features for single-equation models (Rabe-Hesketh, Skrondal, and Pickles 2002). Perhaps the approach can be generalized beyond recursive systems. Still, as it stands, `cmp` represents a significant new direction within the Stata universe. And beyond Stata, there appear to be few comparable tools. Faster computers and the simulated likelihood approach of the GHK algorithm are allowing practitioners to revisit models mostly developed in the late 1970s, applying direct ML estimation where it was once impractical.

References

[1] Amemiya, T. 1973. Regression analysis when the dependent variable is truncated normal. *Econometrica* 41(6): 997–1016.

[2] Amemiya, T. 1974. Multivariate regression and simultaneous equation models when the dependent variables are truncated normal. *Econometrica* 42(6): 999–1012.

[3] Bolduc, D. 1999. A practical technique to estimate multinomial probit models in transportation. *Transportation Research, Part B* 33: 63-79.

[4] Bunch, D.S. 1991. Estimability in the multinomial probit model. *Transportation Research* 25B(1): 1–12.

[5] Cappellari, L., and S. Jenkins. 2003. Multivariate probit regression using simulated maximum likelihood. *Stata Journal* 3(3): 278–94.

[6] Chiburis, R., and M. Lokshin. 2007. Maximum likelihood and twostep estimation of an orderedprobit selection model. *Stata Journal* 7(2): 167–82.

[7] Deaton, A. 1997. *The Analysis of Household Surveys: A Microeconometric Approach to Development Policy.* Baltimore: Johns Hopkins University Press.

[8] Drukker, D.M., and R. Gates. 2006. Generating Halton sequences using Mata. *Stata Journal* 6(2): 214–28.

[9] Gates, R. 2006. A Mata Geweke-Hajivassiliou-Keane multivariate normal simulator. *Stata Journal* 6(2): 190–213.

[10] Genz, A. 1992. Numerical computation of multivariate normal probabilities. *Journal of Computational and Graphical Statistics* 1: 141-49.

[11] Geweke, J. 1989. Bayesian inference in econometric models using Monte Carlo integration. *Econometrica* 57: 1317-39.

[12] Gould, W., J. Pitblado, and W. Sribney. 2006. *Maximum Likelihood Estimation with Stata.* 3rd ed. College Station, TX: Stata Press.

[13] Greene, W.H. 1998. Gender economics courses in liberal arts colleges: further results. *Research in Economic Education* 29(4): 291–300.

[14] Greene, W.H. 2000. *Econometric Analysis*, 4th ed. Upper Saddle River, NJ: Prentice-Hall.

[15] Hajivassiliou, V., and D. McFadden. 1998. The method of simulated scores for the estimation of LDV models. *Econometrica* 66: 863-96.

[16] Heckman, J.J. 1976. The common structure of statistical models of truncation, sample selection, and limited dependent variables and a simple estimator for such models. *Annals of Economic and Social Measurement* 5: 475–492.

[17] Heckman, J.J. 1978. Dummy endogenous variables in a simultaneous equation system. *Econometrica* 46(4): 931–59.

[18] Keane, M.P. 1992. A note on identification in the multinomial probit model. *Journal of Business and Economics Statistics* 10(2): 193–200.

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

[20] Kelejian, H.H. 1971. Two-stage least squares and econometric systems linear in parameters but nonlinear in the endogenous variables. *Journal of the American Statistical Association* 66(334): 373–74.

[21] Pagan, A. 1979. Some consequences of viewing LIML as an iterated Aitken estimator. *Economics Letters* 3:369–372.

[22] Pitt, M.M., and S.R. Khandker. 1998. The impact of group-based credit programs on poor households in Bangladesh: does the gender of participants matter? *Journal of Political Economy* 106(5): 958–96.

[23] Long, J.S., and J. Freese. 2006. *Regression Models for Categorical Dependent Variables Using Stata*. 2nd ed. College Station, TX: Stata Press.

[24] Maddala, G.S. 1983. *Limited-Dependent and Qualitative Variables in Econometrics* Cambridge, UK: Cambridge University Press.

[25] Maddala, G.S., and L. Lee 1976. Recursive models with qualitative endogenous variables. *Annals of Economic and Social Measurement* 5(4): 525–45.

[26] Rabe-Hesketh, S., A. Skrondal, and A. Pickles. 2002. Reliable estimation of generalized linear mixed models using adaptive quadrature. *Stata Journal* 2: 1-21.

[27] Rivers, D., and Q.H. Vuong. 1988. Limited information estimators and exogeneity tests for simultaneous probit models. *Journal of Econometrics* 39: 347–66.

44

[28] Ruud, P.A. 2000. *Classical Econometrics*. New York: Oxford University Press.

[29] Sajaia, Zurab. 2006. Maximum likelihood estimation of a bivariate ordered probit model: implementation and Monte Carlo simulations. Mimeo. World Bank. Washington, DC.

[30] Schmidt, P. 1981. Constrains on the parameters in simultaneous Tobit and probit models. In C.F. Manski and D.F. McFadden, eds., *Structural Analysis of Discrete Data and Econometric Applications*. Cambridge, MA: MIT Press.

[31] Smith, R.J., and R.W. Blundell. 1986. An exogeneity test for a simultaneous equation Tobit model with an application to labor supply. *Econometrica* 54(3): 679–85.

[32] Tobin, J. 1958. Estimation of relationships for limited dependent variables. *Econometrica* 26(1): 24–36

[33] Train, K. 2003. Discrete Choice Methods with Simulation. Cambridge: Cambridge University Press.

[34] Wilde, J. 2000. Identification of multiple equation probit models with endogenous dummy regressors. *Economics Letters* 69: 309–12.

[35] Wooldridge, J.M. 2002. *Econometric Analysis of Cross Section and Panel Data*. Cambridge, MA: MIT Press.

A   Likelihood and scores for fully observed mixed-process SUR

A.1   Practical computation of the likelihood

Recasting (16), (17), and (18), our task is to express the observation-level likelihood

$$
L_i \left(\mathbf{B}, \boldsymbol{\Sigma}, \mathbf{c}; \mathbf{y}_i | \mathbf{x}_i \right) = \frac{\displaystyle\int_C \phi \left(\boldsymbol{\varepsilon}; \boldsymbol{\Sigma} \right) d\boldsymbol{\varepsilon}}{\displaystyle\int_T \phi \left(\boldsymbol{\varepsilon}; \boldsymbol{\Sigma} \right) d\boldsymbol{\varepsilon}} \tag{27}
$$

$$
C = [\underline{c}_1, \bar{c}_1] \times \ldots \times [\underline{c}_J, \bar{c}_J]
$$

$$
T = [\underline{\tau}_1, \bar{\tau}_1] \times \ldots \times [\underline{\tau}_J, \bar{\tau}_J]
$$

in a computationally tractable form. In any given dimension, each region of integration will be finite or infinite or—in the numerator, for uncensored observations—infinitesimal.

Just as in Example 2, if we order equations to put the uncensored observations before the censored ones and partition $\boldsymbol{\varepsilon}$ and $\boldsymbol{\Sigma}$ accordingly, the numerator is

$$\phi\left(\boldsymbol{\varepsilon}_1; \boldsymbol{\Sigma}_{11}\right) \int_C \phi\left(\boldsymbol{\varepsilon}_2 - \boldsymbol{\mu}_{2|1}; \boldsymbol{\Sigma}_{2|1}\right) d\boldsymbol{\varepsilon}_2. \tag{28}$$

Let $\mathbf{T}$ be the Cholesky factorization of $\boldsymbol{\Sigma}_{22}$. Then $\boldsymbol{\Sigma}_{22}^{-1} = \mathbf{T}^{-1'}\mathbf{T}^{-1}$, and the first factor above, a multivariate normal density, can be rewritten in logs as

$$\ln\phi\left(\boldsymbol{\varepsilon}_1; \boldsymbol{\Sigma}_{11}\right) = -\frac{1}{2}\left(\ln|2\pi\boldsymbol{\Sigma}_{11}| + \boldsymbol{\varepsilon}_1'\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\varepsilon}_1\right) = -\frac{1}{2}\left(\ln|2\pi\mathbf{T}\mathbf{T}'| + \boldsymbol{\varepsilon}_1'\mathbf{T}^{-1'}\mathbf{T}^{-1}\boldsymbol{\varepsilon}_1\right)$$
$$= -\frac{1}{2}\left(\ln|2\pi| + \left(\mathbf{T}^{-1}\boldsymbol{\varepsilon}_1\right)'\mathbf{T}^{-1}\boldsymbol{\varepsilon}_1\right) - \ln|\mathbf{T}| = \ln\phi\left(\mathbf{T}^{-1}\boldsymbol{\varepsilon}_1; \mathbf{I}\right) - \ln|\mathbf{T}|$$

where $\phi\left(\cdot; \mathbf{I}\right)$ is a multivariate standard normal p.d.f. The jth entry of $\mathbf{T}^{-1}\boldsymbol{\varepsilon}_1$ is $\mathbf{T}_j^{-1}\boldsymbol{\varepsilon}_1$, where $\mathbf{T}_j^{-1}$ is the jth row of $\mathbf{T}^{-1}$. So the above is

$$\ln\phi\left(\boldsymbol{\varepsilon}_1; \boldsymbol{\Sigma}_{11}\right) = \left[\sum_j \ln\phi\left(\mathbf{T}_j^{-1}\boldsymbol{\varepsilon}_1; 1\right)\right] - \ln|\mathbf{T}|,$$

which can be calculated with standard functions in Mata.

The second term in (28), a cumulative probability, is fairly straightforward to compute if it has dimension 1 or 2. If it has dimension 1, it is

$$\int_C \phi\left(\boldsymbol{\varepsilon}_2 - \boldsymbol{\mu}_{2|1}; \boldsymbol{\Sigma}_{2|1}\right) d\boldsymbol{\varepsilon}_2 = \Phi\left(\frac{\bar{c}^J - \boldsymbol{\mu}_{2|1}}{\sqrt{\boldsymbol{\Sigma}_{2|1}}}\right) - \Phi\left(\frac{\underline{c}^J - \boldsymbol{\mu}_{2|1}}{\sqrt{\boldsymbol{\Sigma}_{2|1}}}\right).$$

If the dimension is 2, then let $\mathbf{V}$ be the 2×2 diagonal matrix that normalizes against the assumed variances, so that $\mathbf{R}_{2|1} = \mathbf{V}\boldsymbol{\Sigma}_{2|1}\mathbf{V}'$ is a correlation matrix. The term equals

$$\int_C \phi\left(\mathbf{V}\left(\boldsymbol{\varepsilon}_2 - \boldsymbol{\mu}_{2|1}\right); \mathbf{V}\boldsymbol{\Sigma}_{2|1}\mathbf{V}'\right) d\boldsymbol{\varepsilon}_2 = \int_C \phi\left(\mathbf{V}\left(\boldsymbol{\varepsilon}_2 - \boldsymbol{\mu}_{2|1}\right); \mathbf{R}_{2|1}\right) d\boldsymbol{\varepsilon}_2.$$

This can be computed with up to 1, 2, or 4 calls to Mata's `binormal()`, as described in section 3.1, depending on how many of the bounds of C are finite. (`binormal()` requires that the variables in the two dimensions have unit variance and accepts a single correlation parameter to characterize their covariance.)

If the cumulative probability has dimension 3 or higher, it is estimated with the GHK algorithm, as

described in Appendix B.

The cumulative probability over the truncation region in the denominator of (27) is computed in the same manner.

A.2 Scores

Since the log likelihood is

$$\ln L_i\left(\mathbf{B}, \boldsymbol{\Sigma}, \mathbf{c}; \mathbf{y}_i | \mathbf{x}_i\right) = \ln \phi\left(\boldsymbol{\varepsilon}_1; \boldsymbol{\Sigma}_{11}\right) + \ln \int_C \phi\left(\boldsymbol{\varepsilon}_2 - \boldsymbol{\mu}_{2|1}; \boldsymbol{\Sigma}_{2|1}\right) d\varepsilon_2 - \ln \int_T \phi\left(\boldsymbol{\varepsilon}; \boldsymbol{\Sigma}\right) d\boldsymbol{\varepsilon},$$

the challenge in computing scores lies mostly in computing derivatives of multivariate normal p.d.f.'s and c.d.f.'s, as well as of $\boldsymbol{\mu}_{2|1}$ and $\boldsymbol{\Sigma}_{2|1}$.

A final step is to translate scores with respect to $\boldsymbol{\varepsilon}$ into ones with respect to $\mathbf{B}$, and to move from derivatives with respect to $\tilde{\boldsymbol{\Sigma}}$ to ones with respect to $\boldsymbol{\Sigma}$, a distinction for multinomial probits that has so far been glossed over. (See section 4.5.) Since the final step involves only linear transformations, it is straightforward and not elaborated on here.

A.2.1 Scores of the multivariate normal p.d.f.

For a d-dimensional normal p.d.f.,

$$\frac{\partial \ln \phi\left(\varepsilon, \boldsymbol{\Sigma}\right)}{\partial \boldsymbol{\varepsilon}} = -\frac{1}{2}\frac{\partial}{\partial \boldsymbol{\varepsilon}}\left(\ln|2\pi\boldsymbol{\Sigma}| + \frac{1}{2}\varepsilon'\boldsymbol{\Sigma}^{-1}\varepsilon\right) = -\frac{1}{2}\left(2\varepsilon'\Sigma^{-1}\right) = -\varepsilon'\Sigma^{-1},$$

the penultimate step using the matrix identity $\partial\left(\mathbf{b}'\mathbf{A}\mathbf{b}\right)/\partial\mathbf{b} = 2\mathbf{b}'\mathbf{A}$.

To compute the derivative with respect to a lower-triangular element j, k of $\boldsymbol{\Sigma}$, we start by viewing all elements of $\boldsymbol{\Sigma}$ as independent, ignoring the necessity of symmetry. Proceeding as above,

$$\frac{\partial \ln \phi\left(\varepsilon, \boldsymbol{\Sigma}\right)}{\partial \boldsymbol{\Sigma}_{jk}} = -\frac{1}{2}\left(\frac{\partial \ln|\boldsymbol{\Sigma}|}{\partial \boldsymbol{\Sigma}_{jk}} + \varepsilon'\frac{\partial \boldsymbol{\Sigma}^{-1}}{\partial \boldsymbol{\Sigma}_{jk}}\varepsilon\right). \tag{29}$$

A formula for the derivative of the determinant turns the first term into

$$\frac{\partial \ln|\boldsymbol{\Sigma}|}{\partial \boldsymbol{\Sigma}_{jk}} = \frac{1}{|\boldsymbol{\Sigma}|}\frac{\partial |\boldsymbol{\Sigma}|}{\partial \boldsymbol{\Sigma}_{jk}} = \frac{1}{|\boldsymbol{\Sigma}|}|\boldsymbol{\Sigma}|\left(\boldsymbol{\Sigma}^{-1}\right)_{jk} = \boldsymbol{\Sigma}_{jk}^{-1},$$

meaning element j, k of $\boldsymbol{\Sigma}^{-1}$. Using a formula for the derivative of the matrix inverse, the second term of

47

(29) contains

$$\frac{\partial \mathbf{\Sigma}^{-1}}{\partial \mathbf{\Sigma}_{jk}} = -\mathbf{\Sigma}^{-1} \frac{\partial \mathbf{\Sigma}}{\partial \mathbf{\Sigma}_{jk}} \mathbf{\Sigma}^{-1} = -\mathbf{\Sigma}^{-1} \mathbf{S}_{jk} \mathbf{\Sigma}^{-1} = -\mathbf{\Sigma}_j^{-1'} \mathbf{\Sigma}_k^{-1}$$

where $\mathbf{S}_{jk}$ is all 0's except for a 1 in position j, k.

Putting all this together,

$$\frac{\partial \ln \phi\left(\boldsymbol{\varepsilon}, \mathbf{\Sigma}\right)}{\partial \mathbf{\Sigma}_{jk}} = -\frac{1}{2}\left(\mathbf{\Sigma}_{jk}^{-1} - \boldsymbol{\varepsilon}' \mathbf{\Sigma}_j^{-1'} \mathbf{\Sigma}_k^{-1} \boldsymbol{\varepsilon}\right) = \frac{1}{2}\left(\left(\mathbf{\Sigma}_j^{-1} \boldsymbol{\varepsilon}\right)\left(\mathbf{\Sigma}_k^{-1} \boldsymbol{\varepsilon}\right) - \mathbf{\Sigma}_{jk}^{-1}\right).$$

This is symmetric in j, k. So to account for the required symmetry of $\mathbf{\Sigma}$—$\mathbf{\Sigma}_{kj}$ moves in tandem with $\mathbf{\Sigma}_{jk}$—we merely double this quantity for off-diagonal entries of $\mathbf{\Sigma}$.

A.2.2   Scores of the multivariate normal c.d.f.

For a 1-dimensional distribution, the derivatives of the c.d.f. are

$$\frac{\partial \Phi\left(\varepsilon, \Sigma\right)}{\partial \varepsilon} = \phi\left(\varepsilon, \Sigma\right)$$

$$\frac{\partial \Phi\left(\varepsilon, \Sigma\right)}{\partial \Sigma} = \frac{\partial \Phi\left(\frac{\varepsilon}{\sqrt{\Sigma}}, 1\right)}{\partial \Sigma} = \phi\left(\frac{\varepsilon}{\sqrt{\Sigma}}, 1\right) \cdot \frac{\partial}{\partial \Sigma}\left(\frac{\varepsilon}{\sqrt{\Sigma}}\right) = \sqrt{\Sigma}\phi\left(\varepsilon, \Sigma\right) \cdot \left(-\frac{1}{2}\right)\frac{\varepsilon}{\Sigma\sqrt{\Sigma}} = -\frac{\varepsilon}{2\Sigma}\phi\left(\varepsilon, \Sigma\right).$$

For a 2-dimensional distribution,

$$\frac{\partial \Phi\left(\boldsymbol{\varepsilon}, \mathbf{\Sigma}\right)}{\partial \varepsilon_1} = \frac{\partial}{\partial \varepsilon_1} \int_{-\infty}^{\varepsilon_1} \int_{-\infty}^{\varepsilon_2} \phi\left(\mathbf{x}, \mathbf{\Sigma}\right) dx_2 dx_1 = \frac{\partial}{\partial \varepsilon_1} \int_{-\infty}^{\varepsilon_1} \phi\left(x_1, \Sigma_{11}\right) \int_{-\infty}^{\varepsilon_2} \phi\left(x_{2|1}, \Sigma_{2|1}\right) dx_2 dx_1$$

$$= \frac{\partial}{\partial \varepsilon_1} \int_{-\infty}^{\varepsilon_1} \phi\left(x_1, \Sigma_{11}\right) \Phi\left(\varepsilon_{2|1}, \Sigma_{2|1}\right) dx_1 = \phi\left(\varepsilon_1, \Sigma_{11}\right) \Phi\left(\varepsilon_{2|1}, \Sigma_{2|1}\right), \tag{30}$$

and likewise, symmetrically, for ε_2. (Here, the definition of $\varepsilon_{2|1}$ slips from being relative to x_1 to being relative to ε_1.)

As for the derivatives of a 2-dimensional c.d.f with respect to elements of $\mathbf{\Sigma}$, we start with Σ_{12}. Using

the rules just devised for the 1-dimensional case, the derivative is

$$\frac{\partial \Phi (\boldsymbol{\varepsilon}, \Sigma)}{\partial \Sigma_{12}} = \frac{\partial}{\partial \Sigma_{12}} \int_{-\infty}^{\varepsilon_1} \phi (x_1, \Sigma_{11}) \, \Phi (\varepsilon_{2|1}, \Sigma_{2|1}) \, dx_1 = \int_{-\infty}^{\varepsilon_1} \phi (x_1, \Sigma_{11}) \frac{\partial \Phi (\varepsilon_{2|1}, \Sigma_{2|1})}{\partial \Sigma_{12}} dx_1$$

$$= \int_{-\infty}^{\varepsilon_1} \phi (x_1, \Sigma_{11}) \left[\frac{\partial \Phi (\varepsilon_{2|1}, \Sigma_{2|1})}{\partial \varepsilon_{2|1}} \frac{\partial \varepsilon_{2|1}}{\partial \Sigma_{12}} + \frac{\partial \Phi (\varepsilon_{2|1}, \Sigma_{2|1})}{\partial \Sigma_{2|1}} \frac{\partial \Sigma_{2|1}}{\partial \Sigma_{11}} \right] dx_1$$

$$= \int_{-\infty}^{\varepsilon_1} \phi (x_1, \Sigma_{11}) \left[\phi (\varepsilon_{2|1}, \Sigma_{2|1}) \frac{\partial \varepsilon_{2|1}}{\partial \Sigma_{12}} - \phi (\varepsilon_{2|1}, \Sigma_{2|1}) \frac{\varepsilon_{2|1}}{2\Sigma_{2|1}} \frac{\partial \Sigma_{2|1}}{\partial \Sigma_{11}} \right] dx_1$$

$$= \int_{-\infty}^{\varepsilon_1} \phi \left((x_1, \varepsilon_2)', \Sigma \right) \left[\frac{\partial \varepsilon_{2|1}}{\partial \Sigma_{12}} - \frac{\varepsilon_{2|1}}{2\Sigma_{2|1}} \frac{\partial \Sigma_{2|1}}{\partial \Sigma_{11}} \right] dx_1. \tag{31}$$

Here,

$$\frac{\partial \varepsilon_{2|1}}{\partial \Sigma_{12}} = \frac{\partial}{\partial \Sigma_{12}} \left(\varepsilon_2 - \frac{\Sigma_{12}}{\Sigma_{11}} x_1 \right) = -\frac{x_1}{\Sigma_{11}}$$

$$\frac{\partial \Sigma_{2|1}}{\partial \Sigma_{12}} = \frac{\partial}{\partial \Sigma_{12}} \left(\Sigma_{22} - \frac{\boldsymbol{\Sigma}_{21} \Sigma_{12}}{\Sigma_{11}} \right) = -\frac{2\Sigma_{12}}{\Sigma_{11}}$$

(keeping in mind that $\Sigma_{12} = \Sigma_{12}$), so the bracketed expression in (31) works out to

$$\frac{\partial \varepsilon_{2|1}}{\partial \Sigma_{12}} - \frac{\varepsilon_{2|1}}{2\Sigma_{2|1}} \frac{\partial \Sigma_{2|1}}{\partial \Sigma_{11}} = -\frac{x_1}{\Sigma_{11}} - \frac{1}{2} \frac{\varepsilon_2 - \frac{\Sigma_{12}}{\Sigma_{11}} x_1}{\Sigma_{22} - \frac{\boldsymbol{\Sigma}_{21} \Sigma_{12}}{\Sigma_{11}}} \left(-\frac{2\Sigma_{12}}{\Sigma_{11}} \right) = -\frac{x_1 - \frac{\Sigma_{12}}{\Sigma_{22}} \varepsilon_2}{\Sigma_{11} - \frac{\boldsymbol{\Sigma}_{21} \Sigma_{12}}{\Sigma_{22}}} = -\frac{x_{1|2}}{\boldsymbol{\Sigma}_{1|2}}.$$

Substituting into (31),

$$\frac{\partial \Phi (\boldsymbol{\varepsilon}, \boldsymbol{\Sigma})}{\partial \Sigma_{12}} = \int_{-\infty}^{\varepsilon_1} \phi \left((x_1, \varepsilon_2)', \boldsymbol{\Sigma} \right) \left(-\frac{x_{1|2}}{\Sigma_{1|2}} \right) dx_1 = \phi (\varepsilon_2, \Sigma_{22}) \int_{-\infty}^{\varepsilon_1} \phi \left(x_{1|2}, \boldsymbol{\Sigma}_{1|2} \right) \left(-\frac{x_{1|2}}{\boldsymbol{\Sigma}_{1|2}} \right) dx_1$$

$$= \phi (\varepsilon_2, \Sigma_{22}) \cdot \phi \left. \left(x_{1|2}, \Sigma_{1|2} \right) \right]_{-\infty}^{\varepsilon_1} = \phi (\varepsilon_2, \Sigma_{22}) \, \phi (\varepsilon_{1|2}, \Sigma_{1|2}) = \phi (\boldsymbol{\varepsilon}, \boldsymbol{\Sigma}) \tag{32}$$

(rather remarkably).

To compute the derivative of the 2-dimensional normal c.d.f with respect to Σ_{11}, let $\boldsymbol{\nu} = \left(\frac{\varepsilon_1}{\sqrt{\Sigma_{11}}}, \varepsilon_2 \right)'$
and

$$\boldsymbol{\Omega} = \mathrm{Var} \left[\boldsymbol{\nu} \right] = \begin{bmatrix} 1 & \frac{\Sigma_{12}}{\sqrt{\Sigma_{11}}} \\ \frac{\Sigma_{21}}{\sqrt{\Sigma_{11}}} & \Sigma_{22} \end{bmatrix}$$

49

so that $\Phi\left(\boldsymbol{\varepsilon}, \boldsymbol{\Sigma}\right) = \Phi\left(\boldsymbol{\nu}, \boldsymbol{\Omega}\right)$. Then

$$\frac{\partial \Phi\left(\boldsymbol{\varepsilon}, \boldsymbol{\Sigma}\right)}{\partial \Sigma_{11}} = \frac{\partial \Phi\left(\boldsymbol{\nu}, \boldsymbol{\Omega}\right)}{\partial \Sigma_{11}} = \frac{\partial \Phi\left(\boldsymbol{\nu}, \boldsymbol{\Omega}\right)}{\partial \nu_1} \frac{\partial \nu_1}{\partial \Sigma_{11}} + \frac{\partial \Phi\left(\boldsymbol{\nu}, \boldsymbol{\Omega}\right)}{\partial \Omega_{12}} \frac{\partial \Omega_{12}}{\partial \Sigma_{11}}$$

$$= \phi\left(\nu_1, \Omega_{11}\right) \Phi\left(\nu_{2|1}, \Omega_{2|1}\right) \left(-\frac{\varepsilon_1}{2\Sigma_{11}\sqrt{\Sigma_{11}}}\right) + \phi\left(\boldsymbol{\nu}, \boldsymbol{\Omega}\right) \left(-\frac{\Sigma_{12}}{2\Sigma_{11}\sqrt{\Sigma_{11}}}\right),$$

the last step expanding with (30) and (32). Since $\phi\left(\nu_1, \Omega_{11}\right) = \sqrt{\Sigma_{11}}\phi\left(\varepsilon_1, \Sigma_{11}\right)$ and $\phi\left(\boldsymbol{\nu}, \boldsymbol{\Omega}\right) = \sqrt{\Sigma_{11}}\phi\left(\boldsymbol{\varepsilon}, \boldsymbol{\Sigma}\right)$,

$$\frac{\partial \Phi\left(\boldsymbol{\varepsilon}, \boldsymbol{\Sigma}\right)}{\partial \Sigma_{11}} = \sqrt{\Sigma_{11}}\phi\left(\varepsilon_1, \Sigma_{11}\right) \Phi\left(\varepsilon_{2|1}, \Sigma_{2|1}\right) \left(-\frac{\varepsilon_1}{2\Sigma_{11}\sqrt{\Sigma_{11}}}\right) + \sqrt{\Sigma_{11}}\phi\left(\boldsymbol{\varepsilon}, \boldsymbol{\Sigma}\right) \left(-\frac{\Sigma_{12}}{2\Sigma_{11}\sqrt{\Sigma_{11}}}\right)$$

$$= -\frac{1}{2\Sigma_{11}} \left(\varepsilon_1\phi\left(\varepsilon_1, \Sigma_{11}\right) \Phi\left(\varepsilon_{2|1}, \Sigma_{2|1}\right) + \Sigma_{12}\phi\left(\boldsymbol{\varepsilon}, \boldsymbol{\Sigma}\right)\right)$$

$$= -\frac{\phi\left(\varepsilon_1, \Sigma_{11}\right)}{2\Sigma_{11}} \left(\varepsilon_1\Phi\left(\varepsilon_{2|1}, \Sigma_{2|1}\right) + \Sigma_{12}\phi\left(\varepsilon_{2|1}, \Sigma_{2|1}\right)\right).$$

The formula for $\dfrac{\partial \Phi\left(\boldsymbol{\varepsilon}, \boldsymbol{\Sigma}\right)}{\partial \Sigma_{22}}$ is of course analogous.

Since cumulative normal distributions above dimension 2 are simulated with the GHK algorithm, the derivatives of that algorithm need to be computed according to the exact formulas for the simulation, rather than with ones like those above. See Appendix B.

A.2.3 Derivatives of $\varepsilon_{2|1}$ and $\Sigma_{2|1}$

To state the derivatives of $\boldsymbol{\varepsilon}_{2|1} = \boldsymbol{\varepsilon}_2 - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\varepsilon_1$ and $\boldsymbol{\Sigma}_{2|1} = \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}$, let $\boldsymbol{\beta} = \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}$ and $\mathbf{P} = \left[-\boldsymbol{\beta}\,|\,\mathbf{I}\right]$. So $\boldsymbol{\varepsilon}_{2|1} = \mathbf{P}\boldsymbol{\varepsilon}$ and $\boldsymbol{\Sigma}_{2|1} = \text{Var}\left[\boldsymbol{\varepsilon}_{2|1}\right] = \mathbf{P}\boldsymbol{\Sigma}\mathbf{P}'$. Then

$$\frac{\partial \boldsymbol{\varepsilon}_{2|1}}{\partial \boldsymbol{\varepsilon}} = \mathbf{P}$$

and

$$\frac{\partial \boldsymbol{\Sigma}_{2|1,ij}}{\partial \boldsymbol{\Sigma}_{kl}} = \frac{\partial \left(\mathbf{P}\boldsymbol{\Sigma}\mathbf{P}'\right)_{ij}}{\partial \boldsymbol{\Sigma}_{kl}} = \frac{\partial \left(\mathbf{P}_i\boldsymbol{\Sigma}\mathbf{P}'_j\right)}{\partial \boldsymbol{\Sigma}_{kl}} = \mathbf{P}_i\frac{\partial \boldsymbol{\Sigma}_j}{\partial \boldsymbol{\Sigma}_{kl}}\mathbf{P}' = \mathbf{P}_i\mathbf{S}_{kl}\mathbf{P}'_j = \mathbf{P}_{ik}\mathbf{P}_{jl}$$

$$\Rightarrow \frac{\partial \text{vec}\left(\boldsymbol{\Sigma}_{2|1}\right)}{\partial \text{vec}\left(\boldsymbol{\Sigma}\right)} = \mathbf{P} \otimes \mathbf{P}.$$

B Formulas for the GHK estimate and scores thereof when integration regions are bounded below and above

This appendix exhibits the formulas for the GHK estimator when lower as well upper bounds of integration are provided. It also shows how scores are computed. It borrows the notation of Gates (2006) and provides almost no motivation.

Our task is to integrate the $\mathbf{0}$-centered d-dimensional normal distribution with covariance $\mathbf{\Sigma}$ over the Cartesian region defined by lower and upper bounds $\underline{\mathbf{x}} = (\underline{x}_1, \ldots, \underline{x}_d)'$, $\bar{\mathbf{x}} = (\bar{x}_1, \ldots, \bar{x}_d)'$, which can have infinite entries. Let $\mathbf{T} = [t_{ij}]_{i,j=1\ldots d}$ be the Cholesky factorization of $\mathbf{\Sigma}$: $\mathbf{\Sigma} = \mathbf{T}\mathbf{T}'$. Let $u_1, u_2, \ldots$ be a sequence of draws distributed across the unit interval $[0, 1)$. Let $\Phi(\cdot)$ be the standard cumulative normal distribution function. Then the simulated probability, p, is estimated by the algorithm:

$\underline{b}_1 := \underline{x}_1 / t_{11}, \ \bar{b}_1 := \bar{x}_1 / t_{11}$

$a_1 := \Phi(\bar{b}_1) - \Phi(\underline{b}_1)$

For $i := 2, \ldots, d$,

$$z_{i-1} := \Phi^{-1}(a_{i-1})$$

$$\underline{b}_i := \left(\underline{x}_i - \sum_{j=1}^{i-1} t_{ij} z_j \right) \bigg/ t_{ii}$$

$$\bar{b}_i := \left(\bar{x}_i - \sum_{j=1}^{i-1} t_{ij} z_j \right) \bigg/ t_{ii}$$

$$a_{i-1} := (1 - u_i) \Phi(\underline{b}_i) + u_i \Phi(\bar{b}_i).$$

$$p := \prod_{i=1}^{d} \left(\Phi(\bar{b}_i) - \Phi(\underline{b}_i) \right)$$

where $:=$ indicates assignment.

To discuss the derivatives of the probability with respect to the parameters $\underline{\mathbf{x}}$, $\bar{\mathbf{x}}$, and vech $(\mathbf{T})$, concatenate them into a single parameter vector δ; and interpret subscripting with "(i)" as taking the first i entries of a vector. Following Bolduc (1999), we state the derivatives in a recursive manner, through repeated use of the chain rule. Let δ_m be a parameter. In general,

$$\frac{\partial p}{\partial \delta_m} = p \frac{\partial \ln p}{\partial \delta_m} = p \sum_{j=1}^{d} \frac{\partial \ln \left(\Phi(\bar{b}_j) - \Phi(\underline{b}_j) \right)}{\partial \delta_m} = p \sum_{j=1}^{d} \frac{\phi(\bar{b}_j) \dfrac{\partial \bar{b}_j}{\partial \delta_m} - \phi(\underline{b}_j) \dfrac{\partial \underline{b}_j}{\partial \delta_m}}{\Phi(\bar{b}_j) - \Phi(\underline{b}_j)}$$

where $\phi(\cdot)$ is the standard normal distribution. So our task is to compute $\dfrac{\partial \underline{b}_j}{\partial \delta_m}$ and $\dfrac{\partial \bar{b}_j}{\partial \delta_m}$ in the above, which we do by differentiating the formulas in the algorithm:

Case δ_m is $\underline{x}_i$ for some i:

$$\frac{\partial \underline{b}_j}{\partial \underline{x}_i} = \begin{cases} \dfrac{1}{t_{jj}} & \text{if } i = j \\[2ex] -\dfrac{1}{t_{jj}} \displaystyle\sum_{k=1}^{j-1} t_{jk} \dfrac{\partial z_k}{\partial \underline{x}_i} & \text{if } i < j \\[2ex] 0 & \text{otherwise} \end{cases} \quad \text{and} \quad \frac{\partial \bar{b}_j}{\partial \underline{x}_i} = \begin{cases} 0 & \text{if } i = j \\[2ex] -\dfrac{1}{t_{jj}} \displaystyle\sum_{k=1}^{j-1} t_{jk} \dfrac{\partial z_k}{\partial \underline{x}_i} & \text{if } i < j \\[2ex] 0 & \text{otherwise} \end{cases}$$

(These two are identical except when $i = j$.)

Case δ_m is $\bar{x}_i$ for some i:

$$\frac{\partial \underline{b}_j}{\partial \bar{x}_i} = \begin{cases} 0 & \text{if } i = j \\[2ex] -\dfrac{1}{t_{jj}} \displaystyle\sum_{k=1}^{j-1} t_{jk} \dfrac{\partial z_k}{\partial \bar{x}_i} & \text{if } i < j \\[2ex] 0 & \text{otherwise} \end{cases} \quad \text{and} \quad \frac{\partial \bar{b}_j}{\partial \bar{x}_i} = \begin{cases} \dfrac{1}{t_{jj}} & \text{if } i = j \\[2ex] -\dfrac{1}{t_{jj}} \displaystyle\sum_{k=1}^{j-1} t_{jk} \dfrac{\partial z_k}{\partial \bar{x}_i} & \text{if } i < j \\[2ex] 0 & \text{otherwise} \end{cases}$$

(Again, these two are identical unless $i = j$.)

Case δ_m is t_{ik} for some $i, k, i \geq k$:

$$\frac{\partial \underline{b}_j}{\partial t_{ik}} = \begin{cases} -\dfrac{\bar{b}_j}{t_{jj}} & \text{if } j = i = k \\[2ex] -\dfrac{z_k}{t_{jj}} & \text{if } j = i > k \\[2ex] -\dfrac{1}{t_{jj}} \displaystyle\sum_{h=1}^{j-1} t_{jh} \dfrac{\partial z_h}{\partial t_{ik}} & \text{if } j > i \geq k \end{cases} \quad \text{and} \quad \frac{\partial \bar{b}_j}{\partial t_{ik}} = \begin{cases} -\dfrac{\bar{b}_j}{t_{jj}} & \text{if } j = i = k \\[2ex] -\dfrac{z_k}{t_{jj}} & \text{if } j = i > k \\[2ex] -\dfrac{1}{t_{jj}} \displaystyle\sum_{h=1}^{j-1} t_{jh} \dfrac{\partial z_h}{\partial t_{ik}} & \text{if } j > i \geq k \end{cases}$$

(These two are the same except when $i = j = k$.)

All three cases contain derivatives of z_k, which we expand recursively with

$$\frac{\partial z_k}{\partial \delta_m} = \frac{\partial \Phi^{-1}(a_k)}{\partial \delta_m} = \frac{1}{\phi(\Phi^{-1}(a_k))} \frac{\partial a_k}{\partial \delta_m} = \frac{1}{\phi(z_k)} \left[u_k \cdot \phi(\bar{b}_k) \frac{\partial \bar{b}_k}{\partial \delta_m} + (1 - u_k) \cdot \phi(\underline{b}_k) \frac{\partial \underline{b}_k}{\partial \delta_m} \right].$$

We can also write this as $\dfrac{\partial z_k}{\partial \delta_m} = \dfrac{\partial z_k}{\partial \bar{b}_k} \dfrac{\partial \bar{b}_k}{\partial \delta_m} + \dfrac{\partial z_k}{\partial \underline{b}_k} \dfrac{\partial \underline{b}_k}{\partial \delta_m}$ where $\dfrac{\partial z_k}{\partial \bar{b}_k} = u_k \dfrac{\phi(\bar{b}_k)}{\phi(z_k)}$ and $\dfrac{\partial z_k}{\partial \underline{b}_k} = (1 - u_k) \dfrac{\phi(\underline{b}_k)}{\phi(z_k)}$. In most cases, as noted parenthetically, $\dfrac{\partial \bar{b}_k}{\partial \delta_m} = \dfrac{\partial \underline{b}_k}{\partial \delta_m}$, so then we can write $\dfrac{\partial z_k}{\partial \delta_m} = \left(\dfrac{\partial z_k}{\partial \bar{b}_k} + \dfrac{\partial z_k}{\partial \underline{b}_k} \right) \dfrac{\partial \bar{b}_k}{\partial \delta_m}$.

After completing the recursion, derivatives with respect to $\mathbf{T}$ are transformed into ones with respect to $\boldsymbol{\Sigma}$ as described in Gates (2006, p. 198).