The Stata Journal (2013)
13, Number 2, pp. 382–397

Fitting the generalized multinomial logit model in Stata

Yuanyuan Gu
Centre for Health Economics Research and Evaluation
University of Technology, Sydney
Sydney, Australia
yuanyuan.gu@chere.uts.edu.au

Arne Risa Hole
Department of Economics
University of Sheffield
Sheffield, UK
a.r.hole@sheffield.ac.uk

Stephanie Knox
Centre for Health Economics Research and Evaluation
University of Technology, Sydney
Sydney, Australia
stephanie.knox@chere.uts.edu.au

Abstract. In this article, we describe the `gmnl` Stata command, which can be used to fit the generalized multinomial logit model and its special cases.

Keywords: st0301, gmnl, gmnlpred, gmnlcov, generalized multinomial logit, scale heterogeneity multinomial logit, maximum simulated likelihood

1 Introduction

Explaining variations in the behaviors of individuals is of central importance in choice analysis. For the last decade, the most popular explanation has been preference or taste heterogeneity; that is, some individuals care more about particular product attributes than do others. This assumption is most naturally represented via random parameter models, among which the mixed logit (MIXL) model has become the standard to use (McFadden and Train 2000).

Recently, however, a group of researchers (for example, Louviere et al. [1999], Louviere et al. [2002], Louviere and Eagle [2006], and Louviere et al. [2007]) has argued that in most choice contexts, much of the preference heterogeneity may be better described as "scale" heterogeneity; that is, with attribute coefficients fixed, the scale of the idiosyncratic error term is greater for some consumers than it is for others. Because the scale of the error term is inversely related to the error variance, this argument implies that choice behavior is more random for some consumers than it is for others. Although the scale of the error term in discrete choice models cannot be separately identified from the attribute coefficients, it is possible to identify relative scale terms across consumers. Thus the statement that all heterogeneity is in the scale of the error term "is observationally equivalent to the statement that heterogeneity takes the form of the vector of utility weights being scaled up or down proportionately as one 'looks' across consumers" (Fiebig et al. 2010). These arguments have led to the scale heterogeneity multinomial logit (S-MNL) model, a much more parsimonious model specification than MIXL.

To accommodate both preference and scale heterogeneity, Fiebig et al. (2010) developed a generalized multinomial logit (G-MNL) model that nests MIXL and S-MNL. Their research also shows that the two sources of heterogeneity often coexist but that their importance varies in different choice contexts.

In this article, we will describe the `gmnl` Stata command, which can be used to fit the G-MNL model and its special cases. The command is a generalization of the `mixlogit` command developed by Hole (2007). We will also present an empirical example that demonstrates how to use `gmnl`, and we will discuss related computational issues.

2 The G-MNL model and its special cases

We assume a sample of N respondents with the choice of J alternatives in T choice situations.[1] Following Fiebig et al. (2010), the G-MNL model gives the probability of respondent i choosing alternative j in choice situation t as

$$\Pr(\text{choice}_{it} = j|\beta_i) = \frac{\exp(\beta_i' x_{itj})}{\sum_{k=1}^{J} \exp(\beta_i' x_{itk})} \qquad (1)$$
$$i = 1, \ldots, N; \quad t = 1, \ldots, T; \quad j = 1, \ldots, J$$

where x_{itj} is a vector of observed attributes of alternative j and β_i is a vector of individual-specific parameters defined as

$$\beta_i = \sigma_i \beta + \{\gamma + \sigma_i(1-\gamma)\}\eta_i \qquad (2)$$

The specification of β_i in (2) is central to G-MNL and differentiates it from previous heterogeneity models. It depends on a constant vector β, a scalar parameter γ, a random vector η_i distributed MVN$(0, \Sigma)$, and σ_i, the individual-specific scale of the idiosyncratic error.

In Fiebig et al. (2010), γ is constrained to be between 0 and 1. In extreme cases, $\gamma = 1$ leads to G-MNL-I: $\beta_i = \sigma_i\beta + \eta_i$ and $\gamma = 0$ leads to G-MNL-II:[2] $\beta_i = \sigma_i(\beta + \eta_i)$. To understand the difference between these two models, Fiebig et al. (2010) describe them with a single equation: $\beta_i = \sigma_i\beta + \eta_i^*$, where σ_i captures scale heterogeneity and η_i^* captures residual preference heterogeneity. Through this, we can see that in G-MNL-I, the standard deviation of η_i^* is independent of the scaling of β, whereas in G-MNL-II, it is proportional to σ_i.

However, an article by Keane and Wasi (forthcoming) points out that $\gamma < 0$ or $\gamma > 1$ still permits sensible behavioral interpretations, and thus there is no reason to impose the constraint. We follow their advice and allow γ to take any value.

1. We could also consider a different number of alternatives and choice situations for each respondent; for example, see Greene and Hensher (2010). The `gmnl` command can handle both of these cases.
2. Greene and Hensher (2010) call this the "scaled mixed logit model".

Three useful special cases of G-MNL are the following:

- MIXL: $\beta_i = \beta + \eta_i$ (when $\sigma_i = 1$)
- S-MNL: $\beta_i = \sigma_i \beta$ (when $\text{var}(\eta_i) = 0$)
- Standard multinomial logit: $\beta_i = \beta$ (when $\sigma_i = 1$ and $\text{var}(\eta_i) = 0$)

The `gmnl` command includes an option for fitting MIXL models, but we recommend that `mixlogit` be used for this purpose because it is usually faster.

To complete the model specification, we need to choose a distribution for σ_i. Although any distribution defined on the positive real line is a theoretical possibility, Fiebig et al. (2010) assume that σ_i is distributed lognormal with standard deviation τ and mean $\overline{\sigma} + \theta z_i$, where $\overline{\sigma}$ is a normalizing constant and z_i is a vector of characteristics of individual i that can be used to explain why σ_i differs across people.

3 Maximum simulated likelihood

The log likelihood for G-MNL is given by

$$\text{LL}(\beta, \gamma, \tau, \theta, \Sigma) = \sum_{i=1}^{N} \ln \left\{ \int \prod_{t=1}^{T} \prod_{j=1}^{J} \Pr(\text{choice}_{it} = j|\beta_i)^{y_{itj}} p(\beta_i|\beta, \gamma, \tau, \theta, \Sigma) d\beta_i \right\} \quad (3)$$

where y_{itj} is the observed choice variable, $\Pr(\text{choice}_{it} = j|\beta_i)$ is given by (1), and $p(\beta_i|\beta, \gamma, \tau, \theta, \Sigma)$ is implied by (2).

Maximizing the log likelihood in (3) directly is rather difficult because the integral does not have a closed-form representation and so must be evaluated numerically. We choose to approximate it with simulation (see Train [2009], for example). The simulated likelihood is

$$\begin{aligned} \text{SLL}(\beta, \gamma, \tau, \theta, \Sigma) &= \sum_{i=1}^{N} \ln \left\{ \frac{1}{R} \sum_{r=1}^{R} \prod_{t=1}^{T} \prod_{j=1}^{J} \Pr(\text{choice}_{it} = j|\beta_i^{[r]})^{y_{itj}} \right\} \\ \beta_i^{[r]} &= \sigma_i^{[r]} \beta + \left\{ \gamma + \sigma_i^{[r]} (1 - \gamma) \right\} \eta_i^{[r]} \\ \sigma_i^{[r]} &= \exp(\overline{\sigma} + \theta z_i + \tau \nu^{[r]}) \end{aligned}$$

where $\eta_i^{[r]}$ is a vector generated from $\text{MVN}(0, \Sigma)$ and $\nu^{[r]}$ is a $N(0, 1)$ scalar. $\eta_i^{[r]}$ and $\nu^{[r]}$ are generated using Halton draws (Halton 1964) and pseudorandom draws, respectively. When testing the code, we found that this combination works better than using Halton draws to generate all the random terms.

Following Fiebig et al. (2010), we set the normalizing constant $\overline{\sigma}$ as $-\ln\{\frac{1}{N} \sum_{i=1}^{N} \exp(\tau \nu_i^{[r]})\}$, where $\nu_i^{[r]}$ is the rth draw for the ith person. We also draw ν from a truncated normal with truncation at ± 2.

4 The gmnl command

4.1 Syntax

`gmnl` is implemented as a `gf0 ml` evaluator. The Halton draws used in the estimation process are generated using the Mata function `halton()` (Drukker and Gates 2006). The generic syntax for the command is as follows:

`gmnl` *depvar* [*varlist*] [*if*] [*in*], `group(`*varname*`)` [`rand(`*varlist*`)` `id(`*varname*`)` `corr` `nrep(`#`)` `burn(`#`)` `gamma(`#`)` `scale(`*matrix*`)` `het(`*varlist*`)` `mixl` `seed(`#`)` `level(`#`)` `constraints(`*numlist*`)` `vce(`*vcetype*`)` *maximize_options*]

The command `gmnlpred` can be used following `gmnl` to obtain predicted probabilities. The predictions are available both in and out of sample; type `gmnlpred ... if e(sample) ...` if predictions are wanted for the estimation sample only.

`gmnlpred` *newvar* [*if*] [*in*] [, `nrep(`#`)` `burn(`#`)` `ll`]

The command `gmnlcov` can be used following `gmnl` to obtain the elements in the coefficient covariance matrix along with their standard errors. This command is only relevant when the coefficients are specified to be correlated; see the `corr` option below. `gmnlcov` is a wrapper for `nlcom` (see [R] **nlcom**).

`gmnlcov` [, `sd`]

The command `gmnlbeta` can be used following `gmnl` to obtain the individual-level parameters corresponding to the variables in the specified *varlist* by using the method proposed by Revelt and Train (2000) (see also Train [2009, chap. 11]). The individual-level parameters are stored in a data file specified by the user. As with `gmnlpred`, the predictions are available both in and out of sample; type `gmnlbeta ... if e(sample) ...` if predictions are wanted for the estimation sample only.

`gmnlbeta` *varlist* [*if*] [*in*], `saving(`*filename*`)` [`replace` `nrep(`#`)` `burn(`#`)`]

4.2 gmnl options

`group(`*varname*`)` specifies a numeric identifier variable for the choice occasions. `group()` is required.

`rand(`*varlist*`)` specifies the independent variables whose coefficients are random (normally distributed). The variables immediately following the dependent variable in the syntax are specified to have fixed coefficients.

`id(`*varname*`)` specifies a numeric identifier variable for the decision makers. This option should be specified only when each individual performs several choices, that is, when the dataset is a panel.

`corr` specifies that the random coefficients be correlated. The default is that they are independent. When the `corr` option is specified, the estimated parameters are the means of the (fixed and random) coefficients plus the elements of the lower-triangular matrix $\mathbf{L}$, where the covariance matrix for the random coefficients is given by $\Sigma = LL'$. The estimated parameters are reported in the following order: the means of the fixed coefficients, the means of the random coefficients, and the elements of the $\mathbf{L}$ matrix. The `gmnlcov` command can be used postestimation to obtain the elements in the Σ matrix along with their standard errors.

If the `corr` option is not specified, the estimated parameters are the means of the fixed coefficients and the means and standard deviations of the random coefficients, reported in that order. The sign of the estimated standard deviations is irrelevant. Although in practice the estimates may be negative, interpret them as being positive.

The sequence of the parameters is important to bear in mind when specifying starting values.

`nrep(`#`)` specifies the number of draws used for the simulation. The default is `nrep(50)`.

`burn(`#`)` specifies the number of initial sequence elements to drop when creating the Halton sequences. The default is `burn(15)`. Specifying this option helps reduce the correlation between the sequences in each dimension. Train (2009, 227) recommends that # should be at least as large as the largest prime number used to generate the sequences. If there are K random coefficients, `gmnl` uses the first K primes to generate the Halton draws.

`gamma(`#`)` constrains the gamma parameter to the specified value in the estimations.

`scale(`*matrix*`)` specifies a matrix whose elements indicate whether their corresponding variable will be scaled (1 = scaled and 0 = not scaled). The matrix should have one row, and the number of columns should be equal to the number of explanatory variables in the model.

`het(`*varlist*`)` specifies the variables in the z_i vector (if any).

`mixl` specifies that a mixed logit model should be estimated instead of a G-MNL model.

`seed(`#`)` specifies the seed. The default is `seed(12345)`.

`level(`#`)`; see [R] **estimation options**.

`constraints(`*numlist*`)`; see [R] **estimation options**.

`vce(`*vcetype*`)`; *vcetype* may be `oim`, `robust`, `cluster` *clustvar*, or `opg`; see [R] ***vce_option***.

maximize_options: `difficult`, `technique(`*algorithm_spec*`)`, `iterate(`#`)`, `trace`, `gradient`, `showstep`, `hessian`, `tolerance(`#`)`, `ltolerance(`#`)`, `gtolerance(`#`)`, `nrtolerance(`#`)`, and `from(`*init_specs*`)`; see [R] **maximize**.

4.3 gmnlpred options

`nrep(#)` specifies the number of draws used for the simulation. The default is `nrep(50)`.

`burn(#)` specifies the number of initial sequence elements to drop when creating the Halton sequences. The default is `burn(15)`. Specifying this option helps reduce the correlation between the sequences in each dimension. Train (2009, 227) recommends that # should be at least as large as the largest prime number used to generate the sequences. If there are K random coefficients, `gmnl` uses the first K primes to generate the Halton draws.

`ll` estimates individual log likelihoods.

4.4 gmnlcov option

`sd` reports the standard deviations of the correlated coefficients instead of the covariance matrix.

4.5 gmnlbeta options

`saving(`*filename*`)` saves individual-level parameters to *filename*. `saving()` is required.

`replace` overwrites *filename*.

`nrep(#)` specifies the number of draws used for the simulation. The default is `nrep(50)`.

`burn(#)` specifies the number of initial sequence elements to drop when creating the Halton sequences. The default is `burn(15)`. Specifying this option helps reduce the correlation between the sequences in each dimension. Train (2009, 227) recommends that # should be at least as large as the largest prime number used to generate the sequences. If there are K random coefficients, `gmnl` uses the first K primes to generate the Halton draws.

5 Computational issues

As in any model estimated using maximum simulated likelihood, parameter estimates of G-MNL would depend on four factors: the random-number seed, number of draws, starting values, and optimization method. If the four factors are fixed, the same maximum likelihood estimates would be obtained at each simulation.

To have a good approximation of the likelihood, we must use a reasonable number of draws: the more draws used, the better the accuracy. However, a larger number of draws almost surely leads to a longer computation time. To determine the minimum number of draws for a desirable level of accuracy remains a theoretical challenge, but empirically, we may run the `gmnl` command several times with an increasing number of draws in each run (with the other three factors fixed) until the estimates stabilize. We should mention that too few draws may lead to serious convergence problems: a good starting point is 500 draws.

Starting values are crucial to achieve convergence, especially for the full model, that is, G-MNL with correlated random parameters (G-MNL correlated). If we see optimization as climbing a hill, then where we start climbing is one of the major factors that decide how long it will take to reach the top or if we can ever get there in the end. If we start from the bottom where the territory is often flat, the direction guidance (that is, the first-order derivatives of the likelihood) may not function well and lead us farther away from the top. For this reason, the default starting values based on the multinomial logit estimates may not be the best, and we often need to choose our own set of starting values.

To estimate "G-MNL correlated", we have tried four different sets of starting values: G-MNL uncorrelated, MIXL uncorrelated, MIXL correlated, and G-MNL correlated with γ fixed as 0. In most cases, these four sets of starting values would all lead to convergence, but the speed might be very different. In any case, we should not be content with only one set of starting values because even if the model converges, it is not guaranteed that we have reached the global maximum. We suggest running the routine multiple times, each with different starting values, and reporting the estimates from the run that obtains the largest likelihood.

The choice of optimization method is another important factor that affects model convergence. Stata allows four options: Newton–Raphson (default), Berndt–Hall–Hall–Hausman, Davidon–Fletcher–Powell, and Broyden–Fletcher–Goldfarb–Shanno. Chapter 8 in Train (2009) describes these methods in detail and concludes that Broyden–Fletcher–Goldfarb–Shanno usually performs better than the others. With `mixlogit` and `gmnl`, however, we have found that Newton–Raphson often works best in the sense that it is more likely to converge than the alternative algorithms. The only problem with Newton–Raphson is that it can be very slow when there are a lot of parameters to estimate.

Finally, we have found that in some cases, different computers can give different results if there are several random parameters in the model and γ is freely estimated. This can happen when the model is numerically unstable and different numbers of processors are used during estimation.

6 Empirical example

We will now present some examples that demonstrate how the `gmnl` command can be used to fit the different models described in section 2. We will start by fitting some relatively simple models, and then we will build up the complexity gradually. The data used in the examples come from a stated preference study on Australian women who were asked to choose whether to have a pap smear test; see Fiebig and Hall (2004). There were 79 women in the sample, and each respondent was presented with 32 scenarios. Thus in terms of the model structure described in section 2, $N = 79$, $T = 32$, and $J = 2$. The dataset also contains five attributes, which are described in table 1. Besides these five attributes, an alternative specific constant (ASC) will be used to measure intangible aspects of the pap smear test not captured by the design attributes (some women would choose or not choose the test just because of these intangible aspects no matter what attributes they are presented with).

Table 1. Pap smear test data. Definition of variables.

Variable	Definition
`knowgp`	1 if the general practitioner is known to the patient; 0 otherwise
`malegp`	1 if the general practitioner is male; 0 if the general practitioner is female
`testdue`	1 if the patient is due or overdue for a pap smear test; 0 otherwise
`drrec`	1 if the general practitioner recommends that the patient have a pap smear test; 0 otherwise
`cost`	cost of test (unit: 10 Australian dollar)

To give an impression of how the data are structured, we have listed the first six observations below. Each observation corresponds to an alternative, and the dependent variable `y` is 1 for the chosen alternative in each choice situation and 0 otherwise. `gid` identifies the alternatives in a choice situation; `rid` identifies the choice situations faced by a given individual; and the remaining variables are the alternative attributes described in table 1 and the ASC (`dummy_test`). In the listed data, the same individual faces three choice situations.

```
. use paptest.dta
. generate cost = papcost/10
. list y dummy_test knowgp malegp testdue drrec cost gid rid in 1/6, sep(2)
> abb(10)
```

	y	dummy_test	knowgp	malegp	testdue	drrec	cost	gid	rid
1.	0	1	1	0	0	0	2	1	1
2.	1	0	0	0	0	0	0	1	1
3.	0	1	1	0	0	1	2	2	1
4.	1	0	0	0	0	0	0	2	1
5.	0	1	0	1	0	1	2	3	1
6.	1	0	0	0	0	0	0	3	1

We start by fitting a relatively simple S-MNL model with a fixed (nonrandom) ASC. Fiebig et al. (2010) have pointed out that ASCs should not be scaled, because they are fundamentally different from observed attributes. We can fit the model with a fixed ASC by using the `scale()` option of `gmnl` as described below.

```
. /*S-MNL with fixed ASC*/
. matrix scale = 0,1,1,1,1,1
. gmnl y dummy_test knowgp malegp testdue drrec cost, group(gid) id(rid)
> nrep(500) scale(scale)
Iteration 0:   log likelihood =  -1452.649  (not concave)
  (output omitted)
Iteration 7:   log likelihood = -1123.7542
Generalized multinomial logit model              Number of obs   =       5056
                                                 Wald chi2(6)    =     238.93
Log likelihood = -1123.7542                      Prob > chi2     =     0.0000
                                  (Std. Err. adjusted for clustering on rid)
```

y	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
dummy_test	-1.938211	.13976	-13.87	0.000	-2.212136	-1.664286
knowgp	1.811842	.4322376	4.19	0.000	.9646719	2.659012
malegp	-.9527219	.319577	-2.98	0.003	-1.579081	-.3263625
testdue	5.305355	1.229367	4.32	0.000	2.895841	7.714869
drrec	2.656325	.6021513	4.41	0.000	1.47613	3.83652
cost	.0043925	.0932526	0.05	0.962	-.1783792	.1871643
/tau	1.458027	.1726576	8.44	0.000	1.119624	1.79643

```
The sign of the estimated standard deviations is irrelevant: interpret them as
being positive
```

To avoid scaling the ASC, we create a matrix whose elements indicate whether their corresponding variable will be scaled (1 = scaled and 0 = not scaled). Here the "scale" matrix defined as $(0, 1, 1, 1, 1, 1)$ corresponds to the variables in the order in which they are specified in the model (`dummy_test`, `knowgp`, etc.). Therefore, among these six variables, only `dummy_test` (that is, the ASC) is not scaled.

We should mention that the number of observations reported in the table, 5,056, is $N \times T \times J$, that is, the total number of choices times the number of alternatives. For most purposes, such as computing information criteria, it is more appropriate to use the total number of choices ($N \times T$); therefore, we do not recommend that you use the `estat ic` command after `gmnl`.

We then let `dummy_test` be random, which leads to our second model: S-MNL with random ASC.[3]

```
. /*S-MNL with random ASC*/
. matrix scale = 1,1,1,1,1,0
. gmnl y knowgp malegp testdue drrec cost, group(gid) id(rid) rand(dummy_test)
> nrep(500) scale(scale) gamma(0)
Iteration 0:   log likelihood = -1431.8448  (not concave)
  (output omitted)
Iteration 8:   log likelihood = -1061.7787
Generalized multinomial logit model              Number of obs   =       5056
                                                 Wald chi2(6)    =     111.46
Log likelihood = -1061.7787                      Prob > chi2     =     0.0000
                                  (Std. Err. adjusted for clustering on rid)
```

y	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
Mean						
knowgp	.6263819	.1431434	4.38	0.000	.3458261	.9069378
malegp	-1.350731	.2024933	-6.67	0.000	-1.74761	-.9538514
testdue	2.954924	.2950128	10.02	0.000	2.37671	3.533139
drrec	.7730114	.1608242	4.81	0.000	.4578018	1.088221
cost	-.1701498	.0585679	-2.91	0.004	-.2849408	-.0553588
dummy_test	-.7052151	.3578936	-1.97	0.049	-1.406674	-.0037565
SD						
dummy_test	2.660664	.2798579	9.51	0.000	2.112152	3.209175
/tau	.9255689	.1032721	8.96	0.000	.7231593	1.127978

```
The sign of the estimated standard deviations is irrelevant: interpret them as
being positive
```

3. Strictly speaking, this is not an S-MNL but a parsimonious form of G-MNL; that is, we model ASCs using preference heterogeneity but model other attributes using scale heterogeneity. This specification of G-MNL has been used by Fiebig et al. (2011) and Knox et al. (2013).

Comparing "S-MNL with fixed ASC" with "S-MNL with random ASC", we can see that the latter model improved the model fit by adding one more parameter, the standard deviation of `dummy_test`, which is statistically significant.[4] The improvement in fit is not surprising because the random ASC captures preference heterogeneity and allows for correlation across choice situations because of the panel nature of the data. The parameter estimates between the two models are somewhat different, but they cannot be compared directly because of differences in scale across models, as indicated by the estimate of τ. Instead, we should run the `gmnlpred` command to compare the predicted probabilities. We shall demonstrate how to do predictions after fitting the full G-MNL model.

The third example is a G-MNL model in which `dummy_test`, `testdue`, and `drrec` are given random coefficients. For the moment, the coefficients are specified to be uncorrelated; that is, the off-diagonal elements of Σ are all 0. To speed up the estimation, we constrain γ to 0 by using the `gamma(0)` option, which implies that the fitted model is a G-MNL-II (or "scaled mixed logit").

```
. /*G-MNL with uncorrelated random coefficients*/
. matrix scale = 1,1,1,0,1,1
. gmnl y knowgp malegp cost, group(gid) id(rid) rand(dummy_test testdue drrec)
> nrep(500) scale(scale) gamma(0)
Iteration 0:   log likelihood = -1414.4896  (not concave)
  (output omitted)
Iteration 19:  log likelihood = -991.41088
Generalized multinomial logit model              Number of obs   =       5056
                                                 Wald chi2(6)    =      67.25
Log likelihood = -991.41088                      Prob > chi2     =     0.0000
                                  (Std. Err. adjusted for clustering on rid)
------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
Mean         |
      knowgp |   .9123367   .1748867     5.22   0.000     .5695651    1.255108
      malegp |  -2.742707   .3543753    -7.74   0.000     -3.43727   -2.048144
        cost |  -.1419785   .0637313    -2.23   0.026    -.2668895   -.0170675
  dummy_test |  -.4904328   .2685654    -1.83   0.068    -1.016811    .0359457
     testdue |    5.79628   .8667601     6.69   0.000     4.097462    7.495099
       drrec |   1.492487   .2652157     5.63   0.000     .9726734      2.0123
-------------+----------------------------------------------------------------
SD           |
  dummy_test |   2.988542   .3213189     9.30   0.000     2.358769    3.618316
     testdue |   3.166774   .4859329     6.52   0.000     2.214363    4.119185
       drrec |   1.356382    .194595     6.97   0.000      .974983    1.737781
-------------+----------------------------------------------------------------
         /tau |   1.177626    .115934    10.16   0.000     .9503993    1.404852
------------------------------------------------------------------------------
The sign of the estimated standard deviations is irrelevant: interpret them as
being positive
. *Save coefficients for later use
. matrix b = e(b)
```

4. Note that we constrain γ to 0 by using the `gamma(0)` option. This is to prevent `gmnl` from attempting to estimate the gamma parameter, because it is not identified in this model.

The square root of the diagonal elements of Σ is estimated and shown in the block under `SD`. All the standard deviations are significantly different from 0, which suggests the presence of substantial preference heterogeneity in the data.

In the last example, we allow the random coefficients of `dummy_test`, `testdue`, and `drrec` to be correlated, which implies that the off-diagonal elements of Σ will not be fixed as zeros. Instead of using the default starting values, we use the parameters from the previous model, setting the starting values for the off-diagonal elements of Σ to 0.

```
. *Starting values
. matrix start = b[1,1..7],0,0,b[1,8],0,b[1,9..10]
. /*G-MNL with correlated random coefficients*/
. gmnl y knowgp malegp cost, group(gid) id(rid) rand(dummy_test testdue drrec)
> nrep(500) from(start,copy) scale(scale) corr gamma(0)
Iteration 0:   log likelihood = -991.41088  (not concave)
  (output omitted)
Iteration 8:   log likelihood =  -987.7783
Generalized multinomial logit model               Number of obs   =       5056
                                                  Wald chi2(6)    =      57.86
Log likelihood =  -987.7783                       Prob > chi2     =     0.0000
                                  (Std. Err. adjusted for clustering on rid)
```

y	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
knowgp	1.016481	.1871831	5.43	0.000	.6496085	1.383353
malegp	-3.082839	.4511159	-6.83	0.000	-3.96701	-2.198668
cost	-.1506127	.0695989	-2.16	0.030	-.2870241	-.0142012
dummy_test	-.5499832	.2755279	-2.00	0.046	-1.090008	-.0099584
testdue	6.372514	.9780339	6.52	0.000	4.455603	8.289425
drrec	2.488203	.5429958	4.58	0.000	1.423951	3.552455
/l11	2.749374	.3800353	7.23	0.000	2.004518	3.494229
/l21	-.155092	.2641671	-0.59	0.557	-.67285	.3626659
/l31	1.116604	.3198175	3.49	0.000	.4897736	1.743435
/l22	3.423451	.5413485	6.32	0.000	2.362427	4.484474
/l32	.719799	.3048254	2.36	0.018	.1223522	1.317246
/l33	1.448777	.2470165	5.87	0.000	.9646339	1.932921
/tau	1.250552	.1531374	8.17	0.000	.9504077	1.550695

The six parameters from `l11` to `l33` are the elements of the lower-triangular matrix L, the Cholesky factorization of Σ ($\Sigma = LL'$). Given the estimate of L, we may recover Σ and the standard deviations of the random coefficients by using `gmnlcov`:

```
. gmnlcov
        v11:  [l11]_b[_cons]*[l11]_b[_cons]
        v21:  [l21]_b[_cons]*[l11]_b[_cons]
        v31:  [l31]_b[_cons]*[l11]_b[_cons]
        v22:  [l21]_b[_cons]*[l21]_b[_cons] + [l22]_b[_cons]*[l22]_b[_cons]
        v32:  [l31]_b[_cons]*[l21]_b[_cons] + [l32]_b[_cons]*[l22]_b[_cons]
        v33:  [l31]_b[_cons]*[l31]_b[_cons] + [l32]_b[_cons]*[l32]_b[_cons] +
> [l33]_b[_cons]*[l33]_b[_cons]
```

y	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
v11	7.559055	2.089718	3.62	0.000	3.463284	11.65483
v21	-.4264059	.7303392	-0.58	0.559	-1.857844	1.005033
v31	3.069963	1.026024	2.99	0.003	1.058993	5.080933
v22	11.74407	3.674788	3.20	0.001	4.541615	18.94652
v32	2.29102	1.166262	1.96	0.049	.0051882	4.576851
v33	3.863872	1.571455	2.46	0.014	.7838763	6.943867

```
. gmnlcov, sd
  dummy_test:  sqrt([l11]_b[_cons]*[l11]_b[_cons])
     testdue:  sqrt([l21]_b[_cons]*[l21]_b[_cons] + [l22]_b[_cons]*[l22]_b[_cons])
       drrec:  sqrt([l31]_b[_cons]*[l31]_b[_cons] +
> [l32]_b[_cons]*[l32]_b[_cons] + [l33]_b[_cons]*[l33]_b[_cons])
```

y	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
dummy_test	2.749374	.3800353	7.23	0.000	2.004518	3.494229
testdue	3.426962	.5361583	6.39	0.000	2.376111	4.477813
drrec	1.965673	.3997244	4.92	0.000	1.182228	2.749119

There are other useful postestimation commands besides `gmnlcov`. For example, to generate predicted probabilities, we may use the `gmnlpred` command:

```
. gmnlpred p_hat, nrep(500)
. list rid gid y p_hat in 1/4
```

	rid	gid	y	p_hat
1.	1	1	0	.51986608
2.	1	1	1	.48013392
3.	1	2	0	.63789177
4.	1	2	1	.36210823

Moreover, if we are also interested in estimating individual log likelihoods, we may use the `ll` option of `gmnlpred`:

```
. gmnlpred loglik, nrep(500) ll
. list rid loglik in 1
```

	rid	loglik
1.	1	-22.560377

Finally, we may use the `gmnlbeta` command to calculate individual-level parameters (Revelt and Train 2000):

```
. gmnlbeta dummy_test testdue drrec, nrep(500) saving(beta) replace
file beta.dta saved
. use beta.dta, clear
. list rid dummy_test testdue drrec in 1/4
```

	rid	dummy_test	testdue	drrec
1.	1	-1.5343953	1.1355702	.36138015
2.	2	-3.1375251	.93454325	.06251802
3.	3	-3.7519709	6.2841202	1.3722745
4.	4	.83317825	1.5324372	.88101046

A file is now created and saved as the `beta.dta` dataset, which contains all the estimated individual β's.

7 Conclusion

In this article, we described the `gmnl` Stata command, which can be used to fit the G-MNL model and its variants. As pointed out in Fiebig et al. (2010), G-MNL is very flexible and nests a rich family of model specifications. In the previous sections, we demonstrated several important models, which are summarized (along with some other useful specifications) below in table 2. This list does not exhaust all the possible models that the `gmnl` routine can estimate. One example is the type of model considered in Fiebig et al. (2011) and Knox et al. (2013), which includes interaction terms between sociodemographic variables and ASCs.

Finally, a word of warning: While we have found that the `gmnl` command can be used successfully to implement a range of model specifications, analysts need to bear in mind that estimation times can be substantial when fitting complex models with large datasets. As discussed in section 5, it may also be necessary to experiment with alternative starting values, number of draws, and estimation algorithms to achieve convergence.

Table 2. Special cases of G-MNL and their Stata commands

Model	Command
MIXL	`gmnl y, group(csid) id(id) rand(x) mixl`
S-MNL	`gmnl y x, group(csid) id(id)`
S-MNL+fixed ASC	`gmnl y asc x, group(csid) id(id) scale(scale)`
S-MNL+random ASC	`gmnl y x, group(csid) id(id) rand(asc) scale(scale) gamma(0)`
G-MNL(uncorrelated)	`gmnl y, group(csid) id(id) rand(x)`
G-MNL(correlated)	`gmnl y, group(csid) id(id) rand(x) corr`
G-MNL(uncorrelated)+fixed ASC	`gmnl y asc, group(csid) id(id) rand(x) scale(scale)`
G-MNL(correlated)+fixed ASC	`gmnl y asc, group(csid) id(id) rand(x) scale(scale) corr`
G-MNL(uncorrelated)+random ASC	`gmnl y, group(csid) id(id) rand(asc x) scale(scale)`
G-MNL(correlated)+random ASC	`gmnl y, group(csid) id(id) rand(asc x) scale(scale) corr`

8 Acknowledgments

We are grateful to a referee and to Kristin MacDonald of StataCorp for helpful comments. The research of Yuanyuan Gu and Stephanie Knox was partially supported by a Faculty of Business Research Grant at the University of Technology in Sydney.

9 References

Drukker, D. M., and R. Gates. 2006. Generating Halton sequences using Mata. *Stata Journal* 6: 214–228.

Fiebig, D. G., and J. Hall. 2004. Discrete choice experiments in the analysis of health policy. *Productivity Commission Conference: Quantitative Tools for Microeconomic Policy Analysis* 6: 119–136.

Fiebig, D. G., M. P. Keane, J. Louviere, and N. Wasi. 2010. The generalized multinomial logit model: Accounting for scale and coefficient heterogeneity. *Marketing Science* 29: 393–421.

Fiebig, D. G., S. Knox, R. Viney, M. Haas, and D. J. Street. 2011. Preferences for new and existing contraceptive products. *Health Economics* 20 (Suppl.): 35–52.

Greene, W. H., and D. A. Hensher. 2010. Does scale heterogeneity across individuals matter? An empirical assessment of alternative logit models. *Transportation* 37: 413–428.

Halton, J. H. 1964. Algorithm 247: Radical-inverse quasi-random point sequence. *Communications of the ACM* 7: 701–702.

Hole, A. R. 2007. Fitting mixed logit models by using maximum simulated likelihood. *Stata Journal* 7: 388–401.

Keane, M., and N. Wasi. Forthcoming. Comparing alternative models of heterogeneity in consumer choice behavior. *Journal of Applied Econometrics*.

Knox, S. A., R. C. Viney, Y. Gu, A. R. Hole, D. G. Fiebig, D. J. Street, M. R. Haas, E. Weisberg, and D. Bateson. 2013. The effect of adverse information and positive promotion on women's preferences for prescribed contraceptive products. *Social Science and Medicine* 83: 70–80.

Louviere, J., and T. Eagle. 2006. Confound it! That pesky little scale constant messes up our convenient assumptions. In *Proceedings of the Sawtooth Software Conference*, 211–228. Sequim, WA: Sawtooth Software.

Louviere, J. J., R. J. Meyer, D. S. Bunch, R. T. Carson, B. Dellaert, W. M. Hanemann, D. Hensher, and J. Irwin. 1999. Combining sources of preference data for modeling complex decision processes. *Marketing Letters* 10: 205–217.

Louviere, J. J., D. Street, L. Burgess, N. Wasi, T. Islam, and A. A. J. Marley. 2007. Modeling the choices of individual decision-makers by combining efficient choice experiment designs with extra preference information. *Journal of Choice Modelling* 1: 128–163.

Louviere, J. J., D. Street, R. Carson, A. Ainslie, J. R. Deshazo, T. Cameron, D. Hensher, R. Kohn, and T. Marley. 2002. Dissecting the random component of utility. *Marketing Letters* 13: 177–193.

McFadden, D., and K. Train. 2000. Mixed MNL models for discrete response. *Journal of Applied Econometrics* 15: 447–470.

Revelt, D., and K. Train. 2000. Customer-specific taste parameters and mixed logit: Households' choice of electricity supplier. Working Paper No. E00-274, Department of Economics, University of California, Berkeley.

Train, K. E. 2009. *Discrete Choice Methods with Simulation*. 2nd ed. Cambridge: Cambridge University Press.

About the authors

Yuanyuan Gu is a research fellow at the Centre for Health Economics Research and Evaluation, University of Technology in Sydney. His recent research focuses on discrete choice modeling with applications in health economics.

Arne Risa Hole is a senior lecturer in economics at the University of Sheffield in the UK. His research interests lie in the area of applied microeconometrics, with a focus on health and labor economics. Since obtaining his PhD, he has been particularly interested in stated preference methods and the econometric analysis of discrete choice data.

Stephanie Knox is a research fellow at the Centre for Health Economics Research and Evaluation, University of Technology in Sydney. Her research interests include the design and application of stated preference methods in the area of health service research.