A MATLAB® Implementation of a Halton Sequence-Based GHK Simulator for Multinomial Probit Models[*]

Jose Miguel M. Abito
Department of Economics
National University of Singapore

April 2005

---

[*] I would like to thank Kurt Schmidheiny for acknowledging my request for his MATLAB® Econometrics Toolbox and Kenneth Train for his online course in UC Berkeley. Nonetheless, all errors are mine.

1 Introduction

The popularity of logit-based models is due to the availability of closed-form expressions for their choice probabilities. Unfortunately, Logit-based models have important limitations particularly the inability to handle random taste variation, general forms of substitution patterns, and repeated choice situations with serially correlated errors (Train, 2003). To remedy these deficiencies, the Multinomial Probit (MNP) model is often used. However closed-form choice probabilities for MNP are not readily available. Numerical integration methods can be utilized to solve these integrals. However, most numerical integration methods are only feasible and practical to use when the dimension of integration is small enough (< 5) because of limitations in computing power. Therefore simulation methods can be employed to address this computational limitation. This is the prime motivation for studying simulation-based methods applied to MNP and other Discrete Choice models with analytically intractable choice probabilities.

In using simulation to estimate the choice probabilities of a high-dimensional MNP model, the first step is to find an appropriate algorithm to use that would provide an estimate with minimal variance relative to the true value of the integral. In general, when estimating an integral, variance reduction techniques such as Antithetic Monte Carlo and Importance Sampling are used to introduce some negative correlation or other “nice” properties among the pseudo-random draws for the simulation. Crude or Non-weighted Monte Carlo techniques that rely solely on pseudo-random generators often perform poorly due to their low equidistribution property (Sandor and Andras, 2004) and high discrepancy (Brandimarte, 2002). With this in mind, different simulators have been used in the MNP context. Among these different methods, the GHK (Geweke, 1989, 1991; Hajivassiliou and McFadden, 1998; Keane, 1990, 1994) simulator is the most widely utilized (Train, 2003) and has been studied and observed to perform better than other simulators (Geweke *et al*., 1994; Borch-Supan and Hajivassiliou, 1993; Hajivassilious *et al*., 1996).

Aside from Monte Carlo based variance reduction techniques, Quasi-Monte Carlo methods can be employed to improve estimation. The main idea behind this is that since the generation of pseudo-random numbers is actually deterministic, the researcher can manipulate and devise alternative deterministic sequences of numbers that display better equidistirbution and low discrepancy properties. Sandor and Train (2002) observe that (t, m, s)-nets and Halton draws perform better than independent draws in an application to maximum simulated likelihood estimation of a mixed logit model. More recently, Sandor and Andras (2004) show that quasi-Monte Carlo samples and samples based on orthogonal arrays have better performance in general, compared to Crude Monte Carlo and Antithetic Monte Carlo samples when used with GHK simulators[1]

---

[1] However, the authors do not examine the “quality” of these GHK simulators by applying it in a MNP model and examining whether estimates using GHK simulators with different samples differ significantly. Thus this paper focuses on the *practicality* of using simulators based on a particular Quasi-Monte Carlo method.

This paper provides a simple and practical incorporation of a quasi-Monte Carlo sample into Kurt Schmidheiny's MNP routine in `MATLAB`® (based on the toolbox used in Schmidheiny [2003]).[2] The quasi-Monte Carlo sample used is the Halton sequence which is the simplest. Specifically, a randomized Halton sequence is incorporated in the GHK simulator of the toolbox. Though more sophisticated and effective randomization techniques for Halton sequences are available (Tuffin, 1996; Wang and Hickernell, 2000), a simple randomization procedure suggested by Train (2003) is used. Lastly, comparisons between a pseudo-random generator based GHK simulator and a randomized Halton sequence-based GHK simulator is carried out using the travel choice data studied in Greene (2003) and Greene and Hensher (1997). Both GHK simulators provide similar estimates even with only few simulation runs. This shows that firstly, the GHK simulator itself performs well enough even with few simulation runs (given that the number of observations is sufficiently large), and secondly, only marginal improvements, if any, can be achieved with a (crude) randomized Halton sequence-based GHK simulator that tend to expend more estimation time.

The next section provides a brief background on how the GHK simulator is constructed in general settings. In Section 3, a non-technical discussion of Halton sequences is given. Section 4 briefly discusses the data used followed by the main results of our experiment. The last section concludes.

2 GHK Simulator

The GHK Simulator is based on Importance Sampling with a recursively truncated multivariate normal as its Importance pdf. The basic idea is to directly approximate a rectangular probability (Gourieroux and Monfort, 1996). As mentioned, the GHK simulator is the most popular simulator for MNP (Train, 2003) and has been observed to perform better than other simulators (Geweke *et al.*, 1994; Borch-Supan and Hajivassiliou, 1993; Hajivassilious *et al.*, 1996).

Consider a general model with J alternatives. The utility of person n from choosing alternative j is given by

$$U_{nj} = V_{nj} + \varepsilon_{nj} \qquad (1)$$

for $j = 1,\dots, J$. After applying the normalization (for identification) method suggested by Train (2003) and expressing (1) as differences against alternative i, we have the following:

---

[2] The main issue is the estimation of choice probabilities using simulation. Issues regarding the full estimation of the MNP model using simulation (whether and how to implement Maximum Simulated Likelihood Estimation or Method of Simulated Moments) are not dealt with in this short paper.

$$\tilde{U}_{nji} = \tilde{V}_{nji} + \tilde{\varepsilon}_{nji} \qquad (1')$$

where $\tilde{\varepsilon}_{ni} = (\tilde{\varepsilon}_{n1i},...,\tilde{\varepsilon}_{nJi})' \sim N(0, \tilde{\Omega}_i)$ and is *(J-1) x 1*. The probability that person *n* chooses alternative *i* will then be given as

$$P_{ni} = \Pr(\tilde{V}_{nji} + \tilde{\varepsilon}_{nji} < 0 \; \forall \; j \neq i) \qquad (2).$$

Note that via Choleski decomposition, $\tilde{\Omega}_i = L_i L_i'$ where[3]

$$L_i = \begin{pmatrix} c_{11} & 0 & \cdots & 0 \\ c_{21} & c_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ c_{J1} & c_{J2} & \cdots & c_{JJ} \end{pmatrix}$$

and consequently (1') can be reinterpreted as

$$\tilde{U}_{ni} = \tilde{V}_{ni} + L_i \eta_n \qquad (1'')$$

where $\tilde{U}_{ni} = (\tilde{U}_{n1i},...,\tilde{U}_{nJi})'$, $\tilde{V}_{ni} = (\tilde{V}_{n1i},...,\tilde{V}_{nJi})'$ and $\eta_n = (\eta_{n1},...,\eta_{nJ})'$. Note also that $\eta_{nj} \sim N(0,1)$. The above transformation allows us to proceed with simulations using independent draws.

Using Bayes' Rule recursively, the choice probability for alternative *i* can then be rewritten conveniently as

$$\begin{aligned} P_{ni} &= \Pr(\eta_1 < \frac{-\tilde{V}_{n1i}}{c_{11}}) \\ &\times \Pr(\eta_2 < \frac{-(\tilde{V}_{n2i} + c_{21}\eta_1)}{c_{22}} \mid \eta_1 < \frac{-\tilde{V}_{n1i}}{c_{11}}) \\ &\times \Pr(\eta_3 < \frac{-(\tilde{V}_{n3i} + c_{31}\eta_1 + c_{32}\eta_2)}{c_{33}} \mid \eta_2 < \frac{-(\tilde{V}_{n2i} + c_{21}\eta_1)}{c_{22}} \cap \eta_1 < \frac{-\tilde{V}_{n1i}}{c_{11}}) \\ &\times \cdots \end{aligned} \qquad (2').$$

The GHK simulator is then calculated as follows (Train, 2003):

1. Calculate $\Pr(\eta_1 < \frac{-\tilde{V}_{n1i}}{c_{11}}) = \Phi(\frac{-\tilde{V}_{n1i}}{c_{11}})$.
2. Draw a value η_1, denoted as η_1^r from a truncated standard normal truncated at $\frac{-\tilde{V}_{n1i}}{c_{11}}$.[4]

---

[3] L_i is *(J-1) x (J-1)*.

[4] Refer to Train (2003) on how to obtain a draw from a truncated standard normal distribution. It is important to note that in drawing from a truncated standard normal, *U(0,1)* draws are used.

3. Calculate $\Pr(\eta_2 < \frac{-(\tilde{V}_{n2i} + c_{21}\eta_1)}{c_{22}} \mid \eta_1 = \eta_1^r) = \Phi(\frac{-(\tilde{V}_{n2i} + c_{21}\eta_1^r)}{c_{22}})$.
4. Draw a value η_2, denoted as η_2^r from a truncated standard normal truncated at $\frac{-(\tilde{V}_{n2i} + c_{21}\eta_1^r)}{c_{22}}$.
5. Calculate
$$\Pr(\eta_3 < \frac{-(\tilde{V}_{n3i} + c_{31}\eta_1 + c_{32}\eta_2)}{c_{33}} \mid \eta_1 = \eta_1^r \cap \eta_2 = \eta_2^r) = \Phi(\frac{-(\tilde{V}_{n3i} + c_{31}\eta_1^r + c_{32}\eta_2^r)}{c_{33}})$$
6. Repeat steps 1-5 for all alternatives except *i*.
7. The simulated probability for the *rth* draw of $\eta_1, \ldots \eta_J$ is given as
$$\breve{P}_{ni}^r = \Phi(\frac{-\tilde{V}_{n1i}}{c_{11}}) \times \Phi(\frac{-(\tilde{V}_{n2i} + c_{21}\eta_1^r)}{c_{22}} \times \Phi(\frac{-(\tilde{V}_{n3i} + c_{31}\eta^r_{1\,1} + c_{32}\eta_2^r)}{c_{33}}) \times \cdots$$
8. Repeat steps 1-7 for *r = 1,…, R* where *R* is the number of replications for the GHK simulator.
9. Finally, the simulated choice probability is $\breve{P}_{ni} = \frac{1}{R}\sum_{r=1}^{R} \breve{P}_{ni}^r$.

This algorithm is implemented in Schmidheiny's `MATLAB`® toolbox.

3 Halton Sequence

This section provides a non-technical introduction to Halton Sequences. For a more technical treatment, consult the references cited in Sandor and Andras (2004). Consider a one-dimensional Halton sequence in base 2. Divide the unit line into equal partitions corresponding to the base chosen (*i.e.* cut in half). The first element of the sequence is 2^{-1}. Add 2^{-2} to 0 and 0.5 to get the second and third elements of the sequence respectively. Our sequence now is {0.5, 0.25, 0.75}. Add 2^{-3} to zero and each element of the sequence in proper order. Our sequence is now {0.5, 0.25, 0.75, 0.125, 0.625, 0.375, 0.875}. Repeat the process until one has the desired number of elements.

The main rationale for choosing a Halton sequence over a simple pseudo-random generator is that we want our draws to mimic a uniform distribution as much as possible. A possible way to do this is to introduce some negative correlation between successive draws so that each side of the distribution is represented almost equiproportionally. This is the same working idea for using antithetic Monte Carlo techniques. With the Halton sequence, each successive draw jumps from one side of the distribution to the other.

One measure of performance for pseudo-random or deterministic samples that approximate the continuous uniform distribution is Discrepancy. Mathematically, Discrepancy is defined as

$$D(\mathbf{x}^1, \ldots, \mathbf{x}^n) = \sup_{\mathbf{x} \in I^m} | S_n(G_{\mathbf{x}}) - n x_1 \cdots x_m |$$

where I^m is an m-dimensional hypercube, G_X is a subspace of I^m, and $S_n(G_X)$ is a function that counts the number of sample points contained in each subspace. Lower discrepancy means that our sample points are more dispersed in I^m which is our desired characteristic.

Consider a comparison of the pseudo-random generator of `MATLAB`® (`rand()`) and a (randomized) Halton sequence in a two-dimensional space:[5]

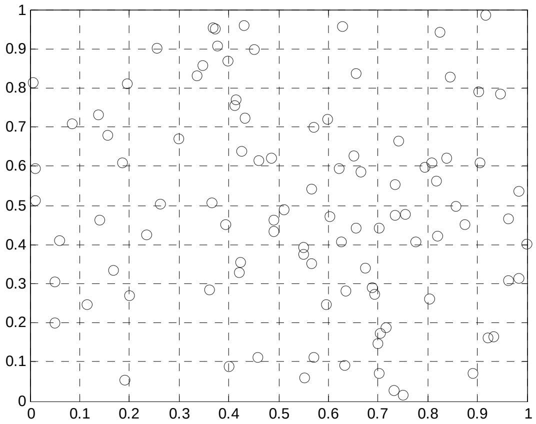

Figure 1a: Pseudo-Random Generator

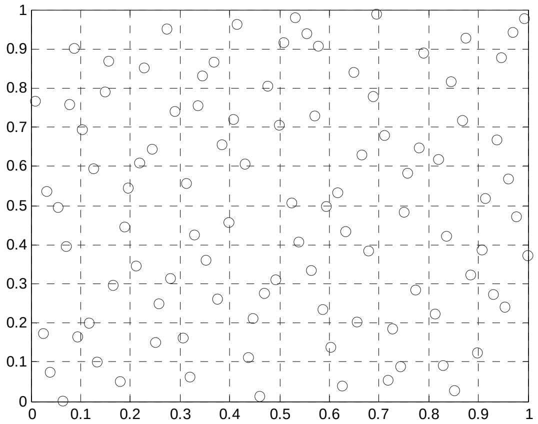

Figure 1b: (Randomized) Halton Sequence

By visual inspection, Figure 1a has higher discrepancy compared to Figure 1b. In other words, the Halton sequence is more dispersed across the region compared to the simple pseudo-random generator and therefore we expect that the former will perform better than the latter in estimation via simulation.

4 Data and Results

Schmidheiny's `MATLAB`® toolbox contains the data set and applies the MNP routine using the data set as a demo. The data set originally comes from Greene and Hensher (1997) and is closely studied in

---

[5] The two-dimensional randomized Halton sequence is of base 2 and 3. See the appendix for the `MATLAB`® code.

Greene (2003). The data set contains 840 observations on 4 travel modes for 210 individuals. The 4 modes consist of `Air`, `Train`, `Bus` or `Car`. The choice attributes in the model are `gc` (generalized cost measure), `ttime` (terminal waiting time), `hinc` (household income), and `hinc2air` (household income interacted with air dummy). Estimation is performed using maximum simulated likelihood. The MSLE results are given below:

Table 1: MSLE Results

	A	B	C	D	E
Constants					
`Air`	0.490338	0.393034	0.237583	0.907016	0.791799
	(0.665986)	(0.609776)	(0.699832)	(0.603322)	(0.688239)
`Train`	1.073865***	0.920257***	0.984482***	1.161474***	1.157948***
	(0.327818)	(0.316232)	(0.358488)	(0.266942)	(0.303521)
`Bus`	0.901261***	0.775379***	0.827658***	0.967196***	0.980992***
	(0.298812)	(0.277521)	(0.320735)	(0.253812)	(0.294183)
Slope Coefficients					
`gc`	-0.008676***	-0.008022***	-0.008299***	-0.009196***	-0.008777***
	(0.001937)	(0.001861)	(0.002240)	(0.001676)	(0.001723)
`ttime`	-0.020413**	-0.017641**	-0.017224**	-0.023799***	-0.023548***
	(0.007984)	(0.007142)	(0.008391)	(0.007210)	(0.007904)
`hinc2air`	0.013081**	0.010907*	0.014265**	0.008765	0.010879*
	(0.006459)	(0.006444)	(0.006303)	(0.005848)	(0.006398)
# of Observations (Individuals)	210	210	210	210	210
Type of draws	**`rand()`**	**`rand()`**	**Halton**	**`rand()`**	**Halton**
# of GHK Replications	1000	10	10	2	2
Substitution Pattern	Unrestricted, Normalized	Unrestricted, Normalized	Unrestricted, Normalized	Unrestricted, Normalized	Unrestricted, Normalized
Computational Time (in minutes)	59.79	4.82	11.31	4.57	6.75

Significance Levels: 1% (***), 5% (**), 10% (*)
Note: Normalization algortihm for unrestricted substitution pattern is based on Train (2003) and is implemented in Schmidheiny's toolbox

Models A, B and D are based on the pseudo-random generator `rand()` and Model A is used as the benchmark model. Models C and E utilize draws from randomized Halton sequences of *k-1 x r* dimension where *k* is the number of alternatives (corresponds to the number of bases used) and *r* is the number of replications (corresponds to the number of elements in each base). With a few minor exceptions, all of the models have similar estimation results which can mean two things. First, the (ordinary) GHK simulator is robust enough to handle extremely small replications and exhibits the same performance as a Halton sequence-based GHK simulator. Second, our experiment is not entirely conclusive from a *theoretical* point of view since there seems to be some divergence in terms of our objective and the measure of performance used in the analysis and comparison of the two samples. Specifically, our objective is to see how well the different samples actually estimate the choice probabilities but the yardstick of performance is based on estimated coefficients for the choice attributes. We are actually looking at how well these samples yield an estimate via MSLE which is affected by the number of observations used in the study and not simply on the number of replications. With a large enough sampled population, the estimate will be rather precise even if the number of replications is small (Train, 2003). Nonetheless from a *practical* point of view, if the researcher has a sufficiently large number of observations and the only issue is the number of replications to be used, then an ordinary GHK simulator performs as well as a Halton-sequence based GHK simulator[6], as mentioned previously.

5 Conclusion

As we have seen in the previous section, a GHK simulator performs as well as a Halton sequence-based GHK simulator, assuming the researcher has enough observations. Though a randomized Halton sequence is easy to incorporate in any programming routine for MNP models (see the Appendix), a considerable amount of computing time is expended during estimation and thus may not be worth it (again, given that we have a sufficiently large number of observations). Nevertheless, other quasi-Monte Carlo samples and samples based on orthogonal arrays can be incorporated. How these samples actually perform in practical applications still remains to be studied.

---

[6] Given that the number of observations is sufficiently large, using a Halton sequence-based GHK simulator has only marginal improvements with respect to the estimates but takes considerably more time to perform the estimation. Therefore, on these grounds and as explained above, an ordinary GHK simulator would be most likely preferred in practical applications.

Appendix

The randomized Halton sequence generator can be easily incorporated into the GHK simulator by replacing the function that calls `rand()` with a function that gives us a randomized Halton sequence of size *k-1 x r*.[7] The `MATLAB`® code for generating randomized Halton sequences is as follows:

```
function r = RandHaltVec(m,n)
% Returns a matrix of random Halton draws of size m x n (n draws
from m bases)
r = [];
vP = GeneratePrime(m);
for i = 1:m
    r(i,:) = RandomHalton(n,vP(i));
end

function r = RandomHalton(m,b)
% Randomizes Halton draws with base b and m elements
Halton_nonR = GetHalton(m,b);
u = rand(1,1);
% The following ensures that our randomized Halton element is within
% (0,1):
for i = 1: m
    if Halton_nonR(i) + u < 1;
        rt(i) = Halton_nonR(i) + u;
    else
        rt(i) = Halton_nonR(i) + u - 1;
    end
end
r =rt';
```

where `GetHalton(m,b)` is a function that provides a Halton sequence in base *b* and with size *m x 1* (Brandimarte, 2002). The function `RandomHalton(m,b)`is based on a randomizing procedure suggested by Kenneth Train in his online course in UC Berkeley[8].

---

[7] This occurs when we draw a sample point from a truncated standard normal.

[8] http://elsa.berkeley.edu/~train/distant.html

References

Borsch-Supan, A. and V. Hajivassiliou (1993), “Smooth Unbiased Multivariate Probability Simulation for Maximum Likelihood Estimation of Limited Dependent Variable Models,” *Journal of Econometrics* **58**, 347-368.

Brandimarte, P. (2002), *Numerical Methods in Finance: A MATLAB®-Based Introduction*, Wiley & Sons, NY.

Geweke, J. (1989), “Bayesian inference in econometric models using Monte Carlo integration,” *Econometrica* **57**, 1317–1339.

Geweke, J. (1991), “Efficient simulation from the multivariate normal and Student-*t* distributions subject to linear constraints,” in E. M. Keramidas, ed., *Computer Science and Statistics: Proceedings of the Twenty-Third Symposium on the Interface*, pp. 571–578. Interface Foundation of North America Inc., Fairfax.

Geweke, J., M. Keane, and D. Runkle (1994), “Alternative Computational Approaches to Inference in the Multinomial Probit Model,” *Review of Economics and Statistics* **76**, 609-632.

Gourieroux, C. and A. Monfort (1996), *Simulation-Based Econometric Methods*, Oxford University Press, NY.

Greene, W (2003), *Econometric Analysis*, 5th Ed., Prentice Hall, NJ.

Greene, W. and D. Hensher (1997), “Multinomial Logit and Discrete Choice Models,” in Greene, W., *LIMDEP Version 7.0 User’s Manual, Revised*, Econometric Software Inc., NY.

Hajivassiliou, V. and D. McFadden (1998), “The method of simulated scores for the estimation of LDV models,” *Econometrica* **66**, 863–896.

Hajivassiliou, V., D. McFadden and P. Ruud (1996), “Simulation of Multivariate Normal Rectangle Probabilities and Their Derivatives: Theoretical and Computational Results,” *Journal of Econometrics* **72**, 85-134.

Sandor, Z. and P. Andras (2004), “Alternative Sampling Methods for Estimating Multivariate Normal Probabilities,” *Journal of Econometrics* **120**, 207-234.

Sandor, Z. and K. Train (2002), “Quasi-Random Simulation of Discrete Choice Models,” *mimeo.*

Schmidheiny, K. (2003), “Income Segregation and Local Progressive Taxation: Empirical Evidence from Switzerland,” University of Bern, *mimeo.*

Train, K. (2003), *Discrete Choice Methods with Simulation,* Cambridge University Press, NY.

Tuffin, B. (1996), “On the Use of Low-Discrepancy Sequences in Monte Carlo Methods,” *Monte Carlo Methods and Applications* **2**, 295-320.

Wang, X. and F. J. Hickernell (2000), “Randomized Halton Sequences,” *Mathematical and Computer Modelling* **32**, 887-899.