

Quasi-Random Simulation of Discrete Choice Models

by

Zsolt Sándor and Kenneth Train

Erasmus University Rotterdam and University of California, Berkeley

June 12, 2002

Abstract

We describe the properties of (t, m, s) -nets and Halton draws. Four types of (t, m, s) -nets, two types of Halton draws, and independent draws are compared in an application of maximum simulated likelihood estimation of a mixed logit model. All of the quasi-random procedures are found to perform far better than independent draws. The best performance is attained by one of the (t, m, s) -nets. The properties of the nets imply that two of them should out-perform the other two, and our results confirm this expectation. The two more-accurate nets perform better than both types of Halton draws, while the two less-accurate nets perform worse than the Halton draws.

1 Introduction

A choice probability is the expectation of a statistic over a density: $P = \int g(\varepsilon)f(\varepsilon)d\varepsilon$, where f is the density of random factors ε and g is a calculable function. This expectation is simulated by evaluating g at R values of ε and averaging the results. If each evaluation point is drawn independently from f , then the simulated probability is unbiased and consistent for the true probability, with variance inversely proportional to R . In maximum simulated likelihood (MSL) estimation, the simulation induces both bias and variance, with the bias arising from the log transformation of the simulated probability. If R rises faster than the square root of the number of observations, then the effects of simulation disappear asymptotically, and MSL is equivalent to maximum likelihood with exact probabilities (McFadden, 1989; Lee, 1995). However, for fixed R , simulation bias and variance are necessarily present.

Instead of taking independent draws, simulation can potentially be improved by selecting evaluation points more systematically. “Quasi-random” draws are designed to provide better coverage than independent draws over the density for which the integral is defined. This improved coverage usually translates into smaller expected approximation error. In the context of MSL, where bias arises along with variance, quasi-random draws have the potential to reduce the root-mean-squared-error (RMSE) of the estimates against those that would be obtained with the infeasible exact probabilities.

Two important types of quasi-random draws are those based on Halton sequences and (t, m, s) -nets, defined below. Halton draws have been examined by Bhat (2001; forthcoming) and Train (2000; 2002) in the context of MSL on discrete choice models and have been found to provide far more accuracy than a comparable number of independent draws. Sándor and András (2001) examined (t, m, s) -nets in the calculation of probit probabilities and found them to perform well relative to Halton and independent draws. They did not, however, examine their use in estimation of model parameters.

The purpose of the current paper is to examine the performance of (t, m, s) -nets relative to Halton and independent draws in MSL estimation. To our knowledge, it is the first such comparison. We conduct a Monte Carlo experiment using data on consumers’ choice of vehicle within a mixed logit specification. We compare the RMSE of the estimated parameters for four kinds of (t, m, s) -nets, two kinds of Halton draws, and independent draws. We find, in summary, that the best (t, m, s) -nets perform better than Halton draws, and that both types of draws perform far better than independent draws. Also, the performance of the (t, m, s) -nets relative to each other corresponds to expectations, given the properties of these nets.

2 Halton draws

We first define a Halton sequence in one dimension and then show how a sequence in multiple dimensions is created. We next describe procedures for introducing randomness, since Halton sequences are deterministic.

2.1 One dimension

A Halton sequence is defined in terms of a base. The sequence in base 10 is most easily explained; sequences in other bases are created the same as in base 10 except with conversion to and from the new base as initial and final steps.

A Halton sequence in base 10 is created in two simple steps:

1. List the integers: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, ...
2. From each integer, create a decimal number by reversing the digits in the integer and putting them after a decimal point. That is, 1 becomes .1, 12 becomes .21, 256 becomes .652. The sequence is now: 0, .1, .2, .3, .4, .5, .6, .7, .8, .9, .01, .11, .21, .31, ...

Note that the sequence cycles through the unit interval every ten elements. That is, the sequence consists of successive subsequences of length 10 with the elements in each subsequence rising in value and the first element of a subsequence being lower than the last element of the immediately previous subsequence. Note also that successive elements in each subsequence are spaced the same distance apart ($1/10$ apart to be precise).

A Halton sequence in any other base is created by converting to this base before step 2 and converting back to base 10 after step 2. The steps for a Halton sequence in base 2 are:

1. List the integers (in base 10): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, ...
2. Convert these integers to base 2. The base-2 integers are: 0, 1, 10, 11, 100, 101, 110, 111, 1000, 1001, 1010, 1011, 1100, 1101, ...
3. From each base-2 integer, create a base-2 decimal number by reversing the digits in the integer and putting them after the decimal point. The sequence becomes: 0, .1, .01, .11, .001, .101, .011, .111, .0001, .1001, .0101, .1101, .0011, .1011, ...
4. Convert these base-2 decimal numbers back to base 10: 0, $1/2$, $1/4$, $3/4$, $1/8$, $5/8$, $3/8$, $7/8$, $1/16$, $9/16$, $5/16$, $13/16$, $3/16$, $11/16$, ... In general, this conversion is calculated as $\sum_{k=1}^m d_k/b^k$ where d_k is the k -th digit after the decimal and b is the base.

The sequence cycles every b elements, and each element in the cycle is $1/b$ units apart.

2.2 Multiple dimensions

The Halton sequence just described provides points on the unit line. Halton sequences are created in multiple dimensions by using a different base for each dimension. For example, a Halton sequence in two dimensions can be

created from the Halton sequences in bases 10 and 2. The sequence is: $(0,0)$, $(.1,.5)$, $(.2,.25)$, $(.3,.75)$,...

The first dimension cycles every 10 elements and the second dimension cycles every 2 elements. Since 10 is a multiple of 2, the cycles overlap and every pair that rises in the second dimension also rises in the first dimension. As a result, the elements are correlated over the two dimensions. To prevent the overlapping of cycles, Halton sequences in multiple dimensions are usually based on prime numbers only, such that none is a multiple of another. For our application, a sequence in five dimensions was created from the Halton sequences for bases 2, 3, 5, 7 and 11.

2.3 Randomization

Randomness can be introduced to a Halton sequence in several ways. Wang and Hickernell (2000) suggest a random start procedure. Draw an integer randomly between 0 and some large K and label the draw N_0 . For a Halton sequence of length L , create a sequence starting at 0 of length $N_0 + L$ and discard the first N_0 elements. Or, stated differently, create a Halton sequence starting at integer N_0 in step 1 above.

In estimation, a set of evaluation points is used for each observation within the sample. Halton sequences with random starting points can be constructed in two ways for estimation. A “short” sequence can be created for each observation by randomly choosing a starting integer separately for each observation. With this method, which we label HS, each observation has its own randomized Halton sequence. Alternatively, one long sequence for the entire sample can be created from a randomly chosen starting point. With N observations and R evaluation points for each observations, a sequence of length $N * R$ is created from a randomly chosen starting point. Each subsequence of length R is assigned to each observation. We label this procedure HL. The potential advantage of HL arises because Halton sequences are created such that each subsequent point fills in an area that had not been covered by previous points. With one long sequence, the evaluation points for each observation can be self-correcting over observations, as discussed by Train (2000, 2002). The disadvantage of HL is that it contains less randomness than HS.

Tuffin (1996) proposes an alternative randomization procedure for Halton sequences, which is discussed and utilized by Bhat (forthcoming). The Halton sequence is shifted by adding a draw from a standard uniform distribution to each element of the sequence. For any element c of the original sequence in one dimension, the new element is $k = c + \mu$ if $c + \mu < 1$ and

$k = c + \mu - 1$ if $c + \mu \geq 1$, where μ is a draw from a standard uniform. Each element is shifted up by μ , and if this shifting pushes the element to the end of the unit line (i.e., to 1), then the shifting “wraps around” back to the beginning of the line (i.e., to 0) and continues. For Halton sequences in multiple dimensions, a separate draw is used for each dimension.

In estimation on a sample of observations, Tuffin’s procedure can be applied to a sequence for the entire sample (analogous to HL for a random starting point.) However, if one sequence is created and then randomly shifted separately for each observation (analogous to HS), the relations among the points in each dimension remain the same for all observations. For this reason, we use the random start procedure in our application.

A randomized Halton sequence is a set of draws from the uniform distribution. To obtain draws from density f with cumulative distribution F , each element c is transformed as $F^{-1}(c)$. This inverse-cumulative transformation assumes independence over dimensions; this independence can always be assured by placing the covariance terms within g rather than f .

3 (t, m, s) -nets in base b

The defining terms of a (t, m, s) -net in base b are most readily understood in the context of an example. In our application, there are five dimensions of integration, and we are using 64 evaluation points for each observation. The term s is the dimension of the space, which in our case is 5. The term m relates to the number of points that are contained in the net. In our case, we have 64 points. For base $b=2$, 64 can be written as 2^6 . The term m is the power to which the base is raised to obtain the length of the sequence, in this case 6. Stated equivalently, a (t, m, s) -net in base b has length b^m . There is a reason for defining the net in terms of m instead of simply the length of the sequence. A (t, m, s) -net can only be constructed for lengths that are some power of the net’s base. A (t, m, s) -net in base 2 can only have length 2, 4, 8, 16, 32, 64, 128, etc.; it cannot have a length of, say, 100. In this regard, (t, m, s) -nets are less flexible than Halton sequences, since a Halton sequence can have any length. A net of length 64 can be created in base 4 as well as base 2, with $m = 3$ such that $4^3 = 64$. In our application, we will utilize nets of the form $(t, 6, 5)$ in base 2, $(t, 3, 5)$ in base 4, and $(t, 2, 5)$ in base 8. We defer the explanation of t until later.

In one dimension, (t, m, s) -nets are created the same as Halton sequences but with an extra step. Recall that in step 3, the sequence is expressed in decimals; for example, in base 2, the decimal sequence is 0, .1, .01, .11, .001,

.101, .011, .111, .0001, .1001, .0101, .1101, .0011, .1011, For a (t, m, s) -net, each element of this sequence of decimals is transformed in a particular way. Consider the fourth element, .110. Let c be the vector comprised of the digits of this decimal. That is, $c = \langle 1, 1, 0 \rangle$. A new vector, k is created by the transformation $k = Mc$ where M is a “generating matrix” and the matrix multiplication is performed modulo 2 (so that k has elements that take 0 or 1). For example, suppose the transformation matrix is

$$M = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

Then the digits of the fourth element of the sequence are transformed to become

$$k = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

where the top element uses the fact that $1 + 1$ modulo 2 is 0 (since 2 divided by 2 has no remainder.) The fourth element is changed from .110 to .010.

Nets in multiple dimensions are created by using the same base for all dimensions but applying a different generating matrix in each dimension. In contrast, a Halton sequence uses a different base in each dimension but the same generating matrix for all dimensions (namely, the identity matrix, which doesn’t change the digits). A Halton sequence is not a (t, m, s) -net since its dimensions are defined in different bases whereas all dimensions of a (t, m, s) -net are defined in the same base. The desirable properties of a (t, m, s) -net, discussed below, arise from the construction of appropriate generating matrices.

The steps for a (t, m, s) -net in base 2 are:

1. List the integers (in base 10): 0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12 ,13, ...
2. Convert these integers to base 2. The base-2 integers are: 0, 1, 10, 11, 100, 101, 110, 111, 1000, 1001, 1010, 1011, 1100, 1101,...
3. From each base-2 integer, create a base-2 decimal number by reversing the digits in the integer and putting them after the decimal point. The sequence becomes: 0, .1, .01, .11, .001, .101, .011, .111, .0001, .1001, .0101, .1101, .0011, .1011,...
4. Transform each element of the sequence by $k = Mc$, using a different M for each dimension. The arithmetic in this calculation is performed modulo 2.

5. Convert these base-2 decimal numbers back to base 10.

For interpretation, consider a (0,2,2)-net in base 2. This net has $2^2 = 4$ elements in 2 dimensions. For this illustration, t is set at 0, which provides the best coverage, as we will see. Each dimension is divided into 2 equal segments (where 2 is the base), creating four squares of size $(1/2) \times (1/2) = 1/4$. This division is shown by the solid lines in Figure 1. Then each of the segments in each dimension is divided into two equal subsegments, creating four smaller squares within each of the larger squares, and a total of 16 squares within the unit square. The goal is to place the 4 points in the unit square in a way that gives the best coverage. There are 16 small squares and only 4 points to allocate among them. What provides the best coverage? The points in Figure 1 constitute a (0,2,2)-net, which has the following desirable coverage properties. First, there is one point in each of the four large squares, and so the four points provide as good coverage over the two dimensions as possible with four points. Second, in each dimension, there is one point in each of the four subsegments along that dimension (that is, between 0 and $1/4$, between $1/4$ and $1/2$, and so on). The points therefore obtain even coverage over each dimension considered separately.

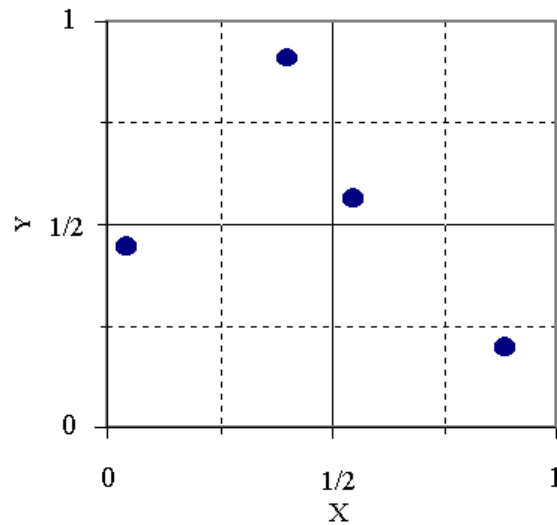


Figure 1: (0,2,2)-net in base 2.

The coverage properties can be stated more precisely. Each of the four

larger squares has size $1/4$ and contains one point. Each tall rectangle, of height 1 and width $1/4$, has size $1/4$ and contains one point. And each long rectangle, of height $1/4$ and length 1, has size $1/4$ and contains one point. The situation can be described in a way that facilitates generalization. The unit square can be partitioned into subspaces (squares or rectangles) in a variety of ways. Consider any partitioning that satisfies the following conditions: each dimension is segmented into 1, 2 or 4 equal parts (of length 1, $1/2$, or $1/4$), with possibly a different number of segments in each dimension, but with each subspace created by the segmentation having a size of $1/4$. Under any such partitioning, the net contains exactly one point in each subspace.

We can now define t . A (t, m, s) -net in base b contains exactly b^t points in each s -dimensional subspace if the subspaces are created by segmenting each dimension into b^k segments of $1/b^k$ length for some $k = 0, \dots, m$ and have volume b^{t-m} . Note that there are b^m/b^t subspaces in any such partitioning, such that, with b^t points in each subspace, the total number of points is b^m . In our $(0,2,2)$ -net in base 2, $t = 0$, such that exactly $2^0 = 1$ point lies in each subspace of size $2^{0-2} = 1/4$ when the subspaces are created by (i) segmenting each dimension into two segments of length $1/2$, such that each subspace is a square of size $1/2$ by $1/2$, or (ii) segmenting the x-dimension into 4 parts of length $1/4$ and the y-dimension into 1 part of length 1, such that each subspace is a tall rectangle of size $1/4$, or (iii) segmenting the x-dimension into 1 part and the y-dimension into four parts, creating long rectangles of size $1/4$. (Stated alternatively: the product of the number of segments over dimensions must equal b^{m-t} . With $b = 2, t = 0$ and $m = 2$, the product must be 4. The possibilities are $2*2$, $1*4$, and $4*1$.)

The coverage properties of different nets can be compared in two ways. First, for given m, s and b , a lower t gives better coverage, since the partitioning consists of more subspaces and fewer points in each subspace. (The placement of points within each subspace is not designed to provide good coverage within the subspace, and so better coverage of the unit space is attained with fewer points in more subspaces.) With $t = 0$, each appropriately defined subspace contains exactly one point. However, a t of 0 is not attainable for all values of m, s and b . The development of methods for creating (t, m, s) -nets, discussed below, has been motivated by (or at least can be explained by) the desire for nets with lower values of t for given values of the m, s and b .

The second comparison relates to the fact that a net with N points can often be created with different bases; for example, a net with 64 points can be obtained with $b = 2$ and $m = 6$, $b = 4$ and $m = 3$, or $b = 8$ and $m = 2$.

It is not necessarily the case that the same value of t can be attained for each of these nets. However, if the same value of t can be attained with different values of b and m , then the coverage properties of the nets can be compared. In particular, for given s, t and number of points b^m , a net with higher m and correspondingly lower b obtains better coverage. This is most readily explained in terms of our application. With 64 points in five dimensions, a t of 0 is attainable with bases 4 and 8: a $(0, 3, 5)$ -net in base 4, which we label N1, and a $(0, 2, 5)$ -net in base 8, which we label N2. N1 provides better coverage than N2, for the following reason. Both nets contain 1 point in each appropriately defined subspace of volume $1/64$. With N1 these subspaces are defined by any of the following dimensions: (a) $1/4$ for three edges and 1 for two edges, (b) $1/16$ for one edge, $1/4$ for one edge, and 1 for three edges, or (c) $1/64$ for one edge and 1 for the other four. N2 defines the subspaces as (a) $1/8$ for two edges and 1 for three edges, or (b) $1/64$ for one edge and 1 for the other four. With N1, the space can be partitioned in three different ways and get a point in each subspace; while with N2, the space can be partitioned in only two different ways and get a point in each subspace.

A few words about the history of (t, m, s) -nets are useful to place the nets that we use in our application in context. Sobol (1967) developed a type of (t, m, s) -net in base 2; for Sobol nets, the value of t depends on s . Faure (1982) developed (t, m, s) -nets in any prime b , so long as $b \geq s$. For these, $t = 0$. Niederreiter (1987) generalized Faure's procedure to powers of primes, for $b \geq s$ with still $t = 0$. Niederreiter (1988) then generalized the method further to apply to prime powers $b < s$. These nets (based on the 1988 generalization) are called Niederreiter nets. When $b \geq s$, Niederreiter nets are the same as the earlier Niederreiter (1987) nets, and when b is a prime as well as being $\geq s$, Niederreiter nets are Faure nets. When $b < s$, the minimum attainable t in Niederreiter nets depends on b and s . For $b = 2$, Niederreiter nets differ from Sobol nets, but have at least as good coverage. In particular, for $s \leq 7$, the value of t is the same for Sobol and Niederreiter nets, while for $s > 7$, a lower t and hence better coverage is attainable with Niederreiter nets than Sobol nets (Niederreiter, 1988).

Even more flexibility in creating nets is provided by the fact that one extra dimension can always be added to any of the nets created by the above-cited procedures. That is, an $(t, m, s+1)$ -net in base b can be created from a (t, m, s) -net in base b , as follows. Each of the $N = b^m$ points in the net in s dimensions is a vector with s elements. Add a final $s+1$ -th element to each vector, with this final element being n/N , $n = 0, \dots, N-1$. For example, a $(0,3,5)$ -net in base 4 can be obtained by creating a Niederreiter $(0,3,4)$ -net

in base 4. Since $b = s$, this Niederreiter net has $t = 0$. A $(0,3,5)$ -net in base 4 is then created by adding $n/64, n = 0, \dots, 63$ as a final element to each of the N points.

More recently, Niederreiter and Xing (1996) developed nets with t the same or lower than is attainable with Niederreiter nets. The possibility of lower t arises only when $b < s$, since $t = 0$ for Niederreiter nets with $b \geq s$. Pirsic (2002) provides an implementation procedure for Niederreiter-Xing nets with $b = 2$ and various values of s . Niederreiter-Xing nets have not yet been implemented for other bases. In our application, we use the $(2,6,5)$ -net in base 2 that Pirsic provides, which we label N3. We also use a Niederreiter $(3,6,5)$ -net in base 2, which we label N4. Note that $t = 3$ is the smallest value of t that is attainable for a Niederreiter $(t,6,5)$ -net in base 2, while $t = 2$ is attainable in a Niederreiter-Xing net with the same m, s and b . This is an example of how Niederreiter-Xing nets can attain better coverage than Niederreiter nets with the same b, s and m .

To summarize, our application uses 64 points, which allows us to have nets in base 2, 4, and 8. The nets that we use are:

- N1: a $(0,3,5)$ -net in base 4, created from a Niederreiter $(0,3,4)$ -net in base 4 by adding an element $n/64$,
- N2: a $(0,2,5)$ -net in base 8, created by Niederreiter's procedure.
- N3: a $(2,6,5)$ -net in base 2, created by Niederreiter-Xing's procedure.
- N4: a $(3,6,5)$ -net in base 2, created from a Niederreiter $(3,6,4)$ -net in base 2 by adding an element $n/64$.

The generating matrices for these nets are given in the appendices. The matrices for N1, N2, and N4 are derived from the implementation procedure that Bratley, Fox and Niederreiter (1992) give for Niederreiter nets. The matrices for N3 are obtained from Pirsic's implementation of Niederreiter-Xing nets.

We expect N1 to perform better than N2, since the former has higher m and correspondingly lower b than the latter, while t, s and b^m are the same. We expect N3 to perform better than N4 since it has a lower t and yet the same values of s, b and m .

Nets N1 and N2 can be characterized as orthogonal array-based Latin hypercubes (Tang, 1993), as explained by Sándor and András (2001). Any $(0, 2, s)$ -net is an orthogonal array-based Latin hypercube (OALH) of strength 2, and vice versa. Any $(0, 3, s)$ -net is also an OALH, but the reverse is not necessarily true. For example, the definition of an OALH does not guarantee

that each property of a $(0,3,5)$ -net in base 4 is satisfied because the rectangles of size $1/4$ by $1/16$ are not guaranteed to contain exactly one point in the OALH.

3.1 Randomization

Owen (1995) suggests a procedure for randomizing a (t, m, s) -net that retains the coverage properties of the net. The randomization consists of, first, randomly re-ordering the numbers in the base, and, second, shifting each element by a random amount. Consider first the re-ordering of numbers. In base 2, there are two numbers, 0 and 1, and there are two possibilities for their ordering: either 1 is “larger” than 0, or 0 is “larger” than 1. An equal probability is given for each possible ordering. An ordering is chosen randomly and the numbers are changed to reflect this new ordering. For example, a draw from a standard uniform is taken. If the draw is below .5, the first ordering is used, and the 1’s and 0’s are left the way they are. If the draw is above .5, the second ordering is used, and, to reflect this ordering, the 1’s are changed into 0’s and the 0’s into 1’s. This reordering is performed on each digit of the decimals from step 3, sequentially for each successive digit such that the reordering of each digit depends on the previous digits. The reason why this is so will be apparent in the graphical interpretation below. For example, the first eight elements of the sequence in base 2 from step 3 are: .000, .100, .010, .110, .001, .101, .011, .111. We take a draw from a uniform to determine the ordering for the first digit. If .632 is drawn, then the 1’s in the first digit are changed to zeros and vice versa. The sequence becomes: .100, .000, .110, .010, .101, .001, .111, .011. For the second and third digit we apply different random reorderings if the previous digits are different. For reordering the second digit we use two different random reorderings depending on whether the first digit is 0 or 1. If the two draws corresponding to 0 and 1 are 0.115 and 0.821, respectively, and the third digit is not changed, then the sequence becomes: .110, .000, .100, .010, .111, .001, .101, .011. For reordering the third digit we apply four different random reorderings depending on whether the first two digits are 00, 01, 10 or 11. For nets in multiple dimensions, the random reordering is applied to each dimension independently.

This randomization has a graphical interpretation. The points in Figure 2 are created using the procedure described in the previous section, prior to randomization. Each point is at the lower-left corner (i.e., origin) of a $1/4$ by $1/4$ subspace; we discuss later how these points are moved into the interior of the subspaces. Randomization changes the placement of the

16 subspaces and hence the placement of these points. In Figure 2, the first dimension is the x-axis. Changing the 1's to 0's and vice versa in the first digit is equivalent to interchanging the two tall $1/2 \times 1$ rectangles, which gives Figure 3. Changing the 1's to 0's and vice versa in the second digit is equivalent to interchanging the $1/4 \times 1$ rectangles within each of the $1/2 \times 1$ rectangles, as shown in Figure 4. Now we can see that using two different random reorderings if the first digit takes 0 and 1 ensures that the random interchange of the $1/4 \times 1$ rectangles within the first $1/2 \times 1$ rectangle is independent of the random interchange of the $1/4 \times 1$ rectangles within the second $1/2 \times 1$ rectangle. The same process is then applied to the other dimension (to the long rectangles).

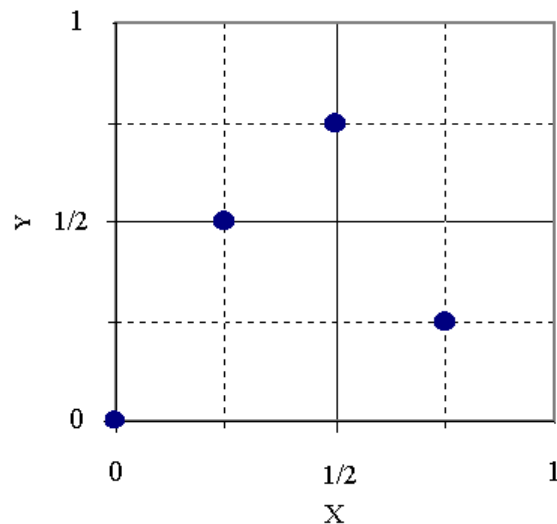


Figure 2: Net created from steps 1-4 without randomization.

As stated above, a (t, m, s) -net created through appropriate generating matrices gives points on the lower-left corners of the subspaces. Random re-ordering of the numbers does not change this attribute. To move the points inside their respective subspaces, a draw is taken from a uniform between 0 and $1/b^m$, where $1/b^m$ is the width of each side of the subspace. In our application, a draw is taken between 0 and $1/64$. A separate draw is added to each element of the sequence. Separate draws are taken for each dimension. These random draws move all the points into their respective

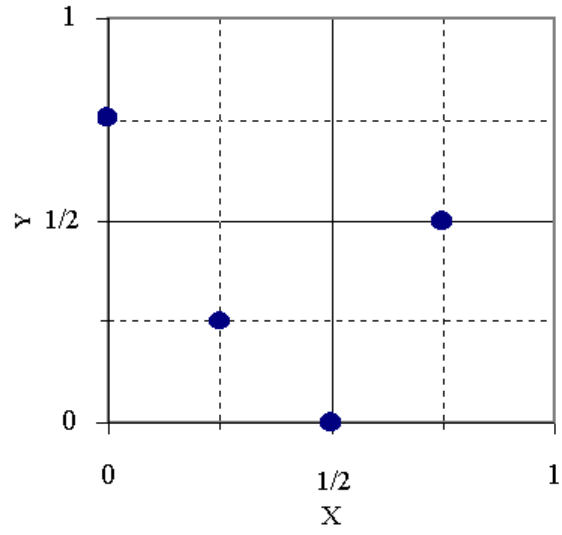


Figure 3: Randomization of first digit.

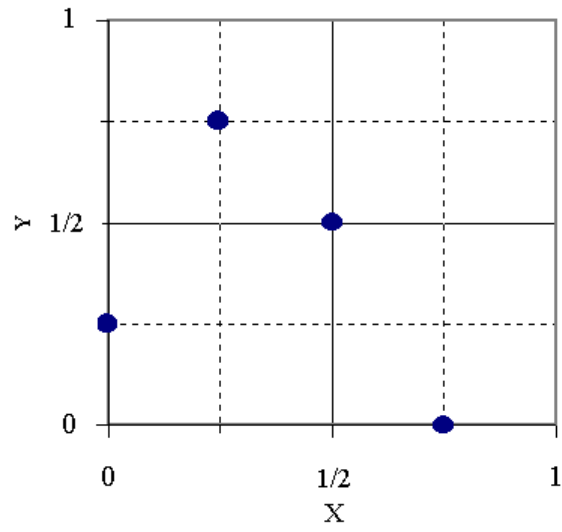


Figure 4: Randomization of the second digit.

subspaces. For example, random shifting applied to the points in Figure 4 provide the points shown earlier in Figure 1. These draws, which are from a uniform distribution, are transformed to the density of interest the same as with Halton sequences.

Owen-randomization could be applied to Halton sequences as shown by Wang and Hickernell (2000). The way it is applied is similar to the randomization of (t, m, s) -nets but here the reordering of the different digits is independent, that is, it does not depend on the previous digits. However, the advantage of Owen-randomization is that it preserves the properties of a (t, m, s) -net. Since Halton sequences are not (t, m, s) -nets and do not have their desirable properties, the motivation for this form of randomization is missing. Also, Wang and Hickernell (2000) show that, for five dimensions, the random start procedure tends to be more efficient than Owen-randomization for Halton sequences. It is not obvious how the simpler randomization methods like the random shift or random start procedures can be adapted to (t, m, s) -nets. If these procedures are applied directly then the (t, m, s) -net property of these sequences will not be guaranteed.

4 Application

We use data described by Train and Hudson (2000) on customers' choice among alternative-fueled vehicles. Each of 538 surveyed customers was given a card that listed three vehicles along with descriptions of the vehicles. The customer was asked to state which of the vehicles he/she would choose to buy. Each vehicle was described in terms of its price, operating cost, range, performance, and engine type. Three engine types were included in the experiments: gas internal combustion (ICV), electric (EV), or gas-electric hybrid (HV). Range was defined as the number of miles that the vehicle could be driven before refueling/recharging. Three levels of performance were distinguished, each of which was described in terms of top speed and number of seconds needed to reach 60 miles per hour. These two performance factors were not varied independently. The three performance levels were coded as 0, 1 and 1.5, since the analysis in Train and Hudson (2000) indicates that customers valued the increment from low to medium performance twice as much as the increment from medium to high. Operating cost was denoted in dollars per month. A mixed logit model (Revelt and Train, 1998; Brownstone and Train, 1999) was specified with a fixed coefficient for price and independent normal coefficients for operating cost, range, performance, an EV dummy, and an HV dummy.

Table 1: Mixed Logit Model of Vehicle Choice
Average estimates and standard errors over ten sets of draws.
10,000 independent random draws per observation.

Explanatory variable		Estimate	Standard error
Price		-0.1278	0.0209
Operating cost	Mean	-0.0689	0.0152
	Std dev	0.0580	0.0342
Range	Mean	1.0730	0.9472
	Std dev	0.1523	10.3274
EV dummy	Mean	-5.8695	1.8540
	Std dev	3.5259	1.4138
HV dummy	Mean	-0.6757	0.4818
	Std dev	2.0843	0.9619
Performance	Mean	0.8701	0.2727
	Std dev	2.311	0.6713

The model was first estimated ten times by MSL with 10,000 independent random draws for each observation in each run. The mean of these estimates are designed the “true” estimates, against which the other methods of constructing draws are compared. The variance of these estimates is extremely small such that interpreting the mean as the simulation-free estimates is warranted. Nevertheless, a more conservative interpretation is that the comparisons reveal how well each method performs relative to taking 10,000 independent draws. Table 1 gives the “true” estimates as well as the mean standard errors. Note that the mean standard errors are not the variance of the estimates over the ten sets of draws. Rather they are the average of the ten standard errors obtained in the ten runs. (More precisely, they are the square root of the average of the squared standard errors.) As such, they reflect sampling variance.

The model was estimated ten times with each of the six types of randomized sequences described above: HL, HS, and N1-N4. Each of these runs used 64 draws per observation. We also estimated the model ten times with 64 independent random draws, which we call R64, and, again with 512 random draws (8 times as many), which we call R512. For each type of draw, the RMSE was calculated for the ten estimates against the “true” estimates. The bias (i.e., the difference between the mean estimates and the “true”) and the standard deviation of the estimates were also calculated.

McFadden (1989) has pointed out that simulation variance is related to sampling variance. In particular, for the method of simulated moments with

fixed weights and an accept-reject simulator, simulation variance is proportional to sampling variance. A similar relation can be expected to occur with other simulation-assisted estimators, though not necessarily as direct proportionality. Intuitively, the reason for the relation between simulation and sampling variance is clear: Large sampling variance means that the log-likelihood function is fairly flat near the maximum; and when the likelihood function is fairly flat near its maximum, errors induced by simulation can move the maximum considerably. Conversely, when the likelihood function is highly peaked at its maximum, which means small sampling variance, simulation error is unlikely to move the maximum as much.

To reflect this relation and facilitate comparisons over parameters, RMSE, bias and standard deviation are expressed as a proportion of the standard errors from Table 1. The results are given in Table 2-4. For example, Table 2 indicates that method N4 provides a RMSE for the first parameter that is 37.7 percent of its standard error.

The methods are reported in these tables in ascending order of performance, based on the average over parameters of the RMSE as a proportion of standard error (the last row of Table 2). One exception to this ordering is R512 which is given last since it uses more draws. The relative performance is similar for each parameter individually as for the average over parameters. Also, the ordering is nearly the same for bias and standard deviation as for RMSE. This result is consistent with McFadden's (1989) observation that simulation bias is proportional to the variance in the simulated probability, such that bias and variance, and hence RMSE, move together.

All of the quasi-random draws greatly outperform the same number of independent random draws (R64). This result is consistent with, and generalizes, the findings of Bhat (2001) and Train (2000), who found that Halton draws greatly outperform independent random draws. As an indication of the potential savings from quasi-random draws, eight times as many independent draws (R512) are required to reach the same level of performance as the best quasi-random method.

The (t, m, s) -nets perform in accordance with expectations. N1 outperforms N2, as expected since N1 has larger m and yet the same t, s , and b^m . N3 outperforms N4, as expected since N3 has lower t and the same m, s , and b .

Consider now the Halton draws. HS, which contains randomization over observations, performs slightly better than HL. The better (t, m, s) -nets (that is, N1 and N3) outperform both of the Halton methods. However, the other two (t, m, s) -nets perform worse than the Halton draws. This result suggests that researchers cannot rely on (t, m, s) -nets to always be more

Table 2: RMSE as share of standard error of estimate

R64	N4	N2	HL	HS	N3	N1	R512
0.45624	0.37703	0.49449	0.17848	0.19757	0.19502	0.17944	0.16082
0.43343	0.25428	0.34965	0.22577	0.17425	0.18536	0.13843	0.13305
0.31277	0.53546	0.34316	0.60126	0.21181	0.20657	0.20708	0.13333
0.34205	0.18757	0.16852	0.12352	0.13154	0.07443	0.14126	0.18009
0.08662	0.07738	0.08208	0.04337	0.04757	0.05956	0.06015	0.06757
0.49436	0.29077	0.29615	0.19987	0.23985	0.19512	0.15641	0.21433
0.92837	0.52265	0.37281	0.21659	0.35563	0.28710	0.23988	0.34307
0.33895	0.17561	0.16964	0.12252	0.19639	0.16860	0.09083	0.10072
0.52460	0.47166	0.25146	0.32478	0.33307	0.21315	0.18886	0.18733
0.41280	0.26837	0.34762	0.18218	0.16642	0.18180	0.11368	0.08774
0.65319	0.33003	0.60770	0.22955	0.18060	0.27523	0.18548	0.10334
Average RMSE as share							
0.45303	0.31735	0.31666	0.22254	0.20315	0.18563	0.15468	0.15558

accurate than Halton draws. Unfortunately, since the coverage for Halton draws and (t, m, s) -nets cannot be directly compared, there is currently no means to determine whether a particular (t, m, s) -net will outperform Halton draws. This issue is a crucial topic for future research.

Appendix A: Generating matrices

N1: (0,3,5)-net in base 4

This net was created by constructing a Niederreiter (0,3,4)-net in base 4 and then adding $n/64$ for the fifth dimension. The generating matrices for a Niederreiter (0,3,4)-net in base 4 are:

$$C_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, C_2 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, C_3 = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, C_4 = \begin{pmatrix} 1 & 3 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Table 3: Bias as share of standard error of estimate

R64	N4	N2	HL	HS	N3	N1	R512
0.40245	0.08611	0.06346	0.09516	0.08398	0.11111	0.06619	0.09838
0.40069	0.08515	0.02790	0.18885	0.08208	0.08873	0.05533	0.10278
0.13339	0.12298	0.12714	0.34857	0.06168	0.08555	0.15657	0.05996
0.26429	0.12579	0.11267	0.08235	0.09025	0.01824	0.09293	0.13959
0.06463	0.06130	0.05666	0.03880	0.04009	0.04839	0.04769	0.05483
0.43984	0.12333	0.05010	0.14954	0.16833	0.05458	0.03259	0.16535
0.60243	0.23262	0.13267	0.14708	0.27031	0.08168	0.08610	0.26170
0.28903	0.09243	0.03255	0.09267	0.13084	0.00559	0.03586	0.05302
0.35815	0.18037	0.00449	0.01456	0.22657	0.04021	0.07968	0.07457
0.32338	0.00661	0.10346	0.11248	0.08822	0.07197	0.02226	0.05005
0.43730	0.06143	0.20908	0.08003	0.00167	0.18745	0.02600	0.08526
Average bias as share							
0.33778	0.10710	0.08365	0.12274	0.11309	0.07214	0.06374	0.10414

Table 4: Standard deviation as share of standard error of estimate

R64	N4	N2	HL	HS	N3	N1	R512
0.22654	0.38692	0.51693	0.15917	0.18851	0.16894	0.17581	0.13410
0.17420	0.25256	0.36739	0.13043	0.16202	0.17154	0.13376	0.08905
0.29821	0.54933	0.33598	0.51642	0.21359	0.19819	0.14286	0.12553
0.22888	0.14667	0.13210	0.09704	0.10088	0.07606	0.11214	0.11994
0.06078	0.04977	0.06260	0.02042	0.02699	0.03659	0.03864	0.04164
0.23787	0.27756	0.30767	0.13978	0.18009	0.19747	0.16126	0.14374
0.74457	0.49335	0.36725	0.16759	0.24360	0.29013	0.23601	0.23384
0.18663	0.15740	0.17550	0.08447	0.15438	0.17762	0.08797	0.09026
0.40405	0.45938	0.26502	0.34200	0.25734	0.22065	0.18049	0.18115
0.27045	0.28280	0.34982	0.15107	0.14874	0.17598	0.11751	0.07595
0.51145	0.34181	0.60147	0.22679	0.19036	0.21243	0.19358	0.06155
Average std dev as share							
0.30397	0.30887	0.31652	0.18502	0.16968	0.17506	0.14364	0.11789

N2: (0,2,5)-net in base 8

This net was constructed as a Niederreiter (0,2,5)-net in base 8 using the following generating matrices:

$$C_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, C_2 = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, C_3 = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix},$$

$$C_4 = \begin{pmatrix} 1 & 3 \\ 0 & 1 \end{pmatrix}, C_5 = \begin{pmatrix} 1 & 4 \\ 0 & 1 \end{pmatrix}.$$

N4: (3,6,5)-net in base 2

This net was created by constructing a Niederreiter (3,6,4)-net in base 2 and then adding $n/64$ for the fifth dimension. The generating matrices for a Niederreiter (3,6,4)-net in base 2 are:

$$C_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, C_2 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

$$C_3 = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}, C_4 = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}.$$

N3: (2,6,5)-net in base 2

This net was created as a Niederreiter-Xing (2,6,5)-net in base 2 using the matrices provided by (Pirsic, 2002), which are:

$$\begin{aligned}
 C_1 &= \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}, C_2 = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}, \\
 C_3 &= \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}, C_4 = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}, \\
 C_5 &= \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}.
 \end{aligned}$$

Appendix B: Look Up Tables for Arithmetic

The addition and multiplication for bases 4 and 8 are presented below. The operations are commutative, i.e., $a+b = b+a$ and $a*b = b*a$, so presenting only the upper triangle of the tables is fully informative.

Table B1: Summation and multiplication rules for base 4

+	0	1	2	3	*	0	1	2	3
0	0	1	2	3	0	0	0	0	0
1		0	3	2	1	0	1	2	3
2			0	1	2	0	2	3	1
3				0	3	0	3	1	2

Table B2: Summation and multiplication rules for base 8

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1		0	6	4	3	7	2	5
2			0	7	5	4	1	3
3				0	1	6	5	2
4					0	2	7	6
5						0	3	1
6							0	4
7								0

*	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1		1	2	3	4	5	6	7
2			3	4	5	6	7	1
3				5	6	7	1	2
4					7	1	2	3
5						2	3	4
6							4	5
7								6

References

- Bhat, C. (2001), ‘Quasi-random maximum simulated likelihood estimation of the mixed multinomial logit model’, *Transportation Research B* **35**, 677–693.
- Bhat, C. (forthcoming), ‘Simulation estimation of mixed discrete choice models using randomized and scrambled halton sequences’, *Transportation Research*.
- Bratley, P., B. Fox and H. Niederreiter (1992), ‘Implementation and tests of low-discrepancy sequences’, *ACM Transactions on Modeling and Computer simulation* **2**, 195–213.
- Brownstone, D. and K. Train (1999), ‘Forecasting new product penetration with flexible substitution patterns’, *Journal of Econometrics* **89**, 109–129.
- Faure, H. (1982), ‘Discrepance de suites associees a un systeme de numeration (en dimension s)’, *Acta. Arithm.* **41**, 337–351.
- Lee, L. (1995), ‘Asymptotic bias in simulated maximum likelihood estimation of discrete choice models’, *Econometric Theory* **11**, 437–483.
- McFadden, D. (1989), ‘A method of simulated moments for estimation of discrete response models without numerical integration’, *Econometrica* **57**, 995–1026.
- Niederreiter, H. (1987), ‘Point sets and sequences with small discrepancy’, *Monatshefte fur Mathematik* **104**, 273–337.

- Niederreiter, H. (1988), ‘Low-discrepancy and low dispersion sequences’, *Journal of Number Theory* **30**, 51–70.
- Niederreiter, H. and C. Xing (1996), ‘Low-discrepancy sequences and global function fields with many rational places’, *Finite Fields and their Applications* **2**, 241–273.
- Owen, A. (1995), Randomly permuted (t, m, s) -nets and (t,s) - sequences, in H.Niederreiter and P.Shiue, eds, ‘Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing’, Springer-Verlag, New York, pp. 299–317.
- Pirsic, G. (2002), ‘A software implementation of niederreiter-xing sequences’, paper at <http://www.dismat.oeaw.ac.at/pirs/niedxing.html>.
- Revelt, D. and K. Train (1998), ‘Mixed logit with repeated choices’, *Review of Economics and Statistics* **80**, 647–657.
- Sándor, Z. and P. András (2001), ‘Alternative sampling methods for estimating multivariate normal probabilities’, Working Paper, Department of Economics, University of Gronigen, The Netherlands.
- Sobol, I. (1967), ‘The distribution of points in a cube and the approximate evaluation of integrals’, *Zh. Vychisl. Mat. i. Mat. Fiz.* pp. 784–802.
- Tang, B. (1993), ‘Orthogonal array-based latin hypercubes’, *Journal of the American Statistical Association* **88**, 1392–1397.
- Train, K. (2000), ‘Halton sequences for mixed logit’, Working Paper No. E00-278, Department of Economics, University of California, Berkeley.
- Train, K. (2002), *Discrete Choice Methods with Simulation*, Cambridge University Press, New York.
- Train, K. and K. Hudson (2000), ‘The impact of information in vehicle choice and the demand for electric vehicles in california’, project report, National Economic Research Associates.
- 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. Hickernell (2000), ‘Randomized halton sequences’, *Mathematical and Computer Modeling* **32**, 887–899.