

## Online Supplement A

This supplement presents the rationale and the description of algorithms used by the quasi Monte Carlo methods compared in our study.

### Halton sequence

The QMC method that is currently the most commonly used for simulating the log-likelihood function of discrete choice models uses the Halton sequence ([Halton, 1960](#)). Following [Kocis and Whiten \(1997\)](#), the  $n$ -th element of the Halton sequence generated with a base  $b_j$ <sup>36</sup> is given by the so called radical inverse function  $\Phi_{b_j}(n)$  defined as follows:

$$\Phi_{b_j}(n) = \sum_{i=0}^{\infty} \alpha_i(j, n) b_j^{-i-1}, \quad (5)$$

where  $\alpha_i(j, n) \in [0, b_j)$  and it is an integer obtained from digit expansion of  $n$  in base  $b_j$ :

$$n = \sum_{i=0}^{\infty} \alpha_i(j, n) b_j^i. \quad (6)$$

The  $K$ -dimensional Halton sequence is given simply by  $K$  one-dimensional Halton sequences generated with different bases (most often  $K$  first prime numbers):

$$x_n = (\Phi_{b_1}(n), \dots, \Phi_{b_K}(n)). \quad (7)$$

The drawback of the Halton sequence is a high correlation between sequences generated using high prime numbers (see Online Supplement B for illustration). This translates into poor performance in evaluating higher dimensional integrals. The way to address this problem is to use so called *scrambling*; in other words, apply a generalized radical inverse function:

$$\Phi_{b_j}(n) = \sum_{i=0}^{\infty} \sigma(\alpha_i(j, n)) b_j^{-i-1}, \quad (8)$$

where  $\sigma(\cdot)$  is an operator of permutations on  $\alpha_i(j, n)$ . Different choices for  $\sigma$  are proposed in the literature (e.g., [Braaten and Weller, 1979](#)). We applied the reverse Radix algorithm ([Kocis and Whiten, 1997](#)).

---

<sup>36</sup> Most often  $b_j$  is some prime number.

The idea of the reverse Radix algorithm is as follows: given the representation of  $\alpha_i(j, n)$  in base 2, the fixed number of its digits are reversed (this means that the Halton sequence in base 2 and scrambled Halton sequence in base 2 are the same). Values that are too large are removed from the sequence.

The last thing to describe is randomization of the sequence. Proposed scrambling is still purely deterministic, so to include some randomness and be able to analyze the variance of the sequence, we applied the so-called *random shift*. When estimating mixed logit,  $N \cdot K$  sequences of length  $R$  have to be generated.<sup>37</sup> Instead, we generate only  $K$  sequences of the length  $N \cdot R$  and divide it into  $N$  parts. Properties of the Halton sequence assure that these sub-sequences still have a good coverage on a unit cube. We apply the following random shifting:

$$x_{jnk} = \{ \varepsilon_{jnk} + u_{nk} \}, \quad (9)$$

where  $\varepsilon_{jnk}$  is an original scrambled Halton draw ( $j \in \{1, \dots, R\}$ ,  $n \in \{1, \dots, N\}$ ,  $k \in \{1, \dots, K\}$ ),  $u_{nk}$  is a standard uniform draw and  $\{ \}$  is a fractional part function. We also tried a different type of random shifting of the following form:

$$x_{jnk} = \{ \varepsilon_{jnk} + u_k \}, \quad (10)$$

which differs, as now uniform draws are the same for different individuals (but different for different attributes). Our initial simulation revealed that the shifting in (9) performed better, so we decided to use this type only.

## Sobol Sequence

The Sobol sequence ([Sobol, 1967](#)) is a *so-called* (t,s)-sequence. To explain the idea behind (t,s)-sequences, we are going to first introduce (t,m,s)-nets. While the Halton sequence aims at obtaining a uniform coverage of  $[0,1]$ , and a multidimensional sequence is created by taking many such sequences generated with different bases, the (t,s)-sequences use only one base number and the multidimensional sequence is obtained by applying different generating matrices to different dimensions. Following [Lemieux \(2009\)](#) and [Bratley and Fox \(1988\)](#), let  $\alpha_i(j, n)$  from equation (6) be transformed in the following way for the  $k$ -th dimension:

---

<sup>37</sup>  $N$  is the number of respondents,  $K$  is the number of random parameters,  $R$  is the desired number of draws.

$$1 \quad (\tilde{\alpha}_0^k(j,n), \tilde{\alpha}_1^k(j,n), \dots)^T = C_k \cdot (\alpha_0(j,n), \alpha_1(j,n), \dots)^T, \quad (11)$$

2 where  $C_k$  is what we call a generation matrix.<sup>38</sup> Then the  $n$ -th element in the  $k$ -th dimension of  
 3 this sequence is given by:

$$4 \quad x_{n-1,k} = \sum_{i=0}^{\infty} \tilde{\alpha}_i^k(j, n-1) b_j^{-i-1}, \quad (12)$$

5 which is almost identical to the inverse radical function in (5). As can be seen, the choice of these  
 6 generation matrices plays a key role. We describe the process of generating them below.

7 Formally defining the (t,m,s)-nets requires one more definition. We are going to say that the point  
 8 set of length  $b_j^m$  is  $(q_1, \dots, q_s)$ -*equidistributed* in base  $b_j$ , if every cell of the form:

$$9 \quad J(\mathbf{r}) = \prod_{k=1}^s \left[ \frac{r_k}{b_j^{q_k}}, \frac{r_k+1}{b_j^{q_k}} \right) \quad (13)$$

10 contains  $b_j^{m-q}$  points of this point set, where  $q = q_1 + \dots + q_s$ , and  $r_k$  are any integers such that  
 11  $0 \leq r_k < b_j^{q_k}$ . Then (t,m,s)-nets in base  $b_j$  can be defined as a sequence of length  $b_j^m$  which is  
 12  $(q_1, \dots, q_s)$ -*equidistributed* whenever  $q \leq m-t$  ([Lemieux, 2009](#)).

13 For an illustration, consider a (0,2,2)-net in base 2, which is a 4-point sequence in two dimensional  
 14 space. The choice of  $(q_1, q_2)$  can be only (0,0), (1,0), (0,1) and (1,1). For the (0,0) case  $J(\mathbf{r})$   
 15 can be only a unit square, so the (0,0)-*equidistribution* condition says that all four points of this  
 16 sequence are in this square (which is true for any sequence). In the (1,0) case,  $J(\mathbf{r})$  can be  
 17  $[0, 1/2) \times [0, 1)$  or  $[1/2, 1) \times [0, 1)$ , so this condition says that in every such horizontal rectangle, two  
 18 points of sequence are placed. The condition of (1,1)-*equidistribution* indicates that in every interval  
 19 of the form  $[i/2, (i+1)/2) \times [j/2, (j+1)/2)$  where  $i, j \in \{0, 1\}$ , one point of the sequence is placed.<sup>39</sup>  
 20 As a result, this sequence has the best coverage one can expect from a 4-point long sequence.

21 Having the definition of (t,m,s)-nets we can simply define a (t,s)-sequence as a sequence for which  
 22 every subsequence  $\mathbf{x}_{l \cdot b_j^h}, \dots, \mathbf{x}_{(l+1) \cdot b_j^h}$  is a (t,h,s)-net. In particular, this means that the first  $b_j^h$  points  
 23 of the (t,s)-sequence are (t,h,s)-net.

---

<sup>38</sup>  $C_k$  elements  $\in \mathbb{Z}_{b_j}$ ; matrix multiplication on the righthand side is also in  $\mathbb{Z}_{b_j}$

<sup>39</sup> These intervals are just squares emerged from partitioning of a unit square in four parts.

As in case of the Halton, scrambling techniques can improve performance of the Sobol sequence. For (t,s)-sequences, it is a more difficult task, however, because we would like the scrambled sequence to possess the properties of the original sequence.

One way of scrambling Sobol sequences is to apply a random linear scramble combined with a random digital shift (Matoušek, 1998). Random digital shift is like the random shift described for the Halton sequence. For a draw from the Sobol sequence  $x_n^k$ , which can be presented in the form of a binary digit expansion  $x_n^k = \sum_{i=0} b_i \cdot 2^{-i}$ , and a draw from a standard uniform distribution  $u^k = \sum_{i=0} u_i^k \cdot 2^{-i}$ , also presented in binary form, the new draw is created by setting:

$$\tilde{x}_n^k = \sum_{i=0} (b_i + u_i^k) \cdot 2^{-i}, \quad (14)$$

where addition is done in  $\mathbb{Z}_2$ .

The random linear scramble is done by using generation matrices of form  $R_k \cdot C_k$  instead of simple  $C_k$ , where  $R_k$  is a lower-triangular non-singular matrix and matrix multiplication is done in  $\mathbb{Z}_2$ . This is called a linear scramble, as the  $n$ -th draw after scrambling is a linear function of  $n$  first draws in original sequence. Both linear scrambling and a random linear digit shift keep  $(q_1, \dots, q_s)$ -equidistribution property of a sequence and, what is more, the scrambling can lower the t-value of a (t,s)-sequence.<sup>40</sup>

The last thing described here is the process of generating the matrices to create sequences with the required properties. Sobol (1967) proposed to create the matrices with  $b_j = 2$ , which we applied in our study. To create the  $k$ -th generation matrix, we need to first define a primitive polynomial in  $\mathbb{Z}_2$  of form:

$$p_k(z) = z^{d_k} + a_{k,1}z^{d_k-1} + \dots + a_{k,d_k} \quad (15)$$

Second, we need  $d_k$  (which is a degree of  $p_k(z)$ ) direction numbers:

$$v_{k,r} = \frac{m_{k,r}}{2^r}, \quad (16)$$

where  $m_{k,r}$  is an odd integer  $\in [1, 2^r - 1]$  and  $v_{k,r}$  are written in binary digit expansion. The generation matrix  $C_k$  is created by setting its columns to these direction numbers presented in

---

<sup>40</sup> Which implies a better coverage.

vector forms.<sup>41</sup> To obtain direction numbers with indices greater than  $d_k$ , the following recursive procedure can be applied:

$$v_{k,r} = a_{k,1}v_{k,r-1} \oplus \dots \oplus a_{k,d_k}v_{k,r-d_k} \oplus (v_{k,r-d_k}/2^{d_k}), \quad (17)$$

where  $\oplus$  is an exclusive or logical function and  $a_{k,i}$  are taken from  $p_k(z)$  polynomials.

Consider an example from [Lemieux \(2009\)](#): in order to generate  $C_3$  we set  $p_3(z) = z^2 + z + 1$  and choose  $v_{3,1}$ ,  $v_{3,2}$  to be 0.5 and 0.75, respectively, which is 0.1 and 0.11 in binary expansion. According to (17) we have:

$$v_{3,3} = (1,1,0)^T \oplus (1,0,0)^T \oplus (0,0,1)^T = (0,1,1)^T. \quad (18)$$

This way we obtained the first three columns of  $C_3$ . To obtain further columns, (17) has to be applied again.

Presentation of a Sobol sequence with generation matrices is relatively intuitive, and shows a connection between the Sobol and Halton methods. Nevertheless, it is easier to implement the following representation of  $(n+1)$ -th element of a Sobol sequence in the  $k$ -th dimension:

$$x_n^k = \alpha_1(1,n) \cdot v_{k,1} \oplus \alpha_2(1,n) \cdot v_{k,2} \oplus \dots \quad (19)$$

Where  $\alpha_i(1,n)$  are defined as in equation (6) with  $b_j = 2$ . [Antonov and Saleev \(1979\)](#) showed that this formula can be rewritten using Gray Code binary representation of  $n$  resulting in:

$$x_n^k = g_1(n) \cdot v_{k,1} \oplus g_2(n) \cdot v_{k,2} \oplus \dots \quad (20)$$

One property of Grey Code representation is that the representation for  $n$  and  $n+1$  differs in only one position. Using this property, the formula in (19) can be written as

$$x_n^k = x_{n-1}^k \oplus v_c, \quad (21)$$

where  $c$  is an index of right-most zero bits in binary representation of  $n$  ([Bratley and Fox, 1988](#)), e.g., in 0.1  $c=2$ , in 0.01  $c=1$ , and in 0.11  $c=3$ .

In our simulation, we used primitive polynomials and direction numbers implemented in Matlab `sobolset` class.

---

<sup>41</sup> I.e., if  $v_{k,r} = 0.11$  then its vector form is  $(1,1,0,\dots)^T$ .

## Modified Latin Hypercube Sampling

Modified Latin hypercube sampling (MLHS) was proposed by [Hess, Train and Polak \(2006\)](#) as a variation of Latin hypercube sampling ([see, e.g., Stein, 1987](#)). Assume that  $P = \{p_{jk}\}$  is a  $R \times K$  matrix of which every column contains an independent, random permutation of sequence  $\{1, 2, \dots, R\}$ . Additionally let  $\Xi = \{\xi_k\}$  be a  $1 \times K$  vector of independent, random uniform draws on  $[0, 1]$  interval. Matrix  $X = \{x_{jk}\}$  of MLHS draws is created by setting:

$$x_{jk} = F^{-1}\left(R^{-1}(p_{jk} + \xi_k - 1)\right), \quad (22)$$

where  $F(\cdot)$  is a cdf of the distribution one wants to draw from.

MLHS is not a low-discrepancy sequence designed as the Halton or Sobol sequence, because generation of a longer sequence requires creating a new one. Nevertheless, it has good coverage properties and, because of the random element  $\xi_k$  and permutations, its variance can be readily analyzed the same way as in the pseudo-random case. In our setting,  $K$  is equal to number of random parameters multiplied by the number of respondents, and  $R$  is a desired number of draws.