Exercise – Understanding the Halton sequence

1. Aim of the exercise

This exercise introduces Halton sequences as a powerful alternative to pseudo-random draws in simulation-based estimation. Halton sequences are low-discrepancy sequences that offer superior coverage of the unit interval, making them especially valuable in quasi-Monte Carlo integration and econometric modeling. While Train (2003) provides a foundational treatment of Halton sequences, this exercise goes further, offering a hands-on, visual, and customizable framework for exploring their behavior. A key contribution of this file is the inclusion of a custom MATLAB function, `halton`, which extends the capabilities of MATLAB's built-in routines. Unlike the default `haltonset` of MATLAB, our function allows users to specify prime bases, apply leap and burn-in adjustments, and incorporate randomization or scrambling techniques based on methods from Train and Bhat. It also transforms the draws into standard normal variates, making it simulation-ready for mixed logit and other models. Through clear visualizations and modular code, this exercise empowers users to experiment with different Halton variants and understand their impact on dispersion, correlation, and simulation accuracy. This file offers both theoretical insight and practical tools for improving your simulation workflow.

2. Theory

This section presents the theoretical foundation of Halton sequences, drawing heavily on the exposition provided by Train (2003). The structure and many of the sentences closely follow Train's treatment, as his work offers a clear and rigorous framework for understanding quasi-random sequences in simulation-based estimation. By adhering to Train's approach, this section aims to preserve the clarity and depth of the original while integrating visual and computational enhancements tailored to this exercise.

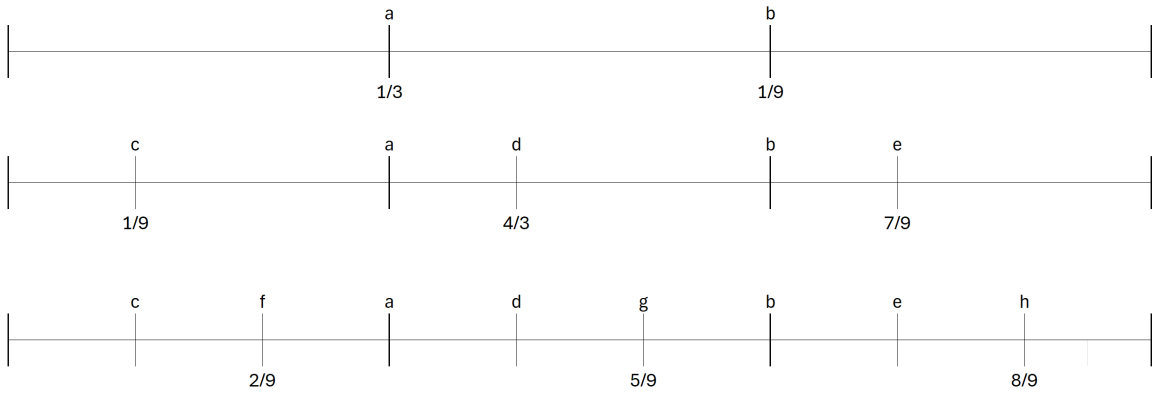
Halton sequences are designed to provide uniform coverage of the unit interval and reduce clustering across observations (Halton, 1960). Each sequence is constructed using a prime number as its base, which ensures low discrepancy and helps avoid unwanted patterns when generating values across multiple dimensions, such as input variables in simulations or numerical integration tasks.

The concept behind the Halton sequence is best illustrated through an example. To understand how it works in base 3, we start with regular counting numbers like 1, 2, 3, 4, and so on. Each number is rewritten using base 3, which means expressing it with only the digits 0, 1, and 2. For example, 1 becomes "1", 2 becomes "2", 3 becomes "10" because $3 = 1 \times 3^1 + 0 \times 3^0$, and 4 becomes "11" because $4 = 1 \times 3^1 + 1 \times 3^0$. Once a number is written in base 3, we reverse its digits. So "10" becomes "01" or "12" becomes "21". These reversed digit strings are then turned into fractions.

Each digit in the reversed base-3 number is then used to build a fraction. The first digit, on the left, is placed in the first decimal position and is divided by 3. The second digit goes in the next position and is divided by 9. The third digit, if present, would be divided by 27, and so on. This means that the farther a digit is from the decimal point, the smaller its contribution to the final value. For example, "02" becomes $\frac{0}{3} + \frac{2}{9} = \frac{2}{9}$, and "21" becomes $\frac{2}{3} + \frac{1}{9} = \frac{7}{9}$.

This process maps each counting number to a point between 0 and 1, and the resulting sequence spreads these points evenly across the interval. For example, in base 3, the first two non-zero values generated are $\frac{1}{3}$ and $\frac{2}{3}$, which divide the unit interval into three equal segments,

as illustrated in the top panel of the figure below. The Halton sequence then continues by placing new points inside each of these segments. The middle panel shows the next layer of refinement, where one point is added to each segment: $\frac{1}{9}$ in the first third, $\frac{4}{9}$ in the second, and $\frac{7}{9}$ in the third. The bottom panel continues this refinement by adding one more point to each segment: $\frac{2}{9}$, $\frac{5}{9}$, and $\frac{8}{9}$. Together, these six values, along with the original breakpoints, form the sequence: $\frac{1}{3}, \frac{2}{3}, \frac{1}{9}, \frac{4}{9}, \frac{7}{9}, \frac{2}{9}, \frac{5}{9}, \frac{8}{9}$.



Note that the lower breakpoints in each segment ($\frac{1}{9}, \frac{4}{9}, \frac{7}{9}$) appear in the sequence before the higher breakpoints ($\frac{2}{9}, \frac{5}{9}, \frac{8}{9}$). Next, each of the nine segments is divided into thirds, and the new breakpoints are added. The sequence continues as $\frac{1}{3}, \frac{2}{3}, \frac{1}{9}, \frac{4}{9}, \frac{7}{9}, \frac{2}{9}, \frac{5}{9}, \frac{8}{9}, \frac{1}{27}, \frac{10}{27}, \frac{19}{27}, \frac{4}{27}, \frac{13}{27}, \ldots$. This process can be repeated to generate as many points as needed.

From a programming perspective, it is straightforward to construct a Halton sequence. The sequence is built iteratively. At each iteration t, we denote the current sequence as s_t, which is a set of numbers. The sequence is extended by adding two new values to each existing element:

$$s_{t+1} = \{s_t, s_t + \tfrac{1}{3^t}, s_t + \tfrac{2}{3^t}\}.$$

We start with the initial sequence $s_0 = \{0\}$. Although zero is not technically part of the Halton sequence, including it as the first element simplifies the construction process, as we will see. It can be removed after the full sequence is generated.

In the first iteration, we add $\frac{1}{3^1}$ and $\frac{2}{3^1}$ to the initial element and append the results, yielding

$$\{0, \tfrac{1}{3}, \tfrac{2}{3}\}.$$

The sequence now contains three elements. In the second iteration, we add $\frac{1}{3^2}$ and $\frac{2}{3^2}$ to each element of the current sequence and append the results. The updated sequence consists of nine elements:

$$\begin{array}{lll} 0 = 0, & 0 + \frac{1}{9} = \frac{1}{9}, & 0 + \frac{2}{9} = \frac{2}{9}, \\ \frac{1}{3} = \frac{1}{3}, & \frac{1}{3} + \frac{1}{9} = \frac{4}{9}, & \frac{1}{3} + \frac{2}{9} = \frac{5}{9}, \\ \frac{2}{3} = \frac{2}{3}, & \frac{2}{3} + \frac{1}{9} = \frac{7}{9}, & \frac{2}{3} + \frac{2}{9} = \frac{8}{9}. \end{array}$$

In the third iteration, we add $\frac{1}{3^3}$ and $\frac{2}{3^3}$ to each element of the current sequence and append

the results:

$$
\begin{array}{lll}
0 = 0, & 0 + \frac{1}{27} = \frac{1}{27}, & 0 + \frac{2}{27} = \frac{2}{27}, \\
\frac{1}{3} = \frac{1}{3}, & \frac{1}{3} + \frac{1}{27} = \frac{10}{27}, & \frac{1}{3} + \frac{2}{27} = \frac{11}{27}, \\
\frac{2}{3} = \frac{2}{3}, & \frac{2}{3} + \frac{1}{27} = \frac{19}{27}, & \frac{2}{3} + \frac{2}{27} = \frac{20}{27}, \\
\frac{1}{9} = \frac{1}{9}, & \frac{1}{9} + \frac{1}{27} = \frac{4}{27}, & \frac{1}{9} + \frac{2}{27} = \frac{5}{27}, \\
\frac{4}{9} = \frac{4}{9}, & \frac{4}{9} + \frac{1}{27} = \frac{13}{27}, & \frac{4}{9} + \frac{2}{27} = \frac{14}{27}, \\
\frac{7}{9} = \frac{7}{9}, & \frac{7}{9} + \frac{1}{27} = \frac{22}{27}, & \frac{7}{9} + \frac{2}{27} = \frac{23}{27}, \\
\frac{2}{9} = \frac{2}{9}, & \frac{2}{9} + \frac{1}{27} = \frac{7}{27}, & \frac{2}{9} + \frac{2}{27} = \frac{8}{27}, \\
\frac{5}{9} = \frac{5}{9}, & \frac{5}{9} + \frac{1}{27} = \frac{16}{27}, & \frac{5}{9} + \frac{2}{27} = \frac{17}{27}, \\
\frac{8}{9} = \frac{8}{9}, & \frac{8}{9} + \frac{1}{27} = \frac{25}{27}, & \frac{8}{9} + \frac{2}{27} = \frac{26}{27}.
\end{array}
$$

The sequence now contains 27 elements. In the fourth iteration, we add $\frac{1}{3^4}$ and $\frac{2}{3^4}$ to each element of this sequence and append the results, and so on. Note that the sequence cycles over the unit interval every three numbers. Within each cycle, the values are ordered in ascending fashion.

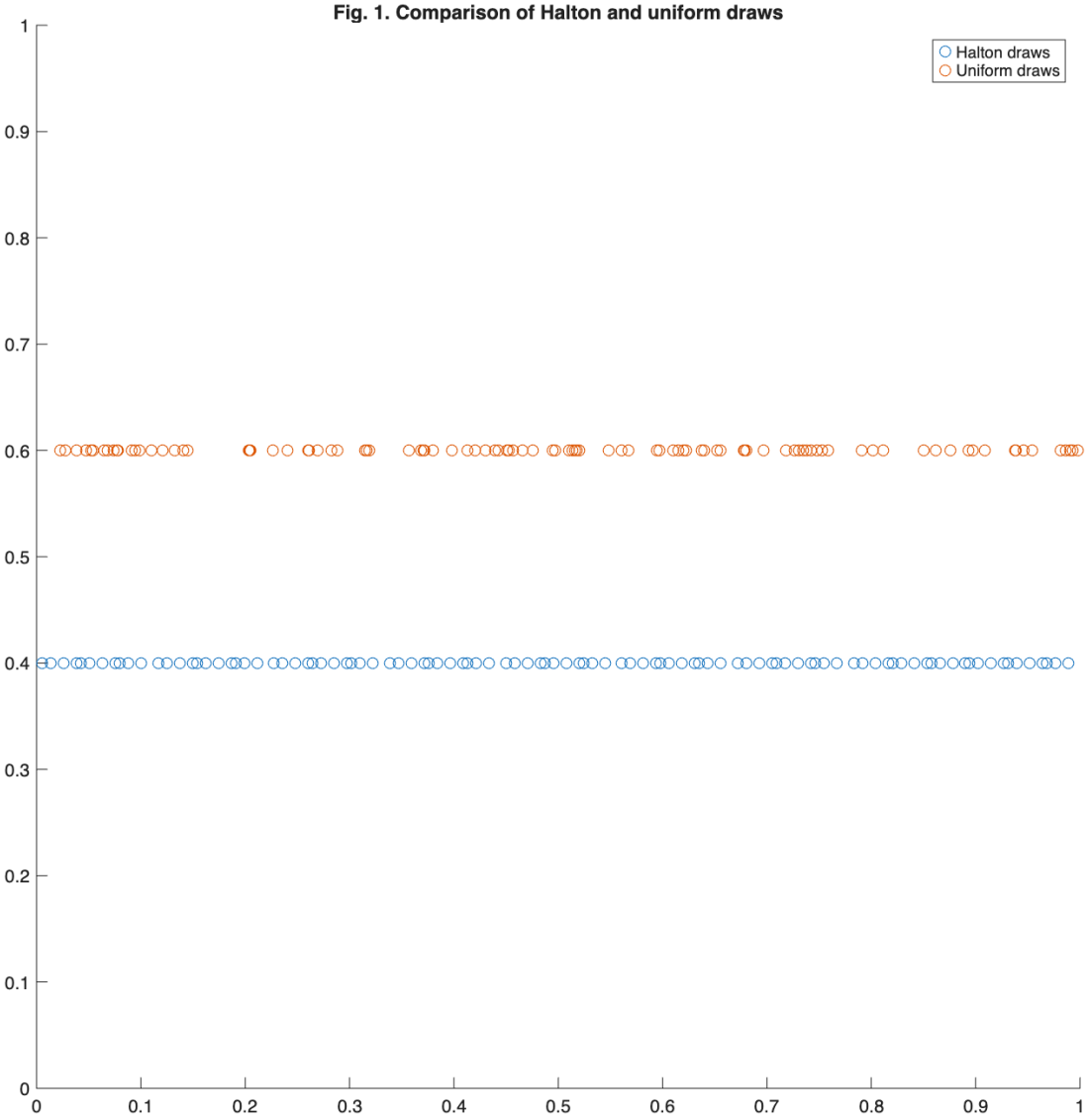
Halton sequences for other prime numbers are created similarly. The sequence for prime 2 is $\{\frac{1}{2}, \frac{1}{4}, \frac{3}{4}, \frac{1}{8}, \frac{5}{8}, \frac{3}{8}, \frac{7}{8}, \frac{1}{16}, \frac{9}{16}, \dots\}$. In general, the sequence for prime k is created iteratively, with the sequence at iteration $t + 1$ being

$$s_{t+1} = \{s_t, s_t + 1/k^t, s_t + 2/k^t, \dots, s_t + (k-1)/k^t\}.$$

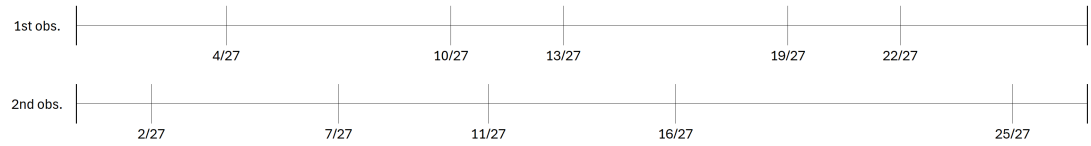
The sequence contains cycles of length k, where each cycle consists of k ascending points on the unit interval equidistant from each other.

Since a Halton sequence is defined on the unit interval, its elements can be interpreted as well-placed draws from a standard uniform distribution. Halton draws provide better coverage than purely random draws, on average, because they are constructed to progressively fill the interval more evenly and densely. The elements in each cycle are equidistant, and each cycle targets regions of the interval that were left uncovered by previous cycles. This structure is illustrated in Figure 1. The MATLAB code used to generate this figure is provided and explained at the end of the file.

Fig. 1. Comparison of Halton and uniform draws



When generating Halton draws for a sample of observations, it is common to construct a single, extended Halton sequence. The initial elements of this sequence are typically discarded, as the early terms of multiple Halton sequences tend to exhibit high correlation. Removing these initial values helps eliminate such correlation and improves the quality of the quasi-random draws. This issue will be explored in more detail later. The remaining elements are then partitioned into groups, with each group serving as the set of draws for one observation. For example, suppose there are two observations and the researcher requires five draws per observation. Using the prime number 3, the researcher generates a total of 20 elements from the Halton sequence: $\left\{0, \frac{1}{3}, \frac{2}{3}, \frac{1}{9}, \frac{4}{9}, \frac{7}{9}, \frac{2}{9}, \frac{5}{9}, \frac{8}{9}, \frac{1}{27}, \frac{10}{27}, \frac{19}{27}, \frac{4}{27}, \frac{13}{27}, \frac{22}{27}, \frac{7}{27}, \frac{16}{27}, \frac{25}{27}, \frac{2}{27}, \frac{11}{27}\right\}$. From these, the first 10 elements are discarded. The remaining 10 elements are then partitioned into two groups of five, with each group serving as the set of draws for one observation. Specifically, the Halton draws for the first observation are $\left\{\frac{10}{27}, \frac{19}{27}, \frac{4}{27}, \frac{13}{27}, \frac{22}{27}\right\}$, and for the second observation $\left\{\frac{7}{27}, \frac{16}{27}, \frac{25}{27}, \frac{2}{27}, \frac{11}{27}\right\}$. These draws are illustrated as follows:



Notice that the gaps in coverage from the first observation are effectively filled by the draws from the second observation. For example, the sizable gap between $\frac{4}{27}$ and $\frac{10}{27}$ in the first observation is bridged by the midpoint, $\frac{7}{27}$, which appears in the second observation. Similarly, the gap between $\frac{13}{27}$ and $\frac{19}{27}$ is filled by $\frac{16}{27}$, again from the second observation. This pattern reflects a key property of Halton sequences: each subsequent subsequence tends to fill in the gaps left by the previous ones, resulting in a more uniform and evenly distributed set of draws.

Thanks to the filling-in property of Halton sequences, simulated probabilities based on these draws tend to exhibit a self-correcting behavior across observations. Specifically, the draws for one observation are often negatively correlated with those of the preceding observation. In our example, the average of the draws for the first observation lies above 0.5, while the average for the second observation falls below 0.5. This negative correlation helps smooth out fluctuations and reduces error in the simulated log-likelihood function, enhancing the stability and accuracy of the estimation.

As the number of draws per observation increases, the coverage within each observation becomes more complete. This enhanced coverage reduces the negative covariance across observations, since fewer gaps remain for subsequent observations to fill. The self-correcting nature of Halton draws is most pronounced when only a small number of draws are used, precisely when such correction is most needed. Nonetheless, overall accuracy improves with a larger number of Halton draws, as each observation benefits from more uniform and thorough coverage of the probability space.

As previously discussed, Halton draws are initially generated from a uniform distribution over the interval [0, 1]. To obtain draws from other univariate densities, each element of the Halton sequence is transformed using the inverse cumulative distribution function (CDF) of the desired target distribution. For instance, to generate draws from a standard normal distribution, one first constructs a Halton sequence, say, using the prime number 3, and then applies the inverse standard normal CDF to each value. For example, if the draw is $\frac{1}{3}$, then $x = \Phi^{-1}\left(\frac{1}{3}\right) \approx -0.43$. This transformation yields a quasi-random draw from the standard normal distribution.

Halton sequences in multiple dimensions are obtained by creating a Halton sequence for each dimension with a different prime for each dimension. For example, a sequence in two dimensions is obtained by creating pairs from the Halton sequence for primes 2 and 3. The points are

$$\varepsilon_1 = \left\langle \frac{1}{2}, \frac{1}{3} \right\rangle, \varepsilon_2 = \left\langle \frac{1}{4}, \frac{2}{3} \right\rangle, \varepsilon_3 = \left\langle \frac{3}{4}, \frac{1}{9} \right\rangle, \varepsilon_4 = \left\langle \frac{1}{8}, \frac{4}{9} \right\rangle, \varepsilon_5 = \left\langle \frac{5}{8}, \frac{7}{9} \right\rangle, \varepsilon_6 = \left\langle \frac{3}{8}, \frac{2}{9} \right\rangle, \ldots$$

To obtain draws for a two-dimensional independent standard normal, the inverse cumulative normal is taken of each element of these pairs.

When creating sequences in several dimensions, it is customary to eliminate the initial part of the series. The initial terms of two Halton sequences are highly correlated, through at least the first cycle of each sequence. For example, the sequences for 7 and 11 begin with $\{\frac{1}{7}, \frac{2}{7}, \frac{3}{7}, \frac{4}{7}, \frac{5}{7}, \frac{6}{7}\}$ and $\{\frac{1}{11}, \frac{2}{11}, \frac{3}{11}, \frac{4}{11}, \frac{5}{11}, \frac{6}{11}\}$. These first elements fall on a line in two dimensions. The correlation dissipates after each sequence has cycled through the unit interval, since sequences with different primes cycle at different rates. Discarding the initial part of the sequence eliminates the correlation. The number of initial elements to discard needs to be at least as large as the largest prime that is used in creating the sequences.

The potential for correlation is the reason that prime numbers are used to create the Halton sequences instead of nonprimes. If a nonprime is used, then there is a possibility that the cycles will coincide throughout the entire sequence, rather than for just the initial elements. For example, if Halton sequences are created for 3 and 6, the sequence for 3 cycles twice for every one cycle of the sequence for 6. Since the elements within a cycle are ascending, the elements in each cycle of the sequence for 3 are correlated with the elements in the cycle of the sequence for 6. Using only prime numbers avoids this overlapping of cycles.

Figure 2 presents sequences based on primes 43 assigned to the x-axis, and 47 assigned to the y-axis, with the first 250 draws omitted. Despite this truncation, clear correlation remains visible in the resulting points. While discarding early terms is intended to reduce alignment between dimensions, especially during the initial cycles, it may not be sufficient when using large, closely spaced primes. These sequences take longer to decorrelate due to their extended cycles, and their deterministic nature means residual structure can persist. As a result, the expected improvement in dispersion may be limited without additional randomization techniques.

The superior coverage and the negative correlation over observations that are obtained with Halton draws combine to make Halton draws far more effective than random draws for simulation. Spanier and Maize (1991) have shown that a small number of Halton draws provide relatively good integration. In the context of discrete choice models, Bhat (2001) found that 100 Halton draws provided more precise results for his mixed logit than 1000 random draws. In fact, the simulation error with 125 Halton draws was half as large as with 1000 random draws and somewhat smaller than with 2000 random draws. Train (2000), Munizaga and Alvarez-Daziano (2001), and Hensher (2001) confirm these results on other datasets.

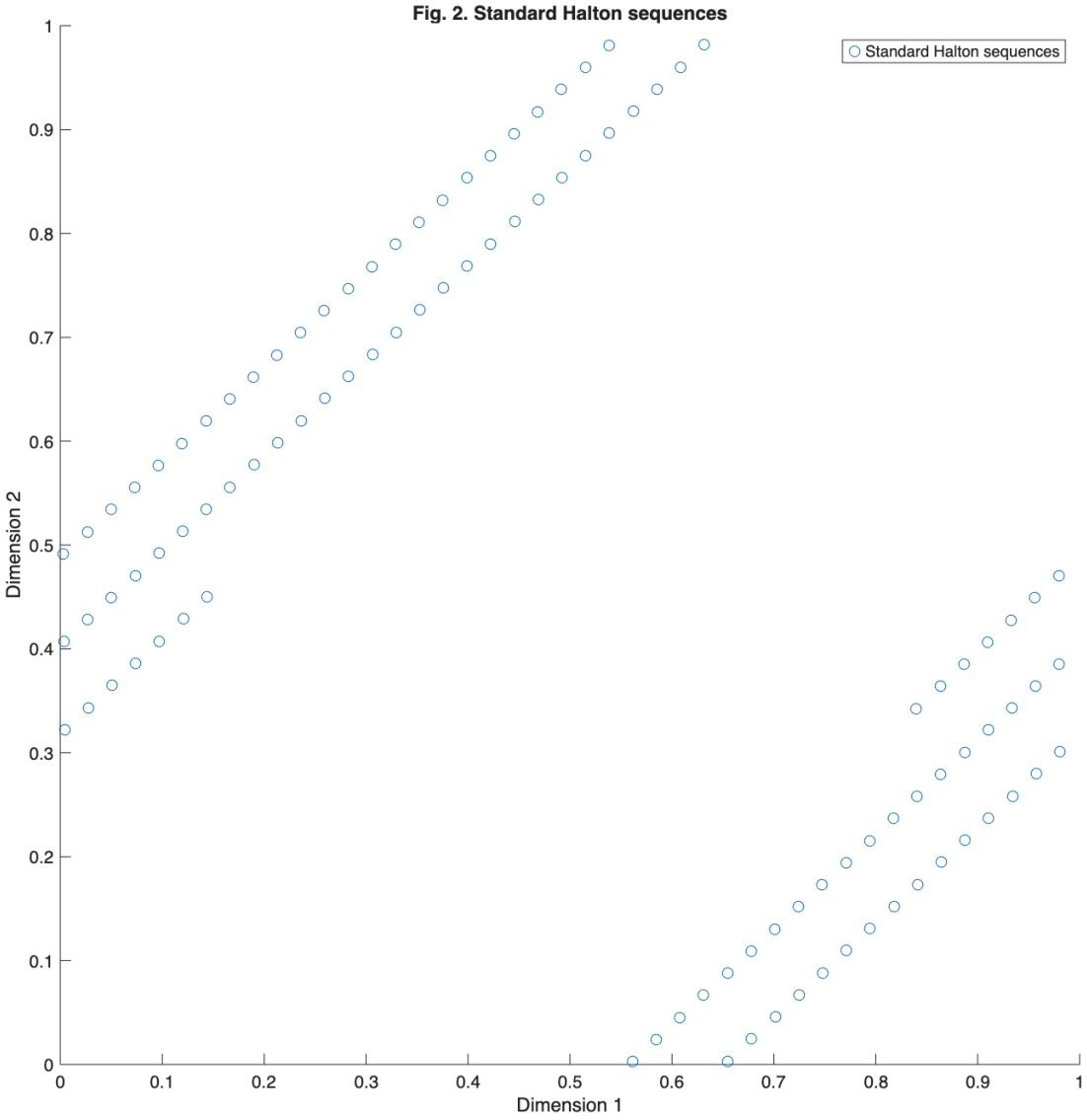


Fig. 2. Standard Halton sequences

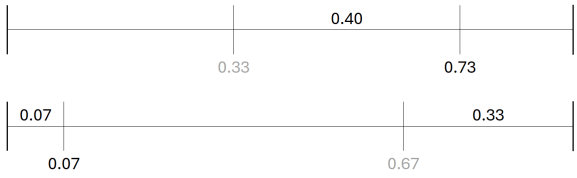
2.1. Randomized Halton sequences

Halton draws are not truly random; they are systematically generated. Yet, the asymptotic properties of simulation-based estimators are typically derived under the assumption of randomness in the draws. This apparent contradiction can be addressed in two ways. First, it is important to recognize that random number generators used in practice do not produce genuinely random values either, they are algorithmic and therefore systematic. Although these generators produce sequences that mimic the properties of true randomness, they are fundamentally pseudorandom. In this light, Halton draws represent a structured approach to numerical integration that can be more precise than pseudorandom methods, which are themselves systematic. Neither approach aligns perfectly with the theoretical notion of randomness, and indeed, it is debatable whether such a concept has a tangible counterpart in the real world. Nonetheless, both methods serve the essential purpose of approximating integrals over probability densities.

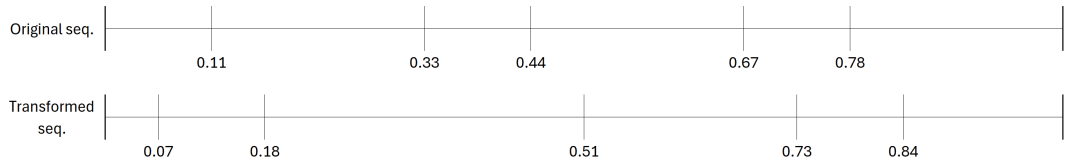
Second, Halton sequences can be transformed in a way that makes them random, at least in the same way that pseudorandom numbers are random. Bhat (2003) describes the process, based on procedures introduced by Tuffin (1996). First, take a draw from a standard uniform density. Label this random draw as μ. Second, add μ to each element of the Halton sequence. If the resulting element exceeds 1, subtract 1 from it. Otherwise, keep the resulting element as is (without subtracting 1).

The transformation is given by the formula $s_n = \text{mod}(s_o + \mu)$, where s_o is the original element of the Halton sequence, s_n is the transformed element, and mod denotes the operation that extracts the fractional part of the expression in parentheses.

The transformation is depicted in the figure below. Suppose the draw of μ from the uniform density is 0.40. The number 0.33 is the first element of the Halton sequence for prime 3. This element is transformed, as shown in the top panel, to $0.33+0.40 = 0.73$, which is just a 0.40 move up the line. The number 0.67 is the second element of the sequence. It is transformed by adding 0.4 and then, since the result exceeds 1, by subtracting 1 to get $0.07(0.67+0.40-1 = 0.07)$. As shown in the bottom panel, this transformation is visualized as moving the original point up by a distance 0.40, but wrapping around when the end of the unit interval is reached. The point moves up 0.33 to where the line ends, and then wraps to the start of the line and continues to move up another 0.07, for a total movement of 0.40.



The figure below shows the transformation for the first five elements of the sequence. The relation between the points and the degree of coverage are the same before and after the transformation. However, since the transformation is based on the random draw of μ, the numerical values of the transformed sequence are random. The resulting sequence is called a randomized Halton sequence. It has the same properties of coverage and negative correlation over observations as the original Halton draws, since the relative placement of the elements is the same; however, it is now random.



In multiple dimensions, each Halton sequence is transformed independently using its own draw from the standard uniform distribution. The figure below illustrates this process for a two-dimensional sequence of length 3, constructed using the prime bases 2 and 3. The sequence corresponding to prime 3 (x-axis) receives a random shift of 0.40, while the sequence for prime 2 (y-axis) receives a shift of 0.35. Each point in the original sequence is translated rightward by 0.40 and upward by 0.35, with wrapping applied when values exceed 1. This preserves the relative structure within each dimension while introducing randomness into the sequence.

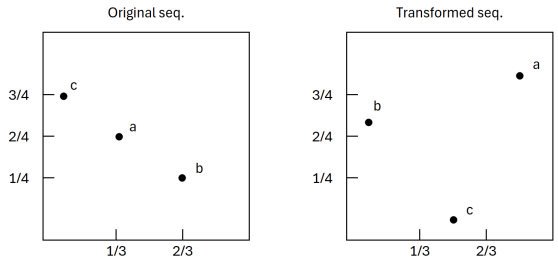


Figure 3 employs the same prime bases as Figure 2 but applies randomization to the Halton sequences.

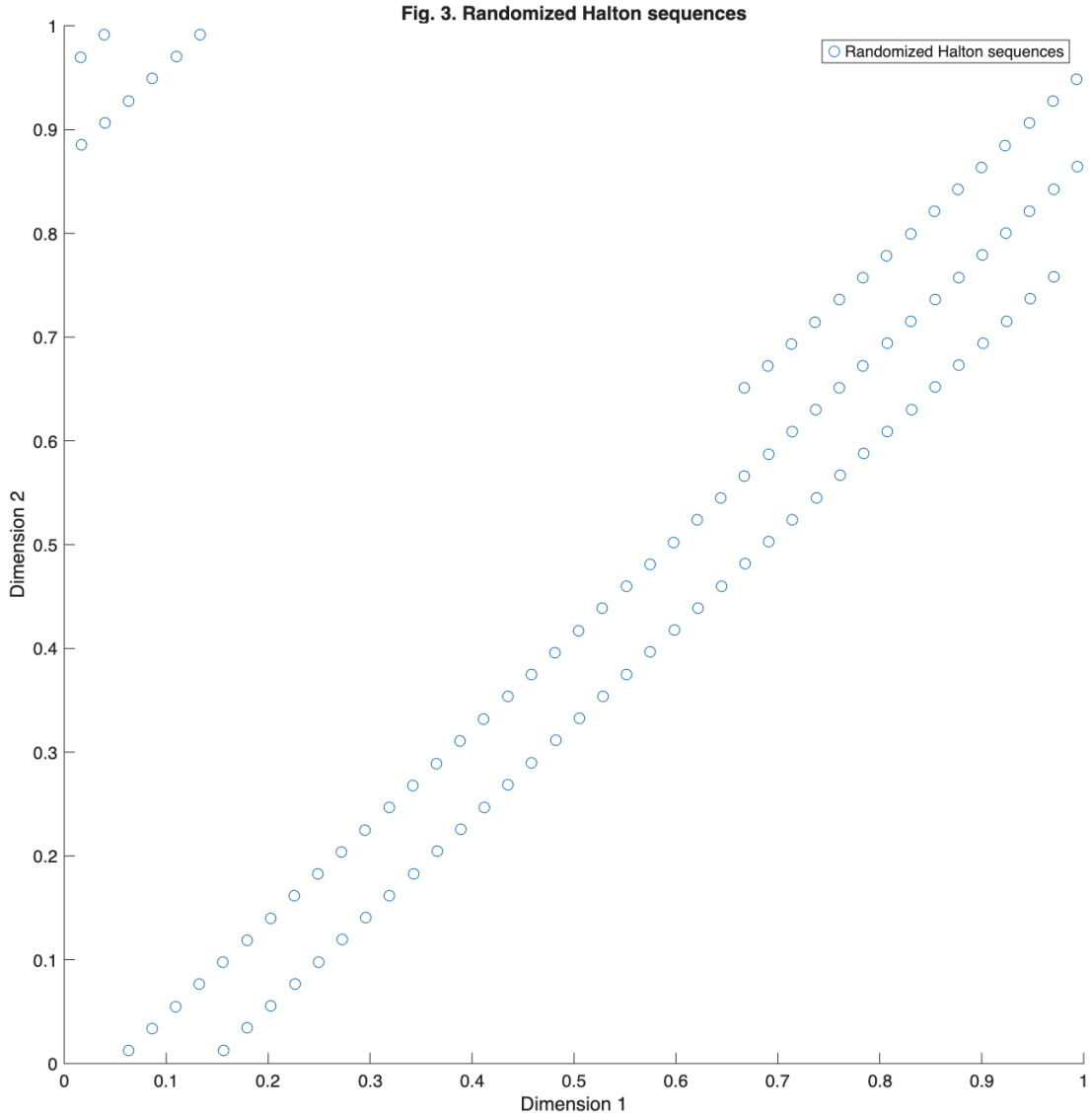


Fig. 3. Randomized Halton sequences

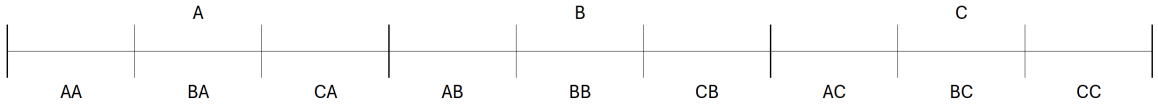
2.2. Scrambled Halton sequences

Another issue with Halton draws arises when they are used in high dimensions. For simulation of high-dimensional integrals, Halton sequences based on large primes are necessary. For example, with 15 dimensions, the primes up to 47 are needed. However, Halton draws defined by large primes can be highly correlated with each other over large portions of the sequence. The correlation is not confined to the initial elements as described earlier, and so cannot be eliminated by discarding these elements. Two sequences defined by large and similar primes periodically become synchronized with each other and stay that way for many cycles. Figure 2 shows the Halton sequence for primes 43 and 47 and demonstrates their high correlation.

Bhat (2003) describes the problem and an effective solution. This correlation can be removed while retaining the desirable coverage of Halton sequences by scrambling the digits of each element of the sequences. The scrambling can be done in various ways. Braatan and Weller (1979) describe a procedure that is most easily explained through an example. Consider the Halton sequence for prime 3:

$$\frac{1}{3}, \frac{2}{3}, \frac{1}{9}, \frac{4}{9}, \frac{7}{9}, \frac{2}{9}, \frac{5}{9}, \frac{8}{9}, \ldots$$

Recall that the sequence is created by dividing the unit interval into three segments, which we label A, B, and C in the following figure:



Each segment of the unit interval is divided into three subsegments. These are labeled using two-letter codes: for example, AA refers to subsegment A of segment A, BA refers to subsegment A of segment B, and so on. The Halton sequence is constructed by selecting the starting points of these segments and subsegments in a specific order. First, we take the starting points of segments B and C, ignoring segment A whose starting point is 0. These are $\frac{1}{3}$ for segment B and $\frac{2}{3}$ for segment C. Next, we consider the subsegments, again ignoring those that begin with A, namely AA, AB, and AC, either because their starting point is 0 or because they duplicate points already included. The remaining subsegments are ordered alphabetically: BA, BB, BC, CA, CB, and CC, with corresponding starting points $\frac{1}{9}, \frac{4}{9}, \frac{7}{9}, \frac{2}{9}, \frac{5}{9}, \frac{8}{9}$.

Segment and subsegment labels beginning with A are excluded because their starting points either coincide with 0 (as in segment A and subsegment AA), or duplicate values already present in the sequence (as in subsegment AB, which shares its starting point with segment B).

To create a scrambled version of the Halton sequence, we simply reverse the alphabetical order of segments B and C, treating C as if it comes before B. With this new ordering, the segments are listed as A, C, B, and the subsegments as AA, AC, AB, CA, CC, CB, BA, BC, BB. The sequence is constructed in the same way as before, but using this new order. As before, we ignore segment A, and take the starting points of segments C and B: $\frac{2}{3}$ for C, $\frac{1}{3}$ for B. We also ignore subsegments AA, AC, and AB, since their starting points are either 0 or already included. The remaining subsegments are listed in the new order: CA, CC, CB, BA, BC, BB, with corresponding starting points: $\frac{2}{9}, \frac{8}{9}, \frac{5}{9}, \frac{1}{9}, \frac{7}{9}, \frac{4}{9}$. So the original sequence is

$$\left\{\frac{1}{3}, \frac{2}{3}, \frac{1}{9}, \frac{4}{9}, \frac{7}{9}, \frac{2}{9}, \frac{5}{9}, \frac{8}{9}\right\},$$

and the scrambled sequence is

$$\left\{\frac{2}{3}, \frac{1}{3}, \frac{2}{9}, \frac{8}{9}, \frac{5}{9}, \frac{1}{9}, \frac{7}{9}, \frac{4}{9}, \frac{8}{9}\right\}.$$

This idea generalizes to higher dimensions, where different digit permutations are applied to Halton sequences constructed from different prime bases. These permutations scramble the sequence by reordering the digits used in its construction, helping to reduce correlation between dimensions.

Figure 4 shows scrambled Halton sequences generated using primes 43 and 47. Unlike the original sequence in Figure 2, the scrambled points are less correlated across dimensions. Bhat demonstrates that scrambled sequences maintain strong performance for high-dimensional integrals, just as unscrambled sequences do for low-dimensional ones.

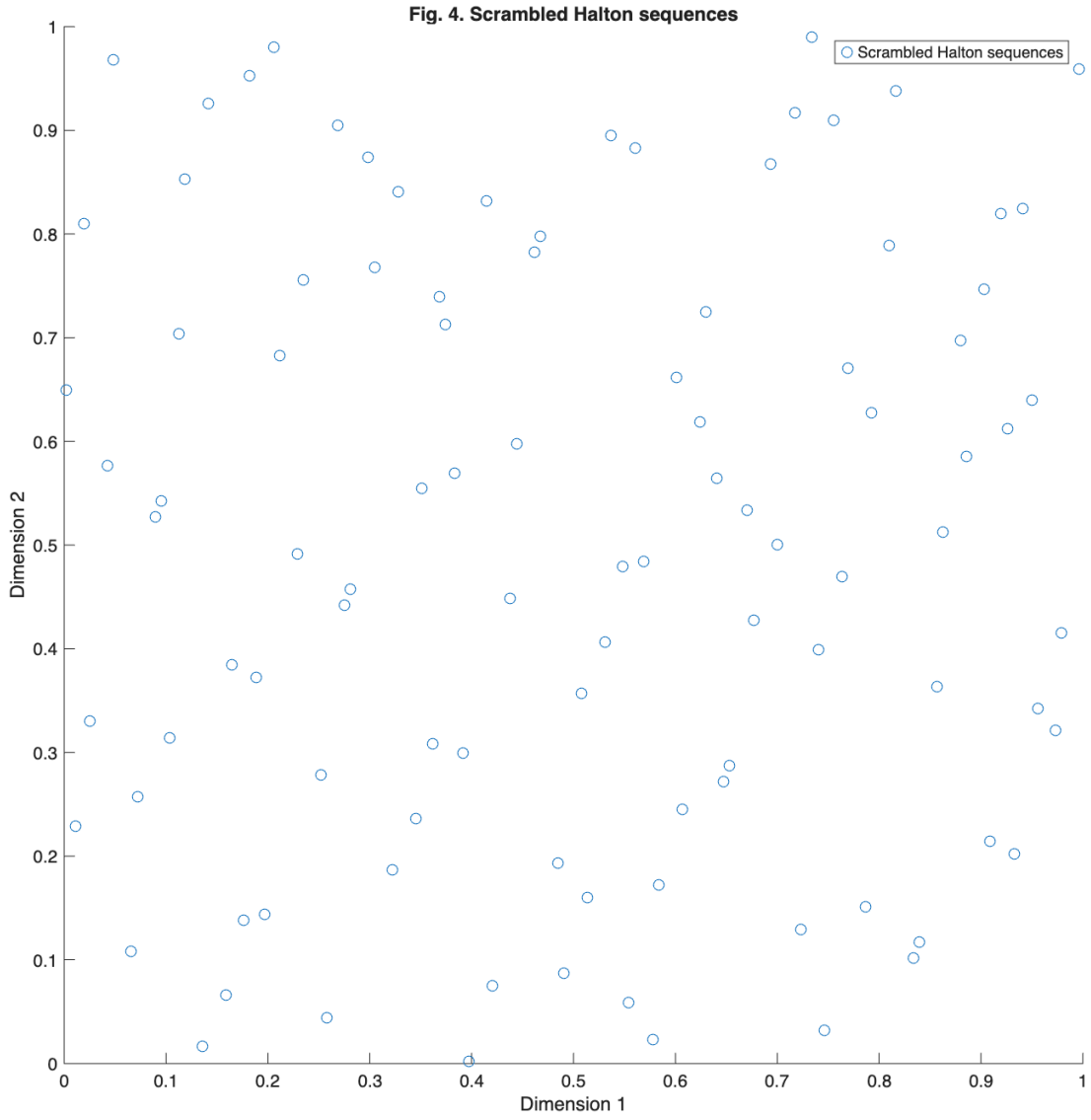


Fig. 4. Scrambled Halton sequences

2.3. Leap in Halton sequences

While Train (2003) provides a comprehensive overview of Halton sequences and their advantages over random draws, his treatment does not explicitly address the concept of "leap" in sequence generation. In contrast, this exercise incorporates the leap parameter to space out the Halton draws, reducing clustering and improving dispersion in high-dimensional settings. The leap modifies the index of the sequence by skipping ahead at fixed intervals, thereby avoiding early-term correlations and enhancing uniformity. This technique is particularly useful when large primes are used or when the number of draws is small, as it mitigates the deterministic structure that can arise in standard Halton sequences. By integrating leap-based sampling, the exercise extends Train's framework and offers a practical enhancement for simulation-based estimation.

Figure 2 showed a standard Halton sequence in two dimensions using large primes. The points exhibit clear correlation, forming diagonal bands, and fail to cover the unit square uniformly. Large areas remain empty, especially near the edges, highlighting the limitations of unmodified Halton draws in higher dimensions. Figure 5, on the other hand, illustrates the effect of applying a leap to the sequence. By skipping indices at regular intervals, the leap disrupts these deterministic patterns and reduces correlation between successive draws. The result is a markedly more uniform and evenly spread sequence across the space. This enhanced dispersion improves the quality of quasi-random sampling and is particularly beneficial for simulation-based estimation methods such as mixed logit models.

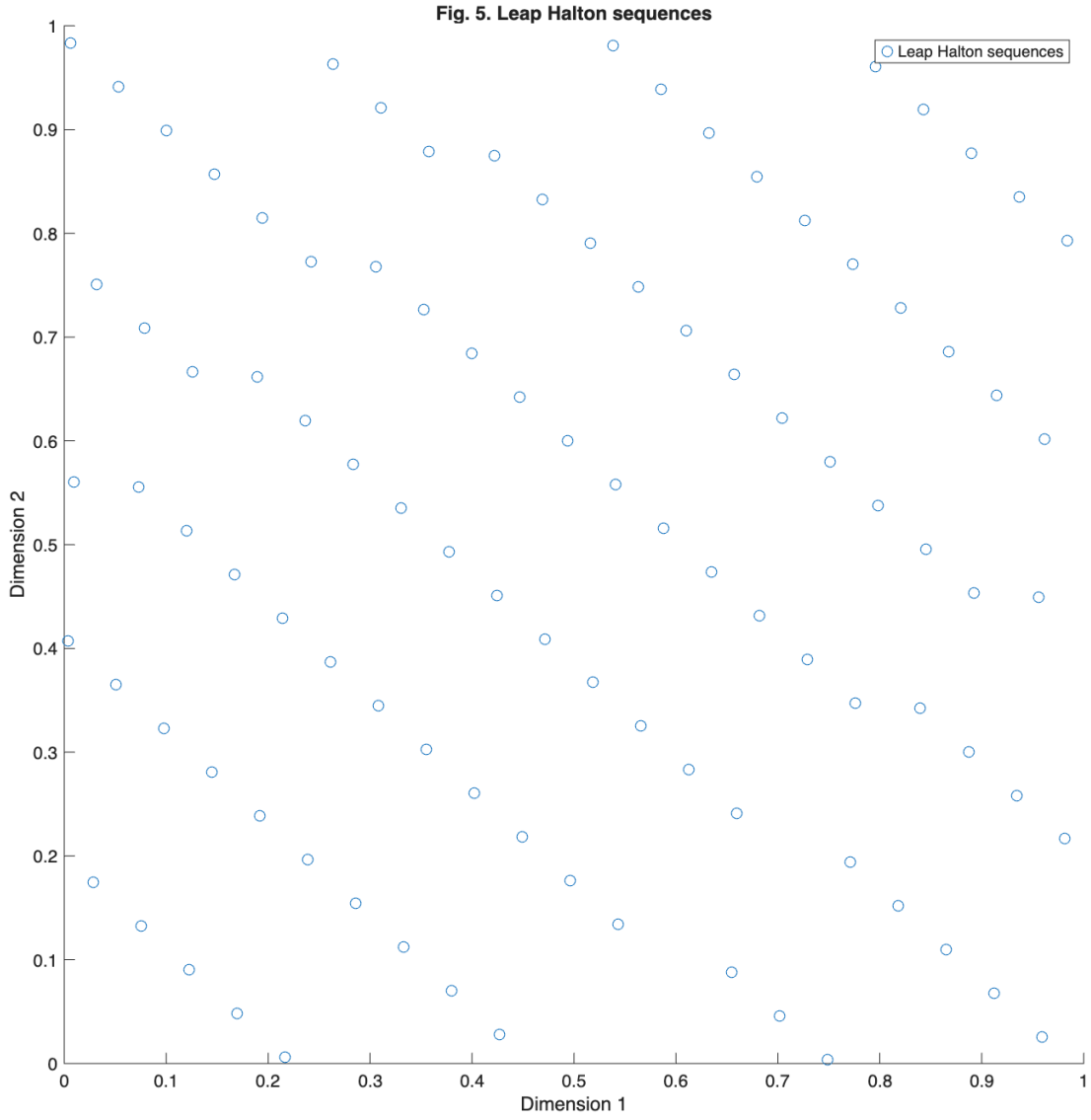


Fig. 5. Leap Halton sequences

Halton sequences are defined on the unit interval, so in two dimensions they naturally map to a unit square and can be visualized directly, as done in Figures 2-5. In contrast, the standard normal density is unbounded and curved, making it impossible to fully represent in a flat two-dimensional plot. To capture its shape and how draws populate its volume, a three-dimensional plot is needed. For practitioners, such visualizations are especially useful: they reveal how transformed draws concentrate around the mean, how well the tails are covered, and whether the sequence achieves the desired dispersion under the target density. This helps diagnose simulation quality and informs decisions about randomization, scrambling, or sample size. This extension is presented in Figures 6-9.

Fig. 6. Standard Halton sampling vs. normal density

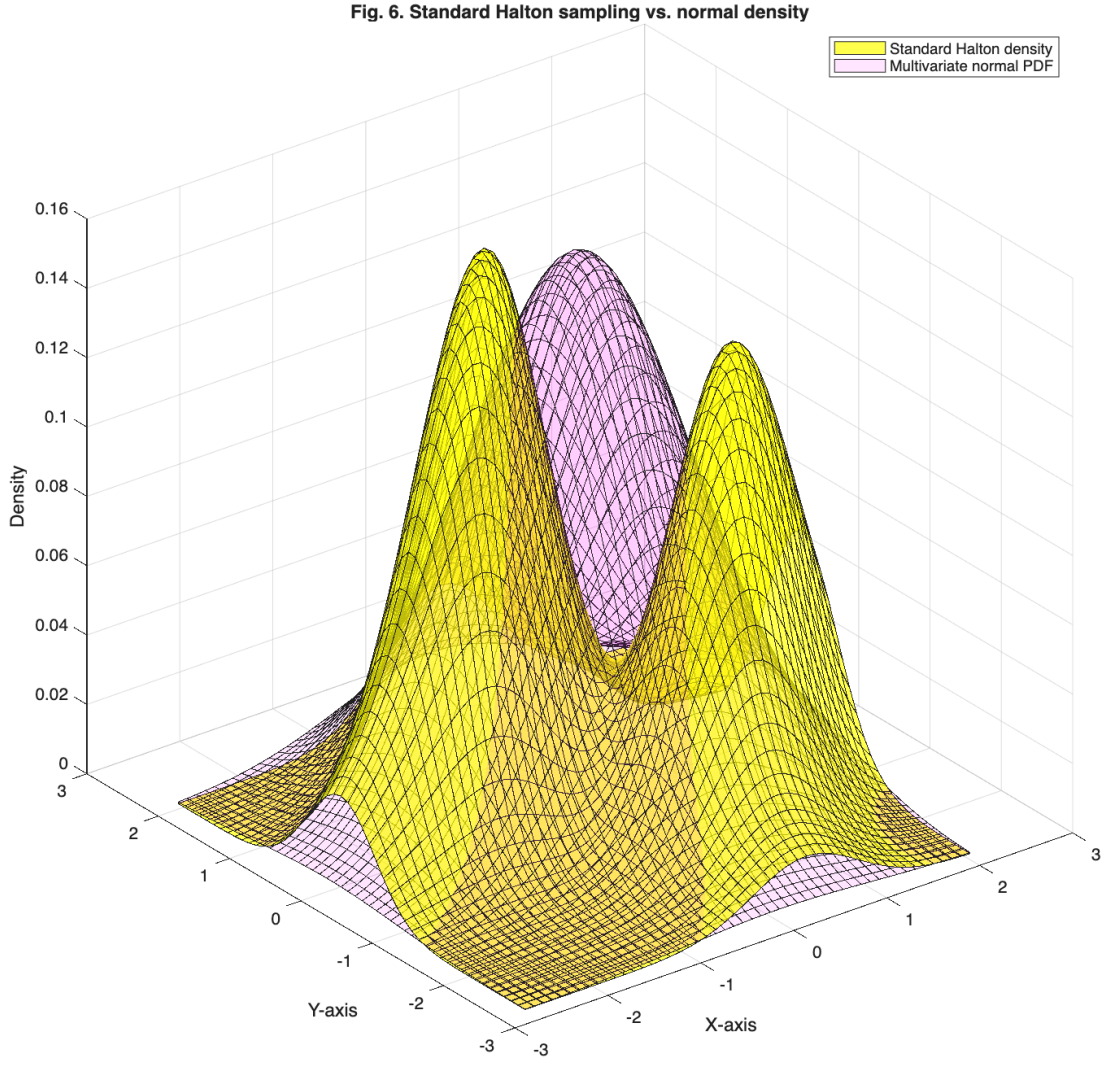


Fig. 7. Randomized Halton sampling vs. normal density

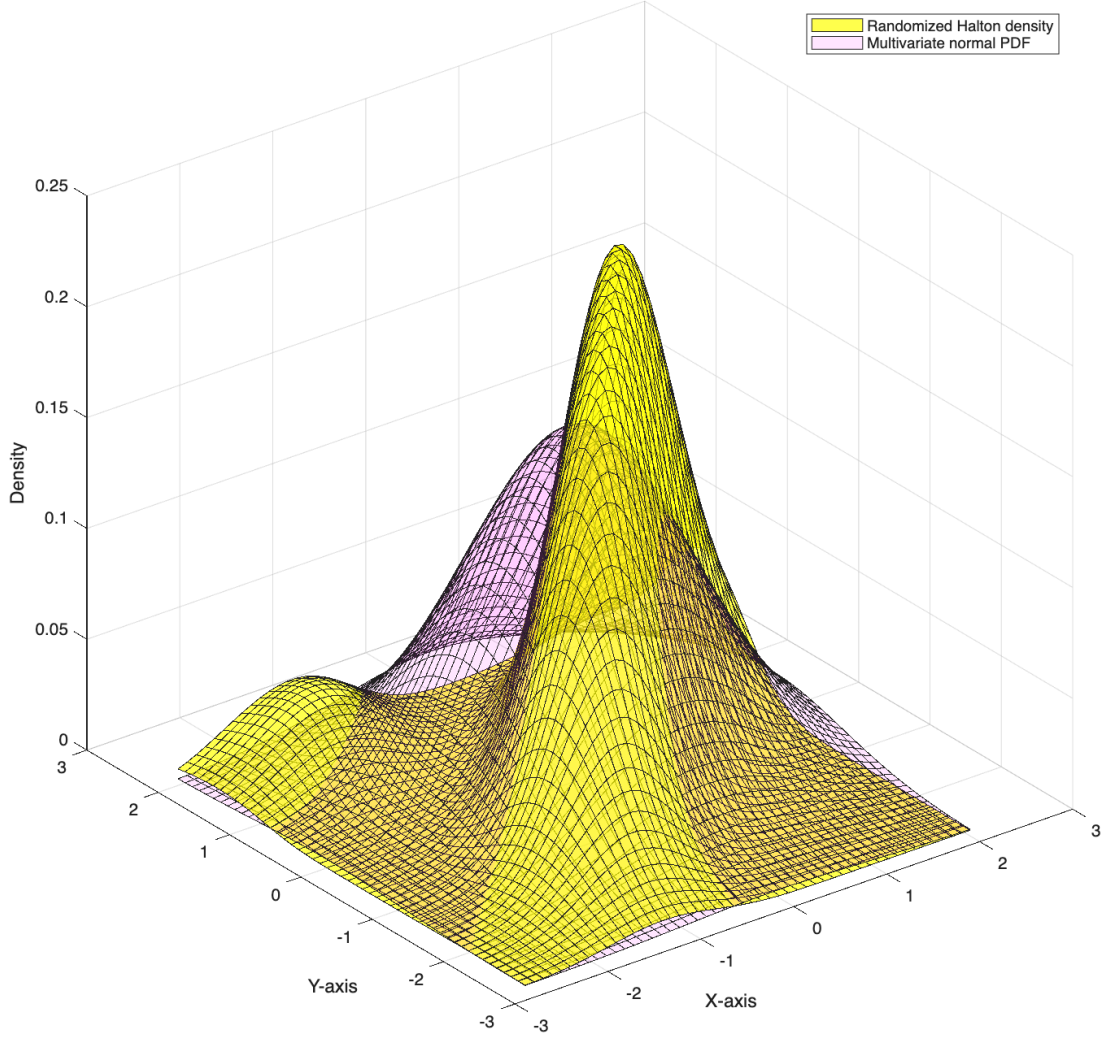


Fig. 8. Scrambled Halton sampling vs. normal density

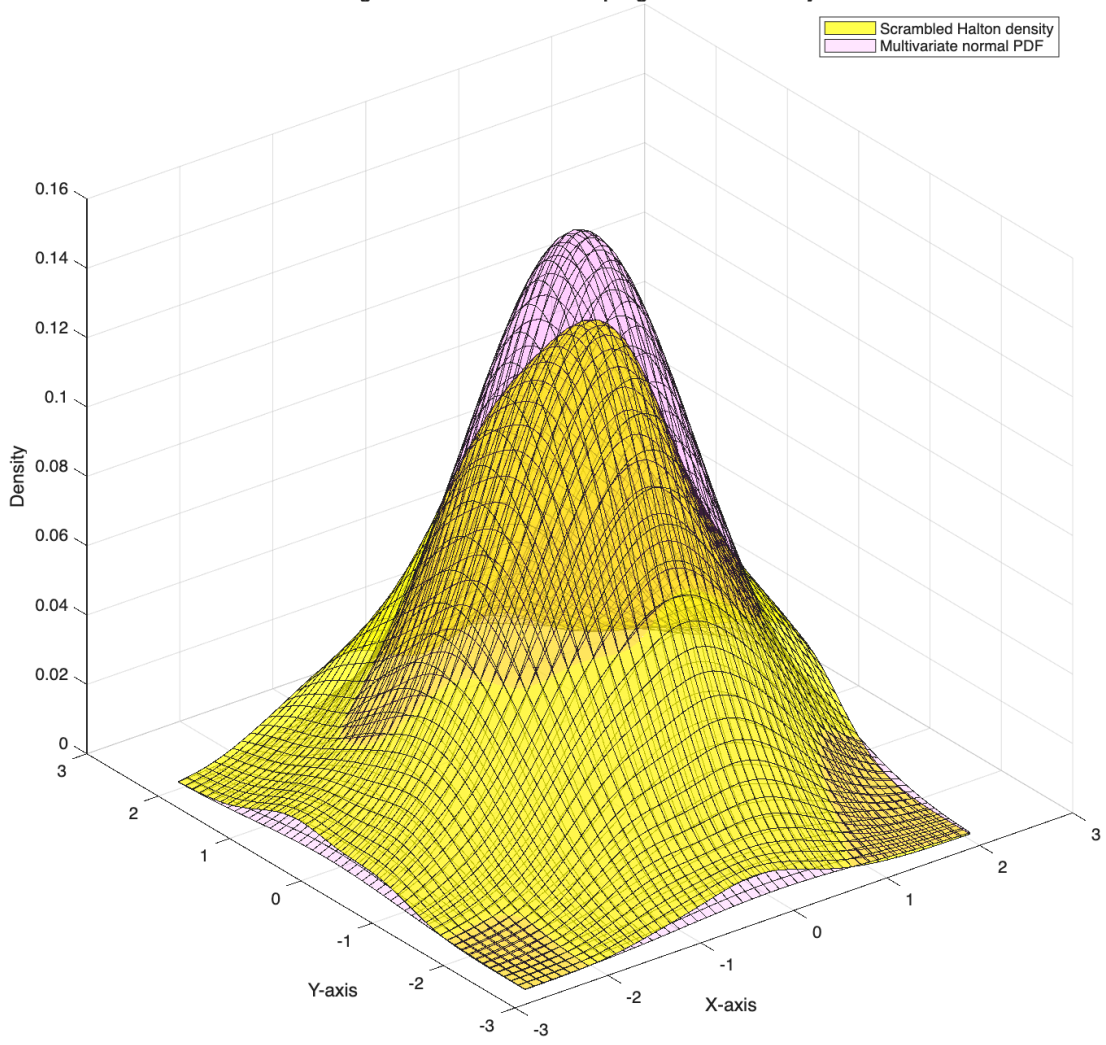
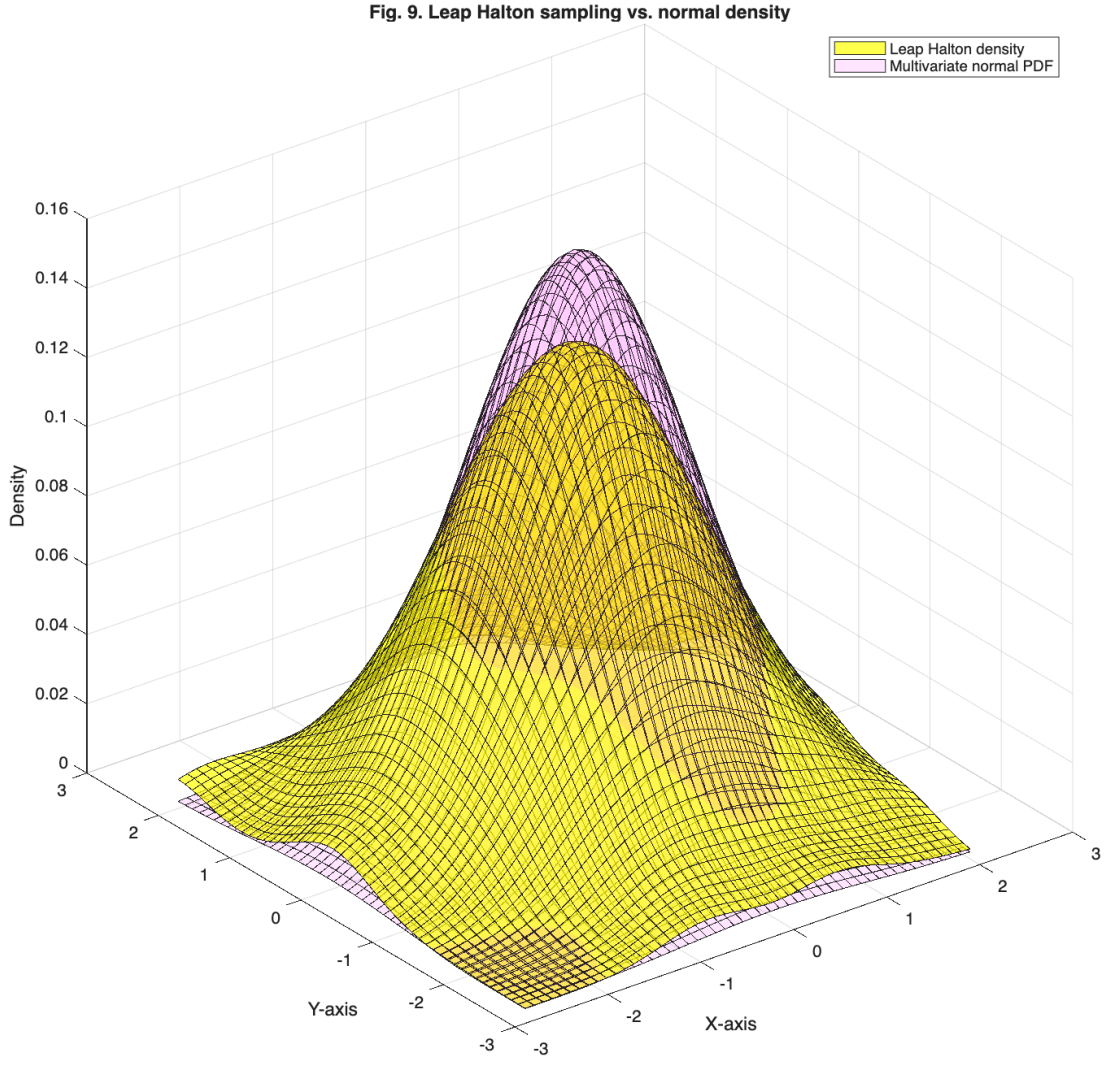


Fig. 9. Leap Halton sampling vs. normal density



The Halton sequence is a widely used method for generating quasi-random draws and offers clear advantages over purely random sampling, particularly in low-dimensional settings. As discussed in Train (2003), more advanced techniques exist with stronger theoretical properties, but the practical gains from adopting them must be weighed against the complexity of implementation. In many cases, simply increasing the number of draws may offer a more straightforward path to improved accuracy.

In the remaining sections, we present the MATLAB code used to generate the figures shown in the theory section. Beyond reproducing the visualizations, the code serves as a practical guide for drawing quasi-random samples using the external `halton` function. It demonstrates how to implement key features such as burn-in, randomization, and transformation to the standard normal distribution, offering a hands-on extension of the theoretical concepts discussed earlier.

3. Set the parameters of the Halton sequence

This section sets the basic parameters for generating a Halton sequence. Begin by clearing the workspace to avoid conflicts with existing variables. Define the number of observational units, the dimensionality of the sequence, and the number of draws per unit. Finally, specify the prime base that will be used to construct the Halton sequence. These settings establish the structure of the quasi-random sequence to be generated.

```
%% 3. Set the parameters of the Halton sequence

% 3.1. Clear the memory
clear;

% 3.2. Set the number of observational units
N = 100;

% 3.3. Set the number of dimensions
dimensions = 1;

% 3.4. Set the number of draws per observational unit
draws = 1;

% 3.5. Define prime base for Halton sequence generation
prime = 3;
```

4. Sampling with Halton sequences and uniform distribution

Here we demonstrate two methods for generating samples. First, we use the external `halton` function to produce quasi-random draws by specifying its input arguments: the prime base used to construct the sequence, the leap size controlling spacing between draws, the number of initial points to discard (burn-in), and logical flags for scrambling and randomization. These parameters control the structure and dispersion of the generated sequence. In this example, both scrambling and randomization are turned off to preserve the original structure of the Halton sequence. Second, we generate purely random samples from a uniform(0,1) distribution using MATLAB's built-in `random` function. This side-by-side setup allows for comparison between structured quasi-random sampling and standard random sampling.

```
%% 4. Sampling with Halton sequences and uniform distribution

% 4.1. Generate a Halton sequence
[H,~] = halton(N,dimensions,draws, ...
    'prime',prime','leap',0,'scramble',0,'random',0,'burn',50);

% 4.2. Generate samples from a uniform(0,1) distribution
U = random("Uniform",0,1,[N,1]);
```

5. Plot the Halton and uniform draws

Here we provide a visual comparison between Halton draws and uniform random draws over the unit interval. We create a scatter plot where both sets of samples are displayed along the horizontal axis, with fixed vertical offsets to separate them visually; Halton draws at height 0.4 and uniform draws at 0.6. The plot illustrates how, at the same sample size, Halton draws are more evenly spread across the interval, avoiding clustering and gaps typical of purely random samples. This supports the observation that quasi-random sequences offer better coverage, even with a small number of draws

```
%% 5. Plot the Halton and uniform draws

% 5.1. Plot
figure
hold on
scatter(H,0.4*ones(N,1),'DisplayName',"Halton draws");
scatter(U,0.6*ones(N,1),'DisplayName',"Uniform draws");
axis([0 1 0 1])
legend('show')
title("Fig. 1. Comparison of Halton and uniform draws")
hold off
```

6. Set the parameters of the Halton sequences

Here we specify parameters for comparing different variants of Halton sequences, as discussed in Train and Bhat. We specify the number of observational units, the dimensionality of the sequence, and the number of draws per unit. Importantly, we select prime bases known to produce strong inter-dimensional correlation in early draws, which helps illustrate the impact of various sequence modifications. These draws will later be transformed using the inverse normal CDF and used to estimate kernel densities, allowing for visual comparison against the standard bivariate normal distribution.

```
%% 6. Set the parameters of the Halton sequences

% 6.1. Clear the memory
clear;

% 6.2. Set the number of observational units
N = 100;

```

```
% 6.3. Set the number of dimensions
dimensions = 2;

% 6.4. Set the number of draws per observational unit
draws = 1;

% 6.5. Define prime bases for Halton sequence generation
prime = [43,47];
```

7. Halton sequences and transformation to standard normal distribution

This section generates four variants of Halton sequences in two dimensions, each designed to reduce correlation and improve coverage in different ways. The first variant is a regular Halton sequence with no leap, scrambling, or randomization. The second applies scrambling, which permutes the digits of the sequence to break deterministic patterns. The third introduces a leap of 4, spacing out the draws to reduce clustering. The fourth uses randomization, which shifts the sequence uniformly to add stochastic variation. Each sequence is also transformed to the standard normal distribution using the inverse CDF, producing a set of draws suitable for simulation-based estimation.

```
%% 7. Halton sequences and transformation to standard normal dist.

% 7.1. Regular Halton
[H_standard_2D,Z_standard_2D] = halton(N,dimensions,draws, ...
    'prime',prime,'leap',0,'scramble',0,'random',0,'burn',50);

% 7.2. Scrambled Halton
[H_scramble_2D,Z_scramble_2D] = halton(N,dimensions,draws, ...
    'prime',prime,'leap',0,'scramble',1,'random',0,'burn',50);

% 7.3. Leap Halton
[H_leap_2D,Z_leap_2D] = halton(N,dimensions,draws, ...
    'prime',prime,'leap',4,'scramble',0,'random',0,'burn',50);

% 7.4. Randomized Halton
[H_random_2D,Z_random_2D] = halton(N,dimensions,draws, ...
    'prime',prime,'leap',0,'scramble',0,'random',1,'burn',50);
```

8. Visual comparison of standard and scrambled Halton draws

Here we present a visual comparison of four Halton sequence variants in two dimensions: standard, randomized, scrambled, and leap. Each scatter plot shows how the draws are distributed across the unit square, highlighting differences in structure and dispersion. These figures help illustrate how each modification affects the uniformity and correlation of the sequence.

```
%% 8. Visual comparison of standard and scrambled Halton draws

% 8.1. Standard Halton sequences in 2D
figure
```

```
scatter(H_standard_2D(:,1),H_standard_2D(:,2), ...
    'DisplayName',"Standard Halton sequences")
ylabel("Dimension 2")
xlabel("Dimension 1")
legend('show')
title("Fig. 2. Standard Halton sequences")

% 8.2. Randomized Halton sequences in 2D
figure
scatter(H_random_2D(:,1),H_random_2D(:,2), ...
    'DisplayName',"Randomized Halton sequences")
ylabel("Dimension 2")
xlabel("Dimension 1")
legend('show')
title("Fig. 3. Randomized Halton sequences")

% 8.3. Scrambled Halton sequences in 2D
figure
scatter(H_scramble_2D(:,1),H_scramble_2D(:,2), ...
    'DisplayName',"Scrambled Halton sequences")
ylabel("Dimension 2")
xlabel("Dimension 1")
legend('show')
title("Fig. 4. Scrambled Halton sequences")

% 8.4. Leap Halton sequences in 2D
figure
scatter(H_leap_2D(:,1),H_leap_2D(:,2), ...
    'DisplayName',"Leap Halton sequences")
ylabel("Dimension 2")
xlabel("Dimension 1")
legend('show')
title("Fig. 5. Leap Halton sequences")
```

9. Density estimation and 3D surface plot setup

This section sets up a grid over the two-dimensional space and estimates kernel densities for each variant of the transformed Halton draws. It uses the `ksdensity` function to approximate the probability density function from the samples, reshapes the results for surface plotting, and computes a reference density from the bivariate standard normal distribution for comparison. This allows for visual evaluation of how closely each Halton variant approximates the target distribution.

```
%% 9. Density estimation and 3D surface plot setup

% 9.1. Define grid for evaluating density surfaces
x1 = linspace(min(Z_standard_2D(:,1)),max(Z_standard_2D(:,1)),50);
x2 = linspace(min(Z_standard_2D(:,2)),max(Z_standard_2D(:,2)),50);
[X1,X2] = meshgrid(x1,x2);
grid_points = [X1(:),X2(:)];
```

```
130
131 % 9.2. Turn the transformed Halton draws into a density function
132 [f_standard,~] = ksdensity(Z_standard_2D,grid_points,'Function','pdf');
133 [f_random,~] = ksdensity(Z_random_2D,grid_points,'Function','pdf');
134 [f_scramble,~] = ksdensity(Z_scramble_2D,grid_points,'Function','pdf');
135 [f_leap,~] = ksdensity(Z_leap_2D,grid_points,'Function','pdf');
136
137 % 9.3. Reshape the densities for the surface plot
138 F_standard = reshape(f_standard,size(X1));
139 F_random = reshape(f_random,size(X1));
140 F_scrambled = reshape(f_scramble,size(X1));
141 F_leap = reshape(f_leap,size(X1));
142
143 % 9.4. Compute reference density from bivariate standard normal dist.
144 Z_reference = mvnpdf(grid_points);
145 Z_reference = reshape(Z_reference,size(X1));
```

10. Compare Halton-based sampling densities to multivariate normal in 3D

This section visualizes and compares the kernel density estimates derived from four Halton sequence variants against the theoretical bivariate standard normal distribution. The variants are standard, randomized, scrambled, and leap. Each figure overlays the estimated density surface from the transformed Halton draws with the reference normal density, allowing for a direct visual assessment of how well each sampling method approximates the target distribution. The use of transparency and color helps distinguish the two surfaces, and the 3D view highlights differences in concentration, spread, and tail behavior across methods.

```
147 %% 10. Compare Halton-based sampling densities to multivariate normal
148
149 % 10.1. Standard Halton vs. multivariate normal density
150 figure
151 hold on
152 surf(X1,X2,F_standard,'FaceColor','y','FaceAlpha',0.7, ...
153     'DisplayName',"Standard Halton density");
154 surf(X1,X2,Z_reference,'FaceColor','m','FaceAlpha',0.1, ...
155     'DisplayName',"Multivariate normal PDF");
156 ylabel('Y-axis');
157 xlabel('X-axis');
158 zlabel('Density');
159 legend('show');
160 title("Fig. 6. Standard Halton sampling vs. normal density");
161 view(3);
162 grid on
163 hold off
164
165 % 10.2. Randomized Halton vs. multivariate normal density
166 figure
167 hold on
168 surf(X1,X2,F_random,'FaceColor','y','FaceAlpha',0.7, ...
169     'DisplayName',"Randomized Halton density");
```

```
surf(X1,X2,Z_reference,'FaceColor','m','FaceAlpha',0.1, ...
    'DisplayName',"Multivariate normal PDF");
ylabel('Y-axis');
xlabel('X-axis');
zlabel('Density');
legend('show');
title("Fig. 7. Randomized Halton sampling vs. normal density");
view(3);
grid on
hold off

% 10.3. Scrambled Halton vs. multivariate normal density
figure
hold on
surf(X1,X2,F_scrambled,'FaceColor','y','FaceAlpha',0.7, ...
    'DisplayName',"Scrambled Halton density");
surf(X1,X2,Z_reference,'FaceColor','m','FaceAlpha',0.1, ...
    'DisplayName',"Multivariate normal PDF");
ylabel('Y-axis');
xlabel('X-axis');
zlabel('Density');
legend('show');
title("Fig. 8. Scrambled Halton sampling vs. normal density");
view(3);
grid on
hold off

% 10.4. Leap Halton vs. multivariate normal density
figure
hold on
surf(X1,X2,F_leap,'FaceColor','y','FaceAlpha',0.7, ...
    'DisplayName',"Leap Halton density");
surf(X1,X2,Z_reference,'FaceColor','m','FaceAlpha',0.1, ...
    'DisplayName',"Multivariate normal PDF");
ylabel('Y-axis');
xlabel('X-axis');
zlabel('Density');
legend('show');
title("Fig. 9. Leap Halton sampling vs. normal density");
view(3);
grid on
hold off
```

12. Final notes

This file is prepared and copyrighted by Elisabeth Beusch, Hartog Horsch, and Tunga Kantarcı. This file and the accompanying MATLAB file are available on GitHub and can be accessed using this link. This file has made use of two references. First is Bhat, C. R., 2003. Simulation estimation of mixed discrete choice models using randomized and scrambled Halton sequences.

Transportation Research Part B: Methodological, 37 (9), 837-855. The second is Train, K., 2003. Discrete Choice Methods with Simulation. Cambridge University Press.