On the Beauty of Uniform Distribution Modulo One

Karl Entacher

This article provides insights into the theory of uniform distribution of sequences modulo one. Basic examples illustrate elementary concepts and special graphical presentations of (local) discrepancy, the classical measure of uniform distribution, exhibit our understanding of the beauty of this theory.

■ Introduction

The *theory of uniform distribution modulo one* was developed extensively within and among several mathematical disciplines and numerous applications, mostly in the fields of *Monte Carlo* and *quasi-Monte Carlo methods*, which include areas like numerical integration, random number generation, stochastic simulation, and approximation theory.

The central goals of this theory are the assessment of equidistribution and the construction of well-distributed point sets and sequences in various mathematical spaces.

The following sections contain several supporting ideas for introducing the theory of uniform distribution in education, offer additional information for researchers, and supplement the theory with impressive images. We start with some elementary examples. Section 2 treats *discrepancy*, which is the classical measure of uniform distribution. In Section 3 we use a special graphical presentation of local discrepancy, showing the beauty of uniform distribution. Section 4 considers further examples of point sets and the graphical visualization of the quality of their distribution.

Classical and recent concepts of the theory and further references are discussed in [1, 2]. For further information on quasi-Monte Carlo methods and their applications, see [3, 4, 5]. For efficient *Mathematica* implementations of "quasi-random numbers," see QR Stream [6]. Note that only small point sets are used for our illustrations; the number of points used in practice is substantially larger.

■ Equidistributed Point Sets

The classical goal of the theory of uniform distribution is the construction of well-distributed point sets and sequences; a classical example is the *Hammersley point set* in dimension two.

Consider integers n, $0 \le n < 2^m$, with $m \in \mathbb{N}$, and let $n = n_0 + n_1\, 2 + \cdots + n_{m-1}\, 2^{m-1}$ be the binary expansion of n. The Hammersley point set in dimension $s = 2$ is defined by

$$P_0 := \{(x_n,\, y_n) = (0.\, n_{m-1}\, n_{m-2} \ldots n_0,\ 0.\, n_0\, n_1 \ldots n_{m-1}),\ 0 \le n < 2^m\}. \qquad (1)$$

The first coordinates x_n of P_0 yield the *van der Corput sequence* for $0 \le n < 2^m$, and the second coordinates y_n consecutively pass through the numbers $n/2^m$. Figure 1 exhibits the behavior of both coordinates x_n and y_n. Note that the sequence x_n orders the numbers in a balanced way in the upper and lower half of the unit interval [0, 1).

```
In[1]:= l = MyNet[4, 0];
        x = Map[First, l];
        y = Map[Last, l];
        ll = {Table[{i, ToString[i - 1]}, {i, 1, 16, 2}], Automatic};
        g1 = ListPlot[x, PlotRange → {0, 1}, Ticks → ll,
           AxesLabel → {"n", "xn"}, PlotStyle → AbsolutePointSize[2],
           DisplayFunction → Identity, AspectRatio → 1];
        g2 = ListPlot[y, PlotRange → {0, 1}, Ticks → ll,
           AxesLabel → {"n", "yn"}, PlotStyle → AbsolutePointSize[2],
           DisplayFunction → Identity, AspectRatio → 1];
        Show[GraphicsArray[{g1, g2}], DisplayFunction → $DisplayFunction]
```

From In[1]:=

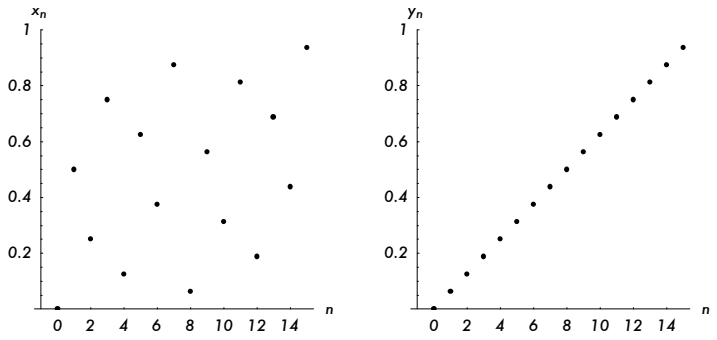

Figure 1. Behavior of the coordinates x_n and y_n of the Hammersley point set for $m = 4$.

If we truncate t bits from each coordinate of P_0, we obtain a modified point set P_t:

$$P_t := \{(0.\, n_{m-1}\, n_{m-2} \ldots n_t,\ 0.\, n_0\, n_1 \ldots n_{m-t-1}),\ 0 \le n < 2^m\}. \qquad (2)$$

Using the function `MyNet[m,t]`, the following graphics illustrate the structural behavior of P_t. Figure 2 shows plots for $m = 6$ and $t = 0, 1, 2, 3$ (from left to right).

```
In[8]:= MyNet[m_, t_] :=
         Map[FromDigits[{Drop[#, -t], 0}, 2] &, {Reverse[#], #} & /@
           Flatten[Outer[List, Sequence @@ Table[{0, 1}, {m}]], m - 1], {2}]
```

```
In[9]:= Show[GraphicsArray[ Partition[Table[Graphics[
            {AbsolutePointSize[2], Map[Point, MyNet[6, i]]},
            Frame → True, AspectRatio → 1, Axes → None,
            PlotRange → {{-0.02, 1.01}, {-0.02, 1.01}},
            FrameTicks → None, DefaultColor → Hue[0.65, 1, 0.75],
            FrameStyle → GrayLevel[0]], {i, 0, 3}], 2]]]
```

From In[9]:=

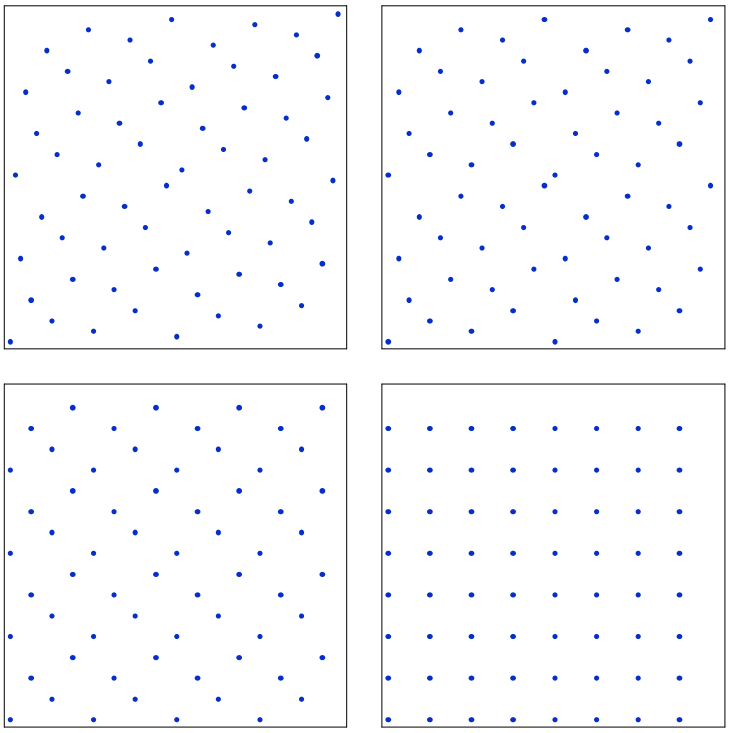

Figure 2. Hammersley point set with bit truncation for $m = 6$ and $t = 0, 1, 2, 3$.

For increasing t, the resolution of the points of P_t decreases and therefore the distribution quality decreases as well. With m even and $0 \le t \le \frac{m}{2}$, the point sets change from the well-distributed set P_0 (each grid line with resolution $1/2^m$ contains exactly one point) to the classical uniform lattice with 2^m points. In general, for the set P_t, each grid line with resolution $1/2^{m-t}$ contains exactly 2^t points (see also Figure 3).

In more modern language, the point set P_t is called a (t, m, s)-net in base 2 and dimension $s = 2$ (m is the parameter for the number of points $N = 2^m$ and t the parameter for the quality of distribution). The definition of an arbitrary (t, m, s)-net requires a slightly modified form of the distribution property described earlier: every half-open elementary s-dimensional subinterval $\mathcal{J}$ of $I = [0, 1)^s$ with volume 2^{m-t} has to contain exactly 2^t points. Figure 3 visualizes the latter requirement for P_0 and P_1, $m = 4$. Some representative elementary two-dimensional intervals $\mathcal{J}$ are indicated by colored rectangles (the lighter rectangles cover the darker ones). Note that the constant number of points within such intervals, by definition, must also be valid for all properly shifted elementary rectangles within I.

```
In[10]:= vec1 = Table[i / 2^4, {i, 1, 2^4 - 1}];
         vec2 = Table[i / 2^3, {i, 1, 2^3 - 1}];
         g1 = Graphics[{{Hue[0.65, 1, 0.75], Rectangle[{0.0, 0.0}, {1, 1 / 16}]},
                {Hue[0.65, 0.9, 0.75], Rectangle[{0.0, 0.0}, {1 / 16, 1}]},
                {Hue[0.65, 0.6, 0.75], Rectangle[{0.0, 0.0}, {1 / 2, 1 / 8}]},
                {Hue[0.65, 0.4, 0.75], Rectangle[{0.0, 0.0}, {1 / 8, 1 / 2}]},
                {Hue[0.65, 0.2, 0.75], Rectangle[{0.0, 0.0}, {1 / 4, 1 / 4}]},
            {AbsolutePointSize[3], Map[Point, MyNet[4, 0]]} },
           {AspectRatio → 1,  Frame → True, Axes → None, FrameTicks → None,
            PlotRange → {{0.0, 1.0}, {0.0, 1.0}}, GridLines → {vec1, vec1}} ];
         g2 = Graphics[{
                {Hue[0.65, 1, 0.75], Rectangle[{0.0, 0.0}, {1, 1 / 8}]},
                {Hue[0.65, 0.9, 0.75], Rectangle[{0.0, 0.0}, {1 / 8, 1}]},
                {Hue[0.65, 0.6, 0.75], Rectangle[{0.0, 0.0}, {1 / 2, 1 / 4}]},
                {Hue[0.65, 0.4, 0.75], Rectangle[{0.0, 0.0}, {1 / 4, 1 / 2}]},
            {AbsolutePointSize[3], Map[Point, MyNet[4, 1]]} },
           {AspectRatio → 1, Frame → True, Axes → None, FrameTicks → None,
            PlotRange → {{0.0, 1.0}, {0.0, 1.0}}, GridLines → {vec2, vec2}} ];
         Show[GraphicsArray[{g1, g2}]]
```

From In[10]:=

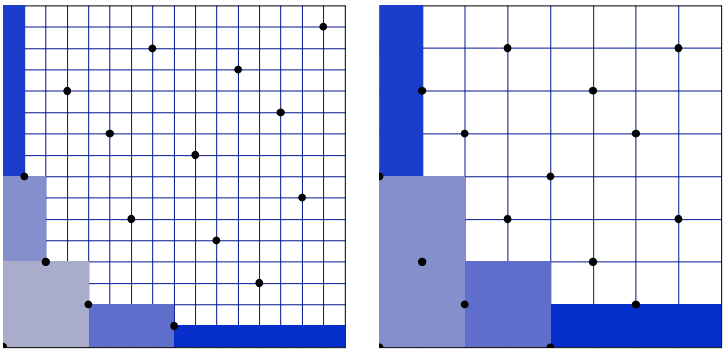

Figure 3. Visualization of the (t, m, s)-net property for P_0 and P_1, $m = 4$.

A modern task in the theory of uniform distribution is the construction of arbitrary (t, m, s)-nets with small parameters t in large dimensions s.

■ Discrepancy

In the previous section, the term equidistributed point set was used without defining equidistribution. An assessment of distribution quality needs a suitable measure. The classical measure in the theory of uniform distribution is *discrepancy*, which in a certain sense measures the deviation of a point set from an ideal distribution.

We will introduce a special form of discrepancy, the so-called *star-discrepancy*. Consider a point set $P = (x_n)_{n=0}^{N-1}$, in the s-dimensional unit cube $I = [0, 1)^s$. The star-discrepancy $D_N^*(P)$ of P is defined as

$$D_N^*(P) := \sup_{\mathcal{J} \in Y^*} D(\mathcal{J}, P), \text{ where } D(\mathcal{J}, P) := \left| \frac{\text{number of } x_n \in \mathcal{J}}{N} - \text{Vol}(\mathcal{J}) \right|, \quad (3)$$

and Y^* denotes the class of all multidimensional subintervals $\mathcal{J}$ of I of the form $\mathcal{J} := \prod_{i=1}^{s} \mathcal{J}_i$, where $\mathcal{J}_i = [0, u_i)$ and $0 \le u_i \le 1$. Hence, the star-discrepancy is attained for a (half-open) subinterval of I with the left corner in the origin (see Figure 3 for some *discrete* examples $\mathcal{J} = [0, u_1) \times [0, u_2)$) providing maximal deviation of the relative number of points within the interval from its volume $\text{Vol}(\mathcal{J}) = \prod_{i=1}^{s} u_i$. The number $D(\mathcal{J}, P)$ is called *local discrepancy*.

The measure $D_N^*(P)$ may also be seen as a number-theoretical adaptation of a well-known goodness-of-fit test in statistics, the two-sided Kolmogorov–Smirnov test that is used to test the hypothesis that a sample P stems from a particular continuous probability distribution. In our case, the target distribution is the uniform distribution on I.

The computation of discrepancy is very time consuming with a complexity of $O(N^s)$ for N points in the s-dimensional unit cube. Hence, an assessment of uniform distribution is usually achieved by theoretical discrepancy estimates or by an application of several other measures of equidistribution, which, although they are easier to compute, are sometimes not that intuitive.

Furthermore, every numerical application of uniformly distributed point sets works with finite precision numbers. Therefore, it is important to apply discrepancy for finite precision subintervals (see Figure 3 for some examples). This leads to the concept of *discrete discrepancy* [3], which is defined in the same way as in equation (3), but now Y^* denotes the class of all (discrete) subintervals $\mathcal{J}$ of the form $\mathcal{J} = \prod_{i=1}^{s} \mathcal{J}_i$, where $\mathcal{J}_i = [0, a_i / M)$, with integers $0 \le a_i \le M$.

The `DiscreteDiscrepancy[p,m]` function gives a list of local discrepancies $D(\mathcal{J}, P)$ for all discrete intervals $\mathcal{J}$ with resolution $M = 2^m$, $m \ge 2$ and a given point set P. To calculate this function, we represent the intervals $\mathcal{J}$ by the list of endpoints $\left(\frac{i}{2^m}, \frac{j}{2^m}\right)$, $1 \le i, j \le 2^m$, and use simple list operations.

```
In[15]:= DiscreteDiscrepancy[p_List, m_Integer] :=
           Module[{intervals, func},
            intervals = Table[{i, j} / 2^m, {i, 1, 2^m}, {j, 1, 2^m}];
            intervals = Flatten[intervals, 1];
            func[l_] := Length[Select[p,
               (#[[1]] < l[[1]] && #[[2]] < l[[2]]) &]];
            Abs[Map[func, intervals] / Length[p] -
              Map[Apply[Times, #] &, intervals]]];
```

The following matrix shows the output of `DiscreteDiscrepancy[`P_1`,3]`. The value at position (i, j) equals local discrepancy $D(\mathcal{J}, P)$ for the interval $\mathcal{J}$ with endpoints $\left(\frac{i}{2^m}, \frac{j}{2^m}\right)$, $1 \le i, j \le 2^m$.

```
In[16]:= Example = DiscreteDiscrepancy[MyNet[3, 1], 3];
         MatrixForm[ Partition[Example, 8]]
```

Out[17]//MatrixForm=

$$\begin{pmatrix}
\frac{7}{64} & \frac{3}{32} & \frac{5}{64} & \frac{1}{16} & \frac{11}{64} & \frac{5}{32} & \frac{9}{64} & \frac{1}{8} \\
\frac{3}{32} & \frac{1}{16} & \frac{1}{32} & 0 & \frac{3}{32} & \frac{1}{16} & \frac{1}{32} & 0 \\
\frac{5}{64} & \frac{1}{32} & \frac{7}{64} & \frac{1}{16} & \frac{9}{64} & \frac{3}{32} & \frac{11}{64} & \frac{1}{8} \\
\frac{1}{16} & 0 & \frac{1}{16} & 0 & \frac{1}{16} & 0 & \frac{1}{16} & 0 \\
\frac{11}{64} & \frac{3}{32} & \frac{9}{64} & \frac{1}{16} & \frac{15}{64} & \frac{5}{32} & \frac{13}{64} & \frac{1}{8} \\
\frac{5}{32} & \frac{1}{16} & \frac{3}{32} & 0 & \frac{5}{32} & \frac{1}{16} & \frac{3}{32} & 0 \\
\frac{9}{64} & \frac{1}{32} & \frac{11}{64} & \frac{1}{16} & \frac{13}{64} & \frac{3}{32} & \frac{15}{64} & \frac{1}{8} \\
\frac{1}{8} & 0 & \frac{1}{8} & 0 & \frac{1}{8} & 0 & \frac{1}{8} & 0
\end{pmatrix}$$

Discrete discrepancy is the maximum of the matrix, $\frac{15}{64}$. In our example, the maximum is attained for two intervals $\mathcal{J}$ with endpoints $(\frac{5}{8}, \frac{5}{8})$ and $(\frac{7}{8}, \frac{7}{8})$. As one may guess, calculating `DiscreteDiscrepancy[`p`,`m`]` in this simple way becomes very difficult for increasing m. However, we can use this function to analyze the structure of local discrepancy for small point sets in order to gain elementary information for a subsequent theoretical discrepancy estimation of larger point sets. This very useful strategy allowed the author to construct a method to estimate the star-discrepancy of arbitrary nets P_t [7].

■ Visualizations

In this section, we visualize the structure of local discrepancy $D(\mathcal{J}, P)$ for our point sets P_t, $t \ge 0$, for all discrete intervals $\mathcal{J}$ with resolution $M = 2^m$. We start with a visualization in dimension two. Therefore, we use `DiscreteDiscrepancy[ ]` and calculate the list of values $D(\mathcal{J}, P)$ for all such two-dimensional intervals, which are represented by their endpoints as described earlier. The following function simply uses `RasterArray[ ]` to color the list of local discrepancies. Small values of $D(\mathcal{J}, P)$ are colored in dark blue. The larger the values become, the brighter the coloring. The brightest squares show where the maximum (and therefore the discrete discrepancy) is attained. The explicit value is shown above each plot.

```
In[18]:= Visualize2D[p_List, m_Integer] := Module[ {mi, ma, result},
           result = DiscreteDiscrepancy[p, m];
           mi = Min[result];
           ma = Max[result];
           Graphics[{ RasterArray[Partition[
              Map[ RGBColor[#, 0.1 + 0.9 #, 0.6] &,
                   result / (ma - mi)], 2^m]]},
           AspectRatio → 1, PlotRange → All,
           PlotLabel → StyleForm[TraditionalForm[ma]]] ];
```

Figure 4 shows `Visualize2D[p,m]` for P_t, $t = 0, 1, 2, 3$ and $m = 6$. Compared to Figure 2, these graphics reflect the change of distribution in a more impressive way.

```
In[19]:= Show[ GraphicsArray[
          {{Visualize2D[MyNet[6, 0], 6], Visualize2D[MyNet[6, 1], 6]},
           {Visualize2D[MyNet[6, 2], 6], Visualize2D[MyNet[6, 3], 6]}}]]
```

From In[19]:=

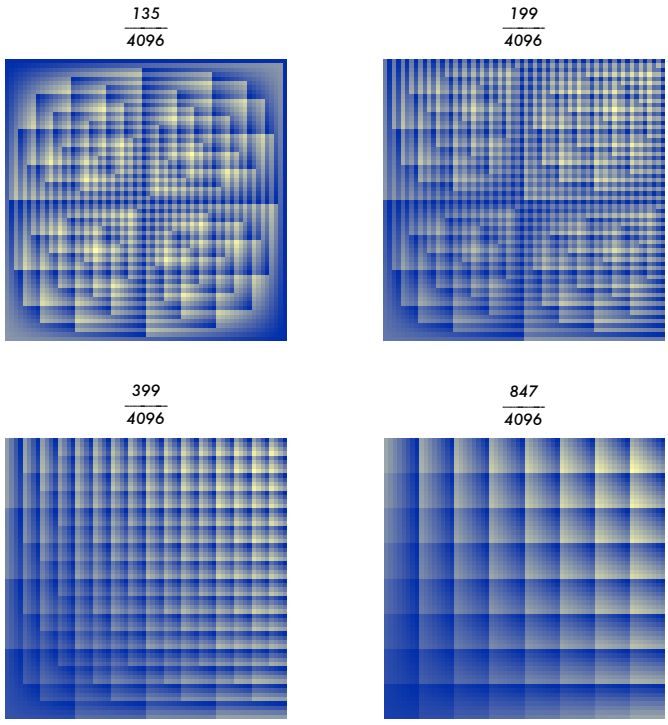

Figure 4. Visualization of local discrepancy for P_t, $t = 0, 1, 2, 3$ and $m = 6$.

Figure 5 also includes the points within the graphic for the larger case $m = 8$ and $t = 0$. A comparison of the first graphic in Figures 4 and 5 shows certain self-similarities. The reader is also asked to visualize the cases $m = 9, 10$.

```
In[20]:= g = Graphics[ {AbsolutePointSize[2],
            Map[Point, 2^8 MyNet[8, 0]]},
           Frame → None, AspectRatio → 1, Axes → None, FrameTicks → None];
         Show[Visualize2D[ MyNet[8, 0], 8], g]
```

From In[20]:=

$$\frac{711}{65536}$$

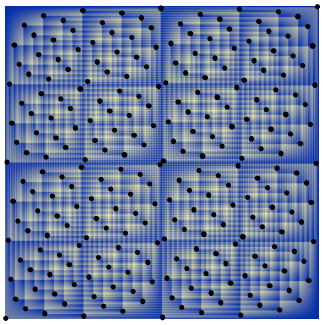

Figure 5. Visualization of local discrepancy of P_0 for $m = 8$.

In Figure 6, the distribution behavior of the point sets is visualized in three dimensions as well. The function `Visualize3D[`p`,`m`]` generates a `Graphics3D` object containing cuboids with heights corresponding to $D(\mathcal{J}, P)$. Figures 6 and 7 show the cases $m = 6$, $t = 0$, and $t = 3$. Two of the plots in Figure 4 are projections into the x-y plane of Figures 6 and 7.

```
In[22]:= Visualize3D[p_List, m_Integer] :=
          Module[ {mi, ma, result, tmp1, tmp2, pos, l, ll},
           l = Table[{i, j} / 2^m, {i, 1, 2^m}, {j, 1, 2^m}];
           l = Flatten[l, 1];
           result = DiscreteDiscrepancy[p, m];
           mi = Min[result]; ma = Max[result];
           tmp1 = (ma - mi) / 2^m; tmp2 = Union[result];
           pos = Table[Flatten[ Position[result, tmp2[[i]]]],
             {i, 1, Length[tmp2]} ];
           ll =
            Table[ {Flatten[{l[[pos[[i]][[j]]]], tmp2[[i]]}],
               Flatten[{l[[pos[[i]][[j]]]] + 1 / 2^m, tmp2[[i]] + tmp1}]},
             {i, 1, Length[tmp2]}, {j, 1, Length[pos[[i]]]} ];
           Table[ Graphics3D[{RGBColor[
               (i - 1) / Length[ll], (i - 1) / Length[ll], 0.7],
              PointSize[0.012], EdgeForm[Thickness[0.001]],
              Map[Cuboid[#[[1]], #[[2]]] &, ll[[i]]]},
             Ticks → {{1}, {0, 1}, {mi, ma}},
             PlotRange → {mi - 2 tmp1, ma + 2 tmp1},
             Boxed → True, BoxRatios → {1, 1, 1},
             ViewPoint → {-1.3, -1.4, 1.9}, Lighting → False,
             Axes → True],
            {i, 1, Length[ll]}] ];
```

```
In[25]:= m = 7;
         vec1 = Reverse[ Table[ 1 / 2^i, {i, 1, m} ]];
         vec2 = Reverse[ Table[ 1 / 3^i, {i, 1, m} ]];
         l = Range[2^m];
         l1 = Map[IntegerDigits[#, 2, m] &, l].vec1;
         l2 = Map[IntegerDigits[#, 3, m] &, l].vec2;
         hal = Transpose[{l1, l2}];
         g = Graphics[ {PointSize[0.02], Map[Point, hal]},
            Frame → True, AspectRatio → 1, Axes → None,
            FrameTicks → None, PlotRegion → {{0, 0.85}, {0, 0.85}}];
         Show[GraphicsArray[{g, Visualize2D[hal, m]}, GraphicsSpacing → 0]]
```

From In[25]:=

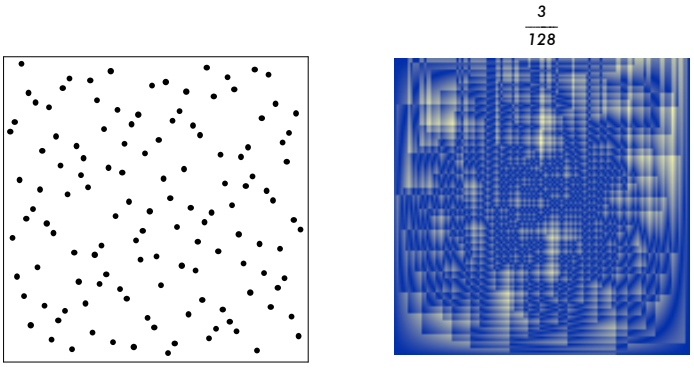

Figure 8. Local discrepancy of a Halton sequence ($m = 7$).

Lattices are essentially different from the nets shown in earlier sections. Special lattices are obtained if we produce all "overlapping" vectors (x_n, x_{n+1}) from a point set $P = (x_n)_{n=0}^{M-1}$, where the numbers x_n are generated with $x_n := y_n / M$ and y_n via the recurrence $y_{n+1} \equiv a\, y_n + b \pmod{M}$. The parameters a and b, $0 \le a$, $b < M$, and the starting value y_0 have to fulfill certain conditions in order to get the full period M for the recurrence (see [3, 8]). The parameter a is responsible for the distribution quality of the lattice. If a is chosen well, the vector $g = (1, a)$ is called a good lattice point, because the corresponding lattice is generated from multiples of g. In Figures 9 and 10 the modulus $M = 2^7$ and the additive constant $b = 1$. In Figure 9 we use the bad parameter $a = 125$ and in Figure 10 the good parameter $a = 117$.

```
In[34]:= m = 7;
         f[x_] := Mod[125 x + 1, 2^m];
         gen = NestList[f, 0, 2^m];
         gen = Partition[gen / 2^m, 2, 1];
         g = Graphics[ {PointSize[0.02], Map[Point, gen]},
            Frame → True, AspectRatio → 1, Axes → None,
            FrameTicks → None, PlotRegion → {{0, 0.85}, {0, 0.85}}];
         Show[GraphicsArray[{g, Visualize2D[gen, m]}, GraphicsSpacing → 0 ]]
```

From In[34]:=

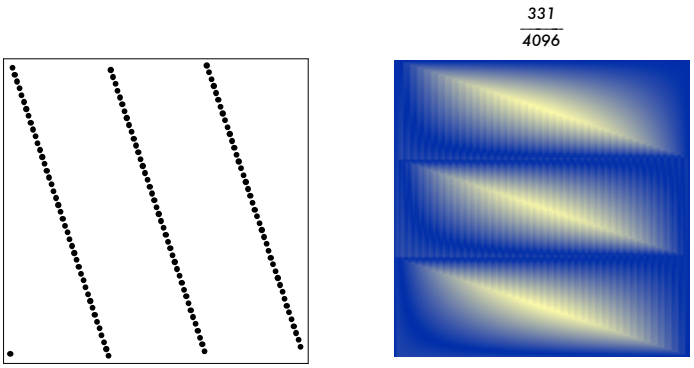

Figure 9. Local discrepancy of bad lattice points with $M = 2^7$ and $a = 125$.

```
In[40]:= m = 7;
         f[x_] := Mod[117 x + 1, 2^m];
         gen = NestList[f, 0, 2^m];
         gen = Partition[gen / 2^m, 2, 1];
         g = Graphics[ {PointSize[0.02], Map[Point, gen]},
            Frame → True, AspectRatio → 1, Axes → None, FrameTicks → None,
            PlotRegion → {{0, 0.85}, {0, 0.85}}];
         Show[GraphicsArray[{g, Visualize2D[gen, m]},
           GraphicsSpacing → 0]]
```

From In[40]:=

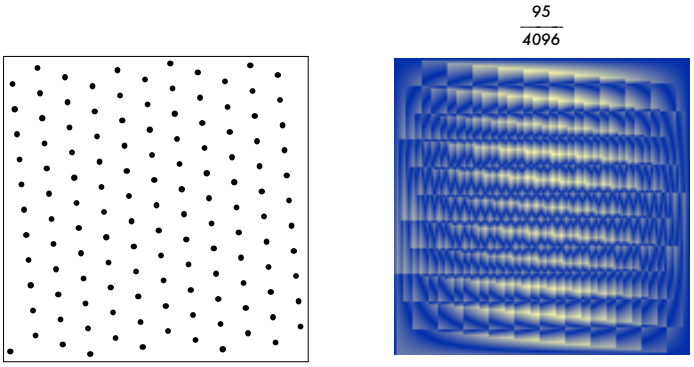

Figure 10. Local discrepancy of good lattice points with $M = 2^7$ and $a = 117$.

The weak distribution for $a = 125$ is clearly seen in the visualization of local discrepancy. The good lattice is more dispersed, like Figure 5. The reader is asked to try to compute the three-dimensional visualization of the bad lattice.

There is another important application of such lattices. The modular generation method yields a classical method for the generation of (*pseudo-*) *random numbers*. If one uses only a small sample of numbers from the previous recurrence with large modulus M (sample size lower than the square root of modulus M recommended), then the numbers behave like "real" random numbers in many applications. The generation method is called the *linear congruential generator* (LCG). Figure 11 visualizes a sample of $N = 2^6$ numbers from a widely used LCG called the "Minimal Standard" generator ($M = 2^{31} - 1$, $a = 16807$, $b = 0$) [9] (for further references see [10]).

In[46]:=
```
m = 6;
f[x_] := Mod[16807 x, 2^31 - 1];
gen = NestList[f, 1, 2^m];
gen = Partition[gen / (2^31 - 1), 2, 1];
g = Graphics[ {PointSize[0.02], Map[Point, gen]},
   Frame → True, AspectRatio → 1, Axes → None,
   FrameTicks → None, PlotRegion → {{0, 0.85}, {0, 0.85}}];
Show[GraphicsArray[{g, Visualize2D[gen, m]},
  GraphicsSpacing → 0]]
```

From In[46]:=

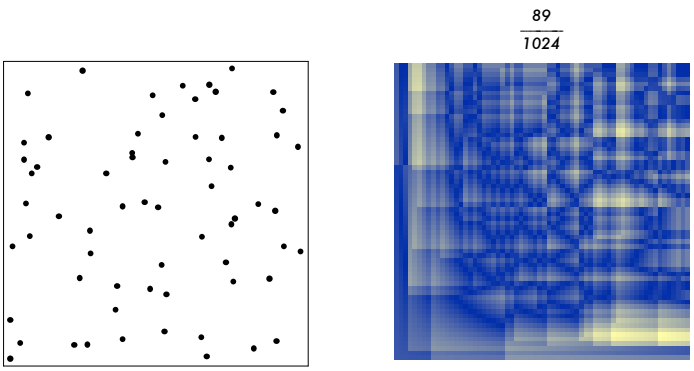

Figure 11. Local discrepancy of a sample from the Minimal Standard random number generator.

Linear congruential pseudorandom number generators have often been criticized for the underlying lattice structures they produce if all overlapping vectors are considered. Therefore, several other (nonlinear) generators have been proposed where overlapping vectors generated from such random numbers are not contained in lattices (or small unions of lattices). Thus, we consider a "baby" version and a more recent nonlinear generator, called the *explicit inversive congruential generator* (EICG), defined by Eichenauer-Herrmann [11]. The reader is asked to recover the modular generation method by means of the following *Mathematica*

implementation. From the baby generator with modulus $M = 2^7 - 1$, we produce all possible overlapping vectors (Figure 12), and from a large EICG with modulus $M = 2^{31} - 1$ and the same multiplier 117, we generate a sample of 2^7 numbers (Figure 13).

In[52]:=
```
m = 7;
p = 2^7 - 1;
f[n_] := PowerMod[117 n, p - 2, p];
gen = Table[f[n], {n, 1, 2^m + 1}];
gen = Partition[gen / (2^7 - 1), 2, 1];
g = Graphics[ {PointSize[0.02], Map[Point, gen]},
   Frame → True, AspectRatio → 1, Axes → None,
   FrameTicks → None, PlotRegion → {{0, 0.85}, {0, 0.85}}];
Show[GraphicsArray[{g, Visualize2D[gen, m]}, GraphicsSpacing → 0]]
```

From In[52]:=

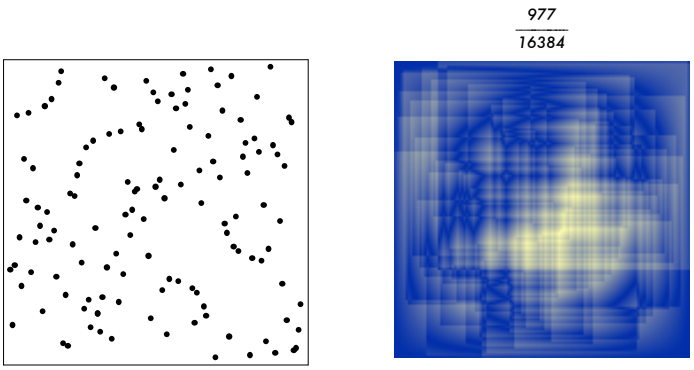

Figure 12. Local discrepancy of a baby EICG with $M = 2^7 - 1$ and $a = 117$.

```
In[59]:= m = 7;
         p = 2^31 - 1;
         f[n_] := PowerMod[117 n, p - 2, p];
         gen = Table[f[n], {n, 1, 2^m + 1}];
         gen = Partition[gen / (2^31 - 1), 2, 1];
         g = Graphics[ {PointSize[0.02], Map[Point, gen]},
            Frame → True, AspectRatio → 1, Axes → None,
            FrameTicks → None, PlotRegion → {{0, 0.85}, {0, 0.85}}];
         Show[GraphicsArray[{g, Visualize2D[gen, m]}, GraphicsSpacing → 0]]
```

From In[59]:=

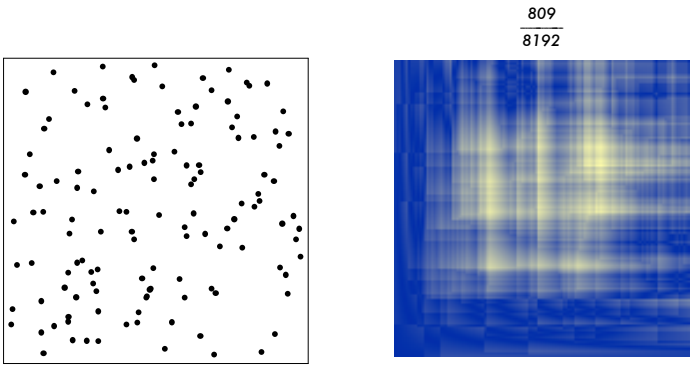

Figure 13. Local discrepancy of a small sample of an EICG with $M = 2^{31} - 1$ and $a = 117$.

■ Summary

In this article we showed some basic principles from the theory of uniform distribution of sequences modulo one using *Mathematica* functions and graphics. The functions can easily be used or extended for introductory lessons in this field. The graphical representation of the structural behavior of local discrepancy provides additional information on the behavior of discrepancy for different point sets and impressive images. These images may also be used to motivate further distribution measures such as the L^2-discrepancy.

■ Acknowledgments

This research was supported by the Austrian Science Fund (FWF) Grants S8303-MAT and S8311-MAT. The author would like to thank the editors and the anonymous referees for their comments and suggestions to improve the quality of the article.

■ References

[1] L. Kuipers and H. Niederreiter, *Uniform Distribution of Sequences*, New York: John Wiley, 1974.

[2] M. Drmota and R. F. Tichy, *Sequences, Discrepancies and Applications*, Vol. 1651 of *Lecture Notes in Mathematics*, Berlin: Springer, 1997.

[3] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*, Philadelphia: SIAM, 1992.

[4] H. Niederreiter et al. (eds.), *Monte Carlo and Quasi-Monte Carlo Methods 1996, 1998, 2000, 2002*, The series of proceedings for the conferences MC&QMC 1996–2002, Springer-Verlag.

[5] P. Hellekalek and G. Larcher, *Random and Quasi-Random Point Sets*, Vol. 138 of *Lecture Notes in Statistics*, Berlin: Springer, 1998.

[6] MathConsult, "*MathDirect* Mathematical Software." (Jan 22, 2000) www.mathdirect.com/products.

[7] K. Entacher, "Discrepancy Estimates Based on Haar Functions," *Mathematics and Computers in Simulation*, **55**, 2001 pp. 49–57.

[8] D. E. Knuth, *The Art of Computer Programming,* Vol. 2, *Seminumerical Algorithms*, 3rd ed., Boston: Addison-Wesley, 1997.

[9] S. K. Park and K. W. Miller, "Random Number Generators: Good Ones Are Hard to Find," *Communications of the Association for Computing Machinery*, **31**, 1988 pp. 1192–1201.

[10] B. Hechenleitner and K. Entacher, "*The Spectral Test Server*," (Sep 1, 2004) spectral.fh-sbg.ac.at.

[11] J. Eichenauer-Herrmann, "Statistical Independence of a New Class of Inversive Congruential Pseudorandom Numbers," *Mathematics of Computation*, **60**, 1993 pp. 375–384.

About the Author

Karl Entacher received his B. S., M. S., and Ph.D. degrees in mathematics from the University of Salzburg and completed his Habilitation on Scientific Computing at the same university. He currently holds a faculty position in the School of Telecommunications Engineering at Salzburg University of Applied Sciences and Technologies. His research interests include simulation, Monte Carlo and quasi-Monte Carlo methods, random number generation, parallel random numbers, and low-discrepancy point sets.

Karl Entacher
School of Telecommunications Engineering
Salzburg University of Applied Sciences & Technologies
Schillerstrasse 30
A-5020 Salzburg
Austria
karl.entacher@fh-sbg.ac.at