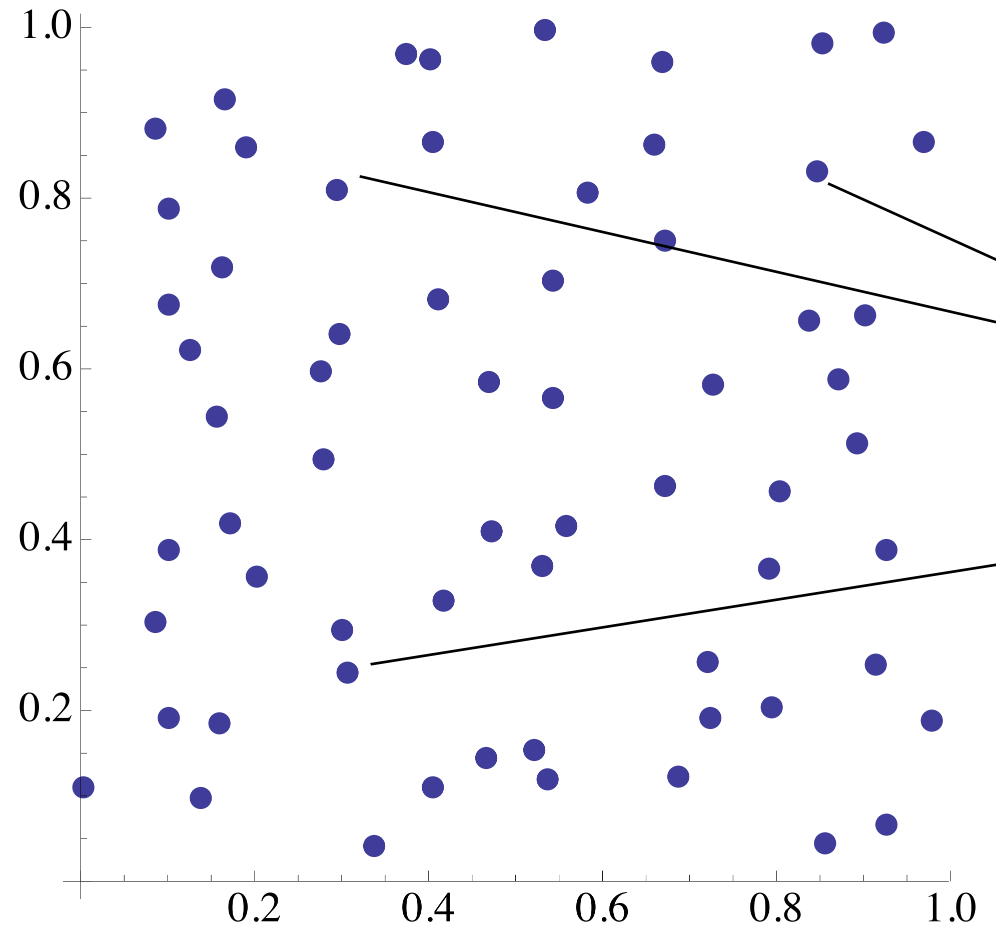
Monte Carlo 2

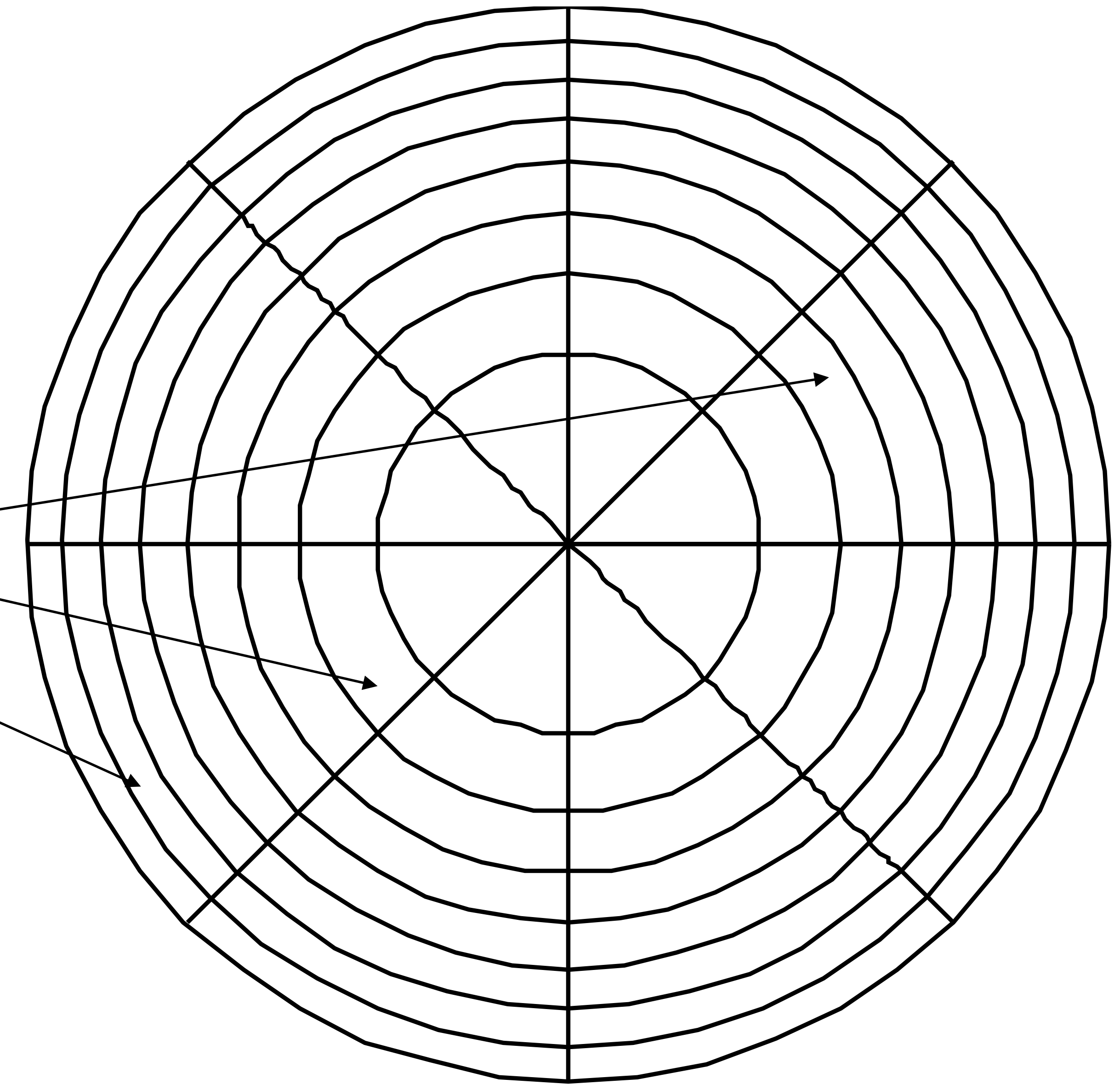
Today

- Discrepancy and Quasi-Monte Carlo (QMC)
- Low-discrepancy constructions
  - Halton, Hammersley, Sobol’
- Randomized low-discrepancy: RQMC
- Spectral analysis of sampling patterns and MC

Warping Samples For MC Integration



$$\xi_i \in [0,1)^2$$

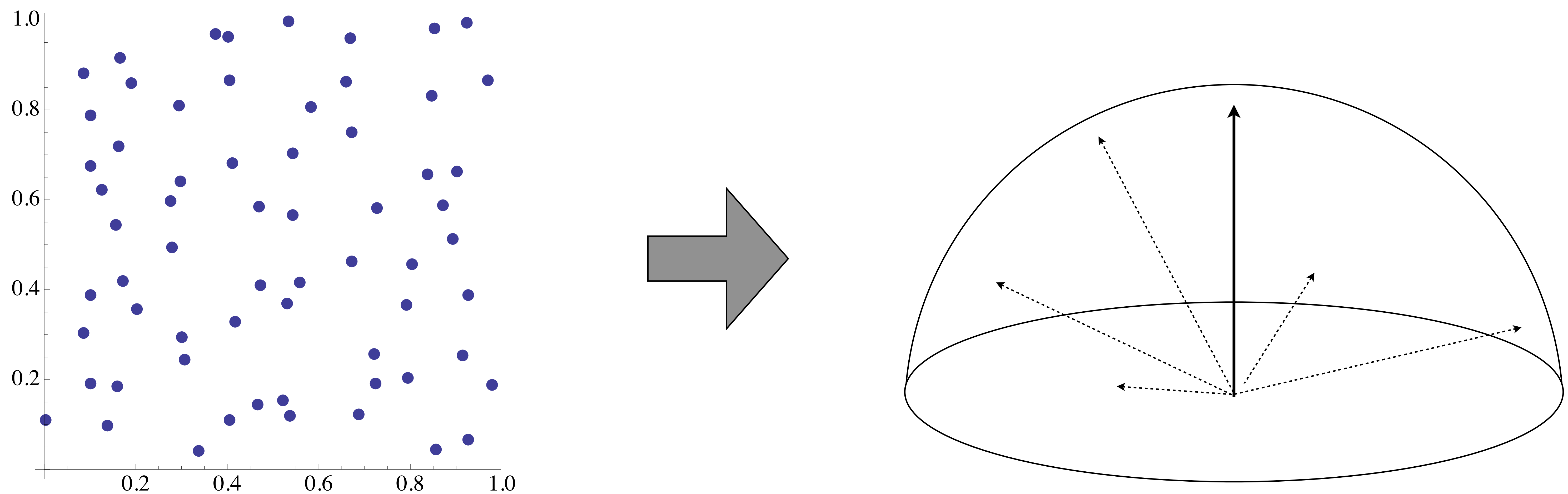


$$\theta = 2\pi\xi_1$$

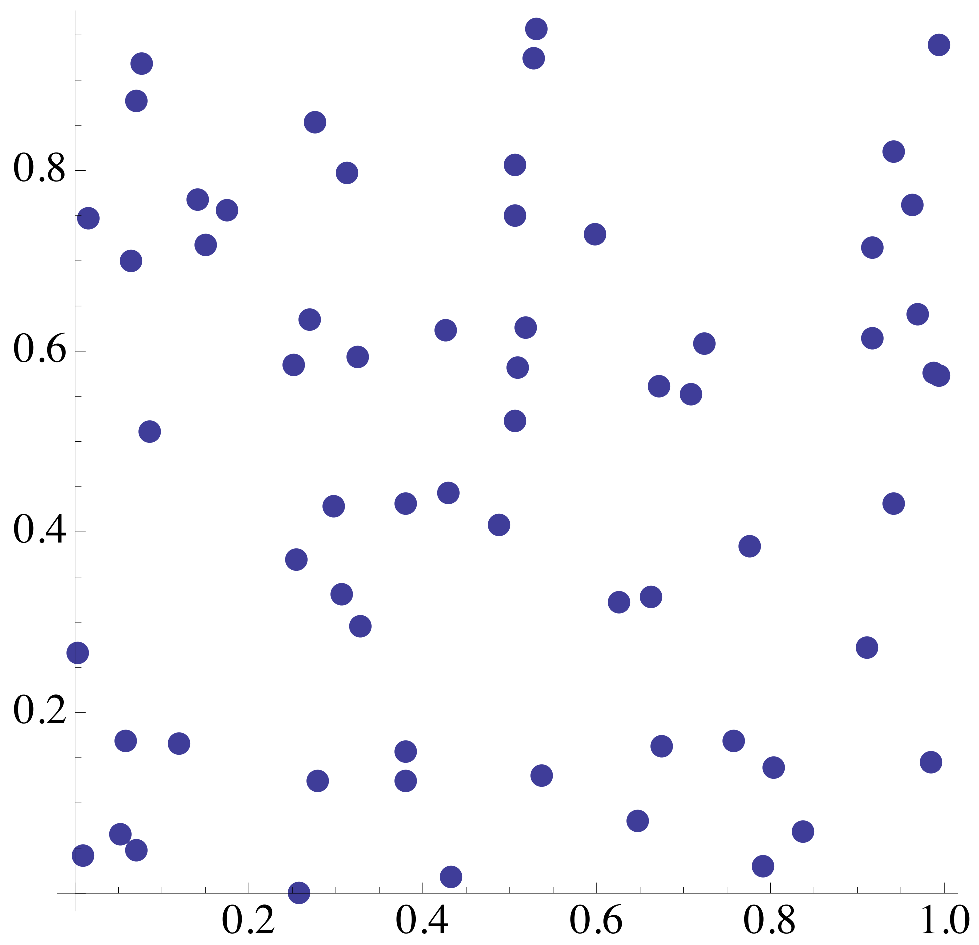
$$r = \sqrt{\xi_2}$$

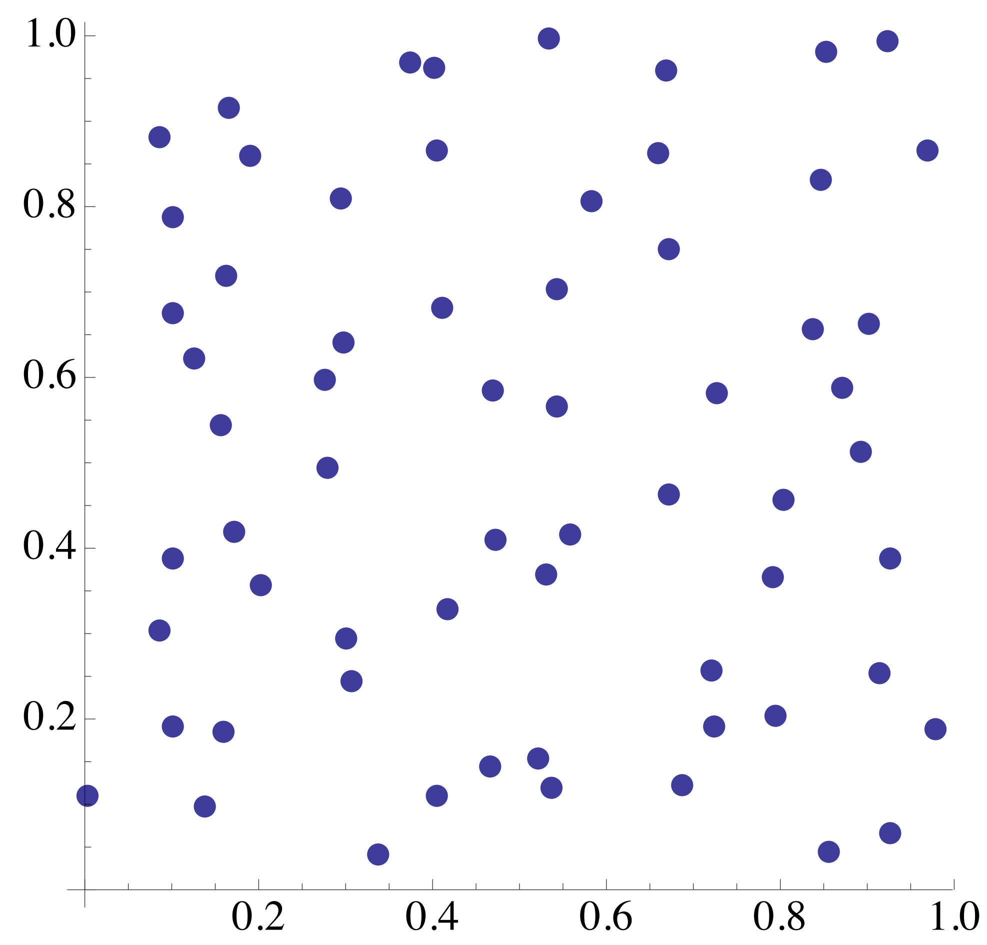
Warping Samples For MC Integration

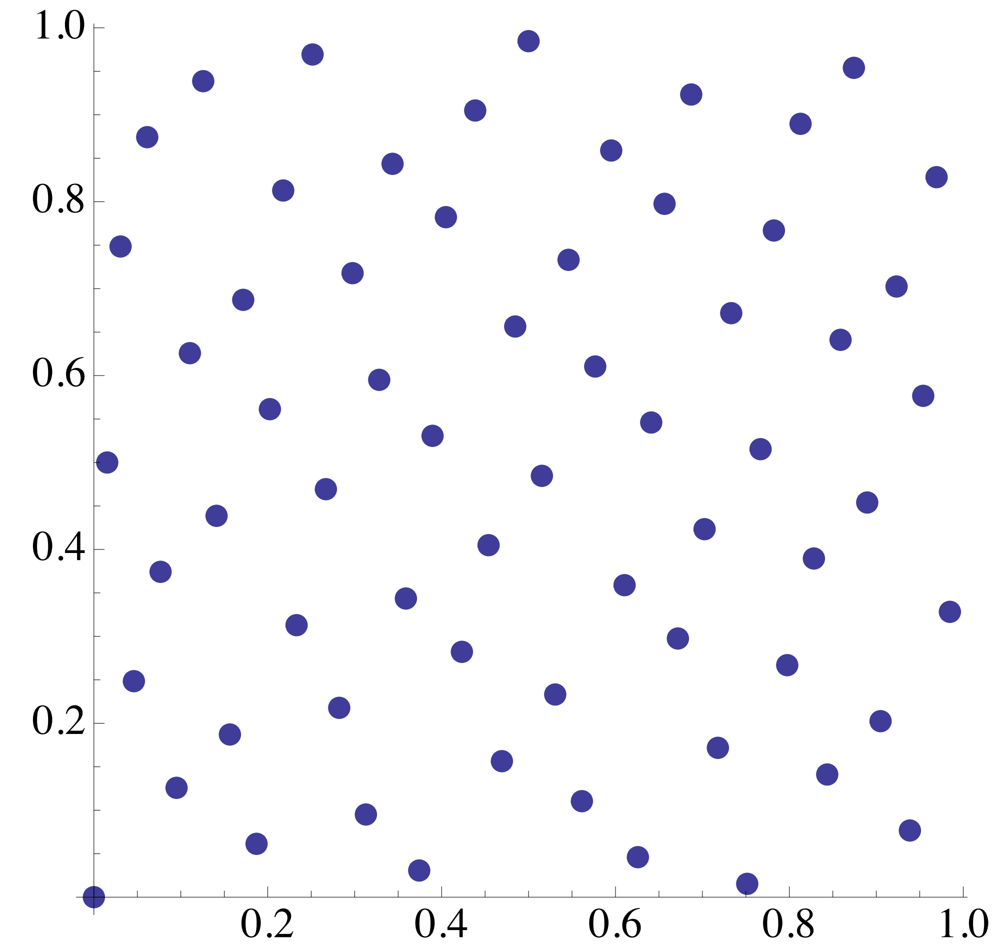
Cosine-weighted hemisphere sampling:



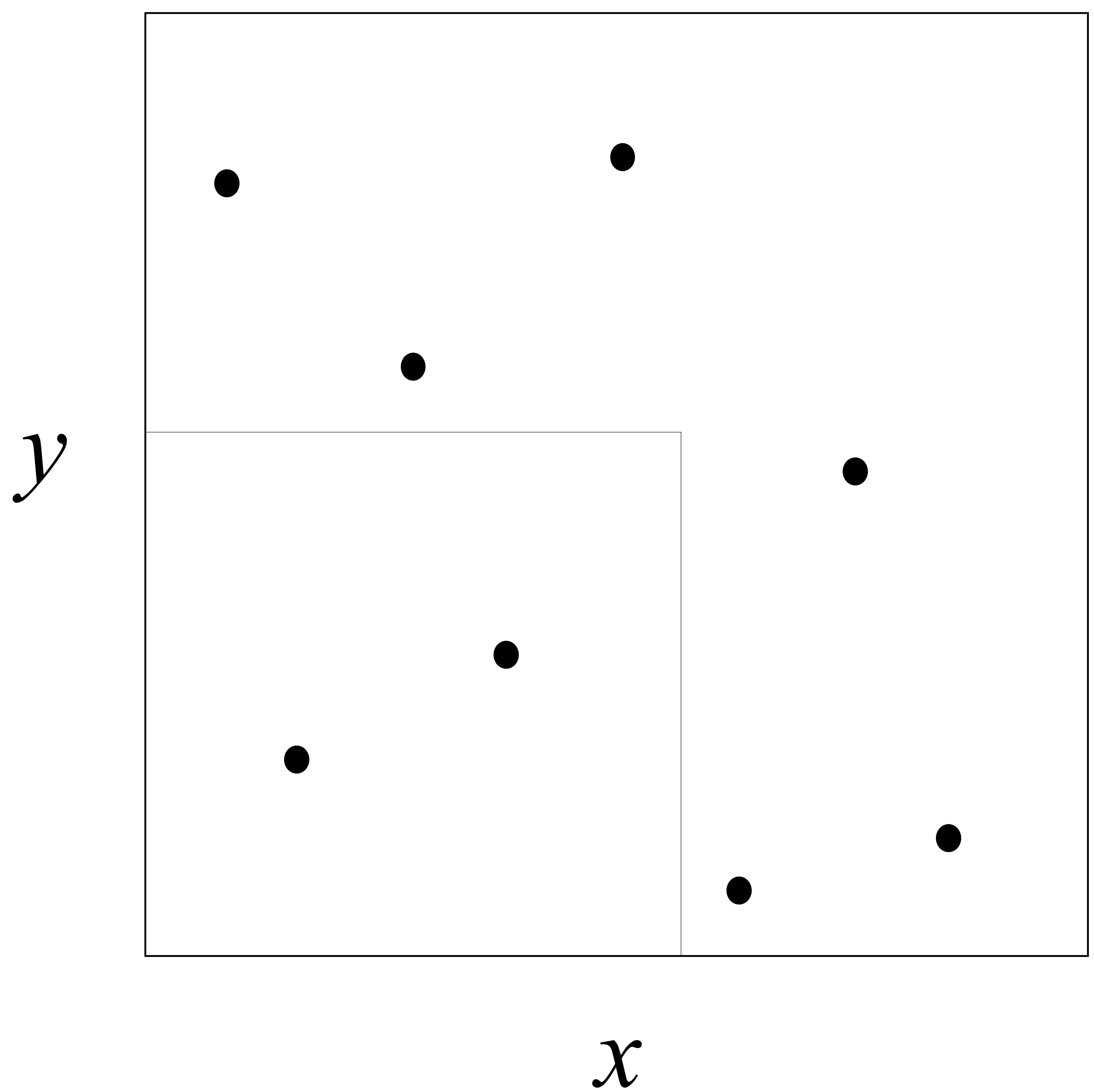
Three 2D Point Sets







Point Set Evaluation: Discrepancy



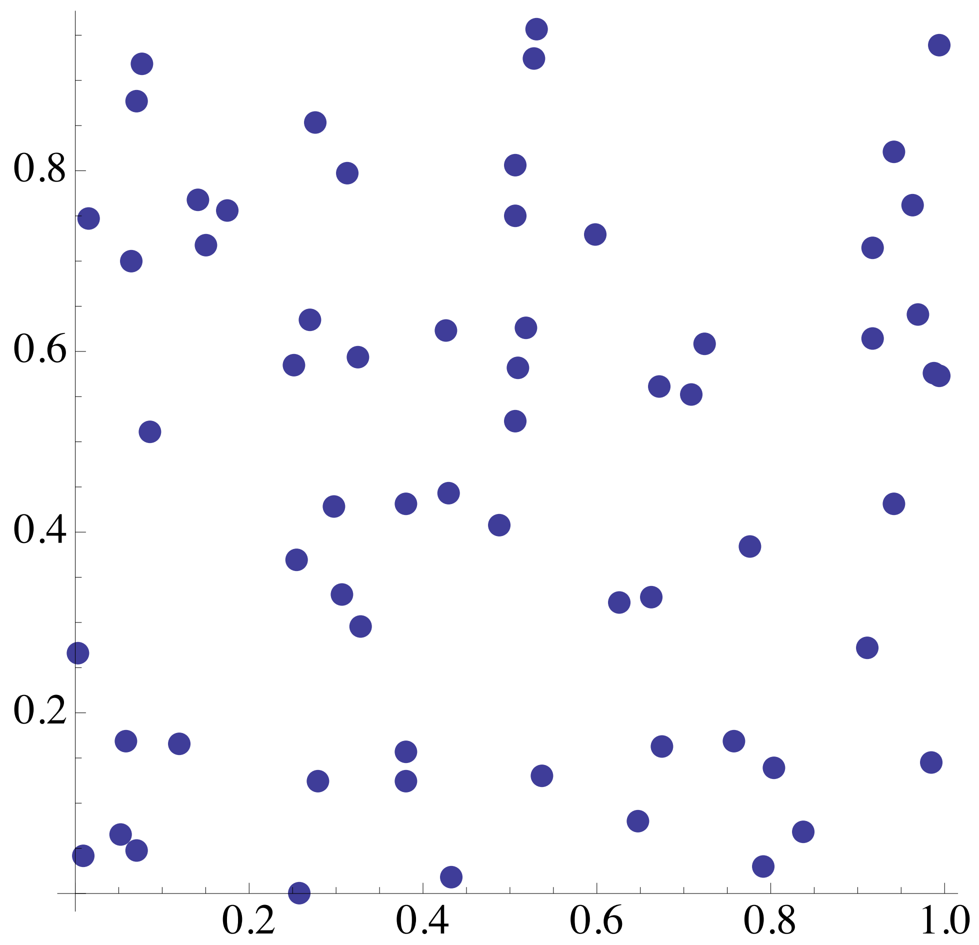
$$\Delta(x,y) = \frac{n(x,y)}{N} - xy$$

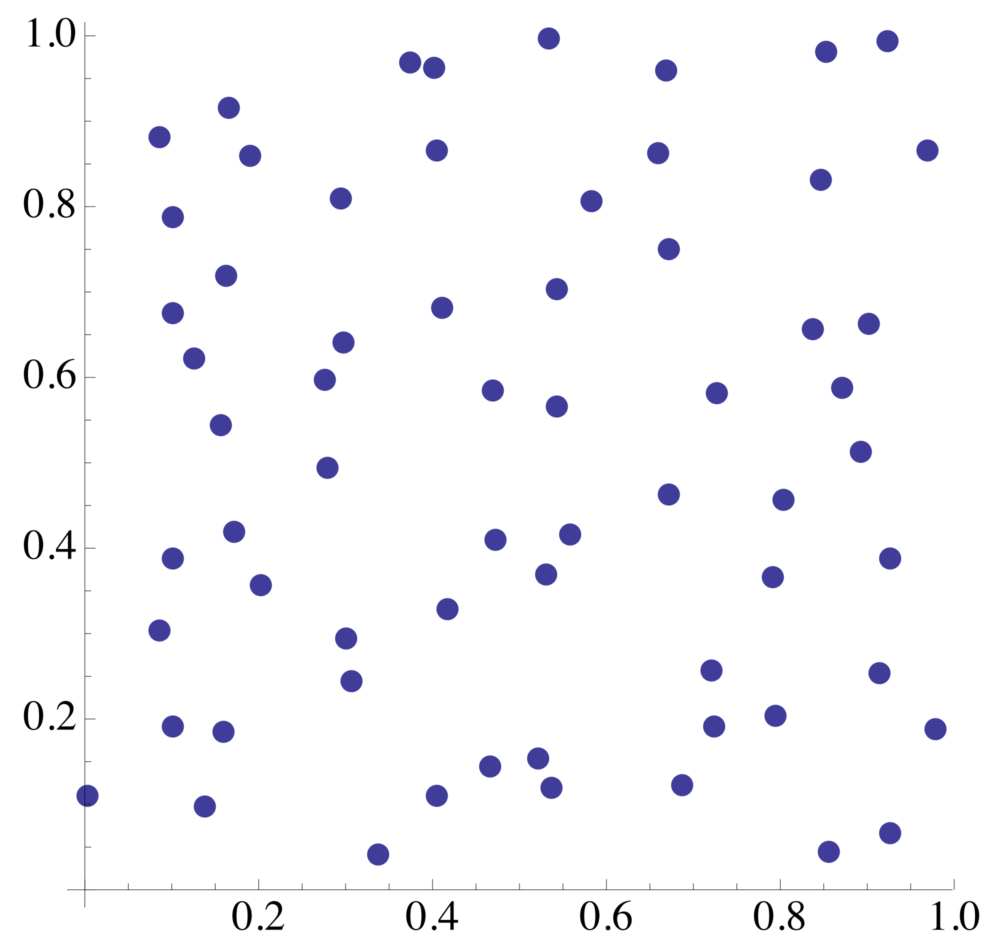
$A = xy$

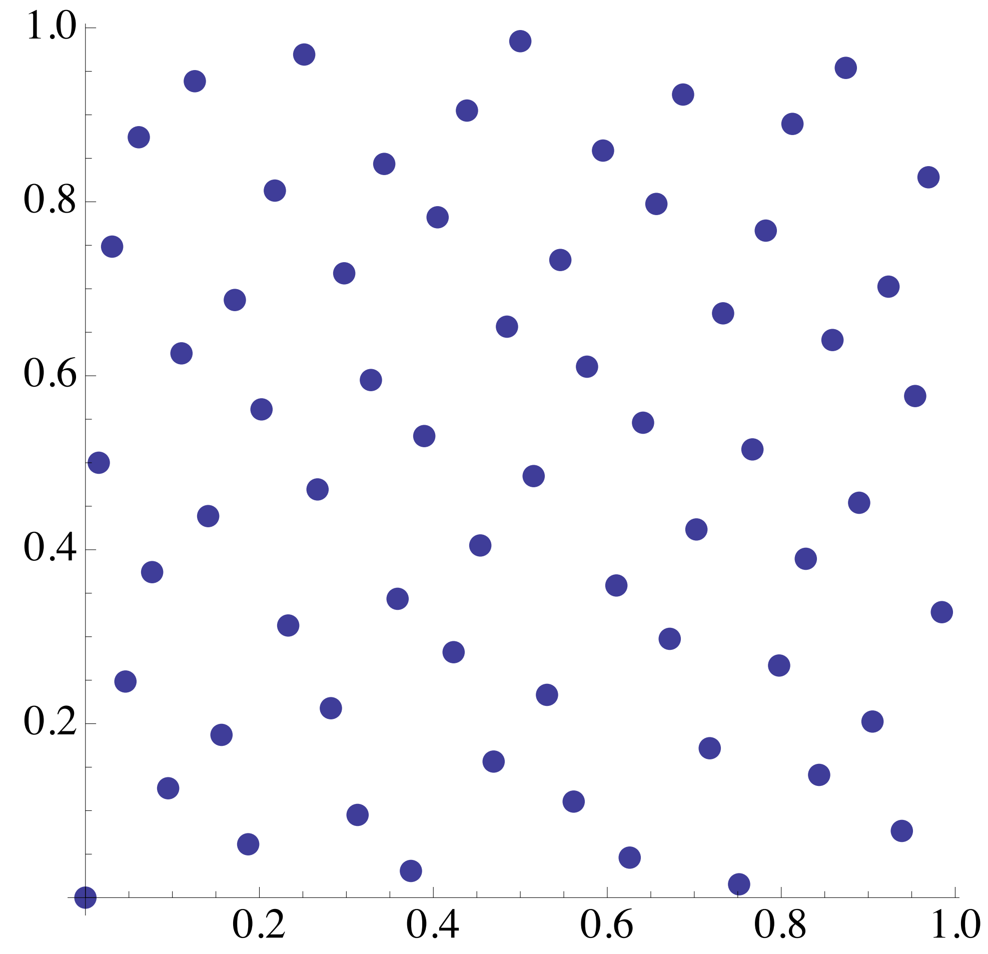
$n(x,y)$ number of samples in A

$$D_N = \max_{x,y} \left| \Delta(x,y) \right|$$

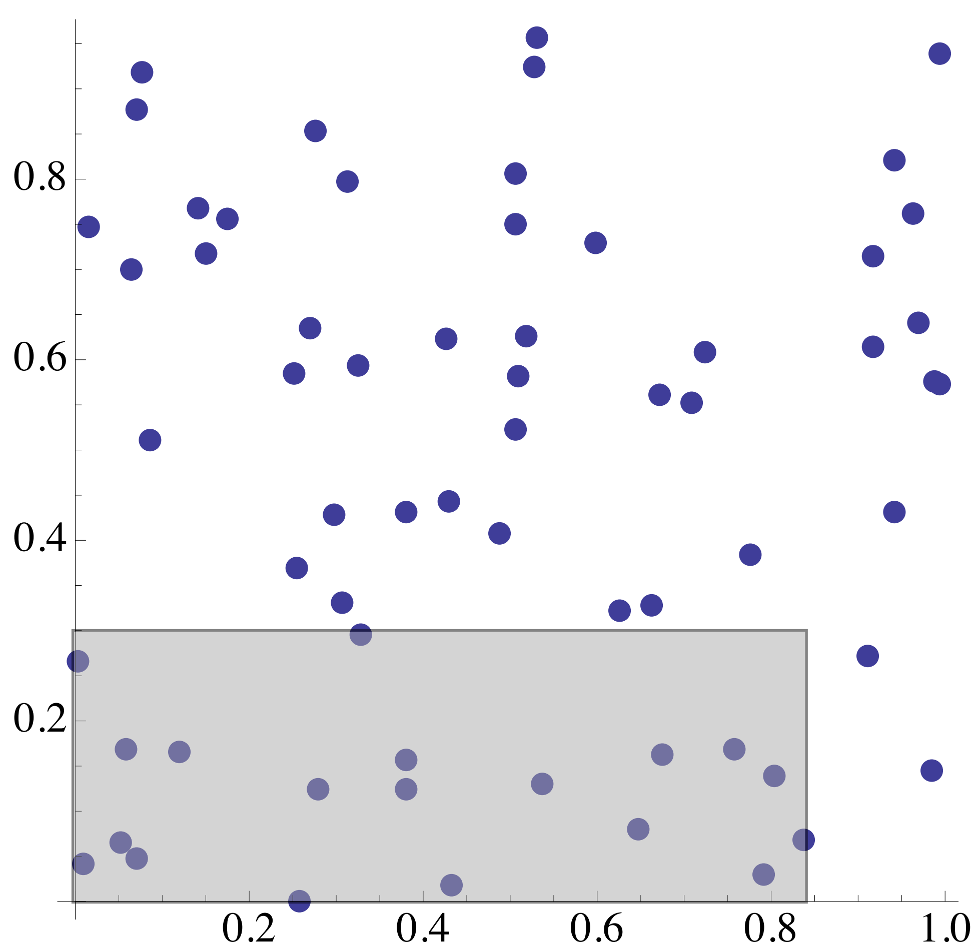
Three 2D Point Sets





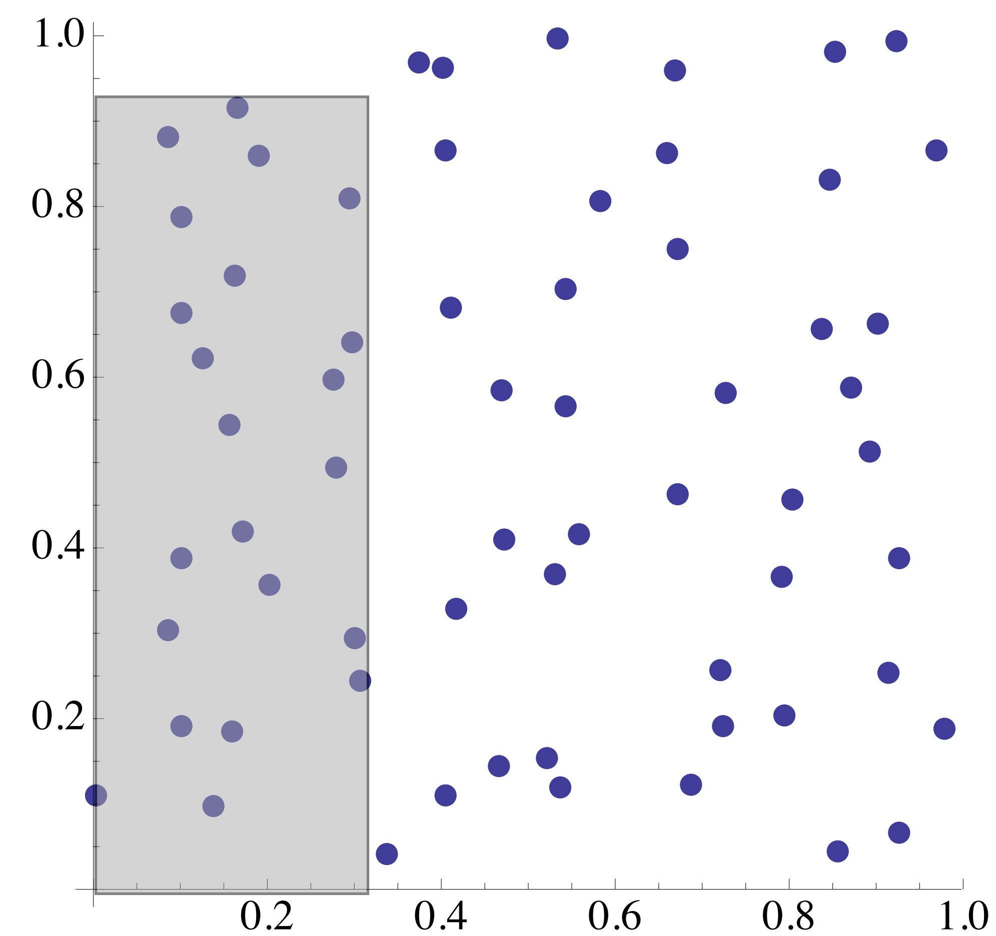


Discrepancy (Empirical)



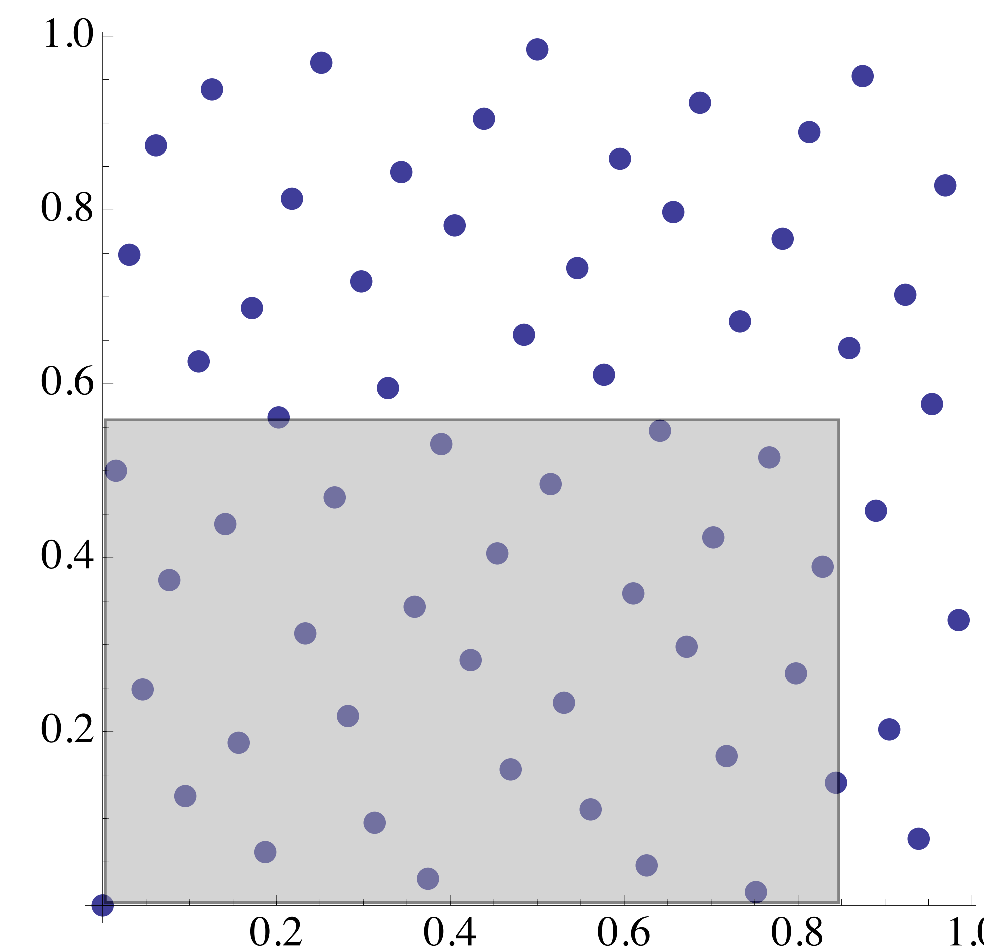
Independent

0.148



Stratified

0.081



Larscher-Pillichshammer

0.041

Low-Discrepancy Definition

An (infinite) *sequence* of n samples in dimension d is low discrepancy if:

$$D_n = O\left(\frac{(\log n)^d}{n}\right)$$

A (finite) *set* of n samples in dimension d is low discrepancy if:

$$D_n = O\left(\frac{(\log n)^{d-1}}{n}\right)$$

Theorem on Total Variation

Koksma-Hlawka inequality:

$$\left| \frac{1}{N} \sum_{i=0}^{N-1} f(X_i) - \int f(x)dx \right| \leq V(f) D_N$$

$$V(f) = \int \left| \frac{\delta f}{\delta x} \right| dx$$

Quasi-Monte Carlo Error Bounds

Although error is bounded as $|e| \leq V(f) D_N$ **not a tight bound!**

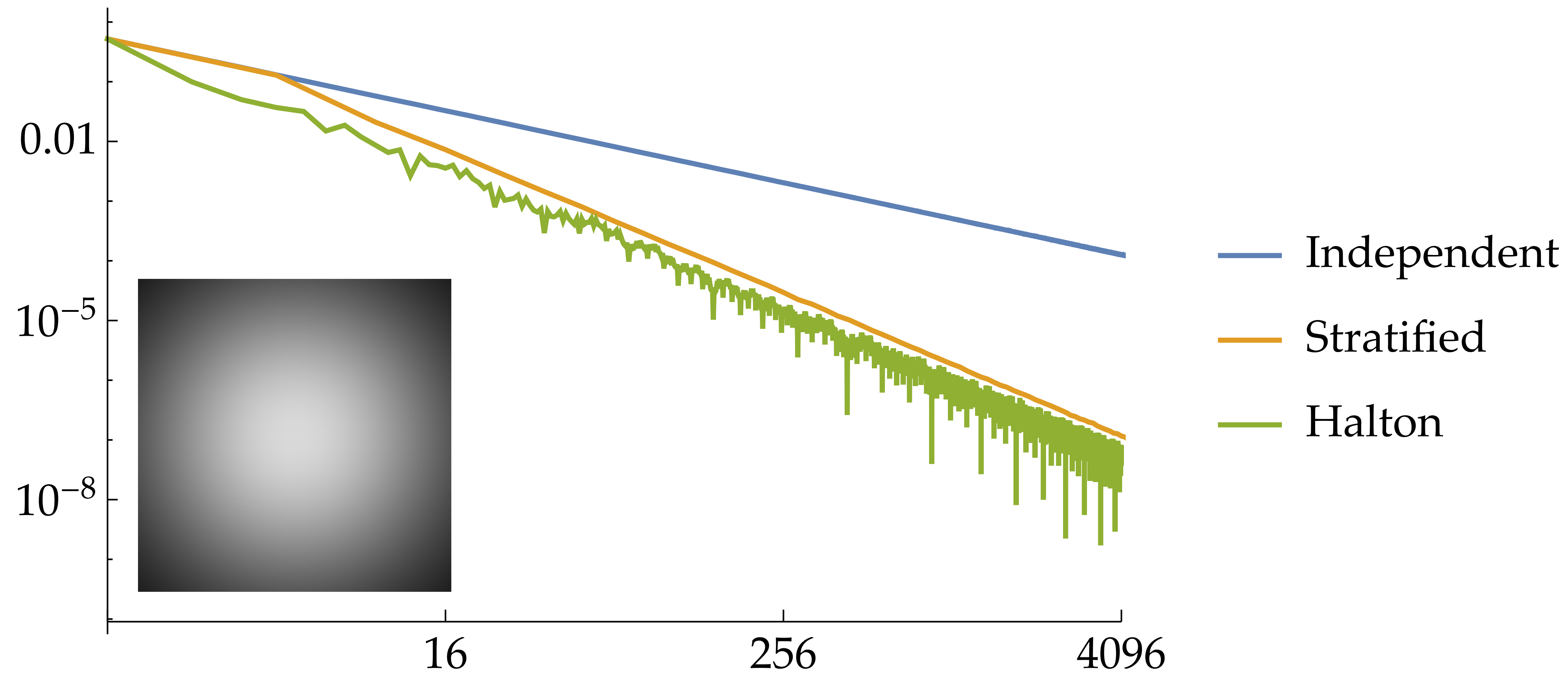
Even worse, $V(f)$ **is sometimes unbounded**

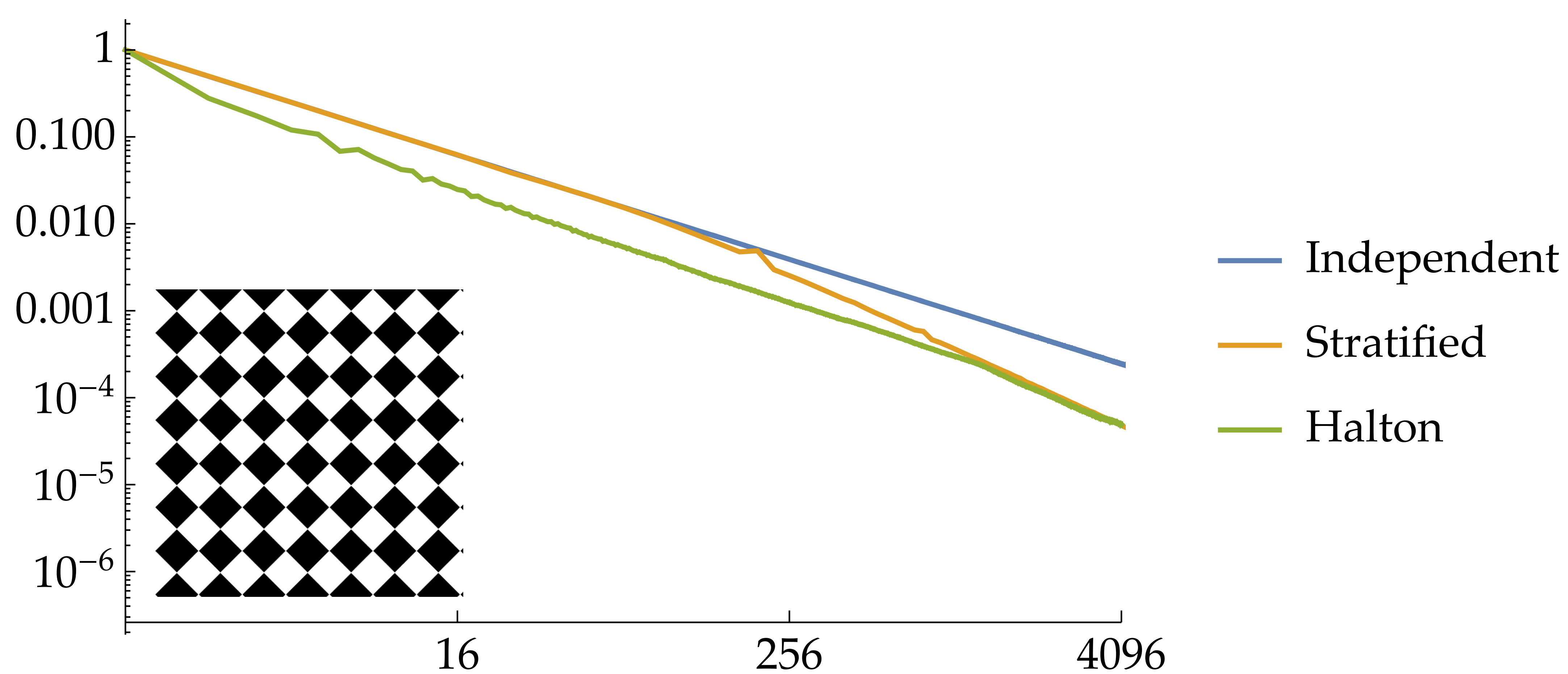
We can use this inequality to show that QMC error converges as:

$$\sim \frac{(log\ N)^d}{N}$$

(recall that MC variance goes at $O(1/N)$**, so error goes at** $O(1/\sqrt{N})$**.)**

Measured Quasi-Monte Carlo Error





Low-Discrepancy Point Sequences

The Radical Inverse

Consider the digits of a number n, expressed in base b

$$n = \sum_{i=1}^{\infty} d_i \, b^{(i-1)}$$

e.g. for n = 6 in base 2, n=110_2, and

$$d_1 = 0, d_2 = 1, d_3 = 1, d_i = 0$$

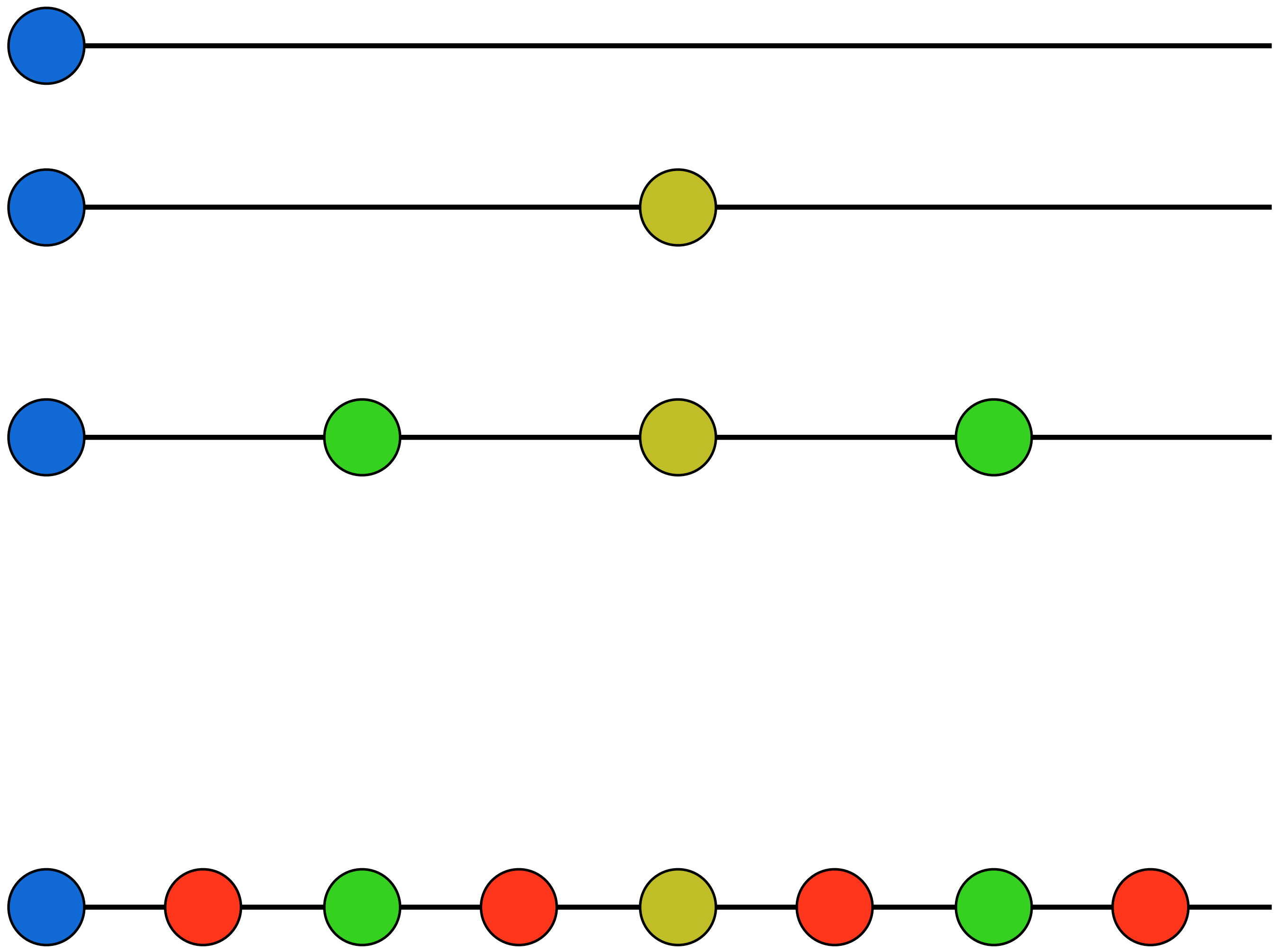
The radical inverse mirrors the digits around the decimal:

$$\Phi_2(6) = 0.011_2 = 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} = 0.375$$

$$\Phi_b(n) = \sum_{i=1}^{\infty} d_i \, b^{-i}$$

1D Low Discrepancy: van der Corput

n	$\Phi_2(n)$
0	0
1	0.5
2	0.25
3	0.75
4	0.125
5	0.625
6	0.375
7	0.875
...	...



Efficient Base 2 Radical Inverse

Assume a fixed number of bits (say 32):

$$\Phi_b(n) = \sum_{i=1}^{32} d_i \, b^{-i}$$

We have the sum: $d_1 \, 2^{-1} + d_2 \, 2^{-2} + \cdots + d_{32} \, 2^{-32}$

Pull out a factor of 2^{-32}**:** $2^{-32}(d_1 \, 2^{31} + d_2 \, 2^{30} + \cdots + d_{32})$

Can also express in terms of bit shifts:

$$2^{-32}((d_1 << 31) + (d_2 << 30) + \cdots + d_{32})$$

Efficient Base 2 Radical Inverse

$$2^{-32}((d_1 << 31) + (d_2 << 30) + \cdots + d_{32})$$

We already have the digits in the bits of n

$$n = \sum_{i=1}^{\infty} d_i \, b^{(i-1)}$$

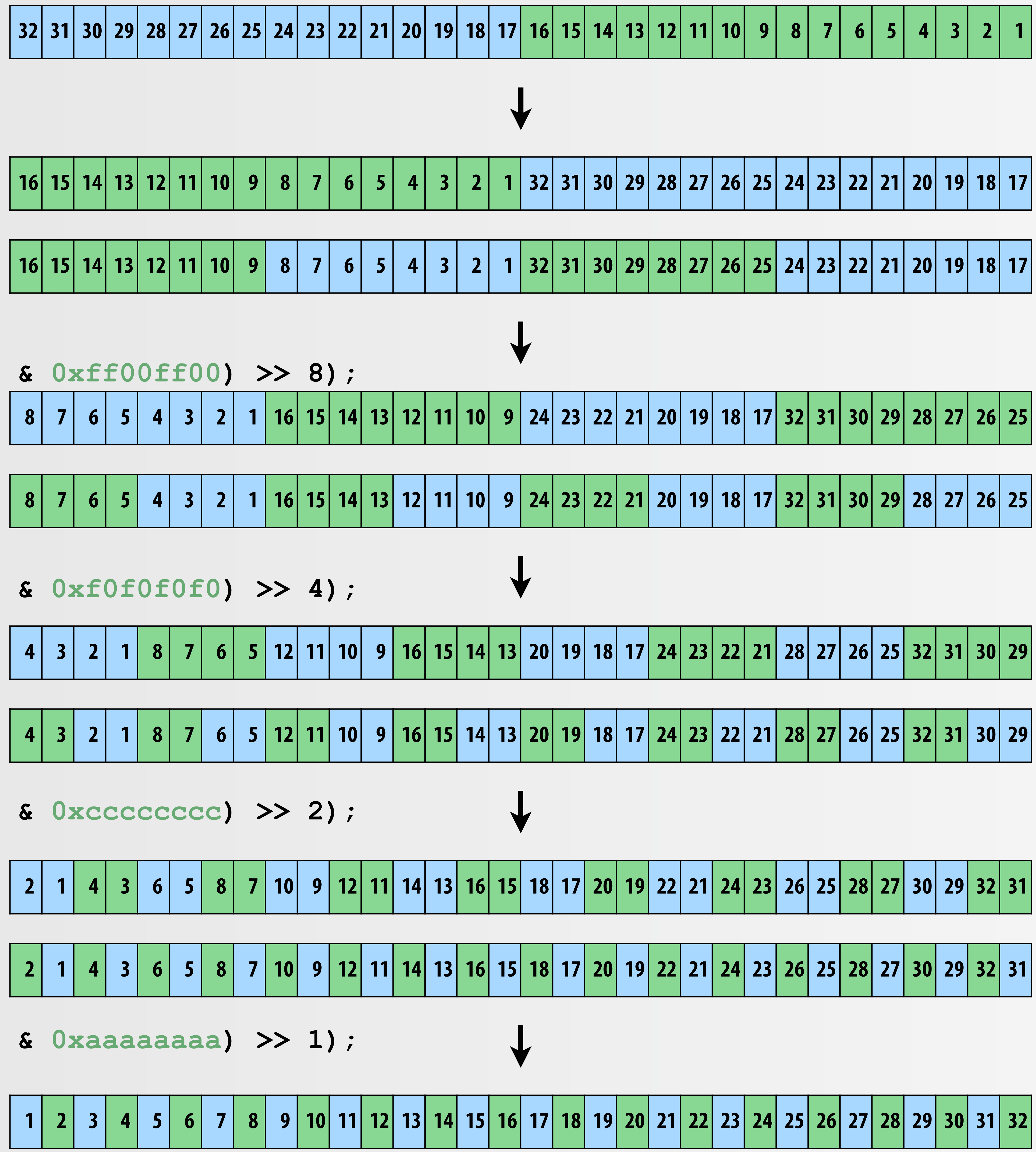
32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

So

- **Reverse the bits**
- **Multiply by** 2^{-32}

Reversing Bits in Parallel

```
uint32_t ReverseBits(uint32_t n) {

    n = (n << 16) | (n >> 16);

    n = ((n & 0x00ff00ff) << 8) | ((n & 0xff00ff00) >> 8);

    n = ((n & 0x0f0f0f0f) << 4) | ((n & 0xf0f0f0f0) >> 4);

    n = ((n & 0x33333333) << 2) | ((n & 0xcccccccc) >> 2);

    n = ((n & 0x55555555) << 1) | ((n & 0xaaaaaaaa) >> 1);
    return n;
}
```



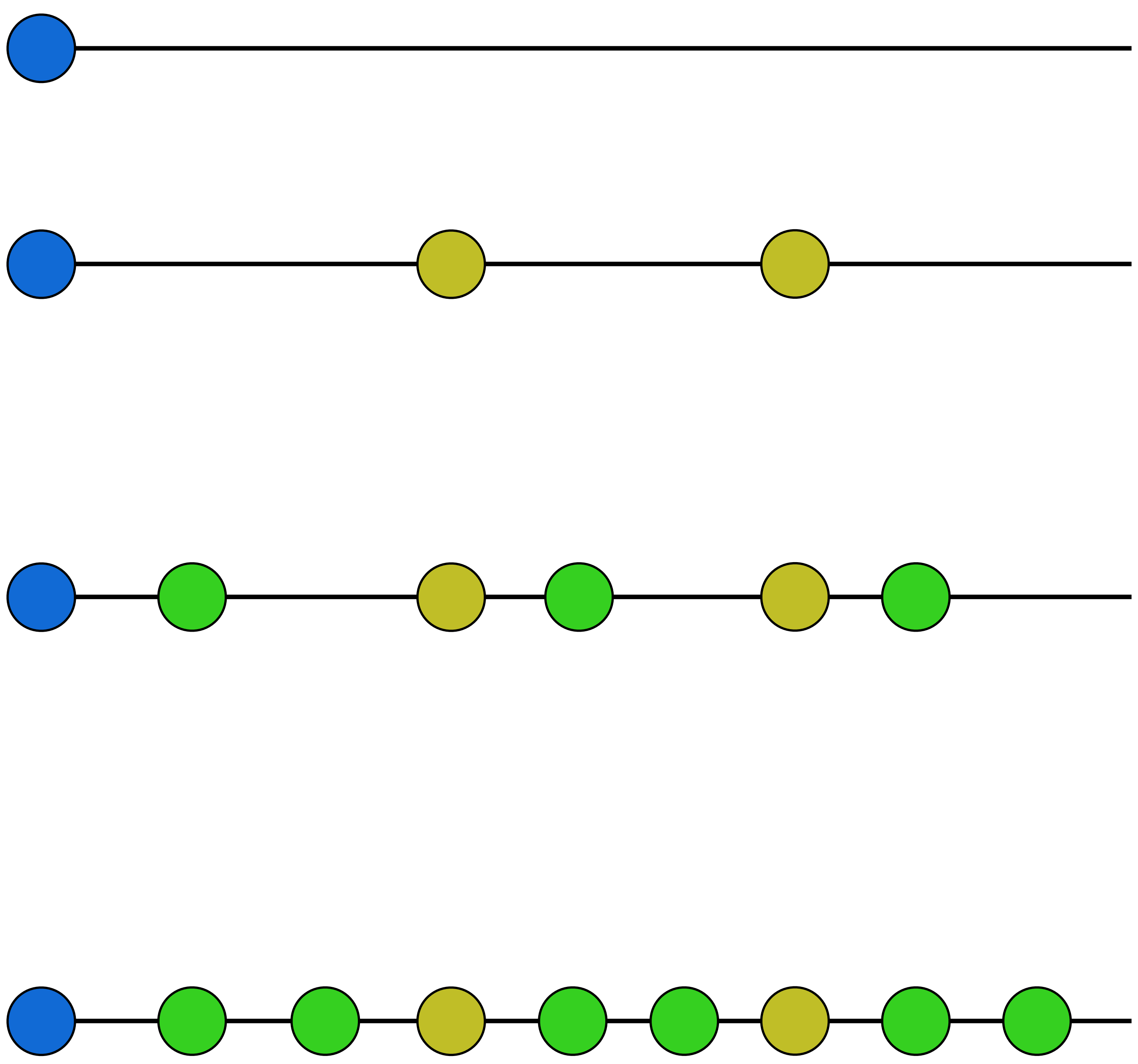
Efficient van Der Corput

```
uint32_t ReverseBits(uint32_t n) {
    n = (n << 16) | (n >> 16);
    n = ((n & 0x00ff00ff) << 8) | ((n & 0xff00ff00) >> 8);
    n = ((n & 0x0f0f0f0f) << 4) | ((n & 0xf0f0f0f0) >> 4);
    n = ((n & 0x33333333) << 2) | ((n & 0xcccccccc) >> 2);
    n = ((n & 0x55555555) << 1) | ((n & 0xaaaaaaaa) >> 1);
    return n;
}
```

```
float RadicalInverse2(uint32_t v) {
  v = ReverseBits(v);
  const float Inv2To32 = 1.f / (1ull << 32);
  return v * Inv2To32;
}
```

Radical Inverse Base 3

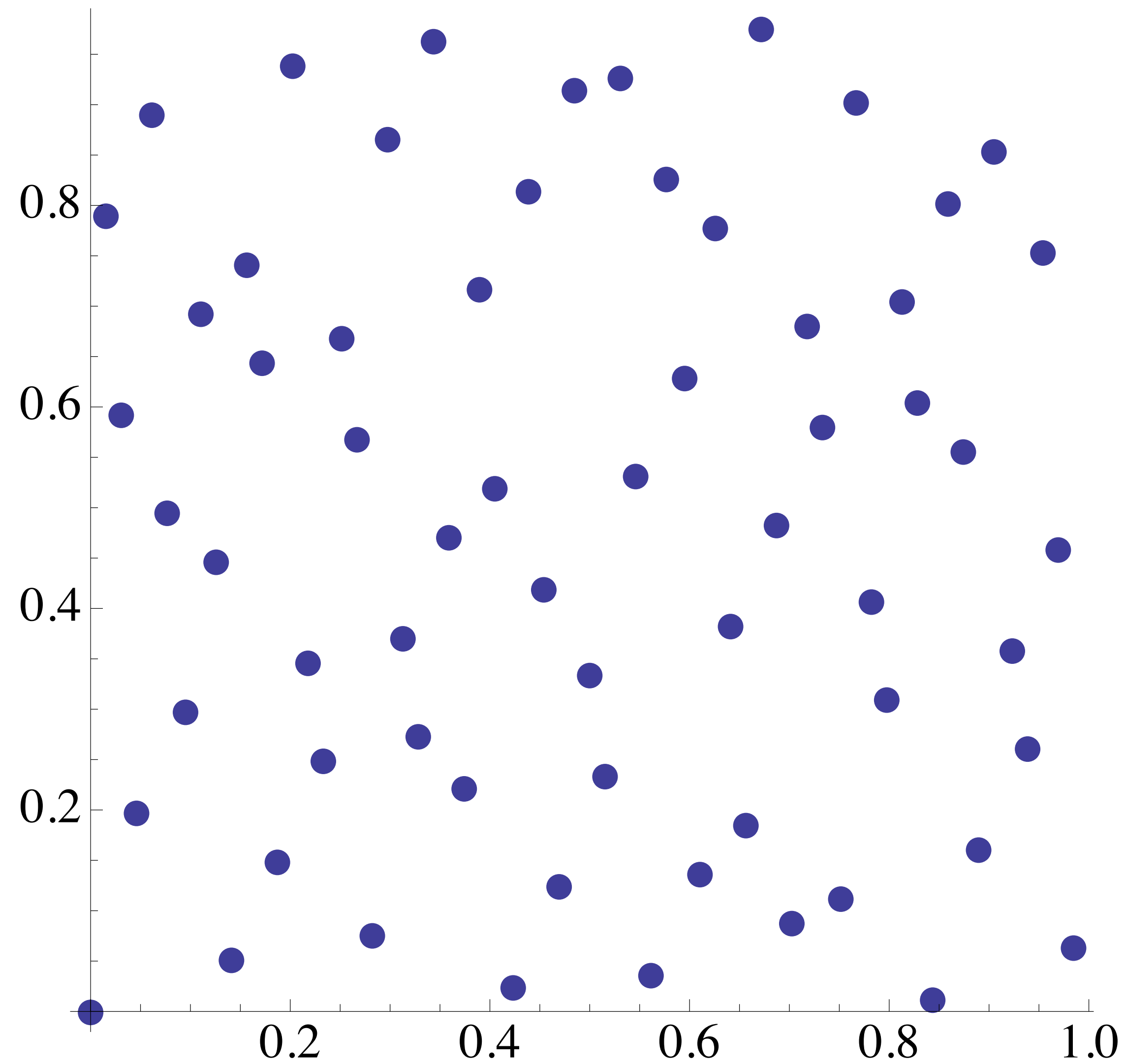
n	$\Phi_3(n)$
0	0
1	0.333..
2	0.666...
3	0.111...
4	0.444...
5	0.777...
6	0.222...
7	0.555...
8	0.888...



The Halton Sequence

Low discrepancy sequence $(\Phi_{b_1}(n), \Phi_{b_2}(n), \Phi_{b_3}(n), \ldots)$

- **The dimensions' bases are relatively prime**
- **Arbitrary number of dimensions**
- **Arbitrary number of points**

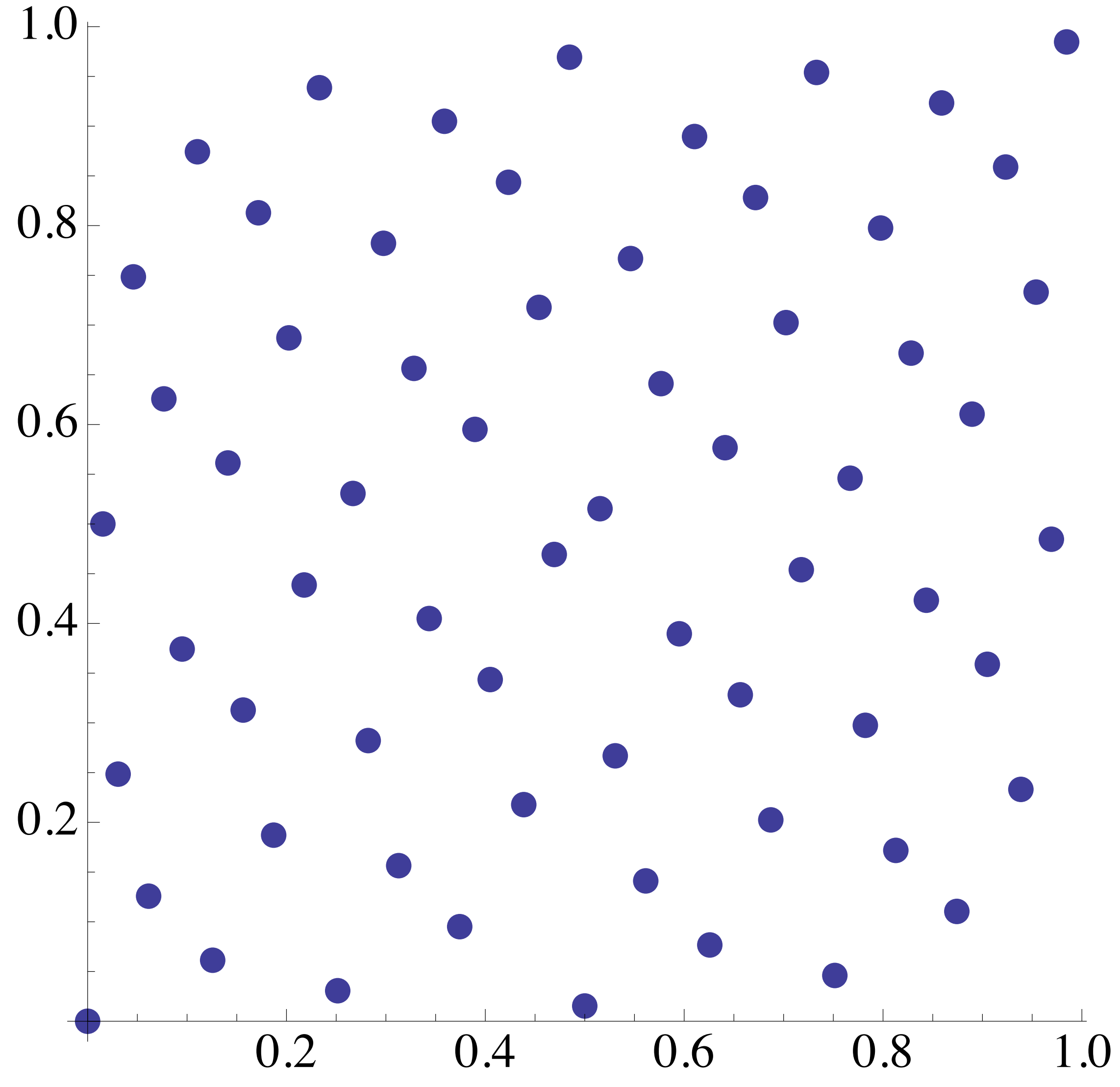


$$(\Phi_2(n), \Phi_3(n))$$

The Hammersley Point Set

If the number of points N is known in advance, set one dimension to n/N

$$(n/N, \Phi_{b_1}(n), \Phi_{b_2}(n), \ldots)$$



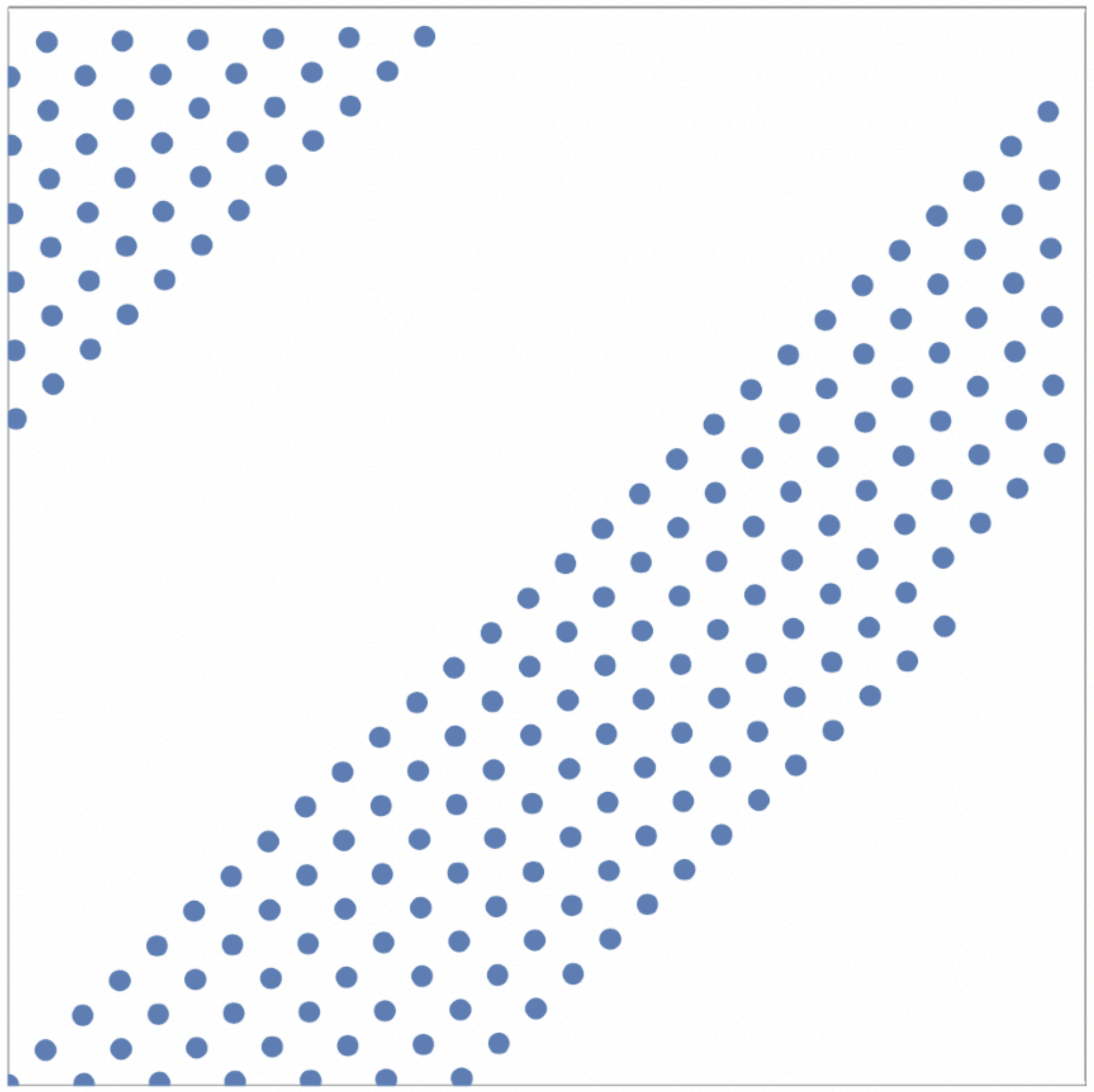
$$(n/N, \Phi_2(n))$$

Slightly lower discrepancy than Halton

Low-Dimensional Projections...

Caution: 2D projections of higher bases may not be great

- The overall pattern remains low-discrepancy over all dimensions, though



$(\Phi_{29}(n), \Phi_{31}(n))$

Randomized Low Discrepancy

$$\Phi_b(n) = \sum_{i=1}^{\infty} d_i \, b^{-i}$$

Radical Inverse

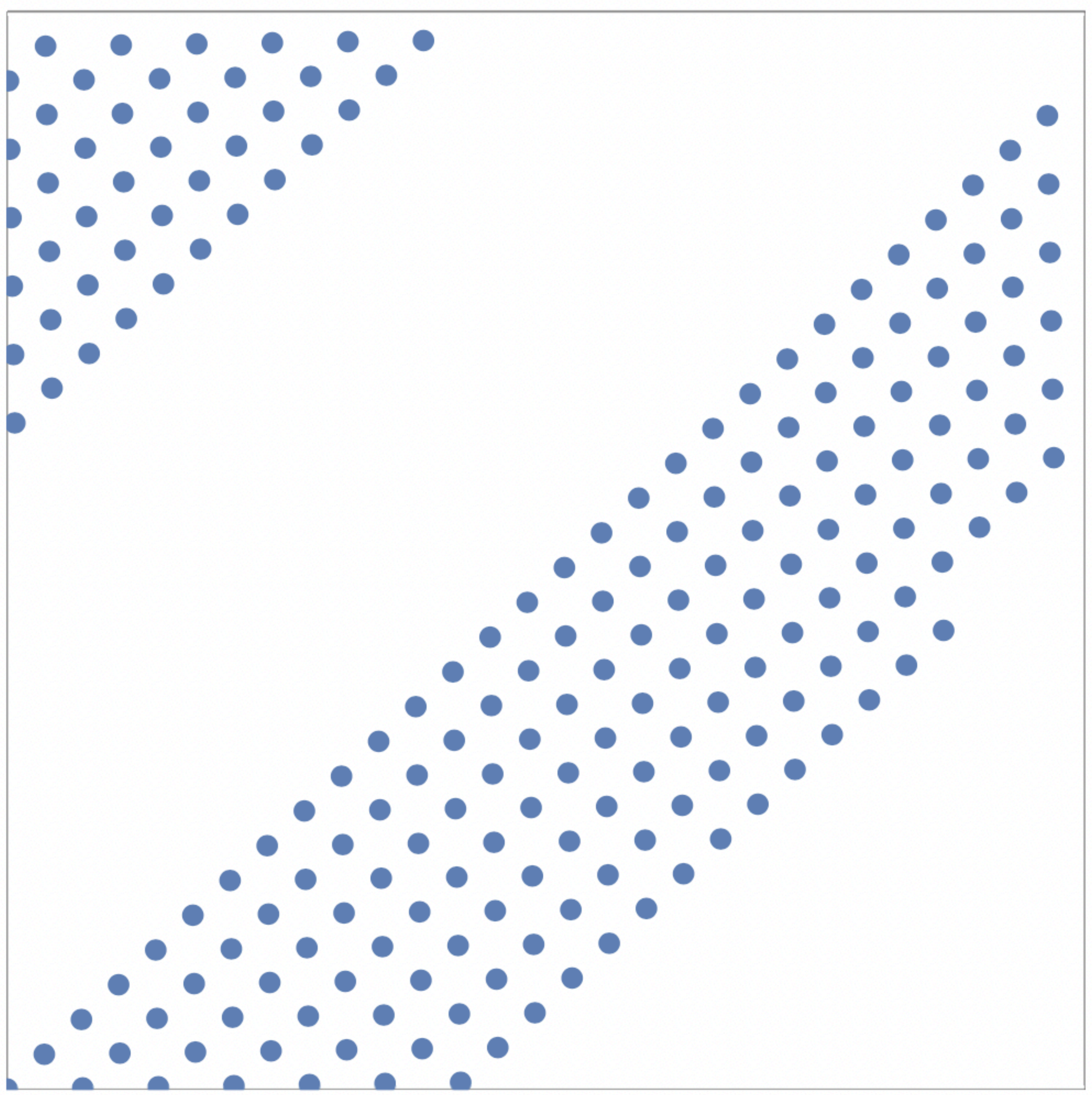
$$\Phi'_b(n) = \sum_{i=1}^{\infty} \sigma_i(d_i) b^{-i}$$

Permuted Radical Inverse

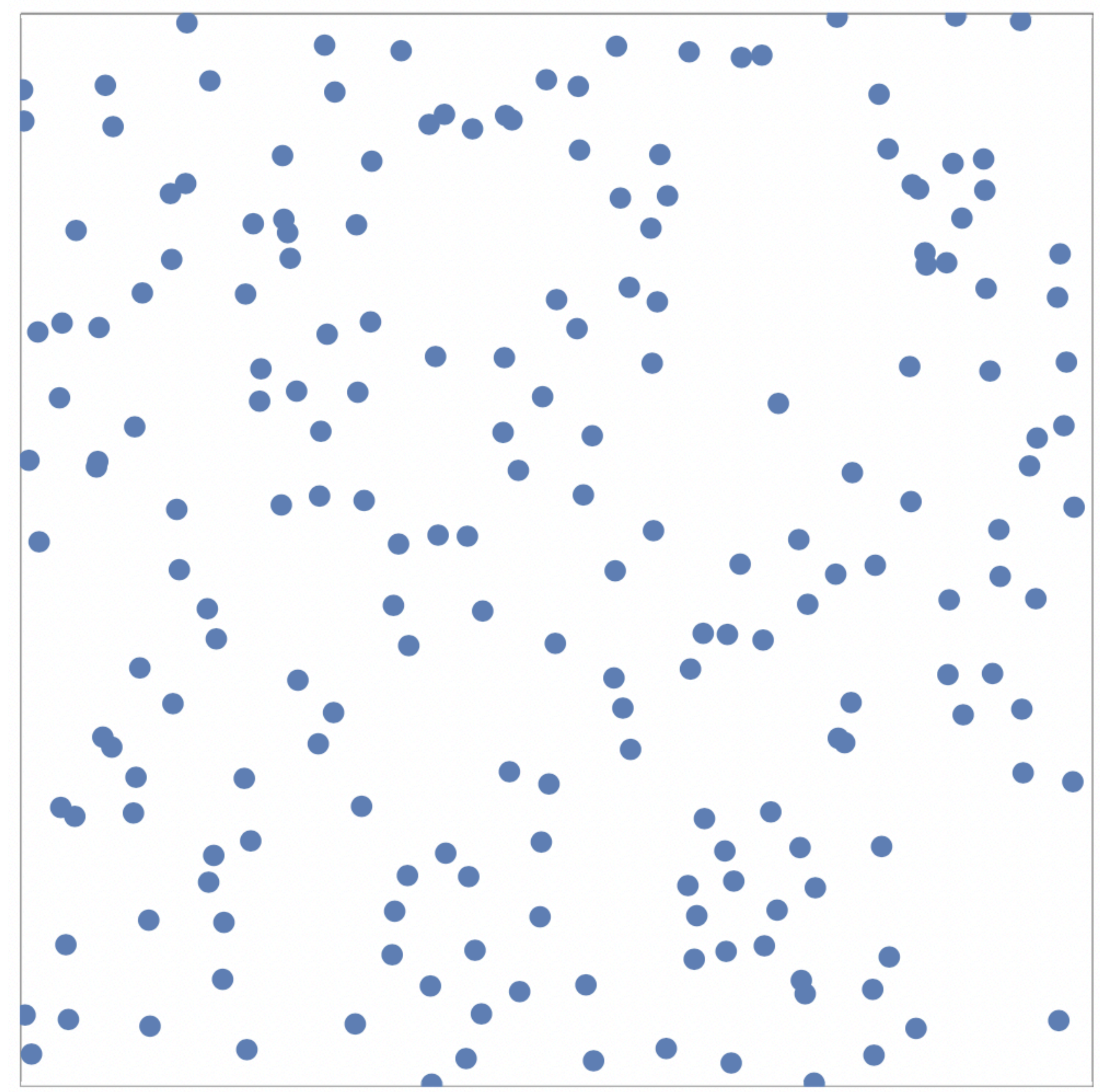
σ_i **are (random) permutations of the digits**

- **Random permutations maintain LD**

Halton + Random Digit Permutations



Unscrambled



Scrambled

$$(\Phi_{29}(n), \Phi_{31}(n))$$

Owen Scrambling

Apply random digit permutations that depend on previous digits

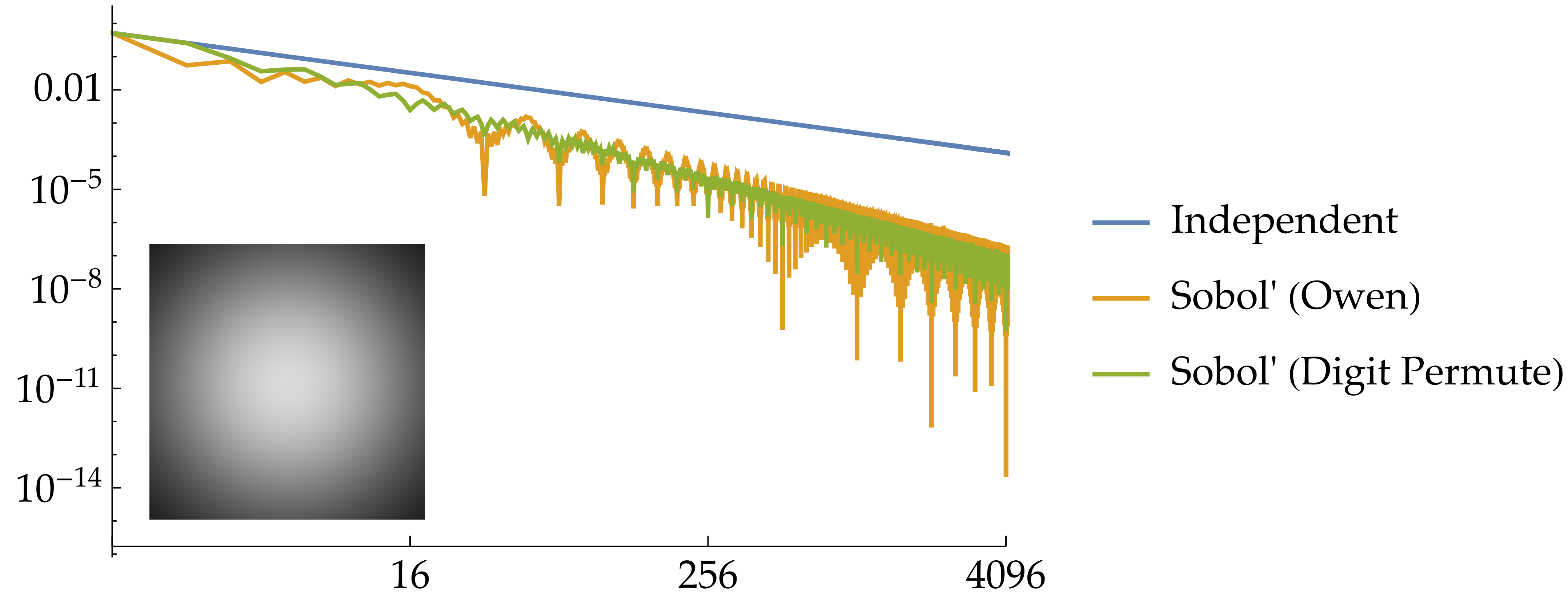
$$\Phi_b'(n) = \sum_{i=1}^{\infty} \sigma_i(d_i) b^{-i}$$

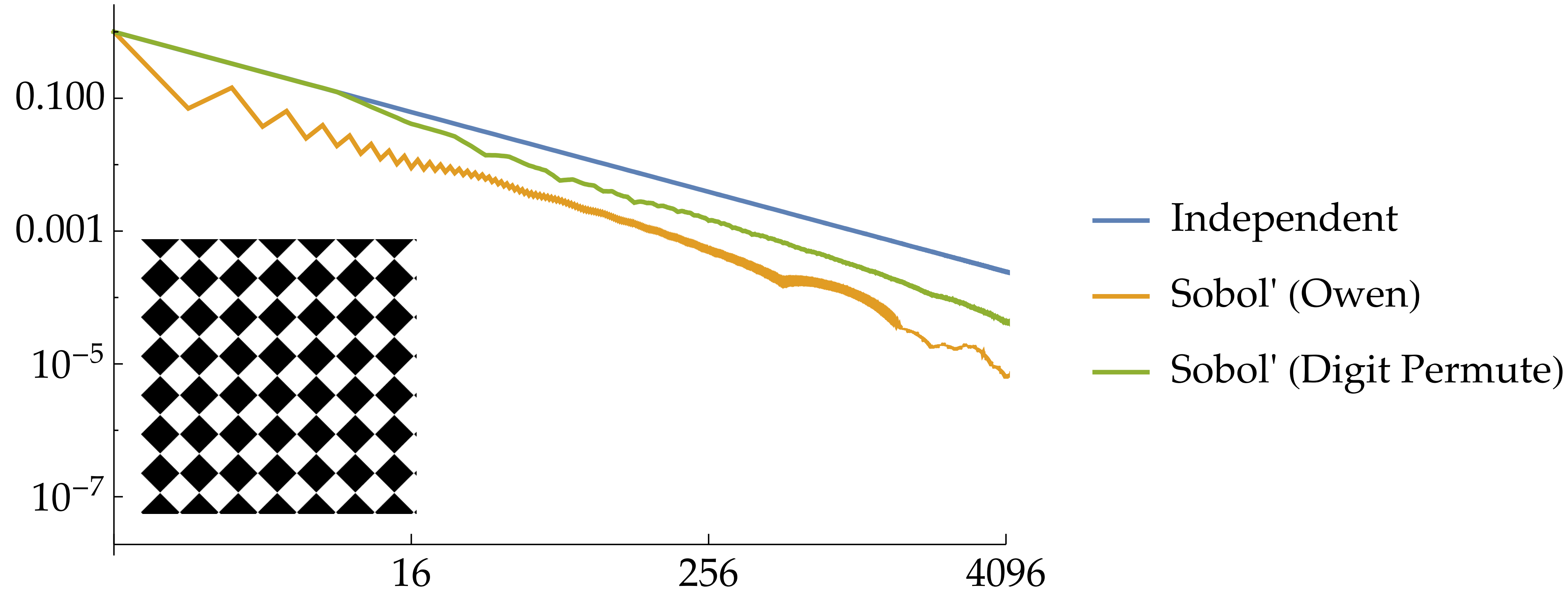
Permuted Radical Inverse

$$\Phi_b'(n) = \sum_{i=1}^{\infty} \sigma_{\{d_1,\ldots,d_{i-1}\}}(d_i) b^{-i}$$

Owen Scrambled Radical Inverse

Error With Owen Scrambling





Sobol’ Point Sets

Generator Matrices

Given a base b and a matrix C, define:

$$c(n) = (b^{-1}, b^{-2}, \ldots, b^{-m})\, C \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_m \end{pmatrix}$$

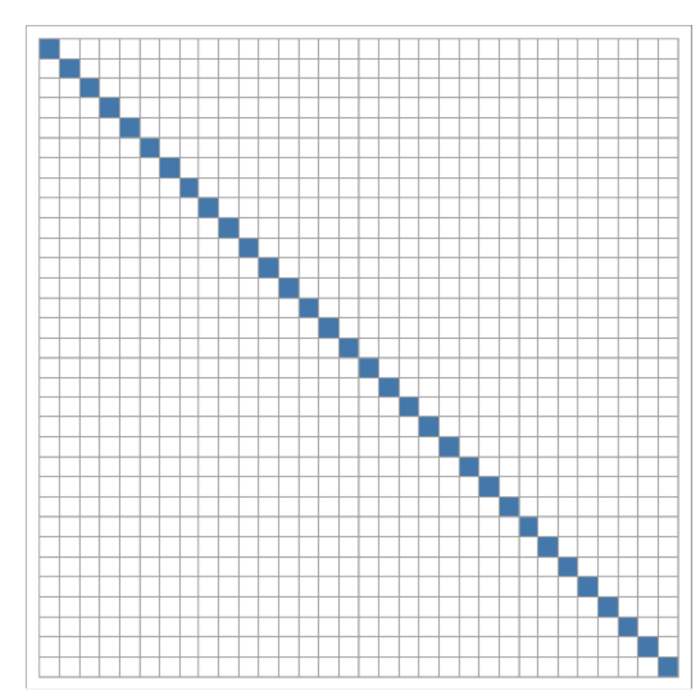
- **where d_i are the base-b digits of n**
- **and arithmetic is done over the ring $\mathbb{Z}_b$**
  - **For our purposes, just do everything "mod b"**

This generates a set of b^m points

Generator Matrices

We'll focus only on b=2, which allows particularly efficient implementation

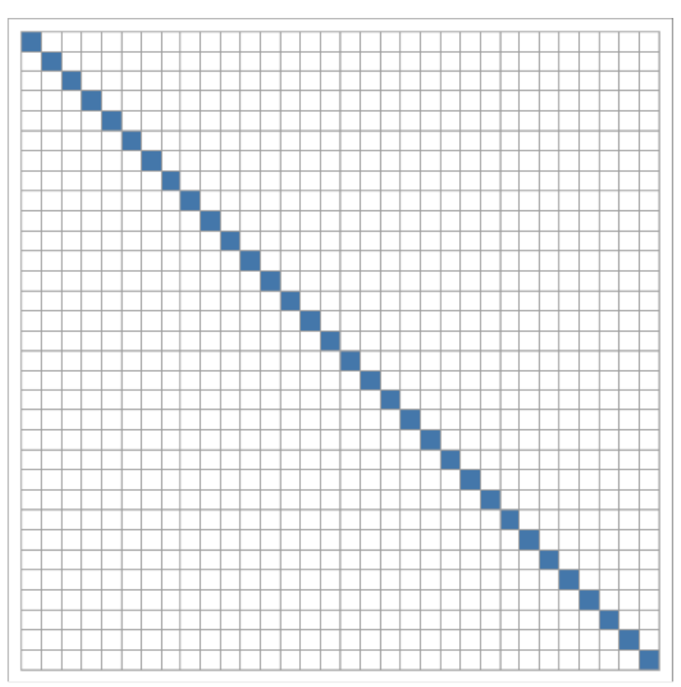
$$c(n) = (2^{-1}, 2^{-2}, \ldots, 2^{-m})\, C \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_m \end{pmatrix}$$

Quiz: what happens if $C =$  **?**

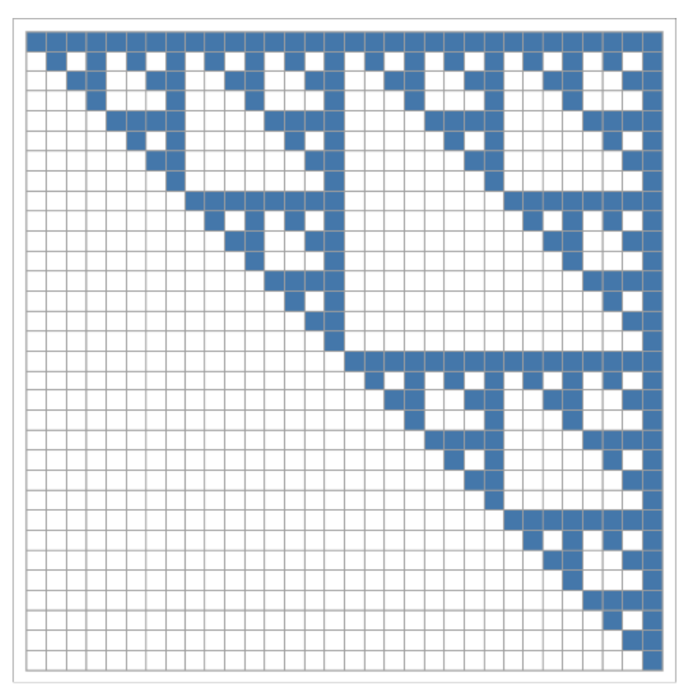
Sobol' Point Sets

Sobol' showed how to find generator matrices for LD point sets in base 2

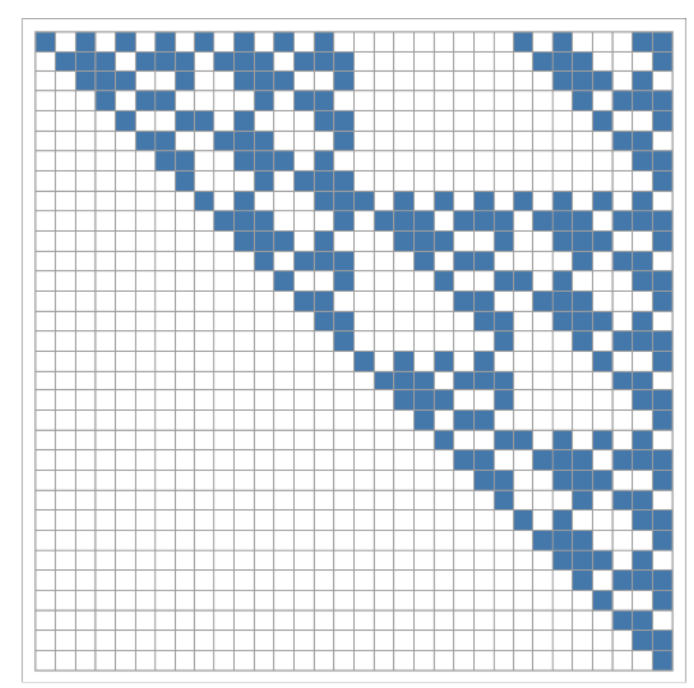
- **Can scale low-discrepancy samples in 1000s of dimensions**



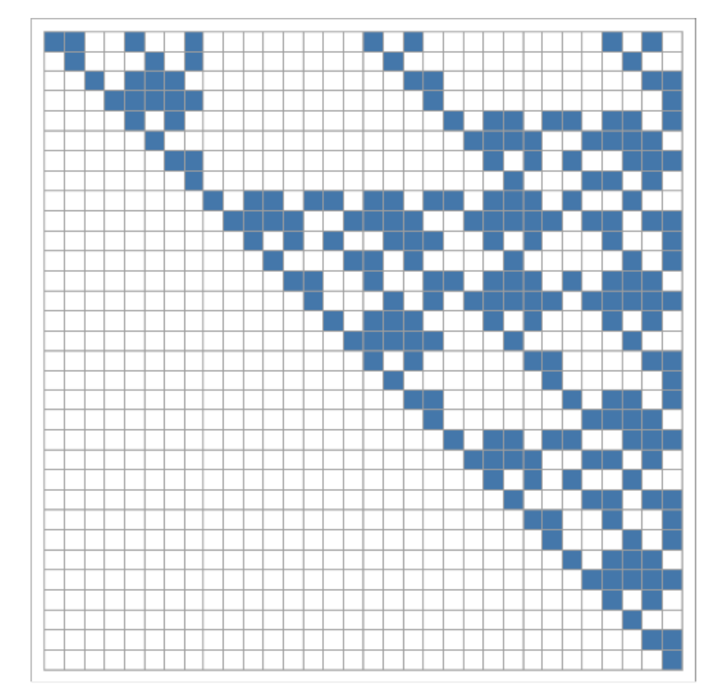
C_0



C_1



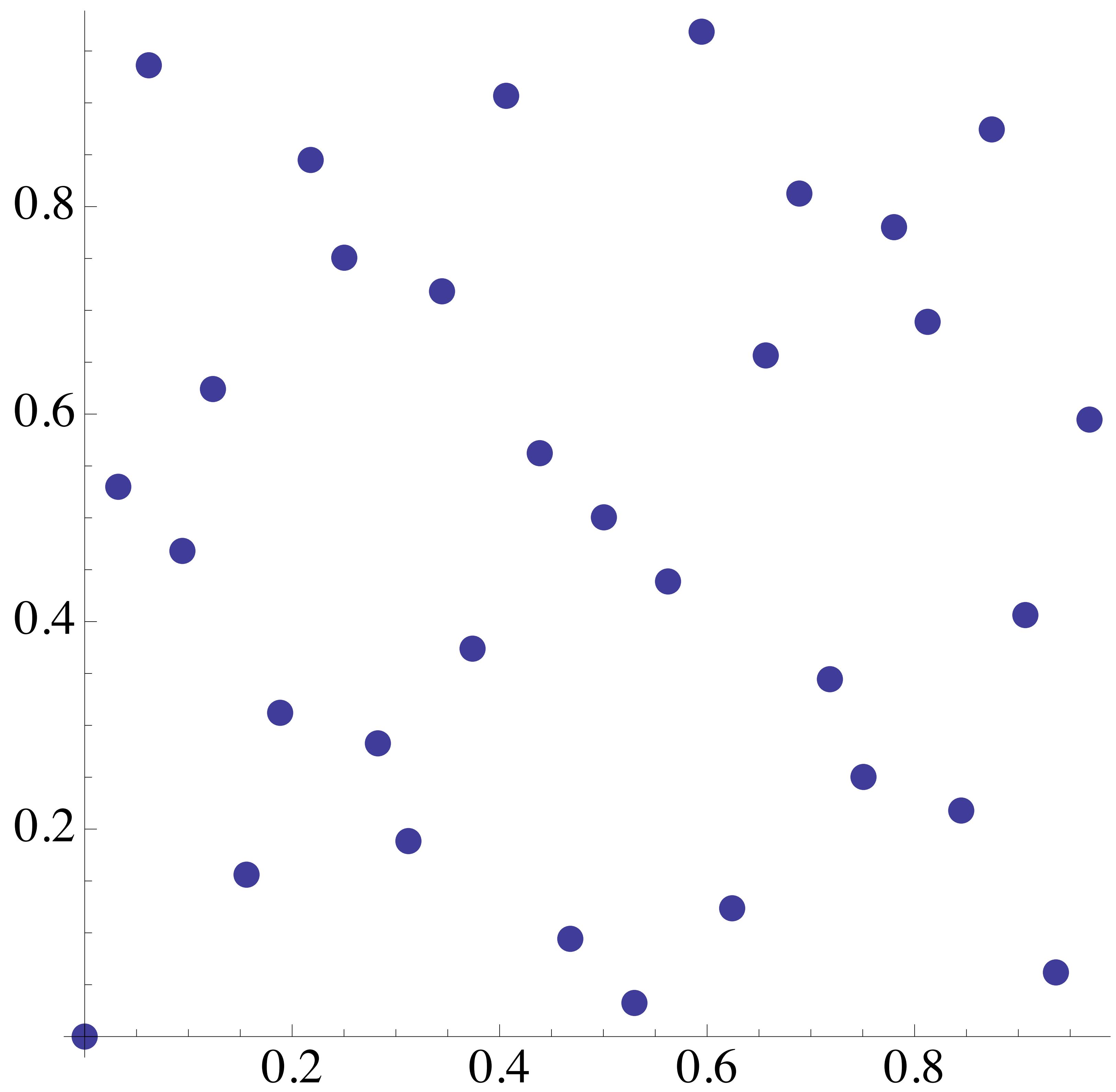
C_2



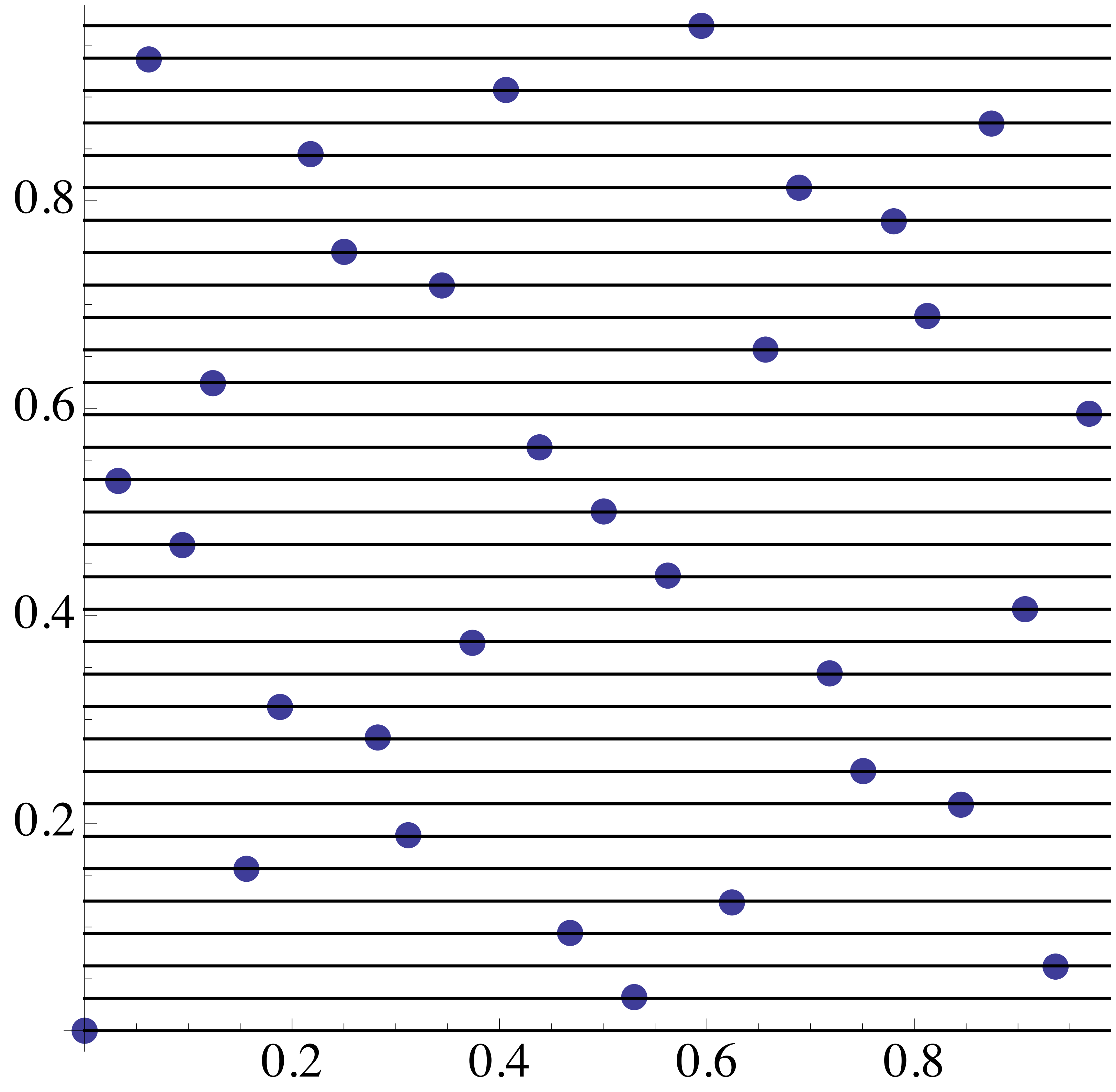
C_3

...

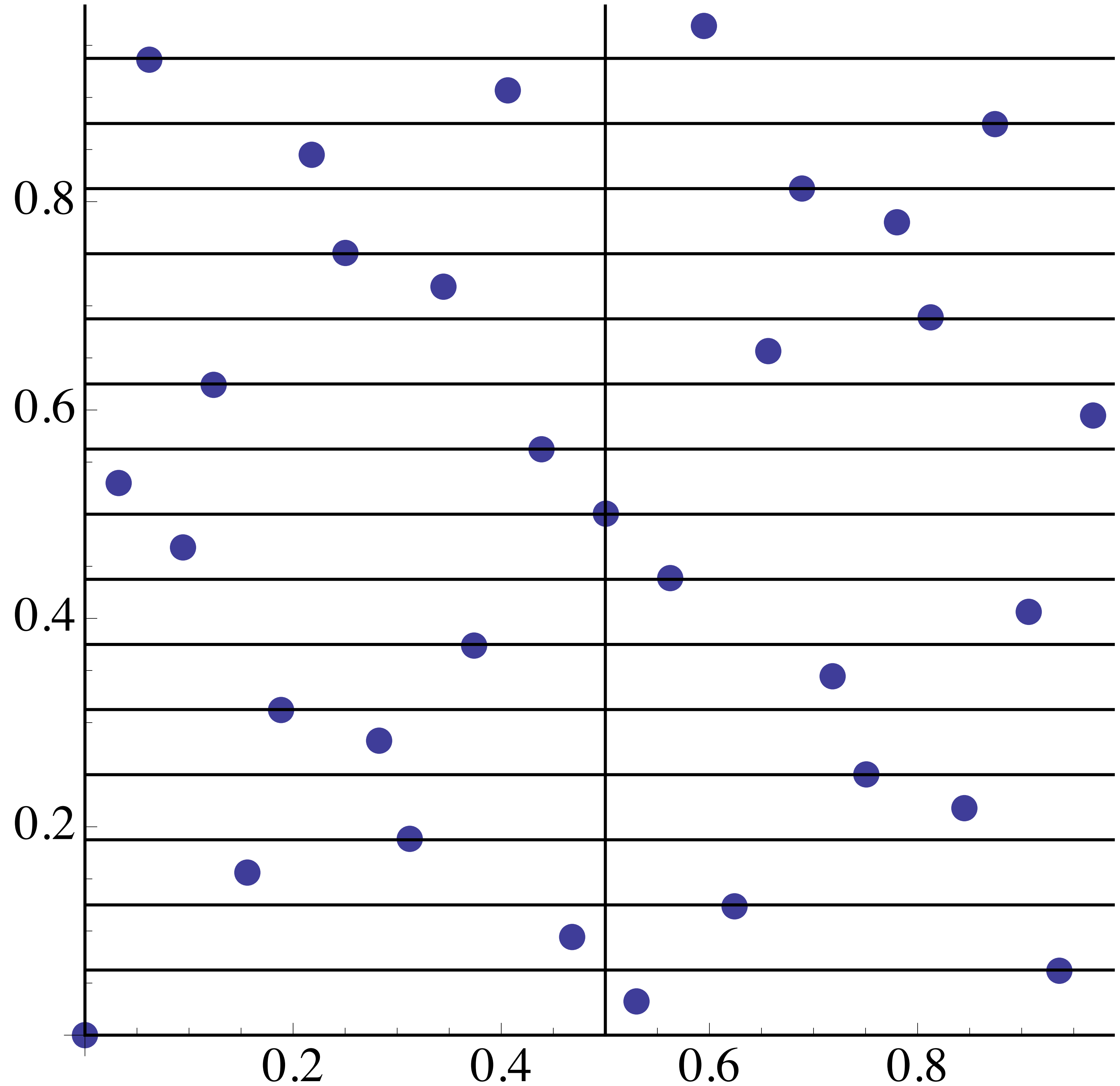
32 2D Sobol’ Points



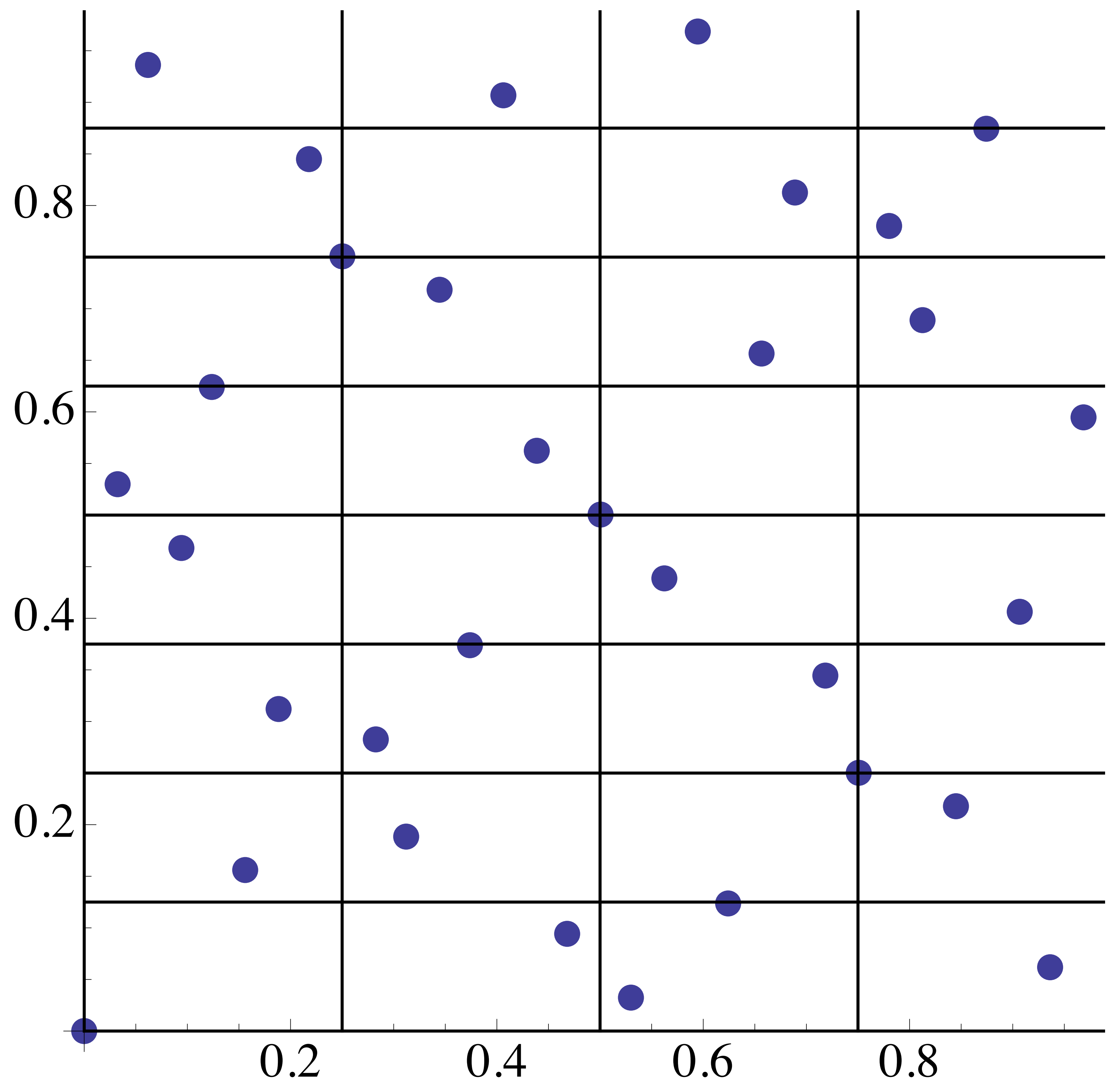
Elementary Intervals (1x32)



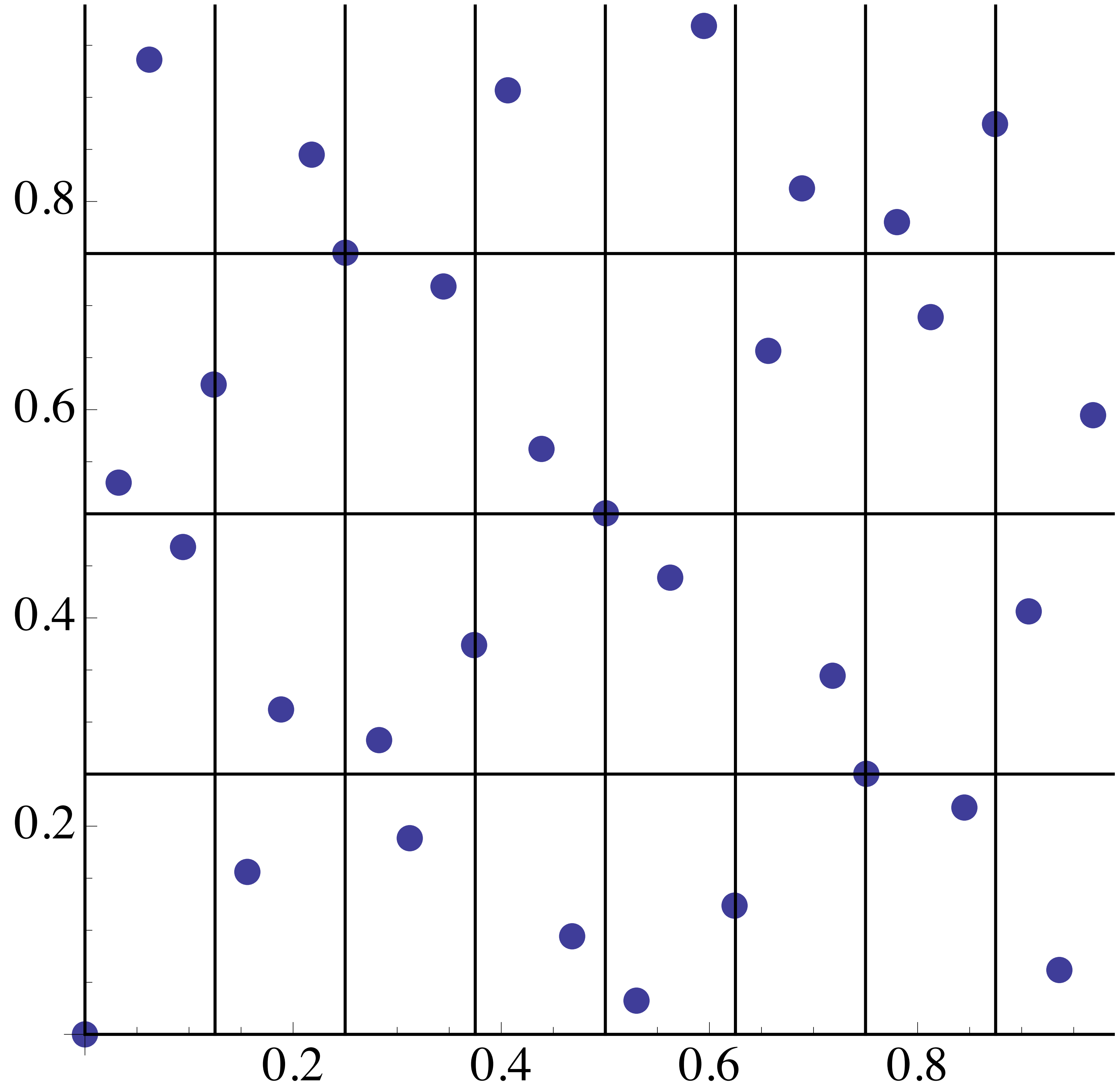
Elementary Intervals (2x16)



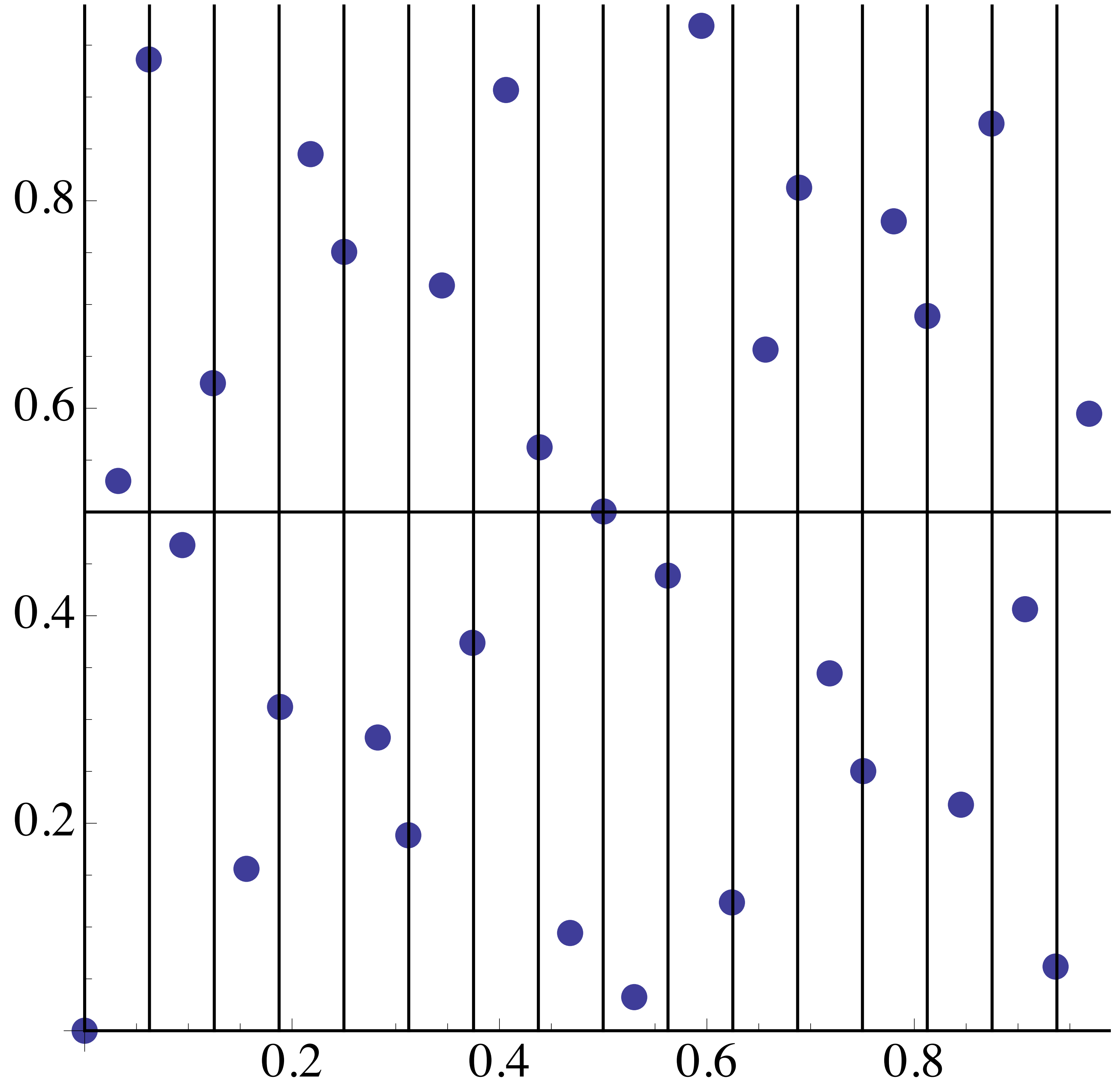
Elementary Intervals (4x8)



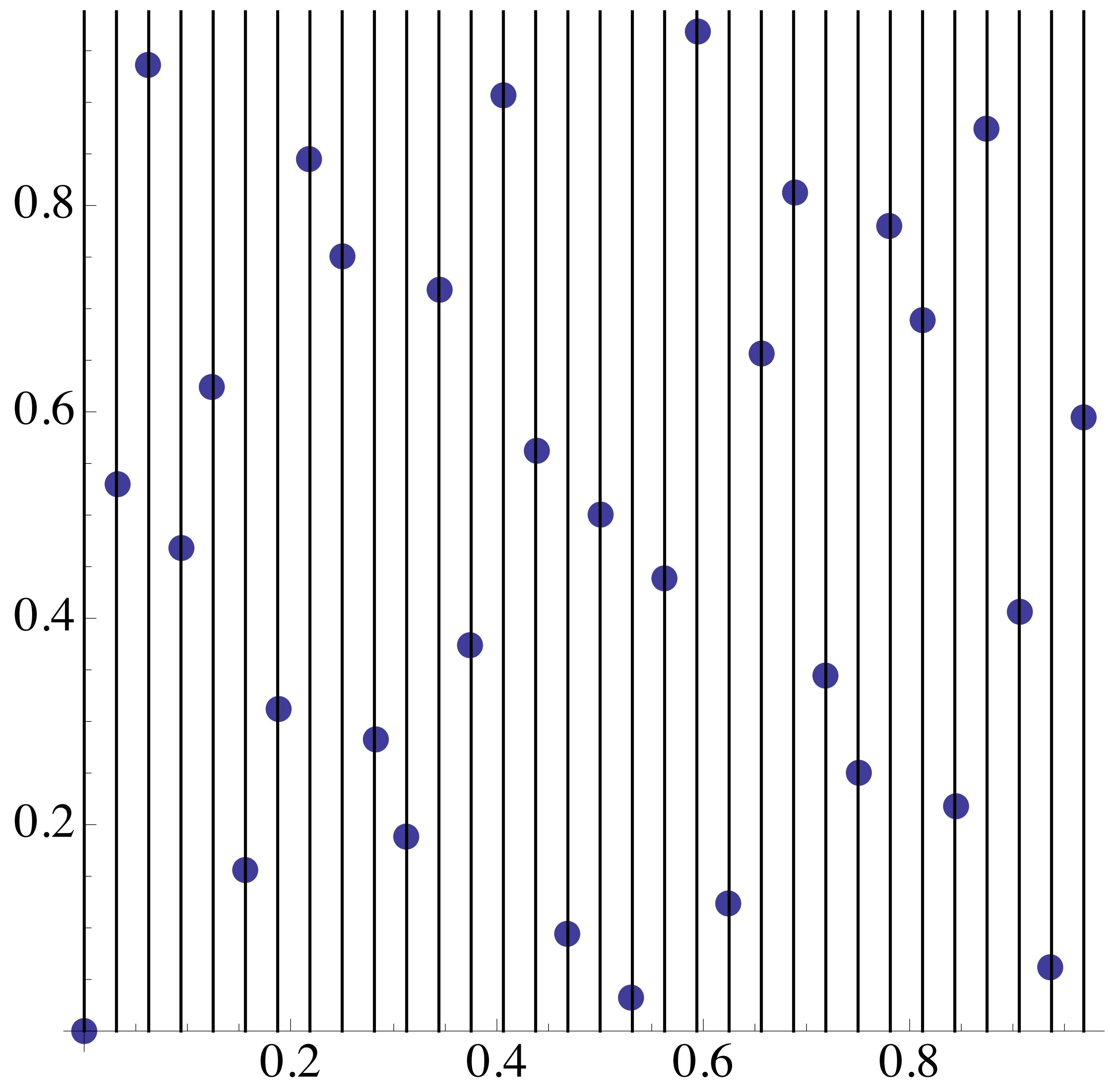
Elementary Intervals (8x4)

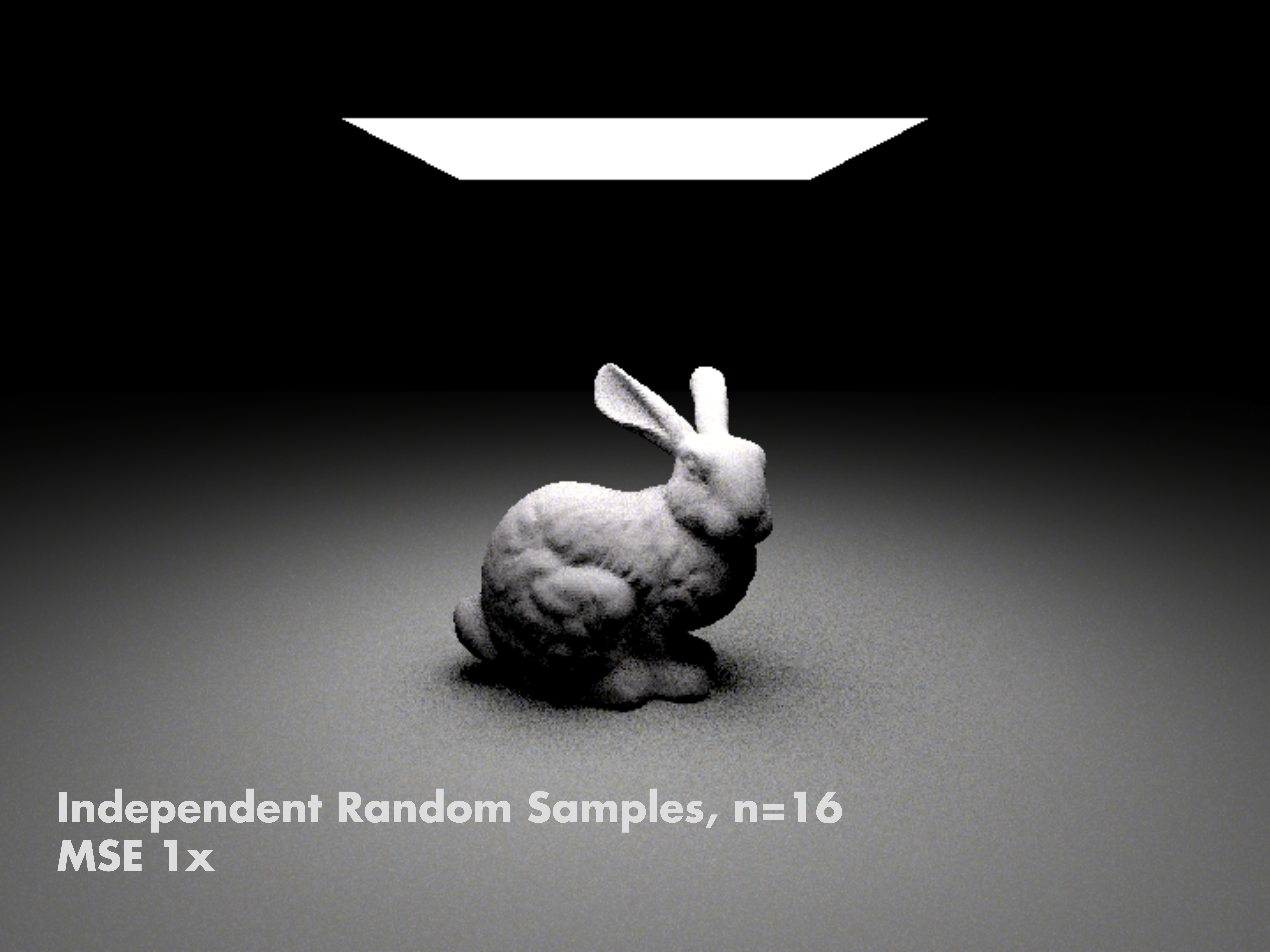


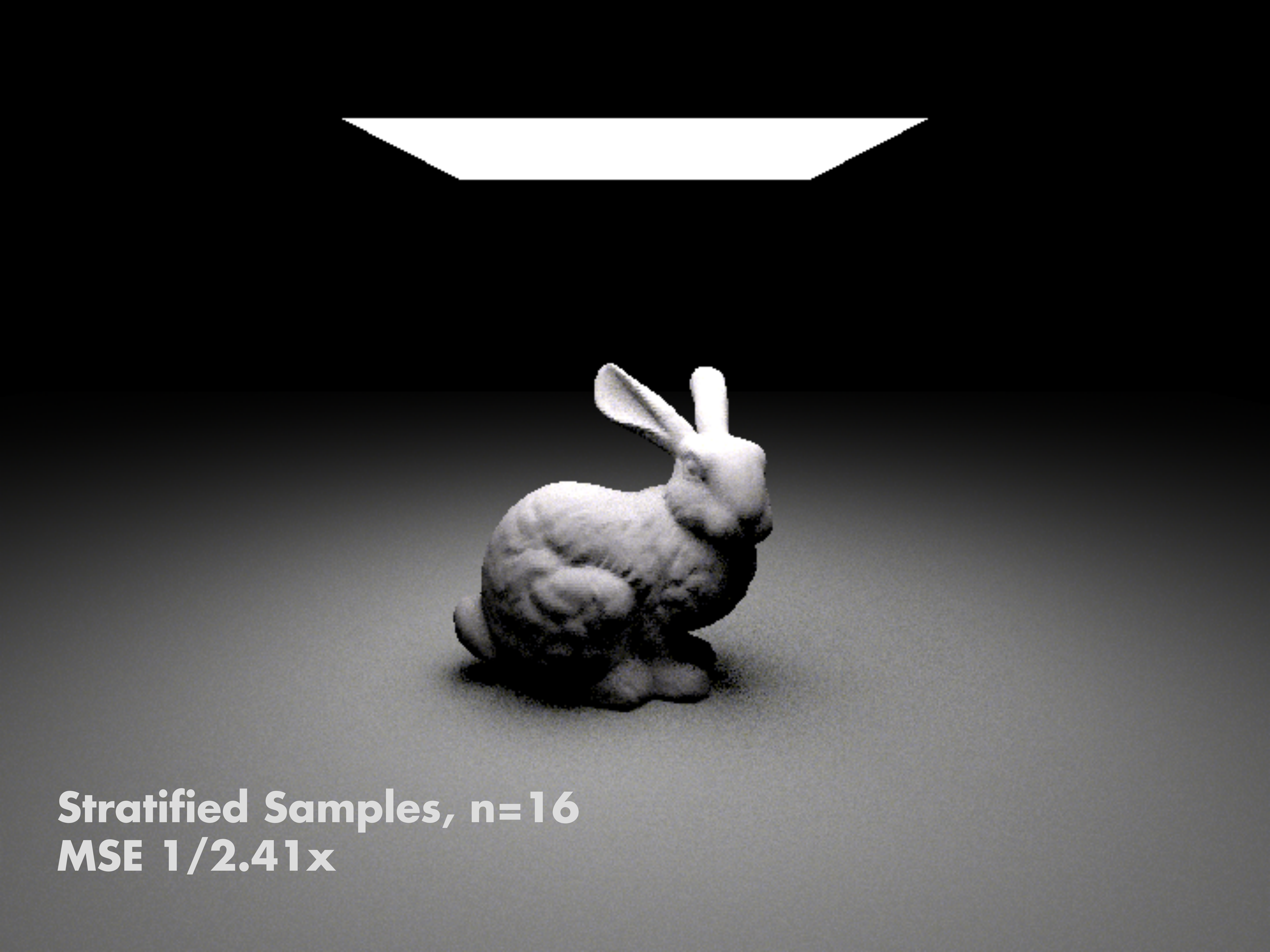
Elementary Intervals (16x2)

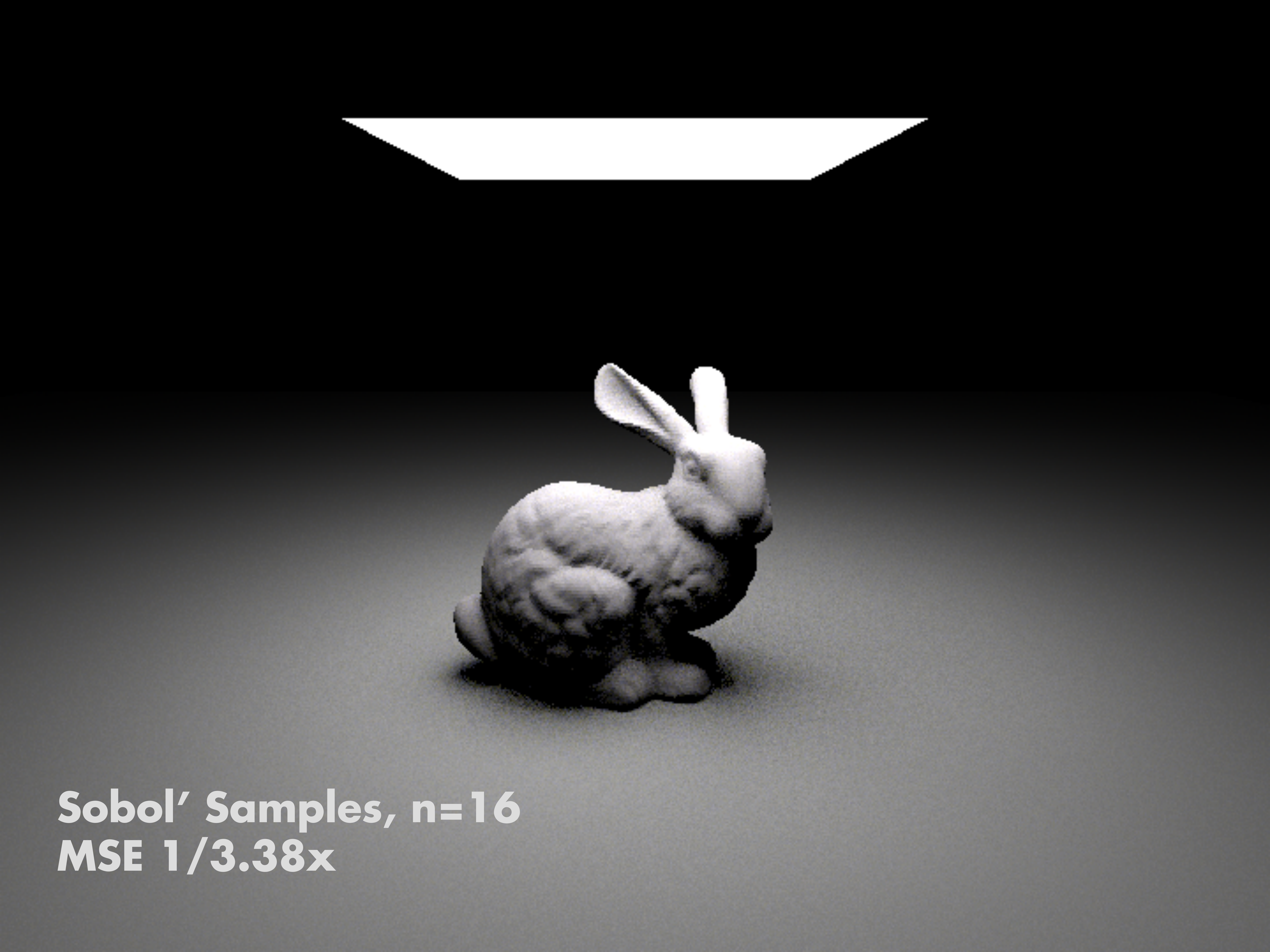


Elementary Intervals (32x1)




Independent Random Samples, n=16
MSE 1x


Stratified Samples, n=16
MSE 1/2.41x

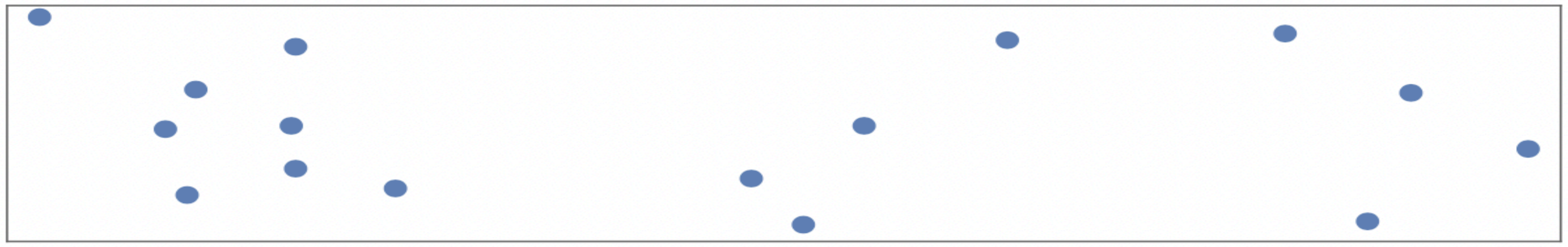

Sobol’ Samples, n=16
MSE 1/3.38x

Independent Random Samples, n=16
MSE 1x

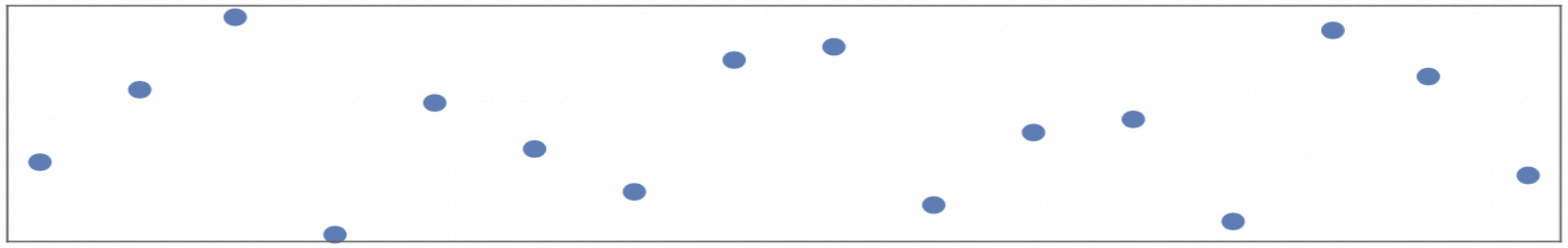

Stratified Samples, n=16
MSE 1/2.12x


Sobol' Samples, n=16
MSE 1/3.95x

Warping Samples to a Quad Light

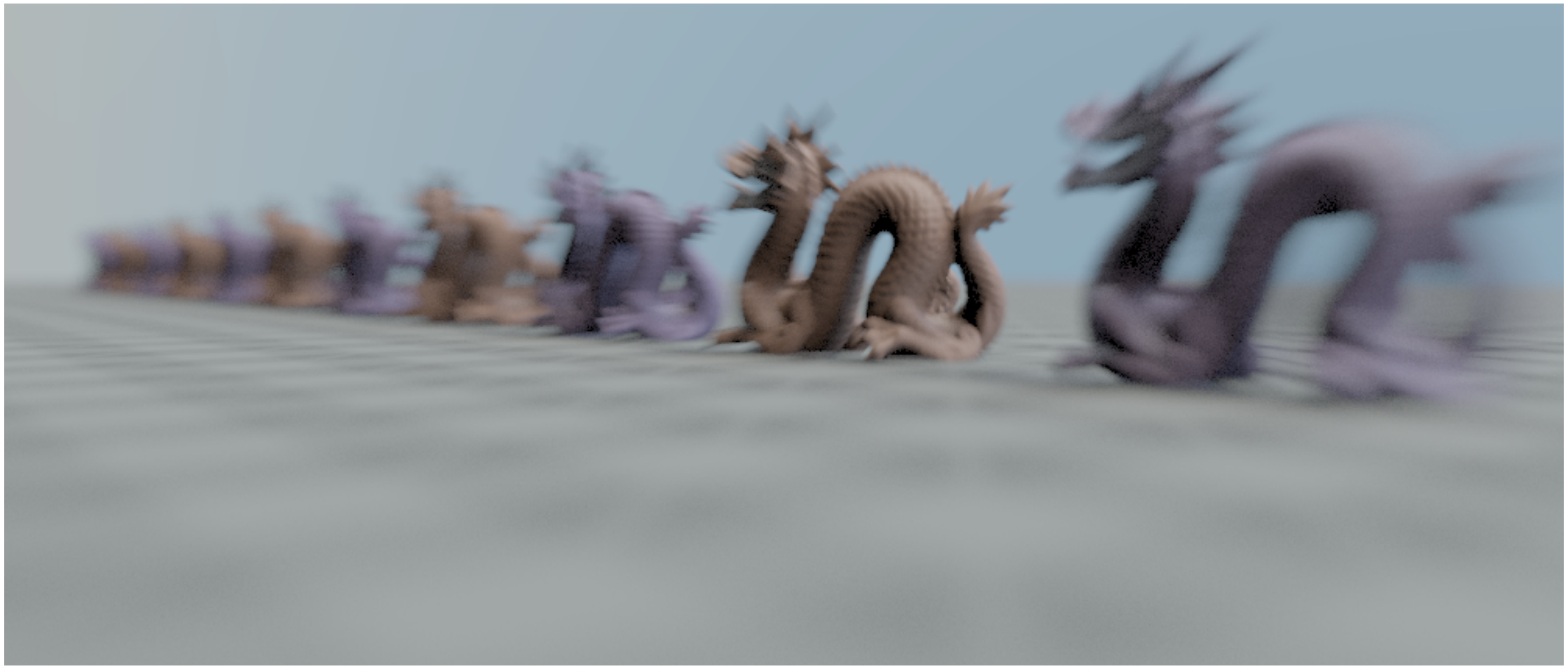


4x4 Stratified



16 Sobol'

Sampling Motion Blur + Defocus



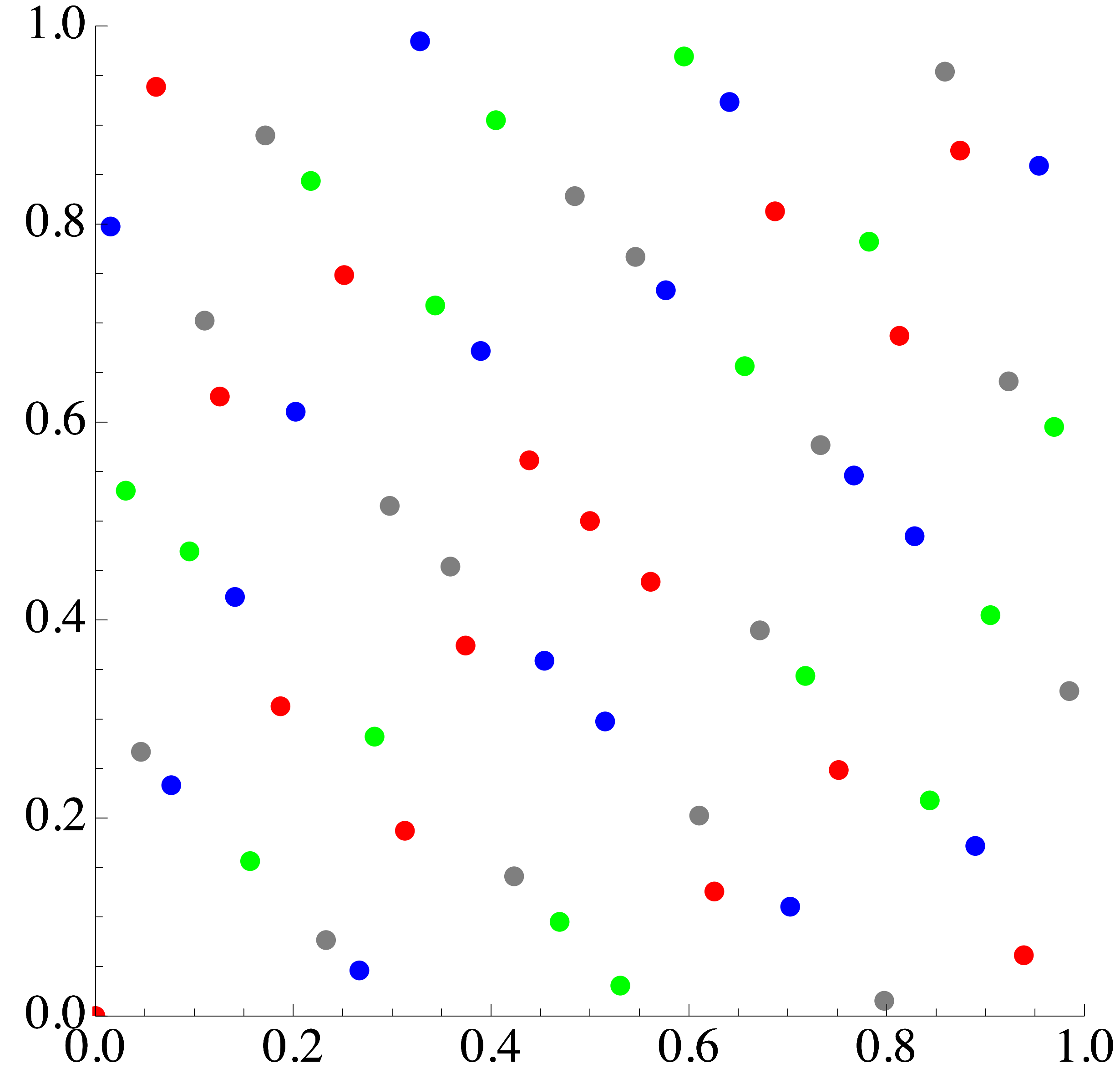
Independent
MSE 1x

Halton
MSE 1/1.13x

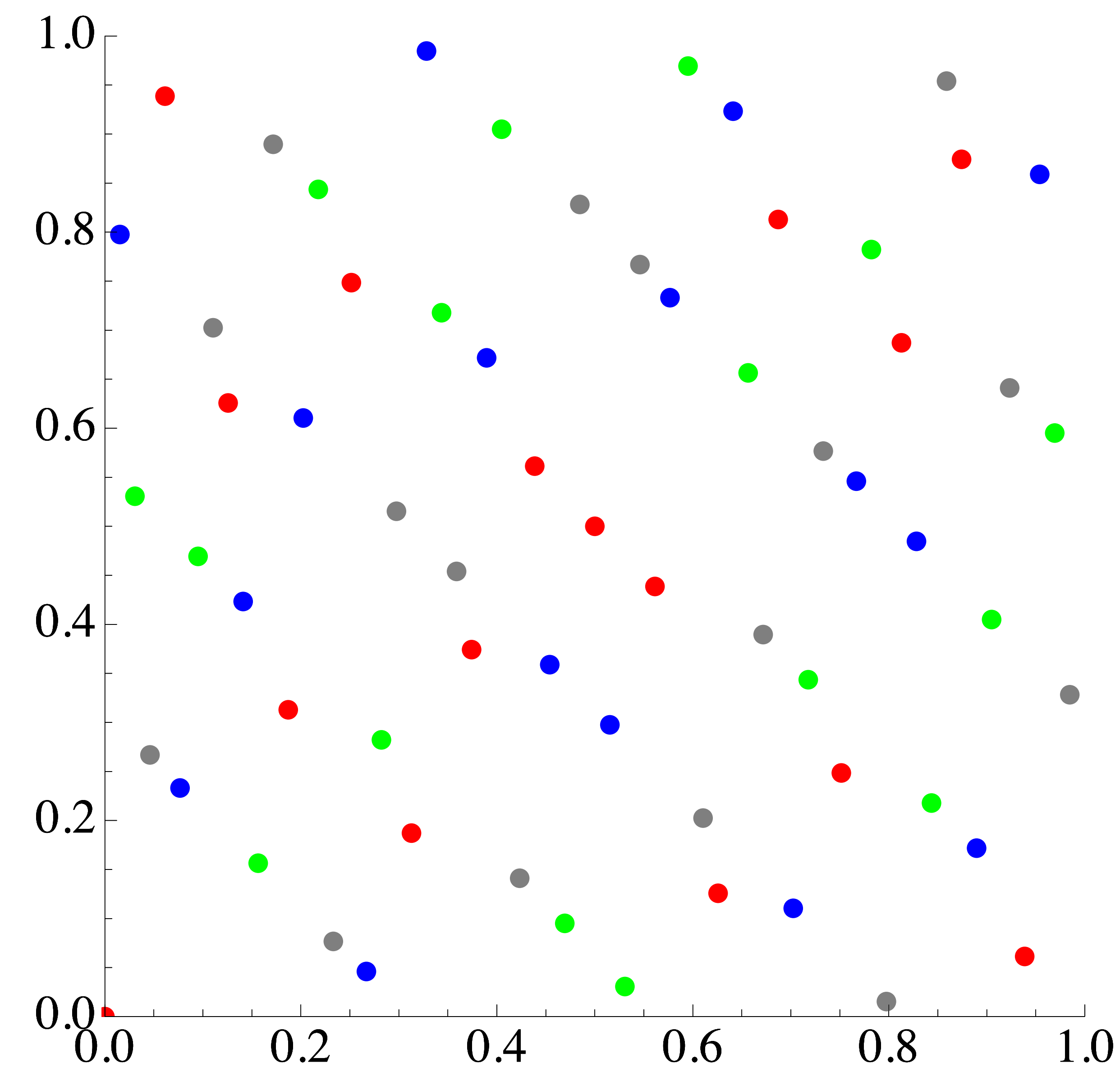
Sobol'
MSE 1/1.80x

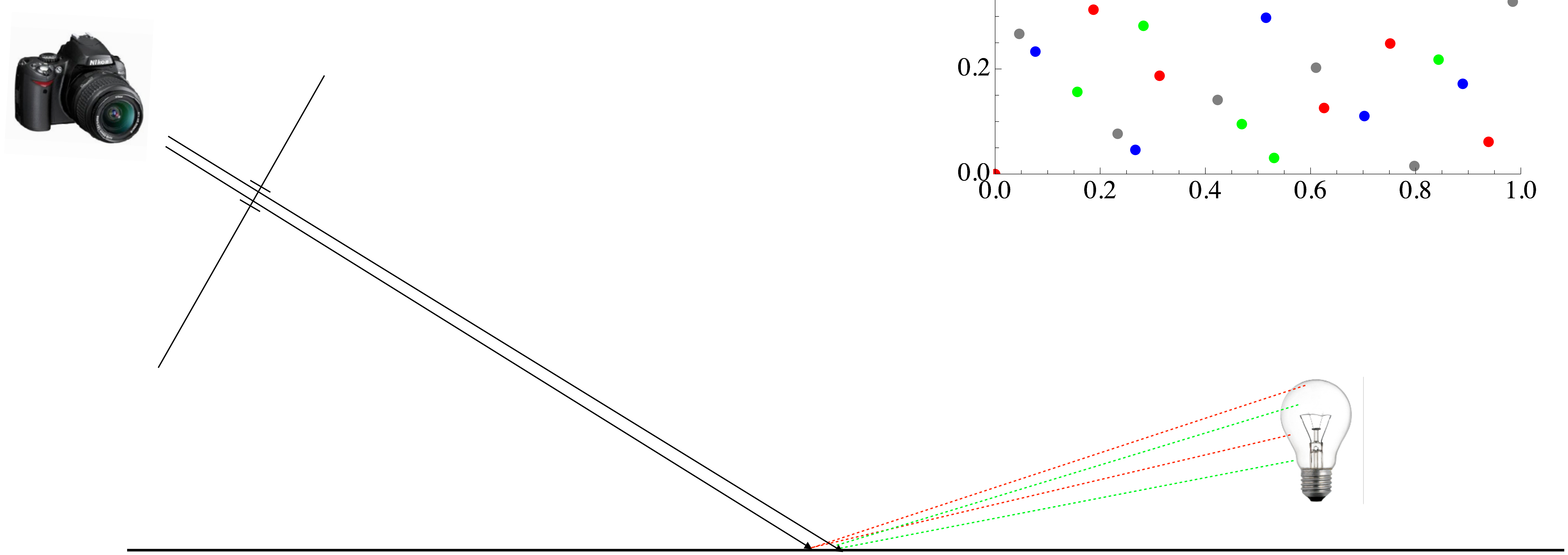
(0,2)-sequences

In addition to satisfying general stratification properties, power-of-two length subsequences are well-distributed with respect to each other



Pixel * Light Sampling





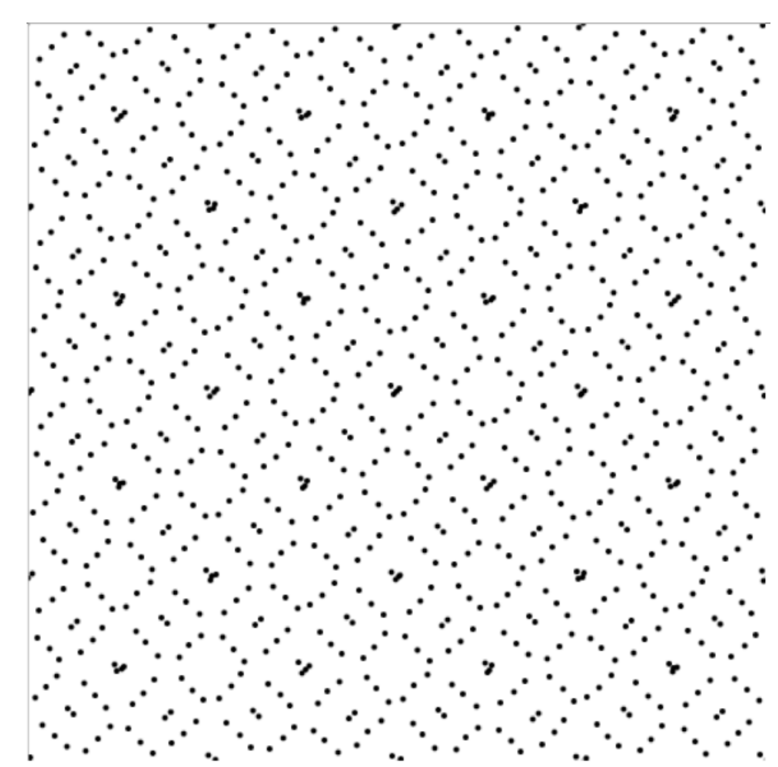
Spectral Analysis of Sampling

Measuring Point Set Quality

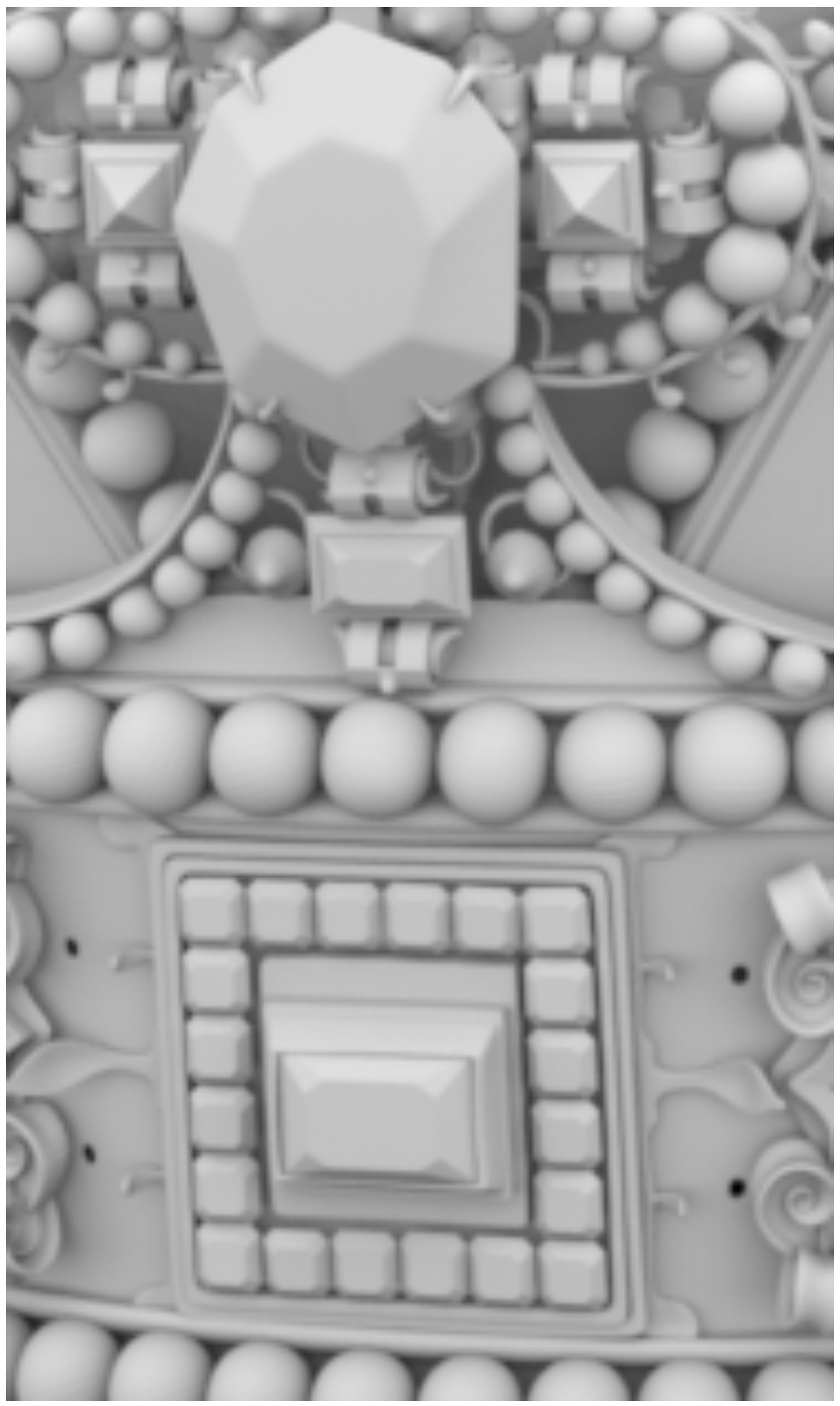
Some problems with discrepancy:

- Anisotropic: rotating the points changes discrepancy
- Not shift-invariant: similarly for translation
- Doesn't account for human perception

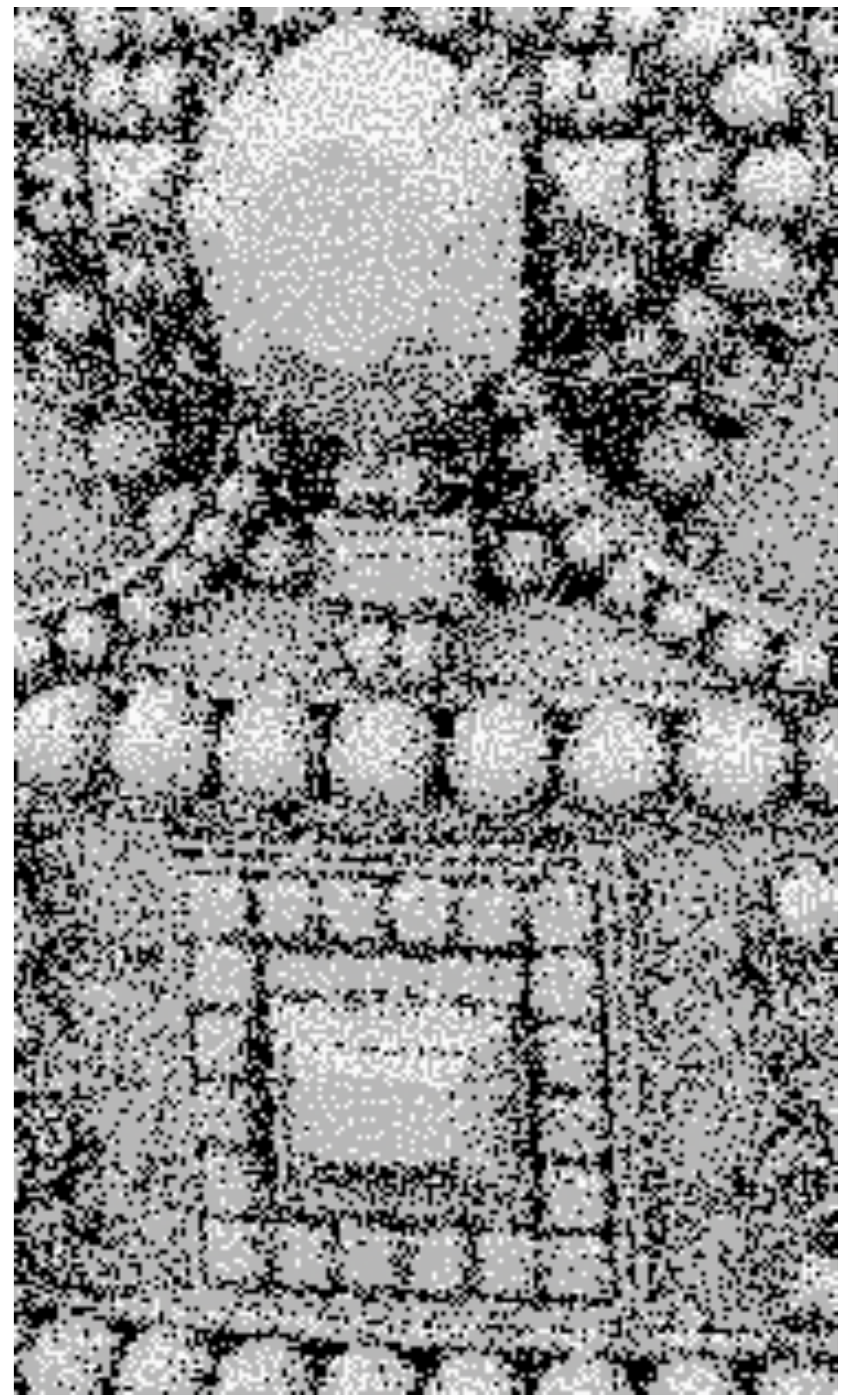
In general, can have low discrepancy yet still have points clumped together:



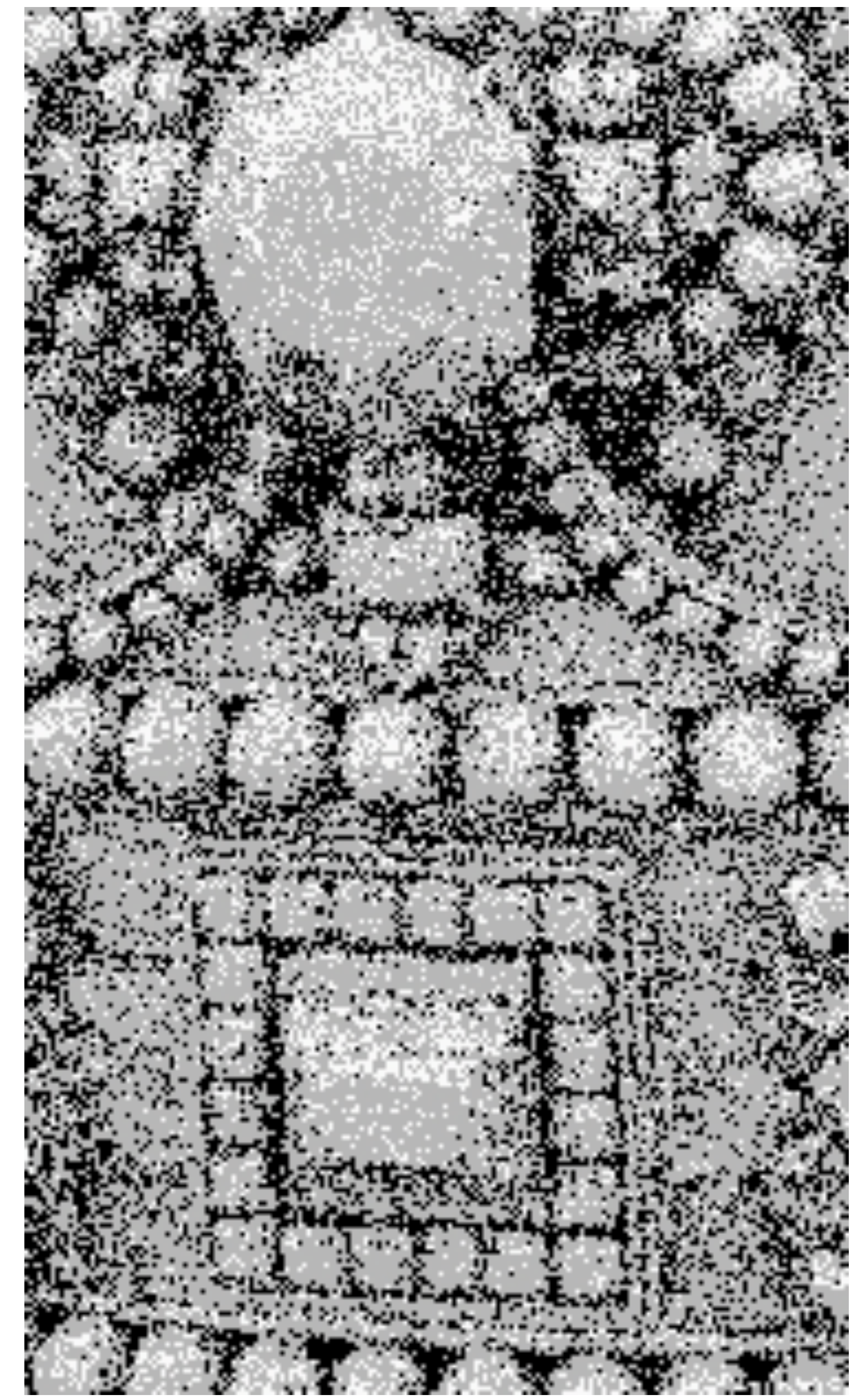
Ambient Occlusion: $\int_\Omega V(\omega)\,\cos\theta\,\mathrm{d}\omega$



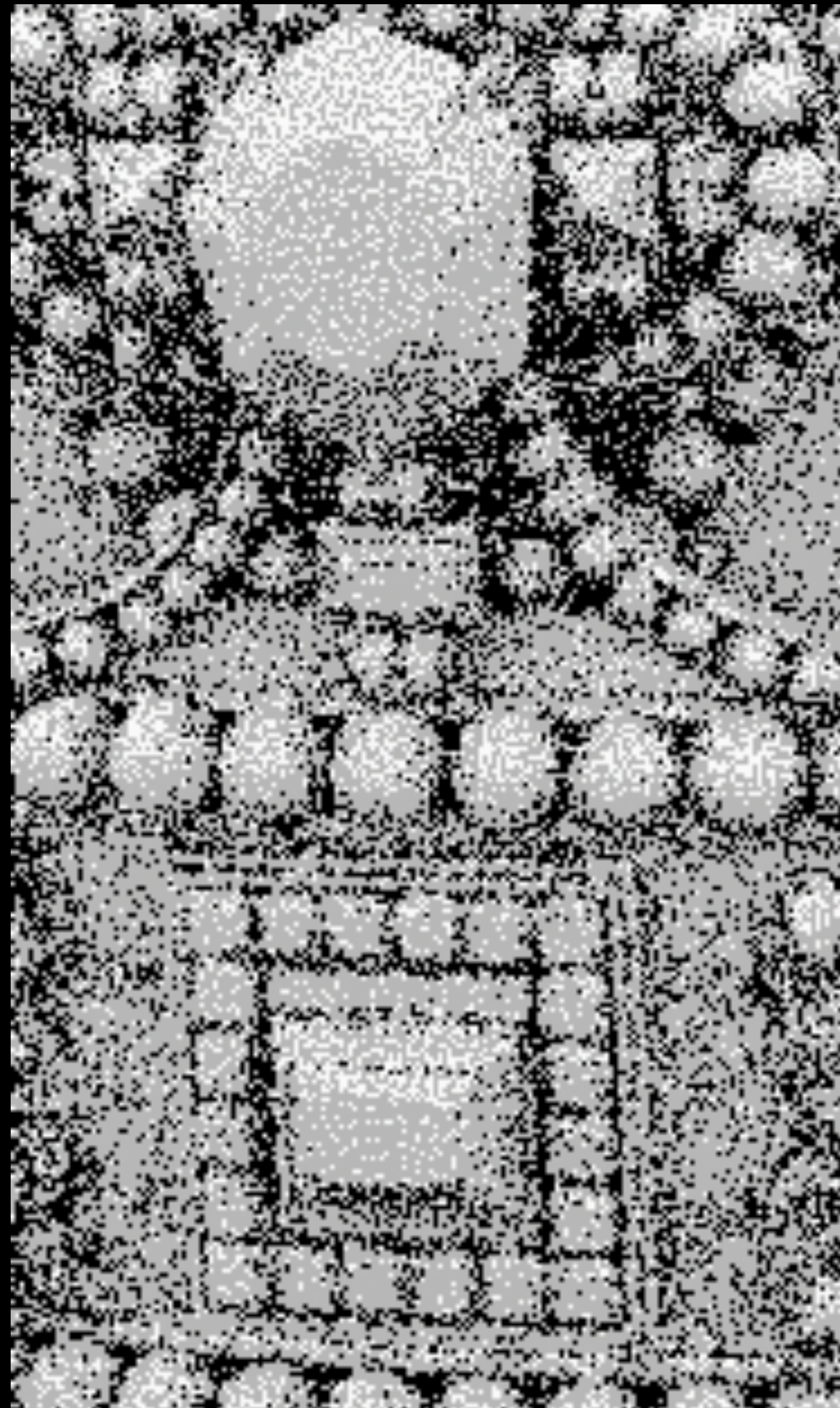
Reference

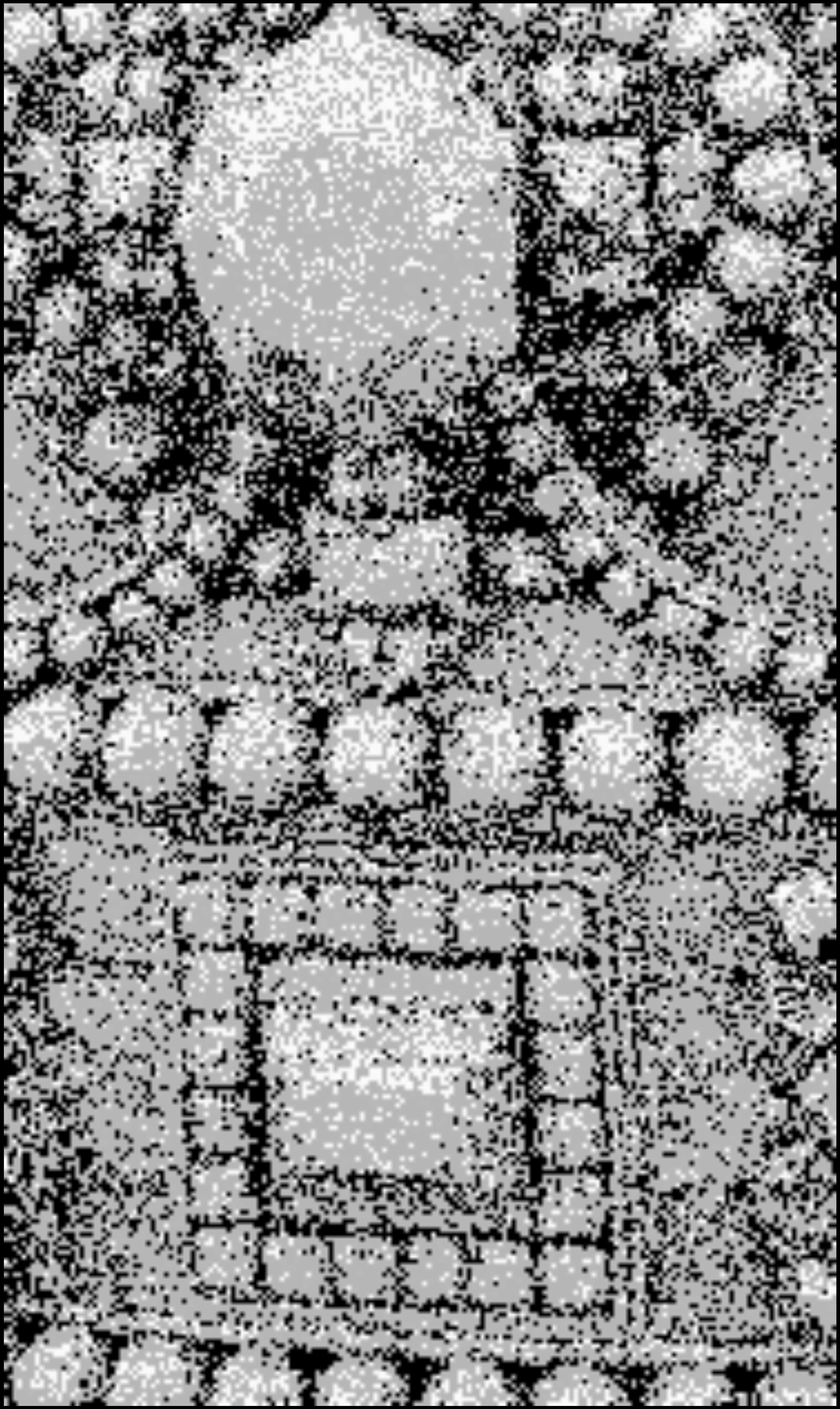


Random A



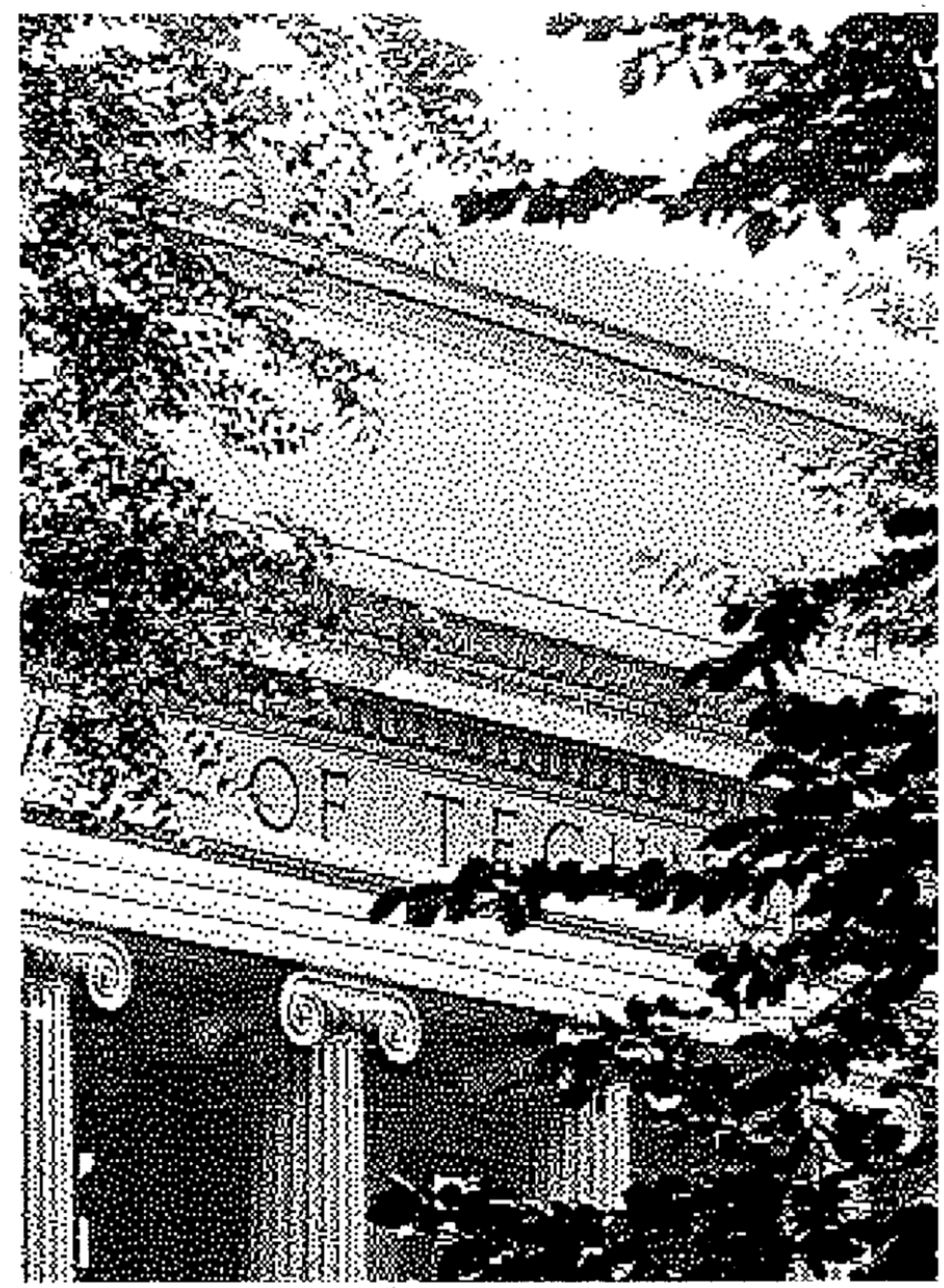
Random B





Blue Noise Dithering (Ulichney)

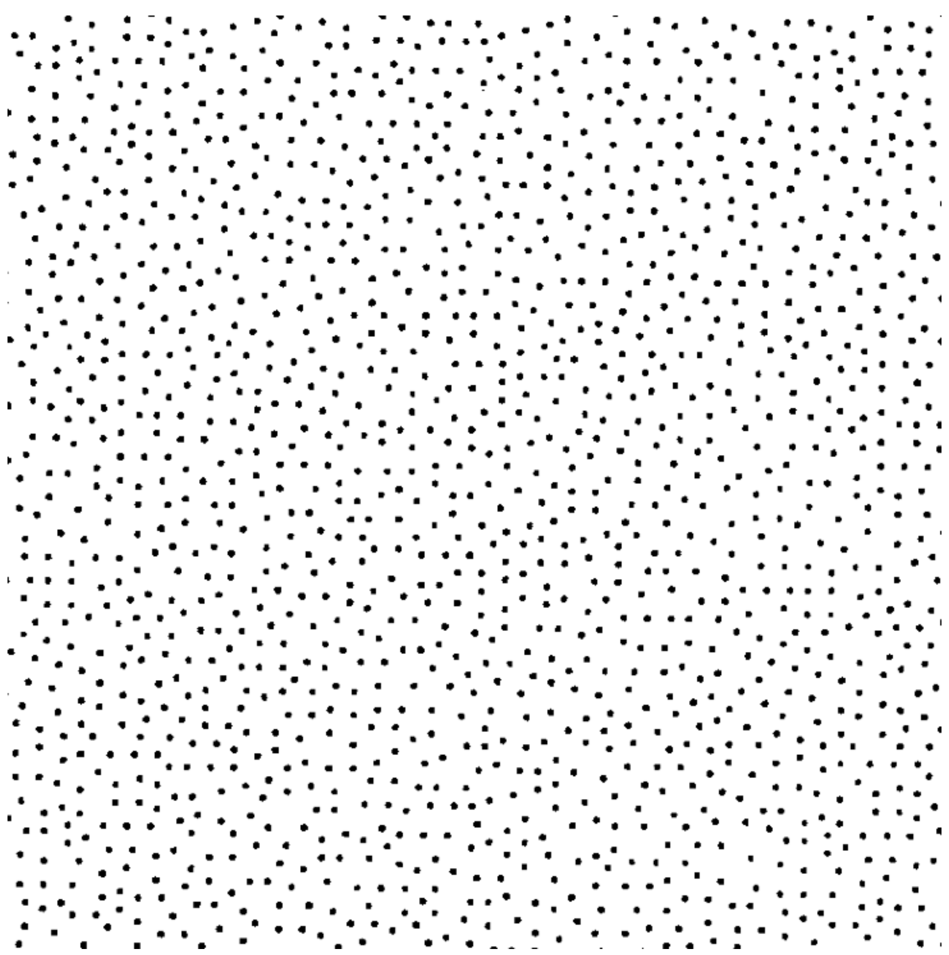
White

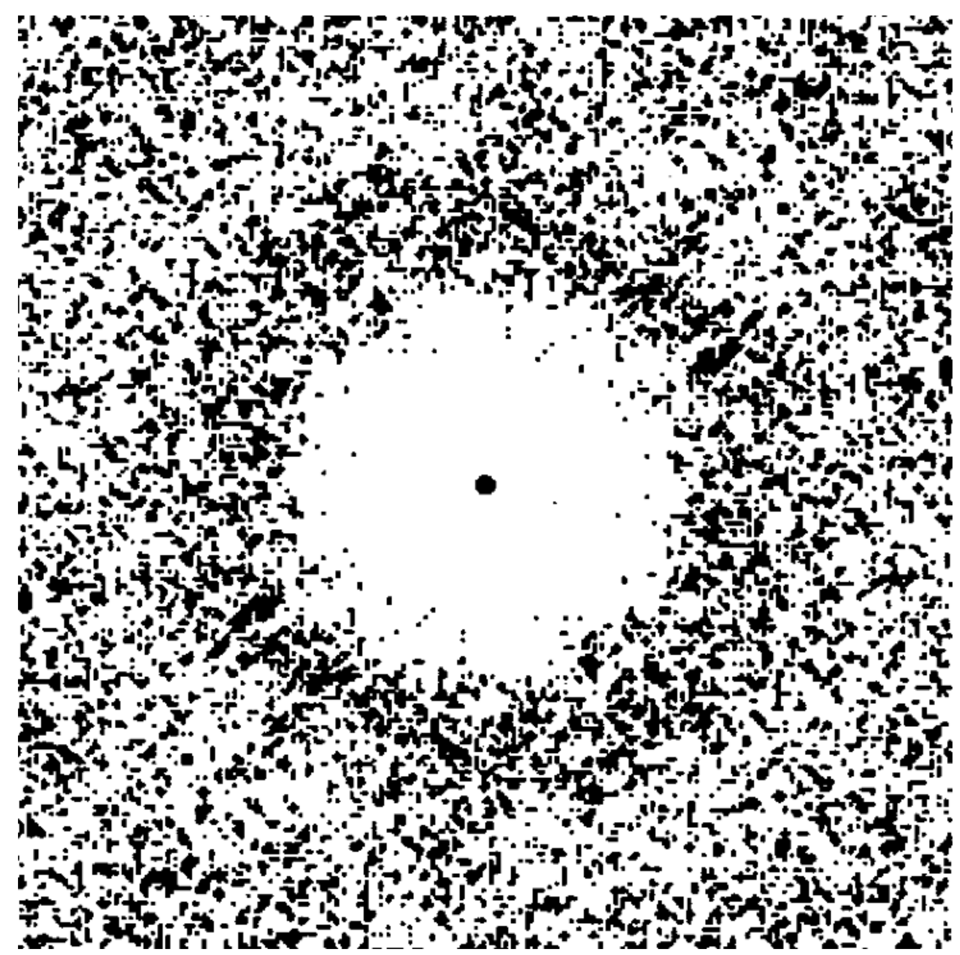


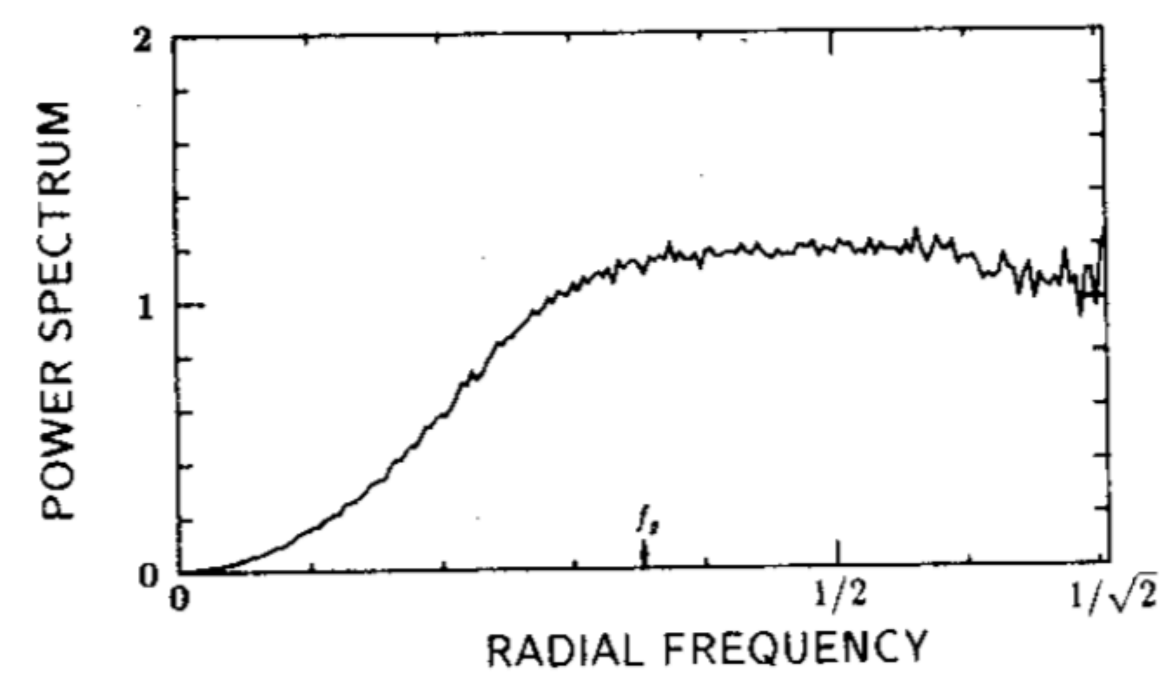
Blue

Power Spectrum of Samples

Samples | **Fourier Transform** | **Radial Power Spectrum**





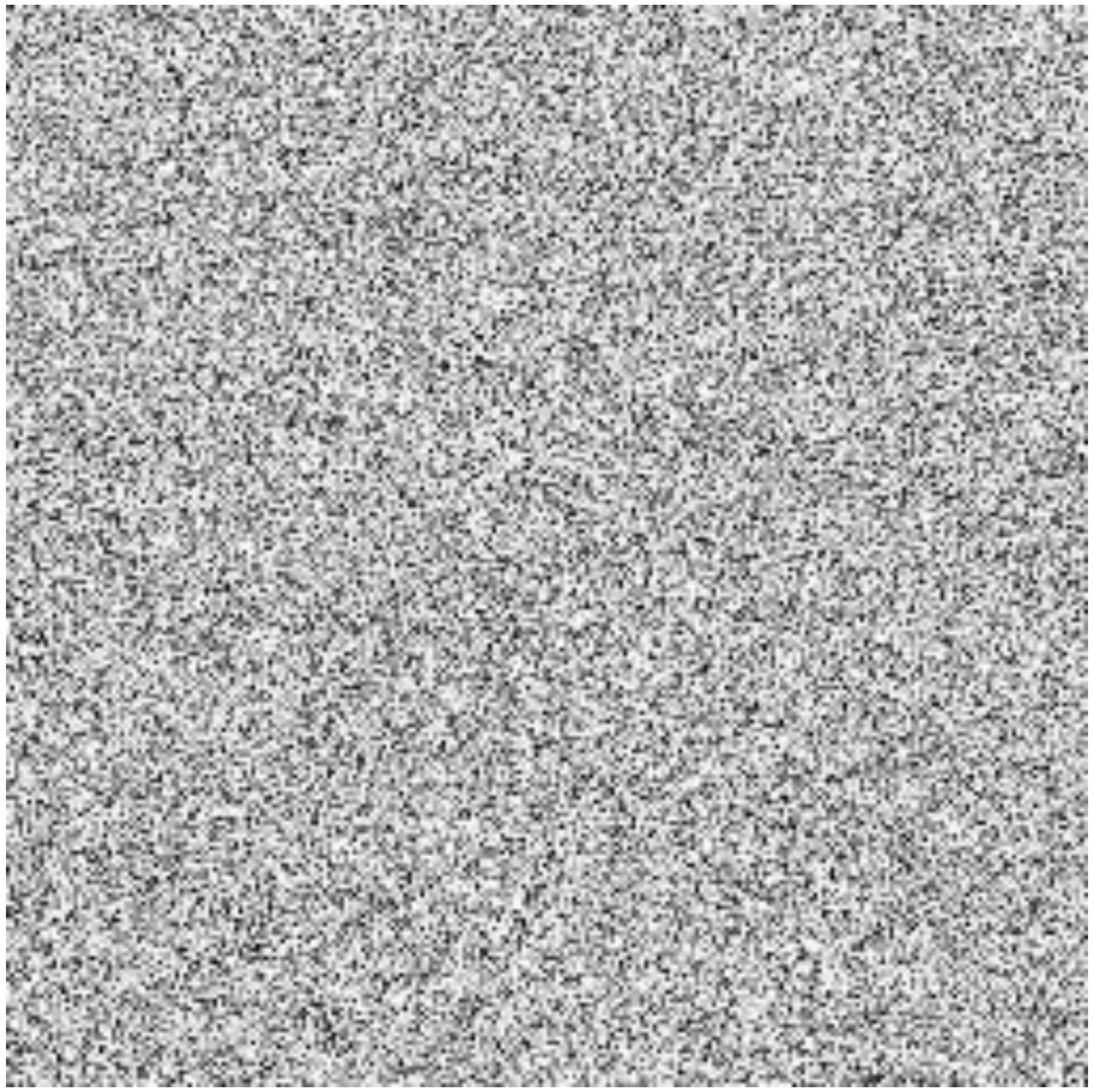


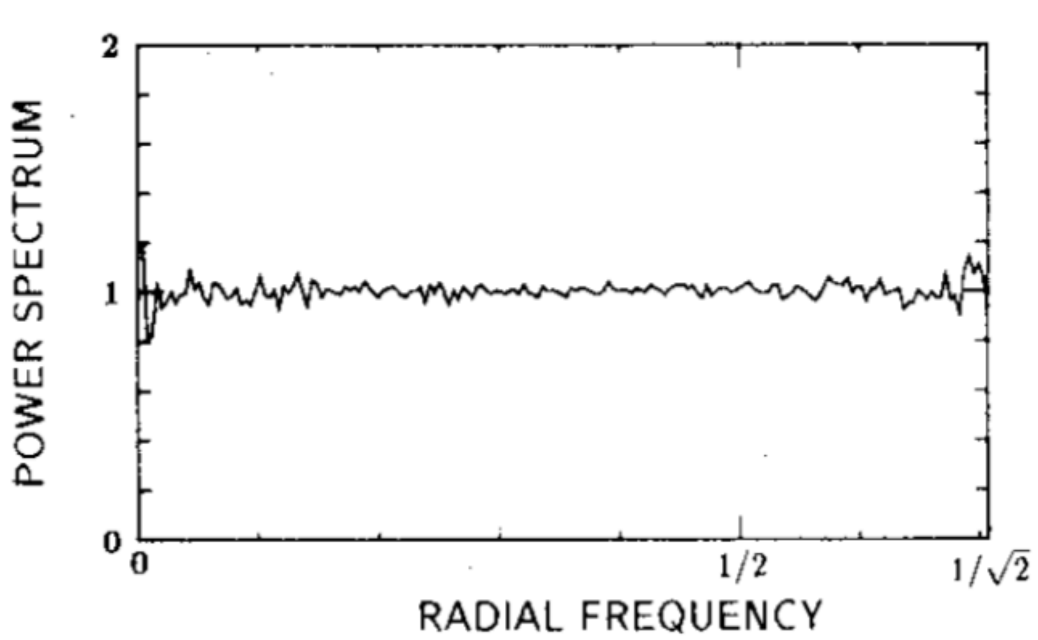
$$P_f(\omega) = F(\omega)\overline{F(\omega)}$$

$$= F(\omega)^2 \qquad f \text{ is even}$$

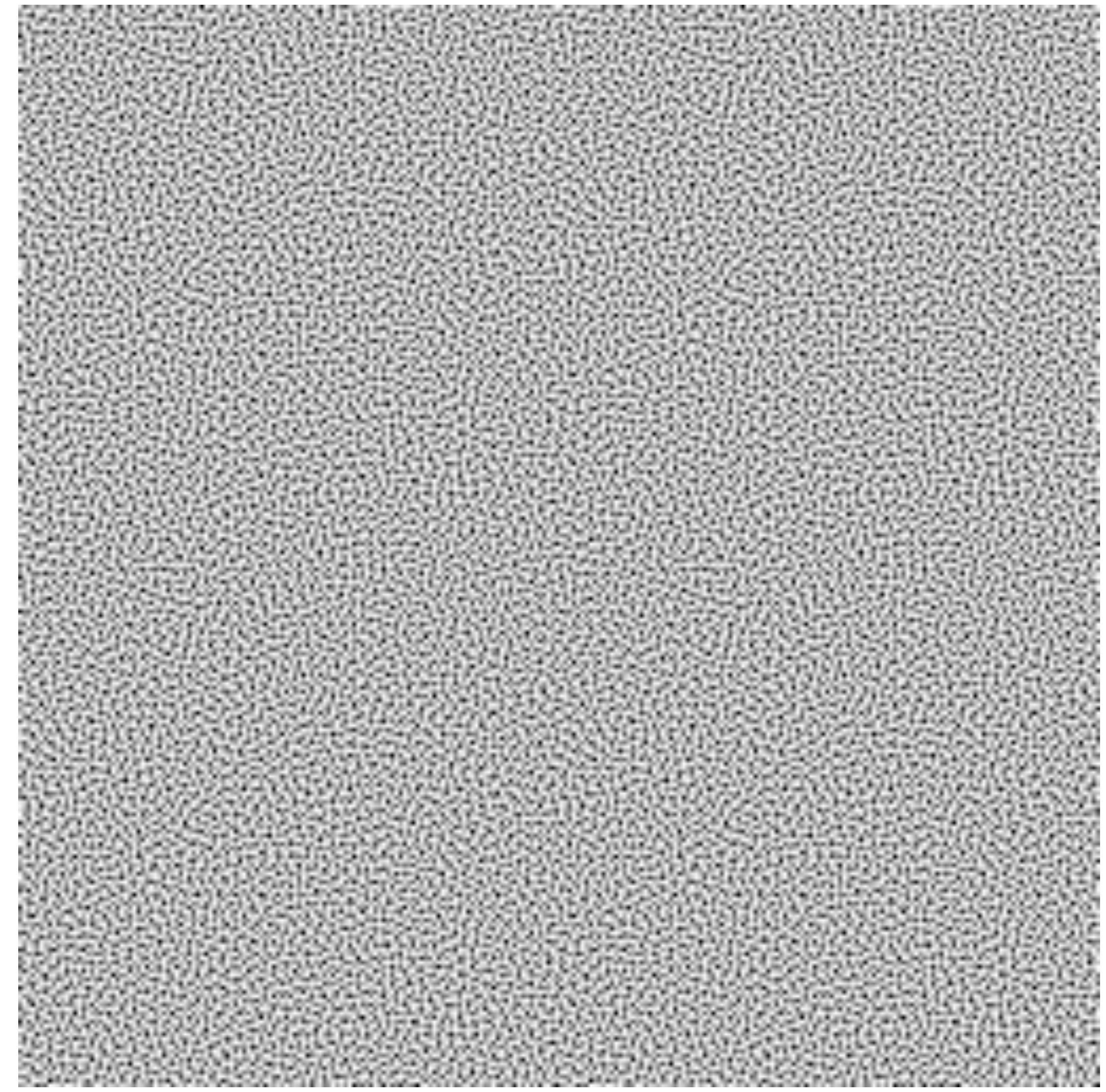
Colors of Noise

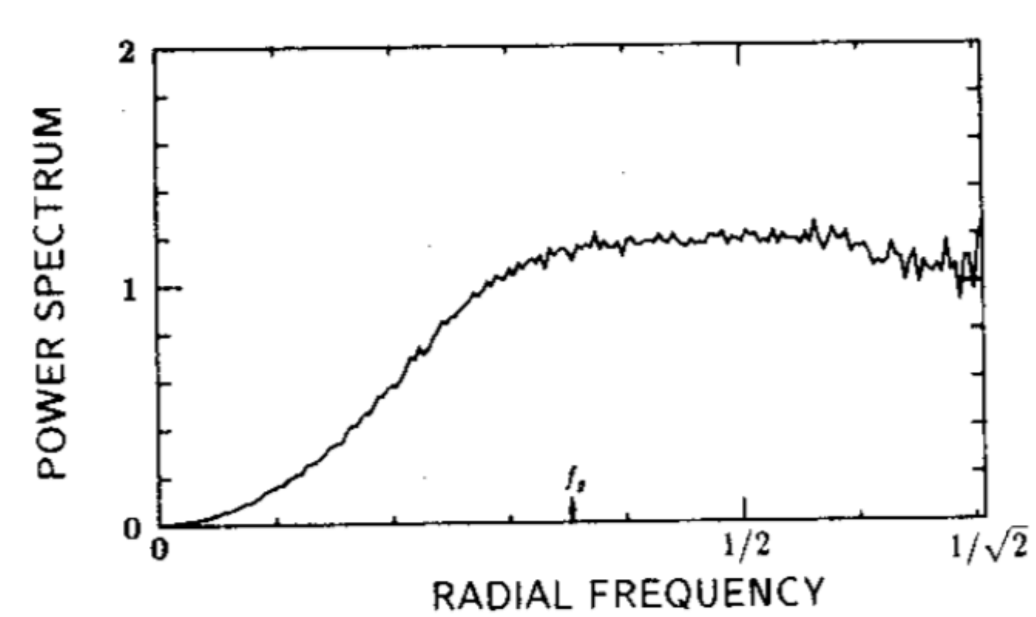
White



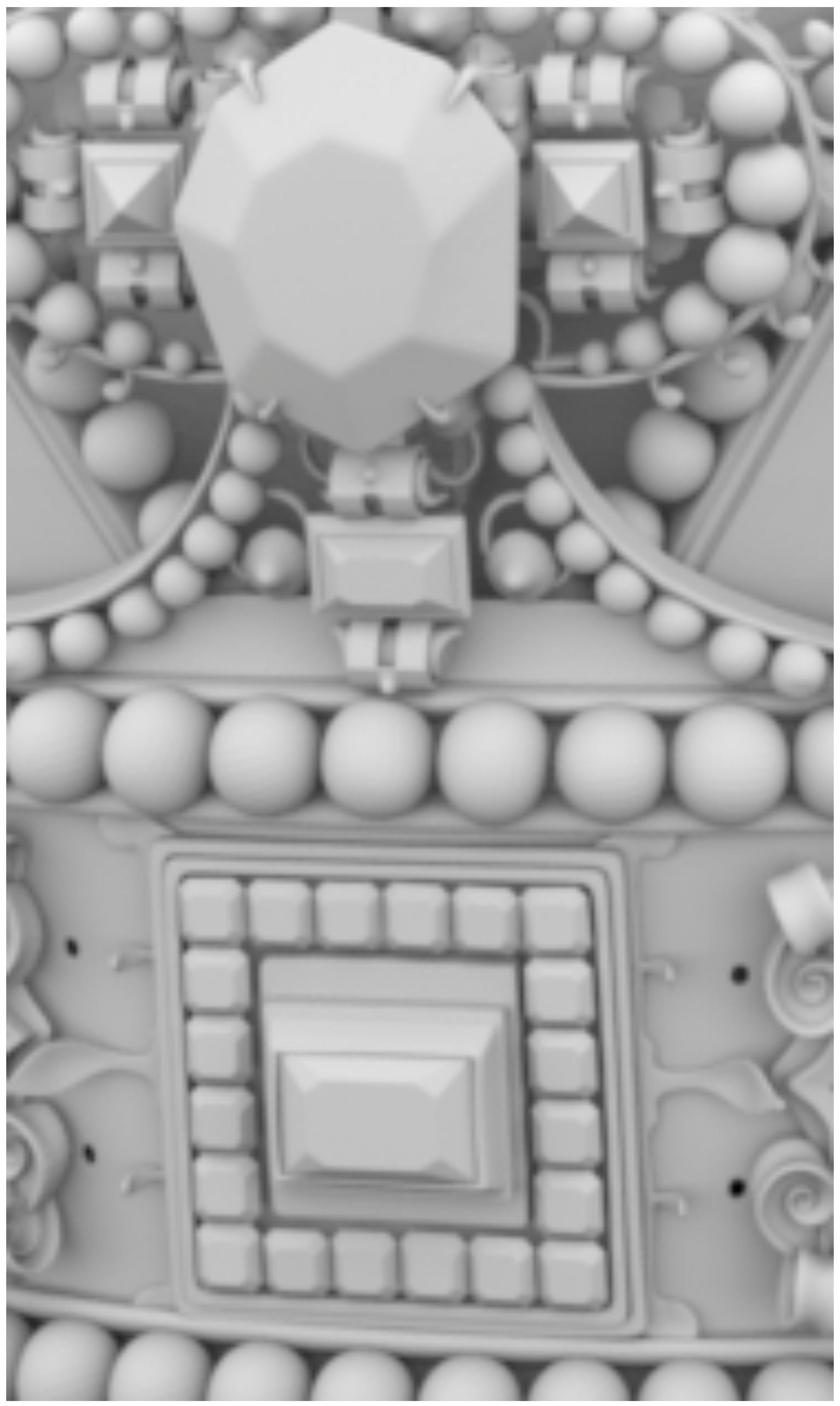


Blue

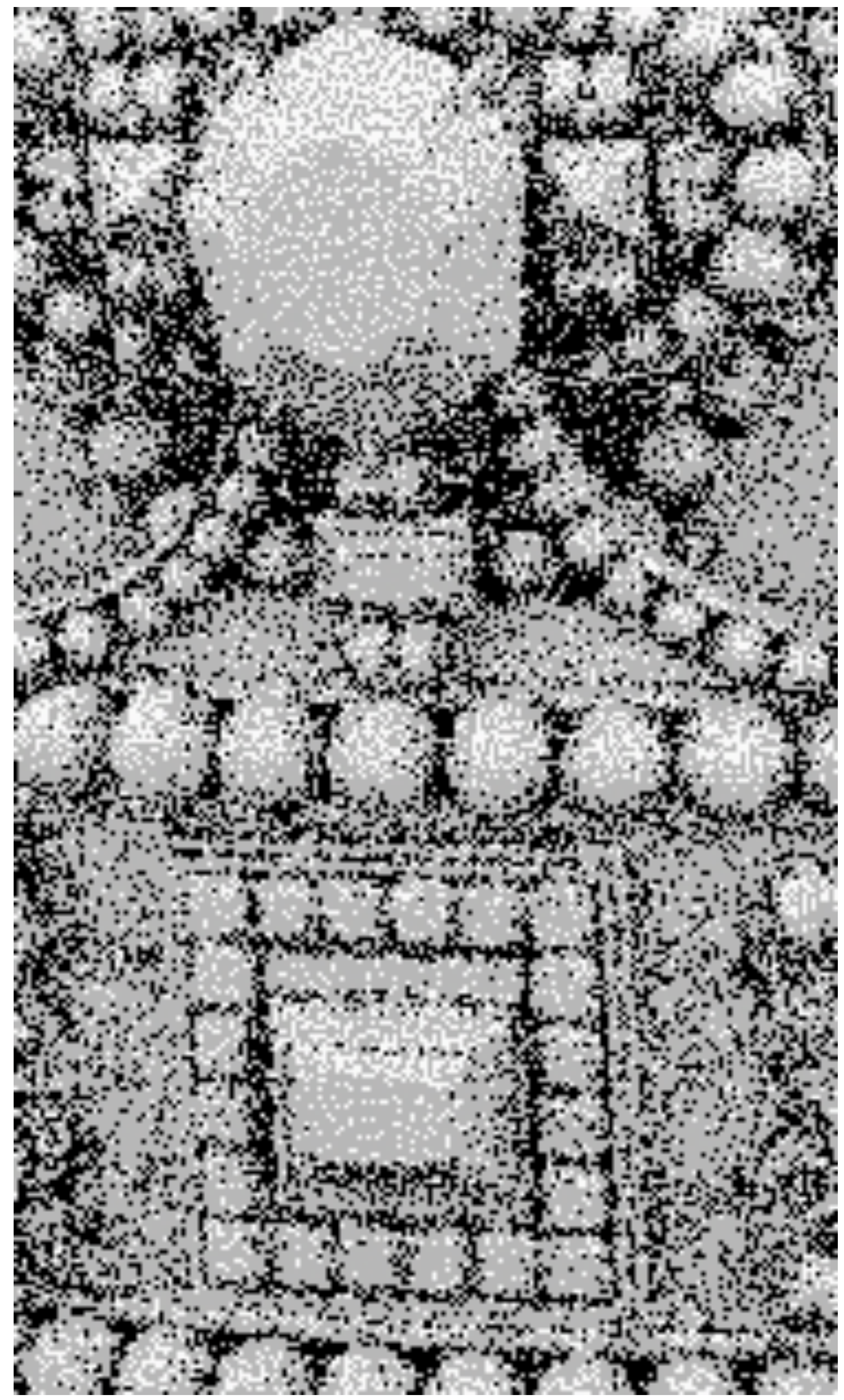




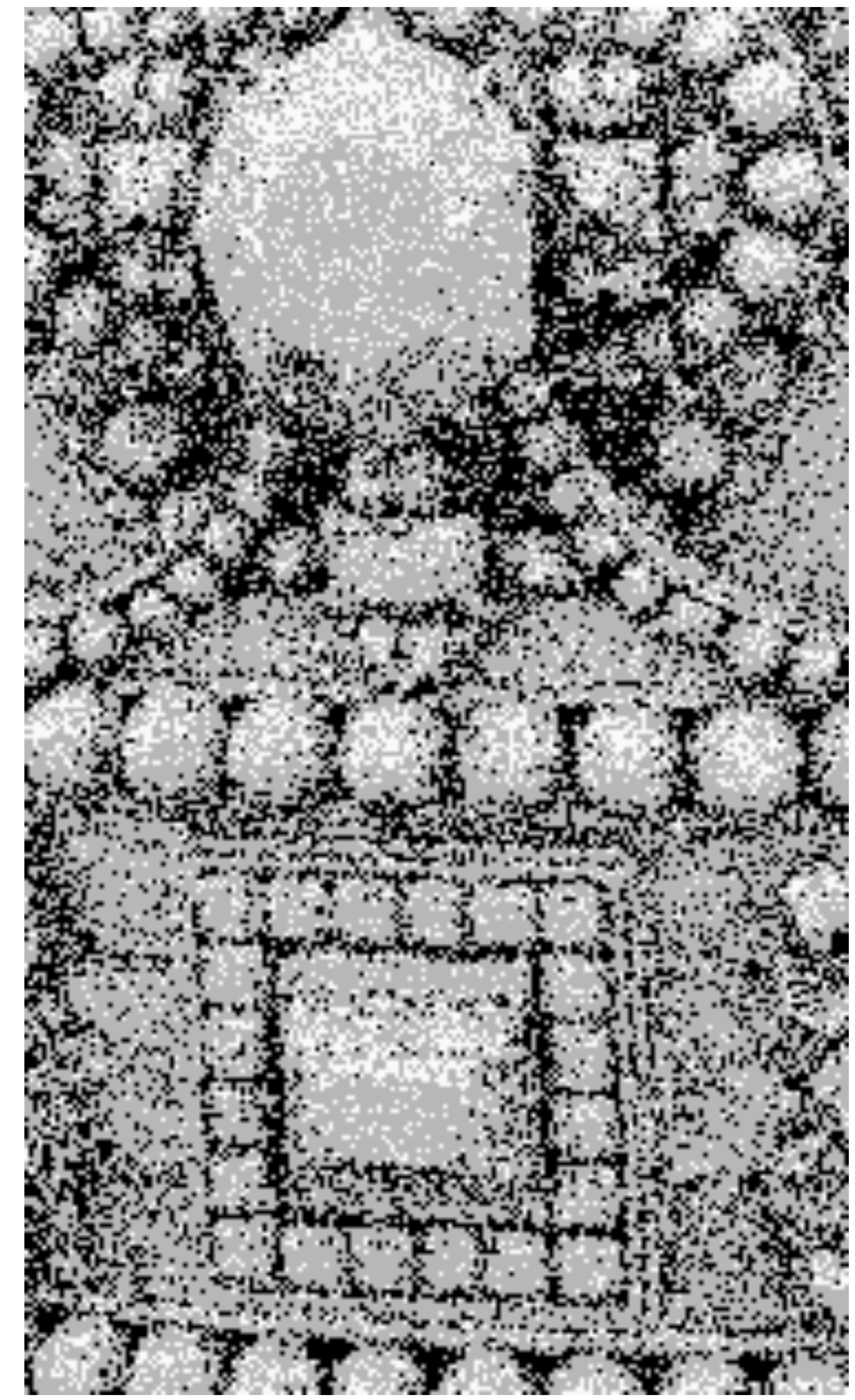
Ambient Occlusion, Revisited $\int_{\Omega} V(\omega)\, \cos\theta \, \mathrm{d}\omega$



Reference

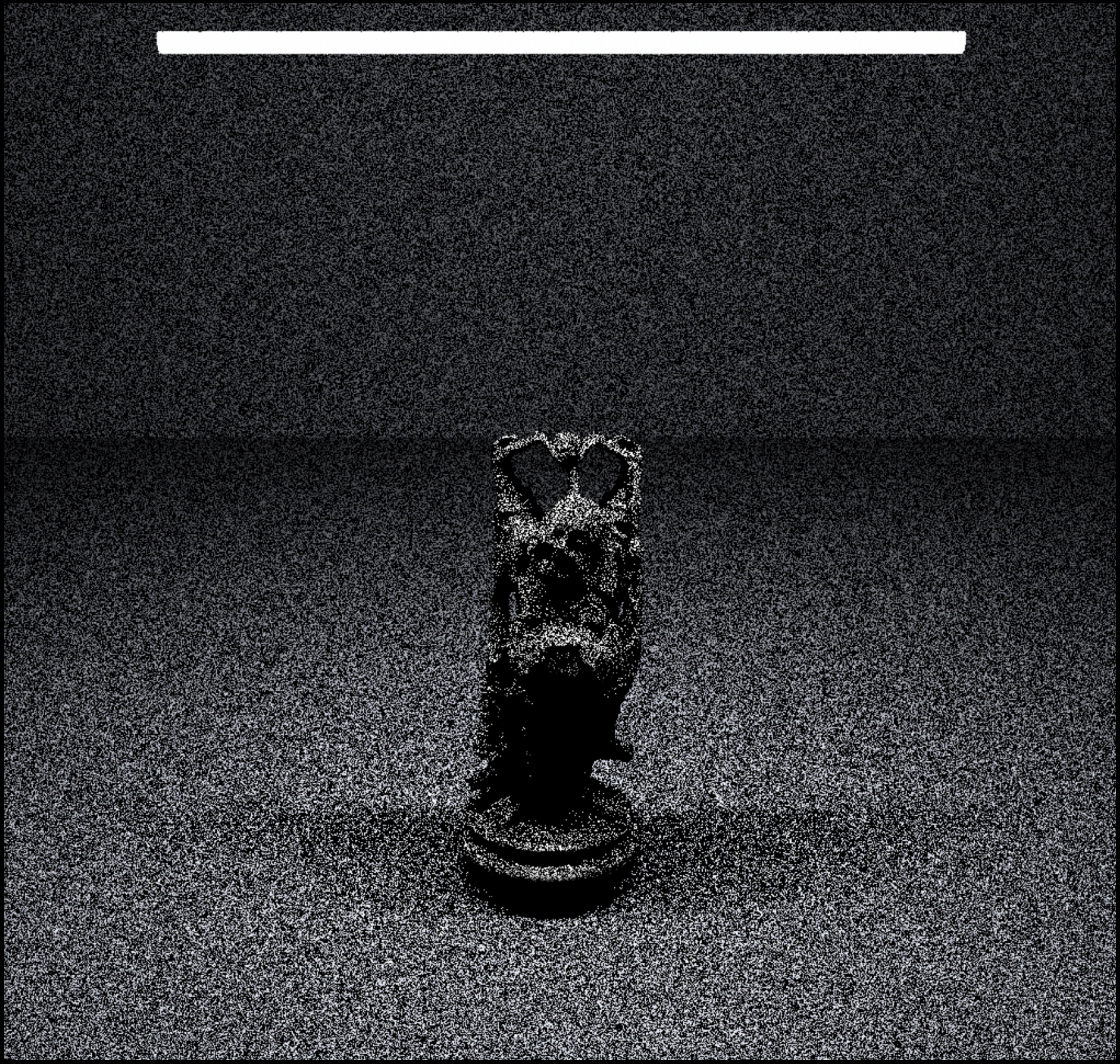


Blue Noise



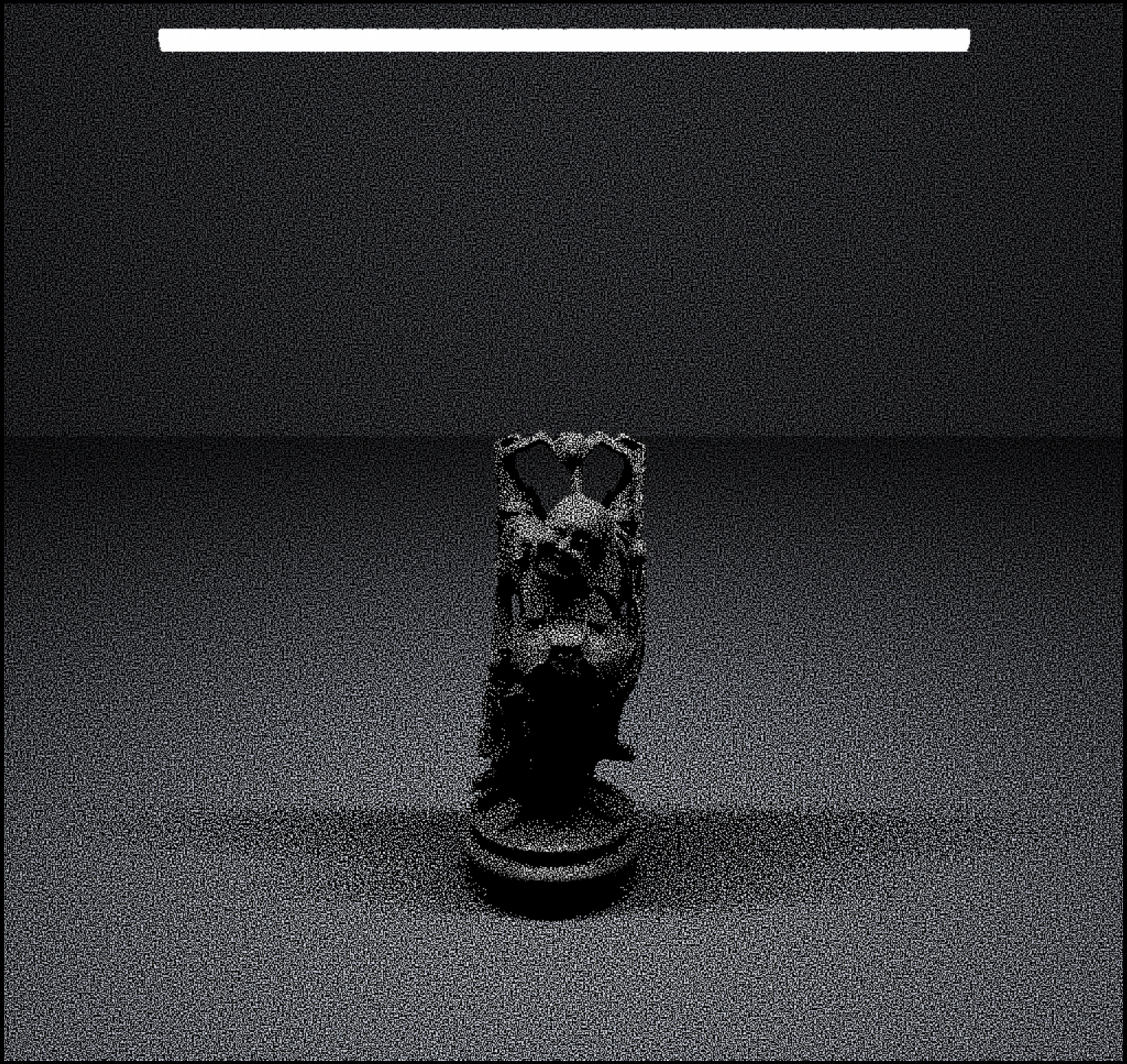
White Noise

(0,2)-sequence



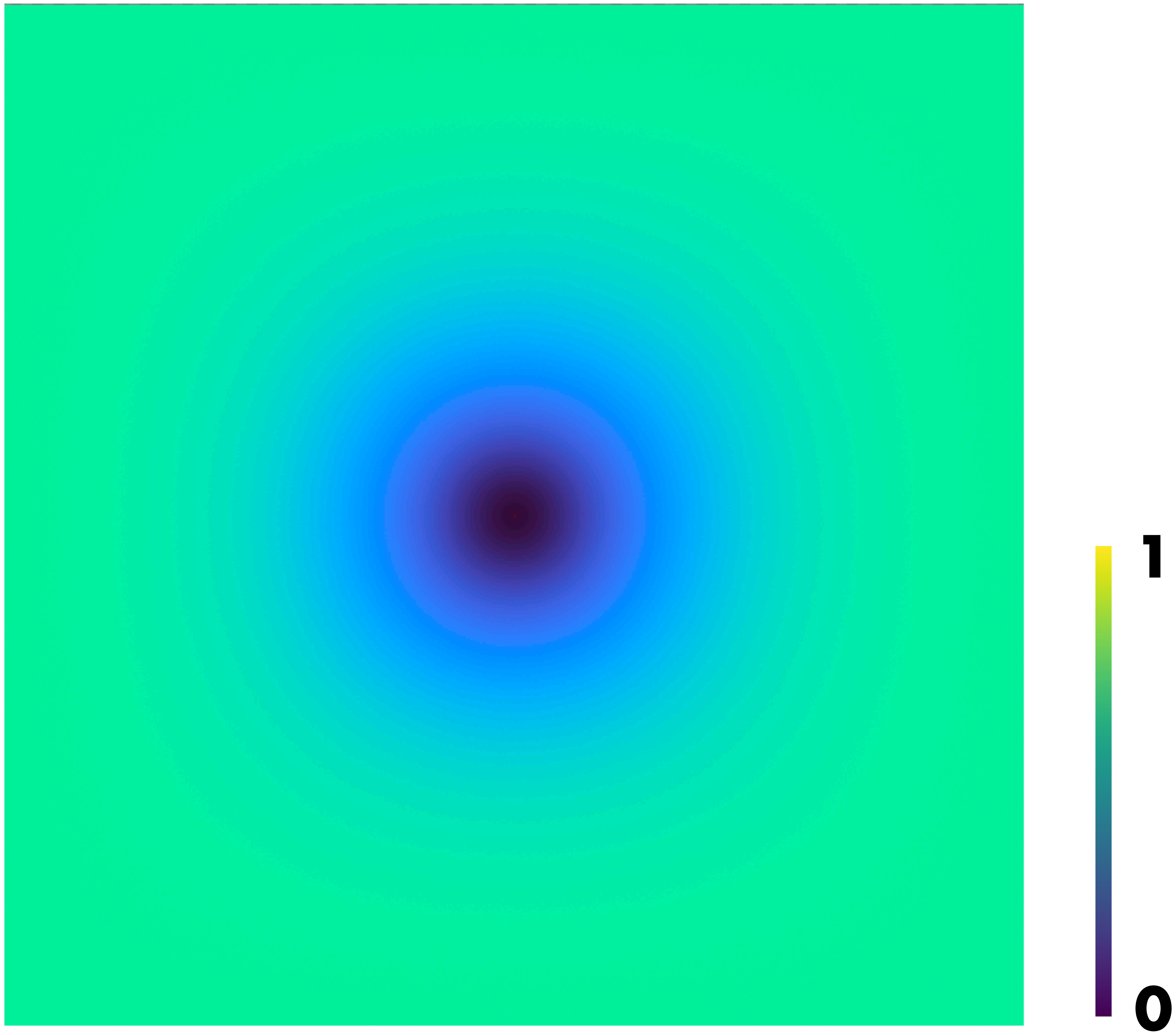
[Ahmed and Wonka 2020]

Blue Noise Sobol’

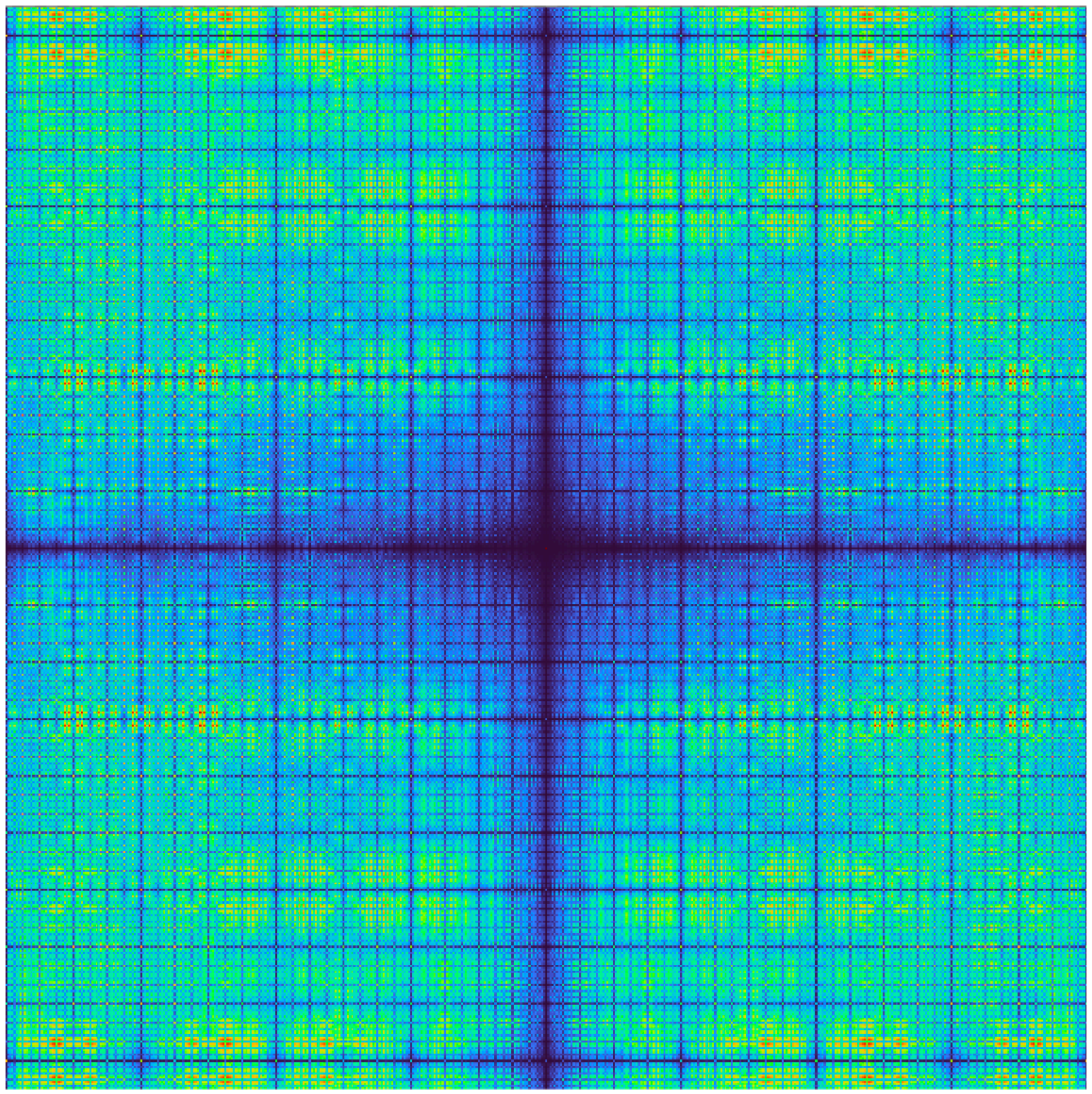


[Ahmed and Wonka 2020]

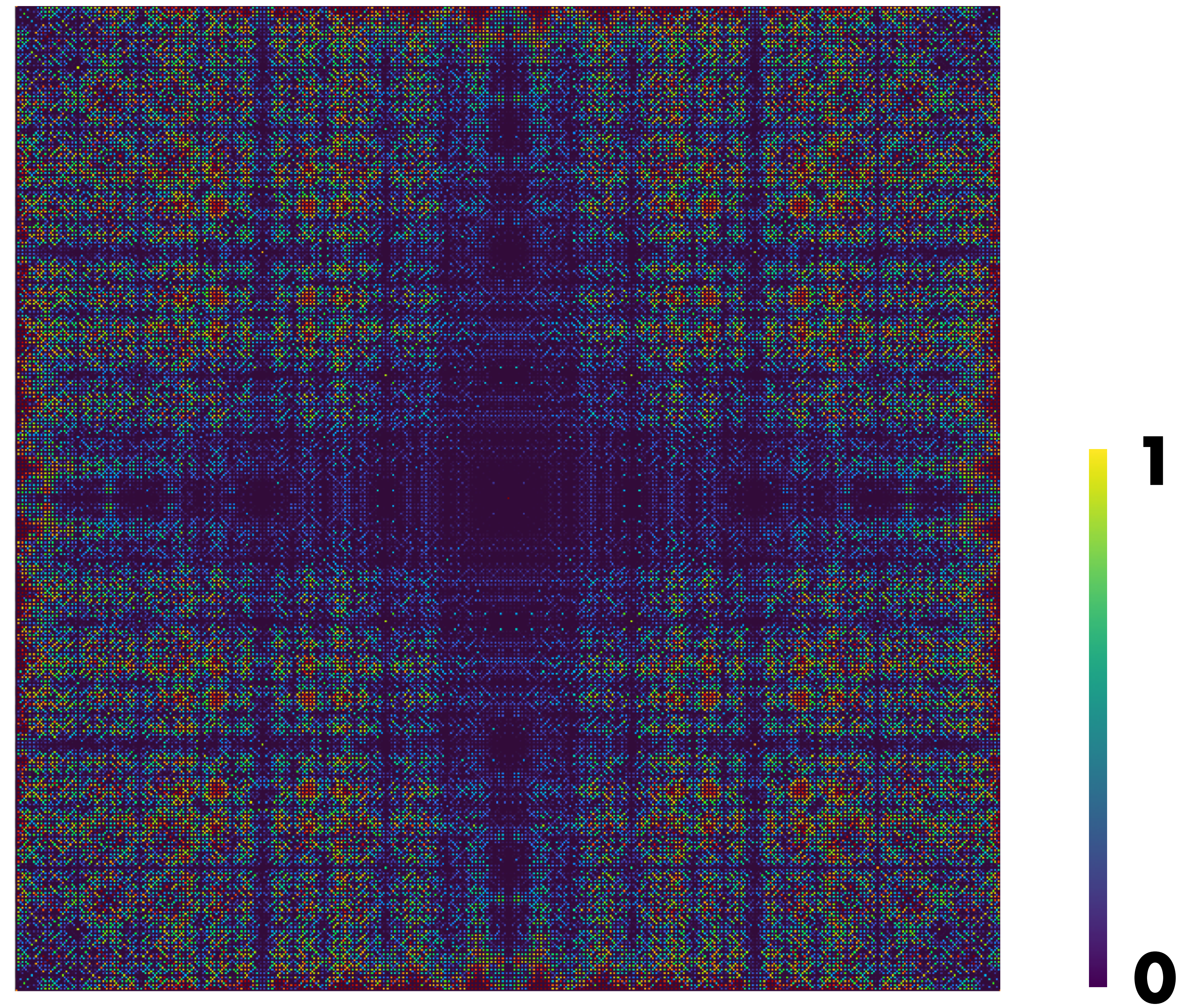
Jittered Power Spectrum



Low Discrepancy Power Spectra



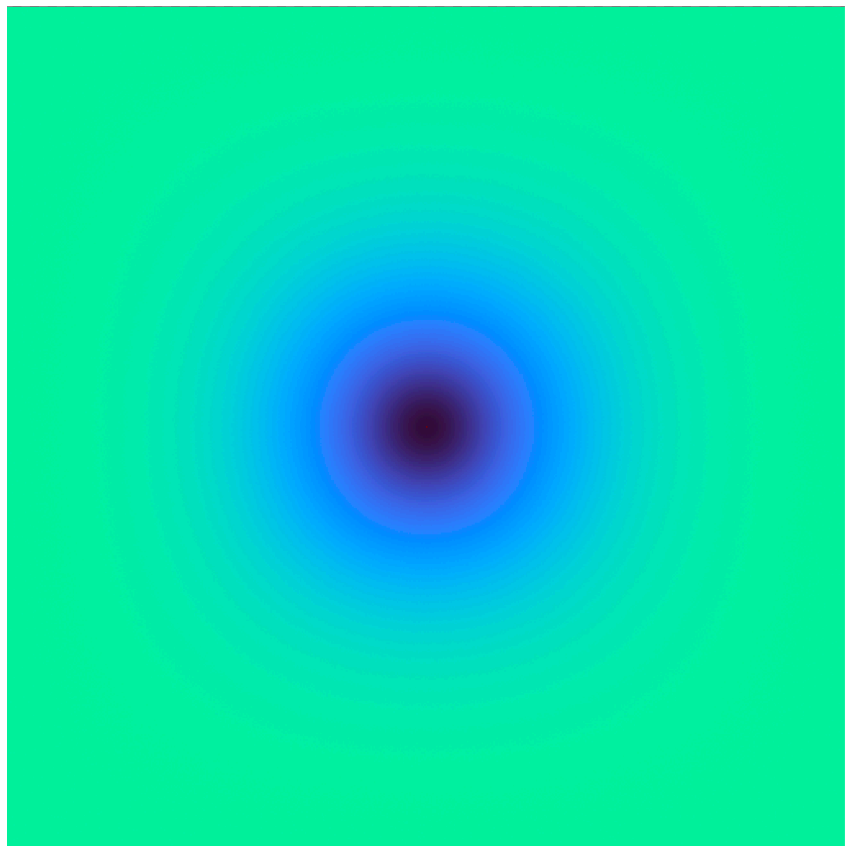
Halton



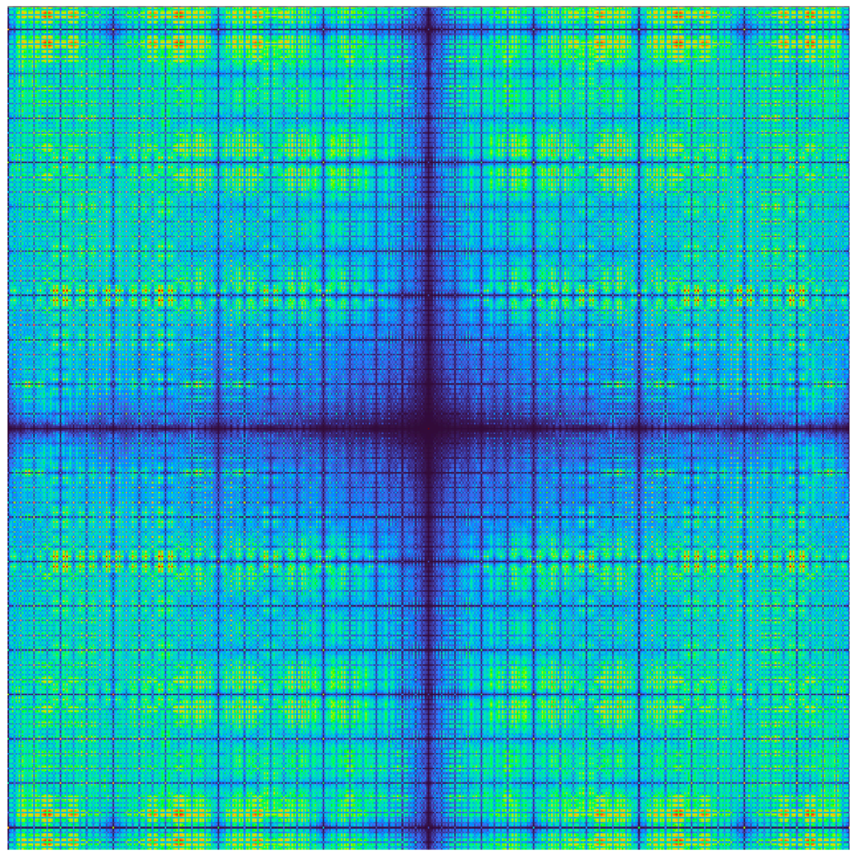
Sobol'

Power Spectra and Aliasing

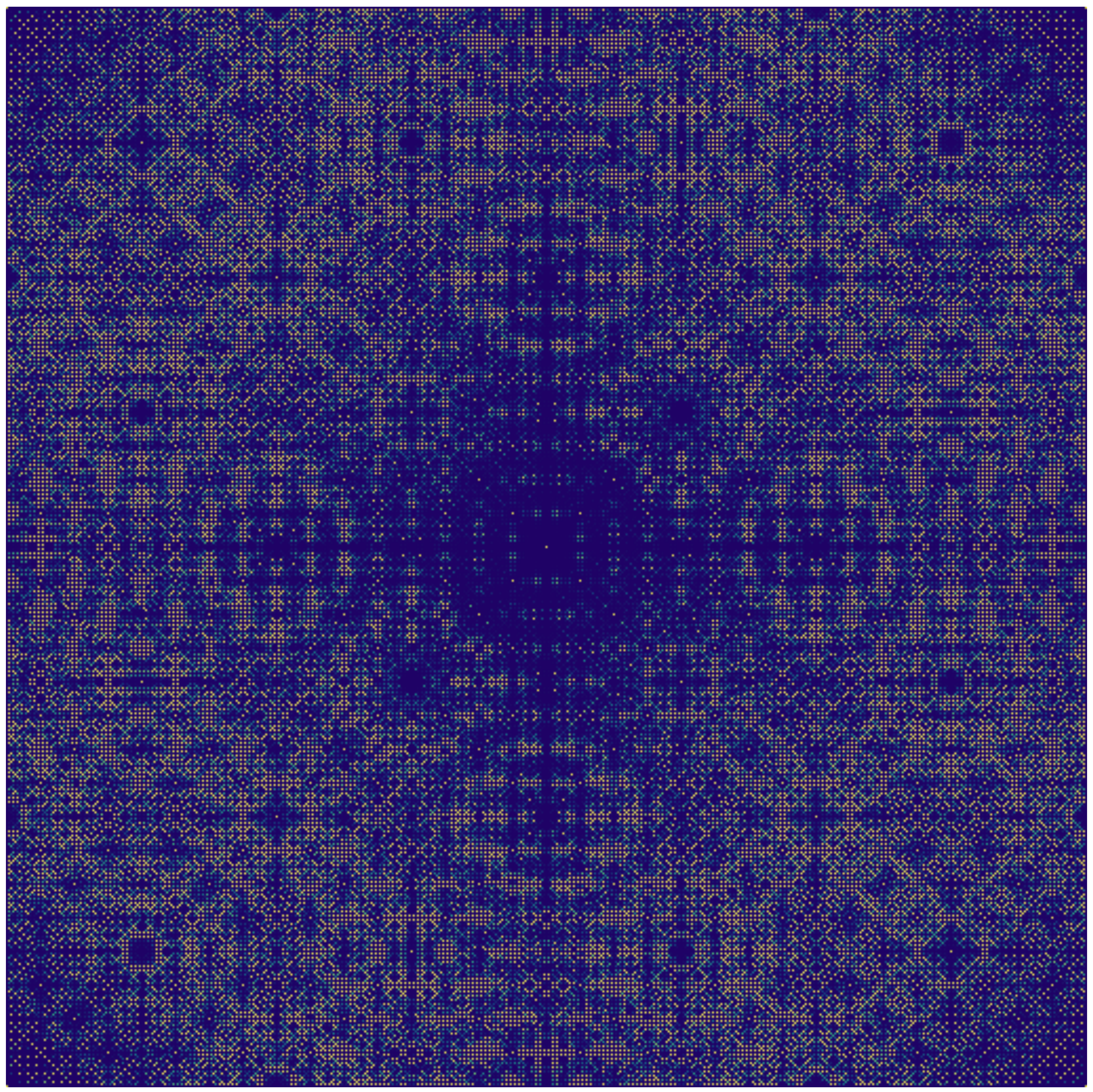
Jittered



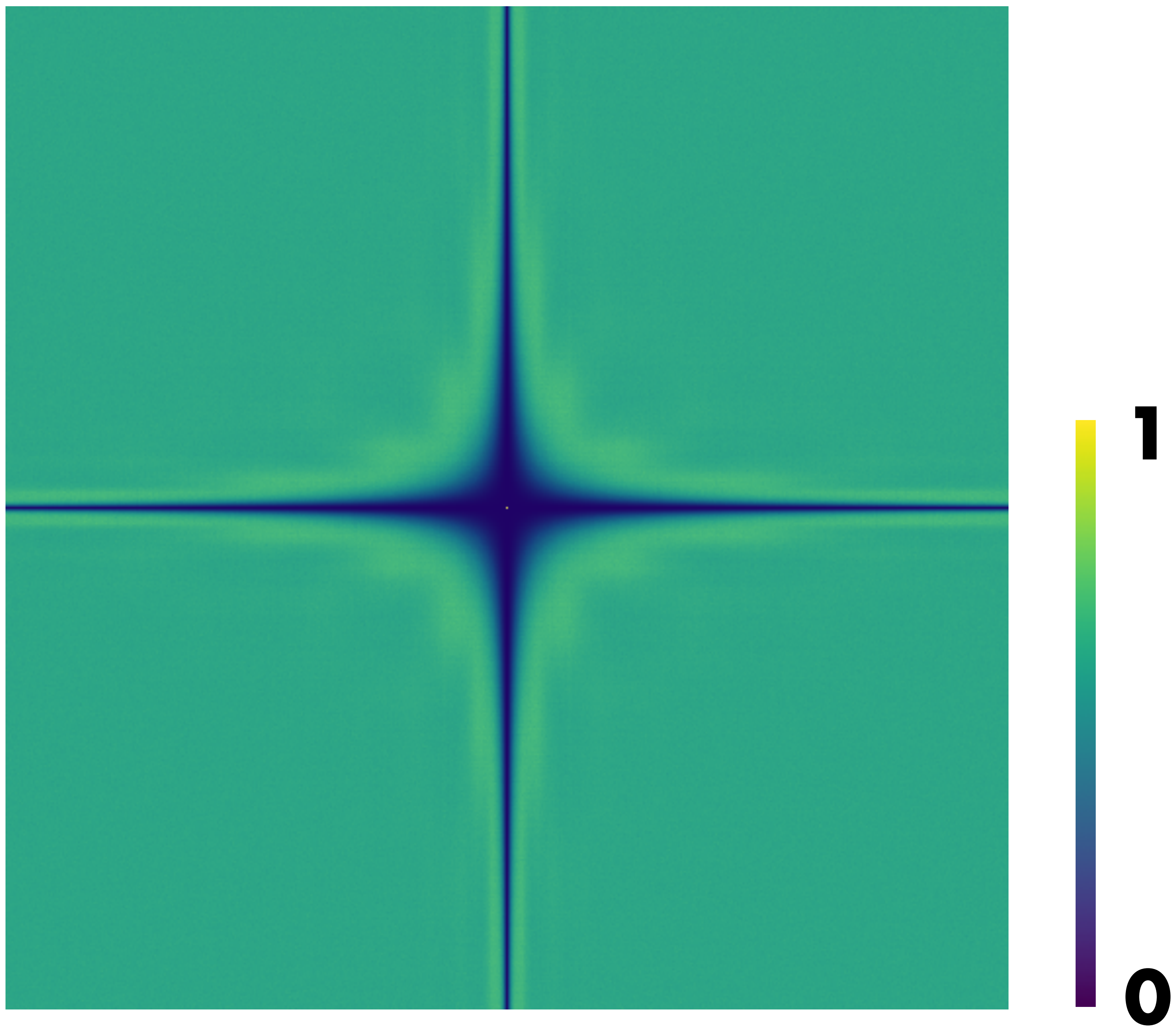
Halton



Randomized LD Power Spectra



Sobol′



Owen Scrambled Sobol′

Integration Error when Sampling

Integral

$$I(f) = \int f(x)\,dx$$

Sampled integral

Given samples x_i**, define** $s(x) = \frac{1}{N}\sum_{i=1}^{N}\delta(x - x_i)$

$$I_s(f) = \int f(x)s(x)\,dx = \frac{1}{N}\sum f(x_i)$$

Integration Error when Sampling

Sampled integral

$$I_s(f) = \int f(x)s(x)\,dx = \frac{1}{N}\sum f(x_i)$$

Since:

$$f(x)s(x) \leftrightarrow F(\omega) \oplus S(\omega)$$

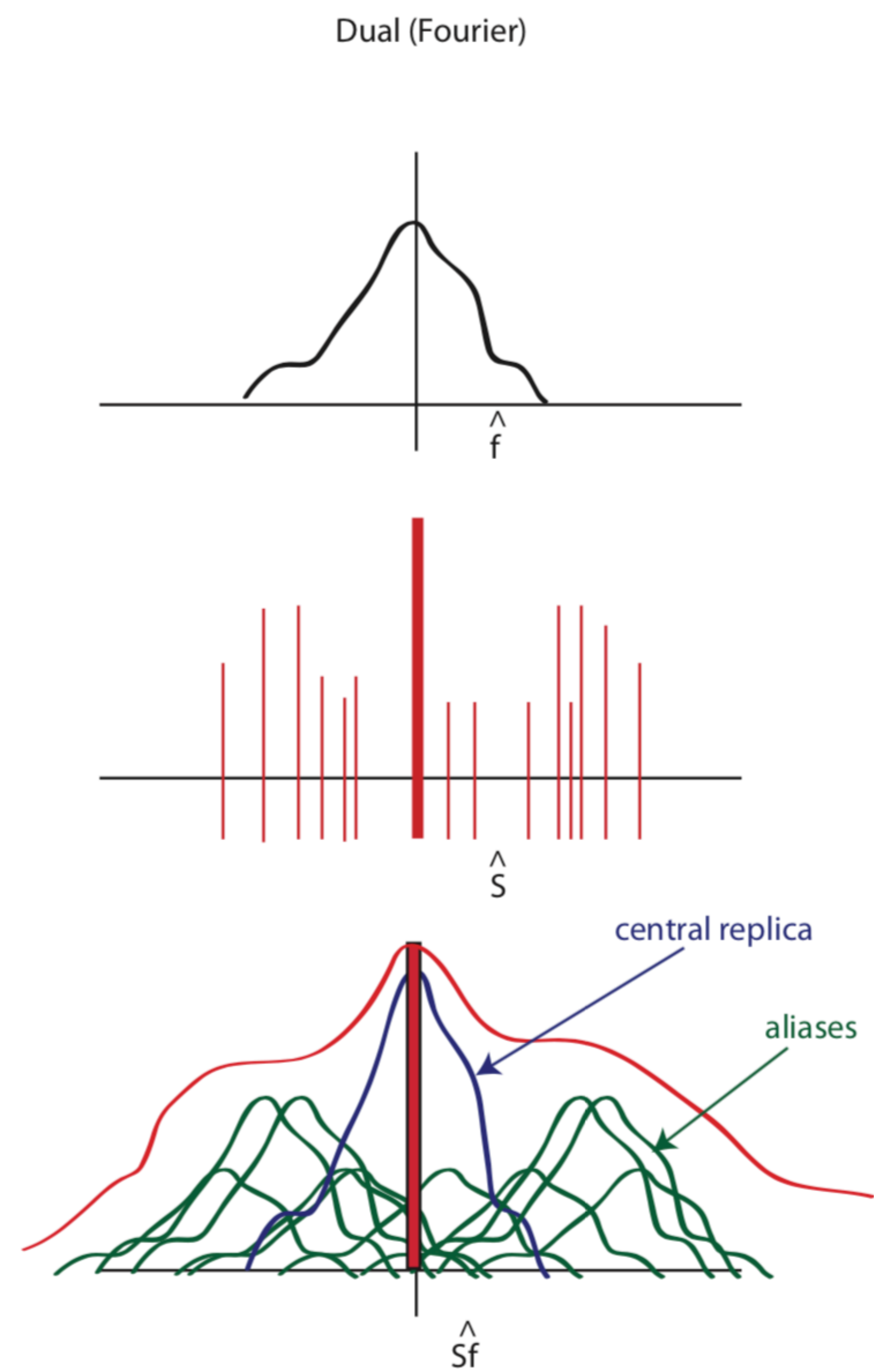
$$I(f) = \int f(x)\,dx = F(0)$$

$$I_s(f) = \int f(x)s(x)\,dx = F(\omega) \oplus S(\omega)|_{\omega=0}$$

Error

$$\Delta = F(0) - F(\omega) \oplus S(\omega)|_{\omega=0}$$

Integration in the Frequency Domain



[Durand 2011]

Variance Analysis

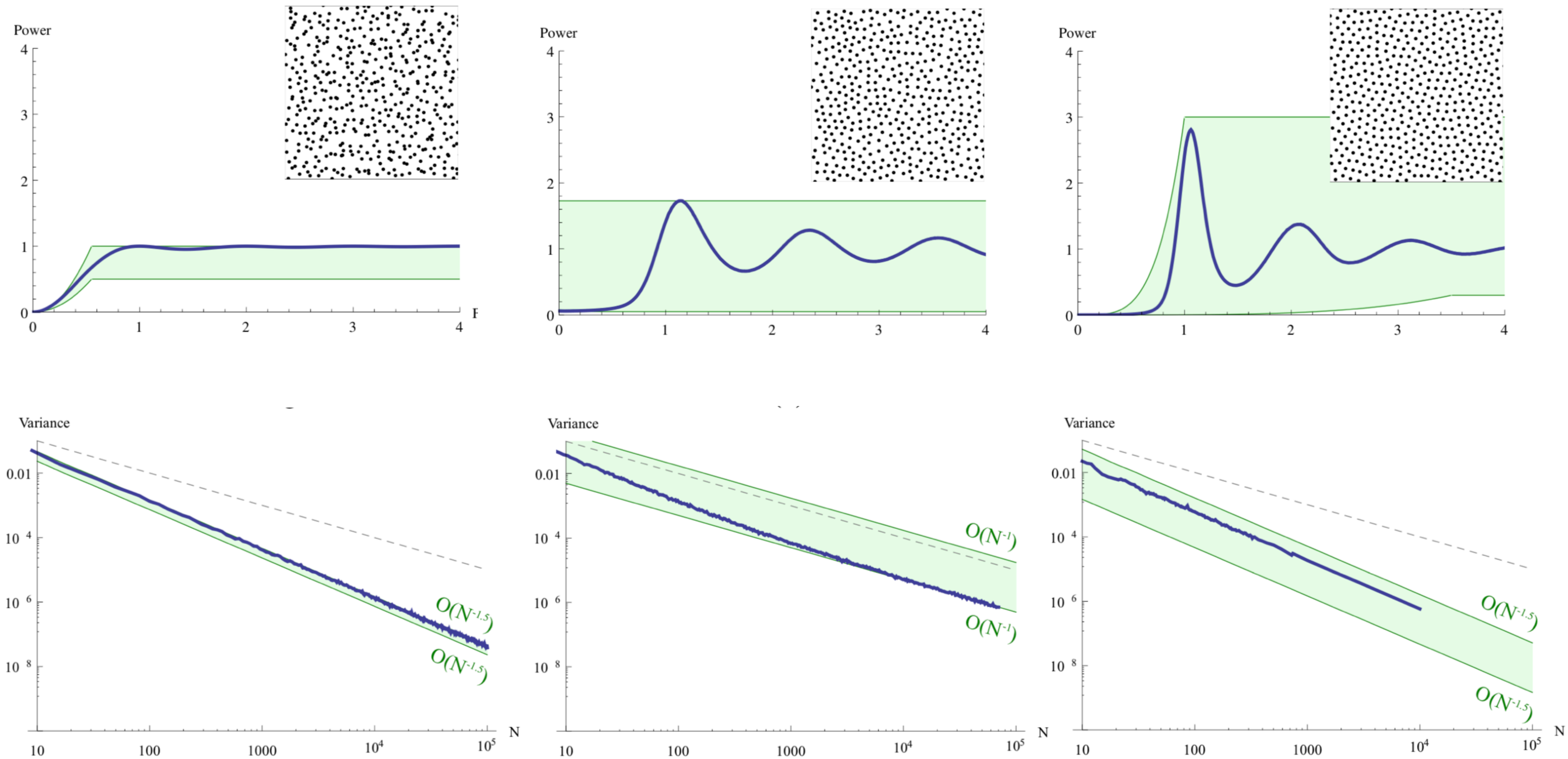
Recall error is: $\Delta = F(0) - F(\omega) \oplus S(\omega)|_{\omega=0}$

Can show that: $V \propto \int_{\Omega - \{0\}} P_F(\omega) P_S(\omega) \, \mathrm{d}\omega$

where $P_*(\omega)$ **is the power spectrum of the signal**

Power Spectra and Variance

Jittered Poisson Disk Blue Noise



[Pilleboue et al. 2015]