ADVANCED SAMPLING

Philipp Slusallek    Karol Myszkowski
Gurprit Singh

Realistic Image Synthesis SS2021
Recall: Monte Carlo Integration

\[ I = \int_{D} f(x) \, dx \]
Recall: Monte Carlo Integration

\[ I = \int_D f(x) \, dx \]
Recall: Monte Carlo Integration

\[ I = \int_D f(x) \, dx \]
Recall: Monte Carlo Integration

\[ I = \int_{D} f(x) \, dx \]

\[ \approx \int_{D} f(x) S(x) \, dx \]
Recall: Monte Carlo Integration

\[ I = \int_{D} f(x) \, dx \]
\[ \approx \int_{D} f(x) S(x) \, dx \]

\[ S(x) = \frac{1}{N} \sum_{k=1}^{N} \delta(x - x_k) \]
Recall: Monte Carlo Integration

\[ I = \int_{D} f(x) \, dx \]

\[ \approx \int_{D} f(x) S(x) \, dx \]

\[ S(x) = \frac{1}{N} \sum_{k=1}^{N} \delta(x - x_k) \]
Recall: Monte Carlo Integration

\[ I = \int_D f(x) \, dx \]
\[ \approx \int_D f(x) S(x) \, dx \]

\[ S(x) = \frac{1}{N} \sum_{k=1}^{N} \delta(x - x_k) \]

How to generate the locations \( x_k \)?
Independent Random Sampling

for (int k = 0; k < num; k++)
{
    samples(k).x = randf();
    samples(k).y = randf();
}
for (int k = 0; k < num; k++)
{
    samples(k).x = randf();
    samples(k).y = randf();
}

Independent Random Sampling
Independent Random Sampling

for (int k = 0; k < num; k++)
{
    samples(k).x = randf();
    samples(k).y = randf();
}

✔ Trivially extends to higher dimensions
Independent Random Sampling

for (int k = 0; k < num; k++)
{
    samples(k).x = randf();
    samples(k).y = randf();
}

✔ Trivially extends to higher dimensions
✔ Trivially progressive and memory-less
for (int k = 0; k < num; k++)
{
    samples(k).x = randf();
    samples(k).y = randf();
}

✔ Trivially extends to higher dimensions
✔ Trivially progressive and memory-less
✘ Big gaps
for (int k = 0; k < num; k++)
{
    samples(k).x = randf();
    samples(k).y = randf();
}

✔ Trivially extends to higher dimensions
✔ Trivially progressive and memory-less
✘ Big gaps
✘ Clumping
Recall: Fourier Theory

Input Image

Power Spectrum

Image courtesy: Laurent Belcour
Recall: Fourier Theory

Input Image  

Power Spectrum

Image courtesy: Laurent Belcour
Recall: Fourier theory

Fourier transform: \[ \hat{f}(\omega) = \int_{D} f(x) e^{-2 \pi i \omega x} \, dx \]
Recall: Fourier theory

Fourier transform: \[ \hat{f}(\vec{\omega}) = \int_D f(\vec{x}) e^{-2\pi i (\vec{\omega} \cdot \vec{x})} \, d\vec{x} \]
Recall: Fourier theory

Fourier transform: \( \hat{f}(\vec{\omega}) = \int_D f(\vec{x}) e^{-2 \pi i (\vec{\omega} \cdot \vec{x})} \, d\vec{x} \)

Sampling function: \( \hat{S}(\vec{\omega}) = \int_D S(\vec{x}) e^{-2 \pi i (\vec{\omega} \cdot \vec{x})} \, d\vec{x} \)
Recall: Fourier theory

Fourier transform: \[ \hat{f}(\vec{\omega}) = \int_D f(\vec{x}) e^{-2\pi i (\vec{\omega} \cdot \vec{x})} \, d\vec{x} \]

Sampling function: \[ \hat{S}(\vec{\omega}) = \int_D \frac{1}{N} \sum_{k=1}^{N} \delta(\|\vec{x} - \vec{x}_k\|) e^{-2\pi i (\vec{\omega} \cdot \vec{x})} \, d\vec{x} \]
Recall: Fourier theory

Fourier transform: \( \hat{f}(\vec{\omega}) = \int_{D} f(\vec{x}) e^{-2 \pi i (\vec{\omega} \cdot \vec{x})} \, d\vec{x} \)

Sampling function: \( \hat{S}(\vec{\omega}) = \int_{D} \frac{1}{N} \sum_{k=1}^{N} \delta(\|\vec{x} - \vec{x}_k\|) \, e^{-2 \pi i (\vec{\omega} \cdot \vec{x})} \, d\vec{x} \)

\[= \frac{1}{N} \sum_{k=1}^{N} e^{-2 \pi i (\vec{\omega} \cdot \vec{x}_k)} \]
Independent Random Sampling

\[
\frac{1}{N} \sum_{k=1}^{N} \delta(\| \vec{x} - \vec{x}_k \|)
\]

\[
\frac{1}{N} \sum_{k=1}^{N} e^{-2\pi \imath \left( \vec{\omega} \cdot \vec{x}_k \right)}
\]
Any sampling pattern with Blue noise characteristics is supposed to be well-distributed within the spatial domain without containing any regular structures. The term Blue noise was coined by Ulichney [3], who investigated a radially averaged power spectra of various sampling patterns. He advocated three important features for an ideal radial power spectrum: First, its peak should be at zero frequency, which is the center of gravity of the power spectrum. Second, the peak should be wide enough to cover the entire frequency range of interest. Third, the power spectrum should decay as fast as possible to ensure that the sampling process is as efficient as possible. For arbitrary dimensions, but due to the first dimension being a regular sampling, knowledge of the sequence is called the Hammersley sequence, which can create an even lower discrepancy point set compared to other sampling methods.

The corresponding Fourier expected power spectra and the corresponding radial mean of their expected power spectra are summarized in Figures 5.8. The corresponding samples are illustrated in Figure 5.6: Illustration of random and some stochastic grid-based sampling patterns with the corresponding power spectrum and radial mean.

\[
\frac{1}{N} \sum_{k=1}^{N} \delta(|\vec{x} - \vec{x}_k|)
\]

\[
\left| \frac{1}{N} \sum_{k=1}^{N} e^{-2\pi i \frac{\omega_x}{N} \cdot \vec{x}_k} \right|^2
\]
Independent Random Sampling

\[
\frac{1}{N} \sum_{k=1}^{N} \delta(\|\vec{x} - \vec{x}_k\|) = \left| \frac{1}{N} \sum_{k=1}^{N} e^{-2\pi i (\vec{\omega} \cdot \vec{x}_k)} \right|^2
\]
Independent Random Sampling

Many sample set realizations

\[ \bar{x}_y \]

Expected power spectrum

\[ \bar{\omega}_y \]

\[ \bar{x}_x \]

\[ \frac{1}{N} \sum_{k=1}^{N} \delta(|\bar{x} - \bar{x}_k|) \]

\[ \bar{\omega}_x \]

\[ \frac{1}{N} \sum_{k=1}^{N} e^{-2\pi i (\bar{\omega} \cdot \bar{x}_k)} \]
Independent Random Sampling

Many sample set realizations

\[
\frac{1}{N} \sum_{k=1}^{N} \delta(|\vec{x} - \vec{x}_k|)
\]

Expected power spectrum

\[
E \left[ \left| \frac{1}{N} \sum_{k=1}^{N} e^{-2\pi i (\vec{\omega} \cdot \vec{x}_k)} \right|^2 \right]
\]

Ulichney advocated three important features for an ideal radial power spectrum; First, its peak should be at

Any sampling pattern with Blue noise characteristics is supposed to be well distributed within the spatial domain without containing any regular structures. The term Blue noise was coined by
Independent Random Sampling

\[ \frac{1}{N} \sum_{k=1}^{N} \delta(|\vec{x} - \vec{x}_k|) \quad E \left[ \left| \frac{1}{N} \sum_{k=1}^{N} e^{-2\pi i (\vec{\omega} \cdot \vec{x}_k)} \right|^2 \right] \]
Independent Random Sampling

\[
\frac{1}{N} \sum_{k=1}^{N} \delta(|\vec{x} - \vec{x}_k|) \quad E \left[ \left( \frac{1}{N} \sum_{k=1}^{N} e^{-2\pi i (\vec{\omega} \cdot \vec{x}_k)} \right)^2 \right]
\]
Any sampling pattern with Blue noise characteristics is supposed to be well distributed within the spatial domain without containing any regular structures. The term Blue noise was coined by Ulichney [64] who investigated a radially averaged power spectra of various sampling patterns. He advocated three important features for an ideal radial power spectrum; First, its peak should be at DC, second, the DC peak should be minimized, and third, the power should be located near to 0.

64 points in 2D. The corresponding sampling power spectra for Halton and Hammersley samples for arbitrary dimensions, but due to the first dimension being a regular sampling, knowledge of the sequence is called the Hammersley sequence, which can create a even lower discrepancy point set than the Halton sampling.

Figure 5.6: Illustration of random and some stochastic grid-based sampling patterns with the corresponding Fourier expected power spectra and the corresponding radial mean of their expected power spectra.
procedure powerSpectrum(samples, spectrumWidth, spectrumHeight)
  int N = samples.size()
  for u: 0 → spectrumWidth{
    for v: 0 → spectrumHeight{
      double real = 0, imag = 0;

      // Further code here

    }
  }

  return power;
}
Source code: Power spectrum

```java
procedure powerSpectrum(samples, spectrumWidth, spectrumHeight)
    int N = samples.size()
    for u: 0 → spectrumWidth{
        for v: 0 → spectrumHeight{
            double real = 0, imag = 0;

            //compute the real and imaginary fourier coefficients
            for(int k=0;k<N;k++){

            }
        }
    }
    return power;
```
Source code: Power spectrum

```java
procedure powerSpectrum(samples, spectrumWidth, spectrumHeight)
    int N = samples.size()
    for u: 0 → spectrumWidth{
        for v: 0 → spectrumHeight{
            double real = 0, imag = 0;

            //compute the real and imaginary fourier coefficients
            for(int k=0;k<N;k++){
                real += cos(2 * π * (u * samples[k].x + v * samples[k].y));
                imag += sin(2 * π * (u * samples[k].x + v * samples[k].y));
            }
        }
    }
    return power;
```
procedure powerSpectrum(samples, spectrumWidth, spectrumHeight)
    int N = samples.size()
    for u: 0 → spectrumWidth{
        for v: 0 → spectrumHeight{
            double real = 0, imag = 0;

            //compute the real and imaginary fourier coefficients
            for(int k=0;k<N;k++){
                real += cos(2 * π * (u * samples[k].x + v * samples[k].y));
                imag += sin(2 * π * (u * samples[k].x + v * samples[k].y));
            }

            //power spectrum is the magnitude square value of the coefficients
            power[u * spectrumWidth + v] = (real*real + imag * imag) / N;
        }
    }
    return power;
Regular Sampling

for (uint i = 0; i < numX; i++)
    for (uint j = 0; j < numY; j++)
    {
        samples(i,j).x = (i + 0.5)/numX;
        samples(i,j).y = (j + 0.5)/numY;
    }
Regular Sampling

for (uint i = 0; i < numX; i++)
    for (uint j = 0; j < numY; j++)
    {
        samples(i,j).x = (i + 0.5)/numX;
        samples(i,j).y = (j + 0.5)/numY;
    }
Regular Sampling

for (uint i = 0; i < numX; i++)
    for (uint j = 0; j < numY; j++)
    {
        samples(i,j).x = (i + 0.5)/numX;
        samples(i,j).y = (j + 0.5)/numY;
    }

✔ Extends to higher dimensions, but…
Regular Sampling

for (uint i = 0; i < numX; i++)
    for (uint j = 0; j < numY; j++)
    {
        samples(i,j).x = (i + 0.5)/numX;
        samples(i,j).y = (j + 0.5)/numY;
    }

✔ Extends to higher dimensions, but...

✘ Curse of dimensionality
Regular Sampling

for (uint i = 0; i < numX; i++)
    for (uint j = 0; j < numY; j++)
    {
        samples(i,j).x = (i + 0.5)/numX;
        samples(i,j).y = (j + 0.5)/numY;
    }

✔ Extends to higher dimensions, but…

✘ Curse of dimensionality

✘ Aliasing
for (uint i = 0; i < numX; i++)
    for (uint j = 0; j < numY; j++)
    {
        samples(i,j).x = (i + 0.5)/numX;
        samples(i,j).y = (j + 0.5)/numY;
    }
for (uint i = 0; i < numX; i++)
    for (uint j = 0; j < numY; j++)
    {
        samples(i,j).x = (i + randf())/numX;
        samples(i,j).y = (j + randf())/numY;
    }
for (uint i = 0; i < numX; i++)
for (uint j = 0; j < numY; j++)
{
    samples(i,j).x = (i + randf())/numX;
    samples(i,j).y = (j + randf())/numY;
}

✔ Provably cannot increase variance
Jittered/Stratified Sampling

for (uint i = 0; i < numX; i++)
    for (uint j = 0; j < numY; j++)
    {
        samples(i,j).x = (i + randf())/numX;
        samples(i,j).y = (j + randf())/numY;
    }

✔ Provably cannot increase variance
✔ Extends to higher dimensions, but…
Jittered/Stratified Sampling

for (uint i = 0; i < numX; i++)
    for (uint j = 0; j < numY; j++)
    {
        samples(i,j).x = (i + randf())/numX;
        samples(i,j).y = (j + randf())/numY;
    }

✔ Provably cannot increase variance
✔ Extends to higher dimensions, but...
✘ Curse of dimensionality
Jittered/Stratified Sampling

for (uint i = 0; i < numX; i++)
    for (uint j = 0; j < numY; j++)
    {
        samples(i,j).x = (i + randf())/numX;
        samples(i,j).y = (j + randf())/numY;
    }

✔ Provably cannot increase variance
✔ Extends to higher dimensions, but…
✘ Curse of dimensionality
✘ Not progressive
Jittered Sampling

Samples | Expected power spectrum | Radial mean

Figure 5.6: Illustration of random and some stochastic grid-based sampling patterns with the corresponding Fourier expected power spectra and the corresponding radial mean of their expected power spectra.

5.3 Blue noise

Any sampling pattern with Blue noise characteristics is supposed to be well distributed within the spatial domain without containing any regular structures. The term Blue noise was coined by Ulichney [47], who investigated a radially averaged power spectrum of various sampling patterns. He advocated three important features for an ideal radial power spectrum; first, its peak should be at...
Chapter 5. Popular sampling patterns

- Independent Random Sampling
- Samples
- Expected power spectrum
- Radial mean

Figure 5.6: Illustration of random and some stochastic grid-based sampling patterns with the corresponding Fourier expected power spectra and the corresponding radial mean of their expected power spectra.

Due to the first dimension being a regular sampling, knowledge of the number of total samples is necessary. Figure 5.7 illustrates the Hammersley point set with 16 and 64 points in 2D. The corresponding sampling power spectra for Halton and Hammersley samples (first two components) are summarised in Figures 5.8.

5.3 Blue noise

Any sampling pattern with Blue noise characteristics is supposed to be well distributed within the spatial domain without containing any regular structures. The term Blue noise was coined by Ulichney [47], who investigated a radially averaged power spectra of various sampling patterns. He advocated three important features for an ideal radial power spectrum: First, its peak should be at zero frequency.
Monte Carlo (16 random samples)
Monte Carlo (16 jittered samples)
Stratifying in Higher Dimensions

Stratification requires $O(N^d)$ samples

- e.g. pixel (2D) + lens (2D) + time (1D) = 5D
Stratifying in Higher Dimensions

Stratification requires $O(N^d)$ samples

- e.g. pixel (2D) + lens (2D) + time (1D) = 5D
  - splitting 2 times in 5D = $2^5 = 32$ samples
  - splitting 3 times in 5D = $3^5 = 243$ samples!
Stratifying in Higher Dimensions

Stratification requires \( O(N^d) \) samples

- e.g. pixel (2D) + lens (2D) + time (1D) = 5D
  - splitting 2 times in 5D = \( 2^5 = 32 \) samples
  - splitting 3 times in 5D = \( 3^5 = 243 \) samples!

Inconvenient for large \( d \)

- cannot select sample count with fine granularity
Uncorrelated Jitter [Cook et al. 84]
Uncorrelated Jitter [Cook et al. 84]

Compute stratified samples in sub-dimensions
Uncorrelated Jitter [Cook et al. 84]

Compute stratified samples in sub-dimensions

- 2D jittered (x,y) for pixel

\[
\begin{array}{cc}
  x_1, y_1 & x_2, y_2 \\
  x_3, y_3 & x_4, y_4 \\
\end{array}
\]
Uncorrelated Jitter [Cook et al. 84]

Compute stratified samples in sub-dimensions

- 2D jittered \((x,y)\) for pixel

- 2D jittered \((u,v)\) for lens
Uncorrelated Jitter [Cook et al. 84]

Compute stratified samples in sub-dimensions

- 2D jittered \((x,y)\) for pixel
- 2D jittered \((u,v)\) for lens
- 1D jittered \((t)\) for time
Uncorrelated Jitter [Cook et al. 84]

Compute stratified samples in sub-dimensions

- 2D jittered (x,y) for pixel
- 2D jittered (u,v) for lens
- 1D jittered (t) for time
- combine dimensions in random order
Depth of Field (4D)

Reference

Random Sampling

Uncorrelated Jitter

Image source: PBRTe2 [Pharr & Humphreys 2010]
Uncorrelated Jitter ➔ Latin Hypercube

Stratify samples in each dimension separately
Uncorrelated Jitter \(\Rightarrow\) Latin Hypercube

Stratify samples in each dimension separately

- for 5D: 5 separate 1D jittered point sets

\[
\begin{array}{cccc}
    x & \bullet & \bullet & \bullet & \bullet \\
      \hline
    x1 & x2 & x3 & x4 \\
\end{array}
\]

\[
\begin{array}{cccc}
    y & \bullet & \bullet & \bullet & \bullet \\
      \hline
    y1 & y2 & y3 & y4 \\
\end{array}
\]

\[
\begin{array}{cccc}
    u & \bullet & \bullet & \bullet & \bullet \\
      \hline
    u1 & u2 & u3 & u4 \\
\end{array}
\]

\[
\begin{array}{cccc}
    v & \bullet & \bullet & \bullet & \bullet \\
      \hline
    v1 & v2 & v3 & v4 \\
\end{array}
\]

\[
\begin{array}{cccc}
    t & \bullet & \bullet & \bullet & \bullet \\
      \hline
    t1 & t2 & t3 & t4 \\
\end{array}
\]
Uncorrelated Jitter $\rightarrow$ Latin Hypercube

Stratify samples in each dimension separately

- for 5D: 5 separate 1D jittered point sets

- combine dimensions in random order

<table>
<thead>
<tr>
<th>x</th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>y1</td>
<td>y2</td>
<td>y3</td>
<td>y4</td>
</tr>
<tr>
<td>u</td>
<td>u1</td>
<td>u2</td>
<td>u3</td>
<td>u4</td>
</tr>
<tr>
<td>v</td>
<td>v1</td>
<td>v2</td>
<td>v3</td>
<td>v4</td>
</tr>
<tr>
<td>t</td>
<td>t1</td>
<td>t2</td>
<td>t3</td>
<td>t4</td>
</tr>
</tbody>
</table>
Uncorrelated Jitter $\Rightarrow$ Latin Hypercube

Stratify samples in each dimension separately

- for 5D: 5 separate 1D jittered point sets
- combine dimensions in random order

Shuffle order
N-Rooks = 2D Latin Hypercube [Shirley 91]

Stratify samples in each dimension separately

- for **2D**: 2 separate 1D jittered point sets
- combine dimensions in random order

\[
\begin{array}{cccc}
  \bullet & \bullet & \bullet & \bullet \\
  x1 & x2 & x3 & x4 \\
\end{array}
\]

\[
\begin{array}{cccc}
  \bullet & \bullet & \bullet & \bullet \\
  y4 & y2 & y1 & y3 \\
\end{array}
\]
Latin Hypercube (N-Rooks) Sampling

[Shirley 91]
Latin Hypercube (N-Rooks) Sampling

// initialize the diagonal
for (uint d = 0; d < numDimensions; d++)
    for (uint i = 0; i < numS; i++)
        samples(d,i) = (i + randf())/numS;

// shuffle each dimension independently
for (uint d = 0; d < numDimensions; d++)
    shuffle(samples(d,:));
Latin Hypercube (N-Rooks) Sampling

// initialize the diagonal
for (uint d = 0; d < numDimensions; d++)
  for (uint i = 0; i < numS; i++)
    samples(d,i) = (i + randf())/numS;

// shuffle each dimension independently
for (uint d = 0; d < numDimensions; d++)
  shuffle(samples(d,:));
// initialize the diagonal
for (uint d = 0; d < numDimensions; d++)
    for (uint i = 0; i < numS; i++)
        samples(d,i) = (i + randf())/numS;

// shuffle each dimension independently
for (uint d = 0; d < numDimensions; d++)
    shuffle(samples(d,:));
// initialize the diagonal
for (uint d = 0; d < numDimensions; d++)
  for (uint i = 0; i < numS; i++)
    samples(d,i) = (i + randf())/numS;

// shuffle each dimension independently
for (uint d = 0; d < numDimensions; d++)
  shuffle(samples(d,:));
Latin Hypercube (N-Rooks) Sampling

// initialize the diagonal
for (uint d = 0; d < numDimensions; d++)
    for (uint i = 0; i < numS; i++)
        samples(d,i) = (i + randf())/numS;

// shuffle each dimension independently
for (uint d = 0; d < numDimensions; d++)
    shuffle(samples(d,:));
Latin Hypercube (N-Rooks) Sampling

// initialize the diagonal
for (uint d = 0; d < numDimensions; d++)
    for (uint i = 0; i < numS; i++)
        samples(d,i) = (i + randf())/numS;

// shuffle each dimension independently
for (uint d = 0; d < numDimensions; d++)
    shuffle(samples(d,:));
// initialize the diagonal
for (uint d = 0; d < numDimensions; d++)
    for (uint i = 0; i < numS; i++)
        samples(d, i) = (i + randf())/numS;

// shuffle each dimension independently
for (uint d = 0; d < numDimensions; d++)
    shuffle(samples(d,:));
Latin Hypercube (N-Rooks) Sampling

// initialize the diagonal
for (uint d = 0; d < numDimensions; d++)
    for (uint i = 0; i < numS; i++)
        samples(d,i) = (i + randf())/numS;

// shuffle each dimension independently
for (uint d = 0; d < numDimensions; d++)
    shuffle(samples(d,:));
Latin Hypercube (N-Rooks) Sampling

// initialize the diagonal
for (uint d = 0; d < numDimensions; d++)
    for (uint i = 0; i < numS; i++)
        samples(d,i) = (i + randf())/numS;

// shuffle each dimension independently
for (uint d = 0; d < numDimensions; d++)
    shuffle(samples(d,:));
// initialize the diagonal
for (uint d = 0; d < numDimensions; d++)
    for (uint i = 0; i < numS; i++)
        samples(d,i) = (i + randf())/numS;

// shuffle each dimension independently
for (uint d = 0; d < numDimensions; d++)
    shuffle(samples(d,:));
Latin Hypercube (N-Rooks) Sampling
Latin Hypercube (N-Rooks) Sampling
Latin Hypercube (N-Rooks) Sampling
Latin Hypercube (N-Rooks) Sampling

Evenly distributed in each individual dimension
Latin Hypercube (N-Rooks) Sampling

Unevenly distributed in n-dimensions

Evenly distributed in each individual dimension
N-Rooks Sampling

Samples | Expected power spectrum | Radial mean

Figure 5.6: Illustration of random and some stochastic grid-based sampling patterns with the corresponding Fourier expected power spectra and the corresponding radial mean of their expected power spectra.

Any sampling pattern with Blue noise characteristics is supposed to be well distributed within the spatial domain without containing any regular structures. The term Blue noise was coined by Ulichney [47], who investigated a radially averaged power spectra of various sampling patterns. He advocated three important features for an ideal radial power spectrum: First, its peak should be at

5.3 Blue noise

sequence is called the Hammersley sequence, which can create an even lower discrepancy point set for arbitrary dimensions, but due to the first dimension being a regular sampling, knowledge of the number of total samples is necessary. Figure 5.7 illustrates the Hammersley point set with 16 and 64 points in 2D. The corresponding sampling power spectra for Halton and Hammersley samples (first two components) are summarised in Figures 5.8.
Multi-Jittered Sampling


- combine N-Rooks and Jittered stratification constraints
Multi-Jittered Sampling
Multi-Jittered Sampling

// initialize
float cellSize = 1.0 / (resX*resY);
for (uint i = 0; i < resX; i++)
    for (uint j = 0; j < resY; j++)
    {
        samples(i,j).x = i/resX + (j+randf()) / (resX*resY);
        samples(i,j).y = j/resY + (i+randf()) / (resX*resY);
    }

// shuffle x coordinates within each column of cells
for (uint i = 0; i < resX; i++)
    for (uint j = resY-1; j >= 1; j--)
        swap(samples(i, j).x, samples(i, randi(0, j)).x);

// shuffle y coordinates within each row of cells
for (unsigned j = 0; j < resY; j++)
    for (unsigned i = resX-1; i >= 1; i--)
        swap(samples(i, j).y, samples(randi(0, i), j).y);
Multi-Jittered Sampling

Initialize
Multi-Jittered Sampling

Shuffle x-coords
Multi-Jittered Sampling

Shuffle x-coords
Multi-Jittered Sampling

Shuffle x-coords
Multi-Jittered Sampling

Shuffle x-coords
Multi-Jittered Sampling

Shuffle x-coords
Multi-Jittered Sampling
Multi-Jittered Sampling

Shuffle y-coords
Multi-Jittered Sampling

Shuffle y-coords
Multi-Jittered Sampling

Shuffle y-coords
Multi-Jittered Sampling

Shuffle y-coords
Multi-Jittered Sampling

Shuffle y-coords
Multi-Jittered Sampling (Projections)
Multi-Jittered Sampling (Projections)
Multi-Jittered Sampling (Projections)
Multi-Jittered Sampling (Projections)
Multi-Jittered Sampling (Projections)

Evenly distributed in each individual dimension
Multi-Jittered Sampling (Projections)

Evenly distributed in 2D!

Evenly distributed in each individual dimension
Multi-Jittered Sampling

Samples | Expected power spectrum | Radial mean
--- | --- | ---

Figure 5.6: Illustration of random and some stochastic grid-based sampling patterns with the corresponding Fourier expected power spectra and the corresponding radial mean of their expected power spectra.

5.3 Blue noise

Any sampling pattern with Blue noise characteristics is supposed to be well distributed within the spatial domain without containing any regular structures. The term Blue noise was coined by Ulichney [47], who investigated a radially averaged power spectra of various sampling patterns. He advocated three important features for an ideal radial power spectrum: First, its peak should be at...
N-Rooks Sampling

Samples | Expected power spectrum | Radial mean
---|---|---

Any sampling pattern with Blue noise characteristics is supposed to be well distributed within the spatial domain without containing any regular structures. The term Blue noise was coined by Ulichney [47], who investigated a radially averaged power spectra of various sampling patterns. He advocated three important features for an ideal radial power spectrum; First, its peak should be at

Figure 5.6: Illustration of random and some stochastic grid-based sampling patterns with the corresponding Fourier expected power spectra and the corresponding radial mean of their expected power spectra.

5.3 Blue noise

Realistic Image Synthesis SS2021
Jittered Sampling

Samples  Expected power spectrum  Radial mean

Figure 5.6: Illustration of random and some stochastic grid-based sampling patterns with the corresponding Fourier expected power spectra and the corresponding radial mean of their expected power spectra.

5.3 Blue noise

Any sampling pattern with Blue noise characteristics is supposed to be well distributed within the spatial domain without containing any regular structures. The term Blue noise was coined by Ulichney [47], who investigated a radially averaged power spectra of various sampling patterns. He advocated three important features for an ideal radial power spectrum; First, its peak should be at
Poisson-Disk/Blue-Noise Sampling

Enforce a minimum distance between points

Poisson-Disk Sampling:

Random Dart Throwing
Random Dart Throwing
Random Dart Throwing

![Diagram of random dart throwing]
Random Dart Throwing
Random Dart Throwing
Random Dart Throwing
5.4 Interpreting and exploiting knowledge of the sampling spectra

Recently [39], it has been shown that the low frequency region of the radial power spectrum (of a given sampling pattern) plays a crucial role in deciding the overall variance convergence rates of sampling patterns used for Monte Carlo integration. Since blue noise sampling patterns contain almost no radial energy in the low frequency region, they are of great interest for future research to obtain fast results in rendering problems. Surprisingly, Poisson Disk samples have shown the convergence rate of $O(N^{-1})$ which is the same as given by purely random samples. This can be explained by looking at the low frequency region in the radial power spectrum of Poisson Disk samples (Fig. 5.9) which is not zero. The importance of the shape of the radial mean power spectrum in the low frequency region demands methods and algorithms that could eventually allow sample generation directly from a target Fourier spectrum.

5.4.1 Radially-averaged periodograms

Figures 5.6, 5.8 and 5.9 depict radially averaged periodograms of the various sampling strategies described in this chapter. These spectra reveal two important characteristics of estimators built using the corresponding sampling strategies.
Blue-Noise Sampling (Relaxation-based)
Blue-Noise Sampling (Relaxation-based)

1. Initialize sample positions (e.g. random)
Blue-Noise Sampling (Relaxation-based)

1. Initialize sample positions (e.g. random)

2. Use an iterative relaxation to move samples away from each other.
Lloyd-Relaxation Method
Lloyd-Relaxation Method
Lloyd-Relaxation Method
Lloyd-Relaxation Method
Lloyd-Relaxation Method
Lloyd-Relaxation Method
Lloyd-Relaxation Method
Lloyd-Relaxation Method
Lloyd-Relaxation Method
Lloyd-Relaxation Method
Lloyd-Relaxation Method
Lloyd-Relaxation Method
CCVT Sampling [Balzer et al. 2009]
CCVT Sampling [Balzer et al. 2009]

5.3.3 Tiling-based methods

There are some tile-based approaches that can be used to generate blue noise samples. Tile-based methods overcome the computational complexity of dart-throwing and/or relaxation-based approaches in generating blue noise sampling patterns. In the computer graphics community, two tile-based approaches are well known: First approach uses a set of precomputed tiles, with each tile composed of multiple samples, and later uses these tiles, in a sophisticated way, to pave the sampling domain. Second approach employs tiles with one sample per tile and uses some relaxation-based schemes, with look-up tables, to improve the overall quality of samples.

5.4 Interpreting and exploiting knowledge of the sampling spectra

Recently, it has been shown that the low frequency region of the radial power spectrum (of a given sampling pattern) plays a crucial role in deciding the overall variance convergence rates of sampling patterns used for Monte Carlo integration. Since blue noise sampling patterns contain almost no radial energy in the low frequency region, they are of great interest for future research to obtain fast results in rendering problems. Surprisingly, Poisson Disk samples have shown the convergence rate of $O(N^{1/3})$ which is the same as given by purely random samples. This can be explained by looking at the low frequency region in the radial power spectrum of Poisson Disk samples (Fig. 5.9) which is not zero. The importance of the shape of the radial mean power spectrum in the low frequency region demands methods and algorithms that could eventually allow sample generation directly from a target Fourier spectrum.

5.4.1 Radially-averaged periodograms

Figures 5.6, 5.8 and 5.9 depict radially averaged periodograms of the various sampling strategies described in this chapter. These spectra reveal two important characteristics of estimators built using the corresponding sampling strategies.
5.4 Interpreting and exploiting knowledge of the sampling spectra

Recently, it has been shown that the low frequency region of the radial power spectrum (of a given sampling pattern) plays a crucial role in deciding the overall variance convergence rates of sampling patterns used for Monte Carlo integration. Since blue noise sampling patterns contain almost no radial energy in the low frequency region, they are of great interest for future research to obtain fast results in rendering problems. Surprisingly, Poisson Disk samples have shown the convergence rate of $O(N^{1/3})$ which is the same as given by purely random samples. This can be explained by looking at the low frequency region in the radial power spectrum of Poisson Disk samples (Fig. 5.9) which is not zero. The importance of the shape of the radial mean power spectrum in the low frequency region demands methods and algorithms that could eventually allow sample generation directly from a target Fourier spectrum.

5.4.1 Radially-averaged periodograms

Figures 5.6, 5.8 and 5.9 depict radially averaged periodograms of the various sampling strategies described in this chapter. These spectra reveal two important characteristics of estimators built using the corresponding sampling strategies.
Low-Discrepancy Sampling

**Deterministic** sets of points specially crafted to be evenly distributed (have low discrepancy).

Entire field of study called Quasi-Monte Carlo (QMC)
The Van der Corput Sequence

Radical Inverse $\Phi_b$ in base 2

Subsequent points “fall into biggest holes”
The Van der Corput Sequence

Radical Inverse $\Phi_b$ in base 2

Subsequent points “fall into biggest holes”

<table>
<thead>
<tr>
<th>$k$</th>
<th>Base 2</th>
<th>$\Phi_b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>.1 = 1/2</td>
</tr>
</tbody>
</table>
The Van der Corput Sequence

Radical Inverse $\Phi_b$ in base 2

Subsequent points "fall into biggest holes"
The Van der Corput Sequence

Radical Inverse $\Phi_b$ in base 2

Subsequent points “fall into biggest holes”

<table>
<thead>
<tr>
<th>$k$</th>
<th>Base 2</th>
<th>$\Phi_b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>.1 = 1/2</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>.01 = 1/4</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>.11 = 3/4</td>
</tr>
</tbody>
</table>
The Van der Corput Sequence

Radical Inverse $\Phi_b$ in base 2

Subsequent points “fall into biggest holes”

<table>
<thead>
<tr>
<th>$k$</th>
<th>Base 2</th>
<th>$\Phi_b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>.1 = 1/2</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>.01 = 1/4</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>.11 = 3/4</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>.001 = 1/8</td>
</tr>
</tbody>
</table>
The Van der Corput Sequence

Radical Inverse $\Phi_b$ in base 2

Subsequent points “fall into biggest holes”

<table>
<thead>
<tr>
<th>$k$</th>
<th>Base 2</th>
<th>$\Phi_b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>.1 = 1/2</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>.01 = 1/4</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>.11 = 3/4</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>.001 = 1/8</td>
</tr>
<tr>
<td>5</td>
<td>101</td>
<td>.101 = 5/8</td>
</tr>
</tbody>
</table>
The Van der Corput Sequence

Radical Inverse $\Phi_b$ in base 2

Subsequent points “fall into biggest holes”

<table>
<thead>
<tr>
<th>$k$</th>
<th>Base 2</th>
<th>$\Phi_b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>..1 = 1/2</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>.01 = 1/4</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>.11 = 3/4</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>.001 = 1/8</td>
</tr>
<tr>
<td>5</td>
<td>101</td>
<td>.101 = 5/8</td>
</tr>
<tr>
<td>6</td>
<td>110</td>
<td>.011 = 3/8</td>
</tr>
</tbody>
</table>
The Van der Corput Sequence

Radical Inverse $\Phi_b$ in base 2

Subsequent points “fall into biggest holes”

<table>
<thead>
<tr>
<th>$k$</th>
<th>Base 2</th>
<th>$\Phi_b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>.1 = $1/2$</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>.01 = $1/4$</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>.11 = $3/4$</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>.001 = $1/8$</td>
</tr>
<tr>
<td>5</td>
<td>101</td>
<td>.101 = $5/8$</td>
</tr>
<tr>
<td>6</td>
<td>110</td>
<td>.011 = $3/8$</td>
</tr>
<tr>
<td>7</td>
<td>111</td>
<td>.111 = $7/8$</td>
</tr>
</tbody>
</table>
The Van der Corput Sequence

Radical Inverse $\Phi_b$ in base 2

Subsequent points “fall into biggest holes”

<table>
<thead>
<tr>
<th>$k$</th>
<th>Base 2</th>
<th>$\Phi_b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>.1 = 1/2</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>.01 = 1/4</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>.11 = 3/4</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>.001 = 1/8</td>
</tr>
<tr>
<td>5</td>
<td>101</td>
<td>.101 = 5/8</td>
</tr>
<tr>
<td>6</td>
<td>110</td>
<td>.011 = 3/8</td>
</tr>
<tr>
<td>7</td>
<td>111</td>
<td>.111 = 7/8</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Realistic Image Synthesis SS2021
Halton and Hammersley Points

**Halton**: Radical inverse with different base for each dimension:

\[ \tilde{x}_k = (\Phi_2(k), \Phi_3(k), \Phi_5(k), \ldots, \Phi_{p_n}(k)) \]
Halton and Hammersley Points

**Halton**: Radical inverse with different base for each dimension:

\[ \vec{x}_k = (\Phi_2(k), \Phi_3(k), \Phi_5(k), \ldots, \Phi_{p_n}(k)) \]

- The bases should all be relatively prime.
Halton and Hammersley Points

**Halton**: Radical inverse with different base for each dimension:

\[ \tilde{x}_k = (\Phi_2(k), \Phi_3(k), \Phi_5(k), \ldots, \Phi_{p_n}(k)) \]

- The bases should all be relatively prime.
- Incremental/progressive generation of samples
Halton and Hammersley Points

**Halton**: Radical inverse with different base for each dimension:

\[ \tilde{x}_k = (\Phi_2(k), \Phi_3(k), \Phi_5(k), \ldots, \Phi_{p_n}(k)) \]

- The bases should all be relatively prime.
- Incremental/progressive generation of samples

**Hammersley**: Same as Halton, but first dimension is \( k/N \):

\[ \tilde{x}_k = (k/N, \Phi_2(k), \Phi_3(k), \Phi_5(k), \ldots, \Phi_{p_n}(k)) \]
Halton and Hammersley Points

**Halton:** Radical inverse with different base for each dimension:

\[ \vec{x}_k = (\Phi_2(k), \Phi_3(k), \Phi_5(k), \ldots, \Phi_{p_n}(k)) \]

- The bases should all be relatively prime.
- Incremental/progressive generation of samples

**Hammersley:** Same as Halton, but first dimension is \( k/N \):

\[ \vec{x}_k = \left( \frac{k}{N}, \Phi_2(k), \Phi_3(k), \Phi_5(k), \ldots, \Phi_{p_n}(k) \right) \]

- Not incremental, need to know sample count, \( N \), in advance
The Hammersley Sequence

1 sample in each “elementary interval”
The Hammersley Sequence

1 sample in each “elementary interval”
The Hammersley Sequence

1 sample in each “elementary interval”
The Hammersley Sequence

1 sample in each “elementary interval”
The Hammersley Sequence

1 sample in each "elementary interval"
The Hammersley Sequence

1 sample in each “elementary interval”
Monte Carlo (16 random samples)
Monte Carlo (16 jittered samples)
Scrambled Low-Discrepancy Sampling
More info on QMC in Rendering

S. Premoze, A. Keller, and M. Raab. 
*Advanced (Quasi-) Monte Carlo Methods for Image Synthesis.* In SIGGRAPH 2012 courses.
How can we predict error from these?
Part 2: Formal Treatment of MSE, Bias and Variance

- Integrand Radial Spectrum
- Samples’ Radial Spectrum

Power vs. Frequency
Convergence rate for Random Samples
Convergence rate for Random Samples

Variance

Increasing Samples
Convergence rate for Random Samples

Increasing Samples

Variance
Convergence rate for Random Samples

Variance

Increasing Samples
Convergence rate for Random Samples

Variance

Increasing Samples

...
Convergence rate for Random Samples

Increasing Samples

Variance
Convergence rate for Random Samples

\[ O(N^{-1}) \]
Convergence rate for Jittered Samples

\[ O(N^{-1}) \]

Increasing Samples
Convergence rate for Jittered Samples

\[ O(N^{-1}) \]

\[ O(N^{-1.5}) \]
Convergence rate
Jittered vs Poisson Disk

\[ O(N^{-1.5}) \]

\[ O(N^{-1}) \]
Convergence rate
Jittered vs Poisson Disk

Increasing Samples

Variance

$O(N^{-1})$

$O(N^{-1.5})$
Convergence rate
Jittered vs Poisson Disk

Variance

Increasing Samples

$O(N^{-1})$

$O(N^{-1.5})$
Convergence rate
Jittered vs Poisson Disk

\[ O(N^{-1}) \]

\[ O(N^{-1.5}) \]
Samples and function in Fourier Domain

Spatial Domain

Fourier Domain
Samples and function in Fourier Domain

Spatial Domain

Fourier Domain
Samples and function in Fourier Domain

Spatial Domain

Fourier Domain

$\hat{S}(\omega)$
Samples and function in Fourier Domain

\[ f(x) \]

Spatial Domain

\[ \hat{S}(\omega) \]

Fourier Domain

\[ -w \quad 0 \quad w \]
Samples and function in Fourier Domain

Spatial Domain

Fourier Domain

$f(x)$

$\hat{S}(\omega)$
Samples and function in Fourier Domain

Spatial Domain

Fourier Domain

\[ f(x) \]

\[ \hat{S}(\omega) \]

\[ \hat{f}(\omega) \]
Convolution

Source: vdumoulin-github
Convolution

Source: vdumoulin-github
Sampling in Primal Domain is Convolution in Fourier Domain

\[ f(x) \cdot S(x) \]
Sampling in Primal Domain is Convolution in Fourier Domain

\[ f(x) S(x) \]

Fredo Durand [2011]
Sampling in Primal Domain is Convolution in Fourier Domain

\[ f(x) \ast S(x) \]

Fredo Durand [2011]
Sampling in Primal Domain is Convolution in Fourier Domain

\[ f(x) \ast S(x) \]  
\[ \hat{f}(\omega) \otimes \hat{S}(\omega) \]  
Fredo Durand [2011]
Sampling in Primal Domain is Convolution in Fourier Domain

\[ f(x) S(x) \]

\[ \hat{f}(\omega) \otimes \hat{S}(\omega) \]

Fredo Durand [2011]
Aliasing in Reconstruction

High Sampling Rate

\[ c \]

\[ -w \quad 0 \quad w \]
Aliasing in Reconstruction

High Sampling Rate

\[ c \]
Aliasing in Reconstruction

High Sampling Rate

Low Sampling Rate

\[ \text{Low Sampling Rate} \quad \text{High Sampling Rate} \]

\[ -w \quad 0 \quad w \]

\[ c \]
Aliasing in Reconstruction

High Sampling Rate

Low Sampling Rate
Aliasing in Reconstruction

High Sampling Rate

Low Sampling Rate

\[ C \]

\[ -w \quad 0 \quad w \]
Aliasing in Reconstruction

High Sampling Rate

Low Sampling Rate
Aliasing in Reconstruction

High Sampling Rate

Low Sampling Rate

\[ C \]

\[ C \]
Error in Monte Carlo Integration

High Sampling Rate

Low Sampling Rate

C

C

-w

0

w
Error in Monte Carlo Integration

High Sampling Rate

Low Sampling Rate

$w$
Error in Monte Carlo Integration
Error in Monte Carlo Integration

High Sampling Rate

Low Sampling Rate

Error in Integration
Aliasing (Reconstruction) vs. Error (Integration)
Aliasing (Reconstruction) vs. Error (Integration)

Fredo Durand [2011]
Belcour et al. [2013]

Error in Integration
Aliasing (Reconstruction) vs. Error (Integration)

Fredo Durand [2011]
Belcour et al. [2013]
Integration in the Fourier Domain
Integration is the DC term in the Fourier Domain

Spatial Domain:

\[ I = \int_D f(x) \, dx \]
Integration is the DC term in the Fourier Domain

Spatial Domain:

\[ I = \int_D f(x) \, dx \]

Fourier Domain:
Integration is the DC term in the Fourier Domain

Spatial Domain:

\[ I = \int_D f(x) \, dx \]

Fourier Domain:

\[ \hat{f}(0) \]
Monte Carlo Estimator in Spatial Domain

\[
\tilde{\mu}_N = \int_D f(x)S(x)dx
\]
Monte Carlo Estimator in Spatial Domain

\[
\tilde{\mu}_N = \int_D f(x) S(x) dx
\]

\[
S(x) = \frac{1}{N} \sum_{k=1}^{N} \delta(x - x_k)
\]
Monte Carlo Estimator in Spatial Domain

\[ \tilde{\mu}_N = \int_D f(x)S(x)dx \]

\[ S(x) = \frac{1}{N} \sum_{k=1}^{N} \delta(x - x_k) \]
Monte Carlo Estimator in Spatial Domain

\[ \tilde{\mu}_N = \int_D f(x) S(x) dx \]

\[ S(x) = \frac{1}{N} \sum_{k=1}^{N} \delta(x - x_k) \]
Monte Carlo Estimator in Spatial Domain

\[ \tilde{\mu}_N = \int_D f(x)S(x)dx \]

\[ S(x) = \frac{1}{N} \sum_{k=1}^{N} \delta(x - x_k) \]
Monte Carlo Estimator in Spatial Domain

\[ \tilde{\mu}_N = \int_D f(x)S(x)dx = \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega)d\omega \]

\[ S(x) = \frac{1}{N} \sum_{k=1}^{N} \delta(x - x_k) \]
Monte Carlo Estimator in Fourier Domain

\[ \tilde{\mu}_N = \int_D f(x)S(x)dx = \int_\Omega \hat{f}^*(\omega)\hat{S}^*(\omega)d\omega \]

\[ S(x) = \frac{1}{N} \sum_{k=1}^{N} \delta(x - x_k) \]
Monte Carlo Estimator in Fourier Domain

\[ \tilde{\mu}_N = \int_D f(x)S(x)dx = \int_{\Omega} \hat{f}^*(\omega)\hat{S}(\omega)d\omega \]

\[ S(x) = \frac{1}{N} \sum_{k=1}^{N} \delta(x - x_k) \]
Monte Carlo Estimator in Fourier Domain

\[ \tilde{\mu}_N = \int_D f(x)S(x)dx = \int_\Omega \hat{f}^* (\omega) \hat{S}(\omega)d\omega \]

\[ S(x) = \frac{1}{N} \sum_{k=1}^{N} \delta(x - x_k) \]

\[ \hat{S}(\omega) = \frac{1}{N} \sum_{k=1}^{N} e^{-i2\pi\omega x_k} \]
How to Formulate Error in Fourier Domain?

\[ I = \hat{f}(0) \]

\[ \tilde{\mu}_N = \int_\Omega \hat{f}^*(\omega) \hat{S}(\omega) d\omega \]

Fredo Durand [2011]
How to Formulate Error in Fourier Domain?

\[ I = \hat{f}(0) \]

\[ \tilde{\mu}_N = \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \]

Fredo Durand [2011]
Error in Spatial Domain

\[ I = \hat{f}(0) \]

\[ \tilde{\mu}_N = \int_\Omega \hat{f}^*(\omega)\hat{S}(\omega) \, d\omega \]

\[ I - \tilde{\mu}_N = \int_D f(x) \, dx - \int_D f(x)S(x) \, dx \]
Error in Spatial Domain

\[ I = \hat{f}(0) \]

\[ \tilde{\mu}_N = \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \]

True Integral

\[ I - \tilde{\mu}_N = \int_D f(x) dx - \int_D f(x) S(x) dx \]
Error in Spatial Domain

\[ I = \hat{f}(0) \]

\[ \tilde{\mu}_N = \int_\Omega \hat{f}^*(\omega)\hat{S}(\omega)d\omega \]

\[ I - \tilde{\mu}_N = \int_D f(x)dx - \int_D f(x)S(x)dx \]

True Integral

Monte Carlo Estimator
Error in Spatial Domain

\[ I = \hat{f}(0) \]

\[ \tilde{\mu}_N = \int_\Omega \hat{f}^*(\omega) \hat{S}(\omega) d\omega \]

\[ I - \tilde{\mu}_N = \int_D f(x) dx - \int_D f(x) S(x) dx \]
Error in Spatial Domain

\[ I = \hat{f}(0) \]

\[ I - \tilde{\mu}_N = \int_D f(x) dx - \int_D f(x) S(x) dx \]

\[ \tilde{\mu}_N = \int_\Omega \hat{f}^*(\omega) \hat{S}(\omega) d\omega \]
Error in Fourier Domain

\[ I = \hat{f}(0) \]

\[ \tilde{\mu}_N = \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \]

\[ I - \tilde{\mu}_N = \int_D f(x) dx - \int_D f(x) S(x) dx \]

\[ I - \tilde{\mu}_N = \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \]

Fredo Durand [2011]
Error in Fourier Domain

\[ I - \hat{\mu}_N = \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \]
Error = \text{Bias}^2 + \text{Variance}
Properties of Error

- Bias
- Variance
Properties of Error

- Bias: Expected value of the Error
- Variance
Properties of Error

- Bias: Expected value of the Error $\langle I - \tilde{\mu}_N \rangle$
- Variance
Properties of Error

• Bias: Expected value of the Error $\langle I - \tilde{\mu}_N \rangle$

• Variance: $\text{Var}(I - \mu_N)$

Subr and Kautz [2013]
Bias in the Monte Carlo Estimator
Bias in Fourier Domain

Error:

\[ I - \tilde{\mu}_N = \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega)\hat{S}(\omega)d\omega \]
Bias in Fourier Domain

Error:

\[ I - \tilde{\mu}_N = \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \]
Bias in Fourier Domain

Error:

\[ I - \tilde{\mu}_N = \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega)\hat{S}(\omega) d\omega \]

Bias:

\[ \langle I - \tilde{\mu}_N \rangle \]
Bias in Fourier Domain

Bias: \[
\langle I - \tilde{\mu}_N \rangle = \hat{f}(0) - \left\langle \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \right\rangle
\]
Bias in Fourier Domain

\[
\langle I - \tilde{\mu}_N \rangle = \hat{f}(0) - \left\langle \int_\Omega \hat{f}^*(\omega) \hat{S}(\omega) d\omega \right\rangle
\]
Bias in Fourier Domain

\[ \langle I - \tilde{\mu}_N \rangle = \hat{f}(0) - \left\langle \int_{\Omega} \hat{f}^*(\omega)\hat{S}(\omega) d\omega \right\rangle \]
Bias in Fourier Domain

\[ \langle I - \tilde{\mu}_N \rangle = \hat{f}(0) - \left\langle \int_\Omega \hat{f}^*(\omega) \hat{S}(\omega) d\omega \right\rangle \]

\[ \langle I - \tilde{\mu}_N \rangle = \hat{f}(0) - \int_\Omega \hat{f}^*(\omega) \langle \hat{S}(\omega) \rangle d\omega \]

Subr and Kautz [2013]
Bias in Fourier Domain

\[ \langle I - \tilde{\mu}_N \rangle = \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \langle \hat{S}(\omega) \rangle \, d\omega \]

Subr and Kautz [2013]
Bias in Fourier Domain

\[ \langle I - \bar{\mu}_N \rangle = \hat{f}(0) - \int_\Omega \hat{f}^*(\omega) \langle \hat{S}(\omega) \rangle d\omega \]

Subr and Kautz [2013]
Bias in Fourier Domain

\[ \langle I - \tilde{\mu}_N \rangle = \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \langle \hat{S}(\omega) \rangle d\omega \]

To obtain an unbiased estimator:  

Subr and Kautz [2013]
Bias in Fourier Domain

\[ \langle I - \tilde{\mu}_N \rangle = \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \langle \hat{S}(\omega) \rangle d\omega \]

To obtain an unbiased estimator: Subr and Kautz [2013]

\[ \langle \hat{S}(\omega) \rangle = 0 \]

for frequencies other than zero
How to obtain $\langle \hat{S}(\omega) \rangle = 0$ ?
Complex form in Amplitude and Phase

\[ \langle \hat{S}(\omega) \rangle = |\langle \hat{S}(\omega) \rangle| e^{-\Phi(\langle \hat{S}(\omega) \rangle)} \]
Complex form in Amplitude and Phase

\[ \langle \hat{S}(\omega) \rangle = |\langle \hat{S}(\omega) \rangle| e^{-\Phi(\langle \hat{S}(\omega) \rangle)} \]
Complex form in Amplitude and Phase

$$\langle \hat{S}(\omega) \rangle = |\langle \hat{S}(\omega) \rangle| e^{-\Phi(\langle \hat{S}(\omega) \rangle)}$$
Phase change due to Random Shift

For a given frequency $\omega$

$$\hat{S}(\omega)$$
Phase change due to Random Shift

For a given frequency $\omega$

$$\hat{S}(\omega)$$
Phase change due to Random Shift

For a given frequency $\omega$

$\hat{S}(\omega)$

Pauly et al. [2000]
Ramamoorthi et al. [2012]
Phase change due to Random Shift

For a given frequency $\omega$

$\hat{S}(\omega)$
Phase change due to Random Shift

Multiple realizations

For a given frequency $\omega$
Phase change due to Random Shift

Multiple realizations

For a given frequency $\omega$
Phase change due to Random Shift

Multiple realizations

For a given frequency $\omega$

Real

Imag
Phase change due to Random Shift

For a given frequency $\omega$

$$\langle \hat{S}(\omega) \rangle = 0$$

Multiple realizations
Phase change due to Random Shift

For a given frequency $\omega$

$$\langle \hat{S}(\omega) \rangle = 0 \quad \forall \omega \neq 0$$
Error = Bias^2 + Variance
Error = \text{Bias}^2 + \text{Variance}
Error = $\text{Bias}^2 + \text{Variance}$

• Homogenization allows representation of error only in terms of variance
Error = Bias$^2$ + Variance

• Homogenization allows representation of error only in terms of variance

• We can take any sampling pattern and homogenize it to make the Monte Carlo estimator unbiased.
Variance in the Fourier domain
Variance in the Fourier domain

Error:

\[ I - \tilde{\mu}_N = \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \]
Variance in the Fourier domain

Error:

\[ I - \tilde{\mu}_N = \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \]

\[ \text{Var}(I - \tilde{\mu}_N) \]
Variance in the Fourier domain

Error: \[ I - \tilde{\mu}_N = \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \]

\[ \text{Var}(I - \tilde{\mu}_N) = \text{Var} \left( \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \right) \]
Variance in the Fourier domain

\[
\text{Var}(I - \tilde{\mu}_N) = \text{Var} \left( \hat{f}(0) - \int_\Omega \hat{f}^*(\omega) \hat{S}(\omega) \, d\omega \right)
\]
Variance in the Fourier domain

\[
\text{Var}(I - \tilde{\mu}_N) = \text{Var} \left( \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \right)
\]
Variance in the Fourier domain

\[
\text{Var}(I - \tilde{\mu}_N) = \text{Var} \left( \hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \right)
\]
Variance in the Fourier domain

\[
\text{Var}(I - \tilde{\mu}_N) = \text{Var}\left(\hat{f}(0) - \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) \, d\omega\right)
\]

\[
\text{Var}(\tilde{\mu}_N) = \text{Var}\left(\int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) \, d\omega\right)
\]
Variance in the Fourier domain

$$\text{Var} (\tilde{\mu}_N) = \text{Var} \left( \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \right)$$
Variance in the Fourier domain

\[
\text{Var}(\tilde{\mu}_N) = \text{Var} \left( \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \right)
\]
Variance in the Fourier domain

\[
\text{Var}(\tilde{\mu}_N) = \text{Var}\left(\int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega\right)
\]
Variance in the Fourier domain

\[
\text{Var}(\tilde{\mu}_N) = \text{Var}\left( \int_{\Omega} \hat{f}^*(\omega) \hat{S}(\omega) d\omega \right)
\]

\[
\text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \text{Var} \left[ \hat{S}(\omega) \right] d\omega
\]

where,

\[
P_f(\omega) = |\hat{f}^*(\omega)|^2 \quad \text{Power Spectrum}
\]
Variance in the Fourier domain

\[ \text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \text{Var} \left( \hat{S}(\omega) \right) d\omega \]
Variance in the Fourier domain

\[
\text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \text{Var} \left( \hat{S}(\omega) \right) d\omega
\]

Subr and Kautz [2013]
Variance in the Fourier domain

\[ \text{Var}(\tilde{\mu}_N) = \int_\Omega P_f(\omega) \text{Var}(\hat{S}(\omega)) \, d\omega \]

Subr and Kautz [2013]

This is a general form, both for homogenised as well as non-homogenised sampling patterns.
Variance in the Fourier domain

\[ \text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \text{Var} \left( \hat{S}(\omega) \right) d\omega \]
Variance in the Fourier domain

\[ \text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \text{Var} \left( \hat{S}(\omega) \right) d\omega \]
Variance in the Fourier domain

\[ \text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \text{Var}\left(\hat{S}(\omega)\right) d\omega \]
Variance in the Fourier domain

\[
\text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \text{Var} \left( \hat{S}(\omega) \right) d\omega
\]

For purely random samples:
Variance in the Fourier domain

\[
\text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \text{Var} \left( \hat{S}(\omega) \right) d\omega
\]

For purely random samples:

\[
\text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \langle P_S(\omega) \rangle d\omega
\]

Fredo Durand [2011]

where,

\[
P_S(\omega) = |\hat{S}(\omega)|^2
\]
Variance in the Fourier domain

\[
\text{Var}(\mu_N) = \int_{\Omega} P_f(\omega) \text{Var} \left( \hat{S}(\omega) \right) d\omega
\]

For purely random samples: \( \langle \hat{S}(\omega) \rangle = 0 \)

\[
\text{Var}(\mu_N) = \int_{\Omega} P_f(\omega) \langle P_S(\omega) \rangle d\omega
\]

Fredo Durand [2011]

where,

\[
P_S(\omega) = |\hat{S}(\omega)|^2
\]
Variance using Homogenized Samples

Homogenizing any sampling pattern makes $\langle \hat{S}(\omega) \rangle = 0$
Variance using Homogenized Samples

Homogenizing any sampling pattern makes $\langle \hat{S}(\omega) \rangle = 0$

\[
\text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \langle P_S(\omega) \rangle d\omega \]

Pilleboue et al. [2015]

where,

$P_S(\omega) = |\hat{S}(\omega)|^2$
Variance using Homogenized Samples

$$\text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \langle P_S(\omega) \rangle \, d\omega$$
Variance using Homogenized Samples

\[
\text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \langle P_S(\omega) \rangle \, d\omega
\]
Variance in terms of n-dimensional Power Spectra

\[ \text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \langle P_S(\omega) \rangle d\omega \]
Variance in terms of n-dimensional Power Spectra

\[ \text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \langle P_S(\omega) \rangle \, d\omega \]
Variance in the Polar Coordinates

\[
\text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \langle P_S(\omega) \rangle \ d\omega
\]
Variance in the Polar Coordinates

\[
\text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \langle P_S(\omega) \rangle \, d\omega
\]

In polar coordinates:
Variance in the Polar Coordinates

\[
\text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \langle P_S(\omega) \rangle \, d\omega
\]

In polar coordinates:

\[
\text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \int_{S^{d-1}} P_f(\rho \mathbf{n}) \langle P_S(\rho \mathbf{n}) \rangle \, d\mathbf{n} \, d\rho
\]
Variance in the Polar Coordinates

In polar coordinates:

$$\text{Var}(\tilde{\mu}_N) = \int_{\Omega} P_f(\omega) \langle P_S(\omega) \rangle d\omega$$

$$\text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \int_{S^{d-1}} P_f(\rho n) \langle P_S(\rho n) \rangle dn d\rho$$
Variance in the Polar Coordinates

\[ \text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^{\infty} \int_{S^{d-1}} P_f(\rho \mathbf{n}) \langle P_S(\rho \mathbf{n}) \rangle d\mathbf{n} d\rho \]
Variance in the Polar Coordinates

$$Var[\tilde{\mu}_N] = M(S^{d-1}) \int_0^\infty \int_{S^{d-1}} P_f(\rho \mathbf{n}) \langle P_S(\rho \mathbf{n}) \rangle d\mathbf{n} d\rho$$
Variance for Isotropic Power Spectra

\[
Var[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \int_{S^{d-1}} P_f(\rho \mathbf{n}) \langle P_S(\rho \mathbf{n}) \rangle d\mathbf{n} d\rho
\]
Variance for Isotropic Power Spectra

\[ \text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \int_{S^{d-1}} P_f(\rho n) \langle P_S(\rho n) \rangle \ d\mathbf{n} \ d\rho \]

For isotropic power spectra:
Variance for Isotropic Power Spectra

\[ \text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \int_{S^{d-1}} P_f(\rho \mathbf{n}) \langle P_S(\rho \mathbf{n}) \rangle \, d\mathbf{n} \, d\rho \]

For isotropic power spectra:
Variance for Isotropic Power Spectra

$$Var[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \int_{S^{d-1}} P_f(\rho \mathbf{n}) \langle P_S(\rho \mathbf{n}) \rangle \, d\mathbf{n} \, d\rho$$

For isotropic power spectra:
Variance for Isotropic Power Spectra

\[ V\text{ar}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_{0}^{\infty} \int_{S^{d-1}} P_f(\rho \mathbf{n}) \langle P_S(\rho \mathbf{n}) \rangle d\mathbf{n} d\rho \]

For isotropic power spectra:

\[ V\text{ar}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_{0}^{\infty} \tilde{P}_f(\rho) \langle \tilde{P}_S(\rho) \rangle d\rho \]
Variance for Isotropic Power Spectra

\[ \text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \int_{S^{d-1}} P_f(\rho \mathbf{n}) \langle P_s(\rho \mathbf{n}) \rangle \, d\mathbf{n} \, d\rho \]

For isotropic power spectra:

\[ \text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \tilde{P}_f(\rho) \langle \tilde{P}_s(\rho) \rangle \, d\rho \]
Variance for Isotropic Power Spectra

\[ \text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \int_{S^{d-1}} P_f(\rho \mathbf{n}) \langle P_S(\rho \mathbf{n}) \rangle \, d\mathbf{n} \, d\rho \]

For isotropic power spectra:

\[ \text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \tilde{P}_f(\rho) \langle \tilde{P}_S(\rho) \rangle \, d\rho \]
Variance in terms of 1-dimensional Power Spectra

\[ \text{Var}[\tilde{\mu}_N] = M(S^{d-1}) \int_0^\infty \tilde{P}_f(\rho) \langle \tilde{P}_s(\rho) \rangle \, d\rho \]
Variance in terms of 1-dimensional Power Spectra

\[ \text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \tilde{P}_f(\rho) \langle \tilde{P}_S(\rho) \rangle \, d\rho \]
Variance: Integral over Product of Power Spectra

\[ \text{Var}[^{\tilde{\mu}_N}] = \mathcal{M}(S^{d-1}) \int_0^\infty \tilde{P}_f(\rho) \langle \tilde{P}_S(\rho) \rangle d\rho \]
Variance: Integral over Product of Power Spectra

\[
\text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \tilde{P}_f(\rho) \langle \tilde{P}_S(\rho) \rangle \, d\rho
\]

For given number of Samples
Variance: Integral over Product of Power Spectra

$$Var[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^{\infty} \tilde{P}_f(\rho) \langle \tilde{P}_S(\rho) \rangle d\rho$$

Integrand Radial Power Spectrum

Sampling Radial Power Spectrum

For given number of Samples
Variance: Integral over Product of Power Spectra

\[ \text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \tilde{P}_f(\rho) \langle \tilde{P}_S(\rho) \rangle \, d\rho \]

For given number of Samples

Integrand Radial Power Spectrum

Sampling Radial Power Spectrum
Variance: Integral over Product of Power Spectra

\[ \text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \tilde{P}_f(\rho) \langle \tilde{P}_S(\rho) \rangle \, d\rho \]

For given number of Samples
Variance: Integral over Product of Power Spectra

\[ \text{Var}[\tilde{\mu}_N] = \mathcal{M}(S^{d-1}) \int_0^\infty \tilde{P}_f(\rho) \langle \tilde{P}_s(\rho) \rangle \, d\rho \]

Integrand Radial Power Spectrum

Sampling Radial Power Spectrum

For given number of Samples
Spatial Distribution vs Radial Mean Power Spectra

Jitter

Poisson Disk
For 2-dimensions

<table>
<thead>
<tr>
<th>Samplers</th>
<th>Worst Case</th>
<th>Best Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jitter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poisson Disk</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCVT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Pilleboue et al. [2015]
For 2-dimensions

<table>
<thead>
<tr>
<th>Samplers</th>
<th>Worst Case</th>
<th>Best Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>$O(N^{-1})$</td>
<td></td>
</tr>
<tr>
<td>Jitter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poisson Disk</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCVT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Pilleboue et al. [2015]
For 2-dimensions

<table>
<thead>
<tr>
<th>Samplers</th>
<th>Worst Case</th>
<th>Best Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
<tr>
<td>Jitter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poisson Disk</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCVT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Pilleboue et al. [2015]
For 2-dimensions

<table>
<thead>
<tr>
<th>Samplers</th>
<th>Worst Case</th>
<th>Best Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
<tr>
<td>Jitter</td>
<td>$O(N^{-1.5})$</td>
<td></td>
</tr>
<tr>
<td>Poisson Disk</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCVT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Pilleboue et al. [2015]
For 2-dimensions

<table>
<thead>
<tr>
<th>Samplers</th>
<th>Worst Case</th>
<th>Best Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
<tr>
<td>Jitter</td>
<td>$O(N^{-1.5})$</td>
<td>$O(N^{-2})$</td>
</tr>
<tr>
<td>Poisson Disk</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCVT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Pilleboue et al. [2015]
For 2-dimensions

<table>
<thead>
<tr>
<th>Samplers</th>
<th>Worst Case</th>
<th>Best Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
<tr>
<td>Jitter</td>
<td>$O(N^{-1.5})$</td>
<td>$O(N^{-2})$</td>
</tr>
<tr>
<td>Poisson Disk</td>
<td>$O(N^{-1})$</td>
<td></td>
</tr>
<tr>
<td>CCVT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Pilleboue et al. [2015]*
For 2-dimensions

<table>
<thead>
<tr>
<th>Samplers</th>
<th>Worst Case</th>
<th>Best Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
<tr>
<td>Jitter</td>
<td>$O(N^{-1.5})$</td>
<td>$O(N^{-2})$</td>
</tr>
<tr>
<td>Poisson Disk</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
<tr>
<td>CCVT</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
</tbody>
</table>

Pilleboue et al. [2015]
### For 2-dimensions

<table>
<thead>
<tr>
<th>Samplers</th>
<th>Worst Case</th>
<th>Best Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
<tr>
<td>Jitter</td>
<td>$O(N^{-1.5})$</td>
<td>$O(N^{-2})$</td>
</tr>
<tr>
<td>Poisson Disk</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
<tr>
<td>CCVT</td>
<td>$O(N^{-1.5})$</td>
<td></td>
</tr>
</tbody>
</table>

Pilleboue et al. [2015]
For 2-dimensions

<table>
<thead>
<tr>
<th>Samplers</th>
<th>Worst Case</th>
<th>Best Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
<tr>
<td>Jitter</td>
<td>$O(N^{-1.5})$</td>
<td>$O(N^{-2})$</td>
</tr>
<tr>
<td>Poisson Disk</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
<tr>
<td>CCVT</td>
<td>$O(N^{-1.5})$</td>
<td>$O(N^{-3})$</td>
</tr>
</tbody>
</table>

Pilleboue et al. [2015]
For 2-dimensions

<table>
<thead>
<tr>
<th>Samplers</th>
<th>Worst Case</th>
<th>Best Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
<tr>
<td>Jitter</td>
<td>$O(N^{-1.5})$</td>
<td>$O(N^{-2})$</td>
</tr>
<tr>
<td>Poisson Disk</td>
<td>$O(N^{-1})$</td>
<td>$O(N^{-1})$</td>
</tr>
<tr>
<td>CCVT</td>
<td>$O(N^{-1.5})$</td>
<td>$O(N^{-3})$</td>
</tr>
</tbody>
</table>

Pilleboue et al. [2015]
Low Frequency Region

Jitter

Poisson Disk
Low Frequency Region

Jitter

Poisson Disk
Low Frequency Region

Jitter

Zoom-in

Poisson Disk

Power

Frequency
Variance for Low Sample Count

Zoom-in

Jitter

Poisson Disk

Frequency

Power

Frequency

Power
Variance for Low Sample Count

Zoom-in

Jitter

Poisson Disk

Power

Frequency
Variance for Increasing Sample Count

\[ \mathcal{O}(N^{-2}) \]

\[ \mathcal{O}(N^{-1}) \]
Experimental Verification
Convergence rate

Variance

Increasing Samples
Convergence rate

Increasing Samples

Variance
Increasing Samples

Variance

Convergence rate
Disk Function as Worst Case
Disk Function as Worst Case

![Graph showing variance vs. N with different line styles for Jittered, Poisson Disk, and various references.](image)

- Jittered
- Poisson Disk
- [Schlömer et al. 2011]
- [DeGoes et al. 2012]
- [Heck et al. 2013]
Disk Function as Worst Case

![Graph showing variance against N for different disk functions.

Legend:
- Green: Jittered
- Yellow: Poisson Disk
- Orange: Schlömer et al. 2011
- Blue: DeGoes et al. 2012
- Pink: Heck et al. 2013]
Disk Function as Worst Case
Gaussian as Best Case
Gaussian as Best Case

![Graph showing variance with different distributions: Jittered, Poisson Disk, [Schlömer et al. 2011], [DeGoes et al. 2012], [Heck et al. 2013].]
Ambient Occlusion Examples
Random vs Jittered

96 Secondary Rays

MSE: 4.74 x 10^{-3}

MSE: 8.56 x 10^{-4}
CCVT vs. Poisson Disk

96 Secondary Rays

MSE: $4.24 \times 10^{-4}$

MSE: $6.95 \times 10^{-4}$
Convergence rates
Convergence rates

![Graph showing convergence rates for different methods: Jittered, Poisson Disk, CCVT, Whitenoise, and Regular. The graph plots variance against the number of samples (N). The y-axis is on a logarithmic scale, ranging from $10^{-5}$ to $10^{-1}$. The lines show the decrease in variance as the number of samples increases.](image-url)
Jittered vs Poisson Disk

- Variance
  - Jittered
  - Poisson Disk

Graph showing variance as a function of N (number of samples).
What are the benefits of this analysis?
What are the benefits of this analysis?

• For offline rendering, analysis tells which samplers would converge faster.
What are the benefits of this analysis?

- For offline rendering, analysis tells which samplers would converge faster.
- For real time rendering, blue noise samples are more effective in reducing variance for a given number of samples.
Acknowledgements

Fourier Analysis of Numerical Integration in Monte Carlo Rendering

Kartic Subr  Gurprit Singh  *Wojciech Jarosz

*First part of slides are from Wojciech Jarosz