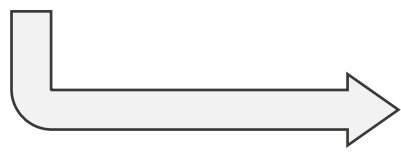


$$L_o = L_e + \int_{\Omega} L_i f \cos \theta_i d\omega_i$$




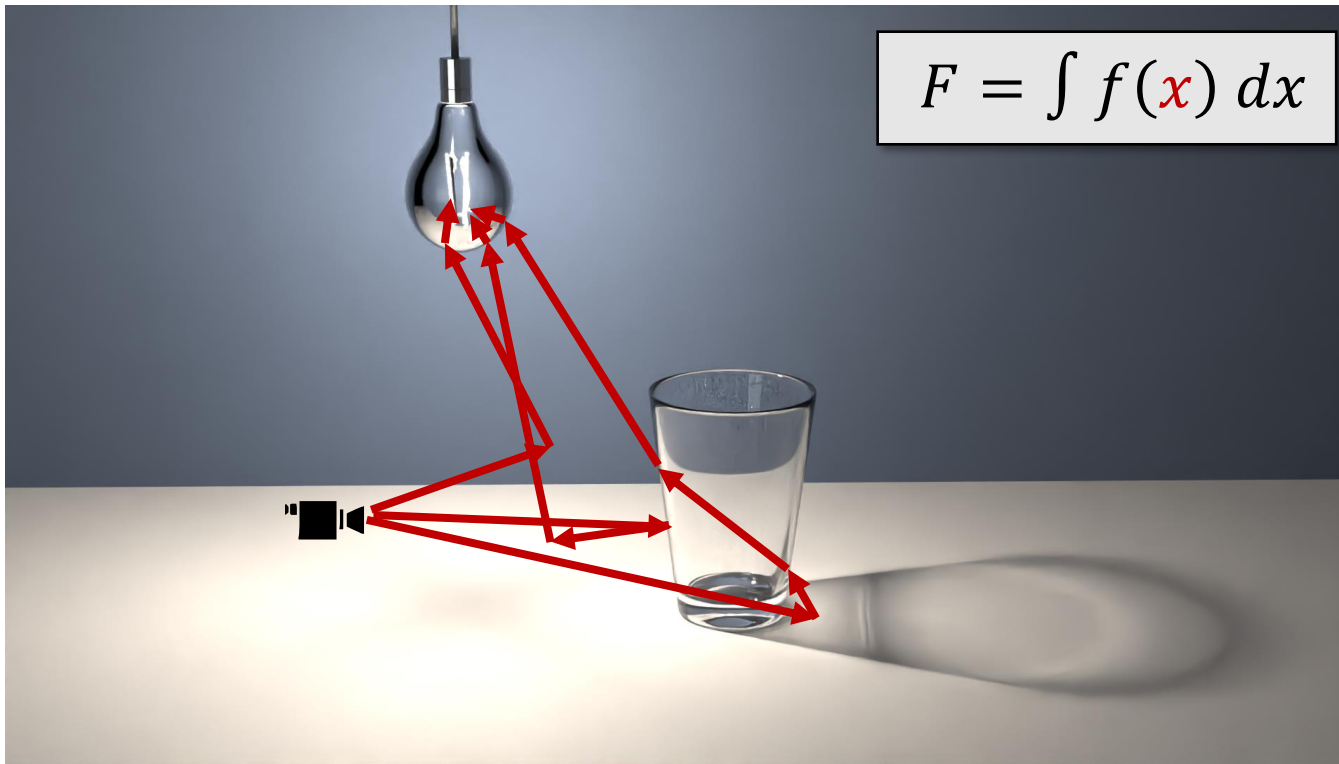
Monte Carlo Integration

The mathematical foundation of light transport simulation

Philipp Slusallek, Pascal Grittmann, Corentin Salaun

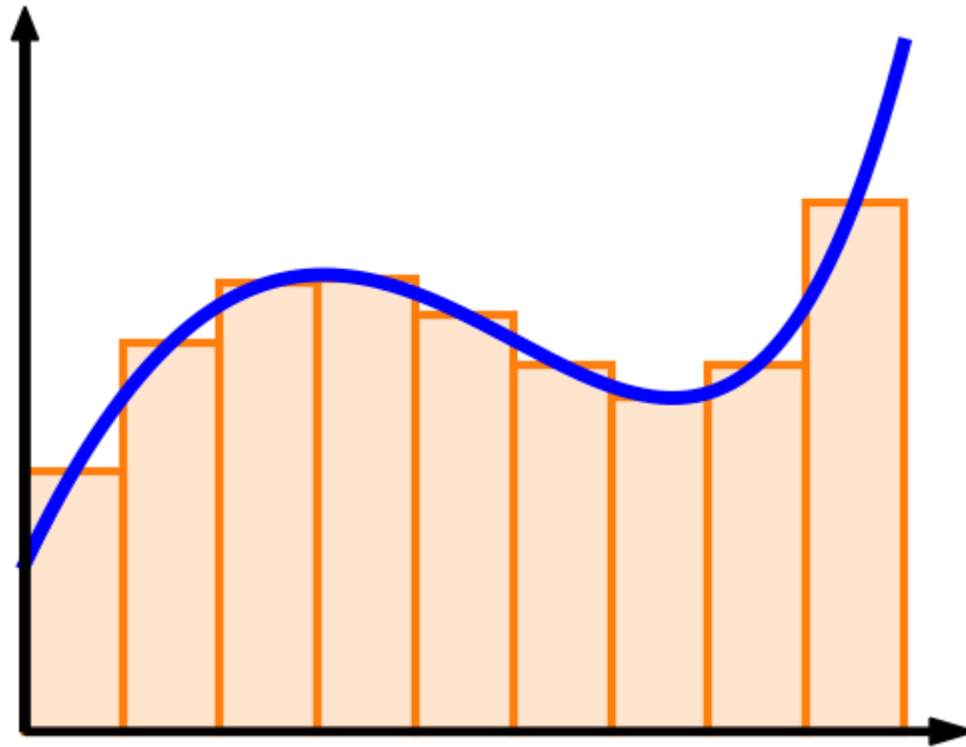
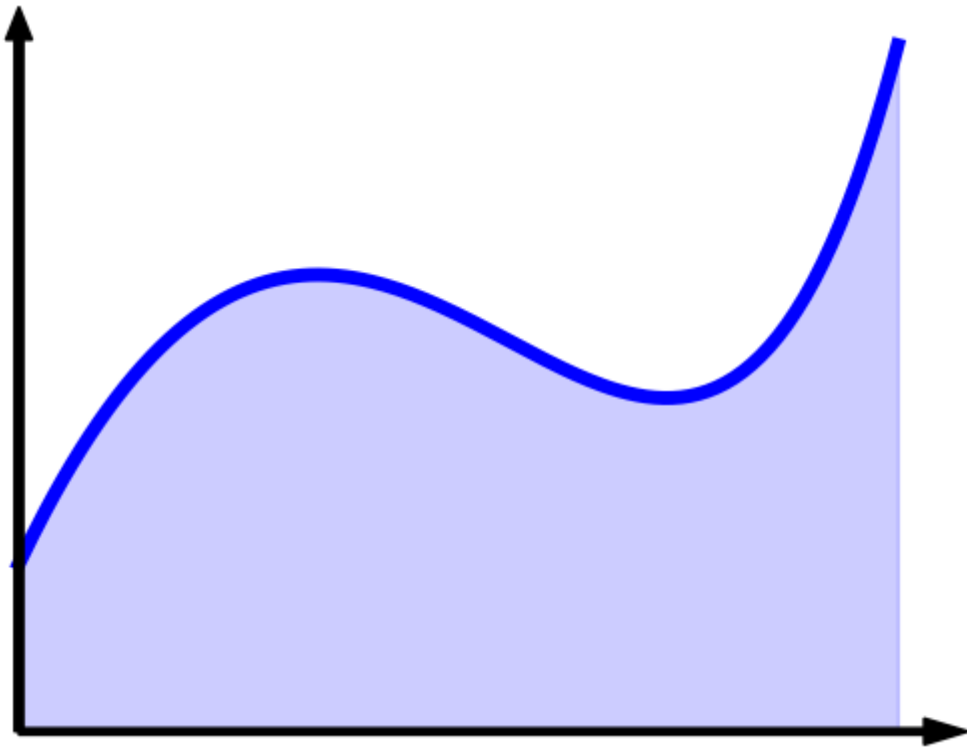
Recap

- Goal: compute pixel value
- Integral over all possible paths connecting the pixel to a light source



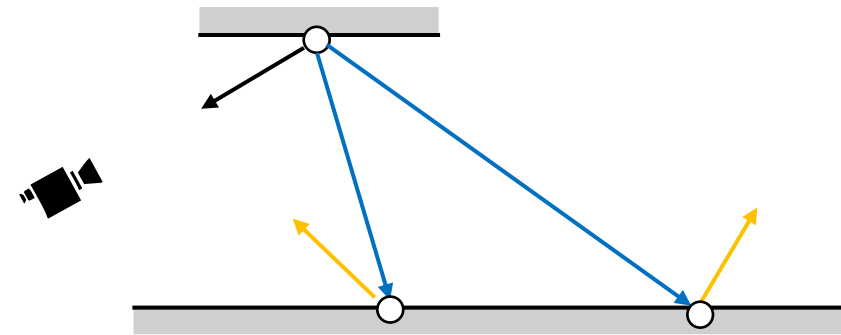
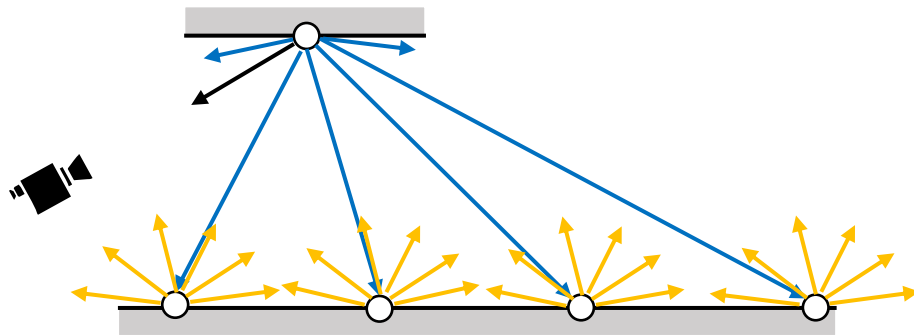
How to compute an integral?

1. Analytically (usually not possible in rendering)
2. Numerically, e.g., quadrature with midpoint rule



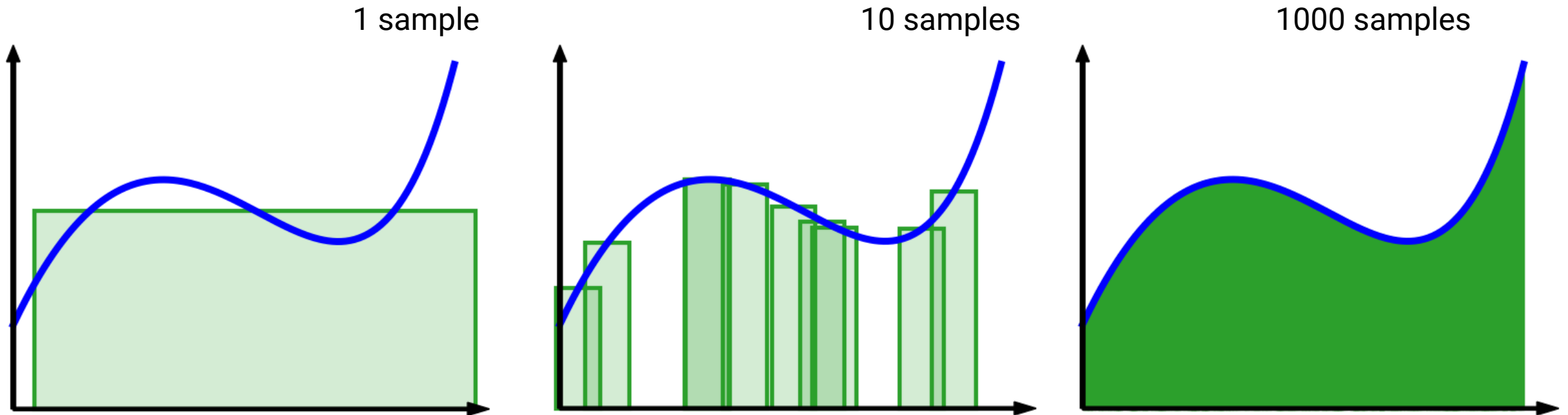
Problem: High dimensionality

- Recursive! → High (infinite) dimensionality
- Simple quadrature: exponential cost
- Monte Carlo: one n-dimensional sample at a time



Monte Carlo integration: Estimate via random samples

- Choose midpoints at random
- Converges to the integral
- Can be done **one sample at a time** – that's why it scales well!



Advantages of Monte Carlo integration

- Scales well: One sample at a time
- Converges to the correct solution
- Early iterations are noisy, but no systematic error (bias)



1 sample



10 samples

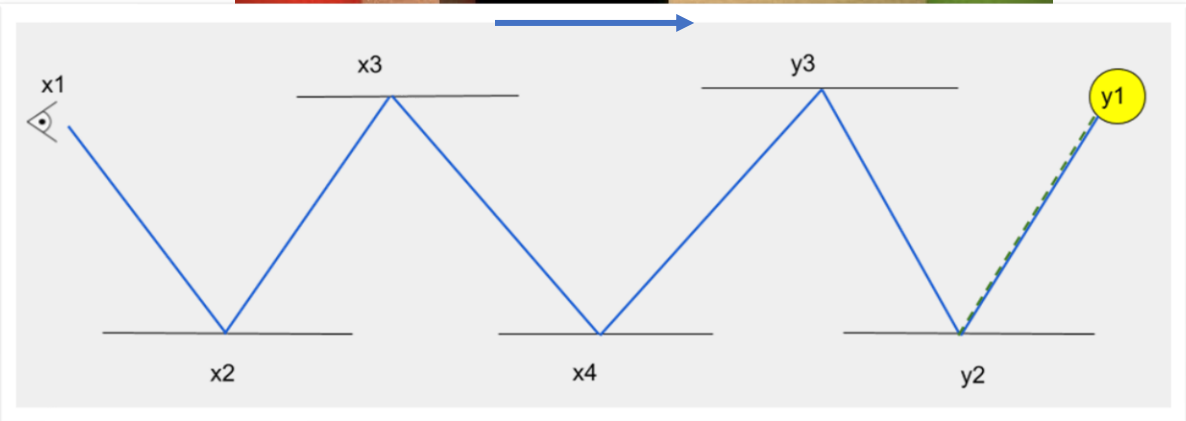
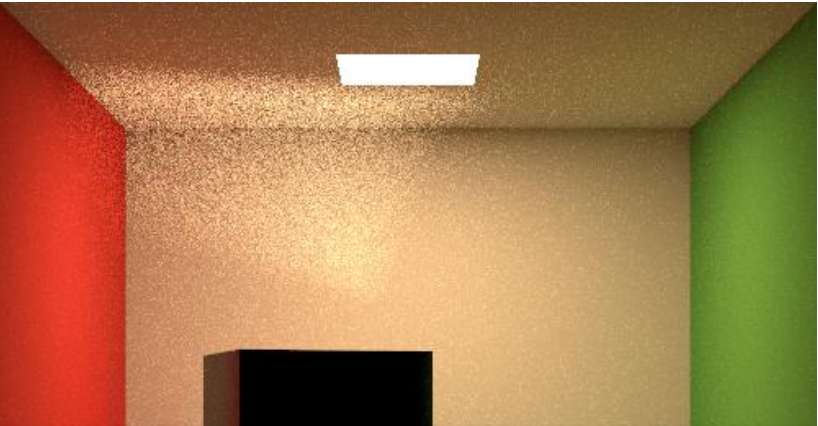


100 samples

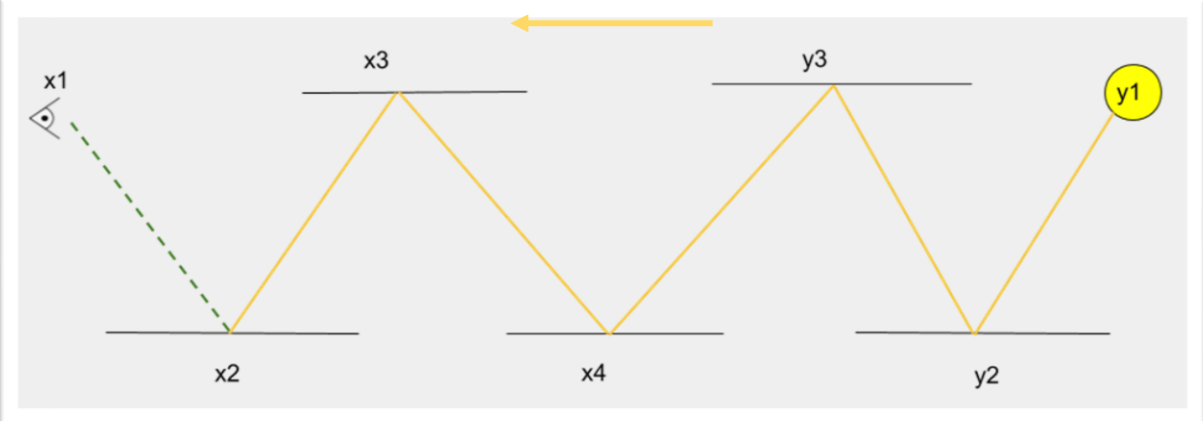
Applied to rendering

- Sample random path between camera and light, e.g., via

Recursive path tracing from the camera (“Path Tracer”)



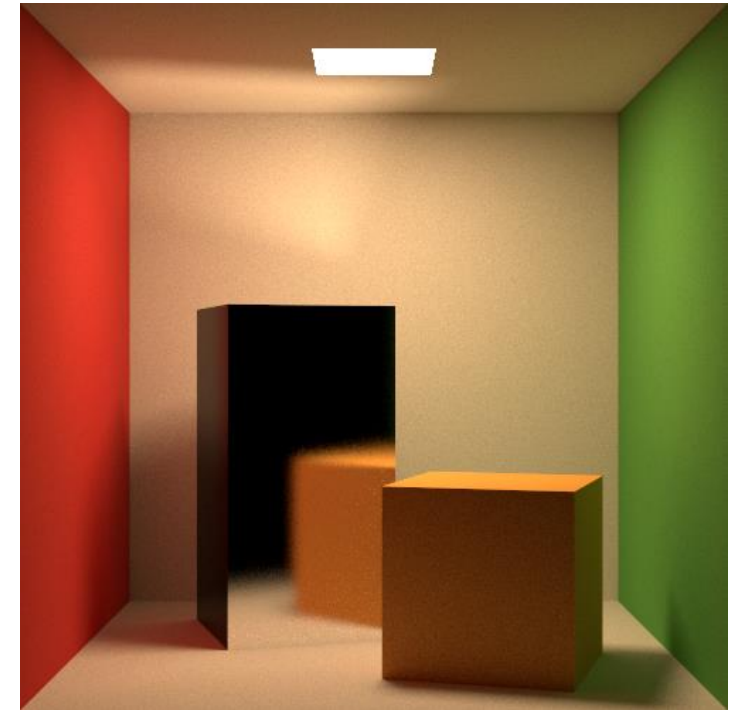
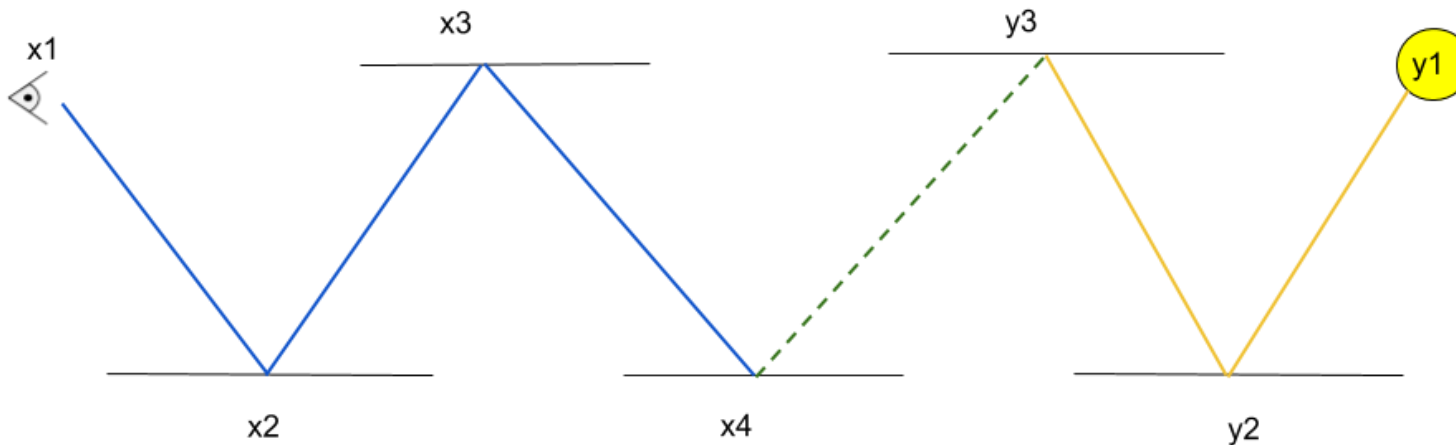
Recursive path tracing from the light (“Light Tracer”)



Applied to rendering

- Sample random path between camera and light, e.g., via

Combine both (“Bidirectional Path Tracing (BDPT)”, “Vertex Connection and Merging (VCM)”, ...)



Discussed in more depth over the next lectures!

A bit of math to back it all up

Probability density (PDF)



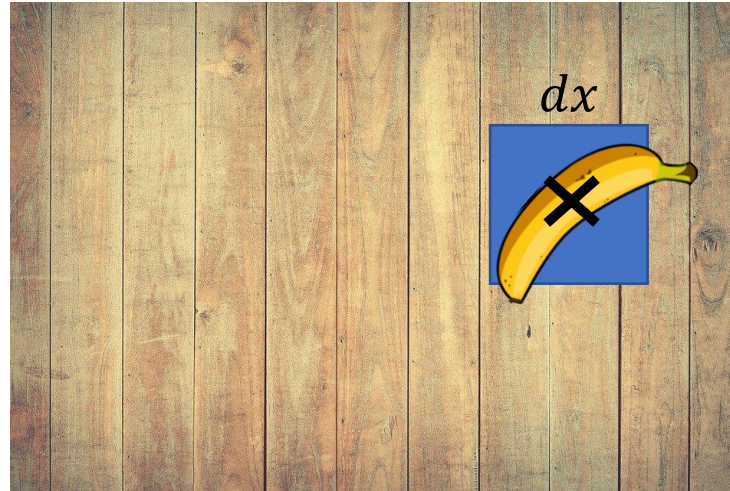
Discrete probability:

Throw a banana. Either lands in the bin or lands on the floor.

$$P(\text{bin}) \leq 1$$

$$P(\text{floor}) \leq 1$$

$$P(\text{bin}) + p(\text{floor}) = 1$$



Continuous probability:

Throw a banana. At what point on the floor will it land?

- Infinitely many possibilities (real numbers)
- Any exact position has zero probability
- $p(x)dx$ is the probability that the banana lands in the differential area dx
- $\int p(x)dx = 1$
- Units (this example): $[p(x)] = [dx]^{-1} = m^{-2}$

Basic properties are much like for discrete probabilities

- Joint PDF:
 - $p(x, y)$
- Can be written based on the conditional PDF
 - $p(x, y) = p(x|y) p(y)$
- We can obtain the marginal PDF
 - $p(x) = \int_Y p(x, y) dy = \int_Y p(x|y) p(y) dy$
- Useful for transforming samples (as we will see later)

Expected value

- For discrete random variables: Sum over all possible values F_i , multiplied by their probability

$$E[F] = \sum F_i P(F_i)$$

- Continuous case: Integral over all possible values $f(x)$ times their PDF $p(x)$

$$E[F] = \int f(x) p(x) dx$$

Monte Carlo Integration

Integral as an expected value

- We want to compute

$$F = \int_{\mathcal{X}} f(x) dx$$

- Idea of MC integration: Rewrite with sampling $x \propto p(x)$

$$F = \int_{\mathcal{X}} f(x) \frac{p(x)}{p(x)} dx = E \left[\frac{f(x)}{p(x)} \right]$$

- Where $p(x)$ is an arbitrary probability density function (PDF)
 - With $p(x) \neq 0$ whenever $f(x) \neq 0$

Primary estimator

- Sample a random x distributed according to $p(x)$ and compute

$$\langle F \rangle_1 = \frac{f(x)}{p(x)}$$

- The expected value is the integrand we are looking for

$$E[\langle F \rangle_1] = E\left[\frac{f(x)}{p(x)}\right] = F$$

The Monte Carlo estimator

- Average many primary estimators

$$\langle F \rangle_n = \frac{1}{n} \sum_{i=1}^n \frac{f(x_i)}{p(x_i)}$$

- Due to the law of large numbers (*consistent* estimator)

$$\lim_{n \rightarrow \infty} \langle F \rangle_n = E[\langle F \rangle_1] = F$$

- Converges to the desired integral!

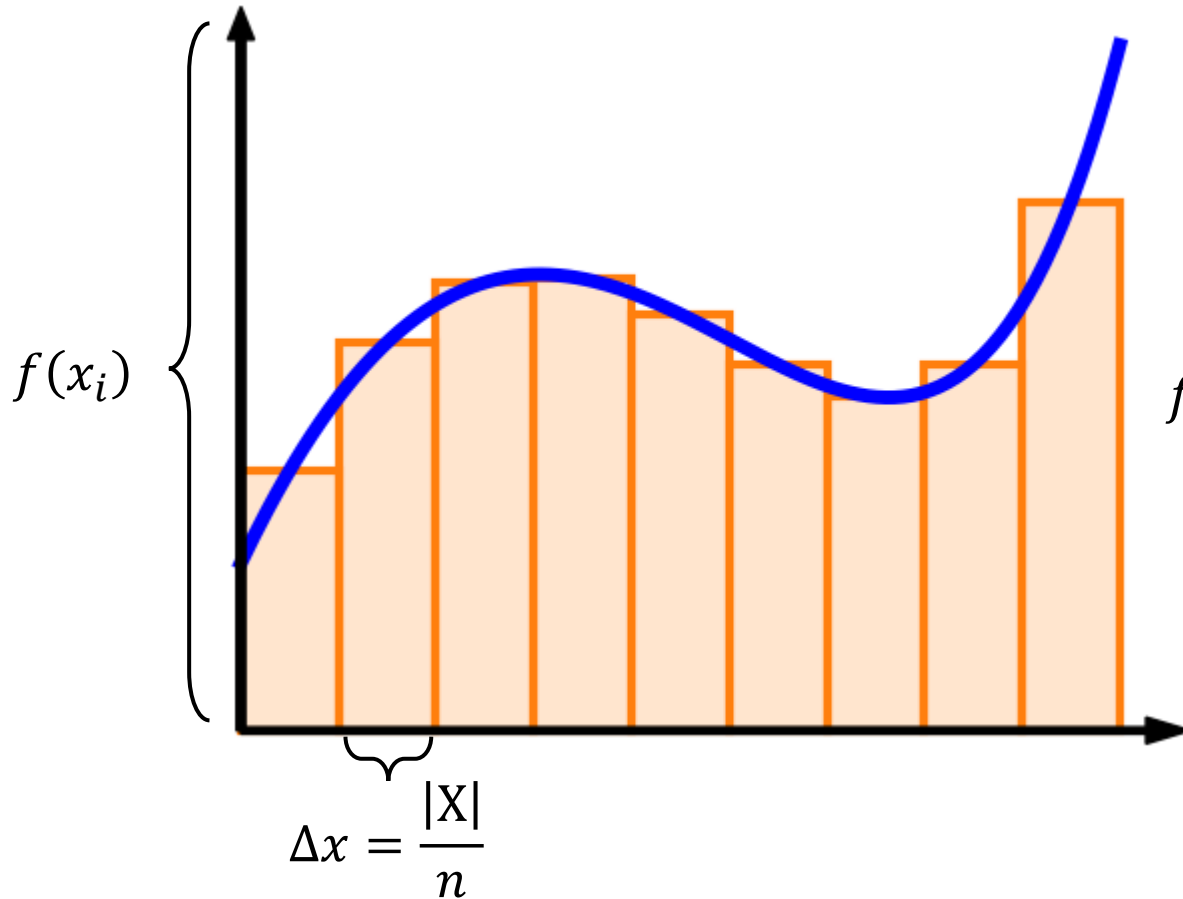
- Estimator is *unbiased*, iff

$$E[\langle F \rangle_N] = F$$

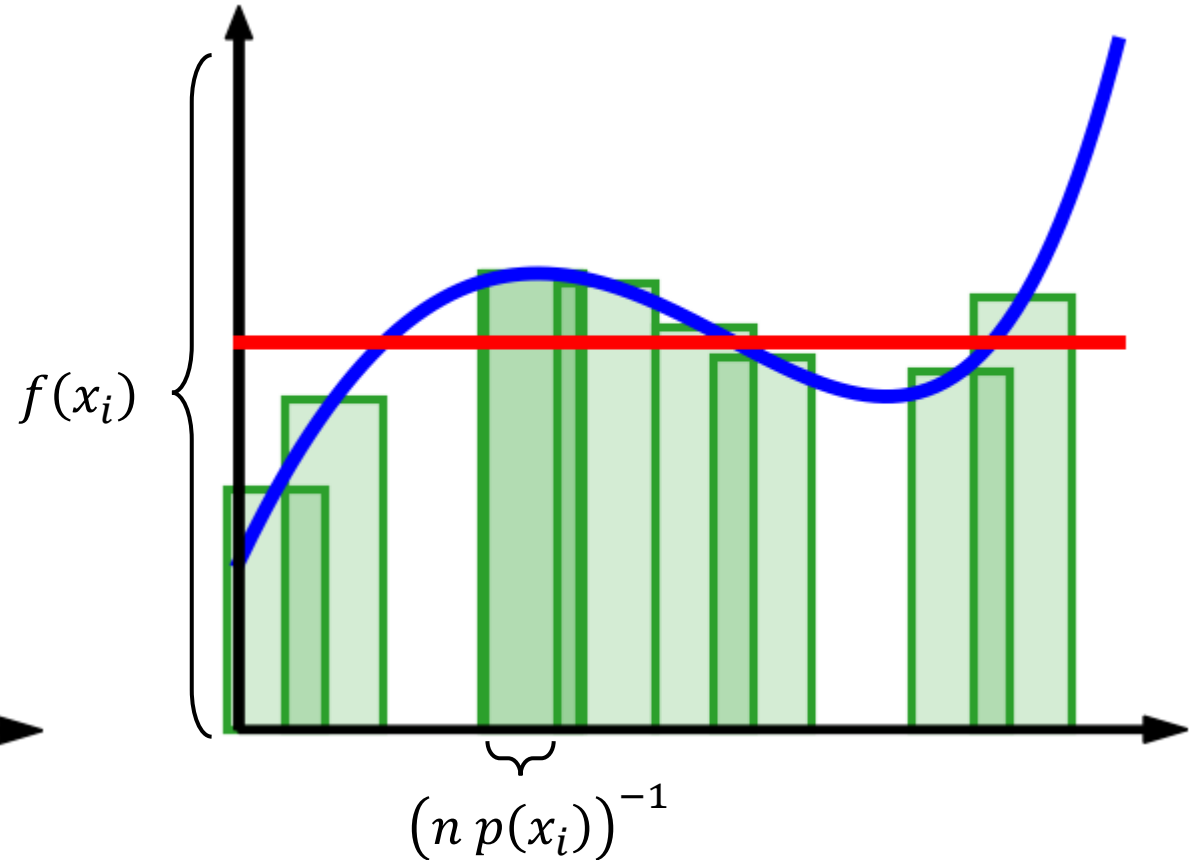
- Even stronger condition

Graphical interpretation

Regular quadrature: $F \approx \sum_{i=1}^n f(x_i)\Delta x$



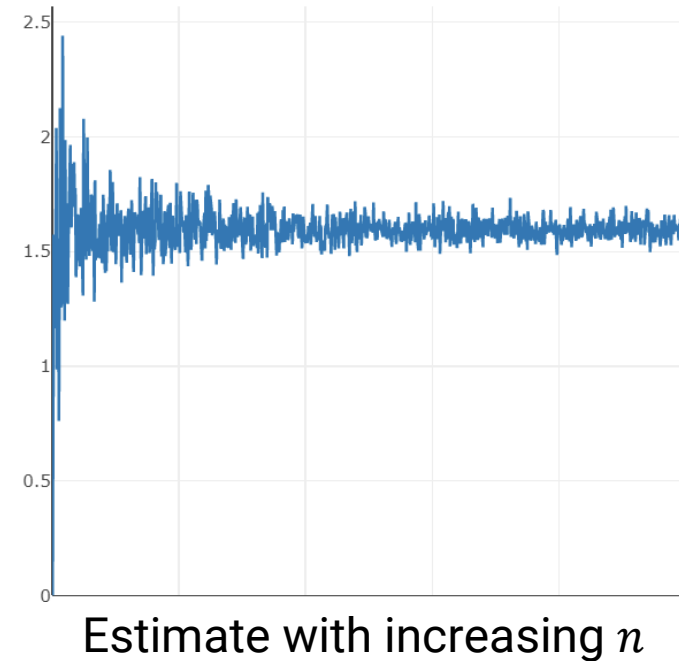
Monte Carlo: $\langle F \rangle_n = \sum_{i=1}^n \frac{f(x_i)}{n p(x_i)}$



A simple example

- Here, we use a uniform PDF $p(x) = 1$

```
int numSamples = 1_000_000;  
double estimate = 0;  
for (int i = 0; i < numSamples; ++i) {  
    x = rng.NextDouble();  
    estimate += Integrand(x) / numSamples;  
}
```



Error and convergence

- Measured by variance: Expected squared error

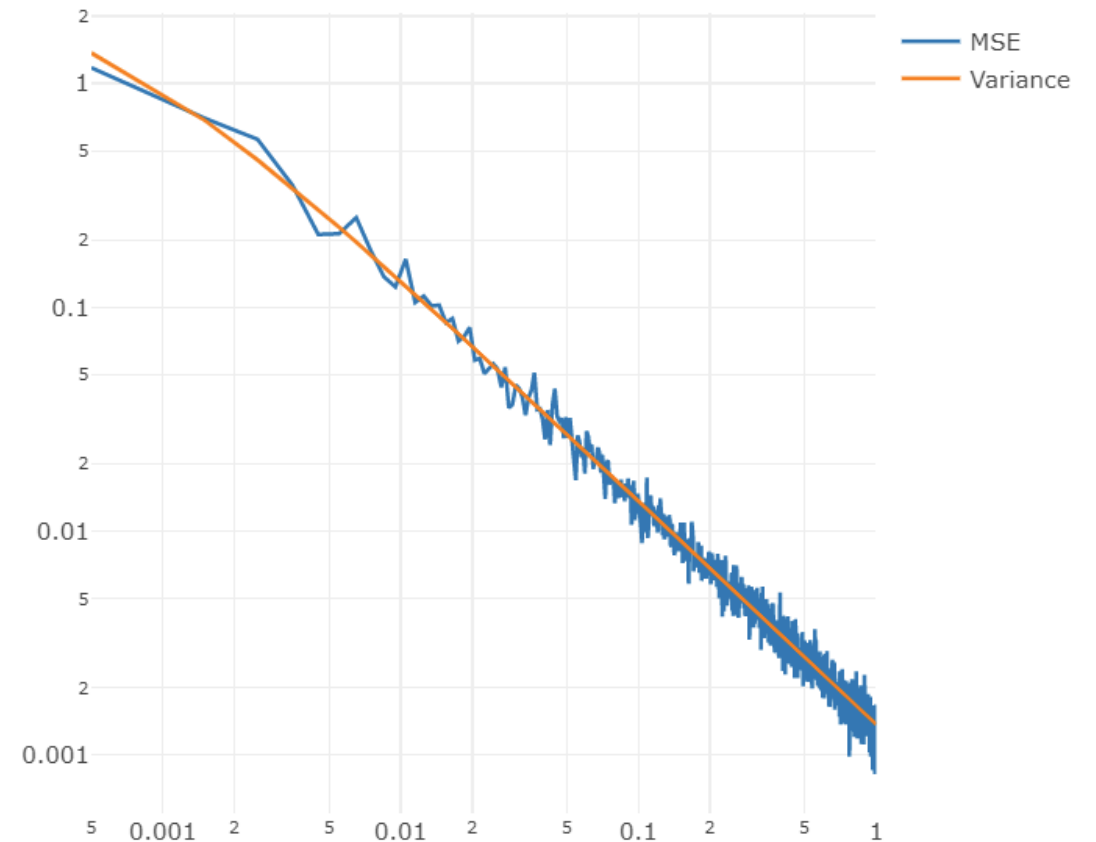
$$V[\langle F \rangle] = E[(\langle F \rangle - F)^2] = E[\langle F \rangle^2] - E[\langle F \rangle]^2$$

- Reduces with increasing n

$$V\left[\sum_{i=1}^n \frac{f(x_i)}{n p(x_i)}\right] = \frac{1}{n} V\left[\frac{f(x)}{p(x)}\right]$$

- Can be reduced by choosing $p(x)$ intelligently

- Standard deviation is simply: $\sigma = \sqrt{V[\langle F \rangle]}$
 - Expected error & in the same units as the function
 - But we will mainly deal with variance



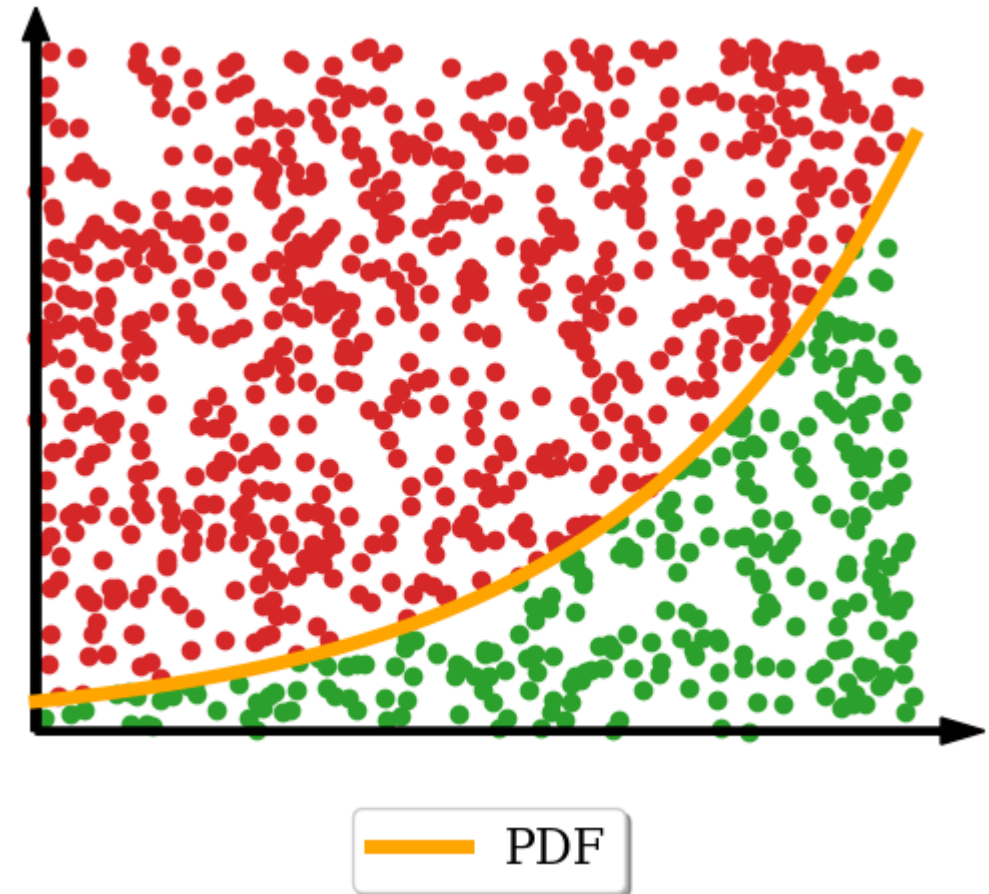
Sample transformation

How to sample according to a non-uniform PDF $p(x)$?

Rejection sampling for sample transformation

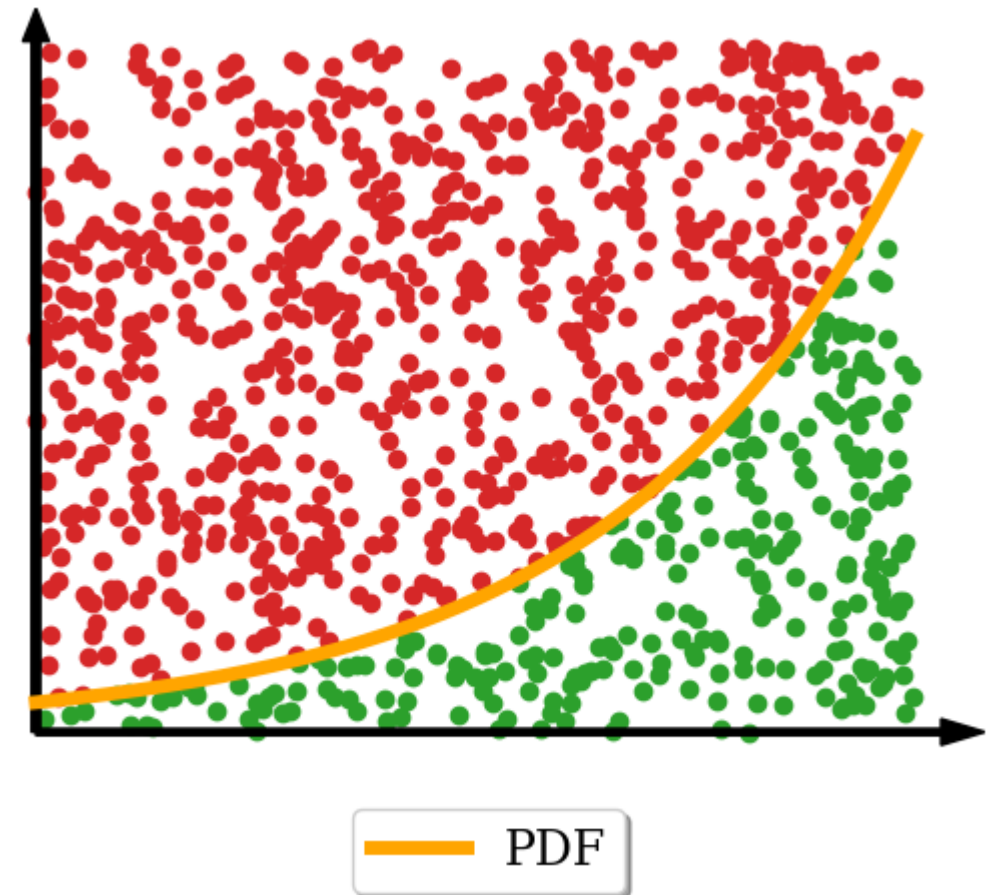
- Inverting a CDF can be difficult → use general alternative
- Uniformly sample pair (x_i, y_i) from the envelope of $p(x)$
- Only keep points (x_i, y_i) below the PDF
- Repeat until desired number of accepted points found

```
double RejectionSample() {  
    while (true) {  
        double x = rng.NextDouble();  
        double y = rng.NextDouble() * yrange;  
        if (y < Pdf(x)) return x;  
    }  
}
```

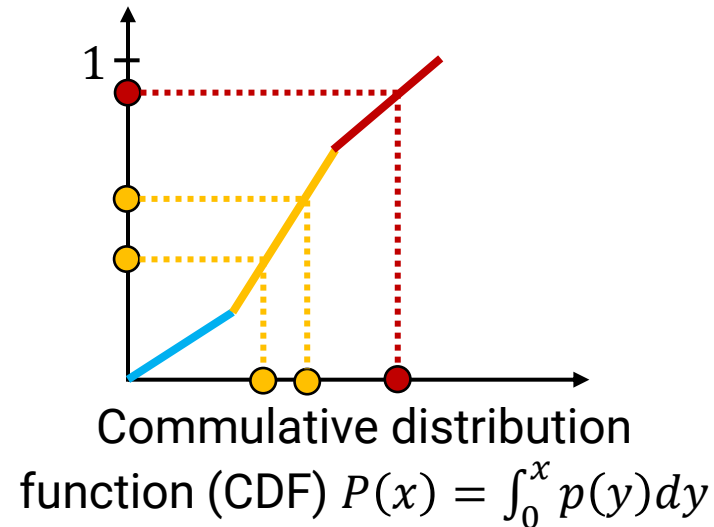
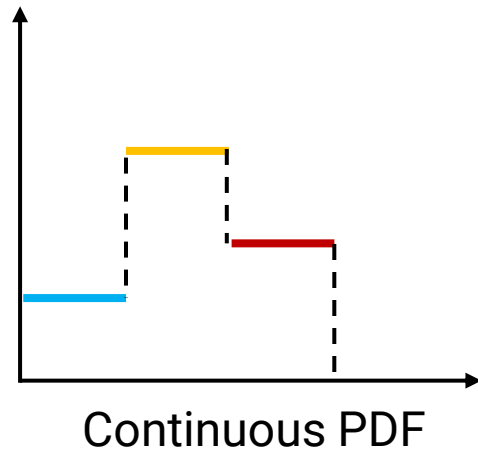
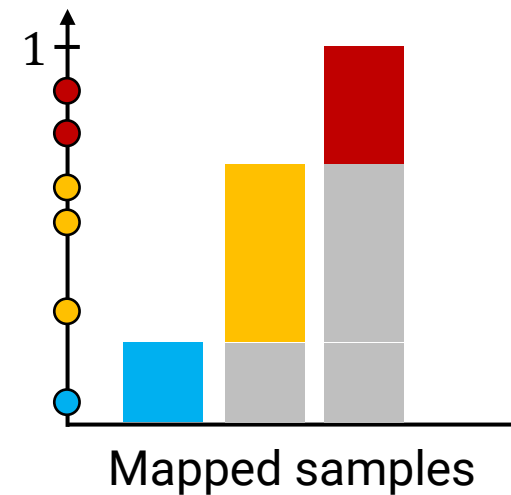
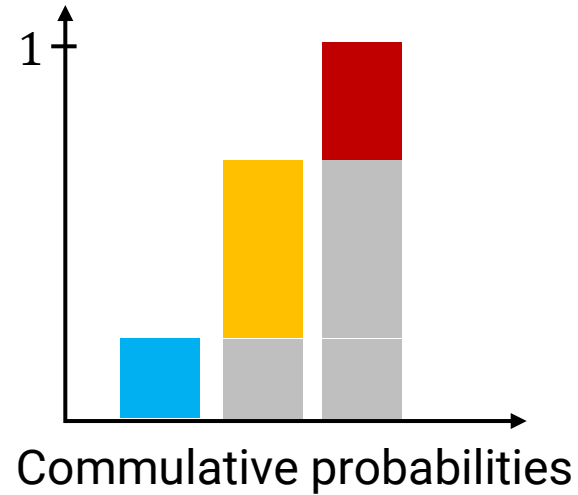
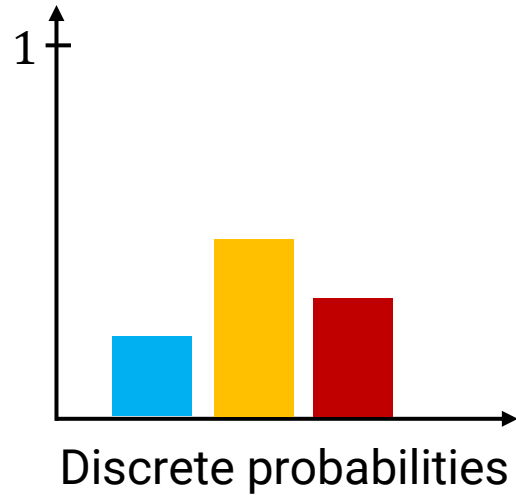


Rejection sampling

- Pros:
 - Easy to implement
 - Deals with any pdf as long as we can evaluate it pointwise
- Cons:
 - Can be inefficient (if many samples are rejected)



CDF inversion



Setting

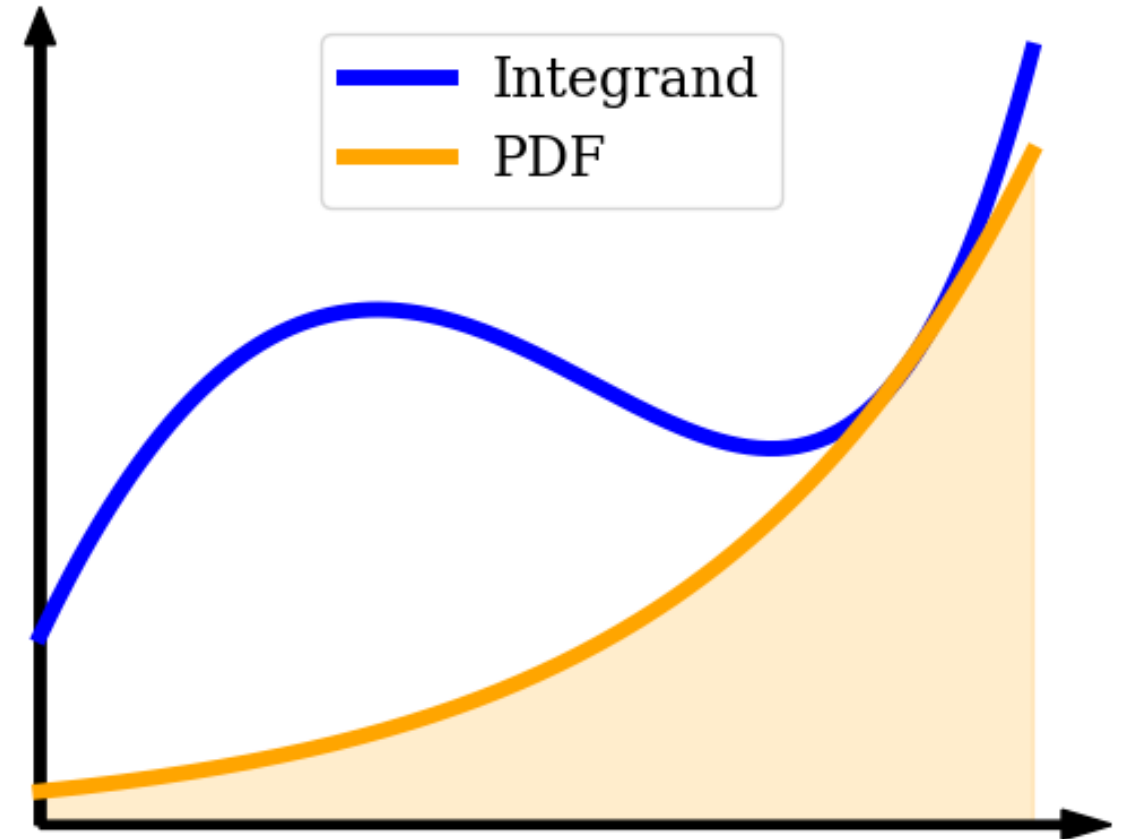
$$x = P^{-1}(y)$$

With uniform y results in

$$x \sim p(x)$$

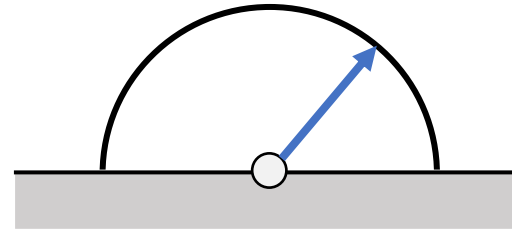
CDF inversion

- Pros:
 - Exact sampling possible
 - (Sometimes) efficient to compute
- Cons:
 - Can be difficult to find invertible CDF
 - (Sometimes) expensive to compute



Example: Uniformly sampling the hemisphere

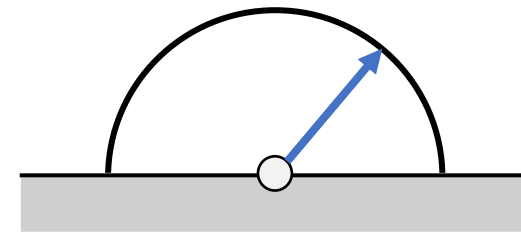
- Target pdf: $p(\omega) \propto 1$
- Normalize such that $\int p(\omega) d\omega = 1$
- We know $\int 1 d\omega = |\Omega| = 2\pi$
- So $p(\omega) = \frac{1}{2\pi}$



$$\int_{\Omega} L_i f \cos \theta d\omega_i$$

CDF inversion for the hemisphere

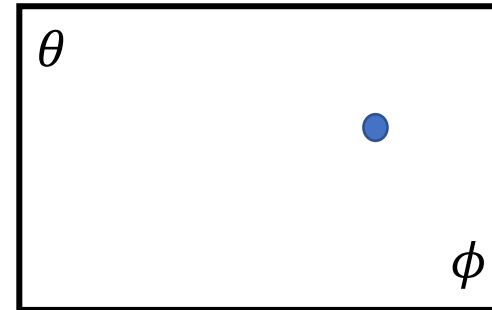
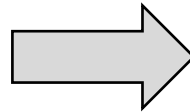
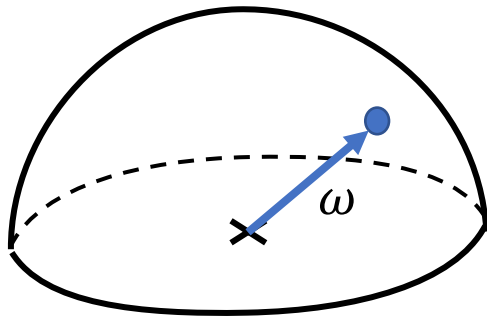
- Goal: Sample according to $p(\omega) = \frac{1}{|\Omega|} = \frac{1}{2\pi}$
- Express ω in spherical coordinates: $p = p(\theta, \phi)$
- Separate into two 1D PDFs: $p(\theta, \phi) = p(\phi)p(\theta|\phi)$
- First sample ϕ then, conditionally, sample θ



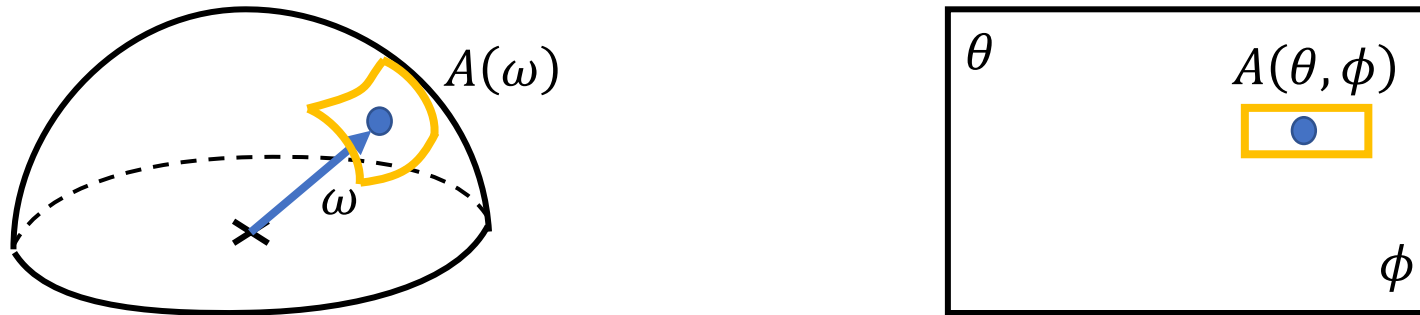
$$\int_{\Omega} L_i f \cos \theta d\omega_i$$

Mapping a direction to spherical coordinates

$$\begin{aligned}x &= \cos \phi \sin \theta \\y &= \sin \phi \sin \theta \\z &= \cos \theta\end{aligned}$$



How does the density change? (Intuition)



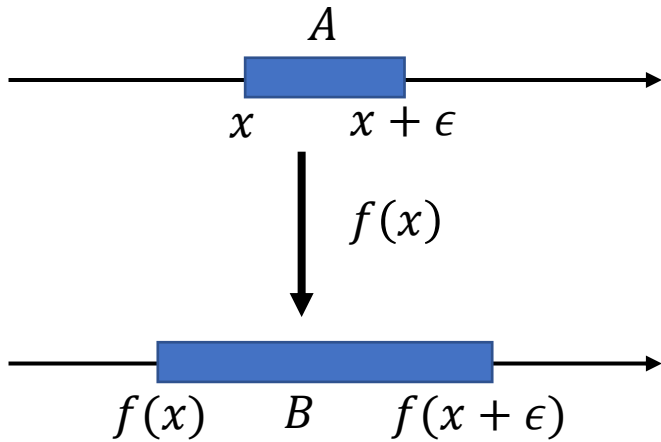
Larger area \rightarrow lower density

Ratio of densities = Inverse ratio of areas

$$\frac{p(d\theta, d\phi)}{p(d\omega)} = \frac{A(d\omega)}{A(d\theta, d\phi)}$$

(intuitively; equality only holds for differential areas $d\omega$ and $d\theta d\phi$)

Even simpler: change of length in 1D



$$\frac{B}{A} = \frac{f(x + \epsilon) - f(x)}{x + \epsilon - x}$$

$$\lim_{\epsilon \rightarrow 0} \frac{B}{A} = \frac{df(x)}{dx}$$

Derivative!

In multiple dimensions: Jacobian determinant

- Mapping from spherical coordinates to cartesian coordinates:

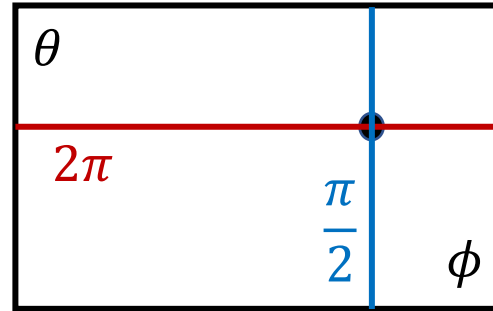
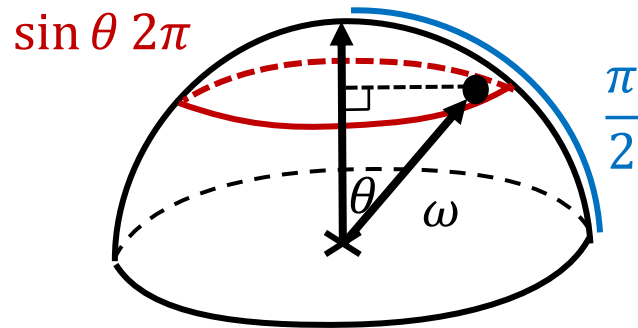
$$f(\theta, \phi, r) = \begin{pmatrix} r \cos \phi \sin \theta \\ r \sin \phi \sin \theta \\ r \cos \theta \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

- Jacobian determinant:

$$\begin{vmatrix} \frac{d}{d\theta} r \cos \phi \sin \theta & \frac{d}{d\phi} r \cos \phi \sin \theta & \frac{d}{dr} r \cos \phi \sin \theta \\ \frac{d}{d\theta} r \sin \phi \sin \theta & \frac{d}{d\phi} r \sin \phi \sin \theta & \frac{d}{dr} r \sin \phi \sin \theta \\ \frac{d}{d\theta} r \cos \theta & \frac{d}{d\phi} r \cos \theta & \frac{d}{dr} r \cos \theta \end{vmatrix} = \begin{vmatrix} r \cos \phi \cos \theta & -r \sin \phi \sin \theta & \cos \phi \sin \theta \\ r \sin \phi \cos \theta & r \cos \phi \sin \theta & \sin \phi \sin \theta \\ -r \sin \theta & 0 & \cos \theta \end{vmatrix} = r^2 \sin \theta$$

- For directions: $r = 1$
- PDF conversion: $p(\theta, \phi) = \sin \theta p(\omega)$

Verified geometrically



Area on hemisphere is locally $\sin \theta$ times as large as the corresponding area on the 2D plane of spherical coordinates

$$\Rightarrow d\omega = \sin \theta d\theta d\phi$$

$$\Rightarrow p(\theta, \phi) = \sin \theta p(\omega)$$

CDF inversion for the uniform hemisphere (continued)

- $p(\omega) = \frac{1}{2\pi} \Rightarrow p(\theta, \phi) = \frac{\sin \theta}{2\pi}$
- Marginal PDF: $p(\phi) = \int_0^{\frac{\pi}{2}} \frac{\sin \theta}{2\pi} d\theta = \frac{1}{2\pi}$
- No big surprise: it's uniform
- Conditional PDF: $p(\theta|\phi) = \frac{p(\theta, \phi)}{p(\phi)} = \sin \theta$
- The CDF is $P(\theta|\phi) = \int_0^\theta \sin x dx = 1 - \cos \theta$
- And its inverse $P^{-1}(y) = \cos^{-1}(1 - y)$

Sampling the uniform hemisphere

Input: 2 uniform random numbers in [0,1]

$$p(\phi) = \frac{1}{2\pi} \Rightarrow \phi = 2\pi x$$

Output: cartesian coordinates (z axis up)
of the direction in the hemisphere

$$\theta = \cos^{-1}(1 - y)$$

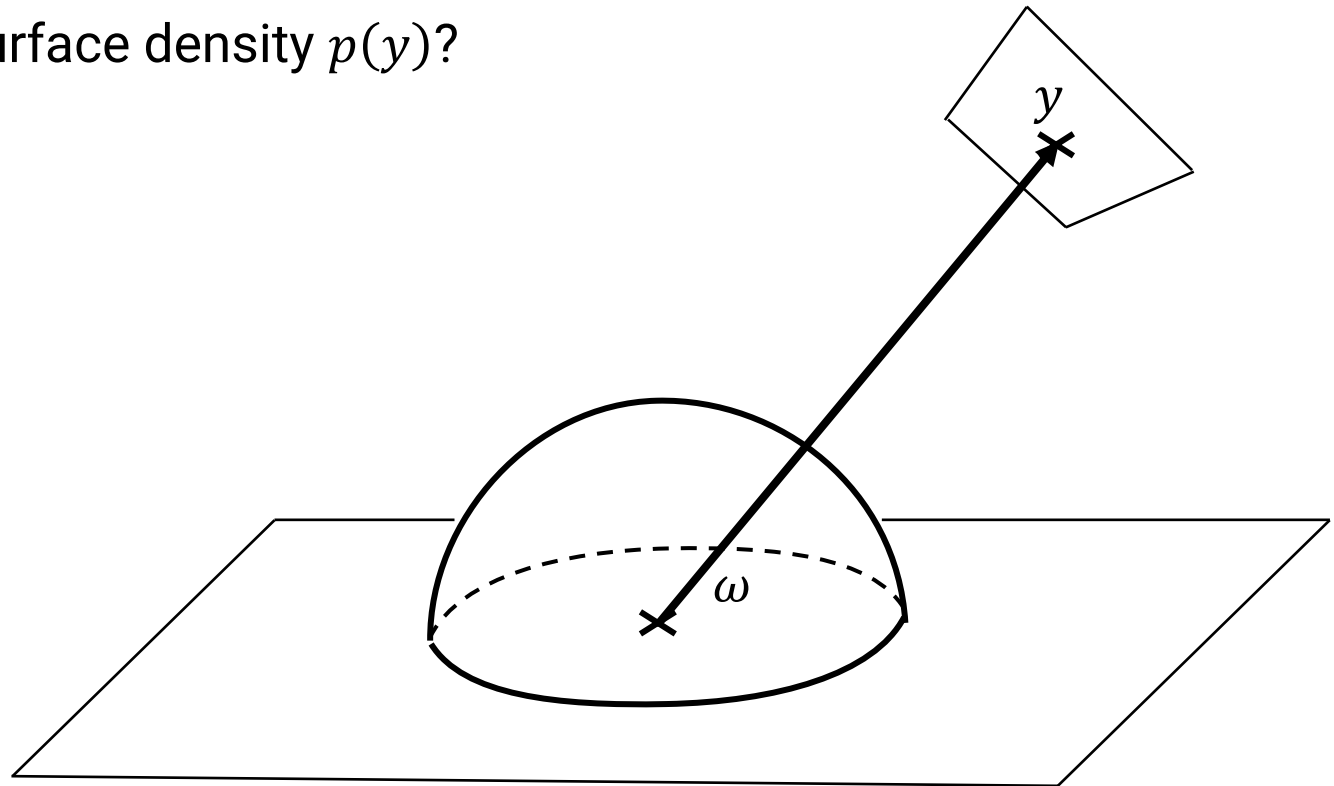
```
Vector3 ToUniformHemisphere(float u, float v) {  
    float phi = 2 * MathF.PI * u;  
  
    float cosTheta = 1 - v;  
    float sinTheta = MathF.Sqrt(1 - cosTheta * cosTheta);  
  
    return new Vector3(  
        sinTheta * MathF.Cos(phi),  
        sinTheta * MathF.Sin(phi),  
        cosTheta  
    );  
}
```

Sampling directions via points on a surface

- Sometimes, we rather sample points on surfaces than directions
- Example: connecting directly to a point on a light
- How can we compute $p(\omega)$ from a surface density $p(y)$?

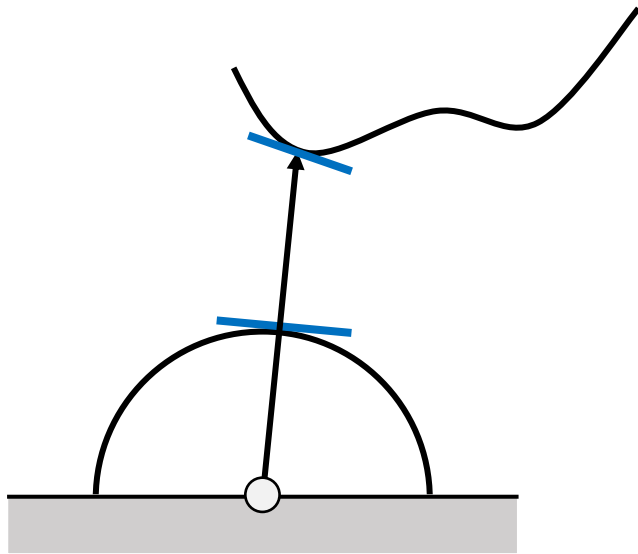
$$p(\omega)d\omega = p(y(\omega))dy(\omega)$$

$$\Leftrightarrow p(\omega) = p(y(\omega)) \frac{dy(\omega)}{d\omega}$$

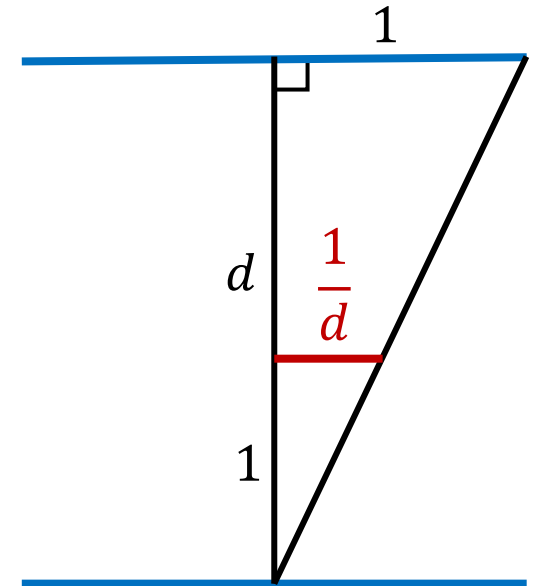
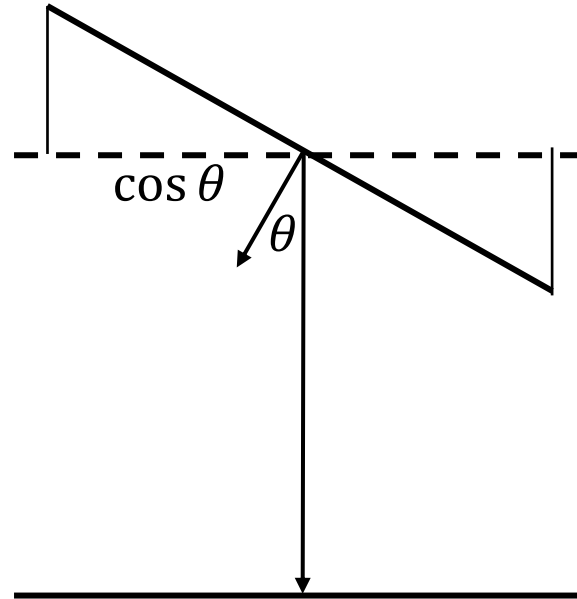


Geometry term

- Surfaces are 2D manifolds
- They locally resemble a plane



$$d\omega = \frac{\cos \theta}{d^2} dy \Leftrightarrow \frac{d\omega}{dy} = \frac{\cos \theta}{d^2}$$



Sampling directions via points on a surface

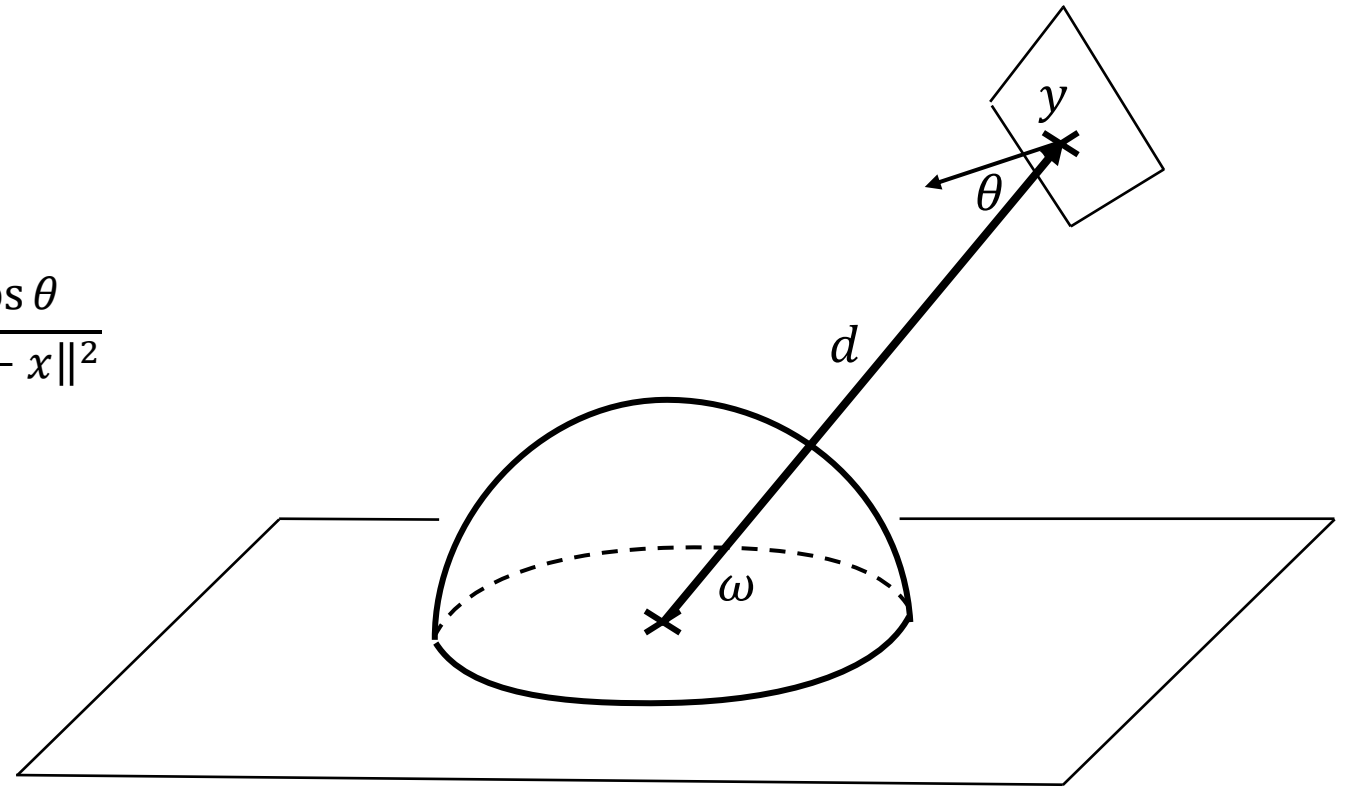
1. Sample a point y on the surface (e.g., light source)

2. Compute

$$p(\omega) = p(y) \frac{\|y - x\|^2}{\cos \theta}$$

3. Monte Carlo estimate:

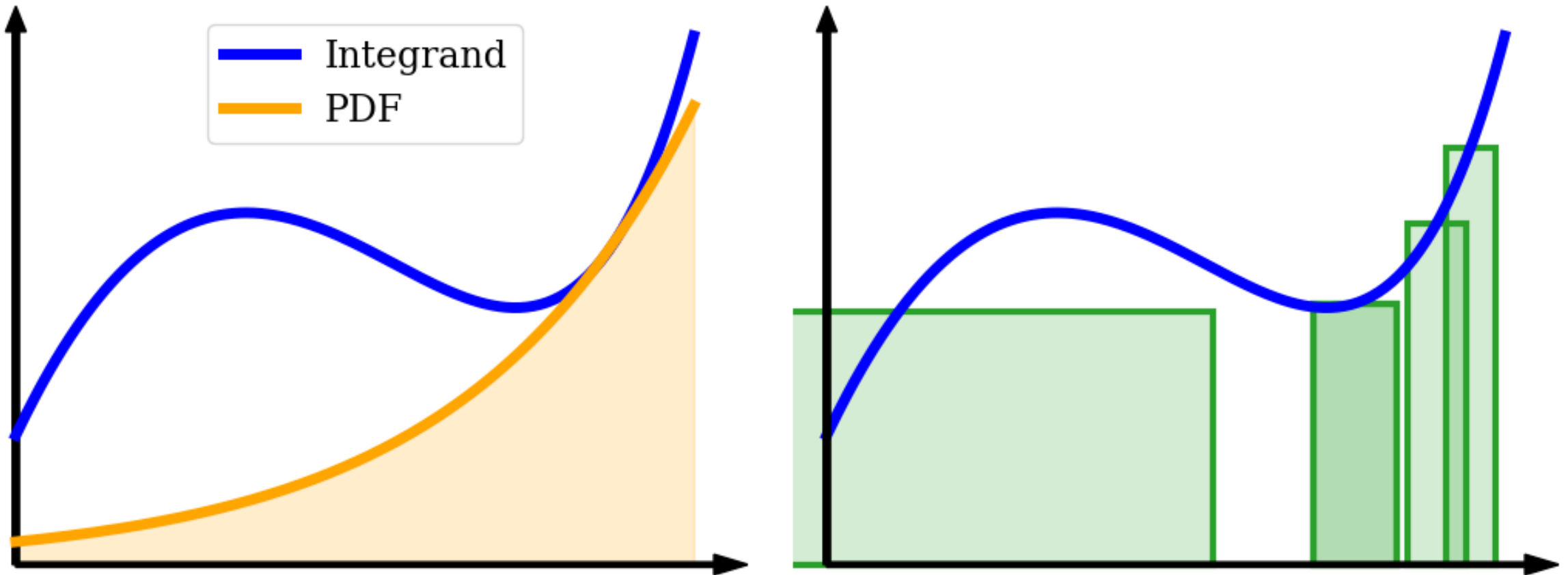
$$\frac{f(x, \omega(x, y))}{p(\omega)} = \frac{f(x, \omega(x, y))}{p(y)} \frac{\cos \theta}{\|y - x\|^2}$$



Variance Reduction

Importance sampling

- Choosing $p(x)$ to focus on “important” regions



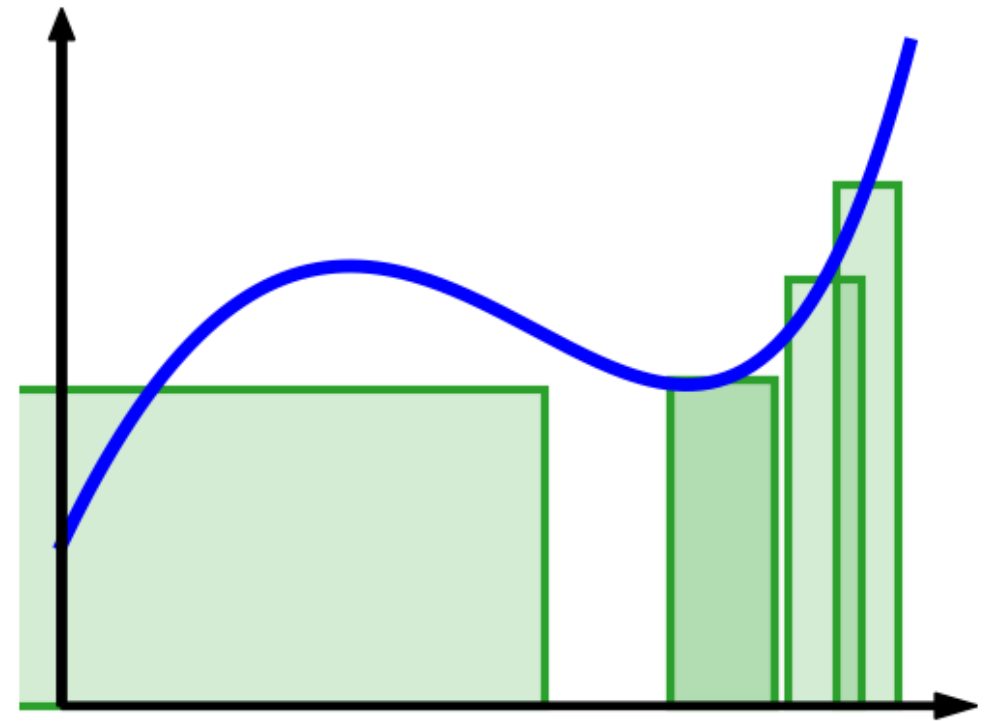
“Zero variance sampling”?

- Ideally: Choose $p(x) \propto f(x) \Rightarrow \frac{p(x)}{f(x)} = C = \text{const.}$
- Then, a single sample already gives the correct result
- However, this requires we know $\int f(x)dx$ already...

$$\int p(x)dx = 1 \Rightarrow C = 1 / \int f(x)dx$$

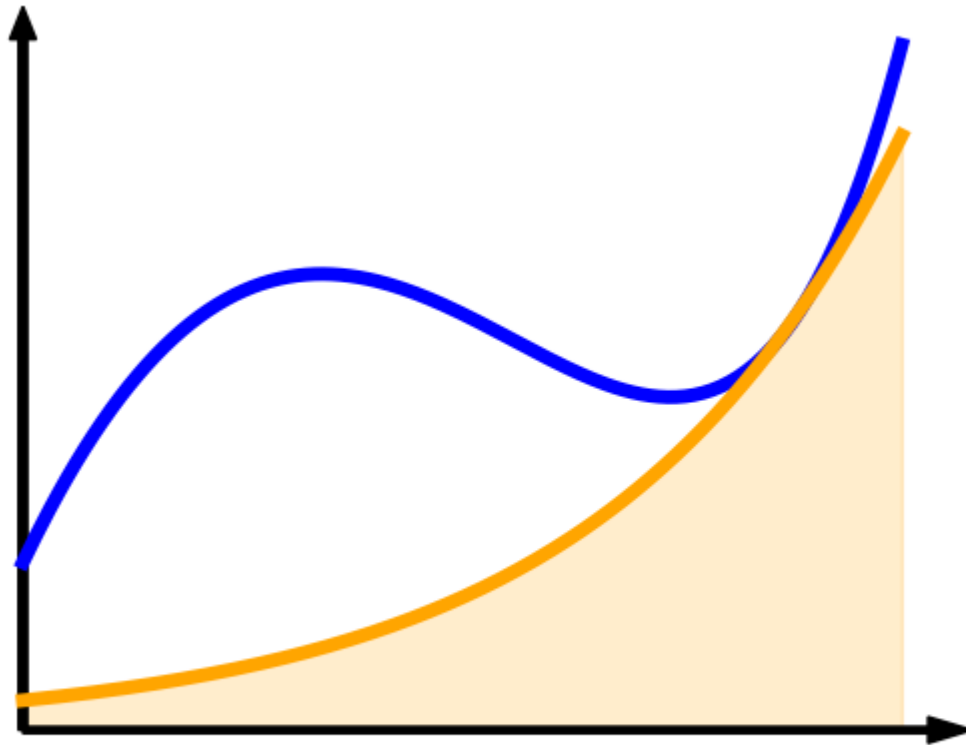
$$\Leftrightarrow p(x) = \frac{f(x)}{\int f(x)dx} \propto f(x)$$

- Unfortunately, does not work ☹️

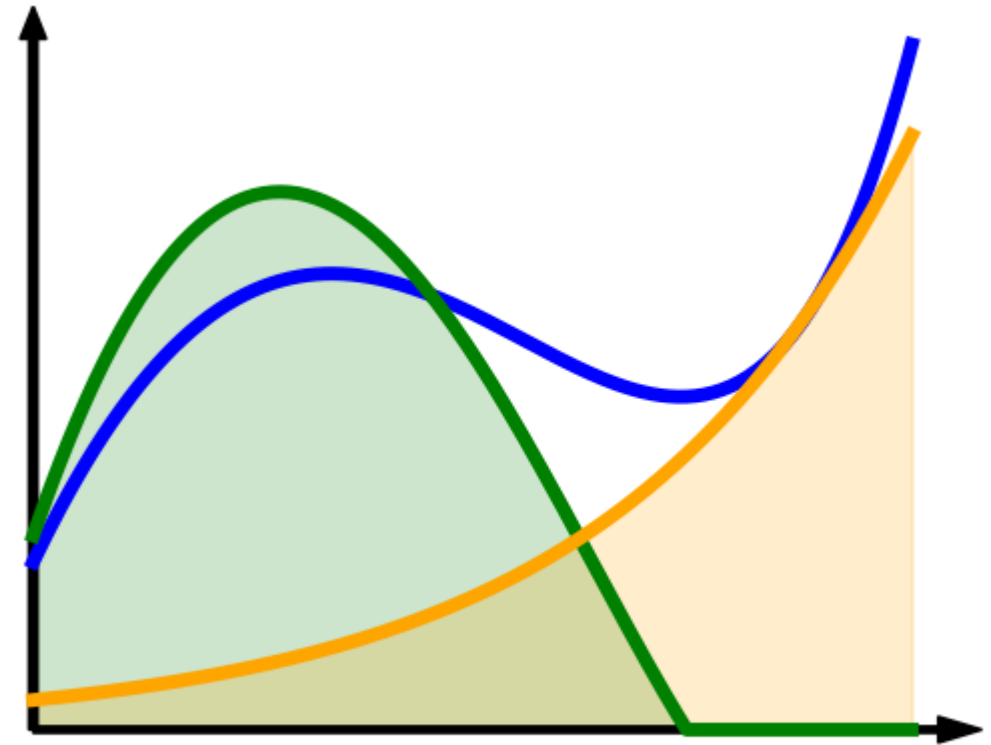


Multiple importance sampling (MIS)

- Idea: use multiple densities that match different regions well



— Integrand — PDF



— Integrand — PDF_A — PDF_B

Simple average

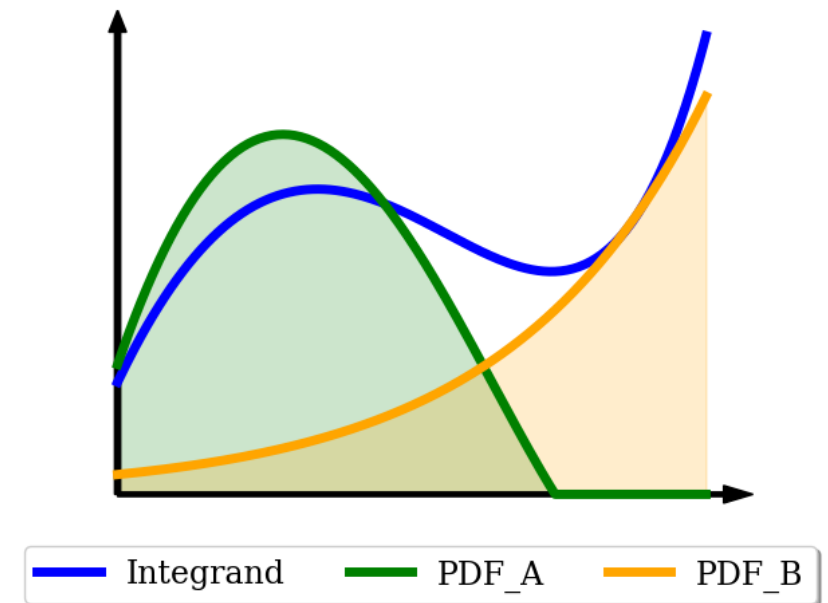
- The simplest way: average both estimators

$$\langle F \rangle_{\text{avg}} = \frac{1}{2} \sum_{i=1}^{n_A} \frac{f(x_i)}{n_A p_A(x_i)} + \frac{1}{2} \sum_{i=1}^{n_B} \frac{f(x_i)}{n_B p_B(x_i)}$$

- The resulting variance is the weighted sum (iff independent):

$$V[\langle F \rangle_{\text{avg}}] = \frac{1}{4} V[\langle F \rangle_A] + \frac{1}{4} V[\langle F \rangle_B]$$

- If either (or both) individual variances are high, this is still bad!
→ Want to tune where to prefer which estimator!



MIS

- Form a linear combination
- Weighting function $w_t(x)$ for each sampling technique t

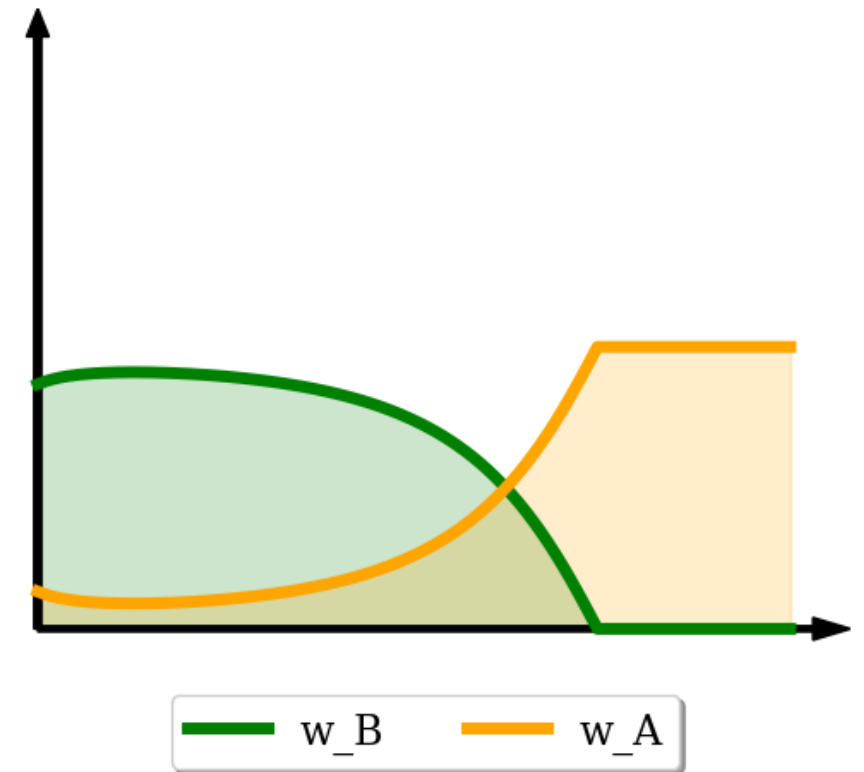
$$\langle F \rangle_{\text{MIS}} = \sum_t \sum_{i=1}^{n_t} w_t(x_{t,i}) \frac{f(x_{t,i})}{n_t p_t(x_{t,i})}$$

- Unbiased estimator is achieved if for all x where $f(x) \neq 0$:

$$\sum_t w_t(x) = 1$$

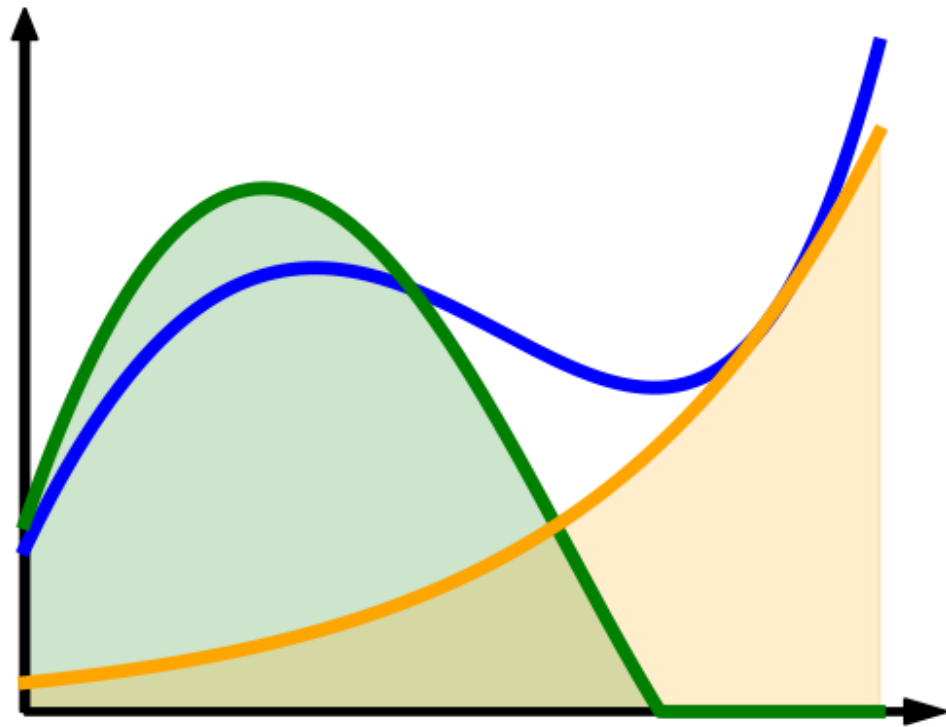
$$p_t(x) = 0 \Rightarrow w_t(x) = 0$$

MIS weights



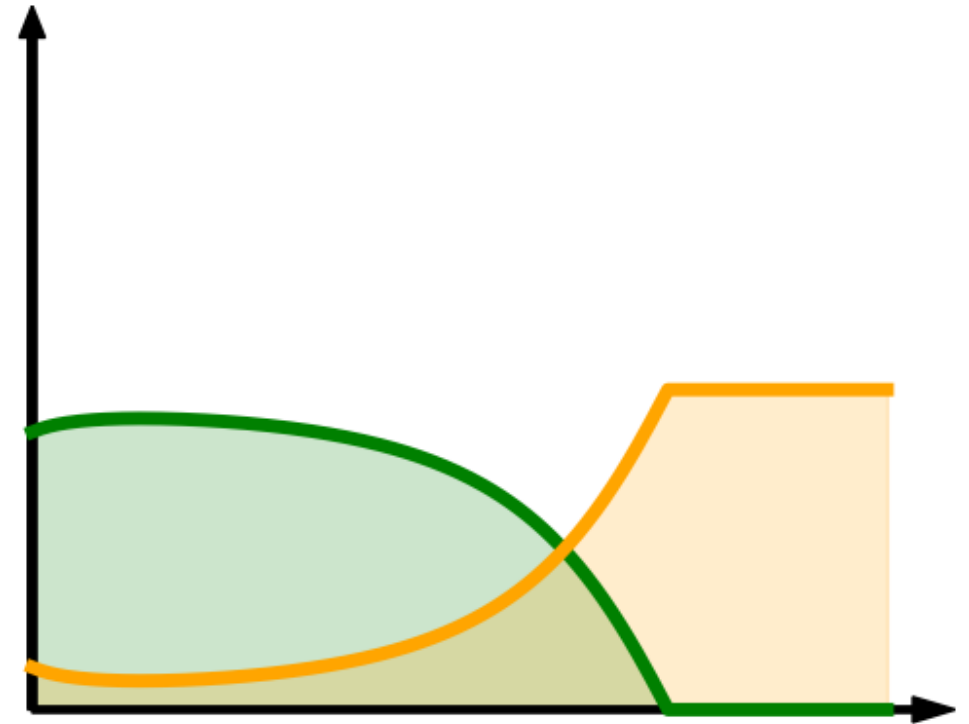
Balance heuristic $w_t(x) = \frac{n_t p_t(x)}{\sum_k n_k p_k(x)}$

Integrand and densities



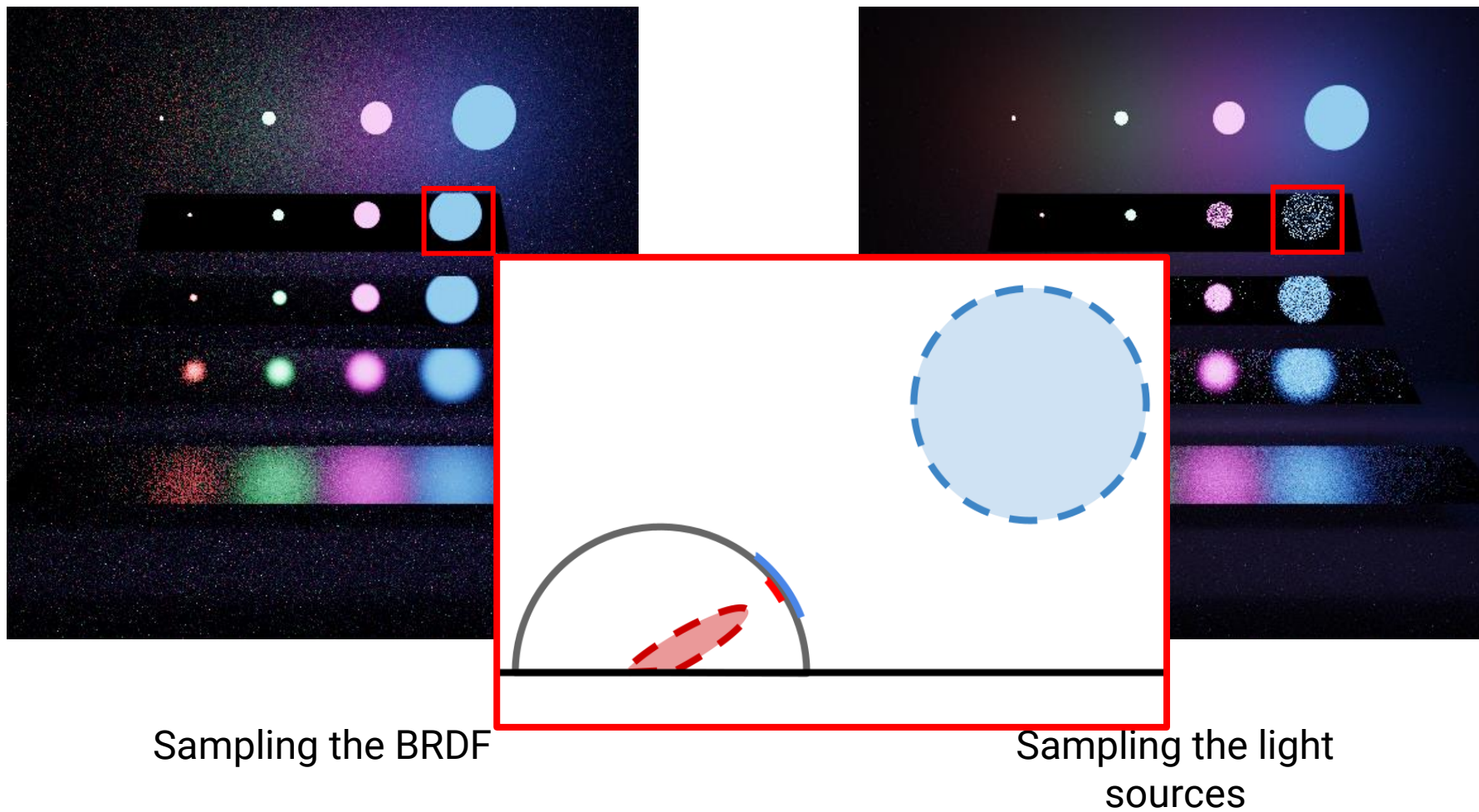
— Integrand — PDF_A — PDF_B

Balance heuristic weights



— w_B — w_A

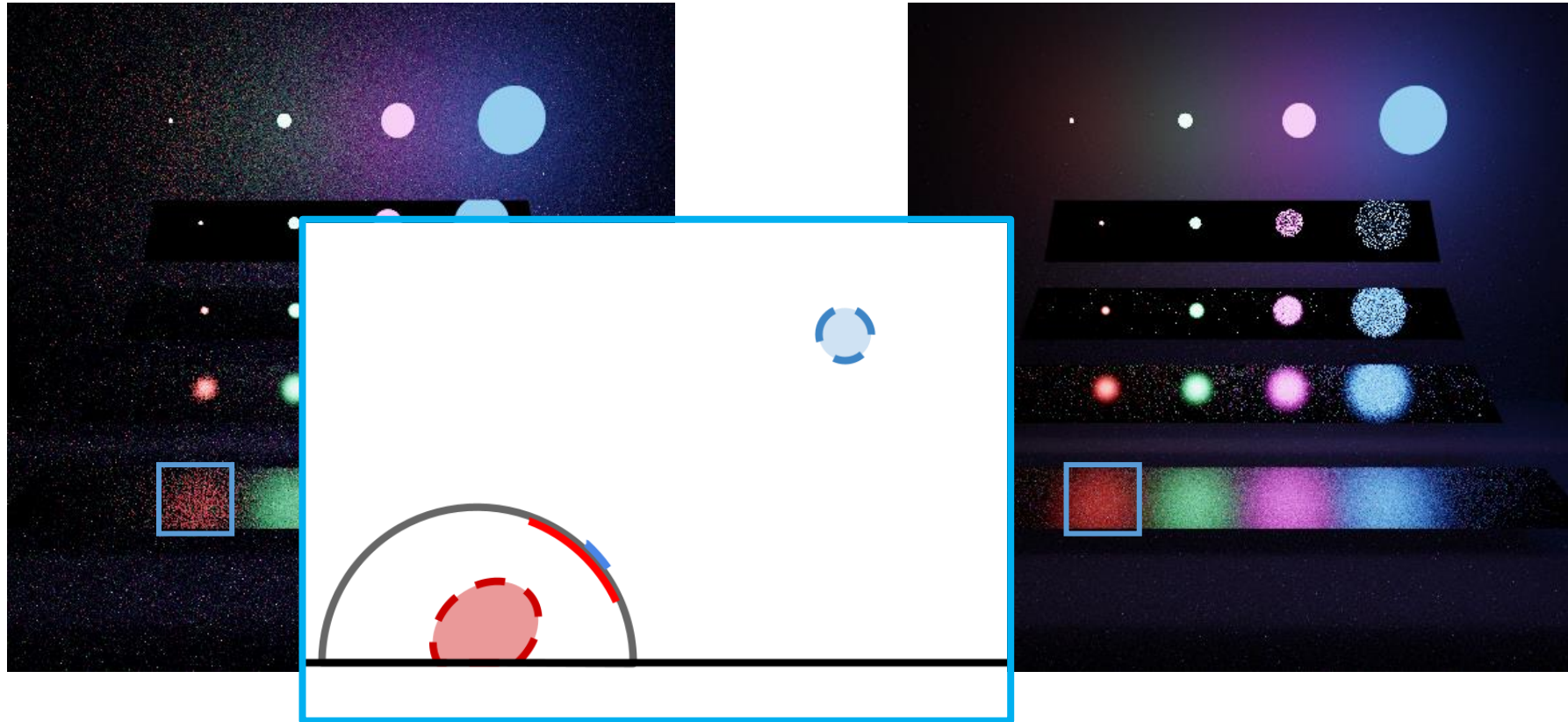
Application to rendering



Sampling the BRDF

Sampling the light sources

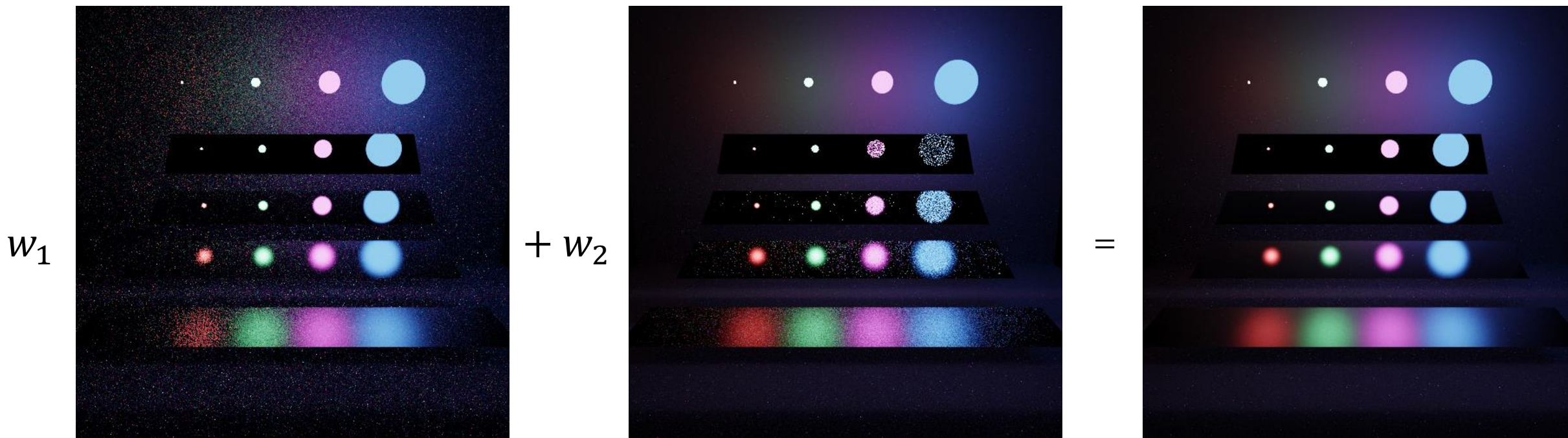
Application to rendering



Sampling the BRDF

Sampling the light sources

Application to rendering



Sampling the BRDF

Sampling the light sources

MIS

Balance heuristic and optimality

- Provably good [Veach & Guibas, 1995], but not optimal
- Minimizes an upper bound of the variance
- Ignores sample / technique correlation
- Performs poorly if some techniques have low variance

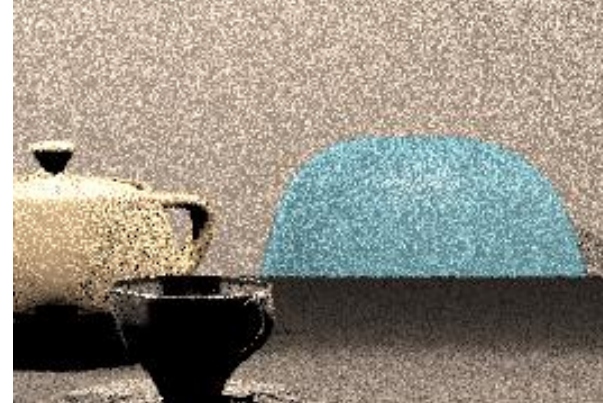
Minimized by optimal weights

$$\sigma_t^2 = \int_{\Omega} \frac{f^2(x)}{n_t p_t(x)} dx - r_t$$

Minimized by balance



Path tracer



Path tracer + **Bidir.**
(Balance heuristic)

Power and maximum heuristics

- Amplify weights where one density is higher
- If high density correlates with low variance, that improves the “low variance” issue

Power heuristic

$$w_t(x) = \frac{(n_t p_t(x))^p}{\sum_k (n_k p_k(x))^p}$$

Maximum heuristic

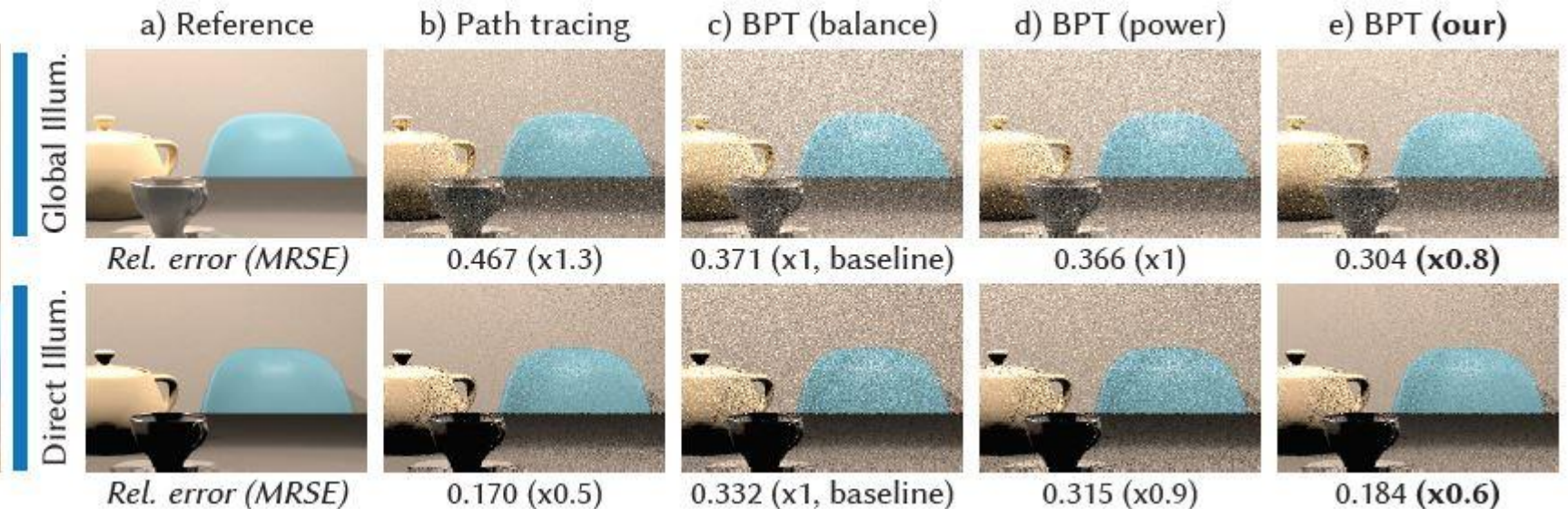
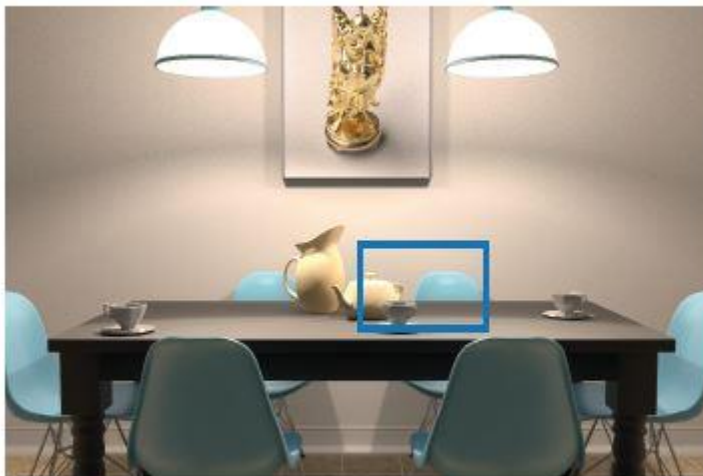
$$w_t(x) = \begin{cases} 1, & n_t p_t(x) > n_k p_k(x) \forall k \\ 0, & \text{else} \end{cases}$$

Variance-aware balance heuristic

- Estimate second moment and variance
- Use to offset the weights in the right direction
- By [Grittmann et al., 2019]

$$v_t = \frac{\int_{\Omega} \frac{f^2(x)}{n_t p_t(x)} dx}{\sigma_t^2}$$

$$w_t(x) = \frac{v_t n_t p_t(x)}{\sum_k v_k n_k p_k(x)}$$



Optimal MIS weights: complex and expensive, but worth it

Minimized by optimal weights

$$\sigma_t^2 = \int_{\Omega} \frac{f^2(x)}{n_t p_t(x)} dx - r_t$$

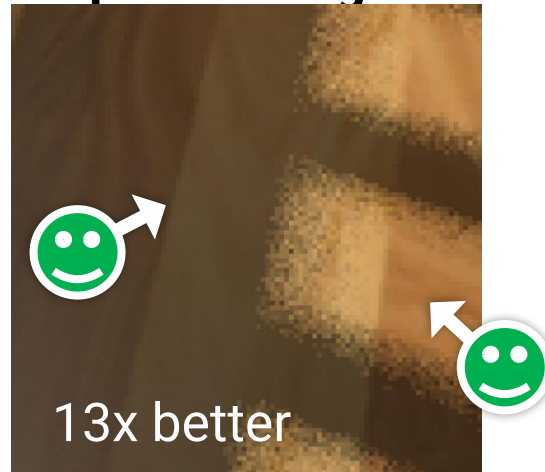
Minimized by balance

$$a_{ij} = \int \frac{p_i p_j}{\sum n_k p_k}$$

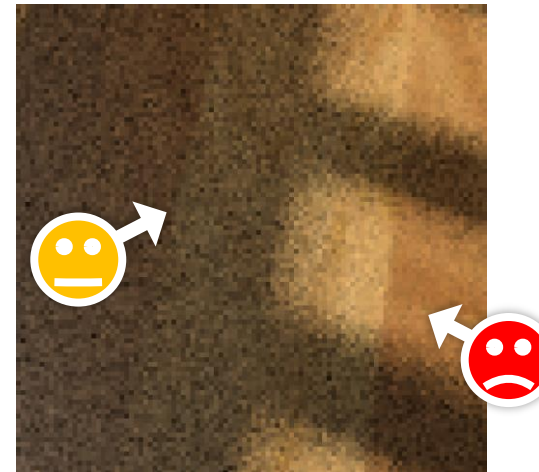
$$b_i = \int \frac{p_i f}{\sum n_k p_k}$$

$$\begin{pmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & a_{NN} \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_N \end{pmatrix}$$

Optimal weights



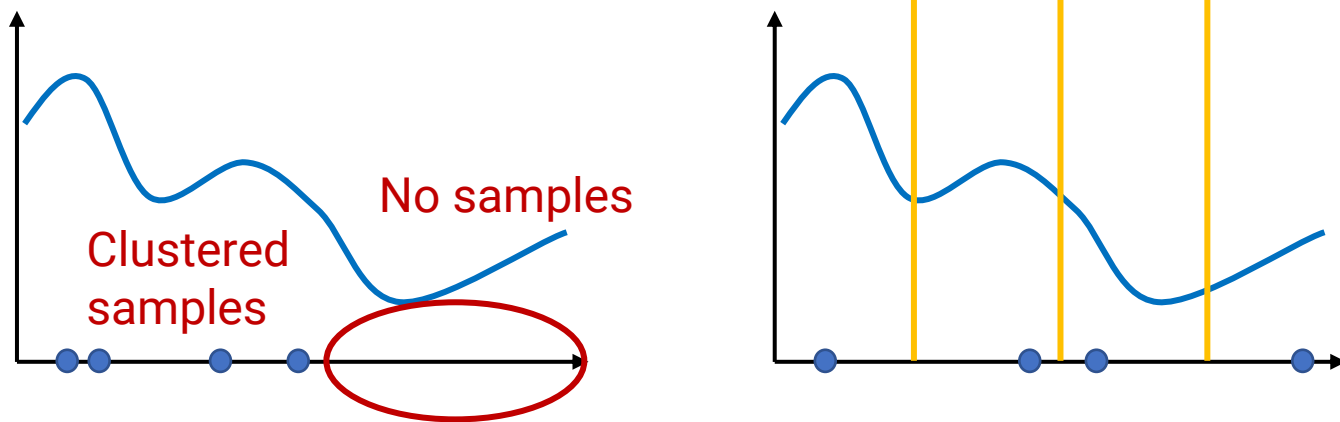
Balance heuristic



By [Kondapaneni et al., 2019]

Stratified sampling

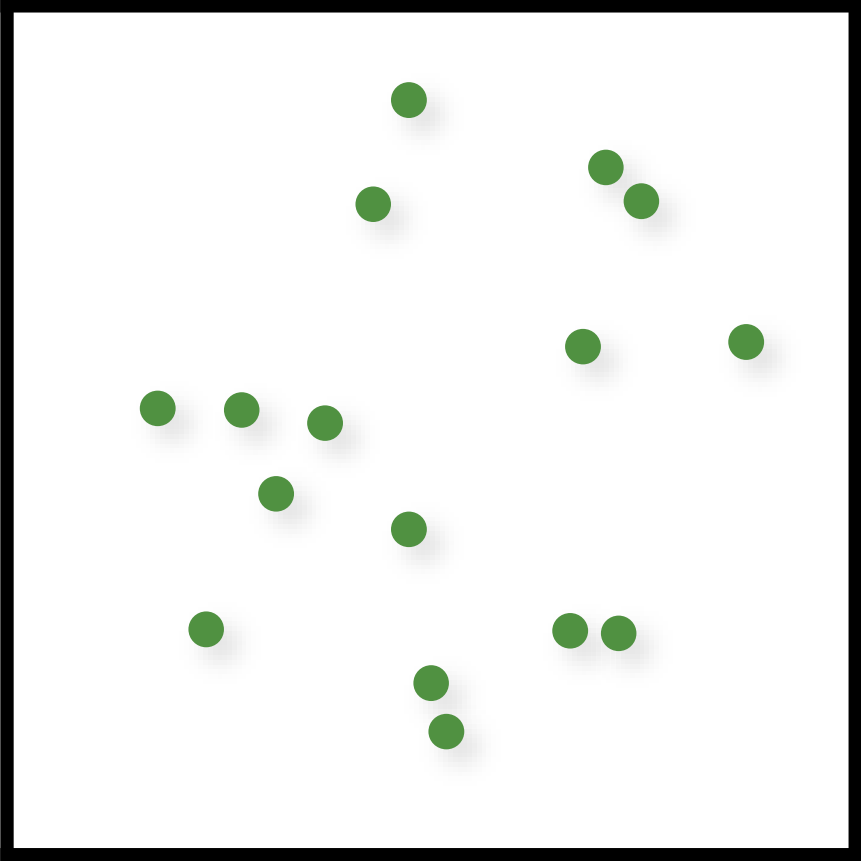
- Subdivide domain into bins
- Sample within each bin
- Less sample clustering → Guarantees that all regions are explored
- Lower variance!



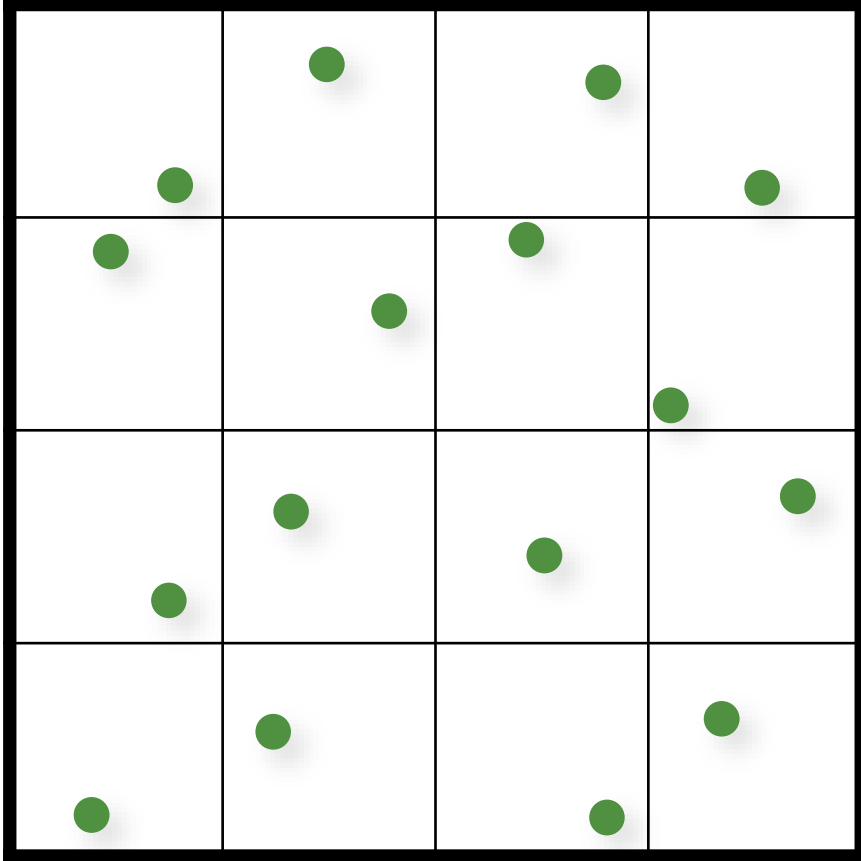
- Discussed in more depth later in the course

Stratified sampling

Random 2D



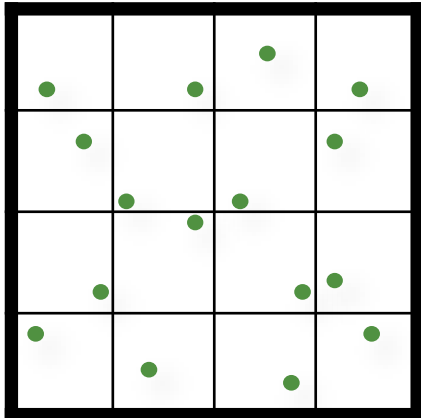
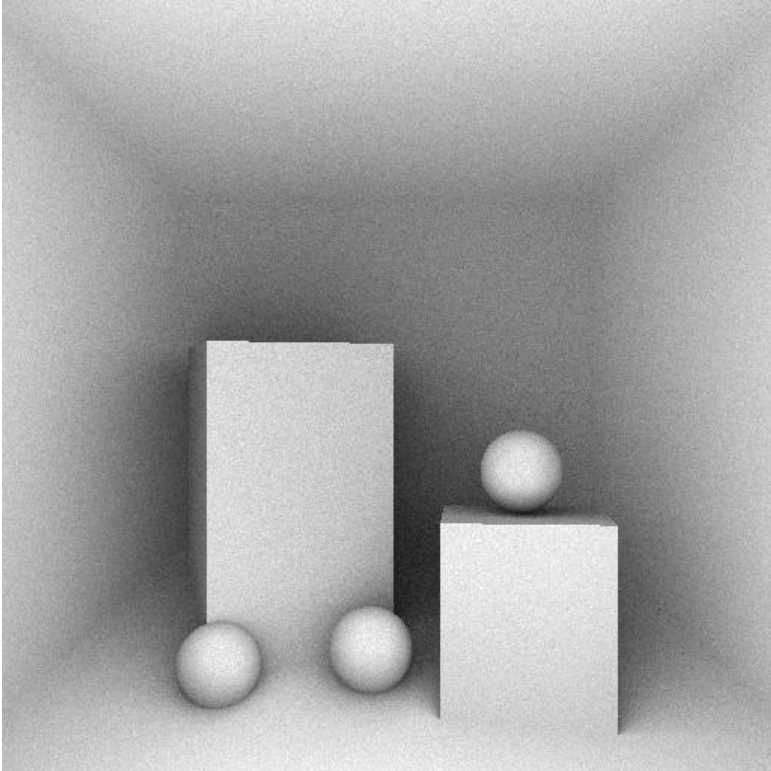
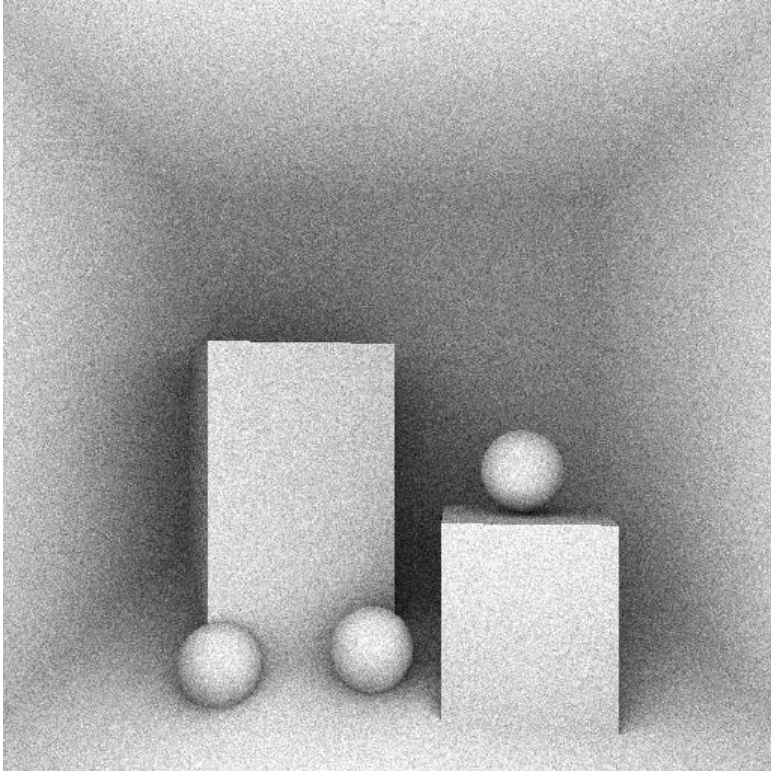
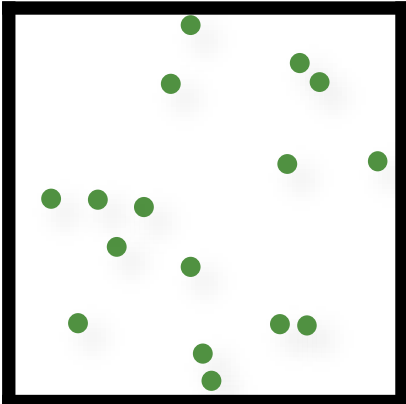
Jittered 2D



Stratified sampling

Random 2D

Jittered 2D

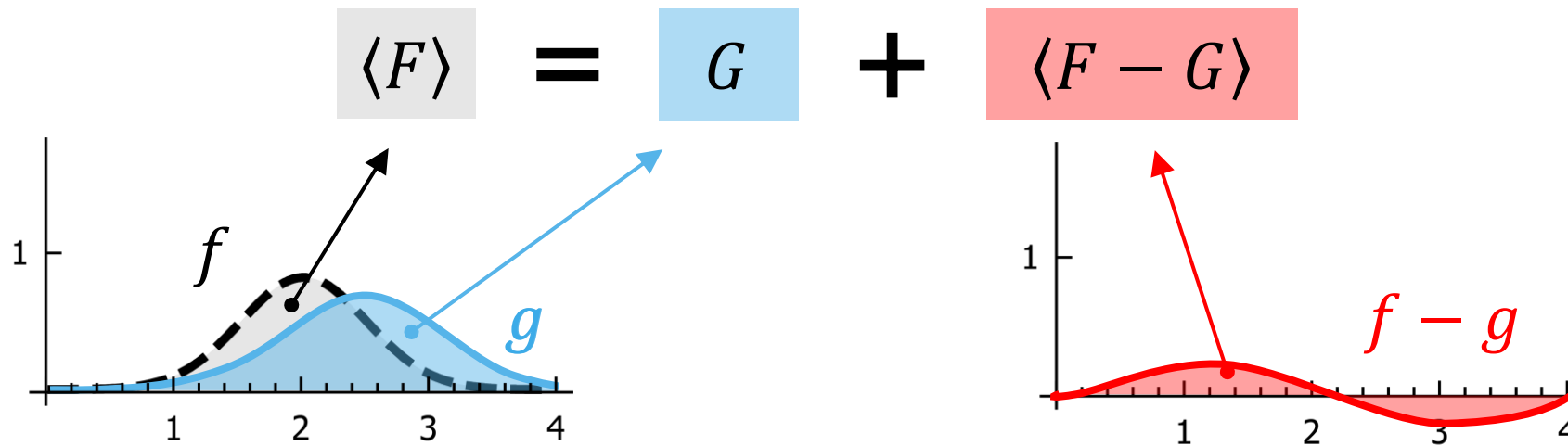


64 spp

Stratified sampling suffers from the curse of dimensionality

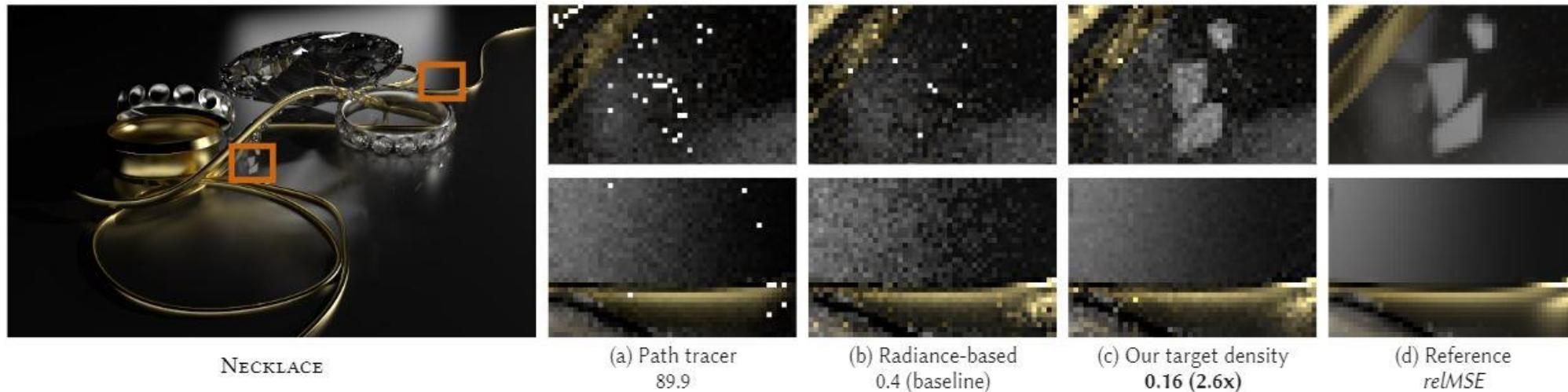
Control Variates

- Start with known integral
- Estimate difference between that and the target
- Can have lower variance if chosen well



Learning to importance sample (“Sample Guiding”)

- Sample in multiple iterations
- Use data from first iteration(s) to learn a PDF that is closer to the optimal one
- Discussed more in-depth later in the course



By [Grittmann et al., 2020]

Summary

What have we learned today?

This lecture

- Rendering equation: recursive integral, infinite dimensionality
- Analytical solution not possible
- Monte Carlo integration: numerical integration method that scales well with dimensionality
- Like “normal” quadrature but with random positions
- Efficient, scales, well, very flexible (many tweak, tricks, improvements possible)

Next up: Apply MC to rendering!

- Path tracing
- Bidirectional path tracing
- Density estimation
- Combinations via MIS
- Learned importance sampling
- Filtering and denoising
- Quasi-Monte Carlo and sampling patterns
- Markov chain Monte Carlo