Monte Carlo Integration

COS 302, Fall 2020

PRINCETON UNIVERSITY

Numerical Integration Problems

- Basic 1D numerical integration
  - Given ability to evaluate $f(x)$ for any x, find

$$\int_a^b f(x)\,dx$$

  - Goal: best **accuracy** with fewest **samples** (# of times *f* is evaluated)
  - Classic problem – many analytic functions not integrable in closed form

$$G(x) = \int_{-\infty}^{x} e^{-t^2}\,dt$$

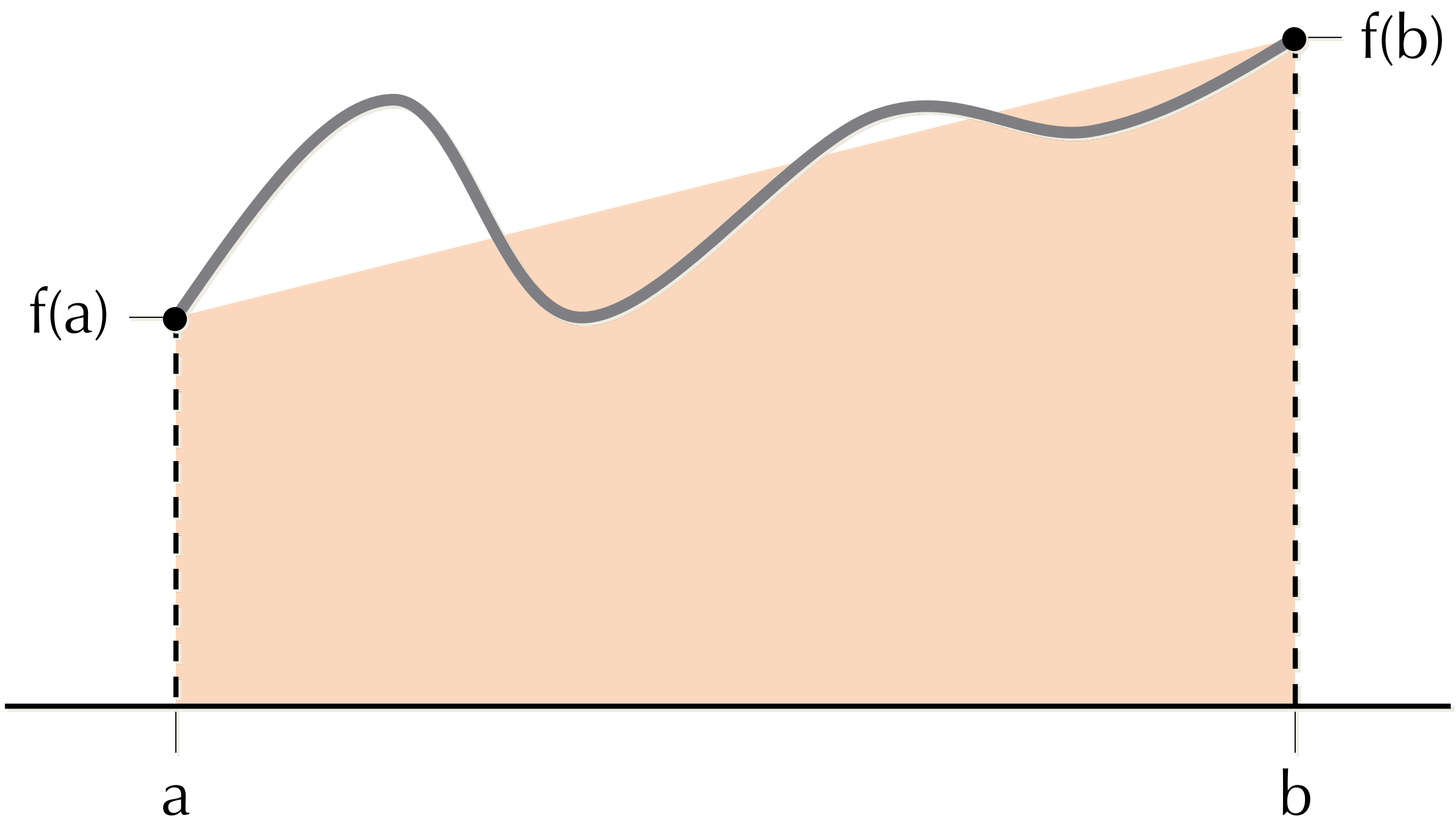
Quadrature

1. Sample $f(x)$ at a set of points
2. Approximate by a friendly function
3. Integrate approximating function

- Choices:
  - Which approximating function?
  - Which sampling points? ("nodes")
    - Even vs. uneven spacing?
  - Fit single function vs. multiple (piecewise)?

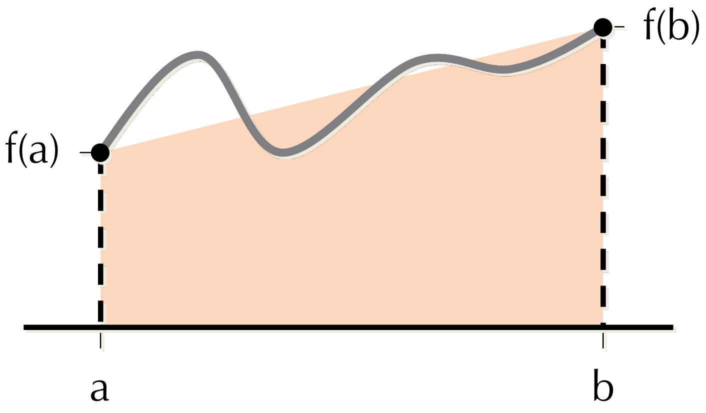
Trapezoidal Rule

- Approximate function by trapezoid



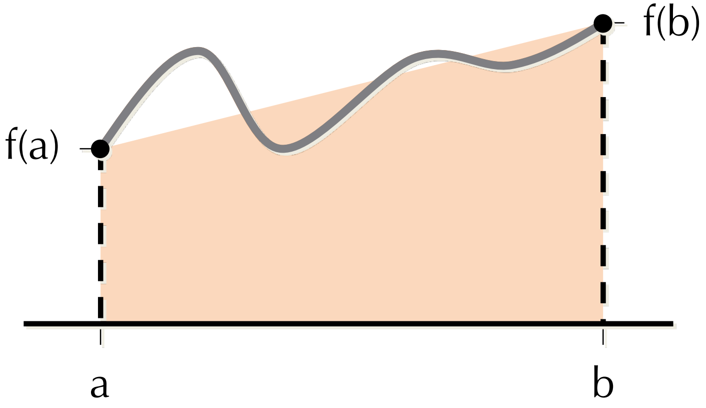
Trapezoidal Rule

$$\int_a^b f(x)\,dx \approx (b-a)\frac{f(a)+f(b)}{2}$$

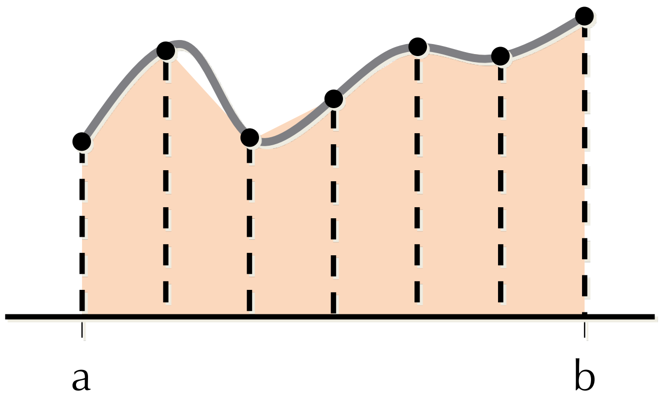


Extended Trapezoidal Rule

$$\int_a^b f(x)\,dx \approx (b-a)\frac{f(a)+f(b)}{2}$$



Divide into segments of width h, piecewise trapezoidal approximation



$$\int_a^b f(x)\,dx \approx h\left(\tfrac{1}{2}f(a)+f(x_1)+\cdots+f(x_{n-1})+\tfrac{1}{2}f(b)\right)$$

Open Methods

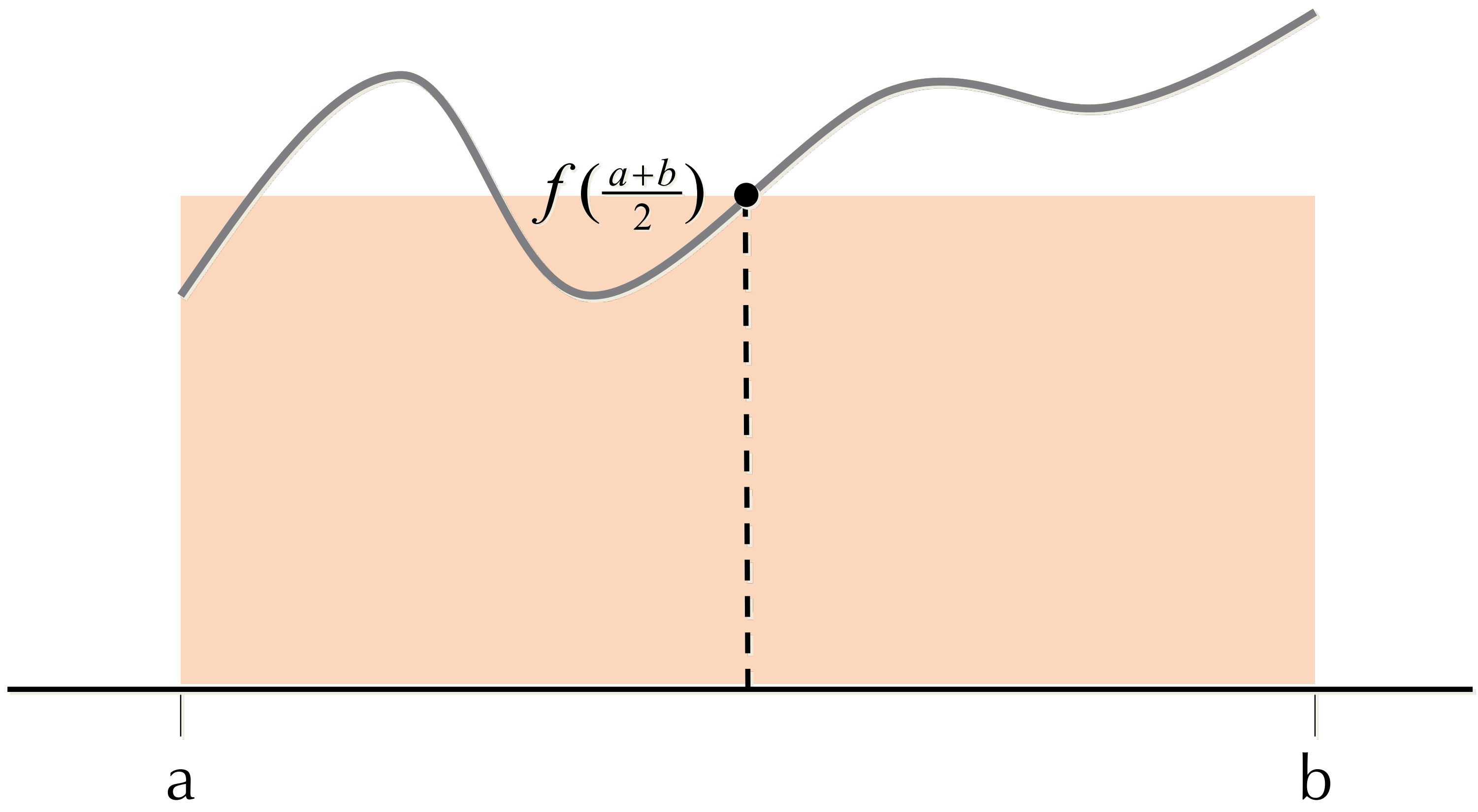
- Trapezoidal rule won't work if function undefined at one of the points where evaluating
  - Common example: function infinite at an endpoint

$$\int_0^1 \frac{dx}{x^2}$$

- Open methods only evaluate function on the *open* interval (i.e., not at endpoints)

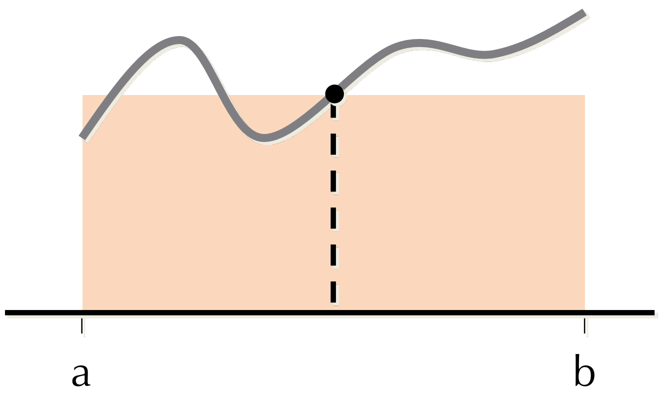
Midpoint Rule

- Approximate function by rectangle evaluated at midpoint

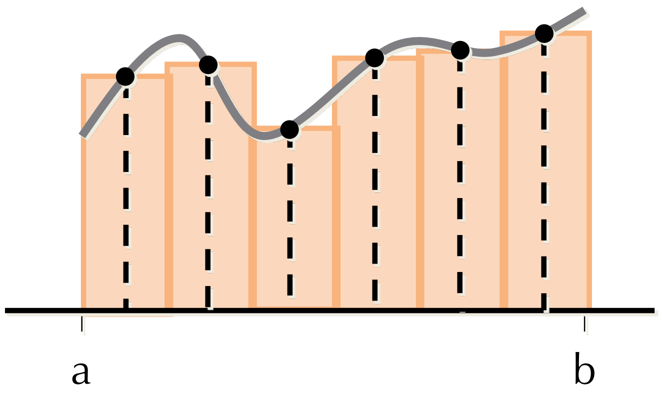


Extended Midpoint Rule

$$\int_a^b f(x)\,dx \approx (b-a)\,f\left(\tfrac{a+b}{2}\right)$$



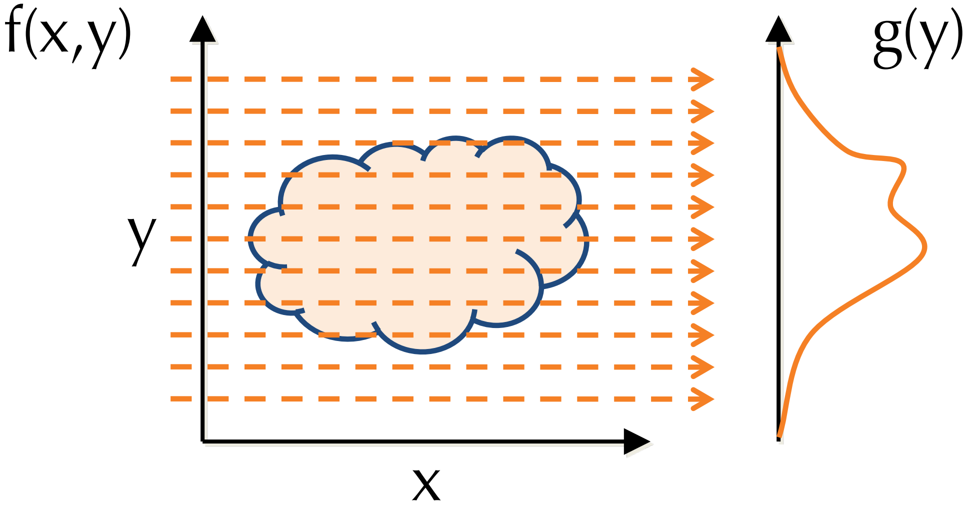
Divide into segments of width h:



$$\int_a^b f(x)\,dx \approx h\left(f\left(a+\tfrac{h}{2}\right)+f\left(a+\tfrac{3h}{2}\right)+\cdots+f\left(b-\tfrac{h}{2}\right)\right)$$

Integration in d Dimensions?

- One option: nested 1-D integration



$$\iint f(x,y)\,dx\,dy = \int g(y)\,dy$$

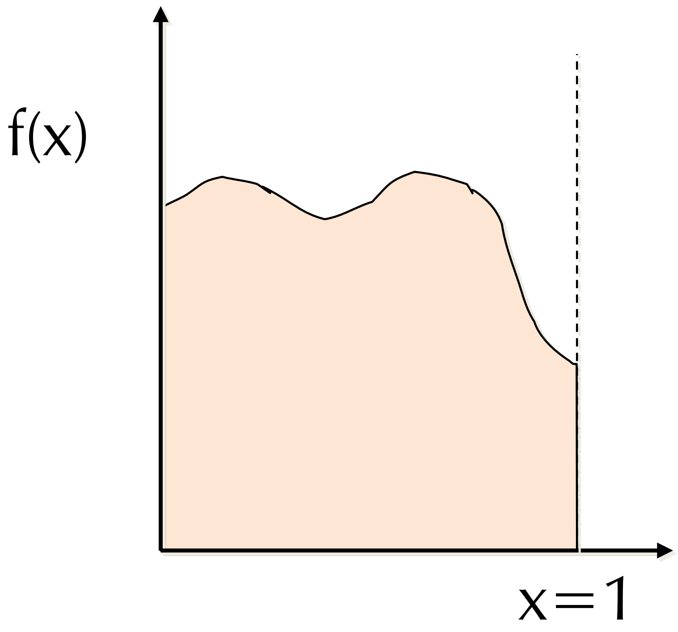
Evaluate the latter numerically, but each "sample" of g(y) is itself a 1-D integral, evaluated using a nested call to a numerical method

Integration in *d* Dimensions?

- Midpoint / trapezoid / other quadrature rule in *d* dimensions?
  - In 1D: (b-a)/h points
  - In 2D: $(b-a)/h^2$ points
  - In general: $O(1/h^d)$ points
- Required # of points ***grows exponentially with dimension***
  - "Curse of dimensionality"
- Other problems, e.g. non-rectangular domains

Rethinking Integration in 1D

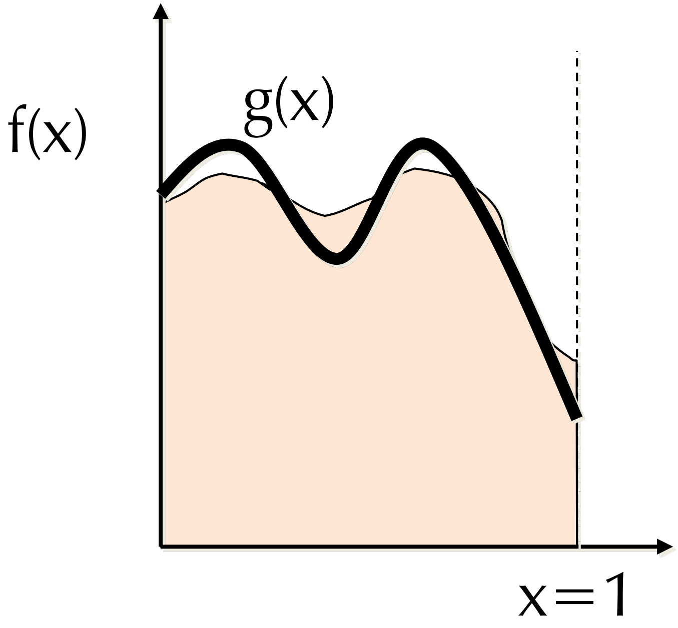
$$\int_0^1 f(x)dx = ?$$



Slide courtesy of
Peter Shirley

We Can Approximate…

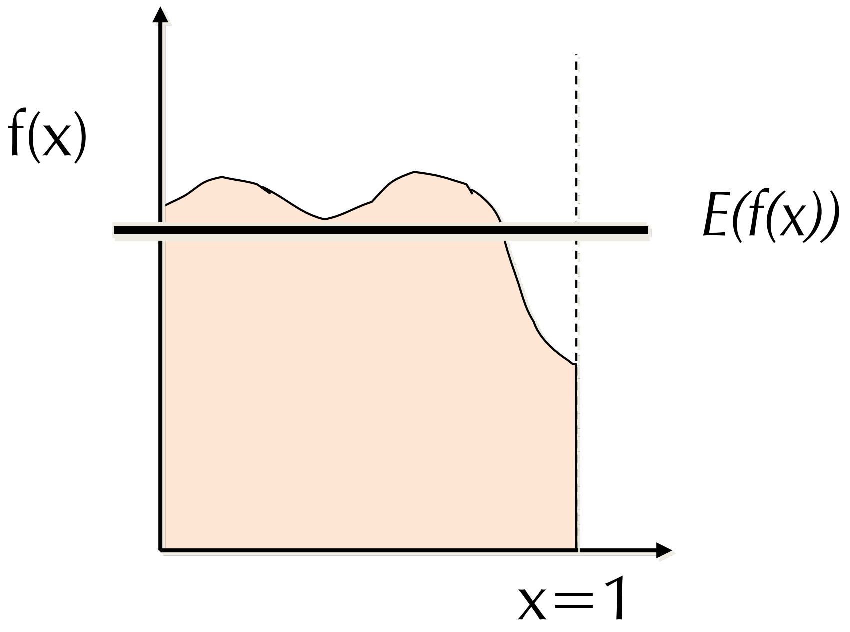
$$\int_0^1 f(x)dx = \int_0^1 g(x)dx$$



Slide courtesy of
Peter Shirley

Or We Can Average

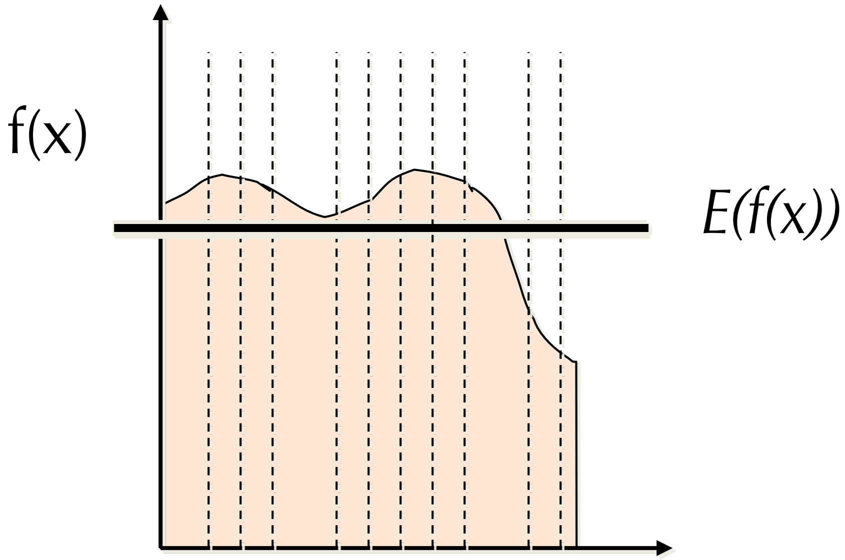
$$\int_0^1 f(x)dx = E(f(x))$$



Slide courtesy of
Peter Shirley

Estimating the Average

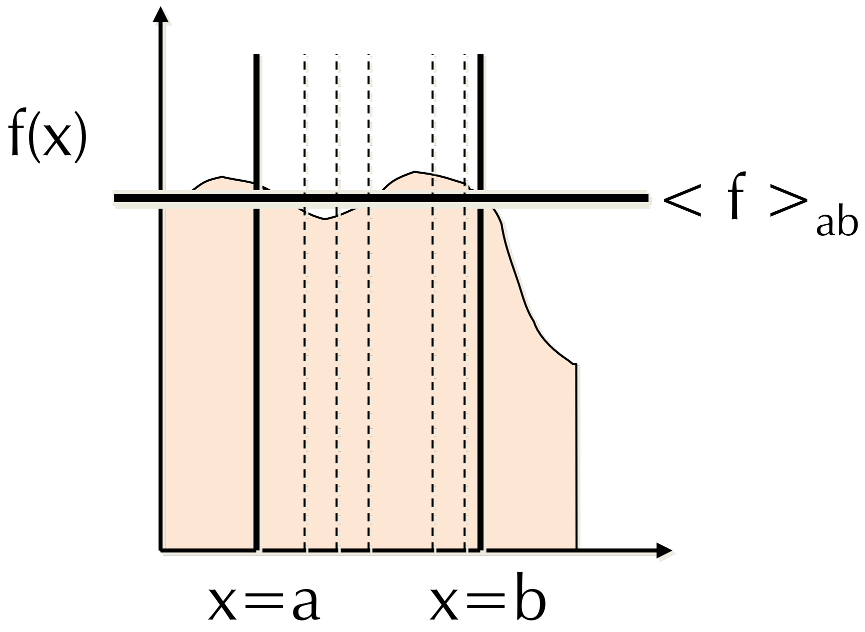
$$\int_0^1 f(x)dx \cong \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$



Slide courtesy of
Peter Shirley

Other Domains

$$\int_a^b f(x)dx \cong \frac{b-a}{N}\sum_{i=1}^{N} f(x_i)$$



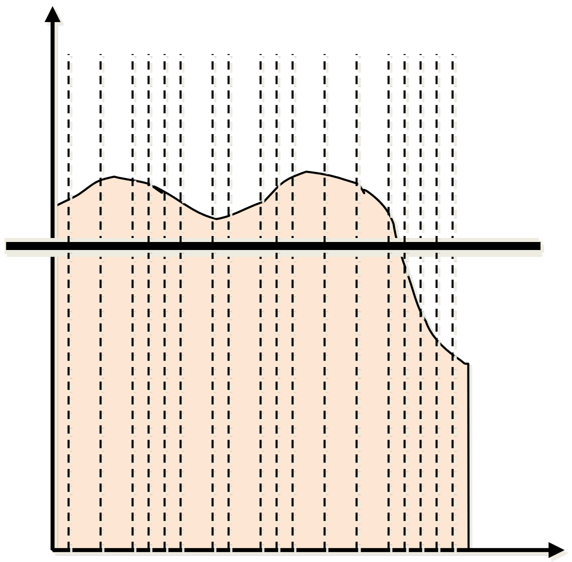
Slide courtesy of
Peter Shirley

"Monte Carlo" Integration

- No "exponential explosion" in required number of samples with increase in dimension
- (Some) resistance to badly-behaved functions

Le Grand Casino de Monte-Carlo

Variance



$$\int_a^b f(x)dx \cong \frac{b-a}{N}\sum_{i=1}^{N} f(x_i)$$

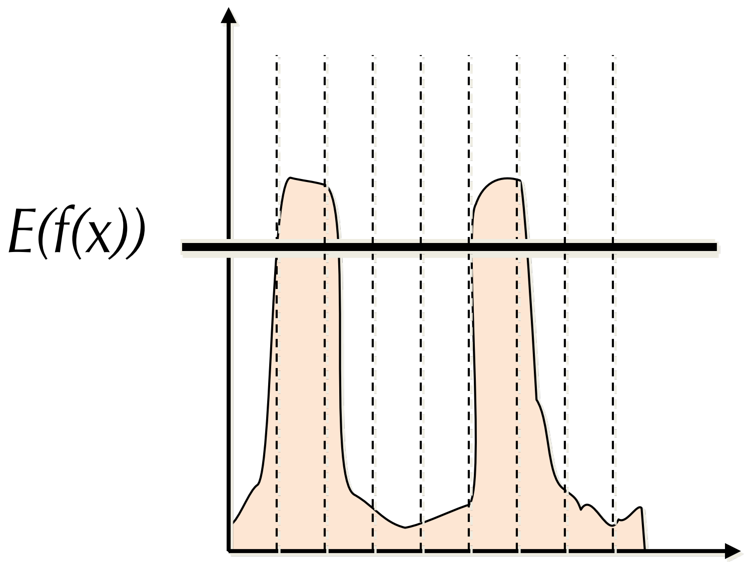
$$Var\left[\frac{b-a}{N}\sum_{i=1}^{N} f(x_i)\right] = \left(\frac{b-a}{N}\right)^2 \sum_{i=1}^{N} Var[f(x_i)]$$

$$= \frac{(b-a)^2}{N} Var[f(x_i)]$$

Variance decreases as 1/N
Error of E decreases as 1/sqrt(N)

Variance

- Problem: variance decreases with 1/N
  - Increasing # samples removes noise slowly



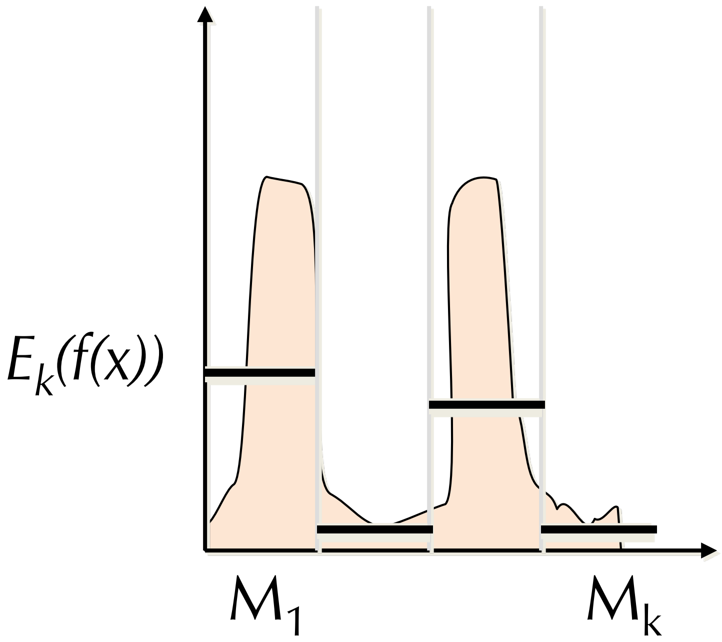
Variance Reduction Techniques

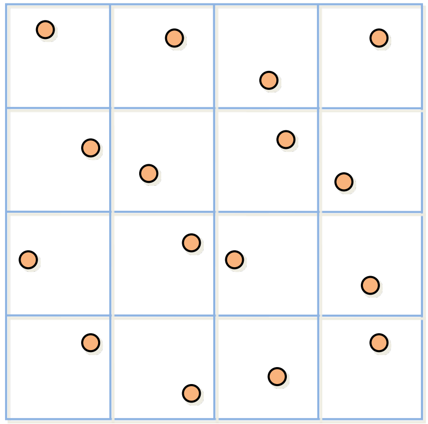
- Problem: variance decreases with 1/N
  - Increasing # samples removes noise slowly

- Variance reduction:
  - Stratified sampling
  - Importance sampling

Stratified Sampling

- Estimate subdomains separately

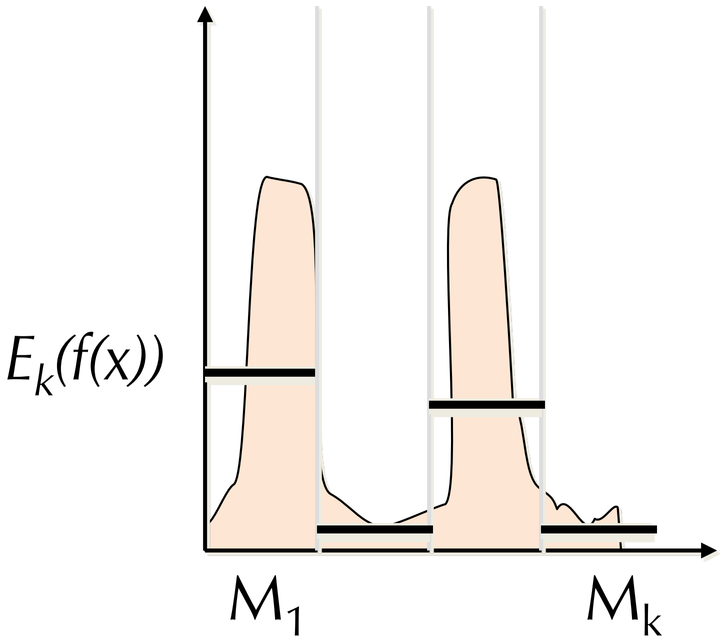




Can do this recursively!

Stratified Sampling

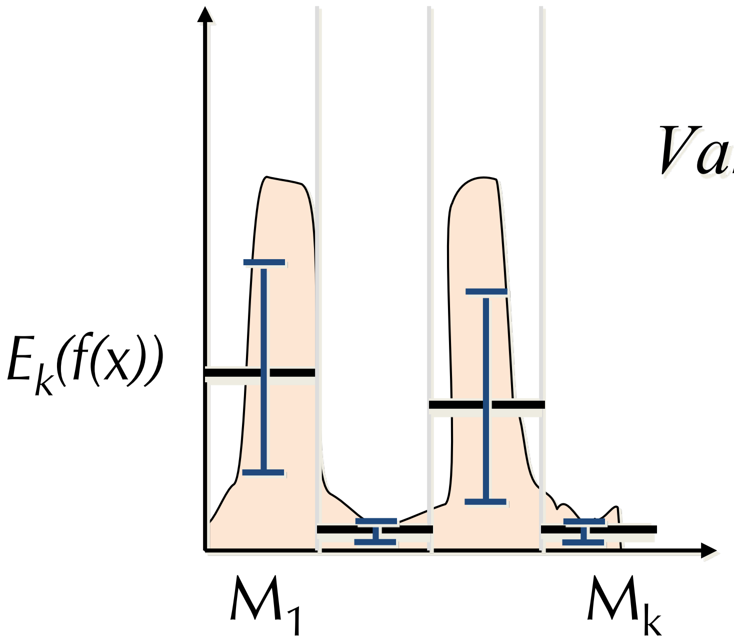
- This is still unbiased



$$E = \sum_{j=1}^{k} \frac{vol(M_j)}{N_j} \sum_{n=1}^{N_j} f(x_{jn})$$

Stratified Sampling

- Less overall variance if less variance in subdomains

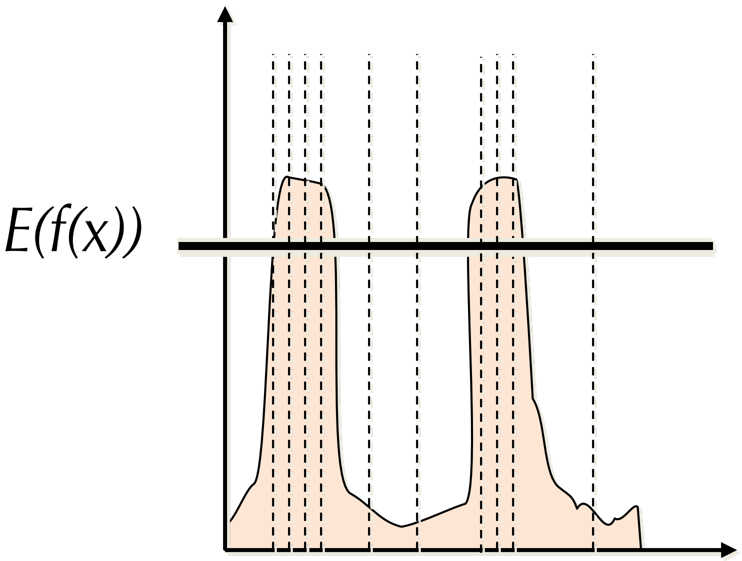


$$Var[E] = \sum_{j=1}^{k} \frac{vol(M_j)^2}{N_j} Var[f(x)]\Big|_{M_j}$$

Total variance minimized when number of points in each subvolume M_j proportional to error in M_j.

Importance Sampling

- Put more samples where f(x) is bigger



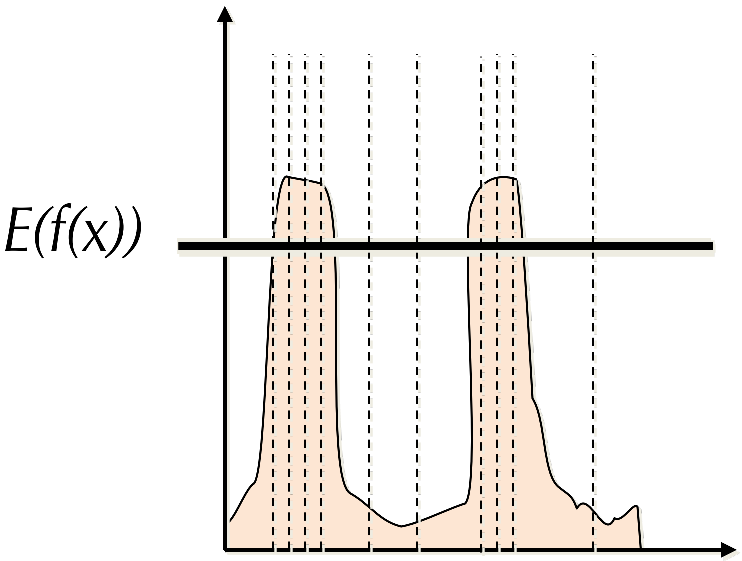
$$\int_{\Omega} f(x)dx = \frac{1}{N}\sum_{i=1}^{N} Y_i$$

where $Y_i = \frac{f(x_i)}{p(x_i)}$

and x_i drawn from $P(x)$

Importance Sampling

- This is still unbiased



$$E[Y_i] = \int_\Omega Y(x)p(x)dx$$

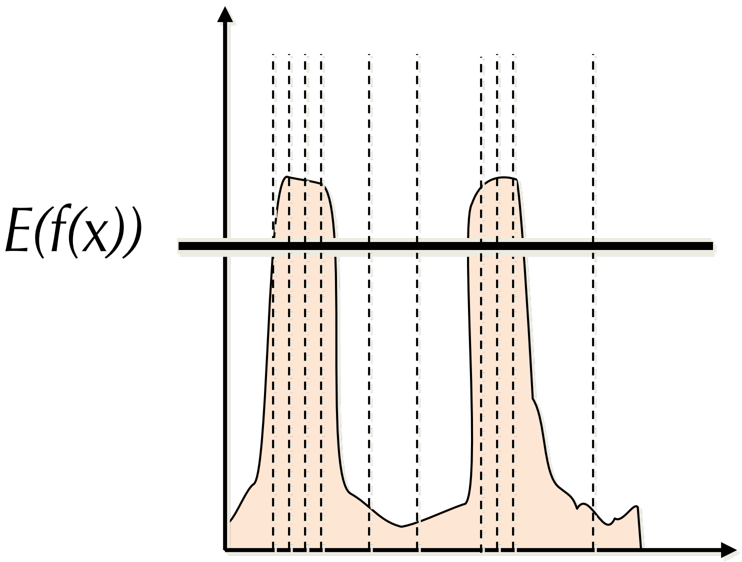
$$= \int_\Omega \frac{f(x)}{p(x)} p(x)dx$$

$$= \int_\Omega f(x)dx$$

for all N

Importance Sampling

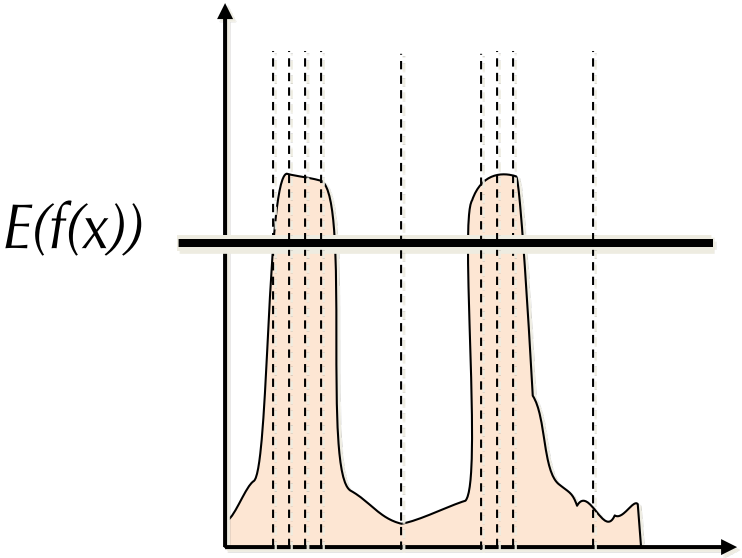
- Variance depends on choice of p(x):



$$Var(E) = \frac{1}{N}\sum_{n=1}^{N}\left(\frac{f(x_n)}{p(x_n)}\right)^2 - E^2$$

Importance Sampling

- Zero variance if p(x) ~ f(x)



$$p(x) = cf(x)$$

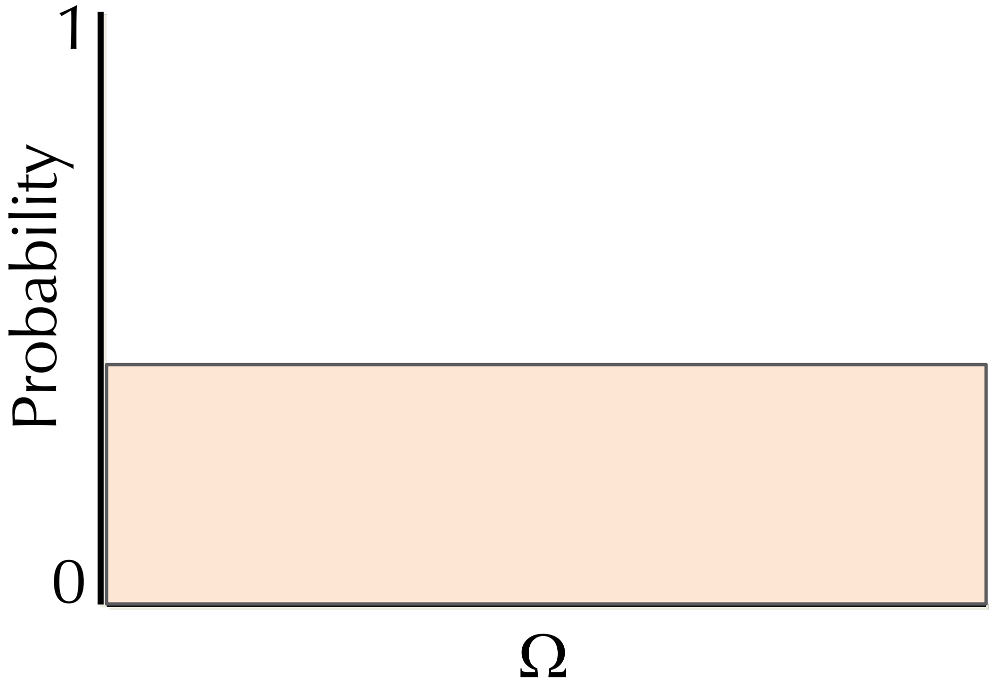
$$Y_i = \frac{f(x_i)}{p(x_i)} = \frac{1}{c}$$

$$Var(Y) = 0$$

Less variance with better importance sampling

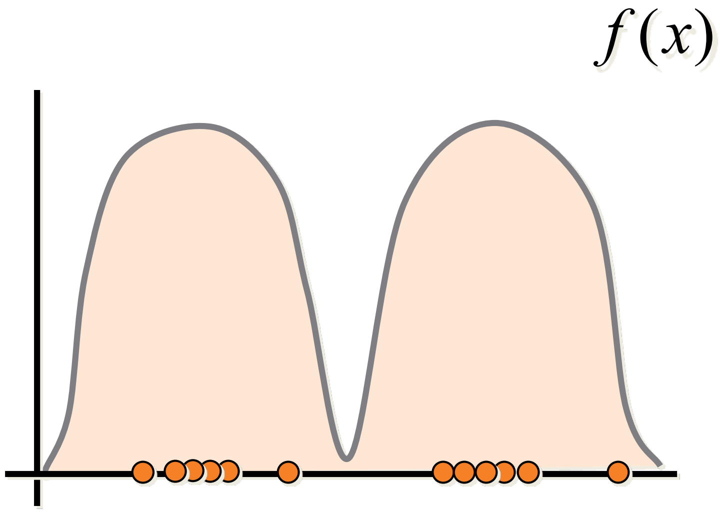
Generating Random Points

- Uniform distribution:
  - Use pseudorandom number generator



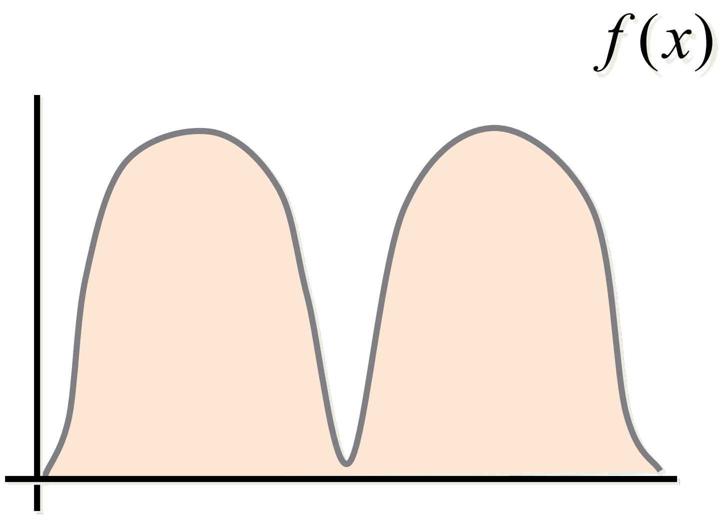
Sampling from a Non-Uniform Distribution

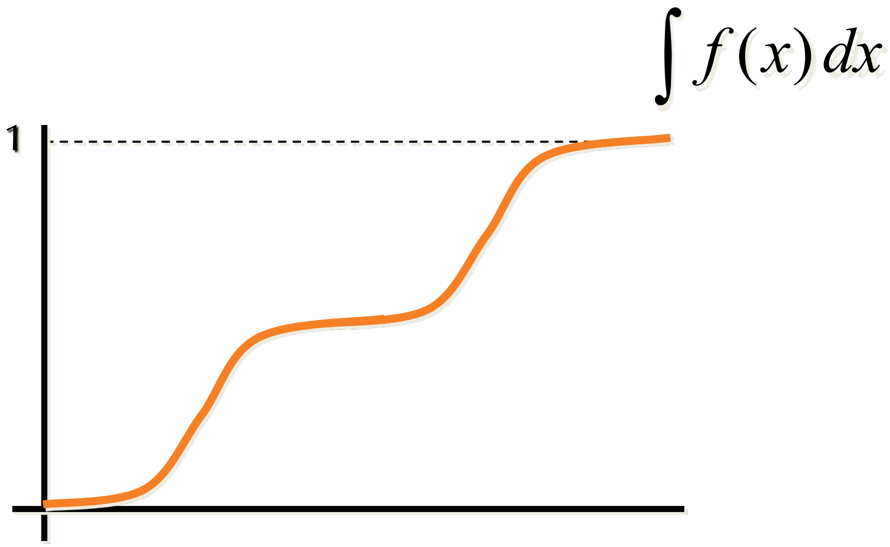
- Inversion method
- Rejection method



Sampling from a Non-Uniform Distribution

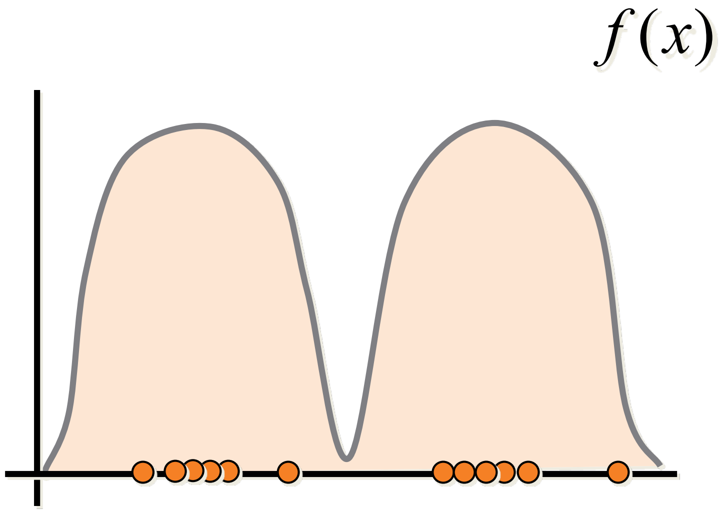
- Inversion method
  - Integrate *f*(*x*): Cumulative Distribution Function

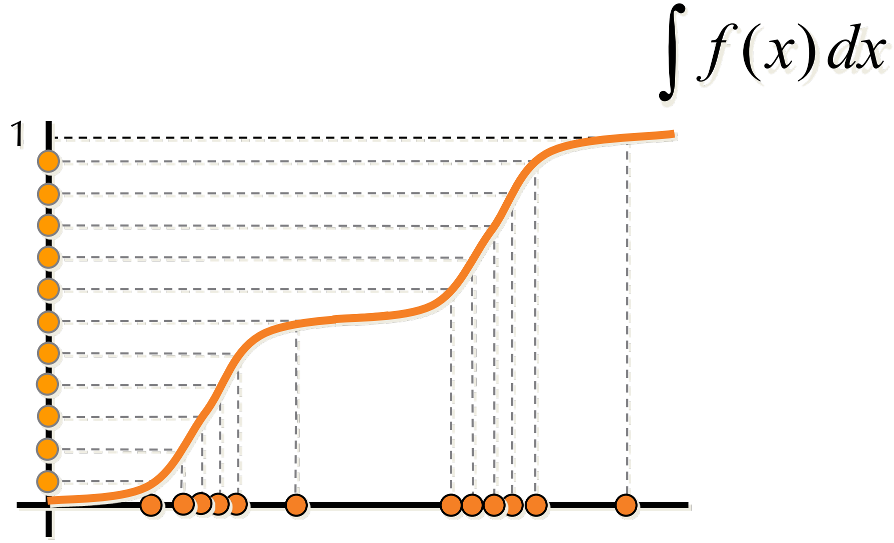




Sampling from a Non-Uniform Distribution

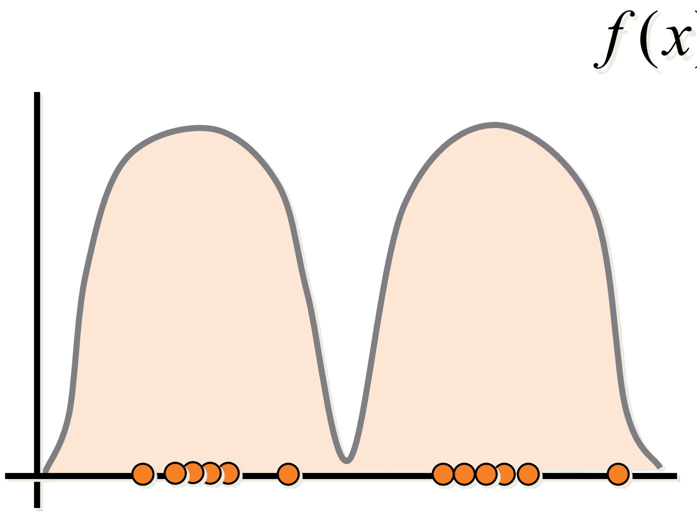
- Inversion method
  - Integrate *f*(*x*): Cumulative Distribution Function
  - Invert CDF, apply to uniform random variable





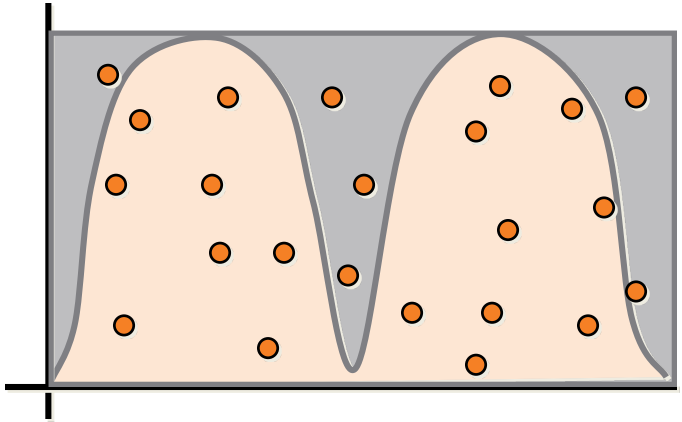
Sampling from a Non-Uniform Distribution

- Inversion method
- Rejection method



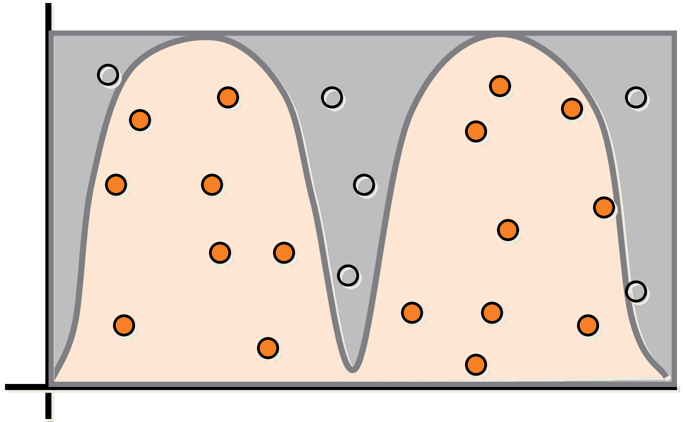
Sampling from a Non-Uniform Distribution

- Rejection method
  - Generate random (x,y) pairs, y between 0 and max(f(x))



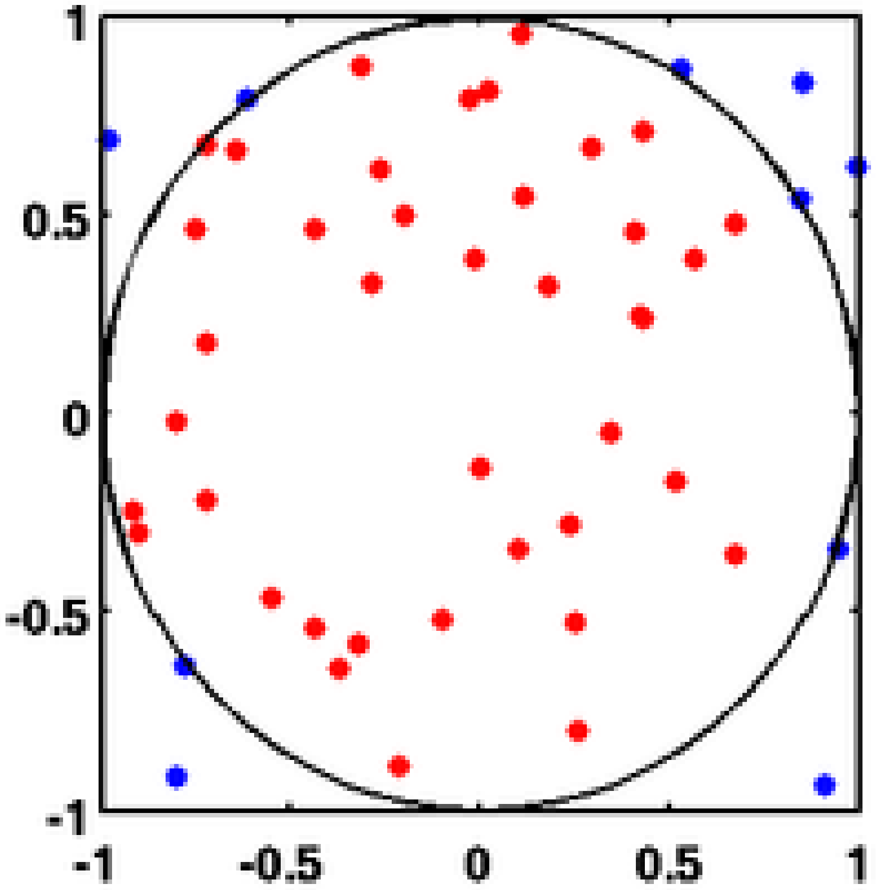
Sampling from a Non-Uniform Distribution

- Rejection method
  - Generate random (x,y) pairs, y between 0 and max(f(x))
  - Keep only samples where y < f(x)

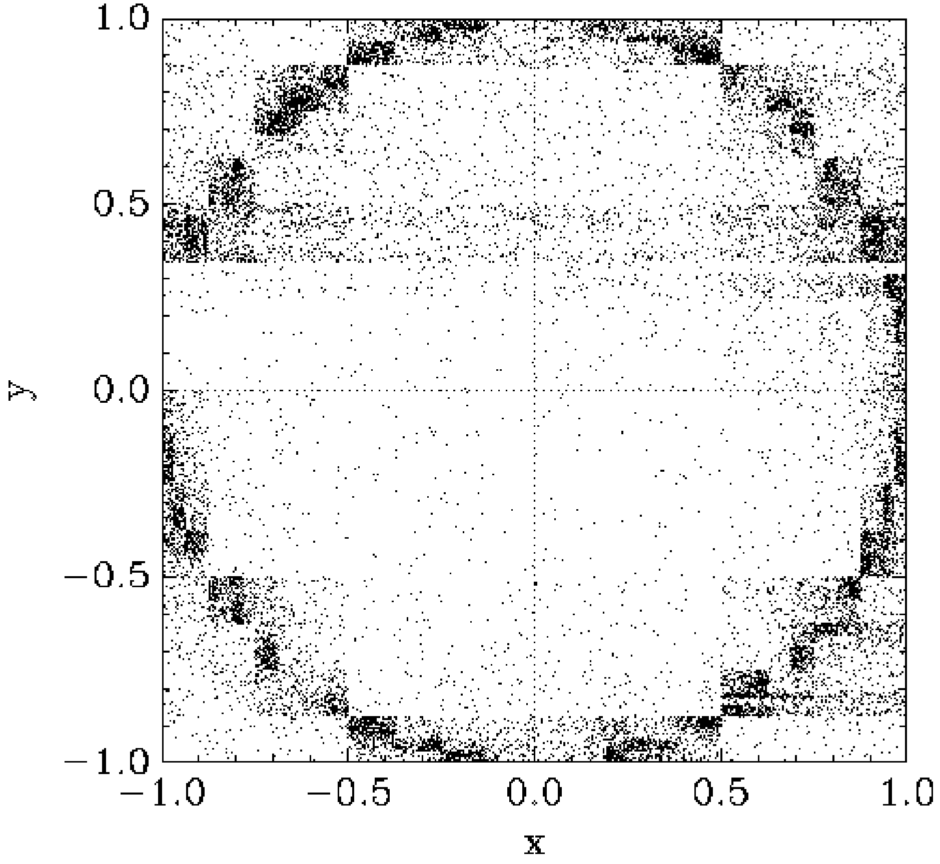


- Pro: does not require CDF, inversion
- Con: wastes samples

Example: Computing pi



With Stratified Sampling

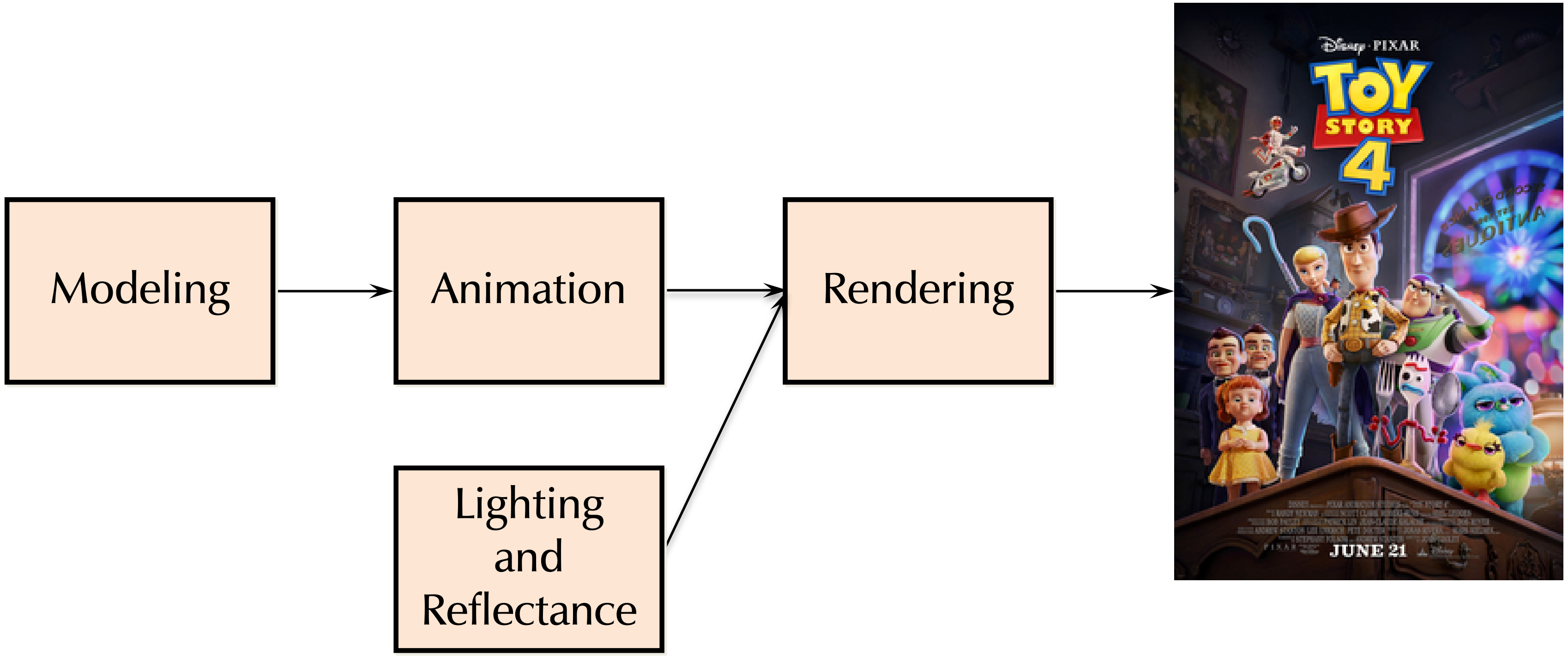


Monte Carlo in Computer Graphics

or, Solving Integral Equations for Fun and Profit

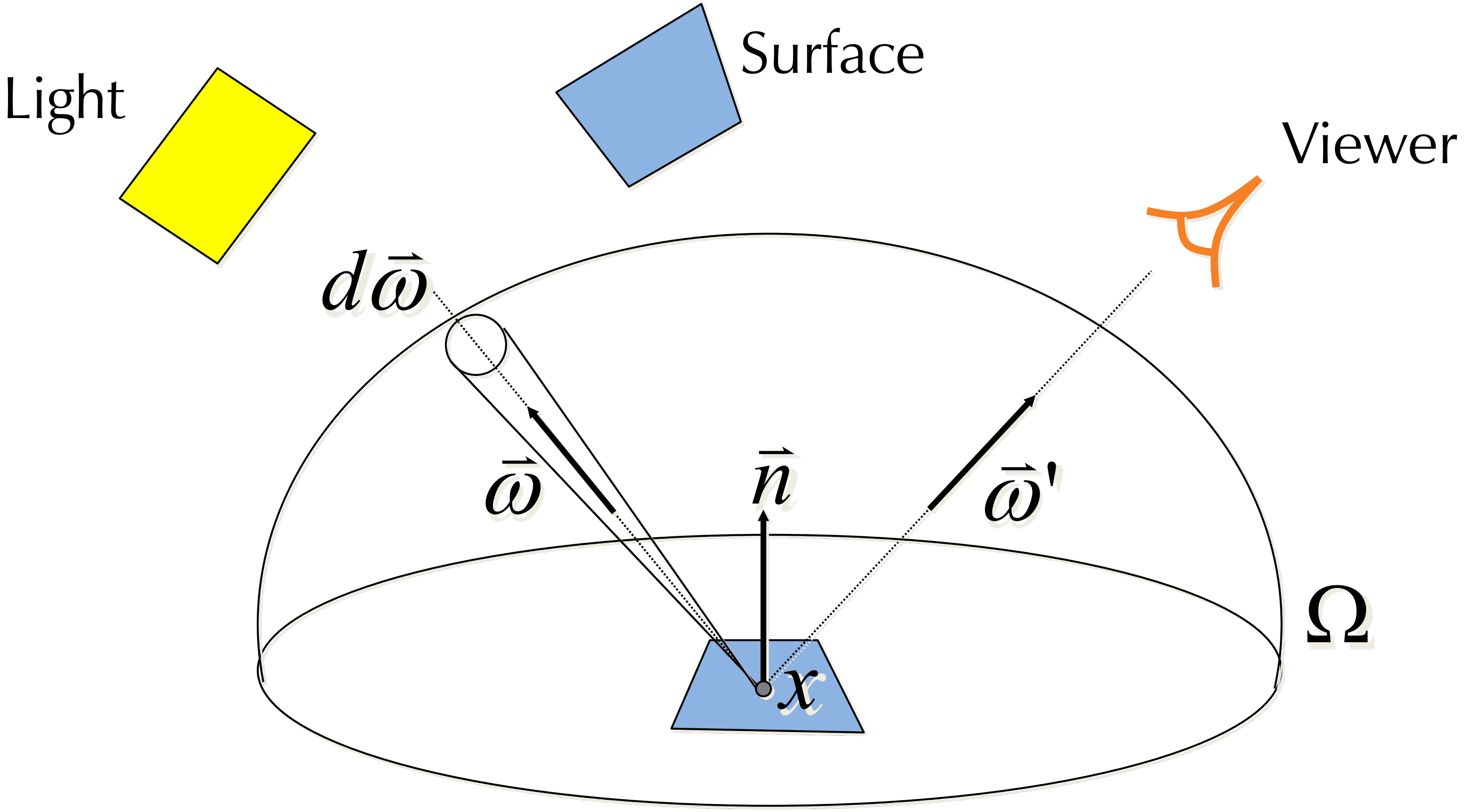
or, Ugly Equations, Pretty Pictures

Computer Graphics Pipeline



Rendering Equation

$$L_o(x,\vec{\omega}') = L_e(x,\vec{\omega}') + \int_{\Omega} L_i(x,\vec{\omega})\, f_r(x,\vec{\omega},\vec{\omega}')(\vec{\omega}\cdot\vec{n})\, d\vec{\omega}$$



[Kajiya 1986]

Rendering Equation

$$L_o(x,\vec{\omega}') = L_e(x,\vec{\omega}') + \int_{\Omega} L_i(x,\vec{\omega})\, f_r(x,\vec{\omega},\vec{\omega}')(\vec{\omega}\cdot\vec{n})\, d\vec{\omega}$$

- This is an *integral equation*
- Hard to solve!
  - Can't solve this in closed form
  - Simulate complex phenomena

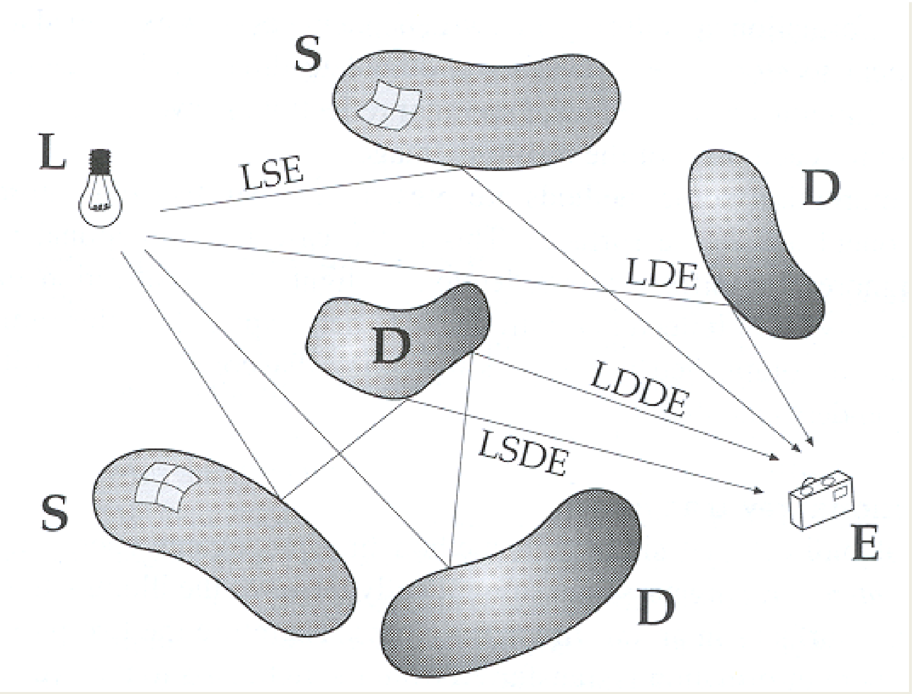
Rendering Equation

$$L_o(x,\vec{\omega}') = L_e(x,\vec{\omega}') + \int_{\Omega} L_i(x,\vec{\omega})\, f_r(x,\vec{\omega},\vec{\omega}')(\vec{\omega}\cdot\vec{n})\, d\vec{\omega}$$

- This is an *integral equation*
- Hard to solve!
  - Can't solve this in closed form
  - Simulate complex phenomena

Monte Carlo Path Tracing

Estimate integral
for each pixel
by random sampling

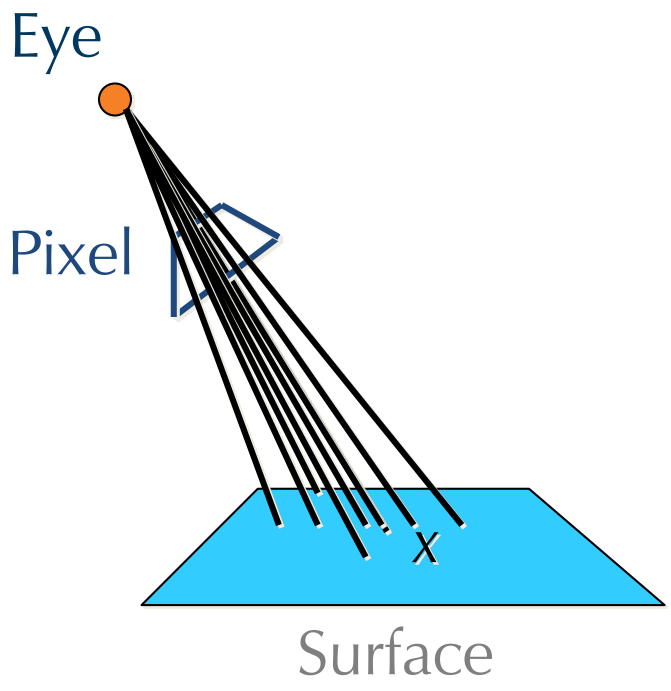


Monte Carlo Global Illumination

- Rendering = integration
  - Antialiasing
  - Soft shadows
  - Indirect illumination
  - Caustics

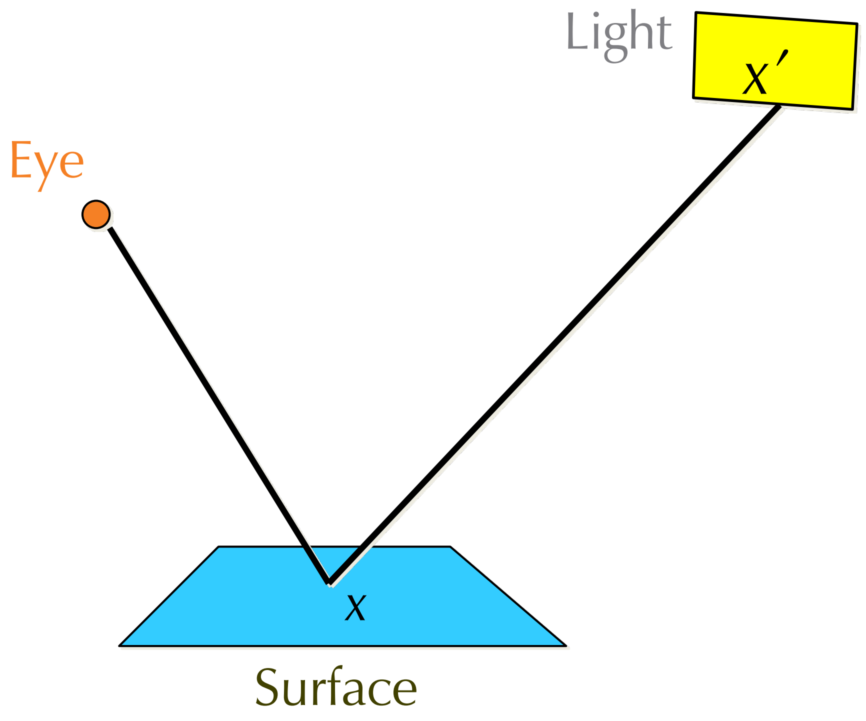
Monte Carlo Global Illumination

- Rendering = integration
  - Antialiasing
  - Soft shadows
  - Indirect illumination
  - Caustics



Monte Carlo Global Illumination

- Rendering = integration
  - Antialiasing
  - Soft shadows
  - Indirect illumination
  - Caustics

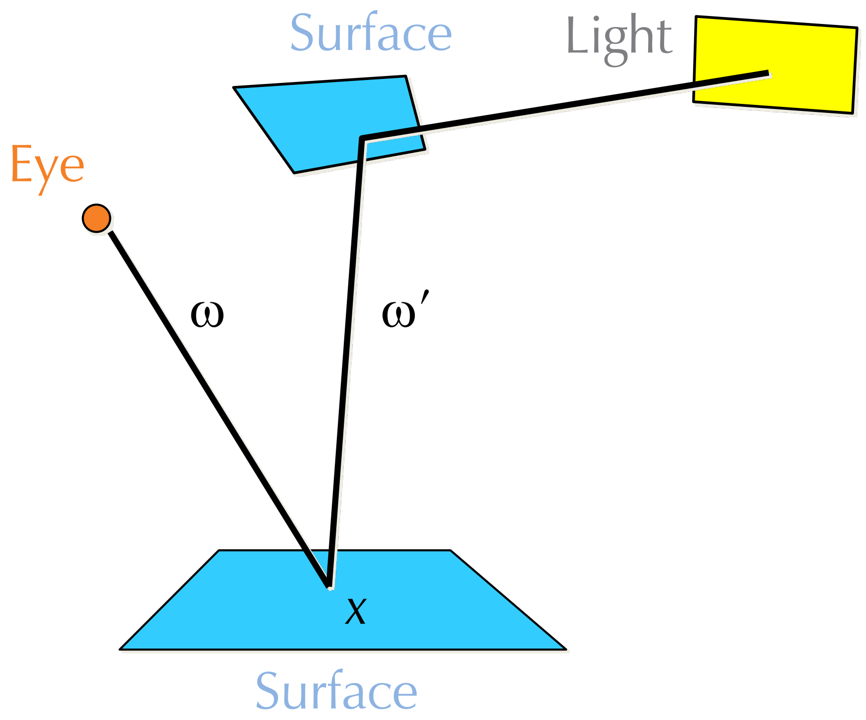


Monte Carlo Global Illumination

- Rendering = integration
  - Antialiasing
  - Soft shadows
  - Indirect illumination
  - Caustics

Monte Carlo Global Illumination

- Rendering = integration
  - Antialiasing
  - Soft shadows
  - Indirect illumination
  - Caustics

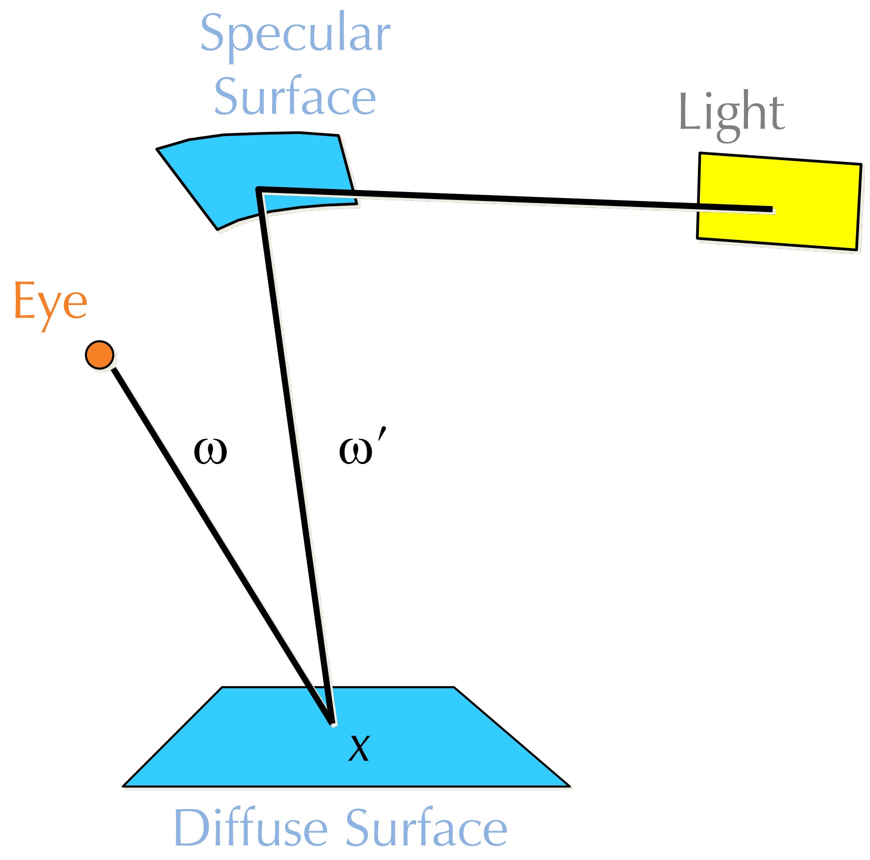


Monte Carlo Global Illumination

- Rendering = integration
  - Antialiasing
  - Soft shadows
  - Indirect illumination
  - Caustics

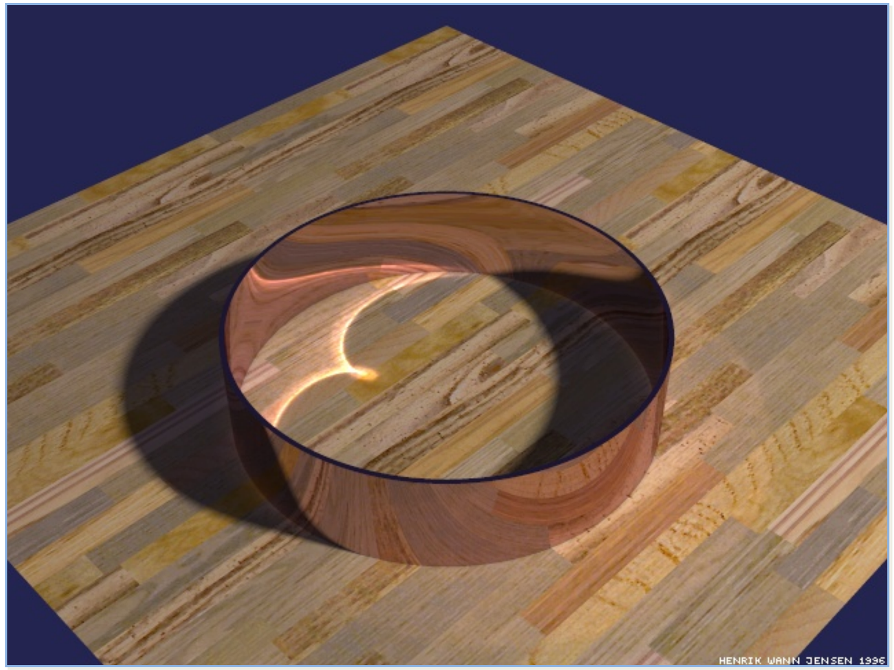
Monte Carlo Global Illumination

- Rendering = integration
  - Antialiasing
  - Soft shadows
  - Indirect illumination
  - Caustics



Monte Carlo Global Illumination

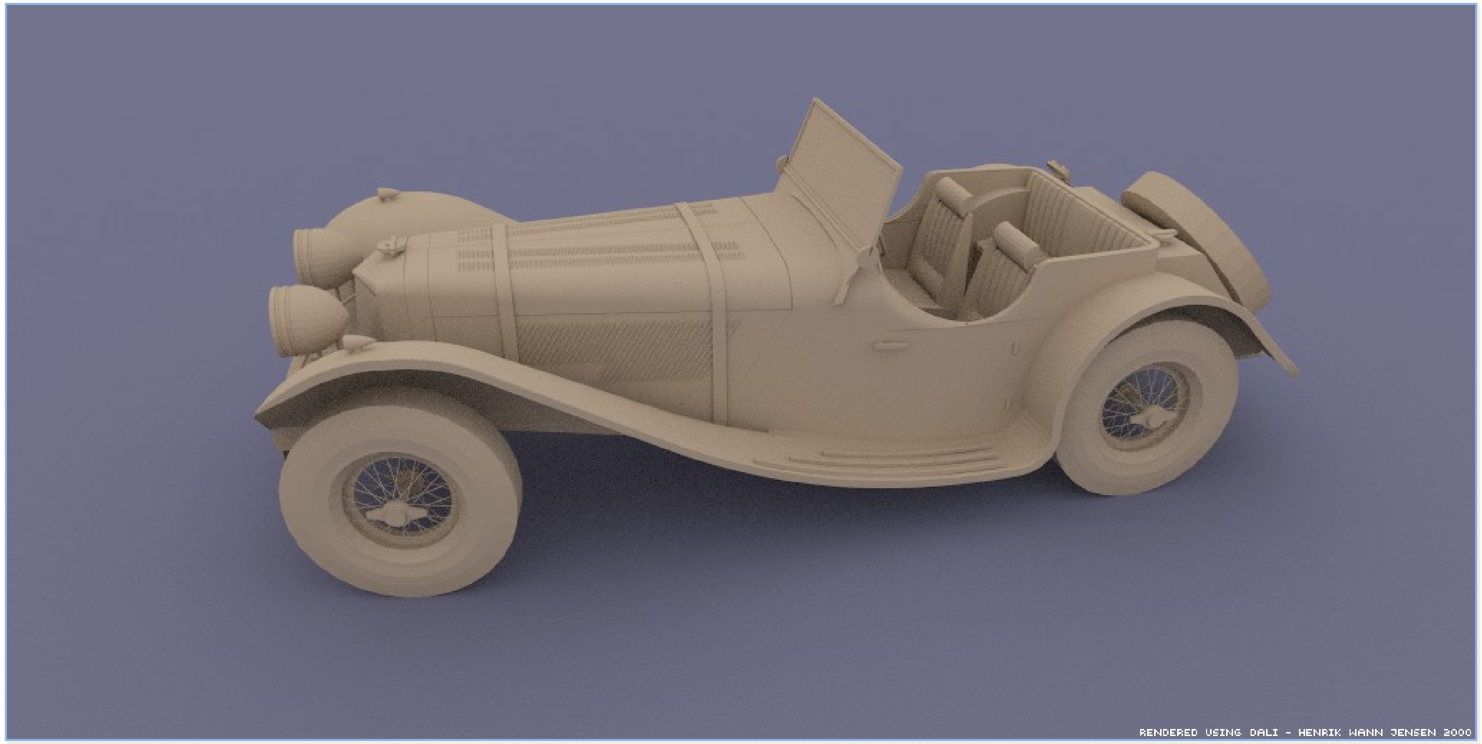
- Rendering = integration
  - Antialiasing
  - Soft shadows
  - Indirect illumination
  - Caustics



Jensen

Challenge

- Rendering integrals are difficult to evaluate
  - Multiple dimensions
  - Discontinuities
    - Partial occluders
    - Highlights
    - Caustics
  - Significant energy carried by "rare" paths

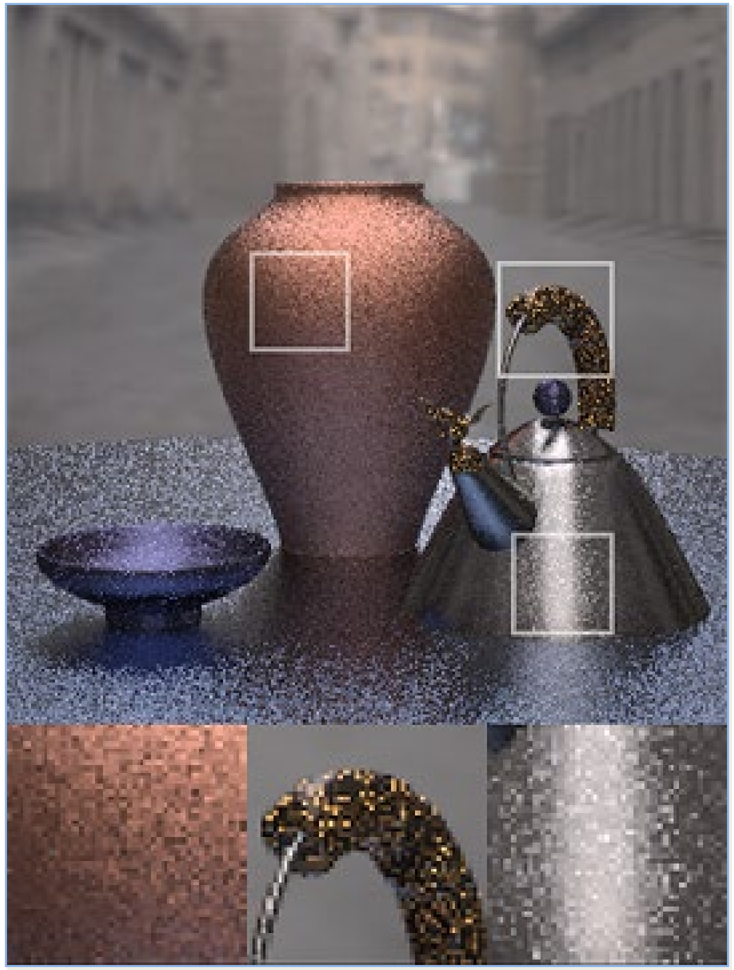


Challenge

- Rendering integrals are difficult to evaluate
  - Multiple dimensions
  - Discontinuities
    - Partial occluders
    - Highlights
    - Caustics
  - Significant energy carried by “rare” paths

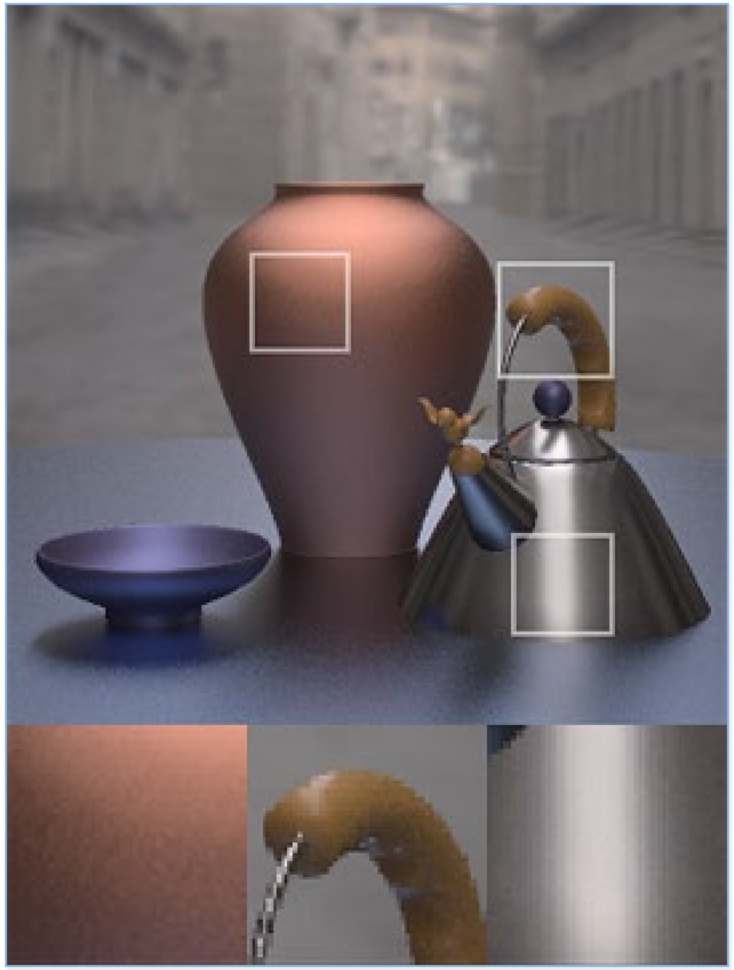
Monte Carlo Path Tracing

- Drawback: can be noisy unless *lots* of paths simulated
- 40 paths per pixel:

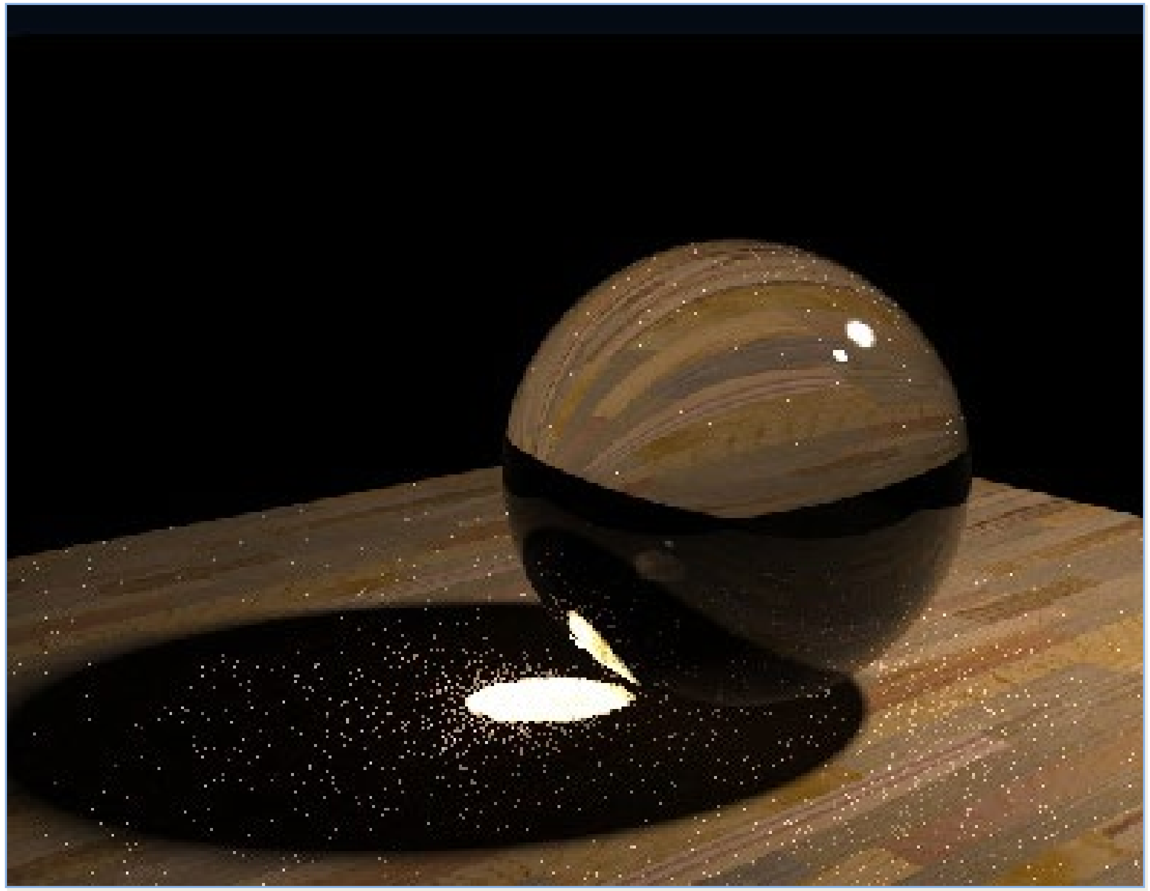


Monte Carlo Path Tracing

- Drawback: can be noisy unless *lots* of paths simulated
- 1200 paths per pixel:



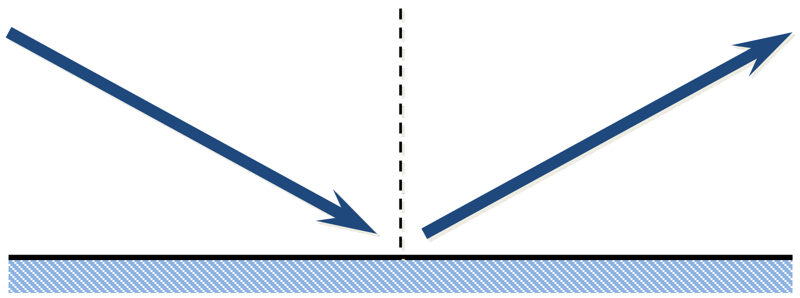
Monte Carlo Path Tracing



1000 paths/pixel

Reducing Variance

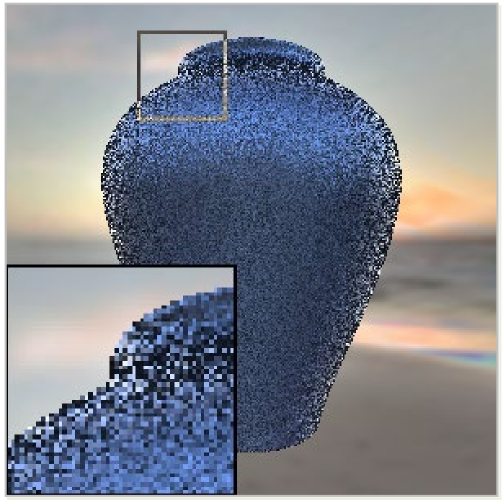
- Observation: some paths more important (carry more energy) than others
  - For example, shiny surfaces reflect most light in the ideal "mirror" direction

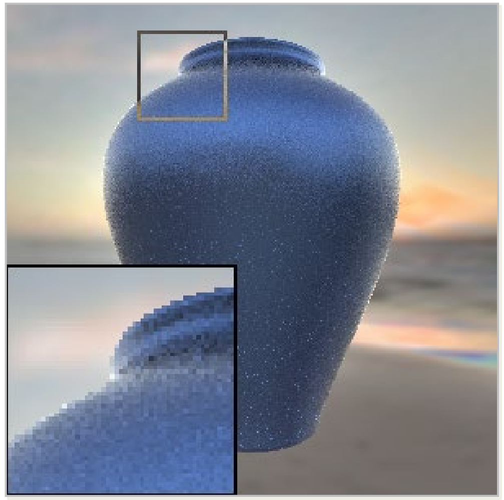


- Importance sampling! Put more paths where there is energy flow!

Effect of Importance Sampling

- Less noise at a given number of samples


Uniform random sampling


Importance sampling

- Equivalently, need to simulate fewer paths for some desired limit of noise