

PROBABILISTIC INFERENCE AND LEARNING

LECTURE 04

SAMPLING

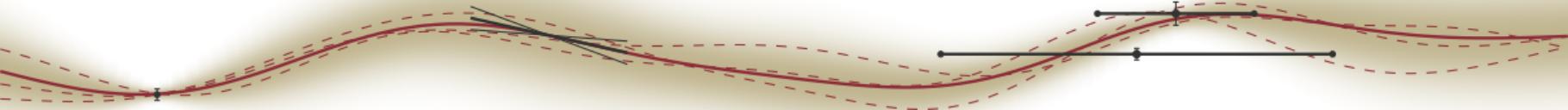
Philipp Hennig

27 April 2020

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN



FACULTY OF SCIENCE
DEPARTMENT OF COMPUTER SCIENCE
CHAIR FOR THE METHODS OF MACHINE LEARNING





#	date	content	Ex	#	date	content	Ex
1	20.04.	Introduction	1	14	09.06.	Logistic Regression	8
2	21.04.	Reasoning under Uncertainty		15	15.06.	Exponential Families	
3	27.04.	Continuous Variables	2	16	16.06.	Graphical Models	9
4	28.04.	Monte Carlo		17	22.06.	Factor Graphs	
5	04.05.	Markov Chain Monte Carlo	3	18	23.06.	The Sum-Product Algorithm	10
6	05.05.	Gaussian Distributions		19	29.06.	Example: Topic Models	
7	11.05.	Parametric Regression	4	20	30.06.	Mixture Models	11
8	12.05.	Understanding Deep Learning		21	06.07.	EM	
9	18.05.	Gaussian Processes	5	22	07.07.	Variational Inference	12
10	19.05.	An Example for GP Regression		23	13.07.	Example: Topic Models	
11	25.05.	Understanding Kernels	6	24	14.07.	Example: Inferring Topics	13
12	26.05.	Gauss-Markov Models		25	20.07.	Example: Kernel Topic Models	
13	08.06.	GP Classification	7	26	21.07.	Revision	



To Computation

Integration is the core computation of probabilistic inference

Probabilistic inference requires **integrals**:

- ▶ Evidences. Example from Lecture 3:

$$p(\pi | x_1, \dots, x_N) = \frac{\prod_i^N p(x_i | \pi)p(\pi)}{\int_0^1 \prod_i^N p(x_i | \pi)p(\pi) d\pi} = \frac{\int \prod_i^N \pi^n(1 - \pi)^{N-n}}{\int_0^1 \prod_i^N \pi^n(1 - \pi)^{N-n} d\pi}$$

- ▶ Expectations (actually, evidences are expectations, too)

$$\langle f \rangle_p := \mathbb{E}_p[f] := \int \underline{f(x)p(x)} dx$$

$$f(x) = x$$

$$f(x) = (x - \mathbb{E}_p(x))^2$$

$$f(x) = x^p$$

$$f(x) = -\log x$$

⋮

"Expectation of f under p "

mean

variance

p -th moment

entropy



The Toolbox

Framework:

$$\int p(x_1, x_2) dx_2 = p(x_1) \quad p(x_1, x_2) = p(x_1 | x_2)p(x_2) \quad p(x | y) = \frac{p(y | x)p(x)}{p(y)}$$

Modelling:

- ▶ Directed Graphical Models
- ▶
- ▶
- ▶
- ▶
- ▶
- ▶
- ▶

Computation:

- ▶ Monte Carlo
- ▶
- ▶
- ▶
- ▶
- ▶



Randomized Methods – Monte Carlo

the idea

- ▶ the “simplest thing to do”: replace integral with sum:

$$\int f(x)p(x) dx \approx \frac{1}{S} \sum_{i=1}^S f(x_i); \quad \int p(x, y) dx \approx \sum_i p(y | x_i); \quad \text{if } x_i \sim p(x)$$

- ▶ this requires being able to **sample** $x_i \sim p(x)$

Definition (Monte Carlo method)

*Algorithms that compute expectations in the above way, using samples $x_i \sim p(x)$ are called **Monte Carlo** methods (Stanisław Ulam, John von Neumann).*



image source: wikipedia u.fruitpunchline

A method from a different age

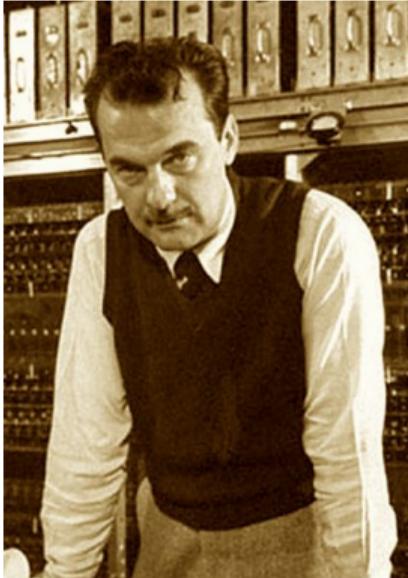
Monte Carlo Methods and the Manhattan Project



images: Los Alamos National Laboratory / wikipedia



Stanisław Ulam
1909–1984



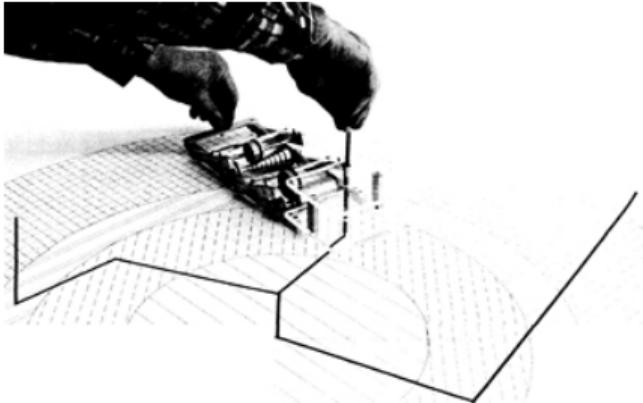
Nicholas Metropolis
1915–1999



John von Neumann
1903–1957

The FERMIAC

analog Monte Carlo computer



Example

a dumb way to compute π

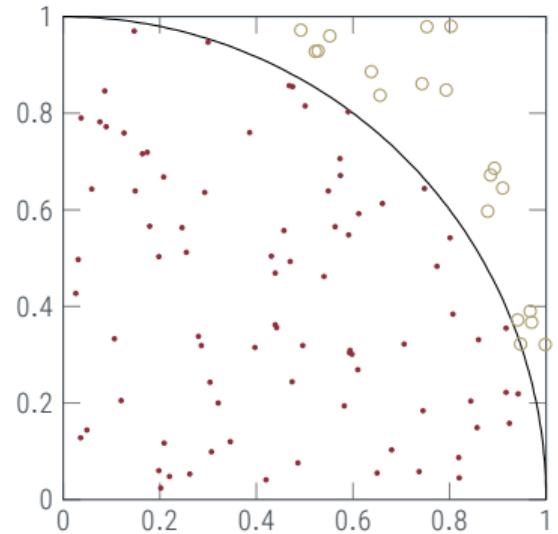
- ▶ ratio of quarter-circle to square: $\frac{\pi/4}{1}$
- ▶ $\pi = 4 \int \mathbb{I}(x^T x < 1) u(x) dx$
- ▶ draw $x \sim u(x)$, check $x^T x < 1$, count

```

1 from numpy.random import rand
2 S = 100000
3 sum((rand(S,2)**2).sum(axis=1) < 1) / S * 4

```

> 3.13708
> 3.14276



Monte Carlo works on **every** Integrable Function

is this a good thing?

$$\phi := \int f(x)p(x) dx = \mathbb{E}_p(f)$$

- ▶ Let $x_s \sim p, s = 1, \dots, S$ iid. (i.e. $p(x_s = x) = p(x)$ and $p(x_s, x_t) = p(x_s)p(x_t) \forall s, t$)

$$\hat{\phi} := \frac{1}{S} \sum_{s=1}^S f(x_s) \quad \leftarrow \text{the Monte Carlo estimator is ...}$$

$$\mathbb{E}(\hat{\phi}) =: \int \frac{1}{S} \sum_{s=1}^S f(x_s)p(x_s) dx_s = \frac{1}{S} \sum_{s=1}^S \int f(x_s)p(x_s) dx_s$$

$$= \frac{1}{S} \sum_{s=1}^S \mathbb{E}(f(x_s)) = \phi \quad \leftarrow \dots \text{an unbiased estimator!}$$

- ▶ the only requirement for this is that $\int f(x)p(x) dx$ exists (i.e. f must be Lebesgue-integrable relative to p). Monte Carlo integration can even work on discontinuous functions.

Sampling converges slowly

expected square error

- The expected square error (variance) drops as $\mathcal{O}(S^{-1})$

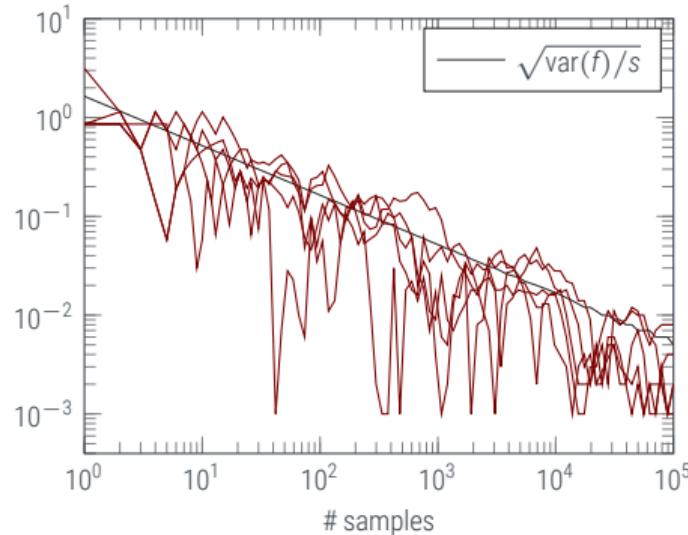
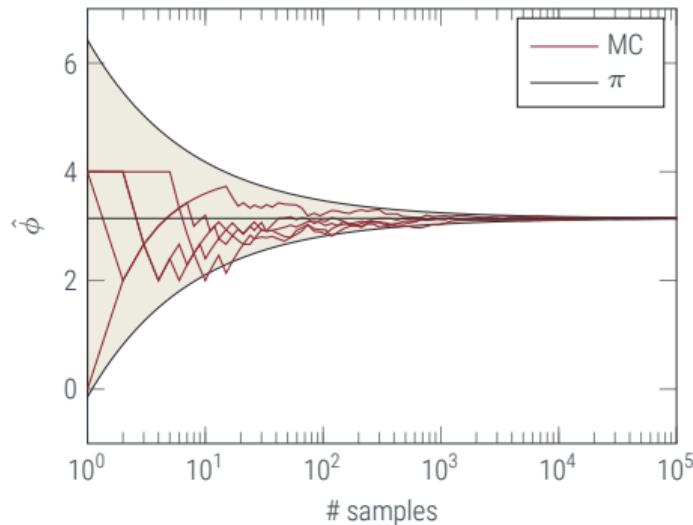
$$\begin{aligned}
 \mathbb{E}(\hat{\phi} - \mathbb{E}(\hat{\phi}))^2 &= \mathbb{E} \left[\frac{1}{S} \sum_{s=1}^S (f(x_s) - \phi) \right]^2 \\
 &= \frac{1}{S^2} \sum_{s=1}^S \sum_{r=1}^S \mathbb{E}(f(x_s)f(x_r)) - \phi \mathbb{E}(f(x_s)) - \mathbb{E}(f(x_r))\phi + \phi^2 \\
 &= \frac{1}{S^2} \sum_{s=1}^S \left(\left(\underbrace{\sum_{r \neq s} \phi^2 - 2\phi^2 + \phi^2}_{=0} \right) + \underbrace{\mathbb{E}(f^2) - \phi^2}_{=: \text{var}(f)} \right) \\
 &= \frac{1}{S} \text{var}(f) = \mathcal{O}(S^{-1})
 \end{aligned}$$

- Thus, the expected error (the square-root of the expected square error) drops as $\mathcal{O}(S^{-1/2})$



sampling is for rough guesses

recall example computation for π



- ▶ need only ~ 9 samples to get *order of magnitude* right ($\text{std}(\phi)/3$)
- ▶ need 10^{14} samples for single-precision ($\sim 10^{-7}$) calculations!
- ▶ sampling is good for **rough estimates**, not for precise calculations
- ▶ Always think of other options before trying to sample!



- ▶ **samples** from a probability distribution can be used to **estimate** expectations, **roughly**, without having to design an elaborate integration algorithm
- ▶ The error of the estimate is **independent** of the dimensionality of the input domain!

How do we generate random samples from $p(x)$?



Reminder: Change of Measure

The transformation law

Theorem (Change of Variable for Probability Density Functions)

Let X be a continuous random variable with PDF $p_X(x)$ over $c_1 < x < c_2$. And, let $Y = u(X)$ be a monotonic differentiable function with inverse $X = v(Y)$. Then the PDF of Y is

$$p_Y(y) = p_X(v(y)) \cdot \left| \frac{dv(y)}{dy} \right| = p_X(v(y)) \cdot \left| \frac{du(x)}{dx} \right|^{-1}.$$

Let $X = (X_1, \dots, X_d)$ have a joint density p_X . Let $g : \mathbb{R}^d \rightarrow \mathbb{R}^d$ be continuously differentiable and injective, with non-vanishing Jacobian J_g . Then $Y = g(X)$ has density

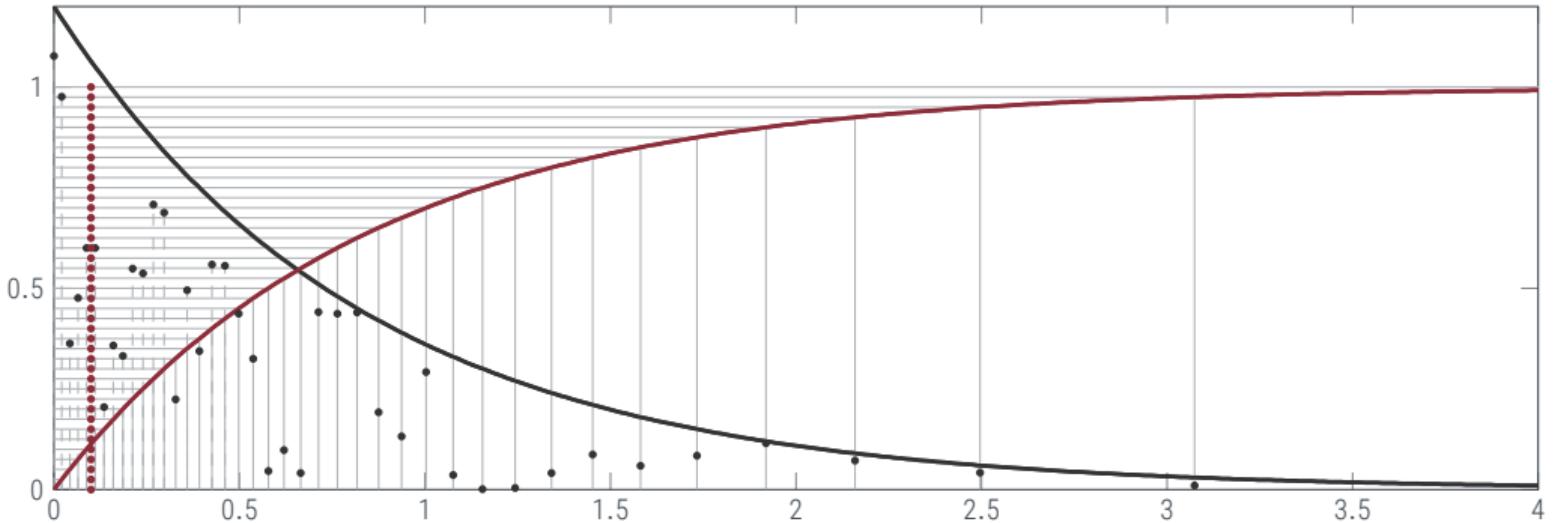
$$p_Y(y) = \begin{cases} p_X(g^{-1}(y)) \cdot |J_{g^{-1}}(y)| & \text{if } y \text{ is in the range of } g, \\ 0 & \text{otherwise.} \end{cases}$$

The Jacobian J_g is the $d \times d$ matrix with $[J_g(x)]_{ij} = \frac{\partial g_i(x)}{\partial x_j}$.



Some special cases

sampling from an exponential distribution is analytic



$$p(x) = \frac{1}{\lambda} e^{-x/\lambda}$$

$$1 - u = 1 - e^{-x/\lambda}$$

$$\int p(x) dx = 1 - e^{-x/\lambda}$$

$$x = -\lambda \log(u)$$

Example: Sampling from a Beta Distribution



uniform variables

Consider $u \sim U[0, 1]$ (i.e. $u \in [0, 1]$, and $p(u) = 1$). The variable $x = u^{1/\alpha}$ has the Beta density

$$p_x(x) = p_u(u(x)) \cdot \left| \frac{\partial u(x)}{\partial x} \right| = \frac{1}{\alpha} x^{\alpha-1} = \mathcal{B}(x; \alpha, 1).$$

Example: Sampling from a Beta Distribution



uniform variables

Consider $u \sim U[0, 1]$ (i.e. $u \in [0, 1]$, and $p(u) = 1$). The variable $x = u^{1/\alpha}$ has the Beta density

$$p_x(x) = p_u(u(x)) \cdot \left| \frac{\partial u(x)}{\partial x} \right| = \frac{1}{\alpha} x^{\alpha-1} = \mathcal{B}(x; \alpha, 1).$$

Homework:

Consider two *independent* variables

$$X \sim \mathcal{G}(\alpha, \theta) \quad Y \sim \mathcal{G}(\beta, \theta)$$

where $\Gamma(\xi; \alpha, \theta) = \frac{1}{\Gamma(\alpha)\theta^\alpha} \xi^{\alpha-1} e^{-\xi/\theta}$ is the *Gamma distribution*. Show that the random variable $Z = \frac{X}{X+Y}$ is Beta distributed, with the density

$$p(Z = z) = \mathcal{B}(z; \alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} z^{\alpha-1} (1-z)^{\beta-1}.$$



- ▶ samples from a probability distribution can be used to **estimate** expectations, **roughly**
- ▶ ‘random numbers’ don’t really need to be **unpredictable**, as long as they have as little **structure** as possible
- ▶ **uniformly distributed random numbers** can be **transformed** into other distributions. This can be done numerically efficiently in some cases, and it is worth thinking about doing so

What do we do if we don’t know a good transformation?



Why is sampling hard?

Sampling is harder than global optimization

To produce exact samples:

- ▶ need to know cumulative density everywhere
- ▶ need to know regions of high density (not just local maxima!)
- ▶ a global description of the entire function

Practical Monte Carlo Methods aim to construct samples from

$$p(x) = \frac{\tilde{p}(x)}{Z}$$

assuming that it is possible to *evaluate* the *unnormalized* density \tilde{p} (but not p) at arbitrary points.

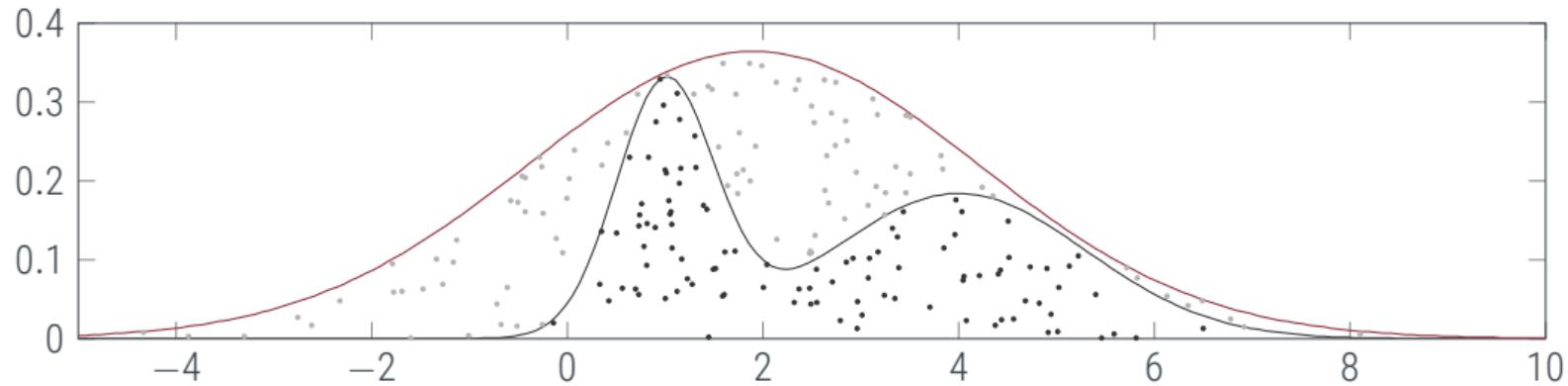
Typical example: Compute moments of a posterior

$$p(x | D) = \frac{p(D | x)p(x)}{\int p(D, x) dx} \quad \text{as} \quad \mathbb{E}_{p(x|D)}(x^n) \approx \frac{1}{S} \sum_s x_i^n \quad \text{with } x_i \sim p(x | D)$$



Rejection Sampling

a simple method [Georges-Louis Leclerc, Comte de Buffon, 1707–1788]

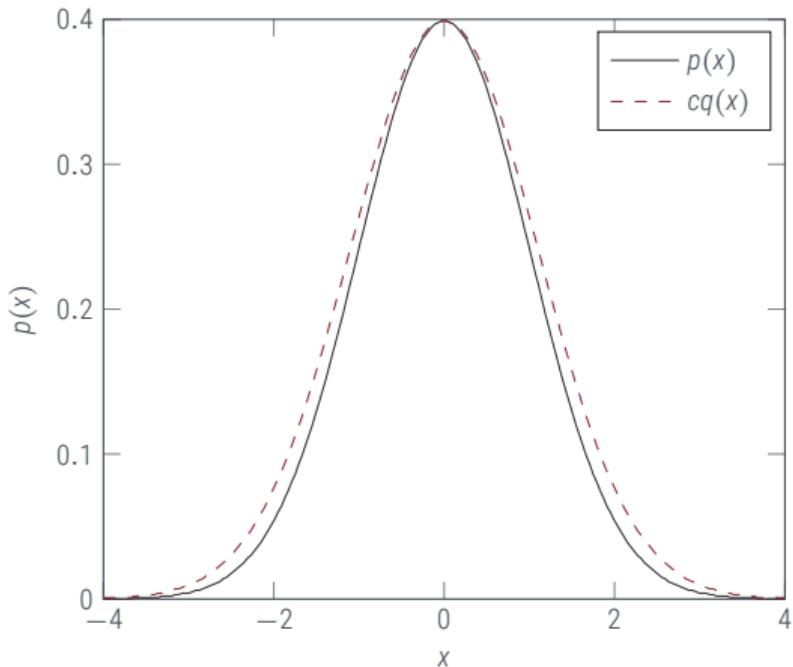


- ▶ for any $p(x) = \tilde{p}(x)/Z$ (normalizer Z not required)
- ▶ choose $q(x)$ s.t. $cq(x) \geq \tilde{p}(x)$
- ▶ draw $s \sim q(x)$, $u \sim \text{Uniform}[0, cq(s)]$
- ▶ **reject** if $u > \tilde{p}(s)$



The Problem with Rejection Sampling

the curse of dimensionality [MacKay, §29.3]



Example:

- ▶ $p(x) = \mathcal{N}(x; 0, \sigma_p^2)$
- ▶ $q(x) = \mathcal{N}(x; 0, \sigma_q^2)$
- ▶ $\sigma_q > \sigma_p$
- ▶ optimal c is given by

$$c = \frac{(2\pi\sigma_q^2)^{D/2}}{(2\pi\sigma_p^2)^{D/2}} = \left(\frac{\sigma_q}{\sigma_p}\right)^D = \exp\left(D \ln \frac{\sigma_q}{\sigma_p}\right)$$

- ▶ acceptance rate is ratio of volumes: $1/c$
- ▶ rejection rate rises **exponentially** in D
- ▶ for $\sigma_q/\sigma_p = 1.1, D = 100, 1/c < 10^{-4}$

Importance Sampling

a slightly less simple method

- ▶ computing $\tilde{p}(x)$, $q(x)$, then **throwing them away** seems **wasteful**
- ▶ instead, rewrite (assume $q(x) > 0$ if $p(x) > 0$)

$$\begin{aligned}\phi &= \int f(x)p(x) dx = \int f(x) \frac{p(x)}{q(x)} q(x) dx \\ &\approx \frac{1}{S} \sum_s f(x_s) \frac{p(x_s)}{q(x_s)} =: \frac{1}{S} \sum_s f(x_s) w_s \quad \text{if } x_s \sim q(x)\end{aligned}$$

- ▶ this is just using a new function $g(x) = f(x)p(x)/q(x)$, so it is an **unbiased estimator**
- ▶ w_s is known as the **importance (weight)** of sample s
- ▶ if normalization unknown, can also use $\tilde{p}(x) = Zp(x)$

$$\begin{aligned}\int f(x)p(x) &= \frac{1}{Z} \frac{1}{S} \sum_s f(x_s) \frac{\tilde{p}(x_s)}{q(x_s)} dx \\ &= \frac{1}{S} \sum_s f(x_s) \frac{\tilde{p}(x_s)/q(x_s)}{\frac{1}{S} \sum_{s'} \tilde{p}(x_{s'})/q(x_{s'})} =: \sum_s f(x_s) \tilde{w}_s\end{aligned}$$

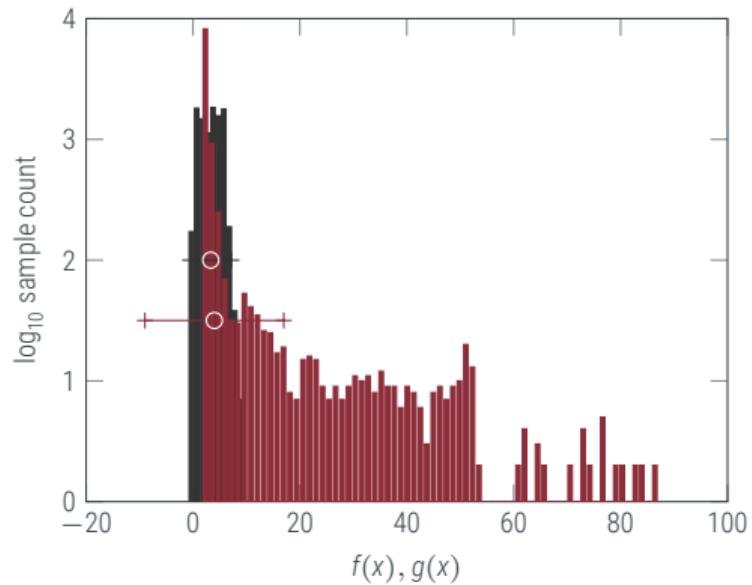
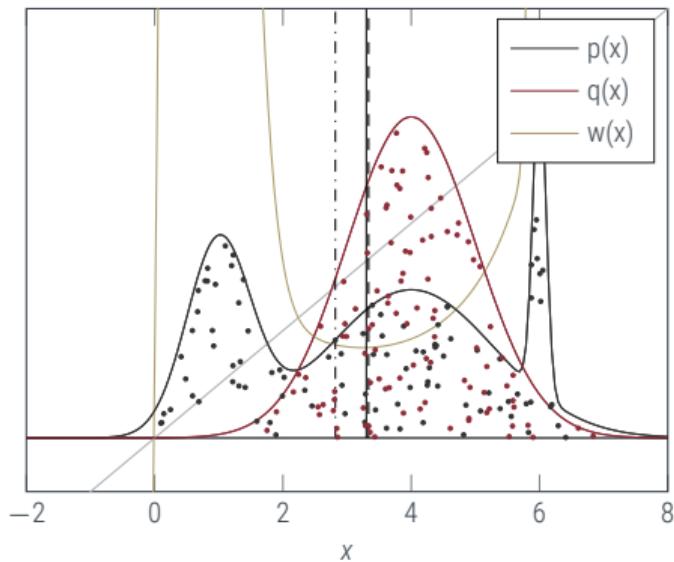
- ▶ this is **consistent**, but **biased**



What's wrong with Importance Sampling?

the curse of dimensionality, revisited

- ▶ recall that $\text{var } \hat{\phi} = \text{var}(f)/S$ – importance sampling replaces $\text{var}(f)$ with $\text{var}(g) = \text{var}\left(\frac{f_p}{q}\right)$
- ▶ $\text{var}\left(\frac{f_p}{q}\right)$ can be very large if $q \ll p$ somewhere. In many dimensions, usually all but everywhere!
- ▶ if p has “undiscovered islands”, some samples have $p(x)/q(x) \rightarrow \infty$





Sampling (Monte Carlo) Methods

Sampling is a way of performing rough probabilistic computations, in particular for **expectations** (including **marginalization**).

- ▶ samples from a probability distribution can be used to **estimate** expectations, **roughly**
- ▶ uniformly distributed random numbers can be **transformed** into other distributions. This can be done numerically efficiently in some cases, and it is worth thinking about doing so
- ▶ Rejection sampling is a primitive but **exact** method that works with **intractable** models
- ▶ Importance sampling makes more efficient use of samples, but can have high variance (and this may not be obvious)

Next Lecture:

- ▶ **Markov Chain Monte Carlo** methods are more elaborate ways of getting **approximate** answers to intractable problems.