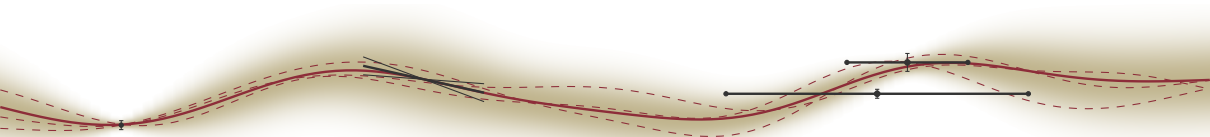
Probabilistic Inference and Learning
Lecture 20
Monte Carlo Methods

Philipp Hennig

09 January 2019

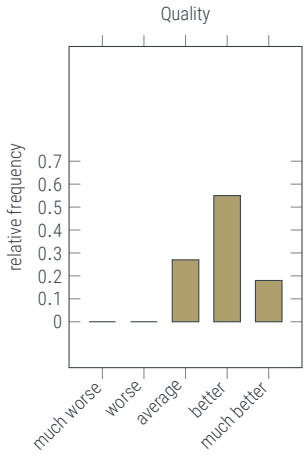
Eberhard Karls
Universität Tübingen

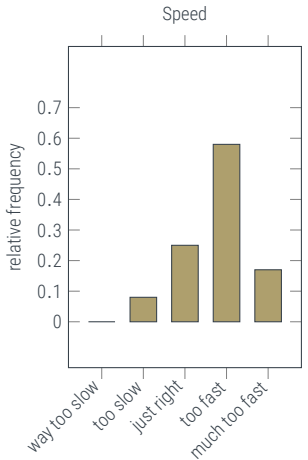
Faculty of Science
Department of Computer Science
Chair for the Methods of Machine Learning

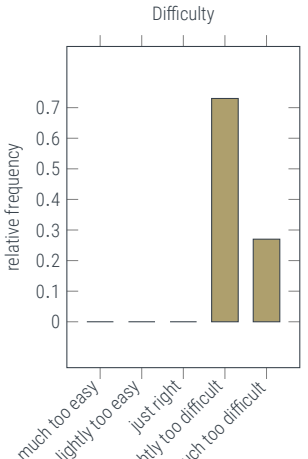


Last Lecture: Debrief

Feedback dashboard







Last Lecture: Debrief

Detailed Feedback

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Things you did not like:

- Example (flipping between slides too much, too fast)
- re-naming variables ($N_k = R_k$)
- second half was too fast

- What does $\mathbb{E}_q(y)$ mean?
- Why is it called 'responsibilities?'

Things you did not understand:

- The example
- Is this type of variational inference called Gibbs sampling?
- It feels like you need a lot of experience to successfully do VI. I'm still not confident I could apply variational inference in practice.
- What was the main takeaway? Should I be able to do the math, understand the concept, or just accept it?

Things you enjoyed:

- **the toolbox** !
- Physics-connection
- the Example
- summary of definitions
- the exercise sheet

The Exam

sign up for it!

- reminder: Exam will take place on

  Wednesday, 13 February 2019, 8-10am (s.t.!), in HS25, Kupferbau

- you have to sign up **twice**!
  - in *Campus* — or, if that's impossible, by mail to your **Prüfungssekretariat** (that's the official sign-up)
  - in Ilias (this is for internal preparation)

Ein Hinweis in Eigener Sache …

http://sciencenotes.de

- 17.01.2019, Schlachthaus Tübingen
- Einlass: 19:30 Uhr, Beginn: 20:00 Uhr

1. Wie sehen Maschinen unsere Welt?, Dr. Wieland Brendel
2. Wie KI weiß, dass sie nicht(s) weiß, Prof. Dr. Matthias Hein
3. Präzise Unsicherheit – Rechenalgorithmen für lernende Maschinen, Prof. Dr. Philipp Hennig
4. Understanding Self-Organization of the Brain, Dr. Anna Levina
5. Open Mic: Künstliche Intelligenz und Wir

LIVE-SET: STRÖME

0. Introduction to Reasoning under Uncertainty
1. Probabilistic Reasoning
2. Probabilities over Continuous Variables
3. Gaussian Probability Distributions
4. Gaussian Parametric Regression
5. More on Parametric Regression
6. Gaussian Processes
7. More on Kernels & GPs
8. A practical GP example
9. Markov Chains, Time Series, Filtering
10. Classification
11. Empirical Example of Classification
12. Bayesianism and Frequentism
13. Stochastic Differential Equations
14. Exponential Families
15. Graphical Models
16. Factor Graphs
17. The Sum-Product Algorithm
18. Mixture Models
19. The EM Algorithm
20. Variational Inference
21. Monte Carlo
22. Markov Chain Monte Carlo
23. Dimensionality Reduction
24. Advanced Modelling Example I
25. Advanced Modelling Example II
26. Advanced Modelling Example III
27. Some Wild Stuff
28. Revision

Probabilistic inference requires expectations, marginals

$$\langle f \rangle_p = \mathbb{E}_p[f] = \int f(x) p(x) \, \mathrm{d}x; \qquad p(y) = \int p(y \mid x) p(x) \, \mathrm{d}x$$

Example: Marginalizing over hyperparameters, e.g. remember Gaussian process regression

$$\begin{aligned}
p(f \mid \theta) &= \mathcal{GP}(f; \mu_\theta, k_\theta) \qquad p(y \mid f) = \mathcal{N}(y; f_X, \sigma^2 I) \\
p(f_x \mid y, \theta) &= \mathcal{N}(f_x; \bar{\mu}_\theta(x), \bar{k}_\theta(x, x)) \\
p(f_x \mid y) &= \int \mathcal{N}(f_x \mid X, y, \theta) p(\theta) \, \mathrm{d}\theta = ? \\
&\approx \mathcal{N}(f_x; \hat{\mu}(x), \hat{k}(x, x)) \quad \text{where} \\
\hat{\mu}(x) &:= \int f_x \cdot \mathcal{N}(f_x \mid X, y, \theta) p(\theta) \, \mathrm{d}\theta \\
\hat{k}(x, x) &:= \int (f_x - \hat{\mu})(f_x - \hat{\mu})^\intercal \cdot \mathcal{N}(f_x \mid X, y, \theta) p(\theta) \, \mathrm{d}\theta
\end{aligned}$$

The Toolbox

Five principal methods for dealing with computational complexity in probabilistic inference

1. **Maximum Likelihood (ML) / Maximum A-Posteriori (MAP)** estimation:
   To estimate θ in $p(D \mid \theta)$ or $p(\theta \mid D)$, set $\hat{\theta} = \arg\max_\theta p$.
2. **Laplace** Approximation: $p(\theta \mid D) \approx \mathcal{N}\left(\theta; \hat{\theta}, -(\nabla\nabla^\intercal \log p(\theta \mid D))^{-1}\right)$
3. **Variational Inference:**
   To approximate $p(\theta \mid D)$, impose structure on $q(\theta)$, then minimize $D_{\mathrm{KL}}(q \| p)$
4. **Monte Carlo:** $\int f(x)p(x)\,\mathrm{d}x \approx \sum_i f(x_i)$ where $x_i \sim p$
5. **Numerical Quadrature:**
   To marginalize θ, compute $\int p(f \mid \theta)\, d\theta \approx \sum_i w_i \cdot p(f \mid \theta_i)$

Disclaimer: The listed items are neither mutually exclusive nor collectively exhaustive. Some of the methods are intricately interrelated.

- the "simplest thing to do": replace integral with sum:

$$\int f(x)p(x)\,\mathrm{d}x \approx \sum_i f(x_i); \qquad \int p(x,y)\,\mathrm{d}x \approx \sum_i p(y \mid x_i); \qquad \text{if } x_i \sim p(x)$$

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

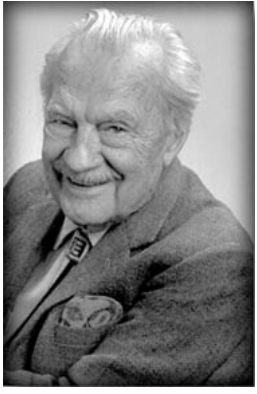
Definition (Monte Carlo method)

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).

A method from a different age

Monte Carlo Methods and the Manhattan Project

Stanisław Ulam
1909–1984

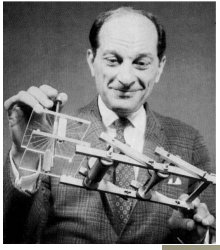


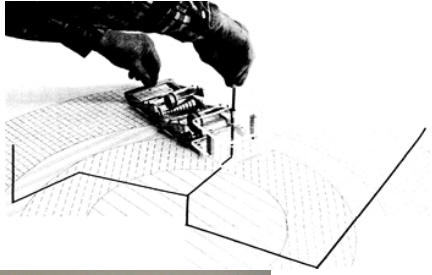
Nicholas Metropolis
1915–1999

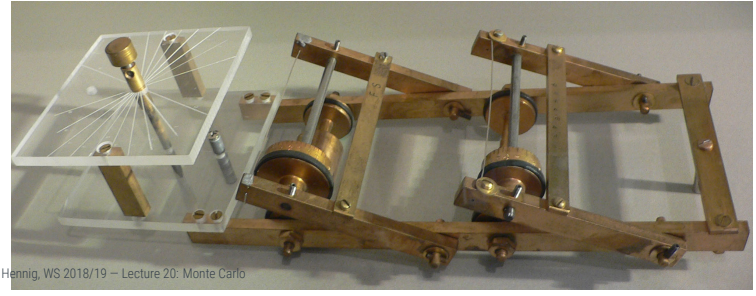
John von Neumann
1903–1957

The FERMIAC

analog Monte Carlo computer







Why Sampling is a Good Idea

sampling gives a **rough** estimate, **fast**

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

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

$$\hat{\phi} := \frac{1}{S}\sum_s f(x_s) \qquad \mathbb{E}(\hat{\phi}) = \frac{1}{S}\sum_s \mathbb{E}(f(x_s)) = \phi \qquad \leftarrow \text{an } \textbf{unbiased} \text{ estimator}$$

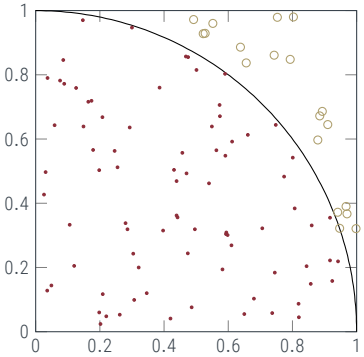
- Its variance (expected square error) drops over time with $\mathcal{O}(S^{-1/2})$

$$\begin{aligned}
\mathbb{E}(\hat{\phi} - \mathbb{E}(\hat{\phi}))^2 &= \mathbb{E}\left[\frac{1}{S}\sum_s (f(x_s) - \phi)\right]^2 \\
&= \frac{1}{S^2}\sum_s \sum_r \mathbb{E}(f(x_s)f(x_r)) - \phi\mathbb{E}(f(x_s)) - \mathbb{E}(f(x_r))\phi + \phi^2 \\
&= \frac{1}{S^2}\sum_s S\cdot(\phi^2 - 2\phi^2 + \phi^2) + \mathbb{E}(f^2) - \phi^2 \qquad = \frac{1}{S}\,\mathrm{var}(f)
\end{aligned}$$

sampling is for rough guesses

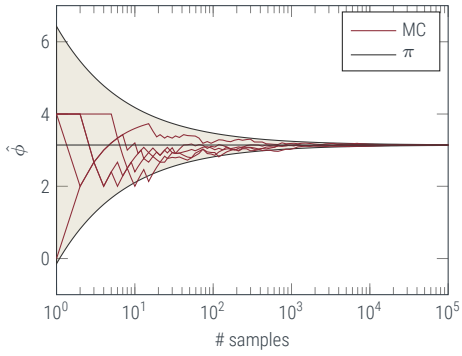
a dumb way to learn π

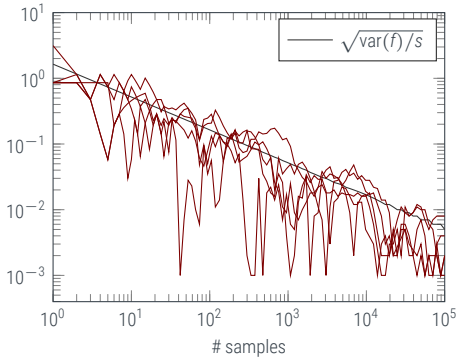
- ratio of quarter-circle to square: $\frac{\pi/4}{1}$
- $\pi = 4 \int \mathbb{I}(x^\intercal x < 1) u(x)\, \mathrm{d}x$
- draw $x \sim u(x)$, check $x^\intercal x < 1$, count
- `4 * sum(sum(rand(100,2).^2,2) < 1) / 100 = 3.2400`
- `4 * sum(sum(rand(1e6,2).^2,2) < 1) / 1e6 = 3.1422`



sampling is for rough guesses

a dumb way to learn π





- need only $\sim$ 9 samples to get *order of magnitude* right (std(ϕ)/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 $p(x)$?

First, though, a deeper question:

What is a **random sample?**

What is a Random Number?

What is a **sequence** or random numbers?

- 662244111144334455553366666666
- 16939937510582097494459230781 6
- 64247275977256550861548754357 4
- 71290426347261059020833604489 5
- 10001111110111111110010100000 1

What is a Random Number?

What is a **sequence** or random numbers?

- 662244111144334455553366666666
- 169399375105820974944592307816
- 642472759772565508615487543574
- 712904263472610590208336044895
- 10001111110111111100101000001

dice, doubled

What is a Random Number?

What is a **sequence** or random numbers?

- 662244111144334455553366666666 — dice, doubled
- 169399375105820974944592307816 — 41-70th digits of π
- 642472759772565508615487543574
- 712904263472610590208336044895
- 10001111110111111100101000001

What is a Random Number?

What is a **sequence** or random numbers?

- 662244111144334455553366666666 — dice, doubled
- 169399375105820974944592307816 — 41-70th digits of π
- 642472759772565508615487543574 — 1561-1590th digits of $1/2(1+\sqrt{5})$
- 712904263472610590208336044895
- 10001111110111111100101000001

What is a Random Number?

What is a **sequence** or random numbers?

662244111144334455553366666666	dice, doubled
169399375105820974944592307816	41-70th digits of π
642472759772565508615487543574	1561-1590th digits of $1/2(1+\sqrt{5})$
712904263472610590208336044895	von Neumann method, seed 908344
10001111110111111100101000001	

What is a Random Number?

What is a **sequence** or random numbers?

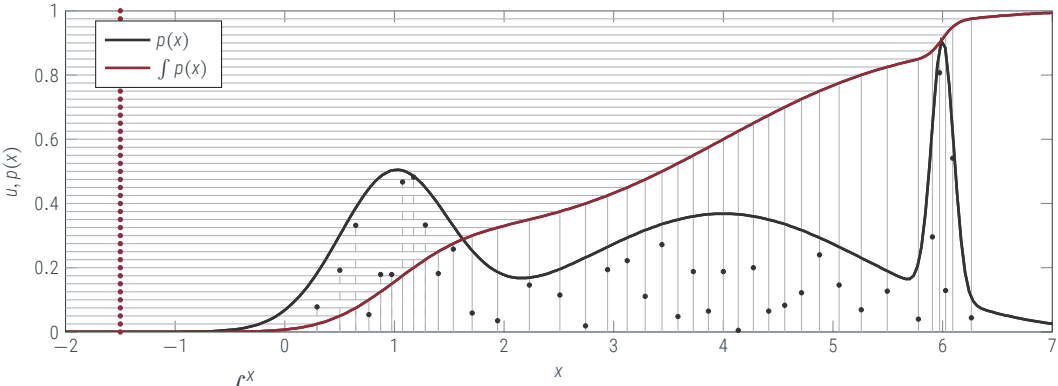
662244111144334455553366666666	dice, doubled
16939937510582097494459230781 6	41-70th digits of π
64247275977256550861548754357 4	1561-1590th digits of $1/2(1+\sqrt{5})$
71290426347261059020833604489 5	von Neumann method, seed 908344
10001111110111111110010100000 1	bits from G. Marsaglia's 'diehard CD'

- for use in Monte Carlo, the important property is **freedom from patterns** (because it implies anytime unbiasedness)
- in fact, use for MC only really requires the right density, disorder is just helpful for the argument
- for use in cryptography, the important property is **unpredictability**
- (aleatory) randomness is a philosophically dodgy concept
- (epistemic) **uncertainty** is a much clearer notion (see first lectures)

`http://www.stat.fsu.edu/pub/diehard/`

Turning uniform samples into other distributions

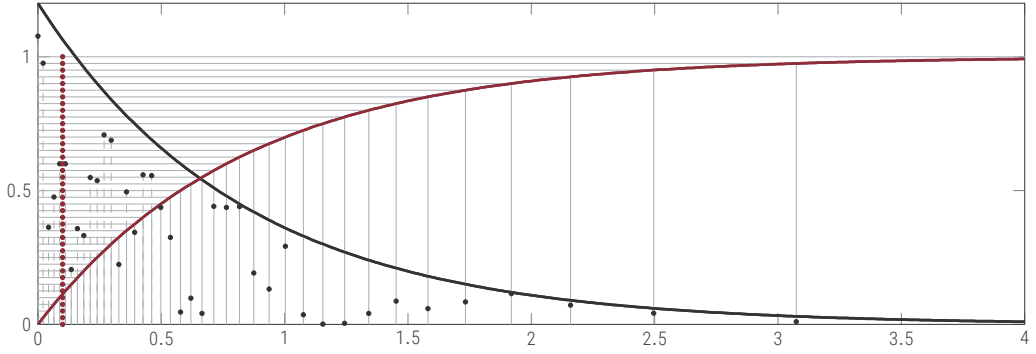
inverse transform sampling



- requires $P(x) = \int_{-\infty}^{x} p(\tilde{x})\, \mathrm{d}\tilde{x}$
- requires solving $x(u) = P^{-1}(u)$, for $u \sim \text{Uniform}[0, 1]$
- tricky for multivariate distributions

Some special cases

sampling from an exponential distribution is analytic



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

$$1 - u = 1 - e^{-x/\lambda} \qquad x = -\lambda \log(u)$$

Sampling from a (Standard) Gaussian

Recall Lecture 4 — The Box-Muller Transform

[Box & Muller, 1958; also Paley & Wiener, 1934]

Illustration: (u/Cmglee, CC BY-SA 3.0)

https://upload.wikimedia.org/wikipedia/commons/1/1f/Box-Muller_transform_visualisation.svg

```
1  procedure RANDN(n)
2      u_0 ⇐ RAND                          // draw first unit random
3      for i=1,...,n do
4          u_i ⇐ RAND                      // draw unit random
5          r ⇐ sqrt(-2 ln u_{i-1})         // compute radius
6          θ ⇐ 2π u_i                      // compute angle
7          if i%2 == 0 then
8              z_i ⇐ r cos θ               // transform to Euclidean
9          else
10             z_i ⇐ r sin θ               // transform to Euclidean
11         end if
12     end for
13     return z ~ N(0, I_n)
14 end procedure
```

Improvements

- Marsaglia's polar form: draw from unit circle instead of square (using rejection sampling). This avoids sin and cos, but requires rejection. Typically faster
- Marsaglia's Ziggurat algorithm: iteratively refined rejection sampling. Even faster, but tough to understand intuitively.

Specialised direct sampling methods exist

It is worth thinking about hard problems if they are important

For example

- Ziggurat Method: For monotonically decaying pdfs: Approximate PDF with an upper and lower bound step function, only spend computational effort on the 'difficult' cases in between (currently the best way to draw normal samples)

  G. Marsaglia; W.W. Tsang
  The Ziggurat Method for Generating Random Variables
  J of Statistical Software 5 (8), 2000

- Subgroup Algorithm: To generate uniformly random elements from a compact group, iteratively draw in an expanding chain of subgroups by drawing uniformly from the cosets (e.g. used to draw rotation, permutation matrices at random)

  P. Diaconis; M. Shahshahani
  The subgroup algorithm for generating uniform random variables
  Probability in the Engineering and Informational Sciences, 1987

- **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

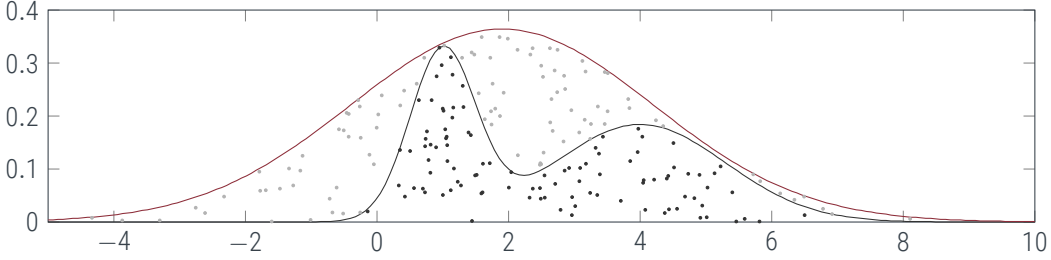
$$p(x) = \frac{\tilde{p}(x)}{Z}$$

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

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

Rejection Sampling

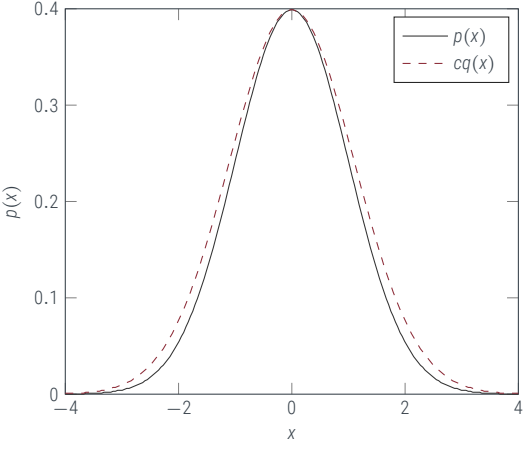
a simple method [Georges-Louis Leclerc, Comte de Buffon, 1707–1788]



- 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 $p(x) > 0$)

$$\phi = \int f(x)p(x)\,\mathrm{d}x = \int f(x)\frac{p(x)}{q(x)}q(x)\,\mathrm{d}x$$
$$\approx \frac{1}{S}\sum_s f(x_s)\frac{p(x_s)}{q(x_s)} =: \frac{1}{S}\sum_s f(x_s)w_s \qquad \text{if } x_s \sim q(x)$$

- 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 $\tilde{p}(x) = Zp(x)$

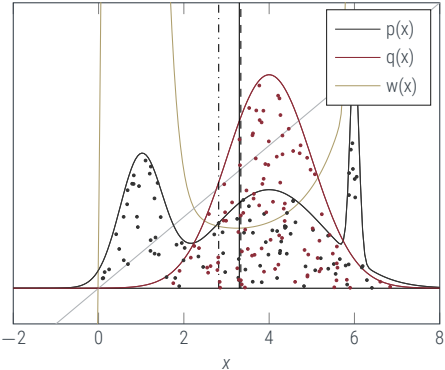
$$\int f(x)p(x) = \frac{1}{Z}\frac{1}{S}\sum_s f(x_s)\frac{\tilde{p}(x_s)}{q(x_s)}\,\mathrm{d}x$$
$$= \frac{1}{S}\sum_s f(x_s)\frac{\tilde{p}(x_s)/q(x_s)}{\frac{1}{S}\sum_{s'} 1\tilde{p}(x_s)/q(x_s)} =: \sum_s f(x_s)\tilde{w}_s$$

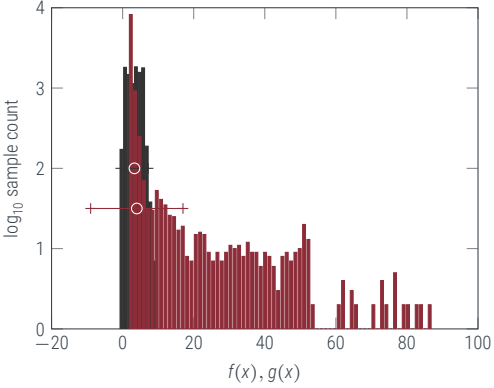
- this is consistent, but biased

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 $\text{var}(g) = \text{var}\left(f\frac{p}{q}\right)$
- $\text{var}\left(f\frac{p}{q}\right)$ can be very large if $q \ll p$ somewhere. In many dimensions, usually all but everywhere!
- if p has "undiscovered islands", some samples have $p(x)/q(x) \to \infty$





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
- 'random numbers' don't need to be **unpredictable**, but rather **unstructured**
- 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.