C24 Advanced Probability Theory

Michael A. Osborne

mosb@robots.ox.ac.uk
www.robots.ox.ac.uk/∼mosb/c24

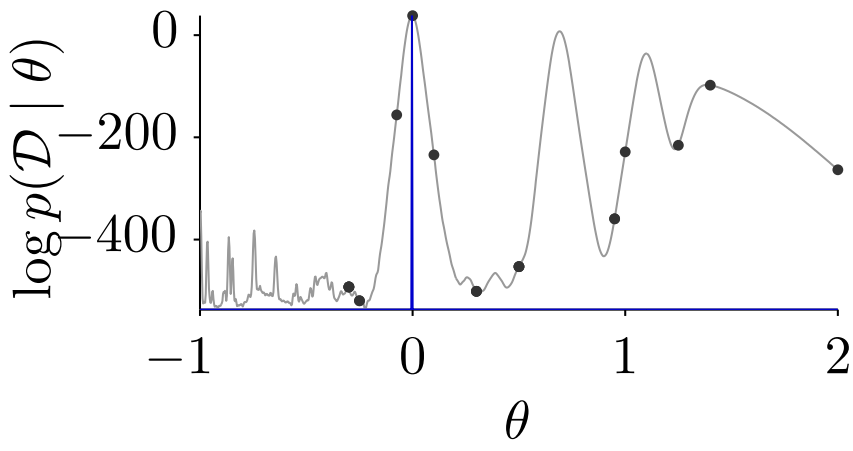
Michaelmas 2014

Topic 3: Approximate Inference

The core challenge of Bayesian inference is integration, marginalising over the unknown,

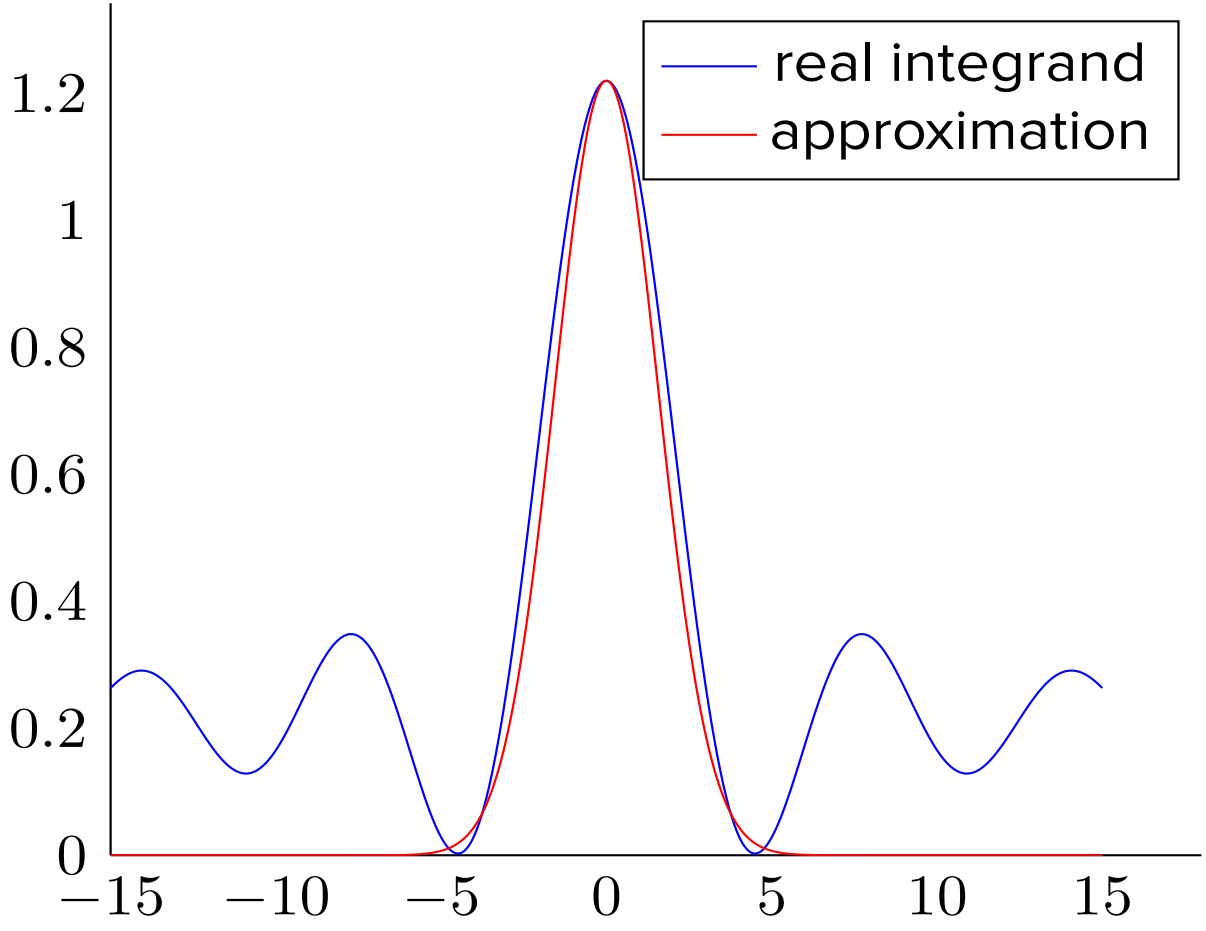
$$\text{e.g. } p(f_\star \mid \mathcal{D}) = \int p(f_\star \mid \mathcal{D}, \theta)\, p(\theta \mid \mathcal{D})\, \mathrm{d}\theta.$$

Recall that MLE and MAP approximate $p(\mathcal{D} \mid \theta)$ and $p(\theta \mid \mathcal{D})$, respectively, as delta functions to resolve intractable integrals. Better alternatives come up with more accurate models of those pdfs that nonetheless allow integration to be performed.



One way to improve upon MAP and MLE is the Laplace approximation.

This approach fits an un-normalised Gaussian around the maximum of an integrand.



Laplace's approximation takes a Taylor expansion for the logarithm of the integrand.

Let's tackle estimating (for e.g. $g(\theta) = p(f_\star \mid \mathcal{D}, \theta)p(\theta \mid \mathcal{D})$)

$$Z = \int g(\theta)\mathrm{d}\theta.$$

Let's expand $\log g(\theta)$ around a point $\hat{\theta}$, as

$$\log g(\theta) \simeq \log g(\hat{\theta}) + (\theta - \hat{\theta})\nabla \log g(\hat{\theta}) + \frac{1}{2}(\theta - \hat{\theta})^\top \left(\nabla\nabla^\top \log g(\hat{\theta})\right)(\theta - \hat{\theta}),$$

where $\nabla \log g(\hat{\theta}) = \nabla \log g(\theta)|_{\theta=\hat{\theta}}$ is the gradient evaluated at $\theta = \hat{\theta}$, and $\nabla\nabla^\top \log g(\hat{\theta}) = \nabla\nabla^\top \log g(\theta)\Big|_{\theta=\hat{\theta}}$ is the Hessian (matrix of all second derivatives) evaluated at $\theta = \hat{\theta}$.

Laplace's approximation takes a Taylor expansion for the logarithm of the integrand.

If we pick $\hat{\theta}$ as the θ that maximises $g(\theta)$, $\nabla \log g(\hat{\theta}) = 0$, hence

$$
\begin{aligned}
Z &= \int \exp \log g(\theta) \mathrm{d}\theta \\
&\simeq \int \exp\left(\log g(\hat{\theta}) + \frac{1}{2}(\theta - \hat{\theta})^\top \left(\nabla\nabla^\top \log g(\hat{\theta})\right)(\theta - \hat{\theta})\right)\mathrm{d}\theta \\
&= g(\hat{\theta})\sqrt{|2\pi\Sigma|} \int \frac{1}{\sqrt{|2\pi\Sigma|}} \exp\left(-\frac{1}{2}(\theta - \hat{\theta})^\top \Sigma^{-1}(\theta - \hat{\theta})\right)\mathrm{d}\theta \\
&= g(\hat{\theta})\sqrt{|2\pi\Sigma|}
\end{aligned}
$$

where $\Sigma = -\left(\nabla\nabla^\top \log g(\hat{\theta})\right)^{-1}$. Note we never actually have to compute this inverse, as the determinant of the inverse is one over the determinant.

Laplace's approximation requires:

1. Finding the $\hat{\theta}$ that maximises $g(\theta)$ (usually with numerical optimisation); and
2. Computing the Hessian of $\log g$ at that point.

We can then write

$$
\begin{aligned}
Z &= \int g(\theta) \mathrm{d}\theta \\
&\simeq g(\hat{\theta}) \frac{1}{\sqrt{\left| -\frac{1}{2\pi} \nabla\nabla^\top \log g(\hat{\theta}) \right|}}.
\end{aligned}
$$

Is the Laplace approximation is invariant to changes of representation of θ?

1. Yes.
2. No.

Is the Laplace approximation is invariant to changes of representation of θ?

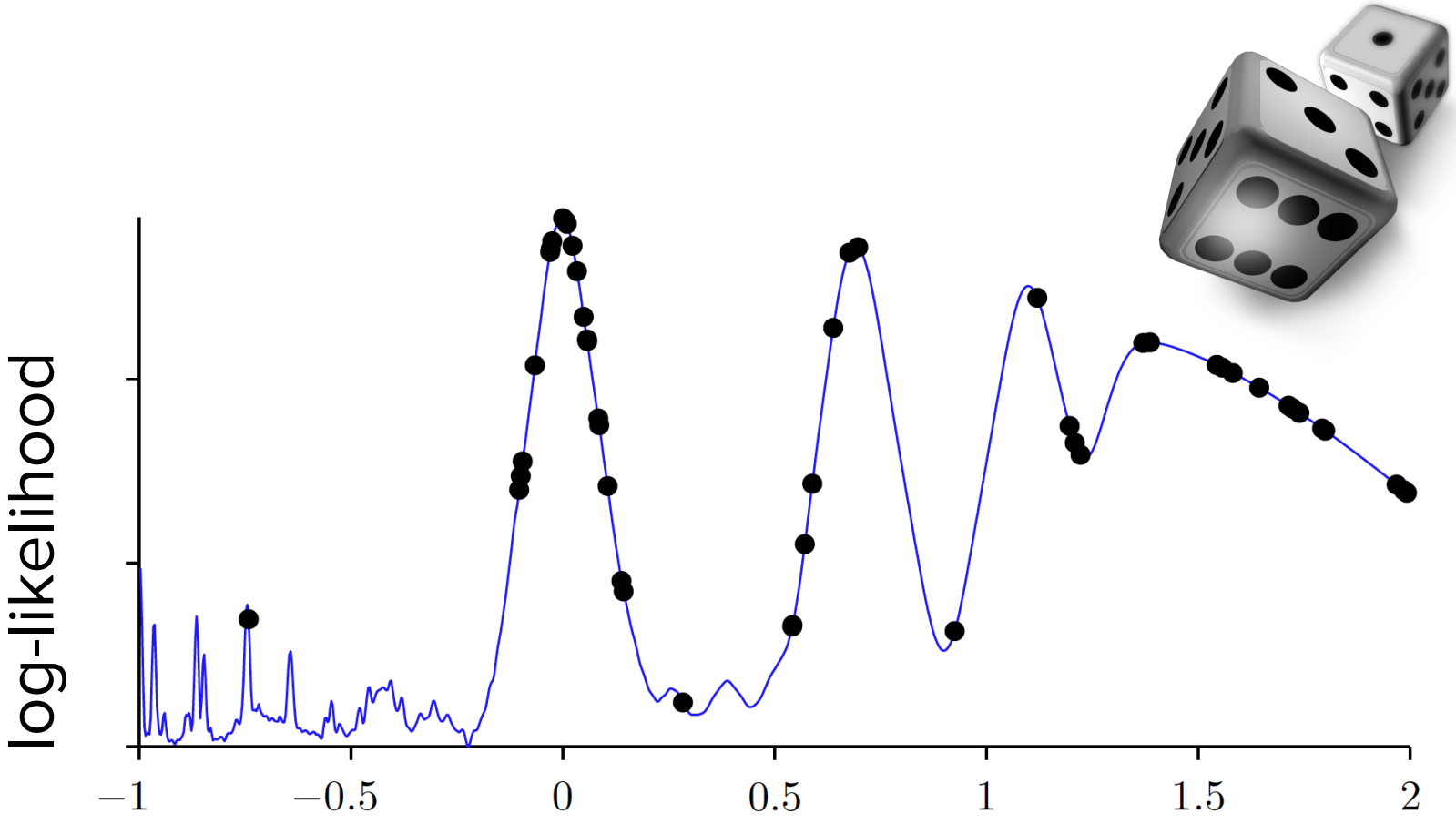
1. Yes.
2. No.

Laplace's method relies on optimising a density, and remember that the optimum is not invariant to changes of variables.

This means that some choices of representation are going to give better estimates for the integral than others.

Monte Carlo techniques have revolutionised Bayesian inference.

The core idea is to (pseudo-) randomly generate samples of an integrand, and then use them to estimate the integral.



Drawing samples, or simply sampling, from a distribution is a bit odd.

We say a random (uncertain) variable Y is drawn (or sampled) from a function $q(\cdot)$ if $p(y) = q(y)$.

Choosing y in such a way means that we are forcing ourselves to be epistemically uncertain about its value, according to $q(y)$.

I think it's strange that you would want to render yourself uncertain about the result of your decision.

Someone else (who knew the state of the pseudo-random number generator) might be able to say exactly what the value of the computation was going to be.

Monte Carlo methods aim to solve the core problem in Bayesian inference.

That is, it estimates the expected value, $\mathbb{E}[a]$, of some function $a(\theta)$ over some density $p(\theta)$.

We assume that we can evaluate $a(\theta)$ for any given θ; however, we cannot usually evaluate $p(\theta)$ directly.

Instead, we have access only to $f(\theta) = {}^{p(\theta)}/_{c}$ for an unknown (normalising) constant c.

Hence the problem is to find

$$\mathbb{E}[a] = \frac{\int a(\theta) f(\theta) \mathrm{d}\theta}{\int f(\theta) \mathrm{d}\theta}$$

given $f(\cdot)$ and $a(\cdot)$ (black box functions that we can evaluate at points of our choice).

Monte Carlo methods aim to solve the core problem in Bayesian inference.

The problem of finding

$$p(f_\star \mid \mathcal{D}) = \int p(f_\star \mid \mathcal{D}, \theta)\, p(\theta \mid \mathcal{D})\, \mathrm{d}\theta.$$

turns into

$$\mathbb{E}[a] = \frac{\int a(\theta) f(\theta) \mathrm{d}\theta}{\int f(\theta) \mathrm{d}\theta}$$

for

1. $a(\theta) = p(f_\star \mid \mathcal{D}, \theta)$, the predictions, and
2. $f(\theta) = p(\mathcal{D} \mid \theta) p(\theta)$, the product of prior and likelihood of parameters, as

$$p(\theta \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \theta) p(\theta)}{\int p(\mathcal{D} \mid \theta) p(\theta) \mathrm{d}\theta}.$$

Monte Carlo uses a simple approximation for the integral.

Monte Carlo schemes generate a set of samples

$$\{\theta_i; i = 1, \ldots, N\}$$

from $p(\theta)$ (in a variety of ways.)

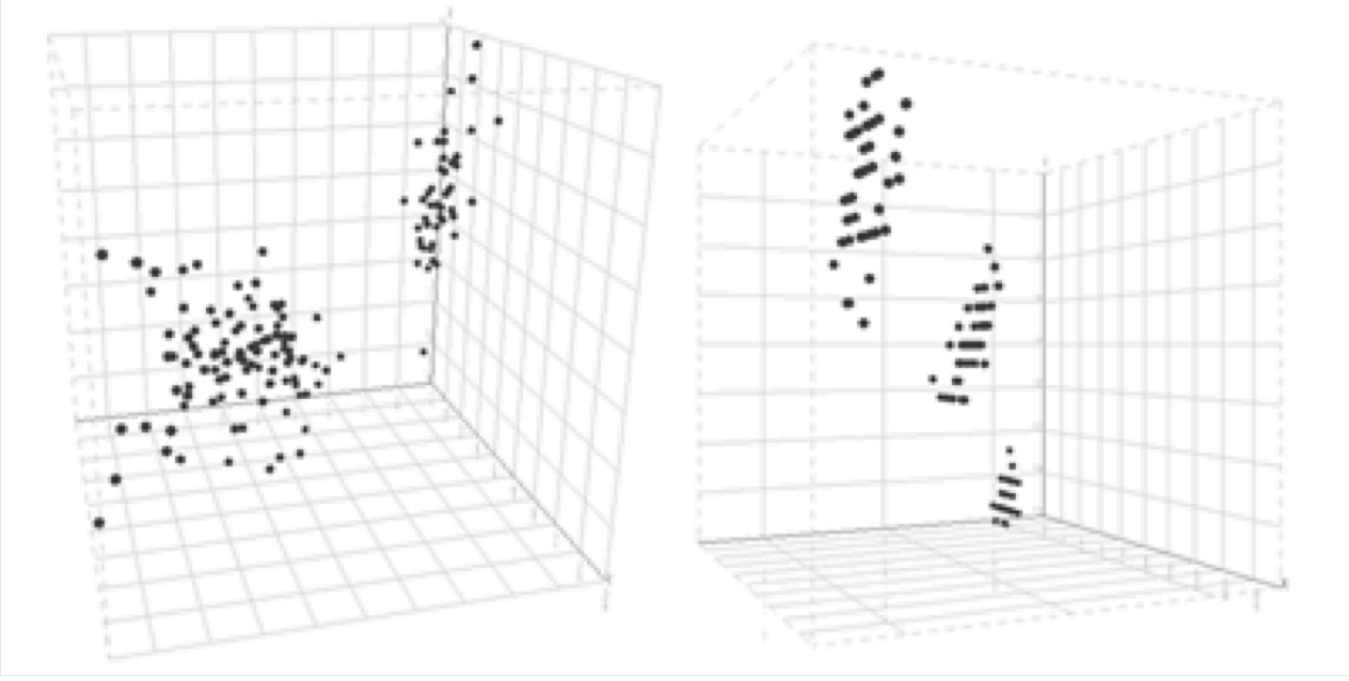
We evaluate $a(\cdot)$ and $f(\cdot)$ at those samples, giving $\{a(\theta_i); i = 1, \ldots, N\}$ and $\{f(\theta_i); i = 1, \ldots, N\}$, respectively.

We then approximate as

$$\mathbb{E}[a] \simeq \frac{1}{N} \sum_{i=1}^{N} a(\theta_i).$$

The space over which we must sample is often high dimensional.

As volume grows exponentially with the dimension, so does the fraction of the space about which a single sample tells us shrink: this is a big challenge.



Rejection sampling uses a distribution $g(\theta)$ from which it is easy to draw samples to then draw samples from $p(\theta)$.

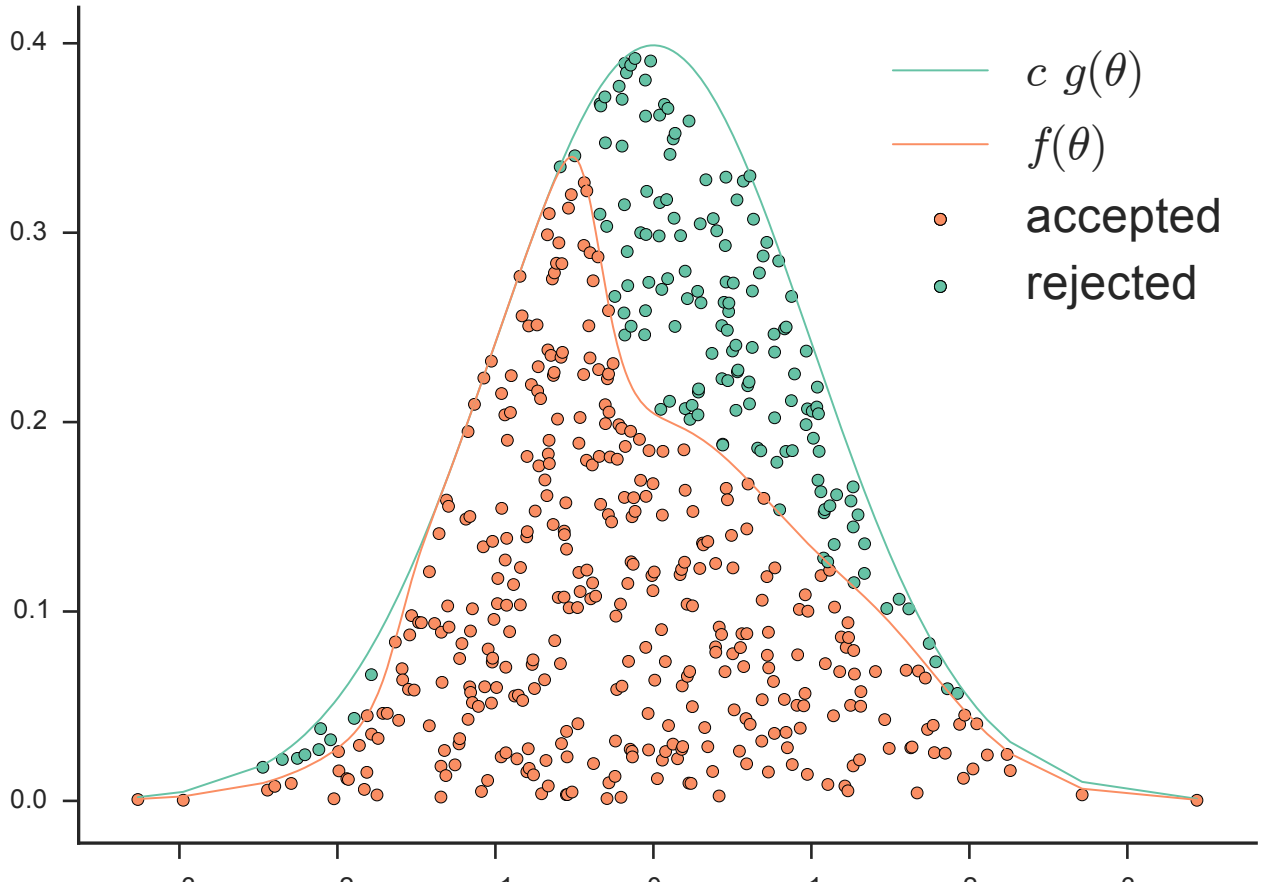
For many standard distributions, like the Gaussian, there exist optimised procedures for drawing samples.

Let's say that we know that $f(\theta) \leq cg(\theta)$, where c is a constant and g is one of those standard distributions.

To generate θ_i, rejection sampling:

1. draws a sample ν from $g(\cdot)$;
2. draws u from the uniform distribution over $[0, 1]$;
3. if $u < \frac{f(\nu)}{cg(\nu)}$, set $\theta_i = \nu$ (accept);
4. otherwise, go back to step 1 (reject) and try again.

1. draw a sample ν from $g(\cdot)$;
2. draw u from the uniform distribution over $[0, 1]$;
3. if $u < \frac{f(\nu)}{cg(\nu)}$, set $\theta_i = \nu$ (accept);
4. otherwise, go back to step 1 (reject) and try again.



Rejection sampling suffers from some practical difficulties.

Firstly, if the bound is not tight (if $f(\theta)$ is much lower than $cg(\theta)$), most samples will be wasted.

Secondly, it requires knowledge of the upper bounding factor c: usually we don't know where the peaks of $f(\theta)$ are.

Rejection sampling is rarely useful for real, high-dimensional, problems, as the rate of accepting a sample reduces exponentially in the dimension of the space.

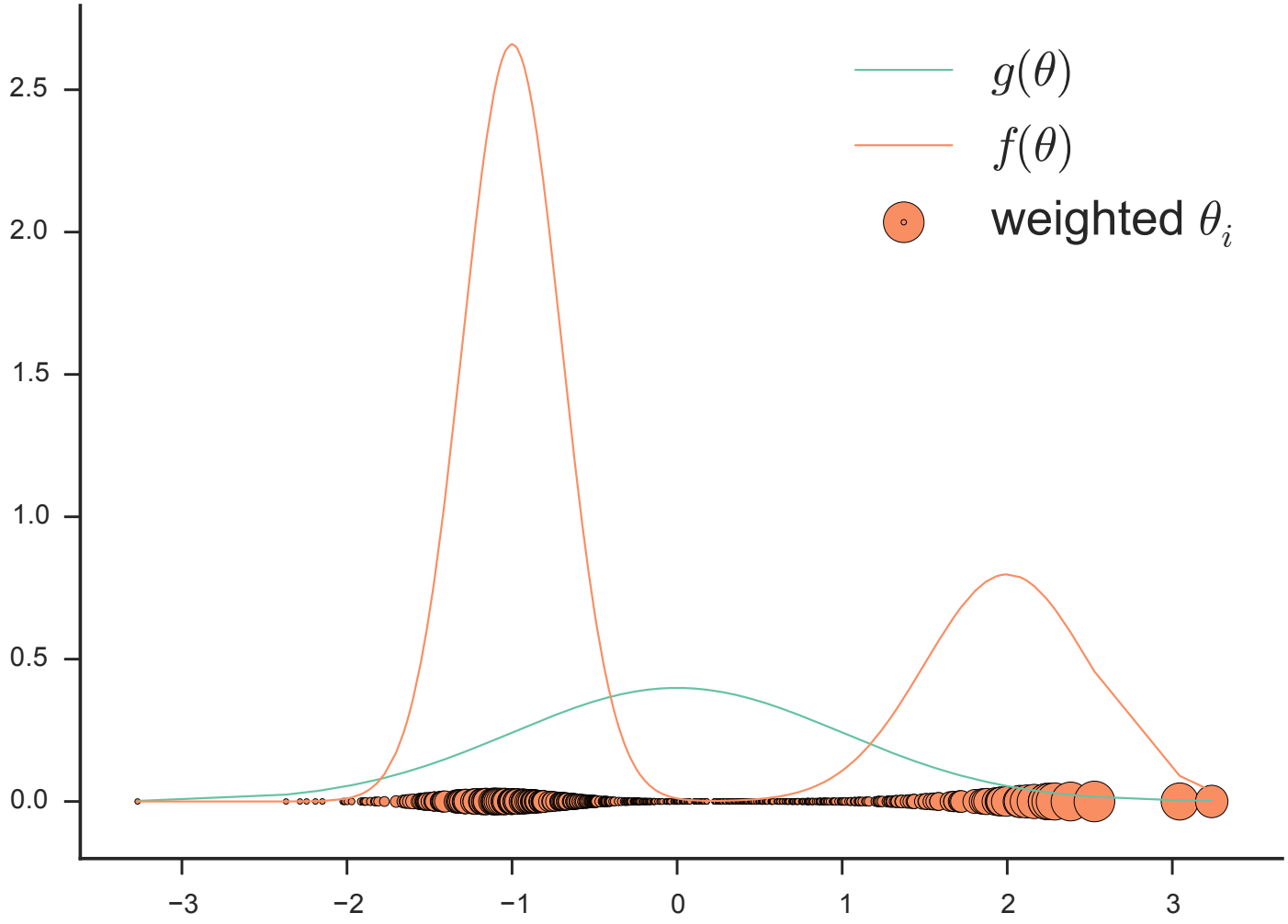
Importance sampling also uses a proposal distribution $g(\theta)$ from which it is easy to draw samples.

Importance sampling gives up trying to sample from $p(\theta)$ and instead simply draws $\{\theta_i; i = 1, \ldots, N\}$ from $g(\theta)$.

We then approximate the expectation using

$$\begin{aligned}
\mathbb{E}[a] &= \frac{\int a(\theta) f(\theta) \mathrm{d}\theta}{\int f(\theta) \mathrm{d}\theta} \\
&= \frac{\int a(\theta) \frac{f(\theta)}{g(\theta)} g(\theta) \mathrm{d}\theta}{\int \frac{f(\theta)}{g(\theta)} g(\theta) \mathrm{d}\theta} \\
&\simeq \frac{\sum_{i=1}^{N} a(\theta_i) \frac{f(\theta_i)}{g(\theta_i)}}{\sum_{i=1}^{N} \frac{f(\theta_i)}{g(\theta_i)}}
\end{aligned}$$

Importance sampling hence weights samples so that they appear more like draws from p.



Importance sampling also suffers from practical difficulties.

Firstly, we must find a proposal g which is broad enough to cover all high values of f.

More importantly, if g is very dissimilar from f, we'll end up with mostly negligibly-weighted samples: not very useful.

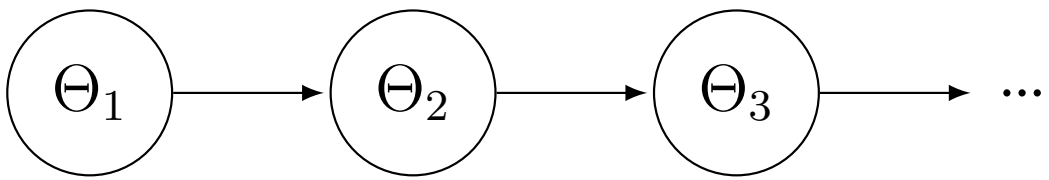
Unfortunately, it's difficult to assess the similarity beforehand!

Markov Chain Monte Carlo (MCMC) methods draw a sample conditional on the previous sample.

MCMC defines

1. an initial distribution $p(\Theta_1 = \theta_1)$ and
2. a transition density $p(\Theta_{i+1} = \theta_{i+1} \mid \Theta_i = \theta_i) = T_i(\theta_{i+1}; \theta_i)$ for drawing the sample θ_{i+1} given that the current sample is θ_i.

If T has no dependence on i, the chain is called homogenous.



Gibbs sampling is an MCMC scheme that sets $\theta_{i+1} = \theta_i$ and then changes a few elements.

This is particularly convenient when θ is very high-dimensional: typically, you may just change one of the elements of θ at a time.

Those elements that are changed are drawn from a specified distribution conditional on the previous value.

To change only the nth element, we often use

$$T_i(\theta_{i+1}^{(n)}; \theta_i) = p(\theta_{i+1}^{(n)} \mid \theta_i^{(-n)})$$

where $\theta^{(-n)}$ is all elements of θ aside from the nth.

Gibbs is hence useful when the conditionals of $p(\theta)$ have a nice form.

Gibbs sampling changes a different set of variables at each iteration.

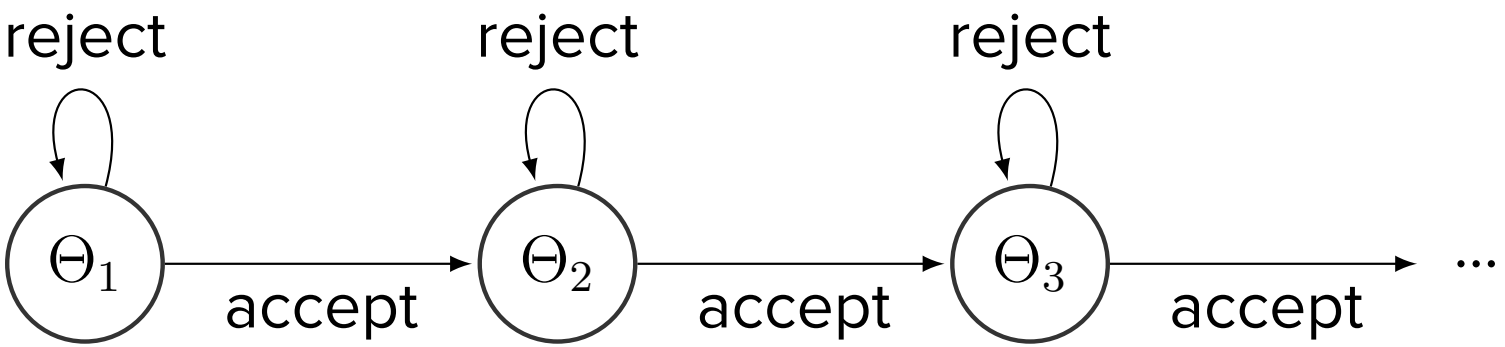
Usually, we just rotate through a pre-specified sequence of variables (e.g. change the first variable, followed by the second, etc.), but the sequence could also itself be determined randomly.

Gibbs is often slow to generate samples where $f(\theta)$ is large, as it can only explore the space with one small, axis-aligned, step at a time.

Metropolis-Hastings (MH) methods apply the idea of rejection to MCMC, taking a different proposal at every iteration.

To generate θ_{i+1}, Metropolis-Hastings:

1. draws a sample ν from a proposal distribution $Q(\cdot;\theta_i)$ (often as simple as a Gaussian centered at θ_i);
2. draws u from the uniform distribution over $[0,1]$;
3. if $u < \min\left(1, \frac{f(\nu)}{f(\theta_i)}\frac{Q(\theta_i;\nu)}{Q(\nu;\theta_i)}\right)$, we set $\theta_{i+1} = \nu$ (accept);
4. else $\theta_{i+1} = \theta_i$ (reject).



Metropolis-Hastings accepts ν if and only if

$$u < \min\left(1, \frac{f(\nu)}{f(\theta_i)} \frac{Q(\theta_i; \nu)}{Q(\nu; \theta_i)}\right)$$

Metropolis-Hastings will tend to favour accepting samples for which:

1. $f(\nu) > f(\theta_i)$ i.e. where we've climbed to a higher value of f; and
2. $Q(\theta_i; \nu) > Q(\nu; \theta_i)$ i.e. where it's easier to get back to θ_i from ν than it is to have got to ν in the first place (so that we don't get stuck!).

Rejection sampling suffers from the requirement of specifying a-priori a proposal g that matches f.

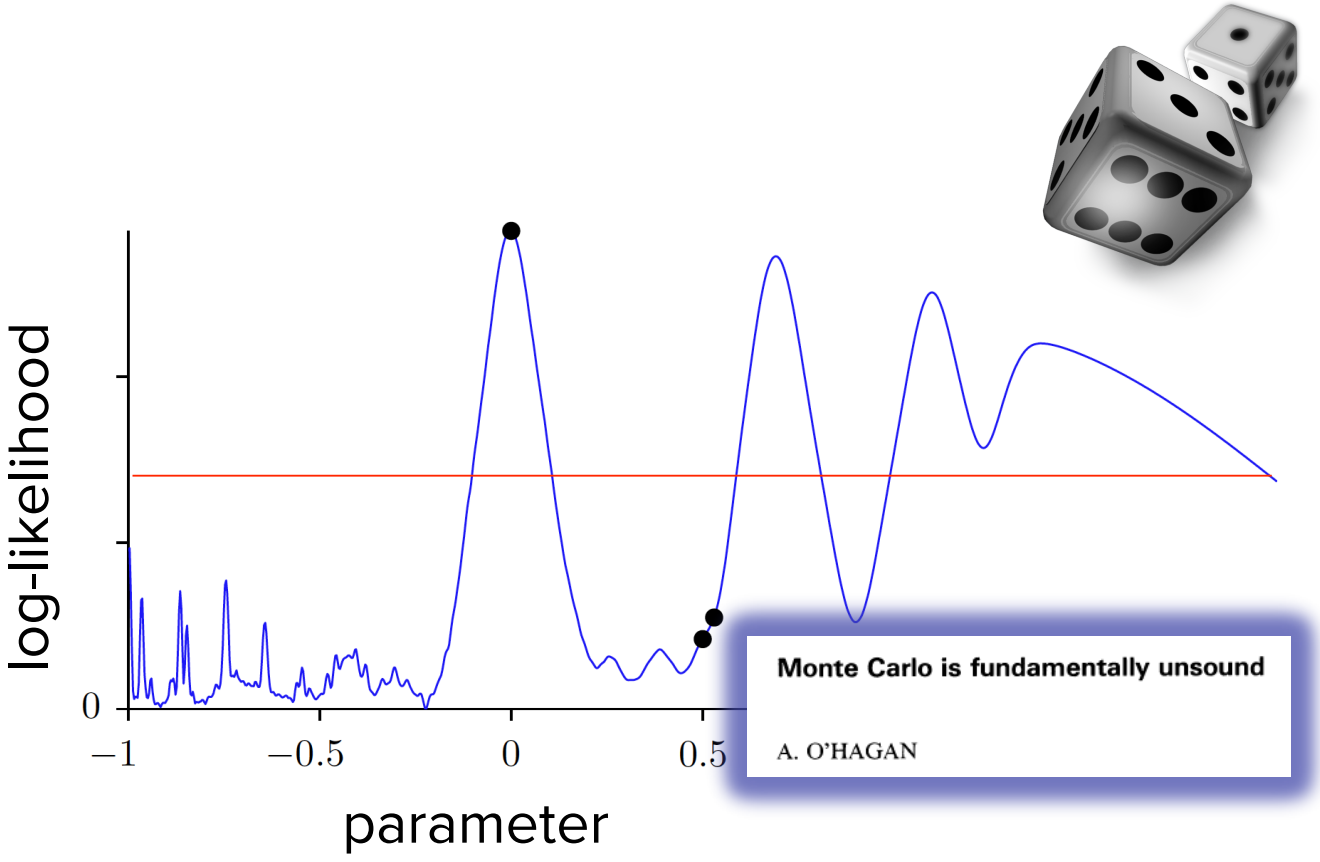
Metropolis-Hastings instead uses a proposal distribution (which need look nothing like f) as a means of (somewhat) local exploration.

The degree of locality needs to be carefully selected: if our steps are too small, we'll never explore the whole space, but if our steps are too large, we'll never exploit local information about high f.

Like Gibbs, Metropolis-Hastings approaches may choose only to modify some subset of the elements of θ_i in moving to θ_{i+1}.

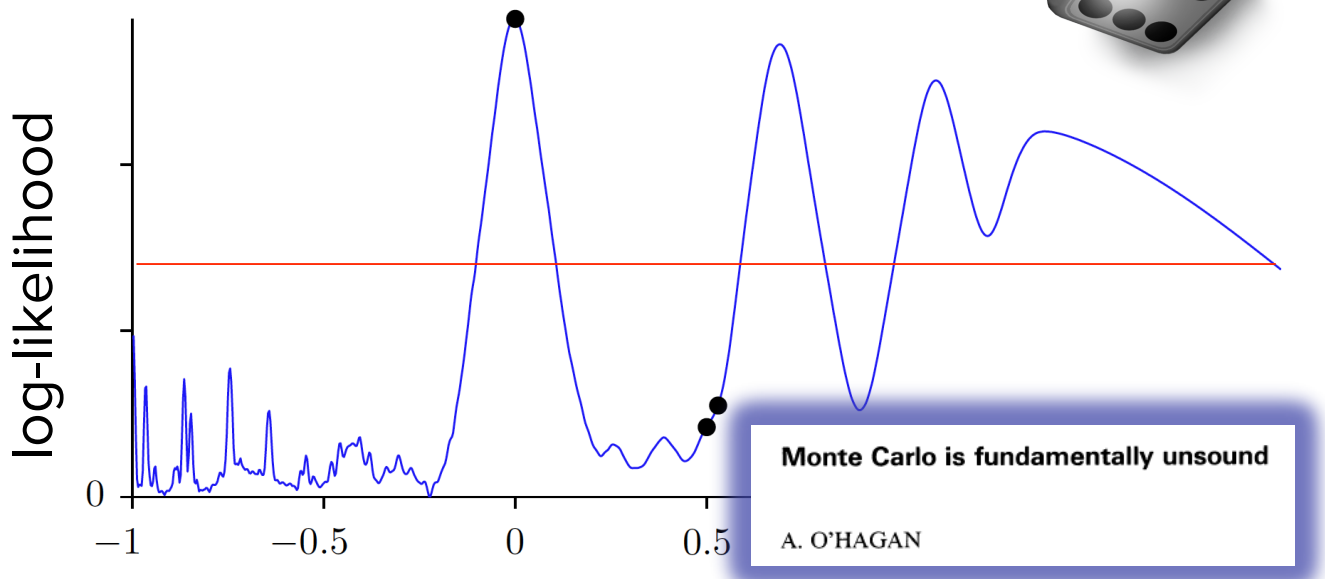
Monte Carlo approaches suffer from several problems:

1. Random sample selection ignores decision theory;
2. The estimate for the integral privileges irrelevant information (how the samples were generated);
3. The estimate also ignores relevant information (like the sample locations).

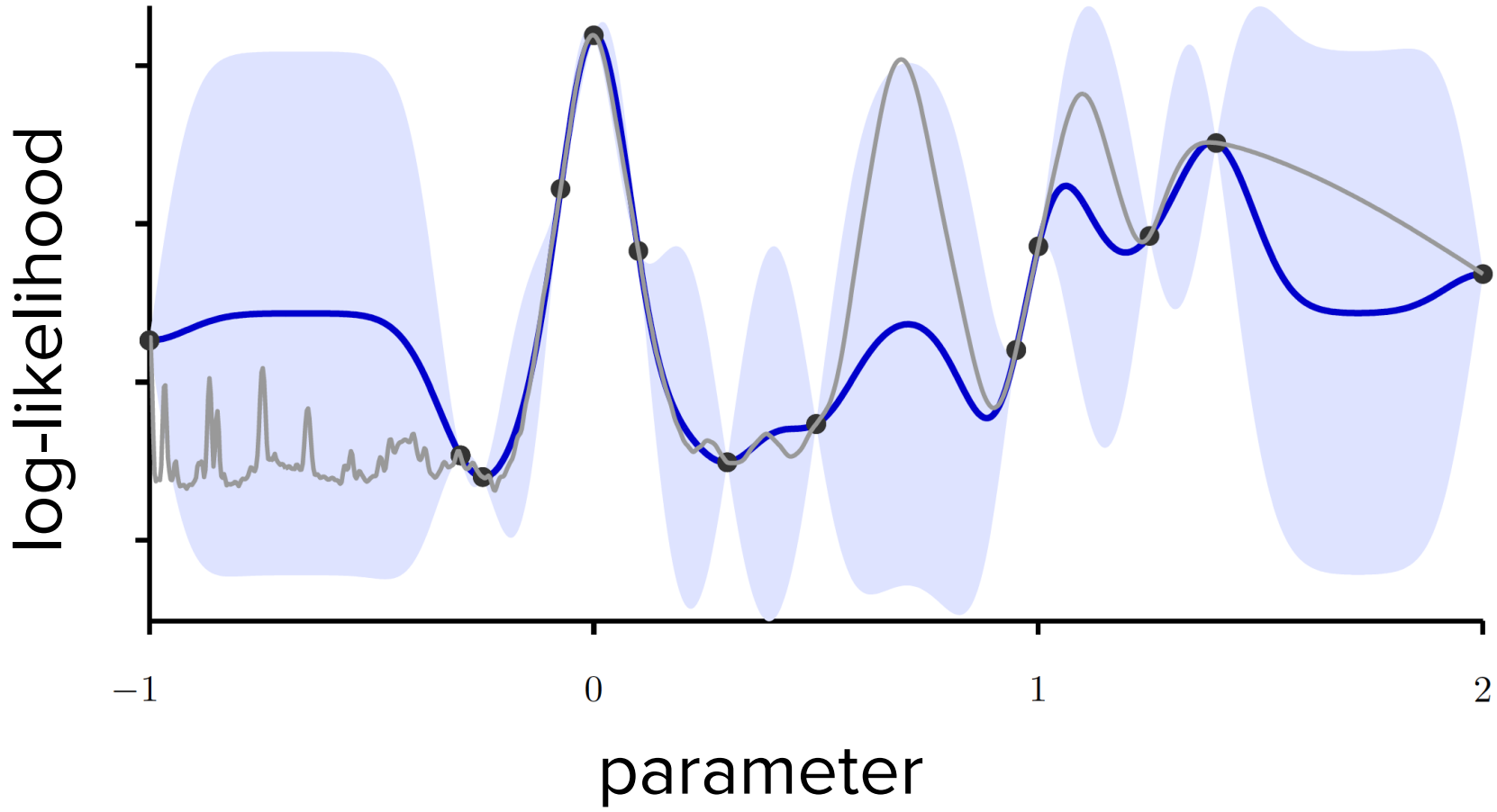


These issues lead to practical consequences:

1. Poor sample efficiency (samples are often selected right next to one another for no good reason);
2. Poor convergence diagnostics (difficulty in determining when you have enough samples);
3. The sampler often has parameters that must be hand-tuned.



Bayesian quadrature tackles these problems by bringing Bayesian inference and decision theory to approximation.



In summary,

1. Laplace's approximation is a simple means of improving upon MLE or MAP.
2. Rejection sampling uses a distribution from which it is easy to draw samples to then draw samples from a desired distribution.
3. Importance sampling weights samples from a distribution to approximate those from a desired distribution.
4. Gibbs sampling changes a few elements of a sample at each step.
5. Metropolis-Hastings randomly samples local to the current location, but only accepts with probability related to the improvement in sample location.
6. All Monte Carlo methods are fundamentally unsound.