2 APPROXIMATE INFERENCE

In the previous chapter we argued that the key computational challenge in performing inference in probabilistic models is being able to evaluate integrals with respect to probability measures defined on high-dimensional spaces. Generally these integrals will not have analytic solutions and for models with even moderate numbers of unobserved variables, numerical quadrature approaches to evaluating integrals are computationally infeasible due to the exponential scaling of computation cost with dimension.

Truth is much too complicated to allow anything but approximations.
—John von Neumann

In this chapter we will review some of the key algorithms proposed for computing *approximate* solutions to inference problems. A unifying aspect to all of these methods is trading off some loss of the accuracy of the answers provided to inferential queries, for a potentially significant increase in computational tractability. The literature on *approximate inference* methods is vast and so necessarily this chapter will only form a partial review of the available methods. We will therefore concentrate on those approaches which are directly relevant to this thesis.

Approximate inference methods can be coarsely divided in to two groups: methods in which the integrals with respect to the target measure are estimated by computing weighted sums over points sampled from a probability measure over the target state space and those in which a more tractable approximation to the target probability measure of interest is found by optimising the approximation to be 'close' in some sense to the target measure. In this chapter we will concentrate sampling-based approaches to approximate inference, focussing in particular on Markov chain Monte Carlo methods, as these form the key basis for the contributions discussed in later chapters.

Although they are not the main focus of this thesis we will make use of several optimisation-based approximate inference methods within the algorithms discussed in the following chapters. We review the ideas underlying these methods in Appendix C.

2.1 MONTE CARLO METHODS

Inference at both the level of computing conditional expectations of unobserved variables in a model and in evaluating evidence terms to allow model comparison involves integrating functions against a probability distribution. Typically the distribution of interest will be defined by a probability density with respect to a base measure. Therefore we wish to be able to compute integrals of the form

$$\int_X f(\boldsymbol{x})\,\mathrm{d}P(\boldsymbol{x}) = \int_X f(\boldsymbol{x})\,p(\boldsymbol{x})\,\mathrm{d}\mu(\boldsymbol{x}) \tag{2.1}$$

where p is the density of a target distribution P on a space X with respect to a base measure μ and f is a measurable function. For instance in the case of computing the *posterior mean* in a Bayesian inference problem with observed variables $\mathbf{y}$ and latent variables $\mathbf{x}$ where the posterior density $\mathrm{p}_{\mathbf{x}|\mathbf{y}}$ is defined with respect to the D-dimensional Lebesgue measure, we would have $p(\boldsymbol{x}) = \mathrm{p}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x} \mid \boldsymbol{y})$ for an observed $\boldsymbol{y}$, $\mu(\boldsymbol{x}) = \lambda^D(\boldsymbol{x})$ and $f(\boldsymbol{x}) = \boldsymbol{x}$. Often we will only be able to evaluate p up to an unknown unnormalising constant i.e. $p(\boldsymbol{x}) = \frac{1}{Z}\tilde{p}(\boldsymbol{x})$ with we able to evaluate $\tilde{p}$ pointwise but Z intractable to compute. For example in a Bayesian inference setting $\tilde{p}(\boldsymbol{x})$ would be the joint density $\mathrm{p}_{\mathbf{x},\mathbf{y}}(\boldsymbol{x}, \boldsymbol{y})$ and Z the model evidence $\mathrm{p}_{\mathbf{y}}(\boldsymbol{y})$. When peforming inference in undirected models, we would instead have that $\tilde{p}$ is the product of unnormalised factors and Z the corresponding normaliser.

2.1.1 Monte Carlo integration

The eponym of the Monte Carlo method is a Monocan casino, favoured haunt of the uncle of Stanisław Ulam, one of the method's inventors.

The framework that unifies all of the methods we will discuss in this section is the *Monte Carlo* method for integration [64]. Let $\mathbf{x}$ be a random vector distributed according to the target distribution i.e. $\mathrm{P}_{\mathbf{x}} = P$. Given an arbitrary measurable function $f : X \to \mathbb{R}$ we define a random variable $\mathrm{f} = f(\mathbf{x})$. Our task is to compute expectations $\mathbb{E}[\mathrm{f}] = \bar{f}$ corresponding to the integral (2.1). We assume that $\mathbb{E}[\mathrm{f}]$ exists and both $\mathbb{E}[\mathrm{f}]$ and $\mathbb{V}[\mathrm{f}]$ are finite. For now we assume we have a way of generating values of N random variables $\{\mathbf{x}_n\}_{n=1}^N$, each marginally distributed according to the target distribution i.e. $\mathrm{P}_{\mathbf{x}_n} = P \;\forall n \in 1 \ldots N$ but potentially not independent of each other. We define random variables

$$\mathrm{f}_n = f(\mathbf{x}_n) \quad \forall n \in \{1 \ldots N\} \quad \text{and} \quad \hat{\mathrm{f}}_N = \frac{1}{N}\sum_{n=1}^{N} \mathrm{f}_n. \tag{2.2}$$

Due to linearity of the expectation operator, we have that

$$\mathbb{E}\left[\hat{\mathsf{f}}_N\right] = \frac{1}{N}\sum_{n=1}^{N}\mathbb{E}[\mathsf{f}_n] = \frac{1}{N}\sum_{n=1}^{N}\bar{f} = \bar{f} \tag{2.3}$$

and so that in expectation $\hat{\mathsf{f}}_N$ is equal to $\bar{f}$, i.e. realisations of $\hat{\mathsf{f}}_N$ are *unbiased estimators* of $\bar{f}$. Note that this result does not require any independence assumptions about the generated random variables. Now considering the variance of $\hat{\mathsf{f}}_N$ we can show that

$$\mathbb{V}\left[\hat{\mathsf{f}}_N\right] = \frac{\mathbb{V}[\mathsf{f}]}{N}\left(1 + \frac{2}{N}\sum_{n=1}^{N-1}\sum_{m=1}^{n-1}\frac{\mathbb{C}[\mathsf{f}_n, \mathsf{f}_m]}{\mathbb{V}[\mathsf{f}]}\right). \tag{2.4}$$

If the generated random variables $\{\mathbf{x}_n\}_{n=1}^N$ and so $\{\mathsf{f}_n\}_{n=1}^N$ are independent, then $\mathbb{C}[\mathsf{f}_n, \mathsf{f}_m] = 0\ \forall m \neq n$. In this case (2.4) reduces to $\mathbb{V}\left[\hat{\mathsf{f}}_N\right] = \mathbb{V}[\mathsf{f}]/N$, i.e. the variance of the *Monte Carlo estimate* $\hat{\mathsf{f}}_N$ for $\bar{f}$ is inversely proportional to the number of samples N. Importantly this scaling does not depend on the dimension of $\mathbf{x}$.

Therefore if we can generate a set of independent random variables from the target density, we can estimate expectations that asymptotically tend to the true value as N increases, with a typical deviation from the true value (as measured by the standard deviation, i.e. the square root of variance) that is $O\left(N^{-\frac{1}{2}}\right)$. In comparison a fourth-order quadrature method such as *Simpson's rule* has an error that is $O\left(N^{-\frac{4}{D}}\right)$ for a grid of N points uniformly spaced across a D dimensional space. Asymptotically for $D > 8$, Monte Carlo integration will therefore give better convergence than Simpson's rule, and even for smaller dimensions large constant factors in the Simpson's rule dependence can mean Monte Carlo performs better for practical N.

Note that computing Monte Carlo estimates from independent random variables is not optimal in terms of minimising $\mathbb{V}\left[\hat{\mathsf{f}}_N\right]$ for a given f; the covariance terms in (2.4) can be negative which can reduce the overall variance. A wide range of *variance reduction* methods have been proposed to exploit this and produce lower variance of Monte Carlo estimates for a given f [31]. Although these methods can be very important in pratice for achieving an estimator with a practical variance for a specific f of interest, we will generally concentrate on the case where we do not necessarily know f in advance and so cannot easily exploit these methods.

Figure 2.1: Binary representation of linear congruential generator sequence $s_{n+1} = 37s_n + 61 \bmod 128$. Columns left to right represents successive integer states in sequence. From least significant (bottom) to most significant (top), the bits in each column have patterns repeating with periods 2, 4, 8, 16, 32, 64, 128.

2.1.2 Pseudo-random number generation

The generation of random numbers is too important to be left to chance.
—Robert R. Coveyou

Virtually all statistical computations involving random numbers in practice make use of *pseudo-random number generators* (PRNGs). Rather than generating samples from truly random processes[1], PRNGs produce deterministic sequences of integers in a fixed range that nonetheless maintain many of the properties of a random sequence. In particular through careful choice of the updates, sequences with a very long period (number of iterations before the sequence begins repeating), a uniform distribution across the numbers in the sequence range and low correlation between successive states can be constructed.

A very simple example of a class of PRNGs is the *linear congruential generator* [32] which obeys the recurrent update

$$s_{n+1} = (as_n + c) \mod m \quad \text{with} \quad 0 < a < m,\ 0 \leq c < m, \tag{2.5}$$

with a, c and m integer parameters. If a, c and m are chosen appropriately, iterating the update (2.5) from an initial seed $0 \leq s_0 < m$, will produce a sequence of states which visits all the integers in $[0, m)$ before repeating. An example state sequence with $m = 128$ is shown in Figure 2.1. In practice, linear congruential generators produce sequences with poor statistical properties, particularly when used to generate random points in high dimensional spaces [35], hence most modern numerical computing libraries use more robust variants such as the *Mersenne-Twister* [36], which is used in all experiments in this thesis.

The raw output of a PRNG is an integer sequence, with typically the sequence elements uniformly distributed over all integers in a range $[0, 2^n)$ for some $n \in \mathbb{N}$. All real values are represented at a finite precision on computers, typically using a floating point representation [1] of *single* (24-bit mantissa) or *double* (53-bit mantissa) precision. Through

1 We consider a true random process as one in which it is impossible to precisely predict the next value in the sequence given the previous values.

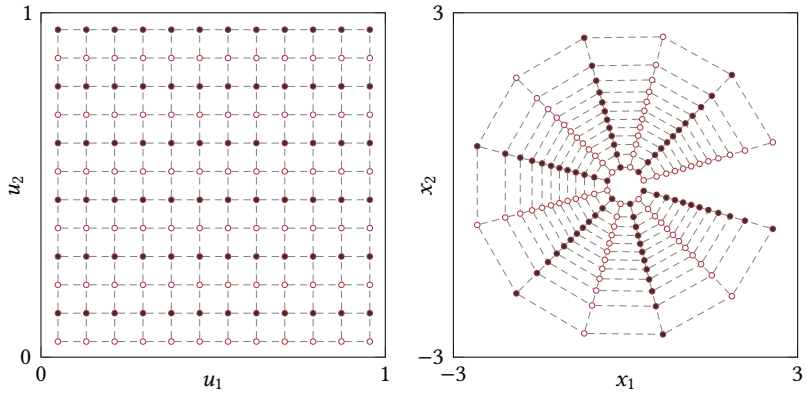


Figure 2.2: Visualisation of Box–Muller transform. Left axis shows uniform grid on $U = [0,1]^2$ and right-axis shows grid points after mapping through $\boldsymbol{g}$ in transformed space $X = \mathbb{R}^2$.

an approriate linear transformation, the integer outputs of a PRNG can be converted to floating-point values uniformly distributed across a finite interval. PRNG implementations typically provide a primitive to generate floating-point values uniformly distributed on $[0,1)$. Given the ability to generate sequences of (effectively) independent samples from a uniform distribution $\mathcal{U}(0,1)$, the question is then how to use these values to produce random samples from arbitrary densities.

2.1.3 Transform sampling

Samples from many standard distributions can be generated by exploiting the transform of random variables relationships discussed in 1.1.4. Let $\mathbf{u}$ be a D-dimensional vector of independent random variables marginally distributed according to $\mathcal{U}(0,1)$ and $\boldsymbol{g} : [0,1)^D \to X$ be a bijective map to a vector space $X \subseteq \mathbb{R}^D$. If we define $\mathbf{x} = \boldsymbol{g}(\mathbf{u})$, then by (1.22) we have that

$$\mathrm{p}_{\mathbf{x}}(\boldsymbol{x}) = \left| \frac{\partial \boldsymbol{g}^{-1}(\boldsymbol{x})}{\partial \boldsymbol{x}} \right|. \tag{2.6}$$

For example for $D = 2$, $X = \mathbb{R}^2$ and a bijective map $\boldsymbol{g}$ defined by

$$\boldsymbol{g}\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{bmatrix} \sqrt{-2\log u_1}\cos(2\pi u_1) \\ \sqrt{-2\log u_1}\sin(2\pi u_2) \end{bmatrix}, \quad \boldsymbol{g}^{-1}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{bmatrix} \exp\left(-\frac{1}{2}(x_1^2 + x_2^2)\right) \\ \frac{1}{2\pi}\arctan\left(\frac{x_1}{x_2}\right) \end{bmatrix},$$

then we have that the density of the transformed $\mathbf{x} = \boldsymbol{g}(\mathbf{u})$ is

$$\mathrm{p}_{\mathbf{x}}(\boldsymbol{x}) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{x_1^2}{2}\right)\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{x_2^2}{2}\right), \tag{2.7}$$

i.e. x_1 and x_2 are independent random variables with standard normal densities $\mathcal{N}(0,1)$. This is the *Box–Muller transform* [8], and allows generation of independent standard normal variables given a PRNG primitive for sampling from $\mathcal{U}(0,1)$. A visualisation of the transformation of space applied by the method is shown in Figure 2.2.

A general method for sampling from univarite distributions is to use an inverse *cumulative distribution function* (CDF) transform. For a probability density p on a scalar random variable, the corresponding CDF $r : \mathbb{R} \to [0,1]$ is defined as

$$r(x) = \int_{-\infty}^{x} p(v)\,\mathrm{d}v \implies \frac{\partial r(x)}{\partial x} = p(x). \tag{2.8}$$

If u is a standard uniform random variable and $x = r^{-1}(u)$ then

$$p_x(x) = \left|\frac{\partial r(x)}{\partial x}\right| = p(x). \tag{2.9}$$

To be able to use the inverse CDF transform method we need to be able to evaluate r^{-1}, sometimes termed the *quantile function*. Often neither the CDF or quantile function of a univariate distribution will have closed form solutions however we can use polynomial approximation methods and iterative solvers to evaluate both to arbitary precision [47]. For some distributions such as the standard normal $\mathcal{N}(0,1)$ even though the CDF and quantile function do not have analytic forms in terms of elementary functions it is common for numerical computing libraries to provide numerical approximations to both which are accurate to within small multiples of machine precision. Although the inverse CDF transform method gives a general recipe for sampling from univariate densities, it is not easy to generalise to multivariate densities and alternatives can be simpler to implement and more numerically stable.

2.1.4 Rejection sampling

An important class of generic sampling methods, particularly due their use as a building block in other algorithms, is rejection sampling [45]. Rejection sampling uses the observation that to sample from a probablity density $p : X \to [0,\infty]$ it is sufficient to uniformly sample from the volume under the graph of $(\boldsymbol{x}, p(\boldsymbol{x}))$.

The key requirement in rejection sampling is to identify a *proposal distribution* Q which we can generate independent samples from and has

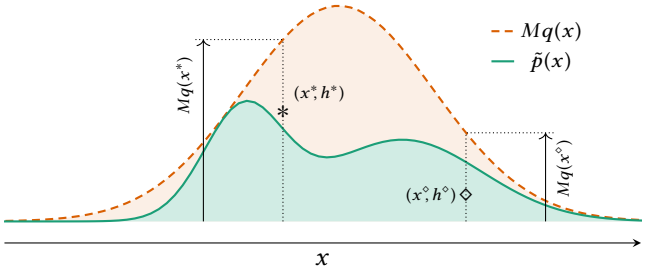


Figure 2.3: Visualisation of rejection sampling. The green curve shows the (unnormalised) target density $\tilde{p}$, the green region underneath representing the area we wish to sample points uniformly from. The dashed orange curve shows the scaled proposal density Mq, with the orange (plus green) region representing the area we uniformly propose values from. Two example proposals are shown: $\diamond$ is under the target density and so accepted; $*$ is outside of the green region and so would be rejected.

Algorithm 1 Rejection sampling.

Input: $\tilde{p}$: unnormalised target density, q : normalised density of proposal distribution Q, M : constant such that $\tilde{p}(\boldsymbol{x}) \leq Mq(\boldsymbol{x})\ \forall \boldsymbol{x} \in X$.
Output: Independent sample from distribution with density $p(\boldsymbol{x}) = \frac{1}{Z}\tilde{p}(\boldsymbol{x})$.

1: **repeat**
2: $\quad \boldsymbol{x} \sim q(\cdot)$
3: $\quad h \sim \mathcal{U}(\cdot \mid 0, Mq(\boldsymbol{x}))$
4: **until** $h \leq \tilde{p}(\boldsymbol{x})$
5: **return** $\boldsymbol{x}$

a density $q = \frac{\partial Q}{\partial \mu}$ that upper bounds the potentially unnormalised target density $\tilde{p}$ across its full support X when multiplied by a known constant M, i.e. $\tilde{p}(x) \leq Mq(x)\ \forall x \in X$. The requirement to be able to generate independent samples from Q can be met for example by distributions amenable to transform sampling, e.g. the standard normal. The second requirement is generally more challenging and as we will see the efficiency of rejection sampling methods is very dependent on how tight the bound can be made.

Algorithm 1 describes the rejection sampling method to produce a single independent sample from a target distribution. A visualisation of how the algorithm works for a univariate target distribution is shown in Figure 2.3. The overall aim is to generate points uniformly distributed across the green area under the (unnormalised) target density curve. This is achieved by generating points uniformly under the dashed orange curve corresponding to the scaled proposal density and then accepting only those which are below the green curve. To generate a point

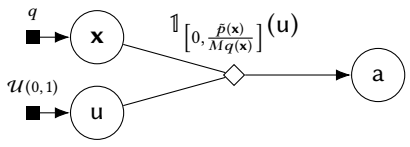


Figure 2.4: Factor graph of rejection sampling process.

under the dashed orange curve we first generate an x from the proposal distribution and then generate an auxilliary 'height' variable by sampling uniformly from $[0, Mq(x)]$. If the sampled height is below the green curve we accept (as in the $\diamond$ example in Figure 2.3) else we reject the sample (as in the $*$ example in Figure 2.3).

Figure 2.4 shows the rejection sampling generative process as a directed factor graph, with $\mathbf{x}$ be a random variable representing the proposal, u the uniform auxiliary variable drawn to sample the 'height' and a a binary variable that indicates whether the proposal is accepted ($a = 1$) or not ($a = 0$). By marginalising out u we have that that

$$\mathrm{p}_{\mathbf{x},\mathrm{a}}(\boldsymbol{x}, a) = q(\boldsymbol{x})\left(\frac{\tilde{p}(\boldsymbol{x})}{Mq(\boldsymbol{x})}\right)^{a}\left(1 - \frac{\tilde{p}(\boldsymbol{x})}{Mq(\boldsymbol{x})}\right)^{1-a}, \tag{2.10}$$

and further marginalising over the proposal $\mathbf{x}$

$$\mathrm{p}_{\mathrm{a}}(a) = \left(\frac{Z}{M}\right)^{a}\left(1 - \frac{Z}{M}\right)^{1-a}. \tag{2.11}$$

Conditioning on the proposal being accepted we therefore have that

$$\mathrm{p}_{\mathbf{x}|\mathrm{a}}(\boldsymbol{x} \mid 1) = \frac{q(\boldsymbol{x})\frac{\tilde{p}(\boldsymbol{x})}{Mq(\boldsymbol{x})}}{\frac{Z}{M}} = \frac{\tilde{p}(\boldsymbol{x})}{Z} = p(\boldsymbol{x}). \tag{2.12}$$

Therefore the accepted proposals are distributed according to the target density as required. Further from (2.11) we have that the $\mathrm{p}_{\mathrm{a}}(1) = \frac{Z}{M}$. This suggests we can use the accept rate to estimate Z but also hints at the difficulty in finding a M which guarantees the upper bound requirement as for $\frac{Z}{M}$ to be a valid probability $M \geq Z$ i.e. M needs to be an upper bound on the unknown normalising constant Z. This relationship also suggests it is desirable to set M as small as possible to maximise the acceptance rate.

Although rejection sampling can be an efficient method of sampling from univariate target distributions (particularly for distributions with log-concave densities where adaptive variants areavailable [22]), it gen-

erally scales very poorly with the dimensionality of the target distribution. This is as the ratio of the volume under the target density to the volume under the scaled proposal density (in terms of Figure 2.3 the ratio of the green area to the green plus orange regions), and so the probability of accepting a proposal, will tend become exponentially smaller as the dimensionality increases. This is an example of the so-called *curse of dimensionality*. Therefore although rejection sampling can be a useful subroutine for generating random variables from low-dimensional densities, in general it is not a viable option for generating samples directly for high-dimensional Monte Carlo integration.

2.1.5 Importance sampling

So far we have considered methods for generating samples directly from a target distribution. Although samples can be of value in themselves for giving a representative set of plausible values from the target distribution (e.g. for visualisation purposes), usually the end goal is to estimate integrals of the form in (2.1).

Importance sampling [30] is a Monte Carlo method which allows arbitrary integrals to be estimated. If Q is a distribution, with density $q = \frac{\partial Q}{\partial \mu}$, which is absolutely continuous with respect to the target distribution (which requires that $p(\boldsymbol{x}) > 0 \Rightarrow q(\boldsymbol{x}) > 0$), then importance sampling is based on the identity

$$\bar{f} = \frac{\int_X f(\boldsymbol{x})\,\tilde{p}(\boldsymbol{x})\,\mathrm{d}\mu(\boldsymbol{x})}{\int_X \tilde{p}(\boldsymbol{x})\,\mathrm{d}\mu(\boldsymbol{x})} = \frac{\int_X f(\boldsymbol{x})\,\frac{\tilde{p}(\boldsymbol{x})}{q(\boldsymbol{x})}\,q(\boldsymbol{x})\,\mathrm{d}\mu(\boldsymbol{x})}{\int_X \frac{\tilde{p}(\boldsymbol{x})}{q(\boldsymbol{x})}\,q(\boldsymbol{x})\,\mathrm{d}\mu(\boldsymbol{x})}. \tag{2.13}$$

Each of the numerator and denominator in (2.13) take the form of an expectation of a measurable function of a random variable $\mathbf{x}$ with distribution Q. Further the denominator is exactly equal to $Z = \int_X \tilde{p}(\boldsymbol{x})\,\mathrm{d}\mu(\boldsymbol{x})$. We therefore have that

$$Z\bar{f} = \mathbb{E}[w(\mathbf{x})f(\mathbf{x})] \;\text{ and }\; Z = \mathbb{E}[w(\mathbf{x})] \;\text{ with }\; w(\boldsymbol{x}) = \frac{\tilde{p}(\boldsymbol{x})}{q(\boldsymbol{x})}. \tag{2.14}$$

If we can generate random variables $\{\mathbf{x}_n\}_{n=1}^N$ each with marginal density q we can therefore form Monte Carlo estimates of both the numerator and denominator. We define $\hat{\mathrm{Z}}_N$ and $\hat{\mathrm{g}}_N$ as

$$\hat{\mathrm{Z}}_N = \frac{1}{N}\sum_{n=1}^{N} w(\mathbf{x}_n) \;\text{ and }\; \hat{\mathrm{g}}_N = \frac{1}{\hat{\mathrm{Z}}}\sum_{n=1}^{N} w(\mathbf{x}_n)f(\mathbf{x}_n). \tag{2.15}$$

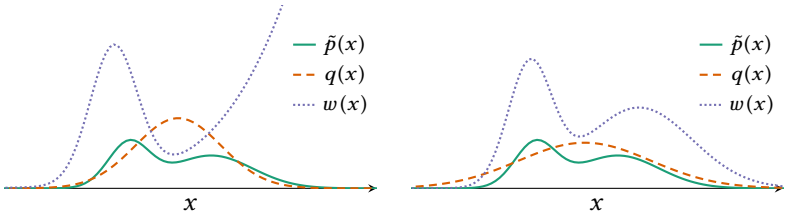


Figure 2.5: Visualisation of importance sampling. On both axes the green curve shows the unnormalised target density $\tilde{p}$, the dashed orange curve the density q values are sampled from and the dotted violet curve the importance weighting function $w(x) = \frac{\tilde{p}(x)}{q(x)}$ to estimate expectations with respect to the target density using samples from q. In the left axis the q chosen is undispersed compared to $\tilde{p}$ leading to very large w values in the right tail. In constrast in the right axis, the broader q leads to less extreme variation in w.

By the same argument as Section 2.1.1, $\mathbb{E}\left[\hat{Z}_N\right] = Z$ and $\mathbb{E}[\hat{g}_N] = Z\bar{f}$. We can therefore use importance sampling to form an unbiased estimate of the unknown normalising constant Z.

If we define $\hat{f}_N = \hat{g}_N / \hat{Z}_N$, then this is a biased[2] estimator for $\bar{f}$ as in general the expectation of the ratio of two random variables is not equal to the ratios of their expectations. However if both the numerator and denominator have finite variance, i.e. $\mathbb{V}\left[\hat{Z}_N\right] < \infty$ and $\mathbb{V}[\hat{g}_N] < \infty$, then $\hat{f}_N$ is a *consistent* estimator for $\bar{f}$ i.e. $\lim_{N\to\infty} \mathbb{E}\left[\hat{f}_N\right] = \bar{f}$.

The $w(x_n)$ values are typically termed the *importance weights*. If a few of the weights are very large, the weighted sums in (2.15) will be dominated by those few values, reducing the effective number of samples in the Monte Carlo estimates. This can particularly be a problem if the are regions of X with low probability under q where $p(\boldsymbol{x}) \gg q(\boldsymbol{x})$. As sampling points in these regions will be a rare event, a large number of samples may be needed to diagnose the issue adding further difficulty. A general recommendation is to use densities q with tails as least as heavy of those of p, and in general the closer the match between q and p the better [34, 48]. Figure 2.5 shows a visualisation of the effect of the choice of q on the importance weights.

When previously discussing rejection sampling, we introduced an auxiliary binary accept indicator variable, a, associated with each proposed

2 In cases where the normalising constant Z is known, we can instead use $w(\boldsymbol{x}) = \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}$ in which case the ratio estimator is not required and an unbiased estimates can be calculated. As the problems we are interested in will generally have unknown Z we do not consider this case further

sample $\mathbf{x}$ (see Figure 2.4). If we generate N independent proposal – indicator pairs $\{\mathbf{x}_n, \mathsf{a}_n\}_{n=1}^N$ then the number of accepted proposals is $\mathsf{N}_{\text{acc}} = \sum_{n=1}^N \mathsf{a}_n$. Conditioned on N_{acc} being a value more than one, the generated rejection sampling variables $\{\mathbf{x}_n, \mathsf{a}_n\}_{n=1}^N$ can be used to form an *unbiased* Monte Carlo estimate of $\bar{f}$ using the estimator

$$\hat{\mathsf{f}}_N^{\text{RS}} = \frac{\sum_{n=1}^N \mathsf{a}_n f(\mathbf{x}_n)}{\sum_{m=1}^N \mathsf{a}_m}, \tag{2.16}$$

which just correponds to computing the empirical mean of the accepted proposals i.e. the standard Monte Carlo estimator. In comparison importance sampling forms a biased but consistent estimator for $\bar{f}$ from N samples $\{\mathbf{x}_n\}_{n=1}^N$ from a density q using the estimator

$$\hat{\mathsf{f}}_N^{\text{IS}} = \frac{\sum_{n=1}^N w(\mathbf{x}_n) f(\mathbf{x}_n)}{\sum_{m=1}^N w(\mathbf{x}_m)}. \tag{2.17}$$

From this perspective the accept indicators a_n in rejection sampling can be seen to act like binary importance weights, in contrast importance sampling using 'soft' weights which mean all sampled $\mathbf{x}_n$ make a contribution to the estimator (assuming $w(\boldsymbol{x}) \neq 0 \; \forall \boldsymbol{x} \in X$). However this correspondence is only loose. The rejection sampling estimator $\hat{\mathsf{f}}_N^{\text{RS}}$ is unbiased unlike $\hat{\mathsf{f}}_N^{\text{IS}}$, but this unbiasedness relies on conditioning on a non-zero value for N_{acc} (i.e. the number of accepted samples to generate) and continuing to propose points until this condition is met. In contrast importance sampling generates a fixed number of samples from q and does not use any auxiliary random variables.

Unlike rejection sampling, there is no need in importance sampling for q to upper-bound the target density (when multiplied by a constant). This allows more freedom in the choice of q however it is still important to choose q to be as close as possible to the target while remaining tractable to generate samples from. In general for target densities defined on high-dimensional spaces, it can be difficult to find an appropriate q such that the variation in importance weights is not too extreme [34].

2.2 MARKOV CHAIN MONTE CARLO

The sampling methods considered in the previous section used independent random variables to form Monte Carlo estimates. When in-

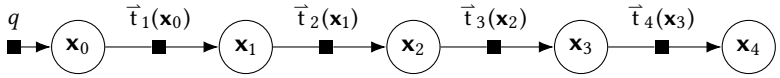


Figure 2.6: Markov chain factor graph. The initial state $\mathbf{x}_0$ is sampled according to a density q and each subsequent state $\mathbf{x}_n$ is then generated from a transition density $\vec{\mathrm{t}}_n$ conditioned on the previous state $\mathbf{x}_{n-1}$.

troducing the Monte Carlo method we saw that is was not necessary for the random variables used in a Monte Carlo estimator to be independent. While it can be impractically computationally expensive to generate independent samples from complex high-dimensional target distributions, simulating a stochastic process which converges in distribution to the target and produces a sequence of *dependent* random variables is often a more tractable task. This is the idea exploited by *Markov chain Monte Carlo* (MCMC) methods.

A *Markov chain* is an ordered sequence of random variables $\{\mathbf{x}_n\}_{n=0}^N$ which have the *Markov property* — for all $n \in \{1 \ldots N\}$, $\mathbf{x}_n$ is conditionally independent of $\{\mathbf{x}_n\}_{m<n-1}$ given $\mathbf{x}_{n-1}$. This conditional independence structure is visualised as a factor graph in Figure 2.6.

For a Markov chain defined on a general measurable state space $(X, \mathcal{F})$, the probability distribution of a state $\mathbf{x}_n$ given the state $\mathbf{x}_{n-1}$ is specified for each $n \in \{1 \ldots N\}$ by a *transition operator*, $\vec{\mathrm{T}}_n : \mathcal{F} \times X \to [0, 1]$. In particular the transition operators define a series of regular conditional distributions for each $n \in \{1 \ldots N\}$

$$\mathrm{P}_{\mathbf{x}_n|\mathbf{x}_{n-1}}(A \,|\, \boldsymbol{x}) = \vec{\mathrm{T}}_n(A \,|\, \boldsymbol{x}) \quad \forall A \in \mathcal{F},\ \boldsymbol{x} \in X. \tag{2.18}$$

We will often assume that the chain is *homogeneous*, i.e. that the same transition operator is used for all steps $\vec{\mathrm{T}}_n = \vec{\mathrm{T}}\ \forall n \in \{1 \ldots N\}$.

The key property required of a transition operator for use in MCMC methods is that the target distribution P is *invariant* under the transition, that is it satisfies

$$P(A) = \int_X \vec{\mathrm{T}}(A \,|\, \boldsymbol{x})\, \mathrm{d}P(\boldsymbol{x}) \quad \forall A \in \mathcal{F}, \tag{2.19}$$

The invariance property means that if a chain state $\mathbf{x}_n$ is distributed according to the target P, all subsequent chain states $\mathbf{x}_{n+1}$, $\mathbf{x}_{n+2} \ldots$ will also be marginally distributed according to the target. Therefore given a single random sample $\mathbf{x}_0$ from the target distribution, a series of de-

pendent states marginally distributed according to the target could be generated and used to form Monte Carlo estimates of expectations.

Being able to generate even one exact sample from a complex high-dimensional target distribution is generally infeasible. Importantly however the marginal distribution on the chain state $\mathsf{P}_{\mathbf{x}_n}$ of a Markov chain with a transition operator which leaves the target distribution invariant will converge to the target distribution irrespective of the distribution of the intial chain state if the target distribution is the *unique* invariant distribution of the chain.

To have a unique invariant distribution, a chain must be *irreducible* and *aperiodic* [62]. For a chain on a measurable space $(X, \mathcal{F})$, irreducibility is defined with respect to a measure ν, which could but does not necessarily need to be the target distribution P. A chain is ν-irreducible if starting at any point in X there is a non-zero probability of moving to any set with positive ν-measure in a finite number of steps, i.e.

$$\forall \boldsymbol{x} \in X,\, A \in \mathcal{F} : \nu(A) > 0 \;\; \exists\, m \in \mathbb{Z}^+ : \mathsf{P}_{\mathbf{x}_m \mid \mathbf{x}_0}(A \mid \boldsymbol{x}) > 0. \tag{2.20}$$

A chain is periodic (and aperiodic otherwise) if disjoint regions of X are visited cyclically, i.e. there exists an integer $r > 1$ and an ordered set of r disjoint P-positive subsets of X, $\{A_i\}_{i=1}^{r}$ such that $\vec{\mathsf{T}}(A_j \mid \boldsymbol{x}) = 1\ \forall \boldsymbol{x} \in A_i,\ i \in \{1 \ldots r\},\ j = (i+1) \mod r$.

If we can construct a ν-irreducible and aperiodic Markov chain $\{\mathbf{x}_n\}_{n=0}^{N}$ which has the target distribution P as its invariant distribution, then a MCMC estimator $\hat{\mathsf{f}}_N = \frac{1}{N}\sum_{n=1}^{N} f(\mathbf{x}_n)$ converges almost surely as $N \to \infty$ to $\bar{f} = \int_X f \mathrm{d}P$ for all starting states except for a ν-null set[3] [39]. This convergence of *time-averages* (i.e. over states at different steps of the Markov chain) to *space-averages* (i.e. with respect to the stationary distribution across the state space), is termed *ergodicity* and is a consequence of the *Birkhoff–Khinchin ergodic theorem* [7].

Although irreducibility and aperiodicity of a Markov chain which leaves the target distribution invariant are sufficient for convergence of MCMC estimators, this does not tell us anything about the rate of that convergence and so how to quantify the error introduced by computing estimates with a Markov chain simulated for only a finite number of

3 The 'except for a ν-null set' caveat can be removed by requiring the stronger property of *Harris recurrence* [26].

steps. Stronger notions of ergodicity can be used to help quantify convergence; we will concentrate on *geometric ergodicity* here. We first define a notion of distance between two measures μ and ν on a measurable space $(X, \mathcal{F})$, the *total variation distance*, as

$$\|\mu - \nu\|_{\text{TV}} = \sup_{A \in \mathcal{F}} |\mu(A) - \nu(A)|. \tag{2.21}$$

For a ν-irreducible and aperiodic chain with invariant distribution P our earlier statement that the distribution on the chain state converges to P can now be restated more precisely as that for ν-almost all initial states $\mathbf{x}_0 = \boldsymbol{x}$, $\lim_{n\to\infty} \|\mathsf{P}_{\mathbf{x}_n|\mathbf{x}_0}(\cdot \mid \boldsymbol{x}) - P\|_{\text{TV}} = 0$. Geometric ergodicity makes a stronger statement that the convergence in total variation distance conditioned is geometric in n, i.e. that

$$\|\mathsf{P}_{\mathbf{x}_n|\mathbf{x}_0}(\cdot \mid \boldsymbol{x}) - P\|_{\text{TV}} \leq M(\boldsymbol{x}) r^n \tag{2.22}$$

for a positive measurable function M which depends on the initial chain state $\boldsymbol{x}$ and rate constant $r \in [0, 1)$. For chains which are geometrically ergodic, we can derive an expression for the *asymptoptic variance* of an MCMC estimator $\hat{\mathsf{f}}_N$ related to the variance of a simple Monte Carlo estimator previously considered in Section 2.1.1.

A stochastic process is stationary if the joint distribution of the states at any set of time points does not change if all those times are shifted by a constant.

As in Section 2.1.1 we define $\mathsf{f}_n = f(\mathbf{x}_n)$ and $\hat{\mathsf{f}}_N = \frac{1}{N}\sum_{n=1}^N \mathsf{f}_n$, with here the $\{\mathbf{x}_n\}_{n=1}^N$ the states of a Markov chain. For a homogeneous Markov chain with a unique invariant distribution P which is *stationary*, the marginal density on the states $\mathsf{P}_{\mathbf{x}_n}$ is equal to P for all n and we can use the expression for the variance of a general Monte Carlo estimator (which did not assume independence of the random variables) stated earlier in (2.4). Further the stationarity of the chain means that the covariance $\mathbb{C}[\mathsf{f}_n, \mathsf{f}_m]$ depends only on the difference $n - m$, and so the variance of the estimator simplifies to

$$\mathbb{V}\left[\hat{\mathsf{f}}_N\right] = \frac{\mathbb{V}[\mathsf{f}]}{N}\left(1 + 2\sum_{n=1}^{N-1}\left(\frac{N-n}{N}\frac{\mathbb{C}[\mathsf{f}_0, \mathsf{f}_n]}{\mathbb{V}[\mathsf{f}]}\right)\right). \tag{2.23}$$

If we multiply both sides of (2.23) by N and define $\rho_n = \frac{\mathbb{C}[\mathsf{f}_0, \mathsf{f}_n]}{\mathbb{V}[\mathsf{f}]}$ (the lag n autocorrelations of f), under the assumption that $\sum_{n=1}^\infty |\rho_n| < \infty$ in the limit of $N \to \infty$ we have that

$$\lim_{N\to\infty}\left(N\,\mathbb{V}\left[\hat{\mathsf{f}}_N\right]\right) = \mathbb{V}[\mathsf{f}]\left(1 + 2\sum_{n=1}^{\infty}\rho_n\right). \tag{2.24}$$

Now considering a chain which is geometrically ergodic from its initial state, if $\mathbb{E}\left[|\mathsf{f}|^{2+\delta}\right]$ is finite for some $\delta > 0$ then it can be shown [11, 19, 54] that (2.24) is also the asymptoptic variance for a MCMC estimator calculated using the chain states.

This motivates a definition of the *effective sample size* (ESS) for an MCMC estimator $\hat{\mathsf{f}}_N$ computed using a geometrically ergodic chain as

$$N_{\text{eff}} = \frac{N}{1 + 2\sum_{n=1}^{\infty} \rho_n}. \tag{2.25}$$

The ESS quantifies the number of independent samples that would be required in a Monte Carlo estimator to give an equivalent variance to the MCMC estimator $\hat{\mathsf{f}}_N$ *in the asymptoptic limit* $N \to \infty$. In practice we cannot evaluate the exact autocorrelations and so we can only compute an estimated ESS, $\hat{N}_{\text{eff}}$, from one or more simulated chains with the estimation method needing to be carefully chosen to ensure reasonable values [61]. Although the assumption of geometric ergodicity can often be hard to verify in practice and ESS estimates can give misleading results in chains far from convergence, when used appropriately estimated ESSs can still be a useful heuristic for evaluating and comparing the effiency of Markov chain estimators and are often available as a standard diagnostic in MCMC software packages [9, 51, 58].

So far we have not discussed how to construct a transition operator giving a chain with the required invariant distribution. As a notational convenience we will consider the transition operator as being specified by a conditional density we term the *transition density* $\vec{\mathsf{t}} : X \times X \to [0, \infty)$ which is defined with respect to a base measure μ (which we assume to be the same as that which the target density we wish to integrate against is defined with respect to, hence the reuse of notation). The transition operator is then

$$\vec{\mathsf{T}}(A \mid \boldsymbol{x}) = \int_A \vec{\mathsf{t}}(\boldsymbol{x}' \mid \boldsymbol{x}) \, \mathrm{d}\mu(\boldsymbol{x}') \quad \forall A \in \mathcal{F}, \ \boldsymbol{x} \in X. \tag{2.26}$$

In practice the probability measure defined by a transition operator will often have a singular component, for example corresponding to a non-zero probability of the chain remaining in the current state. In this case $\vec{\mathsf{T}}$ is not absolutely continuous with respect to μ and a transition density is not strictly well defined. As we did in the previous chapter however we will informally use Dirac deltas to represent a 'density' of

singular measures, and so still consider a transition density as existing. The requirement that the transtion operator leaves the target distribution invariant, can then be expressed in terms of the target density p and transition density $\vec{\mathrm{t}}$ as

$$p(\boldsymbol{x}') = \int_X \vec{\mathrm{t}}(\boldsymbol{x}' \mid \boldsymbol{x})\, p(\boldsymbol{x})\, \mathrm{d}\mu(\boldsymbol{x}) \quad \forall \boldsymbol{x}' \in X. \tag{2.27}$$

Finding a transition density which leaves the target density invariant by satisfying (2.27) seems difficult in general as it involves evaluating an integral against the target density - precisely the computational task which we have been forced to seek approximate solutions to. We can make progress by considering the joint density of a pair of successive states for a chain with invariant distribution P that has converged to stationarity. Then we have that

$$\mathrm{p}_{\mathbf{x}_n, \mathbf{x}_{n-1}}(\boldsymbol{x}', \boldsymbol{x}) = \mathrm{p}_{\mathbf{x}_n \mid \mathbf{x}_{n-1}}(\boldsymbol{x}' \mid \boldsymbol{x})\, \mathrm{p}_{\mathbf{x}_{n-1}}(\boldsymbol{x}) = \vec{\mathrm{t}}(\boldsymbol{x}' \mid \boldsymbol{x})\, p(\boldsymbol{x}). \tag{2.28}$$

We can also consider factorising this joint density into the product of the marginal density of the current state $\mathrm{p}_{\mathbf{x}_n}$ and the conditional density of the previous state given the current state $\mathrm{p}_{\mathbf{x}_{n-1} \mid \boldsymbol{x}_n}$. Due to stationarity $\mathrm{p}_{\mathbf{x}_n}$ is also equal to p and so we have that $\mathrm{p}_{\mathbf{x}_{n-1} \mid \boldsymbol{x}_n}$ must be the density of a transition operator which also leaves P invariant, corresponding to a time reversed version of the original (stationary) Markov chain[4]. If we therefore denote $\overleftarrow{\mathrm{t}} = \mathrm{p}_{\mathbf{x}_{n-1} \mid \boldsymbol{x}_n}$ (and which we will term the *backward transition density* in contrast to $\vec{\mathrm{t}}$ which in this context we will qualify as the *forward transition density*), we have that

$$\vec{\mathrm{t}}(\boldsymbol{x}' \mid \boldsymbol{x})\, p(\boldsymbol{x}) = \overleftarrow{\mathrm{t}}(\boldsymbol{x} \mid \boldsymbol{x}')\, p(\boldsymbol{x}') \quad \forall \boldsymbol{x} \in X,\, \boldsymbol{x}' \in X. \tag{2.29}$$

Integrating both sides with respect to $\boldsymbol{x}$, we have that $\forall \boldsymbol{x}' \in X$

$$\int_X \vec{\mathrm{t}}(\boldsymbol{x}' \mid \boldsymbol{x})\, p(\boldsymbol{x})\, \mathrm{d}\mu(\boldsymbol{x}) = \int_X \overleftarrow{\mathrm{t}}(\boldsymbol{x} \mid \boldsymbol{x}')\, \mathrm{d}\mu(\boldsymbol{x})\, p(\boldsymbol{x}') = p(\boldsymbol{x}'), \tag{2.30}$$

and so that (2.27) is satisfied, with the last inequality arising due to $\overleftarrow{\mathrm{t}}$ being a normalised density on its first argument. Therefore if we can find a pair of transition densities, $\vec{\mathrm{t}}$ and $\overleftarrow{\mathrm{t}}$, satisfying (2.29), then the transition operator specified by $\vec{\mathrm{t}}$ will leave the target distribution P

4 The time reversal of a Markov chain is always itself a a Markov chain irrespective of stationarity (as the defininining conditional independence structure is symmetric with respect to the direction of time), however the reverse of a homogeneous Markov chain which is not stationary will not in general itself be homogeneous.

invariant (and by an equivalent argument so will the transition operator specified by $\overleftarrow{\mathrm{t}}$). We can further simplify (2.29) by requiring that $\overrightarrow{\mathrm{t}} = \overleftarrow{\mathrm{t}} = \mathrm{t}$, i.e. that both forward and backward transition densities (and corresponding operators) take the same form and so that the chain at stationarity is *reversible*, in which case have that

$$\mathrm{t}(\boldsymbol{x}' \mid \boldsymbol{x})\, p(\boldsymbol{x}) = \mathrm{t}(\boldsymbol{x} \mid \boldsymbol{x}')\, p(\boldsymbol{x}') \quad \forall \boldsymbol{x} \in X,\, \boldsymbol{x}' \in X. \tag{2.31}$$

This is often termed the *detailed balance* condition. Importantly both the detailed balance (2.31) and *generalised balance* (2.29) conditions can also be written in terms of the unormalised density $\tilde{p}$ by multiplying both sides by Z, and so can be checked even when Z is unknown.

The restriction to reversible transition operators in detailed balance, while sufficient for (2.27) to hold is not necessary. Markov chains which satisfy the generalised balance condition but not detailed balance are termed *non-reversible*, and there are theoretical results suggesting that non-reversible Markov chains can sometimes achieve significantly improved convergence compared to related reversible chains [14, 29, 43]. While there are several general purpose frameworks for specifying reversible transition operators which leave a target distribution invariant, developing methods for constructing irreversible transition operators with a desired invariant distribution has proven more challenging. The approaches proposed to date are generally limited in practice to special cases such as finite state spaces [59, 60, 63] or chains with tractable invariant distributions such as multivariate normal [6].

Nonetheless non-reversible Markov chains are still commonly used in MCMC applications. Given a set of transition operators which each individually leave a target distribution invariant, the sequential composition of the transition operators will necessarily by induction also leave the target distribution invariant. Even if the individual transition operators are all reversible, the overall sequential composition will generally not be (instead having an adjoint 'backward' operator corresponding to applying the individual transitions in the reversed order). Sequentially combining several reversible transition operators is common in MCMC implementations, though this is more often the result of each individual operator not meaning the requirements for ergodicity in isolation and so needing to be combined with other operators, rather than due to a specific aim of introducing irreversibility.

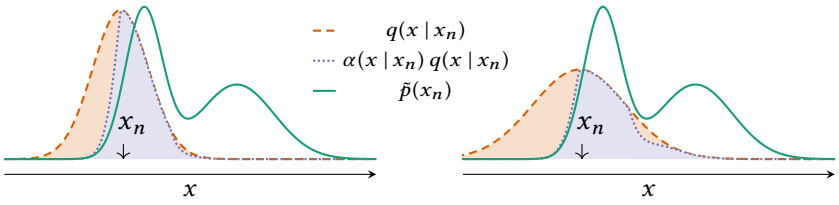


Figure 2.7: Visualisation of Metropolis–Hastings algorithm in a univariate target density. The green curves shows the unnormalised target density. The arrows indicate the current chain state. The orange curves show the density of proposed moves from this state, with the left axis using a narrower proposal than the right. The violet curves show the proposal density scaled by the acceptance probability of the proposed move, this reducing the probability of transitions to states with lower density than the current state. The orange region between the violet and orange curves represents the probability mass reallocated to rejections by the downscaling by the acceptance function. The broader proposal in the right axis has an increased probability of making a move to the other mode in the target density but at a cost of an increased rejection probability.

Having now introduced the key theory underlying MCMC methods, we will now discuss practical implementations of the approach. In the following sub-sections we will review two of the most popular frameworks for constructing reversible transition operators which leave a target distribution invariant: the *Metropolis–Hastings* algorithm and *Gibbs sampling*.

2.2.1 Metropolis–Hastings

Although the algorithm has come to be commonly known by Edward Metropolis' name as first author on the 1953 paper [38], it is believed that Arianna and Marshall Rosenbluth, two of the other co-authors, were the main contributors to the development of the algorithm [24].

The seminal *Metropolis–Hastings* algorithm provides a general framework for constructing Markov chains with a desired invariant distribution and is ubitiqous in MCMC methodology. The original Rosenbluth–Teller–Metropolis variant of the algorithm [38] dates to the very beginnings of the Monte Carlo method, having being first implemented on Los Alamos' MANIAC[5] one of the earliest programmable computers. The method was generalised in a key paper by Hastings [27], and the optimality among several competing alternatives of the form now used demonstrated by Peskun [49]. An extension to Markov chains on trans-dimensional spaces was proposed by Green [23].

An outline of the method is given in Algorithm 2 and a visualisation of its application to a univariate target distribution shown in Figure 2.7.

5 *Mathematical Analyzer, Numerical Integrator and Computer.*

Algorithm 2 Metropolis–Hastings transition.

Input: $\boldsymbol{x}_n$: current chain state, $\tilde{p}$: unnormalised target density,
q : normalised proposal density which we can sample according to.
Output: $\boldsymbol{x}_{n+1}$: next chain state with $\boldsymbol{x}_n \sim p(\cdot) \implies \boldsymbol{x}_{n+1} \sim p(\cdot)$.

1: $\boldsymbol{x}^* \sim q(\cdot \mid \boldsymbol{x}_n)$ ▷ Generate proposed new state
2: $u \sim \mathcal{U}(\cdot \mid 0, 1)$
3: **if** $u < \frac{\tilde{p}(\boldsymbol{x}^*)\, q(\boldsymbol{x}_n \mid \boldsymbol{x}^*)}{\tilde{p}(\boldsymbol{x})\, q(\boldsymbol{x}^* \mid \boldsymbol{x}_n)}$ **then**
4: $\quad \boldsymbol{x}_{n+1} \leftarrow \boldsymbol{x}^*$ ▷ Proposed move accepted
5: **else**
6: $\quad \boldsymbol{x}_{n+1} \leftarrow \boldsymbol{x}_n$ ▷ Proposed move rejected

The key idea is to propose updates to the state using an arbitrary transition operator and then correct for this transition operator not necessarily leaving the target distribution invariant by stochastically accepting or rejecting the proposal. If a proposal is rejected the chain remains at the current state, otherwise the chain state takes on the proposed value. The transition density corresponding to Algorithm 2 is

$$\begin{aligned} \mathrm{t}(\boldsymbol{x}' \mid \boldsymbol{x}) = \alpha(\boldsymbol{x}' \mid \boldsymbol{x})\, q(\boldsymbol{x}' \mid \boldsymbol{x}) + \\ \left(1 - \int_X \alpha(\boldsymbol{x}^* \mid \boldsymbol{x})\, q(\boldsymbol{x}^* \mid \boldsymbol{x})\, \mathrm{d}\mu(\boldsymbol{x}^*)\right) \delta(\boldsymbol{x}' - \boldsymbol{x}), \end{aligned} \tag{2.32}$$

with the *acceptance probability* $\alpha : X \times X \to [0, 1]$ defined as

$$\alpha(\boldsymbol{x}' \mid \boldsymbol{x}) = \min\left\{1, \frac{q(\boldsymbol{x} \mid \boldsymbol{x}')\, p(\boldsymbol{x}')}{q(\boldsymbol{x}' \mid \boldsymbol{x})\, p(\boldsymbol{x})}\right\} = \min\left\{1, \frac{q(\boldsymbol{x} \mid \boldsymbol{x}')\, \tilde{p}(\boldsymbol{x}')}{q(\boldsymbol{x}' \mid \boldsymbol{x})\, \tilde{p}(\boldsymbol{x})}\right\}, \tag{2.33}$$

and $q : X \times X \to [0, \infty)$ the *proposal density*.

The original Rosenbluth–Teller–Metropolis algorithm used a symmetric proposal density $q(\boldsymbol{x}' \mid \boldsymbol{x}) = q(\boldsymbol{x} \mid \boldsymbol{x}')\ \forall \boldsymbol{x} \in X, \boldsymbol{x}' \in X$ (with the extension to the non-symmetric case being due to Hastings [27]), in which case the acceptance probability definition simplifies to

$$\alpha(\boldsymbol{x}' \mid \boldsymbol{x}) = \min\left\{1, \frac{p(\boldsymbol{x}')}{p(\boldsymbol{x})}\right\} = \min\left\{1, \frac{\tilde{p}(\boldsymbol{x}')}{\tilde{p}(\boldsymbol{x})}\right\}. \tag{2.34}$$

Note that in both (2.33) and (2.34) the target density only appears as a ratio and so only need be known up to a constant.

For the purposes of verifying the detailed balance condition (2.31), the density of *self-transitions*, i.e. a transition to the same state, can be ignored as (2.31) is trivially satisfied for $\boldsymbol{x}' = \boldsymbol{x}$. Considering therefore the cases $\boldsymbol{x} \neq \boldsymbol{x}'$ where the Dirac delta term representing the singular

component corresponding to rejected proposals can be neglected, we have $\forall \boldsymbol{x} \in X,\, \boldsymbol{x}' \in X : \boldsymbol{x} \neq \boldsymbol{x}$

$$\begin{aligned}
\mathrm{t}(\boldsymbol{x}' \mid \boldsymbol{x})\, p(\boldsymbol{x}) &= \min\left\{1,\, \frac{q(\boldsymbol{x} \mid \boldsymbol{x}')\, p(\boldsymbol{x}')}{q(\boldsymbol{x}' \mid \boldsymbol{x})\, p(\boldsymbol{x})}\right\} q(\boldsymbol{x}' \mid \boldsymbol{x})\, p(\boldsymbol{x}) && (2.35)\\
&= \min\{q(\boldsymbol{x}' \mid \boldsymbol{x})\, p(\boldsymbol{x}),\, q(\boldsymbol{x} \mid \boldsymbol{x}')\, p(\boldsymbol{x}')\} && (2.36)\\
&= \min\left\{\frac{q(\boldsymbol{x}' \mid \boldsymbol{x})\, p(\boldsymbol{x})}{q(\boldsymbol{x} \mid \boldsymbol{x}')\, p(\boldsymbol{x}')},\, 1\right\} q(\boldsymbol{x} \mid \boldsymbol{x}')\, p(\boldsymbol{x}') && (2.37)\\
&= \mathrm{t}(\boldsymbol{x} \mid \boldsymbol{x}')\, p(\boldsymbol{x}'). && (2.38)
\end{aligned}$$

Therefore the detailed balance condition is satisfied, and the Metropolis–Hastings transition operator leaves the target distribution P invariant.

An important special case for chains on a Euclidean state space $X = \mathbb{R}^D$, is when the proposal transition operator is deterministic and corresponds to a differentiable involution of the current state. Let $\boldsymbol{\phi} : X \to X$ be an involution, i.e. $\boldsymbol{\phi} \circ \boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{x}\ \forall X$ with Jacobian determinant $D_{\boldsymbol{\phi}}(\boldsymbol{x}) = \left|\frac{\partial \boldsymbol{\phi}(\boldsymbol{x})}{\partial \boldsymbol{x}}\right|$ which is defined and non-zero P-almost everywhere. Then if we define a transition operator via the transition density

$$\begin{aligned}
\mathrm{t}(\boldsymbol{x}' \mid \boldsymbol{x}) &= \delta(\boldsymbol{x}' - \boldsymbol{\phi}(\boldsymbol{x}))\alpha(\boldsymbol{x}) + \delta(\boldsymbol{x}' - \boldsymbol{x})(1 - \alpha(\boldsymbol{x})),\\
\alpha(\boldsymbol{x}) &= \min\left\{1,\, \frac{p \circ \boldsymbol{\phi}(\boldsymbol{x})}{p(\boldsymbol{x})} D_{\boldsymbol{\phi}}(\boldsymbol{x})\right\},
\end{aligned} \tag{2.39}$$

then this transition operator will leave the target distribution P invariant. This deterministic transition operator variant is as a special case of the trans-dimensional Metropolis–Hastings extension introduced by Green [20, 23]. To generate from this transition operator from a current state $\boldsymbol{x}$ we compute the proposed move $\boldsymbol{\phi}(\boldsymbol{x})$ and accept the move with probablity $\alpha(\boldsymbol{x})$. We can demonstrate that this transition operator leaves P invariant by directly verifying (2.27)

$$\begin{aligned}
&\int_X \mathrm{t}(\boldsymbol{x}' \mid \boldsymbol{x})\, p(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} && (2.40)\\
&= \int_X \delta(\boldsymbol{x}' - \boldsymbol{\phi}(\boldsymbol{x}))\, \alpha(\boldsymbol{x})\, p(\boldsymbol{x}) + \delta(\boldsymbol{x}' - \boldsymbol{x})(1 - \alpha(\boldsymbol{x}))\, p(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} && (2.41)\\
&= \int_X \delta(\boldsymbol{x}' - \boldsymbol{y})\, \alpha \circ \boldsymbol{\phi}(\boldsymbol{y})\, p \circ \boldsymbol{\phi}(\boldsymbol{y})\, D_{\boldsymbol{\phi}}(\boldsymbol{y})\, \mathrm{d}\boldsymbol{y} + (1 - \alpha(\boldsymbol{x}'))\, p(\boldsymbol{x}') && (2.42)\\
&= p(\boldsymbol{x}') + \alpha \circ \boldsymbol{\phi}(\boldsymbol{x}')\, p \circ \boldsymbol{\phi}(\boldsymbol{x}')\, D_{\boldsymbol{\phi}}(\boldsymbol{x}') - \alpha(\boldsymbol{x}')\, p(\boldsymbol{x}'). && (2.43)
\end{aligned}$$

In going from (2.42) to (2.43) we use a change of variables $\boldsymbol{y} = \boldsymbol{\phi}(\boldsymbol{x})$ in the integral. As $\boldsymbol{\phi}$ is an involution we have that $\boldsymbol{\phi} \circ \boldsymbol{\phi}(\boldsymbol{x}') = \boldsymbol{x}'$ and $D_{\boldsymbol{\phi}} \circ \boldsymbol{\phi}(\boldsymbol{x}') = D_{\boldsymbol{\phi}}(\boldsymbol{x}')^{-1}$ and so

$$\alpha \circ \boldsymbol{\phi}(\boldsymbol{x}')p \circ \boldsymbol{\phi}(\boldsymbol{x}')D_{\boldsymbol{\phi}}(\boldsymbol{x}') = \min\left\{p \circ \boldsymbol{\phi}(\boldsymbol{x}')D_{\boldsymbol{\phi}}(\boldsymbol{x}'), p(\boldsymbol{x}')\right\} = \alpha(\boldsymbol{x}')p(\boldsymbol{x}').$$

The last two terms in (2.43) therefore cancel and so (2.27) is satisfied by the transition operator defined by (2.39).

Although this transition operator leaves the target distribution P invariant, it is clear that it will not generate an ergodic Markov chain. Starting from a point $\boldsymbol{x}$ the next chain state will be either $\boldsymbol{\phi}(\boldsymbol{x})$ if the proposed move is accepted or $\boldsymbol{x}$ if rejected. In the former case the next proposed move will be to $\boldsymbol{\phi} \circ \boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{x}$ i.e. back to the original state. Therefore the chain will visit a maximum of two states. However as noted previously we can sequentially compose individual transition operators which all leave a target distribution invariant. Therefore a deterministic proposal Metropolis–Hastings transition can be combined with other transition operators to ensure the chain is irreducible and aperiodic.

In general for a Metropolis–Hastings transition operator to be irreducible, it is necessary that the proposal operator is irreducible [62], however this is not sufficient. For a target density which is positive everywhere on $X = \mathbb{R}^D$, then a sufficient but not necessary condition for irreducibility is that the proposal density is positive everywhere [54]. If the set of points with a non-zero probability of rejection has non-zero P-measure, then the transition operator is aperiodic [62].

A common choice of proposal density when the target distribution is defined on $\mathbb{R}^D$ is a multivariate normal density centred at the current state i.e. $q(\boldsymbol{x}' \mid \boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}' \mid \boldsymbol{x}, \Sigma)$ which satisfies the positivity condition for irreducibility. In general we would achieve optimal performance with a proposal density covariance Σ which is proportional to the covariance of the target distribution [57]. In practice we do not have access to the true covariance and so typically an isotropic proposal density is used with covariance $\Sigma = \sigma^2\mathbf{I}$ controlled by a single scale parameter σ, often termed the *step size* or *proposal width*. This proposal density is symmetric so the simplified acceptance rule (2.34) can be used, further the proposal density depends only on the difference $\boldsymbol{x}' - \boldsymbol{x}$ with Metropolis–Hastings methods having these properties often termed *random-walk Metropolis*.

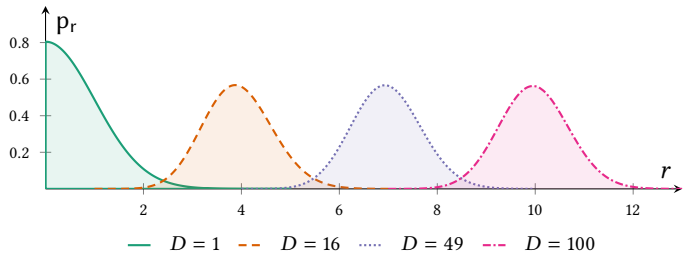


Figure 2.8: Illustration of concentration of measure in a multivariate normal distribution. The plots shows the probability density of the distance from the origin $\mathrm{r} = \|\mathbf{x}\|_2$ of a D-dimensional multivariate normal random vector $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ for different dimensionalities D. As the dimension increases most of the mass concentrates away from the origin around a spherical shell of radius $\sqrt{D}$. For a multivariate normal random vector with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ this generalises to the mass being mainly in an ellipsoidal shell aligned with the eigenvectors of $\boldsymbol{\Sigma}$ and centred at $\boldsymbol{\mu}$.

Random walk Metropolis methods have been extensively theoretically studied, with sufficient conditions known in some cases to ensure geometric ergodicity of a chain [37, 56] though these can be hard to verify in practical problems. There has also been much work on practical guidelines and methods for tuning the free parameters in the algorithm, including approaches for tuning the step-size using acceptance rates [17, 53] and adaptive variants which automatically estimate a non-isotropic proposal covariance [25, 57].

In general the choice of proposal density will be key in determining the efficiency of Metropolis–Hastings MCMC methods. Ideally we want to be able to propose large moves in the state space to reduce the dependencies between successive chain states and so increase the number of effective samples, however this needs to be balanced with maintaining a reasonable acceptance probability with large proposed moves often having a low acceptance probability. Figure 2.7 gives an illustration of this tradeoff in a one-dimensional example.

In high-dimensional spaces this issue is much more severe due to the phenomenon of *concentration of measure*: in probability distributions defined on high-dimensional spaces most of the probability mass will tend to be concentrated into a 'small' subset of the space [2, 34]. An illustration of this phenomenon for the multivariate normal distribution is shown in Figure 2.8, where the mass in high dimensions is mostly located in a thin ellipsoidal shell. The region where most of the mass concentrates, termed the *typical set* of the distribution, will for the tar-

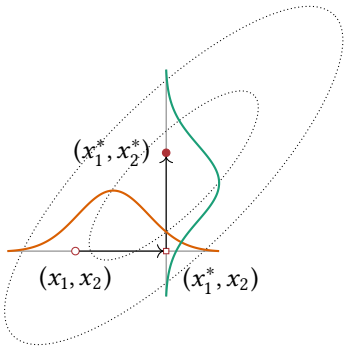


Figure 2.9: Schematic of Gibbs sampling transition in a bivariate normal target distribution (ellipses indicate constant density contours). Given an initial state $\mathbf{x} = (x_1, x_2)$, the x_1 (horizontal) co-ordinate is first updated by independently sampling from the normal conditional $p_{x_1|x_2}(\cdot \mid x_2)$, represented by the orange curve. The new partially updated state is then $\mathbf{x} = (x_1^*, x_2)$. The second x_2 (vertical) co-ordinate is then independently resampled from the normal conditional $p_{x_2|x_1}(\cdot \mid x_1^*)$, shown by the green curve. The final updated state is then $\mathbf{x} = (x_1^*, x_2^*)$.

Algorithm 3 Sequential scan Gibbs transition.

Input: $\boldsymbol{x}_n$: current chain state, I : ordered set of indices of all individual variables in chain state, $\{p_i\}_{i \in I}$: set of complete conditionals of target density p which can all be sampled from.

Output: $\boldsymbol{x}_{n+1}$: next chain state with $\boldsymbol{x}_n \sim p(\cdot) \implies \boldsymbol{x}_{n+1} \sim p(\cdot)$.

1: $\boldsymbol{x} \leftarrow \boldsymbol{x}_n$
2: **for** $i \in I$ **do**
3: $\quad x_i \sim p_i(\cdot \mid \boldsymbol{x}_{\setminus i})$ ▷ Resample x_i from p_i given *current* $\boldsymbol{x}_{\setminus i}$.
4: $\boldsymbol{x}_{n+1} \leftarrow \boldsymbol{x}$

get distributions of interest generally have a significantly more complex geometry. Finding proposals which can make large moves in such settings is challenging: moves in most directions will have a probability of acceptance which exponentially drops to zero as the distance away from the current state is increased and so simple proposal densities which ignore the geometry the typical set such as those used in random-walk Metropolis will need to make very small moves to have a reasonable probability of acceptance [3].

2.2.2 Gibbs sampling

Gibbs sampling [16, 18], originally proposed by Geman and Geman for image restoration using a Markov random field image model, is based on the observation that a valid transition operator for a joint target distribution across many variables, is one which updates only a subset of the variables and leaves the conditional distribution on that subset

given the rest invariant. Although if used in isolation a transition operator which only updates some components of the state will not give an ergodic chain, as discussed previously multiple transition operators can be combined together to achieve ergodicity.

More specifically the original formulation of Gibbs sampling defines a Markov chain by sequentially independently resampling each individual variable in the model from its conditional distribution given the current values of the remaining variables. If I is an index set over the individual variables in the vector target state $\mathbf{x}$, then for each $i \in I$ we partition the state $\mathbf{x}$ into the i^{th} variable x_i and a vector containing all the remaining variables $\mathbf{x}_{\setminus i}$. For each $i \in I$ the target density can be factorised in to the marginal density $p_{\setminus i}$ on $\mathbf{x}_{\setminus i}$ and conditional density p_i on x_i given $\mathbf{x}_{\setminus i}$, i.e.

$$p(\boldsymbol{x}) = p_i(x_i \mid \boldsymbol{x}_{\setminus i})\, p_{\setminus i}(\boldsymbol{x}_{\setminus i}), \tag{2.44}$$

with the conditional densities $\{p_i\}_{i \in I}$ termed the *complete conditionals* of the target density. If each of these complete conditionals corresponds to a distribution we can generate samples from (for example using a transform method or rejection sampling) then we can apply the sequential Gibbs sampling transition operator defined in Algorithm 3 and visualised for a bivariate example in Figure 2.9.

The sequential Gibbs transition is irreducible and aperiodic under mild conditions [10, 55]. Rather than using a deterministic sequential scan through the variables, an alternative is to randomly sample without replacement the variable to update on each iteration; unlike the sequential scan version this defines a reversible transition operator. The random update variant is more amenable to theoretical analysis, however in practice the ease of implementation of the sequential scan variant and computational benefits in terms of memory access locality mean it seems to be more often used in practice [28]. A compromise between the completely random updates and a sequential scan is to randomly permute the update order after each complete scan.

A apparent advantage of Gibbs sampling over Metropolis–Hastings is the lack of a proposal density which needs to be tuned. This is has helped popularise 'black-box' implementations of Gibbs sampling such as the probabilistic modelling packages BUGS [21] and JAGS [50]. A well-known issue with Gibbs sampling however is that its performance

is highly dependent on the parameterisation used for the target density [52], with strong correlations between variables leading to large dependencies between successive states and slow convergence to stationarity. This can be alleviated in some cases by using a suitable reparameterisation to reduce dependencies between variables, however this restores the difficulty of tuning free parameters.

The updates do not necessarily need to be performed by sampling from complete conditionals of single variables - in some cases the complete conditional of a vector variables has a tractable form which can be sampled from as a 'block'; this motivates the name *block Gibbs samling* for such variants. By accounting for the dependencies between the variables in a block this can help alleviate some of the issues with highly correlated targets where applicable.

Compound terms such as *Metropolis-within-Gibbs* are sometimes used to refer to methods which sequentially apply Metropolis transition operators which each update only a subset of variables in the target distribution. We will however consider the defining feature of Gibbs sampling as being exact sampling from one or more conditionals rather than sequentially applying transition operators which update only subsets of variables and so will only refer to 'Gibbs sampling' in that context.

2.3 AUXILIARY VARIABLE METHODS

Although Gibbs sampling and random-walk Metropolis are commonly used in practice, as discussed above both have drawbacks when applied to complex high-dimensional target distributions. One approach which has proven particularly successful for constructing alternative Markov transition operators which can overcome some of these shortcomings is the introduction of *auxiliary variables* in to the chain state. For concreteness of notation in the following discussion we let the variables of interest, which we term the target variables, be represented by the random vector $\mathbf{x} \in X$ and the introduced auxiliary variables by the random vector $\mathbf{a} \in A$. We assume for generality here multiple auxiliary variables are introduced, however methods using a single scalar auxiliary variable are a common special case.

One way of defining a joint distribution across the target and auxiliary variables is to specify an arbitrary conditional distribution $\mathrm{P}_{\mathbf{a}|\mathbf{x}}$

and choose the marginal distribution $\mathsf{P}_{\mathbf{x}}$ to be equal to the target distribution P. Given samples from a joint distribution we can estimate expectations with respect to the marginal distribution on a subset of the variables by simply ignoring the dimensions of the sampled state we wish to marginalise over. Therefore if we can construct a Markov chain with the resulting joint distribution $\mathsf{P}_{\mathbf{x},\mathbf{a}}$ as its unique invariant distribution, then we can use components of the sampled states corresponding to the target variables to estimate expectations with respect to the target distribution. We will consider two MCMC methods apply this approach, *slice sampling* and *Hamiltonian Monte Carlo*, in the follow subsections.

An alternative approach is to instead construct a joint distribution on the target and auxiliary variables such that the conditional distribution $\mathsf{P}_{\mathbf{x}|\mathbf{a}}$ is equal to the target distribution P across some set of values $A^* \subset A$ of the auxiliary variables, which can be expressed in terms of the density $\mathsf{p}_{\mathbf{x}|\mathbf{a}}$ as

$$\mathsf{p}_{\mathbf{x}|\mathbf{a}}(\boldsymbol{x} \mid \boldsymbol{a}) = p(\boldsymbol{x}) \qquad \forall \boldsymbol{x} \in X, \boldsymbol{a} \in A^*. \tag{2.45}$$

If the resulting marginal probability $\mathsf{P}_{\mathbf{a}}(A^*)$ is non-zero, then the sampled states $\{\boldsymbol{x}^{(s)}, \boldsymbol{a}^{(s)}\}_{s=1}^{S}$ of a Markov chain which has $\mathsf{P}_{\mathbf{x},\mathbf{a}}$ as its unique invariant distribution can be used to estimate expectations with respect to the target distribution by computing averages over only the sampled target variable values $\boldsymbol{x}^{(s)}$ for which the corresponding auxiliary variables $\boldsymbol{a}^{(s)}$ take values in A^* (these being at convergence samples from $\mathsf{P}_{\boldsymbol{x}|\boldsymbol{a}}(\cdot \mid \boldsymbol{a})$ and so the target distribution), i.e.

$$\int_X f(\boldsymbol{x}) \,\mathrm{d}P(\boldsymbol{x}) = \lim_{S \to \infty} \frac{\sum_{s=1}^{S} \mathbb{1}_{A^*}(\boldsymbol{a}^{(s)}) f(\boldsymbol{x}^{(s)})}{\sum_{s=1}^{S} \mathbb{1}_{A^*}(\boldsymbol{a}^{(s)})}. \tag{2.46}$$

We will discuss *simulated tempering*, an MCMC method which introduces an auxiliary variable in this manner in a subsequent subsection.

An issue with this approach is that if $\mathsf{P}_{\mathbf{a}}(A^*)$ is small, the number of sampled states with $\boldsymbol{a} \in A^*$ may be very small or even zero. This can require a large number of samples S for the sufficient samples with auxiliary variables in the required set A^* to be generated to allow the MCMC estimates computed using (2.46) to be reliable.

The estimator in (2.46) has a close resemblance to the formulation of the Monte Carlo estimator corresponding to rejection sampling given

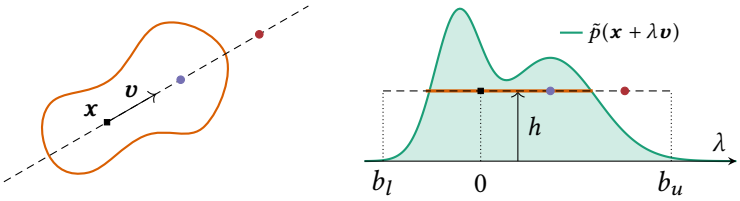


Figure 2.10: Schematic of linear slice sampling, showing 'plan' (left) and 'cross-sectional' (right) views of a bivariate target density. Orange curve (left) and line (right) indicates a constant density slice S_h. The black square indicates current target state value $\boldsymbol{x}$ and the dashed line is *slice line*, the one-dimensional linear sub-space aligned with the vector $\boldsymbol{v}$ which a new value from the state will be sampled on. The extents of the dashed line segment represent the initial bracket new proposed states will be drawn from. Points are proposed on the slice line by drawing a value uniformly from the current bracket. The red circle represents an initial proposed point which is not in the slice and so the right bracket edge is shrunk to this point. The violet circle shows a second sampled point from the new reduced bracket, this point within the slice and so returned as the updated target state.

in (2.16), with averages computed over the subset of samples meeting an 'acceptance' criteria. As noted previously rejection sampling can in fact be considered as an auxiliary variable method, with binary accept indicator variables $\mathrm{a} \in \{0, 1\}$ introduced such that the conditional density $\mathrm{p}_{\mathbf{x}|\mathrm{a}}(\boldsymbol{x} \,|\, 1)$ is equal to the target density $p(\boldsymbol{x})$ as shown in (2.12), i.e. exactly corresponding to the property in (2.45). Rejection sampling can therefore be seen to be another example of this construct, though in this case each pair of target – auxiliary variable samples are generated independently rather than by constructing a Markov chain.

When discussing the rejection sampling estimator (2.16) we saw there was a close link to importance sampling estimator (2.17), with the importance sampling estimator having the potential advantage however of using all of the generated samples in computing estimates. In Chapter 5 we will discuss a related alternative approach to constructing estimators for auxiliary variable methods based on conditioning like simulated tempering, which unlike the estimator in (2.46) allows using all of the samples in a Markov chain to compute estimates.

2.3.1 Slice sampling

Slice sampling is a family of auxiliary variable MCMC methods which exploit the same observation as used to motivate rejection sampling - to sample from a target distribution it is sufficient to uniformly sample

Algorithm 4 Linear slice sampling transition.

Input: $\boldsymbol{x}_n$: current chain state, $\tilde{p}$: unnormalised target density,
q : slice vector density, M : maximum number of step out iterations.
Output: $\boldsymbol{x}_{n+1}$: next chain state with $\boldsymbol{x}_n \sim p(\cdot) \implies \boldsymbol{x}_{n+1} \sim p(\cdot)$.

$h \sim \mathcal{U}(\cdot \mid 0, \tilde{p}(\boldsymbol{x}_n))$ ▷ Sample slice height
$\boldsymbol{v} \sim q(\cdot)$ ▷ Sample vector setting slice line and initial bracket width
$b_u \sim \mathcal{U}(\cdot \mid 0, 1)$ ▷ Uniformly sample bracket around current state
$b_l \leftarrow b_u - 1$
if $M > 0$ **then** $(b_l,\, b_u) \leftarrow$ LinearStepOut$(\boldsymbol{x}_n,\, b_l,\, b_u,\, M)$
$\lambda \sim \mathcal{U}(\cdot \mid b_l, b_u)$
while True **do**
    $\boldsymbol{x}^* \leftarrow \boldsymbol{x}_n + \lambda \boldsymbol{v}$ ▷ Update proposed state
    if $\tilde{p}(\boldsymbol{x}^*) \leq h$ **then** ▷ Proposed point not on slice
        if $\lambda < 0$ **then** $b_l \leftarrow \lambda$ **else** $b_u \leftarrow \lambda$ ▷ Shrink slice bracket
        $\lambda \sim \mathcal{U}(\cdot \mid b_l, b_u)$ ▷ Sample uniformly from new bracket
    else ▷ Proposed state on slice
        return $\boldsymbol{x}^*$
function LinearStepOut$(\boldsymbol{x}_n,\, b_l,\, b_u,\, M)$
    $L \sim \text{UniformInt}(\cdot \mid 0, M)$ ▷ Sample integer uniformly from $[0, M]$
    $U \leftarrow M - L$
    while $L > 0$ **and** $\tilde{p}(\boldsymbol{x}_n + b_l \boldsymbol{v}) > h$ **do** ▷ Step out lower bracket edge
        $b_l \leftarrow b_l - 1$
        $L \leftarrow L - 1$
    while $U > 0$ **and** $\tilde{p}(\boldsymbol{x}_n + b_u \boldsymbol{v}) > h$ **do** ▷ Step out upper bracket edge
        $b_u \leftarrow b_u + 1$
        $U \leftarrow U - 1$
    return b_l, b_u

from the volume beneath a graph of the target density function. Rather than generate independent points from this volume as in rejection sampling, slice sampling instead constructs a transition operator which leaves the uniform distribution on this volume invariant.

The method we will concentrate on here was proposed by Neal [41, 42]. A related algorithm which uses per data-point auxiliary variables in Bayesian inference problems developed by Damien, Wakefield and Walker [13]. Murray, Adams and Mackay later proposed *elliptical slice sampling* [40], an extension of Neal's slice sampling method which is particularly effective for target distributions which are well approximated by a multivariate normal and which we will discuss at the end of this subsection.

Slice sampling defines a Markov chain on an augmented state space by introducing an auxiliary *height* variable $\mathsf{h} \in [0, \infty)$ in addition to the original target state $\mathbf{x} \in X$. The conditional density on the height variable is $\mathrm{p}_{\mathsf{h}|\mathbf{x}}(h \mid \mathbf{x}) = \mathcal{U}(h \mid 0, \tilde{p}(\boldsymbol{x})) = \frac{1}{\tilde{p}(\boldsymbol{x})} \mathbb{1}_{[0, \tilde{p}(\boldsymbol{x}))}(h)$, i.e. uniform

over the interval between zero and the unnormalised target density value. The joint density on the augmented space is then

$$\mathrm{p}_{\mathbf{x},\mathsf{h}}(\boldsymbol{x}, h) = \frac{1}{\tilde{p}(\boldsymbol{x})} \mathbb{1}_{[0,\tilde{p}(\boldsymbol{x}))}(h) \frac{\tilde{p}(\boldsymbol{x})}{Z} = \frac{1}{Z} \mathbb{1}_{[0,\tilde{p}(\boldsymbol{x}))}(h). \tag{2.47}$$

Marginalising (2.47) over $\mathbf{x}$ recovers the target density i.e. $\mathrm{p}_{\mathbf{x}} = p$.

The overall slice sampling transition is formed of the sequential composition of a transition operator which updates h given $\mathbf{x}$ and a second operator which updates $\mathbf{x}$ given h, each leaving the distributions corresponding to the conditional densities $\mathrm{p}_{\mathsf{h}|\mathbf{x}}$ and $\mathrm{p}_{\mathbf{x}|\mathsf{h}}$ respectively invariant, and so by the same argument as for Gibbs sampling the overall transition leaving the target distribution invariant. By construction the conditional density $\mathrm{p}_{\mathsf{h}|\mathbf{x}}$ is a simple uniform density and so the first transition operator is a Gibbs sampling type update in which the height variable is independently resampled from $\mathcal{U}(0, \tilde{p}(\boldsymbol{x}))$, where $\boldsymbol{x}$ is the current value of the target state $\mathbf{x}$.

The conditional density $\mathrm{p}_{\mathbf{x}|\mathsf{h}}(\boldsymbol{x} \mid h)$ is also locally uniform, equal to a positive constant whenever $\tilde{p}(\boldsymbol{x}) > h$ and zero elsewhere. However we can only evaluate the density up to an unknown constant as we cannot compute the measure of the set $S_h = \{\boldsymbol{x} \in X : \tilde{p}(\boldsymbol{x}) > h\}$ that the density is non-zero over. In general S_h, which is the eponymous *slice* of slice sampling (so called as it represents a slice through the volume under the density curve at a fixed height h), will have a complex geometry including potentially consisting of several disconnected components in the case of multi-modal densities. The complexity of the slices generally prevents us therefore from being able to independently sample a new value for $\mathbf{x}$ uniformly from S_h and so we cannot use a full Gibbs sampling scheme corresponding to sequentially independently sampling from $\mathrm{p}_{\mathsf{h}|\mathbf{x}}$ and $\mathrm{p}_{\mathbf{x}|\mathsf{h}}$.

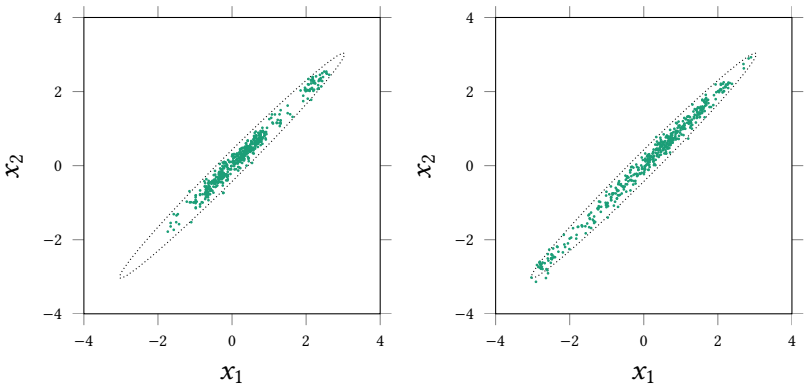
A key contribution of [42] was to introduce an elegant method for constructing a transition operator which leaves $\mathrm{p}_{\mathbf{x}|\mathsf{h}}$ invariant. In particular the method proposed has few free parameters to tune, has an efficiency which is relatively robust to the choices of the free choices that are introduced, and will for smooth target densities always move the target state by some amount (in contrast to the potential for rejections in Metropolis–Hastings methods). This method is summarised in Algorithm 4 and a visualisation of the process shown in Figure 2.10.

An important first step in the algorithm is reducing the problem of generating a point uniformly on the multidimensional slice S_h to making a move on a one-dimensional linear subspace of this slice (motivating our naming of this algorithm *linear slice sampling*) which includes the current $\mathbf{x}$ state. In the original description of the algorithm in [42] the one-dimensional subspace is chosen to be axis-aligned, correponding to updating a single component of the target state.

In this case the restriction of the slice to the one-dimensional subspace is entirely specified by the conditional density on the chosen variable component given the current values of the remaining components in the state. Slice sampling transitions for each variable in the target state can then be applied sequentially akin to Gibb sampling, but with the advantage over Gibbs of not requiring the complete conditionals to be of a tractable form which we can generate exact samples from. If conditional independency structure in the target density means the complete conditionals depend only on local subsets of variables in the target state using updates of this form has the advantage of exploiting this locality. As with Gibbs sampling however applying slice sampling in this manner makes performance strongly dependent on the parameterisation of the target density, with large magnitude correlations likely to lead to slow exploration of the space.

In [42] various specifically multivariate extensions of the algorithm are suggested which could help counter this issue, however they add significant implementation complexity compared to the basic algorithm. A simpler alternative is to define the one-dimensional subspace as being the line defined by a randomly chosen vector and passing through the current value of $\mathbf{x}$. If this vector is generated independently of the current state this is sufficient to ensure the overall transition retains the correct invariant distribution.

If little is known about the target distribution a reasonable default choice is to sample a unit vector of the required dimensionality by generating a random zero-mean isotropic covariance multivariate normal vector and then scaling it to unit norm; if an approximate covariance matrix $\hat{\Sigma}$ is known for the target density then instead generating the vector from $\mathcal{N}(\mathbf{0}, \Sigma)$ prior to normalising might be a better choice (as it favours moves aligned with the principle eigenvectors of Σ) however in this case elliptical slice sampling, which we will discuss shortly, will often be a better choice.



(a) Axis-aligned updates (b) Random direction updates

Figure 2.11: Samples generated using (a) axis-aligned versus (b) random-direction linear slice sampling in a correlated bivariate normal distribution. In both cases 1000 transitions where performed (with random selection of axis to update on each iteration in (a)) with every second sampled state shown. the maximum number of step out iterations is $M = 4$ and the initial bracket width is fixed at $w = 1$. The dotted ellipse shows the contour of the target density which contains 0.99 of the mass. The random direction chain is able to explore the typical set of the target distribution more effectively in this case with the axis-aligned updates leading to slower diffusion along the major axis of the elliptical contour.

This random-direction slice sampling variant is discussed in comparison to elliptical slice sampling in [40]. It is also bears resemblance to the scheme proposed in [12] which uses the same auxiliary variable formulation as slice sampling, but there the random direction is chosen in $X \times [0, \infty)$ i.e. to update both $\mathbf{x}$ and h and not used with the remainder of Neal's slice sampling algorithm. An example comparison of applying axis-aligned and random-direction linear slice sampling updates to a strongly positively correlated bivariate normal target distribution is shown in Figure 2.11. In this toy example the isotropic random-direction updates are able to more effectively explore the target density.

The generation of the vector $\boldsymbol{v}$ determining the one-dimensional subspace of the slice the update is performed on is represented in Algorithm 4 by Line 2 by $\boldsymbol{v}$ being generated from a density q. As well as specifying the *direction* of the slice line, the vector $\boldsymbol{v}$ also specifies a scale along this line. In Neal's description of the algorithm this is represented by the explicit *bracket width* parameter w. Here instead we assume this parameter is implicitly defined by the Euclidean norm of the vector $\boldsymbol{v}$, through suitable choice of q this allowing for direction dependent scales and also the possibility of randomisation of the scale; as we will

see shortly however compared to for example random-walk Metropolis updates with a normal proposal, linear slice sampling is much less sensitive to the choice of scale parameters, therefore a single fixed scale will often be sufficient.

Once the slice line direction and scale has been chosen, the remainder of the algorithm can be split into two stages: selection of an initial bracket on the slice line and including the point corresponding to the current state; iteratively uniformly sampling points within the current bracket, accepting the point if it is within the slice S_h otherwise shrinking the bracket and repeating. The algorithm proposed by Neal ensures both these stages are performed reversibly such that the detailed balance condition (2.31) is maintained.

The *slice bracket* defines a contiguous interval $\lambda \in [b_l, b_u]$ on the slice line $\boldsymbol{x}^*(\lambda) = \boldsymbol{x}_n + \lambda \boldsymbol{v}$ and always includes the point $\lambda = 0$ corresponding to the current state. The initial bracket is chosen by sampling a upper bound b_u uniformly from $[0, 1]$ and then setting $b_l \leftarrow b_u - 1$; in the λ slice line coordinate system this corresponds to a bracket width of one, however in general the slice line vector $\boldsymbol{v}$ can have non-unit length and so defines the initial bracket width in the target variable space. Randomising the positioning of the current state within the bracket ensures reversibility as the resulting bracket would have an equal probability (density) of being selected from any other point in the bracket (which the final accepted point will be within).

In general only a subset of the points in the current slice bracket will be within the slice S_h. As new states are proposed by sampling a point uniformly from the current bracket, the probability of such a proposal being in the slice and so accepted will be equal to the proportion of the bracket that intersects with the slice S_h. In general therefore it is desirable for the bracket to include as much of the slice as possible while not making the proportion of the bracket intersecting with the slice too small such that many points need to be proposed before one on the slice is found. The magnitude of $\boldsymbol{v}$ determines the initial bracket extents and so should ideally chosen based on any knowledge of the typical 'scale' of the target density. Often we will have little prior knowledge about such scaling however and the typical scale will often vary signifcantly across the target space, and so we may choose an initial bracket which includes only a small proportion of the intersection of the slice with the slice line.

The stepping out routine proposed by [42] and detailed in Lines 14 to 23 in Algorithm 4 is designed to counter this issue. The initial slice bracket $[b_l, b_u]$ is iteratively 'stepped-out' by incrementing / decrementing the upper / lower bracket bounds until the corresponding endpoint of the bracket lies outside the slice or a pre-determined maximum number of steps out have been performed. Ideally the step out routine will return a bracket which contains all of the intersection of the slice with slice line while not also including too great a proportion of off slice points; in general the slice may be non-convex or consist of multiple disconnected components and so the intersection of the slice line with the slice may consist of multiple disconnected intervals in which case the stepping out routine will likely only expand the slice to include a subset of these intervals. The adaptivity provided by the stepping out routine will still however generally help to make the performance of the sampler much less sensitive to the choice of the bracket scale in contrast to for example random-walk Metropolis algorithms which typically use a single fixed scale.

Analagously to the randomisation of the initial bracket positioning, in the stepping out routine if a maximum number of step out iterations M is set, the resulting step 'budget' is randomly allocated between increments of the upper bound b_u and decrements of the lower bound b_l such that final extended bracket generated by the step out routine would have an equal probability of being generated from any point within the generated bracket interval. If M is set to zero this corresponds to not performing any stepping out and simply using the initial sampled bracket; although reducing the robustness of the algorithm to the choice of the initial bracket width this option has the advantage of minimising the number of target density evaluations by not requiring additional density evaluations at the bracket endpoints during the step-out routine. An alternative 'doubling' step-out routine was also proposed in [42]. This has the advantage of exponentially expanding the slice bracket compared to the linear growth of the step-out routine described in Algorithm 4 and so can be more efficient in target distributions where the typical scales of the density varies across several orders of magnitude. The doubling procedure requires a more complex subsequent procedure for sampling points in the resulting bracket however to ensure reversibility.

Algorithm 5 Elliptical slice sampling transition.

Input: $\boldsymbol{x}_n$: current chain state, $\tilde{p}$: unnormalised target density,
$\boldsymbol{\mu}, \boldsymbol{\Sigma}$: mean and covariance of normal approximation to target.
Output: $\boldsymbol{x}_{n+1}$: next chain state with $\boldsymbol{x}_n \sim p \implies \boldsymbol{x}_{n+1} \sim p$.

$h \sim \mathcal{U}(0, \tilde{p}(\boldsymbol{x}_n) / \mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}))$ ▷ Sample slice height
$\boldsymbol{v} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ ▷ Sample vector setting slice ellipse
$\theta_u \sim \mathcal{U}(0, 2\pi)$ ▷ Uniformly sample bracket around current state
$\theta_l \leftarrow \theta_u - 2\pi$
$\theta \leftarrow \theta_u$
while TRUE **do**
    $\boldsymbol{x}^* \leftarrow (\boldsymbol{x}_n - \boldsymbol{\mu}) \cos\theta + (\boldsymbol{v} - \boldsymbol{\mu}) \sin\theta + \boldsymbol{\mu}$ ▷ Update proposed state
    if $\tilde{p}(\boldsymbol{x}^*) / \mathcal{N}(\boldsymbol{x}^* \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) \leq h$ **then** ▷ Proposed point not on slice
        if $\theta < 0$ **then** $\theta_l \leftarrow \theta$ **else** $\theta_u \leftarrow \theta$ ▷ Shrink slice bracket
        $\theta \sim \mathcal{U}(\theta_l, \theta_u)$ ▷ Sample uniformly from new bracket
    else
        return $\boldsymbol{x}^*$

Once the initial bracket has been generated and potentially stepped out, the remainder of the algorithm consists of finding a point on the slice line bracket which is within the slice S_h. This is done in an iterative manner by first sampling a point uniformly from the current bracket and checking if it is within the slice or not. If the proposed point is in the slice, the corresponding value for the target variables is returned at the new state. Otherwise the proposed point is set as the new upper or lower bound of the bracket such that the point corresponding to the current state remains within the bracket. This shrinks the bracket by removing an interval where it is known at least some regions of are not within the slice. A new point is then sampled uniformly from the new smaller bracket and the procedure repeats until an acceptable point in the slice is found.

The iterative shrinking of the slice bracket implemented by this procedure introduces a further level of adaptivity in to the slice sampling algorithm, meaning that even if only a small proportion of the initial bracket lies within the slice only relatively few iterations will be needed still till the bracket is shrunk sufficiently for there to be a high probability of proposing a point within the bracket. By ensuring the point corresponding to the current state always remains within the current bracket, reversibility is maintained.

An alternative to the linear slice sampling procedure just described, is the *elliptical slice sampling* method proposed in [40] and described in Algorithm 5. As suggested by the name, in elliptical slice sampling

rather than proposing points on a line instead an elliptical path in the target space is defined and new points proposed on this ellipse.

Elliptical slice sampling is intended for use in target distributions which are reasonably well approximated by a known multivariate normal distribution with density $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. This Gaussian density might correspond to a multivariate normal prior distribution on model latent variables where the dependence between the latent and observed variables is only weak and so the posterior remains well approximated by the prior or a Gaussian approximation fitted directly to the target distribution using an optimisation based approximate inference scheme such as those discussed in Appendix C [46].

In each elliptical slice sampling transition an auxiliary vector $\boldsymbol{v}$ is independently sampled from the distribution with density $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. If the target distribution was exactly described by the multivariate normal density we could use this independent draw directly as the new chain state (though obviously in this case there would be no advantage in formulating as an MCMC method). In reality the target distribution will only approximately described by the multivariate normal density $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and so we wish to instead use this independent draw to define a Markov transition operator that will potentially move the state to a point nearly independent of the current state, but is also able to back off to more conservative proposals closer to the current chain state. This is achieved by defining an elliptical path in target space centred at $\boldsymbol{\mu}$, passing through the current chain state $\boldsymbol{x}_n$ and the auxiliary vector $\boldsymbol{v}$ and parameterised by an angular variable θ

$$\boldsymbol{x}^*(\theta) = (\boldsymbol{x}_n - \boldsymbol{\mu}) \cos\theta + (\boldsymbol{v} - \boldsymbol{\mu}) \sin\theta + \boldsymbol{\mu}. \tag{2.48}$$

If we generated θ uniformly from $\mathcal{U}(0, 2\pi)$ then the corresponding proposed transition $\boldsymbol{x}^*(\theta)$ would exactly leave a distribution with density $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ invariant. As we instead wish to leave the target distribution invariant, a slice sampling algorithm is used to find a θ which accounts for the difference between the target distribution and multivariate normal approximation. An auxiliary slice height variable h is sampled uniformly from $\mathcal{U}(0, \tilde{p}(\boldsymbol{x}_n)/\mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}))$ and used to define a slice $S_h = \{\boldsymbol{x} \in X : \tilde{p}(\boldsymbol{x}_n)/\mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) < h\}$. Similar to the linear slice sampling algorithm, a bracket $[\theta_l, \theta_u]$ on the elliptical path is randomly placed around $\theta = 0$ corresponding to the current state $\boldsymbol{x}_n$. Unlike the

Algorithm 6 Hamiltonian Monte Carlo transition.

Input: $\boldsymbol{x}_n, \boldsymbol{p}_n$: current position–momentum state pair, $\tilde{p}$: differentiable unnormalised target density,
δt : leapfrog integrator step size
Output: $\boldsymbol{x}_{n+1}$: next chain state with $\boldsymbol{x}_n \sim p \implies \boldsymbol{x}_{n+1} \sim p$.

$\boldsymbol{x}^* \leftarrow \boldsymbol{x}_n + \frac{\delta t}{2}\boldsymbol{M}^{-1}\boldsymbol{p}_n$
$\boldsymbol{p}^* \leftarrow \boldsymbol{p}_n - \delta t \nabla\phi(\boldsymbol{x}^*)$
for $s \in \{1 \dots L-1\}$ **do**
    $\boldsymbol{x}^* \leftarrow \boldsymbol{x}^* + \delta t \boldsymbol{L}^{-1}\boldsymbol{M}^{-1}\boldsymbol{p}^*$
    $\boldsymbol{p}^* \leftarrow \boldsymbol{p}^* - \delta t \nabla\phi(\boldsymbol{x}^*)$
$\boldsymbol{x}^* \leftarrow \boldsymbol{x}^* + \frac{\delta t}{2}\boldsymbol{M}^{-1}\boldsymbol{p}^*$
$u \sim \mathcal{U}(0,1)$
$\alpha \leftarrow \exp\left(\phi(\boldsymbol{x}_n) + \frac{1}{2}\boldsymbol{p}_n\boldsymbol{M}^{-1}\boldsymbol{p}_n - \phi(\boldsymbol{x}^*) - \frac{1}{2}\boldsymbol{p}^*\boldsymbol{M}^{-1}\boldsymbol{p}^*\right)$
if $u < \alpha$ **then**
    $\boldsymbol{x}_{n+1}, \boldsymbol{p}_{n+1} \leftarrow \boldsymbol{x}^*, \boldsymbol{p}^*$ ▷ Proposed move accepted
else
    $\boldsymbol{x}_{n+1}, \boldsymbol{p}_{n+1} \leftarrow \boldsymbol{x}_n, -\boldsymbol{p}_n$ ▷ Proposed move rejected
return $\boldsymbol{x}_{n+1}, \boldsymbol{p}_{n+1}$

requirement to choose a suitable initial bracket width in linear slice sampling however, we can define the initial bracket in elliptical slice sampling to include the entire elliptical path i.e. $\theta_l = \theta_u - 2\pi$; we only need to randomise the 'cut-point' defining the initial end-points of the bracket to ensure reversibility. This removes the need to choose an initial bracket width (defined by $|\boldsymbol{v}|$ in our description of the linear slice algorithm) and for any step out procedure, and so beyond choosing the multivariate normal approximation elliptical slice sampling does not have any free settings which need to be tuned.

Once the initial bracket is defined, a directly analagous iterative procedure to that used in the linear slice sampling algorithm is used to find a θ value corresponding to a point in the slice while using rejected proposed points to shrink the bracket. As with linear slice sampling, providing the target density is a smooth function and so the intersection of the elliptical path with the slice is a non-zero measure set, then the state moved to by the elliptical slice sampling transition operator will never be equal to the previous state.

2.3.2 Hamiltonian Monte Carlo

The MCMC algorithms discussed so far have required only the ability to evaluate a (unnormalised) density function for the target distribution of interest. For distributions defined on real-valued variables the target density function $\tilde{p}$ will often be differentiable - that is the gradient

$\frac{\partial \tilde{p}}{\partial \boldsymbol{x}}$ exists P-almost everywhere. In these cases it is natural to consider using the gradient to help guide updates to the state.

A particularly powerful auxiliary variable MCMC method utilising gradient information is *Hamiltonian Monte Carlo* (HMC) [15, 44]. HMC introduces auxiliary *momentum* variables in to the chain state and then uses simulated Hamiltonian dynamics trajectories in the augmented space to generate proposed updates to the momentum–target variables state pair. The simulated Hamiltonian dynamics exhibit key geometric properties that make HMC well suited to performing MCMC in complex target distributions on high-dimensional spaces, with the method often scaling better to high-dimensional target distributions than simpler methods such as random-walk Metropolis and Gibbs sampling [5].

What we refer to as Hamiltonian Monte Carlo here was introduced in [15] as Hybrid Monte Carlo. *The alternative* Hamiltonian Monte Carlo *was suggested by MacKay [34] to emphasise the important role of Hamiltonian dynamics in the method while maintaining the same acronym [4].*

Most implementations of HMC require the target density p is defined with respect to the Lebesgue measure on a Euclidean space $X = \mathbb{R}^D$. Target densities with bounded support on $\mathbb{R}^D$ add complication to the algorithm by requiring checks that proposed updates to the state remain within the support of the target distribution, however often a change of variables can be performed with a bijective transformation that maps to a density with unbounded support, for example taking a log-transform of a positive variable, with the change of variables formula (1.22) used to compute the resulting density on the transformed space. We will consider extensions of the standard HMC approach to distributions defined on implicitly-defined manifolds embedded in a Euclidean space in Chapter 4.

Rather than working directly in terms of the target density p the HMC algorithm is more naturally described in terms of a *potential energy* function $\phi : \mathbb{R}^D \to \mathbb{R}$ which is related to the target density by

$$p(\boldsymbol{x}) = \frac{1}{Z} \exp(-\phi(\boldsymbol{x})) \iff \phi(\boldsymbol{x}) = -\log p(\boldsymbol{x}) - \log Z. \tag{2.49}$$

The original *target variables* $\mathbf{x} \in \mathbb{R}^D$ are augmented with a vector of *momentum variables* $\mathbf{p} \in \mathbb{R}^D$. The conditional density on the momenta given the target variables $\mathrm{p}_{\mathbf{p}|\mathbf{x}}$ is defined in terms of a *kinetic energy* function $\tau : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ which is even in its first argument

$$\mathrm{p}_{\mathbf{p}|\mathbf{x}}(\boldsymbol{p}|\boldsymbol{x}) \propto \exp(-\tau(\boldsymbol{p} \mid \boldsymbol{x})). \tag{2.50}$$

The joint density on the momentum and target variables is then

$$\mathrm{p}_{\mathbf{x},\mathbf{p}}(\boldsymbol{x},\boldsymbol{p}) \propto \exp(-\phi(\boldsymbol{x}) - \tau(\boldsymbol{x} \mid \boldsymbol{p})) = \exp(-h(\boldsymbol{x},\boldsymbol{p})). \tag{2.51}$$

The function $h(\boldsymbol{x},\boldsymbol{p}) = \phi(\boldsymbol{x}) + \tau(\boldsymbol{p} \mid \boldsymbol{x})$ is termed the *Hamiltonian* for the system. A common simplification is for the momenta to be defined to be independent of the target variables with a marginal density

$$\mathrm{p}_{\mathbf{p}}(\boldsymbol{p}) \propto \exp(-\tau(\boldsymbol{p})). \tag{2.52}$$

In this case the Hamiltonian $h(\boldsymbol{x},\boldsymbol{p}) = \phi(\boldsymbol{x}) + \tau(\boldsymbol{p})$ is *separable* - there are no terms jointly dependent on both $\boldsymbol{x}$ and $\boldsymbol{p}$.

In classical mechanics, the Hamiltonian describes the total energy of a mechanical system, and can be used to define a *canonical Hamiltonian dynamic* via the set of *ordinary differential equations* (ODEs)

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \frac{\partial h^{\mathsf{T}}}{\partial \boldsymbol{p}}, \qquad \frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = -\frac{\partial h^{\mathsf{T}}}{\partial \boldsymbol{x}}. \tag{2.53}$$

We define the *flow map* corresponding to this dynamic as a family of mappings $\boldsymbol{\Psi}_t : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}^D \times \mathbb{R}^D$ parameterised by a time $t \in \mathbb{R}$ such that if $(\boldsymbol{x}(t), \boldsymbol{p}(t))$ is the solutions to the set of ODEs (2.53) at a time t given an initial condition $\boldsymbol{x}(0) = \boldsymbol{x}_0, \boldsymbol{p}(0) = \boldsymbol{p}_0$ then

$$(\boldsymbol{x}(t), \boldsymbol{p}(t)) = \boldsymbol{\Psi}_t(\boldsymbol{x}_0, \boldsymbol{p}_0). \tag{2.54}$$

The Hamiltonian flow map has several desirable properties as a proposal generating mechanism for a MCMC method. The Hamiltonian is exactly conserved along the trajectories generated by the flow map, i.e. $h(\boldsymbol{x},\boldsymbol{p}) = h \circ \boldsymbol{\Psi}_t(\boldsymbol{x},\boldsymbol{p})$ for all $t \in \mathbb{R}$ and for any initial $\boldsymbol{x},\boldsymbol{p}$ pair. As $\mathrm{p}_{\mathbf{x},\mathbf{p}}(\boldsymbol{x},\boldsymbol{p}) \propto \exp(-h(\boldsymbol{x},\boldsymbol{p}))$ this means Hamiltonian trajectories remain confined to constant density surfaces in the augmented state space. The Hamiltonian flow map is also *volume preserving* - the Jacobian of the flow map $\mathbf{J}_{\boldsymbol{\Psi}_t}$ has determinant one for all t and starting from any initial $(\boldsymbol{x},\boldsymbol{p})$. This volume preservation is a consequence of a stronger geometric property of the dynamic - that the flow map is *symplectic* [33]. Symplecity of the flow map is implied by the condition

$$\mathbf{J}_{\boldsymbol{\Psi}_t}^{\mathsf{T}} \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{J}_{\boldsymbol{\Psi}_t} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{I} & \mathbf{0} \end{bmatrix} \tag{2.55}$$

being satisfied. The symplectic nature of Hamiltonian dynamics have been argued to be key to the performance of HMC in high-dimensional target distributions [5].

A final crucial property of the Hamiltonian flow map is that it exhibits a time-reversal symmetry under negation of the momenta - if $(\boldsymbol{x}', \boldsymbol{p}') = \boldsymbol{\Psi}_t(\boldsymbol{x}, \boldsymbol{p})$ then $(\boldsymbol{x}, -\boldsymbol{p}) = \boldsymbol{\Psi}_t(\boldsymbol{x}', -\boldsymbol{p}')$. If we define $\boldsymbol{F} : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}^D \times \mathbb{R}^D$ as a 'momentum-flip' operator such that $\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{p}) = (\boldsymbol{x}, -\boldsymbol{p})$ then this time reversal symmetry means that the composition $\boldsymbol{F} \circ \boldsymbol{\Psi}_t$ is an *involution*. Further as $\boldsymbol{F}$ also has Jacobian determinant one, then the composition $\boldsymbol{F} \circ \boldsymbol{\Psi}_t$ also itself has a unit Jacobian determinant.

Using the previous result for the Metropolis–Hastings accept ratio for the special case of a deterministic proposal formed by an involution (2.39), we have that a proposal generated by applying $\boldsymbol{F} \circ \boldsymbol{\Psi}_t$ to the current state pair $(\boldsymbol{x}, \boldsymbol{p})$ for any t has an accept probability

$$\alpha(\boldsymbol{x}, \boldsymbol{p}) = \min\left\{1, \frac{\exp(-h \circ \boldsymbol{F} \circ \boldsymbol{\Psi}_t(\boldsymbol{x}, \boldsymbol{p}))}{\exp(-h(\boldsymbol{x}, \boldsymbol{p}))} \left|\mathbf{J}_{\boldsymbol{F} \circ \boldsymbol{\Psi}_t}(\boldsymbol{x}, \boldsymbol{p})\right|\right\} \tag{2.56}$$

$$= \min\{1, \exp(h(\boldsymbol{x}, \boldsymbol{p}) - h \circ \boldsymbol{F} \circ \boldsymbol{\Psi}_t(\boldsymbol{x}, \boldsymbol{p}))\} = 1. \tag{2.57}$$

This results from the Hamiltonian being conserved under the flow map (and momentum flip operator as τ is even in the momenta) and the composed map having unit Jacobian determinant. Therefore we will always accept proposals formed by integrating the ODEs forward by some length of time from the current state and then flipping the momentum. Further on its own the momentum flip operator $\boldsymbol{F}$ also forms an involution with unit Jacobian determinant which exactly conserves the Hamiltonian, and so can also be applied as a 'proposal' on its own with probability of acceptance of one. If we sequentially alternate updates using $\boldsymbol{F} \circ \boldsymbol{\Psi}_t$ and $\boldsymbol{F}$, each defines a valid Markov transition operator which leaves the (extended) target invariant, and in sequential composition the momentum flips cancel. We can therefore construct a Markov chain which leaves the target distribution with density (2.51) invariant by repeatedly generating new states by integrating the Hamiltonian dynamic forward by arbitary lengths of time.

There are two major problems with this scheme. Firstly for most ϕ and τ we will not be able to integrate the ODEs (2.53) exactly and so cannot evaluate the exact flow map $\boldsymbol{\Psi}_t$. Secondly the scheme as proposed would not be ergodic as the Hamiltonian is conserved by each applica-

tion of the flow map, and so all states generated in this way would be confined to a single constant Hamiltonian manifold in the joint target–momentum space.

The first issue can be resolved by instead approximately integrating the ODEs. Importantly by using a *symplectic integrator* we are able to form approximate Hamiltonian flow maps which maintain the key volume-preservation and time-reversibility properties of the exact flow map dynamic (and in fact, as the name suggests, are also symplectic). There is a large glass of such symplectic or geometric integrators [33] however for separable Hamiltonians a particularly popular and easily implementable scheme is the *Störmer–Verlet* or *leapfrog* integrator. A single leapfrog step of time step δt from a current state pair $(\boldsymbol{x}^{(n)}, \boldsymbol{p}^{(n)})$, corresponds to running the following updates

$$\boldsymbol{p}^{(n+1/2)} \leftarrow \boldsymbol{p}^{(n)} - \frac{\delta t}{2} \nabla \phi(\boldsymbol{x}^{(n)}) \tag{2.58}$$

$$\boldsymbol{x}^{(n+1)} \leftarrow \boldsymbol{x}^{(n)} + \delta t \nabla \tau(\boldsymbol{p}^{(n+1/2)}) \tag{2.59}$$

$$\boldsymbol{p}^{(n+1)} \leftarrow \boldsymbol{p}^{(n+1/2)} - \frac{\delta t}{2} \nabla \phi(\boldsymbol{x}^{(n+1)}) \tag{2.60}$$

2.3.3 Simulated tempering

2.4 SUMMARY

The sampling approaches to approximate inference described in this section allow tractable estimation of the integrals involved in many inference problems. In cases where we can tractably generate independent samples from the target distribution, the $\frac{1}{N}$ scaling of the variance of Monte Carlo estimates of expectations with the number of samples N, independent of the dimensionality of the space being integrated over, allows computation of estimates which are sufficiently accurate for many practical purposes without the infeasible exponential blow-up in computation of quadrature methods. Further simple Monte Carlo methods are trivially parallelisable meaning even if generation of each independent sample is relatively expensive, parallel compute devices such as *graphics processing units* (GPUs) and *central processing unit* (CPU) clusters can easily be exploited if available.

Generating independent samples from distributions on high-dimensional spaces with complex dependencies between the variables is generally

non-tractable however. Transform sampling methods offer a scalable approach for generating independent samples for a few special cases such as the multivariate normal distribution. Rejection sampling is more generally applicable however still requires idendifying a proposal distribution with a density which (scaled by a constant) strictly upper bounds the target distribution density. In high-dimensional distributions it will generally be infeasible to find a suitable proposal distribution which is a sufficiently tight bound, with in general the proportion of accepted samples becoming exponentially small as the dimensionality is increased, making rejection sampling only applicable to low-dimensional distributions in practice.

Importance sampling methods may seem initially to offer a solution to this issue, allowing consistent estimates of the values of arbitrary integrals (including marginal density estimations that may not be directly estimated with standard Monte Carlo approaches) providing we can generate independent sample from a distribution which meets the weak condition of having a density which is non-zero everywhere the integrand is non-zero. The general importance sampling estimator is formed as a ratio of two Monte Carlo estimates (2.15); providing these estimators have finite variance each will show the typical $\frac{1}{N}$ scaling of the estimator variance with the number of samples used. In general however unless the importance distribution used is very closely matched to the target distribution, simple importance sampling methods are also impractical in high-dimensional spaces as the constant factors in the variances of the Monte Carlo estimates can grow exponentially with dimension meaning an infeasibly large number of samples are needed to compute estimates of a useful level of accuracy.

Markov chain Monte Carlo methods offer a more a scalable alternative to approximate inference in complex high-dimensional probabilistic models. MCMC methods exploit the intuition that finding a good 'local' approximation to the target distribution — for example within a small region around a poin or varying along only one direction in the target space — is usually a much more tractable task than finding an approximation which matches the target distribution globally, particularly in high dimensional spaces where concentration of measure will mean the typical set of a distribution is usually concentrated in to small region of the space and even seemingly small mismatches between an approximation and target can lead to very little overlap between

the target and approximation typical sets. Markov chain theory shows how we can exploit such local approximations to make perturbative updates to a Markov chain state such that the realisation of the chain converge to generating dependent samples from the target distribution of interest. Although theory can guarantee MCMC estimates will eventually converge to the correct values it is usually difficult to assess the rate of that convergence in practical problems making it difficult to diagnose chain convergence or a lack thereof. This can make application of MCMC methods more challenging from a user-perspective than simple Monte Carlo methods as some level of expertise is usually needed to diagnose and find solutions to convergence issues.

We discussed three general methods for constructing Markov chains which leave a target distribution invariant — the Metropolis–Hastings method, Gibbs sampling and slice sampling. Each of the constructs offers its own advantages and disadvantages and no one method dominates the others in all aspects. It is also common to combine transition operators of different types, for exampling using different methods to update distinct subset of the variables in a model given the remaining variables, or use multiple updates to the same variables to potentially combine the good properties of the individual operators.

Due in part probably to their ease of implementation, Metropolis–Hastings methods based on simple proposal distribution such as Gaussian random-walk Metropolis methods are very commonly used in practice. Performance of such methods is however usually very dependent on the good choice of the algorithm free parameters such as the proposal width / step size, with poor choices leading to very slow mixing chains. In target distributions with a complex geometry, for example where the appropriate scale for state updates may differ signficiantly across the target space, Metropolis–Hastings methods using simple fixed proposals may be unable to mix well even when optimally tuned. One solution to this issue is to use proposals which exploit more information about the local geometry of the distribution such as the gradient-based Hamiltonian Monte Carlo methods we will discuss in Chapter **??**.

Gibbs sampling methods are also very popular with general purpose implementations such as BUGS [21] and JAGS [50] having supported their application to a wide range of models. Although requiring that we are able to decompose the target distribution into complete condition-

als we can tractably generate independent samples from, in the models where it can be applied Gibbs sampling offers the advantage over Metropolis–Hastings methods of not requiring choosing and tuning the free parameters of the proposal distribution. As noted previously however the performance of Gibbs sampling methods is very dependent on the parameterisation of the target distribution, with parameterisations with strong dependencies between the variables tending to lead to very slowly mixing chains. Although this can be alleviated by reparameterisation or using block-updates to coupled variables in some cases, the resulting extra implementation and tuning requirements erode the simplicity advantage compared to competing methods.

The slice sampling algorithms introduced at the end of the last section are more complex than the corresponding algorithms for Metropolis–Hastings and Gibbs sampling, and this added implementation complexity may explain in part the seemingly less widespread use of slice sampling methods in practice[6], although the relatively much more recent introduction of the algorithm is likely also a factor. The tradeoff achieved for this increased implementation complexity however are algorithms which usually require much less tuning than random-walk Metropolis methods to achieve reasonable performance and are more robust than both Gibbs sampling and random-walk Metropolis methods to target distributions with complex geometries. As noted in [40] due to the multiple target density evaluations per chain state update in slice sampling methods, often a well-tuned Metropolis–Hastings method will be able to achieve a greater efficiency in terms of effective samples per run time. However the sometimes significant user time required for tuning can often outweigh the gains in efficiency over slice sampling, and even if automatic adaptive tuning methods are used these will still often struggle to cope with distributions with locally varying geometry. One of the contributions of this thesis will be illustrating how slice sampling methods can often be applied to inference problems where Metropolis–Hastings methods are more commonly used with sometimes significant improvements in robustness and efficiency.

[To do]

6 As a very rough metric at the time of writing the original 1953 Metropolis et al. publication [38] has 35,288 citations recorded on Google Scholar, the 1984 Geman and Geman publication [18] commonly cited as the original sourcce of the Gibbs sampling algorithm 20,485 citations and the 2003 Neal [42] slice sampling publication, 1,444 citations.

BIBLIOGRAPHY

[1] IEEE Standards Association. 'Standard for Floating-Point Arithmetic'. In: *IEEE 754-2008* (2008).

[2] Alessandro Barp, Francois-Xavier Briol, Anthony D Kennedy and Mark Girolami. 'Geometry and Dynamics for Markov Chain Monte Carlo'. In: *arXiv preprint arXiv:1705.02891* (2017).

[3] Michael Betancourt. 'A Conceptual Introduction to Hamiltonian Monte Carlo'. In: *arXiv preprint arXiv:1701.02434* (2017).

[4] Michael Betancourt. 'The Convergence of Markov chain Monte Carlo Methods: From the Metropolis method to Hamiltonian Monte Carlo'. In: *arXiv preprint arXiv:1706.01520* (2017).

[5] Michael Betancourt, Simon Byrne, Sam Livingstone, Mark Girolami et al. 'The geometric foundations of Hamiltonian Monte Carlo'. In: *Bernoulli* 23.4A (2017), pp. 2257–2298.

[6] Joris Bierkens. 'Non-reversible Metropolis–Hastings'. In: *Statistics and Computing* 26.6 (2016), pp. 1213–1228.

[7] George D Birkhoff. 'Proof of the ergodic theorem'. In: *Proceedings of the National Academy of Sciences* 17.12 (1931), pp. 656–660.

[8] George EP Box, Mervin E Muller et al. 'A note on the generation of random normal deviates'. In: *The Annals of Mathematical Statistics* 29.2 (1958), pp. 610–611.

[9] Bob Carpenter, Andrew Gelman, Matt Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Michael A Brubaker, Jiqiang Guo, Peter Li and Allen Riddell. 'Stan: A probabilistic programming language'. In: *Journal of Statistical Software* (2016).

[10] KS Chan. 'Asymptotic behavior of the Gibbs sampler'. In: *Journal of the American Statistical Association* 88.421 (1993), pp. 320–326.

[11] Kung Sik Chan and Charles J Geyer. 'Discussion: Markov chains for exploring posterior distributions'. In: *The Annals of Statistics* 22.4 (1994), pp. 1747–1758.

[12] Ming-Hui Chen and Bruce Schmeiser. 'Toward black-box sampling: A random-direction interior-point Markov chain approach'. In: *Journal of Computational and Graphical Statistics* 7.1 (1998), pp. 1–22.

[13] P Damien, John Wakefield and Stephen Walker. 'Gibbs sampling for Bayesian non-conjugate and hierarchical models by using auxiliary variables'. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 61.2 (1999), pp. 331–344.

[14] Persi Diaconis, Susan Holmes and Radford M Neal. 'Analysis of a nonreversible Markov chain sampler'. In: *Annals of Applied Probability* (2000), pp. 726–752.

[15] Simon Duane, Anthony D Kennedy, Brian J Pendleton and Duncan Roweth. 'Hybrid Monte Carlo'. In: *Physics Letters B* (1987).

[16] Alan E Gelfand and Adrian FM Smith. 'Sampling-based approaches to calculating marginal densities'. In: *Journal of the American Statistical Association* (1990).

[17] Andrew Gelman, Walter R Gilks and Gareth O Roberts. 'Weak convergence and optimal scaling of random walk Metropolis algorithms'. In: *The annals of applied probability* 7.1 (1997), pp. 110–120.

[18] Stuart Geman and Donald Geman. 'Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images'. In: *IEEE Transactions on pattern analysis and machine intelligence* (1984).

[19] Charles J Geyer. *Markov chain Monte Carlo lecture notes (Statistics 8931)*. Tech. rep. University of Minnesota, 1998.

[20] Charles J Geyer. 'The Metropolis-Hastings-Green Algorithm'. 2003. URL: http://www.stat.umn.edu/geyer/f05/8931/bmhg.pdf.

[21] Wally R Gilks, Andrew Thomas and David J Spiegelhalter. 'A language and program for complex Bayesian modelling'. In: *The Statistician* (1994), pp. 169–177.

[22] Walter R Gilks and Pascal Wild. 'Adaptive rejection sampling for Gibbs sampling'. In: *Applied Statistics* (1992), pp. 337–348.

[23] Peter J Green. 'Reversible jump Markov chain Monte Carlo computation and Bayesian model determination'. In: *Biometrika* (1995), pp. 711–732.

[24] J. E. Gubernatis. 'Marshall Rosenbluth and the Metropolis algorithm'. In: *Physics of Plasmas* 12.5 (2005), p. 057303. URL: http://dx.doi.org/10.1063/1.1887186.

[25] Heikki Haario, Eero Saksman and Johanna Tamminen. 'An adaptive Metropolis algorithm'. In: *Bernoulli* (2001), pp. 223–242.

[26] Theodore E Harris. 'The existence of stationary measures for certain Markov processes'. In: *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability*. Vol. 2. 1956, pp. 113–124.

[27] W Keith Hastings. 'Monte Carlo sampling methods using Markov chains and their applications'. In: *Biometrika* (1970).

[28] Bryan D He, Christopher M De Sa, Ioannis Mitliagkas and Christopher Ré. 'Scan Order in Gibbs Sampling: Models in Which it Matters and Bounds on How Much'. In: *Advances in Neural Information Processing Systems*. 2016, pp. 1–9.

[29] Akihisa Ichiki and Masayuki Ohzeki. 'Violation of detailed balance accelerates relaxation'. In: *Physical Review E* 88.2 (2013), p. 020101.

[30] Herman Kahn and Theodore E Harris. 'Estimation of particle transmission by random sampling'. In: *National Bureau of Standards applied mathematics series* 12 (1951), pp. 27–30.

[31] Dirk P. Kroese, Thomas Taimre and Zdravko I. Botev. 'Variance Reduction'. In: *Handbook of Monte Carlo Methods*. John Wiley & Sons, Inc., 2011, pp. 347–380. ISBN: 9781118014967. DOI: 10.1002/9781118014967.ch9. URL: http://dx.doi.org/10.1002/9781118014967.ch9.

[32] Derrick H Lehmer. 'Mathematical methods in large-scale computing units'. In: *Proceedings of a Second Symposium on Large-Scale Digital Calculating Machinery (1949)*. 1951, pp. 141–146.

[33] Benedict Leimkuhler and Sebastian Reich. *Simulating Hamiltonian dynamics*. Cambridge University Press, 2004.

[34] David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge University Press, 2003.

[35] George Marsaglia. 'Random numbers fall mainly in the planes'. In: *Proceedings of the National Academy of Sciences* 61.1 (1968), pp. 25–28.

[36] Makoto Matsumoto and Takuji Nishimura. 'Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator'. In: *ACM Transactions on Modeling and Computer Simulation (TOMACS)* 8.1 (1998), pp. 3–30.

[37] Kerrie L Mengersen and Richard L Tweedie. 'Rates of convergence of the Hastings and Metropolis algorithms'. In: *The annals of Statistics* 24.1 (1996), pp. 101–121.

[38] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller and Edward Teller. 'Equation of state calculations by fast computing machines'. In: *The Journal of Chemical Physics* (1953).

[39] Sean P Meyn and Richard L Tweedie. *Markov chains and stochastic stability*. Springer Science & Business Media, 1993.

[40] Iain Murray, Ryan Prescott Adams and David J.C. MacKay. 'Elliptical slice sampling'. In: JMLR: W&CP 9 (2010), pp. 541–548.

[41] Radford M Neal. *Markov chain Monte Carlo methods based on 'slicing' the density function*. Tech. rep. 9722. Department of Statistics, University of Toronto, 1997.

[42] Radford M Neal. 'Slice sampling'. In: *Annals of statistics* (2003).

[43] Radford M Neal. 'Improving asymptotic variance of MCMC estimators: Non-reversible chains are better'. In: *arXiv preprint math/0407281* (2004).

[44] Radford M Neal. 'MCMC using Hamiltonian dynamics'. In: *Handbook of Markov Chain Monte Carlo*. Chapman & Hall/CRC, 2011. Chap. 5, pp. 113–162.

[45] John von Neumann. 'Various techniques used in connection with random digits'. In: *National Bureau of Standards applied mathematics series* 3 (1951), pp. 36–38.

[46] Robert Nishihara, Iain Murray and Ryan P Adams. 'Parallel MCMC with generalized elliptical slice sampling.' In: *Journal of Machine Learning Research* 15.1 (2014), pp. 2087–2112.

[47] Sheehan Olver and Alex Townsend. 'Fast inverse transform sampling in one and two dimensions'. In: *arXiv preprint arXiv:1307.1223* (2013).

[48] Art B. Owen. 'Importance sampling'. In: *Monte Carlo theory, methods and examples.* 2013. URL: http://statweb.stanford.edu/~owen/mc/Ch-var-is.pdf.

[49] Peter H Peskun. 'Optimum Monte-Carlo sampling using Markov chains'. In: *Biometrika* 60.3 (1973), pp. 607–612.

[50] Martyn Plummer et al. 'JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling'. In: *Proceedings of the 3rd international workshop on distributed statistical computing.* Vol. 124. Vienna. 2003, p. 125.

[51] Martyn Plummer, Nicky Best, Kate Cowles and Karen Vines. 'CODA: Convergence Diagnosis and Output Analysis for MCMC'. In: *R News* 6.1 (2006), pp. 7–11. URL: http://CRAN.R-project.org/doc/Rnews/Rnews_2006-1.pdf.

[52] Adrian E Raftery and Steven Lewis. *How many iterations in the Gibbs sampler?* Tech. rep. Washington University, Seattle, Department of Statistics, 1991.

[53] Gareth O Roberts and Jeffrey S Rosenthal. 'Optimal scaling for various Metropolis-Hastings algorithms'. In: *Statistical science* 16.4 (2001), pp. 351–367.

[54] Gareth O Roberts and Jeffrey S Rosenthal. 'General state space Markov chains and MCMC algorithms'. In: *Probability Surveys* 1 (2004), pp. 20–71.

[55] Gareth O Roberts and Adrian FM Smith. 'Simple conditions for the convergence of the Gibbs sampler and Metropolis-Hastings algorithms'. In: *Stochastic processes and their applications* 49.2 (1994), pp. 207–216.

[56] Gareth O Roberts and Richard L Tweedie. 'Geometric convergence and central limit theorems for multidimensional Hastings and Metropolis algorithms'. In: *Biometrika* (1996), pp. 95–110.

[57] Jeffrey S Rosenthal et al. 'Optimal proposal distributions and adaptive MCMC'. In: *Handbook of Markov Chain Monte Carlo* (2011), pp. 93–112.

[58] John Salvatier, Thomas V Wiecki and Christopher Fonnesbeck. 'Probabilistic programming in Python using PyMC3'. In: *PeerJ Computer Science* (2016).

[59] Yi Sun, Jürgen Schmidhuber and Faustino J Gomez. 'Improving the asymptotic performance of Markov chain Monte-Carlo by inserting vortices'. In: *Advances in Neural Information Processing Systems*. 2010, pp. 2235–2243.

[60] Hidemaro Suwa and Synge Todo. 'Markov chain Monte Carlo method without detailed balance'. In: *Physical review letters* 105.12 (2010), p. 120603.

[61] Madeleine B Thompson. 'A comparison of methods for computing autocorrelation time'. In: *arXiv preprint arXiv:1011.0175* (2010).

[62] Luke Tierney. 'Markov chains for exploring posterior distributions'. In: *The Annals of Statistics* (1994), pp. 1701–1728.

[63] Konstantin S Turitsyn, Michael Chertkov and Marija Vucelja. 'Irreversible Monte Carlo algorithms for efficient sampling'. In: *Physica D: Nonlinear Phenomena* 240.4 (2011), pp. 410–414.

[64] Stanislaw Ulam and Nicholas Metropolis. 'The Monte Carlo method'. In: *Journal of the American Statistical Association* 44.247 (1949), pp. 335–341.