S2: Advanced Statistical Methods for Data Science

MPhil in Data Intensive Science
University of Cambridge

Dr Christopher J. Moore
Lent Term 2025

cjm96@cam.ac.uk

This course, *S2: Advanced Statistical Methods for Data Science*, is lectured in the Lent term of the MPhil in Data Intensive Science programme at the University of Cambridge, 2023/24. The course aims to provide you with a good understanding of how quantitative conclusions are drawn from scientific data. The course focusses in particular on Bayesian methods and the computational techniques that allow them to be applied to scientific data sets.

Questions about the course can be directed to me at cjm96@cam.ac.uk.

These notes are still new and I will be adding to them during the course; they will inevitably contain mistakes. If you spot any, please tell me at cjm96@cam.ac.uk and I will fix them in the next version.

Contents

Chapter 0

Syllabus and Structure of the Notes

0.1 Syllabus

From the public course page https://mphildis.bigdata.cam.ac.uk/statistical-methods-for-data-science/, the syllabus for this course is as follows...

Probabilistic inference: probabilistic modelling for scientific applications, constructing likelihoods, prior specification, properties of posterior estimates and summaries.

Bayesian computation: Monte Carlo methods, importance sampling, Markov Chain Monte Carlo (Metropolis-Hastings, Gibbs sampling, slice sampling, Hamiltonian Monte Carlo), nested sampling, convergence diagnostics, Bayesian workflow in practice, approximate inference methods.

Model comparison: posterior predictive checks, simulation-based calibration, likelihood ratio, information criteria, Bayes factors, cross-validation.

Advanced topics: hierarchical models, shrinkage and partial pooling, non-parametric Bayesian methods (Gaussian processes), probabilistic graphical models, selection effects, topics in time series analysis and spatial statistics.

0.2 Resources

There are many useful textbooks on this subject. However, no single book one covers all the material at the right level and in the way it will be done in this course. Nevertheless, the following book may be useful and were used in preparing these notes:

- D. S. Sivia, "Data Analysis: A Bayesian Tutorial"
- D. J. C. Mackay, "Information Theory, Inference and Learning Algorithms"

0.3 Course Timetable

There are 24 lectures across the 8 week term. Lectures are on Thursdays (11.00-12.00, room East 1), Mondays (14.00-15.00, room East 2), and Tuesdays (11.00-12.00, room East 1) in the West Hub building.

W1	L1	Thur 23/01/23	11.00-12.00	East 1
	L2	Mon 27/01/23	14.00-15.00	East 2
	L3	Tue 28/01/23	11.00-12.00	East 1
W2	L4	Thur 30/01/23	11.00-12.00	East 1
	L5	Mon 03/02/23	14.00-15.00	East 2
	L6	Tue 04/02/23	11.00-12.00	East 1
W3	L7	Thur 06/02/23	11.00-12.00	East 1
	L8	Mon 10/02/23	14.00-15.00	East 2
	L9	Tue 11/02/23	11.00-12.00	East 1
W4	L10	Thur 13/02/23	11.00-12.00	East 1
	L11	Mon 17/02/23	14.00-15.00	East 2
	L12	Tue 18/02/23	11.00-12.00	East 1
W5	L13	Thur 20/02/23	11.00-12.00	East 1
	L14	Mon 24/02/23	14.00-15.00	East 2
	L15	Tue 25/02/23	11.00-12.00	East 1
W6	L16	Thur 27/02/23	11.00-12.00	East 1
	L17	Mon 03/03/23	14.00-15.00	East 2
	L18	Tue 04/03/23	11.00-12.00	East 1
W7	L19	Thur 06/03/23	11.00-12.00	East 1
	L20	Mon 10/03/23	14.00-15.00	East 2
	L21	Tue 11/03/23	11.00-12.00	East 1
W8	L22	Thur 13/03/23	11.00-12.00	East 1
	L23	Mon 17/03/23	14.00-15.00	East 2
	L24	Tue 18/03/23	11.00-12.00	East 1

0.4 Test LaTeX Environments

Box 0.1: Example box

Extra material outside the main flow of the notes.

Example 0.1: Example exercise with worked solution

An example problem with an accompanying worked solution.

Exercise 0.1: Example exercise
An example problem left as an exercise. Some of these will be discussed in the example classes.

Chapter 1

Probability and Inference

1.1 Statistics, Probability, and Inference

I will begin by trying to give working definitions of these three terms.

Statistics concerns the collection, analysis, and presentation of data. Good statistics aims to get the maximum amount of information from the available data. A common complaint in all areas of science is "we do not have enough data", so we must endeavour to extract as much as possible from that which is available. This is especially true in astronomy where it is often not possible to make repeat observations to get additional data.

The usual situation in science is that we have some (usually quite limited) data and we want to use it to learn something about the Universe. For example, we wish to measure the value of some quantity or to test whether a particular model fits the data. The process of forward modelling involves making predictions; starting from a model or a known set of laws and initial conditions calculating the outcome. These predictions are almost always uncertain and probability is the mathematical discipline concerned with quantifying the likelihood that certain events will occur.

We then have to reason backwards in order to draw conclusions about our model; i.e. solving the "inverse problem". This is inference. Inference can be a purely logical argument leading to a certain conclusion. However, in the sciences, inference is almost always probabilistic, with the observations providing a certain amount of evidence for or against a particular conclusion. It is our task to quantify this evidence.

The roles of probability and inference are illustrated in the cartoon in Fig. 1.1

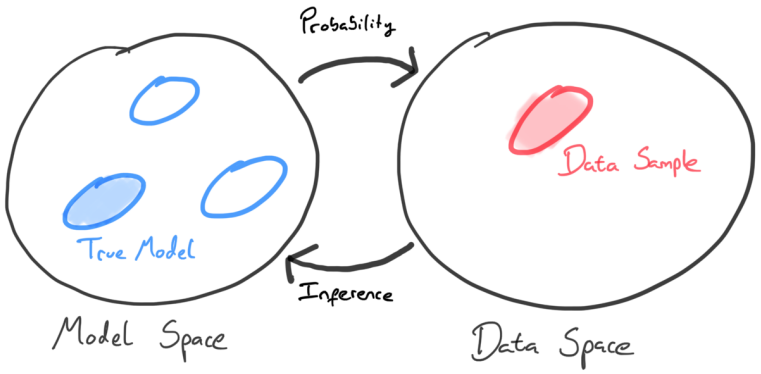


Figure 1.1: An illustration of the roles of probability and inference.

1.2 Bayes' Theorem

Bayes' theorem concerns *conditional probabilities*. These are denoted $P(A|B)$ which is read as "the probability of A being true given that B is true".

The *product rule* (sometimes known as the *chain rule*) of probability states that

$$P(A \text{ and } B) = P(A|B)P(B)\,. \tag{1.1}$$

This can either be considered as one of the axioms of probability or [rearranged for $P(A|B)$] as the definition of the conditional probability.

Two events A and B are said to be *independent* if $P(A \text{ and } B) = P(A)(B)$. Two events are independent if one of them occurring does not affect the probability of the other occurring; from equation 1.1 we see that

$$P(A \text{ and } B) = P(A)P(B) \iff P(A|B) = P(A)\,, \tag{1.2}$$
$$P(B \text{ and } A) = P(B)P(A) \iff P(B|A) = P(B)\,. \tag{1.3}$$

Theorem 1.2.1. *Bayes' theorem;*

$$P(A|B) = P(B|A)\frac{P(A)}{P(B)}\,. \tag{1.4}$$

Proof. Follows from the product rule (Eq. 1.1) and the symmetry of "and"; $P(A \text{ and } B) = P(B \text{ and } A)$. □

Notation: it is common to abbreviate the "and" in probabilities to just a comma; i.e. we write $P(A \text{ and } B) = P(A, B)$.

For the purpose of performing inference, we will usually work with Bayes' theorem in the following alternative form;

$$P(M|D, I) = \frac{P(D|M, I)P(M|I)}{P(D|I)}\,. \tag{1.5}$$

Here M denotes our model (with particular values for all of its parameters) and D denotes the data. Every probability is conditional on I, this denotes any/all other prior information that is assumed throughout the analysis; although conceptually important and sometimes containing subtle assumptions that profoundly affect the results of our inference, I is usually omitted from our notation in all terms in equation 1.5.

Equation 1.5 is so important and occurs so often that special names and symbols are given to each of the individual terms:

1. $P(D|M, I) = \mathcal{L}(D|M)$ is called the **likelihood**, it is the probability of obtaining the data given our model;

2. $P(M|I) = \pi(M)$ is called the **prior**, it the probability we assign to our model being correct *before* performing the experiment;

3. $P(M|D, I)$ is called the **posterior**, it is the probability that our model is correct *after* performing the experiment;

4. $P(D|I) = Z$ is called the **evidence**, it acts as a normalisation constant in Bayes' theorem. (The evidence isn't usually very important in parameter estimation problems, but is plays a central role in model selection). If the model has some free parameters, M = parameters, then from Bayes' theorem (1.5) we see that the evidence plays the

role of a normalisation constant for the posterior probability distribution and is equal to the following integral;

$$Z = \int \mathrm{d}(\text{parameters})\ \pi(\text{parameters})\mathcal{L}(D|\text{parameters})\ . \tag{1.6}$$

Using this notation, Bayes' theorem is written as

$$P(M|D) = \frac{\mathcal{L}(D|M)\pi(M)}{Z}\,. \tag{1.7}$$

In a Bayesian inference problem we are generally given the likelihood and the prior as inputs and the task is to study the posterior (and sometimes compute the evidence).

Box 1.1: Bayesian versus frequentist probabilities

Historically, there has been debate between the *Bayesian* and *frequentist* views of probability. The debate does *not* concern the validity of Bayes' theorem itself (Eq. 1.4) but rather the way it is applied to a model and its parameters (in Eq. 1.5).

In the frequentist view, probability is defined as the relative frequency of an event occurring in the limit of many trials. This is the way most of us are first taught to think about probability at school. This means that we must (at least in principle) be able to at imagine repeating some kind of trial (or experiment) many times in order to assign a probability to the outcome. For this reason, the likelihood, $\mathcal{L}(D|M)$, is a valid frequentist probability because it assigns a probability to the data, D, which is the outcome of an experiment which can be repeated.

In the Bayesian view, probabilities are instead interpreted as representing our state of knowledge or as quantification of a personal belief about something. Anything we don't know with certainty can be assigned a probability. What's the probability that the 10^{100}th digit of Pi is 3? For this reason, the prior and posterior, $\pi(M)$ and $P(M|D)$, are valid Bayesian probabilities but not frequentist ones. (How do you repeat a model?) The Bayesian view allows us apply the tools of probability to a wider range of problems and this has proved very fruitful.

1.3 Likelihoods

The *likelihood function* (or simply the *likelihood*) is the probability of observed data conditioned on a particular choice of the model and any model parameters.

In Eq. 1.5 the likelihood was denoted $\mathcal{L}(D|M)$, where D is the data and M is the model. Up until now we have been agnostic about what exactly D and M are. The data D might include text, time series measurements or images. The model M might involve multiple parameters, both continuous and discrete (e.g. labels for data). For the specific case when the data is a continuous random variable, $D = x$, and when the model depends only on a continuous parameter, $M = \theta$, we will use the notation $\mathcal{L}(x|\theta)$.

The likelihood is a probability distribution for the data conditioned on a particular value of the model parameter(s). For this reason, you will often see the likelihood written in the form $P(x|\theta)$. However, it is also a function of the model parameters (this is the reason for the name *likelihood function*) and this is why you will also often see it written as something like $L(\theta)$. These two guises of the likelihood are illustrated further in Box. 1.2 To avoid the confusion of multiple notations, in these notes we will sick to the hybrid notation $\mathcal{L}(x|\theta)$.

Box 1.2: The two guises of the likelihood

A source emits unstable particles that decay after travelling a distance x. A number N of decays are observed at distances $\{x_1, x_2, \ldots, x_N\}$ from the source.

For a single particle, the distance travelled, $x > 0$, has an exponential distribution with a characteristic decay length λ,

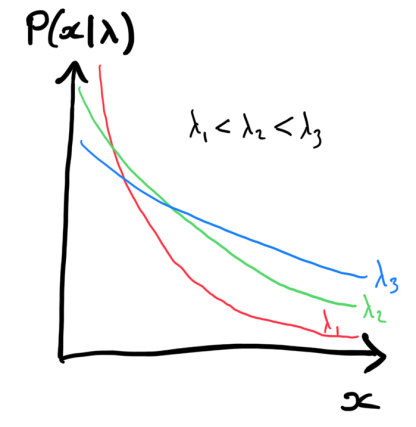
$$P(x|\lambda) = \frac{\exp(-x/\lambda)}{\lambda}. \tag{i}$$

Or, for a number of independent particles

$$P(\{x\}|\lambda) = \frac{\exp(-\sum_i x_i/\lambda)}{\lambda^N}. \tag{ii}$$

This is the probability distribution (or likelihood) of the data, conditioned on a

specific value for λ. If the decay length is known, this allows us the predict decay locations. (Forward model.)



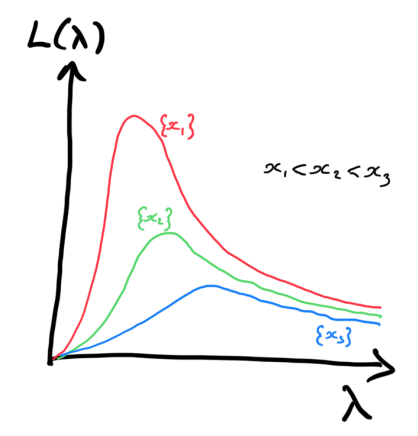
For different values of λ, this describes a set of exponential curves. These curves are all normalised with unit area.

But we've already measured $\{x_i\}$ and want to find the decay length. (This is inference, or the inverse problem. The same expression can be used to answer this, when viewed as a function of λ. This is the likelihood function,

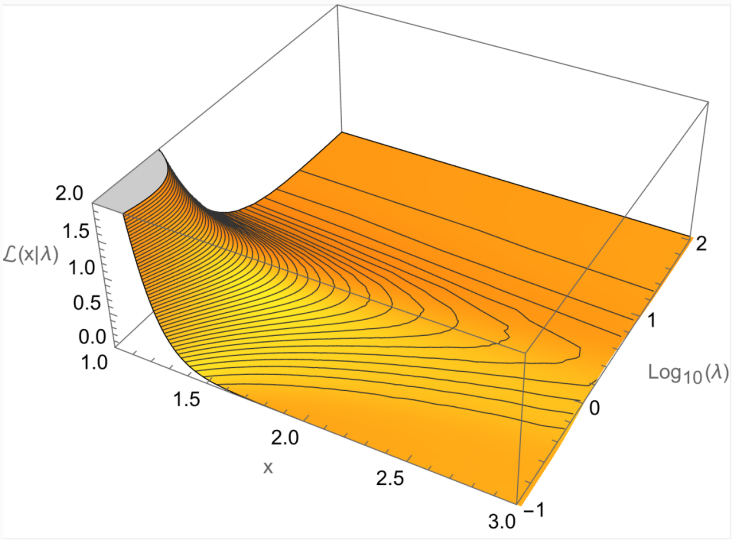
$$L(\lambda) = \frac{\exp(-\sum_i x_i/\lambda)}{\lambda^N}. \tag{iii}$$

We can find the value of the decay length parameter that maximises the likelihood and use this as an estimator of λ. (In practice, it is almost always easier to maximise the log-likelihood, rather than working with the likelihood itself; maximising the log-likelihood is the same as maximising the likelihood because the logarithm function is a monotonic.) The maximum likelihood estimator $\hat{\lambda}$ is defined as

$$\frac{\partial}{\partial\lambda}\Big|_{\lambda=\hat{\lambda}} \log L(\lambda) = 0 \quad \Rightarrow \quad \hat{\lambda} = \frac{\sum_i x_i}{N}. \tag{iv}$$



For different sets of measured decay locations $\{x\}$ this describes a very different set of curves. These curves are NOT normalised. The relationship between the two sets of curves sketched above can be understood by thinking about slices through the surface $\mathcal{L}(x|\lambda) = \exp(-x/\lambda)/\lambda$ taken parallel to both axes.



The likelihood can be thought of in two ways: the *probability distribution* of the data conditioned on the model parameters $P(d|\lambda)$, or as a *function* of the model parameters $L(\lambda)$. But it's always the same expression which we denote here using the hybrid notation $\mathcal{L}(x|\lambda)$.

Constructing a likelihood for a specific problem requires us to know something about the experiment that produced the data. It is also often necessary to make some simplifying assumptions to obtain a useful expression (such as assuming different measurements are perfectly independent, or that errors are exactly Gaussian distributed). These assumptions need to be tested if any subsequent Bayesian inference that uses the likelihood is to be trusted.

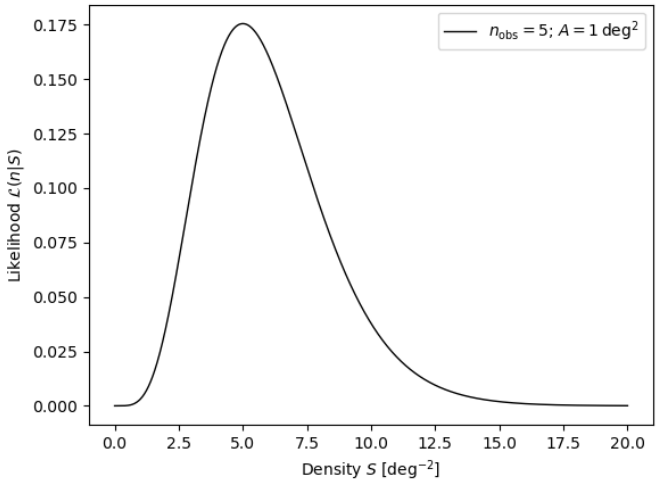
Box 1.3: Number density of stars

We want to measure the density S (i.e. number per square degree) of stars brighter than a particular threshold magnitude on a particular patch of sky. We perform a survey of the area and find $n = 5$ such stars in an area $A = 1$ square degree. For simplicity, we will assume that stars occur independently (no binaries!) and uniformly across this patch of the sky at a constant average density S.

Suppose we knew exactly the number density S. (Obviously we don't! That's the whole point, we are going to try to infer S from the count n. But just suppose.) Then the probability of finding n stars in this area is given by the Poisson distribution with expectation parameter $\lambda = AS$;

$$\mathcal{L}(n|S) = \frac{(AS)^n \exp(-AS)}{n!}. \quad \text{(i)}$$

This is the likelihood; given a particular value of S, how likely is it that we found this many stars?



We'll return to this example in Boxes 1.4 to 1.7 below.

Exercise 1.1:
Think about the area under the curve S plotted above as a function of S in Box 1.3. Only one of the following statements is true. Which?

$$\sum_{n=0}^{\infty} \mathcal{L}(n|S) \stackrel{?}{=} 1 \quad , \quad \int \mathrm{d}S \; \mathcal{L}(n|S) \stackrel{?}{=} 1. \tag{i}$$

1.3.1 Fisher Information

The *Fisher information* is a way of quantifying the information that an particular observation, or data point, x contains about an unknown parameter θ.

The *score* is defined as the partial derivative of the log-likelihood function with respect to a model parameter,

$$\mathcal{S}(\theta) = \frac{\partial \log \mathcal{L}(x|\theta)}{\partial \theta}. \tag{1.8}$$

(If there are multiple parameters then the score is the gradient vector.) At a particular value of the parameter, the score measures the steepness of the log-likelihood and thereby the sensitivity of the log-likelihood to small changes in that parameter. A large value of the score indicates a likelihood function that carries a lot of information about θ.

Suppose that the model for the likelihood is correct and the data really is distributed as $x \sim \mathcal{L}(x|\theta_{\text{true}})$, where $\theta = \theta_{\text{true}}$ is the true value of the parameter. Under some mild regularity conditions, if the score is evaluated at the true parameter, then the expected value of the score in any experiment is zero. In order to show this, we compute the expectation value over many different experiments; i.e. the expectation over the data x.

$$\begin{aligned}
\mathrm{E}_x\left[\mathcal{S}(\theta_{\text{true}})\right] &= \int \mathrm{d}x\; S(\theta_{\text{true}})\mathcal{L}(x|\theta_{\text{true}}) && (1.9)\\
&= \int \mathrm{d}x\; \frac{\partial \log \mathcal{L}(x|\theta)}{\partial \theta}\Big|_{\theta=\theta_{\text{true}}} \mathcal{L}(x|\theta_{\text{true}}) && (1.10)\\
&= \int \mathrm{d}x\; \frac{1}{\mathcal{L}(x|\theta_{\text{true}})} \frac{\partial \mathcal{L}(x|\theta)}{\partial \theta}\Big|_{\theta=\theta_{\text{true}}} \mathcal{L}(x|\theta_{\text{true}}) && (1.11)\\
&= \int \mathrm{d}x\; \frac{\partial \mathcal{L}(x|\theta)}{\partial \theta}\Big|_{\theta=\theta_{\text{true}}} && (1.12)\\
&= \frac{\partial}{\partial \theta}\Big|_{\theta=\theta_{\text{true}}} \int \mathrm{d}x\; \mathcal{L}(x|\theta) && (1.13)\\
&= \frac{\partial}{\partial \theta}\Big|_{\theta=\theta_{\text{true}}} 1 && (1.14)\\
&= 0 && (1.15)
\end{aligned}$$

Where on line 1.12 we have assumed the likelihood is sufficiently regular that the order of integration and differentiation can be changed, and on line 1.14 we have used the fact that the likelihood is a (normalised) probability distribution for the data x.

The Fisher information $\mathcal{I}$ is defined to be variance of the score (evaluated at the true value of the model parameter);

$$\begin{aligned}
\mathcal{I}(\theta) &= \mathrm{Var}_x\big(\mathcal{S}(\theta)\big) && (1.16)\\
&= \mathrm{E}_x\left[\mathcal{S}(\theta_{\text{true}})^2\right] - \mathrm{E}_x\left[\mathcal{S}(\theta_{\text{true}})\right]^2 && (1.17)\\
&= \int \mathrm{d}x\; \mathcal{L}(x|\theta)\mathcal{S}(\theta)^2, && (1.18)
\end{aligned}$$

where in going from the second to the third line we have used Eq. 1.15. Note that because it is defined as a variance, $\mathcal{I} \geq 0$. A higher value of the variance implies that the absolute value of the score is larger on average.

Note the score and Fisher information are both dimensionfull quantities; we have $[\mathcal{S}(\theta)] = [\theta]^{-1}$ and $[\mathcal{I}(\theta)] = [\theta]^{-2}$.

Under some mild regularity conditions, the Fisher information can also be expressed in terms of the second derivative of the log-likelihood w.r.t. the parameter θ (see Ex. 1.1);

$$\mathcal{I}(\theta) = -\mathrm{E}_x \left[\frac{\partial^2}{\partial\theta^2}\Big|_{\theta=\theta_{\mathrm{true}}} \log \mathcal{L}(x|\theta) \right] \tag{1.19}$$

$$= -\int \mathrm{d}x\ \mathcal{L}(x|\theta) \frac{\partial^2}{\partial\theta^2}\Big|_{\theta=\theta_{\mathrm{true}}} \log \mathcal{L}(x|\theta) \tag{1.20}$$

If there are multiple model parameters, i.e. θ becomes θ_μ with $\mu = 1, 2, \ldots, n$, then the Fisher information becomes a square $n \times n$ matrix, called the *Fisher information matrix*;

$$\mathcal{I}_{\mu\nu}(\theta) = -\mathrm{E}_x \left[\frac{\partial^2}{\partial\theta_\mu \partial\theta_\nu}\Big|_{\theta_{\mathrm{true}}} \log \mathcal{L}(x|\theta)| \right] \tag{1.21}$$

$$= -\int \mathrm{d}x\ \mathcal{L}(x|\theta) \frac{\partial^2}{\partial\theta_\mu \partial\theta_\nu}\Big|_{\theta_{\mathrm{true}}} \log \mathcal{L}(x|\theta) \tag{1.22}$$

Example 1.1: Fisher information

Show that the definition of the Fisher information in Eq. 1.19 is equivalent to the definition in Eq. 1.18.

This is similar to the calculation of the expectation of the score performed in Eqs. 1.9 to 1.15 above. Start from the second definition and work back to recover

the first.

$$\mathcal{I}(\theta) = -\int \mathrm{d}x\ \frac{\partial^2 \log \mathcal{L}(x|\theta)}{\partial \theta^2} \mathcal{L}(x|\theta) \tag{i}$$

$$= -\int \mathrm{d}x\ \frac{\partial}{\partial \theta} \left(\frac{\frac{\partial \mathcal{L}(x|\theta)}{\partial \theta}}{\mathcal{L}(x|\theta)} \right) \mathcal{L}(x|\theta) \tag{ii}$$

$$= -\int \mathrm{d}x\ \left(\frac{\frac{\partial^2 \mathcal{L}(x|\theta)}{\partial \theta^2}}{\mathcal{L}(x|\theta)} - \frac{\left(\frac{\partial \mathcal{L}(x|\theta)}{\partial \theta}\right)^2}{\mathcal{L}(x|\theta)^2} \right) \mathcal{L}(x|\theta) \tag{iii}$$

$$= 0 + \int \mathrm{d}x\ \mathcal{S}(\theta)^2 \mathcal{L}(x|\theta). \tag{iv}$$

On the final line, the first term equals zero by reversing the order of differentiation and integration; i.e. $\partial_\theta^2 \int \mathrm{d}x\, \mathcal{L} = \partial_\theta^2 1 = 0$.

Notice that the Fisher information depends on the choice of a particular parametrisation for the model. If ϕ is an alternative parameter related to original parameter by a (smooth) function $\theta(\phi)$, then the Fisher information changes according to

$$\mathcal{I}(\phi) = \mathcal{I}(\theta) \left(\frac{\partial \theta}{\partial \phi} \right)^2, \tag{1.23}$$

where $\mathcal{I}(\theta)$ and $\mathcal{I}(\phi)$ are the Fisher informations for the two parameters respectively.

As already mentioned, the Fisher information is strictly positive, $\mathcal{I}(\theta) \geq 0$. It also has a strictly increasing quality; if two experiments are performed, yielding data x_1 and x_2, then the Fisher information of the combined experiment is greater than either of the individual experiments ($\mathcal{I}_{x_1,x_2}(\theta) \geq \mathcal{I}_{x_1}(\theta)$ and $\mathcal{I}_{x_1,x_2}(\theta) \geq \mathcal{I}_{x_2}(\theta)$). In the special case where the experiments are independent and the likelihood factorises, $\mathcal{L}(x_1, x_2|\theta) = \mathcal{L}(x_1, \theta)\mathcal{L}(x_2|\theta)$, the Fisher information is additive; $\mathcal{I}_{x_1,x_2}(\theta) = \mathcal{I}_{x_1}(\theta) + \mathcal{I}_{x_2}(\theta)$.

The Fisher information also appears in the (expectation of the) Taylor expansion of the log-likelihood about the true value of the model parameters. If we expand the log-likelihood about the true model parameters then we have

$$\begin{aligned} \log \mathcal{L}(x|\theta) &= \mathcal{L}(x|\theta_{\text{true}}) + (\theta - \theta_{\text{true}}) \frac{\partial \log \mathcal{L}(x|\theta)}{\partial \theta}\Big|_{\theta=\theta_{\text{true}}} + \frac{1}{2}(\theta - \theta_{\text{true}})^2 \frac{\partial^2 \log \mathcal{L}(x|\theta)}{\partial^2 \theta}\Big|_{\theta=\theta_{\text{true}}} \dots \\ &= \text{const} + (\theta - \theta_{\text{true}}) \mathcal{S}(\theta) + \frac{1}{2}(\theta - \theta_{\text{true}})^2 \frac{\partial^2 \log \mathcal{L}(x|\theta)}{\partial^2 \theta}\Big|_{\theta=\theta_{\text{true}}} \dots \end{aligned} \tag{1.24}$$

The actual shape of the log-likelihood depends on the realisation of the data, x, in a particular experiment. If we repeated the experiment many times obtaining a different data set x each time, then the likelihoods in all the experiments would all look slightly different. However, we can remove the x dependence by asking what the log-likelihood function looks like on average. Taking the expectation of the above expression over realisations of the data gives

$$\mathrm{E}_x\left[\log \mathcal{L}(x|\theta)\right] = \text{const} - \frac{1}{2}(\theta - \theta_{\text{true}})^2 \mathcal{I}(\theta_{\text{true}}) \ldots \tag{1.25}$$

where we have used the fact that the expectation of the score vanishes (Eq. 1.15) and the definition of the Fisher information in Eq. 1.19.

If there are multiple parameters then the expansion in Eq. 1.25 becomes

$$\mathrm{E}_x\left[\log \mathcal{L}(x|\theta)\right] = \text{const} - \frac{1}{2}\sum_{\mu=1}^{n}\sum_{\nu=1}^{n}(\theta_\mu - \theta_{\text{true},\,\mu})(\theta_\nu - \theta_{\text{true},\,\nu})\mathcal{I}_{\mu\nu}(\theta') \ldots \tag{1.26}$$

The Fisher information is often using in *experiment design*. Because the above expressions do not depend on the data (the dependence on x disappears when taking the expectation), they can be evaluated without actually performing the experiment.

This can also be used to approximate the likelihood (or later in a Bayesian context, the posterior) as a multivariate Gaussian in the vicinity of the true model parameters.

1.4 Bayesian Parameter Estimation

In this section we will see a number of examples of the application of Bayes' theorem to simple parameter estimation problems where the model contains just a single parameter. The computational tools for applying the same Bayesian methodology to more interesting (and realistic) multi-parameter problems are described in the next chapter. This section will also highlight the importance of the prior in Bayesian analysis.

In order to perform a Bayesian analysis we first have to choose a prior for the model parameters. How should the prior be chosen? Usually, the model parameters are continuous, so the prior is a probability density function. So the question becomes how should we choose this PDF? There is no unique answer. However, as a general principle, the prior should be chosen to describe our state of knowledge about the model parameters *before* performing the experiment. The following subsections contain several examples of how this rather vague statement can be applied to the example in Box 1.3.

1.4.1 Ignorance Priors (uniform)

We often pretend that we don't know much (or anything) about the value of model parameters, except perhaps that is must lie in a certain range of values. (Is this every really true?) But if we really don't have any information to guide us then we should try to choose a prior that reflects our ignorance.

For some types of parameters, a uniform distribution is the obvious way to try and achieve this. For location parameters such as the (x, y) coordinates of a point on an image we would naturally want our prior to be invariant under a translation of the origin by a constant Δx;

$$\pi(x)\mathrm{d}x = \pi(x + \Delta x)\mathrm{d}(x + \Delta x) \quad \Rightarrow \quad \pi(x) \propto \text{const.} \tag{1.27}$$

The uniform distribution (within certain limits) is the only distribution that has this property.

Notation: it will sometimes be convenient to use the following notation for the *indicator function* for distributions with compact support;

$$\mathbb{1}_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{else} \end{cases} . \tag{1.28}$$

So in 1D, $x \in \mathbb{R}$, we can write the uniform prior in Eq. 1.27 as

$$\pi(x) = \frac{\mathbb{1}_{(x_{\min}, x_{\max})}(x)}{x_{\max} - x_{\min}}. \tag{1.29}$$

Strictly speaking, the prior is a probability distribution and must therefore be properly normalised (or at least normalisable; meaning its integral is finite). When using a uniform distribution as a prior this means we must specify the minimum and maximum allowed values of the parameter: $x_{\min}$ and $x_{\max}$. However, in practice these are often omitted. Omitting the lower/upper limits means the prior is unnormalisable; a prior that is not normalisable is referred to as an *improper prior*.

Box 1.4: Number density of stars. . . uniform prior

Here, we choose to use a uniform prior on the parameter S. This requires that we choose a prior range; we choose $0 < S < S_{\max} = 20\,\mathrm{deg}^{-2}$. The justification for

this choice of upper limit comes at the end of the analysis when we will find that the posterior has negligible support in the region $S \sim 20\,\mathrm{deg}^{-2}$ (see plot of the posterior Fig. 1.2).

$$\pi(S) = \frac{\mathbb{1}_{(0,S_{\max})}(S)}{S_{\max}} \tag{i}$$

Using the likelihood from Box 1.3, Bayes' theorem gives the posterior as

$$P(S|n) = \frac{(AS)^n \exp(-AS)\mathbb{1}_{(0,S_{\max})}(S)}{S_{\max} n! Z}, \tag{ii}$$

where the evidence (integral evaluated numerically) is

$$Z = \int_0^{S_{\max}} \mathrm{d}S\, \frac{(AS)^n \exp(-AS)}{S_{\max} n!} \approx 0.050. \tag{iii}$$

1.4.2 Ignorance Priors (log uniform)

A uniform distribution is not the right choice for all parameters. For *scale parameters* associated with the magnitude a particular quantity (such as length, mass, lifetime, etc) we would naturally want our prior to be invariant under a change of the units (by a constant factor α) used to measure this quantity;

$$\pi(x)\mathrm{d}x = \pi(\alpha x)\mathrm{d}(\alpha x) \quad \Rightarrow \quad \pi(x) \propto \frac{1}{x}. \tag{1.30}$$

This is equivalent to placing a uniform prior on the logarithm of the scale parameter; $\pi(\log x) \propto \text{const}$. This can be easily checked by a change of variables.

The log-uniform distribution (again, within certain limits) is the only distribution that has the required invariance property. In its properly normalised form, the log-uniform prior is given by

$$\pi(x) = \frac{\mathbb{1}_{(x_{\min},x_{\max})}(x)}{x \log(x_{\max}/x_{\min})}. \tag{1.31}$$

This is also sometimes called *Jeffreys prior*, as a special case of Sec. 1.4.3.

Box 1.5: Number density of stars... log-uniform prior

Here, we choose to use a log-uniform prior on the density parameter, $\pi(S) \propto S^{-1}$. This requires that we choose a non-zero value for the lower limit of the prior range; we choose $S_{\rm min} = 1\,{\rm deg}^{-2}$. (Actually, this choice turns out to be a little bit too large; see posterior in Fig. 1.2).

$$\pi(S) = \frac{\mathbb{1}_{(S_{\rm min},S_{\rm max})}(S)}{S\log(S_{\rm max}/S_{\rm min})} \tag{i}$$

Using the likelihood from Box 1.3, Bayes' theorem gives the posterior as

$$P(S|n) = \frac{A(AS)^{n-1}\exp(-AS)\mathbb{1}_{(S_{\rm min},S_{\rm max})}(S)}{n!\log\left(\frac{S_{\rm max}}{S_{\rm min}}\right)Z}, \tag{ii}$$

where the evidence (integral evaluated numerically) is

$$Z = \int_{S_{\rm min}}^{S_{\rm max}} {\rm d}S\,\frac{A(AS)^{n-1}\exp(-AS)}{n!\log\left(\frac{S_{\rm max}}{S_{\rm min}}\right)} \approx 0.067. \tag{iii}$$

1.4.3 Ignorance Priors (Jeffreys Prior)

Suppose person A has a model with parameter α and likelihood $f(x|\alpha)$. Person A wants to perform a Bayesian analysis with an uninformative prior (or ignorance prior). It might initially be tempting to use either the uniform or log-uniform prior described above; but we will show that in general this is not the correct choice.

Person B is analysing the same data and using an equivalent model that only differs in its choice of parameterisation; person B's parameter is $\beta = \phi(\alpha)$. Both α and β are equally valid parameters and are related by an invertible transformation ϕ. A change of parameter should have no affect on our inference.

Person B's likelihood $g(x|\beta)$ is related to person A's by $f(x|\alpha) = g(x|\phi(\alpha))$. Person B also wants to perform a Bayesian analysis using an uninformative prior.

Suppose both person A and person B both choose to use a uniform prior on their chosen

parameters. This leads to an inconsistency. Transforming person A's uniform prior on α gives a induced density on β of $|\partial\phi^{-1}/\partial\alpha|_{\phi^{-1}(\alpha)}|$; in disagreement with B's uniform prior. Likewise, transforming B's uniform prior on β gives a non-uniform density on α, in disagreement with A's uniform prior. This shows that, in general, using a uniform prior on a model parameter is not an uninformative, or ignorance, prior.

The solution is for both persons A and B to use a Jeffreys prior on their respective parameters. The *Jeffreys prior* is defined to have a density that is proportional to the square root of the Fisher information:

$$\pi_\alpha(\alpha) \propto \sqrt{\mathcal{I}(\alpha)} \tag{1.32}$$

$$\pi_\beta(\beta) \propto \sqrt{\mathcal{I}(\beta)}. \tag{1.33}$$

These choices are consistent because if we now try transforming person A's prior the induced density on β matches person B's prior. (This can be shown using the transformation property of the Fisher information in Eq. 1.23.) Likewise, transforming B's prior gives an induced density on α that matches A's prior.

Jeffreys prior is a non-informative prior distribution for a parameter space. If there are more the one parameter, the multi-dimensional generalisation of Jeffreys prior is to use the square root of the determinant of the Fisher information matrix.

On drawback of Jeffreys priors is that they are often improper.

Exercise 1.2: Jeffreys Prior for translation and scale parameters
Consider the following Gaussian likelihood for a single data point $x \in \mathbb{R}$,

$$\mathcal{L}(x|\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), \tag{i}$$

where the mean, μ, and standard deviation, σ, are model parameters. Show that the Jeffreys priors for the mean and the standard deviation are given by the uniform and log-uniform distributions in Eqs. 1.27 and 1.30 respectively.

1.4.4 Conjugate Priors

In some special cases it is possible to solve to whole inference problem (including finding the evidence) analytically.

Suppose we have some simple analytic expression for the likelihood in a particular problem;

$$\text{data}|\text{parameters} \sim \mathcal{L}. \tag{1.34}$$

It is sometimes possible to find a known family of probability distributions (perhaps parametrised by one or more free parameters α) specifically tailored to this likelihood which have the property that if we use it as a prior when applying Bayes' theorem the posterior is is the same family of distributions (usually with some different parameters α').

$$\text{parameters} \sim \text{NiceDistribution}(\alpha) \qquad \text{(Prior)} \tag{1.35}$$

$$\text{parameters}|\text{data} \sim \text{SameNiceDistribution}(\alpha') \qquad \text{(Posterior)} \tag{1.36}$$

The role of the likelihood is to update our state of knowledge in going from the prior to the posterior. In the special case of a conjugate prior this corresponds to updating the parameters of our distribution according to $\alpha \to \alpha'$.

Box 1.6: Number density of stars... gamma prior

In this problem our likelihood is the Poisson distribution. The gamma distribution is the conjugate prior to the Poisson likelihood in the following sense;

$$\text{If } S \sim \text{Gamma}(k, \theta)\,, \qquad \text{(Prior)} \qquad \text{(i)}$$

$$\text{and } n|S \sim \text{Poisson}(AS)\,, \qquad \text{(Likelihood)} \qquad \text{(ii)}$$

$$\text{then } S|n \sim \text{Gamma}\left(k+n, \frac{\theta}{A\theta+1}\right)\,. \qquad \text{(Posterior)} \qquad \text{(iii)}$$

We will now prove this relationship using Bayes' theorem.

We will use gamma distribution (with *shape parameter* $k = 2$ and *scale parameter*

$\theta = 4\,\mathrm{deg}^{-2}$) as a prior on S. We can denote this is in a couple of different ways:

$$S \sim \mathrm{Gamma}(k, \theta), \tag{iv}$$

$$\pi(S) = \begin{cases} \frac{S^{k-1}\exp(-S/\theta)}{\Gamma(k)\theta^k} & \text{if } S > 0 \\ 0 & \text{else} \end{cases}. \tag{v}$$

Using the likelihood from Box 1.3, Bayes' theorem gives the posterior as

$$P(S|n) = \frac{A^n S^{k+n-1}\exp(-[A\theta+1]S/\theta)}{\Gamma(k)\theta^k n! Z} \quad \text{for } S > 0. \tag{vi}$$

However, we recognise the S dependence Eq.vi as that another gamma distribution for the parameter S with updated shape and scale parameters $k' = k + n$ and $\theta' = \theta/(A\theta + 1)$ respectively. Therefore, we must have

$$P(S|n) = \frac{S^{k+n-1}\exp(-[A\theta+1]S/\theta)}{\Gamma(k+n)\left(\frac{\theta}{A\theta+1}\right)^{k+n}} \quad \text{for } S > 0, \tag{vii}$$

$$\text{Or } S|n \sim \mathrm{Gamma}(k'|\theta'). \tag{viii}$$

As a bonus the conjugate prior also gives us an exact expression for the evidence. We have two expressions for the posterior in eqs. vi and vii, comparing these gives

$$Z = \frac{\Gamma(k+n)(A\theta)^n}{\Gamma(k)(A\theta+1)^{k+n}n!} \approx 0.079. \tag{ix}$$

For most realistic problems it is not possible to find a conjugate prior and it is necessary to resort to the numerical techniques described in the next chapter.

A conjugate prior is mainly an algebraic convenience; it allows us to find a closed-form expression for the posterior (and the evidence). Additionally, because they lead to a such a simple rule for updating parameters, conjugate priors also help to build intuition for the iterative process of Bayesian inference where the likelihood acts to update our prior state of knowledge.

1.4.5 Different Priors, Different Posteriors

In the above examples, we have now obtained posteriors from three different priors for the number density of stars from our survey: in Boxes 1.4 to 1.6 we have obtained different expressions for the posterior $P(S|n)$ (and the evidence $Z = P(n)$) using a uniform, a log-uniform, and a gamma prior.

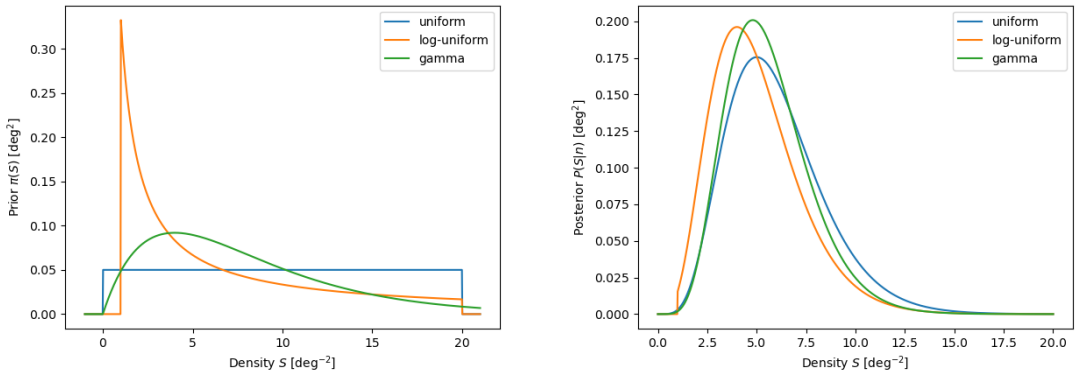


Figure 1.2: Comparison of the three different priors (left) and posteriors (right) on the density parameter S used in the Poisson process example in Boxes 1.3 to 1.6.

Which is right? In fact, can't we make the posterior anything we like simply by a suitable choice of prior?

Firstly - It's not really a matter of right or wrong; these are three reasonable posteriors that follow from 3 reasonable choices of the prior. Secondly - Yes, but this misses the point; we are supposed to choose a reasonable prior ahead of time and then work out the consequences.

Figure 1.2 shows the 3 different posterior distributions we have obtained so far. It takes some getting used to that we can have such different answers to a seemingly simple question as "what is the density?" that are all correct. In practice however, this is usually not an issue for the following reason. If our data is good then our freedom in the choice of a (reasonable) prior becomes unimportant. For example, suppose that instead of measuring $n = 5$ stars in $A = 1$ square degree, we had instead surveyed a larger area and measured $n' = 50$ stars in $A' = 10$ square degrees. In this case the posteriors would look like Fig 1.3. The different priors have a much smaller effect in this case.

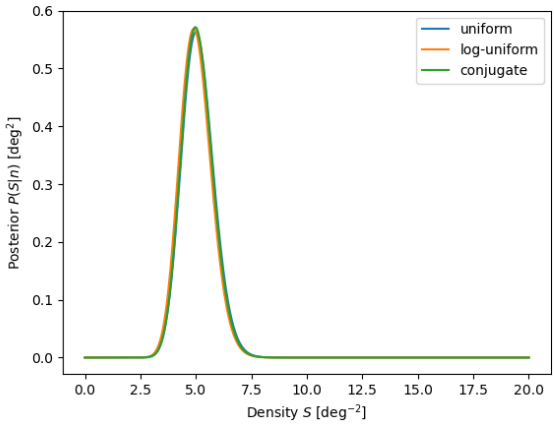


Figure 1.3: Comparison of the three different posteriors on the density parameter S for a survey with $n' = 50$ and $A' = 10\,\mathrm{deg}^2$. With this more informative likelihood function, the effect of the different priors is greatly reduced compared to that seen in Fig. 1.2.

The better our data the more constraining our likelihood and hence the smaller the effect on the posterior of a small change in the prior.

This argument can be turned around. If we ever find ourselves in a situation where making different (reasonable) choices of the prior has a significant effect on our conclusions then this suggests that likelihood doesn't contain much information about what we are trying to measure.

1.5 Iterative Bayesian Inference

Remember, the general principle is that for each experiment the prior should be chosen to describe our state of knowledge before performing the experiment. The likelihood (used in Bayes' theorem) then updates this knowledge and the posterior describes our new state of knowledge after performing the experiment.

If we now perform another experiment aimed at learning more about the same quantity, it is natural to use the posterior from the previous experiment as a prior. It's the same distribution, we're only changing what we call it; the person doing the inference for the first experiment called it the posterior, but the person doing the inference for the second

experiment calls it the prior.

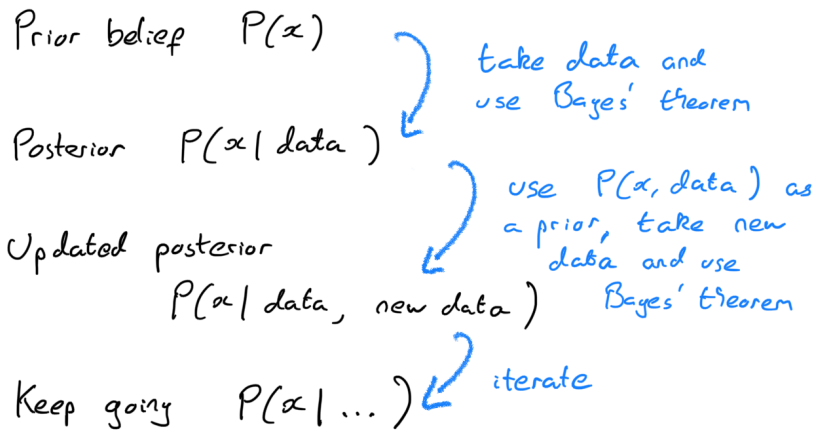


Figure 1.4: A metaphor for the scientific method?

Box 1.7: Number density of stars... multiple surveys

In our first survey (Box 1.3) we found $n_1 = 5$ stars in $A_1 = 1$ square degree. Using the gamma prior ($S \sim \text{Gamma}(k, \theta)$; Box 1.6) we made a measurement of the density and obtained the posterior, $S|n_1 \sim \text{Gamma}(k + n_1, \theta/[A_1\theta + 1])$.

Suppose we now decide that this measurement isn't precise enough and we decide to perform a second survey to improve our measurement; we find $n_2 = 6$ stars in a (different) area $A_2 = 1$ square degree.

We can use the posterior distribution from the first survey as a prior for the second survey. After all, this is the distribution that properly reflects our state of knowledge about S *before* the second survey.

$$S|n_1 \sim \text{Gamma}\left(k + n_1, \frac{\theta}{A_1\theta + 1}\right) \qquad \text{(New Prior)} \qquad \text{(i)}$$

Our likelihood hasn't changed, it is still the Poisson distribution.

$$n_2|S \sim \text{Poisson}(A_2 S) \qquad \text{(Likelihood for Survey 2)} \qquad \text{(ii)}$$

We now use Bayes' theorem to find the new posterior. But we have already solved this problem in Box 1.6 just with some quantities relabelled.

$$S|n_1, n_2 \sim \text{Gamma}\left(k + n_1 + n_2, \frac{\theta}{(A_1+A_2)\theta + 1}\right) \qquad \text{(New Posterior)} \qquad \text{(iii)}$$

We are using the fact that the results of our two surveys are independent; i.e. that n_2 doesn't depend on n_1 in Eq. ii.

At each stage the likelihood acts to update our prior. This is reflected by updating the parameters of the gamma distribution according the rule found in Box 1.6:

$$(k, \theta) \rightarrow \left(k + n_1, \theta' = \frac{\theta}{A_1\theta + 1}\right) \rightarrow \left(k + n_1 + n_2, \theta'' = \frac{\theta'}{A_1\theta' + 1} = \frac{\theta}{(A_1 + A_2)\theta + 1}\right).$$

Reassuringly, the new posterior in Eq. iii is the same one that we would have obtained if we had performed the Bayesian inference in a single step for a single survey of both areas ($A = A_1 + A_2$) that found the same $n = n_1 + n_2$ stars.

Box 1.8: Number density of stars... selection effects (question)

Consider again our survey of an area A_1 that found n_1 stars above a certain threshold magnitude.

Suppose we now learn of a second survey of a larger area $A_2 > A_1$ of sky that found n_2 stars. However, the second survey uses a less sensitive instrument. A star that would have been found in the first survey is only found by the second survey with probability p.

If our goal is to learn as much as we can about S, which survey is better?

One way of answering the question posed in Box. 1.8 would be to perform a Bayesian inference for both surveys (separately) and compare the widths of the posteriors on S (e.g. quantified using the standard deviation). The survey with the narrower posterior is best. However, realistic Bayesian inferences are often expensive and time consuming. It would be nice to be able to answer questions of this type without having to explicitly obtain the posteriors.

The Fisher information introduced in Sec. 1.3.1 is measurement of the amount of information that some data carries about an unknown parameter. It can be used for answering questions of the type posed in Box. 1.8. This is demonstrated in Box. 1.9

Box 1.9: Number density of stars... selection effects (answer)

Continuing from Box. 1.8.

For the first survey, the likelihood is a Poisson distribution with rate $A_1 S$;

$$n_1|S \sim \text{Poisson}(A_1 S) \quad \Rightarrow \quad \mathcal{L}(n_1|S) = \frac{(A_1 S)^{n_1} \exp(-A_1 S)}{n_1!} \tag{i}$$

For this likelihood, the score (defined in Eq. 1.8) is $\mathcal{S}_1(S) = (n_1/S) - A_1$. The definition of the Fisher information (in either Eq. 1.18 or Eq. 1.19) involves an integral with respect to a continuous data variable, x. In this problem the data is the discrete number of stars, n_1. Therefore, the Fisher information will involve a sum over n_1. We can compute the Fisher information from either of the definitions in Eqs. 1.18 or 1.19. Working from Eq. 1.18, we have

$$\mathcal{I}_1(S) = \sum_{n_1=0}^{\infty} \mathcal{L}(n_1|S) \left(\frac{n_1}{S} - A_1\right)^2, \tag{ii}$$

$$= \sum_{n_1=0}^{\infty} \mathcal{L}(n_1|S) \left(\frac{n_1^2}{S^2} - \frac{2n_1 A_1}{S} + A_1^2\right), \tag{iii}$$

$$= \left(\left[\frac{A_1}{S} + A_1^2\right] - 2A_1^2 + A_1^2\right), \tag{iv}$$

$$= \frac{A_1}{S}. \tag{v}$$

In going from the second to the third line we have used the fact that both the mean and the variance of the Poisson distribution are equal to the rate parameter.

Hopefully, it makes sense that the Fisher information is proportional to A_1; the bigger the area we survey the more information we get.

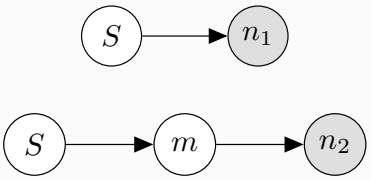
For the second survey, the likelihood is more complicated. The actual number of stars m in the surveyed area is a Poisson random variable with rate parameter $A_2 S$. But the number of stars we actually detect is a Binomial random variable

with m trials and probability p of success for each trial.

$$m|S \sim \text{Poisson}(A_2 S) \tag{vi}$$
$$n_2|m \sim \text{Binomial}(m, p) \tag{vii}$$

The variable m is an example of what is sometime called a *latent variable*.

Probabilistic Graphical Models (PGMs) can be a helpful way of visualising the conditional dependencies of observations (i.e. data) and model variables in more complicated models. PGMs will be discussed in more detail in chapter 4. However, here are the PGMs for the first and second surveys.



Using the law of total probability, the likelihood for the second survey is

$$\mathcal{L}(n_2|S) = \sum_{m=n_2}^{\infty} P(n_2|m)P(m|S), \tag{viii}$$
$$= \sum_{m=n_2}^{\infty} {}^{m}C_{n_2} p^{n_2}(1-p)^{m-n_2} \cdot \frac{(A_2 S)^m \exp(-A_2 S)}{m!}, \tag{ix}$$
$$= \frac{(A_2 p S)^{n_2} \exp(-A_2 S)}{n_2!} \sum_{\mu=0}^{\infty} \frac{\big((1-p)A_2 S\big)^{\mu}}{\mu!}, \tag{x}$$
$$= \frac{(A_2 p_2 S)^{n_2} \exp(-A_2 p S)}{n_2!}. \tag{xi}$$

We recognise this as another Poisson distribution with rate parameter $A_2 p S$. Therefore, reusing the results from above, the score for the second survey is $\mathcal{S}_2(S) = (n_2/S) - A_2$ and the Fisher information is

$$\mathcal{I}_2(S) = \frac{A_2 p}{S}. \tag{xii}$$

The second survey contains more information than the first survey about the density parameter if $A_2 p > A_1$.

1.6 Posterior Estimates and Summary Statistics

Having obtained the Bayesian posterior probability distribution $P(\theta|x)$ what can we do with it?

1.6.1 Marginal distributions

It is often the case that only a subset some of the model parameters are of interest. The other parameters, those whose values we are not interested in, are known as "nuisance" parameters. For example, when analysing a signal in an unknown noise background: there are parameters describing the signal and nuisance parameters describing the noise.

We need to account for the nuisance parameters, but we would also like to remove them from our final result. This can be achieved by integrating out, or *marginalising*, these parameters from the posterior distribution.

We can construct the *1-dimensional marginal posteriors* for each parameter. For example, for the first parameter θ_1 we define

$$P(\theta_1|x) = \int \mathrm{d}\theta_2 \int \mathrm{d}\theta_3 \ldots \int \mathrm{d}\theta_n P(\theta_\mu|x)\,. \tag{1.37}$$

We can construct the *2-dimensional marginal posteriors* for each pair of parameters. For example, for the first two parameters θ_1 and θ_2 we define

$$P(\theta_1, \theta_2|x) = \int \mathrm{d}\theta_3 \int \mathrm{d}\theta_4 \ldots \int \mathrm{d}\theta_n P(\theta_\mu|x)\,. \tag{1.38}$$

It is common to arrange and visualise all possible 1 and 2 dimensional marginal posteriors in a *corner plot*, see Fig. 1.5.

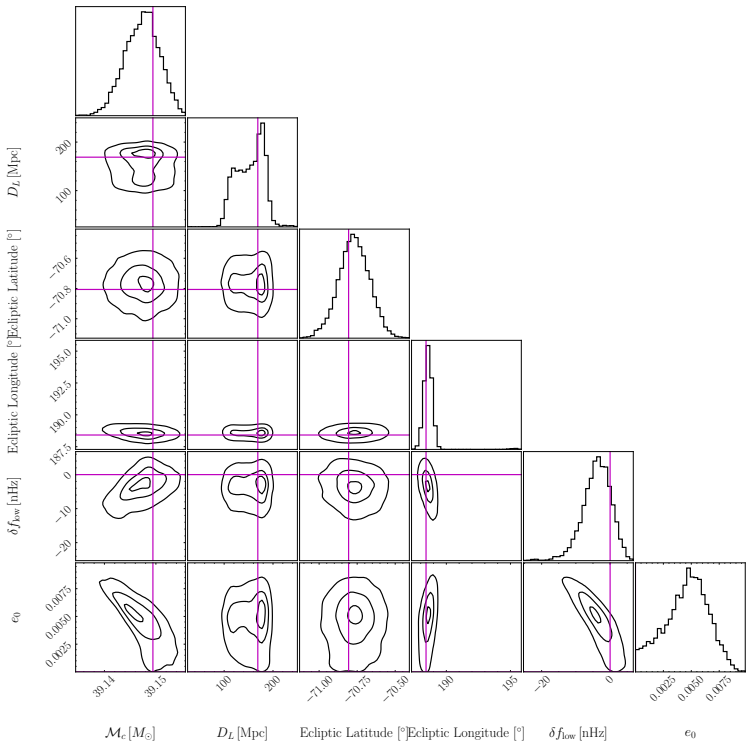


Figure 1.5: An example corner plot.

In problems with more parameters, say $m+n$ parameters $\{\theta_1, \theta_2, \ldots, \theta_n, \phi_1, \phi_2, \ldots, \phi_m\}$, you can marginalise over any number m of the parameters in the posterior as you want to leave a *n-dimensional marginal posterior*;

$$P(\theta_1, \theta_2, \ldots, \theta_n|x) = \int \mathrm{d}^m\phi \; P(\theta_1, \theta_2, \ldots, \theta_n, \phi_1, \phi_2, \ldots, \phi_m|x)\,. \tag{1.39}$$

If we marginalise (integrate) out *all* of the parameters in the posterior distribution we are just left with a constant,

$$\int \mathrm{d}^n\theta_\mu \; P(\theta_\mu|x) = 1\,. \tag{1.40}$$

If the posterior distribution is correctly normalised, then this constant is 1, by definition. However, if we are working with the unnormalised posterior $P(\theta_\mu|x) \propto \mathcal{L}(x|\theta_\mu)\pi(\theta_\mu)$, then this normalising constant is the Bayesian evidence, Z, defined in Eq. 1.6. For this reason, you will sometimes see the *evidence* referred to by the alternative name *marginal likelihood*.

1.6.2 Point Estimates

The posterior is the complete description of our state of knowledge (after the experiment) about the model parameters, θ_μ. However, it is often desirable to be able to condense the information contained in the posterior into just a few numbers, or summary statistics.

If the posterior is dominated by a single peak in the parameter space, then it might make sense to report a single value for our best guess of the model parameter values. A number of these *point estimates* are possible:

1. The posterior mean; e.g. for the first parameter

$$\langle \theta_1 \rangle = \int \mathrm{d}\theta_1 \; \theta_1 P(\theta_1|x) = \int \mathrm{d}^n\theta_\mu \; \theta_1 P(\theta_\mu|x). \tag{1.41}$$

   This can be calculated from either the full n-dimensional posterior or the 1-dimensional marginal posterior distribution.

2. The posterior mode, known as the *maximum a posteriori* (MAP) estimate $\theta_\mu^{\mathrm{MAP}}$ satisfies

$$\frac{\partial P(\theta_\mu|x)}{\partial \theta_\mu}\Big|_{\theta_\mu=\theta_\mu^{\mathrm{MAP}}} = 0. \tag{1.42}$$

   This must be calculated from the full n-dimensional posterior; note, that the marginal posterior distributions are in general NOT peaked at the MAP parameters.

3. The median of each 1-dimensional marginal posterior; e.g. for the first parameter this satisfies

$$\int_{-\infty}^{\bar{\theta}_1} \mathrm{d}\theta_1 \; P(\theta_1|x) = \frac{1}{2}. \tag{1.43}$$

   This must be calculated from the 1-dimensional marginal posterior distributions.

1.6.3 Credible Intervals

It is worth repeating, the posterior is the complete description of our state of knowledge (after the experiment) about the model parameters, θ_μ, including the uncertainties on these parameters. It is often desirable to be able to condense this information into some kind of "error bar".

A *credible interval* is an range of values within which the model parameters lies with a particular probability level. These are used in a similar to confidence intervals in frequentist inference. In more than 1 dimension we use the alternative name *credible region*.

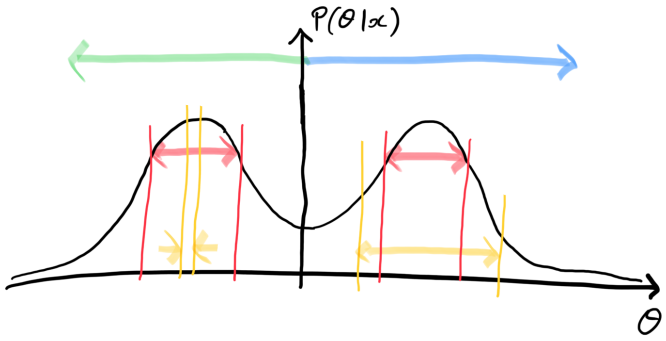


Figure 1.6: The credible interval is not unique. For this distribution the 4 intervals shown in blue, green, red and yellow are all possible 50% credible intervals.

Even when we have chosen a level (e.g. the 50% or 90% credible intervals are commonly used) the credible interval is not unique (see Fig. 1.6).

Several systematic ways of constructing credible intervals that contain a fraction α of the posterior probability are possible:

1. In 1 dimension, choose the *narrowest interval* (or collection of intervals) I that contains a probability mass α.

   This I is *not* invariant under reparametrisation. This flaw can be fixed by changing our notion of width from "narrowest" (i.e. the smallest $\theta_{\max} - \theta_{\min}$) to use the prior probability measure; i.e. the interval that contains the smallest prior probability mass.

   In multiple dimensions, this can be generalised to the region with the smallest volume that contains a probability mass α.

2. In 1 dimension, choose the *highest density interval* which is the interval (or collection of intervals) $I(f_\alpha) = \{\theta : P(\theta|x) > f_\alpha\}$, where f_α is the largest constant such that $\text{Prob}\big(\theta \in I(f_\alpha)\big) \geq \alpha$.

   In multiple dimensions, this can be generalised using *iso-probability contours* in the sample space to bound the *highest density credible region*; i.e. the region $R(f_\alpha)$ defined in the same way as $I(f_\alpha)$.

3. In 1 dimension, choose the *equal-tailed interval* $\theta_{\min} < \theta < \theta_{\max}$ with $\text{Prob}(\theta < \theta_{\min}) = \text{Prob}(\theta > \theta_{\max}) = \alpha/2$.

   In multiple dimensions, this can be applied to each parameter individually using the 1-dimensional marginalised posterior distributions.

1.6.4 Computing summary statistics

If we are given a the posterior as a function of the model parameters, i.e. $f(\theta) \propto P(\theta|x)$, the summary statistics discussed in Secs. 1.6.2 and 1.6.3 can all be calculated, at least in principle. But doing so can be difficult, especially in multiple dimensions. For example, finding means, medians and credible intervals/regions all require integrating over the high-dimensional parameter space. In practice, these summary statistics are not normally calculated directly from $f(\theta)$, but from *stochastic samples* instead. This is the subject of the next chapter 2.

Chapter 2

Bayesian Computation

This section describes computational methods that can be used to study probability distributions by generating *stochastic samples* from these distributions. These methods can be used to learn about any probability distribution, however our main concern will be using them to study the posterior distributions that arise when performing Bayesian inference.

2.1 Why Monte Carlo?

Given a *probability distribution* P (sometimes called the *target distribution*) a set of *stochastic samples* are n independent (pseudo)random variables draw from this distribution;

$$x \overset{\text{iid}}{\sim} P, \quad \text{for } i = 0, 1, \ldots n-1. \tag{2.1}$$

Stochastic samples can be used to answer virtually any question one might have about the target distribution P. They are commonly used in statistical inference, especially Bayesian inference. They can be used for a variety of purposes; for example:

- approximating the full d-dimensional distribution – e.g. by generating a d-dimensional dimensional histogram or kernel density estimate (KDE) of the samples;
- approximating and/or visualising the marginal distributions – e.g. by generating a 1- or 2-dimensional histograms or KDEs of the marginal distributions of P for use in a *corner plot*;

- computing *credible intervals* (or *credible regions*) – if P is a posterior in a Bayesian analysis then stochastic samples can be used to quantify measurement uncertainties by finding intervals within which parameters fall with a particular probability;

- approximating integrals of the form $I = \int \mathrm{d}x\ \varphi(x)P(x)$ – this can be done via the Monte-Carlo sum $S_n = \frac{1}{n}\sum_{i=0}^{n-1} \varphi(x_i)$ which satisfies $S_n \to I$ as $n \to \infty$.

Note that in all the above applications the stochastic samples are being used to calculate a deterministic quantity. This is actually a reasonable definition of a *Monte-Carlo method*.

The final bullet point, Monte Carlo integration, is the most important application. It's worth emphasising the draw backs of Monte Carlo integration[1]. The error on the estimator S_n for the integral I scales as $\sim 1/\sqrt{n}$. This is actually *not* very impressive; in contrast, deterministic quadrature methods exist with *exponential* convergence with the number of integration nodes per dimension (at least for smooth integrands). However, in d dimensions the number of integration nodes required increases exponentially with d; e.g. a regular cubic grid with 10 points along each dimension contains 10^d points. In contrast, Monte Carlo methods converge as $\sim 1/\sqrt{n}$ in any number of dimensions. From this point of view, Monte Carlo methods are bad, but for high-dimensional problems there may not be a better alternative.

A wide variety of numerical methods have been developed to tackle the problem of stochastic sampling. We will first recall a few elementary sampling algorithms (such as transform and rejection sampling, Secs. 2.2.1 and 2.2.2 respectively) and describe why these can't be applied to arbitrary target distributions. Then, we will go on to describe a variety of Markov chain Monte Carlo sampling algorithms that can be applied to any target distribution.

2.2 Stochastic Sampling

Given a *probability distribution* P (over a *sample space* $\mathcal{X}$), a *stochastic sampling* is the problem of constructing a *random sample* $x \in \mathcal{X}$ such that $x \sim P$. (We focus only on *continuous* probability distributions here, although many of the methods we will discuss can be adapted to work with discrete distributions as well.) The distribution P is sometimes called the *target distribution*. Typically, we will require a large number n of *independent*

[1] See the nice discussion in Sokal (1997) "Monte Carlo Methods in Statistical Mechanics: Foundations and New Algorithms", doi:10.1007/978-1-4899-0319-8_6.

samples for any realistic application (see Eq. 2.1). Efficiently generating many such samples from a general target is a hard problem, especially when $\mathcal{X}$ is high dimensional.

In this section we will first describe a few elementary methods useful for generating stochastic samples from simple (usually low-dimensional) distributions. We will see some of the difficulties that arise when trying to apply these methods to more complex distributions. Then we will turn our attention to Markov Chain Monte Carlo (MCMC) methods which can be applied to high-dimensional problems.

We will discuss several varieties of MCMC algorithm. Our emphasis will be on the general features of the algorithms, rather than on the many problem-specific variants. Providing a complete list of sampling methods is impossible because, as we will see, there is almost limitless potential for designing new variations and/or combinations of these algorithm tailored for specific problems.

2.2.1 Transform Sampling

For some simple distributions a function can be found that maps from a known distribution, Q, to the target distribution, P.

Given a random variable $x \sim Q$ and an invertible transformation $f : \mathbb{R}^d \to \mathbb{R}^d$ the random variable $y = f(x)$ is distributed as $y \sim P$, where

$$P(y) = \left|\frac{\partial f}{\partial x}\right|^{-1} P(x). \tag{2.2}$$

The $d \times d$ matrix $\frac{\partial f}{\partial x}$ is the Jacobian of the transformation.

The challenge is to design a transformation f such that y follows the desired target distribution. Unfortunately, for an arbitrary target distribution in multiple dimensions, this is usually not possible to do excatly.

However, in 1 dimension the problem can be solved using the inverse of the CDF.

The *transform sampling* method, or *inverse CDF sampling* method, starts with a uniform random variable $u \sim \mathcal{U}$ [2] and applies a specific transformation $x \equiv F^{-1}(u)$ designed such that the resulting random variable follows the target distribution, $x \sim P$.

[2]Notation: $u \sim \mathcal{U}(a, b)$ denotes a random variable from the uniform distribution on the interval (a, b) with PDF $P(u) = \mathbb{1}_{(a,b)(u)}$. The shorthand $u \sim \mathcal{U}$ denotes a random variable from the uniform distribution on the interval $(0, 1)$.

The *cumulative distribution function* (CDF) $F(x)$ is defined in terms of the PDF via

$$F(x) = \int_{-\infty}^{x} \mathrm{d}x' \; P(x'). \tag{2.3}$$

Because P is a non-negative, normalised PDF, the CDF is a monotonically increasing function satisfying $\lim_{x\to-\infty} F(x) = 0$ and $\lim_{x\to-\infty} F(x) = 1$.

If F is continuous, then it has a unique inverse, $F^{-1}(F(x)) = x$. The function F^{-1} is known variously as the *inverse CDF*, the *percent-point function* (PPF), or *quantile function*.

Given a sample $u \sim \mathcal{U}$, the inverse CDF can be used to produce a sample $x \sim P$.

Theorem 2.2.1. *If $u \sim \mathcal{U}$, then $x \equiv F^{-1}(u)$ satisfies $x \sim P$.*

Proof. Because $\mathcal{U}$ has support in the interval (0,1) and $0 \leq F(x) \leq 1$, it follows that the distribution of x has the same support as P. From its definition, the PDF of x is $P_{\mathcal{U}}(u)|\mathrm{d}u/\mathrm{d}x|$. Differentiating Eq. 2.3 gives $\mathrm{d}u/\mathrm{d}x = P(x)$, and using the fact that $P(u) = 1$ in the supported region, shows that this equals $P(x)$, as required. □

Transform sampling using the inverse CDF sampling is illustrated in Fig. 2.1.

Note, inverse CDF sampling requires finding the CDF (by integrating the PDF) and inverting it; in general, both of these operations might have to be done numerically.

Box 2.1: Using inverse CDF method to sample the standard normal (a.k.a. Gaussian) distribution

The target PDF is

$$P(x) = \frac{\exp\left(\frac{-1}{2}x^2\right)}{\sqrt{2\pi}}. \tag{i}$$

A random variable from this distribution is denoted $x \sim \mathcal{N}(0, 1)$.

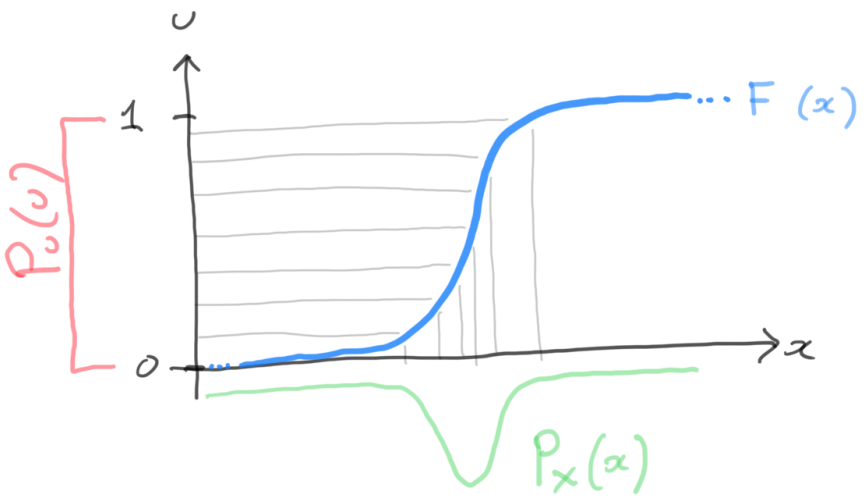


Figure 2.1: An illustration of the inverse CDF method for sampling from a univariate distribution. Applying the function F^{-1} maps a uniformly distributed random variable to a new random variable that follows the desired distribution, $x \sim P$.

The target CDF is

$$F(x) = \int_{-\infty}^{x} \mathrm{d}x' \; P(x') = \frac{1}{2}\left(1 + \mathrm{erf}\left[\frac{x}{\sqrt{2}}\right]\right), \tag{ii}$$

where the error function is defined as $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \mathrm{d}y \; \exp(-y^2)$.

The target inverse CDF is

$$F^{-1}(x) = \sqrt{2}\,\mathrm{erf}^{-1}(2x - 1), \tag{iii}$$

where the inverse error function satisfies $\mathrm{erf}^{-1}\big(\mathrm{erf}[x]\big) = x$.

Therefore, if $u \sim \mathcal{U}$, then $x = \sqrt{2}\,\mathrm{erf}^{-1}(2u - 1)$ is distributed as $x \sim \mathcal{N}(0, 1)$.

Samples from the normal distribution $\mathcal{N}(\mu, \sigma^2)$ can be obtained by applying the shift and scaling transformations $\mu + x\sigma$.

Box 2.2: Using inverse CDF method to sample the exponential distribution

The target distribution is the exponential distribution (with rate parameter $\lambda > 0$) which has PDF

$$P(x) = \begin{cases} \lambda \exp(-\lambda x) & \text{if } x > 0 \\ 0 & \text{else} \end{cases}. \tag{i}$$

A random variable from this distribution is denoted $x \sim \text{Exp}(\lambda)$.

The target CDF is

$$F(x) = \int_0^x \mathrm{d}x' \; P(x') = 1 - \exp(-\lambda x). \tag{ii}$$

The target inverse CDF is

$$F^{-1}(x) = \frac{-\log(1-x)}{\lambda}. \tag{iii}$$

Therefore, if $u \sim \mathcal{U}$, then $x = -\log(1-u)/\lambda$ is distributed as required. However, $(1-u)$ is distributed identically to u (this follows by symmetry), so we may instead use the simpler expression $x = -\log(u)/\lambda$.

2.2.2 Rejection Sampling

In one dimension, the *rejection sampling method* aims to sample a 1D random variable by first uniformly sampling in 2D and keeping only the first coordinate of points that satisfy a certain condition. It is also commonly called the *accept-reject algorithm* and is sometimes attributed to John von Neumann.

The rejection method uses a *proposal distribution* Q. We assume that we are already able to sample efficiently from Q (e.g. by using the inverse CDF method described in 2.2.1). We define a constant $1 < M < \infty$ such that $P(x) \leq MQ(x)$ for all x. Note that this requires that Q must have support everywhere that P has support.

The rejection sampling algorithm proceeds as follows. Random samples are obtained from

both the proposal Q and uniform $\mathcal{U}$ distributions. (The ordered pair (x, Mu) can be thought of as a point in two dimensions.) If a certain acceptance condition is met then x is accepted as a sample from P, otherwise x is rejected and we try again.

Algorithm 2.1 Rejection Sampling

1: $x \sim Q$ ▷ Sample the proposal
2: $u \sim \mathcal{U}$
3: **if** $Mu < P(x)/Q(x)$ **then**
4: **return** x ▷ Accept
5: **else**
6: **go to line** 1 ▷ Reject
7: **end if**

Theorem 2.2.2. *The random x produced from Alg. 2.1 is distributed as $x \sim P$.*

Proof. It will suffice to show the PDF of the conditional distribution of x given $Mu < P(x)/Q(x)$ produced by the algorithm is $P(x)$. Bayes' theorem gives

$$\text{Prob}\left(x = x \middle| u < \frac{P(x)}{MQ(x)}\right) = \text{Prob}\left(u < \frac{P(x)}{MQ(x)} \middle| x = x\right) \frac{\text{Prob}(x = x)}{\text{Prob}\left(u < \frac{P(x)}{MQ(x)}\right)}. \tag{2.4}$$

Using $x \sim Q$ and $u \sim \mathcal{U}$, the factors on the right-hand side can be written

$$\text{Prob}\left(u < \frac{P(x)}{MQ(x)} \middle| x = x\right) = \frac{P(x)}{MQ(x)}, \tag{2.5}$$

$$\text{Prob}(x = x) = Q(x), \tag{2.6}$$

$$\text{and Prob}\left(u < \frac{P(x)}{MQ(x)}\right) = \int_{-\infty}^{\infty} \mathrm{d}x \, \frac{P(x)}{MQ(x)} Q(x) = \frac{1}{M}. \tag{2.7}$$

Substituting Eqs. 2.5, 2.6 and 2.7 back into Eq. 2.4 gives the result,

$$\text{Prob}\left(x = x \middle| u < \frac{P(x)}{MQ(x)}\right) = P(x). \tag{2.8}$$

□

An illustrated of the rejection sampling method is shown in Fig. 2.2.

The rejection algorithm takes an average of M iterations per sample. For best performance the proposal Q should be as similar as possible to the target P so that M can be chosen

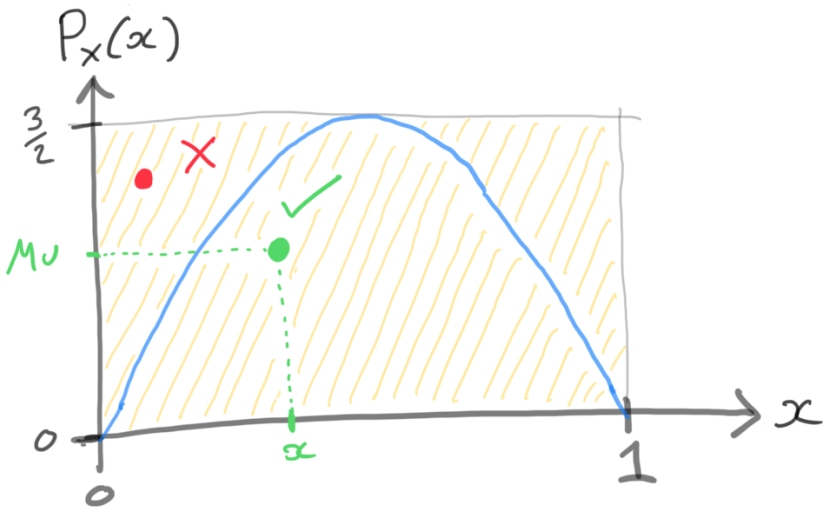


Figure 2.2: An illustration of rejection sampling for the beta distribution $x \sim \text{Beta}(\alpha = 2, \beta = 2)$ with $P(x) = 6x(1-x)$ (blue curve). The proposal distribution is chosen to be the uniform distribution, $Q = \mathcal{U}$. The smallest possible choice for the constant is $M = 3/2$ which corresponds to drawing points uniformly in the yellow shaded region and keeping/rejecting the x-coordinates of the green/red points that lie below/above the line.

to be as small as possible (thereby avoiding "wasted space"). Ideally, Q should also be chosen such that it is easy to sample from efficiently.

In practice, the difficult part of implementing rejection sampling is finding a proposal distribution that is (i) easy to sample from and (ii) encompasses the target distribution without lots of wasted space allowing us to find as small an M as possible. Choosing a good proposal distribution is not always easy (see, for example, example 2.3). To circumvent this, several variations of *adaptive rejection sampling* algorithms exists that aim to build a self-tuning proposal distribution that updates and improves itself each time a sample is rejected. These adaptive methods aim to improve the sampling efficiency in the limit of a large number of samples and remove the burden of choosing the proposal distribution from the user. These adaptive variations of the method are not discussed in detail here.

Rejection sampling can also be used for multivariate distributions. (Nothing in the above algorithm required P to be univariate.) However, in high dimensions the condition $P(x) \leq MQ(x)$ usually forces M to be so large that the method becomes inefficient, unless Q is an extremely good approximation to P. In fact, even finding a suitable M can be difficult

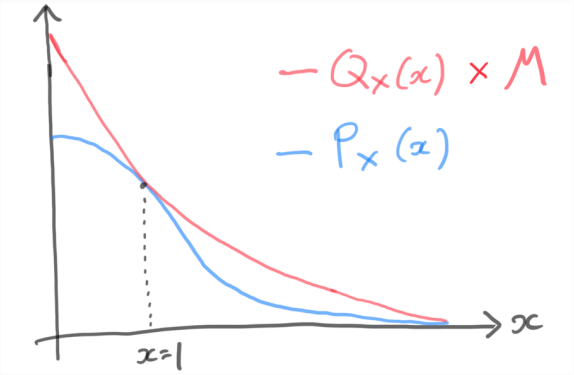
because we do not generally know in advance where the peaks of $P(x)$ are located or how tall they are. For these reasons, rejection sampling is usually only used for univariate distributions.

Box 2.3: Using the rejection method to sample the standard normal distribution

We have already seen how to sample the normal distribution with the inverse CDF method (Example 2.3) now we see how to do it using the rejection method.

We chose as our proposal Q the exponential distribution with unit rate parameter. We already know how to sample from Q (see Example 2.2). Because Q only has support for $x < 0$, we use the rejection method to sample from the normal distribution truncated to $x > 0$ and then make a random choice for the sign.

The ratio of the PDF of our target distribution to the PDF of our proposal distribution is $\exp(x - x^2/2)/\sqrt{2\pi}$. This ratio achieves its maximum of $\sqrt{2e/\pi}$ when $x = 1$. Therefore, we choose $M = \sqrt{2e/\pi}$. (See figure below.)



Step i: Generate two independent uniform random variables; $u_1, u_2 \overset{\text{iid}}{\sim} \mathcal{U}$.

Step ii: If $u_2 < \exp\left(\frac{-(1-x)^2}{2}\right)$ then let $|x| = -\log u_1$, else go back to *Step i.*

Step iii: Choose the sign. E.g. by generating another uniform random variable $u_3 \sim \mathcal{U}$ and if $u_3 < 1/2$, then let $x = |x|$, else let $x = -|x|$.

The resulting $x \sim \mathcal{N}(0, 1)$, as required.

2.2.3 Importance Sampling

Importance sampling is a method for learning about the target distribution P, using samples from another distribution, Q. The distribution Q is called the *proposal distribution*.

Importance sampling is useful when sampling from P is difficult, but when another distribution Q can be found that is (i) easier to sample from, and (ii) that is "close to" P. (The idea of closeness is made precise using the quantity ϵ below). It is necessary that Q has support everywhere that P has support.

The other stochastic sampling algorithms described up to this point aim to produced *equally-weighted* stochastic samples from the target distribution; i.e. $x_i \overset{\text{iid}}{\sim} P$, for $i = 0, 1, \ldots, n-1$. These samples can be used to approximate integrals of the form

$$I = \int \mathrm{d}x\ \varphi(x)P(x) \tag{2.9}$$

via the Monte-Carlo sum

$$S_n = \frac{1}{n}\sum_{i=0}^{n-1} \varphi(x_i). \tag{2.10}$$

The law of large numbers ensures that $S_n \to I$ as $n \to \infty$.

If we instead have samples from another distribution, called the proposal distribution; $x_i \overset{\text{iid}}{\sim} Q$, for $i = 0, 1, \ldots, n-1$. Then, defining the *sample weights* as

$$w_i = \frac{P(x_i)}{Q(x_i)}, \tag{2.11}$$

The *weighted samples* $\{(x_i, w_i) | i = 0, 1, \ldots, n-1\}$ can be used to approximate the integral.

Lemma 2.2.3. *If $x_i \overset{\text{iid}}{\sim} Q$ for $i = 0, 1, \ldots, n-1$, and the corresponding sample weights are defined as in Eq. 2.11, then*

$$\int \mathrm{d}x\ \varphi(x)P(x) \approx \frac{1}{n}\sum_{i=0}^{n-1} w_i\varphi(x_i). \tag{2.12}$$

Proof. Rewritte the integral as $I = \int \mathrm{d}x\ \psi(x)Q(x)$, where $\psi(x) = \varphi(x)\frac{P(x)}{Q(x)}$. Because $x_i \overset{\text{iid}}{\sim} Q$, Eq. 2.10 implies $I \approx \frac{1}{n}\sum_{i=0}^{n-1} \psi(x_i)$. The result follows from definition of ψ. □

Weighted samples can also be used to approximate or visualise the target distribution (or its marginal distributions; e.g. in a corner plot) using *weighted histograms* or *KDEs*.

Importance sampling works well when Q is "close to" P. A perfect proposal would lead to equal weights for all samples. In general, the weights differ and we define the *number of effective samples* as $n_{\text{eff}} = \left(\sum_{i=0}^{n-1} w_i\right)^2 / \sum_{i=0}^{n-1} w_i^2 < n$. This gives a natural measure of the quality of the proposal and the *efficiency* of the sampling, $\epsilon = n_{\text{eff}}/n$ [3].

There are similarities between importance sampling and rejection sampling (see Sec. 2.2.2). In importance sampling, instead of simply discarding bad samples they are instead down weighted.

Box 2.4: Prior reweighting

Importance sampling can be used to change the prior in a Bayesian analysis. Suppose we have already obtained samples $x_0, x_1, \ldots, x_{n-1} \overset{\text{iid}}{\sim} P_1(x|d)$ from the posterior $P_1(x|d) \propto \mathcal{L}(d|x)\pi_1(x)$ and that we now want to investigate the effect of a different prior $\pi_2(x)$ with corresponding posterior $P_2(x|d) \propto \mathcal{L}(d|x)\pi_2(x)$. The samples can be reused with weights given by the ratio of the priors; $\{(x_i, w_i = \pi_2(x_i)/\pi_1(x_i))|i = 0, 1, \ldots, n-1\}$ are weighted samples from P_2.

Exercise 2.1: Bayes factor from weights
Show that for the problem in Box. 2.4 the Bayesian evidence for the second prior can be estimated using

$$Z_2 = \int \mathrm{d}x\ \mathcal{L}(d|x)\pi_2(x) \approx \frac{1}{n}\sum_{i=0}^{n-1} w_i. \tag{i}$$

[3] Kong (1992) "A note on importance sampling using standardized weights", University of Chicago, Dept. of Statistics, Tech. Rep. 348, link.

2.2.4 Combining Random Variables

In Sec. 2.2.1 we saw how given a sample from one distribution we could apply a transformation to obtain a sample from another distribution. Another approach is to start with two or more samples from known distributions and combine them to form a new random variable.

In fact, several common distributions are *defined* using such combinations.

For an arbitrary target distribution, it is usually not possible to design a combination of simple random variables with the right distribution. Therefore, this method is generally only used for a limited number of simple target distributions.

Examples of distributions that can be defined as combinations of simpler distributions:

- *Gamma distribution.* If $x_1, x_2, \ldots, x_n \overset{\text{iid}}{\sim} \text{Exp}(\theta)$, then $\sum_{i=1}^{n} x_i \sim \text{Gamma}(n, \theta)$. (See Box 2.5.)
- *Chi-squared* distribution. If $x_1, x_2, \ldots, x_\nu \overset{\text{iid}}{\sim} \mathcal{N}(0, 1)$, then $\sum_{i=1}^{n} x_i^2 \sim \chi^2_\nu$. (See Box 2.6.)
- *Beta* distribution. If $x \sim \text{Gamma}(\alpha, \theta)$ and independently $y \sim \text{Gamma}(\beta, \theta)$, then $x/(x+y) \sim \text{Beta}(\alpha, \beta)$.

Box 2.5: The gamma distribution

The target distribution is the gamma distribution (with shape parameter $k > 0$ and scale parameter $\theta > 0$) which has PDF

$$P(x) = \begin{cases} \frac{x^{k-1}\exp(-x/\theta)}{\Gamma(k)\theta^k} & \text{if } x > 0 \\ 0 & \text{else} \end{cases}. \qquad \text{(i)}$$

A random variable from this distribution is denoted $x \sim \text{Gamma}(k, \theta)$.

An equivalent definition of the gamma distribution involves multiple independent exponentially distributed random variables; $x_1, x_2, \ldots, x_k \overset{\text{iid}}{\sim} \text{Exp}(\theta)$. The new

random variable formed from the sum $z \equiv \sum_{i=1}^{k} x_i$ is, by definition a gamma-distributed random variable; $z \sim \text{Gamma}(k, \theta)$.

We now show that these definitions are equivalent...

(i) Note that $\text{Gamma}(1, \theta) = \text{Exp}(\theta)$.

(ii) If $x \sim \text{Gamma}(k, \theta)$ and $x \sim \text{Gamma}(k', \theta)$, then $z \equiv x+y \sim \text{Gamma}(k+k', \theta)$. This can be shown by finding the PDF of the new random variable z.

$$P(z) = \int_{-\infty}^{\infty} \mathrm{d}u \; \text{Prob}(x = z - u)\text{Prob}(y = u) \tag{ii}$$

$$= \int_{0}^{z} \mathrm{d}u \; \left(\frac{[z-u]^{k-1} \exp(-[z-u]/\theta)}{\Gamma(k)\theta^k} \right) \left(\frac{u^{k'-1} \exp(-u/\theta)}{\Gamma(k')\theta^{k'}} \right). \tag{iii}$$

Making the change of variables $w = u/z$ gives

$$P(z) = \frac{w^{k+k'-1} \exp(-z/\theta)}{\Gamma(k)\Gamma(k')\theta^{k+k'}} \int_{0}^{1} \mathrm{d}w \; (1-w)^{k-1} u^{k'} \tag{iv}$$

$$\propto z^{k+k'-1} \exp(-z/\theta). \tag{v}$$

This is proportional to Eq. i with $k \to k+k'$. This shows that $z \sim \text{Gamma}(k+k', \theta)$.

(iii) If $x_1, \ldots, x_k \overset{\text{iid}}{\sim} \text{Exp}(\theta)$, then $\sum_{i=1}^{k} x_i \sim \text{Gamma}(k, \theta)$. This follows by induction.

Box 2.6: The chi-squared distribution

The target distribution is the chi-squared distribution (with $\nu \in \mathbb{N}$ degrees of freedom) which has PDF

$$P(x) = \begin{cases} \frac{x^{(\nu/2)-1} \exp(-x/2)}{2^{\nu/2}\Gamma(\nu/2)} & \text{if } x > 0 \\ 0 & \text{else} \end{cases}. \tag{i}$$

A random variable from this distribution is denoted $x \sim \chi^2_k$.

An equivalent definition involves multiple independent normal random variables; $x_1, x_2, \ldots, x_\nu \overset{\text{iid}}{\sim} \mathcal{N}(0, 1)$. The new random variable formed from the quadrature sum $z \equiv \sum_{i=1}^{k} x_i^2$ is, by definition a chi-distributed random variable; $z \sim \chi^2_k$.

We now show that these definitions are equivalent...

The random variables x_i define a point in a ν-dimensional space; the random variable z is the square of the radius vector to this point; $r = \sqrt{z} \Rightarrow \mathrm{d}r = \frac{\mathrm{d}z}{2\sqrt{z}}$.

The probability that z lies in the range $(z, z + \mathrm{d}z)$ can be expressed in two ways: in terms of z (left-hand side) and in terms of x_i (right-hand side);

$$P(z)\mathrm{d}z = \int_{\text{shell}} P(x_1)P(x_2)\ldots P(x_\nu)\mathrm{d}x_1\mathrm{d}x_1\ldots\mathrm{d}x_\nu\,, \tag{ii}$$

where the shell is the region between the two concentric spheres of radii z and $z + \mathrm{d}z$. Using the fact that each individual $x_i \sim \mathcal{N}(0, 1)$, we have

$$P(z)\mathrm{d}z = \int_{\text{shell}} \frac{\exp\left(-\frac{1}{2}\sum_{i=1}^{\nu} x_i^2\right)}{(2\pi)^{\nu/2}}\mathrm{d}x_1\mathrm{d}x_1\ldots\mathrm{d}x_\nu\,. \tag{iii}$$

The integrand depends only on r and is constant over the shell. Therefore, this integral is just the integrand times the area, A, of the shell times its thickness $\mathrm{d}r$;

$$P(z)\mathrm{d}z = \frac{\exp\left(-\frac{1}{2}z\right)}{(2\pi)^{\nu/2}}\int_{\text{shell}} \mathrm{d}x_1\mathrm{d}x_1\ldots\mathrm{d}x_\nu = \frac{\exp\left(-\frac{1}{2}z\right)}{(2\pi)^{\nu/2}}A\mathrm{d}r\,. \tag{iv}$$

The area of a $(\nu-1)$-sphere with radius r is $A = \frac{2r^{\nu-1}\pi^{\nu/2}}{\Gamma(\nu/2)}$. The result follows,

$$P(z)\mathrm{d}z = \frac{z^{(\nu/2)-1}\exp(-z/2)}{2^{\nu/2}\Gamma(\nu/2)}\mathrm{d}z\,. \tag{v}$$

2.2.5 Markov Chain Monte Carlo Methods

Markov Chain Monte Carlo (MCMC) methods are a broad class of stochastic sampling algorithms. They aim to construct a *Markov chain* that has the target distribution as its *stationary distribution*. If one can construct such a chain and evolve it for a suitably large number of iterations, then samples from P may be obtained by taking widely separated points from the chain. In practice, determining when the chain has reached equilibrium (after an initial transient phase usually called the "burnin") and then drawing independent samples from the subsequent correlated chain are important parts of an MCMC algorithm; these issues are discussed in Sec. 2.2.9. Several algorithms exist for constructing MCMC chains with the desired stationary distribution: these include Gibbs sampling (Sec. 2.2.6), Metropolis-Hasting sampling (Sec. 2.2.7), Hamiltonian sampling (Sec. 2.2.8), These meth-

ods are described here and are compared for a simple toy target distribution in Sec. 2.2.10.

As we will see, the defining property of a *Markov chain* is that it has no memory. At each stage the transition probability ρ that tells us how to move to the next point depends only on the current point; it does not depend on any of the previous history of the chain. If the chain satisfies some technical conditions described below, then after evolving for many iterations the distribution of points in the chain will approach a *stationary distribution* π. By carefully choosing ρ it is possible to make the stationary distribution equal any desired target distribution, $\pi = P$.

Definition. A *Markov chain* is a ordered sequence (or *chain*) of random points $x_0, x_1, x_2, \ldots$ in the sample space (i.e. $x_i \in \mathcal{X}$) that satisfies the *Markov property*,

$$P(x_{i+1}|x_0, x_1, \ldots, x_i) = P(x_{i+1}|x_i). \tag{2.13}$$

I.e. the distribution of each point x_i depends only on the previous point x_{i-1} in the chain.

A Markov chain is specified by its *transition probabilities*, $P(x_{i+1}|x_i)$ [4].

Definition. A Markov chain is said to be *time-homogeneous* if the transition probabilities do not depend on the position in the chain position; i.e. if $P(X_{k+1} = x|X_k = y)$ does not depend on k.

If the sample space $\mathcal{X}$ is finite, then the transition probabilities can be described by a matrix (see Box 2.1). The components of this matrix are given by

$$p_{ab} = P(X_{k+1} = b|X_k = a), \tag{2.14}$$

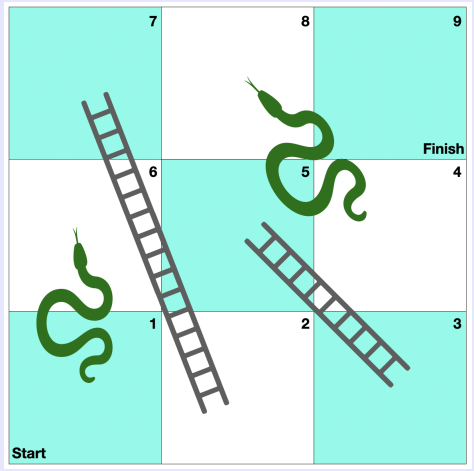
where $a, b \in \mathcal{X}$. If at the k^{th} iteration the chain is in state a, then at the $(k+1)^{\text{th}}$ iteration the probability distribution of the chain's location is given by the a^{th} row of this matrix. The transition matrix is a square matrix ($N \times N$, where $N = |\mathcal{X}|$) where the rows (but not necessarily the columns) represent probability distributions and therefore must sum to 1:

We will be primarily interested in the case where the sample space is continuous; e.g. $\mathcal{X}$ is the parameter space of some model in a Bayesian inference. In this case, the transition probabilities of a time-homogeneous Markov chain are described by a single time-independent transition probability PDF $P(x'|x)$. Given that I am in state x, this tells me the probability that I move to state x'. This transition probability will be denoted $\rho(x', x) \equiv P(x'|x)$.

[4] Strictly speaking, a full specification should also include the initial point x_0. Typically this is set by drawing randomly $x_0 \sim \alpha$ from some *initialisation distribution* α on $\mathcal{X}$. This initialisation distribution will be unimportant for our purposes.

Example 2.1: Snakes and Ladders as a Markov chain

The following game of Snakes and Ladders is an example of a Markov chain on a finite state space. (Snakes and Ladders really is a terrible game!)



This example is from J. R. Norris, "Markov Chains". The state space is the set of squares on the board; $\mathcal{X} = \{1, 2, \ldots, 9\}$. The player starts on the first square; the initialisation distribution is $\alpha_a = (1, 0, 0, 0, 0, 0, 0, 0, 0)$.

On each turn the player rolls a die and moves accordingly; this basic rule doesn't change throughout the game, meaning the Markov chain is time-homogeneous.

Fill in the missing parts of the transition probability matrix. (I'm playing with the rule that if you roll a number too big to move then you stay where you are.)

$$p_{ab} = \begin{pmatrix} 1/6 & 0 & 0 & 1/6 & 2/6 & 0 & 2/6 & 0 & 0 \\ 1/6 & 0 & 0 & 2/6 & 2/6 & 0 & 1/6 & 0 & 0 \\ 1/6 & 0 & 0 & 2/6 & 1/6 & 0 & 1/6 & 0 & 1/6 \\ 1/6 & 0 & 0 & 2/6 & 1/6 & 0 & 1/6 & 0 & 1/6 \\ \square & \square & \square & \square & \square & \square & \square & \square & \square \\ \square & \square & \square & \square & \square & \square & \square & \square & \square \\ \square & \square & \square & \square & \square & \square & \square & \square & \square \\ \square & \square & \square & \square & \square & \square & \square & \square & \square \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

We use the following notation for MCMC chains. Points in the chain are denoted $x_i \in \mathcal{X}$, with $i = 0, 1, 2, \ldots$. When $d = \dim(\mathcal{X}) > 1$, the components of points are denoted $x = (x^0, x^1, \ldots, x^{d-1})$. Subscripts label chain position and superscripts label vector components.

Ultimately, we will be interested in the long-term behaviour of the chain. As a first step, we need to understand the distribution of points in the chain after a certain fixed number of iterations.

Given a time-homogeneous Markov chain with initial position x_0 the distribution of the next point in the chain is, by definition, given by the transition probability; i.e. $P(x_1|x_0) = \rho(x_1, x_0)$. The distribution of the third point in the chain can be obtained as follows.

$$P(x_2|x_0) = \int \mathrm{d}x_1 \; P(x_2|x_1, x_0)P(x_1|x_0) \qquad (\textit{law of total probability}) \tag{2.15}$$

$$= \int \mathrm{d}x_1 \; P(x_2|x_1)P(x_1|x_0) \qquad (\textit{Markov property}) \tag{2.16}$$

$$= \int \mathrm{d}x_1 \; \rho(x_2, x_1)\rho(x_1, x_0) \qquad (\textit{def. of transition prob}) \tag{2.17}$$

It follows by induction that for a given starting position the distribution of the i^{th} point is

$$P(x_i|x_0) = \int \mathrm{d}x_{i-1} \int \mathrm{d}x_{i-2} \; \ldots \int \mathrm{d}x_1 \; \rho(x_i, x_{i-1})\rho(x_{i-1}, x_{i-2}) \ldots \rho(x_1, x_0). \tag{2.18}$$

Exercise 2.2: Finite State Space
Show that for a finite state space $\mathcal{X}$ the distribution of the i^{th} point in the

chain (the discrete analog of Eq. 2.18) is given by the matrix-vector product $P(X_i = a) = (\boldsymbol{\alpha}^{\mathrm{T}} \cdot \mathbf{p} \cdot \mathbf{p} \cdot \ldots \cdot \mathbf{p})_a = (\boldsymbol{\alpha}^{\mathrm{T}} \cdot \mathbf{p}^i)_a$, where the vector $\boldsymbol{\alpha}$ is the initialisation distribution and $\mathbf{p}$ is the transition probability matrix (Eq. 2.14).

Note, nearby points in the chain are *not* independent.

We want our Markov chains to be able to move all over the space $\mathcal{X}$.

Definition. A Markov chain is *irreducible* if for any starting point $x_0 \in \mathcal{X}$ for any (measurable) region $A \subset \mathcal{X}$ there exists an integer $n \geq 1$ such that $\int_A \mathrm{d}x\, P(x_n|x_0) > 0$.

Irreducibility will be an importance property for our intended applications. It ensures that (given enough iterations) our chains have a *go anywhere* tendency that means they can fully explore the sample space $\mathcal{X}$. This property is sometimes called *ergodicity*.

We consider only *irreducible* and *time-homogeneous* Markov chains here. Again, we are interested in the long-term behaviour of the chain. Together, the irreducibility and time-homogeneity ensures that the chain never stops moving, e.g. by converging to a point. However, the distribution of points in the chain might still converge to something.

Definition. A time-homogeneous Markov chain is said to approach a *limiting distribution* λ on the space $\mathcal{X}$ if

$$\lim_{n\to\infty} P(x_n|x_0) = \lambda(x_n). \tag{2.19}$$

A limiting distribution doesn't have to exist, but if it does exist then it will be unique.

If we start the chain in the limiting distribution then it will remain there. To see that this is the case, let $x_0 \sim \lambda$ and consider the distribution of the next point in the chain;

$$P(x_1) = \int \mathrm{d}x_0\, \rho(x_1, x_0)\lambda(x_0) = \lambda(x_1), \tag{2.20}$$

where the final equality follows from from Eq. 2.18 and the definition of λ.

The limiting distribution is an example of a *stationary distribution*.

Definition. Given a Markov chain with transition probabilities $\rho(x', x)$, a distribution π on $\mathcal{X}$ is said to be a *stationary distribution* of the Markov chain if

$$\pi(x') = \int \mathrm{d}x\ \pi(x)\rho(x', x). \tag{2.21}$$

Note that a limiting distribution is necessarily stationary, but a stationary distribution is not necessarily a limiting distribution.

In order to be useful for stochastic sampling, we now need a way not only of determining the stationary distribution, but also of ensuring that it matches out target distribution; i.e. $\pi = P$. The easiest way do this is to impose the stricter condition of *detailed balance*.

Definition. Consider a time-homogeneous Markov chain with transition probabilities $\rho(x', x)$. The chain is said to satisfy *detailed balance* with respect to π if

$$\pi(x)\rho(x', x) = \pi(x')\rho(x, x'). \tag{2.22}$$

In order to see the significance of the *detailed balance equations* in Eq. 2.22, consider two regions $A, B \subset \mathcal{X}$ (see Fig. 2.3). Suppose a chain at position $x \sim \pi$ moves at the next iteration to x'. Consider the probability that it moves from A to B,

$$\begin{aligned} \mathrm{Prob}(x \in A \textbf{ and } x' \in B) &= \mathrm{Prob}(x \in A)\mathrm{Prob}(x' \in B|x) && (2.23)\\ &= \int_{x\in A} \mathrm{d}x\ \pi(x) \int_{x'\in B} \mathrm{d}x'\ \rho(x', x). && (2.24) \end{aligned}$$

The integrand on the right-hand side of Eq. 2.24 is the left-hand side of Eq. 2.22. Similarly, integrating the right-hand side of the detailed balance equation gives the probability that a chain initially in B and modes to A. The detailed balance equation ensure these probabilities are equal for any regions A and B. Detailed balance ensures that there is no net flux of probability anywhere in the space, when the chain is in the stationary distribution.

Lemma 2.2.4. *If a Markov chain satisfies detailed balance with respect to π then π is a stationary distribution of the Markov chain.*

Proof. Integrate Eq. 2.22 w.r.t. x'. On LHS, use fact that π is normalised to get Eq. 2.21. □

We assume without proof if a time-homogeneous, irreducible Markov chain satisfies detailed balance then π is the *unique* stationary distribution and is also a limiting distribution.

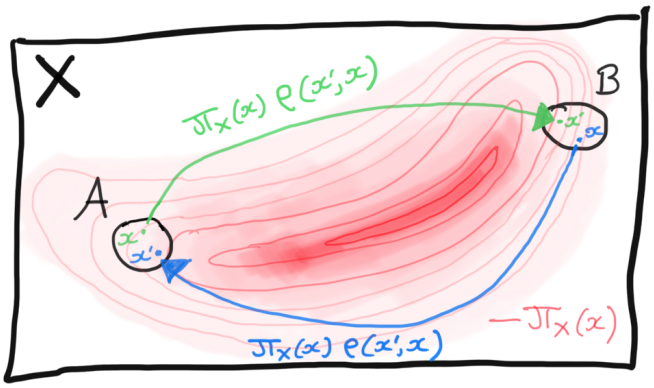


Figure 2.3: Illustration of detailed balance. A Markov chain moving in a 2D space $\mathcal{X}$ has a stationary distribution π (shown in red). If the chain satisfies detailed balance then there is no net flow of probability anywhere in $\mathcal{X}$. In particular, the flow from any region A into any region B (green arrow) is balanced by the flow from B into A (blue arrow).

Note that detailed balance is a sufficient, but not a necessary, condition for the existence of a stationary distribution, π. In other words, the detailed balance condition is a stricter than we really need. However, the detailed balance condition is convenient because it gives an easy way of finding the stationary distribution. The (local) detailed balance conditions in Eq. 2.22 are usually easier to check than the (global) stationarity conditions in Eq. 2.21.

For understanding the following sections, the most important conclusion to take away from this brief discussion of Markov chains is...

If we can design transition probabilities $\rho(x', x)$ for a time-homogeneous, irreducible Markov chain s.t. they satisfy detailed balance with $\pi = P$, then after evolving for enough iterations the distribution of the chain will approach P.

2.2.6 Gibbs Sampling

Gibbs sampling is named after J. W. Gibbs but was not developed by him[5].

The idea that motivates Gibbs sampling is that it is usually easier to sample from 1-

[5]Geman & Geman (1984) "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images", IEEE Transactions on Pattern Analysis and Machine Intelligence, **6** 6, 721–741 doi:10.1109/TPAMI.1984.4767596

dimensional distributions that from d-dimensional ones. And for some target distributions it is easier to sample from their 1-dimensional *conditional* distributions than from either the full distribution or its 1-dimensional *marginalised* distributions.

The PDFs of the conditional distributions can be found from PDF of the joint distribution. For the target distribution $P(x^0, x^1 \ldots, x^{d-1})$ the 1-dimensional conditional distribution $P(x^k|x^{-k})$ (where x^{-k} denotes all the components of the vector x except x^k; i.e. $x^{-k} = \{x^0, \ldots, x^{k-1}, x^{k+1}, \ldots, x^{d-1}\}$) has a PDF proportional to the PDF of the joint distribution;

$$P(x^k|x^{-k}) = \frac{P(x^0, x^1 \ldots, x^{d-1})}{\int \mathrm{d}x^k \; P(x^0, x^1 \ldots, x^{d-1})} \propto P(x^0, x^1 \ldots, x^{d-1}). \tag{2.25}$$

In this context, proportional means that the denominator is not a function of x^k.

A basic version of the Gibbs sampling algorithm proceeds as follows. An arbitrary starting value is chosen (e.g. by drawing x_0 from some arbitrary initialisation distribution α on $\mathcal{X}$) and at each iteration one parameter component is chosen to be updated from the corresponding conditional target distribution.

Algorithm 2.2 Gibbs

1: $x_0 \sim \alpha$ ▷ Initialise
2: $i \leftarrow 0$
3: **while** $i \geq 0$ **do** ▷ Iterate $i = 0, 1, 2, \ldots$
4: $\quad k \sim \mathrm{Cat}(w_0, w_1, \ldots, w_{d-1})$ ▷ Choose component
5: $\quad y \sim P(x^k|x_i^{-k})$ ▷ Sample 1D conditional dist
6: $\quad x_{i+1} \leftarrow (x_i^0, \ldots, x_i^{k-1}, y, x_i^{k+1}, \ldots, x_i^{d-1})$ ▷ Markov transition
7: $\quad i \leftarrow i + 1$
8: **end while**

The output of the Gibbs algorithm (Alg. 2.2) is the Markov chain $x_0, x_1, x_2, \ldots$

The weights $w_0, \ldots, w_{d-1}$ satisfy $\sum_{k=0}^{d-1} w_k = 1$. These are the parameters of a *categorical distribution* that controls how often each component gets updated. [6] The weights can be chosen arbitrary by the user, but are often set equally; i.e. $w_k = 1/d$, for all k.

The Markov chain is formally infinite; the Gibbs algorithm (Alg. 2.2) never terminates. Any practical implementation of Alg. 2.2 (or any of the other MCMC algorithms that will be described below) must periodically test to see if the finite Markov chain obtained

[6] The categorical distribution is a discrete distribution describing a random variable that can take on one of d possible values $\{0, 1, \ldots, d-1\}$, with the specified probabilities $\{w_0, w_1, \ldots, w_{d-1}\}$.

so far looks to have converged and terminate the algorithm. Some possible convergence diagnostics are discussed in Sec. 2.2.9.

The weights $w_0, \ldots, w_{d-1}$ satisfy $\sum_{k=0}^{d-1} w_k = 1$. These are the parameters of a *categorical distribution* that controls how often each component gets updated. The weights can be arbitrary, but are often set equally; i.e. $w_k = 1/d$, for all k.

If the weights set equally, a nice feature of the Gibbs algorithm is that it has no free parameters and requires no user inputs, such as a choice for a proposal distribution.

Box 2.7: Gibbs sampling demo

We will use the Gibbs method to sample the following 2D target distribution,

$$P(x, y) = y \exp(-[xy + y]) \quad \text{with } 0 < x, y < \infty. \tag{i}$$

The 1D distribution of x conditioned on a value of y (i.e. treating y as constant) is

$$P(x|y) \propto \exp(-yx). \tag{ii}$$

We recognise this as being the exponential distribution with rate parameter y; i.e. $x|y \sim \text{Exp}(y)$. Similarly, the 1D of y conditioned on a given value of x is

$$P(y|x) \propto y \exp(-[x + 1]y), \tag{iii}$$

which we recognise as the gamma distribution; i.e. $y|x \sim \text{Gamma}\left(2, \frac{1}{x+1}\right)$.

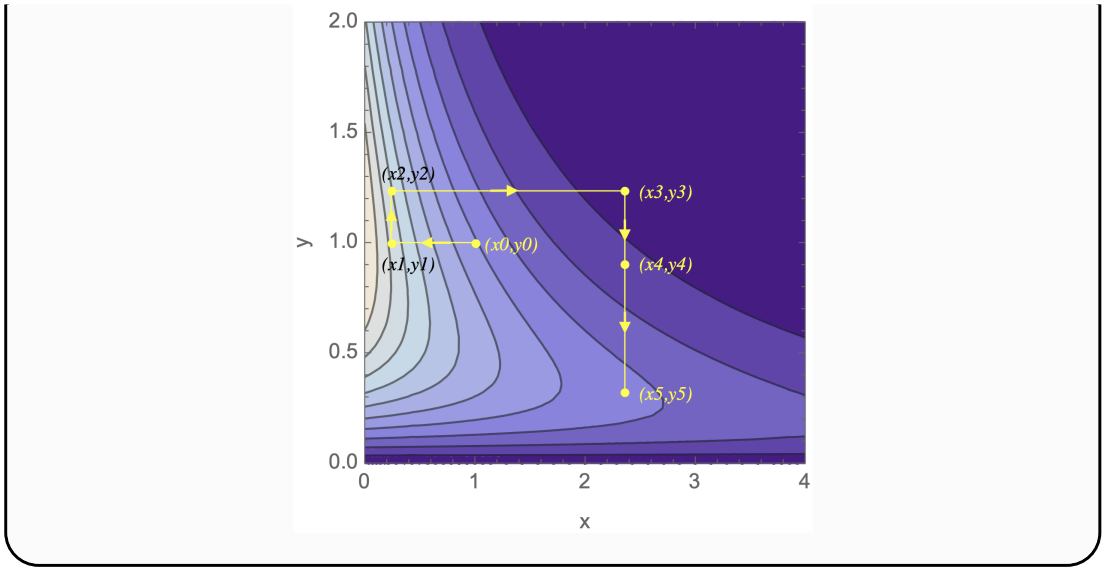
We know how to sample from both the exponential and gamma distributions (see Examples 2.2 and 2.5 respectively) so we can implement the Gibbs method.

We initialise the chain at position $(x_0, y_0) = (1, 1)$ (this is arbitrary).

First iteration. Randomly choose $k \in \{0, 1\}$. If we choose $k = 0$, we update the x-coordinate by drawing $x_1 \sim \text{Exp}(1)$ and move horizontally to $(x_1, y_1 = y_0)$.

Second iteration. If we choose $k = 1$, we update the y-coordinate by drawing $y_2 \sim \text{Gamma}(2, 1/(x_1 + 1))$; move horizontally to $(x_2 = x_1, y_2)$.

Five iterations are shown superposed on the target distribution below.



The Gibbs algorithm clearly defines a time-homogeneous Markov chain. This Markov chain will typically be irreducible (although, see counter example below). The algorithm is designed such that the target distribution is the stationary distribution of the chain.

Theorem 2.2.5. *The Gibbs algorithm (Alg. 2.2) produces a Markov chain $x_0, x_1, \ldots$ that satisfies the detailed balance condition with $\pi = P$.*

Proof. The Gibbs algorithm defines a Markov chain with transition probabilities

$$\rho(y, x) = \sum_k w_k \delta^{(d-1)}(y^{-k} - x^{-k}) P(y^k | x^{-k}), \tag{2.26}$$

Substituting this into the left-hand side of the detailed balance condition in Eq. 2.22 gives

$$\begin{aligned} \text{LHS} &= P(x)\rho(y, x) && (2.27) \\ &= \sum_k P(x^k | x^{-k}) P(x^{-k}) w_k \delta^{(d-1)}(y^{-k} - x^{-k}) P(y^k | x^{-k}), && (2.28) \end{aligned}$$

where on the second line we have used $P(x) = P(x^k | x^{-1}) P(x^{-k})$. Substituting for $\rho(x, y)$

into the right-hand side of the detailed balance condition gives

$$\begin{aligned} \text{RHS} &= P(y)\rho(x,y) && (2.29)\\ &= \sum_k P(y^k|y^{-k})P(y^{-k})w_k\delta^{(d-1)}(x^{-k}-y^{-k})P(x^k|y^{-k}) && (2.30)\\ &= \sum_k P(y^k|x^{-k})P(x^{-k})w_k\delta^{(d-1)}(x^{-k}-y^{-k})P(x^k|x^{-k}), && (2.31) \end{aligned}$$

where on the last line we have used the properties of the Dirac delta function. Comparing Eqs. 2.28 and 2.31 shows that LHS = RHS which proves the result. □

Gibbs sampling is useful when sampling from the full target distribution is difficult, but sampling from the 1-dimensional conditional distributions of each variable is easier.

Note, although the Gibbs algorithm generally produces an irreducible Markov chain this is not strictly guaranteed; see, for example, the counter example in Fig. 2.4.

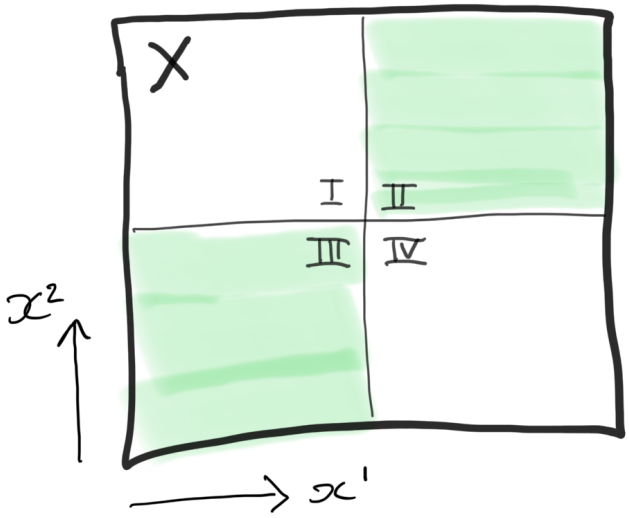


Figure 2.4: Consider a 2D target distribution with support only in the shaded regions II and III; i.e. $\int_I \mathrm{d}x\, P(x) = \int_{IV} \mathrm{d}x\, P(x) = 0$. A chain initialised in II will *never* move to III (or vice versa) under Gibbs evolution because the two steps required to do so (one horizontal and one vertical) necessarily enters one of the forbidden regions I or IV.

Gibbs sampling is closely related to Metropolis–Hastings sampling (see Sec. 2.2.7 and Box 2.9).

Although easy to implement, Gibbs sampling produces a Markov chain that can exhibit undesirable *random walk* behaviour and take a large number of iterations to *diffuse* across the target space. This is true in particular for highly correlated target distributions (see the example in Sec. 2.2.10).

There are many variants of the of Gibbs sampling algorithm, some are described in the following (unnumbered) subsections. Many of these variants are designed to try and reduce or eliminate this random walk behaviour.

2.2.6.1 Gibbs "Sweep" Sampling

Instead of choosing just one component to update at each iteration, in this common variant of the Gibbs algorithm every component gets updated (in a fixed order) at every iteration in a process called a *sweep*.

This "sweep" version of the Gibbs sampling algorithm proceeds as follows. At each iteration every component is updated by drawing from the 1-dimensional conditional distribution conditioned on the values of all of the parameters updated so far.

Algorithm 2.3 Gibbs Sweep

$x_0 \sim \alpha$ ▷ Initialise
$i \leftarrow 0$
while $i \geq 0$ **do** ▷ Iterate $i = 0, 1, 2, \ldots$
  for $k = 0$ **to** $d - 1$ **do** ▷ The "sweep"
    $y^k \sim P(x^k | y^0, \ldots, y^{k-1}, x_i^{k+1}, \ldots, x_i^{d-1})$ ▷ Sample 1D conditional dist
  end for
  $x_{i+1} \leftarrow (y^0, \ldots, y^{d-1})$ ▷ Markov transition
  $i \leftarrow i + 1$
end while

This version of the Gibbs sampling algorithm does not satisfy the detailed balance condition, but does still converge to the target distribution.

Lemma 2.2.6. *The Gibbs sweep algorithm (Alg. 2.3) produces a Markov chain $x_0, x_1, \ldots$ that has P as its unique stationary distribution.*

Proof. We show this for the $(d = 2)$-dimensional case. Let $P(x) \equiv P(x^0, x^1)$ be the target

distribution. Consider a move from (x^0, x^1) to (y^0, y^1); in the Gibbs sweep this proceeds in stages: $(x^0, x^1) \to (y^0, x^1) \to (y^0, y^1)$. The Markov transition probability is

$$\rho\big([y^0, y^1], [x^0, x^1]\big) = P(y^0|x^1)P(y^1|y^0), \tag{2.32}$$

$$= \frac{P(y^0, x^1)}{\int \mathrm{d}a\ P(a, x^1)} \frac{P(y^0, y^1)}{\int \mathrm{d}b\ P(y^0, b)}, \tag{2.33}$$

where the definition of the 1-dimensional conditional distributions in Eq. 2.25 has been used. Substitute this into the following integral for the *stationarity condition* in Eq. 2.21,

$$\int \mathrm{d}x\ P(x)\rho(y, x) = \int \mathrm{d}x^0 \int \mathrm{d}x^1\ P(x^0, x^1)\rho\big([y^0, y^1], [x^0, x^1]\big) \tag{2.34}$$

$$= \frac{P(y^0, y^1)}{\int \mathrm{d}b\ P(y^0, b)} \int \mathrm{d}x^1\ \frac{P(y^0, x^1)}{\int \mathrm{d}a\ P(a, x^1)} \int \mathrm{d}x^0\ P(x^0, x^1) \tag{2.35}$$

$$= P(y_0, y_1) \tag{2.36}$$

$$= P(y). \tag{2.37}$$

This shows the *stationarity condition* is satisfied. This argument can be extended (messily!) to the case of a general number d dimensions. □

There are many possible variants of the Gibbs algorithm, but you have to be careful; it is easy to make a small, seemingly insignificant change to the algorithm that breaks everything. The following *not-Gibbs algorithm* (Alg. 2.4) looks very similar to the Gibbs sweep algorithm (Alg. 2.3); however, this variation this doesn't work. The not-Gibbs algorithm produces a Markov that doesn't converge to the target distribution.

Algorithm 2.4 Not Gibbs (WARNING: this doesn't work!)

$x_0 \sim \alpha$ ▷ Initialise
$i \leftarrow 0$
while $i \geq 0$ **do** ▷ Iterate $i = 0, 1, 2, \ldots$
  for $k = 0$ **to** $d - 1$ **do** ▷ The "sweep"
    $y^k \sim P(x^k | x_i^0, \ldots, x_i^{k-1}, x_i^{k+1}, \ldots, x_i^{d-1})$ ▷ Sample 1D conditional dist
  end for
  $x_{i+1} \leftarrow (y^0, \ldots, y^{d-1})$ ▷ Markov transition
  $i \leftarrow i + 1$
end while

Comparing Algs. 2.3 and 2.4, it can be seen that the parameter updates in the Gibbs sweep have to be performed in serial, not in parallel.

2.2.6.2 Blocked Gibbs Sampling

The basic Gibbs algorithm only updates one parameter component at each iteration; it does this by sampling from the 1-dimensional conditional distribution for that parameter. However, it is not necessary to sample only from 1-dimensional conditional distributions. Sometimes, it may be possible to sample from higher-dimensional conditional distributions thereby updating several parameters at each iteration.

The *blocked Gibbs* sampling algorithm achieves this by grouping parameters together into a number $g < d$ of groups, or *blocks*, of one or more components; e.g.

$$(x^0, x^1, \ldots, x^{d-1}) = (\theta^0, \theta^1, \ldots, \theta^{g-1}), \tag{2.38}$$

where each block $\theta^\mu \in \mathbb{R}^{d_\mu}$ consists of the parameters $\theta^\mu = (x^{\sum_{\mu'<\mu} d_{\mu'}}, \ldots, x^{\sum_{\mu'\leq\mu} d_{\mu'}})$, the index $\mu = 0, 1, \ldots, g-1$, and $\sum_{\mu=0}^{g-1} d_\mu = d$.

The blocked Gibbs algorithm requires the blocks θ^μ to be chosen such that it is possible to sample from the distributions conditioned on the parameters in the other blocks; $P(\theta^\mu|\theta^{-\mu})$.

A basic blocked Gibbs sampling algorithm proceeds as follows.

Algorithm 2.5 Blocked Gibbs

1: $x_0 \sim \alpha$ ▷ Initialise
2: $i \leftarrow 0$
3: **while** $i \geq 0$ **do** ▷ Iterate $i = 0, 1, 2, \ldots$
4: $\quad \mu \sim \mathrm{Cat}(w_0, w_1, \ldots, w_{d-1})$ ▷ Choose block
5: $\quad \Theta^\mu \sim P(\theta^\mu|\theta^{-\mu})$
6: $\quad x_{i+1} \leftarrow (\theta^0, \ldots, \theta^{\mu-1}, \Theta^\mu, \theta^{\mu+1}, \ldots, \theta^{g-1})$ ▷ Markov transition
7: $\quad i \leftarrow i + 1$
8: **end while**

Lemma 2.2.7. *The blocked Gibbs algorithm produces a Markov chain $x_0, x_1, \ldots$ that satisfies the detailed balance condition with $\pi = P$.*

Proof. The proof is essentially identical to that of theorem 2.2.5. □

The version of the blocked Gibbs algorithm presented in Alg. 2.5 chooses which parameter block to update randomly at each iteration. It is of course also possible to perform a *blocked Gibbs sweep* algorithm (a combination of Algs. 2.3 and 2.5) where the parameter blocks are all updated sequentially at each iteration.

2.2.6.3 Gibbs Sampling With Ordered Overrelaxation

A common problem with the basic Gibbs algorithm is that the chain can exhibit undesirable *random walk* behaviour, taking many iterations to *diffuse* across parameter space. (See, for example, the case study in Sec. 2.2.10.) *Overrelaxation* methods seeks to address this by proposing transitions to new positions that are negatively correlated with the current value, while maintaining detailed balance. This can encourage the chain to take big steps to the "other side" of the target distribution, reducing the time needed to explore the space. The effectiveness (or otherwise) of overrelaxation is very problem dependent.

There isn't a unique way to propose transitions that are anticorrelated with the current position; consequently, there isn't a unique overrelaxation algorithm. One example is *Gibbs ordered overrelaxation* (ORR)[7] which uses order statistics to propose the transitions

The Gibbs OOR algorithm proceeds as follows.

Algorithm 2.6 Gibbs Ordered Overrelaxation

$x_0 \sim \alpha$ ▷ Initialise
$i \leftarrow 0$
while $i \geq 0$ **do** ▷ Iterate $i = 0, 1, 2, \ldots$
  $k \sim \text{Cat}(w_0, w_1, \ldots, w_{d-1})$ ▷ Choose component
  $y_0, y_1, \ldots, y_{s-1} \overset{\text{iid}}{\sim} P(x^k | x^{-k})$ ▷ Draw odd number s of possible points
  $\text{list} \leftarrow \textsc{Sort}(\{x_i^k, y_0, y_1, \ldots, y_{s-1}\})$
  $S \leftarrow \textsc{Index}(x_i^k, \text{list})$ ▷ Find index of x_i^k in sorted list
  $Y \leftarrow \text{list}[[s + 1 - S]]$ ▷ If x_i^k in 2^{nd} position, take 2^{nd} last item
  $x_{i+1} = (x^0, \ldots, x^{k-1}, Y, x^{k+1}, \ldots, y^{d-1})$ ▷ Markov transition
  $i \leftarrow i + 1$
end while

The Gibbs OOR method introduces just one tunable parameter, the odd number s. A naive implementation Alg. 2.6 will have a computational cost per iteration that scales linearly S. (However, it is possible to implement Gibbs OOR in a way that does not require drawing $y_0, y_1, \ldots, y_{s-1} \overset{\text{iid}}{\sim} P(x^k | x^{-y})$ explicitly; see Neal [1995].)

[7] Neal (1995) "Suppressing Random Walks in Markov Chain Monte Carlo Using Ordered Overrelaxation" arXiv:bayes-an/9506004.

2.2.7 Metropolis-Hasting Sampling

The Metropolis-Hasting (MH) algorithm is historically probably the most important and most well-studied sampling algorithm.

There is some controversy regarding who should be credited for the development of the algorithm, but it is named after Metropolis *et al.* (1953) and Hastings (1970) [8] [9].

The most important ingredient in the MH algorithm is the *proposal distribution* $Q(y|x)$ on $\mathcal{X}$. The proposal distribution must be supplied by the user; it is an input to the algorithm. If the chain is currently at position x_i, the proposal is used to generate candidates for the next chain position, $y \sim Q(y|x_i)$, which can either be accepted (i.e. $x_{i+1} = y$) or rejected (i.e. $x_{i+1} = x_i$).

The most basic version of the MH algorithm proceeds as follows.

Algorithm 2.7 Metropolis Hastings

$x_0 \sim \alpha$ ▷ Initialise
$i \leftarrow 0$
while $i \geq 0$ **do** ▷ Iterate $i = 0, 1, 2, \ldots$
  $y \sim Q(y|x_i)$ ▷ Proposal
  $a \leftarrow (P(y)Q(x_i|y)) \,/\, (P(x_i)Q(y|x_i))$ ▷ MH acceptance probability
  $u \sim \mathcal{U}$
  if $u < a$ **then**
    $x_{i+1} \leftarrow y$ ▷ Accept
  else
    $x_{i+1} \leftarrow x_i$ ▷ Reject
  end if
  $i \leftarrow i + 1$
end while

The output of the MH algorithm (Alg. 2.7) is the Markov chain $x_0, x_1, x_2, \ldots$

The MH algorithm clearly defines a time-homogeneous Markov chain. For any sensible choice of proposal distribution Q this Markov chain will also generally be irreducible.

[8] Metropolis *et al.* (1953) "Equation of State Calculations by Fast Computing Machines", Journal of Chemical Physics, **21** 6 1087–1092, doi:10.1063/1.1699114

[9] Hastings (1970) "Monte Carlo Sampling Methods Using Markov Chains and Their Applications", Biometrika, **57** 1, 97–109, doi:10.1093/biomet/57.1.97

The MH algorithm effectively defines the transition probabilities for the Markov chain as a kind of mixture of the proposal (for accepted transitions) and delta-function distributions (for rejected transitions). See Box. 2.8. The transition probability is

$$\rho(y, x) = a(y, x)Q(y|x) + \delta^{(d)}(y - x) \int \mathrm{d}y' \left[1 - a(y', x)\right] Q(y'|x), \tag{2.39}$$

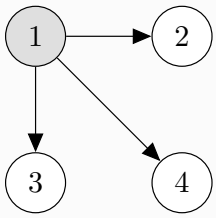
where the *acceptance function* is given by

$$a(y, x) = \min\left(1, \frac{P(y)Q(x|y)}{P(x)Q(y|x)}\right). \tag{2.40}$$

Box 2.8: Understanding the MH transition probability

In order to help understand the expression in Eq. 2.39 for the MH transition probabilities it can help to consider the case of a discrete sample space.

Suppose our sample space consists of just 4 states $\mathcal{X} = \{1, 2, 3, 4\}$. Suppose we are currently in the first state, $x_i = 1$. Consider the Markov transition to the next state x_{i+1}. The value x_{i+1} is a random variable with a probability distribution over the four states given by the transition probability, $\rho(x_{i+1}, 1) = P(x_{i+1}|x_i = 1)$.



The MH algorithm (Alg. 2.7) creates the transition in two stages: first, propose a point $y|x_i \sim Q(y|x_i)$; second, accept or reject the transition with probabilities $a(y, x)$ and $1 - a(y, x)$ respectively.

There is only one way to get to state $x_{i+1} = 2$; we must propose $y = 2$ *and* accept the proposal. The probability that our transition takes us to state 2 is

$$P(x_{i+1} = 2|x_i = 1) = Q(2|1)a(2, 1). \tag{i}$$

However, there are many ways of getting to state $x_{i+1} = 1$; we can propose $y = 1$ *and* accept the proposal, or we can propose any point $y = j$ *and* reject the proposal.

The probability that our transition takes us to state 1 is

$$P(x_{i+1}=1|x_i=1)=Q(1|1)a(1,1)+\sum_{j=1}^{4}Q(j|1)\big(1-a(j,1)\big). \tag{ii}$$

The key step in the algorithm, namely the probabilistic rule for accepting/rejecting proposed points, is designed such that the resulting Markov chain satisfies detailed balance with $\pi = P$. Therefore, from the results in the previous section, we know that chain will have the target distribution P as its unique stationary, limiting distribution.

Theorem 2.2.8. *The MH algorithm (Alg. 2.7) produces a Markov chain $x_0, x_1, \ldots$ that satisfies the detailed balance condition with $\pi = P$.*

Proof. Substituting the expression for the $\rho(y,x)$ in Eq. 2.39 into the detailed balance condition Eq. 2.22 with $\pi = P$ gives

$$P(x)a(y,x)Q(y|x)+\delta^{(d)}(y-x)P(x)\int \mathrm{d}y'\ \big[1-a(y',x)\big]\,Q(y'|x)\stackrel{?}{=} \tag{2.41}$$

$$P(y)a(x,y)Q(x|y)+\delta^{(d)}(x-y)P(y)\int \mathrm{d}y'\ \big[1-a(y',y)\big]\,Q(y'|y). \tag{2.42}$$

The terms involving the delta functions on both sides are clearly equal. Therefore, checking the detailed balance condition reduces to checking

$$P(x)a(y,x)Q(y|x)\stackrel{?}{=}P(y)a(x,y)Q(x|y). \tag{2.43}$$

In the expressions for $a(y,x)$ and $a(x,y)$, exactly one min condition evaluates to 1. Consider each case separately. If $a(y,x)=1$ and $a(x,y)=\frac{P(x)Q(y|x)}{P(y)Q(x|y)}$, then Eq. 2.43 becomes

$$P(x)Q(y|x)\stackrel{?}{=}P(y)\frac{P(x)Q(y|x)}{P(y)Q(x|y)}Q(x|y), \tag{2.44}$$

which is true. The case where $a(x,y)=1$ and $a(y,x)=\frac{P(y)Q(x|y)}{P(x)Q(y|x)}$ proceeds similarly. □

Note that the key step in the MH algorithm (line 5 in Alg. 2.7) depends only on the ratio of two evaluations of the target PDF, $P(y)/P(x_i)$. This means that the algorithm doesn't require the normalised PDF, all that is needed is a function $\hat{P}(x) \propto P(x)$. This is very useful for Bayesian inference where we generally do not know the evidence.

Note that a MCMC chain produced by the MH algorithm can have repeated entries; i.e. we might have $x_i = x_{i+1}$. This occurs when the algorithm rejects a proposed point, setting $x_{i+1} = x_i$. This can lead to the chain getting temporarily stuck in one place. This contrasts with the Gibbs algorithm (Sec. 2.2.6) which moves to a new point at every iteration. A useful diagnostic is the *acceptance fraction* $0 < g < 1$, defined as the number of accepted proposals divided by the total number of iterations.

Implementing the MH algorithm requires the user to make a choice for the proposal distribution, Q. Essentially any choice will give a Markov chain that eventually converges to P eventually, but the rate of convergence can differ wildly. A badly chosen proposal can lead to a chain that takes a long time to explore $\mathcal{X}$. The MH algorithm generally works best if the proposal closely matches the target distribution. For best performance, we can tune the proposal distribution so that $g \sim 30\%$. (But be careful; changing the proposal mid evolution breaks the time-homogeneity of the Markov chain!) As we will see in Sec. 2.2.10, the choice of proposal strongly influences the efficiency of the MH algorithm.

The MH sampling algorithm is closely related to Gibbs sampling; see Box 2.9.

Box 2.9: Relating the Gibbs and MH algorithms

Here we show that the basic Gibbs algorithm (Alg. 2.2) is a special case of the MH algorithm with a particular choice of proposal.

To make the connection, consider lines 4 and 5 in the basic Gibbs algorithm (Alg. 2.2) as defining a MH proposal distribution, Q. The PDF of this proposal is

$$Q(y|x) = \sum_k w_k P(y^k|x^{-k}) \delta^{(d-1)}(x^{-k} - y^{-k}). \tag{i}$$

With this proposal, the MH acceptance function (Eq. 2.40) becomes

$$a(y,x) = \min\left(1, \frac{P(y)Q(x|y)}{P(x)Q(y|x)}\right) \tag{ii}$$

$$= \min\left(1, \frac{\sum_k w_k \frac{P(x^k|y^{-k})\delta^{(d-1)}(y^{-k}-x^{-k})}{P(x)}}{\sum_{k'} w_{k'} \frac{P(y^{k'}|x^{-k'})\delta^{(d-1)}(x^{-k'}-y^{-k'})}{P(y)}}\right) \tag{iii}$$

Replacing $P(x) = P(x^k, x^{-k}) = P(x^k|x^{-k})P(x^{-k})$, and similarly for $P(y)$, and

using the properties of the delta function, this simplifies to

$$a(y,x) = \min\left(1, \frac{\sum_k w_k \frac{\delta^{(d-1)}(y^{-k} - x^{-k})}{P(x^{-k})}}{\sum_{k'} w_{k'} \frac{\delta^{(d-1)}(x^{-k'} - y^{-k'})}{P(y^{-k'})}}\right) = 1. \quad \text{(iv)}$$

With this proposal distribution we accept every proposed point.

From the point of view of someone familiar with MH, Gibbs can be thought of as a special case where one uses a specific proposal built from the the 1D conditionals of the target that always gives an acceptance probability of unity.

From the point of view of someone familiar with Gibbs, MH can be thought of as a generalisation that allows us great deal of extra flexibility. Instead of always sampling from the conditionals, we can now use any distribution we like as a proposal provided that we include an accept/reject step in the algorithm.

As with Gibbs, many variations and extensions of the MH algorithm have been proposed. One such variant, *Hamiltonian Monte Carlo*, is discussed in the next section (Sec. 2.2.8).

2.2.7.1 Metropolis sampling

The *Metropolis algorithm* is a special case of the Metropolis-Hastings algorithm where the proposal is symmetric, $Q(y|x) = Q(x|y)$; i.e. the probability of proposing a move to y from x is the same as proposing x from y. In this case the acceptance function simplifies to

$$a(y,x) = \min\left(1, \frac{P(y)}{P(x)}\right). \quad (2.45)$$

2.2.8 Hamiltonian Monte Carlo

As we will see in the case study in Sec. 2.2.10, a major problem with Gibbs and MH sampling can be that the resulting Markov chains exhibit undesirable *random walk* behaviour and take many iterations to *diffuse* slowly around the sample space. *Hamiltonian Monte Carlo* (HMC) (sometimes *hybrid Monte Carlo*) is a Metropolis-like method that uses gradient information about the target distribution and the equations of Hamiltonian dynamics

to efficiently propose transitions across large distances in the sample space [10] [11] [12].

HMC is a Metropolis-like method in that it requires only a function proportional to the target PDF, not the normalised distribution. Unlike previous algorithms we have studied, HMC will also required that we are able to evaluate the gradient of this function with respect to the parameters. It is convenient to work with the logarithm of the target PDF,

$$\log P(x) = -E(x) + \text{const}, \tag{2.46}$$

where the quantity $E(x)$ is called the *potential* and a particle experiences a *force* equal to $-\nabla E(x)$, where ∇ denotes the d-dimensional gradient vector.

Points $x \in \mathcal{X}$ in the sample space are called *positions*. HMC augments these with new *momenta* variables $p \in T_x\mathcal{X}$ in the tangent space of the same dimension as $\mathcal{X}$. HMC also introduces a distribution Q on the tangent space $T_x\mathcal{X}$. This distribution can be chosen freely and should be easy to sample from; a common choice is the multivariate Gaussian

$$\log Q(p) = -K(p) + \text{const}, \quad \text{where} \quad K(p) = \frac{1}{2} p^{\mathrm{T}} M^{-1} p. \tag{2.47}$$

Here $K(p)$ is called the *kinetic energy* and M is a symmetric, positive definite *mass matrix*.

Position and momentum pairs (x, p) exist in a space (called the *phase space*) of twice the dimensionality of the original $\mathcal{X}$ (technically, the tangent bundle). HMC defines a new distribution R (called the *canonical distribution*) on the phase space defined as having log-PDF equal to the negative of the *Hamiltonian*,

$$\log R(x, p) = -\mathcal{H}(x, p) + \text{const}, \quad \text{where} \quad \mathcal{H}(x, p) = E(x) + K(p). \tag{2.48}$$

The counter-intuitive starting point of HMC is therefore to double dimensionality of the distribution we are working with. This might seem silly but go with it for the moment. Instead of trying to sample $x \sim P$ directly in d dimensions, we will instead try to sample $(x, p) \sim R$ in $2d$ dimensions. We will then simply throw away the momentum p variable leaving $x \sim P$, as desired. This works because we have defined the distribution R such that the target distribution is recovered when we marginalise over p;

$$P(x) = \int \mathrm{d}p \; R(x, p). \tag{2.49}$$

[10] Duane, Kennedy, Pendleton & Roweth (1987) "Hybrid Monte Carlo", Physics Letters B, 195 (2) 216–222 doi:10.1016/0370-2693(87)91197-X.

[11] Neal (1993) "Probabilistic inference using Markov chain Monte Carlo methods", Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, link.

[12] Neal (2011) "Handbook of Markov Chain Monte Carlo", chapter 5 "MCMC Using Hamiltonian Dynamics", CRC Press, link.

This is only useful if we can find an efficient way of sampling from R. To do this, HMC introduces a *fictitious time* parameter, t. The position $x(t)$ and momentum $p(t)$ are promoted to functions of t and evolved according to the equations of *Hamiltonian dynamics*:

$$\frac{\mathrm{d}x^k}{\mathrm{d}t} = \frac{\partial \mathcal{H}}{\partial p^k} \,, \tag{2.50}$$

$$\frac{\mathrm{d}p^k}{\mathrm{d}t} = -\frac{\partial \mathcal{H}}{\partial x^k} \,. \tag{2.51}$$

For any duration s, these equations define a map from states at time t to states at time $t+s$; i.e. $T_s : (x(t), p(t)) \to (x(t+s), p(t+s))$. The Hamiltonian $\mathcal{H}$, and hence the map T_s, does not depend explicitly on t.

Note that the continuous parameter t introduced as part of the Hamiltonian dynamics has nothing to do with the discrete index i which labels points in the Markov chain.

Note that the quantities $\mathcal{H}$, E, and t are called the "Hamiltonian", "potential" and "time" to emphasise an analogy with the equations of classical mechanics. However, this analogy is purely formal; in this formulation, all these quantities are dimensionless.

Hamiltonian dynamics has three well-known properties relevant for HMC. (Proofs can be found in any advanced classical mechanics textbook.)

Lemma 2.2.9. *Hamiltonian dynamics is reversible, meaning T_s has an inverse, T_{-s}.*

Lemma 2.2.10. *Hamiltonian dynamics is volume preserving. Consider all points (x, p) in some region A of phase space (with volume V_A). If B is the image of A under T_s (with volume V_B) then $V_A = V_B$. This result is known as Liouville's theorem.*

Lemma 2.2.11. *Hamiltonian dynamics conserves the Hamiltonian, meaning $\mathrm{d}\mathcal{H}/\mathrm{d}t = 0$. (This is true if the Hamiltonian $\mathcal{H}$ doesn't depend explicitly on t, as is the case here.)*

It is not generally possible to solve Hamilton's equations exactly. Instead, it is necessary to integrate Eqs. 2.50 numerically using small time steps Δt. This is usually done using the *leapfrog method* (Alg. 2.8), where a time step from $(x(t), p(t))$ to $(x(t+\Delta t), p(t+\Delta t))$ is done in three stages: a half step in momentum to find $p_i(t+\frac{1}{2}\Delta t)$, a full step in position to find $x_i(t+\Delta t)$, and finally another half step in position to find $p_i(t+\Delta t)$. For a particular choice of Δt, applying the leapfrog method (Alg. 2.8) repeatedly produces an *approximate Hamiltonian dynamics*.

Algorithm 2.8 Leapfrog step

1: **procedure** LEAPFROG(x, p, Δt, M)
2: $p \leftarrow p - \frac{1}{2}\Delta t \, \nabla|_x E(x)$ ▷ half step for momentum
3: $x \leftarrow x + \Delta t \, M^{-1} \cdot p$ ▷ full step for position
4: $p \leftarrow p - \frac{1}{2}\Delta t \, \nabla|_x E(x)$ ▷ half step for momentum
5: **return** x, p
6: **end procedure**

The key reason for choosing the leapfrog integrator is because it is *sympletic*, meaning that it is exactly *time reversible* (lemma 2.2.12) and *volume preserving* (lemma 2.2.13). This is not true of other numerical integrators, such as the Euler or Runge-Kutta integrators, which only have these properties approximately in the limit of small Δt.

Lemma 2.2.12. *The approximate leapfrog Hamiltonian dynamics is exactly reversible.*

Proof. Consider the state $(x(t + \Delta t), p(t + \Delta t))$. Reversing the sign of p gives $(x(t + \Delta t), -p(t+\Delta t))$. It is easy to check that applying the leapfrog to this state gives $(x(t), -p(t))$. Reversing the sign of p again gives $(x(t), p(t))$, as required. □

Lemma 2.2.13. *The approximate leapfrog Hamiltonian dynamics is exactly volume preserving.*

Proof. This follows from the fact that all three steps of the leapfrog are "shear transformations" in which one set of variables (either x or p) change by an amount that depends only on the other. Shear transformations have unit Jacobian. E.g. consider the middle step of the leapfrog (line 3); this defines a transformation of the form

$$\begin{pmatrix} x \\ p \end{pmatrix} \to \begin{pmatrix} x' \\ p' \end{pmatrix} = \begin{pmatrix} x + \Delta t f(p) \\ p \end{pmatrix}, \quad \text{with Jacobian } J = \begin{vmatrix} \frac{\partial x'}{\partial x} & \frac{\partial x'}{\partial p} \\ \frac{\partial p'}{\partial x} & \frac{\partial p'}{\partial p} \end{vmatrix} = \begin{vmatrix} 1 & \Delta t \frac{\partial f}{\partial p} \\ 0 & 1 \end{vmatrix} = 1. \quad (2.52)$$

The other steps of the leapfrog are of similar form and all have unit determinant. The Jacobian of the composition of transformations is the product of the individual Jacobians. It follows that a full leapfrog step has unit determinant and is volume preserving. □

The approximate leapfrog Hamiltonian dynamics does not perfectly conserve the Hamiltonian due to discretisation errors in the numerical integration. However, for small Δt the Hamiltonian is approximately conserved; it is easy to show that

$$\mathcal{H}\big(x(t + \Delta t), p(t + \Delta t)\big) = \mathcal{H}\big(x(t), p(t)\big) + \mathcal{O}\big(\Delta t^2\big). \quad (2.53)$$

With all this setup in place, we are now in a position to state the HMC algorithm.

Algorithm 2.9 HMC

$x_0 \sim \alpha$ ▷ Initialise
$i \leftarrow 0$
while $i \geq 0$ **do** ▷ Iterate $i = 0, 1, 2, \ldots$
  $p \sim Q$ ▷ Draw random momentum
  $x \leftarrow x_i$ ▷ Integrate from current chain position
  $H_{\text{initial}} \leftarrow \mathcal{H}(x, p)$ ▷ Initial Energy
  for $\ell = 1$ **to** L **do**
    $x, p \leftarrow \text{LEAPFROG}(x, p, \Delta t, M)$ ▷ Integrate
  end for
  $H_{\text{final}} \leftarrow \mathcal{H}(x, p)$ ▷ Final Energy
  $a \leftarrow \exp\left(H_{\text{initial}} - H_{\text{final}}\right)$ ▷ MH acceptance probability
  $u \sim \mathcal{U}$
  if $u < a$ **then**
    $x_{i+1} \leftarrow x$ ▷ Markov transition (accept)
  else
    $x_{i+1} \leftarrow x_i$ ▷ Markov transition (reject)
  end if
  $i \leftarrow i + 1$
end while

The output of the HMC algorithm (Alg. 2.9) is the Markov chain $x_0, x_1, x_2 \ldots$

Box 2.10: Motivating the HMC algorithm

HMC can be viewed as a combination of the blocked Gibbs and MH algorithm, with a clever choice for the proposal distribution.

In addition to the target distribution $P(x)$, we introduce another d-dimensional distribution $Q(p)$. For simplicity, $Q(p)$ is chosen to be a multivariate Gaussian.

We also introduce the augmented distribution $R(x, p) = P(x)Q(p)$ with dimension $2d$. Note, that R is separable, so P and Q are not only the marginal distributions of R (i.e. $P(x) = \int \mathrm{d}p\, R(x, p)$ and $Q(x) = \int \mathrm{d}x\, R(x, p)$) they are also its conditional distributions (from Eq. 2.25, $P(x) = R(x|p)$ and $Q(p) = R(p|x)$).

The idea is to sample from the augmented distribution, $(x_i, p_i) \overset{\text{iid}}{\sim} R$, and simply discard the momentum variables leaving samples $x_i \overset{\text{iid}}{\sim} P$, as required.

Sampling from R is done with a blocked Gibbs algorithm where the position and momentum variables are the two parameter *blocks*. We iterate according to

$$p_{i+1} \sim R(p|x_i) \equiv Q(p), \tag{i}$$
$$x_{i+1} \sim R(x|p_{i+1}) \equiv P(x). \tag{ii}$$

The momentum update i is easy because Q was chosen to be easy to sample. The position update ii is done via a MH step with a clever proposal. The proposal is made using the *leapfrog* to numerically integrate *Hamilton's equations*, for L steps starting from the point (x_i, p_{i+1}). The leapfrog integration is *symmetric* and *volume preserving* which means is defines a symmetric Metropolis proposal. We accept the proposed point with probability $\min(1, a)$, where $\log a$ is the difference between the Hamiltonian at the start and end of the numerical integration.

The reason this proposal is clever is that if the numerical integration is accurate then Hamiltonian is approximately conserved and the acceptance probability is high, $a \approx 1$ And it isn't a disaster if the numerical integration is sometimes a bit inaccurate because even if $a < 1$ this is still a valid MH step.

If L is sufficiently large and Δt sufficiently small, then this algorithm can take large steps in the parameter space with very high acceptance probabilities.

The HMC algorithm clearly defines a time-homogeneous Markov chain. Typically, the Markov chain will also be irreducible (although see discussion below). The HMC is designed such that the resulting Markov chain satisfies detailed balance with $\pi = P$.

Theorem 2.2.14. *The HMC algorithm produces a Markov chain $x_0, x_1, x_2, \ldots$ that satisfies the detailed balance condition with $\pi = P$.*

Proof. This will be proved by showing that the HMC algorithm is a special case of the Metropolis algorithm, so the result follows from theorem 2.2.8.

Suppose the chain is at position x. HMC proposes a transition to a new position y by drawing a random momentum $p \sim Q$ and numerically integrating Hamilton's equations using the leapfrog method. The integration is deterministic; let $y, q = I(x; p)$ denote the result of integrating from an initial state (x, p) to a final state (y, q). Therefore, HMC is

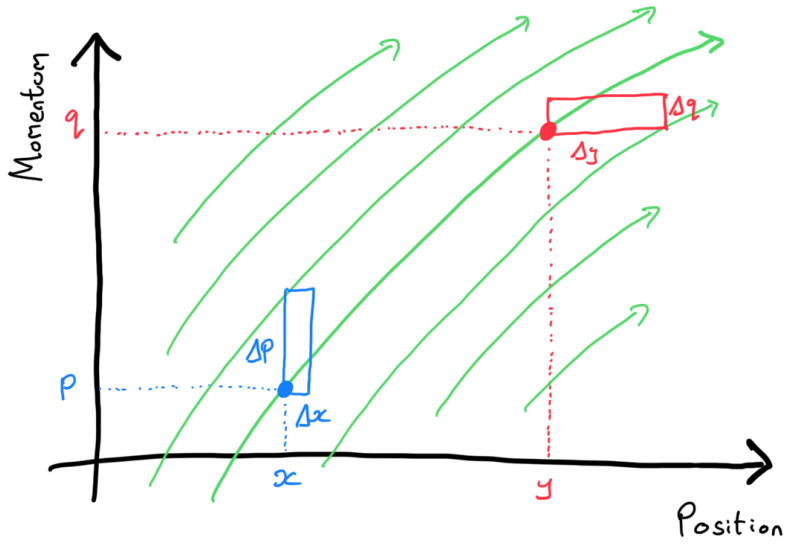
effectively using a proposal distribution with a PDF

$$\text{Proposal}(y|x) = \frac{Q(p)}{\left|\frac{\partial y}{\partial p}\right|}, \tag{2.54}$$

The approximate leapfrog Hamiltonian dynamics is exactly reversible (lemma 2.2.12), so $x, p = I(y, -q)$. Therefore, the proposal from y to x has PDF

$$\text{Proposal}(x|y) = \frac{Q(-q)}{\left|\frac{\partial x}{\partial q}\right|}, \tag{2.55}$$

The fact that the approximate leapfrog Hamiltonian dynamics is exactly volume preserving (lemma 2.2.13), means that the Jacobian terms in Eqs. 2.54 and 2.55 are equal; $\left|\frac{\partial y}{\partial p}\right| = \left|\frac{\partial x}{\partial q}\right|$. This is illustrated in $1+1$ dimensional phase space sketch below where the flow of the Hamiltonian dynamics is indicated in green and $\Delta x \Delta p = \Delta y \Delta q$.



Combining Eqs. 2.54 and 2.55, and using both the volume-preserving property and the fact that $Q(p) = Q(-p)$, gives

$$\frac{\text{Proposal}(x|y)}{\text{Proposal}(y|x)} = \frac{Q(p)}{Q(q)}. \tag{2.56}$$

For HMC to be a valid MH step, we must accept the proposed point with probability

$$a(y,x) = \min\left(1, \frac{P(y)\text{Proposal}(x|y)}{P(x)\text{Proposal}(y|x)}\right) \tag{2.57}$$

$$= \min\left(1, \frac{P(y)Q(q)}{P(x)Q(p)}\right) \tag{2.58}$$

$$= \min\Big(1, \exp\left[\mathcal{H}(x,p) - \mathcal{H}(y,q)\right]\Big). \tag{2.59}$$

But this is just the acceptance probability of the HMC algorithm (Alg. 2.9, line 11). □

Compared to the MH algorithm, HMC has the advantage that the user doesn't have to provide a proposal distribution.

The counter-intuitive starting point of HMC was the doubling the dimensionality of the space to be sampled. You would have been forgiven for thinking that this makes the sampling problem harder. However, recall the discussion in the introduction; Monte Carlo methods converge as $n^{-1/2}$, where n is the number of independent samples, regardless of dimensionality. The hard part is obtaining independent samples! HMC increases the dimensionality in such a way that makes it possible to efficiently propose points across large distances in the parameter space with high acceptance probabilities. This can help the chain to explore the space more efficiently than, say, the standard MH or Gibbs algorithms. Even when proposing large steps across parameter space, the acceptance probability $a = \min(1, \exp(\mathcal{H}_{\text{initial}} - \mathcal{H}_{\text{final}}))$ can still be extremely high because the (discretised) Hamiltonian dynamics (approximately) conserves the Hamiltonian. Under favourable conditions, this can allow HMC to produce more independent samples that other algorithms for the same computational cost.

HMC does have some drawbacks. It is considerably more complicated than the Gibbs or MH algorithms we have seen previously and harder to implement.

Another major drawback of HMC is that it requires the derivatives of $-\log P(x)$ w.r.t. x. This has historically been a major obstacle to the application of HMC to many problems, but the increasing use of automatic differentiation may help here. Also, because the HMC algorithm uses derivatives, it can only be applied to smooth target distributions. (This was not a requirement of the Gibbs or MH algorithm.)

Another drawback is that the user has to tune the values of several algorithm parameters: the time step Δt, the number of integration steps L, and the mass matrix M (this is often chosen to simply be the identity, $M = m\mathbb{I}_d$). Tuning Δt, L and m is important for the performance of the algorithm. If $L\Delta t$ is too small, then the algorithm takes small steps

and requires many iterations to explore the space. If L is too large, then time is wasted on unnecessary integration at each iteration as the particle oscillates around the potential well. If Δt is too large, then the numerical integration becomes inaccurate, the Hamiltonian is not conserved, and the acceptance probability decreases [13].

2.2.8.1 The No U-Turn Sampler (NUTS)

As with Gibbs and MH, there are many variants of the HMC. One particularly important variant is the No U-Turn Sampler (NUTS) which helps to automate the choice for L [14].

2.2.9 Convergence diagnostics

This section discusses some of the issues encountered when implementing MCMC methods. These topics are themselves not really part of the subject of stochastic sampling, but are important practical considerations in its applications. The presentation in section is therefore of a slightly different style; less formal, more practical.

Most theorems about the convergence of Markov chains apply only in the limit of infinite chains. It is hard to say anything definite about the finite length chains that are necessarily used in practice. But we must stop evolving our chains eventually and attempt to draw independent samples from them. This section discusses techniques that try to tell when a chain has reached its stationary distribution and how many independent samples are in the chain. Techniques for accelerating the approach to equilibrium and reducing the IAT are also discussed.

2.2.9.1 The burn in period

Suppose we have run our Markov chain for a while. Focusing on a single parameter, plotting it as a function of the index in the chain gives a *trace plot*. Typically, this looks something like Fig. 2.5. Trace plots are useful for assessing the convergence and diagnosing problems with MCMCs; many packages include functionality for producing them automatically.

[13] Potentially even worse, a particularly unlucky choice of Δt and L can lead to the chain moving through a periodic sequence of points in the sample space. In this case the resulting Markov chain will fail to be irreducible (see Sec. 3.3.6.2). However, this is unlikely to happen in practice.

[14] Hoffman & Gelman (2014) "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo", Journal of Machine Learning Research, **15** 1593-1623, link

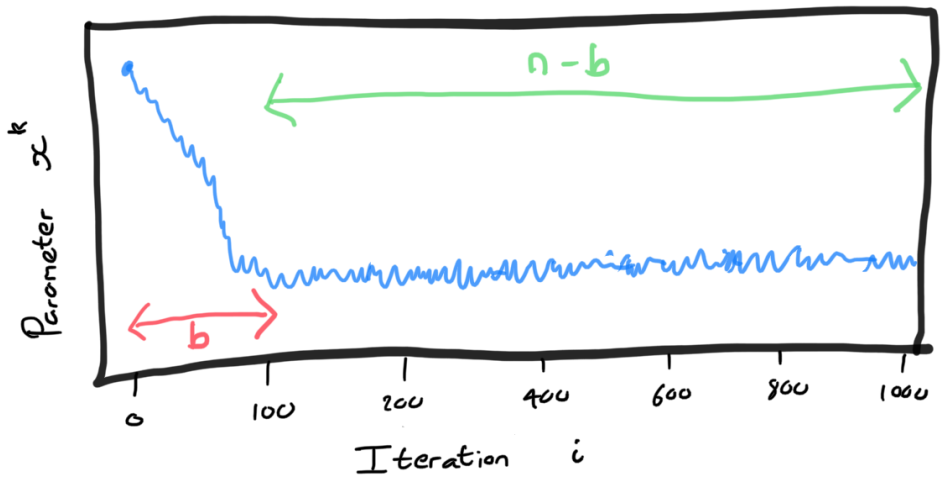


Figure 2.5: A typical MCMC trace plot. The value of one parameter, x^k, is plotted as a function of the chain index i. The burn in period is the early part of the evolution that is not representative of the rest of the chain.

The first part of the chain is called the *burn in*. This is the initial, transient part of the Markov chain's evolution which "remembers" the starting position. The burn in is not representative of the rest of the chain and is therefore not representative of the target distribution. The burn in should be discarded.

The length of the burn in, i.e. the number of iterations b that should be discarded, should be determined by inspecting the trace plots for all parameters and choosing the largest b. The trace plots for different parameters may look very different.

2.2.9.2 The Gelman-Rubin Statistic

Hereafter, it will be assumed that that burn in has been already removed from the chain, leaving us with a chain of length n.

After removing the burn in, it is hoped that the remainder of the chain has converged to the stationary, target distribution. But how to tell if this is the case?

One approach is to evolve several Markov chains and to check if they look statistically the same. If all the chains converge to the same distribution, then this is probably (hopefully!)

the target distribution. Checking if all the chains have converged to the same distribution can be done visually (e.g. by overlaying trace plots from different chains) or statistically with the *Gelman-Rubin* (GR) *statistic* [15]. The GR statistic checks for consistency between the intra chain and inter chain estimates of the variance of an arbitrary function $f(x)$.

Suppose we have N_c chains of length n (after removing the burn in). These points are denoted x_{ij}, where $j = 0, 1, \ldots, N_c - 1$ labels the chain and $i = 0, 1, \ldots, n-1$ labels points in the chain. For a real-valued function $f(x)$ the corresponding GR statistic is

$$R_f \approx \sqrt{\frac{B}{nW}}, \tag{2.60}$$

where the *between-chain* and *within-chain* variances are given respectively by

$$B = \frac{n}{N_c - 1} \sum_{j=0}^{N_c-1} \left(\mathrm{E}[f]_j - \bar{f}\right)^2, \quad \text{and} \quad W = \frac{1}{N_c} \sum_{j=0}^{N_c-1} \mathrm{Var}[f]_j, \tag{2.61}$$

and where the sample means and variances of the individual chains are given by

$$\mathrm{E}[f]_j = \frac{1}{n} \sum_{i=0}^{n-1} f(x_{ij}), \quad \text{and} \quad \mathrm{Var}[f]_j = \frac{1}{n-1} \sum_{i=0}^{n-1} \left(f(x_{ij}) - \mathrm{E}[f]_j\right)^2, \tag{2.62}$$

and where $\bar{f} = \frac{1}{N_c} \sum_{j=0}^{N_c-1} \mathrm{E}[f]_j$ is the overall sample mean taken across all the chains.

Note, R_f depends on the function f. Although we often speak of "the" GR statistic this is really incorrect; the same collection of chains has a different GR statistic for every function f. Typically, we calculate the GR statistic for each parameter by choosing $f(x) = x^k$, for $k = 0, \ldots, d-1$ and let R_k denote these single-parameter GR statistics.

If the chains are all converged, then the GR statistic should be close to one. If $R_k \lesssim 1.2$ for all parameters k, this is a good indication that the chains have all converged.

2.2.9.3 The Autocorrelation Length

Once we are confident that our chains (with the burn in removed) have reached equilibrium, we then want to draw independent samples from these chains . We can't use neighbouring points in the chain because these will not be independent. We typically take every m^{th} point from the chain in a process called *thinning*. But how to choose m?

[15] Gelman & Rubin (1992) "A Single Series from the Gibbs Sampler Provides a False Sense of Security", Statistical Science **7**, 457–511, doi:10.1214/ss/1177011136

Suppose we have a chain of length n (after removing the burn in) and we use all the points in the chain to estimate the expectation of a function $f(x)$ using the Monte-Carlo sum

$$\langle f \rangle = \sum_{i=0}^{n} f(x_i). \tag{2.63}$$

If all points in the chain were independent, then the variance in this estimator would be

$$\mathrm{Var}[S_n] = \frac{1}{n}\sigma_f^2, \quad \text{where} \quad \sigma_f^2 = \int \mathrm{d}x \ \left(f(x) - \langle f \rangle\right)^2 P(x). \tag{2.64}$$

However, because the points are *not* independent the variance is larger by a factor $\tau_f > 1$ which is called the *integrated autocorrelation time* (IAT), or just the *autocorrelation length*.

The autocorrelation length can be computed using the *autocorrelation function*,

$$\rho_f(I) = \frac{c(I)}{c(0)}, \quad \text{where} \quad c(I) = \frac{1}{n-I} \sum_{i=0}^{n-I-1} \left(f(x_i) - \bar{f}\right)\left(f(x_{i+I}) - \bar{f}\right), \tag{2.65}$$

where $\bar{f} = \sum_{i=0}^{n-1} f(x_i)$ is the chain mean and I is called the *lag.*. The IAT is defined as

$$\tau_f = 1 + 2\sum_{I=0}^{n-1} \rho_f(I). \tag{2.66}$$

There are efficient FFT-based methods for calculating the IAT.

Note, τ_f depends on the function being integrated. The same chain has a different IAT for every function f. Typically, the IAT is calculated for each parameter, τ_k, by choosing $f(x) = x^k$. The maximum of these single-parameter IATs $\tau = \max_k(\tau_k)$ is called *the IAT*.

To be safe, we should probably choose the thinning factor to be a few times longer than the maximum single-parameter IAT. Setting $m = 10\,\lceil\tau\rceil$ should be pretty safe.

The largest possible number of independent samples we can obtain from a Markov chain after removing the burn in and thinning is $N_{\text{eff}} \approx (n-b)/m$.

2.2.9.4 Simulated Annealing and Thermodynamic Integration

Anything that can be done to reduce the IAT is useful, because reducing m increases the number of independent samples N_{eff} we can get from a chain of fixed length n. More bang for a fixed computational buck. *Simulated annealing* is one approach to reducing the IAT.

For a target distribution P, we can define the *annealed distribution* $\hat{P}$ as having the PDF $[P_{\mathcal{X}}(x)]^{\beta}$, where the quantity $T = 1/\beta$ is called the annealing *temperature*.

The term *annealing* is an analogy with the metallurgical technique where a material is subjected to controlled periods of heating and cooling to alter its properties.

Simulated annealing aims to reduce the autocorrelation length by smoothing out features of the target distribution. It can also be used for *thermodynamic integration*.

2.2.10 A CASE STUDY: Comparing Gibbs, MH and HMC

Let P be a strongly correlated ($\beta = 0.9$) two-dimensional Gaussian distribution;

$$P = \mathcal{N}\left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}1 & \beta\\ \beta & 1\end{pmatrix}\right), \quad \text{with} \quad P(x,y) \propto \exp\left(\frac{-\left[x^2+y^2-2\beta xy\right]}{2[1-\beta^2]}\right). \tag{2.67}$$

This simple distribution can be efficiently sampled using transform methods (see Example 2.1) so there is no reason to use any MCMC method. Nevertheless, we will use this simple target distribution as a testbed for the three MCMC algorithms described above.

Gibbs sampling. This requires sampling the 1-dimensional conditional distributions $x|y$ and $y|x$. Fortunately, this can be done easily in this case as the conditional distributions are also Gaussian. From the definition in Eq. 2.25, it follows that

$$P(x|y) \propto \exp\left(\frac{-(x-\beta y)^2}{2[1-\beta^2]}\right) \quad \Rightarrow \quad x|y \sim \mathcal{N}(\beta y, 1-\beta^2). \tag{2.68}$$

Because P is symmetric in x and y, the other 1-dimensional conditional distribution is identical, $y|x \sim \mathcal{N}(\beta x, 1-\beta^2)$.

A major benefit of the Gibbs method is that it is extremely easy to implement (provided we have the conditional distributions) and it has no free parameters for the user to tune.

A Markov chain initialised at the origin $(x_0, y_0) = (0, 0)$ is evolved for $n = 10^5$ iterations using the basic Gibbs sampling algorithm described in Sec. 2.2.6. A small part (20 iterations) of the chain is shown in Fig. 2.6.

Because Gibbs iterations use the conditional distributions the chain only moves horizontally or vertically and therefore can only make small steps in this strongly correlated target distribution. This leads to the chain performing an undesirable *random walk* behaviour

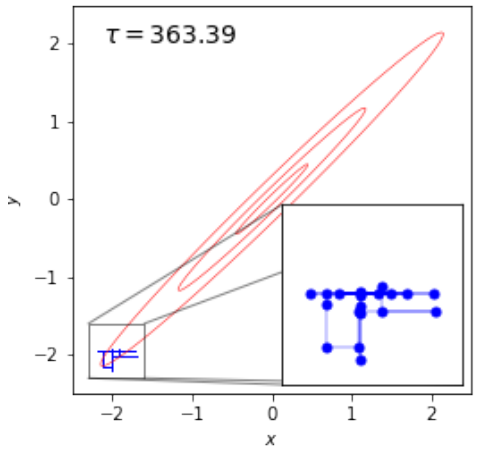


Figure 2.6: Illustration of the Gibbs algorithm showing 20 iterations of the Markov chain (blue) superposed on contours of the target distribution (red). The chain exhibits random walk behaviour; this is not desirable because it will take a long time for the chain to explore the full distribution. The IAT τ is also shown.

that take many iterations to *diffuse* across the target distribution. This can be quantified by the IAT which is rather long. Gibbs sampling struggles with highly correlated distributions.

If we had known in advance what correlation to expect in the target distribution, then we could have define new rotated variables $(u, v) = \mathcal{R}_{\pi/4}(x, y)$ to use in the Gibbs algorithm. This would have dramatically improved the performance of the Gibbs sampling.

Metropolis-Hastings sampling. The MH algorithm is slightly more involved to implement than Gibbs (although it doesn't require the conditional distributions) and requires us to choose a proposal distribution. Here, a symmetric, Gaussian proposal is chosen with $Q = \mathcal{N}([0, 0], \boldsymbol{\Sigma})$; three choices for the covariance matrix $\boldsymbol{\Sigma}$ are considered:

$$\boldsymbol{\Sigma}_{\text{Small}} = \begin{pmatrix} 0.05 & 0 \\ 0 & 0.05 \end{pmatrix}, \quad \boldsymbol{\Sigma}_{\text{Large}} = \begin{pmatrix} 1.5 & 0 \\ 0 & 1.5 \end{pmatrix}, \quad \text{and} \quad \boldsymbol{\Sigma}_{\text{Corr}} = \begin{pmatrix} 1 & \beta \\ \beta & 1 \end{pmatrix}. \tag{2.69}$$

Three Markov chains are initialised at the origin $(x_0, y_0) = (0, 0)$ is evolved for $n = 10^5$ iterations using the MH sampling algorithm described in Sec. 2.2.7 using the three proposals in Eq. 2.69. A small part (40 iterations) of each chain is shown in Fig. 2.7.

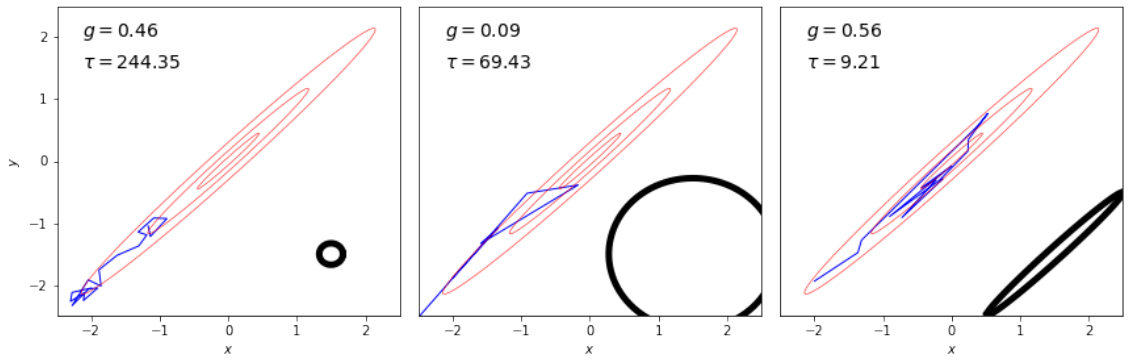


Figure 2.7: Illustration of the MH algorithm with Q_{small} (left), Q_{large} (centre), and Q_{corr} (right). Each plot shows 40 iterations (including duplicate points) of the Markov chain (blue) superposed on contours of the target distribution (red). Black lines show the shapes of the proposal distributions. Values of the acceptance fraction g and IAT τ are also shown.

Unlike Gibbs, MH algorithm can propose moves in any direction. Therefore, with a good choice of Q, it is possible for the chain to move across the target distribution in a small number of iterations. The price that we pay for this is that not all of the proposed moves are accepted and the chain sometimes doesn't move at all. If the proposal is too small, then we get a high acceptance fraction at the cost of small steps; the chain exhibits random walk behaviour (similar to the Gibbs chain) and we get a long IAT. Alternatively, if the proposal is too big it will frequently propose large steps in the wrong direction (e.g. ↘ or ↖) which will likely be rejected; the chain can take bigger steps at the cost of a lower acceptance fraction and because the chain often doesn't move at all, we still get a long IAT. We can try and get the best of both worlds (big steps and a high acceptance fraction with a smaller IAT) by carefully designing a proposal that preferentially proposes steps in the right directions ($\mathbf{\Sigma}_{\text{Corr}}$) but this requires that we know something extra about the shape of the target distribution, which isn't usually the case.

Note, the $\mathbf{\Sigma}_{\text{Corr}}$ proposal distribution is actually the same as the target distribution; i.e. we are stupidly using samples from P to produce samples from P! But this is just a toy example and is included here just show how well the MH algorithm can perform if we have a good proposal distribution.

Hamiltonian sampling. The HMC algorithm is more involved to implement. It also

requires the partial derivatives of the energy function, which for our target distribution are

$$E(x,y) = \frac{x^2 + y^2 - 2\beta xy}{2[1-\beta^2]} \quad \Rightarrow \frac{\partial E}{\partial x} = \frac{x - \beta y}{1-\beta^2}. \tag{2.70}$$

Because P is symmetric, the other derivative is identical; $\partial E/\partial y = \frac{y-\beta x}{1-\beta^2}$.

Three Markov chains are initialised at the origin $(x_0, y_0) = (0, 0)$ and are evolved for $n = 10^5$ iterations using the HMC sampling algorithms described in Sec. 2.2.8. We choose $M = \mathbb{I}_2$ and $\delta t = 0.05$. For the first chain, a small number of leapfrog iterations was used, $L_{\text{Short}} = 5$. For the second, $L_{\text{Medium}} = 50$. And for the third, $L_{\text{Long}} = 500$. A small part (10 iterations) of each chain is shown in Fig. 2.8.

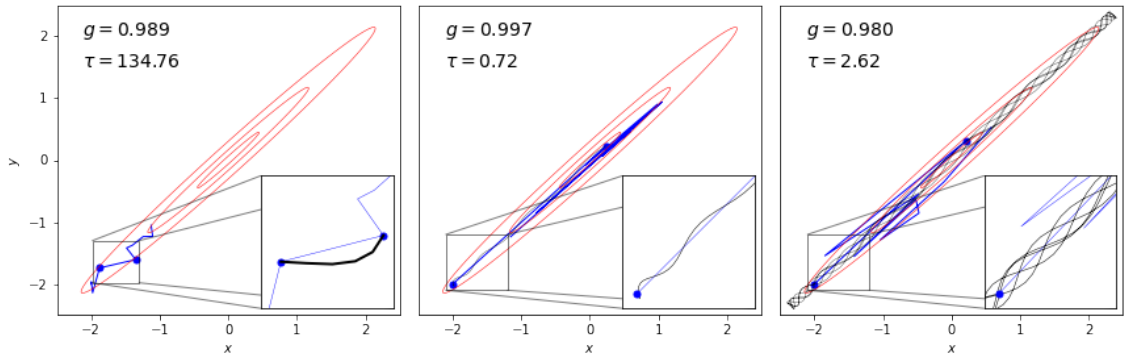


Figure 2.8: Illustration of the HMC algorithm with L_{Short} (left), L_{Medium} (centre), and L_{Long} (right). Each plot shows 10 iterations (including duplicate points) of the Markov chain (blue) superposed on contours of the target distribution (red). For one iteration, the trajectory of the numerical integration that made the proposal is shown in black. Values of the acceptance fraction g and IAT τ are also shown.

The clever feature of HMC is that it can take large steps in the target distribution (provided $L \gg 1$) while keeping the acceptance fraction high. If the numerical integration is accurate (meaning the Hamiltonian is approximately conserved) then $a \approx 1$. The ability to make large steps with a high acceptance fraction (see, for example, the centre or right panels of Fig. 2.8) leads to a small IAT. And unlike with the $\mathbf{\Sigma}_{\text{Corr}}$ MH method, HMC achieves without any detailed prior knowledge of the shape of the target distribution.

There are some drawbacks of HMC. The algorithm is more complicated than Gibbs or MH; each iteration requires the numerical integration of Hamilton's equations of motion. The

user must also provide the derivatives of target PDF (and the use of derivatives restricts the method to smooth target distributions). Another drawback is the number of inputs to the algorithm that the user must provide and tune by hand, such the mass matrix M and integration parameters L and ΔT. In particular, the choice of L is very important. Small L will mean the algorithm takes small steps and we are back to the undesirable situation where the Markov chain executes random walk behaviour (left panel of Fig. 2.8). Large L will lead to wasted time at each iteration numerically integrating the equations as the particle oscillates around the potential well (right panel of Fig. 2.8, black curve). The NUTS algorithm (Sec. 2.2.8.1) can be used to automate the choice for L.

A summary of the performances of the three MCMC algorithms is provided in the first table. On this simple toy problem, the best performing algorithm (in the sense of producing the greatest number of independent samples for the least computational time) is HMC with well-chosen values for L and Δt, closely followed by MH with a well-chosen proposal distribution Q. Crude computational time estimates were obtained from the wall time of $n = 10^5$ iterations of my own Python implementation of each algorithm running on a single laptop core. Care should be taken when drawing general conclusions about the three algorithms from this simple case study; their relative performance can vary considerably with different implementation and/or input parameter choices and for different target distributions, especially in high dimensions.

A summary of the algorithm requirements is provided in the second table.

Algorithm	IAT τ	acceptance frac g	num. ind. samples produced $N_{\text{eff}} = \frac{n}{10\lceil\tau\rceil}$	time per iter [ms]	time per sample [ms]
Gibbs	363.4	=1	27	0.6	2222
MH Q_{Small}	244.4	0.46	41	0.3	732
MH Q_{Large}	69.4	0.09	143	0.3	210
MH Q_{Corr}	9.2	0.56	1000	0.3	30
HMC L_{Short}	134.4	0.99	74	0.2	270
HMC L_{medium}	0.72	0.997	10000	0.6	6
HMC L_{Long}	2.6	0.98	3333	5.0	150

Algorithm	needs $x\|y$?	needs $\nabla \log P$?	user inputs
Gibbs	yes	no	-
MH	no	no	proposal dist, Q
HMC	no	yes	many: L, δt, M

Chapter 3

Model Comparison

In previous chapters we encountered encountered *parameter estimation* problems where we had available to us a suitable model that described the data. There, the Bayesian inference problem to find the best possible values of the model parameters and their uncertainties.

But what if we don't know what model to use? Or, what if we have several competing models which all claim to describe the data, how do we pick the best model? These are questions of *model selection*. As we will see in this chapter, these questions can also be answered with the framework of Bayesian inference.

3.1 Why Not Use the Maximum Likelihood?

A reasonable first thought would be, "why not use the model that fits the data best".

Let's formalise this idea a bit. Suppose we have two models: model A with some parameters λ_A, and model B with parameters λ_B. Given some data D we can right down the *likelihood* for both models: $\mathcal{L}(D|\lambda_A, A)$ and $\mathcal{L}(D|\lambda_B, B)$. We can numerically find the peak likelihood values of both of these distributions and consider the ratio

$$\text{Maximum Likelihood Ratio} = \frac{\max_{\lambda_A}\mathcal{L}(D|\lambda_A, A)}{\max_{\lambda_B}\mathcal{L}(D|\lambda_B, B)} = \frac{\mathcal{L}(D|\hat{\lambda}_A, A)}{\mathcal{L}(D|\hat{\lambda}_B, B)}\,. \tag{3.1}$$

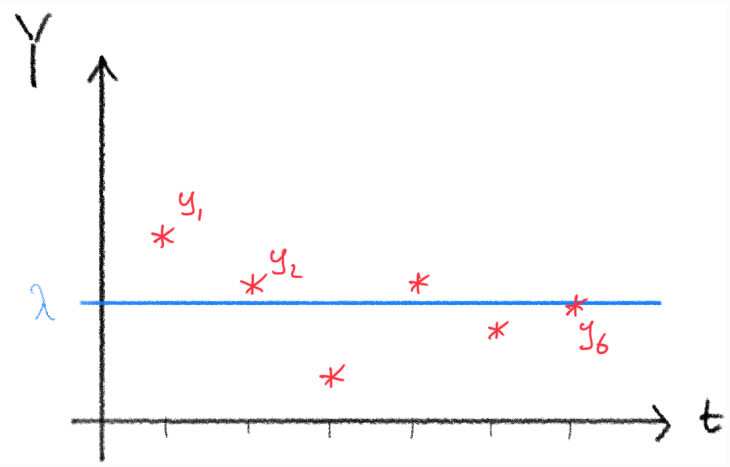
Why is it not enough to select model A is this ratio is larger than 1 and model B otherwise?

The problem with this approach is that it fails to account for the different complexity between models. Some models may have more or fewer parameters than others. Or parameters that can extend over different ranges. A model with a many parameters allowed to vary over wide ranges will generally fit the data well (i.e. it will have a high value of the likelihood). However, this doesn't necessarily mean that this model should be chosen over a simpler model that does almost as good a job at fitting the data.

Box 3.1: Fitting the data well isn't everything

Suppose we have a sequence of measurements $\mathcal{D} = \{y(t_1), y(t_2), \ldots, y(t_N)\}$ of a quantity $Y(t)$ made over an interval if time; where y. For simplicity, assume that all measurements have identical, independent Gaussian errors σ. For example,

$$\mathcal{D} = \{1.5,\ 1.2,\ 0.4,\ 1.1,\ 0.8,\ 1.0\} \quad \text{with} \quad \sigma = 1\,. \tag{i}$$



We have two models:

- A - the quantity $Y = 0$.
- B - the quantity $Y = \lambda$, where λ is a constant, unknown parameter.

For model B, choosing $\lambda = \text{average}(y_i)$ will fit the data best.

There could be more models: e.g. model C, with two parameters ($Y = \lambda_1 + \lambda_2 x$), model D with three ($Y = \lambda_1 + \lambda_2 x + \lambda_3 x^2$), and so on.

It is clear that in (almost[a]) any circumstances B will fit the data better than A (i.e. model B will have a higher peak likelihood value). Model B cannot possibly fit the data any worse than model A because A is nested inside B and can be recovered by setting $\lambda = 0$.

Similarly, model C and will almost surely fit the data better than B, and D will fit better than C, and so on indefinitely.

Does this necessarily mean the data favours model Z? No, we have failed to account properly for the different complexity of the models.

[a]The only exception being when the average of the measurements is exactly zero, in which case the two models will fit equally well

A model that fits the data well using a small number of free parameters is sometimes described as being *parsimonious*.

3.2 The Odds Ratio

Considering the material covered in previous chapters, it is perhaps no surprise that instead of the likelihood we are going to consider the *posterior odds ratio*,

$$\begin{aligned} \text{Posterior Odds Ratio} \equiv \mathcal{O}_{A,B} &= \frac{P(A|D)}{P(B|D)}\,, \\ &= \frac{P(D|A)}{P(D|B)} \times \frac{P(A)}{P(B)}\,. \end{aligned} \tag{3.2}$$

In going from the first to the second line we have used Bayes' theorem twice, once in the numerator and once in the denominator, and the common factor of $P(D)$ has cancelled between the two.

The second term in equation 3.2 is called the *prior odds ratio*. As is usual in a Bayesian analysis, our final answer depends partly on what we believed before performing the experiment. In some (but not all) situations we might want to try and be fair to both models and we might take the prior odds ratio to be unity.

Notice that in our first attempt in equation 3.1 we evaluated the ratio of the likelihoods at the maximum points. Here, in equation 3.2, we do *not* wish to do this. Instead of maximising over the free parameters we will marginalise over them, thereby allowing them

to take on all possible values.

$$\mathcal{O}_{A,B} = \frac{\int \mathrm{d}\lambda_A \; P(D, \lambda_A|A)}{\int \mathrm{d}\lambda_B \; P(D, \lambda_B|B)} \times \frac{P(A)}{P(B)} \,. \tag{3.3}$$

We will now use the *product rule* of probability to rewrite the integrands in this ratio in terms of more familiar probability distributions;

$$\mathcal{O}_{A,B} = \frac{\int \mathrm{d}\lambda_A \; P(D|\lambda_A, A) P(\lambda_A|A)}{\int \mathrm{d}\lambda_B \; P(D|\lambda_B, B) P(\lambda_B|B)} \times \frac{P(A)}{P(B)} \,. \tag{3.4}$$

Here we see again the likelihoods for the two models reappearing. We have also been forced to introduce *priors* for the free parameters in both models: $P(\lambda_A|A)$ and $P(\lambda_B|B)$.

But now we recognise the terms inside the integrals in both the numerator and denominator on equation 3.4, they are the likelihood times the prior for each model. Rewriting Eq. 3.4 in our earlier notation, we have

$$\mathcal{O}_{A,B} = \frac{\int \mathrm{d}\lambda_A \; \mathcal{L}(D|\lambda_A, A) \pi(\lambda_A|A)}{\int \mathrm{d}\lambda_B \; \mathcal{L}(D|\lambda_B, B) \pi(\lambda_B|B)} \times \frac{P(A)}{P(B)} \,. \tag{3.5}$$

But the integral of the likelihood times the prior over the full parameter space of the model is, by definition, just the *Bayesian evidence* for that model model. Using the notation introduced in chapter 1 for the evidence ($P(D|A) = Z_A$, and $P(D|B) = Z_B$) we can write the posterior odds as

$$\mathcal{O}_{A,B} = \frac{Z_A}{Z_B} \times \frac{P(A)}{P(B)} \,. \tag{3.6}$$

The take home message is that when comparing two models for the same data, *compute the ratio of the model evidences*. This ratio then acts to update our relative state of belief in the two models. Before performing the experiment we have the prior odds ratio. After performing the experiment we multiply this by the ratio of evidences to get the posterior odds ratio.

You might worry about the presence of the prior odds ratio in the final answer for $\mathcal{O}_{A,B}$. It might feel unscientific in some way to allow our prior beliefs to affect the answer in this way; "would it not be better to let the data decide?" Unfortunately, this is misguided; it is important to realise that our prior beliefs are always present (and rightly so) when we interpret any data. (Although, in some situations it might be appropriate to set the prior

odds ratio to unity.) In any case, in most situations the evidence ratio ends up dominating over the prior odds ratio. If one model fits the data significantly better than the other then it will have a much higher evidence and this will lead naturally lead us to favour that model regardless of the prior odds. The prior odds ratio is generally significant only in situations where both theories give similarly good fits to the data and/or the more complicated model gives a fit that is not sufficiently better to justify its extra complexity.

Of course, we could have chosen to define the odds ratio the other way around, with B in the numerator and A in the denominator. It makes no difference, we have

$$\mathcal{O}_{A,B} = \frac{1}{\mathcal{O}_{B,A}}\,. \tag{3.7}$$

Box 3.2: An example using the odds ratio

Consider again the data in example 3.1 and the models A and B.

Let us choose a prior odds ratio that treats both models as being equally likely;

$$\frac{P(A)}{P(B)} = 1\,. \tag{i}$$

First, consider the simpler model A. This model has no free parameters. We can write down the likelihood for this model as follows,

$$P(D|A) = \prod_{i=1}^{6} \frac{\exp\left(-\frac{1}{2}y_i^2\right)}{\sqrt{2\pi}} = 1.41 \times 10^{-4}\,. \tag{ii}$$

As there are no free parameters in this model there is no need to perform a marginalisation integral.

Second, consider the more complicated model B. This model has a single free parameter, λ. We can also write down the likelihood for this model as follows,

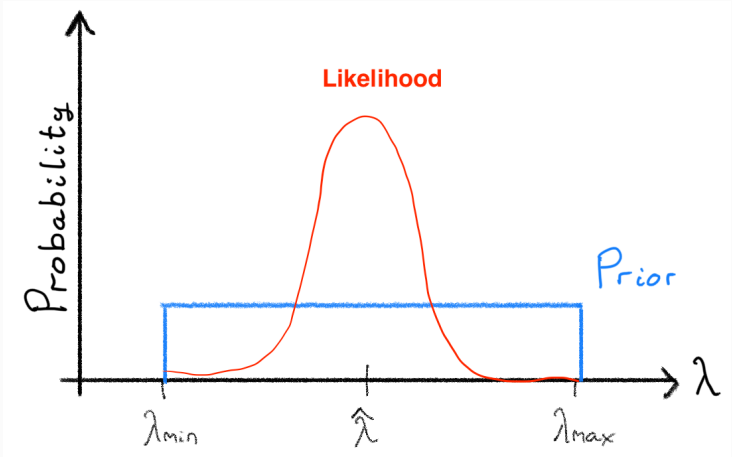
$$P(D|\lambda, B) = \prod_{i=1}^{6} \frac{\exp\left(-\frac{1}{2}(y_i - \lambda)^2\right)}{\sqrt{2\pi}}\,. \tag{iii}$$

The likelihood for model B has a maximum value of 2.84×10^{-3} at $\lambda = \hat{\lambda} = 1$. Therefore, if we were using the maximum likelihood ratio in equation 3.1 to perform model selection, we would favour the more complicated model B.

However, in order to compute the posterior odds ratio in equation 3.2 we must first evaluate the evidence for model B. This involves first specifying a prior for the free parameter λ. Here, I will choose a wide, uniform prior symmetric about zero;

$$P(\lambda|B) = \frac{1}{2\Lambda}\mathbb{1}_{(-\Lambda,\Lambda)}(\lambda), \tag{iv}$$

where $\Lambda \gg 1$. The likelihood and prior for model B are sketched in the figure.



The evidence for model B is now given by the integral

$$P(B|D) = Z_B = \int_{-\Lambda}^{\Lambda} \mathrm{d}\lambda \, \frac{1}{2\Lambda} \prod_{i=1}^{6} \left[\frac{\exp\left(-\frac{1}{2}(y_i - \lambda)^2\right)}{\sqrt{2\pi}} \right]. \tag{v}$$

In the limit of a wide, uninformative prior, $\Lambda \to \infty$, this integral becomes

$$P(B|D) \approx \frac{1}{2\Lambda} \int_{-\infty}^{\infty} \mathrm{d}\lambda \, \prod_{i=1}^{6} \left[\frac{\exp\left(-\frac{1}{2}(y_i - \lambda)^2\right)}{\sqrt{2\pi}} \right] = \frac{1.45 \times 10^{-3}}{\Lambda}. \tag{vi}$$

Therefore, the posterior odds ratio is given by

$$\mathcal{O}_{A,B} = \frac{P(A|D)}{P(B|D)} \times \frac{P(A)}{P(B)} = \frac{1.41 \times 10^{-4}}{1.45 \times 10^{-3}/\Lambda} \times 1 = 0.0973\Lambda\,. \qquad \text{(vii)}$$

Notice that if we choose Λ to be larger than about 10, then the odds ratio becomes greater than unity and we favour the simpler model A. However, if we choose $\Lambda \lesssim 10$ then we favour model B.

At first, it is surprising (and a little worrying) that the answer seems to depend so much on the apparently arbitrary choice of the prior range Λ. In particular, if we were to allow the free parameter in model B to take any value (i.e. $\Lambda \to \infty$) it looks like the posterior odds would always be large and we would be forced to favour model A, regardless of what the data looks like. While this is formally true, in practice models never have parameters that can vary over infinite ranges and after more careful consideration it is only right models with parameters that can vary over wide ranges (or multiple parameters) are properly penalised for this extra freedom.

In this example calculation, if we choose a reasonable value for Λ, we find a moderate posterior odds ratio ($\log \mathcal{O}_{A,B} \sim 0$) indicating there is no clear preference for either model. This raises an interesting question; how large (or small) does the odds ratio have to be for us to claim that there is "decisive" evidence in favour of a particular model? This is really more of a semantic question than a mathematical one; the answer depends on what you mean by "decisive". However, for Harold Jeffreys answer to this question it is interesting to consider the table shown at https://en.wikipedia.org/wiki/Bayes_factor.

Occam's Razor: At this point it is almost obligatory for me to mention the philosophical idea that can be stated something like "the simplest explanation is usually the right one". Some people like to see an analogy between this concept and the Bayesian model selection procedure described above. Some part of this idea seems to be captured by the fact that the evidence ratio penalises a complicated model with many parameters (or parameters that can take values spanning a wider range) and that does not fit the data sufficiently better than a simpler model. This idea is usually attributed to William of Ockham (c. 1287-1347) who was a philosopher, theologian, and Franciscan friar. Ockham did not express the idea mathematically and certainly did not use the idea to perform Bayesian model selection in the way described here.

3.3 Computing the Bayesian evidence

From Bayes' theorem

$$P(x|d) = \frac{P(d|x,M)P(x|M)}{P(d|M)} \tag{3.8}$$

$$= \frac{\mathcal{L}(d|x)\pi(x)}{Z}, \tag{3.9}$$

where the normalising constant, known as the *Bayesian evidence*, is given by

$$Z = \int_{\mathcal{X}} \mathrm{d}x \; \mathcal{L}(d|x)\pi(x). \tag{3.10}$$

Computing the evidence, $Z = P(\text{data}|\text{model})$, is the key step in Bayesian model selection. This involves integrating over the full space of model parameters, $x \in \mathcal{X}$. The integral is usually impossible to perform analytically and difficult to evaluate with standard numerical integration methods (e.g. quadrature) for all but the simplest models. In this part of the course we will discuss various approaches to calculating the evidence.

3.3.1 Analytic computation

Analytically computing the Bayesian evidence is generally only possible when the posterior happens to be a simple, well-known probability distribution. This can happen in simple (usually low-dimensional) problems if we are able to find a *conjugate prior*.

3.3.2 The Laplace Approximation

By now it should be clear that if we want to do Bayesian model comparison then we need a way of finding the evidence for each model that we want to consider.

In general, finding the Bayesian evidence involves numerically evaluating a complicated, high-dimensional integral over the parameter space of the model. In all but the simplest possible problems, these types of integrals are impossible to perform analytically. Worse, they are often extremely difficult to evaluate numerically. In the next section (Sec. 3.3.5) we will discuss a numerical method for these integrals.

However, in this section we will describe a useful method for approximating the evidence called the *Laplace approximation*.

To recap, if our problem involves parameters $\vec{x}$, and we have prior and likelihood functions $\pi(\vec{x}) \equiv P(\vec{x})$ and $\mathcal{L}(\vec{x}) \equiv P(D|\vec{x})$ respectively, then the *unnormalised* posterior distribution $P(\vec{x}) \equiv P(\vec{x}|D)$ is given by

$$P(\vec{x}) = \mathcal{L}(\vec{x})\pi(\vec{x})\,, \tag{3.11}$$

then the normalising evidence $Z \equiv P(D)$ is given by the following integral;

$$Z = \int \mathrm{d}\vec{x}\; P(\vec{x})\,. \tag{3.12}$$

Let us suppose we know that this distribution has a maximum value at the parameters $\hat{\vec{x}}$. (It is usually relatively easy to find the parameters $\hat{\vec{x}}$ numerically.) We now Taylor expand the quantity $\log P(\vec{x})$ about this peak location. In order to keep things simple, initially let's work in just 1-dimension; i.e. when $\vec{x} = x$. We have

$$\log P(x) = \log P(\hat{x}) - \frac{c}{2}(x - \hat{x})^2 + \ldots\,, \tag{3.13}$$

where $c = -\mathrm{d}^2/\mathrm{d}x^2|_{x=\hat{x}} \log P(x)$. There is no linear term because we have chosen to expand about the peak where the first derivative vanishes. We have discarded terms $\mathcal{O}\left([x - \hat{x}]^2\right)$. Taking the exponential of this equation we find that we can approximate the unnormalised posterior as a Gaussian,

$$\log P(x) \approx P(\hat{x}) \exp\left(-\frac{c}{2}(x - \hat{x})^2\right)\,. \tag{3.14}$$

A Gaussian is simple to integrate analytically; we have

$$Z \approx P(\hat{x})\sqrt{\frac{2\pi}{c}}\,. \tag{3.15}$$

This is the Laplace approximation for the evidence in 1-dimension.

Notice that the Laplace approximation for the evidence in equation 3.15 is *not* invariant under a change of parameters. If we change variables $x \to x' = y(x)$, where $y(x)$ a suitable function of the original x, then the second derivative c transforms as $c \to c'$, where

$$c' = \frac{c}{\left(\frac{\mathrm{d}y}{\mathrm{d}x}\right)^2}\,. \tag{3.16}$$

If we instead use c' in equation 3.15 then we obtain a different approximation for Z. This is undesirable because the true evidence (defined in 3.12) is invariant under such a change of parameters; the accuracy of the Laplace approximation depends on which parameterisation we choose to use.

Exercise 3.1:
Derive the result in equation 3.16.

We can repeat this calculation for a general multivariate D-dimensional problem. In this case the parameters vector $\vec{x}$ has components x_i where $i = 1, 2, \ldots, D$. We can approximate the logarithm of the unnormalised posterior distribution by the following multivariate Taylor series where we have expanded about the peak,

$$\log P(\vec{x}) \approx \log P(\hat{\vec{x}}) - \frac{1}{2} C_{ij}(x_i - \hat{x}_i)(x_i - \hat{x}_i)\,. \tag{3.17}$$

The constants C_{ij} form an $D \times D$ matrix and are given by the negative of the matrix of second derivatives (a.k.a. the *Hessian*) of the log-posterior;

$$C_{ij} = -\frac{\partial^2}{\partial x_i \partial x_j}\Big|_{\vec{x}=\hat{\vec{x}}} \log P(\vec{x})\,. \tag{3.18}$$

The derivatives may be evaluated either analytically or numerically. Again, the integral of this multivariate Gaussian is a well known result and involves the determinant of the matrix $\mathbf{C} = C_{ij}$,

$$Z \approx P(\hat{\vec{x}}) \sqrt{\frac{(2\pi)^D}{\det \mathbf{C}}}\,. \tag{3.19}$$

This result is the full *Laplace approximation*. This is a useful semi-analytic method for approximating the Bayesian evidence. It is only semi-analytic because the location of the maximum $\hat{\vec{x}}$ (and possibly the derivatives C_{ij}) must still be found numerically; however, this is usually much easier than numerically evaluating a high dimensional integral.

Again, as with the 1-dimensional version, the *Laplace approximation* in equation 3.19 is *not* invariant under a change of parameterisation, $\vec{x} \to x' = x'(x)$.

3.3.3 Savage-Dickey Density Ratio

The Savage-Dickey density ratio is not actually a method for calculating the evidence, $Z = P(d|M)$. Instead, it allows to calculate the *evidence ratio* (also known as the *Bayes factor*) between two models, $\mathcal{B}_{1,2} = P(d|M_1)/P(d|M_2)$, in the special situation where one of the models, say M_1, is *nested* inside the other model, M_2.

The term *nested models* applies to the situation where a simple model M_1 can be recovered from a more complicated model M_2 by fixing the value(s) of one of more of its parameters. For example, suppose that model M_2 has parameters (ϵ, ϕ) (where $\epsilon \in \mathbb{R}$ and $\phi \in \mathbb{R}^d$) and that model M_1 has parameters ϕ (i.e. the dimensionality of M_1 is one lower than that of M_2) and that when the extra parameter of M_2 takes a specific values, say $\epsilon = 0$, the two models makes the same prediction for the data; i.e.

$$\mathcal{L}(d|\phi, M_1) = \mathcal{L}(d|\epsilon = 0, \phi, M_2). \tag{3.20}$$

The Savage-Dickey method also requires that we use consistent priors on the shared parameters of the two models. The priors must satisfy

$$\pi(\phi|M_1) = \pi(\phi|\epsilon = 0, M_2). \tag{3.21}$$

In practice, this is usually achieved by using a separable prior for the more complicated model, $\pi(\epsilon, \phi|M_2) = f(\epsilon)g(\phi)$ and then using $\pi(\phi|M_1) = g(\phi)$ as the prior for the simpler model.

The evidence for the simpler model M_1 can be written as

$$\begin{aligned}
Z_{M_1} &= P(d|M_1) && (3.22)\\
&= \int \mathrm{d}\phi\ \mathcal{L}(d|\phi, M_1)\pi(\phi|M_1) && (3.23)\\
&= \int \mathrm{d}\phi\ \mathcal{L}(d|\phi, \epsilon = 0, M_2)\pi(\phi|\epsilon = 0, M_2) && (3.24)\\
&= P(d|\epsilon = 0, M_2) && (3.25)\\
&= \frac{P(\epsilon = 0|d, M_2)}{P(\epsilon = 0|M_2)} P(d|M_2), && (3.26)
\end{aligned}$$

where on the second line we have used Eqs. 3.20 and 3.21 and on the final line we have used Bayes' theorem. The evidence for the more complicated model M_2 is $Z_{M_2} = P(d|M_2)$.

Therefore, the Bayes factor is given by

$$\mathcal{B}_{1,2} = \frac{Z_{M_1}}{Z_{M_2}} \tag{3.27}$$

$$= \frac{P(\epsilon = 0|d, M_2)}{P(\epsilon = 0|M_2)}. \tag{3.28}$$

This implies that Bayes factor is given by the ratio of the 1-dimensional posterior PDF evaluated at $\epsilon = 0$ to the 1-dimensional prior PDF evaluated at $\epsilon = 0$.

The prior is generally chosen to be a simple analytical function in which case the denominator of Eq. 3.28 is easy to evaluate. The posterior in the numerator is harder. An MCMC method can be used to sample the posterior for model M_2 to obtain independent samples $(\epsilon_i, \phi_I) \sim P(\epsilon, \phi|d, M_2)$ for $i = 1, 2, \ldots, N$. Discarding the ϕ_i values, the remaining samples ϵ_i can be used to estimate the posterior PDF on $P(\epsilon|d, M_2)$ which can then be evaluated.

Any method can be used to estimate the density $P(\epsilon|d, M_2)$ from the samples ϵ_i. For example, kernel density estimation (KDE) is a common choice. Care must be taken when the value $\epsilon = 0$ lies on a boundary of the prior $f(\epsilon)$ to avoid biasing the density estimate.

3.3.4 Thermodynamic integration

Thermodynamic integration[1] is an MCMC-based method than can be use to estimate the Bayesian evidence integral in Eq. 3.10.

Thermodynamic integration introduces an *annealing*, or *inverse temperature* parameter β into the likelihood. The modified *annealed likelihood* is defined as $\mathcal{L}(d|x, \beta) = \mathcal{L}(d|x)^\beta$ where $0 < \beta \leq 1$. The notation here is chosen to be reminiscent of that in statistical mechanics where $\beta^{-1} = k_\mathrm{B}T$ commonly appears in, for example, the Boltzmann distribution. We then write down a modified version of Bayes' theorem for the model parameters x conditioned on a fixed value of β; this modified posterior is given by

$$P(x|d, \beta) = \frac{\mathcal{L}(d|x)^\beta \pi(x)}{Z(\beta)}, \tag{3.29}$$

$$\text{where} \quad Z(\beta) = \int_{\mathcal{X}} \mathrm{d}x \; \mathcal{L}(d|x)^\beta \pi(x). \tag{3.30}$$

The true, unmodified likelihood is recovered by setting $\beta = 1$ (this is called the low-temperature limit) and a flat likelihood equal to 1 everywhere is obtained by setting $\beta = 0$

[1] Goggans & Chi (2004) "Using Thermodynamic Integration to Calculate the Posterior Probability in Bayesian Model Selection Problems", AIP Conf Proc, 707, 59 https://doi.org/10.1063/1.1751356.

(this is called the high-temperature limit). Therefore, $Z(\beta = 1) = Z$ (this follows from Eq. 3.10) and $Z(\beta = 0) = 1$ (this follows from the fact that prior is a normalised probability distribution).

$$\frac{\mathrm{d}}{\mathrm{d}\beta} \log Z(\beta) = \frac{1}{Z(\beta)} \frac{\mathrm{d}Z}{\mathrm{d}\beta} \tag{3.31}$$

$$= \frac{1}{Z(\beta)} \int_{\mathcal{X}} \mathrm{d}x \; \pi(x) \mathcal{L}(d|x)^{\beta} \log \mathcal{L}(d|x) \tag{3.32}$$

$$= \int_{\mathcal{X}} \mathrm{d}x \; P(x|d, \beta) \log \mathcal{L}(d|x) \tag{3.33}$$

$$= \mathrm{E}_x[\log \mathcal{L}(d|x)|\beta] \tag{3.34}$$

On the last line we have written the integral using a notation that emphasises that this is the expectation of the quantity $\log \mathcal{L}(d|x)$ over realisations of the model parameters x drawn from the modified Bayesian posterior at a fixed value of the inverse temperature β. Using the fact that $Z(1) = Z$ and $Z(0) = 1$, we can integrate Eq. 3.31 to find

$$\log Z = \int_0^1 \mathrm{d}\beta \; \mathrm{E}_x[\log \mathcal{L}(d|x)|\beta]. \tag{3.35}$$

The method of thermodynamic integration relies on the ability of the MCMC methods described earlier in the course to approximate the expectation value of functions of the parameters.

We pick a series of increasing values for the inverse temperatures $0 \leq \beta_\mu \leq 1$ for $\mu = 0, 1, 2, 3, \ldots, M$ with $\beta_0 = 0$ and $\beta_M = 1$. The sequence $\beta_0^{-1}, \beta_1^{-1}, \beta_2^{-1}, \ldots, \beta_M^{-1}$ is called the *temperature ladder*. For each temperature a MCMC is run where the target distribution is the annealed posterior $P(x|d, \beta_\mu)$. These Markov chains are used to obtain N posterior samples $x_i^{(\beta_\mu)} \overset{\text{iid}}{\sim} P(x|d, \beta_\mu)$, for $i = 0, 1, \ldots, N-1$. The expectation in Eq. 3.34 is then approximated by the Monte-Carlo sum,

$$\mathrm{E}_x[\log \mathcal{L}(d|x)|\beta_\mu] \approx \frac{1}{N} \sum_{i=0}^{N-1} \log \mathcal{L}(d|x_i^{(\beta_\mu)}). \tag{3.36}$$

Finally, the evidence can be obtained by evaluating the 1-dimensional integral in Eq. 3.35 using the trapezium rule;

$$\log Z \approx \frac{1}{2} \sum_{\mu=1}^{M} \left(\mathrm{E}_x[\log \mathcal{L}(d|x)|\beta_\mu] + \mathrm{E}_x[\log \mathcal{L}(d|x)|\beta_{\mu-1}] \right) \Delta\beta_\mu, \tag{3.37}$$

where $\Delta\beta_\mu = \beta_\mu - \beta_{\mu-1}$.

This procedure allows us to use MCMC methods to evaluate the evidence integral by running multiple Markov chains at different temperatures and aggregating the results. Because thermodynamic integration requires an MCMC chain to be run at inverse temperature $\beta = 1$, a nice feature of the algorithm produces samples from the posterior at the same time as calculating the Bayesian evidence.

It's not hard to see how, in practice, this procedure can quickly become very expensive. In order for the trapezium rule integral to be accurate, a large number of temperatures need to be used. For each temperature, a Markov chain needs to be run to produce a large number of (independent) samples from the modified posterior to calculate (and reliably estimate the associated error) on the expectation quantity $\mathrm{E}_x[\log \mathcal{L}(d, x)|\beta]$.

Some performance gain can be achieved by running the Markov chains sequentially in order of decreasing temperature (i.e. increasing β) and using information from the previous temperature to initialise the next Markov chain thereby reducing the burn in period. Further improvements are possible by running all the different temperature chains in parallel and allowing them to exchange information during their evolution (in a way that ensures the detailed balance condition is preserved); this approach is known as *parallel-Tempered MCMC*.

3.3.5 Nested Sampling

In chapter 2 we saw many examples of Monte Carlo algorithms that can produce stochastic samples from a given target Bayesian posterior. *Nested sampling* is another example of such an algorithm[2].

There are two reasons why it makes sense to describe nested sampling separately here (as opposed to with the other Monte Carlo algorithms in chapter 2). Firstly, nested sampling also calculates the normalising Bayesian evidence, which makes it a ideal tool when there are multiple models to be compared; in fact, nested sampling is primarily an integration algorithm that produces stochastic samples as something of a by-product. Secondly, as we will see, the nested sampling algorithm needs to be supplemented by another Monte Carlo algorithm that is used to draw points from the prior (subject to a constraint on the likelihood). Many of the algorithms described in chapter 2 can be used for this purpose.

As always, the inputs to the Bayesian inference problem are the prior, $\pi(x)$, and the likelihood $\mathcal{L}(d|x)$, regarded as a function of the model parameters, $x \in \mathcal{X}$. The posterior is

[2]Skilling (2004) "Nested Sampling", AIP Conf. Proc. 735, 395–405 https://doi.org/10.1063/1.1835238

given by Bayes' theorem $P(x|d) \propto \mathcal{L}(d|x)\pi(x)$, where the normalisation constant, known as the *evidence*, is given by the integral

$$Z = \int \mathrm{d}x \; \mathcal{L}(d|x)\pi(x). \tag{3.38}$$

We will be interested in problems where $\mathcal{X}$ is high dimensional and this integral cannot be evaluated (or suitably approximated) analytically.

Let $\mathcal{L}_{\max}$ denote the maximum likelihood value;

$$\mathcal{L}_{\max} = \max_{x \in \mathcal{X}} \mathcal{L}(d|x). \tag{3.39}$$

Let $\xi(L)$ be the prior probability (sometimes called the *prior mass*) associated with points with a likelihood greater than a given value L; i.e.

$$\xi(L) = \int_{\left\{x \middle| \mathcal{L}(d|x) > L\right\}} \mathrm{d}x \; \pi(x). \tag{3.40}$$

From its definition, it follows that $\xi(L)$ is a *decreasing* function satisfying

$$\xi(0) = 1, \quad \text{and} \quad \xi(\mathcal{L}_{\max}) = 0. \tag{3.41}$$

In problems where x is continuous (and provided there are no regions where the likelihood function is exactly flat) the function $\xi(L)$ is *strictly decreasing* and can be inverted to give $L(\xi)$. (In discrete problems where a finite amount of prior probability is assigned to the same value of the likelihood, a small amount of noise, or *jitter*, can be added to the likelihood to obtain a strictly decreasing $\xi(L)$.)

Unfortunately, the functions $\mathcal{L}(d|x)$ and $L(\xi)$ can both reasonably be called the *likelihood*. The former is the usual likelihood expressed as a function of the model parameters x. The latter quantifies how much of the prior mass is associated with likelihoods above a certain threshold and is expressed as a function of the prior mass $0 \leq \xi \leq 1$.

The definition of the function $L(\xi)$ is illustrated in Fig. 3.1.

Consider the small range of the likelihood $(L, L + \mathrm{d}L)$. The quantity $\mathrm{d}\xi$ is prior mass associated with likelihood values in this range and is given by

$$\mathrm{d}\xi = \xi(L) - \xi(L + \mathrm{d}L) \tag{3.42}$$

$$= \int_{\left\{x \middle| L < \mathcal{L}(d|x) < L + \mathrm{d}L\right\}} \mathrm{d}x \; \pi(x) \tag{3.43}$$

Therefore, the range $\mathrm{d}x$ contributes an amount $L\mathrm{d}x$ to the total evidence. Summing all these contributions gives

$$Z = \int_0^1 \mathrm{d}\xi \; L(\xi). \tag{3.44}$$

At first, this appears somewhat miraculous; we have changed the high-dimensional integral in Eq.3.10 into the 1-dimensional integral in Eq.3.44. (This is reminiscent of what was done in the method of thermodynamic integration; see Eq.3.35.) Of course, we have just moved the difficulty into finding and inverting the function $\xi(L)$.

At first, this appears somewhat miraculous; we have changed a hig-dimensional integral into a 1-dimensional integral. Of course, we have just moved the difficulty into finding and inverting the function $\xi(L)$.

Exercise 3.2: Understanding the nested sampling function $\xi(L)$
Let $\mathbf{x} \in \mathbb{R}^n$ be a parameter vector in n dimensions. Use a prior on $\mathbf{x}$ that is uniform inside an n-ball of radius R centered on the origin; i.e.

$$\pi(\mathbf{x}) = \frac{1}{V_n R^n} \begin{cases} 1 & \text{if } |\mathbf{x}| < R \\ 0 & \text{else} \end{cases} , \tag{i}$$

where V_n is the volume of the unit n-ball. Find the functions $\xi(L)$ and $L(\xi)$ for the following likelihood function,

$$\mathcal{L}(d|\mathbf{x}) = \exp\left(\frac{-|\mathbf{x}|^2}{2}\right). \tag{ii}$$

Plot $\xi(L)$ and $L(\xi)$ for $R = 2.5$ in $n = 1$, 3, 10, and 30 dimensions. Notice how for large n regions of high likelihood occupy a small fraction of the prior. (Parameters with high likelihood are rare!)

Given an decreasing sequence of M points in the prior volume, $0 < \xi_M < \xi_{M-1} < \ldots < \xi_1 < 1$, and the associated likelihoods $L_\mu = L(\xi_\mu)$, the evidence integral in Eq. 3.44 can be approximated using the trapezium rule,

$$Z \approx \frac{1}{2}\sum_{\mu=1}^{M}(L_{\mu-1} + L_\mu)\Delta\xi_\mu, \tag{3.45}$$

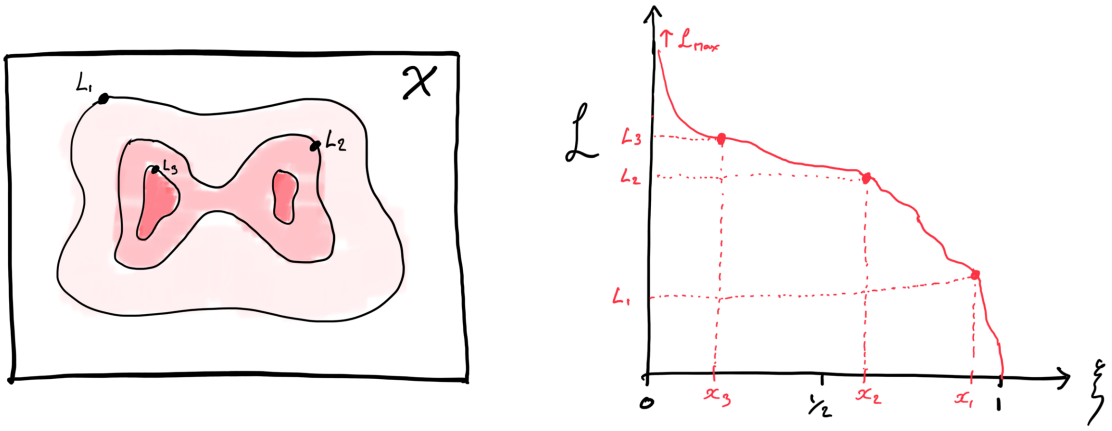


Figure 3.1: Sketches to illustrate the nested sampling function $\xi(L)$. *Left:* iso-probability contours of the likelihood function $\mathcal{L}(d|x)$ in the sample space $x \in \mathcal{X}$. *Right*: the likelihood as a function of ξ. For most realistic problems the function $L(\xi)$ is very strongly peaked at small values of ξ; parameters with high likelihood are rare!

where $\Delta\xi_\mu = \xi_{\mu-1} - \xi_\mu$ and we define $\xi_0 = 1$ and $L_0 = 0$. By relabelling terms in the sum, we can arange this into the slightly more convenient form

$$Z \approx \sum_{\mu=1}^{M} w_\mu L_\mu, \tag{3.46}$$

where the weights are given by $w_\mu = (\xi_{\mu-1} - \xi_{\mu+1})/2$ and we have defined $\xi_{M+1} = \xi_M$.

The nested sampling algorithm begins by initialising the parameter space location of a number of *live points* by drawing from the prior; $x_j \sim \pi$, independently for $j = 0, 1, \ldots, N_{\text{live}}-1$. (It is assumed that we already have the ability to sample from the prior, $x \sim \pi$; in practice, this is not a significant restriction because the prior is usually chosen to have a simple analytic form.) The number of live points, N_{live}, should be be a large in order to eventually obtain an accurate estimate for the evidence; for most Bayesian inference problems, typically N_{live} is chosen to be in the range $10^3 - 10^4$.

The nested sampling algorithm then proceeds iteratively. At each iteration the live point with the lowest value of the likelihood $L_i = \min(\{\mathcal{L}(d|x_j)|j = 0, 1, \ldots, N_{\text{live}}\})$ is replaced by a new live point with a higher likelihood. The position of this new live point is drawn from the prior, $x \sim \pi$, subject to the constraint that $\mathcal{L}(d|x)$ is larger than the likelihood of the point that has just been discarded. (This is the key step and is line 16 in Alg. 3.2 below.) By proceeding in this way, over time the collection of live points climbs up the likelihood

surface, eventually clustering near the peak $\mathcal{L}_{\max}$. The points that are discarded are called *deceased points* and form an increasing sequence with likelihood values $L_i < L_{i+1}$.

Figure 3.2 shows another sketch that attempts to illustrate how the nested sampling algorithm approximates the evidence integral.

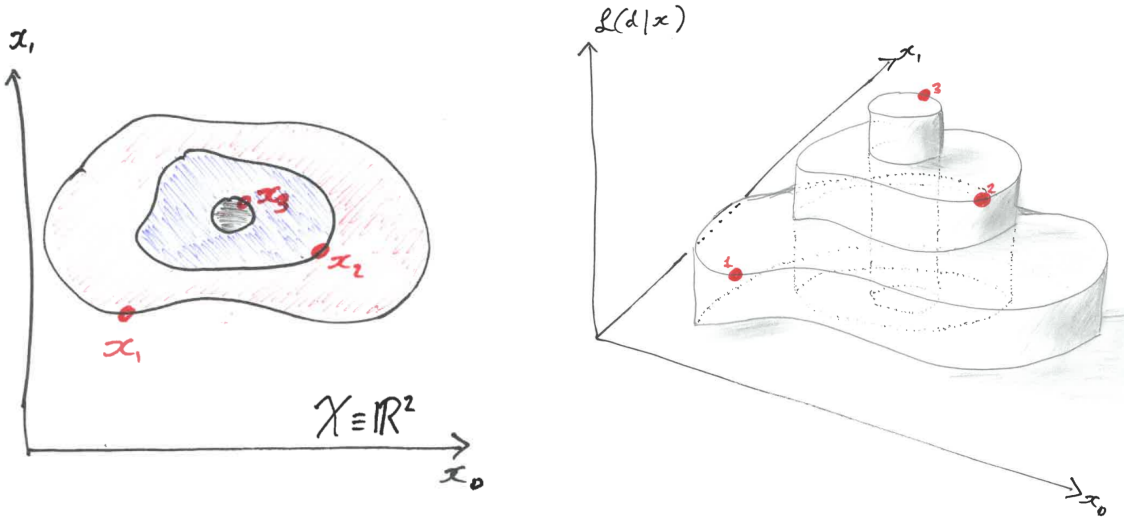


Figure 3.2: Sketches to illustrate nested sampling in 2 dimensions with a uniform prior on both parameters (x_0, x_1). *Left:* iso-probability contours in the space (x_0, x_1); one point on each contour is indicated using a red dot. *Right:* the likelihood as a function of (x_0, x_1). The red points are also shown in this sketch. Nested sampling approximates the area under smooth likelihood function in the way shown in the sketch. The sequence of points breaks the volume into a series of nested ; hence the name nested sampling.

In order to use this increasing sequence of likelihoods L_i in the expression for the evidence in Eq. 3.46 we need to know the associated values of ξ_i. This is determined probabilistically; at each iteration the prior mass shrinks by a factor, $\xi_i = t\xi_{i-1}$, where $0 < t < 1$ is a random variable that follows the distribution of the largest of N_{live} samples drawn uniformly from the interval $(0, 1)$. Therefore, we have

$$P(t) = \begin{cases} N_{\text{live}} t^{N_{\text{live}}-1} & \text{for } 0 < t < 1 \\ 0 & \text{otherwise} \end{cases} . \tag{3.47}$$

It is easy to show that for this distribution $E[\log t] = -1/N_{\text{live}}$ and $\text{Var}[\log t] = 1/N_{\text{live}}^2$. Therefore, we take $\xi_i \approx \exp(-i/N_{\text{live}})$. This will be an accurate estimate if N_{live} is large.

The nested sampling algorithm should be continued until the evidence integral is judged to have converged. The remaining contribution to the evidence from the surviving live points can be conservatively estimated as $\Delta Z \approx L_* \xi_i$, where L_* is the maximum likelihood value amongst the current set of live points. The algorithm is usually terminated when this drops below some user-specified tolerance.

Algorithm 3.1 Nested sampling stopping condition

```
1: procedure StoppingCondition(L_*, ξ_i, tol)
2:     if L_* ξ_i < tol then
3:         continue ← False                          ▷ Converged
4:     else
5:         continue ← True                           ▷ Continue
6:     end if
7:     return continue
8: end procedure
```

The output of the nested sampling algorithm is the estimate for the evidence integral, Z, and the listed of weighted samples $\{(W_i, x_i) | \text{for } i = 0, 1, \ldots, N_{\text{iter}}\}$, where N_{iter} is the number of iterations performed by the algorithm Alg. 3.2.

The nested sampling algorithm proceeds as follows.

Algorithm 3.2 The nested sampling algorithm

for $j = 0$ **to** $N_{\text{live}} - 1$ **do**
    $x_j \sim \pi(x)$ ▷ Initialise live points from prior
    $L_j \leftarrow \mathcal{L}(d|x_j)$
end for
$Z \leftarrow 0$ ▷ Initialise evidence estimate
samples $\leftarrow [\,]$ ▷ Empty list for weighted samples
$i \leftarrow 0$
continue $\leftarrow$ True
while continue **do** ▷ Iterate $i = 0, 1, 2, \ldots$
    idx $\leftarrow \text{argmin}(L_0, L_1, \ldots, L_{N_{\text{live}}-1})$ ▷ Worst live point
    $L_i \leftarrow L_{\text{idx}}$ ▷ Likelihood of deceased point
    $\xi_i \leftarrow \exp(-i/N_{\text{live}})$ ▷ Estimate enclosed prior mass
    $w_i \leftarrow \big(\exp[-(i-1)/N_{\text{live}}] - \exp[-(i+1)/N_{\text{live}}]\big)/2$
    $Z \leftarrow Z + w_i L_i$ ▷ Increment evidence
    AppendTo(samples, $[L_i w_i, x_{\text{idx}}]$) ▷ Store weighted sample
    $x \sim \pi(x|\mathcal{L}(d|x) > L_i)$ ▷ Sample constrained prior
    $x_{\text{idx}} \leftarrow x$ ▷ Replace deceased point
    $L_* \leftarrow \max(L_0, L_1, \ldots, L_{N_{\text{live}}-1})$
    continue $\leftarrow$ StoppingCondition(L_*, ξ_i, tol)
    $i \leftarrow i + 1$
end while

It should be clear from the above discussion that the main challenge for an practical implementation of the nested sampling algorithm is drawing samples from the prior subject to the constraint $L > L'$, where L' is current lowest likelihood value among all of the live points. This can be done in various different ways. In fact any of the MCMC algorithms described this course could be used for this purpose.

3.3.6 Sampling From The Constrained Prior

It should be clear from the above discussion that the main challenge for an practical implementation of the nested sampling algorithm is drawing samples from the prior subject to the constraint $L > L'$, where L' is current lowest likelihood value among all of the live points.

This can be done in various different ways. In fact any of the algorithms described in chapter 2 could be adapted for this purpose.

The following subsections will breifly describe a few methods that have been used for this purpose. This is an active area of research; some of the following subsections contain references to recent papers describing new code packages that tackle this problem.

3.3.6.1 Rejection Sampling From the Prior

The prior is usually chosen to be a simple, analytic distribution. In this case it is often trivial to sample from the (unconstrained) prior.

The obvious drawback of this naive approach is that it results in a steadily decreasing acceptance fraction as the algorithm progresses. At each iteration of the algorithm L' increases and hence the prior probability that satisfies the constraint $L > L'$ decreases and so does the probability of proposing an acceptable point.

3.3.6.2 MCMC

Perhaps the next simplest option is to use MH MCMC. The only modification that is needed to the algorithm Alg. 2.7 is that we do not accept any proposed point with a likelihood less than the current L_i.

3.3.6.3 Rejection Sampling From Bounding Distribution

Multinest https://arxiv.org/abs/0809.3437

3.3.6.4 Galilean Monte Carlo

https://arxiv.org/pdf/1312.5638.pdf

3.3.6.5 Slice Sampling

PolyChord https://arxiv.org/pdf/1502.01856.pdf

3.3.6.6 Normalising Flows

Nessai https://arxiv.org/abs/2302.08526

3.4 Summary

This chapter has discussed the topic of *Bayesian model comparison*. Here we summarise the main points of this chapter, but this time using the alternative terminology of *hypothesis testing*. Up until this point we have also been suppressing the other relevant prior assumptions (denoted $\mathcal{I}$, see equation 1.5) that are always present from our notation; in this summary these will be reinstated into our notation for completeness.

1. We are given some observations/measurements/data, $\mathcal{D}$ and prior assumptions $\mathcal{I}$.
2. We have several competing *hypotheses* which will be denoted $\mathcal{H}_0$, $\mathcal{H}_1$, $\mathcal{H}_2 \ldots$
   Under hypothesis $\mathcal{H}_i$ the data $\mathcal{D}$ is described by model $\mathcal{M}_i$ with free parameters $\vec{\theta}_i$.
3. For each hypothesis we can write down the *likelihood*, $P(\mathcal{D}|\vec{\theta}_i, \mathcal{H}_i, \mathcal{I})$.
4. For each hypothesis we must choose a *prior*, $P(\vec{\theta}_i|\mathcal{H}_i, \mathcal{I})$.
5. For each hypothesis we can write down *Bayes' theorem*,
   $$P(\vec{\theta}_i|\mathcal{D}, \mathcal{H}_i, \mathcal{I}) = \frac{1}{\mathcal{Z}_i} P(\mathcal{D}|\vec{\theta}_i, \mathcal{H}_i, \mathcal{I}) P(\vec{\theta}_i|\mathcal{H}_i, \mathcal{I}) \,, \tag{3.48}$$
   where we have defined the *evidence* for hypothesis $\mathcal{H}_i$ as
   $$\mathcal{Z}_i = \int \mathrm{d}\vec{\theta}_i \; P(\mathcal{D}|\vec{\theta}_i, \mathcal{H}_i, \mathcal{I}) P(\vec{\theta}_i|\mathcal{H}_i, \mathcal{I}) \,, \tag{3.49}$$
   and we recall that we are using the shorthand notation $\mathcal{Z}_i \equiv P(\mathcal{D}|\mathcal{H}_i, \mathcal{I})$.
6. For any pair of hypotheses $\mathcal{H}_i$ and $\mathcal{H}_j$, we can compute the *posterior odds ratio*,
   $$\mathcal{O}_{i,j} \equiv \frac{P(\mathcal{H}_i|\mathcal{D}, \mathcal{I})}{P(\mathcal{H}_j|\mathcal{D}, \mathcal{I})} = \frac{P(\mathcal{H}_i|\mathcal{I})}{P(\mathcal{H}_j|\mathcal{I})} \times \frac{\mathcal{Z}_i}{\mathcal{Z}_j} \,, \tag{3.50}$$
   which is the product of the *prior odds ratio* and the ratio of the *evidences*. The ratio of evidences acts to update our state of belief in light of the data $\mathcal{D}$ and quantifies the relative goodness-of-fit and parsimoniousness of the two models.
7. If the *posterior odds ratio* is greater (less) than 1 then we favour *hypothesis* $\mathcal{H}_i$ ($\mathcal{H}_j$).

Chapter 4

Advanced Topics

4.1 Hierarchical Bayesian Models

4.2 Gaussian Processes

A *Gaussian process* (GP) may be regarded as an infinite-dimensional generalisation of the more familiar finite-dimensional Gaussian distribution.

The main application of Gaussian processes in data science is for regression. Regression is a general term for predicting the values of some continuous quantities. A typically regression problem would involve predicting the value of some smooth function at a particular point given a set of (possibly noisy) observations at some other points. Regression includes other familiar problems such as *interpolation* and *extrapolation*.

Gaussian process regression (GPR) has several advantages compared to other, simpler methods of interpolation (for example, cubic splines):

- GPR can be easily interpolate data in any number of dimensions and on spaces with non-trivial topology,
- GPR works on unstructured data (there is no requirement that the sampled points be arranged in any particular way, such as on a regular square grid),
- GPR allows us to include some prior information about the function being interpo-

lated (such as its level of smoothness, periodic behaviour, or other symmetries) by choosing an appropriate kernel function, (in fact, more generally, GPR provides a very nice Bayesian view of interpolation),

- GPR allows the interpolated functions to become arbitrarily complicated, if the data demands this, while avoiding the danger of overfitting,

- as well as estimating the function value at the interpolated point, GPR also naturally provides a well-motivated estimate for the uncertainty on this value,

- GPs, and by extension GPR, have a lot of very nice formal properties which allows some properties of the interpolated functions to be proved mathematically.

The primary drawback of GPR is the computational costs involved when applying them to large datasets. As we will see, when interpolating a data set containing N points, GPR requires that we work with a covariance matrix of size $N \times N$ which requires a memory footprint of size $\mathcal{O}(N^2)$ to store and computational time of $\mathcal{O}(N^3)$ to invert.

We begin by recapping finite-dimensional Gaussian random variables.

4.2.1 Recap: univariate Gaussian distribution

The Gaussian distribution on a random variable $x \in \mathbb{R}$ has PDF

$$P(x) = \frac{\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}{\sqrt{2\pi\sigma^2}}, \tag{4.1}$$

where $\mu \in \mathbb{R}$ is the mean and σ^2 is the variance. The mean is arbitrary. However, the variance must be positive, $\sigma^2 > 0$.

A random variable from a Gaussian distribution is denoted

$$x \sim \mathcal{N}(\mu, \sigma^2). \tag{4.2}$$

4.2.2 Recap: multivariate Gaussian distribution

A finite d-dimensional Gaussian distribution is a probability distribution on a random variable $\boldsymbol{x} \in \mathbb{R}^d$ with PDF

$$P(\boldsymbol{x}) = \frac{\exp\left(-\frac{1}{2}[\boldsymbol{x}-\boldsymbol{\mu}]^{\mathrm{T}} \cdot \boldsymbol{\Sigma}^{-1} \cdot [\boldsymbol{x}-\boldsymbol{\mu}]\right)}{(2\pi)^{d/2}\,|\boldsymbol{\Sigma}|^{1/2}}, \tag{4.3}$$

where $\boldsymbol{\mu} \in \mathbb{R}^d$ is the mean vector and $\boldsymbol{\Sigma}$ is the $d \times d$ covariance matrix and where the dot $\cdot$ denotes vector/matrix multiplication. The mean vector is arbitrary. However, the covariance matrix $\boldsymbol{\Sigma}$ must be symmetric and *positive definite*. This is the extension of the condition $\sigma^2 > 0$ to more than one dimension.

Definition: a real, symmetric matrix $\boldsymbol{\Sigma}$ is *positive definite* if $\boldsymbol{z}^{\mathrm{T}} \cdot \boldsymbol{\Sigma} \cdot \boldsymbol{z} > 0$ for all nonzero vectors $\boldsymbol{z} \in \mathbb{R}^d$. A matrix which only satisfies the slightly weaker constraint $\boldsymbol{z}^{\mathrm{T}} \cdot \boldsymbol{\Sigma} \cdot \boldsymbol{z} \geq 0$ is said to be *positive semi-definite*.

Exercise 4.1:
Show that a real, symmetric matrix is positive definite (or positive semi-definite) if and only if all of its eigenvalues λ are strictly greater than zero, $\lambda > 0$ (or greater than or equal to zero, $\lambda \geq 0$).

A random vector from a multivariate Gaussian distribution is denoted

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}). \tag{4.4}$$

4.2.3 Gaussian Processes

Instead of distributions over a finite number of random variables (i.e. random vectors) we now wish to consider distributions over random functions. For some set $\mathcal{S}$ we consider functions $f : \mathcal{S} \to \mathbb{R}$ which assign a real number to every point in $\mathcal{S}$. We want to consider random functions; instead of the components of the vector, the random variables for a Gaussian process (GP) are the values of these functions evaluated points in $\mathcal{S}$.

We use the following definition of a GP.

Definition: for any set $\mathcal{S}$, a *Gaussian process* assigns a random value $f(x)$ to each point $x \in \mathcal{S}$ such that for any finite $n \in \mathbb{N}$, and any collection of points $\{x_i | i = 1, 2, \ldots, n\}$ in $\mathcal{S}$ the vector $(f(x_1), f(x_2), \ldots, f(x_n)) \in \mathbb{R}^n$ is distributed as a multivariate Gaussian.

A random function $f : \mathcal{S} \to \mathbb{R}$ from a GP with mean and covariance functions μ and k is denoted

$$f \sim \mathcal{GP}(\mu, k). \tag{4.5}$$

In this sense, GPs can be regarded as a probability distribution on the space of functions $f : \mathcal{S} \to \mathbb{R}$.

Although it appears to be rather abstract, the above definition of a GP is very clever; the use of arbitrary finite collections of points $x_1, x_2, \ldots, x_n$ allows us to talk precisely about infinite-dimensional, smooth functions while only working directly with finite-dimensional random vectors. (These finite objects are also what we work with numerically.)

Notice that in the definition of a GP the set $\mathcal{S}$ can be either finite or infinite. However, the finite-dimensional case simply reduces to the familiar multivariate Gaussian distribution and is therefore of little interest. Unless stated explicitly otherwise, we will usually be interested in the case $\mathcal{S} = \mathbb{R}$.

A GP is specified by specifying the mean and covariance matrix of all possible random vectors $\boldsymbol{f} = (f(x_1), f(x_2), \ldots, f(x_n)) \in \mathbb{R}^n$ produced by the GP. This can be done by defining the *mean function* $\mu : \mathcal{S} \to \mathbb{R}$ and the *covariance function* $k : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$. With these functions specified, we have

$$\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{K}), \tag{4.6}$$

where the mean and covariance matrix of $\boldsymbol{f}$ are given respectively by

$$\boldsymbol{\mu} = \big(\mu(x_1), \mu(x_2), \ldots, \mu(x_n)\big), \tag{4.7}$$

$$\text{and } \boldsymbol{K} = \begin{pmatrix} k(x_1, x_1) & k(x_1, x_2) & \ldots & k(x_1, x_n) \\ k(x_2, x_1) & k(x_2, x_2) & \ldots & k(x_2, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ k(x_n, x_1) & k(x_n, x_2) & \ldots & k(x_n, x_n) \end{pmatrix}. \tag{4.8}$$

The covariance function $k(x, x')$ is often called the *kernel function*, or simply the *kernel*.

The mean function $\mu(x)$ is arbitrary. However, the covariance function $k(x, x')$ must have a special property; for any set of points $x_1, x_2, \ldots, x_n \in \mathcal{S}$ the $n \times n$ matrix in Eq. 4.8 must be a valid covariance matrix; in particular, it must be positive definite. This is ensured by requiring that the covariance function is a *positive definite function*; this is the extension of the condition of a positive definite matrix to the case of infinite-dimensional GPs.

Definition: a covariance function $k : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ is a *positive definite function* if the $n \times n$ matrix with components $K_{ij} = k(x_i, x_j)$ is positive definite for all sets of $n \in \mathbb{N}$ points $x_1, \ldots, x_n$ in $\mathcal{S}$. (A similar definition exists for positive semi-definite functions.)

4.2.4 Examples of Gaussian Processes

The above definition of a GP is quite abstract. In order to get a feeling for what a GP is, it is useful to look at some examples of GPs.

In all these examples the mean function is chosen to be $\mu(x) = 0$, zero-mean GPs. Changing the mean function of a GP amounts to adding a constant function $\mu(x)$ to random functions $f(x) \sim \mathcal{GP}(0, k)$ and is not very interesting. However, as we will see, changing the covariance can have important consequences for the behaviour of GPs.

Squared Exponential Kernel. This is the most commonly used kernel function in GP applications because it is extremely simple and produces functions which are very smooth.

The *squared exponential* kernel function is

$$k_{\mathrm{SE}}(x, x') = \sigma_f^2 \exp\left(\frac{-(x - x')^2}{2\ell^2}\right), \tag{4.9}$$

where $\sigma_f^2 > 0$ is an overall amplitude parameter and $\ell > 0$ is a lengthscale parameter.

The squared exponential kernel is clearly symmetric, and is positive definite (proved later).

Fig. 4.1 shows several realisations of a GP with this covariance. Numerically, we can't work with the infinite-dimensional GP object. Therefore, in order to make this figure, we discretise the x-axis using N regularly spaced points $x_1, \ldots, x_N$, compute the covariance matrix $\boldsymbol{K}$ with components $K_{ij} = k_{\mathrm{SE}}(x_i, x_j)$, and draw realisations from the multivariate normal $\boldsymbol{y} \sim \mathcal{N}(0, \boldsymbol{K})$, where $\boldsymbol{y} \in \mathbb{R}^N$. For reasons of numerical stability, it is common to add a small *noise*, or *jitter*, term to the diagonal of large covariance matrices, so the covariance matrix becomes $K_{ij} = k_{\mathrm{SE}}(x_i, x_j) + J\delta_{ij}$, where $J \ll \sigma_f^2$.

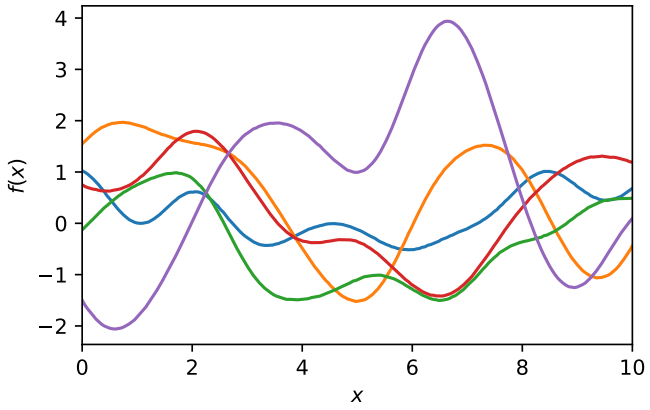


Figure 4.1: 5 realisations from a GP with a squared exponential kernel ($\sigma_f = 1$, $\ell = 1$).

The code used to produce Fig. 4.1 is shown here. This code emphasises an important point: despite their apparently abstract and infinite-dimensional nature, practical applications of GPs require us only to work with finite-dimensional normal distributions; the only significant external function imported by the code below is `scipy.stats.multivariate_normal`.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import multivariate_normal as norm

def squared_exponential_kernel(x1, x2, scale=1.0, lengthscale=1.0, jitter=None):
    """
    Parameters
    ----------
    x1, x2: arrays, shapes (n,) and (m,)
        arrays of points
    scale: float
        amplitude parameter
    lengthscale: float
        lengthscale parameter
    jitter: float
        small numerical tolerance to add to diagonal of k(x, x)

    Returns
    -------
    K: array, shape = (n, m)
        squared exponential covariance matrix k(x1, x2)
    """
    dist = x1[:, np.newaxis] - x2[np.newaxis, :]
    K = scale**2 * np.exp(-0.5*(dist/lengthscale)**2)
    if jitter:
        assert np.allclose(x1-x2, 0.), "jitter only when calculating k(x1, x1)"
        K += jitter * np.eye(x1.shape[0])
    return K

num_realisations = 5
x = np.linspace(0., 10., 201)
cov = linear_kernel(x, x, jitter=1.0e-5)
GP = norm(np.zeros_like(x), cov)

for i in range(num_realisations):
    plt.plot(x, GP.rvs())

plt.xlabel(r'$x$'); plt.ylabel(r'$f(x)$'); plt.show();
```

The definition of the squared exponential kernel given above is for a GP in one dimension (i.e. $\mathcal{S} = \mathbb{R}$). This can be generalised to $\mathcal{S} = \mathbb{R}^n$ by replacing $(x - x')$ with the distance between x and x' calculated with any suitable metric.

Linear Kernel. Another example of GP kernel function is the *linear* kernel

$$k_{\text{linear}}(x, x') = \sigma_f^2 \, x \, x', \tag{4.10}$$

where $\sigma_f^2 > 0$ is an overall amplitude parameter.

Lemma 4.2.1. *The linear kernel function $k_{\text{linear}}(x, x')$ is symmetric positive semi-definite.*

Proof. The kernel is clearly symmetric. In order to show the positive semi-definite property, consider the covariance matrix formed by k_{linear} on any finite set of distinct points $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)$; the covariance matrix is $\boldsymbol{K} = \boldsymbol{x} \cdot \boldsymbol{x}^{\mathrm{T}}$ (where we have set $\sigma_f = 1$ without loss of generality). It can be checked directly that this matrix has one eigenvalue $\lambda = |\boldsymbol{x}|^2 \geq 0$ corresponding to the eigenvector $\boldsymbol{x}$. Any other eigenvector $\boldsymbol{e}$ perpendicular to $\boldsymbol{x}$ satisfies $\boldsymbol{K} \cdot \boldsymbol{e} = 0$ and so must have an eigenvalue of 0. Together with the result of Exercise 4.1, this proves the result. □

In Fig. 4.2 a few realisations of a GP with the linear covariance function are plotted.

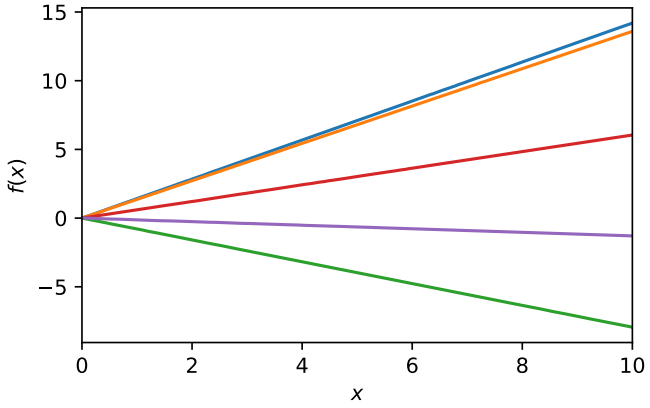


Figure 4.2: 5 realisations from a GP with a linear kernel ($\sigma_f = 1$).

Brownian Motion Kernel. Another familiar example of a GP (although not normally

introduced as such) is *Brownian motion*. The kernel function for Brownian motion is

$$k_{\text{Brownian}}(x, x') = \sigma_f^2 \min(x, x'), \tag{4.11}$$

where $\sigma_f^2 > 0$ is an overall amplitude parameter.

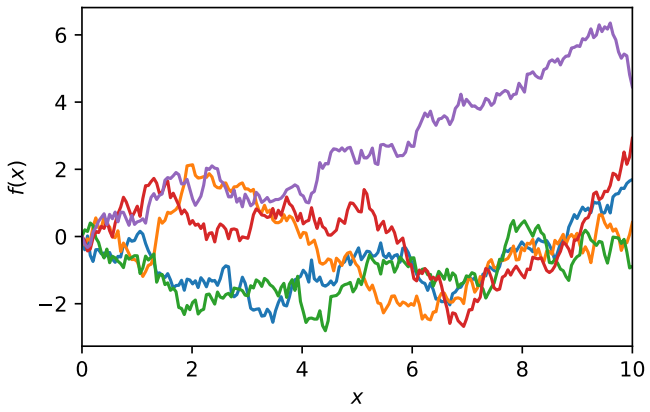


Figure 4.3: 5 realisations from a GP with a Brownian motion kernel ($\sigma_f = 1$).

Lemma 4.2.2. *The Brownian motion kernel function $k_{\text{Brownian}}(x, x')$ is symmetric positive semi-definite.*

Proof. The Brownian motion kernel can be written as

$$k_{\text{Brownian}}(x, x') = \sigma_f^2 \int_0^\infty \mathrm{d}t \; H(t; x) H(t; x'), \tag{4.12}$$

where

$$H(t; x) = \begin{cases} 1 & \text{if } t \leq x, \\ 0 & \text{if } t > x \end{cases}. \tag{4.13}$$

From the definition given in Sec. 4.2.3 above, in order to show that this kernel is positive semi-definite it is sufficient to show that the following quantity is non-negative for any

finite set of points $(x_1, x_2, \ldots, x_N)$ and for any vector $\boldsymbol{z} \in \mathbb{R}^N$.

$$\boldsymbol{z}^{\mathrm{T}} \cdot \boldsymbol{K} \cdot \boldsymbol{z} = \sum_{i,j} z_i z_j K_{ij} \tag{4.14}$$

$$= \sum_{i,j} \sigma_f^2 z_i z_j \int_0^\infty \mathrm{d}t \; H(t; x_i) H(t; x_j) \tag{4.15}$$

$$= \sigma_f^2 \int_0^\infty \mathrm{d}t \; \left(\sum_i z_i H(t; x_i) \right)^2 \tag{4.16}$$

$$\geq 0 \tag{4.17}$$

The final step follows because the integrand is squared and is therefore non-negative. □

In Fig. 4.3 a few realisations of a GP with the linear covariance function are plotted.

Periodic Kernel. The properties of the functions produced by a GP are controlled by the kernel function. This is what makes GPs useful in practice; we can choose a kernel that gives us random functions with specific properties.

A nice example of this is provided by the *periodic kernel*

$$k_{\mathrm{periodic}}(x, x') = \sigma_f^2 \exp\left(\frac{-2 \sin^2\left[\frac{\pi(x - x')}{T} \right]}{\ell^2} \right), \tag{4.18}$$

where $\sigma_f^2 > 0$ is an overall amplitude parameter, $\ell > 0$ is a lengthscale parameter and $T > 0$ is the period parameter.

The periodic kernel is clearly symmetric, it is also positive definite. (Proving this is given as an exercise below.)

In Fig. 4.4 a few realisations of a GP with the Brownian motion covariance function are plotted. All functions produced by this GP are period with period T.

4.2.5 Covariance functions

From the above examples it should be clear that the important properties of a GP determined by its covariance function, $k(x, x')$ [1].

[1] Strictly speaking, also by its mean function $\mu(x)$; however, changing $\mu(x)$ just amounts to trivially adding a constant function to all the outputs of a GP so we focus just on $k(x, x')$.

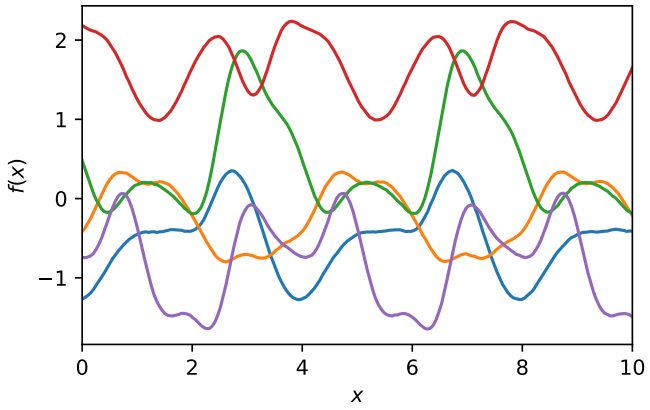


Figure 4.4: 5 realisations from a GP with a periodic kernel ($\sigma_f = 1$, $T = 4$, $\ell = 1$).

We have already seen several examples of covariance functions, or kernels. But not any function can be used as a kernel; it must satisfy the property of positive definiteness. It is difficult to come up with new covariance functions. It is also difficult to tell, in general, if a proposed new covariance function satisfies the positive definite property.

Given one or more "seed" covariance functions (that are already known to be positive definite) there are several ways that these can be combined to come up with new positive definite covariance functions. These methods are useful for constructing new covariance functions with specific properties.

Summation: for any two covariance functions, $k_1(x, x')$ and $k_2(x, x')$, the sum $k(x, x') = k_1(x, x') + k_2(x, x')$ is a new covariance function.

Product: for any two covariance functions, $k_1(x, x')$ and $k_2(x, x')$, the product $k(x, x') = k_1(x, x')k_2(x, x')$ is a new covariance function.

Warping: for any covariance function $k(x, x')$ on the space $\mathcal{S}'$ and a "warping" function $u : \mathcal{S} \to \mathcal{S}'$ the function $k(u(x), u(x'))$ is a new covariance function on the space $\mathcal{S}$. (I.e. the map u "pulls back" the kernel from $\mathcal{S}'$ to $\mathcal{S}$.)

Renormalisation: for any covariance function $k(x, x')$ and function $\alpha : \mathcal{S} \to \mathbb{R}$ the function $\alpha(x)\alpha(x')k(x, x')$ is a new covariance function.

Exercise 4.2: Constructing new covariance functions
Prove that the above procedures (summation, product, warping and renormalisation) do indeed generate new positive definite covariance functions.

Exercise 4.3: The periodic kernel
Starting from the squared exponential kernel in two dimension, use the "warping" procedure with $u(x) = \big(\cos(2\pi x/T), \sin(2\pi x/T)\big)$ to show that periodic kernel is a valid GP kernel. (You should assume that squared exponential kernel is positive definite, which we haven't proved yet. You can also use the trig identity $\big(\cos(x) - \cos(x')\big)^2 + \big(\sin(x) - \sin(x')\big)^2 = 4\sin^2\big((x - x')/2\big)$.)

We will now focus on kernels of GPs over $\mathcal{S} = \mathbb{R}^n$.

A kernel is said to be *stationary* if it depends only on the difference of the inputs; i.e. $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{\tau})$, where $\boldsymbol{\tau} = \boldsymbol{x} - \boldsymbol{x}'$. A stationary kernel is invariant under translations; i.e. $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x} + \boldsymbol{\Delta}, \boldsymbol{x}' + \boldsymbol{\Delta})$ for all $\boldsymbol{\Delta} \in \mathbb{R}^n$.

A stationary kernel is also said to be *isotropic* if it depends only on the Euclidean distance between the inputs; i.e. $k(\boldsymbol{\tau}) = k(r)$, where $r = |\boldsymbol{\tau}|$. An isotropic kernel is invariant under rotations as well as translations.

The squared exponential function is an example of kernel that is both stationary and isotropic.

Stationary kernels must satisfy $k(-\boldsymbol{\tau}) = k(\boldsymbol{\tau})$ (this follows trivially from the fact than $k(\boldsymbol{x}, \boldsymbol{x}')$ is symmetric). Stationary kernels $k(\boldsymbol{\tau})$ also have the following properties which follow directly from the property of positive definiteness:

$$(\textit{positivity}) \quad k(\boldsymbol{\tau} = 0) \geq 0, \tag{4.19}$$

$$(\textit{boundedness}) \quad k(\boldsymbol{\tau}) \leq k(\boldsymbol{\tau} = 0) \text{ for all } \boldsymbol{\tau} \in \mathbb{R}^d. \tag{4.20}$$

Although it can be hard in general to tell whether or not a proposed function $k(\boldsymbol{x}, \boldsymbol{x}')$ is positive definite, stationary kernels $k(\boldsymbol{\tau})$ admit a nice characterisation in terms of their Fourier transforms $\tilde{k}(\boldsymbol{f})$ which enables this question to be answered relatively easily.

Lemma 4.2.3. *A function $k : \mathbb{R}^n \to \mathbb{R}$, denoted $k(\boldsymbol{\tau})$, can be used to define a positive semi-definite stationary kernel $\kappa : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ as $\kappa(\boldsymbol{x}, \boldsymbol{x}) = k(\boldsymbol{x} - \boldsymbol{x})$, if the Fourier transform $\tilde{k}(\boldsymbol{f}) \geq 0$ for all $\boldsymbol{f} \in \mathbb{R}^n$.*

Proof. Supposing that the Fourier transform exists, we can write the function k in terms of its Fourier transform as

$$k(\boldsymbol{\tau}) = \int_{\mathbb{R}^n} \mathrm{d}\boldsymbol{f}\ \tilde{k}(\boldsymbol{f}) \exp\left(-2\pi i \boldsymbol{f} \cdot \boldsymbol{\tau}\right). \tag{4.21}$$

From the definition given in Sec. 4.2.3 above, in order to show that the kernel κ is positive semi-definite it is sufficient to show that the following quantity is non-negative for any finite set of points $(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N) \in \mathbb{R}^n$ and for any vector $\boldsymbol{z} \in \mathbb{R}^N$.

$$\begin{aligned}
\boldsymbol{z}^{\mathrm{T}} \cdot \boldsymbol{K} \cdot \boldsymbol{z} &= \sum_{ij} z_i z_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) && (4.22)\\
&= \sum_{ij} z_i z_j k(\boldsymbol{x}_i - \boldsymbol{x}_j) && (4.23)\\
&= \sum_{i,j} z_i z_j \int_{\mathbb{R}^d} \mathrm{d}\boldsymbol{f}\ \tilde{k}(\boldsymbol{f}) \exp\left(-2\pi i \boldsymbol{f} \cdot [\boldsymbol{x}_i - \boldsymbol{x}_j]\right) && (4.24)\\
&= \int_{\mathbb{R}^d} \mathrm{d}\boldsymbol{f}\ \tilde{k}(\boldsymbol{f}) \left| \sum_i z_i \exp\left(-2\pi i \boldsymbol{f} \cdot \boldsymbol{x}_i\right) \right|^2 . && (4.25)
\end{aligned}$$

This is certain to be positive (as required) if $\tilde{k}(\boldsymbol{f}) \geq 0$ for all $\boldsymbol{f}$. □

This gives us a powerful way to tell is a stationary function $k(\boldsymbol{\tau})$ is a valid positive definite kernel by computing its Fourier transform.

Exercise 4.4: The squared exponential kernel
By taking its Fourier transform, show that the squared exponential kernel is positive definite.

This result can be extended to nonintegrable functions that do not posses a Fourier transform using concepts from measure theory and strengthened to a necessary and sufficient condition for a function to be a valid positive definite kernel.

Theorem 4.2.4. ***(Bochner's theorem.)*** *A continuous function $k(\boldsymbol{\tau})$ is positive semi-definite if and only if it is the Fourier transform of a finite non-negative Borel measure μ on $\mathbb{R}^d$; i.e. if*

$$k(\tau) = \int_{\mathbb{R}^n} \mathrm{d}\mu(\boldsymbol{f}) \, \exp\left(-2\pi i \boldsymbol{f} \cdot \boldsymbol{x}\right). \tag{4.26}$$

4.2.6 Gaussian Process Regression

We will now show how the GPs that have been introduced and discussed in the previous sections can be used to perform regression.

The following property of multivariate Gaussian distributions is at the heart of *Gaussian process regression* (GPR); conditional distributions of Gaussians are also Gaussian.

Lemma 4.2.5. *Let $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Split the vector $\boldsymbol{x} \in \mathbb{R}^n$ into $\boldsymbol{x}^{\mathrm{T}} = (\boldsymbol{x}_1^{\mathrm{T}}, \boldsymbol{x}_2^{\mathrm{T}})$, where $\boldsymbol{x}_1 \in \mathbb{R}^{n_1}$ and $\boldsymbol{x}_2 \in \mathbb{R}^{n_2}$ and $n_1 + n_2 = n$. Similarly, split $\boldsymbol{\mu}^{\mathrm{T}} = (\boldsymbol{\mu}_1^{\mathrm{T}}, \boldsymbol{\mu}_2^{\mathrm{T}})$ and*

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{12}^{\mathrm{T}} & \boldsymbol{\Sigma}_{22} \end{pmatrix}, \tag{4.27}$$

where $\boldsymbol{\Sigma}_{11}$, $\boldsymbol{\Sigma}_{12}$ and $\boldsymbol{\Sigma}_{22}$ are $n_1 \times n_1$, $n_1 \times n_2$ and $n_2 \times n_2$ matrices respectively. The distribution of $\boldsymbol{x}_1$ conditioned on a fixed value of $\boldsymbol{x}_2$ is

$$\boldsymbol{x}_1 | \boldsymbol{x}_2 \sim \mathcal{N}\left(\boldsymbol{\mu}', \boldsymbol{\Sigma}'\right), \tag{4.28}$$

where $\boldsymbol{\mu}' = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12} \cdot \boldsymbol{\Sigma}_{22}^{-1} \cdot (\boldsymbol{x}_2 - \boldsymbol{\mu}_2)$ and $\boldsymbol{\Sigma}' = \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12} \cdot \boldsymbol{\Sigma}_{22}^{-1} \cdot \boldsymbol{\Sigma}_{12}^{\mathrm{T}}$.

Proof. The proof is simple, but messy. The joint PDF of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ is

$$P(\boldsymbol{x}_1, \boldsymbol{x}_2) = \frac{\exp\left(\frac{-1}{2}(\boldsymbol{x}_1^{\mathrm{T}} - \boldsymbol{\mu}_1^{\mathrm{T}}, \boldsymbol{x}_2^{\mathrm{T}} - \boldsymbol{\mu}_2^{\mathrm{T}}) \cdot \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{12}^{\mathrm{T}} & \boldsymbol{\Sigma}_{22} \end{pmatrix}^{-1} \cdot \begin{pmatrix} \boldsymbol{x}_1 - \boldsymbol{\mu}_1 \\ \boldsymbol{x}_2 - \boldsymbol{\mu}_2 \end{pmatrix}\right)}{(2\pi)^{n/2} \begin{vmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{12}^{\mathrm{T}} & \boldsymbol{\Sigma}_{22} \end{vmatrix}^{1/2}}. \tag{4.29}$$

Using the block matrix inversion formula

$$\begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{12}^{\mathrm{T}} & \boldsymbol{\Sigma}_{22} \end{pmatrix}^{-1} = \tag{4.30}$$
$$\begin{pmatrix} \left(\boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12} \cdot \boldsymbol{\Sigma}_{22}^{-1} \cdot \boldsymbol{\Sigma}_{12}^{\mathrm{T}}\right)^{-1} & -\left(\boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12} \cdot \boldsymbol{\Sigma}_{22}^{-1} \cdot \boldsymbol{\Sigma}_{12}^{\mathrm{T}}\right)^{-1} \cdot \boldsymbol{\Sigma}_{12} \cdot \boldsymbol{\Sigma}_{22}^{-1} \\ -\boldsymbol{\Sigma}_{22}^{-1} \cdot \boldsymbol{\Sigma}_{12}^{\mathrm{T}} \cdot \left(\boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12} \cdot \boldsymbol{\Sigma}_{22}^{-1} \cdot \boldsymbol{\Sigma}_{12}^{\mathrm{T}}\right)^{-1} & \boldsymbol{\Sigma}_{22}^{-1} + \boldsymbol{\Sigma}_{22}^{-1} \cdot \boldsymbol{\Sigma}_{12}^{\mathrm{T}} \cdot \left(\boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12} \cdot \boldsymbol{\Sigma}_{22}^{-1} \cdot \boldsymbol{\Sigma}_{12}^{\mathrm{T}}\right)^{-1} \cdot \boldsymbol{\Sigma}_{12} \cdot \boldsymbol{\Sigma}_{22}^{-1} \end{pmatrix},$$

and substituting into Eq. 4.30, the condition probability of $\boldsymbol{x}_1$ given $\boldsymbol{x}_2$ is proportional to

$$P(\boldsymbol{x}_1|\boldsymbol{x}_2) \propto \exp\left(\frac{-1}{2}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1 - \boldsymbol{\Sigma}_{12} \cdot \boldsymbol{\Sigma}_{22}^{-1} \cdot [\boldsymbol{x}_2 - \boldsymbol{\mu}_2])^{\mathrm{T}} \cdot \left(\boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12} \cdot \boldsymbol{\Sigma}_{22}^{-1} \cdot \boldsymbol{\Sigma}_{12}^{\mathrm{T}}\right)^{-1} \cdot (\boldsymbol{x}_1 - \boldsymbol{\mu}_1 - \boldsymbol{\Sigma}_{12} \cdot \boldsymbol{\Sigma}_{22}^{-1} \cdot [\boldsymbol{x}_2 - \boldsymbol{\mu}_2])\right), \tag{4.31}$$

where we have discarded factors that do not depend on $\boldsymbol{x}_1$. We recognise this as having the same $\boldsymbol{x}_1$ dependence as a multivariate normal distribution with mean $\boldsymbol{\mu}'$ and covariance $\boldsymbol{\Sigma}'$. Therefore, after renormalising, we must have the result $\boldsymbol{x}_1|\boldsymbol{x}_2 \sim \mathcal{N}(\boldsymbol{\mu}', \boldsymbol{\Sigma}')$. □

Graphically, the significance of this result is that if you take a slice through a n-dimensional multivariate Gaussian along any m-dimensional hyperplane (with $m < n$), then the PDF evaluated along the slice has the shape of another Gaussian. This is illustrated in Fig. 4.5

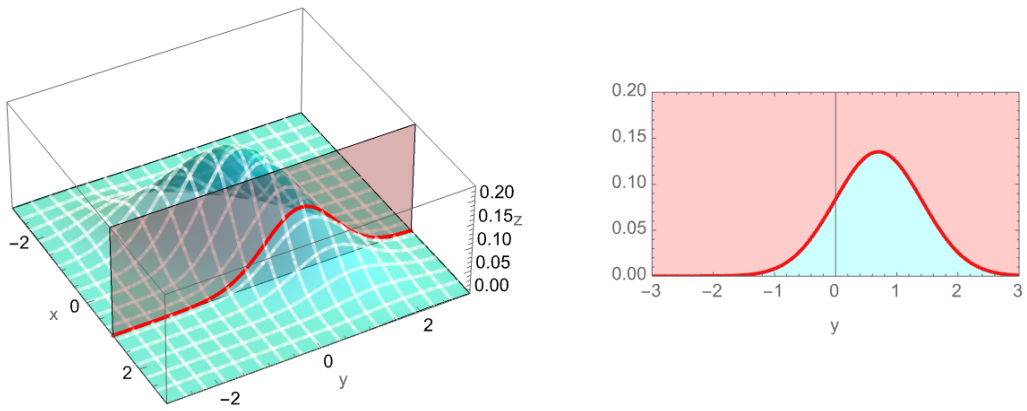


Figure 4.5: *Left:* The 2-dimensional Gaussian $(x, y) \sim \mathcal{N}([0, 0], [[1, \rho], [\rho, 1]])$, where $\rho = 0.7$ *Right:* Up to renormalisation by a constant factor, the function sliced along the line $x = 1$ is the 1-dimensional distribution $(y|x = 1) \sim \mathcal{N}(\rho, 1 - \rho^2)$.

So far we have seen several examples of random functions generated from a Gaussian process. We now turn to the problem of regression; given measurements of a function at some locations our task is to infer the value of a function at some new location.

Suppose we have measured the value of the function at n points $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ and the measured function values are $\boldsymbol{f} = (f(x_1), f_{x_2}, \ldots, f(x_n))$. Our task is to predict the value of the function $f(x_*)$ at some new location x_*.

We use the zero-mean GP $f \sim \mathcal{GP}(0, k)$ as our prior on the function. By the defining property of the GP, the joint distribution of the function at the measured and unmeasured locations is the Gaussian

$$f(x_*), \boldsymbol{f} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{pmatrix} k_{**} & \boldsymbol{k}_*^{\mathrm{T}} \\ \boldsymbol{k}_* & \boldsymbol{K} \end{pmatrix}\right), \tag{4.32}$$

where

$$k_{**} = k(x_*, x_*), \tag{4.33}$$

$$\boldsymbol{k}_*^{\mathrm{T}} = \big(k(x_*, x_1), k(x_*, x_2), \ldots, k(x_*, x_n)\big) \tag{4.34}$$

$$\boldsymbol{K} = \begin{pmatrix} k(x_1, x_1) & k(x_1, x_2) & \ldots & k(x_1, x_n) \\ k(x_2, x_1) & k(x_2, x_2) & \ldots & k(x_2, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ k(x_n, x_1) & k(x_n, x_2) & \ldots & k(x_n, x_n) \end{pmatrix}. \tag{4.35}$$

Then, by the above lemma, the conditional distribution of the function at the unmeasured locations given the values at the measured locations is the Gaussian

$$f(x_*)|\boldsymbol{f} = \mathcal{N}\left(\boldsymbol{k}_*^{\mathrm{T}} \cdot \boldsymbol{K} \cdot \boldsymbol{f},\ k_{**} - \boldsymbol{k}_*^{\mathrm{T}} \cdot \boldsymbol{K}^{-1} \cdot \boldsymbol{k}_*\right). \tag{4.36}$$

Therefore, the mean and the variance of the prediction are

$$\mathrm{E}[f(x_*)] = \boldsymbol{k}_*^{\mathrm{T}} \cdot \boldsymbol{K}^{-1} \cdot \boldsymbol{f}, \tag{4.37}$$

$$\mathrm{Var}[f(x_*)] = k_{**} - \boldsymbol{k}_*^{\mathrm{T}} \cdot \boldsymbol{K}^{-1} \cdot \boldsymbol{k}_*. \tag{4.38}$$

Equation 4.36, and its two immediate corollaries Eqs. 4.37 and 4.38, are the main results of GPR.

Let's take a second to digest what this means. Under our prior assumption, the value of the function at some unobserved location, $f(x_*)$, is a Gaussian random variable. The mean of this Gaussian in Eq. 4.37 is our best prediction for the function value at the new location. Because the new location x_* can be chosen arbitrarily, this can be regarded as

an interpolating function. The variance of this Gaussian in Eq. 4.38 provides a natural estimate of the uncertainty on our prediction of the function value at the new location. This variance is also a function of the new location and will typically be larger the further away x_* is from any of the points in $\boldsymbol{x}$.

The variable x_* was introduced above as a single point in the input space. However, Eqs. 4.37 and 4.38 remain true if we instead think of x_* as a vector of multiple points. In fact, x_* can stand for an arbitrary number of arbitrary points. Recalling the definition of a GP, we see that Eqs. 4.37 and 4.38 can be regarded as the mean and covariance functions of another GP. This new GP is the posterior distribution on the function $f(x)$. GPR provides a very nice Bayesian way of handling the problems of regression, interpolation and extrapolation.

Gaussian processes are a type of conjugate prior on the space of functions.

This Bayesian viewpoint of GPR interpolation is illustrated in Fig. 4.6. The left hand plot shows the GP prior. Recall, the GP prior is a distribution on functions $f(x)$. If we are then given some data, in this case six points from the function $\sin x$, we can perform GPR. GPR can be though of a Bayesian inference problem which results in a GP posterior on functions $f(x)$. This GP posterior has mean and covariance functions given by Eqs. 4.37 and 4.38 respectively. The right hand plot in Fig. 4.6 shows the GP posterior on functions $f(x)$.

It should be stressed that both the mean and the variance of our prediction (Eqs. 4.37 and 4.38) depend on our choice of covariance function. This should be expected because the covariance function is effectively specifying our prior.

There was no particular reason for us restrict ourselves to using only zero-mean GPs as priors in the above analysis. If instead we had used $f \sim \mathcal{GP}(\mu, k)$ as a prior then the estimator in Eq. 4.37 becomes

$$\mathrm{E}[f(x_*)] = \mu(x_*) + \boldsymbol{k}_*^{\mathrm{T}} \cdot \boldsymbol{K}^{-1} \cdot (\boldsymbol{f} - \boldsymbol{\mu}), \quad \text{where} \quad \boldsymbol{\mu}^{\mathrm{T}} = \big(\mu(x_1), \mu(x_2), \ldots, \mu(x_n)\big). \tag{4.39}$$

However, this is trivially equivalent to first subtracting off our chosen mean function, $y(x) = f(x) - \mu(x)$, performing GPR on the function $y(x)$ using the zero-mean GP $y \sim \mathcal{GP}(0, k)$ as a prior, and finally adding the mean function back onto our result at the end. For this reason, most of the literature and software on GPR focusses on zero-mean GPs.

If the measurements of the function values $f(x_i)$ have associated Gaussian errors σ_i (for $i = 1, 2, \ldots, n$) then the covariance matrix in Eqs. 4.37 and 4.38 should be altered by

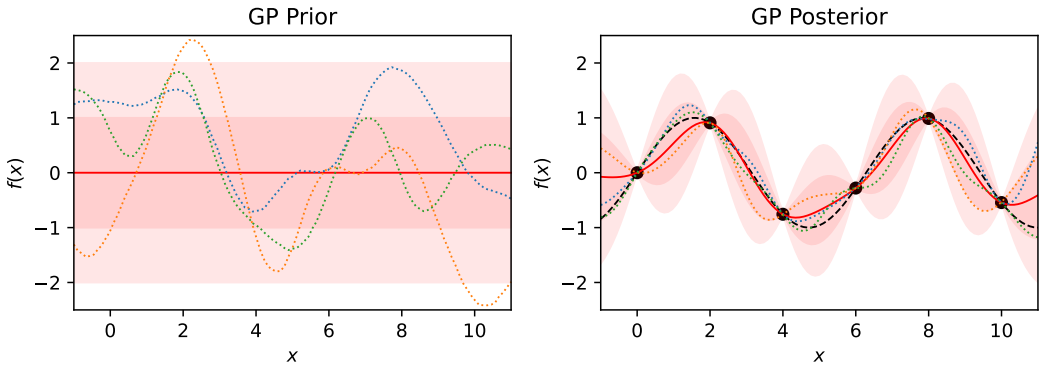


Figure 4.6: *Left:* The GP prior on the function $f(x)$. The GP prior used a squared exponential kernel function with amplitude and lengthscale parameters of $\sigma_f^2 = 1$ and $\ell = 1$ respectively. The solid red line shows the mean of the GP prior and the shaded red regions shows the one and two standard deviation uncertainty regions. The coloured dotted lines show three functions drawn as random realisations from the GP prior. *Right:* The GP posterior on the function $f(x)$. The observed data consists of six points plotted as black dots which come from the function $\sin x$ which is also plotted as a dashed black line. The solid red line shows the mean of the GP prior and the shaded red regions shows the one and two standard deviation uncertainty regions. The coloured dotted lines show three functions drawn as random realisations from the GP posterior.

adding the vector $\boldsymbol{\sigma}^2 = (\sigma_1^2, \sigma_2^2, \ldots, \sigma_n^2)^{\mathrm{T}}$ to its diagonal;

$$\mathrm{E}[f(x_*)] = \boldsymbol{k}_*^{\mathrm{T}} \cdot \left(\boldsymbol{K} + \mathrm{diag}(\boldsymbol{\sigma}^2)\right)^{-1} \cdot \boldsymbol{f}, \tag{4.40}$$

$$\mathrm{Var}[f(x_*)] = k_{**} - \boldsymbol{k}_*^{\mathrm{T}} \cdot \left(\boldsymbol{K} + \mathrm{diag}(\boldsymbol{\sigma}^2)\right)^{-1} \cdot \boldsymbol{k}_*. \tag{4.41}$$

This is like the *jitter* term introduced above and is sometimes useful purely for reasons of numerical stability.

The mean and variance of the posterior GP in Eqs. 4.37 and 4.38 are two of the outputs of GPR. The third and final output is the *evidence* for the GP, $Z = P(\boldsymbol{f})$. The evidence is just the probability of the data $\boldsymbol{f}$ and this can be obtained most simply from $\boldsymbol{f} \sim \mathcal{N}(0, \boldsymbol{K})$,

$$\log Z \equiv \log P(\boldsymbol{f}) = \frac{-1}{2} \boldsymbol{f}^{\mathrm{T}} \cdot \boldsymbol{K}^{-1} \cdot \boldsymbol{f} - \frac{1}{2} \log |\boldsymbol{K}| - \frac{n}{2} \log(2\pi). \tag{4.42}$$

This GP evidence is a function of the kernel parameters, $\log Z(\sigma_f^2, \ell, \ldots)$. As has already been mentioned, the interpolation obtained from GPR depends on the choice of kernel

function, and on the choice of the values of any parameters in the kernel function (e.g. σ_f^2, ℓ, and so on). Parameters of the kernel function are sometimes called *hyperparameters*. The evidence acts as a hyperlikelihood on the kernel hyperparameters and can be used to select the best hyperparameter values and/or perform model selection between competing kernel functions.

Notice, that the equations for the GP (Eqs. 4.37 and 4.38) involve the $(n \times n)$ matrix $\boldsymbol{K}$ (requiring an $\mathcal{O}(n^2)$ space in memory to be stored) and its inverse $\boldsymbol{K}^{-1}$ (requiring a computation with cost $\mathcal{O}(n^3)$). This is the major drawback of GPR and is largely responsible for its limited application to large data sets. When numerically implementing the equations for GPR it is not advisable to directly invert the covariance matrix; Rasmussen & Williams [2] advocate the algorithm in Alg. 4.1 which uses the Cholesky decomposition of the covariance matrix,

$$\boldsymbol{K} = \boldsymbol{L} \cdot \boldsymbol{L}^{\mathrm{T}}, \tag{4.43}$$

where $\boldsymbol{L}$ is a lower-triangular matrix. The Cholesky decomposition can be thought of as a matrix square root. Using the Cholesky decomposition we can write quantities such as

$$\begin{aligned} \boldsymbol{K}^{-1} \cdot \boldsymbol{f} &= (\boldsymbol{L}^{\mathrm{T}})^{-1} \cdot \boldsymbol{L}^{-1} \cdot \boldsymbol{f}, & (4.44) \\ &= \mathrm{Solve}(\boldsymbol{L}^{\mathrm{T}}, \mathrm{Solve}(\boldsymbol{L}, \boldsymbol{f})), & (4.45) \end{aligned}$$

and evaluate quantities like $\boldsymbol{L}^{-1} \cdot \boldsymbol{f}$ using a linear algebra routine for solving triangular systems of equations.

[2] Rasmussen & Williams (2006), "Gaussian Processes for Machine Learning", www.GaussianProcess.org/gpml.

Algorithm 4.1 A stable implementation of GPR. **Inputs:** the n-vectors $\boldsymbol{x}$, $\boldsymbol{f}, \boldsymbol{\sigma}^2$ which are the measured locations, function values and squared errors respectively, the kernel function k, and the location(s) x_* where the function is to be predicted. **Outputs:** the expectation $f_* = \mathrm{E}[f(x_*)]$, variance $V = \mathrm{Var}[f(x_*)]$, and GP log-evidence $\log Z = \log P(\boldsymbol{f})$.

procedure GPR($\boldsymbol{x}$, $\boldsymbol{f}$, $\boldsymbol{\sigma}^2$, k, x_*)
  $\boldsymbol{K} \leftarrow k(\boldsymbol{x}, \boldsymbol{x}) + \mathrm{diag}(\boldsymbol{\sigma}^2)$ ▷ covariance matrix, shape $(n \times n)$
  $\boldsymbol{k}_* \leftarrow k(x_*, \boldsymbol{x})$ ▷ vector, shape $(n,)$
  $k_{**} \leftarrow k(x_*, x_*)$ ▷ scalar
  $\boldsymbol{L} \leftarrow \mathrm{Cholesky}(\boldsymbol{K})$ ▷ Cholesky decomposition; $\boldsymbol{L}$ is lower triangular
  $\boldsymbol{\alpha} \leftarrow \mathrm{Solve}(\boldsymbol{L}^{\mathrm{T}}, \mathrm{Solve}(\boldsymbol{L}, \boldsymbol{f}))$ ▷ Triangular solver can be used
  $f_* \leftarrow \boldsymbol{k}_*^{\mathrm{T}} \cdot \boldsymbol{\alpha}$ ▷ The function expectation
  $\boldsymbol{v} \leftarrow \mathrm{Solve}(\boldsymbol{L}, \boldsymbol{k}_*)$
  $V \leftarrow k_{**} - \boldsymbol{v}^{\mathrm{T}} \cdot \boldsymbol{v}$ ▷ The function variance
  $\log Z \leftarrow \frac{-1}{2} \boldsymbol{f}^{\mathrm{T}} \cdot \boldsymbol{\alpha} - \sum_{i=1}^{n} \log \boldsymbol{L}_{ii} - \frac{n}{2} \log(2\pi)$ ▷ The GP evidence
  return f_*, V, $\log Z$
end procedure

Example 4.1: Gaussian Pocess Regression Example

An example