Bayesian Uncertainty Quantification

High Performance Computing
for Computational Science and Engineering
II

Prof. Dr. Petros Koumoutsakos

Spring 2018

Contents

1 Introduction

In science, we attempt to describe, understand and predict systems via models which depend on parameters. These models are an approximation of the reality and contain several sources of uncertainty, including modeling and numerical errors. Furthermore, we often don't know the parameters of the model or how sensible in the output of the model with respect to the parameters.. We wish to describe the uncertainty of these parameters given observations of the real system. We will here present the steps to complete this process.

In Section 2, we will present the Bayes' theorem and its applications in the field of uncertainty quantification. In Section 3, we will describe how to derive analytically estimations to quantify the uncertainty in parameters. In Section 4, we will introduce the concept of Monte Carlo methods, which are the basis of most numerical methods used in uncertainty quantification. Finally, in Section 5, we will present numerical methods able to sample arbitrary distributions.

2 Bayesian Framework

2.1 Bayes' theorem

Let X and Y be two random variables (r.v.) with densities p_X and p_Y. The Bayes' theorem states that,

$$p_{X|Y}(x|y) = \frac{p_{Y|X}(y|x)\, p_X(x)}{p_Y(y)}. \tag{1}$$

The density $p_{X|Y}$ is called the *posterior* probability. The term $p_{Y|X}$ is viewed as a function of x since on the left hand side of Eq. (1) we condition on the fixed value for the random variable $Y = y$. As a function of x this term is called the *likelihood* functions and is a measure of how likely is to observe the value y for the r.v. Y conditioning on the value x for the r.v. X. Notice that $p_{Y|X}$ is not a probability density as a function of x. The term p_X is called *prior* distribution and represents our belief on the values of X prior to observing any values for the random variable Y. Finally, the denominator is defined as,

$$p_Y(y) = \int p_{Y|X}(y|x)\, p_X(x)\, \mathrm{d}x, \tag{2}$$

and is the normalizing constant that makes the right hand side of Eq. (1) a probability density function.

In order to simplify the notation we drop the dependence of the density on the random variable. Which density is used will be evident from the arguments. For example, when we write $p(x|y)$ then $p = p_{X|Y}$ or $p(X)$ the $p = p_X$.

Bayes' theorem in action The way we will use Bayes theorem in the next sections is the following. First we make some assumptions:

- We assume that we have a computational model that depends on some parameters. These parameters are considered to be random variables and will be denoted by X. A prior distribution can be imposed on them, e.g. if we know that X takes only positive values p_X can be the gamma distribution.
- We have observed a set of data, y. We assume that data are also r.v. that follow a probability distribution.
- The likelihood function of the data, $p_{Y|X}$, is either known explicitly or can modeled based on other assumptions.

Based on these assumptions and using Eq. (1) we are able to find the distribution of the parameters conditioned on the data. Stated differently, we can answer the question "what values for the parameters will make the computational model fit the data better?".

In order to fix the notation, we will denote the random variable that represents the parameters and the data with X and D respectively and a realization from these variables with $\boldsymbol{x}$ and $\boldsymbol{d}$.

Robust prediction The uncertainty in the parameters can be propagated to the output of the model in order to quantify the uncertainty it the predictions. If the prior uncertainty is used, then the prediction is called *prior robust prediction*

$$p(y) = \int p(y \,|\, \boldsymbol{x}) \, p(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}. \tag{3}$$

If the posterior distribution is used the prediction is called *posterior robust prediction*

$$p(y \,|\, \boldsymbol{d}) = \int p(y \,|\, \boldsymbol{x}) \, p(\boldsymbol{x} \,|\, \boldsymbol{d}) \, \mathrm{d}\boldsymbol{x}. \tag{4}$$

Model selection TBW

2.2 Example: The coin flipping problem

> *A coin comes up heads 4 times in 16 flips.*
> *Is this a fair coin?*

Define H the bias-weighting of the coin. For example

- if $H = 0$: a tail comes at every flip,
- if $H = 1$: a head comes at every flip,
- if $H = \frac{1}{2}$: a fair coin.

Here, H plays the role of the model parameter x. Suppose we observe "R heads in N tosses". We want to estimate the posterior distribution of H given the observed data $\boldsymbol{d} = (R, N)$,

$$p(H \,|\, \boldsymbol{d}). \tag{5}$$

Using Bayes' theorem, we write

$$p(H \,|\, \boldsymbol{d}) \propto p(\boldsymbol{d} | H) p(H). \tag{6}$$

Here, we omit the normalization factor for simplicity. We choose a uniform prior,

$$p(H) = \begin{cases} 1, & \text{if } 0 \leq H \leq 1, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Such prior is used when we do not have any prior knowledge about the fairness of the coin: it is equally probable to have a fair coin or to have a coin completely biased towards head. We need to define the likelihood function in Eq. (6). In words, the likelihood function measures the chance to observe certain data if the value of the bias-weighting is given. Assuming independent events, it is easy to observe that the likelihood of obtaining "R heads in N tosses" follows a binomial distribution,

$$p(\boldsymbol{d} | H) \propto H^R (1 - H)^{N-R}. \tag{8}$$

This is intuitively derived by considering

- H^R is the probability of having R "heads",
- $(1 - H)^{N-R}$ is the probability of having $N - R$ "tails".

Note that we again omitted the constant factor in Eq. (8) as it does not depend on H. Posterior distributions of the bias-weighting of the coin H are shown on Fig. 1, starting from three different priors. Comparing Fig. 1 with our original problem, we can see that the probability of the coin to be fair still lies in the confidence region. However it is more likely that the coin is not fair, given the data. We need more data to increase our confidence about H.

2.3 Example: Linear model

Consider the following linear model,

$$y = x + \epsilon, \tag{9}$$

where x and ϵ are independent. We assume the following prior knowledge,

$$x \sim \mathcal{N}\left(\mu, \sigma^2\right), \tag{10}$$

$$\epsilon \sim \mathcal{N}\left(0, 1\right), \tag{11}$$

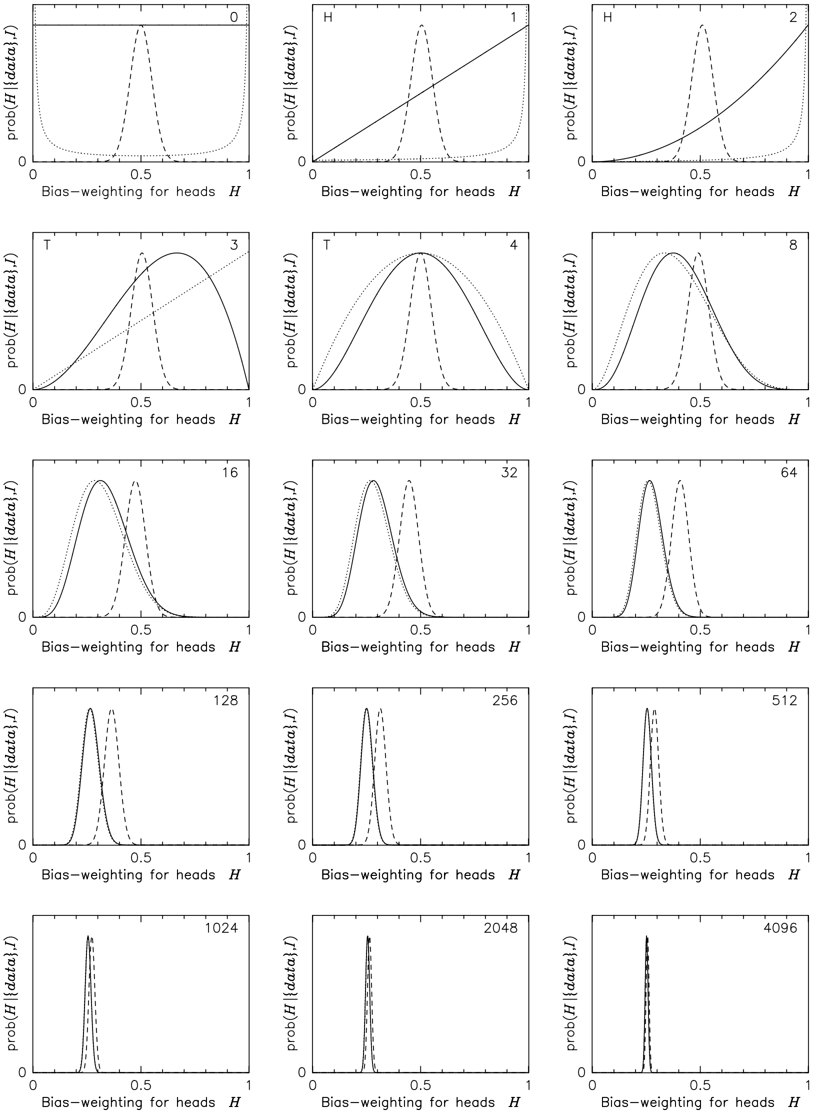

Figure 1: Evolution of the posterior density of the bias-weighting of a coin as the number of data increases. The different lines show posterior densities for different priors. The first figure for 0 data points represents the three priors. It can be seen that after many observations, the three posterior densities converge to the same distribution. However, the effect of the prior is evident for smaller observation data sets, as the biased Gaussian prior converges later to the actual posterior density. Taken from [1].

where μ and σ are given. After a single observation $y = d \in \mathbb{R}$, we can write from Bayes' theorem

$$p(x\,|\,d) = \frac{p(d\,|\,x)p(x)}{p(d)}. \tag{12}$$

From Eq. (9) and Eq. (11), the likelihood $p(d|x)$ is a Gaussian centered at x with variance 1,

$$p(d\,|\,x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(d - x\right)^2\right).$$

Substituting the prior and the likelihood inside Eq. (12) gives the posterior distribution,

$$p(x\,|\,d) \propto \exp\left(-\frac{1}{2}\left(\frac{(d-x)^2}{1} + \frac{(x-\mu)^2}{\sigma^2}\right)\right),$$

which can be written as a normal distribution,

$$p(x\,|\,d) = \mathcal{N}\left(\frac{\mu + d\sigma^2}{1+\sigma^2}, \frac{\sigma^2}{1+\sigma^2}\right).$$

Robust prediction The robust prediction is the probability density of the output of the model. This density takes the uncertainty of the parameters of the model. From the model Eq. (9), we observe that the output is a sum of two random variables. In the case of prior robust prediction, the parameter x is normally distributed (see Eq. (10)). The error ϵ is also normally distributed (see Eq. (11)). Note that the sum of two r.v. $X_1 \sim \mathcal{N}\left(\mu_1, \sigma_1^2\right)$ and $X_2 \sim \mathcal{N}\left(\mu_2, \sigma_2^2\right)$ is given by $Z = X_1 + X_2 \sim \mathcal{N}\left(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2\right)$.

We can then easily write the prior robust prediction as

$$p(y) = \mathcal{N}\left(\mu, \sigma^2 + 1\right).$$

Similarly, the posterior robust prediction can be written as

$$p(y\,|\,d) = \mathcal{N}\left(\frac{\mu + d\sigma^2}{1+\sigma^2}, \frac{\sigma^2}{1+\sigma^2} + 1\right).$$

Note that in this case, the posterior robust prediction gives a smaller confidence interval, so adding data increases the robustness of the prediction. These quantities are represented in Fig. 2.

The same result can be obtained from Eq. (3) or Eq. (4). In this particular case, $p(y\,|\,x) = \mathcal{N}\,(x, 1)$.

3 The Laplace Approximation

For non linear models, the posterior distribution can not be derived analytically in general. Therefore, we summarize such distributions with two quantities:

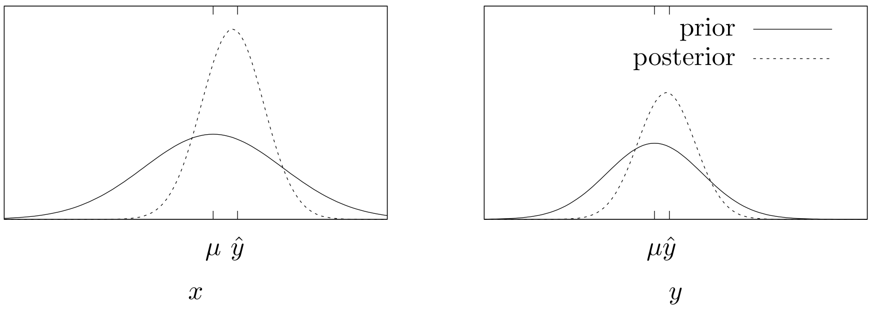

Figure 2: Left: Prior and Posterior distributions, $p(x)$ and $p(x\,|\,d)$, of the parameter x. Adding data increases our confidence in x. Right: Prior and Posterior Robust predictions of y. Again, adding data in this particular case increases the confidence in the prediction.

- the best estimate, which is the model parameter for which the posterior density function is maximized,
- measure of reliability of the best estimate.

The posterior distribution around the best estimate can be locally approximated with a Gaussian distribution, by employing the Laplace approximation method. The Laplace approximation method uses the Taylor expansion of a function around a global maximum in order to construct the exponential form of a function. Since the posterior distribution is approximated by a Gaussian, the logarithm of the posterior plays the role of the function onto which the Taylor expansion is applied. The main idea of the Laplace approximation method is discussed below.

Let $x \in \mathbb{R}$ be a parameter with probability distribution function $p(x)$. In the case of continuous variables, the following two conditions hold true for the global maximum of the distribution, $\hat{x}$,

$$\left.\frac{\partial p}{\partial x}\right|_{\hat{x}} = 0, \tag{13}$$

$$\left.\frac{\partial^2 p}{\partial x^2}\right|_{\hat{x}} < 0. \tag{14}$$

The logarithm of the probability density (which, in the Bayesian framework corresponds to the log-likelihood function) is,

$$\mathcal{L}(x) = \log\big(p(x)\big)\,.$$

By performing Taylor expansion of the logarithm of $p(x)$ around the maximum

$\hat{x}$, which corresponds to the maximum of $p(x)$, we have

$$\mathcal{L}(x) = \mathcal{L}(\hat{x}) + \frac{1}{2} \left.\frac{\partial^2 \mathcal{L}}{\partial x^2}\right|_{\hat{x}} (x-\hat{x})^2 + \mathcal{O}\left((x-\hat{x})^3\right), \tag{15}$$

where we used Eq. (13). Keeping only terms up to second order, we can write the probability distribution as,

$$p(x) \approx A \exp\left(\frac{1}{2} \left.\frac{\partial^2 \mathcal{L}}{\partial x^2}\right|_{\hat{x}} (x-\hat{x})^2 \right),$$

where A is a constant $A = \exp\left(\mathcal{L}(\hat{x})\right)$. We obtained a Gaussian approximation of the probability density function with variance

$$\sigma^2 = -\left(\left.\frac{\partial^2 \mathcal{L}}{\partial x^2}\right|_{\hat{x}} \right)^{-1}.$$

This is positive as the second derivative is negative according to the condition Eq. (14). We can finally write the Gaussian approximation, omitting the normalization constant, as

$$\begin{aligned} p(x) &\approx \sqrt{2\pi\sigma^2} p(\hat{x}) \mathcal{N}(x \,|\, \hat{x}, \sigma) \\ &\propto \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2} (x-\hat{x})^2 \right). \end{aligned}$$

The concept of Laplace approximation is graphically explained in Fig. 3.

2-dimensional approximation In 2D, if the parameters are denoted as $\boldsymbol{x} = (x_1, x_2)$, the first partial derivatives are zero, due to the existence of maximum at the best estimate $\hat{\boldsymbol{x}} = (\hat{x}_1, \hat{x}_2)$,

$$\nabla \mathcal{L}(\boldsymbol{x}) = 0$$

The log-likelihood around the best estimates $(\hat{x}_1, \hat{x}_2)$ is then approximated by Taylor series expansion,

$$\mathcal{L}(\boldsymbol{x}) \approx \mathcal{L}(\hat{\boldsymbol{x}}) + \frac{1}{2} \left(\left.\frac{\partial^2 \mathcal{L}}{\partial x_1^2}\right|_{\hat{\boldsymbol{x}}} (x_1-\hat{x}_1)^2 + \left.\frac{\partial^2 \mathcal{L}}{\partial x_2^2}\right|_{\hat{\boldsymbol{x}}} (x_2-\hat{x}_2)^2 + 2 \left.\frac{\partial^2 \mathcal{L}}{\partial x_1 \partial x_2}\right|_{\hat{\boldsymbol{x}}} (x_1-\hat{x}_1)(x_2-\hat{x}_2) \right).$$

Defining

$$A = \left.\frac{\partial^2 \mathcal{L}}{\partial x_1^2}\right|_{\hat{\boldsymbol{x}}}, \quad B = \left.\frac{\partial^2 \mathcal{L}}{\partial x_2^2}\right|_{\hat{\boldsymbol{x}}}, \quad C = \left.\frac{\partial^2 \mathcal{L}}{\partial x_1 \partial x_2}\right|_{\hat{\boldsymbol{x}}},$$

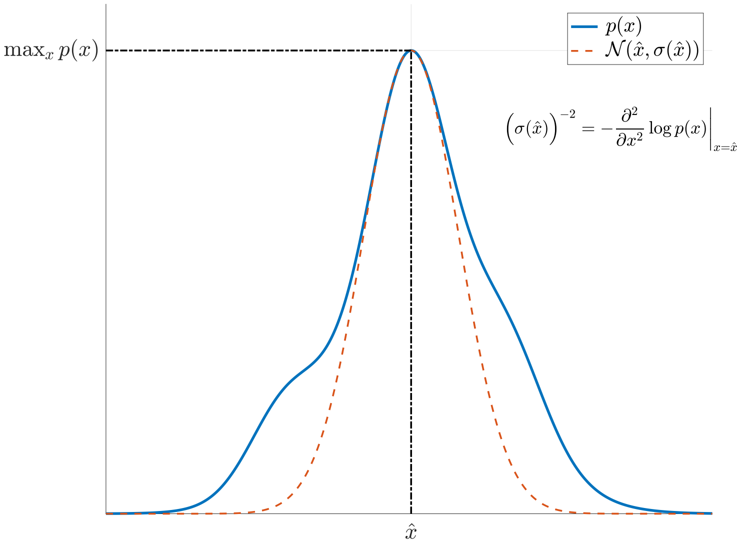

Figure 3: Laplace approximation. In the limit of many observation data the probability distribution function $p(x)$ is locally approximated as a Gaussian around the best estimate, i.e. the value that maximizes the density function.

and introducing the Hessian matrix H of the function $\mathcal{L}$,

$$H = \begin{bmatrix} A & C \\ C & B \end{bmatrix},$$

the Taylor series expansion takes the form,

$$\mathcal{L}(\boldsymbol{x}) \approx \mathcal{L}(\hat{\boldsymbol{x}}) + \frac{1}{2} Q(\boldsymbol{x}).$$

where $Q(\boldsymbol{x})$ is,

$$Q(\boldsymbol{x}) = (\boldsymbol{x} - \hat{\boldsymbol{x}})^T H(\hat{\boldsymbol{x}})(\boldsymbol{x} - \hat{\boldsymbol{x}}).$$

The covariance matrix of the Gaussian approximation is the inverse of the Hessian

$$\Sigma = H^{-1}(\hat{\boldsymbol{x}}).$$

We compute the marginal probability of parameter x_1

$$\begin{aligned} p(x_1) &= \int_{-\infty}^{\infty} p(x_1, x_2) dx_2, \\ &\approx c \exp\left(\frac{1}{2} \frac{AB - C^2}{B} (x - \hat{x}_1)^2\right), \end{aligned}$$

where c is the normalization factor.

D-dimensional approximation In higher dimensions, the Taylor expansion of $\mathcal{L}(\boldsymbol{x})$ about the best estimate $\hat{\boldsymbol{x}}$ extends as follows,

$$\mathcal{L} \approx \mathcal{L}(\hat{\boldsymbol{x}}) + \frac{1}{2}(\boldsymbol{x} - \hat{\boldsymbol{x}})^T \nabla\nabla^T \mathcal{L}(\hat{\boldsymbol{x}})(\boldsymbol{x} - \hat{\boldsymbol{x}}).$$

The Hessian at the best estimate is defined as

$$H(\hat{\boldsymbol{x}}) = \nabla\nabla^T L(\hat{\boldsymbol{x}}),$$

and the covariance matrix is again,

$$\Sigma = H^{-1}(\hat{\boldsymbol{x}}).$$

The posterior distribution is calculated as

$$\begin{aligned} p(\boldsymbol{x}) &\approx \sqrt{(2\pi)^N |\Sigma|}\; p(\hat{\boldsymbol{x}})\; \mathcal{N}(\boldsymbol{x}\,|\,\hat{\boldsymbol{x}}, \Sigma) \\ &\approx c \frac{1}{\sqrt{(2\pi)^N |\Sigma|}} \exp\left(\frac{1}{2}(\boldsymbol{x} - \hat{\boldsymbol{x}})^T \Sigma^{-1} (\boldsymbol{x} - \hat{\boldsymbol{x}})\right). \end{aligned}$$

where c is the normalization constant.

3.1 Example: Back to the coin flipping problem

In Section 2.2 we described the coin flipping problem in the Bayesian framework. We will here apply Laplace approximation to the same problem. In the Bayesian framework, we now approximate the posterior distribution of the parameter with a Gaussian distribution around the best estimate, i.e. the value that maximizes the posterior. We already showed that the posterior probability density function of the parameter $x = H$ has the form

$$p\left(H\,|\,\boldsymbol{d}\right) \propto H^R (1 - H)^{N-R}.$$

Taking the logarithm, we get

$$\mathcal{L}(H) = \text{const} + R\log(H) + (N - R)\log(1 - H),$$

where the constant does not play any role as we want to find the best estimate. The first two derivatives read,

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial H} &= \frac{R}{H} - \frac{N - R}{1 - H}, \\ \frac{\partial^2 \mathcal{L}}{\partial H^2} &= -\frac{R}{H^2} - \frac{N - R}{(1 - H)^2}. \end{aligned}$$

The condition Eq. (13) gives the best estimate $\hat{H} = \frac{R}{N}$. The standard deviation is therefore given by

$$\begin{aligned} \sigma &= \left(- \left. \frac{\partial^2 \mathcal{L}}{\partial H^2} \right|_{\hat{H}} \right)^{-\frac{1}{2}} \\ &= \sqrt{\frac{\hat{H}(1-\hat{H})}{N}}. \end{aligned}$$

Note that the certainty increases as we add more data. We can also notice that it is easier to detect a biased coin than a fair coin. Indeed, the uncertainty is maximized for $\hat{H} = \frac{1}{2}$.

3.2 Example: Gaussian mean estimator

Consider N independent and identically distributed (i.i.d.) observations $\boldsymbol{d} = (d_1, d_2, \ldots d_N)$. We assume that the data are randomly generated from a Gaussian distribution with known variance σ^2 and unknown mean $x = \mu$,

$$p(d_k \,|\, \mu) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2} \left(d_k - \mu\right)^2 \right).$$

What is the best estimate for μ and what is our confidence for this estimate? From Bayes' theorem, the posterior probability of the mean μ is given by

$$p(\mu \,|\, \boldsymbol{d}) \propto p(\boldsymbol{d} | \mu)\, p(\mu).$$

Since the data is i.i.d., the likelihood function takes the form

$$p(\boldsymbol{d} | \mu) = \prod_{k=1}^{N} p(d_k \,|\, \mu).$$

Here, we assume an uninformative uniform prior for the mean of the Gaussian,

$$p(\mu) = \begin{cases} c = \frac{1}{\mu_{max} - \mu_{min}}, & \mu_{\min} \leq \mu \leq \mu_{\max} \\ 0, & \text{otherwise.} \end{cases}$$

The posterior distribution is then given by

$$\begin{aligned} p(\mu \,|\, \boldsymbol{d}) &\propto c \prod_{k=1}^{N} p(d_k \,|\, \mu) \\ &= c \prod_{k=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2} \left(d_k - \mu\right)^2 \right) \\ &= c \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{k=1}^{N} (d_k - \mu)^2 \right). \end{aligned}$$

We compute the log-likelihood,

$$\begin{aligned}\mathcal{L}(\mu) &= \log(p(\mu\,|\,\boldsymbol{d})),\\ &= \text{const} - \sum_{k=1}^{N} \frac{(d_k-\mu)^2}{2\sigma^2}.\end{aligned}$$

The best estimate $\hat{\mu}$ must satisfy

$$\begin{aligned}\left.\frac{\mathrm{d}\mathcal{L}(\mu)}{\mathrm{d}\mu}\right|_{\hat{\mu}} &= \sum_{k=1}^{N} \frac{d_k-\hat{\mu}}{\sigma^2} = 0,\\ &\Rightarrow \sum_{k=1}^{N} d_k = \sum_{k=1}^{N} \hat{\mu},\\ &\Rightarrow \hat{\mu} = \frac{1}{N}\sum_{k=1}^{N} d_k.\end{aligned}$$

We compute the second derivative of the log-likelihood

$$\frac{\mathrm{d}^2\mathcal{L}}{\mathrm{d}\mu^2} = -\sum_{k=1}^{N}\frac{1}{\sigma^2} = -\frac{N}{\sigma^2},$$

which is negative, meaning that $\mathcal{L}(\hat{\mu})$ is indeed a maximum. Finally, the standard deviation of the posterior is equal to $\sigma/\sqrt{N}$.

4 Monte Carlo Methods

In the previous section we saw that the Laplacian approximation method can be used to approximate the posterior distribution of the parameters. This approach is characterized as *deterministic*. Alternatively, we can make use of *stochastic* methods in order to numerically represent the posterior probabilities with randomly generated samples from the underlying distribution.

4.1 Monte Carlo Integration

The main concept of the Monte-Carlo methods is presented here in the case of the numerical computation of an integral of the following form

$$E[f(\boldsymbol{x})] = \int f(\boldsymbol{x})\,p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}, \tag{16}$$

where $\boldsymbol{x}$ is a random vector with density p and f is a given function we want to integrate. Common examples are:

1. model evidence:
$$p(\boldsymbol{d}) = \int p(\boldsymbol{d}\,|\,\boldsymbol{x})p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x},$$

2. robust posterior prediction:

$$p(y \,|\, \boldsymbol{d}) = \int p(y \,|\, \boldsymbol{x}) p(\boldsymbol{x} \,|\, \boldsymbol{d}) \, \mathrm{d}\boldsymbol{x},$$

Assume that $\{\boldsymbol{x}^{(k)}\}_{k=1}^{N}$ are i.i.d. samples drawn from the density p. Using the Law of Large numbers, the expected value of $f(\boldsymbol{x})$ is given by the estimate

$$\hat{\mu}_{f,N} = \frac{1}{N} \sum_{k=1}^{N} f(\boldsymbol{x}^{(k)}).$$

In the limit $N \to \infty$, the sample average converges to the expected value. Defining

$$\begin{aligned} \mu_f &= \mathbb{E}_{[} \left[f \right] (\boldsymbol{x})] \\ \sigma_f^2 &= \mathrm{Var}[f(\boldsymbol{x})] = \mathbb{E}_{[} \left[f \right]^2 (\boldsymbol{x})] - \mu_f^2, \end{aligned}$$

the Law of Large numbers and the Central Limit theorem give

$$\begin{gathered} \lim_{N \to \infty} \hat{\mu}_{f,N} = \mu_f, \\ \hat{\mu}_{f,N} \sim \mathcal{N} \left(\mu_f, \sigma_f^2 / N \right). \end{gathered}$$

Conclusively:

- The error of the sample estimate decreases as $1/\sqrt{N}$.
- The sample estimate is an unbiased estimate of the true value.
- Convergence of the estimate is independent of the dimensionality of the problem.

4.2 Random Number Generators

The whole concept of Monte-Carlo methods relies on the generation of random samples. It is essential to generate such numbers with the desired properties.

Pseudo-random number generators: Algorithms that guarantee the generation of a sequence of integers Z_i that approximately follow a uniform distribution on an interval in the real axis. The general algorithm for the generation of pseudo-random numbers is

$$Z_i = g(Z_{i-1}, \ldots, Z_{i-m}) \mod M,$$

for generating integers in the interval $[0, M-1]$. In this case, Z_i is the remainder of the division of $g(Z_{i-1}, \ldots, Z_{i-m})$ by M. In a simpler form, Z_i can be generated by:

$$Z_i = \alpha Z_{i-1} \mod M,$$

for $Z_0 = 1$ and $i \geq 1$. For the above, M is a large prime number and α an integer [2].

The resulting sequence of random numbers turns out to be ergodic with period $M - 1$. A sequence is accepted when it satisfies certain criteria. For example, for a choice of $M = 10^9$ not all α result into an good quality sequence of random numbers. Recommended options are: $\alpha = 7^5$, $M = 2^{31} - 1$ [3].

Note for non-uniform distributions Integrals of the form

$$I = \int_\Omega f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$$

can be written as

$$I = |\Omega| \int_\Omega f(\boldsymbol{x})\,p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = |\Omega|\,\mathbb{E}_p[f].$$

Here p is the uniform distribution over Ω from which samples are drawn,

$$p(\boldsymbol{x}) = \begin{cases} \frac{1}{|\Omega|}, & x \in \Omega, \\ 0, & \text{otherwise.} \end{cases}$$

For these cases, the pseudo-random number generators are sufficient. In the general case we want to approximate the integral Eq. (16) for a non uniform density p. Usually, the distribution p is known up to a constant factor,

$$p(x) = \frac{\phi(x)}{Z},$$

where $Z = \int_{-\infty}^{\infty} \phi(x)\,\mathrm{d}x$. In the next sections, we will discuss how to generate random numbers from such distributions.

4.3 Importance Sampling

We want to evaluate the integral

$$I = \int f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}, \tag{17}$$

using Monte Carlo integration. Eq. (17) can be written equivalently as

$$\begin{aligned} I &= \int \frac{f(\boldsymbol{x})}{p(\boldsymbol{x})}\,p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}, \\ &= \mathbb{E}_p\left[\frac{f(\boldsymbol{x})}{p(\boldsymbol{x})}\right], \end{aligned}$$

where $p(\boldsymbol{x}) > 0$ is a probability density function with $\int p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = 1$. Therefore, we can approximate I using the Monte Carlo integration technique,

$$\hat{I} = \frac{1}{N} \sum_{k=1}^{N} \frac{f(\boldsymbol{x}^{(k)})}{p(\boldsymbol{x}^{(k)})},$$

where the samples $\{\boldsymbol{x}^{(k)}\}_{k=1}^{N}$ are i.i.d. and follow the density p. There is an infinite amount of choices for p from which we would like to select one which:

1. is easy to sample,
2. minimizes the error of the estimate for a finite number of samples.

A measure of the error of the estimate is given by

$$\mathbb{E}_p\left[\left(\frac{f(\boldsymbol{x})}{p(\boldsymbol{x})} - I\right)^2\right] = \int \frac{f(\boldsymbol{x})^2}{p(\boldsymbol{x})^2}\, p(\boldsymbol{x})\,\mathrm{d}x - I^2. \tag{18}$$

It is easy to show that this is minimized for

$$p(\boldsymbol{x}) = \frac{f(\boldsymbol{x})}{I},$$

but this expression implies that we already know I. In practice, we choose p "similar" to f.

5 Sampling methods

5.1 Function Inversion

Let X be a real random variable with probability density function p_X and corresponding cumulative distribution function

$$F_X(x) = \int_{-\infty}^{x} p_X(r)\,\mathrm{d}r.$$

The idea behind the "Inverse Transform Sampling" method is that samples from the density $p_X(x)$ can be generated by a transformation

$$x = g(u),$$

where $u \sim \mathcal{U}(0,1)$. We will identify the function g such that X follows the desired density p_X. The densities of X and U should satisfy

$$p_X(x)\,\mathrm{d}x = p_U(u)\,\mathrm{d}u, \tag{19}$$

which leads to

$$p_X(x) = p_U(u)\left|\frac{\mathrm{d}u}{\mathrm{d}x}\right| = p_U(u)\left|\frac{\mathrm{d}g(u)}{\mathrm{d}u}\right|^{-1}.$$

For the r.v. U drawn from a uniform probability distribution, $U \sim p_U(u) = \mathcal{U}(u\,|\,0,1)$, the probability of generating a random number between u and $u + \mathrm{d}u$ is

$$p_U(u)\,\mathrm{d}u = \begin{cases} \mathrm{d}u, & 0 \le u < 1, \\ 0, & \text{otherwise.} \end{cases}$$

Therefore, $p_U(u) = 1$ for $u \in [0, 1]$ and integrating Eq. (19) yields

$$\int_{-\infty}^{x} p_X(r)\,\mathrm{d}r = \int_{0}^{u} p_U(r)\,\mathrm{d}r = u.$$

This means, from the definition of F_X, that

$$F_X(x) = u.$$

Assuming that F_X has an inverse F_X^{-1}, we obtain

$$x = g(u) = F_X^{-1}(u).$$

For simple density functions p_X for which F_X^{-1} is known, it is therefore easy to generate samples X. However, the inverse is not available in general, which lead to the developpement of sampling algorithms, as discussed later in this section.

Example: Exponential Distribution Given that the random variable x is distributed according to the probability density function

$$p_X(x) = \lambda e^{-\lambda x},$$

with $\lambda > 0$ and $x \geq 0$, the CDF of x is given by

$$F_X(x) = \int_{0}^{x} \lambda e^{-\lambda \tau}\,\mathrm{d}\tau = 1 - e^{-\lambda x}.$$

Setting the random variable $u := F_X(x)$, x can be sampled from the inverse transformation

$$x = g(u) = F_X^{-1}(u),$$

or equivalently

$$F_X(x) = u \Rightarrow 1 - e^{-\lambda x} = u,$$

which results in

$$x = -\frac{1}{\lambda}\ln(1-u).$$

If samples $\{u^{(k)}\}_{k=1}^{N}$ are drawn from $\mathcal{U}(0, 1)$ then $x^{(k)} = -\frac{1}{\lambda}\ln(1 - u^{(k)})$ follow p_X.

Example: Gaussian Distribution We want to draw samples from the standard normal distribution

$$x \sim \mathcal{N}(0, 1).$$

The inverse transform method can be time consuming for this case, as F_X^{-1} is not known in closed form. An alternative algorithm, called *Box-Muller transformation*, uses the inverse transform method to convert two independent uniform random variables into two independent gaussian random variables. Suppose that $\{r, \phi\}$ is a set of two independent distributed random variables according to the following:

- ϕ is drawn by a uniform distribution with a simple transformation so that

$$p_\Phi(\phi) = \begin{cases} \frac{1}{2\pi}, & 0 \leq \phi \leq 2\pi, \\ 0, & \text{otherwise.} \end{cases}$$

- r is sampled according to the exponential distribution

$$p_R(r) = \frac{1}{2} e^{-\frac{r}{2}},$$

for $r > 0$. The sampling from p_R is performed via an inverse transformation from a uniformly sampled variable, as seen in the previous example.

Since r and ϕ are independent, the joint probability is

$$p_{R,\Phi}(r, \phi) = p_R(r) p_\Phi(\phi).$$

We now define the transformation, $x = \sqrt{r} \cos\phi$ and $y = \sqrt{r} \sin\phi$, so that $r = x^2 + y^2$ and $\phi = \arctan y/x$. The sampling joint distribution for x and y is now computed as

$$\begin{aligned} p_{X,Y}(x, y) &= \frac{1}{2\pi} \frac{1}{2} e^{-\frac{x^2+y^2}{2}}, \\ &= \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}}. \end{aligned}$$

The joint distribution ends in describing two independent normally distributed variables.

5.2 Rejection Sampling

Until now, we have seen how to generate samples from a uniform distribution, through pseudo-random numbers generators and from other distributions, using the inverse transformation method. However, there are many distributions, from which it may be impossible to directly define an inverse transform. In such cases, we turn to methods that only require knowledge of the functional form of the probability density function p up to a constant. The key concept here is the following: In order to generate independent samples from a desired density p one draws from another density q that is easier to sample from and then, instead of applying a transformation to q, some sampled points are rejected according to certain criteria.

Given a density p, we can write

$$p(x) = \int_0^{p(x)} 1 \, \mathrm{d}x = \int_{-\infty}^{\infty} \chi_{[0,p(x)]}(u) \, \mathrm{d}u. \tag{20}$$

The function

$$\chi_{[0,p(x)]}(u) = \begin{cases} 1, & 0 \leq u \leq p(x), \\ 0, & \text{otherwise.} \end{cases}$$

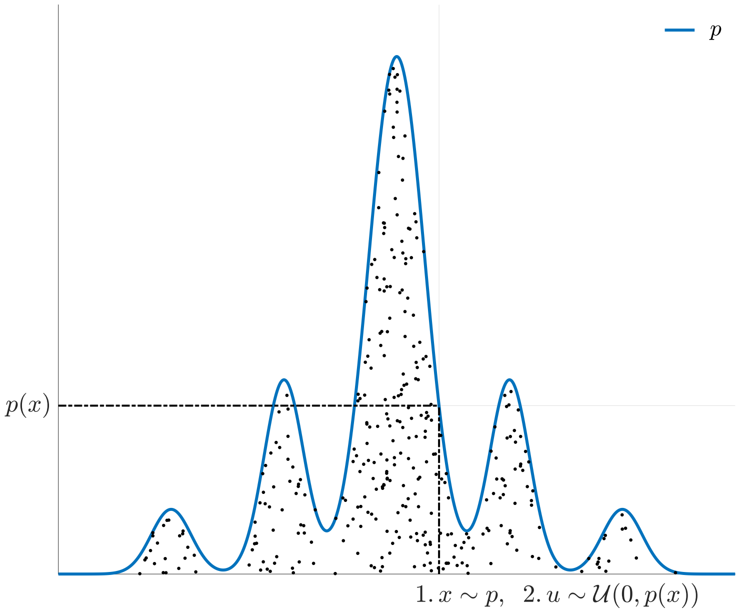

Figure 4: Assuming we can sample from the distribution p, we draw a sample $x \sim p$ and then a uniform number $u \sim \mathcal{U}(0, p(x))$. If we marginalize the samples (x, u) we recover the distribution p, as shown in Eq. (20).

can be seen as the joint distribution $p_{X,U}$ of the random variables X and U following the distribution $p(x)$ and $p(u|x) = \mathcal{U}(u\,|\,0, p(x)$. Marginalizing the joint distribution over U we recover the distribution p,

$$p(x) = \int p_{X,U}(x, u)\, \mathrm{d}u. \tag{21}$$

This property of p is presented in Fig. 4. The dots in the figure correspond to samples drawn from the joint $p_{X,U}$ and they are uniformly distributed under the graph of p.

Acceptance–Rejection technique What if we cannot directly sample from the density p? The answer is simple:

1. find a density q that samples are easily drawn from,

2. scale q by a constant M such that the graph of Mq is always above the graph of p,

3. sample from the joint density $p_{X,U}$ where $x \sim q$ and $u \sim \mathcal{U}(0, Mq(X))$,

4. keep only the points that are bellow the graph of p.

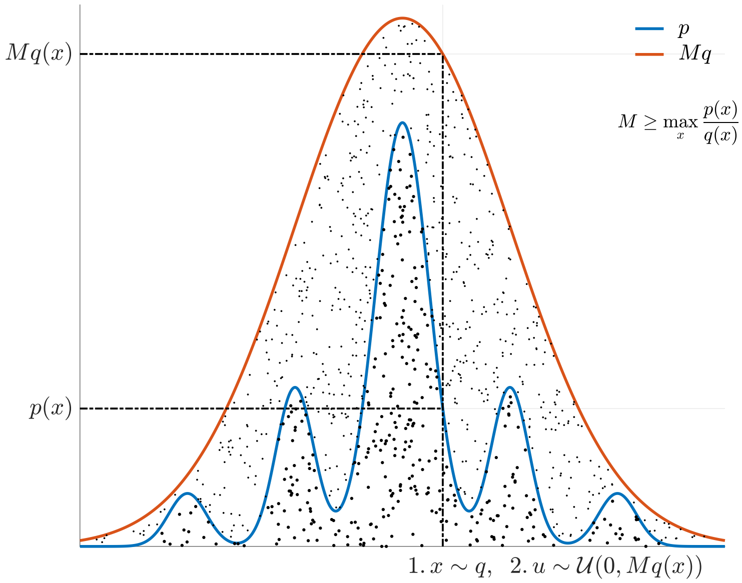

Figure 5: Demonstration of the Accept-Reject algorithm. 1. A sample x is drawn from the distribution q. 2. A random number u is drawn uniformly in $[0, Mq(x)]$. 3. The sample x is accepted if $u < p(x)$, i.e., if the point (x, u) is bellow the graph of p, and rejected otherwise.

This intuitive procedure is called *acceptance–rejection* algorithm and is presented graphically in Fig. 5. The detailed algorithm is presented in Algorithm 1. A basic requirement of the algorithm is that the graph of p should always be bellow the graph of Mq. Equivalently, the constant M must satisfy,

$$M > \max_x \frac{p(x)}{q(x)}.$$

Theorem 1. *The samples generated from Algorithm 1 are distributed according to p.*

Proof. According to the algorithm, we first sample $x \sim q$, then $u \sim \mathcal{U}(0, Mq(x))$ and we accept if $u > p(x)$. Thus, the posterior density, using Bayes' theorem, is given by

$$p(x \,|\, u \leq p(x)) = \frac{p(u \leq p(x) \,|\, x)\, q(x)}{p(\xi \leq p(x))}. \tag{22}$$

The likelihood function corresponds to the probability of a uniformly distributed value in $[0, Mq(x)]$ to be less or equal to $p(x)$. It easy to check that it is equal to

$$p(u \leq p(x) \,|\, x) = \frac{p(x)}{Mq(x)}. \tag{23}$$

Algorithm 1 Rejection sampling algorithm.

```
  Input: densities $p$, $q$ and constant $M > 0$ such that $p(x)/q(x)$
  Output: a sample distributed according to $p$
function REJECTION_SAMPLING($p$, $q$, $M$)
    Generate $x \sim q$                                   ▷ Propose a new sample
    Generate $u \sim \mathcal{U}(0, Mq(x))$
    if $u < p(x)$ then
        return $x$                                        ▷ Accept the proposed sample
    else
        return REJECTION_SAMPLING($p$, $q$, $C$)          ▷ Reject and try again
    end if
end function
```

In order to evaluate the denominator of Eq. (22), we integrate the numerator of Eq. (22) and use Eq. (23),

$$\begin{aligned} p(u \leq p(x)) &= \int p(u \leq p(x) \,|\, x) \, q(x) \, \mathrm{d}x, \\ &= \int \frac{p(x)}{Mq(x)} q(x) \, \mathrm{d}x, \\ &= \int \frac{1}{M} \, p(x) \, \mathrm{d}x, \\ &= \frac{1}{M}. \end{aligned} \tag{24}$$

Inserting Eq. (23) and Eq. (24) in Eq. (22) we obtain

$$p(x \,|\, u \leq p(x)) = \frac{\frac{p(x)}{Mq(x)} \, q(x)}{\frac{1}{M}} = p(x).$$

□

Note: The efficiency of the algorithm depends on whether $u \leq p(x)$. For independent trials, the probability of success is $\frac{1}{M}$ (see Eq. (24)). Thus, that the expected number of trials before accepting the sample is C.

Example: von Neumann The original rejection algorithm (Algorithm 2) was used by von Neumann to draw samples from a density $p(x)$ in $[a, b]$, using a uniform proposal

$$q(x) = \frac{1}{b - a}, \qquad \text{for } a \leq x \leq b.$$

The constant M is given by

$$M \geq \max_{x \in [a,b]} (b - a) p(x).$$

Since M should be as small as possible, we select the lower bound in the above inequality,

$$M = (b-a) \max_{x \in [a,b]} p(x).$$

The procedure to generate one sample is summarized in Algorithm 2.

Algorithm 2 Von Neuman Rejection sampling algorithm.

```
    Input: density $p$, interval $(a, b)$ and constant $M = \max_{x \in [a,b]} p(x)$
    Output: a sample following the density $p$
  function REJECTION_SAMPLING($p$, $a$, $b$, $M$)
      Generate $x \sim \mathcal{U}(a, b)$
      Generate $u \sim \mathcal{U}(0, M/(b-a))$
      if $u < p(x)$ then
          return $x$                                   ▷ Accept the proposed sample
      else
          return REJECTION_SAMPLING($p$, $a$, $b$, $M$)      ▷ Reject and try again
      end if
  end function
```

5.3 Markov Chain Monte Carlo

A Markov chain is a sequence of random numbers $x_1, x_2, \ldots \in \mathbb{R}^d$ with conditional distributions that obey the rule

$$P\left(x_n \,|\, x_{n-1}, x_{n-2}, \ldots, x_1\right) = P\left(x_n \,|\, x_{n-1}\right). \tag{25}$$

The Metropolis-Hasting algorithm [4], introduced by Nicholas Metropolis together with Arianna W. Rosenbluth, Marshall Rosenbluth, Augusta H. Teller, and Edward Teller (M(RT)^2), makes use of the Markov chain properties to generate samples from a probability density function. For a stochastic process $W(x\,|\,y)$ following a Markov chain, the probability density of the states converges to an equilibrium probability density function p_{eq} if the detailed balance equation is satisfied,

$$W(x\,|\,y) p_{eq}(y) = W(y\,|\,x) p_{eq}(x). \tag{26}$$

In statistical physics, we usually know the stochastic process and need to find the equilibrium distribution. Here we want the opposite: we know the density p_{eq} and want to design a suitable process W which generates states distributed according to p_{eq}. The idea of M(RT)^2 is to write this process as a combination of proposition and acceptance terms T and A,

$$W(x\,|\,y) = A(x\,|\,y) T(x\,|\,y). \tag{27}$$

The proposal distribution $T(x'\,|\,y)$ proposes the transition from y to x'. It must normalize to 1:

$$\int T(x\,|\,y) dx = 1.$$

The proposed state x' is then accepted with acceptance probability $A(x' \,|\, y)$. The acceptance must be chosen to satisfy the detailed balance condition Eq. (26),

$$A(x \,|\, y)T(x \,|\, y)p_{eq}(y) = A(y \,|\, x)T(y \,|\, x)p_{eq}(x).$$

We now define

$$q(x \,|\, y) = \frac{T(y \,|\, x)p_{eq}(x)}{T(x \,|\, y)p_{eq}(y)}. \tag{28}$$

Note that $q(x \,|\, y) \geq 0$. It is easy to check that the detailed balance condition is satisfied for

$$A(x \,|\, y) = \min\left[1, q(x \,|\, y)\right].$$

The algorithm used to generate one sample from a given state is summarized in Algorithm 3. Note that it is sufficient to know p_{eq} only up to a constant factor. Indeed, in the M(RT)^2 algorithm, p_{eq} appears only in a ratio (see Eq. (28)).

Algorithm 3 One step of the Metropolis-Hasting sampling algorithm

```
    Input: Current state $y$, proposal density $T$, target density $p_{eq}$
    Output: Next state
  function Metropolis_Hasting_step($y$, $T$, $P_{eq}$)
      generate $x \sim T(.\,|\,y)$                                   ▷ Propose a new state
      set $q \leftarrow T(y\,|\,x)p_{eq}(x)/T(x\,|\,y)p_{eq}(y)$
      if $q > 1$ then
          return $x$
      else
          generate $U \sim \mathcal{U}(0,1)$
          if $U < q$ then            ▷ select the state with probability $A(x\,|\,y) = q$
              return $x$                                         ▷ Accept the new state
          else
              return $y$                                         ▷ Reject the new state
          end if
      end if
  end function
```

We will now demonstrate that the M(RT)^2 algorithm indeed converges to the desired distribution p_{eq}. We define the probability densities ϕ_i of each random variable x_i in the sequence. Given ϕ_n, we can write $\phi_{n+1}(x)$ as a sum of two contributions:

- Probability of accepting a new state,

$$P\left(\text{“previous state was not } x\text{”}\right) = \int A(x \,|\, y)T(x \,|\, y)\phi_n(y)dy,$$

- Probability of not moving away from x, i.e. rejected the proposed state,

$$P\left(\text{“previous state was } x\text{”}\right) = \phi_n(x)\int (1 - A(y \,|\, x))T(y \,|\, x)dy.$$

Combining the above contributions gives

$$\phi_{n+1}(x) = \int A(x|y)T(x|y)\phi_n(y)dy + \phi_n(x)\int (1 - A(y|x))T(y|x)dy. \quad (29)$$

It can be shown that this recursive relation gives an ergodic system (the system will return to the states already visited with probability one and every state is aperiodic). Therefore, according to Theorem 2 (see also [5]), there exist a unique equilibrium distribution to which the recursion Eq. (29) converges.

Theorem 2 (Feller)**.** *If a random variable defines an ergodic system, then it exists a unique probability distribution p_{eq} that is a fixed point in the above recursion.*

Proof. We will show now that the fixed point is indeed p_{eq}. We substitute $\phi_n = p_{eq}$ in Eq. (29) and obtain

$$\begin{aligned}\phi_{n+1}(x) &= \int A(x|y)T(x|y)p_{eq}(y)dy + p_{eq}(x)\int (1 - A(y|x))T(y|x)dy,\\ &= \int \big[A(x|y)T(x|y)p_{eq}(y) - A(y|x)T(y|x)p_{eq}(x)\big]\,dy\\ &\quad + p_{eq}(x)\int T(y|x)dy,\\ &= p_{eq}(x)\int T(y|x)dy,\\ &= p_{eq}(x),\end{aligned}$$

where we used the detailed balance condition Eq. (26). Therefore, p_{eq} is the asymptotic density distribution of the random walk.

□

References

[1] Devinderjit Sivia and John Skilling. *Data analysis: a Bayesian tutorial.* OUP Oxford, 2006.

[2] Derrick H Lehmer. Euclid's algorithm for large numbers. *American Mathematical Monthly*, pages 227–233, 1938.

[3] Donald E Knuth. *The Art of Computer Programming; Volume 2: Seminumeral Algorithms.* 1981.

[4] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

[5] Francesco Petruccione and Peter Biechele. Stochastic methods for physics using java: An introduction. 2000.