Exercises for the tutorials: 1 and 2.

The other exercises are for self-study and exam preparation. All material is examinable unless otherwise mentioned.

Exercise 1. *Importance sampling to estimate tail probabilities*

We would like to use importance sampling to compute the probability that a standard Gaussian random variable x takes on a value larger than 5, i.e

$$\mathbb{P}(x > 5) = \int_5^\infty \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) dx \tag{1}$$

We know that the probability equals

$$\begin{aligned}\mathbb{P}(x > 5) &= 1 - \int_{-\infty}^5 \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) dx &(2)\\ &= 1 - \Phi(5) &(3)\\ &\approx 2.87 \cdot 10^{-7} &(4)\end{aligned}$$

where $\Phi(.)$ is the cumulative distribution function of a standard normal random variable.[1]

(a) *With the indicator function $\mathbb{1}_{x>5}(x)$, which equals one if x is larger than 5 and zero otherwise, we can write $\mathbb{P}(x > 5)$ in form of the expectation*

$$\mathbb{P}(x > 5) = \mathbb{E}[\mathbb{1}_{x>5}(x)], \tag{5}$$

where the expectation is taken with respect to the density $\mathcal{N}(x; 0, 1)$ of a standard normal random variable,

$$\mathcal{N}(x; 0, 1) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right). \tag{6}$$

This suggests that we can approximate $\mathbb{P}(x > 5)$ by a Monte Carlo average

$$\mathbb{P}(x > 5) \approx \frac{1}{n} \sum_{i=1}^n \mathbb{1}_{x>5}(x_i), \qquad x_i \sim \mathcal{N}(x; 0, 1). \tag{7}$$

Explain why this approach does not work well.

Solution. In this approach, we essentially count how many times the x_i are larger than 5. However, we know that the chance that $x_i > 5$ is only $2.87 \cdot 10^{-7}$. That is, we only get about one value above 5 every 20 million simulations! The approach is thus very sample inefficient.

(b) *Another approach is to use importance sampling with an importance distribution $q(x)$ that is zero for $x < 5$. We can then write $\mathbb{P}(x > 5)$ as*

$$\begin{aligned}\mathbb{P}(x > 5) &= \int_5^\infty \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) dx &(8)\\ &= \int_5^\infty \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \frac{q(x)}{q(x)} dx &(9)\\ &= \mathbb{E}_{q(x)}\left[\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \frac{1}{q(x)}\right] &(10)\end{aligned}$$

[1]Credit: This exercise is based on example and exercise 3.5 in Robert and Casella's *Introducing Monte Carlo Methods with R*, Springer 2010.

and estimate $\mathbb{P}(x > 5)$ as a sample average.
We here use an exponential distribution shifted by 5 to the right. It has pdf

$$q(x) = \begin{cases} \exp(-(x-5)) & \text{if } x \geq 5 \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

For background on the exponential distribution, see e.g. https://en.wikipedia.org/wiki/Exponential_distribution.
Provide a formula that approximates $\mathbb{P}(x > 5)$ as a sample average over n samples $x_i \sim q(x)$.

Solution. The provided equation

$$\mathbb{P}(x > 5) = \mathbb{E}_{q(x)}\left[\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \frac{1}{q(x)}\right] \tag{S.1}$$

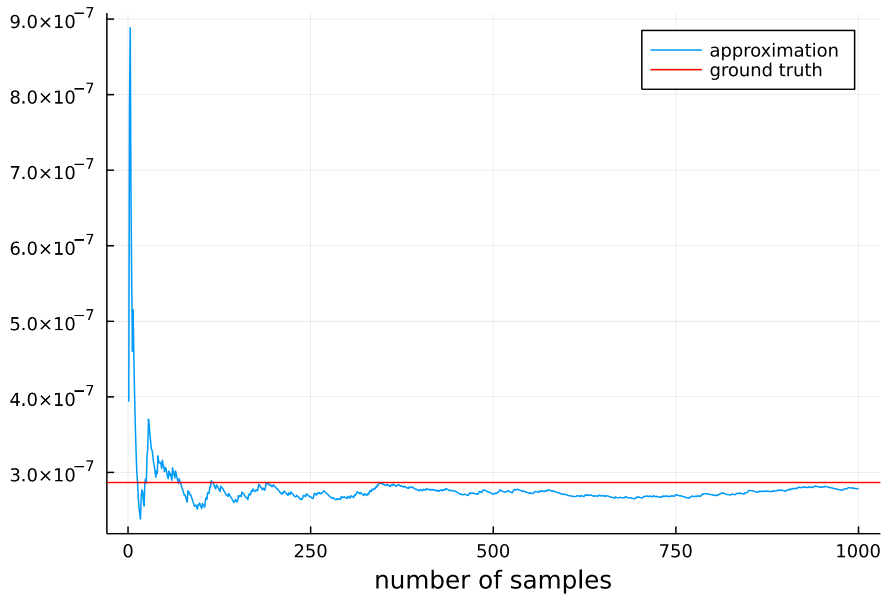
can be approximated as a sample average as follows:

$$\begin{aligned} \mathbb{P}(x > 5) &\approx \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x_i^2}{2}\right) \frac{1}{q(x_i)} & \text{(S.2)} \\ &= \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x_i^2}{2} + x - 5\right) & \text{(S.3)} \end{aligned}$$

with $x_i \sim q(x)$.

(c) *Numerically compute the importance estimate for various sample sizes $n \in [0, 1000]$. Plot the estimate against the sample size and compare with the ground truth value.*

Solution. The following figure shows the importance sampling estimate as a function of the sample size (numbers do depend on the random seed used). We can see that we can obtain a good estimate with a few hundred samples already.



Python code is as follows.

```python
import numpy as np
from numpy.random import default_rng
import matplotlib.pyplot as plt
from scipy.stats import norm

n = 1000
alpha = 5

# compute the tail probability
p = 1-norm.cdf(alpha)

# sample from the importance distribution
rng = default_rng()
vals = rng.exponential(scale=1, size=n) + alpha

# compute average
def w(x):
return 1/np.sqrt(2*np.pi)*np.exp(-x**2/2+x-alpha)

Ihat = np.cumsum(w(vals))/ np.arange(1, n+1)

# plot
plt.plot(Ihat)
plt.axhline(y=p, color="r")
plt.xlabel("number of samples")
```

And code in Julia is:

```julia
using Distributions
using Plots
using Statistics

# compute the tail probability
phi(x) = cdf(Normal(0,1),x)
alpha = 5
p = (1-phi(alpha))

# sample from the importance distribution
n = 1000
exprv = Exponential(1)
x = rand(exprv, n).+alpha;

# compute the approximation
w(x) = 1/sqrt(2*pi)*exp(-x^2/2+x-alpha)
#w(x) = pdf(Normal(0,1),x)/pdf(exprv, x-alpha);

Ihat = zeros(length(x));
for k in 1:length(x)
   Ihat[k] = mean(w.(x[1:k]));
```

```
    end

    # plot
    plt=plot(Ihat, label="approximation");
    hline!([p], color=:red, label="ground truth")
    xlabel!("number of samples")
```

Exercise 2. ***Monte Carlo integration and importance sampling***

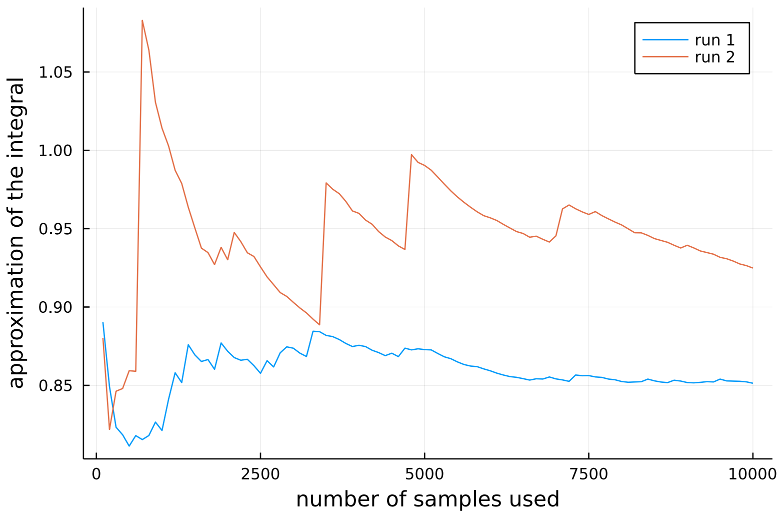
A standard Cauchy distribution has the density function (pdf)

$$p(x) = \frac{1}{\pi} \frac{1}{1 + x^2} \tag{12}$$

with $x \in \mathbb{R}$. A friend would like to verify that $\int p(x)dx = 1$ but doesn't quite know how to solve the integral analytically. They thus use importance sampling and approximate the integral as

$$\int p(x)dx \approx \frac{1}{n} \sum_{i=1}^{n} \frac{p(x_i)}{q(x_i)} \qquad x_i \sim q \tag{13}$$

where q is the density of the auxiliary/importance distribution. Your friend chooses a standard normal density for q and produces the following figure:



The figure shows two independent runs. In each run, your friend computes the approximation with different sample sizes by subsequently including more and more x_i in the approximation, so that, for example, the approximation with $n = 2000$ shares the first 1000 samples with the approximation that uses $n = 1000$.

Your friend is puzzled that the two runs give rather different results (which are not equal to one), and also that within each run, the estimate very much depends on the sample size. Explain these findings.

Solution. While the estimate $\hat{I}_n$

$$\hat{I}_n = \frac{1}{n} \sum_{i=1}^{n} \frac{p(x_i)}{q(x_i)} \tag{S.4}$$

is unbiased by construction, we have to check whether its second moment is finite. Otherwise, we have an invalid estimator that behaves erratically in practice. The ratio $w(x)$ between $p(x)$ and $q(x)$ equals

$$w(x) = \frac{p(x)}{q(x)} \tag{S.5}$$

$$= \frac{\frac{1}{\pi}\frac{1}{1+x^2}}{\frac{1}{\sqrt{2\pi}}\exp(-x^2/2)} \tag{S.6}$$

which can be simplified to

$$w(x) = \frac{\sqrt{2\pi}\exp(x^2/2)}{\pi(1+x^2)}. \tag{S.7}$$

The second moment of $w(x)$ under $q(x)$ thus is

$$\mathbb{E}_{q(x)}\left[w(x)^2\right] = \int_{-\infty}^{\infty} \frac{2\pi}{\pi^2}\frac{\exp(x^2)}{(1+x^2)^2}q(x)dx \tag{S.8}$$

$$= \int_{-\infty}^{\infty} \frac{2\pi}{\pi^2}\frac{\exp(x^2)}{(1+x^2)^2}\frac{1}{\sqrt{2\pi}}\exp(-x^2/2)dx \tag{S.9}$$

$$\propto \int_{-\infty}^{\infty} \frac{\exp(x^2/2)}{(1+x^2)^2}dx \tag{S.10}$$

The exponential function grows more quickly than any polynomial so that the integral becomes arbitrarily large. Hence, the second moment (and the variance) of $\hat{I}_n$ is unbounded, which explains the erratic behaviour of the curves in the plot.

A less formal but quicker way to see that, for this problem, a standard normal is a poor choice of an importance distribution is to note that its density decays more quickly than the Cauchy pdf in (12), which means that the standard normal pdf is "small" when the Cauchy pdf is still "large" (see Figure 1). This leads to large variance of the estimate. The overall conclusion is that the integral $\int p(x)dx$ should not be approximated with importance sampling with a Gaussian importance distribution.

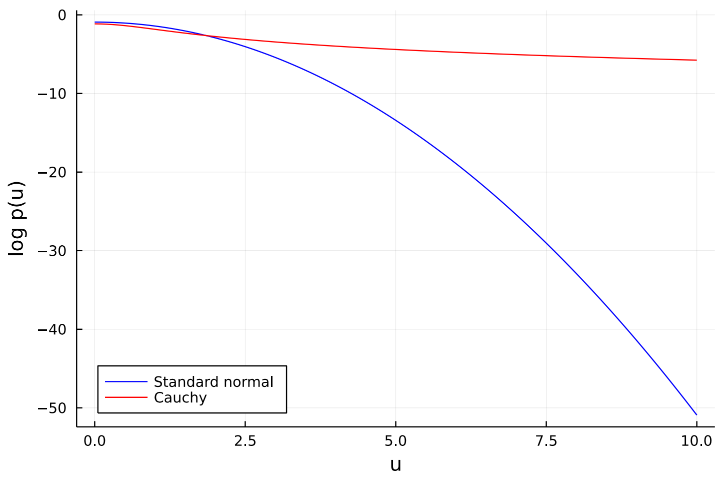


Figure 1: Exercise 2. Comparison of the log pdf of a standard normal (blue) and the Cauchy random variable (red) for positive inputs. The Cauchy pdf has much heavier tails than a Gaussian so that the Gaussian pdf is already "small" when the Cauchy pdf is still "large".

Exercise 3. ***Inverse transform sampling***

The cumulative distribution function (cdf) $F_x(\alpha)$ of a (continuous or discrete) random variable x indicates the probability that x takes on values smaller or equal to α,

$$F_x(\alpha) = \mathbb{P}(x \leq \alpha). \tag{14}$$

For continuous random variables, the cdf is defined via the integral

$$F_x(\alpha) = \int_{-\infty}^{\alpha} p_x(u)\mathrm{d}u, \tag{15}$$

where p_x denotes the pdf of the random variable x (u is here a dummy variable). Note that F_x maps the domain of x to the interval $[0,1]$. For simplicity, we here assume that F_x is invertible.

For a continuous random variable x with cdf F_x show that the random variable $y = F_x(x)$ is uniformly distributed on $[0,1]$.
Hint: Determine the cdf of y.

Importantly, this implies that for a random variable y which is uniformly distributed on $[0,1]$, the transformed random variable $F_x^{-1}(y)$ has cdf F_x. This gives rise to a method called "inverse transform sampling" to generate n iid samples of a random variable x with cdf F_x. Given a target cdf F_x, the method consists of:

- *calculating the inverse F_x^{-1}*
- *sampling n iid random variables uniformly distributed on $[0,1]$: $y^{(i)} \sim \mathcal{U}(0,1)$, $i = 1,\dots,n$.*
- *transforming each sample by F_x^{-1}: $x^{(i)} = F_x^{-1}(y^{(i)})$, $i = 1,\dots,n$.*

By construction of the method, the $x^{(i)}$ are n iid samples of x.

Solution. We start with the cumulative distribution function (cdf) F_y for y,

$$F_y(\beta) = \mathbb{P}(y \leq \beta). \tag{S.11}$$

Since $F_x(x)$ maps x to $[0,1]$, $F_y(\beta)$ is zero for $\beta < 0$ and one for $\beta > 1$. We next consider $\beta \in [0,1]$.

Let α be the value of x that F_x maps to β, i.e. $F_x(\alpha) = \beta$, which means $\alpha = F_x^{-1}(\beta)$. Since F_x is a non-decreasing function, we have

$$F_y(\beta) = \mathbb{P}(y \leq \beta) = \mathbb{P}(F_x(x) \leq \beta) = \mathbb{P}(x \leq F_x^{-1}(\beta)) = \mathbb{P}(x \leq \alpha) = F_x(\alpha). \tag{S.12}$$

Since $\alpha = F_x^{-1}(\beta)$ we obtain

$$F_y(\beta) = F_x(F_x^{-1}(\beta)) = \beta \tag{S.13}$$

The cdf F_y is thus given by

$$F_y(\beta) = \begin{cases} 0 & \text{if } \beta < 0 \\ \beta & \text{if } \beta \in [0,1] \\ 1 & \text{if } \beta > 1 \end{cases} \tag{S.14}$$

which is the cdf of a uniform random variable on $[0,1]$. Hence $y = F_x(x)$ is uniformly distributed on $[0,1]$.

Exercise 4. ***Sampling from the exponential distribution***

The exponential distribution has the density

$$p(x;\lambda) = \begin{cases} \lambda \exp(-\lambda x) & x \geq 0 \\ 0 & x < 0, \end{cases} \tag{16}$$

where λ is a parameter of the distribution. Use inverse transform sampling to generate n iid samples from $p(x;\lambda)$.

Solution. We first compute the cumulative distribution function.

$$\begin{aligned} F_x(\alpha) &= \mathbb{P}(x \leq \alpha) && \text{(S.15)} \\ &= \int_0^\alpha \lambda \exp(-\lambda x) && \text{(S.16)} \\ &= -\exp(-\lambda)\big|_0^\alpha && \text{(S.17)} \\ &= 1 - \exp(-\lambda\alpha) && \text{(S.18)} \end{aligned}$$

It's inverse is obtained by solving

$$y = 1 - \exp(-\lambda x) \tag{S.19}$$

for x, which gives:

$$\begin{aligned} \exp(-\lambda x) &= 1 - y && \text{(S.20)} \\ -\lambda x &= \log(1-y) && \text{(S.21)} \\ x &= \frac{-\log(1-y)}{\lambda} && \text{(S.22)} \end{aligned}$$

To generate samples $x^{(i)} \sim p(x;\lambda)$, we thus first sample $y^{(i)} \sim U(0,1)$, and then set

$$x^{(i)} = \frac{-\log(1-y^{(i)})}{\lambda}. \tag{S.23}$$

Inverse transform sampling can be used to generate samples from many standard distributions. For example, it allows one to generate Gaussian random variables from uniformly distributed random variables. The method is called the Box-Muller transform, see e.g. `https://en.wikipedia.org/wiki/Box-Muller_transform`. How to generate the required samples from the uniform distribution is a research field on its own, see e.g. `https://en.wikipedia.org/wiki/Random_number_generation` and `http://statweb.stanford.edu/~owen/mc/Ch-unifrng.pdf`.

Exercise 5. ***Sampling from a Laplace distribution***

A Laplace random variable x of mean zero and variance one has the density $p(x)$

$$p(x) = \frac{1}{\sqrt{2}} \exp\left(-\sqrt{2}|x|\right) \qquad x \in \mathbb{R}. \tag{17}$$

Use inverse transform sampling to generate n iid samples from x.

Solution. The main task is to compute the cumulative distribution function (cdf) F_x of x and its inverse. The cdf is by definition

$$F_x(\alpha) = \int_{-\infty}^{\alpha} \frac{1}{\sqrt{2}} \exp\left(-\sqrt{2}|u|\right) \mathrm{d}u. \tag{S.24}$$

We first consider the case where $\alpha \le 0$. Since $-|u| = u$ for $u \le 0$, we have

$$\begin{aligned}
F_x(\alpha) &= \int_{-\infty}^{\alpha} \frac{1}{\sqrt{2}} \exp\left(\sqrt{2}u\right) \mathrm{d}u && \text{(S.25)}\\
&= \frac{1}{2} \exp\left(\sqrt{2}u\right)\Big|_{-\infty}^{\alpha} && \text{(S.26)}\\
&= \frac{1}{2} \exp\left(\sqrt{2}\alpha\right). && \text{(S.27)}
\end{aligned}$$

For $\alpha > 0$, we have

$$\begin{aligned}
F_x(\alpha) &= \int_{-\infty}^{\alpha} \frac{1}{\sqrt{2}} \exp\left(-\sqrt{2}|u|\right) \mathrm{d}u && \text{(S.28)}\\
&= 1 - \int_{\alpha}^{\infty} \frac{1}{\sqrt{2}} \exp\left(-\sqrt{2}|u|\right) \mathrm{d}u && \text{(S.29)}
\end{aligned}$$

where we have used the fact that the pdf has to integrate to one. For values of $u > 0$, $-|u| = -u$, so that

$$\begin{aligned}
F_x(\alpha) &= 1 - \int_{\alpha}^{\infty} \frac{1}{\sqrt{2}} \exp\left(-\sqrt{2}u\right) \mathrm{d}u && \text{(S.30)}\\
&= 1 + \frac{1}{2} \exp\left(-\sqrt{2}u\right)\Big|_{\alpha}^{\infty} && \text{(S.31)}\\
&= 1 - \frac{1}{2} \exp\left(-\sqrt{2}\alpha\right). && \text{(S.32)}
\end{aligned}$$

In total, for $\alpha \in \mathbb{R}$, we thus have

$$F_x(\alpha) = \begin{cases} \frac{1}{2}\exp\left(\sqrt{2}\alpha\right) & \text{if } \alpha \le 0 \\ 1 - \frac{1}{2}\exp\left(-\sqrt{2}\alpha\right) & \text{if } \alpha > 0 \end{cases} \tag{S.33}$$

Figure 2 visualises $F_x(\alpha)$.

As the figure suggests, there is a unique inverse to $y = F_x(\alpha)$. For $y \le 1/2$, we have

$$\begin{aligned}
y &= \frac{1}{2} \exp\left(\sqrt{2}\alpha\right) && \text{(S.34)}\\
\log(2y) &= \sqrt{2}\alpha && \text{(S.35)}\\
\alpha &= \frac{1}{\sqrt{2}} \log(2y) && \text{(S.36)}
\end{aligned}$$

For $y > 1/2$, we have

$$\begin{aligned}
y &= 1 - \frac{1}{2} \exp\left(-\sqrt{2}\alpha\right) && \text{(S.37)}\\
-y &= -1 + \frac{1}{2} \exp\left(-\sqrt{2}\alpha\right) && \text{(S.38)}\\
1 - y &= \frac{1}{2} \exp\left(-\sqrt{2}\alpha\right) && \text{(S.39)}\\
\log(2 - 2y) &= -\sqrt{2}\alpha && \text{(S.40)}\\
\alpha &= -\frac{1}{\sqrt{2}} \log(2 - 2y) && \text{(S.41)}
\end{aligned}$$

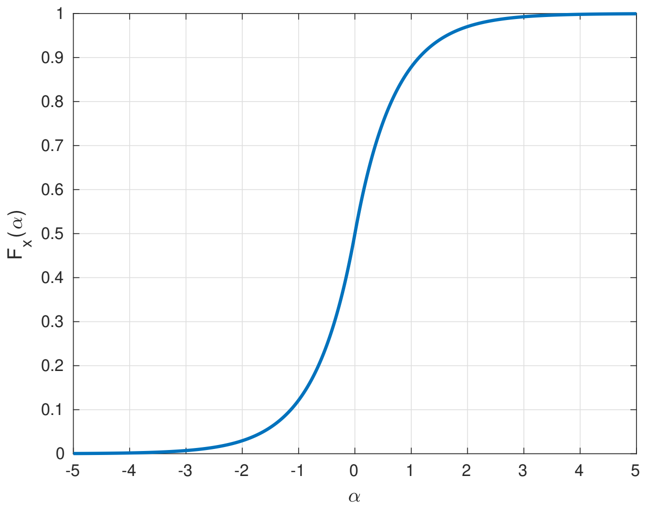
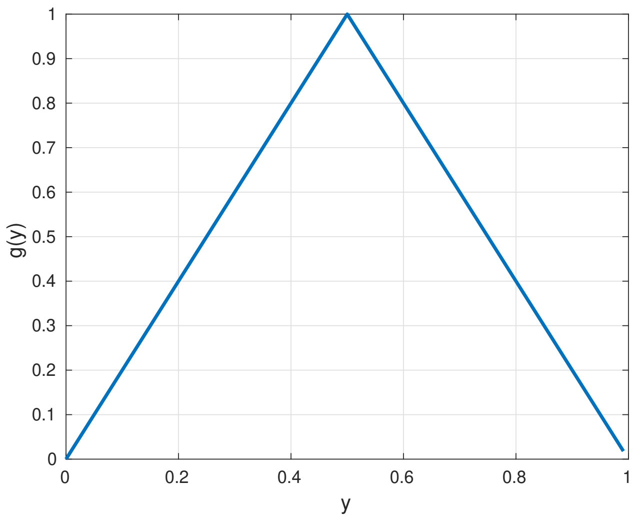


Figure 2: The cumulative distribution function $F_x(\alpha)$ for a Laplace distributed random variable.

The function $y \mapsto g(y)$ that occurs in the logarithm in both cases is

$$g(y) = \begin{cases} 2y & \text{if } y \leq \frac{1}{2} \\ 2 - 2y & \text{if } y > \frac{1}{2} \end{cases}. \tag{S.42}$$

It is shown below and can be written more compactly as $g(y) = 1 - 2|y - 1/2|$.



We thus can write the inverse $F_x^{-1}(y)$ of the cdf $y = F_x(\alpha)$ as

$$F_x^{-1}(y) = -\text{sign}\left(y - \frac{1}{2}\right) \frac{1}{\sqrt{2}} \log\left[1 - 2\left|y - \frac{1}{2}\right|\right]. \tag{S.43}$$

To generate n iid samples from x, we first generate n iid samples $y^{(i)}$ that are uniformly distributed on $[0, 1]$, and then compute for each $F_x^{-1}(y^{(i)})$. The properties of inverse transform sampling guarantee that the $x^{(i)}$,

$$x^{(i)} = F_x^{-1}(y^{(i)}), \tag{S.44}$$

are independent and Laplace distributed.

Exercise 6. *Rejection sampling*

Most compute environments provide functions to sample from a standard normal distribution. Popular algorithms include the Box-Muller transform, see e.g. `https://en.wikipedia.org/wiki/Box-Muller_transform`*. We here use rejection sampling to sample from a standard normal distribution with density $p(x)$ using a Laplace distribution as our proposal/auxiliary distribution.*[2]

The density $q(x)$ of a zero-mean Laplace distribution with variance $2b^2$ is

$$q(x;b) = \frac{1}{2b} \exp\left(-\frac{|x|}{b}\right). \tag{18}$$

We can sample from it by sampling a Laplace variable with variance 1 as in Exercise 5 and then scaling the sample by $\sqrt{2}b$.

Rejection sampling then repeats the following steps:

- *Generate $x \sim q(x;b)$*
- *Accept x with probability $f(x) = \frac{1}{M}\frac{p(x)}{q(x)}$, i.e. generate $u \sim U(0,1)$ and accept x if $u \leq f(x)$.*

(a) *Compute the ratio $M(b) = \max_x \frac{p(x)}{q(x;b)}$.*

Solution. By the definitions of the pdf $p(x)$ of a standard normal and the pdf $q(x;b)$ of the Laplace distribution, we have

$$\frac{p(x)}{q(x;b)} = \frac{\frac{1}{\sqrt{2\pi}}\exp(-x^2/2)}{\frac{1}{2b}\exp(-|x|/b)} \tag{S.45}$$

$$= \frac{2b}{\sqrt{2\pi}}\exp(-x^2/2 + |x|/b) \tag{S.46}$$

The ratio is symmetric in x. Moreover, since the exponential function is strictly increasing, we can find the maximiser of $-x^2/2 + x/b$ for $x \geq 0$ to determine the maximiser of $M(b)$. With $g(x) = -x^2/2 + x/b$, we have

$$g'(x) = -x + 1/b \tag{S.47}$$

$$g''(x) = -1 \tag{S.48}$$

The critical point (for which the first derivative is zero) is $x = 1/b$ and since the second derivative is negative for all x, the point is a maximum. The maximal ratio $M(b)$ thus is

$$M(b) = \frac{2b}{\sqrt{2\pi}}\exp\left(-x^2/2 + |x|/b\right)\Big|_{x=1/b} \tag{S.49}$$

$$= \frac{2b}{\sqrt{2\pi}}\exp\left(-1/(2b^2) + 1/b^2\right) \tag{S.50}$$

$$= \frac{2b}{\sqrt{2\pi}}\exp\left(1/(2b^2)\right) \tag{S.51}$$

(b) *How should you choose b to maximise the probability of acceptance?*

[2]Credit: This exercise is loosely based on Exercise 2.8 in Robert and Casella's *Introducing Monte Carlo Methods with R*, Springer 2010.

Solution. The probability of acceptance is $1/M$. Hence to maximise it, we have to choose b such that $M(b)$ is minimal. We compute the derivatives

$$M'(b) = \frac{2}{\sqrt{2\pi}}\exp(1/(2b^2)) - \frac{2b}{\sqrt{2\pi}}\exp(1/(2b^2))b^{-3} \quad \text{(S.52)}$$
$$= \frac{2}{\sqrt{2\pi}}\exp(1/(2b^2)) - \frac{2}{\sqrt{2\pi}}\exp(1/(2b^2))b^{-2} \quad \text{(S.53)}$$
$$= \frac{2}{\sqrt{2\pi}}\exp(1/(2b^2))(1-b^{-2}) \quad \text{(S.54)}$$
$$M''(b) = -b^{-3}\frac{2}{\sqrt{2\pi}}\exp(1/(2b^2))(1-b^{-2}) + 2b^{-3}\frac{2}{\sqrt{2\pi}}\exp(1/(2b^2)) \quad \text{(S.55)}$$
$$\quad \text{(S.56)}$$

Setting the first derivative to zero gives

$$\frac{2}{\sqrt{2\pi}}\exp(1/(2b^2)) = \frac{2}{\sqrt{2\pi}}\exp(1/(2b^2))b^{-2} \quad \text{(S.57)}$$
$$1 = b^{-2} \quad \text{(S.58)}$$

Hence the optimal $b = 1$. The second derivative at $b = 1$ is

$$M''(1) = 2\frac{2}{\sqrt{2\pi}}\exp(1/2) \quad \text{(S.59)}$$

which is positive so that the $b = 1$ is a minimum. The smallest value of M thus is

$$M(1) = \frac{2b}{\sqrt{2\pi}}\exp(1/(2b^2))\Big|_{b=1} \quad \text{(S.60)}$$
$$= \frac{2}{\sqrt{2\pi}}\exp(1/2) \quad \text{(S.61)}$$
$$= \sqrt{\frac{2e}{\pi}} \quad \text{(S.62)}$$

where $e = \exp(1)$. The maximal acceptance probability thus is

$$\frac{1}{\min_b M(b)} = \sqrt{\frac{\pi}{2e}} \quad \text{(S.63)}$$
$$\approx 0.76 \quad \text{(S.64)}$$

This means for each sample x generated from $q(x; 1)$, there is chance of 0.76 that it gets accepted. In other words, for each accepted sample, we need to generate $1/0.76 = 1.32$ samples from $q(x; 1)$.

The variance of the Laplace distribution for $b = 1$ equals 2. Hence the variance of the auxiliary distribution is larger (twice as large) as the variance of the distribution we would like to sample from.

(c) *Assume you sample from $p(x_1, \ldots, x_d) = \prod_{i=1}^d p(x_i)$ using $q(x_1, \ldots, x_d) = \prod_{i=1}^d q(x_i; b)$ as auxiliary distribution without exploiting any independencies. How does the acceptance probability scale as a function of d? You may denote the acceptance probability in case of $d = 1$ by A.*

Solution. We have to determine the maximal ratio

$$M_d = \max_{x_1,\dots,x_d} \frac{p(x_1,\dots,x_d)}{q(x_1,\dots,x_d)} \tag{S.65}$$

Plugging-in the factorisation gives

$$\begin{aligned} M_d &= \max_{x_1,\dots,x_d} \prod_{i=1}^{d} \frac{p(x_i)}{q(x_i)} && \text{(S.66)}\\ &= \prod_{i=1}^{d} \underbrace{\max_{x_i} \frac{p(x_i)}{q(x_i)}}_{M_1=1/A} && \text{(S.67)}\\ &= \prod_{i=1}^{d} \frac{1}{A} && \text{(S.68)}\\ &= \frac{1}{A^d} && \text{(S.69)} \end{aligned}$$

Hence, the acceptance probability is

$$\frac{1}{M_d} = A^d \tag{S.70}$$

Note that $A \leq 1$ since it is a probability. This means that, unless $A = 1$, we have an acceptance probability that decays exponentially in the number of dimensions if the target and auxiliary distributions factorise and we do not exploit the independencies.

Exercise 7. *Sampling from a restricted Boltzmann machine*

The restricted Boltzmann machine (RBM) is a model for binary variables $\mathbf{v} = (v_1,\dots,v_n)^\top$ *and* $\mathbf{h} = (h_1,\dots,h_m)^\top$ *which asserts that the joint distribution of* $(\mathbf{v},\mathbf{h})$ *can be described by the probability mass function*

$$p(\mathbf{v},\mathbf{h}) \propto \exp\left(\mathbf{v}^\top \mathbf{W}\mathbf{h} + \mathbf{a}^\top\mathbf{v} + \mathbf{b}^\top\mathbf{h}\right), \tag{19}$$

where $\mathbf{W}$ *is a* $n \times m$ *matrix, and* $\mathbf{a}$ *and* $\mathbf{b}$ *vectors of size* n *and* m*, respectively. Both the* v_i *and* h_i *take values in* $\{0,1\}$*. The* v_i *are called the "visibles" variables since they are assumed to be observed while the* h_i *are the hidden variables since it is assumed that we cannot measure them.*

Explain how to use Gibbs sampling to generate samples from the marginal $p(\mathbf{v})$*,*

$$p(\mathbf{v}) = \frac{\sum_{\mathbf{h}} \exp\left(\mathbf{v}^\top \mathbf{W}\mathbf{h} + \mathbf{a}^\top\mathbf{v} + \mathbf{b}^\top\mathbf{h}\right)}{\sum_{\mathbf{h},\mathbf{v}} \exp\left(\mathbf{v}^\top \mathbf{W}\mathbf{h} + \mathbf{a}^\top\mathbf{v} + \mathbf{b}^\top\mathbf{h}\right)}, \tag{20}$$

for any given values of $\mathbf{W}$*,* $\mathbf{a}$*, and* $\mathbf{b}$*.*

Hint: *You may use that*

$$p(\mathbf{h}|\mathbf{v}) = \prod_{i=1}^{m} p(h_i|\mathbf{v}), \qquad p(h_i = 1|\mathbf{v}) = \frac{1}{1 + \exp\left(-\sum_j v_j W_{ji} - b_i\right)}, \tag{21}$$

$$p(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{n} p(v_i|\mathbf{h}), \qquad p(v_i = 1|\mathbf{h}) = \frac{1}{1 + \exp\left(-\sum_j W_{ij} h_j - a_i\right)}. \tag{22}$$

Solution. In order to generate samples $\mathbf{v}^{(k)}$ from $p(\mathbf{v})$ we generate samples $(\mathbf{v}^{(k)}, \mathbf{h}^{(k)})$ from $p(\mathbf{v}, \mathbf{h})$ and then ignore the $\mathbf{h}^{(k)}$.

Gibbs sampling is a MCMC method to produce a sequence of samples $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \ldots$ that follow a pdf/pmf $p(\mathbf{x})$ (if the chain is run long enough). Assuming that $\mathbf{x}$ is d-dimensional, we generate the next sample $\mathbf{x}^{(k+1)}$ in the sequence from the previous sample $\mathbf{x}^{(k)}$ by:

1. picking (randomly) an index $i \in \{1, \ldots, d\}$
2. sampling $x_i^{(k+1)}$ from $p(x_i \mid \mathbf{x}_{\setminus i}^{(k)})$ where $\mathbf{x}_{\setminus i}^{(k)}$ is vector $\mathbf{x}$ with x_i removed, i.e. $\mathbf{x}_{\setminus i}^{(k)} = (x_1^{(k)}, \ldots, x_{i-1}^{(k)}, x_{i+1}^{(k)}, \ldots, x_d^{(k)})$
3. setting $\mathbf{x}^{(k+1)} = (x_1^{(k)}, \ldots, x_{i-1}^{(k)}, x_i^{(k+1)}, x_{i+1}^{(k)}, \ldots, x_d^{(k)})$.

For the RBM, the tuple $(\mathbf{h}, \mathbf{v})$ corresponds to $\mathbf{x}$ so that a x_i in the above steps can either be a hidden variable or a visible. Hence

$$p(x_i \mid \mathbf{x}_{\setminus i}) = \begin{cases} p(h_i \mid \mathbf{h}_{\setminus i}, \mathbf{v}) & \text{if } x_i \text{ is a hidden variable } h_i \\ p(v_i \mid \mathbf{v}_{\setminus i}, \mathbf{h}) & \text{if } x_i \text{ is a visible variable } v_i \end{cases} \tag{S.71}$$

($\mathbf{h}_{\setminus i}$ denotes the vector $\mathbf{h}$ with element h_i removed, and equivalently for $\mathbf{v}_{\setminus i}$)

To compute the conditionals on the right hand side, we use the hint:

$$p(\mathbf{h}|\mathbf{v}) = \prod_{i=1}^{m} p(h_i|\mathbf{v}), \qquad p(h_i = 1|\mathbf{v}) = \frac{1}{1 + \exp\left(-\sum_j v_j W_{ji} - b_i\right)}, \tag{S.72}$$

$$p(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{n} p(v_i|\mathbf{h}), \qquad p(v_i = 1|\mathbf{h}) = \frac{1}{1 + \exp\left(-\sum_j W_{ij} h_j - a_i\right)}. \tag{S.73}$$

Given the independencies between the hiddens given the visibles and vice versa, we have

$$p(h_i \mid \mathbf{h}_{\setminus i}, \mathbf{v}) = p(h_i \mid \mathbf{v}) \qquad p(v_i \mid \mathbf{v}_{\setminus i}, \mathbf{h}) = p(v_i \mid \mathbf{h}) \tag{S.74}$$

so that the expressions for $p(h_i = 1|\mathbf{v})$ and $p(v_i = 1|\mathbf{h})$ allow us to implement the Gibbs sampler.

Given the independencies, it makes further sense to sample the $\mathbf{h}$ and $\mathbf{v}$ variables in blocks: first we sample all the h_i given $\mathbf{v}$, and then all the v_i given the $\mathbf{h}$ (or vice versa). This is also known as block Gibbs sampling.

In summary, given a sample $(\mathbf{h}^{(k)}, \mathbf{v}^{(k)})$, we thus generate the next sample $(\mathbf{h}^{(k+1)}, \mathbf{v}^{(k+1)})$ in the sequence as follows:

- For all h_i, $i = 1, \ldots, m$:
  - compute $p_i^h = p(h_i = 1|\mathbf{v}^{(k)})$
  - sample u_i from a uniform distribution on $[0, 1]$ and set $h_i^{(k+1)}$ to 1 if $u_i \leq p_i^h$.
- For all v_i, $i = 1, \ldots, n$:
  - compute $p_i^v = p(v_i = 1|\mathbf{h}^{(k+1)})$
  - sample u_i from a uniform distribution on $[0, 1]$ and set $v_i^{(k+1)}$ to 1 if $u_i \leq p_i^v$.

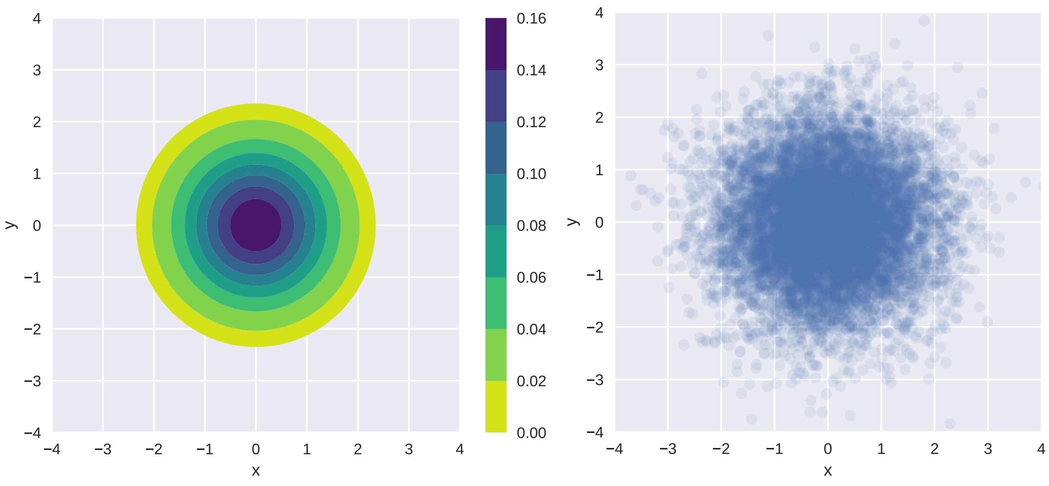
As final step, after sampling S pairs $(\mathbf{h}^{(k)}, \mathbf{v}^{(k)})$, $k = 1, \ldots, S$, the set of visibles $\mathbf{v}^{(k)}$ form samples from the marginal $p(\mathbf{v})$.

Exercise 8. *Basic Markov chain Monte Carlo inference (optional, not examinable)*

This exercise is on sampling and approximate inference by Markov chain Monte Carlo (MCMC). MCMC can be used to obtain samples from a probability distribution, e.g. a posterior distribution. The samples approximately represent the distribution, as illustrated in Figure 3, and can be used to approximate expectations.

We denote the density of a zero mean Gaussian with variance σ^2 by $\mathcal{N}(x; \mu, \sigma^2)$, i.e.

$$\mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{23}$$



(a) True density (b) Density represented by 10,000 samples.

Figure 3: Density and samples from $p(x, y) = \mathcal{N}(x; 0, 1)\mathcal{N}(y; 0, 1)$.

Consider a vector of d random variables $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_d)$ and some observed data $\mathcal{D}$. In many cases, we are interested in computing expectations under the posterior distribution $p(\boldsymbol{\theta} \mid \mathcal{D})$, e.g.

$$\mathbb{E}_{p(\boldsymbol{\theta}|\mathcal{D})}\left[g(\boldsymbol{\theta})\right] = \int g(\boldsymbol{\theta})p(\boldsymbol{\theta} \mid \mathcal{D})\mathrm{d}\boldsymbol{\theta} \tag{24}$$

for some function $g(\boldsymbol{\theta})$. If d is small, e.g. $d \leq 3$, deterministic numerical methods can be used to approximate the integral to high accuracy.[3] *But for higher dimensions, these methods are generally not applicable any more. The expectation, however, can be approximated as a sample average if we have samples $\boldsymbol{\theta}^{(i)}$ from $p(\boldsymbol{\theta} \mid \mathcal{D})$:*

$$\mathbb{E}_{p(\boldsymbol{\theta}|\mathcal{D})}\left[g(\boldsymbol{\theta})\right] \approx \frac{1}{S}\sum_{i=1}^{S} g(\boldsymbol{\theta}^{(i)}) \tag{25}$$

Note that in MCMC methods, the samples $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(S)}$ used in the above approximation are typically not statistically independent.

Metropolis-Hastings is an MCMC algorithm that generates samples from a distribution $p(\boldsymbol{\theta})$, where $p(\boldsymbol{\theta})$ can be any distribution on the parameters (and not only posteriors). The algorithm is iterative and at iteration t, it uses:

- *a proposal distribution $q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(t)})$, parametrised by the current state of the Markov chain, i.e. $\boldsymbol{\theta}^{(t)}$;*
- *a function $p^*(\boldsymbol{\theta})$, which is proportional to $p(\boldsymbol{\theta})$. In other words, $p^*(\boldsymbol{\theta})$ is unnormalised*[4] *and the normalised density $p(\boldsymbol{\theta})$ is*

$$p(\boldsymbol{\theta}) = \frac{p^*(\boldsymbol{\theta})}{\int p^*(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}}. \tag{26}$$

[3] See e.g. https://en.wikipedia.org/wiki/Numerical_integration.

[4] We used the notation $\tilde{p}$ in the lecture slides; p^* is also commonly used, e.g. in Barber's book.

For all tasks in this exercise, we work with a Gaussian proposal distribution $q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(t)})$, whose mean is the previous sample in the Markov chain, and whose variance is ϵ^2. That is, at iteration t of our Metropolis-Hastings algorithm,

$$q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(t-1)}) = \prod_{k=1}^{d} \mathcal{N}(\theta_k; \theta_k^{(t-1)}, \epsilon^2). \tag{27}$$

When used with this proposal distribution, the algorithm is called Random Walk Metropolis-Hastings algorithm.

(a) *Read Section 27.4 in Barber's book to familiarise yourself with the Metropolis-Hastings algorithm.*

(b) *Write a function mh implementing the Metropolis Hasting algorithm, as given in Algorithm 27.3 in Barber's book, using the Gaussian proposal distribution in (27) above. The function should take as arguments*

- *p_star: a function on $\boldsymbol{\theta}$ that is proportional to the density of interest $p^*(\boldsymbol{\theta})$;*
- *param_init: the initial sample — a value for $\boldsymbol{\theta}$ from where the Markov chain starts;*
- *num_samples: the number S of samples to generate;*
- *vari: the variance ϵ^2 for the Gaussian proposal distribution q;*

and return $[\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(S)}]$ — a list of S samples from $p(\boldsymbol{\theta}) \propto p^*(\boldsymbol{\theta})$. For example:

```
def mh(p_star, param_init, num_samples=5000, vari=1.0):
    # your code here
    return samples
```

Solution. Below is a Python implementation.

```
def mh(p_star, param_init, num_samples=5000, vari=1.0):
   x = []

   x_current = param_init
   for n in range(num_samples):

     # proposal
     x_proposed = multivariate_normal.rvs(mean=x_current, cov=vari)

     # MH step
     a = multivariate_normal.pdf(x_current, mean=x_proposed, cov=vari) *
         p_star(x_proposed)
     a = a / (multivariate_normal.pdf(x_proposed, mean=x_current, cov=vari)
         * p_star(x_current))

     # accept or not
     if a >= 1:
       x_next = np.copy(x_proposed)
     elif uniform.rvs(0, 1) < a:
       x_next = np.copy(x_proposed)
     else:
       x_next = np.copy(x_current)

     # keep record
```

```
        x.append(x_next)
        x_current = x_next

    return x
```

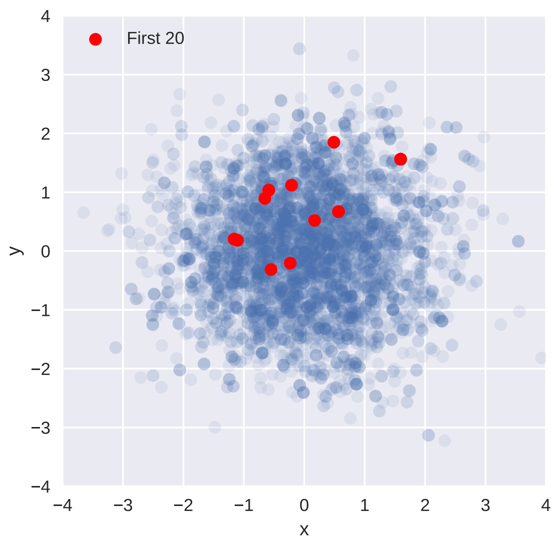
As we are using a symmetrical proposal distribution, $q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^*) = q(\boldsymbol{\theta}^* \mid \boldsymbol{\theta})$, and one could simplify the algorithm by having $a = \frac{p^*(\boldsymbol{\theta}^*)}{p^*(\boldsymbol{\theta})}$, where $\boldsymbol{\theta}$ is the current sample and $\boldsymbol{\theta}^*$ is the proposed sample.

In practice, it is desirable to implement the function in the log domain, to avoid numerical problems. That is, instead of p^*, `mh` will accept as an argument $\log p^*$, and a will be calculated as:

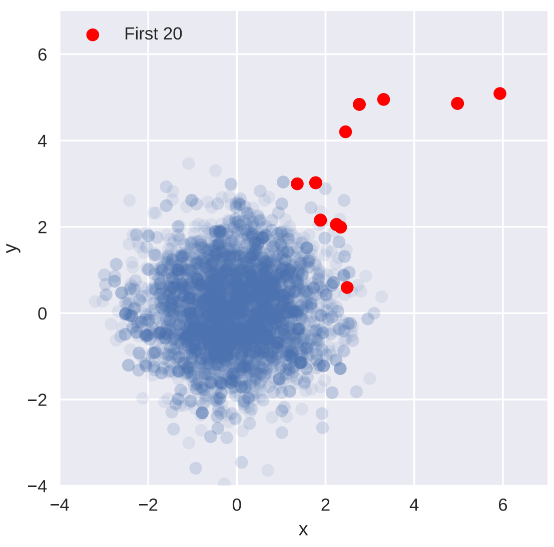
$$a = (\log q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^*) + \log p^*(\boldsymbol{\theta}^*)) - (\log q(\boldsymbol{\theta}^* \mid \boldsymbol{\theta}) + \log p^*(\boldsymbol{\theta}))$$

(c) *Test your algorithm by sampling* $5,000$ *samples from* $p(x, y) = \mathcal{N}(x; 0, 1)\mathcal{N}(y; 0, 1)$. *Initialise at* $(x = 0, y = 0)$ *and use* $\epsilon^2 = 1$. *Generate a scatter plot of the obtained samples. The plot should be similar to Figure 3b. Highlight the first 20 samples only. Do these 20 samples alone adequately approximate the true density?*

Sample another $5,000$ *points from* $p(x, y) = \mathcal{N}(x; 0, 1)\mathcal{N}(y; 0, 1)$ *using* `mh` *with* $\epsilon^2 = 1$, *but this time initialise at* $(x = 7, y = 7)$. *Generate a scatter plot of the drawn samples and highlight the first 20 samples. If everything went as expected, your plot probably shows a "trail" of samples, starting at* $(x = 7, y = 7)$ *and slowly approaching the region of space where most of the probability mass is.*



(a) Starting the chain at $(0, 0)$.



(b) Starting the chain at $(7, 7)$.

Figure 4: $5,000$ samples from $\mathcal{N}(x; 0, 1)\mathcal{N}(y; 0, 1)$ (blue), with the first 20 samples highlighted (red). Drawn using Metropolis-Hastings with different starting points.

Solution. Figure 4 shows the two scatter plots of draws from $\mathcal{N}(x; 0, 1)\mathcal{N}(y; 0, 1)$:

- Figure 4a highlights the first 20 samples obtained by the chain when starting at $(x = 0, y = 0)$. They appear to be representative samples from the distribution, however, they are not enough to approximate the distribution on their own. This would mean that a sample average computed with 20 samples only would have high variance, i.e. its value would depend strongly on the values of the 20 samples used to compute the average.

- Figure 4b highlights the first 20 samples obtained by the chain when starting at $(x = 7, y = 7)$. One can clearly see the “burn-in” tail which slowly approaches the region where most of the probability mass is.

(d) *In practice, we don't know where the distribution we wish to sample from has high density, so we typically initialise the Markov Chain somewhat arbitrarily, or at the maximum a-posterior (MAP) sample if available. The samples obtained in the beginning of the chain are typically discarded, as they are not considered to be representative of the target distribution. This initial period between initialisation and starting to collect samples is called “warm-up”, or also “burn-in”.*

Extended your function mh *to include an additional warm-up argument W, which specifies the number of MCMC steps taken before starting to collect samples. Your function should still return a list of S samples as in (b).*

Solution. We can extend the mh function with a warm-up argument by, for example, iterating for num_samples + warmup steps, and start recording samples only after the warm-up period:

```python
def mh(p_star, param_init, num_samples=5000, vari=1.0, warmup=0):
    x = []
    x_current = param_init
    for n in range(num_samples+warmup):
        ... # body same as before

        if n >= warmup: x.append(x_next)
        x_current = x_next

    return x
```

Exercise 9. *Bayesian Poisson regression* (optional, not examinable)

Consider a Bayesian Poisson regression model, where outputs y_n are generated from a Poisson distribution of rate $\exp(\alpha x_n+\beta)$, where the x_n are the inputs (covariates), and α and β the parameters of the regression model for which we assume a broad Gaussian prior:

$$\alpha \sim \mathcal{N}(\alpha; 0, 100) \tag{28}$$
$$\beta \sim \mathcal{N}(\beta; 0, 100) \tag{29}$$
$$y_n \sim \text{Poisson}(y_n; \exp(\alpha x_n + \beta)) \quad \textit{for } n = 1, \dots, N \tag{30}$$

Poisson$(y; \lambda)$ *denotes the probability mass function of a Poisson random variable with rate λ,*

$$\text{Poisson}(y; \lambda) = \frac{\lambda^y}{y!}\exp(-\lambda), \qquad y \in \{0, 1, 2, \dots\}, \quad \lambda > 0 \tag{31}$$

Consider $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^N$ where $N = 5$ and

$$(x_1, \dots, x_5) = (-0.50519053, -0.17185719, 0.16147614, 0.49480947, 0.81509851) \tag{32}$$
$$(y_1, \dots, y_5) = (1, 0, 2, 1, 2) \tag{33}$$

We are interested in computing the posterior density of the parameters (α, β) given the data $\mathcal{D}$ above.

(a) *Derive an expression for the unnormalised posterior density of α and β given $\mathcal{D}$, i.e. a function p^* of the parameters α and β that is proportional to the posterior density $p(\alpha, \beta \mid \mathcal{D})$, and which can thus be used as target density in the Metropolis Hastings algorithm.*

Solution. By the product rule, the joint distribution described by the model, with $\mathcal{D}$ plugged in, is proportional to the posterior and hence can be taken as p^*:

$$p^*(\alpha, \beta) = p(\alpha, \beta, \{(x_n, y_n)\}_{n=1}^N) \tag{S.75}$$

$$= \mathcal{N}(\alpha; 0, 100)\mathcal{N}(\beta; 0, 100) \prod_{n=1}^{N} \text{Poisson}(y_n \mid \exp(\alpha x_n + \beta)) \tag{S.76}$$

(b) *Implement the derived unnormalised posterior density p^*. If your coding environment provides an implementation of the above Poisson pmf, you may use it directly rather than implementing the pmf yourself.*

Use the Metropolis Hastings algorithm from Question 8(c) to draw $5,000$ samples from the posterior density $p(\alpha, \beta \mid \mathcal{D})$. Set the hyperparameters of the Metropolis-Hastings algorithm to:

- *param_init* $= (\alpha_{\text{init}}, \beta_{\text{init}}) = (0, 0)$,
- *vari* $= 1$, *and*
- *number of warm-up steps* $W = 1000$.

Plot the drawn samples with x-axis α and y-axis β and report the posterior mean of α and β, as well as their correlation coefficient under the posterior.

Solution. A Python implementation is:

```
import numpy as np
from scipy.stats import multivariate_normal, norm, poisson, uniform

xx1 = np.array([-0.5051905265552105, -0.17185719322187715,
    0.16147614011145617, 0.49480947344478954, 0.8150985069051909])
yy1 = np.array([1, 0, 2, 1, 2])
N1 = len(xx1)

def poisson_regression(params):
    a = params[0]
    b = params[1]
    # mean zero, standard deviation 10 == variance 100
    p = norm.pdf(a, loc=0, scale=10) * norm.pdf(b, loc=0, scale=10)
    for n in range(N1):
        p = p * poisson.pmf(yy1[n], np.exp(a * xx1[n] + b))

    return p

# sample
S = 5000
samples = np.array(mh(poisson_regression, np.array([0, 0]), num_samples=S,
     vari=1.0, warmup=1000))
```

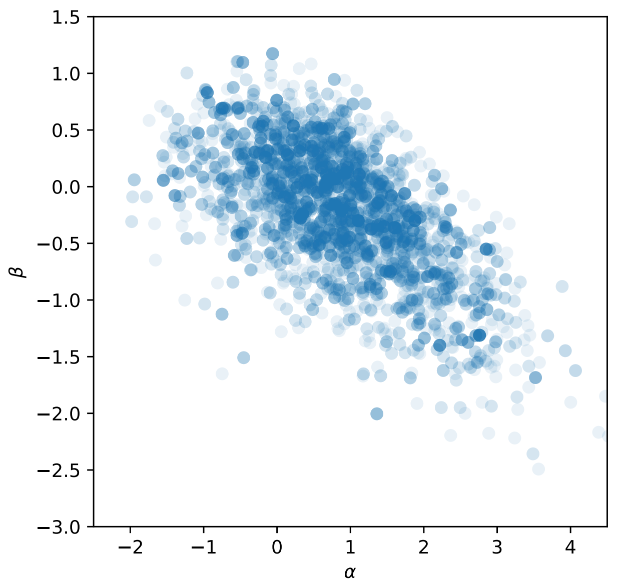


Figure 5: Posterior samples for Poisson regression problem; $\boldsymbol{\theta}_{\text{init}} = (0, 0)$.

A scatter plot showing $5,000$ samples from the posterior is shown on Figure 5. The posterior mean of α is 0.84, the posterior mean of β is -0.2, and posterior correlation coefficient is -0.63. Note that the numerical values are sample-specific.

Exercise 10. *Mixing and convergence of Metropolis-Hasting MCMC (optional, not examinable)*

Under weak conditions, an MCMC algorithm is an asymptotically exact inference algorithm, meaning that if it is run forever, it will generate samples that correspond to the desired probability distribution. In this case, the chain is said to converge.

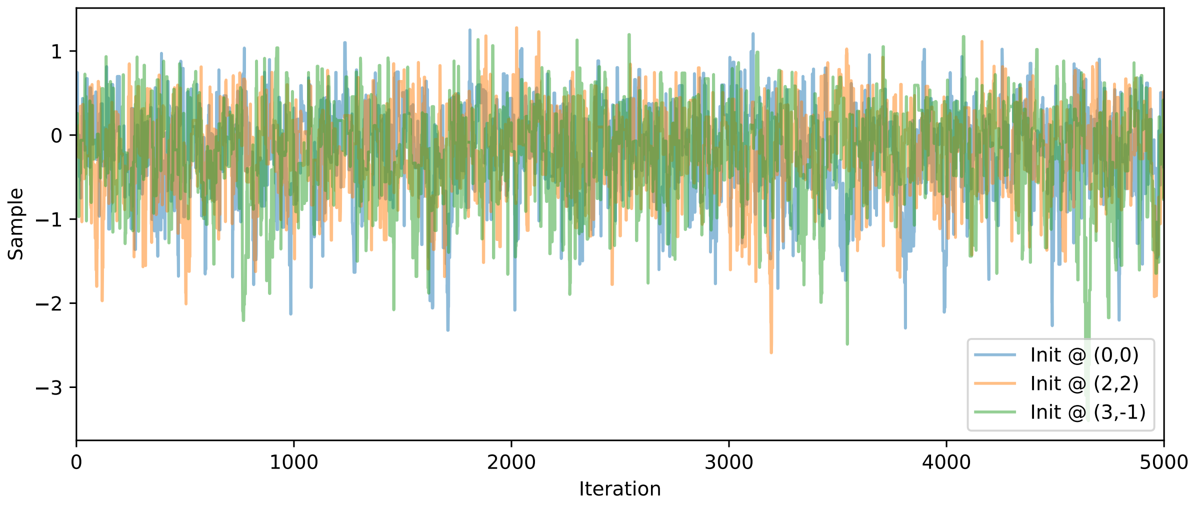
In practice, we want to run the algorithm long enough to be able to approximate the posterior adequately. How long is long enough for the chain to converge varies drastically depending on the algorithm, the hyperparameters (e.g. the variance vari*), and the target posterior distribution. It is impossible to determine exactly whether the chain has run long enough, but there exist various diagnostics that can help us determine if we can "trust" the sample-based approximation to the posterior.*

A very quick and common way of assessing convergence of the Markov chain is to visually inspect the trace plots *for each parameter. A trace plot shows how the drawn samples evolve through time, i.e. they are a time-series of the samples generated by the Markov chain. Figure 6 shows examples of trace plots obtained by running the Metropolis Hastings algorithm for different values of the hyperparameters* vari *and* param_init*. Ideally, the time series covers the whole domain of the target distribution and it is hard to "see" any structure in it so that predicting values of future samples from the current one is difficult. If so, the samples are likely independent from each other and the chain is said to be well "mixed".*

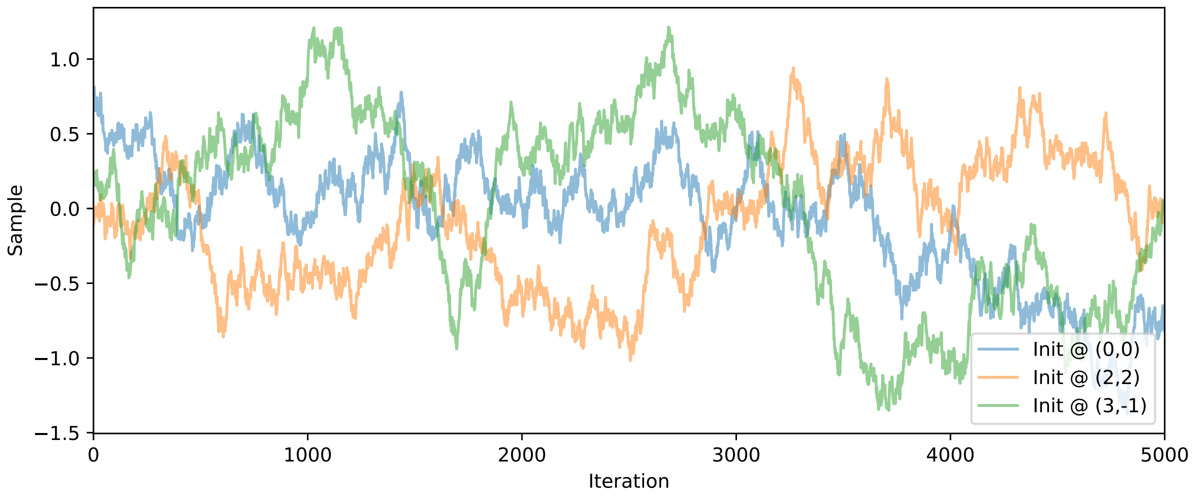
(a) *Consider the trace plots in Figure 6: Is the variance* vari *used in Figure 6b larger or smaller than the value of* vari *used in Figure 6a? Is* vari *used in Figure 6c larger or smaller than the value used in Figure 6a?*

In both cases, explain the behaviour of the trace plots in terms of the workings of the Metropolis Hastings algorithm and the effect of the variance vari*.*

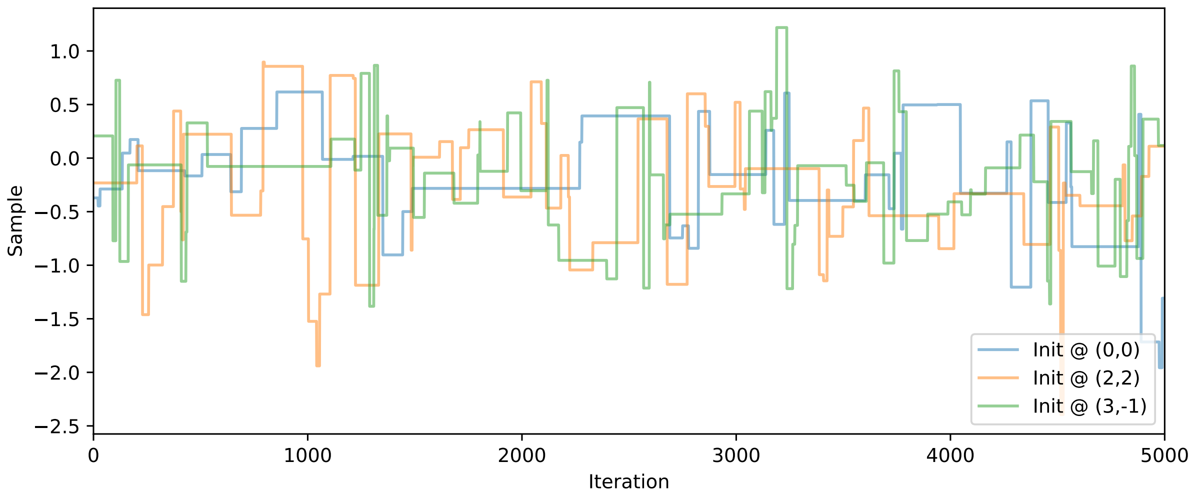
Solution. MCMC methods are sensitive to different hyperparameters, and we usually need to carefully diagnose the inference results to ensure that our algorithm adequately



(a) variance vari: 1



(b) Alternative value of vari



(c) Alternative value of vari

Figure 6: For Question 10(a): Trace plots of the parameter β from Question 9 drawn using Metropolis-Hastings with different variances of the proposal distribution.

approximates the target posterior distribution.

(i) Figure 6b uses a *small* variance (`vari` was set to 0.001) . The trace plots show that the samples for β are very highly correlated and evolve very slowly through time. This is because the introduced randomness is quite small compared to the scale of the posterior, thus the proposed sample at each MCMC iteration will be very close to the current sample and hence likely accepted.

More mathematical explanation: for a symmetric proposal distribution, the acceptance ratio a becomes

$$a = \frac{p^*(\boldsymbol{\theta}^*)}{p^*(\boldsymbol{\theta})}, \tag{S.77}$$

where $\boldsymbol{\theta}$ is the current sample and $\boldsymbol{\theta}^*$ is the proposed sample. For variances that are small compared to the (squared) scale of the posterior, a is close to one and the proposed sample $\boldsymbol{\theta}^*$ gets likely accepted. This then gives rise to the slowly changing time series shown in Figure 6b.

(ii) In Figure 6c, the variance is larger than the reference (`vari` was set to 50) . The trace plots suggest that many iterations of the algorithm result in the proposed sample being rejected, and thus we end up copying the same sample over and over again. This is because if the random perturbations are large compared to the scale of the posterior, $p^*(\boldsymbol{\theta}^*)$ may be very different from $p^*(\boldsymbol{\theta})$ and a may be very small.

(b) *In Metropolis-Hastings, and MCMC in general, any sample depends on the previously generated sample, and hence the algorithm generates samples that are generally statistically dependent. The effective sample size of a sequence of dependent samples is the number of independent samples that are, in some sense, equivalent to our number of dependent samples. A definition of the effective sample size (ESS) is*

$$ESS = \frac{S}{1 + 2\sum_{k=1}^{\infty} \rho(k)} \tag{34}$$

where S is the number of dependent samples drawn and $\rho(k)$ the correlation coefficient between two samples in the Markov chain that are k time points apart. We can see that if the samples are strongly correlated, $\sum_{k=1}^{\infty} \rho(k)$ is large and the effective sample size is small. On the other hand, if $\rho(k) = 0$ for all k, the effective sample size is S.

ESS, as defined above, is the number of independent samples which are needed to obtain a sample average that has the same variance as the sample average computed from correlated samples.

To illustrate how correlation between samples is related to a reduction of sample size, consider two pairs of samples (θ_1, θ_2) and (ω_1, ω_2). All variables have variance σ^2 and the same mean μ, but ω_1 and ω_1 are uncorrelated while the covariance matrix for θ_1, θ_2 is $\mathbf{C}$,

$$\mathbf{C} = \sigma^2 \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}, \tag{35}$$

with $\rho > 0$. The variance of the average $\bar{\omega} = 0.5(\omega_1 + \omega_2)$ is

$$\mathbb{V}(\bar{\omega}) = \frac{\sigma^2}{2}, \tag{36}$$

where the 2 in the denominator is the sample size.

Derive an equation for the variance of $\bar{\theta} = 0.5(\theta_1 + \theta_2)$ and compute the reduction α of the sample size when working with the correlated (θ_1, θ_2). In other words, derive an equation of α in

$$\mathbb{V}(\bar{\theta}) = \frac{\sigma^2}{2/\alpha}. \tag{37}$$

What is the effective sample size $2/\alpha$ as $\rho \to 1$?

Solution. Note that $\mathbb{E}(\bar{\theta}) = \mu$. From the definition of variance, we then have

$$\begin{aligned}
\mathbb{V}(\bar{\theta}) &= \mathbb{E}\left((\bar{\theta} - \mu)^2\right) && \text{(S.78)}\\
&= \mathbb{E}\left(\left(\frac{1}{2}(\theta_1 + \theta_2) - \mu\right)^2\right) && \text{(S.79)}\\
&= \mathbb{E}\left(\left(\frac{1}{2}(\theta_1 - \mu + \theta_2 - \mu\right)^2\right) && \text{(S.80)}\\
&= \frac{1}{4}\mathbb{E}\left((\theta_1 - \mu)^2 + (\theta_2 - \mu)^2 + 2(\theta_1 - \mu)(\theta_2 - \mu)\right) && \text{(S.81)}\\
&= \frac{1}{4}(\sigma^2 + \sigma^2 + 2\sigma^2\rho) && \text{(S.82)}\\
&= \frac{1}{4}(2\sigma^2 + 2\sigma^2\rho) && \text{(S.83)}\\
&= \frac{\sigma^2}{2}(1 + \rho) && \text{(S.84)}\\
&= \frac{\sigma^2}{2/(1 + \rho)} && \text{(S.85)}
\end{aligned}$$

Hence: $\alpha = (1 + \rho)$, and for $\rho \to 1$, $2/\alpha \to 1$.

Because of the strong correlation, we effectively only have one sample and not two if $\rho \to 1$.