Theory and Practice of Monte Carlo Methods

Paweł Lorek and Tomasz Rolski

Mathematical Institute, University of Wrocław

Wrocław, September 22, 2024

Contents

Chapter I

Introduction

1 About the book

Stochastic simulations and the *theory of Monte Carlo* very often refer to the same theory. However, in the opinion of the authors of this book it is worth distinguishing them. The first term will therefore be related to the theory of generators and methods of generating random numbers with prescribed distributions. We should be talking about pseudorandom numbers, because only such can be generated on the computer, but often in the text we will be talking about variables or random numbers. By the Monte Carlo theory, we will understand the theoretical foundations for the development of results, simulation planning, and construction of methods that allow for solving specific tasks, etc.

Contrary to numerical methods, where the developed algorithms allow for deterministic control of errors, in the case of calculations using stochastic methods, we get a random result, and it is vital to understand how such a result should be interpreted, what we mean by error, etc. For this, concepts and methods of mathematical statistics are helpful.

The book is written for students of mathematical sciences of the University of Wrocław, in particular, students interested in computer science and applications of probability calculus. Therefore, the main idea of this book is to simulate various tasks from the probability calculus. The reader learns about areas that are difficult to understand without specialist knowledge of probability and stochastic processes. The reader is not assumed to have detailed knowledge of these subjects but having only basic knowl-

edge from the introductory university course in the theory of probability and mathematical statistics. However, after completing the course, in addition to knowledge of the stochastic simulation and Monte Carlo theory, the authors of the book hope the reader will be familiar with many of the classical problems of probability theory, stochastic models in operations research and telecommunications, etc. In today's literature, random number generators are distinguished from pseudorandom number generators. In this lecture, we will deal with pseudorandom numbers, that is, numbers obtained by some mathematical recursion. On the other hand, we can get sequences of random numbers, for example, by tossing a coin, throwing a dice, observing certain atmospheric phenomena, recording registers, etc. A distinction is made between hardware generators of random numbers and software random number generators. However, we will not do it here. In addition to pseudorandom numbers, we also have quasirandom numbers. They are numerical sequences, which have the property of the so-called "low discrepancy".

The book consists of chapters on:

- an outline of the theory of pseudorandom number generators,
- methods of generating random variables with prescribed distributions,
- basic concepts of errors, significance level, number of replications and their relationships,
- exemplary tasks solved using stochastic simulation methods,
- methods of reducing the number of replications at a given level error,
- methods based on the theory of Markov chains (Markov chain Monte Carlo), mainly for studying the probabilistic algorithms and for sampling from probability distributions on large state spaces. Emphasis is put on applications in stochastic optimisation.
- outline methods useful in Operations Research, like discrete event simulation, and telecommunication modelling.

At the end of this book, there is enclosed the appendix surveying distributions and their properties.

2 Simulations of simple probabilistic tasks

Just as an introduction to simulations, we will discuss how to simulate numbers on a computer and consider some basic models. Pseudorandom numbers are generated on the computer. For example, the simplest way to generate a pseudorandom number in MATLAB is done via the command

```
rand                % random number from (0,1)
rand(M)             % random integer from {0,...,M-1}
```

Similarly, in `Python` (we will often use `NumPy` library)

```
np.random.rand()       # random number from (0,1)
np.random.randint(M)   # random integer from {0,...,M-1}
```

Successive numbers are obtained by repeating those commands. The sequence of pseudorandom numbers "pretends" the theoretical sequence of independent random variables with the same uniform distribution $\mathcal{U}[[0,1))$ or $\mathcal{U}[\{0,\ldots,M-1\})$.

2.1 Properties of the simple symmetric random walk

Chapter 3 of Feller [40] considers the series of symmetric coin toss problems. There are two players, A and B. One of them bets on heads and the other on tails. If player A guesses the result, he wins +1 from player B; otherwise, player B wins one from player A. We assume that the results of subsequent throws are independent. Suppose players start playing with an empty account, i.e., $S_0 = 0$. Let S_n be the fortune of player A after n tosses. It is be convenient to deal with $2N$ tosses, and ties can only appear at steps $2, 4, \ldots, 2N$. One can draw a straight line connecting points $(i-1, S_{i-1}), (i, S_i)$. It is said that the segment $(i-1, S_{i-1}) \to (i, S_i)$ is above the x-axis, if $S_{i-1} \geq 0$ or $S_i \geq 0$.

We can ask various questions about the course of the game, i.e., the realization of $(S_n)_{n=1,\ldots,2N}$. For example:

Q1 Let $L_{2N} = \max\{i : S_i = 0\}$ be the last moment of a tie. What can we say about the distribution of R_{2N}?

Q2 What is the probability $P(\alpha, \beta)$ (where $\alpha, \beta \in [0,1)$ and $\alpha < \beta$) that during $2N$ tosses player A will be winning between α and β fraction of the time?

Q3 How do the oscillations of $(S_n)_{0\leq n\leq 2N}$ look like?

Q4 What is the probability that one of the players will always win?

We formalise these types of tasks as follows, allowing for simulations. Let $X_1, X_2, \ldots,$ be a sequence of i.i.d random variables with distribution $\mathbb{P}(X_j = 1) = \mathbb{P}(X_j = -1) = 1/2$, then define

$$S_0 = 0, \qquad S_n = \sum_{j=1}^{n} X_j, \qquad n = 1, \ldots$$

The sequence $S_0 = 0, S_1, \ldots$ is called the *simple symmetric random walk*. Formally we define

$$L_{2n} = \sup\{m \leq 2n : S_m = 0\}$$

and

$$L^+_{2n} = \#\{j = 1, \ldots, 2n : S_{j-1} \geq 0 \quad \text{or} \quad S_j \geq 0\}.$$

Another characteristic might be that player A wins strictly at m if $S_m > 0$. A total strictly winning time of player A is then given by

$$\#\{1 \leq k \leq 2N : S_k > 0\}.$$

An example of a simulation of a simple symmetric random walk $S_0 = 0, S_1, \ldots, S_N$ is provided in the Listing I.1 (Python).

Matlab/Python listing I.1: Simple symmetric random walk

```
import matplotlib.pyplot as plt
N=500
bi=2*np.random.randint(0,2,2*N)-1     # 2N pseudorandom bits
y=np.cumsum(bi)
plt.plot(np.arange(2*N),y)
plt.show()
```

A sample plot of a random walk is presented in Fig. 2.1

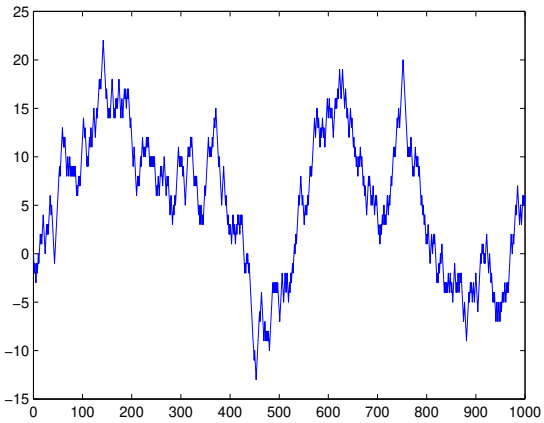


Figure 2.1: A sample realization of a simple symmetric random walk; $2N = 1000$.

Analytical answers in the form of limit theorems – when $N \to \infty$ – are known to all the questions Q1–Q4, see Feller [40].

We now discuss how the realizations of S_n look like. Clearly $|S_n| \leq n$. However, large values of $|S_n|$ occur with small probability, "in practice" the values of S_n are in a smaller range. The weak and strong law of large numbers implies that $\frac{S_n}{n} \to 0$ (in probability and even almost surely). It means that deviations of S_n from 0 grow slower than linearly. On the other hand, the CLT (central limit theorem) states that $\frac{S_n}{n} \xrightarrow{D} \mathcal{N}(0,1)$ (where $\xrightarrow{D}$ denotes the convergence in distribution), the fluctuations of S_n will leave interval $[-\sqrt{n}, \sqrt{n}]$, since 0-1 Kolmogorov's Law implies that $\limsup_{n\to\infty} \frac{S_n}{n} = \infty$. The fluctuations can be estimated more accurately. *The law of iterated logarithm* (see [68], cf. also Chapter VIII.5 in [40]) states that for a random walk S_n, we have

$$\begin{aligned} P\left(\liminf_{n\to\infty} \frac{S_n}{\sqrt{2n\log\log n}} = -1\right) &= 1, \\ P\left(\limsup_{n\to\infty} \frac{S_n}{\sqrt{2n\log\log n}} = +1\right) &= 1. \end{aligned}$$

Thus, dividing S_n by n is *too strong* and dividing it by $\sqrt{n}$ is *too weak*. The fluctuations of S_n from 0 grow proportionally to $\sqrt{2n\log\log n}$, see Figure 2.2.

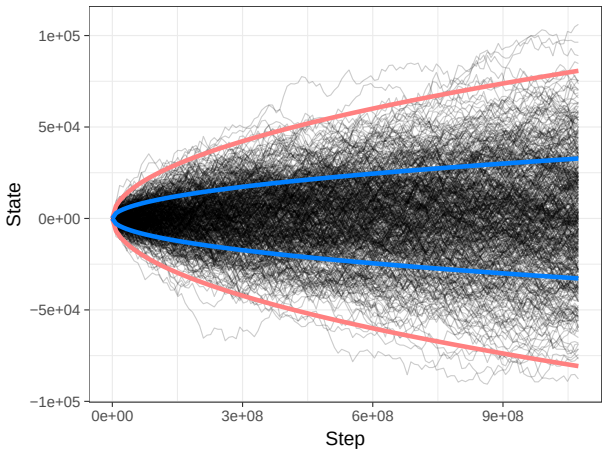


Figure 2.2: 500 trajectories of random walks of length $n = 2^{30}$. Blue plot: $\pm\sqrt{n}$, red plot: $\pm\sqrt{2n \log\log n}$

For example, it is known (see Feller [40], p. 82 or Durrett [34], p. 198) that

$$\mathbb{P}(\alpha \leq L_{2N}^{+}/2N \leq \beta) \to \int_{\alpha}^{\beta} \pi^{-1}(x(1-x))^{-1/2}\, dx, \tag{2.1}$$

for $0 \leq \alpha < \beta \leq 1$. It is called the *arcsine law*. The name comes from the fact that

$$\int_{0}^{t} \pi^{-1}(x(1-x))^{-1/2}\, dx = \frac{2}{\pi}\left(\arcsin\left(\sqrt{t}\right)\right) .$$

A similar result holds for L_{2N}; see Durrett [34] p. 197.

We will try to answer the questions via simulations. Having an algorithm for simulating a random walk, we suggest making histograms for the repetitions of L_{2N}^{+} – the total time of winning of A and L_{2N}. It is also clear that when $N \to \infty$, then of L_{2N}^{+} and L_{2N} also increase to infinity. But as it turns out, time fractions should be considered, i.e., these quantities divided by $2N$. The Monte Carlo method is based on repeating the experiment many times. Let R be the number of experiments performed, called in the Monte Carlo theory *replications*. In Fig 2.3 we have a histogram for L_{1000}^{+} computed from $R = 1000000$ replications.

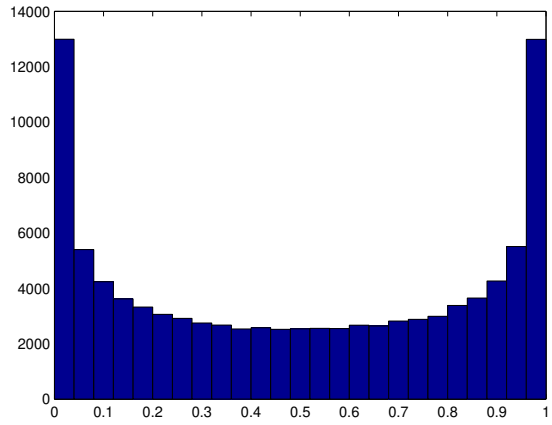


Figure 2.3: Histogram for L^{+}_{1000};R=100000 replications, 25 boxes

2.2 The secretary problem

It is a problem from the area of optimal stopping. The solution is often called "37% rule". It can be expressed in the following way:

- There is one full-time job to fill.
- There are n candidates, the value of n is known.
- The evaluation committee can strictly rank the candidates (no ties).
- Candidates apply in random order. Each order is assumed to be equally probable.
- Following the interview, the candidate is either hired or rejected. The decision cannot be undone afterward.
- The decision is made based on the relative ranking of the candidates met so far.
- The goal is to select the best candidate.

It turns out that the optimal strategy, i.e., the one maximizing the probability of selecting the best candidate, should be searched among the following strategies: leave k of the candidates and then accept the first better candidate than the ones reviewed so far. If there is no such, take the last one. It

turns out that for the optimal strategy, when n is large, we have $k \sim n/e$. We have $1/e \approx 0.37$, hence the name "37% rule".

For small values of n, the optimal k and the corresponding probability of choosing the best candidates are as follows:

n	1	2	3	4	5	6	7	8	9
k	0	0	1	1	2	2	2	3	3
P	1.000	0.500	0.500	0.458	0.433	0.428	0.414	0.410	0.406

his task can be easily simulated if we can generate a random permutation. We can also ask another question. Suppose the candidates have ranks from 1 to n, however, they are not known to the employment committee. As previously, the committee can only sort out the candidates interviewed for the job so far. We can ask whether the optimal choice that maximizes the probability of selecting the best candidate is also maximizing the expected value of the rank of the employed person. This question can be answered by simulation.

2.3 Travelling salesman problem

The salesman leaves city 1 and must visit all the cities $2, \ldots, n$ before returning to 1. The distances between city i and city j are provided in the matrix $\mathbf{M}$ of size $n \times n$. If there is no route from i to j, then $c_{ij} = \infty$. 13 USA cities are presented on the map in Fig. 2.4 – the actual distances are provided in the matrix $\mathbf{M}$ in Eq. (VII.2.4).

Figure 2.4: 13 USA cities. Taken from `https://developers.google.com/optimization/routing/tsp`

For a given permutation σ let $l(\sigma)$ be the total distance travelled:

$$l(\sigma) = \sum_{k=1}^{n-1} \mathbf{M}(\sigma_k, \sigma_{k+1}) + \mathbf{M}(\sigma_n, \sigma_1).$$

The task is to find the permutation σ^* which minimises this distance, i.e.,

$$\sigma^* = \operatorname*{argmin}_{\sigma} l(\sigma).$$

For large n this task is hard to do deterministically. In fact, finding σ^* belongs to the class of optimisation problems known as NP-complete. Roughly speaking, it means there is probably no "fast" (with running time being a polynomial in n) algorithm for finding this optimal solution. However, it turns out that randomised algorithms give unexpectedly good – they find some solution $\hat{\sigma}$ which is close to σ^* in the sense that $l(\hat{\sigma}) \approx l(\sigma^*)$. We will present such optimisation algorithms based on Markov chain Monte Carlo methods in Section **??**.

2.4 Computing integrals with Monte Carlo methods

The Monte Carlo (MC) method is often used to compute integrals. Of course, in non-trivial tasks, these are multidimensional integrals, often over the space not given by a simple formula, which excludes the use of standard numerical methods.

When using the MC method to compute the integral, we use the fact that if $\boldsymbol{U}$ is a random number from $[0,1)^d$ (i.e., the coordinates of the vector $\boldsymbol{U}$ are i.i.d. with the same uniform distribution $\mathcal{U}[0,1)$), then from the strong law of large numbers, with probability 1 we have

$$\frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{U}_i) \to \mathbb{E}f(\boldsymbol{U}) = \int_{[0,1)^d} f(\boldsymbol{u})\, d\boldsymbol{u},$$

provided function f is integrable. It is important to note the speed of this method is of order $O\left(\frac{1}{\sqrt{n}}\right)$ regardless of the dimension of d. Actually, we can say more. Let us start with an arbitrary sequence $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in [0,1)^d$ and $\mathcal{B}$ be a family of subsets from $[0,1)^d$. Then the *discrepancy* of a sequence $\mathbf{x}_1, \mathbf{x}_2, \ldots$ with respect $\mathcal{B}$ is defined as

$$D(\mathbf{x}_1, \ldots, \mathbf{x}_n; \mathcal{B}) = \sup_{A \in \mathcal{B}} \left| \frac{\#\{1 \le i \le n : \mathbf{x}_i \in A\}}{n} - \mathrm{vol}(A) \right|, \tag{2.2}$$

where $\mathrm{vol}(A)$ is the volume of $A \subset \prod_{j=1}^{d}[0,1))$. Suppose that $\mathcal{A}^*$ consists of subsets of form $\prod_{j=1}^{d}[0,u_j)$, where $0 < u_j < 1$, although some other choices of the family of subsets are considered in the literature. If $\mathbf{x}_1, \mathbf{x}_2, \ldots$ is a realization of a sequence of random vectors $\boldsymbol{U}_1, \boldsymbol{U}_2, \ldots$ uniformly distributed $\mathcal{U}(\prod_{j=1}^{d}[0,1))$, then with probability 1

$$\limsup_{n\to\infty} \frac{(2n)^{1/2}}{\log\log n} D(\mathbf{x}_1, \ldots, \mathbf{x}_n; \mathcal{A}) = 1.$$

It means that the discrepancy for a sequence of i.i.d. random vectors $\boldsymbol{U}_1, \boldsymbol{U}_2, \ldots$ behaves as $\log\log n/(2n)^{1/2}$ regardless of the dimension d. It explains formally what it means that the rate of convergence in Monte Carlo methods is of order $1/\sqrt{n}$ (more precisely of order $\mathrm{o}(1/n^{-1/2+\epsilon})$.

3 Information about quasi Monte Carlo method

An alternative method to Monte Carlo (MC), which uses pseudorandom numbers, is the quasi-MC method which uses quasirandom numbers. This method is handy for computing integrals of form $\int_{[0,1)^d} f(\boldsymbol{u})\, d\boldsymbol{u}$. Let us start with the definition of the *low discrepancy sequence* $\mathbf{x}_1, \mathbf{x}_2, \ldots \in [0,1)^d$ if

$$D(\mathbf{x}_1, \ldots, \mathbf{x}_n) = O\left(\frac{(\log n)^d}{n}\right).$$

Instead of generating a sequence of random or pseudorandom numbers the quasi-MC method uses a deterministic sequence $\mathbf{x}_1, \mathbf{x}_2, \ldots$, which has the low discrepancy property and then one computes the integral using the fact that

$$\frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i) \to \mathbb{E} f(\boldsymbol{U}) = \int_{[0,1)^d} f(\boldsymbol{u})\, d\boldsymbol{u}.$$

Sequences with the property of low discrepancy are said to be a quasirandom sequence . Thus for any quasirandom sequence the rate of convergence is of order $O(n^{-(1-\epsilon)})$, where $0 < \epsilon < 1$ is arbitrary (more details below). Of course, we also need some assumptions on function f, which usually holds in practice. Quasi-MC methods are widely used for the valuation of financial products.

In the literature, there are given quasirandom sequences as the Halton sequence, the Sobol sequence or the Faure sequence. For each such sequence of dimension d, it can be shown that

$$D(\mathbf{x}_1, \ldots, \mathbf{x}_n) \leq C_d \frac{(\log n)^d}{n} + O\left(\frac{(\log n)^{d-1}}{n}\right).$$

In `Python`, the procedures for generating Halton and Sobol sequences are available in the library `qmcpy`[1], whereas in `Matlab` they are available in the toolbox *Statistics and machine learning* (`haltonset` and `sobolset`).

Hugo Steinhaus noticed that the so-called golden sequences are useful for computing integrals. By the golden sequence, he meant the sequence $x_n = \{n\alpha\}$, where $\alpha = (\sqrt{5} - 1)/2$ is the golden ratio, i.e., the solution of the golden equation $z^2 + z - 1 = 0$. We denote by $\{x\}$ the fractional part of x. This number has a continued fraction expansion of form

$$\alpha = \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \ldots}}}.$$

The sequence has low discrepancy property. Similarly, it turns out that so-called Kronecker sequences, i.e., sequences $x_n = \{n\beta\}$, where β has a continued fraction expansion of form

$$\beta = \cfrac{1}{b_1 + \cfrac{1}{b_2 + \ldots}},$$

where $b_j \leq M$ is a bounded sequence, are also sequences with low discrepancy. It is unknown whether multidimensional Kronecker sequences of the form $(\{n\beta_1\}, \ldots, \{n\beta_d\})$ have this property, although numerical results suggest, they might have.

Note that we do not study quasirandom numbers in this book; however, we provide one example below.

Example 3.1 (Estimating π). More details on estimating π are provided in subsequent sections. Here we shortly provide one of the simplest methods: we "throw" points chosen "randomly" from $[0, 1)^2$ and compute the fraction

[1] `https://github.com/QMCSoftware/QMCSoftware`, `https://qmcpy.readthedocs.io/`

of points within a quarter of a unit circle. To be more precise, let us take $2R$ i.i.d. numbers $U_{1,1}, U_{1,2}, \ldots, U_{j,1}, U_{j,2}, \ldots, U_{R,1}, U_{R,2}$ and compute the fraction

$$\hat{Y}_R = \frac{1}{R}\sum_{j=1}^{R} Y_j, \quad \text{where } Y_j = 4\mathbf{1}(U_{j,1}^2 + U_{j,2}^2 \leq 1).$$

We have that $\mathbb{E}Y_j = \pi$ for uniformly distributed numbers, thus $\mathbb{E}\hat{Y}_R = \pi$.

We simulated four sets of numbers:

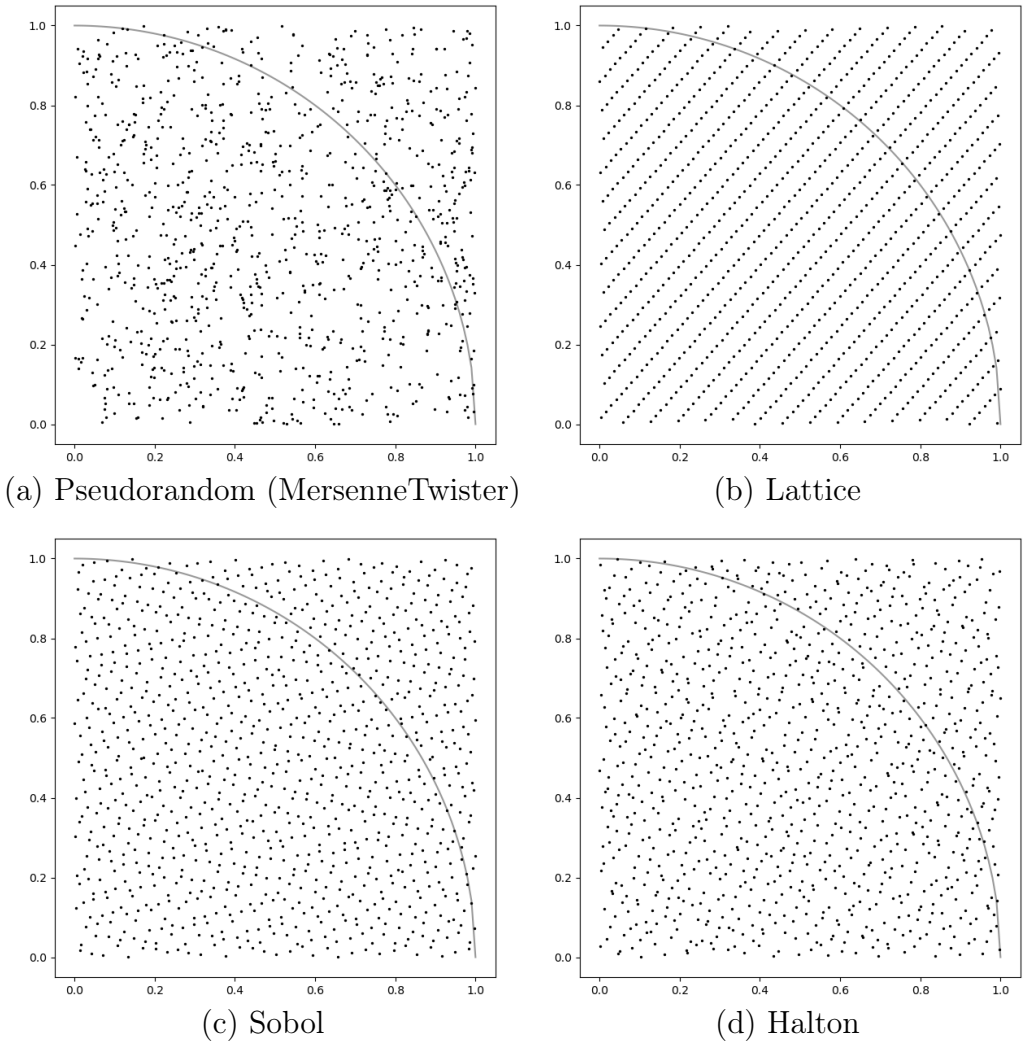
$$U_{1,1}^{\mathtt{A}_i}, U_{1,2}^{\mathtt{A}_i}, \ldots, U_{j,1}^{\mathtt{A}_i}, U_{j,2}^{\mathtt{A}_i}, \ldots, U_{R,1}^{\mathtt{A}_i}, U_{R,2}^{\mathtt{A}_i}, \quad i = 1, 2, 3, 4,$$

where

- $\mathtt{A}_1 = \mathtt{MT19937}$ – pseudorandom numbers using built-in `Python`'s `NumPy` pseudorandom number generator `MT19937`,
- $\mathtt{A}_2 = \mathtt{Lattice}$ – quasirandom numbers using `Lattice` algorithm,
- $\mathtt{A}_3 = \mathtt{Halton}$ – quasirandom numbers using `Halton` algorithm,
- $\mathtt{A}_4 = \mathtt{Sobol}$ – quasirandom numbers using `Sobol` algorithm,

In the last three cases, we used `Python`'s quasirandom numbers library `qmcpy`.

The resulting points for case $R = 1024$ (algorithm `Sobol` requires that the number of points to be generated is a power of 2) are depicted in Fig. 3.5.



(a) Pseudorandom (MersenneTwister) (b) Lattice

(c) Sobol (d) Halton

Figure 3.5: 1024 pseudorandom and quasirandom from $[0,1)^2$.

The results of

$$\hat{Y}_R^{\mathtt{A}_i} = \frac{1}{R}\sum_{j=1}^{R} Y_j^{\mathtt{A}_i} = 3.120, \quad \text{where } Y_j^{\mathtt{A}_i} = 4\mathbf{1}(U_{j,1}^{\mathtt{A}_i})^2 + (U_{j,2}^{\mathtt{A}_i})^2 \leq 1)$$

as well as $|\hat{Y}_R^{\mathtt{A}_i} - \pi|$ (where as π we took its constant value from `NumPy` library) for $R = 256, 512, 1024$ and 2048 are presented in Table 3.1. As we can see, in this example, using quasirandom numbers yields a much better approximation.

R		MT19937	Lattice	Sobol	Halton
2^8	$\hat{Y}_R^{\mathtt{A}}$	3.09375	3.15625	**3.14062**	3.10937
	$\lvert\hat{Y}_R^{\mathtt{A}} - \pi\rvert$	0.04784	0.01465	**0.00967**	0.03222
2^9	$\hat{Y}_R^{\mathtt{A}}$	3.07812	**3.13281**	3.12625	3.16406
	$\lvert\hat{Y}_R^{\mathtt{A}} - \pi\rvert$	0.06346	**0.00878**	0.01465	0.02247
2^{10}	$\hat{Y}_R^{\mathtt{A}}$	3.17187	3.14453	3.14843	**3.14062**
	$\lvert\hat{Y}_R^{\mathtt{A}} - \pi\rvert$	0.03028	0.00294	0.00684	**0.00097**
2^{11}	$\hat{Y}_R^{\mathtt{A}}$	3.06640	3.14453	3.14648	**3.13867**
	$\lvert\hat{Y}_R^{\mathtt{A}} - \pi\rvert$	0.07518	0.00293	0.00489	**0.00292**

Table 3.1: Estimations of π using pseudorandom numbers (MT19937) and quasirandom numbers (Lattice, Sobol and Halton). The best result in each row is bolded.

□

The reader must be warned that quasirandom numbers usually work well for computing integrals, and most often, they fail in other tasks, as seen in the following example.

Remark 3.2 The reader must be warned that quasirandom numbers usually work well for computing integrals, and most often, they fail in other tasks, as seen in the following example.
1. Although the discrepancy of a quasirandom sequence goes to zero faster then for a realization of a uniform i.i.d. sequence, the use for solving probabilistic problems by quasi Monte Carlo methods is restricted mostly to computations of integrals. In Monte Carlo theory for a sequence $U_1, U_2, \ldots$ of i.i.d. uniform random variables, we may consider a sequence of d-vectors $(U_1, \ldots, U_d), (U_{d+1}, \ldots, U_{2d}), \ldots$ as a sequence of random vectors. It is not true for quasirandom numbers. It means that for a low discrepancy sequence $x_1, x_2, \ldots$, the sequence of d-vectors $(x_1, \ldots, x_d), (x_{d+1}, \ldots, x_{2d}), \ldots$ need not to be a low discrepancy one. [2] For each dimension, d, a quasirandom sequence must be constructed separately because grouping into vectors will make the loss of low discrepancy.
2. In Asmussen and Glynn's book ([6], p. 271), there is given a nice

[2] It would be useful to see an example. In the sequel, we show one.

counterexample for the use of the quasi-Monte Carlo method. Thus consider a random walk with an increment of form $X - Y$, where X and Y are exponentially distributed with parameters 1 and $1/2$, respectively. The probability that this random walk exits interval $(-4, 2)$ to the right is $I = (1 - e^2/2)(2e - e^{-2}/2) = 0.174$ in contrast with the result they obtained by quasi Monte Carlo with the use of 4-dimensional Halton that $I = 0.118$.

In the sequel, we present a more extensive experiment showing quasirandom numbers' good and bad properties.

Example 3.3 (Winning probability in a two dimensional game). Consider the following game. We are playing against two other players. Our initial assets are (i_1, i_2) and assets of other players are $(N_1 - i_1, N_2 - i_2)$ (N_j is the total amount of assets with player j). Then, with probability $p_j(i_j)$ we win one dollar with player j and with probability $q_j(i_j)$ we lose it. With the remaining probability $1-(p_1(i_1)+q_1(i_1)+p_2(i_2)+q_2(i_2))$ we do nothing. Once we win completely with player j (i.e., $i_j = N_j$) we do not play with him/her anymore (i.e., $q_j(N_j) = 0 = p_j(N_j)$). We win the whole game if we win with all the players; we lose the whole game if we lose with at least one of the players. Thus, the states of the game are described as $(i_1, i_2), i_j \in \{1, \ldots, N_j\}$ and an extra state **LOSE**. The game is depicted in Fig. 3.6

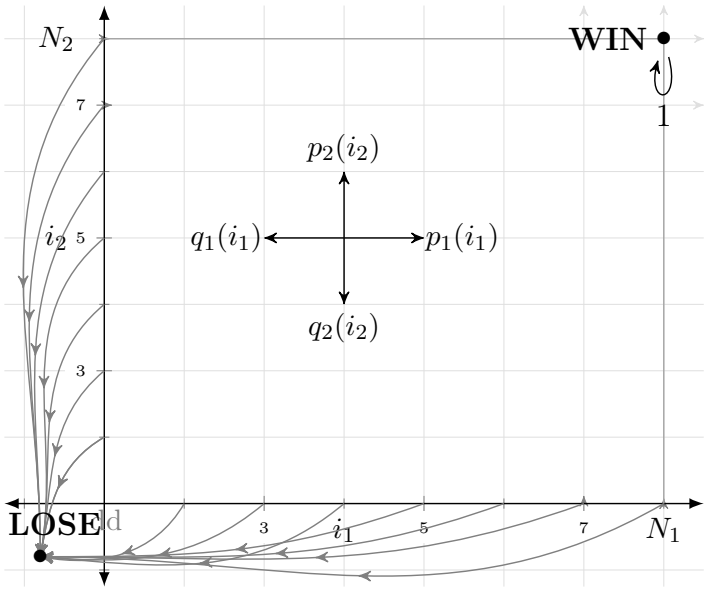


Figure 3.6: A two dimensional game. See description in text.

Let us define

$$\rho((i_1, i_2)) = \mathbb{P}(\text{we win the game } \mid \text{we started in } (i_1, i_2)).$$

For a more formal definition of the game (and more generally, where we play against d other players) see [85] – therein a formula for winning probability is presented, which in our case, is

$$\rho((i_1, i_2)) = \frac{\prod_{j=1}^{2} \left(\sum_{n_j=1}^{i_j} \prod_{r=1}^{n_j-1} \left(\frac{q_j(r)}{p_j(r)} \right) \right)}{\prod_{j=1}^{2} \left(\sum_{n_j=1}^{N_j} \prod_{r=1}^{n_j-1} \left(\frac{q_j(r)}{p_j(r)} \right) \right)}. \tag{3.3}$$

Consider a specific symmetric example with $N_1 = N_2 = 4$ and

$$q_j(i_j) = p_j(i_j) = \frac{1}{4}, j = 1, 2, \ i_j = 1, 2, 3$$

If we start with initial assets $(2, 3)$, the formula (3.3) yields that we will win with probability

$$\rho((2, 3)) = \frac{3}{8} = 0.375.$$

We want to estimate $\rho((2,3))$ via simulations. Again, we will use pseudo-random numbers from Mersenne Twister (denoted by $\mathtt{A}_1$) or quasirandom numbers: `Lattice` (denoted by $\mathtt{A}_2$), `Halton` (denoted by $\mathtt{A}_3$), `Sobol` (denoted by $\mathtt{A}_4$).

We simulate R games, the result of j-th game is denoted as $Y_j^{\mathtt{A}_i} \in \{0,1\}$, where 1 denotes winning and 0 losing. Then, the estimator is

$$\hat{Y}_R^{\mathtt{A}_i} = \frac{1}{R}\sum_{j=1}^{R} Y_j^{\mathtt{A}_i}$$

Say that the current assets are (i_1, i_2). We have to simulate the following moves with the following probabilities:

$$\begin{array}{lcll}
(i_1,i_2) & \to & (i_1+1,i_2) & \text{with prob. } p_1(i_1),\\
(i_1,i_2) & \to & (i_1-1,i_2) & \text{with prob. } q_1(i_1),\\
(i_1,i_2) & \to & (i_1,i_2+1) & \text{with prob. } p_2(i_2),\\
(i_1,i_2) & \to & (i_2,i_2-1) & \text{with prob. } q_2(i_1),\\
(i_1,i_2) & \to & (i_2,i_2) & \text{with prob. } 1-(p_1(i_1)+q_1(i_1)+p_2(i_2)+q_2(i_1)),
\end{array}$$

where if any coordinate of the new state is 0, then we identify it with **LOSE** and if both coordinates are 1 then we identify it with **WIN**. To simulate the move, we could use random variable $U \in [0,1)$ and split the interval $[0,1)$ into five intervals with corresponding lengths, but we want to use two random variables $U_1, U_2 \in [0,1)$ (since in we will use two dimensional quasirandom numbers). That is why first, using U_1 we decide whether we will change the first coordinate or second coordinate, or none. We do it in the following way:

$$\begin{array}{lll}
\text{change 1st coordinate} & \text{if} & U_1 \in [0, p_1(i_1)+q_1(i_1)),\\
\text{change 2nd coordinate} & \text{if} & U_1 \in [p_1(i_1)+q_1(i_1), p_1(i_1)+q_1(i_1)+p_2(i_2)+q_1(i_2)),\\
\text{none} & \text{if} & U_1 \in [1-(p_1(i_1)+q_1(i_1)+p_2(i_2)+q_2(i_1)), 0).
\end{array}$$

If we are to change some coordinates, we proceed as follows using U_2:

If 1st coord.:

$$\begin{aligned}(i_1, i_2) &\to (i_1 + 1, i_2) \quad \text{if} \quad U_2 \in [0, p_1(i_1)/(p_1(i_1) + q_1(i_1))),\\ (i_1, i_2) &\to (i_1 - 1, i_2) \quad \text{if} \quad U_2 \in [p_1(i_1)/(p_1(i_1) + q_1(i_1)), 1).\end{aligned}$$

If 2nd coord.:

$$\begin{aligned}(i_1, i_2) &\to (i_1, i_2 + 1) \quad \text{if} \quad U_2 \in [0, p_2(i_2)/(p_2(i_2) + q_2(i_2))),\\ (i_1, i_2) &\to (i_1, i_2 - 1) \quad \text{if} \quad U_2 \in [p_2(i_2)/(p_2(i_2) + q_2(i_2)), 1).\end{aligned}$$

We used pairs

$$(U_{1,1}^{\mathtt{A}_i}, U_{1,2}^{\mathtt{A}_i}), (U_{2,1}^{\mathtt{A}_i}, U_{2,2}^{\mathtt{A}_i}), \ldots, \quad i = 1, 2, 3, 4.$$

For $\mathtt{A}_1$ (i.e., Mersenne Twister), we took consecutive outputs from the PRNG, whereas in the remaining cases as pairs $(U_{j,1}^{\mathtt{A}_i}, U_{j,2}^{\mathtt{A}_i})$ we used two dimensional quasirandom numbers. In each case, we simulated $R = 4000$ games (note that each game required a random number of pairs – game was simulated till winning or losing).

As we can see in Fig. 3.7, in case of quasirandom numbers, the estimators converge to completely different numbers. The `Sobol` numbers yielded worse results. We took every second point, since using all the points, the game never ended (in a reasonable time). Even then, all the games ended in **WIN** always in 4 steps.

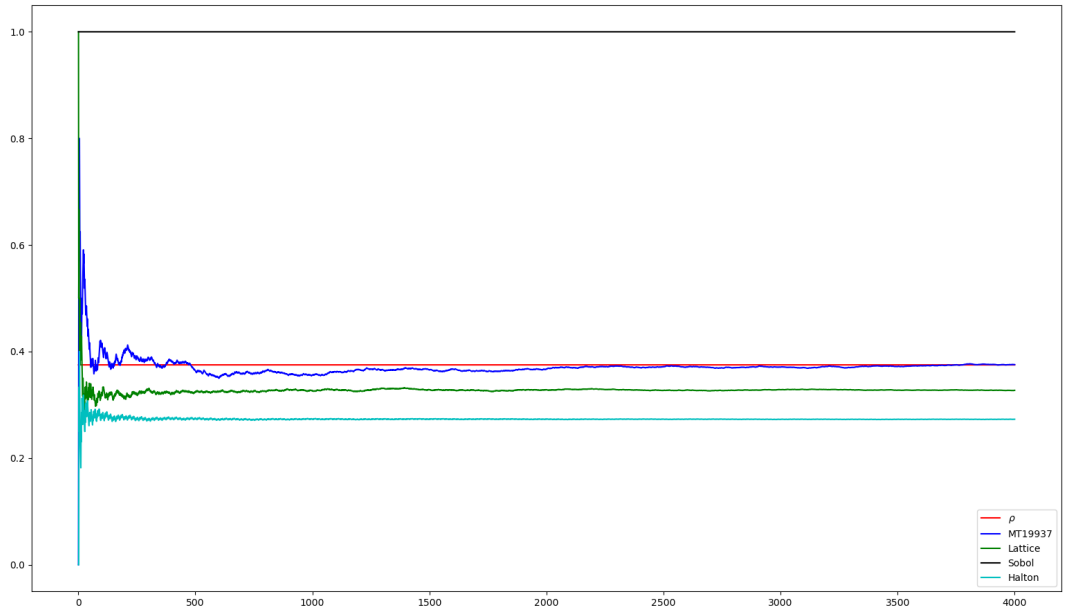


Figure 3.7: Estimating $\rho((2,3)) = 0.375$ using pseudorandom numbers (Mersenne Twister) and quasirandom numbers (`Lattice, Halton, Sobol`) – steps from 1 to $R = 40000$. Red line – actual value of $\rho((2,3))$.

In Table 3.2, the values of estimators as well as average the number of steps of each game is presented.

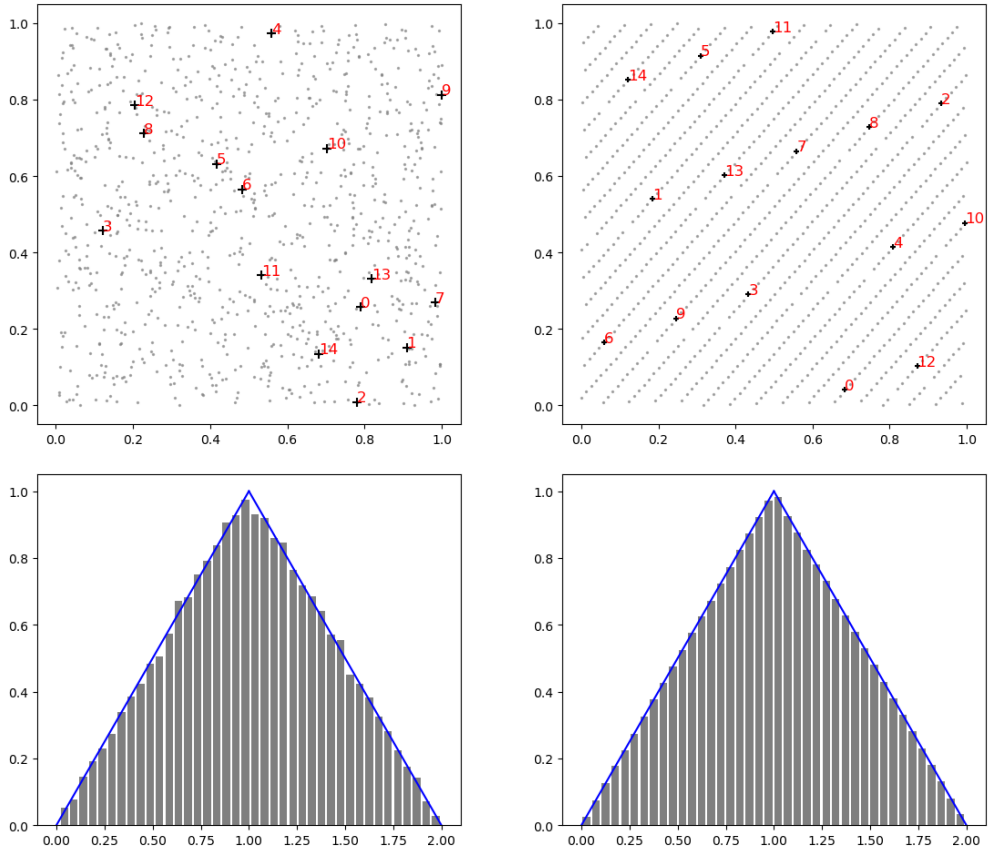
	`MT19937`	`Lattice`	`Sobol`	`Halton`
$\hat{Y}_R^{\mathbf{A}}$	0.3775	0.32725	1.0000	0.2730
Avg nr of steps	8.7715	9.0355	4.0000	9.8145

Table 3.2: Estimating $\rho((2,3)) = 0.375$ using pseudorandom numbers (Mersenne Twister) and quasirandom numbers (`Lattice, Halton, Sobol`). Values of $\hat{Y}_R^{\mathbf{A}_i}$ estimators $i = 1, 2, 3, 4$ for $R = 40000$ and average number of steps of a single game.

In Figures 3.8 and 3.9 we presented:

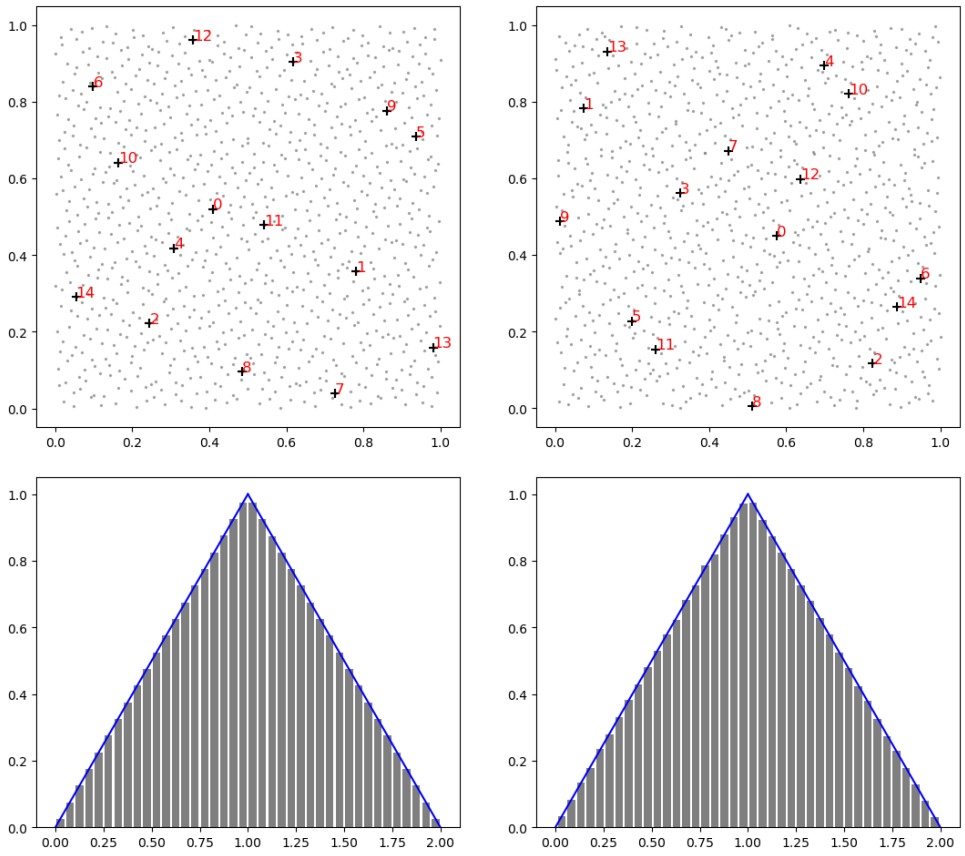
- Top rows: first 1000 points used in simulations, 15 of which are numbered.

- Bottom rows: histograms of sums $U_{k,1}^{A_i} + U_{k,2}^{A_i}$. Blue line – density f of a sum of two independent $\mathcal{U}[0,1)$ random variables.



(a) Pseudorandom (MersenneTwister) (b) Lattice

Figure 3.8: Mersenne Twister and Lattice. Top row: 1000 points used in simulations, first 15 points numbered. Bottom row: histogram of $U_{k,1}^{A_i} + U_{k,2}^{A_i}$. Blue line – density f of a sum of two independent $\mathcal{U}[0,1)$ random variables given in (3.4).



(c) Sobol (d) Halton

Figure 3.9: Sobol and Halton. Top row: 1000 points used in simulations, first 15 points numbered. Bottom row: histogram of $U_{k,1}^{A_i} + U_{k,2}^{A_i}$. Blue line – density f of a sum of two independent $\mathcal{U}[0,1)$ random variables given in (3.4).

As we can see, there is a linear dependence between points. For example, in Halton sequence points $(U_{3,1}^{\mathtt{A}_4}, U_{3,2}^{\mathtt{A}_4})$, $(U_{4,1}^{\mathtt{A}_4}, U_{4,2}^{\mathtt{A}_4})$ and $(U_{7,1}^{\mathtt{A}_4}, U_{7,2}^{\mathtt{A}_4})$ lie on one line. In the case of `Lattice` sequence , the linear dependence is clearer.

A distribution of a sum U_1+U_2 of two independent $\mathcal{U}[0,1)$ random variable

has the symmetric triangular distribution

$$f(x) = \begin{cases} x & \text{if } x \in [0,1), \\ 2-x & \text{if } x \in [1,2). \\ 0 & \text{otherwise.} \end{cases} \tag{3.4}$$

In bottom rows of Figures 3.8 and 3.9 the histogram of all points used in simulations is depicted. As we can see, the histograms for quasirandom numbers are *"too perfect"*. □

4 Bibliographic notes

The reader can find some aspects of the theory of simulation in the Knuth [71] pioneering monograph on the art of computer programming. There are many good textbooks on stochastic simulation methods and Monte Carlo theory on the market. Let us start with an elementary textbook by Ross' [116] book. More advances are Ripley [110] and Madras [88]. Among very good books using the thoroughly modern probabilistic apparatus and the theory of stochastic processes, one can mention the book by Asmussen and Glynn [6], Fishman [42] and Rubinstein and Kroese [119]. Among the books on simulations in specific fields, one can mention Glasserman's book [48] (financial engineering), Fishman [43] (discrete event simulations). A quasi MC method is described in books: Niederreiter [24] and Glasserman [48]. Steinhaus's observation on the low discrepancy of the golden sequence is from [124]. The law of iterated logarithm for discrepancy sequence of i.i.d. random vectors was proven by Kiefer [69]. [3] Subsection II.2.2 is based on the paper by Diaconis [29]. Gina Kolata is the author of a New York Times article [72]. The book by Levin *et al* [81] is devoted to this topic and the convergence speed to the distribution of stationary Markov chains in general.

[3] Kiefer On large deviations of the empirical distribution function. Pacific J. Math. (1961) 649–660.

Chapter II

The theory of generators

1 Introduction

The foundation of stochastic simulation lies in the concept of a random number. Although we will not engage in detailed philosophical discussions regarding the meaning of "random," it is instructive to consider Knuth's perspective as expressed in his seminal work [71], page 2:

> "In a sense, there is no such thing as a random number; for example, is two a random number?"

However, we can rigorously define an *independent sequence of random variables* with a specified *distribution*. This essentially means that each number in the sequence is selected entirely at random, without any dependencies on the selection of other numbers, and that each number conforms to a specified probability distribution.

In this text, we will use the terms *random number* and *random variable* interchangeably.

The primary distribution in this theory is the uniform distribution $\mathcal{U}[0,1)$.

It is important to note that a computer cannot generate all possible numbers within the interval $[0,1)$. Instead, the numbers generated are of the form

$$0, 1/M, 2/M, \ldots, (M-1)/M$$

for some fixed $M = 2^k$. We assume that these numbers are uniformly distributed within this discrete set. In this case. throughout this text, we may omit the explicit mention of the distribution name. For instance, we might

refer to a random variable U with the distribution $\mathcal{U}[0,1)$ simply as a random number U. Symbols representing distributions, along with their definitions and key characteristics, are provided in the Appendix I.2.

Before the advent of the computer age, tables of random numbers were created. The most straightforward table can be made, for example, by drawing balls from a box, throwing a dice, etc., then recording these results. You can also use a 'dice' that gives results from 0 to 9; see Fig. 1.1, and produce through consecutive throws *random digits*.

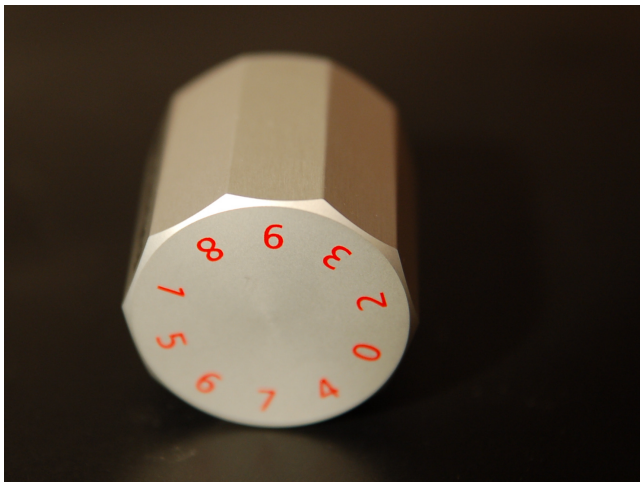


Figure 1.1: Eurandom dice

Then, by grouping a fixed number of digits, preceded by zeros and periods, we obtain the desired table of random numbers. Such tables often have "good properties" (what we understand by this we will clarify later) but when lots of random numbers are needed (e.g., hundreds of thousands) the method is useless.

Today computers are used to create a sequence of random numbers (or, as it is often said, *to generate* them). Probably the first to do so was John von Neumann around 1946. He proposed to create the next number by squaring the previous number and cutting out the middle numbers. "Middle-square method" turned out to be not very good; it has a rutting tendency - the short cycle of repetitions items. For example, a sequence of two-digit numbers starting with 43 according to this algorithm is (we always make the resulting number 4 digits long, padding zeros if needed):

$84, 05, 25, 62, 84, 05, 02, 00, 00, \ldots$. Namely, we have $\mathbf{43}^2 = 1\underline{\mathbf{84}}9$, $\mathbf{84}^2 = 7\underline{\mathbf{05}}6$, $\mathbf{05}^2 = 0\underline{\mathbf{02}}5, \mathbf{02}^2 = 0\underline{\mathbf{00}}4, \mathbf{00}^2 = 0\underline{\mathbf{00}}0$.

A comprehensive theory has emerged regarding generators, i.e., algorithms producing sequences of "random numbers", and the generator's "goodness" analysis. This theory uses the achievements of modern algebra and statistics.

Unfortunately, the computer cannot generate the perfect sequence of independently uniformly distributed random variables. Therefore, it is sometimes referred to as a computer-generated pseudorandom number. Understanding these limitations, we will mainly write about the sequence of random variables with the uniform distribution.

Definition 1.1 A pseudorandom number generator (PRNG) is a 5-tuple (S, s_0, f, V, g), where

- S is a finite state space,
- $s_0 \in S$ is the initial value of the recurrence (*seed*),
- $s_{i+1} = f(s_i)$, where $f : S \to S$,
- V is a finite set of values, and
- $g : S \to V$ (maps the generator's state into the output) and $u_i = g(s_i)$ $(i = 0, 1, \ldots)$ is a generated sequence of pseudorandom numbers.

It is easy to notice that the sequence generated by $s_{n+1} = f(s_n)$ will always fall into a loop: there is a cycle that keeps repeating. The length of such a cycle is called a *period*. The "goodness" of the generator depends on the period. We care about very long periods. Note that theoretically, the cycle cannot be longer than the power of set S.

1.1 Formal definition of a sequence of random numbers; pseudorandom numbers

By a *sequence of random numbers* we mean the sequence of i.i.d. random variables with the same uniform distribution. We will consider (and exhaustively use) the random variables on three types of state spaces: on $[0, 1)$, on $\{0, 1\}$ or on $\bar{M} = \{0, 1, \ldots, M-1\}$ for some natural $M \geq 2$. Respectively, we will talk about random variables U_j, B_j or Y_j, $j = 1, 2, \ldots$
We now recall formal mathematical definition.

- By a *sequence of random numbers* from [0,1) we understand a sequence $U_1, U_2, \ldots (U_j : (\Omega, \mathcal{F}, \mathbb{P}) \to [0,1])$ such that

  i) Each U_j has the uniform distribution, i.e.,

$$\mathbb{P}(U_j \le t) = \begin{cases} 0 & t < 0, \\ t & 0 \le t < 1, \\ 1 & t \ge 1, \end{cases}$$

  ii) $U_1, U_2, \ldots$ are independent identically distributed (i.i.d.), i.e., (for $n = 1, 2, \ldots$)

$$\mathbb{P}(U_1 \le t_1, \ldots, U_n \le t_n) = t_1 t_2 \cdots t_n,$$

  for any $0 \le t_i \le 1, i = 1, \ldots, n$.

- By a *sequence of random bits* we understand a sequence $B_1, B_2, \ldots (B_j : (\Omega, \mathcal{F}, \mathbb{P}) \to \{0,1\})$ such that

  i) $\mathbb{P}(B_j = 0) = \mathbb{P}(B_j = 1) = 1/2$ for each j,

  ii) random variables $B_1, B_2, \ldots$ are i.i.d., i.e., (for $n = 1, 2, \ldots$)

$$P(B_1 = b_1, B_2 = b_2, \ldots, B_n = b_n) = \frac{1}{2^n}.$$

- By a *sequence of random numbers* from $\{0, 1, \ldots, M-1\}$ we understand a sequence $Y_1, Y_2, \ldots (Y_j : (\Omega, \mathcal{F}, \mathbb{P}) \to \{0, 1, \ldots, M-1\})$ such that

  i) $\mathbb{P}(Y_j = i) = \frac{1}{M}, \quad i = 0, \ldots, M-1$, for each j,

  ii) random variables $Y_1, Y_2, \ldots$ are i.i.d., i.e., (for $n = 1, 2, \ldots$)

$$P(Y_1 = y_1, Y_2 = y_2, \ldots, Y_n = y_n) = \frac{1}{M^n}.$$

Note an important and non-trivial property of such sequences of random numbers. If $n_1 < n_2 < \ldots$ is a natural subsequence of natural numbers, then $U_{n_1}, U_{n_2}, \ldots$ is a sequence of random numbers. The same is true for a random sequence of bits.

The subject of further consideration in the book will be a sequence of pseudorandom numbers (bits), i.e., those that imitate sequences of random

numbers (bits). Such sequences can be obtained in various ways, but we will deal with those generated by appropriate algorithms on computers. As mentioned, these are called pseudorandom number generators (PRNGs). We will often omit the word *pseudo*. In literature, it is often written simply *random number generator* (RNG), which, of course, must be a *pseudorandom* number generator. Humans are not able to generate a perfect sequence of numbers, which is a realization of i.i.d. random variables.

Continuing our terminological considerations, note that when referring to *R replications* of a random number U, then if nothing else is said, we have a sequence $U_1, \ldots, U_R$ of i.i.d. random variables with the distribution $\mathcal{U}[0, 1)$.

2 Random permutations

Let $\mathcal{S}_n$ be the set of all permutations of $[n] = \{1, \ldots, n\}$. By a random permutation, we understand a random element taking all values in $\mathcal{S}_n$ with equal probability $1/n!$. In other words, it is a random element on $\mathcal{S}_n$ with distribution $\mathcal{U}[\mathcal{S}_n)$.

2.1 An algorithm for a random permutation

We will now provide an algorithm for generating a random permutation of $\{1, \ldots, n\}$, i.e., such that each result can be obtained with probability $1/n!$. We assume that we have a random number generator at our disposal, i.e., $U_1, U_2, \ldots$, a sequence of i.i.d. random variables with the uniform distribution $\mathcal{U}[0, 1)$. Then, the procedure Fisher-Yates$(n, (U_1, \ldots, U_n))$ returns (uniformly) random permutation of a set $\{1, \ldots, n\}$.

Algorithm 1 Fisher-Yates (Random permutation)

Input: Integer n; sequence of PRNs $\boldsymbol{u} = (u_1, \ldots, u_n), u_i \in [0, 1]$
Output: Permutation of n elements

1: $S[i] = i, i = 1, \ldots, n$
2: **for** $i = n$ downto 1 **do**
3: $\quad k = 1 + \lfloor i u_i \rfloor$
4: $\quad$ **Swap**$(S[k], S[i])$
5: **end for**
6: **Return** S

The algorithm has complexity $\mathrm{O}(n)$ (why ?). The algorithm is a consequence of Exercise II.T.8.

Algorithm 2 is another simple procedure for generating a random permutation (by calling Perm-sort$(n, (U_1, \ldots, U_n))$.

Algorithm 2 Perm-sort (Generating random permutation by sorting)

Input: Integer n; sequence of PRNs $\boldsymbol{u} = (u_1, \ldots, u_n), u_i \in [0, 1)$
Output: Permutation of n elements
1: Sort $u_1, \ldots, u_n$ in increasing order $u_{S[1]} \leq u_{S[2]} \leq \cdots \leq u_{S[n]}$
2: **Return** S

2.2 Shuffling cards

Shuffling cards is a procedure that leads to a random permutation. Assume we have a deck of 52 cards. We want to arrange them *randomly*, i.e., so that each configuration has probability 1/52!. Of course, we need to define what shuffling is formally. Many articles on the theoretical aspects of card shuffling have been published in recent years. A deck of n cards can be identified with numbers $[n] = \{1, \ldots, n\}$, which can be arranged in $n!$ ways, arrangements being named *permutations*. By a random permutation, we will call a random element $X : \Omega \to \mathcal{S}_n$ on a probabilistic state space $(\Omega, \mathcal{F}, \mathbb{P})$ such that $\mathbb{P}(X = \sigma) = 1/n!$ for any permutation $\sigma \in \mathcal{S}_n$. A card shuffling scheme is defined by its *one step*. For example, a Top-To-Random card shuffling is the following scheme: take a card from the top and put it randomly into a deck (more detailed description below). Let X_0 be an initial permutation (say, identity). Then X_k is a permutation obtained after k steps of a card shuffling scheme. For reasonably defined shuffling, it is easy to see that the distribution of X_k converges, as $k \to \infty$, to the uniform distribution on $\mathcal{S}_n$ (using some simple facts from the theory of Markov chains; details can be found in Section VI.2 and Exercise VI.T.5). An important and often non-trivial problem is how far a distribution of X_k from a uniform distribution is. We can measure it e.g., using the total variation distance

$$d_k = \frac{1}{2} \sum_{\sigma \in \mathcal{S}_n} \left| \mathbb{P}(X_k = \sigma) - \frac{1}{n!} \right|.$$

Or similarly, given a card shuffling scheme, how many steps k are needed so that the total variation distance is small? It is expressed by the so-called *mixing time*

$$\tau^{mix}(\varepsilon) = \inf_{k\geq 0} \{d_k \leq \varepsilon\}.$$

We can come up with different shuffling schemes.

Top-To-Random In each step, we place the top card randomly into the deck. More formally, let us denote places between cards $k_1 \sqcup k_2 \sqcup \cdots k_n \sqcup$. We then take k_1 and put it into any of these places with probability $1/n$.

Random-To-Random transposition Given a permutation $\sigma = (\sigma_1, \ldots, \sigma_n)$ we choose two cards (or positions, the distribution is the same) independently at random (i.e., each one with probability $1/n$) and swap them if they are different (if they are the same, we do nothing).

Riffle shuffle and its time reversal Riffle shuffle is one of the most popular shuffling schemes (it is believed that this is the best model describing real shuffling performed, e.g., in casinos). Assume we have n cards. We split them into two piles in the following way: we randomly choose Y according to a binomial distribution $\mathrm{Bin}(n, 1/2)$ (thus, most often, the piles have a similar number of cards). Assume $Y = k$, then the first pile (say kept in the *left* hand) consists of top k cards, and the second pile (kept in the *right* hand) consists of the remaining $n - k$ cards (we do not change their relative ordering while splitting). Having the two piles in both hands we perform the following shuffling: we drop a card, one by one, from each hand with probability proportional to the number of cards. Assume that there are L cards left in the left hand and R cards in the right hand. Then we drop the card from the left hand with probability $L/(L + R)$, and with probability $R/(L + R)$ we drop it from the right hand. We perform it until all cards are dropped. We described one step of the procedure. We repeat it several times. One of the interesting questions is how many steps are needed to be "close" to a uniform permutation. For a Riffle Shuffle with $n = 52$ cards, as it is in a typical deck of cards, exact values of d_k are following:

k	1	2	3	4	5	6	7	8	9	10
d_k	1.000	1.000	1.000	1.000	.924	.614	.334	.167	.085	.043

Table 2.1: Total variation distance between the distribution of a deck after t shuffles and a uniform distribution on $\mathcal{S}_{52}$

(taken from Diaconis [29]). These results suggest that $k = 7$ steps should be enough to obtain a distribution close to random (i.e., well shuffled deck of cards). It was the content of the article in the New York Times ([72] titled *"In shuffling cards, 7 is winning number"*). Table 2.1 shows one problem that we often encounter when we simulate the characteristics of a process, which stabilises, and we want to calculate the quantity under stable conditions. Computing numbers which are in the table would be hard for $n = 52$, but we may try to compute d_k via simulations for a smaller number of cards.

Time reversed Riffle Shuffle. One step of the procedure is the following: Given a deck of n cards, we assign randomly (and independently) bits 0 or 1 to each card. Then we put all the cards with assigned bit 0 to the top *keeping* their relative ordering.

It turns out (see Exercise II.T.19), that the above procedure is a *time reversal of Riffle Shuffle* in the following sense: the probability that Riffle Shuffle transforms a permutation σ into σ' is equal to the probability that time reversed Riffle Shuffle transforms σ' into σ. In particular (proof requires some Markov chain tools), the total variation distance between the distribution of a deck after k time reversed Riffle Shuffles and the uniform distribution on $\mathcal{S}_n$ is the same as the one for Riffle shuffle (thus, for $n = 52$ the values of d_k in Table 2.1 are also valid for time reversed Riffle Shuffle). Note that the latter procedure is simpler to simulate on a computer and may be simpler for an analysis – using relatively simple *strong stationary time* argument it can be shown that $2\log_2 n$ steps are enough for (time-reversed) Riffle Shuffle to mix. Bayer and Diaconis [12] showed that "actual" number of steps is $\frac{3}{2}\log_2 n$ (there is a so-called *cutoff phenomenon* – performing more steps makes d_k close to 0, whereas performing fewer steps makes d_k close to 1).

A common characteristic of these shuffling schemes is that, after mathematical formalization, they align with Markov chain theory. The observation that shuffles converge to a random permutation follows as a straightforward consequence of the fundamental theorems of this theory. However, determin-

ing the rate of this convergence is a non-trivial matter, and current research is directed towards elucidating this aspect.

3 Pseudorandom number generators

3.1 Congruent generators

The simplest example of a pseudorandom number generator is a sequence u_n given by x_n/M, where

$$x_{n+1} = (ax_n + c) \mod M. \tag{3.1}$$

We need to choose

M,	a modulus;	$0 < M$,
a,	a multiplier;	$0 \leq a < M$,
c,	an increment	$0 \leq c < M$,
x_0,	an initial vale (seed);	$0 \leq x_0 < M$.

To obtain a sequence from the interval $[0, 1)$, we take $u_n = x_n/M$. A sequence (x_n) has a period not longer than M. This method of obtaining pseudorandom numbers is called a *linear congruence method*.

The following theorem (see Theorem A on page 17 in Knuth [71]) gives the conditions for a linear congruence to have a period of length M.

Theorem 3.1 *Let $b = a - 1$. For a sequence (x_n) given by (3.1) to have a period of length M it is necessary and sufficient that:*

- *c is coprime with M,*
- *b is a multiplicity of p for every prime number p dividing M,*
- *if 4 is a divisor of M then $a = 1 \mod 4$.*

We now give examples of pseudorandom number generators obtained by the linear congruence method.

Example 3.2 One may take $M = 2^{31} - 1 = 2147483647$, $a = 7^5 = 16867$, $c = 0$. This choice was popular on 32-bit computers. Another example of parameters having good properties is: $M = 2147483563$ and $a = 40014$. For 36-bit computers choosing $M = 2^{35} - 31$, $a = 5^5$ turns out to be good. □

Example 3.3 [Resing [109]]. If $(a, c, M) = (1, 5, 13)$ and $X_0 = 1$, then the sequence (x_n) is as follows: 1,6,11,3,8,0,5,10,2,7,12,4,9,1,..., which has a period 13. If $(a, c, M) = (2, 5, 13)$ and $x_0 = 1$, then we will obtain a sequence $1, 7, 6, 4, 0, 5, 2, 9, 10, 12, 3, 11, 1, \ldots$ with period 12. If, however $x_0 = 8$, then all the sequence elements are equal to 8 (thus, the period is 1). □

If $c = 0$ in (3.1), then the generator is called a *linear multiplicative congruential generator*. In Lewis et al [82] the following parameters were suggested: $M = 2^{31} - 1$, $a = 1680$.

Generalized method of linear congruence generates sequences using a recurrence:

$$x_n = (a_1 x_{n-1} + a_2 x_{n-2} + \ldots + a_k x_{n-k} + c) \mod M.$$

If M is a prime number, then an appropriate choice of a_j constants allows you to get period M^k. In practice, $M = 2^{31} - 1$ is chosen on 32-bit computers.

Terminology and abbreviations. Generators based on the linear congruence method are termed linear congruential generators, denoted by LCG (M, a, c). In the case of generators employing the generalized linear congruential method, they will be denoted by GLCG $(M, (a_1, \ldots, a_k), c)$.

Following L'Ecuyer [77] we now list some popular generators used in commercial packages.

Java

$$\begin{aligned} x_{i+1} &= (25214903917 x_i + 11) \mod 2^{48} \\ u_i &= (2^{27} \lfloor x_{2i}/2^{22} \rfloor + \lfloor x_{2i+1}/2^{21} \rfloor)/2^{53} . \end{aligned}$$

The was a UNIX PRNG called rand48 with the same recurence for x_i and setting $u_i = x_i/2^{48}$.

Visual Basic

$$\begin{aligned} x_i &= (1140671485 x_{i-1} + 12820163) \mod 2^{24} \\ u_i &= x_i/2^{24} . \end{aligned}$$

Excel

$$u_i = (0.9821u_{i-1} + 0.211327) \mod 1 \ .$$

There are also generators based on higher order congruences, e.g. square or cubic. Examples of such generators can be found in Rukhin *et al.* [120].

Another class of congruential generators are quadratic ones, fulfilling a recurrence $x_{i+1} = ax_i^2 + bx_i + c \mod M$, where a, b, c are non-negative integers from $\{0, 1, \ldots, M-1\}$

Quadratic congruential generator I (QCG I). A recurrence gives the sequence

$$x_{i+1} = x_i^2 \mod M,$$

where M is 512-bit prime number specified in hexadecimal as

M = `987b6a6bf2c56a97291c445409920032499f9ee7ad128301b5d0254aa1a9633`
`fdbd378d40149f1e23a13849f3d45992f5c4c6b7104099bc301f6005f9d8115e1`

Quadratic congruential generator II (QCG II). The sequence (x_k) is given by recurrence

$$x_{i+1} = 2x_i^2 + 3x_i + 1 \mod 2^{512}.$$

Cubic congruential generator (CCG). The sequence (x_k) is given by recurrence

$$x_{i+1} = 2x_i^3 \mod 2^{512}.$$

Fibonacci generator. A recurrence gives the sequence

$$x_n = x_{n-1} + x_{n-2} \mod M \, (n \geq 2).$$

Subtractive lagged Fibonacci generator. A modification resulting in a better "independence" property is the so-called lagged Fibonacci generator given by a recurrence

$$x_n = x_{n-k} + x_{n-l} \mod M \, (n \geq \max(k, l)).$$

Knuth proposed this generator with $k = 100$, $l = 37$ and $M = 2^{30}$.

Many problems require very long periods of PRNGs. Such are provided, for example, by the so-called mixed generators.

L'Ecuyer mixed generator For example, let

1. $x_n = (A_1 x_{n-2} - A_2 x_{n-3}) \mod M_1$,
2. $y_n = (B_1 y_{n-1} - B_2 y_{n-3}) \mod M_2$,
3. $z_n = (x_n - y_n) \mod M_1$,
4. if $z_n > 0$, then $u_n = z_n/(M_1 + 1)$; otherwise $u_n = M_1/(M_1 + 1)$.

with a choice $M_1 = 4\,294\,967\,087$, $M_2 = 4\,294\,944\,443$, $A_1 = 1\,403\,580$, $A_2 = 810\,728$, $B_1 = 527\,612$, $B_2 = 1\,370\,589$. Some of the three first x's and y's should be taken as initial values. L'Ecuyer created the algorithm [76] with parameters adjusted by a computer assuring optimal properties of the generator.

Remark 3.4 The Mersenne-twister generator has a reputation for being a very good one. The name comes from the so-called Mersenne numbers of the form $2^p - 1$, where p is a prime number. The most commonly used version of the Mersenne Twister algorithm is based on the Mersenne prime $2^{19937} - 1$. Its period is $2^{19937} - 1$. There are implementations for 32-bit under the name `MT19937-32` or for 64-bit `MT19937-64`. They generate different sequences. On concrete implementations, we will write in a moment below.

3.2 Generators in Python and Matlab

We extensively use `Python`[1], in particular a library `NumPy`[2] (which stands for `Numerical Python`) for generating random numbers. In some cases, we use Matlab.[3]

Python. First, we need to import the `NumPy` library

```
import numpy as np
```

`NumPy` contains 4 PRNGs: `MT19937` (Mersenne Twister 19937), `PCG64` (Permuted Congruential Generator, period 2^{128}, default) , `Philox` (a counber-based PRNG, period $2^{256} - 1$) and `SFC64` (Small Fast Chaotic PRNG, period depends on seed, on average of length 2^{255}).

[1] Version 3.9.2
[2] Version 1.19.4
[3] Version 7.2.0.294 (R2006a).

A good practice is to create a new RNG object and pass it around, invoking `np.random.default_rng(np.random.`method`(seed=`s`))`, where

- method $\in$ {`PCG64, MT19937, Philox, SFC64`},
- a value of s depends on the method:

`PCG64` s can be an integer between 0 and 2^{128}; an array of such integers or `None` (default). If s is `None` then unsigned integers are read from `/dev/urandom` (or the Windows analogue) if available or seed is constructed from the current time otherwise.

`MT19937` s can be an integer between 0 and $2^{32}-1$; an array of such integers or `None` (default). If s is `None` then 624 32-bit unsigned integers are read from `/dev/urandom` (or the Windows analog) if available or seed is constructed from the current time otherwise.

`Philox` s can be an integer between 0 and $2^{64}-1$; an array of such integers or `None` (default). If s is `None` then 624 32-bit unsigned integers are read from `/dev/urandom` (or the Windows analog) if available or seed is constructed from the current time otherwise

`SFC64` Four unsigned 64-bit numbers.

For example, to generate numbers using `MT19937` and `PCG64`, with seeds being equal to 10 in both cases we should run

```
import numpy as np
from numpy.random import MT19937, PCG64
rng_mt19937 = ...
   np.random.default_rng(np.random.MT19937(seed=10))
rng_pcg64 = np.random.default_rng(np.random.PCG64(seed=10))
rng_mt19937.random() # generates u from [0,1)
rng_pcg64.random()   # generates u from 00,1)
```

Matlab. In earlier versions of Matlab, three generators were implemented: '`state`', '`seed`' and '`twister`', the first one being the default one. To set up a generator, one had to envoke a command `rand`(method,`s`), where method $\in$ {state, seed, twister}. In newer versions of Matlab the syntax was changed[4] to `rand`(`s`, method), where

[4] https://www.mathworks.com/help/matlab/math/updating-your-random-number-generator-syntax.html

- method $\in$ {twister, simdTwister, combRecursive, multFibonacci, philox, threefry},
- `s` can be either a positive integer (range depends on the PRNG) or `'default'`.

Example 3.5 We propose to do the following experiments in `Python` and/or Matlab to start with.

- `Python`:

```
import numpy as np
from numpy.random import MT19937, PCG64
for i in range(2):
    rng_mt = np.random.default_rng(MT19937(seed=10))
    rng_pcg = np.random.default_rng(PCG64(seed=10))
    print(rng_mt.random()) # generates u from [0,1)
    print(rng_pcg.random())   # generates u from [0,1)
```

- Matlab: After a new call to Matlab invoke commands:

```
rng(1,'philox')
rand(1,5)
# numbers 0.5361 0.2319 0.7753 0.2390 0.0036 should ...
   be obtained
rng(0,'twister')
rand(1,5)
# numbers 0.8147 0.9058 0.1270 0.9134 0.6324 should ...
   be obtained
```

□

The Mersenne Twister generator. One of the generators utilized by `Python` and Matlab (also implemented in many other packages) is the Mersenne Twister `MT19937` generator, which produces 32-bit or 64-bit sequences of integers. It is characterized by a very long period of $2^{19937} - 1$. This period is a prime number known as a Mersenne prime number. For many applications, an even shorter period is sufficient. The generator requires a significant amount of CPU space and is relatively slow. Another issue is the selection of the seed, which should be highly irregular. If the seed is too regular, obtaining results that closely mimic a random sequence of bits would require many

iterations. The Mersenne Twister was the default PRNG in `NumPy` versions prior to 1.17; afterwards it was changed to `PCG64`, which was shown to be "statistically" better in addition to being more computationally efficient.

3.3 Cryptographic pseudorandom number generators

We will not delve into all the details of cryptographic PRNGs here nor present many such generators. Our primary objective is to define such a PRNG (slightly different from Definition 1.1) and briefly introduce the RC4 scheme. The RC4 scheme is no longer considered a cryptographic PRNG. One of the weaknesses is related to card shuffling and the rate of convergence of Markov chains to the stationary distribution — topics within the scope of this book. Hence, we will provide more details on this algorithm.

There are mainly two differences between a traditional and cryptographic definition of a PRNG:

- The functions f and g (present in Definition 1.1) need to be computable in polynomial time with respect to the input size – in other words, they need to be *efficiently computable*.
- Only PRNGs for which no weaknesses were found via statistical tests can still be called cryptographic PRNGs.

Statistical tests running in polynomial (input size) time are called *polynomial time distinguishers*. Such a distinguisher may be *probabilistic*, which means it uses some extra randomness. We assume that such a distinguisher D outputs either 0 or 1 (which we understand correspondingly as "the input string is not random" and "the input string is random").

We say that a function $\texttt{negl}\colon \mathbb{N} \to \mathbb{R}$ is *negligible* if for every integer $c > 0$ there exists and integer N_c such that

$$\forall(x \in \mathbb{R})\ \forall(x \geq N_c) \qquad |\texttt{negl}(x)| < \frac{1}{x^c}.$$

Definition 3.6 Let $\ell(\cdot)$ be a polynomial and let G be a deterministic polynomial time algorithm such that for any input $seed \in \{0,1\}^n$, the algorithm G outputs a string of length $\ell(n)$. We say that G is a (cryptographic) PRNG if the following two conditions hold:

- For every n we have $\ell(n) > n$.

- For all probabilistic polynomial-time distinguishers D there exists a negligible function `negl` such that

$$|\mathbb{P}(D(x) = 1) - P(D(G(y)) = 1)| \leq \texttt{negl}(n),$$

  where x is chosen from $\mathcal{U}[\{0,1\}^{\ell(n)}]$ (i.e., the sequence of bits of length $\ell(n)$ chosen uniformly at random), seed y is chosen from $\mathcal{U}[\{0,1\}^n)$, and the probabilities are taken over all randomness used by D and the choices of x and y.

One can think of a distinguisher as a statistical test or a battery of statistical tests. According to the definition, if someone finds a distinguisher for a given PRNG, it is no longer considered a cryptographic PRNG. One such recent example is the aforementioned RC4 stream cypher (stream cyphers are indeed PRNGs). In 2014 about 80% of `https` traffic was encrypted using this cipher, however, due to found weaknesses it was removed from TLS 1.3 specification.

NIST (National Institute of Standards and Technology) is a U.S. federal agency that develops and promotes measurement standards and technology, including standards for cryptography. A document Rukhin *et al.* [120] describes a battery of tests designed for cryptographic purposes.

AES. It is worth mentioning that Advanced Encryption Standard (AES) is currently considered one of the best PRNGs in cryptography (no practical attack against this algorithm has been discovered). NIST supports the standard. It uses a block of 128 bits and supports keys of 128, 192 or 256 bits. For some more details on the algorithm, we refer a reader to the Wikipedia page `https://en.wikipedia.org/wiki/Advanced_Encryption_Standard`.

Some cryptographic PRNGs. We will mention some of such PRNGs. A longer list can be found e.g., at `https://en.wikipedia.org/wiki/Cryptographically_secure_pseudorandom_number_generator`. We can distinguish *special* and *number-theoretic* designs:

- Special designs:
  - ChaCha20 – replaced (around 2014) RC4 in OpenBSD, NetBSD and FreeBSD and it replaced SHA-1 in Linux

  - ISAAC – based on a variant of RC4 cipher
  - ANSI X9.17 standard

- Number-theoretic designs:
  - Blum Blum Shub – a security proof is based on the difficulty of the quadratic residuosity problem. However, the algorithm is not often used mainly because it is inefficient.
  - Blum-Micali algorithm – a security proof is based on the difficulty of the discrete logarithm problem. It is also inefficient.
  - Dual_EC_DRBG – a security proof is based on the decisional Diffie-Hellman assumption. It is believed to have a *kleptographic* National Security Agency (NSA) backdoor (as reported by *The Guardian* and *The New York Times* in 2013).

As a curiosity we mention that although the possibility of a backdoor in Dual_EC_DRBG was known, some companies still used it. For example, RSA Security, Inc. received 10 mln USD from the NSA (National Security Agency) to keep using it.

3.3.1 The RC4 stream cipher

RC4 is a symmetric stream cipher-key, which can be used as a pseudorandom number generator (PRNG). Roughly speaking, if Bob and Alice share a secret key K, then they may use RC4 to generate the same sequence of bits in a deterministic way – K plays a role of seed. Then, if Alice wants to encrypt the message, she `xor`s bit representation of a message with the output of RC4 (for details on `xor` operation see Section 3.4). Since for any bits $m, r \in \{0, 1\}$ we have $(m \texttt{ xor } r) \texttt{ xor } r = m$, Bob can decrypt it simply `xor`ing the ciphertext with the output of RC4.

RC4 consists of two algorithms:

(i) **KSA (Key Scheduling Algorithm)** uses a secret key to transform the identity permutation of n cards into some other permutation (one can model KSA as a card shuffle). It starts with the identity permutation (of "a deck of cards"), then uses a card shuffle with random choices replaced by a secret key (assumed to be random) to obtain a "random" permutation.

(ii) **PRGA (Pseudo Random Generation Algorithm)** starts with a permutation generated by KSA and outputs random bits from it, updating permutation at the same time.

We will consider RC4 algorithm, which generates numbers $x_r \in \bar{n} = \{0, \ldots, n-1\}$, where $n = 2^k$. In internal memory it stores (S, i, j), where S is a permutation of $\bar{n}$ and $i, j \in \bar{n}$. In a standard version $k = 8$, then it is recommended that the length of the key is between 40 and 128 bits. The KSA and PRGA algorithms are given in Fig. 3.2.

Algorithm 3 KSA

Input: n, key $K = K[)], \ldots, K[L-1]$ $K[i] \in \{0, \ldots, n-1\}, i = 1, \ldots, L$
Output: Permutation $S = S[0], \ldots, S[n-1]$

for $i := 0$ to $n-1$ **do**
  $S[i] := i$
end for

$j := 0$
for $i := 0$ to $n-1$ **do**
  $j := (j + S[i] + K[i \bmod L]) \bmod n$
  swap$(S[i], S[j])$
end for
$i, j := 0$

Algorithm 4 PRGA

Input: $S = S[0], \ldots, S[n-1]$
Output: Bytes $x_1, x_2, \ldots$

$i := 0$
$j := 0$
$r := 0$

while `GeneratingOutput` **do**
  $i = (i+1) \bmod n$
  $i = (j + S[i]) \bmod n$
  swap$(S[i], S[j])$
  $x_r = S[S[i] + S[j] \bmod n]$
  $r = r + 1$
end while
$i, j := 0$

Figure 3.2: KSA and PRGA: main ingredients of RC4 algorithm.

Remark 3.7 The idea of KSA was to produce a "random" permutation. Consider a case where a secret key K has length $L \geq n$, and each $K[i], i = 0, \ldots, L-1$ is chosen uniformly from the set $\{0, \ldots, n-1\}$. Then, the line

$j := (j + S[i] + K[i \bmod L]) \bmod n$

of Algorithm 3 can be replaced (for analysis of the randomness of the resulting permutation) with

$j := \mathbf{random}(\{0, \ldots, n-1\})$.

In such a case, the Key Scheduling Algorithm (KSA) corresponds to the *Cyclic-To-Random Transposition* card shuffling scheme (at step i, exchange the i-th card with a uniformly chosen one). It is known that it takes $\Theta(n \log n)$ steps to mix (Mironov, while studying RC4, proved an upper bound $O(n \log n)$; a matched lower bound was proven in [99], and in [84], it is conjectured (Conjecture 1) that the mixing time, measured in the separation distance, is asymptotically $n \log n$ as $n \to \infty$).

However, in RC4, only n steps are performed. It means that KSA does not produce a random (uniform) permutation. It is one of the main weaknesses of RC4. Attacks exploiting weaknesses in both the PRGA and KSA, or in the way RC4 was used in specific systems, are presented in [50, 45, 44, 89, 5].

As a result, in 2015, RC4 was prohibited in TLS by IETF, Microsoft, and Mozilla.

3.4 Pseudorandom bit generators.

We refer to $\{0,1\}$-valued sequences as pseudorandom bits. Many pseudorandom number generators generate (pseudo)random sequences of bits that are divided into blocks of a given length and then converted to the appropriate format, for example, for numbers in the $[0,1]$ range. The reverse question is how to obtain a random sequence of bits from a sequence of independent random variables with the same distribution $\mathcal{U}(\bar{M})$, where it can be done if $M = 2^k$. Details can be found in Exercise II.T.3. Another reason to consider pseudorandom bits is cryptography.

Before we provide specific generators, we need to recall/introduce a few terms. We will be dealing with sequences of bits, i.e., $\{0,1\}$-valued sequences, usually of a fixed length. This is related to the design of computers, where such sequences are stored in the so-called registers. The most common are 32-bit or 64-bit registers.

Let us have a short overview of the register operations. Let $p_i, q_i \in \{0,1\}, i = 1, \ldots, n$, $\mathbf{p} = (p_1, \ldots, p_n)$ and $\mathbf{q} = (q_1, \ldots, q_n)$.

- XOR (or EOR, EXOR, $\oplus$). XOR is True if and only if an odd number

of input sentences is true. The operation can be written in a table

p	q	$p \oplus q$
0	0	0
0	1	1
1	0	1
1	1	0

In the case of two sequences of bits, the operation is performed on consecutive pairs:

$$\mathbf{p} \oplus \mathbf{q} = (p_1 \oplus q_1, \ldots, p_n \oplus q_n).$$

- Bit shifts. The left shift is often denoted by $<<$, the right one by $>>$. The second argument indicates multiplicity. For example

$$x = y << 2$$

denote that x is obtained by shifting y by two bits to the left. In the left shift, abandoned places are substituted with zeros.

Now we show a generator using operation $\oplus$.

Example 3.8 [Exclusive OR Generator (XORG)] Let us start with $2^{16}-1 =$ 127-bit seed

$x_1, \ldots, x_{127} =$

000101101101100100010111100100101001101101101000100000010101111

110101001000010101101100000000001001100001011100111111111100111

The next bits are obtained using the procedure:

$$x_i = x_{i-1} \oplus x_{i-127} \qquad i \geq 128.$$

□

4 Testing good properties of PRNGs

Two types of tests assess the quality of a PRNG: theoretical and statistical. Theoretical tests examine the internal structure of the generator under

consideration. Classic examples are *lattice tests* [91] and *spectral tests* [71] (Section 3.3.4). See also [75] for some standard tests from this family. We encourage the reader to perform the following experiment. Consider a sequence generated by a linear congruence

$$x_j = (137x_{j-1} + 187) \mod 256$$

with period 256. Generate 256 pairs $(x_0, x_1), (x_1, x_2), \ldots, (x_{255}, x_0)$ and triples $(x_0, x_1, x_2), (x_1, x_2, x_3), (x_1, x_2, x_3), \ldots, (x_{255}, x_0, x_1)$, then plot the points. What can you see in the plots?

The second class of tests are statistical tests that use probabilistic methods. The PRNG structure is not important here, but the $\mathbf{u} = (u_1, u_2, \ldots, u_n)$ sequence is considered a simple sample, and we test if it comes from a random sequence. Recall that by a simple sample, we mean a sequence of i.i.d. random variables.

The sequence of pseudorandom numbers obtained from a PRNG is supposed to "look" as a sequence of independent random variables with a uniform distribution. After generating a sample of several thousand, it is not easy to check what it looks like, and therefore, you have to resort to statistical methods. Preparing tests to check the "randomness" of a sequence of numbers has been in focus since the beginning of the theory of generators. However, the approach has gained importance since it was noticed that it has utility in cryptography. It has been observed that the ideal situation is when the encrypted string appears completely random, as deviations from randomness can increase the chances of breaking a cryptographic protocol or discovering a private key.

Due to the demand mentioned above, batteries for tests are prepared. The most famous are

- Diehard tests. This is a test bundle developed by George Marsaglia in 1995.
- TestU01. The bundle was developed by Pierre L'Ecuyer and Richard Simard in 2007.
- NIST Test Suite developed by the *National Institute of Standards and Technology*. It is written mainly for cryptography applications.

Among others, the following types of tests are in use:

- uniformity tests – testing if the generated numbers are evenly distributed between zero and one (Kolmogorov-Smirnov test and chi-square goodness-of-fit test); from a pseudorandom bit generator, one should require: any tested subsequence should contain – on average – half zeros and half ones.

- tests based on "balls and boxes" scheme, including serial test – similar to uniformity tests, but for consecutive m-tuples.

- spacing test – we record distances between numbers which do not belong to the interval $[a, b]$ (with some fixed $0 \leq a < b \leq 1$) and compare the numbers to the theoretical geometric distribution.

- tests based on classic combinatorial tasks,

- tests based on the properties of realizations of random walks.

Furthermore we require the following properties.

- Scalability: A test applied to the entire sequence should produce the same results for any selected subsequence.

- Consistency: Generator quality should be tested for arbitrary seeds. It is vital in the context of cryptographic applications. There are known generators (e.g., the Mersenne Twister) whose "good" properties depend on a "randomly" selected seed.

4.1 Preliminary remarks

We will denote random variables with capitals. In particular, by U we denote random variable uniformly distributed $\mathcal{U}[0, 1)$, by B – uniformly distributed $\mathcal{U}[\{0, 1\}]$ (aka 'random bit') and by Y – random variables uniformly distributed $\mathcal{U}(\bar{M})$, i.e., on $\{0, 1, \ldots, M-1\}$. Realizations are written as small characters, e.g. u, b, y, respectively.

4.1.1 Converting numbers

Different tests require different types of random variables: distributed on $[0, 1)$, distributed on $\bar{M} = \{0, \ldots, M-1\}$ or a sequence of random bits. The same is with PRNGs: some return numbers from $[0, 1)$, or from $\{0, \ldots, M-$

1}, others return a sequence of bits. Thus, we need to be able to convert numbers between the above formats. Assume we have a string of numbers $u_1, u_2, \ldots$ from an interval $[0, 1)$, then we may convert them to numbers from a set $\bar{M}$ by

$$y_i = \lfloor M u_i \rfloor. \tag{4.2}$$

Similarly, numbers $y_1, y_2, \ldots \in \bar{M}$ can be converted to numbers from $[0, 1)$ by

$$u_i = \frac{y_i}{M}.$$

There are no real "continuous" numbers on a computer – when a PRNG returns a sequence $u_1, u_2, \ldots$ from $[0, 1)$, usually they have some *"granularity"*. Usually it is a binary expansion with some fixed length (that is why many suggested parameters used in tests are some powers of 2). For example, recall the PRNG from Visual Basic (given in Section 3.1):

$$\begin{aligned} x_i &= (1140671485 x_{i-1} + 12829163) \bmod 2^{24} \\ u_i &= x_i / 2^{24}. \end{aligned}$$

In other words, each u_i can be expressed as $(0.b_1 b_2 \ldots b_{24})_2$, there are 2^{24} different possible values of u_i, thus the best option is to convert it to numbers from $\{0, \ldots, 2^{24} - 1\}$ simply by

$$y_i = \lfloor 2^{24} u_i \rfloor.$$

We need to be careful. For example, we may have a test which requires a sequence of bits as input. Having $u_i \in [0, 1)$ as an output of a PRNG, we could do the following:

$$b_i = \begin{cases} 0 & \text{if} \quad u_i < 0.5 \\ 1 & \text{if} \quad u_i \geq 0.5 \, . \end{cases}$$

Note, however, that in this case, if $u_i = (0, b_1 b_2 \ldots b_{24})_2$, we only take the most significant bit b_1 and disregard all the others. We could easily "cheat" such a test by always setting e.g., $b_2 = \ldots = b_{24} = 0$ and make only b_1 "random". Thus, in such a case we should convert first $u_i \in [0, 1)$ into $y_i = \{0, \ldots, M-1\}$ and then take a binary expansion of y_i as input bits (note that it requires M to be a power of 2).

In subsequent sections, we will use numbers from $[0, 1)$, or from $\{0, \ldots, M-1\}$, or a sequence of bits. Keep in mind that a proper conversion must be done first.

4.1.2 Grouping numbers into multidimensional vectors.

Our goal is to verify the hypothesis that the sequence (u_j) is a realization of i.i.d. uniform $\mathcal{U}[0,1)$ random variables. Remember that such verification requires checking one-dimensional uniformity, as well as multidimensional uniformity. For this purpose, we need to represent the sequence under study as a sequence of vectors, which the representation we will use in the following part.

Assume we have a sequence $u_1, \ldots, u_n$ of $n = mr$ numbers from $[0,1)$. We arrange the numbers into r groups, each being an m-tuple

$$\begin{aligned} \boldsymbol{u}_1 = (u_1, \ldots, u_m), \quad & \boldsymbol{u}_2 = (u_{m+1}, \ldots, u_{2m}), \\ & \ldots, \ \boldsymbol{u}_r = (u_{(r-1)m+1}, \ldots, u_{rm}) \ . \end{aligned} \tag{4.3}$$

This way we may further operate on vectors $\boldsymbol{u}_i, i = 1, \ldots, r$ from a hypercube $[0,1)^m$. Similarly, by performing (4.2) first, i.e., $y_i = \lfloor M u_i \rfloor$ we may group these integers into r vectors from a discrete $\{0, 1, \ldots, M-1\}^m$ hypercube:

$$\begin{aligned} \boldsymbol{y}_1 = (y_1, \ldots, y_m), \quad & \boldsymbol{y}_2 = (y_{m+1}, \ldots, y_{2m}), \\ & \ldots, \ \boldsymbol{y}_r = (y_{(r-1)m+1}, \ldots, y_{rm}) \ . \end{aligned} \tag{4.4}$$

For completeness, having a sequence of $n = mr$ bits $b_1, \ldots, b_n$ we can group them into r numbers from $\{0,1\}^m$ in a similar way

$$\begin{aligned} \boldsymbol{b}_1 = (b_1, \ldots, b_m), \quad & \boldsymbol{b}_2 = (b_{m+1}, \ldots, b_{2m}), \\ & \ldots, \ \boldsymbol{b}_r = (b_{(r-1)m+1}, \ldots, b_{rm}) \ . \end{aligned} \tag{4.5}$$

4.1.3 The concept of p-value

Randomness is tested using a statistic T, a function of a sequence generated by a PRNG. By the *randomness*, we mean that a considered sequence is a realization of a uniformly distributed simple sample, which is our hypothesis $\mathcal{H}_0$. We will say $\mathcal{H}_0$ is the randomness hypothesis in such a case. It will be apparent whatever a sequence is; either $U_1, U_2, \ldots$ (sequence of numbers from $[0,1)$), $Y_1, Y_2, \ldots$ (sequence of numbers from $\bar{M}$) or $B_1, B_2, \ldots$ (sequence of bits). Let G be the distribution function of $T = T(U_1, \ldots, U_n)$ under the randomness hypothesis $\mathcal{H}_0$. Suppose that from the experiment, we obtained a sequence (u_j), which we test whether it is a realization of i.i.d. sequence of uniformly distributed r.v.s (U_j). Let $T(\text{obs}) = T(u_1, \ldots, u_n)$. Under the hypothesis $\mathcal{H}_0$ we then compute the so called p-values, by

- $p = \mathbb{P}(T \geq T(\text{obs})) = 1 - G(T(obs))$ for a one-sided right-tail test,
- $p = \mathbb{P}(T \leq T(\text{obs})) = G(T(obs))$ for a one-sided left-tail test,
- $p = \mathbb{P}(|T| \geq |T(\text{obs})|) = G(-T(obs)) + 1 - G(T(obs))$ for a two-sided test in case the distribution of T is symmetric (for general case see Exercise II.T.27).

We will continue the more detailed study of this concept in Section 4.6, in which we give further hints on the methodology of testing PRNGs. More details on intutions on p-value is given in Section 4.6.

4.2 Two fundamental goodness-of-fit tests: Kolmogorov-Smirnov and chi-square.

In this section, we will describe two important general-purpose tests: Kolmogorov-Smirnov and chi-square tests.

We begin with a small example, which will be used as an illustration for these two considered tests. In Fig. 4.3 a set A of 50 numbers $u_1^A, \ldots, u_{50}^A$, a set B of 50 numbers $u_1^B, \ldots, u_{50}^B$ and a set C of 50 numbers $u_1^C, \ldots, u_{50}^C$ is presented, together with a following partition $\mathcal{P}$ of $[0, 1]$

$$P_1 = (0, 0.15),\ P_2 = [0.15, 0.35),\ P_3 = [0.35, 0.6),\ P_4 = [0.6, 0.8),\ P_5 = [0.8, 1). \tag{4.6}$$

The sequence C comes from a quasi-number generator (see Section I.3).

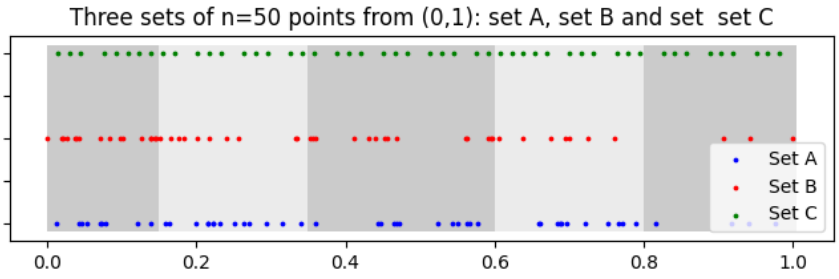


Figure 4.3: Three sets of $n = 50$ points from $[0, 1)$: set A (blue), set B (red) and set C (green). Partition $\mathcal{P}$ given in (4.6) grayed-out.

In the following two subsections, we will introduce the chi-square as mentioned above and Kolmogorov-Smirnov tests and try to answer the question: *Are numbers from set A (resp. B or C) "random"?*

4.2.1 Kolmogorov-Smirnov goodness-of-fit test (KS test)

Let $U_1, \ldots, U_n$ be random variables. We will now test the hypothesis that the marginal distribution of U_i is $\mathcal{U}[0,1)$. For the sequence under consideration we compute the corresponding empirical distribution function

$$\hat{F}_n(t) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(U_i \leq t).$$

Remembering that our hypothesis is the uniform distribution $\mathcal{U}[0,1)$, i.e. $F(t) = t$ for $t \in [0,1)$, our test statistic (often called KS statistic) is

$$D_n = \sup_{0 \leq t \leq 1} |\hat{F}_n(t) - t| \ .$$

From the Glivenko-Cantelli theorem, if U_i are uniformly distributed, then with probability one, the sequence D_n tends to zero. On the other hand, normed variables $\sqrt{n}D_n$ converge in distribution to the λ-Kolmogorov distribution, i.e., to the one with the distribution function

$$\begin{aligned} K(t) &= \lim_{n \to \infty} \mathbb{P}(\sqrt{n}D_n \leq t) = \sum_{j=-\infty}^{\infty} (-1)^j \exp(-2j^2t^2) \\ &= 1 - 2J(\exp(-2t^2), \qquad t > 0, \end{aligned} \tag{4.7}$$

where $J(t) = \sum_{j=1}^{\infty}(-1)^{j-1} \exp(-2j^2t^2))$. Tables of the distribution are available in most programming packages (e.g., in `Python`'s `scipy.stats.kstest`[5]). In particular we have $\lambda_{0.1} = 1.224$, $\lambda_{0.05} = 1.358$ and $\lambda_{0.01} = 1.628$, where

$$1 - K(\lambda_\alpha) = \alpha \ .$$

One problem is that K is the limiting distribution, and we use it to compute the statistic $\sqrt{n}D_n$. For small values of n, the approximation is not good; even for $n = 1000$, the error may be 0.9%. However, substituting in (4.7)

$$t + \frac{1}{6\sqrt{n}} + \frac{t-1}{4n}$$

[5] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.kstest.html`

instead of t, improves the accuracy for $n = 1000$ (100, 10) to 0.003 (0.027, 0.27)%; see [129].

We have the following algorithm for computing D_n. Let $U_{(1)}, \ldots, U_{(n)}$ be the order statistic from a sample $U_1, \ldots, U_n$, i.e., its non-decreasing ordering and let

$$\begin{aligned} D_n^+ &= \max_{1 \le i \le n} \left(\frac{i}{n} - U_{(i)} \right), \\ D_n^- &= \max_{1 \le i \le n} \left(U_{(i)} - \frac{i-1}{n} \right) . \end{aligned}$$

Then

$$D_n = \max(D_n^+, D_n^-) .$$

Summarising, for a given sequence $u_1, \ldots, u_n$ we compute the statistic $D_n(obs)$ and then the p-value using the formula

$$p = 1 - K\left(\sqrt{n} D_n(obs) + \frac{1}{6\sqrt{n}} + \frac{\sqrt{n} D_n(obs) - 1}{4n} \right). \tag{4.8}$$

Example 4.1 (Kolmogorov-Smirnov test for points in sets A, B and C from Fig. 4.3). The empirical distribution functions of points from sets A and B:

$$\hat{F}_n^A(t) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(u_i^A \le t), \quad \hat{F}_n^B(t) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(u_i^B \le t), \quad \hat{F}_n^C(t) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(u_i^C \le t)$$

are depicted in Fig. 4.4.

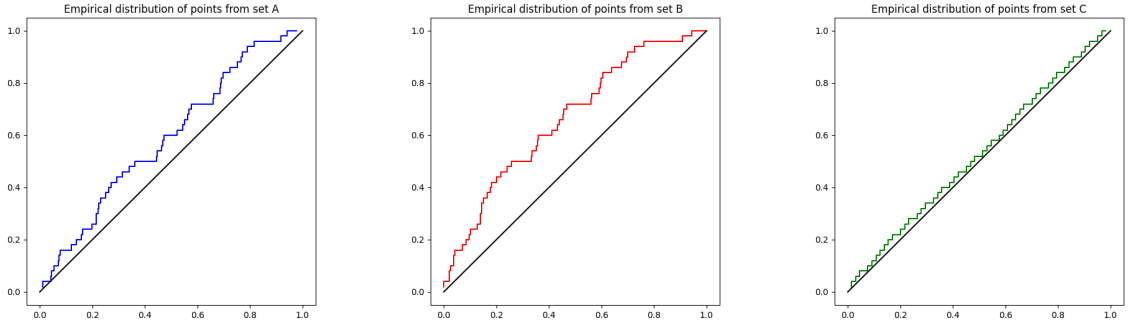


Figure 4.4: Empirical distribution functions $\hat{F}_n^A(t)$ (blue, left), $\hat{F}_n^B(t)$ (red, middle) and $\hat{F}_n^C(t)$ (green, right) of points from set A, B and C respectively. Each plot contains a c.d.f. $F(t) = t$ of the uniform distribution $\mathcal{U}[0,1)$ (black).

As shown in Fig. 4.4, all empirical distribution functions are above $F(t)$. "By an eye inspection" we see that $\hat{F}_n^B(t)$ is "further" from $F(t)$ than $\hat{F}_n^A(t)$, we also see that $\hat{F}_n^C(t)$ is very close to $F(t)$. The corresponding KS statistics confirm these observations:

$$\begin{aligned} D_n^A(obs) &= \sup_{0\le t\le 1} |\hat{F}_n^A(t)| &= 0.12955, \\ D_n^B(obs) &= \sup_{0\le t\le 1} |\hat{F}_n^B(t)| &= 0.2312, \\ D_n^C(obs) &= \sup_{0\le t\le 1} |\hat{F}_n^C(t)| &= 0.0299. \end{aligned}$$

We need yet to compute the corresponding p-values using the suggested formula (4.8):

$$\begin{aligned} p^A &= 1 - K\left(\sqrt{n}D_n^A(obs) + \frac{1}{6\sqrt{n}} + \frac{\sqrt{n}D_n^A(obs) - 1}{4n}\right) = 34.12\%, \\ p^B &= 1 - K\left(\sqrt{n}D_n^B(obs) + \frac{1}{6\sqrt{n}} + \frac{\sqrt{n}D_n^B(obs) - 1}{4n}\right) = 0.78\%, \\ p^C &= 1 - K\left(\sqrt{n}D_n^C(obs) + \frac{1}{6\sqrt{n}} + \frac{\sqrt{n}D_n^C(obs) - 1}{4n}\right) = 99.99\%. \end{aligned}$$

Thus, according to the KS test, we have to accept (we have no reasons to reject) the hypothesis that points from sets A and C were sampled from

$\mathcal{U}[0,1)$ distribution (obtaining statistic of value at least D_n^A is quite likely – 34.12% and obtaining the statistic of value at least D_n^C is very likely – 99.99%), whereas we will reject the hypothesis that points from set B were sampled from $\mathcal{U}[0,1)$ distribution (it is implausible that points sampled from $\mathcal{U}[0,1)$ would "produce" statistic of value at least D_n^B – it happens only in 0.78% of cases). We know that points from the set C are not coming from a PRNG; they are quasirandom numbers. Note that the p-value is suspiciously large, and although the KS test cannot reject the randomness hypothesis, it requires further investigation. □

4.2.2 Chi-square goodness-of-fit test

We begin with recalling basic ideas of the chi-square Pearson test for multinomial distribution $\mathrm{M}(r, p_1, \ldots, p_k)$. We have k possible mutually exclusive outcomes, say $1, \ldots, k$, with probabilities $p_1, \ldots, p_k$ respectively, and r independent trials. Let O_i be the number of outcomes of type i in the sequence of r trials and $E_i = \mathbb{E}O_i = rp_i$. Define the so-called chi-square statistic

$$X_k^2 = \sum_{i=1}^{k} \frac{(O_i - rp_i)^2}{rp_i}.$$

Since $\sum_i O_i = r$ and $\sum_i p_i = 1$ we can write

$$X_k^2 = \sum_{i=1}^{k} \frac{O_i^2}{rp_i} - r.$$

Then as r converges to infinity, the statistic $X_r^2 \xrightarrow{\mathcal{D}} \chi_{k-1}^2$, where χ_{k-1}^2 is chi-square distribution with $k-1$ degrees of freedom. Remark that χ_{k-1}^2 is an approximation distribution for X_k^2 statistic.

Thus, for given observations, we count the number of occurrences O_i for each type $i = 1, \ldots, k$ and compute the χ^2 statistic and the corresponding p-value.

Note that – in contrast to, e.g., the KS test – the chi-square goodness-of-fit is not a specific test but a type of test. It is still up to use what k and r are and their interpretation (see later Section 4.3).

As a **rule of thumb**, we should have (on average) at least five observations of each type. In practice, while designing experiments we should thus take

$$r \geq \left\lceil \frac{5}{\min\limits_{1\leq i\leq k} p_i} \right\rceil . \tag{4.9}$$

In a situation when $k >> r$ tests based on the scheme of arrangements of balls and boxes are recommended (see Section 4.3).

Example 4.2 (Chi-square test for points in sets A, B and C from Fig. 4.3) Consider the partition given in (4.6). We thus identify "type" of outcome with P_i's, $i = 1, \ldots, 5$. Thus $k = 5$, define $p_i = |P_i|$ and

$$O_i^A = \#\{u_j^A : u_j^A \in P_i\}, \quad O_i^B = \#\{u_j^B : u_j^B \in P_i\}, \quad O_i^C = \#\{u_j^B : u_j^C \in P_i\}.$$

Note that P_1 is least likely – $p_1 = |P_1| = 0.15$. Thus, according to the rule of thumb (4.9) we should have at least

$$\left\lceil \frac{5}{0.15} \right\rceil = 34,$$

what is the case (we have $r = 50$). The chi-square statistics for sets A, B and C are thus given correspondingly by

$$X_A^2(obs) = \sum_{i=1}^{k} \frac{(O_i^A - rp_i)^2}{rp_i}, \quad X_B^2(obs) = \sum_{i=1}^{k} \frac{(O_i^B - rp_i)^2}{rp_i}, \quad X_C^2(obs) = \sum_{i=1}^{k} \frac{(O_i^C - rp_i)^2}{rp_i}.$$

The number of points in each P_i together with the corresponding values of chi-square tests are summarized in Table 4.2.

P_i	np_i	O_i^A	$\frac{(O_i^A - rp_i)^2}{rp_i}$	O_i^B	$\frac{(O_i^B - rp_i)^2}{rp_i}$	O_i^C	$\frac{(O_i^C - rp_i)^2}{rp_i}$
$[0, 0.15)$	7.5	9	0.3	16	9.63	8	0.033
$[0.15, 0.35)$	10	14	1.6	10	0.00	10	0.00
$[0.35, 0.6)$	12.5	12	0.02	14	0.18	12	0.02
$[0.6, 0.8)$	10	11	0.1	7	0.9	11	0.1
$[0.8, 1]$	10	4	3.6	3	4.9	9	0.1
chi-square			$X_A^2(obs) =$**5.62**		$X_B^2(obs) =$**15.61**		$X_C^2(obs) =$**0.25**

Table 4.2: Chi-square test for points from sets A, B and C from Fig. 4.3 using partition $\mathcal{P}$ given in (4.6).

Assuming numbers being distributed uniformly on $[0,1)$, statistics X_A^2, X_B^2 and X_C^2 should have (approximately) $\chi^2(4)$ distribution, i.e., chi-square distribution with 4 degrees of freedom. What is left is to compute the corresponding p-values:

$$\begin{aligned} p^A &= 1 - F_{\chi_4^2}(X_A^2(obs)) &= 22.93\%, \\ p^B &= 1 - F_{\chi_4^2}(X_B^2(obs)) &= 0.36\%, \\ p^C &= 1 - F_{\chi_4^2}(X_C^2(obs)) &= 99.26\%. \end{aligned}$$

Thus, according to the chi-square test, we have no reasons to reject the hypothesis that numbers from set A are distributed uniformly – the location of points from set A within the partition $\mathcal{P}$ given in Table 4.2 is quite likely – happens in 22.93% of cases. Similarly, set C – the situation in Table 4.2 is very likely – happens in 99.26% However, the location of points from set B within the partition $\mathcal{P}$ is quite unlikely – it happens only in 0.36% of all cases. Thus, setting the significance level to 5% or even to 1%, we would reject the hypothesis that points from set B are a sample from uniform distribution. □

Note that the distribution of χ_k^2 is Gamma$(k/2, 1/2)$. For $k \geq 50$ types of observations, it is recommended to use the central limit approximation to the χ_k^2 distribution. Recall that

$$\chi_k^2 \stackrel{\mathcal{D}}{=} Z_1^2 + \ldots + Z_k^2.$$

Since $\mathbb{E}\chi_k^2 = k$ and $\mathrm{Var}\chi_k^2 = 2k$, from the central limit theorem we have

$$\frac{\chi_k^2 - k}{\sqrt{2k}} \stackrel{\mathcal{D}}{\to} \mathcal{N}(0,1).$$

Remark 4.3 (On Examples 4.1 and 4.2) Concerning sets A, B and C from Fig. 4.3, actually points from set A were sampled randomly from $[0,1)$ (using the `PCG64` generator built in `NumPy` in `Python`). Neither set B nor C are random. Set B is the following transformation of points from A:

$$u_i^B = \frac{e^{u_i^A - 1} - m}{M - m}, \text{ where } m = \min_i e^{u_i^A - 1} - \varepsilon, \; M = \max_i e^{u_i^A - 1} + \varepsilon$$

with $\varepsilon = 10^{-7}$. The set C is a sequence of so-called quasirandom numbers (see Section I.3). Note that both the chi-square and the KS-test detected "nonrandomness" of the set B, whereas they failed at detecting the "nonrandomness" of the set C. Moreover, both tests stated that it is very likely that C is random (the p-values were $> 99\%$). Intuitively – the set of points is "too perfect"; it can especially be seen on the plot of the empirical distribution function in Fig. 4.4 (right). It shows the importance of applying a **second-level testing** – the procedure is described later in Section 4.6. We continue the example therein, and the "nonrandomness" is easily spotted.

The "nonrandomness" of C is also spotted by the *frequency of pairs test* (see Example 4.5). However, if we shuffle the points in C ("randomly"), then the frequency of pairs test finds no proof of such "nonrandomness".

Spacing test. For a sequence $U_1, U_2, \ldots$ consider *spacings* between the appearance of $U_j \in (\alpha, \beta]$, where $0 \le \alpha < \beta \le 1$. Formally we consider

$$\{i = 1, 2, \ldots : U_i \in (\alpha, \beta]\} = \{S_1, S_2, \ldots\},$$

where $S_1 < S_2 < \ldots$. The consecutive spacings are $C_1 = S_1, C_2 = S_2 - S_1, \ldots$. Assuming a null hypothesis $\mathcal{H}_0$ that (U_i) is an i.i.d. sequence uniformly distributed $\mathcal{U}[0,1]$, sequence $C_1, C_2, \ldots$ is i.i.d. and

$$\mathbb{P}(C_1 = l) = \mathbb{P}(U_1 \notin (\alpha, \beta], \ldots, U_{l-1} \notin (\alpha, \beta], U_l \in (\alpha, \beta]) = (1 - \delta)^{l-1}\delta^l,$$

for $l = 1, 2, \ldots$, where $\delta = \beta - \alpha$.

For our computations we have to make some modifications, because as an input we have a finite sequence $U_1, \ldots, U_n$. If $U_j \notin (\alpha, \beta]$ for all $j = 1, \ldots, n$, then we define $C_1 = n$ and $K = 1$, otherwise

$$\{i = 1, 2, \ldots, n : U_i \in (\alpha, \beta]\} = \{S_1, S_2, \ldots, S_K\},$$

where $S_1 < S_2 < \ldots < S_K$ and the sequence of spacing is $C_1 = S_1, C_2 = S_2 - S_1, \ldots, S_K - S_{K-1}$.

$$C_1 = \min\{i \ge 0 : U_{i+1} \in (\alpha, \beta]\}, \quad C_{j+1} = \min\{i \ge 0 : U_{C_j+1+i} \in (\alpha, \beta]\}.$$

We suggest the following test. For given numbers $u_1, \ldots, u_n$ and fixed α and β (and thus δ) compute the spacings $C_1, \ldots, C_K$ (note that K depends on the sequence). Consider boxes

$$A_0 = \{0\},\ A_1 = \{1\},\ \ldots,\ A_{s-1} = \{s-1\},\ A_s = \{s, s+1, \ldots,\}.$$

We suggest taking

$$s \geq \max\left\{5, \left\lceil \frac{5(1-\delta)}{\delta} \right\rceil \right\}.$$

For $\delta \leq 1/2$ from the Chebyshev inequality, we have

$$P\left(C \geq s\right) \leq P\left(C \geq \frac{5(1-\delta)}{\delta}\right) = P\left(C \geq \frac{(1-\delta)}{\delta} + 4\frac{(1-\delta)}{\delta}\right) =$$

$$\leq P\left(C \geq \frac{(1-\delta)}{\delta} + \frac{1}{2}\cdot 4\frac{\sqrt{1-\delta}}{\delta}\right) = P\left(C \geq \mathbb{E}C + 2\mathrm{Var}C\right) \leq \frac{1}{4}.$$

Thus, at last, $3/4$ of the probability mass will be in boxes $A_0, \ldots, A_{s-1}$ and at most $1/4$ in the last box A_s. For $\delta > 1/2$ we will always take $s = 5$.

Under the randomness hypothesis $\mathcal{H}_0$ we should have on average Kp_i spacings in boxes $A_i, i = 0, \ldots, s$, where

$$p_i = \delta(1-\delta)^i, i = 0, \ldots, s-1, \quad p_s = 1 - \sum_{i=0}^{s-1} p_i.$$

We will test it using the chi-square test. We compute the number of observed spacings:

$$O_i = \#\{j : C_j \in A_i\}, \quad i = 0, \ldots, s$$

and compute the statistic

$$X^2(obs) = \sum_{i=0}^{s} \frac{(O_i - Kp_i)^2}{Kp_i}.$$

Under $\mathcal{H}_0$ this should have χ^2 distribution with s degrees of freedom, thus we may compute the corresponding p-value.

As the following example demonstrates, the choice of boxes in the chi-square goodness-of-fit test is a delicate matter.

Example 4.4 (Spacing test for Matlab generators). In earlier versions of Matlab, a default generator showed some strange behaviour. In the generator called `state`, one records small values, spacings between $(0, \delta]$. We let $\delta = \beta - \alpha = 0.01$. In this experiment with $R = 5 \cdot 10^7$ replications, the histogram for spacing of the small values is presented in Fig. 4.5 (a), whereas the histogram for large values is presented in Fig. 4.5 (b)

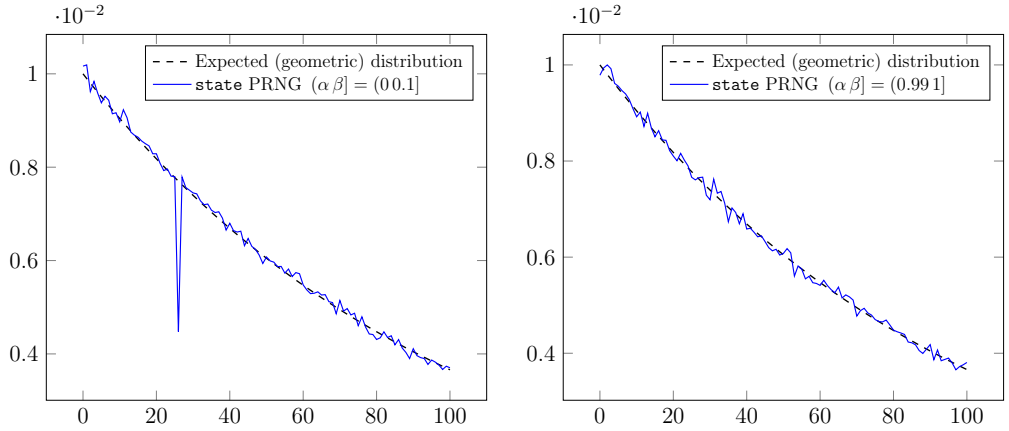


Figure 4.5: $R = 5 \cdot 10^7$ numbers invoked by `rand('state',0)`; (a) histogram of small values $(\alpha, \beta] = (0, 0.01]$; (b) histogram of large values $(\alpha, \beta] = (0.99, 1]$;

The appearance of individual lengths should resemble the function $\delta(1-\delta)^k$ (for a large number of random numbers) in both cases, however in the case of $(\alpha, \beta] = (0, 0.01]$ we can observe a mysterious deviation at value 26. Note that there is no such deviation in the case of $(\alpha, \beta] = (0.99, 1]$.

For a comparison, the same experiment was performed using the `twister` generator, no such deviation was observed; see Fig. 4.6.

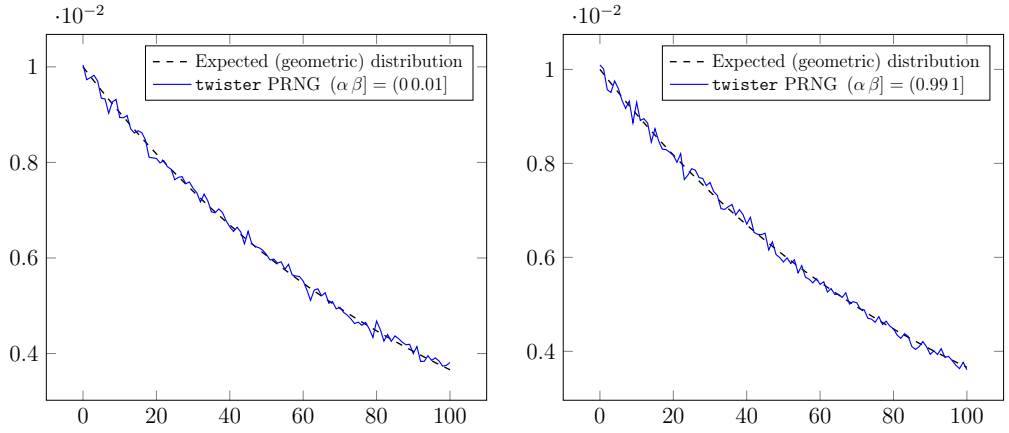


Figure 4.6: $R = 5 \cdot 10^7$ numbers invoked by `rand('twister',0)`; (a) histogram of small values $(\alpha, \beta] = (0, 0.01]$; (b) histogram of large values $(\alpha, \beta] = (0.99, 1]$;

In Table 4.3 the results of chi-square for this spacing test with boxes $A_0 = \{0\}, A_1 = \{1\}, \ldots, A_{103} = \{103\}, A_{104} = \{104, 105, \ldots\}$ are presented. After generating $n = 5 \cdot 10^7$ points the spacings between numbers in intervals $(\alpha, \beta]$ were recorded – the number of spacings is thus also random; in our experiment, it varied from 499539 to 500522. Experiments were performed for generators `rand('state',0)` and `rand('twister',0)` for $(\alpha, \beta]$ equal to $(0, 0.01]$ and $(0.99, 1]$ respectively, and the results are presented in Tables 4.3 and 4.4. The chi-square statistic $X^2(obs)$ for cases in which no mysterious deviation was recorded had values of 83.160, 94.856 and 97.54. However, for the case of `state` generator and $(\alpha, \beta] = (0, 0.01]$, the value of the statistic is extremely different – 758.9. There are 2237 observations within box A_{26}, whereas on average there should be $Kp_{26} = 3847.512$ of them. It implies that the value of just one term

$$\frac{(O_{26} - kp_{26})^2}{kp_{26}} = 674.137$$

is very large (compared to 0.012, 0.554 and 1.554 for other cases), which results in extremely small p-value $4.12 \cdot 10^{-100}$.

	`rand('state',0)`		`rand('twister',0)`	
K	499649	499558	500522	499539
$(\alpha, \beta]$	$(0, 0.01]$	$(0.99, 1]$	$(0, 0.01]$	$(0.99, 1]$
$\frac{(O_{26}-kp_{26})^2}{kp_{26}}$	674.137	0.012	0.554	1.554
$X^2(obs)$	758.90	83.160	94.856	97.540
p-value	$\mathbf{4.12 \cdot 10^{-100}}$	0.934	0.728	0.659

Table 4.3: Chi-square results for spacing test with boxes $A_i = \{i\}, i = 0, \ldots, 103, A_{104} = \{104, \ldots\}$.

It is not necessary to consider so many – 105 – boxes. It is natural to consider larger boxes. In Table 4.4 the results for 27 boxes $A'_0 = \{0, 1, 2, 3\}, A'_1 = \{4, 5, 6, 7\} \ldots, A'_{25} = \{100, 101, 102, 103\}, A'_{26} = \{104, \ldots\}$ are presented. Note that in this case, the final p-value for (and only for) `state` generator and $(\alpha, \beta] = (0, 0.01]$ is still very small ($1.18 \cdot 10^{-24}$). The histograms for these bins (for both `state` and `twister`, but only for the case $(\alpha, \beta] = (0, 0.01]$) are presented in Fig. 4.4.

	`rand('state',0)`		`rand('twister',0)`	
$(\alpha, \beta]$	$(0, 0.01]$	$(0.99, 1]$	$(0, 0.01]$	$(0.99, 1]$
$X^2(obs)$	178.25	24.125	17.624	26.503
p-value	$\mathbf{1.18 \cdot 10^{-24}}$	0.568	0.889	0.435

Table 4.4: Chi-square results for spacing test with boxes $A'_0 = \{0, 1, 2, 3\}, A'_1 = \{4, 5, 6, 7\} \ldots, A'_{25} = \{100, 101, 102, 103\}, A'_{26} = \{104, \ldots\}$.

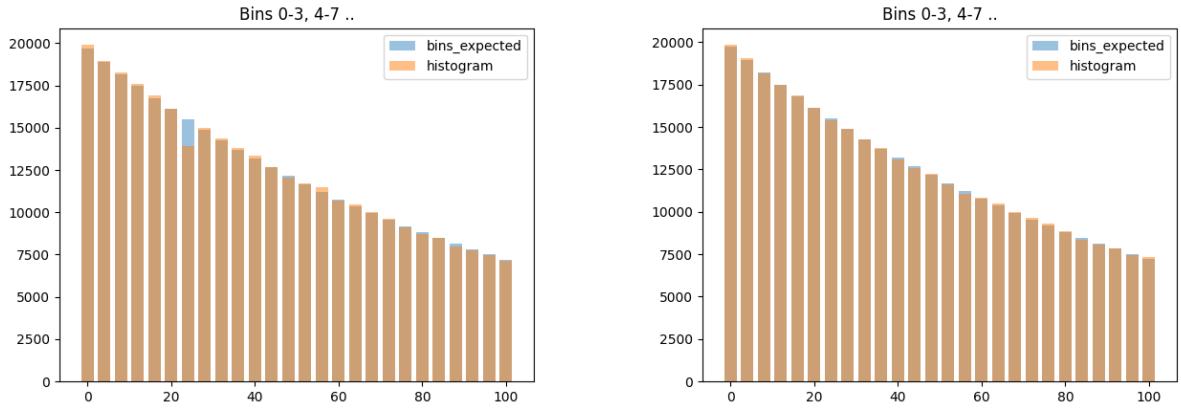


Figure 4.7: Histograms for boxes (bins) of size 4 for `state` and `twister` generators (both for case $(\alpha, \beta] = (0, 0.01]$) together with the expected number of observations with boxes. The last box $A'_{26} = \{104, \ldots\}$ is not depicted.

The experiments were performed in Matlab R2017a (9.2.0.538062).

□

Serial test (m,L). [Knuth [71], p. 60.] The goal is to test whether a sequence of numbers $u_1, \ldots, u_n$ from $[0, 1)$ is a realization of i.i.d. random variables with the uniform distribution $\mathcal{U}[0, 1)$.

Consider a hypercube in $[0, 1)^m$ and further split each of m sides into $L \in \mathbb{N}$ segments $[i/L, (i+1)/L)$, $i = 1, \ldots, L$ of equal lengths. In this way, we obtain $k = L^m$ sub-hypercubes, which we identify with boxes. Assume $n = mr$. We group the sequence $u_1, \ldots, u_n$ into r vectors from $[0, 1)^m$ following (4.3). Recall, we obtain

$$\begin{aligned} \boldsymbol{u}_1 = (u_1, \ldots, u_m), \quad \boldsymbol{u}_2 = (u_{m+1}, \ldots, u_{2m}), \\ \ldots, \ \boldsymbol{u}_r = (u_{(r-1)m+1}, \ldots, u_{rm}) \ . \end{aligned} \tag{4.10}$$

The example with $m = 3, L = 3$ is presented in Figure 4.8.

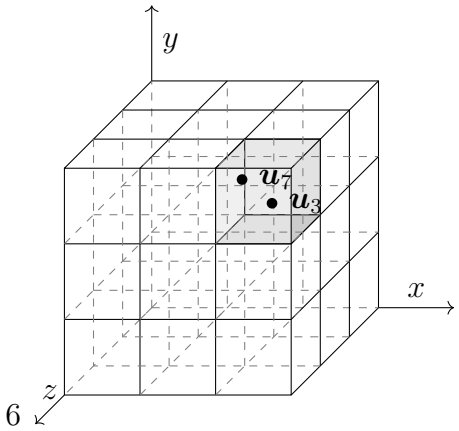


Figure 4.8: Sample cube $[0,1)^m, m = 3$, each side split into $L = 3$ segments obtaining $k = L^3 = 27$ smaller sub-hypercubes. Sample points $\boldsymbol{u}_3 = (0.8, 0.73, 0.7)$ and $\boldsymbol{u}_7 = (2.1, 2.6, 2.35)$ are presented together with the corresponding box $A_{2,2,2} = \left\{(v_1, v_2, v_3) : v_i \in \left(\frac{2}{3}, 1\right), i = 1, 2, 3\right\}$, or correspondingly $\boldsymbol{y}_3 = \boldsymbol{y}_7 = (2, 2, 2)$ and $A'_{2,2,2} = (2, 2, 2)$ (see description in the text).

The sub-hypercubes are as follows:

$$A_{q_1,\ldots,q_m} = \left\{(v_1, \ldots, v_m) : v_i \in \left[\frac{q_i}{L}, \frac{q_i + 1}{L}\right), i = 1, \ldots, m\right\}, \tag{4.11}$$

where $q_i \in \{0, \ldots, L-1\}$. For example, in Fig. 4.8 we have $A_{2,2,2} = \left\{(v_1, v_2, v_3) : v_i \in \left(\frac{2}{3}, 1\right), i = 1, 2, 3\right\}$. Sometimes it is convenient to enumerate the sub-hypercubes $1, \ldots, k$, then we may say that the j-th point $\boldsymbol{u}_j$ is in the l-th sub-hypercube $(1 \le j \le r, 1 \le l \le k)$.

It is convenient to convert first each u_i into $y_i = \lfloor Lu_i \rfloor$, i.e., to follow (4.4), then we obtain r vectors each being an m-dimensional number with integer values

$$\begin{aligned} \boldsymbol{y}_1 = (y_1, \ldots, y_m), \quad & \boldsymbol{y}_2 = (y_{m+1}, \ldots, y_{2m}), \\ & \ldots, \ \boldsymbol{y}_r = (y_{(r-1)m+1}, \ldots, y_{rm}) \ . \end{aligned}$$

In such a case, a representation of sub-hypercubes is straightforward:

$$A'_{q_1,\ldots,q_m} = (q_1, \ldots, q_m),$$

see Fig. 4.8 for an example.

The testing procedure. We group the output of the PRNG of length $n = mr$ into r vectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r$ from $(0,1)^m$. Then we count the number of vectors within each sub-hypercube $A_{q_1,\ldots,q_m}$ defined in (4.11), i.e.,

$$O_{q_1,\ldots,q_m} = \#\{i : \boldsymbol{u}_i \in A_{q_1,\ldots,q_m}\}$$

for each $(q_1, \ldots, q_m) \in \{0, \ldots, L-1\}^m$. Recall we have $k = L^m$. Finally, we compute

$$X^2(obs) = \sum_{(q_1,\ldots,q_m)\in\{0,\ldots,L-1\}^m} \frac{(O_{q_1,\ldots,q_m} - r/k)^2}{r/k}.$$

We may enumerate all sub-hypercubes $A_{q_1,\ldots,q_m}$ as consecutive numbers $1, \ldots, k$, then compute O_i – the number of vectors $\boldsymbol{u}_j$ within i-th sub-hypercube and express the statistic as

$$X^2(obs) = \sum_{i=1}^{k} \frac{(O_i - r/k)^2}{r/k}.$$

Fixing k (i.e., L and m), under the randomness $\mathcal{H}_0$ hypothesis (that the sequence was sampled from the uniform distribution), for large r, statistic X^2 is approximately chi-square with $k-1$ degrees of freedom.

It is known that (see Holst [59], L'Ecuyer et al [79])

- if $r \to \infty$ and k is fixed, then X^2 converges to the chi-square distribution with $k-1$ degrees of freedom,
- if $r \to \infty$ and $k \to \infty$ so that $r/k \to \gamma$ and $0 < \gamma < \infty$, then $T = (X^2 - \mu_{r,k})/\sigma_{r,k}$ converges in distribution to the standard normal distribution, where $\mu_{r,k} = \mathbb{E}X^2$ and $\sigma^2_{r,k} = \mathrm{Var}X^2$. In Exercise II.T.17 we recommend to demonstrate that $\mathbb{E}X^2 = k-1$; however, a more difficult is to show that $\mathrm{Var}X^2 \sim 2(k-1)$. It is recommended that $0 \le \gamma < 5$, otherwise the distribution of the statistic X^2 is better approximated by the chi-square distribution.

To accept or reject the $\mathcal{H}_0$ hypothesis, we use the test for $r \gg k$, rejecting the hypothesis if $X^2(obs)$ is very large (or very small). To be more exact, we compute the p-value:

$$p - \text{value} = 1 - F_{\chi^2_{k-1}}(X^2(obs)).$$

The use of second-level testing is recommended (see Section 4.6).

It is left to the reader to adapt this testing procedure for a sequence $y_1, y_2, \ldots$, where $y_i = \lfloor Lu_i \rfloor$.

Due to a large number of sub-hypercubes when L is large, the serial test is usually used for $m = 1$ (*frequency test*) or for $m = 2$ (*frequency of pairs test*), both of which are described below. In practice, for larger m, other tests like the *collision test* or *birthday spacing test* (to be described in due course) are recommended (instead of the serial test).

Serial test belongs to a family of tests based on "balls and boxes" scheme (described in general in Section 4.3) – from now on we identify a point $\boldsymbol{u}_j$ with a *ball* and a sub-hypercube $A_{q_1,\ldots,q_m}$ with a *box*.

Frequency test (Serial test with $m = 1$). It is a special case of a serial test with $m = 1$. In this test, we thus study whether the marginal distribution is the uniform distribution $\mathcal{U}[0, 1)$. Let us split the interval $[0, 1)$ into L intervals of equal length. We identify the resulting intervals $A_j = \{u : u \in [j/L, (j + 1)/L)\}, j = 0, \ldots, L - 1$ with *boxes* and the numbers $u_1, u_2, \ldots$ with *balls*. In our balls and boxes scheme, we thus have $m = 1$ and $k = L$ boxes, each having – under randomness hypothesis $\mathcal{H}_0$ – equal probability $\mathbb{P}(U_j \in A_j) = 1/k$. Using the chi-square test, we may test whether the number of points (balls) u_j within each box has a multinomial distribution $\mathrm{M}(n, (1/k, \ldots, 1/k))$.

For a convenience, instead of real numbers $u_1, \ldots, u_n \in [0, 1)$ we may consider a sequence $y_1, \ldots, y_n \in \{0, 1, \ldots, k - 1\}$ using the transformation $y_j = \lfloor ku_j \rfloor$. Then simply the boxes are $A'_j = j, j = 0, \ldots, k-1$ and y_j denotes the number of the box with j-th "ball". Depending on computer architecture, we can choose k to be some power of 2; for example $L = 64 = 2^6$ (recall, $m = 1$, thus $k = L$), then y_j represents 6 most significant bits in a binary representation of u_j. Thus, for each $i \in \{0, \ldots, k - 1\}$ we compute

$$O_i = \#\{j : y_j = i\}$$

and apply the chi-square test

$$X^2(\mathrm{obs}) = \sum_{i=0}^{k-1} \frac{\left(O_i - \frac{n}{k}\right)^2}{\frac{n}{k}}$$

with $k - 1$ degrees of freedom.

Frequency of pairs test (Serial test with $m = 2$). [[71], p. 61.] It is a special case of the serial test with $m = 2$. Assume thus that we have $n = 2r$ numbers $u_1, \ldots, u_m$. This time we will have a look at pairs $(u_{2i-1}, u_{2i}), i = 1, \ldots, r$. For an i.i.d. sequence $U_1, U_2, \ldots, U_n$ from the distribution $\mathcal{U}[0, 1)$, the sequence (U_{2j-1}, U_{2j}) $j = 1, 2, \ldots, r$ is a sequence of i.i.d. pairs and (U_1, U_2) are independent uniformly $\mathcal{U}[[0, 1)^2)$ distributed random variables. In other words, we will test the distribution of r points in a unit square $[0, 1)^2$. We split each side of a square into L intervals of equal lengths and construct $k = L^2$ boxes:

$$A_{s,t} = \{(v, w) : v \in [s/L, (s + 1)/L), w \in [t/L, (t + 1)/L)\}.$$

Afterwards, we count the number of pairs (u_{2i-1}, u_{2i}) falling into each box $A_{q,r}$. Similarly, as in the frequency test, it is convenient to use the transformation

$$y_j = \lfloor Lu_j \rfloor,$$

then the boxes are $A'_{s,t} = \{(s, t)\}$. We compute the number of pairs equal to (s, t):

$$O_{s,t} = \#\{j : y_{2j-1} = s, y_{2j} = t\}$$

and apply the chi-square test

$$X^2(obs) = \sum_{(s,t)\in\{0,\ldots,L-1\}^2} \frac{\left(O_{st} - \frac{r}{k}\right)^2}{\frac{r}{k}} = \sum_{(s,t)\in\{0,\ldots,L-1\}^2} \frac{\left(O_{s,t} - \frac{r}{L^2}\right)^2}{\frac{r}{L^2}}$$

with $k - 1 = L^2 - 1$ degrees of freedom.

Example 4.5 We continue the example of three sets A, B and C of $n = 50$ points given in Fig. 4.3. Let us take $L = 3$, thus we consider $r = 25$ pairs $(u_1, u_2), \ldots, (u_{49}, u_{50})$ with $k = L^2 = 9$ boxes. On average, we expect $25/9 \approx 2.77$ pairs to be within each box. The number of observed pairs in each box as well as chi-square statistics for each set, are presented in Fig. 4.9

2.77	2.77	2.77
2.77	2.77	2.77
2.77	2.77	2.77

Expected nr of observations

4	4	3
4	2	1
3	2	2

$X^2_A(obs) = 3.44$
set A

6	6	2
4	3	0
1	1	2

$X^2_B(obs) = 13.52$
set B

0	6	8
3	0	8
0	0	0

$X^2_C(obs) = 37.28$
set C

2	4	2
1	3	3
6	3	1

$X^2_{C'}(obs) = 7.04$
set C'

Figure 4.9: Expected number of observations and observed values for sets A, B, C and C' (see description in the text) and corresponding values of the chi-square statistics

Recall that set C is a sequence of quasi-random numbers – we can see that half of the boxes are empty; this suggests that the frequency of pairs test should reject the hypothesis that they are random (recall, it was not spotted by KS or chi-square test). The corresponding p-values are:

$$\begin{aligned} p_A &= 1 - F_{\chi^2_8}(X^2_A(obs)) = 90.38\%, \\ p_B &= 1 - F_{\chi^2_8}(X^2_B(obs)) = 9.57\%, \\ p_C &= 1 - F_{\chi^2_8}(X^2_C(obs)) = 0.001\%, \end{aligned}$$

As we can see – p_C is very small, thus we would reject the hypothesis that the points from set C are realizations of uniformly distributed random variables. Note that this test takes into account the ordering of the points. Let C' be the points from set C which are randomly shuffled[6]. The following p-value was then obtained:

$$p_{C'} = 1 - F_{\chi^2_8}(X^2_{C'}(obs)) = 53.23\%.$$

Thus we have no reasons to reject the uniformity hypothesis. Such a p-value is aligned with intuition, the points from set C – as can be seen in Fig. 4.3

[6]Using `NumPy`'s `np.random.shuffle`

– are quite equally "distributed" on $[0, 1)$; thus, forgetting their ordering, the number of pairs within each $A_{s,t}$ should be similar. (By the way – they seem to be "too equally distributed" – what was spotted by both the KS and chi-square tests). The second-level testing (see Section 4.6) approach quickly spots the nonrandomness of points from C. □

4.3 Tests based on "balls and boxes" schemes

It is a group of tests based on the arrangement of r balls in k boxes. Balls and boxes are numbered from 1 to r and from 1 to k respectively. Our null hypothesis $\mathcal{H}_0$ is that the sequence of numbers to be tested (an output of a PRNG) is a realization of i.i.d. random numbers with the uniform distribution.

In this scheme, we assume that under $\mathcal{H}_0$, we know $p_i = 1/k$ – the probability of placing each ball (independently of all the other balls) in box $i, i = 1, \ldots, k$. The scheme is sketched in Fig. 4.10.

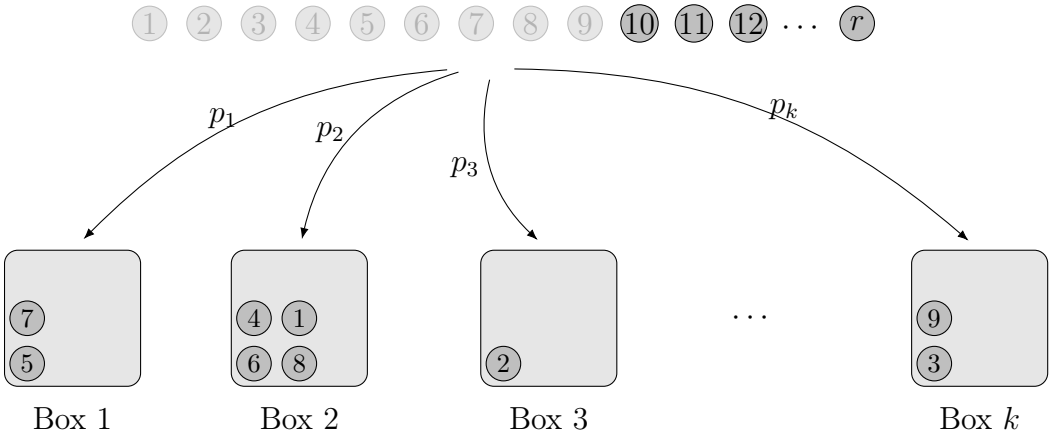


Figure 4.10: A general procedure of placing r balls into k boxes. Each ball is placed (independently of others) to box i with probability $p_i, i = 1, \ldots, k$. Nine of the balls were already placed in boxes.

We want to emphasize that "balls and boxes" is quite a general scheme. Choosing specific "balls" and "boxes" defines a test and how to convert a sequence (u_j) into a scheme of balls and boxes we will describe in Section 4.3.1 .

We will start with a classic anecdote.

Birthday problem. In Section 2 of Feller [40], a birthday problem is presented. Consider a group of r individuals (e.g., students) – we record their birthdays. Each birthday is a box ($k = 365$ of them), and each individual is a ball. Assuming all 365 possible birthdays are equally likely, we obtain the arrangement scheme of r balls into k boxes. The classic question is the probability of at least one **multiple-birth** situation, i.e. the probability that we have more than one ball in at least one box. Since the probability that all birthdays are different is $(365)_r/365^r$, thus the desired probability is

$$p = 1 - \frac{365!}{(365-r)!(365)^r} \, .$$

Equivalently,

$$\begin{aligned} p &= 1 - \frac{365 \cdot 364 \cdot \ldots \cdot (365-r+1)}{365^r} \\ &= 1 - \left(1 - \frac{1}{365}\right)\left(1 - \frac{2}{365}\right) \ldots \left(1 - \frac{r-1}{365}\right). \end{aligned}$$

It turns out that a group size of at least 23 is enough to get the probability that at least two individuals have birthdays on the same day is greater than $1/2$.

We can generalize this anegdote for general k and r. Then, the probability of a multiple-birth is

$$p_{k,r} = 1 - \prod_{j=1}^{r-1} \left(1 - \frac{j}{k}\right).$$

The following approximation is used

$$1 - p_{k,r} \approx 1 - \exp(-r^2/(2k)), \tag{4.12}$$

which is obtained in the following way. We have

$$\log(1 - p_{k,r}) = \sum_{j=1}^{r-1} \log\left(1 - \frac{j}{k}\right),$$

using $\log(1-x) \approx -x$ we get

$$\log(1 - p_{k,r}) \approx -\sum_{j=1}^{r-1} \frac{j}{k} = -\frac{(r-1)r}{2k},$$

hence (4.12) follows. The concept of a multiple-birth is translated into a collision - that is when balls are thrown sequentially one by one, and if a ball meets another one in a box, then a **collision** happened. Denote by $C_{k,r}$ the number of collisions after throwing r balls into k boxes. Note that $p_{k,r} = \mathbb{P}(C_{k,r} > 0)$. The above result can be turned into a formal test, however, typically its extensions are used: *collision test* or *birthday spacing test*, both described in due course.

4.3.1 Converting a sequence (u_j) into the scheme of balls and boxes.

Suppose we want to check whether a sequence (u_j), where $u_j \in [0, 1)$, fulfills the required property of randomness. Recall that it should possess one-dimensional uniformity, which can be verified by the Kolmogorov-Smirnov and chi-square tests. (see Section 4.2), however it should have some 'independence properties', which in principle can be verified by one of serial (m, L) tests. The problem is that in some cases (for eample for large L), the number of boxes is too large, and then we can use the scheme of balls and boxes.

As it was mentioned in Section 4.1.2 , the sequence is converted into a sequence of r vectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r$, which are our balls. On the other hand boxes are sub-hypercubes defined in (4.11).

Collision test. In terms of balls and boxes, tests like a serial one may be used when each box contains some (nontrivial) number of balls. In the case when the number of boxes is significantly larger than the number of balls (i.e., $k >> r$, as in the birthday problem), some other tests are recommended, in particular the so-called collision test. Assume we have Rn numbers from a PRNG. We split them into groups of n numbers. Further, having $n = mr$ numbers we split them into r vectors (m-dimensional), as in Section 4.1.2 . Below we will describe how to compute the number of collisions from $n = mr$ numbers. We define somehow boxes (e.g., as in serial test). We "randomly" place r balls in k boxes one after the other. The test is based on statistic $C_{k,r}$ (thus, later we will have values of R such statistics, $C_{k,r}^{(1)}, \ldots, C_{k,r}^{(R)}$), which is a number after placing r balls in k boxes, where a *collision* is defined as follows. After placing a ball, if the box was empty so far, we write down that it is occupied from now on. If, for some next ball, the chosen box is non-empty, a *collision* is recorded. It can be shown that the probability of

having exactly c collisions is as follows

$$\mathbb{P}(C_{k,r} = c) = \frac{k(k-1)\dots(k-r+c+1)S_2(r, r-c)}{k^r}, \tag{4.13}$$

where $S_2(r,s)$ is the Stirling number of the second kind. By $S_2(r,s)$ it is meant the number of ways to partition a set of r labelled objects into s non-empty non-labelled subsets. Look at the idea behind the formula (4.13). If the number of collisions is $c = 0$, then the pattern is

$$\{*\}, \{*\}, \dots, \{*\} - \ r \text{ subsets},$$

the number of collision is $c = 1$, then the pattern is

$$\{*,*\}, \{*\}, \dots, \{*\} - \ r-1 \text{ subsets},$$

the number of collision is $c = 2$, then the pattern is

$$\{*,*,*\}, \{*\}, \dots, \{*\} \quad \text{or} \quad \{*,*\}, \{*,*\}, \dots, \{*\} - \ r-2 \text{ subsets},$$

and so on. We can see that there are c collisions if and only if the set of balls are partitioned into $r-c$ subsets, that is, there are $S_2(r, r-c)$ of them. Partitions have to be inserted into k boxes. We can do it in $k(k-1)\dots(k-(r-c)+1)$ ways. The number of all possible placements of r balls into k boxes is k^r. Altogether it gives (4.13). We recommend the reader to complete the above arguments in Exercise II.T.24

Computing $\mathbb{P}(C = c)$ or $\mathbb{P}(C \le c)$ can be (computationally) hard; the approximations are used. At the end of this subsection, we sketch a proof of the following fact:

Proposition 4.6 *If $r, k \to \infty$ and $r^2/(2k) \to \lambda$, where $0 < \lambda < \infty$, then*

$$\mathbb{P}(C_{k,r} = c) \to \frac{\lambda^c}{c!} e^{-\lambda}, \qquad c = 0, 1, \dots \tag{4.14}$$

Note that for given values of r and k, the true value of λ may differ from $r^2/(2k)$ significantly – as we show later, the difference is of order $o\left(r^2/k\right)$. For example, for $k = 2^{20}$ (thus $L = 2, m = 20$) and $r = 2^{14}$ we have $r^2/2k = 128$. In practice, we do use the Poisson approximation; however, we do not use $r^2/(2k)$ as the approximation for λ, but we compute $\lambda = \mathbb{E}C_{k,r}$ for given k and r directly, as it can be derived relatively easy. Denote by $C^*_{k,r}$ the

Poisson random variable with parameter $\mathbb{E}C_{k,r}$, Below in (4.16) we show that $\mathbb{E}C_{k,r} = r - k\left(1-(1-\frac{1}{k})^r\right)$. For the aforementioned $k = 2^{20}$ and $r = 2^{14}$ using the formula we have $\lambda = \mathbb{E}C_{k,r} = 127.3282$. The values of $\mathbb{P}(C^*_{k,r} \leq c)$, as well as values of $\mathbb{P}(C_{k,r} \leq c)$ computed directly from (4.13) for several values of c are provided in Table 4.5.

c	101	108	119	126	134	145	153
$\mathbb{P}(C_{k,r} \leq c)$	.009	.043	.244	.476	.742	.946	.989
$\mathbb{P}(C^*_{k,r} \leq c)$	.0092	.0448	.2461	.4766	.7401	.9439	.9881
$\mathbb{P}(\widetilde{C}_{k,r} \leq c)$	.0078	.0396	.2281	.4530	.7205	.9367	.9860

Table 4.5: Exact values of $\mathbb{P}(C_{k,r} \leq c)$ and its Poisson approximations $\mathbb{P}(C^*_{k,r} \leq c)$ (with mean 127.3282) and $\mathbb{P}(\widetilde{C}_{k,r} \leq c)$ (with mean 128) for $k = 2^{20}$ and $r = 2^{14}$

For completeness, the values of $\mathbb{P}(\widetilde{C}_{k,r} \leq c)$ are also included in the table, where $\widetilde{C}_{k,r}$ is a random variable with Poisson distribution with mean $r^2/(2k) = 128$. Note that approximation of $C^*_{k,r}$ is a good approximation of $C_{k,r}$, in particular much better than $\widetilde{C}_{k,r}$.

Example 4.7 We proceed to use the collision test for a sequence of bits as follows. Note that we would compute only one number of collisions from r balls and k boxes. To proceed, we need to repeat the procedure several times and check whether the obtained values of the numbers of collisions are consistent with the actual distribution of $C_{k,r}$ (or its approximation $C^*_{k,r}$), what will be accomplished using the chi-square goodness-of-fit test. The recommendation is for $r = 2^{14}$, $k = 2^{20}$ and $L = 2$.

We could construct bins (and corresponding probabilities) using the Table 4.5, however we recommend to follow the suggestions provided in Koçak [125]. The bins (and their boundaries) as well as their exact probabilities are presented in Table 4.6 (for significantly larger values of r and k we suggest to use the approximation $C^*_{k,r}$)

box number s	nr of collisions	Probability p_s of box s
1	0-113	0.106253
2	114-118	0.109894
3	119-121	0.088373
4	122-124	0.100719
5	125-127	0.106608
6	128-130	0.104997
7	131-133	0.096632
8	124-137	0.106367
9	138-142	0.091574
10	≥ 143	0.088913

Table 4.6: 10 bins and their exact probabilities for collision test for $r = 2^{14}$ and $k = 2^{20}$.

The rule of thumb (4.9) for the chi-square goodness-of-fit test is to have (on average) at least 5 observations in the least probable box, which is 119-121 in our case. Thus, the number of experiments should be at least $\lceil \frac{5}{0.088373} \rceil = 56$. To perform the test, we therefore need to take $56 \cdot 2^{14} \cdot 20 = 18\,350\,080 = 2.29376$ MB bits from a PRNG. In other words, assume that we have a sequence of bits $b_1, \ldots, b_{Rn}$, where $n = mr$, with specific values $R = 56, m = 20, r = 2^{14}$. First, we split the bits into R subsequences, each of length mr, denote j-th sequence as $(b_1^{(j)}, \ldots, b_{20r}^{(j)})$. We further split the sequence into r 20-tuples (as done in (4.5))

$$\boldsymbol{b}_1^{(j)} = (b_1^{(j)}, \ldots, b_{20}^{(j)}), \quad \boldsymbol{b}_2^{(j)} = (b_{21}^{(j)}, \ldots, b_{40}^{(j)}),$$
$$\ldots, \ \boldsymbol{b}_r^{(j)} = (b_{20(r-1)+1}^{(j)}, \ldots, b_{20r}^{(j)}) \ ,$$

i.e., each $\boldsymbol{b}_i^{(j)} \in \{0,1\}^{20}$ represents a ball. Boxes are defined as $A_{q_1,\ldots,q_{20}} = (q_1, \ldots, q_{20})$, where $q_i \in \{0,1\}, i = 1, \ldots, 20$, there are $k = 2^{20}$ of them. For each $j = 1, \ldots, R = 56$ we compute the number of collisions $C_{k,r}^{(j)}$ followed by computing a number of collisions within each box:

$$O_s = \#\{j : C_{k,r}^{(j)} \text{ is in } s\text{-th box}\}$$

and

$$X^2(obs) = \sum_{s=1}^{10} \frac{(O_s - Rp_s)^2}{Rp_s},$$

where boxes and their probabilities p_s are presented in Table 4.6. Finally, we compute the corresponding p-value.

Note that we can represent boxes slightly differently. Assume again we are given a sequence of bits $b_1, \dots, b_{Rn}$ with $n = mr$. As previously $R = 56$ and $r = 2^{14}$. First, we split the bits into R subsequences, each of length mr. Denote j-th sequence as $(b_1^{(j)}, \dots, b_{mr}^{(j)})$. We treat consecutive pair of bits $(b_{2i-1}^{(j)}, b_{2i}^{(j)})$ as a vector from $\{0, 1, 2, 3\}$ and then we group these in sequences of length 10:

$$\boldsymbol{y}_1^{(j)} = (b_1^{(j)}, \dots, b_{10}^{(j)}), \quad \boldsymbol{y}_2^{(j)} = (b_{11}^{(j)}, \dots, b_{20}^{(j)}), \\ \dots, \ \boldsymbol{y}_r^{(j)} = (b_{10(r-1)+1}^{(j)}, \dots, b_{10r}^{(j)}) \ ,$$

In other words, we have $m = 10$, but still $k = 4^{10} = 2^{20}$, thus, precisely the same procedure can be applied. Note that we can still use the values in Table 4.6, the theoretical results are the same, but the value of the final p-value will be (most probably) different. □

Noticing that in Example 4.7 we have a Poissonian approximation with a large mean, one might consider a Central Limit Theorem approximation of $C_{r,k}$. For the scheme of balls and boxes with the number of empty boxes $E_{k,r}$, there is a known version of the Central Limit Theorem; see, e.g., [130].

We will provide more details at the end of the following section.

Theoretical results on the collision test. Let us enumerate boxes with consecutive integers $1, \dots, k$. Suppose X_i is the number of balls in the i-th box. We have $\mathbb{P}(X_i = 0) = (1 - \frac{1}{k})^r$ and

$$\begin{aligned} C_{k,r} &= \sum_{i=1}^{k} (X_i - 1)\mathbf{1}(X_i \geq 1) \\ &= \sum_{i=1}^{k} X_i \mathbf{1}(X_i \geq 1) - \sum_{i=1}^{k} \mathbf{1}(X_i \geq 1) \\ &= r - k + E_{k,r}, \end{aligned} \tag{4.15}$$

where $E_{k,r} = \#\text{empty boxes}$. Hence

$$\mathbb{E}C_{k,r} = r - \mathbb{E}\#\{\text{non-empty boxes}\} = r - k\left(1 - (1 - \tfrac{1}{k})^r\right). \tag{4.16}$$

Now expand

$$\left(1-\frac{1}{k}\right)^r = 1 - r\frac{1}{k} + \binom{r}{2}\frac{1}{k^2} + o\left(\frac{r^2}{k^2}\right),$$

and hence

$$\mathbb{E}C_{k,r} = r - k\left(1 - \left(1 - \frac{r}{k} + \binom{r}{2}\frac{1}{k^2} + o\left(\frac{r^2}{k^2}\right)\right)\right) \sim \frac{r^2}{2k} + o\left(\frac{r^2}{k}\right).$$

For $r = 2^{14}$ and $k = 2^{20}$ we obtain $\mathbb{E}C_{k,r} \approx \frac{r^2}{2k} = 128$, while the true value is $\lambda = 127.3282$.

We now sketch a proof of Poissonian asymptotics (4.14). We will need the following Lemma, which can be found in Durrett [35].

Lemma 4.8 *Suppose that for an array $(c_{r,j})$ we have $\max_{1\le j\le r}|c_{r,j}| \to 0$, $\sum_{j=1}^r c_{r,j} \to \lambda$, and $\sup_r \sum_{j=1}^r |c_{r,j}| < \infty$. Then*

$$\prod_{j=1}^{r}(1 + c_{r,j}) \to e^{\lambda}.$$

Lemma 4.9 *Suppose that $\frac{r^2}{2k} \to \lambda$. Then*

$$\frac{k(k-1)\dots(k-r+c+1)}{k^{r-c}} \to e^{-\lambda}\,.$$

Proof We have

$$\frac{k(k-1)\dots(k-r+c+1)}{k^{r-c}} = \prod_{i=1}^{r-c-1}\left(1-\frac{i}{k}\right).$$

Using Lemma 4.8 with $c_{r-c-1,j} = -\frac{i}{k}$ we have

$$\sum_{i=1}^{r-c-1} c_{r-c-1,j} = -\tfrac{1}{k}\sum_{i=1}^{r-c-1} i = -\tfrac{1}{k}\frac{(r-c-1)(r-c)}{2} \sim -\frac{r^2}{2k}.$$

To complete the proof of (4.13) we will use the following asymptotics for $S_2(r, r-c)$:

$$S_2(r, r-c) \sim \frac{(r-c)^{2c}}{2^c c!}.$$

Hence, assuming $\frac{r^2}{2k} \to \lambda$,

$$\frac{S_2(r,r-c)}{k^c} \sim \left(\frac{(r-c)^2}{(2k)^c}\right)\frac{1}{c!} \to \frac{\lambda^c}{c!}.$$

□

Recall that $E_{k,r}$ denotes the number of empty boxes after a scheme of balls and boxes is applied. We compute the mean and variance of $E_{k,r}$. Consider an i-th box and a single throw of a ball. The probability that it it does not fall into the i-th box is $1-1/k$, and after the series of r balls, the probability that the i-th box is empty is $(1-1/k)^r$. Denote

$$a_1 = (1-1/k)^r, \qquad a_2 = (1-2/k)^r.$$

We introduce

$$Y_i = \begin{cases} 1 & \text{the } i\text{-box is empty,} \\ 0 & \text{otherwise,} \end{cases}$$

then

$$\mathbb{E}E_{k,r} = \mathbb{E}\sum_{i=1}^{k} Y_i = k(1-1/k)^r = ka_1.$$

For the variance we must know that $\mathrm{Var}Y_i = a_1 - a_1^2$ and

$$\mathrm{Cov}\,(Y_i, Y_j) = \mathbb{E}[Y_iY_j] - (\mathbb{E}Y_i)^2 = (1-2/k)^r - (1-1/k)^{2r}.$$

Hence

$$\begin{aligned} \mathrm{Var}\sum_{i=1}^{k} Y_i &= \sum_{i=1}^{k} \mathrm{Var}Y_i + \sum_{i\neq j} \mathrm{Cov}\,(Y_i, Y_j) \\ &= k(a_1 - a_1^2) + (k^2-k)(a_2 - a_1^2). \end{aligned}$$

We will consider a limiting approximation for the distribution of $E_{k,r}$ in the following regime:

(A^*) $r \to \infty$ and $k = \alpha r$ for some $\alpha > 0$.

Notice that if (A^*) holds, then

$$e_{k,r} = \mathbb{E}E_{k,r} = k\left(1 - \frac{\alpha}{k}\right) \sim ke^{-\alpha}, \tag{4.17}$$

$$\begin{aligned} \sigma^2_{k,r} &= \mathrm{Var}E_{k,r} = k(a_1 - a_1^2) + (k^2-k)(a_2 - a_1^2) \\ &\sim ke^{-\alpha}(1 - (1+\alpha)e^{-\alpha}). \end{aligned} \tag{4.18}$$

We leave the proof of these asymptotics in Exercise II.T.25. We have the following form of the Central Limit Theorem:

Proposition 4.10 *[Weiss [130]] Under condition (A*) we have*

$$\frac{E_{k,r} - e_{k,r}}{\sigma_{k,r}} \xrightarrow{\mathcal{D}} \mathcal{N}(0, e^{-\alpha}(1 - (1+\alpha)e^{-\alpha})).$$

Using the asymptotics (4.17) and (4.18) and the relation (4.15), noticing that variances of $C_{k,r}$ and $E_{k,r}$ are the same, we can write

$$\frac{E_{k,r} - ke^{-\alpha}}{\sqrt{k}} \xrightarrow{\mathcal{D}} \mathcal{N}(0, e^{-\alpha}(1 - (1+\alpha)e^{-\alpha}))$$

and

$$\frac{E_{k,r} - r + ke^{-\alpha}}{\sqrt{k}} \xrightarrow{\mathcal{D}} \mathcal{N}(0, e^{-\alpha}(1 - (1+\alpha)e^{-\alpha})).$$

The above normal approximation gives the possibility of testing generators in regimes other than $r^2/2k \to \lambda$; namely $k/r \to \alpha$. Examples of the use of such the regime can be found in the work of Tsang et al [126].

Birthday spacing test. Another well-known method for testing the quality of PRNG is the so-called *birthday spacing test*. As its author Marsaglia [92] writes, the test is difficult for many generators to pass. Assume we have Rn numbers from a PRNG. We split them into groups of n numbers. Further, having $n = mr$ numbers we split them into into r vectors (m-dimensional), as in Section 4.1.2 . It is based on the K statistic defined as follows (thus, later, we will have R values of the statistics, $K^{(1)}, \ldots, K^{(R)}$). Below we describe how to compute one statistic K for given $n = mr$ numbers, from which we construct $Y_1, \ldots, Y_r$ vectors (m-dimensional), each $Y_i \in [k] = \{1, \ldots, k\}$. We arrange them in a non-decreasing order $Y_{(1)} \leq \ldots \leq Y_{(r)}$. Then we define the so-called *spacings* by

$$S_1 = Y_{(2)} - Y_{(1)}, \ldots, S_{r-1} = Y_{(r)} - Y_{(r-1)}, S_r = k - Y_{(r)} + Y_{(1)},$$

which we sort again in the non-decreasing order $S_{(1)}, \ldots, S_{(r)}$. Then K is the number of equal spacings, i.e.

$$K = \#\{j \in \{1, \ldots, r\} : S_{(j-1)} = S_{(j)}\}.$$

For an illustration in Table 4.7, we generated 14 birthdays in a one-hundred-day year.

	birthday	birthday sorted	spacings	spacings sorted
i	Y_i	$Y_{(i)}$	S_i	$S_{(i)}$
1	92	4	14	0
2	80	18	22	1
3	96	40	26	$\underline{2}$
4	66	66	0	$\underline{2}$
5	4	66	2	$\underline{2}$
6	85	68	3	3
7	94	71	4	$\underline{4}$
8	68	75	1	$\underline{4}$
9	76	76	4	5
10	75	80	5	7
11	40	85	7	8
12	66	92	2	14
13	18	94	2	22
14	71	96	8	26

Table 4.7: Birthday spacings example: $r = 14, k = 100$.

In this example, we have two groups with equal spacings: 2,2,2 and 4,4, thus $K = 2 + 1 = 3$. In Fig. 4.11, we gave another example illustrating the spacing in a circle.

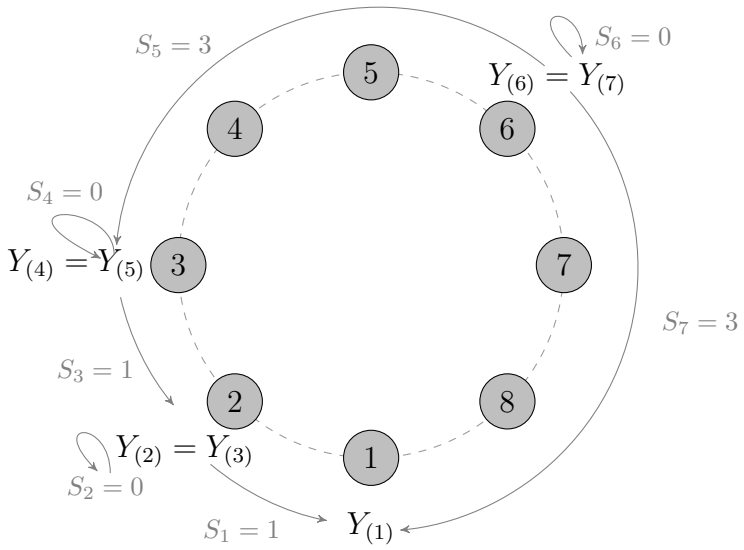


Figure 4.11: Case: $k = 8, r = 7$, $Y_1 = 2, Y_2 = 3, Y_3 = 3, Y_4 = 6, Y_5 = 2, Y_6 = 1, Y_7 = 6$.

For the birthday spacings test, the exact distribution is difficult to obtain. It turns out that for large r, k, if the value of $\lambda = r^3/(4k)$ is small, then assuming randomness hypothesis $\mathcal{H}_0$, K has approximately Poisson distribution with a parameter λ. [7] We denote this Poisson random variable as K^*. Knuth [71] (p. 72) provided exact values of $\mathbb{P}(K = s)$ for $s \leq 2$ for $r = 2^9 = 512$, $k = 2^{25}$ (which gives $\lambda = 1$). The values (together with a Poisson approximation K^* of K) are provided in Table 4.8.

$s =$	0	1	2	≥ 3
$\mathbb{P}(K = s)$	0.3688016	0.3690335	0.1834712	0.786920
$\mathbb{P}(K^* = s)$	0.3679	0.3679	0.1839	0.8003

Table 4.8: Exact values of $\mathbb{P}(K = s)$ and its Poisson approximations $\mathbb{P}(K^* = s)$ (with mean $\lambda = r^3/(4k) = 1$) for $k = 2^{25}$ and $r = 2^9$.

[7] It turns out that the error with this approximation is of order $O(\lambda^2/r)$. In the paper of L'Ecuyer and Simarda [87] it is suggested that $r^3 \leq k^{5/4}$.

This Poissonian approximation can be inferred from Exercises 3.3.2-28-30 in Knuth [71].

Testing procedure for $r = 2^9, k = 2^{25}$. Assume we have a sequence of bits $b_1, \ldots, b_{Rn}$ and $n = mr$, where $m = 25$ and $r = 2^9$. As in the collision test, we split the bits into R subsequences, each of length mr. Denote j-th sequence as $(b_1^{(j)}, \ldots, b_{mr}^{(j)})$. We further split them into r 25-tuples

$$\boldsymbol{y}_1^{(j)} = (b_1^{(j)}, \ldots, b_{25}^{(j)}), \quad \boldsymbol{y}_2^{(j)} = (b_{26}^{(j)}, \ldots, b_{50}^{(j)}), \\ \ldots, \ \boldsymbol{y}_r^{(j)} = (b_{25(r-1)+1}^{(j)}, \ldots, b_{25r}^{(j)}) \ ,$$

We thus have $r = 2^9$ birthdays (balls) within $k = 2^{25}$ days (boxes). For $\boldsymbol{y}_1^{(j)}, \ldots, \boldsymbol{y}_r^{(j)}$ we compute the value of the statistic $K^{(j)}$. Finally, performing it for R sequences, we compute the number of values of these statistics within boxes $A_0 = \{0\}, A_1 = \{1\}, A_2 = \{2\}, A_3 = \{3, 4, \ldots, \}$:

$$O_s = \#\{j : K^{(j)} \text{ is in box } A_s\}, \quad s = 0, 1, 2, 3.$$

As usual, we compute

$$X^2(obs) = \sum_{s=0}^{3} \frac{(O_s - Rp_s)^2}{Rp_s},$$

where probabilities $p_s = \mathbb{P}(K = s)$ are provided in Table 4.8. Under $\mathcal{H}_0$, the above statistic should have χ^2 distribution with 3 degrees of freedom, thus we may compute the corresponding p-value. Marsaglia and Tsang [92] recommend other values: $r = 2^{12}$, $k = 2^{32}$, $\lambda = 4$ or $r = 2^{10}, k = 2^{24}, \lambda = 16$.

Now we proceed to tests based on the classical combinatorial problems of probability.

Poker test. Similarly, as in the frequency of pairs test, we consider a sequence of numbers $y_1, y_2, \ldots, y_n \in \bar{M}$, where $n = 5r$, which – assuming $\mathcal{H}_0$ – is a realization of a sequence of i.i.d. random variables $Y_1, Y_2, \ldots$ with the uniform distribution $\mathcal{U}[\bar{M})$. We group the numbers into r 5-tuples (similarly as in (4.3))

$$\boldsymbol{y}_1 = (y_1, \ldots, y_5), \quad \boldsymbol{y}_2 = (y_6, \ldots, y_{10}), \\ \ldots, \ \boldsymbol{y}_r = (y_{(r-1)5+1}, \ldots, y_{5r}) \ .$$

Following [70, p. 63], each 5-tuple $\boldsymbol{y}_i$ can be classified as one of the following simplified pokers ranks:

5 different = all different,

4 different = one pair,

3 different = two pairs or three of a kind,

2 different = full or four of a kind,

1 different = five of a kind.

In our boxes and balls notation, $\boldsymbol{y}_i$'s are balls, and we have $k = 5$ boxes (s-box corresponds to "s different"). Note that each possibility appears in some options. For actual poker ranks, we recommend Exercise II.T.18; however using them for testing seems to be troublesome.

We need to compute the theoretical probability p_s, that a 5-tuple $(Y_1, \ldots, Y_5)$ consists of s different (as defined above) numbers ($s = 1, \ldots, 5$) – under $\mathcal{H}_0$ i.e., assuming that Y_i is chosen uniformly from $\bar{M}$. Let $S_2(5, s)$ denote the number of possible ways to partition a 5-element set into s nonempty subsets (these are so-called Stirling numbers of the second kind). We leave it to the reader to show that $S_2(5, 1) = 1, S_2(5, 2) = 15, S_2(5, 3) = 25, S_2(5, 4) = 10, S_2(5, 5) = 1$. Since the number of different s element sequences is $M(M-1)\cdots(M-s+1)$, the desired probability is equal to

$$p_s = \frac{M(M-1)\cdots(M-s+1)S_2(5,s)}{M^5}.$$

Summarising, we compute the number of 5-tuples within each box

$$O_s = \#\{j : \boldsymbol{y}_j \in \{s \text{ different}\}\}$$

and compute the statistic

$$X^2(obs) = \sum_{s=1}^{5} \frac{(O_s - rp_s)^2}{rp_s},$$

which under $\mathcal{H}_0$ has the χ^2 distribution with 4 degrees of freedom.

4.4 Tests based on random walks.

Let $b_1, b_2, \ldots$ be a sequence of bits (i.e., $b_i \in \{0,1\}$), which under $\mathcal{H}_0$ is a realization of a sequence of random bits $B_1, B_2, \ldots$. Denote

$$X_i = 2B_i - 1,\ S_0 = 0,\ S_k = \sum_{i=1}^{k} X_i,\ k = 1, \ldots \tag{4.19}$$

The sequence X_i is $\{-1, +1\}$ valued ($\mathbb{E}X_i = 0, \mathrm{Var}X_i = 1$), the process (S_k) is called a *symmetric random walk*. We have already discussed some properties of (S_k) in Section I.2.1.

Frequency (monobits) test. From n bits $B_1, \ldots, B_n$ we compute X_i and S_i, $i = 1, \ldots, n$ as in (4.19). Under the randomness hypothesis $\mathcal{H}_0$, from the central limit theorem, we have that $S_n/\sqrt{n}$ is approximately normal $\mathcal{N}(0,1)$, and $T = |S_n/\sqrt{n}|$ has the so-called half-normal distribution function G, namely

$$G(z) = \lim_{n\to\infty} \mathbb{P}\left(\left|\frac{S_n}{\sqrt{n}}\right| \le z\right) = \Phi(z) - \Phi(-z),$$

hence

$$G(z) = \frac{1}{\sqrt{2\pi}} \int_{-z}^{z} e^{-\frac{x^2}{2}}\, dx = \frac{2}{\sqrt{\pi}} \int_{0}^{\frac{z}{\sqrt{2}}} e^{-y^2}\, dy = \mathrm{erfc}\left(\frac{z}{\sqrt{2}}\right)\ .$$

For a sequence $b_1, b_2, \ldots, b_n$ one computes $s_n = \sum_{j=1}^{n} x_j$, where $x_j = 2b_j - 1$ and $T(obs) = |s_n/\sqrt{n}|$. Then the p-value is computed from the formula

$$p = \mathrm{erfc}\left(\frac{T(\mathrm{obs})}{\sqrt{2}}\right).$$

Frequency (monobit) test in blocks. Consider a sequence of random bits $B_1, B_2, \ldots, B_n$ of length $n = mr$ splitted into r blocks, each of length m. The idea is a kind of extension of the frequency (monobit) test – we will use the CLT applied to a simple random walk within each block. To be more exact, let $S_m^{(i)}$ be the value of a random walk at step m within block i, i.e.,

$$S_m^{(i)} = \sum_{j=1}^{m} X_{(i-1)m+j} = \sum_{j=1}^{m} (2B_{(i-1)m+j} - 1), \quad i = 1, \ldots, r.$$

Clearly, $\mathbb{E}S_m^{(i)} = 0$ and $\mathrm{Var}S_m^{(i)} = m$. Let

$$Z_i = \frac{S_m^{(i)}}{\sqrt{m}} = \frac{1}{\sqrt{m}} \sum_{j=1}^{m} (2B_{(i-1)m+j} - 1), \quad i = 1, \ldots, r.$$

For m large enough, one may assume that random variables $Z_1, \ldots, Z_r$ are i.i.d. $\mathcal{N}(0,1)$-distributed, hence

$$\frac{1}{m} \sum_{i=1}^{r} \left(S_m^{(i)}\right)^2 =_{\mathrm{d}} \sum_{i=1}^{r} Z_i^2$$

has under $\mathcal{H}_0$ the chi-square distribution with r degrees of freedom. To summarise, for a given sequence of bits $b_1, b_2, \ldots, b_n$ (with $n = mr$) we compute

$$T(obs) = \frac{1}{m} \left(s_m^{(i)}\right)^2 = \frac{1}{m} \sum_{i=1}^{r} \left(\sum_{j=1}^{m} (2b_{(i-1)m+j} - 1) \right)^2$$

and the corresponding p-value.

Arcsine law based test. For a random walk (S_k) define

$$D_k = \mathbf{1}(S_k > 0 \vee S_{k-1} > 0), \quad k = 1, 2, \ldots,$$

i.e., D_k is equal to 1 if the number of ones exceeds the number of zeros at step k or step $k-1$, and 0 otherwise. Consider a sequence $B_1, \ldots, B_n$ of even length n. Recall that the number of segments that lie above the x-axis is denoted by $L_{2r}^+ = \sum_{j=1}^{2r} D_j$. The arcsine law states that for any $w \in [0,1)$ we have

$$\lim_{r \to \infty} \mathbb{P}\left(\frac{L_{2r}^+}{2r} \leq w \right) = \frac{1}{\pi} \int_0^w \frac{dt}{\sqrt{t(1-t)}} = \frac{2}{\pi} \arcsin \sqrt{w}.$$

The probability $\mathbb{P}(L_{2r}^+ \leq w \cdot 2r)$ is the chance that the random walk was above x-axis for at most w fraction of the time – the limiting distribution (see Fig. 4.12 for this distribution and the corresponding c.d.f, cf. with the previous Fig. I.2.3 for a histogram of L_{1000}^+) indicates that the fractions of time spent above and below the x-axis are more likely to be unequal than close to each other. It may be against the intuition – since on average $\mathbb{E}S_k = 0$ –

the thing is that this expectation is over paths. One path will most likely be over or below the x-axis most of the time.

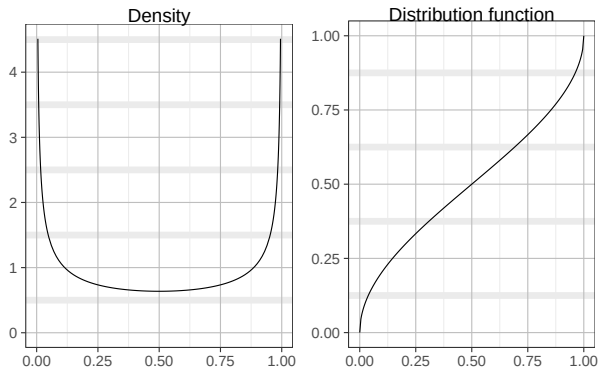


Figure 4.12: Density function $\frac{2}{\pi}\arcsin\sqrt{w}$ and the corresponding c.d.f.

Given a sequence of bits $b_1, b_2, \ldots, b_n$ (for even n) from a PRNG we compute

$$s_k = \sum_{i=1}^{k} x_i,\ x_i = 2b_i - 1, \quad d_k = \mathbf{1}(s_k > 0 \vee s_{k-1} > 0), \quad k = 1, \ldots, n$$

and finally, the fraction of time instants at which the number of ones exceeds the number of zeros

$$T(\text{obs}) = \frac{1}{n}\sum_{k=1}^{n} d_k.$$

Under the hypothesis $\mathcal{H}_0$ (that $b_1, b_2, \ldots, b_n$ is a realization of the Bernoulli process), the limiting distribution of $T_n = \frac{1}{n}\sum_{k=1}^{n} d_k$ is provided by the arcsine law, thus we compute the corresponding p-value in the following way:

$$p = \mathbb{P}(T_n > T(\text{obs})) \approx 1 - \frac{2}{\pi}\arcsin\sqrt{T(\text{obs})}.$$

The detailed analysis, as well as the second-level (see Section 4.6) test based on the arcsine law can be found in [86]. Also the analysis of errors in approximations is provided therein.

Random excursion test. For a sequence of $\{0, 1\}$-valued random variables $B_1, \ldots, B_n$ let S_k denote the corresponding random walk (4.19) and let

J_n to be the number of zeros among S_k, $k = 1, \dots, n$. Assuming the randomness hypothesis $\mathcal{H}_0$, i.e., that $B_1, \dots, B_n$ is a realization of a Bernoulli process; it is known that

$$\lim_{n\to\infty} \mathbb{P}\left(\frac{J_n}{\sqrt{n}} \le t\right) = \sqrt{\frac{2}{\pi}} \int_0^t e^{-x^2/2}\, dx. \tag{4.20}$$

According to NIST report [120], the test rejects the randomness if J_n is too small. Let t_0 be such that

$$0.01 \approx \sqrt{\frac{2}{\pi}} \int_0^{t_0} e^{-x^2/2}\, dx.$$

We have approximately $t_0 = 0.01253346951$. It means that we have an example of a one-sided left-tail test. We recommend to use p-values and test them accordingly.

Hence, for a given sequence of bits $b_1, \dots, b_n$ we compute the corresponding $s_0 = 0, s_k = \sum_{i=1}^{k}(2b_i - 1), k = 1, \dots, n$ and the number of zeros it these sums, denote it by $J_n(\text{obs})$. If $J_n(\text{obs}) \le 0.008\sqrt{n}$, the randomness conjecture is rejected. If $J_n(\text{obs})$ is the observed number of excursions, then the p-value is

$$\mathbb{P}(J_n \le J(\text{obs})) = P\left(\frac{1}{2}, \frac{J_n^2(\text{obs})}{2n}\right),$$

where $P(a, x) = \frac{\gamma(a,x)}{\Gamma(a)}$ and $\gamma(a, x) = \int_0^x t^{a-1}e^{-t}\, dt$.

A piece of the random walk between a visit to zero before the next return to zero is called an excursion; that is if $S_k = 0$, and $S_{k+1} \neq 0$, $S_{k+2} \neq 0$, ... $S_{l-1} \neq 0$ and $S_l = 0$, then the excursion is $S_{k+1}, \dots, S_{l-1}$. The 0-th excursion is, when $k = 0$. Let $L(x)$ be the number of visits to x during the 0-th excursion. Then

$$\mathbb{P}(L(x) = k) = \begin{cases} 1 - \frac{1}{2|x|} & \text{for } k = 0, \\ \frac{1}{4x^2}\left(1 - \frac{1}{2|x|}\right)^{k-1} & \text{for } k \ge 1. \end{cases} \tag{4.21}$$

As an exercise, we leave the reader to verify

$$\mathbb{E}L(x) = 1, \qquad \text{Var}L(x) = 4|x| - 2$$

and

$$\mathbb{P}(L(x) \ge a+1) = 2|x|\mathbb{P}(L(x) = a+1) = \frac{1}{2|x|}\left(1 - \tfrac{1}{2|x|}\right)^a, \qquad a = 0, 1, \dots.$$

	$\pi_0(x)$	$\pi_1(x)$	$\pi_2(x)$	$\pi_3(x)$	$\pi_4(x)$	$\pi_5(x)$
x=1	.5000	.2500	.1250	.0625	.0312	.0312
x=2	.7500	.0625	.0469	.0352	.0264	.0791
x=3	.8333	.0278	.0231	.0193	.0161	.0804
x=4	.8750	.0156	.0137	.0120	.0105	.0733
x=5	.9000	.0100	.0090	.0081	.0073	.0656
x=6	.9167	.0069	.0064	.0058	.0053	.0588
x=7	.9286	.0051	.0047	.0044	.0041	.0531

Table 4.9: Frequencies of number of visits in x

Define

$$\begin{aligned}
\pi_0(x) &= \mathbb{P}(L(x)=0) = 1 - \frac{1}{2|x|}, \\
\pi_k(x) &= \mathbb{P}(L(x)=k) = \frac{1}{4x^2}\left(1-\frac{1}{2|x|}\right)^{k-1}, \\
\pi_5(x) &= \mathbb{P}(L(x)\geq 5) = \frac{1}{2|x|}\left(1-\frac{1}{2|x|}\right)^{4}.
\end{aligned}$$

For a representative collection of x-values (for example NIST recommends among others $1 \leq x \leq 7$) the values of $\pi_i(x), i = 0, \ldots, 7$ are presented in Table 4.9. For a given input we record $J_n(obs)$ excursions and we count $E_k(x)$, the number of excursions with k visits to x and for each x we compute

$$X^2(obs) = \sum_{k=1}^{5} \frac{(E_k(x) - J_n(obs)\pi_k(x))^2}{J_n(obs)\pi_k(x)},$$

which under the randomness hypothesis $\mathcal{H}_0$ has a chi-square distribution with 5 degrees of freedom. Note that n must be large enough to assure that $J_n(obs)\pi_k(x) \geq 5$ for all k and x, that is $J_n(obs) \geq 1300$. We leave it to the reader to estimate how long the input has to be.

4.5 Permutation tests

Derangement Permutation Test. Consider a random permutation σ of $[m] = \{1, 2, .., m\}$ and define $X = \#\{i : \sigma(i) = i\}$. By the derangement of a

permutation we mean a permutation of the elements of a set, such that no element appears in its original position. In other words, a derangement is a permutation with no fixed points. Let $\mathcal{D}_m$ denote a set of all derangement's of $\{1, 2, .., m\}$, denote $d_m = |\mathcal{D}_m|$.

Lemma 4.11 $d_m = \sum_{k=0}^{m} (-1)^k \binom{m}{k} (m-k)!$.

Proof Recall the inclusion-exclusion principle. For any sets $A_1, \ldots, A_m$ we have

$$\left| \bigcup_{i=1}^{m} A_i \right| = \sum_{k=1}^{m} (-1)^{k+1} \left(\sum_{1 \le i_1 < \cdots < i_k \le m} |A_{i_1} \cap \cdots \cap A_{i_k}| \right).$$

First, we will calculate a number of permutations of $\{1, 2, .., m\}$ which have *at least* one fixed point, denote it by d'_m. Let A_i be a set of permutations σ of $\{1, 2, .., m\}$ such that $\sigma(i) = i$. For fixed $1 \le i_1 < \cdots < i_k \le m$ we have $|A_{i_1} \cap \cdots \cap A_{i_k}| = (m-k)!$. Thus

$$d'_m = \left| \bigcup_{i=1}^{n} A_i \right| = \sum_{k=1}^{m} (-1)^{k+1} \left(\sum_{1 \le i_1 < \cdots < i_k \le m} (m-k)! \right) = \sum_{k=1}^{m} (-1)^{k+1} \binom{m}{k} (m-k)!$$

and

$$d_m = m! - d'_m = \sum_{k=0}^{m} (-1)^k \binom{m}{k} (m-k)!$$

□

Note that d_m can be rewritten as

$$d_m = m! \sum_{k=2}^{m} (-1)^k \frac{1}{k!}.$$

Thus, the probability that a random permutation has no fixed points (i.e., the probability of a derangement) is

$$\frac{d_m}{m!} = \sum_{k=2}^{m} (-1)^k \frac{1}{k!},$$

which is close to e^{-1}, since from Taylor's formula

$$\left| e^{-1} - \sum_{j=2}^{m} (-1)^j \frac{1}{j!} \right| \le \frac{1}{(m+1)!}.$$

A rough error is of order less than 2^{-m}.

Thus we propose the following test. Consider a sequence $u_1, u_2, \dots u_n$, of length $n = mr$. We split the sequence (as earlier in (4.3)) into r vectors being m-tuples:

$$\begin{aligned} \boldsymbol{u}_1 = (u_1, \dots, u_m), \quad & \boldsymbol{u}_2 = (u_{m+1}, \dots, u_{2m}), \\ & \dots, \ \boldsymbol{u}_r = (u_{(r-1)m+1}, \dots, u_{rm}) \ . \end{aligned}$$

Using Algorithm 1 (Fisher-Yates algorithm) from Section 2.1, for each r-tuple $\boldsymbol{u}_i = (u_{(i-1)m+1}, \dots, u_{im})$ $i = 1, \dots, r$ we compute the pseudorandom permutation σ^i of $\{1, 2, .., m\}$. For the i-th random permutation we set

$$\xi_i = \begin{cases} 1 & \text{if } \ \sigma^i \text{ is a derangement,} \\ 0 & \text{otherwise.} \end{cases}$$

Under the randomness hypothesis $\mathcal{H}_0$ that $u_1, \dots, u_n$ is a realization of an i.i.d. sequence of $\mathcal{U}[0,1)$ random variables, we have that ξ_i are i.i.d. and $\mathbb{P}(\xi_i = 1) = e^{-1}$. Thus we perform a variant of the monobit test computing

$$T(\text{obs}) = \left| \frac{\left(\sum_{j=1}^{r} \xi_j \right) - re^{-1}}{\sqrt{re^{-1}(1 - e^{-1})}} \right|$$

and the corresponding p-value from the formula

$$\operatorname{erfc}\left(\frac{T(\text{obs})}{\sqrt{2}} \right),$$

where erfc is the complementary error function (see Exercise II.T.9).

Derangement Permutation Test is given in Algorithm 5. Recall, Fisher-Yates was given in Algorithm 1 in Section 2.1.

Algorithm 5 Derangement Permutation Test

Input: $\boldsymbol{u} = (u_1, u_2, \dots u_n), n = mr$

1: $\xi = 0$
2: **for** $i = 1$ to r **do**
3: $\quad \sigma = \mathsf{Fisher\text{-}Yates}(m, (u_{(i-1)m+1}, \dots, u_{im}))$
4: $\quad$ **if** σ is a derangement **then**
5: $\quad\quad \xi = \xi + 1$
6: $\quad$ **end if**
7: **end for**
8: $T(\text{obs}) = \frac{\left|\xi - re^{-1}\right|}{\sqrt{re^{-1}(1-e^{-1})}}$
9: **Return** p-value $= \text{erfc}\left(\frac{T(\text{obs})}{\sqrt{2}}\right)$

Fixed point permutation tests. For a random permutation σ of $\{1, \dots, m\}$ let X be defined like in the derangement permutation test (i.e., the number of fixed points in a random permutation). We have

$$\begin{aligned} P(X = k) &= \frac{1}{m!} \# \cdot \{\sigma \in \mathcal{S}_m : \sigma \text{ has exactly } k \text{ fixed points}\} \\ &= \frac{1}{m!}\binom{m}{k} d_{m-k} \to \frac{1}{m!}\frac{m!}{k!(m-k)!}(m-k)!e^{-1} = \frac{1}{k!}e^{-1}, \end{aligned}$$

where in $\to$ we used fact that $d_{m-k} \to (m-k)!e^{-1}$ as $m \to \infty$. It means that $X \xrightarrow{\mathcal{D}} N$ as $m \to \infty$, where N is a Poisson random variable with parameter 1. Similarly as in derangement test, we split a sequence $u_1, u_2, \dots u_n$ of length $n = mr$ into r vectors $\boldsymbol{u}_1, \dots, \boldsymbol{u}_r$, each being an m-tuple. From each $\boldsymbol{u}_i$ we construct a permutation $\sigma^i, i = 1, \dots, r$ via the Fisher-Yates algorithm (Algorithm 1) from Section 2.1. Define

$$\xi_i = \#\{j : \sigma^i(j) = j\},$$

i.e., the number of fixed points of σ_i. Consider boxes

$$A_0 = \{0\}, A_1 = \{1\}, \dots, A_9 = \{9\}, A_{10} = \{10, 11, \dots\}$$

and compute $O_j, j = 0, \dots, 10$, the number of ξ_i (i.e., the number of corresponding vectors $\boldsymbol{u}_i$) which are in box A_j, i.e.,

$$O_j = \#\{i \in \{1, \dots, r\} : \xi_i \in A_j\}.$$

Under the randomness hypothesis we should expect rp_j permutations (thus vectors) having j fixed points for $j = 0, \ldots, 10$, where $p_j = e^{-1}/j!, j = 0, \ldots, 9$ and $p_{10} = 1 - \sum_{j=0}^{9} p_j$. We test it using the chi-square test computing the statistic

$$X^2(obs) = \sum_{j=0}^{10} \frac{(O_j - rp_j)^2}{rp_j}$$

and the corresponding p-value (assuming $\mathcal{H}_0$ the statistic has X^2 distribution with 10 degrees of freedom).

4.6 p-values and second-level testing

More on p-values. This section will provide more insight and details for already mentioned and used p-values. Let us start informally. Assume that some PRNG returns n bits. Our task is to check whether they are "random". Assuming they are indeed random, we may think they are the results of flipping a fair coin n times. Under this assumption, any sequence is equally likely, each having probability $1/2^n$. Say $n = 100$, and suppose we observed all zeros. It could happen, but the chance of observing all zeros is $7.88 \cdot 10^{-31}$, i.e., it is very unlikely. That is why we would say that this PRNG is not good, and we would be correct with a very large probability.

The above example leads to the following idea of testing: *beforehand* we decide on some event for which we know the probability, which is small (some predefined number α, usually 0.05 or 0.01, called the *significance level*). Then, if this event occurs based on the output of a PRNG, we say that *the randomness hypothesis that this PRNG produces random numbers is rejected*. Returning to the above example: if we observe $n = 100$ zeros, we would say the PRNG is bad, since it has a tiny probability. All zeros seem too restrictive; less restrictive could be:

- $E = \{$there will be at least 63 zeros$\}$. We may compute

$$\mathbb{P}(E) = \sum_{k=63}^{100} \binom{100}{k} \frac{1}{2^{100}} = 0.00601.$$

Then, if we fix, say $\alpha = 0.01$ and there are at least 63 ones in 100 bits, we would state that the PRNG is not good – since the probability of having this

event was $0.00601 \leq \alpha$, and we earlier decided that it is "too unlikely" for a PRNG to be considered good.

Now we will make the above intuitions formal using statistical methods. Instead of events, it is convenient to use statistic T. In our case we verify the hypothesis $\mathcal{H}_0$ that the sequence we are interested in is the realization of a random sequence, i.e., a sequence of independent random numbers with the same distribution (uniform). In the hypothesis testing theory, two hypotheses are considered: $\mathcal{H}_0$ (in our case, that the sample comes from a random sequence) and $\mathcal{H}_1$ that the randomness assumption is not valid at some point. Two possible errors can be made:

- type I error – reject $\mathcal{H}_0$ even though it is true,
- type II error – accept $\mathcal{H}_0$ even though it is not true.

In classical statistics, the procedure is as follows. The test is performed using a T statistic. Typically, it will be a non-negative function of the tested sequence, although T can take real numbers in some situations.

The probability of type I error α is assumed (e.g., $\alpha = 0.05$, although some authors suggest 0.01). Often α is called the *significance level*. While testing we define the acceptance region $\mathcal{G}$ or critical region $\mathcal{G}^c$. It $T \in \mathcal{G}^c$, then $\mathcal{H}_0$ is rejected. As we already mentioned in Section 4.1.3 , we may have right-tail, left-tail and two-sided tests.

Thus, assuming $\mathcal{H}_0$ is true, we have $\mathbb{P}(T \notin \mathcal{G}) = \alpha$ for a continuous p.d.f. T, or otherwise $\mathbb{P}(T \notin \mathcal{G}) \geq \alpha$. The type II error informs about the quality of the test in a class of tests with a given significance level.

In the theory of generator testing, we do not use type II errors. However, when testing the randomness of sequences generated by a PRNG, it is recommended to use the concept of the so-called p-value instead of the standard procedure with a fixed significance level. Let T have **a continuous distribution function** $F(t)$ and the test is of one-sided right-tail type. We calculate the test statistic $T(obs)$ for a long sequence of numbers and then compute the so-called p-value:

$$p = 1 - F(T(obs)) \,. \tag{4.22}$$

Typically, when comparing various PRNGs, the ones with larger p-value are considered "better", the ones with p-values smaller than a prescribed significance level α are considered as the ones producing "non-random" sequences.

We have to keep in mind the following important property of p-values, which will proved in the next chapter in Section III.1. If p.d.f. F of statistics T is continuous, then $F(T)$ is $\mathcal{U}[0,1]$ distributed, which also means that $1 - F(T)$ is also uniformly distributed.

The first-level testing is the procedure we applied in previous examples: for one (usually very long) sequence of numbers (the output of a PRNG) we compute one p-value and then try to infer whether the sequence was randomly generated or not. However, it is crucial to be aware that p-values for statistics with continuous distribution functions are uniformly distributed $\mathcal{U}[0,1)$ (see Corollary III.1.3). Thus, if we set the significance level to say $\alpha = 0.05$ and we obtain, e.g., p-value 0.0093, usually (in first-level testing), we would reject $\mathcal{H}_0$. But note that on average, in 1 out of 20 p-values, we expect it to be smaller than 0.05. The same is with, e.g., large p-values. Recall Examples 4.1 and 4.2, where for points from set C we obtained p-values $> 99\%$. Again, in first-level testing, it means that we have no reasons to reject $\mathcal{H}_0$, but the intuition is that they are *suspiciously* good. Indeed, if we compute p-values for such numbers several times (i.e., for different sets $C_1, C_2, \ldots$, produced in a similar way to C), then the p-values would always be "*very good*" ($>99\%$), which is in obvious contradiction to the uniformity of these p-values. It is exactly what the second-level testing procedure does (computes several p-values and checks their uniformity) and which easily spots non-randomness in the above mentioned Examples 4.1 and 4.2.

Remark 4.12 Note that all p-values in previous sections were computed for distributions with **continuous** distribution function – usually chi-square distribution (mainly) or normal distribution. The p-values computed for discrete distributions are **not** uniformly distributed. In such a case, instead of testing the uniformity of p-values (which we do not have for discrete distributions), we test the distribution of the obtained statistics, usually using goodness-of-fit tests. It is precisely what it was done in the collision and birthday spacing tests. We computed some R statistics – e.g., $C_{k,r}^{(1)}, \ldots, C_{k,r}^{(R)}$ in case of a collision test or $K^{(1)}, \ldots, K^{(R)}$ in case of a birthday spacing test – and then tested if they are from a given distribution (Poisson in both cases) using the chi-square test.

It is left for the reader in Exercise II.T.23 to prove that p-values for discrete distribution is not uniformly distributed.

The second-level testing. The statistic T was computed for a random sequence; therefore, the p-value (4.22) is a random variable. If T is a continuous random variable, then under the hypothesis $\mathcal{H}_0$ it has the uniform distribution $\mathcal{U}[0,1)$ (the justification for this fact will be proven in Proposition III.1.2 (b)). That is why the following so-called *second-level testing* methodology works out. Instead of computing one p-value using the statistic T for one very long sequence of numbers from a PRNG, we

i) split it into subsequences;

ii) compute p-values for each subsequence (using the statistic T),

iii) test the uniformity of these p-values.

To be more precise. Assume we test the output of a PRNG of length Rn. We split it into R subsequences, each length n. For the i-th subsequence we compute the $T^{(i)}$ statistic ($i = 1, \ldots, R$) and then the corresponding p-value $p^{(i)} = 1 - F(T^{(i)})$. Assuming the hypothesis $\mathcal{H}_0$ we obtain the sequence $p^{(1)}, \ldots, p^{(R)}$ of i.i.d. random variables uniformly distributed $\mathcal{U}[0,1)$. We now need to test if these p-values are indeed i.i.d samples from the distribution $\mathcal{U}[0,1)$. It can be done in a variety of ways. We suggest following NIST's recommendation using the following chi-square uniformity test. We fix the partition $\mathcal{P}^s = \{P_i\}_{i\in\{1,\ldots,s\}}$ of the interval $[0,1)$ – consisting of s subintervals of equal length – and we count the number of p-values in this range. Namely,

$$P_i = \left[\frac{i-1}{s}, \frac{i}{s}\right), \quad 1 \le i \le s. \tag{4.23}$$

NIST recommends using $s = 10$. For $1 \le i \le s$ we define E_i (expected number of p-values in P_i) and O_i (the observed number of p-values in P_i):

$$O_i = |\{j : \ p^{(j)} \in P_i, 1 \le j \le R\}|, \quad E_i = \ \tfrac{R}{s} \ \text{ if } i \in \{1, \ldots, s\}$$

Then we compute the chi-square statistic

$$X^2_{\text{final}}(\text{obs}) = \sum_{i=1}^{s} \frac{(O_i - E_i)^2}{E_i}.$$

Under $\mathcal{H}_0$ the statistic X^2_{final} has (approximately) the chi-square distribution with $s-1$ degrees of freedom. We thus compute *the final* p-value using formula

$$p_{\chi^2,\text{final}} = 1 - F_{\chi^2,s-1}(X^2_{\text{final}}(\text{obs})), \tag{4.24}$$

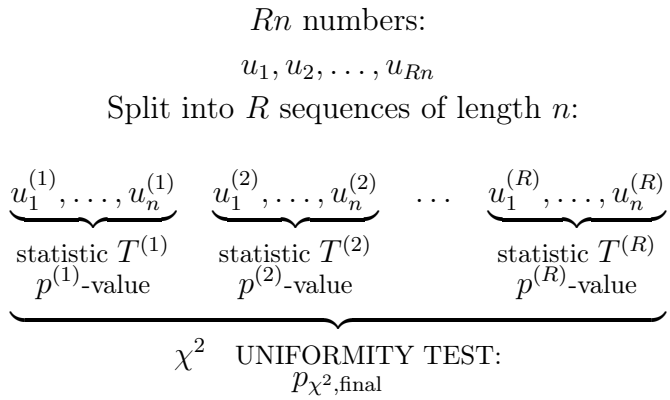


Figure 4.13: Sketched idea of the second-level testing for tests with continuous statistic T.

where $F_{\chi^2,s-1}$ is the chi-square distribution with $s-1$ degrees of freedom. The procedure is sketched in Fig. 4.13.

Generally, it is recommended to generate $R = 1000 - 10000$ sequences of length $n = 2^{20} - 2^{32}$.

The experiment can be repeated for different $n = 2^l$ by increasing l and observing if there is no "breakdown" of the final p-value. NIST [120] recommends to assume the significance level $\alpha \in [1/1000, 1/100]$ for this final p-value.

It turns out that second-level testing is *better* than first-level testing. There are, of course, nuances with choosing n and R, but in general, a second-level testing is more *reliable*. The exact definition of *reliability* and proving that second-level testing is more reliable is out of the scope of this textbook. Details can be found in [107] or [106]. The reliability analysis for a test based on the arcsine law is also provided in [86].

One of the cases when the second-level testing should be applied (actually, it is always recommended to use it) is when a p-value of a given PRNG using a given statistic is suspiciously large. It was the case with a quasirandom numbers – the p-values computed for one sequence (i.e., without using the second-level testing approach) using Kolmogorov-Smirnov test or chi-square test resulted in p-values exceeded 99% (see Examples 4.1 and 4.2 respectively). Indeed, using different seeds for quasirandom numbers, these tests always return such p-values – what is easily spotted by the second-level test,

as presented below in Example 4.13.

Example 4.13 We will continue the example of PRNGs which resulted in sets of points A, B and C presented in Fig. 4.3. Recall that in Example 4.1, we showed that Kolmogovor-Smirnov stated that we have no reasons to reject points A and C as random; we would reject only points B as random. The same was confirmed later in Example 4.3 by chi-square test with the partition given in (4.6).

In both cases, the p-values for set C were "very good", they exceeded 99%. It turns out that it was "too good", which is easily detected by the second-level testing approach.
We do the following: We produce $R = 200$ sequences, all of length $n = 50$ (thus, in Fig. 4.3 only one such sequence per PRNG is provided). To be more exact, we do the following:

- We produce $A_1, \ldots, A_R$ sets from Python's `NumPy` built-in PRNG `numpy.random.rand(n)`, set A_i was produced with $seed = i$.
- We prouce $B_1, \ldots, B_R$ – they are simply the following transformations of $A_1, \ldots, A_n$ sets. For given point $u_j^{A_i}$ from set A_i we compute

$$u_j^{B_i} = \frac{e^{u_j^{A_i}} - m_i}{M_i - m_i}, \quad \text{where } m_i = \min_k \left(e^{u_k^{A_i} - 1} - \varepsilon \right),\ M_i = \max_k \left(e^{u_k^{A_i} - 1} - \varepsilon \right)$$

  with $\varepsilon = 10^{-17}$.
- We produce quasirandom sets $C_1, \ldots, C_R$ using `Lattice` object from `Python`'s `qmcpy`[8] library. Set C_i is produced with $seed = i$.

Then we perform KS-test for each set obtaining p-values $p_{\text{KS}}^{A_i}, p_{\text{KS}}^{B_i}, p_{\text{KS}}^{C_i}, i = 1, \ldots, R$. Similarly, on the same sets, we perform the chi-square test with the partition partition given in (4.6) obtaining p-values $p_{\chi^2}^{A_i}, p_{\chi^2}^{B_i}, p_{\chi^2}^{C_i}, i = 1, \ldots, R$. The p-values are presented in Fig. 4.14 and 4.15.

[8] `https://pypi.org/project/qmcpy/`

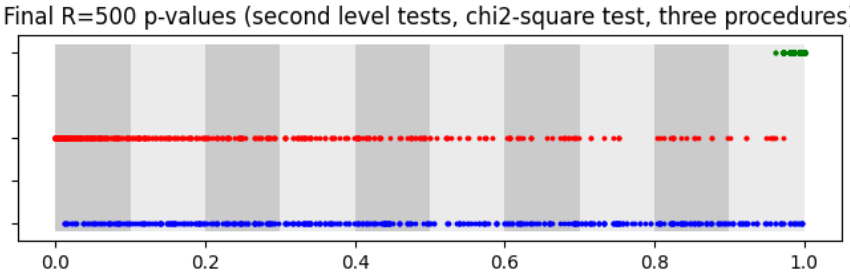


Figure 4.14: Three sets of R p-values $p_{\chi^2}^{A_i}$ (blue), $p_{\chi^2}^{B_i}$ (red), $p_{\chi^2}^{C_i}$ (green), $i = 1, \ldots, R$: Partition $\mathcal{P}^{10}$ given in (4.6) grayed-out.

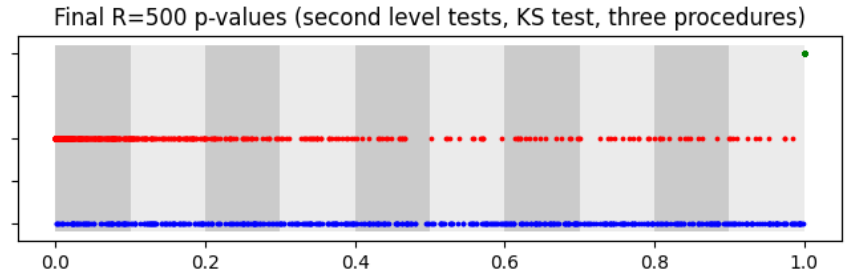


Figure 4.15: Three sets of R p-values $p_{\mathrm{KS}}^{A_i}$ (blue), $p_{\mathrm{KS}}^{B_i}$ (red), $p_{\mathrm{KS}}^{C_i}$ (green), $i = 1, \ldots, R$: Partition $\mathcal{P}^{10}$ given in (4.23) grayed-out.

As suspected – all the p-values p^{KS,C_i} and p^{χ^2,C_i} were "too perfect" – all around 99%. The number of p-values within each interval (box) is given in Table 4.10.

		P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}
χ^2	A	47	59	55	53	57	33	46	46	58	46
	B	226	69	50	40	41	16	26	7	16	9
	C	0	0	0	0	0	0	0	0	0	500
KS-test	A	57	55	45	52	43	41	55	48	48	56
	B	285	74	35	28	15	12	17	12	12	10
	C	0	0	0	0	0	0	0	0	0	500

Table 4.10: Number of p-values in intervales $P_i, i = 1, \ldots, 10$ for the partition $\mathcal{P}^{10}$ given in (4.23)

It is "clear" that p-values $p_{\text{KS}}^{C_i}$ and $p_{\chi^2}^{C_i}$ are not a realization of i.i.d. sequence with the distribution $\mathcal{U}[0,1)$. Concerning the p-values $p_{\text{KS}}^{B_i}$ and $p_{\chi^2}^{B_i}$, note that there are just a few of them in P_{10} (10 and 9 respectively), whereas there are many of them in P_1 (285 and 226 respectively), it turns out that they are also not uniformly distributed. We test it using the partition $\mathcal{P}^{10}$ which is the partition given in (4.23) for $s = 10$. We thus compute final p-values (4.24) and $\chi^2(obs)$ statistic obtaining:

		A	B	C
χ^2 test	$\chi^2_{\text{final}}(obs)$	11.480	758.72	4500.0
	p_{final}	0.244	$1.625 \cdot 10^{-157}$	0.0
KS-test	$\chi^2_{\text{final}}(obs)$	6.04	1295.12	4500.0
	p_{final}	0.73	$3.503 \cdot 10^{-273}$	0.0

In the end, using the second-level approach, we can only accept the method for producing points from sets A_i as a good PRNG, the other methods are bad PRNGs (what we expected from the beginning, knowing how the numbers were generated). □

An alternative (to the NIST's recommendation of using the chi-square test) method of testing uniformity of p-values in the second-level tests. There is another approach to analysing the sequence $p_1, \ldots, p_R$. We will focus on $\alpha = 1/100$. For the i-th subsequence define *fail* if $p_i \leq \alpha$ and then set $B_i = 0$, otherwise *pass* and $B_i = 1$. We have $\mathbb{P}(B_i = 1) = 1 - \alpha$, $\mathbb{E}B_i = 1 - \alpha$ and $\text{Var}B_i = \alpha(1 - \alpha)$. The fraction of passing is then $\hat{B} = \sum_{i=1}^{R} B_i/R$. We have $\mathbb{E}\hat{B} = 1 - \alpha$ and $\text{Var}\hat{B} = \alpha(1 - \alpha)/R$. Using the

CLT and the three-sigma rule (i.e., for a standard normal random variable Z we have $\mathbb{P}(|Z| > 3) \approx 0.0028$) we may write

$$\mathbb{P}\left(\hat{B} \notin [\mathbb{E}\hat{B} \pm 3\sqrt{\mathrm{Var}\hat{B}}]\right] = \mathbb{P}\left(\hat{B} \notin \left[1 - \alpha \pm 3\frac{\sqrt{\alpha(1-\alpha)}}{\sqrt{R}}\right]\right) \approx 0.0028.$$

Thus, if $\hat{B} \notin \left[1 - \alpha \pm 3\frac{\sqrt{\alpha(1-\alpha)}}{\sqrt{R}}\right]$ then we do not accept $\mathcal{H}_0$.

Remarks on errors in approximations. Note that so far, we assumed we know the distribution of a statistic T_n (i.e., the distribution of a statistic under consideration, assuming the randomness hypothesis, we add here subscript n to emphasise the dependence on sequence length). However, we often know an *approximation* of the considered distribution in practice. For example, if $U_i \in \{0, 1\}$, then $X_i = 2U_i - 1 \in \{-1, +1\}$ and the statistic $T_n = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} X_i$ has *approximately* standard normal $\mathcal{N}(0, 1)$ distribution. It turns out that to design a *reliable* test, it is crucial to know some bounds on the error of the approximations. We will not discuss here the details on how to design such a test; we refer to [106, 107] (also the definition of a *reliable* test is given therein). Let $X_1, X_2, \ldots$ be mean-zero i.i.d. random variables with $E|X_i|^3 < \infty$ and let $EX_i^2 = \sigma^2$. The central limit theorem states that the limiting c.d.f. F_n^Y of $Y_n = \sum_{i=1}^{n} \frac{X}{\sigma\sqrt{n}}$ is the c.d.f. Φ of the standard normal distribution. It means that *for large* n we can approximate Y_n by $\mathcal{N}$ and the approximation error is bounded by the Berry-Esseen inequality

$$\sup_x |F_n^Y(x) - \Phi(x)| \leq \frac{C_0 E|X_1|^3}{\sigma^3\sqrt{n}},$$

where C_0 is a positive constant (in original paper [39] $C_0 \leq 7.59$, in [127] it was shown that $C_0 \leq 0.4785$). See [106, 107] for a detailed example exploiting Berry-Esseen inequality for the binary matrix rank test. The reliability analysis for the test based on the arcsine law is also provided in [86]. Apart from an error made because one uses an asymptotic distribution, another error can be caused by the use of poor numerical procedures to compute the value of the limiting distributions. For example, there is a problem with the precise calculation of the incomplete gamma function for significant values of a.

5 Bibliographical comments

Knuth [71] is the main text for Section II. A comprehensive survey on random numbers and their testing is presented in an article by Ripley [111]. For more information on constructing PRNGs, consult the survey articles by L'Ecuyer [11] (Chapter 4) and [74].

Discussion of the cases of the "non-random" effect for MATLAB's generator `state` can be found in Savicky [122];https://www.cs.cas.cz/ savicky/-papers/rand2006.pdf . The example of a poor generator from Example 4.4 is taken from `http://www.mathworks.com/support/solutions/en/data/1-10HYAS/index.html?solution=1-10HYAS`.

Testing the quality of PRNGs is a vast field today. The primary source of information is Knuth's monograph [71], Chapter III, as well as papers by L'Ecuyer [11] (Chapter 4), [78, 74, 79], Marsaglia [92]. For sequences of bits, a central resource is NIST's report [120].

The concept of the second-level testing is from [107, 106]. The Mersenne Twister generator was developed in 1997 by Makoto Matsumoto and Takuji Nishimura in [94].

An idea to use normal approximations for the number of collisions for testing was taken from [126].

6 Exercises

Theoretical exercises

II.T.1 Prove the following inequalities:

Markov inequality: For a nonnegative random variable X we have

$$\mathbb{P}(X \geq a) \leq \frac{\mathbb{E}X}{a}, \quad \text{for any } a > 0. \tag{6.25}$$

Chebyshev inequality: For a real-valued random variable X we have

$$\mathbb{P}(|X - \mathbb{E}X| \geq b) \leq \frac{\mathrm{Var}X}{b^2}, \quad \text{for any } b > 0. \tag{6.26}$$

II.T.2 Suppose that $B_1, B_2, \ldots$, is a sequence of random bits, and for a given k, define a sequence of strings $Y_1, Y_2, \ldots$ from $\bar{M}$, where $M = 2^k$ by

$$Y_1 = \sum_{j=1}^{k} B_j 2^{j-1}, \; Y_2 = \sum_{j=1}^{k} B_{k+j} 2^{j-1}, \ldots \; .$$

Show that $Y_1, Y_2, \ldots$ is a sequence of random numbers from $\bar{M}$.

II.T.3 Let $Y_1, Y_2, \ldots$ be independent random variables with the same distribution $\mathcal{U}(\bar{M})$, where M is a natural number and $\bar{M} = \{0, 1, \ldots, M-1\}$. Assume that $M = 2^k$ and let $b(x)$ be a binary expansion of $x \in \bar{M}$ in k-digit representation (with padding zeros when needed). Show that the sequence of zero-one random variables defined by

$$B_1, B_2, B_2, \ldots = b(Y_1), b(Y_2), \ldots$$

is a random sequence of bits. Show that this property does not hold if M is not a power of 2.

II.T.4 Let $U_1, \ldots$ be a sequence of random independent variables with the same distribution $\mathcal{U}[0, 1)$ and $M \in \{2, \ldots\}$. Show that

$$X_i = \lfloor MU_i \rfloor, \qquad i = 1, \ldots$$

is a random sequence with values in $\bar{M} = \{0, 1, \ldots, M-1\}$.

II.T.5 Suppose that $B_1, B_2, \ldots$ is a sequence of random bits and for a given n, let $\mathbf{\Pi} = (\Pi_1, \ldots, \Pi_n)$ be a random permutation. We assume that $\mathbf{\Pi}$ and $B_1, B_2, \ldots$ are independent. Show that the sequence

$$B_{\Pi_1}, \ldots, B_{\Pi_n}$$

is also a sequence of random bits.

II.T.6 Suppose that $B_1, B_2, \ldots$ is a sequence of random bits and $x_1, x_2, \ldots$ is a deterministic sequence of bits ($x_i = 0, 1$). Show that the sequence $x_1 \oplus B_1, x_2 \oplus B_2, \ldots$ is also a sequence of random bits. Prove that for bits x and b we have $(x \oplus b) \oplus b = x$.

II.T.7 Verify that the following PRNG's fulfil Definition 1.1 (show S, f, V and g): LCG(M, a, c), LCG$(M, a_1, \ldots, a_k, c)$, Java, quadratic congruential generator (QCG I), lagged Fibonacci generator, L'Ecuyer mixed generator.

II.T.8 Justify that the Fisher-Yates procedure generates a random permutation by proving the following fact. We inductively define the sequence of n permutations of the set $[n]$. We set σ_0 to be the identity permutation. For $k = 1, 2, ..., n-1$ we define inductively σ_k with σ_{k-1} as follows: replace an element on position k with element on position J_k, where J_k is uniformly distributed on $\{k, \ldots, n\}$ and independent of of $J_1, \ldots, J_{k-1}$. Show that σ_{n-1} is a random permutation.

II.T.9 *Error function* $\operatorname{erf}(x)$ is defined as

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt, \qquad x \in \mathbb{R}$$

and *complementary error function* $\operatorname{erfc}(x)$

$$\operatorname{erfc}(x) = 1 - \operatorname{erf}(x) \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2}\, dt.$$

The inverse to the error function is denoted by $\operatorname{erfinv}(x)$, where $x \in (-1, 1)$, that is $\operatorname{erf}(\operatorname{erfinv}(x)) = x$. Similarly, define the inverse to the complementary error function $\operatorname{erfcinv}(x)$. Show that $\operatorname{erfcinv}(x) =$

erfinv$(1-x)$. Prove that if $\Phi(x)$ is the c.d.f. of the standard normal distribution, then

$$\Phi(x) = \frac{1}{2} + \frac{1}{2}\mathrm{erf}(x/\sqrt{2}) = \frac{1}{2}\mathrm{erfc}(-x/\sqrt{2}),$$

and

$$\Phi^{-1}(x) = \sqrt{2}\mathrm{erfinv}(2x-1).$$

II.T.10 Let

$$\gamma(a,x) = \int_0^x t^{a-1}e^{-t}\,dt$$

and $\Gamma(a) = \gamma(a,\infty)$. There are two *incomplete gamma functions* (igamf)

$$P(a,x) = \frac{\gamma(a,x)}{\Gamma(a)}, \qquad Q(a,x) = 1 - P(a,x).$$

Let $F(x)$ be the c.d.f. of the gamma distribution Gamma(a,λ) and $G(x) = 1 - F(x)$.

a) Express F and G in terms of $P(a,x)$ and $Q(a,x)$.

b) Let W has the chi-square distribution with k degree of freedom, which is the gamma distribution Gamma$(k/2, 1/2)$ with density function

$$\frac{x^{\frac{k}{2}-1}e^{-\frac{x}{2}}}{2^{\frac{k}{2}}\Gamma\left(\frac{k}{2}\right)}.$$

Suppose statistic W is used in a one-sided right-tail test. After conducting the experiment, we have obtained $W = W(\mathrm{obs})$. Show that the p-value is expressed by $p = Q(k/2, 2W(\mathrm{obs}))$.

II.T.11 Show that

$$\frac{2}{\sqrt{\pi}}\int_0^y e^{-x^2}\,dx = P(1/2, y^2).$$

II.T.12 Let N have a Poisson distribution with parameter $\lambda > 0$. Show that tail distribution function $\bar{F}_\lambda(k) = \sum_{j=k+1}^{\infty} \lambda^j \exp(-\lambda)/j!$ can be expressed as

$$P(N > k) = \bar{F}_\lambda(k) = P(k+1, \lambda) = 1 - Q(k+1, \lambda).$$

Assume $N(\mathrm{obs})$ events are observed. Express p-value in terms of the incomplete gamma function.

II.T.13 Let $\bar{f} = \int_{[0,1]^d} f(\boldsymbol{u})\, d\boldsymbol{u}$ and $\sigma_f^2 = \mathrm{Var} f(\boldsymbol{U})$, where $\boldsymbol{U}$ is uniformly distributed on $[0,1] \times [0,1]$. Show that for a sequence $\boldsymbol{U}_1, \boldsymbol{U}_2, \ldots$ of i.i.d. random vectors distributed as $\boldsymbol{U}$, we have

$$\mathbb{P}\left(\left| \frac{1}{n} \sum_{j=1}^{n} f(\boldsymbol{U}_j) - \bar{f} \right| < \frac{\sigma_f}{\sqrt{\delta n}} \right) \leq 1 - \delta.$$

II.T.14 [Feller, vol. 1, p.94] In the classical random arrangement problem of r balls in k boxes, each arrangement has probability k^{-r}. Let $E_{k,r}$ be the number of empty boxes. Show that the probability of exactly c empty boxes $p_c(r,k) = \mathbb{P}(E_{k,r} = c)$ is

$$\binom{k}{c} \sum_{j=0}^{k-c} (-1)^j \binom{k-c}{j} \left(1 - \frac{r+j}{k}\right)^r.$$

Prove that if k, r increase to ∞ in a way that $\lambda = k e^{-\frac{r}{k}}$ remains bounded, then for $c = 0, 1, \ldots$

$$p_c(r,k) \to \frac{\lambda^c}{c!} e^{-\lambda}.$$

II.T.15 Continuing Example II.T.14. Recall that $C_{k,r}$ is the number of collisions in the classical random arrangement problem of r balls in k boxes, each arrangement has probability k^{-r}. Consider regimes:

(A') $k \exp(-\frac{r}{k}) = \lambda$ for $k, r \to \infty$ for some $\lambda > 0$.

(A'') for $r, k \to \infty$ and $r^2/(2k) = \lambda$, for some $0 < \lambda < \infty$

Show that under regime (A') we have $\mathbb{E} C_{k,r} \to \lambda$ for some $0 < \lambda$ if and only if under regime (A'') we have $\mathbb{E} E_{k,r} \to \lambda$.

II.T.16 Consider the variance $\sigma_{k,r}^2 = \mathrm{Var} E_{k,r}$. Show that for $r = \alpha k$ for some $\alpha > 0$ and $k \to \infty$ we have

$$\sigma_{k,r}^2 \to e^{-\alpha}(1 - (1+\alpha)e^{-\alpha}).$$

II.T.17 Consider the serial test, and let O_i $(i = 1, \ldots, k)$ be the number of balls falling into the i-th box. Suppose that hypothesis $\mathcal{H}_0$ holds. Argue that random vector $(O_1, \ldots, O_r)$ has multinomial distribution $\mathrm{M}(r, 1/k, \ldots, 1/k)$. Define $X_k^2 = \sum_{i=1}^{k} \frac{(O_i - r/k)^2}{r/k}$. Show that $\mathbb{E}(X_k^2) = k - 1$.

II.T.18 In a poker card game one draws 5 cards from 52 deck. The number of all hands is $W = \binom{52}{5} = 2\ 598\ 960$. Justify the following results:

- one pair: $\frac{1\,098\,240}{W} = 0.422569 = 1/24$
- two pairs: $\frac{123\,552}{W} = 0.047539 = 1/21$
- three of a kind: $\frac{54\,912}{W} = 0.0211 = 1/44$
- full house: $\frac{3\,744}{W} = 0.00144 = 1/694$
- four of a kind: $\frac{624}{W} = 0.00024 = 1/4165$
- flush: $\frac{5\,148}{W} = 0.00198 = 1/50$
- straight: $\frac{9 \cdot 4^5}{W} = \frac{9216}{W} = 0.003546 = 1/282$
- straight flush: $\frac{9 \cdot 4}{W} = \frac{36/W}{W} = 0.0000138 = 1/72193$.

II.T.19 [Riffle shuffle and its time reversal]. Let $p_{RS}(\sigma, \sigma')$ be the probability that a single step of a Riffle shuffle procedure (see page 35) transforms permutation σ into σ'. Similarly, $p_{TR-RS}(\sigma, \sigma')$ denote the probability that a single step of a Time Reversed Riffle Shuffle (see page 36) transforms σ into σ'. Show that for any $\sigma, \sigma' \in \mathcal{S}_n$ we have $p_{RS}(\sigma, \sigma') = p_{TR-RS}(\sigma', \sigma)$.

II.T.20 [Geometric interpretation of Riffle shuffle]. Let $x_1, \ldots, x_n$ be realizations of i.i.d. $\mathcal{U}[0, 1)$ random variables and denote by $x_{(1)} \leq \ldots \leq x_{(n)}$ their sorted values. For each $x_{(i)}$ consider the transformation

$$T(x) = 2x \bmod 1$$

(the fractional part of $2x$) and let $y_i = T(x_{(i)})$. The y_i may not be in order, rearrange them (permute via σ) as

$$y_{\sigma(1)} \leq \ldots \leq y_{\sigma(1)}.$$

Show that the probability of obtaining a permutation σ is equal to $p_{RS}(id, \sigma)$, where id is an identity permutation and p_{RS} is given in Exercise II.T.19.

In other words, the induced permutation has the same distribution as the one in Riffle shuffle scheme.

II.T.21 Let σ be a random permutation of $\{1, \ldots, M\}$ and define $H = \sum_{i=1}^{n} H_i$, where $H_i = \mathbf{1}(\sigma(i) = i)$. Calculate $\mathbb{E}H, \mathrm{Cov}\,(H_i, H_j)$ and $\mathrm{Var}H$.

II.T.22 Let σ be a random permutation of $\{1, \ldots, M\}$ and let C_m be a number of cycles of length $m \leq M$. Total number of cycles is given by $C = \sum_{i=1}^{M} C_m$.

- How many, on average, cycles of length j are in a random permutation? I.e., compute $\mathbb{E}C_j$.
- How many, on average, cycles (of any length) are in a random permutation? I.e., compute $\mathbb{E}C$.

II.T.23 Suppose that statistic T takes values $\{a_1, a_2, \ldots, a_n\}$ (n can be ∞) with the corresponding probabilities $w_1, w_2, \ldots, w_n$. Let F be the corresponding distribution function of T. We define one-sided right-tail p-value similarly as in continuous case:

$$p = 1 - \mathbb{P}(T \leq T(obs)) = 1 - F(T(obs)).$$

The possible values of p are

$$p \in \{r_1, r_2, \ldots, r_n\}, \quad \text{where } r_k = \sum_{i=k}^{n} w_i.$$

- Show that $\mathbb{P}(p = r_k) = w_k$.
- Compute the distribution of one-sided left-tail p-values.
- Compute the distribution of two-sided p-values.

II.T.24 Let $C_{k,r}$ be the number of collisions in the collision test, with k boxes and r balls. Show that

$$\mathbb{P}(C_{k,r} = c) = \frac{k(k-1)\ldots(k-r+c+1)S_2(r, r-c)}{k^r},$$

where $S_2(r,d)$ is the Stirling number of the second kind. By $S_2(r,d)$ it is meant the number of ways to partition a set of r labelled objects into d non-empty non-labelled subsets.

II.T.25 Under (A^*) (page 79), demonstrate asymptotics (4.17) and (4.18). Hint: Use use $(1-x)^a = \sum_{j=1}^{\infty} \binom{a}{j}(-x)^j$, where $\binom{a}{j} = a(a-1)\cdots(a-j+1)/j!$.

II.T.26 Let X be a discrete random variable taking values $\{a_1, a_2, \ldots, a_n\}$ (n can be ∞) with the corresponding probabilities $w_1, w_2, \ldots, w_n$, denote the distribution function of X by F_X. Let Y be a random variable with a continuous distribution function F_Y. Suppose that statistic T is a mixture of X and Y, i.e., for some $q \in (0,1)$ we have

$$F_T(t) = qF_X(t) + (1-q)F_Y(t).$$

We define one-sided right-tail p-value as usual: $p = 1-\mathbb{P}(T \le T(obs)) = 1 - F_T(T(obs))$. Indicate all possible values of p-values.

II.T.27 For a statistic T assuming values in $\mathbb{R}$ in the case of two-sided test with non-symmetric distribution of T, one defines p-value by

$$p = 2\min\left\{\mathbb{P}(T \ge T(\text{obs})), \mathbb{P}(T \le T(\text{obs}))\right\}. \tag{6.27}$$

Assume that the c.d.f. G of T is continuous.

a) Show that p is uniformly distributed $\mathcal{U}[0,1)$.

b) Show that if T is symmetric about zero, then

$$p = \mathbb{P}(|T| \ge |T(\text{obs})|).$$

II.T.28 Consider the example of flipping coins 100 times provided in the beginning of Section 4.6. Statistic T has binomial distribution. Consider one-sided right-tail test and also two-sided test.

Compute $2\min\{\mathbb{P}(T \le 64), \mathbb{P}(T \le 36)\}$. Note that under $\mathcal{H}_0$, statistics T is symmetric about its mean.

- Compute numerically its exact value.
- Use the CLT approximation. Is it good?

Lab exercises

II.L.1 Implement LCG with parameters $M = 2^{11} - 1, a = 5, c = 1$. Generate 10^4 numbers $u_1, \ldots, u_{10^4}$ starting with a seed of your choice, and plot points $(i, u_i), i = 1, \ldots, 10^4$.

- Perform Kolmogorov-Smirnov test (us built-in one) to check uniformity of the numbers.
- What is a period of this PRNG? (implement a function for computing it). Does it depend on a seed? Are assumptions of Theorem 3.1 fulfilled?
- Answer above question for $a = 2048$.

II.L.2 Consider Fisher-Yates Algorithm 1 with $n = 52$ and a modification (call it Algorithm B) where line 3 is replaced with:

$$k = \textbf{random}(\{1, 2, \ldots, n\}).$$

In other words, in Algorithm B at step i element at position i is swapped with an element on a random position. Generate $R = 10^4$ permutations with Fisher-Yates and Algorithm B.

- Perform the fixed point permutation test (report p-values) for both algorithms with boxes $A_0 = \{0\}, A_1 = \{1\}, \ldots, A_9 = \{9\}, A_{10} = \{10, 11, \ldots\}$. Plot histograms of the fixed points distribution of both algorithms and overlay the expected distribution of fixed points for visual comparision.
- Perform the derangement test for both algorithms, report p-values.

II.L.3 Consider the Mersenne Twister PRNG (use the built-in generator) and your implementation of LCG with $M = 2^{31} - 1, a = 16807, c = 1$. Generate $n = 10^5$ numbers $y_1, \ldots, y_n$ using both generators. Compute the correlation between consecutive numbers – compute Pearson correlation coefficient between sequences $(y_1, \ldots, y_{n-1})$ and $(y_2, \ldots, y_n)$, i.e.,

$$\rho = \frac{\sum_{i=1}^{n-1}(y_i - \bar{y})(y_{i+1} - \bar{y})}{\sqrt{\sum_{i=1}^{n-1}(y_i - \bar{y})^2}\sqrt{\sum_{i=1}^{n-1}(y_{i+1} - \bar{y})^2}}, \quad \text{where } \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i.$$

Which generator produces "better" (i.e., produces consecutive numbers with lower correlation) numbers?

II.L.4 Simulate random walk by generating randomly (use some built-in procedures) bits $b_1, b_2, ...$

- Plot the position of a random walk for first 1000 steps, i.e., plot (k, D_k), where $D_k = \sum_{i=1}^{k} b_i$.
- Repeat the simulations 1000 times and record final positions $D^1_{1000}, D^2_{1000}, \ldots, D^{1000}_{1000}$. Plot scaled values, dividing each D^i_{1000} by $\sqrt{1000}$.
- Perform the frequency monobit test, report the final p-value. Are (according to this test) generated bits random?
- What would change if you sort each group of 1000 bits first?

II.L.5 Consider the following numbers $u_i = \frac{i}{M} \bmod 1$ with $M = 100$, $i = 1, \ldots, 10^6$. Perform Kolmogorov-Smirnov and chi-square goodness-of-fit tests (use setups from Section 4.2). Perform both first and second-level testing, and present your conclusions.

II.L.6 Consider setup from Exercise II.T.17. Justify by simulations that $\mathrm{Var}(X^2_k) \sim 2(k-1)$.

Projects

Project 1 John estimates that he will meet three partners in his life and marry one. He assumes that he is able to compare them. However, being an honorary person, (i) if he decides to date a new partner, he cannot return to the one who has already been rejected, (ii) at the time of the decision to marry, dating with others is impossible, (iii) must decide on one of the partners you meet. John assumes that you can rank between partners: 1 = good, 2 = better, 3 = the best, but he does not know this ranking. John meets more partners in random order.

Write a report on what strategy John should adopt. Especially in the report the following questions should be answered. When preparing report, analyse the error using the fundamental formula

$$b = \frac{1.96\sigma}{\sqrt{R}}$$

(see subsection IV.1.5, where b - error, R - number of replications, and σ variance of the estimator). So if we want to estimate r with an error less than $b = 0.02$ we can use the fundamental formula with the variance obtained earlier from the pilot simulation of 1000 replicates.

- (a) John decides to get married for the first time he meets. Calculate the probability of P (theoretical) that he would marry the best. Also calculate the expected rank of r

- (b) Use simulation to determine P and r for the following strategy. John never gets married marries the former, marries the latter if it is better than the former, otherwise marries the third.

- (c) Consider the task with 10 partners who have ranks $1.2, \ldots, 10$. Theoretically calculate the probability of P and the expected rank of r for the strategy as in problem (a).

- (d) Define a set of strategies that are an adaptation of the strategies from the task (b) for 10 partners. Give yet another strategy for Bob. For each strategy, calculate P and r. Indic. Please note that how to we do not decide on a candidate, then it is not for her return.

Project 2 Consider the following random number generators: state, twister and from Excel $u_i = (0.9821u_{i-1} + 0.211327) \mod 1$. In addition, consider the pseudo-random bit sequences from date.e, date.sqrt2 (from the website www.math.uni.wroc.pl/řolski).

(i) For these strings, analyse their " randomness " using at least 3 tests (for each string, respectively). For example, Kolmogorov-Smirnov, batch test and birth day interval test. Instead of this ostenti you can use the collision test.

(ii) Examine two cases for the generator " twister ". The first one when the generator starts from the grain $u_o = 0$, and the second one when from the grain $u_0 = 1812433253$.

Chapter III

Generating random variables

This chapter explores methods for generating random numbers, commonly known as random variables, with given distributions. The discussion is rooted in probability theory, focusing on random variables defined on the probabilistic space $(\Omega, \mathcal{F}, \mathbb{P})$.

1 Inverse Transform Method (ITM) for Exponential and Pareto Distributions

We aim to generate a random variable Z with cumulative distribution function (c.d.f.) F using the Inverse Transform Method (ITM). For this purpose, we employ U, a random variable with a uniform distribution $\mathcal{U}[0,1)$. In the sequel, we will continue without explanation to write $U_1, U_2, \ldots$ for the sequence of i.i.d. random variables with the common distribution $\mathcal{U}[0,1)$.
Let $F^{\leftarrow}(t)$ be the generalized inverse function defined as

$$F^{\leftarrow}(t) = \inf\{x : \ t \leq F(x)\} \ . \tag{1.1}$$

In case when the c.d.f. $F(x)$ is a strictly increasing and continuous function, $F^{\leftarrow}(t)$ is the inverse of $F^{-1}(t)$ of $F(x)$. Therefore, $F^{\leftarrow}(t)$ will be called the generalized inverse function. In Fig. 1.1, situations for constructing the generalized inverse function are sketched (in Fig. 1.2, the corresponding generalized inverse function is plotted).
Let us point out some of the properties of the generalized inverse function $F^{\leftarrow}(t)$:

(P1) $F^{\leftarrow}(u)$ is non-decreasing, left-continuous at u and admits a limit from the right at u.

(P2) If F is strictly increasing and $t \in F(\mathbb{R})$, then it is clear that the following equivalence holds: $t = F(x)$ if and only if $F^{\leftarrow}(t) = x$.

(P3) If the condition that $t \in F(\mathbb{R})$ is not satisfied, then an example can be given such that $F^{\leftarrow}(t) = x$, but $t < F(x)$ (see $x_1 = F^{\leftarrow}(t_1)$ in Fig. 1.1).

(P4) $F^{\leftarrow}(t) = \inf\{x : t \leq F(x)\} = \sup\{x : t > F(x)\}$.

(P5) $F^{\leftarrow}(F(x)) \leq x$. If F is strictly increasing, then $F^{\leftarrow}(F(x)) = x$.

(P6) $F(F^{\leftarrow}(u)) \geq u$. We leave the reader to find an example on a figure that $F(F^{\leftarrow}(u)) \neq u$.

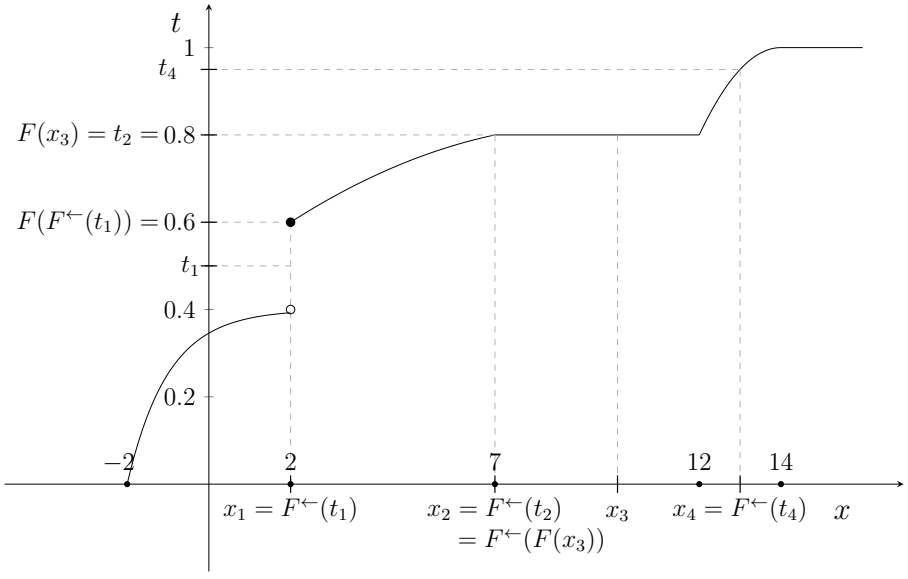


Figure 1.1: An example of a distribution function F and some properties of the generalized inverse function $F^{\leftarrow}$.

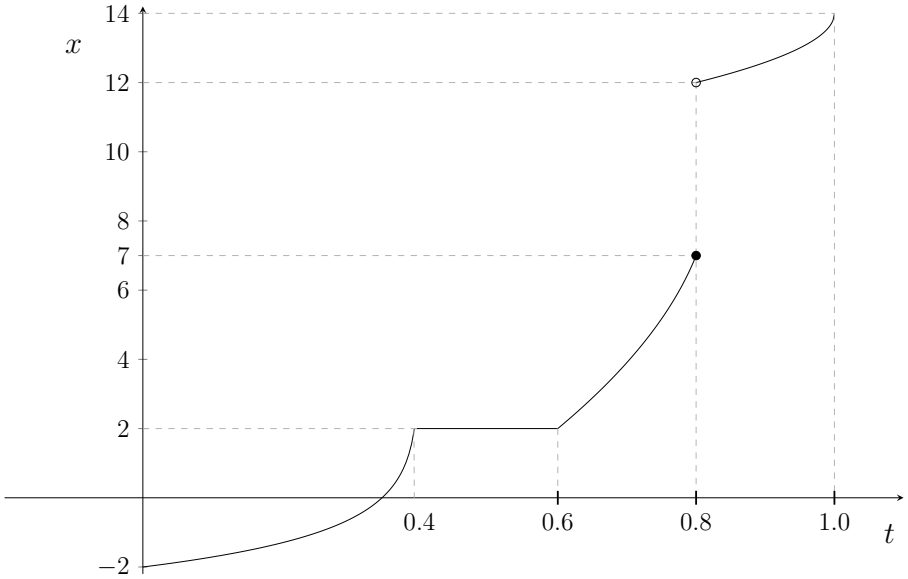


Figure 1.2: The generalized inverse function $F^{\leftarrow}$ of F from Fig. 1.1.

We will denote the *tail* of a c.d.f. $F(x)$ by

$$\overline{F}(x) = 1 - F(x).$$

For a tail distribution c.d.f. $\overline{F}$ define

$$\begin{aligned} \overline{F}^{\leftarrow}(u) &= \inf\{x : \overline{F}(x) \le u\} \\ &= \inf\{x : 1 - u \le F(x)\} = F^{\leftarrow}(1-u). \end{aligned}$$

The following properties of $F^{\leftarrow}(u)$ will be useful.

Lemma 1.1 (a) *$u \le F(x)$ if and only if $F^{\leftarrow}(u) \le x$.*
(b) *If $F(x)$ is continuous, then $F(F^{\leftarrow}(t)) = t$.*

Proof
(a) See Fig. 1.1
(b) We always have $F(F^{\leftarrow}(u)) \ge u$, see property (P6). On the other hand, take any $u \in (0,1)$ and let x be such that $u = F(x)$ (we can always find such x because F is continuous). Notice the relationship $F^{\leftarrow}(u) \le x$. Hence

$$F(F^{\leftarrow}(u)) \le F(x) = u,$$

which completes the proof of part (b). □

Proposition 1.2 *(a) Suppose U is a uniformly distributed random variable. Then $F^{\leftarrow}(U)$ is a random variable with the distribution F.*
(b) If F is continuous and $X \sim F$, then $F(X)$ has the uniform distribution $\mathcal{U}[0,1)$.

Proof (a) From Lemma 1.1 (a) we have

$$\mathbb{P}(F^{\leftarrow}(U) \leq x) = \mathbb{P}(U \leq F(x)) = F(x) \ .$$

(b) From Lemma 1.1 (a) we have

$$\mathbb{P}(F(X) \geq u) = \mathbb{P}(X \geq F^{\leftarrow}(u)) \ .$$

Since a random variable X has a continuous c.d.f., so it is atomless, we can thus write

$$\mathbb{P}(X \geq F^{\leftarrow}(u)) = \mathbb{P}(X > F^{\leftarrow}(u)) = 1 - F(F^{\leftarrow}(u)) \ ,$$

which by a virtue of Lemma 1.1 (b) equals to $1 - u$. □

As a consequence, we have a direct corollary

Corollary 1.3 *Let a c.d.f. F of T be continuous. Then, the p-value has the uniform $\mathcal{U}[0,1)$ distribution.*

Proof As usual $U \sim \mathcal{U}[0,1)$. There are three scenarios for p-value as defined in Section 4.1.3 .

One-sided, right-tail In this case p-value

$$p = 1 - F(T) \overset{\text{Prop. 1.2 (b)}}{=} 1 - U \overset{\mathcal{D}}{=} U.$$

One-sided, left-tail

$$p = F(T) \overset{\text{Prop. 1.2 (b)}}{=} U.$$

Two-sided, symmetric In this case, the distribution of T is

$$F_{|T|}(t) = \mathbb{P}(-t \le T \le t) \overset{\text{continuity of}F}{=} \mathbb{P}(-t < T \le t) = F(t) - F(-t),$$

which is continuous. Hence

$$p = F_{|T|}(|T|) \overset{\text{Prop. 1.2 (b)}}{=} U.$$

□

We are now in a position to write the following algorithm for generating a random variable with a given continuous c.d.f. F. The algorithm (and whole method) is called ITM (Inverse Transform Method). In many software packages, `rand` is the call for a random number uniformly distributed $\mathcal{U}[0,1)$.

Algorithm 6 ITM; generating $X \sim F$

Input: c.d.f. F
Output: $X \sim F$
1: Generate $U \sim \mathcal{U}[0,1)$
2: Set $X = F^{\leftarrow}(U)$
3: **return** X

Another variant of the ITM algorithm is to take $Y = \overline{F}^{\leftarrow}(U)$.

Note something else that will be useful later. If $U_1, U_2, \ldots$ is a sequence of i.i.d. uniformly distributed random variables, then $U_1', U_2' \ldots$ is also a sequence of i.i.d. uniformly distributed random variables, where $U_j' = 1 - U_j$. Unfortunately, explicit formulas for $F^{\leftarrow}(x)$ or $\overline{F}^{\leftarrow}(x)$ are known in a few cases only, below, we will discuss some of them.

Exponential distribution Exp(λ). In this case

$$F(x) = \begin{cases} 0, & x < 0 \\ 1 - \exp(-\lambda x), & x \ge 0. \end{cases} \tag{1.2}$$

If $X \sim \text{Exp}(\lambda)$, then $\mathbb{E}X = 1/\lambda$ and $\text{Var}X = 1/\lambda^2$. It is a light-tail distribution because

$$\mathbb{E}e^{sX} = \frac{\lambda}{\lambda - s}$$

is well defined for $0 \leq s < \lambda$. We have

$$\overline{F}^{\leftarrow}(u) = -\frac{1}{\lambda}\log(u) \ .$$

Hence, we can generate an exponentially distributed random variable $\mathrm{Exp}(\lambda)$ using the following simple algorithm.

Algorithm 7 ITM-Exp; generating $X \sim$Exp(λ)

Input: λ
Output: $X \sim$Exp(λ)
1: Generate $U \sim \mathcal{U}[0,1)$
2: Set $X = -\log(U)/\lambda$
3: **return** X

We now look at an example of a heavy-tailed random variable.

Pareto distribution Par(α). The c.d.f. of the Pareto $\mathrm{Par}(\alpha)$ distribution is

$$F(x) = \begin{cases} 0 & x < 0, \\ 1 - \dfrac{1}{(1+x)^{\alpha}}, & x \geq 0 \ . \end{cases} \tag{1.3}$$

If $X \sim \mathrm{Par}(\alpha)$, then $\mathbb{E}X = 1/(\alpha - 1)$ provided that $\alpha > 1$, and $\mathbb{E}X^2 = 2/((\alpha-1)(\alpha-2))$ provided that $\alpha > 2$. It is a heavy-tailed random variable because

$$\mathbb{E}e^{sX} = \infty$$

for all $s > 0$. The inverse function of $\overline{F}$ is

$$\bar{g}(x) = x^{-1/\alpha} - 1.$$

Note that our definition of the Pareto distribution has one parameter only. If one wants to have a more flexible class to accommodate the given mean m and shape parameter α, we use the fact that random variable $m(\alpha-1)X$ has mean m (assuming $\alpha > 1$). Then $m(\alpha-1)X$ has the tail distribution

$$\mathbb{P}(m(\alpha-1)X > x) = \left(\frac{1}{1 + x/(m(\alpha-1))}\right)^{\alpha} = \frac{(m(\alpha-1))^{\alpha}}{(m(\alpha-1)+x)^{\alpha}}.$$

Setting $c = (\alpha - 1)m$, we can see that one can introduce a two-parameter family of Pareto distribution $\mathrm{Par}(\alpha, c)$. Thus, it is said that the random variable Y has Pareto $\mathrm{Par}(\alpha, c)$ distribution, if its c.d.f. is

$$F(x) = \begin{cases} 0, & x < 0 \\ 1 - \frac{c^\alpha}{(c+x)^\alpha}, & x \geq 0 \, . \end{cases} \tag{1.4}$$

Remark that there is no unique definition of Pareto distribution in the literature. Therefore, one can find variants other than those given in these notes.

Algorithm 8 ITM-Par; generating $X \sim \mathrm{Par}(\alpha)$

Input: α
Output: $X \sim \mathrm{Par}(\alpha)$
1: Generate $U \sim \mathcal{U}[0, 1)$
2: $X = U^{(-1/\alpha)} - 1$
3: **return** X

2 ITM variants for lattice distributions; geometric distribution

Suppose a distribution is concentrated on $\{0, 1, \ldots\}$. Then it suffices to know the probability function $(p_k, \ k = 0, 1, \ldots)$.

Proposition 2.1 *Let U be uniformly distributed random variable $\mathcal{U}[0, 1)$. Random variable*

$$Y = \min \left\{ k : U \leq \sum_{i=0}^{k} p_i \right\}$$

has the probability function (p_k).

Proof Notice that

$$\mathbb{P}(Y = l) = \mathbb{P}\left(\sum_{i=0}^{l-1} p_i < U \leq \sum_{i=0}^{l} p_i \right),$$

and that interval $\left(\sum_{i=0}^{l-1} p_i, \sum_{i=0}^{l} p_i\right]$ has length p_l. Recall the convention that $\sum_{i=0}^{-1} = 0$. □

Hence we have the following algorithm for generating lattice random variables Y.

Algorithm 9 ITM-d; generating $Y \sim (p_k)$

Input: (p_k)
Output: $Y \sim (p_k)$
1: $Y = 0$
2: Generate $U \sim \mathcal{U}[0, 1)$
3: $S = p_0$
4: **while** $S < U$ **do**
5: $\quad Y = Y + 1, \;\; S = S + p_Y$
6: **end while**
7: **return** Y

The number of calls in the loop is equal to 1 when $U \leq p_0$, equal to 2 when $p_0 < U \leq p_0 + p_1$, etc. Hence we have the following proposition.

Proposition 2.2 *Let N be the number of calls in the loop while in Algorithm 9. Then N has a probability function*

$$\mathbb{P}(N = k) = p_{k-1}, \qquad k = 1, 2, \ldots,$$

and hence $\mathbb{E}N = \mathbb{E}Y + 1$.

We thus see that the algorithm ITM-d for lattice distributions with infinite (or very large) mean is not good.

Geometric distribution Geo(p) The geometric distribution has a probability function

$$p_k = (1 - p)p^k, \qquad k = 0, 1, \ldots$$

A random variable $X \sim \text{Geo}(p)$ has mean $\mathbb{E}X = p/q$ and

$$X = \lfloor \log U / \log p \rfloor$$

has distribution Geo(p), since

$$\begin{aligned}\mathbb{P}(X \geq k) &= \mathbb{P}(\lfloor \log U / \log p \rfloor \geq k) \\ &= \mathbb{P}(\log U / \log p \geq k) \\ &= \mathbb{P}(\log U \leq k \log p) \\ &= \mathbb{P}(U \leq p^k) = p^k \ .\end{aligned}$$

$(a, b, 0)$ **class of distributions** Suppose that $(p_0, p_1, \ldots)$ is a probability function. In some cases, the following quotient

$$c_{k+1} = \frac{p_{k+1}}{p_k}$$

have a simple form. In actuarial mathematics, a popular class of probability functions (p_k) is the $(a, b, 0)$ class of distributions satisfying the recursion $p_k = p_{k-1}(a + \frac{b}{k})$ $(k = 1, 2, \ldots)$ for some a, b, and $p_0 > 0$ (remember that $p_0 + \ldots = 1$). See also Exercise III.T.20, where the $(a, b, 1)$ class of distributions on $1, 2, \ldots$ is defined We will now give examples.

- $X \sim \mathrm{B}(n, p)$ with probability function $p_k = \binom{n}{k} p^k (1-p)^{n-k}$. Then

$$p_0 = (1-p)^n, \quad c_{k+1} = \frac{p(n-k)}{(1-p)(k+1)}, \qquad k = 0, \ldots, n-1.$$

- $X \sim \mathrm{Poi}(\lambda)$ with probability function $p_k = e^{-\lambda}\lambda^k/k!$. Then

$$p_0 = e^{-\lambda}, \quad c_{k+1} = \frac{\lambda}{k+1}, \qquad k = 0, 1, \ldots$$

- $X \sim \mathrm{NB}(r, p)$, for $r > 0$ and $0 < p < 1$, has probability function

$$p_k = \frac{\Gamma(r+k)}{\Gamma(r)k!}(1-p)^r p^k, \qquad k = 0, 1, \ldots .$$

  Using the generalized binomial coefficients, we can write

$$\begin{aligned} p_k &= \frac{\Gamma(r+k)}{\Gamma(r)\Gamma(k+1)}(1-p)^r p^k \\ &= \binom{k+r-1}{k}(1-p)^r p^k, \quad k = 0, 1, \ldots . \end{aligned}$$

Then

$$p_0 = (1-p)^r, \quad c_{k+1} = \frac{(r+k)p}{k+1}, \qquad k = 0, 1, \ldots$$

Above, we used the following property of the gamma function $\Gamma(r+k+1) = (r+k)\Gamma(r+k)$.

In a case when the function c_{k+1} is known and has a simple form, instead of ITM-d algorithm, we may use the following algorithm called ITR. Suppose X has probability function (p_n). Then $X = l$ if and only if $p_0 + \ldots + p_{l-1} \leq U < p_0 + \ldots + p_{l-1} + p_l$ or equivalently $p_0 + \ldots + p_{l-1} \leq U < p_0 + \ldots + p_{l-1} + p_{l-1}(p_l/p_{l-1})$.

Algorithm 10 ITR; generating $X \sim (p_k)$

Input: p_0 and c_{k+1}, $k = 0, 1, \ldots$

$S = p_0$
$P = p_0$
$X = 0$
Generate $U \sim \mathcal{U}[0, 1)$
while $U > S$ **do**
    $X = X + 1$
    $P = P \cdot c_X$
    $S = S + P$
end while
return X

3 Ad hoc methods.

We now present a few methods that leverage special properties of the distributions.

3.1 B(n, p)

Let $X \sim \mathrm{B}(n, p)$. Then

$$X \stackrel{\mathcal{D}}{=} \sum_{j=1}^{n} \xi_j,$$

where $\xi_1, \ldots, \xi_n$ is a sequence of Bernoulli trials, i.e., i.i.d. random variables such that $\mathbb{P}(\xi_j = 1) = p$ and $\mathbb{P}(\xi_j = 0) = 1 - p$.

Algorithm 11 DB; generating $X \sim \mathrm{B}(n, p)$

Input: p and n – success probability and number of trials
Output: $X \sim \mathrm{B}(n, p)$

```
X = 0
for i = 1 : n do
    Generate U ~ U[0, 1)
    if U ≤ p then
        X = X + 1
    end if
end for
return X
```

We recommend the reader compare the efficiency of the specialized Algorithm DB with a general Algorithm ITR.

3.2 Erl(n, λ)

Let $Y_1, \ldots, Y_n$ be i.i.d. random variables with the common exponential distribution $\mathrm{Exp}(\lambda)$. Then

$$X = Y_1 + \ldots + Y_n$$

has the Erlang distribution $\mathrm{Erl}(n, \lambda)$. It is known that Erlang distribution is a special case of $\mathrm{Gamma}(n, \lambda)$ with p.d.f.

$$f(x) = \frac{1}{\Gamma(n)} \lambda^n x^{n-1} e^{-\lambda x}, \qquad x \geq 0 \ .$$

Since a random number with the exponential distribution $\mathrm{Exp}(\lambda)$ is generated by $-\log(\texttt{rand})/\lambda$, a random number with Erlang distribution $X \sim \mathrm{Erl}(n, \lambda)$ can be generated by

$$\begin{aligned} X &= -\frac{1}{\lambda}(\log U_1 + \ldots + \log U_n) \\ &= -\log(U_1 \cdot \ldots \cdot U_n)/\lambda \ , \end{aligned}$$

where $U_1 \cdot \ldots \cdot U_n$ are i.i.d. uniformly distributed. Note that there is a risk of occurrence a floating-point error when n is large.

By the way, the tail of $\mathrm{Erl}(\lambda, n)$ can be expressed as (see Exercise III.T.15)

$$\begin{aligned}\bar{F}(x) &= 1 - F(x) = P(X > x) \\ &= e^{-\lambda x}\left(1 + \lambda x + \frac{(\lambda x)^2}{2!} + \ldots + \frac{(\lambda x)^{n-1}}{(n-1)!}\right). \end{aligned} \tag{3.5}$$

This identity will be useful in a moment.

3.3 Poi(λ)

Consider $X \sim \mathrm{Poi}(\lambda)$. Let $\tau_1, \tau_2, \ldots$ be i.i.d. random variables with the common distribution Exp(1). Consider the renewal process $\sigma_1 = \tau_1, \sigma_2 = \tau_1 + \tau_2, \ldots$ and count the number N of σ_j points falling into interval $(0, \lambda]$; see Fig. 3.3.

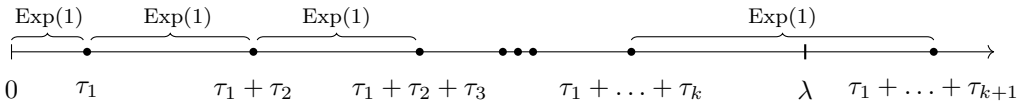


Figure 3.3: Renewal process and the Poisson distribution, see description in the text

We can write

$$N = \#\{i = 1, 2, \ldots : \tau_1 + \ldots + \tau_i \le \lambda\}. \tag{3.6}$$

If all the points are above λ, then set $N = 0$.

Proposition 3.1 *We have*

$$N \stackrel{\mathcal{D}}{=} \mathrm{Poi}(\lambda).$$

Proof We have $\mathbb{P}(N = 0) = \mathbb{P}(\tau_1 > \lambda) = e^{-\lambda}$. Moreover, note the following equivalence

$$\{N \le k\} = \left\{\sum_{j=1}^{k+1} \tau_j > \lambda\right\}.$$

Since (see (3.5))

$$\mathbb{P}\left(\sum_{j=1}^{k+1} \tau_j > x\right) = e^{-x}\left(1 + x + \frac{x^2}{2!} + \ldots + \frac{x^k}{k!}\right)$$

we have

$$\begin{aligned}\mathbb{P}(N=k) &= \mathbb{P}(\{N\le k\})-\mathbb{P}(\{N\le k-1\})\\ &= \mathbb{P}\left(\sum_{j=1}^{k+1}\tau_j>\lambda\right)-\mathbb{P}\left(\sum_{j=1}^{k}\tau_j>\lambda\right)\\ &= \frac{\lambda^k}{k!}e^{-\lambda}.\end{aligned}$$

□

Formula (3.6) can be rewritten as follows

$$N=\begin{cases}0, & \tau_1>\lambda\\ \max\{i\ge 1:\tau_1+\ldots+\tau_i\le\lambda\}, & \tau_1\le\lambda.\end{cases} \tag{3.7}$$

Based on this, we can write the following algorithm.

Algorithm 12 DP; generating $X\sim\text{Poi}(\lambda)$

Input: $\lambda>0$
Output: $X\sim\text{Poi}(\lambda)$
1: $X=0$
2: Generate $U\sim\mathcal{U}[0,1)$
3: $S=-\log(U)$
4: **while** $S<\lambda$ **do**
5: Generate $U\sim\mathcal{U}[0,1)$
6: $Y=-\log(U)$
7: $S=S+Y$
8: $X=X+1$
9: **end while**
10: **return** X

The algorithm DP can be further simplified using the following calculations. If $\tau_1>\lambda$, then $-\log U_1<\lambda$, which is equivalent to $U_1<\exp(-\lambda)$.

Furthermore formula (3.7) can be rewritten as follows:

$$\begin{aligned} N &= \max\{n \geq 0 : \tau_1 + \ldots + \tau_n \leq \lambda\} \\ &= \max\{n \geq 0 : (-\log U_1) + \ldots + (-\log U_n) \leq \lambda\} \\ &= \max\{n \geq 0 : \log \prod_{i=1}^{n} U_i \geq -\lambda\} \\ &= \max\{n \geq 0 : \prod_{i=1}^{n} U_i \geq e^{-\lambda}\} \,. \end{aligned}$$

In the above, the convention is that if $n = 0$, then $\tau_1 + \ldots + \tau_n = 0$ (or $\prod_{i=1}^{0} u_i = 1$.

3.4 (Central) chi-square distribution

If Z has a standard normal distribution $\mathcal{N}(0,1)$, then a simple argument demonstrates that Z^2 has the Gamma$(1/2, 1/2)$ distribution. Supposing random variables $Z_1, \ldots, Z_n$ are i.i.d. with the common distribution $\mathcal{N}(0,1)$, then

$$X^2 \stackrel{\mathcal{D}}{=} Z_1^2 + \ldots + Z_n^2 \tag{3.8}$$

has Gamma$(n/2, 1/2)$ distribution. Such a distribution is said to be *chi-square* with n degrees of freedom. This distribution is fundamental in statistics. We will learn how to generate a random number with standard normal distribution in due course. However, note that for even $n = 2k$, the distribution Gamma$(k, 1/2)$, which is Erlang Erl$(k, 1/2)$, which we already know how to simulate (see Section 3.2)

4 Uniform distribution

Consider a random vector $\mathbf{X}$ assuming values in $A \subset \mathbb{R}^d$. Let $\operatorname{Vol} D$ denote the volume of a subset D. For example, in case $d = 1$ it is the length, in case $d = 2$ it is the area, etc. We will assume that $\operatorname{Vol} A < \infty$. A random vector $\mathbf{X}$ is said to have a uniform distribution $\mathcal{U}[A)$ in $A \subset \mathbb{R}^n$, if

$$\mathbb{P}(\mathbf{X} \in D) = \frac{\operatorname{Vol} D}{\operatorname{Vol} A}, \qquad D \subset A \,.$$

Sometimes it is useful to write in the form of the p.d.f

$$f(\mathbf{x}) = \frac{1}{\operatorname{Vol} A}\mathbf{1}(x \in A)\, d\mathbf{x}.$$

In this case we write $\mathbf{X} \sim \mathcal{U}(A)$ and say that $\mathbf{X}$ is uniformly distributed in A. Recall that at our disposal, there is a random number U uniformly distributed $\mathcal{U}[0,1)$ in the interval $(0,1)$. If one needs a random number $V \sim \mathcal{U}[a,b)$ uniformly distributed in interval (a,b), then we have to transform linearly random number U:

$$V = (b-a)U + a\ .$$

Furthermore, if one needs a random vector $\boldsymbol{V} = (V_1, \ldots, V_d)$ uniformly distributed in $\prod_{i=1}^{d}(a_i, b_i)$, then we may use independent random variables $V_i \stackrel{\mathcal{D}}{=} (b_i - a_i)U_i + a_i$ $(i = 1, \ldots, d)$, where $U_1, \ldots, U_d$ are i.i.d. $\mathcal{U}[0,1)$ distributed random variables. We leave the reader a short justification for this fact.

To generate a random vector uniformly distributed in general area A, e.g. a ball or within an ellipsoid, let us make one observation. Suppose that A and B are subsets of $\mathbb{R}^n$, for which there exist volumes $\operatorname{Vol} B = \int_B d\mathbf{x} < \infty$ and $\operatorname{Vol} A = \int_A d\mathbf{x}$ and let $A \subset B$. The key to writing an algorithm for generating $\mathbf{Y} \sim \mathcal{U}(A)$ is the following fact.

Proposition 4.1 *Let $A \subset B$ and $\operatorname{Vol} B < \infty$. If $\mathbf{X}$ is uniformly distributed $\mathcal{U}[B)$, then $\mathbf{Y} \stackrel{\mathcal{D}}{=} (\mathbf{X}|\mathbf{X} \in A)$ is uniformly distributed $\mathcal{U}(A)$.*

Proof We use that for $D \subset A$

$$\begin{aligned} \mathbb{P}(\mathbf{Y} \in D) &= \mathbb{P}(\mathbf{X} \in D | \mathbf{X} \in A) \\ = \frac{\mathbb{P}(\mathbf{X} \in D, \mathbf{X} \in A)}{\mathbb{P}(\mathbf{X} \in A)} &= \frac{\frac{\operatorname{Vol} D}{\operatorname{Vol} B}}{\frac{\operatorname{Vol} A}{\operatorname{Vol} B}} = \frac{\operatorname{Vol} D}{\operatorname{Vol} A}. \end{aligned}$$

Hence we have the following algorithm for generating a random vector $\mathbf{X} = (X_1, \ldots, X_d)^T$ uniformly distributed $\mathcal{U}(A)$, where A is a bounded subset of $\mathbb{R}^d$. The boundedness of the set A means that it can be contained in a rectangle $B = [a_1, b_1] \times \ldots \times [a_d, b_d]$. We can easily generate a random vector $(V_1, \ldots, V_n)$ uniformly distributed $\mathcal{U}(B)$, because components V_i are independent and uniformly distributed $\mathrm{U}[a_i, b_i]$ $(i = 1, \ldots, d)$ respectively.

Algorithm 13 $\mathcal{U}(A)$; generating $X \sim \mathcal{U}(A)$

Input: A rectangle $B = [a_1, b_1] \times \ldots \times [a_n, b_n]$ containing A
Output: $X \sim \mathcal{U}(A)$

1: Generate i.i.d. $U_1, \ldots, U_d$, $U_i \sim \mathcal{U}[0, 1)$
2: **for** $i = 1$ **to** d **do**
3: $\quad V_i = (b_i - a_i) \cdot U_i + a_i$
4: **end for**
5: **if** $V = (V_1, \ldots, V_d) \in A$ **then**
6: $\quad X = V$
7: **else**
8: $\quad$ **goto** 1
9: **end if**
10: **return** X

Uniform distribution in the ball B_d. Algorithm 13 is inefficient for sampling a point from a unit ball for large dimension d. Namely, define d-ball

$$B_d = \{(x_1, \ldots, x_d) : x_1^2 + \ldots + x_d^2 \le 1\},$$

and d-cube $A_d = [-1, 1]^d$. The volume of d-ball is

$$\operatorname{Vol} B_d = \frac{\pi^{d/2}}{\Gamma(\frac{d}{2} + 1)}.$$

Say $d = 2k$, then

$$\operatorname{Vol} B_d = \frac{\pi^k}{k!}, \quad \operatorname{Vol} A_d = 2^{2k}$$

and the probability that $V \in A$ (line 5 in Algorithm 13) is $\frac{\pi^k}{2^{2k} k!}$. Note that from the Stirling's formula $k! \sim (2\pi k)^{1/2}(k/e)^k$, this probability is asymptotically $(2\pi k)^{-1/2} \left(\frac{\pi e}{4k}\right)^k$, that is, converges to 0 exponentially fast, e.g., it is 2.0524×10^{-14} for $d = 30$.

5 Composition method

The method is based on an assumption the distribution of the random number X is a mixture of distributions. There are possible different form of

mixing distributions. In the case when X has a p.d.f. $f(x)$ it is

$$f(x) = \sum_{j=1}^{n} p_j g_j(x),$$

where $g_j(x)$ are p.d.f.s (n can be infinity ∞) and (p_j) is a probability function. Suppose further that we can simulate random variables with these p.d.f.s. To generate a random number with a p.d.f. $f(x)$, we first draw J with the probability function (p_j) and if $J = j$, then we draw a random number with a p.d.f. $g_j(x)$, and this quantity X is the finally generated number.

This method has a wider application in simulation than it could be expected. A significant case is when

$$f(x) = \sum_{j=0}^{\infty} p_j(\lambda) g_j(x), \tag{5.9}$$

where $p_j(\gamma) = \frac{\gamma^j}{j!} e^{-\gamma}$ is the Poisson probability function and $g_j(x)$ is a p.d.f. of gamma distribution Gamma(α_j, β).

Example 5.1 [Noncentral chi-squared distribution] The chi-squared distribution appears in statistics, financial mathematics and others. The standard one has only one parameter- the number of freedom d- a natural number. The noncentral one has two parameters: the noncentrality parameter $\lambda > 0$ and the number of freedom $d > 0$. For integer d, the noncentral chi-square distribution is the one of $(Z_1 + a_1)^2 + \ldots + (Z_d + a_d)^2$, where $Z_1, \ldots, Z_d$ are i.i.d. standard normal random variables. In this case, the non-centrality parameter $\lambda = a_1^2 + \ldots + a_d^2$. It turns out (see, e.g., [88]) that the p.d.f. $f(x)$ can be expressed as the function of the form as (5.9) with

$$p_j(\lambda) = \frac{(\frac{\lambda}{2})^j}{j!} e^{-\lambda/2}, \qquad g_j(x) \sim \text{Gamma}\left(j + \frac{d}{2}, \frac{1}{2}\right). \tag{5.10}$$

It can be expressed by

$$\chi^{'2} = X_N,$$

where N is Poisson Poi$(\frac{\lambda}{2})$) and X_j is Gamma$\big(j + \frac{d}{2}, \frac{1}{2}\big)$ and N is independent from (X_j). One can notice that p.d.f. $\sum_{j=0}^{\infty} p_j g_j(x)$ has a sense for general $d > 0$, so we get the most general definition of noncentral chi-square. It is clear that for $p_j(\lambda) \to 0$ as $\lambda \to 0$ for $j \neq 0$ and 1 for $j = 0$ and thus this definition contains as its boundary case the classical central chi-square distribution. □ □

Example 5.2 [Dagpunar [26]] In atomic physics, one considers a distribution with the following p.d.f.:

$$f(x) = C \sinh(\sqrt{xc}) e^{-bx}, \qquad x \geq 0$$

where $b, c > 0$ are fixed parameters and

$$C = \frac{2b^{3/2} e^{-c/(4b)}}{\sqrt{\pi c}} \ .$$

Expanding $\sinh((xc)^{1/2})$ into a series we obtain

$$\begin{aligned} f(x) &= C \sum_{j=0}^{\infty} \frac{(xc)^{(2j+1)/2} e^{-bx}}{(2j+1)!} \\ &= C \sum_{j=0}^{\infty} \frac{\Gamma(j+\frac{2}{3}) c^{j+\frac{1}{2}}}{b^{j+\frac{3}{2}}(2j+1)!} \frac{b^{j+3/2} x^{j+\frac{1}{2}} e^{-bx}}{\Gamma(j+\frac{2}{3})} \\ &= \sum_{j=0}^{\infty} C \frac{\Gamma(j+\frac{2}{3}) c^{j+\frac{1}{2}}}{b^{j+\frac{2}{3}}(2j+1)!} g_j(x) \ , \end{aligned}$$

where $g_j(x)$ is the p.d.f. of the distribution Gamma$(j+\frac{3}{2}, b)$. Using now that

$$\Gamma\left(j+\frac{3}{2}\right) = \left(j-1+\frac{3}{2}\right)\left(j-2+\frac{3}{2}\right)\ldots\left(1+\frac{3}{2}\right)\left(\frac{3}{2}\right)\Gamma\left(\frac{3}{2}\right)$$

(see Abramowitz & Stegun [1] formula 6.1.16) and that $\Gamma(\frac{3}{2}) = \pi^{\frac{1}{2}}/2$ (formula 6.1.9) we have

$$\begin{aligned} f(x) &= C \sum_{j=0}^{\infty} \frac{\pi^{1/2} c^{j+\frac{1}{2}}}{2^{2j+1} j! b^{j+\frac{3}{2}}} g_j(x) \\ &= \sum_{j=0}^{\infty} \frac{(c/4b))^j}{j!} e^{-c/(4b)} g_j(x) \ , \end{aligned}$$

thus we obtain a mixture of p.d.f.s of the distribution Gamma$(j+3/2, b)$, and (p_j) being Poisson distribution with parameter $c/(4b)$. It is still worth noting that if Y is a random number with the distribution Gamma$(j+\frac{3}{2}, \frac{1}{2})$, then $Y/(2b)$ is a random number with the distribution Gamma$(j+\frac{3}{2}, b)$. Finally, notice that Gamma$(j+\frac{3}{2}, \frac{1}{2})$ is a chi-square distribution with $2j+1$ degrees of freedom. We showed earlier how to generate such random numbers easily. □

Another form is when probability function $(p_n(\lambda))$ has a parameter λ. The following is an important example.

Example 5.3 Let $p_n(x) = \frac{x^n}{n!}e^{-x}$ for $n = 0, 1, \ldots$ be the Poisson distribution with parameter x. Suppose that this parameter comes from a random variable $X > 0$ with Gamma(α, λ) distribution. Then unconditional probability function is

$$p_n = \int_0^\infty \frac{x^n}{n!}e^{-x}\frac{\lambda^\alpha x^{\alpha-1}}{\Gamma(\alpha)}e^{\lambda x}\,dx, \qquad n = 0, \ldots.$$

From the definition of gamma distribution we obtain

$$\begin{aligned} p_n &= \frac{\Gamma(\alpha+n)}{\Gamma(\alpha)\Gamma(n+1)}\left(\frac{1}{1+\lambda}\right)^\alpha\left(\frac{1}{1+\lambda}\right)^n \\ &= \binom{n+\alpha-1}{n}\left(\frac{1}{1+\lambda}\right)^\alpha\left(\frac{1}{1+\lambda}\right)^n, \quad n = 0, 1, \ldots, \end{aligned}$$

that is the mixture is NB$(\alpha, 1/(1+\lambda))$ negative binomial distribution. □

6 Acceptance-rejection method

We now show a general method for generating a random number, introduced by John von Neumann. This is the most adaptable method for sampling from complicated distributions under some mild assumptions. The method is called the *acceptance-rejection method*.

6.1 AR-d method

We begin by presenting the method for discrete distributions, which we will abbreviate by AR-d (Acceptance-Rejection-discrete). Suppose we want to generate a random number X with probability function $(p_j, j \in E)$, where E is a finite or denumerable space of values of X (for example $\mathbb{Z}$, $\mathbb{Z}_+$,$\mathbb{Z}^2$, etc.), while we know how to generate another random number Y with a probability function $(q_j, j \in E)$ such that there exists $c > 0$ for which

$$p_j \le cq_j, \qquad \forall (j \in E).$$

If E is finite, then such the c always exists. We leave it to the reader to think about infinite cases when such c does not exist. We will also see them

in due course. Clearly, $c > 1$ unless both the probability functions are the same (why?).

In the algorithm below, we will generate independently $U \sim \mathcal{U}[0,1)$ and Y with a probability function $(q_j, j \in E)$ and define the so-called acceptance region by

$$A = \{cUq_Y \leq p_Y\} = \left\{Uq_Y \leq \frac{1}{c}p_Y\right\}.$$

The basis of our algorithm will be the following fact.

Proposition 6.1 *We have*

$$\mathbb{P}(A) = 1/c \text{ and } \mathbb{P}(Y = j|A) = p_j \quad \text{for} \quad j \in E.$$

Proof Without loss of generality we may assume that $q_j > 0$ for all $j \in E$. Simply, if for some j we would have $q_j = 0$, then also $p_j = 0$ and this state can be eliminated. Furthermore, we have

$$\begin{aligned}
\mathbb{P}(Y = j|A) &= \frac{\mathbb{P}(\{Y = j\} \cap \{cUq_Y \leq p_Y\})}{\mathbb{P}(A)} \\
&= \frac{\mathbb{P}(Y = j, U \leq p_j/(cq_j))}{\mathbb{P}(A)} \\
&= \frac{\mathbb{P}(Y = j)\mathbb{P}(U \leq p_j/(cq_j))}{\mathbb{P}(A)} \\
&= q_j \frac{p_j/(cq_j)}{\mathbb{P}(A)} = \frac{p_j}{c\mathbb{P}(A)}\,.
\end{aligned}$$

Now we have to take advantage of the fact that

$$1 = \sum_{j \in E} \mathbb{P}(Y = j|A) = \frac{1}{c\mathbb{P}(A)},$$

which means that $\mathbb{P}(A) = 1/c$ and thus $\mathbb{P}(Y = j|A) = p_j$.

□

The following algorithm gives a recipe for generating a random number X with probability function $(p_j, j \in E)$.

Algorithm 14 AR-d; generating $X \sim (p_j)$

Input: Constant $c > 0$ such that $p_j \le c q_j$ and an algorithm for generating r.v. Y with probability function (q_j)
Output: $X \sim (p_j)$
1: **repeat**
2: Generate $Y \sim (q_j)$
3: Generate $U \sim \mathcal{U}[0,1)$ (independent from Y)
4: **until** $cUq_Y \le p_Y$
5: **return** $X = Y$

We can see that the number L of repetitions in the loop has a probability function $\mathbb{P}(L = k) = \frac{1}{c}(1-\frac{1}{c})^{k-1}$, $k = 1, 2, \ldots$, which, in our terminology, is a truncated geometric distribution TGeo($\mathbb{P}(A)$)=TGeo($1 - \frac{1}{c}$). The expected number of repeatings in the loop is c, so we have to look for the least possible c.

Example 6.2 We want to simulate X with distribution $p_j = \frac{6}{\pi^2}\frac{1}{j^2}, j = 1, \ldots$. As a proposal distribution, let us take Y with distribution $q_j = \frac{1}{j(j+1)}, j = 1, \ldots$, which can be simply generated as $Y = \lfloor U^{-1} \rfloor$ for $U \sim \mathcal{U}[0,1)$ (see Exercise III.T.17). We have

$$\frac{p_j}{q_j} = \frac{6}{\pi^2}\frac{1}{j^2} \cdot j(j+1) = \frac{6}{\pi^2}\frac{j+1}{j}.$$

We take $c = \max(\frac{p_j}{q_j}) = 12/\pi^2$. Let us expand step 4 of the AR-d Algorithm 14:

$$cUq_Y \le p_Y \quad \Longleftrightarrow \quad \frac{12}{\pi^2} U \frac{1}{Y(Y+1)} \le \frac{6}{\pi^2}\frac{1}{Y^2} \quad \Longleftrightarrow \quad 2UY \le Y + 1$$

Summarizing, the procedure is given in Algorithm 15.

Algorithm 15 AR-d for Example 6.2

Output: $X \sim (p_j), p_j = 6/(\pi^2 j^2)$
1: **repeat**
2: Generate independent $U, U_0 \sim \mathcal{U}[0,1)$ and set $Y = \lfloor U_0^{-1} \rfloor$
3: **until** $2UY \le Y + 1$
4: **return** $X = Y$

□

6.2 AR-c method

We now present a continuous version of the acceptance-rejection method (AR-c). Suppose we want to generate a random number X with a p.d.f. $f(x)$ on E (for example, $E = \mathbb{R}_+, \mathbb{R}, \mathbb{R}^d$, etc.), while we know how to generate a random number Y assuming values also in E with a p.d.f. $g(x)$. Let $c < \infty$ be such that

$$f(x) \le cg(x), \qquad x \in E.$$

Clearly, $c > 1$ unless $f = g$. Similarly, as it was in a discrete case, without loss of generality, we may assume that if for some $x \in E$, $g(x) = 0$, then also $f(x) = 0$.

In the algorithm below, we generate $U \sim \mathcal{U}[0,1)$ and independently Y with the p.d.f. $g(x)$. Similarly (as in discrete case), we define acceptance region as

$$A = \{cUg(Y) \le f(Y)\} = \left\{Ug(Y) \le \frac{1}{c} f(Y)\right\}.$$

The basis of our algorithm will be the following fact, which we state for $E \subset \mathbb{R}^n$. Proposition 6.1 for continuous case has the following form:

Proposition 6.3

$$\mathbb{P}(A) = 1/c \qquad \textit{and} \quad \mathbb{P}(Y \in B|A) = \int_B f(y)\,dy \qquad \text{for} \quad B \subset E.$$

Proof The joint distribution of (U, Y) has a p.d.f. $du \times g(y)\,dy$ on $[0,1) \times E$. We have

$$\begin{aligned}
\mathbb{P}(Y \in B|A) &= \frac{\mathbb{P}(Y \in B, cUg(Y) \le f(Y))}{\mathbb{P}(A)} \\
&= \frac{\int_E \int_{[0,1)} \mathbf{1}(y \in B, cug(y) \le f(y))\,du\,g(y)\,dy}{\mathbb{P}(A)} \\
&= \frac{\int_B (f(y)/(cg(y)))\,g(y)\,dy}{\mathbb{P}(A)} \\
&= \int_B f(y)\,dy \frac{1}{c\mathbb{P}(A)}.
\end{aligned}$$

Substituting $B = E$, we can see that $\mathbb{P}(A) = 1/c$. Since the equation holds for all $B \subset E$, we conclude that f is the desired p.d.f.. □

The following algorithm gives a recipe for generating a random number X with p.d.f. $(f(x), x \in E)$.

Algorithm 16 AR-c; generating $X \sim f(x)$

Input: Constant $c > 0$ such that $f(x) \le cg(x)$ and an algorithm for generating r.v. Y with p.d.f. $g(x)$
Output: $X \sim f(x)$
1: **repeat**
2: Generate Y and U
3: **until** $cUg(Y) \le f(Y)$
4: **return** $X = Y$

Example 6.4 [Generating a normal random number] A popular method for generating a standard normal random number $\mathcal{N}(0,1)$ uses algorithm AR-c. We first generate X with the half-normal distribution

$$f(x) = \frac{2}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}, \qquad 0 < x < \infty \,. \tag{6.11}$$

It is the p.d.f. of $|Z|$, where Z has a standard normal distribution. We can take Y to be exponentially distributed Exp(1), which has the p.d.f. $g(x) = e^{-x}, x > 0$. We need to find the maximum of a function

$$\frac{f(x)}{g(x)} = \frac{2}{\sqrt{2\pi}} e^{x-\frac{x^2}{2}}, \qquad 0 < x < \infty \,,$$

which is equivalent to finding the maximum of $x - x^2/2$ in the region $x > 0$. The maximum is achieved at $x = 1$, which gives us

$$c = \max \frac{f(x)}{g(x)} = \sqrt{\frac{2e}{\pi}} \approx 1.32 \,.$$

Acceptance region is defined by $U \le f(Y)/(cg(Y))$. Notice that in the simulation, we can use

$$\frac{f(x)}{cg(x)} = e^{x-x^2/2-1/2} = e^{-(x-1)^2/2} \,.$$

To generate a standard normal random number $Z \sim \mathcal{N}(0,1)$, we need first to generate X with p.d.f. $f(x)$ and then draw $+1$ with probability $1/2$ and -1 with probability $1/2$ (see Exercise III.T.28). Based on this, we can write the following algorithm to generate a random number $\mathcal{N}(0,1)$. □

Algorithm 17 AR-N ; generating $X \sim \mathcal{N}(0,1)$

Output: $X \sim \mathcal{N}(0,1)$

repeat
  Generate $U \sim \mathcal{U}[0,1)$
  $Y = -\log(U)$
  Generate $U \sim \mathcal{U}[0,1)$
until $U \leq \exp(-(Y-1)^2/2)$
Generate $U \sim \mathcal{U}[0,1)$
Generate $I = 2 \cdot \mathrm{floor}(2 \cdot U) - 1$
return $Z = IY$

Example 6.5 [Distribution Gamma(α, λ); Madras [88]]

Case $\alpha = 1$. This is Exp(λ) distribution; we know how to simulate it.

Case $\alpha < 1$. Without loss of generality, we may assume that $\lambda = 1$ (why?). If you wish to generate a random variable with the Gamma(α, β) distributuion, it is sufficient to generate an r.v. with the distribution Gamma$(\alpha, 1)$ – see Exercise III.T.28). Let us consider the following p.d.f.:

$$g(x) = \begin{cases} Kx^{\alpha-1}, & 0 < x < 1, \\ Ke^{-x}, & x \geq 1. \end{cases}$$

The normalising condition $\int_0^\infty g(x)\,dx = 1$ yields

$$K = \frac{\alpha e}{\alpha + e}\ .$$

It is convenient to write

$$g(x) = K(x^{\alpha-1}\mathbf{1}(0 < x < 1) + e^{-x}\mathbf{1}(x \geq 1)).$$

Set $g_1(x) = \alpha x^{\alpha-1}\mathbf{1}(0 < x < 1)$, $g_2(x) = ee^{-x}\mathbf{1}(x \geq 1)$, since

$$\int_0^1 x^{\alpha-1}\,dx = \frac{1}{\alpha}, \qquad \int_1^\infty e^{-x}\,dx = \frac{1}{e}.$$

We have

$$g(x) = p_1 g_1(x) + p_2 g_2(x)\ ,$$

where $p_1 = e/(\alpha+e)$ and $p_2 = \alpha/(\alpha+e)$. We can thus simulate Y with p.d.f. g using the composition method. The reader is invited to write a procedure for generating a random number with p.d.f.s $g_1(x)$ and $g_2(x)$ respectively. Notice that there exists c such that

$$f(x) \leq cg(x), \qquad x \in \mathbb{R}_+.$$

To compute c we will look for

$$\begin{aligned} \min_{x>0} \frac{g(x)}{f(x)} &= \min_{x>0} \frac{K(x^{\alpha-1}\mathbf{1}(0<x<1) + e^{-x}\mathbf{1}(x \geq 1))}{x^{\alpha-1}e^{-x}/\Gamma(\alpha)} \\ &= \min_{x>0} K\Gamma(\alpha)(e^x\mathbf{1}(0<x<1) + x^{1-\alpha}\mathbf{1}(x \geq 1)) = K\Gamma(\alpha). \end{aligned}$$

Hence $c = \frac{1}{K\Gamma(\alpha)}$ and so we can apply the AR-c algorithm to generate X. The reader is invited to complete the details and write the procedure.

Case $\alpha > 1$. Consider $g(x) = be^{-bx}\mathbf{1}(0 \leq x < \infty)$ (i.e., Exp(b)) with $b < 1$. In this case

$$c = c(b) = \sup_{x \geq 0} \frac{f(x)}{g(x)} = \sup_{x \geq 0} \frac{1}{b\Gamma(\alpha)} x^{\alpha-1}e^{-(1-b)x} < \infty \ .$$

We have first to find $x(b)$ maximizing $x^{\alpha-1}\exp(-(1-b)x)$, which is $x(b) = (\alpha-1)/(1-b)$. Hence

$$c = c(b) = \frac{1}{b\Gamma(\alpha)} \frac{(\alpha-1)^{\alpha-1}}{(1-b)^{\alpha-1}} e^{-(\alpha-1)} = \frac{1}{\Gamma(\alpha)}(\alpha-1)^{\alpha-1}e^{-(\alpha-1)} \frac{1}{b(1-b)^{\alpha-1}}$$

Next, constant b should be chosen such that $c(b)$ was minimal and so maximizing the numerator $b(1-b)^{\alpha-1}$ we obtain $b = \alpha^{-1}$. Hence the optimal c is $\alpha^\alpha e^{-(\alpha-1)}/\Gamma(\alpha)$. □

Unknown constant(s). In many (practial) scenarios, we only know f, or f and g, up to normalising constants, i.e.,

$$f(x) = \frac{1}{c_f} f^*(x), \quad g(x) = \frac{1}{c_g} g^*(x),$$

where $f^*(x)$ and $g^*(x)$ are known, but c_f and c_g are unknonw (or infeasible to compute). In the method we need to have $c \in (0, \infty)0$ such that $f(x) \leq cg(x)$ for all x, what is equivalent to $f^*(x) \leq c^* g(x)$ with $c^* = \frac{c \cdot c_f}{c_g}$. In other words, the acceptance region is

$$A = \left\{ U \leq \frac{f(Y)}{cg(Y)} \right\} = \left\{ U \leq \frac{f^*(Y)}{c^* g^*(Y)} \right\}.$$

Practially, it means that we can ignore the normalising constans from the beginning: If we can find c^* such that

$$f^*(x) \leq c^* g^*(x),$$

then it is correct to accept with probability $\frac{f^*(x)}{c^* g^*(x)}$. Note that the mean number of accept/reject trials is equal to c again (which equals $c_g c^*/c_f$). For discrete distributions the method works in a similar way, which we present in Algorithm 18.

Algorithm 18 AR-d; generating $X \sim (p_j), p_j = \frac{1}{c_p} p_j^*$

Input: Constant $c^* > 0$ such that $p_j^* \leq c^* q_j^*$ (where $q_j = \frac{1}{c_q} q_j^*$) and an algorithm for generating r.v. Y with probability function (q_j)

1: **repeat**
2: Generate $Y \sim (q_j)$
3: Generate $U \sim \mathcal{U}[0, 1)$ (independent from Y)
4: **until** $c^* U q_Y^* \leq p_Y^*$
5: **return** $X = Y$

Example 6.6 Let

$$f(x) = \frac{1}{c_f} e^{-x^2/2} \left(1 - e^{-\sqrt{x^2+1}} \right), \quad x \in \mathbb{R}$$

(c_f - a normalising constant, we do not know it). Denote

$$f(x) = \frac{1}{c_f} f^*(x)$$

and take $g(x) = (2\pi)^{-1/2} e^{-x^2/2} := \frac{1}{c_g} e^{-x^2/2}$ (p.d.f. of $\mathcal{N}(0, 1)$, which we know how to simulate). We have

$$\frac{f^*(x)}{g^*(x)} = 1 - e^{-\sqrt{x^2+1}} \leq 1 = c^*.$$

As noted, the condition $U \leq \frac{f(Y)}{cg(Y)}$ is equivalent to

$$U \leq \frac{f^*(Y)}{c^* g^*(Y)} = \frac{e^{-Y^2/2}\left(1 - e^{-\sqrt{Y^2+1}}\right)}{e^{-Y^2/2}} = 1 - e^{-\sqrt{Y^2+1}}.$$

Algorithm 19 is a summary of the method.

Algorithm 19 AR for Example 6.6

1: **repeat**
2: Generate $Y \sim \mathcal{N}(0,1)$
3: Generate $U \sim \mathcal{U}[0,1)$
4: **until** $U \leq 1 - e^{-\sqrt{Y^2+1}}$
5: **return** Y

□

At this point, we encourage the reader to devise the procedure for simulating an r.v. with a Beta distribution using AR-c method.

In context of Acceptance-Rejection method distribution p (p.d.f. f) is often called a *target distribution*, whereas distribution q (p.d.f. $g(x)$) is s often called a *proposal distribution*.

7 Generating Gaussian random numbers

7.1 Ad hoc methods for $\mathcal{N}(0,1)$.

We will present two variants of the Box-Muller method for generating pairs of independent random variables Z_1, Z_2 with the common normal distribution $\mathcal{N}(0,1)$. The idea is to use polar coordinates. We start with the following lemma.

Lemma 7.1 *Let Z_1, Z_2 be independent random variables with a common standard normal distribution $\mathcal{N}(0,1)$, and let (D, Θ) be their polar coordinates representation. Then D and Θ are independent, D has p.d.f.*

$$f(r) = re^{-r^2/2}1(r > 0) \tag{7.12}$$

and Θ is uniformly distributed $\mathcal{U}[0, 2\pi)$.

Proof Clearly

$$\begin{aligned} Z_1 &= D\cos\Theta, \\ Z_2 &= D\sin\Theta. \end{aligned}$$

The joint p.d.f. of the vector (Z_1, Z_2) is $\frac{1}{2\pi}\exp(-(x_1^2+x_2^2)/2)$. Consider the transformation $x_1 = r\cos\theta$ and $x_2 = r\sin\theta$. The Jacobian of this transformation is

$$\det\begin{pmatrix} \frac{\partial x_1}{\partial r} & \frac{\partial x_1}{\partial \theta} \\ \frac{\partial x_2}{\partial r} & \frac{\partial x_2}{\partial \theta} \end{pmatrix} = \begin{vmatrix} \cos\theta & -r\sin\theta \\ \sin\theta & r\cos\theta \end{vmatrix} = r.$$

By formula (I.2.1) from the Appendix, a random vector (D, Θ) has the joint p.d.f.

$$f_{(D,\Theta)}(r,\theta) = \frac{1}{2\pi} e^{-\frac{r^2}{2}} r 1(r>0) \qquad 0<\theta\le 2\pi, \quad r>0.$$

□

The distribution with the p.d.f. (7.12) is called the *Rayleigh distribution.*

In the following lemma we show how to generate random numbers with Rayleigh distribution.

Lemma 7.2 *If D has the Rayleigh distribution, then $Y = D^2$ is exponentially distributed Exp(1/2). Consequently, if U is uniformly distributed $\mathcal{U}[0,1)$, then random variable $(-2\log U)^{1/2}$ has the Rayleigh distribution.*

Proof Notice that $Y = D^2$ is exponentially distributed $\mathrm{Exp}(1/2)$ because

$$\begin{aligned} \mathbb{P}(D^2 > x) &= \mathbb{P}(D > \sqrt{x}) \\ &= \int_{\sqrt{x}}^{\infty} r e^{-r^2/2}\, dr = e^{-x/2}, \qquad x>0. \end{aligned}$$

Since $(-2\log U)$ is exponentially distributed $\mathrm{Exp}(1/2)$, the square root from this random variable has the Rayleigh distribution. □

As a result, we have the following fact,which is the base of the Box-Muller algorithm.

Proposition 7.3 *Let U_1 and U_2 be two independent random variables with the common uniform distribution $\mathcal{U}[0,1)$ and*

$$\begin{aligned} Z_1 &= (-2\log U_1)^{1/2}\cos(2\pi U_2), \\ Z_2 &= (-2\log U_1)^{1/2}\sin(2\pi U_2)\ . \end{aligned}$$

Then Z_1, Z_2 are independent random variables with common normal distribution $\mathcal{N}(0,1)$.

Proof We have $Z_1 = D\cos\Theta$, $Z_2 = D\sin\Theta$, where D, Θ are independent, D has Rayleigh distribution and Θ is uniformly distributed on $(0, 2\pi]$. One has to generate random number Θ uniformly distributed on $(0, 2\pi]$ (simple) and D with p.d.f. $re^{-\frac{r^2}{2}}$, $r > 0$, which we know how to generate from Lemma 7.2

Algorithm 20 BM; Generate Z_1, Z_2 with a standard normal distribution $\mathcal{N}(0,1)$

1: Generate U_1, U_2 i.i.d. $\mathcal{U}[0,1)$ random variables
2: Compute $D = -2\log(U_1), V = 2\pi U_2$
3: Set $Z_1 = \sqrt{D}\cos V,\ Z_2 = \sqrt{D}\sin V$
4: **return** Z_1, Z_2

The second method is a Box-Muller modification, attributed to Marsaglia-Brey. It is based on the following fact.

Proposition 7.4 *Let*

$$\begin{aligned} Y_1 &= \{-2\log(V_1^2+V_2^2)\}^{1/2}\frac{V_1}{(V_1^2+V_2^2)^{1/2}}, \\ Y_2 &= \{-2\log(V_1^2+V_2^2)\}^{1/2}\frac{V_2}{(V_1^2+V_2^2)^{1/2}}, \end{aligned}$$

and V_1, V_2 are independent random variables with a common uniform distribution $\mathcal{U}[-1,1)$. Then

$$(Z_1, Z_2) \stackrel{\mathcal{D}}{=} ((Y_1, Y_2)|V_1^2 + V_2^2 \le 1)$$

is a pair of independent random variables with the common normal distribution $\mathcal{N}(0,1)$.

The proof of the above is based on the following lemma. Let B_2 be a ball with radius 1 centered at the origin of $\mathbb{R}^2$.

Lemma 7.5 *If $(W_1, W_2) \sim \mathcal{U}[B_2)$, and (D', Θ') is its a representation of a point (W_1, W_2) in polar coordinates, then D' and Θ' are independent, ${D'}^2$ has a uniform distribution $\mathcal{U}[0, 1)$, and Θ' is uniformly distributed $\mathcal{U}[0, 2\pi)$.*

Proof The joint distribution of (W_1, W_2) has p.d.f.

$$\frac{1}{\pi}\mathbf{1}((w_1, w_2) \in B_2)\, dw_1\, dw_2$$

and in polar coordinates

$$W_1 = D' \cos\Theta', \qquad W_2 = D' \sin\Theta' \ .$$

We will use Proposition I.2.1. Thus we have a 1-1 mapping $\boldsymbol{g} : B_2 \setminus \{(0,0)\} \to (0,1)\times(0,2\pi)$ and its inverese $\boldsymbol{h} = \boldsymbol{g}^{-1}$. In this case the Jacobian $J_{\boldsymbol{h}}(r,\theta) = r$ for $0 < r < 1$ and $0 \le \theta \le 2\pi]$. Let $A = \{t_1 < \theta \le t_2, a < r \le b\}$; see Fig. 7.4 (left). Then

$$\begin{aligned}
\mathbb{P}(t_1 < \Theta' \le t_2, a < D' \le b) &= \mathbb{P}((W_1, W_2) \in \boldsymbol{h}(A)) \\
&= \frac{1}{\pi}\int_{B_2} 1((x,y) \in \boldsymbol{h}(A))\, dx\, dy \\
&= \frac{1}{\pi}\int_{t_1}^{t_2}\int_a^b r\, dr\, d\theta \\
&= \frac{1}{2\pi}(t_2 - t_1) \times (b^2 - a^2) \\
&= \mathbb{P}(t_1 < \Theta' \le t_2)\mathbb{P}(a < D' \le b).
\end{aligned}$$

Hence we have that D' and Θ' are independent, Θ' is uniformly distributed $\mathcal{U}[0, 2\pi)$. Furthermore $\mathbb{P}(D' \le x) = x^2\mathbf{1}(0 \le x \le 1)$, that is $(D')^2$ is uniformly distributed $\mathcal{U}[0, 1)$.

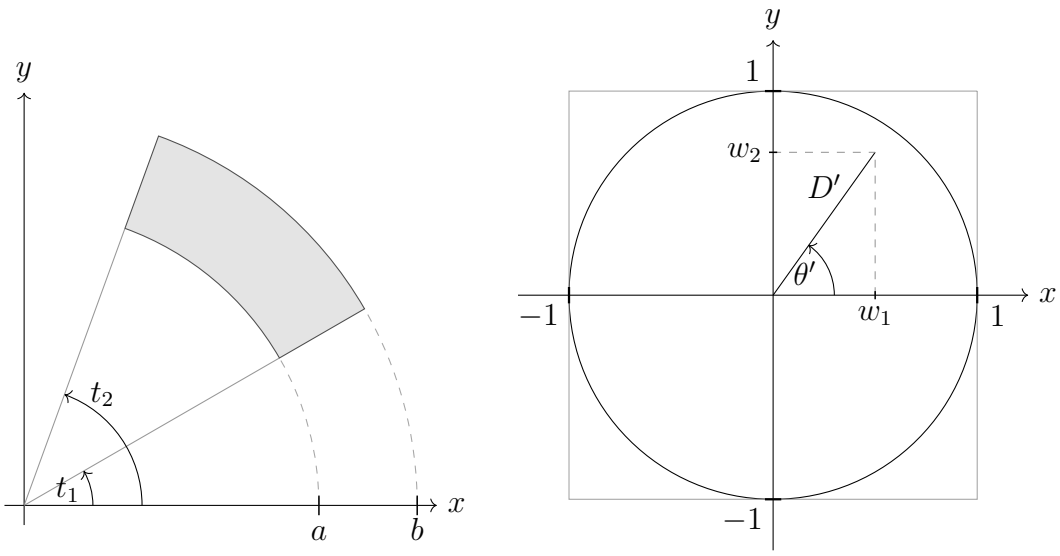


Figure 7.4: Illustration for Lemma 7.5.

Proof of Proposition 7.4 is based on an observation that, with notations from Lemma 7.5, under condition $V_1^2 + V_2^2 \le 1$, we have $(D')^2 = V_1^2 + V_2^2 \sim \mathcal{U}[0,1)$ and $\cos\Theta' = V_1/(V_1^2+V_2^2)^{1/2}$, i.e.

$$\left((V_1^2+V_2^2)^{1/2}, \frac{V_1}{(V_1^2+V_2^2)^{1/2}}\right) \stackrel{\mathcal{D}}{=} (D', \cos\Theta'),$$

where $(D')^2$ has distribution $\mathcal{U}[0,1)$ and $\Theta' \sim \mathcal{U}[0,2\pi)$; see Fig. 7.4 (right).

Algorithm 21 MB; generate Z_1, Z_2 with a standard normal distribution $\mathcal{N}(0,1)$

repeat
  Generate U_1, U_2 i.i.d. $\mathcal{U}[0,1)$ random variables
  Compute $V_1 = 2U_1 - 1, V_2 = 2U_2 - 1$
  Set $X = V_1^2 + V_2^2$
until $X \le 1$
$Y = \sqrt{(-2\log X)/X}$
Set $Z_1 = V_1 \cdot Y, Z_2 = V_2 \cdot Y$
return Z_1, Z_2

7.2 ITM Method for normal random variables

To generate standard normal numbers $\mathcal{N}(0,1)$ with the ITM method, we need to invert the c.d.f. $\Phi(x)$. Unfortunately, there is no explicit formula for Φ^{-1}, therefore, numerical methods are needed. Various algorithms exist for computing $\Phi^{-1}(u)$. As is customary with the use of numerical algorithms, we should be aware of their errors.

The problem is to solve the equation $\Phi(x) = u$ for a given $u \in (0,1)$. The starting point can be the Newton's root-finding algorithm for the function $x \to \Phi(x) - u$. We first make some simplifications, namely that since

$$\Phi^{-1}(1-u) = -\Phi^{-1}(u),$$

it suffices to search either for u either in $[1/2, 1)$ or for $(0, 1/2)$. *Newton's methods* is an iterative method given by recursion

$$\begin{aligned} x_{n+1} &= x_n - \frac{\Phi(x_n) - u}{\phi(x_n)} \\ &= x_n + (u - \Phi(x_n))e^{0.5x_n^2 + c}, \end{aligned} \tag{7.13}$$

where $c = \log\sqrt{2\pi}$. It is suggested to start with a point

$$x_0 = \pm\sqrt{|-1.6\log(1.0004 - (1-2u)^2)|}.$$

The sign $\pm$ depends on whether $u \geq 1/2$ (plus) or $u < 1/2$ (minus). The method requires the knowledge of function Φ and the exponential function.

In Algorithm 22, another method, the so-called Beasley-Springer-Moro algorithm for approximating the inverse of the normal distribution, is presented (following Glasserman [48])

Algorithm 22 Beasley-Springer-Moro algorithm for approximating the inverse normal $x = \Psi^{-1}(u)$

Input: $u \in (0,1)$

$a_0 = 2.50662823884;\ a_1 = -18.61500062529;\ a_2 = 41.39119773534;$
$a_3 = -25.44106049637;\ b_0 = -8.47351093090;\ b_1 = 23.08336743743;$
$b_2 = -21.06224101826;\ b_3 = 3.13082909833;\ c_0 = 0.3374754822726147$
$c_0 = 0.3374754822726147;\ c_1 = 0.9761690190917186;\ c_2 = 0.1607979714918209;$
$c_3 = 0.0276438810333863;\ c_4 = 0.0038405729373609;\ c_5 = 0.0003951896511919;$
$c_6 = 0.0000321767881768;\ c_7 = 0.0000002888167364;\ c_8 = 0.0000003960315187;$
$y = u - 0.5$
if $|y| < 0.42$ **then**
    $r = y^2$
    $x = \frac{((a_3 r + a_2) r + a_1) r + a_0}{(((b_3 r + b_2) r + b_1) r + b_0) + 1}$
else
    $r = u$
    if $(y > 0)$ **then**
        $r = 1 - u$
    end if
    $r = \log(-\log(r))$
    $x = c_0 + r(c_1 + r(c_2 + r(c_3 + r(c_4 + r(c_5 + r(c_6 + r(c_7 + r c_8)))))))$
    if $(y < 0)$ **then**
        $x = -x$
    end if
end if
return x

To make it even better, one more step can be applied to the result by recursion (7.13) starting from the resulting $x = \Phi^{-1}(u)$. Glasserman [48] states that the error does not exceed 10^{-15}.

7.3 Generating random vectors $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

It is said that a random vector $\mathbf{X} \in \mathbb{R}^d$ has a multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with the mean vector $\boldsymbol{\mu}$ (we identify vectors with columns) and the covariance matrix $\boldsymbol{\Sigma}$ if any combination $\sum_{j=1}^{d} a_j X_j$ has a univariate normal distribution, $\mathcal{N}(\boldsymbol{a}^T\boldsymbol{\mu}, \boldsymbol{a}^T\boldsymbol{\Sigma}\boldsymbol{a})$, where $\boldsymbol{a}^T = (a_1, \ldots, a_d)$. An immediate consequence of this definition is as follows. Suppose that $\mathbf{B}$ is $d \times d$-matrix and let

$\mathbf{Y} = \mathbf{BX}$. Then $\mathbf{Y}$ is multivariate normal because $\boldsymbol{a}\mathbf{Y} = \boldsymbol{a}^T\mathbf{BX} = (\boldsymbol{a}^T\mathbf{B})\mathbf{X}$ is multivariate normal. When we are talking about a d-dimensional random vector $\mathbf{X}$, then $\boldsymbol{\mu}$ is a d-dimensional vector and $\boldsymbol{\Sigma}$ an $d \times d$ matrix. Recall that a covariance matrix is symmetric and non-negative definite.

Example 7.6 For $n = 2$, the covariance matrix can be represented as follows:

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix},$$

where $\rho = \mathrm{Corr}(X_1, X_2)$, $\sigma_i^2 = \mathrm{Var}X_i$ $(i = 1, 2)$. Then assuming non-singularity of $\boldsymbol{\Sigma}$

$$\boldsymbol{\Sigma}^{-1} = \mathbf{C} = \frac{1}{1-\rho^2} \begin{pmatrix} \frac{1}{\sigma_1^2} & -\frac{\rho}{\sigma_1\sigma_2} \\ -\frac{\rho}{\sigma_1\sigma_2} & \frac{1}{\sigma_2^2} \end{pmatrix}.$$

To generate such a two-dimensional normal vector $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, substitute

$$X_1 = \sigma_1(\sqrt{1-|\rho|}Z_1 + \sqrt{|\rho|}Z_3), \quad X_2 = \sigma_2(\sqrt{1-|\rho|}Z_2 + \mathrm{sign}(\rho)\sqrt{|\rho|}Z_3),$$

where Z_1, Z_2, Z_3 are independent random variables with the common distribution $\mathcal{N}(0, 1)$. The fact that $(X_1, X_2)^T$ has a two-dimensional normal distribution is evident. One has to demonstrate that the covariance matrix of this random vector is correct. □

For a general d, unless the covariance matrix has a special form allowing to find of an ad hoc method, the following procedure is recommended for generating a random vector $\mathbf{X} = (X_1, \ldots, X_d)^T \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Each covariance matrix $\boldsymbol{\Sigma}$ is symmetric non-negative-definite, and in the sequel we assume it is non-singular (i.e. positive-definite). Then it can be factorized as follows

$$\boldsymbol{\Sigma} = \mathbf{A}\mathbf{A}^T \tag{7.14}$$

for some matrix $\mathbf{A}$.

Remark 7.7 Factorization (7.14) is not unique, unless we require that $\mathbf{A}$ is again positive-definite. In this case we denote the resulted matrix by $\boldsymbol{\Sigma}^{1/2}$. It can be computed by the so called diagonalization method, which will be recalled in Section 1.2.1 .

Proposition 7.8 *If* $\mathbf{Z} = (Z_1, \ldots, Z_d)^T$ *are independent random variables with the common distribution* $\mathcal{N}(0,1)$*, then for* $\mathbf{A}$ *from (7.14) we have that*

$$\mathbf{X} = \mathbf{AZ} + \boldsymbol{\mu} \tag{7.15}$$

has distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$*.*

Proof Random vector $\mathbf{AZ} + \boldsymbol{\mu}$ has jointly multivariate normal distribution because for a vector $\boldsymbol{a}$, random vector is normally distributed. Thus it suffices to find its mean vector and the covariance matrix. Thus

$$\mathbb{E}\mathbf{AZ} + \boldsymbol{\mu} = \mathbf{A}\mathbb{E}\mathbf{Z} + \boldsymbol{\mu} = \boldsymbol{\mu},$$

and, for the covariance matrix, we can omit $\boldsymbol{\mu}$

$$\mathbb{E}(\mathbf{AZ})(\mathbf{AZ})^T = \mathbb{E}\mathbf{AZZ}^T\mathbf{A}^T = \mathbf{A}\,\mathbf{Id}\,\mathbf{A}^T = \boldsymbol{\Sigma}.$$

□

To compute a 'root' $\mathbf{A}$ from the covariance matrix $\boldsymbol{\Sigma}$ one can use procedures often available in programming packages. They are frequently based on the so-called Cholesky decomposition, resulted in a lower triangular matrix. For example, one can check that $\boldsymbol{\Sigma} = \mathbf{AA}^T$ for

$$\boldsymbol{\Sigma} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 4 & 1 \\ 1 & 1 & 8 \end{pmatrix}, \quad \mathbf{A} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & \sqrt{3} & 0 \\ 1 & 0 & \sqrt{7} \end{pmatrix}.$$

Note that in Matlab or Python, the numerical $\mathbf{A}$ would be obtained, i.e., $\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1.7321 & 0 \\ 1 & 0 & 2.6458 \end{pmatrix}$. Cholesky decomposition is not a unique way to present $\boldsymbol{\Sigma} = \mathbf{AA}^T$.

In practice, we will assume that $\boldsymbol{\Sigma}$ is a positive-definite one, thus non-singular. Then the distribution of $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ has p.d.f.

$$f(\mathbf{x}) = f(x_1, \ldots, x_d) = \frac{1}{\sqrt{(2\pi)^d \det(\boldsymbol{\Sigma}}} e^{-\frac{\sum_{j,k=1}^d c_{ij} x_j x_k}{2}}, \tag{7.16}$$

where

$$\mathbf{C} = (c_{jk})_{j,k=1}^d = \boldsymbol{\Sigma}^{-1}.$$

In case of i.i.d. standard normal vector $\mathbf{Z} = (Z_1, \ldots, Z_d)$, its p.d.f. is

$$\frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{z_1^2 + \ldots + z_d^2}{2}\right) = \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{\mathbf{z}^T\mathbf{z}}{2}\right).$$

Let $\mathbf{\Sigma} = \mathbf{A}\mathbf{A}^T$. From formula (7.15), $\mathbf{X} = \mathbf{A}\mathbf{Z}$ has multi-normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. The Jacobian of mapping $\mathbf{z} = \mathbf{A}^{-1}\mathbf{x}$ is $\det \mathbf{A}$. We now use Proposition I.2.1 to obtain (7.16).

8 More on uniforms; d-ball, d-sphere, d-ellipsoid.

In this section, we explore the generation of random points in various mathematical objects such as a d-ball, $d-1$-sphere, and within d-ellipsoid, etc, without delving into acceptance rejection methods.

Consider geometrical objects in $\mathbb{R}^d$. For d-dimensional objects, their measurement involves the volume, denoted by $\mathrm{Vol}\,(\cdot)$. Let $B \subset \mathbb{R}^d$ be a subset with $0 < \mathrm{Vol}\,(B) < \infty$. Following the concepts introduced in Section 8, a random point $X \in B$ is said to be uniformly distributed on B, denoted as $\mathcal{U}[B)$, if its p.d.f. is

$$f(x) = \frac{1}{\mathrm{Vol}(B)}\mathbf{1}(x \in B),$$

implying $\mathbb{P}(X \in D) = \mathrm{Vol}\,(D)/\mathrm{Vol}\,(B)$ for all $D \subset B$. However, challenges arise when dealing with objects in $\mathbb{R}^d$ of dimension less than d, particularly those of dimension $d-1$.

One problem of interest is generating a uniformly distributed point in the d-ball defined as:

$$B_d = \left\{\mathbf{x} = (x_1, \ldots, x_d) : \sum_{j=1}^{d} x_j^2 \leq 1\right\}$$

with volume

$$\mathrm{Vol}(B_d) = \frac{\pi^{d/2}}{\Gamma(d/2+1)},$$

and on the $d-1$-sphere defined as

$$S_{d-1} = \left\{\mathbf{x} = (x_1, \ldots, x_d) : \sum_{j=1}^{d} x_j^2 = 1\right\},$$

with surface area

$$\text{Area}(S_{d-1}) = \frac{d\pi^{d/2}}{\Gamma(d/2+1)} = \frac{2\pi^{d/2}}{\Gamma(d/2)}.$$

It is said that $\sigma_{d-1}(d\mathbf{x})$ is the uniform distribution on the sphere S_{d-1}, if it has p.d.f. $f(\mathbf{x}) = \frac{\Gamma(d/2)}{2\pi^{d/2}}$ for $\mathbf{x} \in S_{d-1}$.

Change of variables formula; general theory. Here, we present an extension of the well known change of variable formula (I.2.1), previously utilized in this chapter. We consider subsets $C \subset \mathbb{R}^d$ with dimensions $k = d-1$ or $k = d$. For $k = d-1$, these subsets are surfaces, such as $d-1$-spheres or hiper-spheres or for $d = 2$, a curve) or d-balls for $k = d$. Parametric representations are employed:

$$\mathbf{x}(\boldsymbol{t}) = (x_1(t_1, \ldots, t_k), \ldots, x_d(t_1, \ldots, t_k)), \qquad \boldsymbol{t} = (t_1, \ldots, t_k) \in \Delta,$$

where $\Delta \subset \tilde{\Delta}$ and $\mathbf{x}(\boldsymbol{t})$ is smooth on $\tilde{\Delta}$. When $d-1$, we deal with surface areas denoted by Area$(\cdot)$. Integrals are expressed as $\int_C f\, dV$ for $k = d$ and surface (line) integrals $\int_C f\, dA$ for $k = d-1$.
Let

$$\boldsymbol{J}(\boldsymbol{t}) = \left(\frac{\partial x_j(\boldsymbol{t})}{\partial t_i}\right)_{i=1,\ldots,d, j=1,\ldots,k}$$

be the matrix of derivatives (of size $k \times d$) and set

$$\sqrt{g(\boldsymbol{t})} = \sqrt{|\det(\boldsymbol{J}(\boldsymbol{t})\boldsymbol{J}^T(\boldsymbol{t}))|}.$$

If $k = d$, then it is a classical Jacobian. The key formula here, known as the change of variable formula, for a continuous function f on C for $k = d-1$ is:

$$\int_C f\, dA = \int_\Delta f(\mathbf{x}(\boldsymbol{t}))\sqrt{g(\boldsymbol{t})}\, d\boldsymbol{t}. \tag{8.17}$$

For case $k = d$, the formula is:

$$\int_C f\, dV = \int_\Delta f(\mathbf{x}(\boldsymbol{t}))\sqrt{g(\boldsymbol{t})}\, d\boldsymbol{t}. \tag{8.18}$$

Case $d = 2$. For $d = 2$, dealing with the length of curves ($k = d - 1$), the integral is referred to as a line integral. Suppose a curve or arc C is given in a parametric form $(x(t), y(t)) \in \mathbb{R}^2$ for $t \in [t_1, t_2]$. In this case, the area is the length of such a curve/arc, given by

$$\text{Area}(C) = \int_C 1\, dA = \int_{t_1}^{t_2} \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2}\, dt.$$

Example 8.1 Consider a unit circle $S_1 = \{(x_1, x_2) : x_1^2 + x_2^2 = 1\}$, parametrized by $x_1 = \cos\theta, x_2 = \sin\theta$, where $\theta \in [0, 2\pi)$. The matrix of derivatives is $\boldsymbol{J} = (-\sin\theta, \cos\theta)$, and

$$\boldsymbol{J}^T(\theta) = \begin{pmatrix} -\sin\theta \\ \cos\theta \end{pmatrix}.$$

Then $dA = \sqrt{g(\theta)}\, d\theta$, so

$$dA = \sqrt{(-\sin\theta)^2 + (\cos\theta)^2}\, d\theta = d\theta.$$

The full arc length that of the circle is 2π. Denote by $\sigma_1(\cdot)$ the uniform distribution on S_1, with p.d.f.

$$f(x_1, x_2) = \begin{cases} \frac{1}{2\pi}, & (x_1, x_2) \in S_1 \\ 0 & \text{otherwise} \end{cases}$$

or $d\sigma_1(s) = \frac{1}{2\pi}\, d\theta$. This identification of a segment C on S_1 by a central angle θ complements the material studied earlier in Section 7. □

Example 8.2 Consider a line segment $[A_0, A_1]$ in $\mathbb{R}^2$, where $A_0 = (x_0, y_0)$ and $A_1 = (x_1, y_1)$. Without loss of generality, set $x_0 = 0, y_0 = 0$. Let us parametrize this line segment by $y = \frac{y_1}{x_1} t$, where $t \in [0, x_1]$, i.e. $\mathbf{x}(t) = (t, \frac{y_1}{x_1} t)$. Then the matrix of derivatives is $\boldsymbol{J} = (1, \frac{y_1}{x_1})$ and hence

$$\sqrt{g(t)} = \sqrt{\boldsymbol{J}(t)\boldsymbol{J}^T(t)} = \sqrt{1 + \left(\frac{y_1}{x_1}\right)^2}.$$

Then the length between points A_0 and $\frac{y_1}{x_1} x$ is

$$L(x) = \int_0^x \sqrt{1 + \left(\frac{y_1}{x_1}\right)^2}\, dt = \frac{x}{x_1} l_1, \quad x \le x_1,$$

where $l_1 = \sqrt{(x_1)^2 + (y_1)^2}$, that is the length of the segment. Let us see what is a random point $\mathbf{X}$ on the line segment $[A_0, A_1]$. It is (Ux_1, Uy_1), where $U \sim \mathcal{U}[0,1)$. We now introduce a natural parametrization $t = (x/x_1)l_1$ on the line segment $[A_0, A_1]$, which is the distance of t from A_0. In this parametrisation, the random point is Ul_1, which has p.d.f. $f(t) = 1/l_1$ for $t \in [A_0, A_1]$. Consider c.d.f. $F(t) = \frac{L(t)}{l_1}$ for $t \in [0, l_1]$ and a random number U uniformly distributed $\mathcal{U}[[0,1)$. Then $F^{-1}(U)$ is the sought-for random point, i.e. the solution of $U = x/l_1$ or $x = Ul_1$. □

Example 8.3 Let us continue with Example 8.2, considering C to be a piecewise broken line $[A_0, A_1], [A_1, A_2], \ldots, [A_{n-1}, A_n]$, where $A_i = (x_i, y_i)$. We denote it by C. The length of the i-th segment is l_i. Let t be the distance of a point along the line starting from the point A_0. Depending on $t \in [l_1 + \ldots + l_{i-1}, l_1 + \ldots + l_{i-1} + l_i]$ it lies on the line segment $[A_{i-1}, A_i]$. Then a composition method can express the random point $\mathbf{X}$ on C. Choose the i-th segment with probability $l_i/(l_1 + \ldots + l_n)$ and place $\mathbf{X}$ uniformly on this segment. Therefore it has p.d.f.

$$f(t) = \sum_{i=1}^{n} \frac{l_i}{l_1 + \ldots + l_n} \frac{1}{l_i} = \frac{1}{l_1 + \ldots + l_n}, \quad t \in C.$$

□

8.1 Generating random point in B_3 and on S_2.

Let us pause for a moment at the case $d = 3$. For a ball B_3, its volume is $4\pi/3$. In this case *spherical coordinates* are given by

$$\begin{aligned} x &= r\cos\theta\sin\phi, \\ y &= r\sin\theta\sin\phi, \\ z &= r\cos\phi, \end{aligned} \tag{8.19}$$

where $r > 0, \theta \in [0, 2\pi)$ and $\phi \in [0, \pi)$. The Jacobian is $r^2 \sin\phi$, so the volume element is $dV = r^2\, dr \sin\phi\, d\phi\, d\theta$. On the sphere S_2, we have spherical coordinates

$$\begin{aligned} x &= \cos\theta\sin\phi, \\ y &= \sin\theta\sin\phi, \\ z &= \cos\phi. \end{aligned} \tag{8.20}$$

Here $\theta \in [0, 2\pi)$ and $\phi \in [0, \pi)$. In this case the area element is $dA = \sin\phi \, d\phi \, d\theta$.

Denote mapping (8.20) from spherical to Cartesian coordinates by $\boldsymbol{h}$, and $\boldsymbol{g} = \boldsymbol{h}^{-1}$. Let $f(\mathbf{x}) = \frac{3}{4\pi}\mathbf{1}(\mathbf{x} \in B_3)$ be the p.d.f. of the uniform distribution $\mathcal{U}[B_3)$. Using the change of variables formula (I.2.1), we immediately obtain the following result. In this proposition $\mathbf{X} \in B_3$ is a random point in ball B_3 and (D, Φ, Θ) is the point in spherical coordinates, i.e.

$$(D\cos\Theta\sin\Phi, D\sin\Theta\sin\Phi, D\cos\Theta).$$

Proposition 8.4 *Let $C \subset B_3$ and $f(\mathbf{x})$ be the p.d.f. of the uniform distribution $\mathcal{U}[B_3)$. Then*

$$\int_C f(\mathbf{x})\, d\mathbf{x} = \int \frac{3}{4\pi}\, d\mathbf{x} = \int_{\boldsymbol{g}C} 3r^2\, dr\, \frac{1}{2}\sin\phi\, d\phi\, \frac{1}{2\pi}\, d\theta. \tag{8.21}$$

As a result, (D, Φ, Θ) are independent random variables, $D \stackrel{\mathcal{D}}{=} U^{1/3}$, Φ has a p.d.f. $\frac{1}{2}\sin\phi$ and Θ is uniformly distributed on $[0, 2\pi)$. We have $\Phi \stackrel{\mathcal{D}}{=} \arccos(1-2U)$, where $U \sim \mathcal{U}[0, 1)$

Proof Formula (8.21) follows from (I.2.1). Hence, we see that D has a p.d.f. $3r^2\mathbf{1}(r \in (0, 1])$. On the other side $\mathbb{P}(U^{1/3} \le x) = x^3$, for $x \in (0, 1]$, which has a p.d.f $3r^2$. The c.d.f F of Φ is

$$F(x) = \frac{1}{2}\int_0^x \sin\phi\, d\phi = \frac{1}{2}(1 - \cos x),$$

so $F^{-1}(y) = \arccos(1-2y)$. Hence $\Phi \stackrel{\mathcal{D}}{=} \arccos(1-2U)$, where $U \sim \mathcal{U}[0, 1)$. □

For the sphere S_2 the surface area is $2\pi^{3/2}/(\pi^{1/2}/2)$ because $\Gamma(3/2) = \sqrt{\pi}/2$. Thus, the uniform distribution $\sigma_2(\cdot)$ has a p.d.f. $f(\mathbf{x}) = \frac{1}{4\pi}$, $\mathbf{x} \in S_2$. In this case *spherical coordinates* are

$$x = \cos\theta\sin\phi, \qquad y = \sin\theta\sin\phi, \tag{8.22}$$

where $\theta \in [0, 2\pi)$ and $\phi \in [0, \pi)$. Notice that for a point $(x, y, z) \in S_2$

$$z^2 = 1 - x^2 - y^2 = 1 - (\cos\theta\sin\phi)^2 - (\sin\theta\sin\phi)^2 = (\cos\phi)^2.$$

Hence, $z = \cos\phi$. Denote the mapping (8.22) from spherical to Cartesian coordinates by $\boldsymbol{h}$. The matrix of derivatives is

$$\boldsymbol{J}(\phi,\theta) = \begin{pmatrix} \cos\phi\cos\theta & \cos\phi\cos\theta & -\sin\phi \\ -\sin\phi\sin\theta & \sin\phi\cos\theta & 0 \end{pmatrix}$$

and since

$$\det \boldsymbol{J}(\phi,\theta)\boldsymbol{J}^T(\phi,\theta) = \det \begin{pmatrix} 1 & 0 \\ 0 & \sin^2\phi \end{pmatrix}$$

we have $\sqrt{|\det \boldsymbol{J}(\phi,\theta)\boldsymbol{J}^T(\phi,\theta)|} = \sin\phi$. Hence, the surface element is $dA = \sin\phi\, d\,\phi\, d\theta$.

The uniform distribution σ_2 on the sphere S_2, by the change of variable formula (8.17) in spherical coordinates has the p.d.f.

$$f(\theta,\phi)\, d\theta\, d\phi = \frac{1}{4\pi}\, d\mathbf{x} = \frac{1}{4\pi}\sin\phi\, d\theta\, d\phi = \frac{1}{2\pi}\, d\theta\, \frac{1}{2}\sin\phi\, d\phi, \tag{8.23}$$

where $\theta \in [0, 2\pi)$ and $\phi \in [0, \pi)$. Let (Φ, Θ) be a random point on S_2 in spherical coordinates. From (8.23), we see that Θ and Φ are independent the marginal p.d.f. of Θ is $1/(2\pi)$, and Φ is $\frac{1}{2}\sin\phi$.

Remark 8.5 An old question was how to simulate Φ. Note that a naïve (but wrong) idea would be to simulate $\Phi \sim \mathcal{U}[0, \pi)$.

The following fact serves as the foundation for the so-called Marsaglia algorithm, a technique for generating random points on the sphere S_2.

Proposition 8.6 *Suppose that Z_1, Z_2, Z_3 are i.i.d. standard normal $\mathcal{N}(0,1)$ random variables and let*

$$X_1 = \frac{Z_1}{\sqrt{Z_1^2+Z_2^2+Z_3^2}}, X_2 = \frac{Z_1}{\sqrt{Z_1^2+Z_2^2+Z_3^2}}, X_3 = \frac{Z_3}{\sqrt{Z_1^2+Z_2^2+Z_3^2}}.$$

Then (X_1, X_2, X_3) is a random point on the sphere S_2.

Proof Consider the mapping $\boldsymbol{g}_1$ that transforms a point $\mathbf{z} \in \mathbb{R}^3 - \{\mathbf{0}\}$ from Euclidean to spherical coordinates:

$$\mathbb{R}^3 - \{\mathbf{0}\} \ni \mathbf{z} \to \boldsymbol{y} = (r, \theta, \phi) \in (0,\infty) \times [0, 2\pi) \times [0, \pi)\,, \tag{8.24}$$

where

$$\mathbf{z} = (r\cos\theta\sin\phi, r\sin\theta\sin\phi, r\cos\phi).$$

The Jacobian of $\boldsymbol{h}_1 = \boldsymbol{g}_1^{-1}$ is $J_{\boldsymbol{h}_1}(\boldsymbol{y}) = r^2\sin\phi$. The joint p.d.f. of (Z_1, Z_2, Z_3) is

$$\frac{1}{(2\pi)^{3/2}}\exp\left(\frac{z_1^2+z_2^2+z_3^2}{2}\right).$$

Applying the change of variables formula (I.2.1), we get

$$\frac{1}{(2\pi)^{3/2}}\exp\left(-\frac{z_1^2+z_2^2+z_3^2}{2}\right)\,dz_1dz_2dz_3 = \frac{1}{(2\pi)^{3/2}}\exp\left(-\frac{r^2}{2}\right)r^2\,dr\,\sin\phi\,d\phi\,d\theta.$$

Let C be defined as

$$C = \{(z_1,z_2,z_3)\in\mathbb{R}^3 : \left(\frac{z_1}{\sqrt{z_1^2+z_2^2+z_3^2}}, \frac{z_2}{\sqrt{z_1^2+z_2^2+z_3^2}}, \frac{z_3}{\sqrt{z_1^2+z_2^2+z_3^2}}\right)\in E\},$$

where E is a subset of S_2. In the spherical coordinates, we have

$$z_1 = r\cos\theta\sin\phi, \quad z_2 = r\sin\theta\sin\phi, \quad z_3 = r\cos\phi,$$

and the variables

$$x_1 = \frac{z_1}{\sqrt{z_1^2+z_2^2+z_3^2}} = \cos\theta\sin\phi, \quad x_2 = \frac{z_2}{\sqrt{z_1^2+z_2^2+z_3^2}} = \sin\theta\sin\phi,$$

and

$$x_3 = \frac{z_3}{\sqrt{z_1^2+z_2^2+z_3^2}} = \cos\phi$$

are independent of r. Using the change of variables formula (I.2.1), we then have

$$\begin{aligned}
&\int_E \frac{1}{(2\pi)^{3/2}}\exp\left(-\frac{x_1^2+x_2^2+x_3^2}{2}\right)\,dA \\
&= \int_C \frac{1}{(2\pi)^{3/2}}\exp\left(-\frac{r^2}{2}\right)r^2\,dr\sin\phi\,d\phi\,d\theta \\
&= \int_0^\infty r^2\exp\left(-\frac{r^2}{2}\right)\,dr\int_{C'}\frac{1}{(2\pi)^{3/2}}\sin\phi\,d\phi\,d\theta,
\end{aligned}$$

where $C = (0, \infty) \times C'$ and $C' \subset C$. Let $\boldsymbol{g}_2 : S_2 \to [0, 2\pi) \times [0, \pi)$ map from Euclidean to spherical coordinates. Using the fact that

$$\int_0^\infty r^2 e^{-r^2/2}\, dr = \sqrt{\frac{\pi}{2}},$$

we get

$$\begin{aligned} \int_C \frac{1}{(2\pi)^{3/2}} \exp\left(-\frac{r^2}{2}\right) r^2\, dr \sin\phi\, d\phi\, d\theta &= \frac{1}{4\pi} \int_{\boldsymbol{g}_2(E)} \sin\phi\, dr d\theta d\phi \\ &= \frac{1}{4\pi} \int_E dA, \end{aligned}$$

which completes the proof because $\frac{1}{4\pi} dA$ is the p.d.f. of σ_2. □

Remark 8.7 Suppose that $\mathbf{X}$ is a random point on S_2 and let $\mathbf{P}$ be an orthonormal matrix ($\mathbf{P}^{-1} = \mathbf{P}^T$). Then for $\mathbf{Z} = (Z_1, Z_2, Z_3)^T$, where Z_1, Z_2, Z_3 are i.i.d. standard normal variables, we have $\mathbf{PZ} \stackrel{\mathcal{D}}{=} \mathbf{Z}$. Hence a random point $\mathbf{X}$ on S_2 possesses the property $\mathbf{X} \stackrel{\mathcal{D}}{=} \mathbf{PX}$.

8.2 Spherical coordinates

In general, a point $\mathbf{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d \setminus \{\mathbf{0}\}$ in spherical coordinates is expressed by

$$\begin{aligned} x_1 &= r\cos\phi_1 \\ x_2 &= r\sin\phi_1\cos\phi_2 \\ x_3 &= r\sin\phi_1\sin\phi_2\cos\phi_3 \\ &\vdots \\ x_{d-1} &= r\sin\phi_1 \ldots \sin\phi_{d-2}\cos\theta \\ x_d &= r\sin\phi_1 \ldots \sin\phi_{d-2}\sin\theta, \end{aligned}$$

where $r > 0$, $\phi_1, \ldots, \phi_{d-2} \in [0, \pi), \theta \in [0, 2\pi)$. The Jacobian is

$$J(r, \phi_1, \ldots, \phi_{d-1}, \theta) = r^{d-1} \sin^{d-2}\phi_1 \sin^{d-3}\phi_2 \ldots \sin\phi_{d-2}.$$

Then the volume element is

$$dV = r^{d-1}\, dr\ \sin^{d-2}\phi_1\, d\phi_1\ \sin^{d-3}\phi_2\, d\phi_2 \ldots \sin\phi_{d-2}\, d\phi d - 2\, d\theta.$$

Similarly the surface area on S_{d-1} is

$$dA = \sin^{d-2}\phi_1\, d\phi_1\, \sin^{d-3}\phi_2\, d\phi_2 \ldots \sin\phi_{d-2}\, d\phi_{d-2}\, d\theta.$$

It is computed from $\sqrt{|\det(\boldsymbol{J}(\boldsymbol{t})\boldsymbol{J}^T(\boldsymbol{t}))|}$, where $\boldsymbol{J}(\phi,\theta) = \left(\frac{\partial \mathbf{x}}{\partial \boldsymbol{t}}\right)$ is $(d-1)\times d$ the matrix of derivatives. In particular, for $d=3$

$$\boldsymbol{J}(\phi,\theta) = \begin{pmatrix} \cos\phi\cos\theta & \cos\phi\sin\theta & -\sin\phi \\ -\sin\phi\sin\theta & \sin\phi\cos\theta & 0 \end{pmatrix}$$

and therefore $|\det(\boldsymbol{J}(\phi,,\theta)\boldsymbol{J}^T(\phi,\theta))| = \sin\phi$.

We need the following counterpart of Lemma 7.1 for general dimension. For this, we prove the following lemma.

Lemma 8.8 *For the ball $B_d \subset \mathbb{R}^d$*

$$\begin{aligned} \mathrm{Vol}(B_d) &= \frac{1}{d}\left(\int_0^\pi \sin^{d-2}\phi_1\, d\phi_1\right)\ldots\left(\int_0^\pi \sin\phi_{d-1}\, d\phi_{d-2}\right)\left(\int_0^{2\pi} d\phi_{d-1}\right) \\ &= \frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)} \\ \mathrm{Area}(S_{d-1}) &= \left(\int_0^\pi \sin^{d-2}\phi_1\, d\phi_1\right)\ldots\left(\int_0^\pi \sin\phi_{d-1}\, d\phi_{d-2}\right)\left(\int_0^{2\pi} d\phi_{d-1}\right) \\ &= \frac{d\pi^{d/2}}{\Gamma(\frac{d}{2}+1)}. \end{aligned}$$

Corollary 8.9

$$\frac{1}{\mathrm{Vol}\,(B_d)} = dc_1\cdots c_{d-2}\frac{1}{2\pi},$$

where $c_j = 1/(I_j)$ and $I_j = \int_0^\pi \sin^j x\, dx$.

Proof It is known that $I_1 = 2$ and for $n \geq 2$

$$\begin{aligned} I_n = \int_0^\pi \sin^n x\, dx &= \begin{cases} 2\frac{n-1}{n}\frac{n-3}{n-2}\cdots\frac{2}{3} & n \text{ odd} \\ \pi\frac{n-1}{n}\frac{n-3}{n-2}\cdots\frac{1}{2} & n \text{ even.} \end{cases} \\ &= \begin{cases} 2\frac{(n-1)!!}{n!!} & n \text{ odd} \\ \pi\frac{(n-1)!!}{n!!} & n \text{ even} \end{cases} \end{aligned}$$

Suppose that $d \geq 2$ is even. Then for $d = 2k$

$$\mathrm{Vol}(B_d) = \frac{1}{d}\left[\pi\frac{(d-3)!!}{(d-2)!!}\right]\left[2\frac{(d-4)!!}{(d-3)!!}\right]\ldots[2]\,[2\pi] = \frac{1}{d}\pi^{k-1}\frac{1}{(d-2)!!}2^{k-1}(2\pi) = \frac{2^k\pi^k}{d!!}.$$

Since $d = 2k$ we have $(2k-2)!! = 2^{k-1}(k-1)!$. Hence $\mathrm{Vol}(B_d) = \pi^{d/2}/k!$. Suppose now $d = 2k+1$.Then

$$\mathrm{Vol}(B_d) = \frac{1}{d}\left[2\frac{(d-3)!!}{(d-2)!!}\right]\left[\pi\frac{(d-4)!!}{(d-3)!!}\right]\ldots[2]\,[2\pi] = \frac{1}{d}\pi^{k}\frac{1}{(d-2)!!}2^{k+1}.$$

Hence

$$\mathrm{Vol}(B_d) = \frac{2(2\pi)^k}{d!!}.$$

The proof is completed using

$$(2k-1)!! = \pi^{-1/2}2^k\Gamma\left(k+\frac{1}{2}\right).$$

□

Lemma 8.10 *Let $(D, \Phi_1, \ldots, \Phi_{d-2}, \Theta)$ be spherical coordinates of point $(Z_1, \ldots, Z_d) \in \mathbb{R}^d$, where Z_i are i.i.d. standard normal random variables. Then $(D, \Phi_1, \ldots, \Phi_{d-2}, \Theta)$ are independent random variables, D^2 is chi-square distributed χ^2_d with d degrees of freedom, i.e.*

$$f_{\chi^2_d}(r) = \frac{1}{2^{d/2}\Gamma(d/2)}r^{d/2-1}e^{-r/2}, \qquad r > 0.$$

Furthermore Φ_i have p.d.f. $c_{d-i-1}\sin^{d-i-1}x$, $x \in [0,\pi)$, where $(c_i)^{-1} = \int_0^\pi \sin^i x\,dx$ and $\Theta \sim \mathcal{U}[0, 2\pi)$.

Proof From formula (I.2.1) we see that the joint p.d.f of point $(Z_1, \ldots, Z_d)$ in generalized spherical coordinates is

$$\begin{aligned}&\frac{1}{(2\pi)^{d/2}}\exp\left(-\frac{z_1^2 + \ldots + z^d}{2}\right)\\ &= \frac{1}{(2\pi)^{d/2}}e^{-r^2/2}r^{d-1}\sin^{d-2}\phi_1\ldots\sin\phi_{d-2}\,dr\,d\phi_1\ldots d\phi_{d-2}\,d\theta.\end{aligned}$$

We have

$$\int_0^\infty e^{-r^2/2} r^{d-1} = 2^{d/2-1}\Gamma(d/2),$$

and so

$$g_D(r) = \frac{1}{2^{d/2-1}\Gamma(d/2)} e^{-r^2} r^{d-1}$$

is a p.d.f. of D. Let $g_{\Phi_j}(\phi_j) = c_{d-j-1}\sin^{d-j-1}\phi_j$ where $c_j^{-1} = \int_0^\pi \sin^j \phi_j \, d\phi_j$ be the p.d.f. of Φ_j and $g_\Theta(\theta) = 1/(2\pi)$. Then

$$\begin{aligned}
&\frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{z_1^2 + \ldots + z^d}{2}\right) \\
&\quad = \left[\frac{1}{(2\pi)^{d/2}} 2^{d/2-1}\Gamma(d/2) c_1^{-1} \ldots c_{d-2}^{-1} 2\pi\right] g_D(r) g_{\Phi_1}(\phi_1) \ldots g_{\Phi_{d-2}}(\phi_{d-2})\, g_\Theta(\theta).
\end{aligned}$$

With the use of Lemma 8.8, we now see that the expression in brackets equals 1 because $\Gamma(d/2)/\Gamma(d/2+1) = 2/d$. To complete the proof, we have to note that

$$\frac{\mathbb{P}(D^2 \le x)}{dx} = \frac{d}{dx}\int_0^{\sqrt{x}} \frac{1}{2^{d/2-1}\Gamma(d/2)} e^{-r^2/2} r^{d-1}\, dr = \frac{1}{2^{d/2}\Gamma(d/2)} e^{-x/2} x^{d/2-1},$$

which is the p.d.f of the chi-square distribution with d degree of freedom. □

It means that $\mathbf{Z}$ can be represented by a random variable D and an independent random point on S^{d-1}, where D^2 is χ_d^2 distributed.

In Table 8.1 we show values of c_i^{-1} for $i = 2, \ldots, 8$.

i	2	3	4	5	6	7	8
c_i	$2/\pi$	$3/4$	$8/(3\pi)$	$15/16$	$16/(5\pi)$	$35/32$	$128/(35\pi)$

Table 8.1: Values of $c_i^{-1} = \int_0^\pi \sin^i \phi \, d\phi$ for $i = 2, \ldots, 8$.

A corresponding general result to Proposition 8.6 allows generating a random point on S_{d-1}. Its proof and a proof of the next proposition we leave to the reader.

Proposition 8.11 *Let $\mathbf{Z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{Id})$ and $\mathbf{X} = \mathbf{Z}/||\mathbf{Z}||$, where $||\mathbf{Z}||^2 = \sum_{j=1}^d Z_j^2$. Then $\mathbf{X}$ is a random point on S_{d-1}.*

8. MORE ON UNIFORMS; D-BALL, D-SPHERE, D-ELLIPSOID.

Knowing how to generate a random point of sphere S_{d-1}, one can ask how to get a random point inside ball B_d. The following is an extension of Lemma 7.1 for $d = 2$, which was demonstrated in Section 7.

Lemma 8.12 *Let* $\mathbf{X}$ *be a random point uniformly distributed within ball* B_d. *and let* $(D, \Phi_1, \ldots, \Phi_{d-2}, \Theta)$ *be its generalized spherical coordinates representation. Then* $D, \Phi_1, \ldots, \Phi_{d-2}, \Theta$ *are independent random variables,* D *has p.d.f.*

$$f(r) = dr^{d-1}, \qquad 0 < r \leq 1$$

and $(\Phi_1, \ldots, \Phi_{d-2}, \Theta)$ *is a random point on* $d-1$*-sphere* S_{d-1}.

Proof We have to consider a mapping $\boldsymbol{g} : \mathbb{R}^d \ni \mathbf{x} \to (r, \phi_1, \ldots, \phi_{d-1}, \theta)$ and then apply the change of variable formula (I.2.1).

Proposition 8.13 *Let* $\mathbf{X}$ *be a random point on* $d-1$*-sphere uniformly distributed* $\mathcal{U}[S_{d-1}]$ *(in Cartesians coordinates). Then* $\mathbf{Y} = U^{1/d}\mathbf{X} \in B_d$ *is uniformly distributed* $\mathcal{U}[B_d)$, *where* U *and* $\mathbf{X}$ *are independent.*

Proof We have

$$\frac{d}{2\pi c_1 \cdots c_{d-2}} = \mathrm{Vol}(B_d),$$

where $c_j^{-1} = \int_0^\pi sin^j x\, dx$. Let $B_d \ni \boldsymbol{y} \to (r, \phi_1, \ldots, \phi_{d-2}, \theta)$ be a mapping $\boldsymbol{g}$ from Cartesian to spherical coordinates in $\mathbb{R}^d$. Let $\boldsymbol{h} = \boldsymbol{g}^{-1}$. Then the Jacobian is

$$r^{d-1} \sin^{d-2}(\phi_1) \cdots \sin \phi_{d-2},$$

If $\mathbf{Y}$ in new coordinates $(D, \mathbf{X})$ is uniformly distributed in B_d, then p.d.f. is

$$\begin{aligned} &\frac{1}{\mathrm{Vol}(B_d)} r^{d-1} \sin^{d-2}(\phi_1) \cdots \sin \phi_{d-2} dr\, d\phi_1 \cdots \phi_{d-2} \\ &\quad = dr^{d-1} \frac{1}{c_{d-2}} \sin^{d-2} \cdots \frac{1}{c_1} \sin \phi_{d-2} \frac{1}{2\pi}. \end{aligned}$$

Since $U^{1/d}$ has p.d.f dr^{d-1} for $r \in (0, 1]$, the proof is completed.

8.3 Ellipsoids

Before we discuss how to generate a random point within a d-ellipsoid let us note the following fact.

Lemma 8.14 *Consider a random point* $\mathbf{X}$ *on a subset* $B \subset \mathbb{R}^d$ *uniformly distributed* $\mathcal{U}[B)$*, where* $0 < \mathrm{Vol}(B) < \infty$*, and let* $\boldsymbol{T}$ *be a* $d \times d$ *non-singular matrix. We consider* $\mathbf{X}$ *as a column vector. Then* $\boldsymbol{T}\mathbf{X}$ *is a random point in* $\boldsymbol{T}B$ *with uniform distribution having p.d.f.* $\frac{1}{|\det \boldsymbol{T}|\mathrm{Vol}(B)}\mathbf{1}(x \in \boldsymbol{T}B)$.

Proof Let $f(\mathbf{x}) = 1/\mathrm{Vol}(B)$ be a p.d.f. of $\mathbf{X}$ and $\boldsymbol{y} = \boldsymbol{T}\mathbf{x}$. Then $\mathbf{x} = T^{-1}\mathbf{x}$ and the Jacobian of this transformation is $\det \boldsymbol{T}^{-1} = 1/\det(\boldsymbol{T})$. Then from the change of variables formula (I.2.1), $\boldsymbol{T}\mathbf{X}$ has p.d.f. $1/(|\det(\boldsymbol{T})|\mathrm{Vol}(B)\mathbf{1}(\boldsymbol{y} \in \boldsymbol{T}B)$. □

Consider a hyper-ellipsoid $\mathcal{E}_d = \{\mathbf{x} \in \mathbb{R}^d : \mathbf{x}^T\mathbf{A}^{-1}\mathbf{x} \leq 1\}$, where $\mathbf{A}$ is a positive-definite matrix, which is diagonalisable, which means that for some orthogonal matrix $\mathbf{P}$ and vector $\boldsymbol{a}$ we have $\mathbf{A} = \mathbf{P}\boldsymbol{\Lambda}\mathbf{P}^T$, where $\boldsymbol{\Lambda} = \mathbf{diag}(\boldsymbol{a}^2)$ and $\boldsymbol{a}^2 = (a_1^2, \ldots, a_d^2)$. We aim to demonstrate that $\mathcal{E}_d = \mathbf{L}B_d$, where

$$\mathbf{L} = \mathbf{P}\mathbf{diag}(\boldsymbol{a}).$$

In other words, for $B_d \ni \mathbf{x} \to \boldsymbol{y} = \mathbf{L}\mathbf{x}$, we have $\mathbf{x} = \mathbf{L}^{-1}\boldsymbol{y}$. For $\mathbf{x} \in B_d$, we have

$$1 \geq \mathbf{x}^T\mathbf{x} = (\mathbf{L}^{-1}\boldsymbol{y})^T(\mathbf{L}^{-1}\boldsymbol{y}) = \boldsymbol{y}^T(\mathbf{L}\mathbf{L}^T)^{-1}\boldsymbol{y}$$

or, substituting $\mathbf{A} = \mathbf{L}\mathbf{L}^T$ we obtain $\boldsymbol{y}\mathbf{A}^{-1}\boldsymbol{y} \leq 1$. Hence, $\boldsymbol{y} = \mathbf{L}\mathbf{x}$ belongs to

$$\boldsymbol{y}^T\mathbf{A}^{-1}\boldsymbol{y} \leq 1.$$

In particular, setting $\mathbf{P} = \mathbf{Id}$, we get

$$\sum_i^d \frac{y_i^2}{a_i^2} \leq 1,$$

which represents a hyper-ellipsoid with principal axes parallel to coordinate axes. Any ellipsoid not centered at the origin can be obtained from this by rotation with general orthonormal $\mathbf{P}$ and a shift.

Therefore, utilizing Lemma 8.14, we present the following algorithm for generating random points within an ellipsoid $\mathcal{E}_d$.

Generating random points $\mathbf{Y}_1, \ldots, \mathbf{Y}_R \sim \mathcal{U}[\mathcal{E}_d)$ We provide the following algorithm for generating a random point within an ellipsoid $\mathbf{x}^T\mathbf{A}^{-1}\mathbf{x} = 1$, where $\mathbf{A}$ is a symmetric positive definite matrix. Let us diagonalize $\mathbf{A} = \mathbf{P}\mathbf{diag}(\boldsymbol{a})\mathbf{diag}(\boldsymbol{a})\mathbf{P}^T$, where $\mathbf{P}$ is an orthonormal matrix.

1: Generate R random points $\mathbf{X}_1, \ldots, \mathbf{X}_R \in B_d$.

2: Diagonalize $\mathbf{A} = \mathbf{P}\mathbf{diag}(\boldsymbol{a})\mathbf{diag}(\boldsymbol{a})\mathbf{P}^T$.

3: Set $\mathbf{L} = \mathbf{P}\mathbf{diag}(\boldsymbol{a})$

4: Output $\mathbf{Y}_i = \mathbf{L}\mathbf{X}_i$, $i = 1, \ldots, R$

Example 8.15 Consider a two-dimensional ellipsoid with principal axes parallel to coordinate axes, i.e., $(x/a)^2 + (y/b)^2 \leq 1$. This corresponds to a a diagonal matrix $\mathbf{A} = \begin{pmatrix} a^2 & 0 \\ 0 & b^2 \end{pmatrix}$. Since it is diagonal, it implies that $\mathbf{P} = \mathbf{I}$ and $\mathbf{L} = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}$. Thus, a point uniformly distributed within the ellipse can be obtained through the following steps:

1. Choose $U_1, U_2 \sim \mathcal{U}[0, 1)$
2. Set $R = \sqrt{U_2}$
3. Set $\mathbf{x} = (x, y)^T$ with $x = R\cos(2\pi U_1), y = R\sin(2\pi U_1)$.
4. **Return $\boldsymbol{y} = \mathbf{L}\mathbf{x}$.**

Exemplary $n = 500$ points $\mathbf{x}_i, i = 1, \ldots, n$ from the unit ball B_1 and corresponding $\boldsymbol{y}_i = \mathbf{L}\mathbf{x}_i, i = 1, \ldots, n$ with $a = 5$ and $b = 1$ are depicted in Fig. 8.5.

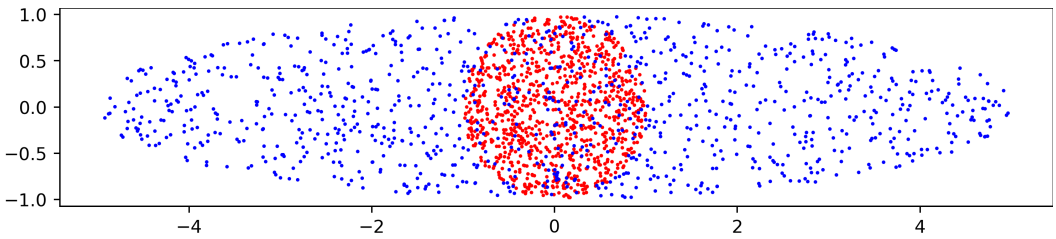


Figure 8.5: () points $\mathbf{x}_i, i = 1, \ldots, 500$ chosen uniformly on a ball B_1 and (blue) points transformed $\mathbf{x}_i' = \mathbf{L}\mathbf{x}_i, i = 1, \ldots, 500$.

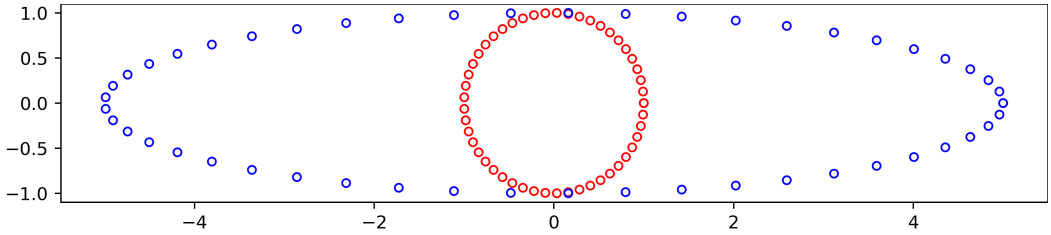
□

We now discuss how to generate a random point on an ellipse. [1] If we follow the idea of generating a random point within an ellipsoid, we obtain something wrong when generating a random point on an ellipsoid. To see it, let us do the following experiment. Consider a two-dimensional ellipse with principal axes parallel to coordinate axes, i.e., $(x/5)^2 + y^2 = 1$. This corresponds to a diagonal matrix $\mathbf{A} = \begin{pmatrix} 5^2 & 0 \\ 0 & 1 \end{pmatrix}$. Let a circle $S_1 = \{\mathbf{x}^T\mathbf{x} = 1\}$ be a circle and the mapping and $\mathbf{L} = \begin{pmatrix} 5 & 0 \\ 0 & 1 \end{pmatrix}$.

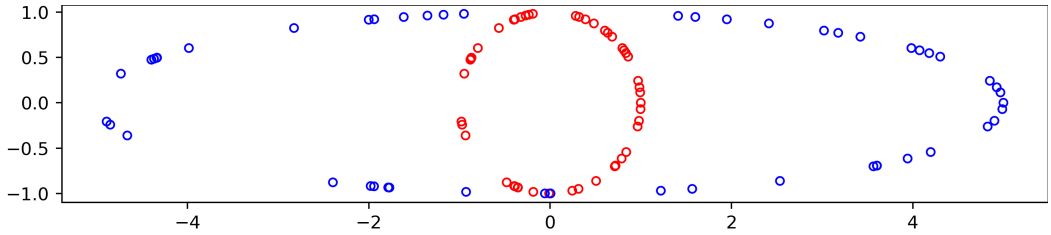
On Figure 8.6a), a regular grid on the circle S_1 (red circles) and its transformation to the ellipse $(x/5)^2+y^2 = 1$ (blue circles) reveal unequal distances between points. Let $\sigma_1(\cdot)$ be a uniform distribution on S_1.

Although $\mathbf{L}$ maps a random element $\mathbf{X} \in B_2$ to a random element $\mathbf{LX} \in \mathcal{E}_2$ within the ellipse, on Figure 8.6 b) we observe that after mapping $x := 5x$ and $y := y$ the circle transforms to an irregular grid on $(x/5)^2 + y^2 = 1$.

[1] To generate a point on an 2-ellipsoid, see https://math.stackexchange.com/questions/973101/how-to-generate-points-uniformly-distributed-on-the-surface-of-an-ellipsoid



a) Regular grid $\mathbf{x}_i, i = 1, \ldots, 50$ on a circle S_1 (red points) and points transformed $\boldsymbol{y}_i = \mathbf{L}\mathbf{x}_i, i = 1, \ldots, 50$



b) Points $\mathbf{x}_i, i = 1, \ldots, 50$ chosen uniformly from S_1 (red points) and points transformed $\boldsymbol{y}_i = \mathbf{L}\mathbf{x}_i, i = 1, \ldots, 50$

Figure 8.6: Points on a circle S_1, regular grid (top row) and chosen uniformly (bottom row) transformed to an ellipse $(x/5)^2 + y^2 = 1$.

In other words, on Figure 8.6, we display exemplary 50 random points $(5\cos\Theta, \sin\Theta)$, where Θ is uniformly distributed $\mathcal{U}[0, 2\pi)$. This experiment illustrates that a method from In the sequel, we show how to generate a random point on the ellipse, which requires numerical methods to be applied. Consider an ellipse

$$\left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 = 1,$$

its parametrisation in spherical coordinates is following

$$\begin{aligned} x &= a\cos\theta, \\ y &= b\sin\theta. \end{aligned}$$

In other words, $E_1 = \{(a\cos\theta, b\sin\theta) : \theta \in [0, 2\pi]\}$, and let C be an arc on this ellipse from point $A = (a, 0)$ to point $B = (a\cos\theta, b\sin\theta)$. Then, the

matrix of derivatives is $\boldsymbol{J}(\theta) = (-a\sin\theta, b\cos\theta)$ and

$$\begin{aligned}\sqrt{g(\theta)} &= \sqrt{|\boldsymbol{J}(\theta)\boldsymbol{J}(\theta)^T|} = \sqrt{a^2\sin^2\theta + b^2\cos^2\theta} = \sqrt{a^2\sin^2\theta + b^2(1-\sin^2\theta)} \\ &= \sqrt{b^2 - (b^2-a^2)\sin^2\theta} = b\sqrt{1 - \tfrac{b^2-a^2}{b^2}\sin^2\theta}.\end{aligned}$$

We denote by $L(\theta)$ the distance between point $A = (a, 0)$ and $B = (a\cos\theta, b\sin\theta)$. Then

$$\int_C 1\, dA = L(\theta) = b\int_0^\theta \sqrt{1 - \frac{b^2-a^2}{b^2}\sin^2 s}\, ds.$$

In the above formula, we can recognise the so-called incomplete elliptic integral of the second kind

$$E(\theta, m) = \int_0^\theta \sqrt{1 - m\sin^2 s}\, ds.$$

Hence, $L(\theta) = bE(\theta, m)$, where $m = \frac{b^2-a^2}{b^2}$. Note that an infinitesimal element of length is

$$dA = b\sqrt{1 - m\sin^2\theta}.$$

Thus, to draw a random point on E_1, we need to know the distribution of the angle Θ, which makes point $(a\cos\Theta, b\sin\Theta)$ random on E_1. Its c.d.f. is $F(\theta) = L(\theta)/L(2\pi) = E(\theta, m)/E(2\pi, m)$, so we have representation $\Theta \stackrel{\mathcal{D}}{=} F^{-1}(U)$, where $U \sim \mathcal{U}[0, 1)$. Unfortunately, we cannot express elliptic integrals in terms of elementary function, and therefore we have to approach computing the solution $u = L(x)$ by a numerical method.

Incomplete elliptic integral and its inverse in Python `SciPy` (which stands for *Scientific Python*) is a popular scientific library which contains the incomplete elliptic integral of the second kind $E(\theta, m)$ `scipy.special.ellipeinc`[2]. The library `pynverse`[3] is popular for finding numerically roots of $y = f(x)$ for quite a general class of bounded functions f. It minimises $(f(x) - y)^2$ using the Brent's method[4], which is a combination of bisection, secant and inverse quadratic interpolation methods. In Python listing III.1, an example

[2] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.special.ellipeinc.html`

[3] `https://pypi.org/project/pynverse/`

[4] `https://en.wikipedia.org/wiki/Brent%27s_method`

for sampling one point uniformly from an ellipse with paramters $a = 5, b = 1$ (then $m = 24$) is presented.

Python listing III.1: sampling a point from an ellipse uniformly at random

```
from pynverse import inversefunc
from scipy.special import ellipeinc
import numpy as np

m = 24
U = np.random.uniform()
F_fun = (lambda x: ellipeinc(x,m)/ellipeinc(2*np.pi,m))
theta = inversefunc(F_fun , y_values=U)

x=a*np.cos(theta)
y=b*np.sin(theta)
```

Example 8.16 We continue the example with an ellipse $(x/5)^2 + y^2 = 1$. We have $a = 5, b = 1$ thus $m = 24$. The c.d.f. $F(\theta)$ is depicted in Fig. 8.7.

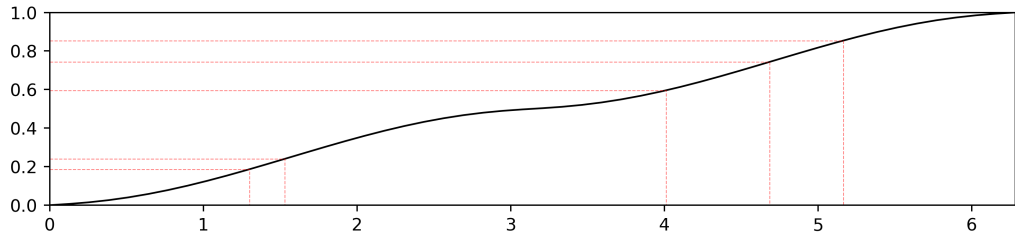


Figure 8.7: A c.d.f. F for an ellipse $(x/5)^2+y^2 = 1$ together with 5 uniformly chosen $U_1, \ldots, U_5$ and numerically computed $\theta_i = F^{-1}(U_i), i = 1, \ldots, 5$

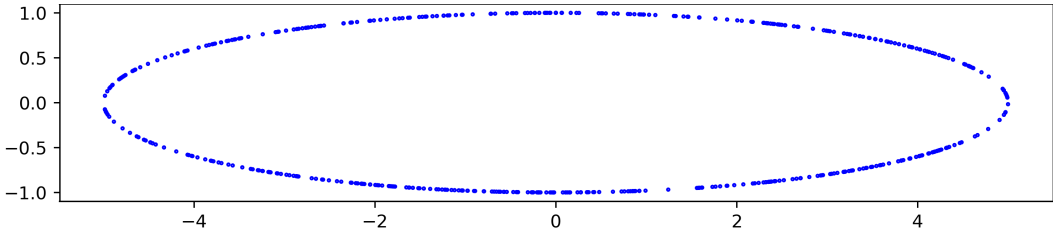


Figure 8.8: 500 points from ellipse $(x/5)^2 + y^2 = 1$.

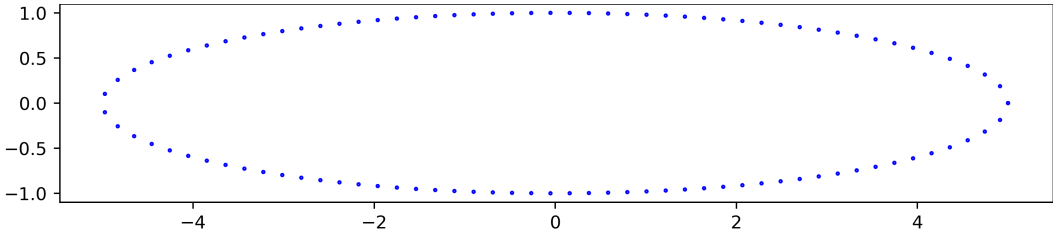


Figure 8.9: 500 equally distanced points from ellipse $(x/5)^2 + y^2 = 1$.

□

8.4 Numerically sampled random point on an ellipsoid.

Unfortunately, sampling a random point on a surface of dimension 2 in $\mathbb{R}^3$ has some limitations. For example, consider an ellipsoid E_2 given by

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1.$$

In this case, we can find that

$$\begin{aligned} g(\phi,\theta) &= a^2b^2\cos^4\theta\cos^2\phi\sin^2\phi + 2a^2b^2\cos^2\theta\cos^2\phi\sin^2\theta\sin^2\phi + \\ &\quad a^2b^2\cos^2\phi\sin^4\theta\sin^2\phi + b^2c^2\cos^2\theta\sin^4\phi + a^2c^2\sin^2\theta\sin^4\phi \\ &= \sin^2\phi\left[c^2(a^2-b^2)\cos^2\theta(\cos^2(\phi)-1) + a^2((b^2-c^2)\cos^2\phi + c^2)\right] \end{aligned} \tag{8.25}$$

This means that we should sample a random point $\mathbf{X}$ having p.d.f.

$$\sqrt{g(\phi,\theta)}/\int_0^{\pi}\int_0^{2\pi}\sqrt{g(\phi,\theta)}\,d\phi\,d\theta,$$

which seems quite troublesome; this is a two-dimensional density, and coordinates are not independent (unless special cases of a, b, c). For example, for $a = b = 1$

$$\sqrt{g(\phi,\theta)} = \sqrt{(1-c^2)\cos^2\phi + c^2}\sin\phi,$$

in which Φ and Θ are independent. For general values of a, b, c, we propose the following method to sample a point from a surface C given by

$x_1(\boldsymbol{t}), x_2(\boldsymbol{t}), x_3(\boldsymbol{t})$, where $\boldsymbol{t} \in [k_0, k_1] \times [l_0, l_1]$. In the case of C being an ellipsoid $x^2/a^2 + y^2/b^2 + z^2/c^2 = 1$, we have the following general spherical parametrisation

$$\begin{aligned} x(\boldsymbol{t}) &= a\cos\theta\sin\phi, \\ y(\boldsymbol{t}) &= b\sin\theta\sin\phi, \\ z(\boldsymbol{t}) &= c\cos\phi, \end{aligned} \tag{8.26}$$

where $\boldsymbol{t} = (\theta, \phi)$ with $\theta \in [0, 2\pi)$ and $\phi \in [0, \pi)$.

This numerical method is based on the proof of the formula (8.17) by approximating the surface by parallelograms. The rough idea is the following: we split the surface C into small pieces, approximate their areas, and then we sample one of the pieces proportionally to its area. Afterwards, we sample within the chosen part of the surface. In some sense, it is similar to sampling uniformly from a piecewise broken line in Example 8.3.

On a rectangle $[k_0, k_1] \times [l_0, l_1]$, we construct a regular grid by splitting $[k_0, k_1]$ into n_k intervals and $[l_0, l_1]$ into n_l intervals. Thus, the grid points are

$$\mathbf{p}_{i,j} = (k_0 + \kappa i, l_0 + \ell j), \quad i = 0, \ldots, n_k,\ j = 0, \ldots, n_l,$$

where

$$\kappa := \frac{(k_1 - k_0)}{n_k}, \qquad \ell = \frac{(l_1 - l_0)}{n_l}.$$

Thus, n_k and n_l are the precision parameters of the procedure.

Note that the rectangle

$$\mathtt{Rect}_{i,j} = [\mathbf{p}_{i,j}, \mathbf{p}_{i+1,j}, \mathbf{p}_{i,j+1}, \mathbf{p}_{i+1,j+1}]$$

is transformed into some area on the surface $C_{i,j} \subseteq C$. We use the convention that $n_k + 1 = 0$ and $n_l + 1 = 0$. The rectangles $\mathtt{Rect}_{i,j}$ are disjoint, and so are the surfaces $C_{i,j}$. The area of the piece of surface is

$$\text{Area}\,(C_{i,j}) = \int_{C_{i,j}} dA = \int_{\mathtt{Rect}_{i,j}} \sqrt{g(\boldsymbol{t})}\, d\boldsymbol{t} = \int_{k_0+\kappa i}^{k_0+\kappa(i+1)} \int_{l_0+\ell j}^{l_0+\ell(j+1)} \sqrt{g(k,l)}\; dl\; dk.$$

In case of $i = n_k$ we compute $\int_0^{\kappa} \ldots dl$, similarly in case of $j = n_l$. Now we approximate the area of $C_{i,j}$ by the area parallelogram $\mathtt{Paral}_{i,j}$ spanned on points $\mathbf{q}^1_{i,j}, \mathbf{q}^2_{i,j}, \mathbf{q}^3_{i,j}$, which are

$$\begin{aligned}
\mathbf{q}_{i,j}^{(1)} &= (x(k_0+\kappa i, l_0+\ell j),\ y(k_0+\kappa i, l_0+\ell j),\ z(k_0+\kappa i, l_0+\ell j)),\\
\mathbf{q}_{i,j}^{(2)} &= (x(k_0+\kappa(i+1), l_0+\ell j),\ y(k_0+\kappa(i+1), l_0+\ell j),\ z(k_0+\ell(i+1), l_0+\ell j)),\\
\mathbf{q}_{i,j}^{(3)} &= (x(k_0+\kappa i, l_0+\ell(j+1)),\ y(k_0+\kappa i, l_0+\ell(j+1)),\ z(k_0+\kappa i, l_0+\ell(j+1))),
\end{aligned}$$

see Fig. 8.10.

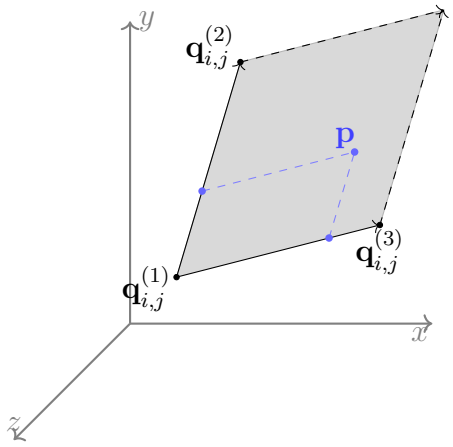


Figure 8.10: A parallelogram span over $\mathbf{q}_{i,j}^{(1)} = (1,1,1), \mathbf{q}_{i,j}^{(2)} = (3,5,4), \mathbf{q}_{i,j}^{(3)} = (6,4,7)$. Points chosen uniformly on $(\mathbf{q}_{i,j}^{(1)}, \mathbf{q}_{i,j}^{(2)})$ and on $(\mathbf{q}_{i,j}^{(1)}, \mathbf{q}_{i,j}^{(3)})$ and resulting random point in a parallelogram depicted.

Note that although points $\mathbf{q}_{i,j}^{(1)}, \mathbf{q}_{i,j}^{(2)}, \mathbf{q}_{i,j}^{(3)}$ lie on C, this need not to be true with the point

$$(x(k_0+\kappa(i+1), l_0+\ell(j+1)),\ y(k_0+\kappa(i+1), l_0+\ell(j+1)),\ z(k_0+\kappa(i+1), l_0+\ell(j+1))).$$

We thus approximate Area $(C_{i,j})$ by the area of the parallelogram, i.e., by Area $(\texttt{Paral}_{i,j})$. Now we define a categorical distribution on $C_{i,j}$

$$\mathbb{P}(X = C_{i,j}) = \frac{\text{Area}\,(\texttt{Paral}_{i,j})}{\sum_{i',j'} \text{Area}\,(\texttt{Paral}_{i',j'})}. \tag{8.27}$$

Once a parallelogram X is sampled, we sample a uniform point $\mathbf{p}$ within it (by simply sampling uniformly along corresponding vectors – see Fig. 8.10) and then chose the closest to it (not depicted).

Example 8.17 We simulated uniformly distributed points on an ellipsoid

$$\left(\frac{x}{2}\right)^2 + \left(\frac{y}{3}\right)^2 + \left(\frac{z}{5}\right)^2 = 1.$$

We used $n_k = n_l = 100$; thus, the grid was of size 10^4. In other words, the ellipsoid's surface was split into 10^4 pieces. The resulting points are depicted in Fig. 8.11

side view top view under angle

Figure 8.11: Uniformly distributed points on ellipsoid $x^2/2^2+y^2/3^2+z^2/5^2 = 1$. In side and top views point "behind" are not depicted.

In Fig. 8.12 the heatmap of areas of $\texttt{Paral}_{i,j}$ (i.e., approximations to the areas of $C_{i,j}$) are depicted, cf. with a similar Fig. 8.13 of a sphere with radius 2.

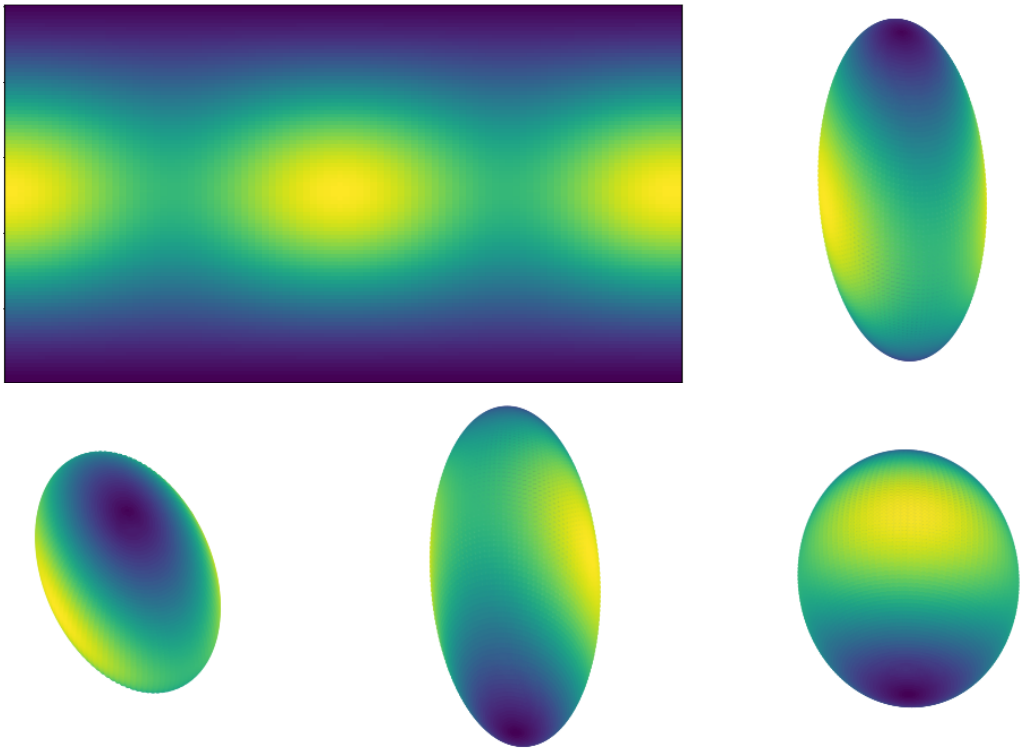


Figure 8.12: Ellipsoid $x^2/2^2 + y^2/3^2 + z^2/5^2 = 1$. Left-top: heatmap of distribution (8.27), x-axis $\theta \in [0, 2\pi)$ vs y-axis $\phi \in [0, \pi)$. Bright-yellow colors correspond to high values; dark-blue colors correspond to low values. Remaining images: several views of the ellipsoid colored correspondingly.

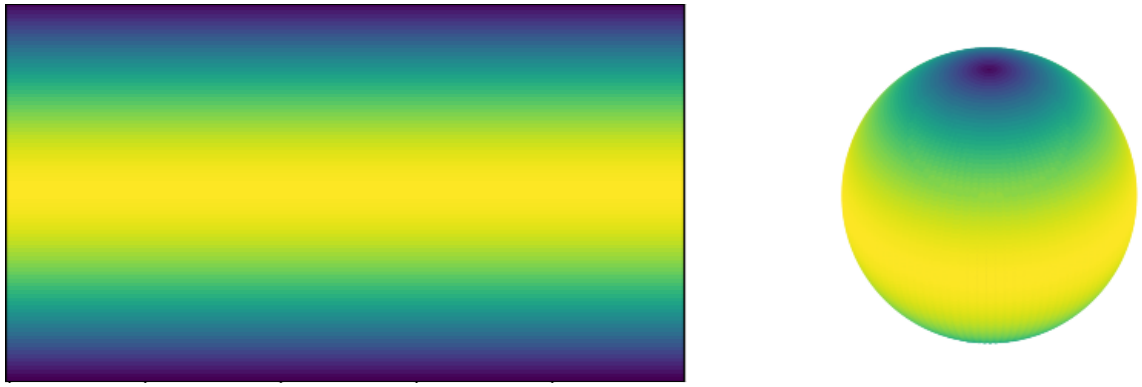


Figure 8.13: Sphere $x^2/2^2 + y^2/2^2 + z^2/2^2 = 1$. See description in Fig. 8.12.

Heatmaps from Figs 8.12 and 8.13 are depicted as regular 3d functions in Fig. 8.14.

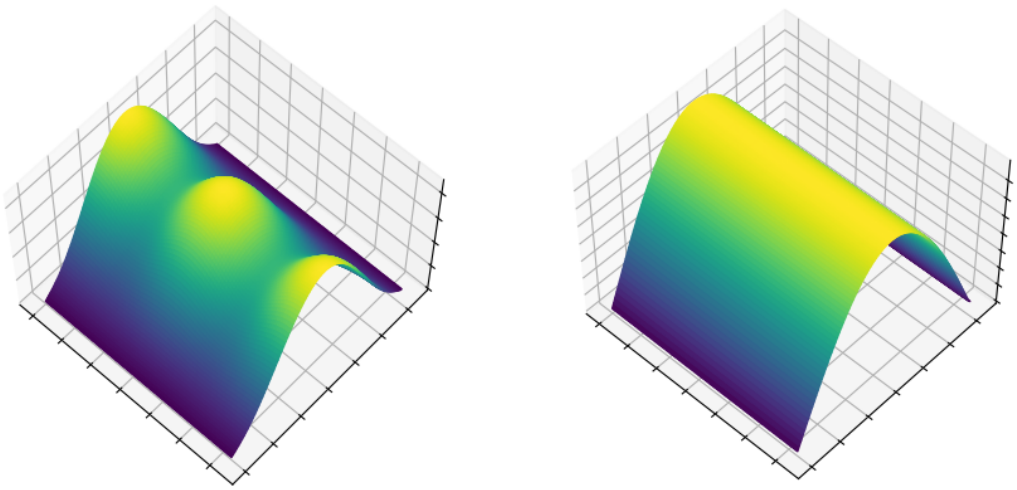


Figure 8.14: Heatmaps from Figs 8.12 (left) and 8.13 (right) depicted regular 3d functions.

By the way, if we uniformly sample angle $\theta \sim \mathcal{U}[0, 2\pi)$ and $\phi \sim \mathcal{U}[0, \pi)$, the points on an ellipsoid computed using parametrization (8.26) are of course not uniformly distributed, see Fig. 8.15.

The exact surface area of the ellipsoid is given by

$$\text{Area}\,(C) = \int_0^{\pi} \int_0^{2\pi} \sqrt{g(\phi,\theta)}\, d\phi\, d\theta,$$

with $g(\phi,\theta)$ given in (8.25). One may compute (using elliptic functions) that the exact area of an ellipse with $a = 2, b = 3, c = 5$ is

$$\text{Area}\,(C) = 134.7751.$$

From the simulations, the total area of parallelograms (thus, an approximation to Area (C)) was **134.7555**, therefore very close (note also that the approximation is smaller than the actual area – what was expected since the parallelograms are inside the ellipsoid).

Due to Knud Thomsen [5] we have the following approximation of the surface area of an ellipsoid

$$\text{Area}\,(C) \approx \tilde{S} := 4\pi \left(\frac{(ab)^p + (bc)^p + (ac)^p}{3} \right)^{\frac{1}{p}}$$

with $p = 1.6075$. The approximation is guaranteed to have a relative error $\leq 1.41544\%$. In our case $a = 2, b = 3, c = 5$ we have $\tilde{S} = 134.8149868$. Note that the error from simulations was smaller.

side view top view under angle

Figure 8.15: Non-uniformly (incorrectly) sampled points on an ellipsoid $x^2/2^2 + y^2/3^2 + z^2/5^2 = 1$: Angle θ sampled uniformly from $(0, 2\pi)$ and angle ϕ sampled uniformly from $(0, \pi)$; point (x, y, z) computed using parametrization (8.22).

[5] http://www.numericana.com/answer/ellipsoid.htm#thomsen

□

Remark 8.18 We considered rectangle with points $\mathbf{p}_{i,j},\ \mathbf{p}_{i+1,j},\ \mathbf{p}_{i,j+1},\ \mathbf{p}_{i+1,j+1}$ which was transformed into a parallelogram. Instead of this, we can build two triangles based on these points: first triangle on $\mathbf{p}_{i,j},\ \mathbf{p}_{i+1,j},\ \mathbf{p}_{i,j+1}$ and second one based on $\mathbf{p}_{i+1,j},\ \mathbf{p}_{i,j+1},\ \mathbf{p}_{i+1,j+1}$. These triangles would be transformed into two triangles – note that although the triangles in (ϕ, θ) space are on the same hyperplane, the transformed triangles are not. Then, we can proceed with a similar procedure, creating distributions on these transformed triangles. The procedure would have slightly better accuracy for the same values of n_k and n_l.

Remark 8.19 (Example from [2]). A set $K \subset \mathbb{R}^d$ is said to be star-shaped if there exists an $s_0 \in K$ such that for all $s \in K$, the line segment from s_0 to s lies in K. If s_0 is the centre, we have a star-shaped set at the origin. Examples are different types of generalised hyper-ellipsoids like a fermatoid

$$\sum_{j=1}^{d} \left(\frac{x_i}{a_i} \right)^{2m} = 1,$$

where $m = 1, 2, \ldots$.

Suppose that $K \subset \mathbb{R}^d$ is a bounded star-shaped geometrical object at the origin, which we parametrise in the following way:

$$(\rho(\mathbf{x}), \mathbf{x}), \qquad \mathbf{x} \in S_{d-1},$$

where $\mathbf{x} = \phi_1, \ldots, \phi_{d-2}, \theta$, and $\rho(\mathbf{x})$ is the radial function

$$\rho(\mathbf{x}) = \max\{\lambda > 0 : \lambda \mathbf{x} \in K\}.$$

We aim to compute $\mathrm{Vol}(K)$. Let us pass to spherical coordinates

$$\begin{aligned}
\int_{\mathbb{R}^d} \mathbf{1}_K(\mathbf{x})\, d\mathbf{x} &= \int_0^\infty \int_{S_{d-1}} \mathbf{1}_K(r\mathbf{x}) r^{d-1} \sin^{d-2}\phi_1 \ldots \sin\phi_{d-2}\, d\phi_1 \ldots\, d\phi_{d-2}\, d\theta\, dr\, d\mathbf{x} \\
&= \int_{S_{d-1}} \int_0^{\rho(\mathbf{x})} r^{d-1}\, dr\, d\mathbf{x} \\
&= \frac{1}{d} \int_{S_{d-1}} \rho(\mathbf{x})^d\, d\mathbf{x}. \\
&= \frac{\mathrm{Area}(S_{d-1})}{d} \frac{1}{\mathrm{Area}(S_{d-1})} \int_{S_{d-1}} \rho(\mathbf{x})^d\, d\mathbf{x} \\
&= \mathrm{Vol}(B_d) \frac{1}{\mathrm{Area}(S_{d-1})} \int_{S_{d-1}} \rho(\mathbf{x})^d\, d\mathbf{x} \qquad (8.28) \\
&= \frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)} \frac{1}{\mathrm{Area}(S_{d-1})} \int_{S_{d-1}} \rho(\mathbf{x})^d\, d\mathbf{x},
\end{aligned}$$

because in equation (8.28) we used $\mathrm{Area}(S_{d-1}) = d\mathrm{Vol}(B_d)$. Let $V \sim \mathcal{U}[S_{d-1}]$ be a random point on S_{d-1} and $V_1, \ldots, V_R$ be replications of V. Then

$$\hat{V}_R = \frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)} \frac{1}{R} \sum_{j=1}^{R} \rho(V_j)^d$$

is an MC estimator of $\mathrm{Vol}(K)$. In a paper [2] it was reported a Monte Carlo estimation of the volume of fermatoids and hyper-ellipsoids with the use of estimator $\hat{V}_R$. The authors reported an experiment in that they first generated a uniformly distributed set of points on the surface of the unit sphere and had a sample of such R points on the sphere. They employed these points to send out random rays from the origin to intercept the ellipsoid's surface. Then $\rho(V_j)$ is for the j-th length of the ray to o intercept the surface of K.

Remark 8.20 Notice the following fact. If $\mathbf{X}$ is a random point on the surface C and if $\mathbf{P}$ is an orthogonal matrix and $\boldsymbol{m}$ a vector in $\mathbb{R}^d$, then $\mathbf{PX} + \boldsymbol{m}$ is a random point on surface $\mathbf{P}C + \boldsymbol{m}$. It explains why our method can be applied to the general form of an ellipsoid $\mathbf{x}^T\mathbf{A}\mathbf{x} = 1$, where $\mathbf{A}$ is a positive-definite matrix which can be diagonalised.

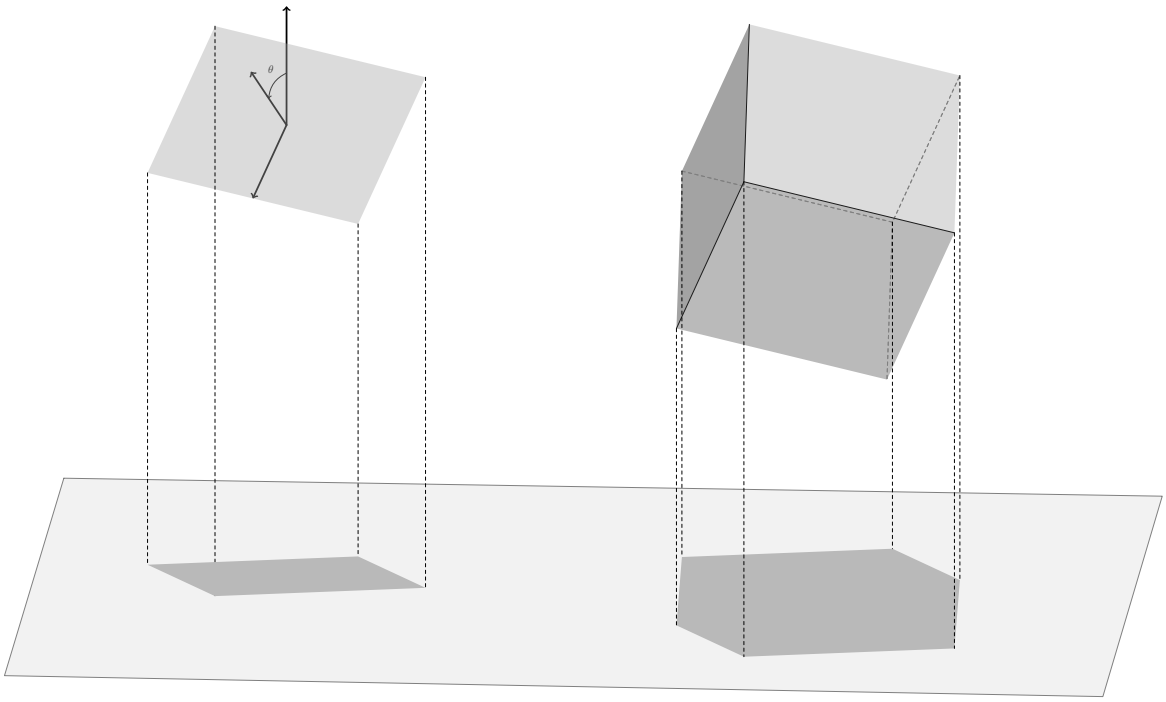


Figure 8.16: Shadow of a square (left) and shadow of a cube (right)

8.5 Example: Average area of a shadow of a convex $3d$ body.

Consider a bounded convex polyhedron $B \subset \mathbb{R}^3$ with faces $F_1, \ldots, F_m$. For example, a cube has square faces $F_1, \ldots, F_6$. Imagine sun is infinitely far away, so its rays fall parallel onto the square. The figure casts a shadow on the "ground" – a point (x, y, z) from the surface of B is projected into $(x, y, 0)$. The projected shadow, a $2d$ figure, is denoted as $\texttt{shadow}(B)$.

Let F be a face of B (e.g., square or triangle) (it is a $2d$ figure placed in $3d$ space). Assume that axis z is perpendicular to the "ground", and the face is rotated: azimuth is θ and altitude is ϕ. Note that azimuth has no influence on the area of a projected shadow (it only rotates the face) and only altitute ϕ influences shadow's area; See Fig. 8.16 (left).

The area of the shadow is $|\cos(\phi)|$ times the area of face F, i.e.,

$$\text{Area}\,(\texttt{shadow}(F)) = \text{Area}\,(F) \cdot |\cos\phi|.$$

To demonstrate it, consider first F being a triangle. Since changing azimuth does not change the area of the shadow, we may assume that one side (edge) of the triangle is parallel to the ground, and since its area is the length of the side on the ground times the height, which is for the shadow equal to the height$\times|\cos\phi|$. Now consider a bounded convex polyhedron consisting of faces $F_1, \ldots, F_m$. Every sun's ray intersects exactly **exactly two faces** – except vertices and edges – whose projected shadow area is 0. It implies, that the area of the shadow of B is half of the area of shadows of all faces, i.e.,

$$\text{Area}\,(\texttt{shadow}(B)) = \frac{1}{2}\sum_{i=1}^{m} \text{Area}\,(\texttt{shadow}(F_i)) = \frac{1}{2}\sum_{i=1}^{m} \text{Area}\,(F_i) \cdot |\cos\phi|. \tag{8.29}$$

Average shadow. Consider now *randomly* rotated face. We need to define what randomly rotated face is: the rotation can be described by a normal vector attached to the center of the face. The normal vector is then rotated randomly: we choose a point on a unit sphere and allign the vector to this point. Recall (8.22), spherical coordinates of (x, y, z).

$$\begin{aligned} x &= \cos\theta \sin\phi, \\ y &= \sin\theta \sin\phi, \\ z &= \cos\phi \end{aligned} \tag{8.30}$$

If we take $\Theta = 2\pi U_1$ and $\Phi = \arccos(1-2U_2)$ with U_1, U_2 independent $\mathcal{U}[0,1)$ random variables, then the corresponding (x, y, z) is chosen randomly from unit sphere (see .....). In other words, Θ has uniform distribution on $(0, 2\pi)$ and Φ has density $\frac{1}{2}\sin\phi$ on $(0,\pi)$. Thus, the average shadow of a face F is

given by

$$\begin{aligned}
\mathbb{E}[\text{Area}\,(\texttt{shadow}(F))] &= \text{Area}\,(F)\int_0^{2\pi}\frac{1}{2\pi}\int_0^{\pi}|\cos\phi|\frac{1}{2}\sin\phi\;d\theta\;d\phi \\
&= \frac{\text{Area}\,(F)}{2}\int_0^{\pi}|\cos\phi|\sin\phi\;d\phi \\
&= \frac{\text{Area}\,(F)}{4}\left[2\int_0^{\pi/2}\sin(2\phi)\;d\phi\right] \\
&= \frac{\text{Area}\,(F)}{2}.
\end{aligned}$$

Thus, using (8.29) we have a simple formula for an average shadow of a convex polyhedron B:

$$\begin{aligned}
\mathbb{E}(\text{Area}\,(\texttt{shadow}(B))) &= \frac{1}{2}\sum_{i=1}^{m}\mathbb{E}[\text{Area}\,(\texttt{shadow}(F_i))] \\
&= \frac{1}{2}\sum_{i=1}^{m}\frac{\text{Area}\,(F_i)}{2} = \frac{\text{Area}\,(\texttt{surface}(B))}{4}.
\end{aligned} \tag{8.31}$$

In particular, for B being a cube with edge of length s, its surface area is $\text{Area}\,(\texttt{surface}(B) = 6s^2$ and thus $\mathbb{E}(\text{Area}\,(\texttt{shadow}(B))) = \frac{3}{2}s^2$.

The formula (8.31) holds for any bounded convex body B, which can be shown by approximating B by a sequence of bounded convex polyhedrons B_n. Note that it agrees for a ball: a ball of radius r has surface are $4\pi r^2$, thus its expected area of a shadow is $4\pi r^2/4 = \pi r^2$, which is the area of a circle (here of course a shadow is constantly the same circle, indendently of rotation of a ball).

9 Simulating bivariate random vectors; using conditional inverse transform method or copulas

9.1 Conditional inverse transform method

We focus here on two-dimensional random variables; however, the method works in the multidimensional case. Consider (X, Y) with a joint p.d.f. $f(x, y)$. Assume we know that X takes values in $[x_0, x_1)$, whereas Y takes values in $y \in [y_0, y_1)$, i.e. that

$$\int_{x_0}^{x_1} \int_{y_0}^{y_1} f(x, y)\, dx\, dy = 1.$$

Marginal c.d.f.s are

$$F_X(x) = \int_{y_0}^{y_1} f(x, y)\, dy \qquad F_Y(y) = \int_{x_0}^{x_1} f(x, y)\, dx.$$

Conditional c.d.f. of Y under condition $X = \mathsf{x}$ is (provided $f_X(\mathsf{x}) \neq 0$)

$$F_{Y|X=\mathsf{x}}(y) = \frac{F_{X=\mathsf{x},Y}(\mathsf{x}, y)}{f_X(\mathsf{x})},$$

where

$$F_{X=\mathsf{x},Y}(\mathsf{x}, y) = \int_{y_0}^{y} f(\mathsf{x}, y)\, dy.$$

We may compute c.d.f $F_{Y|X=\mathsf{x}}$ by computing the first partial derivative of $F(x, y)$ with respect to x at $x = \mathsf{x}$; denote it by $F_x'(\mathsf{x}, y)$.

Of course, if X and Y are independent, we compute their marginal distributions and sample them independently. If they are not independent, then we compute marginal distribution of X (or Y) and sample from the suitable conditional distribution. To be more precise:

- Compute marginal $f_X(x) = \int_{x_0}^{x_1} f(x, y)\ dx$ and its c.d.f. $F_X(x) = \int_{x_0}^{x} f_X(s)\ ds$.
- Sample from this marginal (ITM): $U_1 \sim \mathcal{U}[0, 1), X = F_X^{\leftarrow}(U_1)$.

- Conditional: Say $X = \mathsf{x}$. Generate Y given $X = \mathsf{x}$, i.e., from $F_{Y|X=\mathsf{x}}(y) = F_x'(\mathsf{x}, y)$. This can be done by the ITM method: Generate $U_2 \sim (0, 1)$ (independent of U_1) and set $Y = F^{\leftarrow}_{Y|X=\mathsf{x}}(U_2)$.

- Output (X, Y).

Example 9.1 Random variable (X, Y) has a joint p.d.f.

$$f(x, y) = x \exp(-x(y + 1)), \text{ for } x, y \geq 0.$$

The marginal p.d.f.s are

$$f_X(x) = e^{-x}, \quad f_Y(y) = \frac{1}{(y + 1)^2},$$

i.e., the exponential (with mean 1) and Pareto distributions Par(2) (with mean 1), respectively. We sample from Exp(1) simply taking $X = -\log(U_1)$ for $U_1 \sim \mathcal{U}[0, 1)$. Say $X = \mathsf{x}$. The marginal distribution

$$f_{Y|X=\mathsf{x}}(y) = \frac{f(\mathsf{x}, y)}{f_X(\mathsf{x})} = \frac{\mathsf{x} \exp(-\mathsf{x}(y + 1))}{\exp(-\mathsf{x})} = \mathsf{x} \exp(-\mathsf{x}y),$$

which is an exponential distribution with parameter x, which in turn we sample in a usual way by $Y = -\log(U_2)/\mathsf{x}$ for $U_2 \sim \mathcal{U}[0, 1)$ (independent from U_1). □

9.2 Bivariate copulas

By a copula $C(u_1, u_2)$, one means a bivariate c.d.f. with uniformly $\mathcal{U}[0, 1)$ distributed marginal c.d.f.s. Suppose that (X, Y) is a random vector with a joint c.d.f. $F_{X,Y}(t_1, t_2)$. Marginal distributions are $F_X(t_1) = F_{X,Y}(t_1, \infty)$ and $F_Y(t_2) = F_{X,Y}(\infty, t_2)$ respectively. Note that we do not assume that random variables X, Y are independent. In this case, the knowledge of marginal c.d.f.s F_X and F_Y is not sufficient, and a dependence structure is given by a copula.

Formally, by a bivariate *copula* we mean any two-dimensional c.d.f. with uniform $\mathcal{U}[0, 1)$ marginal distributions. We will denote it by $C(t_1, t_2)$. Thus, a function $C : [0, 1] \times [0, 1]$ is a **copula** if

1. $\lim_{u_i \to 0} C(u_1, u_2) = 0$, for $i = 1, 2$,

2. $\lim_{u_1 \to 1} C(u_1, u_2) = u_2$ and $\lim_{u_2 \to 1} C(u_1, u_2) = u_1$,

3. $C(v_1, v_2) - C(u_1, v_2) - C(v_1, u_2) + C(u_1, u_2) \geq 0$ for any $u_1 \leq v_1$, $u_2 \leq v_2$.

One can prove the following result. Remark that a general version for d-dimensional random vectors is called *Sklar's theorem*.

Theorem 9.2 *For any c.d.f. $F_{X,Y}(t_1, t_2)$ of a random vector (X, Y), there exists a copula $C(u_1, u_2)$ such that*

$$F_{X,Y}(t_1, t_2) = C(F_X(t_1), F_Y(t_2)), \qquad -\infty < t_1, t_2 < \infty. \tag{9.32}$$

If the marginal distributions $F_X(t)$ and $F_Y(t)$ ***are continuous, then*** *C* ***is unique****. Conversely, if C is a copula, and F_X, F_Y are c.d.f.s, then the function*

$$F_{X,Y}(t_1, t_2) = C(F_X(t_1), F_Y(t_2))$$

is a two-dimensional c.d.f. with marginal distributions F_X, F_Y.

Note that from (9.32) we immediately get that if (X, Y) has a joint density $f_{X,Y}$, we have a formula

$$f_{X,Y}(t_1, t_2) = f_X(t_1) f_Y(t_2) c(F_X(t_1), F_Y(t_2)), \tag{9.33}$$

where $c(u_1, u_2)$ is a joint p.d.f. of a copula C and f_x, f_Y are marginal p.d.f.s. From the above formula, we may also write the expression for $\mathbb{E}(k_1(X)k_2(Y))$ for (X, Y) in terms of a copula with density c:

$$\mathbb{E}(k(X)k(Y)) = \int \int k_1(t_1) k_2(t_2) f_X(t_1) f_Y(t_2) c(F_X(t_1), F_Y(t_2)) dt_1 \, dt_2.$$

Since marginal c.d.f.s are continuous, copulas cannot have atoms, and therefore $C(v_1, v_2)$ are continuous functions of two variables v_1, v_2. It turns out (see e.g. Embrechts *et al* [38]), that we have a decomposition

$$C(v_1, v_2) = pC_{\mathrm{s}}(v_1, v_2) + (1 - p)C_{ac}(v_1, v_2),$$

where $0 \leq p \leq 1$, C_{s} is a singular c.d.f, and C_{ac} is an absolute continuous part (that is having a p.d.f.). Examples of singular copulas are C_L and C_U and of the decomposition is the copula in (9.36). Therefore the only interesting in the sequel for us is the case when p.d.f. C has joint p.d.f. $c(v_1, v_2)$. In this case, a joint distribution of (X, Y) is given by a triplet (F_X, F_Y, C).

Recall a concept of a *generalised inverse* $F^{\leftarrow}(t)$ given in (1.1) in Section 1. We have the following prescription for C (given joint distribution $F_{X,Y}$).

Proposition 9.3 *Let $F_{X,Y}$ be a joint c.d.f. with continuous marginal c.d.f.s F_X, F_Y. Then the unique copula of $F_{X,Y}$ is given by*

$$C_{X,Y}(u_1, u_2) = F_{X,Y}(F_X^{\leftarrow}(u_1), F_Y^{\leftarrow}(u_2)). \tag{9.34}$$

Below is a generalisation of the well known property that if $F(t)$ is continuous distribution of random variable X, then $F(X)$ is uniformly $\mathcal{U}[0,1)$-distributed.

Proposition 9.4 *In the case of continuous marginal c.d.f.s F_X, F_Y, we have that $(F_X(X), F_Y(Y))$ has the c.d.f., which is a copula $C(u_1, u_2)$ defined by (X, Y).*

From now on we assume that marginal distributions of a joint distribution are continuous. In the survival analysis, instead of c.d.f.s one uses survival function. Therefore let us introduce the corresponding **tail copula**

$$C^*(t_1, t_2) = 1 - t_1 - t_2 + C(t_1, t_2),$$

which is the probability $\mathbb{P}(U_1 > t_1, U_2 > t_2)$.

Lemma 9.5 *For a bivariate survival function $S_{X,Y}(t_1, t_2) = \mathbb{P}(T_x > t_1, T_y > t_2)$ with marginal survival functions denoted S_X and S_Y respectively, we have*

$$S_{X,Y}(t_1, t_2) = C^*_{X,Y}(1 - S_X(t_1), 1 - S_Y(t)).$$

General procedure for sampling. Let $(V_1, V_2) \sim C$. Simulate first V_1, say $V_1 = v_1$, where $V_1 \sim \mathcal{U}[0,1)$. Remembering that marginals have p.d.f. equal 1, the conditional p.d.f. of V_2 given $V_1 = \mathsf{v}_1$ is the first derivative of $C(v_1, v_2)$ with respect v_1 evaluated at $v_1 = \mathsf{v}_1$; denoted by $C'_{v_1}(\mathsf{v}_1, v_2)$.

Thus, we may sample from a copula C in the following way (cf. with Section 9.1):

- Sample independent $U_2, V_1 \sim \mathcal{U}[0,1)$. Say $U_2 = \mathsf{u}_2, V_1 = \mathsf{v}_1$.
- Set $V_2 = (C'_{v_1})^{\leftarrow}(\mathsf{v}_1, \mathsf{u}_2)$.
- Return (V_1, V_2).

Note that in some cases computing the derivative of C_1 and/or general inverse $(C'_{v_1})^{\leftarrow}(\mathsf{v}_1, \mathsf{u}_2)$ may be difficult or impossible in a closed form. Often copula-specific methods are used.

Fréchet-Hoeffding bounds The following inequality will explain the possibilities of the application of copulas, and their use in practical situations.

Theorem 9.6 *[Fréchet-Hoeffding bounds] For a two-dimensional c.d.f. $F(x,y)$ of (Y_1,Y_2) with the same marginal distribution $F(x) = \mathbb{P}(Y_1 \le x) = \mathbb{P}(Y_2 \le x)$,*

$$(F(x)+F(y)-1)_+ \le F(x,y) \le \min(F(x),F(y)). \tag{9.35}$$

The lower bound is attained for $\mathbb{P}(F^{\leftarrow}(U) \le x, F^{\leftarrow}(1-U) \le y)$.

The remark on the lower bound can be justified as follows. Using Lemma 1.1 (a) we have

$$\begin{aligned}\mathbb{P}(F^{\leftarrow}(U) &\le x, F^{\leftarrow}(1-U) \le y) = \mathbb{P}(U \le F(x), 1-U \le F(y)) \\ &\le F(x)+F(y)-1,\end{aligned}$$

provided $F(x)+F(x)+F(y)-1 \ge 0$.

Remark 9.7 The inequality (9.35) can be expressed in terms of one of the stochastic ordering of random vectors $\mathbf{X}_1 = (X_{11}, X_{12})$ and $\mathbf{X}_2 = (X_{21}.X_{22})$ with joint c.d.f. $F_1(x,y)$ and $F_2(x,y)$ respectively.

- : It is said that $\mathbf{X}_1 \le_{\text{lo}} \mathbf{X}_2$ if $F_1(x,y) \ge F_2(x,y)$ for all x, y,
- Another stochastic order is defined via the tails of c.d.f. $\bar{F}_i(x,y) = 1+F_i(x,y)-F_i(x,\infty)-F_i(\infty,y)$, that is the measure of upper orthants. Then we define $F_1 \le_{\text{uo}} F_2$ if $\bar{F}_1(x,y) \le \bar{F}_2(x,y)$ for all x, y.

Let U_1, U_2 be i.i.d. uniformly $\mathcal{U}[0,1)$ distributed random variables. If $F_2(x,y) = F(x,y)$ and $F_1(x,y)$ is the c.d.f. of $(F^{\leftarrow}(U_1), F^{\leftarrow}(1-U_2)$, then in terms of stochastic ordering we have $F_1 \ge_{\text{lo}} F_2$. However, since in our case $F_1(x,y)$ and $F_2(x,y)$ have the same marginal c.d.f. $F(x)$ it is also true that $F_1 \le_{\text{uo}} F_2$. In the book of Müller and Stoyan [100] there is the following Theorem 3.3.16: $\mathbf{X}_1 \le_{\text{uo}} \mathbf{X}_2$ if and only if for every univariate non-decreasing function k_i (plus some integrability condition) $\mathbb{E}k_1(X_{11})k_2(X_{12}) \le \mathbb{E}k_1(X_{21})k_2(X_{22})$. Hence we can conclude as follows: for all (X_1^*, X_2^*) with the same marginals $X_1^* \sim X, X_2^* \sim X$ we have

$$\text{Corr}\,(F^{\leftarrow}(U), F^{\leftarrow}(1-U)) \le \text{Corr}\,(X_1^*, X_2^*).$$

We will come back to these derivations later in the section on antithetic sampling; see Section V.1.2.

9.3 Some classical copulas

We will denote a random vector with c.d.f. $C(t_1, t_2)$ by (V_1, V_2). Simple copulas are:

- The Fréchet upper bound copula

$$C_U(u_1, u_2) = \min(u_1, u_2), \qquad 0 \leq u_1, u_2 \leq 1;$$

  simulated by $V_1 = U, V_2 = U$, where $U \sim \mathcal{U}[0, 1)$.

- The Fréchet lower bound copula

$$C_L(u_1, u_2) = \max(0, u_1 + u_2 - 1), \qquad 0 \leq u_1, u_2 \leq 1;$$

  simulated by $V_1 = U, V_2 = 1 - U$, where $U \sim \mathcal{U}[0, 1)$.

- the independence formula

$$C_I(u_1, u_2) = u_1 u_2, \qquad 0 \leq u_1, u_2 \leq 1;$$

  simulated it by $V_1 = U_1$ $V_2, = U_2$, where U_1, U_2 are independent and uniformly distributed $\mathcal{U}[0, 1)$.

Remark 9.8 The Fréchet upper bound corresponds to the unit mass spread over the main diagonal $u_1 = u_2$, the Frechet lower bound corresponds to the unit mass spread over the diagonal $u_1 = 1 - u_2$. Clearly C_I is the c.d.f. of independent uniforms. A convex mixture of C_U, C_L, C_I is again a copula. Fréchet proposed a new copula

$$C_{\alpha,\beta} = \alpha C_U + (1 - \alpha - \beta) C_I + \beta C_L, \tag{9.36}$$

where $\alpha, \beta \geq 0, \alpha + \beta \leq 1$.

Example 9.9 [Normal copula] Recall a bivariate normal distribution (X, Y) with mean zero and correlation matrix

$$\mathbf{\Sigma} = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix},$$

where $\mathrm{Var}(X) = \mathrm{Var}(Y) = 1$ and ρ is the correlation between X, Y. Its p.d.f. is

$$\phi_{\mathbf{\Sigma}}(t_1, t_2) = \frac{1}{2\pi\sqrt{1 - \rho^2}} \exp\left(-\frac{z}{2(1 - \rho^2)}\right),$$

where $z = t_1^2 - 2\rho t_1 t_2 + t_2^2$. We denote by $\Phi_{\mathbf{\Sigma}}$ the corresponding bivariate normal c.d.f.. According to Proposition 9.3, the Gaussian (normal) copula is

$$C(u_1, u_2) = \Phi_{\mathbf{\Sigma}}(\Phi^{-1}(u_1), \Phi^{-1}(u_2)),$$

which written out gives

$$C(u_1, u_2) = \int_{-\infty}^{\Phi^{-1}(u_1)} \int_{-\infty}^{\Phi^{-1}(u_2)} \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}(x^2 + y^2 - 2\rho xy)\right) dx\, dy.$$

To simulate (V_1, V_2), notice that according to Proposition 9.4

$$(V_1, V_2) = (\Phi(X), \Phi(Y)), \tag{9.37}$$

where $(X, Y) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. We learnt in Section 7 how to simulate normally distributed (X, Y). In a case when (X, Y) is a bivariate normal vector with arbitrary covariance matrix $\mathbf{\Sigma}$, then we simulate it by

$$(V_1, V_2) = (\Phi(X/\sqrt{\mathrm{Var}X}), \Phi(Y/\sqrt{\mathrm{Var}Y})).$$

We leave to the reader to show that when $\rho \to 0$, the limiting copula is C_I and when $\rho \to \pm 1$, the limiting copula is C_U and C_L respectively.

In Figure 9.17 there are shown clouds of 200 replications of normal copulas with $\rho = 0.9$ and $\rho = -0.8$ respectively.

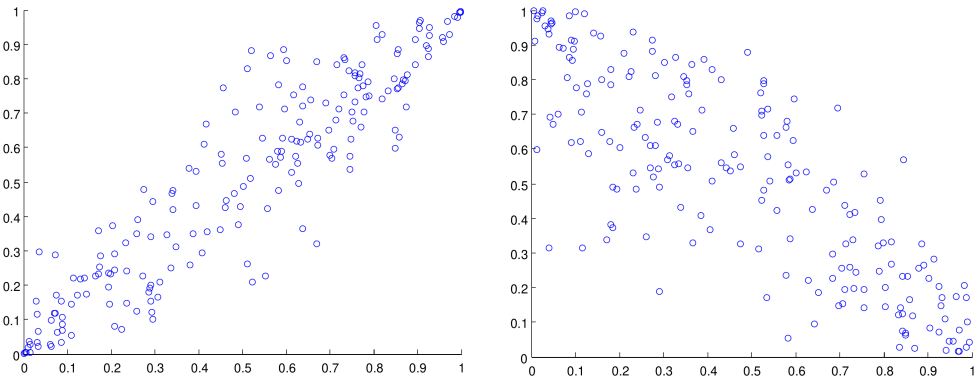


Figure 9.17: Simulated normal copula: $\rho = 0.9$ (left), $\rho = -0.8$ (right)

□

Archimedean copulas is a popular class of copulas. A copula C is called Archimedean if

$$C(u_1, u_2) = \psi^{[-1]}(\psi(u_1) + \psi(u_2)),$$

where $\psi : [0,1] \to \mathbb{R}_+$ is continuous, strictly decreasing and strictly convex function such that $\psi(1) = 0$ and

$$\psi^{[-1]} = \begin{cases} \psi(t) & 0 \le t \le \psi(0), \\ 0 & \psi(0) \le t \le \infty. \end{cases}$$

Notice that for $\psi(t) = -\log t$ we have $C_I(x, y) = xy$.

Suppose we are able to have a closed form of $C'_{\mathsf{u}_1}(\mathsf{u}_1, u_2)$ and its inverse w.r.t. u_2 in closed forms. We then have the following algorithm for generating copula (V_1, V_2).

Algorithm 23 ITM copula

1: Generate i.i.d. U_1 and U_2 uniformly $\mathcal{U}[0,1)$ distributed.
2: Substitute $V_1 = U_1$ and $V_2 = (C'_{u_1})^{\leftarrow}(V_1, U_2)$

Example 9.10 [Clayton] A non-trivial Archimedean copula is the so called Clayton copula. It is defined by $\psi(u) = (t^{-\theta} - 1)/\theta$ and then

$$C(u_1, u_2) = \{\max(u_1^{-\theta} + u_2^{-\theta} - 1, 0)\}^{-1/\theta},$$

$\theta \in (-1, 0) \cup (0, \infty)$. Assuming $\theta > 0$, $C(u_1, u_2) = (u_1^{-\theta} + u_2^{-\theta} - 1)^{-1/\theta}$ and then

$$\begin{aligned} C'_{u_1}(v_1, v_2) &= -\frac{1}{\theta}(v_1^{-\theta} + v_2^{-\theta} - 1)^{-\theta^{-1}-1}(-\theta v_1^{-\theta-1}) \\ &= (u_1^{-\theta} + u_2^{-\theta} - 1, 0)^{-\frac{1+\theta}{\theta}}(u_1)^{\frac{1+\theta}{\theta}} = (1 + u_1^{\theta}(u_2(u_2^{-\theta} - 1)))^{-\frac{1+\theta}{\theta}}. \end{aligned}$$

Hence, for u_1, inverting with respect u_2, we obtain

$$(C'_{u_1}(\mathsf{u}_1, u_2))^{\leftarrow} = \left(\frac{u_2^{-\frac{\theta}{1+\theta}} - 1}{\mathsf{u}_1^{\theta}} + 1 \right)^{-1/\theta}.$$

In Figure 9.18 a cloud a 100 replications of the Clayton formula is presented for $\theta = 10$ and 2 respectivley.

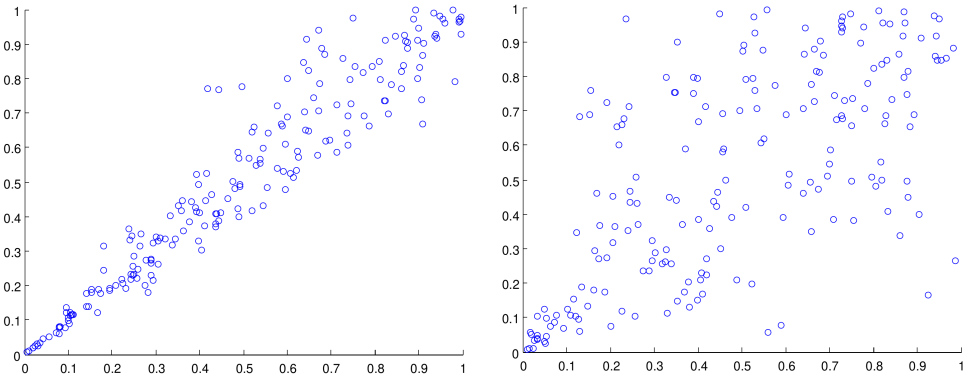


Figure 9.18: Simulated Clayton copula: $\theta = 10$ (left), $\theta = 2$ (right)

.

□

Example 9.11 [Frank] The copula is defined by

$$\psi(u) = -\log\left(\frac{\exp(-\theta u) - 1}{\exp(-\theta) - 1}\right),$$

and

$$C(u_1, u_2) = -\frac{1}{\theta}\log\left(1 + \frac{(\exp(-\theta u_1) - 1)(\exp(-\theta u_2) - 1)}{\exp(-\theta) - 1}\right),$$

$\theta \in \mathbb{R} - \{0\}$. The first derivative w.r.t. u_1 is

$$C_1(u_1, u_2) = \frac{e^{-\theta u_1}(e^{-\theta u_2} - 1)}{e^{-\theta} - 1 + (e^{-\theta u_1} - 1)(e^{-\theta u_2} - 1)}.$$

The inverse to $C'_{u_1}(u_1, u_2)$ with respect to u_2 is

$$(C'_{u_1})^{\leftarrow}(u_1, u_2) = -\frac{1}{\theta}\log\left(\frac{(e^{-\theta} - 1)u_2}{e^{-\theta u_1} - (e^{-\theta u_1} - 1)u_2} + 1\right).$$

In Figure 9.19 a cloud of 200 replications of Frank's copula are shown.

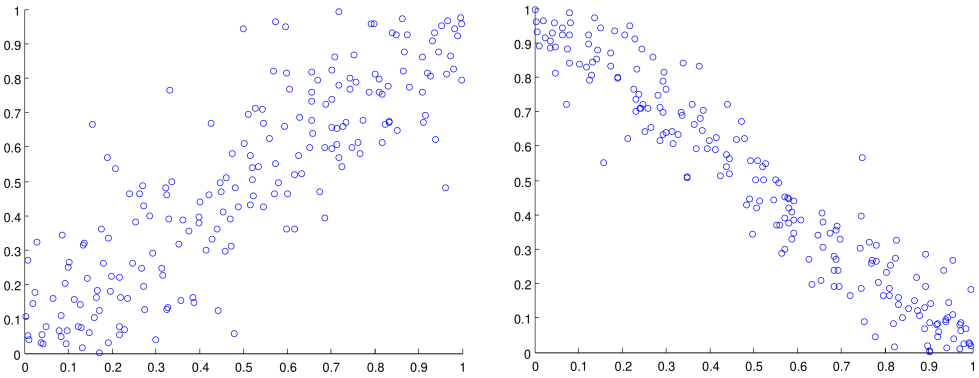


Figure 9.19: Simulated Frank copula: $\theta = 10$ (left), $\theta = -20$ (right).

□

Note that for a general Archimedean copula (V_1, V_2) case defined by $\psi(t)$, there exists the following algorithm. We will need:

$$K(t) = t - \frac{\psi(t)}{\psi'(t+)},$$

which is a c.d.f.

Algorithm 24 ψ-Archimeadean copula

1: Generate i.i.d. U_1 and U_2 uniformly $\mathcal{U}[0,1)$ distributed.
2: Set $T = K^{-1}(U_2)$
3: Set $V_1 = \psi^{[-1]}(U_1\psi(T))$
4: Set $V_2 = \psi^{[-1]}((1-U_1)\psi(T))$

The influence of a dependence structure on a joint distribution can be seen from the following simulations of 100 replications of pairs (X, Y) with exponental and Pareto marginals cut off at 10:

- of (X, Y) from Example 9.1; displayed in Figure 9.20 below,
- of $(-\log V_1, V_2^{\frac{1}{2}} - 1)$, where (V_1, V_2) is the copula from (9.37), that is a normal copula; with correlation $\rho = -0.5$ and 0.8 respectively. dispalyed in Figure 9.21

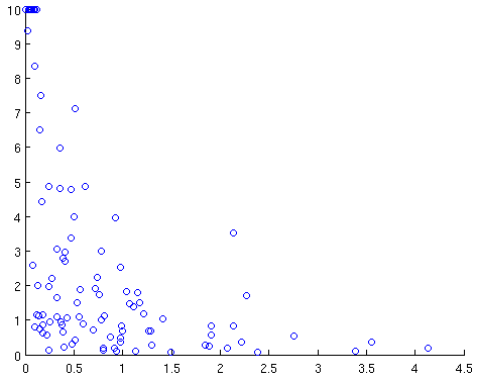


Figure 9.20: Cloud of 100 replications of (X, Y) from Example 9.1.

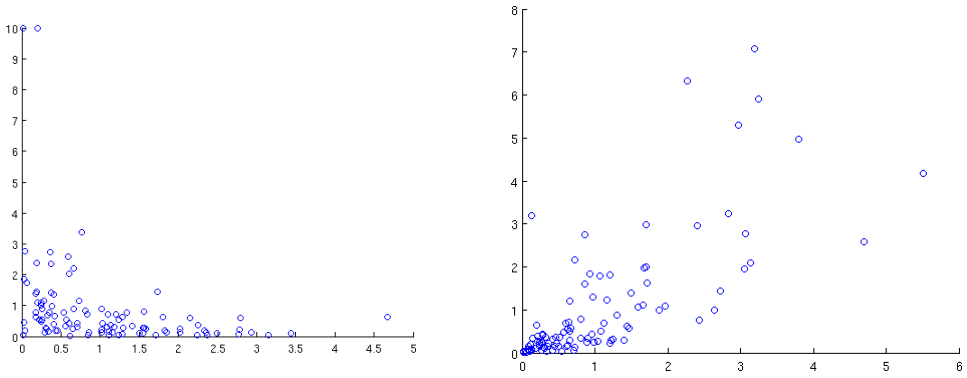


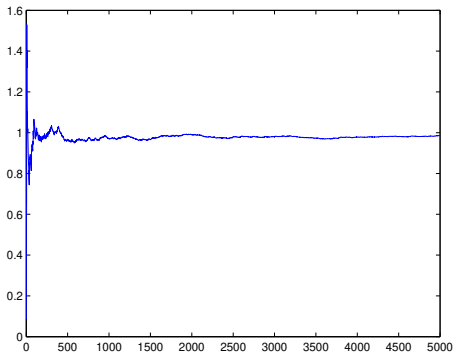
Figure 9.21: Cloud of 100 replications of (X, Y) with exponential Exp(1) and Pareto Par(2) (with mean 1) marginals; normal copula: $\rho = -0.5$ (left), $\rho = 0.8$ (right).

10 More samples of simulations

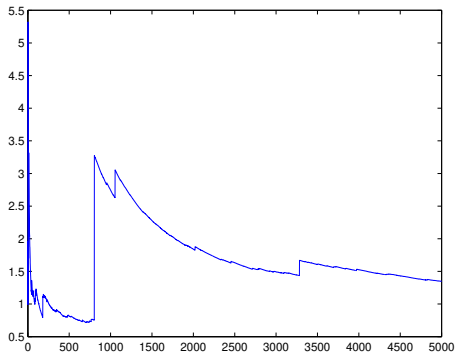
We will now demonstrate sample simulations using the algorithms covered in this chapter. The second important objective will be to visually 'observe' random variables.

Example 10.1 Suppose we generate a sequence $X_1, \ldots, X_R$ of i.i.d. random numbers and set $S_n = \sum_{i=1}^{n} X_i, n = 1, \ldots, R$ and $S_0 = 0$. In Fig. 10.22 we display the values of S_n/n, $n = 1, \ldots, 5000$ for the distributions $X_i \sim$Exp(1)

and $X_i = 0.2Y_i$, where $Y_i \sim$Par(1.2). Note that Pareto variables Par(1.2) have mean 5, that is why they are multiplied 0.2, so that the mean of X_i is 1. It is worth paying attention to these simulations. You can see that for Pareto variables, the convergence is dramatically slower, which is an effect of their heavy tail. Thus for Monte Carlo computation of the mean with a specified accuracy, one must make more replications. We will deal with this topic in the next chapter.



(a)



(b)

Figure 10.22: Plots of $S_n/n, n = 1, \ldots, 5000$ where $S_n = \sum_{i=1}^{n} X_i$ and X_i are i.i.d. random variables with distribution a) Exp(1); b) Pareto(1.2) multiplied by 0.2.

□

Example 10.2 In Fig. 10.23 the results of simulating $R = 15000$ random variables:

$$X_1, \ldots, X_{5000}, X_{5001}, \ldots, X_{10000}, X_{10001}, \ldots, X_{15000}$$

are depicted. All of them are independent, but their distributions are different:

- $X_1, \ldots, X_{5000}$ are i.i.d. with distribution $\mathcal{N}(1,1)$,
- $X_{5001}, \ldots, X_{10000}$ are i.i.d. with distribution Exp(1),
- $X_{10000}, X_{10001}, \ldots, X_{15000}$, where $X_i = 0.2Y_i$ and $Y_i, i = 10000, \ldots, 15000$ are i.i.d. with the distribution Par(1.2).

We recommend paying attention to the nature of the individual fragments in Fig. 10.23. Although all the variables have the mean 1, in the first fragment (for normally distributed variables) there are a few variables scattered, in the second part, a bit more, while a large scatter can be observed in the last part (i.e. for the Pareto variables). It is due to the decay tail becoming heavier and heavier.

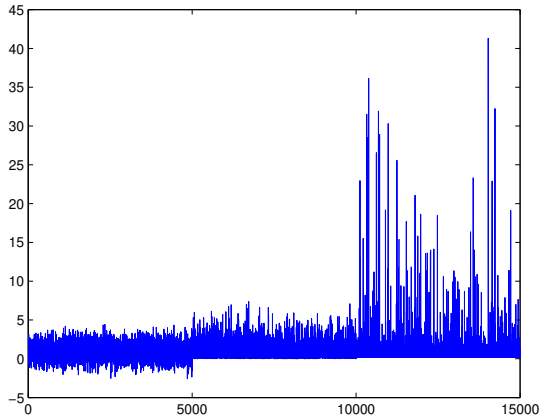


Figure 10.23: Simulated independent random variables: $X_1, \ldots, X_{5000}$ i.i.d. $\mathcal{N}(1,1)$; $X_{5001}, \ldots, X_{10000}$ i.i.d. Exp(1); $X_{10000}, X_{10001}, \ldots, X_{15000}$ i.i.d. Par(1.2) multiplied by 0.2.

□

Example 10.3 We now depict the results of simulations of a random vector (X_1, X_2) with a two-dimensional normal distribution. In each case we have $\mathrm{Var}(X_1) = 3$ and $\mathrm{Var}(X_2) = 1$. In Fig. 10.24, the correlation coefficient $\rho = 0.1$ was used, whereas in Fig. 10.25 (a) and (b) this coefficient was equal to 0.6 and 0.9 respectively. [6]

[6] skrypt: normalnyRho.m

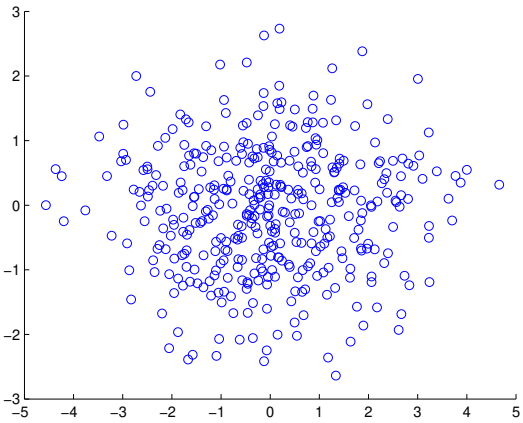
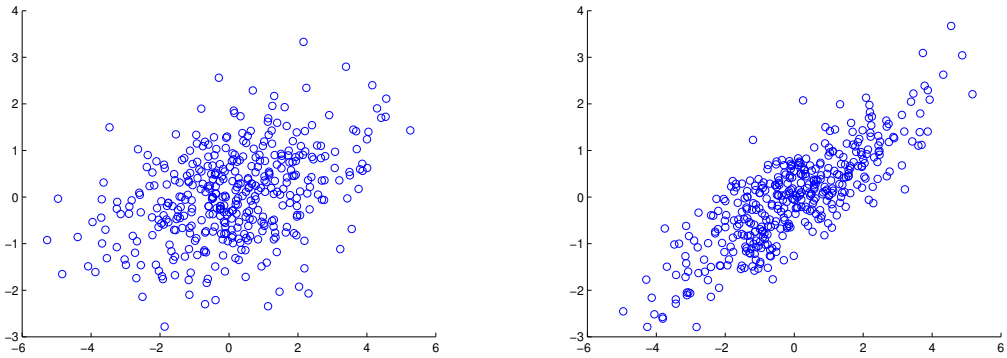


Figure 10.24: The result of simulations of normally distributed random vector (X_1, X_2) with $\mathrm{Var}X_1 = 3$, $\mathrm{Var}X_2 = 1$ and with $\rho = 0.1$. 400 replications.



(a) (b)

Figure 10.25: The result of simulations of normally distributed random vector (X_1, X_2) with $\mathrm{Var}X_1 = 3$, $\mathrm{Var}X_2 = 1$ and with a) $\rho = 0.6$; b) $\rho = 0.9$. 400 replications.

□

10.1 Bibliographical comments for Chapter III

On the topic of generating random variables with given distributions, an extensive literature exists. For example, refer to an elementary book by Ross [115], a fundamental monograph by Asmussen and Glynn [6], an extensive monograph by Fishman [42], and Dagpunar [26]. A fascinating little booklet by Madras [88] is also recommended. Properties of generalized inverse functions can be found in Embrechts and Hofert [37].

There is a substantial body of literature on generating random elements uniformly distributed on random objects scattered around in different places. A good reference for the general theory of area measure on manifolds, such as hyper-spheres and hyper-ellipsoids, is Chapter 5 in the book by Munkres [103]. For a fundamental tool like spherical coordinates, see, for example, a note by Blumenson [16]. The relationship between normal random variables and uniformly distributed points on a hyper-sphere has been explored in papers by Muller [102, 101, 18]. A modification of the Box-Muller method was proposed by Marsaglia and Bray [93]. For some other methods not discussed in this chapter, refer to the paper by Harman and Lacko [57].

For copulas, consult the book by Nelsen [105] and the paper by Embrechts et al. [38].

Note other methods to generate a random point on the sphere is in Marsaglia [90].

11 Exercises

Theoretical exercises

III.T.1 Demonstrate properties P2,P5, P6 of the generalized inverse $F^{\leftarrow}(t)$ from Section 1 (ITM method; exponential and Pareto distribution), with the help of diagrams like enclosed in this section.

III.T.2 Let $x_1 < x_2 < \ldots < x_n$ and $p_i > 0$, $\sum_{i=1}^n p_i = 1$. Suppose that c.d.f is given

$$F(x) = \begin{cases} 0 & x < x_1 \\ p_1 + \ldots + p_i & x_i \le x < x_{i+1}, \quad i = 1, \ldots, n-1, \\ 1 & x_n \le x. \end{cases}$$

Write down $F^{\leftarrow}(t)$.

III.T.3 Suppose that $X_1, \ldots, X_R$ are i.i.d. with a continuous cumulative distribution function and let $\hat{F}_R(x) = \sum_{i=1}^R \mathbf{1}(X_i \le x)/R$ be the empirical distribution function. Find $\hat{F}_R^{\leftarrow}(u)$. What happens when we leave out the assumption of continuity?

III.T.4 Let X be a random variable with a c.d.f. F and consider a conditional random variable $Y \stackrel{\mathcal{D}}{=} [X|X \in (a,b)]$, where $\mathbb{P}(X \in (a,b]) > 0$ (or $F(b) > F(a)$).
a) Show that $Y \sim G$, where

$$G(x) = \begin{cases} 0 & x < a, \\ \frac{F(x)-F(a)}{F(b)-F(a)}, & a \le x < b, \\ 1 & x \ge b \end{cases}$$

b) Let

$$V = F(a) + (F(b) - F(a))U, \quad \text{where } U \sim \mathcal{U}[0,1).$$

Show that $Y \stackrel{\mathcal{D}}{=} F^{\leftarrow}(V)$ has the required c.d.f. G.

III.T.5 Let $X \sim F$. Show that a random variable distributed as the overshoot $Y \stackrel{\mathcal{D}}{=} (X - a|X > a)$ can be generated as

$$F^{\leftarrow}(U\overline{F}(a) + F(a)) - a.$$

Here $U \sim \mathcal{U}[0,1)$ and $\overline{F}(a) < 1$.

III.T.6 Modify Algorithm 10 (ITR) to allow generating lattice random variables on $\mathbb{Z}$ with probability function $(p_i,\ i \in \mathbb{Z})$.

III.T.7 Let $U_1, U_2, \ldots$ be i.i.d. $\mathcal{U}[0,1)$ distributed random variables. Define

$$N = \min\{n : n \geq 2,\ U_n > U_{n-1}\}.$$

(a) Show that $\mathbb{E}N = e$ (hence, e can be estimated simulating uniform random varaibles and stopping the first time one exceeds its direct predecessor, one needs to repeat the procedure several time to estimate $\mathbb{E}N$).

(b) Compute $\mathrm{Var}N$.

Hint: Compute $\mathbb{P}(N > n)$ and then the distribution $\mathbb{P}(N = n) = \mathbb{P}(N > n-1) - \mathbb{P}(N > n)$.

III.T.8 There is given c.d.f.

$$F(x) = \begin{cases} 0 & \text{for} \quad x < 0, \\ \frac{1}{2}x & \text{for} \quad x \in [0,1), \\ \frac{1}{2} & \text{for} \quad x \in [1,2), \\ \frac{3}{4} & \text{for} \quad x \in [2,3), \\ \frac{3}{8} + \frac{1}{8}x & \text{for} \quad x \in [3,5), \\ 1 & \text{for} \quad x \geq 5. \end{cases}$$

Write a procedure to generate $Y \sim F$ with the use of the composition method.

III.T.9 Write a procedure to generate a random number X with triangular distribution Tri(a,b), having density

$$f(x) = \begin{cases} c(x-a), & a < x < (a+b)/2, \\ c((b-x), & (a+b)/2 < x < b, \end{cases}$$

where the normalising constant is $c = 4/(b-a)^2$.

Hint: Consider first the density of $U_1 + U_2$.

III.T.10 Suppose that $U \sim \mathcal{U}[0,1)$. Demonstrate that

$$X = \log(U/(1-U))$$

has logistic distribution with c.d.f. $F(x) = 1/(1+e^{-x})$.

III.T.11 Show the procedure for generating random number X with density $f(x) = n(1-x)^{n-1}$.

Hint: You can compute the density function of $\min(U_1, \ldots, U_n)$.

III.T.12 Show two procedures of generating random number with density function given by

$$f(x) = \sum_{j=1}^{N} a_j x^j, \qquad 0 \le x \le 1,$$

where $a_j > 0$. What condition must coefficients (a_i) meet?

Hint. In one of methods apply the composition method.

III.T.13 Random variable Z has Cauchy distribution if it has density $f(z) = \frac{1}{\pi(1+z^2)}, z \in \mathbb{R}$.

(a) Let $U \sim U(0,1)$. Show that $Z = \tan(\pi(U - 1/2))$ has Cauchy distribution.

(b) Let X and Y be independent standard normal random variables. Show that $Z = X/Y$ has Cauchy distribution.

III.T.14 For a non-negative random variable Y with c.d.f. F we define a cumulative hazard rate function $H(t)$ as

$$H(t) = -\log(1 - F(t)).$$

(a) Show that if Y has a density f, then $H(t)$ can be expressed as $H(t) = \int_0^t h(s)ds$,

$$h(t) = \lim_{\triangle t \to 0^+} \frac{\mathbb{P}(t \le Y \le t + \triangle t)}{(1 - F(t))\triangle t} = \frac{f(t)}{1 - F(t)},$$

for all t being a continuity point of f. A function $h(t)$ is called a hazard rate function. It has the meaning $\mathbb{P}(t < Y \le t + \triangle t | Y > t) = h(t)\triangle t$.

(b) Show that one may sample from Y with a cumulative hazard rate in the following way:

i. Sample $X \sim$Exp(1) (standard exponential random variable)

ii. Return $Y = H^{\leftarrow}(X)$, where $H^{\leftarrow}(x) = \inf\{y : H(y) \geq x\}$.

III.T.15 Random variable X has Erl(n, λ) (Erlang) distribution if it has density

$$f(x) = \frac{1}{\Gamma(n)} \lambda^n x^{n-1} e^{-\lambda x}, x \geq 0.$$

Show that

$$\bar{F}(x) = 1 - F(x) = P(X > x) = e^{-\lambda x} \left(1 + \lambda x + \frac{(\lambda x)^2}{2!} + \ldots + \frac{(\lambda x)^{n-1}}{(n-1)!}\right).$$

III.T.16 Define the Weibull W(α, c) distribution by,

$$F(x) = \begin{cases} 0 & x < 0 \\ 1 - \exp(-cx^{\alpha}), & x \geq 0 \, . \end{cases}$$

We assume $c > 0, \alpha > 0$. Show that

$$X = \left(\frac{-\log U}{c}\right)^{1/\alpha}$$

has distribution W(α, c), where $U \sim \mathcal{U}[0, 1)$.

III.T.17 It is said that Y has a discrete Pareto distribution (DPar(α)) if it has probability function

$$p_k = \frac{1}{k^{\alpha}} - \frac{1}{(1+k)^{\alpha}}, \qquad k = 1, 2, \ldots .$$

a) Show that $Y = \lceil X \rceil$ is DPar(α), when X has Pareto Par(α) distribution.

b) Show that $Y = \lfloor U^{-1/\alpha} \rfloor$ has DPar(α) distribution.

c) Show that $\mathbb{P}(Y = k) \sim \alpha k^{-\alpha - 1}$ for $k \to \infty$.

III.T.18 Write down an algorithm and compute the acceptance probability in AR-d method for generating random number for generating random number X with probability function $(p_k, \ k = 1, 2, \ldots)$, where $p_k = \frac{90}{\pi^4} \frac{1}{k^4}$.

III.T.19 Show how to generate a random variable with a Laplace distribution (or bilateral exponential distribution)

$$f(x) = \begin{cases} e^{2x}, & -\infty < x < 0, \\ e^{-2x}, & 0 \geq x < \infty. \end{cases}$$

III.T.20 Recall that $(a, b, 0)$ is a class of probability functions on $\{0, 1, \ldots\}$ following reccurence

$$p_k = p_{k-1}\left(a + \frac{b}{k}\right), \quad k = 1, 2, \ldots$$

(a) Show that binomial, Poisson and negative binomial belong to this class.

(b) Consider similar class $(a, b, 1)$ of probability functions on $\{1, 2, \ldots\}$ fulfilling the same recurrence, but for $k = 2, 3, \ldots$. Show that truncated binomial, truncated Poisson and truncated negative binomial as well as logarithmic distribution with probability function $p_k = p^k/(-k\log(1-p)), k = 1, 2, \ldots$, belong to this class.

For a distribution p_k on $\{0, 1, \ldots\}$ we define a truncated distribution as

$$p'_k = \frac{p_k}{c}, k = 1, 2, \ldots, \quad \text{where} \quad c = \sum_{k \geq 1} p_k.$$

III.T.21 In Example 6.4 we demonstrated how to generate random number with normal distribution $\mathcal{N}(0, 1)$ by AR-c method with $g(y) = e^{-1}\mathbf{1}(y \geq 0)$. Present the version of the algorithm in which we would use distribution $\mathrm{Exp}(\lambda)$ for some $\lambda > 0$, that is $g_2(y) = \lambda e^{-\lambda y}\mathbf{1}(y \geq 0)$. For which λ is the acceptance probability the biggest?

III.T.22 Show (and justify) AR-c method for generating random number with normal distribution $X \sim \mathcal{N}(0, 1)$ with the use of Cauchy distributed Y. Show optimal c.

Can we simulate by the AR-c methods Cauchy distributed random number with the use of normally distributed Y?

III.T.23 What is the distribution of X, the output of Algorithm 25?

Algorithm 25

1: Generate Y
2: **repeat**
3: Generate $U_1 \sim \mathcal{U}[0,1)$ and set $Y = \left\lfloor -\frac{\log U_1}{\log 2} \right\rfloor$
4: Generate $U_2 \sim \mathcal{U}[0,2)$
5: **until** $U_2 \leq \frac{2^{Y+1}}{4Y!}$
6: **return** $X = Y$

III.T.24 What is the distribution of X, the output of Algorithm 26?

Algorithm 26

1: **repeat**
2: Generate Y uniformly on $\{1, 2, \ldots, n\}$.
3: Generate $U \sim \mathcal{U}[0,1)$
4: **until** $U \leq \frac{Y}{n}$
5: **return** $X = Y$

III.T.25 What is the distribution of X, the output of Algorithm 27?

Algorithm 27

1: **repeat**
2: Generate $U_1 \sim \mathcal{U}[0,1)$
3: Set

$$Y = j \quad \text{if} \quad \frac{(j-1)j}{n(n+1)} \leq U_1 \leq \frac{j(j+1)}{n(n+1)}.$$

(Y takes values $1, 2, \ldots, n$).
4: Generate $U \sim \mathcal{U}[0,1)$

5: **until** $U \cdot Y \leq 1$
6: **return** $X = Y$

III.T.26 Consider density function

$$f(x) = Cxe^{-x^4}, \qquad x \geq 0,$$

where $C = 4/\sqrt{\pi}$. Show two different algorithms for generating random numbers with this distribution.

III.T.27 Suppose that two-dimensional distribution of random vector $\mathbf{X} = (X_1, X_2)$ is given by

$$f(x_1, x_2) = ce^{-|\mathbf{x}|^4} = ce^{-(x_1^2+x_2^2)^2}, \qquad (x_1, x_2) \in \mathbb{R}^2,$$

where $c = \frac{\pi^{3/2}}{2}$.

(a) Present this density in polar coordinates (R, Θ). Show that R and Θ are independent. Think how to generate $\mathbf{X} \sim f$ using polar coordinates.

(b) Show how to apply AR-c algorithm. Do we need to know the exact value of c?

III.T.28 Let η be a random variable with distribution $\mathbb{P}(\eta = -1) = \mathbb{P}(\eta = -1) = 1/2$ which is independent from X having half-normal distribution, that is having p.d.f.

$$f(x) = \frac{2}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}, \qquad 0 < x < \infty\ .$$

Show that ηX is normally distributed $\mathcal{N}(0, 1)$.

III.T.29 Show two algorithms for generating a random point with distribution $\mathcal{U}[B]$, where $B = \{x \in \mathbb{R}^2 : |x| \leq 1\}$.

Hint. In one of the algorithms use Lemma 7.5 on polar characterization of standard normal variables.

III.T.30 Justify the following methods of generating the multinomial distribution $\mathrm{M}(n, \boldsymbol{a}), \boldsymbol{a} = (a_1, \ldots, a_n)$.

- The first one is a generalization of ad hoc methods for binomial distribution for Section 3.1 of Chapter 3. It consists in generating independent random vectors with multinomial distribution $\mathrm{M}(1, \boldsymbol{a})$ taking values $(0, \ldots, 1, \ldots 0)$ (1 is on the i-th place) with probability a_i and next summing up these vectors. Namely we draw first the component number according $\nu = \min\{j \in \{1, \ldots, k\} : \sum_{i=1}^{j} a_i \geq U\}$ to obtain $\mathbf{X}$ having 1 on the ν coordinate and 0 otherwise.

- The second method uses a unique property of multinomial $\mathbf{X} \sim \mathrm{M}(n, a_1, \ldots, a_n)$ of vector $\mathbf{X} = (X_1, \ldots, X_d)$: the distribution of X_{j+1} under condition $X_1 = (k_1, \ldots, X_j = k_j)$ is binomial $\mathrm{B}(n - k_1 - \ldots - k_j, a_{j+1}/(a_1 + \ldots + a_j))$ and X_1 has binomial distribution $\mathrm{B}(n, a_1)$.

III.T.31 (see Flury [46]) Let $\mathbf{X} \in \mathbb{R}^n$ be a random vector and $Y \in \mathbb{R}$ a random variable.

(i) If $\mathbf{X}$ has density $g(\mathbf{x})$ and for $c > 1$, random variable $(Y|\mathbf{X} = \mathbf{x})$ has distribution $U(0, cg(\mathbf{x}))$, then random vector $(\mathbf{X}, Y) \in \mathbb{R}^{n+1}$ is uniformly distributed $\mathrm{U}(\mathcal{A})$, where

$$\mathcal{A} = \{(\mathbf{x}, y) : \mathbf{x} \in \mathbb{R}^n, y \in \mathbb{R}, 0 < y < cg(\mathbf{x})\}.$$

(ii) If $(n+1)$-dimensional random vector

$$(\mathbf{X}, Y)$$

is uniformly distributed $\mathcal{U}(\mathcal{B})$ where,

$$\mathcal{B} = \{(\mathbf{x}, y) : \mathbf{x} \in \mathbb{R}^n, y \in \mathbb{R}, 0 < y < f(\mathbf{x})\},$$

then $\mathbf{X}$ has density function $f(\mathbf{x})$.

(iii) Consider how with (i) (ii) and the result from the Section 8 one can prove Proposition 6.3.

III.T.32 In actuarial mathematics one considers *Makeham distribution* of the future life time T_x of a life of age x

$$\mathbb{P}(T_x > t) = e^{-At - m(c^{t+x} - c^x)}.$$

Write a pseudocode for Makeham(A, m, c, x) $(A, m > 0, c > 1)$ for generating random numbers. For a concrete realization take $A = 5 \cdot 10^{-4}$, $m = 7.5858 \cdot 10^{-5}$ and $c = \log 1.09144$ (the so-called Danish G82 mortlaity tables for males).

Hint. This tail distribution function can be factorized $e^{-At - m(c^{t+x} - c^x)} = e^{-At} e^{-m(c^{t+x} - c^x)}$. Think how to generate a random number with tail distribution $e^{-m(c^{t+x} - c^x)}$.

III.T.33 (*) Show that Clayton copula with parameter $\theta > 0$ can be sampled in the following way: draw independent $X \sim \text{Gamma}(1/\theta, 1)$, $U_1, U_2 \sim \mathcal{U}[0,1)$ and substitute

$$(V_1, V_2) = \left(\left(1 - \frac{\log U_1}{X}\right)^{-1/\theta}, \left(1 - \frac{\log U_2}{X}\right)^{-1/\theta} \right).$$

Lab exercises

III.L.1 Suppose that W is the uniform distribution on the sphere S_2. Show that in the spherical coordinates the density is

$$f(\theta, \phi) = \frac{1}{4\pi} \sin\phi, \qquad \theta \in (0, 2\pi), \phi \in (0, \pi).$$

Let (X, Y, Z) be a random point on S_2, hence the marginal density of Θ is $1/(2\pi)$ and Φ is $\frac{1}{2}\sin\phi$. Simulate 100 such random points and plot them.

For a comparison, simulate points such that both, Θ and Φ have uniform distributions – on $(0, 2\pi)$ and $(0, \pi)$ respectively. Plot the points and comment on them.

Note: Roughly speaking, you are asked to repeat the procedure presented e.g., in Fig. 8.15, but on a sphere instead of an ellipsoid.

III.L.2 Simulate 1000 random variables via Inverse Transform Method (ITM) with Exponential and Pareto distributions (choose some parameters). Verify the implementation by plotting the histogram of generated values and overlaying the theoretical probability dendsity function. Perform some goodness-of-fit test (e.g., the Kolmogorov-Smirnov test) to compare the generated data with the theoretical exponential distribution.

III.L.3 Consider geometric distribution with success parameter $p = 1/100$. Simulate 1000 such numbers with two methods: with ITM method and directly by taking $\lfloor \log U / \log p \rfloor$ for $U \sim \mathcal{U}[0,1)$. Compare running times of the methods.

III.L.4 Implement the Acceptance-Rejection method to generate random variables following a standard normal distribution. As proposal distribution take Exp(1). Compare this method with the Box-Muller method

by analyzing the performance (time/steps taken) and the quality of the generated numbers (e.g., using chi-square test).

III.L.5 Think of an Acceptance-Rejection method to generate random variables with density provided in Example 5.2, i.e.,

$$f(x) = C\sinh(\sqrt{xc})e^{-bx}, \qquad x \geq 0$$

where $b, c > 0$ are fixed parameters and C is a normalising constant. Consider exponential distribution as a proposal distribution. Implement the method for $b = 4, c = 2$.

Recall that $\sinh(t) = (e^t - e^{-t})/2$.

III.L.6 Consider a set $\mathcal{S}_n$ of permutations of $\{1, \ldots, n\}$. By $\sigma_{\text{id}} = (1, 2, \ldots, n)$ we denote the identity permutation. For $\sigma, \sigma' \in \mathcal{S}_n$ a function

$$d(\sigma, \sigma') = \frac{1}{n}\sum_{i=1}^{n} |\sigma(i) - \sigma'(i)|$$

is a distance function (why?). Define a distribution on permutations

$$\pi(\sigma) = \frac{1}{C}\lambda^{d(\sigma,\sigma_{\text{id}})}, \quad \text{where } \lambda \in (0, 1).$$

In a sense, permutations "close" to the identity permutation are more likely. Implement a sampling procedure for $\lambda = 0.75$ and $n = 52$ using Acceptance-Rejection method with uniform proposal distribution. Simulate 1000 permutations from distribution π and estimate the acceptance probability of the AR algorithm.

Is it feasible to run the algorithm form smaller values of λ, e.g., $\lambda = 0.6$ or $\lambda = 0.5$?

Additionally, you may implement this with the so-called Kendal τ distance

$$d_{\text{Kendal}}(\sigma, \sigma') = \sum_{1 \leq i < j \leq n} \mathbf{1}\left((\sigma_i < \sigma_j \text{ and } \sigma'_i > \sigma'_j) \text{ or } (\sigma_i > \sigma_j \text{ and } \sigma'_i < \sigma'_j)\right).$$

The distance counts the number of pairwise inversions needed to transform σ to σ'

Note: Distribution π (usually with the Kendal τ distance) is often called *Mallows distribution*, often used in ranking and sorting problems.

In Exercise VII.L.2 after Chapter on the Markov chain Monte Carlo methods, the reader will be asked to implement Metropolis algorithm for this distribution, which is applicable to a wider class of parameters λ .

III.L.7 Consider the setup of Exercise III.L.6, but instead of a distance $d(\sigma, \sigma_{\mathrm{id}})$ take a number of fixed points, i.e.,

$$\pi(\sigma) = \frac{1}{C}\lambda^{\mathrm{FP}(\sigma)}, \quad \text{where } \mathrm{FP}(\sigma) = \#\{i : \sigma(i) = i\}.$$

Implement Acceptance-Rejection for $n = 52$ and $\lambda = 0.75$. Can it be applied for e.g., $\lambda = 2$? What about the efficiency then?

Later, in chapter on Markov chain Monte Carlo methods, you will be asked to implement Metropolis algorithm for this distribution, see Exercise VII.L.3.

III.L.8 Generate and plot 1000 points from 2D normal distribution with mean (0,0) and covariance matrix

$$\mathbf{\Sigma} = \begin{pmatrix} 1 & 0.6 \\ 0.6 & 2 \end{pmatrix}.$$

Use the Cholesky decomposition (you may use built-in procedure for this, but you are allowed to sample only i.i.d. standard normal random variables). Analyze the empirical correlation of the generated random vectors and compare them with the theoretical correlation.

III.L.9 Consider 2D random walk. Initially we are at (0,0). One step has a 2D normal distribution with parameters from Exercise III.L.8.

- Generate 1000 steps and plot the path of the random walk.
- Generate 1000 of such random walks and plot their final positions.
- Scale the final positions by dividing both coordinates by $\sqrt{1000}$. Compare the plotted points with points plotted in Exercise III.L.8. Verify if the distribution of the end points is consistent with the expected bivariate normal distribution.

III.L.10 Erdős–Rényi graph $G(n,p)$ is a random graph constructed in a following way: given n vertices an edge between any pair of vertices exists with probability p (independently of other edges).

- The degree of a vertex in such a graph is a random variable. It is known that the limiting distribution (as $n \to \infty$) is Poiss(λ) with $\lambda = np$. Verify this via simulations. For $n = 100$ and $p = 0.1$ simulate Erdős–Rényi graph $G(n,p)$. Repeat the whole procedure 200 times. Estimate the degree distribution (accumulated over 200 simulations) and compare it with Poisson distribution with $\lambda = np = 0.1$. Plot the observed and expected distributions for comparison and perform chi-square test (report the final p-value).
- It is known that for $p < \frac{\log n}{n}$ the graph is likely to be disconnected and that for $p > \frac{\log n}{n}$ the graph is likely to be connected (and as p increases, the probability of the graph being connected approaches 1). From 200 simulations estimate the probability of being connected. Verify if the results are in line with the theory. Note that for $n = 100$ we have $\log n/n = 0.04605$. Perform simulations for $p = 0.1$ (probability that the graph is connected should be large) and for $p = 0.01$ (the probability should be small).

Hint: You may simply store a graph as an adjecancy matrix, then sampling $G(n,p)$ is straightforward. Sample upper triangular 0-1 matrix (i.i.d., 1 with probability p), then symmetrize it. To efficiently check if a graph is connected, you may use a simple depth-first search (DFS), breadth-first search (BFS) or Union-Find algorithm (which is described in Section V.1.1, page 288).

III.L.11 Repeat the experiment from the Example 10.2, simulating independent copies $X_1, \ldots, X_{1500}$, where

- $X_1, \ldots, X_{5000}$ are i.i.d. Weibull $\mathrm{W}(2,c)$ with c chosen to have $\mathbb{E}X_1 = 1$,
- $X_{5001}, \ldots, X_{10000}$ are i.i.d. Weibull $\mathrm{W}(1/2,c)$ with c chosen to have $\mathbb{E}X_{5001} = 1$,
- $X_{10001}, \ldots, X_{15000}$, are i.i.d. with the log-normal distribution $\mathrm{LN}(0,b)$ with b chosen to have $\mathbb{E}X_{10001} = 1$.

Comment on the results.

III.L.12 Simulate $n = 1000$ 2D normally distributed points $(X_1^1, X_2^1), \ldots, (X_1^n, X_2^n)$ with mean (0,0) and covariance matrix $\mathbf{\Sigma} = \begin{pmatrix} 1 & 0.7 \\ -0.7 & 1 \end{pmatrix}$. Plot points (X_1^i, X_2^i) together with points $(U_i, U_i') = (F(X_1^i), F(X_2^i))$, where F is a c.d.f. of a standard normal $\mathcal{N}(0,1)$ distribution. In other words, (U_i, U_i') are sampled from Gaussian copula. Verify the uniformity of the sequences (U_i) and (U_i') by making histograms and applying a goodness-of-fit test. How shoud the task be modified when $\mathrm{Var} X_2 = 4$?

III.L.13 Supose that $K \subset \mathbb{R}^3$ is $K = E_1 \cup E_2$, where

$$\begin{aligned} E_1 &= \left\{(x,y,z) : \frac{x^2}{a^2} + y^2 + z^2 = 1\right\} \\ E_2 &= \left\{(x,y,z) : x^2 + y^2 + \frac{z^2}{c^2} = 1\right\} \end{aligned}$$

Choose $b = 1$ and $a = c = 5$. Estimate by a Monte Carlo method $\mathrm{Vol}\,(K)$. Analize two possible methods: crude MC and from Remark 8.19