Stochastic Simulations

Autumn Semester 2024

Prof. Fabio Nobile *Assistant: Matteo Raviola*

Lab 02 – 19 September 2024

Random Variable Generation

Exercise 1

Consider the random variable X with cumulative distribution function (CDF) $F\colon [-1,3] \to [0,1]$ given by

$$F(x) = \begin{cases} 0\ , & -1 \le x < 0\ , \\ 1 - \frac{2}{3}e^{-x/2}\ , & 0 \le x < 2\ , \\ 1\ , & 2 \le x \le 3\ . \end{cases}$$

Implement the inverse-transform method to generate n independent copies of the random variable X. Assess the quality of the realizations by comparing the empirical CDF with the theoretical CDF F for various values of n.

Exercise 2

Consider the random variable X with probability density function (PDF) f, which is only known up a multiplicative constant. Specifically, let $f(x) := k\tilde{f}(x)$ with

$$\tilde{f}(x) := \big(\sin^2(6x) + 3\cos^2(x)\sin^2(4x) + 1\big)e^{-x^2/2}\ ,$$

where the normalization constant $k = \frac{1}{\int_{\mathbb{R}} \tilde{f}(x)\,dx}$ is unknown.

1. Argue that $\tilde{f}(x)$ can be bounded by $C\phi(x)$, where C is an appropriately chosen constant and ϕ denotes the PDF of the standard normal distribution, that is $\phi(x) = e^{-x^2/2}/\sqrt{2\pi}$. Find an acceptable value for C.

2. Generate $n = 10^4$ random variables according to the PDF f using the Acceptance–Rejection Method. **Hint:** use `scipy.stats.norm.rvs()` to sample normally distributed random variables in `Python`.

3. Derive an estimate of the normalization constant k, using your procedure's acceptance probability. Compare it to the exact value $k = 0.1696542774$. Furthermore, compare the empirical CDF to the theorized, normalized CDF $F(x) = \int_{-\infty}^{x} f(u)\,du$. **Hint**: you can use the file `trueF_RVG_2.py` from the course's website to plot the theorized CDF.

Exercise 3

An element $(x, y) \in \mathbb{R}^2$ may be represented by its polar coordinates $(\rho, \Theta) \in [0, \infty) \times [0, 2\pi)$ defined by

$$\rho(x, y) = \sqrt{x^2 + y^2},$$

and

$$\Theta(x, y) = \begin{cases} \tan^{-1}\left(\frac{y}{x}\right) & \text{if } x > 0 \text{ and } y \geq 0, \\ \tan^{-1}\left(\frac{y}{x}\right) + \pi & \text{if } x < 0, \\ \tan^{-1}\left(\frac{y}{x}\right) + 2\pi & \text{if } x > 0 \text{ and } y \leq 0, \\ 0 & \text{if } x = y = 0, \end{cases}$$

where $\tan^{-1} : \mathbb{R} \to (-\pi/2, \pi/2)$.

1. Show by calculation that if the random variables $X, Y \sim N(0, 1)$ are independent, then the polar coordinate representation of (X, Y) satisfies

   $$\rho^2 \sim \exp(1/2) \quad \text{and} \quad \Theta \sim U([0, 2\pi)).$$

   Show further that ρ and Θ are independent.

2. In the opposite direction, show that if $\rho^2 \sim \exp(1/2)$ and $\Theta \sim U([0, 2\pi))$ and ρ and Θ are independent, then the Cartesian representation of the polar coordinates (ρ, Θ),

   $$X = \rho \cos(2\pi\Theta) \quad \text{and} \quad Y = \rho \sin(2\pi\Theta),$$

   satisfies $X, Y \sim N(0, 1)$ with X and Y being independent.

3. In order to construct an Acceptance–Rejection (AR) method for generating standard normal random variables consider the auxiliary PDF $g(x) = e^{-|x|}/2$. For your auxiliary PDF, determine a $C \geq 1$ such that

   $$\frac{e^{-x^2/2}}{\sqrt{2\pi}} \leq Cg(x), \qquad \forall x \in \mathbb{R}\,.$$

   Hint: See lecture notes for how to sample from the PDF g.

4. Implement the above AR method and the Box–Muller method in `Python` and use the built-in timer function `time()` within the `time` module, to compare the performance of the respective methods in terms of runtime per sample.

Exercise 4

Let $\boldsymbol{X} = (X_1, X_2, \ldots, X_n)^T \sim \mathcal{U}\big((0, 1)^n\big)$ and denote by $X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(n)}$ the ordered sample (i.e. the order statistic).

1. Implement a procedure to generate the order statistic $X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(n)}$, $n \in \mathbb{N}$, based on *sorting* a collection $\boldsymbol{X}$ of i.i.d. uniform random variables.

2. Prove the following properties:

(a) Show that

$$\mathbb{P}\big(X_{(j)} \le x\big) = \sum_{i=j}^{n} \binom{n}{i} x^i (1-x)^{n-i} \,,$$

for any $x \in (0,1)$. Furthermore, use this fact to infer the distribution of the random variable $\max\{X_1, X_2, \dots, X_n\}$.

(b) Then show that

$$\mathbb{P}\big(X_{(j)} \le z \big| X_{(k)} = x_k \;,\; \forall\, k > j\big) = \left(\frac{z}{x_{j+1}}\right)^j \,,$$

for all $z \le x_{j+1}$ and any $j < n$.

3. Use the facts above to implement a procedure that enables generating copies from the order statistic $X_{(1)} \le X_{(2)} \le \cdots \le X_{(n)}$ *without sorting*. Compare this procedures and the procedure based on sorting with respect to time for various values of n. What do you observe? Explain.
Hint: To measure time in Python, you can use

4. Implement a procedure that generates uniform random vectors in the unit simplex

$$\mathcal{S} = \Big\{(x_1, x_2, \dots, x_n)^T \in \mathbb{R}^n \colon x_i \ge 0 \;\forall\, i \;,\; \sum_{i=1}^{n} x_i \le 1\Big\} \,.$$

Assess your sampling procedure by visualizing $N = 1000$ sampling points for $n = 3$
Hint 1: You can use the following template to do a 3D scatter plot in Python.

Hint 2: Notice that a vector, whose coordinates are distributed according to the order statistic of a collection of i.i.d. $\mathcal{U}(0,1)$, takes values in the "wedge" $\mathcal{W} = \big\{(u_1, u_2, \dots, u_n)^T \in \mathbb{R}^n \colon 0 \le u_i \le 1 \;\forall\, i \;,\; u_1 \le u_2 \le \cdots \le u_n\big\}$. The unit simplex $\mathcal{S}$ is then simply the image of the "wedge" $\mathcal{W}$ under the linear transformation $\boldsymbol{x} = A\boldsymbol{u}$ where

$$A = \begin{pmatrix} 1 & 0 & \dots & 0 \\ -1 & 1 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & -1 & 1 \end{pmatrix} .$$

Exercise 5 (optional)

The PDF for the Cauchy distribution centered at $x_0 \in \mathbb{R}$ and with scale parameter $\gamma \in \mathbb{R}$ is given by

$$f(x; x_0, \gamma) = \frac{1}{\pi\gamma\left(1 + \left(\frac{x - x_0}{\gamma}\right)^2\right)} .$$

1. Show by integration of the PDF that the CDF of the Cauchy distribution with $x_0 = 0$ and $\gamma = 1$ is given by $F(x; 0, 1) = \tan^{-1}(x)/\pi + 1/2$, for $\tan^{-1} : \mathbb{R} \to (-\pi/2, \pi/2)$.

2. Show that if $X_1, X_2 \sim N(0,1)$ are independent, then $X = X_1/X_2$ is Cauchy distributed with $x_0 = 0$ and $\gamma = 1$.

3. Based on the information from the 5.1 and 5.2, describe and implement two algorithms for sampling $X \sim F(\cdot; 0, 1)$ in `Python`. Compare the performance of the respective algorithms in terms of runtime per sample.

4. It is possible to extend the preceding methods to sample from $X \sim F(\cdot; x_0, \gamma)$ for any $x_0 \in \mathbb{R}$ and $\gamma > 0$. How?

Exercise 6 (optional)

The density of a random variable X, given a sample $X_1, X_2, \ldots, X_n \overset{\text{iid}}{\sim} X$, may be approximated by a mixed distribution generated by so called kernel density estimation. In its simplest form, kernel density estimation consists of the following steps:

(i) Choose a so called kernel function $K \in \{f : \mathbb{R} \to \mathbb{R}_+ \mid \|f\|_{L^1(\mathbb{R})} = 1\}$. (So the kernel is itself a PDF.)

(ii) For some $n \in \mathbb{N}$, generate a sequence of i.i.d. random variable $X_1, X_2, \ldots, X_n$ from the distribution of X.

(iii) Define the kernel density estimator by

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} K_\delta(x - X_i),$$

where

$$K_\delta(x - X_i) := \frac{1}{\delta} K\left(\frac{x - X_i}{\delta}\right), \quad \delta > 0,$$

and δ is an appropriately chosen scaling parameter relating to the width of the kernel.

The Burr type XII distribution has the CDF

$$F(x; \alpha, c, k) = \begin{cases} 0 & x \leq 0 \\ 1 - \frac{1}{\left(1+\left(\frac{x}{\alpha}\right)^c\right)^k}, \quad x \in (0, \infty), & x > 0, \end{cases}$$

with parameters $\alpha, c, k > 0$.

1. Consider the Gaussian density kernel function

$$K(x) = \frac{e^{-x^2/2}}{\sqrt{2\pi}},$$

and implement a kernel density estimator for

$$X \sim \text{BurrXII}(\alpha = 1, c = 2, k = 4)$$

in `Python`. Samples of X can be obtained using the `scipy.stats` class `burr12`. In your code, for varying $n = [100, 10^5]$ and $\delta(n) = n^{-1/5}$, compute kernel density estimators

$$f_n(x) = \frac{1}{n} \sum_{i=1}^{n} K_{\delta(n)}(x - X_i),$$

and plot f_n and the PDF of BurrXII(1,2,4). Furthermore, for each value of n sample $N = 200000$ i.i.d. random variables $Y_i^n \sim f_n(x)$ by means of the composition method.

Hint: The `numpy.random` built-in function `randint` and might come handy.

2. Study how well the empirical CDF of $Y_1^n, Y_2^n, \ldots, Y_N^n$, which we denote by $F_n^N(x)$, converges towards $F(x; 1, 3, 1)$. That is, investigate how fast

$$D_n^N = \sup_{x \in [-2,5]} |F_n^N(x) - F(x; 1, 2, 4)|$$

decreases as n increases.