2 Numerical Distributions.

We have seen in the previous Section how to generate uniform random numbers. However in many applications one needs random numbers with other types of distribution. In general we want to produce numbers $x \in [a, b]$ with probability density $f(x)$, i.e. such that

$$\text{Prob}\,[x_1 < X < x_2] \;=\; \int_{x_1}^{x_2} f(x)\mathrm{d}x \tag{2.23}$$

There are various methods to obtain the correct distribution. The simplest one works as follows: define the distribution function $F(x)$ which is the probability that the random variable X does not exceed x:

$$F(x) \;=\; \text{Prob}\,[X < x] \;=\; \int_{a}^{x} f(x)\mathrm{d}x \tag{2.24}$$

$F(x)$ is an increasing function with $F(a) = 0$, $F(b) = 1$. If $F(x)$ is continuous and strictly increasing there exists an inverse function $F^{-1}(x)$. Then to generate a random number X with distribution $F(x)$ one simply generates a uniform random number U between 0 and 1 and computes $X = F^{-1}(U)$. To prove that this is correct notice that

$$\text{Prob}\,[X < x] \;=\; \text{Prob}\,[F^{-1}(U) < x] \;=\; \text{Prob}\,[U < F(x)] \;=\; F(x) \tag{2.25}$$

Exercise 1: Use this method to generate random numbers with density function:

1. $f(x) = px^{p-1}$ in the interval $[0, 1]$;
2. $f(x) = \mu e^{-\mu x}$ for $x \geq 0$;
3. $f(x) = 2xe^{-x^2}$ for $x \geq 0$.

Exercise 2: Develop an algorithm which computes two independent normally distributed variables X_1 and X_2.

Hint: notice that the couple (X_1, X_2) have density $f(x_1, x_2) = e^{-(x_1^2+x_2^2)/2}/2\pi$ which is easy to generate in polar coordinates.

This method can be applied in a limited number of cases since it requires the explicit computation of $F^{-1}(x)$. A more general technique is Von Neumann's rejection method. Suppose you want to generate a random variable with probability density $f(x)$ such that $F^{-1}(x)$ does not have a simple closed form. Choose then a second probability density $g(x)$ simple enough to allow a quick generation of random numbers distributed according to it and such that $f(x) \leq cg(x)$ for all x. Here c is a constant which in principle can be arbitrarily chosen as long as the bound is satisfied (thus in all cases $c > 1$). However, in order to obtain an efficient algorithm c must be as small as possible. Then the algorithm works as follows:

1. Generate X according to the density $g(x)$ and a uniform random number U between 0 and 1;

2. If $U \geq f(X)/cg(X)$ go back to step 1 and repeat with a new X and U; otherwise output X.

The probability of rejection is clearly

$$\int \left(1 - \frac{f(x)}{cg(x)}\right) g(x)\mathrm{d}x \;=\; 1 - \frac{1}{c} \tag{2.26}$$

This means that step 1 will be executed c times on average (with standard deviation $\sqrt{c(c-1)}$, prove it!) and thus a good generator requires c not to be very far from 1.

The idea of the method is very simple. If X is a random variable with distribution $g(x)$ and U is uniform between 0 and 1, then the couple (x, y) with $x = X$, $y = cUg(X)$ is uniformly distributed in the plane region $R = \{(x, y) : a \leq x \leq b, 0 \leq y \leq cg(x)\}$. Then, in order to obtain X distributed with density $f(x)$ we accept the points such that $y < f(x)$ and reject the others. The accepted points are uniformly distributed in the region $R' = \{(x, y) : a \leq x \leq b, 0 \leq y \leq f(x)\}$ and thus X is distributed with probability density $f(x)$.

Exercise 3: Use the rejection method to generate :

1. $f(x) = \sqrt{2/\pi}\; e^{-x^2}$ for $x \geq 0$;

2. $f(x) = x^{a-1}e^{-x}/\Gamma(a)$, $a > 1$ for $x \geq 0$;

In the first case use a function $g(x)$ of the form $g(x) = A$ for $0 \leq x \leq p$, $g(x) = (A/p)x \exp(p^2 - x^2)$ for $x \geq p$. In the second case choose $g(x) = Ax^{a-1}$ for $0 \leq x \leq a - 1$, $g(x) = Bx^{-p}$ for $x \geq a - 1$.

Compute the acceptance in both cases and determine the optimal value of p.

Exercise 4: A simple method to generate random unit vectors in d-dimensional space is the following:

1. Generate $U_1, \ldots, U_d$ uniform random numbers between -1 and 1 and compute $r^2 = U_1^2 + U_2^2 + \ldots + U_d^2$;

2. If $r > 1$ go back to step 1; otherwise the required vector is $(U_1/r, \ldots, U_d/r)$.

Explain why this method is correct and compute the acceptance.

Exercise 5: Suppose you want to produce random numbers x in $(-1, 1)$ with distribution

$$\frac{1}{\pi}(1 - x^2)^{-1/2}\,\mathrm{d}x \tag{2.27}$$

Show that both these methods are correct:

Method A: generate a uniform random number $U \in [0, 1]$, then compute $x \;=\; \sin \pi(2U-1)$.

Method B: generate two uniform random numbers U and V in $[0,1]$. Reject them if $U^2+V^2>1$. Otherwise compute

$$x = \frac{U^2-V^2}{U^2+V^2} \tag{2.28}$$

Method B has the advantage of requiring only elementary operations.

Exercise 6: Consider the following algorithm. Pick U_1 and U_2 uniformly in $[0,1]$. If $U_1 \leq U_2$ stop. Otherwise select U_3 and stop if $U_3 \geq U_2$. Otherwise continue this process until you have $U_1 > U_2 > \ldots U_n$ and $U_n \leq U_{n+1}$. If n is odd define $X = U_1$, while if n is even reject all U_i and start again. Show that $X \in [0,1]$ is distributed with probability density proportional to e^{-X}.

Hint: prove firstly that

$$\mathrm{Prob}\,(U_1 > U_2 \ldots > U_n) = \frac{1}{n!} \tag{2.29}$$

and

$$\mathrm{Prob}\,(x < U_1 < x + \mathrm{d}x\,|U_1 > \ldots > U_n) = nx^{n-1}\mathrm{d}x \tag{2.30}$$

Let us finally discuss the so called binning (or stratified generation) method. This technique is a simple extension of the rejection method. The idea is very simple: suppose you want to generate random numbers X with probability density $f(x)$, $a \leq x \leq b$. Then rewrite

$$f(x) = \sum_{k=1}^{n} p_k f_k(x) \tag{2.31}$$

where

$$p_k = \int_{a_k}^{a_{k+1}} f(x)\,\mathrm{d}x \tag{2.32}$$

and

$$f_k(x) = \frac{1}{p_k} f(x)\chi([a_k, a_{k+1}]) \tag{2.33}$$

Here $\chi([\alpha,\beta])$ is the characteristic function of the interval $[\alpha,\beta]$ and $a = a_1 < a_2 \ldots < a_n < a_{n+1} = b$.

Thus the generation of X can be obtained by firstly choosing k with probability p_k and then computing $X \in [a_k, a_{k+1}]$ with probability density $f_k(x)$. If the constants a_k are properly chosen the generation of $X \in [a_k, a_{k+1}]$ can be done using the rejection method with simple trial functions, in most cases a constant $g(x)$ will be efficient enough.

In order to apply this technique a fast way of choosing k with probability p_k is needed. An ingenious trick is the so-called method of aliases. Suppose you want to generate a random number X such that $X = x_0$ with probability p_0, $X = x_1$ with probability p_1, …, $X = x_{n-1}$ with probability p_{n-1}. Let us introduce three auxiliary arrays $P[k]$, $Q[k]$ and $Y[k]$, $k = 0, \ldots, n-1$. Initialize $Q[k] \leftarrow np_k$. Then define $P[k]$ and $Y[k]$ using the following algorithm:

1. Find k_1 such that $0 < Q[k_1] \leq 1$. Set $P[k_1] \leftarrow Q[k_1]$;

2. If $Q[k_1] = 1$ go to step 4;
3. Find m_1 such that $Q[m_1] > 1$. Set $Y[k_1] \leftarrow x_{m_1}$ and $Q[m_1] \leftarrow Q[m_1] + P[k_1] - 1$;
4. Set $Q[k_1] \leftarrow 0$. If $Q[k] = 0$ for all k exit.

Then the generation algorithm works as follows: choose two random numbers U and V uniformly distributed between 0 and 1. Let i be the integer part of nU. Then, if $V \leq P[i]$ set $X \leftarrow x_i$ otherwise $X \leftarrow Y_i$. It is easy to check that each x_i is generated with the correct probability.