Probability mass functions (PMFs)

Fakher Sagheer

October 7, 2022

1 Introduction

The purpose of this lab is to teach you how to generate different discrete probability distributions in different programming languages. You may work in C++, Java, Matlab or Python.

1.1 Linear congruential generator

In C++, you need to create an instance of a random number generator. Most widely studied and used random number generator is "Linear congruential generator". Another popular random number generator is "Mersenne Twister generator". I have explained only "Linear congruential generator" here. The pseudo-random sequence of number using the linear congruential generator is given by the following equation

$$x_{n+1} = (ax_n + b) \; mod \; m \tag{1}$$

where $m > 0$, $0 < a < m$, $0 \leq b < m$ and $0 \leq x_0 < m$ is the seed. In C++, we can create an instance of a LCG by creating an object of type "minstd_rand". The parameter of the constructor is the seed value.

1.2 Uniform distribution

We can create a uniform distribution (discrete) by creating an object like the following

$$uniform\_int\_distribution < int > uniform\_int\_number(m, n); \tag{2}$$

where m,n are the lower and upper limit of the discrete uniform distribution.

1.3 Geometric distribution

We can create a realization of geometric distribution by creating an object of type geometric_distribution

$$geometric\_distribution < int > geometric\_number(p); \tag{3}$$

where p is the probability of success.

1.4 Binomial distribution

We can create a realization of binomial distribution by creating an object of type binomial_distribution

$$binomial\_distribution < int > binomial\_number(n, p); \tag{4}$$

where n is the total number of trials and p is the probability of success.

1.5 Poisson distribution

We can create a realization of Poisson distribution by creating an object of type poisson_distribution

$$poisson\_distribution < int > poisson\_number(\mu); \tag{5}$$

where μ is the average number of occurrences.

2 Generating any discrete random variable

In order to generate a discrete random realization according to any pmf, we first need to generate a continuous standard uniform realization. F(x) is the CDF(Cumulative distribution function).

2.1 Continuous uniform random variable

In C++, we generate a standard continuous uniform realization using the following line of code

$$uniform\_real\_distribution < double > uniform\_real\_number(a, b); \tag{6}$$

where a and b are the lower and upper bounds of continuous uniform random variable. In this case, they should be a=0, b=1.
In order to generate any discrete random variable, we follow the following algorithm

1. For each x_i, find $F(x_i)$ i = 1 to N.
2. Generate a random continuous uniform u $\sim$ U(0,1).
3. Locate the smallest x_i where $u < F(x_i)$.
4. Set $x = x_i$.
5. Return x.

2.1.1 Weights of the packages and cost of shipping a package

Let the pmf of weights of the packages X and the shipping cost Y = g(X) defined as below

$$p_X[x] = \begin{cases} 0.15 & x = 1, 2, 3, 4 \\ 0.1 & x = 5, 6, 7, 8 \\ 0 & otherwise \end{cases} \tag{7}$$

$$Y = \begin{cases} 105X - 5X^2 & 1 \leq X \leq 5 \\ 500 & 6 \leq X \leq 10 \end{cases} \tag{8}$$

1. Write a function that outputs n samples of the shipping cost.
2. Write a function which computes the expectation and variance of Y random variable.

2.1.2 Histogram of the Poisson distribution

Generate n independent samples of the Poisson (5) random variable Y. For each $y \in S_Y$, let n(y) denote the number of times that y was observed. Thus $\sum_{y \in S_Y} n(y) =$ n, and the relative frequency of y is R(y) = n(y) / n,. Compare the relative frequency of y against $p_y[y]$ by plotting R(y) and $p_y[y]$ on the same graph as functions of y for n = 100, n, = 1000 and n = 10,000. How large should n, be to have reasonable agreement?