ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 6, No. 1, 2011, *pp*. 023-026

Generating Multi-dimensional Discrete Distribution Random Number

Xiaoping Hu [1], Hairong Cui [2]

[1] School of Economics and Management, Southeast University, Nanjing, 210096

[2] College of Engineering, Nanjing Agricultural University, Nanjing, 210032

(*Received September 12, 2010, accepted December 20, 2010*)

Abstract. Multi-dimensional discrete random number plays an important role in computer simulations. However, the existing high-level computer languages usually only give one-dimensional discrete random number, and can not appease the needs of computer simulation and experiment. Most existing studies focus on continuous distribution and one-dimensional discrete distribution. In this paper, theory and method on how to obtain random numbers subjected to an arbitrary multi-dimensional discrete probability distribution applying random numbers with uniform distribution are given.

Keywords: Multi-dimensional Discrete Distribution, Random Number Generation, Uniform Distribution, Algorithm

1. Introduction

Random number plays an important role in computer simulations, for example, one of the core issues of Monte Carlo method is to generate a string of random number sequence consistent with the actual probability distribution as the simulated data source. While many high-level computer programming languages often provide random number generation function, it is unable to generate random numbers subjected to arbitrary probability distributions, especially the multi-dimensional discrete distribution, without meeting the needs of computer simulation and experiment. Therefore, how to generate random numbers consistent with various probability distributions in the field of computer simulations must be first resolved.

There are a huge number of general as well as special methods for generating random variates from discrete or continuous distribution. An introduction on this topic can be found in Reference [1]. For more recent results see Reference [2]. In addition, Rajasekaran and Ross (1993) have presented three simple algorithms for generating a non-uniform discrete random variate with changing distributions. Orlov (2009) shows a universal optimization for generating a uniformly distributed integer in an interval, based on the underlying uniform distribution. This optimization provides up to 25% run-time improvement. A two-stage procedure about generating random samples from the multinomial distribution based on the property of the multinomial distribution is given in (2007). Wu, et al. (2002) presents an efficient algorithm for generating random numbers for the distributions with $|R|$ discrete uniform outcomes. A new algorithm of generating multivariate nonnormal distribution random numbers is given based on the Copula function in (2007). In this paper we focus on generating random samples from the multi-dimensional discrete distribution using random numbers with uniform distribution. The outline of the paper is as follows: An algorithm of generating random samples from multi-dimensional discrete distribution using random numbers with uniform distribution is shown in Section 2.1, and theoretical derivation and programming realized problem about the algorithm are given in Section 2.2 and 2.3. A numerical simulation is shown in Section 3. The conclusion is shown in Section 4.

2. Algorithm of generating random number

2.1. Algorithm

First, the two definitions are given：

Definition 1 Set E is a randomized trial, and Ω is its sample space. $X_i=X_i(\omega)$ $(\omega\in\Omega,\ i=1, 2, \dots, m)$ are one-dimensional random variables on Ω, then $(X_1, X_2, \dots, X_m)$ is called the m-dimensional random vector on

Ω. If $(X_1, X_2, \dots, X_m)$ just takes finite or countable values, then $(X_1, X_2, \dots, X_m)$ is called the m-dimensional discrete random vector.[1]

Definition 2 Set $(X_1, X_2, \dots, X_i)$ is an i-dimensional discrete random vector, then for arbitrary i-dimensional real array $(x_1, x_2, \dots, x_i)$,

$$P(x_1, x_2, \dots, x_i)=P(X_1=x_1, X_2=x_2, \dots, X_i=x_i),\ i=1, 2, \dots, m$$

is called joint probability distribution of $(x_1, x_2, \dots, x_i)$, and $P(x_1, x_2, \dots, x_i)$ (i=1, 2, … , m-1)are called the i-dimensional order marginal distribution of the m-dimensional discrete random vector $(X_1, X_2, \dots, X_m)$.

Now, we give the idea, that is, how the random number vector $r=(r(1), r(2), \dots, r(m))$ subjected to m-dimensional discrete distribution is generated by random numbers with uniform distribution. Firstly, respectively calculate the order marginal distribution $P(x_1, x_2, \dots, x_i)$ (i=1, 2, … , m-1). Secondly, sampling orderly, each sampling first produces a [0,1] uniform distribution random number, and then uses an appropriate transformation to obtain random number $r(i)$ subjected to the corresponding conditional probability distribution $P(x_i| r(1), r(2), \cdots, r(i-1))$ (i=1, 2, … , m-1). Finally, give out the random number vector $(r(1), r(2), \dots, r(m))$.

```
Input: $m$-dimensional discrete distribution $d$
Output: the random number vector $r=(r(1), r(2), \cdots, r(m))$
Begin:
Step 1. Obtain the order marginal distribution $P(x_1, x_2, \dots, x_i)$ ($i=1, 2, \dots, m-1$) of $d$;
Step 2. Define the function $ur2i(x, dd)$, i.e. transform random number $x$ from [0,1] uniform distribution U(0, 1) to random number with the probability distribution $dd$;
Step 3. carry out the following actions:
    ① Generate U(0, 1) random number $x$, and then call the function $ur2i(x, P(x_1))$ to obtain the random number $r(1)$ obeying the distribution $P(x_1)$;
    ② Calculate the conditional distribution
        $P(x_i \mid r(1),r(2),\cdots,r(i-1)) = \frac{P(r(1),r(2),\cdots,r(i-1),x_i)}{P(r(1),r(2),\cdots,r(i-1))},\quad i=2,3,\cdots,m$,
    Generate U(0, 1) random number $x$, and then call the function
    $ur2i(x, P(x_i| r(1), r(2), \cdots, r(i-1)))$
    to obtain the random number $r(i)$ obeying the conditional distribution
    $P(x_i| r(1), r(2), \cdots, r(i-1))$;
    ③ If $i=m$, end; otherwise $i=i+1$, return ②;
Step 4. Output the random number vector $r=(r(1), r(2), \cdots, r(m))$.
End
```

2.2. Theoretical derivations

The following theorem is founded:

Theorem Given $r(i) \sim P(x_i \mid r(1), r(2), \cdots, r(i-1))$, i=1, 2,…,m, then the random number vector $r=(r(1), r(2),\dots, r(m))$ obtained by the above algorithm obeys m-dimensional discrete probability distribution $d=P(x_1, x_2, \dots, x_m)$.

Proof: Suppose

$$r=(r(1),r(2),\cdots,r(m)) \sim d' = P'(x_1,x_2,\cdots,x_m).$$

Now proof

$$d' = d,$$

i.e.,

$$P'(x_1,x_2,\cdots,x_m) = P(x_1,x_2,\cdots,x_m).$$

[1] This paper only considers the case of taking finite values.

For *m*=1, obviously

$$r(1) \sim P(x_i)\text{, i.e., } P^{'}(x_1) = P(x_1)\,.$$

For *m* =2

$$\begin{aligned} P^{'}(r(1),r(2)) &= P^{'}(r(1)) \cdot P^{'}(r(2)/r(1)) \\ &= P(r(1)) \cdot P(r(2)/r(1)) \\ &= P(r(1),r(2)), \end{aligned}$$

so

$$r = (r(1),r(2)) \sim P(x_1,x_2)\text{, i.e. } P^{'}(x_1,x_2) = P(x_1,x_2)\,.$$

Suppose for *m* =*k*

$$r = (r(1),r(2),\cdots,r(k)) \sim P(x_1,x_2,\cdots,x_k)\,,$$

i.e.,

$$P^{'}(x_1,x_2,\cdots,x_k) = P(x_1,x_2,\cdots,x_k),$$

then for *m* =*k*+1

$$\begin{aligned} & P^{'}(r(1),r(2),\cdots,r(k),r(k+1)) \\ &= P^{'}(r(k+1) \mid r(1),r(2),\cdots,r(k)) \cdot P^{'}(r(1),r(2),\cdots,r(k)) \\ &= P(r(k+1) \mid r(1),r(2),\cdots,r(k)) \cdot P(r(1),r(2),\cdots,r(k)), \end{aligned}$$

i.e.,

$$P^{'}(x_1,x_2,\cdots,x_{k+1}) = P(x_1,x_2,\cdots,x_{k+1})\,.$$

So

$$\forall m \in \mathbb{Z}^+, P^{'}(x_1,x_2,\cdots,x_m) = P(x_1,x_2,\cdots,x_m)\,. \qquad \blacksquare$$

2.3. Programming realized problem

Many high-level computer programming languages restrict the use of high-dimensional array, for example C. Although some computer languages allow the use of high-dimensional array, for example Matlab, they often encounter difficulties in high-dimensional array indexed question. Therefore, according to the storage structure of high-dimensional array, transforming the high-dimensional array into one-dimensional array can solve this problem. Matlab formula as an example is given:

Let n_1, n_2, … , n_m are the length of each dimension of *m*-dimensional array *d*, and then the storage structure of any element *x* (i_1, i_2, … , i_m) of *d* in Matlab is

$$\begin{aligned} index = {} & (i_m - 1) \times n_1 \times n_2 \times \cdots \times n_{m-1} + (i_{m-1} - 1) \times n_1 \times n_2 \times \cdots \times n_{m-2} \\ & + \cdots + (i_2 - 1) \times n_1 + i_1 \end{aligned}\,.$$

So $x(i_1, i_2, \ldots, i_m) = y(index)$, where *y*(*index*) is the corresponding one-dimensional array element.

3. Simulation

Suppose a two-dimensional discrete random vector (ξ,η). Table 1 shows the two-dimensional discrete random vector (ξ,η).

Table 1 two-dimensional discrete random vector (ξ,η)

ξ \ η	1	2
1	0.1	0.2
2	0.3	0.4

The corresponding probability distribution is

$$d = \begin{pmatrix} 0.1 & 0.2 \\ 0.3 & 0.4 \end{pmatrix}.$$

Generate 100, 0000 random numbers subjected to the distribution *d* applying the above algorithm, and calculate the appearance probability p_{ij} of the sample $x=(i, j)$ $(i, j=1, 2)$

$$(p_{ij})=\begin{pmatrix}0.0999 & 0.1999\\0.2997 & 0.4005\end{pmatrix},$$

and the relative error rp_{ij},

$$(rp_{ij})=\begin{pmatrix}0.0009 & 0.0007\\0.0009 & 0.0012\end{pmatrix}.$$

Then

$$\max\_rp_{ij}=\max_{i,j}(rp_{ij})=1.2e-3,$$

$$mean\_rp_{ij}=\frac{1}{4}\sum_{i,j}rp_{ij}=9.2687e-4.$$

When generate 100, 0000 random numbers, the calculation accuracy is 10^{-4}.

4. Conclusion

This paper is aimed at giving algorithm about generating random numbers subjected to the multi-dimensional discrete probability distribution which high-level computer programming languages can not generate. The theoretical derivation, proof, and the formula of programming problem are given. When generate 100, 0000 random numbers, the calculation accuracy is 10^{-4}. It can be seen that the algorithm solves the random number generation of multi-dimensional discrete probability distribution.

5. Acknowledgements

The authors would like to thank Chang, a PhD student in Canada for his interest in the problems studied here and for his helpful comments. The author thanks also Hongsheng lv for interesting discussions and reviewers for his suggestions that allow improving the paper.

6. References

[1] L. Devroye. *Non-Uniform Random Variate Generation.* New York: Springer, 1986.

[2] C. Robert, and G. Casella. *Monte Carlo Statistical Methods* (second ed.) New York: Springer, 2004.

[3] S. Rajasekaran, and K. W. Ross. Fast Algorithms for Generating Discrete Random Variates with Changing Distributions. *ACM Transactions on Modeling and Computer Slmulation*. 1993, **3**: 1-19.

[4] M. Orlov. Optimized random number generation in an interval. *Information Processing Letters.* 2009, **109**: 722-725.

[5] S. Malefaki, and G. Iliopoulos. Short Communication: Simulating from a multinomial distribution with large number of categories. *Computational Statistics & Data Analysis*. 2007, **51**: 5471-5476.

[6] P. Wu, K. Huang, and S. Ouyang. Bit-parallel random number generation for discrete uniform distributions. *Computer Physics Communications.* 2002, **144**: 252-260.

[7] H. Xiaoping, H. Jianmin, and L. Hongsheng. Generating Multivariate Nonnormal Distribution Random Numbers Based on Copula Function. *Journal of Information and Computing Science*. 2007, **2**: 191-196.