

Original Paper

Beta-Binomially and Dirichlet-multinomially Distributed Random Number Generation Based on the Polya Urn Scheme

(Received on 31, August 1991)

by B. L. QUARSHIE, Hisakazu SHINDO and Mitsunobu KURIHARA*

Abstract

A statistical model for describing mixing states of a binary mixture was proposed by Yoshizawa and Shindo [1]. This model, based on a beta-binomial distribution, enables one to describe various variations of characteristics in space and time scales. In this paper, an algorithm for generating beta-binomially distributed random numbers is presented on the basis of the Polya urn scheme. Focusing on the ratio changes enables one to successfully eliminate an overflow problem which occurs in the scheme. Although the algorithm is not perfectly efficient, it is precise and easily understandable. An algorithm for the n-ary case, Dirichlet-multinomially distributed random number generation, is also presented as a natural extension of the binary case.

1 Introduction

A beta-binomial model proposed by Yoshizawa and Shindo [1] is a statistical description of mixed states of a binary mixture. This model was derived as a marginal distribution of the joint beta and binomial distributions. From this model, a completely segregated state (CSS) is expressed as a two-point distribution and a completely mixed state (CMS), as a binomial distribution. This is consistent with the fact that the situation (the distributions) can be described by varying only one parameter. In this paper, we examine the relationship between the Polya Urn Scheme (PUS) [2] and the beta-binomial distribution and, accordingly, present an algorithm for generating beta-binomially distributed random numbers. Although this method is not perfectly efficient, the generation algorithm is precise and easily understandable. The generation algorithm will be easily extended to the n-ary or Dirichlet-Multinomial case.

2 Descriptive Model of the Mixing State

2.1 Completely Segregated State (CSS) and Completely Mixed State (CMS)

Generally, a mixture of two kinds of particles (e.g. A and B) having the same characteristics can be treated as an infinite population. If the proportion of A and B is p' and $q' (= 1 - p')$, respectively, and a spot sample of size n from the mixture is considered, the number of particles of an observed kind (A for example) can be expressed as a random variable X , whose realized value is x . The spot sample is equivalent to an increment in mixing problems. If particles of A and B are completely separated into two distinct states, we refer to the mixture as being in a CSS. Neglecting the boundary, particles can be drawn from either A or B. Consequently, X is distributed according to the following two-point distribution:

$$\Pr[X = x] = \begin{cases} q' = 1 - p', & \text{if } x = 0 \\ p', & \text{if } x = n \\ 0, & \text{otherwise} \end{cases}$$

(1)

*Department of Electrical Engineering and Computer Science

The expected value μ_s and variance σ_s^2 of this distribution are respectively,

$$\mu_s = E[X] = np' \quad (2)$$

$$\sigma_s^2 = V[X] = n^2 p'(1-p') \quad (3)$$

In the CMS, A and B occur randomly within the mixture. In this case, the number of particles in the observed component A, in a spot sample of size n has a random variable X , which is binomially distributed as

$$\Pr[X = x] = \binom{n}{x} p'^x (1-p')^{n-x} \quad (4)$$

where

$$x = 0, 1, \dots, n.$$

The expected value μ_r and variance σ_r^2 of the binomial distribution are as follows:

$$\mu_r = E[X] = np' \quad (5)$$

$$\sigma_r^2 = V[X] = np'(1-p') \quad (6)$$

As the mixing progresses from CSS to CMS, the variance, eq. (3), decreases and becomes eq. (6). This change, as described by Lacey [3], could define the following mixing index M , i.e.,

$$M = \frac{\sigma_s^2 - \sigma^2}{\sigma_s^2 - \sigma_r^2} \quad (7)$$

where σ^2 is the value of the variance of the observed component in an arbitrarily mixed state.

2.2 Incompletely Mixed State (IMS)

For an IMS, suppose we consider the entire mixture to be partially mixed, then for a portion of the mixture we obtain a certain distribution for the random variable of the observed component. For all intents and purposes, the random variable of the observed component A, under this assumption of partially mixed state, can be approximately expressed as a binomial distribution. Even though the mixture has a strong affinity with the binomial distribution, it can be better expressed by introducing the beta distribution. This is due to the fact that it is partially (incompletely) mixed. Consequently, the partial mixture can be described with a random variable P whose probability density function $f(p)$ is,

$$f(p) = \frac{p^{\tau p'-1} (1-p)^{\tau(1-p')-1}}{B(\tau p', \tau(1-p'))} \quad (8)$$

where $0 < p < 1$, with $\tau(>0)$ and p' as the parameters of the beta density. The distribution of X , conditional on $P = p$, is considered as the following binomial distribution

$$\Pr[X = x|p] = \binom{n}{x} p^x (1-p)^{n-x} \quad (9)$$

where $x = 0, 1, \dots, n$. Therefore, the distribution of X can be expressed as a marginal distribution of the joint distribution $\Pr[X = x|p]f(p)$, i.e.,

$$\begin{aligned} \Pr[X = x] &= \int_0^1 \binom{n}{x} p^x (1-p)^{n-x} \frac{p^{\tau p'-1} (1-p)^{\tau(1-p')-1}}{B(\tau p', \tau(1-p'))} dp \\ &= \binom{n}{x} \frac{B(x + \tau p', n - x + \tau(1-p'))}{B(\tau p', \tau(1-p'))} \end{aligned} \quad (10)$$

where $x = 0, 1, \dots, n$ and $B(a, b)$ is a beta function. The expected value μ and variance σ^2 of this distribution are respectively,

$$\mu = E[X] = np' \quad (11)$$

$$\sigma^2 = V[X] = \frac{1}{\tau+1} n^2 p'(1-p') + \frac{\tau}{\tau+1} np'(1-p') \quad (12)$$

Substituting the variances, eqs. (12), (3) and (6) into eq. (7), the mixing index M is expressed as

$$M = \frac{\tau}{\tau+1} \quad (13)$$

The mixing index therefore depends on only one parameter τ . The CSS and CMS can thus be obtained if M is either 0 or 1 respectively, and the IMS is described when M varies between 0 and 1 (i.e. $0 < M < 1$).

3 Polya Urn Scheme (PUS)

The beta-binomial distribution can be obtained from the PUS and indeed can be called a Polya-Eggenberger distribution [2]. Initially, b black balls and w white balls are put in an urn. From this urn we randomly draw a ball. The ball is put back into the urn together with c balls of the same color and the process is repeated. For n trials of this process, the probability of picking x black balls is given by the following Polya distribution

$$\Pr[X = x] = \binom{n}{x} \times \frac{b(b+c)\dots[b+(x-1)c]w(w+c)\dots[w+(n-x-1)c]}{(b+w)(b+w+c)\dots[b+w+(n-1)c]} \quad (14)$$

where $x = 0, 1, \dots, n$.

The expected value μ and variance σ^2 of this distribution are respectively,

$$\mu = E[X] = \frac{nb}{b+w} \quad (15)$$

$$\sigma^2 = V[X] = \frac{nbw(b+w+nc)}{(b+w)^2(b+w+c)} \quad (16)$$

From the beta-binomial distribution expressions (given in eqs. (9), (10), and (13)) and the Polya distribution expression eq.(14), the following two relationships can be derived from the expected values and the variances of the two distributions given in eqs. (11) and (12), and eqs. (15) and (16).

$$p' = \frac{b}{b+w} \quad (17)$$

$$\tau = \frac{b+w}{c} \quad (18)$$

4 Beta-binomially Distributed Random Number Generation

The parameters that connect the beta-binomial and Polya distributions, expressed in eqs. (17) and (18), can be essentially obtained from the PUS. Thus it is possible to generate beta-binomially distributed random numbers from this model. However, if we carry out the PUS without modification on the computer, an overflow situation will result as we keep adding balls to the urn. This naturally leads us to simulate the fraction of each kind we observe in each trial. For example, if

$$p_0 : 1 - p_0 = \frac{b}{b+w} : \frac{w}{b+w} \quad (19)$$

is the initial ratio of black and white balls inside the urn, then if a black ball is drawn randomly from the urn and we return it together with c black balls into the urn, the ratio of black to white balls becomes

$$\frac{b+c}{b+w+c} : \frac{w}{b+w+c} \quad (20)$$

Using eqs. (13), (17) and (18) the ratio can be rewritten as

$$\frac{\tau p_0}{\tau+1} + \frac{1}{\tau+1} : \frac{\tau(1-p_0)}{\tau+1} = \frac{Mp_0 + (1-M)}{M + (1-M)} : \frac{M(1-p_0)}{M + (1-M)} \quad (21)$$

In general, after drawing x black balls in i trials, the ratio of black to white balls inside the urn can be expressed as follows:

$$p_i : (1 - p_i) =$$

$$\frac{b+xc}{b+w+ic} : \frac{w+(i-x)c}{b+w+ic} = \frac{Mp_0 + x(1-M)}{M+i(1-M)} : \frac{M(1-p_0) + (i-x)(1-M)}{M+i(1-M)} \quad (22)$$

After drawing another black ball, the ratio will become

$$p_{i+1} : (1 - p_{i+1}) = \frac{b+(x+1)c}{b+w+(i+1)c} : \frac{w+(i-x)c}{b+w+(i+1)c} = \frac{Mp_0 + (x+1)(1-M)}{M+(i+1)(1-M)} : \frac{M(1-p_0) + (i-x)(1-M)}{M+(i+1)(1-M)} \quad (23)$$

If this ball is a white ball, the following ratio would be obtained

$$p_{i+1} : (1 - p_{i+1}) = \frac{b+xc}{b+w+(i+1)c} : \frac{w+(i-x-1)c}{b+w+(i+1)c} = \frac{Mp_0 + x(1-M)}{M+(i+1)(1-M)} : \frac{M(1-p_0) + (i-x-1)(1-M)}{M+(i+1)(1-M)} \quad (24)$$

Putting

$$Y = \frac{M+i(1-M)}{M+(i+1)(1-M)}$$

into eqs. (23) and (24) gives the following in the $(i+1)$ -th trial in each case:

$$\text{Blackball} : p_{i+1} = p_i Y + 1 - Y \quad (25)$$

$$\text{Whiteball} : p_{i+1} = p_i Y \quad (26)$$

From these results, if the proportion p , the mixing index M of the mixture, and the spot sample size n are chosen, the number of balls of the observed component will correspond to the random number x generated by the following algorithm:

- Determine proportion p' , sample size n , mixing index M and k (the required number of random numbers)
- For $i = 1$ to k do:
 - Let $p = p'$ and $x = 0$
 - For $j = 0$ to $n-1$ do:
 - Compute

$$Y = \frac{M+j(1-M)}{M+(j+1)(1-M)}$$

- Generate a uniformly distributed random number (r)
- If ($r < p$) then

- Increase x by 1
- Compute $p = pY + 1 - Y$
- Otherwise
 - Compute $p = pY$
- Output a Beta-binomially distributed random number x .

5 Dirichlet-Multinomial Distribution

By a similar argument used to derive the beta-binomial distribution, eq. (10), a Dirichlet-multinomial distribution can be obtained. The expression below, therefore, becomes a natural consequence. That is, by first expressing the partial probability, the probability density function of the random variables $P_j (j = 0, 1, \dots, k)$ with parameters ℓ_j and $p_j (j = 0, 1, \dots, k)$ as

$$f(p_0, p_1, \dots, p_k; \ell_0, \ell_1, \dots, \ell_k) = \frac{\Gamma(\tau)}{\prod_{j=0}^k \Gamma(\ell_j)} p_0^{\ell_0-1} p_1^{\ell_1-1} \dots p_k^{\ell_k-1} \quad (27)$$

where $\sum \ell_j = \tau$ and $\sum p_j = 1$. If $\{P_j\} = \{p_j\}$ is the main condition, and $\{X_j\}$ are the random variables of the multinomial distribution, then it follows that,

$$\Pr\{\{X_j\} = \{x_j\} | \{p_j\}\} = \frac{n!}{x_0! x_1! \dots x_k!} p_0^{x_0} p_1^{x_1} \dots p_k^{x_k} \quad (28)$$

where $\sum x_j = n$. The joint marginal distribution of $\{X_j\}$ and $\{P_j\}$ is

$$\begin{aligned} \Pr\{X_0 = x_0, X_1 = x_1, \dots, X_k = x_k\} &= \\ \int_0^1 \dots \int_0^1 \frac{n!}{x_0! x_1! \dots x_k!} \prod_{j=0}^k p_j^{x_j} \times \\ \frac{\Gamma(\tau)}{\prod_{j=0}^k \Gamma(\ell_j)} \prod_{j=0}^k p_j^{\ell_j-1} dp_0 dp_1 \dots dp_k &= \\ \frac{n! \Gamma(\tau)}{\Gamma(n + \tau)} \prod_{j=0}^k \frac{\Gamma(x_j + \ell_j)}{x_j! \Gamma(\ell_j)} &= \end{aligned} \quad (29)$$

where $\tau = \sum \ell_j$ and $p_j = \frac{1}{\tau_j}$ are the relationships between the parameters. From the Dirichlet distribution eq. (29), the expected value $E[X_j]$, the variance $V[X_j]$ and the covariance $Cov[X_j, X_k]$ are expressed as follows:

$$E[X_j] = np_j \quad (30)$$

$$V[X_j] = np_j(1 - p_j) \frac{\tau + n}{\tau + 1} \quad (31)$$

$$Cov[X_j, X_k] = -np_j p_k \frac{\tau + n}{\tau + 1} \quad (32)$$

By performing similar substitutions for each variance of the mixing index, eq. (7), we obtain an expression whose variances generally correspond to covariance matrices as

$$M = \frac{|\sum_s|^{1/\tau-1} - |\sum_r|^{1/\tau-1}}{|\sum_s|^{1/\tau-1} - |\sum_r|^{1/\tau-1}} \quad (33)$$

i.e., $|\sum_s|$, $|\sum_r|$ and $|\sum|$ are the generalized variances of the CSS, CMS and arbitrary states. Computing this gives the following expression which is similar to that of the binomial case, i.e.,

$$M = \frac{\tau}{\tau + 1} \quad (34)$$

6 Multinomial Polya Urn Scheme (MPUS)

Consider an urn inside which are balls of $k+1$ kinds of color $C_j (j = 0, 1, \dots, k)$. From this urn a ball is drawn randomly. This ball together with s balls of the same color are returned into the urn and the experiment is repeated. For n trials of this experiment, suppose we draw a ball of color C_j . Let X_j be the random variable describing the number of balls drawn. Then if x_j is the realized value, the joint distribution of X_j is

$$\begin{aligned} \Pr\left[\bigcap_{j=0}^k (X_j = x_j)\right] &= \\ \binom{n}{x_0, x_1, \dots, x_k} \frac{\prod_{j=0}^k c_j(c_j + s) \dots [c_j + (x_j - 1)s]}{c(c + s)(c + 2s) \dots [c + (n - 1)s]} &= \end{aligned} \quad (35)$$

where $c = \sum c_j$, $n = \sum x_j$, $\tau = \frac{c}{s}$ and $p_j = \frac{c_j}{c}$ are the parameters of the Dirichlet-multinomial distribution eq. (29).

7 Dirichlet-Multinomially Distributed Random Number Generation

Naturally, by extending the binomial case, we can easily derive an algorithm for generating Dirichlet-multinomially distributed random numbers. Suppose in an experiment the proportion $p_j (j = 0, 1, \dots, k)$ of some color in a mixture, with mixing index M and spot sample size n , is drawn randomly. Then the number of balls drawn, c_j , will be equivalent to the random number which is generated with the following algorithm:

- Determine proportions π_j , sample size n , mixing index M , number of kinds s and number of samples k .

- For $i = 1$ to k do:

- Let $p_j = \pi_j$ and $x_j = 0$ ($j = 0, 1, 2, \dots, s-1$)
- For $j = 0$ to $n-1$ do:
 - Calculate the cumulative sum $cp_0 = 0$,
 $cp_a = \sum_{\ell=0}^a p_\ell$ ($a = 1, 2, \dots, s-1$)
 - Compute

$$Y = \frac{M + j(1-M)}{M + (j+1)(1-M)}$$

- Generate a uniformly distributed random number r
 - If $(cp_\ell \leq r < cp_{\ell+1})$ ($\ell = 0, 1, \dots, s-2$)
 - Increase x_ℓ by 1
 - Substitute $p_\ell = Y p_\ell + 1 - Y$
 - Otherwise
 - $p_a = Y p_a$ ($a \neq \ell$)
- Output Dirichlet-Multinomially distributed random numbers.

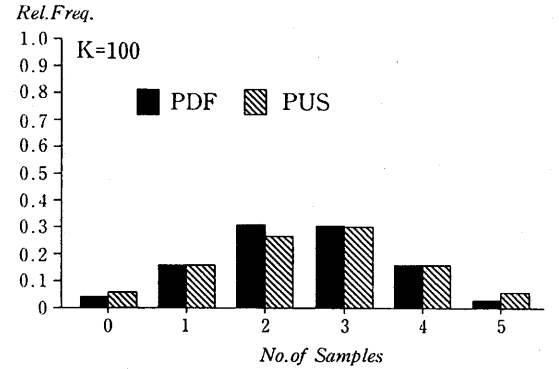
8 Illustration - Comparison of PUS Data with Theoretical Data

To see how well data generated with these algorithms agree with the beta-binomial and Dirichlet-multinomial distributions, the relative frequencies of the data were plotted as shown in Figs. 1, 2 and 3. Figure 1 shows the histograms of the beta-binomial probability function (pdf) along with data from the algorithm for a sample size $n = 5$, $p = 0.5$ and a mixing index, $M = 0.0, 0.5$ and 1.0 . Figures 2 and 3 show histograms of two of the three Dirichlet-multinomial probability functions (pdf) in an experiment with three different kinds of balls in an urn, along with data from the algorithm for a sample size $n = 5$, $p = 0.2, 0.3, 0.5$, and mixing index, $M = 0.0, 0.5$ and 1.0 . The third histogram when $p = 0.5$ is the same as in fig. 1. An examination of these graphs shows an appreciable agreement between data generated using these algorithms and expected data from theory.

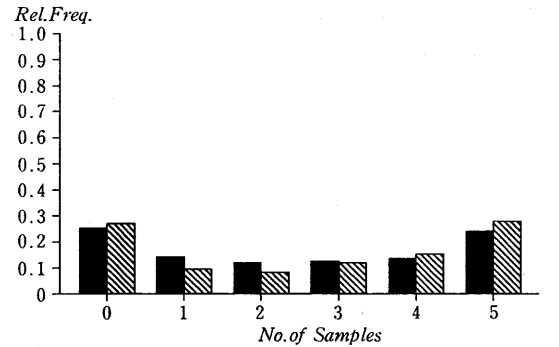
9 Conclusion

An attempt has been made in this paper to describe the mixing state of a mixture using the PUS. With this model we derived an expression for the beta-

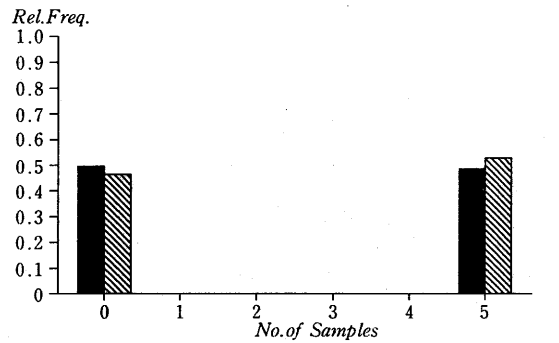
binomial distribution and proposed a method for generating random numbers. Furthermore, we showed the multinomial case to be an extension of the binomial case. We also presented an algorithm for generating Dirichlet-multinomially distributed random numbers. Though a faster generation can be obtained by the usual inverse transformation method, using recurrence between each probability, the appreciable results obtained using these algorithms, coupled with the physical meaning derived from the PUS, makes the method more practical and readily understood.



(c) $M = 1.0$ CMS



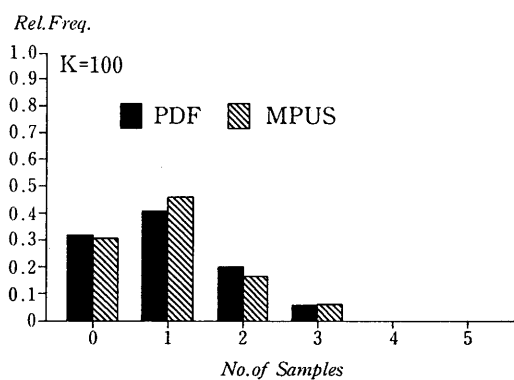
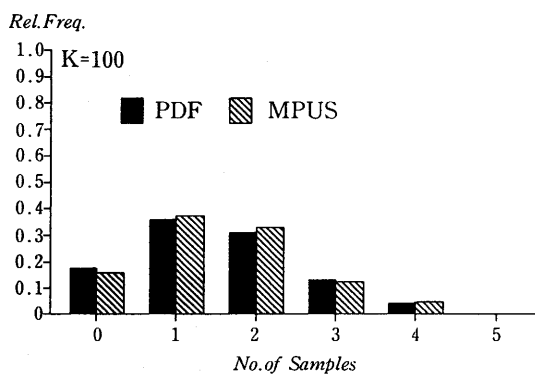
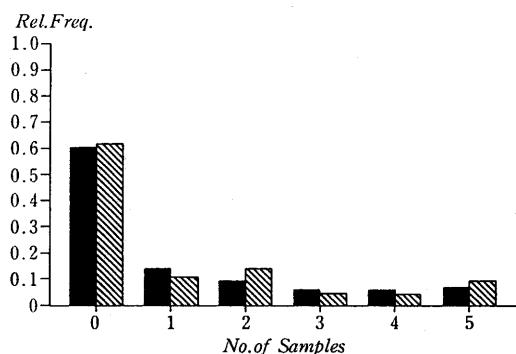
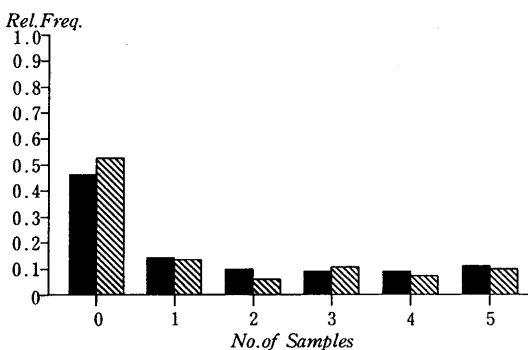
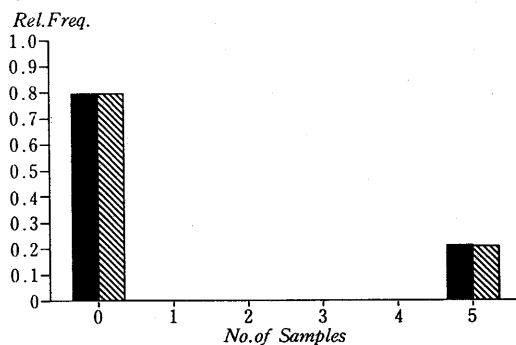
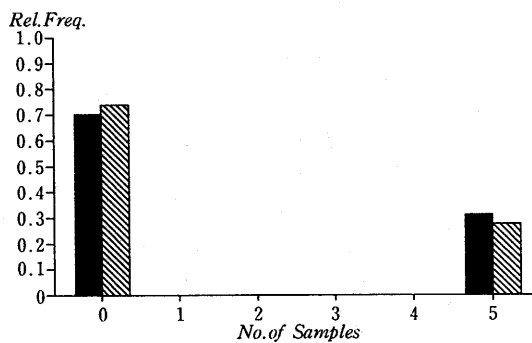
(b) $M = 0.5$ IMS



(a) $M = 0.0$ CSS

Fig.1

Comparison of PDF with PUS Data : Beta-Binomial of Distribution, $n = 5$ and $p = 0.5$

(c) $M = 1.0$ CMS(c) $M = 1.0$ CMS(b) $M = 0.5$ IMS(b) $M = 0.5$ IMS(a) $M = 0.0$ CSS(a) $M = 0.0$ CSS**Fig.2**

Comparison of PDF with MPUS Data : Dirichlet-Multinomial Distribution, $n = 5$ and $p = 0.2$

Fig.3

Comparison of PDF with MPUS Data : Dirichlet-Multinomial Distribution, $n = 5$ and $p = 0.3$

References

- [1] T. YOSHIKAWA, and H. SHINDO, *Description of the mixed state and the Beta-Binomial Distribution*. Hinshitsu (In Japanese) Vol.7, No.3, pp. 14-20 (1977).
- [2] N.L. JOHNSON, and S. KOTZ, *Urn Models and their Applications*, John Wiley and Sons Inc (1977).
- [3] P.C.M. LACEY, *Development in the theory of particle mixing*, J. Appl. Chem., 4,257 (1954).

Dirichlet-multinomially and Beta-binomially Distributed Random Number Generation Program

```
# include < stdio.h >
# define MAXS 10 /* max. no. of kinds */
# define MAXK 1000 /* max. no. of random nos. */
main()
{ int i,ii,j,jj,k,l,n,r,s;
  float
dmdrn[MAXK][MAXS],pai[MAXS],p[MAXS];
  float mix,y,rno,cp[MAXS];

  printf(Mixing index :M);/*mixing index*/
  printf(Sample size :n);/*sample size*/
  printf(Number of kinds :s);/*number of kinds*/
  printf(Necessary no. of DMD random nos.:k);
  scanf(mix,n,s,k);
  for(i=0;i<s;i++) scanf(pai[i]);/*read each pro-
portion.*/

  printf(Mixing index :M=mix);
  printf(Sample size :n=n);
  printf(Number of kinds :s=s);
  printf(Number of samples :k=k);
  for(i=0;i<s;i++) printf(i+1.pai[i]);

  for(ii=0;ii<k;ii++);/*repeat the following*/
  { for(j=0;j<s;j++) cp[j]=0;/*initialize
cum.prob.*/
    for(j=0;j<s;j++) p[j]=pai[j];/*restore each
proportion.*/
    for(j=0;j<s;j++) dmdrn[ii][j]=0;/*initialize
each ball*/
    for(jj=0;jj<n;jj++)
    { y=(mix+jj(1-mix))/(mix+(jj+1)(1-
mix));
      cp[0]=p[0];
      for(j=1;j<s;j++)
        cp[j]=cp[j-1]+p[j];/*cum. prob.*/
      cp[s-1]=1.0;
```

```
      r=rand(); rno=r/32767.0;/*generation of
uniform random number*/
      for(l=0;l<s;l++) if(rno<=cp[l])
        { dmdrn[ii][l]+1;/*increase the
corresponding ball*/
          break;/*escape this loop if matching
occurs*/
        }
      for(j=0;j<s;j++) p[j]*=y;/*change the
proportions.*/
      p[l]=1.0-y+p[l];/*adjust the correspond-
ing prop.*/
    }
    for(j=0;j<s;j++)
      printf(dmdrn[ii][j]);/*output DMD random number.
      printf(" ");
    }
}
```