Working Paper Series

On the Generation of Correlated Artificial Binary Data

Friedrich Leisch
Andreas Weingessel
Kurt Hornik

Working Paper No. 13
June 1998

Working Paper Series

June 1998

SFB
'Adaptive Information Systems and Modelling in Economics and Management Science'

Vienna University of Economics
and Business Administration
Augasse 2–6, 1090 Wien, Austria

in cooperation with
University of Vienna
Vienna University of Technology

http://www.wu-wien.ac.at/am

This piece of research was supported by the Austrian Science Foundation (FWF) under grant SFB#010 ('Adaptive Information Systems and Modelling in Economics and Management Science').

On the Generation of Correlated Artificial Binary Data

Friedrich Leisch Andreas Weingessel
Kurt Hornik

Institut für Statistik und Wahrscheinlichkeitstheorie
Technische Universität Wien
Wiedner Hauptstraße 8–10/1071
A-1040 Wien, Austria
http://www.ci.tuwien.ac.at
email: *firstname.lastname*@ci.tuwien.ac.at

1 Introduction

The generation of random variates from multivariate binary distributions has not gained as much interest in the literature as, e.g., multivariate normal or Poisson distributions (Bratley et al., 1987; Dagpunar, 1988; Devroye, 1986). Binary distributions are a special case of discrete distributions, where variables can take only two values such as "yes" and "no" or 1 and 0.

Binary variables are important in many types of applications. Our main interest is in the segmentation of marketing data, where data come from customer questionnaires with "yes/no" questions. Artificial data provide a valuable tool for the analysis of segmentation tools, because data with known structure can be constructed to mimic situations from the real world (Dolnicar et al., 1998). Questionnaire data can be highly correlated, when several questions covering the same field are likely to be answered similarly by a subject.

In this paper we present a computationally fast method to simulate multivariate binary distributions with a given correlation structure. After some remarks on multivariate binary distributions in Section 2, we present the algorithm in Section 3 and give some examples in Section 4. The implementation of the algorithm in R, an implementation of the S statistical language, is described in the appendix.

2 Multivariate Binary Distributions

In this paper we deal with variables which take only binary values, typically encoded by $\{0, 1\}$. In the simplest case the variables are scalar, then the corresponding distribution is also called the *alternative distribution*. In the following, binary random variables will be denoted by $A, B, \ldots$ or $A_1, A_2, \ldots$, respectively. Realizations of these random variables will be denoted by corresponding lower case letters. The alternative distribution is fully determined by the single value $p_A := \mathbb{P}(A = 1)$, which is also the expectation of A, i.e., $\mathbb{E}\, A = p_A$. The variance is given by $\mathrm{var}(A) = p_A(1 - p_A)$; for convenience we will write $q_A = (1 - p_A) = \mathbb{P}(A = 0)$.

Consider two binary random variables A and B which are not necessarily independent. Then the joint distribution of A and B is fully determined by p_A, p_B and either p_{AB}, $p_{A|B}$ or $p_{B|A}$

where

$$\begin{aligned} p_{AB} &:= \mathbb{P}(A=1, B=1) \\ p_{A|B} &:= \mathbb{P}(A=1|B=1) \\ p_{B|A} &:= \mathbb{P}(B=1|A=1) \end{aligned}$$

The remaining probabilities can easily be derived from Bayes Theorem

$$p_{AB} = p_{A|B}p_B = p_{B|A}p_A$$

or relationships like

$$\mathbb{P}(A=1, B=0) = p_A - p_{AB}$$

This bivariate binary distribution can easily be generalized to the multivariate case, where $A = (A_1, \ldots, A_d)' \in \{0,1\}^d$ is a vector with (possibly dependent) binary components. For a full description of an unrestricted distribution of A we need $2^d - 1$ parameters, e.g., the probabilities of all 2^d possible values of A (the last probability is determined by the condition that the sum equals 1).

3 Generation of Binary Random Variates

3.1 The Inversion Method

The task of generating a sample of size n of a binary random variable A with given p_A can be accomplished if a random number generator is available which produces random numbers U which are uniformly distributed in $[0,1]$. To get a sample $a_1, \ldots, a_n$ of A we generate a sample $u_1, \ldots, u_n$ of U and set

$$a_i = \begin{cases} 0, & u_i \leq q_A \\ 1, & u_i > q_A \end{cases}$$

The algorithm above is a special case of the inversion algorithm for discrete random variables (e.g., Devroye, 1986, p. 83ff). Let M denote an arbitrary discrete random variable taking values in $\mathbb{Z}$ with probabilities $p_i = \mathbb{P}(M = i)$. Then if we use again a uniform random variable U we can use the transformation

$$\sum_{i<M} p_i < U \leq \sum_{i \leq M} p_i$$

This relationship cannot explicitly be solved for U or M, such that for each transformation we have to perform a lookup search in the table of p_i's. To generate a random number m for M we generate a random number u for U and then set $m = j$ if

$$\sum_{i<j} p_i < u \leq \sum_{i \leq j} p_i \tag{1}$$

With this method d-dimensional binary data can be generated by specifying the probabilities of all d-tuples of $\{0,1\}$. If, for example, $d = 2$, one has to specify $\mathbb{P}(A=0, B=0), \mathbb{P}(A=0, B=1), \mathbb{P}(A=1, B=0)$, and $\mathbb{P}(A=1, B=1)$. All d-tuples can be enumerated by interpreting the binary vector as a binary number, resulting in index numbers $\{0, \ldots, 2^d - 1\}$ and corresponding probabilities $p_0, \ldots, p_{2^d-1}$. Another way of enumerating the d-tuples is by size of the p_i, resulting in computational advantages (faster computation of the lookup Equation (1) when using a sequential search algorithm).

Direct specification of all p_i is only feasible for small d, because of the exponential growth of the number of p_i's. In practice a lower-dimensional parameterization of the p_i is used such that for some function f

$$p_i = f(i, \theta), \quad \theta = (\theta_1, \ldots, \theta_n)'$$

with the dimension $n << 2^d$ of the new parameter vector θ much smaller then 2^d. If pairwise combinations of components are given by their correlation, then only $n = O(d^2)$ parameters have to specified. E.g., autologistic models fall into this category.

The inversion method is very powerful, because it can generate random values from arbitrary distributions. On the other hand, it requires a search in 2^d values for each random number generated, which makes it rather slow.

3.2 Direct Conversion of Real-Valued Random Variates

A (computationally faster) method for generating samples from a binary vector $A = (A_1, \ldots, A_d)$ is the following: Let $X = (X_1, \ldots, X_d)$ be a d-dimensional normally distributed vector with mean μ and covariance matrix Σ. Normally distributed random variates can easily be transformed to binary values by componentwise thresholding: $a_i = 1 \Longleftrightarrow x_i > 0$. Due to the construction

$$p_{A_i} = \mathbb{P}(A_i = 1) = \mathbb{P}(X_i > 0)$$

and

$$p_{A_i A_j} = \mathbb{P}(A_i = 1, A_j = 1) = \mathbb{P}(X_i > 0, X_j > 0),$$

where $\mathbb{P}(X_i > 0)$ depends, for fixed variances, only on μ_i whereas $\mathbb{P}(X_i > 0, X_j > 0)$ depends on μ_i, μ_j and on the correlation between X_i and X_j.

Efficient random number generators for normal distributions can be found in all advanced mathematic environments, hence data generation is very fast (involving only one additional comparison and assignment per value). If only random numbers from the standard normal distribution (zero mean, unit covariance matrix) can be generated, then these have to be transformed to general multivariate normals. Let Ω be the (matrix) square root of Σ such that $\Omega\Omega' = \Sigma$. Then $Y = \Omega X + \mu \sim N(\mu, \Sigma)$ if $X \sim N(0, I)$.

3.2.1 Determination of μ and Σ

Let Y_i be a 1-dimensional normally distributed random variable with mean μ_i and unit variance. Hence,

$$\mathbb{P}(Y_i > 0) = \mathbb{P}((Y_i - \mu_i) > -\mu_i) = \mathbb{P}((Y_i - \mu_i) \leq \mu_i)$$

where the second equality holds, because $(Y_i - \mu_i)$ is normally distributed with zero mean. If we choose μ_i to be the p_{A_i}-quantile of the standard normal distribution and restrict all variances to 1, then $\mathbb{P}(Y_i > 0) = p_{A_i}$. The mean vector μ is determined by the desired marginal probabilities p_{A_i} for the components of A. Allowing for different variances of the Y_i introduces additional degrees of freedom (and therefore modeling capabilities). For simplicity, we do not analyze this case in the following.

What is still missing is a relation between the covariance matrix Σ_b of the binary variables and the covariance matrix Σ of the normal distribution. By specifying a covariance matrix only pairwise relations between the components of the d-dimensional sample can be specified. In the following we will restrict ourself to the bivariate case for ease of notation.

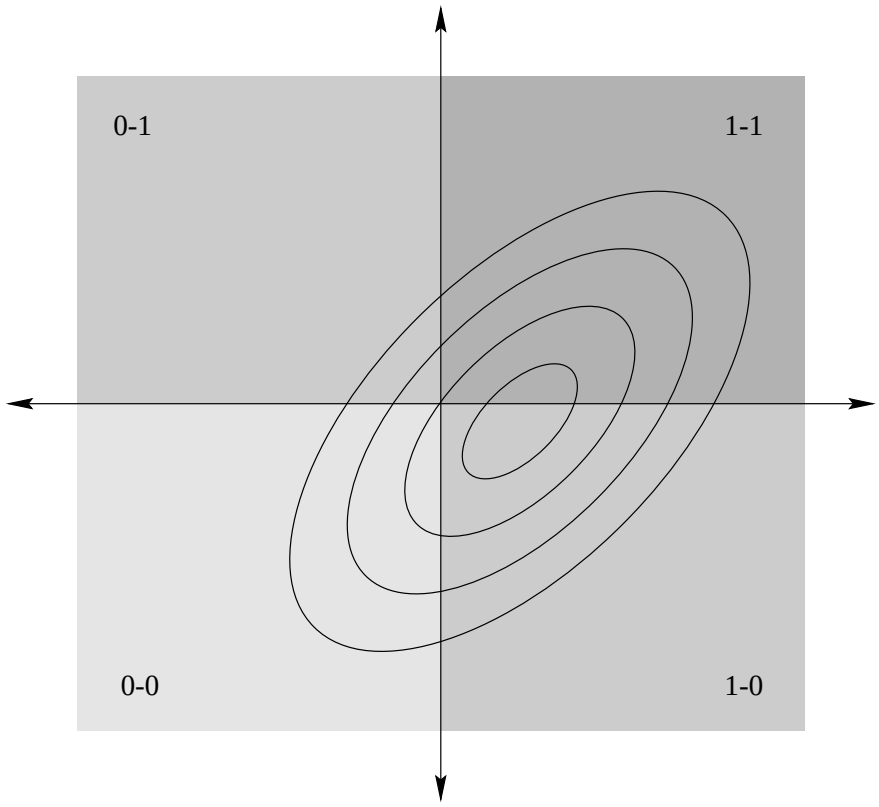

Figure 1: Conversion of bivariate normal distribution to binary.

The correlation coefficient r_{AB} of two binary random variables A and B can be written as

$$r_{AB} = \frac{p_{AB} - p_A p_B}{\sqrt{p_A q_A p_B q_B}} \tag{2}$$

such that

$$p_{AB} = r_{AB}\sqrt{p_A q_A p_B q_B} + p_A p_B. \tag{3}$$

If A and B are converted from two normal random variables X and Y as described above, then p_{AB} can be related to the normal distribution by

$$p_{AB} = \mathbb{P}(X > 0, Y > 0) = \mathbb{P}(\bar{X} > -\mu_X, \bar{Y} > -\mu_Y) = L(-\mu_X, -\mu_Y, \rho),$$

where $\bar{X} := X - \mu_X$ and $\bar{Y} := Y - \mu_Y$ have a standard bivariate normal distribution with correlation coefficient $\rho = \rho_{XY}$; and

$$L(h, k, \rho) := \mathbb{P}(\bar{X} \geq h, \bar{Y} \geq k) = \int_h^\infty \int_k^\infty \phi(x, y; \rho) dy dx$$

with

$$\phi(x, y; \rho) = \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\left(-\frac{x^2 - 2\rho x y + y^2}{2(1-\rho^2)}\right)$$

being the density function of $(\bar{X}, \bar{Y})$.

The values of $L(h, k, \rho)$ are tabulated (see the references in Patel & Read, 1982, p. 293f) or can be obtained by numerical integration or Monte Carlo simulation (see Figure 2). For $\mu_X = \mu_Y = 0$ there is the simple relation

$$p_{AB} = L(0, 0, \rho) = \frac{1}{4} + \frac{\arcsin(\rho)}{2\pi}$$

or

$$\rho = \sin\left(2\pi\left(p_{AB} - \frac{1}{4}\right)\right)$$

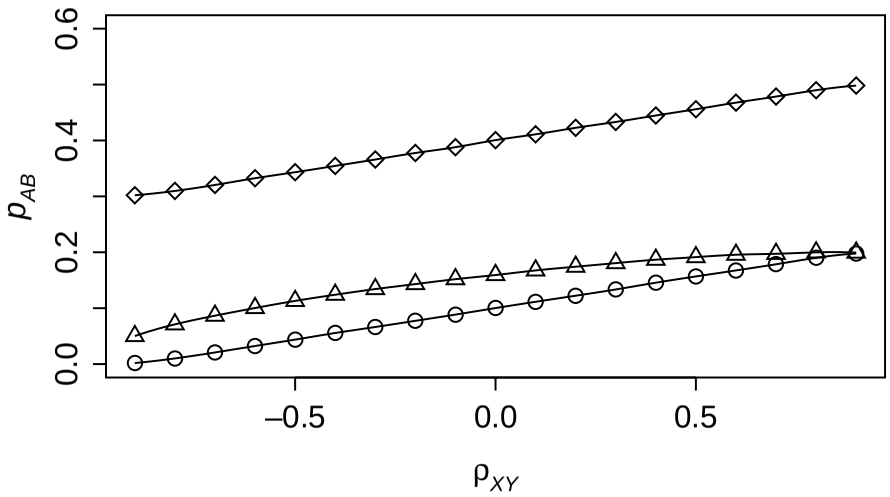

Figure 2: p_{AB} versus ρ_{XY} for three different combinations of marginal probabilities p_A/p_B: 0.2/0.5(∘), 0.2/0.8(△), 0.5/0.8(⋄). All values were derived by Monte Carlo simulation on 10^6 values and interpolated using cubic splines.

3.2.2 Pseudocode for Direct Conversion

This leads to the following algorithm for generating a d-dimensional binary vector A with given first and second moments $(p_{A_i}, p_{A_i A_j})$:

1. Choose the marginal probabilities p_{A_i} and the covariance matrix Σ_b.
2. Set $\mu_i = \Phi^{-1}(p_{A_i})$ where Φ is the standard normal distribution function.
3. Find an appropriate covariance matrix Σ for the normal distribution.
4. Generate a sample of a d-dimensional normal variables X with mean μ and covariance matrix Σ.
5. Set $a_i = 1 \iff x_i > 0$.

Note that any desired marginal probabilities for the components of A can easily be obtained by using the corresponding quantiles of the normal distribution as mean vector. The covariance matrix Σ still needs some handcrafting, as not every square matrix is positive definite and hence an admissible covariance matrix.

3.3 Specifying the Pairwise Relations

The pairwise relations between the variables $A_i, i = 1, \ldots, d$ can be either given by a covariance matrix Σ_b or by specifying the pairwise probabilities $p_{A_i A_j}, \forall i \neq j$. In both cases one has to ensure that the values specified are valid.

A covariance matrix Σ_b is valid, if it is symmetric and positive definite, that is all eigenvalues of Σ_b have to be non-negative. This property can be easily checked, but if it turns out that the matrix is not positive definite, it is not clear which elements of the matrix have to be changed in order to make the matrix positive definite.

If pairwise probabilities are specified, we are not aware of any sufficient conditions that can guarantee us the validity of the specification. We can, however, give some necessary conditions for the pairwise probabilities which will be derived in the following.

From

$$p_{AB} = p_A p_{B|A} \leq p_A$$

we get

$$p_{AB} \leq \min(p_A, p_B). \tag{4}$$

Let $p_{A \vee B}$ be the probability that at least one of A and B equals 1. Then, we get

$$1 \geq p_{A \vee B} = p_A + p_B - p_{AB}$$

which gives

$$p_{AB} \geq \max(p_A + p_B - 1, 0). \tag{5}$$

Fulfilling Conditions 4 & 5 is not sufficient as is shown by the following simple example. Let $p_A = p_B = p_C = 1/2$ and $p_{AB} = p_{AC} = p_{BC} = 0$. Then, (4) & (5) are fulfilled and one can easily construct a 2-dimensional binary distribution which is valid for p_A, p_B, and p_{AB} by specifying that the pairs $(0, 1)$ and $(1, 0)$ have a probability of $1/2$ each. However, there is no way to add the third variable C.

The reason is that

$$\begin{aligned} 1 &\geq p_{A \vee B \vee C} = p_A + p_B + p_C - p_{AB} - p_{AC} - p_{BC} + p_{ABC} \\ &\geq p_A + p_B + p_C - p_{AB} - p_{AC} - p_{BC} \end{aligned}$$

which is not fulfilled in the above example.

This above example can be generalized to d dimensions by setting $p_{A_i} = 1/(d-1), i = 1, \ldots, d$ and $p_{A_i A_j} = 0, \forall i \neq j$. Then, for any $(d-1)$ of the d variables, a distribution can be found which fulfills all pairwise conditions by giving every $(d-1)$-tuple with exactly one 1 the probability $1/(d-1)$, but there is no valid distribution for all d variables.

That means, if we want to define a d-dimensional binary distribution by specifying $p_{A_i}, i = 1, \ldots, d$ and $p_{A_i A_j}, \forall i \neq j$, we have to ensure that for all $2 \leq k \leq d$ and all possible choices $j_1, \ldots, j_k$ of k elements out of d the condition

$$1 \geq \sum_{i=1}^{k} p_{A_{j_i}} - \sum_{i,l=1, i \neq l}^{k} p_{A_{j_i} A_{j_l}}$$

is fulfilled.

4 Examples

4.1 Example 1: 3×3

We first give a short example for the generation of 3-dimensional binary data, when marginal probabilities and pairwise joint probabilities are given. In this example we will also demonstrate the usage of the R functions of the `bindata` package which is described in the appendix.

Suppose the desired marginal probabilities are

$$p_1 = \mathbb{P}(A_1 = 1) = 0.2, \quad p_2 = 0.5, \quad p_3 = 0.8$$

and the pairwise joint probabilities are

$$p_{12} = \mathbb{P}(A_1 = 1, A_2 = 1) = 0.05, \quad p_{13} = 0.15, \quad p_{23} = 0.45$$

The joint probabilities depend on the marginal probabilities, e.g., the valid range for p_{12} is [0, 0.2]. Of course one could also specify correlations instead of joint probabilities and compute the joint probabilities using Equation 3. With Equation 2 we get the correlation matrix

$$R = \begin{vmatrix} 1.0000 & -0.2500 & -0.0625 \\ -0.2500 & 1.0000 & 0.2500 \\ -0.0625 & 0.2500 & 1.0000 \end{vmatrix}$$

The mean vector of the normal distribution is given by the 0.2, 0.5 and 0.8 standard normal quantiles

$$\mu = (-0.842, 0, 0.842)$$

For $\rho \in \{-0.9, -0.8, \ldots, 0.0\}$ we compute $L(-\mu_i, -\mu_j, \rho)$ by Monte Carlo simulation and interpolate these values with cubic splines, see Figure 2. Inversion of the interpolations results in the covariance matrix

$$\Sigma = \begin{vmatrix} 1.0000 & -0.4464 & -0.1196 \\ -0.4464 & 1.0000 & 0.4442 \\ -0.1196 & 0.4442 & 1.0000 \end{vmatrix}$$

which can be derived in R by

```
sigma <- commonprob2sigma(commonprob, SimulVals)
```

where `commonprob` is the matrix with main diagonal elements p_1, p_2, p_3 and non-diagonal elements p_{12}, p_{13}, p_{23}, and `SimulVals` contains the results of the Monte Carlo simulation (R function `simul.commonprob`).

Note that the Monte Carlo simulations have to be performed *only once* for each combination of marginal probabilities and can then be stored for further usage. The `bindata` package already contains the results of such simulations in the data set `SimulVals`.

If we now generate 100.000 random binary vectors with

```
x <- rmvbin(100000, margprob = diag(commonprob), sigma=sigma)
```

or

```
x <- rmvbin(100000, commonprob = commonprob)
```

we get empirical correlations of

$$\hat{R} = \begin{vmatrix} 1.0000 & -0.2526 & -0.0602 \\ -0.2526 & 1.0000 & 0.2530 \\ -0.0602 & 0.2530 & 1.0000 \end{vmatrix}$$

and empirical probabilities

$$\begin{array}{lll} \hat{p}_1 = 0.1994 & \hat{p}_2 = 0.4998 & \hat{p}_3 = 0.8010 \\ \hat{p}_{12} = 0.0495 & \hat{p}_{13} = 0.1504 & \hat{p}_{23} = 0.4506 \end{array}$$

4.2 Example 2: 5×5

To compare our method with autologistic models we used the parameters for an autologistic model from Table 9 of Dolnicar et al. (1998). The autologistic model is given by

$$p_n = \mathbb{P}(x = x^n) = c \exp(x^{n\prime} \Theta x^n), \quad \forall x^n \in \{0,1\}^5$$

where $c = c(\Theta)$ is a normalization constant such that the p_n sum to 1, Θ is a 5×5 parameter matrix and the $x^n = (x_1^n, \ldots, x_5^n)', n = 1, \ldots, 2^5$ are all binary vectors of length 5. In this example we use the parameters

$$\Theta = \begin{vmatrix} -2.865 & 0.973 & 0.966 & 0.958 & 0.951 \\ 0.973 & -3.058 & 0.944 & 0.936 & 0.928 \\ 0.966 & 0.944 & -3.297 & 0.920 & 0.912 \\ 0.958 & 0.936 & 0.920 & -3.612 & 0.904 \\ 0.951 & 0.928 & 0.912 & 0.904 & -4.075 \end{vmatrix}$$

We first compute the pattern probabilities p_n, then the marginal probabilities that the i-th component x_i is 1, and finally the pairwise probabilities of components x_i and x_j being 1:

$$P = \begin{vmatrix} 0.854 & 0.831 & 0.827 & 0.820 & 0.806 \\ 0.831 & 0.849 & 0.823 & 0.816 & 0.803 \\ 0.827 & 0.823 & 0.842 & 0.812 & 0.799 \\ 0.820 & 0.816 & 0.812 & 0.834 & 0.794 \\ 0.806 & 0.803 & 0.799 & 0.794 & 0.818 \end{vmatrix}$$

This matrix is now converted to a covariance matrix of the 5-dimensional normal distribution as described above. Sampling 100.000 variates yields an empirical probability matrix of

$$\hat{P} = \begin{vmatrix} 0.854 & 0.828 & 0.824 & 0.818 & 0.807 \\ 0.828 & 0.850 & 0.821 & 0.816 & 0.804 \\ 0.824 & 0.821 & 0.842 & 0.812 & 0.800 \\ 0.818 & 0.816 & 0.812 & 0.834 & 0.796 \\ 0.807 & 0.804 & 0.800 & 0.796 & 0.818 \end{vmatrix}$$

The maximum absolute deviation is 0.00310 for $\mathbb{P}(x_1 = 1, x_2 = 1)$.

In Table 1 we give the pattern frequencies of 10.000 simulated data and compare them to the theoretical values of the autologistic model. We see that the autologistic model "prefers" pattern where 4 out of 5 digits are 1, their frequency is 152% to 226% as compared to the normal model. For the patterns with 2 or 3 times 1 the normal model produces them about twice as often as the autologistic model.

	00000	00001	00010	00011	00100	00101	00110	00111
Autolog	1168	20	32	3	43	5	7	5
Normal	1083	24	31	6	57	11	16	9
	01000	01001	01010	01011	01100	01101	01110	01111
Autolog	55	6	10	6	13	9	15	61
Normal	69	9	26	10	33	18	28	27
	10000	10001	10010	10011	10100	10101	10110	10111
Autolog	67	8	12	8	17	12	20	85
Normal	81	14	27	13	28	23	36	41
	11000	11001	11010	11011	11100	11101	11110	11111
Autolog	22	16	26	116	37	167	278	7652
Normal	34	32	47	61	48	110	164	7784

Table 1: Pattern Frequencies

4.3 Example 3: 10×10

As a last example we create random pattern probabilities p_n for all 10-dimensional binary vectors, giving the joint probability matrix

$$P = \begin{vmatrix} .258 & .102 & .036 & .069 & .096 & .020 & .116 & .021 & .044 & .069 \\ .102 & .406 & .074 & .099 & .142 & .054 & .183 & .055 & .072 & .107 \\ .036 & .074 & .179 & .036 & .059 & .011 & .082 & .014 & .021 & .050 \\ .069 & .099 & .036 & .255 & .090 & .023 & .113 & .022 & .036 & .072 \\ .096 & .142 & .059 & .090 & .375 & .047 & .167 & .044 & .076 & .105 \\ .020 & .054 & .011 & .023 & .047 & .131 & .060 & .008 & .015 & .026 \\ .116 & .183 & .082 & .113 & .167 & .060 & .458 & .060 & .088 & .132 \\ .021 & .055 & .014 & .022 & .044 & .008 & .060 & .133 & .014 & .029 \\ .044 & .072 & .021 & .036 & .076 & .015 & .088 & .014 & .195 & .046 \\ .069 & .107 & .050 & .072 & .105 & .026 & .132 & .029 & .046 & .288 \end{vmatrix}$$

We simulated 500.000 data points with our method resulting in the empirical probability matrix

$$\hat{P} = \begin{vmatrix} .258 & .102 & .036 & .069 & .095 & .020 & .116 & .020 & .044 & .069 \\ .102 & .406 & .074 & .099 & .141 & .054 & .183 & .055 & .072 & .106 \\ .036 & .074 & .180 & .036 & .058 & .011 & .083 & .014 & .020 & .049 \\ .069 & .099 & .036 & .255 & .089 & .023 & .113 & .022 & .036 & .072 \\ .095 & .141 & .058 & .089 & .375 & .047 & .166 & .044 & .076 & .105 \\ .020 & .054 & .011 & .023 & .047 & .131 & .060 & .008 & .015 & .025 \\ .116 & .183 & .083 & .113 & .166 & .060 & .457 & .060 & .088 & .132 \\ .020 & .055 & .014 & .022 & .044 & .008 & .060 & .132 & .014 & .029 \\ .044 & .072 & .020 & .036 & .076 & .015 & .088 & .014 & .195 & .046 \\ .069 & .106 & .049 & .072 & .105 & .025 & .132 & .029 & .046 & .289 \end{vmatrix}$$

The maximum absolute deviation is 0.00149 for $\mathbb{P}(x_7 = 1)$.

5 Summary

In this paper we presented an algorithm to generate multivariate binary distributions with a given correlation structure or with given pairwise joint probabilities. This algorithm is computationally faster than algorithms based on the inversion method. We have shown empirically that this algorithm is capable of generating data whose correlation structure is a very close approximation to the specified structure.

An open theoretical question is which combinations of pairwise joint probabilities or correlations lead to a valid multivariate binary distribution. This is an interesting problem for any algorithm which generates d-dimensional binary data out of such lower-dimensional representations of all 2^d probabilities.

Acknowledgement

The authors wish to thank Karl Grill and Pál Révész for helpful discussions.

References

Bratley, P., Fox, B. L., & Schrage, L. E. (1987). *A Guide to Simulation.* New York: Springer Verlag.

Dagpunar, J. (1988). *Principles of Random Variate Generation.* Oxford Science Publications. Oxford: Clarendon Press.

Devroye, L. (1986). *Non-Uniform Random Variate Generation.* Springer Verlag.

Dolnicar, S., Leisch, F., Weingessel, A., Buchta, C., & Dimitriadou, E. (1998). *A Comparison of Several Cluster Algorithms on Artificial Binary Data Scenarios from Tourism Marketing.* Working Paper Series 7, SFB "Adaptive Information Systems and Modeling in Economics and Management Science", http://www.wu-wien.ac.at/am.

Patel, J. K. & Read, C. B. (1982). *Handbook of the Normal Distribution*, vol. 40 of *Statistics: Textbooks and Monographs.* New York and Basel: Marcel Dekker, Inc.

A The `bindata` package for R

All simulations in this paper were performed using R, a free implementation of the S statistical language. Both R and the `bindata` package can best be obtained from the comprehensive R archive network CRAN, with the master site at

`http://www.ci.tuwien.ac.at/R`

```
check.commonprob(commonprob)
```

Arguments

`commonprob`

Matrix of pairwise probabilities.

Description

The main diagonal elements `commonprob[i,i]` are interpreted as probabilities p_{A_i} that a binary variable A_i equals 1. The off-diagonal elements `commonprob[i,j]` are the probabilites $p_{A_iA_j}$ that both A_i and A_j are 1.

This programs checks some necessary conditions on these probabilities which must be fulfilled in order that a joint distribution of the A_i with the given probabilities can exist.

The conditions checked are

$$0 \leq p_{A_i} \leq 1$$

$$\max(0, p_{A_i} + p_{A_j} - 1) \leq p_{A_iA_j} \leq \min(p_{A_i}, p_{A_j}), i \neq j$$

$$p_{A_i} + p_{A_j} + p_{A_k} - p_{A_iA_j} - p_{A_iA_k} - p_{A_jA_k} \leq 1, i \neq j, i \neq k, j \neq k$$

Value

`check.commonprob` returns `TRUE`, if all conditions are fulfilled. The attribute `"message"` of the return value contains some information on the errors that were found.

Author(s)

Andreas Weingessel

References

Friedrich Leisch, Andreas Weingessel and Kurt Hornik (1998). On the generation of correlated artificial binary data. Working Paper Series, SFB "Adaptive Information Systems and Modelling in Economics and Management Science", Vienna University of Economics, http://www.wu-wien.ac.at/am

See Also

`simul.commonprob` `commonprob2sigma`

Examples

```
check.commonprob(cbind(c(0.5, 0.4), c(0.4, 0.8)))

check.commonprob(cbind(c(0.5, 0.25), c(0.25, 0.8)))

check.commonprob(cbind(c(0.5, 0, 0), c(0, 0.5, 0), c(0, 0, 0.5)))
```

commonprob2sigma	*Calculate a Covariance Matrix for the Normal Distribution out of a Matrix of Joint Probabilites.*

```
commonprob2sigma(commonprob, simulvals)
```

Arguments

`commonprob`

Matrix of Pairwise Probabilities.

`simulvals`

Array received by `simul.commonprob`.

Description

Computes a covariance matrix for a normal distribution which corresponds to a binary distribution with marginal probabilites given by `diag(commonprob)` and pairwise probabilities given by `commonprob`.

For the simulations the values of `simulvals` are used.

If a non-valid covariance matrix is the result, the program stops with an error in the case of NA arguments and yields are warning message if the matrix is not positive definite.

Value

A covariance matrix is returned with the same dimensions as `commonprob`.

Author(s)

Friedrich Leisch

References

Friedrich Leisch, Andreas Weingessel and Kurt Hornik (1998). On the generation of correlated artificial binary data. Working Paper Series, SFB "Adaptive Information Systems and Modelling in Economics and Management Science", Vienna University of Economics, http://www.wu-wien.ac.at/am

See Also

`simul.commonprob`

Examples

```
m <- cbind(c(1/2,1/5,1/6),c(1/5,1/2,1/6),c(1/6,1/6,1/2))
sigma <- commonprob2sigma(m)
```

condprob — *Conditional Probabilities of Binary Data*

```
condprob(x)
```

Arguments

`x`	Matrix of binary data with rows corresponding to cases and columns corresponding to variables.

Description

Returns a matrix containing the conditional probabilities $P(x_i = 1 | x_j = 1)$ where x_i corresponds to the `i`-th column of `x`.

Author(s)

Friedrich Leisch

ra2ba

```
ra2ba(x)
```

Arguments

`x`	Array of arbitrary dimension

Description

Converts all values of the real valued array `x` to binary values by thresholding at 0.

Author(s)

Friedrich Leisch

Examples

```
x <- array(rnorm(10), dim=c(2,5))
ra2ba(x)
```

```
rmvbin(n, margprob, commonprob=diag(margprob),
       bincorr=diag(length(margprob)),
       sigma=diag(length(margprob)),
       colnames=NULL, simulvals=NULL)
```

Description

Creates correlated multivariate binary random variables by thresholding a normal distribution. The correlations of the components can be specified either as common probabilities, correlation matrix of the binary distribution, or covariance matrix of the normal distribution. Hence, only one of the arguments `commonprob`, `bincorr` and `sigma` may be specified. Default are uncorrelated components.

`n` samples from a multivariate normal distribution with mean and variance chosen in order to get the desired margin and common probabilities are sampled. Negative values are converted to 0, positive values to 1.

Author(s)

Friedrich Leisch

References

Friedrich Leisch, Andreas Weingessel and Kurt Hornik (1998). On the generation of correlated artificial binary data. Working Paper Series, SFB "Adaptive Information Systems and Modelling in Economics and Management Science", Vienna University of Economics, http://www.wu-wien.ac.at/am

See Also

`commonprob2sigma`,`check.commonprob`, `simul.commonprob`

Examples

```
# uncorrelated columns:
rmvbin(10, margprob=c(0.3,0.9))

# correlated columns
m <- cbind(c(1/2,1/5,1/6),c(1/5,1/2,1/6),c(1/6,1/6,1/2))
rmvbin(10,commonprob=m)

# same as the second example, but faster if the same probabilities are
# used repeatedly (coomonprob2sigma rather slow)
sigma <- commonprob2sigma(m)
rmvbin(10,marginprob=diag(m),sigma=sigma)
```

`simul.commonprob` *Simulate Joint Binary Probabilities*

```
simul.commonprob(margprob, corr=0, method="integrate", n1=10^5, n2=10)

data(CommonProb)
```

Arguments

`margprob`
: Vector of marginal probabilities.

`corr`
: Vector of correlation values for normal distribution.

`method`
: Either `"integrate"` or `"monte carlo"`.

`n1`
: Number of normal variates if method is `"monte carlo"`.

`n2`
: Number of repetitions if method is `"monte carlo"`.

Description

Compute common probabilities of binary random variates generated by thresholding normal variates at 0. The output of this function is used by `rmvbin`. For all combinations of `marginprob[i]`, `marginprob[j]` and `corr[k]`, the probability that both components of a normal random variable with mean `qnorm(marginprob[c(i,j)])` and correlation `corr[k]` are larger than zero is computed.

The probabilities are either computed by numerical integration of the multivariate normal density, or by Monte Carlo simulation.

For normal usage of `rmvbin` it is not necessary to use this function, one simulation result is provided as variable `CommonProb` in this package and loaded by default.

Author(s)

Friedrich Leisch

References

Friedrich Leisch, Andreas Weingessel and Kurt Hornik (1998). On the generation of correlated artificial binary data. Working Paper Series, SFB "Adaptive Information Systems and Modelling in Economics and Management Science", Vienna University of Economics, http://www.wu-wien.ac.at/am

See Also

`rmvbin`

Examples

```
simul.commonprob(seq(0,1,0.5), seq(-1,1,0.5), meth="mo", n1=10^4)
```