journal homepage: www.intl.elsevierhealth.com/journals/cmpb

Generating correlated discrete ordinal data using R and SAS IML

Noor Akma Ibrahim[a,*], *Suliadi Suliadi*[b]

[a] *Institute for Mathematical Research & Dept. of Mathematics, Faculty of Science, Universiti Putra Malaysia, 43400 UPM Serdang, Selangor Darul Ehsan, Malaysia*
[b] *Dept. of Statistics, Bandung Islamic University, Jl. Tamansari No. 1 Bandung 40116, Indonesia*

ARTICLE INFO

Article history:
Received 28 April 2010
Received in revised form
25 April 2011
Accepted 3 June 2011

Keywords:
Generating data
Correlated ordinal data
Simulation study
R language
SAS IML

ABSTRACT

Correlated ordinal data are common in many areas of research. The data may arise from longitudinal studies in biology, medical, or clinical fields. The prominent characteristic of these data is that the within-subject observations are correlated, whilst between-subject observations are independent. Many methods have been proposed to analyze correlated ordinal data. One way to evaluate the performance of a proposed model or the performance of small or moderate size data sets is through simulation studies. It is thus important to provide a tool for generating correlated ordinal data to be used in simulation studies. In this paper, we describe a macro program on how to generate correlated ordinal data based on R language and SAS IML.

1. Introduction

Correlated ordinal data are common in many research areas. These data may arise from longitudinal studies such as those in biology, medical, or clinical fields. The prominent characteristic of these data is that the within-subject observations are correlated, whilst between-subject observations are independent. They may also come from area-based surveys, where data from the same area are correlated while those from different areas are independent.

Miller et al. [1] extended the generalized estimating equation (GEE) of Liang and Zeger [2] and Prentice [3] to analyze correlated ordinal data. They modeled the association parameter using a second set of estimating equations for the correlation. Other related works have also been reported by Williamson et al. [4], Fahrmeir and Pritscher [5], and Parsons et al. [6].

With advances in computing technology, simulations have become a very important element in many fields of study. Simulations enable researchers to evaluate a method or model under specified conditions. One important aspect of simulations is the possibility of generating data with specific distributions. Simulations are also important in assessing methods of data analysis, especially in evaluating the characteristics of estimates of a proposed method for moderate and small size data sets. It is imperative to provide a tool that can generate such data. In this paper, we provide a macro program to generate correlated ordinal data using R language and SAS IML.

Several methods or algorithms for generating correlatedordinal data have been proposed. Gange [7] proposed an

* *Corresponding author*. Tel.: +60 38946 6873; fax: +60 38942 3789.
E-mail addresses: nakma@putra.upm.edu.my (N.A. Ibrahim), suliadi@gmail.com (S. Suliadi).
0169-2607/$ – see front matter © 2011 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/j.cmpb.2011.06.003

algorithm to generate multivariate categorical variates using iterative proportional fitting. This method is computationally intensive, even for a small number of covariates. Biswas [8] also provided an algorithm to generate correlated ordinal data. However, this algorithm is specific for autoregressive-type correlations; it also assumes that the data sets have similar or identical distributions.

A more flexible algorithm was presented by Demirtas [9], which can be used to generate any type of correlation and does not have the assumption of identical distributions. It splits the ordinal levels to binary levels (0 & 1) and proceeds with the generation of correlated binary variates. Another flexible method was reported by Lee [10]. Lee [10] used linear programming to find the solution of a joint distribution, given an association between variates. However, he did not provide any programming codes for the algorithm.

In this paper, we provide R and SAS IML macro programs to generate correlated ordinal data. These macros use the algorithm given by Lee [10]. We take advantage of the existing linear programming macro in SAS IML and the package for linear programming in R. In Section 2, we present the algorithm for generating ordinal data given by [10]. Section 3 describes the algorithm implementation into R language and SAS IML and also gives an example of the macro. This paper ends with a discussion in Section 4. The complete macro R language and SAS IML codes are given in the appendices.

2. Lee's algorithm for generating correlated ordinal data

We employed R and SAS IML macros to generate ordinal correlated data based on Lee's algorithm [10]. Lee [10] considered two types of algorithms, the convex combination method and Archimedian copulas. Here, we use the convex combination method. This method uses Goodman–Kruskal's Γ coefficient as a measurement of the association. Lee's algorithm is given as follows.

Suppose we want to generate n correlated random variables $Y_1, Y_2, \ldots, Y_n$ with Γ_{ij} as the association (correlation) between Y_i and Y_j for $i \neq j = 1, 2, \ldots, n$ and $\Gamma_{ij} = \Gamma_{ji}$. The random variable Y_i has an ordinal scale with q possible values, i.e., $Y_i \in \{1, 2, \ldots, q\}$. Lee's algorithm aims to find the joint probability of all $Y_1, Y_2, \ldots, Y_n$ for all possible level combinations. Thus, it needs to produce a $q \times q \times \cdots \times q = q^n$ table of joint probability. However, this algorithm only needs to find a $q \times q$ table of joint probability for Y_i and Y_j for all $i<j$. Using this result, it proceeds to find the joint probability of all Y_i values using linear programming.

Let $\pi_i = (\pi_{i1}, \pi_{i2}, \ldots, \pi_{iq})^T$ be the marginal probability, where $\pi_{ir} = P(Y_i = r)$ for $r = 1, 2, \ldots, q$. Because there are q levels, then, for Y_i and Y_j, we can construct a $q \times q$ table of contingency (denoted by π^{ij}) with cell $\pi_{(ir)(js)} = P(Y_i = r, Y_j = s)$ as the joint probability. Goodman–Kruskal's Γ coefficient for random samples Y_i and Y_j is defined by:

$$\Gamma_{ij} = \frac{\Pi_c - \Pi_d}{\Pi_c + \Pi_d} \tag{1}$$

where:

$$\Pi_c = 2\sum_{r<h}\sum_{s<k}\pi_{(ir)(js)}\pi_{(ih)(jk)} \text{ and } \Pi_d = 2\sum_{r<h}\sum_{s>k}\pi_{(ir)(js)}\pi_{(ih)(jk)}$$

The range of Γ is $[-1, 1]$ with zero corresponding independence. The maximal table (π^{ij}_{max}, where $\Gamma = 1$) is always composed of the cells given by (see [10]):

$$\begin{aligned}\pi_{(ir)(js)} = \; & \min(\pi_{ir}, \pi_{js}) - \min(\pi_{i[r-1]}, \pi_{js}) \\ & - \min(\pi_{ir}, \pi_{j[s-1]}) + \min(\pi_{i[r-1]}, \pi_{j[s-1]})\end{aligned} \tag{2}$$

with $\pi_{i0} = \pi_{j0} = 0$. It is straightforward to construct a table of independent correlation (π^{ij}_{ind}), given the fact that if two events are independent $\pi_{(ir)(js)} = \pi_{ir}\pi_{js}$. The minimal table ($\pi^{ij}_{min}$, where $\Gamma = -1$) is obtained as with the maximal table but by inverting the order of level either Y_i or Y_j.

Suppose that, given π_i and π_j, we have to find the π^{ij0} table, a joint distribution table of Y_i and Y_j with $\Gamma_{ij} = \Gamma^0$. We only need to find λ, $0 \leq \lambda \leq 1$, such that:

$$\pi^{ij0} = \lambda\pi^{ij}_{min} + (1 - \lambda)\pi^{ij}_{max}. \tag{3}$$

π^{ij0} exists as long as $-1 \leq \Gamma^0 \leq 1$ as well as λ. Lee's algorithm aims to construct all π^{ij0} values for $i<j = 2, 3, \ldots, n$. Thus, it needs to construct $n(n-1)/2$ tables of π^{ij0}. All π^{ij0} values will be used as restrictions in the linear programming.

The next step is to construct a q^n table of joint probability of $Y_1, Y_2, \ldots, Y_n$, given some restrictions (π^{ij0} values) that have already been obtained. To obtain the joint probability, Lee [10] proposed solving a system of linear equations. The first restriction of the linear system is that the sum of the joint probability of q^n table equals unity. The second set of restrictions is given by a known marginal probability, i.e., π_i for all $i = 1, \ldots, n$, as well as by the $n(n-1)/2$ tables of π^{ij0}. We do not need to use all of these restrictions, as some of them will be redundant. From these restrictions, we can construct a linear system in the form $A\pi^0 = x$, with $\pi^0 = (\pi^0_{11\ldots1}, \pi^0_{11\ldots2}, \ldots, \pi^0_{11\ldots q}, \ldots, \pi^0_{qq\ldots q})^T$, where $\pi^0_{ab\ldots}$ is the joint probability of $Y_1, Y_2, \ldots, Y_n$ that corresponds to $Y_{ab\ldots}$. $Y_{ab\ldots}$ is defined by $Y_1 = a$, $Y_2 = b$, …. As an example, $\pi^0_{11\ldots1} = P(Y_1 = 1, Y_2 = 1, \ldots, Y_n = 1)$ and $\pi^0_{qq\ldots q} = P(Y_1 = q, Y_2 = q, \ldots, Y_n = q)$. Note that the first restriction is $\sum\cdots\sum\pi_{ab\ldots} = 1$. For the corresponding π^0, we set $Y = (Y_{11\ldots1}, Y_{11\ldots q}, \ldots, Y_{qq\ldots q})^T$ and sort π^0 and Y based on π^0 in descending order; the results are given as $\pi^* = (\pi_{(1)}, \pi_{(2)}, \ldots, \pi_{(N)})^T$ and $W^* = (w_1, w_2, \ldots, w_N)^T$, where $N = q^n$. Please note that W^*, which is equivalent to Y, is the sample space of n random samples $Y_1, Y_2, \ldots, Y_n$. For example, w_1 may represent $Y_1 = 2$, $Y_2 = 3$, $Y_3 = q, \ldots, Y_n = 1$. The vector x is constructed based on all restrictions: (1) $\sum\cdots\sum\pi_{ab\ldots} = 1$; (2) known (given) π_i, for $i = 1, 2, \ldots, n$; and (3) π^{ij0} for $i<j = 2, 3, \ldots, n$.

The last step is to generate random variables $Y_1, Y_2, \ldots, Y_n$, by means of the inversion algorithm using the following steps [10]:

(i). Set $\pi_0 = 0$ and define $z_0, z_1, \ldots, z_{N+1}$ by $z_0 = 0; z_i = z_{i-1} + \pi_{(i-1)}$, for $i = 1, \ldots, N$; and $z_{N+1} = 1$.
(ii). Generate a random variable U from uniform distribution over [0, 1].
(iii). Return w_i if $z_i \leq U < z_{i+1}$.