Journal of Statistical Software
November 2015, Volume 68, Code Snippet 2. doi: 10.18637/jss.v068.c02

OrdNor: An R Package for Concurrent Generation of Correlated Ordinal and Normal Data

Anup Amatya
New Mexico State University

Hakan Demirtas
University of Illinois at Chicago

Abstract

In this article, operational details of the R package **OrdNor** that is designed for the concurrent generation of correlated ordinal and normal data are described, and examples of some important functions are given. The package provides needed tools that have been lacking for generating multivariate data with a mixture of ordinal and normal components.

Keywords: random number generation, polychoric correlation, phi coefficient, simulation.

1. Introduction

Scientific investigation typically involves gathering variables measured on different scales simultaneously. The data collected may be related to interdependent outcomes that are simultaneously targeted by a study, or a research project may be assessing the impact of correlated predictors or risk factors acting together to produce two or more correlated outcomes. For example, different levels of dietary intervention can affect multiple health indicators including BMI (body mass index), blood serum cholesterol level etc., which are inherently correlated. Yet another way by which correlated variables of different types can arise is when conducting longitudinal studies. In such studies, the response variable and time varying covariates are measured within a subject, which induces correlation among the observations. Sophisticated analytical methods are increasingly used to analyze correlated data (Agresti 2002). In order to evaluate and compare the methods for parameter estimation and hypotheses testing in the context of correlated data, authors often resort to Monte Carlo simulation. These simulation studies require a mechanism to generate a set of artificial data that mimic the marginal distribution and correlation structure of data sets encountered in real studies.

Major contributions in random number generation have been focused on efficient generation of samples for either univariate or multivariate distributions of one particular type. There have been several theoretical developments and software implementations in this regard (Emrich and Piedmonte 1991; Lee 1993; Park, Park, and Shin 1996; Gange 1995; Oman and Zucker 2001; Qaqish 2003; Biswas 2004; Demirtas 2006; Yahav and Shmueli 2012; Ferrari and Barbiero 2012). However, there has been only sporadic interest in generation of multivariate data with mixed data types. Examples of such efforts are found in Demirtas and Doganay (2012) for generating mixture of correlated binary and normal variables, and in Ruscio and Kaczetow (2008) for generating multivariate non-normal data via a sample and iterate (SI) algorithm. We further the effort in mixed data generation by elaborating on an easy-to-use R package that provides functions for concurrent generation of ordinal and normal variables with a given marginal distribution and correlation structure.

The organization of the rest of the article is as follows. In Section 2, we outline the algorithm implemented for the simultaneous generation of correlated ordinal and normal variables. In Section 3, we describe the operational details of the R package **OrdNor**. In Section 4, we present an example and evaluate the performance of the package. In Section 5, we present a run-time comparison between proposed and alternative algorithm. We conclude the paper with a discussion in Section 6.

2. Algorithm

2.1. Prerequisites

Multivariate normal and multivariate ordinal data generation with underlying normal distribution is well-studied. When it comes to joint generation of normal and ordinal variables, the key is to establish a connection between point-polyserial and polyserial correlations. Once such relationship is established, one can concurrently generate a set of ordinal and normal variables in a unified manner given marginal proportions and a set of correlations. Specific algorithmic steps are given below.

Let $Y_1, Y_2, \ldots, Y_j$ be a set of ordinal variables with proportion vectors $p_1, p_2, \ldots, p_j$ and $W_l \sim N(\mu_l, \sigma_l^2)$, where $l = 1, 2, \ldots, k$. The $(j+k) \times (j+k)$ correlation matrix is Σ. Without loss of generality, assume that variables are in a certain order, i.e., the data consist of $Y_1, \ldots, Y_j$, $W_1, \ldots, W_k$. Then, Σ is comprised of three components: Σ_{OO}, Σ_{ON}, and Σ_{NN}, where O and N correspond to ordinal and normal, respectively. In this setup, Σ_{OO} is a $j \times j$ submatrix and Σ_{NN} is a $k \times k$ submatrix of Σ that stand for the correlations between the ordinal-ordinal and normal-normal combinations, respectively. Similarly, Σ_{ON} represents a $j \times k$ submatrix whose elements are the correlations between ordinal and normal variables. Required parameter values are either specified or estimated from a real data set that is to be mimicked.

2.2. Algorithm description

1. Work with centered and scaled versions of W variables, $Z_l = (W_l - \mu_l)/\sigma_l \sim N(0,1)$, $l = 1, 2, \ldots, k$. Dealing with standard normals is easier.

2. Check if Σ is positive definite. With a linear transformation in Step 1, correlations will remain unchanged.

3. Check if the elements of Σ_{OO} are within limits. The lower and upper bounds of valid correlation values are found by the generate, sort, and correlate (GSC) algorithm (Demirtas

and Hedeker 2011) whose steps are given below.

   (a) Generate random samples from the intended distributions independently using a large number of observations (e.g., $N = 100,000$). Here, the intended distributions are two univariate multinomial distributions.

   (b) To estimate the lower bound, sort the two variables in opposite directions (i.e., from smallest to largest for one of the variables, and from largest to smallest for the other). Then, compute the sample correlation.

   (c) To estimate the upper bound, sort the two variables in the same direction, and compute the sample correlation.

4. Check if the elements of Σ_{ON} are within limits using a GSC algorithm. Here, the intended distributions are multinomial and standard normal distributions.

5. Compute the polychoric correlations for the OO combinations using an iterative procedure as described by Ferrari and Barbiero (2012). The steps involved in the computation are as follows.

   (a) Set $\delta^+_{X_O,Y_O} = \delta_{X_O,Y_O}$ where δ_{X_O,Y_O} is the target value of a pairwise correlation between two ordinal variables.

   (b) Generate X and Y from the bivariate normal distribution $N_2(\mathbf{0}, R_{OO})$ with a large number of data points, where

$$R_{OO} = \begin{pmatrix} 1 & \delta^+_{X_O,Y_O} \\ \delta^+_{X_O,Y_O} & 1 \end{pmatrix}.$$

   (c) Transform X and Y into ordinal variable X^*_O and Y^*_O by discretization based on the target marginal probabilities.

   (d) Compute $\delta^*_{X_O,Y_O}$ as sample correlation between X^*_O and Y^*_O.

   (e) Iteration: If $(\delta^*_{X_O,Y_O} - \delta_{X_O,Y_O}) > \epsilon$, update R_{OO} such that $\delta^+_{X_O,Y_O} = \delta^+_{X_O,Y_O} \times \frac{\delta_{X_O,Y_O}}{\delta^*_{X_O,Y_O}}$ and go back to Step 5b. Otherwise, $\delta^+_{X_O,Y_O}$ is the intermediate correlation.

6. Repeat Step 5 for each pair of ordinal variables.

7. Construct a correlation matrix Σ^+_{OO} for all ordinal components of the multivariate distribution. Σ^+_{OO} is a symmetric submatrix whose diagonal elements are 1.

8. Compute the polyserial correlations for the ON combinations as follows.

   (a) Generate X and Y independently with a large number of data points, where X and Y are independent standard normal variables.

   (b) Transform X into an ordinal variable X_O by discretization based on the target marginal probabilities.

   (c) Estimate the upper bound for correlation $\max(\mathsf{COR}(X_O, Y))$ using GSC; that is sort the two variables in the same direction, and compute the sample correlation.

   (d) Compute $\hat{c}$ via $\hat{c} = \max(\mathsf{COR}(X_O, Y))/\max(\mathsf{COR}(X, Y))$. As $\max(\mathsf{COR}(X, Y)) = 1$, $\hat{c} = \max(\mathsf{COR}(X_O, Y))$.

   (e) Find $\delta_{X,Y}$ by $\delta_{X_O,Y}/\delta_{X,Y} = \hat{c}$, where $\delta_{X_O,Y}$ is the target value of the pairwise correlation between an ordinal and a normal variable. $\delta_{X,Y}$ is an intermediate pairwise correlation between two standard normal variables, such that ordinalization of one variable of the pair is expected to result in the target value of pairwise correlation of an ordinal-normal pair.

9. Repeat Step 8 for each pair of ordinal and normal variables.

10. Construct a correlation submatrix Σ^+_{ON} for all combinations of ordinal and normal components of the multivariate distribution. This is a submatrix of the overall correlation matrix, and it is pertinent to the ordinal-normal part. Hence, the matrix may or may not be square.

11. Construct an overall correlation matrix, Σ^+, using Σ_{NN}, and the results from Steps 5–10.

12. Check if Σ^+ is positive definite. If it is not, compute the nearest positive definite correlation matrix.

13. Generate multivariate normal data with a mean vector of $(0, \ldots, 0)_{k+j}$ and correlation matrix of Σ^+.

14. Obtain ordinal variables $Y_1, Y_2, \ldots, Y_j$, by the thresholds determined by marginal proportion vectors, $p_1, p_2, \ldots, p_j$ using quantiles of the normal distribution.

15. Go back to the original scale for normal variables by reverse centering and scaling, $W_l = Z_l \times \sigma_l + \mu_l$, $l = 1, 2, \ldots, k$.

3. The OrdNor package

The **OrdNor** package (Amatya and Demirtas 2015) provides functions for concurrently generating correlated ordinal and normal data as described above. The package is available via the Comprehensive R Archive Network (CRAN) at `http://CRAN.R-project.org/package=OrdNor`. Once the package has been installed, the results given in this paper are reproducible.

The package contains a data generating function `genOrdNor` that generates the data set with mixed ordinal and normal variables with pre-specified marginal distribution and correlation structure. The data generating function is supported by three core functions `IntermediateON`, `IntermediateOO`, and `cmat.star`. The remaining three auxiliary functions `valid.limits`, `validate.target.cormat`, and `validate.plist` provide necessary protection against erroneous user inputs. The `valid.limits` function is a wrapper that encompasses the functions `Limit_forON` and `Limit_forOO`. Throughout the article, we use the following substitutes given in Table 1.

The following is a summary of the functions contained in the package. Their functionality, in the context of correlated ordinal and normal data generation, is described in the next three subsections.

`genOrdNor`: Generates a data set with ordinal and normal variables.

Label	Meaning
`plist`	A list of cumulative probability vectors corresponding to each ordinal variable.
`CorrMat`	A positive definite target correlation matrix whose entries are within the valid limits.
`OOCorrMat`	A submatrix of `CorrMat` corresponding to pairwise ordinal variables.
`ONCorrMat`	A submatrix of `CorrMat` corresponding to ordinal-normal pairs.
`mean.vec`	A vector of mean values for normal variables.
`sd.vec`	A vector of standard deviation values for normal variables.
`n`	Number of rows to be generated.
`no.ord`	Number of ordinal variables.
`no.norm`	Number of normal variables.

Table 1: Terms used as substitutes for the definition of parameters.

`IntermediateON`**:** Computes intermediate correlations for O-N pairs before ordinalization.

`IntermediateOO`**:** Computes intermediate correlations for O-O pairs before ordinalization.

`cmat.star`**:** Computes the correlation of intermediate MVN data.

`Limit_forON`**:** Finds the feasible correlation range for a pair of ordinal and normal variables.

`Limit_forOO`**:** Finds the feasible correlation range for a pair of ordinal variables.

`ordinalize`**:** Ordinalizes the standard normal variable.

`validate.target.cormat`**:** Checks the target correlation matrix.

`valid.limits`**:** Computes the lower and upper bounds of correlation in the form of two matrices.

3.1. Data generating function

The `genOrdNor` function generates the mixed multivariate data with given marginal probabilities and correlations (Steps 13 to 15). It takes `n`, `plist`, `cmat.star`, `mean.vec`, `sd.vec`, `no.ord` and `no.norm` as input arguments. The `validate.target.cormat` function is called to insure input arguments are correctly specified. Next, the `cmat.star` function is called and its output is used to generate the intermediate multivariate normal data. It is accomplished through the `rmvnorm` function from the **mvtnorm** package (Genz, Bretz, Miwa, Mi, Leisch, Scheipl, and Hothorn 2015; Genz and Bretz 2009). Finally, the `ordinalize` function transforms the first `no.ord` components of the multivariate normal data to multivariate ordinal data (Step 14). The remaining `no.norm` components are scaled and shifted to match the marginal distributions of the normal variables.

3.2. Core functions

The `IntermediateOO` function computes the intermediate values of pairwise correlations between ordinal-ordinal pairs. It takes `plist` and `OOCorrMat` as the input arguments, calls the `ordinalize` function, and returns the intermediate matrix of pairwise correlations between ordinal variables as output. The `OOCorrMat` is a symmetric submatrix of the overall correlation matrix pertinent to the ordinal-ordinal part.

The following example shows the use of `IntermediateOO` for computing intermediate values of pairwise correlations between three ordinal components of a seven variables system as described in Step 6.

```
R> set.seed(1000)
R> marginal <- list(0.3, cumsum(c(0.30, 0.40)), cumsum(c(0.4, 0.2, 0.3)))
R> OOCorrMat <- matrix(c(1.0000000, 0.1767231, 0.3006186, 0.1767231,
+    1.0000000, -0.1399237, 0.3006186, -0.1399237, 1.0000000), 3, 3)
R> IntermediateOO(marginal, OOCorrMat)

          [,1]       [,2]       [,3]
[1,] 1.0000000  0.2579297  0.4421591
[2,] 0.2579297  1.0000000 -0.1712243
[3,] 0.4421591 -0.1712243  1.0000000
```

The `IntermediateON` function computes the intermediate values of pairwise correlations between ordinal and normal variables. It takes `plist` and `ONCorrMat` as the input arguments, calls the `ordinalize` function, and returns the intermediate matrix of pairwise correlations between ordinal and normal variables as output. As `ONCorrMat` is a submatrix of the overall correlation matrix pertinent to the ordinal-normal part, it may or may not be square. Even when it is square, it may not be symmetric.

```
R> set.seed(1000)
R> marginal <- list(0.3, cumsum(c(0.30, 0.40)), cumsum(c(0.4, 0.2, 0.3)))
R> ONCorrMat = matrix(c(0.20878586, 0.007379469, -0.27810121, 0.30889965,
+    0.182164203, 0.18854796, -0.03481523, 0.391789550, -0.39909073,
+    -0.26690257, -0.372371652, -0.08703733), 4, 3)
R> IntermediateON(marginal, ONCorrMat)

             [,1]        [,2]        [,3]
[1,]  0.275663000  0.20286867 -0.43959233
[2,]  0.009688273  0.21002325 -0.29349050
[3,] -0.366716693 -0.03877934 -0.41007599
[4,]  0.407009283  0.43626976 -0.09584821
```

In the illustration above, parameters of three ordinal variables with two, three, and four categories, and target values of pairwise correlations between these three ordinal and four normal variables are specified. The function utilizes these values to determine intermediate values of corresponding pairwise correlations as described in Step 8 of the algorithm.

The `cmat.star` function wraps the two functions above, takes `CorrMat`, `plist`, `no.ord` and `no.norm` as input arguments and returns the complete matrix of the intermediate pairwise correlations, i.e., Σ^+. The function utilizes the `ordcont` function from the **GenOrd** package (Barbiero and Ferrari 2015), the `nearPD` function from the **Matrix** package (Bates and Maechler 2015) and the `is.positive.definite` function from the **corpcor** package (Schäfer, Opgen-Rhein, Zuber, Ahdesmäki, Duarte Silva, and Strimmer 2015) to accomplish various operations. In case Σ^+ is non-positive definite, it returns the positive definite matrix nearest to the Σ^+ as described in Step 12.

3.3. Auxiliary functions

Trivial specification problems are captured by three validation functions. These are auxiliary functions that are called by the core functions. The core functions expect marginal probabilities of ordinal variables to be specified in the form of cumulative probabilities. The consequence of specifying cumulative probabilities is that the cumulative probability of the last category is not explicitly provided but is implied, i.e., 1. If there are any skipped categories (i.e., a category with 0 probability), the consecutive entries in the corresponding probability vectors will be the same. The `validate.plist` function checks whether `plist` meets these requirements. Further, it checks the entries of each component vector of the list and makes sure that the values are within the plausible range (0–1). Finally, it checks the conformity of the number of components in the list with the specified number of ordinal variables. The `validate.plist` function calls the `Limit_forOO` and `Limit_forON` functions to accomplish the aforementioned validation. In the event of erroneous input, a fatal error message identifies the nature of the violation. The function executes without an error message if the specification is correct.

The `validate.target.cormat` function checks `CorrMat` for basic properties of a correlation matrix such as symmetry and positive definiteness. In addition, it verifies that all the correlations are within the valid range for the specified marginal probabilities of ordinal variables. The lower and upper bounds of pairwise correlations between two ordinal variables are determined by the two marginal probability vectors, whereas bounds are imposed by the marginal probabilities of the ordinal component of an ordinal-normal pair. This function calls function `valid.limits` to determine the range of valid values of pairwise correlations. The range violation error message indicates that mixed ordinal and normal data with the specified correlations cannot be generated due to distributional constraints.

In the following two examples are given where errors are caught by `validate.target.cormat`.

```
R> set.seed(1000)
R> no.ord <- 3
R> no.norm <- 4
R> mean.vec <- runif(no.norm, 5, 25)
R> mean.vec <- runif(no.norm, 5, 25)
R> sd.vec <- runif(no.norm, 5, 25)
R> marginal <- list(0.3, cumsum(c(0.30, 0.40)), cumsum(c(0.4, 0.2, 0.3)))
R> Sigma1 <- matrix(0, 7, 7)
R> Sigma1[lower.tri(Sigma1)] <- c(0.44, 0.75, 0.52, 0.77, -0.09, -0.67,
+    -0.35, 0.02,0.46,  0.98, -0.93, -0.70, 0.47, -1.00, -0.22, -0.08,
+    -0.22, -0.20, -0.64, 0.90, -0.09)
R> Sigma1 <- Sigma1 + t(Sigma1)
R> diag(Sigma1) <- 1
R> validate.target.cormat(marginal, Sigma1, no.ord, no.norm)

Error in validate.target.cormat(marginal, Sigma1, no.ord, no.norm) :
  Specified correlation matrix is not positive definite!

R> set.seed(1000)
R> Sigma2 <- matrix(c(1.0000000, 0.9672239, 0.8885764, 0.7513335, 0.3898887,
+    0.5813620, -0.4199120, 0.9672239, 1.0000000, 0.8908442, 0.7687458,
+    0.1500516,0.6786741, -0.5896419, 0.8885764, 0.8908442, 1.0000000,
+    0.7484841, 0.3102147, 0.7085261, -0.2208334, 0.7513335, 0.7687458,
+    0.7484841, 1.0000000, 0.2398711, 0.1810759, -0.3896097, 0.3898887,
+    0.1500516, 0.3102147, 0.2398711, 1.0000000, -0.2001247, 0.5699228,
+    0.5813620, 0.6786741, 0.7085261, 0.1810759, -0.2001247, 1.0000000,
+    -0.3796390, -0.4199120, -0.5896419, -0.2208334, -0.3896097, 0.5699228,
+    -0.3796390, 1),7,7)
R> validate.target.cormat(marginal, Sigma2, no.ord, no.norm)

Target matrix is not valid. The following values are invalid.
Corr[ 1 , 2 ] must be between -0.844 and 0.843
Corr[ 1 , 3 ] must be between -0.774 and 0.696
Corr[ 2 , 3 ] must be between -0.866 and 0.866
Error in validate.target.cormat(marginal, Sigma2, no.ord, no.norm) :
  Range violation occurred in the target correlation matrix.
```

The `ordinalize` function discretizes the standard normal variable into categories defined by the cumulative marginal probability values in `pvec`. It is used by the other three core functions and the data generating function to transform standard normal variables to ordinal variables.

4. Example

This section demonstrates the performance of the package via an example. A data set composed of three ordinal (Y_1, Y_2, Y_3) and four normal variables (W_1, W_2, W_3, W_4) with specific marginal distributions and correlation structure (true values are given in Tables 3 and 4), and a sample size of 2000 is generated. The ordinal components were assumed to have two, three and four categories, respectively. Empirical cumulative marginal probabilities for each variable and correlations between each pair are calculated from the generated data. Then the simulation was repeated 1000 times. Th set of cumulative marginal probabilities specified for this example is listed in Table 2. For example, 0.7 in the second column (P_{Y2}) is the cumulative probability of Y_2 to fall in either the first or the second category of that variable, which implies that the probability of the second category is 0.4 (i.e., $0.7 - 0.3$) and of the third category is 0.3. Numbers within parentheses are the empirical 95% confidence intervals (CIs) of the generated ordinal data corresponding to the true cumulative probability of categories of each variable. For instance, the proportion of generated Y_2 that fell in the first or the second category was between $(0.680, 0.721)$ for 95% of the simulated data. Obviously, as `n` gets larger, the confidence intervals are expected to become narrower.

The correlation structure specified for the seven variables in this example is shown in Table 4. The numbers within the parentheses are the 95% confidence limits of the corresponding correlations. For instance, a correlation of 0.18 is desired between Y_1 and Y_2. In the simulated

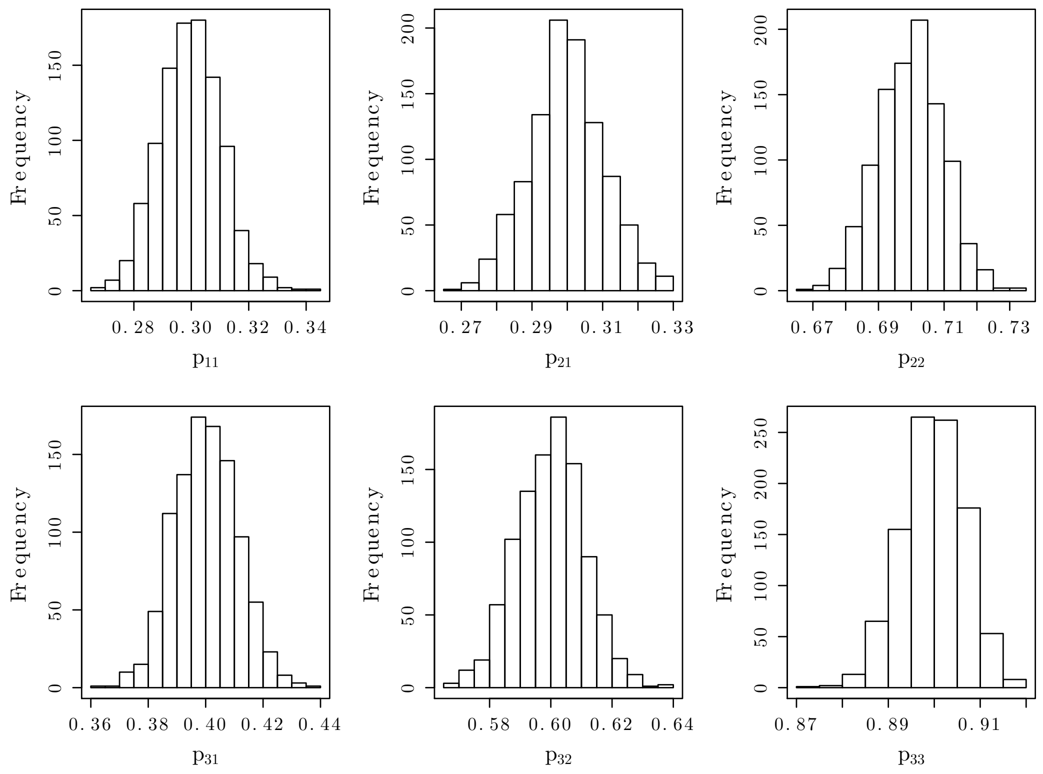

Figure 1: Distributions of empirical cumulative marginal probabilities.

data, the empirical correlation between Y_1 and Y_2 was within $(0.139, 0.225)$ for 95% of the simulations.

Figures 1 and 2 demonstrate distributions of empirical marginal probabilities and correlations, respectively. For brevity, we only present figures for the ordinal-ordinal and ordinal-normal pairs. We observe that 98% of all the simulated data have empirical probabilities within 10% of the true probabilities; and 99% of all the simulated data have empirical correlations within 10% of the true correlations. The close resemblance between the specified and empirically computed quantities on average suggests that the procedure works properly.

5. A comparison of run-time with the SI algorithm

The method proposed by Ruscio and Kaczetow (2008) is also capable of generating simulated data with ordinal and normal components. The algorithm provides an alternative framework to generate data with various combinations of mixed variable types. The method identifies intermediate correlation matrix through an iterative, trial-and-error process.

For the purpose of run-time comparison between **OrdNor** and the SI algorithm, we utilize the R code (`GenData`) that is available online at `http://www.tcnj.edu/~ruscio/taxometrics.html#NonnormalData`.

We generate 15 simulated data sets using each algorithm. We use the same parametric values that are specified in Section 4. Each data set is composed of 3 ordinal variables and 4 normal

P_{Y1}	P_{Y2}	P_{Y3}
0.3 (0.280, 0.321)	0.3 (0.278, 0.322)	0.4 (0.380, 0.422)
	0.7 (0.681, 0.720)	0.6 (0.578, 0.621)
		0.9 (0.887, 0.912)

Table 2: Cumulative marginal probabilities and corresponding empirical 95% CIs.

	W_1	W_2	W_3	W_4
Mean	11.5 (10.955, 12.086)	24.3 (23.429, 25.072)	19.1 (18.326, 19.794)	17.8 (17.391, 18.194)
SD	12.8 (12.409, 13.184)	18.9 (18.319, 19.488)	15.8 (15.302, 16.321)	9.5 (9.203, 9.790)

Table 3: Means and SD (standard deviation), and corresponding empirical 95% CIs.

	Y_1	Y_2	Y_3	W_1	W_2	W_3	W_4
Y_1	1						
Y_2	0.18 (0.134, 0.218)	1					
Y_3	0.30 (0.264, 0.340)	−0.14 (−0.182, −0.099)	1				
W_1	0.21 (0.169, 0.252)	0.01 (−0.040, 0.054)	−0.28 (−0.321, −0.238)	1			
W_2	0.31 (0.271, 0.349)	0.18 (0.139, 0.222)	0.19 (0.149, 0.231)	−0.03 (−0.075, 0.014)	1		
W_3	−0.03 (−0.072, 0.016)	0.39 (0.355, 0.426)	−0.40 (−0.435, −0.362)	−0.09 (−0.137, −0.043)	−0.26 (−0.299, −0.219)	1	
W_4	−0.27 (−0.306, −0.230)	−0.37 (−0.405, −0.332)	−0.09 (−0.133, −0.047)	−0.08 (−0.123, −0.036)	0.36 (0.322, 0.400)	−0.04 (−0.083, 0.006)	1

Table 4: A correlation matrix for multivariate data with three ordinal and four normal variables. The values in parentheses correspond to the empirical 95% CIs.

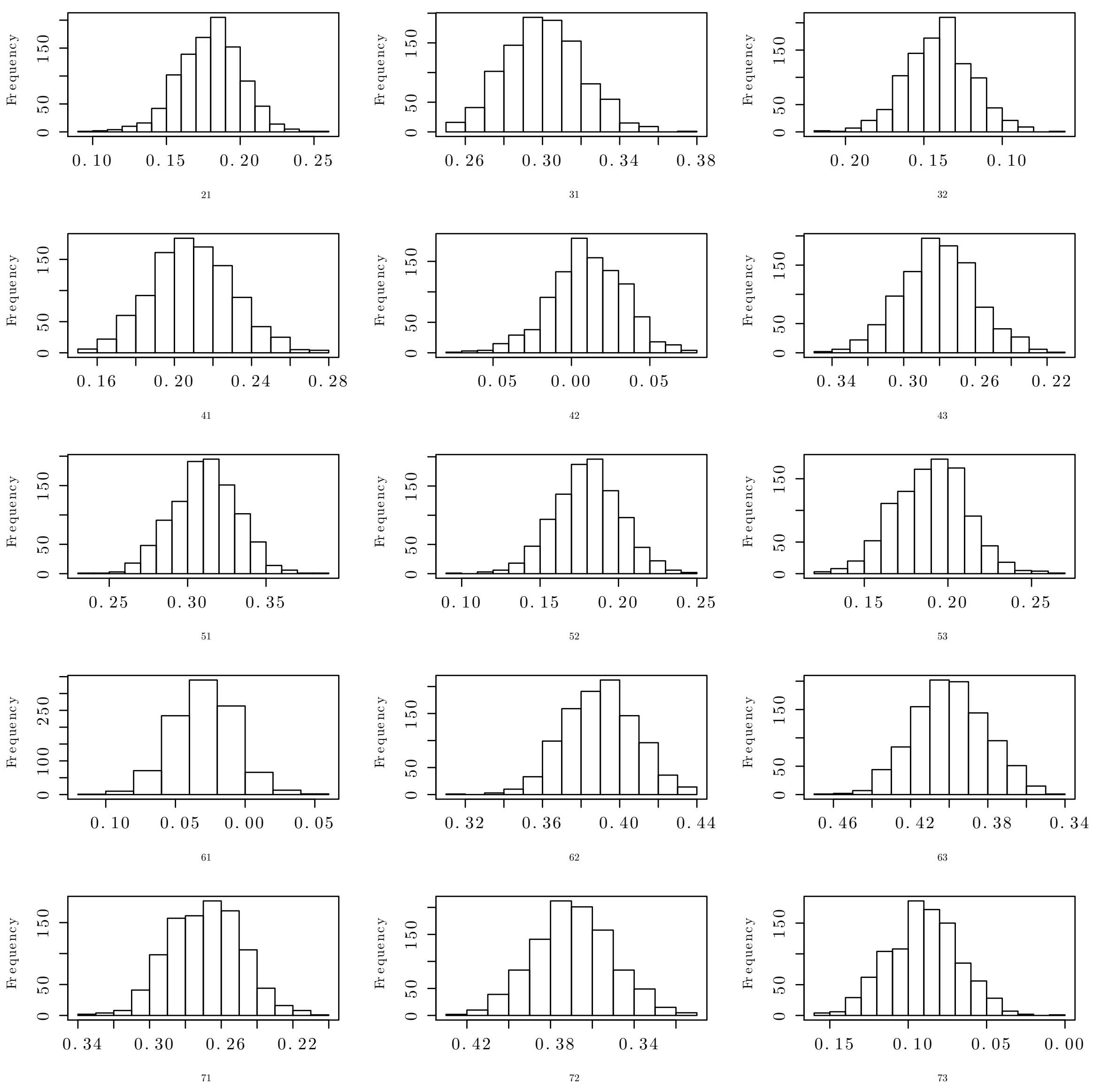

Figure 2: Distributions of empirical pairwise correlations. The first row gives the pairwise correlations between the ordinal-ordinal pairs. The other four rows show the pairwise correlations between the ordinal-normal pairs.

variables. We compare the performance of each algorithm for four different sample sizes.

The results of the comparison are presented in Table 5. The results show that **OrdNor** can be substantially faster for larger data sets. However, for smaller data sets, `GenData` runs as fast or faster than **OrdNor**. The comparative advantage in speed depends on the number of variables, the sample size, and the number of data sets being generated. `GenData` seems to run faster when generating a single data set with a small sample size and a large number of variables. However, **OrdNor** seems to be advantageous when generating large number of sample data sets with the same population parameter values.

Sample size	Time elapsed (s)	
	OrdNor	`GenData`
200	3.95	5.47
2,000	4.06	8.48
20,000	5.00	54.53
50,000	7.06	145.73

Table 5: Run-time comparison between **OrdNor** and `GenData` for generating 10 simulated data sets.

6. Discussion

The paper described the **OrdNor** package which is a useful tool for generating correlated mixed multivariate ordinal and normal data with specified marginal distributions and correlation structure. The underlying idea of the procedure is to generate intermediate multivariate normal data whose components after discretization will give the ordinal components of the multivariate ordinal-normal data with desired specifications. Overall, the performance of the algorithm is very satisfactory in that deviations are within tolerable limits of errors. In general, empirical estimates of parameters based on multivariate ordinal-normal data generated using the package are within 10% of the true parameter values. Larger number of rows would further diminish already minor differences. Our simulation studies (not shown for brevity) suggest that the package successfully implements the algorithm across many real-life scenarios as long as there are no specification errors and correlations are within the feasible limits (Demirtas and Hedeker 2011). In case such complications occur, appropriate warning/error messages will be generated to alert the user. In summary, the **OrdNor** package provides an efficient implementation of the concurrent ordinal and normal data generation algorithm. The package provides a valuable tool for investigators who need a mixed data generation mechanism for their research.

References

Agresti A (2002). *Categorical Data Analysis*. 2nd edition. John Wiley & Sons, New York.

Amatya A, Demirtas H (2015). ***OrdNor**: Concurrent Generation of Ordinal and Normal Data with Given Correlation Matrix and Marginal Distributions.* R package version 2.0, URL `http://CRAN.R-project.org/package=OrdNor`.

Barbiero A, Ferrari P (2015). ***GenOrd**: Simulation of Ordinal and Discrete Variables with Given Correlation Matrix and Marginal Distributions.* R package version 1.3.0, URL `http://CRAN.R-project.org/package=GenOrd`.

Bates D, Maechler M (2015). ***Matrix**: Sparse and Dense Matrix Classes and Methods.* R package version 1.2-2, URL `http://CRAN.R-project.org/package=Matrix`.

Biswas A (2004). "Generating Correlated Ordinal Categorical Random Samples." *Statistics and Probability Letters*, **70**(1), 25–35. `doi:10.1016/j.spl.2004.08.001`.

Demirtas H (2006). "A Method For Multivariate Ordinal Data Generation Given Marginal Distributions and Correlations." *Journal of Statistical Computation and Simulation*, **76**(11), 1017–1025. doi:10.1080/10629360600569246.

Demirtas H, Doganay B (2012). "Simultaneous Generation of Binary and Normal Data with Specified Marginal and Association Structures." *Journal of Biopharmaceutical Statistics*, **22**(2), 223–236. doi:10.1080/10543406.2010.521874.

Demirtas H, Hedeker D (2011). "A Practical Way for Computing Approximate Lower and Upper Correlation Bounds." *The American Statistician*, **65**(2), 104–109. doi:10.1198/tast.2011.10090.

Emrich J, Piedmonte M (1991). "A Method for Generating High-Dimensional Multivariate Binary Variates." *The American Statistician*, **45**(4), 302–304. doi:10.1080/00031305.1991.10475828.

Ferrari P, Barbiero A (2012). "Simulating Ordinal Data." *Multivariate Behavioral Research*, **47**(4), 566–589. doi:10.1080/00273171.2012.692630.

Gange S (1995). "Generating Multivariate Categorical Variates Using the Iterative Proportional Fitting Algorithm." *The American Statistician*, **49**(2), 134–138. doi:10.1080/00031305.1995.10476130.

Genz A, Bretz F (2009). *Computation of Multivariate Normal and t Probabilities*. Lecture Notes in Statistics. Springer-Verlag, Heidelberg. doi:10.1007/978-3-642-01689-9.

Genz A, Bretz F, Miwa T, Mi X, Leisch F, Scheipl F, Hothorn T (2015). ***mvtnorm**: Multivariate Normal and t Distributions*. R package version 1.0-3, URL http://CRAN.R-project.org/package=mvtnorm.

Lee A (1993). "Generating Random Binary Deviates Having Fixed Marginal Distributions and Specified Degrees of Association." *The American Statistician*, **47**(3), 209–215. doi:10.2307/2684980.

Oman S, Zucker D (2001). "Modeling and Generating Correlated Binary Variables." *Biometrika*, **88**(1), 287–290. doi:10.1093/biomet/88.1.287.

Park C, Park T, Shin D (1996). "A Simple Method for Generating Correlated Binary Variates." *The American Statistician*, **50**(4), 306–310. doi:10.2307/2684925.

Qaqish B (2003). "A Family of Multivariate Binary Distributions for Simulating Binary Variables with Specified Marginal Means and Correlations." *Biometrika*, **90**(2), 455–463. doi:10.1093/biomet/90.2.455.

Ruscio J, Kaczetow W (2008). "Simulating Multivariate Nonnormal Data Using an Iterative Technique." *Multivariate Behavioral Research*, **43**(3), 355–381. doi:10.1080/00273170802285693.

Schäfer J, Opgen-Rhein R, Zuber V, Ahdesmäki M, Duarte Silva AP, Strimmer K (2015). ***corpcor**: Efficient Estimation of Covariance and (Partial) Correlation*. R package version 1.6.8, URL http://CRAN.R-project.org/package=corpcor.

Yahav I, Shmueli G (2012). "On Generating Multivariate Poisson Data in Management Science Applications." *Applied Stochastic Models in Business and Industry*, **28**(1), 91–102. doi:10.1002/asmb.901.

Affiliation:

Anup Amatya
Department of Public Health Sciences
New Mexico State University
1338 International Mall, RM 102
Las Cruces, NM 88011, United States of America
E-mail: aamatya@nmsu.edu

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
November 2015, Volume 68, Code Snippet 2 *Submitted:* 2014-01-09
doi:10.18637/jss.v068.c02 *Accepted:* 2014-12-01