Package 'PoisBinOrdNonNor'

January 10, 2018

Type Package

Title Generation of Up to Four Different Types of Variables

Version 1.4

Date 2018-01-10

Author Hakan Demirtas, Rachel Nordgren, Rawan Allozi

Maintainer Rawan Allozi <ralloz2@uic.edu>

Description Generation of a chosen number of count, binary, ordinal, and continuous random variables, with specified correlations and marginal properties.

License GPL-2 | GPL-3

Depends Matrix, corpcor, MASS, GenOrd, BB

Suggests moments

NeedsCompilation no

Repository CRAN

Date/Publication 2018-01-10 16:13:11 UTC

R topics documented:

`PoisBinOrdNonNor-package`

Generation of up to Four Different Types of Variables

Description

Simultaneous generation of a chosen number of count, binary, ordinal, and continuous random variables, with specified correlations and marginal distributions. Throughout the package, the word 'Poisson' is used to imply count data under the assumption of Poisson distribution; and continuous variables can take any shape allowed by Fleishman polynomials.

Generation of a chosen number of count, binary, ordinal, and continuous (via Fleishman polynomials) random variables, with specified correlations and marginal properties. The correlation matrix and the generated data follow the order of Poisson, binary, ordinal and continuous.

Details

Package:	PoisBinOrdNonNor
Type:	Package
Version:	1.4
Date:	2018-01-10
License:	GPL-2 | GPL-3

This package consists of five public functions. The function `check.params` validates the input parameters to avoid obvious specification errors of the marginal parameters. The function `validate.cor.mat` validates an input target correlation matrix to make sure that it is a legitimate correlation matrix, and then calls `lower.upper.cors` with the rest of the input parameters to generate approximate maximum and minimum feasible bounds, and then checks that each entry is within its bounds. The function `find.cor.mat.star` creates the intermediate correlation matrix. Finally, given the output from `find.cor.mat.star` along with the other variable specifications, the function `genPBONN` generates the simultaneous random data, following the target correlation matrix and the marginal input parameters.

Note

The approximation used to find the correlation for Poisson variables is not very accurate once lambda is less than 1, and becomes less accurate as lambda gets closer to 0.

A flag is used to specify if ordinal probabilities are cumulative–default is FALSE.

Binary variables can be listed separately or combined with ordinal variables–the results will be equivalent. Any variables listed as ordinal are affected by the cumulative flag.

Author(s)

Hakan Demirtas, Rachel Nordgren, Rawan Allozi

Maintainer: Rawan Allozi <ralloz2@uic.edu>

References

Amatya, A. & Demirtas, H. (2015) Simultaneous generation of multivariate mixed data with Poisson and normal marginals. *Journal of Statistical Computation and Simulation* **85:15**, 3129–3139.

Demirtas, H. (2014). Joint generation of binary and nonnormal continuous data. *Journal of Biometrics and Biostatistics* **5:3:1000199**, 1–9.

Demirtas, H. & Hedeker, D. (2011) A practical way for computing approximate lower and upper correlation bounds. *American Statistician* **65:2**, 104–109.

Demirtas, H. & Hedeker, D. (2016). Computing the point-biserial correlation under any underlying continuous distribution. *Communications in Statistics – Simulation and Computation*, **45:8**, 2744–2751.

Demirtas, H., Hedeker, D. & Mermelstein, R. J. (2012) Simulation of massive public health data by power polynomials. *Statistics in Medicine* **31:27**, 3337–3346.

`check.params` *Validates the adjusted marginal parameter lists*

Description

This function validates the lists of marginal parameters for the chosen number of Poisson, ordinal (and binary), and continuous (via Fleishman polynomials) random variables. The list for ordinal/binary variables must be cumulative.

Usage

```
check.params(no.pois = 0, no.ordbin = 0, no.nonn = 0, pois.list = list(),
    ordbin.list = list(), nonn.list = list())
```

Arguments

`no.pois`	The number of Poisson random variables desired. Defaults to 0.
`no.ordbin`	The number of ordinal and binary random variables desired. Defaults to 0.
`no.nonn`	The number of continuous random variables desired, created using Fleishman polynomials. Defaults to 0.
`pois.list`	A list of the lambda values, which must be greater than 0. Length will be equal to no.pois, or an error will be thrown. Defaults to an empty list.
`ordbin.list`	A list of vectors containing the cumulative probabilities for each variable. Each vector should have entries between 0 and 1 inclusive, in increasing order. Length must be equal to no.ordbin. Defaults to an empty list.
`nonn.list`	A list of vectors containing the first four moments of each variable, in order. If only two parameters are supplied, they will be assumed to be skew and excess kurtosis, with mean=0 and variance=1. If only three parameters are supplied, they will be assumed to be variance, skew and excess kurtosis, with mean=0. If less than two parameters or more than four parameters are supplied for any variable, an error will be raised. Variance must be positive, and excess kurtosis must be greater than or equal to skew^2 - 2. Length must be equal to no.nonn. Defaults to an empty list.

Details

Each list of transformed parameters is sent to the appropriate helper function. Requirements for each are mentioned in the description above.

Value

TRUE if all items meet necessary criteria. If not, program will have stopped with a specific error message.

Examples

```
## Not run:
#This pois.list will produce an error since the first entry is 0.
check.params(no.pois = 2, pois.list = list(0, 1))

#This ordbin.list will produce an error since it is not cumulative.
check.params(no.ordbin = 2, ordbin.list = list(.25, c(.25, 0, .75)))

#This ordbin.list will produce an error since the last entry is > 1.
check.params(no.ordbin = 2, ordbin.list = list(.25, c(.25, .5, 1.25)))

#This ordbin.list will produce an error since the first entry < 0.
check.params(no.ordbin = 2, ordbin.list = list(.25, c(-.25, .5, .75)))

#This nonn.list will produce an error since 0 < 2^2 - 2
check.params(no.nonn = 2, nonn.list = list(c(2,0), c(.5, 1, 2, 0)))

#This nonn.list will produce an error since the variance = 0.
check.params(no.nonn = 2, nonn.list = list(c(0,0), c(.5, 0, 0, 0)))

## End(Not run)

check.params(no.pois = 1, pois.list = list(1), no.ordbin = 2,
    ordbin.list = list(.25, c(.25, .25, .75)), no.nonn = 1,
    nonn.list = list(c(0, 1, 0, 3)))
```

---

find.cor.mat.star *Finds intermediate correlation matrix*

Description

This function calculates an intermediate correlation matrix for a chosen number of Poisson, binary, ordinal, and continuous (via Fleishman polynomials) random variables, with specified target correlations and marginal properties.

The correlation matrix follows the order of Poisson, binary, ordinal, continuous.

Usage

```
find.cor.mat.star(cor.mat, no.pois = 0, no.bin = 0, no.ord = 0,
    no.nonn = 0, pois.list = list(), bin.list = list(), ord.list = list(),
    is.ord.list.cum = FALSE, nonn.list = list())
```

Arguments

`cor.mat`	The desired target correlation matrix.
`no.pois`	The number of Poisson random variables desired. Defaults to 0.
`no.bin`	The number of binary random variables desired. Defaults to 0.
`no.ord`	The number of ordinal random variables desired. Defaults to 0.
`no.nonn`	The number of continuous random variables desired, created using Fleishman polynomials. Defaults to 0.
`pois.list`	A list of the lambda values, which must be greater than 0. Length will be equal to no.pois, or an error will be thrown. Defaults to an empty list.
`bin.list`	A list of vectors containing the probabilities for each variable. Each vector should have 2 entries between 0 and 1 inclusive, and sum to 1. Length must be equal to no.bin. Defaults to an empty list.
`ord.list`	A list of vectors containing the probabilities for each variable. If is.ord.list.cum is TRUE, each vector should have entries between 0 and 1, in increasing order. Otherwise, each vector should have entries between 0 and 1 inclusive that sum to 1. Length must be equal to no.ord. Defaults to an empty list.
`is.ord.list.cum`	Flag for whether the ordinal list supplied contains cumulative probabilities. Defaults to FALSE.
`nonn.list`	A list of vectors containing the first four moments of each variable, in order. If only two parameters are supplied, they will be assumed to be skew and excess kurtosis, with mean = 0 and variance = 1. If only three parameters are supplied, they will be assumed to be variance, skew and excess kurtosis, with mean = 0. If less than two parameters or more than four parameters are supplied for any variable, an error will be raised. Variance must be positive, and excess kurtosis must be greater than or equal to skew^2-2. Length must be equal to no.nonn. Defaults to an empty list.

Details

First, the target correlation matrix and input parameters are checked using `validate.cor.mat`.

Second, the input lists are transformed and combined.

Third, each entry is tranformed in order.

- Poisson-Poisson entries follow Amatya and Demirtas (2016).
- Ordinal/Binary-Ordinal/Binary entries are transformed using a call to *ordcont* in **GenOrd** package.
- Continuous-Continuous entries use the correlation matrix for the underlying Fleishman polynomials.

- All mixed entries use the approximation from Demirtas and Hedeker 2016: corr(X, Z) = cor(X, Y) * cor(Y, Z) where Y is a standard normal variable. In other words, the association between X and Z is assumed to be fully explained by the association between X and Y and the association between Y and Z.

Fourth, the resulting matrix is checked for validity. If it is not a valid correlation matrix, a warning message is printed and the nearest valid matrix is returned instead. (*nearPD* from **Matrix)**

Value

A (no.pois+no.bin+no.ord+no.nonn) by (no.pois+no.bin+no.ord+no.nonn) square matrix, containing the intermediate correlation values.

References

Amatya, A. & Demirtas, H. (2015) Simultaneous generation of multivariate mixed data with Poisson and normal marginals. *Journal of Statistical Computation and Simulation* **85:15**, 3129–3139.

Demirtas, H. (2014). Joint generation of binary and nonnormal continuous data. *Journal of Biometrics and Biostatistics* **5:3:1000199**, 1–9.

Demirtas, H. & Hedeker, D. (2011) A practical way for computing approximate lower and upper correlation bounds. *American Statistician* **65:2**, 104–109.

Demirtas, H. & Hedeker, D. (2016). Computing the point-biserial correlation under any underlying continuous distribution. *Communications in Statistics – Simulation and Computation*, **45:8**, 2744–2751.

Demirtas, H., Hedeker, D. & Mermelstein, R. J. (2012) Simulation of massive public health data by power polynomials. *Statistics in Medicine* **31:27**, 3337–3346.

Examples

```
validate.cor.mat(cor.mat = .3 * diag(3) + .7, no.pois = 3,
    pois.list = list(.25, .5, 1))
find.cor.mat.star(cor.mat = .3 * diag(3) + .7, no.pois = 3,
    pois.list = list(.25, .5, 1))

validate.cor.mat(cor.mat = .8 * diag(3) + .2, no.ord = 3,
    ord.list = list(c(.2, .8), c(.1, .2, .3, .4), c(.8, 0, .1, .1)))
find.cor.mat.star(cor.mat = .8 * diag(3) + .2, no.ord = 3,
    ord.list = list(c(.2, .8), c(.1, .2, .3, .4), c(.8, 0, .1, .1)))

validate.cor.mat(cor.mat = .5 * diag(3) + .5, no.pois = 1, no.nonn = 1,
    no.ord = 1, pois.list = list(.5), ord.list = list(c(.8, 0, .1, .1)),
    nonn.list = list(c(0, 1, 0, 1)))
find.cor.mat.star(cor.mat = .5 * diag(3) + .5, no.pois = 1, no.nonn = 1,
    no.ord = 1, pois.list = list(.5), ord.list = list(c(.8, 0, .1, .1)),
    nonn.list = list(c(0, 1, 0, 1)))
```

genPBONN *Engine (generation) function for the PoisBinOrdNonNor package*

Description

This function generates a chosen number of Poisson, binary, ordinal, and continuous (via Fleishman polynomials) random variables, with specified correlations and marginal properties.

The correlation matrix and the generated data follow the order of Poisson, binary, ordinal and continuous.

Usage

```
genPBONN(n, cmat.star, no.pois = 0, no.bin = 0, no.ord = 0,
    no.nonn = 0, pois.list = list(), bin.list = list(),
    ord.list = list(), is.ord.list.cum = FALSE, nonn.list = list())
```

Arguments

n	The number of rows in the generated data matrix.
cmat.star	The intermediate correlation matrix.
no.pois	The number of Poisson random variables desired. Defaults to 0.
no.bin	The number of binary random variables desired. Defaults to 0.
no.ord	The number of ordinal random variables desired. Defaults to 0.
no.nonn	The number of continuous random variables desired, created using Fleishman polynomials. Defaults to 0.
pois.list	A list of the lambda values, which must be greater than 0. Length will be equal to no.pois, or an error will be thrown. Defaults to an empty list.
bin.list	A list of vectors containing the probabilities for each variable. Each vector should have 2 entries between 0 and 1 inclusive, and sum to 1. Length must be equal to no.bin. Defaults to an empty list.
ord.list	A list of vectors containing the probabilities for each variable. Each vector should have entries between 0 and 1 inclusive, and sum to 1. Length must be equal to no.ord. Defaults to an empty list.
is.ord.list.cum	Flag for whether the ordinal list supplied contains cumulative probabilities. Defaults to FALSE.
nonn.list	A list of vectors containing the first four moments of each variable, in order. If only two parameters are supplied, they will be assumed to be skew and excess kurtosis, with mean = 0 and variance = 1. If only three parameters are supplied, they will be assumed to be variance, skew and excess kurtosis, with mean = 0. If less than two parameters or more than four parameters are supplied for any variable, an error will be raised. Variance must be positive, and excess kurtosis must be greater than or equal to skew^2-2. Length must be equal to no.nonn. Defaults to an empty list.

Details

After transformation and checking of parameters, a n by (no.pois+no.bin+no.ord+no.nonn) matrix of standard normal random data is generated, using cmat.star as the correlation matrix.

Then for each variable, the appropriate transformation is applied to each column of the data generated.

Value

A n by (no.pois+no.bin+no.ord+no.nonn) matrix. Each column corresponds to a variable, and each row is one random sample.

References

Amatya, A. & Demirtas, H. (2015) Simultaneous generation of multivariate mixed data with Poisson and normal marginals. *Journal of Statistical Computation and Simulation* **85:15**, 3129–3139.

Demirtas, H. (2014). Joint generation of binary and nonnormal continuous data. *Journal of Biometrics and Biostatistics* **5:3:1000199**, 1–9.

Demirtas, H. & Hedeker, D. (2011) A practical way for computing approximate lower and upper correlation bounds. *American Statistician* **65:2**, 104–109.

Demirtas, H. & Hedeker, D. (2016). Computing the point-biserial correlation under any underlying continuous distribution. *Communications in Statistics – Simulation and Computation*, **45:8**, 2744–2751.

Demirtas, H., Hedeker, D. & Mermelstein, R. J. (2012) Simulation of massive public health data by power polynomials. *Statistics in Medicine* **31:27**, 3337–3346.

Examples

```
## Not run:
cmat.star <- find.cor.mat.star(cor.mat = .8 * diag(8) + .2, no.pois = 2, no.ord = 4,
    no.nonn = 2, pois.list = list(1, 2), ord.list = list(c(.2, .8), c(.5, .5),
    c(.1, .2, .3, .4), c(.8, 0, .1, .1)), nonn.list = list(c(-1, 1, 0, 1), c(0, 3, 0, 2)))
mydata <- genPBONN(1000, no.pois = 2, no.ord = 4, no.nonn = 2,
    cmat.star = cmat.star, pois.list = list(1, 2),
    ord.list = list(c(.2, .8), c(.5, .5),c(.1, .2, .3, .4),
    c(.8, 0, .1, .1)), nonn.list = list(c(-1, 1, 0, 1), c(0, 3, 0, 2)))

apply(mydata, 2, mean)
apply(mydata, 2, var)
library(moments)
apply(mydata, 2, skewness)
apply(mydata, 2, kurtosis) - 3
lapply(apply(mydata[, 1:6], 2, table), prop.table)
cor(mydata)

## End(Not run)
```

`lower.upper.cors` *Computes lower and upper correlation bounds*

Description

This function calculates the approximate upper and lower correlation bounds for all variable pairs.

Usage

```
lower.upper.cors(no.pois = 0, no.bin = 0, no.ord = 0, no.nonn = 0,
    pois.list = list(), bin.list = list(), ord.list = list(),
    is.ord.list.cum=FALSE, nonn.list = list())
```

Arguments

`no.pois`	The number of Poisson random variables desired. Defaults to 0.
`no.bin`	The number of binary random variables desired. Defaults to 0.
`no.ord`	The number of ordinal random variables desired. Defaults to 0.
`no.nonn`	The number of continuous random variables desired, created using Fleishman polynomials. Defaults to 0.
`pois.list`	A list of the lambda values, which must be greater than 0. Length will be equal to no.pois, or an error will be thrown. Defaults to an empty list.
`bin.list`	A list of vectors containing the probabilities for each variable. Each vector should have 2 entries between 0 and 1 inclusive, and sum to 1. Length must be equal to no.bin. Defaults to an empty list.
`ord.list`	A list of vectors containing the probabilities for each variable. If is.ord.list.cum is TRUE, each vector should have entries between 0 and 1, in increasing order. Otherwise, each vector should have entries between 0 and 1 inclusive that sum to 1. Length must be equal to no.ord. Defaults to an empty list.
`is.ord.list.cum`	Flag for whether the ordinal list supplied contains cumulative probabilities. Defaults to FALSE.
`nonn.list`	A list of vectors containing the first four moments of each variable, in order. If only two parameters are supplied, they will be assumed to be skew and excess kurtosis, with mean=0 and variance=1. If only three parameters are supplied, they will be assumed to be variance, skew and excess kurtosis, with mean=0. If less than two parameters or more than four parameters are supplied for any variable, an error will be raised. Variance must be positive, and excess kurtosis must be greater than or equal to skew^2-2. Length must be equal to no.nonn. Defaults to an empty list.

Details

First, the parameters are transformed as necessary and then checked for validity using check.params. Then, for each pair of variables, we generate approximate upper and lower correlation bounds, using Demirtas and Hedeker's method, which involves generating a large sample of data and sorting it to find approximate bounds.

Value

min	A (no.pois+no.bin+no.ord+no.nonn) by (no.pois+no.bin+no.ord+no.nonn) square matrix showing the calculated lower correlation bound for each pair of variables.
max	A (no.pois+no.bin+no.ord+no.nonn) by (no.pois+no.bin+no.ord+no.nonn) square matrix showing the calculated upper correlation bound for each pair of variables.

Note

If (no.pois+no.bin+no.ord+no.nonn) is equal to 1, 1 is returned. (If less than 1, an error is thrown at the beginning of the function.)

The correlation matrix follows the order of Poisson, binary, ordinal, continuous.

References

Demirtas, H. & Hedeker, D. (2011) A practical way for computing approximate lower and upper correlation bounds. *American Statistician* **65:2**, 104–109.

Examples

```
lower.upper.cors(no.nonn = 2, nonn.list = list(c(0, 1, 0, 1), c(0, 1, 0, 2)))
lower.upper.cors(no.pois = 2, pois.list = list(.5, 1))
lower.upper.cors(no.ord = 3,  ord.list = list(c(.2, .8), c(.1, .2, .3, .4),
    c(.8, 0, .1, .1)))
lower.upper.cors(no.pois = 1, no.nonn = 1, no.ord = 1, pois.list = list(.5),
    ord.list = list(c(.8, 0, .1, .1)), nonn.list = list(c(0, 1, 0, 1)))
```

---

validate.cor.mat *Validates the target correlation matrix*

Description

This function validates the user-inputted target correlation matrix for the chosen number of Poisson, binary, ordinal, and continuous (via Fleishman polynomials) random variables.

The correlation matrix follows the order of Poisson, binary, ordinal, continuous.

Usage

```
validate.cor.mat(cor.mat, no.pois = 0, no.bin = 0, no.ord = 0,
    no.nonn = 0, pois.list = list(), bin.list = list(),
    ord.list = list(), is.ord.list.cum=FALSE, nonn.list = list())
```

Arguments

cor.mat	The desired target correlation matrix.
no.pois	The number of Poisson random variables desired. Defaults to 0.
no.bin	The number of binary random variables desired. Defaults to 0.
no.ord	The number of ordinal random variables desired. Defaults to 0.
no.nonn	The number of continuous random variables desired, created using Fleishman polynomials. Defaults to 0.
pois.list	A list of the lambda values, which must be greater than 0. Length will be equal to no.pois, or an error will be thrown. Defaults to an empty list.
bin.list	A list of vectors containing the probabilities for each variable. Each vector should have 2 entries between 0 and 1 inclusive, and sum to 1. Length must be equal to no.bin. Defaults to an empty list.
ord.list	A list of vectors containing the probabilities for each variable. If is.ord.list.cum is TRUE, each vector should have entries between 0 and 1, in increasing order. Otherwise, each vector should have entries between 0 and 1 inclusive that sum to 1. Length must be equal to no.ord. Defaults to an empty list.
is.ord.list.cum	Flag for whether the ordinal list supplied contains cumulative probabilities. Defaults to FALSE.
nonn.list	A list of vectors containing the first four moments of each variable, in order. If only two parameters are supplied, they will be assumed to be skew and excess kurtosis, with mean = 0 and variance = 1. If only three parameters are supplied, they will be assumed to be variance, skew and excess kurtosis, with mean = 0. If less than two parameters or more than four parameters are supplied for any variable, an error will be raised. Variance must be positive, and excess kurtosis must be greater than or equal to skew^2-2. Length must be equal to no.nonn. Defaults to an empty list.

Details

First, the matrix is checked for the correct dimensions, then for symmetry, and then for being positive definite. Then each entry is checked to be in [-1, 1] and the diagonal entries must be 1. If not, an error is thrown.

Second, the parameters are transformed and then checked for validity using check.params.

Third, the minimum and maximum correlation matrices are found using lower.upper.cors. If all entries are within the bounds, TRUE is returned. Otherwise, a list of which cells are invalid and their required bounds are printed, and FALSE is returned.

Value

TRUE if the correlation matrix is valid, and FALSE if not.

Examples

```
validate.cor.mat(cor.mat = .2 * diag(3) + .8, no.pois = 3,
    pois.list = list(.25, .5, 1))
validate.cor.mat(cor.mat = .7 * diag(3) + .3, no.ord = 3,
    ord.list = list(c(.2, .8), c(.1, .2, .3, .4), c(.8, 0, .1, .1)))
validate.cor.mat(cor.mat = .25 * diag(3) + .75, no.pois = 1,
    no.nonn = 1, no.ord = 1, pois.list = list(.5),
    ord.list = list(c(.8, 0, .1, .1)), nonn.list=list(c(0, 1, 0, 1)))
validate.cor.mat(cor.mat = .35 * diag(3) + .65, no.pois = 1,
    no.nonn = 1, no.ord = 1, pois.list = list(.5),
    ord.list = list(c(.8, 0, .1, .1)), nonn.list=list(c(0, 1, 0, 1)))
```

Index