Package: PoisBinNonNor (via r-universe)

October 31, 2024

Type Package

Title Data Generation with Poisson, Binary and Continuous Components

Version 1.3.3

Date 2021-03-21

Author Gul Inan, Hakan Demirtas, Ran Gao

Maintainer Ran Gao <rgao8@uic.edu>

Description Generation of multiple count, binary and continuous variables simultaneously given the marginal characteristics and association structure. Throughout the package, the word 'Poisson' is used to imply count data under the assumption of Poisson distribution. The details of the method are explained in Amatya et al. (2015) <DOI:10.1080/00949655.2014.953534>.

License GPL-2 | GPL-3

Depends BB, corpcor, Matrix, mvtnorm

NeedsCompilation no

Date/Publication 2021-03-22 05:30:05 UTC

Repository https://bernice0321.r-universe.dev

RemoteUrl https://github.com/cran/PoisBinNonNor

RemoteRef HEAD

RemoteSha c70434217bfc61c12cde3e3577ec3a126f9e2806

Contents

`PoisBinNonNor-package` *Data Generation with Count, Binary and Continuous Components*

Description

Provides R functions for generation of multiple count, binary and continuous variables simultaneously given the marginal characteristics and association structure. Continuous variables can be of any nonnormal shape allowed by the Fleishman polynomials, taking the normal distribution as a special case.

Details

Package:	PoisBinNonNor
Type:	Package
Version:	1.3.3
Date:	2021-03-21
License:	GPL-2 | GPL-3

The package consists of fourteen functions. The functions `validation.bin`, `validation.corr`, and `validation.skewness.kurtosis` validate the specified quantities. `correlation.limits` returns the lower and upper bounds of pairwise correlations of Poisson, binary and continuous variables. `correlation.bound.check` validates pairwise correlation values.
`intermediate.corr.PP`, `intermediate.corr.BB`, `intermediate.corr.CC`, `intermediate.corr.PB`, `intermediate.corr.PC`, and `intermediate.corr.BC` compute intermediate correlation matrix for Poisson-Poisson combinations, binary-binary, continuous-continuous, Poisson-binary, Poisson-continuous, binary-continuous combinations, respectively. The function `overall.corr.mat` assembles the final correlation matrix. The engine function `gen.PoisBinNonNor` generates mixed data in accordance with the specified marginal and correlational quantities. Throughout the package, variables are supposed to be inputted in a certain order, namely, first count variables, next binary variables, and then continuous variables should be placed.

Author(s)

Gul Inan, Hakan Demirtas, Ran Gao

Maintainer: Ran Gao <rgao8@uic.edu>

References

Amatya, A. and 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. and Hedeker, D. (2011). A practical way for computing approximate lower and upper correlation bounds. The American Statistician, 65(2), 104-109.

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

---

correlation.bound.check
Checks if the pairwise correlation among variables are within the feasible range

---

Description

This function checks if there are range violations among correlation of Poisson-Poisson, Poisson-binary, Poisson-continuous, binary-binary, binary-continuous, and continuous-continuous combinations.

Usage

```
correlation.bound.check(n.P, n.B, n.C, lambda.vec = NULL, prop.vec = NULL,
coef.mat = NULL, corr.vec = NULL, corr.mat = NULL)
```

Arguments

`n.P`	Number of Poisson variables.
`n.B`	Number of binary variables.
`n.C`	Number of continuous variables.
`lambda.vec`	Rate vector for Poisson variables.
`prop.vec`	Proportion vector for binary variables.
`coef.mat`	Matrix of coefficients produced from `fleishman.coef`.
`corr.vec`	Vector of elements below the diagonal of correlation matrix ordered columnwise.
`corr.mat`	Specified correlation matrix.

Value

The function returns TRUE if no specification problem is encountered. Otherwise, it returns an error message.

References

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

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

See Also

validation.corr, correlation.limits

Examples

```
## Not run:
n.P<-1
n.B<-1
n.C<-1
lambda.vec<-c(1)
prop.vec<-c(0.3)
coef.mat<-matrix(c(-0.3137491,0.8263239,0.3137491,0.0227066),4,1,byrow=F)
corr.mat=matrix(c(1,0.2,0.1,0.2,1,0.5,0.1,0.5,1),3,3)
correlation.bound.check(n.P,n.B,n.C,lambda.vec,prop.vec,coef.mat,corr.vec=NULL,
corr.mat)

n.P<-2
n.B<-2
n.C<-2
lambda.vec<-c(1,2)
prop.vec<-c(0.3,0.5)
coef.mat<-matrix(c(
-0.3137491,  0.0000000,
 0.8263239,  1.0857433,
 0.3137491,  0.0000000,
 0.0227066, -0.0294495),4,2,byrow=F)
corr.mat=matrix(0.8,6,6)
diag(corr.mat)=1
correlation.bound.check(n.P,n.B,n.C,lambda.vec,prop.vec,coef.mat,corr.vec=NULL,
corr.mat)

## End(Not run)
```

---

correlation.limits	*Computes lower and upper correlation bounds for each pair of variables*

Description

This function computes lower and upper limits for pairwise correlations of Poisson-Poisson, Poisson-binary, Poisson-continuous, binary-binary, binary-continuous, and continuous-continuous combinations.

Usage

```
correlation.limits(n.P, n.B, n.C, lambda.vec = NULL, prop.vec = NULL,
coef.mat = NULL)
```

Arguments

`n.P`	Number of Poisson variables.
`n.B`	Number of binary variables.
`n.C`	Number of continuous variables.
`lambda.vec`	Rate vector for Poisson variables.
`prop.vec`	Proportion vector for binary variables.
`coef.mat`	Matrix of coefficients produced from `fleishman.coef`.

Details

While the function computes the exact lower and upper bounds for pairwise correlations among binary-binary variables as formulated in Demirtas et al. (2012), it computes approximate lower and upper bounds for pairwise correlations among Poisson-Poisson, Poisson-binary, Poisson-continuous, binary-continuous, and continuous-continuous variables through the method suggested by Demirtas and Hedeker (2011).

Value

The function returns a matrix of size (n.P + n.B + n.C)*(n.P + n.B + n.C), where the lower triangular part of the matrix contains the lower bounds and the upper triangular part of the matrix contains the upper bounds of the feasible correlations.

References

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

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

See Also

`validation.corr`, `correlation.bound.check`

Examples

```
## Not run:
n.P<-3
n.B<-2
n.C<-3
lambda.vec<-c(1,2,3)
prop.vec<-c(0.3,0.5)
coef.mat<-matrix(c(
-0.3137491,  0.0000000,  0.1004464,
```

```
 0.8263239,  1.0857433,  1.1050196,
 0.3137491,  0.0000000, -0.1004464,
 0.0227066, -0.0294495, -0.0400078),4,3,byrow=F)

#Correlation limits among Poisson variables
correlation.limits(n.P,n.B=0,n.C=0,lambda.vec,prop.vec=NULL,coef.mat=NULL)

#See also Cor.PP.Limit in R package PoisNor

#Correlation limits among binary variables
correlation.limits(n.P=0,n.B,n.C=0,lambda.vec=NULL,prop.vec,coef.mat=NULL)

#See also correlation.limits in R package BinNonNor

#Correlation limits among continuous variables
correlation.limits(n.P=0,n.B=0,n.C,lambda.vec=NULL,prop.vec=NULL,coef.mat)

#Correlation limits among Poisson and binary variables and within themselves.
correlation.limits(n.P,n.B,n.C=0,lambda.vec,prop.vec,coef.mat=NULL)

#Correlation limits among Poisson and continuous variables and within themselves.
correlation.limits(n.P,n.B=0,n.C,lambda.vec,prop.vec=NULL,coef.mat)

#Correlation limits among binary and continuous variables and within themselves.
correlation.limits(n.P=0,n.B,n.C,lambda.vec=NULL,prop.vec,coef.mat)

#Correlation limits among Poisson, binary, and continuous variables and within themselves.
correlation.limits(n.P,n.B,n.C,lambda.vec,prop.vec,coef.mat)

n.P<-2
lambda.vec=c(-1,1)
correlation.limits(n.P,n.B=0,n.C=0,lambda.vec,prop.vec=NULL,coef.mat=NULL)


## End(Not run)
```

`fleishman.coef` *Computes the coefficients of Fleishman third order polynomials*

Description

Computes the coefficients of Fleishman third order polynomials given the marginal skewness and kurtosis parameters of continuous variables.

Usage

```
fleishman.coef(n.C, skewness.vec = NULL, kurtosis.vec = NULL)
```

Arguments

`n.C`	Number of continuous variables.
`skewness.vec`	Skewness vector for continuous variables.
`kurtosis.vec`	Kurtosis vector for continuous variables.

Details

The execution of the function may take some time since it uses multiple starting points to solve the system of nonlinear equations based on the third order Fleishman polynomials. However, since users need to run it only once for a given set of specifications, it does not constitute a problem.

Value

A matrix of coefficients. The columns represent the variables and rows represent the corresponding a,b,c, and d coefficients.

References

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

Fleishman, A.I. (1978). A method for simulating non-normal distributions. Psychometrika, 43(4), 521-532.

See Also

validation.skewness.kurtosis

Examples

```
## Not run:
#Consider four continuous variables, which come from
#Exp(1),Beta(4,4),Beta(4,2) and Gamma(10,10), respectively.
#Skewness and kurtosis values of these variables are as follows:
n.C<-4
skewness.vec=c(2,0,-0.4677,0.6325)
kurtosis.vec=c(6,-0.5455,-0.3750,0.6)
coef.mat=fleishman.coef(n.C,skewness.vec,kurtosis.vec)

n.C<-1
skewness.vec=c(0)
kurtosis.vec=c(-1.2)
coef.mat=fleishman.coef(n.C,skewness.vec,kurtosis.vec)

n.C<-1
skewness.vec1=c(3)
kurtosis.vec1=c(5)
coef.mat=fleishman.coef(n.C,skewness.vec1,kurtosis.vec1)

## End(Not run)
```

`gen.PoisBinNonNor` *Simulates a sample of size n from a set of multivariate Poisson, binary, and continuous data*

Description

This function simulates a sample of size n from a set of multivariate Poisson, binary, and continuous data with pre-specified marginals and a correlation matrix.

Usage

```
gen.PoisBinNonNor(n, n.P, n.B, n.C, lambda.vec = NULL, prop.vec = NULL,
mean.vec=NULL, variance.vec=NULL, coef.mat = NULL, final.corr.mat)
```

Arguments

`n`	Number of variates.
`n.P`	Number of Poisson variables.
`n.B`	Number of binary variables.
`n.C`	Number of continuous variables.
`lambda.vec`	Rate vector for Poisson variables.
`prop.vec`	Proportion vector for binary variables.
`mean.vec`	Mean vector of continuous variables.
`variance.vec`	Variance vector of continuous variables.
`coef.mat`	Matrix of coefficients produced from `fleishman.coef`.
`final.corr.mat`	Final correlation matrix produced from `overall.corr.mat`

Value

A matrix of size n*(n.P + n.B + n.C), of which the first n.P columns are Poisson variables, the next n.B columns are binary variables, and the last n.C columns are continuous variables.

Examples

```
## Not run:
n=100000
n.P<-2
n.B<-2
n.C<-2
lambda.vec<-c(2,3)
prop.vec<-c(0.3,0.5)
mean.vec<-c(0,0)
variance.vec<-c(1,1)
coef.mat=matrix(rep(c(0,1,0,0), each=2),4,2,byrow=T)
corr.mat=matrix(0.4,6,6)
diag(corr.mat)=1
```

```
final.corr.mat=overall.corr.mat(n.P,n.B,n.C,lambda.vec,prop.vec,
coef.mat,corr.vec=NULL,corr.mat)

mymixdata=gen.PoisBinNonNor(n,n.P,n.B,n.C,lambda.vec,prop.vec,
mean.vec,variance.vec,coef.mat,final.corr.mat)

#Check marginals
#apply(mymixdata,2,mean)
#cor(mymixdata)

n=100000
n.P<-2
n.B<-2
n.N<-2
lambda.vec<-c(2,3)
prop.vec<-c(0.3,0.5)
mean.vec=c(1,0.5)
variance.vec=c(1,0.02777778)
skewness.vec=c(2,0)
kurtosis.vec=c(6,-0.5455)
coef.mat=fleishman.coef(2,skewness.vec, kurtosis.vec)
corr.mat=matrix(0.3,6,6)
diag(corr.mat)=1
final.corr.mat=overall.corr.mat(n.P,n.B,n.N,lambda.vec,prop.vec,
coef.mat,corr.vec=NULL,corr.mat)
mymixdata=gen.PoisBinNonNor(n,n.P,n.B,n.N,lambda.vec,prop.vec,
mean.vec, variance.vec,coef.mat,final.corr.mat)

#Check marginals
#apply(mymixdata,2,mean)[4:5]
#apply(mymixdata,2,var)[4:5]
#cor(mymixdata)

## End(Not run)
```

`intermediate.corr.BB` *Computes an intermediate normal correlation matrix for binary variables given the specified correlation matrix*

Description

Computes an intermediate normal correlation matrix for binary variables before dichotomization given the specified correlation matrix.

Usage

```
intermediate.corr.BB(n.P, n.B, n.C, prop.vec, corr.vec = NULL, corr.mat = NULL)
```

Arguments

n.P	Number of Poisson variables.
n.B	Number of binary variables.
n.C	Number of continuous variables.
prop.vec	Proportion vector for binary variables.
corr.vec	Vector of elements below the diagonal of correlation matrix ordered column-wise.
corr.mat	Specified correlation matrix.

Value

A correlation matrix of size n.B*n.B.

References

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

See Also

intermediate.corr.PB, intermediate.corr.BC

Examples

```
## Not run:
n.P<-2
n.B<-2
n.C<-2
prop.vec=c(0.4,0.7)
corr.vec = NULL
corr.mat=matrix(c(1.0,-0.3,-0.3,-0.3,-0.3,-0.3,
-0.3,1.0,-0.3,-0.3,-0.3,-0.3,
-0.3,-0.3,1.0,0.4,0.5,0.6,
-0.3,-0.3,0.4,1.0,0.7,0.8,
-0.3,-0.3,0.5,0.7,1.0,0.9,
-0.3,-0.3,0.6,0.8,0.9,1.0),6,by=TRUE)

intmatBB=intermediate.corr.BB(n.P,n.B,n.C,prop.vec,corr.vec=NULL,corr.mat)
intmatBB

## End(Not run)
```

intermediate.corr.BC *Computes intermediate correlation matrix for binary and continuous variables given the specified correlation matrix*

Description

This function computes the intermediate correlation matrix for binary-continuous combinations as formulated in Demirtas et al. (2012).

Usage

```
intermediate.corr.BC(n.P, n.B, n.C, lambda.vec = NULL, prop.vec = NULL,
coef.mat = NULL, corr.vec = NULL, corr.mat = NULL)
```

Arguments

`n.P`	Number of Poisson variables.
`n.B`	Number of binary variables.
`n.C`	Number of continuous variables.
`lambda.vec`	Rate vector for Poisson variables.
`prop.vec`	Proportion vector for binary variables.
`coef.mat`	Matrix of coefficients produced from `fleishman.coef`.
`corr.vec`	Vector of elements below the diagonal of correlation matrix ordered column-wise.
`corr.mat`	Specified correlation matrix.

Value

A correlation matrix of size n.B*n.C.

References

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

See Also

`intermediate.corr.BB`, `intermediate.corr.CC`

Examples

```
## Not run:
n.B<-2
n.C<-4
prop.vec=c(0.4,0.7)
coef.mat=matrix(c(
 -0.31375,  0.00000,  0.10045, -0.10448,
  0.82632,  1.08574,  1.10502,  0.98085,
  0.31375,  0.00000, -0.10045,  0.10448,
  0.02271, -0.02945, -0.04001,  0.00272),4,byrow=TRUE)
corr.vec = NULL
corr.mat=matrix(c(1.0,-0.3,-0.3,-0.3,-0.3,-0.3,
-0.3,1.0,-0.3,-0.3,-0.3,-0.3,
-0.3,-0.3,1.0,0.4,0.5,0.6,
-0.3,-0.3,0.4,1.0,0.7,0.8,
-0.3,-0.3,0.5,0.7,1.0,0.9,
-0.3,-0.3,0.6,0.8,0.9,1.0),6,byrow=TRUE)

intmatBC=intermediate.corr.BC(n.P=0,n.B,n.C,lambda.vec=NULL,prop.vec,coef.mat,
corr.vec=NULL,corr.mat)
intmatBC

n.B<-1
n.C<-1
prop.vec=0.6
coef.mat=matrix(c(-0.31375,0.82632,0.31375,0.02271),4,1)
corr.vec=NULL
corr.mat=matrix(c(1,-0.3,-0.3,1),2,2)

intmatBC=intermediate.corr.BC(n.P=0,n.B,n.C,lambda.vec=NULL,prop.vec,coef.mat,
corr.vec=NULL,corr.mat)
intmatBC

## End(Not run)
```

`intermediate.corr.CC` *Computes an intermediate correlation matrix for continuous variables given the specified correlation matrix*

Description

This function computes the intermediate correlation matrix for continuous-continuous combinations as formulated in Demirtas et al. (2012).

Usage

```
intermediate.corr.CC(n.P, n.B, n.C, coef.mat = NULL, corr.vec = NULL, corr.mat = NULL)
```

Arguments

n.P	Number of Poisson variables.
n.B	Number of binary variables.
n.C	Number of continuous variables.
coef.mat	Matrix of coefficients produced from fleishman.coef.
corr.vec	Vector of elements below the diagonal of correlation matrix ordered column-wise.
corr.mat	Specified correlation matrix.

Value

A correlation matrix of size n.C*n.C.

References

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

Vale, C.D. and Maurelli, V.A. (1983). Simulating multivariate nonnormal distributions. Psychometrika, 48(3), 465-471.

See Also

intermediate.corr.PC, intermediate.corr.BC

Examples

```
## Not run:
n.P=2
n.C=4
coef.mat=matrix(c(
 -0.31375,  0.00000,  0.10045, -0.10448,
  0.82632,  1.08574,  1.10502,  0.98085,
  0.31375,  0.00000, -0.10045,  0.10448,
  0.02271, -0.02945, -0.04001,  0.00272),4,byrow=TRUE)
corr.vec = NULL
corr.mat=matrix(c(1.0,-0.3,-0.3,-0.3,-0.3,-0.3,
-0.3,1.0,-0.3,-0.3,-0.3,-0.3,
-0.3,-0.3,1.0,0.4,0.5,0.6,
-0.3,-0.3,0.4,1.0,0.7,0.8,
-0.3,-0.3,0.5,0.7,1.0,0.9,
-0.3,-0.3,0.6,0.8,0.9,1.0),6,byrow=TRUE)

intmatCC=intermediate.corr.CC(n.P,n.B=0,n.C,coef.mat,corr.vec=NULL,corr.mat)
intmatCC

## End(Not run)
```

`intermediate.corr.PB` *Computes the pairwise entries of the intermediate normal correlation matrix for all Poisson-binary combinations given the specified correlation matrix.*

Description

This function computes the pairwise entries of the intermediate normal correlation matrix for all Poisson-binary combinations given the specified correlation matrix as formulated in Amatya and Demirtas (2015).

Usage

```
intermediate.corr.PB(n.P, n.B, n.C, lambda.vec = NULL, prop.vec = NULL,
coef.mat = NULL, corr.vec = NULL, corr.mat = NULL)
```

Arguments

`n.P`	Number of Poisson variables.
`n.B`	Number of binary variables.
`n.C`	Number of continuous variables.
`lambda.vec`	Rate vector for Poisson variables.
`prop.vec`	Proportion vector for binary variables.
`coef.mat`	Matrix of coefficients produced from `fleishman.coef`.
`corr.vec`	Vector of elements below the diagonal of correlation matrix ordered column-wise.
`corr.mat`	Specified correlation matrix.

Value

A matrix of n.P*n.B.

References

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

See Also

`intermediate.corr.PP`, `intermediate.corr.BB`

Examples

```
## Not run:
n.P<-2
n.B<-1
lambda.vec<-c(2,3)
prop.vec<-c(0.3)
corr.mat=matrix(c(1,0.2,0.1,0.2,1,0.5,0.1,0.5,1),3,3)

intmatPB=intermediate.corr.PB(n.P,n.B,n.C=0,lambda.vec,prop.vec,coef.mat=NULL,
corr.vec=NULL,corr.mat)
intmatPB

## End(Not run)
```

`intermediate.corr.PC` *Computes the pairwise entries of the intermediate normal correlation matrix for all Poisson-continuous combinations given the specified correlation matrix.*

Description

This function computes the pairwise entries of the intermediate normal correlation matrix for all Poisson-continuous combinations given the specified correlation matrix as formulated in Amatya and Demirtas (2015).

Usage

```
intermediate.corr.PC(n.P, n.B, n.C, lambda.vec = NULL, prop.vec = NULL, coef.mat = NULL,
corr.vec = NULL, corr.mat = NULL)
```

Arguments

`n.P`	Number of Poisson variables.
`n.B`	Number of binary variables.
`n.C`	Number of continuous variables.
`lambda.vec`	Rate vector for Poisson variables.
`prop.vec`	Proportion vector for binary variables.
`coef.mat`	Matrix of coefficients produced from `fleishman.coef`.
`corr.vec`	Vector of elements below the diagonal of correlation matrix ordered column-wise.
`corr.mat`	Specified correlation matrix.

Value

A correlation matrix of size n.P*n.C.

References

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

See Also

intermediate.corr.PP, intermediate.corr.CC

Examples

```
## Not run:
n.P=2
n.C=4
lambda.vec=c(2,3)
coef.mat=matrix(rep(c(0,1,0,0),each=4),4,byrow=TRUE)
corr.vec = NULL
corr.mat=matrix(c(1.0,-0.3,-0.3,-0.3,-0.3,-0.3,
-0.3,1.0,-0.3,-0.3,-0.3,-0.3,
-0.3,-0.3,1.0,0.4,0.5,0.6,
-0.3,-0.3,0.4,1.0,0.7,0.8,
-0.3,-0.3,0.5,0.7,1.0,0.9,
-0.3,-0.3,0.6,0.8,0.9,1.0),6,byrow=TRUE)

intmatPC=intermediate.corr.PC(n.P,n.B=0,n.C,lambda.vec,prop.vec=NULL,
coef.mat,corr.vec=NULL,corr.mat)

intmatPC

#See also cmat.star in  R package PoisNor
#cmat.star(no.pois=2,no.norm=4,corMat=corr.mat,lamvec=lambda.vec)

## End(Not run)
```

---

intermediate.corr.PP *Computes an intermediate normal correlation matrix for Poisson variables given the specified correlation matrix*

Description

This function computes the intermediate normal correlation matrix for Poisson-Poisson combinations before inverse cdf matching as formulated in Amatya and Demirtas (2015).

Usage

```
intermediate.corr.PP(n.P, n.B, n.C, lambda.vec, corr.vec = NULL, corr.mat = NULL)
```

Arguments

`n.P`	Number of Poisson variables.
`n.B`	Number of binary variables.
`n.C`	Number of continuous variables.
`lambda.vec`	Rate vector for Poisson variables
`corr.vec`	Vector of elements below the diagonal of correlation matrix ordered column-wise.
`corr.mat`	Specified correlation matrix.

Value

A correlation matrix of size n.P*n.P.

References

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

See Also

`intermediate.corr.PB`, `intermediate.corr.PC`

Examples

```
n.P<-3
lambda.vec<-c(1,2,3)
corr.mat<-matrix(c(1,0.352,0.265,0.352,1,0.121,0.265,0.121,1),n.P,n.P)
intmatPP=intermediate.corr.PP(n.P,n.B=0,n.C=0,lambda.vec,corr.vec=NULL,corr.mat)
intmatPP

## Not run:
#See also cmat.star in  R package PoisNor
#cmat.star(no.pois=3,no.norm=0,corMat=corr.mat,lamvec=lambda.vec)

## End(Not run)
```

`overall.corr.mat` *Computes the final intermediate correlation matrix*

Description

This function computes the final correlation matrix by combining pairwise intermediate correlation matrix entries for Poisson-Poisson, Poisson-binary, Poisson-continuous, binary-binary, binary-continuous, and continuous-continuous combinations. If the resulting correlation matrix is not positive definite, a nearest positive matrix will be used.

Usage

```
overall.corr.mat(n.P, n.B, n.C, lambda.vec = NULL, prop.vec = NULL, coef.mat = NULL,
corr.vec = NULL, corr.mat = NULL)
```

Arguments

`n.P`	Number of Poisson variables.
`n.B`	Number of binary variables.
`n.C`	Number of continuous variables.
`lambda.vec`	Rate vector for Poisson variables.
`prop.vec`	Proportion vector for binary variables.
`coef.mat`	Matrix of coefficients produced from `fleishman.coef`.
`corr.vec`	Vector of elements below the diagonal of correlation matrix ordered column-wise.
`corr.mat`	Specified correlation matrix.

Value

A correlation matrix of size (n.P+N.B+n.C)*(n.P+N.B+n.C).

See Also

`intermediate.corr.PP`, `intermediate.corr.BB`, `intermediate.corr.CC`,

`intermediate.corr.PB`, `intermediate.corr.PC`, `intermediate.corr.BC`

Examples

```
## Not run:
n.P<-1
n.B<-1
n.C<-1
lambda.vec<-c(1)
prop.vec<-c(0.3)
coef.mat<-matrix(c(0,1,0,0),4,1)
corr.vec=NULL
corr.mat=matrix(c(1,0.2,0.1,0.2,1,0.5,0.1,0.5,1),3,3)

finalmat=overall.corr.mat(n.P,n.B,n.C,lambda.vec,prop.vec,coef.mat,
corr.vec=NULL,corr.mat)
finalmat

## End(Not run)
```

validation.bin — *Validates the marginal specification of the binary variables*

Description

Checks whether the marginal specification of the binary part is valid and consistent.

Usage

```
validation.bin(n.B, prop.vec = NULL)
```

Arguments

n.B	Number of binary variables.
prop.vec	Proportion vector for binary variables.

Value

The function returns TRUE if no specification problem is encountered. Otherwise, it returns an error message.

Examples

```
n.B<-3
prop.vec<-c(0.25,0.5,0.75)
validation.bin(n.B, prop.vec)

## Not run:
n.B<-3
validation.bin(n.B)

n.B<--3
prop.vec<-c(0.25,0.5,0.75)
validation.bin(n.B, prop.vec)

n.B<-0
prop.vec<-c(0.25,0.5,0.75)
validation.bin(n.B, prop.vec)

n.B<-5
prop.vec<-c(0.25,0.5,0.75)
validation.bin(n.B, prop.vec)

n.B<-3
prop.vec<-c(0.25,0.5,-0.75)
validation.bin(n.B, prop.vec)

## End(Not run)
```

`validation.corr` *Validates the specified correlation matrix*

Description

This function validates the specified correlation vector and/or matrix for appropriate dimension, symmetry, range, and positive definiteness. If both correlation matrix and correlation vector are supplied, it checks whether the matrix and vector are conformable.

Usage

```
validation.corr(n.P, n.B, n.C, corr.vec = NULL, corr.mat = NULL)
```

Arguments

`n.P`	Number of Poisson variables.
`n.B`	Number of binary variables.
`n.C`	Number of continuous variables.
`corr.vec`	Vector of elements below the diagonal of correlation matrix ordered column-wise.
`corr.mat`	Specified correlation matrix.

Value

The function returns TRUE if no specification problem is encountered. Otherwise, it returns an error message.

See Also

`correlation.limits`, `correlation.bound.check`

Examples

```
n.P<-1
n.B<-1
n.C<-1
corr.vec=c(0.2,0.1,0.5)
validation.corr(n.P,n.B,n.C,corr.vec,corr.mat=NULL)

n.P<-2
n.B<-2
n.C<-2
corr.mat=matrix(0.5,6,6)
diag(corr.mat)=1
validation.corr(n.P,n.B,n.C,corr.vec=NULL,corr.mat)

## Not run:
n.P<-2
```

```
n.B<-2
n.C<-1
corr.mat=matrix(0.5,6,6)
diag(corr.mat)=1
validation.corr(n.P,n.B,n.C,corr.vec=NULL,corr.mat)

n.P<-2
n.B<-2
n.C<-2
corr.mat=matrix(0.5,6,6)
corr.mat[1,2]=0.4
diag(corr.mat)=1
validation.corr(n.P,n.B,n.C,corr.vec=NULL,corr.mat)

## End(Not run)
```

`validation.skewness.kurtosis`
Validates the marginal specification of the continuous variables

---

Description

Checks whether the marginal specification of the continuous part is valid and consistent.

Usage

```
validation.skewness.kurtosis(n.C, skewness.vec = NULL, kurtosis.vec = NULL)
```

Arguments

`n.C`	Number of continuous variables.
`skewness.vec`	Skewness vector for continuous variables.
`kurtosis.vec`	Kurtosis vector for continuous variables.

Value

The function returns TRUE if no specification problem is encountered. Otherwise, it returns an error message.

References

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

Examples

```
n.C<-3
skewness.vec=c(0,2,3)
kurtosis.vec=c(-1.2,6,8)
validation.skewness.kurtosis(n.C,skewness.vec,kurtosis.vec)

## Not run:
n.C<--1
skewness.vec=c(0)
kurtosis.vec=c(-1.2)
validation.skewness.kurtosis(n.C,skewness.vec,kurtosis.vec)

n.C<-3
skewness.vec=c(0,2,3)
kurtosis.vec=c(-1.2,6,5)
validation.skewness.kurtosis(3)

n.C<-3
skewness.vec=c(0,2,3)
kurtosis.vec=c(-1.2,6,5)
validation.skewness.kurtosis(n.C,skewness.vec)
validation.skewness.kurtosis(n.C,kurtosis.vec)

n.C<-0
skewness.vec=c(0,2,3)
kurtosis.vec=c(-1.2,6,8)
validation.skewness.kurtosis(n.C,skewness.vec,kurtosis.vec)

n.C<-2
skewness.vec=c(0,2,3)
kurtosis.vec=c(-1.2,6,8)
validation.skewness.kurtosis(n.C,skewness.vec,kurtosis.vec)

n.C<-2
skewness.vec=c(0,2,3)
kurtosis.vec=c(-1.2,6)
validation.skewness.kurtosis(n.C,skewness.vec,kurtosis.vec)

skewness.vec=c(2,3)
kurtosis.vec=c(1,5)
validation.skewness.kurtosis(n.C,skewness.vec,kurtosis.vec)

## End(Not run)
```

Index