Package 'RandomCorMatrix'

November 14, 2025

Type Package

Title Random Correlation Matrix

Version 0.1.0

Author Harry Joe, Dorota Kurowicka

Maintainer Harry Joe <Harry.Joe@ubc.ca>

Description Generate random correlation matrices with correlations in (0,1) or (-1,1), with goal of specifying one moment or two moments.

License GPL-3

Depends R (>= 3.3)

Encoding UTF-8

LazyData true

R topics documented:

`beta_mom2ab` *Beta distributions : method of moments estimation*

Description

Beta(a,b) on (0,1) or (-1,1): map from moments (mean, SD) to (a,b)

Usage

```
beta_mom2ab(mu,s,imp1=FALSE)
```

Arguments

`mu`	mean parameter, could be sample mean of data
`s`	SD parameter, could be sample SD of data
`imp1`	indicator of minus/plus1, TRUE if interval is (-1,1), FALSE if (0,1). Note that s^2<1-mu^2 for (-1,1) and s^2<mu*(1-mu) for (0,1)

Value

list with (a,b,a+b) where a,b come from method of moments estimation

Examples

```
beta_mom2ab(0.4,0.2,imp1=TRUE)
beta_mom2ab(0.4,0.2,imp1=FALSE)
```

`corvec2mat` *Convert from correlations in vector form to a correlation matrix*

Description

Convert from correlations in vector form to a correlation matrix

Usage

```
corvec2mat(rvec)
```

Arguments

`rvec`	correlations in vector form of length d*(d-1)/2 in the order r12,r13,r23,r14,... r[d-1,d]

Value

dxd correlation matrix

Examples

```
rvec = c(0.3,0.4,0.5,0.4,0.6,0.7)
Rmat = corvec2mat(rvec)
print(Rmat) # column 1 has 1, 0.3, 0.4, 0.4
# example from rposcorr_2mom which outputs one triangle of correlation matrix
a1 = 4.0; b1 = 8.0; d = 4
out2pos = rposcorr_2mom(3,d,a1,b1,123,1)
colnames(out2pos) = c("r12","r13","r23","r14","r24","r34")
print(out2pos)
print(corvec2mat(out2pos[1,]))
print(corvec2mat(out2pos[2,]))
```

`get2betapars_mp1` *Recursion for a[ell], b[ell] of Beta random variables in (-1,1) for partial correlations in row/tree ell starting from Beta(a1,b1) distribution in (-1,1) for correlations in row 1 of C-vine, so that all correlations have mean and SD of a Beta(a1,b1) random variable in (-1,1)*

Description

Get (a[ell],b[ell]) for Beta parameters of random partial correlations in tree ell=2,...,K, starting from Beta(a1,b1) in (-1,1) in row 1 of C-vine, for the goal to have constant E(R_[ell,ell+1]) and E(R^2_[ell,ell+1]) by C-vine tree

Usage

```
get2betapars_mp1(a1,b1,K, iprint=FALSE)
```

Arguments

`a1`	Beta parameter for correlations in C-vine tree 1
`b1`	Beta parameter for correlations in C-vine tree 1
`K`	integer >=2, simulate random correlation matrices to dimension K K=2 get Beta parameter for R_[23;1], K=3 get Beta parameters for R_[23;1], R_[34;12], K general: get Beta parameters for R_[23;1],...,R_[K,K+1;1...K-1]
`iprint`	TRUE for intermediate prints at tree levels

Value

Kx2 matrix with Beta parameters avec,bvec; feasibility to row i, where i<=K; avec[(i+1):K]=0 and bvec[(i+1):K]=0 if not feasible

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines. Journal of Multivariate Analysis. https://doi.org/10.1016/j.jmva.2025.105519

Examples

```
ab = get2betapars_mp1(a1=2,b1=2, 3, iprint=TRUE); print(ab)
ab = get2betapars_mp1(a1=1,b1=1, 3, iprint=TRUE) # fails at 3
ab = get2betapars_mp1(a1=1,b1=0.5, 3, iprint=TRUE) # fails at R_{34}
ab = get2betapars_mp1(a1=1,b1=2, 3, iprint=TRUE) # fails at R_{34}
ab = get2betapars_mp1(a1=5,b1=5, 6, iprint=TRUE)
ab = get2betapars_mp1(a1=5,b1=5, 10, iprint=TRUE)
ab = get2betapars_mp1(a1=5,b1=5, 12, iprint=TRUE) # fails at 11
ab = get2betapars_mp1(a1=8,b1=5, 6, iprint=TRUE); print(ab)
ab = get2betapars_mp1(a1=5,b1=8, 6, iprint=TRUE)  # fails at 4
ab = get2betapars_mp1(a1=6,b1=4.8, 6, iprint=TRUE)
ab = get2betapars_mp1(a1=8,b1=6.4, 6, iprint=TRUE)
```

`get2betapars_pos` *Recursion for a[ell], b[ell] of Beta random variables in (0,1) for partial correlations in row/tree ell starting from positive Beta(a1,b1) distribution for correlations in row 1 of C-vine, so that all correlations have mean and SD of a Beta(a1,b1) random variable in (0,1)*

Description

Get (a[ell],b[ell]) for Beta parameters of random partial correlations in tree ell=2,...,K, starting from Beta(a1,b1) in (0,1) in row 1 of C-vine, for the goal to have constant E(R_[ell,ell+1]) and E(R^2_[ell,ell+1]) by C-vine tree

Usage

```
get2betapars_pos(a1,b1,K, iprint=FALSE)
```

Arguments

`a1`	Beta parameter for correlations in C-vine tree 1
`b1`	Beta parameter for correlations in C-vine tree 1
`K`	integer >=2, simulate random correlation matrices to dimension K K=2 get Beta parameter for R_[23;1], K=3 get Beta parameters for R_[23;1], R_[34;12], K general: get Beta parameters for R_[23;1],...,R_[K,K+1;1...K-1]
`iprint`	TRUE for intermediate prints at tree levels

Value

Kx2 matrix with Beta parameters avec,bvec; feasibility to row i, where i<=K; avec[(i+1):K]=0 and bvec[(i+1):K]=0 if not feasible

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines. Journal of Multivariate Analysis. https://doi.org/10.1016/j.jmva.2025.105519

Examples

```
ab = get2betapars_pos(a1=2,b1=2, 3, iprint=TRUE); print(ab)
ab = get2betapars_pos(a1=1,b1=1, 3, iprint=TRUE); print(ab)
ab = get2betapars_pos(a1=1,b1=0.5, 3, iprint=TRUE) # fails for R_{34}
ab = get2betapars_pos(a1=1,b1=2, 3, iprint=TRUE)
ab = get2betapars_pos(a1=0.5,b1=0.5, 3, iprint=TRUE) # fails for R_{34}
ab = get2betapars_pos(a1=5,b1=5, 6, iprint=TRUE)
ab = get2betapars_pos(a1=5,b1=5, 10, iprint=TRUE)
ab = get2betapars_pos(a1=5,b1=5, 14, iprint=TRUE)
ab = get2betapars_pos(a1=5,b1=8, 6, iprint=TRUE)
```

`getbetapars_mp1` *With a[ell]=a1, recursion for second parameter b[ell] of Beta random variables in (-1,1) for partial correlations in row/tree ell starting from Beta(a1,b) distribution in (-1,1) for correlations in row 1 of C-vine, so that all correlations have mean 2*a1/(a1+b)-1*

Description

Get mu[ell] and b[ell], with a[ell]=a1, for Beta parameters of random partial correlations in row/tree ell=2,...,K, starting from Beta(a1,b) in (-1,1) in row 1, for the goal that all correlations have mean 2*a1/(a1+b)-1

Usage

```
getbetapars_mp1(a1,b,K,iprint=FALSE)
```

Arguments

`a1`	first parameter Beta(a1,b) on (-1,1) for correlation in tree/row 1
`b`	second parameter Beta(a1,b) on (-1,1) for correlation in tree/row 1
`K`	desired level: simulate random correlation matrices to dimension K
`iprint`	TRUE for intermediate prints at tree levels

Value

(muvec,bvec) as 2-column matrix: muvec is vector of mean parameters for random partial correlations in trees 2 to ell, ell<K if K is not feasible; bvec is vector of second Beta parameter if considering Beta(a1,bvec[ell]) for partial correlation in tree ell

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines. Journal of Multivariate Analysis. https://doi.org/10.1016/j.jmva.2025.105519

Examples

```
tem1 = getbetapars_mp1(a1=3,b=2,K=8,iprint=TRUE)
tem2 = getbetapars_mp1(a1=4,b=2,K=8,iprint=TRUE)
tem3 = getbetapars_mp1(a1=6,b=2,K=8,iprint=TRUE)
```

`getbetapars_pos` *With a[ell]=a1, recursion for second parameter b[ell] of Beta random variables in (0,1) for partial correlations in row/tree ell starting from positive Beta(a1,b) distribution for correlations in row 1 of C-vine, so that all correlations have mean a1/(a1+b)*

Description

Get mu[ell] and b[ell], with a[ell]=a1, for Beta parameters of random partial correlations in row/tree ell=2,...,K, starting from Beta(a1,b) in (0,1) in row 1, for the goal that all correlations have mean a1/(a1+b)

Usage

```
getbetapars_pos(a1,b,K,iprint=FALSE)
```

Arguments

a1	first parameter Beta(a1,b) on (0,1) for correlation in tree/row 1
b	second parameter Beta(a1,b) on (0,1) for correlation in tree/row 1
K	desired level: simulate random correlation matrices to dimension K
iprint	TRUE for intermediate prints at tree levels

Value

(muvec,bvec) as 2-column matrix: muvec is vector of mean parameters for random partial correlations in trees 2 to ell, ell<K if K is not feasible; bvec is vector of second Beta parameter if considering Beta(a1,bvec[ell]) for positive partial correlation in tree ell. For the method of allowing some negative partial correlations, use muvec and not bvec.

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines. Journal of Multivariate Analysis. https://doi.org/10.1016/j.jmva.2025.105519

Examples

```
tem1 = getbetapars_pos(a1=1,b=1,K=6,iprint=TRUE)
tem2 = getbetapars_pos(a1=1,b=0.5,K=6,iprint=TRUE)
tem3 = getbetapars_pos(a1=1,b=0.36,K=6,iprint=TRUE) # increasing bvec
tem4 = getbetapars_pos(a1=1,b=0.35,K=6,iprint=TRUE)  # bvec decreasing to 0
tem4b = getbetapars_pos(a1=1,b=0.35,K=7,iprint=TRUE)  # message mu_new>=1
tem5 = getbetapars_pos(a1=1,b=0.35,K=12,iprint=TRUE) # stops at 7
tem6 = getbetapars_pos(a1=2,b=0.32,K=12,iprint=TRUE) # stops at 4
```

---

moments_mp1 *For R=B with B~Beta(a,b) on (-1,1), get 4 moments*

Description

For R=B with B~Beta(a,b) on (-1,1), get 4 moments

Usage

```
moments_mp1(a,b)
```

Arguments

a	Beta distribution first parameter
b	Beta distribution second parameter

Value

6-vector (a, b, mu, gam, nu, eta) where mu=E(B), gam=E[sqrt(1-B^2)], nu=E(B^2), eta=E[B*sqrt(1-B^2)]

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines Journal of Multivariate Analysis

---

moments_pos *For R=B with B~Beta(a,b) on (0,1), get 4 moments*

Description

For R=B with B~Beta(a,b) on (0,1), get 4 moments

Usage

```
moments_pos(a,b)
```

Arguments

a	Beta distribution first parameter
b	Beta distribution second parameter

Value

6-vector (a, b, mu, gam, nu, eta) where mu=E(B), gam=E[sqrt(1-B^2)], nu=E(B^2), eta=E[B*sqrt(1-B^2)]

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines Journal of Multivariate Analysis

---

pcor2cor_cvine *Generate correlation matrix based on partial correlation C-vine*

Description

Generate correlation matrix based on partial correlation C-vine

Usage

```
pcor2cor_cvine(pc)
```

Arguments

pc	dxd array with partial correlation rho_[jk;1:(j-1)] in position (j,k) for j<k

Value

dxd correlation matrix

Examples

```
# See help pages for rcormat_posC and rcormat_mp1C
```

qqbeta *QQ plot for Beta(a,b) distribution on (0,1) or (-1,1)*

Description

QQ plot for Beta(a,b) distribution on (0,1) or (-1,1)

Usage

```
qqbeta(x,a,b,pos1=0,pos2=0)
```

Arguments

x	data vector on (0,1) or (-1,1)
a	alpha parameter>0
b	beta parameter>0
pos1	first subscript of correlation
pos2	second subscript of correlation

Value

a QQ plot is produced, 0 is returned

Examples

```
set.seed(1234)
n = 1000
x = rbeta(n,3,2)
mn = mean(x); s = sd(x)
ab = beta_mom2ab(mn,s)
qqbeta(x,ab[1],ab[2])
set.seed(12343)
n = 1000
x = 2*rbeta(n,5,3)-1
mn = mean(x); s = sd(x)
ab = beta_mom2ab(mn,s,imp1=TRUE)
qqbeta(x,ab[1],ab[2])
```

rcormat_mp1C *Generate dxd correlation matrix with partial correlation C-vine.*

Description

Random correlation matrix via partial correlation C-vine

Usage

```
rcormat_mp1C(d, avec, bvec, ipermute=TRUE)
```

Arguments

d	dimension >=3
avec	vector of dimension d-1, first parameter of Beta(a_ell,b_ell) in (-1,1) by tree 1 to d-1
bvec	vector of dimension d-1, second parameter Beta distribution in (-1,1) by tree 1 to d-1
ipermute	permute flag, if TRUE random row/column permutation is applied

Value

dxd positive correlation matrix, if TRUE row/column permutation is applied See rmp1corr_1mom and rmp1corr_2mom to generate many correlation matrices via Fortran90

Examples

```
# correlations in (-1,1) given mu
d = 4
mat = matrix(1:d^2,d,d)
iupper = upper.tri(mat)
a1 = 3; b1 = 2
mubobj = getbetapars_mp1(a1,b1,d-1,iprint=FALSE)
print(mubobj)
avec = rep(a1,d-1)
bvec = mubobj[,'bvec']
nsim = 500
set.seed(1234)
out1_mp1 = matrix(0,nsim,d*(d-1)/2)
for(isim in 1:nsim)
{ rr = rcormat_mp1C(d,avec,bvec,ipermute=TRUE)
  out1_mp1[isim,] = rr[iupper]
}
print(rr)
colnames(out1_mp1) = c("r12","r13","r23","r14","r24","r34")
print(summary(out1_mp1))
print(apply(out1_mp1,2,sd))
# mp1 given 2 moments
```

```
a1 = 8; b1 = 5
mubobj = get2betapars_mp1(a1,b1,d-1,iprint=FALSE)
print(mubobj)
avec = mubobj[,'a']
bvec = mubobj[,'b']
nsim = 500
set.seed(1234)
out2_mp1 = matrix(0,nsim,d*(d-1)/2)
for(isim in 1:nsim)
{ rr = rcormat_mp1C(d,avec,bvec,ipermute=TRUE)
  out2_mp1[isim,] = rr[iupper]
}
colnames(out2_mp1) = c("r12","r13","r23","r14","r24","r34")
print(summary(out2_mp1))
print(apply(out2_mp1,2,sd))
```

`rcormat_posC` *Generate dxd correlation matrix with all positive correlations via positive partial correlation C-vine.*

Description

Random positive correlation matrix via positive partial correlation C-vine

Usage

```
rcormat_posC(d, avec, bvec, ipermute=TRUE)
```

Arguments

d	dimension >=3
avec	vector of dimension d-1, first parameter of Beta(a_ell,b_ell) by tree 1 to d-1
bvec	vector of dimension d-1, second parameter Beta distribution by tree 1 to d-1
ipermute	permute flag, if TRUE random row/column permutation is applied

Value

dxd positive correlation matrix;See rposcorr_1mom and rposcorr_2mom to generate many correlation matrices via Fortran90

Examples

```
# positive given mu
d = 4
mat = matrix(1:d^2,d,d)
iupper = upper.tri(mat)
a1 = 1; b1 = 3
mubobj = getbetapars_pos(a1,b1,d-1,iprint=FALSE)
```

```
print(mubobj)
avec = rep(a1,d-1)
bvec = mubobj[,'bvec']
nsim = 100
set.seed(1234)
out1_pos = matrix(0,nsim,d*(d-1)/2)
for(isim in 1:nsim)
{ rr = rcormat_posC(d,avec,bvec,ipermute=TRUE)
  out1_pos[isim,] = rr[iupper]
}
print(rr)
colnames(out1_pos) = c("r12","r13","r23","r14","r24","r34")
print(summary(out1_pos))
print(apply(out1_pos,2,sd))
# positive given 2 moments
a1 = 4; b1 = 8
mubobj = get2betapars_pos(a1,b1,d-1,iprint=FALSE)
print(mubobj)
avec = mubobj[,'a']
bvec = mubobj[,'b']
nsim = 100
set.seed(1234)
out2_pos = matrix(0,nsim,d*(d-1)/2)
for(isim in 1:nsim)
{ rr = rcormat_posC(d,avec,bvec,ipermute=TRUE)
  out2_pos[isim,] = rr[iupper]
}
colnames(out2_pos) = c("r12","r13","r23","r14","r24","r34")
print(summary(out2_pos))
print(apply(out2_pos,2,sd))
```

`recursion_mp1` *Recursion for mean and second moment for random correlations in (-1,1) by tree of C-vine*

Description

Recursion for mean and second moment for random correlations in (-1,1) by tree

Usage

```
recursion_mp1(avec, bvec, iprint=FALSE)
```

Arguments

`avec`	positive vector length d-1>=2; avec[ell] is first Beta parameter for partial correlation in tree ell
`bvec`	positive vector length d-1>=2; bvec[ell] is second Beta parameter for partial correlation in tree ell
`iprint`	TRUE for intermediate prints

Value

matrix with columns a, b, ER, ER2, SD, ER_aperm, SD_aperm with row length = length(bvec); aperm means after random permutation. The output can be used for correlation matrices up to length(bvec)

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines Journal of Multivariate Analysis

Examples

```
out = recursion_mp1(avec=rep(1,11), bvec=seq(1,6,0.5), iprint=TRUE)
```

---

recursion_pos *Recursion for mean and second moment for random correlations in (0,1) by tree of C-vine*

Description

Recursion for mean and second moment for random correlations in (0,1) by tree

Usage

```
recursion_pos(avec, bvec, iprint=FALSE)
```

Arguments

avec	positive vector length d-1>=2; avec[ell] is first Beta parameter for partial correlation in tree ell
bvec	positive vector length d-1>=2; bvec[ell] is second Beta parameter for partial correlation in tree ell
iprint	TRUE for intermediate prints

Value

matrix with columns a, b, ER, ER2, SD, ER_aperm, SD_aperm with row length = length(bvec); aperm means after random permutation. The output can be used for correlation matrices up to length(bvec)

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines Journal of Multivariate Analysis

Examples

```
out = recursion_pos(avec=rep(1,11), bvec=seq(1,6,0.5), iprint=TRUE)
bb = c(1,1.467401,2.062294,2.748234,3.496715,4.288931,5.112821,5.96039)
out = recursion_pos(avec=rep(1,8), bvec=bb, iprint=TRUE)
```

`rmp1corr_1mom` *Generate random correlation matrices with C-vine partial correlations, fixed mean*

Description

Generate random correlation matrices with C-vine partial correlations with aim of a fixed mean for all correlations

Usage

```
rmp1corr_1mom(nsim,d,a1,b,seed,ipermute=TRUE)
```

Arguments

`nsim`	number of simulations
`d`	dimension of correlation matrix, >=3
`a1`	first parameter of Beta(a1,b) random variable in (-1,1) for row 1
`b`	second parameter of Beta(a1,b) random variable in (-1,1) for row 1; aim is for mean correlation value of 2*a1/(a1+b)-1
`seed`	positive integer seed for fortran90 code
`ipermute`	permute flag, TRUE (default) for random row/column permutation

Details

Loop is done through link to Fortran90 code

Value

If input (d,a1,b) is feasible, output matrix nsim by d(d-1)/2,each row has a correlation matrix with lower triangle by row; r12, r13, r23, r14, ..., r[d-1,d]. If input (d,a1,b) is infeasible, output is 0.

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines. Journal of Multivariate Analysis. https://doi.org/10.1016/j.jmva.2025.105519

Examples

```
 # failed case
rmp1corr_1mom(5,6,2.,3.0,123)
# good case
a1 = 3.0; b = 2.0; d = 4
out1mp1 = rmp1corr_1mom(1000,d,a1,b,123,1)
print(apply(out1mp1,2,summary))
print(apply(out1mp1,2,sd))
mubobj = getbetapars_mp1(a1,b,d-1,iprint=FALSE)
avec = rep(a1,d-1)
bvec = mubobj[,'bvec']
recur = recursion_mp1(rep(a1,d-1),mubobj[,'bvec'])
print(recur)
s = recur[d-1,7]  # theoretical SD after permutation
mn = recur[d-1,6]
cat("theoretical mean/s", mn,s,"\n")
```

`rmp1corr_2mom` *Generate random correlation matrix with C-vine partial correlations, with fixed mean and SD*

Description

Generate random correlation matrix with C-vine partial correlations with aim of a fixed mean and a fixed SD for all correlations

Usage

```
rmp1corr_2mom(nsim,d,a1,b1,seed,ipermute=TRUE)
```

Arguments

`nsim`	number of simulations
`d`	dimension of correlation matrix, >=3
`a1`	first parameter of Beta(a1,b1) random variable in (-1,1) for row 1
`b1`	second parameter of Beta(a1,b1) random variable in (-1,1) for row 1; aim is for mean correlation value of 2*mu0-1 and variance 4*mu0*(1-mu0)/(a1+b1+1) where mu0 = a1/(a1+b1)
`seed`	positive integer seed for fortran90 code
`ipermute`	permute flag, TRUE (default) for random row/column permutation

Details

Loop is done through link to Fortran90 code

Value

If input (d,a1,b1) is feasible, output matrix nsim by d(d-1)/2,each row has a correlation matrix with lower triangle by row; r12, r13, r23, r14, ..., r[d-1,d]. If input (d,a1,b1) is infeasible, output is 0.

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines. Journal of Multivariate Analysis. https://doi.org/10.1016/j.jmva.2025.105519

Examples

```
 # failed case
rmp1corr_2mom(5,4,1.0,0.6,123)
# good case
a1 = 8.0; b1 = 5.0; d = 4
out2mp1 = rmp1corr_2mom(1000,4,a1,b1,123,1)
print(apply(out2mp1,2,summary))
print(apply(out2mp1,2,sd))
thmn = a1/(a1+b1)
thvar = thmn*(1-thmn)/(a1+b1+1)
cat("theoretical mean/s", 2*thmn-1, 2*sqrt(thvar),"\n")
colnames(out2mp1) = c("r12","r13","r23","r14","r24","r34")
rcor = cor(out2mp1)
print(round(rcor,3))
par(mfrow=c(3,3))
qqbeta(out2mp1[,1],a1,b1,1,2)
qqbeta(out2mp1[,2],a1,b1,1,3)
qqbeta(out2mp1[,3],a1,b1,2,3)
qqbeta(out2mp1[,4],a1,b1,1,4)
qqbeta(out2mp1[,5],a1,b1,2,4)
qqbeta(out2mp1[,6],a1,b1,3,4)
```

`rposcorr_1mom` *Generate random correlation matrices with all positive values based on C-vine with positive partial correlations, fixed mean*

Description

Generate random correlation matrices with all positive values based on C-vine with positive partial correlations, with aim of a fixed mean for all correlations

Usage

```
rposcorr_1mom(nsim,d,a1,b,seed,ipermute=TRUE)
```

Arguments

nsim	number of simulations
d	dimension of correlation matrix, >=3
a1	first parameter of Beta(a1,b) random variable in (0,1) for row 1
b	second parameter of Beta(a1,b) random variable in (0,1) for row 1; aim is for mean correlation value of a1/(a1+b)
seed	positive integer seed for fortran90 code
ipermute	permute flag, TRUE (default) for random row/column permutation

Details

Loop is done through link to Fortran90 code

Value

If input (d,a1,b) is feasible, output matrix nsim by d(d-1)/2,each row has a correlation matrix with lower triangle by row; r12, r13, r23, r14, ..., r[d-1,d]. If input (d,a1,b) is infeasible, output is 0.

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines. Journal of Multivariate Analysis. https://doi.org/10.1016/j.jmva.2025.105519

Examples

```
 # failed case
rposcorr_1mom(5,4,1.0,0.3,123)
# good case
a1 = 1.5; b = 0.5; d = 6
out1pos = rposcorr_1mom(1000,d,a1,b,123,1)
print(apply(out1pos,2,summary))
print(apply(out1pos,2,sd))
mubobj = getbetapars_pos(a1,b,d-1,iprint=FALSE)
print(mubobj)
avec = rep(a1,d-1)
bvec = mubobj[,'bvec']
recur = recursion_pos(rep(a1,d-1),mubobj[,'bvec'])
print(recur)
s = recur[d-1,7]  # theoretical SD after permutation
mn = recur[d-1,6]
cat("theoretical mean/s", mn,s,"\n")
```

rposcorr_2mom *Generate random correlation matrices with all positive values based on C-vine with positive partial correlations, fixed mean and SD*

Description

Generate random correlation matrices with all positive values based on C-vine with positive partial correlations, with aim of a fixed mean and a fixed SD for all correlations

Usage

```
rposcorr_2mom(nsim,d,a1,b1,seed,ipermute=TRUE)
```

Arguments

nsim	number of simulations
d	dimension of correlation matrix, >=3
a1	first parameter of Beta(a1,b1) random variable in (0,1) for row 1
b1	second parameter of Beta(a1,b1) random variable in (0,1) for row 1; aim is for mean correlation value of mu=a1/(a1+b1) and variance mu*(1-mu)/(a1+b1+1)
seed	positive integer seed for fortran90 code
ipermute	permute flag, TRUE (default) for random row/column permutation

Details

Loop is done through link to Fortran90 code

Value

If input (d,a1,b1) is feasible, output matrix nsim by d(d-1)/2,each row has a correlation matrix with lower triangle by row; r12, r13, r23, r14, ..., r[d-1,d]. If input (d,a1,b1) is infeasible, output is 0.

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines. Journal of Multivariate Analysis. https://doi.org/10.1016/j.jmva.2025.105519

Examples

```
 # failed case
rposcorr_2mom(5,4,1.0,0.5,123)
# good case
a1 = 4.0; b1 = 8.0; d = 4
out2pos = rposcorr_2mom(1000,d,a1,b1,123,1)
print(apply(out2pos,2,summary))
print(apply(out2pos,2,sd))
thmn = a1/(a1+b1)
```

```
thvar = thmn*(1-thmn)/(a1+b1+1)
cat("theoretical mean/s", thmn,sqrt(thvar),"\n")
colnames(out2pos) = c("r12","r13","r23","r14","r24","r34")
rcor = cor(out2pos)
print(round(rcor,3))
par(mfrow=c(3,3))
qqbeta(out2pos[,1],a1,b1,1,2)
qqbeta(out2pos[,2],a1,b1,1,3)
qqbeta(out2pos[,3],a1,b1,2,3)
qqbeta(out2pos[,4],a1,b1,1,4)
qqbeta(out2pos[,5],a1,b1,2,4)
qqbeta(out2pos[,6],a1,b1,3,4)
```

`rposcorr_method2` *Generate random correlation matrix with all positive values based on C-vine with distribution of partial correlations conditional on previous trees*

Description

Generate random correlation matrix with all positive values based on C-vine with distribution of partial correlations conditional on previous trees

Usage

```
rposcorr_method2(nsim,d,a1,b1,seed,afix=0,ipermute=TRUE)
```

Arguments

`nsim`	number of simulations
`d`	dimension of correlation matrix, >=4
`a1`	first parameter of Beta(a1,b1) random variable in (0,1) for row 1
`b1`	second parameter of Beta(a1,b1) random variable in (0,1) for row 1
`seed`	positive integer seed for fortran90 code
`afix`	fixed first parameter for Beta random variables in trees 2 to d-1, default will set afix=a1
`ipermute`	permute flag, TRUE (default) for random row/column permutation

Details

Loop is done through link to Fortran90 code

Value

If input (d,a1,b1) is feasible, output matrix nsim by d(d-1)/2,each row has a correlation matrix with lower triangle by row; r12, r13, r23, r14, ..., r[d-1,d]. Rows with r[d-1,d]>=1 are rejection cases; do not keep them. Use a small nsim value to check if acceptance rate is high. If input (d,a1,b1) is infeasible, output is 0.

References

Joe and Kurowicka (2026), Random correlation matrices generated via partial correlation C-vines. Journal of Multivariate Analysis. https://doi.org/10.1016/j.jmva.2025.105519

Examples

```
 d = 5; a1 = 2; b1 = 3; seed = 123; dd = d*(d-1)/2
out5 = rposcorr_method2(nsim=500,d,a1,b1,seed)
colnames(out5) = c("r12","r13","r23","r14","r24","r34","r15","r25","r35","r45")
iok = which(out5[,dd]<1)
out5 = out5[iok,]
print(dim(out5))
print(apply(out5,2,summary))
print(apply(out5,2,sd))
thmn = a1/(a1+b1)
thvar = thmn*(1-thmn)/(a1+b1+1)
cat("If high acceptance rate, theoretical mean/s", thmn,sqrt(thvar),"\n")
rcor = cor(out5)
print(round(rcor,3))
par(mfrow=c(3,3))
qqbeta(out5[,1],a1,b1,1,2)
qqbeta(out5[,2],a1,b1,1,3)
qqbeta(out5[,3],a1,b1,2,3)
qqbeta(out5[,4],a1,b1,1,4)
qqbeta(out5[,5],a1,b1,2,4)
qqbeta(out5[,6],a1,b1,3,4)
qqbeta(out5[,7],a1,b1,1,5)
qqbeta(out5[,8],a1,b1,2,5)
qqbeta(out5[,9],a1,b1,3,5)
```

Index