NORTA

The function

The arguments of the function are:

- N - number of datapoints to simulate
- marginals - a character vector with the name of the marginal distributions (see ?distributions)
- parms - a list of lists containing the parameters of the marginals distributions
- Sigma_X - the desired correlation matrix Σ_X of the simulated datapoints.
- warning - a logical value controlling whether to output a warning message if a correction of the intermediate correlation matrix was made
- message - a logical value controlling whether to print the feasible region for each pair of marginals
- xmax,xmin,ymax,ymin,tol - arguments passed to the numerical integration function corresponding to the limits of the doble integral and the tolerance considered.

The output of the function:

- A list containing the simulated dataset, the intermediate correlation matrix (Σ_Z) and the Cholesky decomposition (M in the paper)

Independent case

Let's see if the algorithm works in the most trivial case (when there is no correlation).

```
source("NORTA.R")
set.seed(1234)

Sigma_X = matrix(rep(0,4),2,2)
diag(Sigma_X) = 1
marginals = c("binom","unif")
parms = list(m1 = list(size = 10, prob = 0.2),
             m2 = list(min=0,max=1))

data = NORTA(2e5,marginals,Sigma_X,parms,-8,8,-8,8)
```

```
## The feasible interval for marginal 1  and marginal 2  is:
##  [ -0.78678 , 0.99621 ]
```

The correlation matrix obtained for binomial and uniform marginals:

```
cor(data$data)
```

```
##              [,1]        [,2]
## [1,] 1.000000000 0.008331768
## [2,] 0.008331768 1.000000000
```

The correlation is near 0!, let's see which Σ_Z we obtain and compare it with R's random number generation functions

```
data$Sigma_Z
```

```
##           [,1]     [,2]
## [1,] 1.000000 0.014942
## [2,] 0.014942 1.000000
```

```
a = runif(2e5,0,1)
b = rbinom(2e5, size = 10, prob = 0.2)

cor(a,b)
```

```
## [1] 0.00458002
```

Now let's see if both marginals are continous (uniform and gamma)

```
Sigma_X = matrix(rep(0,4),2,2)
diag(Sigma_X) = 1
marginals = c("gamma","unif")
parms = list(m1 = list( shape= 1, scale = 0.3),
             m2 = list(min=0,max=1))

data = NORTA(2e5,marginals,Sigma_X,parms,-8,8,-8,8)
```

```
## The feasible interval for marginal 1  and marginal 2  is:
##  [ -0.52226 , 0.90749 ]
```

```
cor(data$data)
```

```
##              [,1]         [,2]
## [1,] 1.0000000000 0.0009327956
## [2,] 0.0009327956 1.0000000000
```

```
data$Sigma_Z
```

```
##               [,1]          [,2]
## [1,]  1.000000e+00 -3.921671e-07
## [2,] -3.921671e-07  1.000000e+00
```

The function also works quite well!, In GhoshAnderson2003 PDF the authors state that for 2 dimensional random vectors, NORTA works exremely well, and even for dimension 3.

An example with n = 4

Let's simulate a sample from a 4-valued random vector with normal, uniform, binomial and beta marginals.

```
marginals = c("norm", "unif","binom","beta")
parms = list(m1 = list(mean = 1,sd = 3),
             m2 = list(min=-1,max=2),
             m3 = list(size = 10, prob = 0.2),
             m4 = list(shape1 = 0.5, shape2 = 2,ncp = 0.5))

Sigma_X = matrix(c(1.0,0.6,0.4,0.2,
                   0.6,1.0,0.3,0.22,
                   0.4,0.3,1.0,0.2,
                   0.2,0.22,0.2,1.0),4,4)
Sigma_X
```

```
##      [,1] [,2] [,3] [,4]
## [1,]  1.0 0.60  0.4 0.20
## [2,]  0.6 1.00  0.3 0.22
## [3,]  0.4 0.30  1.0 0.20
## [4,]  0.2 0.22  0.2 1.00
```

```
data = NORTA(2e5,marginals, Sigma_X, parms,xmin = -8, xmax = 8, ymin = -8, ymax = 8,message = T)
```

```
## The feasible interval for marginal 1  and marginal 2  is:
##  [ -1.62538 , 0.9852 ]
## The feasible interval for marginal 1  and marginal 3  is:
##  [ -0.9778 , 0.98761 ]
## The feasible interval for marginal 2  and marginal 3  is:
##  [ -0.94592 , 0.96819 ]
## The feasible interval for marginal 1  and marginal 4  is:
##  [ -0.94885 , 0.95113 ]
## The feasible interval for marginal 2  and marginal 4  is:
##  [ -1.61709 , 0.95435 ]
## The feasible interval for marginal 3  and marginal 4  is:
##  [ -0.83514 , 0.98224 ]
```

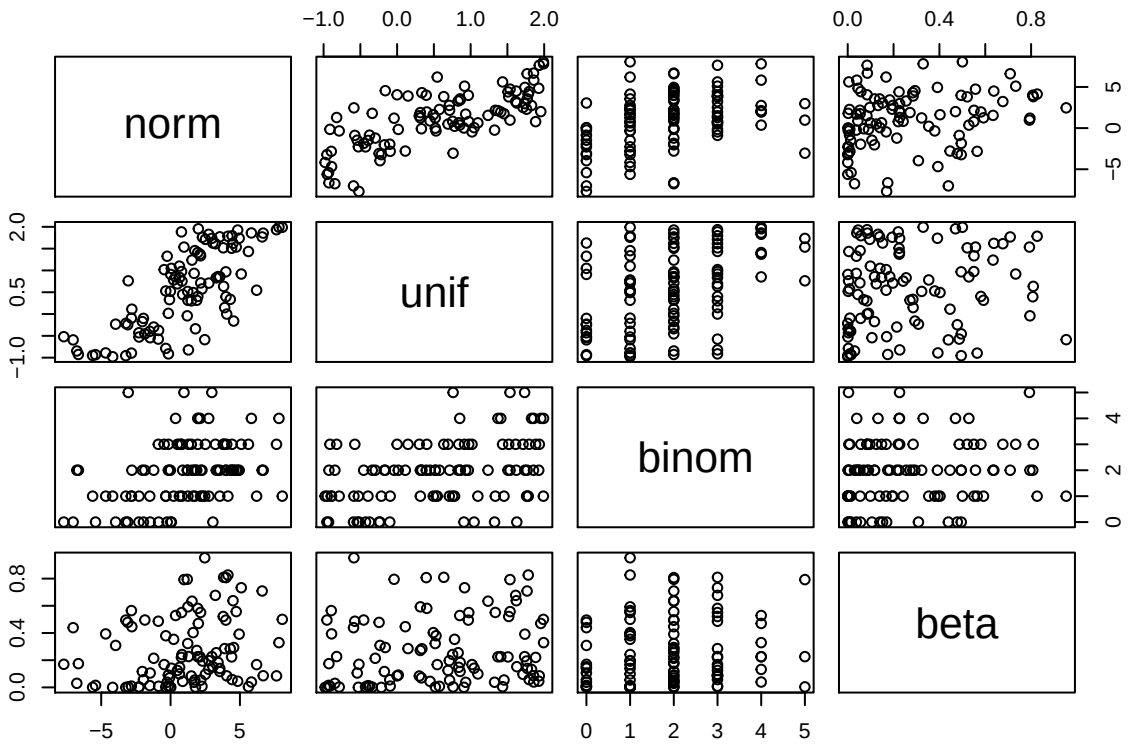
```
colnames(data$data) = marginals

cor(data$data)
```

```
##            norm      unif     binom      beta
## norm  1.0000000 0.6008276 0.3833506 0.2044810
## unif  0.6008276 1.0000000 0.2740214 0.2202436
## binom 0.3833506 0.2740214 1.0000000 0.1758480
## beta  0.2044810 0.2202436 0.1758480 1.0000000
```

```
err = cor(data$data) - Sigma_X
err
```

```
##                norm          unif       binom          beta
## norm   0.0000000000  0.0008276203 -0.01664945  0.0044810140
## unif   0.0008276203  0.0000000000 -0.02597861  0.0002436437
## binom -0.0166494475 -0.0259786111  0.00000000 -0.0241519917
## beta   0.0044810140  0.0002436437 -0.02415199  0.0000000000
```

```
pairs(data$data[sample(1:1e4,100),])
```



We can observe that the estimates of the feasibility interval for some pairs of marginals is not computed efficiently. This could lead to errors if the desired correlation matrix has entries that are close to ± 1

Some comments

Only the default distributions in R are available, if you need to implement more distributions you need to hardcode and add to the NORTA.R script. Consider this if you have in mind to simulate from empirical cdfs.