Van Der Corput seried and quasi random directions

X. Freulon

2022-11-02

Initialisation

```
rm(list=ls())
library("gstlearn")
```

The Van de Corput sequence

The van der Corput sequence is the simplest one-dimensional low-discrepancy sequence over the unit interval; it was first described in 1935 by the Dutch mathematician J. G. van der Corput. It is constructed by reversing the base-p representation of the sequence of natural numbers (1, 2, 3, ...).

The p-ary representation of the positive integer $n(\geq 1)$ is

$$n = \sum_{k=0}^{L-1} d_k(n)\, p^k$$

where p is the base in which the number n is represented, and $0 \leq d_p(n) < p$, i.e. the k-th digit in the p-ary expansion of n. The n-th number in the van der Corput sequence is

$$g_p(n) = \sum_{k=0}^{L-1} d_k(n)\, \frac{1}{p^{k+1}}$$

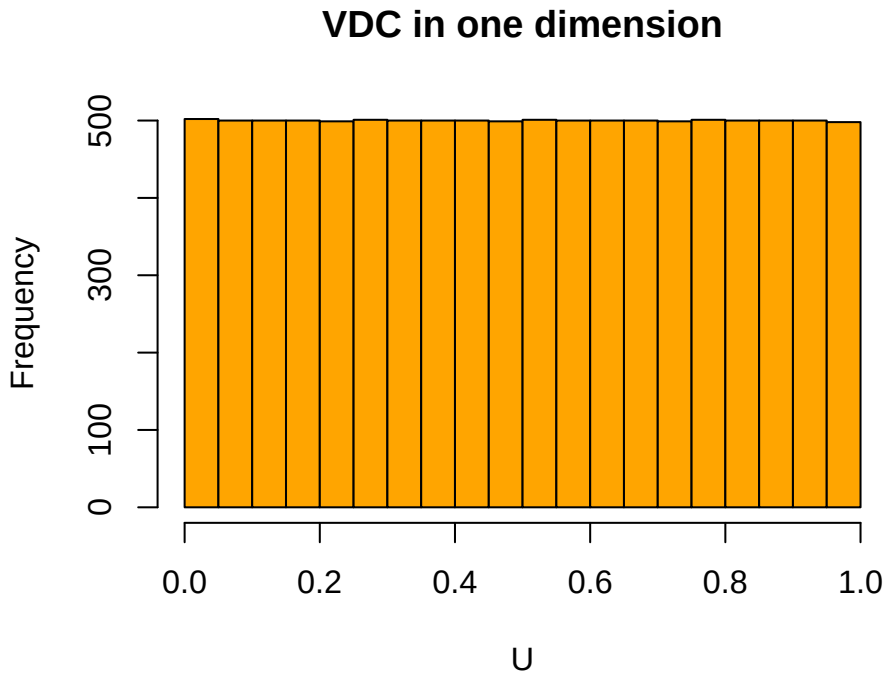
Evaluation of the Van der Corput sequence

A single dimension

```
U = vanDerCorput(n = 10000, nd=1)$getColumn(0)
summary(U)
```

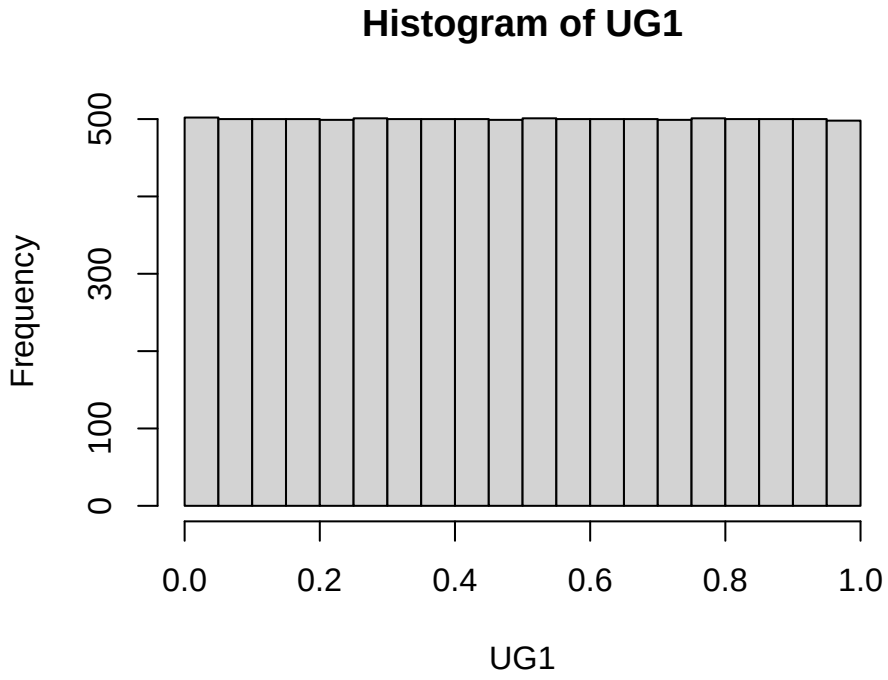
```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.000061 0.249970 0.499939 0.499833 0.749908 0.999878
```

```
hist(U, col = "orange", main = "VDC in one dimension")
```

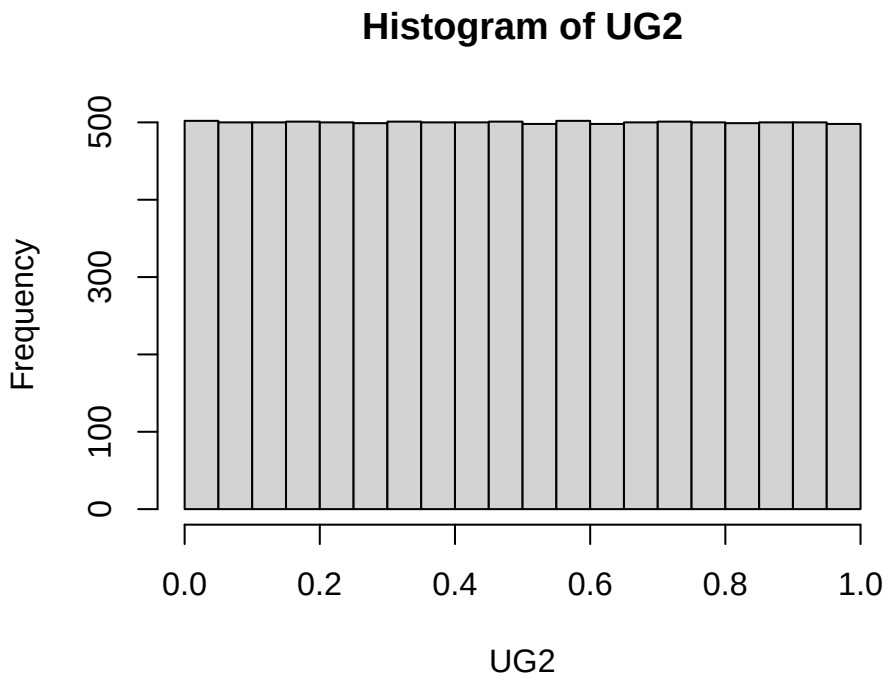


Two dimensions

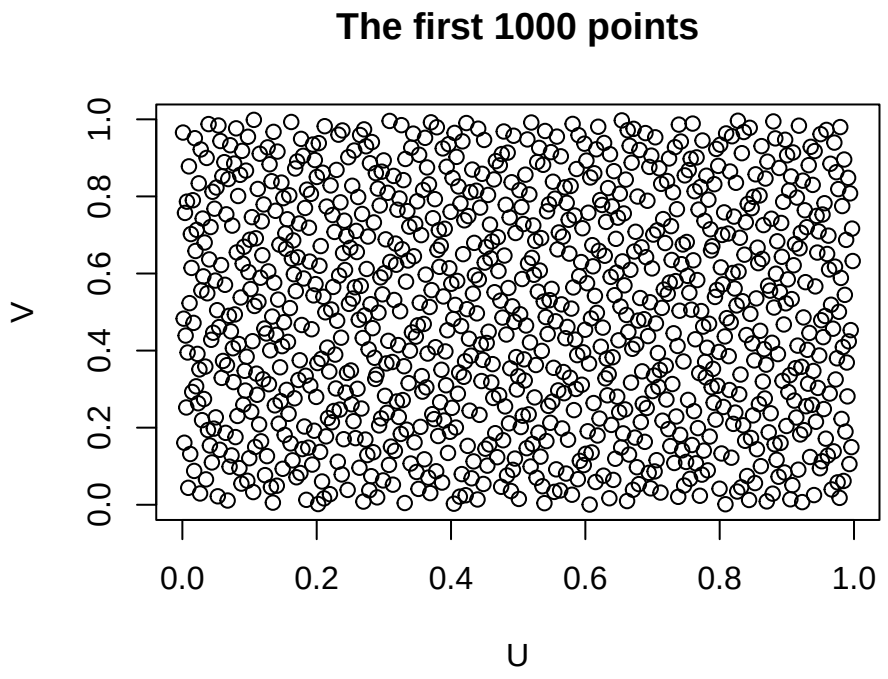
```
UG <- vanDerCorput(n = 10000, nd = 2)
UG1 = UG$getColumn(0)
UG2 = UG$getColumn(1)

hist(UG1)
```



```
hist(UG2)
```



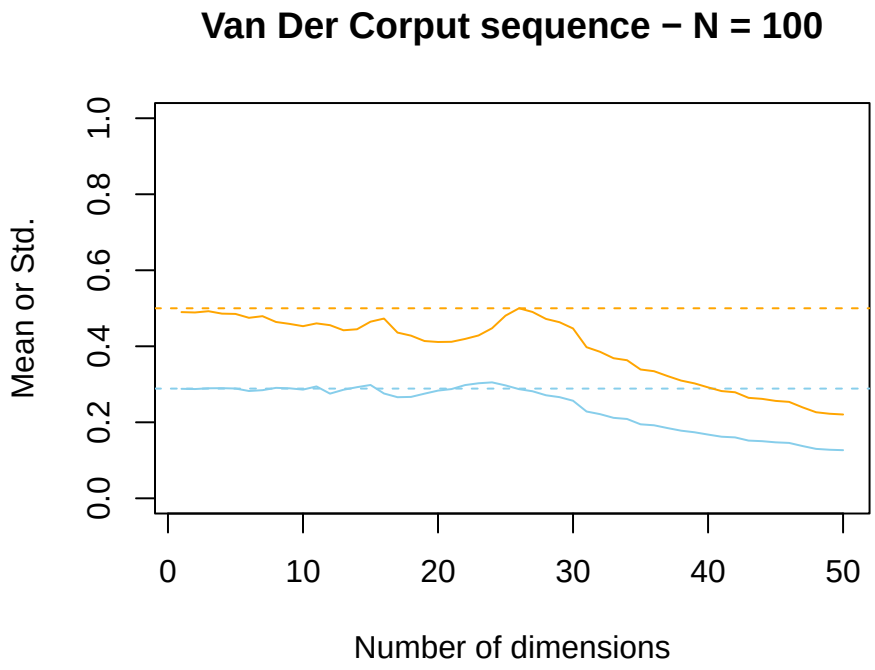

```
plot(UG1[1:1000], UG2[1:1000],
     xlab = "U", ylab = "V", main = "The first 1000 points")
```



Many dimensions and few points

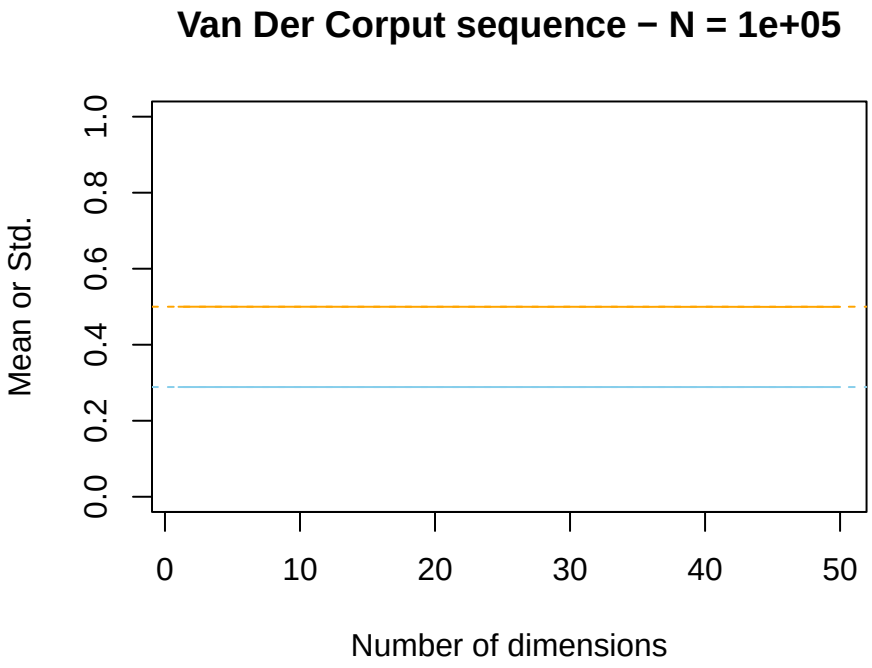
```
np <- 100
nd <- 50
Umat <- vanDerCorput(n = np, nd = nd)
U = matrix(Umat$getValues(),nrow=Umat$getNRows())
```

```
means <- apply(X = U, MARGIN = 2, FUN = mean)
stds  <- apply(X = U, MARGIN = 2, FUN = sd)
plot(NULL, NULL, xlim = c(1,50), ylim = c(0,1),
     xlab = "Number of dimensions",
     ylab = "Mean or Std.", main = paste0("Van Der Corput sequence - N = ", np))
abline(h = 0.5, col = "orange", lty = 2)
abline(h = sqrt(1/12), col = "skyblue", lty = 2)
lines(1:50, means, lty = 1, col = "orange")
lines(1:50, stds , lty = 1, col = "skyblue")
```



Many dimensions and many points

```
np <- 100000
nd <- 50
Umat <- vanDerCorput(n = np, nd = nd)
U = matrix(Umat$getValues(),nrow=Umat$getNRows())
means <- apply(X = U, MARGIN = 2, FUN = mean)
stds  <- apply(X = U, MARGIN = 2, FUN = sd)
plot(NULL, NULL, xlim = c(1,50), ylim = c(0,1),
     xlab = "Number of dimensions",
     ylab = "Mean or Std.", main = paste0("Van Der Corput sequence - N = ", np))
abline(h = 0.5, col = "orange", lty = 2)
abline(h = sqrt(1/12), col = "skyblue", lty = 2)
lines(1:50, means, lty = 1, col = "orange")
lines(1:50, stds , lty = 1, col = "skyblue")
```



n-sphere and (quasi) random directions

The n-sphere is defined as

$$S^n = \{s \in \mathbb{R}^{n+1} : ||x|| = 1\}$$

and a direction in $\mathbb{R}^{n+1}$ is a point on the half n-sphere.

A simple approach to generating a uniform point on S^n uses the fact that the multivariate normal distribution with independent standardized components is radially symmetric, i.e., it is invariant under orthogonal rotations. Therefore, if $Y \sim \mathcal{N}(\mathbf{0}_{n+1}, \mathbf{I}_{n+1})$, then $S_n = Y/||Y||$ has the uniform distribution on the unit n-sphere.

For the simulation of a direction, i.e. a point on S^n_+, the last axis is selected as a reference and points with a negative coordinate along this axis are replaced by their symmetric points relatively to the origin.

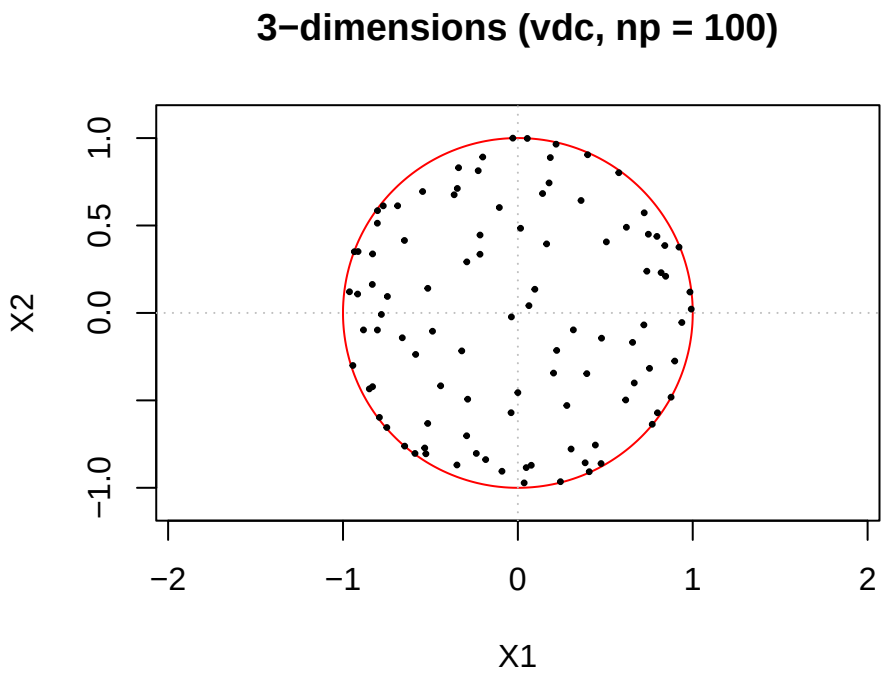
Finally, in order to build a quasi random values on the half n-sphere, the normal variables are simulated using the inverse method and the pseudo random generator on $[0,1]^{n+1}$ is replaced by the Van der Corput sequence which is a quasi random generator on $[0,1]^{n+1}$.

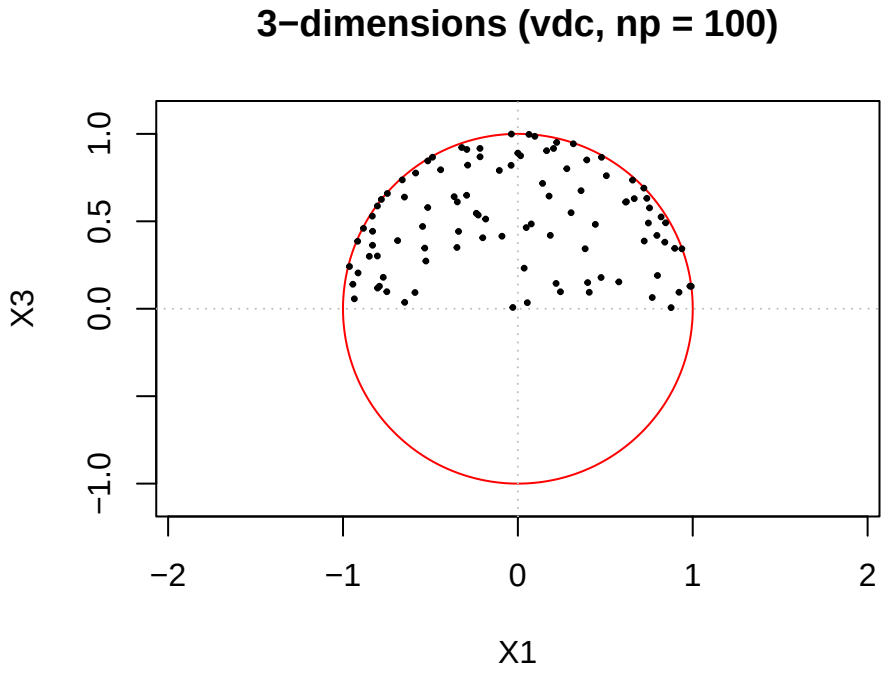
Test in two-dimensions

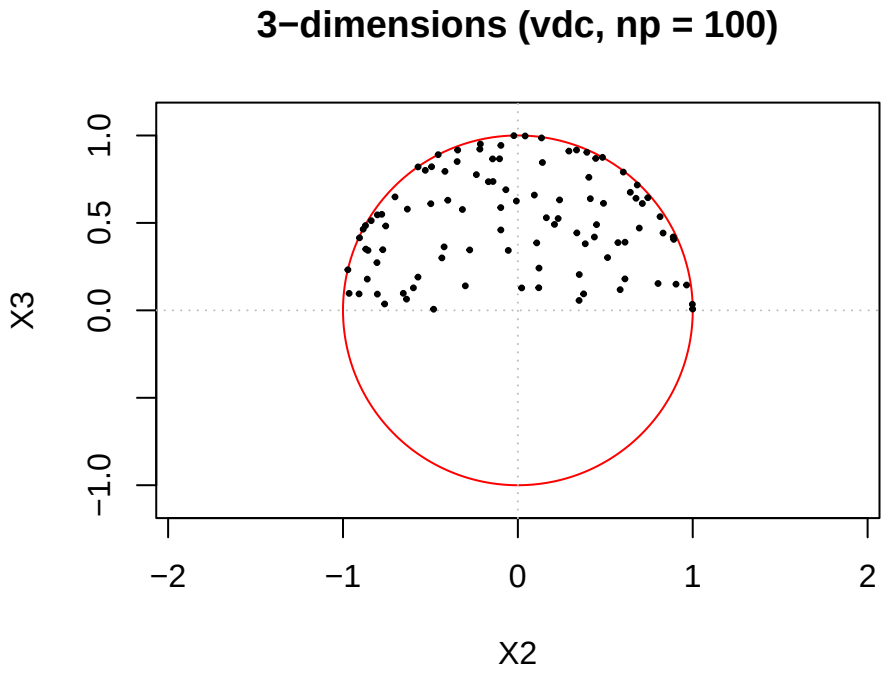
```
phi <- 2*pi*(0:1000)/1001
cs <- cos(phi)
sn <- sin(phi)
```

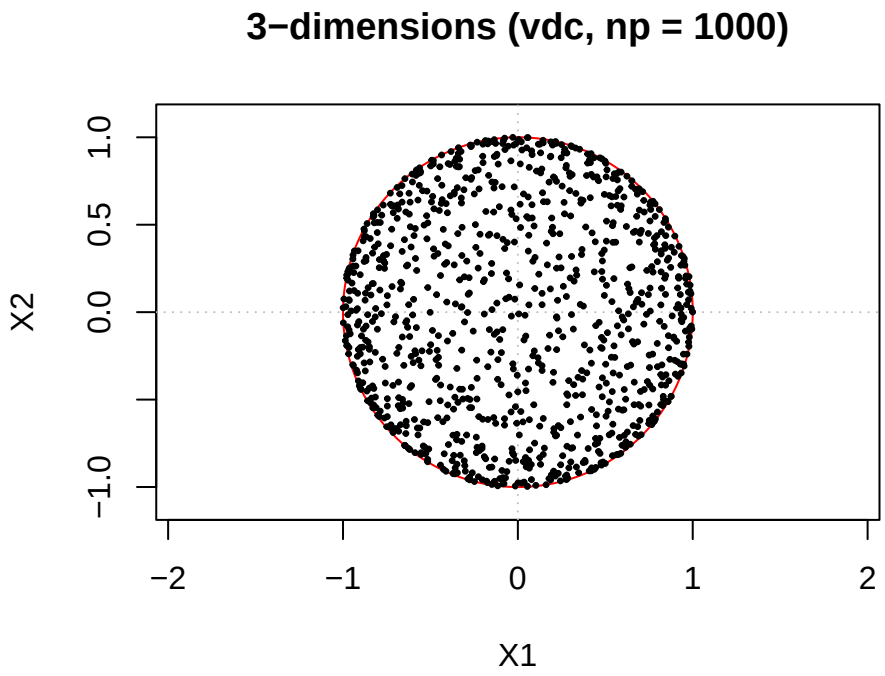
```
# Directions in R^3
nd <- 3
for (meth in c("vdc", "prng")) {
  for (np in c(100, 1000, 10000)){
    for (ix in 1:(nd-1)) {
      for (iy in (ix+1):nd){
        plot_dir(np, nd = nd, meth = meth, ix = ix, iy = iy)
      }
```

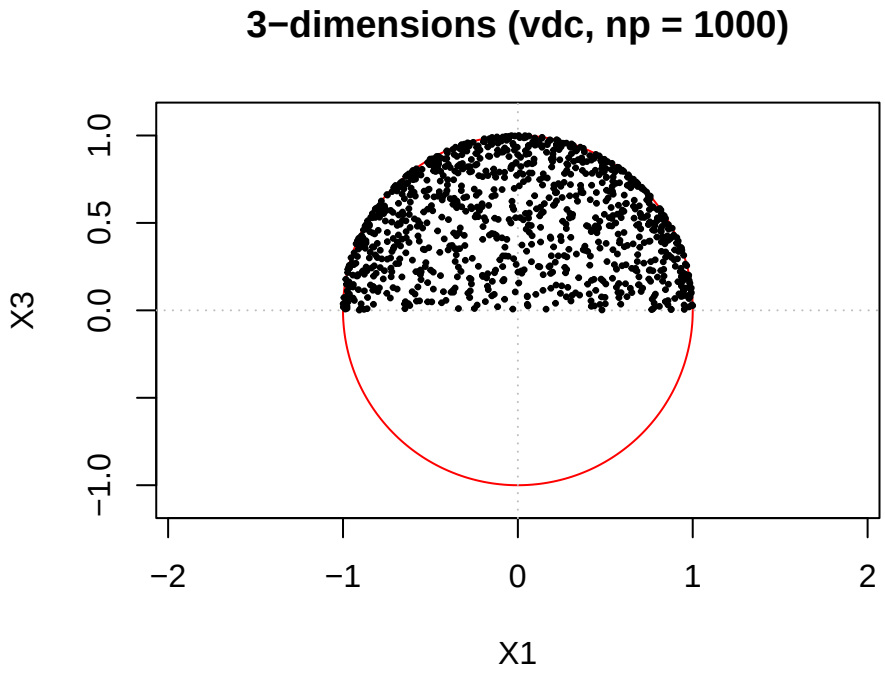
```
    }
  }
}
```

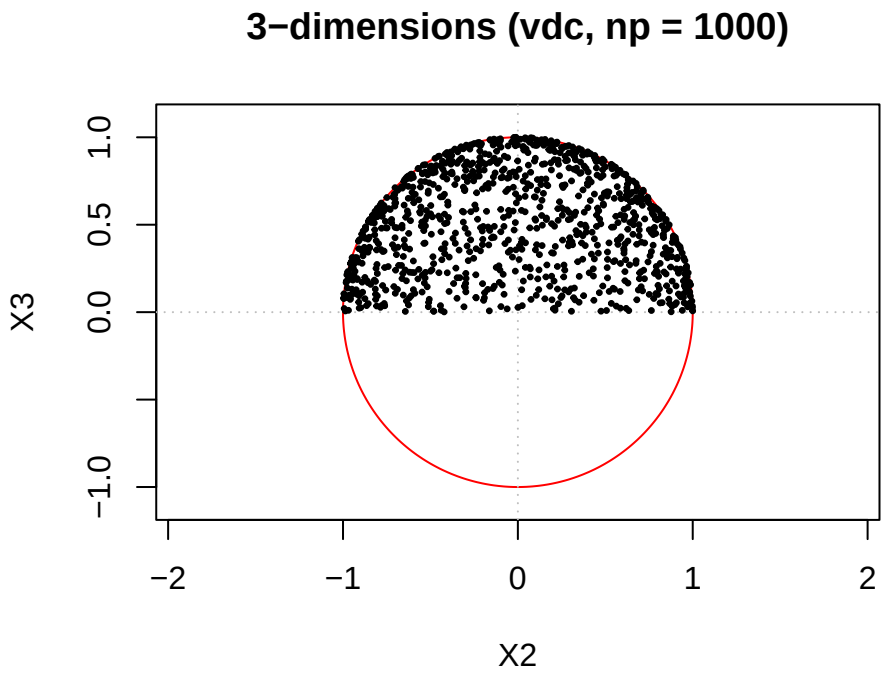

3−dimensions (vdc, np = 100)
X2
X1
1.0
0.5
0.0
−1.0
−2
−1
0
1
2

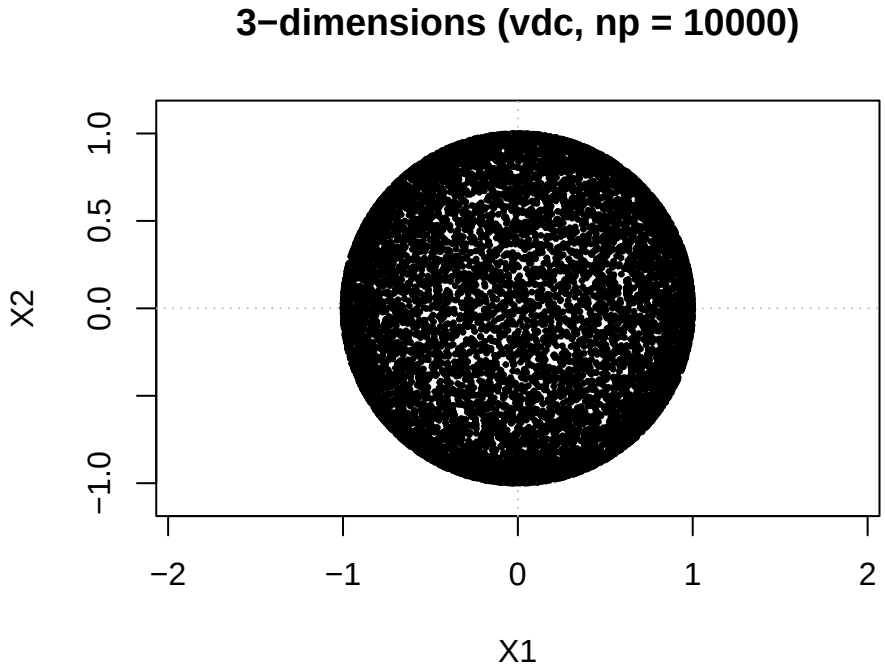

3−dimensions (vdc, np = 100)
X3
X1
1.0
0.5
0.0
−1.0
−2
−1
0
1
2

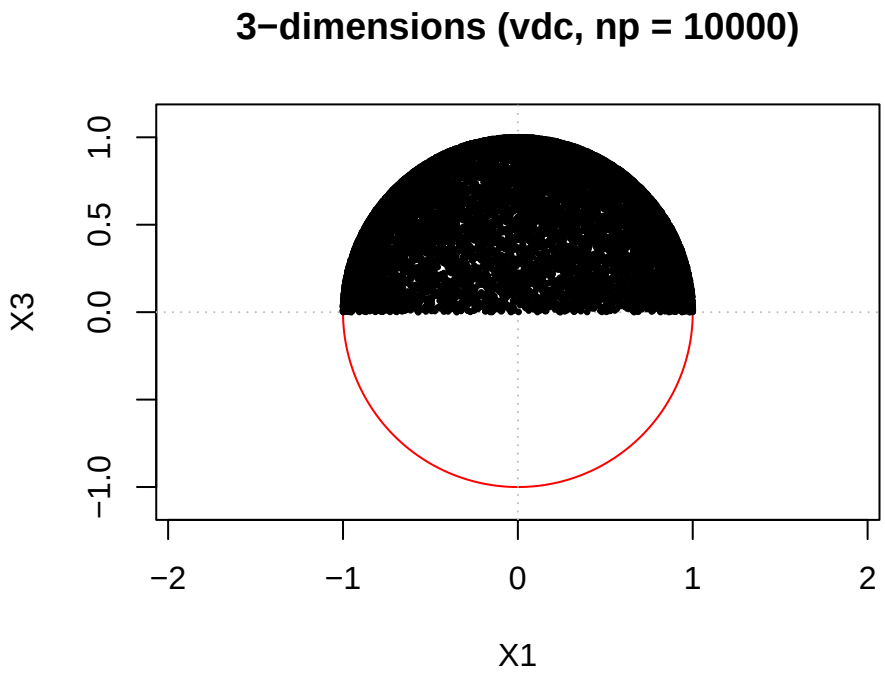

3−dimensions (vdc, np = 100)
X3
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X2

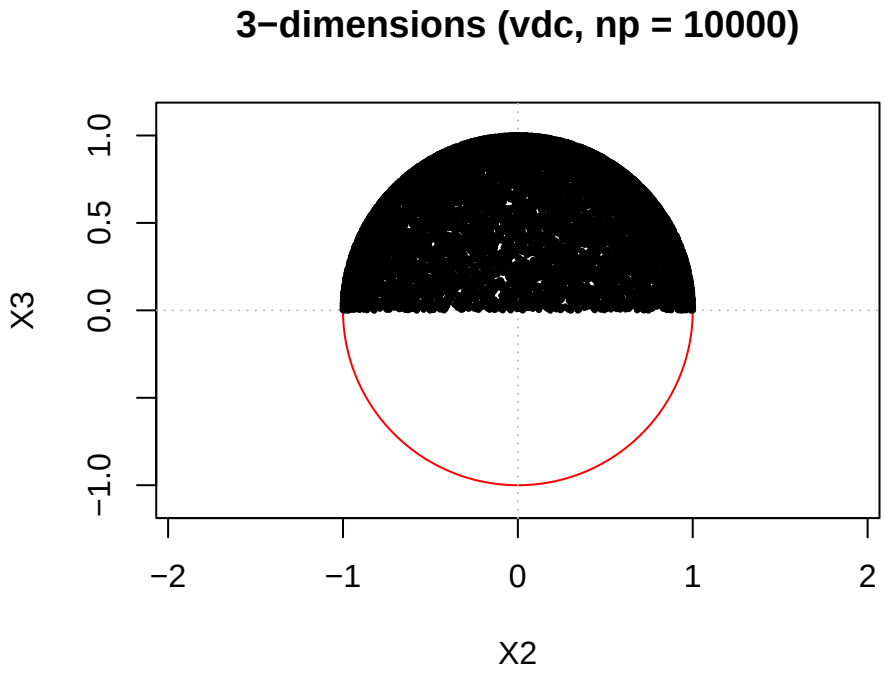

3−dimensions (vdc, np = 1000)
X2
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X1

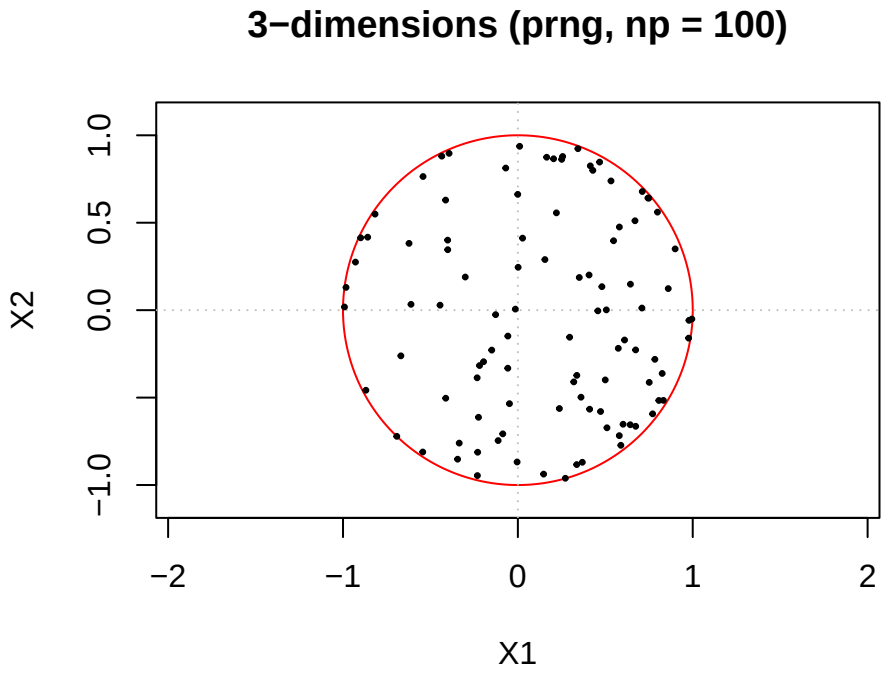

3−dimensions (vdc, np = 1000)
X3
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X1

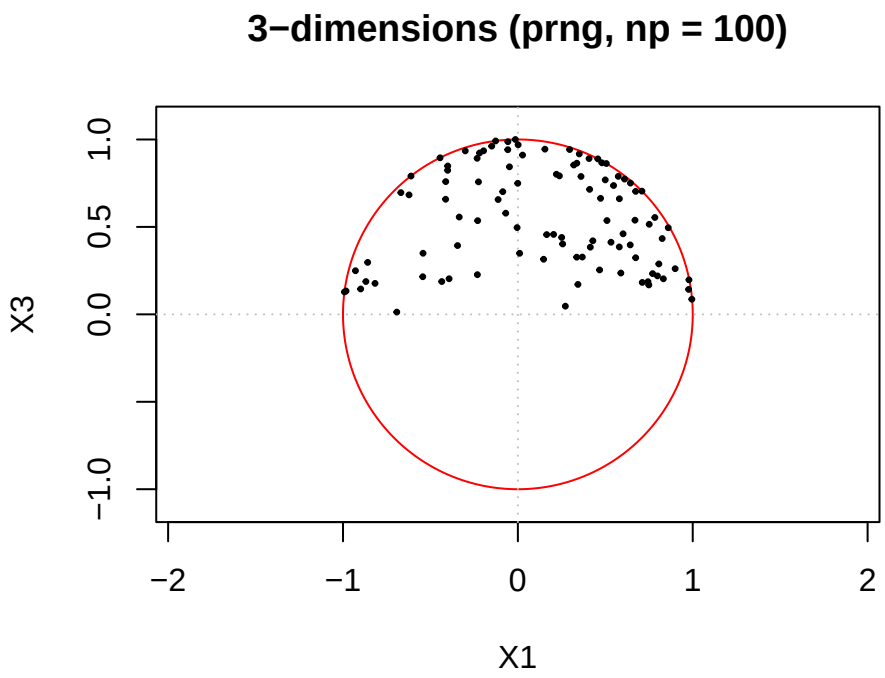

3−dimensions (vdc, np = 1000)
X3
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X2

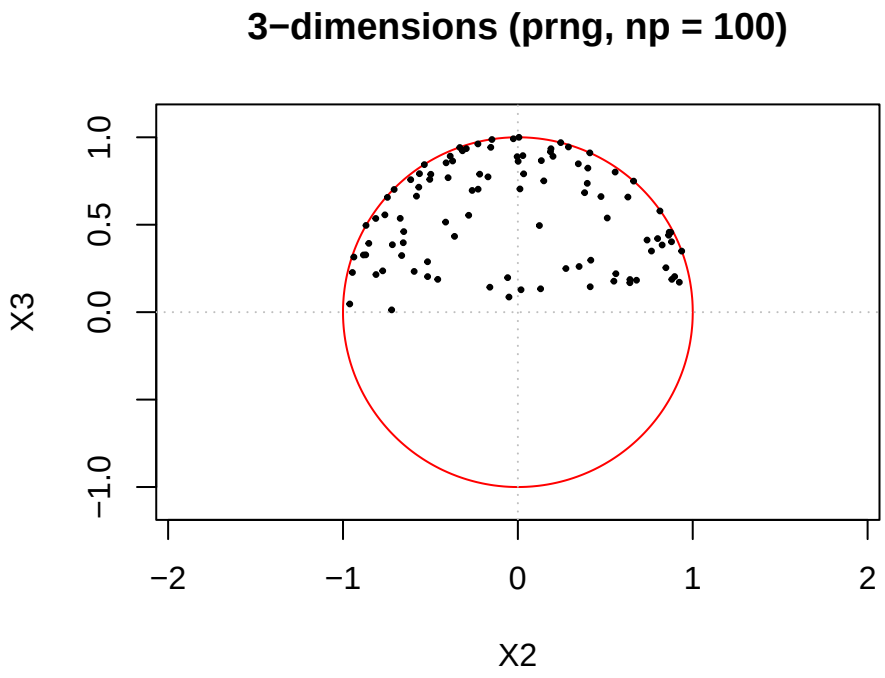

3−dimensions (vdc, np = 10000)
X2
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X1

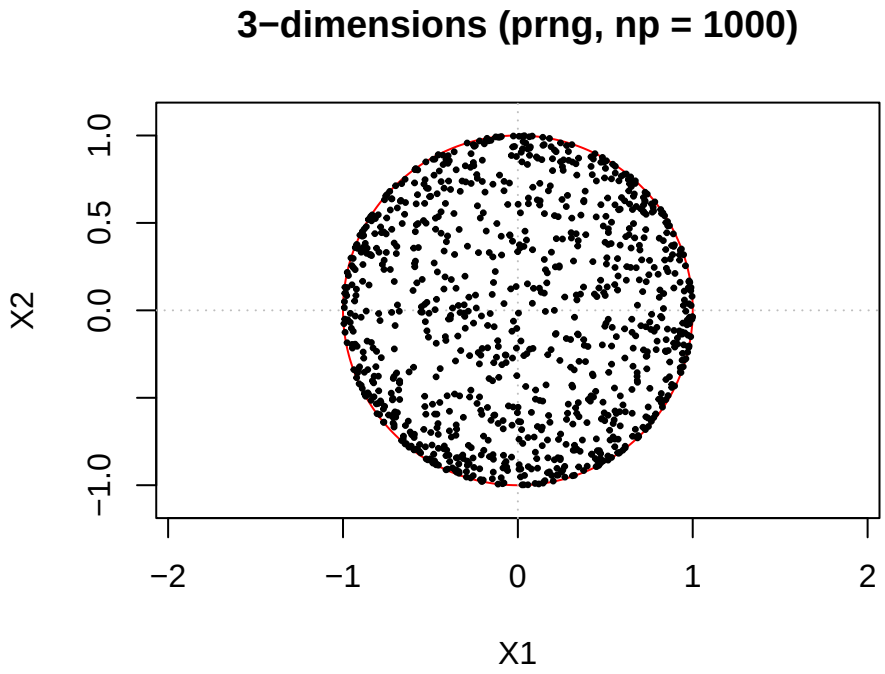

3−dimensions (vdc, np = 10000)
X3
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X1

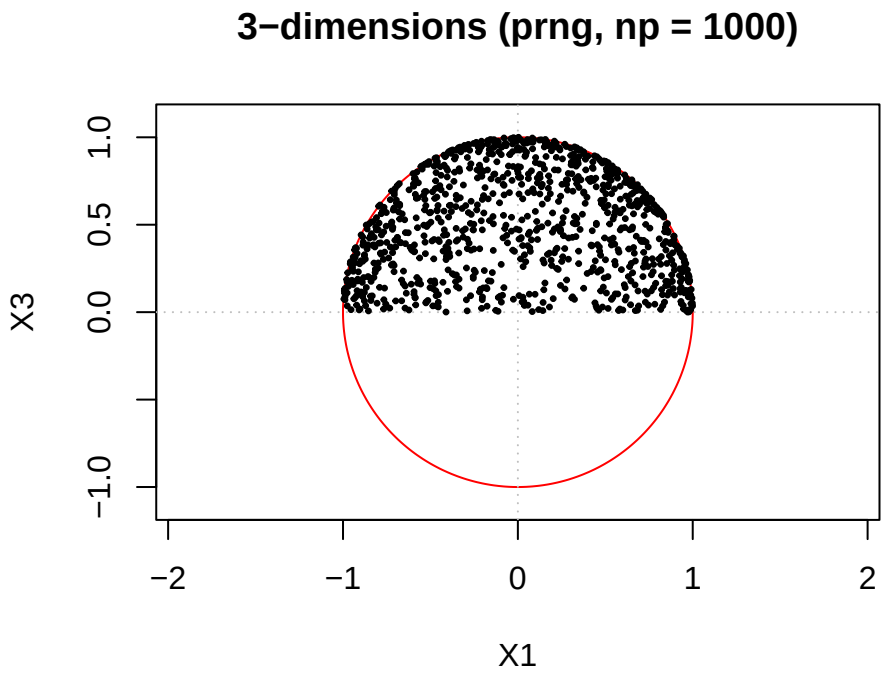

3−dimensions (vdc, np = 10000)
X3
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X2

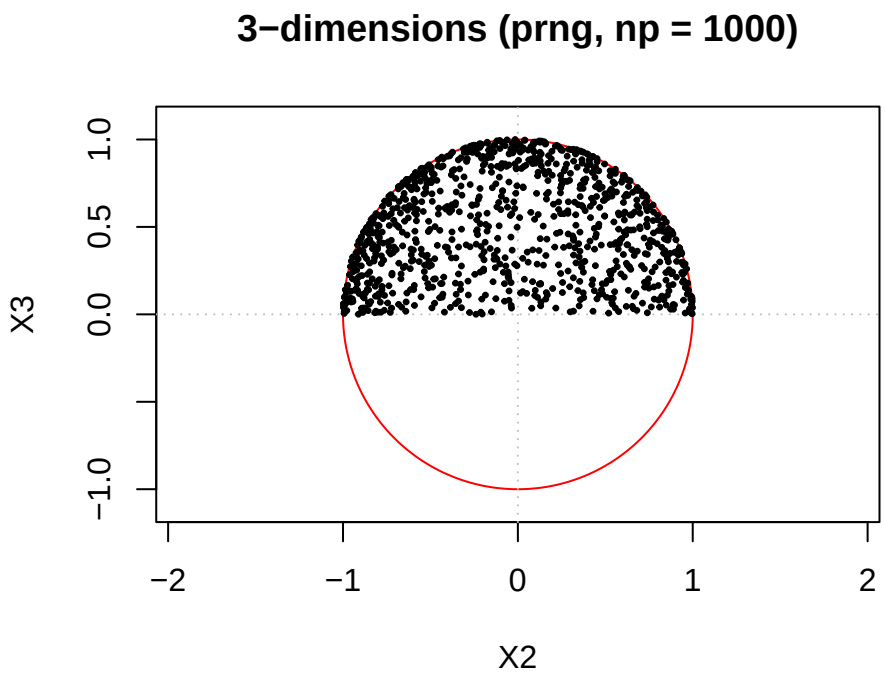

3−dimensions (prng, np = 100)
X2
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X1

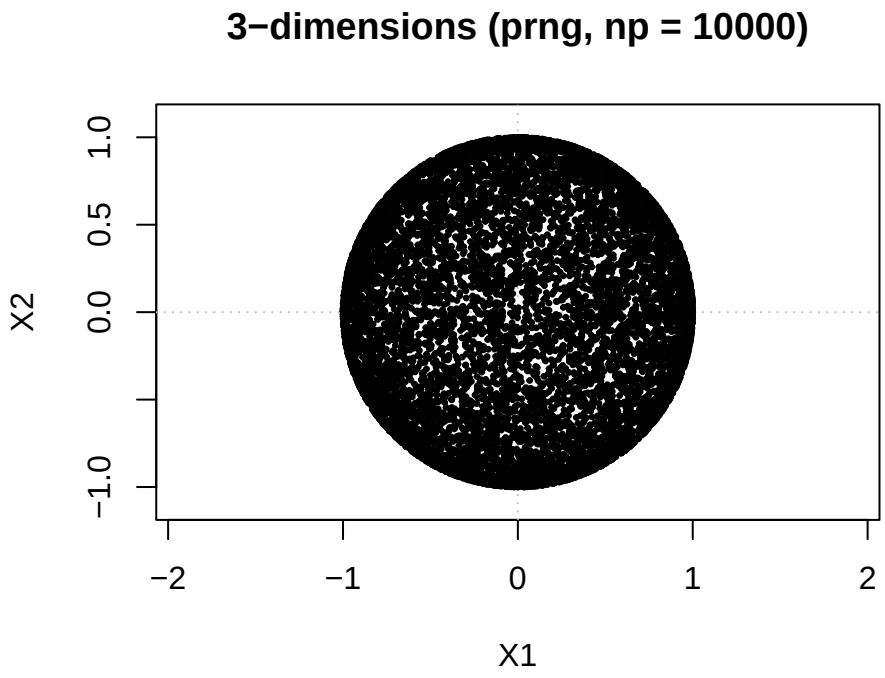

3−dimensions (prng, np = 100)
X3
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X1

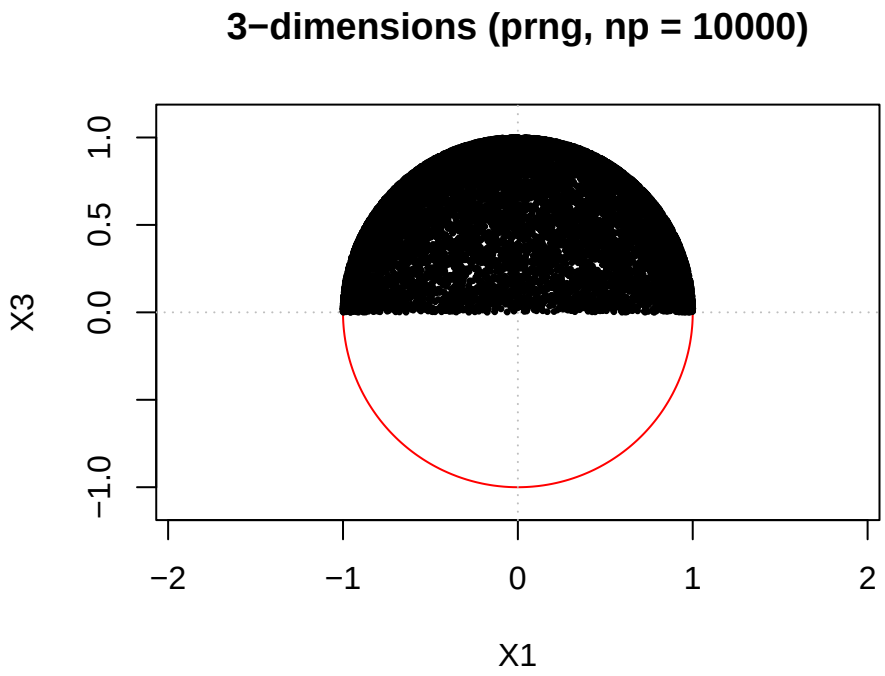

3−dimensions (prng, np = 100)
X3
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X2

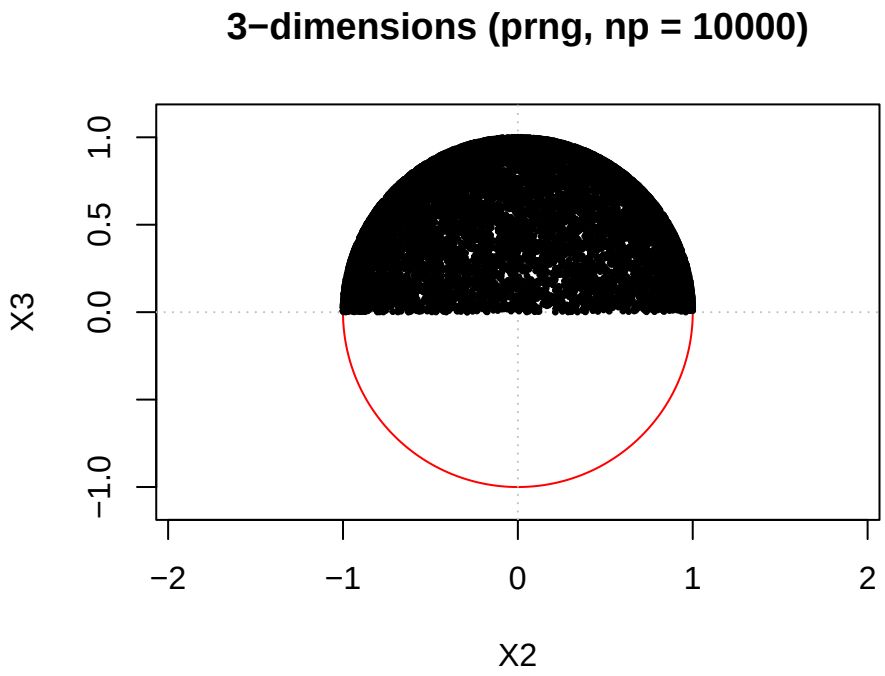

3−dimensions (prng, np = 1000)
X2
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X1


3−dimensions (prng, np = 1000)
X3
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X1

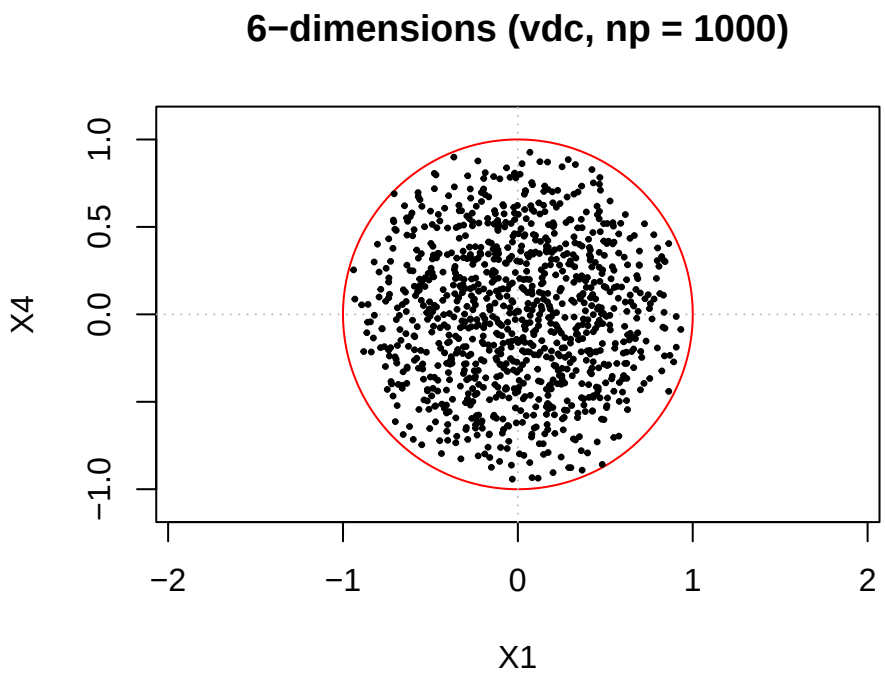

3−dimensions (prng, np = 1000)
X3
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X2


3−dimensions (prng, np = 10000)
X2
1.0
0.5
0.0
−1.0
−2
−1
0
1
2
X1






```
# Directions in R^6
np <- 1000
nd <- 6
meth <- "vdc"
ix <- 1
iy <- 4
plot_dir(np, nd = nd, meth = meth, ix = ix, iy = iy)
```



References

- Devroye, L. (1986) Non-Uniform Random Variate Generation. New York: Springer-Verlag.
- Harman, R., & Lacko, V. (2010). On decompositional algorithms for uniform sampling from n-spheres and n-balls. Journal of Multivariate Analysis, 101(10), 2297-2304.
- Muller, M. E. (1959). A note on a method for generating points uniformly on n-dimensional spheres. Communications of the ACM, 2(4), 19-20.
- Sibuya, M. (1962). A method for generating uniformly distributed points onN-dimensional spheres. Annals of the Institute of Statistical Mathematics, 14(1), 81-85.