Laboratory 1

Basics of generating random variables

Piotr Ginalski

Firstly, let's generate uniform random variables.

```
n <- 12

set.seed(10)
x <- runif(n)
x
```

```
##  [1] 0.50747820 0.30676851 0.42690767 0.69310208 0.08513597 0.22543662
##  [7] 0.27453052 0.27230507 0.61582931 0.42967153 0.65165567 0.56773775
```

We can easily approximate normal distribution by CLT.

```
normal_variable <- (sum(x) - n/2) / (sqrt(n/12))
normal_variable
```

```
## [1] -0.9434411
```

And we have a function which returns realization of normal distribution.

```
normal <- function(){
  n <- 12
  x <- runif(n)
  normal_variable <- (sum(x) - n/2) / (sqrt(n/12))
  return(normal_variable)
}
normal()
```

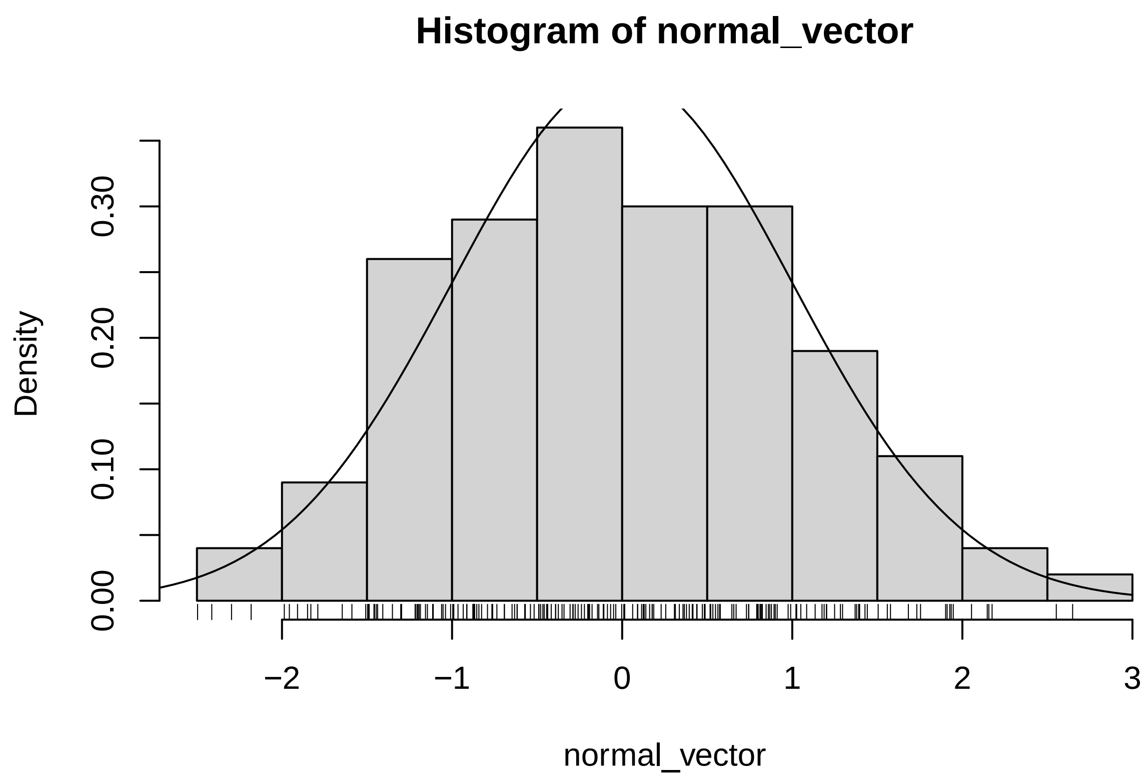
```
## [1] -0.3419482
```

Generating m realizations.

```
m <- 200
normal_vector <- vector()
for (i in 1:m){
  normal_vector <- c(normal_vector, normal())
}
```

We plot the result.

```
{
hist(normal_vector, prob = TRUE)
rug(normal_vector)
curve(dnorm, from = -3, to = 3, add = TRUE)
}
```

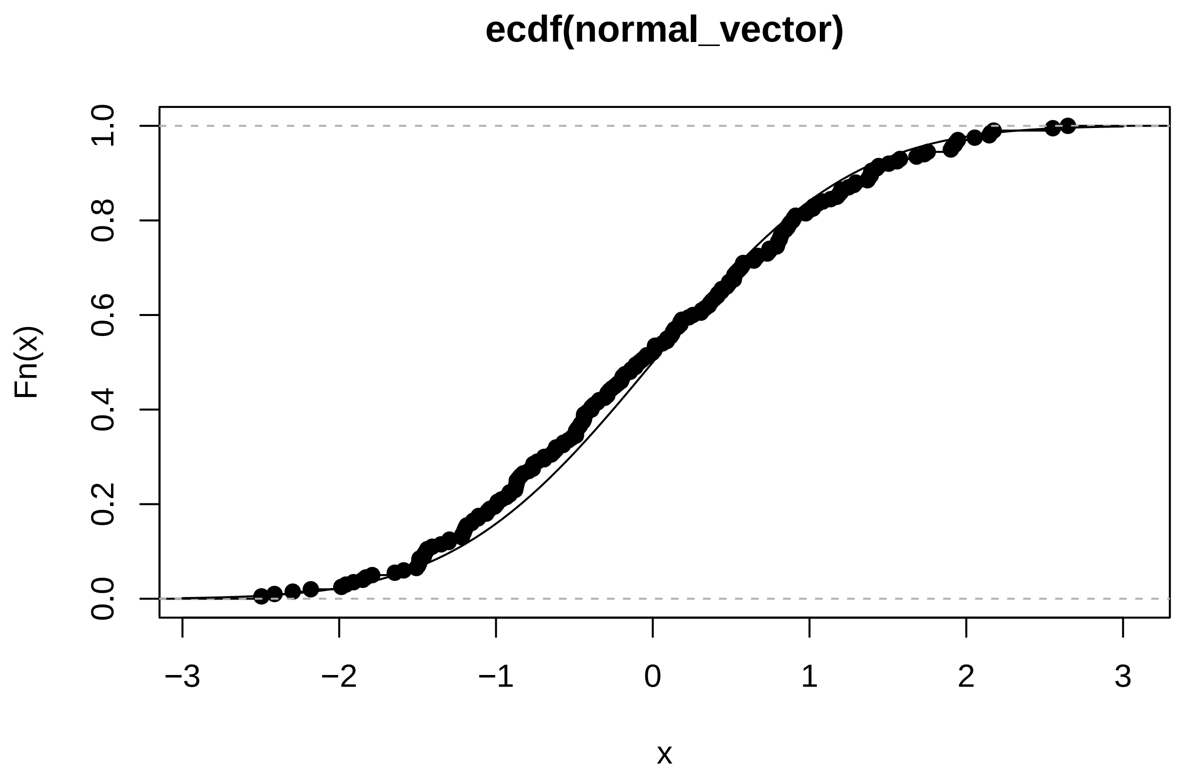


Plotting the empirical cumulative distribution function.

```
ecdf(normal_vector)
```

```
## Empirical CDF
## Call: ecdf(normal_vector)
##  x[1:200] = -2.4961, -2.4128, -2.2965,  ..., 2.5524, 2.6483
```

```
{
plot(ecdf(normal_vector))
curve(pnorm, from = -3, to = 3, add = TRUE)
}
```



```
ks.test(normal_vector, pnorm)
```

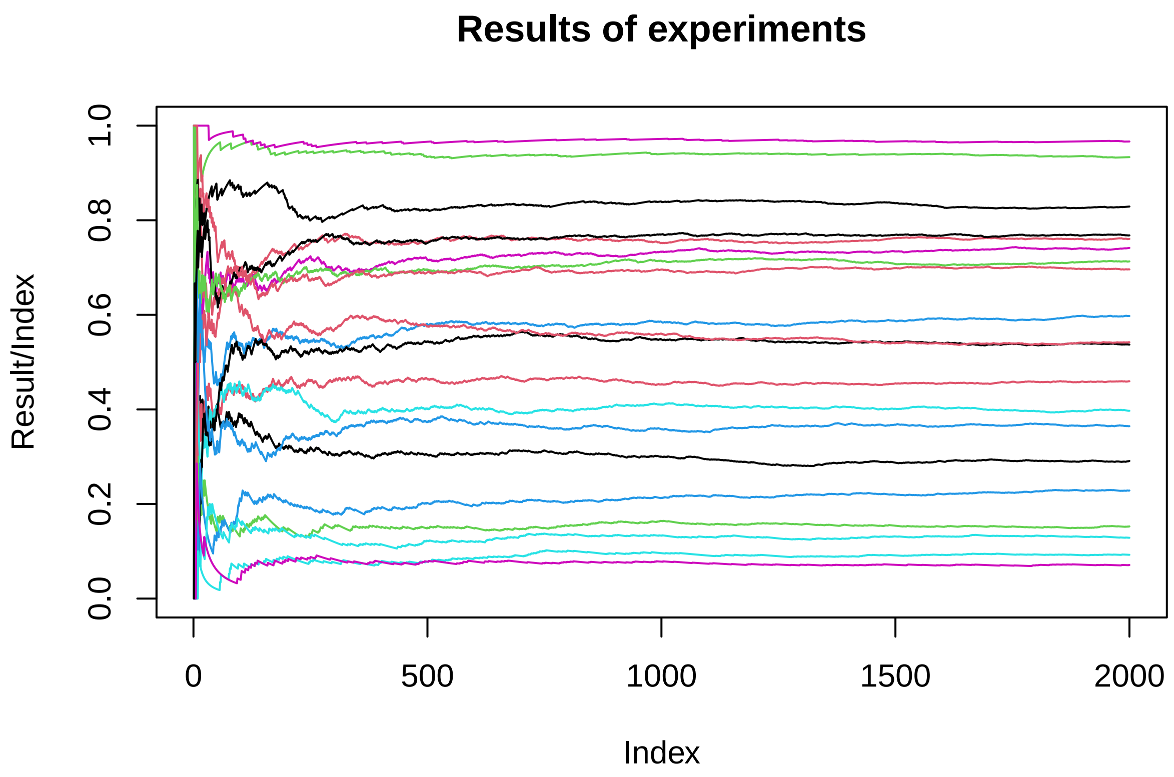
```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  normal_vector
## D = 0.061753, p-value = 0.4306
## alternative hypothesis: two-sided
```

Proceeding to another task and defining the Polya urn scheme

```
bern <- function(alfa, beta){
  x <- runif(1)
  return(as.integer(x < alfa / (alfa + beta)))
}

experiment <- function(alfa, beta, n){
X <- vector()
for (i in 1:n){
  Xn <- bern(alfa + sum(X), beta + length(X) - sum(X))
  X <- c(X, Xn)
}

S <- cumsum(X)
return(S)
}
```

Performing the Polya experiment.

```
alfa <- 1
beta <- 1
n <- 2000
data_for_plot <- data.frame(x = 1:n)

for(i in 1:20){
  results <- experiment(alfa, beta, n)
  name <- paste0("experiment", i)
  data_for_plot[name] <- results / (1:n)
}

# Creating plot
matplot(data_for_plot$x, data_for_plot[, -1], type = "l", lty = 1,
        xlab = "Index", ylab = "Result/Index", main = "Results of experiments")
```



```
generate_betas <- function(alfa, beta, n, m){
 betas <- vector()

  for(i in 1:m){
    results <- experiment(alfa, beta, n)
    betas <- c(betas, results[n]/n)
  }
 return(betas)
```

```r
}

my_betas <- generate_betas(1, 1, 200, 200)
ks.test(my_betas, "pbeta", shape1 = 1, shape2 = 1)
```

```
## Warning in ks.test.default(my_betas, "pbeta", shape1 = 1, shape2 = 1): ties
## should not be present for the one-sample Kolmogorov-Smirnov test

## 
##  Asymptotic one-sample Kolmogorov-Smirnov test
## 
## data:  my_betas
## D = 0.06, p-value = 0.4676
## alternative hypothesis: two-sided
```

```r
my_betas <- generate_betas(3, 2, 200, 200)
ks.test(my_betas, "pbeta", shape1 = 3, shape2 = 2)
```

```
## Warning in ks.test.default(my_betas, "pbeta", shape1 = 3, shape2 = 2): ties
## should not be present for the one-sample Kolmogorov-Smirnov test

## 
##  Asymptotic one-sample Kolmogorov-Smirnov test
## 
## data:  my_betas
## D = 0.050814, p-value = 0.6801
## alternative hypothesis: two-sided
```

```r
my_betas <- generate_betas(11, 1, 200, 200)
ks.test(my_betas, "pbeta", shape1 = 11, shape2 = 1)
```

```
## Warning in ks.test.default(my_betas, "pbeta", shape1 = 11, shape2 = 1): ties
## should not be present for the one-sample Kolmogorov-Smirnov test

## 
##  Asymptotic one-sample Kolmogorov-Smirnov test
## 
## data:  my_betas
## D = 0.041921, p-value = 0.8736
## alternative hypothesis: two-sided
```

```r
my_betas <- generate_betas(0.5, 0.5, 200, 200)
ks.test(my_betas, "pbeta", shape1 = 0.5, shape2 = 0.5)
```

```
## Warning in ks.test.default(my_betas, "pbeta", shape1 = 0.5, shape2 = 0.5): ties
## should not be present for the one-sample Kolmogorov-Smirnov test

## 
##  Asymptotic one-sample Kolmogorov-Smirnov test
## 
## data:  my_betas
## D = 0.049444, p-value = 0.7125
## alternative hypothesis: two-sided
```

```r
my_betas <- generate_betas(5, 31, 200, 200)
ks.test(my_betas, "pbeta", shape1 = 5, shape2 = 31)
```

```
## Warning in ks.test.default(my_betas, "pbeta", shape1 = 5, shape2 = 31): ties
## should not be present for the one-sample Kolmogorov-Smirnov test
```

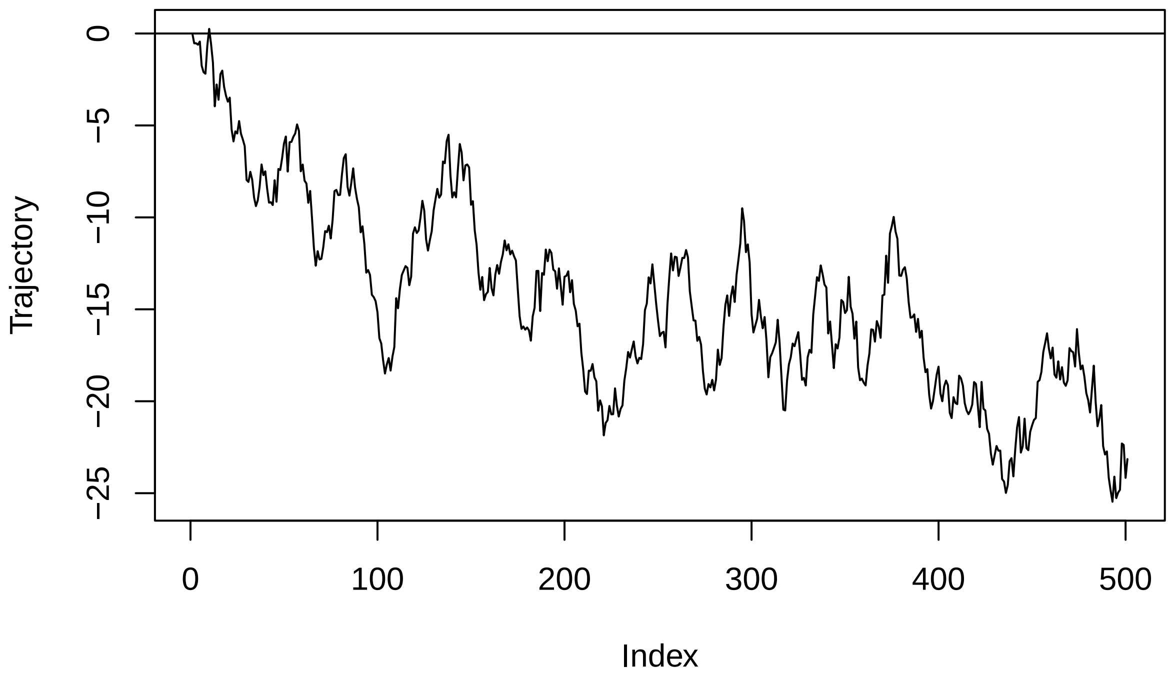
```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  my_betas
## D = 0.071277, p-value = 0.2615
## alternative hypothesis: two-sided
```

Proceeding to the next task. Clearing the environment.

```r
rm(list = ls())
```

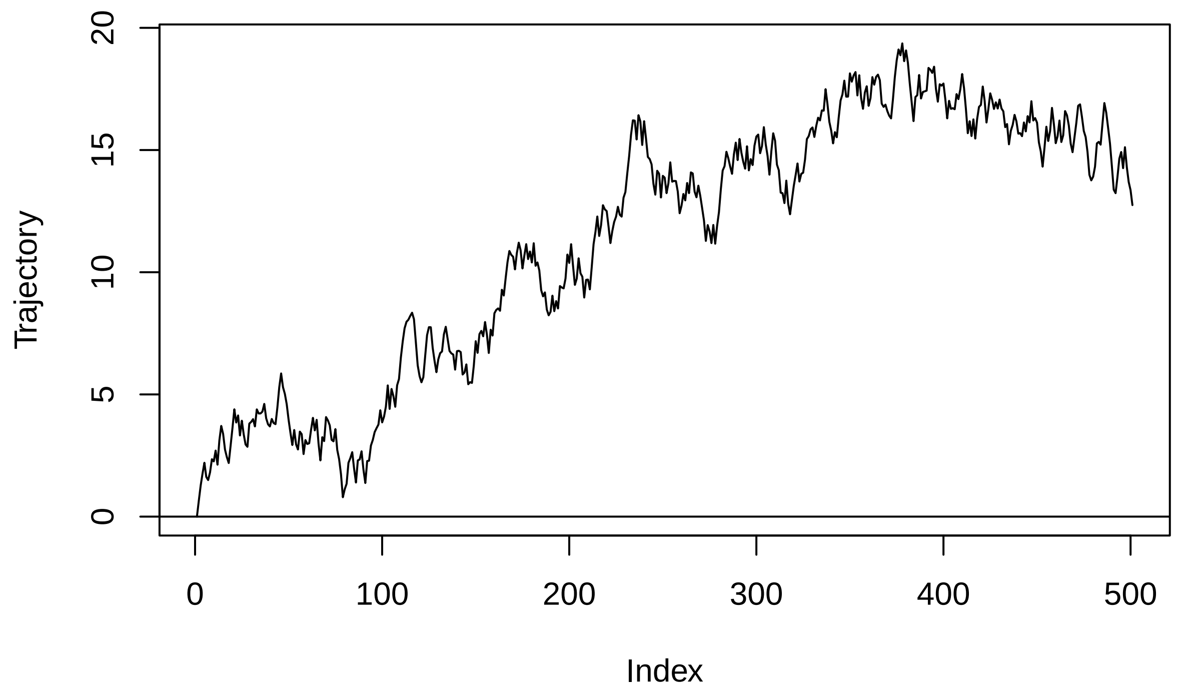
Performing exercise 1.8 from the script. We consider the arcsine law and define the random variable T_n – the proportion of time we were positive during the games. The function random_walk takes an argument 'distribution' which specifies the distribution to sample X_i from.

```r
random_walk <- function(distribution, n, ...){
Sn <- cumsum(distribution(n, ...))
Sn <- c(0, Sn)
Tn <- cumsum(as.integer(Sn > 0)) / (1:(n+1))

return(list(Sn = Sn, Tn = Tn))
}


two_pointed <- function(n, alfa, beta){
  x <- runif(n)
  return(as.integer(x < alfa / (alfa + beta))*2 - 1)
}

normal_walk <- random_walk(rnorm, 500, mean = 0, sd = 1)
unif_walk <- random_walk(runif, 500, min = -1, max = 1)
bern_walk <- random_walk(two_pointed, 500, alfa = 1, beta = 1)

plot(normal_walk$Sn, type = "l", ylab = "Trajectory") + abline(h = 0)
```



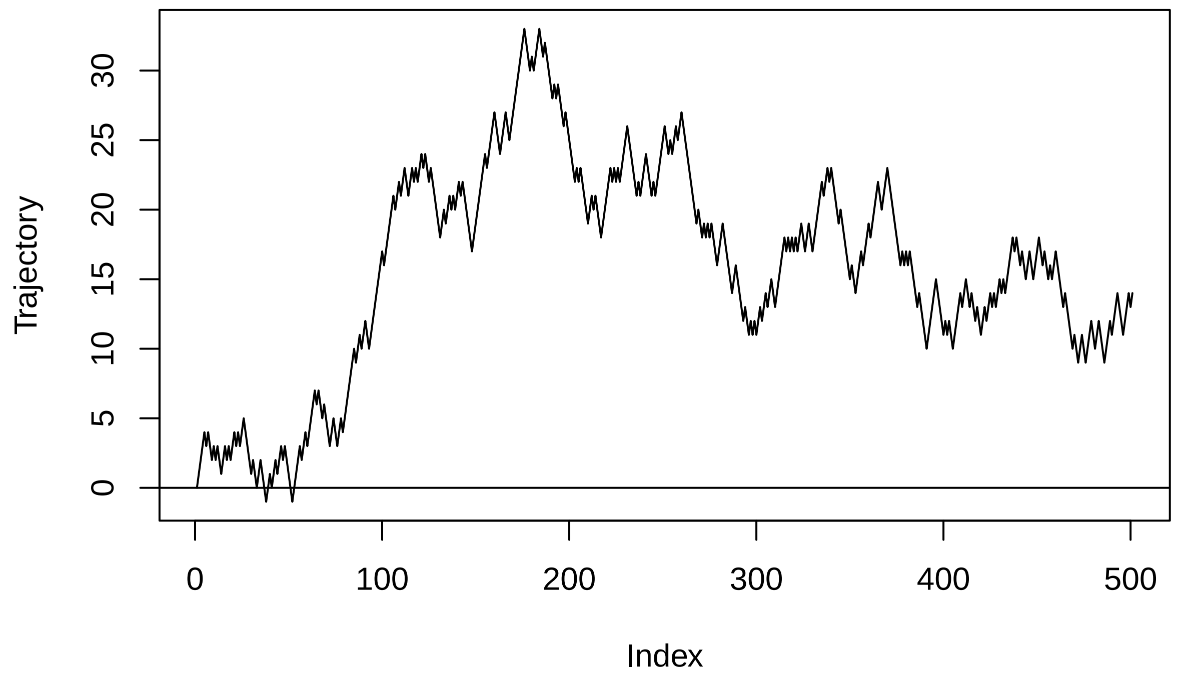
```
## integer(0)
```

```
plot(unif_walk$Sn, type = "l", ylab = "Trajectory") + abline(h = 0)
```



```
## integer(0)
```

```
plot(bern_walk$Sn, type = "l", ylab = "Trajectory") + abline(h = 0)
```



```
## integer(0)
```

```
generate_Tn <- function(distribution, n, m, ...){
result <- vector()
for (i in (1:m)){
  walk <- random_walk(distribution, n, ...)
  result <- c(result, walk$Tn[length(walk$Tn)])
}
return(result)
}

realization_Tn <- generate_Tn(rnorm, 500, 50, mean = 0, sd = 1)
ks.test(realization_Tn, "pbeta", shape1 = 1/2, shape2 = 1/2)
```

```
## Warning in ks.test.default(realization_Tn, "pbeta", shape1 = 1/2, shape2 =
## 1/2): ties should not be present for the one-sample Kolmogorov-Smirnov test
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  realization_Tn
## D = 0.078596, p-value = 0.9169
## alternative hypothesis: two-sided
```

Proceeding to the next task. Clearing the environment.

```r
rm(list = ls())
```

Creating discrete variables with fixed probabilities.

```r
argmin <- vector()

for (i in 1:10000){
W = rexp(3, rate = c(1, 2, 5))
argmin <- c(argmin, which.min(W))
}

table(argmin) / length(argmin)
```

```
## argmin
##      1      2      3
## 0.1280 0.2412 0.6308
```

```r
# Function for generating discrete variables
generate_discretes <- function(n, freq){
  argmin <- vector()
  for (i in 1:n){
    W = rexp(length(freq), rate = freq)
    argmin <- c(argmin, which.min(W))
  }
  return(argmin)
}

n <- 1000
vector1 <- c(1,2,3)
vector2 <- c(1, 1, 10, 2, 3)
vector3 <- c(8, 12)

discrete_realization <- generate_discretes(n, vector1)
table(discrete_realization) / n
```

```
## discrete_realization
##     1     2     3
## 0.183 0.343 0.474
```

```r
discrete_realization <- generate_discretes(n, vector2)
table(discrete_realization) / n
```

```
## discrete_realization
##     1     2     3     4     5
## 0.062 0.051 0.602 0.124 0.161
```

```r
discrete_realization <- generate_discretes(n, vector3)
table(discrete_realization) / n
```

```
## discrete_realization
##     1     2
## 0.394 0.606
```

And finally, let's clear environment.

```r
rm(list = ls())
```