Exercise 1

Write a class `HaltonSequencePiFromHypersphere`, providing an approximation of the value of π via the approximation of the integral

$$V_d = \int_{-1}^{1} \cdots \int_{-1}^{1} \mathbf{1}_{\{x_1^2+\cdots+x_d^2 \leq 1\}} dx_1 \ldots dx_d = 2^d \int_{0}^{1} \cdots \int_{0}^{1} \mathbf{1}_{\{(2(x_1-0.5))^2+\cdots+(2(x_d-0.5))^2 \leq 1\}} dx_1 \ldots dx_d$$

and the equations

$$V_{2k} = \frac{\pi^k}{k!},$$
$$V_{2k+1} = \frac{2(4\pi)^k k!}{(2k+1)!}.$$

for $k \geq 1$ natural number, where the evaluations points $(x_1^i, \ldots, x_d^i)$, $i = 1, \ldots, n$, with n number of sample points, are now provided by an Halton sequence with a given d-dimensional base.

You can write a class `HaltonSequence`, with a method providing the sample points, or directly use the one in

`info.quantlab.numericalmethods.lecture.randomnumbers`

in the `numerical-methods-lecture` project.

The class `HaltonSequencePiFromHypersphere` must also provide a `public` method which returns the error in the approximation (note that here, for a given base, only one value of the approximation is produced, so it does not make sense to consider a vector of approximations as for the Monte-Carlo method).

Experiment on the quality of the approximation of the two methods by printing the average error produced by `MonteCarloPiFromHypersphere` of Exercise 2 of Handout 4 for 100 computations and the error given by `HaltonSequencePiFromHypersphere`, using in both cases 100000 sample points, for different dimensions.

Regarding the choice of the base of `HaltonSequencePiFromHypersphere`, consider the following cases:

- all the elements of the base are equal to each other (for example, `base =` $\{$`2,2,2,2`$\}$ for dimension 4);
- the elements of the base are different to each other, but share common divisors (for example, `base =` $\{$`2,4,6,8`$\}$ for dimension 4);
- the elements of the base are different to each other, and do not share common divisors (for example, `base =` $\{$`2,3,5,7`$\}$ for dimension 4).

What do you observe regarding the approximation error? How can you explain this behaviour?

Exercise 2

Write a class with two `public` methods, returning the discrepancy and the star discrepancy, respectively, of a set $\{x_1, \ldots, x_n\}$ of one-dimensional points. These points are not supposed to be sorted when given. Test your implementation computing the discrepancy and the star discrepancy of the sets

$$A_1 = \{1/8, 1/4, 1/2, 3/4\}$$

and

$$A_2 = \{1/4, 1/2, 5/8, 3/4\}.$$

You have to get

$$D(A_1) = \frac{3}{8}, \quad D^*(A_1) = \frac{1}{4}, \quad D(A_2) = \frac{1}{2}, \quad D^*(A_2) = \frac{1}{4}.$$

Hint:

The discrepancy may be computed as

$$D(\{x_1,\dots,x_n\}) = \max_{a\in\{0,x_1,\dots,x_n\}} \max_{b\in\{x_1,\dots,x_n,1\},b>a} \max\left(b-a-\frac{|x_i\in(a,b)|}{n}, \frac{|x_i\in[a,b]|}{n}-(b-a)\right). \quad (1)$$

One can then use representation (1) by first computing

$$\max_{b\in\{x_1,\dots,x_n,1\},b>a} \max\left(b-a-\frac{|x_i\in(a,b)|}{n}, \frac{|x_i\in[a,b]|}{n}-(b-a)\right) \quad (2)$$

for $a \in \{x_1,\dots,x_n\}$ fixed, and then computing the discrepancy as the maximum between the star discrepancy, which is

$$\max_{b\in\{x_1,\dots,x_n,1\},b>a} \max\left(b-\frac{|x_i\in(0,b)|}{n}, \frac{|x_i\in[0,b]|}{n}-b\right),$$

and the maximum of the values of (2).

Exercise 3

Consider an interface `UniformRandomNumberSequence` taking care of the generation of a sequence of (pseudo) random numbers uniformly distributed in the interval $(0,1)$. Suppose that this interface has a method

```
double[] getSequenceOfRandomNumbers(),
```

which returns a one-dimensional array of random numbers uniformly distributed in $(0,1)$.

Imagine now you write a class `TwoDimensionalFunctionIntegration` whose goal is to compute the Monte-Carlo approximation of an integral

$$\int_0^1 \int_0^1 f(x,y)dxdy, \quad (3)$$

where $f : (0,1)\times(0,1) \to \mathbb{R}$. Assume this class has a constructor

```
TwoDimensionalFunctionIntegration(UniformRandomNumberSequence sequenceGenerator,
                    DoubleUnaryOperator integrand).
```

(a) Consider a class implementing the interface `UniformRandomNumberSequence` by a Van-der-Corput sequence with a given base. Can you pass an object of such a class to the constructor above, together with a given `DoubleUnaryOperator`, to achieve a good approximation of the integral in (3)? Give a short explanation of your answer.

(b) Is there any method you would add to the interface `UniformRandomNumberSequence` in order to get a better approximation of the integral in (3) when you pass an object of a class implementing such an interface to the constructor of `TwoDimensionalFunctionIntegration`?

Note: this is *theoretical* exercise, not a coding one. Of course if you like you can write the code in order to have a look at what happens, but that would not be the solution.