Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Лабораторна робота № 3**

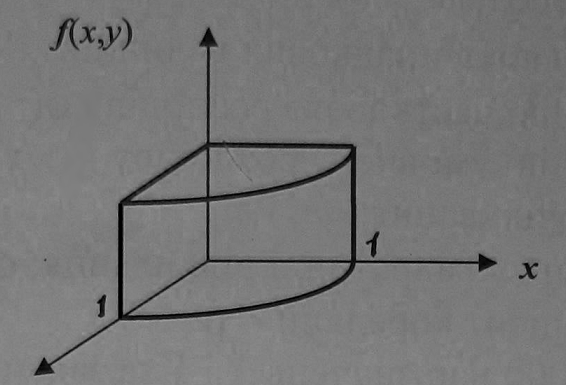
*Системи випадкових величин*

Виконав студент групи ІО-01

Редько Олександр

# Завдання

Задана функція розподілу двох випадкових величин.



Легко визначити, що .

# Аналітичні розрахунки

## Функція часткового розподілу

## Функція умовного розподілу

## Математичне очікування

## Середньоквадратичне відхилення

Тоді,

Аналогічно визначається :

Отже,

## Коефіцієнт кореляції

# Програмна генерація вибірки 1000 пар значень *X* і *Y* за заданою функцією

## Експериментальне визначення мат. очікування, середньоквадратичного відхилення, коефіцієнта кореляції

mean X = 0.4212182111179136

mean Y = 0.42249320873915647

standart deviation X = 0.26581552590063506

standart deviation Y = 0.2614796868170328

correlation = -0.3201958265604174

## Лістинг коду

**import** java.util.Random;

**public** **class** SystemsOfRandomNumbers {

/\*\*

\* величина вибірки

\*/

**private** **static** **final** **int** *NUMBERS* = 1000;

**public** **static** **void** main(String[] args) {

**double**[] arrayX = **new** **double**[*NUMBERS*];

**double**[] arrayY = **new** **double**[*NUMBERS*];

*generateArrayX*(arrayX);

*generateArrayY*(arrayY, arrayX);

System.*out*.println("\n mean X = " + *mean*(arrayX));

System.*out*.println("\n mean Y = " + *mean*(arrayY));

System.*out*.println("\n standart deviation X = " + *standartDeviation*(arrayX));

System.*out*.println("\n standart deviation Y = " + *standartDeviation*(arrayY));

System.*out*.println("\n correlation = " + *correlation*(arrayX, arrayY));

}

/\*\*

\* Генерація числа x.

\* Реалізується як знаходження кореня рівняння методом половинного ділення

\* **@param** r число від 0 до 1

\* **@return** згенерований x від 0 до 1

\*/

**public** **static** **double** solveEquation (**double** r){

**double** a = 0, b = 1, x = a, xt, eps = 0.00000000001;

**do**{

xt = x;

x = (b + a)/2;

**if**(*FuncX*(x, r) \* *FuncX*(a, r) > 0)

a = x;

**else**

b = x;

}**while** (Math.*abs*(x - xt) >= eps);

**return** x;

}

/\*\*

\* Генерація масиву з x, розподілених за заданим функцією щільності

\* вірогідності (четверта частина циліндра)

\* **@param** array

\*/

**public** **static** **void** generateArrayX(**double** [] arrayX){

**double**[] arrayRandoms = **new** **double**[*NUMBERS*];

*generateRandom*(arrayRandoms);

**for**(**int** i = 0; i < arrayX.length; i++){

arrayX[i] = *solveEquation*(arrayRandoms[i]);

}

}

**public** **static** **void** generateArrayY(**double**[] arrayY, **double**[] arrayX){

**double**[] arrayRandoms = **new** **double**[*NUMBERS*];

*generateRandom*(arrayRandoms);

**for**(**int** i = 0; i < arrayX.length; i++){

arrayY[i] = arrayRandoms[i] \* Math.*sqrt*(1 - arrayX[i] \* arrayX[i]);

}

}

/\*\*

\* Інтеграл від функції часткового розподілу

\* **@param** x x

\* **@param** r випадкове число від 0 до 1

\* **@return**

\*/

**public** **static** **double** FuncX(**double** x, **double** r){

**return** 4 \* (Math.*asin*(x)/2 + x \* Math.*sqrt*(1 - x \* x)/2)/Math.*PI* - r;

}

/\*\*

\* Математичне очікування.

\* Реалізується як середнє значення вибірки

\* **@param** array масив вибірки

\* **@return** aver математичне очікування m

\*/

**public** **static** **double** mean(**double**[] array){

**double** Sum = 0.0;

**double** aver = 0.0;

**for** (**int** i = 0; i < array.length; i++){

Sum += array[i];

}

aver = Sum/array.length;

**return** aver;

}

/\*\*

\* Середньоквадратичне відхилення випадкової величини

\* **@param** array вибірка

\* **@return** значення сигма

\*/

**public** **static** **double** standartDeviation(**double**[] array){

**double** disp = 0;

**for**(**int** i = 0; i < array.length; i++) {

disp += Math.*pow*((array[i] - *mean*(array)), 2);

}

disp = disp/array.length;

**return** Math.*sqrt*(disp); //сигма - корінь із дисперсії

}

/\*\*

\* Коваріація систем двох величин

\* **@param** arrayX вибірка X

\* **@param** arrayY вибірка Y

\* **@return** cov

\*/

**public** **static** **double** covariance(**double**[] arrayX, **double**[] arrayY){

**double** cov = 0;

**for**(**int** i = 0; i < arrayX.length; i++) {

cov += (arrayX[i] - *mean*(arrayX)) \* (arrayY[i] - *mean*(arrayY));

}

cov = cov/arrayX.length;

**return** cov;

}

/\*\*

\* Коефіцієнт кореляції

\* **@param** arrayX вибірка X

\* **@param** arrayY вибірка Y

\* **@return** cor = cov/(sigmaX, sigmaY)

\*/

**public** **static** **double** correlation(**double**[] arrayX, **double**[] arrayY){

**return** *covariance*(arrayX, arrayY)/(*standartDeviation*(arrayX) \*

*standartDeviation*(arrayY));

}

/\*\*

\* Генерує масив рандомних чисел із інтервалу [0, 1]

\* **@param** array

\*/

**public** **static** **void** generateRandom(**double** [] array){

Random R = **new** Random();

**for** (**int** i = 0; i < array.length; i++){

array[i] = R.nextDouble();

}

}

/\*\*

\* Метод для друку на екран масиву

\* **@param** array

\*/

**public** **static** **void** showArray(**double**[] array){

**for**(**int** i = 0; i < array.length; i++){

System.*out*.printf(" " + array[i]);

}

}

}