Національний технічний університет України  
«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота № 5

З дисципліни «Теорія ймовірностей та математична статистика»

На тему «Моделювання випадкових процесів»

Виконав:

студент ІІ курсу ФІОТ

Групи ІО-41

Логвинчук А. І.

Перевірив:

Доц. Марковський О. П.

Київ - 2015

Варінат: №13 Марківський 3 стани Неперервний

ПРОГРАМНИЙ КОД

**public** **class** Main {

**static** **final** **double**[][] ***intencityMatrix*** = {{0., 0.5, 2},{0., 0., 0.1},{1, 1.5, 0.}};

**static** **double**[][] getCoefMatrix(**double**[][] matrix) {

**int** rank = matrix.length;

**double**[][] coefMatrix = **new** **double**[rank][rank];

**for** (**int** i = 0; i < rank; i++) {

coefMatrix[rank-1][i] = 1.;

}

**for** (**int** i = 0; i < rank-1; i++ ) {

**for** (**int** j = 0; j < rank; j++) {

**if** (i == j) {

**for** (**int** k = 0; k < rank; k++ ){

coefMatrix[i][j] -= matrix[i][k];

}

} **else** {

coefMatrix[i][j] = matrix[j][i];

}

}

}

**return** coefMatrix;

}

**static** **double**[] getTheoreticalValues(**double**[][] matrix) {

**double**[][] coeffs = *getCoefMatrix*(***intencityMatrix***);

**double**[] bVector = **new** **double**[coeffs.length];

Arrays.*fill*(bVector, 0.);

bVector[bVector.length-1] = 1.;

RealVector constants = **new** ArrayRealVector(bVector, **false**);

RealMatrix coefficients = **new** Array2DRowRealMatrix(coeffs, **false**);

DecompositionSolver solver = **new** LUDecomposition(coefficients).getSolver();

RealVector result = solver.solve(constants);

**return** result.toArray();

}

**static** **double** [] getExperimentalValues(**double**[][] matrix) {

**int** rank = matrix.length;

**double**[] result = **new** **double**[rank];

**double** modelTime = 0.;

Random rnd = **new** Random();

**int** currentState = 0;

**for** (**int** count = 0; count < 1000; count++){

**int** nextState = 0;

**boolean** firstEntry = **true**;

**double** t = 0.;

**for** (**int** i = 0; i < rank; i++) {

**if** (matrix[currentState][i] != 0.) {

**if** (firstEntry) {

t = (-1. / matrix[currentState][i])\*Math.*log*(rnd.nextDouble());

nextState = i;

firstEntry = **false**;

} **else** {

**double** k = (-1. / matrix[currentState][i])\*Math.*log*(rnd.nextDouble());

**if** (k < t) {

t = k;

nextState = i;

}

}

}

}

modelTime += t;

result[currentState] += t;

currentState = nextState;

}

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

result[i] /= modelTime;

}

**return** result;

}

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

**double**[] theoretical = *getTheoreticalValues*(***intencityMatrix***);

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

System.***out***.println("P" + (i+1) + "t = " + theoretical[i]);

}

System.***out***.println();

//HERE GOES THE EMPIRICAL PART

**double**[] experimental = *getExperimentalValues*(***intencityMatrix***);

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

System.***out***.println("P" + (i+1) + "e = " + experimental[i]);

}

}

ПЕРЕВІРКА РЕЗУЛЬТАТІВ

Для матриці:

|  |  |  |
| --- | --- | --- |
| Характеристика | Теоретичне  значення | Експериментальне значення |
|  | 0.02173913043478261 | 0.021648742843280375 |
|  | 0.9239130434782609 | 0.9251570137605012 |
|  | 0.05434782608695653 | 0.053194243396218414 |