МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА» (БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №5

по дисциплине: «Теория информации»

Выполнил: ст. группы ПВ-211

Чувилко Илья Романович

Проверил:

Твердохлеб В.В.

Белгород 2023 г.

Тема: «Канальные матрицы »

Содержание отчета

1 Для канала связи, описываемого канальной матрицей (1) со стороны источника, построить канальную матрицу со стороны приемника

2 Для канала связи, описываемого канальной матрицей (2) со стороны источника, построить канальную матрицу со стороны приемника. Принять во внимание, что на некоторых позиция величины необходимо восстановить

Ход работы:

Задание 1 Для канала связи, описываемого канальной матрицей (1) со стороны источника, построить канальную матрицу со стороны приемника

$$\begin{bmatrix} 0{,}98 & 0{,}01 & 0{,}01 \\ 0{,}1 & 0{,}75 & 0{,}15 \\ 0{,}2 & 0{,}3 & 0{,}5 \end{bmatrix} \quad (1)$$

Вероятности появления символов источника следующие: $p(a_1)=0{,}2$, $p(a_2)=0{,}1$, $p(a_3)=0{,}7$.

Для решения задачи напишем следующую функцию, переводящую матрицу источника в матрицу приемника

```
def GetDestinationMatrix(Pba:list, Pa:list):
    Pb = [1.0]*len(Pba)
    print("Vector Pb:\n", Pb)
    for i in range(len(Pba)):
        for j in Pba[i]:
            Pb[i] *= Pba[i][j]
    Pab = []
    for j in range(len(Pba[0])):
        for i in range(len(Pba)):
            Pab[i][j] = Pba[i][j] * Pa[i] / Pb[j]
    return Pb, Pab
```

Запустим программу при заданном условии:

```
# Press the green button in the gutter to run the script.
if __name__ == '__main__':
    Pa = [0.2, 0.1, 0.7]
    Pba = [[0.98, 0.01, 0.01], [0.1, 0.75, 0.15], [0.2, 0.3, 0.5]]
    Pb, Pab = GetDestinationMatrix(Pba, Pa)
    print("Vector Pb:\n", Pb)
    print("Vector Pab:\n", Pab)
```

Результат работы программы:

```
Vector Pb:
[0.3460, 0.2870, 0.3670]

Vector Pab:
[[0.5665, 0.070, 0.3670], [0.0289, 0.2613, 0.0409], [0.4046, 0.7317, 0.9537]]
```

Для проверки корректности выполнения программы напишем следующую функцию:

```
def CheckDestinationMatrix(Pab):
    CheckPab = [[]*len(Pab)]
    for j in range(len(Pab)):
        CheckPab[j] = 0
        for i in range(len(Pab[j])):
            CheckPab[j] += Pab[i][j]
    return CheckPab
```

Запустим программу при заданном условии:

```
# Press the green button in the gutter to run the script.
if __name__ == '__main__':
    Pa = [0.2, 0.1, 0.7]
    Pba = [[0.98, 0.01, 0.01], [0.1, 0.75, 0.15], [0.2, 0.3, 0.5]]
    Pb, Pab = GetDestinationMatrix(Pba, Pa)
    print("Vector Pb:\n", Pb)
    print("Vector Pab:\n", Pab)
    print("Vector CheckPab:\n", CheckDestinationMatrix(Pab))
```

Результат работы программы:

```
Vector Pb:
[0.3460, 0.2870, 0.3670]

Vector Pab:
[[0.5665, 0.070, 0.3670], [0.0289, 0.2613, 0.0409], [0.4046, 0.7317, 0.9537]]

Vector Pb:
[1.0000, 1.0000, 1.0000]
```

Сумма всех столбцов равна 1, следовательно, матрица получена верно.

Задание 2 Для канала связи, описываемого канальной матрицей (2) со стороны источника, построить канальную матрицу со стороны приемника. Принять во внимание, что на некоторых позиция величины необходимо восстановить.

$$\begin{bmatrix} 0{,}1 & 0{,}7 & x & 0{,}1 & 0{,}05 \\ 0{,}1 & 0{,}55 & 0{,}05 & 0{,}2 & 0{,}1 \\ 0{,}3 & 0{,}15 & 0{,}1 & x & 0{,}15 \\ 0{,}25 & 0{,}1 & 0{,}2 & 0{,}15 & 0{,}3 \\ 0{,}4 & x & x & 0{,}4 & 0{,}1 \end{bmatrix} \quad (2)$$

Вероятности появления символов источника следующие: $p(a_1)=0{,}1$, $p(a_2)=0{,}2$, $p(a_3)=0{,}4$, $p(a_4)=0{,}18$, $p(a_5)=0{,}12$.

Восстановим матрицу:

$$P_{\frac{b}{a}} = \begin{bmatrix} 0.1 & 0.7 & 0.05 & 0.1 & 0.05 \\ 0.1 & 0.55 & 0.05 & 0.2 & 0.1 \\ 0.3 & 0.15 & 0.1 & 0.3 & 0.15 \\ 0.25 & 0.1 & 0.2 & 0.15 & 0.3 \\ 0.4 & 0.03 & 0.07 & 0.4 & 0.1 \end{bmatrix}$$

Запустим программу для заданной таблицы:

```
if __name__ == '__main__':
    Pa = [0.1, 0.2, 0.4, 0.18, 0.12]
    Pba = [[0.1, 0.7, 0.05, 0.1, 0.05],
           [0.1, 0.55, 0.05, 0.2, 0.1],
           [0.3, 0.15, 0.1, 0.3, 0.15],
           [0.25, 0.1, 0.2, 0.15, 0.3],
           [0.4, 0.03, 0.07, 0.4, 0.1]]
    Pb, Pab = GetDestinationMatrix(Pba, Pa)
    print("Vector Pb:\n", Pb)
    print("Vector Pab:\n", Pab)
    print("Vector CheckPab:\n", CheckDestinationMatrix(Pab))
```

Результат работы программы:

```
Vector Pb:
[0.2430, 0.2616, 0.0994, 0.2450, 0.1510]

Vector Pab:
[[0.0412, 0.2676, 0.0503, 0.0408, 0.0331],
[0.0823, 0.4205, 0.1006, 0.1633, 0.1325],
[0.4938, 0.2294, 0.4024, 0.4898, 0.3974],
[0.1852, 0.0688, 0.3622, 0.1102, 0.3576],
[0.1975, 0.0138, 0.0845, 0.1959, 0.0795]]

Vector Pb:
[1.0000, 1.0000, 1.0000, 1.0000, 1.0000]
```

Сумма всех столбцов равна 1, следовательно, матрица получена верно.

Вывод: в ходе лабораторной работы изучили теоретический материал про канальные матрицы, научились решать задания по заданной теме.