1 Задание

Реализовать алгоритм на языке программирования Python 3 без использования специальных библиотек.

2 Задача 5

Рассмотреть пример решения задачи аппроксимации данных с использованием искусственной нейронной сети на основе радиально-симметричных функций. Дана экспериментальная зависимость в виде набора из 9 пар точек: значений независимой переменной x и соответствующих им значений функции отклика y, представленных в таблице.

Таблица 1 – Экспериментальная выборка данных

№ примера	*x*	*y*
1	–2,0	–0,48
2	–1,5	–0,78
3	–1,0	–0,83
4	–0,5	–0,67
5	0,0	–0,20
6	0,5	0,70
7	1,0	1,48
8	1,5	1,17
9	2,0	0,20

Требуется, используя данную выборку в качестве обучающей, получить аппроксимирующую модель в виде нейронной сети на основе радиально-симметричных функций. Единственный вход данной сети – значение независимой переменной, единственный выход – соответствующее ей значение функции. Структура сети включает скрытых нейронов (радиальных элементов). Требуется указать центры и

радиусы скрытых радиальных элементов. В качестве центров радиальных элементов использовать значения независимой переменной в опытах 1, 3, 5, 7 и 9.

Указания: использовать функцию Гаусса, евклидову норму. Рассчитать веса по формуле: $w = \left(G^T G\right)^{-1} G^T y$.

3 Код программы

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

from math import exp
import numpy as np
from numpy import linalg as LA
import matplotlib.pyplot as plt

class RadialNeuron:
    def __init__(self, x):
        self.c = x

    def count(self, x):
        return exp(-((x - self.c) ** 2) / ((2 * 1.5 ** 2)))

values = [
    [-2.0, - 0.48],
    [-1.5, -0.78],
    [-1.0, -0.83],
    [-0.5, -0.67],
    [0.0, -0.20],
    [0.5, 0.70],
    [1.0, 1.48],
    [1.5, 1.17],
    [2.0, 0.20]
]

neurons = [RadialNeuron(values[2 * i][0]) for i in range(5)]
h = np.array([[neurons[i].count(values[j][0]) for i in range(5)] for j in
range(9)])
y = np.array([[values[j][1]] for j in range(9)])
w = np.dot(LA.inv(np.dot(h.transpose(), h)), np.dot(h.transpose(), y))

x = [value[0] for value in values]
y1 = [value[1] for value in values]
y2 = [sum([neurons[i].count(value[0]) * w[i][0] for i in range(5)]) for value
in values]

n = len(x)
s = 0
for i in range(n):
    s += abs(1 - y1[i] / y2[i])
print('Средняя относительная ошибка аппроксимации:', int(s / n * 100), '%')

plt.scatter(x, y1, c='red', label='Исходные экспериментальные точки')
plt.plot(x, y2, label='Полученная аппроксимирующая зависимость')
plt.legend()
plt.grid(True)
plt.title('Положение исходных экспериментальных точек\nотносительно графика
полученной нейросетевой аппроксимирующей зависимости')
plt.show()
```

4 Результат работы программы

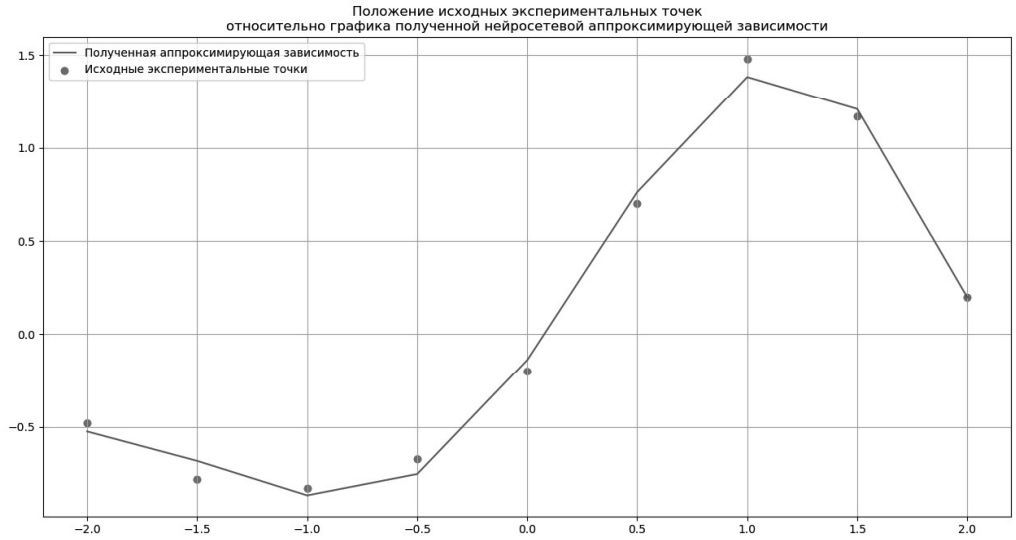

Рисунок 1 – График нейросетевой аппроксимирующей зависимости

На рисунке 1 показано положение исходных экспериментальных точек относительно графика полученной нейросетевой аппроксимирующей зависимости. Как видно, на всем исследуемом диапазоне точки находятся очень близко к линии графика. При использованных настройках нейронной сети средняя относительная ошибка аппроксимации, рассчитанная для экспериментальных точек, составила 11 %.