|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления» (ИУ)

КАФЕДРА «Информационная безопасность» (ИУ8)

Отчёт

по лабораторной работе № 5

по дисциплине «Теория Систем и Системный Анализ»

**Тема: «Двумерный поиск для подбора коэффициентов простейшей нейронной сети на примере решения задачи линейной регрессии экспериментальных данных»**

Вариант 16

Выполнил: Сердюкова М.Ю.,

студентка группы ИУ8-32

Проверил: Коннова Н. С.,

доцент каф. ИУ8

г. Москва,

2020 г.

# 1. Цель работы

Знакомство с простейшей нейронной сетью и реализация алгоритма поиска ее весовых коэффициентов на примере решения задачи регрессии экспериментальных данных.

# 2. Постановка задачи

В зависимости от варианта работы (табл. 4.1) найти линейную регрессию функции y(x) (коэффициенты наиболее подходящей прямой c, d) по набору ее N дискретных значений, заданных равномерно на интервале [a,b] со случайными ошибками e(i) = A\*rnd(-0.5;0.5). Выполнить расчет параметров c, d градиентным методом. Провести двумерный пассивный поиск оптимальных весовых коэффициентов нейронной сети (НС) регрессии.

# 3. Ход работы

1. Рассчитать равномерно распределенные на вашем интервале точки (N дискретных значений, N вводит пользователь программы),

2. Построить график заданной по варианту функций f (x) ;

3. При помощи генератора псевдослучайных чисел с указанными параметрами осуществить зашумление рассчитанных точек, далее использовать их при регрессии;

4. Отобразить эти зашумленные точки на графике;

5. Реализовать программно нейросетевую модель,

6. Используя указанные по варианту методы, осуществить поиск параметров c\*, d\*;

7. Построить график полученной приближенной регрессии;

8. Рассчитайте и выведите получившуюся погрешность приближения, используя рассчитанные при помощи найденных параметров c\*, d \* точки и исходные отсчеты.

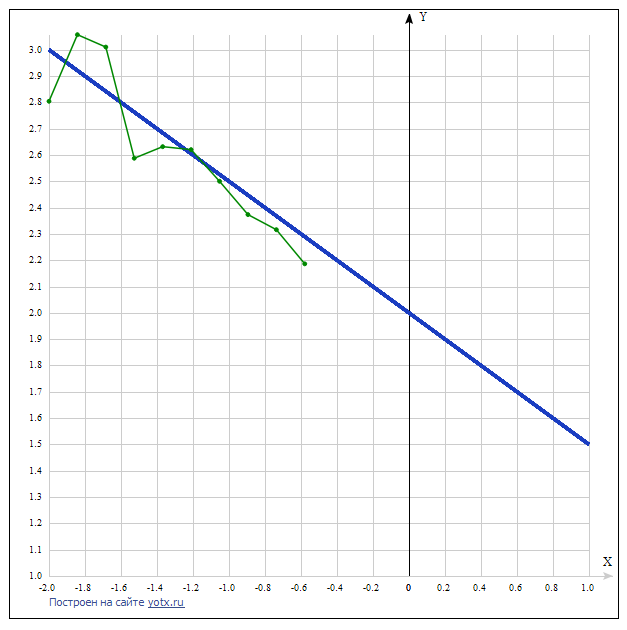


Рисунок 1 – Нейросетевая линейная регрессия экспериментальных данных

Сплошная жирная синяя линия - график функции y = cx+d (y=-0.5x+2);

Сплошная тонкая зелёная линия – зашумленные точки

**4. Выводы**

По результатам лабораторной работы я познакомилась с устройством простейшей нейронной сети и реализовала алгоритм поиска ее коэффициентов.

# Приложение 1. Код программы main.cpp.

#include <iostream>

#include <random>

const int N = 20;

const double fi = 1.618;

double randPoint(double a, double b) {

std::random\_device rd;

std::mt19937 gen(rd());

std::uniform\_real\_distribution<> dist(a, b);

return dist(gen);

}

double f(double x) { return -0.5 \* x + 2; }

static double mnhtn(const std::pair<double, double> &s,

const std::pair<double, double> &f) {

return abs(s.first - f.first) + abs(s.second - s.second);

}

class Function {

std::vector<double> X;

std::vector<double> F;

public:

Function(int a, int b) {

double part = double(b - a) / (N - 1);

for (int i = 0; i < N; i++) {

X.push\_back(a + i \* part);

F.push\_back(f(X[i]) + randPoint(-0.2, 0.2));

std::cout << "(" << X[i] << "," << F[i] << ")" << std::endl;

}

investigation();

}

void withoutNoise() {

for (int i = 0; i < N; i++) {

F.push\_back(f(X[i]));

}

investigation();

}

double minSq(double c, double d) {

double mms = 0;

for (int i = 0; i < N; i++) {

mms += (F[i] - (c \* X[i] + d)) \* (F[i] - (c \* X[i] + d));

}

return mms;

}

double cInvestigation(double d) {

double c11 = -3;

double c22 = 3;

double c1 = c22 - ((c22 - c11) / fi);

double c2 = c11 + ((c22 - c11) / fi);

while ((c22 - c11) / 2 >= 0.1) {

if (minSq(c1, d) > minSq(c2, d)) {

c11 = c1;

c1 = c2;

c2 = c22 - (c1 - c11);

}

if (minSq(c1, d) < minSq(c2, d)) {

c22 = c2;

c2 = c1;

c1 = c11 + (c22 - c2);

}

}

return (c22 + c11) / 2;

}

double investigation() {

double d;

double c;

double minD = 7;

double minC = 3;

for (int t = 0; t < 60; t++) {

d = randPoint(-1, 5);

c = cInvestigation(d);

if (minSq(c, d) < minSq(minC, minD)) {

minC = c;

minD = d;

}

}

std::cout << "с\* = " << minC << " d\* = " << minD << std::endl;

}

};

int main() {

Function func(-2, 1);

func.withoutNoise();

return 0;

}

**Ссылка на git-репозиторий:** https://github.com/SerdukovaM/TS\_lab5