Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №4

за 1 семестр

По дисциплине: «МиАПР»

Тема: «Нелинейные ИНС в задачах прогнозирования. Адаптивный шаг»

Выполнила:

Студентка 2 курса

Группы ПО-4(1)

Байдук Я. А.

Проверил:

Крощенко А.А.

2020

Лабораторная работа №4

Нелинейные ИНС в задачах прогнозирования. Адаптивный шан

Цель работы: изучить обучение и функционирование нелинейной ИНС при решении задач прогнозирования.

Вариант 2

Задание:

Спрогнозировать нелинейный временной ряд, применяя параметры лабораторной работы №3. При этом необходимо использовать алгоритм обучения многослойной ИНС с адаптивным шагом.

Для тестирования использовать функцию



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| вариант | a | b | с | d | Кол-во входов ИНС | Кол-во НЭ в скрытом слое |
| 2 | 0.2 | 0.2 | 0.06 | 0.2 | 8 | 3 |

Код программы:

#include <iostream>

using namespace std;

class InputNeuron {

public:

double Wes;

void random\_Wes() {

Wes = 2 \* ((rand() % 10) \* 0.1) - 1;

}

void izm\_ves(double a, double y, double t, double x) {

Wes -= a \* (y - t) \* x;

}

void set\_Wes(int Wes) {

this->Wes = Wes;

}

double get\_Wes() {

return Wes;

}

};

class HiddenNeuron {

public:

double Wes;

double Sum, znach;

void random\_Wes() {

Wes = 2 \* ((rand() % 10) \* 0.1) - 1;

}

void izm\_ves(double a, double y1, double gamma, double y2) {

Wes -= a \* gamma \* y1 \* (1 - y1) \* y2;

}

};

int main() {

setlocale(LC\_ALL, "ru");

system("color f0");

const int number = 30,

num\_input\_neuron = 8,

num\_hidden\_neuron = 3;

int sigm\_alpha = 0.2;//скорость обучения (исп. сигмоидной ф-ции

double a = 0.2, b = 0.2, c = 0.06, d = 0.2;

double Em = 0.00001,

A = 0.1,

Emax;

double standart\_y[number + 15], y,

Tx[num\_hidden\_neuron],

Ty = 2 \* ((rand() % 10) \* 0.1) - 1,

Error,

Error\_i[number - num\_input\_neuron];

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

double x = 0.1 \* i;

standart\_y[i] = a \* cos(b \* x) + c \* sin(d\*x);

}

InputNeuron W[num\_hidden\_neuron\*num\_input\_neuron];

HiddenNeuron Sig[num\_hidden\_neuron];

for (int i = 0; i < num\_input\_neuron\*num\_hidden\_neuron; i++) {

W[i].random\_Wes();

}

for (int i = 0; i < num\_hidden\_neuron; i++) {

Sig[i].random\_Wes();

Sig[i].Sum = 0;

Tx[i] = 2 \* ((rand() % 10) \* 0.1) - 1;

}

do {

y = 0;

Emax = 0;

for (int Prohod = 0; Prohod < number - num\_input\_neuron; Prohod++) {

double A1 = 0;

y = 0;

for (int i = 0; i < num\_hidden\_neuron; i++) {

for (int j = 0; j < num\_input\_neuron; j++) {

Sig[i].Sum += W[j].Wes \* standart\_y[Prohod + j];

}

Sig[i].Sum -= Tx[i];

Sig[i].znach = (1 / (1 + exp(-Sig[i].Sum)));

Sig[i].Sum = 0;

}

for (int i = 0; i < num\_hidden\_neuron; i++) {

y += Sig[i].znach\*Sig[i].Wes;

}

y -= Ty;

Error = y - standart\_y[Prohod + num\_input\_neuron];

Error\_i[Prohod] = Error;

for (int i = 0; i < num\_hidden\_neuron; i++) {

Sig[i].Wes -= A \* Sig[i].znach\*Error;

}

Ty += A \* Error;

for (int i = 0; i < num\_hidden\_neuron; i++) {

for (int j = 0; j < num\_input\_neuron; j++)

W[(i \* 10) + j].Wes -= sigm\_alpha \* standart\_y[Prohod + j] \* (Sig[i].Wes\*Error\*Sig[i].znach\*(1 - Sig[i].znach));

Tx[i] += sigm\_alpha \* (Sig[i].Wes\*Error\*Sig[i].znach\*(1 - Sig[i].znach));

}

double Summa = 0;

for (int i = 0; i < num\_hidden\_neuron; i++) {

Summa += Sig[i].znach;

}

if (A1 != A) {

A = 1 / (1 + Summa);

}

A1 = A;

//использование адаптивного шага обучения

double Err1 = 0,//числитель

Err2 = 0,//1 скобки знаменателя

Err3 = 0; // 2 скобки знаменателя

for (int i = 0; i < num\_hidden\_neuron; i++) {

Err1 = pow(Sig[i].Wes \* Error \* Sig[i].znach \* (1 - Sig[i].znach), 2.0) \* Sig[i].znach \* (1 - Sig[i].znach);

}

Err1 \*= 4;

Err2 = 1 + pow(standart\_y[Prohod + num\_input\_neuron], 2.0);

for (int i = 0; i < num\_hidden\_neuron; i++) {

Err3 = pow(Sig[i].Wes \* Error \* Sig[i].znach \* (1 - Sig[i].znach), 2.0) \* pow(Sig[i].znach, 2.0) \* pow((1 - Sig[i].znach), 2.0);

}

sigm\_alpha = Err1 / (Err2 \* Err3); // выражение адаптивного шага обучения(формула 24)

}

for (int i = 0; i < number - num\_input\_neuron; i++) {

Emax += (pow(Error\_i[i], 2.0) \* 0.5);

}

cout << Emax << endl;

} while (Emax > Em);

cout << "Эталон" << "\t\t\t\t" << "Прогноз" << "\t\t\t\tОтклонение" << endl;

for (int Proh = 0; Proh < 15; Proh++) {

y = 0;

for (int i = 0; i < num\_hidden\_neuron; i++) {

for (int j = 0; j < num\_input\_neuron; j++) {

Sig[i].Sum += W[j].Wes \* standart\_y[number - num\_input\_neuron + Proh + j];

}

Sig[i].Sum -= Tx[i];

Sig[i].znach = (1 / (1 + exp(-Sig[i].Sum)));

Sig[i].Sum = 0;

}

for (int i = 0; i < num\_hidden\_neuron; i++) {

y += Sig[i].znach \* Sig[i].Wes;

}

y -= Ty;

standart\_y[number + Proh] = y;

double x = 0.1\*((double)Proh + (double)number);

cout << a \* cos(b \* x) + c \* sin(d\*x) << "\t\t\t";

cout << standart\_y[number + Proh] << "\t\t\t";

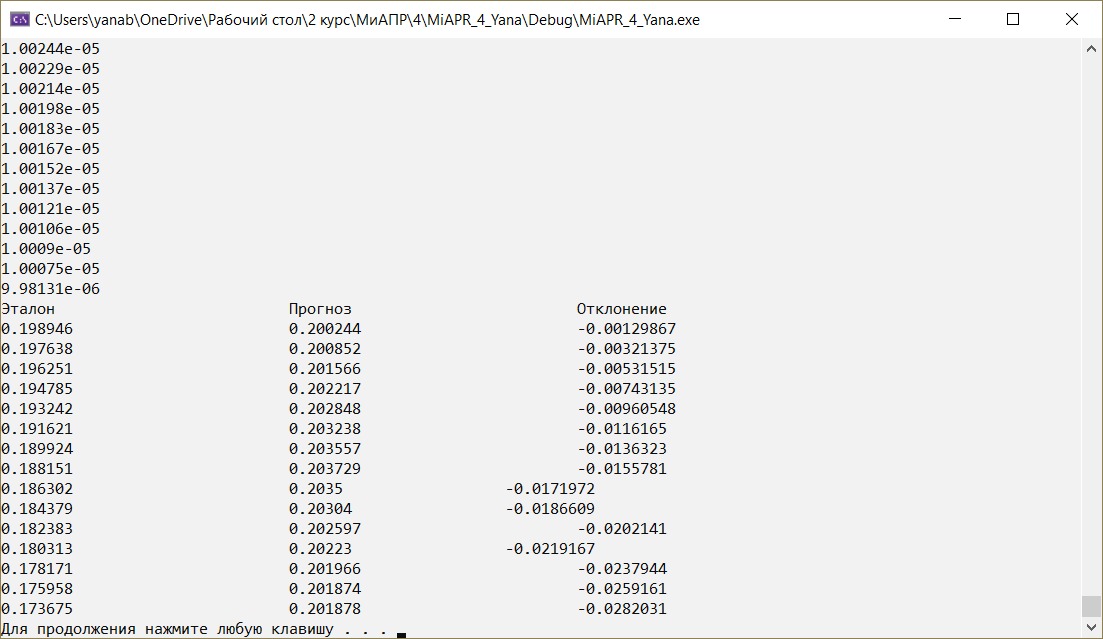
cout << a \* cos(b \* x) + c \* sin(d \* x) - standart\_y[number + Proh] << endl;

}

system("pause");

return 0;

}



Вывод: изучила обучение и функционирование нелинейной ИНС при решении задач прогнозирования.