***Министерство образования Республики Беларусь***

***Учреждение Образования***

***«Брестский Государственный Технический Университет»***

***Кафедра ИИТ***

**Лабораторная работа №6**

**По дисциплине «Традиционные и интеллектуальные**

**информационные технологии»**

**Тема: «Итерационные методы поиска экстремумов»**

**Выполнил:**

Студент 1-го курса

Группы ИИ-15(1)

Волк И. А.

**Проверил:**

Анфилец С. В.

Брест 2018

Цель: найти экстремумы методом градиентного спуска.

Ход работы

Задание и Ход работы:  
1. Ознакомиться с понятием градиента (градиент.pdf) и методом градиентного спуска. (Вспомнить  
дифференцирование функций)  
2. Написать программу поиска максимумов и минимумов функции F(x,y,z) методом градиентного поиска  
(с постоянным шагом).  
Входные данные в программу: Пользователь вводит диапазон поиска и стартовую точку поиска и  
выбирает направление поиска (min, max). (Стартовую точку поиска можно генерировать случайно в  
указанном диапазоне)  
Выходные данные: Найденный экстремум — точка и соответствующее значение функции.  
Вид функции и Варианты:  
F(x,y,z) = f1\*f2 + f3\*f4

7 | xy+1| z3 | z + 4x | z+1

Код программы:

#include "stdafx.h"

#include <Windows.h>

#include <iostream>

#include <cstdlib>

#include <conio.h>

#include <cmath>

#include <stdarg.h>

using namespace std;

void bubbleSort(double\*, int);

void menu();

double function(double[]);

void inputMinAndMax(double\*\*, int);

double \* findExtremum( double\*\*, int, double(\*)(double[]), bool);

double \* findGrad(double\*, int, double(\*)(double[]), bool);

bool fit(double\*, int, double\*\*);

bool equal(double\*, double\*, int);

int main()

{

const unsigned NUM\_OF\_VARIABLES = 3;

while (1)

{

double \*\* minAndMax = new double\*[NUM\_OF\_VARIABLES];

for (int i = 0; i < NUM\_OF\_VARIABLES; i++)

minAndMax[i] = new double[2];

menu();

inputMinAndMax(minAndMax, NUM\_OF\_VARIABLES);

double \* min = findExtremum(minAndMax, NUM\_OF\_VARIABLES, &function, false);

double \* max = findExtremum(minAndMax, NUM\_OF\_VARIABLES, &function, true);

cout << "Min coordinates:" << endl;

for (int i = 0; i < NUM\_OF\_VARIABLES; i++)

cout << "Variable #" << i + 1 << ": " << min[i] << endl;

cout << "\nMin value: " << function(min) << endl << endl;

cout << "Max coordinates:" << endl;

for (int i = 0; i < NUM\_OF\_VARIABLES; i++)

cout << "Variable #" << i + 1 << ": " << max[i] << endl;

cout << "\nMax value: " << function(max) << endl << endl;

system("pause");

}

system("pause");

return 0;

}

void menu()

{

system("cls");

cout << "1. Find the extremum\n2. Exit";

while (1)

{

int input = \_getch();

if (input == 50)

{

system("cls");

cout << "Good lock!";

Sleep(1000);

exit(0);

}

else if (input == 49)

break;

}

}

void inputMinAndMax(double \*\* arr, int numOfVar)

{

system("cls");

for (int i = 0; i < numOfVar; i++)

{

cout << "Variable #" << i + 1 << endl;

cout << "Min: "; cin >> arr[i][0];

cout << "Max: "; cin >> arr[i][1];

bubbleSort(arr[i], 2);

cout << endl;

}

system("cls");

}

double \* findExtremum(double \*\* minAndMax, int numOfVar, double(\*function)(double[]), bool max = true)

{

double \* position = new double[numOfVar];

for (int i = 0; i < numOfVar; i++)

position[i] = (double)(rand() % (int)(minAndMax[i][1] - minAndMax[i][0])) + minAndMax[i][0];

while (1)

{

double \* grad = findGrad(position, numOfVar, function, max);

double \* newPosition = new double[numOfVar];

for (int i = 0; i < numOfVar; i++)

newPosition[i] = position[i];

for (int i = 0; i < numOfVar; i++)

{

newPosition[i] += grad[i] \* 0.01;

if (!fit(newPosition, numOfVar, minAndMax))

newPosition[i] -= grad[i] \* 0.01;

}

if (equal(position, newPosition, numOfVar))

return position;

if (((max)?(function(newPosition) - function(position))

:(function(position) - function(newPosition))) < 0.000001)

return newPosition;

for (int i = 0; i < numOfVar; i++)

position[i] = newPosition[i];

}

return position;

}

double \* findGrad(double \* position, int numOfVar, double(\*function)(double[]), bool max)

{

double \* grad = new double[numOfVar];

for (int i = 0; i < numOfVar; i++)

{

double \* newPosition = new double[numOfVar];

for (int j = 0; j < numOfVar; j++)

newPosition[j] = position[j];

newPosition[i] += 0.001;

grad[i] = (max)?(function(newPosition) - function(position))

:(function(position) - function(newPosition));

}

double length = 1.0;

for (int i = 0; i < numOfVar; i++)

length += (pow(grad[i], 2.0));

length = pow(length, 0.5);

for (int i = 0; i < numOfVar; i++)

grad[i] /= length;

return grad;

}

bool fit(double\* coordinates, int numOfVar, double\*\* minAndMax)

{

for (int i = 0; i < numOfVar; i++)

if (coordinates[i]<minAndMax[i][0] || coordinates[i]>minAndMax[i][1])

return false;

return true;

}

double function(double arrOfVar[])

{

return ((arrOfVar[0]\* arrOfVar[1] + 1)\*pow(arrOfVar[2], 3.0)

+ (arrOfVar[2]+4\* arrOfVar[0])\*(arrOfVar[2]+1));

}

void bubbleSort(double\* arr, int size)

{

for (int i = 1; i < size; i++)

if(arr[i-1]>arr[i])

{

int var = arr[i];

arr[i] = arr[i - 1];

arr[i - 1] = var;

}

}

bool equal(double\* arr1, double\* arr2, int size)

{

for (int i = 0; i < size; i++)

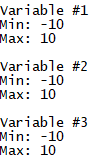
if (arr1[i] != arr2[i])

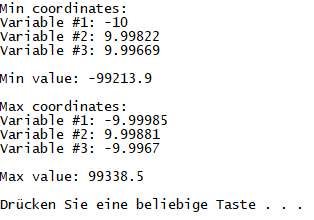
return false;

return true;

}

Output:





Вывод: по ходу лабораторной работы научился находить экстремум с помощью градиентного спуска.