МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Ижевский государственный технический университет имени М. Т. Калашникова

(ФГБОУ ВО «ИжГТУ имени М. Т. Калашникова»)

Кафедра «Программное обеспечение»

Отчет по лабораторной работе №2

по дисциплине «Исследование операций и теория принятия решений»

на тему «Критерии и критериальные функции»

Выполнили:

студенты группы Б18-191-2 Р. А. Гумметов

Я. П. Семёнов

Принял:

старший преподаватель П. П. Лугачев

Ижевск 2021

1. ПОСТАНОВКА ЗАДАЧИ

Исследование функции с помощью метода регулярного симплекса. Функция 20x2 + 30y2 + 50. График функции представлен на рисунке №1.

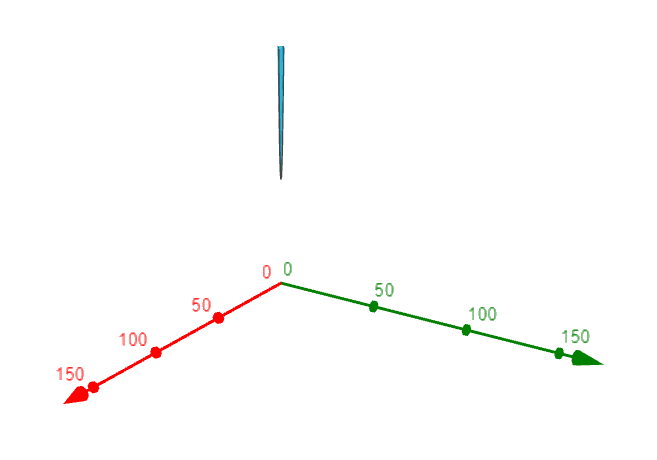


Рис.

2. ОПИСАНИЕ МЕТОДА

1. Вычислить значение во всех трех точках и найти среди значений максимум;
2. В том случае, если необходимо найти минимум, строится новый треугольник (симплекс), с новой точкой x4, которая противоположна максимальной точке. (Пример представлен на рисунке №2);
3. Пункты 1 и 2 повторяются до нахождения минимума.

Критерии прекращения алгоритма:

1. Значения функций на текущем и предыдущем шаге не сильно отличаются, например, разница составляет меньше погрешности (е Б 0.0001)
2. Зацикливание. В этом случае необходимо уменьшить симплекс.

1

2

3

4

3

1

Рис. 2

КОД ПРОГРАММЫ

#include"MyForm.h"

#include<list>

#include<algorithm>

#include<vector>

#include<string>

#include<cmath>

#defineM\_PI 3.14159265358979323846

usingnamespace System;

usingnamespace System::Windows::Forms;

[STAThreadAttribute]

void Main(array<String^>^ args) {

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

My2Lab::MyForm form;

Application::Run(% form);

}

double step = 4;

int c = 50;

publicclassMyPoint

{

public:

double x;

double y;

MyPoint(doublexx, doubleyy)

{

x = xx;

y = yy;

}

};

double function(MyPointa) { return (20 \* a.x \* a.x) + (30 \* a.y \* a.y) + c; }

System::Void My2Lab::MyForm::pictureBox1\_Click(System::Object^ sender, System::EventArgs^ e)

{

intn\_counter = 0;

MyPoint x1(2, 3);

MyPoint x2(x1.x + step / 2, x1.y + step);

MyPoint x3(x1.x + step, x1.y);

std::vector<MyPoint> list = { x1, x2, x3 };

gr = pictureBox1->CreateGraphics();

gr->TranslateTransform(pictureBox1->Width / 2, pictureBox1->Height / 2);

gr->RotateTransform((float)M\_PI);

pen = gcnewPen(Color::Red);

doublemaxprev = -1;

double max;

while (true)

{

label3->Text = Convert::ToString(n\_counter);

label3->Update();

n\_counter++;

gr->DrawLine(pen, (float)list[0].x \* 20, (float)list[0].y \* 20, (float)list[1].x \* 20, (float)list[1].y \* 20);

gr->DrawLine(pen, (float)list[1].x \* 20, (float)list[1].y \* 20, (float)list[2].x \* 20, (float)list[2].y \* 20);

gr->DrawLine(pen, (float)list[0].x \* 20, (float)list[0].y \* 20, (float)list[2].x \* 20, (float)list[2].y \* 20);

System::Threading::Thread::Sleep(200);

std::vector<double>maxlist;

for (auto i : list)

{

maxlist.push\_back(function(i));

}

max = \*std::max\_element(maxlist.begin(), maxlist.end());

//выход

doubleminmin = \*(std::min\_element(maxlist.begin(), maxlist.end()));

if (maxprev != -1 &&maxprev - max < 0.001 || minmin == c)

{

doubleff = \*(std::min\_element(maxlist.begin(), maxlist.end()));

if (\*(std::min\_element(maxlist.begin(), maxlist.end())) != c)

{

step /= 2;

x1=MyPoint(2, 3);

x2=MyPoint(x1.x + step / 2, x1.y + step);

x3=MyPoint(x1.x + step, x1.y);

list=std::vector<MyPoint>{ x1, x2, x3 };

maxprev = -1;

max = -100;

}

else {

label1->Text =

(System::Convert::ToString(\*std::min\_element(maxlist.begin(), maxlist.end())));

break;

}

}

int index = std::distance(maxlist.begin(),std::find(maxlist.begin(), maxlist.end(), max));

if (index == maxlist.size()) continue;

double x0 = 0;

double y0 = 0;

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

{

if (i != index)

{

x0 += list[i].x;

y0 += list[i].y;

}

}

x0 /= 2;

y0 /= 2;

double x = list[index].x;

double y = list[index].y;

list[index]=MyPoint(

x0 + (x - x0) \* cos(M\_PI) - (y - y0) \* sin(M\_PI),

y0 + (y - y0) \* cos(M\_PI) + (x - x0) \* sin(M\_PI));

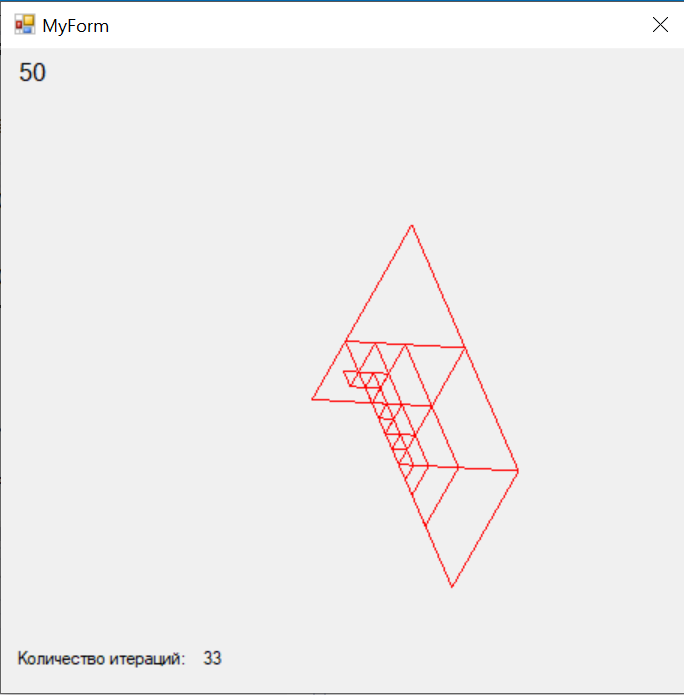
maxprev = max;

}

return System::Void();

}

ПРИМЕР РАБОТЫ ПРОГРАММЫ



ВЫВОД

В ходе лабораторной работы изучен и реализован метод регулярного симплекса.