Липецкий государственный технический университет

Кафедра АСУ

ДОМАШНЯЯ РАБОТА №4

по дисциплине

Математическое программирование

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Глубоков Г.В.

подпись, дата

Группа АИ-20-1

Руководитель

К.т.н., доцент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Качановский Ю.П.

подпись, дата

Липецк, 2022 г.

# 

# Задание кафедры:

Определить не менее 4 базисных точек методом Хука-Дживса

Вариант 7

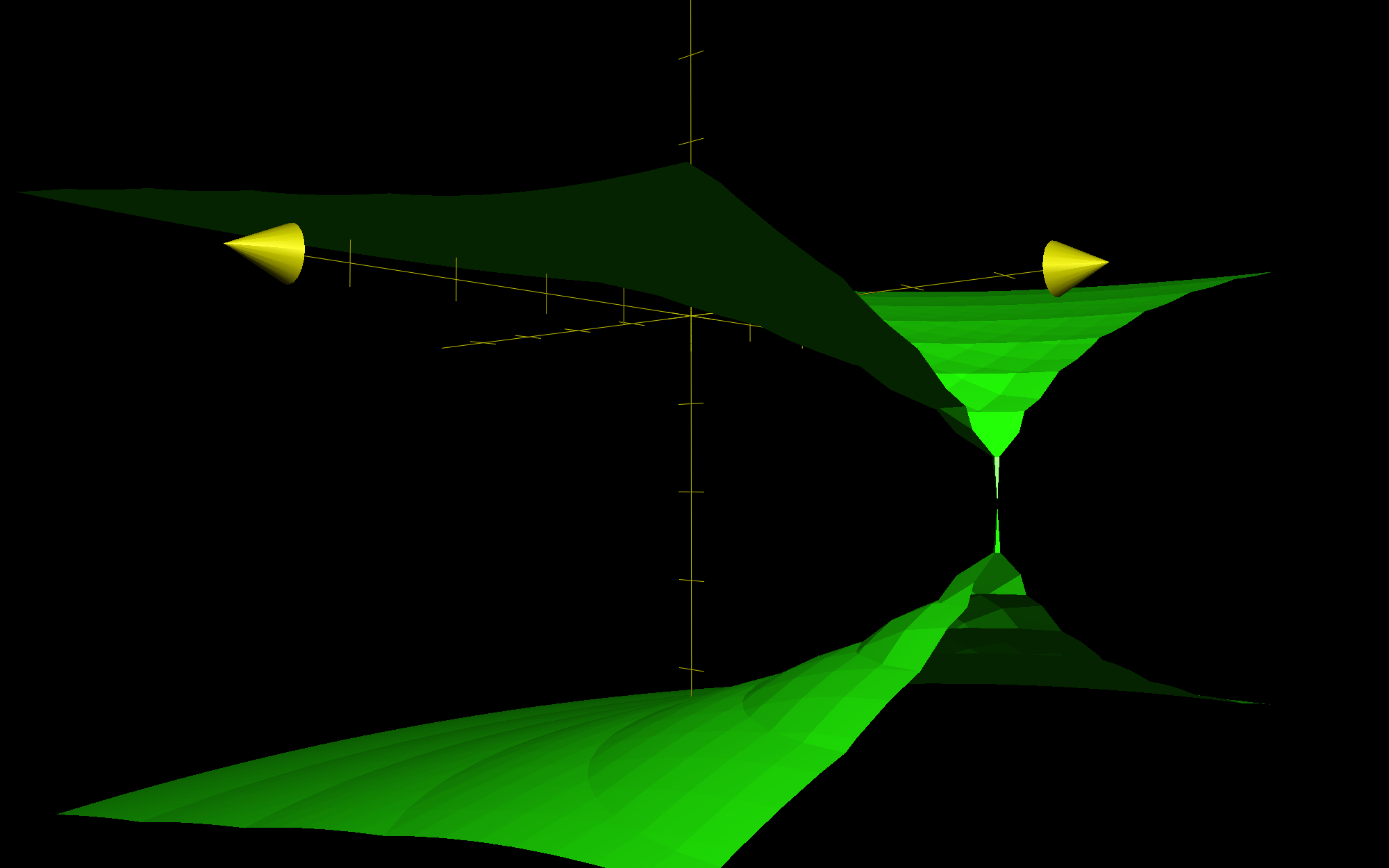


Рисунок 1- График функции

Ход выполнения домашнего задания

Найдем точку минимума и значение функции с помощью метода Хука-Дживса. Напишем код программы, представленный в приложении 1, скомпилируем и запустим ее.

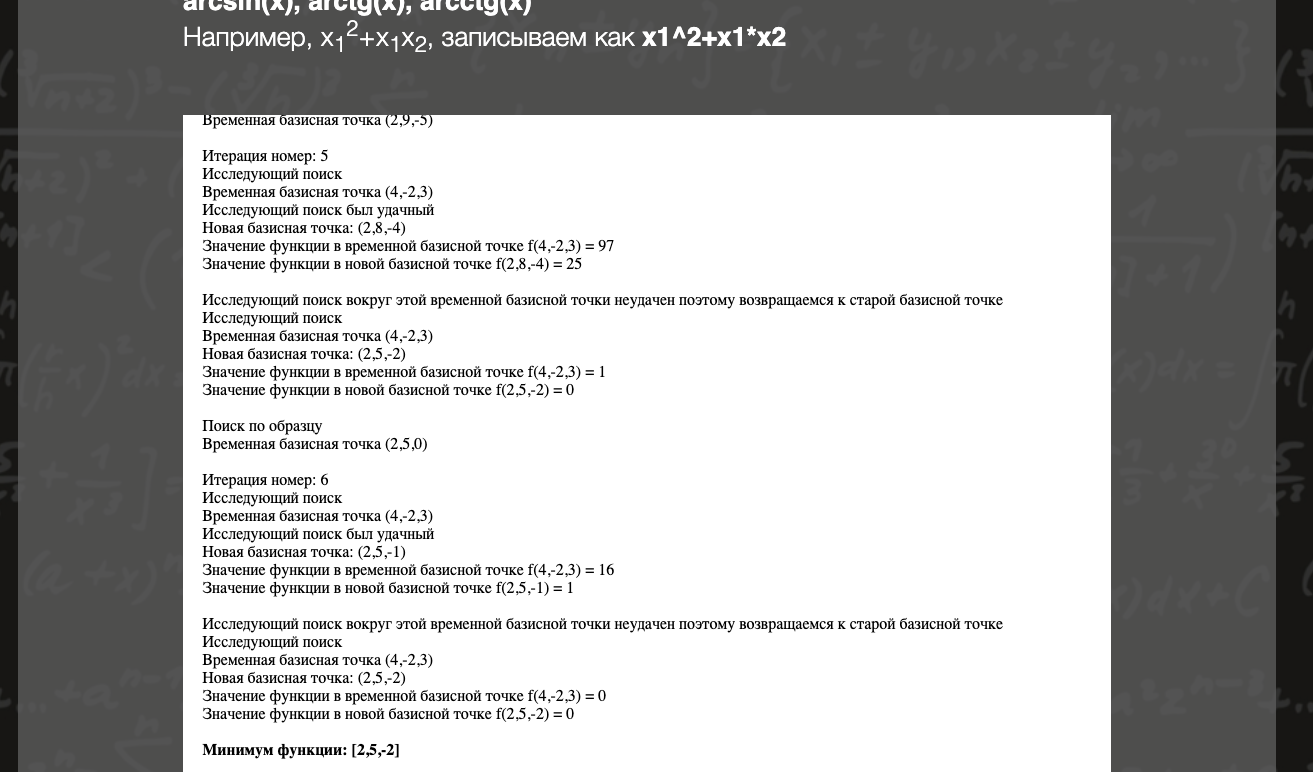
Получаем результат, изображенный на рисунке 2. 

Рисунок 2 – Метод Хука-Дживса

За 6 итераций была найдена точка минимума X = (2, 5, -2)

с значением функции F(x) = 0

**Полный результат программы**

**Начальные значения**  
Функция: (x1-2)^2+(x2-5)^2+(x3+2)^4  
Начальная точка: (4-23)  
Приращение: (111)  
альфа: 2  
Бетта: 2  
Эпсилон: 0.001

Итерация номер: 1  
Исследующий поиск  
Временная базисная точка (4,-2,3)  
Исследующий поиск был удачный  
Новая базисная точка: (3,-1,2)  
Значение функции в временной базисной точке f(4,-2,3) = 678  
Значение функции в новой базисной точке f(3,-1,2) = 293  
  
Поиск по образцу  
Временная базисная точка (1,1,0)

Итерация номер: 2  
Исследующий поиск  
Временная базисная точка (4,-2,3)  
Исследующий поиск был удачный  
Новая базисная точка: (2,2,-1)  
Значение функции в временной базисной точке f(4,-2,3) = 33  
Значение функции в новой базисной точке f(2,2,-1) = 10  
  
Поиск по образцу  
Временная базисная точка (0,8,-7)

Итерация номер: 3  
Исследующий поиск  
Временная базисная точка (4,-2,3)  
Исследующий поиск был удачный  
Новая базисная точка: (1,7,-6)  
Значение функции в временной базисной точке f(4,-2,3) = 638  
Значение функции в новой базисной точке f(1,7,-6) = 261  
  
Исследующий поиск вокруг этой временной базисной точки неудачен поэтому возвращаемся к старой базисной точке  
Исследующий поиск  
Временная базисная точка (4,-2,3)  
Новая базисная точка: (2,3,-2)  
Значение функции в временной базисной точке f(4,-2,3) = 10  
Значение функции в новой базисной точке f(2,3,-2) = 4  
  
Поиск по образцу  
Временная базисная точка (2,5,-4)

Итерация номер: 4  
Исследующий поиск  
Временная базисная точка (4,-2,3)  
Исследующий поиск был удачный  
Новая базисная точка: (2,5,-3)  
Значение функции в временной базисной точке f(4,-2,3) = 16  
Значение функции в новой базисной точке f(2,5,-3) = 1  
  
Поиск по образцу  
Временная базисная точка (2,9,-5)

Итерация номер: 5  
Исследующий поиск  
Временная базисная точка (4,-2,3)  
Исследующий поиск был удачный  
Новая базисная точка: (2,8,-4)  
Значение функции в временной базисной точке f(4,-2,3) = 97  
Значение функции в новой базисной точке f(2,8,-4) = 25  
  
Исследующий поиск вокруг этой временной базисной точки неудачен поэтому возвращаемся к старой базисной точке  
Исследующий поиск  
Временная базисная точка (4,-2,3)  
Новая базисная точка: (2,5,-2)  
Значение функции в временной базисной точке f(4,-2,3) = 1  
Значение функции в новой базисной точке f(2,5,-2) = 0  
  
Поиск по образцу  
Временная базисная точка (2,5,0)

Итерация номер: 6  
Исследующий поиск  
Временная базисная точка (4,-2,3)  
Исследующий поиск был удачный  
Новая базисная точка: (2,5,-1)  
Значение функции в временной базисной точке f(4,-2,3) = 16  
Значение функции в новой базисной точке f(2,5,-1) = 1  
  
Исследующий поиск вокруг этой временной базисной точки неудачен поэтому возвращаемся к старой базисной точке  
Исследующий поиск  
Временная базисная точка (4,-2,3)  
Новая базисная точка: (2,5,-2)  
Значение функции в временной базисной точке f(4,-2,3) = 0  
Значение функции в новой базисной точке f(2,5,-2) = 0

**Минимум функции: [2,5,-2]**

Приложение 1

const math = require('mathjs');

const parser = math.parser();

class Huk {

input\_data = {

// //sqrt((x-y)^2+(x^2+2-y-5)^2)

// //8x^2+4\*x\*y+5\*y^2

// //(x1-2)^2+(x2-5)^2+(x3+2)^4

// function: "(x1-2)^2+(x2-5)^2+(x3+2)^4",

// alpha: 2,

// betta: 2,

// eps: 0.0001,

// //1,1,1

// h: [1,1,1],

// //4,-2,3

// x0: [4,-2,3],

// type\_func: "f(x1,x2,x3)",

// epsilant: 0.1,

}

resES = [

{

flag: 1,

}

];

export\_data = {}

valueFunc(x) {

try {

let arg = "f(" + x.join(",") + ")";

parser.evaluate("f"+this.input\_data.type\_func + '=' + this.input\_data.function)

return (parser.evaluate(arg))

} catch (e) {

throw "Проверьте корректность введённых данных ( "+e+" )";

console.log(e);

}

}

reduceH(h) {

let flag = 0, i;

for (i = 0; i < h.length; i++) {

if (h[i] > this.input\_data.epsilant) {

h[i] /= this.input\_data.alpha;

flag = 1;

}

}

return {

flag: flag,

h: h,

};

}

exploratory\_search(x0, h) {

let time\_base\_point = x0.slice(0), i, new\_base\_point, j, flag = 1, z\_tbp, z\_x0;

let new\_h = {}

for (j = 0; flag === 1; j++) {

console.log("h= " + h);

for (i = 0; i < x0.length; i++) {

time\_base\_point[i] += h[i];

z\_tbp = this.valueFunc(time\_base\_point);

z\_x0 = this.valueFunc(x0);

if (z\_tbp < z\_x0) {

continue;

} else {

time\_base\_point[i] -= 2 \* h[i];

z\_tbp = this.valueFunc(time\_base\_point);

z\_x0 = this.valueFunc(x0);

if (z\_tbp < z\_x0) {

continue;

} else {

time\_base\_point[i] += h[i];

}

}

}

if (z\_tbp < z\_x0) {

new\_base\_point = time\_base\_point.slice(0);

time\_base\_point = x0.slice(0);

break;

} else {

new\_h = this.reduceH(h.slice(0));

h = new\_h.h;

if (new\_h.flag === 0) {

flag = new\_h.flag;

new\_base\_point = x0;

break;

}

}

}

return {

time\_base\_point,

new\_base\_point,

flag,

}

}

search\_obr(x1, x0, b) {

try{

let i, x2 = [];

for (i = 0; i < this.input\_data.x0.length; i++) {

x2[i] = x1[i] + b \* (x1[i] - x0[i]);

}

return x2;

}

catch (e){

throw "Ошибка сервера";

}

}

main(req) {

this.export\_data = {};

try {

this.export\_data = {

request\_params: [

{

iteration: 0,

initial\_data: {},

ES: {

h: [],

time\_base\_point: this.input\_data.x0,

f\_NBP: 0,

f\_TBP: 0,

new\_base\_point: this.input\_data.x0,

},

SO: {

new\_base\_point: this.input\_data.x0,

f\_NBP: 0,

f\_TBP: 0,

time\_base\_point: this.input\_data.x0,

},

}

]

}

this.input\_data = req;

this.export\_data.request\_params[0].initial\_data.function = req.function;

this.export\_data.request\_params[0].initial\_data.x0 = req.x0;

this.export\_data.request\_params[0].initial\_data.h = req.h;

this.export\_data.request\_params[0].initial\_data.alpha = req.alpha;

this.export\_data.request\_params[0].initial\_data.betta = req.betta;

this.export\_data.request\_params[0].initial\_data.epsilant = req.epsilant;

this.resES[0].time\_base\_point = this.input\_data.x0;

this.resES[0].new\_base\_point = this.input\_data.x0;

let flag = 0;

console.log(" H = " + this.input\_data.h + ", Альфа = " + this.input\_data.alpha + ", Бетта = " + this.input\_data.betta);

console.log("Начальная базисная точка: " + this.input\_data.x0);

console.log("Номер итерации " + 1);

this.export\_data.request\_params[1] = {

iteration: 1,

ES: {},

SO: {},

};

this.resES[1] = this.exploratory\_search(this.resES[0].time\_base\_point.slice(0), this.input\_data.h.slice(0));

this.export\_data.request\_params[1].ES.time\_base\_point = this.resES[0].time\_base\_point;

this.export\_data.request\_params[1].ES.new\_base\_point = this.resES[1].new\_base\_point;

this.export\_data.request\_params[1].ES.f\_TBP = this.valueFunc(this.resES[0].time\_base\_point);

this.export\_data.request\_params[1].ES.f\_NBP = this.valueFunc(this.resES[1].new\_base\_point);

console.log("Новая базисная точка: " + this.resES[1].new\_base\_point)

if (this.resES[0].flag === 0) {

console.log("Минимум функции =" + this.resES[i].new\_base\_point);

flag = 1;

}

for (let i = 2; flag === 0; i++) {

this.resES[i] = {};

this.export\_data.request\_params[i] = {

iteration: i,

ES: {},

SO: {},

msg: "",

DES: {}

};

console.log("Поиск по образцу");

this.resES[i].time\_base\_point = this.search\_obr(this.resES[i - 1].new\_base\_point, this.resES[i - 2].new\_base\_point, this.input\_data.betta);

this.export\_data.request\_params[i - 1].SO.new\_base\_point = this.resES[1].new\_base\_point;

this.export\_data.request\_params[i - 1].SO.time\_base\_point = this.resES[i].time\_base\_point;

console.log("Временная базисная точка: " + this.resES[i].time\_base\_point)

console.log("Номер итерации " + i);

console.log("Исследующий поиск");

console.log("F(" + this.resES[i].time\_base\_point + ") = " + this.valueFunc(this.resES[i].time\_base\_point));

this.resES[i] = this.exploratory\_search(this.resES[i].time\_base\_point.slice(0), this.input\_data.h.slice(0));

this.export\_data.request\_params[i].ES.time\_base\_point = this.resES[1].time\_base\_point;

this.export\_data.request\_params[i].ES.new\_base\_point = this.resES[i].new\_base\_point;

this.export\_data.request\_params[i].ES.f\_TBP = this.valueFunc(this.resES[i].time\_base\_point);

this.export\_data.request\_params[i].ES.f\_NBP = this.valueFunc(this.resES[i].new\_base\_point);

console.log("Новая базисная точка: " + this.resES[i].new\_base\_point)

if (this.resES[i].flag === 0) {

console.log("Минимум функции =" + this.resES[i].new\_base\_point);

flag = 1;

break;

}

if (this.valueFunc(this.resES[i].new\_base\_point) < this.valueFunc(this.resES[i - 1].new\_base\_point)) {

continue;

} else {

console.log(this.valueFunc(this.resES[i].new\_base\_point) + ">" + this.valueFunc(this.resES[i - 1].new\_base\_point))

console.log("Исследующий поиск вокруг этой временной базисной точки неудачен поэтому возвращаемся к старой базисной точке")

this.export\_data.request\_params[i].msg = "Исследующий поиск вокруг этой временной базисной точки неудачен поэтому возвращаемся к старой базисной точке";

this.resES[i].time\_base\_point = this.resES[i - 1].new\_base\_point.slice(0);

console.log("Исследующий поиск");

console.log("F(" + this.resES[i].time\_base\_point + ") = " + this.valueFunc(this.resES[i].time\_base\_point));

this.resES[i] = this.exploratory\_search(this.resES[i].time\_base\_point.slice(0), this.input\_data.h.slice(0));

this.export\_data.request\_params[i].DES.time\_base\_point = this.resES[1].time\_base\_point;

this.export\_data.request\_params[i].DES.new\_base\_point = this.resES[i].new\_base\_point;

this.export\_data.request\_params[i].DES.f\_TBP = this.valueFunc(this.resES[i].time\_base\_point);

this.export\_data.request\_params[i].DES.f\_NBP = this.valueFunc(this.resES[i].new\_base\_point);

console.log("Новая базисная точка: " + this.resES[i].new\_base\_point)

if (this.resES[i].flag === 0) {

console.log("Минимум функции =" + this.resES[i].new\_base\_point);

flag = 1;

break;

}

}

}

this.export\_data.request\_params.forEach(rp => {

if (Object.keys(rp.SO).length === 0) {

delete rp.SO

}

})

return this.export\_data;

}

catch (e){

console.log(e)

this.export\_data.err=e;

return this.export\_data;

}

}

}

module.exports = Huk;

На рисунках 3 и 4 изображён пример нахождения минимума функции в программе MoDS (2 ЛР)

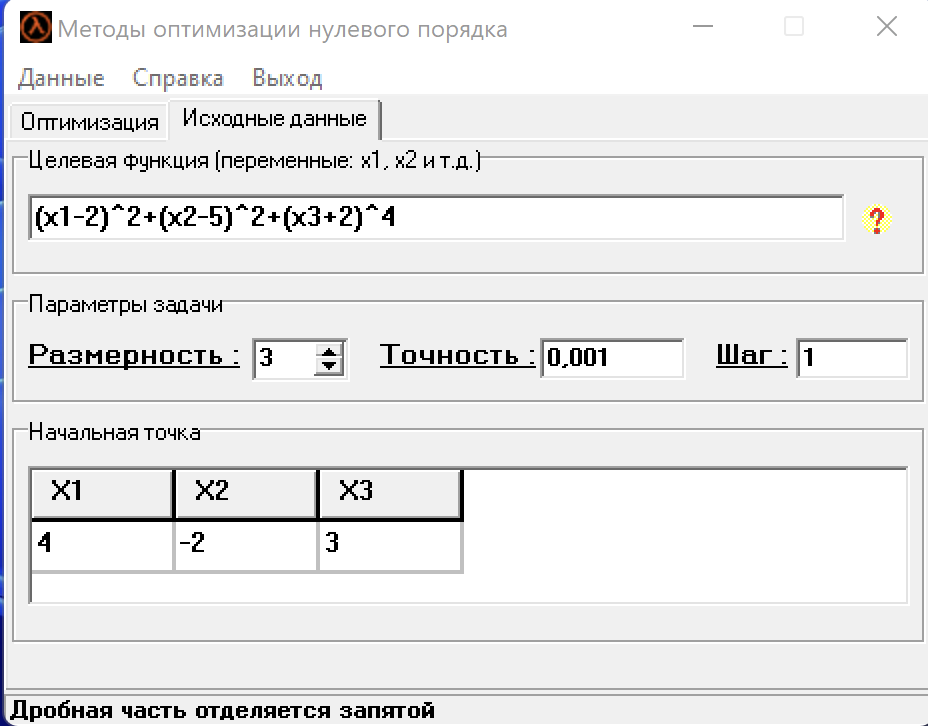


Рисунок 3 – Исходная функция

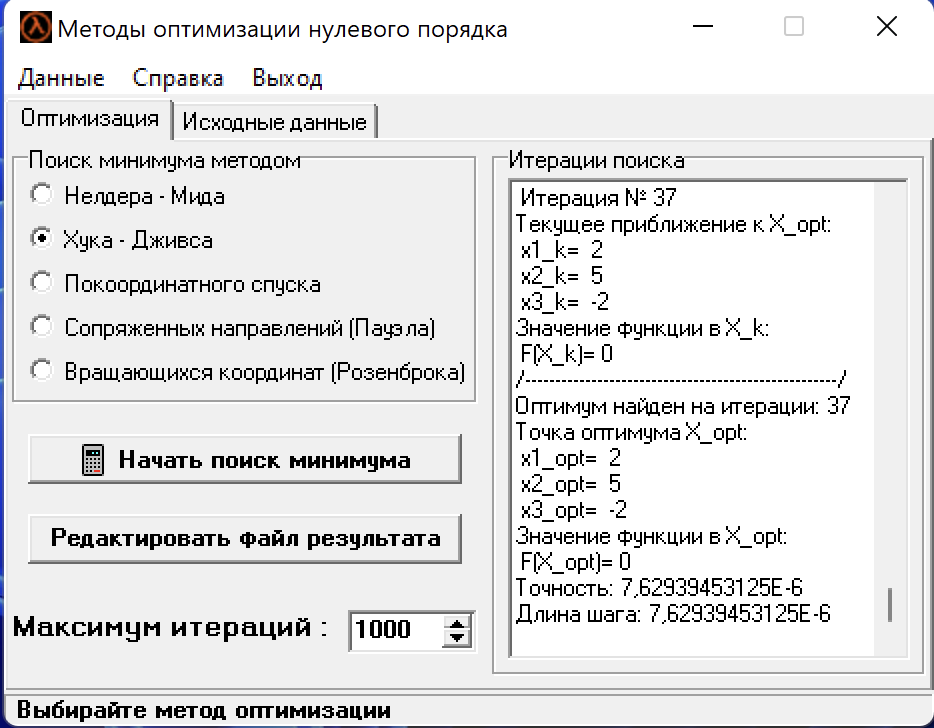


Рисунок 4 – Результат выполнения