МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И РОБОТОТЕХНИКИ

Кафедра “Системы автоматизированного проектирования”

**Пояснительная записка к курсовому проекту**

по дисциплине “Оптимизация проектных решений”

**«**Подсистема принятия решений на основании методов Гаусса, метода Нелдер-Мида, главного и minmax критереев**»**

**Выполнил:** студент группы 10702314 Краско И.С.

**Проверил:** доцентКовалева И. Л.

Минск 2017

СОДЕРЖАНИЕ

[Введение 3](#_Toc247903894)

[1 Постановка задачи 5](#_Toc247903895)

[2 Описание алгоритмов](#_Toc247903896) 7

[3 Программная реализация алгоритмов](#_Toc247903896) 13

[3.1 Реализация нескольких окон](#_Toc247903898) 13

[3.2 Реализация алгоритмов](#_Toc247903898) 14

[4 Руководство пользователя 1](#_Toc247903902)6

5 Тестирование…………………………………………………………………..28

5.1 Тестирование основных методов ……………………………………….28

[Заключение](#_Toc247903911) 34

[Приложение](#_Toc247903911) 35

[Литература](#_Toc247903912) 36

ВВЕДЕНИЕ

Основная особенность оптимизации проектных решений заключается в том, что вни.мание конструктора, главным образом, должно быть сосредоточено на выборе метода синтеза возможных решений. И только после того, как множество возможных решений сформировано либо найден эффективный алгоритм формирования возможных решений, приступают к выбору метода поиска оптимального решения. По существу все методы оптимизации относятся к поиску оптимума на дискретном множестве возможных решений. В отдельных случаях, когда множество возможных решений представляется какой-либо структурой определенного класса, могут применяться известные методы оптимизации. В остальных случаях единственной альтернативой методу глобального перебора служит тот или иной метод ограниченного перебора возможных вариантов проектных решений.

Как правило, математические модели, применяемые при проектировании, характеризуются наличием значительного количества оптимизированных параметров и критериев оптимальности. При этом постановка задачи может меняться от простой линейной однокритериальной до явно выраженных многокритериальных. Исходя из вышеизложенного подсистема принятия решений, предназначеннная для оптимизации, должна включать в себя:

* программные средства для решения задач однокритериальной оптимизации;
* программные средства, включающие алгоритмы и их реализацию, для решения задач многокритериальной оптимизации;
* интерфейсные средства, позволяющие легко взаимодействовать с подсистемой пользователю с минимальными знаниями в данной области.

Для создания такой подсистемы была выбрана платформа для разработки – MATLAB. MATLAB — это высокоуровневый язык и интерактивная среда для программирования, численных расчетов и визуализации результатов.

Пользуясь MATLAB, Вы сможете:

* Производить всевозможные операции над матрицами, решать линейные уравнения, работать с векторами;
* Вычислять корни многочленов любой степени, производить операции над многочленами, дифференцировать, экстраполировать и интерполировать кривые, строить графики любых функций;
* Проводить статистический анализ данных с использованием цифровой фильтрации, статистической регрессии;
* Решать дифференциальные уравнения. В частных производных, линейных, нелинейных, с граничными условиями – не важно, матлаб все решит;
* Выполнять операции целочисленной арифметики.

**1 Постановка задачи**

Целью курсового проекта является разработка подсистемы принятия решений. Для этого следует выполнить следующие задачи:

1. Проанализировать методы однокритериальной оптимизации: деление пополам, метод Гаусса, методы многокритериальной оптимизации: метод Нелдер-Мида, метод главного критерия, минмакс;
2. Разработать программу для реализацию указанных выше методов
3. Руководство пользователя
4. Выполнить тестирование.

Тема курсового проекта актуальна.

Исходные данные:

1. Платформа для разработки - MATLAB
2. Методы однокритериальной оптимизации:

* Однокритериальный метод – деление пополам;
* Двухкритериальный метод – метод Гаусса. Дополнительно пользователь вводит ограничение на каждый из параметров;

1. Методы многокритериальной оптимизации:

Для всех методов многокритериальной оптимизации пользователь вводит количество критериев, ограничения на каждый из критериев, значения функций, а также значение направления поиска оптимума (на max или на min);

* Метод главного критерия. Дополнительно пользователь задает какой из критериев будет главным и вводит ограничения на остальные критерии;
* Метод minmax;
* Метод Нелдер-Мида, количество параметров – два.

На основании выше изложенного была разработана функциональная схема, представленная на cхемах 1.1, 1.2.

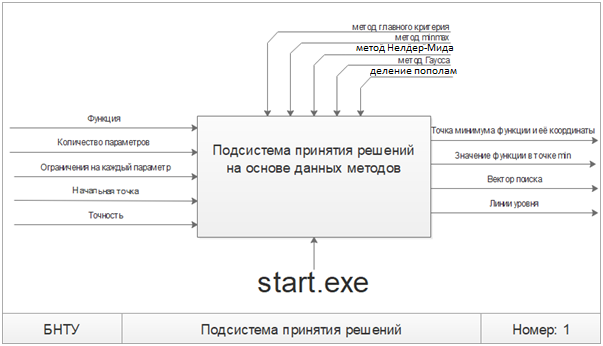


Рис 1.1 – Схема IDEF0

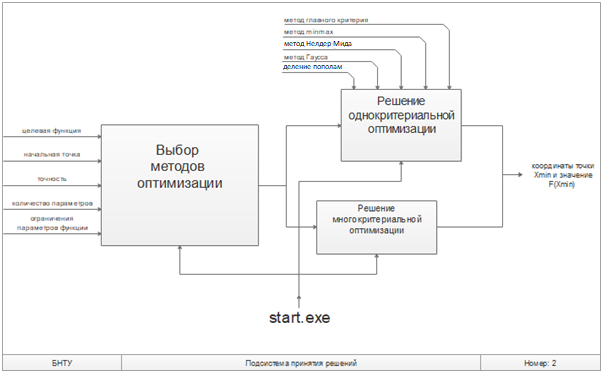


Рис 1.2 – Схема IDEF0

**2 Описание используемых алгоритмов**

Для реализации поиска минимума многопараметрической функции был задан метод Нелдер-Мида.

*Алгоритм «Метод Нелдер-Мида»*

Метод Нелдера — Мида, также известный как метод деформируемого многогранника и симплекс-метод, — метод безусловной оптимизации функции от нескольких переменных, не использующий производной (точнее — градиентов) функции, а поэтому легко применим к негладким и/или зашумлённым функциям.(рис 2.1).

*Изложение метода*

Параметрами метода являются:

* коэффициент отражения \alpha >0, обычно выбирается равным 1.
* коэффициент сжатия \beta >0, обычно выбирается равным 0.5.
* коэффициент растяжения \gamma >0, обычно выбирается равным 2.

Инициализация.Произвольным образом выбирается n+1 точка x_i = \left(x^{(1)}_i,x^{(2)}_i,\ldots,x^{(n)}_i\right), образующие симплекс n-мерного пространства. В этих точках вычисляются значения функции: f_1=f(x^{(1)}_i), f_2=f(x^{(2)}_i),\ldots, f_{n+1}=f(x^{(n+1)}_i).

1. Сортировка. Из вершин симплекса выбираем три точки: x_h с наибольшим (из выбранных) значением функции f_h, x_g со следующим по величине значением f_g и x_l с наименьшим значением функции f_l. Целью дальнейших манипуляций будет уменьшение по крайней мере f_h.

2. Найдём центр тяжести всех точек, за исключением x_h: x_c=\frac{1}{n} \sum_{i\neq h} x_i . Вычислять f_c=f(x_c) не обязательно.

3. Отражение. Отразим точку x_h относительно x_c с коэффициентом \alpha (при \alpha=1 это будет центральная симметрия, в общем случае — гомотетия), получим точку x_r и вычислим в ней функцию: f_r=f(x_r). Координаты новой точки вычисляются по формуле x_r = (1+\alpha)x_c - \alpha x_h

4. Далее сравниваем значение f_r со значениями f_h, f_g, f_l:

4а. Если f_r<f_l, то производим растяжение. Новая точка x_e = (1-\gamma)x_c + \gamma x_r и значение функции f_e=f(x_e).

Если f_e<f_l, то заменяем точку x_h на x_e и заканчиваем итерацию (на шаг 8).

Если f_e>f_l, то заменяем точку x_h на x_r и заканчиваем итерацию (на шаг 8).

4b. Если f_l < f_r < f_g, то заменяем точку x_h на x_r и переходим на шаг 8.

4с. Если f_h > f_r > f_g, то меняем обозначения x_r, x_h (и соответствующие значения функции) местами и переходим на шаг 5.

4d. Если f_r > f_h, то переходим на шаг 5.

5. Сжатие. Строим точку x_s = \beta x_h + (1-\beta) x_c и вычисляем в ней значение f_s

6. Если f_s < f_h, то заменяем точку x_h на x_s и переходим на шаг 8.

7. Если f_s > f_h, то производим сжатие симплекса — гомотетию к точке с наименьшим значением x_0: x_i \to x_0 + (x_i-x_0)/2 для всех требуемых точек x_i.

8. Последний шаг — проверка сходимости. Может выполняться по-разному, например, оценкой дисперсии набора точек. Суть проверки заключается в том, чтобы проверить взаимную близость полученных вершин симплекса, что предполагает и близость их к искомому минимуму. Если требуемая точность ещё не достигнута, можно продолжить итерации с шага 1.

Схема нахождения минимума многопараметрической функции методом адаптивного случайного поиска представлена на рисунке 2.2.

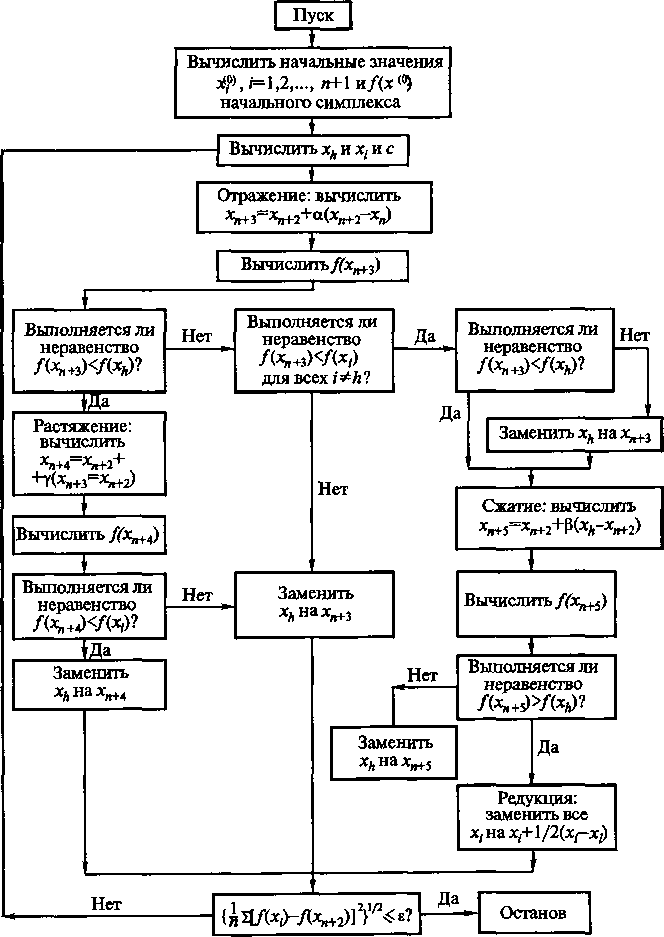


Рисунок. 2.1 – Схема метода Нелдер-Мида

*Алгоритм «Деление пополам»*

Разделим интервал [a,b] на две равные части (рисунок 2.2), а затем каждую из частей еще на две равные части.

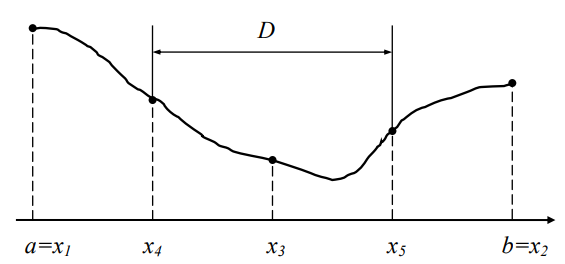


Рисунок 2.2 Метод деления пополам.

Это первый этап поиска минимума. На нем после пяти вычислений функции (два – на краях и три -- внутри интервала [a,b]) интервал неопределенности сужается вдвое, то есть на этом этапе α = 0.5. Новый интервал неопределенности [x4,x5] снова разделим пополам, а затем каждую половину снова поплам.

Теперь значение функции по краям и в его середине уже извесны. Поэтому для завершения поиска на этом этапе требуется вычислить только два значения функции, после чего интервал неопределенности снова усеньшается вдвое. Это преимущество рассмотренного метода сохранится в дальнейшем.

Схема алгоритма представлена ниже:(рисунок 2.3):

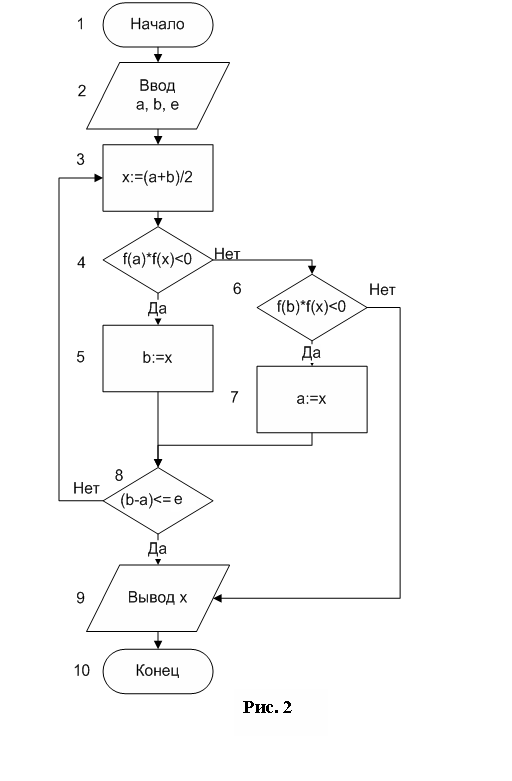


Рисунок. 2.3 – Схема метода деления пополам

*Алгоритм «Метод Гаусса»*

Метод Гаусса— прямой метод решения задач многомерной оптимизации.

Пусть необходимо найти [минимум](https://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BD%D0%B8%D0%BC%D1%83%D0%BC) действительнозначной [функции](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) {\displaystyle f({\vec {x}})\to \min \_{{\vec {x}}\in \mathbb {R} ^{n}}}

Суть метода заключается в том, чтобы на каждой итерации по очереди минимизировать функцию вдоль каждой из координат.

Таким образом метод как бы «поднимется» по координатам, используя на шагах одной итерации для вычисления следующей координаты точки приближения все предыдущие значения координат, вычисленные на той же итерации, в этом и состоит схожесть с одноимённым методом решения СЛАУ.

При завершении итерации, точка, полученная на последнем шаге этой итерации, берётся в качестве следующего приближения.

Процедура продолжается до тех пор, пока не будет достигнута заданная точность (рисунок 2.4).

Улучшением данного метода является метод покоординатного спуска Гаусса - Зейделя.

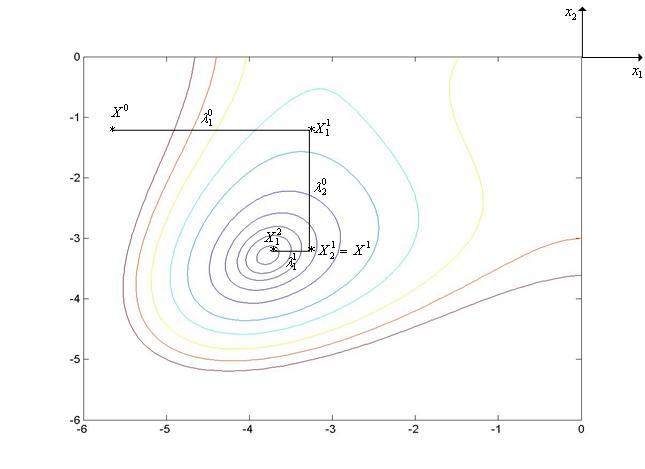


Рисунок. 2.4 – Метод Гаусса

Алгоритм нахождения (рисунок 2.5):

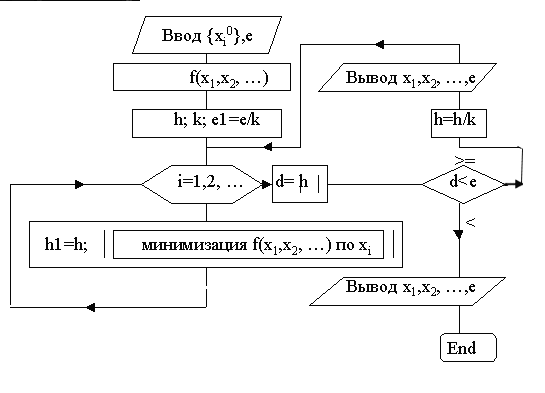


Рисунок 2.5 – Схема метод Гаусса

*Алгоритм «Метод главного критерия»*

Алгоритм используется в многокритериальной оптимизации. Идея метода состоит в максимизации (минимизации) главного критерия, при условии, что значение других критериев не превышают заданных заранее пороговых величин. Этот метод предполагает сведение задачи многокритериальной оптимизации к однокритериальной оптимизации.

*Алгоритм «Метод minmax»*

Алгоритм используется в многокритериальной оптимизации. Идея метода следующая: пусть имеется некоторая матрица со значениями функций. Данная матрица дополняется еще одним столбцом, содержащим наименьший результат в каждой строке, затем среди этих вариантов выбирают максимальный и отображают.

**3 Программная реализация**

**3.1 Реализация нескольких окон**

Программа состоит из трёх окон – главного (Рис. 3.1) и двух других окон, которые открываются по нажатию соответсвующих кнопок на главном окне (Рисунок. 3.2 и 3.3).



Рисунок. 3.1 – Главное окно программы

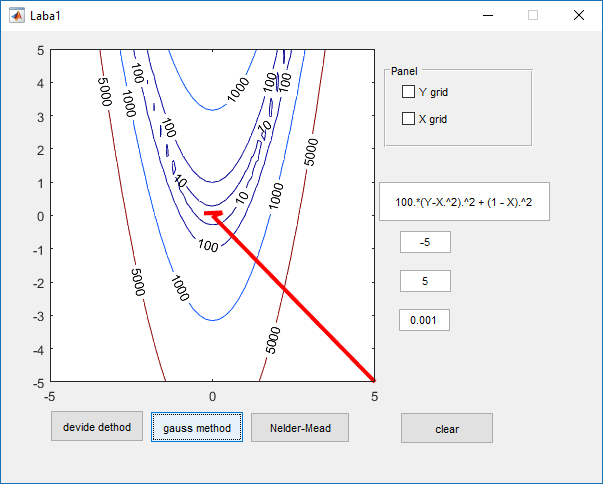


Рис. 3.2 – Окно для однокритериального поиска

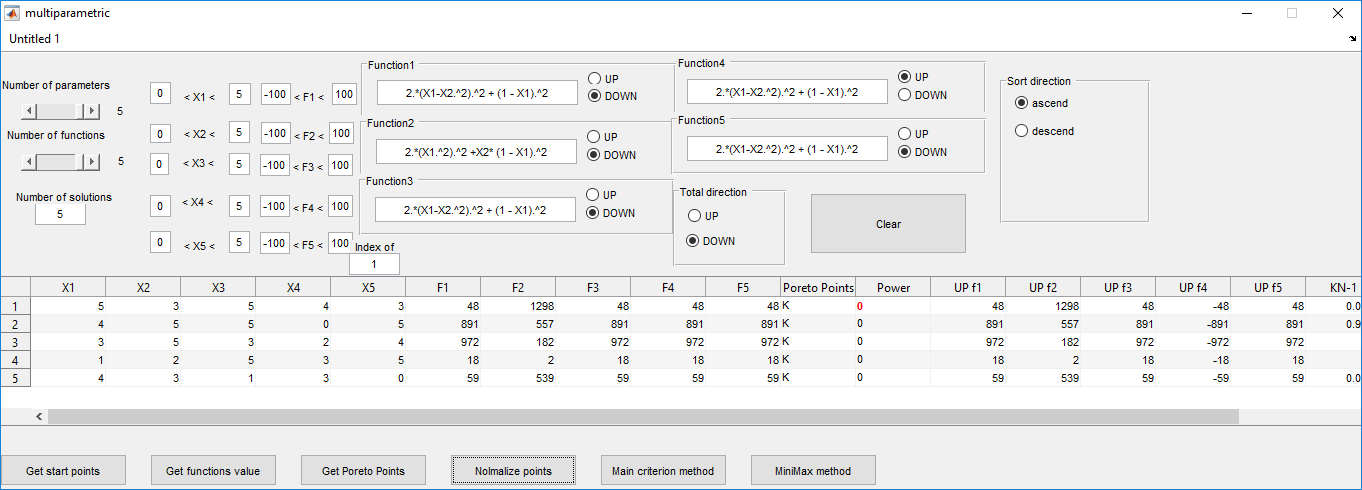


Рис. 3.3 – Окно для многокритериального поиска

Запуск двух окон из главного реализован следующим образом:

function pushbuttonMonoCroterial\_Callback(hObject, eventdata, handles)

fig = openfig('Laba1.fig');

handl = guihandles(fig);

guidata(fig, handl);

function pushbuttonManyCriterial\_Callback(hObject, eventdata, handles)

fig = openfig('multiparametric.fig');

handl = guihandles(fig);

guidata(fig, handl);

**3.2 Реализация алгоритмов**

Алгоритмы однокритериального поиска:

*Алгоритм «Метод Гаусса»*

while abs(zl-min\_con)>eps\*eps\*eps

count\_steps = count\_steps + 1;

A = Ax;

B = Bx;

A1 = A1y;

B1 = B1y;

y\_0 = min\_y;

x\_0 = min\_x;

zl = min\_z;

while (B - A) > 3\*eps

m = A + (B - A)/4;

n = B - (B - A)/4;

x\_sr = (B + A)/2;

v1 = [m, y\_0];

v2 = [x\_sr, y\_0];

v3 = [n, y\_0];

if fun2(v1) < fun2(v2)

B = x\_sr;

end

if fun2(v1) > fun2(v2)

if fun2(v3) < fun2(v2)

A = x\_sr;

end

if fun2(v3) >= fun2(v2)

A = m;

B = n;

end

end

end

while A < B

v4 = [A, y\_0];

Z = fun2(v4);

if min\_z > Z

min\_z=Z;

min\_x=A;

end

A = A + eps\*eps;

end

X1 = [x\_0,min\_x];

Yl = [y\_0, y\_0];

Zl = [zl, min\_z];

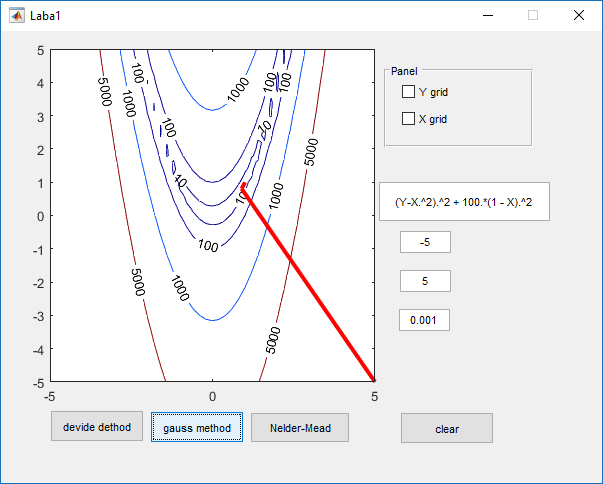


Рисунок 3.4 – Метод Гаусса

*Алгоритм «Деление пополам»*

function [x, y] = devide(func, ax1, bx1,ay,by, del,v)

x\_avg = (ax + bx) / 2 ;

while abs((ax - bx)) >= del

avg\_left = (ax + x\_avg) / 2;

avg\_right = (x\_avg + bx) / 2;

if(func(ax,v(2)) > func(x\_avg,v(2)) && func(bx,v(2)) > func(x\_avg,v(2)))

ax = avg\_left;

bx = avg\_right;

else if(func(ax,v(2)) > func(x\_avg,v(2)))

ax = x\_avg;

else

bx = x\_avg;

end

end

x\_avg = (ax + bx) / 2;

if (func(x\_avg,v(2)) < z\_min)

z\_min = func(x\_avg,v(2));

x\_min = x\_avg;

y\_min = v(2);

end;

end;

y\_avg = (ay + by) / 2;

while abs((ay - by)) > del

avg\_left = (ay + y\_avg) / 2;

avg\_right = (y\_avg + by) / 2;

if(func(x\_min,ay) > func(x\_min,y\_avg) && func(x\_min,by) > func(x\_min,y\_avg))

ay = avg\_left;

by = avg\_right;

else if(func(x\_min,ay) > func(x\_min,y\_avg))

ay = y\_avg;

else

by = y\_avg;

end

end

y\_avg = (ay + by) / 2;

if(func(x\_min,y\_avg) < z\_min)

z\_min = func(x\_min,y\_avg);

y\_min = y\_avg;

end

end

x = x\_min;

y = y\_min;

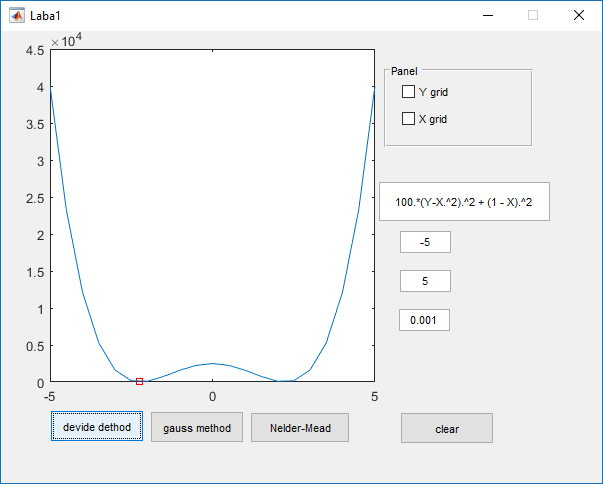


Рисунок 3.5 – Метод деления пополам

Алгоритмы многокритериального поиска:

*Алгоритм «Метод Нелдер-Мида»*

function pushbuttonRandomSearch\_Callback(hObject, eventdata, handles)

for i=1:spaceCount

numberOfParam = strcat('X', num2str(i));

string = strcat(string, numberOfParam, ',');

end

string = string(1:end-1);

string = strcat(string, ') ', str);

func\_handle = str2func(string);

f = func\_handle(X,Y);

[C,h]= contour(X,Y,f,[0,0.1,1,10,100,1000]);

clabel(C,h);

Xk = [startPointCoordinates{1}, startPointCoordinates{2}];

while step > minStep

Xk0 = Xk;

for j=1:stepCount

ang = -1 + (10)\*rand(1);

Yj = [cos(ang)\*step + Xk(1), sin(ang)\*step + Xk(2)];

if func\_handle(Yj(1), Yj(2)) < func\_handle(Xk(1), Xk(2))

Zj = [Xk(1) + 0.5\*(Yj(1) - Xk(1)), Xk(2) + 0.5\*(Yj(2) - Xk(2))];

if func\_handle(Zj(1), Zj(2)) < func\_handle(Xk(1), Xk(2))

Xk = Zj;

j = stepCount;

end

else

if (j < stepCount)

j = j + 1;

else

step = step / 2;

j = 1;

continue;

end

end

end

line([Xk0(1), Xk(1)], [Xk0(2), Xk(2)],'Color','g','LineWidth', 3);

end

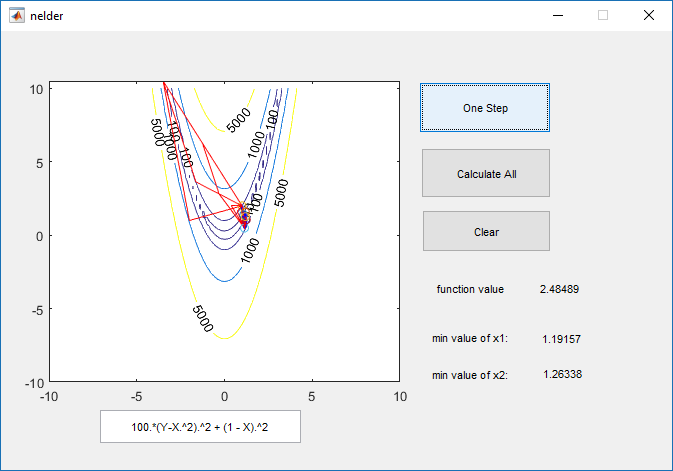
**

Рисунок 2.6 – Метод Нельдер-Мида

*Алгоритм «Метод главного критерия»*

if strcmp(totalDirection, 'UP')

value = intmin('int32');

else

value = intmax('int32');

end

for i=1:numberOfSolutions

arrayForCompare(i) = mainColumn(i);

for j=1:numberOfFunctions - 1

if (matrixWithoutColumn(i, j) <= str2num(cell2mat(arrayOfMin(j)))) || (matrixWithoutColumn(i, j) >= str2num(cell2mat(arrayOfMax(j))))

arrayForCompare(i) = value;

j = numberOfFunctions;

end

end

end

if strcmp(totalDirection, 'UP')

indexInColumn = getIndexOfMax(arrayForCompare);

else

indexInColumn = getIndexOfMin(arrayForCompare);

end

*Алгоритм «Метод minmax»*

for i=1:numberOfSolutions

if ~strcmp(tableData(i, numberOfParameters + numberOfFunctions + 1), 'C')

currPoint = cell2mat(tableData(i, numberOfParameters + numberOfFunctions \* 2 + 3 : numberOfParameters + numberOfFunctions\*3 + 2));

tableData(i, rowCount + 1) = {getDistanceBetweenPoint(currPoint, numberOfSolutions, numberOfFunctions)};

end

end

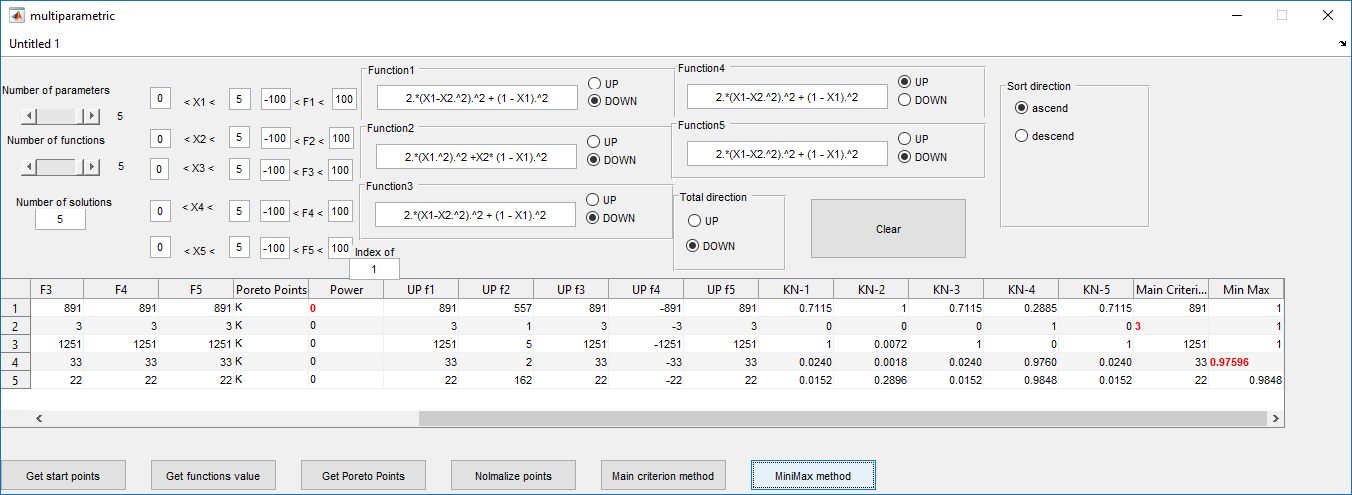
****

Рисунок 2.7 -- Метод главного критерия и minmax.

1. **Руководство пользователя**

Для открытия программы запускаем файл *start.exe*. Откроется окно с двумя кнопками, для выбора однокритериальной или многокритериальной оптимизации соответственно (Рис 4.1).

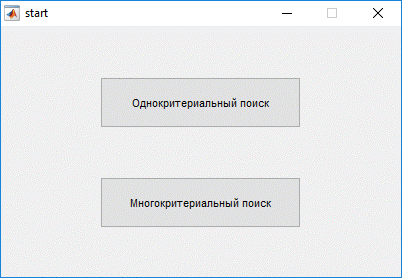


Рис. 4.1 – Главное окно программы

При нажатии на первую кнопку, откроется окно с методами однокритериальной оптимизации (Рис. 4.2)

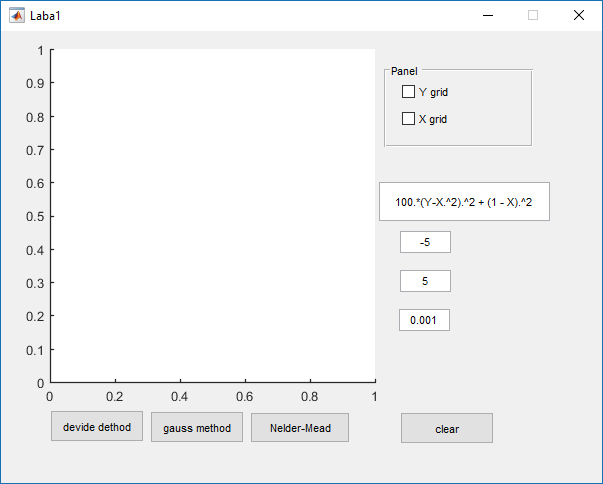


Рис. 4.2 – Окно для однокритериального поиска

Для вызова метода дихотомии необходимо нажать на соответствующую кнопку «деление пополам». При необходимости пользователь может ввести свою функцию в соответствующее текстовое поле. Результат работы метода представлен на рис. 4.3.

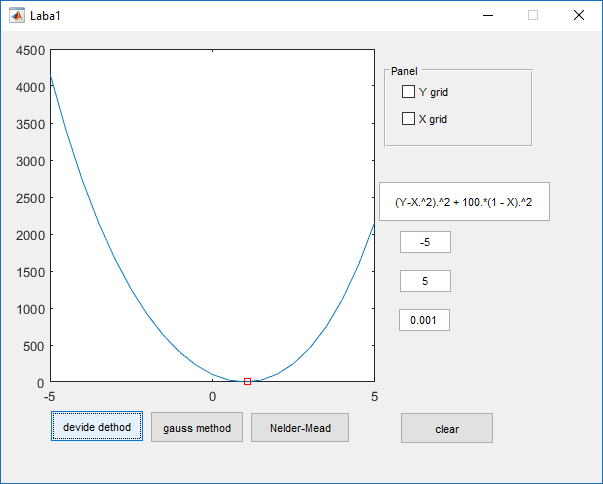


Рис. 4.3 – Результат работы деления пополам

Для вызова метода Гаусса необходимо нажать на соответствующую кнопку «Гаусс». Дополнительно пользователь может ввести свои начальные данные: задать начальную точку, значение точности и ввести функцию. Результат работы представлен на рис. 4.3

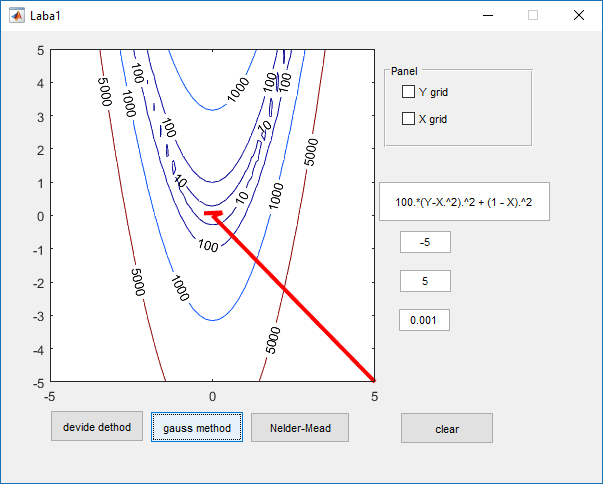
****

Рис. 4.3 – Результат работы метода Гаусса

Для вызова метода Нелдер-Мида, пользователю предварительно необходимо задать начальные параметры. Сначала пользователь вводит количество критериев, затем нажимает кнопку «CalculateALL». Далее, выводятся значения функции и ее параметров на форме. Результат работы представлен на рис. 4.5.

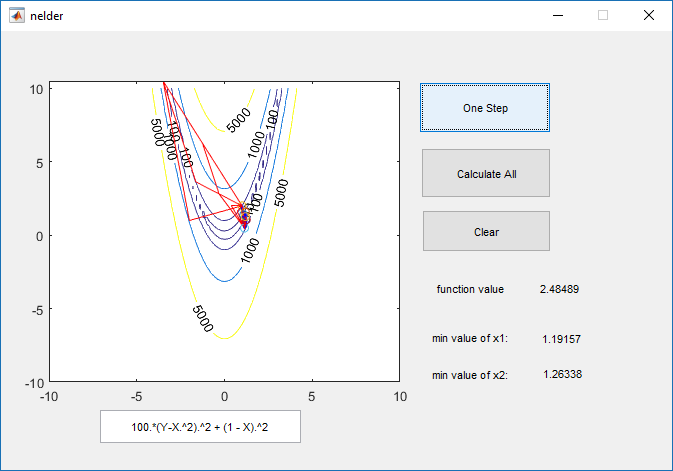


Рис. 4.4 – Результат работы метода Нелдер-Мида

Также в этом окне есть кнопка «Очистить», при нажатии на которую окно вывода будет полностью очищено (Рис. 4.5).

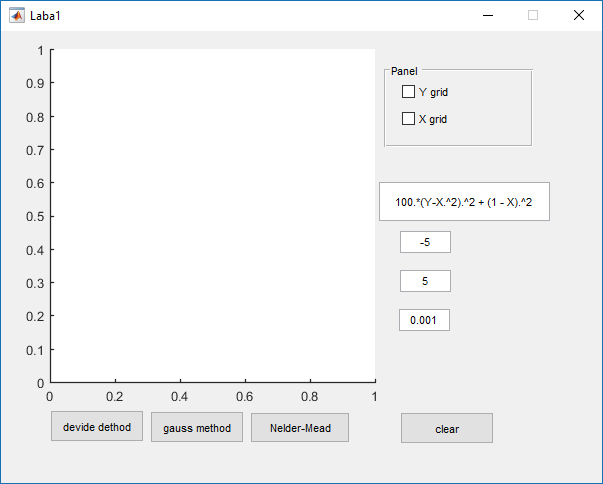
****

Рис. 4.6 – Результат работы кнопки «Очистить»

Если закрыть окно однокритериальной оптимизации и в стартовом окне нажать на кнопку «Многокритериальная оптимизация», откроется окно для многокритериальных расчетов. В этом окне пользователь сам может задать количество критериев, ограничения на них, количество функций, ввести в соответствующие поля сами функции, выставить их направление (на max или на min), а также выбрать главный критерий и задать ограничения на остальные. Пример окна для многокритериальных расчетов представлен на рис. 4.7.

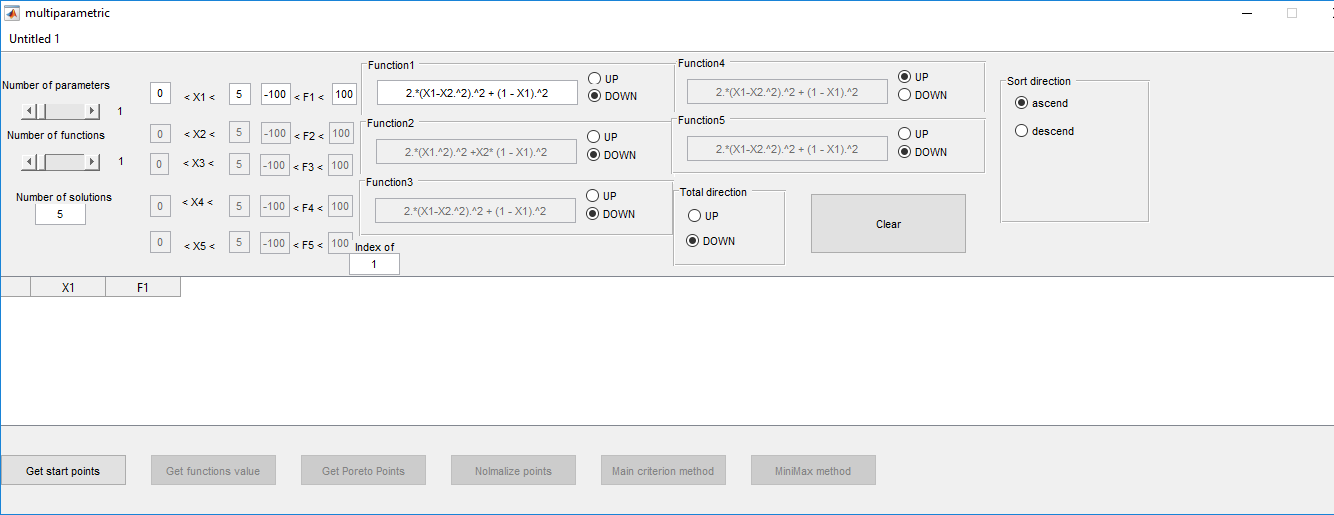


Рис. 4.7 – Окно для многокритериального поиска

После ввода всех нужных параметров пользователь нажимает на кнопку «Get start point» и в таблицу генерируются параметры, соответствующие заданным условиям (Рис. 4.8).

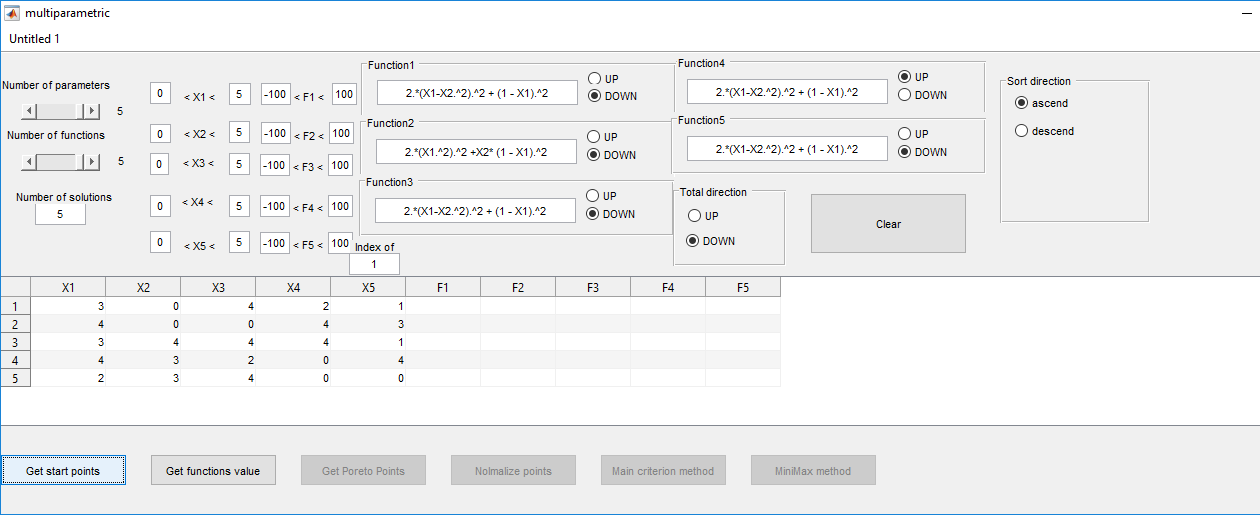


Рис. 4.8 – Результат генерации параметров

Далее необходимо рассчитать значения функций для сгенерированных параметров. Для этого необходимо нажать на кнопку «Get functions value» (Рис. 4.9).

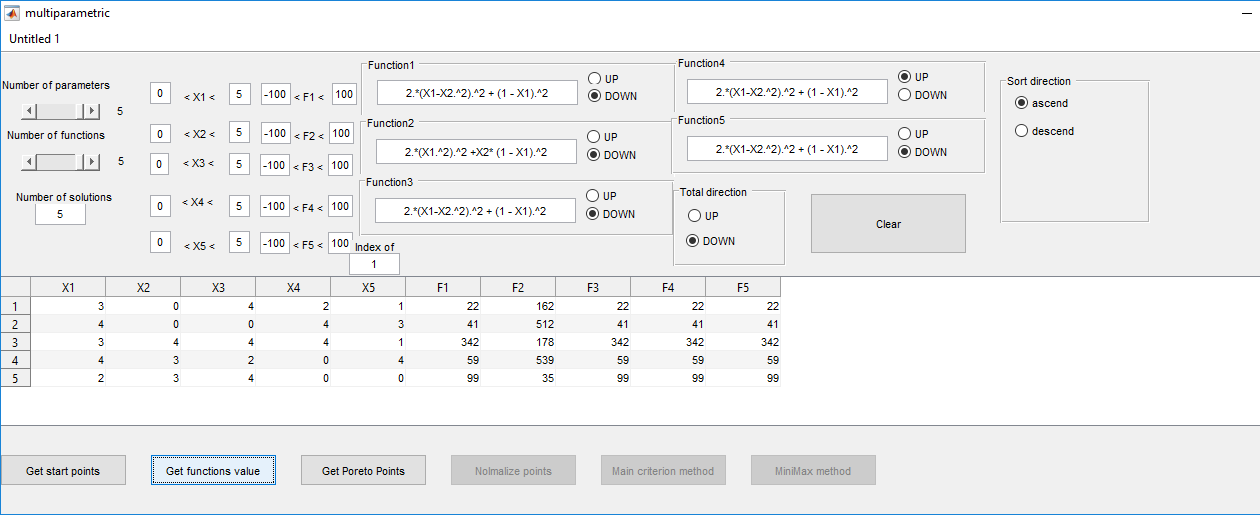


Рис. 4.9 – Результат вычисления значений функций

При нажатии на следующую кнопку «Get Poreto points», используя уже полученные на предыдущих этапах данные, находятся и выводятся в таблицу области компромиссов и согласия. Также в другой столбец выводятся значения мощности. Наилучшее из значений выделяется красным цветом (Рис. 4.24).

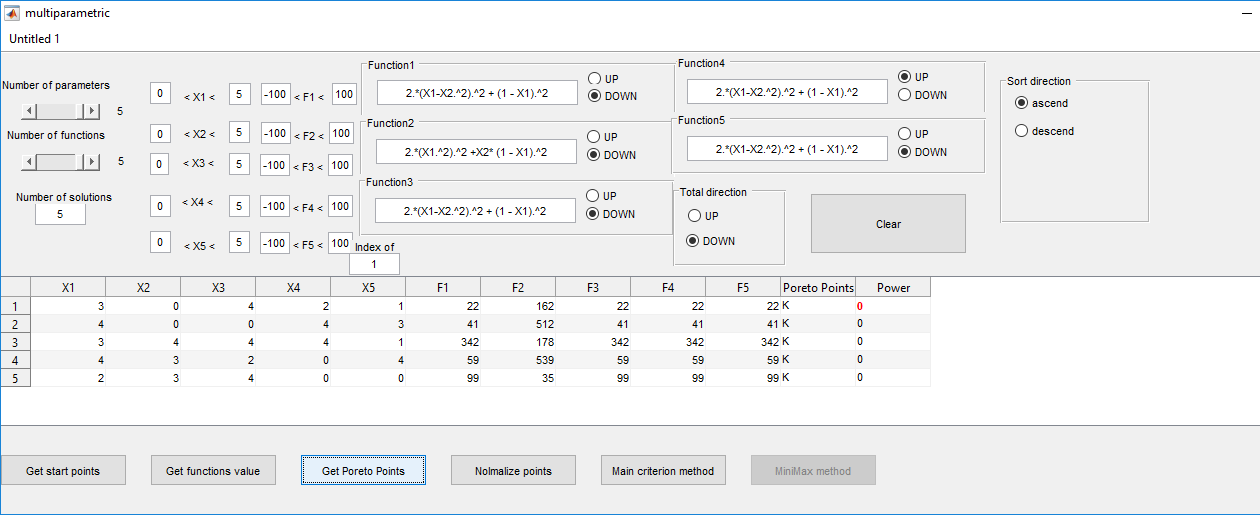


Рис. 4.10 – Результат работы метода расстояний до идеальной точки

Далее нажимаем кнопку «Perfect point method» и программа производит расчет нормализованных расстояний, используя метод расстояний до идеальной точки и выводит в таблицу свертку и значения после нормализации (Рис. 4.11).

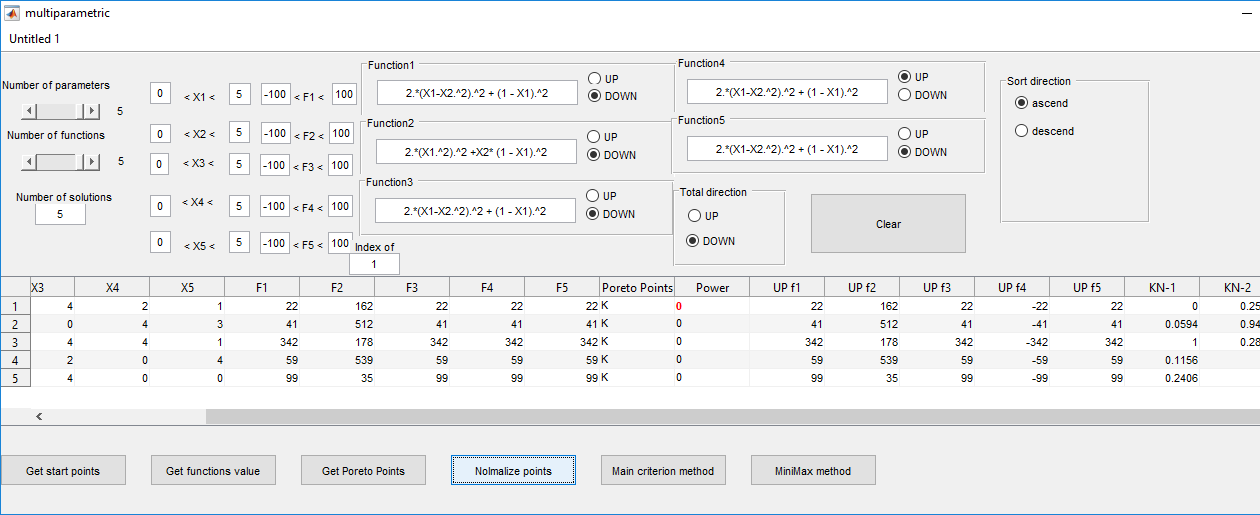


Рис. 4.11 – Результат работы нормализации

При нажатии на кнопку «Main criterion method» программа производит расчет, использую введенные ранее пользователем данные и выводит столбец главного критерия и подсвечивает красным цветом искомое значение (Рис. 4.12).

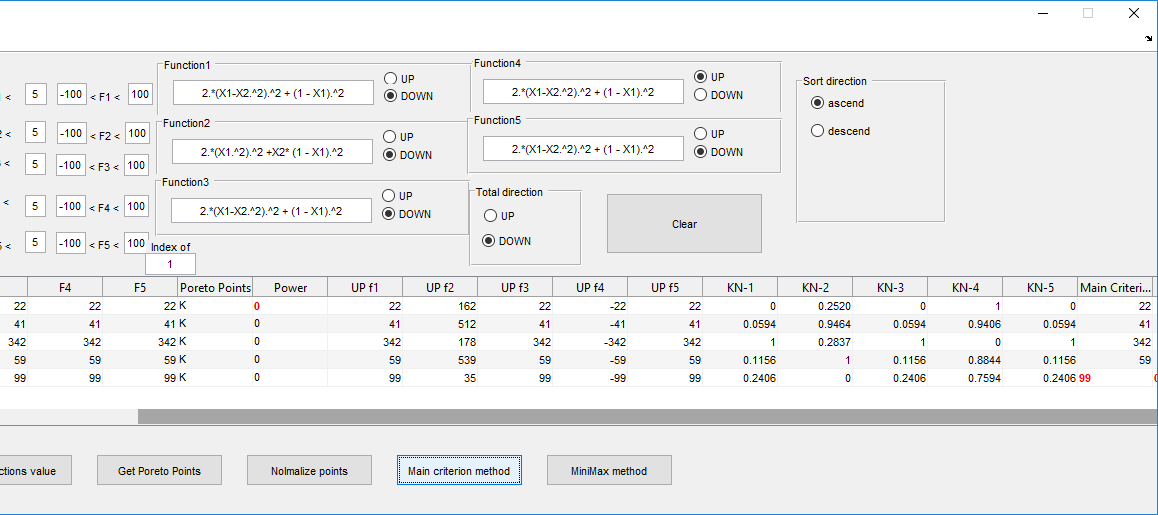


Рис. 4.12 – Результат работы метода главного критерия

После нажатия кнопки «MinMax method» программа выводит результат работы этого метода в дополнительный столбец и подсвечивает красным цветом минимальное значение (Рис. 4.13).

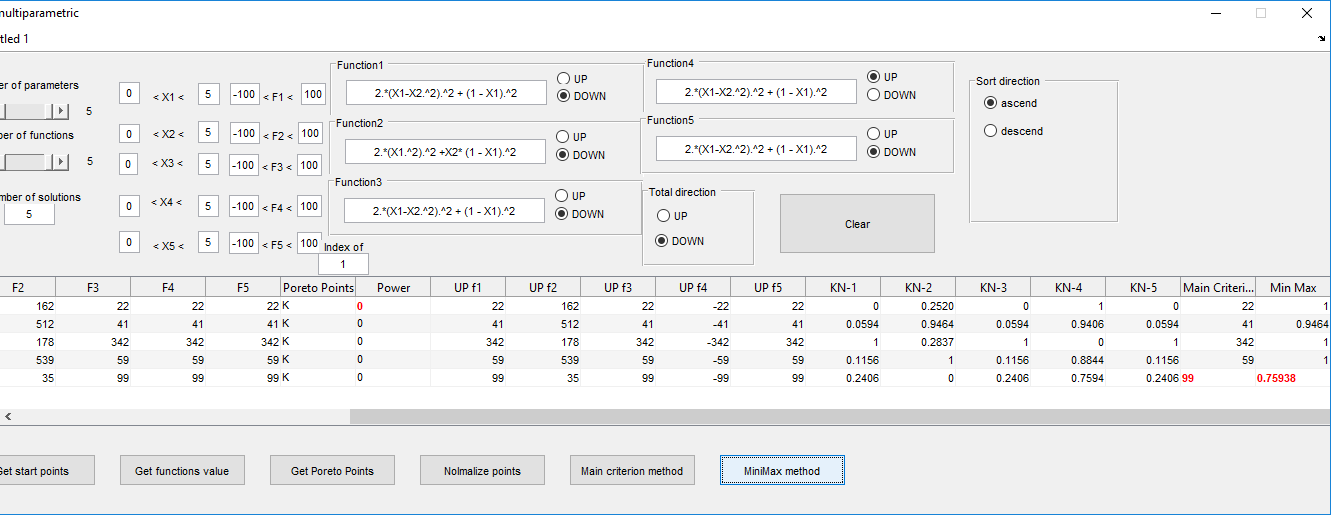


Рис. 4.13 – Результат работы метода minmax

Также в этом окне есть кнопка «Очистить» при нажатии на которую все данные из таблиц удаляются (Рис. 4.14).

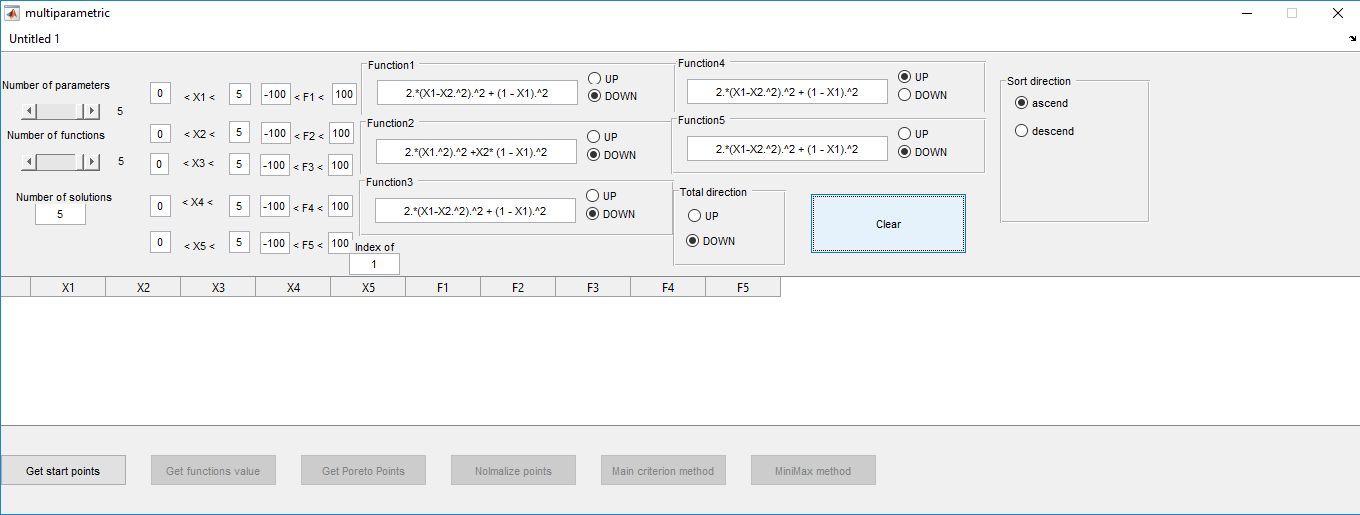


Рис. 4.14 – Результат работы кнопки «Clear»

**5 Тестирование приложения**

**5.1 Тестирование основных методов**

Для начала протестируем правильность работы двухпараметрического метода Гаусса. Кроме ввода функции для этого метода также необходимо вводить ограничения на параметры и точность. Результаты тестирования данного метода представлены на рисунках 5.1 – 5.3.

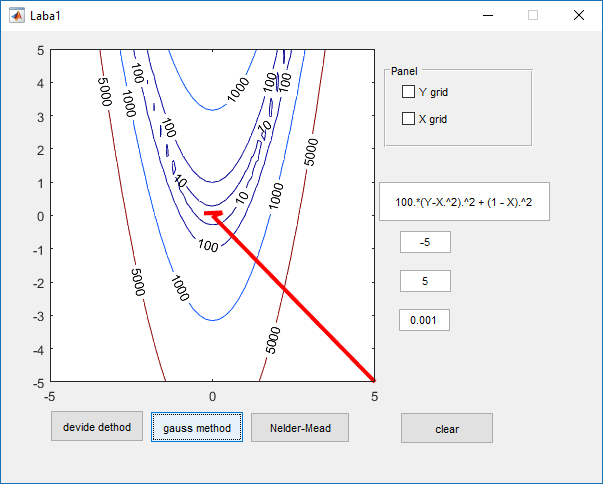


Рис. 5.1 – Тест функции 1 методом Гаусса

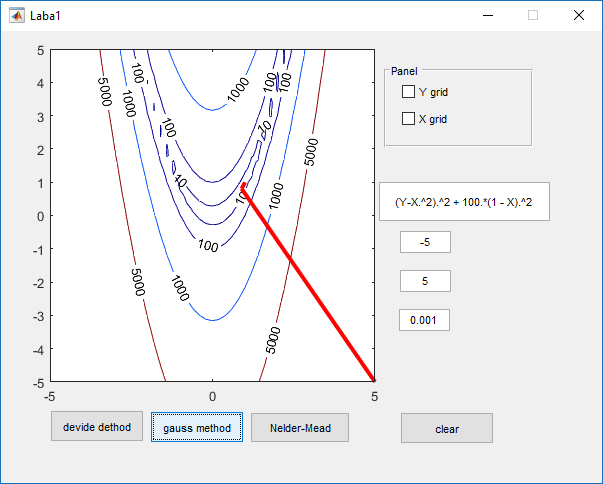


Рис.5.2 – Тест функции 2 методом Гаусса

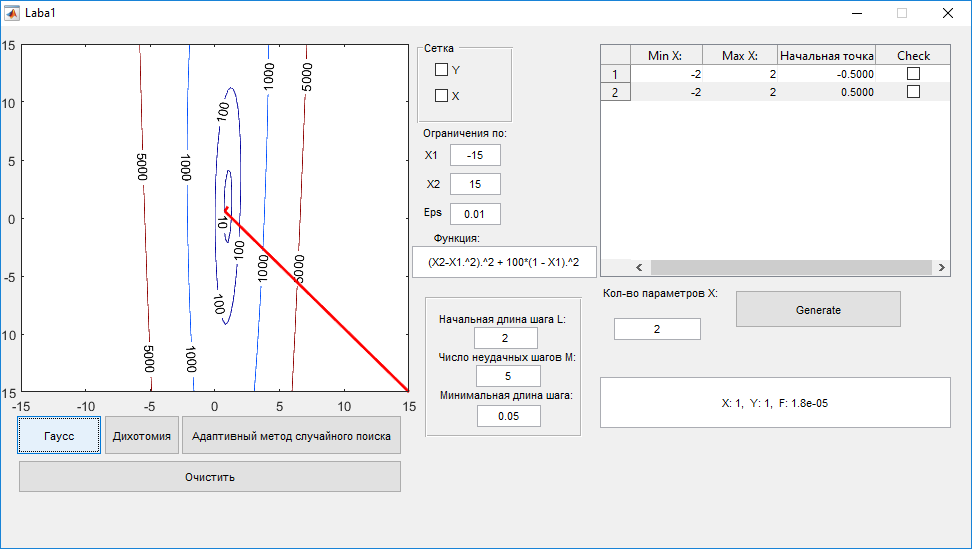


Рис. 5.3 – Тест функции 3 методом Гаусса

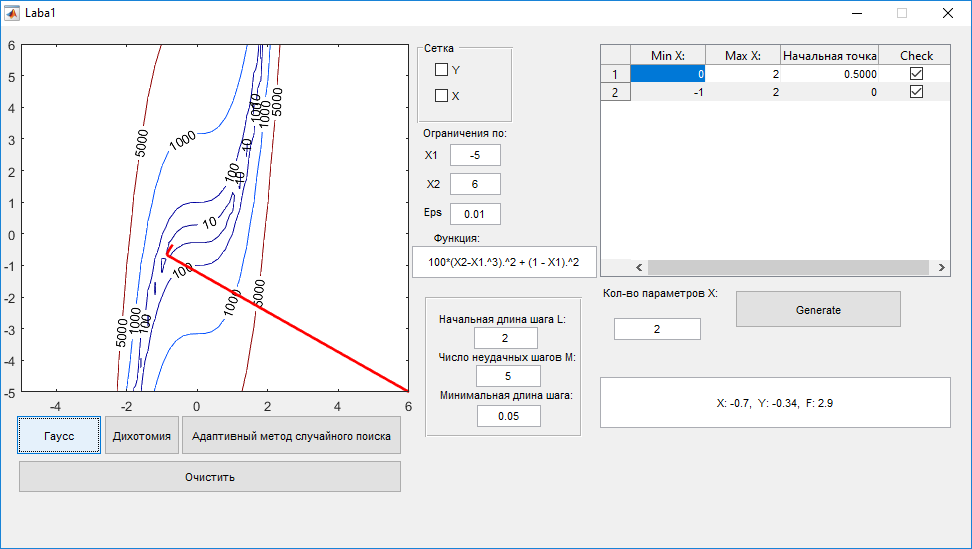


Рис. 5.4 – Тест функции 4 методом Гаусса

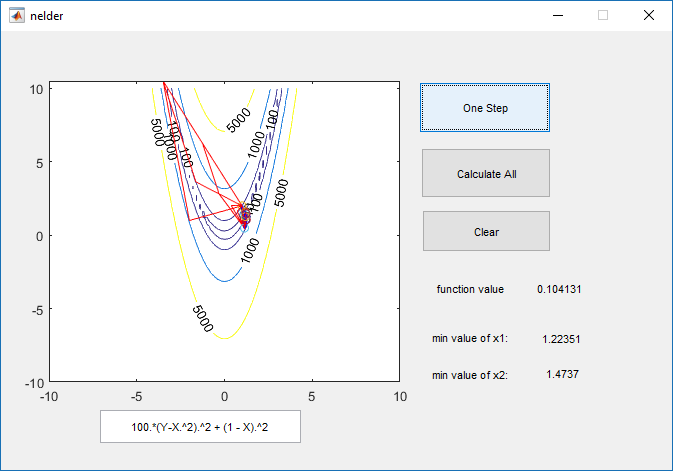


Рис 5.4 – тестирование метода Нелдер-Мида

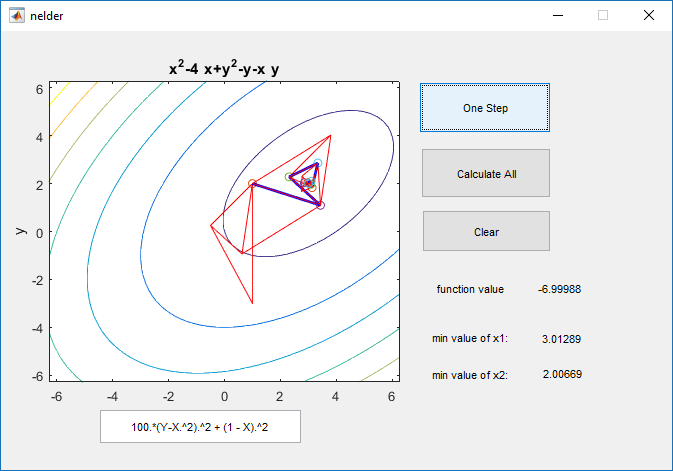


Рисунок 5.5 – тестирование метода Нелдер-Мида

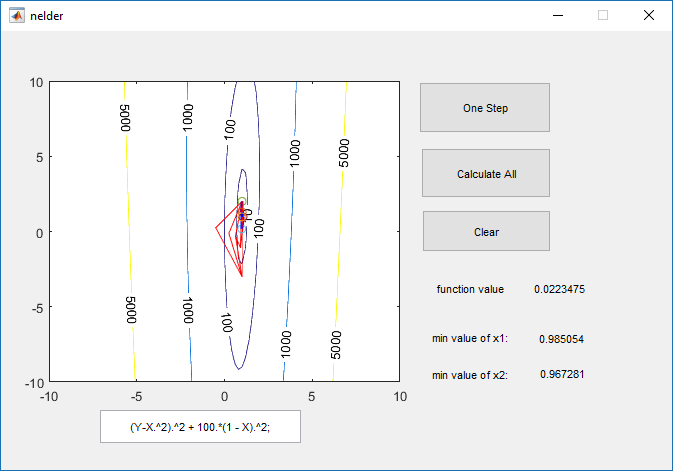


Рисунок 5.6 – тестирование метода Нелдер-Мида

В результате тестирования этого метода ошибок выявлено не было, так же, как и в случае с методом деления пополам. Следовательно, данный алгоритм реализован правильно.

Следующим этапом будет тестирование второй части программы, предназначенной для многокритериальных расчетов. Тестирование было проведено на примере функции (x 1 – x 22 )2 + (1- x1) 2

Были протестированы различные варианты работы данной части подсистемы, такие как ввод различного количества параметров для решений (максимально возможного и нет), ввод нескольких вариантов количества функций, различные варианты направлений и др. В ходе тестирования были выявлены некоторые особенности работы программы. Одним из них является вывод сообщения об невозможности проведения нормализации из-за недостатка параметров, что безусловно является преимуществом данного ПО, так как облегчает его использование и делает его более понятным. Пример данного сообщения приведен на рис. 5.5.

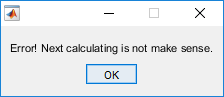


Рис. 5.5– Сообщение об ошибке

Однако в программе были найдены и дефекты. Одним из таковых является неправильная работа метода minmax. Ошибка заключается в том, что если пользователь ввел не максимальное кол-во параметров, то метод не сработает и не выведет никакое сообщение об ошибке. Пример неправильности работы этого метода представлен на рис. 5.6

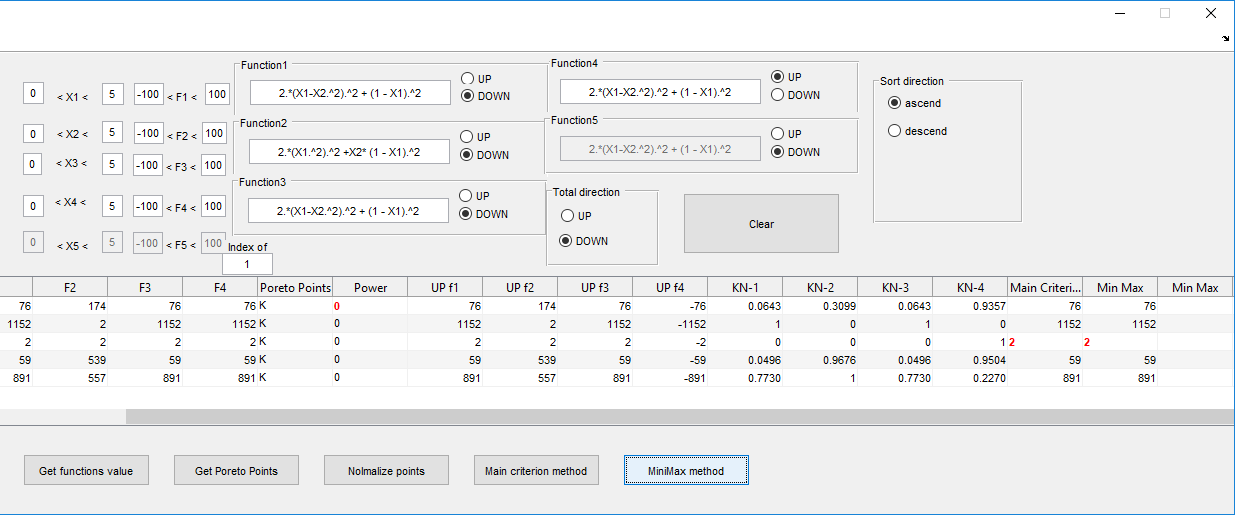


Рис. 5.6 – Пример дефекта работы метода minmax

На основе проведенных тестов можно сделать вывод, что часть подсистемы, предназначенной для многокритериальных расчетов, выполняет свои функции, но имеет некоторые ошибки в реализации и не может выполнить всю работу в полном объеме. В свою очередь часть подсистемы, которая выполняет функции однокритериальной и многокритериальной оптимизации, реализована на высоком уровне, в полной мере справляется со всеми возложенными на нее задачами и не имеет ошибок как в работе алгоритмов, так и в пользовательском интерфейсе.

**Заключение**

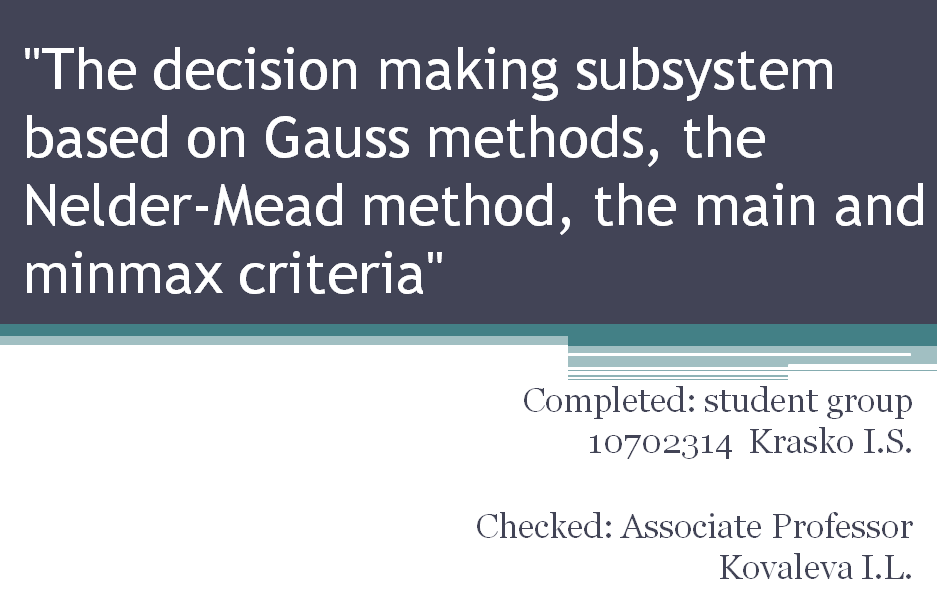
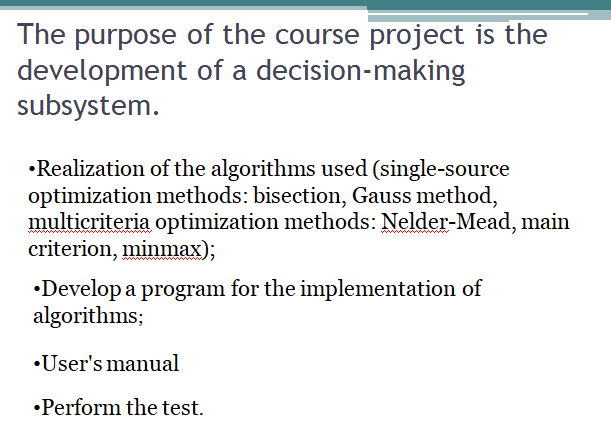
В ходе курсового проекта было разработано приложение для решений однокритериальных и многокритериальных задач.

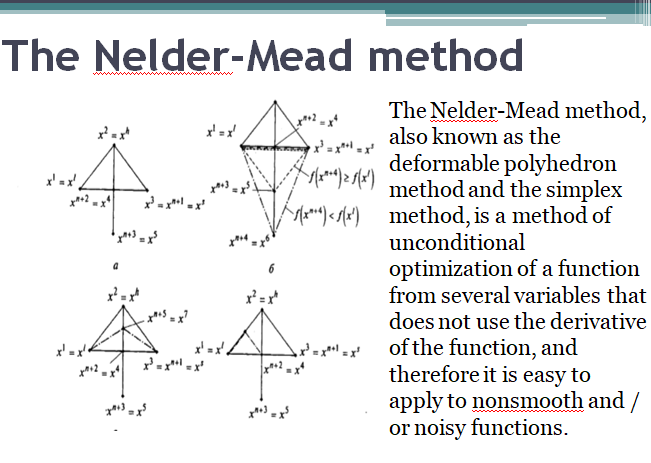
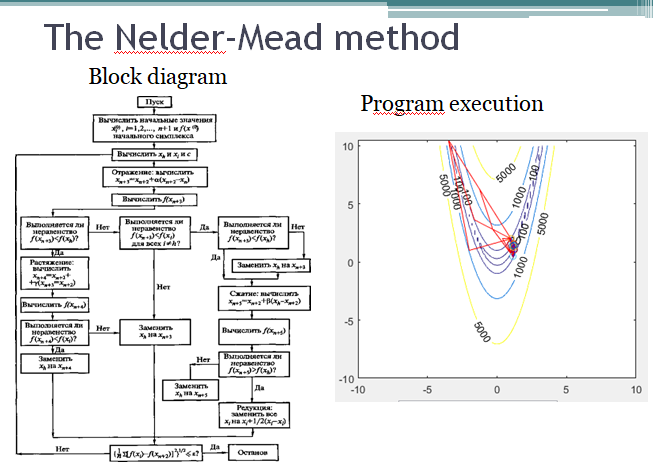
Разработали программную реализацию данных нам методов на MATLAB.

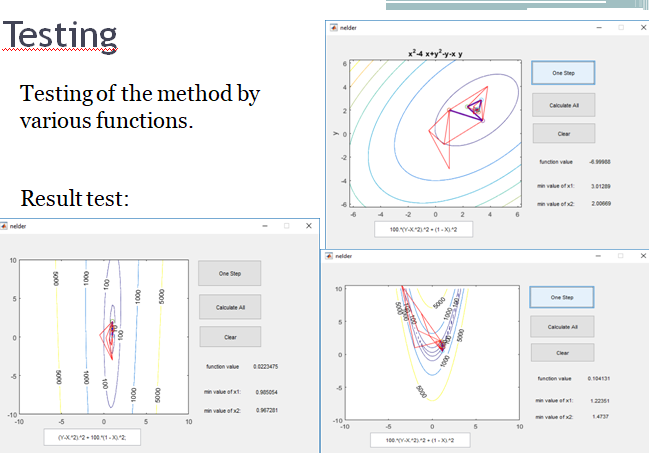
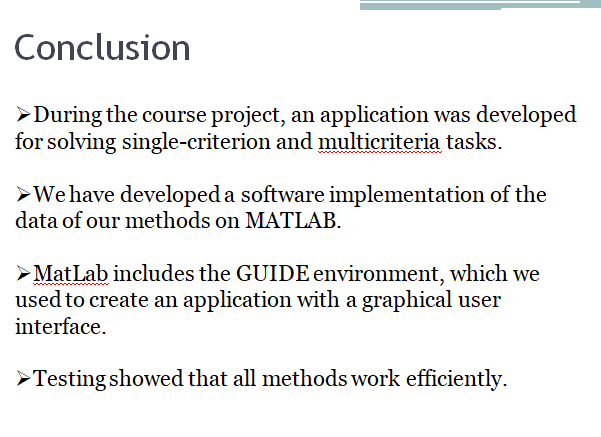
В состав MatLab входит среда GUIDE, ее мы использовали для создания приложения с графическим интерфейсом пользователя.

Тестирование показало, что все методы работают эффективно.

**Приложение**

# **ЛИТЕРАТУРА**

1. Инженерный пакет Matlab / [Электронный ресурс] / Официальная документация Matlab - Минск, 2017 - Режим доступа: [https://www.mathworks.com/help/matlab/](https://goo.gl/H1Xk1p)
2. ООП в Matlab Language - [Электронный ресурс] / Официальная документация - Минск 2017 - Режим доступа: https://www.mathworks.com/help/matlab/matlab\_oop/why-use-object-oriented-design.html
3. Конструктор класса в Matlab Language - [Электронный ресурс] / Официальная документация - Минск 2017 - Режим доступа: https://www.mathworks.com/help/matlab/matlab\_oop/class-constructor-methods.html
4. Атрибуты свойств в Matlab Language - [Электронный ресурс] / Официальная документация - Минск 2017 - Режим доступа: https://www.mathworks.com/help/matlab/matlab\_oop/property-attributes.html
5. Атрибуты методов в Matlab Language - [Электронный ресурс] / Официальная документация - Минск 2017 - Режим доступа: https://www.mathworks.com/help/matlab/matlab\_oop/method-attributes.html
6. Статические методов в Matlab Language - [Электронный ресурс] / Официальная документация - Минск 2017 - Режим доступа: https://www.mathworks.com/help/matlab/matlab\_oop/static-methods.html -

Ссылки на методы в Matlab Language - [Электронный ресурс] / Официальная документация - Минск 2017 - Режим доступа: https://www.mathworks.com/help/matlab/matlab\_prog/creating-a-function-handle.html