САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ПЕТРА ВЕЛИКОГО

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Институт компьютерных наук и кибербезопасности

высшая школа компьютерных технологий и информационных систем

**РАСЧЁТНОЕ ЗАДАНИЕ**

**«Нелинейное программирование. Условная оптимизация целевой функции»**

по дисциплине «Системный анализ и принятие решений»

Выполнил:

студент гр. 5130901/20102

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Вагнер А.А.

(подпись)

Преподаватель:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сиднев А.Г.

(подпись)

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

Санкт-Петербург

2024

Оглавление

[Исходные данные. Вариант 1. 3](#_Toc182084104)

[1. Метод Лагранжа 4](#_Toc182084105)

[2. Метод Била 8](#_Toc182084106)

[3. Метод проекции градиента 13](#_Toc182084107)

[4. Метод возможных направлений 18](#_Toc182084108)

[5. Метод штрафных функций 22](#_Toc182084109)

[6. Метод барьерных функций 25](#_Toc182084110)

[7. Метод Франка-Вульфа 28](#_Toc182084111)

Исходные данные. Вариант 1.

Содержанием задания является поиск условного максимума целевой функции следующими методами:

* Лагранжа (ограничение 5);
* Била (ограничения 1, 2, 3, 4);
* проекции градиента (ограничения 1, 2, 3, 4);
* возможных направлений (ограничения 6, 7);
* штрафных функций (ограничения 1, 2, 3, 4);
* барьерных функций (ограничения 1, 2, 3, 4);
* Метод Франка-Вульфа (ограничения 1, 2, 3, 4).

1. Метод Лагранжа

При

Напишем функцию Лагранжа:

Условие стационарности L(X,V) в точке (Xi, Vi):

Решим систему уравнений:

Матрица Гесса:

Она отрицательно определена, поэтому данная точка является точкой максимума L(X,V), соответственно X\* является решением задачи условной оптимизации.

Изображение выглядит как текст, снимок экрана, диаграмма, круг

Автоматически созданное описание

Рис. 1 – Результат работы программы

Листинг 1 – Код программы

function [] = Main()

clc;

close all;

clear all;

mas = [-6,-9,4,20,60,-1,1,1,1,1,0,4,6,2,1,9,36];

start\_x1 =0;

start\_x2 = 0;

stop\_x1 = 7;

stop\_x2 = 7;

fprintf('Решение задачи методом Лагранжа:\nУсловия стационарности в точке (X\*,V\*):\n\n');

[ Dermas ] = PartialDerivative( mas, 1 );

fprintf('\nПолучили СЛАУ. Решаем и получаем единственное решение:\n\n');

[ x1,x2,v ] = GetAnswer ( Dermas );

fprintf('\nЗначение функции:\n');

[ f ] = F(x1, x2, mas);

[ H ] =PlotGraph( mas,x1,x2,start\_x1,start\_x2,stop\_x1,stop\_x2);

hold on;

plot([start\_x1 stop\_x1],[x2 x2],':');

plot([x1 x1],[start\_x2 stop\_x2],':');

hold off;

legend('Линии равного уровня','Точка максимума');

end

function [ Dermas ] = PartialDerivative( mas, indicator )

Dermas(1,1)=mas(1)\*2;

Dermas(1,2)=mas(3);

Dermas(1,3)=mas(4);

Dermas(2,1)=mas(3);

Dermas(2,2)=mas(2)\*2;

Dermas(2,3)=mas(5);

if (indicator ==0)

fprintf ('dL/dx1 = %dx1 + ( %dx2 ) + ( %d )= 0\n',Dermas(1,1),Dermas(1,2),Dermas(1,3));

fprintf ('dL/dx2 = %dx1 + ( %dx2 ) + ( %d )= 0\n',Dermas(2,1),Dermas(2,2),Dermas(2,3));

end;

if (indicator ==1)

Dermas(3,1)=mas(10);

Dermas(3,2)=mas(11);

Dermas(3,3)=-mas(14);

Dermas(3,4)=0;

Dermas(1,4)=mas(10);

Dermas(2,4)=mas(11);

fprintf ('dL/dx1 = %dx1 + ( %dx2 ) + ( %d ) + ( %dv ) = 0\n',Dermas(1,1),Dermas(1,2),Dermas(1,3),Dermas(1,4));

fprintf ('dL/dx2 = %dx1 + ( %dx2 ) + ( %d ) + ( %dv ) = 0\n',Dermas(2,1),Dermas(2,2),Dermas(2,3),Dermas(2,4));

fprintf ('dL/dv = %dx1 + ( %dx2 ) + ( %d ) = 0\n',Dermas(3,1),Dermas(3,2),Dermas(3,3));

end;

end

function [ x1,x2,v ] = GetAnswer ( Dermas )

A=[Dermas(1,1), Dermas(1,2),Dermas(1,4);

Dermas(2,1), Dermas(2,2),Dermas(2,4);

Dermas(3,1), Dermas(3,2),Dermas(3,4);];

b = [-Dermas(1,3); -Dermas(2,3); -Dermas(3,3)];

x = inv(A)\*b;

x1 = x(1,1);

x2 = x(2,1);

v = x(3,1);

fprintf('(X\*,V\*) = (x1\*,x2\*,v\*) = (%-.4f,%-.4f,%-.4f)\n',x1,x2,v);

end

function [handler] = PlotGraph( mas,x1,x2,start\_x1,start\_x2,stop\_x1,stop\_x2)

x\_1=start\_x1:.1:stop\_x1;

x\_2=start\_x2:.1:stop\_x2;

[x\_1,x\_2]=meshgrid(x\_1,x\_2);

w=(mas(1)\*x\_1.^2 +mas(2)\*x\_2.^2 + mas(3)\*x\_1.\*x\_2 + mas(4)\*x\_1 + mas(5)\*x\_2);

handler = figure;

contour(x\_1,x\_2,w,50)

hold on;

plot (x1, x2, 'o-k','LineWidth', 2);

xlabel('x1');

ylabel('x2');

hold off

end

function [f] =F(x1, x2, mas)

f = mas(1)\*x1^2+mas(2)\*x2^2+mas(3)\*x1\*x2+mas(4)\*x1+mas(5)\*x2;

fprintf('\nf(x)=%-.4f\n\n',f);

end

1. Метод Била

Приведём к канонической форме:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | 1 | -1 | 4 |
|  | -1 | -1 | 6 |
|  |  |  |  |

Данная точка является допустимой, но не оптимальной.

Найдем соотношение между приращением свободной переменной и изменениями базисных переменных и и производной:

при при

при

Так как производная обращается в ноль раньше базисных переменных, введём в задачу новую свободную переменную . Переменную х2 включим в базис.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | 2 | -9 | 30 |
|  | 1 | -1 | 4 |
|  | -1 | -1 | 6 |

Пересчитаем таблицу

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | -2 | 1 | -30 |
|  | -7 | -1 | -6 |
|  | 11 | -1 | -24 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Выразим целевую функцию через новые свободные переменные:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Выберем ведущий столбец как максимальную производную в точке а ведущую строку как минимальное частное деления на отрицательные элементы столбца.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  | 100 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Выразим целевую функцию через новые свободные переменные:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Решение

Изображение выглядит как текст, снимок экрана, диаграмма, График

Автоматически созданное описание

Рис. 2 Траектория решения методом Била

Листинг 2 – Код программы

function [] = Main()

clc;

close all;

clear all;

mas = [-6,-9,4,20,60,-1,1,1,1,1,0,4,6,2,1,9,36];

start\_x1 =0;

start\_x2 = 0;

stop\_x1 = 8;

stop\_x2 = 8;

x1\_bil=[0, 0, 2.18181818];

x2\_bil=[0, 3.333333, 3.818181];

[ H2 ] = PlotBil( mas,x1\_bil,x2\_bil,start\_x1,start\_x2,stop\_x1,stop\_x2 );

end

function [ handler ] = PlotBil( mas,x1\_bil,x2\_bil,start\_x1,start\_x2,stop\_x1,stop\_x2 )

x=start\_x1:.1:stop\_x1;

y=start\_x2:.1:stop\_x2;

[ H ] =PlotGraph( mas,x1\_bil,x2\_bil,start\_x1,start\_x2,stop\_x1,stop\_x2);

handler =H;

hold on;

plot (x,x\*0,':b','LineWidth', 1);

plot (0\*y,y,':b','LineWidth', 1);

if (mas(7)==0)

plot (x, (mas(12)-mas(6)\*x),':b','LineWidth', 1);

else

plot (x, (mas(12)-mas(6)\*x)/mas(7),':b','LineWidth', 1);

end;

if (mas(9)==0)

plot (x, (mas(13)-mas(8)\*x),':b','LineWidth', 1);

else

plot (x, (mas(13)-mas(8)\*x)/mas(9),':b','LineWidth', 1);

end;

legend('Линии равного уровня','Ход решения','Ограничения');

hold off;

end

function [handler] = PlotGraph( mas,x1,x2,start\_x1,start\_x2,stop\_x1,stop\_x2)

x\_1=start\_x1:.1:stop\_x1;

x\_2=start\_x2:.1:stop\_x2;

[x\_1,x\_2]=meshgrid(x\_1,x\_2);

w=(mas(1)\*x\_1.^2 +mas(2)\*x\_2.^2 + mas(3)\*x\_1.\*x\_2 + mas(4)\*x\_1 + mas(5)\*x\_2 );

handler = figure;

contour(x\_1,x\_2,w,50)

hold on;

plot (x1, x2, 'o-k','LineWidth', 2);

xlabel('x1');

ylabel('x2');

hold off

end

1. Метод проекции градиента

**Шаг 1.**

;

направление допустимо

**Шаг 2.**

Найдём матрицу проекции P

**Шаг 3.**

Проверим условие останова:

=> Найденное решение потимально

Решение:

mas = [-6,-9,4,20,60,-1,1,1,1,1,0,4,6,2,1,9,36];

Изображение выглядит как текст, снимок экрана, диаграмма, линия

Автоматически созданное описание

Рис. 3 Траектория поиска решения методом проекции градиента

Листинг 3 – Код программы

function [] = Main()

clc;

close all;

clear all;

mas = [-6,-9,4,20,60,-1,1,1,1,1,0,4,6,2,1,9,36];

start\_x1 =0;

start\_x2 = 0;

stop\_x1 = 8;

stop\_x2 = 8;

x1\_grad=[0, 1.5, 2.4205];

x2\_grad=[0, 4.5, 3.5795];

[ H2 ] = PlotBil( mas,x1\_grad,x2\_grad,start\_x1,start\_x2,stop\_x1,stop\_x2 );

function [ handler ] = PlotBil( mas,x1\_bil,x2\_bil,start\_x1,start\_x2,stop\_x1,stop\_x2 )

x=start\_x1:.1:stop\_x1;

y=start\_x2:.1:stop\_x2;

[ H ] =PlotGraph( mas,x1\_bil,x2\_bil,start\_x1,start\_x2,stop\_x1,stop\_x2);

handler =H;

hold on;

plot (x,x\*0,':b','LineWidth', 1);

plot (0\*y,y,':b','LineWidth', 1);

if (mas(7)==0)

plot (x, (mas(12)-mas(6)\*x),':b','LineWidth', 1);

else

plot (x, (mas(12)-mas(6)\*x)/mas(7),':b','LineWidth', 1);

end;

if (mas(9)==0)

plot (x, (mas(13)-mas(8)\*x),':b','LineWidth', 1);

else

plot (x, (mas(13)-mas(8)\*x)/mas(9),':b','LineWidth', 1);

end;

legend('Линии равного уровня','Ход решения','Ограничения');

hold off;

end

end

function [handler] = PlotGraph( mas,x1,x2,start\_x1,start\_x2,stop\_x1,stop\_x2)

x\_1=start\_x1:.1:stop\_x1;

x\_2=start\_x2:.1:stop\_x2;

[x\_1,x\_2]=meshgrid(x\_1,x\_2);

w=(mas(1)\*x\_1.^2 +mas(2)\*x\_2.^2 + mas(3)\*x\_1.\*x\_2 + mas(4)\*x\_1 + mas(5)\*x\_2 );

handler = figure;

contour(x\_1,x\_2,w,50)

hold on;

plot (x1, x2, 'o-k','LineWidth', 2);

xlabel('x1');

ylabel('x2');

hold off

end

1. Метод возможных направлений

Найдём направление в соответствии с методом безусловной оптимизации, воспользуемся методом Ньютона.

Найдём наибольшее возможное u.

Последующие шаги проведём в MatLab

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Шаг |  |  |  |  |  |  |
| 0 | 2 | 3 | 1 | 1 | - | 69 |
| 1 | 1 | -0.1735 | 1.6173 | 1.926 | 2.7793 | 111.2857 |
| 2 | 1.1832 | 2.1086 | 1.8168 | 1.8914 | - | 111.5629 |
| 3 | 1 | -0.1393 | 1.8246 | 1.9053 | 1.1279 | 112.0686 |
| 4 | 1.0916 | 2.1064 | 1.9084 | 1.8936 | - | 112.1158 |
| 5 | 1 | -0.1255 | 1.9097 | 1.8960 | 0.4631 | 112.2018 |
| 6 | 1.0555 | 2.1084 | 1.9445 | 1.8916 | - | 112.2098 |
| 7 | 1 | -0.1198 | 1.9447 | 1.8920 | 0.1911 | 112.2245 |
| 8 | 1.0408 | 2.1097 | 1.9592 | 1.8903 | - | 112.2259 |
| 9 | 1 | -0.1175 | 1.9592 | 1.8904 | 0.0790 | 112.2284 |

Результат -

Изображение выглядит как текст, линия, снимок экрана, диаграмма

Автоматически созданное описание

Рис. 4 Траектория поиска решения методом возможных направлений

Листинг 4 – Код программы

function [] = Main()

clc;

close all;

clear all;

mas = [-6,-9,4,20,60,-1,1,1,1,1,0,4,6,2,1,9,36];

start\_x1 =0;

start\_x2 = 0;

stop\_x1 = 3;

stop\_x2 = 4;

fprintf('Метод возможных направлений:\n');

[x1\_voz,x2\_voz] = VozmNapr (mas);

[ H4 ] =PlotGraph( mas,x1\_voz,x2\_voz,start\_x1,start\_x2,stop\_x1,stop\_x2);

hold on;

ellipse(sqrt(mas(17)/mas(15)),sqrt(mas(17)/mas(16)),0,0,0,'b',300)

legend('Линии равного уровня','Ход решения','Ограничения');

hold off;

end

function [x,y] = VozmNapr (mas)

x = [1;1];

h = [mas(1)\*2,mas(3);mas(3),mas(2)\*2];

u=100;

j=1;

i=0;

k=-1;

while k<=10

y2=sqrt((mas(17)-mas(15)\*k\*k)/mas(16));

y=[k,y2];

y1y2(k+2,:)=y;

k=k+1;

end

while (u>0.1)

x1 = x(1,1); x2 = x(2,1);

grf = [mas(1)\*2\*x1 + mas(3)\*x2 + mas(4);mas(2)\*2\*x2 + mas(3)\*x1 + mas(5)];

grg =[(-mas(15)\*2\*x1);(-mas(16)\*2\*x2)];

if(-mas(15)\*x1^2 - mas(16)\*x2.^2 + mas(17) <10^-6)

if(grg(2,1)==grf(2,1))

k1 = 0;

if(grg(2,1)>=0)

u = grg(2,1);

k2 = 1;

else

u = -grg(2,1);

k2 = -1;

end

else

d = (grf(1,1)\*grg(2,1)-grg(1,1)\*grf(2,1))/(grg(2,1)-grf(2,1));

if(d>=0)

u = d;

k1 = 1;

else

u = -d;

k1 =-1;

end

k2 = (u-grf(1,1)\*k1)/grf(2,1);

end

K = [k1;k2];

t = -(grf'\*K)/(K'\*h\*K);

else

i=1;

K = -inv(h)\*grf;

u=1;

t=1;

end

k1=K(1,1)

k2=K(2,1)

x1x2(:,j) = x;

disp('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_')

disp(' Номер шага: '), disp(j),

disp(' Вычисленные значения для X: '),

disp(' x1 ; x2 = '), disp(x(:,1)),

disp('u= '), disp(u)

disp(' Значение функции: ')

f=mas(1)\*x1.^2+mas(2)\*x2.^2+mas(3)\*x1\*x2+mas(4)\*x1+mas(5)\*x2;

disp(f)

if (u>0)

a(j) = mas(15)\*k1.^2+mas(16)\*k2.^2;

b(j) = mas(15)\*x1.\*k1+mas(16)\*x2.\*k2;

c(j) = mas(15)\*x1.^2+mas(16)\*x2.^2-mas(17);

t1(j) = (-b(j)+sqrt(b(j)\*b(j)-a(j)\*c(j)))/(a(j));

t2(j) = (-b(j)-sqrt(b(j)\*b(j)-a(j)\*c(j)))/(a(j));

t=min([t max([t1(j) t2(j)])]);

x=x+t\*K;

end

j = j + 1;

i=0;

end

x = x1x2(1,:);

y = x1x2(2,:);

end

function [handler] = PlotGraph( mas,x1,x2,start\_x1,start\_x2,stop\_x1,stop\_x2)

x\_1=start\_x1:.1:stop\_x1;

x\_2=start\_x2:.1:stop\_x2;

[x\_1,x\_2]=meshgrid(x\_1,x\_2);

w=(mas(1)\*x\_1.^2 +mas(2)\*x\_2.^2 + mas(3)\*x\_1.\*x\_2 + mas(4)\*x\_1 + mas(5)\*x\_2);

handler = figure;

contour(x\_1,x\_2,w,50)

hold on;

plot (x1, x2, 'o-k','LineWidth', 2);

xlabel('x1');

ylabel('x2');

hold off

end

1. Метод штрафных функций

Формула штрафной функции

Решим эту задачу при помощи скрипта MatLab.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Шаг |  |  |  |  |
| 0 | 0.0001 | 2.999989 | 3.999992 | 149.9999 |
| 1 | 0.001 | 2.99989 | 3.99992 | 149.999 |
| 2 | 0.01 | 2.9989 | 3.9992 | 149.990009 |
| 3 | 0.1 | 2.89 | 3.92 | 149.095 |
| 4 | 10 | 2.421053 | 3.578947 | 147.368421 |
| 5 | 100 | 2.421053 | 3.578947 | 147.368421 |
| 6 | 1000 | 2.421052 | 3.578947 | 147.368419 |

Результат -

Изображение выглядит как текст, линия, снимок экрана, диаграмма

Автоматически созданное описание

Рис. 5 - Траектория поиска методом штрафных функций

Листинг 5 – Код программы

function [] = Main()

clc;

clearvars;

close all;

C = [-6, -9, 4, 20, 60];

x\_1(1) = 6;

x\_2(1) = 6;

a\_k = [-1 1; 1 1; -1 0; 0 -1];

b\_k = [4 6 0 0];

p = 1;

u\_mult = 10;

u = 0.0001;

delta = 0.000001;

f = @(x) C(1)\*(x(1))^2 + C(2)\*(x(2))^2 + C(3)\*x(1)\*x(2) + C(4)\*x(1) + C(5)\*x(2);

g\_1 = @(x) a\_k(1,1)\*x(1) + a\_k(1,2)\*x(2) - b\_k(1);

g\_2 = @(x) a\_k(2,1)\*x(1) + a\_k(2,2)\*x(2) - b\_k(2);

g\_3 = @(x) a\_k(3,1)\*x(1) + a\_k(3,2)\*x(2) - b\_k(3);

g\_4 = @(x) a\_k(4,1)\*x(1) + a\_k(4,2)\*x(2) - b\_k(4);

penalty = @(y) max(0, y)^p;

i = 1;

x\_delta = inf;

while x\_delta > delta

fm = @(x) -(f(x) - u\*(penalty(g\_1(x)) + penalty(g\_2(x)) + penalty(g\_3(x)) + penalty(g\_4(x))));

x0 = [x\_1(i), x\_2(i)];

A = [];

b = [];

x = fmincon(fm, x0, A, b);

x\_1(i+1) = x(1);

x\_2(i+1) = x(2);

x\_delta = sqrt((x\_1(i+1) - x\_1(i))^2 + (x\_2(i+1) - x\_2(i))^2);

fprintf("Номер шага i = %d\n", i - 1);

fprintf("u = %f\n", u);

fprintf("x = [%f %f]\n", x(1), x(2));

fprintf("F(x, u) = %f\n", -fm(x));

fprintf("\n");

i = i + 1;

u = u\_mult\*u;

end

x1\_range = -3:.1:10;

x2\_range = -1:.1:10;

[x1\_range, x2\_range] = meshgrid(x1\_range, x2\_range);

levels = ( C(1)\*x1\_range.^2 + C(2)\*x2\_range.^2 + C(3)\*x1\_range.\*x2\_range + C(4)\*x1\_range + C(5)\*x2\_range );

figure;

contour(x1\_range, x2\_range, levels, 50, 'DisplayName', 'Линии уровня');

hold on;

x1\_1 = -0.5:.1:10;

x2\_1 = (4 + 1.\*x1\_1)./1;

x1\_2 = -2:.1:10;

x2\_2 = (6 - 1.\*x1\_2)./1;

x2\_3 = -2:.1:10;

x1\_3 = 0.\*x2\_3;

x1\_4 = -2:.1:10;

x2\_4 = 0.\*x1\_4;

plot(x1\_1, x2\_1, '-.k', 'DisplayName', 'Ограничение 1');

plot(x1\_2, x2\_2, '--k', 'DisplayName', 'Ограничение 2');

plot(x1\_3, x2\_3, '-k', 'DisplayName', 'Ограничение 3');

plot(x1\_4, x2\_4, '-k', 'DisplayName', 'Ограничение 4');

plot(x\_1, x\_2, '-r.', 'DisplayName', 'Траектория поиска');

x0 = [x\_1(1), x\_2(1)];

fm = @(x) -f(x);

x = fmincon(fm, x0, a\_k, b\_k);

plot(x(1), x(2), '-square', 'DisplayName', 'Оптимальная точка')

fprintf("x\* = [%f %f]\n", x);

fprintf("F(x\*) = %f\n", f(x));

fprintf("F(x\*, u) = %f\n", -fm(x));

legend

end

1. Метод барьерных функций

Формула штрафной функции

Также воспользуемся MatLab.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Шаг |  |  |  |  |
| 0 | 1000 | 2.731635 | 1.444239 | 3732.433874 |
| 1 | 100 | 2.426172 | 1.972446 | 468.699136 |
| 2 | 10 | 2.198467 | 3.068059 | 168.716625 |
| 3 | 1.0 | 2.353952 | 3.492088 | 147.773509 |
| 4 | 0.1 | 2.412540 | 3.568949 | 147.190864 |
| 5 | 0.01 | 2.420177 | 3.577929 | 147.327784 |
| 6 | 0.001 | 2.420971 | 3.578839 | 147.362056 |
| 7 | 0.0001 | 2.421105 | 3.578876 | 147.367554 |
| 8 | 0.00001 | 2.421067 | 3.578932 | 147.368311 |
| 9 | 0.000001 | 2.421071 | 3.578928 | 147.368408 |
| 10 | 0 | 2.421072 | 3.578928 | 147.368419 |

Результат -

Изображение выглядит как текст, линия, снимок экрана, диаграмма

Автоматически созданное описание

Рис. 6 - Траектория поиска решения методом барьерных функций

Листинг 6 – Код программы

function [] = Main()

clc;

clearvars;

close all;

C = [-6, -9, 4, 20, 60];

x\_1(1) = 1;

x\_2(1) = 1;

a\_k = [-1 1; 1 1; -1 0; 0 -1];

b\_k = [4 6 0 0];

u\_mult = 0.1;

u = 1000;

delta = 0.000001;

f = @(x) C(1)\*(x(1))^2 + C(2)\*(x(2))^2 + C(3)\*x(1)\*x(2) + C(4)\*x(1) + C(5)\*x(2);

g\_1 = @(x) a\_k(1,1)\*x(1) + a\_k(1,2)\*x(2) - b\_k(1);

g\_2 = @(x) a\_k(2,1)\*x(1) + a\_k(2,2)\*x(2) - b\_k(2);

g\_3 = @(x) a\_k(3,1)\*x(1) + a\_k(3,2)\*x(2) - b\_k(3);

g\_4 = @(x) a\_k(4,1)\*x(1) + a\_k(4,2)\*x(2) - b\_k(4);

barrier = @(y) -log(-y);

i = 1;

x\_delta = inf;

while x\_delta > delta

fm = @(x) -(f(x) - u\*(barrier(g\_1(x)) + barrier(g\_2(x)) + barrier(g\_3(x)) + barrier(g\_4(x))));

x0 = [x\_1(i), x\_2(i)];

A = [];

b = [];

x = fmincon(fm, x0, A, b);

x\_1(i+1) = x(1);

x\_2(i+1) = x(2);

x\_delta = sqrt((x\_1(i+1) - x\_1(i))^2 + (x\_2(i+1) - x\_2(i))^2);

fprintf("Номер шага i = %d\n", i - 1);

fprintf("u = %f\n", u);

fprintf("x = [%f %f]\n", x(1), x(2));

fprintf("F(x, u) = %f\n", -fm(x));

fprintf("\n");

i = i + 1;

u = u\_mult\*u;

end

x1\_range = -3:.1:10;

x2\_range = -1:.1:10;

[x1\_range, x2\_range] = meshgrid(x1\_range, x2\_range);

levels = ( C(1)\*x1\_range.^2 + C(2)\*x2\_range.^2 + C(3)\*x1\_range.\*x2\_range + C(4)\*x1\_range + C(5)\*x2\_range );

figure;

contour(x1\_range, x2\_range, levels, 50, 'DisplayName', 'Линии равного уровня');

hold on;

x1\_1 = -0.5:.1:10;

x2\_1 = (4 + 1.\*x1\_1)./1;

x1\_2 = -2:.1:10;

x2\_2 = (6 - 1.\*x1\_2)./1;

x2\_3 = -2:.1:10;

x1\_3 = 0.\*x2\_3;

x1\_4 = -2:.1:10;

x2\_4 = 0.\*x1\_4;

plot(x1\_1, x2\_1, '-.k', 'DisplayName', 'Первое ограничение');

plot(x1\_2, x2\_2, '--k', 'DisplayName', 'Второе ограничение');

plot(x1\_3, x2\_3, '-k', 'DisplayName', 'Ограниение по x1');

plot(x1\_4, x2\_4, '-k', 'DisplayName', 'Ограниение по x2');

plot(x\_1, x\_2, '-r.', 'DisplayName', 'Траектория поиска решения');

x0 = [x\_1(1), x\_2(1)];

fm = @(x) -f(x);

x = fmincon(fm, x0, a\_k, b\_k);

plot(x(1), x(2), '-square', 'DisplayName', 'Оптимальная точка')

fprintf("x\* = [%f %f]\n", x);

fprintf("F(x\*) = %f\n", f(x));

fprintf("F(x\*, u) = %f\n", -fm(x));

legend

end

1. Метод Франка-Вульфа

Исходная задача:

Введём векторы:

Тогда целевая функция для минимизации:

Система ограничений:

Умножим последние две строки на -1

Добавим столбцы k1 и k2

Теперь задача выглядит следующим образом

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  | *1* | *-1* | *0* | *0* | *0* | *0* | *4* |
|  | *-1* | *-1* | *0* | *0* | *0* | *0* | *6* |
|  | *-12* | *4* | *1* | *0* | *1* | *-1* | *20* |
|  | *4* | *-18* | *0* | *1* | *-1* | *-1* | *60* |
|  | *8* | *14* | *-1* | *-1* | *0* | *2* | *-80* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  | *-14* | *-1* | *0* | *1* | *-1* | *-1* | *-12* |
|  | *22* | *-1* | *0* | *1* | *-1* | *-1* | *-48* |
|  | *200* | *4* | *-18* | *-4* | *-14* | *22* | *-600* |
|  | *-4* | *1* | *0* | *-1* | *1* | *1* | *-60* |
|  | *-200* | *14* | *18* | *4* | *14* | *-22* | *600* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

Пересчитаем значение функции

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Результат -

Изображение выглядит как текст, снимок экрана, диаграмма, линия

Автоматически созданное описание

Рис. 7 - Траектория поиска решения методом Франка-Вульфа

Листинг 7 – Код программы

function [] = Main()

clc;

close all;

clear all;

mas = [-6,-9,4,20,60,-1,1,1,1,1,0,4,6,2,1,9,36];

start\_x1 =0;

start\_x2 = 0;

stop\_x1 = 8;

stop\_x2 = 8;

x1\_fw=[0, 0, 2.18182, 2.42105];

x2\_fw=[0, 3.333333, 3.81818, 3.57895];

[ H2 ] = PlotFW( mas,x1\_fw,x2\_fw,start\_x1,start\_x2,stop\_x1,stop\_x2 );

end

function [ handler ] = PlotFW( mas,x1\_fw,x2\_fw,start\_x1,start\_x2,stop\_x1,stop\_x2 )

x=start\_x1:.1:stop\_x1;

y=start\_x2:.1:stop\_x2;

[ H ] =PlotGraph( mas,x1\_fw,x2\_fw,start\_x1,start\_x2,stop\_x1,stop\_x2);

handler =H;

hold on;

plot (x,x\*0,':b','LineWidth', 1);

plot (0\*y,y,':b','LineWidth', 1);

if (mas(7)==0)

plot (x, (mas(12)-mas(6)\*x),':b','LineWidth', 1);

else

plot (x, (mas(12)-mas(6)\*x)/mas(7),':b','LineWidth', 1);

end;

if (mas(9)==0)

plot (x, (mas(13)-mas(8)\*x),':b','LineWidth', 1);

else

plot (x, (mas(13)-mas(8)\*x)/mas(9),':b','LineWidth', 1);

end;

legend('Линии равного уровня','Ход решения','Ограничения');

hold off;

end

function [handler] = PlotGraph( mas,x1,x2,start\_x1,start\_x2,stop\_x1,stop\_x2)

x\_1=start\_x1:.1:stop\_x1;

x\_2=start\_x2:.1:stop\_x2;

[x\_1,x\_2]=meshgrid(x\_1,x\_2);

w=(mas(1)\*x\_1.^2 +mas(2)\*x\_2.^2 + mas(3)\*x\_1.\*x\_2 + mas(4)\*x\_1 + mas(5)\*x\_2);

handler = figure;

contour(x\_1,x\_2,w,50)

hold on;

plot (x1, x2, 'o-k','LineWidth', 2);

xlabel('x1');

ylabel('x2');

hold off

end