# Sprawozdanie z laboratorium Teorii Optymalizacji

| | |
|---|---|
| **Imię i nazwisko** | Jacek Gołda |
| **Temat ćwiczenia** | Metody zmiennej metryki |
| **Data i godzina wykonania ćwiczenia** | 6 kwietnia 2016, 14:00 |

## 1 Wstęp

Celem laboratorium było zbadanie metod zmiennej metryki, w szczególności porównanie efektywności i wyników ich działania.

## 2 Ćwiczenie 3

### Treść zadania:

Zaobserwować działania różnych metod zmiennej metryki dla funkcjonału:

$$Q(x_1, x_2) = x_1^2 + a \cdot x_2^2$$

Wybierając kolejno $a = 1$, $a = 20n$, $a = 12n$

Za parametr $n$ przyjęto wartość 4.

### Rozwiązanie:

### 2.1 Rozwiązanie analityczne

Zauważono, że funkcjonał celu jest sumą dwóch wyrażeń nieujemnych, zerujących się w zerze. Wynika stąd, że minimum funkcjonału będzie osiągane w punkcie $\widehat{x} = [0, 0]^T$ i wartość minimalna funkcji będzie wynosiła 0.

### 2.2 Rozwiązanie numeryczne

W celu zrealizowania rozwiązania numerycznego, konieczne było zmodyfikowanie funkcji obliczających wartość funkcjonału celu i gradientu funkcjonału celu. Gradient obliczono za pomocą toolboxu do obliczeń symbolicznych. Zmodyfikowano również pliki, w których te funkcje były wywoływane (dodano argument `a` do wywołań) — nie zamieszczono tych plików w sprawozdaniu, ponieważ niewiele by do niego wnosiły)

Listing 1: Wartość funkcjonału celu

```
function [q,x]=koszt(a, x,z,d)

if nargin==3, x=x+z;
elseif nargin==4, x=x+z*d;
end
q=x(1)^2+a*x(2)^2;
```

Listing 2: Gradient funkcjonału celu

```
function g=gradie(a, x)

g=[2*x(1)
   2 * a * x(2)];
```

Ponownie wykorzystano funkcje służące do rysowania poziomic funkcjonału celu i kolejnych przybliżeń znalezionych przez algorytm (pojedynczego rozwiązania). Zostały one napisane podczas poprzednich laboratoriów, znajdują się w poprzednim sprawozdaniu, nie zamieszczano ich ponownie w tym sprawozdaniu.

Następnie napisano skrypt minimalizujący funkcjonał i rysujący rozwiązania:

```
clear all;
close all;
clc
diary('wyjscie_z_konsoli');
diary on

n = 4;

for par=0:5

fprintf('\tMetoda dla parametru %d\n', par)


% DLA WSPOLCZYNNIKA a = 1
a = 1;
fprintf('a = %d\n', a)

% rysowanie poziomic
[x1, x2] = meshgrid(-20:0.01:20, -20:0.01:20);
q = x1.^2 + a * x2 .^ 2;
poziomice = [0.01, 10, 100, 250, 400];

w_x0 = [20, 0;
        -15, -15;
        0, 20];

figureHandle = figure;
figureHandle = rysujMape(figureHandle, x1, x2, q, poziomice);
for i=1:size(w_x0, 1)
    x0 = w_x0(i, :)';
    zmenad
    figureHandle = rysujRozwiazanie(figureHandle, x_rozw);
    display(sprintf('ilosc iteracji (lacznie z x0) to: %d', size(x_rozw, 2)));
end
print('-depsc2',sprintf('wykres_n_%d_a_1.eps', par))



% DLA WSPOLCZYNNIKA a = 80 (20n)
a = 20*n;
fprintf('a = %d\n', a)

% rysowanie poziomic
[x1, x2] = meshgrid(-20:0.01:20, -3:0.01:3);
q = x1.^2 + a * x2 .^ 2;
poziomice = [0.01, 10, 100, 250, 400];

w_x0 = [20, 0;
        15, -5;
```

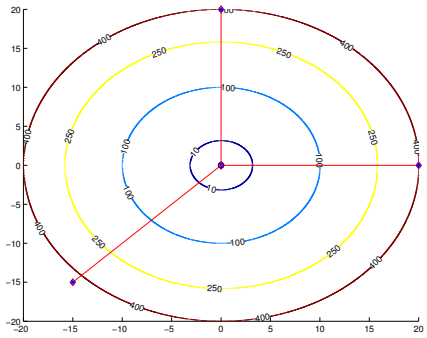
```
        0, 7;
        -20, -0.5];

figureHandle = figure;
figureHandle = rysujMape(figureHandle, x1, x2, q, poziomice);
for i=1:size(w_x0, 1)
    x0 = w_x0(i, :)';
    zmenad
    figureHandle = rysujRozwiazanie(figureHandle, x_rozw);
    display(sprintf('ilosc iteracji (lacznie z x0) to: %d', size(x_rozw, 2)));
end
print('-depsc2',sprintf('wykres_n_%d_a_80.eps', par))



% DLA WSPOLCZYNNIKA a = 48;
a = 12*n;
fprintf('a = %d\n', a)

% rysowanie poziomic
[x1, x2] = meshgrid(-10:0.01:10, -2:0.01:2);
q = x1.^2 + a * x2 .^ 2;
poziomice = [0.01, 3, 10,  30, 70, 100];

w_x0 = [10, 0;
        -10, -5;
        7, 20];

figureHandle = figure;
figureHandle = rysujMape(figureHandle, x1, x2, q, poziomice);
for i=1:size(w_x0, 1)
    x0 = w_x0(i, :)';
    zmenad
    figureHandle = rysujRozwiazanie(figureHandle, x_rozw);
    display(sprintf('ilosc iteracji (lacznie z x0) to: %d', size(x_rozw, 2)));
end
print('-depsc2',sprintf('wykres_n_%d_a_48.eps', par))
end

diary off
```

Dla kolejnych wartości współczynników $a$ badano następujące punkty początkowe:

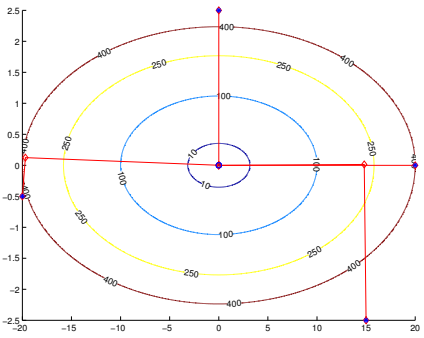
| Współczynnik $a$ | $x_1$ | $x_2$ |
|---|---|---|
| 1 | 20 | 0 |
| | -15 | -15 |
| | 0 | 20 |
| 80 | 20 | 0 |
| | 15 | -2.5 |
| | 0 | 2.5 |
| | -20 | -0.5 |
| 48 | 10 | 0 |
| | -10 | -1.5 |
| | 7 | 1.5 |

Uzyskano następujące wyniki:

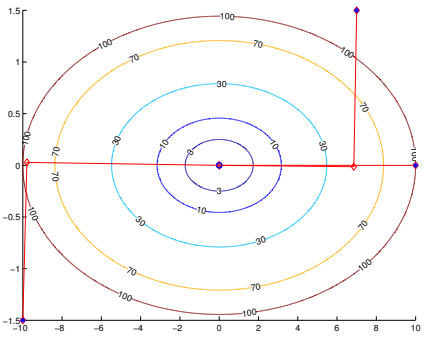
### Metoda Davidona - Fletchera - Powella



(a) $a = 1$

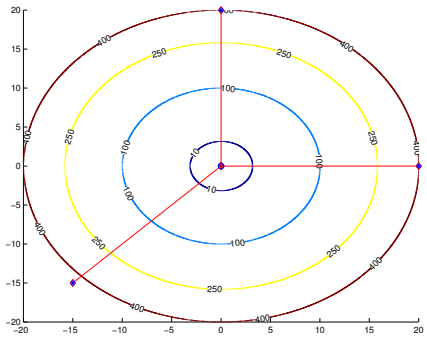


(b) $a = 80$

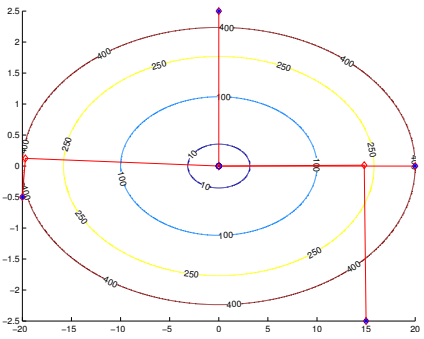


(c) $a = 48$

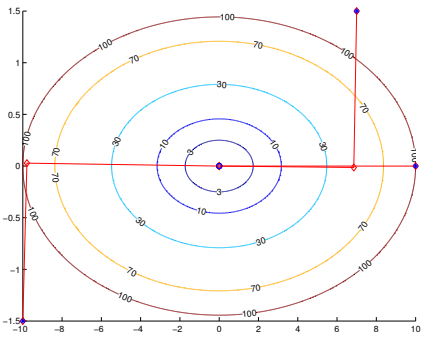
### Metoda Wolfe'a - Broydena - Davidona



(a) $a = 1$

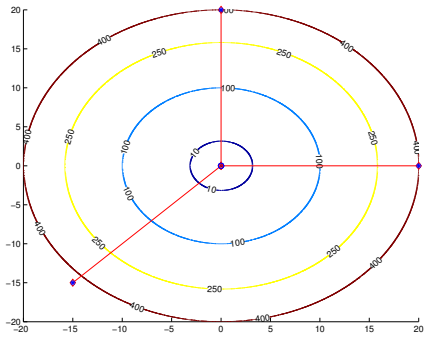


(b) $a = 80$

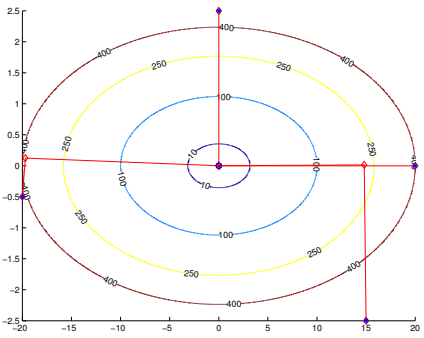


(c) $a = 48$

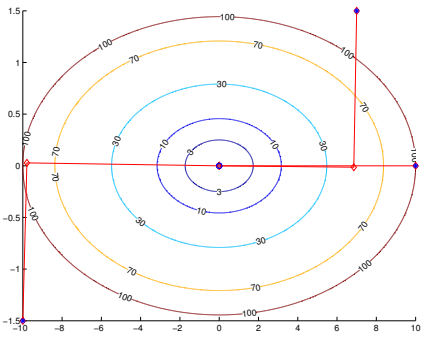
### Metoda Broydena - Fletchera - Goldfarba - Shanno



(a) $a = 1$

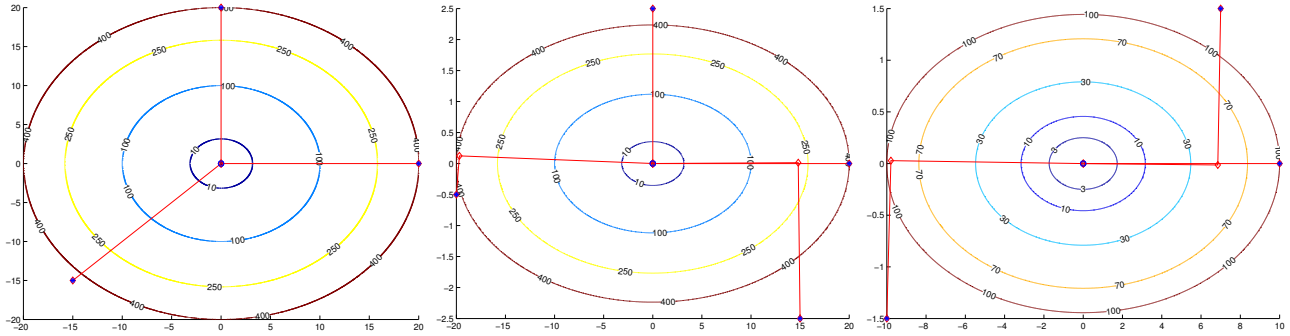


(b) $a = 80$



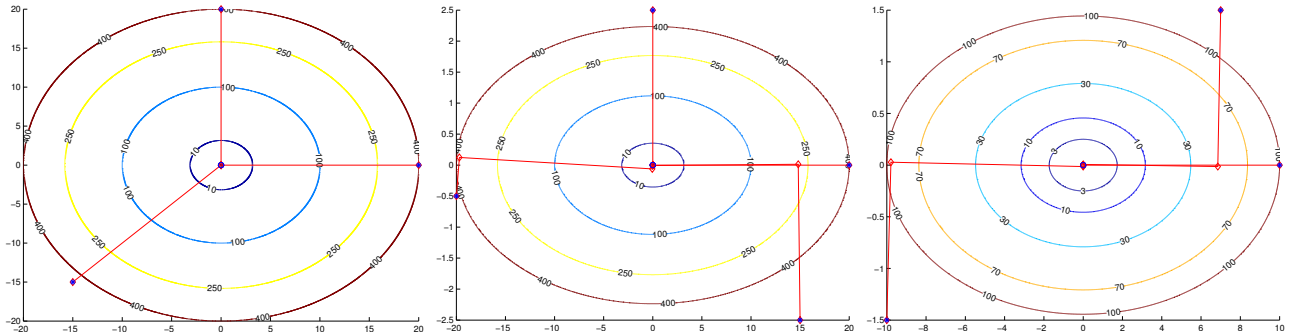
(c) $a = 48$

### Metoda Pearsona 1



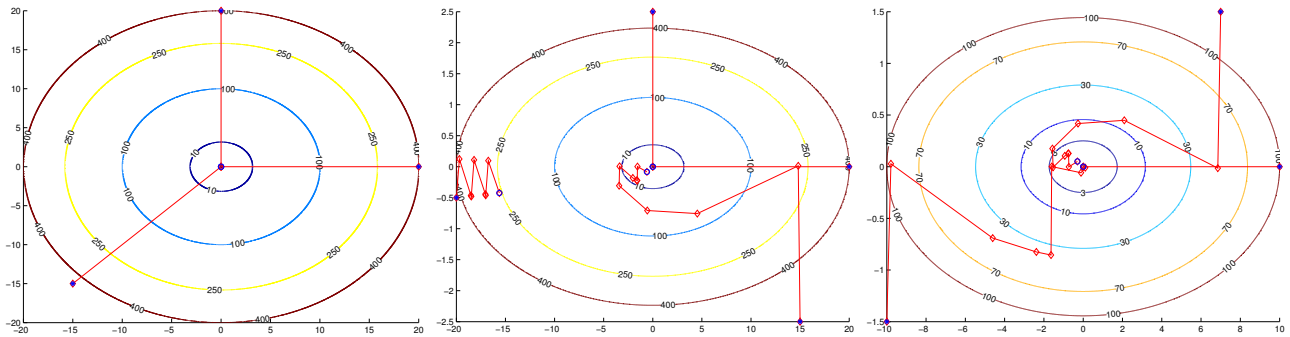
(a) $a = 1$ (b) $a = 80$ (c) $a = 48$

### Metoda Pearsona 2



(a) $a = 1$ (b) $a = 80$ (c) $a = 48$

### Metoda McCormicka



(a) $a = 1$ (b) $a = 80$ (c) $a = 48$

Z wykresów można wyciągnąć następujące wnioski:

1. Gdy poziomice są okręgami, wszystkie algorytmy znajdują minimum w pierwszej iteracji bez względu na dobór punktu początkowego, ponieważ dla dowolnego punktu początkowego pierwsza wyznaczona prosta przechodzi przez punkt w którym znajduje się minimum i w efekcie algorytm kończy pracę po pierwszej iteracji.

2. W przypadku kiedy zbiory poziomicowe są elipsami, dobór punktu początkowego jest bardzo istotny. Jeżeli punkt początkowy znajdzie się na osi elipsy, minimum zostanie znalezione w pierwszej iteracji (oczywiście w praktyce dobór takiego punktu początkowego jest nierealny — wymagałby znajomości położenia minimum). W przeciwnym wypadku algorytm wykonuje więcej iteracji. Tutaj zaczynają być widoczne różnice pomiędzy algorytmami: pierwsze

trzy sprawują się bardzo dobrze, wykonując trzy iteracje w przypadku takiego punktu początkowego. Drugi algorytm Pearsona wykonuje 5 iteracji, czyli nieco więcej, a algorytm McCormicka wyczerpuje limit 10 iteracji, więc nie wiadomo, czy zakończył pracę w wyniku znalezienia minimum, czy w wyniku wyczerpania liczby iteracji. Przez ogląd można stwierdzić, że w trzech z czterech przypadków zakończył pracę w wyniku wyczerpania liczby iteracji. W jednym przypadku nie da się tego sprawdzić tylko przez ogląd wykresów.

# 3 Ćwiczenie 4

## Treść zadania:

*Badanie porównawcze gradientowych metod optymalizacji*
Badanie porównawcze metod zmiennej metryki przeprowadzić dla funkcjonału:

$$Q(x_1, x_2) = 6x_1^2 + 6x_1x_2 + x_2^2 + 4.5\big(e^{x_1} - x_1 - 1\big) + 1.5\big(e^{x_2} - x_2 - 1\big)$$

Przyjąć, że norma gradientu w punkcie zatrzymania algorytmu ma być nie większa niż 0.001. Pole obserwacji graficznej: $|x_1| \leqslant 3$, $|x_2| \leqslant 3$. Punkt startowy: $(-3, 3)$ (dla metody największego spadku wypróbować też punktu $(-3, 1)$). Maksymalna liczba iteracji: 20. Maksymalny czas obserwacji: 7 minut.

Wszystkie wartości $x_1$, $x_2$ przesunięte są o $3n$

Za parametr $n$ przyjęto wartość 4.

## 3.1 Rozwiązanie numeryczne

Ponownie zmodyfikowano funkcje obliczające wartość funkcjonału celu i jego gradientu (ponownie skorzystano z toolboxu do obliczeń symbolicznych). Korzystano z plików zmodyfikowanych przy okazji poprzedniego ćwiczenia, stąd nadal przekazywany jest parametr `a`, jednak jest on ignorowany. Skorzystano również z funkcji rysujących poziomice funkcjonału celu i kolejne przybliżenia rozwiązania optymalnego wykorzystywane w poprzednim zadaniu.

Listing 3: Wartość funkcjonału celu

```
function [q,x]=koszt(a, x,z,d)

if nargin==3, x=x+z;
elseif nargin==4, x=x+z*d;
end

n = 4;
x1 = x(1) - 3 * n;
x2 = x(2) - 3 * n;

q=6*x1 ^2 + 6*x1*x2 + x2 ^ 2 + ...
    4.5*(exp(x1) - x1 - 1) + ...
    1.5 * (exp(x2) - x2 - 1);
```

Listing 4: Gradient funkcjonału celu

```
function g=gradie(a, x)

n = 4;
x = x - 3 * n;
x1 = x(1);
x2 = x(2);
```

```
g=[12*x1 + 6*x2 + (9*exp(x1))/2 - 9/2
  6*x1 + 2*x2 + (3*exp(x2))/2 - 3/2];
```

Skorzystano ze zmodyfikowanej wersji funkcji do opracowywania wyników. Obecnie zwraca ona różnicę pomiędzy rozwiązaniem optymalnym, a bieżącym przybliżeniem.

Listing 5: Funkcja opracowująca wyniki

```
function [argumenty, wartosci] = opracuj(arg, wart)
    ilosc_iteracji = size(arg, 2);
    ostatni_argument = [12;12];
    modul_x = arg - ostatni_argument * ones(1, ilosc_iteracji);
    argumenty = sqrt(modul_x(1, :) .^ 2 + modul_x(2, :) .^ 2 );
    argumenty = argumenty';
    ostatnia_wartosc = 0;
    wartosci = wart - ostatnia_wartosc * ones(1, ilosc_iteracji);
    wartosci = wartosci';
end
```

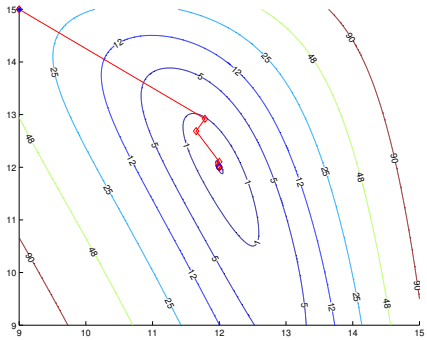
Napisano skrypt służący do rozwiązywania zadania.

```
clear all;
close all;
clc

% wspolczynnik wystepujacy w zadaniu
n = 4;
a = 0;

wartosci_poziomic = [
    0.01,
    1,
    5,
    12,
    25,
    48,
    90
    ];

[z1, z2] = meshgrid(9:0.01:15, 9:0.01:15);
x1 = z1 - 3*n;
x2 = z2 - 3*n;
q = 6 * x1 .^ 2 + 6 * x1 .* x2 + x2 .^ 2 + ...
    4.5 * ( exp(x1)    - x1 - 1) + ...
    1.5 * ( exp(x2) - x2 - 1);


% METODA Davidona Fletchera Powella

% rysowanie poziomic
figureHandle = figure;
figureHandle = rysujMape(figureHandle, z1, z2, q, wartosci_poziomic);

% wywolanie metody
x0 = [9; 15];
par = 0;
zmenad

% opracowanie wynikow
[modul_x, roznica_q] = opracuj(x_rozw, q_rozw);
dfp = [(1:length(modul_x))', modul_x, roznica_q, grad_rozw'];
```

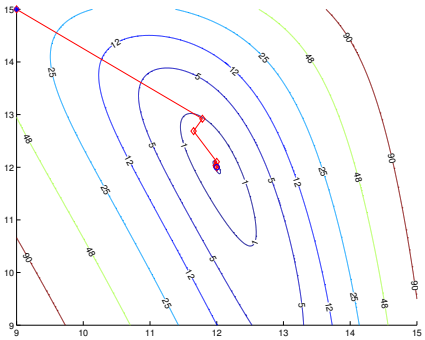
```
% rysuj punkty i linie
figureHandle = rysujRozwiazanie(figureHandle, x_rozw);
display(sprintf('ilosc iteracji to: %d', size(x_rozw, 2) - 1));
print -depsc2 'wykres_dfp.eps'


% METODA Wolfe'a Broydena Davidona

% rysowanie poziomic
figureHandle = figure;
figureHandle = rysujMape(figureHandle, z1, z2, q, wartosci_poziomic);

% wywolanie metody
x0 = [9; 15];
par = 1;
zmenad

% opracowanie wynikow
[modul_x, roznica_q] = opracuj(x_rozw, q_rozw);
wbd = [(1:length(modul_x))', modul_x, roznica_q, grad_rozw'];

figureHandle = rysujRozwiazanie(figureHandle, x_rozw);
display(sprintf('ilosc iteracji to: %d', size(x_rozw, 2) - 1));
print -depsc2 'wykres_wbd.eps'


% METODA Broydena Fletchera Goldfarba Shanno

% rysowanie poziomic
figureHandle = figure;
figureHandle = rysujMape(figureHandle, z1, z2, q, wartosci_poziomic);

% wywolanie metody
x0 = [9; 15];
par = 2;
zmenad

% opracowanie wynikow
[modul_x, roznica_q] = opracuj(x_rozw, q_rozw);
bfgs = [(1:length(modul_x))', modul_x, roznica_q, grad_rozw'];

% rysuj punkty i linie
figureHandle = rysujRozwiazanie(figureHandle, x_rozw);
display(sprintf('ilosc iteracji to: %d', size(x_rozw, 2) - 1));
print -depsc2 'wykres_bfgs.eps'


% METODA Pearsona 1

% rysowanie poziomic
figureHandle = figure;
figureHandle = rysujMape(figureHandle, z1, z2, q, wartosci_poziomic);

% wywolanie metody
x0 = [9; 15];
par = 3;
zmenad

% opracowanie wynikow
[modul_x, roznica_q] = opracuj(x_rozw, q_rozw);
p1 = [(1:length(modul_x))', modul_x, roznica_q, grad_rozw'];
```

```
% rysuj punkty i linie
figureHandle = rysujRozwiazanie(figureHandle, x_rozw);
display(sprintf('ilosc iteracji to: %d', size(x_rozw, 2) - 1));
print -depsc2 'wykres_p1.eps'


% METODA Pearsona 2

% rysowanie poziomic
figureHandle = figure;
figureHandle = rysujMape(figureHandle, z1, z2, q, wartosci_poziomic);

% wywolanie metody
x0 = [9; 15];
par = 4;
zmenad

% opracowanie wynikow
[modul_x, roznica_q] = opracuj(x_rozw, q_rozw);
p2 = [(1:length(modul_x))', modul_x, roznica_q, grad_rozw'];

% rysuj punkty i linie
figureHandle = rysujRozwiazanie(figureHandle, x_rozw);
display(sprintf('ilosc iteracji to: %d', size(x_rozw, 2) - 1));
print -depsc2 'wykres_p2.eps'

% METODA McCormicka

% rysowanie poziomic
figureHandle = figure;
figureHandle = rysujMape(figureHandle, z1, z2, q, wartosci_poziomic);

% wywolanie metody
x0 = [9; 15];
par = 5;
zmenad

% opracowanie wynikow
[modul_x, roznica_q] = opracuj(x_rozw, q_rozw);
m = [(1:length(modul_x))', modul_x, roznica_q, grad_rozw'];

% rysuj punkty i linie
figureHandle = rysujRozwiazanie(figureHandle, x_rozw);
display(sprintf('ilosc iteracji to: %d', size(x_rozw, 2) - 1));
print -depsc2 'wykres_m.eps'
```

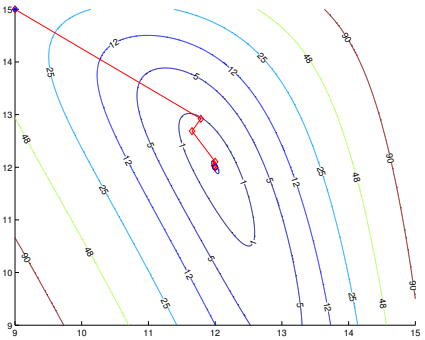
Uzyskano następujące wyniki:



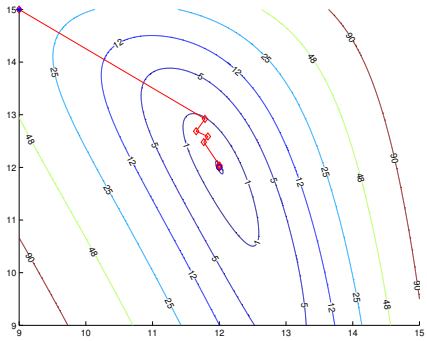
(a) Metoda Davidona - Fletchera - Powella



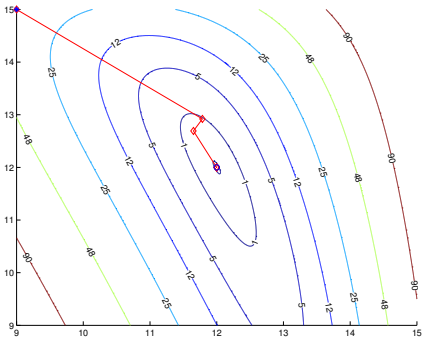
(b) Metoda Wolfe'a - Broydena - Davidona



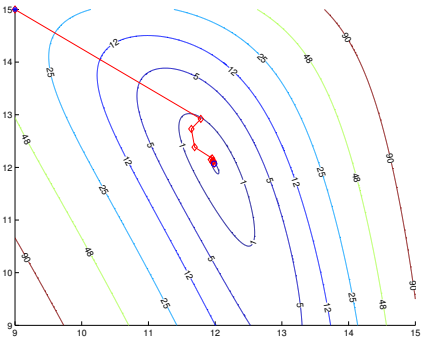
(c) Metoda Broydena - Fletchera - Goldfarba - Shanno



(a) Metoda Pearsona 1



(b) Metoda Pearsona 2



(c) Metoda McCormicka

Ilości wykonywanych iteracji zestawiono w tabeli:

| Metoda | Ilość iteracji |
|---|---|
| Davidona - Fletchera - Powella | 7 |
| Wolfe'a - Broydena - Davidona | 7 |
| Broydena - Fletchera - Goldfarba - Shanno | 7 |
| Pearsona 1 | 8 |
| Pearsona 2 | 8 |
| McCormicka | 10 |

Kolejne wartości odległości od rozwiązania optymalnego, a także różnic wartości funkcji celu bieżącego przybliżenia i rozwiązania optymalnego przedstawiono w tabelkach:

**Metoda Davidona - Fletchera - Powella**

| Numer iteracji | Odległość od minimum | Różnica wartości funkcji celu | Moduł gradientu |
|---|---|---|---|
| 1.0000 | 4.2426 | 42.3523 | 27.7978 |
| 2.0000 | 0.9455 | 0.9209 | 27.7978 |
| 3.0000 | 0.7695 | 0.4572 | 3.5051 |
| 4.0000 | 0.1024 | 0.0189 | 1.5666 |
| 5.0000 | 0.0324 | 0.0007 | 0.7222 |
| 6.0000 | 0.0295 | 0.0005 | 0.0692 |
| 7.0000 | 0.0000 | 0.0000 | 0.0411 |
| 8.0000 | 0.0000 | 0.0000 | 0.0002 |

**Metoda Wolfe'a - Broydena - Davidona**

| Numer iteracji | Odległość od minimum | Różnica wartości funkcji celu | Moduł gradientu |
|---|---|---|---|
| 1.0000 | 4.2426 | 42.3523 | 27.7978 |
| 2.0000 | 0.9455 | 0.9209 | 27.7978 |
| 3.0000 | 0.7694 | 0.4572 | 3.5051 |
| 4.0000 | 0.1031 | 0.0191 | 1.5678 |
| 5.0000 | 0.0325 | 0.0007 | 0.7252 |
| 6.0000 | 0.0296 | 0.0005 | 0.0698 |
| 7.0000 | 0.0000 | 0.0000 | 0.0412 |
| 8.0000 | 0.0000 | 0.0000 | 0.0002 |

**Metoda Broydena - Fletchera - Goldfarba - Shanno**

| Numer iteracji | Odległość od minimum | Różnica wartości funkcji celu | Moduł gradientu |
|---|---|---|---|
| 1.0000 | 4.2426 | 42.3523 | 27.7978 |
| 2.0000 | 0.9455 | 0.9209 | 27.7978 |
| 3.0000 | 0.7694 | 0.4572 | 3.5051 |
| 4.0000 | 0.1030 | 0.0190 | 1.5677 |
| 5.0000 | 0.0325 | 0.0007 | 0.7253 |
| 6.0000 | 0.0296 | 0.0005 | 0.0697 |
| 7.0000 | 0.0000 | 0.0000 | 0.0412 |
| 8.0000 | 0.0000 | 0.0000 | 0.0002 |

Wszystkie trzy powyższe metody sprawdziły się równie dobrze i najlepiej ze wszystkich — zakończyły pracę w wyniku znalezienia minimum. Wykonały po 7 iteracji (pierwsza to punkt startowy). Końcowa wartość odległości od minimum nie jest większa niż $10^{-4}$.

**Metoda Pearsona 1**

| Numer iteracji | Odległość od minimum | Różnica wartości funkcji celu | Moduł gradientu |
|---|---|---|---|
| 1.0000 | 4.2426 | 42.3523 | 27.7978 |
| 2.0000 | 0.9455 | 0.9209 | 27.7978 |
| 3.0000 | 0.7697 | 0.4575 | 3.5051 |
| 4.0000 | 0.6118 | 0.2976 | 1.5677 |
| 5.0000 | 0.5322 | 0.2015 | 1.5463 |
| 6.0000 | 0.0359 | 0.0033 | 0.9806 |
| 7.0000 | 0.0296 | 0.0005 | 0.3228 |
| 8.0000 | 0.0057 | 0.0001 | 0.0348 |
| 9.0000 | 0.0001 | 0.0000 | 0.0500 |

**Metoda Pearsona 2**

| Numer iteracji | Odległość od minimum | Różnica wartości funkcji celu | Moduł gradientu |
|---|---|---|---|
| 1.0000 | 4.2426 | 42.3523 | 27.7978 |
| 2.0000 | 0.9455 | 0.9209 | 27.7978 |
| 3.0000 | 0.7753 | 0.4637 | 3.5051 |
| 4.0000 | 0.0019 | 0.0000 | 1.5664 |
| 5.0000 | 0.0009 | 0.0000 | 0.0187 |
| 6.0000 | 0.0008 | 0.0000 | 0.0013 |
| 7.0000 | 0.0001 | 0.0000 | 0.0016 |
| 8.0000 | 0.0000 | 0.0000 | 0.0008 |
| 9.0000 | 0.0000 | 0.0000 | 0.0001 |

Metody zakończyły pracę po 8 iteracjach — o jedną więcej niż powyższe. Również zatrzymały się w wyniku znalezienia minimum. W przypadku pierwszej metody Pearsona widać, że znalezione przez nią przybliżenie jest dalej od rozwiązania optymalnego niż w przypadku poprzednich metod. Druga metoda Pearsona sprawdziła się znacznie lepiej — znalazła tak dobre rozwiązanie jak poprzednie metody, lecz poświęciła na to jedną iterację więcej.

**Metoda McCormicka**

| Numer iteracji | Odległość od minimum | Różnica wartości funkcji celu | Moduł gradientu |
|---|---|---|---|
| 1.0000 | 4.2426 | 42.3523 | 27.7978 |
| 2.0000 | 0.9455 | 0.9209 | 27.7978 |
| 3.0000 | 0.8125 | 0.5068 | 3.5051 |
| 4.0000 | 0.4920 | 0.3237 | 1.5605 |
| 5.0000 | 0.1787 | 0.0233 | 2.5938 |
| 6.0000 | 0.1622 | 0.0162 | 0.4547 |
| 7.0000 | 0.1262 | 0.0114 | 0.2300 |
| 8.0000 | 0.1162 | 0.0111 | 0.3122 |
| 9.0000 | 0.1155 | 0.0111 | 0.3778 |
| 10.0000 | 0.1055 | 0.0067 | 0.3830 |
| 11.0000 | 0.0704 | 0.0040 | 0.1367 |

Ponownie najgorzej sprawdziła się metoda McCormicka. Wykorzystała wszystkie 10 iteracji. Jak widać z wartości w tabelce, ostatnie znalezione rozwiązanie jest znacznie dalej od minimum niż w przypadku pozostałych metod.

# 4 Ćwiczenie 5

## Treść zadania:

„Dolina bananowa" Rossenbrocka. Wyznaczyć minimum funkcji:

$$Q(x_1, x_2) = 30n(x_2 - x_1^2)^2 + (1 - x_1)^2$$

na mapę poziomic doliny nanieść punkty pośrednie poszczególnych kroków.

Za parametr $n$ przyjęto wartość 4.

## Rozwiązanie:

### 4.1 Rozwiązanie analityczne

Zauważono, że funkcja składa się z dwóch nieujemnych wyrażeń. Po przyrównaniu ich do zera, uzyskano, że minimum będzie znajdować się w punkcie $\widehat{x} = [1, 1]^T$

### 4.2 Rozwiązanie numeryczne

Zmodyfikowano funkcje obliczające wartość funkcji celu i gradientu:

Listing 6: Wartość funkcjonału celu

```
function [q,x]=koszt(a, x,z,d)

if nargin==3, x=x+z;
elseif nargin==4, x=x+z*d;
end

n = 4;
x1 = x(1);
x2 = x(2);

q=30 * n * (x2 - x1 .^ 2 ) .^ 2 + (1 - x1 ).^ 2;
```

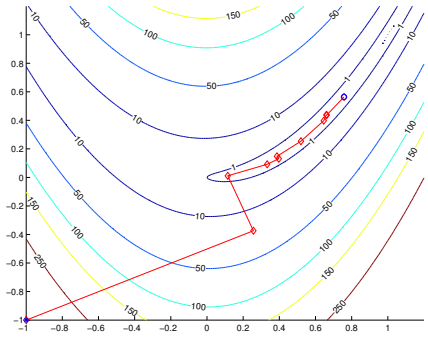
Listing 7: Gradient funkcjonału celu

```
function g=gradie(a, x)

n = 4;
x1 = x(1);
x2 = x(2);

g=[2*x1 - 120*n*x1*(- x1^2 + x2) - 2
            30*n*(- 2*x1^2 + 2*x2)];
```

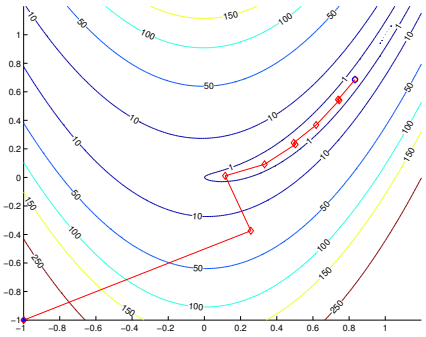
Zadanie rozwiązano bardzo zbliżonym skryptem, jak zadanie poprzednie. Zmodyfikowano jedynie wartości, dla których rysowane są poziomice i punkty początkowe. Nie zamieszczono skryptu, ponieważ nie wnosiłby dużo do sprawozdania.

W przypadku wszystkich metod jako punkt początkowy obrano punkt $x_0 = [-1, -1]^T$

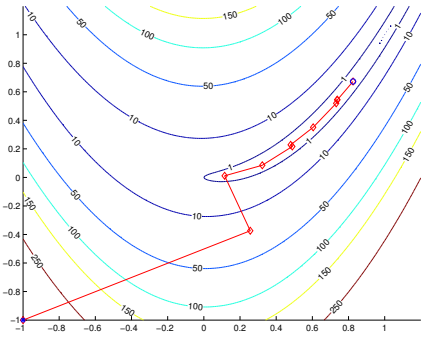
Uzyskano następujące wyniki:



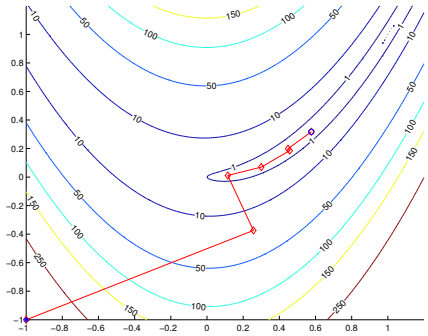
(a) Metoda Davidona - Fletchera - Powella



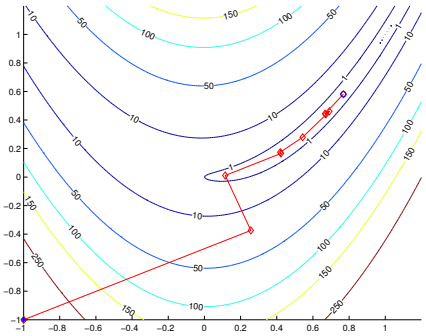
(b) Metoda Wolfe'a - Broydena - Davidona



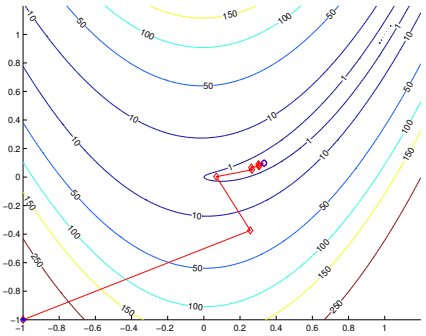
(c) Metoda Broydena - Fletchera - Goldfarba - Shanno



(a) Metoda Pearsona 1



(b) Metoda Pearsona 2



(c) Metoda McCormicka

Ilości wykonywanych iteracji zestawiono w tabeli:

| Metoda | Ilość iteracji |
|---|---|
| Davidona - Fletchera - Powella | 10 |
| Wolfe'a - Broydena - Davidona | 9 |
| Broydena - Fletchera - Goldfarba - Shanno | 10 |
| Pearsona 1 | 9 |
| Pearsona 2 | 9 |
| McCormicka | 9 |

Pomimo iż niektóre metody skończyły pracę przed wyczerpaniem limitu iteracji, to wyniki są nadal dość dalekie od rozwiązania optymalnego. Analogiczne informacje, jak w poprzednim zadaniu zaprezentowane są w tabelkach poniżej:

**Metoda Davidona - Fletchera - Powella**

| Numer iteracji | Odległość od minimum | Różnica wartości funkcji celu | Moduł gradientu |
|---|---|---|---|
| 1 | 2,82842712474619 | 484 | 1076,89182372233 |
| 2 | 1,56235577613490 | 23,8005713984897 | 1076,89182372233 |
| 3 | 1,32721629044590 | 0,783983815415468 | 118,077711229242 |
| 4 | 1,12666196279059 | 0,487107668801515 | 1,73108666754834 |
| 5 | 1,05755146213142 | 0,448651700842871 | 4,78837021386479 |
| 6 | 1,04977583212975 | 0,376196899280944 | 7,47900733956953 |
| 7 | 0,887350286678641 | 0,262702433953182 | 0,975740016772635 |
| 8 | 0,699091086115679 | 0,170893813659188 | 5,01388492407929 |
| 9 | 0,656781114089060 | 0,115883469916943 | 7,06902378321218 |
| 10 | 0,656337903730479 | 0,114293816475259 | 1,38484308644274 |
| 11 | 0,498694673904113 | 0,0719286135144581 | 0,425647297102391 |

Algorytm uzyskał dość dobry wynik, bardzo zbliżony do wyniku drugiego algorytmu Pearsona, ale w odróżnieniu od niego wykorzystał wszystkie możliwe iteracje (w tabelce znajduje się 11 wierszy, ponieważ uwzględniony jest również punkt początkowy). Algorytm Pearsona skończył o jedną iterację wcześniej.

**Metoda Wolfe'a - Broydena - Davidona**

| Numer iteracji | Odległość od minimum | Różnica wartości funkcji celu | Moduł gradientu |
|---|---|---|---|
| 1 | 2,82842712474619 | 484 | 1076,89182372233 |
| 2 | 1,56235577613490 | 23,8005713984897 | 1076,89182372233 |
| 3 | 1,32721631666988 | 0,783983811103343 | 118,077711229242 |
| 4 | 1,12664285029944 | 0,487082897335934 | 1,73108776187775 |
| 5 | 0,915120680964819 | 0,303798103847217 | 4,78826041952040 |
| 6 | 0,910871548608057 | 0,256184082833977 | 6,75446817586009 |
| 7 | 0,739342957682910 | 0,170518629410032 | 0,722651311859129 |
| 8 | 0,524848827318402 | 0,0930870522788360 | 4,85746936726612 |
| 9 | 0,523860366590557 | 0,0680718630856001 | 6,22269057804578 |
| 10 | 0,355533471769559 | 0,0357759380346189 | 0,292351230046323 |

**Metoda Broydena - Fletchera - Goldfarba - Shanno**

| Numer iteracji | Odległość od minimum | Różnica wartości funkcji celu | Moduł gradientu |
|---|---|---|---|
| 1 | 2,82842712474619 | 484 | 1076,89182372233 |
| 2 | 1,56235577613490 | 23,8005713984897 | 1076,89182372233 |
| 3 | 1,32719968168200 | 0,783986700429827 | 118,077711229242 |
| 4 | 1,13783224721679 | 0,502577723926761 | 1,73041611258292 |
| 5 | 0,934992024542521 | 0,322687038598867 | 4,91423816115789 |
| 6 | 0,930334486072714 | 0,270788032178845 | 6,95376064267518 |
| 7 | 0,759643171522079 | 0,181419758739935 | 0,753620254571319 |
| 8 | 0,550285671527806 | 0,102548550076324 | 4,87754533834599 |
| 9 | 0,526247085186151 | 0,0691505162739285 | 6,35754680257711 |
| 10 | 0,525883815094522 | 0,0686711451982404 | 0,794989633683640 |
| 11 | 0,371105208681140 | 0,0391804930552318 | 0,315935908843947 |

Powyższe algorytmy uzyskały dość zbliżone do siebie wyniki, jednocześnie są to dwa najlepsze wyniki (najbliżej rozwiązania optymalnego) spośród badanych algorytmów. Przy tym algorytm Wolfe'a-Broydena-Davidona jest o tyle lepszy, że wykonał o jedną iterację mniej.

**Metoda Pearsona 1**

| Numer iteracji | Odległość od minimum | Różnica wartości funkcji celu | Moduł gradientu |
|---|---|---|---|
| 1 | 2,82842712474619 | 484 | 1076,89182372233 |
| 2 | 1,56235577613490 | 23,8005713984897 | 1076,89182372233 |
| 3 | 1,32721575941531 | 0,783982706775743 | 118,077711229242 |
| 4 | 1,32721575940639 | 0,783982706756263 | 1,73108460838949 |
| 5 | 1,32721575939529 | 0,783982706735531 | 1,73108460836919 |
| 6 | 1,16389512192093 | 0,538428456418052 | 1,73108460828526 |
| 7 | 0,978417362102525 | 0,364986997248831 | 5,06441787656064 |
| 8 | 0,978417362039734 | 0,364986996964280 | 7,24225307048185 |
| 9 | 0,972793487004333 | 0,304981519552937 | 7,24225305604503 |
| 10 | 0,802861244851049 | 0,207343253854669 | 0,825953446513685 |

Algorytm uzyskał znacznie gorszy wynik niż pozostałe algorytmy. Skończył pracę o jedną iterację przed limitem.

**Metoda Pearsona 2**

| Numer iteracji | Odległość od minimum | Różnica wartości funkcji celu | Moduł gradientu |
|---|---|---|---|
| 1 | 2,82842712474619 | 484 | 1076,89182372233 |
| 2 | 1,56235577613490 | 23,8005713984897 | 1076,89182372233 |
| 3 | 1,32726454472140 | 0,784084563827515 | 118,077711229242 |
| 4 | 1,01410665001646 | 0,354045678838106 | 1,73127383758946 |
| 5 | 1,00961313835233 | 0,337575575545219 | 3,57302166841712 |
| 6 | 0,853117394435998 | 0,238097316520924 | 0,917202047893591 |
| 7 | 0,621932228549685 | 0,128906420590032 | 4,92158666347841 |
| 8 | 0,649278569570144 | 0,125084450502630 | 6,32158489105394 |
| 9 | 0,648013813403266 | 0,110920508312337 | 4,35539449880906 |
| 10 | 0,478781689589754 | 0,0657437983801014 | 0,400622197615117 |

**Metoda McCormicka**

| Numer iteracji | Odległość od minimum | Różnica wartości funkcji celu | Moduł gradientu |
|---|---|---|---|
| 1 | 2,82842712474619 | 484 | 1076,89182372233 |
| 2 | 1,56235577613490 | 23,8005713984897 | 1076,89182372233 |
| 3 | 1,36418816060472 | 0,867272253971135 | 118,077711229242 |
| 4 | 1,19986223863766 | 0,587748528378136 | 1,85394347159581 |
| 5 | 1,18892258180060 | 0,544299400249260 | 4,99848676049245 |
| 6 | 1,15559170383891 | 0,512212164679975 | 1,35036716190224 |
| 7 | 1,15459642348630 | 0,512179774064815 | 3,77912997223911 |
| 8 | 1,15459484125259 | 0,512179773970197 | 3,90369865321640 |
| 9 | 1,14638129540016 | 0,486729249379637 | 3,90389802638026 |
| 10 | 1,12199816063951 | 0,465682078997954 | 1,25097347741906 |

Ponownie najgorszy wynik uzyskał algorytm McCormicka. Wykonał jedną iterację mniej, niż dopuszczalny limit. Wynik jest znacznie niższej jakości niż w przypadku pozostałych algorytmów — znalezione rozwiązanie suboptymalne jest niecałe 3 razy dalej od najlepszego znalezionego rozwiązania przez algorytmu przy tej samej liczbie iteracji.

# 5 Wnioski

W trakcie laboratorium zbadano metody zmiennej metryki: metodę Davidona - Fletchera - Powella, metodę Wolfe'a - Broydena - Davidona, metodę Broydena - Fletchera - Goldfarba - Shanno, pierwszą metodę Pearsona, drugą metodę Pearsona i metodę McCormicka.

W pierwszym zadaniu zbadano działanie metod dla funkcjonału będącego formą kwadratową. W sytuacji, gdy zbiory poziomicowe funkcjonału celu były okręgami, wszystkie metody znajdowały minimum w pierwszej iteracji. W sytuacji,

gdy współczynnik $a$ był różny od jedynki i w konsekwencji zbiory poziomicowe były elipsami, algorytmy potrzebowały większej liczby iteracji. Metody: Davidona - Fletchera - Powella, Wolfe'a - Broydena - Davidona, Broydena - Fletchera - Goldfarba - Shanno i pierwsza metoda Pearsona potrzebowały trzech iteracji, podczas gdy druga metoda Pearsona do znalezienia minimum potrzebowała pięciu iteracji. Metoda McCormicka wyczerpała limit dziesięciu iteracji, a mimo to w trzech przypadkach nawet przez ogląd wykresu można było stwierdzić, że metoda była dość daleko od znalezienia minimum.

Lokalizacja punktu początkowego ponownie okazała się istotnym czynnikiem dla efektywności algorytmu. W przypadku form kwadratowych, których zbiory poziomicowe były elipsami, a nie okręgami, znajdowano rozwiązanie w pierwszej iteracji tylko wtedy, gdy punkt początkowy został zlokalizowany na którejś z osi elipsy. W przeciwnym wypadku do znalezienia minimum konieczna była większa liczba iteracji.

W czwartym i piątym zadaniu porównano efektywność algorytmów. Algorytmy wykonały zbliżoną liczbę iteracji — dziewięć albo dziesięć. Dziesieć iteracji jest limitem, więc w takiej sytuacji nie wiadomo, czy zadziałało kryterium stopu związane z maksymalną liczbą iteracji, czy z osiągnięciem minimum.

Ponownie kiepski efekt osiągnęła metoda McCormicka. Znalezione przez nią rozwiązanie suboptymalne w zadaniu piątym było niecałe 3 razy bardziej oddalone od rozwiązania optymalnego niż najlepsze znalezione innymi algorytmami rozwiązanie suboptymalne.

W piątym zadaniu dość dobrze sprawdziły się algorytmy Wolfe'a - Broydena - Davidona i Broydena - Fletchera - Goldfarba - Shanno. Rozwiązania suboptymalne znalezione przez te algorytmy były oddalone o około 0.36 od rozwiązania optymalnego — jest to najlepszy wynik spośród badanych algorytmów.

Nieco gorszy efekt dało zastosowanie algorytmów Davidona - Fletchera - Powella i drugiej metody Pearsona. Odległość od rozwiązania optymalnego wyniosła około 0.48. Kolejna była pierwsza metoda Pearsona z odległością około 0.8 od rozwiązania optymalnego.