Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Algorytmy Ewolucyjne

Sprawozdanie z projektu 1

Michał Kwarciński, Kacper Marchlewicz

Warszawa, 2023

Spis treści

1. Wstęp teoretyczny

1.1. Treść zadania

Znajdź minimum funkcji Rosenbrock'a („bananowej") bez ograniczeń:

$$f(x) = (1 - x + a)^2 + 100[y - b - (x - a)^2]^2 \tag{1.1}$$

Stałe a, b oraz punkty startowe zostały wzięte z załączonej tablicy z wiersza o numerze 1.

a	b	X1	Y1	X2	Y2	X3	Y3	X4	Y4
-1	-1,5	1	-0,5	0	-2,5	-2	-2,5	-2	-0,5

1.2. Zastosowane algorytmy

W projekcie zdecydowaliśmy się porównać:

— Metoda quasi-Newtona aproksymująca gradient
— Metoda quasi-Newtona z wprowadzonym gradientem
— Algorytm obszaru zaufania
— Algorytm Neldera-Meada

Metoda quasi-Newtona

Nazywana również metodą zmiennej metryki jest algorytmem znajdowania ekstremów lokalnych funkcji. Metody te bazują na metodzie Newtona, znajdują punkty stacjonarne funkcji. Hesjan minimalizowanej funkcji nie musi być obliczany - jest przybliżany przez analizowanie kolejnych wektorów gradientu. Obliczony przez nas gradient wynosi:

$$dx = 2x - 400(x - a)(-(x - a)^2 + y - b) \tag{1.2}$$

$$dy = 200(-(x - a)^2 - b + y) \tag{1.3}$$

Algorytm obszaru zaufania

W optymalizacji matematycznej region zaufania jest podzbiorem obszaru funkcji celu, który jest aproksymowany za pomocą funkcji modelu. Jeśli w regionie zaufania zostanie znaleziony adekwatny model funkcji celu, region jest rozszerzany, natomiast, jeśli przybliżenie jest słabe, region jest zmniejszany. Dopasowanie ocenia się porównując stosunek oczekiwanej poprawy z aproksymacji modelu z rzeczywistą poprawą obserwowaną w funkcji celu.

Algorytm Neldera-Meada

Zwany również sympleksową metodą spadku jest bezgradientową metodą minimalizacji bez ograniczeń n-wymiarowych funkcji, może być stosowana do funkcji nieróżniczkowalnych. Metoda sprawdza się dobrze nawet dla mocno nieliniowych funkcji, jednak wymaga sporych nakładów pracy numerycznej szczególnie przy dużej liczbie zmiennych decyzyjnych.

2. Prezentacja wyników

2.1. Ograniczenia algorytmów

W celu lepszego porównania wszystkich użytych algorytmów nadaliśmy im jedno, takie same ograniczenie: $TolFun = 1e^{-10}$. Oznacza to, że algorytm przerwie działanie gdy:

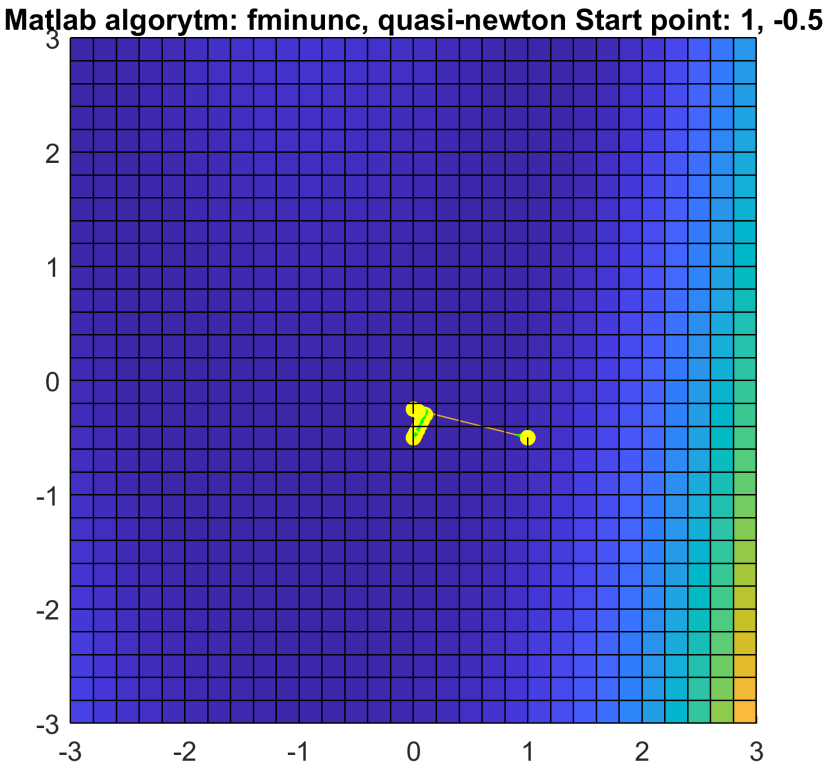
$$|f(x_i) – f(x_i + 1)| < TolFun.$$

2.2. Metoda quasi-Newtona aproksymująca gradient

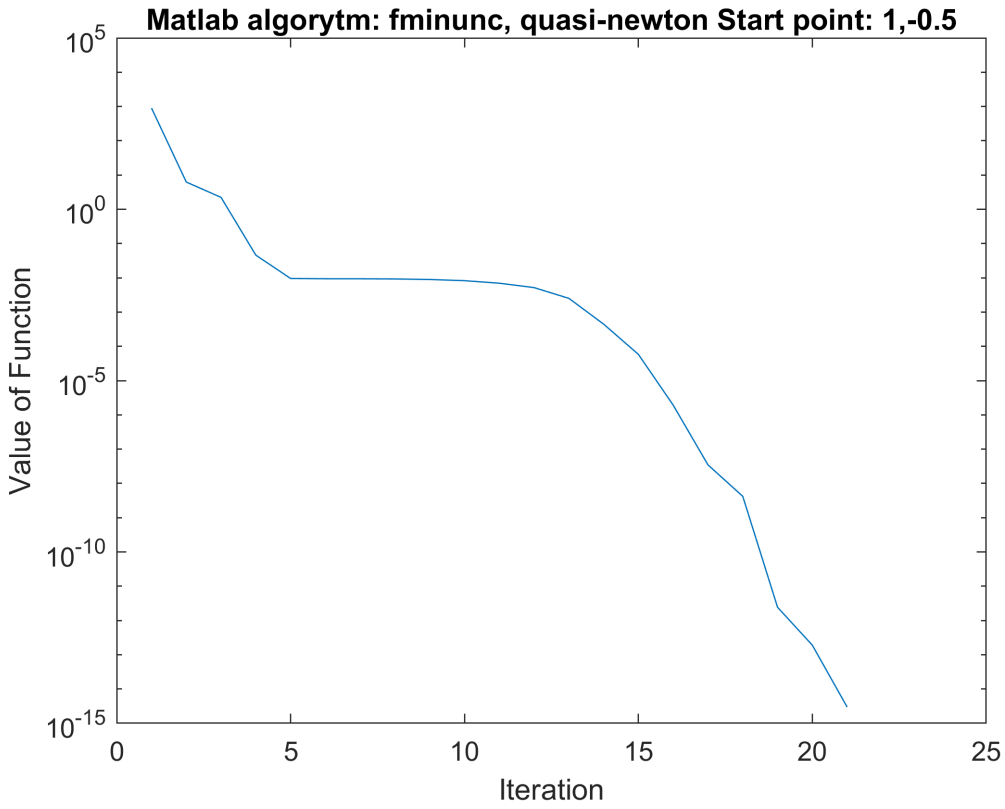
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(1, -0.5)	($1.8162e^{-08}$, -0.5)	$2.9801e^{-15}$	75	20

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.2.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.1.



Rys. 2.1: Wykres konturowy funkcji z wykorzystaniem metody quasi-Newtona aproksymującej gradient

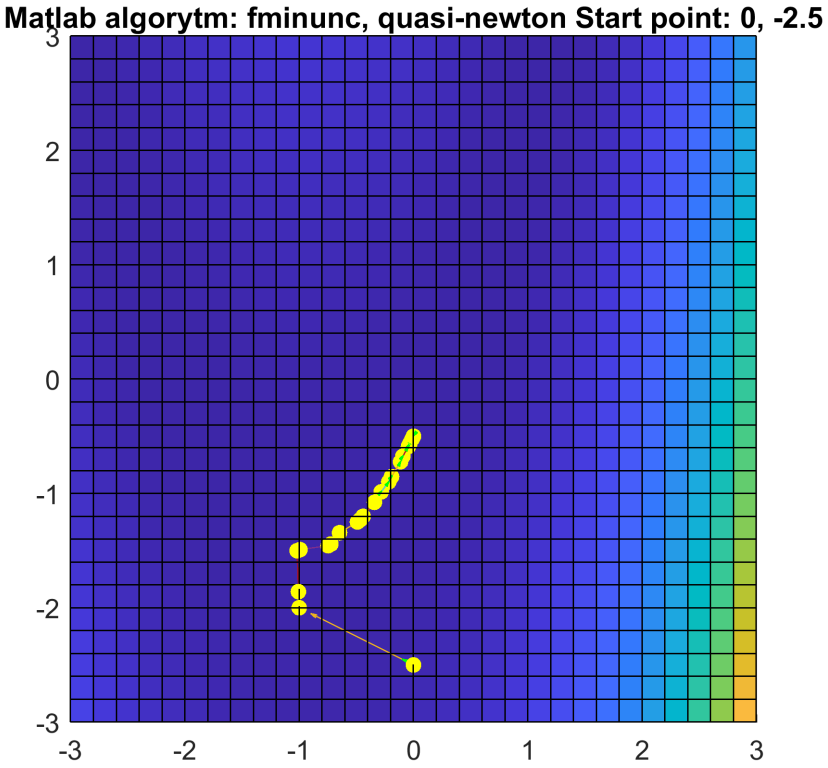


Rys. 2.2: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody quasi-Newtona aproksymującej gradient

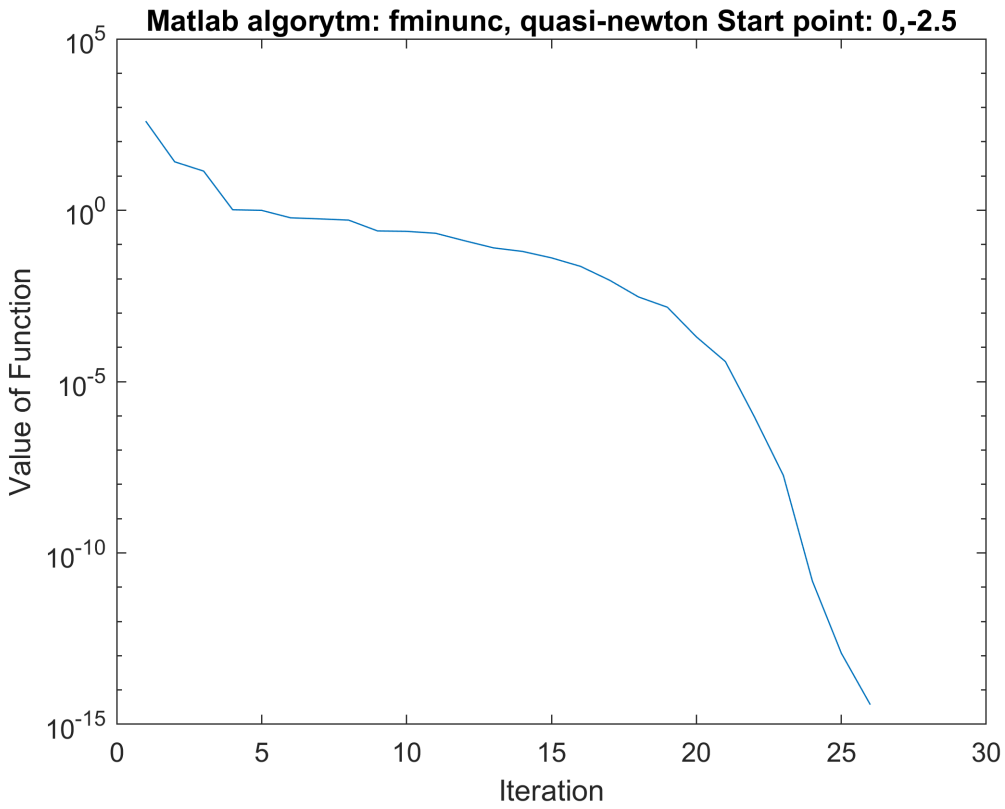
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(0, -2.5)	($6.1077e^{-08}$, -0.5)	$3.7308e^{-15}$	117	25

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.4 .

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.4 .



Rys. 2.3: Wykres konturowy funkcji z wykorzystaniem metody quasi-Newtona aproksymującej gradient

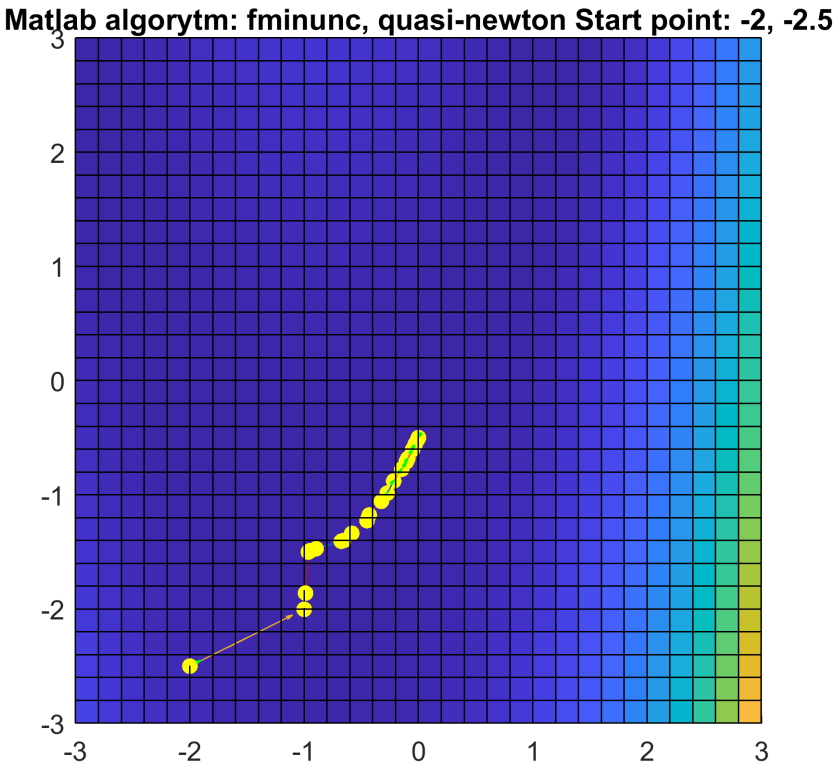


Rys. 2.4: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody quasi-Newtona aproksymującej gradient

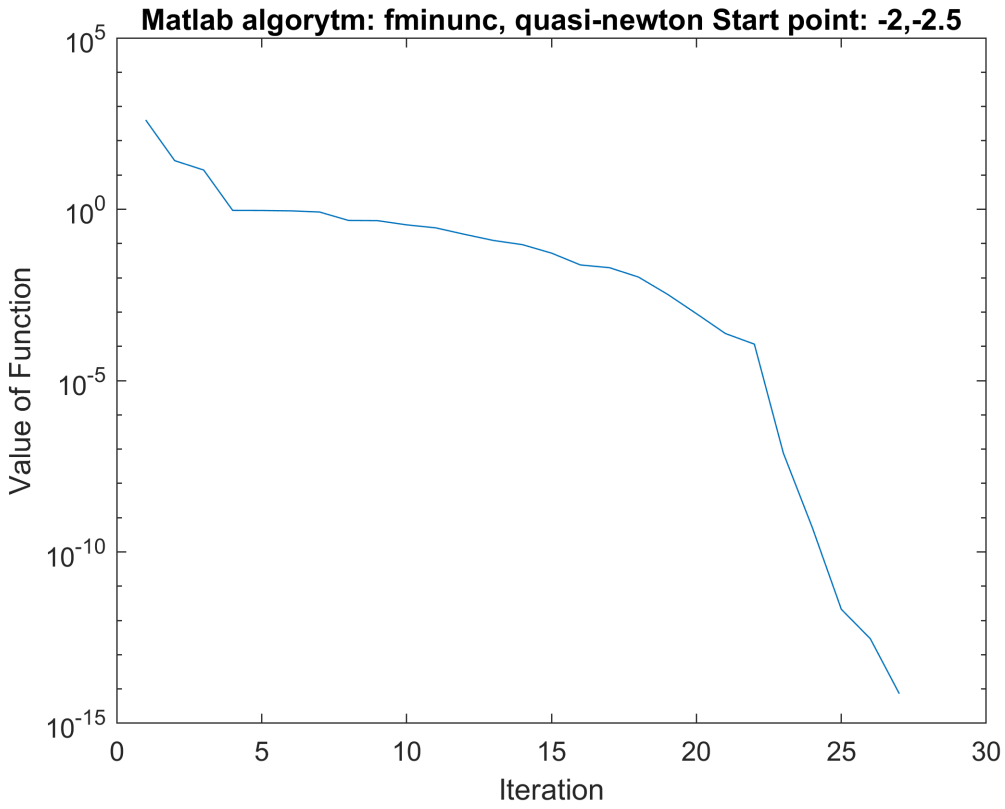
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(-2, -2.5)	$(-5.7714e^{-11}$, -0.5)	$7.2974e^{-15}$	120	26

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.5.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.6.



Rys. 2.5: Wykres konturowy funkcji z wykorzystaniem metody quasi-Newtona aproksymującej gradient

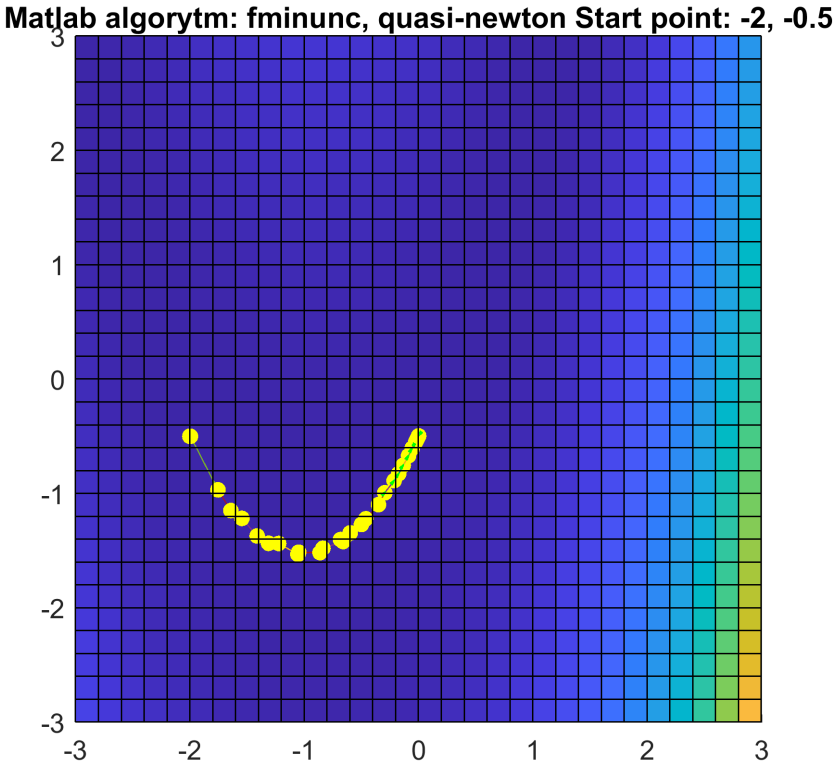


Rys. 2.6: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody quasi-Newtona aproksymującej gradient

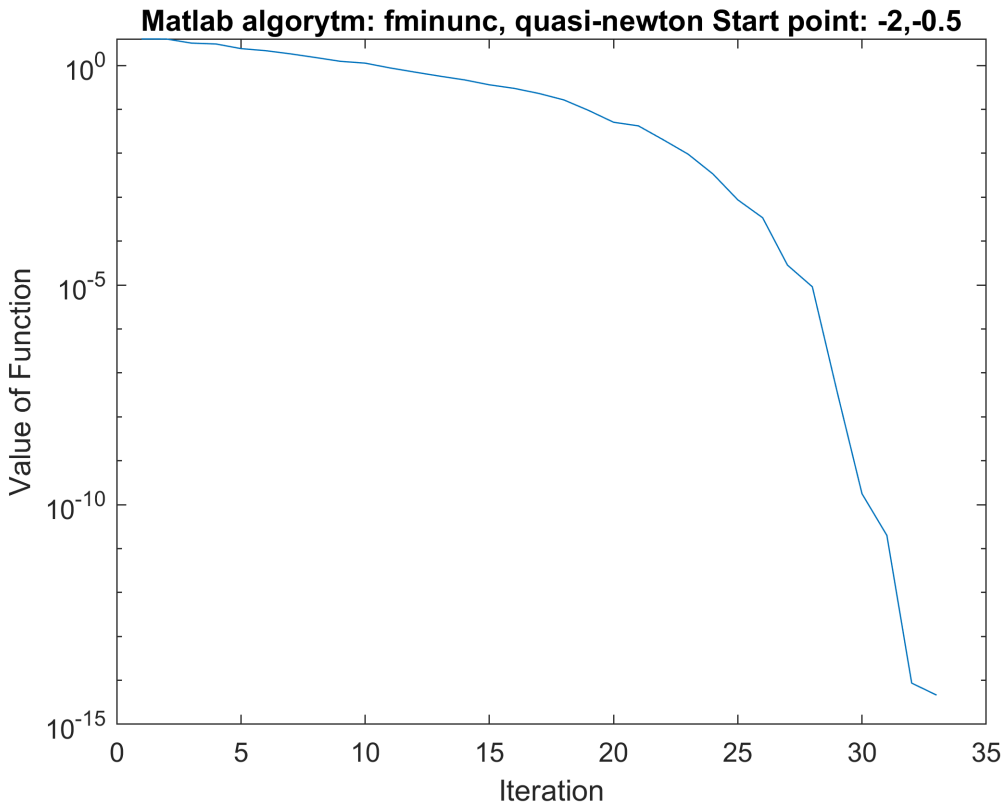
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(-2, -0.5)	($1.8112e^{-10}$, -0.5)	$4.6097e^{-15}$	174	32

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.7.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.8.



Rys. 2.7: Wykres konturowy funkcji z wykorzystaniem metody quasi-Newtona aproksymującej gradient



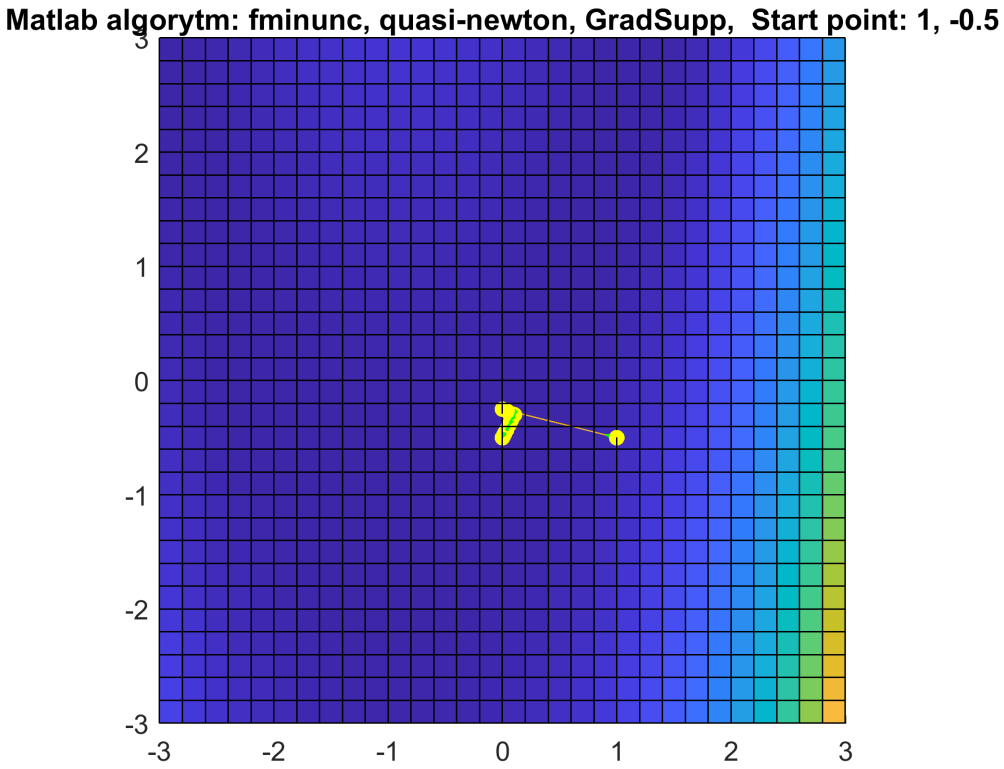
Rys. 2.8: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody quasi-Newtona aproksymującej gradient

2.3. Metoda quasi-Newtona z wprowadzonym gradientem

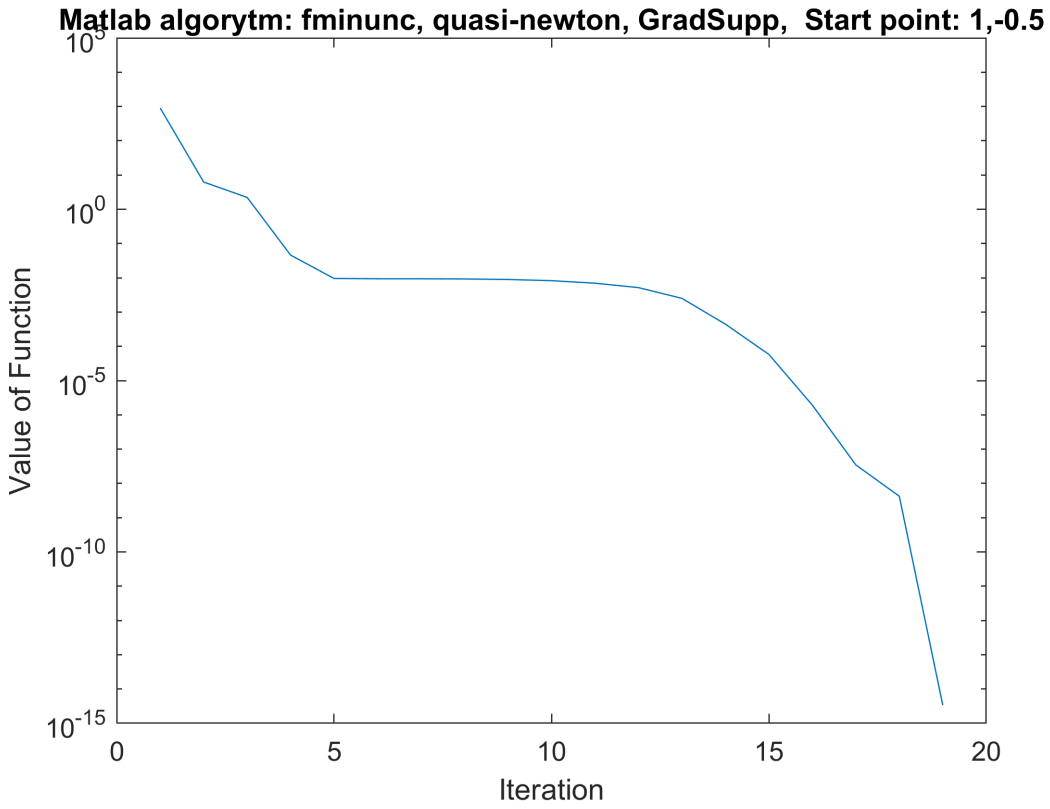
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(1, -0.5)	$(-5.824e^{-08}$, -0.5)	$3.3919e^{-15}$	20	18

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.10.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.9.



Rys. 2.9: Wykres konturowy funkcji z wykorzystaniem metody quasi-Newtona z wprowadzonym gradientem

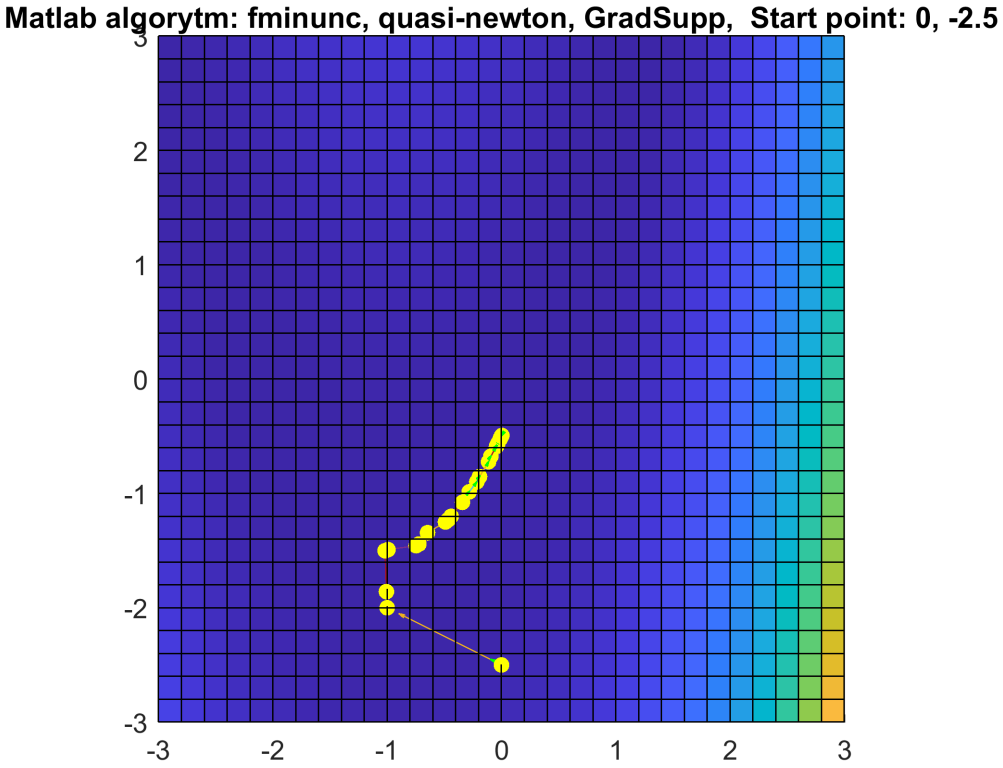


Rys. 2.10: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody quasi-Newtona z wprowadzonym gradientem

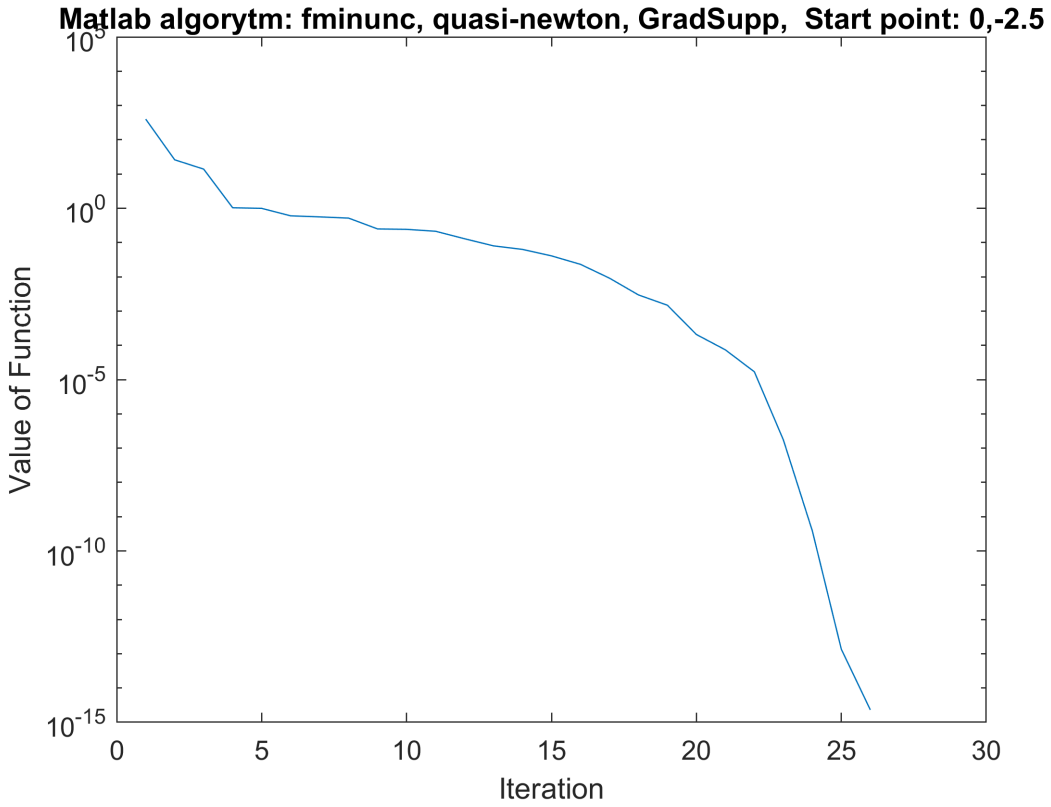
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(0, -2.5)	($3.4398e^{-08}$, -0.5)	$2.3107e^{-15}$	34	25

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.12.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.11.



Rys. 2.11: Wykres konturowy funkcji z wykorzystaniem metody quasi-Newtona z wprowadzonym gradientem

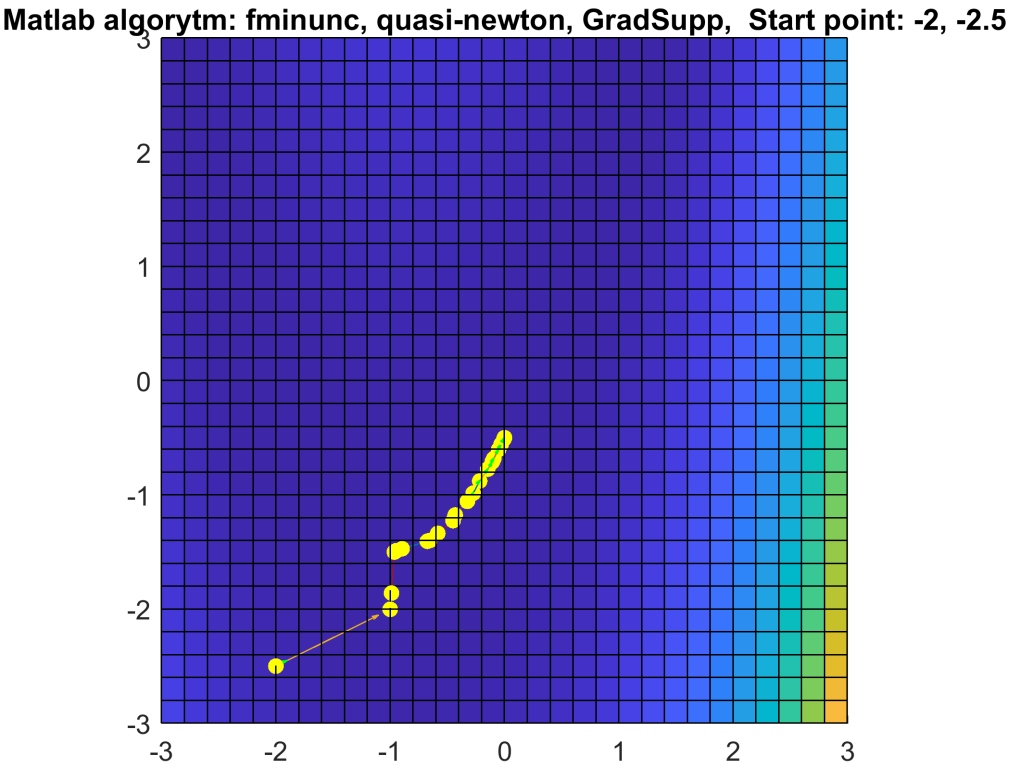


Rys. 2.12: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody quasi-Newtona z wprowadzonym gradientem

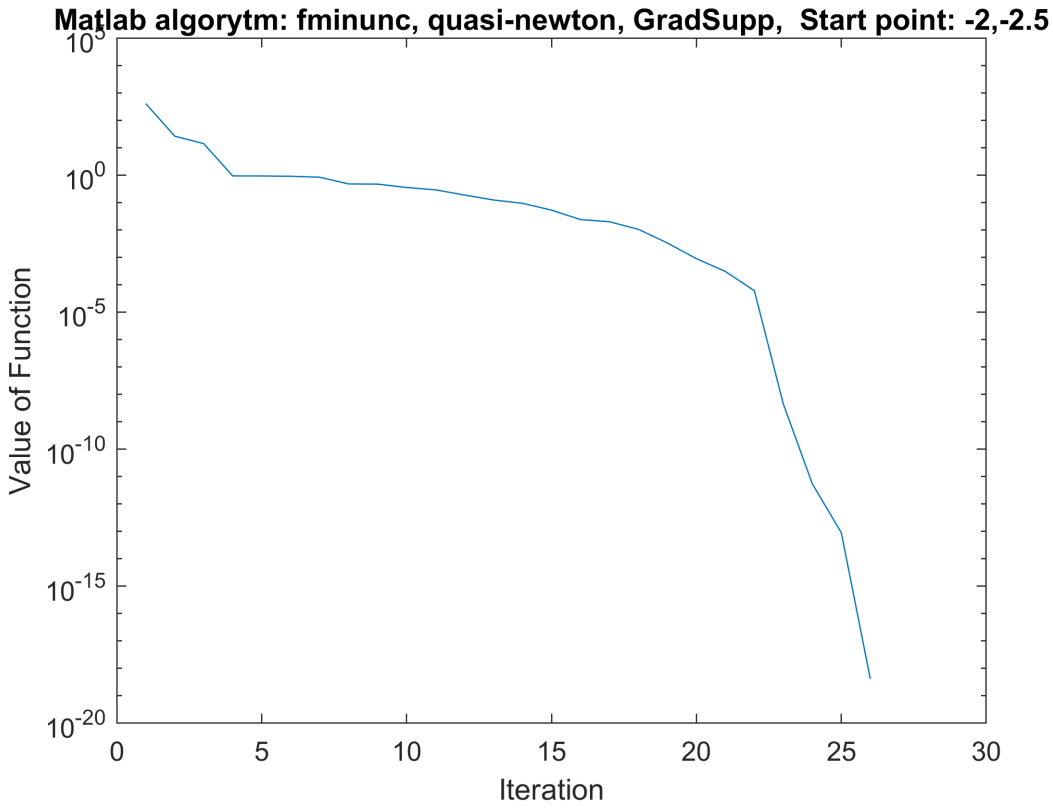
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(-2, -2.5)	$(-5.1199e^{-11}$, -0.5)	$4.1732e^{-19}$	31	25

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.14.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.13.



Rys. 2.13: Wykres konturowy funkcji z wykorzystaniem metody quasi-Newtona z wprowadzonym gradientem

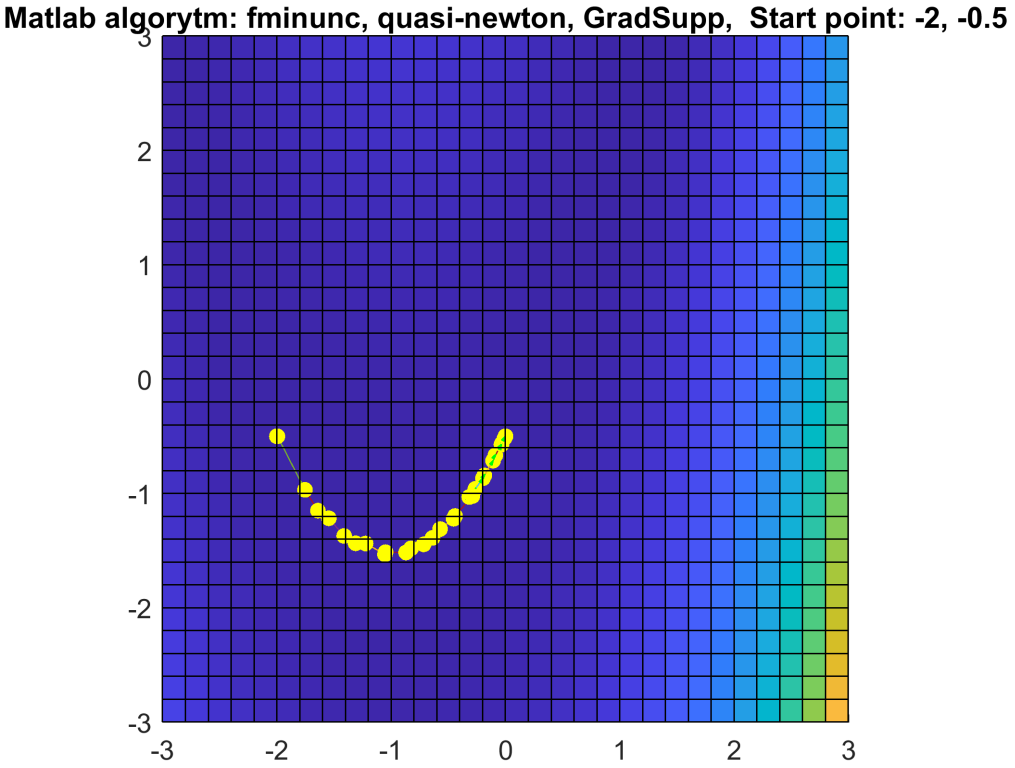


Rys. 2.14: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody quasi-Newtona z wprowadzonym gradientem

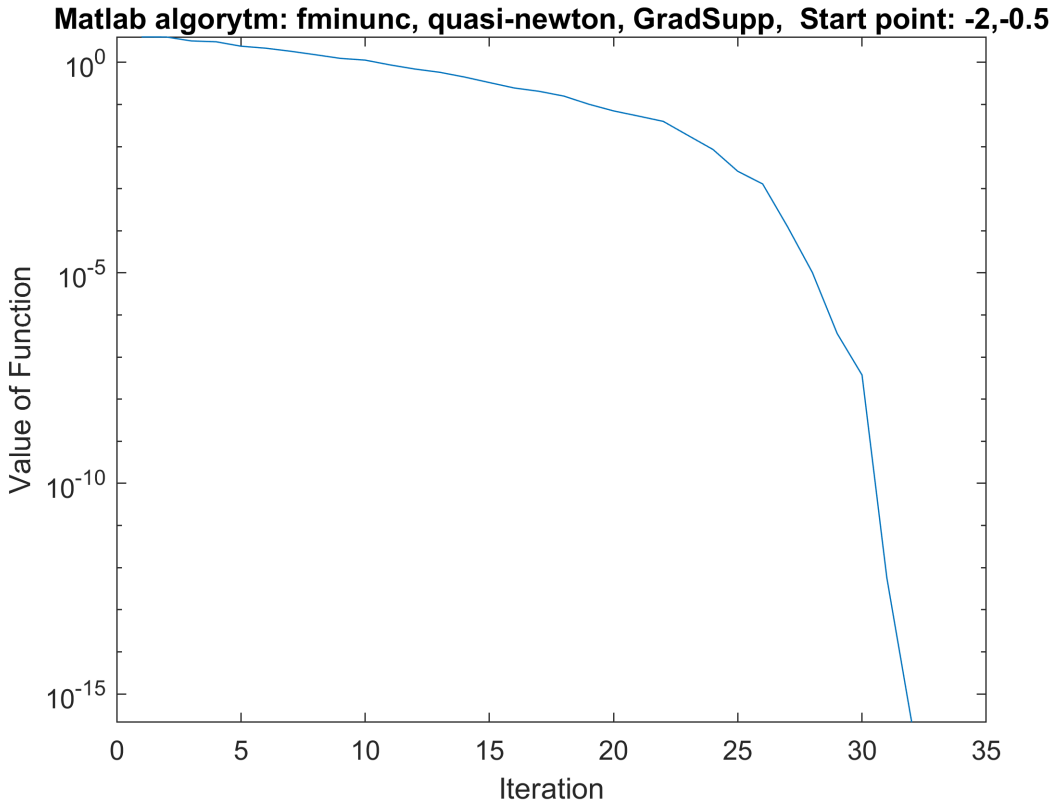
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(-2, -0.5)	$(-1.4062e^{-08}, -0.5)$	$2.1276e^{-16}$	44	31

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.16.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.15.



Rys. 2.15: Wykres konturowy funkcji z wykorzystaniem metody quasi-Newtona z wprowadzonym gradientem



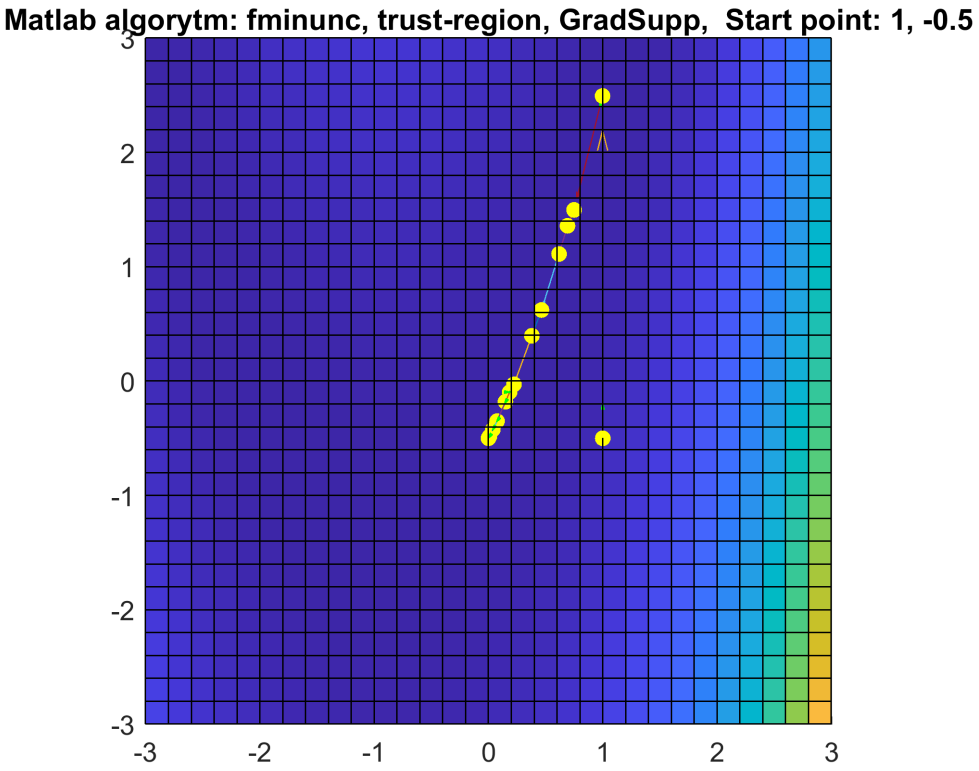
Rys. 2.16: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody quasi-Newtona z wprowadzonym gradientem

2.4. Obszar zaufania z gradientem aproksymowany przez solver

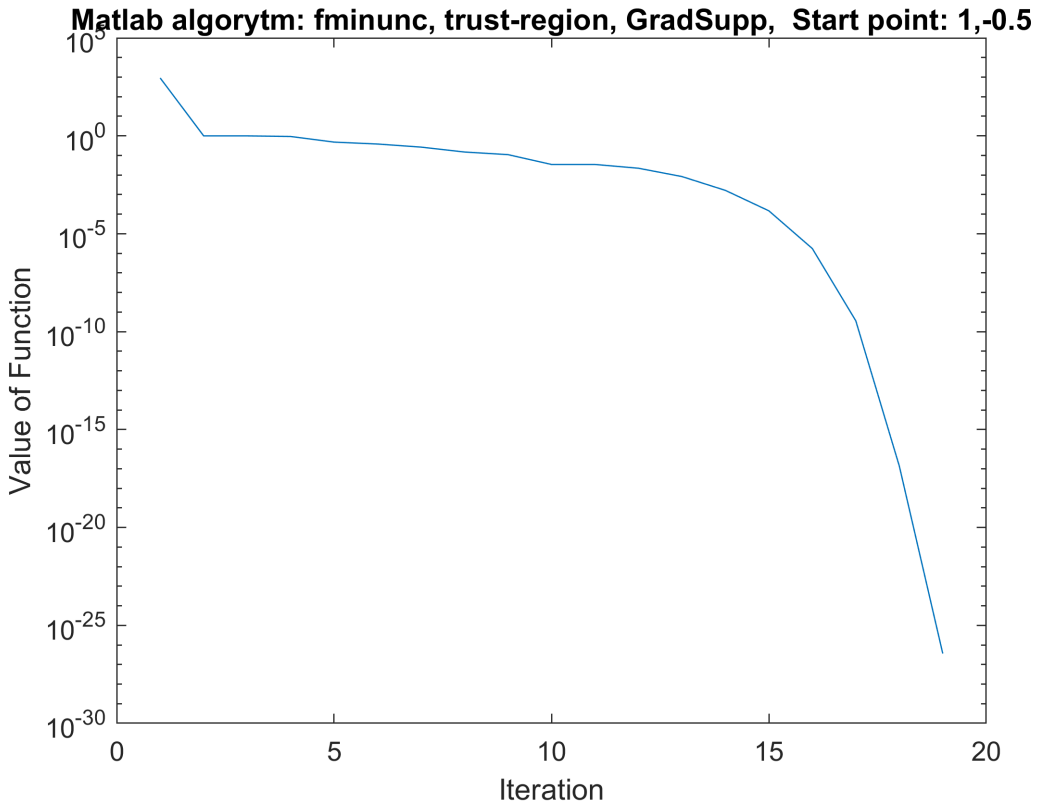
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(1, -0.5)	($2.1037e^{-14}$, -0.5)	$4.4523e^{-28}$	19	18

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.18.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.17.



Rys. 2.17: Wykres konturowy funkcji z wykorzystaniem metody obszaru zaufania

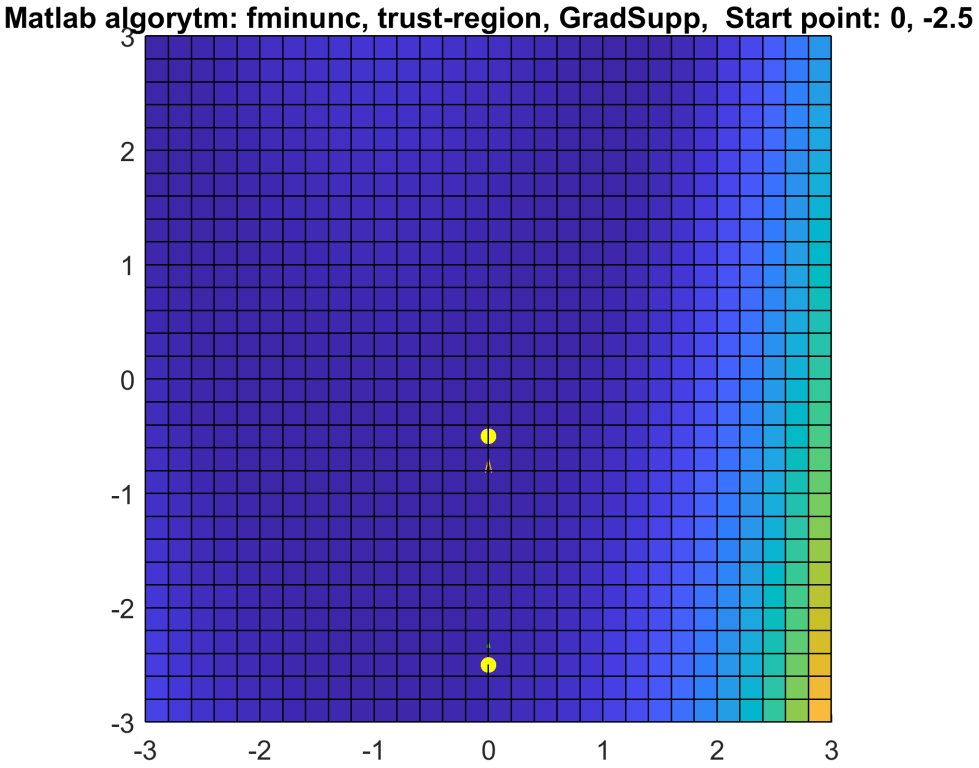


Rys. 2.18: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody obszaru zaufania

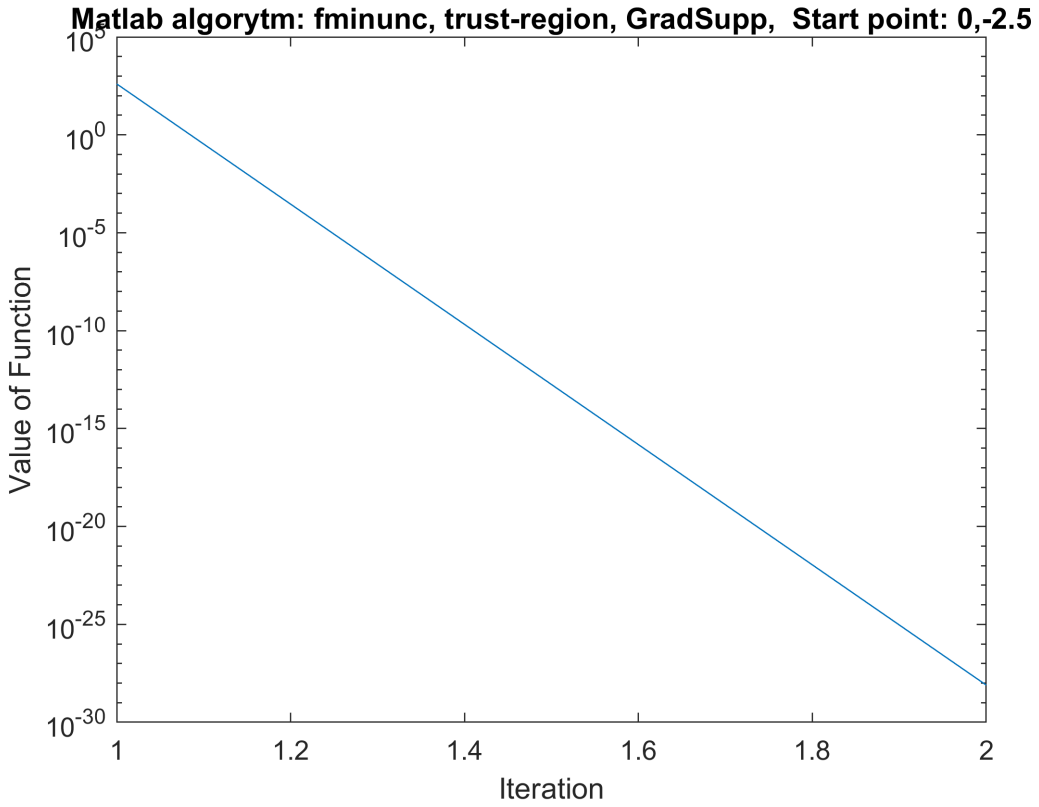
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(0, -2.5)	($3.3307e^{-16}$, -0.5)	$7.8997e^{-29}$	2	1

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.20.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.19.



Rys. 2.19: Wykres konturowy funkcji z wykorzystaniem metody obszaru zaufania

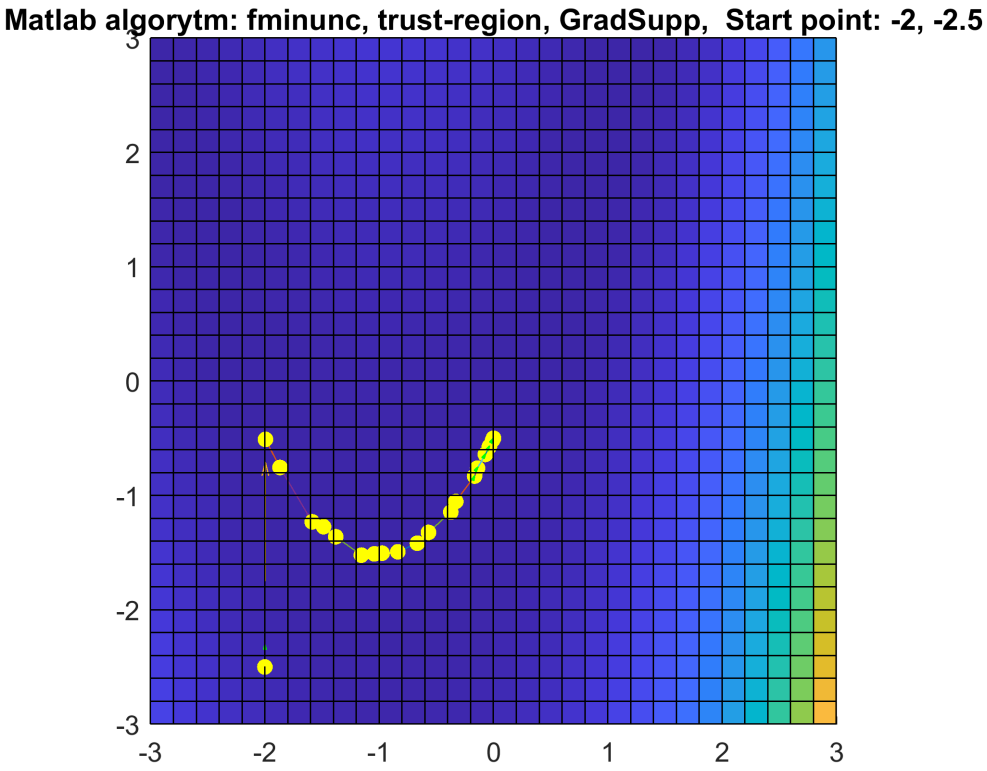


Rys. 2.20: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody obszaru zaufania

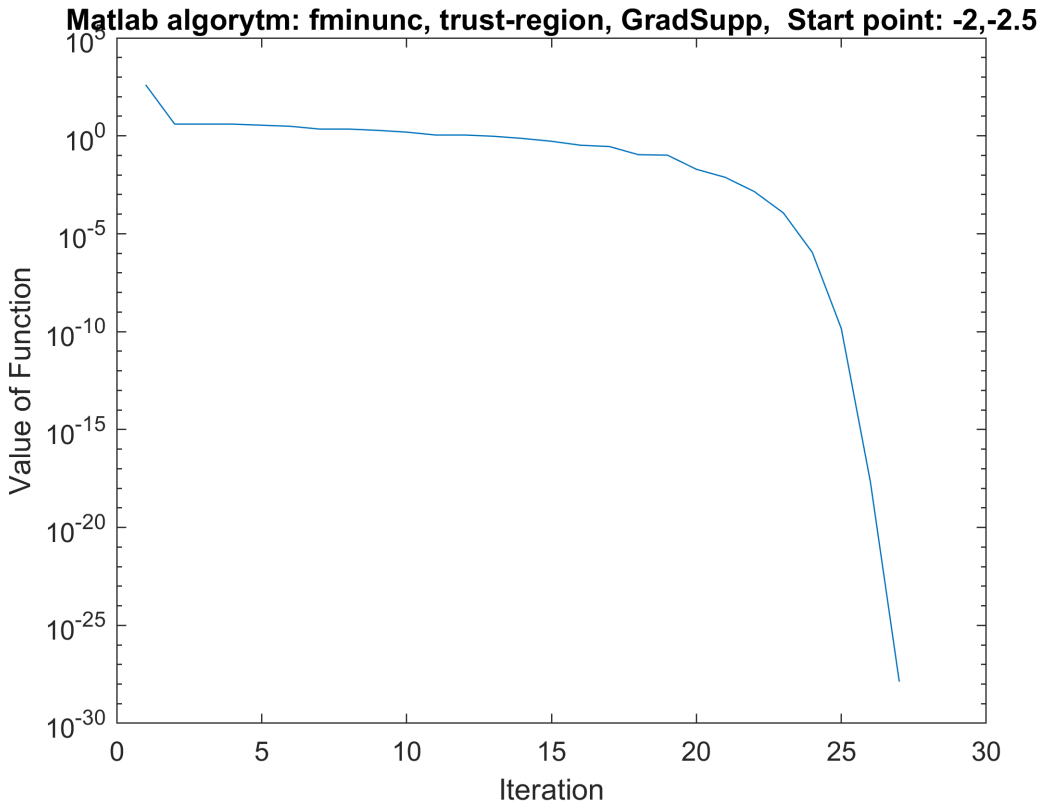
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(-2, -2.5)	$(1.3791e^{-14}$, -0.5)	$1.8952e^{-28}$	27	26

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.22.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.21.



Rys. 2.21: Wykres konturowy funkcji z wykorzystaniem metody obszaru zaufania

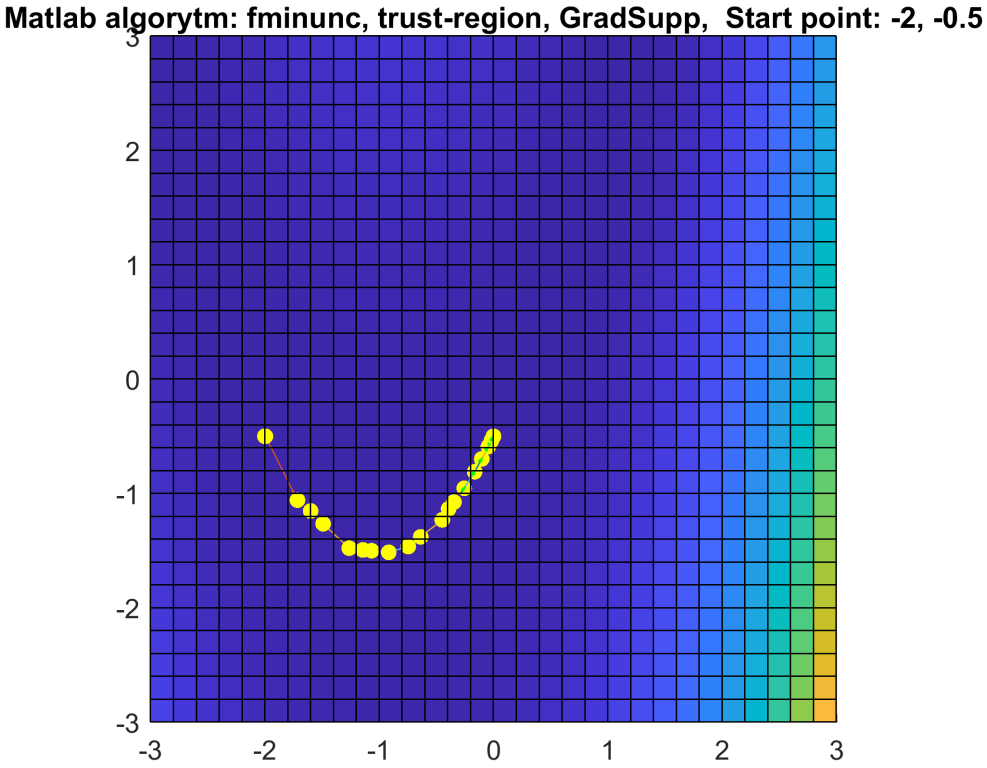


Rys. 2.22: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody obszaru zaufania

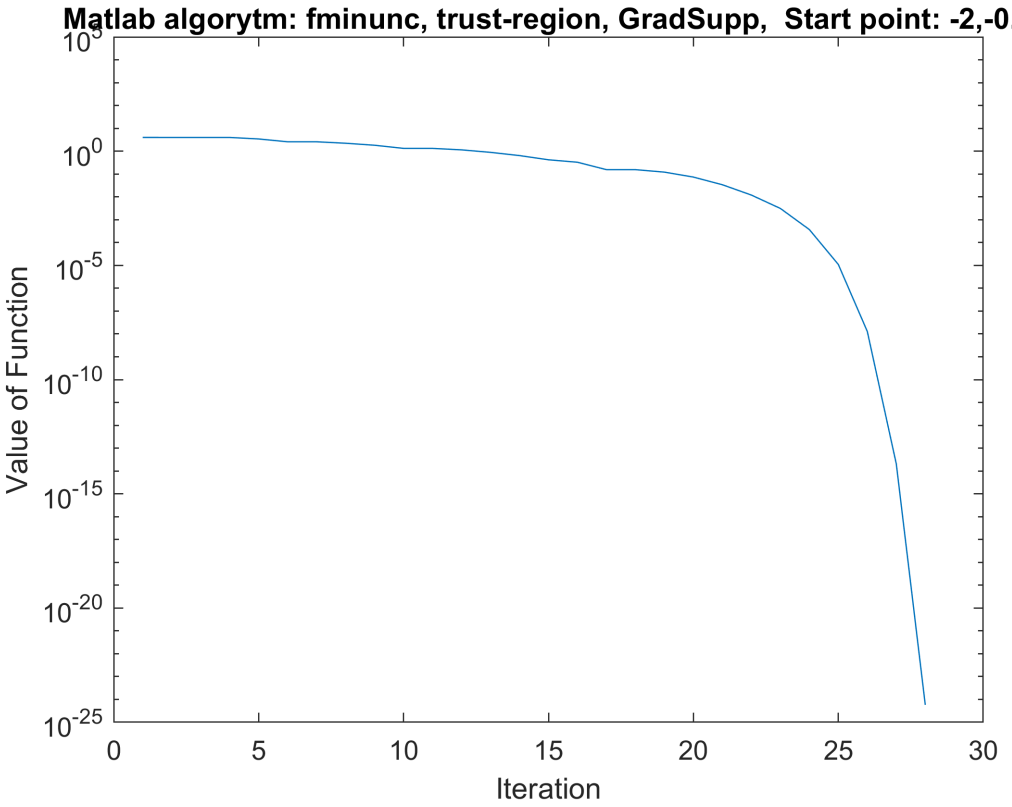
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(-2, -0.5)	$(-6.3511e^{-13}, -0.5)$	$4.0968e^{-25}$	28	27

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.24.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.23.



Rys. 2.23: Wykres konturowy funkcji z wykorzystaniem metody obszaru zaufania



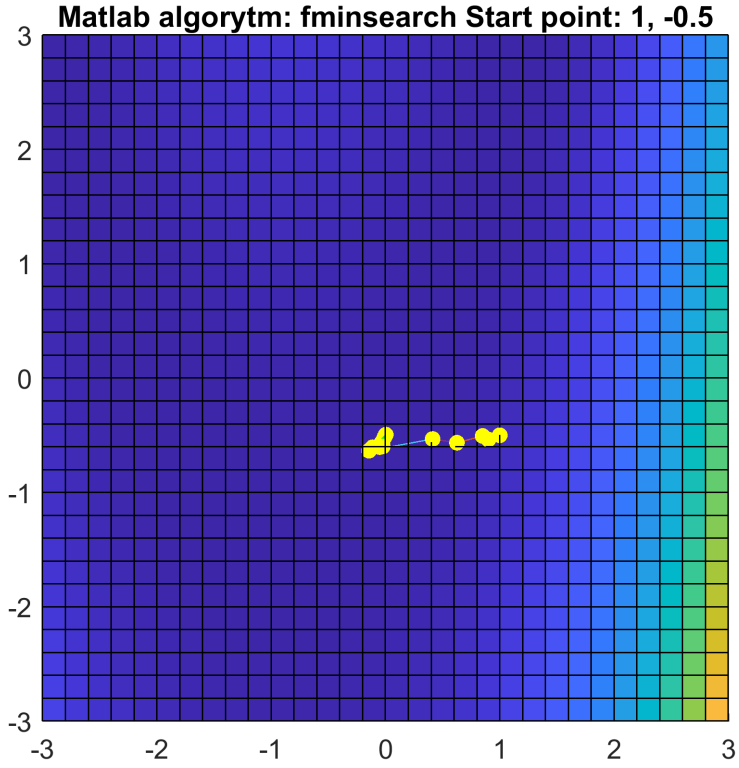
Rys. 2.24: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody obszaru zaufania

2.5. Algorytm Neldera-Meada

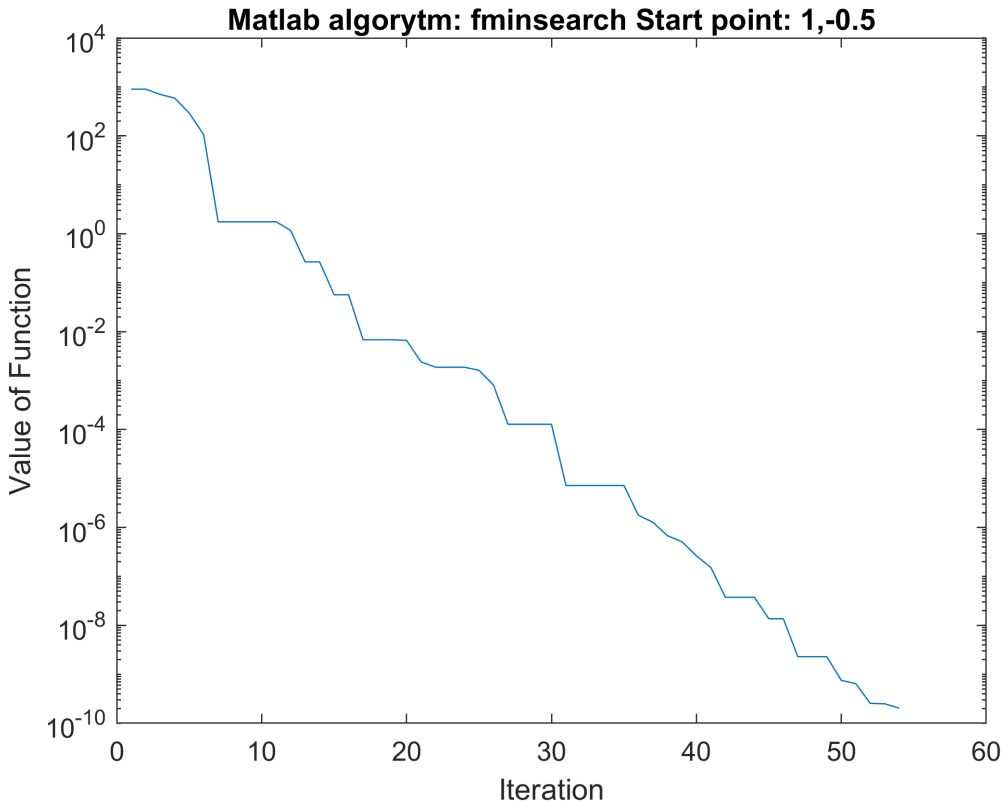
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(1, -0.5)	$(1.428e^{-05}$, -0.49997)	$2.0399e^{-10}$	100	53

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.26.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.25.



Rys. 2.25: Wykres konturowy funkcji z wykorzystaniem metody Neldera-Meada

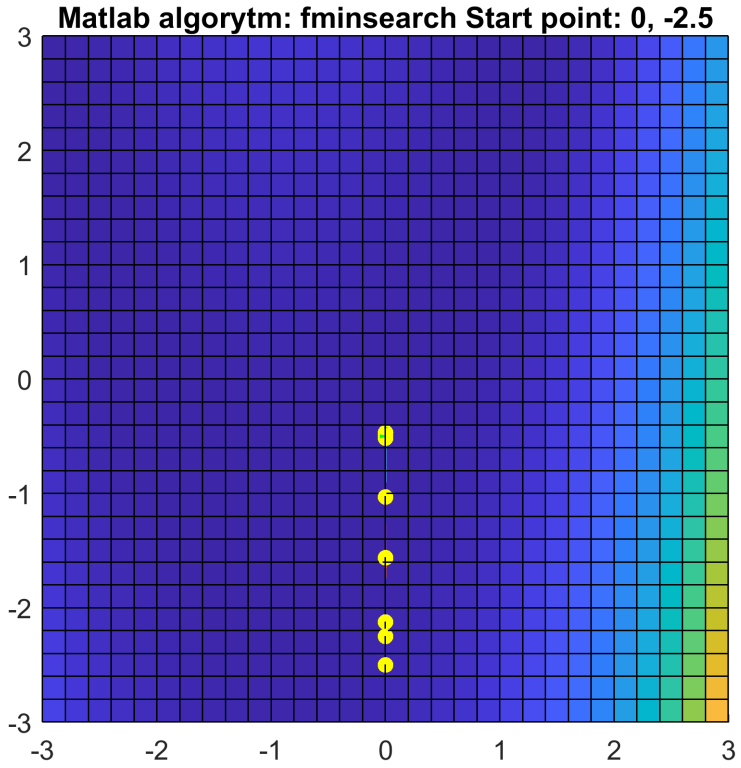


Rys. 2.26: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody Neldera-Meada

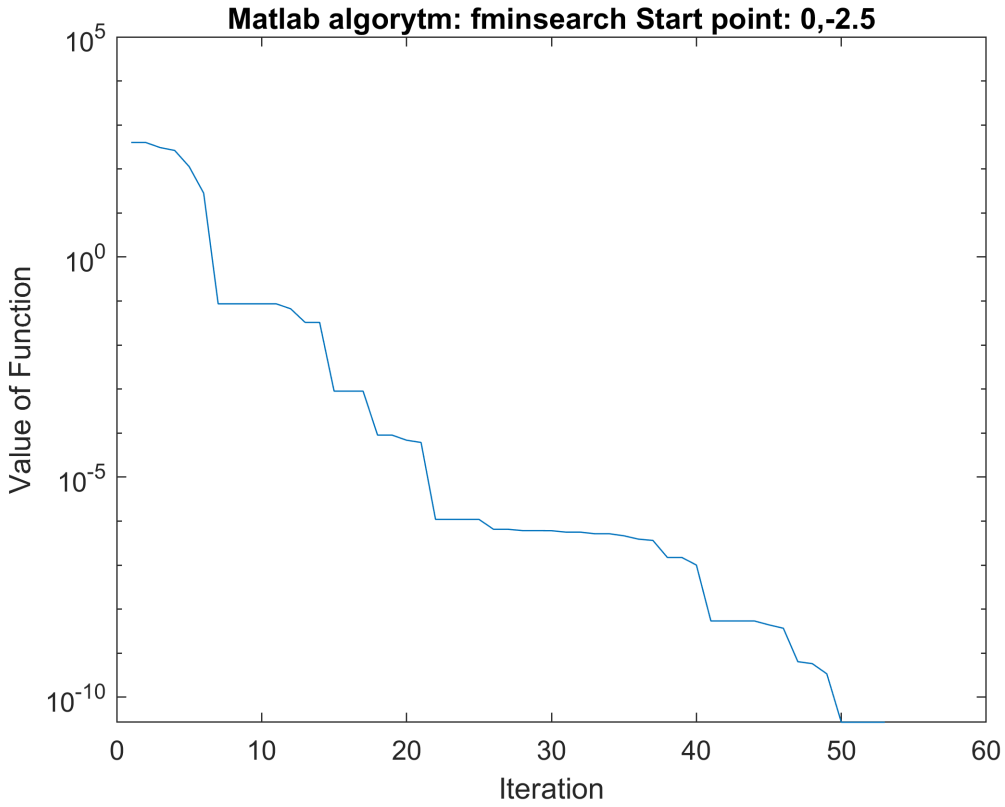
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(0, -2.5)	($2.5916e^{-06}$, -0.5)	$2.6934e^{-11}$	98	52

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.28.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.27.



Rys. 2.27: Wykres konturowy funkcji z wykorzystaniem metody Neldera-Meada

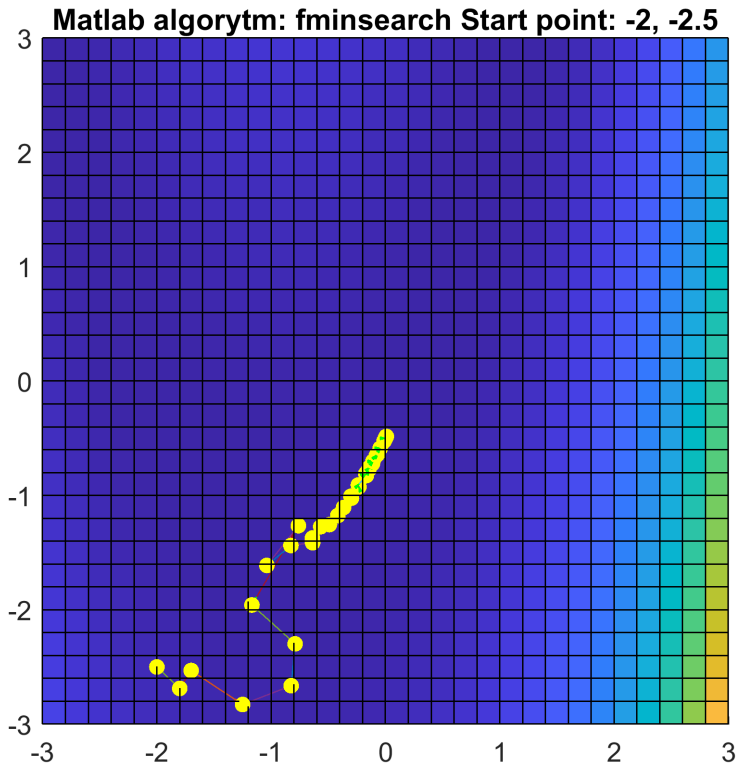


Rys. 2.28: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody Neldera-Meada

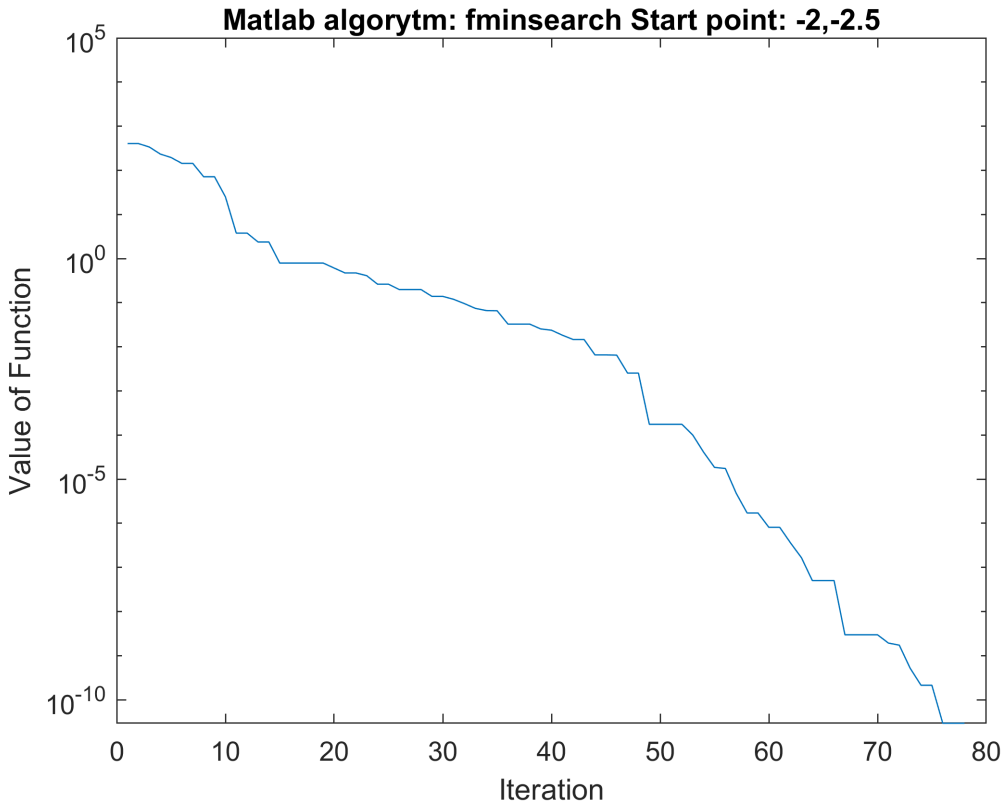
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(-2, -2.5)	$(-4.544e^{-06}$, -0.50001)	$2.9339e^{-11}$	146	77

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.30.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.29.



Rys. 2.29: Wykres konturowy funkcji z wykorzystaniem metody Neldera-Meada

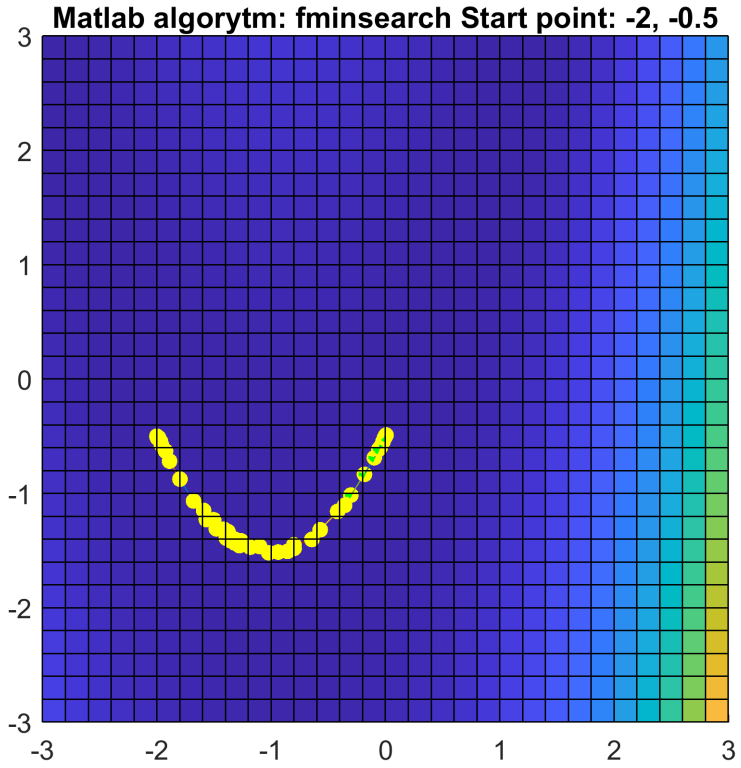


Rys. 2.30: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody Neldera-Meada

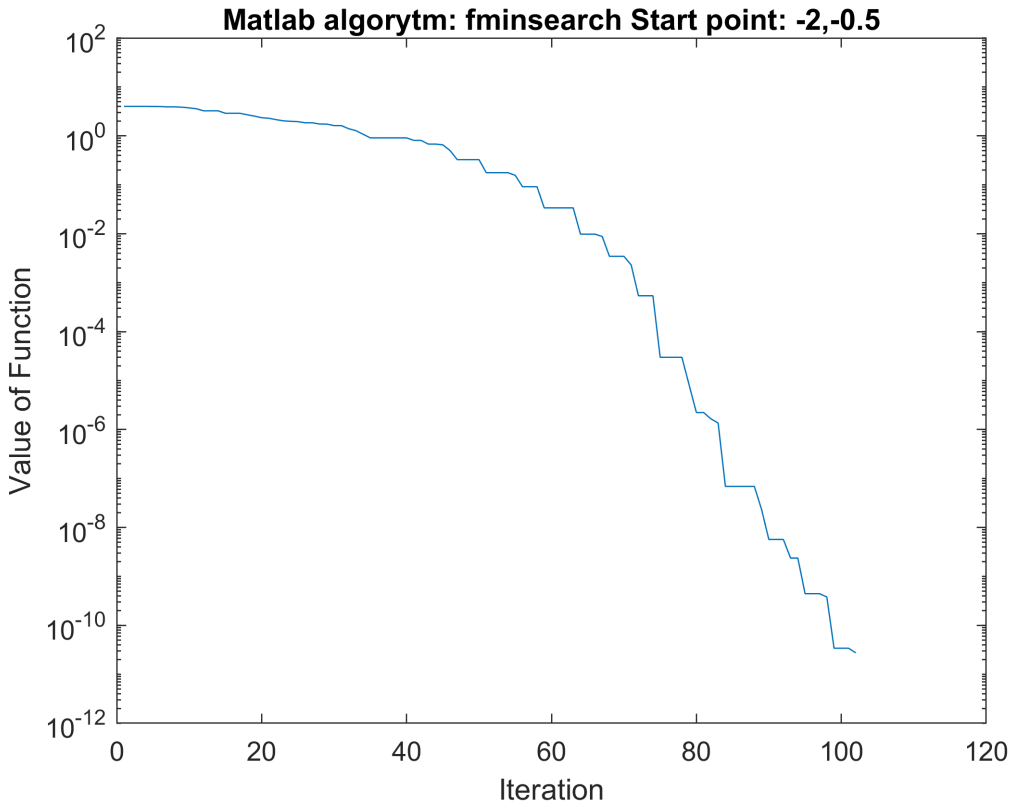
p. startowy	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
(-2, -0.5)	($4.4202e^{-06}$, -0.49999)	$2.7413e^{-11}$	188	101

Wykres wartości funkcji celu (w formie logarytmicznej) w funkcji numeru iteracji znajduje się na rysunku 2.32.

Wykres konturowy funkcji z naniesionymi trajektoriami znajduje się na rysunku 2.31.



Rys. 2.31: Wykres konturowy funkcji z wykorzystaniem metody Neldera-Meada



Rys. 2.32: Wykres wartości funkcji w formie logarytmicznej z wykorzystaniem metody Neldera-Meada

3. Podsumowanie

3.1. Porównanie i ocena działania metod

Dla punktu startowego: (1, -0.5)

algorytm	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
q-n aproks.	($1.8162e^{-08}$, -0.5)	$2.9801e^{-15}$	75	20
q-n grad	($-5.824e^{-08}$, -0.5)	$3.3919e^{-15}$	20	18
ob. zaufania	($2.1037e^{-14}$, -0.5)	$4.4523e^{-28}$	19	18
n-m	($1.428e^{-05}$, -0.49997)	$2.0399e^{-10}$	100	53

Dla punktu startowego: (0, -2.5)

algorytm	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
q-n aproks.	($6.1077e^{-08}$, -0.5)	$3.7308e^{-15}$	117	25
q-n grad	($3.4398e^{-08}$, -0.5)	$2.3107e^{-15}$	34	25
ob. zaufania	($3.3307e^{-16}$, -0.5)	$7.8997e^{-29}$	2	1
n-m	($2.5916e^{-06}$, -0.5)	$2.6934e^{-11}$	98	52

Dla punktu startowego: (-2, -2.5)

algorytm	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
q-n aproks.	($-5.7714e^{-11}$, -0.5)	$7.2974e^{-15}$	120	26
q-n grad	($-5.1199e^{-11}$, -0.5)	$4.1732e^{-19}$	31	25
ob. zaufania	($1.3791e^{-14}$, -0.5)	$1.8952e^{-28}$	27	26
n-m	($-4.544e^{-06}$, -0.50001)	$2.9339e^{-11}$	146	77

Dla punktu startowego: (-2, -0.5)

algorytm	p. końcowy	wartość k. wyniku	l. obliczeń funkcji celu	l. iteracji
q-n aproks.	($1.8112e^{-10}$, -0.5)	$4.6097e^{-15}$	174	32
q-n grad	($-1.4062e^{-08}$, -0.5)	$2.1276e^{-16}$	44	31
ob. zaufania	($-6.3511e^{-13}$, -0.5)	$4.0968e^{-25}$	28	27
n-m	($4.4202e^{-06}$, -0.49999)	$2.7413e^{-11}$	188	101

Licząc minimum ręcznie 1.2 i 1.3 otrzymaliśmy punkt (0, -0.5). Jest to zgodne z uzyskanymi wynikami otrzymanymi z testowanych algorytmów.

Można zauważyć, że metoda obszaru zaufania daje najdokładniejsze wyniki w najkrótszej liczbie iteracji. Dzieje się tak, gdyż algorytm aproksymując funkcję celu stara się przewidywać

odległość estymaty od optimum. Odpowiednie ograniczenia kroków zapobiegają nadmiernemu przesuwaniu się w przestrzeni.

Można zauważyć, że wersja metody quasi-Newtonowskiej, której podajemy gradient generuje podobny wynik końcowy, potrzebuje do tego niewiele więcej iteracji ale dużo więcej obliczeń funkcji celu (około 4 razy więcej). Jest to prawdopodobnie spowodowane potrzebą aproksymacji gradientu w każdej iteracji tz. w każdej iteracji wybierane są dodatkowe punkty, znajdującej się w otoczeniu punktu głównego, służące do aproksymacji gradientu.

Metoda Neldera-Meada sprawdziła się najgorzej ze wszystkich algorytmów. Wyniki były w porównaniu do reszty obarczone niższą dokładnością, która została osiągnięta w największej liczbie iteracji. Wynika to z korzystania jedynie z wartości funkcji celu - metoda ta nie oblicza bezpośrednio, ani nie estymuje gradientu funkcji celu. Funkcje na wykresach 2.26, 2.28 i 2.30 (z pominięciem 2.20 - przy dwóch iteracjach ciężko próbować cokolwiek wnioskować z wykresu) można przybliżyć do prostej, algorytm powoli, krok po kroku dochodzi do rozwiązania.