Algorytmy Ewolucyjne
Projekt 1 - sprawozdanie

Tomasz Indeka, 293457

1. Wstęp

Zadanie polega na przetestowaniu dostępnych w MATLABie funkcji optymalizacji gładkiej bez ograniczeń. Do tego zadania wybrałem następujące funkcje:

- *fminunc* z algorytmem Quasi-Newtonowskim
- *fminunc* z algorytmem Quasi-Newtonowskim i załączonym gradientem
- *fminunc* z algorytmem rejonu zaufania i załączonym gradientem
- *fminsearch*

Poszukiwanie minimum odbywało się z kilku różnych ustalonych punktów:

- $x = 1,\ y = 0,$
- $x = 0,\ y = -2,$
- $x = -2,\ y = -2,$
- $x = -2,\ y = 0.$

1.1. Testowana funkcja

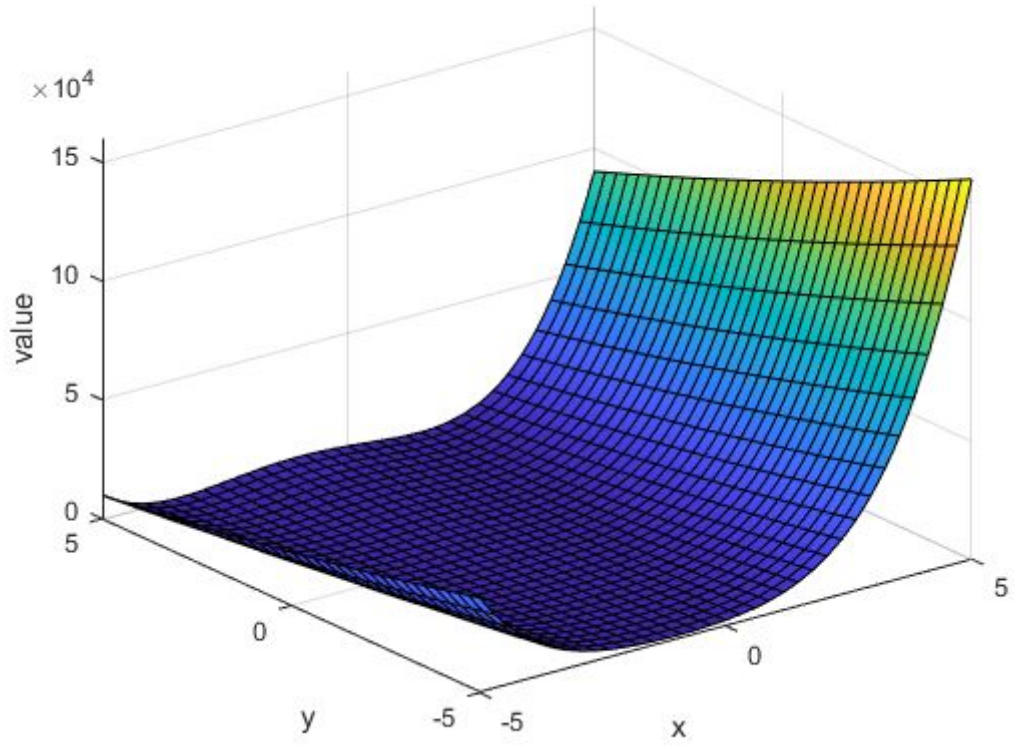
Funkcja Rosenbrocka:

$$f(x,y) = (1 - x + a)^2 + 100(y - b - (x - a)^2)^2,$$

gdzie:

$$a = -1,$$
$$b = -1.$$

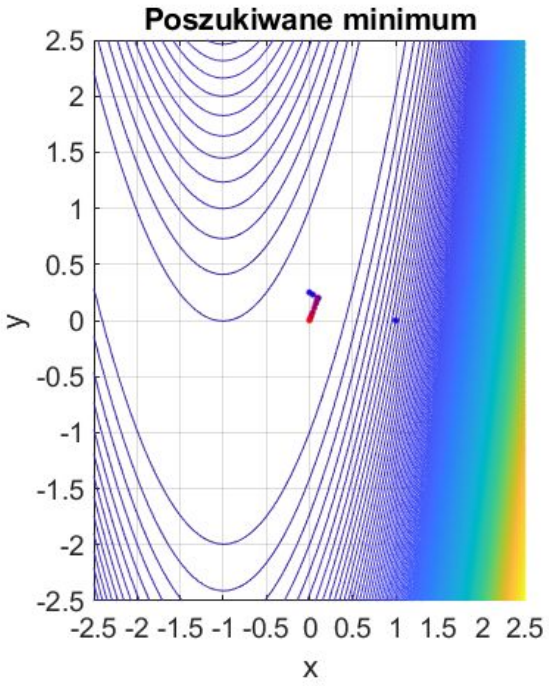
Testowana funkcja prezentuje się następująco:

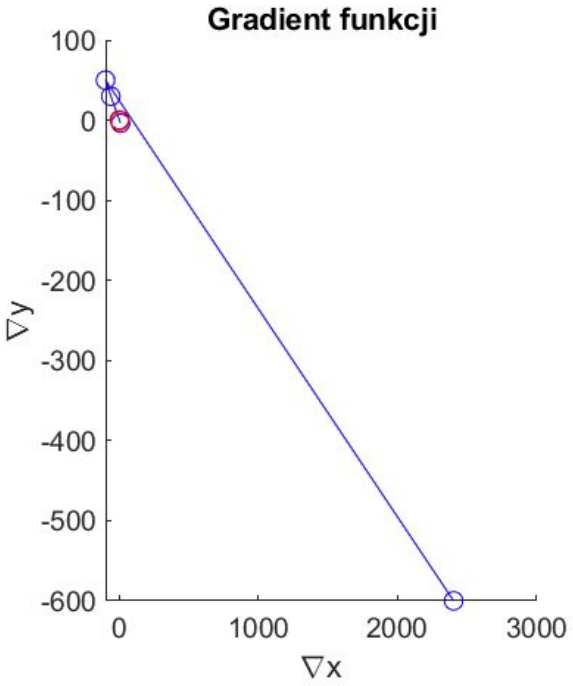


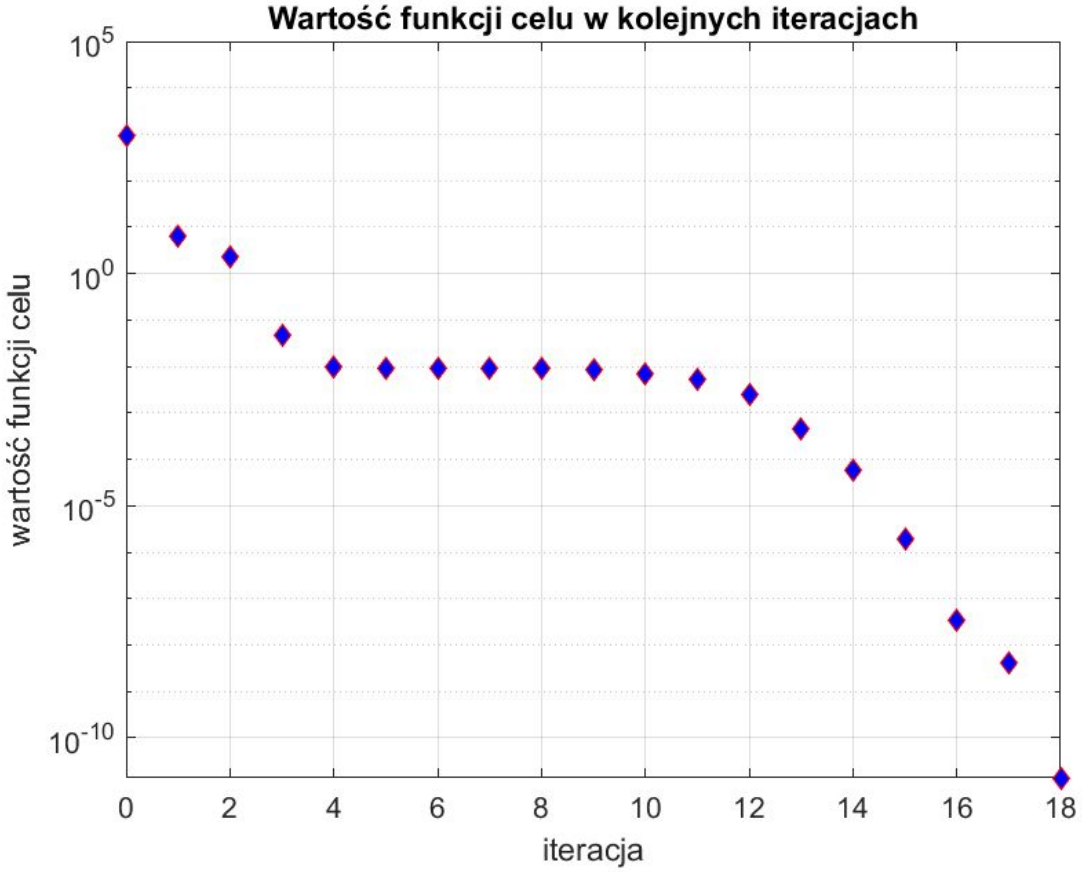
2. Wyniki

2.1. Metoda Quasi-Newtonowska

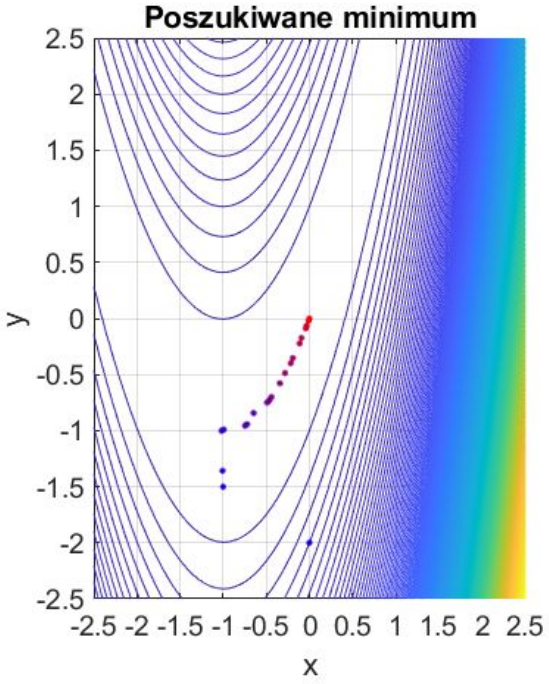
Punkt początkowy: x=1.000000, y=0.000000
Znalezione optimum lokalne: x=-0.000004, y=-0.000007
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 18 Ilość obliczeń funkcji celu: 60

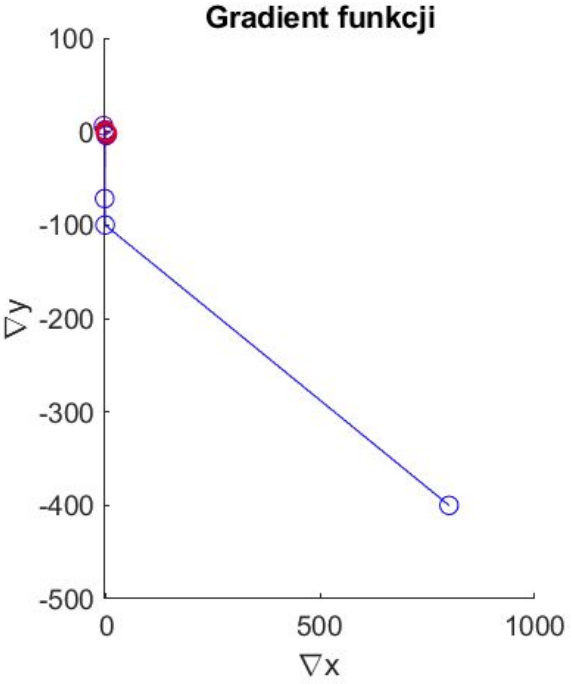


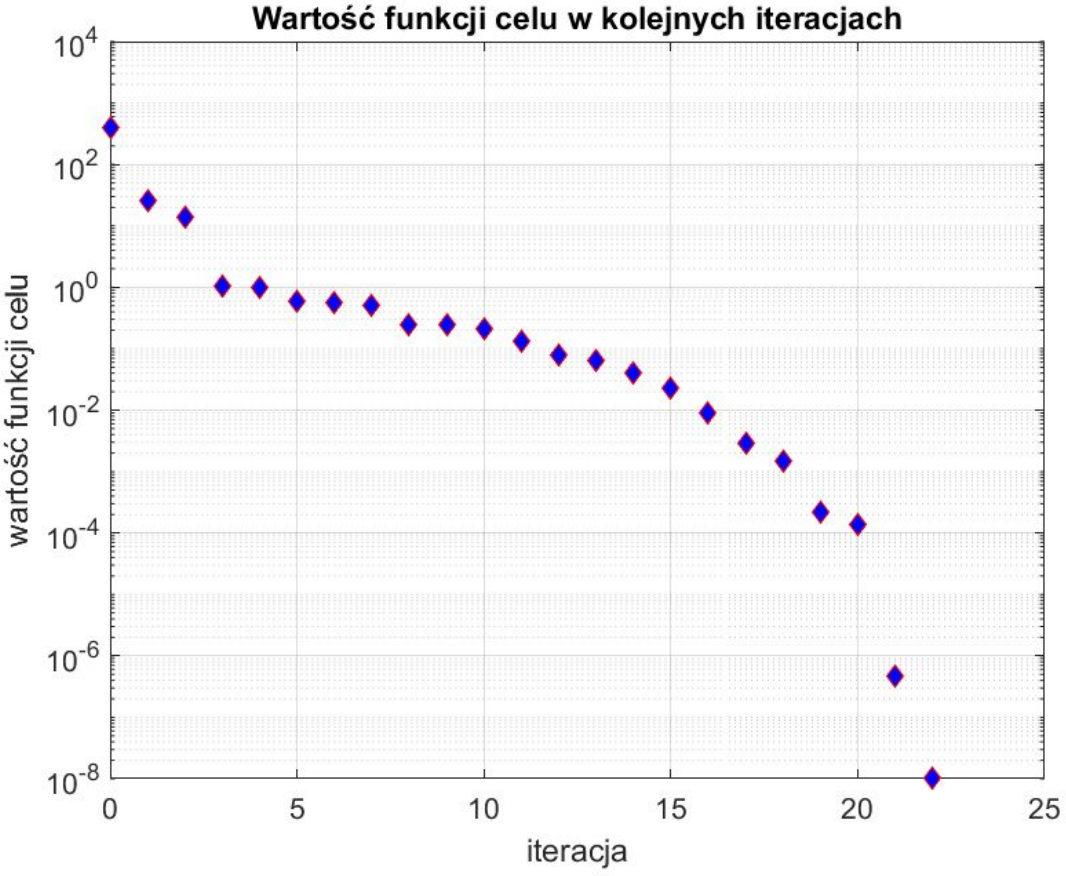




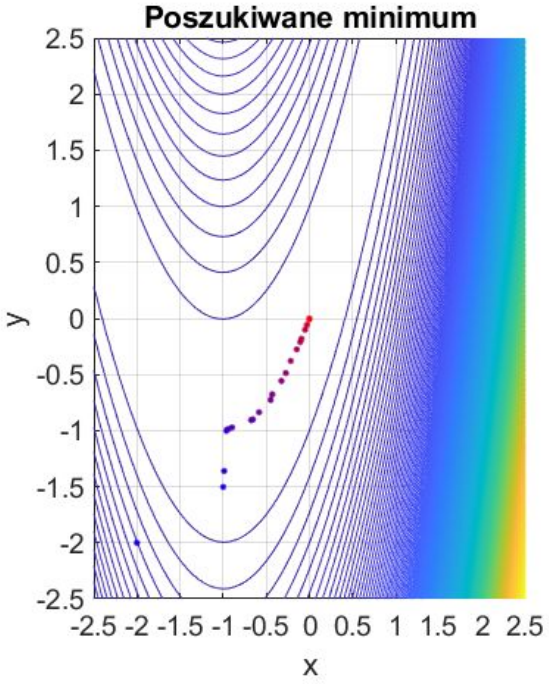
Punkt początkowy: x=0.000000, y=-2.000000
Znalezione optimum lokalne: x=-0.000101, y=-0.000203
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 22 Ilość obliczeń funkcji celu: 93

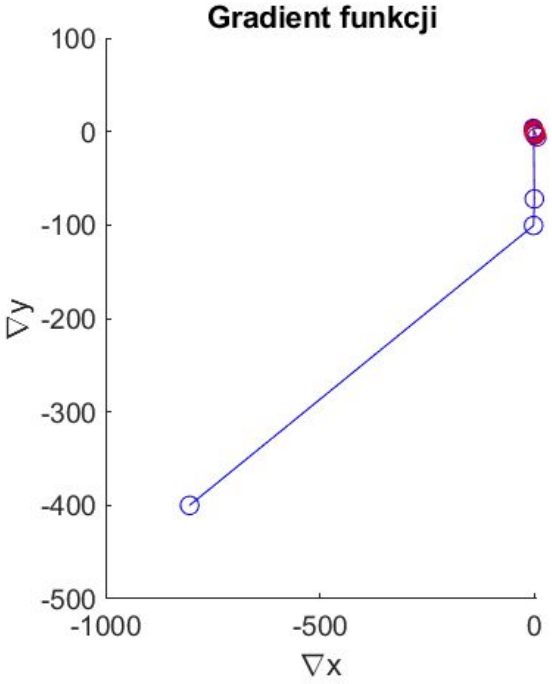


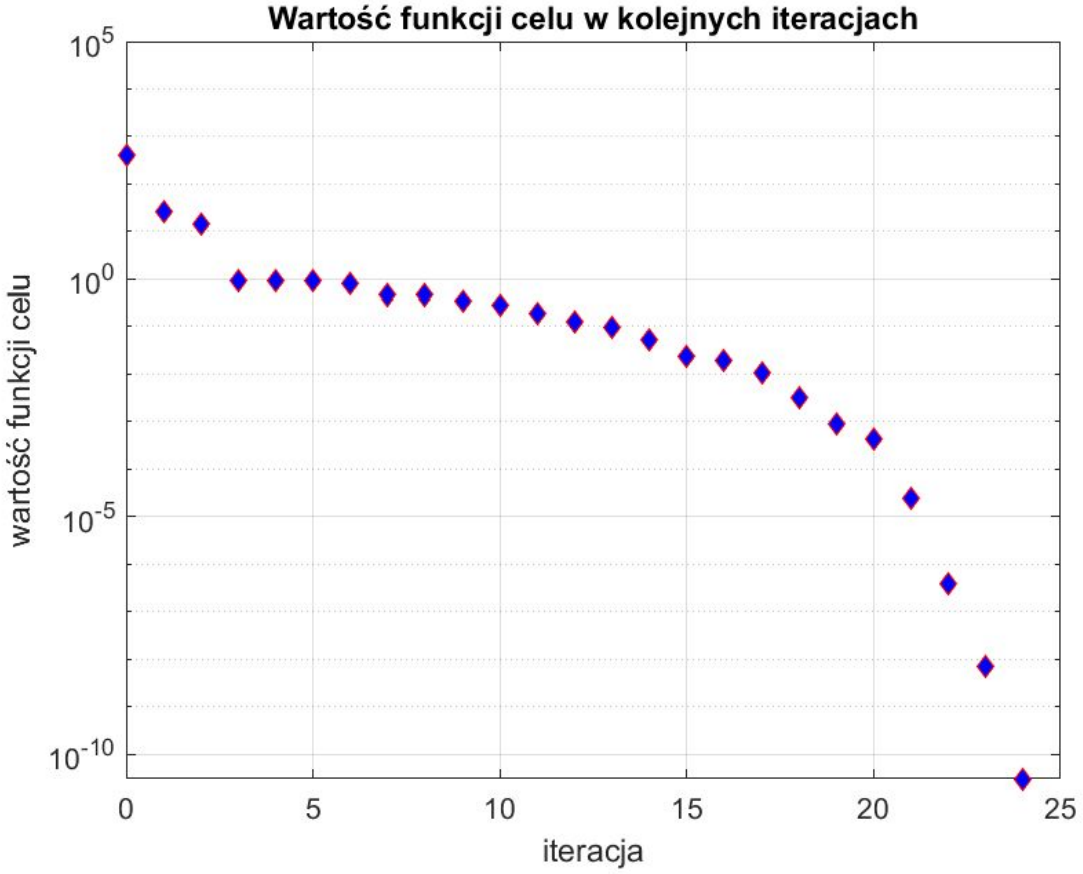




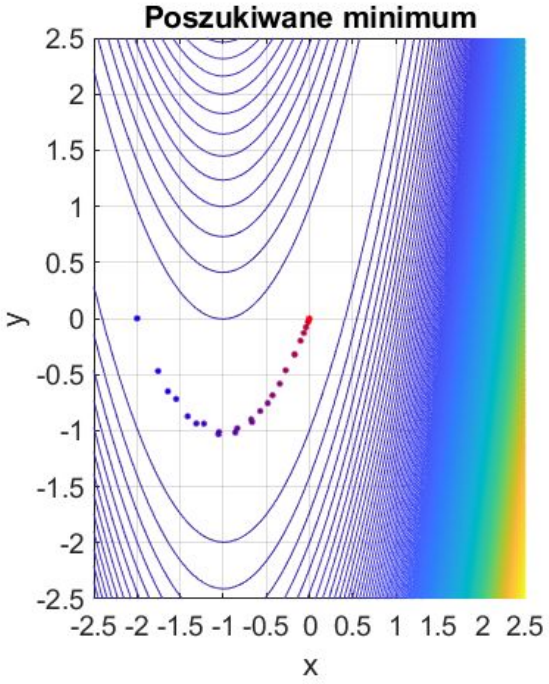
Punkt początkowy: x=-2.000000, y=-2.000000
Znalezione optimum lokalne: x=0.000004, y=0.000008
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 24 Ilość obliczeń funkcji celu: 90

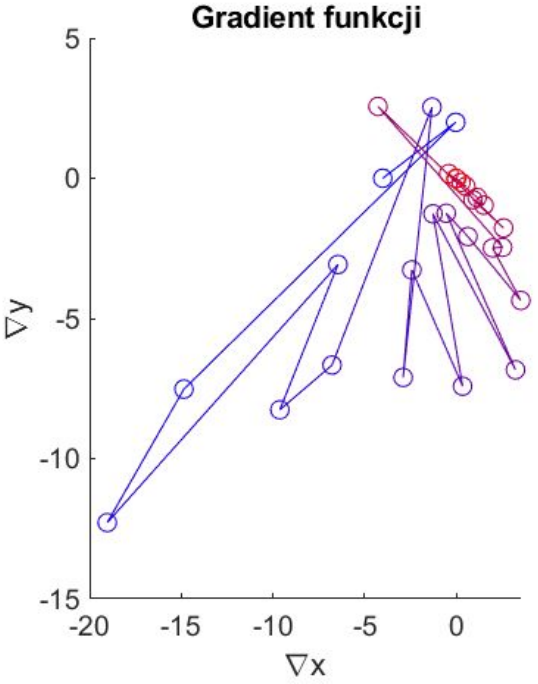


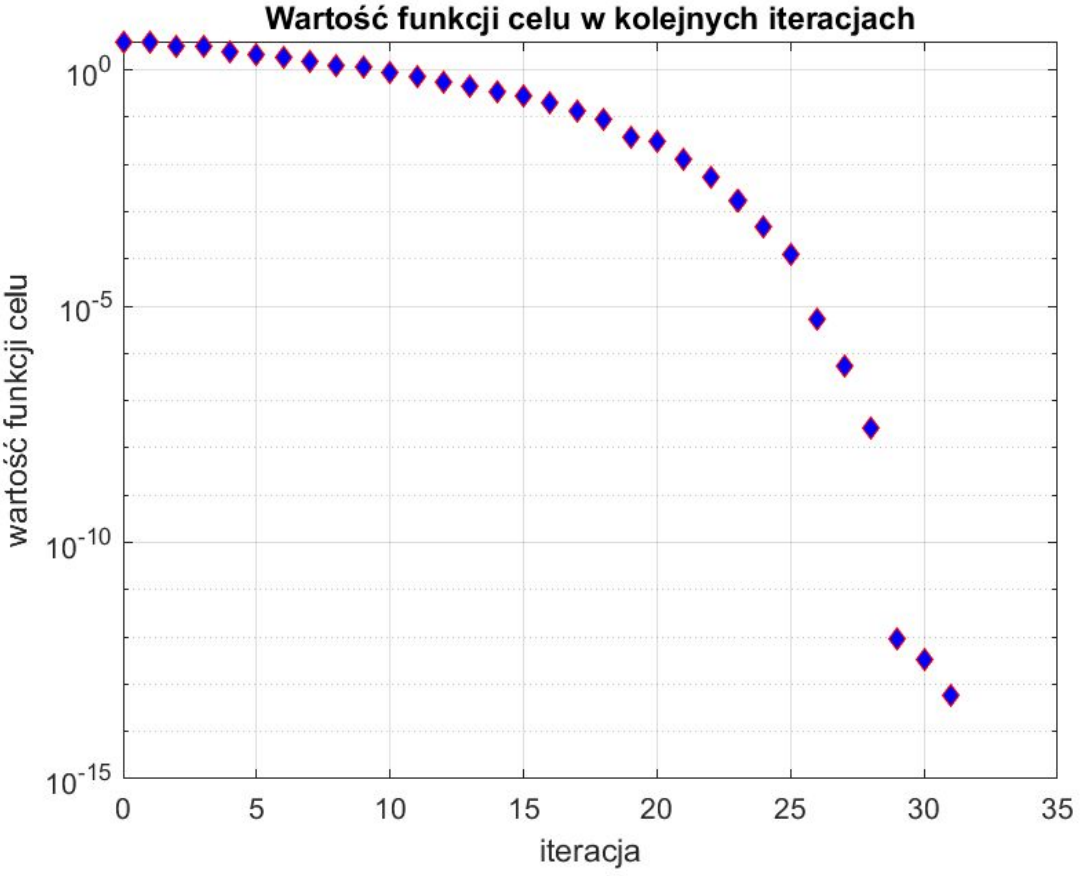




Punkt początkowy: x=-2.000000, y=0.000000
Znalezione optimum lokalne: x=-0.000000, y=-0.000000
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 31 Ilość obliczeń funkcji celu: 138

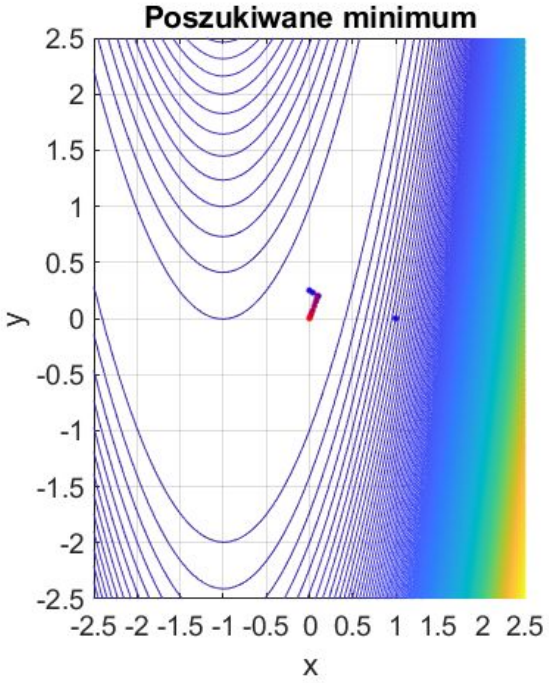


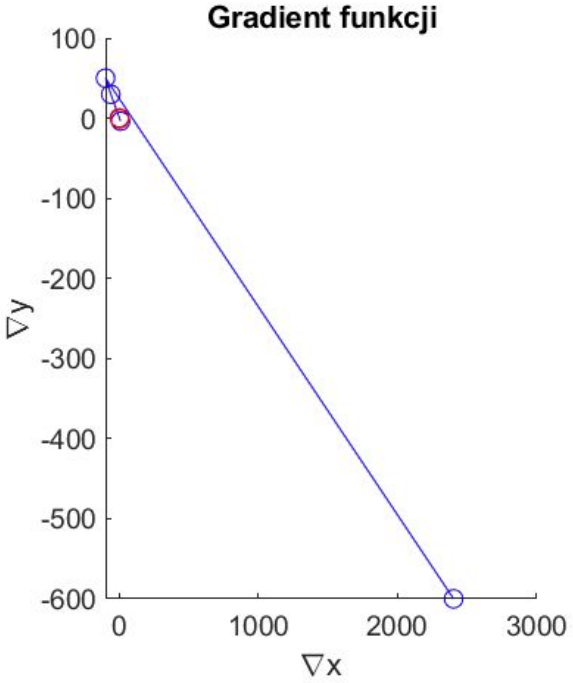


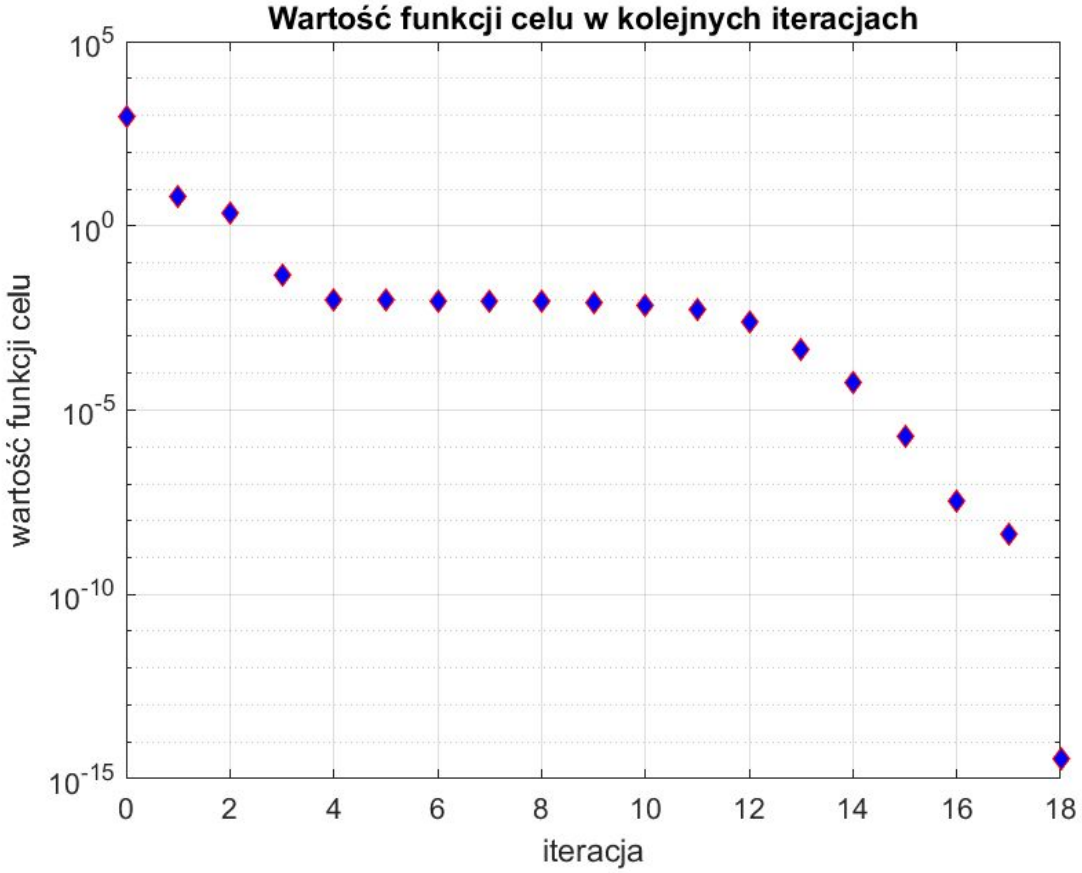


2.2. Metoda Quasi-Newtonowska z załączonym gradientem

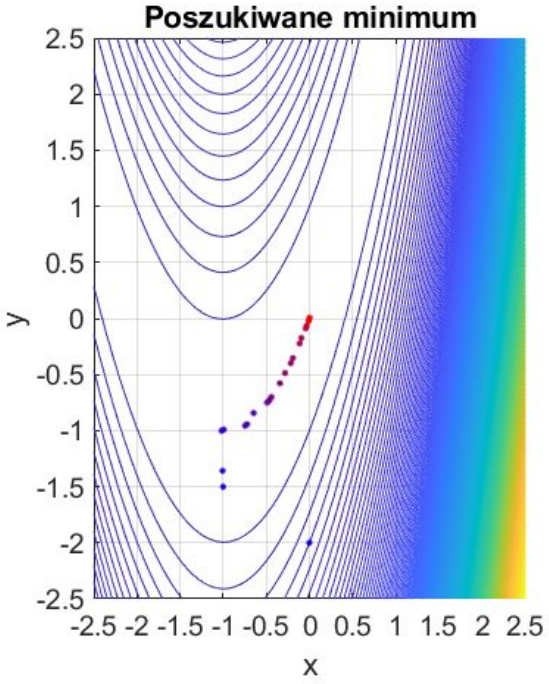
Punkt początkowy: x=1.000000, y=0.000000
Znalezione optimum lokalne: x=-0.000000, y=-0.000000
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 18 Ilość obliczeń funkcji celu: 20

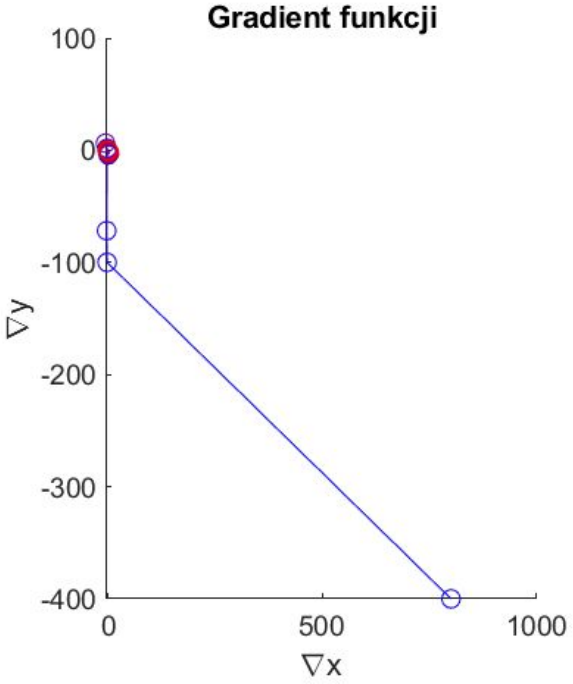


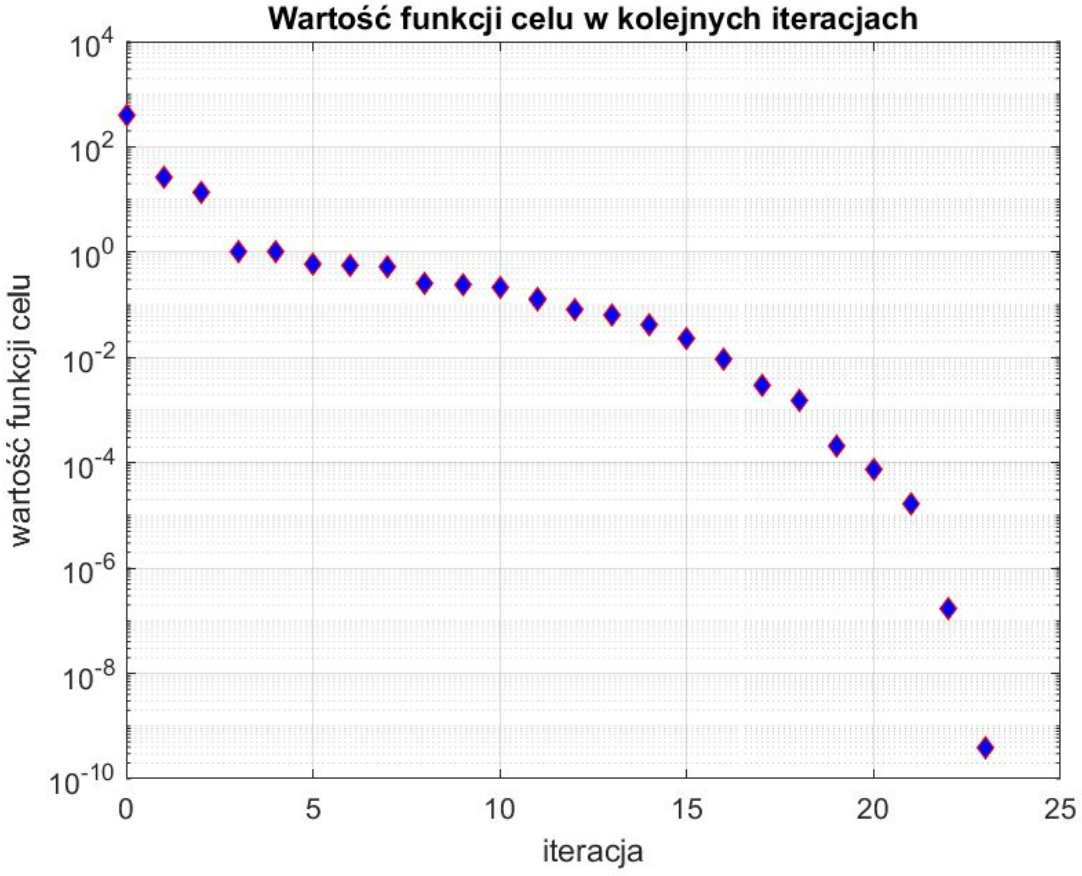




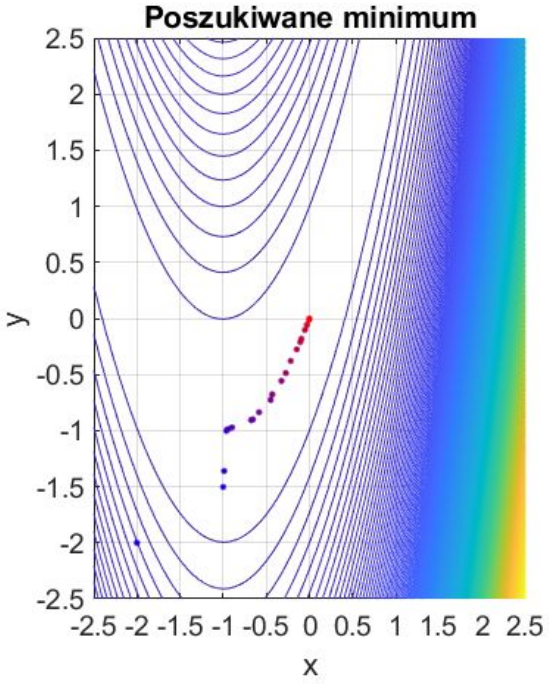
Punkt początkowy: x=0.000000, y=-2.000000
Znalezione optimum lokalne: x=-0.000013, y=-0.000025
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 23 Ilość obliczeń funkcji celu: 32

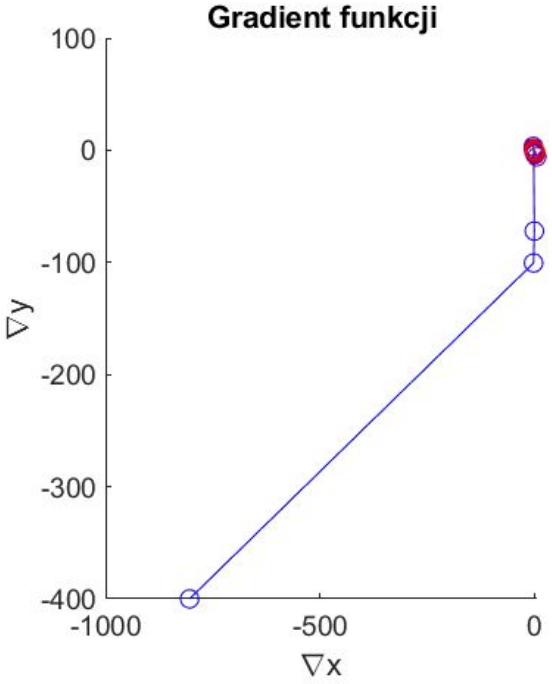


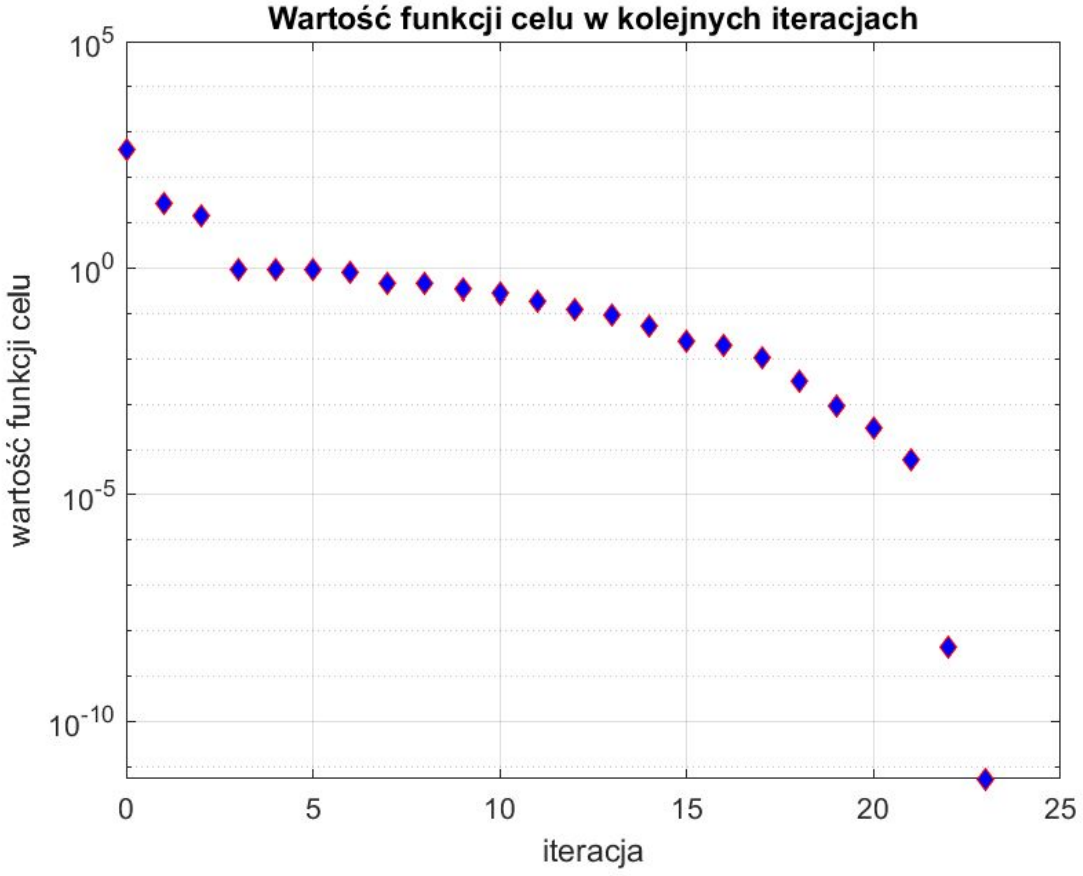




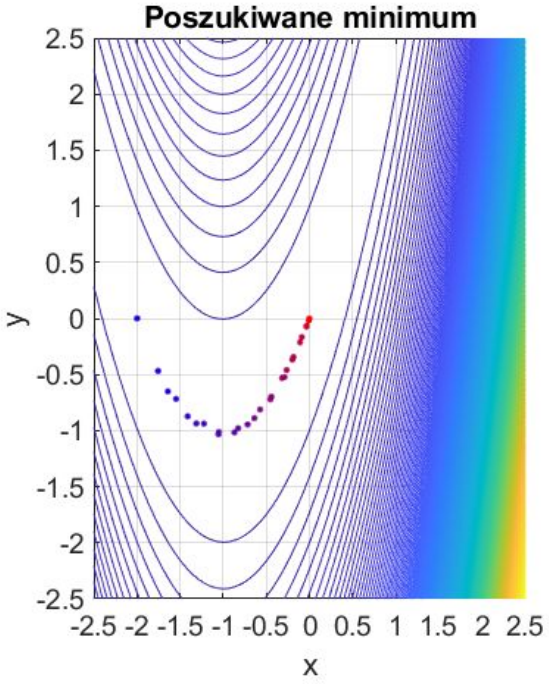
Punkt początkowy: x=-2.000000, y=-2.000000
Znalezione optimum lokalne: x=-0.000002, y=-0.000005
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 23 Ilość obliczeń funkcji celu: 29

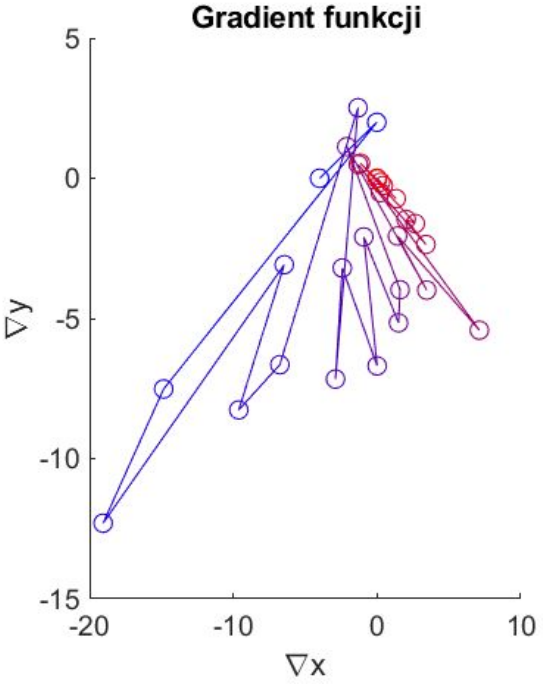


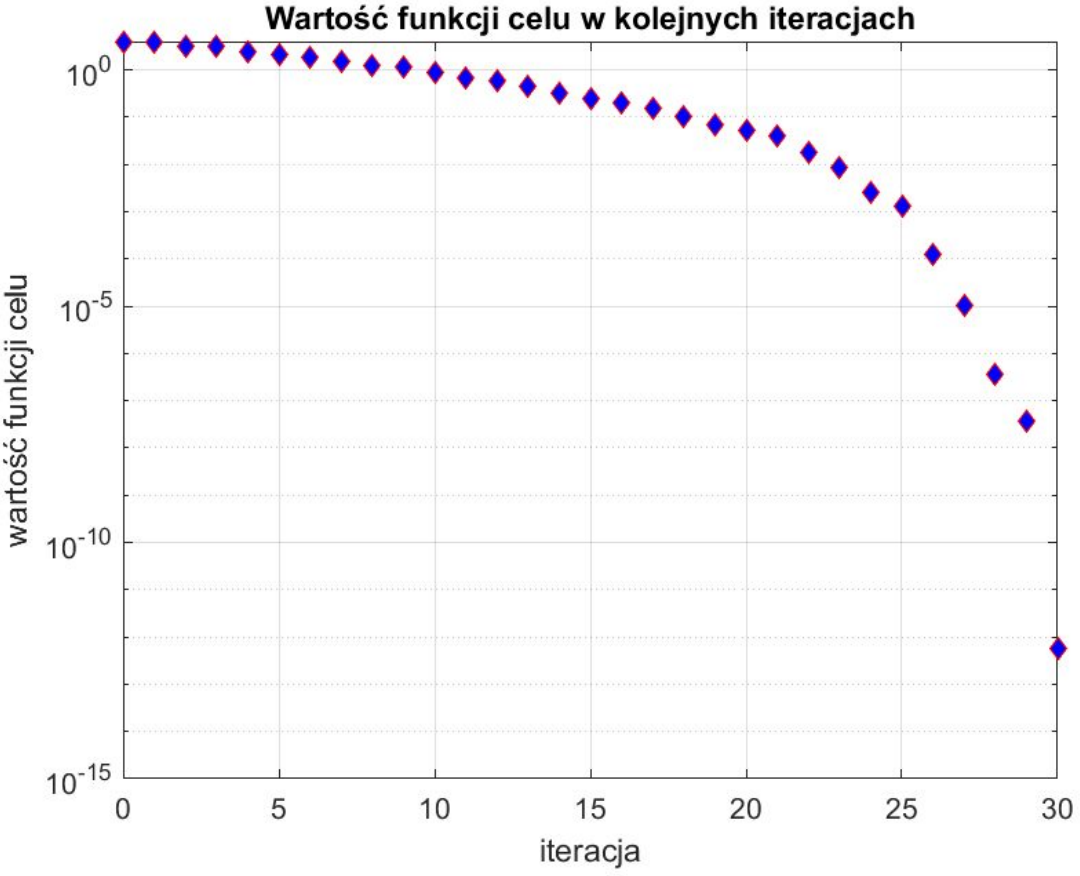




Punkt początkowy: x=-2.000000, y=0.000000
Znalezione optimum lokalne: x=-0.000001, y=-0.000002
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 30 Ilość obliczeń funkcji celu: 43

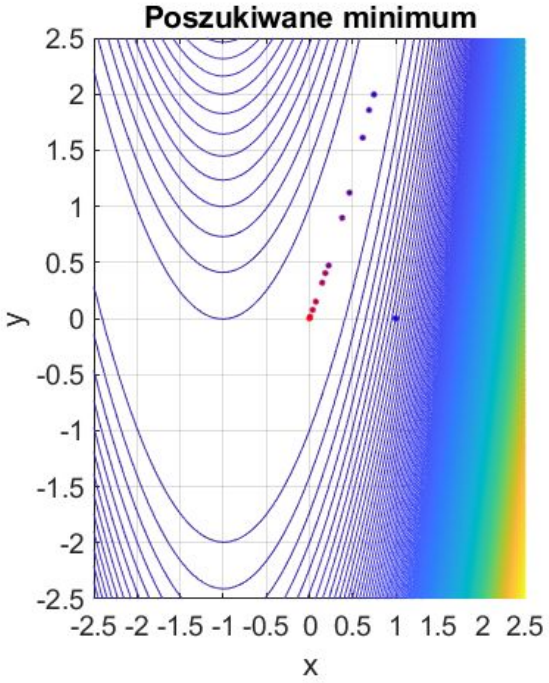


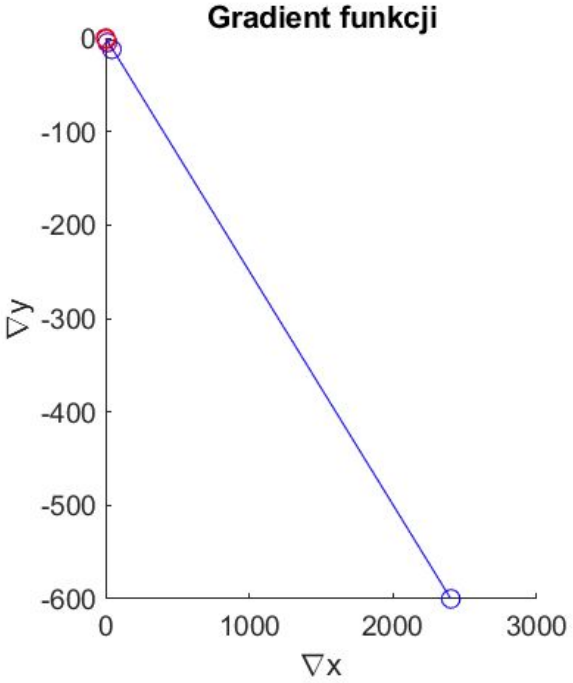


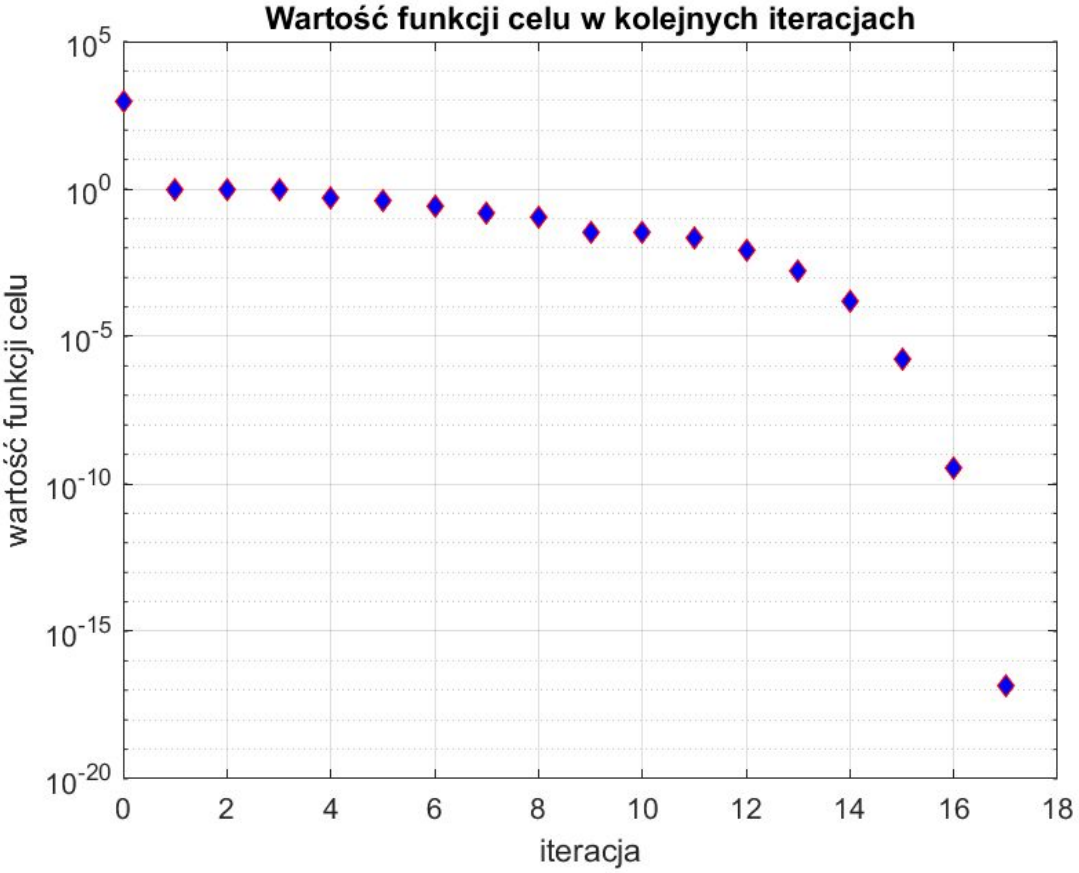


2.3. Metoda rejonu zaufania z załączonym gradientem

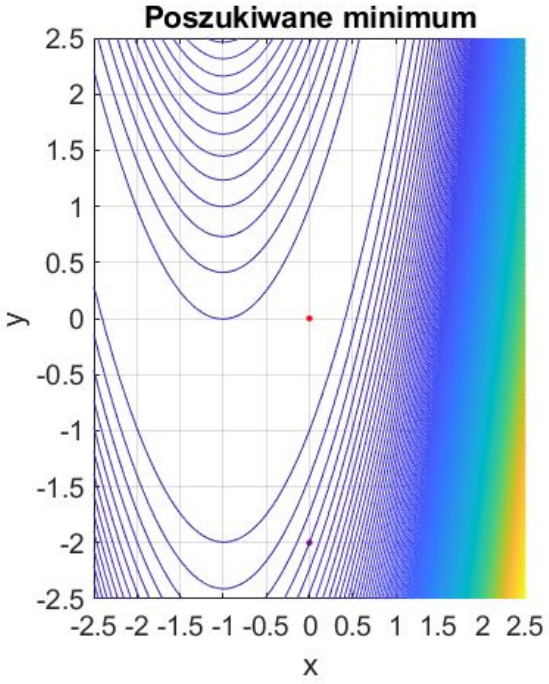
Punkt początkowy: x=1.000000, y=0.000000
Znalezione optimum lokalne: x=0.000000, y=0.000000
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 17 Ilość obliczeń funkcji celu: 18

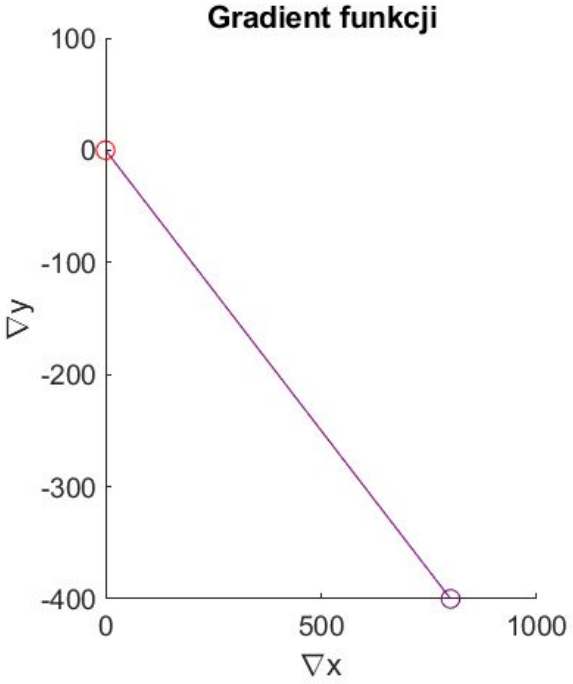


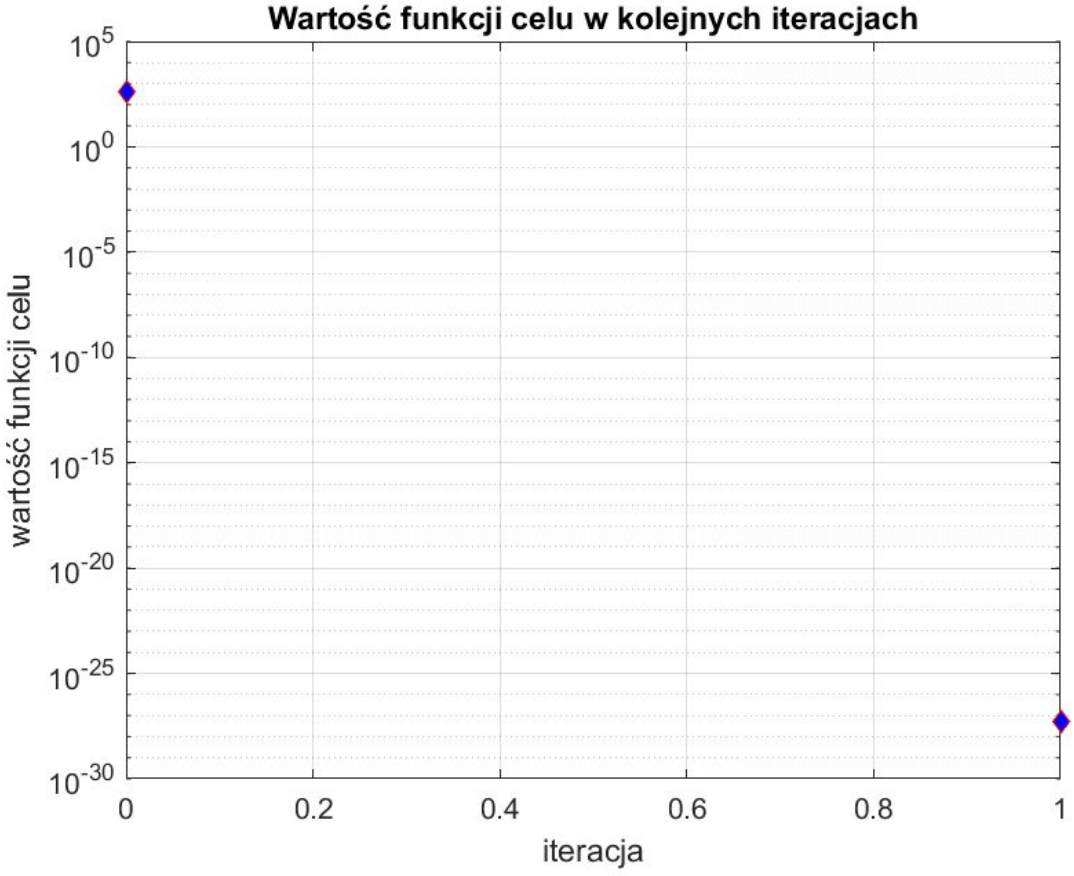




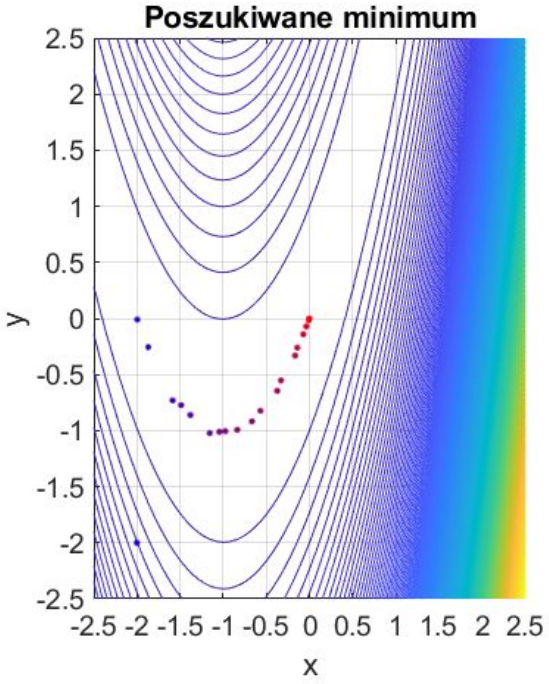
Punkt początkowy: x=0.000000, y=-2.000000
Znalezione optimum lokalne: x=0.000000, y=0.000000
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 1 Ilość obliczeń funkcji celu: 2

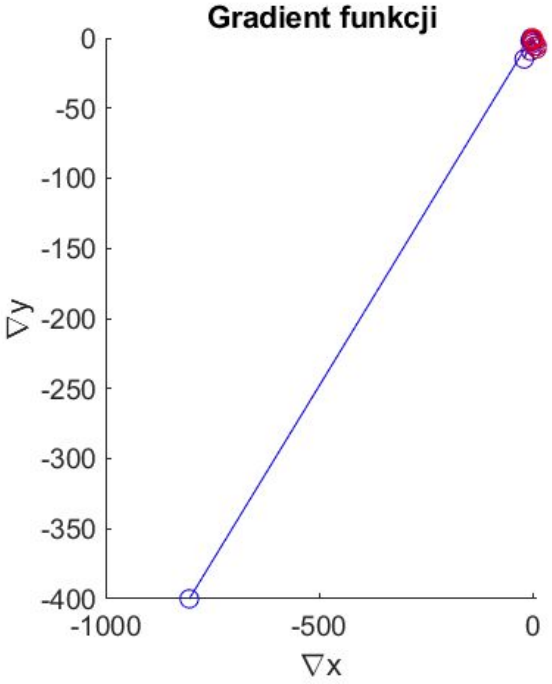


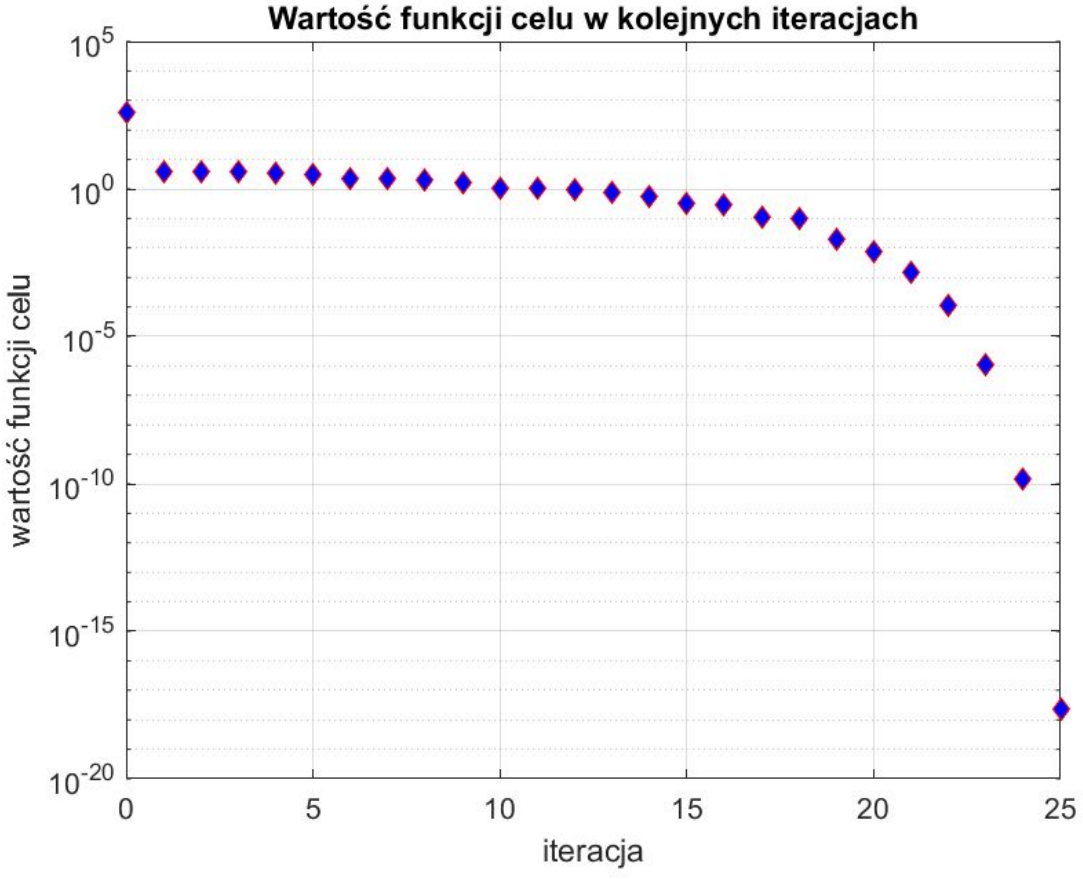




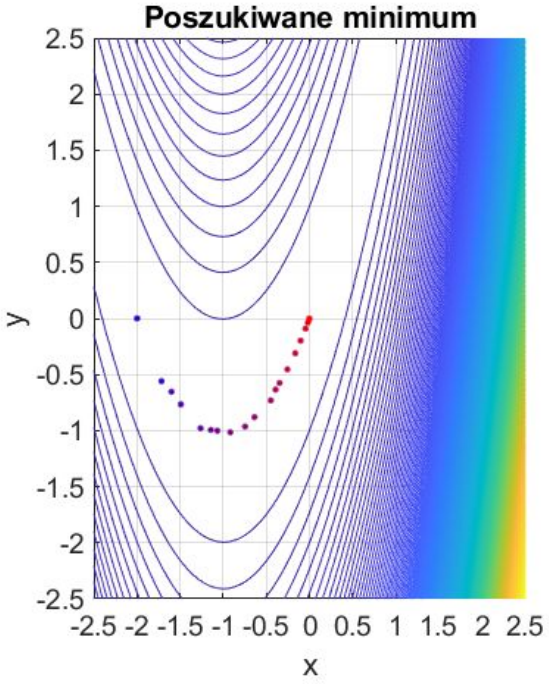
Punkt początkowy: x=-2.000000, y=-2.000000
Znalezione optimum lokalne: x=-0.000000, y=-0.000000
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 25 Ilość obliczeń funkcji celu: 26

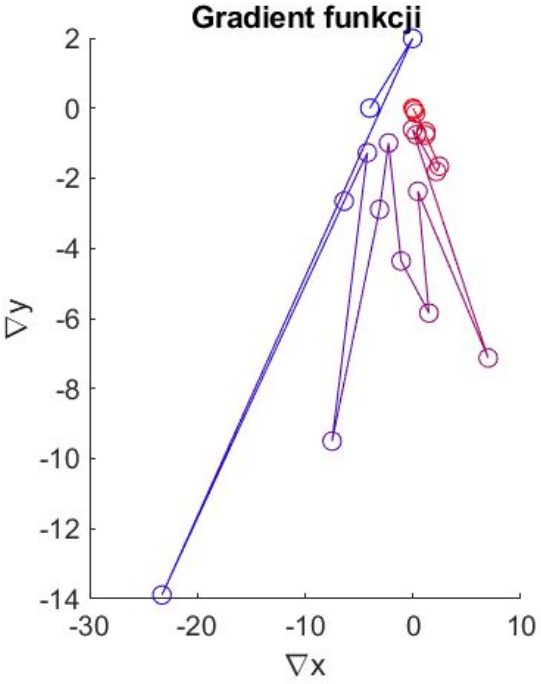


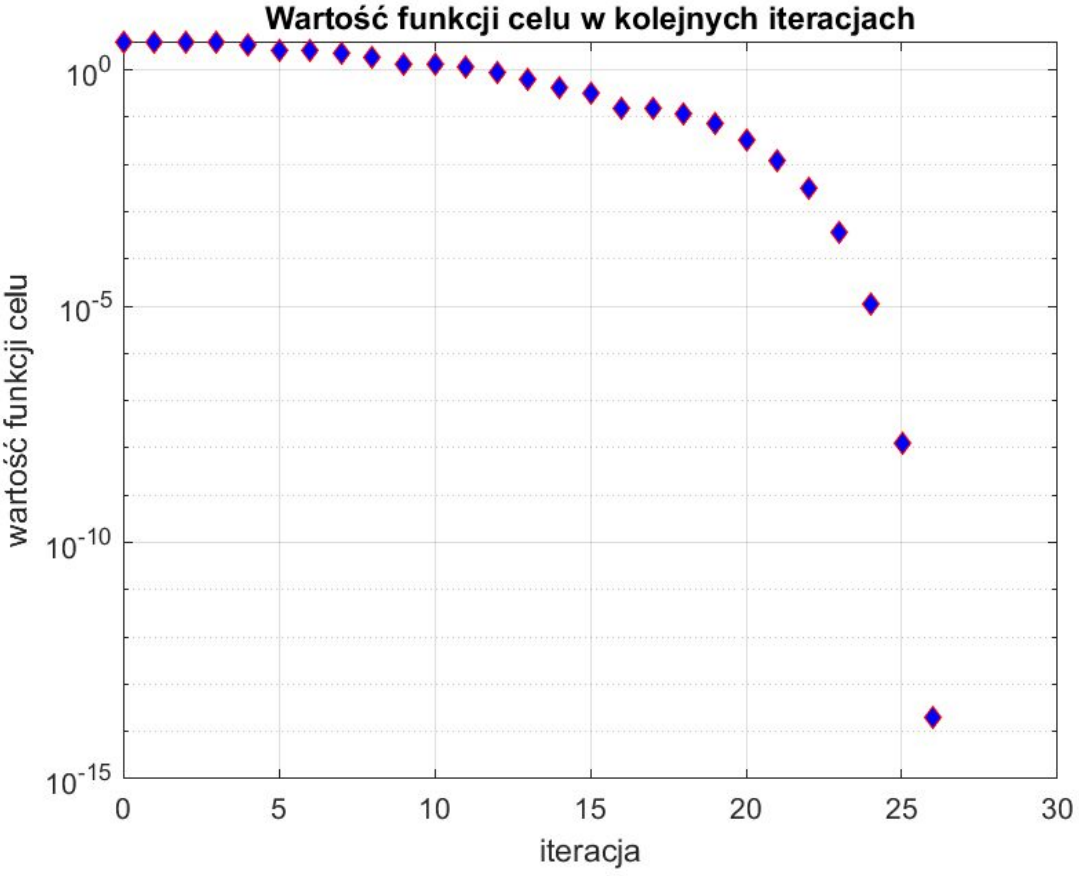




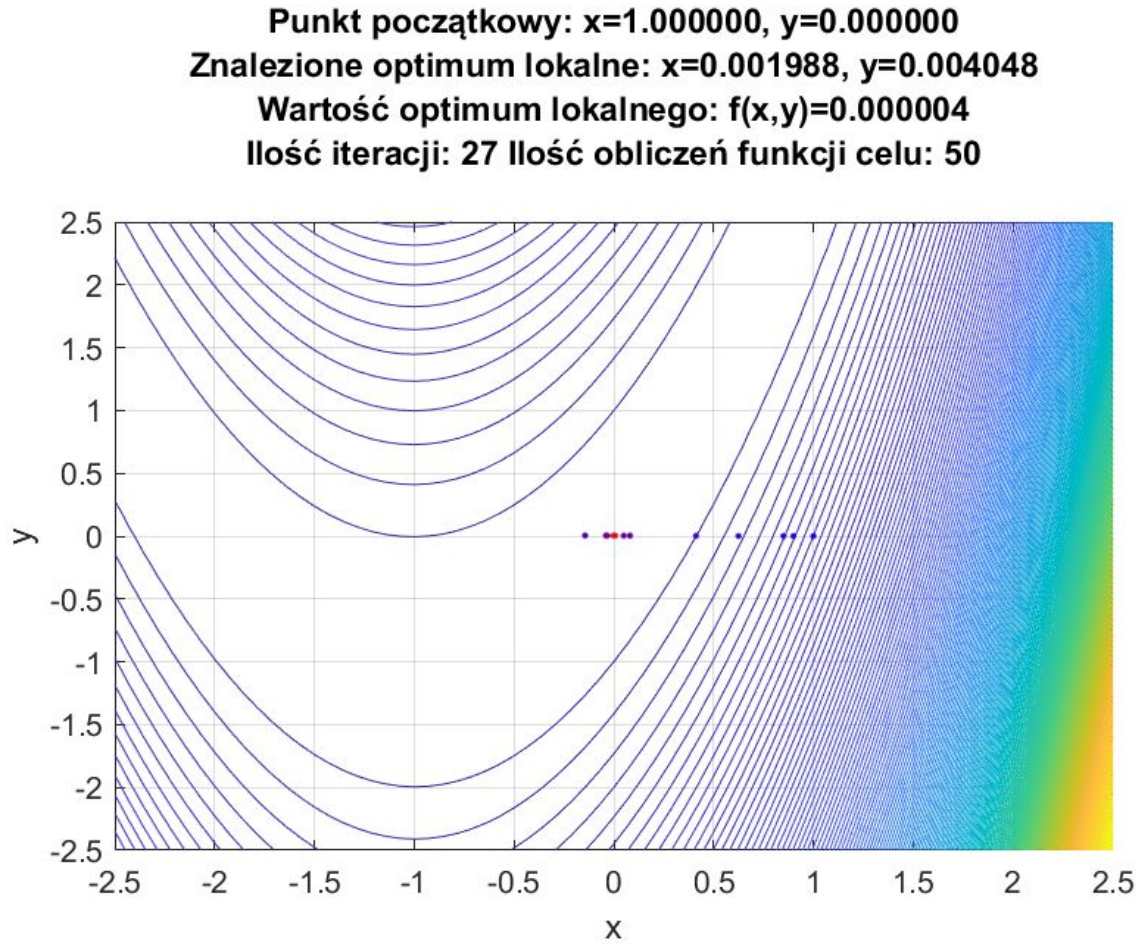
Punkt początkowy: x=-2.000000, y=0.000000
Znalezione optimum lokalne: x=-0.000000, y=-0.000000
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 26 Ilość obliczeń funkcji celu: 27

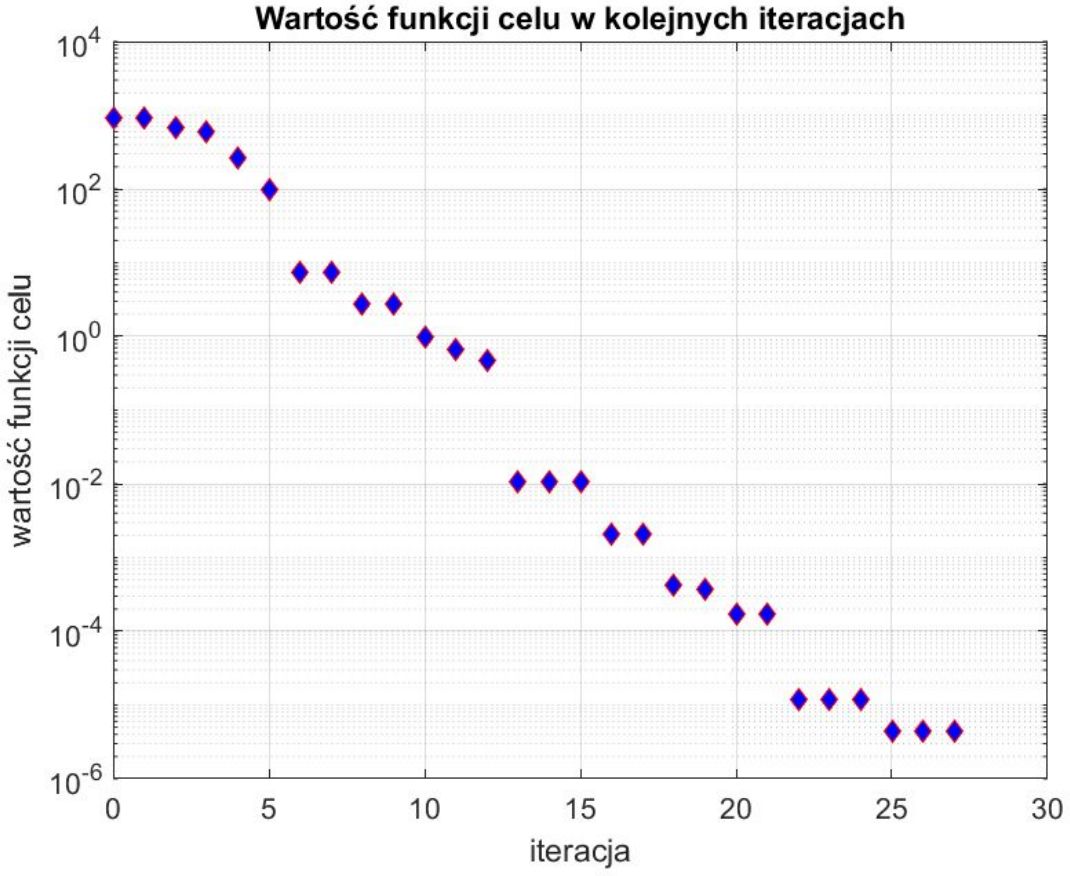




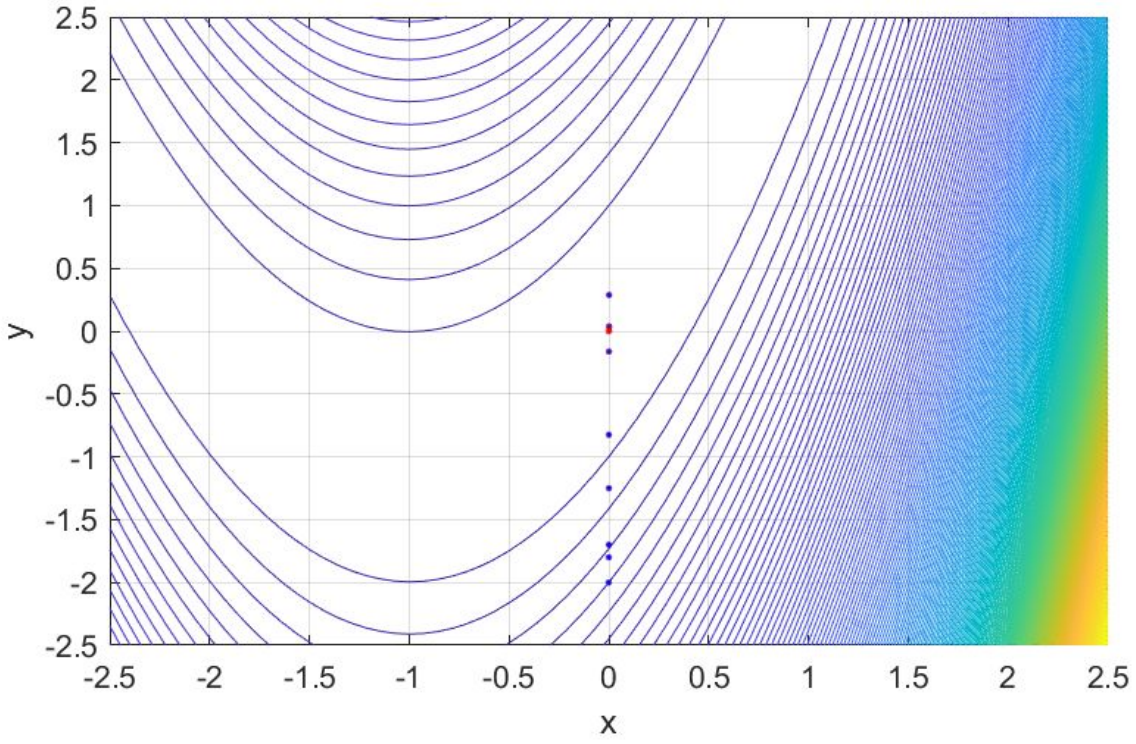


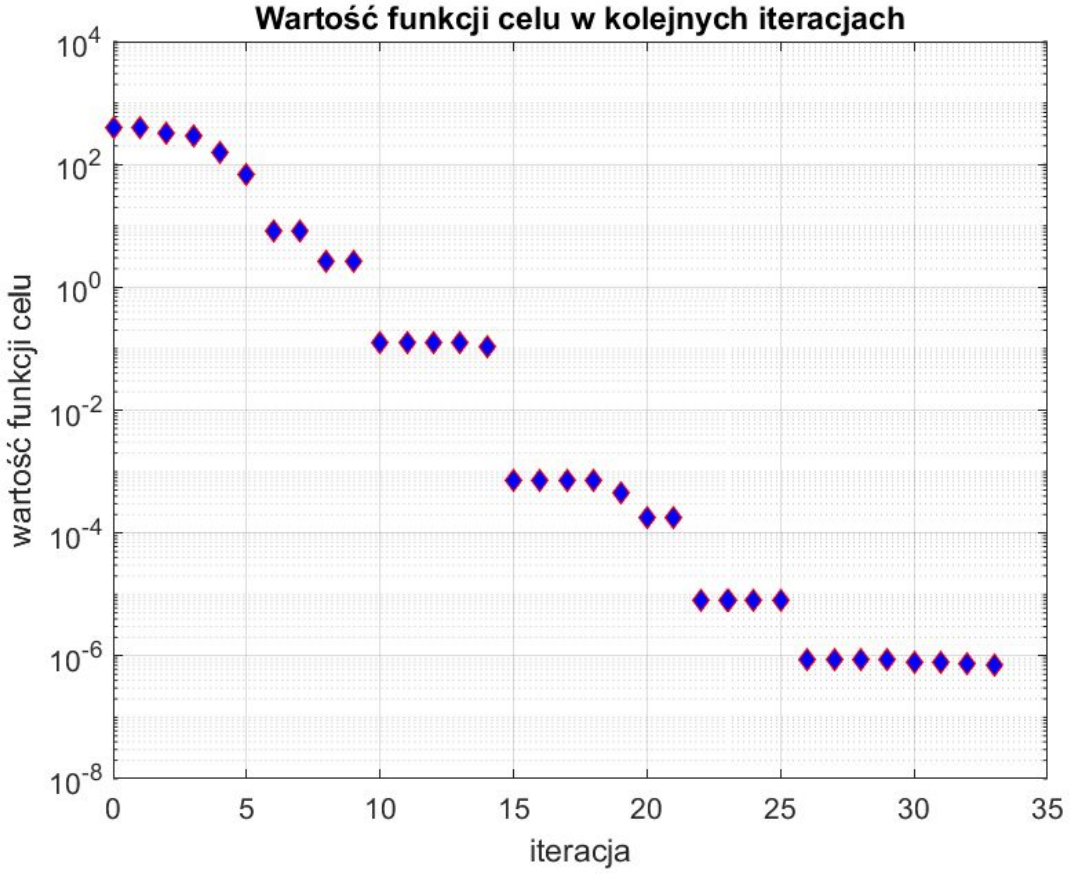
2.4. Metoda minimalizacji dostępna w MATLABie



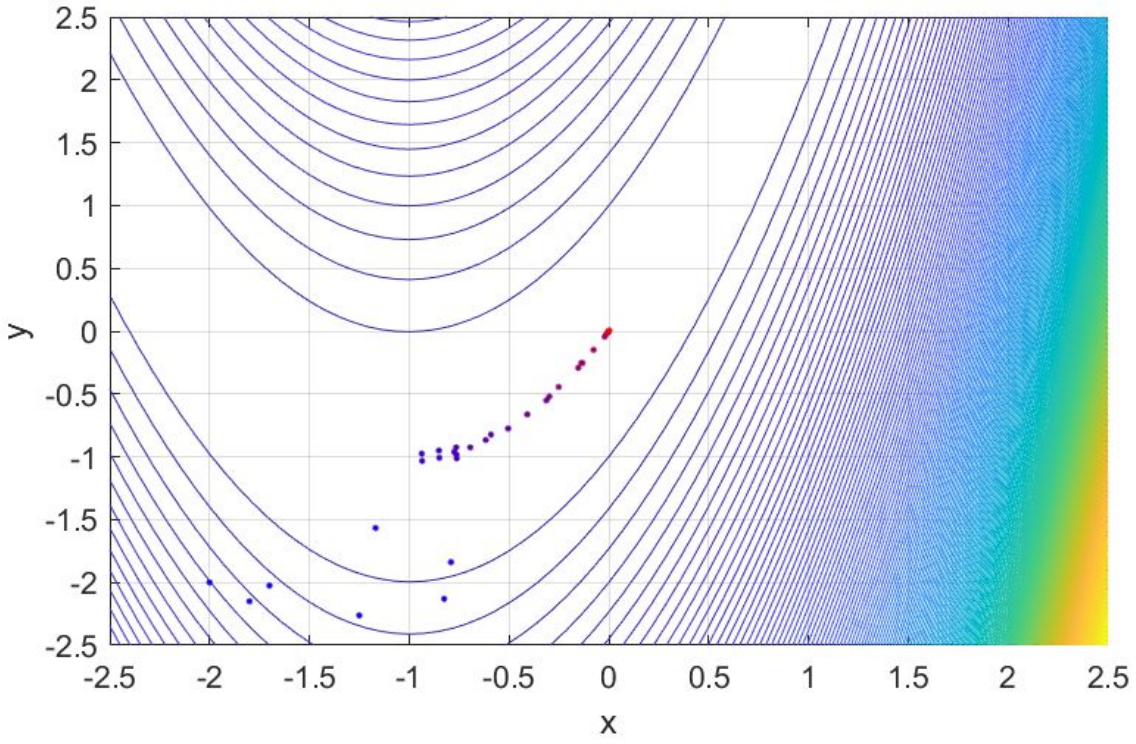


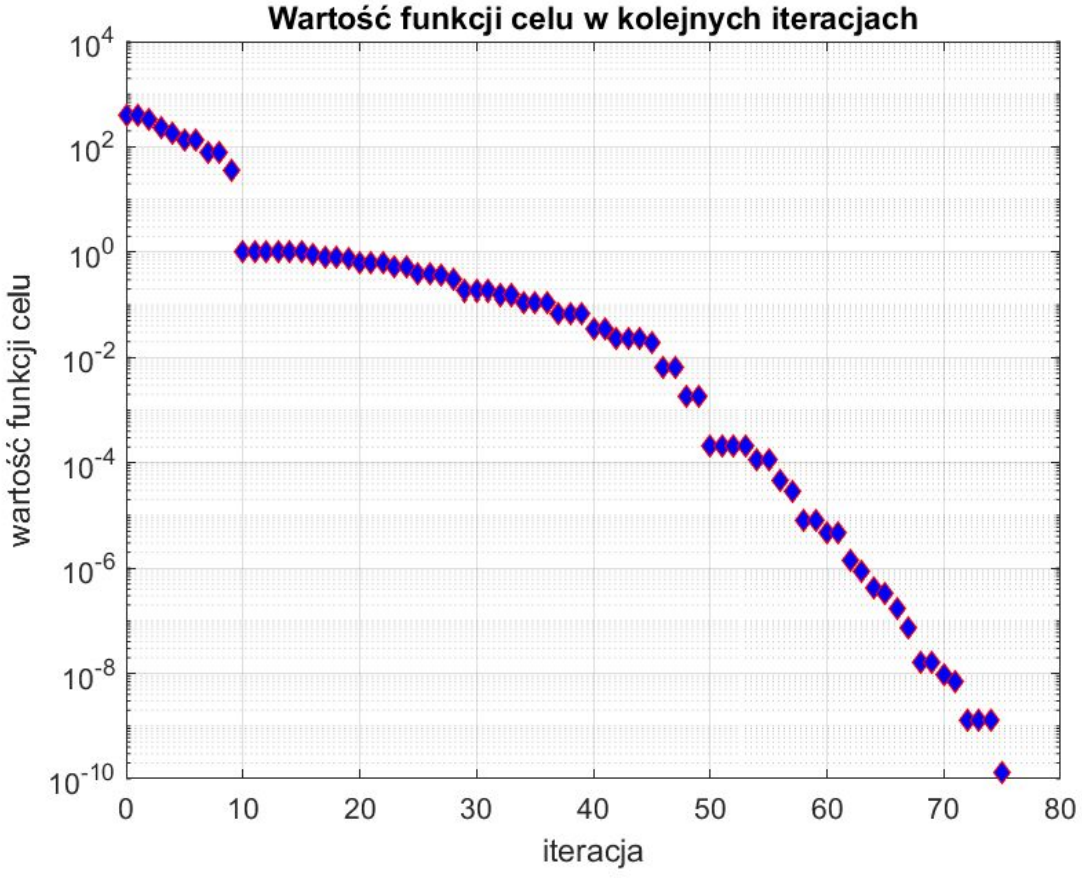
Punkt początkowy: x=0.000000, y=-2.000000
Znalezione optimum lokalne: x=0.000853, y=0.001707
Wartość optimum lokalnego: f(x,y)=0.000001
Ilość iteracji: 33 Ilość obliczeń funkcji celu: 61



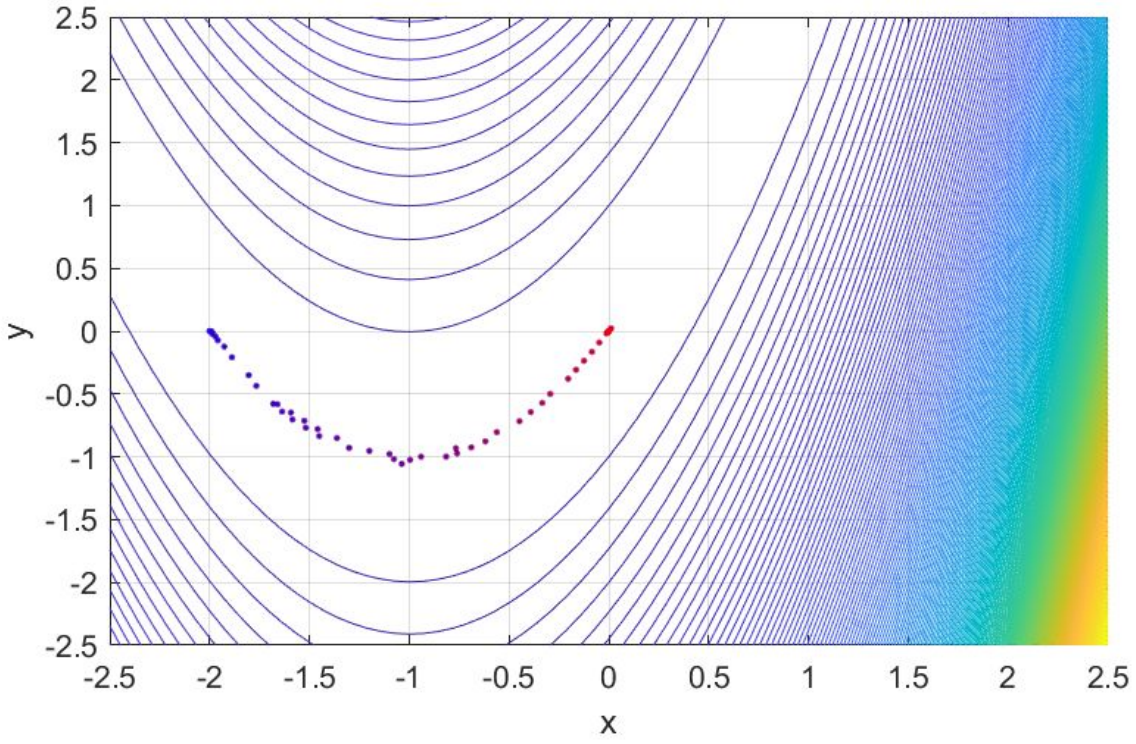


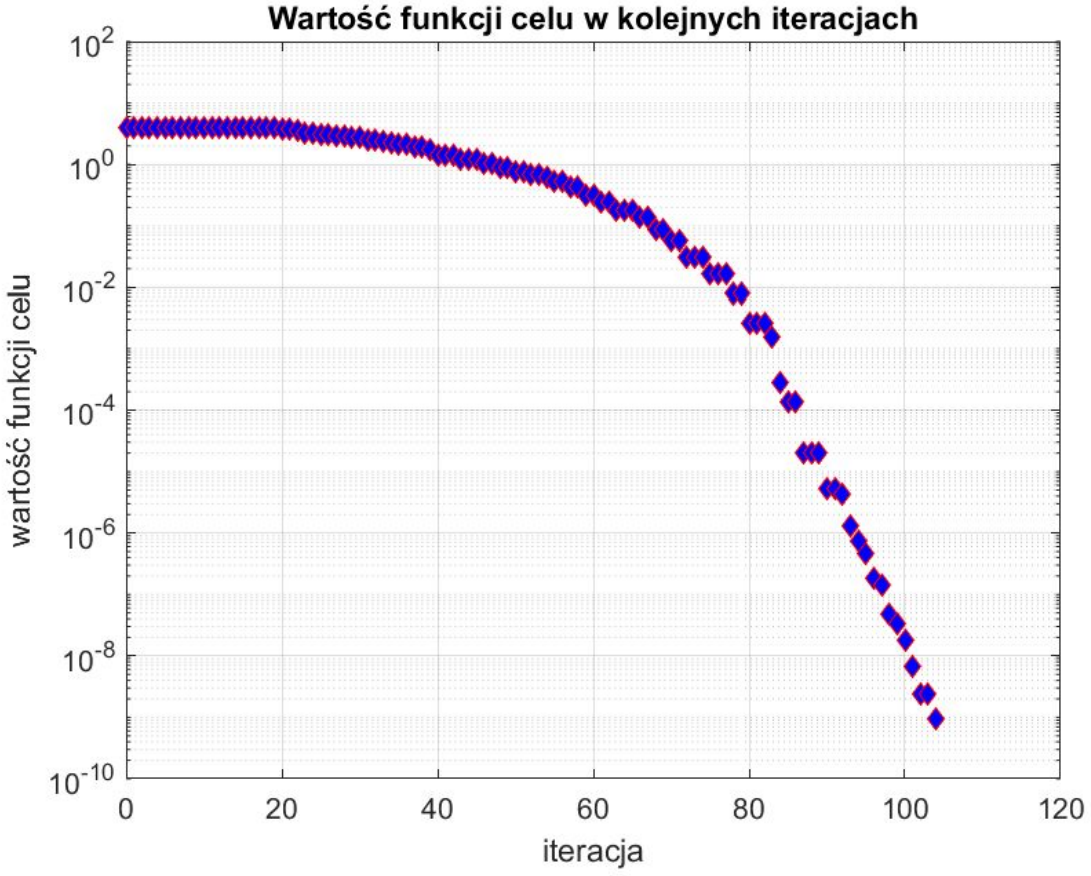
Punkt początkowy: x=-2.000000, y=-2.000000
Znalezione optimum lokalne: x=0.000002, y=0.000006
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 75 Ilość obliczeń funkcji celu: 144





Punkt początkowy: x=-2.000000, y=0.000000
Znalezione optimum lokalne: x=0.000026, y=0.000049
Wartość optimum lokalnego: f(x,y)=0.000000
Ilość iteracji: 104 Ilość obliczeń funkcji celu: 195





3. Ocena działania algorytmów

Najprostszy w użyciu algorytm *fminsearch* jest jednocześnie najwolniejszy i pochłania najwięcej zasobów - posiada statystycznie największą ilość obliczeń funkcji celu. Dzieje się tak ponieważ nie korzysta w żaden sposób z gradientu funkcji. Algorytm *fminunc* w przeciwieństwie do *fminsearch* korzysta z gradientu wyliczanego przez siebie lub dostarczanego równaniem. Skutkuje znacznym polepszeniem jego czasu szukania optimum i zużycia zasobów.
Dostarczenie gradientu w postaci równania do funkcji optymalizacyjnej znacznie poprawia jego zapotrzebowanie na zasoby. Można to zaobserwować na przykładzie algorytmu Quasi-Newton, gdzie ilość iteracji w obu testowanych wersjach jest zbliżona, natomiast ilość obliczeń funkcji celu jest znacząco mniejsza w przypadku, w którym dostarczony został gradient.
Algorytm Quasi-Newton z załączonym gradientem i Trust-Region z załączonym gradientem działają z podobną skutecznością jeśli chodzi o ilość iteracji i obliczeń funkcji celu (przypadek 2 w metodzie Trust-Region, gdzie algorytm znalazł rozwiązanie w 1 iteracji traktuję jako szczęśliwy przypadek). Mimo wszystko Trust-Region otrzymuje trochę lepsze wyniki niż Quasi-Newton.
Dokładność obliczonych optimów wynosi dla algorytmu *fminunc* wynosi $1,0e-6.$ W przypadku 4 punktu startowego i algorytmu regionu zaufania i Quasi-Newton wynik był bliski tej dokładności, ale algorytm został przerwany, ponieważ kroki algorytmu przekraczały jego minimalne wartości. Algorytm *fminsearch* obliczył optima z dokładnością $1,0e-4$ i wszystkie znalezione rozwiązania trzymały tą dokładność.

3.1. Tabela porównawcza

Algorytm	Punkt startowy	Punkt końcowy	Wartość funkcji	Liczba iteracji	Liczba obliczeń funkcji celu
Quasi-Newton	[1; 2]	[-3.704e-06; -7.394e-06]	1.374e-11	18	60
	[0; -2]	[-1.009e-04; -2.026e-04]	1.026e-08	22	93
	[-2; -2]	[3.943e-06; 7.505e-06]	3.014e-11	24	90
	[-2; 0]	[-2.371e-07; -4.739e-07]	5.624e-14	31	138
Quasi-Newton z załączonym gradientem	[1; 2]	[-5.824e-08; -1.165e-07]	3.392e-15	18	20
	[0; -2]	[-1.341e-05; -2.540e-05]	3.839e-10	23	32
	[-2; -2]	[-2.328e-06; -4.646e-06]	5.433e-12	23	29
	[-2; 0]	[-7.517e-07; -1.506e-06]	5.658e-13	30	43
Trust-Region z załączonym gradientem	[1; 2]	[3.484e-09; 6.841e-09]	1.376e-17	17	18

	[0; -2]	[9.994e-17; 2.220e-15]	4.931e-28	1	2
	[-2; -2]	[-1.416e-09; -2.884e-09]	2.271e-18	25	26
	[-2; 0]	[-9.353e-08; -1.979e-07]	2.048e-14	26	27
fminsearch	[1; 2]	[1.988e-03; 4.048e-03]	4.420e-06	27	50
	[0; -2]	[8.525e-04; 1.707e-03]	7.270e-07	33	61
	[-2; -2]	[2.199e-06; 5.538e-06]	1.349e-10	75	144
	[-2; 0]	[2.556e-05; 4.938e-05]	9.562e-10	104	195