

Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Wstęp do sztucznej inteligencji

Raport z laboratorium nr 1

Jan Kuc

Warszawa, 2022

Spis treści

1. Wstęp	2
1.1. Ogólna postać algorytmów optymalizacji	2
1.2. Algorytm gradientu prostego	2
2. Eksperymenty i wyniki	4
2.1. Funkcja jednowymiarowa	4
2.1.1. Testy dla $x_0=-1.5$	4
2.1.2. Testy dla $x_0=2.1$	7
2.1.3. Testy dla $x_0=-4$	9
2.1.4. Testy dla $x_0=0.2$	11
2.1.5. Wnioski	13
2.2. Funkcja dwuwymiarowa	14
2.2.1. Testy dla $x_0=[2,-2]$	14
2.2.2. Testy dla $x_0=[1,2.5]$	16
2.2.3. Testy dla $x_0=[-1,-1.5]$	18
2.2.4. Testy dla $x_0=[0.2,0.6]$	21
2.2.5. Testy dla $x_0=[-2.3,1.8]$	23
2.2.6. Wnioski	24

1. Wstęp

Celem ćwiczenia była implementacja algorytmu gradientu prostego, a następnie zastosowanie go do znalezienia minimum dwóch funkcji: \mathbf{f} i \mathbf{g} . Następnie, należało przeprowadzić eksperymenty, w celu zbadania wpływu rozmiaru kroku dla empirycznie dobieranych, różnych punktów początkowych.

Do badań wykorzystano poniższe funkcje:

$$f(x) = \frac{1}{4}x^4$$

$$g(x) = 2 - \exp(-x_1^2 - x_2^2) - 0.5 \exp(-(x_1 + 1.5)^2 - (x_2 - 2)^2)$$

Oraz ich gradienty:

$$\nabla f(x) = x^3$$

$$\nabla g(x) = \begin{bmatrix} 2x_1 \exp(x_1^2 - x_2^2) + (x_1 + 1.5) \exp(-(x_1 + 1.5)^2 - (x_2 - 2)^2) \\ 2x_2 \exp(-x_1^2 - x_2^2) + (x_2 - 2) \exp(-(x_1 + 1.5)^2 - (x_2 - 2)^2) \end{bmatrix}$$

1.1. Ogólna postać algorytmów optymalizacji

Ogólną postać problemu optymalizacji lokalnej bez ograniczeń, przy założeniu, że funkcja f jest różniczkowalna, można przedstawić za pomocą wzoru:

$$f^x = \min\{f(x) : x \in R^n\}$$

Dla $x \in R^n \nabla f(x) = 0$, tzn. punktu startowego nie będącego punktem stacjonarnym.

Wyznaczamy nowy punkt $x^{k+1} = x^k + \alpha d$

Gdzie:

$d \in R^n$ jest kierunkiem poprawy α oznacza długość kroku, w którym dokonywane jest przesunięcie w kierunku d , tak aby $f(x^{k+1}) = f(x^k + \alpha d) < f(x^k)$

1.2. Algorytm gradientu prostego

Algorytm gradientu prostego należy do rodziny algorytmów kierunków poprawy, których ogólne działania przedstawiono powyżej. W każdej iteracji wykonywany jest krok w najlepszym kierunku wyznaczonym przez ujemny gradient funkcji. Zależnie od wybranej metody optymalizacji, długość kroku w optymalnym kierunku może być stała w każdym kroku lub dobrana za pomocą metod optymalizacji jednowymiarowej w sposób, który zagwarantuje możliwie największy spadek wartości funkcji. Działanie jest wolne, algorytm jest zbieżny liniowo.

W trakcie implementacji algorytmu należy również zadbać o dobranie odpowiednich **kryteriów stopu**, a więc warunków, dla których spełnienie chociaż jednego oznacza zakończenie działania algorytmu.

W tym przypadku jako kryteria stopu wykorzystano:

1. $k \leq max_k$ (maksymalna liczba iteracji)
2. $\|\nabla f(x_k)\| \leq \epsilon$ (gradient bliski zeru - test stacjonarności)
3. $\|x_{k+1} - x_k\| \leq \epsilon$ (brak poprawy rozwiązania)

2. Eksperymenty i wyniki

Podczas przeprowadzonych eksperymentów badany był głównie wpływ zmiany długości kroku początkowego na działanie algorytmu, przy różnych punktach startowych.

Oprócz tego, dobierane były parametry takie jak:

- β - współczynnik zmiany długości kroku
- ϵ - dopuszczalna tolerancja
- *max_it* - maksymalna liczba iteracji

Dla każdego wykonanego testu zwracane są dane zawierające informacje o parametrach początkowych, a także:

- punkt stanowiący znalezione rozwiązanie
- wartość funkcji w wyznaczonym punkcie
- rozmiar kroku e , gdyż może on ulec zmianie w trakcie działania algorytmu
- kryterium stopu, które zatrzymało algorytm
- wykonaną liczbę iteracji

W przypadku rozwiązywania problemu funkcji jednowymiarowej, program generuje dwa wykresy:

1. wykres wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej
2. wykres $f(x_i)$ - będący trajektorią punktów generowanych przez optymalizator w każdej iteracji

Dla problemu funkcji dwuwymiarowej otrzymujemy:

1. wykres konturowy prezentujący trajektorię punktów generowanych w kolejnych iteracjach algorytmu
2. wykres trójwymiarowy również pokazujący ścieżkę generowanych punktów

2.1. Funkcja jednowymiarowa

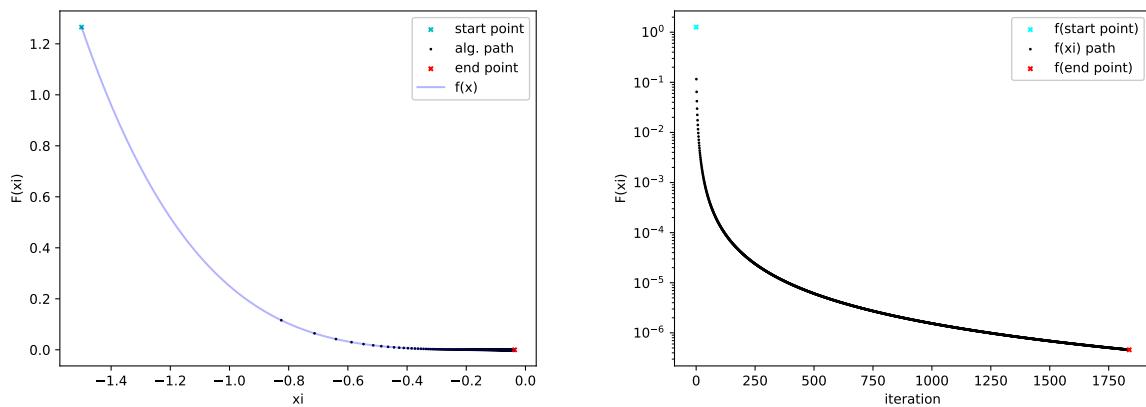
Wybrano 4 punkty początkowe $x_0: -1.5, 2.1, -4, 0.2$, dla których przeprowadzono eksperymenty zmiany długości kroku.

2.1.1. Testy dla $x_0=-1.5$

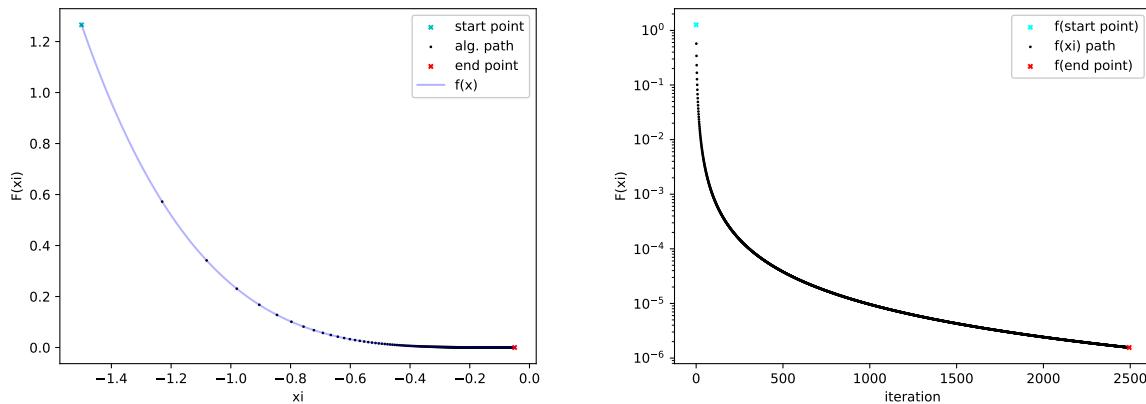
x_0	krok e	β	ϵ	max_it	x_i	$f_{\min}(x_i)$	kryt. stopu	l. iteracji	krok e
-1.5	0.2	0.9	0.00001	10000	-0.0368	4.6e-07	brak poprawy	1836	0.2
-1.5	0.08	0.9	0.00001	10000	-0.0499	1.56e-06	brak poprawy	2494	0.08
-1.5	0.01	0.9	0.00001	10000	-0.099	2.5e-05	brak poprawy	4976	0.01
-1.5	0.004	0.9	0.00001	10000	-0.136	8.48e-05	brak poprawy	2494	0.004
-1.5	0.6	0.9	0.00001	10000	0.0255	1.06e-07	brak poprawy	1273	0.6
-1.5	0.9	0.9	0.00001	10000	-0.022	6.28e-08	brak poprawy	1112	0.81

Tab. 2.1: Wyniki dla algorytmu gradientu prostego, przy zmianie długości kroku e

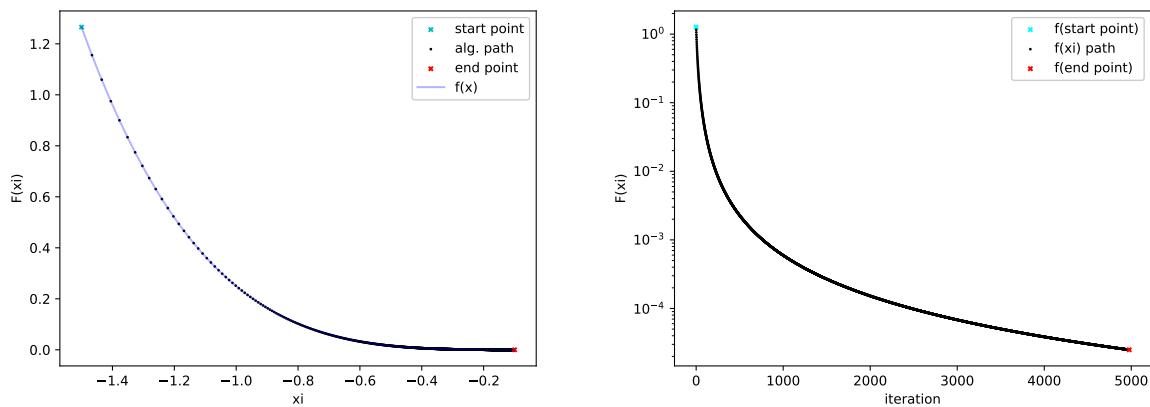
$$x_0 = -1.5$$



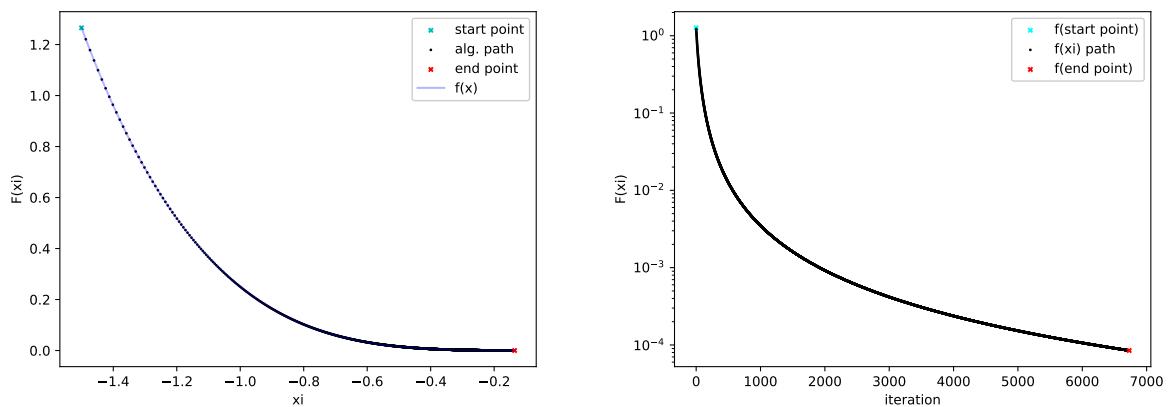
Rys. 2.1: Wykresy: trajektorii znajdowania minimum funkcji, a tak e zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = -1.5$ i początkowej d ugo ci kroku $e = 0.2$.



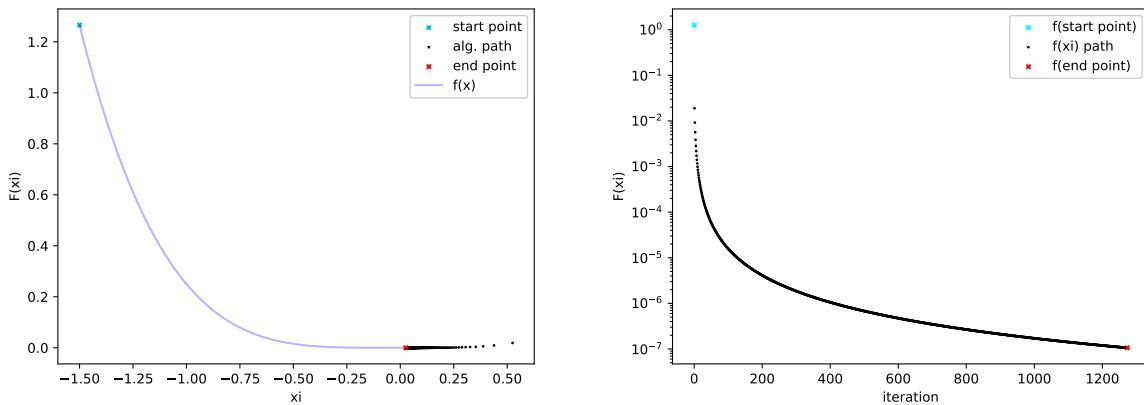
Rys. 2.2: Wykresy: trajektorii znajdowania minimum funkcji, a tak e zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = -1.5$ i początkowej d ugo ci kroku $e = 0.08$.



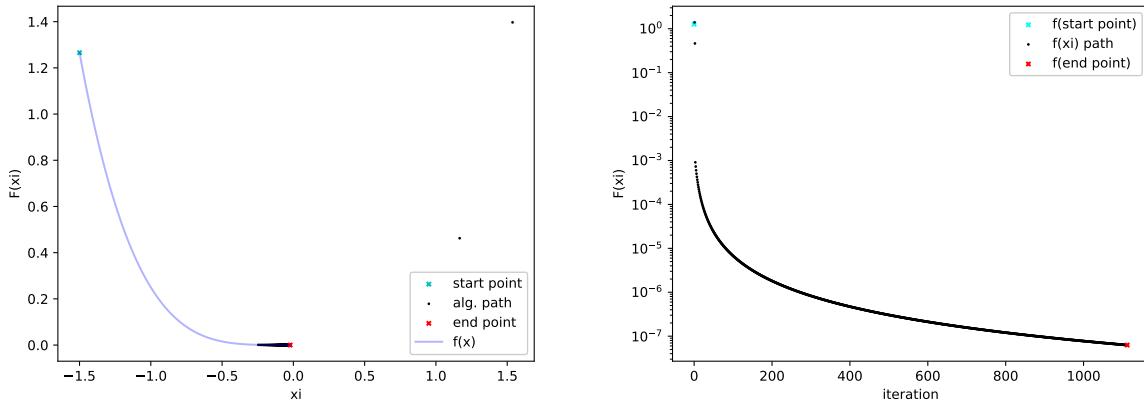
Rys. 2.3: Wykresy: trajektorii znajdowania minimum funkcji, a tak e zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = -1.5$ i początkowej d ugo ci kroku $e = 0.01$.



Rys. 2.4: Wykresy: trajektorii znajdowania minimum funkcji, a tak e zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = -1.5$ i początkowej d ugo ci kroku $e = 0.004$.



Rys. 2.5: Wykresy: trajektorii znajdowania minimum funkcji, a także zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = -1.5$ i początkowej długości kroku $e = 0.6$.

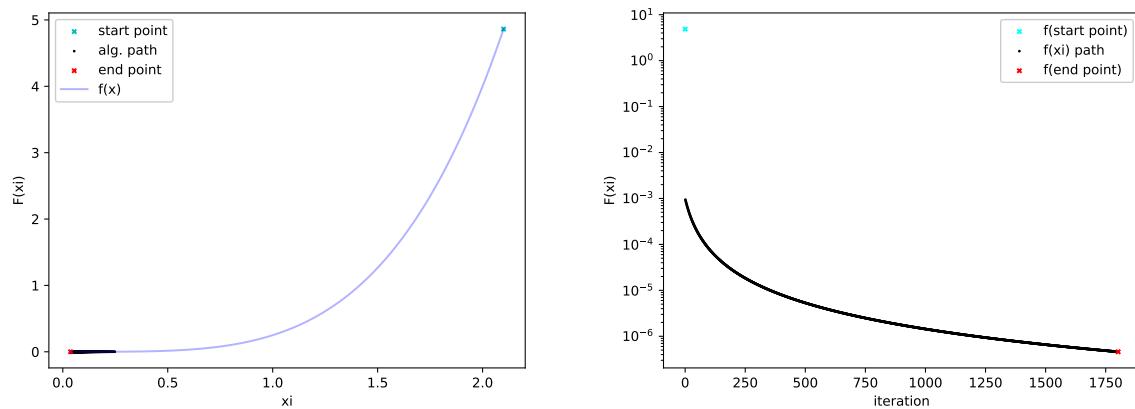


Rys. 2.6: Wykresy: trajektorii znajdowania minimum funkcji, a także zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = -1.5$ i początkowej długości kroku $e = 0.9$.

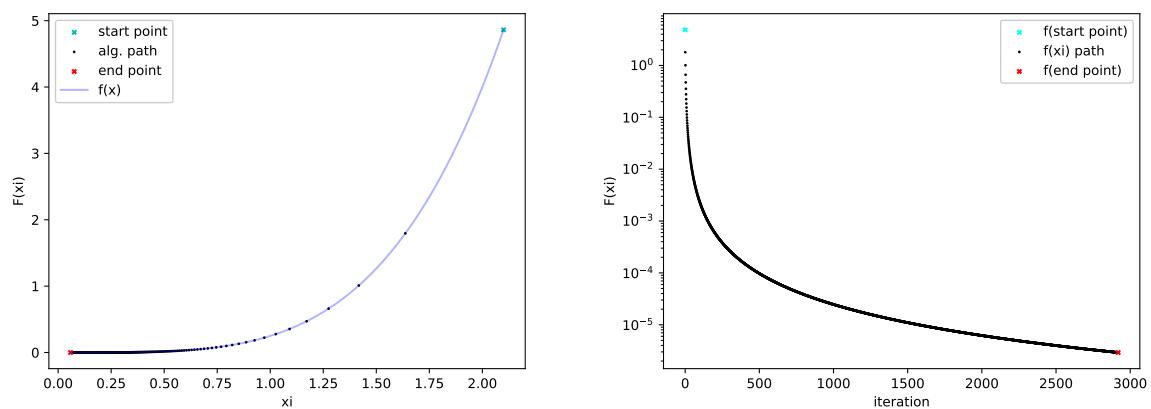
2.1.2. Testy dla $x_0=2.1$

x_0	krok e	β	ϵ	max_it	x_i	f.min(x_i)	kryt. stopu	l. iteracji	krok e
2.1	0.2	0.9	0.00001	10000	0.0368	4.6e-07	brak poprawy	1802	0.2
2.1	0.05	0.9	0.00001	10000	0.058	2.92e-06	brak poprawy	2919	0.05
2.1	0.005	0.9	0.00001	10000	0.126	6.3e-05	brak poprawy	6275	0.005
2.1	0.7	0.9	0.00001	10000	0.024	8.81e-08	brak poprawy	1216	0.61

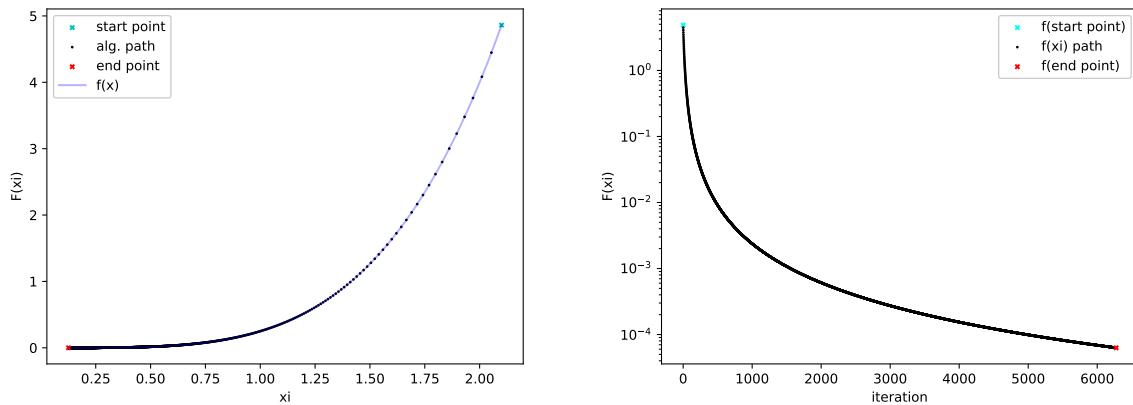
Tab. 2.2: Wyniki dla algorytmu gradientu prostego, przy zmianie długości kroku e
 $x_0 = 2.1$



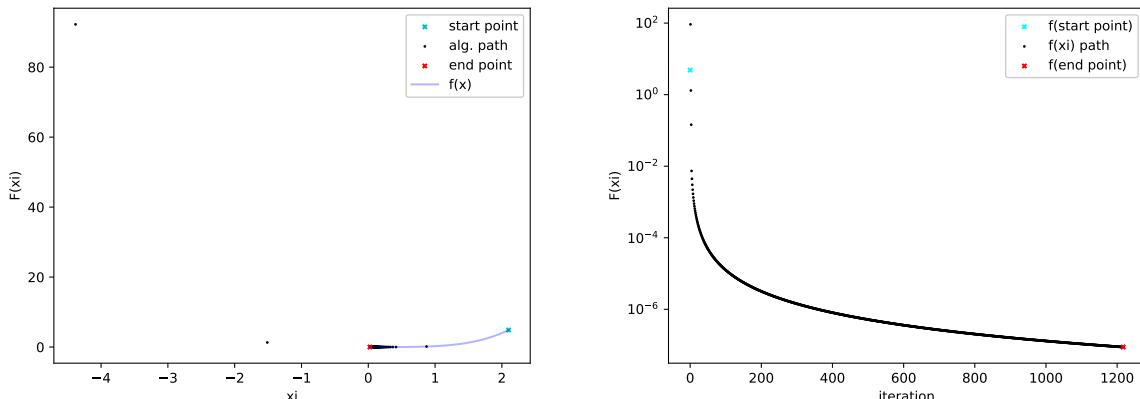
Rys. 2.7: Wykresy: trajektorii znajdowania minimum funkcji, a tak e zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = 2.1$ i początkowej d ugo ci kroku $e = 0.2$.



Rys. 2.8: Wykresy: trajektorii znajdowania minimum funkcji, a tak e zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = 2.1$ i początkowej d ugo ci kroku $e = 0.05$.



Rys. 2.9: Wykresy: trajektorii znajdowania minimum funkcji, a tak e zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = 2.1$ i początkowej d ugo ci kroku $e = 0.005$.

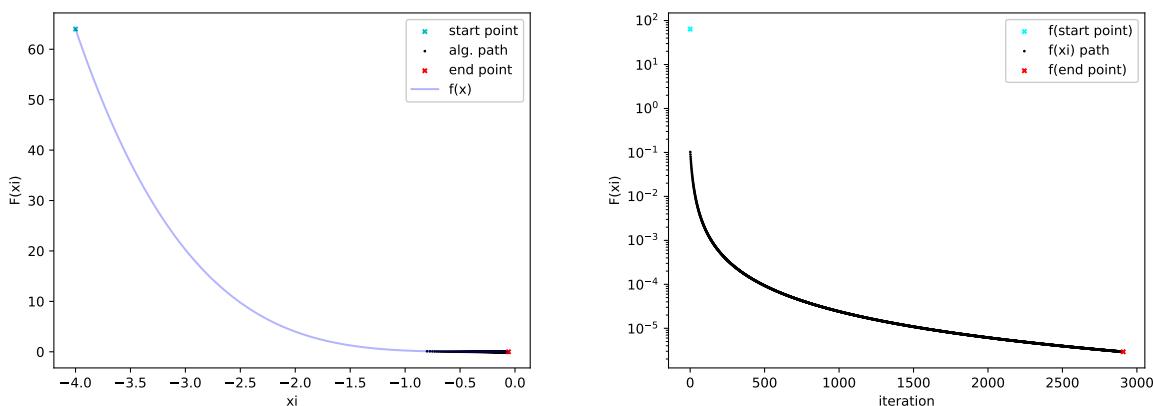


Rys. 2.10: Wykresy: trajektorii znajdowania minimum funkcji, a tak e zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = 2.1$ i początkowej d ugo ci kroku $e = 0.7$.

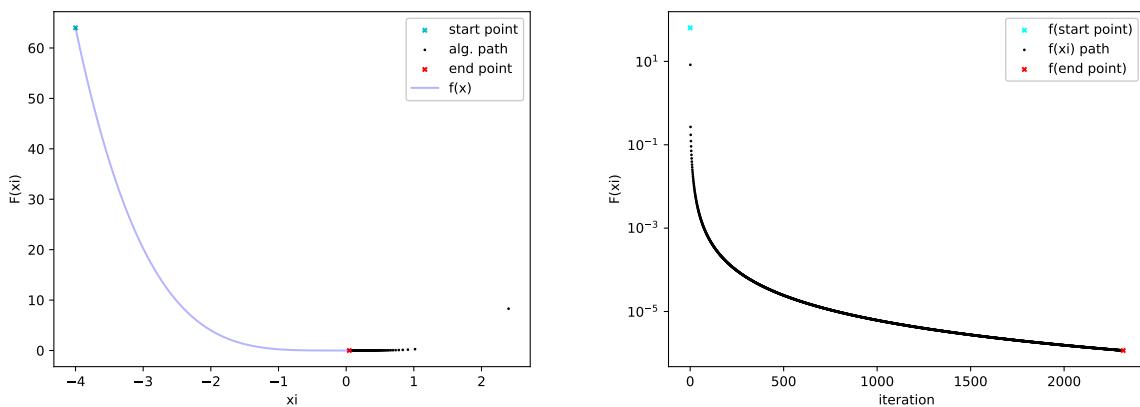
2.1.3. Testy dla $x_0=-4$

x_0	krok e	β	ϵ	max_it	x_i	$f_{\min}(x_i)$	kryt. stopu	l. iteracji	krok e
-4	0.05	0.9	0.00001	10000	-0.0585	2.92e-06	brak poprawy	2908	0.05
-4	0.1	0.9	0.00001	10000	0.046	1.16e-06	brak poprawy	2316	0.1
-4	0.002	0.9	0.00001	10000	-0.171	0.00021	brak poprawy	8532	0.002

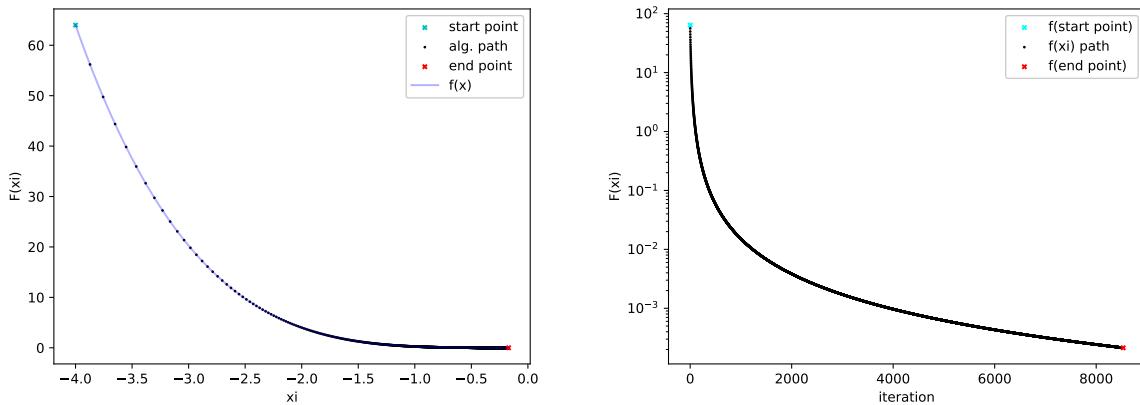
Tab. 2.3: Wyniki dla algorytmu gradientu prostego, przy zmianie d ugo ci kroku e
 $x_0 = -4$



Rys. 2.11: Wykresy: trajektorii znajdowania minimum funkcji, a także zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = -4$ i początkowej długości kroku $e = 0.05$.



Rys. 2.12: Wykresy: trajektorii znajdowania minimum funkcji, a także zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = -4$ i początkowej długości kroku $e = 0.1$.

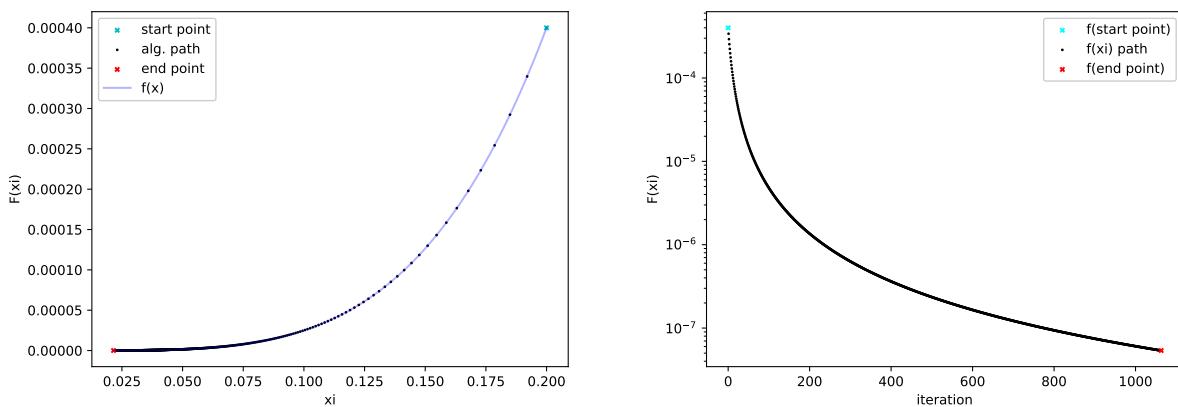


Rys. 2.13: Wykresy: trajektorii znajdowania minimum funkcji, a także zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = -4$ i początkowej długości kroku $e = 0.002$.

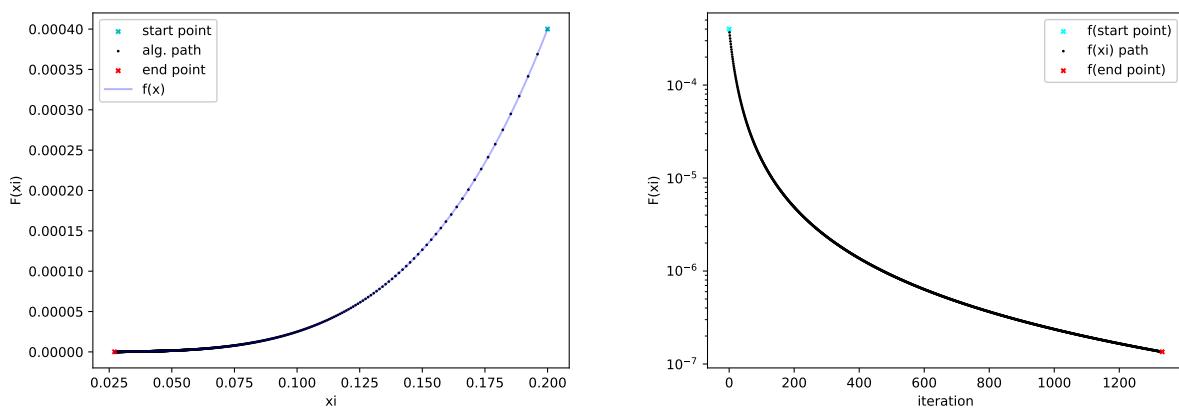
2.1.4. Testy dla $x_0=0.2$

x_0	krok e	β	ϵ	max_it	xi	$f_{\min}(xi)$	kryt. stopu	l. iteracji	krok e
0.2	1	0.9	0.00001	10000	0.022	5.38e-08	brak poprawy	1063	1
0.2	0.5	0.9	0.00001	10000	0.027	1.35e-07	brak poprawy	1332	0.5
0.2	0.08	0.9	0.00001	10000	0.05	1.56e-06	brak poprawy	2344	0.08
0.2	0.007	0.9	0.00001	10000	0.11	4.02e-05	brak poprawy	3847	0.007
0.2	2	0.9	0.00001	10000	0.021	5.39e-08	brak poprawy	530	2

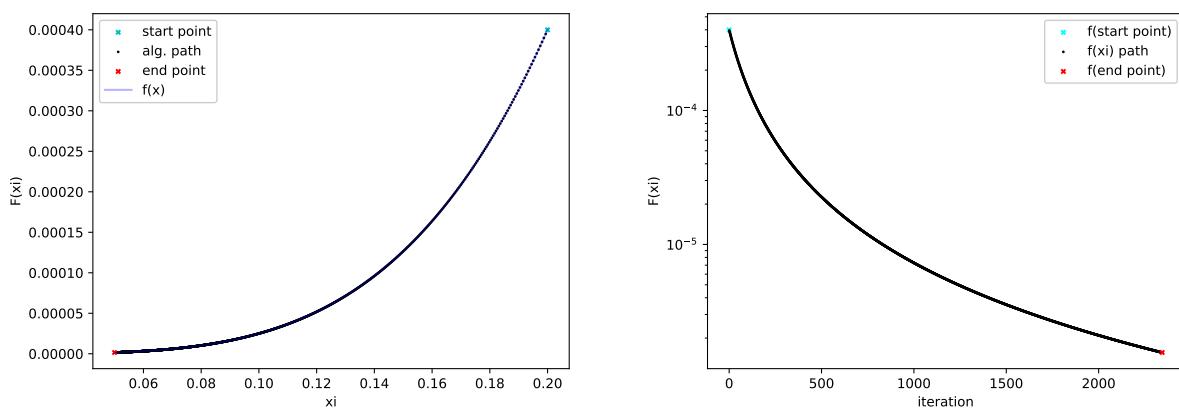
Tab. 2.4: Wyniki dla algorytmu gradientu prostego, przy zmianie długości kroku e
 $x_0 = 0.2$



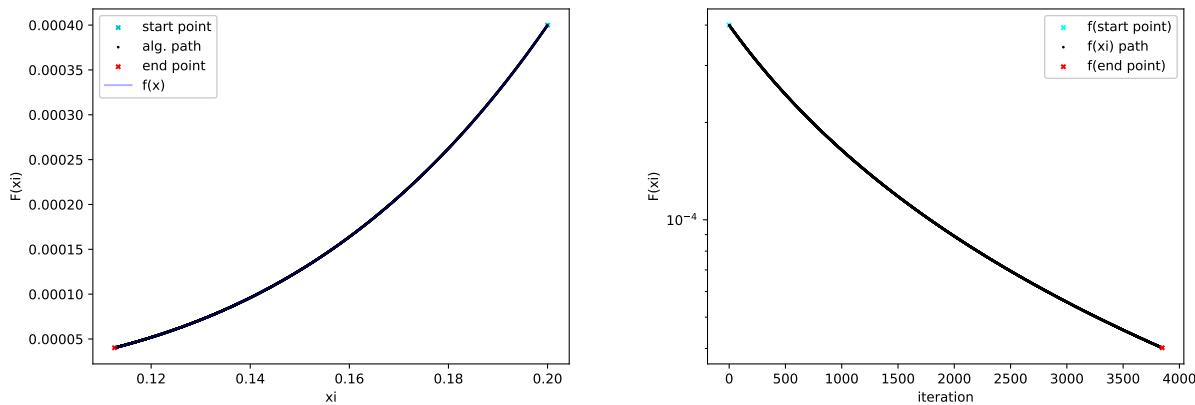
Rys. 2.14: Wykresy: trajektorii znajdowania minimum funkcji, a także zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = 0.2$ i początkowej długości kroku $e = 1$.



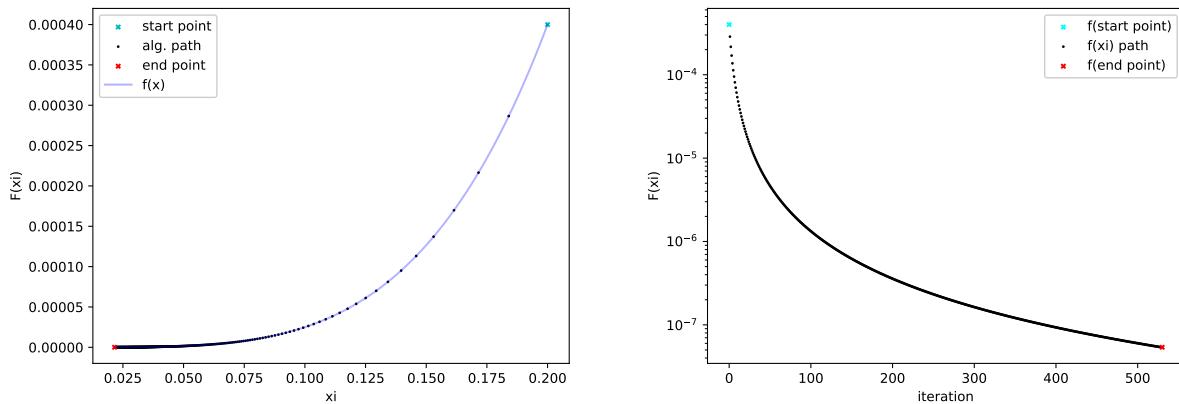
Rys. 2.15: Wykresy: trajektorii znajdowania minimum funkcji, a tak e zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = 0.2$ i początkowej d ugo ci kroku $e = 0.5$.



Rys. 2.16: Wykresy: trajektorii znajdowania minimum funkcji, a tak e zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = 0.2$ i początkowej d ugo ci kroku $e = 0.08$.



Rys. 2.17: Wykresy: trajektorii znajdowania minimum funkcji, a także zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = 0.2$ i początkowej długości kroku $e = 0.007$.



Rys. 2.18: Wykresy: trajektorii znajdowania minimum funkcji, a także zmiany wartości funkcji celu w funkcji liczby iteracji w skali logarytmicznej, dla punktu startowego $x_0 = 0.2$ i początkowej długości kroku $e = 2$.

2.1.5. Wnioski

Testy działania metody gradientu prostego wykonane dla problemu minimalizacji funkcji $f(x) = \frac{1}{4}x^4$, dla czterech różnych punktów startowych oraz zmian długości kroku e , pokazują, że w przypadku punktów **x0** rozsądnie oddalonych od okolic szukanego rozwiązania, algorytm działa szybciej i daje lepsze rezultaty wraz ze wzrostem długości kroku. Dla stosunkowo małych wartości e (np. 0.05, 0.01, 0.004) algorytm również znajduje dość zadowalające rozwiązania, lecz wartość funkcji celu jest o kilka rzędów wielkości większa niż dla najlepszego uzyskanego rozwiązania dla: $x_0=-1.5$, $e=0.9$. W przypadku punktu startowego $x_0=-1.5$ powtórzono testy dla kroków 0.01 i 0.9, przy zmniejszonej wartości tolerancji $\epsilon=0.0000001$, co zostało zaprezentowano w tabeli 2.5.

x0	krok e	β	ϵ	max_it	xi	f_min(xi)	kryt. stopu	l. iteracji	krok e
-1.5	0.01	0.9	0.0000001	10000	-0.071	6.21e-06	max iteracje	10000	0.01
-1.5	0.9	0.9	0.0000001	10000	-0.007	7.87e-10	max iteracje	10000	0.8

Tab. 2.5: Wyniki dla algorytmu gradientu prostego, przy zmianie długości kroku e $x_0 = -1.5$, przy zmniejszonej wartości tolerancji ϵ .

Tym razem widoczna jest zmiana kryterium stopu z powtarzającego się podczas głównych testów braku poprawy, na przekroczenie maksymalnej liczby iteracji. Tym samym zauważalna jest poprawa otrzymywanych wyników, lecz jeszcze bardziej uwydatnia się przewaga długości kroku 0.9.

Dla punktu startowego $x_0=0.2$, a więc znajdującego się stosunkowo blisko minimum funkcji znajdującego się w $x=0$, widać różnicę przede wszystkim w szybkości znajdowania jakościowych rozwiązań problemu, względem reszty testowanych punktów, znajdujących się dalej.

Im bardziej oddalony punkt startowy od minimum znajdującym się w punkcie $x=0$ oraz im mniejsza długość kroku e , tym dla minimalizacji funkcji jednowymiarowej algorytm działa wolniej i po większej liczbie iteracji osiąga satysfakcyjne rezultaty.

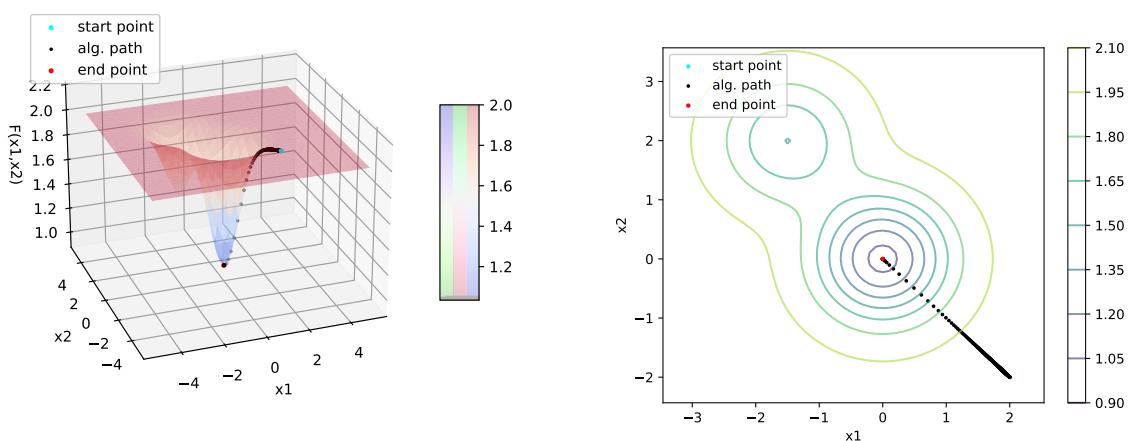
2.2. Funkcja dwuwymiarowa

Wybrano 4 punkty początkowe x_0 : $[2, -2]$, $[1, 2.5]$, $[-1, -1.5]$, $[0.2, 0.6]$, $[-2.3, 1.8]$ dla których przeprowadzono eksperymenty zmiany długości kroku.

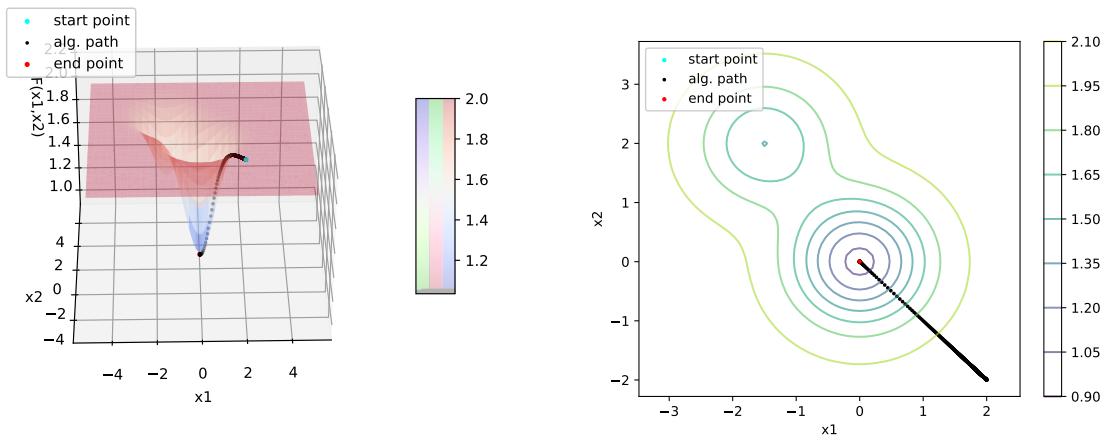
2.2.1. Testy dla $x_0=[2, -2]$

x0	krok e	β	ϵ	max_it	xi	f_min(xi)	kryt. stopu	l. iteracji	krok e
[2,-2]	0.2	0.9	1e-06	10000	[-0.00146, 0.002]	0.999	brak poprawy	581	0.2
[2,-2]	0.08	0.9	1e-06	10000	[-0.00146, 0.002]	0.999	brak poprawy	1450	0.08
[2,-2]	0.01	0.9	1e-06	10000	[1.582, -1.582]	1.99	maks iteracje	10000	0.01
[2,-2]	0.8	0.9	1e-06	10000	[-0.00146, 0.002]	0.999	gradient bliski zeru	2494	0.8

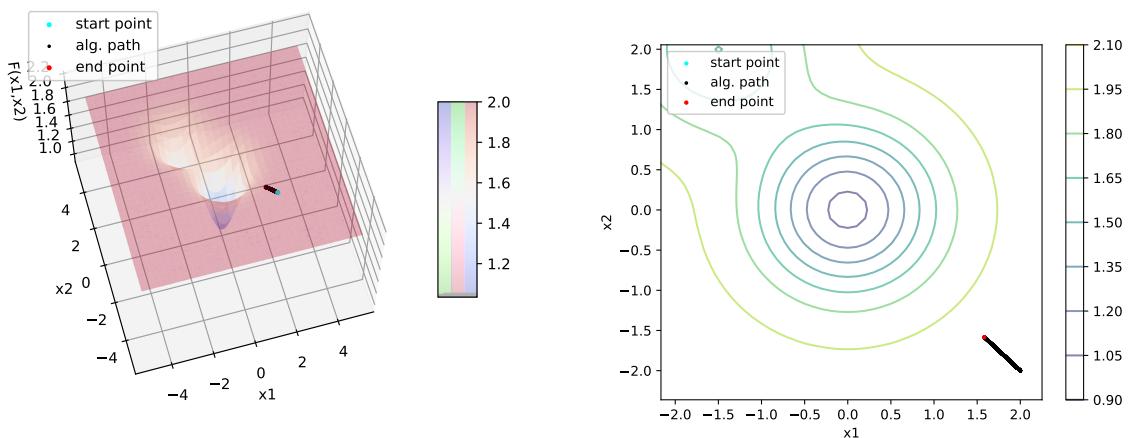
Tab. 2.6: Wyniki dla algorytmu gradientu prostego, przy zmianie długości kroku e $x_0 = [2, -2]$



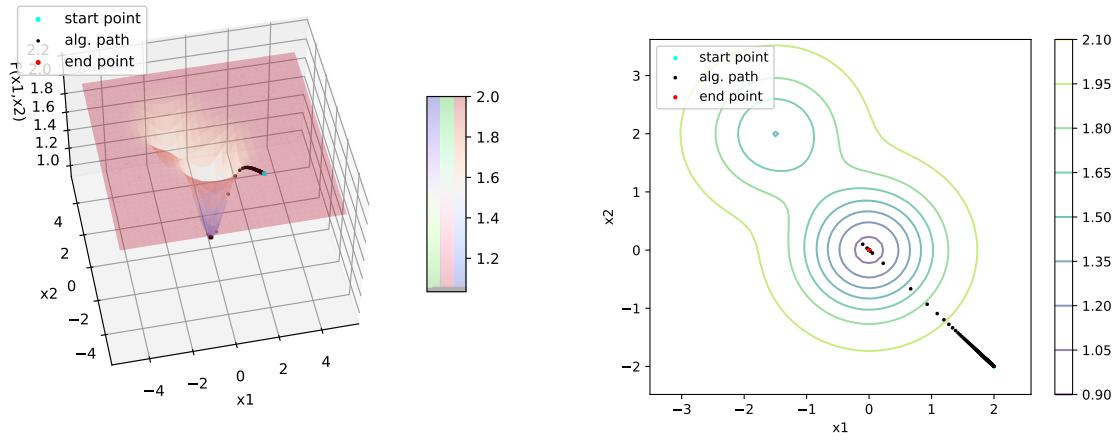
Rys. 2.19: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [2, -2]$ i początkowej długości kroku $e = 0.2$.



Rys. 2.20: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [2, -2]$ i początkowej długości kroku $e = 0.08$.



Rys. 2.21: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [2, -2]$ i początkowej długości kroku $e = 0.01$.

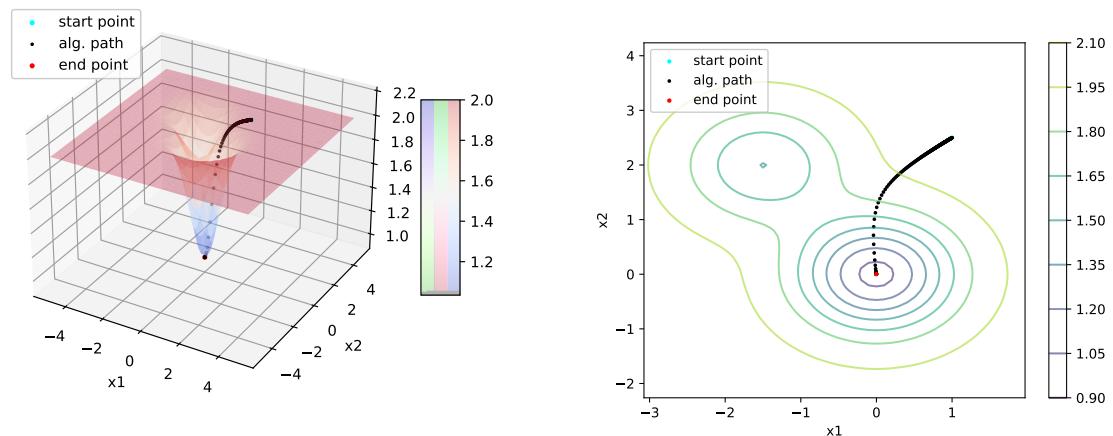


Rys. 2.22: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [2, -2]$ i początkowej długości kroku $e = 0.8$.

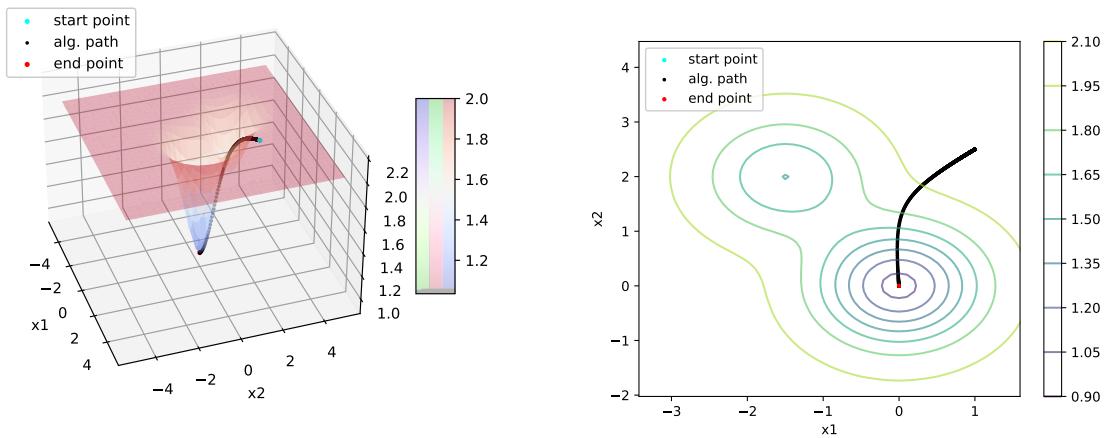
2.2.2. Testy dla $x_0=[1,2.5]$

x_0	krok e	β	ϵ	max_it	x_i	$f_{\min}(x_i)$	kryt. stopu	l. iteracji	krok e
[1,2.5]	0.2	0.9	1e-06	10000	[-0.00147, 0.002]	0.999	brak poprawy	242	0.2
[1,2.5]	0.05	0.9	1e-06	10000	[-0.00147, 0.002]	0.999	brak poprawy	965	0.05
[1,2.5]	0.01	0.9	1e-06	10000	[-0.00147, 0.002]	0.999	brak poprawy	4746	0.01
[1,2.5]	0.005	0.9	1e-06	10000	[-0.00147, 0.0021]	0.999	brak poprawy	9422	0.005
[1,2.5]	0.9	0.9	1e-06	10000	[-0.00146, 0.002]	0.999	brak poprawy	102	0.9

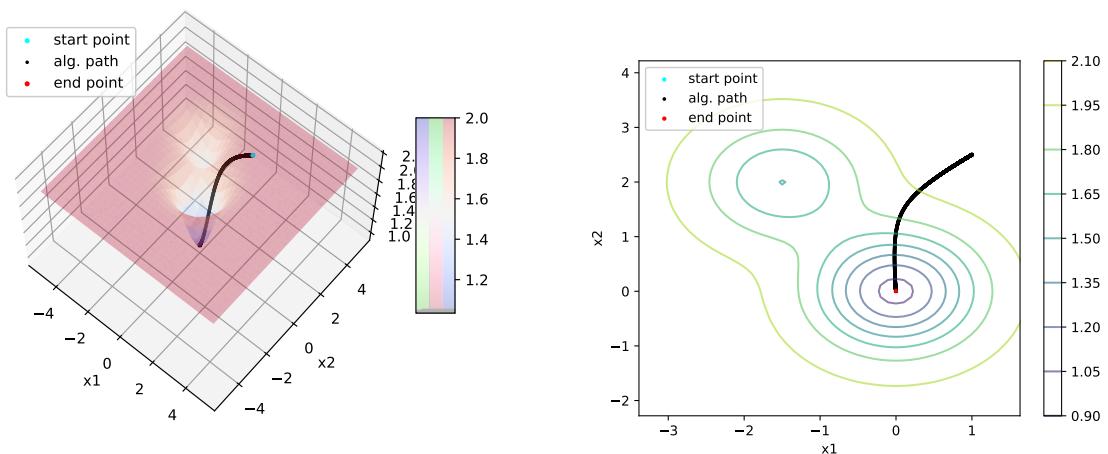
Tab. 2.7: Wyniki dla algorytmu gradientu prostego, przy zmianie długości kroku e
 $x_0 = [1,2.5]$



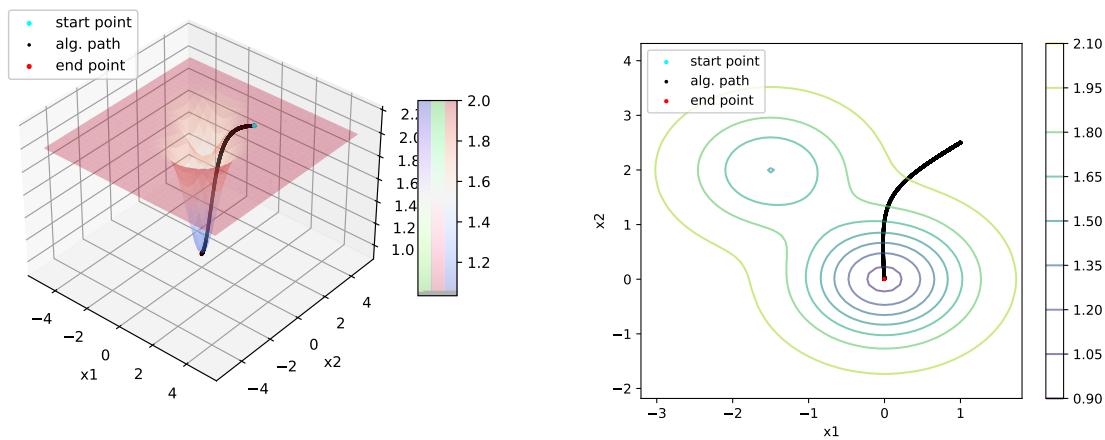
Rys. 2.23: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [1, 2.5]$ i początkowej długości kroku $e = 0.2$.



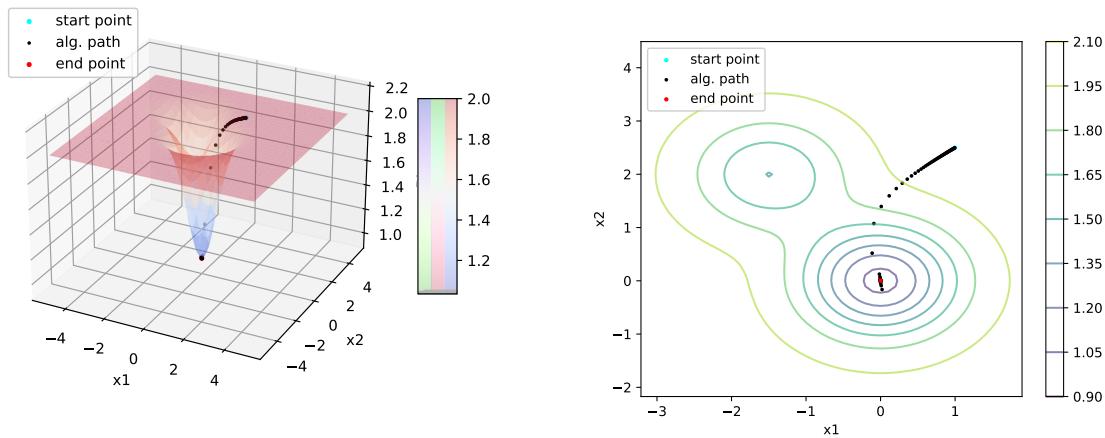
Rys. 2.24: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [1, 2.5]$ i początkowej długości kroku $e = 0.05$.



Rys. 2.25: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [1, 2.5]$ i początkowej długości kroku $e = 0.01$.



Rys. 2.26: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [1, 2.5]$ i początkowej długości kroku $e = 0.005$.

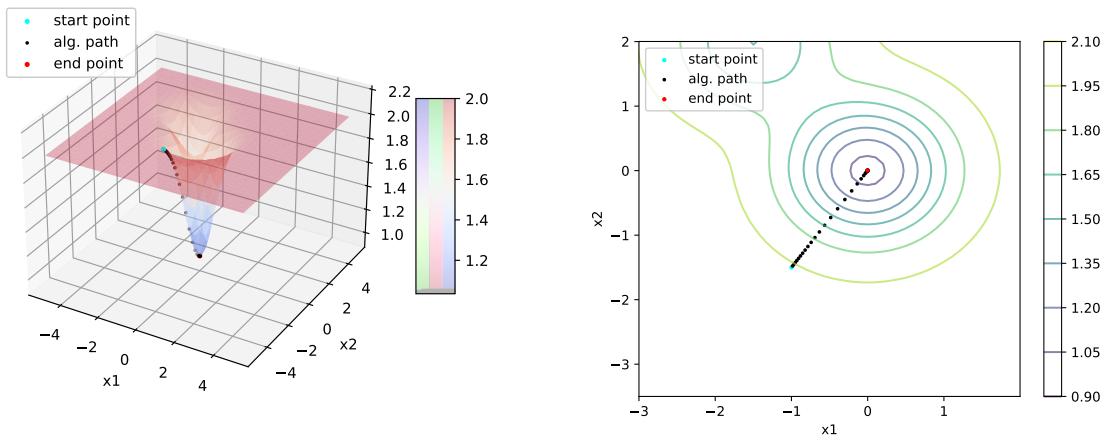


Rys. 2.27: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [1, 2.5]$ i początkowej długości kroku $e = 0.9$.

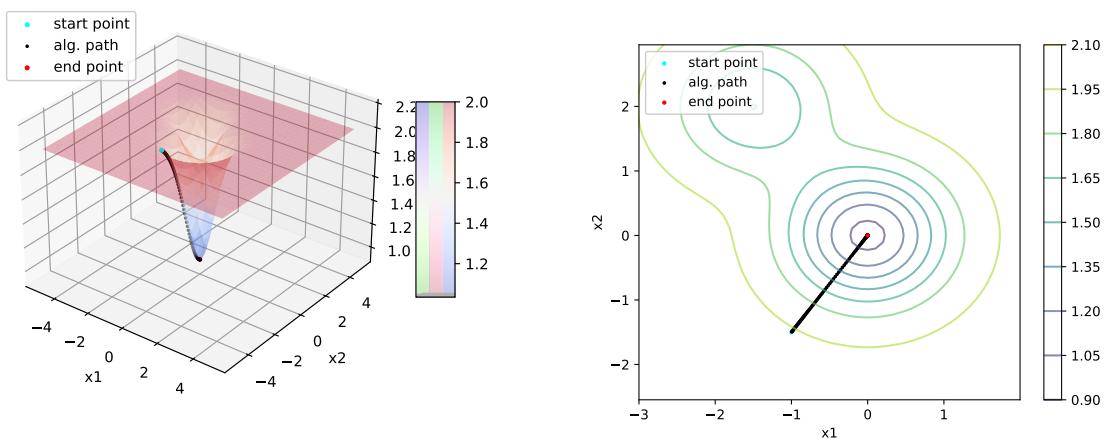
2.2.3. Testy dla $x_0=[-1,-1.5]$

x_0	krok e	β	ϵ	max_it	xi	$f_{\min}(xi)$	kryt. stopu	l. iteracji	krok e
[-1,-1.5]	0.2	0.9	1e-06	10000	[-0.00146, 0.002]	0.999	brak poprawy	43	0.2
[-1,-1.5]	0.05	0.9	1e-06	10000	[-0.00147, 0.0019]	0.999	brak poprawy	169	0.05
[-1,-1.5]	0.005	0.9	1e-06	10000	[-0.0015, 0.00187]	0.999	brak poprawy	1480	0.005
[-1,-1.5]	0.001	0.9	1e-06	10000	[-0.0017, 0.0015]	0.999	brak poprawy	6597	0.001
[-1,-1.5]	0.9	0.9	1e-06	10000	[-0.00146, 0.002]	0.999	gradient bliski zeru	58	0.9

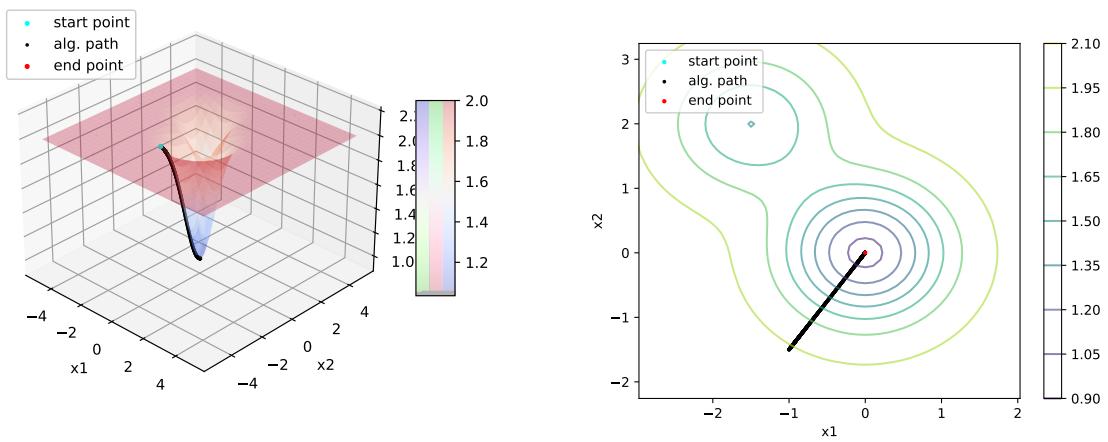
Tab. 2.8: Wyniki dla algorytmu gradientu prostego, przy zmianie długości kroku e
 $x_0 = [-1,-1.5]$



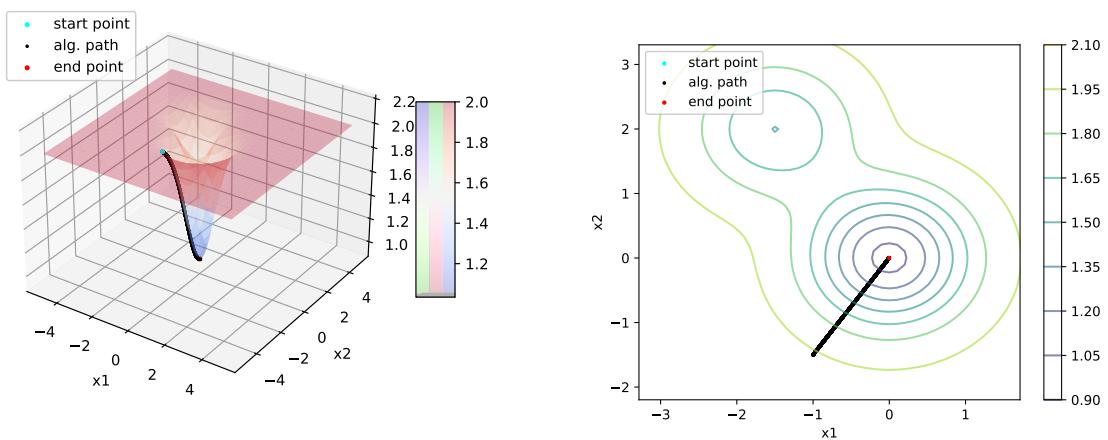
Rys. 2.28: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [-1, -1.5]$ i początkowej długości kroku $e = 0.2$.



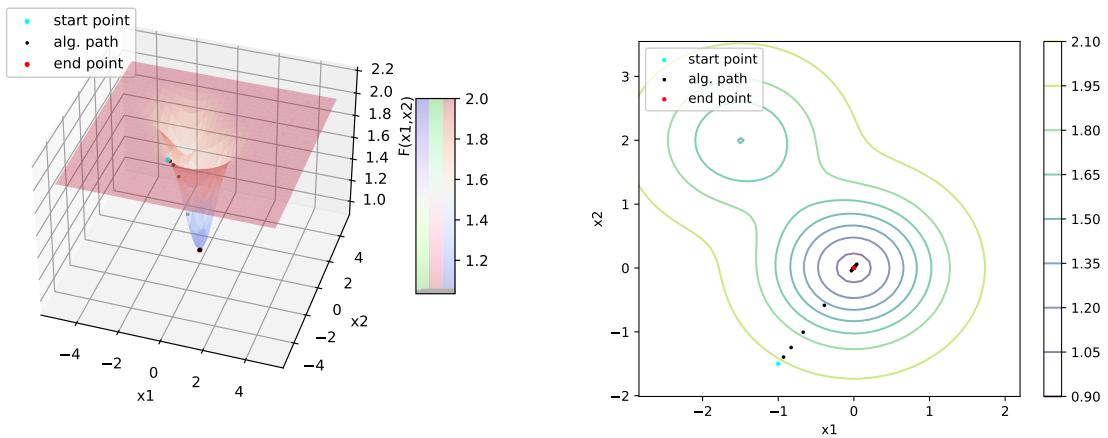
Rys. 2.29: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [-1, -1.5]$ i początkowej długości kroku $e = 0.05$.



Rys. 2.30: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [-1, -1.5]$ i początkowej długości kroku $e = 0.01$.



Rys. 2.31: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [-1, -1.5]$ i początkowej długości kroku $e = 0.001$.

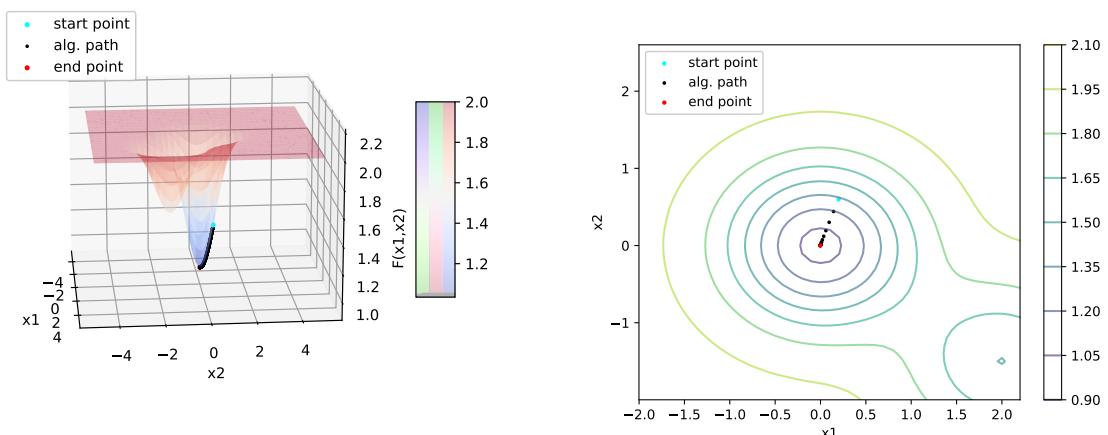


Rys. 2.32: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [-1, -1.5]$ i początkowej długości kroku $e = 0.9$.

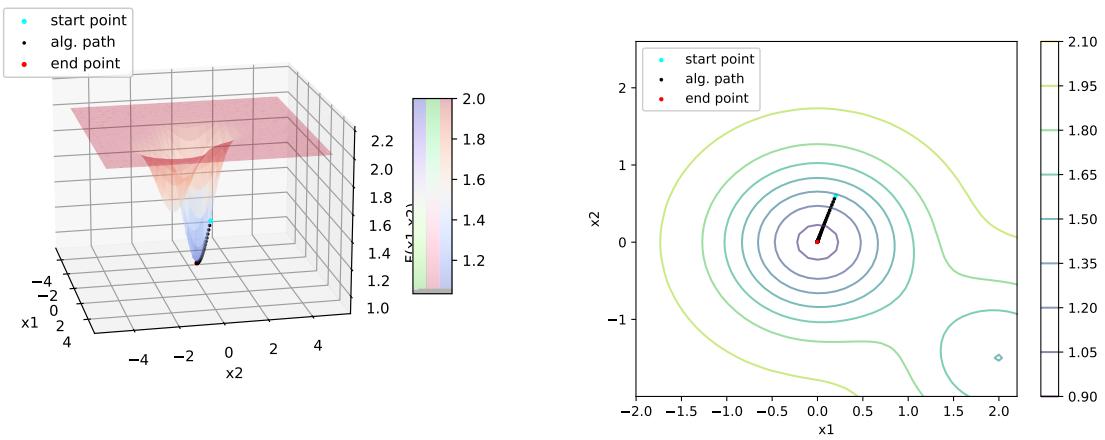
2.2.4. Testy dla $x_0=[0.2,0.6]$

x_0	krok e	β	ϵ	max_it	x_i	$f_{\min}(x_i)$	kryt. stopu	l. iteracji	krok e
[0.2,0.6]	0.2	0.9	1e-06	10000	[-0.00146, 0.002]	0.999	brak poprawy	27	0.2
[0.2,0.6]	0.05	0.9	1e-06	10000	[-0.00147, 0.0019]	0.999	brak poprawy	110	0.05
[0.2,0.6]	0.001	0.9	1e-06	10000	[-0.0013, 0.0024]	0.999	brak poprawy	3692	0.001
[0.2,0.6]	0.9	0.9	1e-06	10000	[-0.00146, 0.002]	0.999	gradient bliski zeru	56	0.9

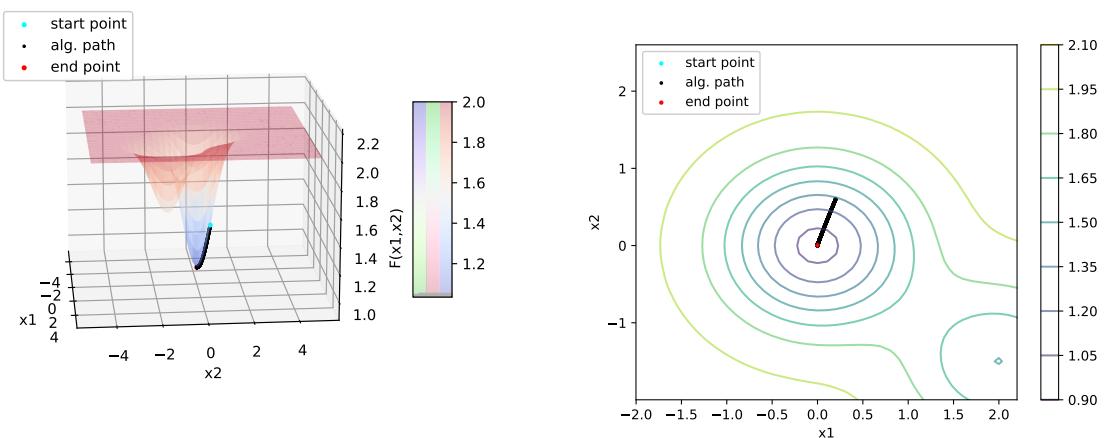
Tab. 2.9: Wyniki dla algorytmu gradientu prostego, przy zmianie długości kroku e
 $x_0 = [0.2,0.6]$



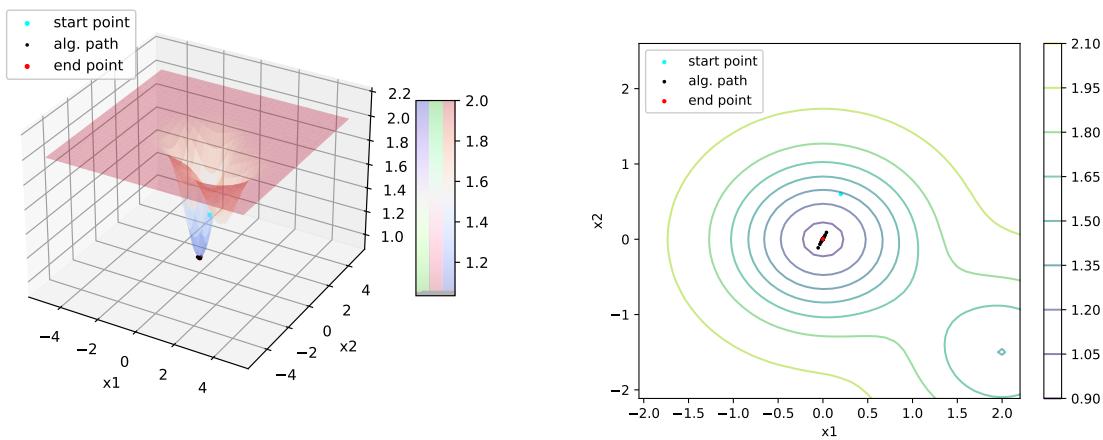
Rys. 2.33: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [0.2, 0.6]$ i początkowej długości kroku $e = 0.2$.



Rys. 2.34: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [0.2, 0.6]$ i początkowej długości kroku $e = 0.05$.



Rys. 2.35: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [0.2, 0.6]$ i początkowej długości kroku $e = 0.001$.

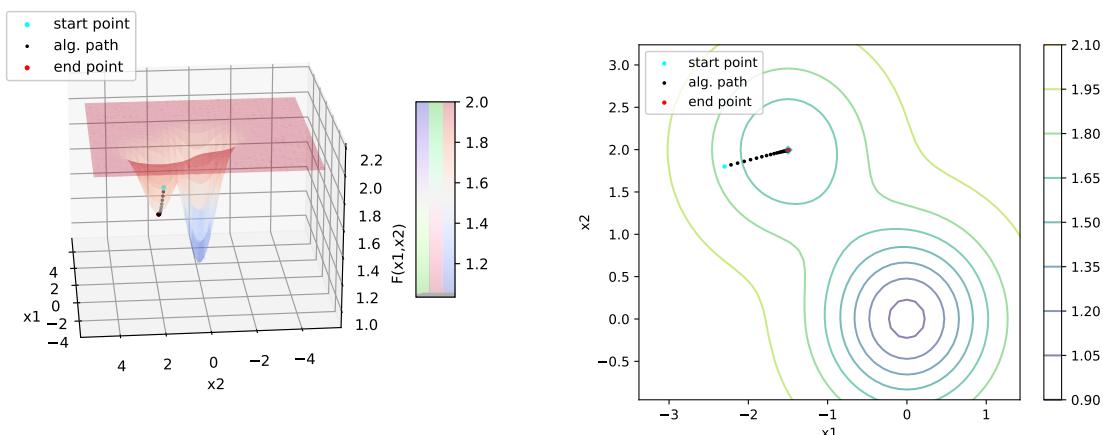


Rys. 2.36: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [0.2, 0.6]$ i początkowej długości kroku $e = 0.9$.

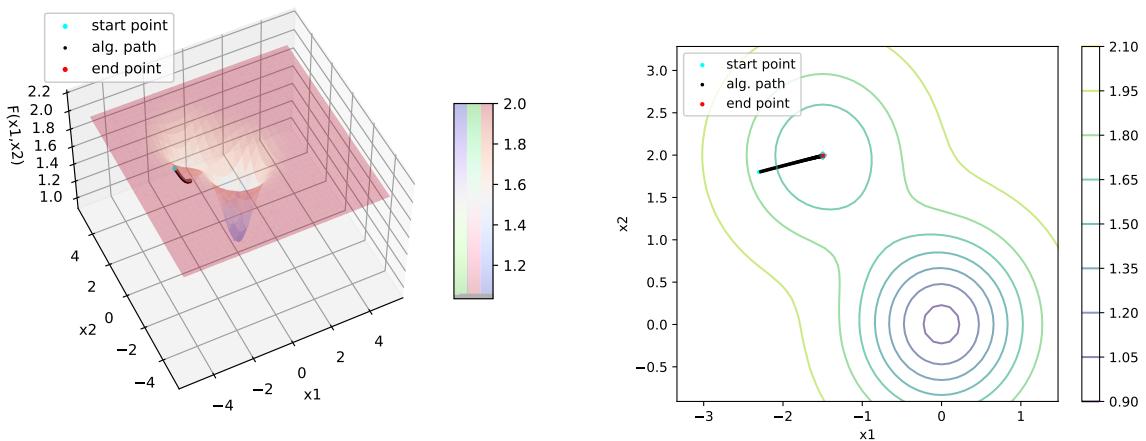
2.2.5. Testy dla $x_0=[-2.3,1.8]$

x_0	krok e	β	ϵ	max_it	x_i	$f_{\min}(x_i)$	kryt. stopu	l. iteracji	krok e
$[-2.3,1.8]$	0.2	0.9	1e-06	10000	$[-1.494, 1.99]$	1.49	brak poprawy	50	0.2
$[-2.3,1.8]$	0.05	0.9	1e-06	10000	$[-1.494, 1.99]$	1.498	brak poprawy	215	0.05

Tab. 2.10: Wyniki dla algorytmu gradientu prostego, przy zmianie długości kroku e
 $x_0 = [-2.3,1.8]$



Rys. 2.37: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [-2.3, 1.8]$ i początkowej długości kroku $e = 0.2$.



Rys. 2.38: Wykresy: trajektorii znajdowania minimum funkcji w trójwymiarze oraz na mapie konturowej dla punktu startowego $x_0 = [-2.3, 1.8]$ i początkowej długości kroku $e = 0.2$.

2.2.6. Wnioski

Metodę gradientu prostego przetestowano również dla problemu minimalizacji funkcji dwóch zmiennych $g(x) = 2 - \exp(-x_1^2 - x_2^2) - 0.5 \exp(-(x_1 + 1.5)^2 - (x_2 - 2)^2)$, dla czterech różnych punktów startowych oraz zmian długości kroku e . Podobnie jak w przypadku funkcji jednowymiarowej, wyniki eksperymentów pokazują, że algorytm działa najlepiej, a więc najszybciej zbiega do satysfakcjonujących przybliżeń minimum funkcji, dla długości kroku e rzędu części dziesiętnych. Zauważalna jest również różnica w liczbie iteracji, po której algorytm jest zatrzymany, zależna od odległości punktu startowego od współrzędnych punktu minimum. Przeprowadzono testy dla punktu startowego znajdującego się blisko punktu [0,0] - dla $x_0=[0.2, 0.6]$. Wyniki widoczne w tabeli 2.9, dla $e = 0.2$ i $e = 0.9$ pokazują, jak szybko w tych przypadkach zachodziła minimalizacja.

Natomiast dla punktu startowego [2,-2], a więc dosyć oddalonego od współrzędnych minimum, lecz nadal rozsądnie, dla małego kroku tj. $e = 0.01$ algorytm przeszedł przez maksymalną liczbę iteracji i nadal znajdował się daleko od minimum, co jest wyraźnie widoczne na rysunku 2.21. Z kolei, dla x_0 znajdujących się bliżej minimum, algorytm radził sobie nawet z bardzo małymi długościami kroku, przykładowo dla $x_0=[-1, -1.5]$ i $e = 0.001$, odnaleziono zadowalające rozwiązanie po 6597 iteracjach.

Szczególnym przypadkiem są testy wykonane dla punktu $x_0=[-2.3, 1.8]$ (Rysunki: 2.37 i 2.38), a więc takiego, znajdującego się blisko minimum lokalnego, lecz nie globalnego funkcji. Algorytm dla tego punktu znajduje minimum w okolicy [-1.5, 2] i nie trafia do minimum globalnego przeszukiwanej funkcji.