WSI Metoda Najszybszego Wzrostu

Konrad Karpiuk, 11.03.2024

1. Treść zadania

Zaimplementować metodę najszybszego wzrostu. Zastosować metodę do znalezienia optimum funkcji booth:

$$f(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$$

w 2 wymiarach. Oraz dla funkcji o numerach od 1 do 3 z CEC 2017 w 10 wymiarach.

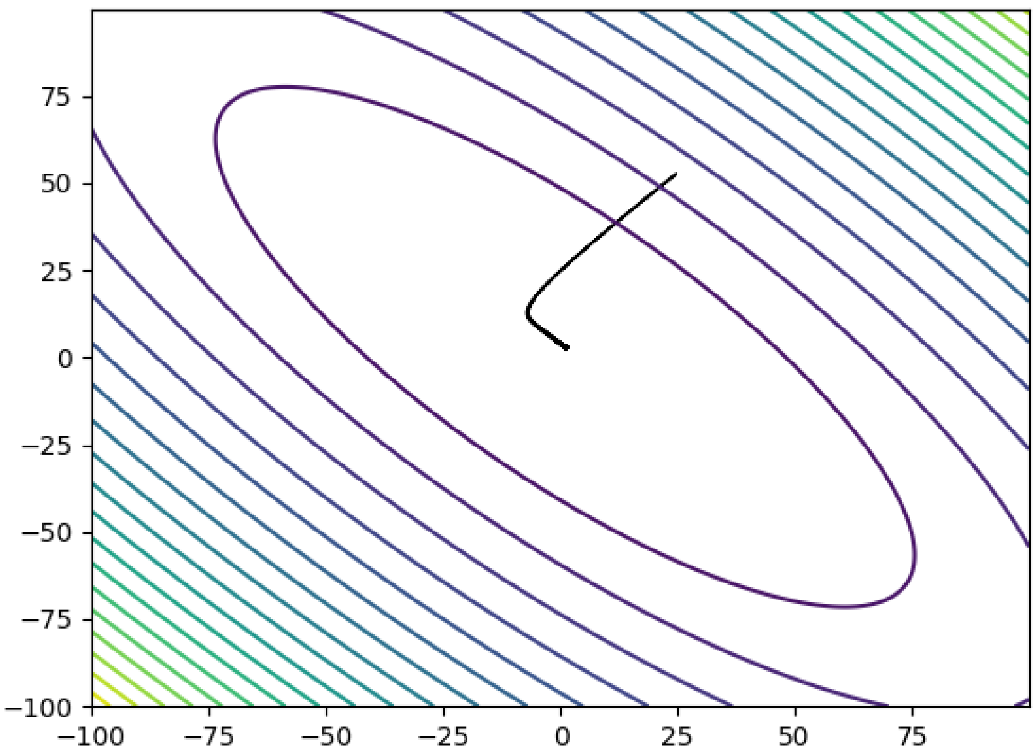
Wszystkie wykresy narysowałem w płaszczyźnie rozpiętej przez elementy X1 i X2 wektora X.

2. Funkcja booth

Wyniki dla funkcji booth:

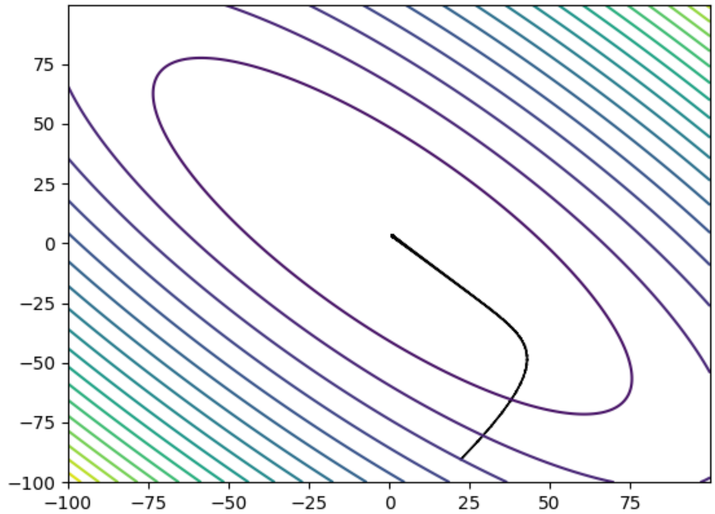
Punkt 1:

	X1	X2	F(X)
start	24,76	52,65	24582,62
end	1.00	3.00	2,40*e-17



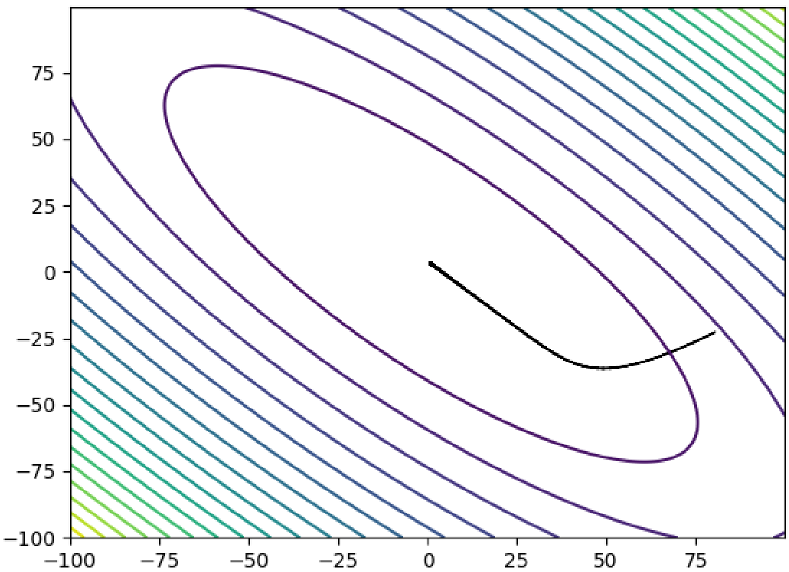
Punkt 2:

	X1	X2	F(X)
start	22,15	-90,63	22225,33
end	1.00	3.00	2,47*e-17



Punkt 3:

	X1	X2	F(X)
start	80,44	-22,71	18517,59
end	1.00	3.00	2,44*e-17



Komentarz do wykresów:

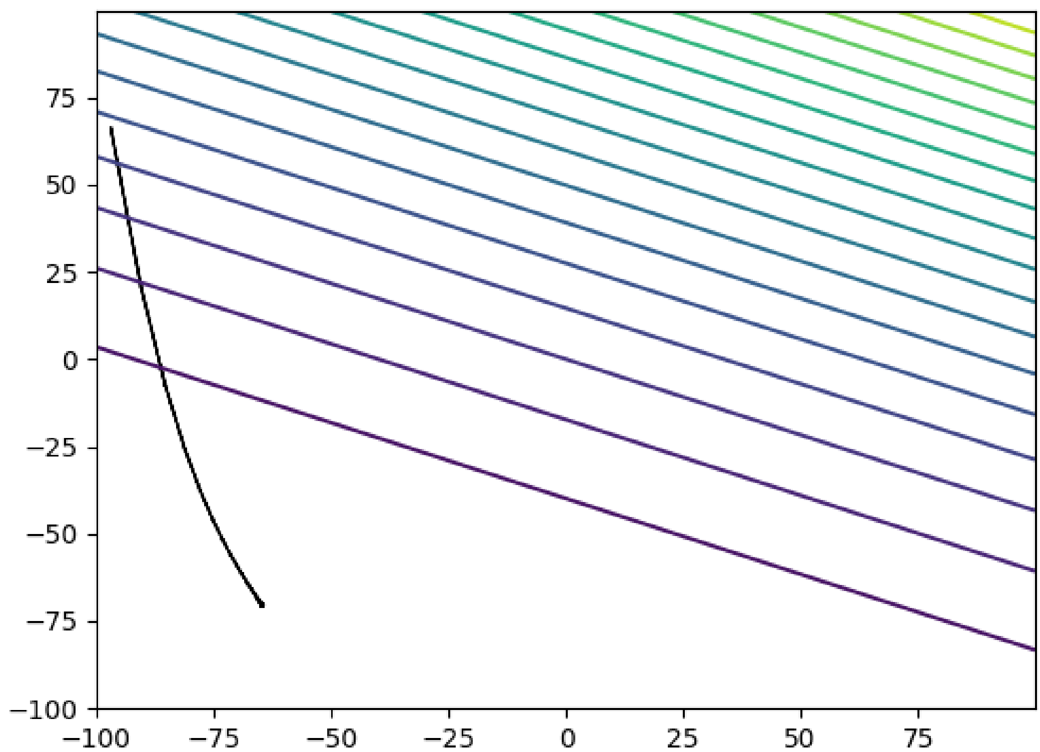
Osiągnięte minimum: 2,47*e-17, co w przybliżeniu daje 0. Minimum to osiągnięto dla wszystkich trzech sprawdzonych punktów. Parametr beta został ustawiony na 0,01. Pozwoliło to na osiągnięcie minimum po ok. 1100 iteracji. Algorytm zachowuje się zgodnie z oczekiwaniami – w kolejnych iteracjach wybiera współrzędne, dla których wartość funkcji jest coraz mniejsza.

3. Funkcja f1 z CEC 2017

Parametr beta użyty dla funkcji f1 = 1e-8
Liczba iteracji = 20 000

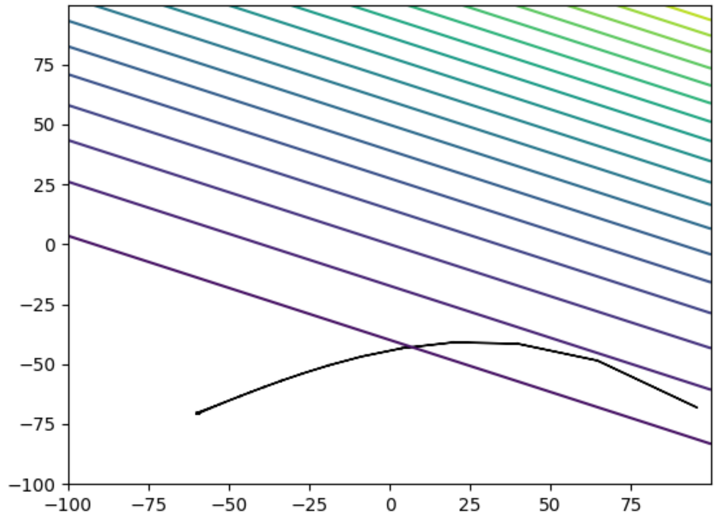
Punkt 1:

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	F(X)
start	-96,94	66,33	0,65	84,03	71,04	37,84	69,00	-33,79	68,65	-3,73	6*e+11
end	-64,74	-70,43	-29,61	-58,33	22,09	59,94	20,85	18,56	76,68	-24,18	347,78



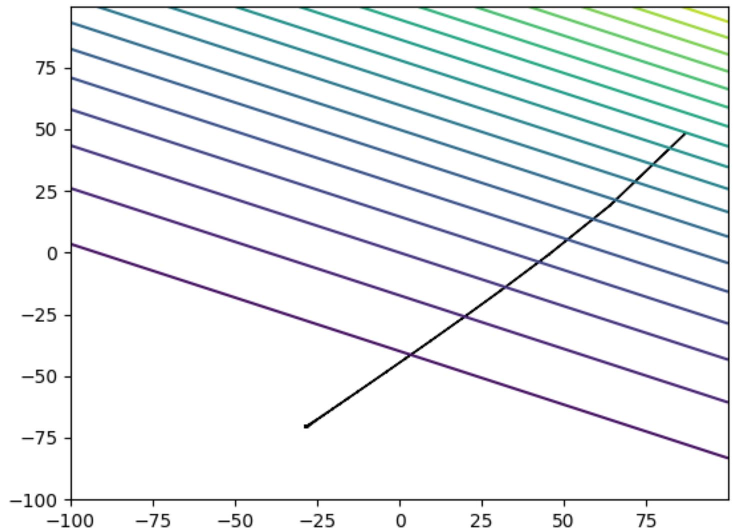
Punkt 2:

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	F(X)
start	95,65	-68,16	83,37	-58,79	21,89	27,99	-27,73	13,60	-59,37	90,37	9*e+11
end	-59,71	-70,43	-29,61	-58,33	22,09	59,94	26,02	18,56	76,68	-28,43	154,36



Punkt 3:

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	F(X)
start	87,03	48,43	29,61	-0,39	81,64	-41,39	64,51	-38,08	-50,35	88,93	1*e+12
end	-28,15	-70,43	-29,61	-58,33	22,09	59,94	58,43	18,56	76,68	-55,04	2134,94



Komentarz do wykresów:

Dla dwóch pierwszych punktów znaleziono bardzo podobne minimum, mimo rozpoczęcia pracy algorytmu dla X1 i X2 znajdujących się po drugiej stronie przedziału wyznaczającego ograniczenia kostkowe. W punkcie trzecim algorytm zbiegł do innego punktu, ale również udało się odnaleźć o wiele niższą wartość funkcji niż w punkcie początkowym. Najniższa znaleziona wartość funkcji = 154,36 dla punktu 2.

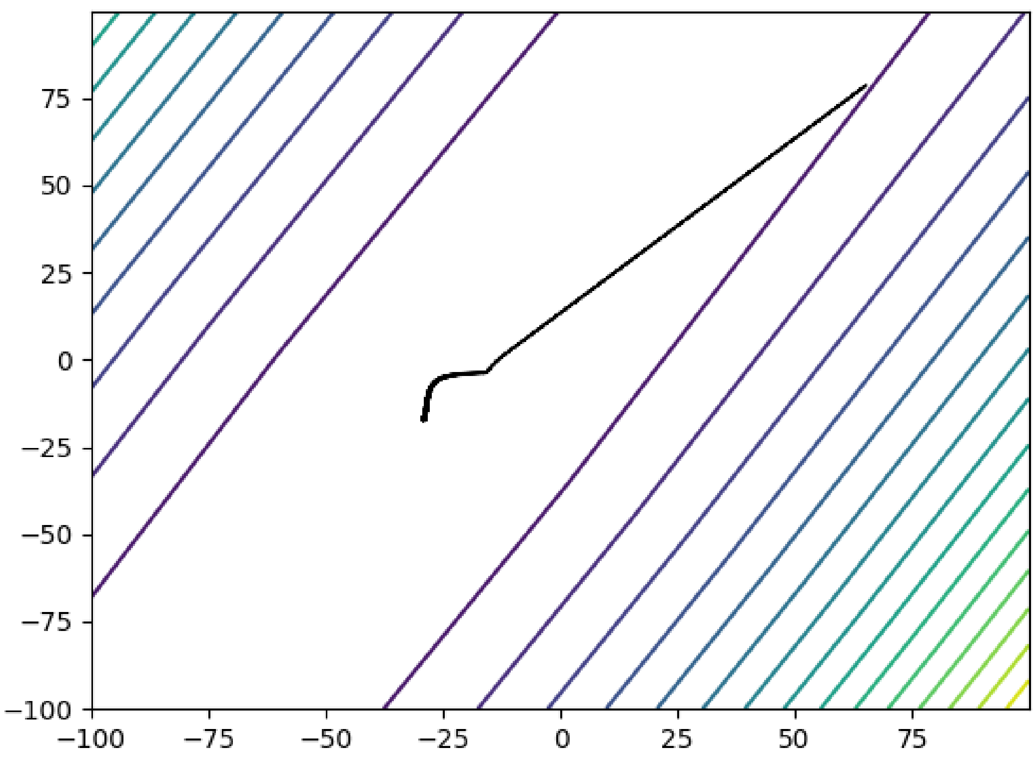
4. Funkcja f2 z CEC 2017

Funkcja f2 okazała się być bardzo trudna do minimalizacji ze względu duże wartości gradientu. Przy małym parametrze beta algorytm bardzo nieznacznie zmieniał wartości X, natomiast przy dużym parametrze beta funkcja natychmiast rozbiegała do wartości granicznych. Dlatego zastosowałem duży parametr beta z jednoczesnym ograniczeniem maksymalnej zmiany wartości X w każdym wymiarze o 1, co pozwoliło na znalezienie minimum.

Parametr beta = 0,001
Liczba iteracji = 10 000

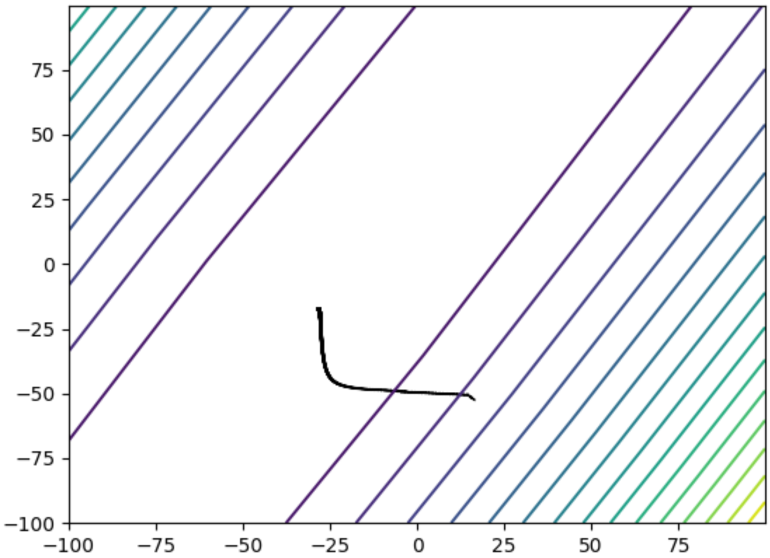
Punkt 1:

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	F(X)
start	65,22	78,60	-62,40	6,83	-80,46	-39,21	55,38	61,61	32,45	57,99	1,82*e+21
end	-29,17	-16,99	-71,78	-0,60	4,09	-37,66	43,66	74,58	-62,85	71,90	203,74



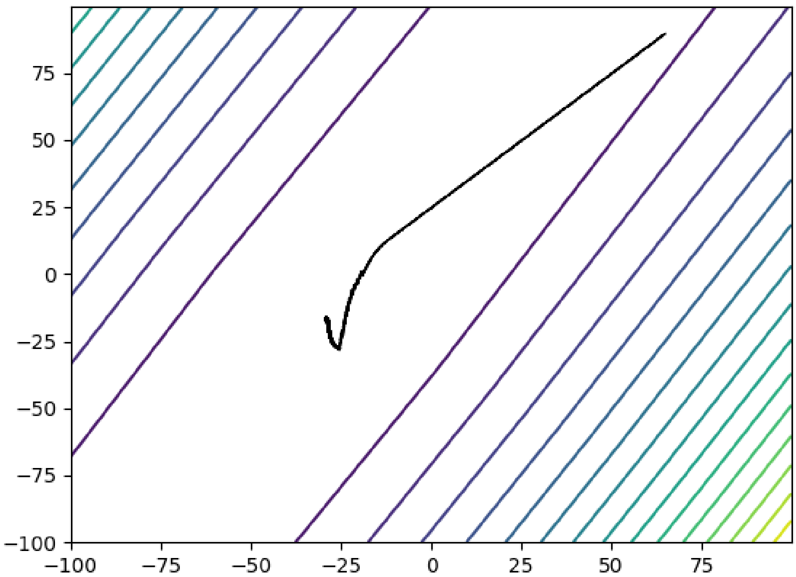
Punkt 2:

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	F(X)
start	16,55	-52,54	92,61	-70,01	28,00	-36,07	24,00	12,29	-39,76	66,29	2*e+10
end	-29,17	-16,99	-71,78	-0,60	4,09	-37,66	43,66	74,58	-62,85	71,90	328,43



Punkt 3:

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	F(X)
start	64,95	89,66	-39,56	-71,30	26,22	-34,27	49,20	43,35	-67,61	-82,42	3,06*e+2 1
end	-29,01	-16,72	-52,97	-0,86	4,17	-36,76	45,80	74,55	-62,77	71,11	222,70



Komentarz do wykresów:

Dla każdego punktu algorytm skutecznie odnalazł punkt o wartości dużo niższej niż dla punktu początkowego oraz wartości we wszystkich współrzędnych końcowego wektora X w każdym punkcie są podobne.

Najniższa znaleziona wartość funkcji = 203,74 dla punktu 1.

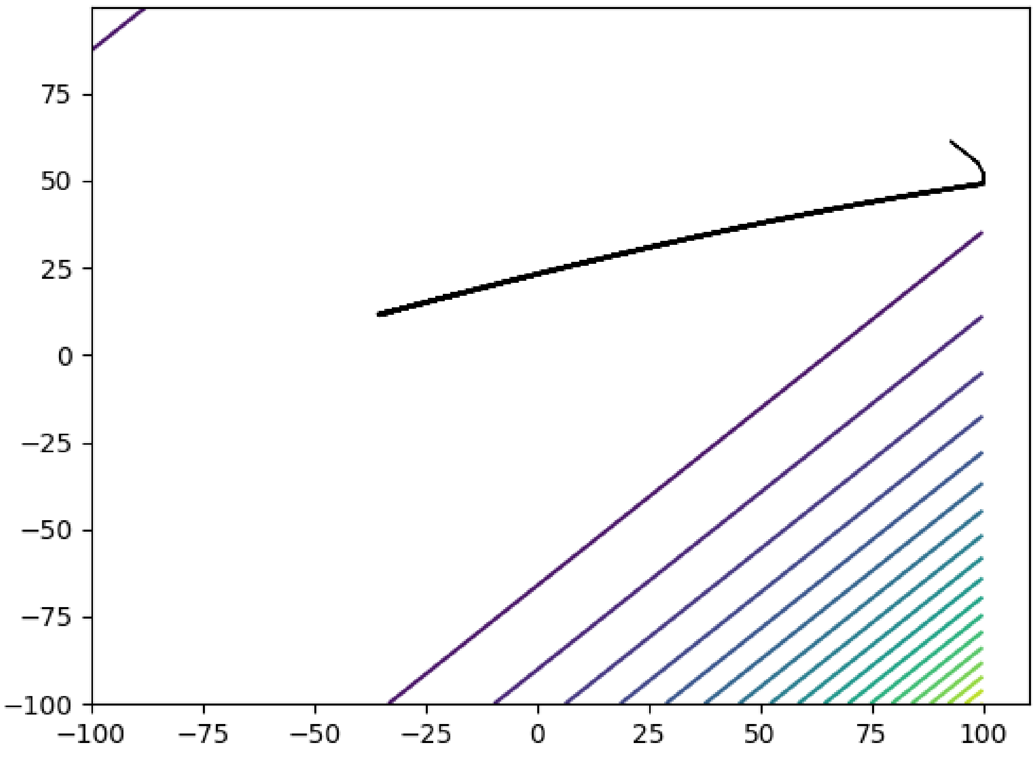
5. Funkcja f3 z CEC 2017

Funkcja f3 również okazała się być bardzo wrażliwa na wielkość parametru beta, dlatego ponownie zastosowałem dużą wartość parametru z jednoczesnym ograniczeniem maksymalnej długości kroku.

Parametr beta = 0,0001
Liczba iteracji = 10 000

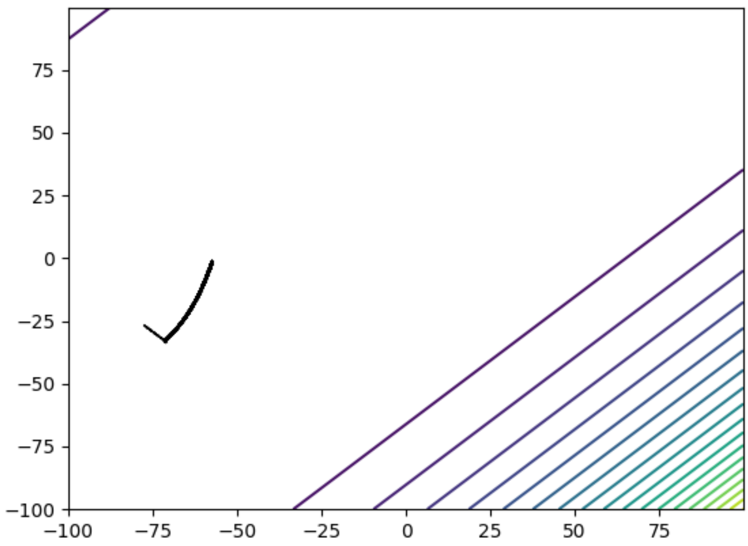
Punkt 1:

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	F(X)
start	92,67	61,19	-10,04	14,88	-45,13	-29,90	-3,88	-54,56	35,65	13,65	9,80*e+9
end	-35,39	11,56	30,56	10,68	-48,61	7,08	8,21	-8,04	-49,12	-46,62	977,91



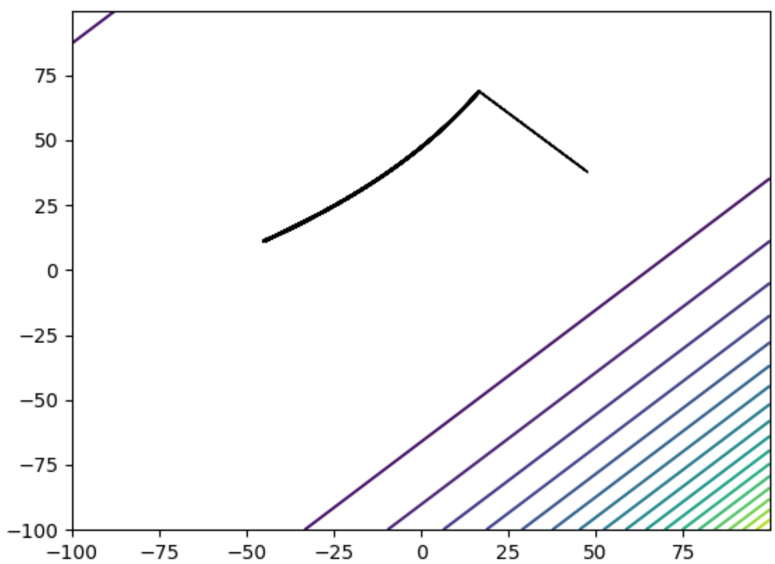
Punkt 2:

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	F(X)
start	-77,61	-26,73	61,73	98,88	-72,32	28,55	-82,86	50,07	-25,04	25,89	7,81*e+8
end	-57,44	-1,58	39,94	20,83	-51,77	7,86	-0,44	5,57	-57,49	-48,94	665,39



Punkt 3:

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	F(X)
start	47,73	37,64	-86,28	81,43	76,16	85,73	75,55	67,47	-37,18	1,75	4,57*e+11
end	-44,85	11,21	17,35	14,27	-27,66	7,17	13,71	3,06	-54,56	-45,13	1347,89



Komentarz do wykresów:

W przypadku powyższej funkcji w płaszczyźnie wyznaczonej przez x1 i x2 zauważyć można wypłaszczenie zajmujące znaczny obszar w obrębie ograniczeń kostkowych. Pochodne cząstkowe względem x1 i x2 są na tym wypłaszczeniu jest bardzo małe, przez co zbieganie do minimum zajmuje wiele iteracji. Na każdym wykresie nastąpiło wejście w obszar wypłaszczenia z innej strony, co spowodowało znaczną rozbieżność w tym, jak wygląda końcowy wektor X.

Najniższa znaleziona wartość funkcji = 665,39 dla punktu 2.

6. Podsumowanie

Metoda najszybszego wzrostu jest prostą w założeniach i łatwą w implementacji metodą pozwalającą na szybkie znajdowanie minimum wielowymiarowych funkcji. W swojej prostocie jest niestety podatna na ograniczenia wynikające z natury optymalizowanej funkcji. Dotyczy to głównie sytuacji, gdy gradient funkcji jest bardzo mały w miejscach poza minimum globalnym, czy funkcji, które posiadają minima lokalne. Aby poradzić sobie z takimi ograniczeniami należałoby wprowadzić zmienną długość kroku lub zastosować inne metody szukania minimum.