Wprowadzenie do sztucznej inteligencji | cw. 1

Damian D'Souza

Zad 1.

1. Jakie rozwiązania i jaką wartość funkcji oceny uzyskano? Czy uzyskano takie same rozwiązania?

   Dla danych wejściowych:

   ```
   m = np.array([8, 3, 5, 2]) #masa przedmiotów
   M = np.sum(m)/2 #niech maksymalna masa plecaka będzie równa połowie masy przedmiotów
   p = np.array([16, 8, 9, 6]) #wartość przedmiotów
   ```

   Program korzystający z metody wyczerpującej wybierze przedmioty o indeksach 1 i 2 co da łączną wartość 17 i zapełnienie 8/9, z kolei przy użyciu heurystyki wybrane zostaną kolejno przedmioty 3 i 1, ponieważ mają one największe stosunki wartości do masy. Używając heurystyki łączna wartość wyniesie 14 przy zapełnieniu 5/9.

2. Jak dużą instancję problemu (liczba przedmiotów) da się rozwiązać w około minutę metodą przeglądu wyczerpującego? Ile czasu zajmie rozwiązanie tego problemu metodą zachłanną (używając heurystyki)? Odpowiednio długie wektory *m* i *p* należy wylosować, *M* = np.sum(*m*)/2.

   Wyczerpująca

l. elementów	min	śr	std	max
5	0,00007	0,000116	0,000076	0,00039
10	0,001563	0,001699	0,000466	0,004228
15	0,049128	0,050971	0,002234	0,064533
20	1,617771	1,664687	0,034101	1,818593
25	52,482227	53,494768	0,627241	54,925919

   Metodą wyczerpującą w około minutę można rozwiązać problem z 25 przedmiotami.

Heurystyka

l. elementów	min	śr	std	max
5	0,000005	0,00001	0,000008	0,000044
10	0,000009	0,000013	0,000009	0,000055
15	0,000014	0,000018	0,00001	0,000061
20	0,000021	0,000026	0,000007	0,000053
25	0,000027	0,000032	0,00001	0,000072

Ten sam problem przy użyciu heurystyki można rozwiązać średnio w około 0,000032s.

3. Jak bardzo wydłuży obliczenia dodanie jeszcze jednego przedmiotu?

   Jak widać z powyższej tabeli, czas wykonania algorytmu metodą wyczerpującą wzrasta wykładniczo, ponieważ dodanie jednego elementu podwaja liczbę możliwych kombinacji do sprawdzenia. Dlatego czas wykonania dla 26 elementów wynosiłby około 2 minut.

4. Jakie wnioski można wyciągnąć na podstawie wyników tego ćwiczenia?

   Metoda wyczerpująca, choć gwarantuje znalezienie dokładnego rozwiązania, nie jest optymalna w przypadku rozwiązywania dużych instancji problemu ze względu na swoją czasochłonność i złożoność obliczeniową. Działa ona dobrze przy małej liczbie przedmiotów, np. 6, ponieważ w takiej sytuacji istnieje zaledwie 64 możliwych kombinacji do sprawdzenia. W tak małym przypadku metoda ta zapewnia dokładność i brak ryzyka wyboru nieoptymalnych przedmiotów, co może się zdarzyć przy zastosowaniu metody zachłannej. Jednakże w miarę wzrostu liczby przedmiotów liczba możliwych kombinacji rośnie wykładniczo, co sprawia, że metoda wyczerpująca staje się niepraktyczna.

   Dlatego w przypadku dużych problemów, mimo że metoda zachłanna nie zawsze prowadzi do najlepszego rozwiązania, często okazuje się znacznie bardziej efektywna czasowo, co czyni ją odpowiednią do rozwiązywania problemów, w których priorytetem jest znalezienie dobrego, choć niekoniecznie optymalnego rozwiązania w krótkim czasie. Z tego powodu metoda zachłanna, mimo swoich ograniczeń, jest preferowana przy większych instancjach problemów

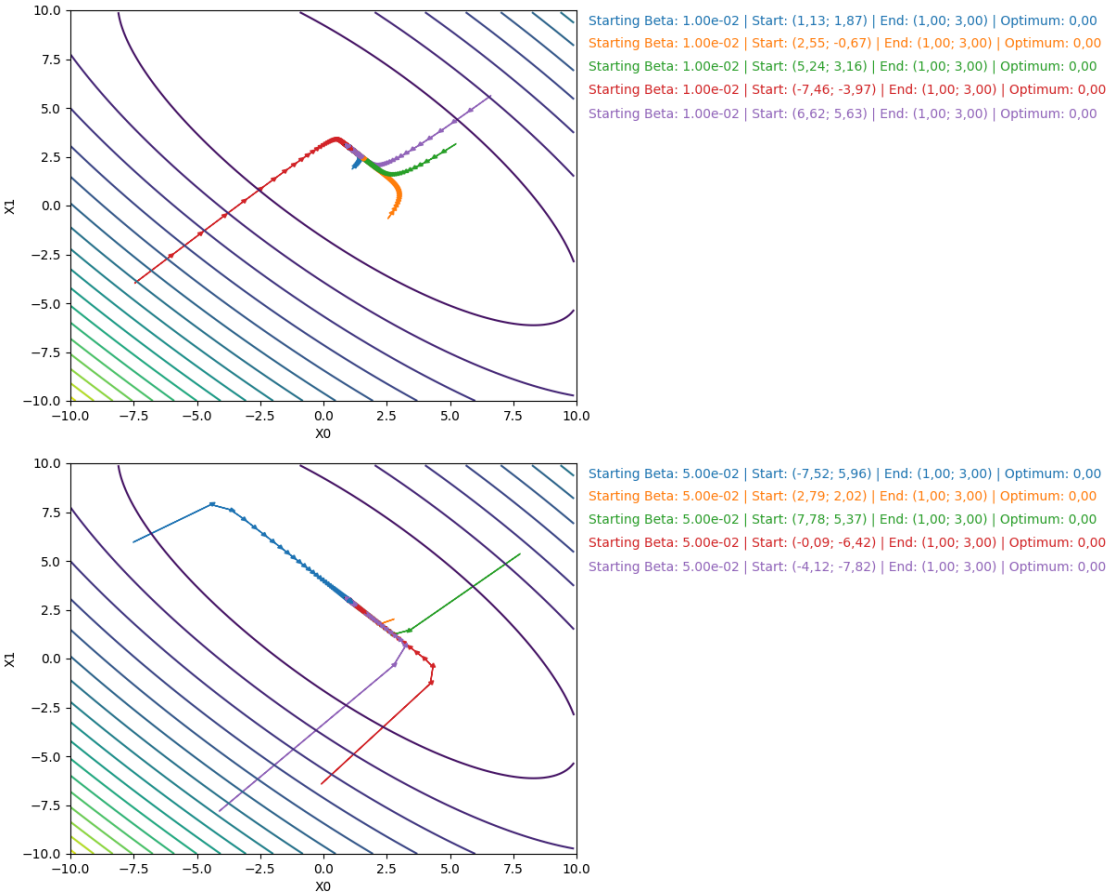
Zad. 2

1. Jak wartość parametru beta wpływa na szybkość dojścia do optimum i zachowanie algorytmu? Jakiej bety użyto dla każdej z funkcji?

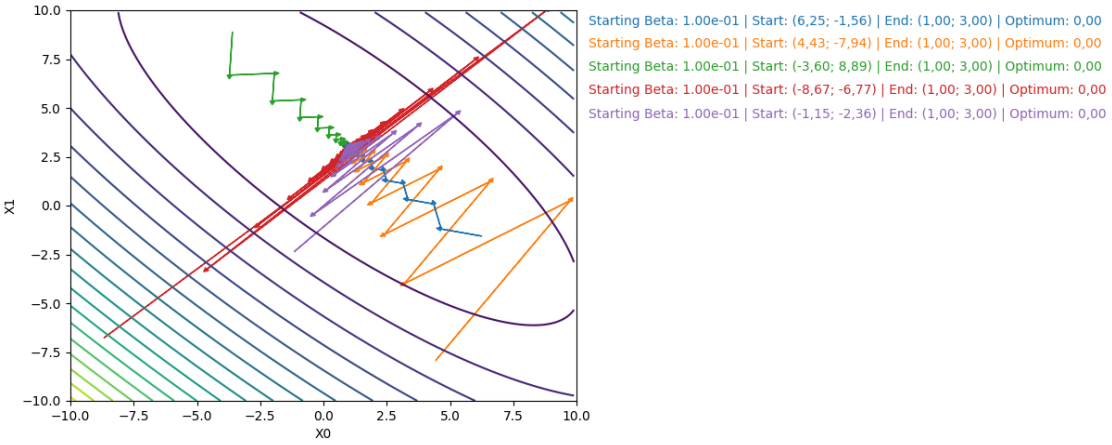
   Do optimum można dojść szybciej używając większego parametru beta, jednak w zależności od wielkości gradientu funkcji w punkcie wykorzystanie zbyt dużej bety może spowodować oscylacje wokół optimum, co spowolni lub uniemożliwi jego odnalezienie, z kolei zastosowanie zbyt małej bety może spowodować, że algorytm będzie się wykonywał nieskończenie długo.

   - Funkcja booth

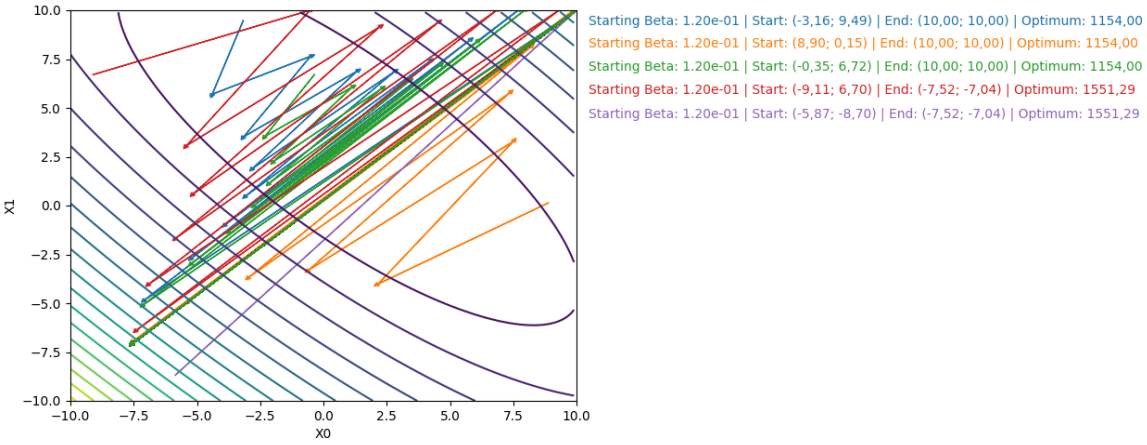
   Znalezienie optimum funkcji booth z parametrem beta w zakresie 0,01 do 0,1 nie stanowi problemu.



Od wartości parametru 0,1 punkty zaczynają oscylować, jednak optimum jest wyznaczane w mniej niż 1000 iteracji.



W przypadku kroku większego niż 0,12, znalezienie optimum staje się niemożliwe w rozsądnym czasie – punkty oscylują wokół optimum, a program kończy się dopiero po przekroczeniu limitu 40 000 iteracji.



W celu znalezienia optymalnych rozwiązań funkcji z zestawu CEC 2017 przeprowadziłem dwie próby, testując różne wartości parametru beta. W jednej z prób parametr beta był stały, a w drugiej zastosowałem zmienny krok wyznaczany na podstawie krótkiego współczynnika kroku, obliczanego za pomocą metody Barzilaia-Borweina.

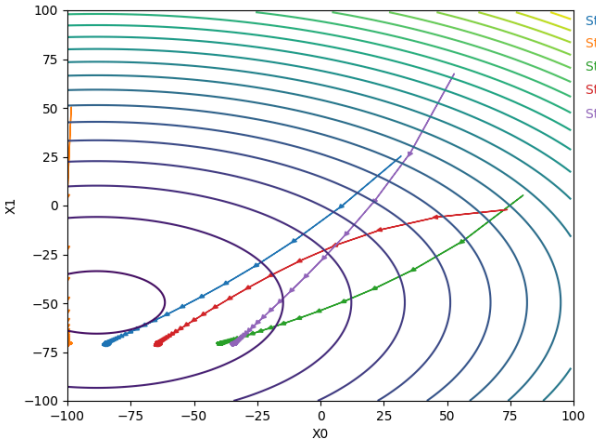
$$\beta_k = \frac{\Delta x \cdot \Delta g}{\Delta g \cdot \Delta g}$$

$$\Delta x = x_k - x_{k-1}$$

$$\Delta g = \nabla q(x_k) - \nabla q(x_{k-1})$$

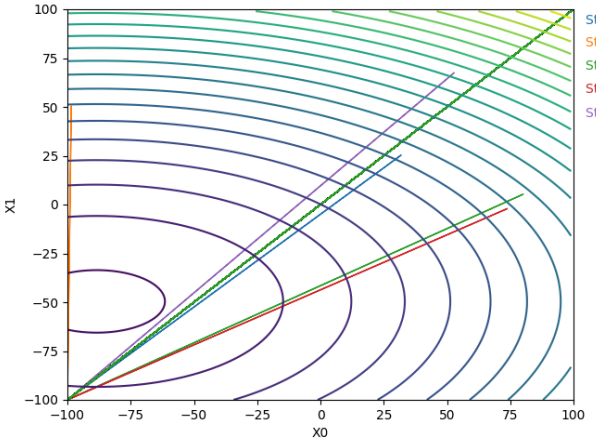
- F1

Dla funkcji F1 najlepsze wyniki uzyskałem z krokiem równym 1e-8, na podstawie rysunku mogłoby się wydawać że punkty zbliżają się do minimum jednak ogranicza się to tylko niektórych wymiarów. Jak widać na wykresie wartości funkcji są bardzo zróżnicowane i dalekie od wartości minimalnej.



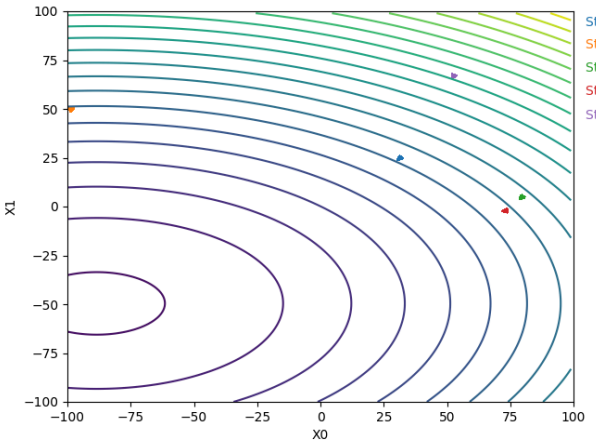
Starting Beta: 1.00e-08 | Start: (31,91; 25,35) | End: (-84,55; -70,43) | Optimum: 2470,58
Starting Beta: 1.00e-08 | Start: (-98,48; 50,55) | End: (-99,96; -70,43) | Optimum: 5623,18
Starting Beta: 1.00e-08 | Start: (80,14; 5,30) | End: (-39,08; -70,43) | Optimum: 825,56
Starting Beta: 1.00e-08 | Start: (73,83; -2,09) | End: (-64,27; -70,43) | Optimum: 323,51
Starting Beta: 1.00e-08 | Start: (52,87; 67,52) | End: (-33,76; -70,43) | Optimum: 1380,13

Dla parametru beta równego 1e-4 punkty doszły do krańca przedziału i nic więcej nie udało się uzyskać, ponieważ parametr był zbyt mały.



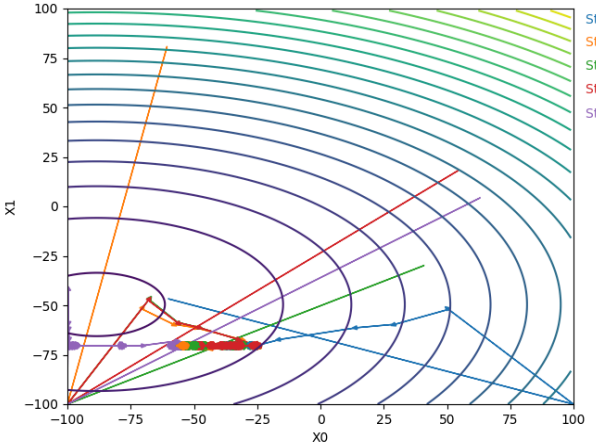
Starting Beta: 1.00e-04 | Start: (31,91; 25,35) | End: (100,00; 100,00) | Optimum: 2166372
Starting Beta: 1.00e-04 | Start: (-98,48; 50,55) | End: (100,00; 100,00) | Optimum: 1980292
Starting Beta: 1.00e-04 | Start: (80,14; 5,30) | End: (100,00; 100,00) | Optimum: 29525467
Starting Beta: 1.00e-04 | Start: (73,83; -2,09) | End: (100,00; 100,00) | Optimum: 17704723
Starting Beta: 1.00e-04 | Start: (52,87; 67,52) | End: (100,00; 100,00) | Optimum: 3118066

Z kolei przy użyciu mniejszego parametru 1e-15 położenie ostatniego punktu w ciągu 40 000 iteracji praktycznie nie uległo zmianie względem punktu początkowego.

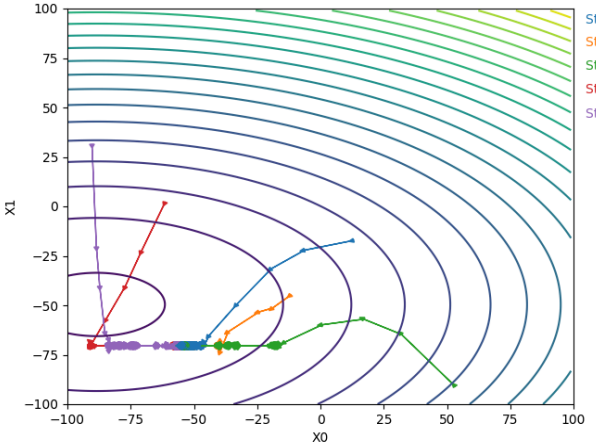


Starting Beta: 1.00e-15 | Start: (31,91; 25,35) | End: (31,81; 25,25) | Optimum: 448105090(
Starting Beta: 1.00e-15 | Start: (-98,48; 50,55) | End: (-98,49; 50,37) | Optimum: 41168485
Starting Beta: 1.00e-15 | Start: (80,14; 5,30) | End: (80,05; 5,20) | Optimum: 642754353602
Starting Beta: 1.00e-15 | Start: (73,83; -2,09) | End: (73,72; -2,11) | Optimum: 9803196149
Starting Beta: 1.00e-15 | Start: (52,87; 67,52) | End: (52,80; 67,35) | Optimum: 114920031

Zastosowanie zmiennego współczynnika beta skróciło czas poszukiwania optimum oraz zwiększyło dokładność w porównaniu do współczynnika stałego, nawet przy zbyt małej lub zbyt dużej wartości początkowej. Wyniki były porównywalne dla wszystkich wartości kroku, jednak dla zbyt małych wartości początkowo punkty zmierzały w kierunku końca przedziału



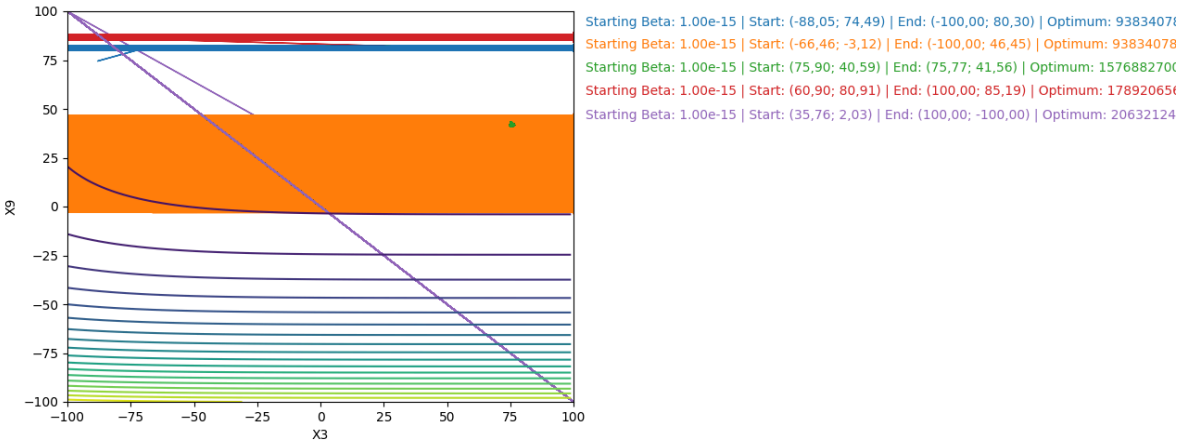
Starting Beta: 1.00e-04 | Start: (-60,14; -46,73) | End: (-55,22; -70,43) | Optimum: 100,01
Starting Beta: 1.00e-04 | Start: (-60,62; 80,80) | End: (-55,22; -70,43) | Optimum: 100,01
Starting Beta: 1.00e-04 | Start: (40,93; -29,89) | End: (-55,23; -70,43) | Optimum: 100,00
Starting Beta: 1.00e-04 | Start: (54,46; 18,15) | End: (-55,22; -70,43) | Optimum: 100,01
Starting Beta: 1.00e-04 | Start: (63,16; 4,35) | End: (-55,34; -70,43) | Optimum: 100,01

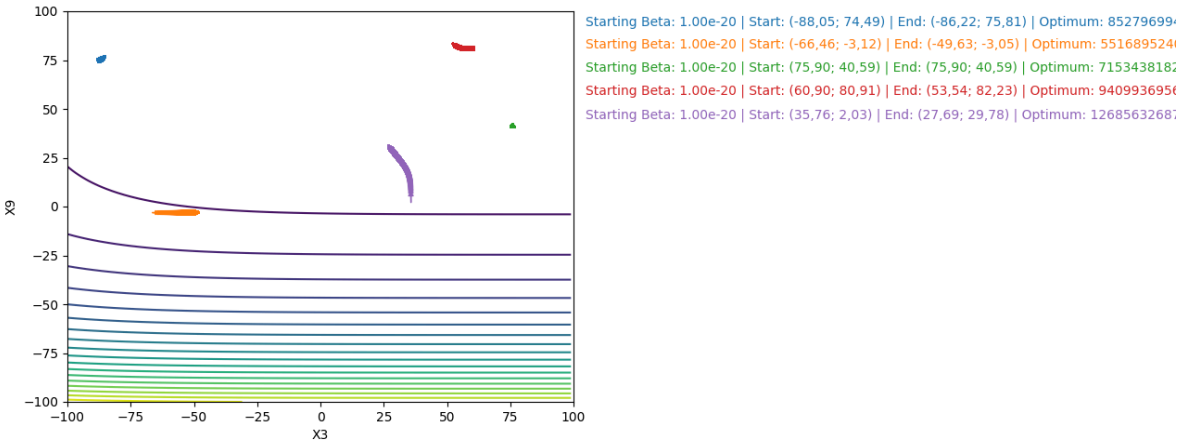


Starting Beta: 1.00e-15 | Start: (13,11; -17,22) | End: (-55,27; -70,43) | Optimum: 100,00
Starting Beta: 1.00e-15 | Start: (-11,66; -44,76) | End: (-55,19; -70,43) | Optimum: 100,02
Starting Beta: 1.00e-15 | Start: (53,27; -90,98) | End: (-55,23; -70,43) | Optimum: 100,01
Starting Beta: 1.00e-15 | Start: (-61,37; 2,11) | End: (-55,35; -70,43) | Optimum: 100,01
Starting Beta: 1.00e-15 | Start: (-90,30; 31,42) | End: (-55,29; -70,43) | Optimum: 100,00

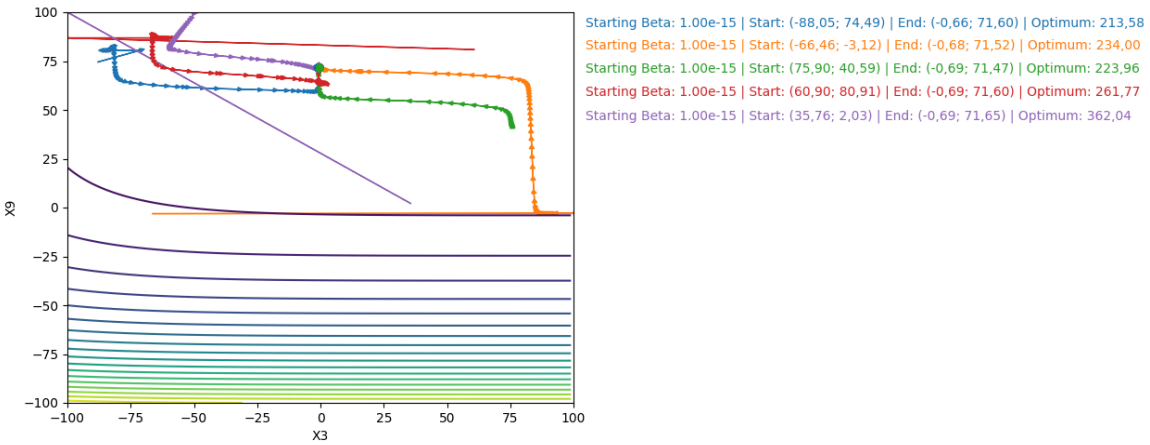
- F2

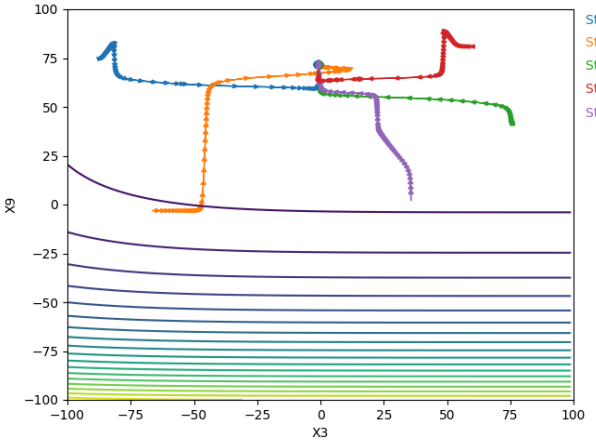
Osiągnięcie optimum funkcji F2 przy stałym współczynniku okazało się niemożliwe. Przy wyższych wartościach parametru punkty zbiegały do krańców przedziału, natomiast przy niższych ich położenie pozostawało niezmienne lub zmieniało się bardzo powoli, co doprowadziło do zakończenia działania programu po 40 000 iteracjach.





Zastosowanie zmiennego parametru pozwala na określenie punktów w pobliżu optimum, jednak użycie mniejszej wartości początkowej umożliwia osiągnięcie optimum z większą dokładnością.

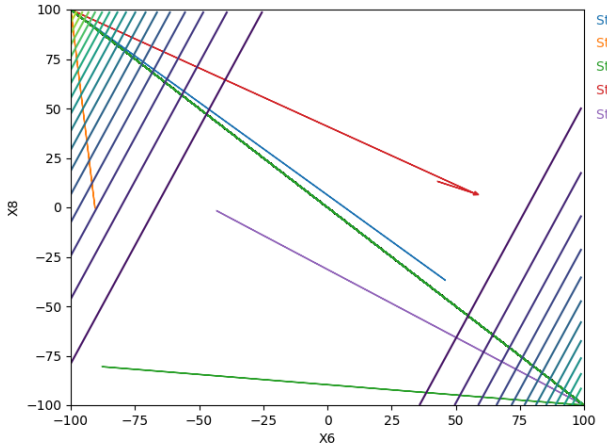




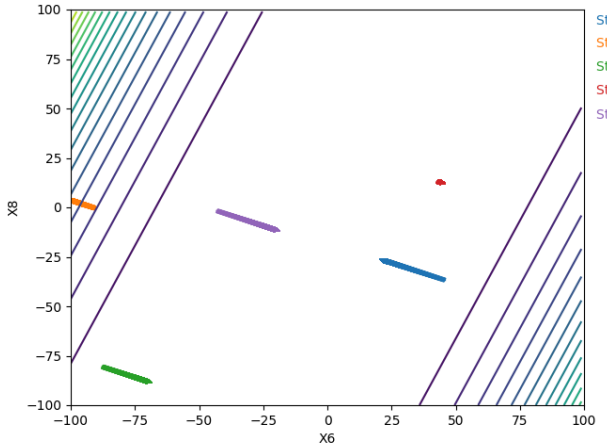
Starting Beta: 1.00e-20 | Start: (-88,05; 74,49) | End: (-0,66; 71,60) | Optimum: 205,65
Starting Beta: 1.00e-20 | Start: (-66,46; -3,12) | End: (-0,69; 71,49) | Optimum: 211,29
Starting Beta: 1.00e-20 | Start: (75,90; 40,59) | End: (-0,69; 71,45) | Optimum: 219,35
Starting Beta: 1.00e-20 | Start: (60,90; 80,91) | End: (-0,67; 71,60) | Optimum: 201,12
Starting Beta: 1.00e-20 | Start: (35,76; 2,03) | End: (-0,68; 71,47) | Optimum: 261,44

- F3

Podobnie jak w przypadku funkcji F2, nie udało się osiągnąć optimum funkcji F3 przy użyciu stałego współczynnika kroku. Przy większych wartościach, takich jak 1e-5, punkty zbiegały do krańców przedziału, natomiast przy mniejszych, takich jak 1e-10, pozostawały w miejscu lub przemieszczały się bardzo nieznacznie.

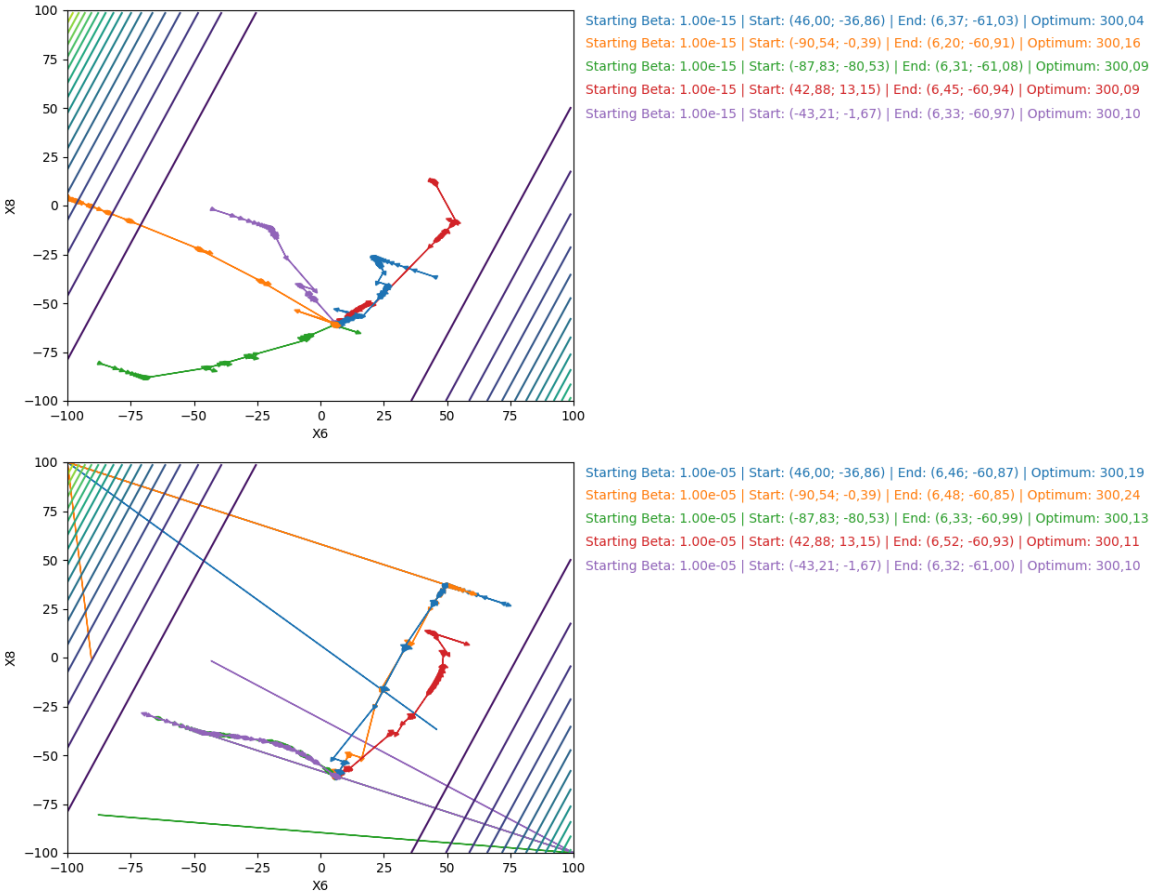


Starting Beta: 1.00e-05 | Start: (46,00; -36,86) | End: (100,00; -100,00) | Optimum: 52932(
Starting Beta: 1.00e-05 | Start: (-90,54; -0,39) | End: (100,00; -100,00) | Optimum: 529320
Starting Beta: 1.00e-05 | Start: (-87,83; -80,53) | End: (-100,00; 100,00) | Optimum: 58459
Starting Beta: 1.00e-05 | Start: (42,88; 13,15) | End: (-100,00; 100,00) | Optimum: 584597
Starting Beta: 1.00e-05 | Start: (-43,21; -1,67) | End: (-100,00; 100,00) | Optimum: 584597



Starting Beta: 1.00e-10 | Start: (46,00; -36,86) | End: (22,32; -26,93) | Optimum: 101628,4
Starting Beta: 1.00e-10 | Start: (-90,54; -0,39) | End: (-99,89; 3,53) | Optimum: 122763,10
Starting Beta: 1.00e-10 | Start: (-87,83; -80,53) | End: (-70,33; -87,87) | Optimum: 117310,
Starting Beta: 1.00e-10 | Start: (42,88; 13,15) | End: (44,15; 12,62) | Optimum: 112488,16
Starting Beta: 1.00e-10 | Start: (-43,21; -1,67) | End: (-20,50; -11,21) | Optimum: 148348,8

Zastosowanie zmiennego kroku pomogło w zbliżeniu się do optimum funkcji z wystarczającą dokładnością, dla wszystkich wartości początkowych parametru wyniki były porównywalne, ale aby uniknąć niepotrzebnych skoków na krańce przedziału lepiej wybrać mniejszą wartość, taką jak 1e-15.



2. Zalety/wady algorytmu?

   Algorytm najszybszego wzrostu jest bardzo prosty i łatwy do zaimplementowania, działa też szybko. Istnieje też wiele wariantów i metod, które mogą przyspieszyć i polepszyć pracę tego algorytm np. metoda Barzilaia-Borweina, którą zastosowałem. W przypadku bardziej skomplikowanych danych takich jak funkcje benchmarkowe CEC algorytm nie będzie działał dobrze lub nie będzie działał w ogóle.

3. Wnioski

   Analiza algorytmu najszybszego wzrostu ujawniła znaczący wpływ parametru beta na skuteczność tej metody optymalizacji. Zastosowanie zbyt dużych wartości prowadzi do oscylacji wokół optimum lub do dążenia do krańców przedziałów, natomiast wybór zbyt małej wartości znacząco spowalnia

działanie algorytmu. Wykazano również, że algorytm nie działa optymalnie w przypadku skomplikowanych danych, takich jak wielowymiarowe funkcje benchmarkowe CEC, ale sprawdza się bardzo dobrze i szybko dla prostszych danych, jak funkcja Booth. Dalsze optymalizacje podstawowej metody mogą znacząco usprawnić jej działanie. Zastosowanie zmiennego kroku nie tylko zwiększyło dokładność znajdowanych optymalnych rozwiązań, ale również znacząco przyspieszyło działanie algorytmu.