Kamil Marszałek 331401

Ćwiczenie 1

Zadanie 1

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

Uzyskane rozwiązania różnią się od siebie. Dla zestawu danych podanego w instrukcji:

- przegląd wyczerpujący:

```
Mass limit: 9.0

Generating all combinations:
Max price: 17 Max mass: 8
Time: 0.000974s
Max price: 17 Mass of prods: 8
Time: 0.000974s
```

- rozwiązanie heurystyczne:

```
Mass limit: 9.0


Heuristic solution:
Max price: 14 Mass of prods: 5
Time: 0.000323s
```

Wyniki różnią się, ponieważ w rozwiązaniu heurystycznym przedmioty są dodawane do plecaka w kolejności malejącego stosunku p/m, co nie zawsze prowadzi do uzyskania najlepszego rezultatu.

Przeprowadziłem badania, aby ocenić różnice między rozwiązaniami generowanymi przez oba podejścia. Dla każdego n ze zbioru {5, 10, 15, 20, 25} wykonano 25 prób. Liczby reprezentujące masy i wartości zostały wylosowane z przedziału od 1 do 100. W tabeli przedstawiono minimalną, maksymalną oraz średnią różnicę między wynikami uzyskanymi za pomocą heurystyki a metodą przeglądu wyczerpującego.

n	MAX DIFFERENCE	MIN DIFFERENCE	MEAN DIFFERENCE
5,00	58,00	0,00	3,72
10,00	23,00	0,00	3,48
15,00	30,00	0,00	5,36
20,00	44,00	0,00	9,24
25,00	33,00	0,00	7,92

Na podstawie przedstawionych danych można stwierdzić, że rozwiązanie heurystyczne ma szansę być zbliżone do optymalnego, jednak nie gwarantuje to jego optymalności. Rozwiązanie to dostarcza jedynie przybliżenia optymalnego wyniku, co oznacza, że w niektórych przypadkach może ono odbiegać od rozwiązania wyczerpującego. Warto zauważyć, że choć heurystyka może być efektywna czasowo, nie zawsze prowadzi do uzyskania najlepszego możliwego rezultatu.

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.

- Przegląd wyczerpujący:

Dane wejściowe: tablica mas oraz wartości zostaje wylosowana z zakresu od 1 do 100. Masa zostaje obliczona jako połowa sumy masy wszystkich przedmiotów.

Tabela przedstawia wyniki pomiarów czasu wykonania dla zestawów obejmujących 5, 10, 15, 20 oraz 25 elementów. Uwzględniono wartość minimalną, maksymalną, średnią oraz odchylenie standardowe. Dla każdego n wykonano 25 pomiarów.

n	MINIMUM TIME	MAXIMUM TIME	MEAN TIME	STANDARD DEVIATION
5,000000	0,000063	0,000081	0,000066	0,000004
10,000000	0,001986	0,002182	0,002050	0,000045
15,000000	0,064156	0,084782	0,067720	0,003831
20,000000	2,113288	2,254450	2,211796	0,029239
25,000000	70,562099	81,101627	72,568020	2,079178

-Metoda heurystyczna:

Dane wejściowe: tablica mas oraz wartości zostaje wylosowana z zakresu od 1 do 100. Masa zostaje obliczona jako połowa sumy masy wszystkich przedmiotów.

W tabeli zebrano wyniki pomiarów czasu wykonania dla 5, 10, 15, 20 oraz 25 elementów. Zostały podane wartość minimalna, maksymalna, średnia oraz odchylenie standardowe. Dla każdego n wykonano 25 pomiarów.

n	MINIMUM TIME	MAXIMUM TIME	MEAN TIME	STANDARD DEVIATION
5,000000	0,000007	0,000017	0,000008	0,000002
10,000000	0,000015	0,000019	0,000017	0,000001
15,000000	0,000029	0,000063	0,000038	0,000008
20,000000	0,000046	0,000078	0,000053	0,000007
25,000000	0,000054	0,000071	0,000060	0,000004

Zauważamy, że dla tego samego zbioru danych metoda heurystyczna generuje wyniki znacznie szybciej niż metoda przeglądu wyczerpującego. Dodatkowo dla heurystyki przeprowadzono pomiary dla n ze zbioru {10, 1000, 10 000, 1 000 000, 10 000 000}.

n	MINIMUM TIME	MAXIMUM TIME	MEAN TIME	STANDARD DEVIATION
10,000000	0,000014	0,000029	0,000016	0,000004
1000,000000	0,001400	0,015700	0,002225	0,002755
10000,000000	0,017779	0,037901	0,024420	0,006911
1000000,000000	5,035122	7,094380	5,850277	0,598856
10000000,000000	63,452949	94,724692	74,180522	8,157808

Z przedstawionych danych możemy dostrzec, że w czasie kiedy wykonywaliśmy przegląd wyczerpujący dla 25 elementów możemy znaleźć rozwiązanie metodą heurystyczną dla 10 mln elementów.

Jak bardzo wydłuży obliczenia dodanie jeszcze jednego przedmiotu?

- Dla rozwiązania przez przegląd wyczerpujący:

n	MINIMUM TIME	MAXIMUM TIME	MEAN TIME	STANDARD DEVIATION
10,000000	0,001981	0,002339	0,002091	0,000093
11,000000	0,003997	0,004367	0,004146	0,000100
12,000000	0,008135	0,009414	0,008380	0,000293
13,000000	0,016244	0,017122	0,016623	0,000222
14,000000	0,032370	0,035141	0,033796	0,000652

Widzimy że dodanie jednego elementu powoduje wydłużenie czasu około 2-krotnie.

-Dla rozwiązania heurystycznego:

n	MINIMUM TIME	MAXIMUM TIME	MEAN TIME	STANDARD DEVIATION
10,000000	0,000015	0,000059	0,000018	0,000007
11,000000	0,000017	0,000077	0,000021	0,000010
12,000000	0,000018	0,000071	0,000025	0,000010
13,000000	0,000025	0,000065	0,000034	0,000008
14,000000	0,000029	0,000073	0,000037	0,000008

Można zauważyć, że dodanie jednego elementu powoduje zwiększenie czasu, ale nie w sposób tak dynamiczny jak dla przeglądu wyczerpującego. Zwiększenie ilości elementów o jeden powoduje wzrost czasu wykonania programu średnio w sposób proporcjonalny do przyrostu liczby elementów. Jeśli dodamy jeden element do zbioru dużo większego od jeden to nie dostrzeżemy dużego wzrostu czasu wykonania. Tabela poniżej przedstawia wyniki pomiarów dla n należącego do zbioru {100, 101, 102, 103, 104}:

n	MINIMUM TIME	MAXIMUM TIME	MEAN TIME	STANDARD DEVIATION
100,000000	0,000120	0,000218	0,000146	0,000019
101,000000	0,000124	0,000269	0,000149	0,000027
102,000000	0,000122	0,000258	0,000145	0,000020
103,000000	0,000121	0,000201	0,000143	0,000011
104,000000	0,000122	0,000304	0,000154	0,000034

Jakie wnioski można wyciągnąć na podstawie wyników tego ćwiczenia?

Zauważmy, że rozwiązanie heurystyczne może nie znaleźć najlepszego wyniku, ale pozwala rozwiązywać w sposób przybliżony problem plecakowy dla dużo większej ilości elementów. Nie jest możliwe w rozsądnym czasie znalezienie dokładnego rozwiązania metodą przeglądu wyczerpującego dla dużej ilości elementów (w przypadku mojej konfiguracji w ciągu ok. jednej minuty da się rozwiązać problem dla maksymalnie 25 elementów). Rozwiązanie heurystyczne działa szybko nawet dla dużego zbioru danych. Dodanie pojedynczego elementu przy dużej ilości elementów w zbiorze nie powoduje zwiększenia czasu wykonania dla heurystyki. Natomiast, dla przeglądu wyczerpującego można dostrzec 2-krotny wzrost czasu wykonania dla każdego dodatkowego elementu w zbiorze.

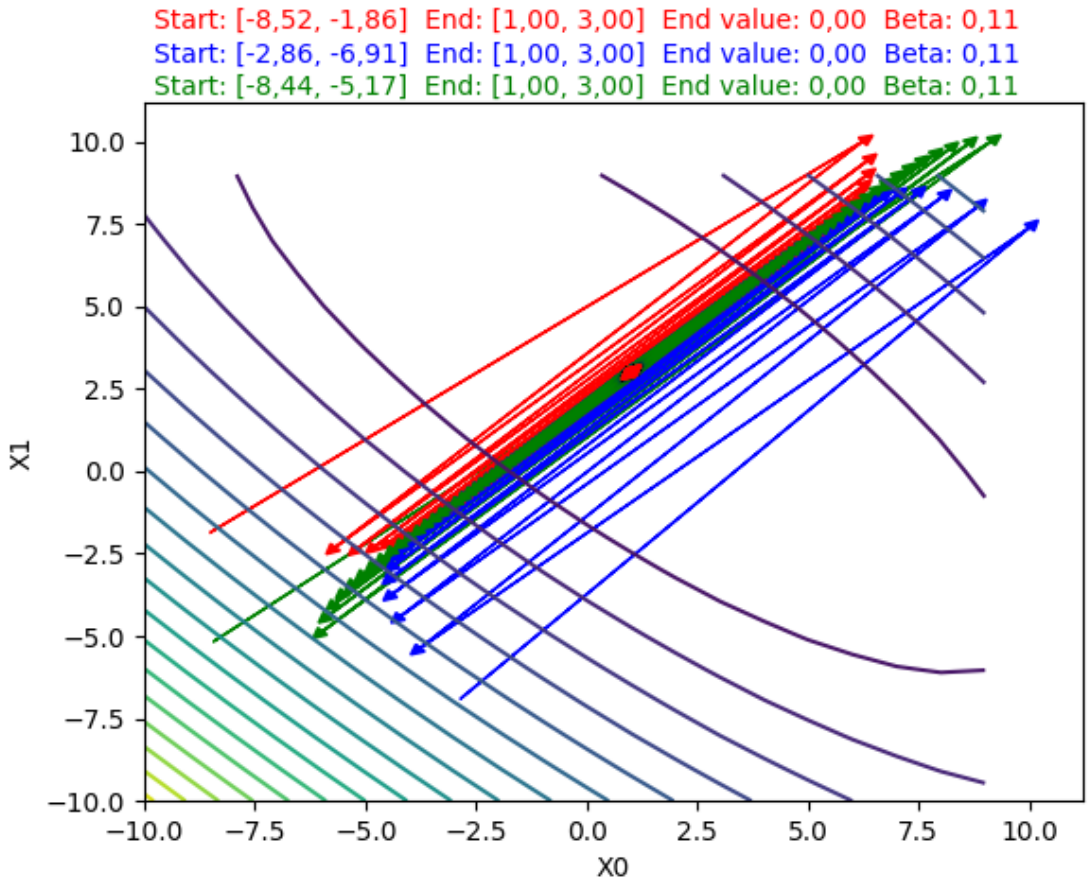
Zadanie 2

Metoda najszybszego wzrostu ze stałym współczynnikiem kroku

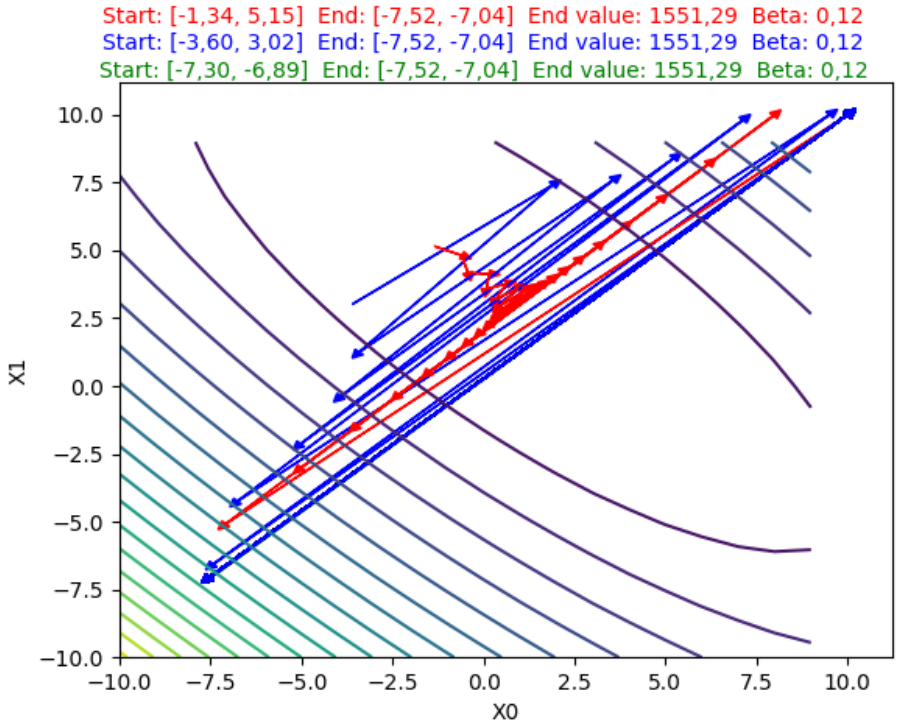
Booth

Dla prostej funkcji booth wersja ze stałym parametrem beta w zupełności wystarczy. Optimum jest szybko znalezione, potrzebna jest niewielka liczba iteracji. Maksymalna wartość bety dla której jest znajdowane minimum to 0,11.

Wykres dla współczynnika beta wynoszącego 0,11

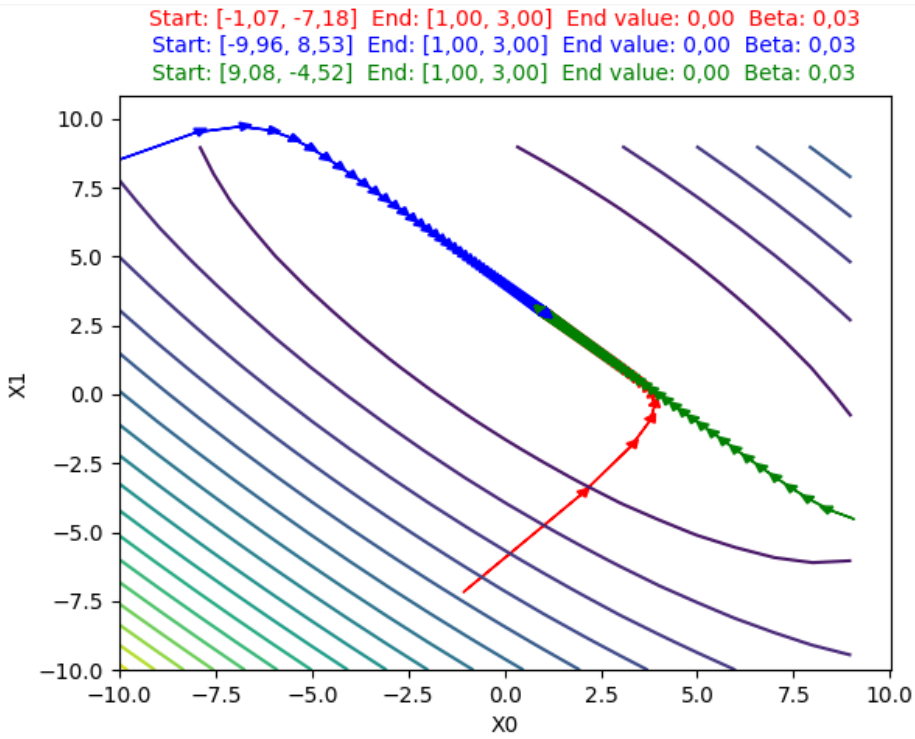


Wykres dla bety wynoszącej 0,12



Beta, która przekracza 0,11 powoduje oscylacje.

Wykres dla parametru kroku poniżej 0,11

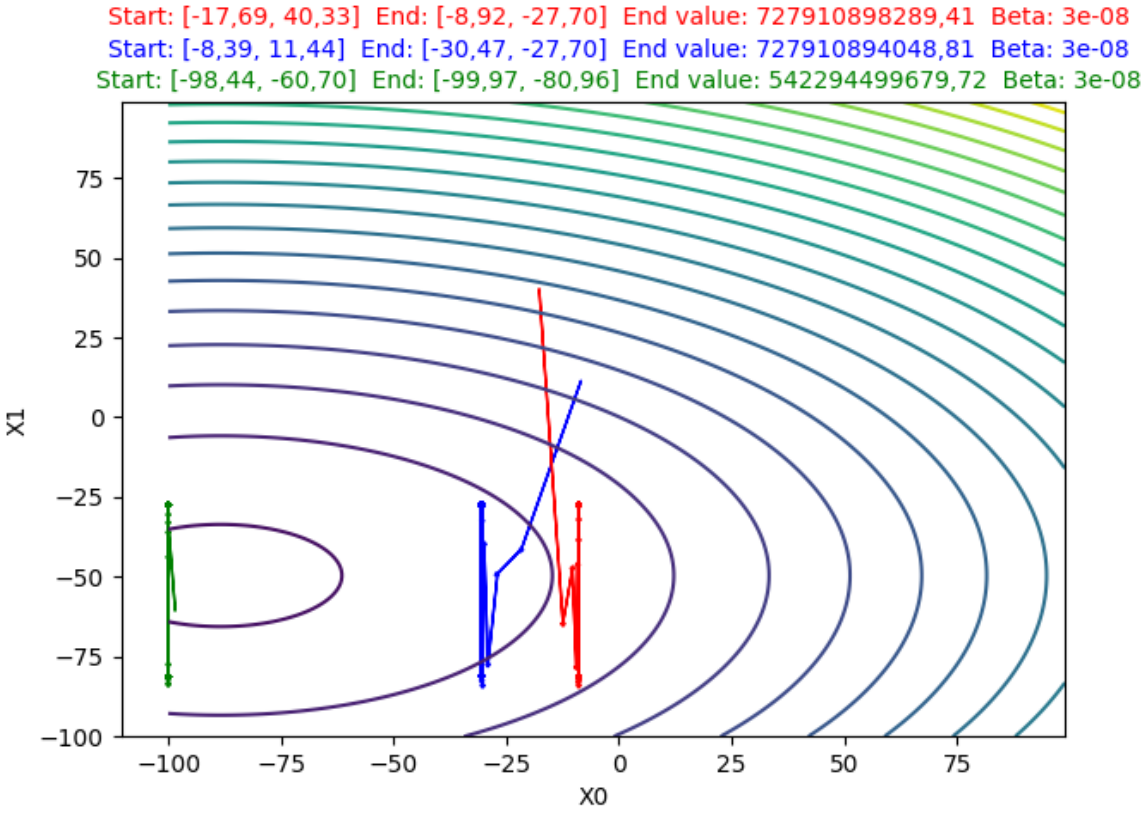


Dla współczynnika beta mniejszego od 0,11 też uzyskujemy optimum, ale wymagane jest więcej iteracji, aby dojść do minimum w punkcie [1, 3] o wartości 0.

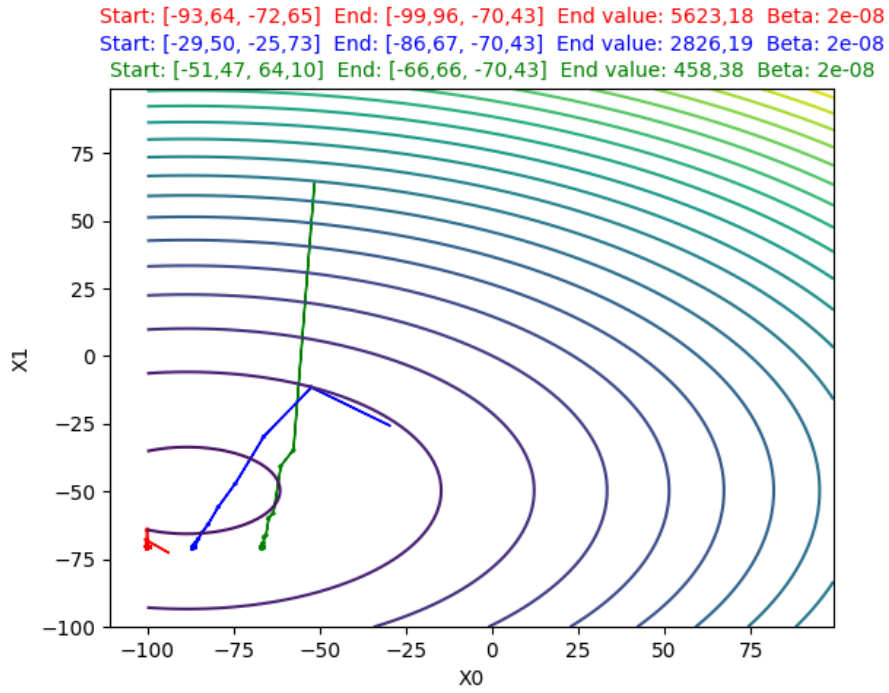
Funkcja f1

Parametr beta musi być mniejszy lub równy od 2e-08. W przeciwnym razie mogą wystąpić oscylacje, co uniemożliwi obliczenie optimum. Im większa wartość beta, tym szybciej działa algorytm, ale jeśli przekroczymy wartość graniczną, nie będziemy w stanie uzyskać oczekiwanego rezultatu. Algorytm zatrzymuje się w różnych punktach, które niekoniecznie są sobie równe, co prowadzi do tego, że nie jest w stanie efektywnie znaleźć optimum. Wynik zależy od punktu startowego, który jest wybierany losowo.

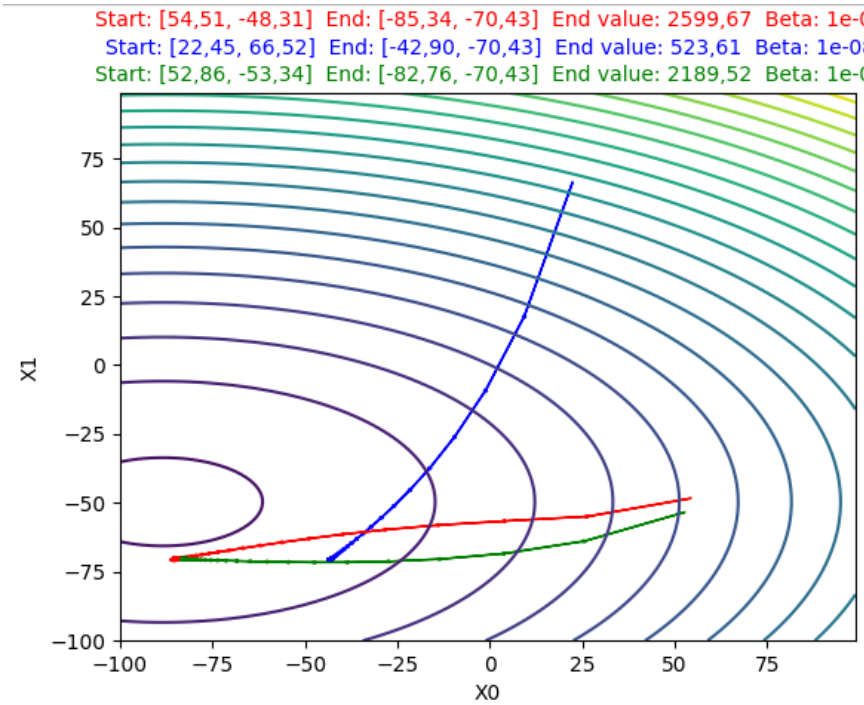
Dla bety równej 3e-08 otrzymujemy oscylacje:



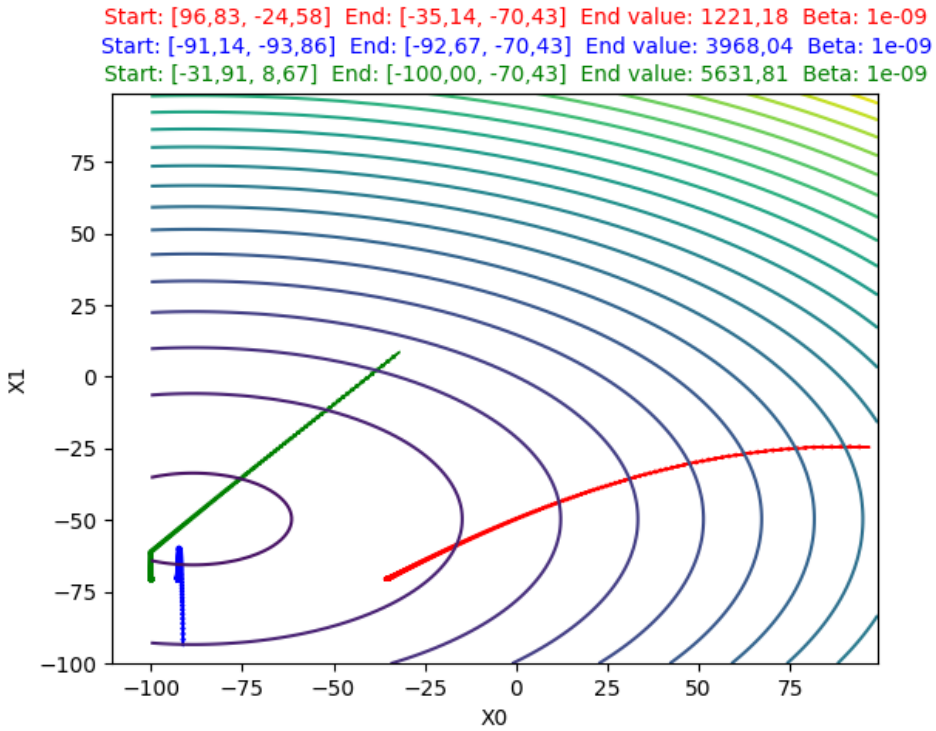
Dla bety wynoszącej 2e-08 otrzymujemy już pewne wyniki, choć są one dość mocno rozbieżne:

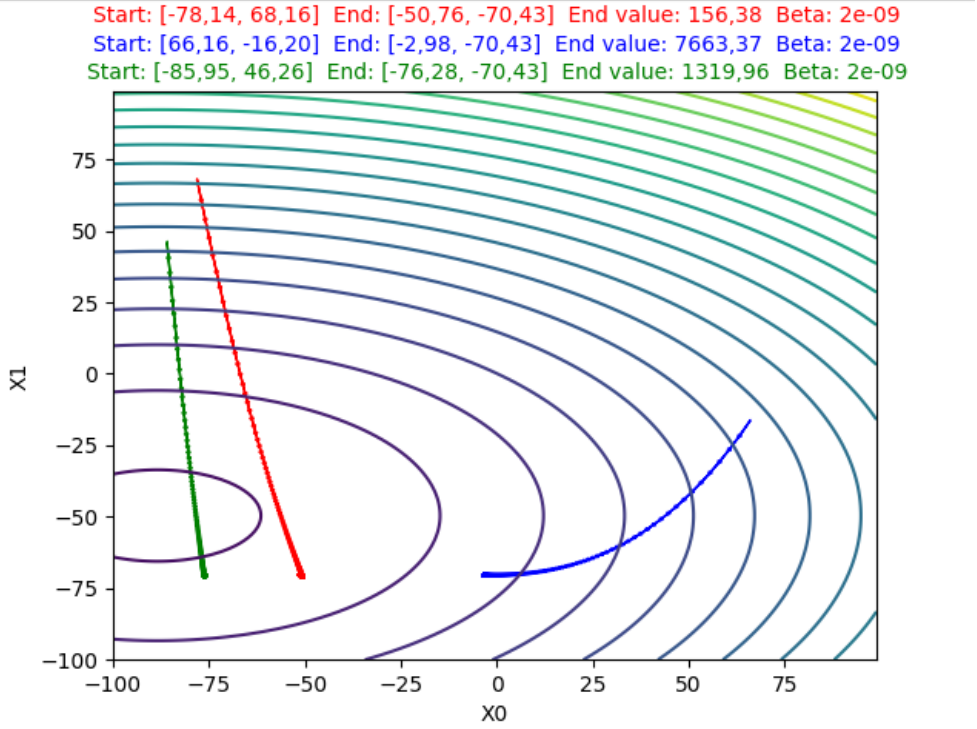


Dla bety wynoszącej 1e-08 funkcja nie zbiega w rozsądnym czasie, ale jest na dobrej drodze:



Dla mniejszych współczynników kroku to zbieganie jest wolniejsze stąd liczba potrzebnych iteracji jest tym większa im mniejsza jest beta.

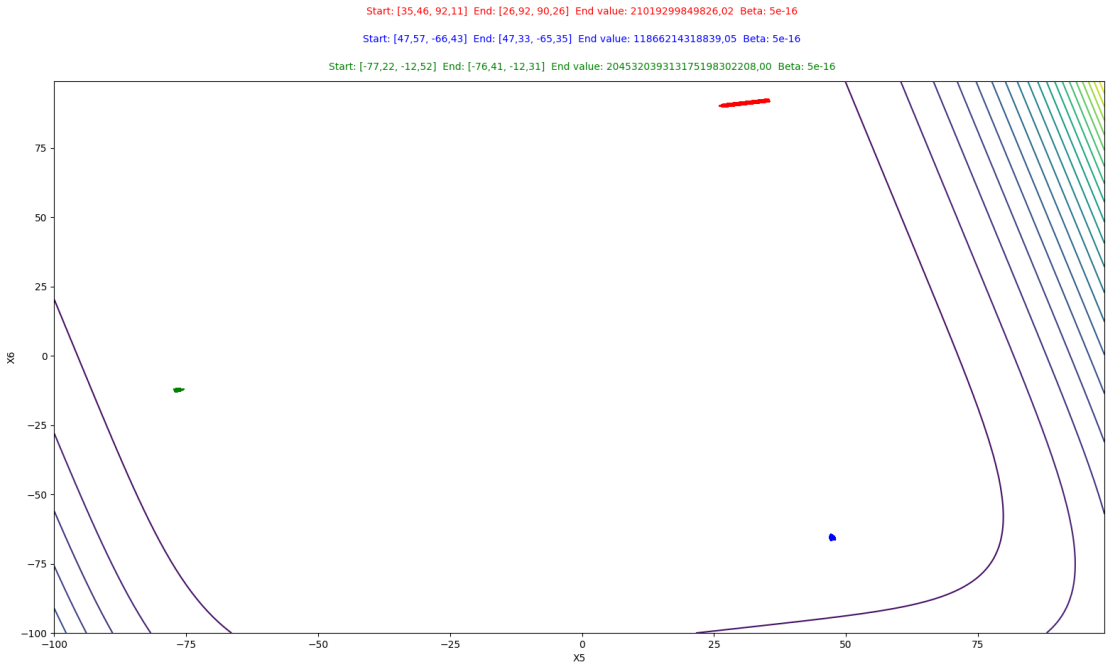


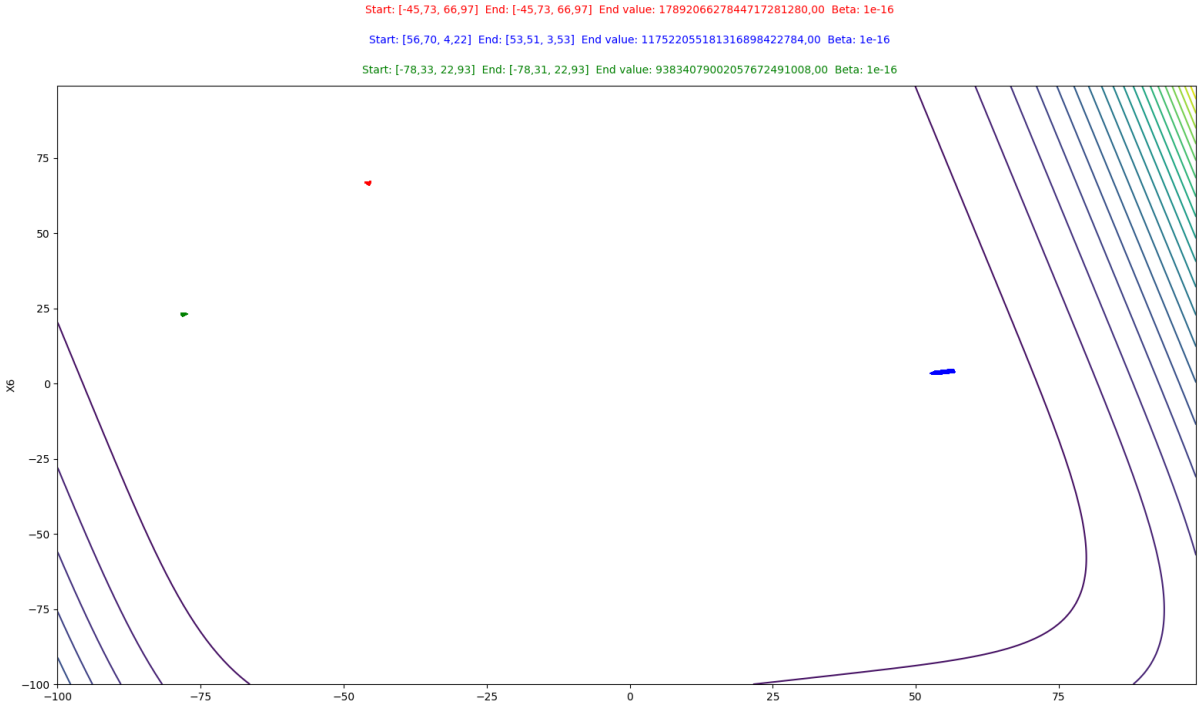


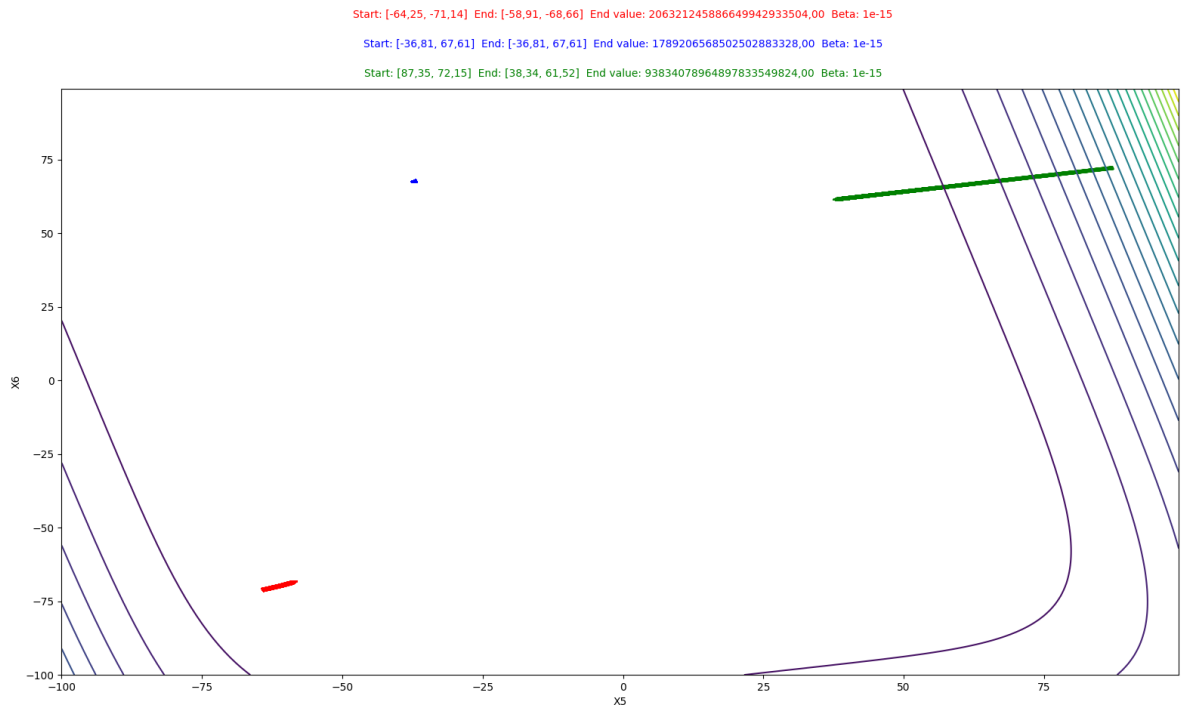
Zdecydowanie najmniej iteracji otrzymujemy dla największego parametru beta. Aczkolwiek najlepszy rezultat udało się uzyskać dla parametru beta wynoszącego 2e-09, jest to jednak kwestia wylosowanego punktu początkowego aniżeli parametru beta. Optimum w tym przypadku wynosiło 156,38. Z kolei dla zbyt małej bety nie znajdujemy optimum przed osiągnięciem limitu iteracji (max. 10000).

Funkcja f2

Dla funkcji f2 ciężko dobrać parametr beta, aby uzyskać sensowny rezultat w rozsądnym czasie. Funkcja f2 jest mocno niestabilna co skutkuje częstym zatrzymywaniem się w minimach lokalnych lub lokalnych oscylacjach. Z kolei dla zbyt małej bety nie dochodzimy do końca algorytmu tylko osiągamy limit iteracji. Rezultat mocno zależy od punktu startowego.





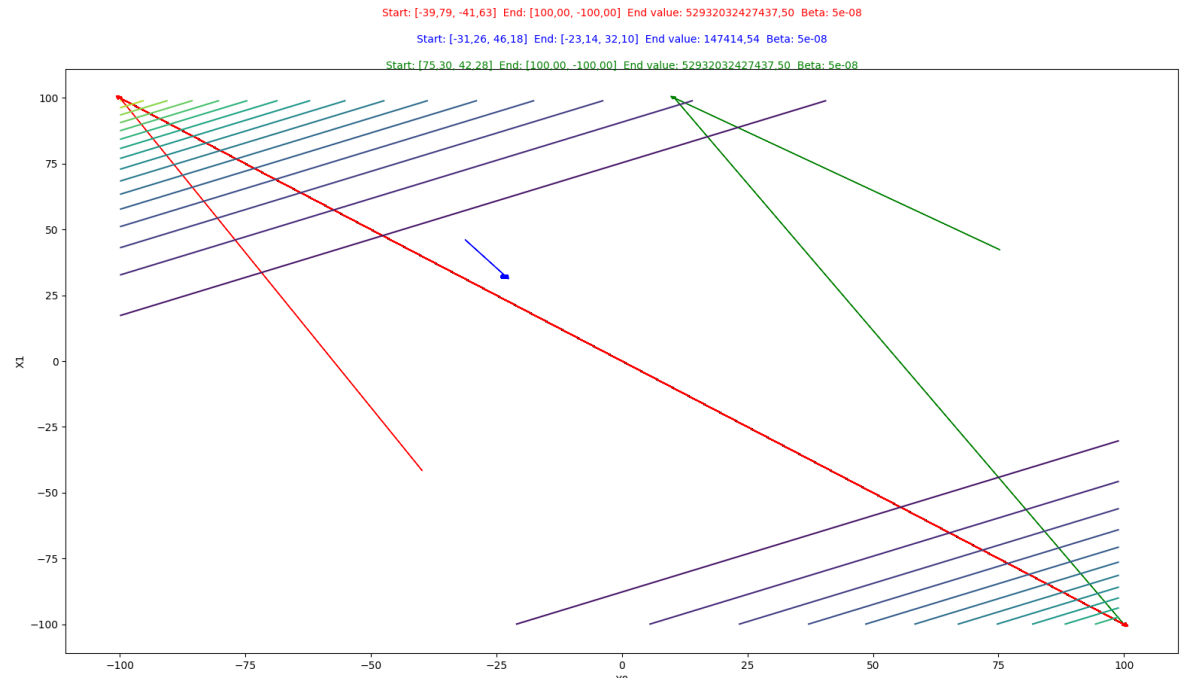


Nie udało mi się metodą ze stałym współczynnikiem beta wyznaczyć sensownego rozwiązania.

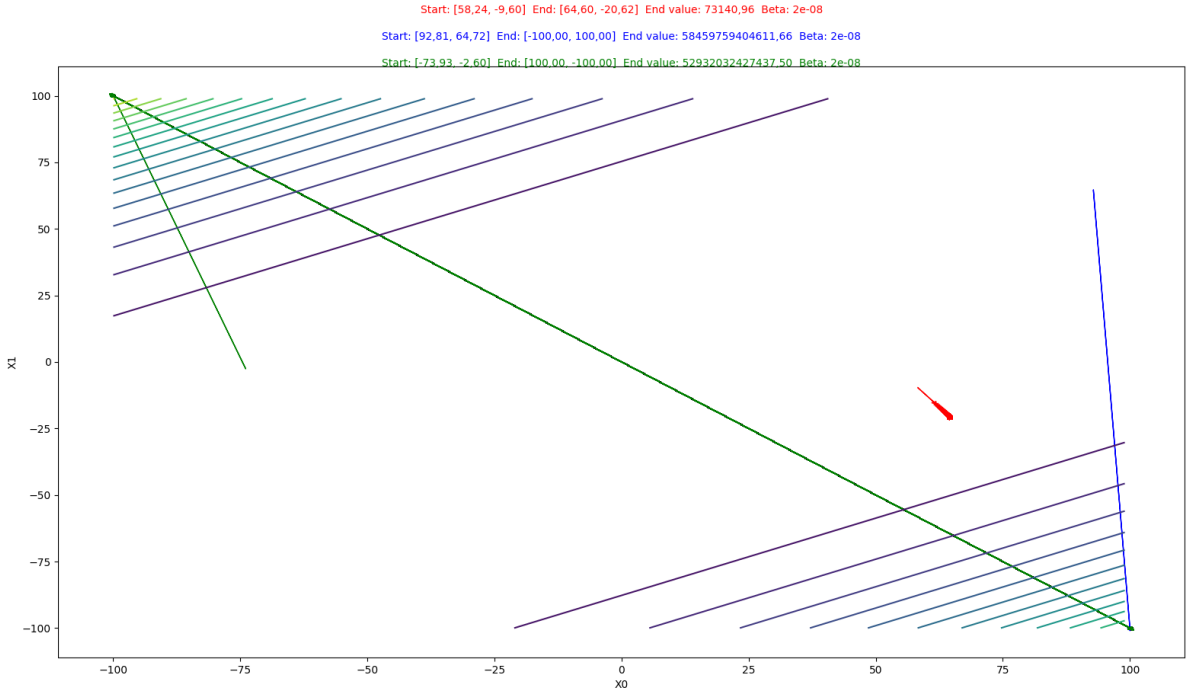
Funkcja f3

Dla funkcji f3 znalezienie minimum przy stałym współczynniku beta jest możliwe, ale bardzo czasochłonne, ponieważ f3 podczas zbliżania się do minimum mocno „zwalnia”.

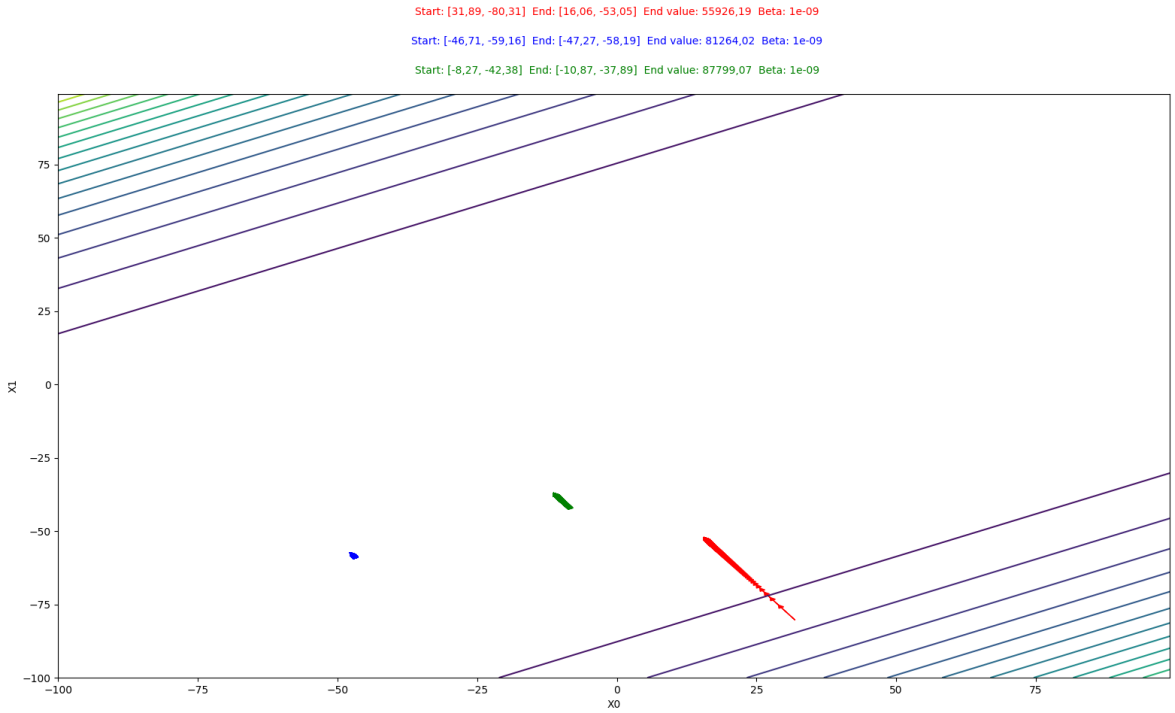
Dla Beta = 5e-08 zachodzą oscylacje, w pojedynczym przypadku punkt zaczął gdzieś zmierzać:



Dla beta = 2e-08 funkcja raz wpadła w oscylacje, a w pozostałych przypadkach znajduje się daleko od optimum:



Dla mniejszego współczynnika beta funkcja kończy się poprzez osiągnięcie limitu iteracji:



Aby zbadać minima funkcji f1, f2, f3 zdecydowałem się użyć metody Barzilai-Borwein, która polega na odpowiednim dostosowywaniu parametru beta w czasie trwania wykonywania procesu minimalizacji.

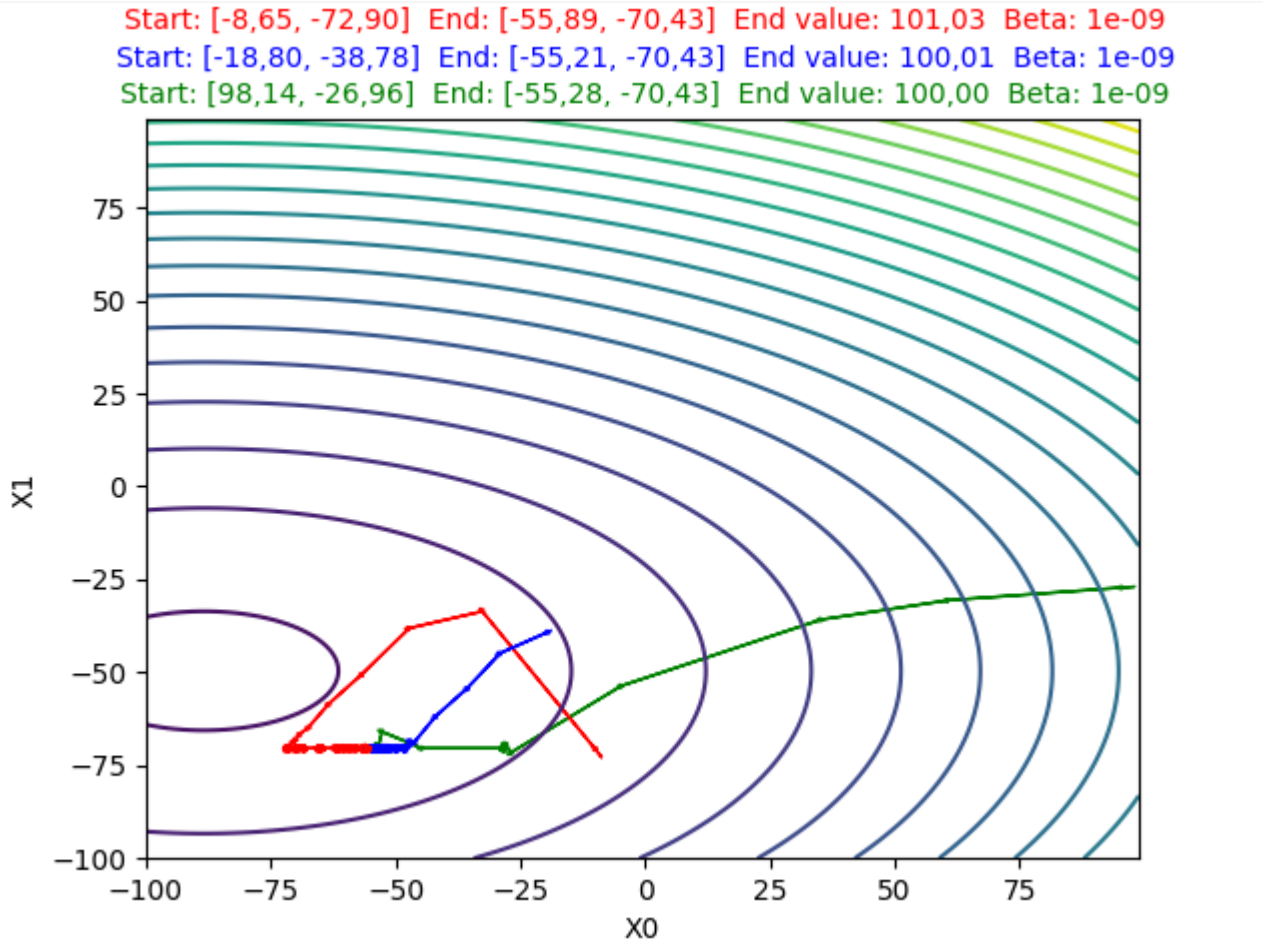
Metoda najszybszego wzrostu ze współczynnikiem kroku Barzilai-Borwein:

Tym razem współczynnik beta będzie się zmieniał zgodnie z wzorem na tak zwany ‘short BB step’:

$$[\text{short BB step}]\ \alpha_k^{SHORT} = \frac{\Delta x \cdot \Delta g}{\Delta g \cdot \Delta g}.$$

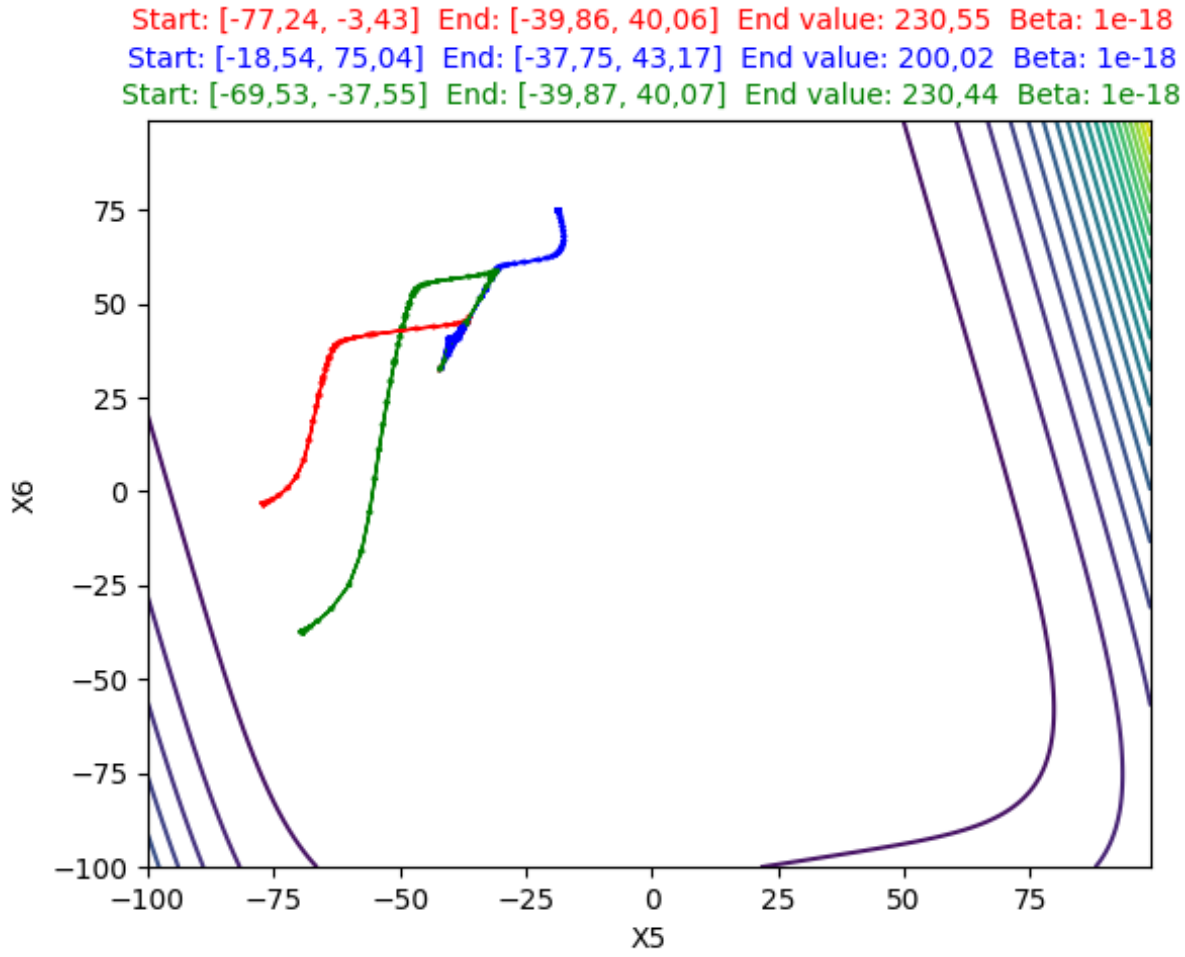
Dzięki zastosowaniu tej metody jesteśmy w stanie w sposób dość dokładny wyznaczyć optima:

Funkcja f1:



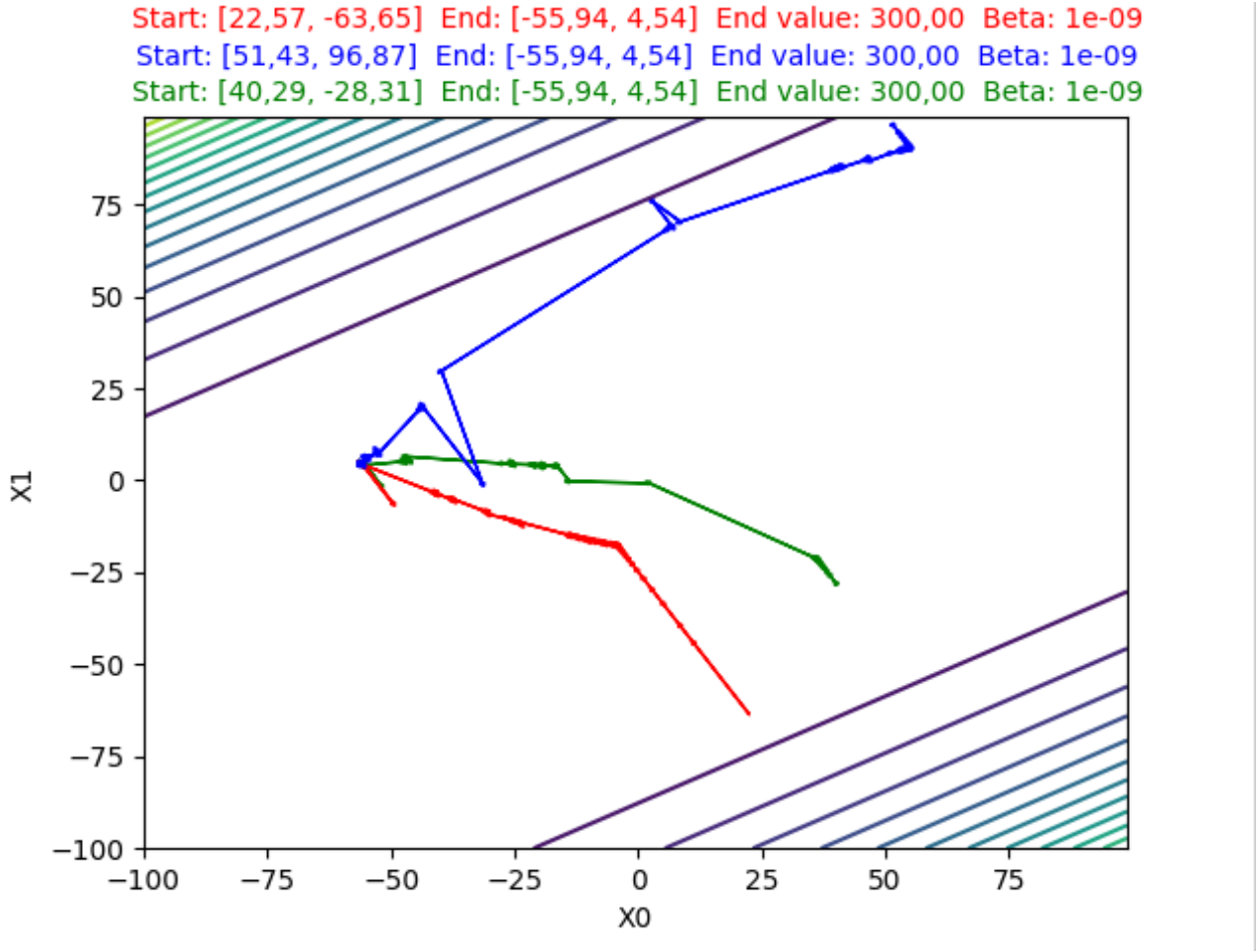
Uzyskane minima różnią się w sposób pomijalny od siebie. Dzięki zastosowaniu metody Barzilai-Borwein minimum jest uzyskiwane bardzo szybko i wynosi ono około 100.

Funkcja f2:



Dla funkcji f2 również udało się dzięki metodzie Barzilai-Borwein uzyskać minimum, które wynosi 200.

Funkcja f3:



Dla f3 ze wszystkich wylosowanych punktów zatrzymujemy się w optimum wynoszącym około 300.

Zalety/wady algorytmu:

- metoda najszybszego wzrostu ze stałą betą:

- jest prosta w implementacji
- dla prostych funkcji takich jak booth działa bardzo dobrze
- nie jest w stanie radzić sobie ze skomplikowanymi funkcjami takimi jak f1, f2, f3
- jest wolna

- metoda najszybszego wzrostu – wersja Barzilai-Borwein:

- dobrze radzi sobie nie tylko z prostymi funkcjami, ale również z tymi bardziej skomplikowanymi
- jest szybki
- jest bardziej skomplikowany niż wersja ze stałą betą.

Wnioski:

Metoda najszybszego wzrostu sama w sobie działa tylko w niektórych przypadkach. Kiedy używałem stałego współczynnika beta była ona w stanie znaleźć minimum sprawnie tylko dla tak prostej funkcji jak booth. Niestety dla bardziej skomplikowanych funkcji algorytm nie zachowuje się w sposób oczekiwany – przy zbyt dużym współczynniku beta zachodzą oscylacje, natomiast przy zbyt małym funkcja nie zbiega w sposób wystarczająco szybki do swojego minimum. Optymalizacja za pomocą współczynnika beta, który jest dostosowywany na podstawie obecnego tempa i kierunku wzrostu funkcji, tak jak w metodzie Barzilai-Borwein, znacząco poprawia szybkość znalezienia minimum oraz przyczynia się do zwiększenia dokładności.