WSI, Ćwiczenie 1 część 2

Michał Kwarciński

Polecenie:

1. Zaimplementować metodę najszybszego wzrostu/spadku. Gradient wyliczyć numerycznie.
2. Narysować zachowanie algorytmu.
3. Zastosować metodę do znalezienia optimum funkcji booth w 2 wymiarach, po czym do znalezienia optimum funkcji o numerach od 1 do 3 z CEC 2017 w 10 wymiarach. Ograniczenia kostkowe przestrzeni to -100, 100.

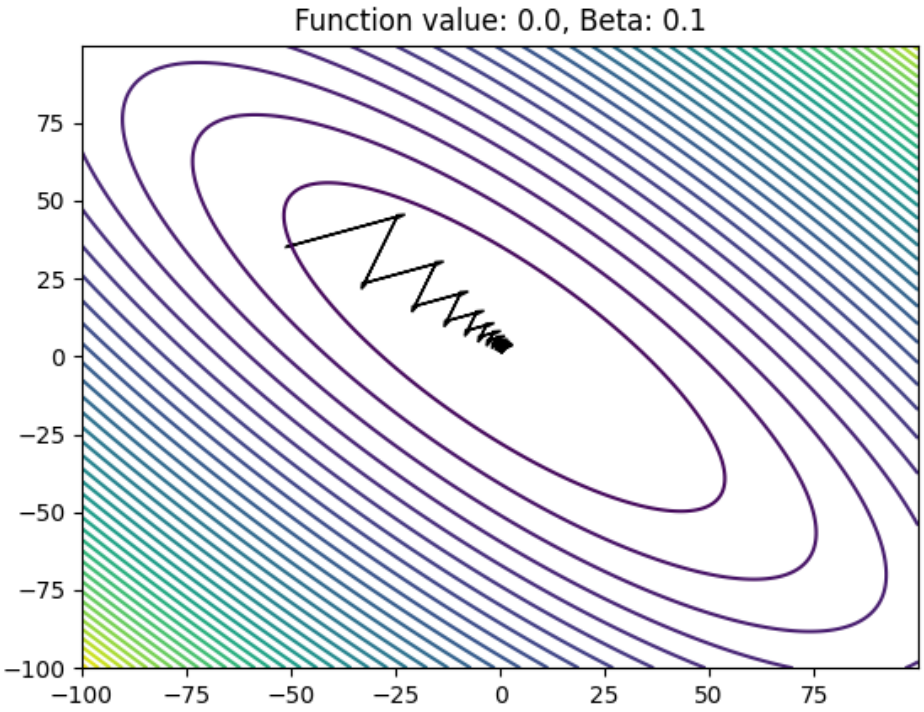
W celu uniknięcia sytuacji, w której z kroku na krok algorytm nie robi postępów, założyłem, że jeśli wszystkie wartości bezwzględne różnic między współrzędnymi kolejnych punktów będą mniejsze niż 1/10 bety (kroku), to algorytm zostanie przerwany.

Wartości funkcji w minimum zostały przybliżone do dwóch miejsc po przecinku.

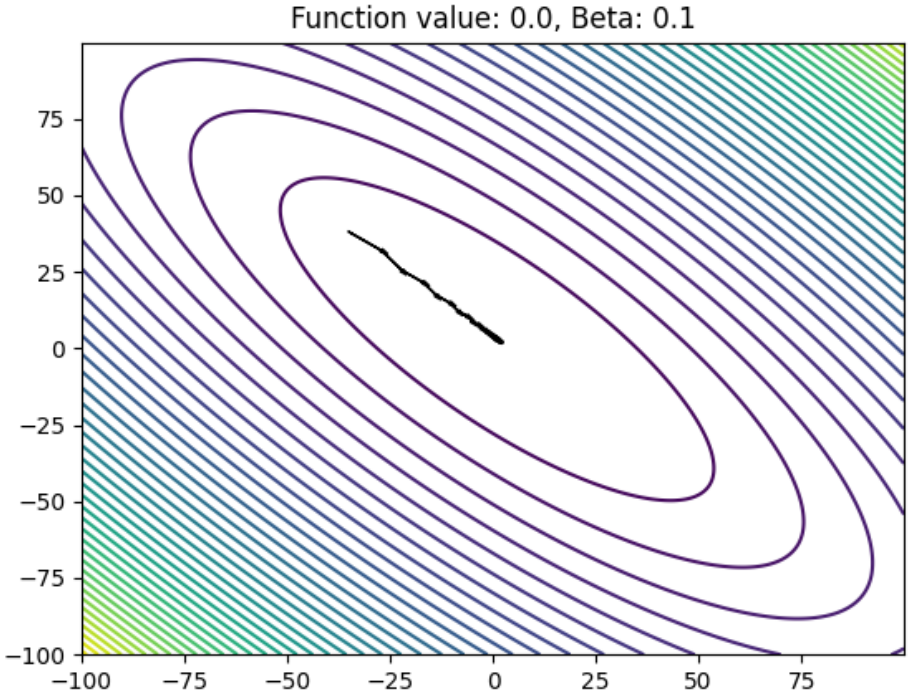
Funkcja booth:

Beta = 0,1:

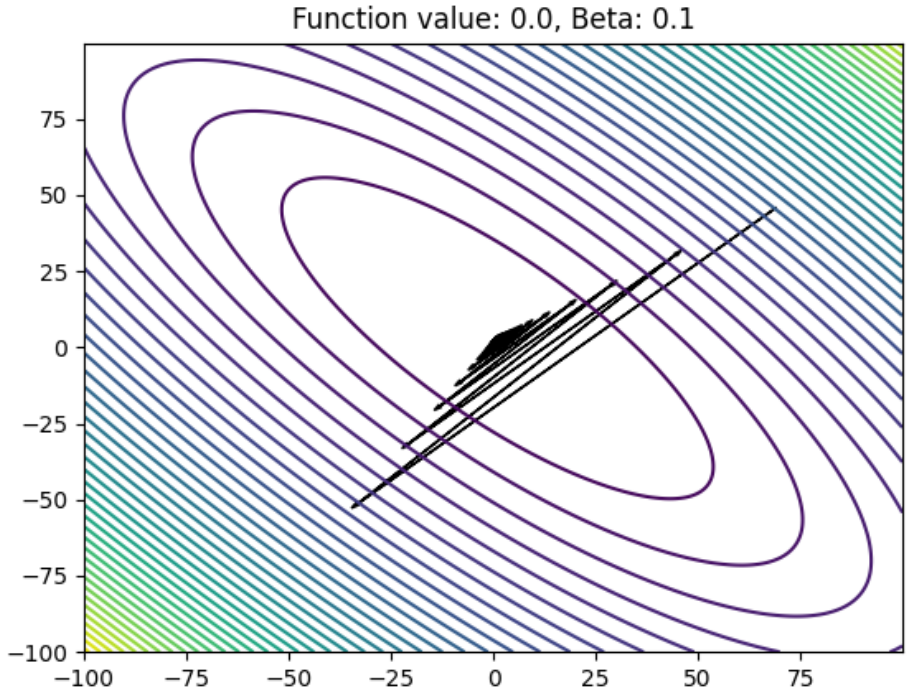
Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
-51,3; 35,05	0,99; 3,01	0,00



Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
-35,2; 38,15	0,97; 3,04	0,00

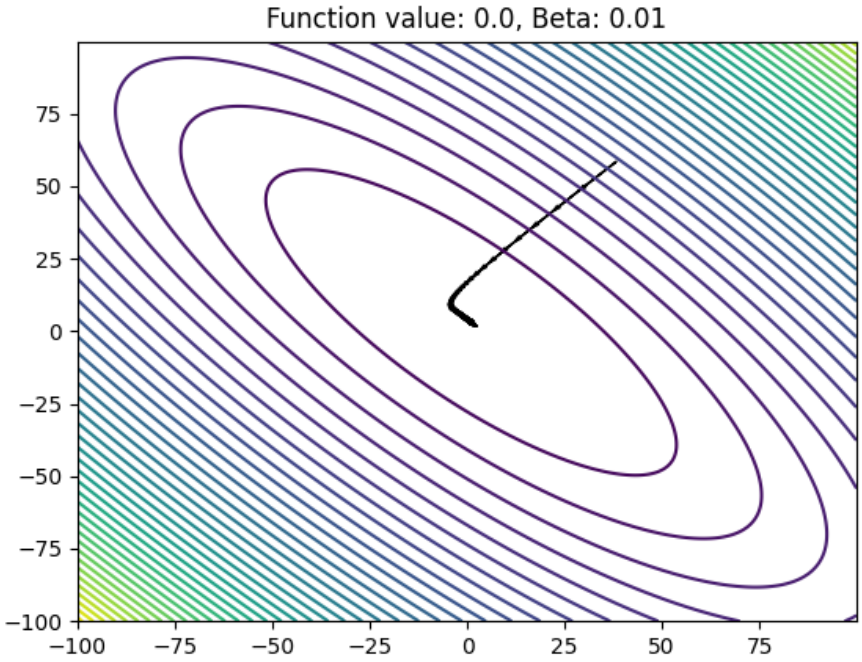


Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
69,04; 45,8	0,99; 2,99	0,00

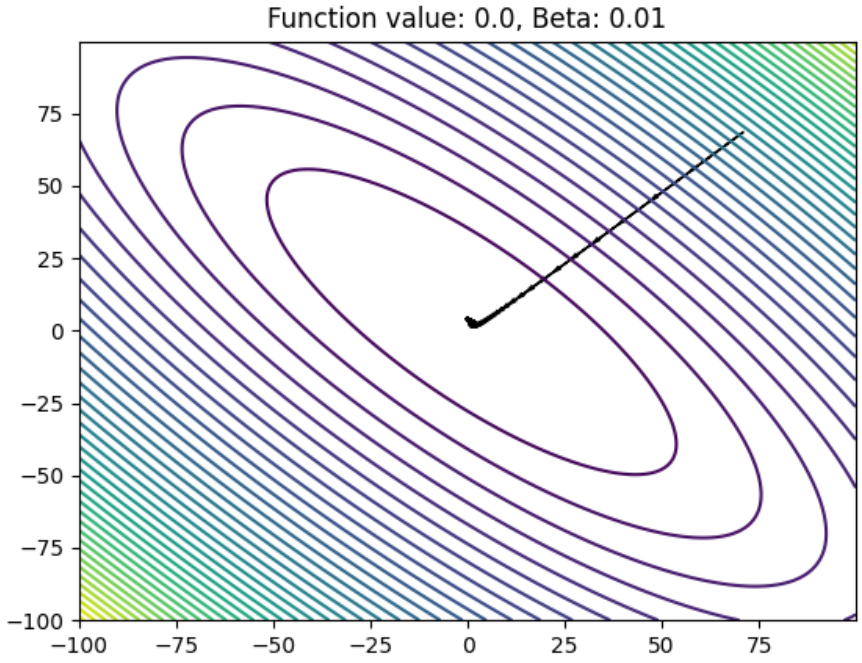


Beta = 0,01:

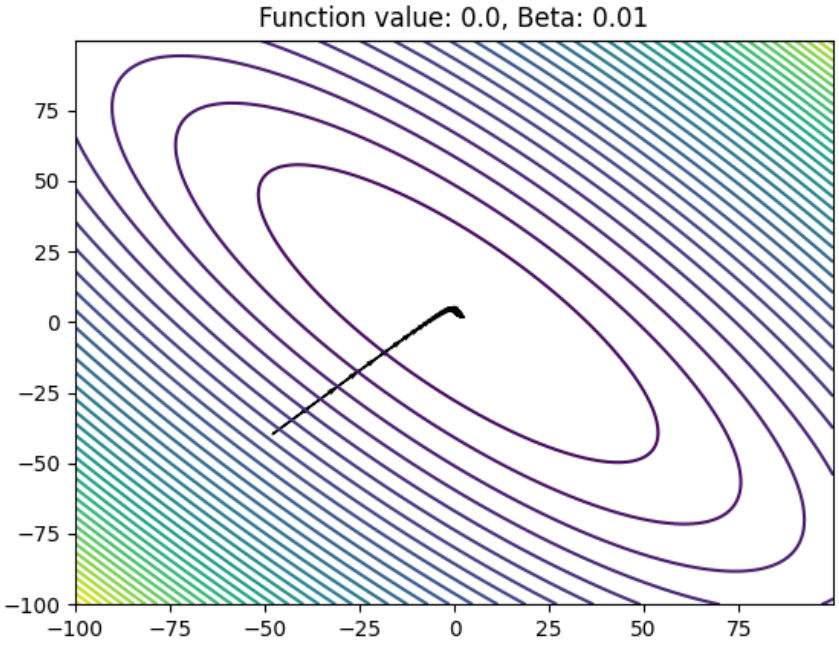
Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
38,30; 58,48	0,95; 3,04	0,00



Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
70,97; 68,56	1,00; 2,99	0,00



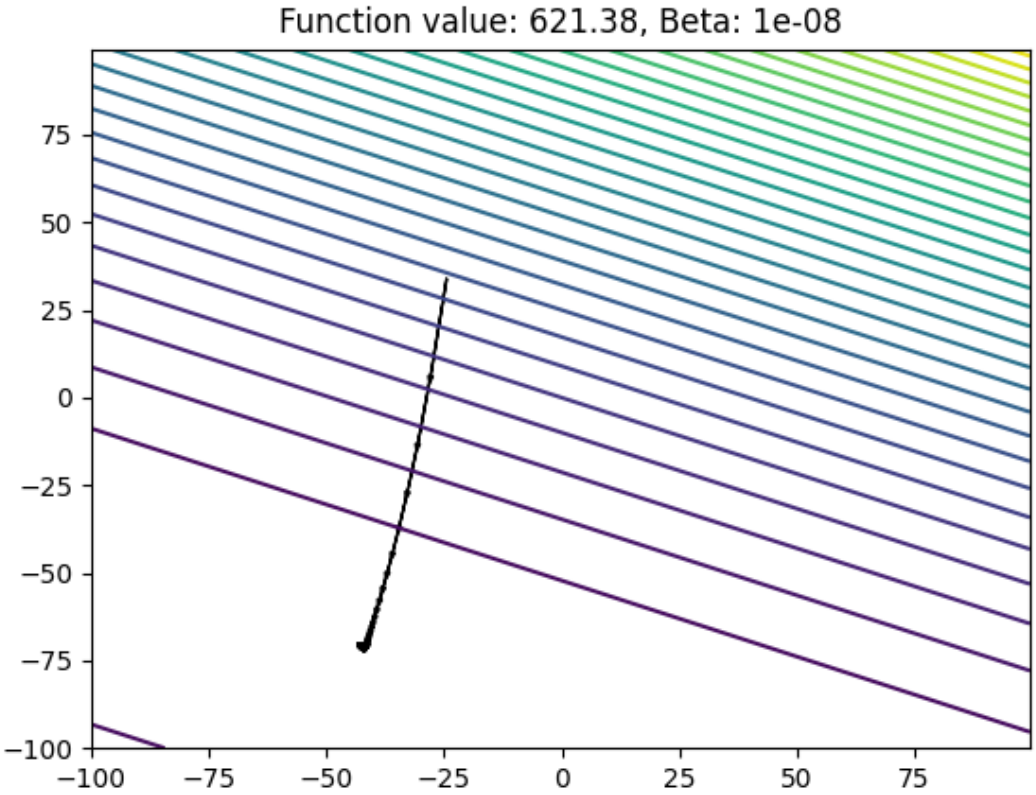
Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
-47,99; -39,73	0,99; 3,00	0,00



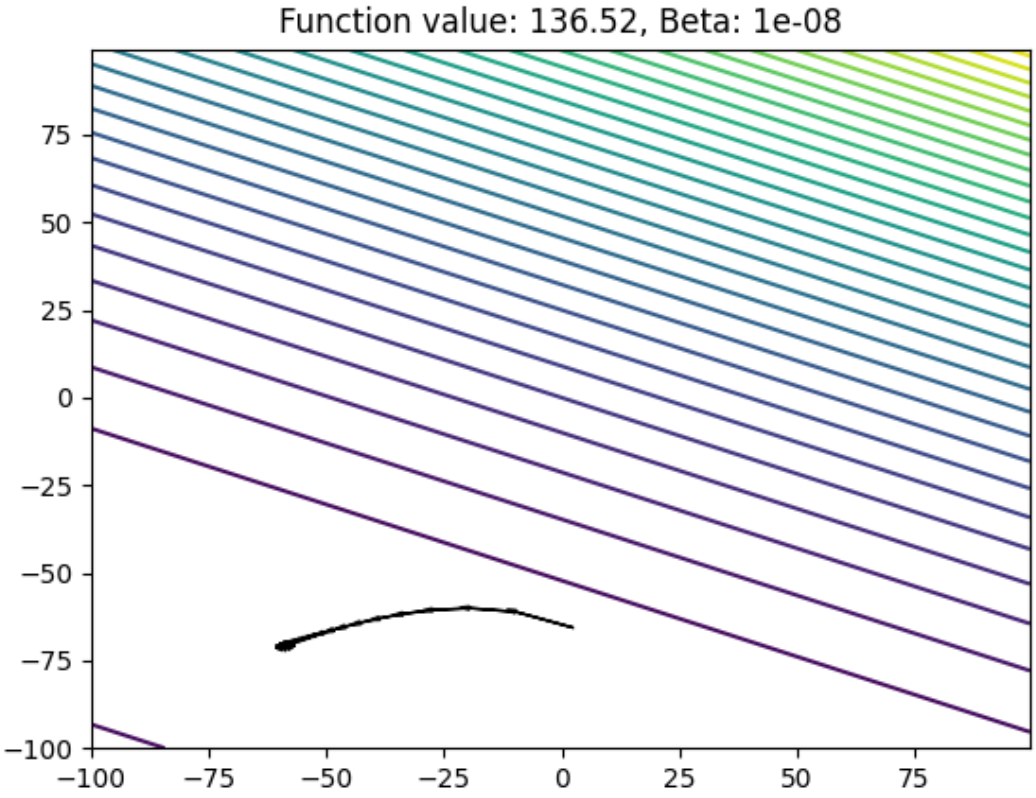
Funkcja f1

Beta = 10^{-8}:

Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
-24,47; 34,04; 38,54; 4,08; -39,07; 35,66; 81,96; 80,24; 40,0; 21,92	-41,55; -70,43; -29,61; -58,33; 22,09; 59,94; 44,67; 18,56; 76,68; -43,74	621,38

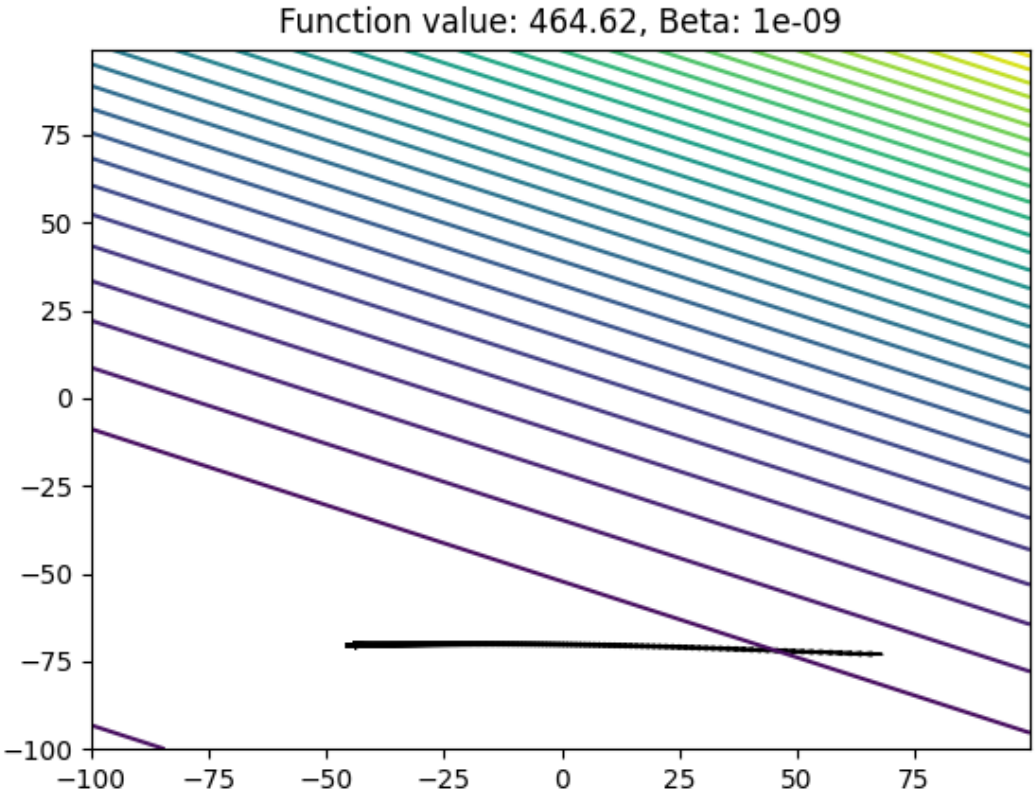


Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
2,43; -65,64; 10,88; 30,74; -53,58; 55,89; 47,18; 22,47; 47,71; 68,42	-58,91; -70,43; -29,61; -58,33; 22,09; 59,94; 26,84; 18,56; 76,68; -29,1	136,52



Beta = 10^{-9}:

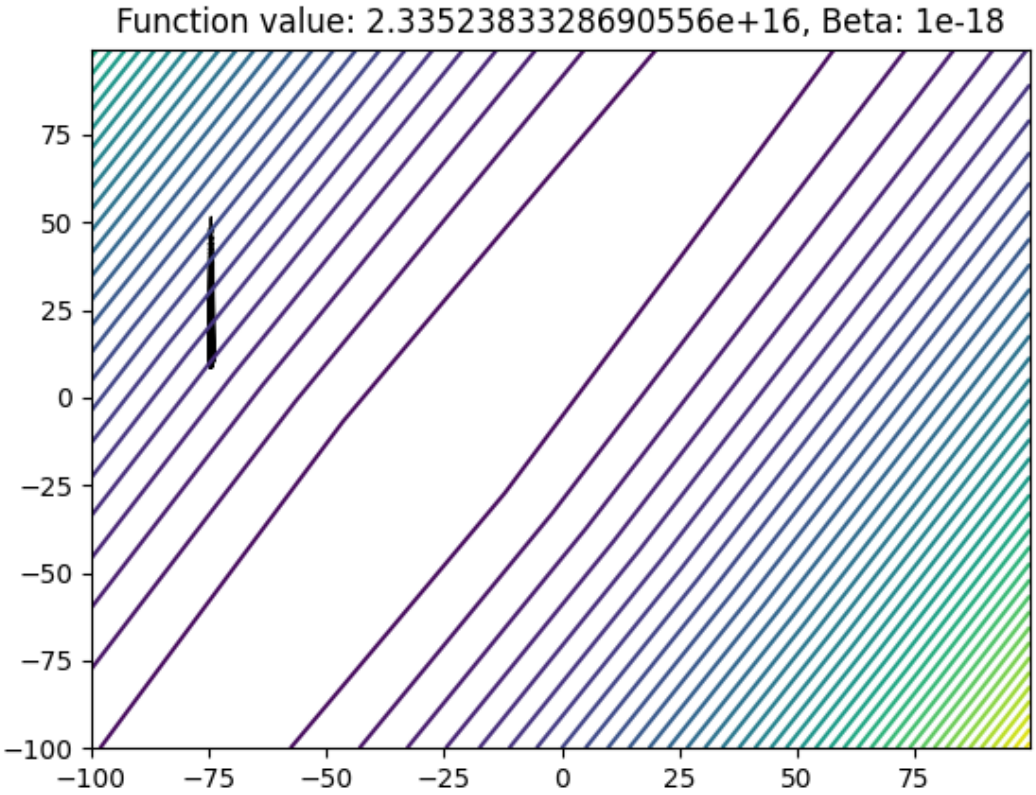
Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
68,14; -73,04; -42,13; -44,02; -52,77; 18,21; -60,25; 79,57; 48,35; -34,08	-43,79; -70,43; -29,61; -58,33; 22,09; 59,94; 42,36; 18,56; 76,68; -41,85	464,62



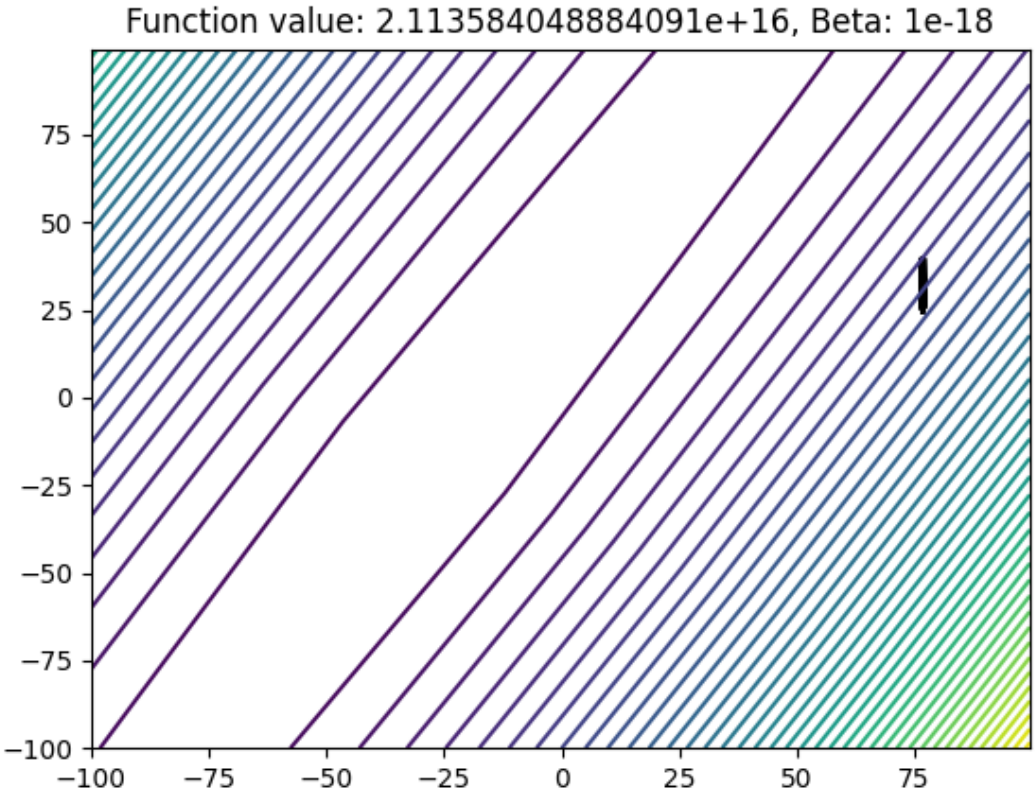
Funkcja f2

Beta = 10^{-18}:

Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
-74,54; 51,56; 38,94; -69,05; 23,85; -10,78; -26,22; 17,85; -22,82; 45,49	-43,79; -70,43; -29,61; -58,33; 22,09; 59,94; 42,36; 18,56; 76,68; -41,85	2,34e+16

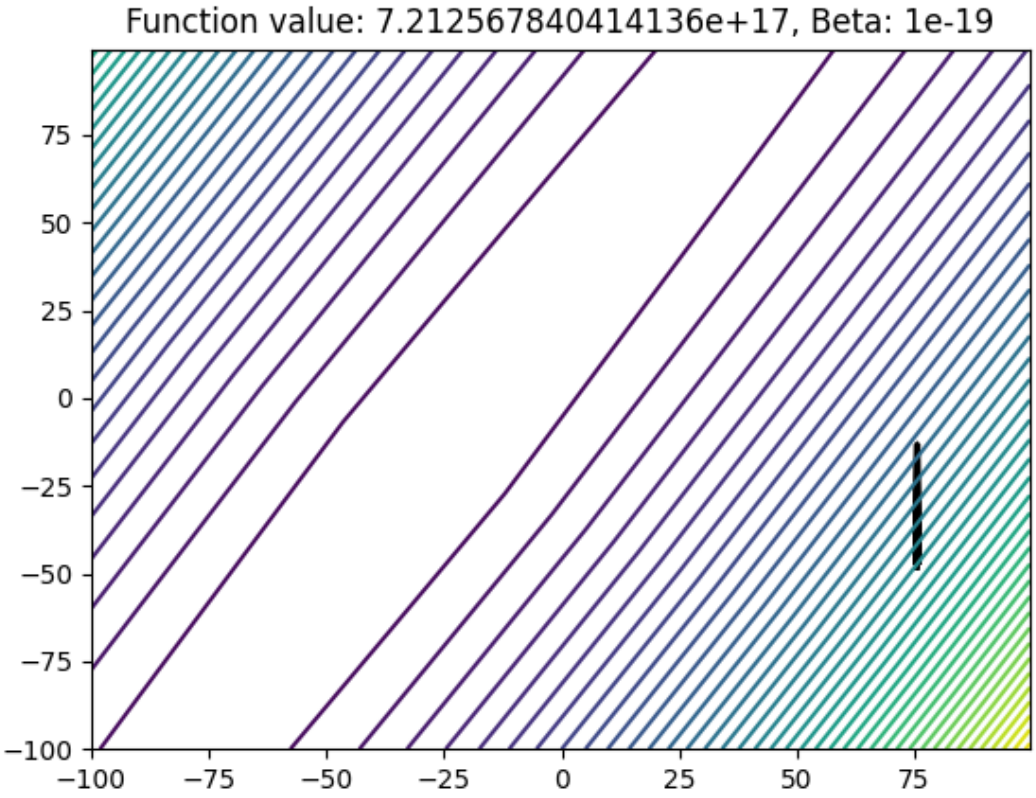


Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
76,71; 39,96; -28,34; -86,54; 56,48; 85,73; 30,04; 68,48; 28,39; 44,35	-43,79; -70,43; -29,61; -58,33; 22,09; 59,94; 42,36; 18,56; 76,68; -41,85	2,11e+16



Beta = 10^{-19}:

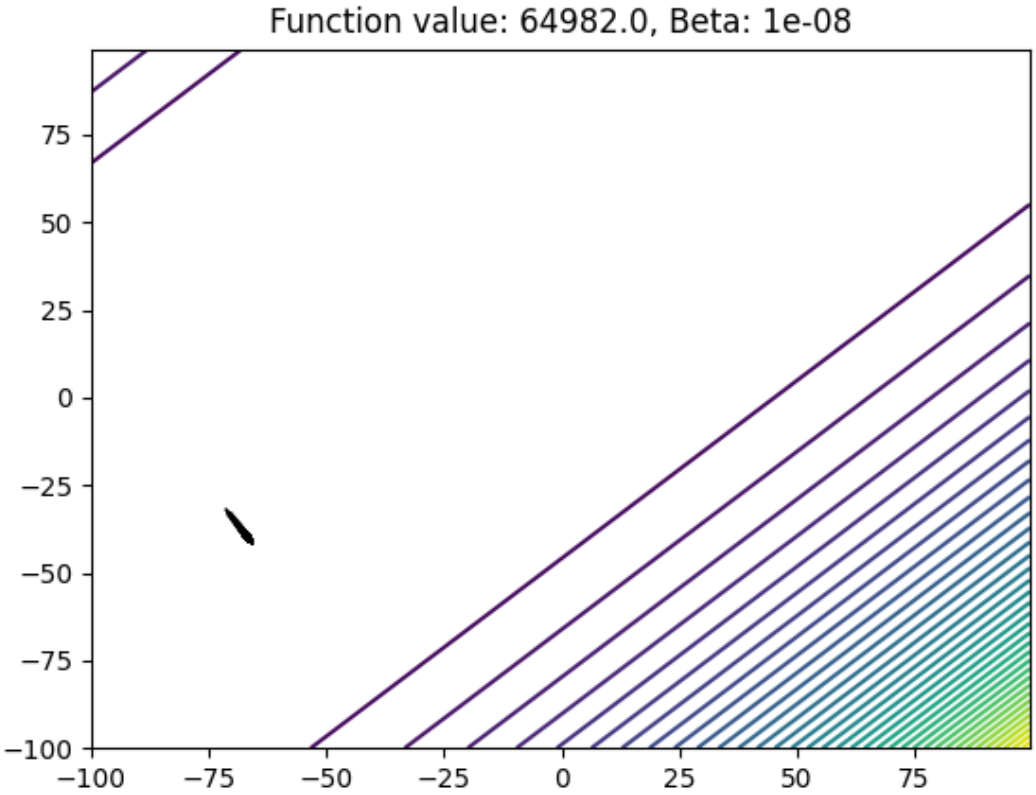
Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
75,54; -12,51; 67,27; -21,57; -77,81; -70,36; -10.55; -77,77; -60,19; -57,34	-43,79; -70,43; -29,61; -58,33; 22,09; 59,94; 42,36; 18,56; 76,68; -41,85	7,21e+16



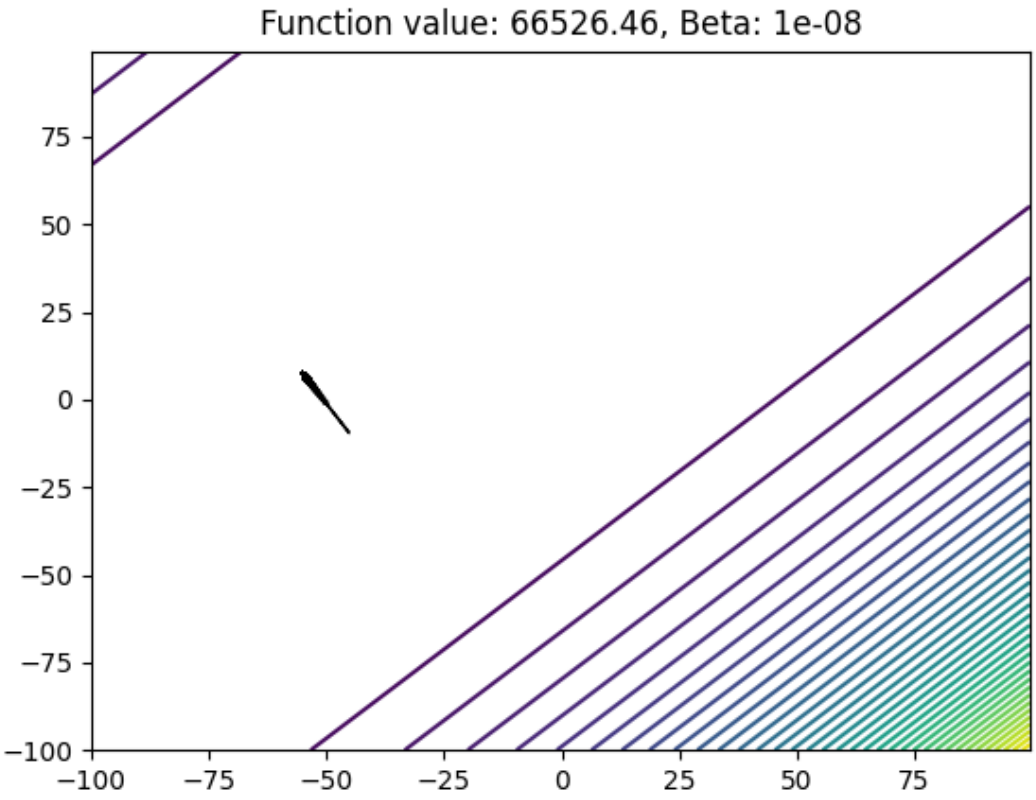
Funkcja f3

Beta = 10^{-8}:

Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
75,54; -12,51; 67,27; -21,57; -77,81; -70,36; -10.55; -77,77; -60,19; -57,34	-66,8; -39,82; -61,93; 57,77; 11,91; -90,27; 66,65; 53,45; -50,68; -47,79	64982,0

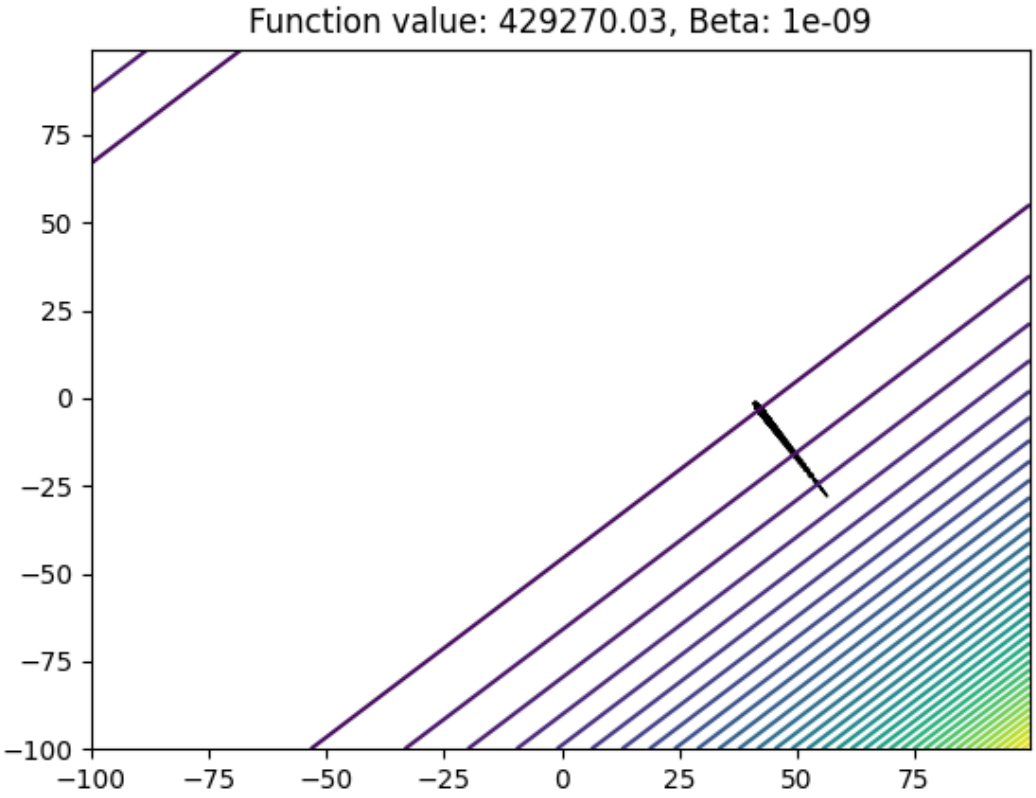


Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
-45,19; -9,48; 109,61; -85,07; -15,74; 29,74; 36,36; 34,23; 36,16; 44,06	-54,32; 6,23; 106,05; -87,0; -13,5; 16,02; 1,.65; 32.29; 43,59; 39,16	66526,46



Beta = 10^{-9}:

Punkt startowy	Punkt uznany za optimum	Wartość funkcji w optimum
56,43; -27,96; -83,81; 72,97; 96,77; -51,46; 49,37; -48,14; -76,92; 85,69	-43,79; -70,43; -29,61; -58,33; 22,09; 59,94; 42,36; 18,56; 76,68; -41,85	429270,03



Pytania

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

- Duża beta powoduje większe skoki, daje możliwość szybszego zbliżenia się do optimum. Używanie jej może jednak powodować niepożądane działanie algorytmu. Wejdzie on w oscylacje, które spowodują brak możliwości znalezienia żadnego optimum.
- Mała beta powoduje małe kroki, co może skutkować nieosiągnięciem minimum w określonej liczbie iteracji algorytmu.
- Dla funkcji booth wystarczająca beta wynosiła 0,01, gdyż 0,1 skutkowała agresywnym zachowaniem algorytmu.
- Dla funkcji f1 parametr beta zostawiłbym na wartości 10^{-8}– dała ona najlepszy wynik.
- Najwyższa wartość bety dla funkcji f2, przy której algorytm wykonał 1000 iteracji bez osiągnięcia RuntimeWarning, to 10^{-18}.
- Dla funkcji f3 działający parametr beta wyniósł 10^{-9}.

Zalety/wady algorytmu?

Zalety:

- Czas działania algorytmu dla nieskomplikowanych funkcji jest bardzo niski.
- Algorytm, przy odpowiednim kroku (nie za dużym), znajdzie przynajmniej minimum lokalne.

Wady:

- Nie ma gwarancji znalezienia minimum globalnego.
- Im bardziej skomplikowana funkcja, tym gorzej algorytm sobie radzi.

Wnioski

Uruchomienie funkcji booth ukazuje pożądane działanie algorytmu.
Dla funkcji cec2017 ciężko powiedzieć, czy algorytm działał poprawnie. Dla f1 jest możliwe odczytanie informacji z grafik, natomiast dla f2 i f3 jest to zdecydowanie trudniejsze. Owe funkcje musiały korzystać z bardzo małej bety, inaczej szybko wpadały w oscylacje prowadzące do zbliżenia się do nieskończoności (RuntimeWarning).
Funkcje f1, f2, f3 trafiały na co najwyżej minima lokalne.
Algorytm nie gwarantuje uzyskania najlepszego wyniku. Przy zbyt dużych betach algorytm może zacząć oscylować i nie znajdzie oczekiwanego rozwiązania. Dla małych parametrów beta algorytm działa wolno.