WSI – Raport z zadania 1

Michał Pędziwiatr

Numer indeksu: 331421

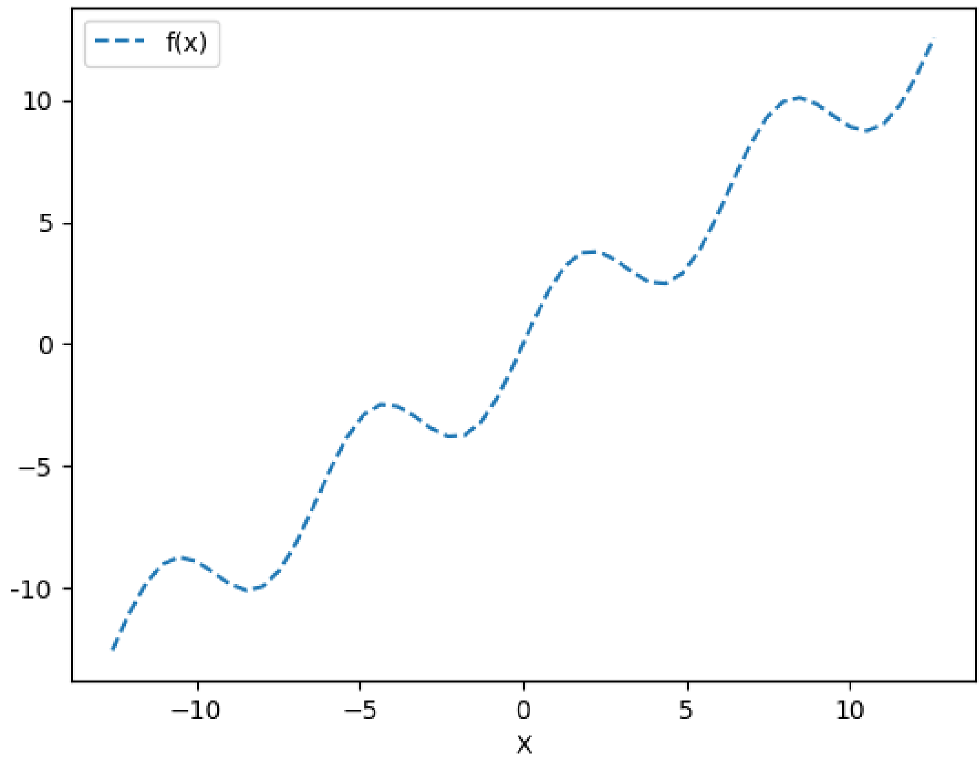
Wstępne założenia

Metoda gradientu prostego to algorytm mający na celu odnalezienie współrzędnych ekstremum funkcji z jak największą dokładnością oraz wydajnością jednocześnie. By to uczynić wykorzystuje on wzór: $x_{k+1} = x_k + \alpha_k * d_k$, w którym x_k oraz x_{k+1} oznaczają odpowiednio obecną oraz następną pozycje algorytmu, d_k to wektor nadający mu kierunek, a α_k stanowi współczynnik długości kroku, znany także jako „learning rate".

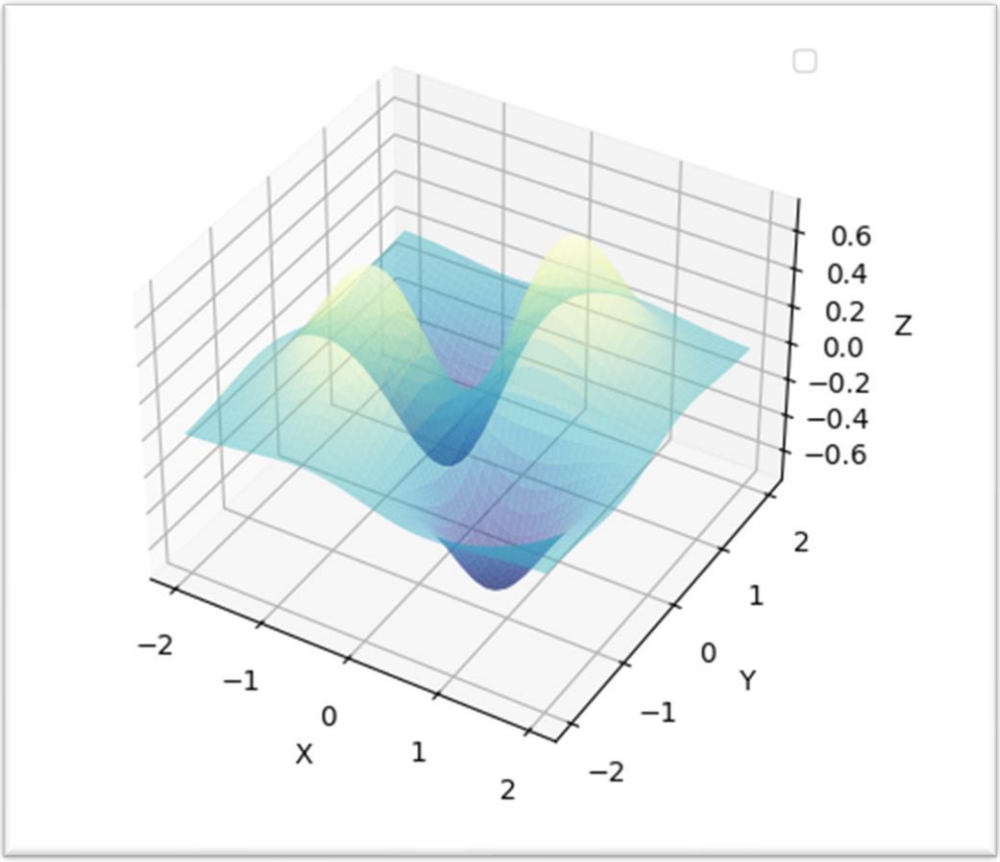
W praktyce lokalizacja algorytmu obliczana jest więc poprzez przesunięcie go z jego obecnego położenia o iloczyn gradientu, czyli wektora składającego się z pochodnych funkcji (lub pojedynczej pochodnej w przypadku funkcji jednej zmiennej) oraz stałego współczynnika. Warto dodać, że od znaku owego współczynnika zależeć będzie, czy algorytm dążyć będzie do maksimum czy minimum lokalnego funkcji.

W tym zadaniu gradient prezentowany będzie na dwóch funkcjach zaprezentowanych na poniższych wykresach:

- f(x) = x + sin(x):



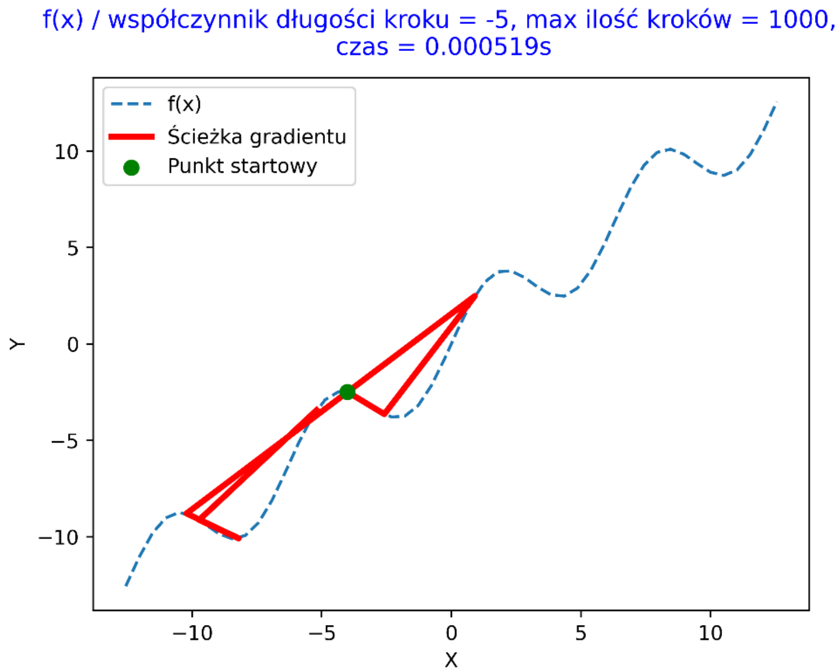
- $g(x,y) = \frac{4xy}{e^{x^2+y^2}}$:

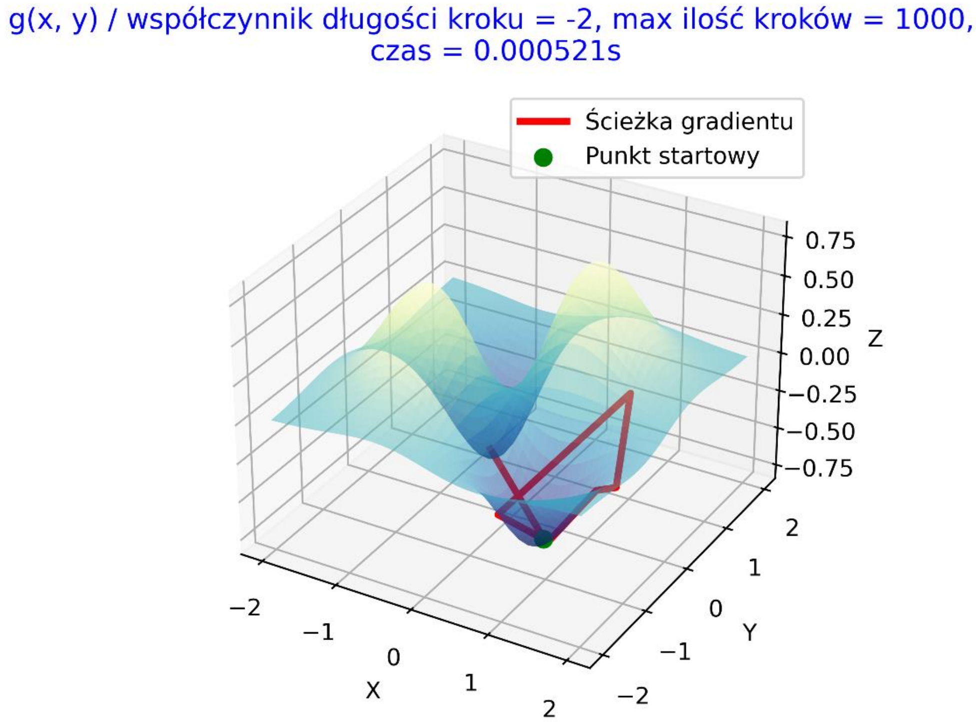


Wpływ parametrów funkcji „grad_descent” na proces optymalizacji

Parametry takie jak długość kroku, limit maksymalnej ich liczby oraz rozmieszczenie punktu startowego potrafią w zależności od sytuacji, mieć duże znaczenie na ostateczny wynik działania algorytmu. Kluczowe jest więc dobranie odpowiednich wartości odpowiadających naszym potrzebom, oraz funkcji na której operujemy.

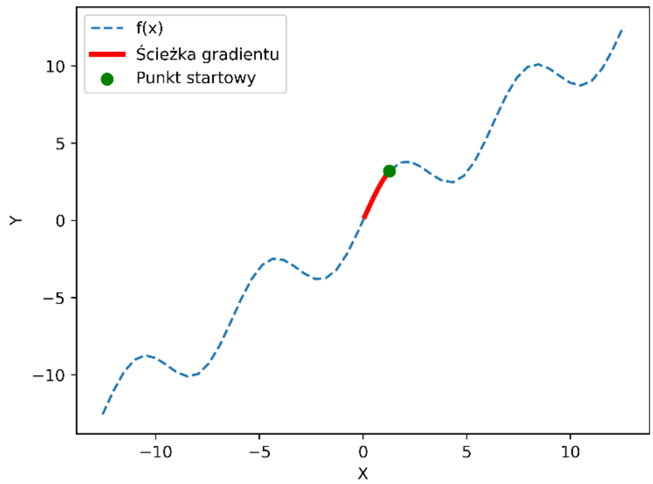
Pomimo iż zwiększenie współczynnika długości kroku często zoptymalizuje działanie algorytmu przyśpieszając jego działanie, należy pamiętać, że dzieje się to kosztem mniejszej dokładności. Wynika to z „przestrzeliwania" szukanego ekstremum co w skrajnych przypadkach (takich jak widoczne poniżej) może znacznie oddalić algorytm od celu, tym samym kompletnie zaburzając wyniki końcowe. Widać to szczególnie wyraźnie na przykładzie funkcji f(x) oraz g(x,y) ze względu na ich strome krawędzie:



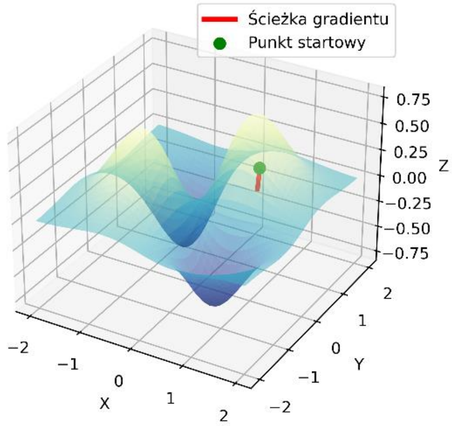


Nie tylko zbyt duże wartości współczynnika α_k mogą negatywnie wpłynąć na nasz wynik. Mimo tego, iż z założenia zmniejszenie długości kroku algorytmu zwiększy jego dokładność, to wymaga to odpowiednio wysokiego (lub braku) limitu kroków jakich może on wykonać. W przeciwnym wypadku skończyć się to może brakiem możliwości zbliżenia się do ekstremum, kompletnie burząc plan zwiększenia dokładności.

f(x) / współczynnik długości kroku = -0.001, max ilość kroków = 500,
czas = 0.000503s

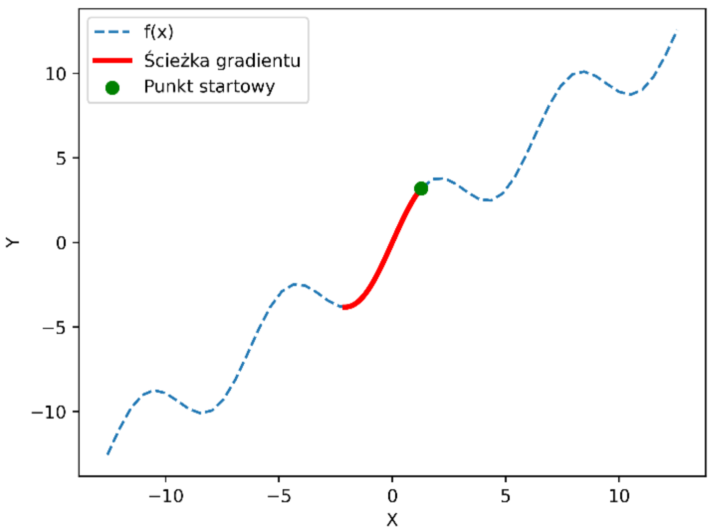


g(x, y) / współczynnik długości kroku = -0.001, max ilość kroków = 100,
czas = 0.000999s

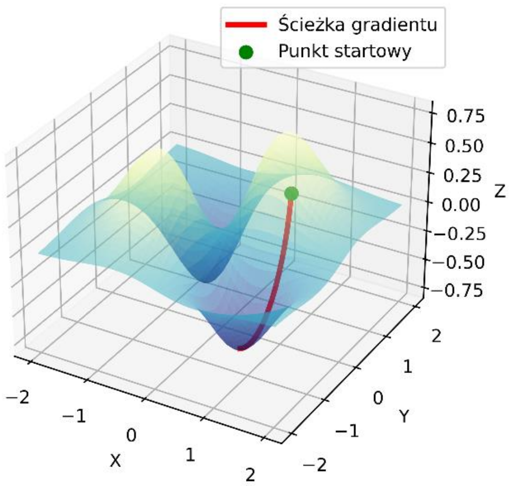


Tak już zostało to wspomniane, mimo wszystko większa dokładność sprawia, że niskie wartości długości kroku uczącego często są najlepszą decyzją przy projektowaniu algorytmu gradientu prostego. Należy jednak pamiętać by odpowiednio dostosować do nich dozwoloną ilość kroków.

f(x) / współczynnik długości kroku = -0.001, max ilość kroków = 5000
czas = 0.005196s



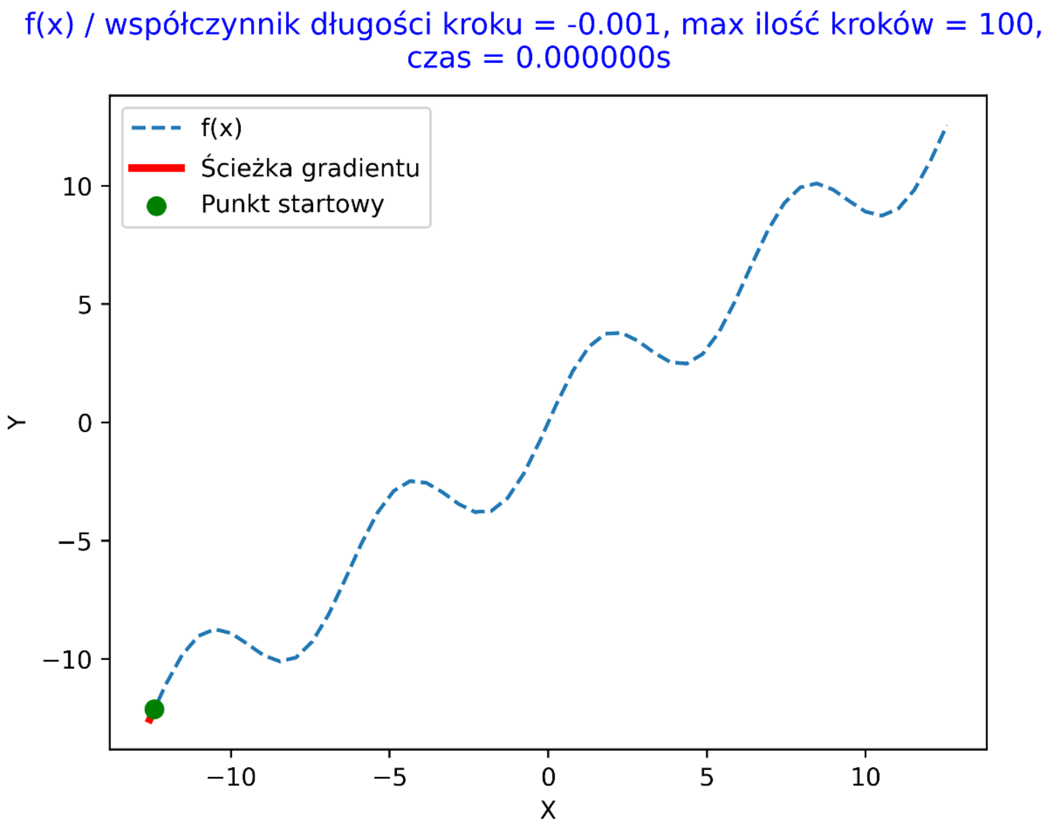
g(x, y) / współczynnik długości kroku = -0.001, max ilość kroków = 5000,
czas = 0.028477s



Dobrym pomysłem może być także wprowadzenie nowego ograniczenia działania funkcji, takiego jak kryterium stopu czy zbieżności, co potencjalnie oszczędziłoby moc obliczeniową przy wynikających z tego wyższych ilościach iteracji. W przypadku naszych funkcji odległości pomiędzy ekstremum są jednak dość podobne, co sprawia że limit będący ilością iteracji spełnia swoje zadanie bez znaczącej utraty wydajności.

Jak można się domyślić, kolejnym ważnym aspektem w kontekście poszukiwania ekstremum jest także lokalizacja punktu startowego algorytmu. To od niej zależy, do którego ekstremum lokalnego, a być może i globalnego będzie dążyć gradient. Różne lokalizacje mogą także wydłużyć lub skrócić drogę algorytmu, tym samym umożliwiając lub uniemożliwiając dotarcie do ekstremum w danej ilości iteracji.

Przykładem tego jak duże znaczenie może mieć wybór punktu startowego jest poniższy przypadek, gdzie jego lokalizacja znajdowała się tuż przy minimum, którego szukała wyznaczonego przez koniec funkcji:

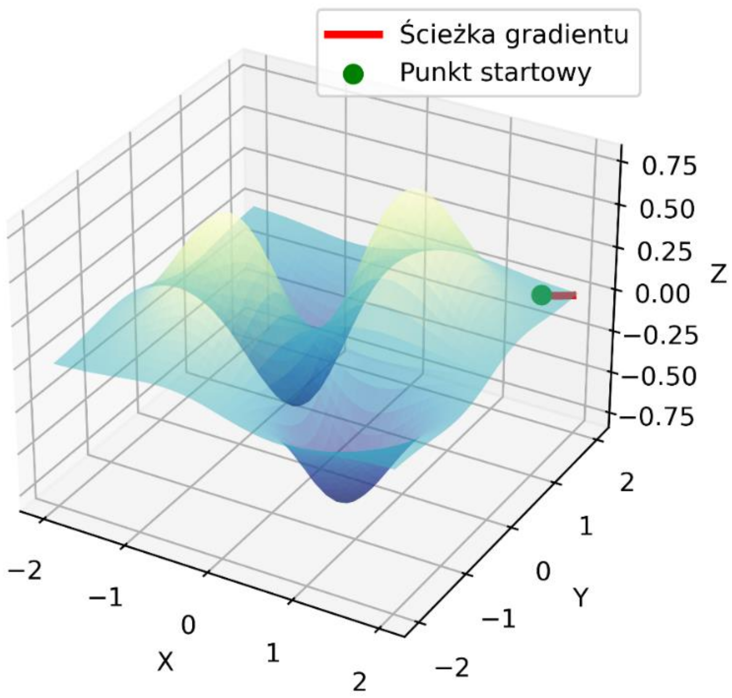


```
Funkcja f(x) : współczynnik długości kroku: 0.001, max ilość kroków: 100, czas: 0.000000s
Punkt startowy: x = -12.420697512884814, y = -12.130380643260635
Punkt końcowy: x = -12.564342242951149, y = -12.56028550291687

Funkcja f(x) : współczynnik długości kroku: 0.001, max ilość kroków: 5000, czas: 0.000000s
Punkt startowy: x = -12.420697512884814, y = -12.130380643260635
Punkt końcowy: x = -12.564342242951149, y = -12.56028550291687
```

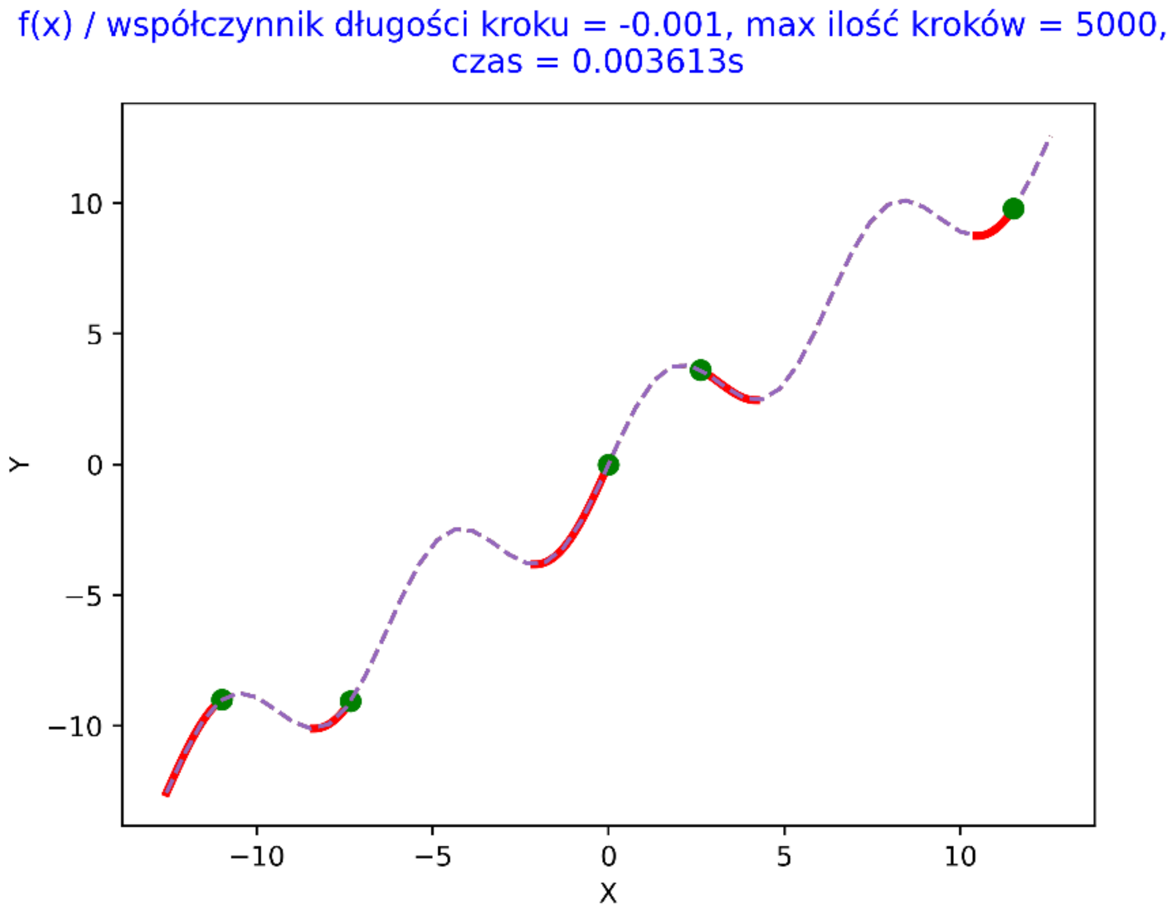
Jak widać pomimo faktu, iż wybrana wartość współczynnika długości kroku jest bardzo mała, przez nietypowe położenie punktu startowego wyniki pomiędzy 100 a 500 iteracjami nie różnią się wcale.

Warto nadmienić, że pozycja punktu początkowego może również zmylić algorytm, doprowadzając do wyniku nie będącym ekstremum funkcji. Dziać się może tak w funkcjach z określoną dziedziną (tak jak w naszym przypadku), lub w innych miejscach nieciągłości funkcji. Jest to spowodowane tym, że algorytm szukając ekstremum w pewnym sensie przewiduje jego położenie, bez rzeczywistego „patrzenia w przód”. Gdy pochodna, a więc także kąt nachylenia funkcji wskazuje na to, że podążając daną drogą natknie się na ekstremum, będzie nią właśnie podążał. Punkty nieciągłości jednak z założenia nie są spójne, dlatego nie zawsze da się je przewidzieć w taki sposób. Zjawisko to dobrze widoczne jest przy funkcji g(x,y), ze względu na jej stosunkowo strome krawędzie:



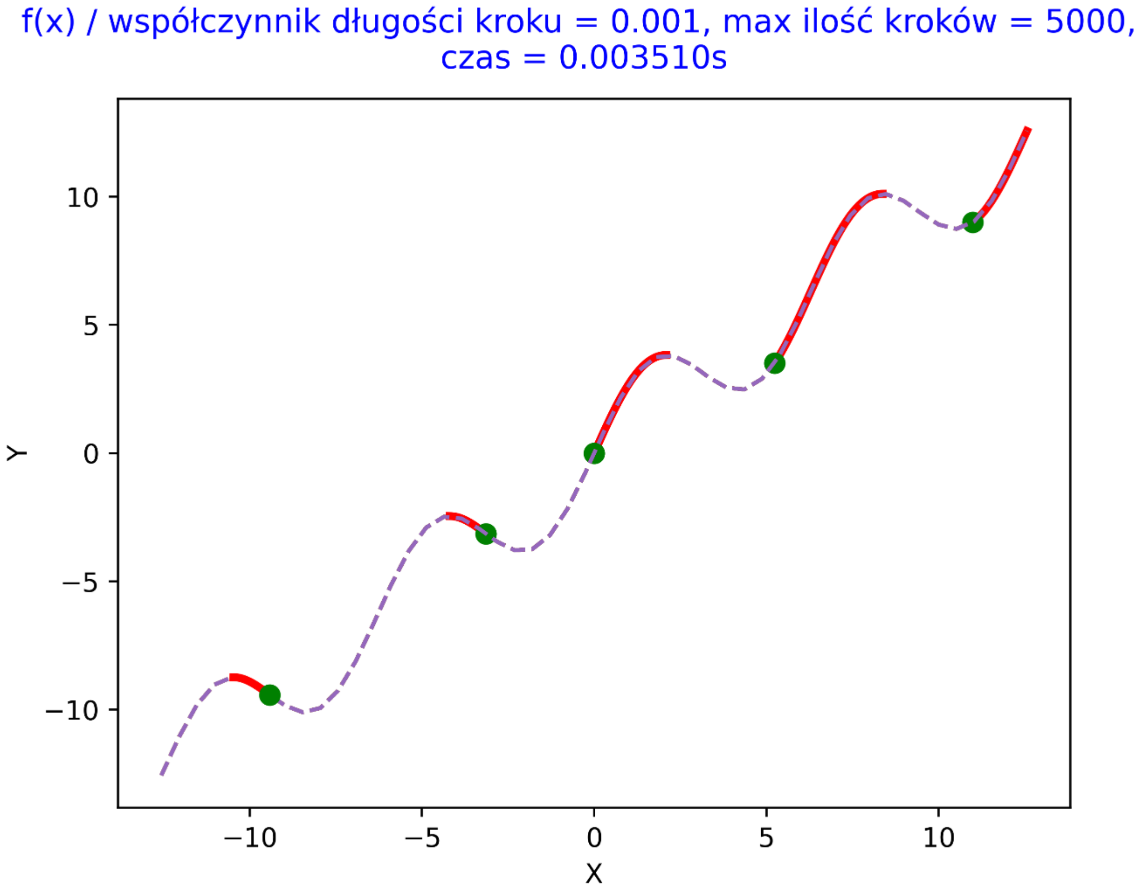
Ekstrema funkcji

Wszystkie Minimum funkcji f(x)



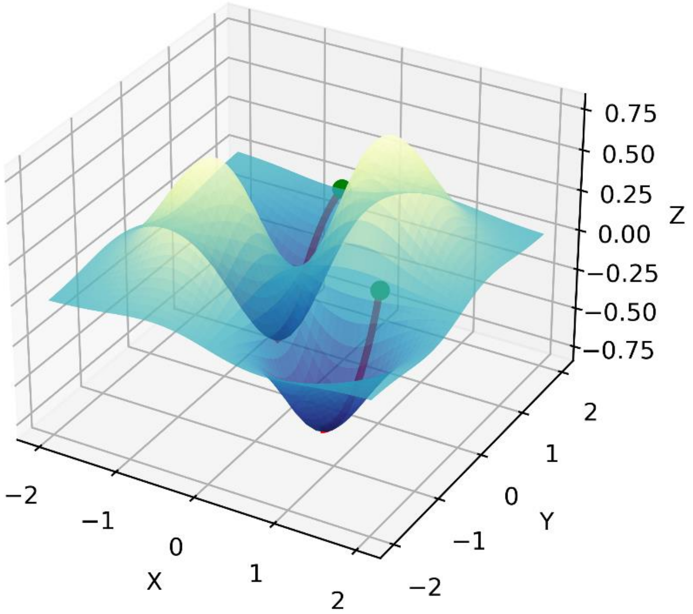

```
================================================================
Punkt końcowy: x = -12.563684677717286, y = -12.558312809259247
Punkt końcowy: x = -8.377431446211421, y = -10.109631197923933
Punkt końcowy: x = -2.094097088425078, y = -3.826445833043932
Punkt końcowy: x = 4.187974571081921, y = 2.45673997325767
Punkt końcowy: x = 10.472124475327337, y = 8.739924723614827
================================================================
```

Wszystkie Maksimum funkcji f(x):



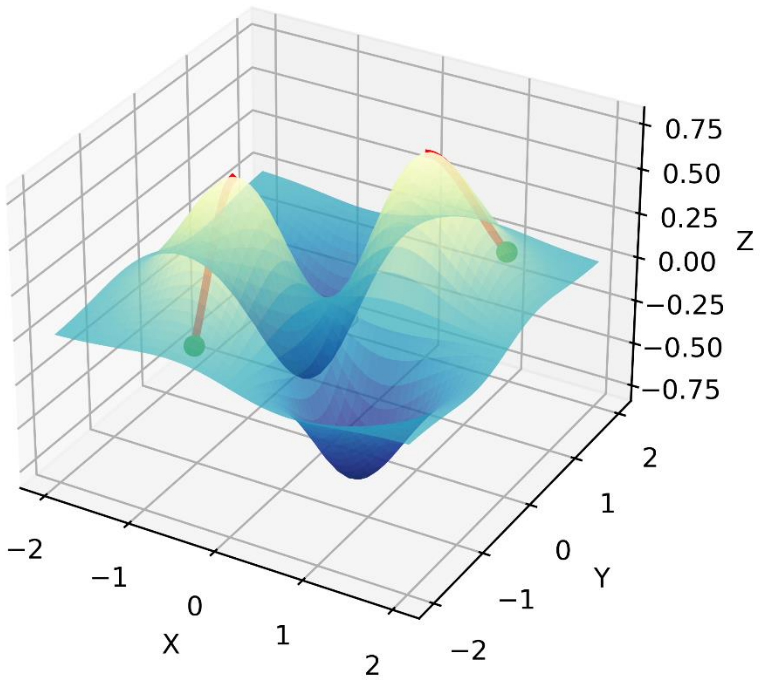

```
==========================================================================
Punkt końcowy: x = 12.56368467717286, y = 12.558312809259247
Punkt końcowy: x = 8.376983762086834, y = 10.109630908811425
Punkt końcowy: x = 2.094097088425078, y = 3.826445833043932
Punkt końcowy: x = -4.1884919581928495, y = -2.4567394742469437
Punkt końcowy: x = -10.471677265372454, y = -8.73992478142653
==========================================================================
```

Wszystkie minimum funkcji g(x, y)




```
Punkt końcowy: x = 0.7071070634691303, y = -0.7071064995568045, z = -0.7357588823426506
Punkt końcowy: x = -0.7071042857042512, y = 0.70716923101256, z = -0.7357588765949988
```

Wszystkie maksimum funkcji g(x,y)




```
Punkt końcowy: x = -0.7071060050613878, y = -0.7071152974046836, z = 0.7357588822352754
Punkt końcowy: x = 0.7071080746627904, y = 0.7071070101793592, z = 0.7357588823403457
```