WSI

Jakub Sobolewski
300371 Informatyka Sztuczna Inteligencja

17 marca 2022

Zadanie

1) Zaimplementować metodę najszybszego wzrostu/spadku (minimalizacja, spodziewam się stałego współczynnika kroku, jeśli jednak ktoś chce zrobić więcej i zastosować zmienny współczynnik to ma taką możliwość). Gradient wyliczamy numerycznie.
2) Narysować zachowanie algorytmu (kolejne kroki algorytmu jako strzałki na tle poziomic funkcji celu). Uwaga: w praktycznych zadaniach optymalizacji nie da się narysować funkcji celu ponieważ zadania mają wiele wymiarów (np. 100), oraz koszt wyznaczenia oceny jednego punktu jest duży.
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 (na wykresie narysować kroki w wybranych 2 wymiarach z 10). Ograniczenia kostkowe przestrzeni to -100, 100.

Środowisko

Python 3.9.7

Rozwiązanie

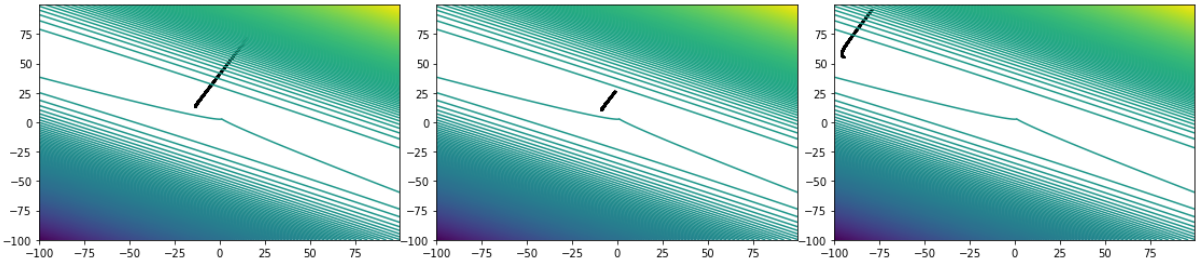
załączone w pliku script.ipynb

Sprawozdanie

Funkcja Booth

Parametry:
wymiary: 2
β: 0.000001
kroki: 10000

Wartości funkcji celu wynosiły kolejno 632, 324, 20309.

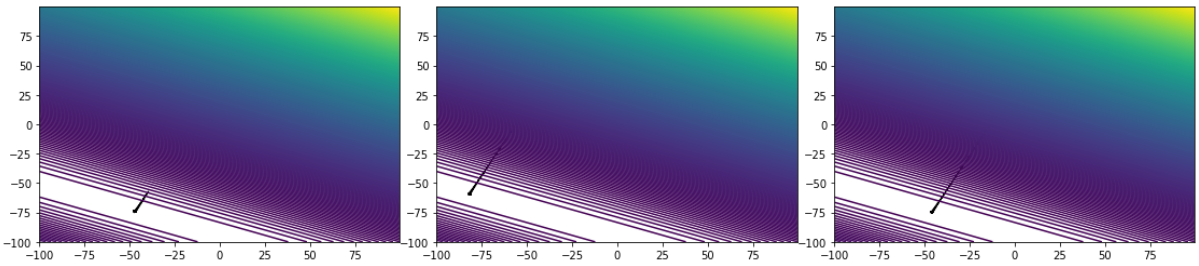


Rysunek 1: Gradienty funkcji Booth

Funkcja F1

Parametry:
wymiary: 10
β: 0.00000001
kroki: 1000

Wartości funkcji celu wynosiły kolejno 769, 1850, 351.

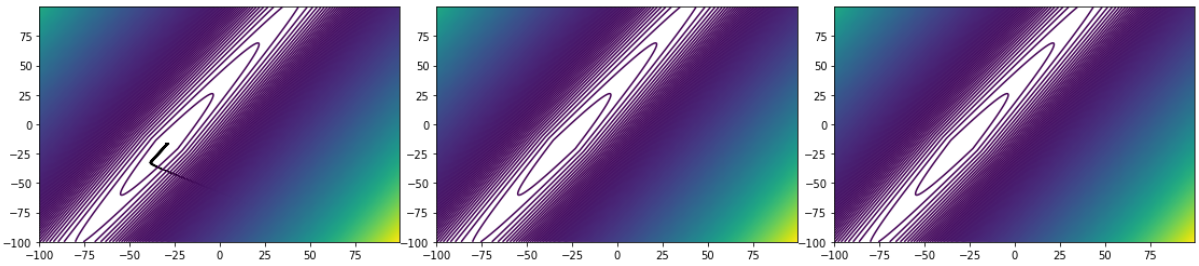


Rysunek 2: Gradienty funkcji F1

Funkcja F2

Parametry:
wymiary: 10 i 2
β: 0.00000001
kroki: 1000

W 2 wymiarach udaje się znaleźć optimum, przy większej liczbie wymiarów, ciężko znaleźć dobrą beta, raz jest za duża i problem nie jest zbieżny a raz za wolno się uczy.
Wartości funkcji celu wynosiły kolejno 200 (2 wymiary), 243428399153146 (10 wymiarów), 13074639241675 (10 wymiarów).

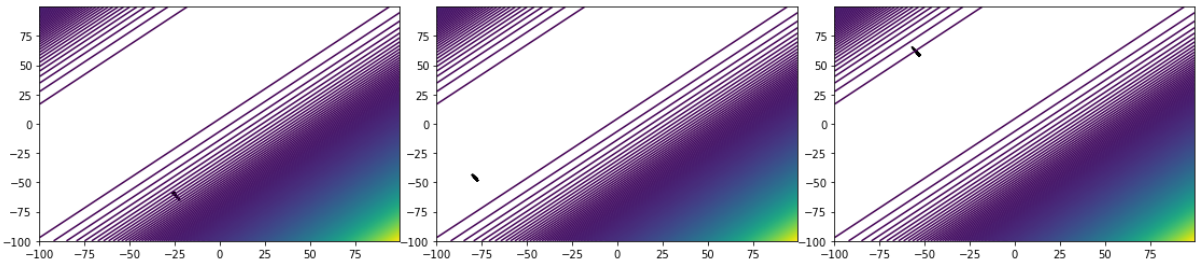


Rysunek 3: Gradienty funkcji F2

Funkcja F3

Parametry:
wymiary: 10
β: 0.00000001
kroki: 1000, 10000

Wartości funkcji celu wynosiły kolejno 64579 (1000 kroków), 43858 (10000 kroków), 25628 (10000 kroków).



Rysunek 4: Gradienty funkcji F3

Pytania

Jak wartość parametru beta wpływa na szybkość dojścia do optimum i zachowanie algorytmu?
Za mała beta powoduje dużą ilość iteracji potrzebnych do osiągnięcia optimum. za duża beta powoduje niestabilność, czyli $f(x) = \infty$ lub wartość funkcji zmienia się skokowo w górę i w dół.

Zalety/wady algorytmu?
Zalety: prosty, zwraca rozwiązanie optymalne (o ile nie utknie w minimum lokalnym).
Wady: nie działa dla nieznanej funkcji (wtedy trzeba inaczej gradient liczyć).

Wnioski
Każda funkcja zachowuje się inaczej, jednych łatwiej znaleźć optimum innych trudniej, dobranie hiperparametów ma kluczowe znaczenie w szukaniu optymalnego rozwiązania.