SPRAWOZDANIE

Zajęcia: Matematyka Konkretna

Prowadzący: prof. dr hab. Vasyl Martsenyuk

Zadanie 7
Temat: Metoda gradientu prostego. Stosowanie do algorytmu wstecznej propagacji błędu
Wariant 13

Łukasz Pindel
Informatyka II stopień,
stacjonarne,
2 semestr,
Gr. 1

1. Polecenie:

Zadaniem do zrealizowania jest odnalezienie wartości minimalnej funkcji dwóch zmiennych f() oraz zmiennych x i y metodą gradientu wraz z wizualizacją w 3D odpowiednio do określonego zadania. Można skorzystać z dowolnych bibliotek Python.

2. Wprowadzane dane:

Wariant 13 (odpowiednio wariant 1) – funkcja dwóch zmiennych

$$1.\ \ f(x,y) = (x+3y)^3 + 2 * x, \quad x \in [1;100], \quad y \in [1;100]$$

Rysunek 1: Funkcja dwóch zmiennych

3. Wykorzystane komendy:

Funkcja celu (function):

$$f(x,y)=(x+3y)^3+2x.$$

Ta funkcja definiuje funkcję celu, która ma być minimalizowana.

Pochodne funkcji celu (dx i dy):

$$df/dx=3(x+3y)^2+2 \text{ i } df/dy=9(x+3y)^2.$$

Te funkcje obliczają pochodne cząstkowe funkcji celu względem x i y.

Metoda spadku gradientu (gradient_descent):

*x = x - learning_rate * dx(x, y)*
*y = y - learning_rate * dy(x, y)*
history.append([x, y, function(x, y)])

Funkcja **gradient_descent()** implementuje metodę spadku gradientu. Przechodzi przez określoną liczbę iteracji i aktualizuje wartości x i y w kierunku przeciwnym do gradientu funkcji celu, pomnożonego przez stałą współczynnika uczenia. **X** aktualizuje wartość x w kierunku przeciwnym do pochodnej cząstkowej względem x, a **Y** wartość y w kierunku przeciwnym do pochodnej cząstkowej względem y. **history.append()** zapisuje historię zmian punktów w trakcie spadku gradientu. Historia zmian jest zapisywana w postaci listy punktów.

Generowanie siatki punktów (x_vals, y_vals, X, Y, Z):

Z = function(X, Y)
X, Y = np.meshgrid(x_vals, y_vals)

Z = function(X, Y) oblicza wartości funkcji celu dla punktów X, a **np.meshgrid(x_vals, y_vals)** tworzy siatkę punktów X i Y do wygenerowania powierzchni funkcji celu

Parametry metody spadku gradientu:

learning_rate = 0.0001
iterations = 1000

Stała współczynnika uczenia (**learning_rate**) oraz liczba iteracji (**iterations**) określają parametry metody spadku gradientu.

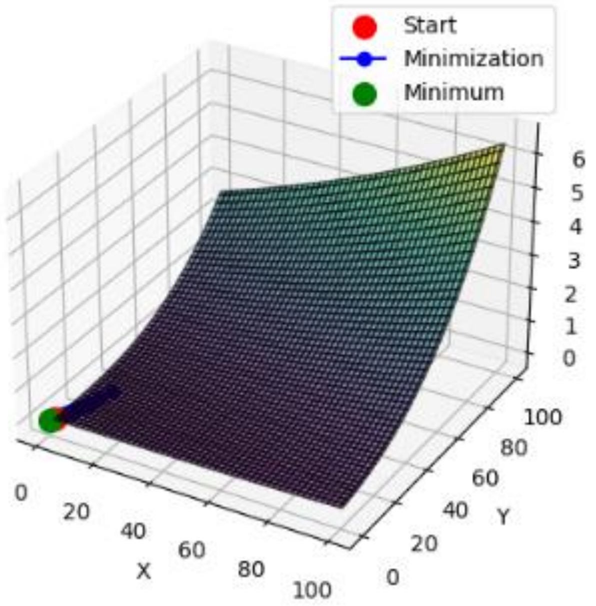
Wykres 3D:

ax.plot_surface(X, Y, Z, cmap='viridis', alpha=0.6, edgecolor='k')
ax.plot(history[:, 0], history[:, 1], history[:, 2], color='blue', marker='o', label='Minimization')

ax.plot_surface(X, Y, Z, cmap='viridis', alpha=0.6, edgecolor='k') - rysuje powierzchnię funkcji celu, a funkcja **ax.plot(history[:, 0], history[:, 1], history[:, 2], color='blue', marker='o', label='Minimization')** rysuje trajektorię spadku gradientu.

Link do repozytorium:
https://github.com/denniak/MK/tree/main/MK_7

4. Wynik działania:



Rysunek 2: Otrzymany wykres funkcji dwóch zmiennych wraz z zaznaczonym minimum

5. Wnioski:

Na podstawie otrzymanego wyniku można stwierdzić, że zastosowana metoda spadku gradientu umożliwia znalezienie minimum funkcji celu. Parametry metody, takie jak stała współczynnika uczenia i liczba iteracji, odpowiednio ustawione na wartości 0.0001 oraz 1000, mają istotny wpływ na efektywność minimalizacji. Rysunek wykresu 3D wskazuje dokładnie, gdzie znajduje się minimum funkcji dwóch zmiennych – na samym początku układu współrzędnych (zaznaczone zielonym punktem), co jest kluczowym wnioskiem dla analizy tej funkcji.