WSI Labolatorium 1

Jonatan Kasperczak 341208

March 2025

1 Wprowadzenie

Celem projektu jest zbadanie zachowania **metody gradientu prostego** (ang. *gradient descent*) dla różnych funkcji testowych w przestrzeni $\mathbb{R}^n$. W szczególności rozważono trzy funkcje:

- **Funkcja kwadratowa** (*Quadratic*):

$$f_{\text{square}}(x) \;=\; \sum_{i=1}^{n} x_i^2,$$

- **Funkcja Rosenbrocka** (*Rosenbrock*):

$$f_{\text{rosen}}(x) \;=\; \sum_{i=1}^{n-1} \Big[\, 100\left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2 \Big],$$

- **Funkcja Ackley** (*Ackley*):

$$f_{\text{ackley}}(x) \;=\; -20\,\exp\!\Big(-0.2\sqrt{\tfrac{1}{n}\sum_{i=1}^{n} x_i^2}\Big) \;-\; \exp\!\Big(\tfrac{1}{n}\sum_{i=1}^{n}\cos\big(2\pi x_i\big)\Big) \;+\; 20 \;+\; e.$$

2 Parametry oraz ich domyślne wartości

- **Wymiar przestrzeni**: $n = 10$.
- **Punkt startowy** x_0 losowany z przedziału $[-10, 10]^n$.
- **Kroki** α badane na kilku poziomach, np. 10^{-3}, 10^{-4}, 10^{-5}.
- **Maksymalna liczba iteracji**: $\text{maxIter} = 20000$.
- **Tolerancja** $\text{tol} = 1\times 10^{-6}$.

3 Pseudokod solvera metodą gradientu prostego

Algorytm **gradientu prostego** aktualizuje wektor x w każdej iteracji:

$$x_{\text{next}} \;\leftarrow\; x - \alpha\,\nabla f(x),$$

gdzie $\alpha > 0$ jest stałym krokiem (tzw. *learning rate*). Jako podstawowy **warunek stopu** zastosowano zmniejszenie normy przesunięcia między kolejnymi iteracjami poniżej wartości zadanej tolerancji (ang. *tolerance*):

$$\|\,x_{t+1} - x_t\| \;<\; \text{tol}.$$

Algorithm 1 Metoda Gradientu Prostego z warunkiem: $\|x_{t+1} - x_t\| < \text{tol}$

Require: Funkcja celu $f : \mathbb{R}^n \to \mathbb{R}$; punkt startowy $x_0 \in \mathbb{R}^n$; krok uczenia $\alpha > 0$; maksymalna liczba iteracji maxIter; tolerancja tol.

$x \leftarrow x_0$
success $\leftarrow$ False
for $i = 1$ to maxIter **do**
  $f_{\text{val}} \leftarrow f(x)$
  $g \leftarrow \nabla f(x)$ (wyliczamy gradient)
  $x_{\text{next}} \leftarrow x - \alpha\, g$ (krok w stronę spadku)
  if $\|\, x_{\text{next}} - x \,\| < \text{tol}$ **then**
    success $\leftarrow$ True
    break
  end if
  $x \leftarrow x_{\text{next}}$
end for
return (x, $f(x)$, i, success)

4 Przeprowadzone testy, ich wyniki i obserwacje

Poniżej zaprezentowano przykładowe wyniki uzyskane dla każdej z badanych funkcji. Wyniki przedstawiają:

- **f_opt**: końcową wartość funkcji
- **x_opt**: wyznaczony punkt minimalizujący (osiągnięty przez solver)
- **iterations**: całkowitą liczbę wykonanych iteracji
- **success**: czy osiągnięto warunek stopu przed maxIter

4.1 Funkcja Quadratic

```
=== Test funkcji Quadratic ===

--- alpha=0.001 ---
Solver zakonczony w 5235 iteracjach (czas: 0.8962s). f_opt=2.4818e-07, ||x||=0.000, success=True

--- alpha=0.0001 ---
Solver zakonczony w 20000 iteracjach (czas: 2.7919s). f_opt=1.0551e-01, ||x||=0.325, success=False

--- alpha=1e-05 ---
Solver zakonczony w 20000 iteracjach (czas: 2.7769s). f_opt=1.4144e+02, ||x||=11.893, success=False
```

Wnioski: Widzimy, że dla $\alpha = 10^{-3}$ solver skutecznie zbliża się do zera (*success=True*), zaś dla mniejszych kroków (10^{-4} i 10^{-5}) wymaga większej ilości iteracji lub lepiej dynamicznie dopasowanego kroku uczenia, aby osiągnąć oczekiwane minimum.

4.2 Funkcja Rosenbrocka

```
=== Test funkcji Rosenbrock ===

--- alpha=0.001 ---
Solver zakonczony w 20000 iteracjach (czas: 7.0676s). f_opt=9.6910e+06, ||x||=31.623, success=False

--- alpha=0.0001 ---
Solver zakonczony w 20000 iteracjach (czas: 7.0503s). f_opt=9.5175e+05, ||x||=12.719, success=False

--- alpha=1e-05 ---
Solver zakonczony w 20000 iteracjach (czas: 7.0439s). f_opt=8.8527e+00, ||x||=0.044, success=False
```

Wnioski: Rosenbrock jest trudniejszą funkcją do optymalizacji metodą gradientu prostego. Widzimy duże wartości f_{opt} i brak sukcesu. Przy małych α pojawiają się lepsze wyniki (niższe f_{opt}), ale wciąż nie jest osiągany warunek stopu.

4.3 Funkcja Ackley

```
=== Test funkcji Ackley ===

--- alpha=0.001 ---
Solver zakonczony w 1258 iteracjach (czas: 0.5488s). f_opt=1.3459e+01, ||x||=17.653, success=True

--- alpha=0.0001 ---
Solver zakonczony w 10428 iteracjach (czas: 4.4720s). f_opt=1.3459e+01, ||x||=17.653, success=True

--- alpha=1e-05 ---
Solver zakonczony w 20000 iteracjach (czas: 8.5694s). f_opt=1.4795e+01, ||x||=17.615, success=False
```

Wnioski: Dla funkcji Ackley solver często „utknął” w punktach dających wartość około $13 \sim 15$. Przy odpowiednim doborze α i większej liczbie iteracji można jeszcze poprawić wynik, ale metoda gradientu prostego nie zawsze jest najlepsza dla funkcji typu Ackley (z wieloma lokalnymi minimami).

5 Wykresy

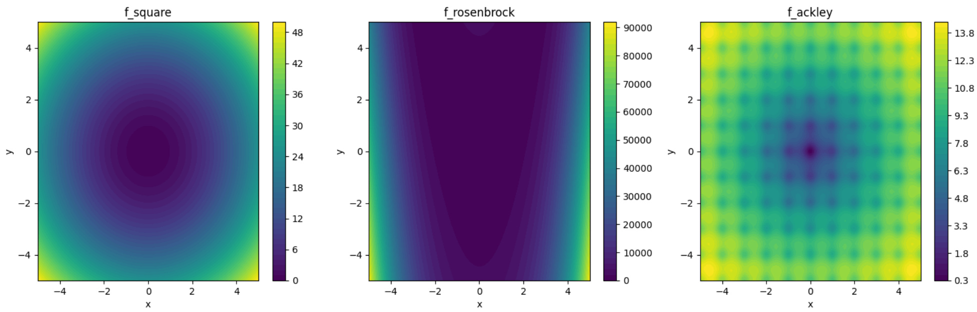

Figure 1: Wykresy funkcji w 2D

Porównanie krzywych zbieżności dla różnych wartości α.]

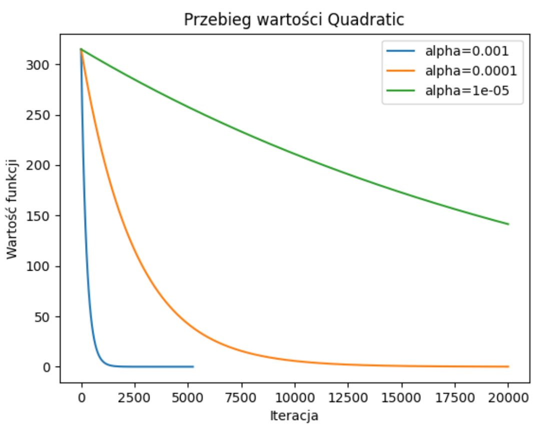

Figure 2: Przebieg wartości dla funkcji kwadratowej w kolejnych iteracjach metody gradientu prostego dla różnych kroków α

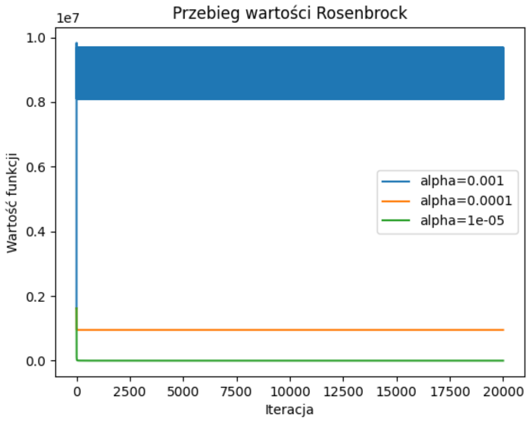

Figure 3: Przebieg wartości dla funkcji Rosenbrock w kolejnych iteracjach metody gradientu prostego dla różnych kroków α

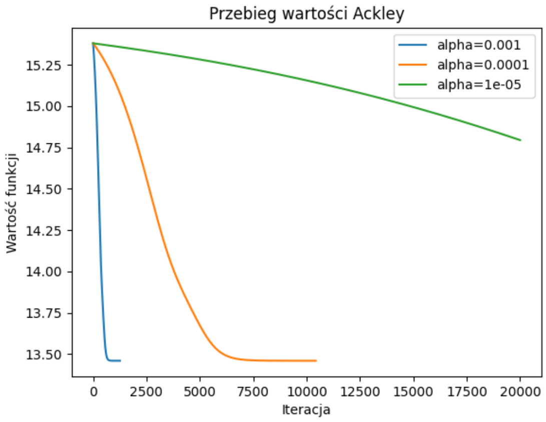

Figure 4: Przebieg wartości dla funkcji Ackley w kolejnych iteracjach metody gradientu prostego dla różnych kroków α

6 Podsumowanie

Wyniki pokazują, że:

- Dla **funkcji kwadratowej** zbyt mały krok może skutkować wolną zbieżnością, a zbyt duży – rozbieganiem się lub utknięciem.
- Dla wartości $\alpha = 0.001$ algorytm osiąga sukces średnio po 5300 iteracjach, a dla wartości α o rząd wielkości mniejszej nie udaje mu się osiągnąć wartości poniżej określonej tolerancji.
- **Rosenbrock** jest trudny dla prostego gradientu z jednym stałym α; potrzebny jest mniejszy krok, więcej iteracji lub bardziej wyrafinowana metoda (np. *line search*, *quasi-Newton*).
- Algorytm dla wartości parametru $\alpha = 0.001$ oraz 0.0001 wybiega poza ustalony zakres, natomiast już dla rzędu wielkości $10e-5$ funkcja raz osiągnęła zbieżność po około 5000 iteracji. Nie jest to powtarzalne, więc zależało bardzo od losowości parametrów początkowych.
- **Ackley** ma liczne minima lokalne, co utrudnia znalezienie globalnego minimum metodami spadku wzdłuż gradientu.
- W przypadku funkcji Ackley aby nie wpaść w minima lokalne nie można użyć zbyt małej wartości parametru α.

W praktyce częstymi modyfikacjami metody gradientu są: adaptacyjny dobór kroku, warunek stopu na normę gradientu czy zastosowanie metod wyższego rzędu.