Sprawozdanie z Ćwiczenia nr. 1

Adam Szostek nr. indeksu: 331443

23 października 2024

1 Opis implementowanego algorytmu

Implementowany algorytm to gradient prosty. Jest to jedna z metod optymalizacji mająca na celu odnajdywać minima funkcji. Działanie algorytmu:

1. Inicjalizacja punktu początkowego $\mathbf{x}_0$.
2. Iteracyjnie aktualizujemy wartość $\mathbf{x}$ zgodnie z regułą:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \beta \nabla f(\mathbf{x}_k)$$

gdzie:

- β to współczynnik kroku,
- $\nabla f(\mathbf{x}_k)$ to gradient funkcji celu w punkcie $\mathbf{x}_k$.

3. Iteracje powtarzamy do momentu, gdy norma gradientu osiągnie wartość bliską zeru:

$$\|\nabla f(\mathbf{x}_k)\| \approx 0$$

róznica między kolejnymi punktami będzie zadowalająco mała lub do osiągnięcia maksymalnej ilości iteracji (1500). Przyjęta dokładnośc to 1e-06.

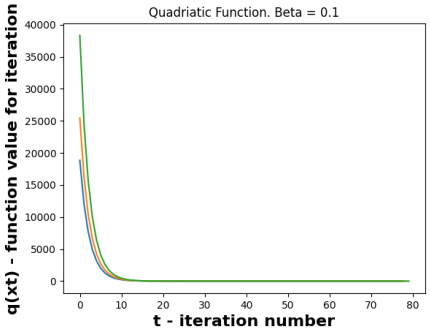
2 Opis planowanych eksperymentów numerycznych

W celu przestestowania zbieżności algorytmu użyte zostały trzy funkcje: kwadratowa oraz f3 i f12 z benchmarku CEC 2017. Dla każdej funkcji przedstawiono działanie algorytmu dla trzech wartości parametru kroku β na wykresach wartości funkcji w zależności od kroku iteracji. Punkt wejściowy generowany był losowo z dziedziny $[-100, 100]^{10}$. Wykresy zawierają po kilka krzywych reprezentujących działanie algorytmu dla róznych punktów startowych. Niektóre wykresy rozbieżne do ∞ mają mniej iteracji dla ich czytelności.

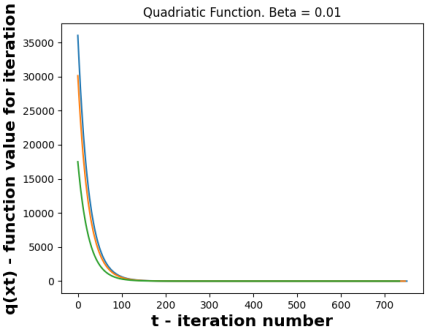
3 Opis uzyskanych wyników

3.1 Funkcja kwadratowa

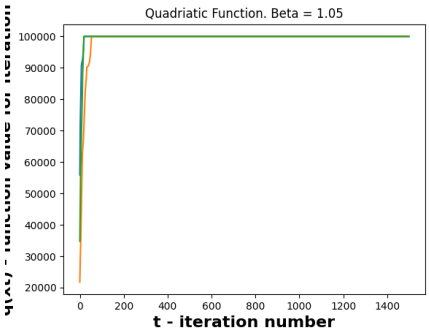
Dla $\beta = 0{,}1$ algorytm bardzo szybko znajduje minimum funkcji. Dla $\beta = 0{,}01$ obserwujemy 10-cio krotny wzrost czasu działania algorytmu, jednakże minimum również zostaje odnalezione. Dla $\beta = 1{,}05$ algorytm jest robieżny. Na wykresach widać 3 próby dla każdego parametru β.



Rysunek 1: Wykres zbieżności dla f. kwadratowej, β=0,1



Rysunek 2: Wykres zbieżności dla f. kwadratowej, β=0,01



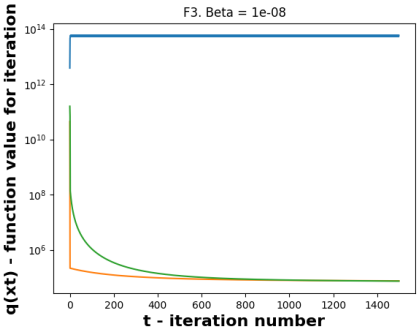
Rysunek 3: Wykres zbieżności dla f. kwadratowej, β=1,05

	$\beta = 0{,}1$	$\beta = 0{,}01$	$\beta = 1{,}05$
Średni czas	0,007 s	0,067 s	0,14 s

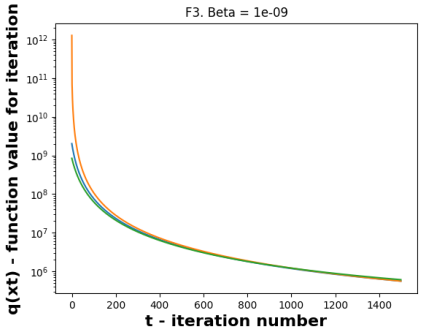
Tabela 1: Czas działania algorytmu dla funkcji kwadratowej.

3.2 Funkcja f3

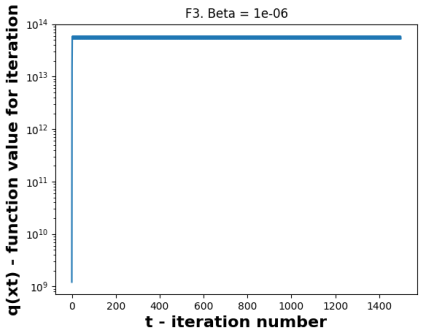
Dla f3 algorytm doszedł do maksymalnej ilości iteracji, tym samym nie odnajdując satysfakcjonującego przyjętą dokładność minimum. Na początku, wartości f. celu szybko maleją, po czym algorytm osiąga punkt, w którym dalsza redukcja wartości funkcji jest minimalna. Warto zwrócić uwagę, że dla tych samych β w zależności od punktu startowego algorytm może być robieżny lub zbieżny. Świadczy to o skomplikowanej budowie funkcji CEC. Na wykresach użyto skali logarytmicznej dla osi Y i na każdym wykresie widać 3 próby dla parametru β (poza ostatnim).



Rysunek 4: Wykres zbieżności dla f3, β=1e-08



Rysunek 5: Wykres zbieżności dla f3, β=1e-09



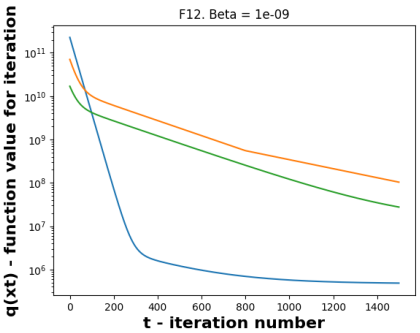
Rysunek 6: Wykres zbieżności dla f3, , β=1e-06, 1 próba dla czytelności.

	β = 1e-09	β = 1e-08	β = 1e-06
Średni czas	0,58 s	0,57 s	0,58 s

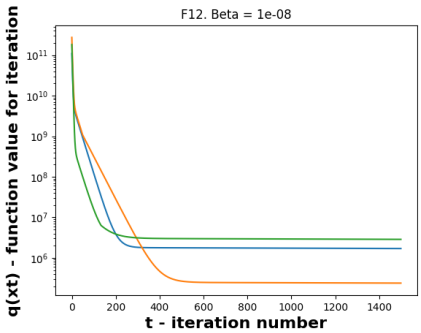
Tabela 2: Czas działania algorytmu dla funkcji f3.

3.3 Funkcja f12

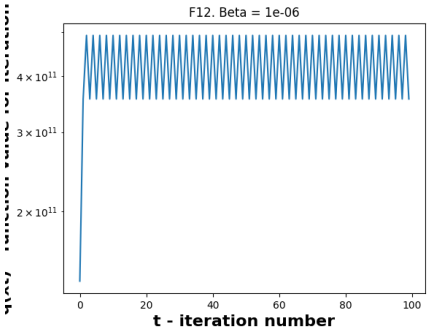
Zaimplementowany algorytm nie znalazł satysfakcjonującego przyjętą dokładność minimum. Dla niektórych β możemy obserwować, że wartości funkcji zbiegają do pewnego momentu, po czym ich zmiana w dalszych krokach algorytmu jest znikoma. Różna wartość funkcji, przy której algorytm zaczyna spowalniać, może świadczyć o istnieniu wielu minimum lokalnych. Dla β = 1e-06 obserwujemy oscylację wartości funkcji celu, która ma miejsce gdy parametr β jest za duży. Wtedy, algorytm "skacze" wokół minimum, zamiast stabilnie się do niego zbliżać. Na wykresach użyto skali logarytmicznej dla osi Y.



Rysunek 7: Wykres zbieżności dla f12, β=1e-09, 3 próby.



Rysunek 8: Wykres zbieżności dla f12, β=1e-08, 3 próby.



Rysunek 9: Wykres zbieżności dla f12, , β=1e-06, 1 próba, Mniejsza ilość iteracji dla czytelności.

	β = 1e-09	β = 1e-08	β = 1e-06
Średni czas	1,51 s	1,50 s	1,53 s

Tabela 3: Czas działania algorytmu dla funkcji f12.

4 Wnioski z przeprowadzonych badań

Algorytm gradientu prostego działa bardzo dobrze dla prostych funkcji takich jak np. kwadratowa. W przypadku funkcji z benchmarku CEC2017, które są dużo bardziej skomplikowane, algorytm nie jest optymalny. Dzieje się tak m.in. ponieważ gradient funkcji może łatwo utknąć na tzw. plateau, czyli miejscach gdzie funkcja jest płaska, ale nie są one minimami. Dobrze widać to na wykresach zbieżności dla f3 i f12, gdzie algorytm, w pewnym momencie, znacznie spowalnia i ostatecznie nie odnajduje satysfakcjonującego przyjętą dokładność minimum. Dodatkowo, gradient w "stromych" miejscach funkcji potrafi mieć ogromne wartości co może doprowadzić do wybrania złego kierunku poszukiwań i w rezultacie do rozbieżności algorytmu. Można również stwierdzić, że algorytm jest bardzo wrażliwy na parametr kroku β, ponieważ nawet najmniejsze jego zmiany powodują rozbieżność algorytmu. Co istotne ważny jest również punkt startowy, gdyż dla tej samej wielkości β algorytm potrafi być rozbieżny lub zbieżny zależnie od tego gdzie zacznie działanie. Na przykładzie funkcji kwadratowej obserwujemy, że przy mniejszym parametrze kroku β widać wolniejsze zmierzanie do minimum, podczas gdy dla większych jest ono szybsze.

Podsumowując, algorytm gradientu prostego jest dobry, gdy chcemy zoptymalimować nim prostą, mało skomplikowaną funkcję, jednakże wymaga dokładnego wyboru paramteru kroku β i punktu startowego.