Sprawozdanie: Lab 1 [WSI] - Maciej Groszyk 289761

1. Wstęp:

Przedmiotem laboratorium było opracowanie algorytmu gradientu prostego i algorytmu newtona, który powinien działać w dwóch trybach:

- Ze stałym parametrem kroku
- Z adaptacja parametru kroku z nawrotami

Po implementacji rozwiązań należało zbadać zbieżność obu algorytmów używając następującej funkcji:

$$f(x) = \Sigma\alpha^{\frac{i-1}{n-1}} x_i^2 x = [-100, 100]^n$$

Należało również zbadać również wpływ wartości parametru kroku na zbieżność obu metod. A także porównać czas działania obu algorytmów W swoich badaniach należało rozważyć następujące wartości parametru:

- $\alpha = \{1, 10, 100\}$
- $n = \{10, 20\}$

2. Algorytm gradientu prostego:

Metoda gradientu prostego jest iteracyjnym algorytmem wyszukiwania minimum zadanej funkcji celu f. [1]

Schemat algorytmu:

1. Wybierz wektor punktów startowych x_0
2. $x_{k+1} = x_{k+1} - step \cdot \nabla f(x_k)$
3. Sprawdź kryterium stopu, jeśli jest spełniony to STOP
4. Jeżeli $f(x_{k+1}) \geq f(x_k)$ to zmniejsz wartość step i powtórz punkt 2 dla kroku k-tego (opcjonalne)
5. Powtórz punkt 2 dla następnego kroku $(k + 1)$

W celu określenia czy punkt w danym kroku dostatecznie dobrze przybliża minimum funkcji celu w metodzie gradientu prostego zostało zastosowane kryterium stopu:

- W momencie gdy norma euklidesowa (norma L2) jest poniżej ustalonej wartości tolerancji
- Przekroczenie maksymalnej liczby iteracji

3. Algorytm Newtona

Metoda gradientu prostego jest iteracyjnym algorytmem wyszukiwania minimum zadanej funkcji celu f. [1]

Schemat algorytmu:

1. Wybierz wektor punktów startowych
2. $d_k = (\nabla^2 f(x_k))^{-1} \cdot \nabla f(x_k)$
3. $x_{k+1} = x_k - d_k$
4. Sprawdź kryterium stopu, jeśli nie jest spełniony wykonaj ponownie krok 2.

Jako kryterium stopu w algorytmie Newtona można użyć tego samego warunku co w algorytmie gradientu prostego

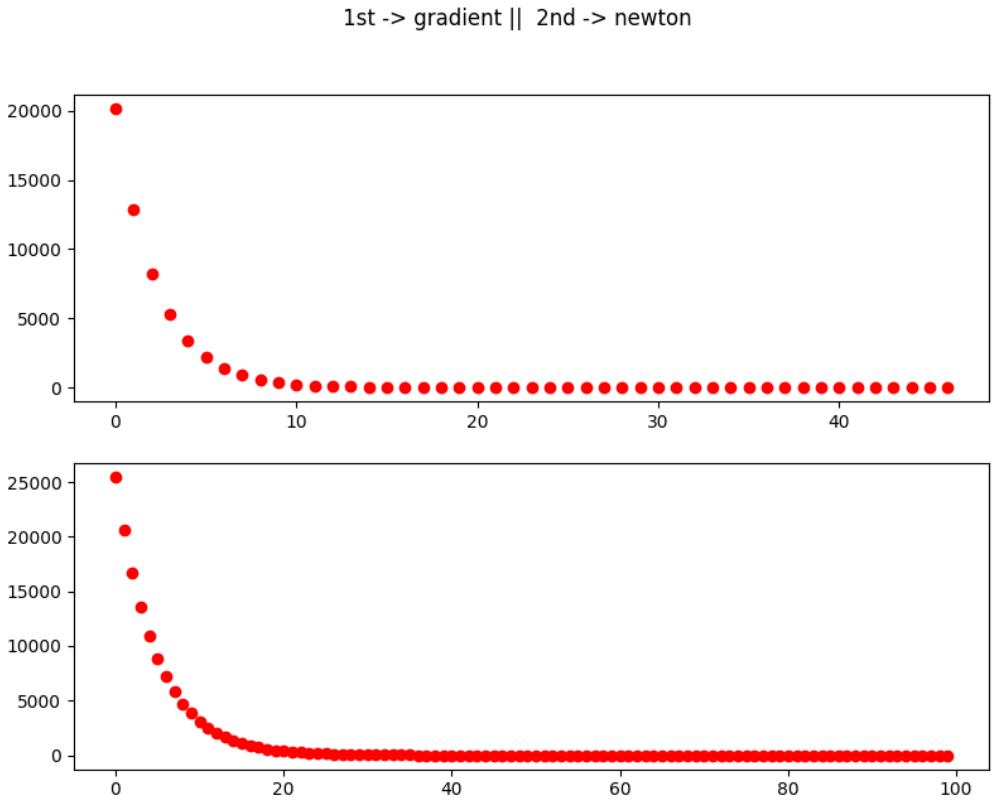
Dodatkowo można dodać do algorytmu w celu jego optymalizacji adaptacje parametru kroku.

4. Omówienie wyników

Wartość tolerancji - 0.01
Maksymalna liczba iteracji = 1000

4.1 Wyniki dla 10 wymiarowego wektora wejściowego z parametrem alfa równym 1.

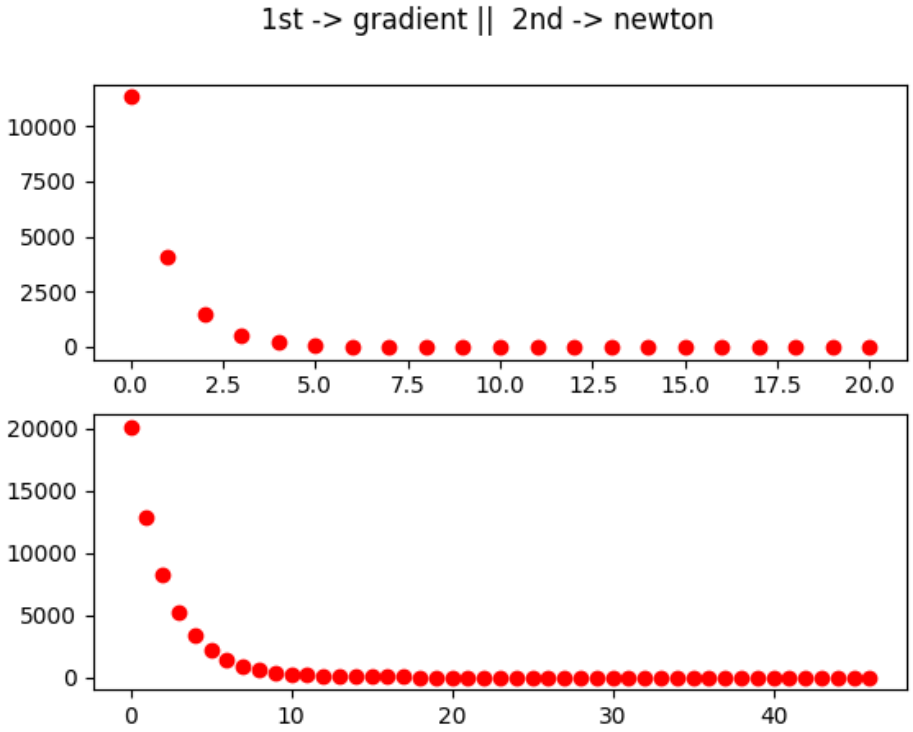
Krok = 0.1
Wektor wejściowy =[76, -5, 96, 18, -43, -63, 54, 71, 48, 7]



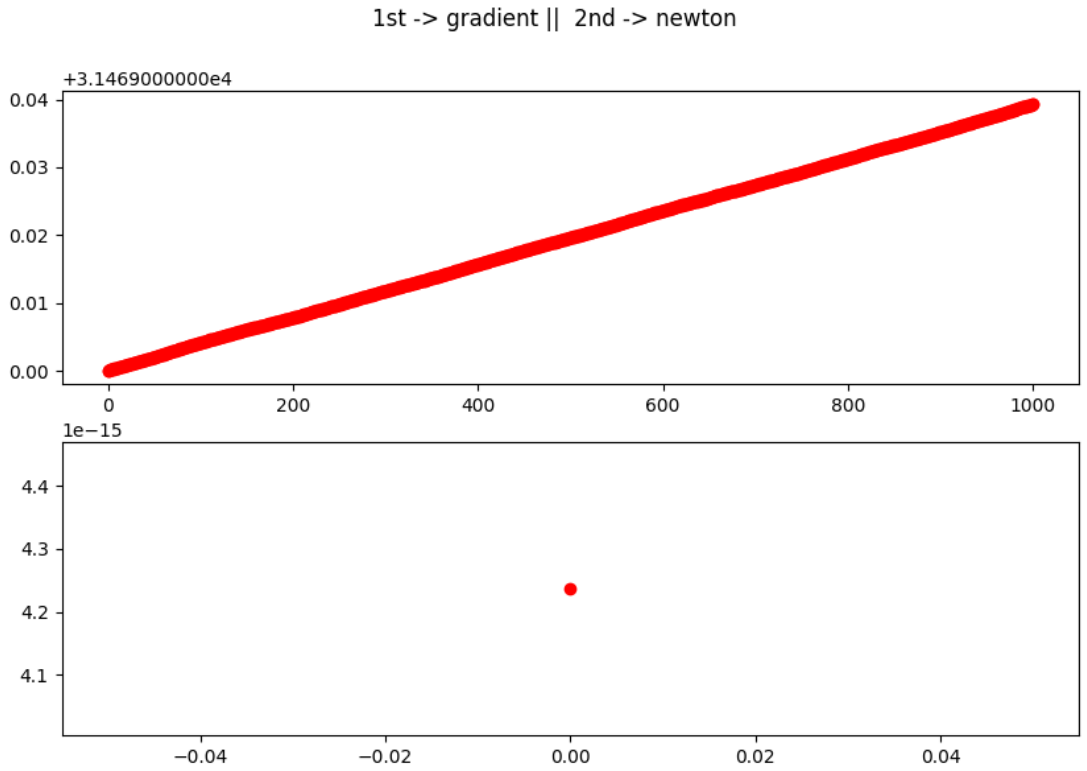
Dla takiego zestawu parametrów algorytm gradientu prostego potrzebował 47 iteracji aby spełnić kryterium stopu, natomiast algorytm newtona wykonał 100 iteracji.

W przypadku stopniowego zwiększania parametru kroku liczba iteracji zaczyna maleć.

Dla kroku = 0.2:



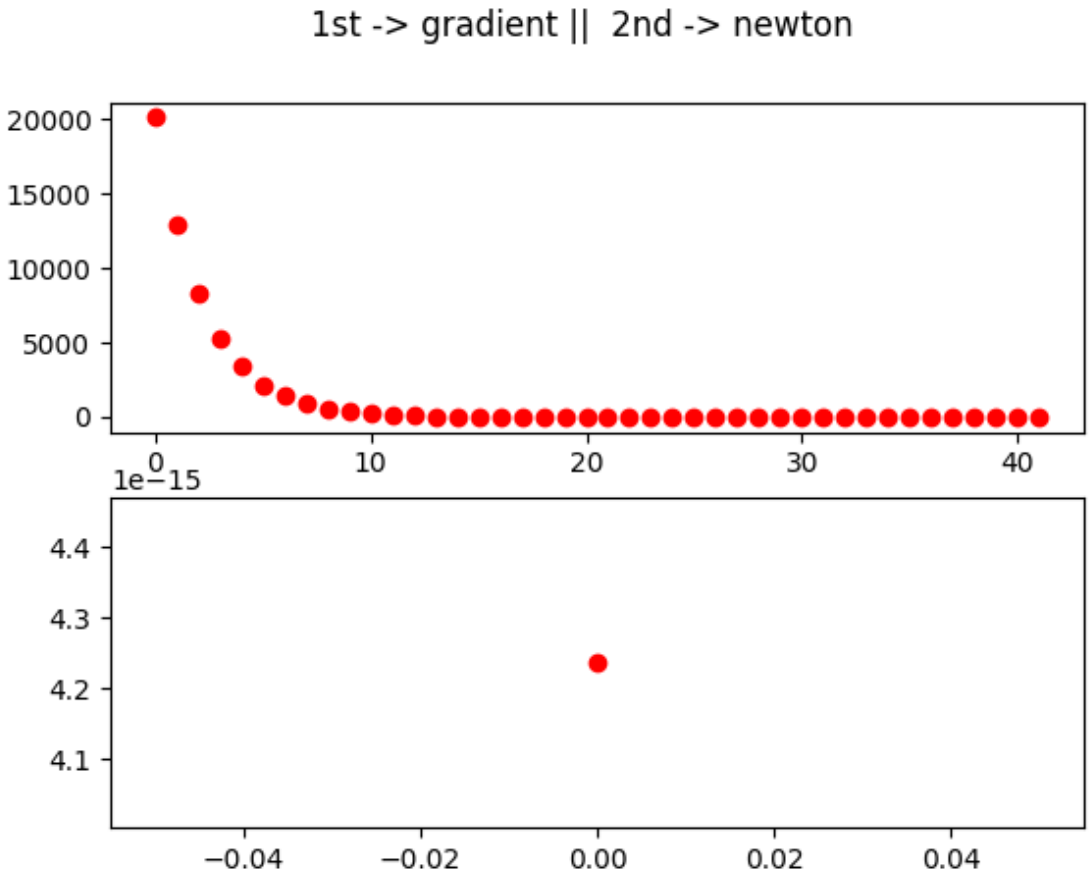
Dla kroku = 1:



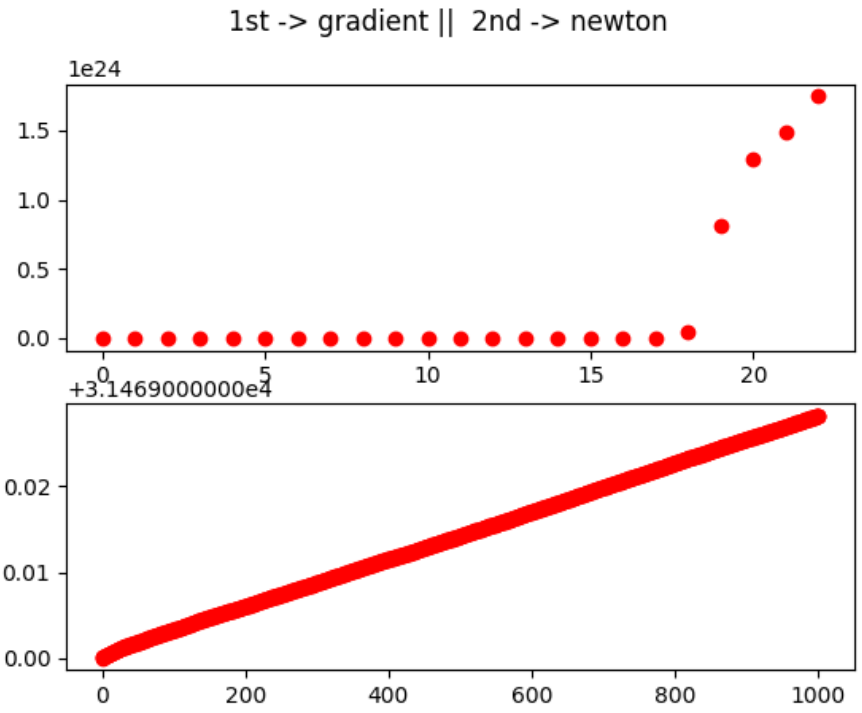
Możemy zaobserwować że metoda gradientu nie potrafi znaleźć minimum, natomiast algorytm newtona znajduje je w jednym kroku. Dzięki możliwości dopasowywania kroku w metodach możemy zoptymalizować ilość kroków potrzebnych do znalezienia minimum. Warunek dla zmiany kroku w kolejnej iteracji:

$x_{k+1} - x_k < step\ conditional$

step conditiona = 0.0001 w przypadku gdy będzie za duży istnieje możliwość zmniejszania kroku do bardzo małych wartości. Gdy zostanie on spełniony możemy dopasować zmianę kroku w zależności czy krok zadany jest za mały bądź za duży. W tym przypadku aby funkcja gradientu była wstanie znaleźć miejsce zerowe należy dopasować krok np zmniejszając w momencie spełnienia warunku o 10%

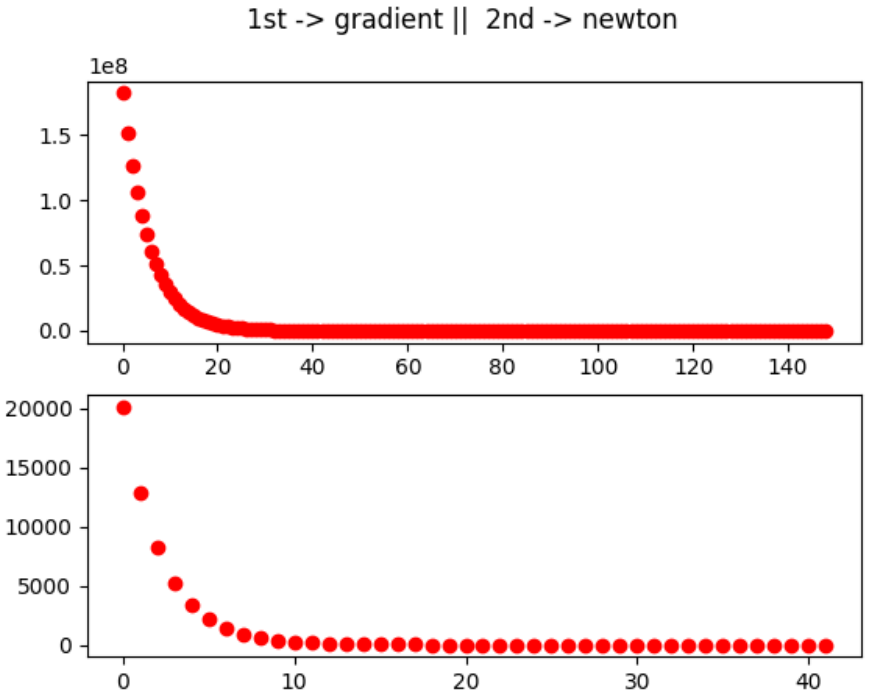


Dla kroku = 2 żadna z metod nie jest w stanie znaleźć minimum, gdy nie ustawimy dynamicznej zmiany parametru kroku.



Ze zmianą kroku o 10%:

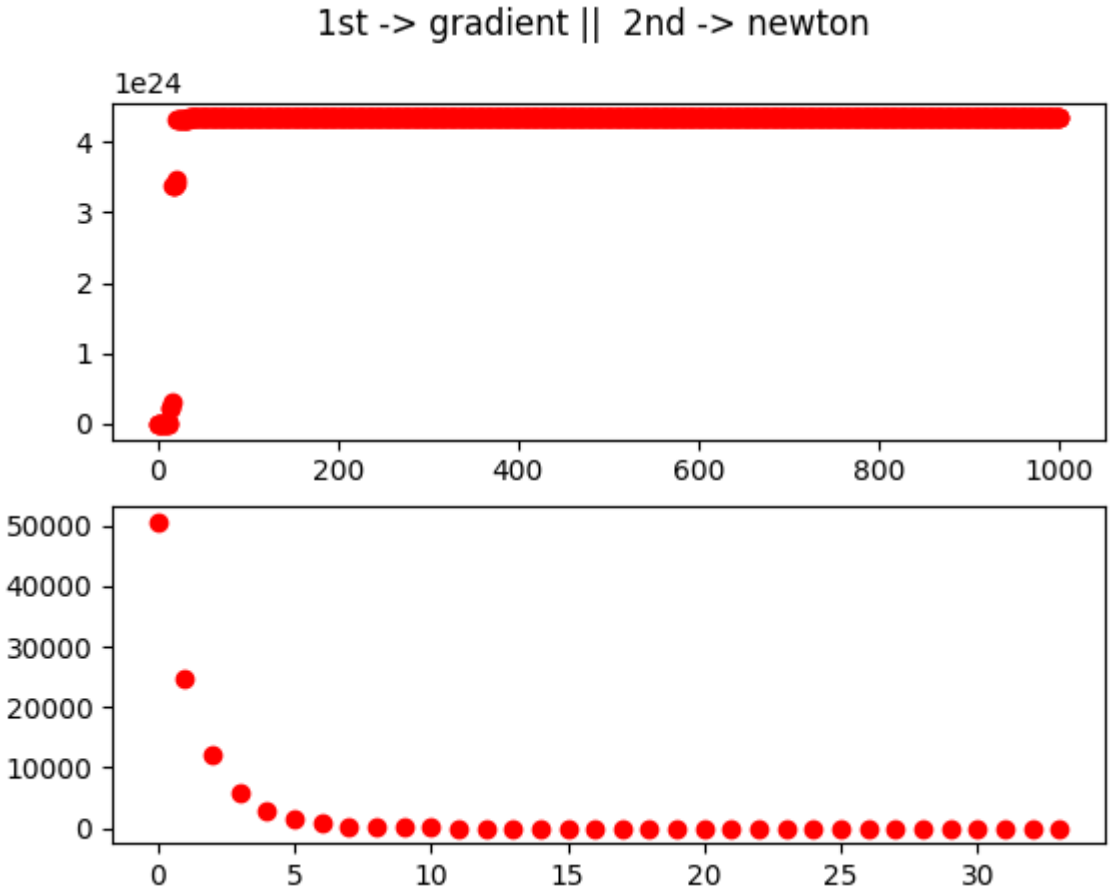
W tym przypadku również dynamiczna zmiana parametru kroku pomaga znaleźć minimum:



Wnioski: Parametr kroku jest bardzo istotny dla poprawnego działania algorytmu. Bez implementacji adaptacji kroku trzeba uważnie dobierać jego wartość. Przy zbyt dużym kroku możemy zaobserwować rozbieżność funkcji, natomiast przy zbyt małym algorytm nie znajdzie minimum w rozsądnym czasie trwania algorytmu.

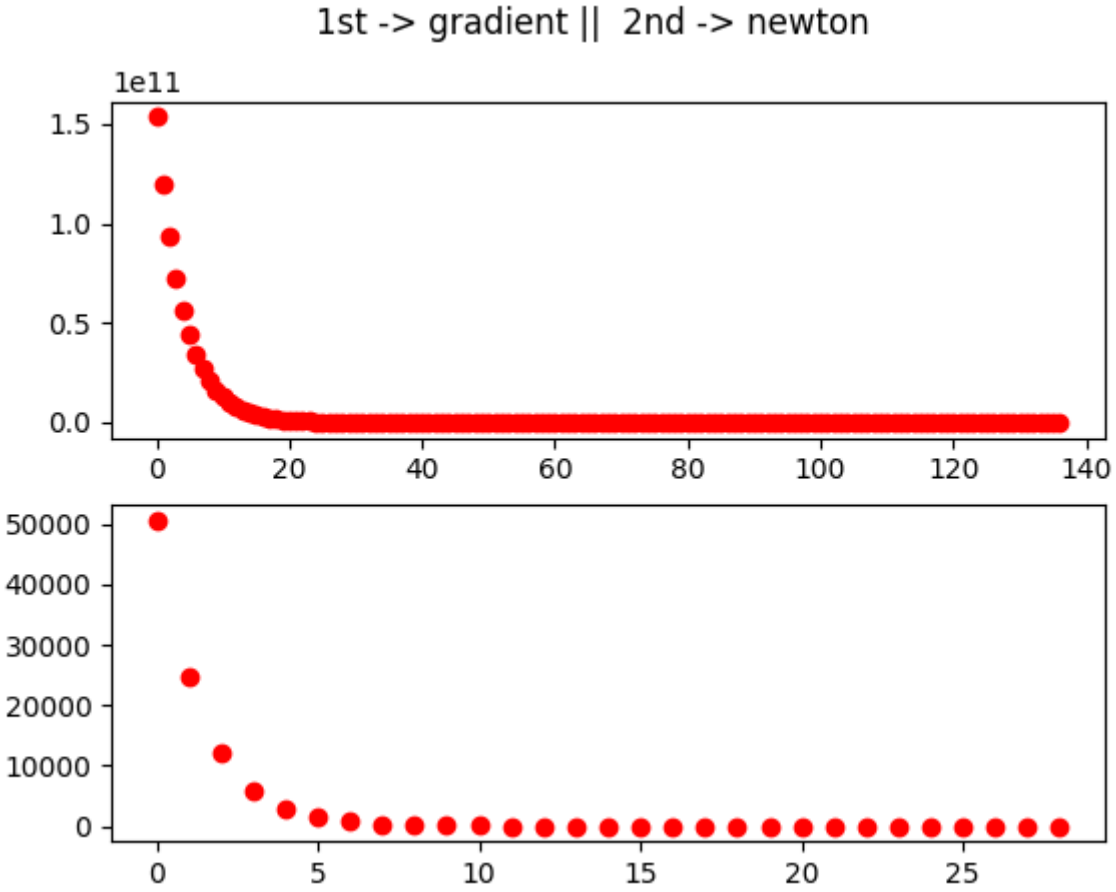
4.2 Wyniki dla 10 wymiarowego wektora wejściowego z parametrem alfa równym 10, 100.

W momencie zwiększenia parametru alfa = 10, przy kroku 0.1 metoda gradientu prostego nie może znaleźć minimum funkcji, natomiast metoda Newtona znajduje w ponad 110 iteracjach. W momencie zwiększenia kroku do 0.3 metoda newtona odpowiednio szybciej znajduje minimum funkcji.



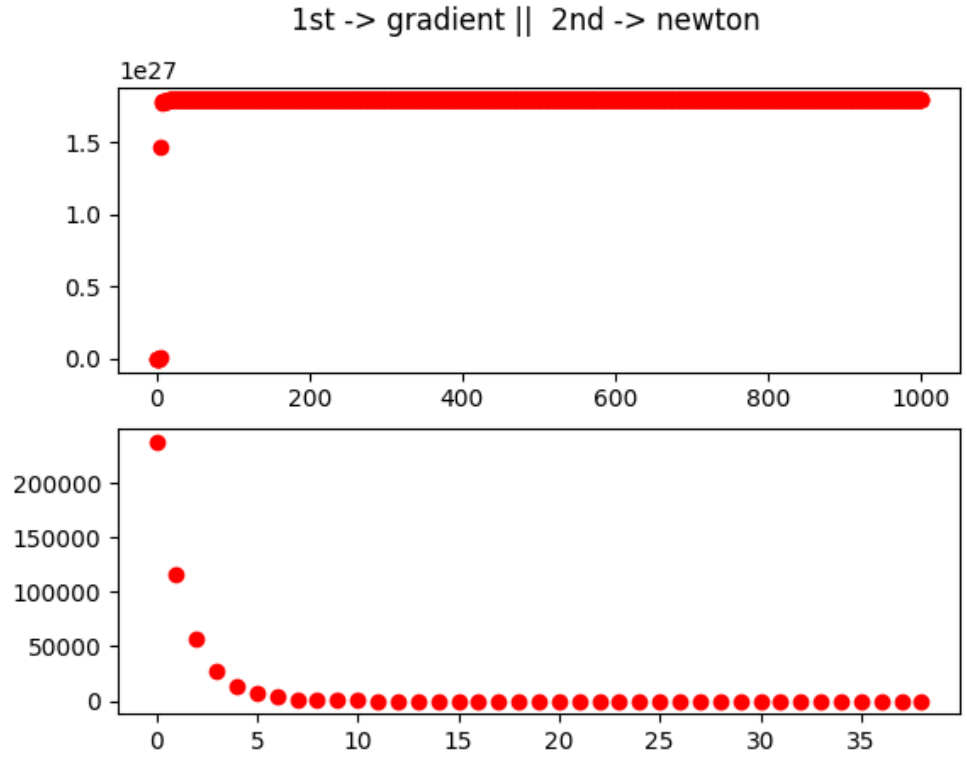
Natomiast funkcja gradientu wychodzi poza zakres poszukiwań.

W momencie dynamicznego zmniejszania parametru kroku o 10%. Metoda gradientu znajduje minimum w 137 iteracjach. A algorytm Newtona potrzebuje 29 iteracji do znalezienia minimum.

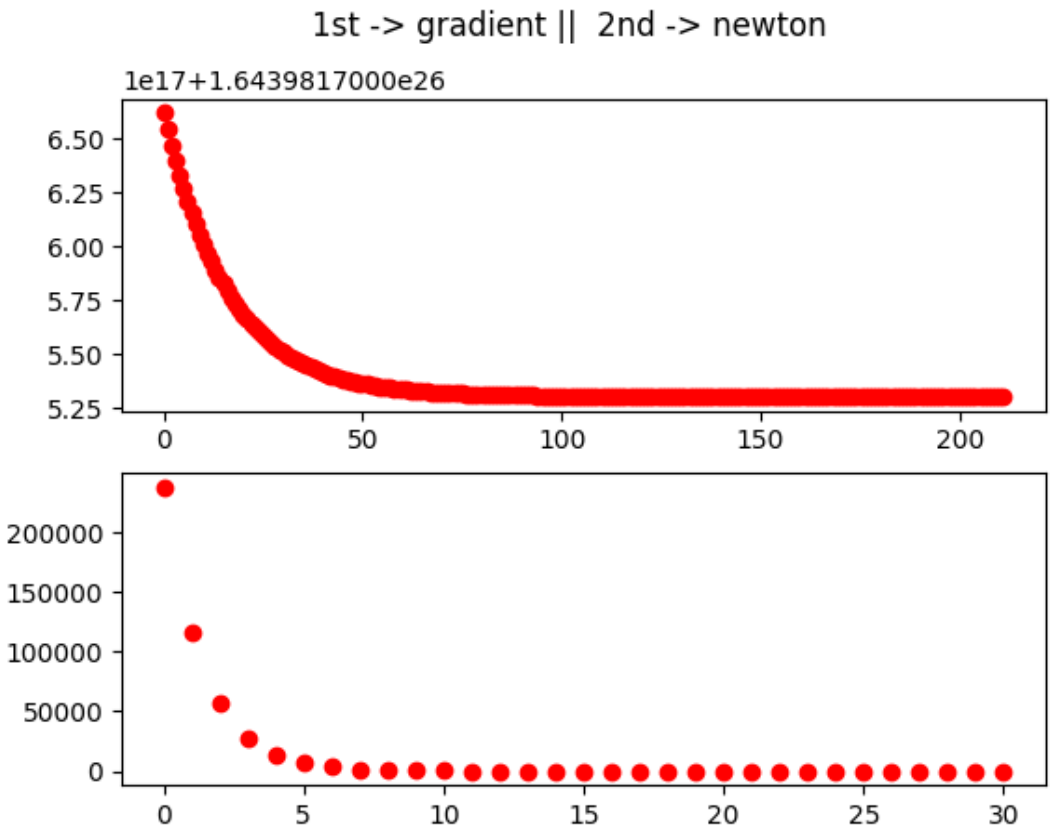


W przypadku alpha ustawionego na 100 i kroku 0.1 metoda gradientu podobnie jak przy alpha = 10 nie może znaleźć minimum funkcji. Metoda newtona znajduję minimum trochę wolniej niż dla poprzedniej wartości alpha, w 130 iteracjach.

Dla kroku 0.3 możemy zaobserwować podobne zachowanie algorytmów jak w poprzedniej konfiguracji:



W przypadku zastosowania dynamicznego zmniejszania kroku o 10%:

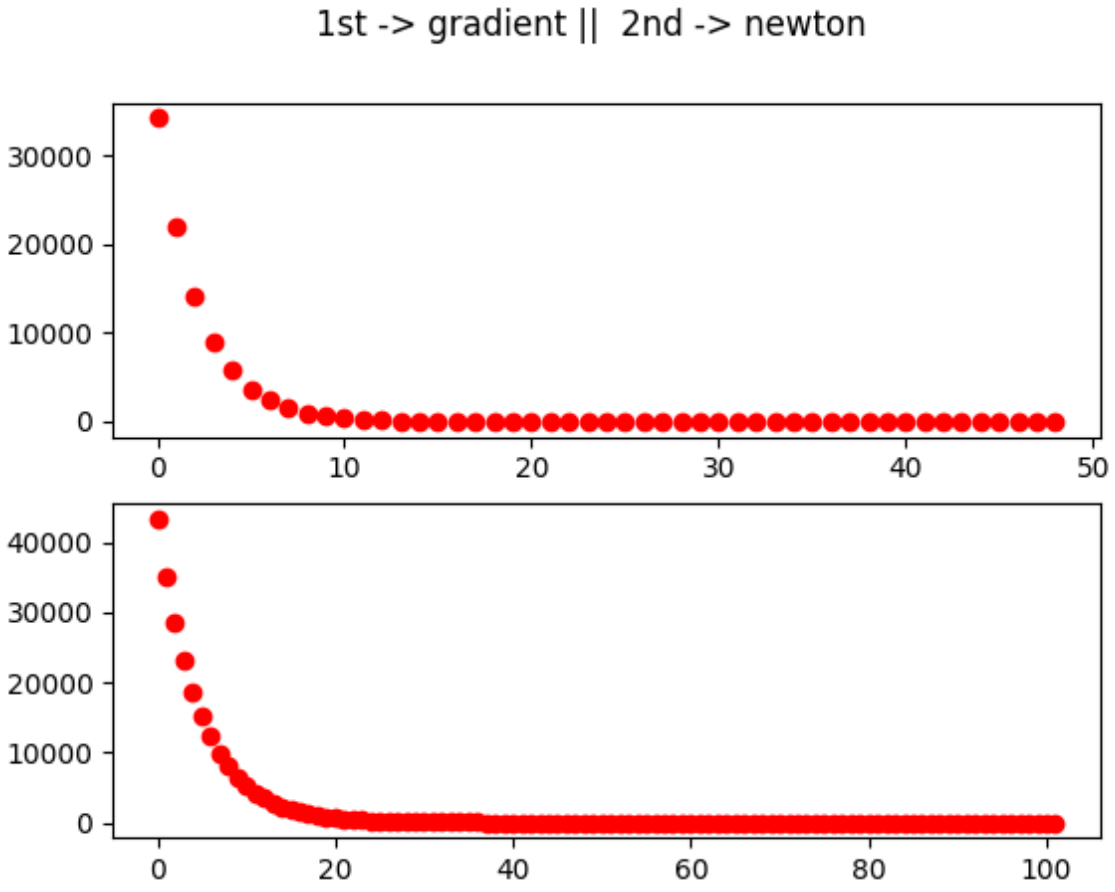


Wnioski: Przy zwiększeniu parametru alpha metoda gradientu radzi sobie dużo gorzej. Metoda Newtona jest w stanie znaleźć minimum funkcji nawet bez użycia funkcji dopasowania kroku. Należy uważać w przypadku jej użycia, gdyż z powodu zbyt małego step condition metoda może zmniejszyć krok do bardzo małych wartości które nie znajdą odpowiednio szybko poszukiwanego minimum.

4.3 Wyniki dla 20 wymiarowego wektora.

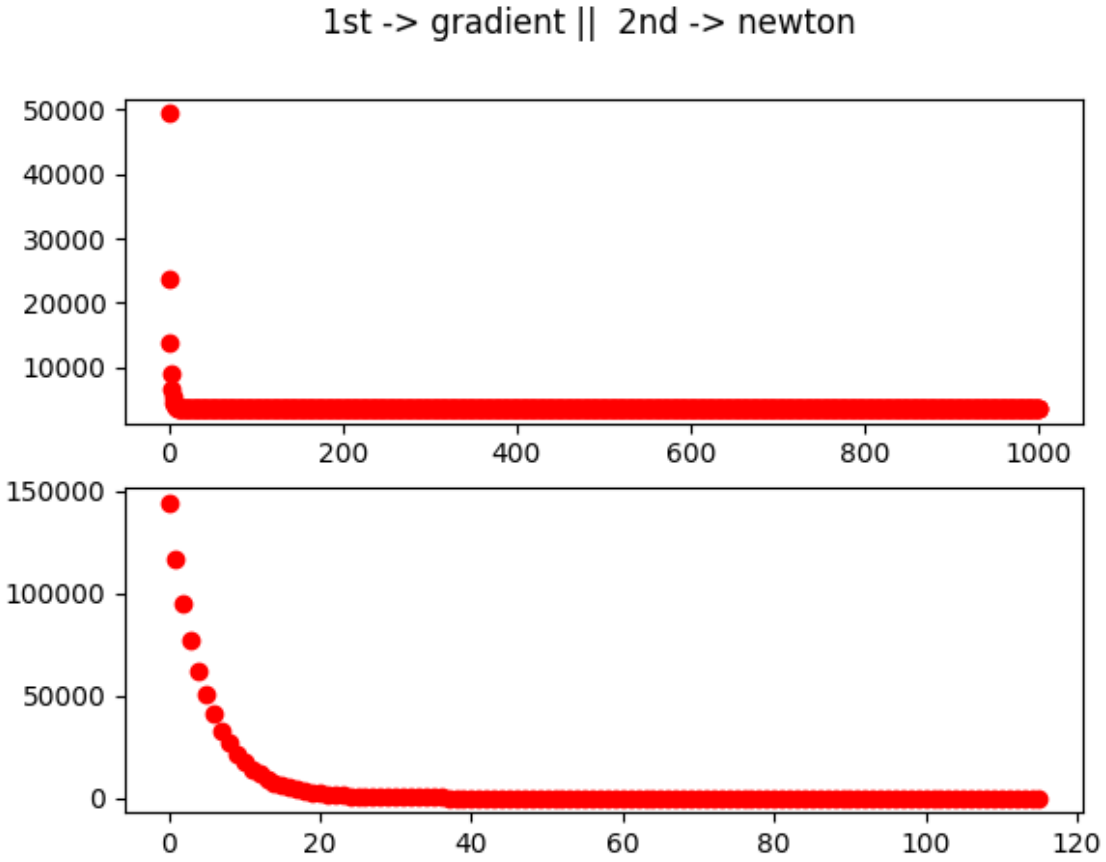
wektor = [-3, 84, -11, -57, 84, -15, 56, 69, 70, -57, -28, -80, 10, 20, 73, -28, -28, -44, -54, -19]

W przypadku większej wymiarowości wektora, alphie = 1, krok = 0.1 możemy zauważyć że algorytmy radzą sobie bez problemu. Wyniki są podobne do tych otrzymanych dla 10 wymiarowego wektora.

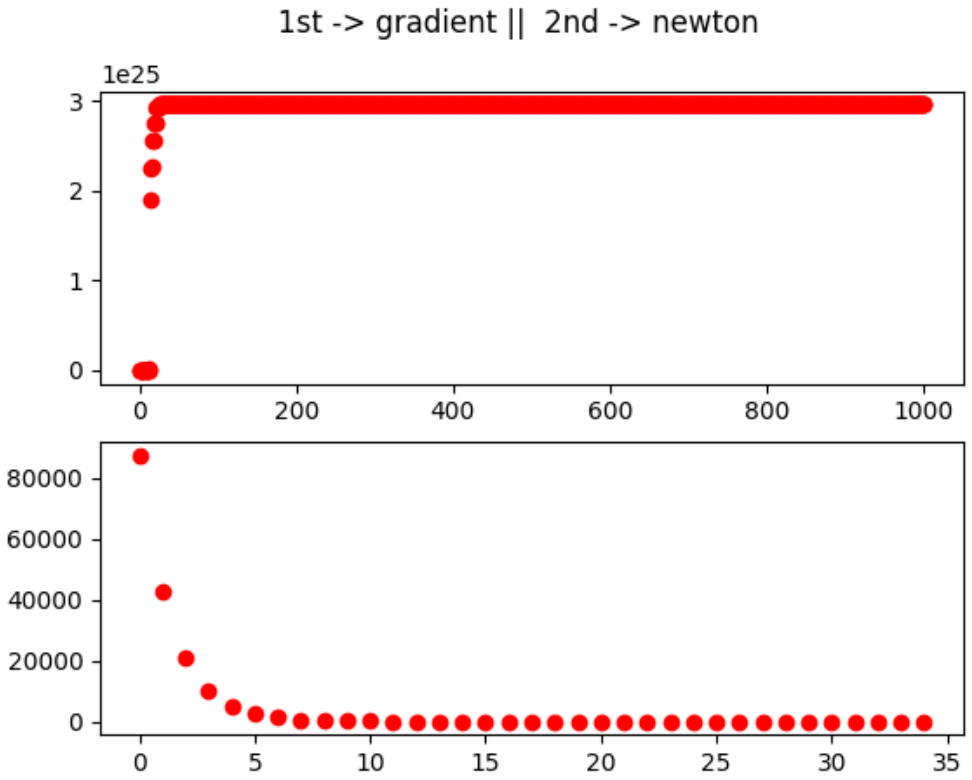


Tutaj także zwiększenie kroku pozwoli na uzyskanie mniejszej ilości iteracji

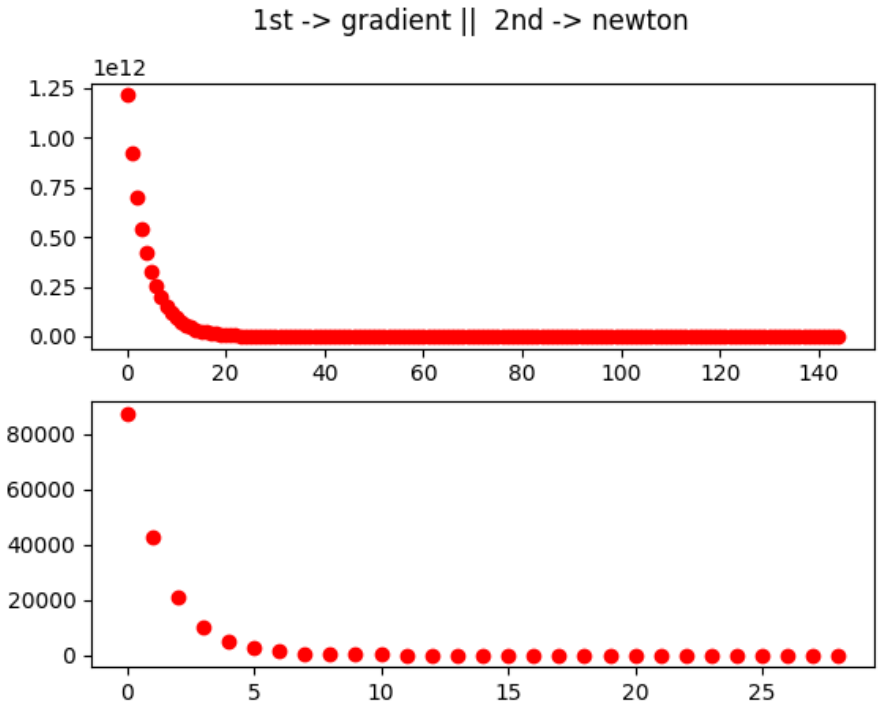
Alpha = 10, krok = 0.1
Możemy zaobserwować to samo co dla wektora 10 wymiarowego. Metoda gradientu prostego nie jest w stanie znaleźć minimum, natomiast algorytm newtona potrzebował na to 116 iteracji:



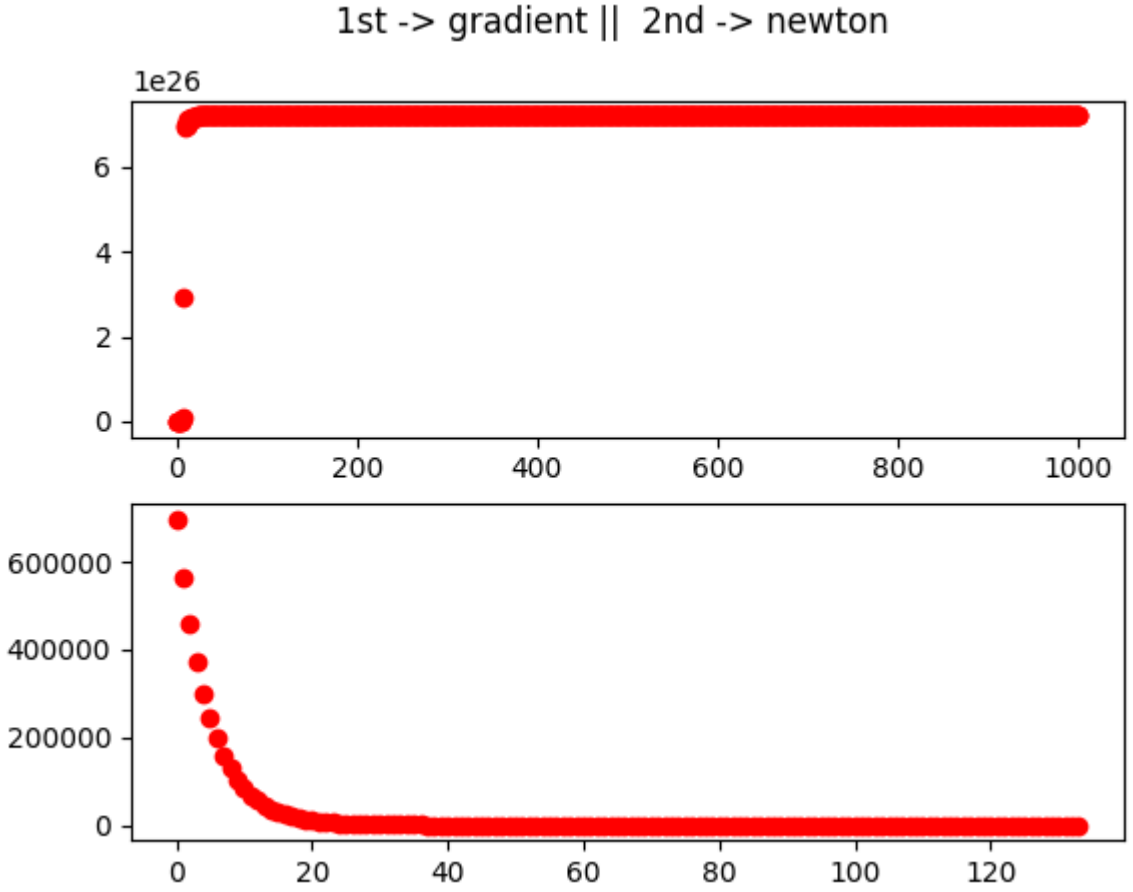
krok 0.3:



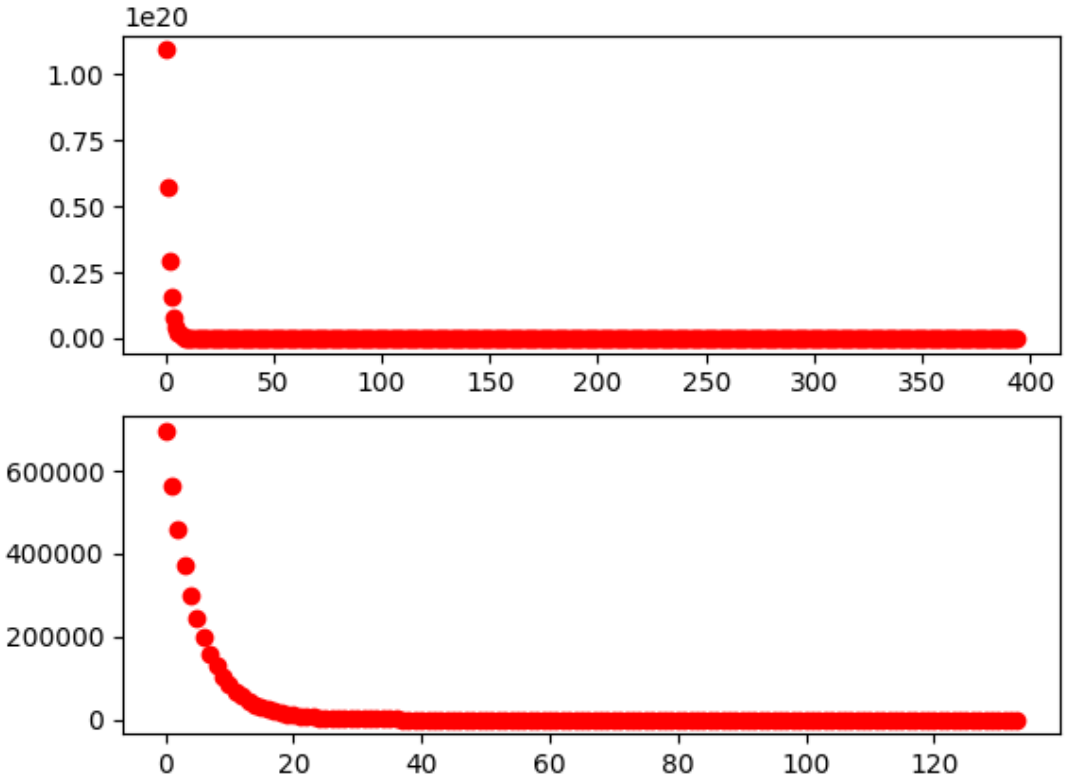
Wykresy są bardzo podobne do tych które powstały przy tej samej konfiguracji dla 10 wymiarowego wektora. W momencie zastosowania dynamicznego zmniejszania kroku o 10 % przy tej konfiguracji metoda gradientowa jest w stanie znaleźć minimum:



Przy alphie = 100 i kroku 0.1



Możemy zaobserwować że metoda gradientu jest rozbieżna, natomiast metoda newtona znalazła minimum w 134 iteracje. Przy zmniejszaniu kroku o 20% metoda gradientu jest w stanie znaleźć minimum w 395 iteracjach. Należy też pamiętać o zmniejszeniu step conditional do 1e-7 wartości.



5. Wnioski

Metoda gradientu prostego w przypadku dużych alpha jest rozbieżna, bądź nie może znaleźć minimum. W przypadku zastosowania funkcji dopasowania kroku z odpowiednio dobranym step conditionalem metoda gradientu znalazła minimum za każdym razem. Natomiast metoda newtona zazwyczaj nie potrzebowała dopasowania kroku w czasie działania. Jest to konieczne w przypadku stosunkowo dużych wartości kroku. Przy większych alpha algorytm newtona radził sobie znacznie lepiej oraz potrzebował zazwyczaj mniej iteracji do znalezienia minimum. W przypadku zadania zbyt dużego kroku obie metody mogą być rozbieżne.