Laboratorium 6

Podstawy Metod Komputerowych w Obliczeniach Inżynierskich
Jacobian, Hessian, Podstawy Optymalizacji

Marek Wodziński

AGH UST

25.11.2020

Spis Treści

Zadanie 1

Zadanie 1

Napisz funkcję implementującą metodę najszybszego spadku. Funkcja powinna mieć interfejs analogiczny do zaimplementowanej na zajęciach funkcji gradientu prostego, jednak bez definiowania określonego "learning rate". Zaproponuj jak automatycznie zakończyć działanie funkcji gdy ekstremum zostanie osiągnięte. Metodę przetestuj na przykładzie optymalizacji parametrów modelu wykorzystywanego w metodzie najmniejszych kwadratów.

```
def steepest_descent(X, y, model, model_evaluator,
    jacobian_calculator, max_iters):
    # argumenty jak na zajeciach, gdzie max_iters
    # oznacza maksymaln
    # (niezgry zadan) liczb  iteracji
    pass
```

Zadanie 2

Zadanie 2

Na podstawie implementacji metody Newtona, dokonaj implementacji metody Gaussa-Newtona. Jakie są zalety metody Gaussa-Newtona względem metody Newtona? Jakie wady? Przetestuj zaimplementowaną funkcję na przykładzie problemu najmniejszych kwadratów.

```
def gauss_newton_method(X, y, model,
    model_evaluator, max_iters):
    # argumenty analogiczne jak na zajeciach
    pass
```

Zadanie 3

Zadanie 3

Zaimplementuj funkcję do liczenia Jacobianu dowolnej funkcji względem dowolnej zmiennej. Interfejs powinien być następujący:

```
def universal_jacobian(f, x)
    # f - funkcja (Callable)
    # przymujaca jako argument wektor x
    # x - wektor zawieracy wartosci zmiennej
    # dla ktorych powinien zostac policzony Jacobian
    pass
# Przyklad
grid_1, grid_2, .. grid_N <- dziedzina funkcji
y <- Funkcja N zmiennych
f <- funkcja interpolacyjna na podstawie y
univeral_jacobian(f, np.array([v1, v2, ..., vN]
```

Zacznij od funkcji 1 i 2 zmiennych.
Wykorzystaj gradient centralny.