Łukasz Stępień gr. IV

**Algorytmy geometryczne**

**sprawozdanie z ćw. 2**

1. Cel ćwiczenia:

Ćwiczenie algorytmów wyznaczającą otoczkę wypukłą Grahama i Jarvisa, wizualizacja ich przebiegu oraz porównanie.

2. Dane techniczne:

Język implementacji: Python

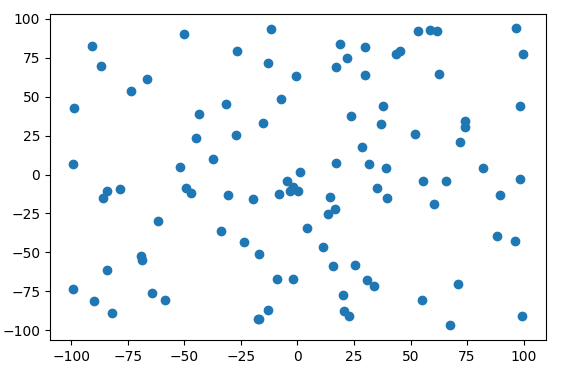
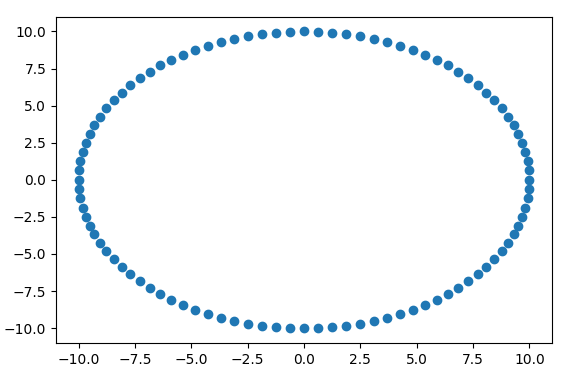
Środowisko programistyczne: Jupyter Notebook

System operacyjny: Microsoft Windows 10 Pro x64

Procesor: Intel(R) Core(TM) i5-10400F CPU @ 2.90GHz, 2904 MHz

3. Zestawy danych i ich wizualizacja:

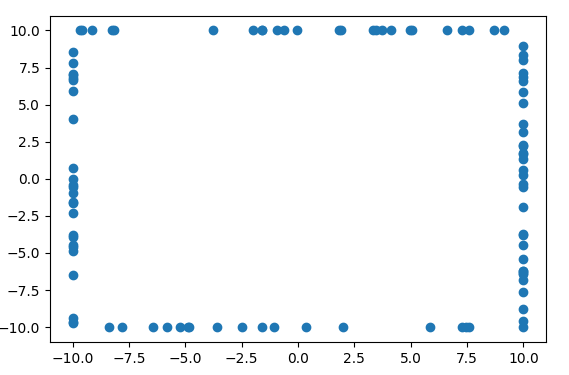
Na początku zaimplementowałem funkcję, które umożliwiają wygenerowanie specjalnych zestawów danych. Poniżej przedstawiam wizualizację dla małych, sprecyzowanych parametrów:

* 100 losowo wygenerowanych punktów o współrzędnych z przedziału [-100, 100]:

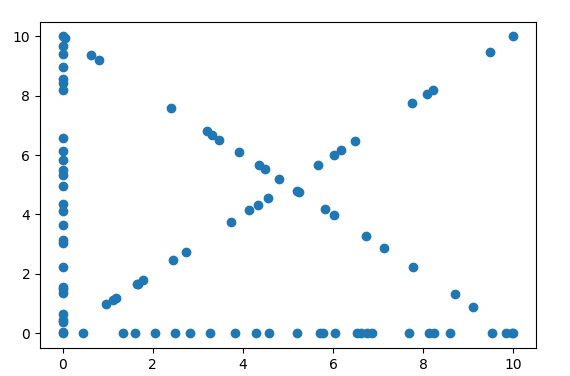
Wykres Zestaw danych A

* 100 losowo wygenerowanych punktów leżących na okręgu o środku (0,0) i promieniu R=10:

Wykres Zestaw danych B

* 100 losowo wygenerowanych punktów leżących na bokach prostokąta o wierzchołkach (-10, 10), (-10,-10), (10,-10), (10,10):

Wykres Zestaw danych C

* wierzchołki kwadratu (0, 0), (10, 0), (10, 10), (0, 10) oraz punkty wygenerowane losowo w sposób następujący: po 25 punktów na dwóch bokach kwadratu leżących na osiach i po 20 punktów na przekątnych kwadratu.

Wykres Zestaw danych D

4. Implementacja przydatnych funkcji

Zaimplementowałem kilka funkcji, z których będę korzystał przy algorytmach Grahama i Jarvisa.

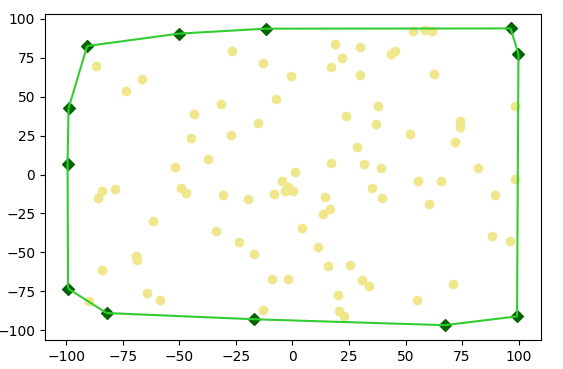
* *det–* oblicza wyznacznik na podstawie macierzy 3x3, kod własny
* *categorize\_point –* klasyfikuje dany punkt względem położenia wobec prostej, korzysta z *det*
* *d –* oblicza odległość pomiędzy dwoma punktami
* *min\_y –* wyznacza punkt z najmniejszą współrzędną y
* *quicksort –* sortuje zbiór punktów względem kąta utworzonego przez półprostą przechodzącą przez ten punkt i środek układu współrzędnych, korzysta z funkcji *categorize\_point*
* *show\_res –* wyświetla wyniki działania algorytmu, umożliwia zapis wyniku do pliku tekstowego

5. Algorytm Grahama

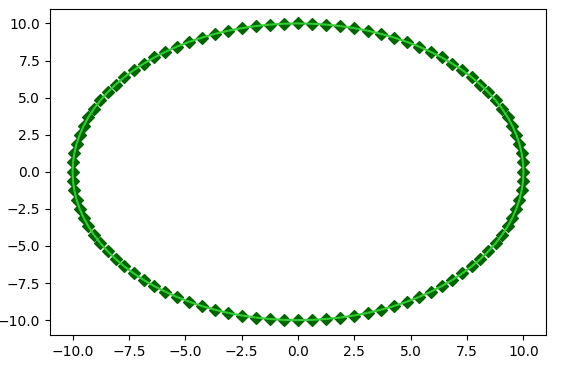
Zaimplementowałem trzy różne warianty algorytmu Grahama:

* *graham* – wyłącznie wyznaczenie otoczki wypukłej zbioru
* *graham\_svis –* wyznaczenie otoczki wypukłej oraz jej wizualizacja
* *graham\_dvis* - wyznaczenie otoczki wypukłej oraz stworzenie wizualizacji przebiegu algorytmu

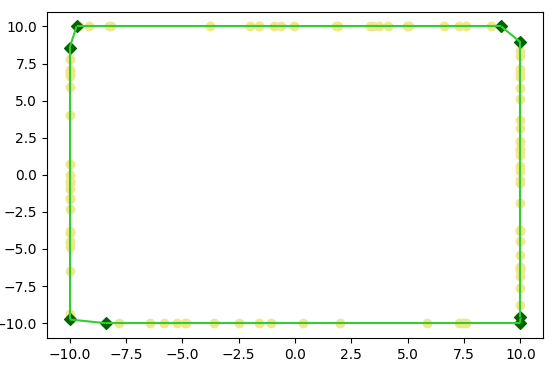
Poniżej przestawiam wynik algorytmu Grahama dla zbiorów z punktu 3.:

*  Zestaw danych A - liczebność punktów w otoczce: 12

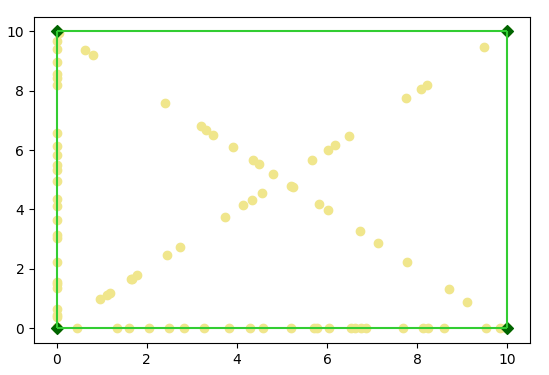
Wykres Wynik algorytmu Grahama dla A

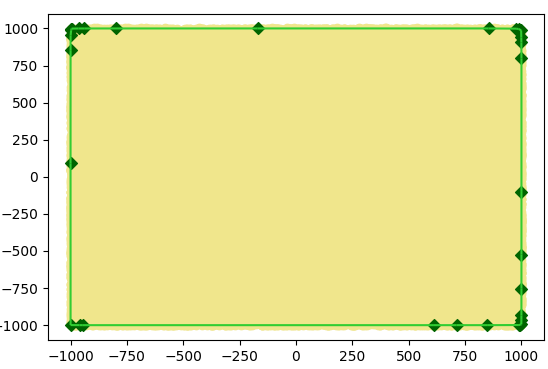
* Zestaw danych B - liczebność punktów w otoczce: 100

Wykres Wynik algorytmu Grahama dla B

* Zestaw danych C - liczebność punktów w otoczce: 8

Wykres Wynik algorytmu Grahama dla C

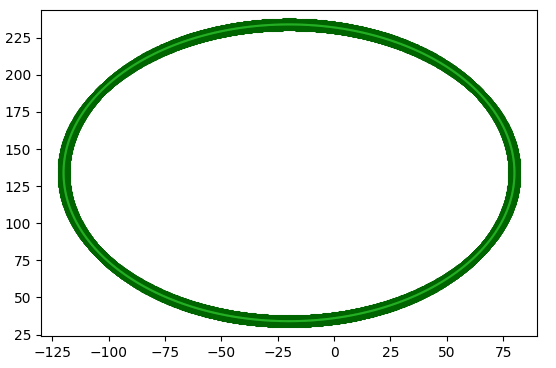
* Zestaw danych D - liczebność punktów w otoczce: 4

Poniżej przestawiam wynik algorytmu Grahama dla zmodyfikowanych zbiorów z punktu 3.:

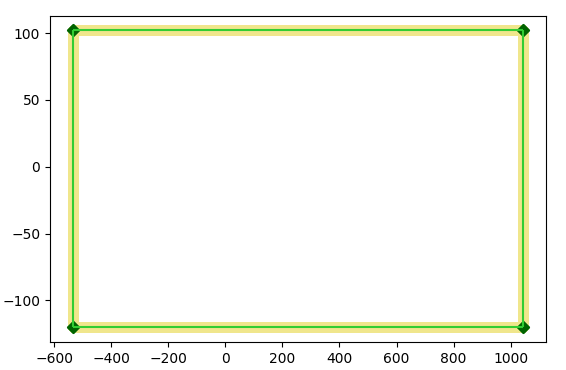
Wykres Wynik algorytmu Grahama dla D

* zmodyfikowany zestaw danych A - liczebność punktów w otoczce: 31

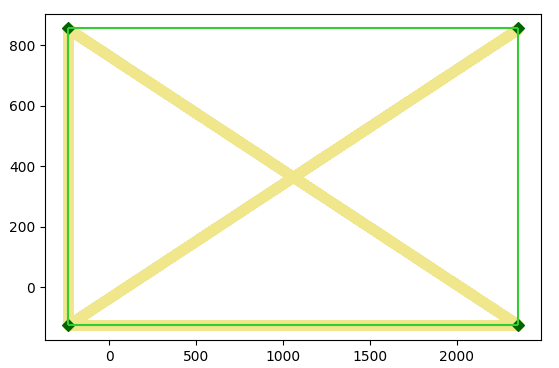
Wykres Wynik algorytmu Grahama dla zmodyfikowanego A

* zmodyfikowany zestaw danych B - liczebność punktów w otoczce: 3000

Wykres Wynik algorytmu Grahama dla zmodyfikowanego B

* zmodyfikowany zestaw danych C - liczebność punktów w otoczce: 8

Wykres Wynik algorytmu Grahama dla zmodyfikowanego C

* zmodyfikowany zestaw danych D - liczebność punktów w otoczce: 4

Wykres Wynik algorytmu Grahama dla zmodyfikowanego D

6. Algorytm Jarvisa

Implementację algorytmu Jarvisa przeprowadziłem tym samym schematem co algorytm Grahama. Umożliwiłem wizualizację oraz zapis wyników, które dla danych zbiorów A-D oraz zmodyfikowanych A-D są identyczne, wykresy również są takie same. Jedyną różnicę w działaniu algorytmów można zauważyć w czasie ich wykonywania.

7. Porównanie algorytmu Grahama oraz Jarvisa

Przeprowadziłem testy porównujące czas dwóch algorytmów na czterech różnych rodzajach zbiorów (A-D) z kolejno zwiększającą się wielkością danych. Dokładną ich liczbę zamieściłem w tabeli 1., a wyniki testów w tabeli 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST** | **ZESTAW A** | **ZESTAW B** | **ZESTAW C** | **ZESTAW D** |
| *1* | 100 | 100 | 100 | 90 |
| *2* | 1 000 | 300 | 1 000 | 400 |
| *3* | 50 000 | 500 | 10 000 | 4 000 |
| *4* | 100 000 | 1 000 | 100 000 | 40 000 |
| *5* | 200 000 | 2 000 | 200 000 | 80 000 |
| *6* | 300 000 | 3 000 | 300 000 | 120 000 |

Tabela Liczebność testowanych zbiorów (w punktach)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **TEST** | **ZESTAW A [s]** | | **ZESTAW B [s]** | | **ZESTAW C [s]** | | **ZESTAW D [s]** | |
| *Graham* | *Jarvis* | *Graham* | *Jarvis* | *Graham* | *Jarvis* | *Graham* | *Jarvis* |
| *1* | 0,0060 | 0,0020 | 0,0110 | 0,0229 | 0,0070 | 0,0010 | 0,0070 | 0,0010 |
| *2* | 0,0190 | 0,0160 | 0,0272 | 0,1965 | 0,0209 | 0,0120 | 0,0090 | 0,0030 |
| *3* | 1,1949 | 1,4092 | 0,0711 | 0,5695 | 0,2982 | 0,1236 | 0,1566 | 0,0290 |
| *4* | 2,4201 | 3,6510 | 0,2789 | 2,1914 | 3,7822 | 1,2910 | 1,8138 | 0,2952 |
| *5* | 5,2897 | 8,2724 | 1,1108 | 8,6645 | 7,9329 | 2,6838 | 3,7741 | 0,6158 |
| *6* | 8,3984 | 12,3651 | 2,4869 | 20,2933 | 12,5669 | 4,0579 | 6,1573 | 0,9979 |

Tabela Wyniki testów czasowych

Powyższe wyniki przestawiłem również za pomocą wykresów liniowych:

* dla zestawu A:

Wykres Zestaw A

* dla zestawu B:

Wykres Zestaw B

* dla zestawu C:

Wykres Zestaw C

* dla zestawu D:

Wykres Zestaw D

7. Podsumowanie

Każdy ze zbiorów z punktu 3. posiada pewne charakterystyczne cechy, które pozwalają przetestować algorytmy wyznaczające otoczkę wypukłą:

* zestaw danych A – zbiór to punkty należące do wnętrza prostokąta, dane najbardziej prawdopodobne, typowy przypadek,
* zestaw danych B – zbiór to punkty należące do okręgu, wszystkie punkty należą do otoczki,
* zestaw danych C – zbiór to punkty leżące na bokach prostokąta, bez punktów na jego wierzchołkach, tylko 8 punktów należących do otoczki – po dwa na każdy bok,
* zestaw danych D - punkty leżące na wierzchołkach kwadratu, jego dwóch bokach oraz przekątnych, tylko 4 punkty w otoczce – na wierzchołkach kwadratu.

W zestawach C i D ważne jest to, że punkty współliniowe nie należą do otoczki, co uwzględniłem w implementacji obu algorytmów.

Analizując i porównując złożoność obliczeniową obu algorytmów (Graham O(*nlogn*), Jarvis O(*nk*), gdzie *n* to liczba punktów w zbiorze, a *k* to liczba punktów należących do otoczki) można zauważyć, że ważną rolę w porównywaniu obu algorytmów odgrywa liczba punktów należących do otoczki. Tam, gdzie ich liczba jest duża, szybciej obliczenia wykona algorytm Grahama, zaś w przeciwnym przypadku szybciej upora się z nimi algorytm Jarvisa. Te różnicę bardzo dobrze widać na wykresach z punktu 7.:

* zestaw A – różnica w czasie wykonywania obu algorytmów nie jest ogromnie duża, lecz algorytm Grahama dla takiego typu zestawu będzie przeprowadzał obliczenia szybciej,
* zestaw B – różnica jest wyraźnie zauważalna (przy 6. teście około 17 sekund), parametr *k* będzie równał się *n*, przez co algorytm Jarvisa w tym zestawie zyskuje nieakceptowalną złożoność O(*n2*),
* zestaw C – w przeciwieństwie do dwóch poprzednich zestawów, tutaj wygrywa algorytm Jarvisa ze względu na bardzo mały parametr *k* (przy 6. teście różnica około 8 sekund),
* zestaw D – bardzo duża przewaga algorytmu Jarvisa ze względu na małą wielkość parametru *k.*

8. Wnioski

W tym ćwiczeniu udało mi się poprawnie zaimplementować dwa algorytmy wyznaczające otoczkę wypukłą danego zbioru punktów. Przedstawiłem również wizualnie ich przebieg krok po kroku. Wykonałem po 6 testów dla każdego rodzaju zbioru danych A-D, dzięki czemu przedstawiłem wady i zalety obu algorytmów. Można z nich wywnioskować, że algorytm Grahama opłaca się używać w przypadkach typowych oraz gdy spodziewamy się dużej ilości punktów w otoczce. Zaś algorytm Jarvisa o wiele lepiej sprawdza się przy małej liczbie punktów w otoczce.