Franciszek Jawor grupa nr 2 15:00 – 16:30 PN

Sprawozdanie z laboratorium numer 2: Otoczka wypukła

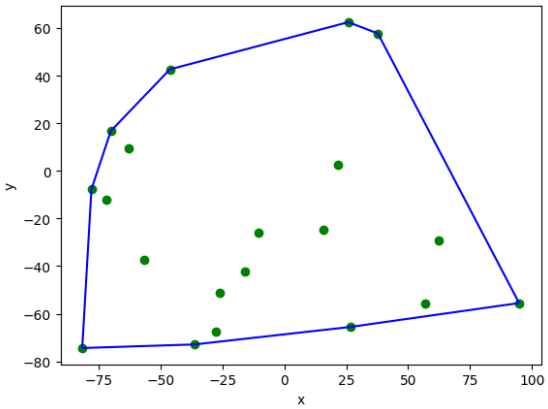
1. Plan ćwiczenia

Celem ćwiczenia, jest zapoznanie się z dwoma algorytmami, służącymi do wyznaczania otoczki wypukłej oraz porównaniem ich, pod kątem szybkości działania, dla różnych zbiorów punktów.

2. Wstęp teoretyczny

Otoczką wypukłą $CH(Q)$, nazywamy najmniejszy zbiór wypukły, zawierający Q, gdzie Q jest dowolnym, niepustym zbiorem punktów.

Natomiast zbiorem wypukłym Q nazywamy zbiór, gdzie dla każdego punktu $p, q \in Q$ odcinek pq jest zawarty w Q.



Rysunek 2.1 Przykładowy zbiór z zaznaczoną otoczką wypukłą

Do wyznaczenia otoczki wypukłej korzystamy z dwóch algorytmów:

Algorytm Grahama

Ideą algorytmu jest wybieranie punktu o najmniejszym kącie, względem poprzedniego punktu. Zaczynamy od znalezienia punktu p_0 z najmniejszą współrzędną y (ten punkt na pewno znajdzie się w otoczce), następnie sortujemy resztę punktów względem kąta, który tworzą z prostą $y = y_0$, usuwamy punkty o tym samym kącie, zostawiając tylko ten, który znajduje się najdalej od p_0. Następnie dodajemy pierwsze 3 punkty na stos, w każdej iteracji sprawdzamy czy kolejny punkt p_i leży po lewej stronie prostej, wyznaczonej przez przedostatni i ostatni punkt ze stosu. Jeśli tak, to idziemy do kolejnego punktu, a jeśli nie, to ściągamy punkty ze stosu, dopóki punkt p_i nie będzie leżał po lewej stronie prostej, lub stos się nie skończy. Na koniec działania algorytmu, stos zawiera wyłącznie punkty należące do otoczki wypukłej.

Algorytm Jarvisa (gift wraping algorithm)

Ideą tego algorytmu, jest „owijanie” zbioru punktów. Ponownie zaczynamy od punktu p_0 z najmniejszą współrzędną x i szukamy punktu, który tworzy najmniejszy kąt prostą $y = y_0$. Następnie szukamy punktu, który tworzy najmniejszy kąt z prostą wyznaczoną przez 2 poprzednie punkty z otoczki i dodajemy go do otoczki, powtarzamy te kroki, aż nie natrafimy na punkt p_0.

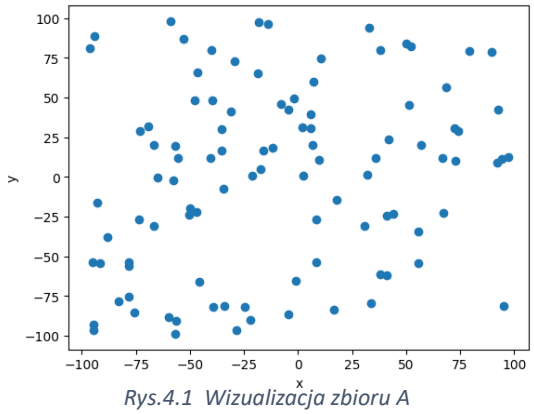
3. Środowisko i użyte narzędzia

Do wykonania ćwiczenia zostało wykorzystane narzędzie jupyter notebook oraz język programowania python. Do wygenerowania pseudolosowych punktów oraz obliczania wyznaczników macierzy, użyta została biblioteka numpy, natomiast do wizualizacji otrzymanych wyników- narzędzie stworzone przez koło naukowe BIT, bazujące na bibliotece matplotlib. Funkcje były uruchamiane na procesorze Intel Core I7 8550U i systemie Windows 11.

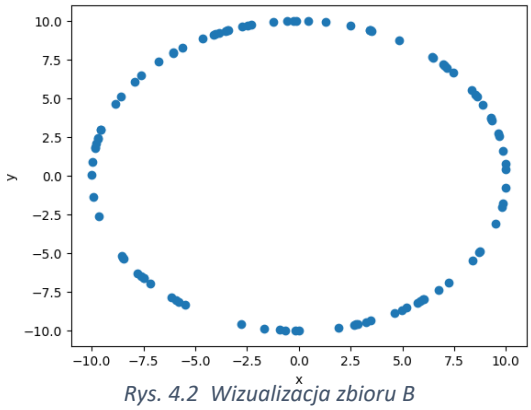
4. Plan ćwiczenia

Pierwszym ćwiczeniem było wygenerowanie 4 zbiorów punktów:

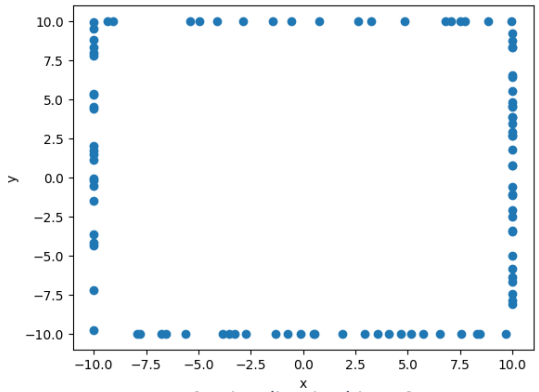
A. 100 losowych punktów z przedziału $[-100, 100]$
B. 100 losowych punktów leżących na okręgu $O = (0, 0),\ R = 10$
C. 100 losowych punktów leżących na bokach prostokąta o wierzchołkach $(-10, -10)$, $(10, -10)$, $(10, 10)$, $(10, -10)$
D. 25 punktów leżących na bokach i 20 punktów leżących na przekątnych kwadratu o wierzchołkach $(0, 0)$, $(0, 10)$, $(10, 10)$, $(10, 0)$



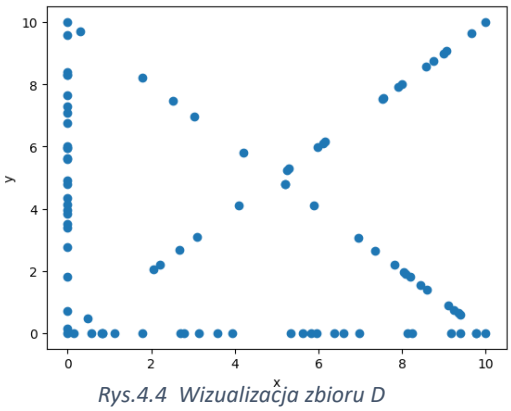
Rys.4.1 Wizualizacja zbioru A



Rys. 4.2 Wizualizacja zbioru B



Rys. 4.3 Wizualizacja zbioru C



Rys.4.4 Wizualizacja zbioru D

Następnym ćwiczeniem jest wyznaczenie otoczki wypukłej dla podanych zbiorów za pomocą obu algorytmów oraz porównanie ich wydajności. W tym celu, w algorytmie Grahama, korzystam z wbudowanej funkcji sort oraz własnej funkcji compare, która określa położenie punktu, względem prostej. W tym celu korzystam z wyznacznika macierzy 2x2. Punkty leżące na tej samej linii (względem pierwszego punktu otoczki), usuwam, za wyjątkiem najdalszego. W algorytmie Jarvisa, również korzystam z wyznacznika macierzy. Tolerancja, którą przyjąłem, to $\boldsymbol{\varepsilon = 10^{-24}}$.

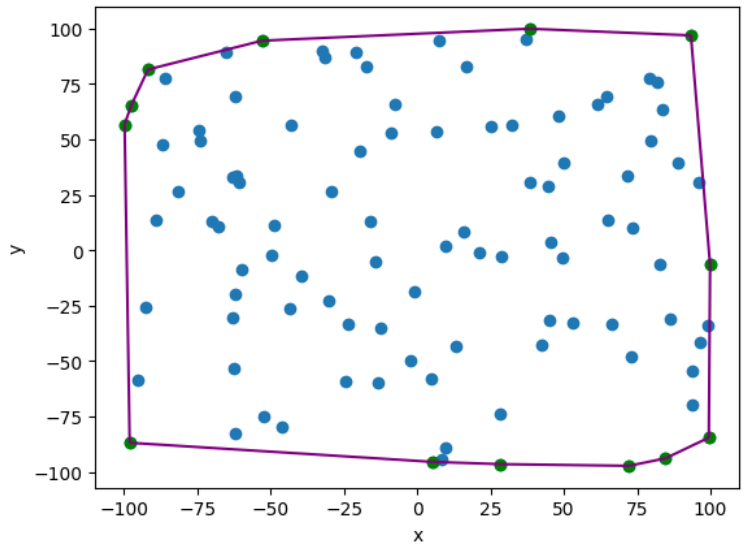
5. Analiza wyników

Najpierw przedstawię liczbę wierzchołków, należących do otoczki, w zależności od użytego algorytmu.

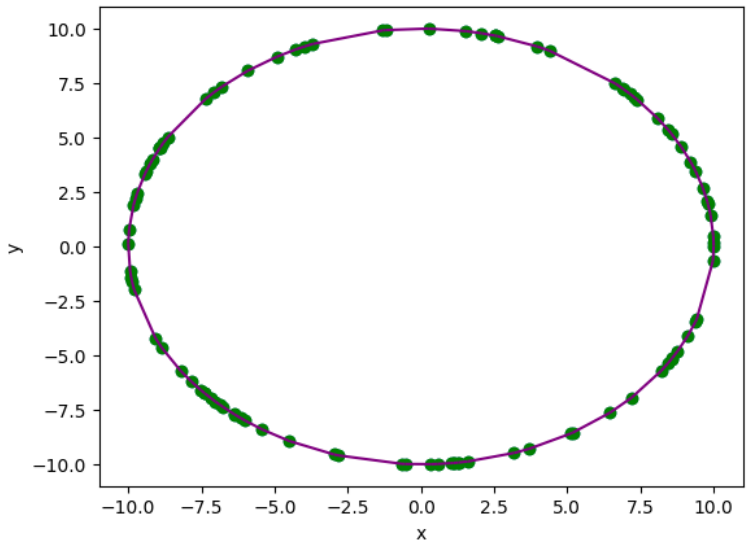
Tabela 5-1 Liczba punktów należących do otoczki w zależności od algorytmu

Algorytm	Zbiór A	Zbiór B	Zbiór C	Zbiór D
Graham	13	100	8	9
Jarvis	13	100	8	9

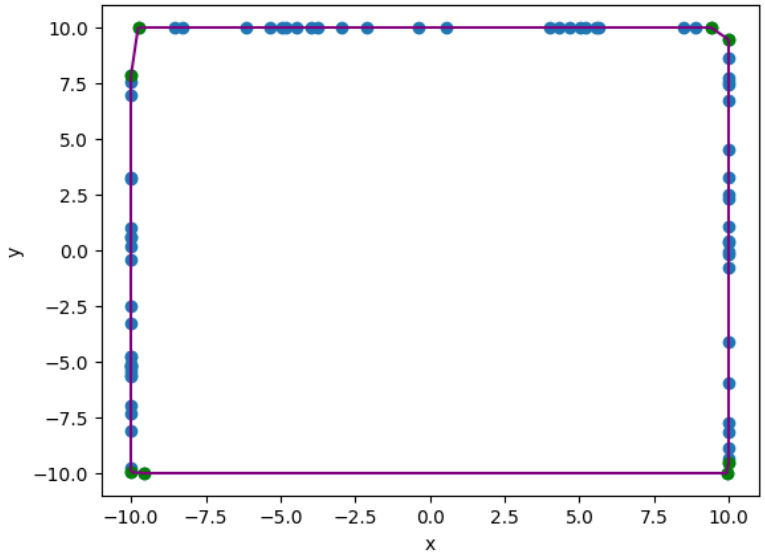
Jak widać, wyznaczone otoczki, są identyczne, niezależnie od użytego algorytmu. Teraz przedstawię wizualizacje otoczek dla powyższych zbiorów. Kolorem zielonym oznaczone zostały punkty tworzące otoczkę, kolorem fioletowym- linie między tymi punktami, a niebieskim- reszta punktów.



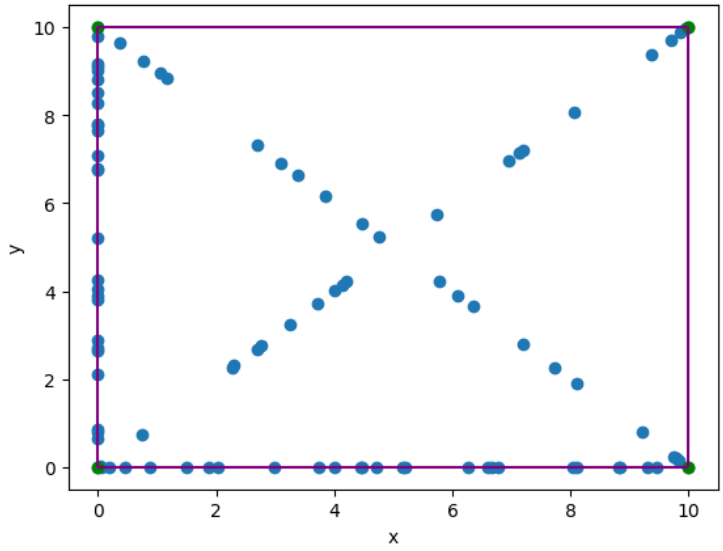
Rys. 5.1 Wizualizacja otoczki dla zbioru A



Rys. 5.2 Wizualizacja otoczki dla zbioru B



Rys. 5.3 Wizualizacja otoczki dla zbioru C



Rys. 5.4 Wizualizacja otoczki dla zbioru D

Złożoność algorytmów

Algorytm Grahama, ze względu na sortowanie punktów, ma złożoność $O(nlogn)$. Natomiast w przypadku algorytmu Jarvisa jest to $O(nk)$, gdzie n to liczba wszystkich punktów w zbiorze, a k- liczba punktów należących do otoczki. Dla małych otoczek algorytm Jarvisa może okazać się skuteczniejszy, jednak pesymistyczna złożoność, to $O(n^2)$, więc dla dużej otoczki, będzie działał dużo gorzej.

Aby dokładniej zaprezentować różnice między algorytmami wprowadzam modyfikację do powyższych zbiorów:

A. n losowych punktów z przedziału $[-1000, 1000]$
B. m losowych punktów leżących na okręgu $O(0,0),\ R = 500$
C. n losowych punktów leżących na bokach prostokąta o wierzchołkach $(-100,-100)$, $(-100,200),\ (200,200),\ (200,-100)$
D. n*0,25 punktów leżących na bokach i n*0,2 punktów leżących na przekątnych kwadratu o wierzchołkach $(-50,-50),\ (50,-50),\ (50,50),\ (-50,50)$

$$n \in \{10^2, 10^3, 10^4, 10^5\}, \qquad m \in \{10^1, 10^2, 10^3, 10^4\}$$

Tabela 5-2 Czasy wykonania algorytmów dla poszczególnych zbiorów

Zbiory punktów	Algorytm	Liczba punktów			
		10^2	10^3	10^4	10^5
Zbiór A	Graham	0,000s	0,0116s	0,0939s	1,2211s
	Jarvis	0,000s	0,0095s	0,1735s	1,5253s
Zbiór C	Graham	0,000s	0,0095s	0,1531s	2,1355s
	Jarvis	0,000s	0,0055s	0,0541s	0,5313s
Zbiór D	Graham	0,0015s	0,0115s	0,1501s	2,553s
	Jarvis	0,000s	0,003s	0,0242s	0,2347s

Tabela 5-3 Czasy wykonania algorytmów dla zbioru B

Zbiory punktów	Algorytm	Liczba punktów			
		10^1	10^2	10^3	10^4
Zbiór B	Graham	0,000s	0,0005s	0,0076s	0,1114s
	Jarvis	0,000s	0,0071s	0,7615s	78,0294s

Analizując wyniki z tabeli 5-2 oraz 5-3, widać, że algorytm Grahama radzi sobie zdecydowanie lepiej przy dużych otoczkach (zbiór B). Natomiast algorytm Jarvisa jest dużo szybszy przy małych otoczkach (zbiór C i D), ponieważ jego złożoność jest niemal liniowa. Dla zbioru D, Jarvis ma złożoność O(4n).

6. Wnioski

Powyższe zbiory, bardzo dobrze ukazały wady i zalety poszczególnych algorytmów. Na podstawie otrzymanych wyników, można zauważyć, że dobranie odpowiedniego algorytmu, do wyznaczania otoczki, nie jest takie oczywiste. Testy na zbiorze B, ukazały słabość Jarvisa, ponieważ wszystkie punkty należały do otoczki. Jednak w zbiorach, dla których otoczka, była prostokątem, algorytm Grahama radził sobie gorzej.