1. **Dane techniczne:**

Język: Python

Translator: Visual Studio Code

Procesor: AMD Ryzen 7 5800H

System operacyjny: Windows 11

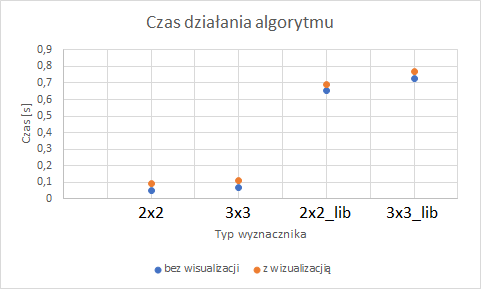
1. **Realizacja ćwiczenia:**

Generujemy losowe punkty na płaszczyźnie, na okręgu i na odcinku przy użyciu funkcji uniform z biblioteki random. Punkty na okręgu są wybierane przy pomocy równania parametrycznego okręgu, a dla punków na odcinku wybieramy losową współrzędną x i dla niej obliczamy y odpowiednio dla podanych wierzchołków odcinka. Do określenia po której stronie prostej znajduje się punkt użyto wyznaczników 2x2 i 3x3 z czego każdy z nich miał 2 wariancje: obliczanie wyznacznika metodą Sarrusa, obliczanie wyznacznika z użyciem funkcji det z biblioteki numpy. Podczas analizowania wyników sprawdzano wyniki i czas działania poszczególnych wyznaczników oraz zmieniano tolerancje dla zera w zakresie (,).

Zmieniamy również float64 na float32, żeby zobaczyć co się dzieje dla mniejszej precyzji.

1. **Analiza wyników:**

Na początku zmierzono czas działania algorytmu przy użyciu różnych wyznaczników:

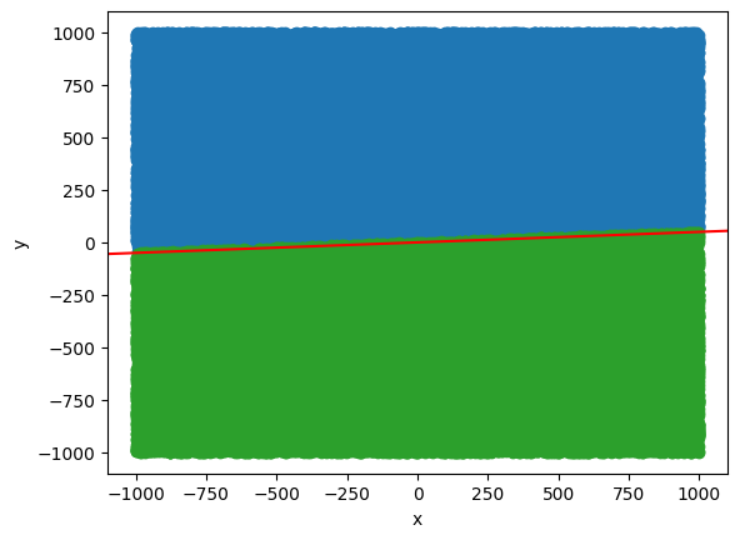


Rysunek 1: Wykres czasu działania algorytmu dla poszczególnych wyznaczników

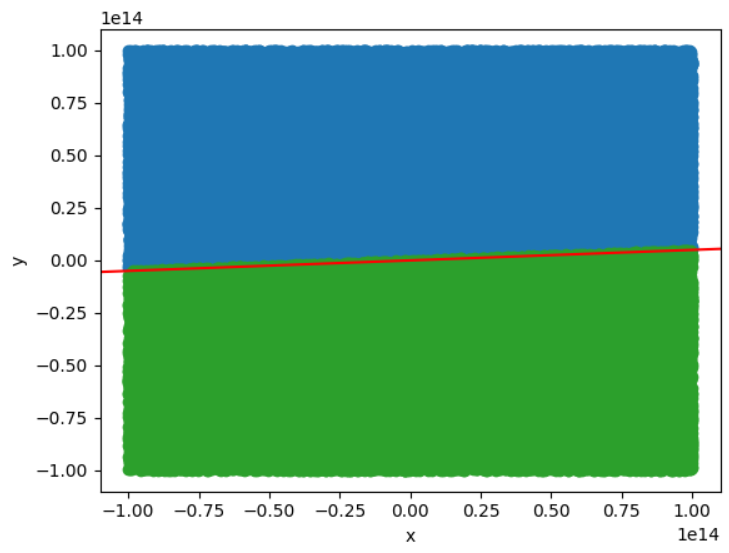
Pomiary na rysunku 1. wykonano dla 10000 punktów na płaszczyźnie z zakresu (-1000,1000). Zakres i umieszczenie punktów nie miało większego znaczenia w czasie działania, więc pomiary wykonano tylko dla tego zakresu. Łatwo zauważyć, że czas liczenia wyznaczników metodą sarrusa jest znacznie szybszy niż używanie funkcji bibliotecznych, można też zauważyć, że policzenie wyznacznika 2x2 jest minimalnie szybsze niż liczenie wyznacznika 3x3.

Następnie przeanalizowano podziały punktów względem prostej dla różnych rozmieszczeń punktów (10000 punktów na płaszczyźnie w zakresie (-1000,1000), 10000 punktów na płaszczyźnie w zakresie (-,), 1000 punków na okręgu, 1000 punktów na odcinku ab a=[-1.0, 0.0], b=[1.0, 0.1]).

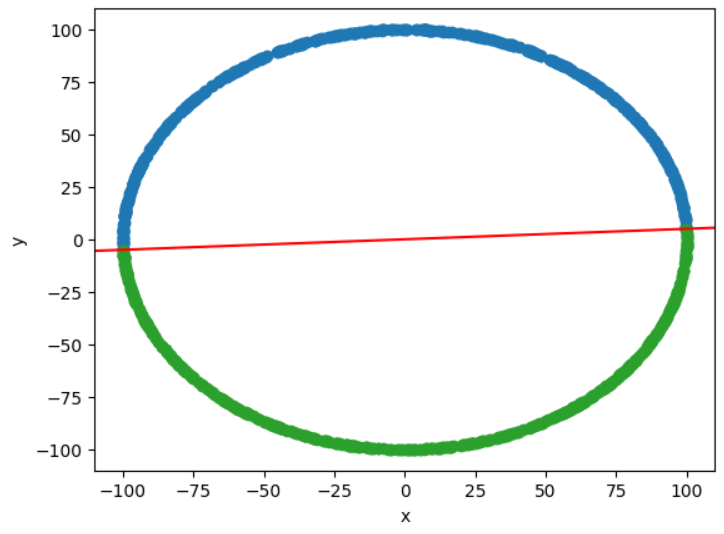
Punkty niebieskie będą oznaczać punkty po lewej stronie prostej, zielone po prawej stronie prostej, a pomarańczowe na prostej.



Rysunek 2: Wykres rozłożenia 10000 punktów względem prostej ab dla zakresu (-1000, 1000)



Rysunek 3: Wykres rozłożenia 10000 punktów względem prostej ab dla zakresu (-,)



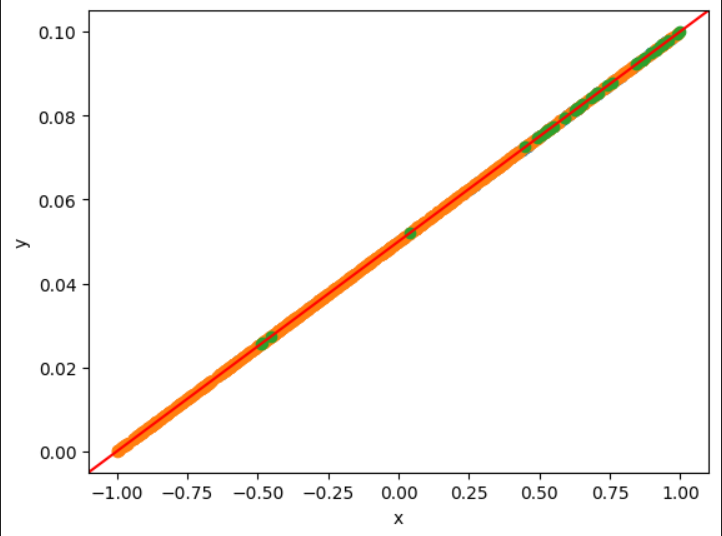
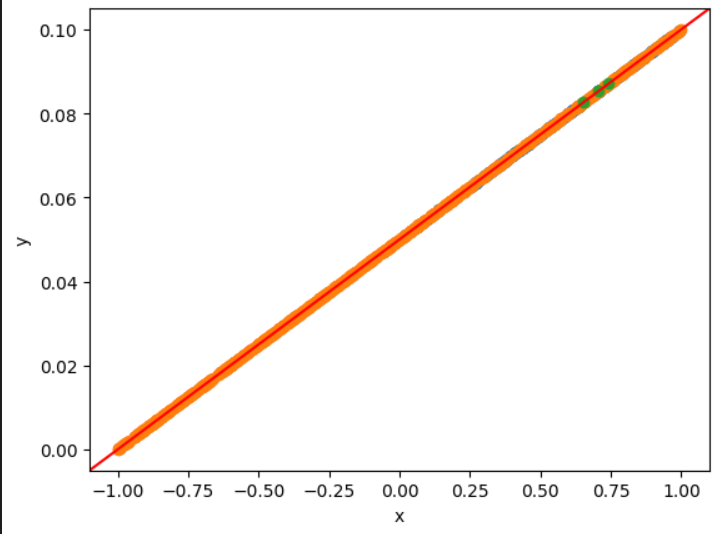
Rysunek 4: Wykres rozłożenia 1000 punktów na okręgu o promieniu 100

Dla rysunku 2. otrzymaliśmy wartości: 49948 punkty po lewej stronie, 50052 punktów po prawej stronie i 0 punktów na prostej.

Dla rysunku 3. otrzymaliśmy wartości: 50054 punkty po lewej stronie, 49946 punktów po prawej stronie i 0 punktów na prostej.

Dla rysunku 4. otrzymaliśmy wartości: 499 punkty po lewej stronie, 501 punktów po prawej stronie i 0 punktów na prostej.

We wszystkich tych przypadkach zmienianie wyznacznika nie miało żadnego wpływu na wynik. Dla wszystkich wartości epsilon mniejszych od wynik także nie uległ zmianie i żaden punkt nie został zaliczony do prostej.

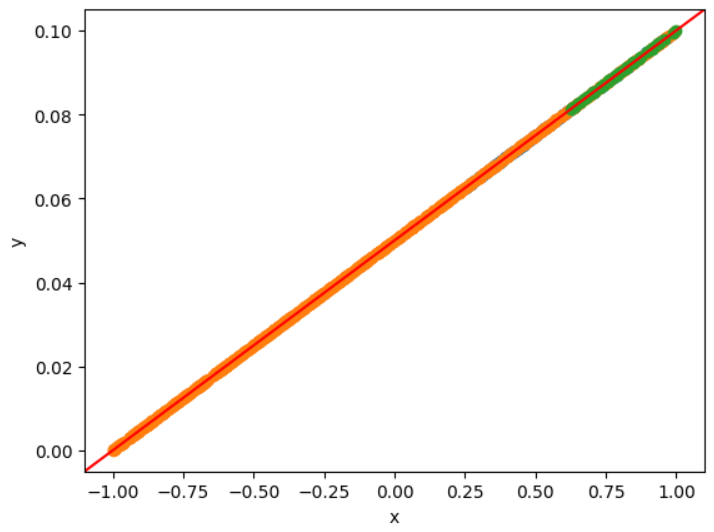
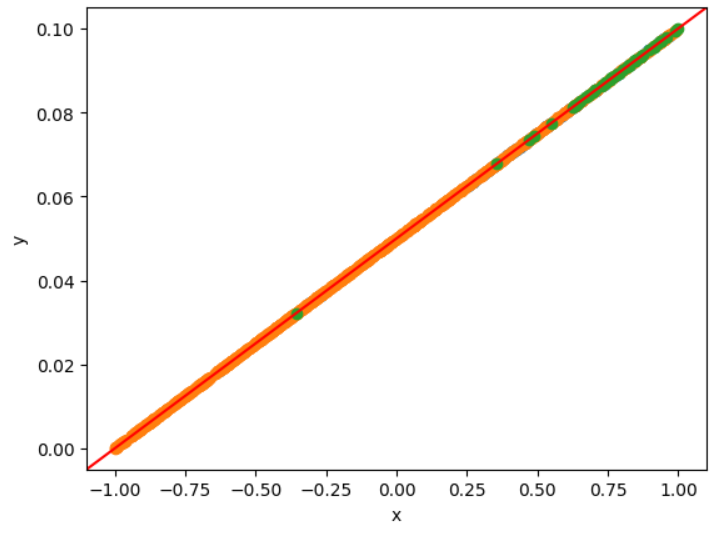


Rysunek 5,6: Wykresy rozmieszczenia 1000 punktów na odcinku ab dla epsilon =

Rysunek 5. Przedstawia wyznacznik biblioteczny 3x3 a rysunek 6. biblioteczny 2x2.

Dla rysunku 5. otrzymaliśmy wartości: 187 punkty po lewej stronie, 3 punktów po prawej stronie i 810 punktów na prostej.

Dla rysunku 6. otrzymaliśmy wartości: 88 punkty po lewej stronie, 38 punktów po prawej stronie i 874 punktów na prostej.

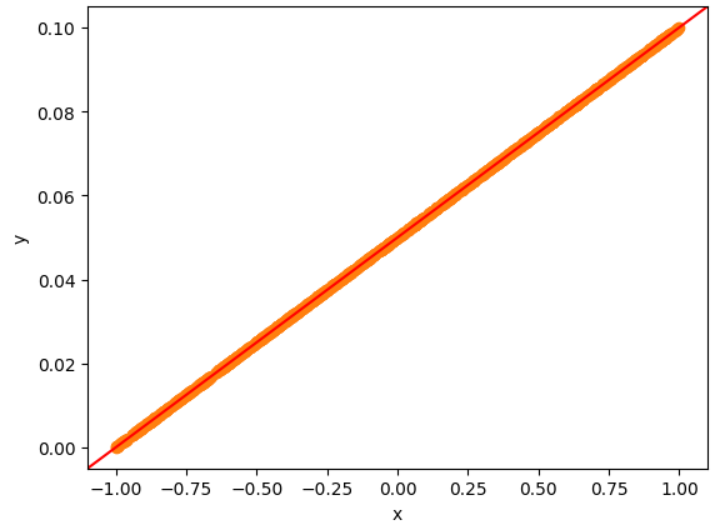
Rysunek 6,7: Wykresy rozmieszczenia 1000 punktów na odcinku ab dla epsilon =

Rysunek 6. Przedstawia wyznacznik 3x3 a rysunek 7. Wyznacznik 2x2.

Dla rysunku 6. otrzymaliśmy wartości: 119 punkty po lewej stronie, 46 punktów po prawej stronie i 835 punktów na prostej.

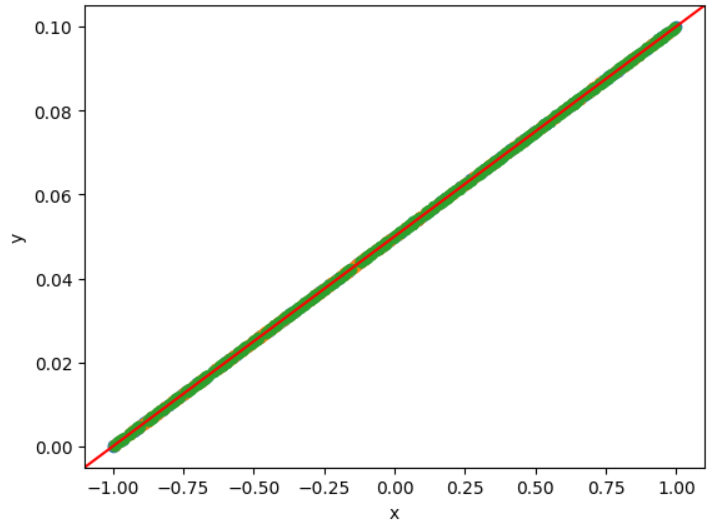
Dla rysunku 7. otrzymaliśmy wartości: 96 punkty po lewej stronie, 46 punktów po prawej stronie i 858 punktów na prostej.

Można zauważyć że znacznie więcej punktów ląduje po lewej stronie odcinka niż po prawej. Można także zauważyć że przy użyciu wyznacznika 3x3 trochę mniej punktów ląduje na odcinku niż przy użyciu wyznacznika 2x2. Użycie funkcji bibliotecznych w tym przypadku nie ma większego wpływu na wynik, a jedyną różnicą jest to że więcej punków ląduje po lewej stronie odcinka.



Rysunek 8: Wykresy rozmieszczenia 1000 punktów na odcinku ab dla epsilon >=

Dla epsilon >= , niezależnie od wyboru wyznacznika, wszystkie punkty trafiają na odcinek tak jak na rysunku 8.

Po zmniejszeniu precyzji z float64 na float32 dla epsilon >= wykresy wyglądały tak samo jak na rysunku 8. niezależnie od wyboru wyznacznika.

Jednak dla epsilon < większość punków przestała być zaliczana do prostej (rysunek 9).

Wybór wyznacznika nie miał w tym przypadku większego znaczenia i wyniki były rozłożone równomiernie.

Otrzymaliśmy wartości: 343 punkty po lewej stronie, 370 punktów po prawej stronie i 287 punktów na prostej. Rysunek 9: Rozmieszczenie punktów dla eps <

1. **Podsumowanie:**

Wyznaczniki obliczane metodą Sarrusa są o wiele szybsze, jednak co do dokładności obliczeń nie różnią się bardzo od ich bibliotecznych odpowiedników.

Do sprawdzania, czy pewien punkt leży na prostej, trzeba uwzględnić tolerancję dla zera co najmniej , jednak nie należy wybierać wartości większych niż , ponieważ punkty które nie należą do prostej mogą zostać przypadkiem do niej zaliczone.