Algorytmy Geometryczne

Sprawozdanie z ćwiczenia 1. „Predykaty geometryczne”

Maciej Wiśniewski

Grupa 3 Poniedziałek 16.45 A

Data wykonania 14.10.2024 Data oddania 28.10.2024

1. Dane techniczne

Specyfikacja komputera: system ***Ubuntu 24.04.01 Linux 5.15 x64***, procesor ***AMD Ryzen 7 5825U with Radeon 2GHz 8 rdzeni***, ***16GB pamięci RAM.***

Ćwiczenie zostało napisane w języku *Python 3.9.2*0 w *Jupyter Notebook* w środowisku programistycznym *Visual Studio Code*. Aby wykonać ćwiczenie posłużono się bibliotekami: *matplotlib, numpy i pandas*. Do wykonania wizualizacji użyto narzędzia graficznego wykonanego przez *Koło Naukowe BIT*.

2. Cel ćwiczenia

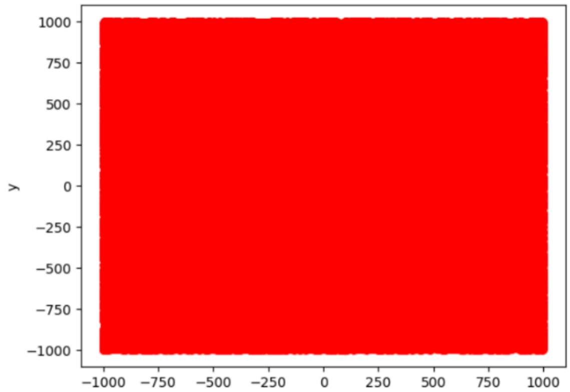
Celem ćwiczenia było zaznajomienie się z podstawowymi pojęciami *geometrii obliczeniowej* oraz implementacja kluczowych *predykatów geometrycznych*, takich jak określanie położenia punktu względem prostej. Ponadto zadanie obejmowało przeprowadzenie *testów, wizualizację uzyskanych danych* oraz *opracowanie wyników.*

3. Realizacja ćwiczenia

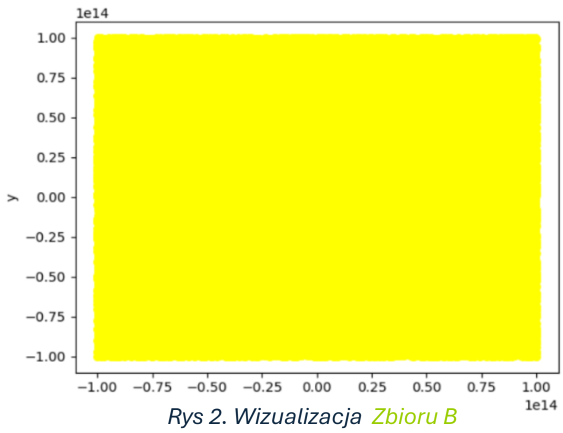
Zadanie oparto o generacji czterech zbiorów i następnej kategoryzacji punktów w nich zawartych względem orientacji do odcinka $\overrightarrow{ab}$ dla $\boldsymbol{a} = [-1.0, 0.0]$, $\boldsymbol{b} = [1.0, 0.1]$. Zbiory, o które oparto ćwiczenie to:

- **Zbiór A** - 10^5 punktów na płaszczyźnie $[-1000, 1000]^{\,2}$
- **Zbiór B** - 10^5 punktów na płaszczyźnie $[-10^{14}, 10^{14}]^{\,2}$
- **Zbiór C** - 1000 punktów leżących na okręgu w układzie współrzędnych o środku $\boldsymbol{O} = (0, 0)$ i promieniu $\boldsymbol{R} = 100$
- **Zbiór D** - 1000 punktów o współrzędnych niezależnych z przedziału [−1000, 1000] leżących na prostej zdefiniowanej przez wektor $(\boldsymbol{a}, \boldsymbol{b})$, gdzie $\boldsymbol{a} = [-1.0, 0.0]$, $\boldsymbol{b} = [1.0, 0.1]$

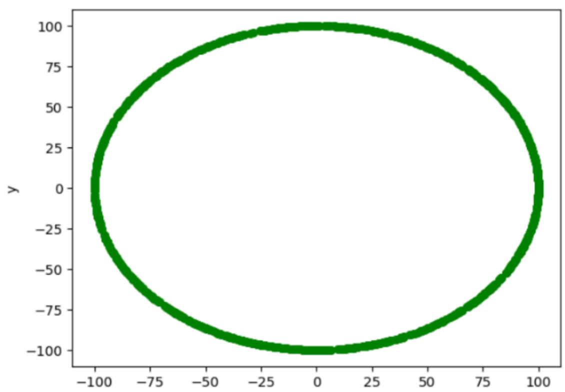
Aby uzyskać równomierny rozkład punktów w zbiorach użyto metody ***random.uniform*** dostępnej w bibliotece ***numpy,*** która generuje liczby typu float64 z wybranego zakresu domkniętego. Dodatkowo, każdy zestaw punktów został wygenerowany używając typów liczbowych *float64* i *float32* aby móc określić ich potencjalny wpływ na wyniki. Dodatkowo podczas generowania punktów z **Zbiorze C** użyto metod ***numpy.cos()*** i ***numpy.sin()*** korzystając z parametrycznego równania okręgu. Punkty położone na prostej w **Zbiorze D** zostały wygenerowane używając równania parametrycznego prostej przechodzącej przez dwa punkty $\boldsymbol{a}$ i $\boldsymbol{b}$ w przestrzeni dwuwymiarowej.



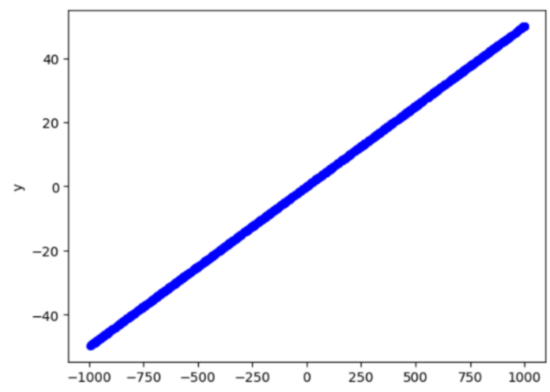
Rys. 1 Wizualizacja Zbioru A



Rys 2. Wizualizacja Zbioru B



Rys. 3 Wizualizacja Zbioru C



Rys. 4 Wizualizacja Zbioru D

Wszystkie punkty z określonych zbiorów zostały sklasyfikowane według ich położenia względem linii wyznaczone przez wektor ***ab***, gdzie ***a*** = [−1.0, 0.0], ***b*** = [1.0, 0.1]. Dla każdego punktu przypisujemy mu jedno z podanych położeń względem prostej oraz kolor : ***na prawo od prostej*** - ***pomarańczowy***, ***na lewo od prostej - zielony***, ***na prostej*** - ***fioletowy***. Położenie to określamy dzięki obliczeniu wyznacznika macierzy wybranego punktu wraz z punktami ***a*** = [−1.0, 0.0] oraz ***b*** = [1.0, 0.1]. Do generowania wykorzystujemy cztery różne metody liczenia wyznacznika(det) macierzy:

- ***mat_det_2x2*** – wyznacznik macierzy 2x2 liczony „ręcznie”,
- ***mat_det_2x2_lib*** – wyznacznik macierzy liczony z wykorzystaniem metody ***numpy.linalg.det(),***
- ***mat_det_3x3*** – wyznacznik macierzy 3x3 liczony „ręcznie”,
- ***mat_det_3x3_lib*** – wyznacznik macierzy 3x3 liczony z wykorzystaniem metody ***numpy.linalg.det().***

Dwie różne precyzje float’a:

- ***float64,***
- ***float32.***

Cztery różne tolerancje błędów:

- ***0,***
- $\mathbf{10^{-8}}$,
- $\mathbf{10^{-10}}$,
- $\mathbf{10^{-12}}$,
- $\mathbf{10^{-14}}$.

Podsumowując dane dla każdego zbioru zostały wygenerowane, uwzględniając wszystkie kombinacje następujących cech: metoda obliczania wyznacznika, typ liczbowy oraz tolerancja zera. W efekcie powstało łącznie 40 przypadków dla każdego ze zbiorów. Informacje dotyczące użytych parametrów podczas generacji oraz liczby punktów przypisanych do poszczególnych wartości kategorii zapisano w plikach CSV z użyciem biblioteki ***pandas***. Nazwy plików odpowiadają nazwom poszczególnych zbiorów danych.

Do wizualizacji danych posłużono się narzędziem graficznym przygotowanym przez Koło Naukowo BIT. Do wizualizacji punktów użyto metody ***Visualizer().*** Zakwalifikowanym punktom przydzielony kolory: na lewo od prostej - ***zielony***, na prawo od prostej – ***pomarańczowy,*** na prostej - ***fioletowy***. ***Czerwoną*** linią zaznaczono odcinek ***ab***.

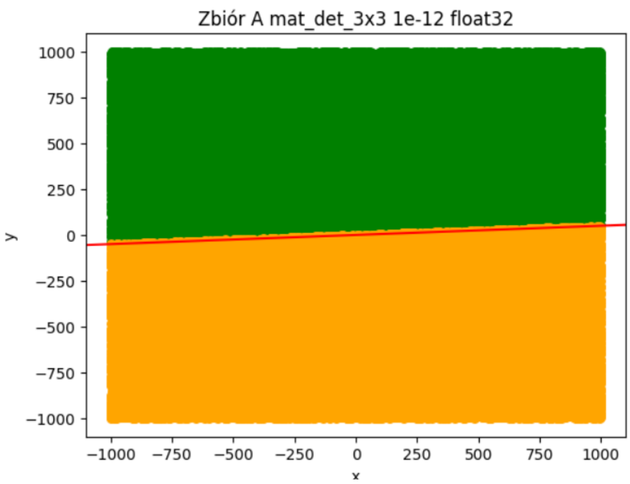
4. Analiza wyników

4.1. Zbiór A

W **Zbiorze A** kategoryzacja punktów była identyczna niezależnie od parametrów generowania danych. Liczba punktów w każdej kategorii została przedstawiona w Tabeli 1 oraz na Wykresie 1.

Punkty_na_lewo	Punkty_na_linii	Punkty_na_prawo
49960	0	50040

Tabela 1: Wyniki kategoryzacji punktów ze **Zbioru A**



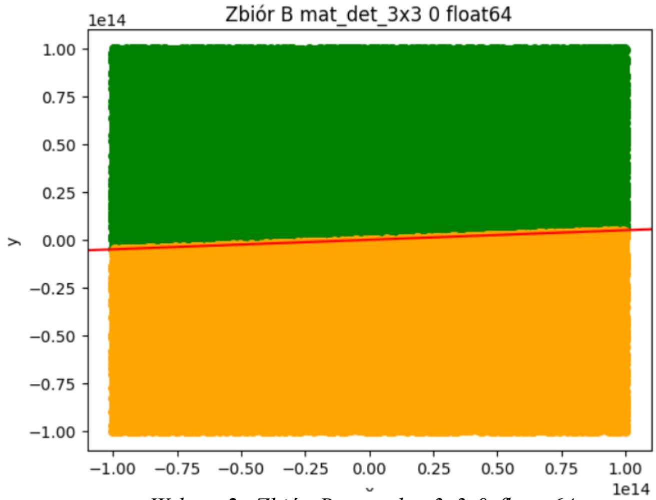
Wykres 1: Zbiór A mat_det_3x3 1e-12 float 32
Przykładowy wykres ze ***Zbioru A***

4.2. Zbiór B

W Zbiorze B głównym czynnikiem różnicującymi wyniki kategoryzacji punktów były metody obliczania wyznacznika macierzy, pod uwagę należy wziąć różnice pomiędzy metodami obliczenia wyznacznika macierzy **mat_det_2x2** i **mat_det_2x2_lib**,a metodami uzyskiwania wyznacznika macierzy **mat_det_3x3** i **mat_det_3x3_lib**. Wszystkie wyniki dla metod liczenia wyznacznika macierzy **mat_det_3x3** i **mat_det_3x3_lib**. wyglądały tak jak w Tabeli 2 i na Wykresie 2:

Punkty_na_lewo	Punkty_na_linii	Punkty_na_prawo
50006	0	49994

Tabela 2: Wyniki kategoryzacji punktów ze Zbioru B dla macierzy **mat_det_3x3** i **mat_det_3x3_lib**



Wykres 2: Zbiór B mat_det_3x3 0 float 64 Przykładowy wykres ze Zbioru B z wyznacznikami macierzy 3x3

Dla pozostałych wizualizacji, czyli tych zawierających metody obliczeniowe macierzy **2x2** i **2x2_lib,** znaczący wpływ na wyniki miał wybór między ***floatem32*** i ***floatem64***. Co warto wspomnieć, dla dokładność **epsilon = 0** wystąpiły punkty, które zostały zakwalifikowane jako leżące na prostej. Różnice zostały w poniższej tabeli :

float	wyznacznik	epsilon	Punkty_na_lewo	Punkty_na_linii	Punkty_na_prawo
float64	mat_det_2x2	0.0	50004	4	49992
float64	mat_det_2x2	1e-14	50004	0	49996
float64	mat_det_2x2	1e-12	50004	0	49996
float64	mat_det_2x2	1e-10	50004	0	49996
float64	mat_det_2x2	1e-08	50004	0	49996
float64	mat_det_2x2_lib	0.0	50001	7	49992
float64	mat_det_2x2_lib	1e-14	50001	0	49999
float64	mat_det_2x2_lib	1e-12	50001	0	49999
float64	mat_det_2x2_lib	1e-10	50001	0	49999
float64	mat_det_2x2_lib	1e-08	50001	0	49999
float32	mat_det_2x2	0.0	0	100000	0
float32	mat_det_2x2	1e-14	0	0	100000
float32	mat_det_2x2	1e-12	0	0	100000
float32	mat_det_2x2	1e-10	0	0	100000
float32	mat_det_2x2	1e-08	0	0	100000
float32	mat_det_2x2_lib	0.0	6663	86673	6664
float32	mat_det_2x2_lib	1e-14	6663	0	93337
float32	mat_det_2x2_lib	1e-12	6663	0	93337
float32	mat_det_2x2_lib	1e-10	6663	0	93337
float32	mat_det_2x2_lib	1e-08	6663	0	93337

Tabela 3: Porównanie kategoryzacji ze Zbioru B dla wyznaczników **2x2** i **2x2_lib**

4.2.1. Analiza wyników

Na podstawie tabeli można wyciągnąć kilka wniosków dotyczących rozkładu punktów względem linii, w zależności od zastosowanych parametrów: typu liczbowego (**float64** lub **float32**), metody wyznaczania wyznacznika (**mat_det_2x2** i **mat_det_2x2_lib**), oraz wartości **epsilon**, która określa tolerancję zera. Analizę można podzielić na kilka aspektów:

1. Typ liczbowy:
   float64: Gdy używany jest typ **float64**, liczba punktów zaklasyfikowanych jako **"Punkty na lewo"**, **"Punkty na linii"** i "Punkty na prawo" zmienia się w zależności od wartości **epsilon** oraz metody obliczeń. Jednak w większości przypadków nie występuje wiele punktów na linii (liczba ta wynosi głównie 0).
   float32: W przypadku **float32**, niemal we wszystkich konfiguracjach znaczna liczba punktów (86,673) zostaje zaklasyfikowana jako leżące na linii, co sugeruje, że niższa precyzja **float32** wpływa na dokładność klasyfikacji. Wartość **epsilon** nie ma znaczącego wpływu na wyniki, co sugeruje, że dokładność samego typu **float32** dominuje nad wpływem tolerancji **epsilon**.
2. Wartość epsilon:
   Przy **epsilon = 0** w przypadku **float64**, klasyfikacja punktów jest wyraźnie podzielona między "na lewo" i "na prawo", natomiast brak jest punktów na linii.
   Dla wyższych wartości **epsilon**, takich jak **1e-8**, zauważalnie większa liczba punktów jest klasyfikowana jako leżące na linii. Dotyczy to szczególnie metody **mat_det_2x2_lib** z typem **float32**, gdzie dominują punkty zaklasyfikowane jako leżące na linii.

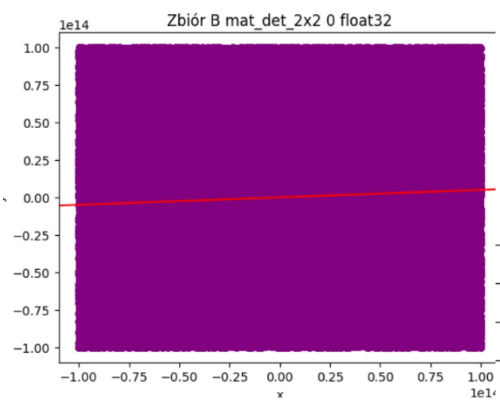
Poddajmy szczegółowej analizie przypadki w Tabeli 4, gdzie wystąpiły pojedyncze punkty na prostej:

float	wyznacznik	epsilon	Punkt X	Punkt Y	Y – X*(0.05)+0.05
float64	mat_det_2x2	0.0	85504712754837.25	4281580026820.7344	6344389078,82
float64	mat_det_2x2	0.0	-67859098213101.94	-3396479513077.672	-3524602422,62
float64	mat_det_2x2	0.0	-52971292046400.32	-2654751617818.828	-6187015498,85
float64	mat_det_2x2	0.0	78689836007142.16	3954278919949.125	19787119591,97
float64	mat_det_2x2_lib	0.0	85504712754837.25	4281580026820.7344	6344389078,82
float64	mat_det_2x2_lib	0.0	-94284372728446.16	-4666523144133.406	47695492288,85
float64	mat_det_2x2_lib	0.0	-67859098213101.94	-3396479513077.672	-3524602422,62
float64	mat_det_2x2_lib	0.0	-52971292046400.32	-2654751617818.828	-6187015498,85
float64	mat_det_2x2_lib	0.0	-51600507039819.13	-2571620443334.9688	8404908655,95
float64	mat_det_2x2_lib	0.0	-64740537784892.97	-3228943223308.047	8083665936,56
float64	mat_det_2x2_lib	0.0	-68004561289321.22	-3388887654508.0625	11340409957,95

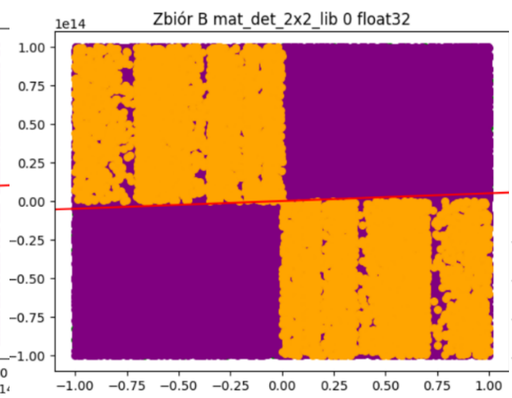
Tabela 4: Rozkład punktów w Zbiorze B, gdzie występują pojedyncze punkty na linii

Jak wykazała ręczna analiza, rzeczywista wartość różni się od tej uzyskanej w kategoryzacji.

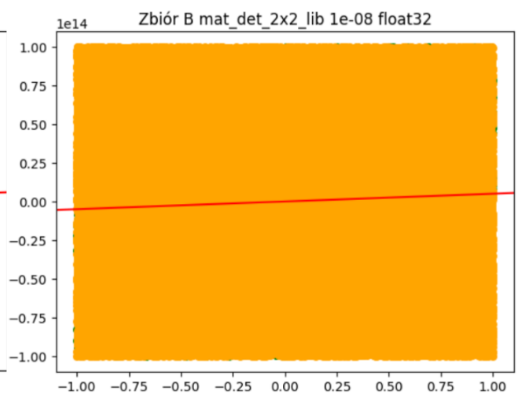
Dla precyzji **float32** można zaobserwować zdecydowanie wyróżniające się wyniki w Tabeli 3. Są one spowodowane ograniczonym zakresem typu **float32**. Natomiast przy wykorzystaniu wyznacznika macierzy **3x3** wyniki są zbliżone do tych uzyskanych dla precyzji **float64**.



Wykres 3: Zbiór B met_det_2x2 0 float32



Wykres 4: Zbiór B met_det_2x2_lib 0 float32



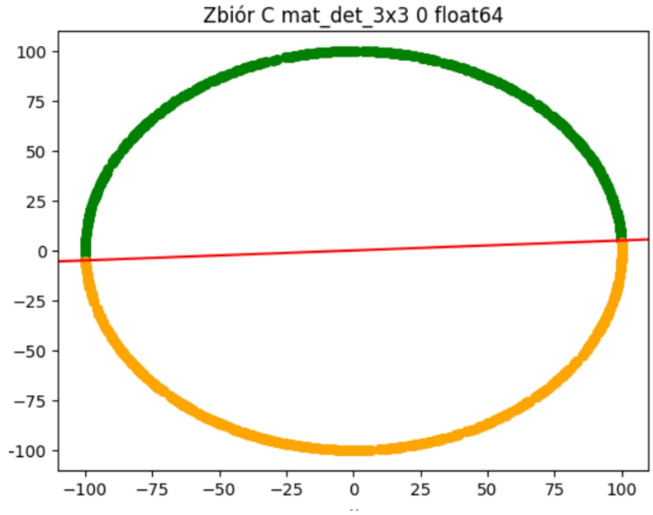
Wykres 5: Zbiór B met_det_2x2 1e-08 float32

4.3. Zbiór C

Analogicznie do **Zbioru A**, w **Zbiorze C** niezależnie od parametrów generowania danych, kategoryzacja punktów była identyczna. Ilość punktów w każdej z kategorii przedstawia Tabela 5 oraz Wykres 6.

Punkty_na_lewo	Punkty_na_linii	Punkty_na_prawo
481	0	519

Tabela 5: Wyniki kategoryzacji punktów ze **Zbioru C**

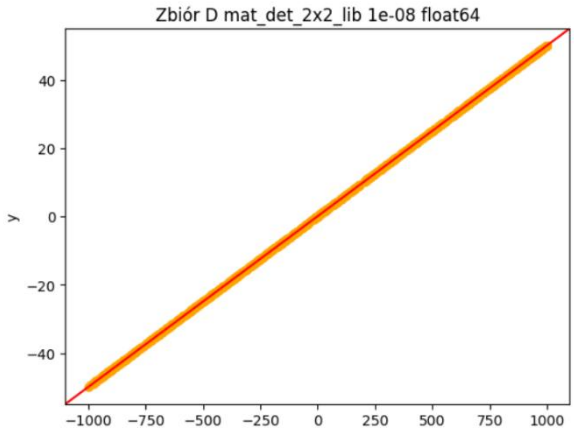


Wykres 6: Zbiór C mat_det_3x3 0 float 64
Przykładowy wykres ze Zbioru C

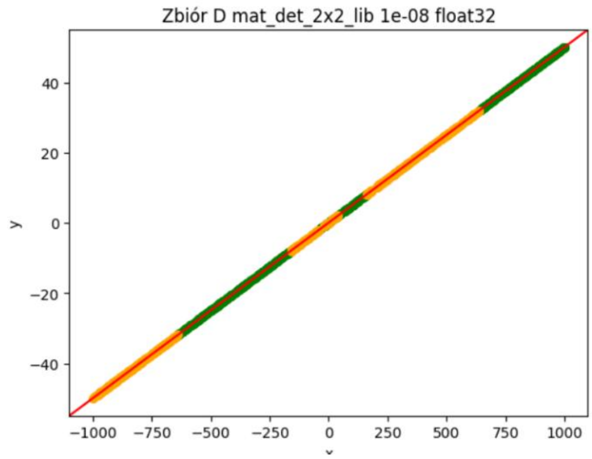
4.4 Zbiór D

W **Zbiorze D** występuje duża rozbieżność w kwalifikacji punktów dla różnych doborów parametrów testowych, mimo że teoretycznie każdy z punktów został wygenerowany na prostej. Wyniki przedstawione są na następnej stronie w Tabela 6.

Precyzja liczby zmiennoprzecinkowej znacząco wpływała na grupowanie danych. Typ ***float64*** znacznie częściej klasyfikował punkty jako leżące na ***prawo od prostej*** niż ***float32*** we wszystkich przypadkach testowych. Przykłady tego zjawiska przedstawiono na Wykresach 7 i 8.

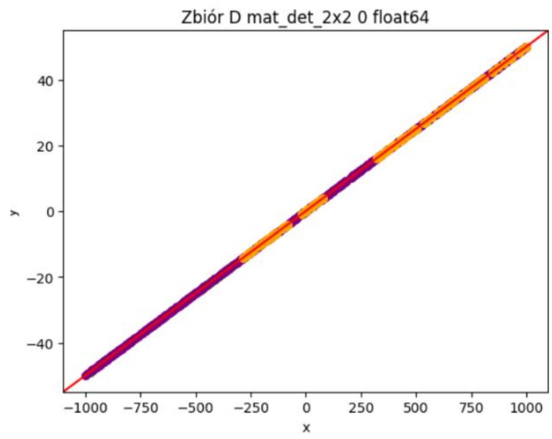


Wykres 7: Zbiór D mat_det_2x2_lib 1e-08 float 64`

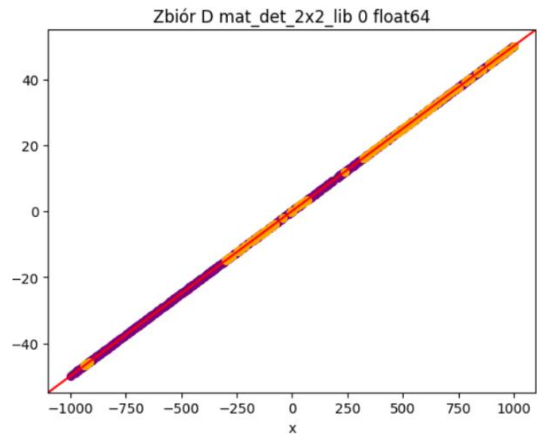


Wykres 8: Zbiór D mat_det_2x2_lib 1e-08 float 32

Kategoryzacja przynosiła zaskakujące wyniki, żadna z czterdziestu przypisań nie wykazała wszystkich punktów ***na prostej***. Żaden z wyników nie wykazał pozytywnego grupowania. Zdecydowanie najwięcej kategoryzacji punktów ***na prostej*** wystąpiła dla **epsilon = 0,** gdzie obliczenia wykazały do 707 (Wykres 9) i 701(Wykres 10) punktów ***na prostej***. Największe wyniki punktów ***na prostej*** występowały również dla wyznaczników z macierzy **mat_det_2x2** i **mat_det_2x2_lib**. **Wszystkie** kategoryzacje, w których wystąpiły punkty na prostej miały miejsce dla **epsilon = 0**. Zdecydowana większość kategoryzacji wykazała, że najwięcej punktów leży **na prawo** do prostej.



Wykres 9: Zbiór D mat_det_2x2 0 float 64



Wykres 10: Zbiór D mat_det_2x2_lib 0 float 64

float	wyznacznik	epsilon	Punkty_na_lewo	Punkty_na_linii	Punkty_na_prawo
float64	mat_det_3x3	0.0	120	560	320
float64	mat_det_3x3	1e-14	0	0	1000
float64	mat_det_3x3	1e-12	0	0	1000
float64	mat_det_3x3	1e-10	0	0	1000
float64	mat_det_3x3	1e-08	0	0	1000
float64	mat_det_2x2	0.0	158	707	135
float64	mat_det_2x2	1e-14	153	0	847
float64	mat_det_2x2	1e-12	91	0	909
float64	mat_det_2x2	1e-10	0	0	1000
float64	mat_det_2x2	1e-08	0	0	1000
float64	mat_det_3x3_lib	0.0	358	317	325
float64	mat_det_3x3_lib	1e-14	17	0	983
float64	mat_det_3x3_lib	1e-12	0	0	1000
float64	mat_det_3x3_lib	1e-10	0	0	1000
float64	mat_det_3x3_lib	1e-08	0	0	1000
float64	mat_det_2x2_lib	0.0	163	701	136
float64	mat_det_2x2_lib	1e-14	153	0	847
float64	mat_det_2x2_lib	1e-12	111	0	889
float64	mat_det_2x2_lib	1e-10	0	0	1000
float64	mat_det_2x2_lib	1e-08	0	0	1000
float32	mat_det_3x3	0.0	234	594	172
float32	mat_det_3x3	1e-14	234	0	766
float32	mat_det_3x3	1e-12	234	0	766
float32	mat_det_3x3	1e-10	234	0	766
float32	mat_det_3x3	1e-08	233	0	767
float32	mat_det_2x2	0.0	158	673	169
float32	mat_det_2x2	1e-14	158	0	842
float32	mat_det_2x2	1e-12	158	0	842
float32	mat_det_2x2	1e-10	158	0	842
float32	mat_det_2x2	1e-08	156	0	844
float32	mat_det_3x3_lib	0.0	469	51	480
float32	mat_det_3x3_lib	1e-14	407	0	593
float32	mat_det_3x3_lib	1e-12	397	0	603
float32	mat_det_3x3_lib	1e-10	397	0	603
float32	mat_det_3x3_lib	1e-08	393	0	607
float32	mat_det_2x2_lib	0.0	515	0	485
float32	mat_det_2x2_lib	1e-14	515	0	485
float32	mat_det_2x2_lib	1e-12	515	0	485
float32	mat_det_2x2_lib	1e-10	515	0	485

Tabela 6: Klasyfikacja punktów ze Zbioru D

5. Wnioski

Bazując na przeprowadzonym eksperymencie można wysnuć następujące wnioski:

- W przypadku ***Zbioru A*** i ***Zbioru C*** rozkład klasyfikacji punktów pozostawał stabilny, niezależnie od zastosowanych parametrów. Oznacza to, że generowanie punktów z losowego rozkładu na płaszczyźnie jest mniej wrażliwe na zmienne takie jak precyzja liczbowa czy tolerancja zerowa.
- Wyniki pokazały, że precyzja zmiennoprzecinkowa odgrywa kluczową rolę w dokładności klasyfikacji punktów względem linii. Typ ***float64*** zapewniał bardziej precyzyjne obliczenia, co przekładało się na spójne wyniki, nawet przy niskich wartościach tolerancji (***epsilon***). Typ ***float32*** okazał się bardziej podatny na błędy zaokrągleń, co szczególnie widoczne było w ***Zbiorze B***, gdzie liczba punktów klasyfikowanych jako leżące ***"na linii"*** była wyraźnie wyższa niż w przypadku ***float64,*** co pokazuje że mniejsza precyzja jest częściej skłonna do błędów.
- Zaimplementowane metody wyznacznika (***mat_det_2x2*** i ***mat_det_3x3)*** lepiej klasyfikowały punkty na prostej niż ich biblioteczne odpowiedniki dla mniejszych wartości epsilon w przypadku **Zbioru D.** Niemniej jednak nie były one wystarczająco precyzyjne dla pojedynczych punktów w ***Zbiorze B***. Implementacje biblioteczne okazały się mniej dokładne w rozpoznawaniu położenia punktów dokładnie na prostej w **Zbiorze D**, co sugeruje, że w zależności od sytuacji, ręcznie dopasowane algorytmy mogą oferować wyższą dokładność niż gotowe funkcje, i odwrotnie.
- Wyniki ćwiczenia pokazały, że dobór odpowiednich metod obliczeń, precyzji danych oraz wartości tolerancji jest kluczowy w geometrii obliczeniowej. Wyniki wskazują również na potrzebę ostrożności w przypadku pracy z dużymi wartościami współrzędnych i wykorzystania bibliotek numerycznych, przykładowo ***numpy***. Szczególnie ważne jest również stosowanie odpowiednio dobranej wartości tolerancji ***epsilon***, aby uniknąć błędów w klasyfikacji.
- Ćwiczenie pozwoliło na zgłębienie zagadnień związanych z numerycznymi aspektami geometrii obliczeniowej, a także pokazało wyzwania wynikające z niedokładności reprezentacji liczb zmiennoprzecinkowych, co ma istotne znaczenie przy pracy z dużymi zbiorami danych i dużymi zakresami wartości.