Algorytmy Geometryczne, Ćwiczenie 1 - Predykaty Geometryczne

Maciej Kmąk

Październik 2024

1 Cel ćwiczenia

Celem ćwiczenia było wdrożenie do podstawowych predykatów geometrycznych, graficzne przedstawienie zbiorów oraz analiza wyników w kontekście różnych metod obliczania wyznaczników macierzy, wartości tolerancji ε oraz różnych precyzji liczb zmiennoprzecinkowych. Ćwiczenie miało na celu zbadanie, w jaki sposób zmiana tych parametrów wpływa na klasyfikację punktów względem prostej wyznaczonej przez dwa punkty oraz ocenę wpływu precyzji obliczeń na ostateczny podział każdego ze zbiorów.

2 Dane techniczne

Program (w pliku „kmak_kod_1.ipynb") jest napisany w języku Python w środowisku Jupyter Notebook. Wykresy punktów powstały dzięki narzędziu przygotowanemu przez koło naukowe Bit, które korzysta z biblioteki matplotlib. Do generowania punktów została użyta funkcja z biblioteki NumPy - random.uniform(). Do obliczania wyznacznika z wykorzystaniem funkcji bibliotecznych użyto funkcję linalg.det() z biblioteki NumPy (nazwy tych funkcji to mat_det_3x3_lib oraz mat_det_2x2_lib). Zaimplementowano także funkcję bez użycia bibliotek korzystając, dla macierzy 3x3, z reguły Sarrusa oraz ze wzoru na obliczanie wyznacznika macierzy 2x2(nazwy tych funkcji to mat_det_3x3 oraz mat_det_2x2). Wyniki przedstawię dla pięciu tolerancji $\varepsilon = 0, 10^{-14}, 10^{-12}, 10^{-10}, 10^{-8}$, oraz dla dwóch precyzji liczb typu zmiennoprzecinkowego: float64 (domyślnie używaną w bibliotece NumPy) oraz float32 (konwersji dokonano również przy użyciu biblioteki NumPy). Przedstawiony program został wyegzekwowany na komputerze z procesorem Intel® Core™ i5-1235U z systemem operacyjnym Windows 11 Pro.

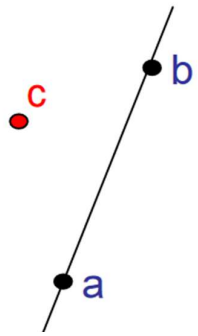
3 Wstęp teoretyczny

Sposobem, dzięki któremu możemy określić po której stronie prostej znajduje się punkt (Rysunek 3.1) jest obliczenie wartości wyznacznika macierzy 3×3:

$$\det(a,b,c) = \begin{vmatrix} a_x & a_y & 1 \\ b_x & b_y & 1 \\ c_x & c_y & 1 \end{vmatrix}$$

lub wyznacznika macierzy 2×2:

$$\det(a,b,c) = \begin{vmatrix} a_x - c_x & a_y - c_y \\ b_x - c_x & b_y - c_y \end{vmatrix}$$



Rysunek 3.1 Rysunek prostej poprowadzonej przez dwa punkty oraz trzeciego punktu

W zależności od wartości tego wyznacznika:

$$det(a,b,c) = \begin{cases} > 0 & - \quad punkt\ jest\ po\ lewej\ stronie\ prostej \\ < 0 & - \quad punkt\ jest\ po\ prawej\ stronie\ prostej \\ = 0 & - \quad punkt\ leży\ na\ prostej \end{cases}$$

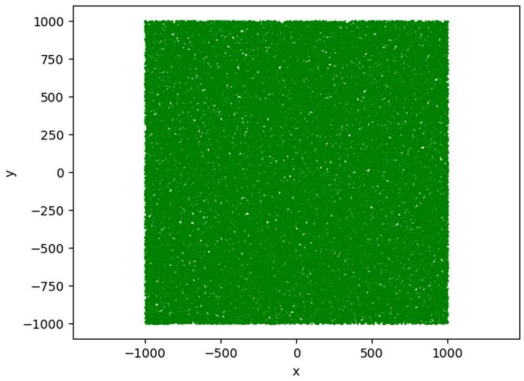
W rozwiązaniu będziemy używać różnych tolerancji ε jako wartości dla zera.

4 Przebieg doświadczenia

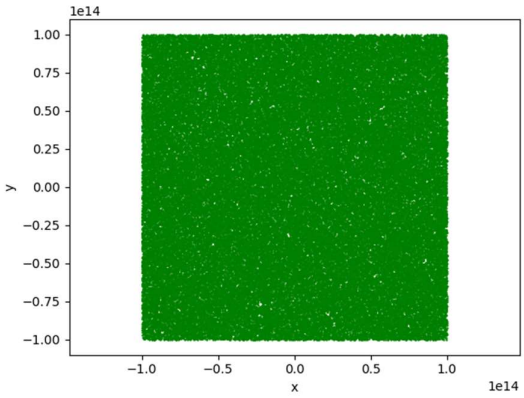
4.1 Generowane zbiory

A. 10^5 losowych punktów o współrzędnych z przedziału [−1000, 1000]
dalej zwany zbiorem A
B. 10^5 losowych punktów o współrzędnych z przedziału [-10^{14}, 10^{14}]
dalej zwany zbiorem B
C. 1000 losowych punktów leżących na okręgu o środku (0, 0) i promieniu R = 100
dalej zwany zbiorem C
D. 1000 losowych punktów o współrzędnych z przedziału [−1000, 1000] leżących na prostej wyznaczonej przez wektor a i b, dalej zwany zbiorem D

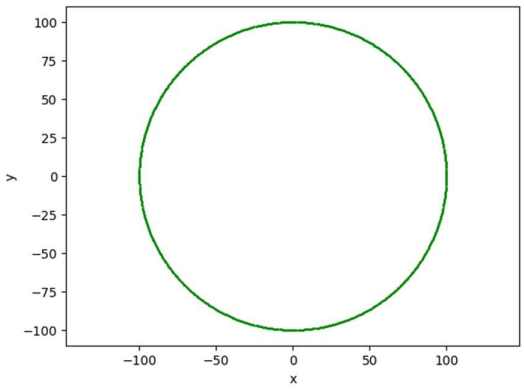
4.2 Wizualizacja zbiorów



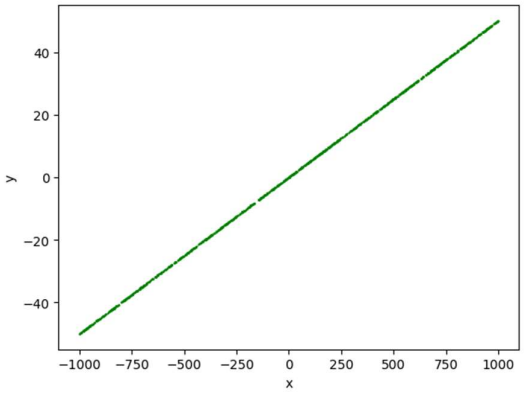
Wykres 4.2.1 Zbiór A



Wykres 4.2.1 Zbiór B



Wykres 4.2.3 Zbiór C



Wykres 4.2.4 Zbiór D

Teraz dla każdego zbioru przedstawię podziały wylosowanych punktów dotyczące położenia względem prostej przechodzącej przez punkty *a* i *b* gdzie *a* = (-1.0, 0.0) i *b* =(1.0, 0.1). Punkty różnią się kolorami ze względu na klasyfikację: na lewo od prostej - zielone, na prostej - fioletowe, na prawo - pomarańczowe.

5 Opracowanie danych

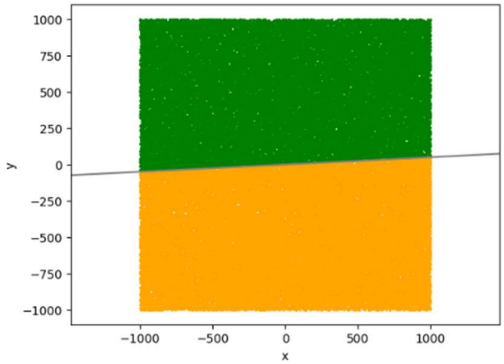
5.1 Zbiór A

W poniższej tabeli (Tabela 5.1.1) znajdują się wyniki działania programu dla zbioru A, wszystkich dostępnych dokładności float (64 i 32) oraz tolerancji dla zera (0, 10^{-14}, 10^{-12}, 10^{-10}, 10^{-8}).

Tabela 5.1.1 Wyniki działania programu dla zbioru A

Używana funkcja	Dokładność float64			Dokładność float32		
	Liczba punktów na lewo	Liczba punktów na prostej	Liczba punktów na prawo	Liczba punktów na lewo	Liczba punktów na prostej	Liczba punktów na prawo
	Tolerancja ε=0			Tolerancja ε=0		
mat_det_3x3	49802	0	50198	49802	0	50198
mat_det_3x3_lib	49802	0	50198	49802	0	50198
mat_det_2x2	49802	0	50198	49802	0	50198
mat_det_2x2_lib	49802	0	50198	49802	0	50198
	Tolerancja ε=10^{-14}			Tolerancja ε=10^{-14}		
mat_det_3x3	49802	0	50198	49802	0	50198
mat_det_3x3_lib	49802	0	50198	49802	0	50198
mat_det_2x2	49802	0	50198	49802	0	50198
mat_det_2x2_lib	49802	0	50198	49802	0	50198
	Tolerancja ε=10^{-12}			Tolerancja ε=10^{-12}		
mat_det_3x3	49802	0	50198	49802	0	50198
mat_det_3x3_lib	49802	0	50198	49802	0	50198
mat_det_2x2	49802	0	50198	49802	0	50198
mat_det_2x2_lib	49802	0	50198	49802	0	50198
	Tolerancja ε=10^{-10}			Tolerancja ε=10^{-10}		
mat_det_3x3	49802	0	50198	49802	0	50198
mat_det_3x3_lib	49802	0	50198	49802	0	50198
mat_det_2x2	49802	0	50198	49802	0	50198
mat_det_2x2_lib	49802	0	50198	49802	0	50198
	Tolerancja ε=10^{-8}			Tolerancja ε=10^{-8}		
mat_det_3x3	49802	0	50198	49802	0	50198
mat_det_3x3_lib	49802	0	50198	49802	0	50198
mat_det_2x2	49802	0	50198	49802	0	50198
mat_det_2x2_lib	49802	0	50198	49802	0	50198

Dla zbioru A możemy zauważyć, że zarówno precyzja liczb zmiennoprzecinkowych jak i tolerancja zera (ε) nie ma wpływu na klasyfikację punktów do poszczególnych podzbiorów. Niezależnie od sporej liczby punktów (10^5), żaden nie został określony jako leżący na prostej. Graficzne przedstawienie wyników wygląda następująco (Wykres 5.1.1):



Wykres 5.1.1. Wykres zbioru A dla każdej precyzji, tolerancji i funkcji.

5.2 Zbiór B

W poniższej tabeli (Tabela 5.2.1) znajdują się wyniki działania programu dla zbioru B, wszystkich dostępnych dokładności float (64 i 32) oraz tolerancji dla zera (0, 10^{-14}, 10^{-12}, 10^{-10}, 10^{-8}).

Tabela 5.2.1 Wyniki działania programu dla zbioru A

Używana funkcja	Dokładność float64			Dokładność float32		
	Liczba punktów na lewo	Liczba punktów na prostej	Liczba punktów na prawo	Liczba punktów na lewo	Liczba punktów na prostej	Liczba punktów na prawo
	Tolerancja ε=0			Tolerancja ε=0		
mat_det_3x3	49836	0	50164	49836	0	50164
mat_det_3x3_lib	49836	0	50164	49836	0	50164
mat_det_2x2	49835	2	50163	0	100000	0
mat_det_2x2_lib	49834	4	50162	6652	86628	6720
	Tolerancja ε=10^{-14}			Tolerancja ε=10^{-14}		
mat_det_3x3	49836	0	50164	49836	0	50164
mat_det_3x3_lib	49836	0	50164	49836	0	50164
mat_det_2x2	49835	2	50163	0	100000	0
mat_det_2x2_lib	49834	4	50162	6652	86628	6720
	Tolerancja ε=10^{-12}			Tolerancja ε=10^{-12}		
mat_det_3x3	49836	0	50164	49836	0	50164
mat_det_3x3_lib	49836	0	50164	49836	0	50164
mat_det_2x2	49835	2	50163	0	100000	0
mat_det_2x2_lib	49834	4	50162	6652	86628	6720
	Tolerancja ε=10^{-10}			Tolerancja ε=10^{-10}		
mat_det_3x3	49836	0	50164	49836	0	50164
mat_det_3x3_lib	49836	0	50164	49836	0	50164
mat_det_2x2	49835	2	50163	0	100000	0
mat_det_2x2_lib	49834	4	50162	6652	86628	6720
	Tolerancja ε=10^{-8}			Tolerancja ε=10^{-8}		
mat_det_3x3	49836	0	50164	49836	0	50164
mat_det_3x3_lib	49836	0	50164	49836	0	50164
mat_det_2x2	49835	2	50163	0	100000	0
mat_det_2x2_lib	49834	4	50162	6652	86628	6720

Wyniki dla zbioru B są bardziej rozbieżne i możemy zauważyć różne wyniki w zależności od precyzji liczb zmiennoprzecinkowych – w tych grupach wyniki są już takie same dla ustalonej funkcji obliczającej wartość wyznacznika. Zauważmy natomiast, że dla ustalonego ε wyniki różnią się.

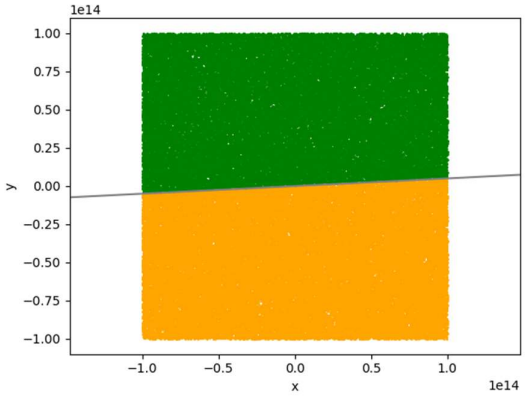
Dla liczb typu float64 i funkcji liczących wyznacznik 3×3 (zarówno implementowanej samodzielnie jak i bibliotecznie) żaden punkt nie został zakwalifikowany jako leżący na prostej (Wykres 5.2.3). W przypadku obydwu funkcji liczących wyznacznik 2×2 niektóre z punktów zostały zakwalifikowane jako leżące na prostej. Są to:
W przypadku funkcji 2x2: [(-97960596026523.36, -4885744900974.625),
(-86922985351721.36, -4353852952769.9062)].
Natomiast w przypadku funkcji bibliotecznej 2x2_lib:
[(-97960596026523.36, -4885744900974.625), (-88326024737460.94, -4410331996867.9375),
(-86922985351721.36, -4353852952769.9062), (93696284131929.0, 4674951251163.5)]
Po dokonaniu obliczeń arytmetycznych liczby te w rzeczywistości na prostej nie leżą. Widzimy natomiast, że zbiór punktów 2x2 zawiera się w zbiorze 2x2_lib.

Dla liczb typu float32 widzimy, że funkcje 3x3 w każdym przypadku dostarczyły nam tym samych wyników co dla float64. Jednak w przypadku obliczania wyznacznika macierzy 2×2 możemy zauważyć znaczne zmiany w klasyfikacji punktów. Dla bibliotecznej implementacji 2x2_lib znaczna część punktów została błędnie zakwalifikowana jako leżąca na prostej, było to około 86,6%. (Dla porównania wykresy: 5.2.1 i 5.2.2). Dla implementacji własnej liczenia wyznacznika 2×2, dla każdej tolerancji zera wszystkie punkty zostały błędnie zakwalifikowane jako leżące na prostej (Wykres 5.2.4). Skąd może wynikać aż tak duży błąd?

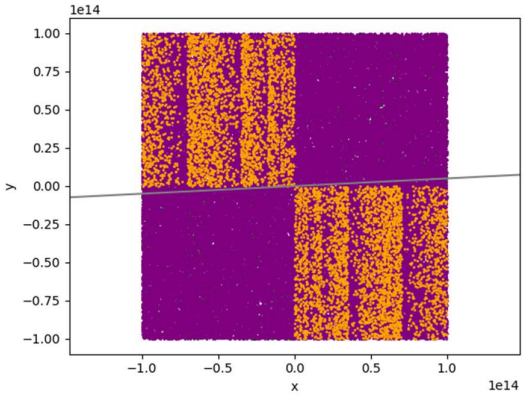
Fałszywe określenie tych punktów jako leżących na prostej może wynikać z błędów wynikających z operacji na liczbach zmiennoprzecinkowych (ang. *floating-point arithmetic*). Operacje na liczbach zmiennoprzecinkowych są podatne na niedokładności, ponieważ komputer reprezentuje liczby rzeczywiste w sposób przybliżony. Przy tak dużych liczbach, jak te podane w przykładzie, błędy zaokrągleń mogą być na tyle duże, że wpłyną na wynik testu kolinearności punktów. Wartości rzędu 10^{13} jak w przykładzie zbioru B znacznie przekraczają precyzję float32, więc niemal na pewno doszło do utraty precyzji (dla porównania Wykres 5.2.1 i Wykres 5.2.2).

Podsumowując, macierz 3x3 jest bardziej odporna na te błędy, ponieważ obliczenia są bardziej rozłożone, co zmniejsza wpływ jednorazowego błędu zaokrągleń na wynik.

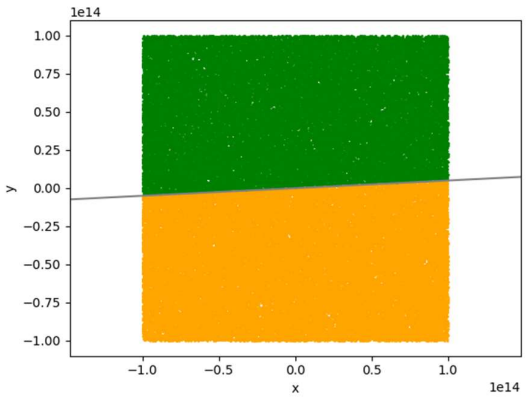
Poniżej znajdują się wybrane ilustracje przedstawiające wykresy klasyfikacji punktów zbioru B w zależności od wybranych parametrów.



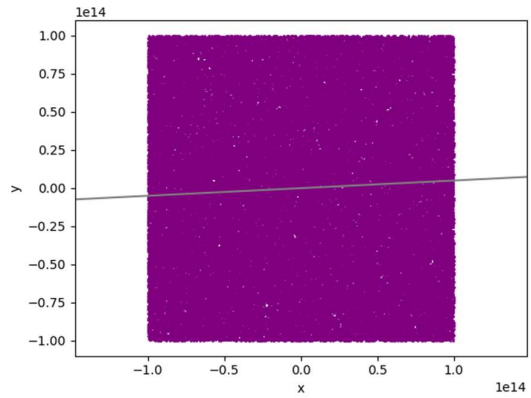
Wykres 5.2.1 Zbiór B dla precyzji float64, tolerancji ε=0 oraz funkcji mat_det_2x2_lib



Wykres 5.2.2 Zbiór B dla precyzji float32, tolerancji ε=0 oraz funkcji mat_det_2x2_lib



Wykres 5.2.3 Zbiór B dla precyzji float64, tolerancji $\varepsilon=10^{-10}$ oraz funkcji mat_det_3x3



Wykres 5.2.4 Zbiór B dla precyzji float32, tolerancji $\varepsilon=10^{-12}$ oraz funkcji mat_det_2x2

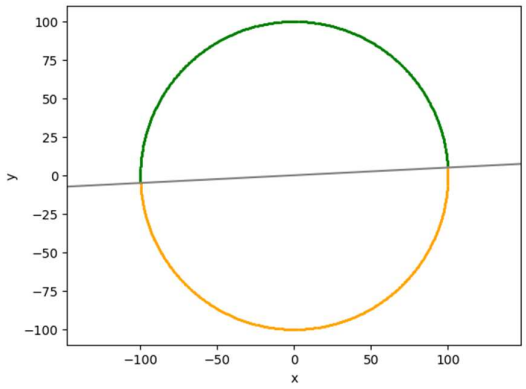
5.3 Zbiór C

W poniższej tabeli (Tabela 5.3.1) znajdują się wyniki działania programu dla zbioru C, wszystkich dostępnych dokładności float (64 i 32) oraz tolerancji dla zera (0, 10^{-14}, 10^{-12}, 10^{-10}, 10^{-8}).

Tabela 5.3.1 Wyniki działania programu dla zbioru C

Używana funkcja	Dokładność float64			Dokładność float32		
	Liczba punktów na lewo	Liczba punktów na prostej	Liczba punktów na prawo	Liczba punktów na lewo	Liczba punktów na prostej	Liczba punktów na prawo
	Tolerancja ε=0			Tolerancja ε=0		
mat_det_3x3	499	0	501	499	0	501
mat_det_3x3_lib	499	0	501	499	0	501
mat_det_2x2	499	0	501	499	0	501
mat_det_2x2_lib	499	0	501	499	0	501
	Tolerancja $\varepsilon=10^{-14}$			Tolerancja $\varepsilon=10^{-14}$		
mat_det_3x3	499	0	501	499	0	501
mat_det_3x3_lib	499	0	501	499	0	501
mat_det_2x2	499	0	501	499	0	501
mat_det_2x2_lib	499	0	501	499	0	501
	Tolerancja $\varepsilon=10^{-12}$			Tolerancja $\varepsilon=10^{-12}$		
mat_det_3x3	499	0	501	499	0	501
mat_det_3x3_lib	499	0	501	499	0	501
mat_det_2x2	499	0	501	499	0	501
mat_det_2x2_lib	499	0	501	499	0	501
	Tolerancja $\varepsilon=10^{-10}$			Tolerancja $\varepsilon=10^{-10}$		
mat_det_3x3	499	0	501	499	0	501
mat_det_3x3_lib	499	0	501	499	0	501
mat_det_2x2	499	0	501	499	0	501
mat_det_2x2_lib	499	0	501	499	0	501
	Tolerancja $\varepsilon=10^{-8}$			Tolerancja $\varepsilon=10^{-8}$		
mat_det_3x3	499	0	501	499	0	501
mat_det_3x3_lib	499	0	501	499	0	501
mat_det_2x2	499	0	501	499	0	501
mat_det_2x2_lib	499	0	501	499	0	501

Dla zbioru C możemy zauważyć, podobną sytuację jak dla zbioru A: zarówno precyzja liczb zmiennoprzecinkowych jak i tolerancja zera ε nie ma wpływu na klasyfikację punktów do poszczególnych podzbiorów. Ze względu na specyfikę tego zbioru – jego małą liczność i niskie prawdopodobieństwo bliskości punktów do prostej – zmiana metody obliczeń nie wpływa na końcową klasyfikację punktów. W efekcie różnice w wynikach między metodami są nieistotne. Graficzne przedstawienie wyników wygląda następująco (Wykres 5.3.1):



Wykres 5.3.1. Wykres zbioru C dla każdej precyzji, tolerancji i funkcji.

5.4 Zbiór D

W poniższej tabeli (Tabela 5.4.1) znajdują się wyniki działania programu dla zbioru D, wszystkich dostępnych dokładności float (64 i 32) oraz tolerancji dla zera (0, 10^{-14}, 10^{-12}, 10^{-10}, 10^{-8}).

Tabela 5.4.1 Wyniki działania programu dla zbioru D

Używana funkcja	Dokładność float64			Dokładność float32		
	Liczba punktów na lewo	Liczba punktów na prostej	Liczba punktów na prawo	Liczba punktów na lewo	Liczba punktów na prostej	Liczba punktów na prawo
	Tolerancja ε=0			Tolerancja ε=0		
mat_det_3x3	287	331	382	292	396	312
mat_det_3x3_lib	371	340	289	479	57	464
mat_det_2x2	161	690	149	171	658	171
mat_det_2x2_lib	155	710	135	496	0	504
	Tolerancja $\varepsilon=10^{-14}$			Tolerancja $\varepsilon=10^{-14}$		
mat_det_3x3	0	1000	0	292	396	312
mat_det_3x3_lib	37	913	50	421	158	421
mat_det_2x2	149	709	142	171	658	171
mat_det_2x2_lib	145	729	126	496	0	504
	Tolerancja $\varepsilon=10^{-12}$			Tolerancja $\varepsilon=10^{-12}$		
mat_det_3x3	0	1000	0	292	396	312
mat_det_3x3_lib	0	1000	0	414	174	412
mat_det_2x2	79	829	92	171	658	171
mat_det_2x2_lib	98	804	98	496	0	504
	Tolerancja $\varepsilon=10^{-10}$			Tolerancja $\varepsilon=10^{-10}$		
mat_det_3x3	0	1000	0	292	396	312
mat_det_3x3_lib	0	1000	0	414	174	412
mat_det_2x2	0	1000	0	171	658	171
mat_det_2x2_lib	0	1000	0	496	0	504
	Tolerancja $\varepsilon=10^{-8}$			Tolerancja $\varepsilon=10^{-8}$		
mat_det_3x3	0	1000	0	291	398	311
mat_det_3x3_lib	0	1000	0	413	179	408
mat_det_2x2	0	1000	0	171	661	168
mat_det_2x2_lib	0	1000	0	494	6	500

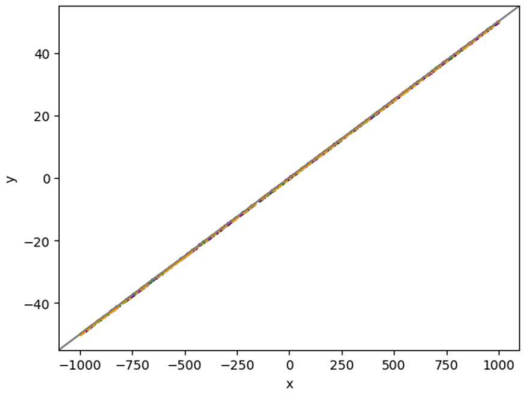
Dla zbioru D czyli punktów leżących dokładnie na zadanej prostej możemy zauważyć, że wyniki w przypadku różnych parametrów znacznie odbiegają od oczekiwanych.

Dla liczb o precyzji float64 stosunkowo lepiej poradziły sobie funkcje liczące wyznaczniki macierzy 3×3. Jednak dla bardzo restrykcyjnej tolerancji (ε=0) żadna z funkcji nie zakwalifikowała wszystkich punktów jako leżących na prostej (Wykres 5.4.1). Dla małej tolerancji ($\varepsilon=10^{-14}$) jedynie funkcja mat_det_3x3 określiła, że wszystkie punkty leżą na prostej (Wykres 5.4.2). Dla większej tolerancji ($\varepsilon=10^{-10} \vee \varepsilon=10^{-8}$) wszystkie proponowane funkcje poprawnie zakwalifikowały punkty jako leżące na prostej (Wykres 5.4.4).

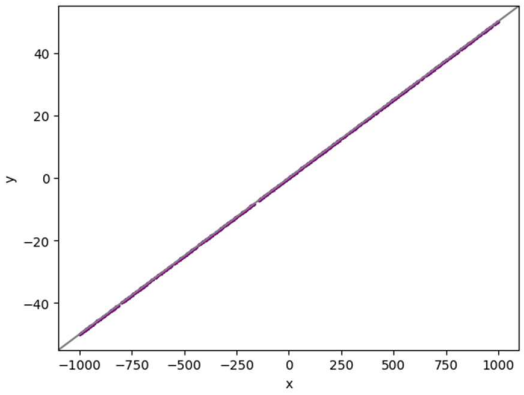
Dokładność float32 znacząco ogranicza precyzję obliczeń, szczególnie dla funkcji bibliotecznych obliczających wyznacznik macierzy 2x2_lib (Wykres 5.4.3). Poprawne przypisanie punktów do kategorii nie było jednak możliwe nawet przy dużej tolerancji dla zera(Wykres 5.4.5). Kluczowe znaczenie miała precyzja – wyznaczniki 3×3, nawet przy mniejszej tolerancji, klasyfikowały punkty bezbłędnie, podczas gdy wyznaczniki 2×2 były mniej dokładne(Wykresy 5.4.2 i 5.4.6).

Format float32 jest wyraźnie mniej precyzyjny i bardziej podatny na błędy klasyfikacji szczególnie widoczne jest to w przypadku funkcji bibliotecznych oraz przy niższych wartościach tolerancji. Ograniczona precyzja float32 powoduje, że nawet przy większej tolerancji klasyfikacja punktów na prostej jest mniej dokładna, a liczba punktów zaklasyfikowanych poza prostą jest wyraźnie wyższa.

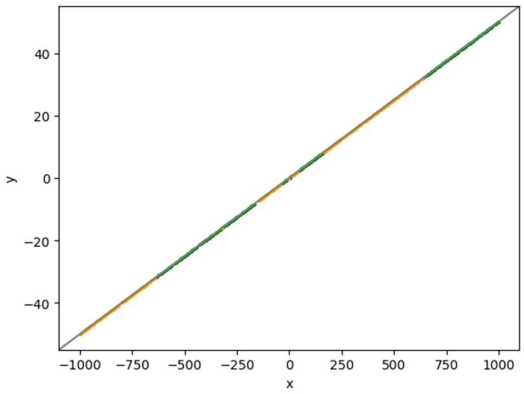
Poniżej znajdują się wybrane ilustracje przedstawiające wykresy klasyfikacji punktów zbioru D w zależności od wybranych parametrów.



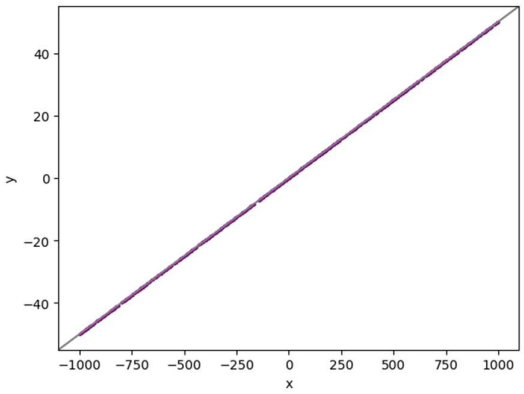
Wykres 5.4.1 Zbiór D dla precyzji float64, tolerancji $\varepsilon=0$ oraz funkcji mat_det_3x3



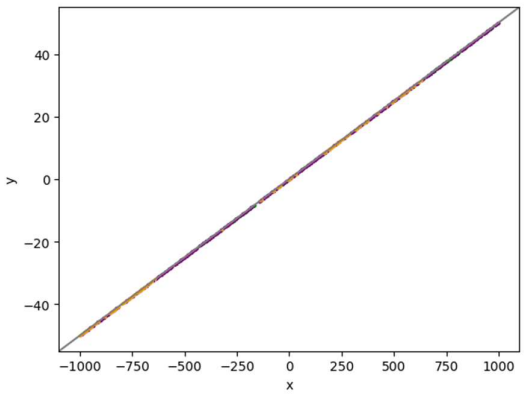
Wykres 5.4.2 Zbiór D dla precyzji float64, tolerancji $\varepsilon=10^{-14}$ oraz funkcji mat_det_3x3



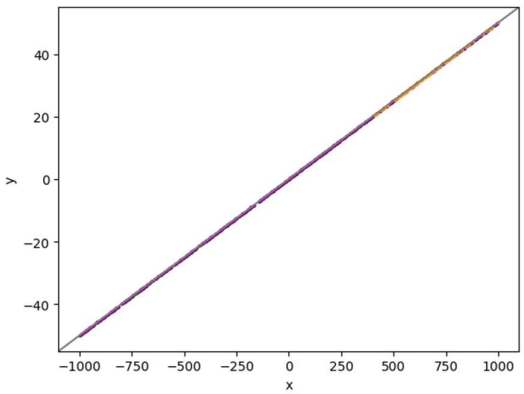
Wykres 5.4.3 Zbiór D dla precyzji float32, tolerancji $\varepsilon=10^{-8}$ oraz funkcji mat_det_2x2_lib



Wykres 5.4.4 Zbiór D dla precyzji float64, tolerancji $\varepsilon=10^{-8}$ oraz funkcji mat_det_2x2_lib



Wykres 5.4.5 Zbiór D dla precyzji float32, tolerancji $\varepsilon=10^{-8}$ oraz funkcji mat_det_2x2



Wykres 5.4.6 Zbiór D dla precyzji float64, tolerancji $\varepsilon=10^{-12}$ oraz funkcji mat_det_2x2_lib

6 Podsumowanie zagadnienia

Ćwiczenie to jest szczegółową analizą działania predykatów geometrycznych dla różnych zbiorów punktów oraz wpływu różnych parametrów obliczeniowych, takich jak dokładność liczb zmiennoprzecinkowych oraz tolerancja ε.

6.1 Metoda obliczania wyznacznika.

Z przeprowadzonej analizy wynika, że metoda obliczania wyznacznika macierzy 3×3 okazała się najskuteczniejsza i najdokładniejsza spośród wszystkich czterech rozważanych metod obliczeniowych. Wyznacznik macierzy 3×3 okazał się bardziej odporny na błędy numeryczne niż macierz 2×2, co można tłumaczyć większą liczbą operacji, które rozpraszają błędy. Obliczenia na macierzy 3×3 są bardziej stabilne w kontekście analizy geometrycznej, gdyż większa liczba operacji arytmetycznych zwiększa dokładność wyniku.

6.2 Tolerancja ε.

W przypadku punktów, które leżą bardzo blisko analizowanej prostej lub wręcz na niej, rola tolerancji ε staje się kluczowa. Ustawienie odpowiedniej wartości ε pozwala na właściwe uwzględnienie błędów numerycznych, które wynikają z ograniczonej precyzji obliczeń zmiennoprzecinkowych (szczególnie przy float32). Dobrze ilustruje to czwarty zbiór punktów(Zbiór D), w którym przy większym ε równym 10^{-8} lub 10^{-10} oraz precyzji float64, wszystkie punkty zostały sklasyfikowane jako znajdujące się na prostej, niezależnie od wybranej funkcji obliczającej wartość wyznacznika. Natomiast przy ε wynoszącym 0, w zależności od wybranej funkcji tylko 30-70% punktów zostało przypisanych jako leżących na prostej.

6.3 Precyzja liczb zmiennoprzecinkowych.

Precyzja ma kluczowe znaczenie w kontekście obliczeń numerycznych, szczególnie w przypadkach, gdzie dokładność jest niezbędna do prawidłowej klasyfikacji punktów w analizach geometrycznych. W przypadku zbiorów B i D, obserwacje dotyczące float64 i float32 ujawniają istotne różnice w stabilności wyników, które mają wpływ na skuteczność algorytmów. Zastosowanie wyższej precyzji (float64) jest niezbędne, gdy dane zawierają duże współrzędne lub wymagają wysokiej dokładności. Lepsza precyzja pozwala na uzyskanie stabilniejszych wyników, minimalizując wpływ błędów zaokrągleń, zapewnia znacznie większą dokładność, co w rezultacie prowadzi do lepszej klasyfikacji, zwłaszcza w sytuacjach, gdzie punkty są blisko prostej.

7 Wnioski

To ćwiczenie podkreśla, jak istotne są odpowiednie wybory parametrów obliczeniowych w analizach geometrycznych. W zależności od charakterystyki danych, takich jak odległości między punktami czy wartości współrzędnych, należy dostosować precyzję liczb zmiennoprzecinkowych i metodę obliczeniową. Stosowanie precyzji float64, odpowiedniej tolerancji ε i wyznaczników macierzy 3x3 okazuje się korzystne dla większych i bardziej złożonych danych, co pozwala uzyskać stabilniejsze i dokładniejsze wyniki.

Graficzne przedstawienie wszystkich możliwych kombinacji danych, tolerancji i precyzji można znaleźć w dołączonym pliku z kodem „kmak_kod_1.ipynb”. W sprawozdaniu zostały zawarte analizy oraz wnioski, które pomogą w zrozumieniu wyników uzyskanych z tych wizualizacji. Przeanalizowane zostały różne aspekty, takie jak wpływ tolerancji na wyniki oraz różnice w precyzji przy różnych zestawach danych.