

Algorytmy geometryczne – laboratorium 1.

Predykaty Geometryczne

1. Dane techniczne:

- System operacyjny: Windows 11 Pro (x64)
- Procesor: AMD Ryzen 5 7530U (2.0 – 4.5 GHz)
- Pamięć RAM: 16GB (2133 MHz)
- Środowisko: Jupyter Notebook
- Język programowania: Python 3.13.7

Podczas przeprowadzenia doświadczenia użyto narzędzia do wizualizacji danych autorstwa Koła Naukowego BIT oraz bibliotek *numpy*, *pandas*, *matplotlib*. By zagwarantować powtarzalność generowanych danych użyto polecenia `np.random.seed(42)`. By uniknąć sytuacji, w której ważne informacje zostaną zakryte na wykresie, wprowadzono modyfikacje w funkcji obsługującej jego generowanie. Kategorie punktów nakładane były na wykres w kolejności od najbardziej licznych do najmniej licznych, co gwarantuje minimalizację wystąpień przysłonięcia jego detali.

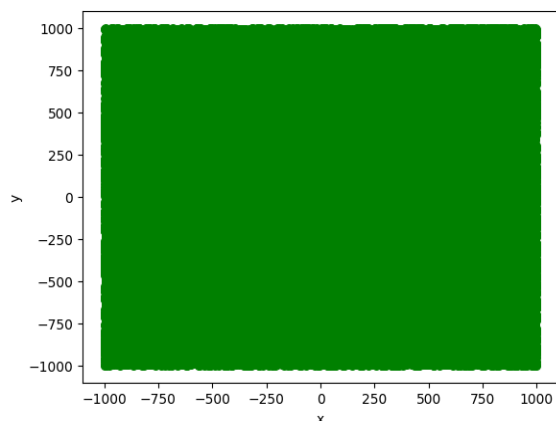
2. Cel ćwiczenia

- Wdrożenie do predykatów geometrycznych.
- Analiza wyników na podstawie różnych metod obliczania wyznaczników, współczynnika tolerancji ϵ precyzji liczb zmiennoprzecinkowych oraz ich wpływu na końcową klasyfikację punktów z wygenerowanych zbiorów.
- Ocena metod obliczania wyznaczników

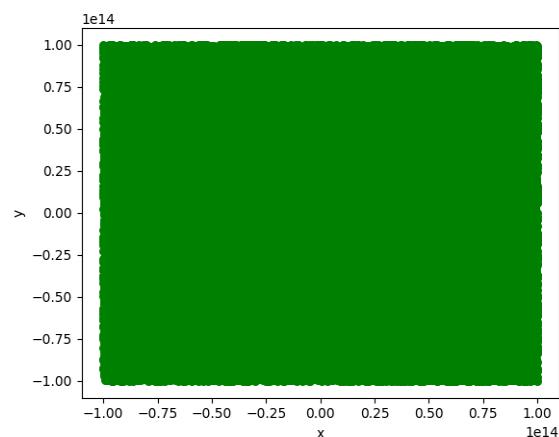
3. Przeprowadzenie doświadczenia

3.1. Przygotowanie zbiorów punktów

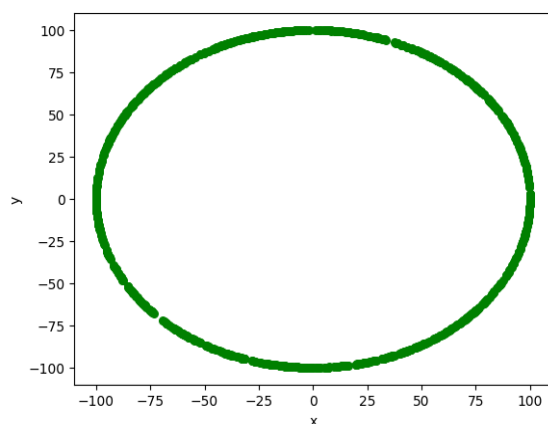
Na początku ćwiczenia wygenerowano 4 zbiory punktów typu `double` (korzystając z funkcji biblioteki *numpy* - `numpy.random.uniform`) należących do dwuwymiarowej przestrzeni (wykresy nr 1, 2, 3 oraz 4).



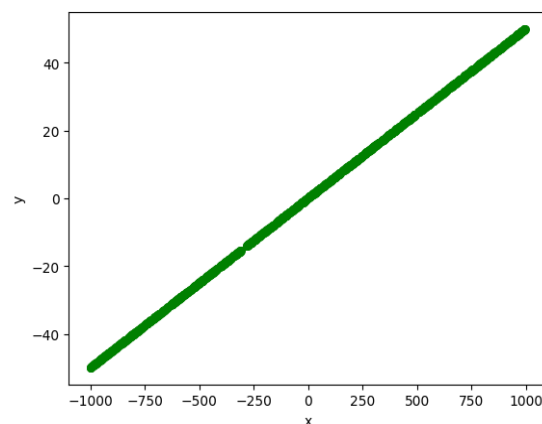
Wykres nr 1: Zbiór A - 10^5 losowych punktów w przestrzeni 2D o współrzędnych z przedziału $x, y \in [-1000, 1000]$



Wykres nr 2: Zbiór B - 10^5 losowych punktów w przestrzeni 2D o współrzędnych z przedziału $x, y \in [-10^{14}, 10^{14}]$



Wykres nr 3: Zbiór C - 1000 losowych punktów w przestrzeni 2D leżących na okręgu o środku $O = (0,0)$ i promieniu $R = 100$



Wykres nr 4: Zbiór D - 1000 losowych punktów w przestrzeni 2D o współrzędnej z przedziału $x \in [-1000, 1000]$ leżących na prostej wyznaczonej przez wektor \vec{ab} , gdzie $a = (-1.0, 0.0)$ oraz $b = (1.0, 0.1)$

3.2. Metody wyznaczania położenia punktu względem prostej

Do określenia położenia punktu C o współrzędnych (C_x, C_y) względem prostej wyznaczonej przez wektor \vec{AB} , gdzie $A = (A_x, A_y)$ oraz $B = (B_x, B_y)$ wykorzystano metody związane z obliczaniem wyznacznika:

macierzy 2x2

$$\det(A, B, C) = \det \begin{bmatrix} A_x - C_x & A_y - C_y \\ B_x - C_x & B_y - C_y \end{bmatrix}$$

macierzy 3x3

$$\det(A, B, C) = \det \begin{bmatrix} A_x & A_y & 1 \\ B_x & B_y & 1 \\ C_x & C_y & 1 \end{bmatrix}$$

$\det(A, B, C) < 0$ \Rightarrow punkt C znajduje się na prawo od prostej AB (punkty A, B, C są ustawione zgodnie z kierunkiem ruchu wskazówek zegara),

$\det(A, B, C) = 0$ \Rightarrow punkt C znajduje się na prostej AB,

$\det(A, B, C) > 0$ \Rightarrow punkt C znajduje się na lewo od prostej AB (punkty A, B, C są ustawione przeciwnie do kierunku ruchu wskazówek zegara),

Wartości wyznaczników obliczone zostały przy użyciu samodzielnie napisanych funkcji oraz funkcji bibliotecznych `numpy.linalg.det()`.

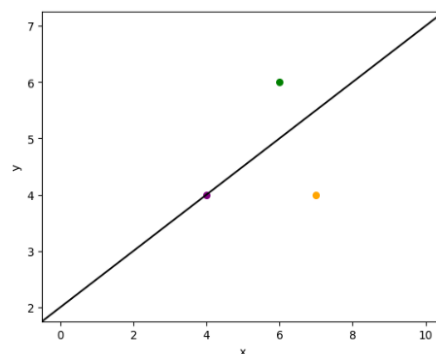
3.3. Parametry i legenda

Na wykresach punkty oznaczano w następujący sposób (wykres nr 5):

- na lewo od prostej – kolorem zielonym,
- na prostej – kolorem fioletowym,
- na prawo od prostej – kolorem żółtym.

Punkty klasyfikowano względem prostej wykorzystując 2 zakresy precyzji liczb zmiennoprzecinkowych (float64 i float32) oraz różnych wartości współczynnika tolerancji ε .

$$\varepsilon \in [0, 10^{-12}, 10^{-10}, 10^{-8}, 10^{-6}]$$



Wykres nr 5: Przykład klasyfikacji punktów względem prostej

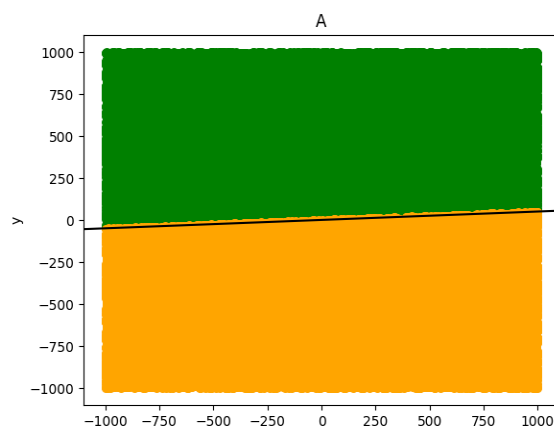
4. Opracowanie wyników

4.1. Zbiór punktów A

Precyzja	Współczynnik tolerancji ε	Funkcja obliczania wyznacznika	Liczba punktów na lewo od prostej	Liczba punktów na prostej	Liczba punktów na prawo od prostej
float 32-bit float 64-bit	0 10^{-12} 10^{-10} 10^{-8} 10^{-6}	mat_det_2x2 mat_det_2x2_lib mat_det_3x3 mat_det_3x3_lib	50187	0	49813

Tabela nr 1: wyniki klasyfikacji punktów ze zbioru A względem prostej

Wyniki klasyfikacji punktów ze zbioru A (Wykres nr 6) względem prostej były poprawne, identyczne, niezależnie od użytej precyzji liczb zmiennoprzecinkowych, współczynnika tolerancji ε oraz funkcji obliczania wyznacznika. Przetawiono je w tabeli nr 1.

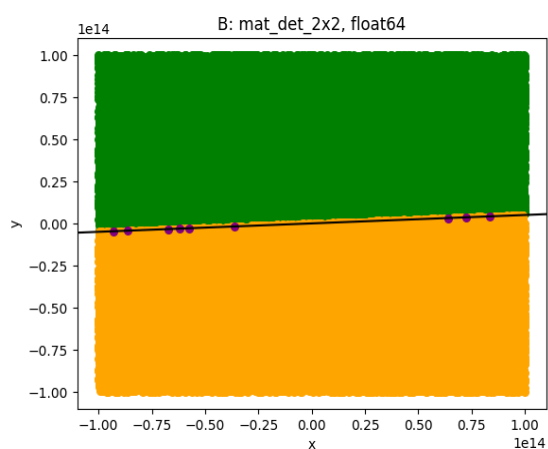


Wykres nr 6: Wynik poprawnej klasyfikacji punktów ze zbioru A względem prostej

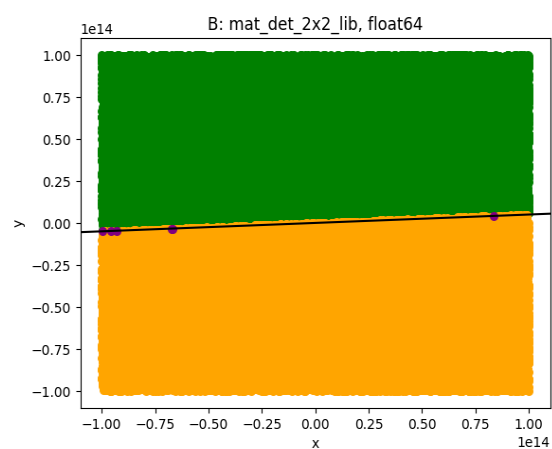
4.1. Zbiór punktów B

Precyzja	Współczynnik tolerancji ϵ	Funkcja obliczania wyznacznika	Liczba punktów na lewo od prostej	Liczba punktów na prostej	Liczba punktów na prawo od prostej
float 64-bit	0	mat_det_2x2	50106	9	49885
	10^{-12}	mat_det_2x2_lib	50109	6	49885
	10^{-10}				
	10^{-8}	mat_det_3x3	50110	0	49890
	10^{-6}	mat_det_3x3_lib			
float 32-bit	0	mat_det_2x2	0	100000	0
	10^{-12}	mat_det_2x2_lib	6756	86543	6701
	10^{-10}				
	10^{-8}	mat_det_3x3	50110	0	49890
	10^{-6}	mat_det_3x3_lib			

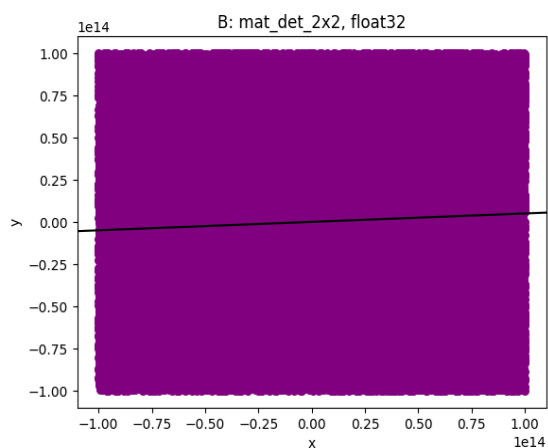
Tabela nr 2: wyniki klasyfikacji punktów ze zbioru B względem prostej



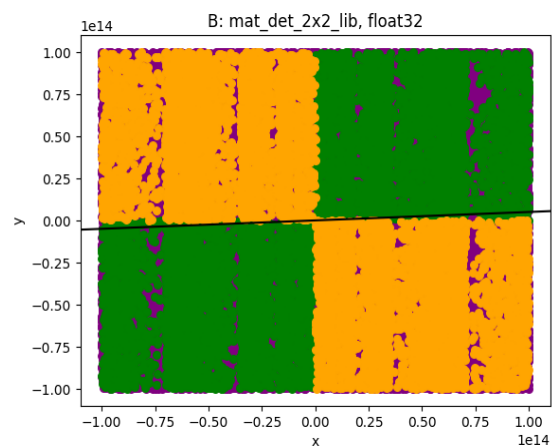
Wykres 7: Wynik klasyfikacji punktów ze zbioru B względem prostej przy użyciu precyzji float 64-bit oraz funkcji `mat_det_2x2`



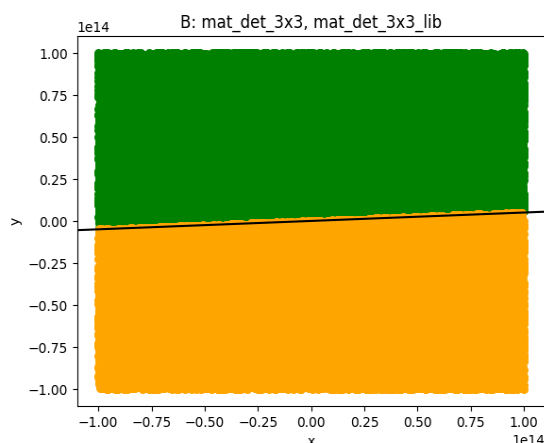
Wykres 8: Wynik klasyfikacji punktów ze zbioru B względem prostej przy użyciu precyzji float 64-bit oraz funkcji `mat_det_2x2_lib`



Wykres 9: Wynik klasyfikacji punktów ze zbioru B względem prostej przy użyciu precyzji float 32-bit oraz funkcji `mat_det_2x2`



Wykres 10: Wynik klasyfikacji punktów ze zbioru B względem prostej przy użyciu precyzji float 32-bit oraz funkcji `mat_det_2x2_lib`



Wykres 11: Wynik poprawnej klasyfikacji punktów ze zbioru B względem prostej przy użyciu funkcji *mat_det_3x3* oraz *mat_det_3x3_lib*

Wyniki uzyskane dla drugiego zestawu danych różniły się od wyników danych ze zbioru A. Przedstawiono je w tabeli nr 2.

Dla precyzji liczb zmiennoprzecinkowych float 64-bit funkcja *mat_det_2x2* oraz *mat_det_2x2_lib* zakwalifikowały kilka (9 oraz 6) punktów jako leżące na prostej (Wykres nr 7 oraz 8).

Punkty zakwalifikowane, jako leżące na prostej, zostały:

W przypadku funkcji *mat_det_2x2*

(72443481321400.9, 3621267469103.3594)
 (-61993095980030.35, -3101246309216.2188)
 (83586675577520.0, 4184026316262.7344)
 (-86121550566616.34, -4297733967149.703)
 (-57374726486293.984, -2871233091784.8594)
 (-67415056808338.68, -3371795403953.9375)
 (-36407833871820.83, -1821741753849.9062)
 (63853289530308.81, 3207340519930.547)
 (-92950665609956.84, -4640211893473.547)

W przypadku funkcji *mat_det_2x2_lib*

(83586675577520.0, 4184026316262.7344)
 (-95550062462810.5, -4724811809189.469)
 (-67016182911087.95, -3369120095237.125)
 (-67415056808338.68, -3371795403953.9375)
 (-92950665609956.84, -4640211893473.547)
 (-99569688543491.39, -4979987395579.781)

Dla precyzji liczb zmiennoprzecinkowych float 32-bit oraz funkcji *mat_det_2x2* i *mat_det_2x2_lib* wystąpiły błędy w wyznaczaniu położenia punktów względem prostej (wykres nr 9 oraz 10). Własnoręcznie pisana funkcja wyznacznika 2x2 zakwalifikowała każdy z punktów jako leżący na prostej – wartość wyznacznika wyniosła w każdym przypadku 0 (wykresy 9 oraz 10). Funkcja biblioteczna zakwalifikowała niektóre punkty należące do pierwszej i trzeciej ćwiartki układu współrzędnych jako leżące po lewej stronie prostej, a punkty należące do drugiej i czwartej ćwiartki, jako leżące po prawej stronie prostej. Większość punktów, podobnie jak w przypadku funkcji pisanej własnoręcznie, została jednak zakwalifikowana jako leżące na prostej.

Różnice te wynikają ze specyfiki zapisu liczb zmiennoprzecinkowych oraz sposobu wykonywania operacji arytmetycznych na takich liczbach. Przy dużych wartościach liczbowych kolejne reprezentowalne liczby są coraz bardziej oddalone od siebie. Wynika to ze to ze wzrostu cechy. W konsekwencji odejmowanie dużych, zbliżonych do siebie wartości może prowadzić do utraty precyzji i błędnych wyników. Mimo, że współrzędne punktów powinny różnić się od siebie, to ze względu na ograniczoną precyzję ich zapisu różnice te mogą zostać pominięte. W efekcie różnica takich wartości przyjmuje 0, co skutkuje błędnym obliczeniem wyznacznika, a tym samym niepoprawną klasyfikacją punktów względem prostej. Błędy tego typu należą do tzw. błędów zaokrągleń.

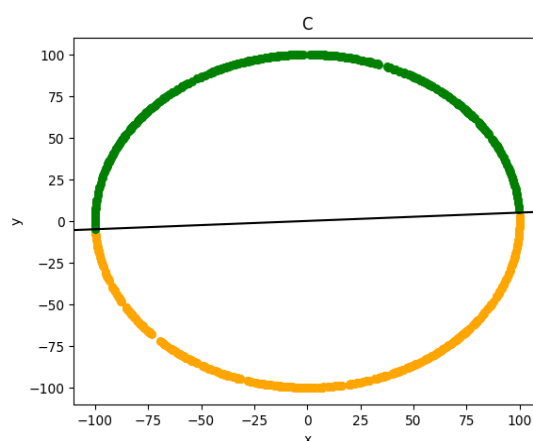
Funkcje wyznaczników 3×3 nie zakwalifikowały żadnego z punktów jako leżącego na prostej, niezależnie od precyzji liczb zmiennoprzecinkowych oraz wartości współczynnika tolerancji ε (wykres nr 11).

4.3. Zbiór punktów C

Precyzja	Współczynnik tolerancji ε	Funkcja obliczania wyznacznika	Liczba punktów na lewo od prostej	Liczba punktów na prostej	Liczba punktów na prawo od prostej
float 32-bit float 64-bit	0 10^{-12} 10^{-10} 10^{-8} 10^{-6}	mat_det_2x2 mat_det_2x2_lib mat_det_3x3 mat_det_3x3_lib	498	0	502

Tabela nr 3: wyniki klasyfikacji punktów zbioru C względem prostej

Wyniki klasyfikacji punktów ze zbioru C względem prostej były poprawne, identyczne, niezależnie od użytej precyzji liczb zmiennoprzecinkowych, współczynnika tolerancji ε oraz funkcji obliczania wyznacznika (Wykres nr 12). Przedstawiono je w tabeli nr 3.

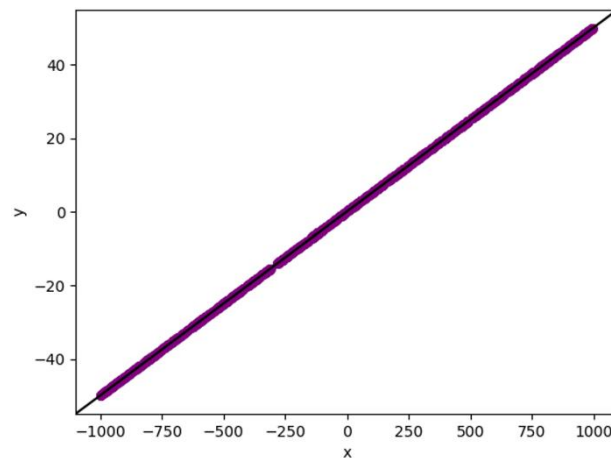


Wykres 12: Wynik poprawnej klasyfikacji punktów ze zbioru C względem prostej

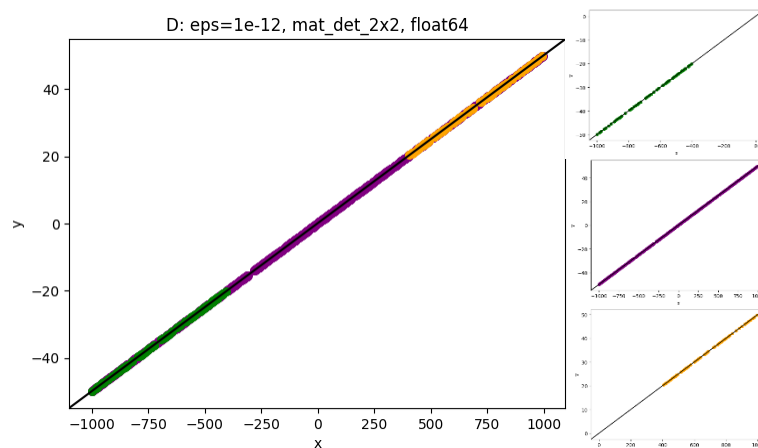
4.4. Zbiór punktów D

Precyzja	Współczynnik tolerancji ε	Funkcja obliczania wyznacznika oraz wynik klasyfikacji w postaci [liczba punktów na lewo], [na prostej], [na prawo]			
		det_2x2	det_2x2_lib	det_3x3	det_3x3_lib
float 64-bit	0	138, 722, 140	153, 696, 151	68, 755, 177	366, 296, 338
	10^{-12}	81, 841, 78	106, 795, 99	0, 1000, 0	0, 1000, 0
	10^{-10}	0, 1000, 0	0, 1000, 0	0, 1000, 0	0, 1000, 0
	10^{-8}	0, 1000, 0	0, 1000, 0	0, 1000, 0	0, 1000, 0
	10^{-6}	0, 1000, 0	0, 1000, 0	0, 1000, 0	0, 1000, 0
float 32-bit	0	164, 663, 173	511, 0, 489	250, 577, 173	501, 68, 431
	10^{-12}	164, 663, 173	511, 0, 489	250, 577, 173	421, 193, 386
	10^{-10}	164, 663, 173	511, 0, 489	250, 577, 173	421, 193, 386
	10^{-8}	164, 663, 173	510, 1, 489	250, 577, 173	420, 194, 386
	10^{-6}	161, 668, 171	491, 36, 473	199, 662, 139	267, 482, 251

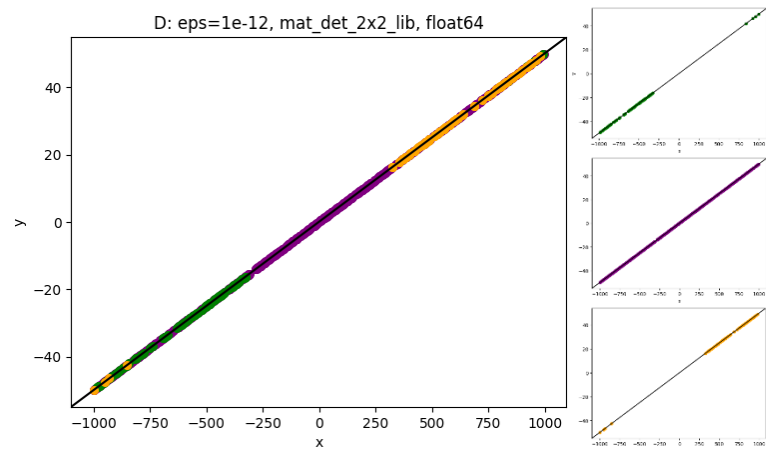
Tabela nr 4: wyniki klasyfikacji punktów zbioru D względem prostej



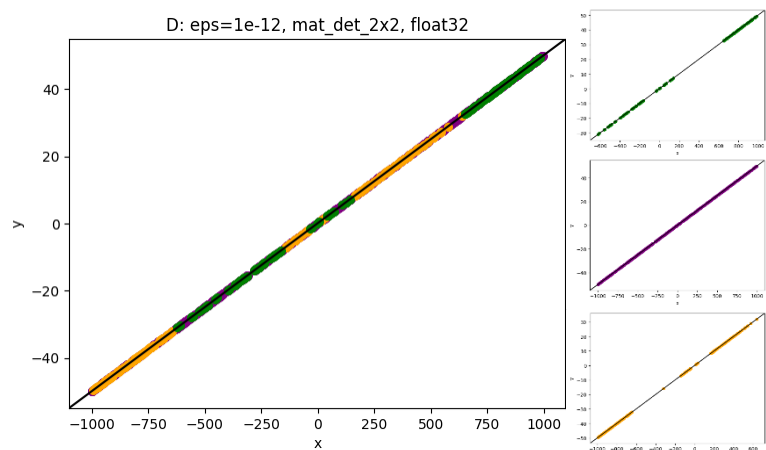
Wykres 13: Poprawna klasyfikacja punktów ze zbioru C względem prostej



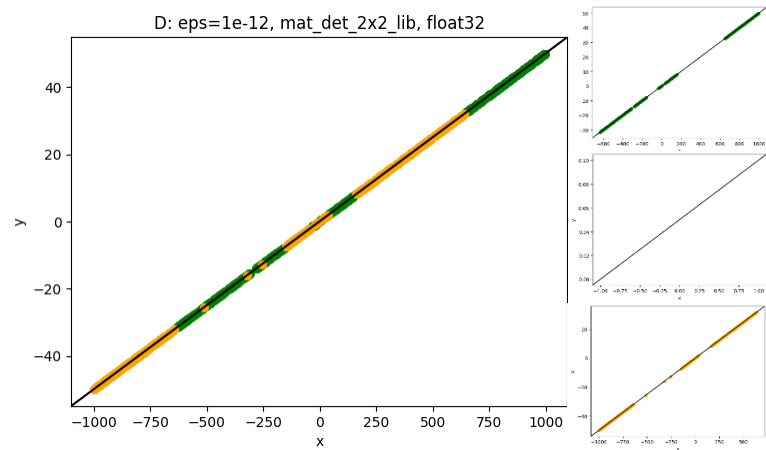
Wykres 14: klasyfikacja punktów ze zbioru C względem prostej dla funkcji mat_det_2x2 , $\varepsilon = 10^{-12}$, float64

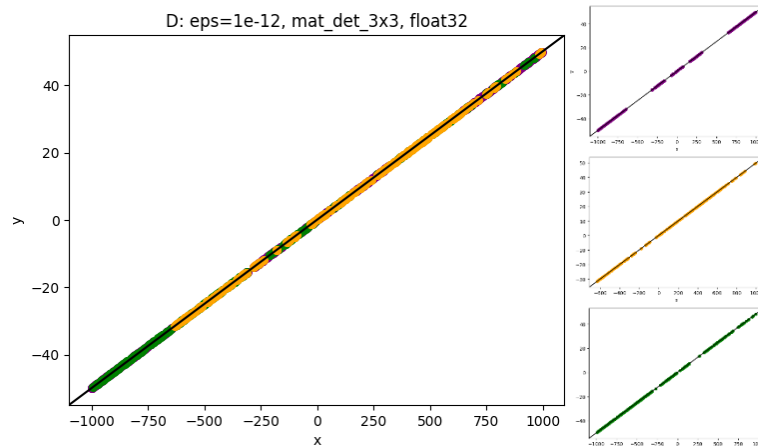


Wykres 15: klasyfikacja punktów ze zbioru C względem prostej dla funkcji mat_det_2x2_lib , $\varepsilon = 10^{-12}$, float64

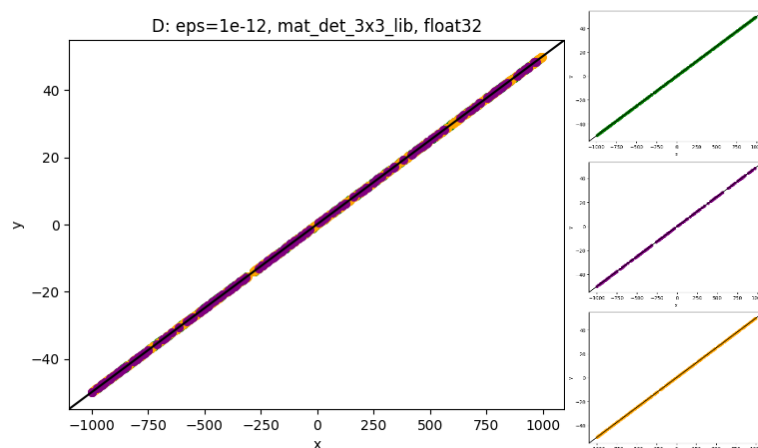


Wykres 16: klasyfikacja punktów ze zbioru C względem prostej dla funkcji mat_det_2x2 , $\varepsilon = 10^{-12}$, float32





Wykres 18: klasyfikacja punktów ze zbioru C względem prostej dla funkcji mat_det_3x3 , $\varepsilon = 10^{-12}$, float32



Wykres 19: klasyfikacja punktów ze zbioru C względem prostej dla funkcji mat_det_3x3_lib , $\varepsilon = 10^{-12}$, float32

W przypadku punktów ze zbioru D występuje duża rozbieżność w ich kwalifikacji dla doboru parametrów testowych, mimo że każdy z nich wygenerowano jako punkt leżący na prostej. Wyniki przedstawiono w tabeli nr 4.

Dla funkcji mat_det_2x2 oraz funkcji mat_det_2x2_lib osiągnięto maksymalną dokładność kwalifikacji punktów stosując współczynnik tolerancji $\varepsilon \leq 10^{-10}$.

Dla funkcji mat_det_2x2 oraz funkcji mat_det_3x3_lib osiągnięto maksymalną dokładność kwalifikacji punktów stosując współczynnik tolerancji $\varepsilon \leq 10^{-12}$.

Klasyfikację punktów dowolnej funkcji obliczania wyznacznika, precyzji float 64-bit oraz $\varepsilon \leq 10^{-10}$ przedstawiono na wykresie nr 13.

Dla precyzji float 64-bit $\varepsilon = 10^{-12}$ zdecydowanie gorzej wypadły funkcje obliczania wyznacznika 2x2. Kwalifikacje punktów dla tych sposobów przedstawiają wykresy nr 14 oraz 15.

W przypadku precyzji float 32-bit wszystkie klasyfikacje zawierały błędy, ich wyniki przedstawiają wykresy o nr od 16 do 19, Tutaj w każdym przypadku funkcje napisane własnoręcznie wypadły zdecydowanie lepiej od funkcji bibliotecznej. Największą niedokładność zaobserwować można było w przypadku punktów o dużych współrzędnych.

Przyczyna takiego zachowania algorytmu jest taka sama jak w przypadku kwalifikacji punktów ze zbioru B, Tutaj jednak utrata precyzji powoduje dużo słabsze wyniki, powoduje to dużo bardziej dokładny przypadek testowy.

4.4. Porównanie czasu działania dla funkcji obliczania wyznacznika

Porównanie czasu działania różnych implementacji obliczania wyznacznika przeprowadzono dla punktów o współrzędnych z przedziału $[-1000, 1000]$ oraz dla $\varepsilon = 10^{-12}$. Jego wyniki przedstawiono w tabeli nr 5.

Liczność zbioru punktów	Czas działania mat_det_2x2	Czas działania mat_det_2x2_lib	Czas działania mat_det_3x3	Czas działania mat_det_3x3_lib
10^2	0,0002 s	0,0006 s	0,0001 s	0,0007 s
10^3	0,0006 s	0,0046 s	0,0007 s	0,0045 s
10^4	0,0685 s	0,0454 s	0,0065 s	0,0460 s
10^5	0,0535 s	0,4277 s	0,0633 s	0,4523 s
10^6	0,5303 s	4,3098 s	0,6751 s	4,7397 s
10^7	5,5641 s	47,0198 s	7,6138 s	53,9695 s

Tabela nr 5: czas działania algorytmów obliczania wyznacznika

Dla dużych zbiorów czas działania funkcji napisanych własnoręcznie jest mniejszy o około rząd wielkości od czasu działania funkcji bibliotecznych. Dla małych zbiorów różnica czasu wykonywania obliczeń jest praktycznie niezauważalna.

5. Wnioski

Na podstawie analizy wyników doświadczenia nasuwają się następujące wnioski:

- 1) Precyzja przechowywania danych liczbowych zmiennoprzecinkowych ma ogromny wpływ na zachowanie programu. Precyzja float 64-bit często daje lepsze rezultaty, niż float 32-bit, szczególnie dla dużych liczb.
- 2) Dobór odpowiedniego algorytmu obliczania wyznacznika jest ważny dla osiągnięcia dokładnej kwalifikacji punktów – poprawnego wyniku. Własnoręcznie pisane funkcje

wykazywały dużo lepszą dokładność, niż funkcje biblioteczne dla precyzji float 32-bit.

- 3) Dobór współczynnika tolerancji ε także wpływa, szczególnie w przypadku testowym D, na ogólną tolerancję błędów algorytmów obliczania wyznacznika. Tutaj najlepiej sprawdza się wartość 10^{-12} .
- 4) Dobrą praktyką jest graficzne przedstawienie wyników doświadczenia – pomaga to określić zależności i cechy wspólne błędów kwalifikacji dla przypadków testowych.
- 5) W przypadkach testowych A oraz C (zbioru punktów o małych współrzędnych) wybór powyższych parametrów nie miał wpływu na wynik kwalifikacji punktów względem prostej.
- 6) W przypadkach testowych o dużych współrzędnych należy przyłożyć dużą uwagę do wyboru odpowiednich parametrów oraz algorytmu, by uniknąć utraty precyzji oraz błędów kwalifikacji. Warto przeprowadzić podobne doświadczenie. Ułatwi to proces wyboru odpowiedniego podejścia.
- 7) Czas działania funkcji napisanych własnoręcznie różni się o około rząd wielkości od czasu działania funkcji bibliotecznych dla dużych zbiorów. Dla małych zbiorów różnica czasu wykonywania obliczeń jest praktycznie niezauważalna.