Algorytmy geometryczne

Sprawozdanie z ćwiczeń 1.

Joanna Kulig, grupa 5

Dane urządzenia, na którym wykonano ćwiczenie:

Procesor: 1,4 GHz Czterordzeniowy procesor Intel Core i5
Komputer z systemem macOS Monterey
Środowisko: jupyter notebook
Do rozwiązania ćwiczenia wykorzystano język Python wraz z bibliotekami numpy, matplotlib oraz random.

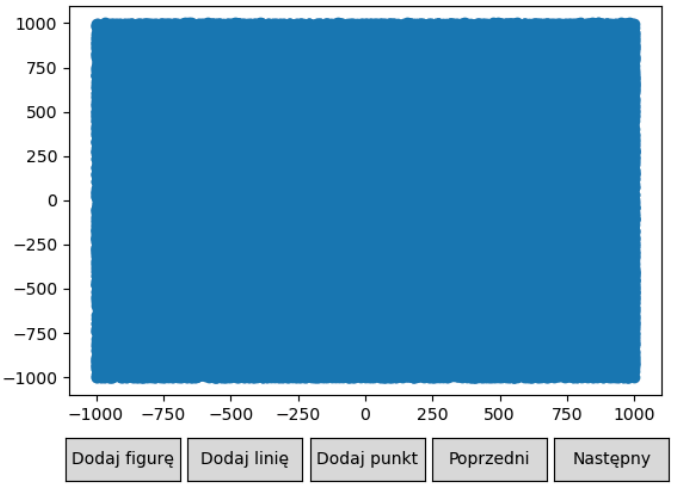
Ćwiczenie polegało na zestawieniu oraz porównaniu wyników klasyfikacji położenia punktów względem odcinka w zależności od precyzji obliczeń, tolerancji dla zera oraz użytej metody obliczania wyznacznika.

1. Wygenerowanie zbiorów punktów.

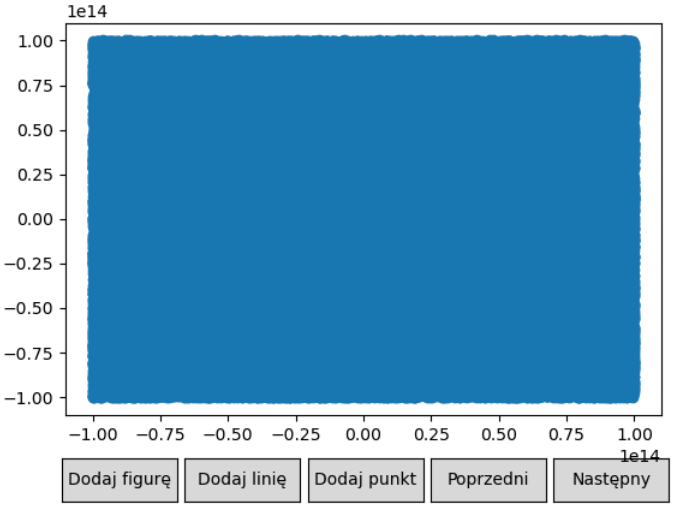
Losowe wygenerowanie punktów wykonano przy wykorzystaniu funkcji random. Wygenerowano osobne zbiory dla float32 i float64. Wyniki zestawiono w osobnych tablicach.

1. **Zestaw danych 1**: 10^5 losowych punktów o współrzędnych z przedziału $[-10^3, 10^3]$
2. **Zestaw danych 2**: 10^5 losowych punktów o współrzędnych z przedziału $[-10^{14}, 10^{14}]$
3. **Zestaw danych 3**: 10^3 losowych punktów leżących na okręgu o środku w punkcie (0,0) i promieniu R=100. Do wygenerowania punktów wykorzystano funkcje trygonometryczne z biblioteki numpy.
4. **Zestaw danych 4**: 10^3 losowych punktów o współrzędnych z przedziału $[-10^3, 10^3]$ leżących na prostej wyznaczonej przez wektor [a, b], gdzie a = [-1, 0], b = [1, 0.1]. Do wygenerowania punktów wykorzystano równanie prostej przechodzącej przez dwa podane punkty

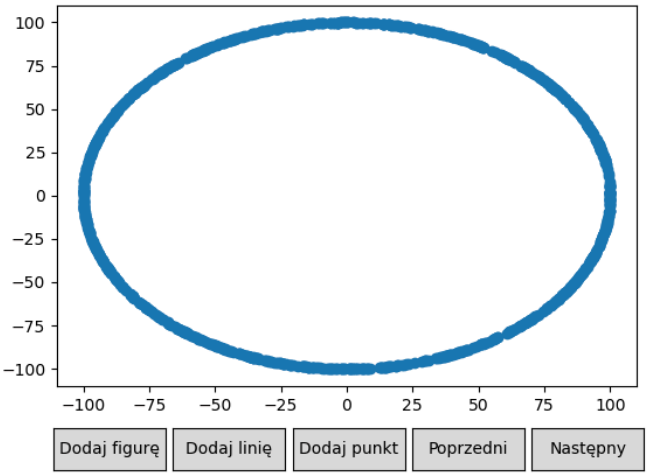
Zestawy punktów zostały zwizualizowane przy wykorzystaniu narzędzia graficznego opartego o bibliotekę matplotlib.



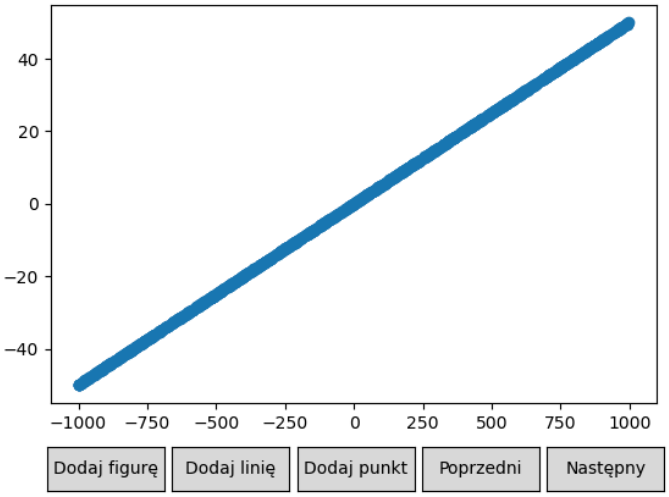
Wykres 1.1 Zestaw danych 1



Wykres 1.2 Zestaw danych 2



Wykres 1.3 Zestaw danych



Wykres 1.4 Zestaw danych 4

2. Metody obliczania wyznacznika.

W obliczeniach wykorzystano następujące wyznaczniki:

- Wyznaczniki 2x2 oraz 3x3 korzystając z funkcji z biblioteki numpy: **np.linalg.det()**
- Wyznaczniki 2x2 oraz 3x3 obliczone przy użyciu własnej, zaimplementowanej funkcji opartej na poniższych wzorach:

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

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

gdzie:
a, b - punkty wyznaczające odcinek względem którego wykonujemy klasyfikację punktów
c - sprawdzany punkt

3. Tolerancja dla zera oraz precyzja.

Wykorzystano następujące tolerancje dla zera:

$$10^{-16}, 10^{-14}, 10^{-12}, 10^{-10}$$

Wykorzystano następujące precyzje:

float64 oraz float32 z biblioteki numpy

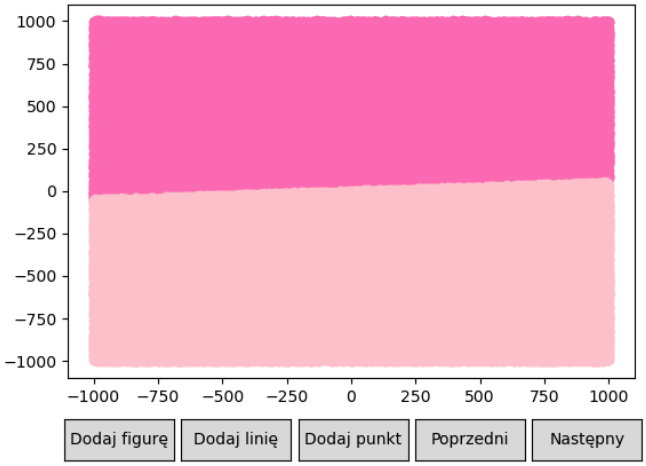
4. Kategoryzacja.

Do zrealizowania zadania została wykorzystana funkcja, która dla każdego zbioru punktów, każdego z czterech sposobów obliczania wyznacznika oraz każdej z wybranych tolerancji dla zera oblicza wyznacznik i klasyfikuje punkty na odpowiednią stronę prostej. Odpowiednie kolory odpowiadają klasyfikacji:

- Kolor ciemnoróżowy - punkt leży po lewej stronie prostej,
- Kolor jasnoróżowy - punkt leży po prawej stronie prostej,
- Kolor zielony - punkt leży na prostej.

5. Wizualizacje klasyfikacji punktów - precyzja float64

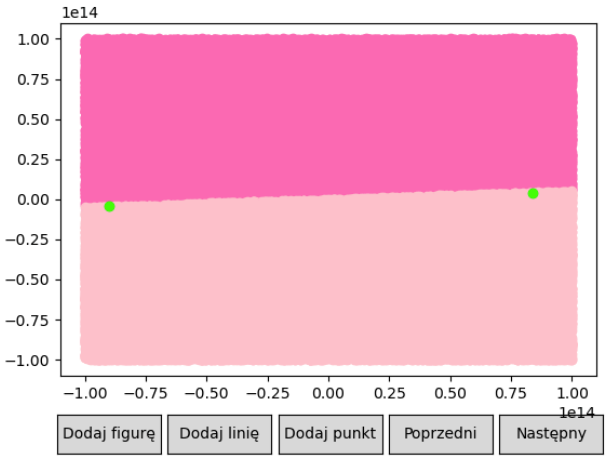
Dla zestawu danych 1 otrzymano takie same wyniki dla wszystkich sposobów liczenia wyznacznika oraz wszystkich tolerancji dla zera.



Wykres 2.1 Klasyfikacja dla zestawu danych 1 dla każdego z sposobów obliczania wyznacznika oraz każdej z wybranych tolerancji dla zera

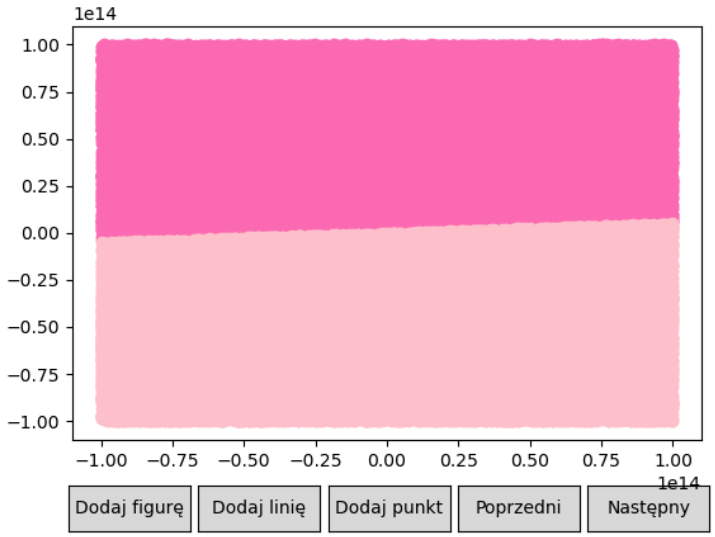
Po lewej : 59968 punktów, **po prawej:** 50032 punktów, **współliniowe**: 0 punktów

Dla zestawu danych 2 otrzymano różne wyniki. Własny wyznacznik 2x2 sklasyfikował kilka punktów współliniowych, natomiast dla pozostałych sposobów otrzymano takie same wyniki.



Wykres 2.2.1 Klasyfikacja dla zestawu danych 2 dla wyznacznika 2x2 własnej implementacji oraz każdej z wybranych tolerancji dla zera

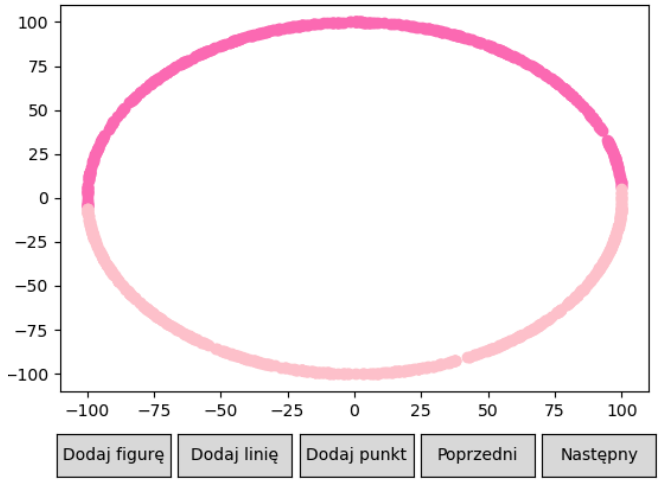
Po lewej : 50027 punktów, **po prawej:** 49971 punktów, **współliniowe**: 2 punktów



Wykres 2.2.2 Klasyfikacja dla zestawu danych 2 dla wyznaczników 3x3 oraz wyznacznika 2x2 z biblioteki numpy dla każdej z wybranych tolerancji dla zera

Po lewej : 50028 punktów, **po prawej:** 49972 punktów, **współliniowe**: 0 punktów

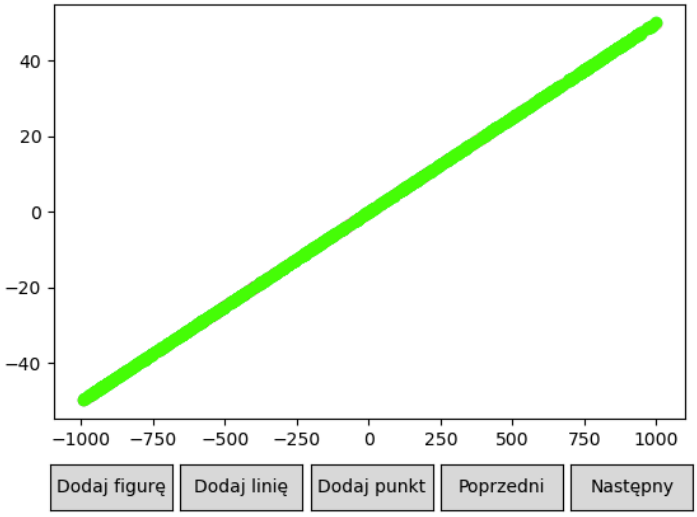
Dla zestawu danych 3 otrzymano jednakowe wyniki dla każdego sposobu obliczania wyznacznika oraz każdej z wybranych tolerancji dla zera



Wykres 2.3 Klasyfikacja dla zestawu danych 3 dla każdego z wybranych sposobów liczenia wyznacznika oraz każdej z wybranych tolerancji dla zera

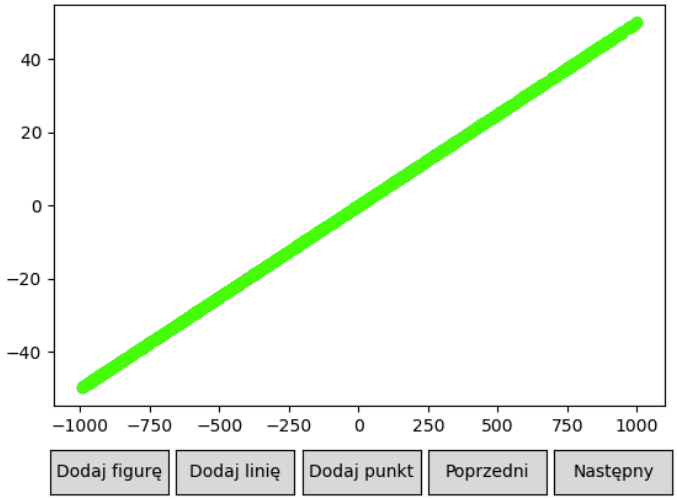
Po lewej : 475 punktów, **po prawej:** 525 punktów, **współliniowe**: 0 punktów

W zestawie danych 4, wbrew oczekiwaniom, nie wszystkie punkty zostały zaklasyfikowane jako współliniowe. Punkty niewspółliniowe zaklasyfikowano dla tolerancji dla zera 10^{-16} oraz 10^{-14} w przypadku wyznaczników 2x2, oraz dla tolerancji dla zera 10^{-16} w przypadku wyznaczników 3x3.



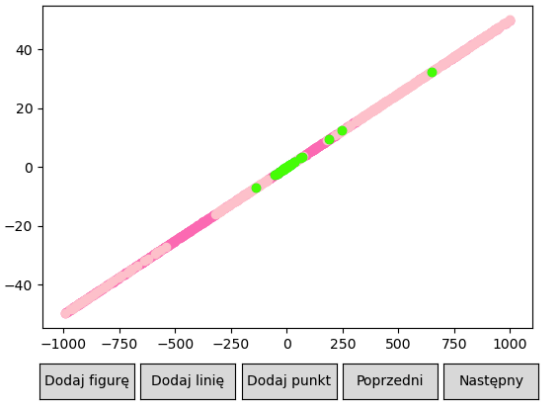
Wykres 2.4.1 Klasyfikacja dla zestawu danych 4 dla tolerancji dla zera 10^{-16} dla wyznacznika 2x2 własnej implementacji

Po lewej : 123 punktów, **po prawej:** 157 punktów, **współliniowe**: 720 punktów



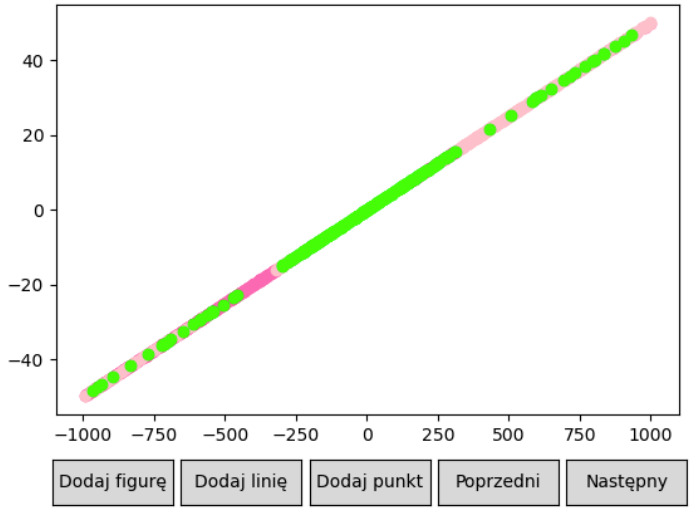
Wykres 2.4.2 Klasyfikacja dla zestawu danych 4 dla tolerancji dla zera 10^{-14} dla wyznacznika 2x2 własnej implementacji

Po lewej : 114 punktów, **po prawej:** 137 punktów, **współliniowe**: 749 punktów



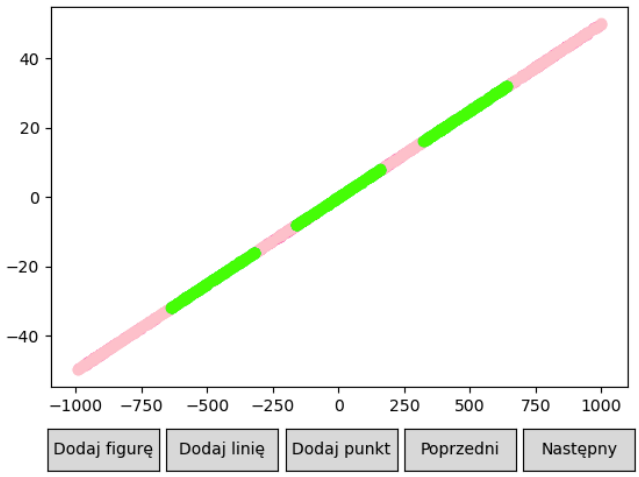
Wykres 2.4.3 Klasyfikacja dla zestawu danych 4 dla wyznacznika 2x2 z biblioteki numpy dla tolerancji dla zera 10^{-16}

Po lewej : 459 punktów, **po prawej:** 502 punktów, **współliniowe**: 39 punktów



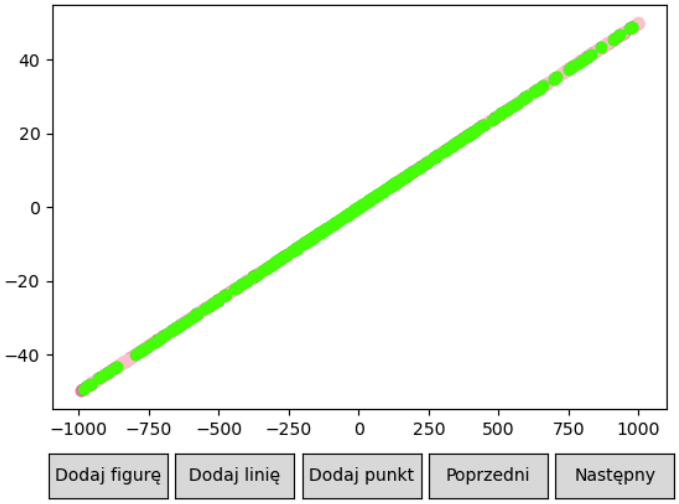
Wykres 2.4.4 Klasyfikacja dla zestawu danych 4 dla wyznacznika 2x2 z biblioteki numpy dla tolerancji dla zera 10^{-14}

Po lewej : 340 punktów, **po prawej:** 382 punktów, **współliniowe**: 278 punktów



Wykres 2.4.5 Klasyfikacja dla zestawu danych 4 dla tolerancji dla zera 10^{-16} dla wyznacznika 3x3 własnej implementacji

Po lewej : 119 punktów, **po prawej:** 364 punktów, **współliniowe**: 417 punktów

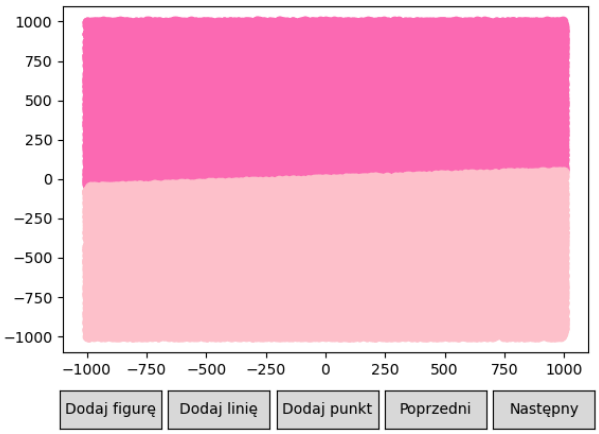


Wykres 2.4.6 Klasyfikacja dla zestawu danych 4 dla tolerancji dla zera 10^{-16} dla wyznacznika 3x3 z biblioteki numpy

Po lewej : 340 punktów, **po prawej:** 346 punktów, **współliniowe**: 314 punktów

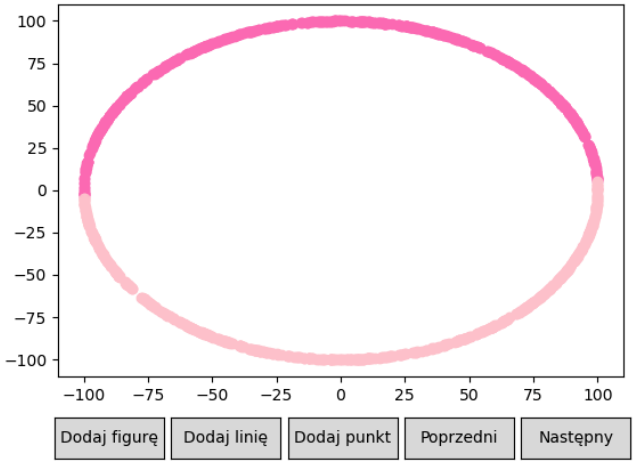
6. Wizualizacje klasyfikacji punktów - precyzja float32

Podobnie jak w przypadku precyzji float64, dla zestawu danych 1 oraz zestawu danych 3 otrzymano takie same wyniki dla każdej tolerancji dla zera i dla każdego sposobu obliczania wyznacznika.



Wykres 3.1 Klasyfikacja dla zestawu danych 1 dla każdego sposobu obliczania wyznacznika i dla każdej z wybranych tolerancji dla zera

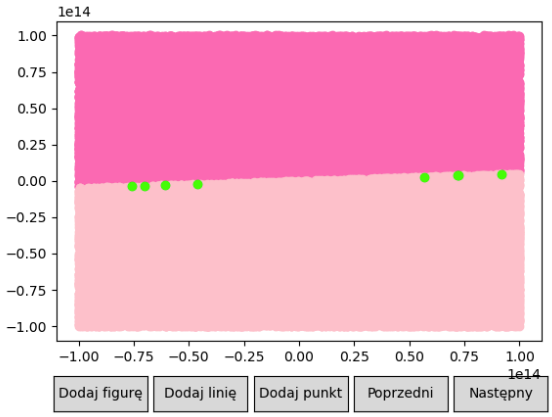
Po lewej : 50056 punktów, **po prawej:** 49944 punktów, **współliniowe**: 314 punktów



Wykres 3.2 Klasyfikacja dla zestawu danych 3 dla każdego sposobu obliczania wyznacznika i dla każdej z wybranych tolerancji dla zera

Po lewej : 500 punktów, **po prawej:** 500 punktów, **współliniowe**: 314 punktów

Dla zestawu danych 2 otrzymano takie same wyniki dla każdej tolerancji dla zera dla wyznacznika 2x2 własnej implementacji - pojawiły się tam punkty współliniowe. W metodach 3x3 obliczania wyznacznika i dla każdej z wybranych tolerancji wyniki były takie same, ale nie sklasyfikowały one punktów współliniowych. W metodzie obliczania wyznacznika 2x2 z użyciem biblioteki numpy pojawiły się inne wyniki, ale bez punktów współliniowych.



Wykres 3.3 Klasyfikacja dla zestawu danych 2 dla wyznacznika 2x2 własnej implementacji dla każdej z wybranych tolerancji dla zera

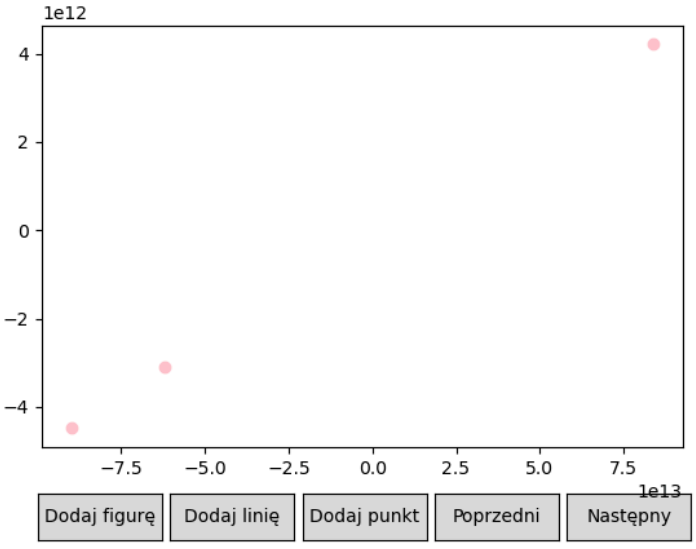
Po lewej : 50032 punktów, **po prawej:** 49960 punktów, **współliniowe**: 8 punktów

W zestawie danych 4 żadna z metod obliczania wyznacznika nie uzyskała samych punktów współliniowych dla żadnej z wybranych tolerancji dla zera.

7. Różnice w klasyfikacji dla metod obliczania wyznacznika własnej implementacji oraz przy użyciu biblioteki numpy dla precyzji float64

Wykorzystano funkcję, która dla wybranego zestawu danych oraz wybranej metody obliczania wyznacznika (2x2 lub 3x3) wyznacza ilość punktów, w których klasyfikacji nastąpiła różnica dla funkcji własnej implementacji oraz funkcji z biblioteki numpy.

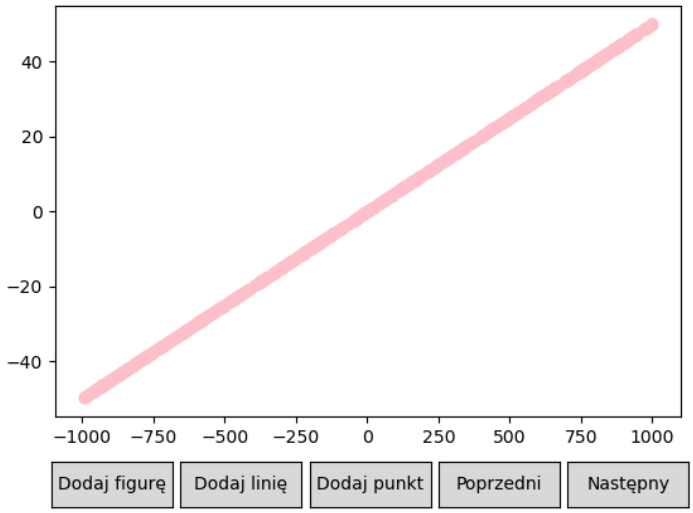
Dla zestawu danych 1 i zestawu danych 3 nie wykryto żadnych różnic. Dla zestawu danych 2 wykryto różnice dla metody obliczania wyznacznika 2x2.



Wykres 4.1 Różnice w klasyfikacji zestawu danych 2 dla wyznaczników 2x2 przy precyzji float64

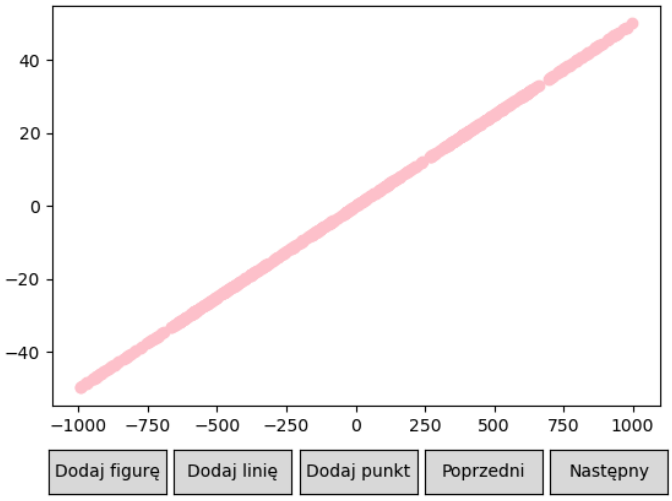
Liczba różnic : 3 punkty

Dla zestawu danych 4 wykryto różnice zarówno dla wyznaczników 2x2, jak i 3x3.



Wykres 4.2 Różnice w klasyfikacji zestawu danych 4 dla wyznaczników 2x2 przy precyzji float64

Liczba różnic : 732 punkty



Wykres 4.3 Różnice w klasyfikacji zestawu danych 4 dla wyznaczników 3x3 przy precyzji float64

Liczba różnic : 572 punkty

8. Zestawienie otrzymanych wyników

Dla precyzji float32:

Tabela 1: Zestaw danych 1 - zestawienie wyników dla precyzji float32

Nazwa wyznacznika	Tolerancja	Po lewej:	Po prawej:	Współliniowe:
Wyznacznik 2x2 własnej implementacji:	10^{-16}	50056	49944	0
	10^{-14}	50056	49944	0
	10^{-12}	50056	49944	0
	10^{-10}	50056	49944	0
Wyznacznik 3x3 własnej implementacji:	10^{-16}	50056	49944	0
	10^{-14}	50056	49944	0
	10^{-12}	50056	49944	0
	10^{-10}	50056	49944	0
Wyznacznik 2x2 z biblioteki numpy	10^{-16}	50056	49944	0
	10^{-14}	50056	49944	0
	10^{-12}	50056	49944	0
	10^{-10}	50056	49944	0
Wyznacznik 3x3 z biblioteki numpy	10^{-16}	50056	49944	0
	10^{-14}	50056	49944	0
	10^{-12}	50056	49944	0
	10^{-10}	50056	49944	0

Tabela 2: Zestaw danych 2 - zestawienie wyników dla precyzji float32

Nazwa wyznacznika	Tolerancja	Po lewej:	Po prawej:	Współliniowe:
Wyznacznik 2x2 własnej implementacji:	10^{-16}	50032	49960	8
	10^{-14}	50032	49960	8
	10^{-12}	50032	49960	8
	10^{-10}	50032	49960	8
Wyznacznik 3x3 własnej implementacji:	10^{-16}	50035	49965	0
	10^{-14}	50035	49965	0
	10^{-12}	50035	49965	0
	10^{-10}	50035	49965	0
Wyznacznik 2x2 z biblioteki numpy	10^{-16}	50037	49963	0
	10^{-14}	50037	49963	0
	10^{-12}	50037	49963	0
	10^{-10}	50037	49963	0
Wyznacznik 3x3 z biblioteki numpy	10^{-16}	50035	49965	0
	10^{-14}	50035	49965	0
	10^{-12}	50035	49965	0
	10^{-10}	50035	49965	0

Tabela 3: Zestaw danych 3 - zestawienie wyników dla precyzji float32

Nazwa wyznacznika	Tolerancja	Po lewej:	Po prawej:	Współliniowe:
Wyznacznik 2x2 własnej implementacji:	10^{-16}	500	500	8
	10^{-14}	500	500	8
	10^{-12}	500	500	8
	10^{-10}	500	500	8
Wyznacznik 3x3 własnej implementacji:	10^{-16}	500	500	0
	10^{-14}	500	500	0
	10^{-12}	500	500	0
	10^{-10}	500	500	0
Wyznacznik 2x2 z biblioteki numpy	10^{-16}	500	500	0
	10^{-14}	500	500	0
	10^{-12}	500	500	0
	10^{-10}	500	500	0
Wyznacznik 3x3 z biblioteki numpy	10^{-16}	500	500	0
	10^{-14}	500	500	0
	10^{-12}	500	500	0
	10^{-10}	500	500	0

Tabela 4: Zestaw danych 4 - zestawienie wyników dla precyzji float32

Nazwa wyznacznika	Tolerancja	Po lewej:	Po prawej:	Współliniowe:
Wyznacznik 2x2 własnej implementacji:	10^{-16}	400	400	200
	10^{-14}	396	400	204
	10^{-12}	389	393	218
	10^{-10}	389	393	218
Wyznacznik 3x3 własnej implementacji:	10^{-16}	460	467	73
	10^{-14}	389	393	218
	10^{-12}	389	393	218
	10^{-10}	389	393	218
Wyznacznik 2x2 z biblioteki numpy	10^{-16}	499	495	6
	10^{-14}	465	469	66
	10^{-12}	389	393	218
	10^{-10}	389	393	218
Wyznacznik 3x3 z biblioteki numpy	10^{-16}	473	463	64
	10^{-14}	389	393	218
	10^{-12}	389	393	218
	10^{-10}	389	393	218

Dla precyzji float64

Tabela 5: Zestaw danych 1 - zestawienie wyników dla precyzji float64

Nazwa wyznacznika	Tolerancja	Po lewej:	Po prawej:	Współliniowe:
Wyznacznik 2x2 własnej implementacji:	10^{-16}	49968	50032	0
	10^{-14}	49968	50032	0
	10^{-12}	49968	50032	0
	10^{-10}	49968	50032	0
Wyznacznik 3x3 własnej implementacji:	10^{-16}	49968	50032	0
	10^{-14}	49968	50032	0
	10^{-12}	49968	50032	0
	10^{-10}	49968	50032	0
Wyznacznik 2x2 z biblioteki numpy	10^{-16}	49968	50032	0
	10^{-14}	49968	50032	0
	10^{-12}	49968	50032	0
	10^{-10}	49968	50032	0
Wyznacznik 3x3 z biblioteki numpy	10^{-16}	49968	50032	0
	10^{-14}	49968	50032	0
	10^{-12}	49968	50032	0
	10^{-10}	49968	50032	0

Tabela 6: Zestaw danych 2 - zestawienie wyników dla precyzji float64

Nazwa wyznacznika	Tolerancja	Po lewej:	Po prawej:	Współliniowe:
Wyznacznik 2x2 własnej implementacji:	10^{-16}	50027	49971	2
	10^{-14}	50027	49971	2
	10^{-12}	50027	49971	2
	10^{-10}	50027	49971	2
Wyznacznik 3x3 własnej implementacji:	10^{-16}	50028	49972	0
	10^{-14}	50028	49972	0
	10^{-12}	50028	49972	0
	10^{-10}	50028	49972	0
Wyznacznik 2x2 z biblioteki numpy	10^{-16}	50027	49973	0
	10^{-14}	50027	49973	0
	10^{-12}	50027	49973	0
	10^{-10}	50027	49973	0
Wyznacznik 3x3 z biblioteki numpy	10^{-16}	50028	49972	0
	10^{-14}	50028	49972	0
	10^{-12}	50028	49972	0
	10^{-10}	50028	49972	0

Tabela 7: Zestaw danych 3 - zestawienie wyników dla precyzji float64

Nazwa wyznacznika	Tolerancja	Po lewej:	Po prawej:	Współliniowe:
Wyznacznik 2x2 własnej implementacji:	10^{-16}	475	535	0
	10^{-14}	475	525	0
	10^{-12}	475	525	0
	10^{-10}	475	525	0
Wyznacznik 3x3 własnej implementacji:	10^{-16}	475	535	0
	10^{-14}	475	525	0
	10^{-12}	475	525	0
	10^{-10}	475	525	0
Wyznacznik 2x2 z biblioteki numpy	10^{-16}	475	535	0
	10^{-14}	475	525	0
	10^{-12}	475	525	0
	10^{-10}	475	525	0
Wyznacznik 3x3 z biblioteki numpy	10^{-16}	475	535	0
	10^{-14}	475	525	0
	10^{-12}	475	525	0
	10^{-10}	475	525	0

Tabela 8: Zestaw danych 4 - zestawienie wyników dla precyzji float64

Nazwa wyznacznika	Tolerancja	Po lewej:	Po prawej:	Współliniowe:
Wyznacznik 2x2 własnej implementacji:	10^{-16}	123	157	720
	10^{-14}	114	137	749
	10^{-12}	0	0	1000
	10^{-10}	0	0	1000
Wyznacznik 3x3 własnej implementacji:	10^{-16}	219	364	417
	10^{-14}	0	0	1000
	10^{-12}	0	0	1000
	10^{-10}	0	0	1000
Wyznacznik 2x2 z biblioteki numpy	10^{-16}	459	502	39
	10^{-14}	340	382	278
	10^{-12}	0	0	1000
	10^{-10}	0	0	1000
Wyznacznik 3x3 z biblioteki numpy	10^{-16}	340	346	314
	10^{-14}	0	0	1000
	10^{-12}	0	0	1000
	10^{-10}	0	0	1000

9. Porównanie czasu działania funkcji obliczających wyznacznik

Wykorzystano funkcję, która dla danego zestawu danych mierzy czas, w którym dana funkcja obliczyła wyznacznik. Do testu wygenerowano 10^6 zestawów punktów a, b i c. Funkcjom obliczania wyznacznika z biblioteki numpy zajęło to dużo więcej czasu niż funkcjom własnej implementacji.

```
Czas obliczeń dla funkcji 2x2 z biblioteki numpy to: 07.066931 sekundy.
Czas obliczeń dla funkcji 3x3 z biblioteki numpy to: 07.775131 sekundy.
Czas obliczeń dla funkcji 2x2 własnej implementacji to: 00.388798 sekundy.
Czas obliczeń dla funkcji 3x3 własnej implementacji to: 00.508374 sekundy.
```

Rysunek 1: Wynik funkcji obliczającej czas.

10. Wnioski

Na podstawie danych zawartych w zestawieniu otrzymanych wyników można stwierdzić, że różnice w klasyfikacji położenia punktu względem odcinka w zależności od wszystkich parametrów są znaczące.

Dzięki wygenerowaniu innych zestawów danych dla różnych precyzji można zauważyć, że precyzja float64 daje dużo lepsze wyniki niż float32. Jest to szczególnie zauważalne dla zestawu danych 4, gdzie przy użyciu precyzji float32, mimo pomniejszającej się tolerancji dla zera, niewiele punktów zostaje zaklasyfikowanych jako współliniowe. W przypadku precyzji float64 dla małej tolerancji dla zera wszystkie punkty zostają zaklasyfikowane jako współliniowe.

Dla zestawów danych 1 i 3 precyzja float64 nie wykazała różnic w klasyfikacji punktów. W przypadku zestawu danych 2 różnice były nieznaczne. Najciekawszym okazał się zestaw danych 4, gdzie wbrew oczekiwaniom nie wszystkie punkty okazały się być współliniowe. Powodem tego mogły być ewentualne niedokładności w generowaniu punktów oraz obliczaniu wyznacznika. Wybierając jednak odpowiednio małą tolerancję dla zera otrzymuje się wyniki zgodne z oczekiwanymi. Wybierając tolerancję dla zera 10^{-12} można zauważyć, że bez względu jakim sposobem obliczono wyznacznik, wszystkie punkty są zaklasyfikowane jako współliniowe. Dla tolerancji dla zera 10^{-14} sposoby obliczania wyznacznika 3x3 klasyfikują wszystkie punkty jako współliniowe.

Funkcje obliczania wyznacznika z biblioteki numpy wypadają dużo gorzej pod względem czasu obliczeń niż funkcje własnej implementacji. Dzieje się tak dlatego, że biblioteki wykorzystują dużo niepotrzebnych funkcji. Wykonując proste zadania lepiej użyć funkcji własnej implementacji.

Do przyszłych projektów wybiorę kombinację precyzji float64, tolerancji dla zera 10^{-14} oraz wyznacznika własnej implementacji. Taki wybór daje dobre wyniki dla wszystkich zbiorów, ma szybki czas działania jednocześnie zachowując stosunkowo małą tolerancję dla zera, dzięki czemu istnieje duże prawdopodobieństwo, że otrzymana klasyfikacja nie będzie odbiegać od teoretycznej.