Algorytmy geometryczne

Sprawozdanie - ćwiczenie 1.

Dominik Adamczyk

Gr. 2

1. Treść i cel ćwiczenia:

Ćwiczenie polegało na wygenerowaniu zadanych zbiorów punktów, oraz podzieleniu ich na 3 zbiory zależnie od położenia tych punktów względem danej w ćwiczeniu prostej. Ćwiczenie należało przeprowadzić dla różnych metod obliczania wyznaczników i różnych tolerancji dla zera, w celu pokazania różnic mogących występować w zależności od przyjętej metodologii.

Wszystkie pomiary wykonano na laptopie z procesorem Intel i5-1135g7 o taktowaniu 4.2Ghz. Funkcje zostały napisane w środowisku Jupyter Notebook w języku Python 3.10 z pomocą narzędzia załączonego na potrzeby laboratoriów. System operacyjny na którym wykonywany był projekt to Windows 11.

2. Zbiory danych:

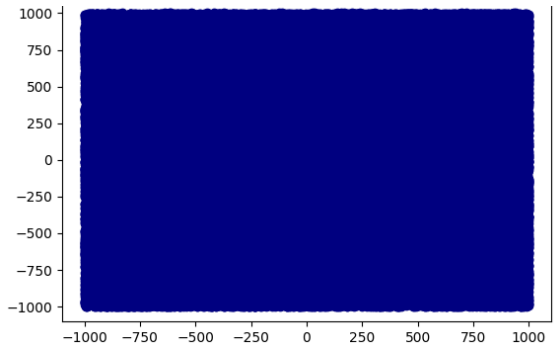
Na potrzeby ćwiczenia wygenerowane zostały 4 zbiory punktów rzeczywistych:

- Dane A: 10^5 losowych punktów o współrzędnych z przedziału [-1000; 1000]
- Dane B: 10^5 losowych punktów o współrzędnych z przedziału $[-10^{14};\ 10^{14}]$
- Dane C: 1000 losowych punktów leżących na okręgu o środku (0, 0) i promieniu R = 100
- Dane D: 1000 losowych punktów o współrzędnych z przedziału [-1000, 1000] leżących na prostej wyznaczonej przez wektor (a, b) gdzie a = [-1.0, 0.0], b = [1.0, 0.1]

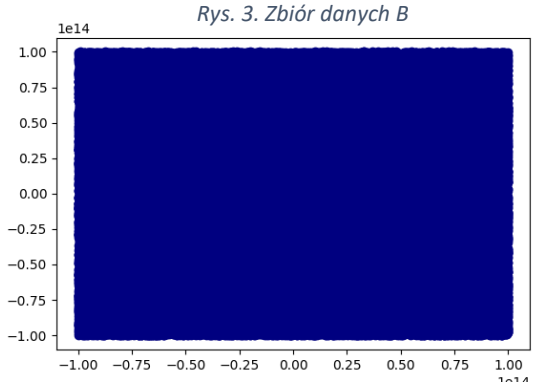
Punkty zostały wygenerowane przy pomocy biblioteki random i funkcji random.uniform. W przypadku zbioru C dodatkowo użyto biblioteki numpy i parametryzacji okręgu jako $(\cos t\,, \sin t)$. Do wygenerowanie zbioru D posłużono się parametryzacją prostej, po wcześniejszym obliczeniu jej równania.

Poniższe wykresy prezentują zbiory punktów wygenerowane przy pomocy dołączonego narzędzia korzystającego z biblioteki matplotlib.

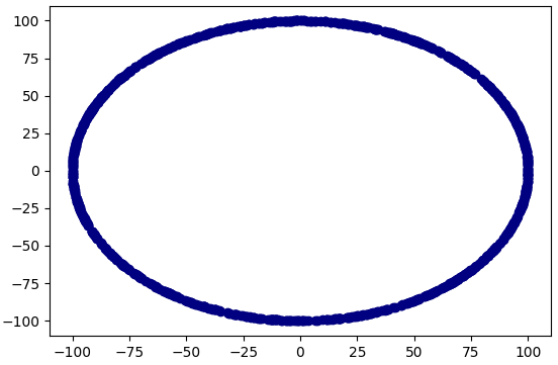
Rys. 4. Zbiór danych A



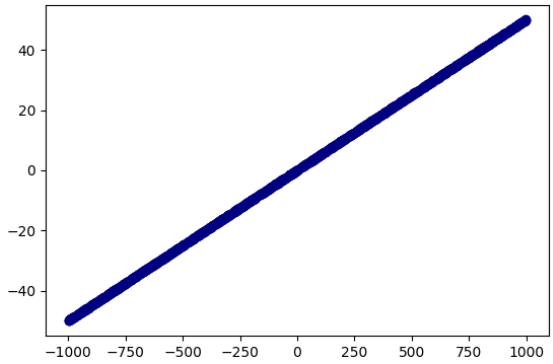
Rys. 3. Zbiór danych B



Rys. 1. Zbiór danych C



Rys. 2. Zbiór danych D



3. Przyjęte tolerancje oraz metody obliczania wyznacznika

Aby kategoryzacja punktów była możliwa konieczne jest przyjęcie funkcji, która będzie odpowiadała za określenie położenia punktu względem prostej. W tym zadaniu porównane będą skuteczności różnych metod obliczania wyznacznika konieczne do dokonania owej kategoryzacji. Dodatkowo pomiary zostaną przeprowadzone dla różnych tolerancji dla zera, tak aby móc stwierdzić względną dokładność różnych metod numerycznych.

Tabela 1. Wyznaczniki użyte do obliczeń

Rodzaj macierzy	Implementacja	Funkcja obliczająca wyznacznik
2x2	Własna implementacja	$\det(a,b,c) = \begin{vmatrix} a_x - c_x & a_y - c_y \\ b_x - c_x & b_y - c_y \end{vmatrix}$
	Biblioteka numpy	numpy.linalg.det()
3x3	Własna implementacja	$\det(a,b,c) = \begin{vmatrix} a_x & a_y & 1 \\ b_x & b_y & 1 \\ c_x & c_y & 1 \end{vmatrix}$
	Biblioteka numpy	numpy.linalg.det()

a, b to punkty podane w 2., natomiast c to punkt badany

Warto zwrócić uwagę, że funkcja wyznacznika importowana z biblioteki numpy nie zmienia się w zależności od rodzaju macierzy. Wyznacznik biblioteczny jest uniwersalny dla n-wymiarowych macierzy kwadratowych.

Przyjęte tolerancje dla zera wynoszą: 1e-17, 1e-14, 1e-11. W trakcie testów sprawdzone zostały też inne tolerancje z przedziału [1e-20; 1e-10], jednakże ze względu na czytelność wyników w omówieniu prezentowane są trzy powyższe – wystarcza to do zauważenia różnic między wyznacznikami.

4. Podział punktów

Zgodnie z treścią ćwiczenia punkty z zestawów danych zostały podzielone na trzy kategorie: na lewo od prostej, na prawo od prostej, oraz współliniowe. Prezentowane poniżej podziały zostały wykonane dla tolerancji dla zera równej 1e-17, pozostałe tolerancje są uwzględnione w tabelach końcowych. Punkty współliniowe, ze względu na swoją istotność zostały wyróżnione nie tylko kolorem, ale także większym rozmiarem.

Na potrzeby kategoryzacji, w tym punkcie przyjęto następujący schemat kolorów:

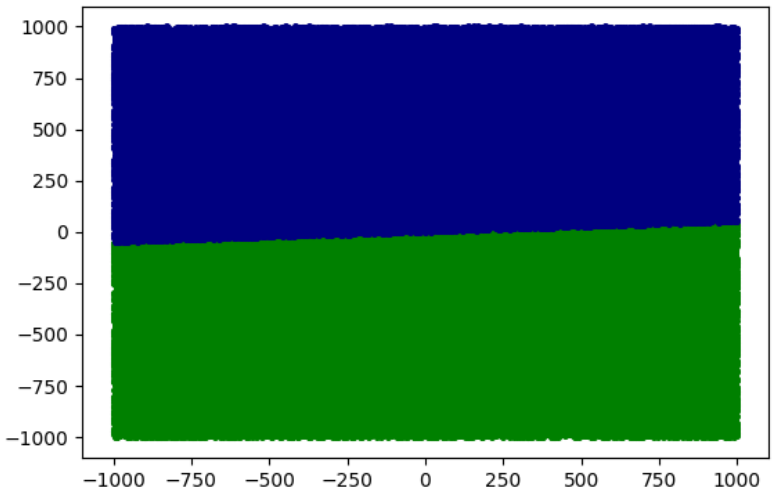
Niebieski – punkty po lewej stronie prostej
Zielony – punkty po prawej stronie prostej
Różowy – punkty współliniowe z prostą

4.1 Dane A

W przypadku pierwszego zestawu danych uzyskano taki sam wynik niezależnie od metody obliczeń, czy tolerancji. Ze względu na niewielki przedział generowanych punktów nie występują niejednoznaczności w obliczeniach.

Rys. 5 Podział punktów z zestawu A.

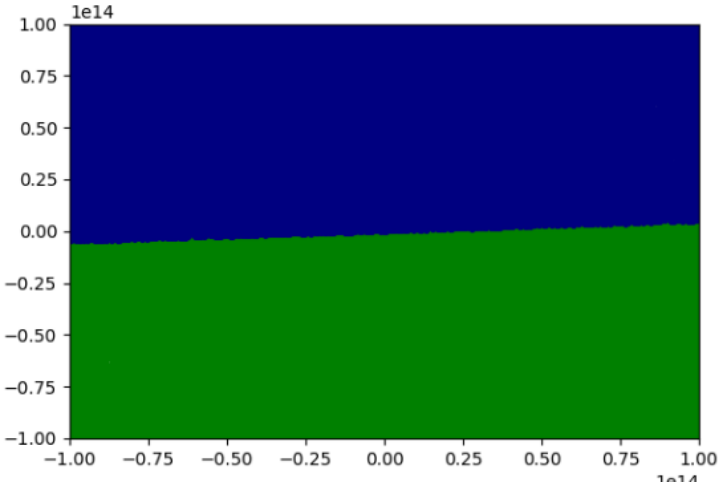
Po lewej: 50385, współliniowe: 0, po prawej: 49615



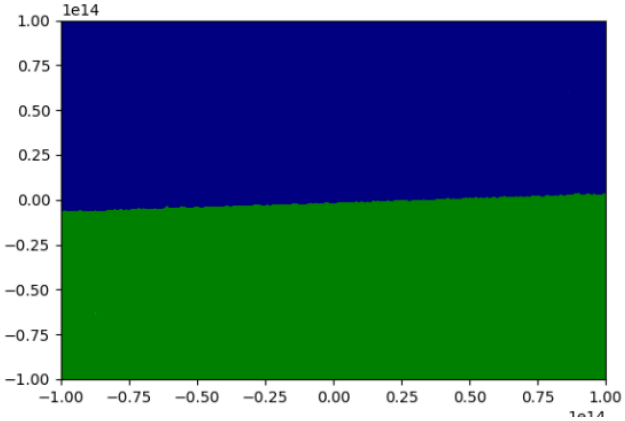
4.2 Dane B

Dla zestawu danych B wyniki różnią się w zależności od zastosowanych metod obliczeń. Wyznaczniki 3x3 zwracają te same wyniki, natomiast wyznaczniki 2x2 różnią się od 3x3 jak i między sobą

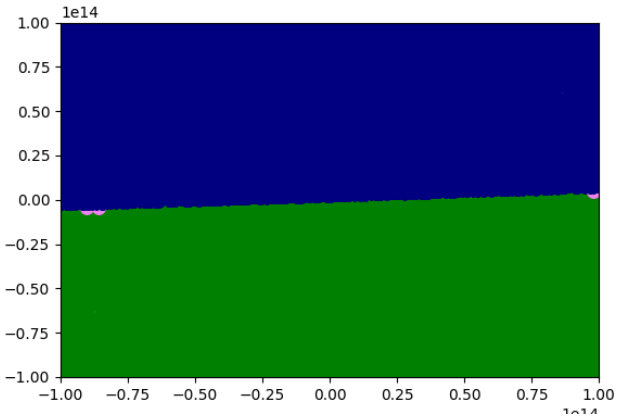
Rys.6 Podział punktów z zestawu B – wyznaczniki 3x3.
Po lewej: 50051, współliniowe: 0, po prawej: 49949



Rys.7 Podział punktów z zestawu B – wyznacznik 2x2 numpy.
Po lewej: 50053, współliniowe: 0, po prawej: 49947



Rys.8 Podział punktów z zestawu B – wyznacznik 2x2 własny.
Po lewej: 50050, współliniowe: 3, po prawej: 49947



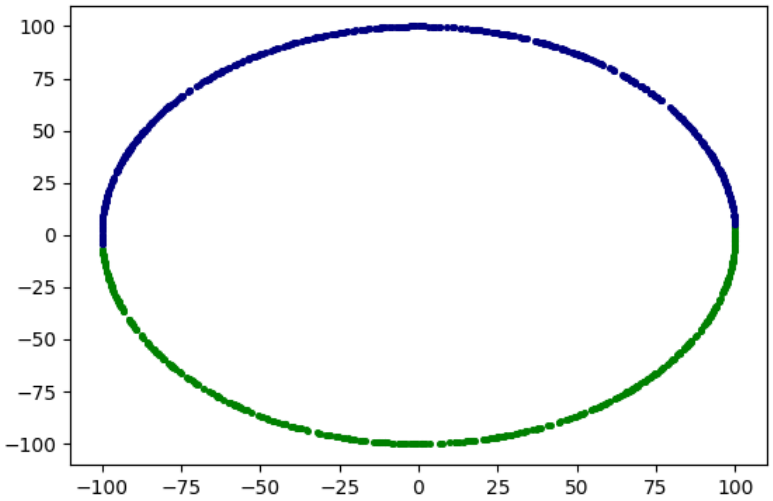
W przypadku obliczeń przy pomocy własnej implementacji wyznacznika 2x2 3 punkty zostały skategoryzowane jako współliniowe. Warto zwrócić uwagę, że są to punkty leżące blisko granic zakresu, który wynosi $[-10^{14};\ 10^{14}]$. Dla tak dużych liczb precyzja obliczeń spada, czego skutkiem są niedokładności.

4.3 Dane C

Trzeci zestaw danych prezentuje takie same wyniki dla każdej metody liczenia wyznacznika. Punkty losowane są na małym przedziale, a przez ich dystrybucję na okręgu, potencjalnych punktów współliniowych jest niewiele.

Rys.9 Podział punktów z zestawu C.

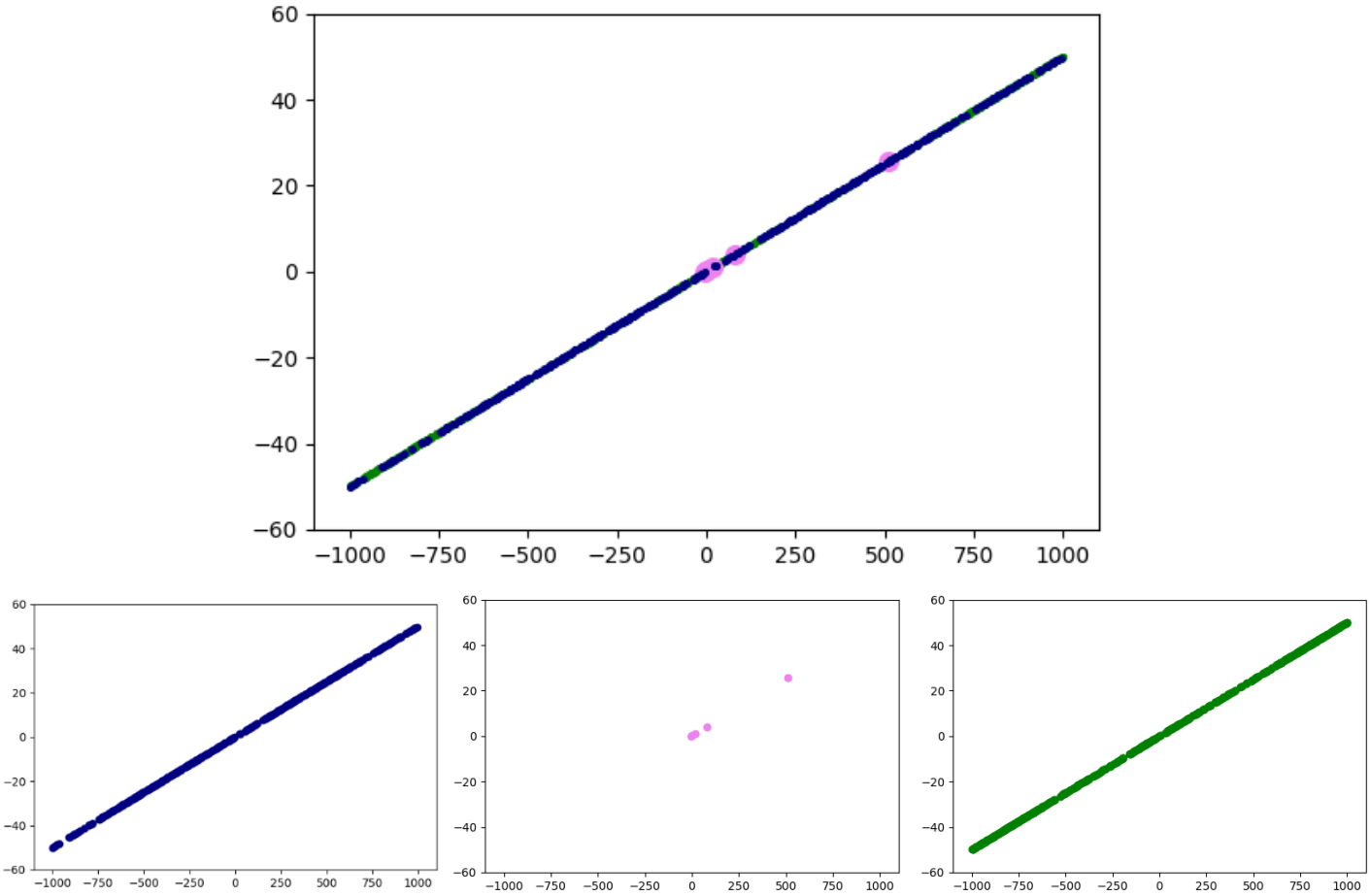
Po lewej: 529, współliniowe: 0, po prawej:471



4.4 Dane D

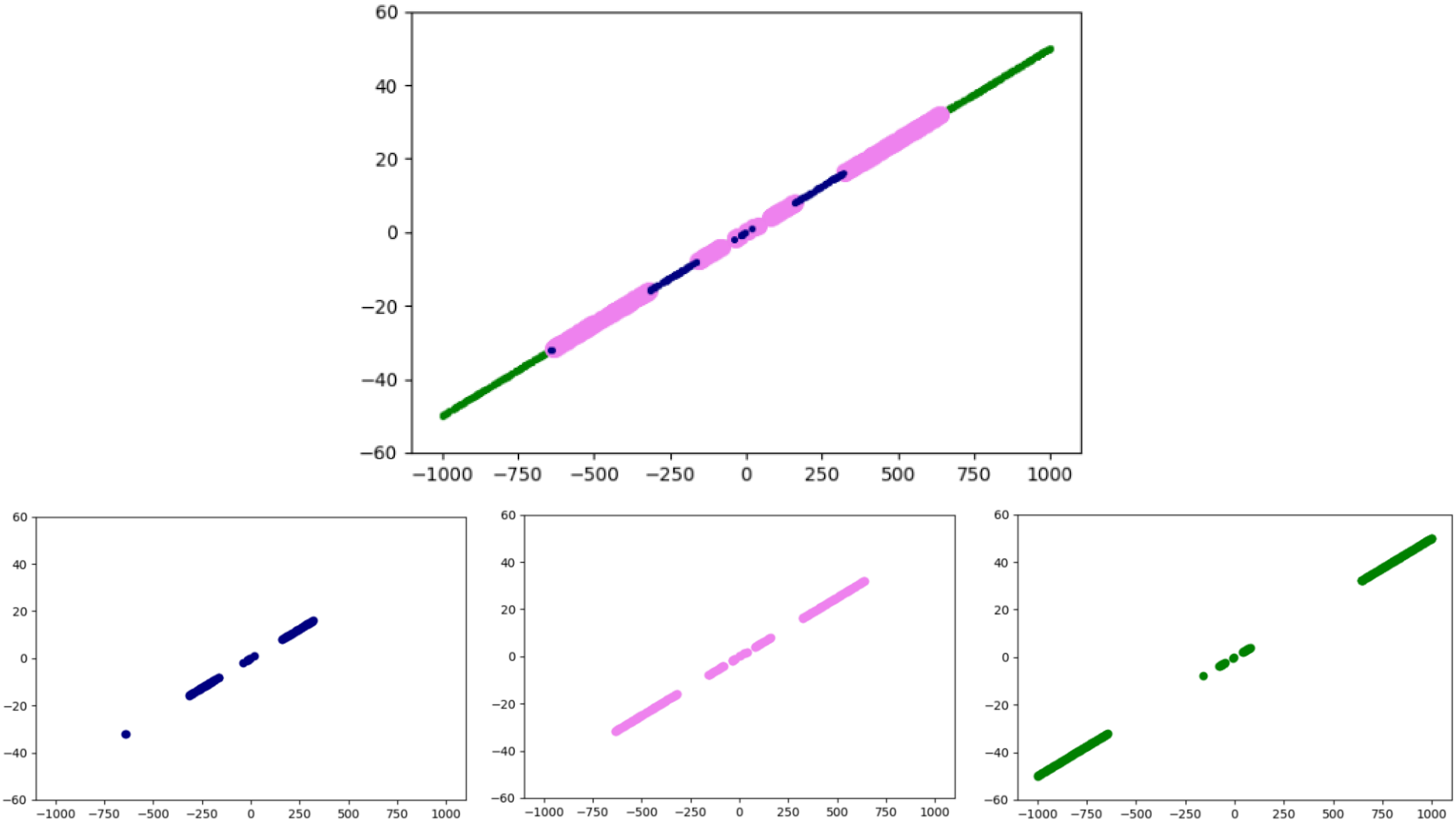
Najwięcej różnic w kategoryzacji punktów w zależności od zastosowanego wyznacznika pojawia się w zestawie czwartym. Błędy spowodowane są skończoną precyzją obliczeń połączoną z małą tolerancją dla zera.

Rys.10 Podział punktów z zestawu D – wyznacznik 3x3 numpy.
Po lewej: 449, współliniowe: 5, po prawej: 546



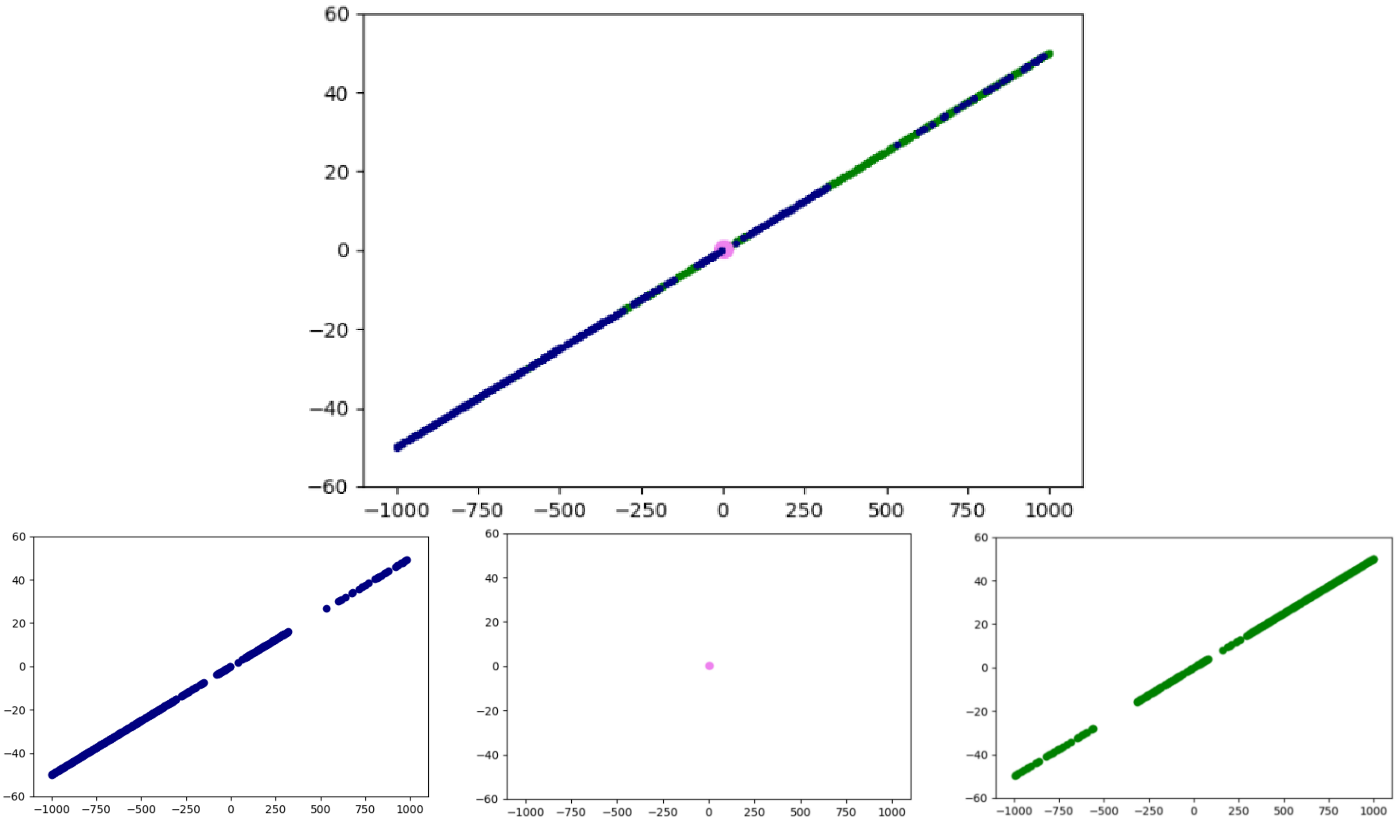
Rys.11 Podział punktów z zestawu D – wyznacznik 3x3 .

Po lewej: 160, współliniowe: 400, po prawej: 440



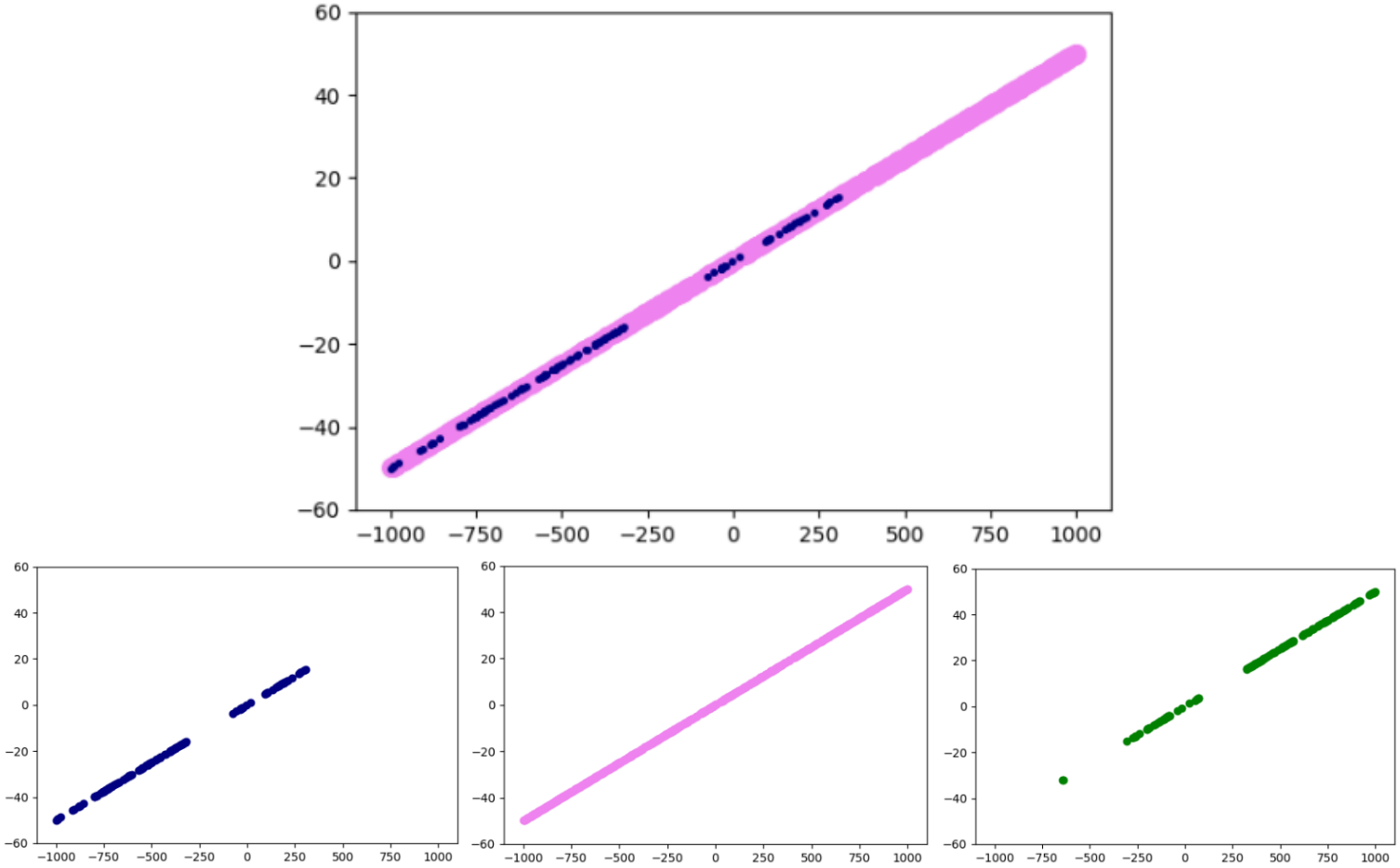
Rys.12 Podział punktów z zestawu D – wyznacznik 2x2 numpy.

Po lewej: 499, współliniowe: 2, po prawej: 499



Rys.13 Podział punktów z zestawu D – wyznacznik 2x2 własny.

Po lewej: 133, współliniowe: 711, po prawej: 156



Jak widać na wykresach najbardziej precyzyjny okazuje się być wyznacznik 2x2 własnej implementacji. Wyznaczniki z biblioteki są w stanie poprawnie skategoryzować pojedyncze punkty leżące blisko 0. Warto zwrócić uwagę na symetrię środkową, która występuje podczas liczenia wyznacznikiem 3x3 własnej implementacji. Punkty o zbliżonych współrzędnych trafiają do tych samych kategorii, co obrazuje niedokładności obliczeniowe dla danych przedziałów liczbowych. Podobne obserwacje można przeprowadzić w wyznacznikach 2x2, z tą różnicą, że punkty skategoryzowane, jako te po lewej stanowią przybliżone „odbicie” tych umieszczonych po prawej.

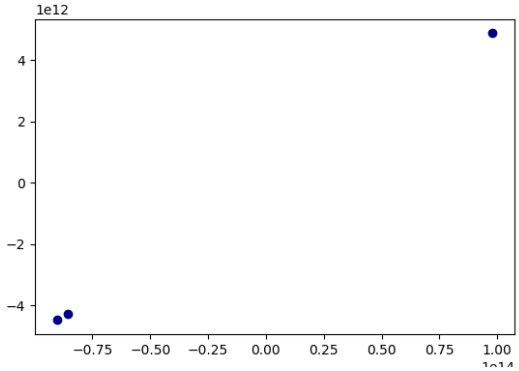
5. Wizualizacja różnic pomiędzy klasyfikacją punktów

Kolejnym etapem ćwiczenia jest sprawdzenie ilości różnic pomiędzy sklasyfikowanymi zbiorami punktów i przedstawienie tych różnic w formie graficznej. Posłuży do tego funkcja, która porównuje ze sobą skategoryzowane zbiory punktów prezentuje różnice na wykresie.
Jak zostało zaznaczone wcześniej, dla danych A i C nie zaobserwowano różnic, dlatego w tym punkcie skupię się na zestawach B i D.

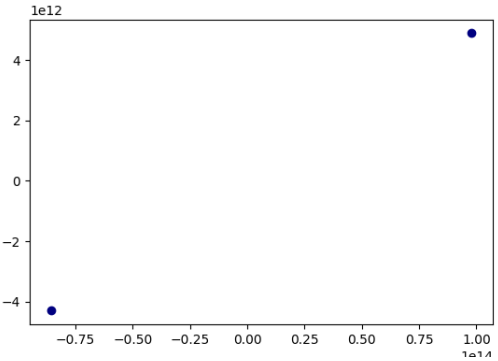
5.1 Dane B

W przypadku wyznaczników 3x3 (numpy, a własny) nie zaobserwowano różnic. Te pojawiają się dla wyznaczników 2x2.

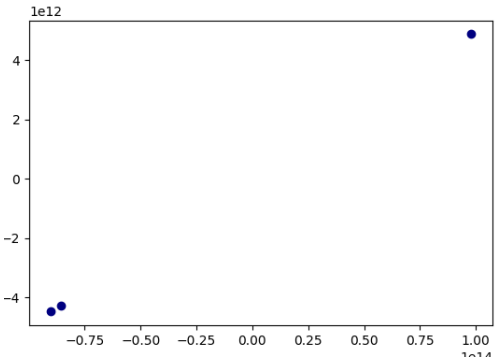
Rys.14 Dane B – różnice między wyznacznikiem 2x2 numpy, a 2x2 własnym
Ilość różnic: 3



Rys.15 Dane B – różnice między wyznacznikiem 2x2 numpy, a 3x3 numpy
Ilość różnic: 2



Rys.16 Dane B – różnice między wyznacznikiem 2x2 własnym, a 3x3 własnym
Ilość różnic: 3

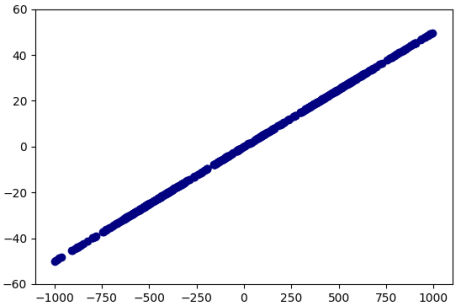


Powyższe wykresy pokazuję, że jedyne różnice występują dla wartości granicznych zadanego przedziału. W praktyce operacje na dużych liczbach, jak w tym przykładzie, powodują problemy z dokładnością.

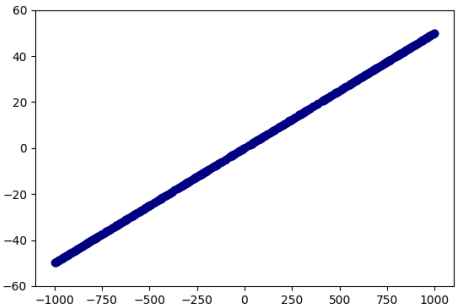
5.2 Dane D

Najwięcej różnic widocznych jest w czwartym zestawie punktów. Pojawiają się one na całej długości. Niezależnie od porównywanych wyznaczników punkty pokrywające się stanowią mniejszość, pokazuje to skalę rozbieżności pomiędzy poszczególnymi funkcjami wyznaczników.

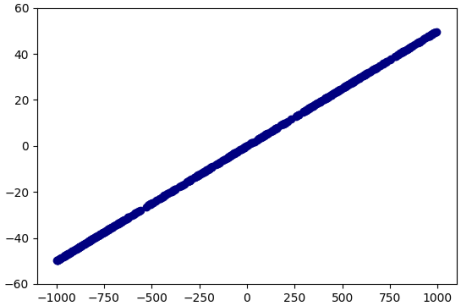
Rys.17 Dane D – różnice między wyznacznikiem 3x3 własnym, a 3x3 numpy
Ilość różnic: 542



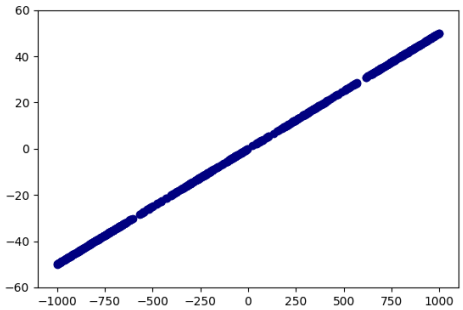
Rys.18 Dane D – różnice między wyznacznikiem 2x2 własnym, a 2x2 numpy
Ilość różnic: 736



Rys.19 Dane D – różnice między wyznacznikiem 3x3 numpy, a 2x2 numpy
Ilość różnic: 512



Rys.20 Dane D – różnice między wyznacznikiem 3x3 własnym, a 2x2 własnym
Ilość różnic: 678



6. Tabele wyników dla różnych tolerancji

Tabela 2. Wyniki testów dla zestawu danych A

Tolerancja	Wyznacznik	Lewo	Wzdłuż	Prawo
Dla wszystkich testów uzyskano takie same wyniki		50385	0	49615

Tabela 3. Wyniki testów dla zestawu danych B

Tolerancja	Wyznacznik	Lewo	Wzdłuż	Prawo
Niezależnie od przyjętej tolerancji uzyskano takie same wyniki	2x2 własny	50050	3	49947
	2x2 numpy	50053	0	49947
	3x3 własny	50051	0	49949
	3x3 numpy	50051	0	49949

Tabela 4. Wyniki testów dla zestawu danych C

Tolerancja	Wyznacznik	Lewo	Wzdłuż	Prawo
Dla wszystkich testów uzyskano takie same wyniki		529	0	471

Tabela 5. Wyniki testów dla zestawu danych D

Tolerancja	Wyznacznik	Lewo	Wzdłuż	Prawo
1e-17	2x2 własny	133	711	156
	2x2 numpy	499	2	499
	3x3 własny	160	400	440
	3x3 numpy	546	5	449
1e-14	2x2 własny	126	720	154
	2x2 numpy	451	86	463
	3x3 własny	0	1000	0
	3x3 numpy	12	863	125
1e-11	2x2 własny	0	1000	0
	2x2 numpy	0	1000	0
	3x3 własny	0	1000	0
	3x3 numpy	0	1000	0

7. Wydajność metod liczenia wyznacznika

Kolejnym istotnym elementem, obok dokładności metody obliczania wyznacznika jest jej wydajność. Poniżej przedstawiona jest tabela, w której prezentowane są średnie czasy działania funkcji kategoryzacji, zależnie od wykorzystanej metody obliczania wyznacznika. Uśrednione zostały czasy wykonania funkcji dla różnych tolerancji – dla każdej tolerancji od nowa obliczana była tablica wyznaczników. Jedynymi różnicami pomiędzy funkcjami kategoryzacji był sposób liczenia wyznacznika, więc różnica czasów wykonania tych funkcji jest bezpośrednio od niego zależna. Do obliczenia czasu wykonywania się programu użyta została funkcja time zaimportowana z biblioteki time.

Tabela 6. Czasy działania funkcji kategoryzacji zależnie od metody obliczania wyznacznika

Zbiór danych	Wyznacznik	Średni czas działania [s]
Dane A (10^5 punktów)	2x2 własny	0.2755
	2x2 numpy	0.6049
	3x3 własny	0.5493
	3x3 numpy	0.6842
Dane B (10^5 punktów)	2x2 własny	0.2714
	2x2 numpy	0.5955
	3x3 własny	0.5301
	3x3 numpy	0.6629
Dane C (1000 punktów)	2x2 własny	0.0030
	2x2 numpy	0.0060
	3x3 własny	0.0057
	3x3 numpy	0.0065
Dane D (1000 punktów)	2x2 własny	0.0026
	2x2 numpy	0.0060
	3x3 własny	0.0050
	3x3 numpy	0.0063

Najszybszy okazuje się być wyznacznik 2x2 implementacji własnej. Jest on w praktyce dwa razy wydajniejszy, niż pozostałe. Kolejnym pod względem wydajności jest wyznacznik 3x3 implementacji własnej. Wyznaczniki z biblioteki nie różnią się znacząco między sobą pod względem czasu wykonania. Jako, że są przystosowane do działania na n-wymiarowych macierzach, to ich wydajność jest gorsza, niż tych z implementacji własnej. Sytuacja prezentuje się analogicznie w kwestii dokładności.

8. Wnioski i podsumowanie

Po analizie wyników powyższych testów bez problemu widać, jak duże różnice mogą się pojawić w zależności od przyjętej metodologii liczenia wyznacznika i jak istotne jest dobranie odpowiedniej metody i tolerancji do badanych zagadnień. Problemy w poprawnym określeniu położenia punktu pojawiają się, gdy operuje się na liczbach z dużego zakresu (tak jak w danych B), bądź gdy punkty są współliniowe, lecz precyzja obliczeń nie jest wystarczająca, by tą współliniowość uznać. W takich wypadkach konieczne jest dobranie odpowiedniego algorytmu liczenia wyznacznika, jak i odpowiedniej tolerancji dla zera.

Powyższe wyniki pokazuję, że używanie wyznaczników z biblioteki numpy jest nieefektywne. W przypadkach granicznych ich dokładność znacząco odbiega od własnych implementacji, a w dodatku ich obliczanie jest bardziej wymagające dla procesora.

Wyznacznik 2x2 własnej implementacji sprawdza się dosyć dobrze, gdy konieczne jest przypisanie punktów z niewielką tolerancją dla zera – to on osiągną najwyższy wynik dla tolerancji 1e-17, gdzie wyznacznik 3x3 okazał się prawie dwa razy gorszy. Jednakże w przypadku wyznacznika 2x2 mogą występować błędy dla dużych zakresów liczb (tak jak w przypadku danych B).

Jeżeli możliwe jest zmniejszenie tolerancji, to najlepszym wyborem okazuje się być wyznacznik 3x3 implementacji własnej. Chociaż nie jest najwydajniejszy, to był w stanie wskazać poprawnie wszystkie punkty dla tolerancji 1e-14 w zestawie danych D, a także nie zwracał punktów współliniowych dla zestawu B. Najmniejsza tolerancja, która pozwala poprawnie sklasyfikować wszystkie punkty z zestawu D dla każdego wyznacznika to 1e-11.

Warto też zaznaczyć, że są przypadki w których wybór metody obliczania wyznacznika nie będzie miał większego znaczenia. Zauważyć to można na przykładzie danych A i C. Punkty które są rozproszone na małym zakresie i które nie są gęsto zgromadzone w obszarze bliskim zadanej linii nie będą stwarzały problemów związanych z precyzją obliczeń (a przynajmniej jest to wysoce nieprawdopodobne, aby tak się stało). W takich wypadkach najzasadniejsze wydaje się kierowanie wydajnością.