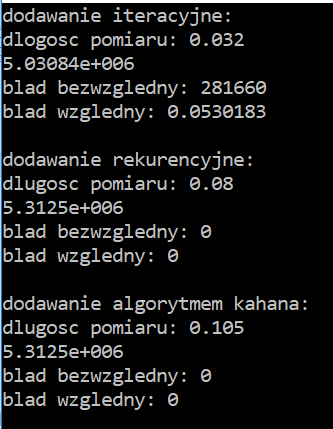
**Sprawozdanie**

Mownit – lab1

Mateusz Naróg

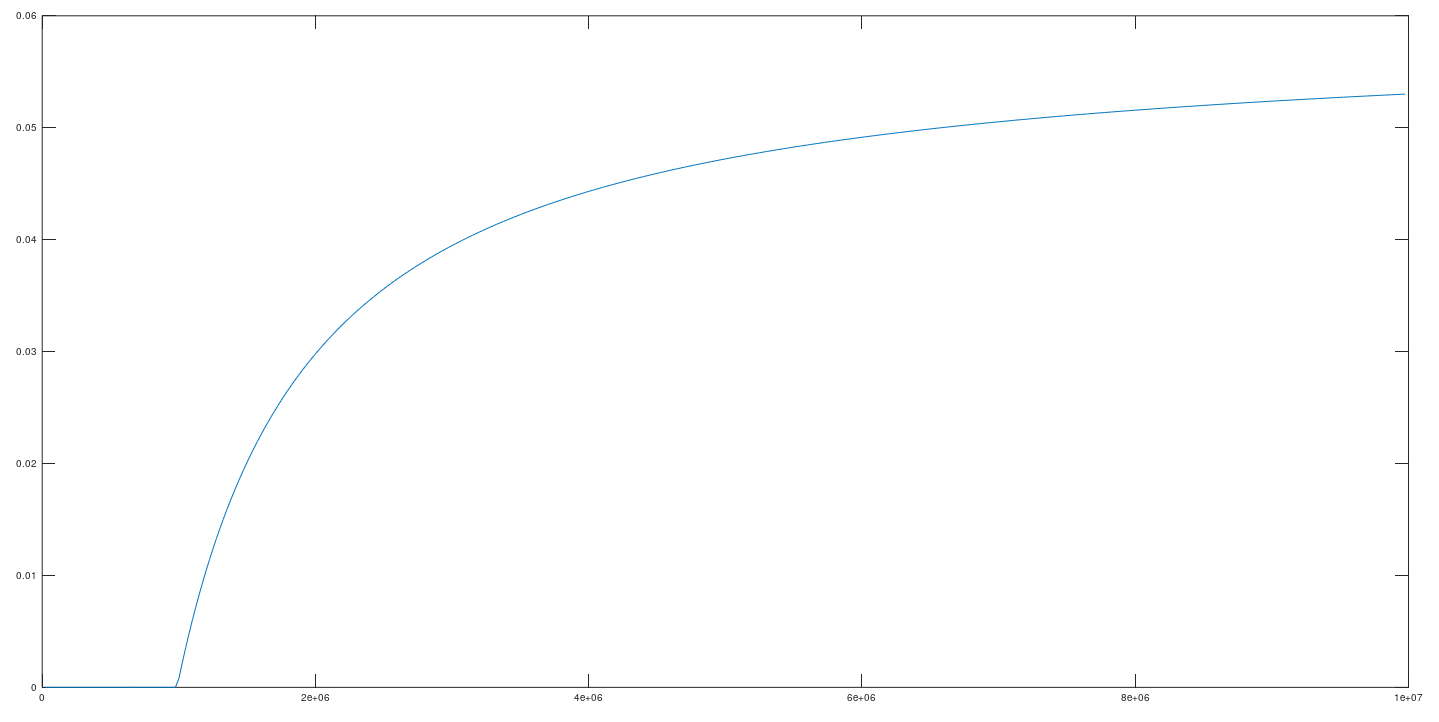
# Zadanie 1 i 2

**Podsumowanie 3 algorytmów dodających:**



Jak widać najszybsze jest dodawanie iteracyjne, jednak okazało się ono również najmniej dokładne. Wynika to z faktu, iż po pewnej liczbie iteracji będziemy dodawać małą liczbę do stosunkowo dużej liczby, przez co będzie ucinana końcówka dodawania (ze względu na skończoną pamięć przechowywująca wartości).

**Wzrost wartości błędów względnych:**



Zgodnie z wcześniejszymi przypuszczeniami obserwujemy wzrost wartości błędu względnego dopiero od pewnej liczbie iteracji. Dodatkowo możemy zauważyć, że wraz ze wzrostem liczby iteracji, tempo wzrostu wartości błędu względnego maleje. Dzieje się tak ze względu na to, że dla coraz większej liczby iteracji, wynik dodawania jest coraz większy i wpływ małej liczby staje się coraz mniejszy.

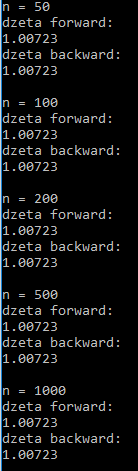
**Dane, dla których sumowanie rekurencyjne zwraca niezerowy błąd**

W przypadku, gdy cała tablica była wypełniona tylko jedną wartością nie było problemu przy sumowaniu 2 sąsiednich liczb, ponieważ nie trzeba było „ucinać” końcówki. Natomiast wystarczy, że damy na przemian jedną dużą liczbą i jedną mała. Wtedy sumowanie rekurencyjne powinno zwrócić błąd. Oczywiście nie jest to jedyny możliwy układ danych. Wystarczy, że na dowolnym etapie rekurencji będziemy dodawać do siebie liczby o dużej różnicy między sobą.

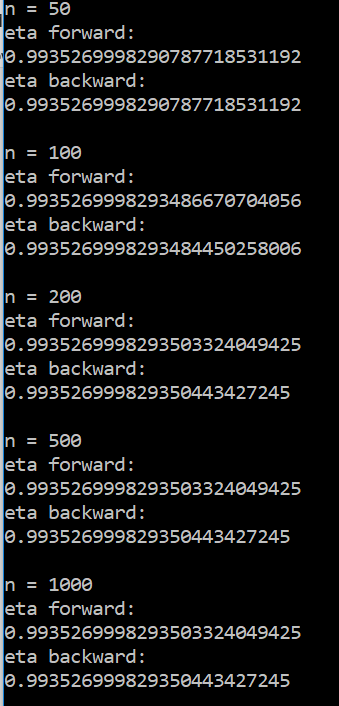
**Algorytm Kahana**

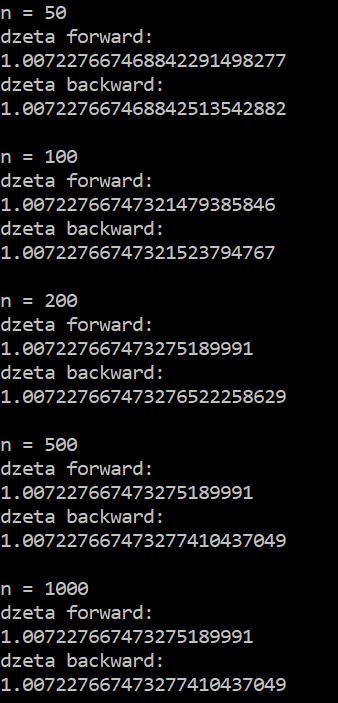
Lepsze własności numeryczne w algorytmie Kahana wynikają z uwzględnienia „ucinanej” końcówki, która wpływała dość znacząco na wynik w przypadku sumowania iteracyjnego. Jest za to odpowiedzialna zmienna err, która właśnie przechowuje tą „uciętą” końcówkę i uwzględnia ją w następnej iteracji.

# Zadanie 3



Dla pojedynczej precyzji otrzymano takie same wyniki sumując od przodu jak i od tyłu.



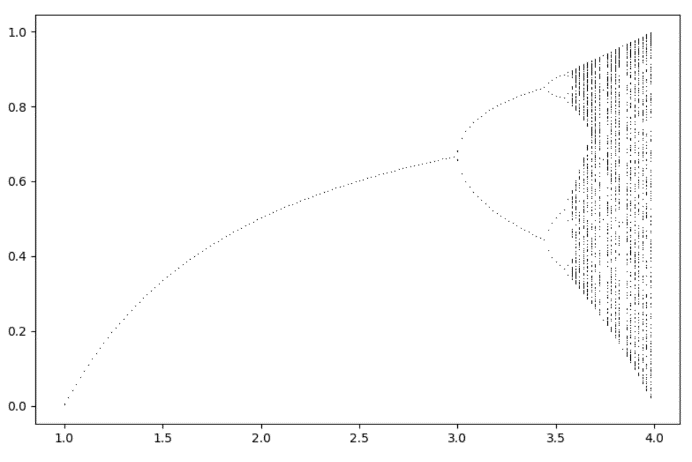
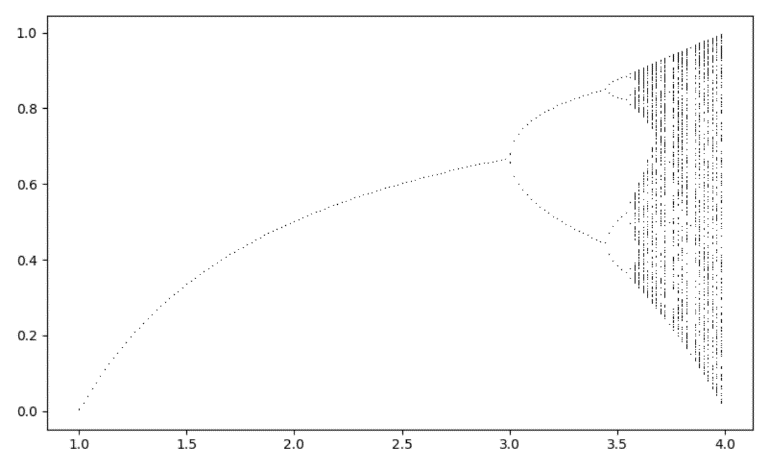
W przypadku podwójnej precyzji różnica w wynikach była zauważalna dopiero po ustawieniu większej precyzji wyświetlania

Jak widać różnice są widoczne dopiero na dość odległych miejscach po przecinku

Różnice w wyniku wynikające ze sposobu iteracji (w przód oraz wstecz) prawdopodobnie wynikają z faktu, iż dodawanie małej liczby do dużej sumy nie będzie na tyle dokładne, jak w przypadku gdy na początku będziemy dodawać małe liczby do siebie, a wartość kolejnych będzie rosła. W tym przypadku ta dokładność powinna być większa. Zatem to zależy od sposobu ułożenia liczb.

# Zadanie 4

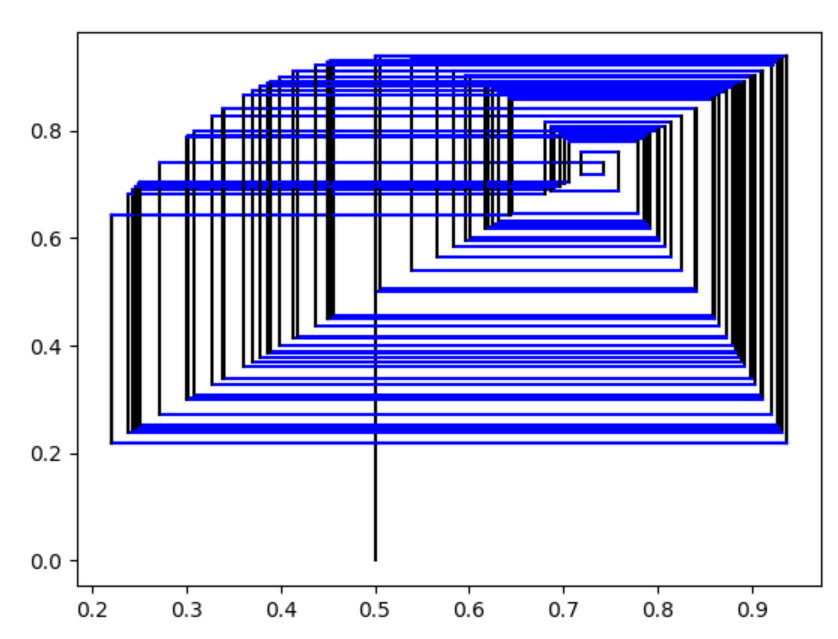
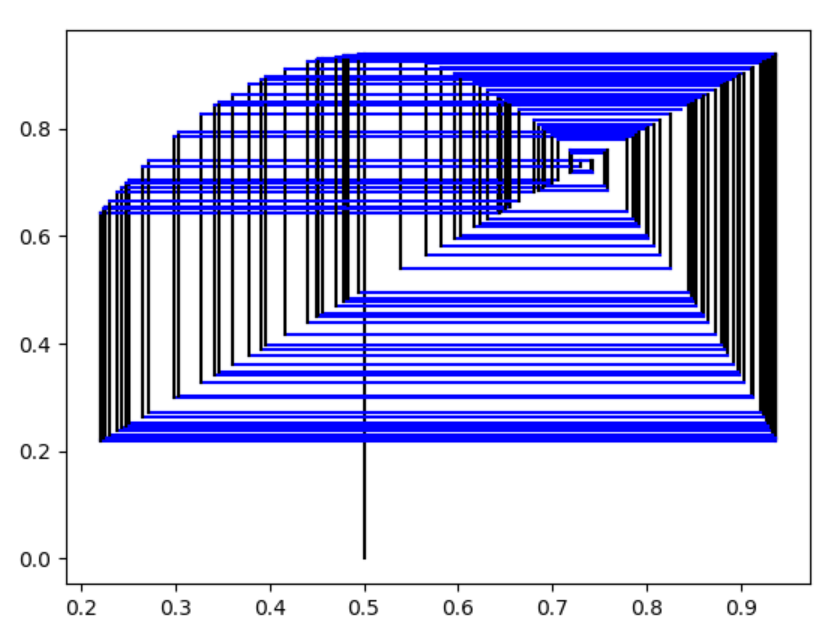
**Podpunkt a**

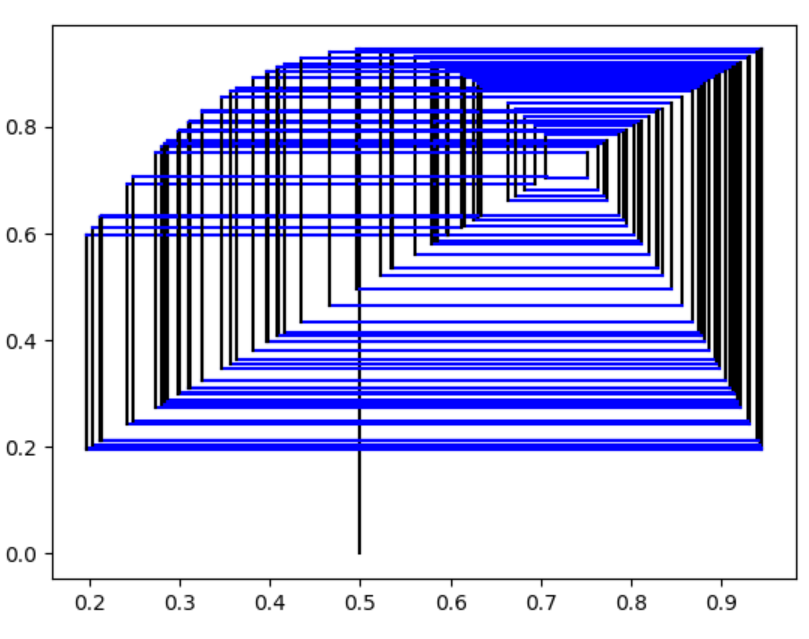
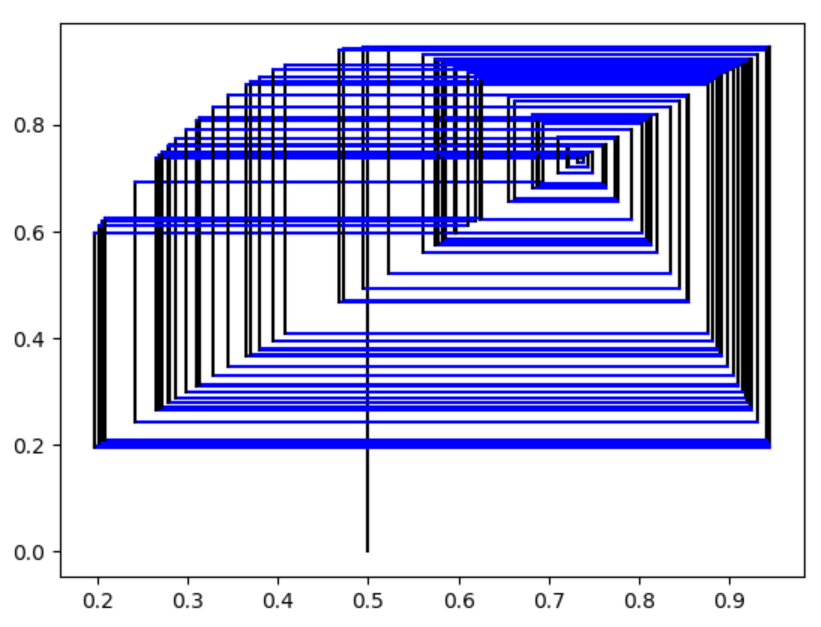
x0=0.5 x0=0.8543

Możemy zauważyć, że niezależnie od wyboru x0 te wykresy są podobne. Kolejnym faktem, którym można zauważyć jest to, że rozdwojenie następuje dla r ≈ 3. Zatem możemy wnioskować, że duży wpływ na wygląd tego wykresu ma właśnie wartość r.

**Podpunkt b**

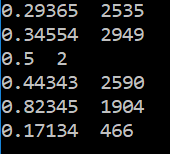
Po lewej stronie znajdują się trajektorie z pojedynczą precyzją, natomiast po prawej z podwójną

r=3.75

r=3.78

Na pierwszy rzut oka można powiedzieć, że trajektorie dla danego r są podobne, jednak po dłuższej analizie można stwierdzić, że w przypadku podwójnej precyzji te wykresy są bardziej równomierne, natomiast dla pojedynczej precyzji możemy zauważyć w pewnych miejscach większe zagęszczenie.

**Podpunkt c**



Po lewej stronie podana jest wartość x0, natomiast po prawej liczba iteracji potrzebnych do osiągnięcia przez x wartości 0.  
Ciężko doszukać się pewnego schematu, dzięki któremu można byłby przewidywać przybliżoną liczbę iteracji. Dodatkowo dla pewnych wartości x0 jak np. 0.8, liczba iteracji przekraczała 25000, a wartość x dalej nie osiągnęła 0. Ciężko zatem stwierdzić czy dla pewnych liczb początkowych wartość 0 jest nieosiągalna, czy potrzebują znacznie większej liczby iteracji.