Wiktor Garbarek

Pracownia nr 1 z Analizy Numerycznej

Sprawozdanie do zadania **P1.1**

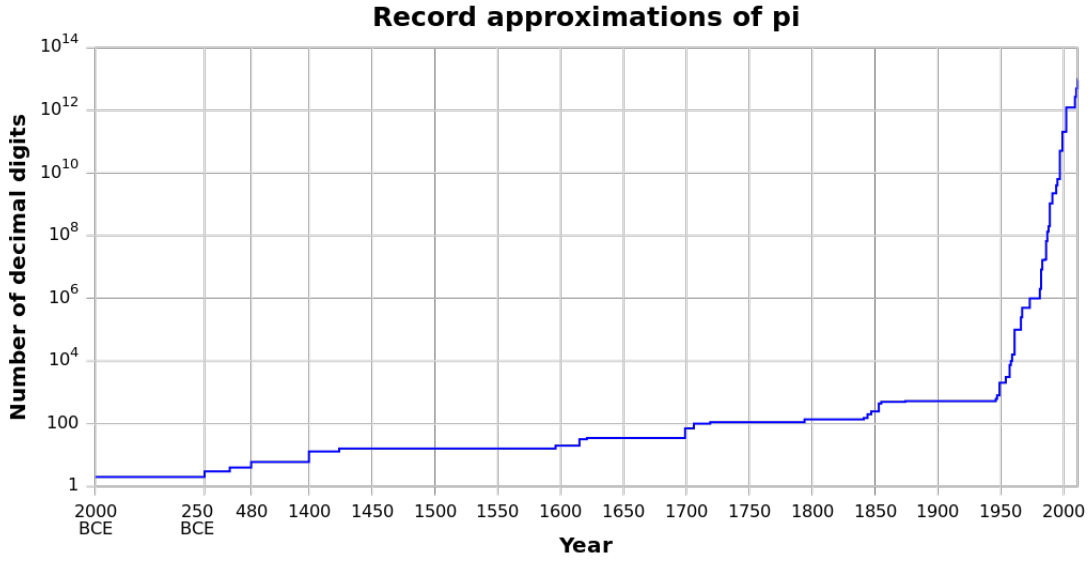
Wrocław, Listopad 2017

1. Parę słów o liczbie π i jej przybliżeniach w przeszłości.

Każdy z nas słyszał o liczbie π. Ta magiczna stała, o której dowiadujemy się już w szkole podstawowej, definiowana jako stosunek obwodu okręgu do długości jego średnicy, jest równa $\pi = 3.14159....$ Pojawia się w każdej nauce ścisłej, czasami w bardzo nieoczekiwanych miejscach co tylko potęguje potrzebę posiadania dobrej metody znajdowania jej przybliżenia. Lecz co pojawia się w miejscu wielokropka? Co więcej możemy powiedzieć o niej samej i jej własnościach, które być może pomogą nam (lub utrudnią) znajdowanie przybliżeń? Jak znaleźć dalsze cyfry po przecinku przybliżenia liczby π? I skąd będziemy wiedzieć, że one są poprawne?

Problem przybliżenia liczby π, czy to do celów inżynierskich, czy z ludzkiej ciekawości, towarzyszył matematykom od zarania dziejów. Już w starożytnym Babilonie wiadomo było, że π jest stałą oraz $\pi \approx 3$, ponad tysiąc lat później **Archimedes** oszacował π jako $\frac{223}{71} < \pi < \frac{22}{7}$ wykorzystując obwody wielokątów foremnych opisanych i wpisanych na danym okręgu, a już na początku naszej ery Chińscy matematycy pokazali, że $\pi \approx \frac{355}{113}$.

W XVI wieku, **Ludolph van Ceulen** obliczył π z dokładnością do 35 cyfr po przecinku wykorzystując 2^{62}-kąt foremny.[1] Później jeszcze pod koniec XVIII wieku Jurij Vega, słoweński matematyk, znalazł 136 początkowych cyfr dziesiętnych przybliżenia liczby π, a w następnym stuleciu, angielski matematyk William Shanks otrzymał ponad 500 cyfr dokładnych rozwinięcia dziesiętnego ludolfiny. Oczywiście z powodu rozwoju technologii, jak i samej matematyki, kolejni amatorzy i entuzjaści znajdowali kolejne cyfry dokładne naszej bohaterki, więc moglibyśmy tak wyliczać jeszcze długo. Wartym jednak zanotowania jest, że z tego co autorowi wiadomo, dzisiaj znamy $22,459,157,718,361$ cyfr dziesiętnych rozwinięcia liczby π za sprawą Petera Trueba. Wynik ten został otrzymany za pomocą algorytmu Chudnovsky'ego.



Wykres 1: Wykres rekordów dokładności przybliżeń liczby π w zależności od roku - źródło: *wikipedia.org*

[1] Stąd też liczbę π nazywamy też *ludolfiną*.

2. O problemach związanych ze liczbą π.

Okazuje się, że bohaterka tego akapitu jest bardzo problematyczna jeśli chodzi o jej własności. Gdyby π było wymierne, to jej rozwinięcie dziesiętne byłoby okresowe bądź skończone i problem byłby zdecydowanie prostszy. Jednak jak wszyscy wiemy - liczba π jest niewymierna.[2] Wbrew pozorom nie jest to fakt taki trywialny do udowodnienia - od dawna matematycy przewidywali, że liczba π jest niewymierna, ale przekonujący dowód dał dopiero **Johann Heinrich Lambert** w 1761 roku wykorzystując własności pewnego rozwinięcia funkcji $\tan(x)$.

Oprócz tego pojawia się jeszcze jeden mały problem - *przestępność*. W skrócie można powiedzieć, że liczba π jest "niewymierna w inny sposób" niż takie $\sqrt{2}$ czy $\sqrt[3]{5}$. Te dwie wspomniane liczby, jak łatwo zauważyć, są pierwiastkami wielomianów o współczynnikach wymiernych (Odpowiednio: x^2-2 oraz x^3-5). A co z liczbą π? Okazuje się, że dla liczby π nigdy nie znajdziemy wielomianu o współczynnikach wymiernych, dla którego $w(\pi)=0$. Dowód tego faktu przedstawił **Ferdinand von Lindemann** relatywnie niedawno, bowiem w 1882 roku. To sprawia, że nie możemy znaleźć przybliżenia π poprzez szukanie pierwiastka jakiegoś wielomianu - chociażby metodą bisekcji, metodą Newtona czy bardziej zaawansowaną metodą, która dałaby nam jeszcze szybszą zbieżność do π. Pojawia się więc problem: jak tego przybliżenia można szukać? Pierwsze co nasuwa się na myśl to fakt, że można wykorzystać jakąś formułę rekurencyjną wzoru opartego na geometrycznej interpretacji liczby π (tak zrobimy w tej pracy). Drugim pomysłem, już bardziej wysublimowanym, byłoby znalezienie przybliżenia odpowiedniej sumy bądź sumy częściowej jakiegoś szeregu zbieżnego do π.[3]

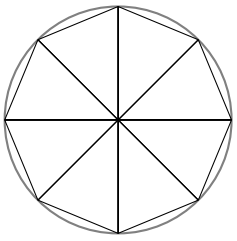
3. Podstawowe twierdzenia, wykorzystane wzory i opis zadania.

Będziemy szukać przybliżenia liczby π częściowo wykorzystując pomysł Archimedesa - nie wykorzystamy jednak obwodu, a pola wielokątów foremnych. Dokładniej mówiąc, wykorzystamy wzór na pole 2^{n+1}-kąta foremnego oraz formuły rekurencyjne przedstawione w następnym rozdziale, które ten wzór spełnia. Wykażemy, że wzór ten rzeczywiście spełnia te rekurencje oraz powiemy jakie problemy od strony numerycznej niesie ze sobą wykorzystanie tych wzorów. Oszacujemy też błędy powstające przy każdej iteracji, powstałe przez niedokładność reprezentacji liczby jako słowo maszynowe. Wszelkie testy numeryczne wykonamy za pomocą programu napisanego w języku *Julia* wykorzystując arytmetyki pojedynczej oraz podwójnej precyzji (W Julii reprezentowane odpowiednio przez typy $Float32$ oraz $Float64$). Na koniec spróbujemy odpowiedzieć sobie, jak szybko ta metoda jest zbieżna do szukanego przez nas przybliżenia π.

Twierdzenie 1. *Wzór na pole 2^{n+1}-kąta foremnego wpisanego w okrąg o promieniu 1 wyraża się wzorem*

$$P_{2^{n+1}} = 2^n \sin\frac{\pi}{2^n}.$$

$n=8$



Dowód. Zauważmy, że nasz 2^{n+1}-kąt możemy podzielić na tyle samo trójkątowo wspólnym wierzchołku w środku okręgu tak jak na rysunku obok. Każdy z tych trójkątów ma pole równe $\frac{1\cdot 1\cdot \sin\alpha}{2}$. Gdzie $\alpha = 2\pi/2^{n+1}$. W takim razie

$$P_{2^{n+1}} = 2^{n+1}\cdot\frac{1\cdot 1\cdot\sin\frac{\pi}{2^n}}{2} = 2^n\sin\frac{\pi}{2^n}$$

□

Łatwo zauważyć, że wtedy ciąg $P_{2^{n+1}}$ zbiega do pola okręgu o promieniu 1, którego pole jest równe szukanej liczbie π. Żeby jednak nie machać rękami, udowodnijmy sobie to dokładnie.

[2] Chociaż pewnego razu matematyk-amator - Edward J. Goodwin w 1897 chciał prawnie ustanowić, że $\pi = 3.2$

[3] A takich szeregów czy rozwinięć w sumy jest naprawdę dużo! Od znanej wszystkim tożsamości Eulera dla sumy odwrotności kwadratów kolejnych liczb naturalnych, którą moglibyśmy pomnożyć przez 6, a później wziąć pierwiastek z wyniku (okazuje się jednak, że nie jest to zbyt efektywna metoda), po pewne sumy wykorzystujące formuły Machina, które są bardzo szybko zbieżne.

Twierdzenie 2.

$$\lim_{n\to\infty} 2^n \sin\frac{\pi}{2^n} = \pi$$

Dowód. Łatwo zauważyć, że skoro $\lim_{x\to 0}\frac{\sin x}{x} = \lim_{x\to 0}\frac{\cos x}{1} = 1$, to

$$\lim_{n\to\infty} 2^n \sin\frac{\pi}{2^n} = \lim_{n\to\infty} \pi\cdot\frac{\sin\frac{\pi}{2^n}}{\frac{\pi}{2^n}} = \pi\cdot\lim_{x\to 0}\frac{\sin x}{x} = \pi$$

bowiem $\frac{\pi}{2^n} \overset{n\to\infty}{\to} 0$ □

Udowodnijmy też jedną prostą tożsamość trygonometryczną, którą wykorzystamy przy dowodzie wzorów rekurencyjnych.

Lemat 1.

$$2\sin^2\frac{x}{2} = 1 - \cos x$$

Dowód. Wiemy, że $cos(2x) = \cos^2 x - \sin^2 x = (\cos^2 x + \sin^2 x) - 2\sin^2 x = 1 - 2\sin^2 x$. W takim razie $2\sin^2 x = 1 - \cos(2x)$. Po położeniu $x := x/2$ otrzymujemy wprost tezę. □

4. Dowody wzorów rekurencyjnych.

Twierdzenie 3. *Gdy ciąg x_k jest dany wzorem ogólnym $x_k := 2^k \sin\frac{\pi}{2^k}$, to spełnia on nastpujące równanie rekurencyjne dla warunku początkowego $x_1 = 2$*

$$x_{k+1} = 2^k\sqrt{2\Big(1-\sqrt{1-(x_k/2^k)^2}\Big)} \tag{1}$$

Dowód. Przeprowadzimy dowód indukcyjny względem k. Baza indukcji jest prosta do pokazania, bowiem $x_1 = 2^1\sin(\pi/2^1) = 2$. Załóżmy więc, że wzór (1) jest prawdziwy dla x_k, udowodnimy go dla x_{k+1}. Uprośćmy prawą stronę równania (1) wykorzystując założenie indukcyjne oraz Lemat 1:

$$2^k\sqrt{2\left(1-\sqrt{1-\left(\frac{2^k\sin(\frac{\pi}{2^k})}{2^k}\right)^2}\right)} = 2^k\sqrt{2\left(1-\sqrt{1-\sin^2\left(\frac{\pi}{2^k}\right)}\right)} =$$

$$= 2^k\sqrt{2\Big(1-\cos\frac{\pi}{2^k}\Big)} = 2^k\sqrt{4\sin^2\frac{\pi}{2^{k+1}}} = 2^{k+1}\sqrt{\sin^2\frac{\pi}{2^{k+1}}}$$

a skoro $0 < \frac{\pi}{2^{k+1}} < \pi$ dla każdego $k > 0$ to wnioskujemy

$$\sqrt{\sin^2\frac{\pi}{2^{k+1}}} = \sin\frac{\pi}{2^{k+1}}$$

czyli prawa strona to $2^{k+1}\sin\frac{\pi}{2^{k+1}} = x_{k+1}$, co kończy dowód. □

Twierdzenie 4. *Gdy ciąg x_k jest dany wzorem ogólnym $x_k := 2^k \sin\frac{\pi}{2^k}$, to spełnia on nastpujące równanie rekurencyjne dla warunku początkowego $x_1 = 2$*

$$x_{k+1} = \frac{2x_k}{\sqrt{2\big(1+\sqrt{1-(x_k/2^k)^2}\big)}} \tag{2}$$

Dowód. Wykorzystamy twierdzenie 1. Zauważmy:

$$x_{k+1} = 2^k\sqrt{2(1-\sqrt{1-(x_k/2^k)^2})}\cdot\frac{\sqrt{1+\sqrt{1-(x_k/2^k)^2}}}{\sqrt{1+\sqrt{1-(x_k/2^k)^2}}} =$$

$$= 2^k\cdot\sqrt{2}\cdot\frac{\sqrt{1^2-\sqrt{1-(x_k/2^k)^2}^2}}{\sqrt{1+\sqrt{1-(x_k/2^k)^2}}} = 2^k\cdot\frac{2}{\sqrt{2}}\cdot\frac{\sqrt{x_k/2^k}^2}{\sqrt{1+\sqrt{1-(x_k/2^k)^2}}} =$$

$$= \frac{2^k\cdot 2\cdot\frac{1}{2^k}\cdot x_k}{\sqrt{2}\sqrt{1+\sqrt{1-(x_k/2^k)^2}}} = \frac{2x_k}{\sqrt{2\big(1+\sqrt{1-(x_k/2^k)^2}\big)}}.$$

□

Twierdzenie 5. *Gdy ciąg x_k jest dany wzorem ogólnym $x_k := 2^k \sin\frac{\pi}{2^k}$, to spełnia on nastpujące równanie rekurencyjne dla warunków początkowych $x_1 = 2, x_2 = 2\sqrt{2}$*

$$x_{k+1} = x_k \sqrt{\frac{2x_k}{x_k + x_{k-1}}} \tag{3}$$

Dowód. Oczywiście łatwo sprawdzić, że ciąg $\{x_k\}$ spełnia warunki początkowe. Wykorzystamy teraz twierdzenie 2.

$$x_{k+1} = \frac{2x_k}{\sqrt{2\big(1+\sqrt{1-(x_k/2^k)^2}\big)}} = x_k \cdot \sqrt{\frac{4}{2\big(1+\sqrt{1-(x_k/2^k)^2}\big)}} =$$
$$= x_k \sqrt{\frac{2x_k}{x_k\big(1+\sqrt{1-(x_k/2^k)^2}\big)}} = x_k \sqrt{\frac{2x_k}{x_k + x_k\sqrt{1-(x_k/2^k)^2}}}.$$

Pozostaje więc wykazać, że $x_k\sqrt{1-(x_k/2^k)^2} = x_{k-1}$ co jest równoważne pokazaniu tożsamości $2^k\sqrt{1-(\frac{2^k \sin \pi/2^k}{2^k})^2} = 2^{k-1}\sin\frac{\pi}{2^{k-1}}$. Uprośćmy lewą stronę:

$$2^k \sin\frac{\pi}{2^k}\sqrt{1-(\frac{2^k\sin\pi/2^k}{2^k})^2} = 2^k\sin\frac{\pi}{2^k}\sqrt{1-sin\frac{\pi}{2^k}^2} =$$
$$= 2^k \sin\frac{\pi}{2^k}\cos\frac{\pi}{2^k} = 2^k\frac{\sin\left(2\cdot\frac{\pi}{2^k}\right)}{2} = 2^{k-1}\sin\frac{\pi}{2^{k-1}},$$

co kończy dowód. □

5. Propagacja błędów w wykorzystanych wzorach.

Sprawdzimy jaki błąd może pojawić się przy obliczeniach kolejnych wyrazów ciągu x_k.

Uwaga 1. Uznajemy, że w arytmetyce *fl* działania wykonują się z błędem rzędu precyzji arytmetyki. Dokładniej mówiąc zachodzi

$$fl(\bar{a}\circ\bar{b}) = (\bar{a}\circ\bar{b})(1+\epsilon),$$

dla $\circ \in \{+,-,*,/\}$ gdzie $|\epsilon| \le u$, a liczbę $u := 2^{-t-1}$ nazywamy precyzją arytmetyki. Analogicznie też zachodzi

$$fl(\sqrt{\bar{a}}) = \sqrt{\bar{a}}(1+\epsilon).$$

Zauważmy, że w naszym modelu liczba 2^k oblicza się dokładnie oraz mnożenie przez potęgę dowolną (całkowitą) potęgę dwójki także nie powoduje błędu, gdyż polega tylko na zmianie cechy drugiego czynnika - mantysa pozostaje bez zmian.

Uwaga 2. Przez relację $a \lesssim b$ będziemy rozumieć relację $a \underset{1}{\leqslant} b$ określoną w [1], s. 15-16. Dla propagacji (nagromadzenia) błędów zachodzi następujący wzór:

$$\prod_{i=1}^{n}(1+\epsilon_i)^{k_i} = 1+\eta,$$

gdzie dla $k_i \in \{-1,1\}$ zachodzi $|\eta| \lesssim nu$.

Szkic dowodu jest taki, że nierówność

$$1-nu \le \prod_{i=1}^{n}(1+\epsilon_i)^{k_i} \le 1+\frac{nu}{1-nu} = \frac{1}{1-nu}$$

wynika wprost z nierówności Bernoulliego, gdzie wiemy, że skoro

$$1-u < \frac{1}{1+u} < 1+\epsilon_i < 1+u < \frac{1}{1-u}$$

to

$$1-nu < (1-u)^n < \left(\frac{1}{1+u}\right)^n < \prod_{i=1}^{n}(1+\epsilon_i)^{k_i} < (1+u)^n < \left(\frac{1}{1-u}\right)^n < \frac{1}{1-nu}$$

Oprócz tego zachodzi także prosta do zaobserwowania zależność, że $\sqrt{1+\epsilon} = 1+\gamma$, gdzie $|\gamma| \lesssim \frac{1}{2}u$.

5.1. Błędy między kolejnymi iteracjami we wzorze (1).

Zauważmy, że w tej formule może wystąpić problem utraty cyfr znaczących ze względu na odejmowanie bliskich sobie liczb.[4]

$$\begin{aligned} fl(x_{k+1}) &= fl\left(2^k\sqrt{2\Big(1-\sqrt{1-(\bar{x}_k/2^k)^2}\Big)}\right) = \\ &= 2^k\sqrt{2\left(1-\sqrt{\left(1-\left(\frac{\bar{x}_k}{2^k}\right)^2(1+\epsilon_4)\right)(1+\epsilon_3)}\right)(1+\epsilon_2)(1+\epsilon_1)} \\ &= 2^k\sqrt{2\left(1-\sqrt{\left(1-\left(\frac{\bar{\bar{x}}_k}{2^k}\right)^2\right)(1+\eta_2)}\right)(1+\eta_1)} \end{aligned}$$

Gdzie $|\epsilon_i| \leq u$ dla $i = 1, ..., 4$, gdzie ϵ_1 pochodzi z pierwiastkowania, $\epsilon_{2,3}$ - odejmowania, ϵ_4 - podniesienia do potęgi drugiej (albo prościej - mnożenia). Oraz przyjmijmy, że $1+\eta_1 = \sqrt{1+\epsilon_2}(1+\epsilon_1)$,$1+\eta_2 = \sqrt{1+\epsilon_3}$, $\bar{\bar{x}}_k = \bar{x}_k(1+\eta_3) = \bar{x}_k\sqrt{1+\epsilon_4}$. Wtedy łatwo zauważyć, że $|\eta_1| \lesssim (1+1/2)u = \frac{3}{2}u$, $|\eta_2| \lesssim \frac{1}{2}u$ oraz $|\eta_3| \lesssim \frac{1}{2}u$
Dla przejrzystości obliczeń półóżmy chwilowo $T := 1 - \left(\bar{\bar{x}}_k/2^k\right)^2$

$$\begin{aligned} fl(x_{k+1}) &= 2^k\sqrt{2\left(1-\sqrt{T}(1+\eta_2)\right)(1+\eta_1)} \\ &= 2^k\sqrt{2\left(1-\sqrt{T}\right)\left(1-\frac{\eta_2\sqrt{T}}{1-\sqrt{T}}\right)}(1+\eta_1) \\ &= 2^k\sqrt{2\left(1-\sqrt{T}\right)}\cdot\sqrt{1-\frac{\eta_2\sqrt{T}}{1-\sqrt{T}}}(1+\eta_1) \\ &= 2^k\sqrt{2\left(1-\sqrt{1-\left(\bar{\bar{x}}_k/2^k\right)^2}\right)}\cdot(1+\eta_4) \end{aligned}$$

Gdzie $1+\eta_4 = \sqrt{1-\frac{\eta_2\sqrt{T}}{1-\sqrt{T}}}(1+\eta_1) = \sqrt{1-\frac{\eta_2\sqrt{1-\left(\bar{\bar{x}}_k/2^k\right)^2}}{1-\sqrt{1-\left(\bar{\bar{x}}_k/2^k\right)^2}}}(1+\eta_1)$.

W takim razie

$$|\eta_4| \lesssim (\frac{3}{2} + \frac{\frac{1}{2}\cdot\frac{1}{2}\cdot\sqrt{T}}{1-\sqrt{T}})u = (\frac{3}{2} + \frac{\frac{1}{4}}{\frac{1}{\sqrt{T}}-1})u$$

Zakładając, że do pewnego momentu ciąg $\bar{\bar{x}}_k$ jest ograniczony przez jakąś małą stałą, otrzymujemy, że dla odpowiednio dużego k wyrażenie $\frac{\bar{\bar{x}}_k}{2^k}$ jest dowolnie blisko zera, więc wyrażenie $\frac{1}{\sqrt{T}} - 1 = \frac{1}{\sqrt{1-\left(\bar{\bar{x}}_k/2^k\right)^2}} - 1$ jest dowolnie blisko zera. Z tego łatwo wywnioskować, że gdy $2^k \gg \bar{\bar{x}}_k$, to błąd η_4 jest dowolnie duży.

[4] Oprócz tego może pojawić się inny problem - liczba 2^k dla odpowiednio dużych k musi przekroczyć zakres arytmetyki, którą się posługujemy, co może spowodować wyniki pokroju NaN czy Inf.

5.2. Błędy między kolejnymi iteracjami we wzorze (2).

Zauważmy, że ta formuła nie zawiera odejmowania bliskich sobie liczb, więc zjawisko utraty cyfr znaczących nie powinno wystąpić.

$$\begin{aligned} fl(x_{k+1}) &= fl\left(\frac{2x_k}{\sqrt{2\big(1+\sqrt{1-(x_k/2^k)^2}\big)}}\right) \\ &= \frac{2\bar{x}_k}{\sqrt{2\left(1+\sqrt{\left(1-\left(\frac{\bar{x}_k}{2^k}\right)^2(1+\epsilon_5)\right)(1+\epsilon_4)}\right)(1+\epsilon_3)(1+\epsilon_2)}}(1+\epsilon_1) \\ &= \frac{2\bar{\bar{x}}_k\frac{1}{1+\eta_2}}{\sqrt{2\left(1+\sqrt{(1-(\frac{\bar{\bar{x}}_k}{2^k})^2)(1+\epsilon_4)}\right)}}(1+\eta_1) \\ &= \frac{2\bar{\bar{x}}_k}{\sqrt{2\left(1+\sqrt{1-(\frac{\bar{\bar{x}}_k}{2^k})^2}(1+\eta_3)\right)}}\frac{1+\eta_1}{1+\eta_2} \end{aligned}$$

Gdzie $|\epsilon_i| \leq u$ dla $i = 1, ..., 5$, gdzie ϵ_1 pochodzi z działania dzielenia, ϵ_2 - pierwiastkowania,ϵ_3 - dodawania, ϵ_4 - odejmowania, ϵ_5 - podniesienia do potęgi drugiej.
Niech $1+\eta_1 = \frac{(1+\epsilon_1)}{(1+\epsilon_2)\sqrt{1+\epsilon_3}}$ oraz $\bar{\bar{x}}_k = \bar{x}_k(1+\eta_2) = \bar{x}_k\sqrt{1+\epsilon_5}$ oraz $1+\eta_3 = \sqrt{1+\epsilon_4}$.
W takim razie $|\eta_1| \lesssim (1+1+1/2)u = \frac{5}{2}u$, $|\eta_2| \lesssim (1/2)u = \frac{1}{2}u$ oraz $|\eta_3| \lesssim (1/2)u = \frac{1}{2}u$. Niech $1+\delta_1 = \frac{1+\eta_1}{1+\eta_2}$, a więc $|\delta_1| \lesssim (5/2+1/2)u = 3u$ Dla przejrzystości obliczeń półóżmy chwilowo $T := 1 - \left(\bar{\bar{x}}_k/2^k\right)^2$

$$\begin{aligned} fl(x_{k+1}) &= \frac{2\bar{\bar{x}}_k}{\sqrt{2\Big(1+\sqrt{T}(1+\eta_3)\Big)}}(1+\delta_1) \\ &= \frac{2\bar{\bar{x}}_k}{\sqrt{2\Big((1+\sqrt{T})(1+\frac{\eta_3\sqrt{T}}{1+\sqrt{T}})\Big)}}(1+\delta_1) \\ &= \frac{2\bar{\bar{x}}_k}{\sqrt{2\big(1+\sqrt{T}\big)}}\frac{1+\delta_1}{\sqrt{1+\frac{\eta_3\sqrt{T}}{1+\sqrt{T}}}} \\ &= \frac{2\bar{\bar{x}}_k}{\sqrt{2\Big(1+\sqrt{1-\left(\bar{\bar{x}}_k/2^k\right)^2}\Big)}}\cdot(1+\gamma), \end{aligned}$$

gdzie

$$1+\gamma = \frac{1+\delta_1}{\sqrt{1+\frac{\eta_3\sqrt{T}}{1+\sqrt{T}}}},$$

a więc znowu

$$|\gamma| \lesssim (\frac{1}{2}\frac{\frac{1}{2}\sqrt{T}}{1+\sqrt{T}}+3)u \leq (3+1/4)u.$$

5.3. Błędy między kolejnymi iteracjami we wzorze (3).

Zauważmy, że w tym wzorze wykonujemy relatywnie mało operacji w porównaniu do wzorów (2) i (3). Wykorzystujemy pierwiastkowanie tylko raz oraz nie ma odejmowania liczb tych samych rzędów, więc nie

powinno wystąpić zjawisko utraty cyfr znaczących.

$$\begin{aligned} fl(x_{k+1}) &= fl\left(x_k\sqrt{\frac{2x_k}{x_k+x_{k+1}}}\right) \\ &= \bar{x}_k\sqrt{\frac{2\bar{x}_k}{(\bar{x}_k+\bar{x}_{k+1})(1+\epsilon_4)}(1+\epsilon_3)}(1+\epsilon_1)(1+\epsilon_2) \\ &= \bar{x}_k\sqrt{\frac{2\bar{x}_k}{\bar{x}_k+\bar{x}_{k+1}}}\sqrt{\frac{1+\epsilon_3}{1+\epsilon_4}}(1+\epsilon_1)(1+\epsilon_2) \\ &= \bar{x}_k\sqrt{\frac{2\bar{x}_k}{\bar{x}_k+\bar{x}_{k+1}}}(1+\eta) \end{aligned}$$

Gdzie błędy $|\epsilon_i| \leq u$ dla $i = 1, ..., 4$ oraz ϵ_1 jest błędem zaokrąglenia powstałym podczas mnożenia, ϵ_2 - pierwiastkowania, ϵ_3 - dzielenia, ϵ_4 - dodawania. Oznaczmy w takim razie $1+\eta = \sqrt{\frac{1+\epsilon_3}{1+\epsilon_4}}(1+\epsilon_1)(1+\epsilon_2)$
Skoro $|\epsilon_i| \leq u$ to $|\eta| \lesssim (1/2+1/2+1+1)u = 3u$

6. Testy empiryczne wzorów (1), (2) oraz (3) i obserwacje.

Dla każdego ze wzorów obliczymy najpierw pierwsze 50 wyrazów wykorzystując arytmetykę pojedynczej i podwójnej precyzji. Wykorzystujemy do tego, odpowiednio, typy *Float*32 i *Float*64 dostępne w Julii. Obliczymy też odpowiadające im błędy względne przybliżenia wykorzystując arytmetykę wysokiej precyzji - w naszym wypadku ustalimy arytmetykę 256-bitową, reprezentowaną w Julii przez typ *BigFloat*, oraz stałą *pi* typu *Irrational* również dostępną w Julii.

6.1. Testy wzoru (1), błędy względne wyników oraz komentarz.

i	Pojedyncza precyzja	Błąd
1	2.0000000000000000000000000	3.6338e-01
2	2.8284270763397216796875000	9.9684e-02
3	3.0614674091339111328125000	2.5505e-02
4	3.1214444637298583984375000	6.4134e-03
5	3.1365461349487304687500000	1.6064e-03
6	3.1403334140777587890625000	4.0083e-04
7	3.1412856578826904296875000	9.7720e-05
8	3.1415188312530517578125000	2.3498e-05
9	3.1412079334259033203125000	1.2246e-04
10	3.1424512863159179687500000	2.7331e-04
11	3.1424512863159179687500000	2.7331e-04
12	3.1622776985168457031250000	6.5843e-03
13	3.1622776985168457031250000	6.5843e-03
14	2.8284270763397216796875000	9.9684e-02
15	0.0000000000000000000000000	1.0000e+00
...		
50	0.0000000000000000000000000	1.0000e+00

i	Podwójna precyzja	Błąd
1	2.0000000000000000000000000	3.6338e-01
2	2.8284271247461902909492440	9.9684e-02
3	3.0614674589207186983230710	2.5505e-02
4	3.1214451522580528575190330	6.4131e-03
5	3.1365484905459406483885230	1.6056e-03
6	3.1403311569547391890466770	4.0155e-04
7	3.1412772509327568926096320	1.0040e-04
8	3.1415138011441454679584200	2.5100e-05
9	3.1415729403678827047485810	6.2749e-06
10	3.1415877252799608854161310	1.5687e-06
11	3.1415914215046352175875200	3.9218e-07
12	3.1415923456110768086091410	9.8033e-08
13	3.1415925765450043449789060	2.4524e-08
14	3.1415926334634632408437680	6.4065e-09
15	3.1415926548075892021927300	3.8764e-10
16	3.1415926453212152935634550	2.6320e-09
17	3.1415926073757196590463540	1.4710e-08
...		
25	3.1424512724941338071005250	2.7331e-04
26	3.1622776601683795227870630	6.5842e-03
27	3.1622776601683795227870630	6.5842e-03
28	3.4641016151377543863532080	1.0266e-01
29	4.0000000000000000000000000	2.7324e-01
30	0.0000000000000000000000000	1.0000e+00
...		
49	0.0000000000000000000000000	1.0000e+00
50	0.0000000000000000000000000	1.0000e+00

Możemy zauważyć, że w przypadku obliczeń z pojedynczą precyzją nie mamy szans na stabilizację wyniku. Nie mamy też szans osiągnąć więcej niż 5 dokładnych cyfr rozwinięcia dziesiętnego - najdokładniejsze z nich, x_8 osiąga 5 cyfr dokładnych i ma błąd względny rzędu 10^{-5} co jest dość dużą wartością. W przypadku obliczeń wykorzystujących arytmetykę z podwójną precyzją także nie mamy szans na stabilizację wyniku, ale możemy zdecydowanie dokładniej obliczyć π - wynik mający pięć początkowych cyfr dokładnych otrzymujemy dla x_8, a osiem cyfr dokładnych dla x_{15}.

Tak jak przewidywaliśmy przy oszacowaniu błędu między kolejnymi iteracjami, występuje zjawisko utraty cyfr znaczących przy obliczaniu kolejnych wyrazów. Obliczenia z wykorzystaniem wzoru (1) psują się dla odpowiednio dużego i niezależnie od wyboru precyzji arytmetyki. Przykładowo, dla arytmetyki 256-bitowej, psuje się wyraz x_{131}, a dla 512-bitowej - x_{260}. Wynik nie stabilizuje się w okolicy liczby π, a najlepszymi przybliżeniami ludolfiny jest x_8 dla pojedynczej precyzji oraz x_{15} dla podwójnej precyzji.

6.2. Testy wzoru (2), błędy względne wyników oraz komentarz do wyników.

i	Pojedyncza precyzja	Błąd
1	2.0000000000000000000000	3.6338e-01
2	2.8284270763397216796875 00	9.9684e-02
3	3.0614674091339111328125 00	2.5505e-02
4	3.1214449405670166015625 00	6.4132e-03
5	3.1365482807159423828125 00	1.6057e-03
6	3.1403310298919677734375 00	4.0159e-04
7	3.1412770748138427734375 00	1.0045e-04
8	3.1415135860443115234375 00	2.5168e-05
9	3.1415727138519287109375 00	6.3470e-06
10	3.1415874958038330078125 00	1.6418e-06
11	3.1415913105010986328125 00	4.2752e-07
12	3.1415922641752150390625 00	1.2395e-07
13	3.1415925025939941406250 00	4.8063e-08
14	3.1415925025939941406250 00	4.8063e-08
...		
49	3.1415925025939941406250 00	4.8063e-08
50	3.1415925025939941406250 00	4.8063e-08

i	Podwójna precyzja	Błąd
1	2.0000000000000000000000	3.6338e-01
2	2.8284271247461898460860 34	9.9684e-02
3	3.0614674589207178101446 52	2.5505e-02
4	3.1214451522580519693406 13	6.4131e-03
5	3.1365484905459388720316 84	1.6056e-03
6	3.1403311569547525117229 72	4.0155e-04
7	3.1412772509327724357319 77	1.0040e-04
8	3.1415138011443008991818 72	2.5100e-05
9	3.1415729403670913377766 28	6.2749e-06
...		
13	3.1415925765848724537931 96	2.4511e-08
14	3.1415926343385631724913 76	6.1279e-09
15	3.1415926487769856301213 17	1.5320e-09
16	3.1415926523865911335064 99	3.8299e-10
...		
24	3.1415926535897757965187 79	5.5519e-15
25	3.1415926535897895632842 85	1.1698e-15
26	3.1415926535897931159979 63	3.8982e-17
27	3.1415926535897940041763 83	2.4373e-16
28	3.1415926535897944482655 93	3.8509e-16
...		
49	3.1415926535897944482655 93	3.8509e-16
50	3.1415926535897944482655 93	3.8509e-16

Możemy zauważyć, że formuła (2) jest zdecydowanie lepsza od formuły (1) - dla pojedynczej precyzji wynik stabilizuje się dla 7 początkowych cyfr dokładnych, dla podwójnej - dla 15 cyfr. W obu precyzjach, stabilizację dla 5 cyfr początkowych osiągamy dla wyrazu x_8, a dla podwójnej precyzji, stabilizację 8 cyfr początkowych - przy wyrazie x_{14}.

Tak jak przewidywaliśmy przy propagacji błędu między kolejnymi iteracjami, błędy są rzędów odpowiednio $10^{-8} \approx 2^{-26}$ oraz $10^{-16} \approx 2^{-53}$ i są na poziomie błędu reprezentacji liczby.

6.3. Testy wzoru (3), błędy względne wyników oraz komentarz do wyników.

i	Pojedyncza precyzja	Błąd
1	2.00000000000000000000000	3.6338e-01
2	2.82842707633972167968750	9.9684e-02
3	3.06146740913391113281250	2.5505e-02
4	3.12144517898559570312500	6.4131e-03
5	3.13654851913452148437500	1.6056e-03
6	3.14033102989196777343750	4.0159e-04
7	3.14127707481384277343750	1.0045e-04
8	3.14151334762573242187500	2.5244e-05
9	3.14157247543334960937500	6.4229e-06
10	3.14158701896667480468750	1.7936e-06
11	3.14159083366394042968750	5.7930e-07
12	3.14159154891967773437500	3.5163e-07
13	3.14159154891967773437500	3.5163e-07
...		
49	3.14159154891967773437500	3.5163e-07
50	3.14159154891967773437500	3.5163e-07

i	Podwójna precyzja	Błąd
1	2.00000000000000000000000	3.6338e-01
2	2.82842712474619029094924	9.9684e-02
3	3.06146745892071869832307	2.5505e-02
4	3.12144515225805285751903	6.4131e-03
5	3.13654849054593976021010	1.6056e-03
6	3.14033115695475339990139	4.0155e-04
7	3.14127725093277332391039	1.0040e-04
8	3.14151380114430134327108	2.5100e-05
9	3.14157294036709133777662	6.2749e-06
...		
13	3.14159257658487245379319	2.4511e-08
14	3.14159263433856317249137	6.1279e-09
15	3.14159264877698607421052	1.5320e-09
16	3.14159265238659157759570	3.8299e-10
...		
24	3.14159265358977535242956	5.6933e-15
25	3.14159265358978867510586	1.4526e-15
26	3.14159265358979222781954	3.2170e-16
27	3.14159265358979311599796	3.8982e-17
28	3.14159265358979311599796	3.8982e-17
...		
49	3.14159265358979311599796	3.8982e-17
50	3.14159265358979311599796	3.8982e-17

Możemy zauważyć, że wzór (3) jest nieco lepszy od wzoru (2). Oczywiście zbieżność musi być taka sama, bo obliczamy przybliżenia wyrazów tego samego ciągu. Wyniki obliczeń dla pojedynczej i podwójnej precyzji są bardzo podobne jak wyniki dla wzoru (2). Tutaj, w przypadku podwójnej precyzji na korzyść wzoru (3), ponieważ dostajemy jeszcze jedną cyfrę znaczącą rozwinięcia dziesiętnego, chociaż dla arytmetyk wyższych precyzji różnica ta nie jest zauważalna. Tak jak poprzednio stabilizację dla 5 cyfr początkowych osiągamy dla wyrazu x_8, a dla 8 cyfr początkowych - przy wyrazie x_{14}. Plusem tego wzoru jest zdecydowanie fakt, że wykonujemy o wiele mniej działań w porównaniu do wzoru (2).

Tak jak przewidywaliśmy przy propagacji błędu między kolejnymi iteracjami, błędy są rzędów odpowiednio $10^{-7} \approx 2^{-23}$ oraz $10^{-17} \approx 2^{-56}$ i są na poziomie błędu reprezentacji liczby.

7. Wykładnik zbieżności.

Łatwo zauważyć, że błąd między liczbą π, a naszymi przybliżeniami osiąganymi przy wzorach (1), (2) i (3) zmniejsza się o jeden rząd co około 2 iteracje. Spróbujmy więc znaleźć wykładnik zbieżności naszej metody.

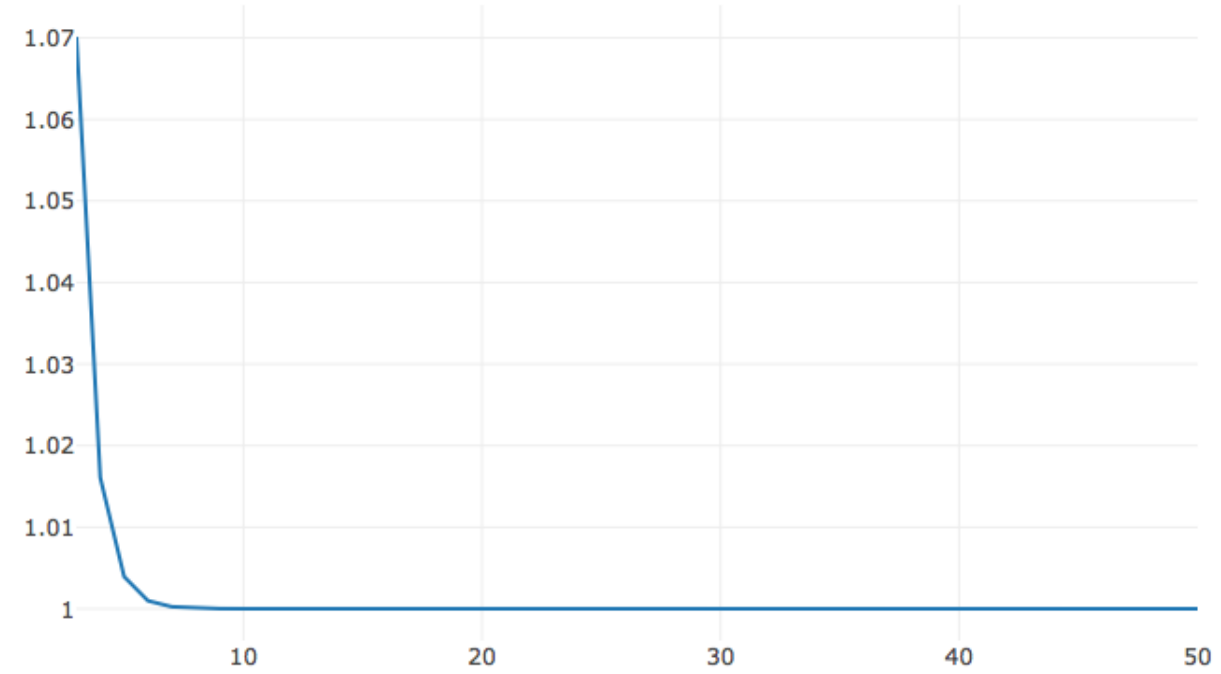
Fakt 1. Niech x_n będzie ciągiem zbieżnym do pewnej stałej. Wtedy

$$q \approx \frac{\log|\frac{x_{n+1}-x_n}{x_n-x_{n-1}}|}{\log|\frac{x_n-x_{n-1}}{x_{n-1}-x_{n-2}}|}$$

gdzie q to wykładnik zbieżności.

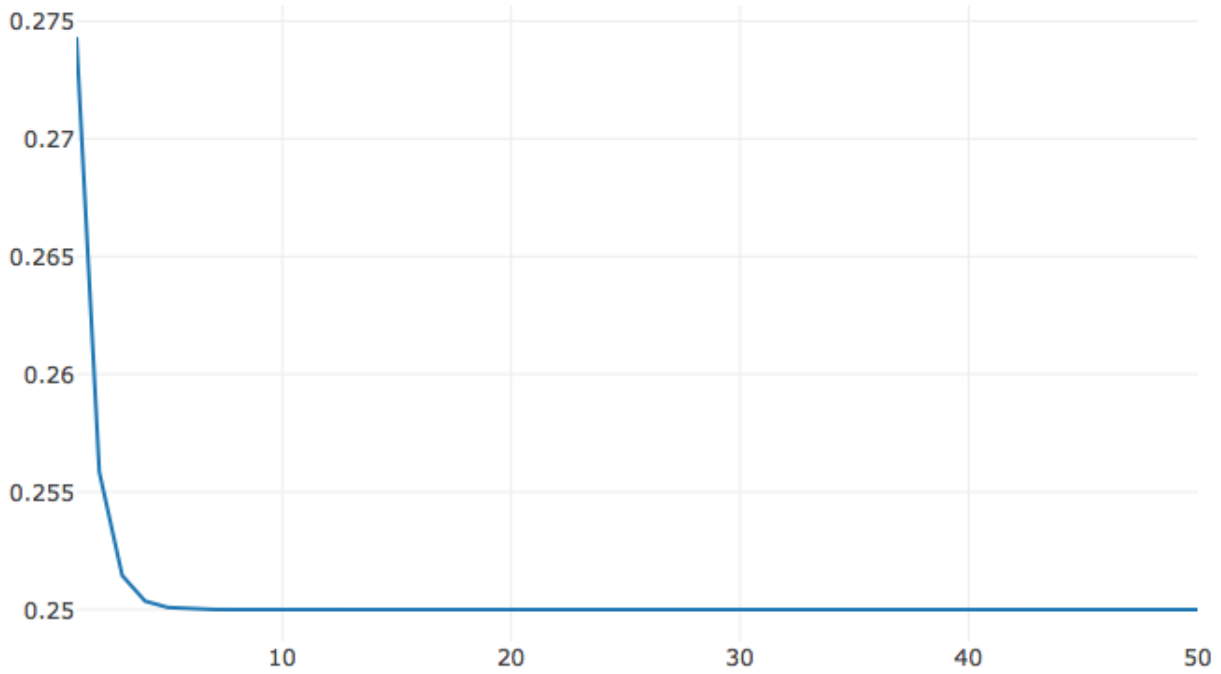
Skoro wszystkie ciągi rekurencyjne mają za zadanie przybliżyć wyrażenie $x_k = 2^k \sin \pi/2^k$ to w takim razie wzory (1), (2) oraz (3) są zbieżne tak samo szybko. Poszukamy więc wykładnika zbieżności wykorzystując

wzór (3).



Wykres 2: Wartości q_n dla $n = 3, ..., 50$

Na tej podstawie możemy przewidywać, że zbieżność jest liniowa, sprawdźmy wartości ciągu $\frac{|x_{n+1}-\pi|}{|x_n-\pi|}$ - jeśli zbiega on do stałej z przedziału $(0, 1)$ to zbieżność rzeczywiście jest liniowa.



Wykres 3: Wartości $\frac{|x_{n+1}-\pi|}{|x_n-\pi|}$ dla $n = 1, ..., 50$

W takim razie możemy przewidywać, że

$$\lim_{n\to\infty} \frac{|x_{n+1}-\pi|}{|x_n-\pi|} = 1/4.$$

Spróbujmy więc udowodnić następujące twierdzenie:

Twierdzenie 6.

$$\lim_{n\to\infty} \frac{|2^{n+1}\sin \pi/2^{n+1} - \pi|}{|2^n \sin \pi/2^n - \pi|} = \frac{1}{4}$$

Dowód. Wykorzystując rachunek pochodnych łatwo możemy sprawdzić, że $f(x) = x - \sin x > 0$, gdy $x > 0$. W takim razie, kładąc $x := \pi/2^k$ otrzymujemy, że $\frac{\pi}{2^k} > \sin \pi/2^k$, a więc $\pi > 2^k \sin \pi/2^k$. W takim razie

$$\begin{aligned}\lim_{n\to\infty} \frac{|2^{n+1}\sin(\pi/2^{n+1}) - \pi|}{|2^n \sin(\pi/2^n) - \pi|} &= \lim_{n\to\infty} \frac{\pi - 2^{n+1}\sin(\pi/2^{n+1})}{\pi - 2^n \sin(\pi/2^n)} = \\ &= \lim_{n\to\infty} \frac{2^{n+1}}{2^n} \cdot \frac{\frac{\pi}{2^{n+1}} - \sin(\pi/2^{n+1})}{\frac{\pi}{2^n} - \sin(\pi/2^n)} = 2 \lim_{n\to\infty} \frac{\frac{\pi}{2^{n+1}} - \sin(\pi/2^{n+1})}{\frac{\pi}{2^n} - \sin(\pi/2^n)}\end{aligned}$$

Połóżmy $x := \frac{\pi}{2^{n+1}} \overset{n\to\infty}{\to} 0$ i wykorzystajmy kilkukrotnie regułę de l'Hospitala (możemy, ponieważ w każdym kroku mamy wyrażenie nieoznaczone $0/0$).

$$\begin{aligned}\lim_{n\to\infty} \frac{|2^{n+1}\sin(\pi/2^{n+1}) - \pi|}{|2^n \sin(\pi/2^n) - \pi|} &= 2 \lim_{x\to 0} \frac{x - \sin x}{2x - \sin 2x} = 2 \lim_{x\to 0} \frac{1 - \cos x}{2 - 2\cos 2x} = \\ &= 2 \lim_{x\to 0} \frac{\sin x}{4 \sin 2x} = 2 \lim_{x\to 0} \frac{\cos x}{8 \cos 2x} = 2 \cdot \frac{1}{8} = \frac{1}{4}\end{aligned}$$

□

Twierdzenie 6 ostatecznie dowodzi, że formuły (1),(2) i (3) są zbieżne liniowo do π.

8. Podsumowanie obliczeń.

Łatwo zauważyć, że wzór (1) jest bezużyteczny w kontekście obliczeń numerycznych. Dla odpowiednio dużego k nasze x_k nijak nie przypomina liczby π ze względu na utratę cyfr znaczących przy odejmowaniu bliskich sobie liczb. Proste przekształcenie pozwala nam wyprowadzić wzór (2), w którym nie doświadczamy utraty cyfr znaczących - propagowany błąd jest ograniczony i jest rzędu błędu reprezentacji, a obliczanie kolejnych wyrazów ciągu stabilizuje się, aczkolwiek wykonujemy dość dużo działań w każdej iteracji.
Wzór (3) jest nieco lepszy od wzoru (2), a co za tym idzie jest najlepszym wzorem od strony numerycznej z tych tutaj rozpatrywanych - wykonujemy mało działań w każdej iteracji, a propagowany błąd jest rzędu błędu reprezentacji. Jedyna wada jaką można dostrzec, to taka, że naiwna rekurencyjna implementacja tego wzoru jest kosztowna od strony pamięciowej - mamy co najmniej dwa wywołania rekurencyjne w każdym kroku, więc czas wykonania funkcji jest $O(2^n)$, aczkolwiek każdy student informatyki po Wstępie do Informatyki powinien poradzić sobie z implementacją tego wzoru działającą w czasie liniowym od ilości iteracji. Wszystkie te wzory są zbieżne liniowo do liczby π, co nie jest najlepszym wynikiem. Zdecydowanie lepszy efekt dałoby wykorzystanie algorytmu Chudnovsky'ego czy sum wyprowadzonych przez Machina wykorzystujących funkcję $\arctan(x)$.

Literatura

[1] J. i M. Jankowscy, *Przegląd metod i algorytmów numerycznych*, cz. 1, WNT, 1981.

[2] Michelle Schatzman, *Numerical analysis: a mathematical introduction*, Clarendon Press, Oxford, 2002

[3] W. Cheney, D. Kincaid, *Analiza numeryczna*, WNT, 2006.

[4] `https://en.wikipedia.org/wiki/Approximations_of_pi` (Z dnia 10.11.2017).

[5] `https://en.wikipedia.org/wiki/Chronology_of_computation_of_pi` (Z dnia 10.11.2017).