Obliczenia Naukowe – Sprawozdanie Laboratoria 1.

Jakub Kogut

21 października 2025

1 Zadanie 1 – Rozpoznanie arytmetyki

1.1 Mashine epsilon

W pierwszej części zadania należało wyznaczyć najmniejszą liczbę $macheps > 0$ taką, że

$$fl(macheps + 1) > 1.$$

Aby znaleść daną liczbe iteracyjne można dodwać do 1 coraz mniejsze potęgi 2, to znaczy iterować do momentu, aż $fl(1 + 2^{-k})$ przestanie być większe od 1. W ten sposób otrzymujemy:

Typ zmiennej	Wartość *macheps* eksperymentalna	$eps(T)$	float.h
Float16	$9.76 \cdot 10^{-4}$	$9.76 \cdot 10^{-4}$	–
Float32	$1.19 \cdot 10^{-7}$	$1.19 \cdot 10^{-7}$	$1.19 \cdot 10^{-7}$
Float64	$2.22 \cdot 10^{-16}$	$2.22 \cdot 10^{-16}$	$2.22 \cdot 10^{-16}$

Z wyników można wnioskować, że eksperymentalnie wyznaczone wartości *macheps* zgadzają się z teoretycznymi wartościami $eps(T)$ dla poszczególnych typów zmiennoprzecinkowych, jak i również z wartościami podanymi w pliku float.h.
Z eksperymentu wynika, że zwiększając precyzję reprezentacji liczby zmiennoprzecinkowej, zmniejsza się wartość *macheps*. Oznacza to, że liczby reprezentowane z większą precyzją mogą być bliżej siebie, co pozwala na dokładniejsze obliczenia.
Związek *macheps* z precyzją arytmetyki
Jest to ta sama liczba, na wykładzie oznaczaliśmy przez ϵ najmniejszą możliwą liczbę w danej arytmetyce, taką że $fl(1+\epsilon) > 1$. W związku z tym *macheps* jest miarą precyzji danej arytmetyki zmiennoprzecinkowej.

1.2 Najmniejsza Dodatnia Liczba Maszynowa

W kolejnej części zadania należało wyznaczyć kolejną liczbę identyfikującą arytmetykę – η taką, że

$$\eta > 0$$

Podobnie jak w poprzedniej części zadania możemy iteracyjnie dzielić 1 przez 2 aż do momentu, gdy $fl(2^{-k})$ przestanie być większe od 0. W ten sposób otrzymujemy:

Typ zmiennej	Wartość η eksperymentalna	$nextfloat(FloatT(0.0))$
Float16	$6.10 \cdot 10^{-5}$	$6.10 \cdot 10^{-5}$
Float32	$1.18 \cdot 10^{-38}$	$1.18 \cdot 10^{-38}$
Float64	$2.22 \cdot 10^{-308}$	$2.22 \cdot 10^{-308}$

Tak samo jak wcześniej, eksperymentalnie wyznaczone wartości η zgadzają się z wartościami zwracanymi przez funkcję $nextfloat(FloatT(0.0))$ dla poszczególnych typów zmiennoprzecinkowych.

Analogicznie, zwiększając precyzję reprezentacji liczby zmiennoprzecinkowej, zmniejsza się wartość η. Oznacza to, że liczby reprezentowane z większą precyzją mogą być bliżej zera, co pozwala na dokładniejsze obliczenia w pobliżu zera.
Związek η z MIN_{SUB}
Tak samo, w tym przypadku η jest równe MIN_{SUB}, czyli najmniejszej dodatniej liczbie nieznormalizeowenej w danej arytmetyce zmiennoprzecinkowej.
Co zwraca funkcja $floatmin(T)$?
Funkcja $floatmin(T)$ zwraca najmniejszą dodatnią liczbę znormalizowaną w danej arytmetyce zmiennoprzecinkowej, to jest MIN_{nor}. Dla odpowienio typów $Float32$ i $Float64$ wartości te wynoszą odpowiednio $1.18 \cdot 10^{-38}$ oraz $2.22 \cdot 10^{-308}$, co zgadza się z wynikami podanymi na wykładzie.

1.3 Największa Dodatnia Liczba Maszynowa

W ostatniej części zadania należy wyznaczyć górną granicę reprezentowalnych liczb w danej arytmetyce zmiennoprzecinkowej, czyli MAX.
Aby wyznaczyć tę liczbę skorzystamy z wzoru:

$$MAX = (2 - 2^{-p}) \cdot 2^{emax}$$

gdzie p to precyzja arytmetyki, a *emax* to maksymalna wartość wykładnika (w arytmetyce z której korzystamy – spełniającą standard IEEE 754 – największa możliwa mantysa to właśnie $2 - 2^{-p}$). Do wyznaczenia wartości *emax* skorzystamy właśnie z wskazanej funkcji $isinf$ mnożąc kolejne potęgi 2 aż do momentu, gdy $fl(2^k)$ zwróci nam wartość nieskończoności, natomiast aby obliczyć p postępujemy analogicznie jak w pierwszej części zadania, przy wyzanaczniu *macheps* i licząc ilość iteracji potrzebnych do uzyskania tej wartości. W ten sposób otrzymujemy:

Typ zmiennej	Wartość MAX eksperymentalna	$floatmax(FloatT))$	float.h
Float16	$6.55 \cdot 10^4$	$6.55 \cdot 10^4$	–
Float32	$3.40 \cdot 10^{38}$	$3.40 \cdot 10^{38}$	$3.40 \cdot 10^{38}$
Float64	$1.79 \cdot 10^{308}$	$1.79 \cdot 10^{308}$	$1.79 \cdot 10^{308}$

Podobnie jak w poprzednich częściach zadania, eksperymentalnie wyznaczone wartości MAX zgadzają się z wartościami zwracanymi przez funkcję $floatmax(FloatT)$ dla poszczególnych typów zmiennoprzecinkowych, jak i również z wartościami podanymi w pliku float.h.

2 Zadanie 2

W zadaniu należy sprawdzić, czy możliwe jest wyznaczenie *macheps* za obliczenia

$$3 \cdot (4/3 - 1) - 1$$

w arytmetyce zmiennoprzecinkowej $Float64$.
Wynik obliczeń to:

Typ zmiennej	Wartość eksperymentalna	*macheps*
Float16	$-9.76 \cdot 10^{-4}$	$9.76 \cdot 10^{-4}$
Float32	$1.19 \cdot 10^{-7}$	$1.19 \cdot 10^{-7}$
Float64	$-2.22 \cdot 10^{-16}$	$2.22 \cdot 10^{-16}$

Faktycznie, możliwe jest wyznaczenie *macheps* za pomocą podanego wyrażenia.

3 Zadanie 3

W zadaniu należy pokazać, że w arytmetyce $Float64$ liczby z przedziału $[1,2]$ są rozmieszczone równomiernie z krokiem 2^{-52} i następnie sprawdzić z jakimi krokami są rozmieszczone liczby z przedziałów $[1/2,1]$ oraz $[2,4]$.
Aby znaleść krok rozmieszczenia liczb w danym przedziale, możemy skorzystać z funkcji $nextfloat$, która zwraca następną reprezentowalną liczbę zmiennoprzecinkową. W ten sposób możemy obliczyć różnicę między kolejnymi liczbami w danym przedziale i w ten sposób wyznaczyć wartość δ.
Otrzymujemy:

Przedział	Krok rozmieszczenia liczb (δ)
$[1,2]$	$2.22 \cdot 10^{-16} \approx 2^{-52}$
$[1/2,1]$	$1.11 \cdot 10^{-16} \approx 2^{-53}$
$[2,4]$	$4.44 \cdot 10^{-16} \approx 2^{-51}$

Po znalezieniu kroku możemy zapisać reprezentację liczby z danego przedziału $[2^e, 2^{e+1}]$ jako:

$$x = 2^e + k\delta, \quad k \in \{0,1,2,\dots,2^{52}-1\}$$

Możemy potwierdzić nasze wyniki przy użyciu funkcji *bitstring*, która zwraca reprezentację binarną liczby zmiennoprzecinkowej. Dla liczb z przedziału $[1,2]$ wygląda to następująco:

Liczba	Bity cechy	Bity mantysy
1.00000000000000000000	01111111111	0000000000000000000000000000000000000000000000000000
1.00000000000000022204	01111111111	0000000000000000000000000000000000000000000000000001
1.00000000000000044409	01111111111	0000000000000000000000000000000000000000000000000010
1.00000000000000066613	01111111111	0000000000000000000000000000000000000000000000000011
...	...	...

Zwiększając liczbę o krok δ zmieniamy ostatni bit mantysy, co potwierdza nasze wyniki (pomiędzy liczbami różniącymy się o krok nie ma już innej reprezentacji). Analogicznie można to zrobić dla pozostałych przedziałów.

4 Zadanie 4

Należy znaleść eksperymentalnie najmniejszą liczbę x w arytmetyce $Float64$, $1 < x < 2$, taką że:

$$x \otimes (1/x)) \neq 1$$

Aby ją wyznaczyć, możemy zacząć od liczby 1 i zwiększać ja przy użyciu funkcji $nextfloat$ aż do momentu, gdy warunek przestanie być spełniony. W ten sposób otrzymujemy:

$$x = 1.000000057228997$$

Jest to najmniejsza liczba w arytmetyce $Float64$, spełniająca podany warunek.

5 Zadanie 5

W tym zadaniu należy zaimplementować 4 różne algorytmy obliczania iloczynu skalarnego dwóch wektorów w arytmetyce $Float32$ oraz $Float64$ i porównać ich wyniki między sobą oraz z wynikiem dokładnym.
Badany jest tutaj wpływ kolejności wykonywanych działań na błąd wyniku.
Zostały zaimplementowane następujące algorytmy:

1. **w przód** – $\sum_{i=1}^{n} x_i y_i$
2. **w tył** – $\sum_{i=n}^{1} x_i y_i$
3. **sortowanie rosnąco** – sortujemy iloczyny $x_i y_i$ rosnąco (w zależności od wartości bezwzględnej, osobno dodajemy ujemne i dodatnie)
4. **sortowanie malejąco** – analogicznie jak wyżej, ale sortujemy malejąco

Po przeprowadzeniu eksperymentów na wektorach:

$$x = [2.718281828, -3.141592654, 1.414213562, 0.5772156649, 0.3010299957]$$

$$y = [1486.2497, 878366.9879, -22.37492, 4773714.647, 0.000185049]$$

oraz obliczeniu dokładnego wyniku iloczynu skalarnego (wynoszącego $-1.00657107000000 \cdot 10^{-11}$) otrzymujemy następujące wyniki:

Algorytm	Wynik Float32	Wynik Float64
1.	$-4.9994430 \cdot 10^{-1}$	$1.025188136829667 \cdot 10^{-10}$
2.	$-4.5434570 \cdot 10^{-1}$	$-1.564330887049437 \cdot 10^{-10}$
3.	$-5.0000000 \cdot 10^{-1}$	$0.000000000000000 \cdot 10^{0}$
4.	$-5.0000000 \cdot 10^{-1}$	$0.000000000000000 \cdot 10^{0}$

Jak widać, wyniki są obarczone dużym błędem, a różne algorytmy dają różne wyniki, co potwierdza, że kolejność wykonywania działań ma znaczenie.

6 Zadanie 6

W tym zadaniu należało obliczyć wartości funkcji: $f(x) = \sqrt{x^2+1} - 1$ oraz $g(x) = \frac{x^2}{\sqrt{x^2+1}+1}$ Iterując po wartościach $x = 8^{-1}, 8^{-2}, 8^{-3}, \ldots$ aż do momentu, gdy wynik obliczeń przestanie się zmieniać (w arytmetyce $Float64$) otrzymujemy następujące wyniki:

x	f(x)	g(x)
8^{-1}	$7.7822185373186414 \cdot 10^{-3}$	$7.7822185373187065 \cdot 10^{-3}$
8^{-2}	$1.2206286282867573 \cdot 10^{-4}$	$1.2206286282875901 \cdot 10^{-4}$
8^{-3}	$1.9073468138230965 \cdot 10^{-6}$	$1.9073468138265659 \cdot 10^{-6}$
8^{-4}	$2.9802321943606103 \cdot 10^{-8}$	$2.9802321943606116 \cdot 10^{-8}$
8^{-5}	$4.6566128730773926 \cdot 10^{-10}$	$4.6566128719931904 \cdot 10^{-10}$
8^{-6}	$7.2759576141834259 \cdot 10^{-12}$	$7.2759576141569561 \cdot 10^{-12}$
8^{-7}	$1.1368683772161603 \cdot 10^{-13}$	$1.1368683772160957 \cdot 10^{-13}$
8^{-8}	$1.7763568394002505 \cdot 10^{-15}$	$1.7763568394002489 \cdot 10^{-15}$
8^{-9}	$0.0000000000000000 \cdot 10^{0}$	$2.7755575615628914 \cdot 10^{-17}$
...	...	...
8^{-170}	$0.0000000000000000 \cdot 10^{0}$	$4.4501477170144028 \cdot 10^{-308}$
8^{-171}	$0.0000000000000000 \cdot 10^{0}$	$6.9533558078350043 \cdot 10^{-310}$
8^{-172}	$0.0000000000000000 \cdot 10^{0}$	$1.0864618449742194 \cdot 10^{-311}$
8^{-173}	$0.0000000000000000 \cdot 10^{0}$	$1.6975966327722179 \cdot 10^{-313}$
8^{-174}	$0.0000000000000000 \cdot 10^{0}$	$2.6524947387065904 \cdot 10^{-315}$
8^{-175}	$0.0000000000000000 \cdot 10^{0}$	$4.1445230292290475 \cdot 10^{-317}$
8^{-176}	$0.0000000000000000 \cdot 10^{0}$	$6.4758172331703867 \cdot 10^{-319}$
8^{-177}	$0.0000000000000000 \cdot 10^{0}$	$1.0118464426828729 \cdot 10^{-320}$
8^{-178}	$0.0000000000000000 \cdot 10^{0}$	$1.5810100666919889 \cdot 10^{-322}$
8^{-179}	$0.0000000000000000 \cdot 10^{0}$	$0.0000000000000000 \cdot 10^{0}$

Chodziaż funkcje f oraz g są algebraicznie tożsame, to jednak ich wyniki różnią się znacznie. Funkcja f znacznie szybciej odchyla się od rzeczywistej wartość, dzieje się tak ze względu na duży błąd powstały przy odejmowaniu dwóch bliskich sobie liczb:

$$lim_{x \to 0} \sqrt{x^2 + 1} = 1$$

zatem przy małych wartościach x obie liczby w wyrażeniu $\sqrt{x^2 + 1} - 1$ są bardzo bliskie 1, co powoduje powstanie dużego błędu numerycznego. W przypadku funkcji g nie występuje odejmowanie bliskich sobie liczb, przez co błąd numeryczny jest znacznie mniejszy.

6.1 Zadanie 7

W zadaniu należało obliczyć przybliżoną wartość pochodnej funkcji $f(x) = \sin x + \cos 3x$ w punkcie $x_0 = 1$ za pomocą wzoru:

$$f'(x_0) \approx \tilde{f}'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h}$$

Obliczając wartości funkcji dla kolejnych wartości $h = 2^{-n}, n \in \{0, 1, 2, \dots, 54\}$ otrzymujemy następujące wyniki:

h	$\tilde{f}'(x_0)$	Błąd bezwzględny
2^0	$8.694677 \cdot 10^{-1}$	$7.525254 \cdot 10^{-1}$
2^{-1}	$4.730729 \cdot 10^{-1}$	$3.561306 \cdot 10^{-1}$
2^{-2}	$2.998405 \cdot 10^{-1}$	$1.828982 \cdot 10^{-1}$
2^{-3}	$2.507786 \cdot 10^{-1}$	$1.338363 \cdot 10^{-1}$
2^{-4}	$2.381337 \cdot 10^{-1}$	$1.211914 \cdot 10^{-1}$
2^{-5}	$2.349485 \cdot 10^{-1}$	$1.180062 \cdot 10^{-1}$
2^{-6}	$2.341506 \cdot 10^{-1}$	$1.172084 \cdot 10^{-1}$
2^{-7}	$2.339511 \cdot 10^{-1}$	$1.170088 \cdot 10^{-1}$
2^{-8}	$2.339012 \cdot 10^{-1}$	$1.169589 \cdot 10^{-1}$
2^{-9}	$2.338887 \cdot 10^{-1}$	$1.169464 \cdot 10^{-1}$
2^{-10}	$2.338856 \cdot 10^{-1}$	$1.169433 \cdot 10^{-1}$
2^{-11}	$2.338848 \cdot 10^{-1}$	$1.169425 \cdot 10^{-1}$
2^{-12}	$2.338846 \cdot 10^{-1}$	$1.169423 \cdot 10^{-1}$
2^{-13}	$2.338846 \cdot 10^{-1}$	$1.169423 \cdot 10^{-1}$
...	...	...
2^{-50}	$1.250000 \cdot 10^{-1}$	$8.057718 \cdot 10^{-3}$
2^{-51}	$2.500000 \cdot 10^{-1}$	$1.330577 \cdot 10^{-1}$
2^{-52}	$-5.000000 \cdot 10^{-1}$	$6.169423 \cdot 10^{-1}$
2^{-53}	$1.000000 \cdot 10^{0}$	$8.830577 \cdot 10^{-1}$
2^{-54}	$0.000000 \cdot 10^{0}$	$1.169423 \cdot 10^{-1}$

Analizując otrzymane wyniki, możemy zauważyć, że początkowo wraz ze zmniejszaniem wartości h błąd bezwzględny przybliżenia pochodnej również się zmniejsza. Jednak od pewnego momentu (około $h = 2^{-27}$) błąd zaczyna rosnąć wraz ze zmniejszaniem wartości h. Dzieje się tak ze względu na błąd zaokrągleń, który zaczyna dominować nad błędem wynikającym z przybliżenia różnicowego. Gdy h jest bardzo małe, różnica $f(x_0 + h) - f(x_0)$ staje się bardzo mała i może być reprezentowana z ograniczoną precyzją arytmetyki zmiennoprzecinkowej, co prowadzi do znacznych błędów w obliczeniach.