Sprawozdanie z listy 1

Jarosław Socha
268463

21 października 2023

1 Liczby maszynowe

Zadanie dzieli się na 3 podproblemy wyznaczenia konkretnej liczby maszynowej:

- **Wyznaczanie epsilona maszynowego**
  Epsilon maszynowy to najmniejsza liczba ϵ, dla której $fl(1+\epsilon) > 1$. Oczekiwana wartość ϵ to dwukrotność względnego błędu arytmetyki 2^{-t}, czyli 2^{1-t}, gdzie t to liczba cyfr mantysy. W związku z tym, jako że wiemy, że ma być to potęga liczby 2, to wystarczy przyjąć $\epsilon = 1.0$ a następnie dzielić go przez 2 aż nieprawdą będzie, że $fl(1+\epsilon) > 1$ lub $fl(1+\epsilon) = 1+\epsilon$.
  Napisałem osobną procedurę dla każdego z typów Float16, Float32 i Float64. Wyniki:

```
$ julia 1/macheps16.jl
Wynik:       0.000977
0001010000000000

Sprawdzenie: 0.000977
0001010000000000
```

```
$ julia 1/macheps32.jl
Wynik:       1.1920929e-7
00110100000000000000000000000000
Sprawdzenie: 1.1920929e-7
00110100000000000000000000000000
```

```
$ julia 1/macheps64.jl
Wynik:       2.220446049250313e-16
0011110010110000000000000000000000000000000000000000000000000000

Sprawdzenie: 2.220446049250313e-16
0011110010110000000000000000000000000000000000000000000000000000
```

  Sprawdzenie za pomocą funkcji *eps*.

- **Wyznaczanie liczby η**

  η to najmniejsza liczba większa od zera dla danego typu, znana również pod nazwą MIN_{sub}. Wyznaczamy ją dzieląc $x = 1.0$ przez 2 aż $fl(x/2) = 0$. Wyniki:

```
$ julia 1/eta16.jl
Wynik:       6.0e-8
0000000000000001

Sprawdzenie: 6.0e-8
0000000000000001

floatmin: 6.104e-5
0000010000000000
```

```
$ julia 1/eta32.jl
Wynik:       1.0e-45
00000000000000000000000000000001

Sprawdzenie: 1.0e-45
00000000000000000000000000000001

floatmin: 1.1754944e-38
00000000100000000000000000000000
```

```
$ julia 1/eta64.jl
Wynik:       5.0e-324
0000000000000000000000000000000000000000000000000000000000000001

Sprawdzenie: 5.0e-324
0000000000000000000000000000000000000000000000000000000000000001

floatmin: 2.2250738585072014e-308
0000000000010000000000000000000000000000000000000000000000000000
```

Wyniki sprawdzane przez $nextfloat(0.0)$. Wartość $floatmin$ danego typu to minimalna wartość znormalizowana, czyli MIN_{nor}.

- **Wyznaczanie liczby MAX**
  MAX jest największą liczbą reprezentowalną w danym typie, która nie jest nieskończonością. Procedura jej wyznaczenia wygląda następująco: Na początku $x = 1$ mnożymy przez 2 aż $x \cdot 2 = \infty$. Następnie ustalamy $\delta = \frac{x}{2}$ i dopóki nieprawdą jest, że $fl(x+\delta) = \infty$ to dodajemy δ do x a następnie zmniejszamy δ o połowę ($\delta = \frac{\delta}{2}$). Korzystamy z tego, że idąc od lewej do prawej w reprezentacji dwójkowej ciągu najpierw wypełniamy cechę maksymalną liczbą jedynek, a następnie mantysę. Wynik:

```
$ julia 1/max16.jl
Wynik:       6.55e4
0111101111111111

Sprawdzenie: 6.55e4
0111101111111111

```

```
$ julia 1/max32.jl
Wynik:       3.4028235e38
01111111011111111111111111111111

Sprawdzenie: 3.4028235e38
01111111011111111111111111111111
```

```
$ julia 1/max64.jl
Wynik:       1.7976931348623157e308
0111111111101111111111111111111111111111111111111111111111111111

Sprawdzenie: 1.7976931348623157e308
0111111111101111111111111111111111111111111111111111111111111111
```

  Sprawdzenie przy pomocy $floatmax$.

Wyniki zgadzają się z plikiem nagłówkowym $float.h$.

2 Kahan

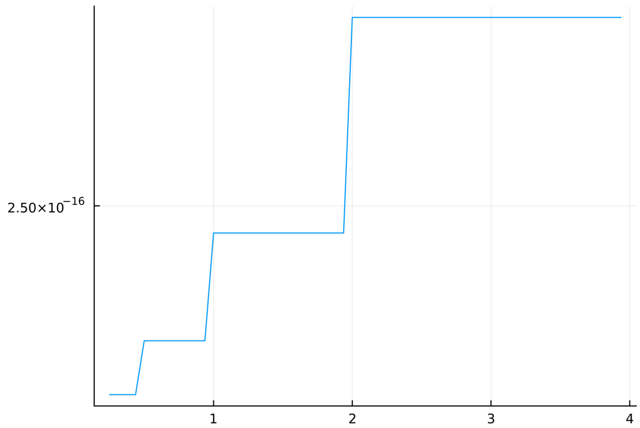
Celem zadania jest sprawdzenie, czy obliczając działanie $3 \cdot (\frac{4}{3} - 1) - 1$ w arytmetyce Float otrzymamy epsilon maszynowy. Wynik:

```
$ julia 2/Kahan.jl
Poprawny 16: 0.000977
Kahan 16:    -0.000977
Poprawny 32: 1.1920929e-7
Kahan 32:    1.1920929e-7
Poprawny 64: 2.220446049250313e-16
Kahan 64:    -2.220446049250313e-16
```

Tak, otrzymujemy poprawny epsilon, choć czasami ma on inny znak.

3 Odstępy

Celem zadania jest sprawdzić odstępy między kolejnymi liczbami w przedziale. Odstęp taki wynosi 2^{m-t}, gdzie m to liczba którą tworzy cecha, a t to liczba cyfr mantysy. Spodziewamy się że odległość między kolejnymi liczbami w przedziałach $[2^a, 2^{a+1}]$ będzie zależała wykładniczo od a (podwajała się z każdym nowym przedziałem). Wynik:



Rysunek 1: Odstęp na danym przedziale

Wynik zgadza się z przewidywaniami, liczby są równo rozmieszczone w danych przedziałach, a odstępy na dwukrotnie większych przedziałach również rosną dwukrotnie.

4 Mnożenie przez odwrotność

Celem jest znaleźć najmniejszą liczbę, dla której ta liczba pomnożona przez swoją odwrotność jest różna od 1, czyli takie x, że $fl(x \cdot fl(\frac{1}{x})) \neq 1$. Iterujemy po kolejnych liczbach od 1 w górę dodając epsilon obliczony w zadaniu poprzednim (2^{-52}) aż nie spełnimy nierówności. Wynik:

```
$ julia 4/smallest.jl
X to 1.000000057228997
1/X to 0.9999999427710061
1/x * x to 0.9999999999999999
```

5 Iloczyn skalarny

Zadanie ma na celu sprawdzić różnice w iloczynie skalarnym prawie prostopadłych wektorów w zależności od kolejności liczenia. Badane kolejności to:

1. w kolejności rosnącej (po indeksach)
2. w kolejności malejącej (po indeksach)
3. od najmniejszego do największego osobno dla ujemnych i dodatnich
4. odwrotnie do 3.

Wynik:

```
$ julia 5/dot32.jl
1) -0.12593332
2) -0.23105995
3) 5.510957e6
4) 4008.25
```

```
$ julia 5/dot64.jl
1) 1.0251881368296672e-10
2) -1.5643308870494366e-10
3) 5.510957390881553e6
4) 4008.4028031835333
```

Wartość dokładna to $1.00657107000000 \cdot 10^{11}$. Zadanie jest źle uwarunkowane dla prawie prostopadłych wektorów, więc trudnym jest uzyskać bliski wynik. Mała precyzja Float32 sprawia, że żaden wynik nie jest blisko, widać jednak że w dwóch ostatnich punktach krok odjęcia od siebie dwóch podobnych liczb tworzy bardzo duży błąd.

6 Błąd odejmowania

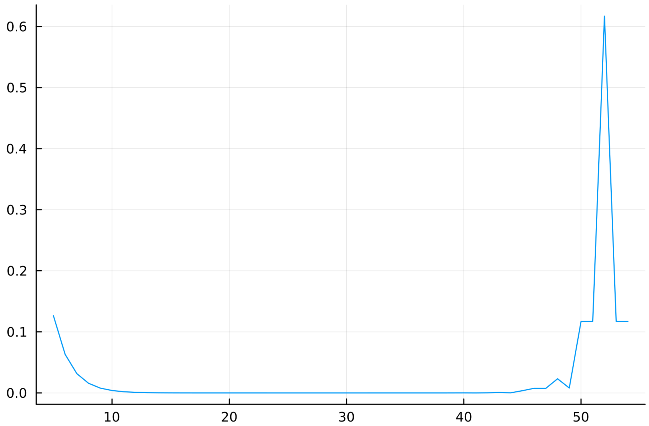
Celem jest zobaczyć błąd odejmowania bliskich sobie liczb, na przykładzie wyrażeń $\sqrt{x^2+1}-1$ i $\frac{x^2}{\sqrt{x^2+1}+1}$, które powinny dawać te same wyniki. Wynik:

```
$ julia 6/func.jl
[0.0077822185373186414, 0.00012206286282867573, 1.9073468138230965e-6, 2.9802321943606103e-8,
4.656612873077393e-10, 7.275957614183426e-12, 1.1368683772161603e-13, 1.7763568394002505e-15,
0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0]
[0.0077822185373187065, 0.00012206286282875901, 1.907346813826566e-6, 2.9802321943606116e-8,
4.6566128719931904e-10, 7.275957614156956e-12, 1.1368683772160957e-13, 1.7763568394002489e-15,
2.7755575615628914e-17, 4.336808689942018e-19, 6.776263578034403e-21, 1.0587911840678754e-22,
1.6543612251060553e-24, 2.5849394142282115e-26, 4.0389678347315804e-28, 6.310887241768095e-30]
```

Jak widać, wartość pierwszego wyrażenia od pewnego momentu to 0, gdyż odejmujemy od siebie bardzo bliskie liczby

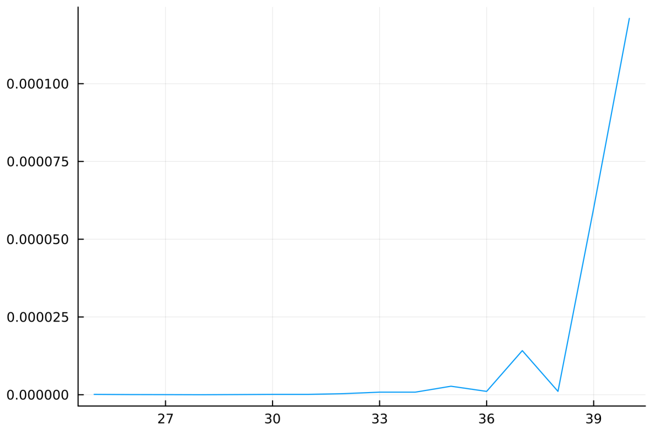
7 Pochodna

Celem jest zbadać błąd przy liczeniu pochodnej wzorem $\bar{f}'(x_0) \approx \frac{f(x_0+h)-f(x_0)}{h}$ badamy różnicę między przybliżeniem dla funkcji $f(x) = sin(x) + cos(3x)$, a wartością rzeczywistej pochodnej $f'(x) = cos(x) - 3sin(3x)$ dla $x_0 = 1$ i dla $h \in [2^0, 2^{-54}]$. Wynik:



Rysunek 2: Wykres błędu w zależności od h ($\bar{f}'(x_0) - f'(x_0)$)

Bardzo duże h tworzą duże błędy, a przy malutkich h powstają artefakty, więc aby zobaczyć moment, w którym przybliżenie odbiega od wartości, zawęzimy wartości h do przedziału $[2^{-25}, 2^{-40}]$



Rysunek 3: Wykres błędu w zależności od h dla mniejszego przedziału

Widać, że różnica zaczyna sie zwiększać już od $h = 2^{-32}$ i szybko rośnie. Są ku temu dwa powody, pierwszy to dodawanie do siebie dużej i małej liczby $(1.0 + h)$, a drugi to odejmowane od siebie bliskich sobie liczb $(f(1.0 + h) - f(1))$.