Sprawozdanie z drugiej listy zadań na laboratorium *Obliczenia Naukowe*

Paweł Rubin

Listopad 2019

1 Zadanie 1

1.1 Opis zadania

Zadanie polegało na ponownym obliczeniu iloczynu skalarnego z piątego zadania z listy pierwszej ze zmienionymi danymi oraz zaobserwowanie różnic w wynikach.

1.2 Wyniki

Tabele 1 oraz 2 przedstawiają wyniki dla arytmetyki `Float64` oraz `Float32`

algorytm	stary wynik	nowy wynik
a	1.0251881368296672e-10	-0.004296342739891585
b	-1.5643308870494366e-10	-0.004296342998713953
c	0.0	-0.004296342842280865
d	0.0	-0.004296342842280865

Tabela 1: Wyniki dla `Float64`

algorytm	stary wynik	nowy wynik
a	-0.3472038161853561	-0.3472038161889941
b	-0.4543457	-0.4543457
c	-0.5	-0.5
d	-0.5	-0.5

Tabela 2: Wyniki dla `Float32`

1.3 Wnioski

W arytmetyce Float64 wszystkie algorytmy zwróciły poprawny wynik - w przeciwieństwie do sytuacji z poprzedniej listy zadań. Można wywnioskować, że algorytmy są poprawne, ale zadanie zostało źle uwarunkowane.

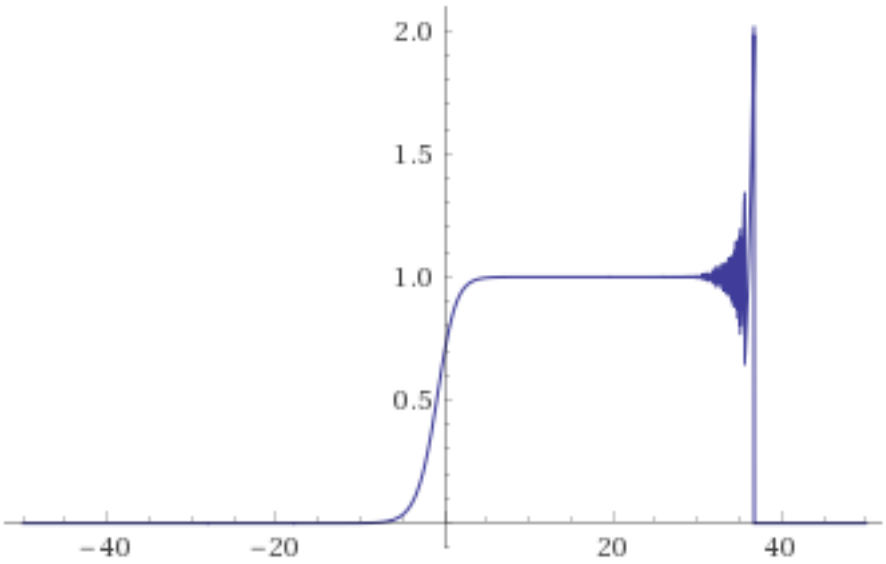
2 Zadanie 2

2.1 Opis zadania

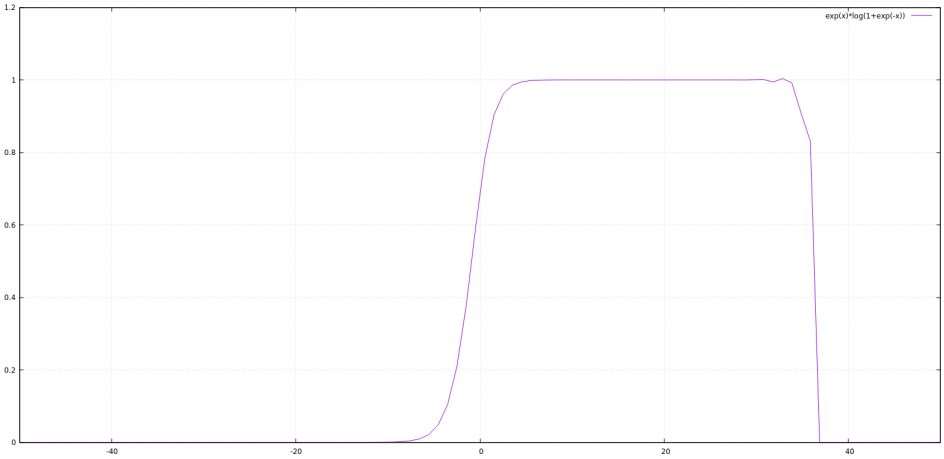
Zadanie polegało na narysowaniu wykresu funkcji $f(x) = e^x \ln(1 + e^{-x})$ w co najmniej dwóch dowolnych programach do wizualizacji. Następnie należało obliczyć granicę $\lim_{x \to +\infty} f(x)$ oraz porównać otrzymany wynik z wykresami.

2.2 Wykresy

Wykresy 1 oraz 2 przedstawiają wykres funkcji $f(x) = e^x \ln(1 + e^{-x})$. Wygenerowane zostały za pomocą programów: wolframalpha oraz gnuplot.



Rysunek 1: Wolframalpha



Rysunek 2: gnuplot

2.3 Granica

Obliczmy granicę funkcji $f(x) = e^x \ln(1 + e^{-x})$.

$$\lim_{x\to+\infty} f(x) = \lim_{x\to+\infty} \frac{\ln(1+e^{-x})}{e^{-x}} = \lim_{x\to+\infty} \frac{e^{-x}}{(1+e^{-x})e^{-x}} = \lim_{x\to+\infty} \frac{1}{1+e^{-x}} = 1$$

2.4 Wnioski

Zaobserwujmy, że wykresy nie zgadzają się z wyliczoną granicą. Użyte programy nie poradziły sobie z błędami obliczeń wartość e^{-x} dąży do wartości poniżej epsilonu maszynowego. Możemy wywnioskować, że napotkany problem numeryczny jest trudny do rozwiązania.

3 Zadanie 3

3.1 Opis zadania

Zadanie polegało na rozwiązywaniu układów równań liniowych postaci $\mathbf{A}x = b$ dla macierzy Hilberta stopnia n oraz losowej macierzy stopnia n z zadanym wskaźnikiem uwarunkowania.

3.2 Wyniki

Tabele 3 oraz 4 przedstawiają otrzymane wyniki odpowiednio dla macierzy Hilberta oraz macierzy losowej.

n	rank(A)	cond(A)	gauss(A) error	inversion(A) error
1	1	1.0	0.0	0.0
3	3	524.056777586062	8.118051169482656e-15	1.7907430486334138e-14
5	5	476607.2502421033	2.8186181571329407e-13	3.654697861370179e-12
7	7	4.753673559839011e8	7.410208490784287e-9	1.1004084486340423e-8
9	9	4.931538348163301e11	1.0485260733621061e-5	5.958958489267314e-6
11	10	5.219813567997335e14	0.004840612082131825	0.006720407723584139
13	11	2.376786926717726e18	0.8773261344142039	1.5119571944035046
15	12	2.4022870188034854e17	35.372359774035836	31.13950131676681
17	12	4.7507771590947725e17	14.381303216697646	15.299960220214384
19	13	4.215819618491786e18	7.007414463759739	16.272559569359128
21	13	1.1722320697256827e18	24.901674675482468	30.69512419621628
23	13	1.381128625309956e18	26.757380735394854	17.756487444686794
25	13	1.5473037464050412e19	10.684594541709313	16.76781301728111
27	14	6.557147810863727e18	10.35173741375024	40.23120101224578
29	14	5.422106953279447e19	32.705177271887095	61.47621691372046

Tabela 3: Wyniki dla macierzy Hilberta

3.3 Wnioski

Błędy dla macierzy Hilberta implikują, że dla owej macierzy zadanie to jest źle uwarunkowane. Oba algorytmy cechują się wysokimi błędami dla dużych danych. Z obliczonego wskaźnika

n	rank(A)	cond(A)	gauss(A) error	inversion(A) error
5	5	1.0000000000000007	1.5700924586837752e-16	2.482534153247273e-16
5	5	10.000000000000002	2.808666774861361e-16	9.930136612989092e-17
5	5	999.9999999999652	5.391158871405725e-15	4.282065584006262e-15
5	5	9.999999992296916e6	1.2460181490526317e-10	1.0173998942504929e-10
5	5	9.999324054897975e11	3.3689904615933196e-5	3.313279762026277e-5
5	4	1.4054769696761516e16	0.43009768671879184	0.32593761486264483
10	10	1.0000000000000007	2.4575834280036907e-16	2.432376777795247e-16
10	10	10.000000000000004	4.775249788392736e-16	3.7155169009665004e-16
10	10	999.9999999999384	3.100684163596976e-15	2.501330256990098e-15
10	10	1.0000000005499553e7	9.627345504191591e-11	8.226092940106949e-11
10	10	9.999695765505404e11	3.340050683621052e-6	3.441768800327365e-6
10	9	7.40217544391983e15	0.16574199715778518	0.19711419078024486
20	20	1.0000000000000009	4.697175049207787e-16	4.516575026604739e-16
20	20	9.99999999999999	5.461575137144e-16	5.879991017139367e-16
20	20	1000.0000000000329	1.5535586861869665e-14	1.8184372033248764e-14
20	20	9.999999995949227e6	4.225432714706223e-10	4.39192647792817e-10
20	20	9.999964704437794e11	1.654828321161298e-5	6.295830193619524e-6
20	19	8.345699327558959e15	0.16204647830195226	0.155839887138793

Tabela 4: Wyniki dla losowej macierzy

uwarunkowania macierzy można oszacować błąd wyniku.

4 Zadanie 4

4.1 Opis zadania

Zadanie polegało na zapoznaniu się z biblioteką `Polynomials` oraz z wielomianem Wilkinsona.

$$wilkison(x) = \prod_{i=1}^{20}(x-i)$$

4.2 Rozwiązanie

Zadanie zostało rozwiązane za pomocą funkcji z biblioteki `Polynomials`.

1. `Poly(a::Vector)` - Construct a polynomial from its coefficients, lowest order first.
2. `poly(r::AbstractVector)` - Construct a polynomial from its roots. This is in contrast to the `Poly` constructor, which constructs a polynomial from its coefficients.
3. `roots(p::Poly)` - Return the roots (zeros) of p, with multiplicity. The number of roots returned is equal to the order of p. By design, this is not type-stable, the returned roots may be real or complex.
4. `polyval(p::Poly, x::Number)` - Evaluate the polynomial p at x.

k	z_k	$\lvert P(z_k)\rvert$	$\lvert p(z_k)\rvert$	$\lvert z_k - k\rvert$
1	0.9999999999999977	512.0	2560.0	2.3314683517128287e-15
2	1.9999999999967497	29696.0	13312.0	3.2502889268926083e-12
3	2.999999999866293	52736.0	20992.0	1.337068233908667e-10
4	4.000000011019143	681472.0	943616.0	1.1019142931445458e-8
5	4.999999748927312	5.4272e6	6.1952e6	2.510726879734193e-7
6	6.000003401569265	1.6007168e7	1.7334272e7	3.4015692653710516e-6
7	6.999966502694603	7.9131648e7	8.123904e7	3.349730539703444e-5
8	8.000248650417012	6.33514496e8	6.377088e8	0.0002486504170118309
9	8.99863071666689	1.982439424e9	1.988410368e9	0.0013692833331102605
10	10.005643046417331	7.538121216e9	7.548366848e9	0.005643046417331377
11	10.9832728819987	2.1405586432e10	2.1419172352e10	0.016727118001300667
12	12.039704742162508	5.8907896832e10	5.8929570816e10	0.039704742162507856
13	12.93541042916589	1.39891236352e11	1.3992241664e11	0.06458957083411043
14	14.086351318967639	3.61559837184e11	3.6160536064e11	0.08635131896763859
15	14.916551086146429	7.63436727296e11	7.63491395072e11	0.08344891385357123
16	16.056101927823747	1.876228873728e12	1.87629699072e12	0.05610192782374668
17	16.970222485080672	3.779840352256e12	3.779930002432e12	0.029777514919327785
18	18.009738396580197	7.442458112e12	7.442565720576e12	0.009738396580196707
19	18.99796473418517	1.4829895143936e13	1.4830035581952e13	0.002035265814829046
20	20.000189920314238	2.1896970131456e13	2.1897133972992e13	0.00018992031423792355

Tabela 5: Wyniki dla wielomianu Wilkinsona

4.3 Wyniki

Tabele 5 oraz 6 przedstawiają uzyskane wyniki.

4.4 Wnioski

Z otrzymanych wyników można wywnioskować, że zadanie jest źle uwarunkowane - zaburzone współczynniki. Mała zmiana w danych generuje duży błąd. Błędy wynikają z braku dokładności w arytmetyce Float64.

5 Zadanie 5

5.1 Opis zadania

Zadanie polegało na liczeniu wartości następującego równania rekurencyjnego:

$$p_{n+1} = p_n + rp_n(1 - p_n)$$

- 40 iteracji w arytmetyce Float32
- 40 iteracji w arytmetyce Float32 z obcięciem do trzech miejsc po przecinku do 10. iterację
- 40 iteracji w arytmetyce Float64

k	z_k
1	1.000000000000076 + 0.0im
2	1.9999999999888531 + 0.0im
3	3.0000000005321574 + 0.0im
4	3.9999999854321384 + 0.0im
5	5.000000238159716 + 0.0im
6	6.000002465516928 + 0.0im
7	6.999740236517096 + 0.0im
8	8.006972846450305 + 0.0im
9	8.91828053511307 + 0.0im
10	10.095261779402826 - 0.6421736837273344im
11	10.095261779402826 + 0.6421736837273344im
12	11.793382133721813 - 1.652045904899795im
13	11.793382133721813 + 1.652045904899795im
14	13.992264932675088 - 2.5187943499538363im
15	13.992264932675088 + 2.5187943499538363im
16	16.730710830243574 - 2.812619231321066im
17	16.730710830243574 + 2.812619231321066im
18	19.502430503477196 - 1.9403277005007826im
19	19.502430503477196 + 1.9403277005007826im
20	20.84690345245782 + 0.0im

Tabela 6: Wyniki dla wielomianu Wilkinsona przy zmienionym współczynniku przy x^{19}

5.2 Wyniki

Tabela 7 przedstawia uzyskane wyniki.

Typ	wynik
Float32, obcięcie	0.71587336
Float32	0.25860548
Float64	0.011611238029748606

Tabela 7: Wyniki eksperymentów

5.3 Wnioski

Powyższe obliczenia pokazują jak ważna jest dokładność reprezentacji, gdyż potrafi ona mieć kolosalny wpływ na uzyskane wyniki obliczeń.

6 Zadanie 6

6.1 Opis zadania

W tym zadaniu również badamy równanie rekurencyjne, mianowicie:

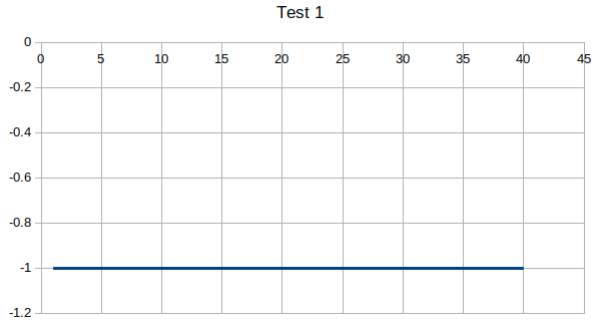
$$x_{n+1} = x_n^2 + c$$

dla następujących wartości parametru c oraz x_0:

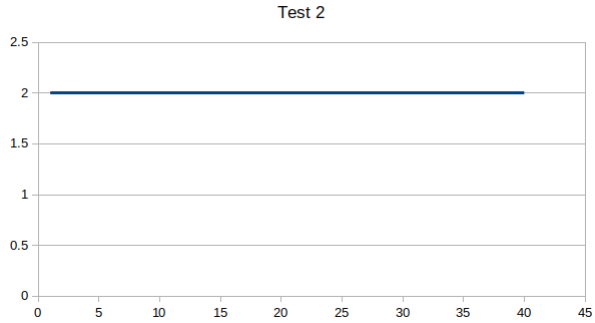
- Test 1: $c = -2$, $x_0 = 1$
- Test 2: $c = -2$, $x_0 = 2$
- Test 3: $c = -2$, $x_0 = 1.99999999999$
- Test 4: $c = -1$, $x_0 = 1$
- Test 5: $c = -1$, $x_0 = -1$
- Test 6: $c = -1$, $x_0 = 0.75$
- Test 7: $c = -1$, $x_0 = 0.25$

6.2 Wyniki

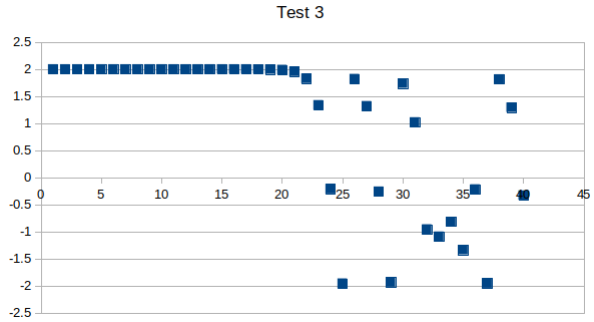
Wykresy 3, 4, 5, 6, 7, 8 oraz 9 przedstawiają wyniki przeprowadzonych eksperymentów.



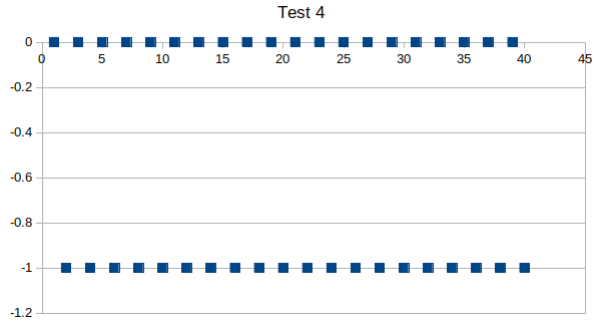
Rysunek 3: Test 1



Rysunek 4: Test 2



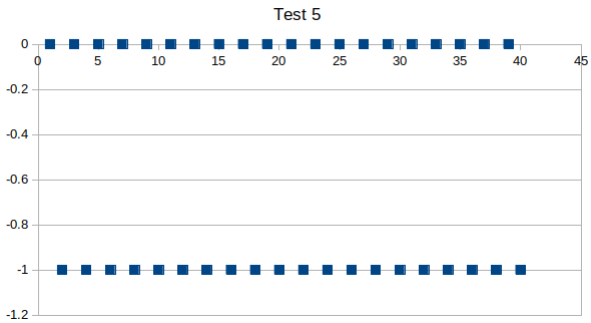
Rysunek 5: Test 3



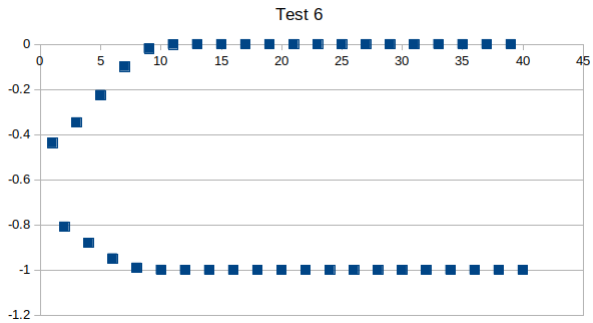
Rysunek 6: Test 4

6.3 Wnioski

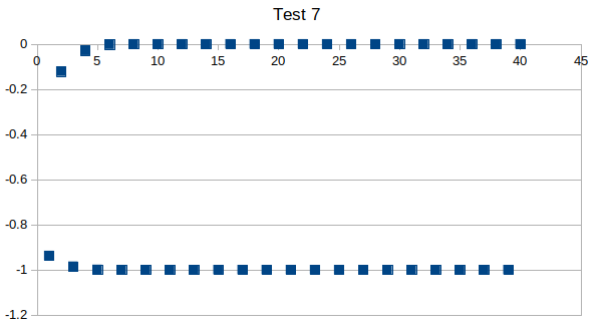
Błędy numeryczne narastają z każdą wynonaną iteracją. Nawet mała zmiana danych znacząco wpływa na wynik.



Rysunek 7: Test 5



Rysunek 8: Test 6



Rysunek 9: Test 7