Sprawozdanie

Nie interesuj się

15 kwietnia 2020

Zadanie Pierwsze

Cel: Zbadanie wpływu niewielkich zmian danych na końcowy wynik.

Przebieg doświadczenia: Badanie będzimy przeprowadzać na przerobionym już wcześnie przykładzie z listy pierwszej. Doświadczenie przeprowadzone poprzednio polegało na obliczeniu iloczynu skalarnego dla dwóch zadanych wektorów

$A = [2.718281828, -3.141592654, 1.414213562, 0.5772156649, 0.3010299957]$
$B = [1486.2497, 878366.9879, -22.37492, 4773714.647, 0.000185049]$

Tym razem zminiejszymy dokładność, poprzez pozbycie się liczb znaczących w A_4 (pozbywamy się najmniejszej cyfry części ułamkowej - '9'), oraz A_5 (pozbywamy się ostatniej cyfry części ułamkowej - '7'). Po dokonaniu obliczeń porównamy je z poprzednimi wynikami.

Poprzednim razem badania przeprowadzaliśmy dla czterech różnych algorytmów, tym razem także je wykorzystamy, są to:

1. Algorytm 1 ("w przód") - przejście wprost po wektorze, sumując kolejne wyrazy, zaczynając od początku tablicy
2. Algorytm 2 ("w tył") - przejście wprost po wektorze, sumując kolejne wyrazy, zaczynając od końca tablicy
3. Algorytm 3 ("od najmniejszego do największego") - posortowanie iloczynów rosnąco a następnie dodanie ich po kolei do dwóch akumulatorów, jeden dla liczb dodatnich, drugi dla liczb ujemnych a ostatecznie zsumowanie obu akumulatorów.
4. Algorytm 4 ("od największego do najmniejszego") - posortowanie iloczynów malejąco a następnie dodanie ich po kolei do dwóch akumulatorów, jeden dla liczb dodatnich, drugi dla liczb ujemnych a ostatecznie zsumowanie obu akumulatorów.

Wyniki:

Poprzednio otrzymane wyniki przedstawiają się następująco:

Algorytm	**Float32**	**Float64**
1	-0.4999443	$1.0251881368296672 * 10^{-10}$
2	-0.4543457	$-1.5643308870494366 * 10^{-10}$
3	-0.5	0.0
4	-0.5	0.0

Aktualne wyniki po usunięciu w/w cyfr:

Algorytm	**Float32**	**Float64**
1	-0.4999443	-0.004296342739891585
2	-0.4543457	-0.004296342998713953
3	-0.5	-0.004296342842280865
4	-0.5	-0.004296342842280865

Wnioski:

W przypadku Float32, zmiana nie miała wpływu na wynik ze względu na zbyt małą precyzję obliczeń. W przypadku dwukrotnego zwiększenia dokładności (Float64), otrzymaliśmy dużą różnicę w wynikach. Nie otrzymaliśmy już zer, a wyniki poszczególnych algorytmów są teraz do siebie zbliżone. Zmiana rzędu 10^{-9} spowodowała zmiejszenie ortogonalności (prostopadłości) wektora. To zadanie jest przykładem zadania źle uwarunkowanego, w którym niewielka zmiana danych powoduje olbrzymie zmiany w wynikach. W tym przypadku jest to spowodowane prostopadłością wektorów.

Zadanie Drugie

Cel: Porównanie wykresów funkcji $f(x)$ z obliczoną granicą tejże funkcji.

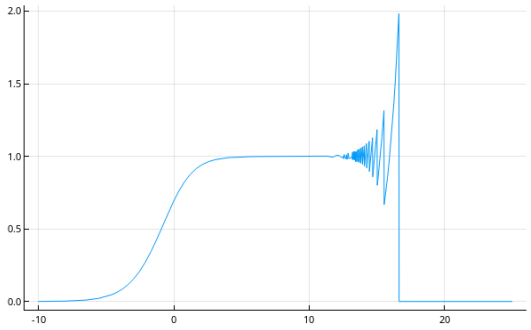
Funkcja:

$$f(x) = e^x * ln(1 + e^{-x})$$

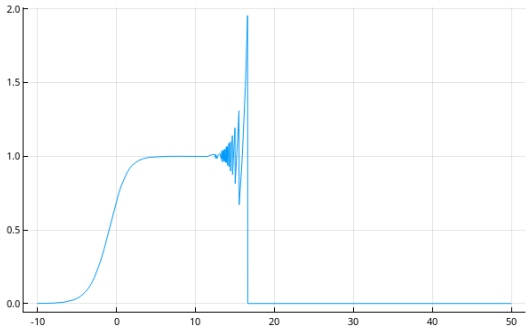
Obliczona pochodna:

$$\lim_{x\to\infty} e^x * ln(1 + e^{-x}) = \lim_{x\to\infty} \frac{ln(1 + e^{-x})}{e^{-x}} \overset{de'hospital}{=} \lim_{x\to\infty} \frac{1}{1 + e^{-x}} = 1$$

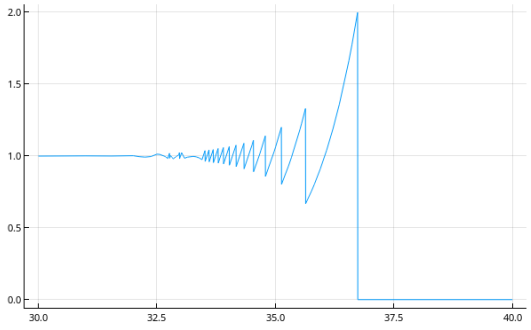
Wykresy:



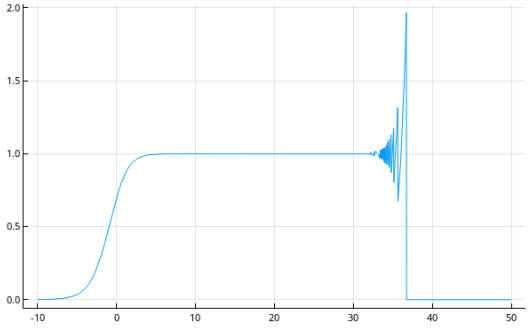
Float32, zakres [-10,25]



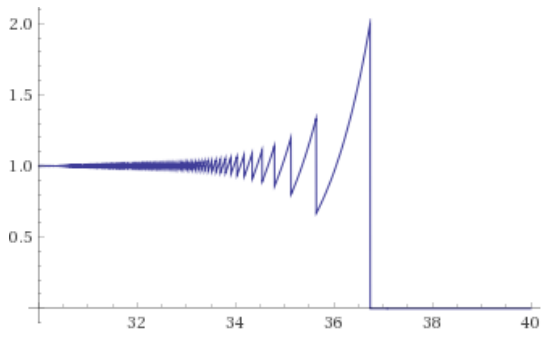
Float32, zakres [-10,50]



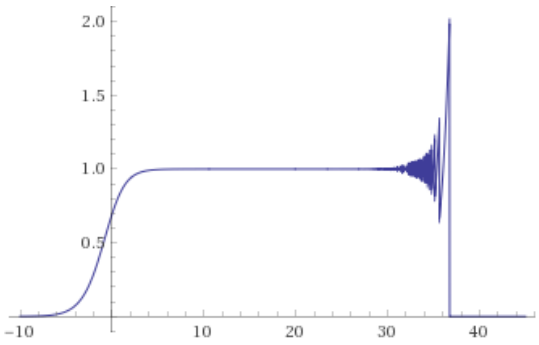
Float64, zakres [30,40]



Float64, zakres [-10,50]



WolframAlpha, zakres [30,40]

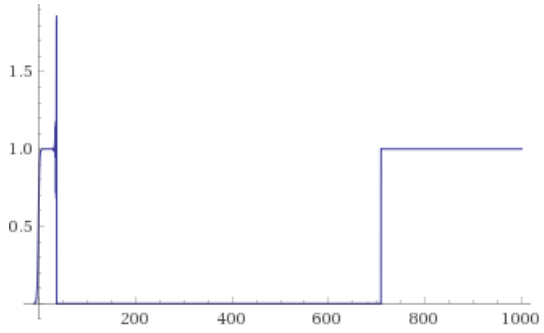


WolframAlpha, zakres [-10,45]

Wnioski:

Otrzymane wykresy różnią się w znacznym stopniu od oczekiwanego rezultatu. Oczekiwany wykres miał być zbieżny do jedynki, podczas gdy na otrzymanych wykresach widoczne są zaburzenia. Jest to spowodowane pochłanianiem małych wartości e^{-x} przez jedynkę. Dodatkowo, w momencie w którym $1 + e^{-x} = 1$ ($e^{-x} = 0$ dla zadanej precyzji) to $ln(1 + e^{-x}) = 0$.

Bardzo ciekawy wypadek zauważa się w przypadku użycia wolframalpha, bo po pewnym momencie zwraca on „nagle" poprawny wynik:



WolframAlpha, zakres [-10,1000]

Może być to spowodowane faktem iż wolfram posiada wiele sposobów obliczania pochodnych, więc kiedy kończy mu się zakres zmiennej, przechodzi na inną metodę, która podaje już dobry wynik.

Zadanie Trzecie

Cel: Rozwiązanie układu równań liniowych

Funkcja:

$$Ax = b$$

gdzie:
A - macierz, taka że $A \in \mathbb{R}^{n \times n}$
b - wektor prawych stron $\mathbf{b} \in \mathbb{R}^n$

Wyniki:

Dla macierzy Hilberta:

n	**rząd**	**wskaźnik uwarunkowania**	**błąd Gaussa**	**błąd macierzy odwrotnej**
1	1	1.0	0.0	0.0
2	2	19.28147006790397	$4.227603326225575e-16$	$1.4043333874306803e-15$
3	3	524.0567775860644	$6.312995352117186e-16$	0.0
4	4	15513.73873892924	$2.1819484005185015e-15$	$4.547473508864641e-13$
5	5	476607.25024259434	$8.99737540851766e-12$	$1.4911297558868156e-11$
6	6	$1.4951058642254665e7$	$1.8866554820446764e-10$	$3.5689314550268525e-10$
7	7	$4.75367356583129e8$	$1.8614175017153493e-8$	$1.231617281152939e-8$
8	8	$1.5257575538060041e10$	$1.9217530132542664e-7$	$1.3503856270597747e-7$
9	9	$4.931537564468762e11$	$5.09208671414057e-6$	$9.542631496737485e-6$
10	10	$1.6024416992541715e13$	$5.508778842434553e-5$	0.0002289249459044258
11	10	$5.222677939280335e14$	0.005207147400142043	0.008849600746399502
12	10	$1.7514731907091464e16$	0.1407902677571603	0.4039460424541017
13	11	$3.344143497338461e18$	355.6363996139929	212.78257870436312
14	11	$6.200786263161444e17$	3.5720053119125916	1.7744875020766255
15	11	$3.674392953467974e17$	9.583022643862336	5.293638550842115
16	12	$7.865467778431645e17$	11.968066192267113	10.846229140698135
17	12	$1.263684342666052e18$	5.455797564762254	8.982730231514317
18	12	$2.2446309929189128e18$	26.309523680963647	20.038300611698062
19	13	$6.471953976541591e18$	22.59171398621913	15.359549749376397
20	13	$1.3553657908688225e18$	5.658540758579833	7.992554067235351

Dla macierzy losowej:

n	**rząd**	**wskaźnik uwarunkowania**	**błąd Gaussa**	**błąd macierzy odwrotnej**
5	5	1.0000000000000002	$2.895107444979072e-16$	$2.531698018113677e-16$
5	5	10.000000000000012	$3.2934537262255424e-16$	$2.808666774861361e-16$
5	5	999.9999999999957	$2.9918399161602327e-15$	$8.407255028117242e-15$
5	5	$9.999999998272197e6$	$6.098113193176361e-10$	$4.920721043551264e-10$
5	5	$1.0000569317412089e12$	$2.090745266471979e-5$	$2.2491352852287535e-5$
5	4	$6.163423754077854e15$	0.2055523184662983	0.059292706128157104
10	10	1.0000000000000007	$1.7901808365247238e-16$	$1.6088660122137096e-16$
10	10	9.999999999999996	$2.1355566272775288e-16$	$2.603703785810335e-16$
10	10	999.9999999999403	$9.215519889170636e-16$	$3.6404482152219 93e-15$
10	10	$1.0000000004304033e7$	$1.1869428686627522e-11$	$5.182444144318803e-11$
10	10	$1.0000237542007422e12$	$3.68036568648752e-5$	$3.532974000347535e-5$
10	9	$8.183587485979556e15$	0.11886587922969834	0.12082884330738251
20	20	1.0000000000000016	$4.0867669957713655e-16$	$4.6510262453954645e-16$
20	20	9.999999999999988	$5.822058658345461e-16$	$4.6510262453954645e-16$
20	20	1000.0000000000192	$2.7950391086350172e-14$	$2.958989099482218e-14$
20	20	$1.0000000002579732e7$	$6.2329496742127 94e-11$	$9.084754966320824e-11$
20	20	$1.0000894285296765e12$	$2.633880965470748e-5$	$2.4529436866417026e-5$
20	19	$9.36557482791752e15$	0.3098253119813586	0.30186113977303886

Wnioski:

Patrząc na wyniki otrzymane z obliczeń dla macierzy losowej możemy stwierdzić, że potwierdzają one wpływ wskaźnika uwarunkowania macierzy na wielkość błędu względnego. Są one proporcjonalne, tj. jeżeli rośnie wskaźnik uwarunkowania, rośnie także błąd względny. Każdy problem o dużym wskaźniku uwarunkowania nazywamy problemem źle uwarunkowanym.

Przykładem problemu źle uwarukowanego jest analizowana przez nas macierz Hilberta - wraz ze wzrostem wielkości macierzy rośnie także współczynnik uwarunkowania. Dodatkowo, z analizy wyników można wywnioskować że metoda Gaussa jest skuteczniejsza przy wyliczaniu błędu względnego dla macierzy Hilberta.

Zadanie Czwarte

Cel: Obliczyć 20 miejsc zerowych wielomianu p Wilkinsona ($p(x) = (x-20)(x-19)...(x-2)(x-1)$) w postaci naturalnej i sprawdzenie poprawności (błędu otrzymanych pierwiastków) a następnie doświadczenie powtórzyć zmieniając $(-210) * x^{19} \rightarrow (-210 - 2^{-23}) * x^{19}$, w zapisie binarnym zmiana wygląda nastepująco:

$$(-210) \rightarrow 11000000011010100100000000000000000000000\mathbf{0}00000000000000000000$$

$$(-210 - 2^{-23}) \rightarrow 11000000011010100100000000000000000000000\mathbf{1}00000000000000000000$$

Wyniki:

Dla macierzy Wilkinsona przed zaburzeniem:

k	z_k	$\|P(z_k)\|$	$\|p(z_k)$	$\|z_k - k$
1	0.9999999999996989	36352.0	38400.0	$3.0109248427834245e-13$
2	2.0000000000283182	181760.0	198144.0	$2.8318236644508943e-11$
3	2.9999999995920965	209408.0	301568.0	$4.0790348876384996e-10$
4	3.9999999837375317	$3.106816e6$	$2.844672e6$	$1.626246826091915e-8$
5	5.000000665769791	$2.4114688e7$	$2.3346688e7$	$6.657697912970661e-7$
6	5.999989245824773	$1.20152064e8$	$1.1882496e8$	$1.0754175226779239e-5$
7	7.000102002793008	$4.80398336e8$	$4.78290944e8$	0.00010200279300764947
8	7.999355829607762	$1.682691072e9$	$1.67849728e9$	0.0006441703922384079
9	9.002915294362053	$4.465326592e9$	$4.457859584e9$	0.002915294362052734
10	9.990413042481725	$1.2707126784e10$	$1.2696907264e10$	0.009586957518274986
11	11.025022932909318	$3.5759895552e10$	$3.5743469056e10$	0.025022932909317674
12	11.953283253846857	$7.216771584e10$	$7.2146650624e10$	0.04671674615314281
13	13.07431403244734	$2.15723629056e11$	$2.15696330752e11$	0.07431403244734014
14	13.914755591802127	$3.65383250944e11$	$3.653447936e11$	0.08524440819787316
15	15.075493799699476	$6.13987753472e11$	$6.13938415616e11$	0.07549379969947623
16	15.946286716607972	$1.555027751936e12$	$1.554961097216e12$	0.05371328339202819
17	17.025427146237412	$3.777623778304e12$	$3.777532946944e12$	0.025427146237412046
18	17.99092135271648	$7.199554861056e12$	$7.1994474752e12$	0.009078647283519814
19	19.00190981829944	$1.0278376162816e13$	$1.0278235656704e13$	0.0019098182994383706
20	19.999809291236637	$2.7462952745472e13$	$2.7462788907008e13$	0.00019070876336257925

Dla macierzy Wilkinsona po zaburzeniu

k	z_k	$\|P(z_k)\|$	$\|p(z_k)$	$\|z_k - k$
1	0.9999999999998357	20992.0	22016.0	$1.6431300764452317e-13$
2	2.0000000000550373	349184.0	365568.0	$5.503730804434781e-11$
3	2.99999999660342	$2.221568e6$	$2.295296e6$	$3.3965799062229962e-9$
4	4.000000089724362	$1.046784e7$	$1.0729984e7$	$8.972436216225788e-8$
5	4.99999857388791	$3.9463936e7$	$4.3303936e7$	$1.4261120897529622e-6$
6	6.000020476673031	$1.29148416e8$	$2.06120448e8$	$2.0476673030955794e-5$
7	6.99960207042242	$3.88123136e8$	$1.757670912e9$	0.00039792957757978087
8	8.007772029099446	$1.072547328e9$	$1.8525486592e10$	0.007772029099445632
9	8.915816367932559	$3.065575424e9$	$1.37174317056e11$	0.0841836320674414
10	10.095455630535774	$7.143113638035824e9$	$1.4912633816754019e12$	0.6519586830380406
11	10.095455630535774	$7.143113638035824e9$	$1.4912633816754019e12$	1.1109180272716561
12	11.793890586174369	$3.357756113171857e10$	$3.2960214141301664e13$	1.665281290598479
13	11.793890586174369	$3.357756113171857e10$	$3.2960214141301664e13$	2.045820276678428
14	13.992406684487216	$1.0612064533081976e11$	$9.545941595183662e14$	2.5188358711909045
15	13.992406684487216	$1.0612064533081976e11$	$9.545941595183662e14$	2.7128805312847097
16	16.73074487979267	$3.315103475981763e11$	$2.7420894016764064e16$	2.9060018735375106
17	16.73074487979267	$3.315103475981763e11$	$2.7420894016764064e16$	2.825483521349608
18	19.5024423688181	$9.539424609817828e12$	$4.2525024879934694e17$	2.454021446312976
19	19.5024423688181	$9.539424609817828e12$	$4.2525024879934694e17$	2.004329444309949
20	20.84691021519479	$1.114453504512e13$	$1.3743733197249713e18$	0.8469102151947894

Wnioski:

Jak widać nawet minimalne odchylenie przekształca w bardzo duży wynik końcowy. W tym wielomianie zamiast zer otrzymujemy wyniki rzędu bilionów. Natomiast drobna zmiana powoduje ogromne różnice w wynikach między dwoma wielomianami (w obu nie uzyskujemy zer). To przypadek źle uwarunkowanego zadania. Warto pamiętać, że wyniki zależą od ilości cyfr znaczących w przypadku arytmetyki Float64 wynosi ona od 15 do 17 cyfr znaczących.

Zadanie Piąte

Cel: Badanie modelu wzrostu populacji p_n

Wzór rekurencyjny:

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

gdzie:
r - dana stała
p_0 - wielkość populacji stanowiąca procent maksymalnej wielkości populacji dla danego stanu środowiska.

Wyniki:

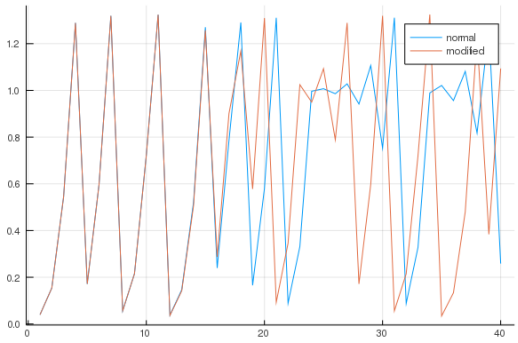
Dla obciętego wyniku (zmodyfikowana rekurencja) w 10 iteracji, oraz normalna rekurencja (Float32):

n	**czysta rekurencja**	**zmodyfikowana rekurencja**
1	0.0397	0.0397
2	0.15407173	0.15407173
3	0.5450726	0.5450726
4	1.2889781	1.2889781
5	0.1715188	0.1715188
6	0.5978191	0.5978191
7	1.3191134	1.3191134
8	0.056273222	0.056273222
9	0.21559286	0.21559286
10	0.7229306	0.722
11	1.3238364	1.3241479
12	0.037716985	0.036488414
13	0.14660022	0.14195944
14	0.521926	0.50738037
15	1.2704837	1.2572169
16	0.2395482	0.28708452
17	0.7860428	0.9010855
18	1.2905813	1.1684768
19	0.16552472	0.577893
20	0.5799036	1.3096911
21	1.3107498	0.09289217
22	0.088804245	0.34568182
23	0.3315584	1.0242395
24	0.9964407	0.94975823
25	1.0070806	1.0929108
26	0.9856885	0.7882812
27	1.0280086	1.2889631
28	0.9416294	0.17157483
29	1.1065198	0.59798557
30	0.7529209	1.3191822
31	1.3110139	0.05600393
32	0.0877831	0.21460639
33	0.3280148	0.7202578
34	0.9892781	1.3247173
35	1.021099	0.034241438
36	0.95646656	0.13344833
37	1.0813814	0.48036796
38	0.81736827	1.2292118
39	1.2652004	0.3839622
40	0.25860548	1.093568

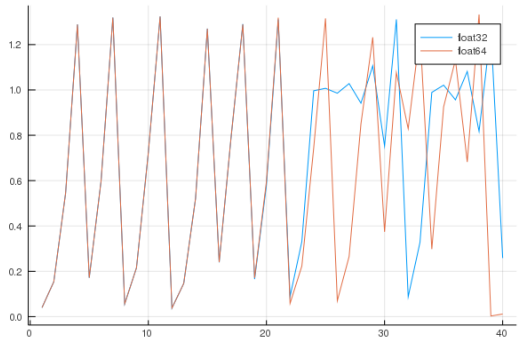
Porównawcze wyniki dla Float32 i Float64:

n	**Float32**	**Float64**
1	0.0397	0.0397
2	0.15407173	0.15407173000000002
3	0.5450726	0.5450726260444213
4	1.2889781	1.2889780011888006
5	0.1715188	0.17151914210917552
6	0.5978191	0.5978201201070994
7	1.3191134	1.3191137924137974
8	0.056273222	0.056271577646256565
9	0.21559286	0.21558683923263022
10	0.7229306	0.722914301179573
11	1.3238364	1.3238419441684408
12	0.037716985	0.03769529725473175
13	0.14660022	0.14651838271355924
14	0.521926	0.521670621435246
15	1.2704837	1.2702617739350768
16	0.2395482	0.24035217277824272
17	0.7860428	0.7881011902353041
18	1.2905813	1.2890943027903075
19	0.16552472	0.17108484670194324
20	0.5799036	0.5965293124946907
21	1.3107498	1.3185755879825978
22	0.088804245	0.058377608259430724
23	0.3315584	0.22328659759944824
24	0.9964407	0.7435756763951792
25	1.0070806	1.315588346001072
26	0.9856885	0.07003529560277899
27	1.0280086	0.26542635452061003
28	0.9416294	0.8503519690601384
29	1.1065198	1.2321124623871897
30	0.7529209	0.37414648963928676
31	1.3110139	1.0766291714289444
32	0.0877831	0.8291255674004515
33	0.3280148	1.2541546500504441
34	0.9892781	0.29790694147232066
35	1.021099	0.9253821285571046
36	0.95646656	1.1325322626697856
37	1.0813814	0.6822410727153098
38	0.81736827	1.3326056469620293
39	1.2652004	0.0029091569028512065
40	0.25860548	0.011611238029748606

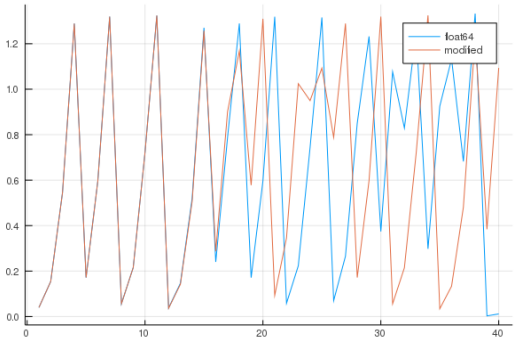
Wykresy:



Porównanie graficzne (tabela pierwsza)



Porównanie graficzne (tabela druga)



Porównanie graficzne float64 z ucietą rekurencją

Wnioski:

Widzimy, że wykonane przez nas ekperymenty mają związek ze sprzężeniem zwrotnym (dane wyjściowe przekładają się na dane wejściowe do następnego obliczenia z serii obliczeń). Oznacza to że nawt mała zmiana na początku ciągu obliczeń może powodować olbrzymie odchylenia w dalszych iteracjach. Ponadto widzimy punkt załamania w którym dana arytmetyka traci swoją precyzję i od tego punktu x dane bedą zaszumione. Główna różnica między trzema wynikami (obcięty float32, float32 i float64) jest położenie tego punktu na osi (0,x), które rośnie wraz z precyzją arytmetyki.

Zadanie Szóste

Cel: Badanie równanie rekurencyjnego

Wzór:

$$x_{n+1} = x_n^2 + c, dla\, n = 0, 1...$$

Wyniki:

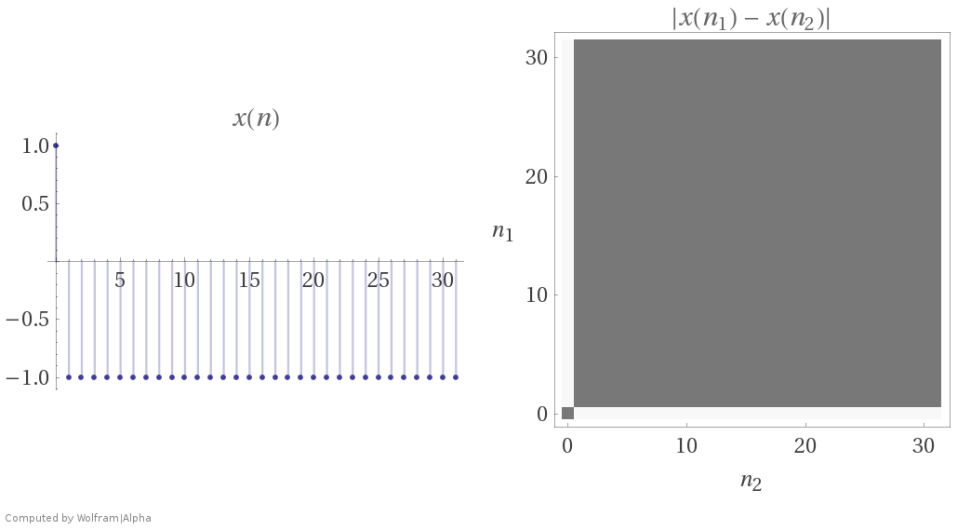
Dla $c = -2$:

iteracja	$x_0 = 1$	$x_0 = 2$	$x_0 = 1.99999999999999$
1	-1	2	1.99999999999996
2	-1	2	1.9999999999998401
3	-1	2	1.9999999999993605
4	-1	2	1.999999999997442
5	-1	2	1.9999999999897682
6	-1	2	1.9999999999590727
7	-1	2	1.999999999836291
8	-1	2	1.9999999993451638
9	-1	2	1.9999999973806553
10	-1	2	1.999999989522621
11	-1	2	1.9999999580904841
12	-1	2	1.9999998323619383
13	-1	2	1.9999993294477814
14	-1	2	1.9999973177915749
15	-1	2	1.9999892711734937
16	-1	2	1.9999570848090826
17	-1	2	1.999828341078044
18	-1	2	1.9993133937789613
19	-1	2	1.9972540465439481
20	-1	2	1.9890237264361752
21	-1	2	1.9562153843260486
22	-1	2	1.82677862987391
23	-1	2	1.3371201625639997
24	-1	2	-0.21210967086482313
25	-1	2	-1.9550094875256163
26	-1	2	1.822062096315173
27	-1	2	1.319910282828443
28	-1	2	-0.2578368452837396
29	-1	2	-1.9335201612141288
30	-1	2	1.7385002138215109
31	-1	2	1.0223829934574389
32	-1	2	-0.9547330146890065
33	-1	2	-1.0884848706628412
34	-1	2	-0.8152006863380978
35	-1	2	-1.3354478409938944
36	-1	2	-0.21657906398474625
37	-1	2	-1.953093509043491
38	-1	2	1.8145742550678174
39	-1	2	1.2926797271549244
40	-1	2	-0.3289791230026702

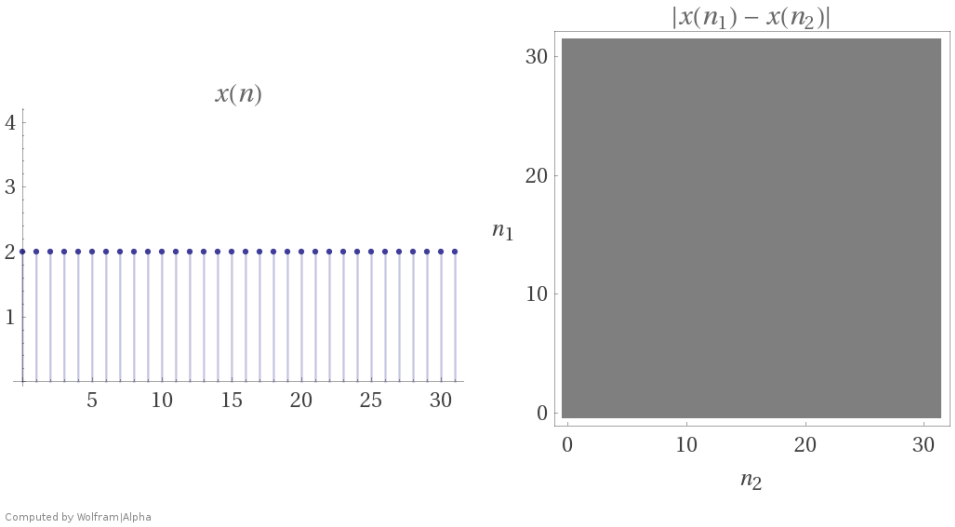
Dla $c = -1$:

iteracja	$x_0 = 1$	$x_0 = -1$	$x_0 = 0.75$	$x_0 = 0.25$
1	0	0	-0.4375	-0.9375
2	-1	-1	-0.80859375	-0.12109375
3	0	0	-0.3461761474609375	-0.9853363037109375
4	-1	-1	-0.8801620749291033	-0.029112368589267135
5	0	0	-0.2253147218564956	-0.9991524699951226
6	-1	-1	-0.9492332761147301	-0.0016943417026455965
7	0	0	-0.0989561875164966	-0.9999971292061947
8	-1	-1	-0.9902076729521999	$-5.741579369278327e-6$
9	0	0	-0.01948876442658909	-0.9999999999670343
10	-1	-1	-0.999620188061125	$-6.593148249578462e-11$
11	0	0	-0.0007594796206411569	-1.0
12	-1	-1	-0.9999994231907058	0.0
13	0	0	$-1.1536182557003727e-6$	-1.0
14	-1	-1	-0.9999999999986692	0.0
15	0	0	$-2.6616486792363503e-12$	-1.0
16	-1	-1	-1.0	0.0
17	0	0	0.0	-1.0
18	-1	-1	-1.0	0.0
19	0	0	0.0	-1.0
20	-1	-1	-1.0	0.0
21	0	0	0.0	-1.0
22	-1	-1	-1.0	0.0
23	0	0	0.0	-1.0
24	-1	-1	-1.0	0.0
25	0	0	0.0	-1.0
26	-1	-1	-1.0	0.0
27	0	0	0.0	-1.0
28	-1	-1	-1.0	0.0
29	0	0	0.0	-1.0
30	-1	-1	-1.0	0.0
31	0	0	0.0	-1.0
32	-1	-1	-1.0	0.0
33	0	0	0.0	-1.0
34	-1	-1	-1.0	0.0
35	0	0	0.0	-1.0
36	-1	-1	-1.0	0.0
37	0	0	0.0	-1.0
38	-1	-1	-1.0	0.0
39	0	0	0.0	-1.0
40	-1	-1	-1.0	0.0

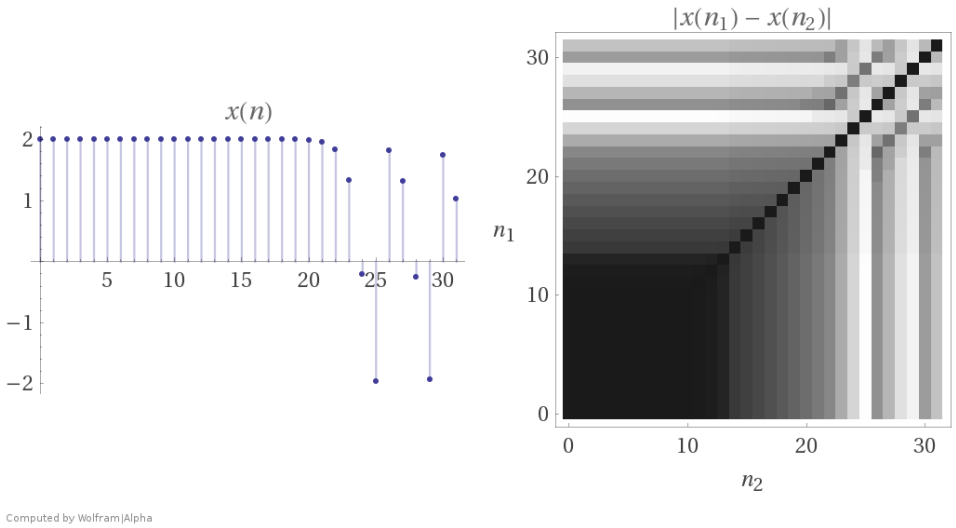
Wykresy:



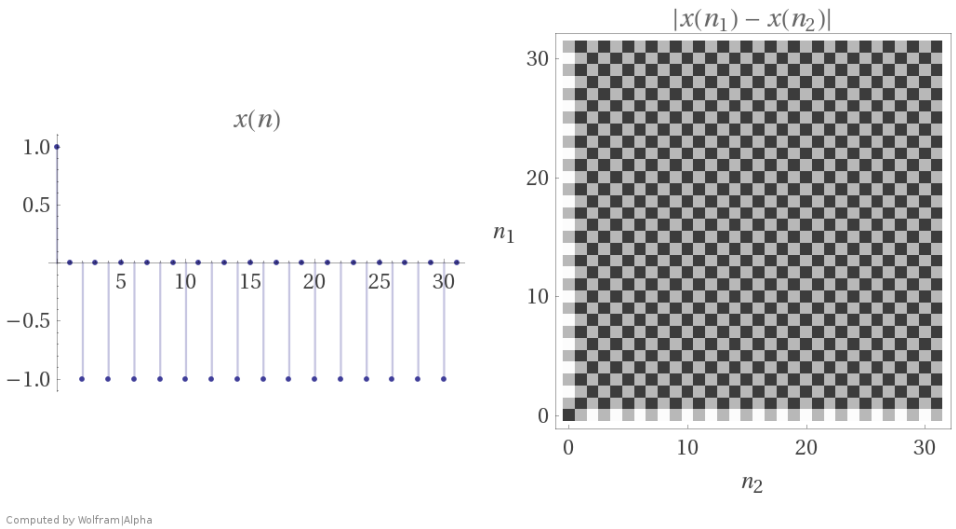
Graficzne przedstawienie $x_0 = -1$ i $c = -2$



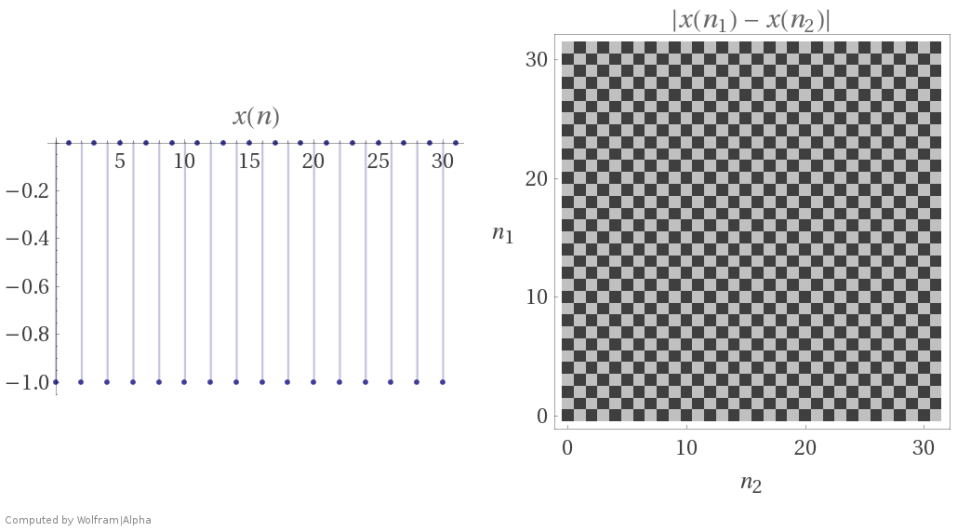
Graficzne przedstawienie $x_0 = 2$ i $c = -2$



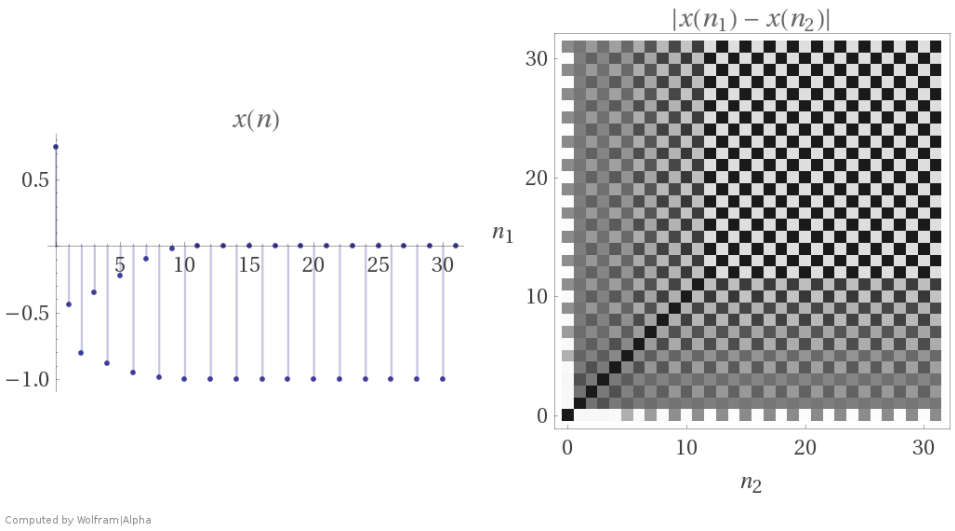
Graficzne przedstawienie $x_0 = -1.99999999999999$ i $c = -2$



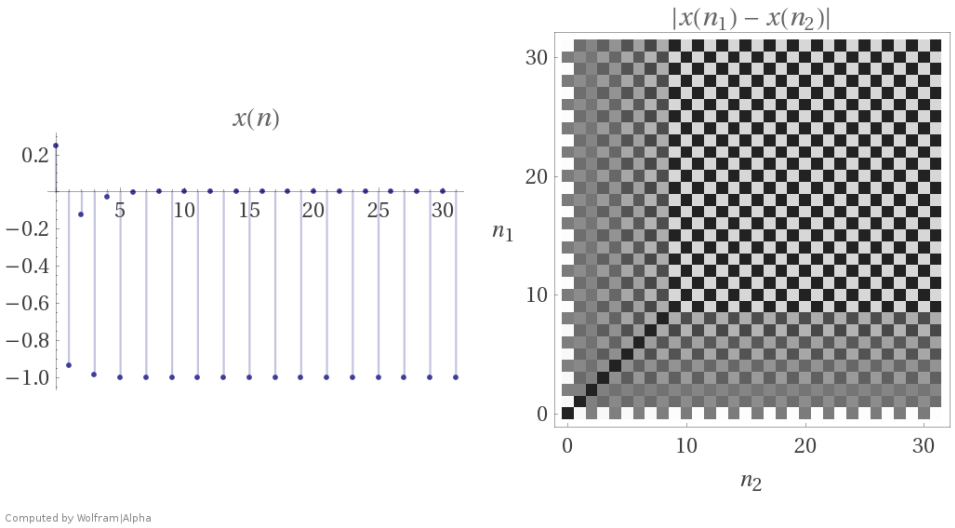
Graficzne przedstawienie $x_0 = 1$ i $c = -1$



Graficzne przedstawienie $x_0 = -1$ i $c = -1$



Graficzne przedstawienie $x_0 = 0.75$ i $c = -1$



Graficzne przedstawienie $x_0 = 0.25$ i $c = -1$

Wnioski:

Wykonane przez nas doświadczenie ma związek z pojęciem *układu stabilnego*. Stabilność układu określa to jak pewien układ matematyczny jest wrażliwy na zmiany danych. Dla $x_0 = -1, c = -2$ oraz $x_0 = 2, c = -2$ obserowany układ jest układem stabilnym, otrzymujemy ciągi zbieżne odpowiednio do -1 i 2. Dla $x_0 = -1.99999999999999, c = -2$ otrzymaliśmy układ niestabliny, od pewnego punktu otrzymane wyniki są losowe. Dla $c = -1$ każdy zaprezentowany układ jest stabilny, ponieważ mimo początkowej niestabilności, wraz we wzrostem iteracji tworzą się podciągi zbieżne.