Obliczenia naukowe - lab 2

Jakub Musiał 268442

Listopad 2023

Zadanie 1 - Iloczyn skalarny

Problem

Powtórzyć experyment z zadania 5 z listy 1 dla zmienionych danych.
Experyment - porównanie metod sumowania iloczynów w algorytmie obliczania iloczynu skalarnego (w przód, w tył, od największego do najmniejszego oraz od najmniejszego do największego) w pojedynczej oraz podwójnej precyzji.

Oryginalne wartości wektorów:

$$\begin{cases} x = [2.718281828, -3.141592654, 1.414213562, 0.5772156649, 0.3010299957] \\ y = [1486.2497, 878366.9879, -22.37492, 4773714.647, 0.000185049] \end{cases}$$

W tym eksperymencie modyfikacja danych to usunięcie ostatniej 9 z x_4 oraz ostatniej 7 z x_5.

Rozwiązanie

Program z rozwiązaniem: `ex1.jl`

Wyniki i obserwacje

W *tabelach 1 i 2* możemy zauważyć, że obliczone wartości dla pojedynczej precyzji są identyczne, co wynika z tego, że obcieliśmy wartości na 10 miejscu po przecinku, co przekracza zakres precyzji - $\varepsilon_{float32} = 2^{-24} \approx 6 \cdot 10^{-8}$.
Dla precyzji podwójnej (*tabele 3 i 4*) jednak możemy zaobserwować duże zmiany rzędów uzyskanych wartości - dla metod sumowania "w przód" i "w tył" jest to zmiana o 10^{10}.
To pokazuje, że, rozważając iloczyn skalarny wektorów prawie prostopadłych, bardzo małe modyfikacje danych wejściowych powodują duże zmiany wyników.
Zadanie jest więc źle uwarunkowane.

Algorytm	S	δ
W przód	-0.4999443	$4.966805766328285e10$
W tył	-0.4543457	$4.5137965582009605e10$
Od największych	-0.5	$4.967359135506108e10$
Od najmniejszych	-0.5	$4.967359135506108e10$

Table 1: Wartości iloczynu skalarnego wraz z błędem względnym w precyzji `Float32` (dane z zadania 5 z listy 1)

Algorytm	S	δ
W przód	-0.4999443	$4.966805766328285e10$
W tył	-0.4543457	$4.5137965582009605e10$
Od największych	-0.5	$4.967359135506108e10$
Od najmniejszych	-0.5	$4.967359135506108e10$

Table 2: Wartości iloczynu skalarnego wraz z błędem względnym w precyzji `Float32` dla zmienionych danych

Algorytm	S	δ
W przód	$1.0251881368296672e-10$	9.184955313981629
W tył	$-1.5643308870494366e-10$	16.54118664516592
Od największych	0.0	1.0
Od najmniejszych	0.0	1.0

Table 3: Wartości iloczynu skalarnego wraz z błędem względnym w precyzji `Float64` (dane z zadania 5 z listy 1)

Algorytm	S	δ
W przód	-0.004296342739891585	$4.2682954815672344e8$
W tył	-0.004296342998713953	$4.2682957386999655e8$
Od największych	-0.004296342842280865	$4.268295583288099e8$
Od najmniejszych	-0.004296342842280865	$4.268295583288099e8$

Table 4: Wartości iloczynu skalarnego wraz z błędem względnym w precyzji `Float64` dla zmienionych

Zadanie 2 - Badanie granicy funkcji

Problem

Obliczyć granicę funkcji $f(x) = e^x \cdot \ln(1 + e^{-x})$ i porównać uzyskany wynik z wygenerowanymi w 2 programach do wizualizacji wykresami zadanej funkcji.

Rozwiązanie

Policzmy granicę funkcji f w nieskończoności:

$$\lim_{x\to\infty} e^x \cdot \ln(1 + e^{-x}) = \lim_{x\to\infty} \frac{\ln(1+e^{-x})}{e^{-x}} \overset{\text{de l'Hospital}}{=} \lim_{x\to\infty} \frac{-\frac{e^{-x}}{1+e^{-x}}}{-e^{-x}} = \lim_{x\to\infty} \frac{1}{1+e^{-x}} = 1$$

Wizualizacje wykresów funkcji f wykonałem w programach:

- `Julia` - biblioteka `Plots` (`ex2.jl`)
- `Python` - biblioteka `matplotlib.pyplot` (`ex2.py`)

Wyniki i obserwacje

Na poniższych wykresach można zauważyć rozbieżność z wyliczoną wcześniej granicą funkcji. Zauważmy, że dla $x < x_0$ bliskich x_0 wartości funkcji są coraz bardziej rozbieżne oraz $(\forall x > x_0)(f(x) = 0)$.

To wynika z ograniczonej prezycji arytmetyki zmiennopozycyjnej. W podwójnej precyzji standardu `IEEE 754` mamy 52 bity na reprezentację mantysy.

Dla $e^{-x} < 2^{-52} \implies x > 52 \cdot \ln(2) \approx 36.04365 = x_0$ mamy więc:

$$\begin{gathered} fl(1 + e^{-x}) = 1 \\ \Downarrow \\ \text{fl}(\ln(1 + e^{-x})) = fl(ln(1)) = 0 \\ \Downarrow \\ \text{fl}(\text{f}(\text{x})) = \text{fl}(\text{e}^x \cdot \ln(1 + e^{-x})) = fl(e^x \cdot 0) = 0 \end{gathered}$$

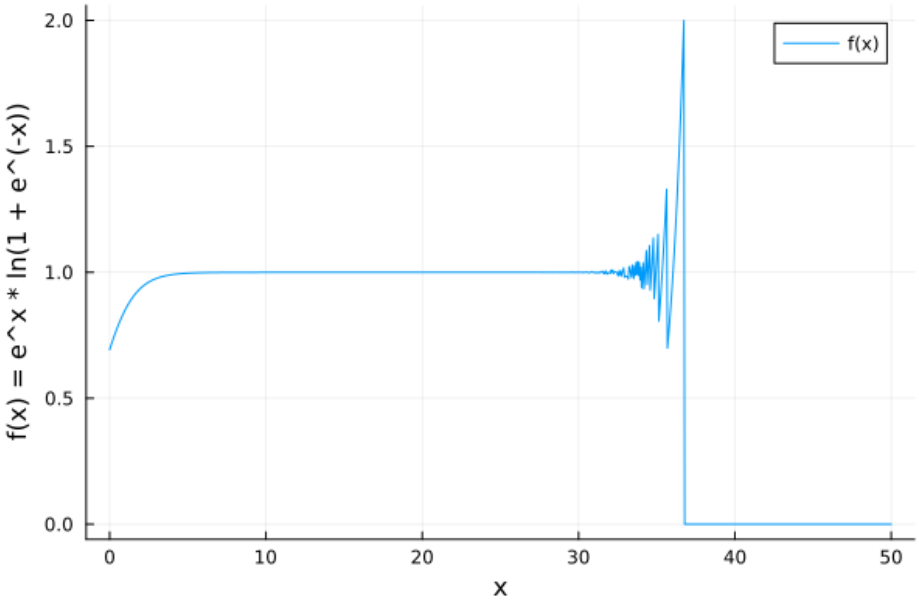


Figure 1: Wykres funkcji f uzyskany w języku `Julia`

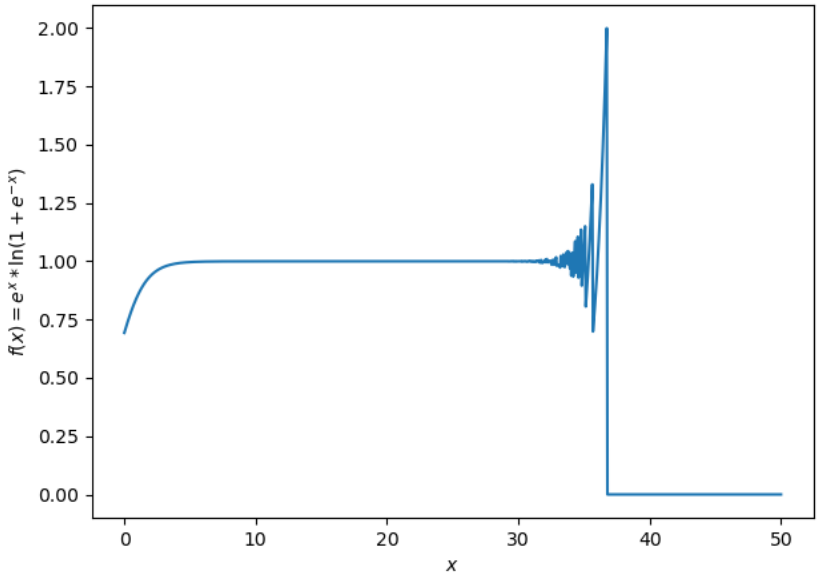


Figure 2: Wykres funkcji f uzyskany w języku `Python`

Zadanie 3 - Rozwiązywanie liniowych układów rownań

Problem

Rozwiązać układ równań liniowych $Ax = b$, gdzie $A \in \mathbb{R}^{n \times n}$ jest wygenerowaną macierzą współczynników, a $b = Ax : x = (1, ..., 1)^T$ jest wektorem prawych stron.

Zadany układ równań należy rozwiązać metodą eliminacji Gaussa oraz metodą inwersji dla macierzy A generowanej jako:

- $A = \mathbf{H}_n$ - macierz Hilberta n-tego stopnia
- $A = \mathbf{R}_n$ - losowa macierz kwadratowa stopnia n (dla ustalonych wartości $cond(A)$)

Rozwiązanie

Program z rozwiązaniem: `ex3.jl`

Wyniki i obserwacje

Na podstawie poniższych tabel możemy zauważyć, że obie metody dają wyniki o zbliżonej dokładności zarówno dla macierzy Hilberta, jak i macierzy losowych.
Dodatkowo, patrząc na wyniki dla macierzy losowych, możemy zauważyć, że rozmiar macierzy ma niewielkie znaczenie dla błędów badanych metod - dla macierzy o różnych rozmiarach, ale takich samych wskaźnikach uwarunkowania uzyskujemy wyniki o bardzo zbliżonych błędach.
Porównując także wyniki uzyskane dla losowych macierzy z tymi dla macierzy Hilberta, lecz dla podobnych wartości $cond(A)$ - przykładowo porównując wyniki dla macierzy $\mathbf{H_6}$ oraz $A = \mathbf{R_5}$: $cond(A) = 10^7$ - również zaobserwujemy zbliżone rzędy błedów.
Możemy zatem stwierdzić, że głównym czynnikiem determinującym rząd wielkości błędu rozwiązania jest wskaźnik uwarunkowania, co znaczy, że algorytmy działają poprawnie, natomiast zadania mogą być źle uwarunkowane, jak w przypadku macierzy Hilberta.

n	$cond(A)$	$rank(A)$	δ_{gauss}	$\delta_{inversion}$
1	1.0	1	0.0	0.0
2	19.281470067903967	2	$5.661048867003676e-16$	$1.1240151438116956e-15$
3	524.0567775860627	3	$8.351061872731819e-15$	$9.825526038180824e-15$
4	15513.738738929662	4	$4.2267316576255873e-13$	$3.9600008750140806e-13$
5	476607.2502419338	5	$1.256825919192874e-12$	$8.128168770215688e-12$
6	$1.4951058641781438e7$	6	$1.5435074657413347e-10$	$1.0423794065751672e-10$
7	$4.753673568815896e8$	7	$6.520804933066021e-9$	$4.3299229851434615e-9$
8	$1.5257575568347378e10$	8	$3.6010489197068436e-7$	$4.0236799996435915e-7$
9	$4.9315332284138226e11$	9	$1.3216991540025553e-5$	$1.4626798972086921e-5$
10	$1.602493053861801e13$	10	0.0004194170177181955	0.00040714905218460087
11	$5.2247797042670644e14$	10	0.01004906783345069	0.010645959401385671
12	$1.642582584316724e16$	11	0.5502106922296848	0.6697890564301745
13	$4.4691072287967375e18$	11	70.1556197115221	82.66675811171989
14	$3.213787720967528e17$	11	9.649642437452474	10.094732062453225
15	$3.3659212797388026e17$	12	692.4295360390742	715.740988667373
16	$2.2179227009259395e18$	12	10.414656083840297	8.442143351389534
17	$9.830888596045286e17$	12	18.67581817300634	17.157982115668773
18	$2.58632622754956e18$	12	5.40548300394664	3.742412802776696
19	$3.42284739358336e18$	13	15.073941146224387	16.84769281513296
20	$6.806966466008104e18$	13	28.79267493699834	30.751202239608727

Table 5: Błędy względne wyników obliczonych metodą Gaussa i inversji dla macierzy Hilberta

n	$cond(A)$	$rank(A)$	δ_{gauss}	$\delta_{inversion}$
5	10^0	5	$1.719950113979703e-16$	$1.7901808365247238e-16$
5	10^1	5	$1.2161883888976234e-16$	$1.719950113979703e-16$
5	10^3	5	$4.606673749484534e-14$	$5.32315529767373e-14$
5	10^7	5	$4.6258744328352526e-10$	$4.428329574522588e-10$
5	10^{12}	5	$1.4311845989993493e-5$	$1.3892648149843546e-5$
5	10^{16}	4	0.4701807484683632	0.5764835175254149
10	10^0	10	$3.2368285245694683e-16$	$1.7554167342883504e-16$
10	10^1	10	$5.404859424301949e-16$	$4.657646926676368e-16$
10	10^3	10	$1.0871629027882736e-14$	$1.3286081194048358e-14$
10	10^7	10	$5.3124054213478024e-11$	$1.348469495294427e-10$
10	10^{12}	10	$2.452555006986748e-5$	$2.0626679921248615e-5$
10	10^{16}	9	0.19643775452930137	0.18250673926399288
20	10^0	20	$6.030054506389723e-16$	$4.742874840267547e-16$
20	10^1	20	$5.461575137144e-16$	$2.7081250392525437e-16$
20	10^3	20	$1.1811750363909761e-14$	$1.1291792337224575e-14$
20	10^7	20	$1.0828552075050355e-12$	$8.5128898422199e-11$
20	10^{12}	20	$1.7043123271689405e-5$	$1.764361921628692e-5$
20	10^{16}	19	0.24777843015168452	0.199723624960062

Table 6: Błędy względne wyników obliczonych metodą Gaussa i inversji dla macierzy losowych

Zadanie 4 - Pierwiastki wielomianu Wilkinsona

Problem

Obliczyć pierwiastki wielomianu Wilkinsona $p(x) = \Pi_{i=1}^{20}(x - i)$ za pomocąfunkcji `roots` z pakietu `Polynomials` języka `Julia`, znając postać normalną P tego wielomianu oraz porównać uzyskane wyniki z jego rzeczywistymi pierwiastkami.
Następnie należy powtórzyć eksperyment dla zmodyfikowanego współczynnika przy x^{19} ($-210 - 2^{-23}$ zamiast -210).

Rozwiązanie

Program z rozwiązaniem: `ex4.jl`

Wyniki i obserwacje

W *tabeli 7* możemy zauważyć, że obliczone pierwiastki wielomianu są bardzo zbliżone do rzeczywistych wartości, jednak ze względu na to, że w postaci normalnej współczynniki wielomianu są bardzo duże dla niższych potęch x, dostajemy wyniki bardzo oddalone od oczekiwanego - 0.
Dodatkowo w *tabeli 8* możemy zauważyć, że niewielka zmiana wartości jednego ze współczynników powoduje, że część pierwiastków nie jest możliwa do obliczenia w ciele $\mathbb{R}$ i przez to otrzymujemy wyniki zespolone.
Te dwa spostrzeżenia pokazują, że zadanie jest źle uwarunkowane.

k	z_k	$\lvert P(z_k)\rvert$	$\lvert p(z_k)\rvert$	$\lvert z_k - k\rvert$
1	0.9999999999998084	23323.616390897252	24347.616390897056	$1.9162449405030202e-13$
2	2.0000000000114264	64613.550791712885	80997.5507918065	$1.1426415369442111e-11$
3	2.9999999998168487	18851.098984644806	101795.09897958105	$1.8315127192636282e-10$
4	3.999999983818672	$2.6359390809003003e6$	$2.37379508196076e6$	$1.6181327833209025e-8$
5	5.000000688670983	$2.3709842874839526e7$	$2.306984278668964e7$	$6.88670983350903e-7$
6	5.999988371602095	$1.2641076289358065e8$	$1.2508366146559621e8$	$1.162839790502801e-5$
7	7.000112910766231	$5.2301629899144447e8$	$5.2055763533357024e8$	0.00011291076623098917
8	7.999279406281878	$1.798432141726085e9$	$1.7942387274786062e9$	0.0007205937181220534
9	9.003273831140774	$5.121881552672067e9$	$5.115158339932115e9$	0.003273831140774064
10	9.989265687778465	$1.4157542666785017e10$	$1.4147337313301882e10$	0.010734312221535092
11	11.027997558569794	$3.586354765112257e10$	$3.584844758752149e10$	0.027997558569794023
12	11.94827395840048	$8.510931555828575e10$	$8.508835580052173e10$	0.051726041599520656
13	13.082031971969954	$2.2136146301419052e11$	$2.2133136429365445e11$	0.08203197196995404
14	13.906800565193148	$3.812024574451268e11$	$3.811638891032201e11$	0.09319943480685211
15	15.081439299377482	$8.809029239560208e11$	$8.808498064028993e11$	0.0814392993774824
16	15.942404318674466	$1.6747434633806333e12$	$1.6746775852847332e12$	0.05759568132553383
17	17.026861831476396	$3.3067827086376123e12$	$3.3066967643946914e12$	0.026861831476395537
18	17.99048462339055	$6.166202940769282e12$	$6.16609562105193e12$	0.009515376609449788
19	19.001981084996206	$1.406783619602919e13$	$1.4067702719729383e13$	0.001981084996206306
20	19.999803908064397	$3.284992217648231e13$	$3.2849758339688676e13$	0.00019609193560299332

Table 7: Pierwiastki wielomianiu Wilkinsona

k	z_k	$\lvert P'(z_k)\rvert$	$\lvert z_k - k\rvert$
1	$0.9999999999999805 + 0.0i$	2168.9361669986724	$1.9539925233402755e-14$
2	$1.9999999999985736 + 0.0i$	29948.438957395843	$1.4264145420384011e-12$
3	$3.000000000105087 + 0.0i$	239010.53520956426	$1.0508705017286957e-10$
4	$3.9999999950066143 + 0.0i$	939293.8049425513	$4.993385704921138e-9$
5	$5.000000034712704 + 0.0i$	$7.44868039679552e6$	$3.4712703822492585e-8$
6	$6.000005852511414 + 0.0i$	$1.4689332508961653e7$	$5.852511414161654e-6$
7	$6.999704466216799 + 0.0i$	$5.817946400915084e7$	0.00029553378320112955
8	$8.007226654064777 + 0.0i$	$1.3954205929609105e8$	0.0072266540647767386
9	$8.917396943382494 + 0.0i$	$2.459617755654851e8$	0.082603056617506
10	$10.09529034477879 - 0.6432770896263527i$	$2.291018560461982e9$	0.6502965968281023
11	$10.09529034477879 + 0.6432770896263527i$	$2.291018560461982e9$	1.110092326920887
12	$11.793588728372308 - 1.6522535463872843i$	$2.077690789102519e10$	1.6650968123818863
13	$11.793588728372308 + 1.6522535463872843i$	$2.077690789102519e10$	2.0458176697496047
14	$13.99233053734825 - 2.5188196443048287i$	$9.390730597798799e10$	2.5188313205122075
15	$13.99233053734825 + 2.5188196443048287i$	$9.390730597798799e10$	2.7129043747424584
16	$16.73073008036981 - 2.8126272986972136i$	$9.592356563898315e11$	2.906000476898456
17	$16.73073008036981 + 2.8126272986972136i$	$9.592356563898315e11$	2.8254873227453055
18	$19.50243895868367 - 1.9403320231930836i$	$5.050467401799687e12$	2.4540193937292005
19	$19.50243895868367 + 1.9403320231930836i$	$5.050467401799687e12$	2.004328632592893
20	$20.84690887410499 + 0.0i$	$4.858653129933677e12$	0.8469088741049902

Table 8: Pierwiastki zaburzonego wielomianu wilkinsona

Zadanie 5 - Model wzrostu populacji

Problem

Wykonać 40 iteracji wzoru opisującego model wzrostu populacji:

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

Dla zadanych wartości: $p_0 = 0.01 \land r = 3$ w pojedynczej oraz podwójnej precyzji, a także w pojedynczej precyzji, obcinając wynik p_10 do 3 miejsc po przecinku.

Rozwiązanie

Program z rozwiązaniem: `ex5.jl`

Wyniki i obserwacje

Na podstawie wyników z *tabeli 9* możemy stwierdzić, że błedy obliczeniowe bardzo łatwo propagują się na dalsze obliczenia.
Porównując wyniki dla pojedynczej i podwójnej precyzji, możemy zauważyć, że od pewnego momentu niedokładność obliczeń wynika ze zbyt małej precyzji - nie jesteśmy w stanie dokładnie reprezentować kolejnych wartości, co wpływa na narastanie błędów w dalszych obliczeniach.
Podobną rzecz możemy zauważyć, porównując wyniki dla precyzji `Float32` z oraz bez obcięcia wyniku w 10-tej iteracji. Tutaj błąd obliczeniowy staje się bardziej znaczący nawet szybciej niż w przypadku różnicy precyzji, którą wcześniej obserwowaliśmy, a ostateczny wynik jest zdecydowanie mniej dokładny.
Możemy zatem stwierdzić, że model ten numerycznie niestabilny, ponieważ niewielkie błedy bardzo łatwo propagują się na następne iteracje, skutkując niedokładnymi wynikami.

n	p_n - `Float32`	p_n - `Float32` (obcięcie p_{10})	p_n - `Float64`
0	0.01	0.01	0.01
1	0.0397	0.0397	0.0397
2	0.15407173	0.15407173	0.15407173000000002
3	0.5450726	0.5450726	0.5450726260444213
4	1.2889781	1.2889781	1.2889780011888006
5	0.1715188	0.1715188	0.17151914210917552
6	0.5978191	0.5978191	0.5978201201070994
7	1.3191134	1.3191134	1.3191137924137974
8	0.056273222	0.056273222	0.056271577646256565
9	0.21559286	0.21559286	0.21558683923263022
10	0.7229306	**0.722**	0.722914301179573
11	1.3238364	1.3241479	1.3238419441684408
12	0.037716985	0.036488414	0.03769529725473175
13	0.14660022	0.14195944	0.14651838271355924
14	0.521926	0.50738037	0.521670621435246
15	1.2704837	1.2572169	1.2702617739350768
16	0.2395482	0.28708452	0.24035217277824272
17	0.7860428	0.9010855	0.7881011902353041
18	1.2905813	1.1684768	1.2890943027903075
19	0.16552472	0.577893	0.17108484670194324
20	0.5799036	1.3096911	0.5965293124946907
21	1.3107498	0.09289217	1.3185755879825978
22	0.088804245	0.34568182	0.058377608259430724
23	0.3315584	1.0242395	0.22328659759944824
24	0.9964407	0.94975823	0.7435756763951792
25	1.0070806	1.0929108	1.315588346001072
26	0.9856885	0.7882812	0.07003529560277899
27	1.0280086	1.2889631	0.26542635452061003
28	0.9416294	0.17157483	0.8503519690601384
29	1.1065198	0.59798557	1.2321124623871897
30	0.7529209	1.3191822	0.37414648963928676
31	1.3110139	0.05600393	1.0766291714289444
32	0.0877831	0.21460639	0.8291255674004515
33	0.3280148	0.7202578	1.2541546500504441
34	0.9892781	1.3247173	0.29790694147232066
35	1.021099	0.034241438	0.9253821285571046
36	0.95646656	0.13344833	1.1325322626697856
37	1.0813814	0.48036796	0.6822410727153098
38	0.81736827	1.2292118	1.3326056469620293
39	1.2652004	0.3839622	0.0029091569028512065
40	0.25860548	1.093568	0.011611238029748606

Table 9: Wartości wyników wzoru rekurencyjnego dla poszczególnych iteracji

Zadanie 6 - Ciąg rekurencyjny

Problem

Wykonać 40 iteracji wzoru rekurencyjnego:

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

Dla zadanych par wartości: c oraz x_0 w podwójnej precyzji.

Rozwiązanie

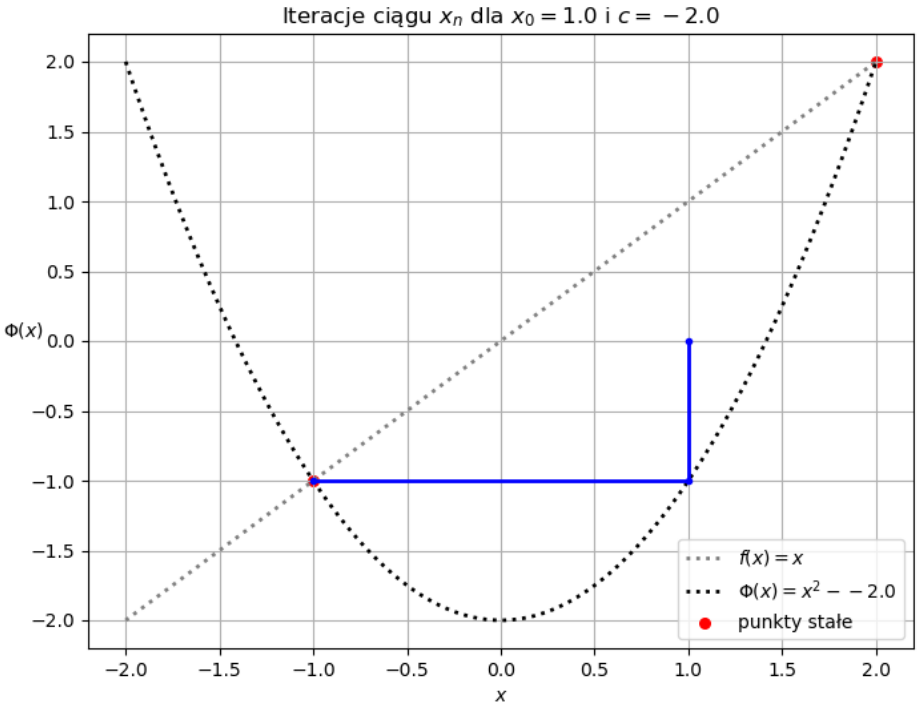
Program z rozwiązaniem: `ex6.jl`

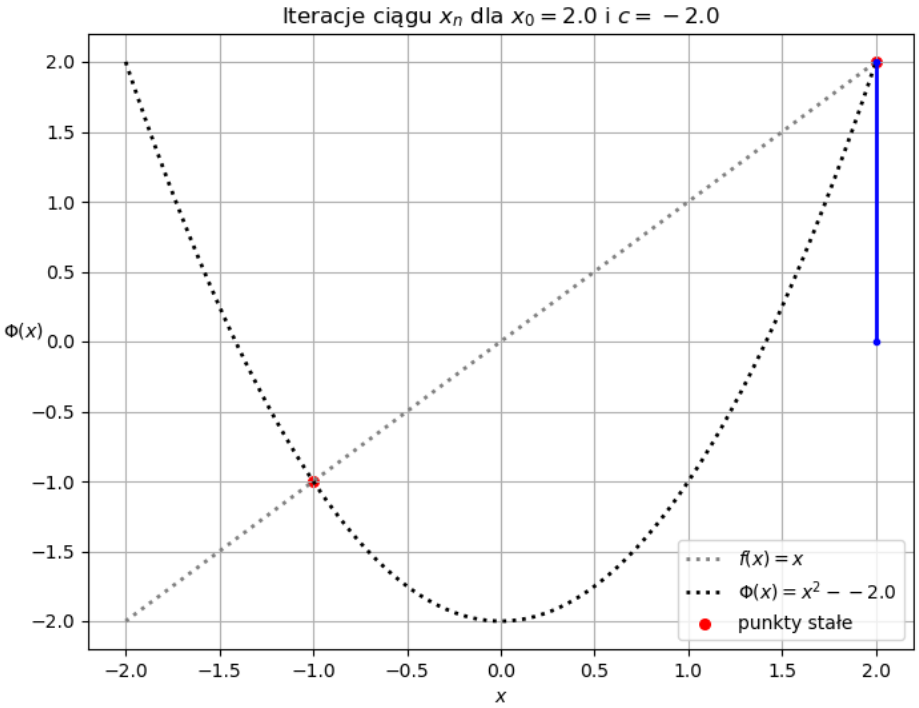
Wyniki i obserwacje

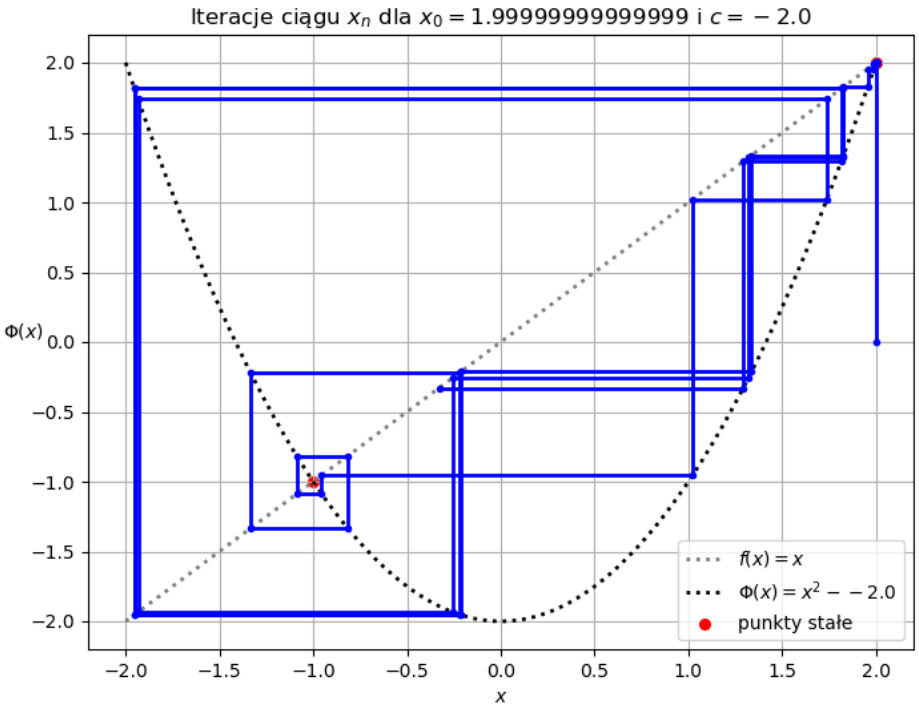
Zauważmy, że pukty stałe badanej funkcji możemy wyliczyć, rozwiązując równanie $\alpha = \alpha^2 + c$.

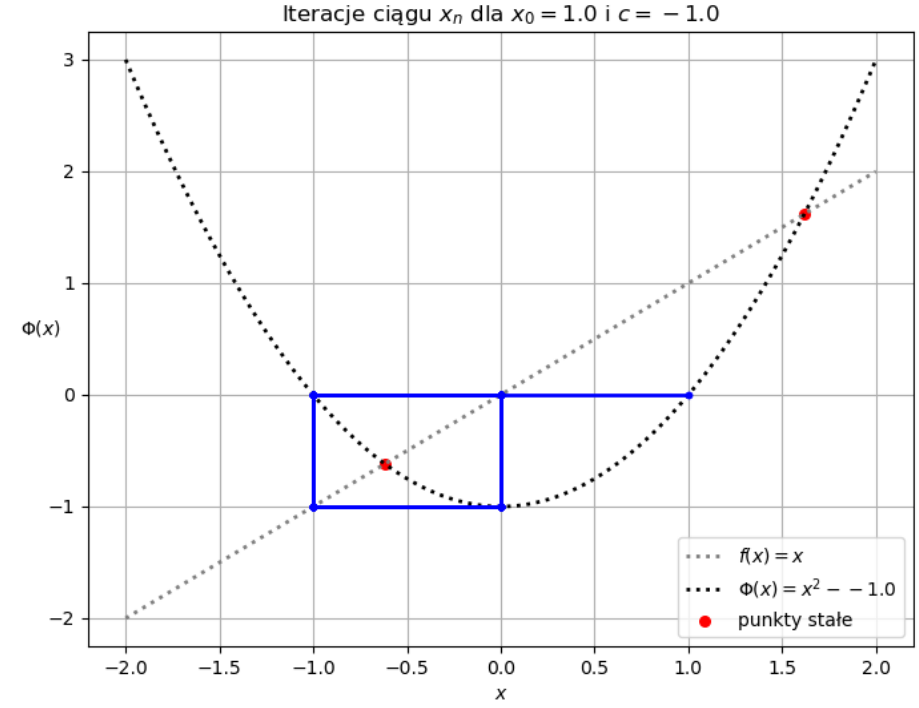
Zatem dla $c = -2$ otrzymujemy punkty stałe $\alpha \in \{-1, 2\}$. Możemy więc na podstawie poniższych wykresów stwierdzić, że badana metoda jest zbieżna dla $x_0 \in \{1, 2\}$, natomiast dla $x_0 = 1.99999999999999$ metoda jest rozbieżna.

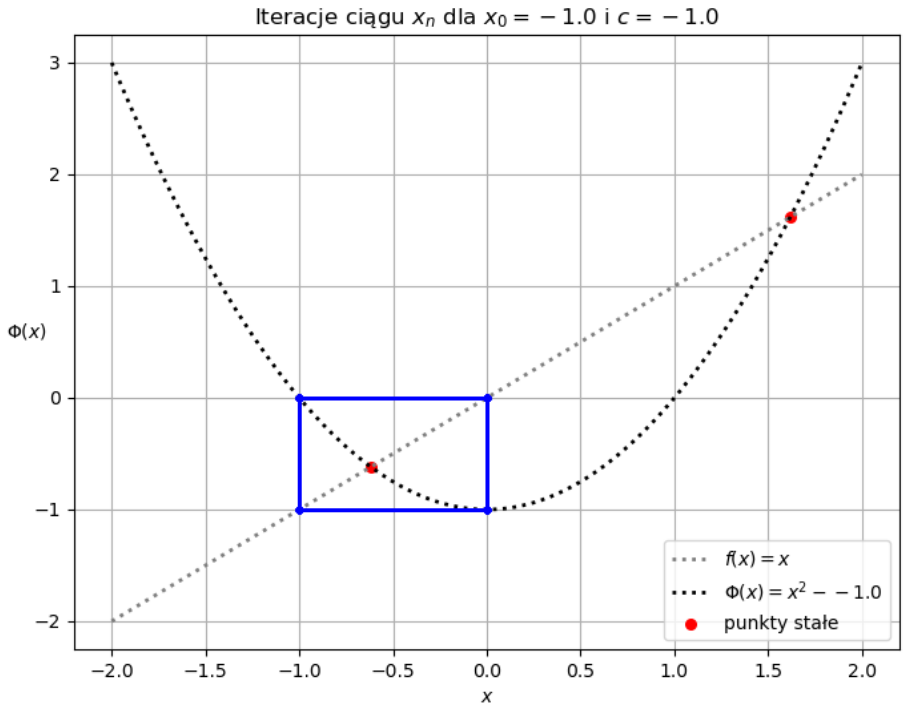
Tak samo możemy wyznaczyć punkty stałe metody dla $c = -1$. Tutaj otrzymujemy wartości $\alpha = \frac{1 \pm \sqrt{5}}{2}$ - są to wartości niewymierne, więc nawet przy doborze odpowiednich danych ($x_0 = \alpha$) nie otrzymalibyśmy w praktyce ciągów zbieżnych przez wzgląd na błąd reprezentacji wartości α.
Dla badanych całkowitych wartości x_0 ($x_0 \in \{1, -1\}$) otrzymujemy takie same ciągi (z dokładnością do pierwszego wyrazu, którego znak redukuje się po podnoszeniu do kwadratu), które zwracają cyklicznie wartości $0 = (\pm 1)^2 - 1$ oraz $-1 = 0^2 - 1$.
Natomiast dla $x_0 \in \{0.25, 0.75\}$ w pewnym momencie również otrzymujemy ciągi cykliczne $\{0, -1\}$, coSSS widać w *tabeli 11*, jednak tutaj wynika to z błędów obliczeniowych przy podnoszeniu coraz mniejszych wartości do kwadratu (bardzo szybkie zmiejszanie wartości cechy) oraz odejmowania $c = 1$.

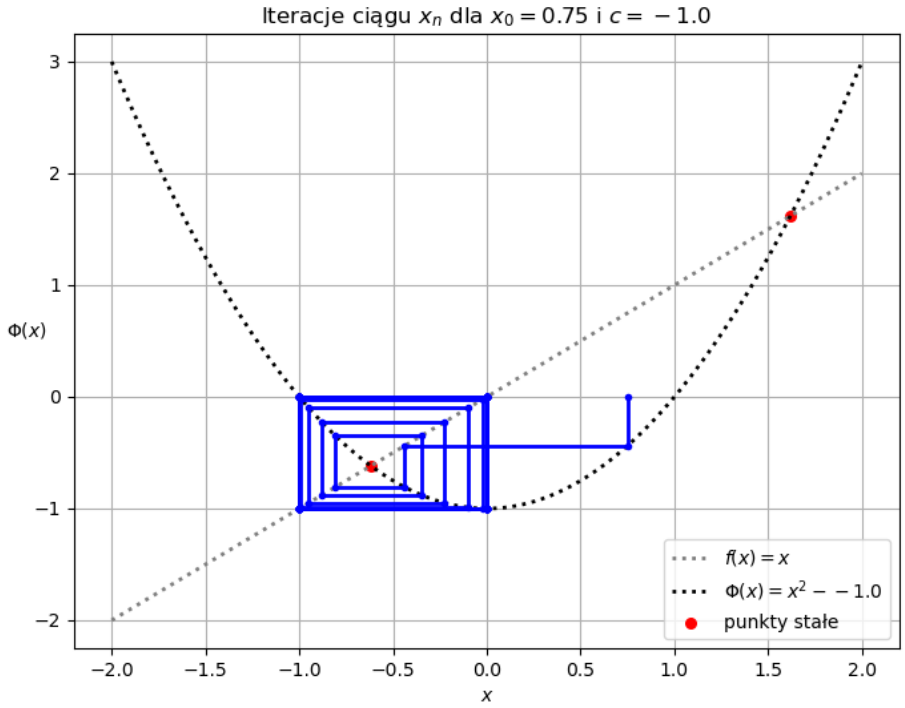

Iteracje ciągu x_n dla $x_0 = 1.0$ i $c = -2.0$
$\Phi(x)$
x
$f(x) = x$
$\Phi(x) = x^2 - -2.0$
punkty stałe

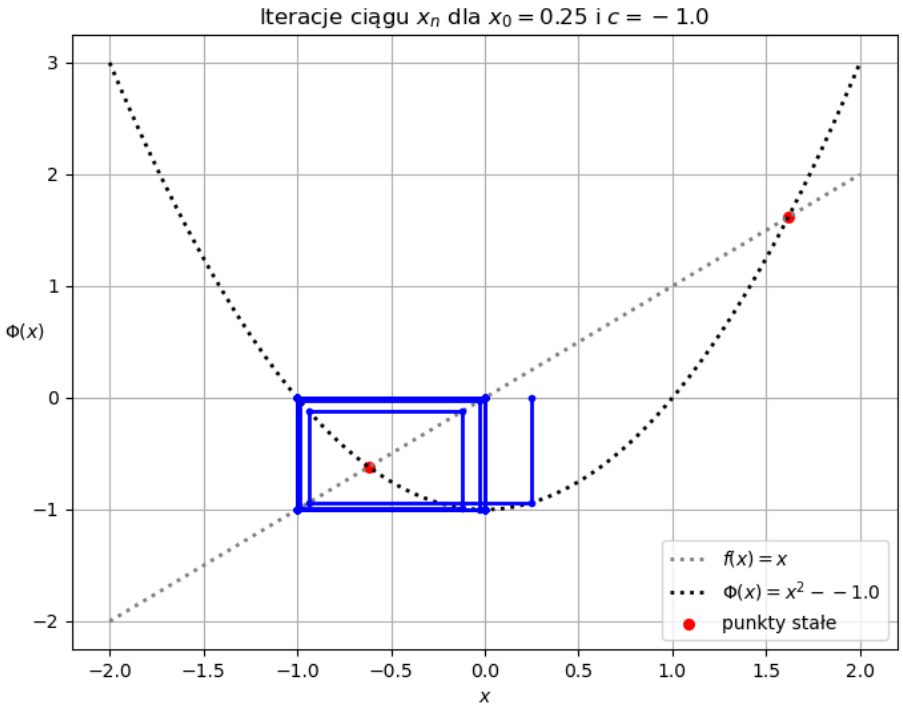

Iteracje ciągu x_n dla $x_0 = 2.0$ i $c = -2.0$
$\Phi(x)$
x
$f(x) = x$
$\Phi(x) = x^2 - -2.0$
punkty stałe


Iteracje ciągu x_n dla $x_0 = 1.99999999999999$ i $c = -2.0$
$\Phi(x)$
x
$f(x) = x$
$\Phi(x) = x^2 - -2.0$
punkty stałe


Iteracje ciągu x_n dla $x_0 = 1.0$ i $c = -1.0$
$\Phi(x)$
x
$f(x) = x$
$\Phi(x) = x^2 - -1.0$
punkty stałe


Iteracje ciągu x_n dla $x_0 = -1.0$ i $c = -1.0$
$\Phi(x)$
x
$f(x) = x$
$\Phi(x) = x^2 - -1.0$
punkty stałe


Iteracje ciągu x_n dla $x_0 = 0.75$ i $c = -1.0$
$\Phi(x)$
x
$f(x) = x$
$\Phi(x) = x^2 - -1.0$
punkty stałe


Iteracje ciągu x_n dla $x_0 = 0.25$ i $c = -1.0$
$\Phi(x)$
x
$f(x) = x$
$\Phi(x) = x^2 - -1.0$
punkty stałe

n	$x_0 = 1.0$	$x_0 = 2.0$	$x_0 = 1.99999999999999$
0	1.0	2.0	1.99999999999999
1	−1.0	2.0	1.99999999999996
2	−1.0	2.0	1.9999999999998401
3	−1.0	2.0	1.9999999999993605
4	−1.0	2.0	1.999999999997442
5	−1.0	2.0	1.9999999999897682
6	−1.0	2.0	1.9999999999590727
7	−1.0	2.0	1.999999999836291
8	−1.0	2.0	1.9999999993451638
9	−1.0	2.0	1.9999999973806553
10	−1.0	2.0	1.999999989522621
11	−1.0	2.0	1.9999999580904841
12	−1.0	2.0	1.9999998323619383
13	−1.0	2.0	1.9999993294477814
14	−1.0	2.0	1.9999973177915749
15	−1.0	2.0	1.9999892711734937
16	−1.0	2.0	1.9999570848090826
17	−1.0	2.0	1.999828341078044
18	−1.0	2.0	1.9993133937789613
19	−1.0	2.0	1.9972540465439481
20	−1.0	2.0	1.9890237264361752
21	−1.0	2.0	1.9562153843260486
22	−1.0	2.0	1.82677862987391
23	−1.0	2.0	1.3371201625639997
24	−1.0	2.0	−0.21210967086482313
25	−1.0	2.0	−1.9550094875256163
26	−1.0	2.0	1.822062096315173
27	−1.0	2.0	1.319910282828443
28	−1.0	2.0	−0.2578368452837396
29	−1.0	2.0	−1.9335201612141288
30	−1.0	2.0	1.7385002138215109
31	−1.0	2.0	1.0223829934574389
32	−1.0	2.0	−0.9547330146890065
33	−1.0	2.0	−1.0884848706628412
34	−1.0	2.0	−0.8152006863380978
35	−1.0	2.0	−1.3354478409938944
36	−1.0	2.0	−0.21657906398474625
37	−1.0	2.0	−1.953093509043491
38	−1.0	2.0	1.8145742550678174
39	−1.0	2.0	1.2926797271549244
40	−1.0	2.0	−0.3289791230026702

Table 10: Wartości wywołań funkcji rekurencyjnej w poszczególnych iteracjach dla $c = -2$

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

Table 11: Wartości wywołań funkcji rekurencyjnej w poszczególnych iteracjach dla $c = -1$