Poniedziałek 7:30, TN

Obliczenia naukowe: sprawozdanie z laboratorium

LISTA NR 2

JAKUB IWON

1. Zadanie 1

1.1. Opis:

Celem zadania było sprawdzenie jaki wpływ na wynik obliczania iloczynu skalarnego dwóch zadanych wektorów mają niewielkie zmiany danych.

1.2. Rozwiązanie i wnioski:

Wyniki dla oryginalnych oraz zaburzonych danych:

	Float64 (dane oryginalne)	**Float64 (dane zaburzone)**	**Float32 (dane oryginalne)**	**Float32 (dane zaburzone)**
Algorytm A	$1.0251881368296672 * 10^{-10}$	-0.004296342739891585	-0.4999443	-0.4999443
Algorytm B	$-1.5643308870494366 * 10^{-10}$	-0.004296342998713953	-0.4543457	-0.4543457
Algorytm C	0.0	-0.004296342842280865	-0.5	-0.5
Algorytm D	0.0	-0.004296342842280865	-0.5	-0.5

Jak widać w powyższej tabeli niewielkie zmiany danych znaczącą wpłynęły na wynik obliczeń dla arytmetyki Float64.

Definiując wskaźnik uwarunkowania zadania obliczenia iloczynu skalarnego dwóch wektorów jako:

$$cond(a,b) = \frac{\sum_{i=1}^{n} |a_i b_i|}{|\sum_{i=1}^{n} a_i b_i|}$$

oszacować można jaki wpływ na wynik ma zaburzenie danych.

Wartość wskaźnika dla danych z zadania to 548297681477783689. Tak duża wartość świadczy o złym uwarunkowaniu zadania przez co mały błąd danych znacząco zniekształca wynik.

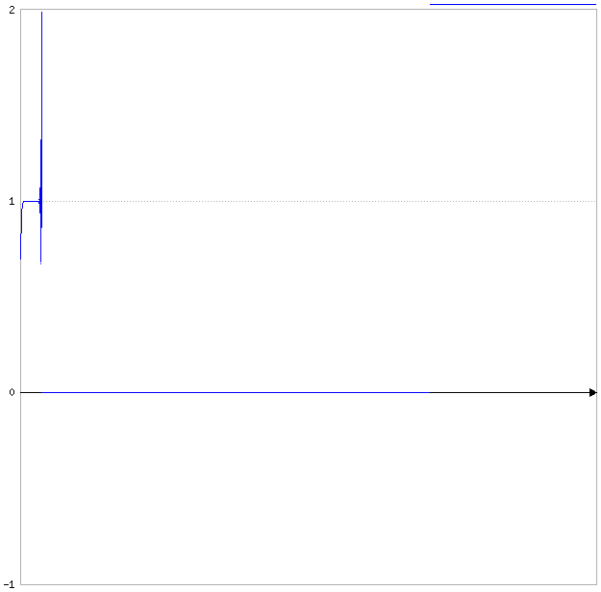
W arytmetyce Float32 zaburzenie danych nie wpłynęło na zmianę wyniku. Powodem tego jest ograniczona precyzja Float32 w parze z bardzo małym zaburzeniem danych.

2. Zadanie 2

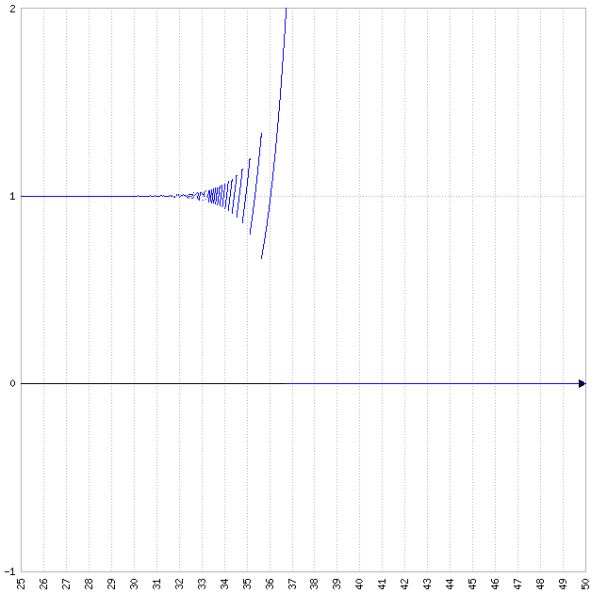
2.1. Opis

Celem zadania było narysowanie funkcji $f(x) = e^x \ln(1 + e^x)$ oraz policzenie granicy $\lim_{x \to \infty} f(x)$ a następnie porównanie obliczonego wyniku z danymi zawartymi na wykresie.

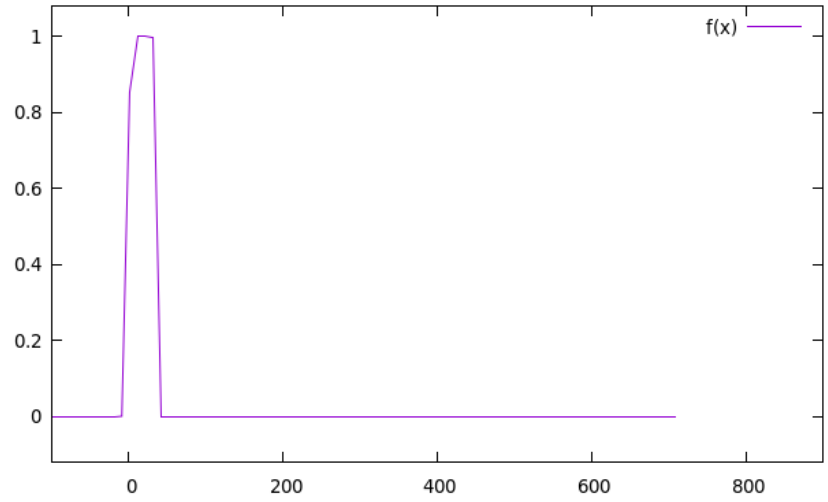
2.2. Rozwiązanie i wnioski



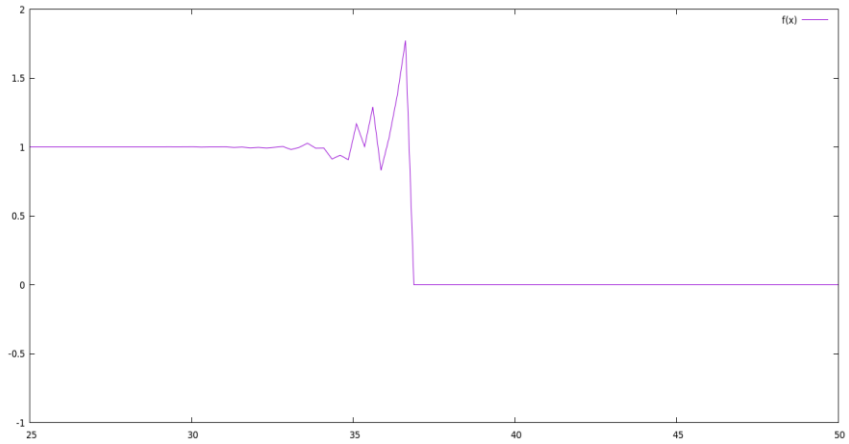
Rys. 1: *f(x) narysowana za pomocą programu MAFA function plotter w przedziale [0, 1000]*



Rys. 2: *f(x) narysowana za pomocą programu MAFA function plotter w przedziale [25, 50]*



Rys. 3: *f(x) narysowana za pomocą programu gnuplot w przedziale [0, 1000]*



Rys. 4: *f(x) narysowana za pomocą programu gnuplot w przedziale [25, 50]*

Patrząc na powyższe wykresy można zauważyć nietypowe zachowanie funkcji w przedziale [30, 40].

Wartość funkcji dla dalszych argumentów wynosi 0 aż do argumentu bliskiemu 700 gdzie w przypadku pierwszego programu do wizualizacji, wartość funkcji wynosi nieskończoność natomiast w przypadku drugiego programu nie jest zdefiniowana.

Dane odczytane z powyższych wykresów stoją w sprzeczności z faktyczną granicą $\lim_{x \to \infty} f(x) = 1$.

W wytłumaczeniu tych zjawisk pomocna jest poniższa tabela.

Wartości obliczone dla wybranych argumentów w języku `Julia` w arytmetyce `Float64:`

x	$f(x)$	e^x	e^{-x}
36	0.9572857056195249	4.311231547115195e15	2.3195228302435696e-16
37	0.0	1.1719142372802612e16	8.533047625744066e-17
709	0.0	8.218407461554972e307	1.216780750623423e-308
710	NaN	Inf	4.47628622567513e-309

Patrząc na powyższą tabelę widzimy, że dla $x = 37$ wartość e^{-x} jest mniejsza od epsilonu maszynowego dla typu Float64 co sprawia, że $\ln(1 + e^{-x})$ w komputerze równa się $\ln(1) = 0$, przez co $f(x) = 0$. Wartość funkcji równa się zeru aż do $x = 710$ kiedy to e^x przekracza największa reprezentowalną liczbę w Float64 prowadząc do wartości $f(x) = 0 * \infty$.

Ograniczenia precyzji komputera tłumaczą niepoprawną reprezentację $f(x)$ zwracaną przez programy wizualizujące.

3. Zadanie 3

3.1. Opis

Celem zadania było sprawdzenie wpływu stopnia uwarunkowania macierzy A na rozwiązanie układu równań liniowych

$$Ax = b$$

gdy macierz A była generowana losowo oraz gdy A była macierzą Hilberta.

3.2. Rozwiązanie i wnioski

Fragment wyniku działania programu dla Macierzy Hilberta

Stopień macierzy	**Wskaźnik uwarunkowania**	**Błąd względny (odwrotność)**	**Błąd względny (eliminacja Gaussa)**
2	19.28147006790397	1.4043333874306803e-13%	5.6610488670036755e-14%
5	476607.25024259434	3.3544360584359634e-10%	1.6828426299227196e-10%
8	1.5257575538060041e10	3.07748390309622e-5%	6.124089555723088e-6%
9	4.931537564468762e11	0.000454126830317664 3%	0.000387516341850324 76%
10	1.6024416992541715e13	0.02501493411824885 8%	0.00867039023709691%
20	1.3553657908688225e18	2206.269725787049%	754.9915039472976%
63	7.897351241650672e19	36877.04616967009%	22205.839122356207%
64	1.3690530852035506e19	17464.924366871874%	17192.266834918115%
65	2.221598860924685e19	6472.944447418329%	5338.614374157294%
83	2.7339614131689046e19	22047.23482315335%	12329.915681200062%
84	3.0284391467048763e19	11374.79706797343%	12313.2831003026%
85	7.496800669751853e19	5836.952373000002%	5162.567496918943%
86	9.021010807368111e19	25239.88133836656%	26004.712788998422%
100	2.209807011860762e19	34573.293917736955%	29215.146010180717%

Patrząc na powyższą tabelę można zauważyć związek między wskaźnikiem uwarunkowania macierzy a błędem względnym. Im większy wskaźnik uwarunkowania tym większy błąd wyniku. Wskaźnik uwarunkowania rośnie wraz ze wzrostem stopnia macierzy.

Wskaźnik uwarunkowania dla macierzy Hilberta bardzo gwałtownie rośnie wraz ze wzrostem stopnia macierzy. Przekłada się to na duży wzrost błędu.

Stopień macierzy	Wskaźnik uwarunkowania	Błąd względny (odwrotność)	Błąd względny (eliminacja Gaussa)
5	1	1.9860273225978185e-14%	1.1102230246251565e-14%
	10	4.3284461991572717e-14%	4.094300213216722e-14%
	1000	3.366653157991702e-12%	3.5149597838873174e-12%
	10000000	2.8634383055029537e-8%	2.086844394258356e-8%
	1000000000000	0.0014374880738385598%	0.00200518850516447 7%
	10000000000000000	24.576490926289697%	25.096970739600682%
10	1	1.4895204919483637e-14%	3.475547814546182e-14%
	10	2.6272671962866383e-14%	3.217732024427419e-14%
	1000	3.8510498592143697e-13%	4.800992561380036e-14%
	10000000	1.1786153034374924e-8%	1.2556213111222846e-8%
	1000000000000	0.002258316085093425%	0.002892892322673306%
	10000000000000000	3.639317093391286%	2.373661457957 7493%
20	1	4.2130001622920403e-14%	7.147801729532275e-14%
	10	9.053594779595602e-14%	8.53500874144267e-14%
	1000	1.0080953447600035e-12%	1.1069819938008627e-12%
	10000000	9.087668147401415e-9%	1.0606727916922677e-8%
	1000000000000	0.0011060155396216585%	0.001030265730251 3865%
	10000000000000000	13.886018244462553%	7.755538066398941%

W przypadku macierzy losowej również możemy zauważyć podobną zależność między wskaźnikiem uwarunkowania a błędem względnym.

Widać również, że wielkość błędu zależna jest wyłącznie od wskaźnika uwarunkowania macierzy (nie jest zależna od stopnia macierzy).

4. Zadanie 4

4.1. Opis

Celem zadania było obliczenie pierwiastków wielomianu Wilkinsona oraz sprawdzenie poprawności obliczonych wartości.

4.2. Rozwiązanie i wnioski

Wyniki działania programu:

k	z_k	$\lvert P(z_k)\rvert$	$\lvert p(z_k)\rvert$	$\lvert z_k - k\rvert$
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

Pierwiastki wielomianu (z_k) zostały policzone za pomocą funkcji `roots.` Wyniki te nie są dokładne co skutkuje błędnymi wartościami wielomianów $P(z_k)$ oraz $p(z_k)$ (stworzonymi za pomocą funkcji `Poly` oraz `poly`). Do obliczenia wartości wielomianu dla danego argumentu posłużyła funkcja `polyval.`

Niedokładność wyników można tłumaczyć niedokładną reprezentacją w komputerze współczynników, z których wielomian był skonstruowany. Złe uwarunkowanie zadania policzenia pierwiastków dla tego wielomianu powoduje, że niewielkie zaburzenie współczynników powoduje duże zaburzenie wyników.

Obliczone pierwiastki wielomianu Wilkinsona po dokonaniu zmiany współczynnika -210 na -210-2^{-23}:

k	z_k	$\|P(z_k)\|$	$\|p(z_k)\|$	$\|z_k - k\|$
1	0.9999999999998357 + 0.0im	20992.0	3.012096e6	1.6431300764452317e-13
2	2.0000000000550373 + 0.0im	349184.0	7.37869763029606e19	5.503730804434781e-11
3	2.99999999660342 + 0.0im	2.221568e6	3.3204139201100146e20	3.3965799062229962e-9
4	4.000000089724362 + 0.0im	1.046784e7	8.854437817429645e20	8.972436216225788e-8
5	4.99999857388791 + 0.0im	3.9463936e7	1.8446726974084148e21	1.4261120897529622e-6
6	6.000020476673031 + 0.0im	1.29148416e8	3.320450195282314e21	2.0476673030955794e-5
7	6.99960207042242 + 0.0im	3.88123136e8	5.422366528916045e21	0.00039792957757978087
8	8.007772029099446 + 0.0im	1.072547328e9	8.289399860984229e21	0.007772029099445632
9	8.915816367932559 + 0.0im	3.065575424e9	1.1607472501770085e22	0.0841836320674414
10	10.095455630535774 - 0.6449328236240688im	7.143113638035824e9	1.7212892853671066e22	0.6519586830380406
11	10.095455630535774 + 0.6449328236240688im	7.143113638035824e9	1.7212892853671066e22	1.1109180272716561
12	11.793890586174369 - 1.6524771364075785im	3.357756113171857e10	2.8568401004080516e22	1.665281290598479
13	11.793890586174369 + 1.6524771364075785im	3.357756113171857e10	2.8568401004080516e22	2.045820276678428
14	13.992406684487216 - 2.5188244257108443im	1.0612064533081976e11	4.934647147685479e22	2.5188358711909045
15	13.992406684487216 + 2.5188244257108443im	1.0612064533081976e11	4.934647147685479e22	2.7128805312847097
16	16.73074487979267 - 2.812624896721978im	3.315103475981763e11	8.484694713574187e22	2.9060018735375106
17	16.73074487979267 + 2.812624896721978im	3.315103475981763e11	8.484694713574187e22	2.825483521349608
18	19.5024423688181 - 1.940331978642903im	9.539424609817828e12	1.318194782057474e23	2.454021446312976
19	19.5024423688181 + 1.940331978642903im	9.539424609817828e12	1.318194782057474e23	2.004329444309949
20	20.84691021519479 + 0.0im	1.114453504512e13	1.591108408283123e23	0.8469102151947894

Bardzo niewielka zmiana jednego ze współczynników spowodowała znaczące zmiany w wartościach obliczonych pierwiastków. Niektóre z obliczonych pierwiastków stały się liczbami zespolonymi. Ta dysproporcja to dowód na złe uwarunkowanie zadania.

5. Zadanie 5

5.1. Opis

Celem zadania było sprawdzenie wpływu jaki ma wprowadzenie błędu do danych w obliczeniach wartości wyrażenia rekurencyjnego: $p_{n+1} = p_n + rp_n(1 - p_n)$. W drugiej części zadania należało porównać dokładność wyniku obliczania tego wyrażenia dla arytmetyki Float64 oraz Float32.

5.2. Rozwiązanie i wnioski

Błąd do danych wprowadzamy po 10-tej iteracji, obcinając wynik do trzech cyfr po przecinku.

Wyniki dla wybranych iteracji:

Iteracja	Wynik bez obcięcia	Wynik z obcięciem
1	0.0397	0.0397
2	0.15407173	0.15407173
3	0.5450726	0.5450726
10	0.7229306	0.722
11	1.3238364	1.3241479
12	0.037716985	0.036488414
15	1.2704837	1.2572169
20	0.5799036	1.3096911
25	1.0070806	1.0929108
30	0.7529209	1.3191822
35	1.021099	0.034241438
40	0.25860548	1.093568

W skutek obcięcia wyniku po 10-tej iteracji w jednym z przebiegów widać jak stopniowo mała rozbieżność wzrasta. Przy 40-stej iteracji wyniki obu przebiegów są całkowicie różne.

Błąd rośnie w bardzo szybkim tempie co spowodowane jest tym, że iterowana funkcja jest kwadratowa.

Wybrane iteracje dla obliczeń bez obcięcia w arytmetyce Float32 oraz Float64:

Iteracja	Float32	Float64
1	0.0397	0.0397
2	0.15407173	0.15407173000000002
3	0.5450726	0.5450726260444213
10	0.7229306	0.21558683923263022
11	1.3238364	1.3238419441684408
12	0.037716985	0.03769529725473175
15	1.2704837	1.2702617739350768
20	0.5799036	0.5965293124946907
25	1.0070806	1.315588346001072
30	0.7529209	0.37414648963928676
35	1.021099	0.9253821285571046
40	0.25860548	0.011611238029748606

Już od drugiej iteracji można zauważyć rozbieżność w wynikach. Jako, że arytmetyka Float64 jest bardziej precyzyjna można przypuszczać, że wyniki obliczone w tym standardzie są dokładniejsze, jednakże i w tym przypadku wynik końcowy również obarczony jest błędem.

Błędy popełnione w arytmetyce Float32 są popełniane również w Float64 na dalszym etapie obliczeń. Zwiększona precyzja pozwala na początkowo mniejszy błąd w stosunku do Float32 jednak nie gwarantuje wyeliminowania niedokładności.

6. Zadanie 6

6.1. Opis

Celem zadania było zbadanie zachowania ciągów generowanych przez iterację wyrażenia $x_{n+1} = x_n^2 + c$ dla poszczególnych parametrów x_0 oraz c.

6.2. Rozwiązanie i wnioski

Wybrane iteracje dla poszczególnych parametrów:

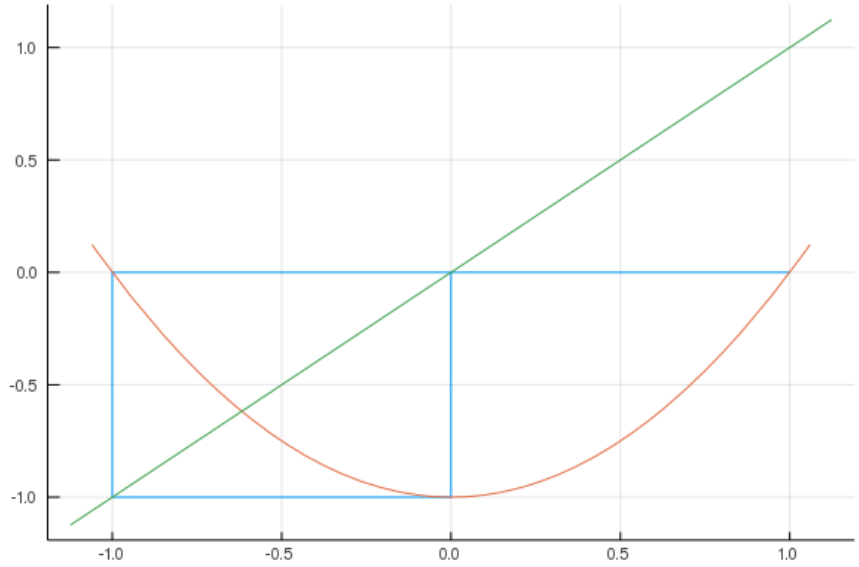
c	-2			-1			
Iteracja \ x_0	1	2	1.99999999999999	1	-1	0.75	0.25
1	-1.0	2.0	1.999999999999	0.0	0.0	-0.4375	-0.9375
2	-1.0	2.0	1.999999999999	-1.0	-1.0	-0.80859375	-0.12109375
3	-1.0	2.0	1.999999999999	0.0	0.0	-0.34617614	-0.98533630
4	-1.0	2.0	1.999999999997	-1.0	-1.0	-0.88016207	-0.02911236
5	-1.0	2.0	1.999999999989	0.0	0.0	-0.22531472	-0.02911236
10	-1.0	2.0	1.999999989522	-1.0	-1.0	-0.99962018	-6.5931482e-11
11	-1.0	2.0	1.999999958090	0.0	0.0	-0.00075947	-1.0
20	-1.0	2.0	1.989023726436	-1.0	-1.0	-1.0	0.0
21	-1.0	2.0	1.956215384326	0.0	0.0	0.0	-1.0
22	-1.0	2.0	1.826778629873	-1.0	-1.0	-1.0	0.0
30	-1.0	2.0	1.738500213821	-1.0	-1.0	-1.0	0.0
31	-1.0	2.0	1.022382993457	0.0	0.0	0.0	-1.0
32	-1.0	2.0	-0.954733014689	-1.0	-1.0	-1.0	0.0
33	-1.0	2.0	-1.088484870662	0.0	0.0	0.0	-1.0
34	-1.0	2.0	-0.815200686338	-1.0	-1.0	-1.0	0.0
40	-1.0	2.0	-0.328979123002	-1.0	-1.0	-1.0	0.0

Niektóre z ciągów zachowują się stabilnie zbiegając do jednej wartości bądź tworząc cykl.

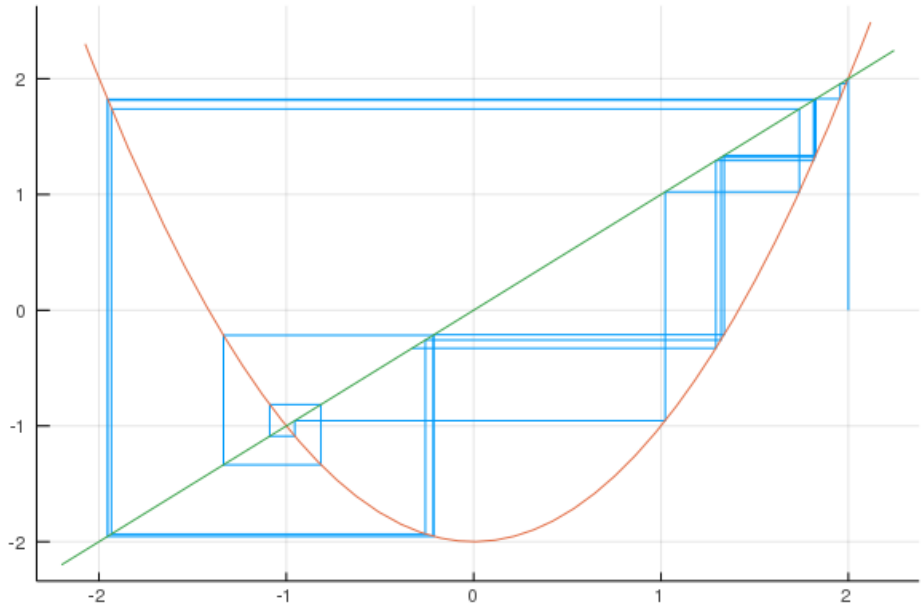
Przykładem ciągu niestabilnego jest ciąg trzeci.

Można zauważyć, że drobna zmiana parametru x_0 z wartości 2 na 1.99999999999999 całkowicie zmienia przebieg ciągu czyniąc go niestabilnym.

Zachowanie ciągów można obserwować za pomocą iteracji graficznej.



Rys. 4: Przykład ciągu stablinego, tworzącego cykl 0.0, -1.0 ($x_0 = 1, c = -1$)



Rys. 5: Przykład ciągu niestablinego ($x_0 = 1.999, c = -2$)