Obliczenia Naukowe- Lista 2

Szymon Skoczylas

8 listopada 2020

Zadanie 1

Cel zadania

Porównać wyniki zadania 5 z listy pierwszej z wynikami tych samych algorytmów, ale ze zmodyfikowanymi danymi wejściowymi. Z wektora x z x_4 usunięta została ostatnia 9, a z x_5 ostatnia 7.

Wyniki i wykonanie

Do wykonania tego zadania został użyty program stworzony na potrzeby listy pierwszej. Wyniki zwracane przez program bez modyfikacji wektora x:

```
[szymo@szymoHost lista_1]$ julia zad5.jl
Float64:
1.0251881368296672e-10
-1.5643308870494366e-10
0.0
0.0

Float62
-0.4999443
-0.4543457
-0.5
-0.5
```

Wyniki zwracane po modyfikacji wektora x:

```
[szymo@szymoHost lista_2]$ julia zad1.jl
Float64:
-0.004296342739891585
-0.004296342998713953
-0.004296342842280865
-0.004296342842280865

Float32
-0.4999443
-0.4543457
-0.5
-0.5
```

Wnioski

Po usunięciu wyspecyfikowanych w treści zadania wartości, nie widać zmian w wartościach zwracanych dla typu $Float32$. Natomiast, dla typu $Float64$ wyniki zauważalnie odbiegają od tych z listy pierwszej. Brak zmian dla arytmetyki $Float32$ jest wynikiem jej znacznie niższej precyzji niż arytmetyki $Float64$. Wysoka precyzja $Float64$ pozwala na dużo dokładniejsze przechowywanie wartości wejściowej. Tak duża wrażliwośc algorytmu na, wydawałoby się, małe zmiany pokazuje, że problem przedstawiony w poleceniu został źle uwarunkowany.

Zadanie 2

Cel zadania

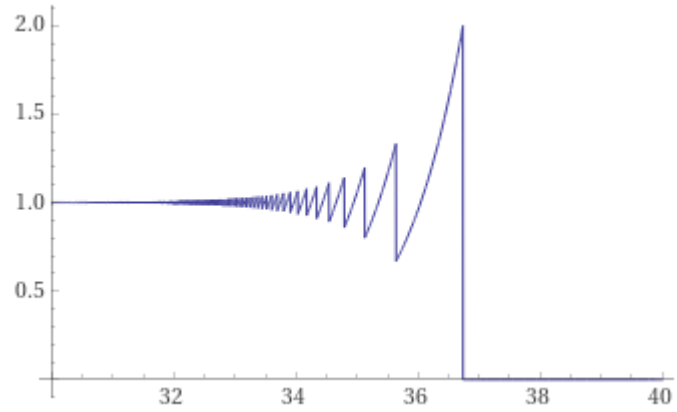
Stworzyć wykresy funkcji $f(x) = e^x ln(1 + e^{-x})$, do tego celu użyte zostały: $WolframAlpha$ oraz pakiet $Plots$ dostępny w języku $Julia$. Następnie porównano wykresy z rzeczywistą wartoscią granicy.

Wyniki i wykonianie

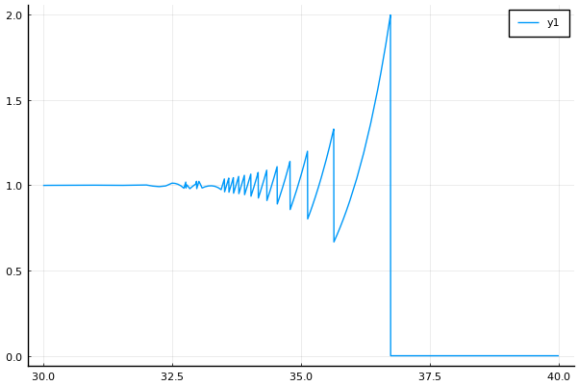
Za pomocą funkcji $limit()$ dostępnej w języku $Julia$, otrzymano następujący wynik:

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

Wykres wygenerowany przez $WolframAlpha$:



Wykres otrzymany za pomocą pakietu $Plots$:



Wnioski

Realna granica funkcji przy $x \to \infty$ wynosi 1. Z wykresów stworzoych przez programy, widzimy, że funkcja załamuje się w pewnym momencie, przez co błędnie można określić wartośc granicy jako 0. Anomalia ta jest następstwem tego, że dla większych x wartość e^{-x} będzie bardzo bliska 0 i w wyniku tego zostanie pochłonięta przez 1. Więc od pewnego momentu będziemy liczyć cały czas $ln(1)$. Wyrażenie to jest równe 0, przez co, na wykresie występuje fałszywa zbieżność do 0.

Zadanie 3

Cel zadania

Celem tego zadania było rozwiącanie równanie w postaci $Ax = b$ dla macierzy Hilberta stopnia n, oraz losowej macierzy o stopniu n z określonym wskaźnikiem uwarunkowania ($c = 1, 10, 10^3, 10^7, 10^{12}, 10^{16}$). Następnie policzyć błędy względne, oraz porównać otrzymane wyniki z dokładnymi.

Wyniki i wykonanie

Następujący kod posłużył do wykonania zadania:

```
b = A*x
gauss_elimination, marix_inversion = A\b, inv(A)*b
elimination_err, inversion_err = norm(gauss_elimination - x) / norm(x), norm(marix_inversion - x) / norm(x)
```

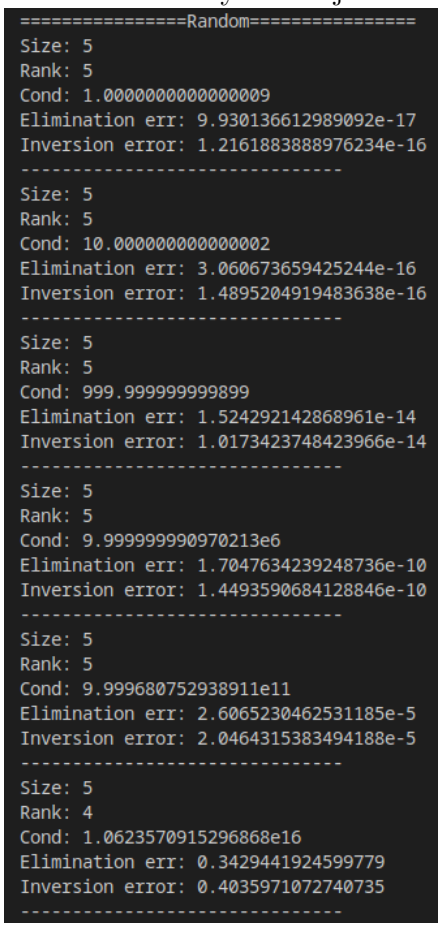
Wyniki dla macierzy Hiliberta:

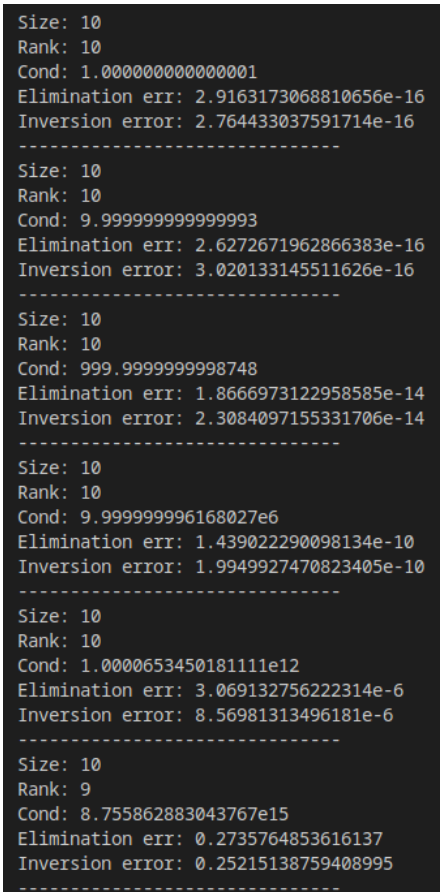
```
===============Hilbert===============
Size: 1
Rank: 1
Cond: 1.0
Elimination err: 0.0
Inversion err: 0.0
------------------------------
Size: 2
Rank: 2
Cond: 19.28147006790397
Elimination err: 4.227603326225575e-16
Inversion err: 1.4043333874306803e-15
------------------------------
Size: 3
Rank: 3
Cond: 524.0567775860644
Elimination err: 6.312995352117186e-16
Inversion err: 0.0
------------------------------
Size: 4
Rank: 4
Cond: 15513.73873892924
Elimination err: 2.1819484005185015e-15
Inversion err: 4.547473508864641e-13
------------------------------
Size: 5
Rank: 5
Cond: 476607.25024259434
Elimination err: 8.99737540851766e-12
Inversion err: 1.4911297558868156e-11
------------------------------
Size: 6
Rank: 6
Cond: 1.4951058642254665e7
Elimination err: 1.8866554820446764e-10
Inversion err: 3.5689314550268525e-10
------------------------------
```

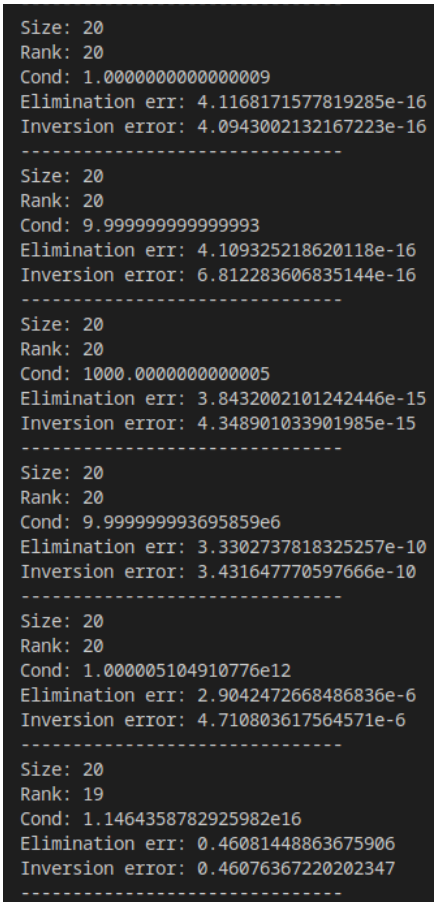
```
Size: 7
Rank: 7
Cond: 4.753673565831296e8
Elimination err: 1.8614175017153493e-8
Inversion err: 1.231617281152939e-8
------------------------------
Size: 8
Rank: 8
Cond: 1.5257575538060041e10
Elimination err: 1.9217530132542664e-7
Inversion err: 1.3503856270597747e-7
------------------------------
Size: 9
Rank: 9
Cond: 4.931537564468762e11
Elimination err: 5.09208671414057e-6
Inversion err: 9.542631496737485e-6
------------------------------
Size: 10
Rank: 10
Cond: 1.6024416992541715e13
Elimination err: 5.508778842434553e-5
Inversion err: 0.0002289249459044258
------------------------------
Size: 11
Rank: 10
Cond: 5.222677939280335e14
Elimination err: 0.005207147400142043
Inversion err: 0.008849600746399502
------------------------------
Size: 12
Rank: 11
Cond: 1.7514731907091464e16
Elimination err: 0.1407902677571603
Inversion err: 0.4039460424541017
------------------------------
```

```
Size: 13
Rank: 11
Cond: 3.344143497338461e18
Elimination err: 355.6363996139929
Inversion err: 212.78257870436312
------------------------------
Size: 14
Rank: 11
Cond: 6.200786263161444e17
Elimination err: 3.5720053119125916
Inversion err: 1.7744875020766255
------------------------------
Size: 15
Rank: 12
Cond: 3.674392953467974e17
Elimination err: 9.583022643862336
Inversion err: 5.293638550842115
------------------------------
Size: 16
Rank: 12
Cond: 7.865467778431645e17
Elimination err: 11.968066192267113
Inversion err: 10.846229140698135
------------------------------
Size: 17
Rank: 12
Cond: 1.263684342666052e18
Elimination err: 5.455797564762254
Inversion err: 8.982730231514317
------------------------------
Size: 18
Rank: 12
Cond: 2.2446309929189128e18
Elimination err: 26.309523680963647
Inversion err: 20.038300611698062
------------------------------
```

Wyniki dla macierzy losowej:







Wyniki

Mając do dyspozycji powyższe dane, łatwo zauważyć, że, na ogół, wraz ze wzrostem rozmiaru macierzy rośnie błąd względny obu metod, zarówno eliminacji Gaussa jak i metody inwersji. Po wartościach w dla macierzy Hilberta można wywnioskować złe uwarunkowanie zadania. Wynik dla macierzy losowych są analogiczne, przy czym błędy obliczeń są mniejsze, jak i również ich tempo wzrostu nie jest tak duże jak w przypadku macierzy Hilberta.

Zadanie 4

Cel zadania

Celem zadania było obliczenie miejsc zerowych wielomianu Wilkinsona (zarówno w postaci kanonicznej jak i iloczynowej):

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

Następnie powtórzyć eksperyment, ale zmienić współczynnik -210 na $-210 - 1^{-23}$

Wyniki i wykonanie

Do wykonania zadania posłużył następujący algorytm (użyty został pakiet *Polynomials*):

```
function find_roots(coff)
    canonical_figure, product_figure = Poly(reverse(coff)), poly(Float64[1.0:20.0;])
    coff_roots = roots(canonical_figure)

    for k in 1:20
        println("k=", k)
        println("zk=", coff_roots[k])
        println("|P(zk)| =", abs(polyval(canonical_figure, coff_roots[k])))
        println("|P(zk)| =", abs(polyval(product_figure, coff_roots[k])))
        println("|zk-k|=", abs(coff_roots[k] - k))
        println("--------------------------------")
    end
end
```

Wyniki dla wielomianu wyjściowego:

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

Wynik dla wielomianu ze zmodyfikowanym współczynnikiem:

```
==================MODIFIED COFFICIENT
k=1
zk=0.9999999999998357 + 0.0im
|P(zk)| =20992.0
|P(zk)| =22016.0
|zk-k|=1.6431300764452317e-13
--------------------------------
k=2
zk=2.0000000000550373 + 0.0im
|P(zk)| =349184.0
|P(zk)| =365568.0
|zk-k|=5.503730804434781e-11
--------------------------------
k=3
zk=2.99999999660342 + 0.0im
|P(zk)| =2.221568e6
|P(zk)| =2.295296e6
|zk-k|=3.3965799062229962e-9
--------------------------------
k=4
zk=4.000000089724362 + 0.0im
|P(zk)| =1.046784e7
|P(zk)| =1.0729984e7
|zk-k|=8.972436216225788e-8
--------------------------------
k=5
zk=4.99999857388791 + 0.0im
|P(zk)| =3.9463936e7
|P(zk)| =4.3303936e7
|zk-k|=1.4261120897529622e-6
--------------------------------
k=6
zk=6.000020476673031 + 0.0im
|P(zk)| =1.29148416e8
|P(zk)| =2.06120448e8
|zk-k|=2.0476673030955794e-5
--------------------------------
k=7
zk=6.99960207042242 + 0.0im
|P(zk)| =3.88123136e8
|P(zk)| =1.757670912e9
|zk-k|=0.00039792957757978087
--------------------------------
k=8
zk=8.007772029099446 + 0.0im
|P(zk)| =1.072547328e9
|P(zk)| =1.8525486592e10
|zk-k|=0.007772029099445632
--------------------------------
k=9
zk=8.915816367932559 + 0.0im
|P(zk)| =3.065575424e9
|P(zk)| =1.37174317056e11
|zk-k|=0.0841836320674414
--------------------------------
k=10
zk=10.095455630535774 - 0.64493282362406 88im
|P(zk)| =7.143113638035824e9
|P(zk)| =1.4912633816754019e12
|zk-k|=0.6519586830380407
--------------------------------
k=11
zk=10.095455630535774 + 0.64493282362406 88im
|P(zk)| =7.143113638035824e9
|P(zk)| =1.4912633816754019e12
|zk-k|=1.1109180272716561
--------------------------------
k=12
zk=11.793890586174369 - 1.65247713640757 85im
|P(zk)| =3.357756113171857e10
|P(zk)| =3.2960214141301664e13
|zk-k|=1.665281290598479
--------------------------------
k=13
zk=11.793890586174369 + 1.65247713640757 85im
|P(zk)| =3.357756113171857e10
|P(zk)| =3.2960214141301664e13
|zk-k|=2.0458202766784277
--------------------------------
k=14
zk=13.992406684487216 - 2.51882442571084 43im
|P(zk)| =1.0612064533081976e11
|P(zk)| =9.545941595183662e14
|zk-k|=2.518835871190904
--------------------------------
k=15
zk=13.992406684487216 + 2.51882442571084 43im
|P(zk)| =1.0612064533081976e11
|P(zk)| =9.545941595183662e14
|zk-k|=2.7128805312847097
--------------------------------
k=16
zk=16.73074487979267 - 2.81262489672197 8im
|P(zk)| =3.3151034759817633e11
|P(zk)| =2.7420894016764064e16
|zk-k|=2.9060018735375106
--------------------------------
k=17
zk=16.73074487979267 + 2.81262489672197 8im
|P(zk)| =3.3151034759817633e11
|P(zk)| =2.7420894016764064e16
|zk-k|=2.825483521349608
--------------------------------
k=18
zk=19.5024423688181 - 1.94033197864290 3im
|P(zk)| =9.539424609817828e12
|P(zk)| =4.2525024879934694e17
|zk-k|=2.4540214463129764
--------------------------------
k=19
zk=19.5024423688181 + 1.94033197864290 3im
|P(zk)| =9.539424609817828e12
|P(zk)| =4.2525024879934694e17
|zk-k|=2.0043294443099486
--------------------------------
k=20
zk=20.84691021519479 + 0.0im
|P(zk)| =1.114453504512e13
|P(zk)| =1.3743733197249713e18
|zk-k|=0.8469102151947894
--------------------------------
```

Wnioski

Powyższe zadanie jest zadaniem źle uwarunkowanym. Można tu zobaczyć jak mała zmiana współczynnika powoduje duże błędy w późniejszych obliczeniach. Będy te również są skutkiem niedokładności arytmetyki *Float*64, która powoduje błędy zaokrągleń, co sprawia, że wartości współczynników nie mogą być przechowywane z należytą dokładnością.

Zadanie 5

Cel zadania

Celem tego zadania było policzenie wartości następującego równania:

$$p_{n+1} = p_n + rp_n(1 - p_n) \text{ dla } n = 0, 1...$$

Przeprowadzno następujące eksperymenty:

1. 40 iteracji dla $Float32$
2. 40 iteracji dla $Float64$
3. 40 iteracji, ucinając wszystko, co 10 iterację, poza 3 pierwszymi miejscami po przecinku

Wyniki i wykonanie

W celu wyliczenia wartości danego wyrazu równania, posłużono się następującym algorytmem rekurencyjnym:

```
function series_expression(number, r_const, start_val)
    number == 0 && return start_val
    curr_val = series_expression(number-1, r_const, start_val)
    return curr_val + r_const * curr_val * (1 - curr_val)
end
```

Otrzymano następujące wyniki:

```
==================40 iterations==================
-for Float32: 0.25860548
-for Float64: 0.011611238029748606
=============40 iterations cut every 10=============
-value: 0.71587336
```

Wnioski

Możemy zaobserwować bardzo duże różnice w wynikach dla każdej z 3 prób. Wyniki dla $Float32$ są dużo większe niż dla $Float64$. Wynik dla metody z obcięciem dał wynik ok. czterokrotny wzrost wyniku. Jest to doskonały przykład na to jak błędy spowodowane przez niedokładność reprezentacji kumulują się i zabużają końcowy wynik.

Zadanie 6

Cel zadania

Celem zadania było policzenie wartości następującego równania rekurencyjnego:

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

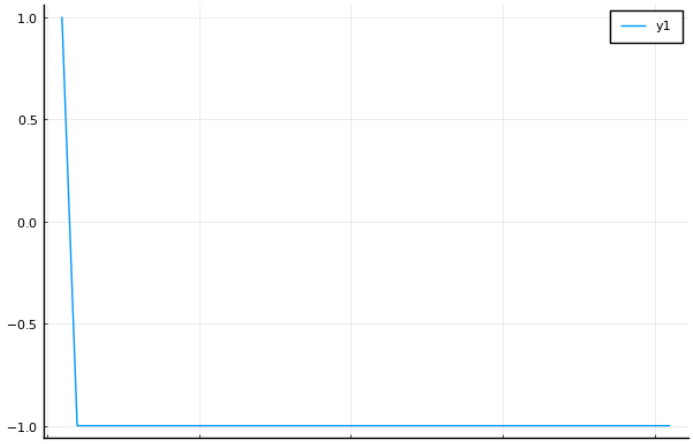
dla następujących danych:

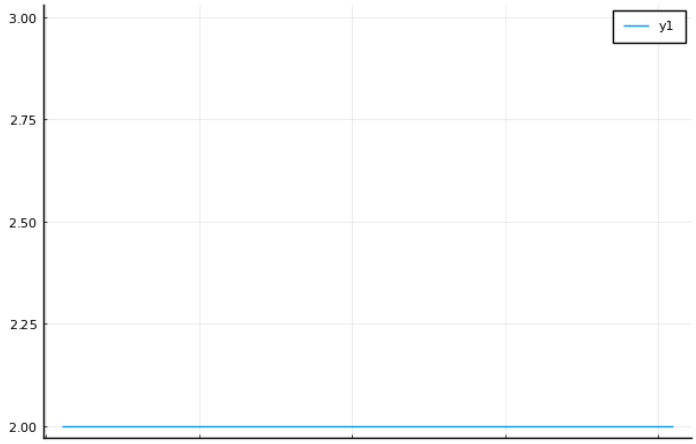
1. $c = -2, x_0 = 1$
2. $c = -2, x_0 = 2$

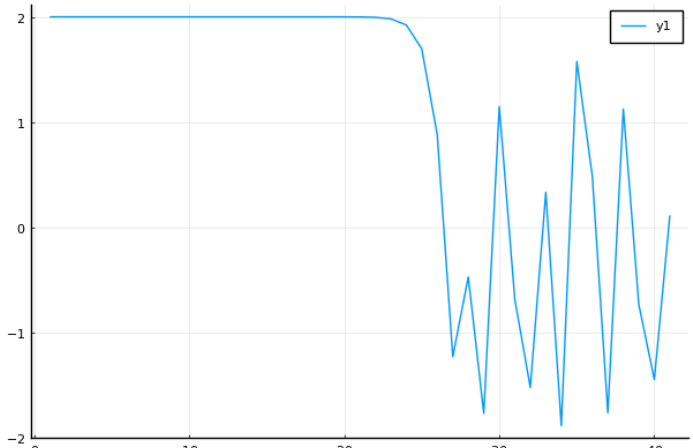
3. $c = -2, x_0 = 1.99999999999999$
4. $c = -1, x_0 = 1$
5. $c = -1, x_0 = -1$
6. $c = -1, x_0 = 0.75$
7. $c = -1, x_0 = 0.25$

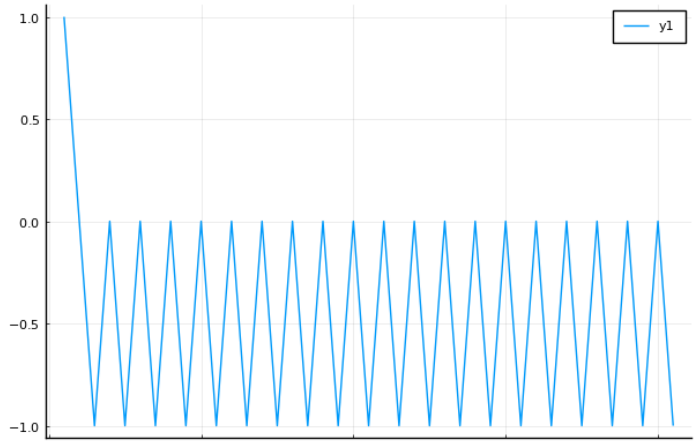
Wyniki i wykonanie

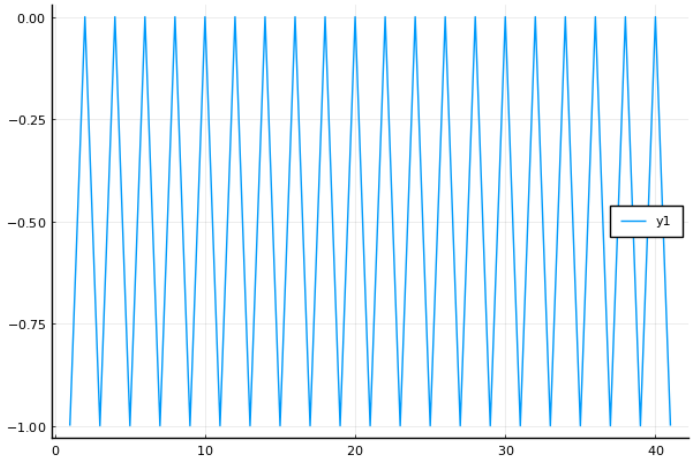
Wyniki, kolejno dla każdego z podpunktów:

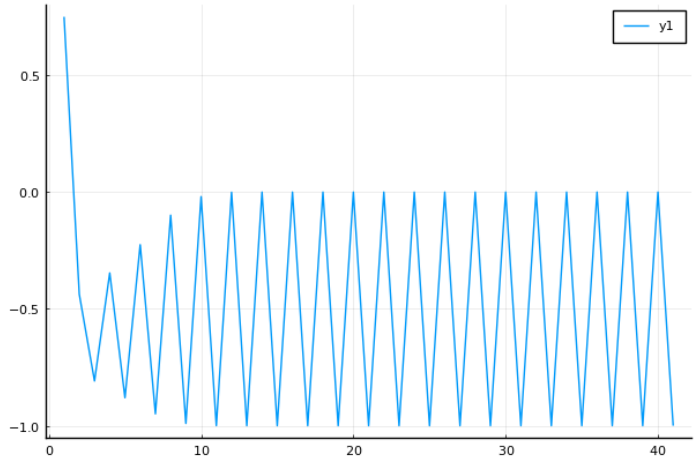


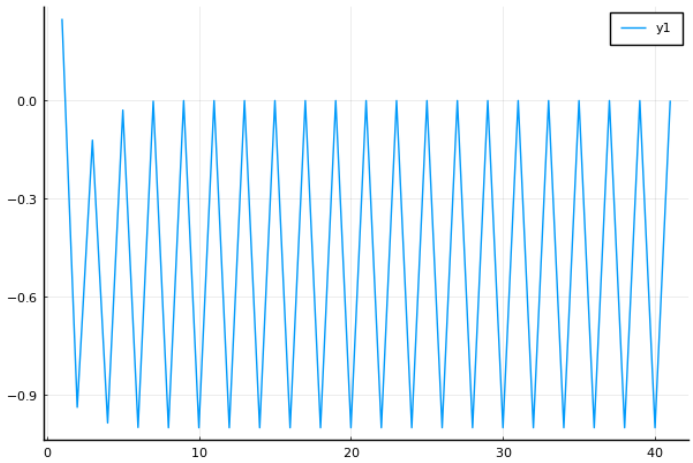












Wnioski

Jak można zauważyć, w 2 pierwszych przypadkach, gdzie $c = -2$ ciągi mają cały czas taką samą wartość, równą x_0. W 3 punkcie widać błędy spowodowane niedokładną reprezentacją liczby 1.99999999999999. Czego skutki potęgują się z każdą kolejną iteracją. Dla pozostałych przypadków ($c = -1$) ciągi praktycznie oscylują pomiędzy dwiema wartościami. Wynika to z tego, że cały czas działamy na liczbach bliskich 0. Podobnie jak w poprzednima zadaniu, doświadczenie te pokazują jak nawarstwiające sie małe błędy, powodują dużą rozbierzność w końcowych wynikach. W tym zadaniu jest to potęgowane z powodu użycia funkcji kwadratowej.