**Sprawozdanie**

Obliczenia naukowe 2019/2020 – lista 2  
Tomasz Janik

# 1. Wstęp

Jest to sprawozdanie, w którym zaprezentowane są rozwiązania zadań z pierwszej listy z kursu Obliczenia naukowe oraz wnioski jakie zostały z nich wyciągnięte. Zadania zostały rozwiązane z pomocą programów napisanych w języku Julia oraz programów pomocniczych.

# 2. Rozwiązania zadań

## 2.1. Zadanie 1

#### 2.1.1. Opis problemu i rozwiązanie

W tym zadaniu należało powtórzyć eksperyment z zadania 5 z listy 1. ze zmodyfikowanymi danymi. W tym celu została użyta funkcja napisana na potrzeby poprzedniej listy, gdzie wprowadzono zmodyfikowane dane.

#### 2.1.2. Wyniki i wnioski

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Metoda | Float32 | Różnica | Float64 | Różnica |
| a („w przód”) | -0.4999443 | 0.0 | -0.004296342739891585 | 0.004296342842410399 |
| b („w tył”) | -0.4543457 | 0.0 | -0.004296342998713953 | 0.004296342842280865 |
| c | -0.5 | 0.0 | -0.004296342842280865 | 0.004296342842280865 |
| d | -0.5 | 0.0 | -0.004296342842280865 | 0.004296342842280865 |

W tabeli poniżej przedstawiono wyniki otrzymane po zmianie danych oraz różnice w porównaniu do oryginalnych wartości.

Jak można zauważyć dla arytmetyki Float32 wyniki się nie zmieniły. Wynika to z faktu ograniczonej precyzji arytmetyki, przez co nasza zmiana nie miała wpływu na obliczenia.

Zmiany możemy zaobserwować dla arytmetyki Float64, gdzie usunięcie ostatniej cyfry po przecinku spowodowało, że wyniki znacznie się od siebie różnią. Widać więc, że nawet niewielka zmiana danych ma wpływ na otrzymany wynik. Z tego wynika, że zadanie to jest **źle uwarunkowane.**

## 2.2. Zadanie 2

### 2.2.1. Opis problemu i rozwiązanie

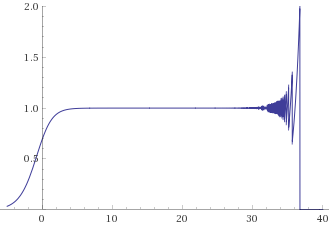
W tym zadaniu należało narysować wykres funkcji w co najmniej dwóch programach do wizualizacji danych. Następnie policzyć granicę tej funkcji w nieskończoności i porównać z otrzymanymi wykresami

### 2.2.2. Wyniki i wnioski

Poniże znajdują się 2 wykresy zadanej funkcji. Pierwszy z nich został wykonany za pomocą programu Matlab, a drugi programu WolframAlpha.

## 

Wykres – Matlab



Wykres - WolframAlpha

Granica wyliczona za pomocą Wolframa wynosi 1.

Łatwo zauważyć, że mniej więcej od x = 30 zaczynamy odbiegać od oczekiwanej wartości. Najpierw pojawiają się coraz większe odchylenia od granicy, a następnie wartość funkcji spada do 0.

Odchylenia te wynikają z faktu, że między różnica a drugim składnikiem mnożenia w pewnym momencie staję się na tyle duża, że pojawiają się błędy w obliczeniach.

Natomiast fakt otrzymywania wartości 0 od pewnego momentu, wynika z tego, że jest bardzo małe w porównaniu z 1, więc w pewnym momencie otrzymamy   
, a co za tym idzie wynik mnożenia również jest równy 0.

## 2.3. Zadanie 3

### 2.3.1. Opis problemu i rozwiązanie

W tym zadaniu należało rozwiązać układ równań liniowych zadany przez macierz współczynników **A** oraz wektor prawych stron **b** na dwa sposoby, metodą eliminacji Gaussa   
 oraz za pomocą macierzy odwrotnej Następnie należało porównać otrzymane wyniki z dokładnymi, czyli policzyć błędy względne. W tym celu zostały napisane funkcje hilbert() oraz random().

Macierz współczynników była generowana jako macierz Hilberta stopnia n lub jako macierz losowa z zadanym wskaźnikiem uwarunkowania za pomocą dołączonych funkcji.

### 2.3.2. Wyniki i Wnioski

W pierwszej znajdują się wyniki uzyskane dla macierzy Hilberta, a w drugiej dla macierzy losowej.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n | Rank(A) | Cond(A) | Błąd metodą Gaussa | Błąd metodą inwersji |
| 1 | 1 | 1.0 | 0.0 | 0.0 |
| 2 | 2 | 19.28147006790397 | 5.661048867003676e-16 | 1.4043333874306803e-15 |
| 3 | 3 | 524.0567775860644 | 8.022593772267726e-15 | 0.0 |
| 4 | 4 | 15513.73873892924 | 4.137409622430382e-14 | 0.0 |
| 5 | 5 | 476607.25024259434 | 1.6828426299227195e-12 | 3.3544360584359632e-12 |
| 6 | 6 | 1.4951058642254665e7 | 2.618913302311624e-10 | 2.0163759404347654e-10 |
| 8 | 8 | 1.5257575538060041e10 | 6.124089555723088e-8 | 3.07748390309622e-7 |
| 11 | 10 | 5.222677939280335e14 | 0.00015827808158590435 | 0.007618304284315809 |
| 12 | 11 | 1.7514731907091464e16 | 0.13396208372085344 | 0.258994120804705 |
| 13 | 11 | 3.344143497338461e18 | 0.11039701117868264 | 5.331275639426837 |
| 16 | 12 | 7.865467778431645e17 | 54.15518954564602 | 29.84884207073541 |
| 20 | 13 | 1.3553657908688225e18 | 7.549915039472976 | 22.062697257870493 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n | c | Rank(A) | Błąd metodą Gaussa | Błąd metodą inwersji |
| 5 | 1.0 | 5 | 1.9860273225978183e-16 | 1.4895204919483638e-16 |
| 5 | 10.0 | 5 | 5.15985034193911e-16 | 7.021666937153401e-16 |
| 5 | 1000.0 | 5 | 2.6682334361182925e-15 | 8.846185720293485e-15 |
| 5 | 1.0e7 | 5 | 3.6311939421285176e-11 | 2.6031257322754125e-11 |
| 5 | 1.0e12 | 5 | 3.200676991392471e-5 | 2.5788030671929275e-5 |
| 5 | 1.0e16 | 4 | 0.04964427479904247 | 0.20515298112994582 |
| 10 | 1.0 | 10 | 3.1006841635969763e-16 | 2.0770370905276122e-16 |
| 10 | 1000.0 | 10 | 1.2379360112333173e-14 | 7.76870578861531e-15 |
| 10 | 1.0e7 | 10 | 7.188227408760592e-11 | 8.482031652700439e-11 |
| 10 | 1.0e12 | 10 | 3.620665362851843e-6 | 6.579661678566108e-6 |
| 10 | 1.0e16 | 9 | 0.07046663204187811 | 0.06578292010944026 |
| 20 | 1.0 | 20 | 3.657001166779273e-16 | 4.3638968010370034e-16 |
| 20 | 10.0 | 20 | 5.09978018830275e-16 | 5.347542221830667e-16 |
| 20 | 1000.0 | 20 | 4.0687909271516324e-14 | 3.659588801429782e-14 |
| 20 | 1.0e7 | 20 | 1.1384355183189299e-10 | 9.503414544309313e-11 |
| 20 | 1.0e12 | 20 | 1.4757604641700831e-5 | 1.7881393432617188e-5 |
| 20 | 1.0e16 | 19 | 0.09748747456054 | 0.10614654890174244 |

Jak można zauważyć dla macierzy Hilberta błąd rośnie wraz ze wzrostem rozmiaru macierzy podobnie dla obu algorytmów. Tak samo rośnie również wskaźnik uwarunkowania. Można zatem stwierdzić, że zadanie to jest źle uwarunkowane dla macierzy Hilberta.

Na przykładzie macierzy losowej możemy zauważyć, że niezależnie od rozmiaru, błędy mają podobną wartość dla takich samych wskaźników uwarunkowania. Rzędy wielkości błędów rosną wprost proporcjonalnie do rzędów wskaźnika uwarunkowania. Widać tutaj dobrze zależność między wskaźnikiem uwarunkowania, a faktycznymi wartościami uzyskanych błędów.

## 2.4. Zadanie 4

### 2.4.1. Opis problemu i rozwiązanie

W tym zadaniu należało znaleźć miejsca zerowe dla zadanego wielomianu (Wilkinsona), przy użyciu biblioteki Polynomials dla języku Julia. Następnie należało porównać otrzymane wyniki z rzeczywistymi. Następnie należało powtórzyć eksperyment Wilkinsona, tj. zmienić współczynnik −210 na . Wyjaśnić zjawisko.

### 2.4.2. Wyniki i wnioski

W tabelach poniżej przedstawiono najpierw wyniki dla podstawowego wielomianu, a następnie dla zmodyfikowanego.

|  |  |  |  |
| --- | --- | --- | --- |
| k |  |  |  |
| 1 | 36352.0 | 38400.0 | 3.0109248427834245e-13 |
| 2 | 181760.0 | 198144.0 | 2.8318236644508943e-11 |
| 3 | 209408.0 | 301568.0 | 4.0790348876384996e-10 |
| 4 | 3.106816e6 | 2.844672e6 | 1.626246826091915e-8 |
| 5 | 2.4114688e7 | 2.3346688e7 | 6.657697912970661e-7 |
| 6 | 1.20152064e8 | 1.1882496e8 | 1.0754175226779239e-5 |
| 7 | 4.80398336e8 | 4.78290944e8 | 0.00010200279300764947 |
| 8 | 1.682691072e9 | 1.67849728e9 | 0.0006441703922384079 |
| 9 | 4.465326592e9 | 4.457859584e9 | 0.002915294362052734 |
| 11 | 3.5759895552e10 | 3.5743469056e10 | 0.025022932909317674 |
| 12 | 7.216771584e10 | 7.2146650624e10 | 0.04671674615314281 |
| 13 | 2.15723629056e11 | 2.15696330752e11 | 0.07431403244734014 |
| 14 | 3.65383250944e11 | 3.653447936e11 | 0.08524440819787316 |
| 15 | 6.13987753472e11 | 6.13938415616e11 | 0.07549379969947623 |
| 16 | 1.555027751936e12 | 1.554961097216e12 | 0.05371328339202819 |
| 17 | 3.777623778304e12 | 3.777532946944e12 | 0.025427146237412046 |
| 18 | 7.199554861056e12 | 7.1994474752e12 | 0.009078647283519814 |
| 19 | 1.0278376162816e13 | 1.0278235656704e13 | 0.0019098182994383706 |
| 20 | 2.7462952745472e13 | 2.7462788907008e13 | 0.00019070876336257925 |

|  |  |  |  |
| --- | --- | --- | --- |
| k |  |  |  |
| 1 | 20992.0 | 22016.0 | 1.6431300764452317e-13 |
| 2 | 349184.0 | 365568.0 | 5.503730804434781e-11 |
| 3 | 2.221568e6 | 2.295296e6 | 3.3965799062229962e-9 |
| 4 | 1.046784e7 | 1.0729984e7 | 8.972436216225788e-8 |
| 5 | 3.9463936e7 | 4.3303936e7 | 1.4261120897529622e-6 |
| 6 | 1.29148416e8 | 2.06120448e8 | 2.0476673030955794e-5 |
| 7 | 3.88123136e8 | 1.757670912e9 | 0.00039792957757978087 |
| 8 | 1.072547328e9 | 1.8525486592e10 | 0.007772029099445632 |
| 9 | 3.065575424e9 | 1.37174317056e11 | 0.0841836320674414 |
| 11 | 7.143113638035824e9 | 1.4912633816754019e12 | 1.1109180272716561 |
| 12 | 3.357756113171857e10 | 3.2960214141301664e13 | 1.665281290598479 |
| 13 | 3.357756113171857e10 | 3.2960214141301664e13 | 2.045820276678428 |
| 14 | 1.0612064533081976e11 | 9.545941595183662e14 | 2.5188358711909045 |
| 15 | 1.0612064533081976e11 | 9.545941595183662e14 | 2.7128805312847097 |
| 16 | 3.315103475981763e11 | 2.7420894016764064e16 | 2.9060018735375106 |
| 17 | 3.315103475981763e11 | 2.7420894016764064e16 | 2.825483521349608 |
| 18 | 9.539424609817828e12 | 4.2525024879934694e17 | 2.454021446312976 |
| 19 | 9.539424609817828e12 | 4.2525024879934694e17 | 2.004329444309949 |
| 20 | 1.114453504512e13 | 1.3743733197249713e18 | 0.8469102151947894 |

Jak można zauważyć w pierwszej tabeli, żadne z obliczonych miejsc zerowych nie jest poprawne. Dodatkowo błąd rośnie wraz z kolejnymi pierwiastkami wielomianu. Wynika to z faktu, że w arytmetyce Float64 jesteśmy wstanie zapisać jedynie 15-17 cyfr znaczących w systemie dziesiętnym. Ze względu na wielkość współczynników nie da się ich zapisać dokładnie, przez co generujemy coraz większe błędy mnożąc przez coraz większe argumenty.

W drugiej tabeli widzimy, że po zmianie jednego ze współczynników o jedynie , sprawiło, że otrzymywane wyniki znacząca się zmieniły. Dodatkowo niektóre obliczone pierwiastki były zespolone. Z tego wynika, że zadanie jest źle uwarunkowane ze względu na zaburzenia współczynników.

## 2.5. Zadanie 5

### 2.5.1. Opis problemu i rozwiązanie

W tym zadaniu należało wykonać zadaną ilość iteracji zadanego równania rekurencyjnego.

Najpierw należało wykonać 40 iteracji równania, a następnie znowu wykonać je tyle samo razy, lecz po 10. iteracji obciąć wynik do 3 miejsc po przecinku i porównać otrzymane wyniki.

Następnie należało wykonać standardowo 40 iteracji wyrażenia w arytmetykach Float32 oraz Float64 i porównać otrzymane wyniki.

W tym celu zostały napisane funkcje p1 – do pierwszego podpunktu oraz p2 – do drugiego.

### 2.5.2. Wyniki i wnioski

W poniższych tabelach przedstawiono porównanie wyników dla obu podpunktów.

|  |  |  |
| --- | --- | --- |
| n | p(n) | P(n) ze zmianą |
| 1 | 0.0397 | 0.0397 |
| … | | |
| **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 |
| 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** |

|  |  |  |
| --- | --- | --- |
| n | Float32 | Float32 |
| 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 |
| … | | |
| 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 |
| 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** |

W obu tabelach widać narastające różnicę w otrzymywanych wynikach. W pierwszej tabeli błąd pojawia się od zastosowanego obcięcia, a w drugiej nawarstwia się od samego początku.

Jest to spowodowane tym, że w równiach rekurencyjnych polegamy na wyniku poprzednich iteracji, co sprawia, że jeśli pojawi się błąd to nawarstwia się on w kolejnych iteracjach. Widać to i w przypadku naszego pozornie nieznacznego obcięcia oraz różnicach w precyzji obliczeń w różnych arytmetykach. W każdej kolejnej iteracji otrzymywane wyniki tracą ze sobą związek.

Badane równanie rekurencyjne jest więc numerycznie niestabilne.

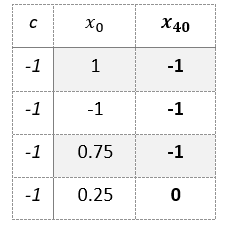
## 2.6. Zadanie 6

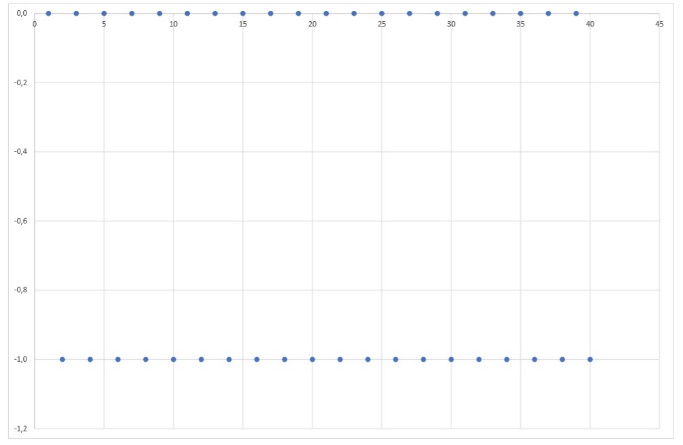
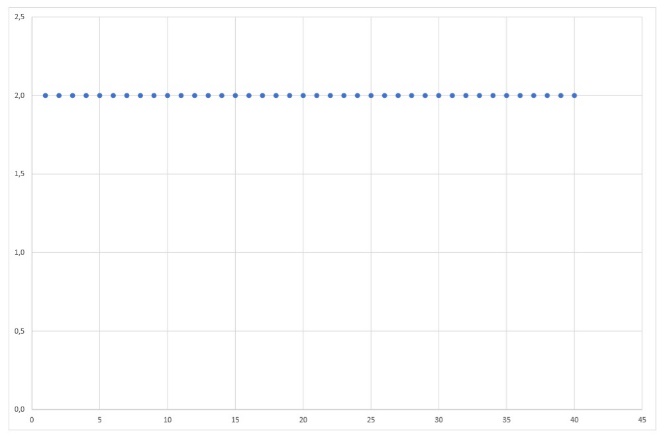
### 2.6.1. Opis problemu i rozwiązanie

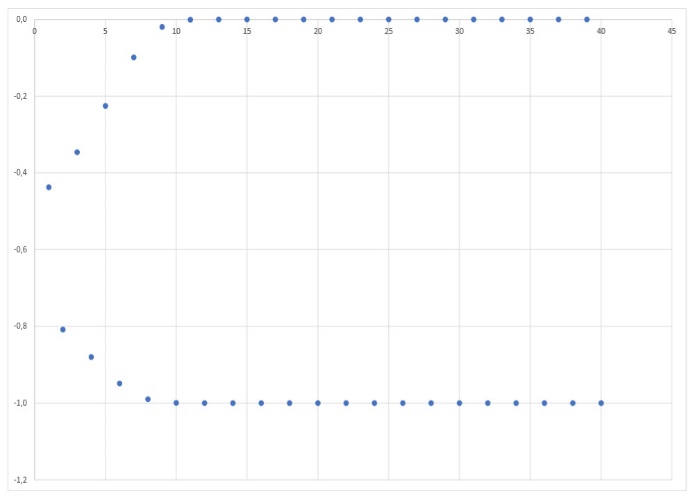
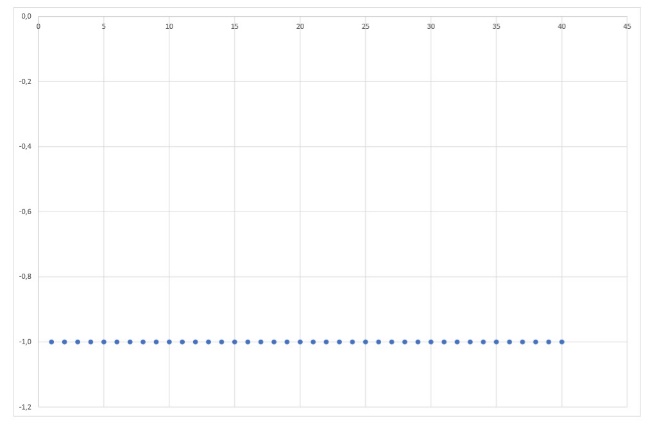
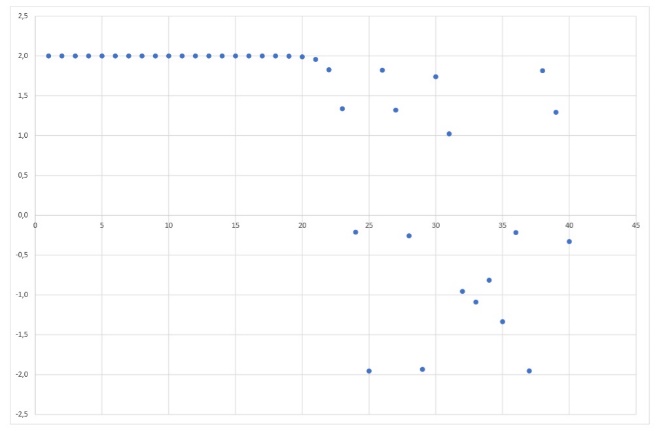
W tym zadaniu należało sprawdzić zachowanie zadanego równanie rekurencyjne dla różnych współczynników oraz c przy 40 iteracjach.

**2.6.2. Wyniki i wnioski**

|  |  |  |
| --- | --- | --- |
| c |  |  |
| -2 | 1.99999999999999 | **-0.3289791230026702** |
| -2 | 1 | **-1** |
| -2 | 2 | **2** |

W tabelach znajdują się wyniki po 40 iteracjach, a poniżej załączone zostały wykresy ciągów.





Jak można zauważyć dla wartości całkowitych nie dzieję się nic specjalnego, jedyną różnicą jest oscylacja w przypadku . Dla wartości ułamkowych zachodzą za to ciekawe zjawiska. Dla oraz widzimy jak kolejne wyrazy ciągów zbiegają do granicy w liczbie całkowitej i od pewnego momentu są stabilne. Dla oraz ciąg na początku jest stabilny, a od pewnego wyrazu zaczyna dawać wartości losowe.

Z tego wynika, że dla współczynnika wzór rekurencyjny jest stabilny, ponieważ mimo ograniczonej precyzji arytmetyki, doszliśmy w pewnym momencie do stabilnych wartości.

Natomiast dla współczynnika zauważamy, że obliczanie kolejnych wyrazów ciągu jest niestabilne, ponieważ błędy się nawarstwiają i od pewnego momentu zaczynamy dostawać wartości losowe.

# 3. Podsumowanie

Rozwiązania zadań miały na celu ukazać jak bardzo złe uwarunkowanie zadania, bądź niestabilność procesu numerycznego może zaburzyć.