Agnieszka Kurzajewska

Nr indeksu: 244994

Sprawozdanie

Zadanie nr 1.

Zadanie polega na dwukrotnym policzeniu sumy wektorów na 4 różne sposoby i w dwóch różnych precyzjach (Float32 i Float64) – za drugim razem z obcięciem wartości wektorów o jedno miejsce o przecinku.

Sposoby obliczeń:

1° w przód

2° w tył

3° od największego do najmniejszego

4° od najmniejszego do największego

Wartości wektorów:

x = [2.718281828, −3.141592654, 1.414213562, 0.5772156649, 0.3010299957]

y = [1486.2497, 878366.9879, −22.37492, 4773714.647, 0.000185049]

x’= [2.718281828, −3.141592654, 1.414213562, 0.577215664, 0.301029995]

y’ = [1486.2497, 878366.9879, −22.37492, 4773714.647, 0.000185049]

Float32:

|  |  |  |
| --- | --- | --- |
| sposób | x, y | x’, y’ |
| 1° | 1.0251881368296672e-10 | -0.004296342739891585 |
| 2° | -0.4543457 | -0.4543457 |
| 3° | -0.5 | -0.5 |
| 4° | -0.5 | -0.5 |

Float64:

|  |  |  |  |
| --- | --- | --- | --- |
| sposób | | x, y | x’, y’ |
| 1° | 1.0251881368296672e-10 | | -0.004296342739891585 |
| 2° | -1.5643308870494366e-10 | | -0.004296342998713953 |
| 3° | 0.0 | | -0.004296342842280865 |
| 4° | 0.0 | | -0.004296342842280865 |

Wnioski:

We Float32 wyniki dla różnych wartości wektorów są takie same. Wynika to z małej precyzji tej arytmetyki. Składowe wektorów nie są zapisywane w dokładny sposób, więc zmiana 11 miejsca po przecinku nie wypłynęła na zmianę wyniku.

We Float64, wyniki różnią się od siebie, w szczególności dla pierwszego i drugiego sposobu obliczeń.

Z uwagi na to, że wyniki zmieniają się ze względu na dane wejściowe ( w zależności od różnej precyzji) a nie od zastosowanego algorytmu ( dla Float32 wyniki wyszły poprawne, mimo, że zastosowano taki sam algorytm jak dla Float64), mamy do czynienia z zadaniem źle uwarunkowanym, ponieważ współczynnik uwarunkowania ( stosunek zmiany wartości wyniku względem zmiany wartości danych wejściowych) jest bardzo wysoki.

Zadanie nr 2.

Zadanie polega na obliczeniu granicy oraz wygenerowaniu kilku wykresów funkcji:

Obliczona granica funkcji przy x dążącym do nieskończoności wynosi:

Poniżej zamieszczono wykresy wygenerowane w 3 różnych programach – w dużym i bardzo małym przybliżeniu.

Wykresy funkcji są takie same tylko dla dużego przybliżenia na punkt przecięcia się osi OX i OY.

|  |  |  |
| --- | --- | --- |
| [http://pl.easima.co1m/](http://pl.easima.com/)  A picture containing athletic game, text  Description automatically generated | <https://obliczone.pl/>  A picture containing wall, text, indoor, shoji  Description automatically generated | <https://www.desmos.com/>  A picture containing sky, object  Description automatically generated |
|  |  |  |

Dla małych wartości x wszystkie wykresy są podobne. Jednak dla x ≈ 37 wyniki zaczynają się znacznie różnić. Wynika to z faktu, że dla większych wartości x , wartość funkcji rośnie wykładniczo, a wartość logarytmu maleje, przez co wraz ze wzrostem x mnożone są liczby coraz bardziej oddalone od siebie, co generuje błędy obliczeniowe. Można zauważyć, że pomimo odbiegania wykresu w punkcie x ≈ 37, każda następna wartość funkcji równa jest 0, co nie jest zgodne z obliczoną granicą funkcji 1.

Zadanie nr 3.

Zadanie polega na rozwiązaniu układu równań i obliczenia błędu względnego dla dwóch macierzy: Hilberta i macierzy wygenerowanej losowo z użyciem parametrów rozmiaru i wskaźnika uwarunkowania. Równania są rozwiązane za pomocą dwóch sposobów: eliminacji Gaussa i za pomocą funkcji inv(A)\*b. Dla obliczeń zastosowano arytmetykę Float64.

Wartości błędów względnych dla obu metod:

Macierz Hilberta:

|  |  |  |  |
| --- | --- | --- | --- |
| Rząd macierzy | Wskaźnik uwarunkowania | Eliminacja Gaussa | inv(A)\*b |
| 1 | 1 | 0.0 | 0.0 |
| 2 | 19.28147006790397 | 5.661048867003676e-16 | 1.1240151438116956e-15 |
| 3 | 524.0567775860644 | 8.022593772267726e-15 | 9.825526038180824e-15 |
| 4 | 15513.73873892924 | 4.4515459601812086e-13 | 2.950477637286781e-13 |
| 5 | 476607.25024259434 | 1.6828426299227195e-12 | 8.500055777753297e-12 |
| 6 | 1.4951058642254665e7 | 2.618913302311624e-10 | 3.3474135070361745e-10 |
| 7 | 4.75367356583129e8 | 1.2606867224171548e-8 | 5.163959183577243e-9 |
| 8 | 1.5257575538060041e10 | 1.026543065687064e-7 | 2.698715074276819e-7 |
| 9 | 4.931537564468762e11 | 4.83235712050215e-6 | 9.175846868614517e-6 |
| 10 | 1.6024416992541715e13 | 0.0006329153722983848 | 0.00045521422517408853 |

Macierz losowa:

|  |  |  |  |
| --- | --- | --- | --- |
| Rząd macierzy | Wskaźnik uwarunkowania | Eliminacja Gaussa | inv(A)\*b |
| 5 |  | 2.764433037591714e-16 | 2.7194799110210365e-16 |
| 5 |  | 3.7155169009665e-16 | 2.7194799110210365e-16 |
| 5 |  | 3.7155169009665e-16 | 3.7155169009665e-16 |
| 5 |  | 1.2161883888976234e-16 | 7.129326349629486e-15 |
| 5 |  | 3.163183238604098e-10 | 2.6731521947473123e-10 |
| 4 |  | 0.4200750875647017 | 0.37253396809651357 |
| 10 |  | .248030287623391e-16 | 2.6506211417561425e-16 |
| 10 |  | 2.9163173068810656e-16 | 2.2752801345137457e-16 |
| 10 |  | 2.0420741064951578e-14 | 1.9983675205560382e-14 |
| 10 |  | 2.328872126921145e-10 | 2.84522042456994e-10 |
| 10 |  | 1.5275793715854914e-5 | 1.4873242757059617e-5 |
| 9 |  | 1.2858102914526368 | 1.24179295611289 |
| 20 |  | 5.026748538604306e-16 | 4.256659361141682e-16 |
| 20 |  | 5.416250078505087e-16 | 4.3920512659784095e-16 |
| 20 |  | 2.2696716992800822e-14 | 2.3716193316436377e-14 |
| 20 |  | 3.0464162400936694e-10 | 2.5259862524245926e-10 |
| 20 |  | 4.557110399855689e-5 | 4.468450143251334e-5 |
| 19 |  | 0.5103437846724539 | 0.5179840684009378 |

Dla macierzy Hilberta wartości błędów względnych dla obu metod są zbliżone. Wskaźnik uwarunkowania rośnie wraz ze wzrostem rzędu macierzy i wraz z wartością błędu względnego. Oznacza to, że im wyższy rząd macierzy Hilberta, tym mniejsza szansa na otrzymanie poprawnego wyniku. Wynika z tego, że zadanie jest źle uwarunkowane dla podanej macierzy, ponieważ to nie algorytm, ale dane wejściowe (w tym wypadku macierze Hilberta) generują błędy obliczeniowe.

Dla macierzy wygenerowanej losowo wartości błędów względnych dla tych dwóch metod są prawie takie same. Tutaj także następuje zwiększenie się błędu względnego przy zwiększaniu rzędu macierzy, ale tempo wzrostu nie jest jednak tak duże, jak w przypadku macierzy Hilberta.

Zadanie nr 4.

Zadanie polega na obliczeniu miejsc zerowych trzech wielomianów - w postaci naturalnej i kanonicznej oraz minimalnie zmodyfikowanej postaci naturalnej. Do wykonania zadania należy użyć funkcji języka Julia: Poly, poly, polyval.

* Poly – generuje wielomian ze współczynników
* poly – generuje wielomian z miejsc zerowych
* polyval – oblicza wartość w punkcie x wielomianu podanego za pomocą miejsc zerowych
* roots – zwraca listę miejsc zerowych wielomianu w postaci normalnej

Obliczone wartości miejsc zerowych dla podanego wielomianu P:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| k | Obliczone miejsce zerowe | P() | p() | -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 |

Obliczone wartości miejsc zerowych dla zmienionego wielomianu P:

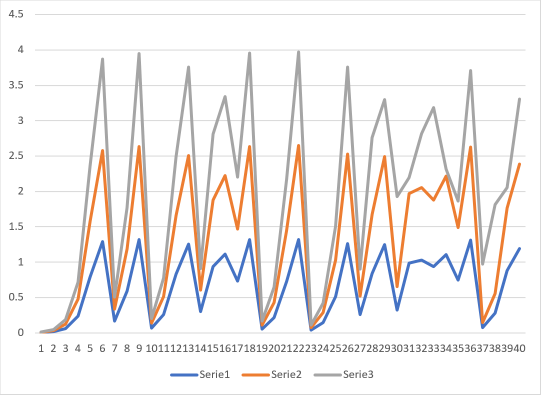
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| k | Obliczone miejsce zerowe | P() | p() | -k |
| ­1 | 0.9999999999996989 | 20992.0 | 22016.0 | .6431300764452317e-13 |
| 2 | 2.0000000000283182 | 349184.0 | 365568.0 | 5.503730804434781e-11 |
| 3 | 2.9999999995920965 | 2.221568e6 | 2.295296e6 | 3.3965799062229962e-9 |
| 4 | 3.9999999837375317 | 1.046784e7 | 1.0729984e7 | 8.972436216225788e-8 |
| 5 | 5.000000665769791 | 3.9463936e7 | 4.3303936e7 | 1.4261120897529622e-6 |
| 6 | 5.999989245824773 | 1.29148416e8 | 2.06­­120448e8 | 2.0476673030955794e-5 |
| 7 | 7.000102002793008 | 3.88123136e8 | 1.757670912e9 | 0.00039792957757978087 |
| 8 | 7.999355829607762 | 1.072547328e9 | 1.8525486592e10 | 0.007772029099445632 |
| 9 | 9.002915294362053 | 3.065575424e9 | 1.37174317056e11 | 0.0841836320674414 |
| 10 | 9.990413042481725 | 7.143113638035824e9 | 1.4912633816754019e12 | 0.6519586830380406 |
| 11 | 11.025022932909318 | 7.143113638035824e9 | 1.4912633816754019e12 | 1.1109180272716561 |
| 12 | 11.953283253846857 | 3.357756113171857e10 | 3.2960214141301664e13 | 1.665281290598479 |
| 13 | 13.07431403244734 | 3.357756113171857e10 | 3.2960214141301664e13 | 2.045820276678428 |
| 14 | 13.914755591802127 | 1.0612064533081976e11 | 9.545941595183662e14 | 2.5188358711909045 |
| 15 | 15.075493799699476 | 1.0612064533081976e11 | 9.545941595183662e14 | 2.7128805312847097 |
| 16 | 15.946286716607972 | 3.315103475981763e11 | 2.7420894016764064e16 | 2.9060018735375106 |
| 17 | 17.025427146237412 | 3.315103475981763e11 | 2.7420894016764064e16 | 2.825483521349608 |
| 18 | 17.99092135271648 | 9.539424609817828e12 | 4.2525024879934694e17 | 2.454021446312976 |
| 19 | 19.00190981829944 | 9.539424609817828e12 | 4.2525024879934694e17 | 2.004329444309949 |
| 20 | 19.999809291236637 | 1.114453504512e13 | 1.3743733197249713e18 | 0.8469102151947894 |

Można zauważyć, że najbardziej wiarygodne wyniki dawało obliczanie miejsc zerowych bezpośrednio za pomocą funkcji roots(). Mimo, że różnice między obliczonymi miejscami zerowymi a faktycznymi danymi jest niewielka, odkłada się ona w takim stopniu, że obliczona wartość wielomianu pod dwiema postaciami w znaczycym stopniu różni się od 0 - są to rzędy sięgające dziesiątek bilionów. Odchylenie od prawidłowego wyniku zwiększa się wraz z obliczaniem wartości funkcji dla kolejnego miejsca zerowego. Jest to kolejny przykład zadania źle uwarunkowanego, gdzie niedokładne dane przechodzą przez szereg funkcji i zniekształcają się do tego stopnia, że błąd odkładający się przy każdym działaniu uniemożliwia otrzymanie wiarygodnego wyniku.

Zadanie nr 5.

Zadanie polega na obliczeniu kolejnych wartości ciągu rekurencyjnego := + r (1 − )

* 40 iteracji w arytmetyce Float32 (kolor niebieski)
* 40 iteracji w arytmetyce Float65 (kolor szary)
* 40 iteracji w arytmetyce Float32 zaokrąglając 10 wyraz do 3 miejsc po przecinku



Jak widać na powyższym wykresie, różnica w wartościach kolejnych liczb jest ogromna, a wynika to z drobnej zmiany jak np. zastosowanie arytmetyki innej precyzji lub zaokrąglenie jednego z wyników. Różnica ta zaczyna rosnąć z każdym wyrazem, co wynika z tego, że mały błąd zaimplementowany na wejściu „podwaja” swoją wartość przy każdej iteracji. Jeśli wykonujemy większą liczbę operacji (tak jak tutaj – 80 powtórzeń) różnica między wynikami jest nieproporcjonalnie duża do różnicy danych wejściowych. Jest więc to zadanie źle uwarunkowane dla podanych wartości.

Zadanie nr 6.

Zadanie polega na przetestowaniu generowania ciągu liczbowego:

dla 7 zestawów parametrów:

1. c = −2 i = 1

2. c = −2 i = 2

3. c = −2 i = 1.99999999999999

4. c = −1 i = 1

5. c = −1 i = −1

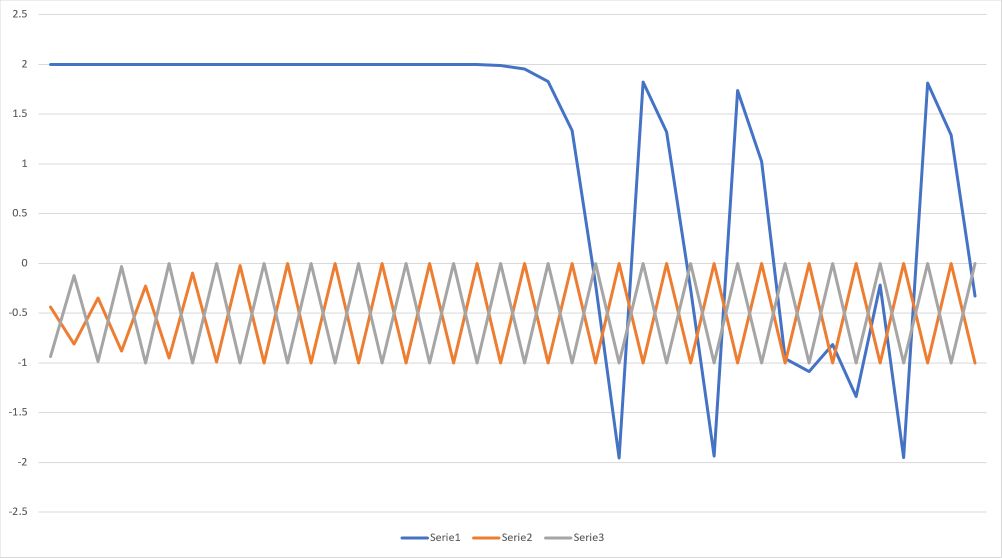
6. c = −1 i = 0.75

7. c = −1 i = 0.25

Dla opcji 1, 2, 4 i 5 wygenerowane ciągi są stałe albo ich wartości oscylują między dwiema wartościami, więc nie występują tam żadne błędy obliczeniowe.

Poniżej za pomocą wykresu przestawiono

* Niebieską linią: c = −2 i = 1.99999999999999
* Pomarańczową linią: c = −1 i = 0.75
* Szarą linią: c = −1 i = 0.25



W przypadkach 3, 6, 7 niedokładności obliczeniowe nakładają się z każdą kolejną iteracją, bo każda kolejna iteracja bierze jako dane wejściowe wynik poprzedniej iteracji już z błędem, a następnie jeszcze zwiększa ten błąd. Przykład nr 3 jest najlepszym przykładem zadania źle uwarunkowanego dla podanego wzoru rekurencyjnego, ponieważ to dane wejściowe, a nie działania arytmetyczne, jakie na nich wykonujemy generują błędy obliczeniowe i wysoki współczynnik uwarunkowania.