Obliczenia naukowe

Sprawozdanie nr 2

Barbara Banaszak
236514

5-11-2018

1 Zadanie 1

1.1 Opis problemu

Zadanie polega na powtórzeniu obliczeń z zadania 5, z listy 1 po dokonaniu dwóch nieznacznych zmian (usuwamy ostatnią 9 z x_4 i ostatnią 7 z x_5, czyli obliczeniu iloczynu skalarnego wektorów:
$x = [2.718281828, -3.141592654, 1.414213562, 0.577215664, 0.301029995]$
$y = [1486.2497, 878366.9879, -22.37492, 4773714.647, 0.000185049]$

1.2 Rozwiązanie

Powtarzamy obliczenia, jakie wykonywaliśmy w zadaniu 5 z listy 1, dla arytmetyk Float32 i Float64:

1. $\sum_{i=1}^{5} x_i y_i$
2. $\sum_{i=5}^{1} x_i y_i$
3. dodajemy dodatnie liczby w porządku od najmniejszej do największej, ujemne w porządku od najmniejszej do największej, poczym dodajemy do siebie sumy częściowe
4. odwrotnosć punktu 3.

1.3 Wyniki

Zadanie 5, Lista 1		
Sposób	Float32	Float64
1.	-0.4999443	1.0251881368296672e-10
2.	-0.4543457	-1.5643308870494366e-10
3.	-0.5	0.0
4.	-0.5	0.0

Bieżące wyniki		
Sposób	Float32	Float64
1.	-0.4999443	-0.004296342739891585
2.	-0.4543457	-0.004296342998713953
3.	-0.5	-0.004296342842280865
4.	-0.5	-0.004296342842280865

1.4 Wnioski

Możemy zauważyć, że w przypadku arytmetyki Float32 dokonane zmiany nie wpłynęły na otrzymane wyniki. Stało się tak, dlatego że precyzja tej arytmetyki (rzędu 10^{-7}) jest znacząco mniejsza niż precyzja zapisu liczb (rzędu 10^{-9}), w których dokonaliśmy zmian. Natomiast dla obliczeń w arytmetyce Float64 dokonane zmiany sprawiły, że otrzymaliśmy zupełnie inne wyniki. Precyzja arytmetyki jest na tyle duża, że zmiany zostały uchwycone. Możemy także zauważyć, że dla każdego sposobu obliczenia iloczynu skalarnego otrzymaliśmy niemal identyczne wyniki, co pozwala przypuszczać, że wynik ten jest poprawny.

2 Zadanie 2

2.1 Opis problemu

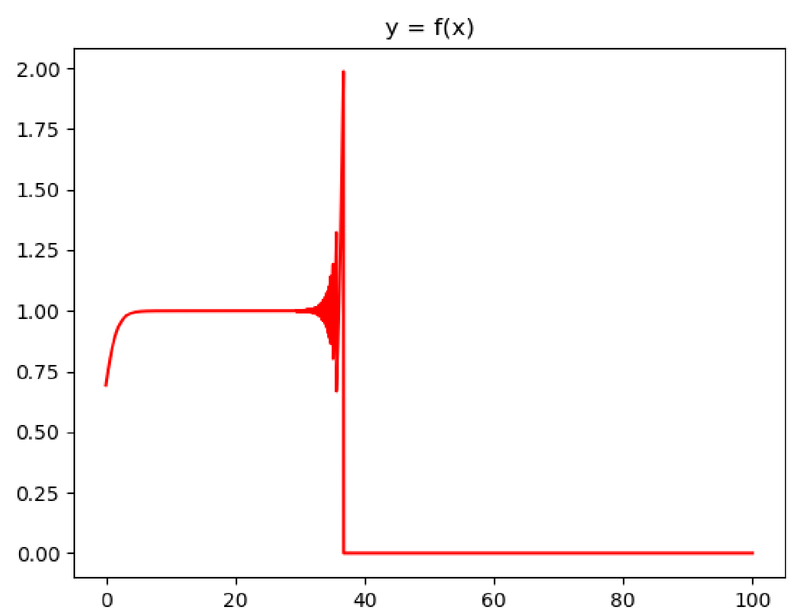
W zadaniu mamy policzyć granicę funkcji $f(x) = e^x ln(1 + e^{-x})$ $\lim_{x\to\infty} f(x)$ oraz porównać obliczoną granicę z wykresami tej funkcji.

2.2 Rozwiązanie

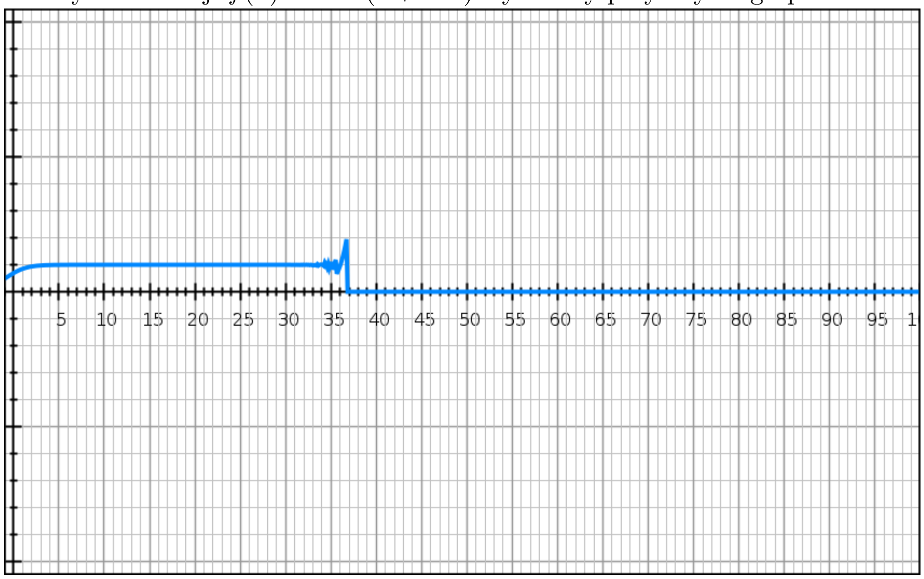
Wykresy funkcji $f(x) = e^x ln(1 + e^{-x})$ narysujemy przy użyciu biblioteki Pythona matplotlib oraz programu graphsketch.

2.3 Wyniki

Wykres funkcji $f(x) = e^x ln(1 + e^{-x})$ wykonany z użyciem matplotlib:



Wykres funkcji $f(x) = e^x ln(1 + e^{-x})$ wykonany przy użyciu graphsketch:



Granica $\lim_{x->\infty} f(x)$ policzona analitycznie z wykorzystaniem reguły d'lHospitala:

$\lim_{x->\infty} e^x ln(1 + e^{-x}) = \lim_{x->\infty} \frac{ln(1+e^{-x})}{e^{-x}} = [\frac{0}{0}] = \lim_{x->\infty} \frac{\frac{d}{dx} ln(1+e^{-x})}{\frac{d}{dx} e^{-x}} = \lim_{x->\infty} \frac{-e^{-x}}{-e^{-x}(e^{-x}+1)} = \lim_{x->\infty} \frac{1}{e^{-x}+1} = 1$

2.4 Wnioski

Wyraźnie widać, że granica funkcji wyliczona analitycznie nie pokrywa się z granicą funkcji jaką sugerują nam narysowane wykresy. Z wykresów wynika, że $\lim_{x\to\infty} f(x) = 0$, natomiast w rzeczywistości wynosi ona 1. Na wykresach możemy także zaobserwować dość nietypową oscylację w przedziale $(30, 40)$ niecharakterystyczną dla funkcji $f(x)$. Zaburzenia te wywołuje dodawanie bardzo małej wartości e^{-x} do jedynki, a potem mnożenie otrzymanej wartości przez dużą liczbę e^x. Widzimy także, że wartości funkcji kończąc oscylować, gwałtownie spadają do 0. Dzieje się tak dlatego, że wartości e^{-x} są na tyle małe, że zostają pochłonięte przez 1. Otrzymujemy wtedy $e^x ln(1)$, czyli mnożenie $0 \cdot \infty$, czyli tak naprawdę 0.

3 Zadanie 3

3.1 Opis problemu

Zadanie polega na wygenerowaniu macierzy Hilberta H_n (przy pomocy funkcji **hilb**), macierzy losowej R_n (przy pomocy funkcji **matcond**) i rozwiązaniu równania $b = Ax$, gdzie $A-$ macierz, $x = (1, 1, ..., 1)^T$, $b-$wektor prawych stron, za pomocą metody eliminacji Gaussa ($x = A/b$) oraz poprzez odwrócenie macierzy ($x = A^{-1} * b$). Eksperymenty należy przeprowadzić:

- dla H_n - z rosnącym stopniem $n > 1$
- dla R_n - dla $n = 5, 10, 20$ z rosnącym wskaźniekiem uwarunkowania $c = 1, 10, 10^3, 10^7, 10^{12}, 10^{16}$

dla każdego przypadku należy policzyć błąd względny ($\widetilde{x}$).

3.2 Rozwiązanie

Do rozwiązania zadania zostały zaimplementowane funkcje służące do rozwiązania równania liniowego metodą Gaussa(**gaussianElimination()**) oraz metodą inwersji (**inversionMethod()**), a także funkcja licząca błąd względny uzyskanych rozwiązań (**errorCount()**).

3.3 Wyniki

Wyniki dla macierzy Hilberta H_n				
Rozmiar	Rząd	COND	Błąd względny Gaussa	Błąd względny inwersji
1x1	1	1.0	0.0	0.0
2x2	2	1.928147006790e+01	5.661048867004e-16	1.404333387431e-15
3x3	3	5.240567775861e+02	8.022593772268e-15	0.000000000000e+00
4x4	4	1.551373873893e+04	4.137409622430e-14	0.000000000000e+00
5x5	5	4.766072502426e+05	1.682842629923e-12	3.354436058436e-12
6x6	6	1.495105864225e+07	2.618913302312e-10	2.016375940435e-10
7x7	7	4.753673565831e+08	1.260686722417e-08	4.713280397232e-09
8x8	8	1.525757553806e+10	6.124089555723e-08	3.077483903096e-07
9x9	9	4.931537564469e+11	3.875163418503e-06	4.541268303177e-06
10x10	10	1.602441699254e+13	8.670390237097e-05	2.501493411825e-04
11x11	11	5.222677939280e+14	1.582780815859e-04	7.618304284316e-03
12x12	11	1.751473190709e+16	1.339620837209e-01	2.589941208047e-01
13x13	11	3.344143497338e+18	1.103970111787e-01	5.331275639427e+00
14x14	12	6.200786263161e+17	1.455408712766e+00	8.714992751048e+00
15x15	12	3.674392953468e+17	4.696668350857e+00	7.344641453111e+00
16x16	12	7.865467778432e+17	5.415518954565e+01	2.984884207074e+01
17x17	12	1.263684342666e+18	1.370723668384e+01	1.051694237837e+01
18x18	12	2.244630992919e+18	9.134134521198e+00	7.575475905055e+00
19x19	13	6.471953976542e+18	9.720589712656e+00	1.223376139376e+01
20x20	13	1.355365790869e+18	7.549915039473e+00	2.206269725787e+01

Wyniki dla macierzy losowej R_n				
Rozmiar	Rząd	COND	Błąd względny Gaussa	Błąd względny inwersji
5x5	5	1.000000000000e+00	1.790180836525e-16	2.275280134514e-16
5x5	5	1.000000000000e+01	2.979040983897e-16	3.748544367384e-16
5x5	5	1.000000000000e+03	1.491372965516e-14	1.280949133596e-14
5x5	5	9.999999997246e+06	5.668391800431e-11	1.041250292910e-10
5x5	5	9.999787335635e+11	1.053021419619e-05	1.332416473127e-05
5x5	4	5.653235649955e+15	1.175643838705e-01	1.685436258410e-01
10x10	10	1.000000000000e+00	2.192514798397e-16	1.922962686384e-16
10x10	10	1.000000000000e+01	3.457769959780e-16	4.915166856007e-16
10x10	10	1.000000000000e+03	3.829867159113e-16	4.514118214889e-15
10x10	10	1.000000000263e+07	5.399383229208e-10	5.445265623845e-10
10x10	10	9.999840038091e+11	1.801229963125e-05	2.250802715155e-05
10x10	9	9.583832983647e+15	2.436173418446e-02	3.163821364268e-02
20x20	20	1.000000000000e+00	4.335559509131e-16	3.805652793121e-16
20x20	20	1.000000000000e+01	7.850462293419e-16	5.051209746510e-16
20x20	20	9.999999999999e+02	1.600539653396e-14	1.745445735611e-14
20x20	20	9.999999998702e+06	8.055483454194e-11	6.267467824323e-11
20x20	20	9.999542399355e+11	7.104570148002e-06	4.533715943910e-06
20x20	19	6.693353984402e+15	1.526719144206e-02	4.163410185773e-02

3.4 Wnioski

Przyglając się otrzymanym wynikom widzimy, że wskaźnik uwarunkowania macierzy ma decydujący wpływ na dokładność rozwiązania układu równań, im wskaźnik uwarunkowania macierzy będzie większy tym większy jest błąd rozwiązania. Widać także wyraźnie, że macierz Hilberta jest macierzą źle uwarunkowanąwskaźnik uwarunkowania dla macierzy H_n jest wielokrotnie większy niż dla macierzy R_n i im większa macierz Hilberta wskaźnik ten rośnie jeszcze bardziej. Możemy również zauważyć, że w przypadku macierzy Hilberta rozwiązując układ równań dużo bardziej dokładne wyniki otrzymaliśmy korzystając z metody inwersji.

4 Zadanie 4

4.1 Opis problemu

Zadanie polega na obliczeniu pierwiastków "złośliwego wielomianu" Wilkinsona:
$P(x) = x^{20} - 210x^{19} + 20615x^{18}1256850x^{17} + 53327946x^{16} - 1672280820x^{15} + 40171771630x^{14} - 756111184500x^{13} + 11310276995381x^{12} - {}^{11} + 1307535010540395x^{10} - 10142299865511450x^{9} + 63030812099294896x^{8} - 311333643161390640x^{7} + 1206647803780373360x^{6} - 3599979517947607200x^{5} + 8037811822645051776x^{4} - 12870931245150988800x^{3} + 13803759753640704000x^{2} - 8752948036761600000x + 2432902008176640000$, ich sprawdzeniu oraz powtórzeniu obliczeń dla wielomianu Wilkinsona ze współczynnikiem -210 zmienionym na $-210 - 2^{-23}$.

4.2 Rozwiązanie

Do wykonania zadania połużyliśmy się następującymi funkcjami dostępnymi w języku Julia w pakiecie Polynomials:

- **roots (P)** liczenie pierwiatków wielomianu P
- **Poly (p)** tworzenie wielomianu z tablicy p zawierającej współczynniki wielomianu
- **poly (p)** tworzenie wielomianu z tablicy p zawierającej miejsca zerowe wielomianu
- **polyval (P,x)** liczenie wartości wielomianu P w x z wykorzystaniem metody Hornera

Z wykorzystaniem powyższych funkcji tworzymy wielomiany P (na podstawie współczynników wielomianu Wilkinsona) oraz p (na podstawie miejsc zerowych wielomianu Wilkinsona). Liczymy miejsca zerowe wielomianu P oraz wyliczamy: $|P(z_k)|$, $|p(z_k)|$, $|z_k - k|$, gdzie $0 \leqslant k \leqslant 20$.

4.3 Wyniki

Tabele z wynikami zostały umieszczone na ostatniej stronie.
Pierwiatki wielomianu Wilkinsona policzone metodą **roots**:
$[0.9999999999996989, 2.0000000000283182, 2.9999999995920965, 3.9999999837375317, 5.000000665769791, 5.999989245824773, 7.000102002793008, 7.999355829607762, 9.002915294362053, 9.990413042481725, 11.025022932909318, 11.953283253846857, 13.07431403244734, 13.914755591802127, 15.075493799699476, 15.946286716607972, 17.025427146237412, 17.99092135271648, 19.00190981829944, 19.999809291236637]$

4.4 Wnioski

Możemy zauważyć, że obliczone wartości pierwiastków wielomianu Wilkinsona nie różnią się znacząco od ich rzeczywistych wartości, szczególnie dla mniejszych wartości pierwiastków, dla których mamy błąd względny rzędu $10^{-13}, 10^{-11}, 10^{-10}$. Można by powiedzieć, że odchylenia wartości pierwiastków są niewielkie, więc możemy je zignorować. Jednakże okazuje się, że pomimo nieznacznego błędu, licząc wartość wielomianu Wilkinsona dla obliczonych pierwiastków zamiast spodziewanego zera dostajemy wartości rzędu 10^6 i większe. Dzieje się tak, dlatego że współczynniki wielomianu są bardzo duże i błąd obliczenia pierwiastka jest za każdym razem powielany. Nie bez znaczenia jest również, to że arytmetyka Float64 ma w języku Julia od 15 do 17 cyfr znaczących, zatem nie wszystkie współczynniki wielomianu będą dokładnie reprezentowane. Widać to jeszcze wyraźniej dla wielomianu Wilkinsona z zaburzonym współczynnikiem. W wielomianie, który ma wszystkie miejsca zerowe rzeczywiste, nagle pojawiają się pierwiastki zespolone.

5 Zadanie 5

5.1 Opis problemu

Zadanie polega na przeprowadzeniu eksperymentów, z wykorzystaniem modelu logistycznego opisanego następującym równaniem rekurencyjnym:
$p_{n+1} := p_n + rp_n(1 - p_n)$ dla $n = 1, 2, 3, ...$

1. dla danych $p = 0.01$ oraz $r = 3$ wykonać 40 iteracji, następnie wykonać ponownie 40 iteracji, gdzie po 10 iteracji zastosować obcięcie do trzech cyfr po przecinku.
2. dla danych $p = 0.01$ oraz $r = 3$ wykonać 40 iteracji dla arytmetyk Float32, Float64

5.2 Rozwiązanie

W celu rozwiązania zadania zostały zaimplementowane 2 algorytmy, wykonujące podaną rekurencję dla zadanych danych, w tym jeden uwzględniający dokonywanie obcięcia.

5.3 Wyniki

Tabela przedstawiająca wybrane iteracje			
Nr	Float32	Float32 z obcięciem	Float64
1	0.0397	0.0397	0.0397
2	0.15407173	0.15407173	0.15407173000000002
3	0.5450726	0.5450726	0.5450726260444213
4	1.288978	1.288978	1.2889780011888006
5	0.17151916	0.17151916	0.17151914210917552
6	0.59782016	0.59782016	0.5978201201070994
7	1.3191137	1.3191137	1.3191137924137974
8	0.05627179	0.05627179	0.056271577646256565
9	0.21558762	0.21558762	0.21558683923263022
10	0.7229164	0.7229164	0.722914301179573
11	1.3238412	1.323813	1.3238419441684408
12	0.03769815	0.03780961	0.03769529725473175
13	0.14652914	0.14694974	0.14651838271355924
14	0.5217042	0.5230163	0.521670621435246
15	1.270291	1.271427	1.2702617739350768
20	0.5943414	0.5096725	0.5965293124946907
21	1.3176405	1.2593918	1.3185755879825978
22	0.062032342	0.2793641	0.058377608259430724
30	0.8095675	0.6959832	0.37414648963928676
31	1.2720714	1.330755	1.0766291714289444
32	0.23378885	0.0102933645	0.8291255674004515
39	1.1972201	1.241536	0.0029091569028512065
40	0.48887253	0.341909	0.011611238029748606

5.4 Wnioski

Przeprowadzone eksperymenty pokazują, że pozornie drobne zmiany w danych, jak obcięcie kilku miejsc po przecinku (które mogłoby w rzeczywistości wynikać np. z niedokładnego przechowywania danych) mogą mieć bardzo duży wpływ na wyniki końcowe.

W przypadku obliczeń wykonywanych w różnych arytmetykach otrzymaliśmy jeszcze bardziej różniące się wyniki. Różnice te wynikają z różnej precyzji obu tych arytmetyk i także pogłębiają się wraz z kolejnym wywołaniem funkcji rekurencyjnej.

6 Zadanie 6

6.1 Opis problemu

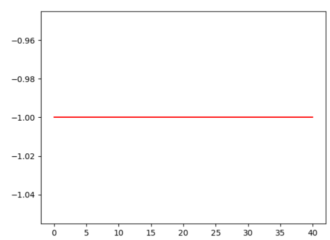
Zadanie polega na przeprowadzeniu 40 iteracji równania rekurencyjnego:
$x_{n+1} := x_n^2 + c$, dla $n = 1, 2, 3, ...$ dla danych:

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

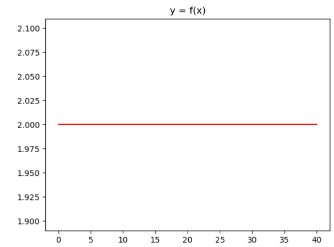
6.2 Rozwiązanie

W programie została zaimplementowana funkcja licząca rekurencyjnie podane wyrażenie, jako parametry przyjmująca c i x_0. Użyto także biblioteki matplotlib Pythona, w celu narysowania wykresów iteracji graficznej.

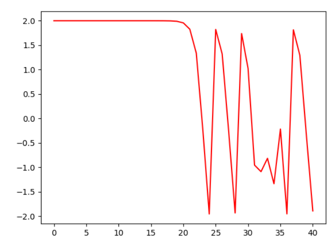
6.3 Wyniki



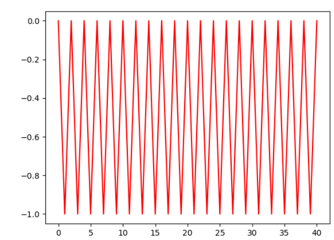
$c = -2\ x_0 = 1$



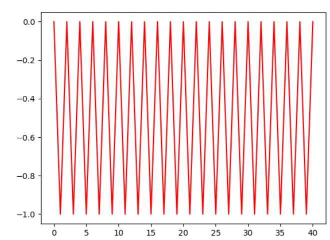
$c = -2\ x_0 = 2$



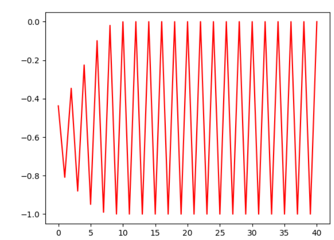
$c = -1\ x_0 = 1.99999999999999$



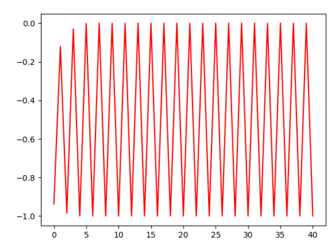
$c = -1\ x_0 = 1$



$c = -1\ x_0 = -1$



$c = -1\ x_0 = 0.75$



$c = -1\ x_0 = 0.25$

6.4 Wnioski

W tym zadaniu mamy doczynienia z podobnym problemem co w zadaniu 5. Dla podpunktu 1)i2) otrzymaliśmy funkcje stałe i dla parametrów z tych podpunktów rekurencje rzeczywiście tworzą ciąg stały. Funkcja zachowuje się stabilnie. W podpunkcie 3) można zaobserwować jak niedokładność zapisu, po niedługim czasie sprawia, że otrzymywane wyniki stają się nieuporządkowane. W podpunktach 4), 5) funkcja także zachowuje się stabilnie, a kolejne jej iteracje, dają rzeczywiste wartości kolejnych elementów ciągu. Natomiast dla podpunktów 7), 8) możemy zaobserwować jak po kilku iteracjach funkcja również się stabilizuje. Z powyższego zadania można wysnuć wniosek, że przeprowadzenie iteracji graficznych może pomagać nam w ocenie prawdziwości wyników obliczeń.

Tabele do zadania 4

Wyniki dla wielomianu Wilkinsona			
Nr	$\|P(z_k)\|$	$\|p(z_k)\|$	$\|z_k - k\|$
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
10.	-1.2707126784e10	-1.2696907264e10	0.009586957518274986
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

Wybrane wyniki dla wielomianu Wilkinsona ze zmienionym współczynnikiem			
Nr	z_k	$\|P(z_k)\|$	$\|p(z_k)\|$
1.	0.9999999999998357 + 0.0im	20992.0 + 0.0im	22016.0 + 0.0im
2.	2.0000000000550373 + 0.0im	349184.0 + 0.0im	365568.0 + 0.0im
3.	2.99999999660342 + 0.0im	2.221568e6 + 0.0im	2.295296e6 + 0.0im
4.	4.000000089724362 + 0.0im	1.046784e7 + 0.0im	1.0729984e7 + 0.0im
5.	4.99999857388791 + 0.0im	3.9463936e7 + 0.0im	4.3303936e7 + 0.0im
10.	10.095455630535774 - 0.6449328236240688im	6.226891264e9 - 3.499985376e9im	5.2861543168e11 - 1.394428986688e12im
11.	10.095455630535774 + 0.6449328236240688im	6.226891264e9 + 3.499985376e9im	5.2861543168e11 + 1.394428986688e12im
12.	11.793890586174369 - 1.6524771364075785im	1.2597167616e10 - 3.1124973568e10im	-2.8965491949568e13 - 1.5728191007232e13im
15.	13.992406684487216 + 2.5188244257108443im	6.2258671616e10 + 8.5938636096e10im	-9.26688892655616e14 - 2.29123777935168e14im
19.	19.5024423688181 + 1.940331978642903im	2.046021647872e12 + 9.317425465376e12im	-1.3107912683617587e17 + 4.0454423319554586e17im
20.	20.84691021519479 + 0.0im	1.114453504512e13 + 0.0im	1.3743733197249713e18 + 0.0im

Wybrane wyniki dla wielomianu Wilkinsona ze zmienionym współczynnikiem		
Nr	z_k	$\|z_k - k\|$
1.	0.9999999999998357 + 0.0im	1.6431300764452317e-13
2.	2.0000000000550373 + 0.0im	5.503730804434781e-11
3.	2.99999999660342 + 0.0im	3.3965799062229962e-9
4.	4.000000089724362 + 0.0im	8.972436216225788e-8
5.	4.99999857388791 + 0.0im	1.4261120897529622e-6
10.	10.095455630535774 - 0.6449328236240688im	0.6519586830380407
11.	10.095455630535774 + 0.6449328236240688im	1.1109180272716561
12.	11.793890586174369 - 1.6524771364075785im	1.665281290598479
15.	13.992406684487216 + 2.5188244257108443im	2.7128805312847097
19.	19.5024423688181 + 1.940331978642903im	2.0043294443099486
20.	20.84691021519479 + 0.0im	0.8469102151947894