Metody Obliczeniowe w Nauce i Technice
Natalia Bratek, 12.03.2024
Laboratorium 02

Arytmetyka komputerowa cd.

1. Treści zadań

1.1 Zadanie pierwsze

Napisać algorytm do obliczenia funkcji wykładniczej e^x przy pomocy nieskończonych szeregów
e^x= 1+ x + x^2/2! + x^3/3! + …

(1a) Wykonując sumowanie w naturalnej kolejności, jakie kryterium zakończenia obliczeń przyjmiesz?

(1b) Proszę przetestować algorytm dla x= +-1, +-5, +-10 i porównać wyniki z wynikami wykonania standardowej funkcji exp(x)

(1c) Czy można posłużyć się szeregami w tej postaci do uzyskania dokładnych wyników dla x < 0?

(1d) Czy możesz zmienić wygląd szeregu lub w jakiś sposób przegrupować składowe żeby uzyskać dokładniejsze wyniki dla x < 0?

1.2 Zadanie drugie

Które z dwóch matematycznie ekwiwalentnych wyrażeń x**2 - y**2 oraz (x - y)*(x + y) może być obliczone dokładniej w arytmetyce zmienno-przecinkowej? Dlaczego?

1.3 Zadanie trzecie

Dla jakich wartości x i y, względem siebie, istnieje wyraźna różnica w dokładności dwóch wyrażeń?
Zakładamy, że rozwiązujemy równanie kwadratowe ax**2 + bx + c = 0, z a = 1.22, b = 3.34 i c = 2.28, wykorzystując znormalizowany system zmienno-przecinkowy z podstawa beta = 10 i dokładnością p = 3.
(a), ile wyniesie obliczona wartość b**2 – 4ac?
(b) jaka jest dokładna wartość wyróżnika w rzeczywistej (dokładnej) arytmetyce?
(c) jaki jest względny błąd w obliczonej wartości wyróżnika?

2. Rozwiązania zadań

2.1 Algorytm do obliczenia funkcji wykładniczej e^x przy pomocy nieskończonych szeregów

```
def calculate_exponential(x, epsilon=1e-10):

    sum = 1.0
    factorial = 1.0
    power_of_x = 1.0
    i = 1
    term = power_of_x

    while term > epsilon:
        factorial *= i
        power_of_x *= x
        term = power_of_x / factorial
        sum += term
        i += 1

    return sum

x = -10
print(calculate_exponential(x))
```

Rys.1

2.1.1 Kryterium zakończenia

Obliczenia zatrzymują się, gdy bezwzględna wartość kolejnego wyrazu szeregu jest mniejsza niż zadany próg ϵ. To podejście jest szczególnie przydatne, gdy chcemy zapewnić, że każdy dodany wyraz wnosi znaczący wkład do sumy końcowej.

$$|an|<\epsilon$$

Gdzie *an* to *n*-ty wyraz szeregu, a ϵ to zadana wartość progowa.

W moim przypadku $\epsilon = 10^{-10}$.

2.1.2 Testowanie algorytmu i porównanie wyników z wynikami wykonania standardowej funkcji exp(x)

	calculate_exponential(x)	math.exp(x)	Różnica
-1	0.36787944117144245	0.36787944117144233	1.1102230246251565e-16
1	2.7182818284590455	2.718281828459045	4.440892098500626e-16
-5	0.006737946999084638	0.006737946999085467	1.439820215169138e-15
5	148.41315910257657	148.4131591025766	2.842170943040401e-14
-10	4.539992967040021e-05	4.5399929762484854e-05	3.288713675966502e-13
10	22026.465794806714	22026.465794806718	7.275957614183426e-12

Tabela 1

2.1.3 Czy można posłużyć się szeregami w tej postaci do uzyskania dokładnych wyników dla x < 0?

Szereg Maclaurina dla *e^x* jest zbieżny dla wszystkich wartości *x*, zarówno dodatnich, jak i ujemnych. Oznacza to, że niezależnie od wartości *x*, sumowanie odpowiednio dużej liczby wyrazów szeregu pozwoli na osiągnięcie dowolnie wysokiej dokładności przybliżenia wartości *e^x*.
W przypadku ujemnych wartości *x*, szereg nadal skutecznie zbiega do wartości *e^x*, ale wyrazy szeregu zmieniają znaki na przemian, co jest wynikiem podnoszenia ujemnego *x* do potęgi.
Dzięki alternującym się znakom, każdy dodatni wyraz szeregu jest częściowo anulowany przez następny, ujemny wyraz, co prowadzi do zbieżności szeregu.
Jednakże, w praktycznym użyciu algorytmu dla *x*<0, szczególnie dla dużych wartości |*x*|, konieczne może być użycie większej liczby wyrazów szeregu w celu osiągnięcia pożądanej dokładności. To dlatego, że początkowe wyrazy mogą mieć duże wartości bezwzględne, zanim szereg zacznie skutecznie konwergować.
Należy także zauważyć, że dla bardzo małych wartości *x* (bliskich zero) szereg szybko zbiega do e^x, i niewiele wyrazów jest potrzebnych do osiągnięcia wysokiej dokładności.

2.1.4 Czy możesz zmienić wygląd szeregu lub w jakiś sposób przegrupować składowe żeby uzyskać dokładniejsze wyniki dla x < 0 ?

Dla ujemnych wartości *x*, zwłaszcza gdy |*x*| jest duże, bezpośrednie stosowanie szeregu Maclaurina dla *e^x* może wymagać dużej liczby wyrazów do osiągnięcia wysokiej dokładności ze względu na wolną konwergencję. Aby poprawić dokładność i szybkość konwergencji dla *x*<0, można zastosować kilka technik. Jedną z nich jest przekształcenie pierwotnego szeregu do postaci bardziej przyjaznej dla ujemnych *x*. Dla ujemnych *x*, szczególnie skuteczne może być wykorzystanie własności funkcji wykładniczej, tj. $e^{-x} = \frac{1}{e^x}$, co pozwala na sumowanie szeregu dla wartości dodatniej *x* i następnie odwrócenie wyniku. To zmniejsza problem dużych ujemnych wartości *x* do problemu obliczania *e^x* dla *x* dodatnich, gdzie szereg zbiega szybciej.

2.2 Wyrażenie obliczone bardziej dokładnie w arytmetyce zmiennoprzecinkowej

Wyrażenia matematyczne $x^2 - y^2$ oraz (x − y)*(x +y) są ekwiwalentne. W przypadku pierwszego wyrażenia, $x^2 - y^2$obliczenia polegają na wykonaniu dwóch operacji potęgowania, a następnie odejmowaniu wyników. Każda z tych operacji może wprowadzić błąd zaokrąglenia, zwłaszcza jeśli wartości x^2 i y^2 są bliskie sobie. Odejmowanie dwóch bliskich wartości może prowadzić do znaczącej utraty precyzji ze względu katastrofalną cancelację". W przypadku drugiego wyrażenia, (*x*−*y*)·(*x*+*y*), wykonuje się jedno odejmowanie i jedno dodawanie, a następnie mnoży się wyniki. To podejście zmniejsza ryzyko katastrofalnej cancelacji, ponieważ mnożenie zwykle nie prowadzi do tak znacznej utraty precyzji, jak odejmowanie. W szczególności, jeśli *x* i *y* są bliskie sobie, różnica *x*−*y* będzie mała, ale mnożenie małej różnicy przez sumę *x*+*y* jest mniej podatne na problemy z precyzją niż odejmowanie dwóch dużych, bliskich sobie wartości.

2.3 Różnica w dokładności

$$ax^2 + bx + c = 0$$

a = 1.22
b = 3.34
c = 2.28
β = 10
p = 3

2.3.1 Wartość obliczona

Przy tych założeniach obliczam wartość wyrażenia.

$b^2 - 4ac$ = ?
b^2 = 3.34 * 3.34 = 11,1556 = 11.2
4ac = 4 * 1.22 * 2.28 = 11.1264 = 11.1

$b^2 - 4ac$ = 11.2 - 11.1 = 0.1

2.3.2 Dokładna wartość wyróżnika w rzeczywistej arytmetyce

$b^2 - 4ac$ = 11,1556 - 11.1264 = 0.0292

2.3.3 Błąd względny

Błąd względny = $\frac{0.1 - 0.0292}{0.1} \approx 2.42466$

3. Bibliografia

1. Wykład
2. https://en.wikipedia.org/wiki/Catastrophic_cancellation