Projekt 1

Remigiusz Kamiński

Marzec 2023

1 Sumowanie szeregów potegowych

W projekcie bedziemy sumować szereg na cztery różne sposoby:

1.1 Sumowanie od poczatku

```
    public static double taylor_exp(double x, int n) {
double result = 0;
for (int i = 0; i <= n; i++) {
double numerator = power(x, i);
double denominator = factorial(i);

result += numerator / denominator;
}
return result;
}
```

1.2 Sumowanie od końca

```
    public static double taylor_exp_end(double x, int n) {
double result = 0;
for (int i = n; i >= 0; i--) {
double numerator = power(x, i);
double denominator = factorial(i);

result += numerator / denominator;
}
return result;
}
```

1.3 Sumujac od poczatku ale obliczajac kolejny element na podstawie poprzedniego

```
    public static double taylor_exp_prev(double x, int n) {
double result = 1;
```

```
double previous = 1;

for (int i = 1; i <= n; i++) {
double term = previous * x / i;
previous = term;
result += previous;
}
return result;
}
```

1.4 Sumujac od końca ale obliczajac kolejny element na podstawie poprzedniego

```
public static double taylor_exp_prev_end(double x, int n) {
double previous = power(x, n) / factorial(n);
double result = previous;

for (int i = n; i >= 0; i--) {
double term = previous * i / x;
result += term;
previous = term;
}
return result;
}
```

1.5 Dla porównania danych bedziemy używać funkcji wbudowanej

```
Math.exp(x)
```

2 Hipotezy

2.1 Hipoteza 1: sumowanie od końca daje dokładniejsze wyniki niż sumowanie od poczatku.

Do sprawdzenia tej hipotezy policzymy średnie wartości błedu, które wynosza:

1. Dla sumowania od poczatku: 5.5318772971850983e-05
2. Dla sumowania od końca: 5.5318772971880474e-05
3. Dla sumowania od poczatku z elementem: 5.531877297183153e-05
4. Dla sumowania od końca z elementem: 5.53187729718537e-05

Patrzac na te wyniki widzimy, że wartości błedu sa bardzo do siebie zbliżone, aczkolwiek sumowanie od końca rzeczywiście daje delikatnie lepsze wyniki, dzieki czemu te hipoteze można potwierdzić.

2.2 Hipoteza 2: Używajac rozwiniecia wokół 0 (szereg MacLaurina), przy tej samej liczbie składników szeregu dokładniejsze wyniki uzyskujemy przy małych argumentach

Aby sprawdzić te hipoteze możemy zrobić szybki test wywołujac funkcje:

```
double err = Math.exp(Math.PI) - taylor_exp(Math.PI, 20);
double err2 = Math.exp(Math.PI * 2) - taylor_exp(Math.PI * 2, 20);
System.out.println(err);
System.out.println(err2);
```

Nastepnie dostaniemy wyniki:

1. Dla err: 6.283649156557658E-10
2. Dla err2: 0.001572878553361079

Obserwujac te wyniki, mniejszy bład wychodzi przy mniejszym argumencie, co oznacza, że hipoteze potwierdzamy.

2.3 Hipoteza 3: Sumowanie elementów obliczanych na podstawie poprzedniego daje dokładniejsze wyniki niż obliczanych bezpośrednio ze wzoru

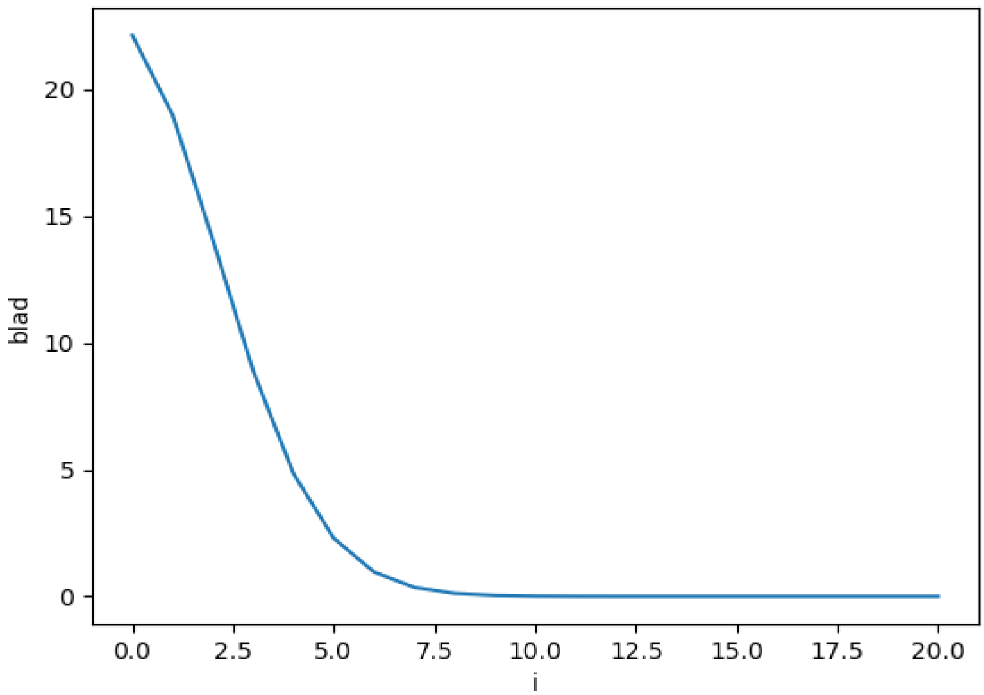
Patrzac na błedy, które powychodziły można stwierdzić iż ta hipoteza została obalona, gdyż najmniejszy bład wyszedł w zwykłym sumowaniu od końca, a nastepnie dopiero z sumowaniem i obliczaniem kolejnego elementu na podstawie poprzedniego.

3 Pytania

3.1 Jak zależy dokładność obliczeń (Bład) od liczby sumowanych składników?

Zbudujemy do tego sobie funkcje:

```
FileWriter fw2 = new FileWriter("exp_function_dokl.csv");
for (int i = 0; i <= 20; i += 1) {
      double error = Math.exp(Math.PI) - taylor_exp(Math.PI, i);
    fw2.write(String.format(Locale.US, "%d;%.15f%n", i, error));
}
fw2.close();
```



3.1 Wyniki z poprzedniej funkcji ukazane na wykresie

Na tej podstawie widzimy, że im wiecej sumowanych składników tym mniejszy błąd czyli tym dokładniejsze obliczenia.