Analiza numeryczna (L)
21.10.2020 r.

- Informacje
- Powtórka
  - reprezentacja zmiennopozycyjna liczb → l. rzeczywiste → l. całkowite; $x = s \cdot m \cdot 2^c$, $m \in [\frac{1}{2}, 1)$, $c \in \mathbb{Z}$ → IEEE 754
  - błędy względne i bezwzględne
  - X_{fl}, zmiennopozycyjna realizacja $+,-,*,/$

$$fl(a \circ b) = (a \circ b)(1+\varepsilon), \quad |\varepsilon| \le 2^{-t}$$

$\circ \in \{+,-,*,/\}$, $a, b \in X_{fl}$ $(a = rd(a),\ b = rd(b))$

zaokrąglenie mantysy; t – liczba bitów przeznaczona na zapamiętanie mantysy

  - utrata cyfr znaczących
  - twierdzenie o kumulacji błędów

Wykład 2. Arytmetyka zmiennopozycyjna i podstawy teorii błędów – część 2.

Uwarunkowanie zadania

Przykład 1

Rozważmy wielomian w postaci

$$w(x) = (x-1)(x-2)(x-3)\dots(x-20) = \sum_{i=0}^{20} a_i x^i, \quad a_{20} = 1,\ a_{19} = -210$$

Wprowadźmy wielomian V_ε zadany wzorem

Założenie: ε – małe

$$V_\varepsilon(x) := w(x) - \varepsilon \cdot x^{19} = \sum_{i=0}^{20} b_i x^i \Rightarrow b_i = a_i \text{ (dla } i = 1,2,\dots,18 \text{ oraz dla } i = 20),\quad \boxed{b_{19} = a_{19} - \varepsilon}$$

Wniosek ($\varepsilon = 2^{-30}$)

MAŁA zmiana danych (tu: zmiana jednego współczynnika wielomianu) ⇒ DUŻĄ zmianę wyniku (tu: zupełnie inny wykres i zbiór pierwiastków)

Przykład 2

$$H_n := \left[\frac{1}{i+j-1}\right] \in \mathbb{R}^{n \times n} \leftarrow \text{macierz Hilberta}$$

$$\hat{H}_n := \left[rd\left(\frac{1}{i+j-1}\right)\right] \in \mathbb{R}^{n \times n} \leftarrow \text{wprowadzenie macierzy Hilberta do pamięci komputera}$$

Obliczmy wyznaczniki

$\det(H_n)$, $\det(\hat{H}_n)$ → całkiem różne wyniki!

(macierze bardzo podobne)

Wniosek

MAŁA zmiana danych (tu: elementy macierzy) ⇒ DUŻA zmiana wyniku (tu: wartość wyznacznika)

Przykład 3

Rozważmy układ równań liniowych postaci:

$$H_n \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = b_n, \quad \text{gdzie} \quad b_n := H_n \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbb{R}^n$$

Pytanie: Jakie jest rozwiązanie tego układu?

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

W praktyce chcąc rozwiązać ten układ przy pomocy komputera rozwiązujemy inne zadanie:

$$\hat{H}_n \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \hat{b}_n, \quad \text{gdzie} \quad \hat{b}_n = rd(b_n)$$

Wyniki, tj. rozwiązania obu układów równań, są zupełnie różne (np. dla $n = 15$)

Wniosek

MAŁA zmiana danych (tu: elementy macierzy Hilberta i wektora prawej strony) ⇒ DUŻA zmiana wyniku (tu: rozwiązanie układu równań)

Intuicja: **Efekt motyla!**

Definicja (uwarunkowanie zadania)

Zadanie nazywamy źle uwarunkowanym jeśli ==mała względna zmiana danych== powoduje ==DUŻĄ względną zmianę wyniku==

Uwagi

1° Przy rozwiązywaniu zadań źle uwarunkowanych przy pomocy komputera z użyciem arytmetyki zmiennopozycyjnej należy zachować szczególną ostrożność. !

2° Jak badać uwarunkowanie zadania? ⇓ Nie jest to łatwe

Pokażmy na dwóch jak sprawdzać uwarunkowanie zadania.

Przykład 1

Zbadajmy uwarunkowanie zadania obliczania wartości ustalonej funkcji f w punkcie x:

h – małe

$$\left|\frac{f(x+h) - f(x)}{f(x)}\right| = \left|\frac{f(x+h) - f(x)}{h}\right| \left|\frac{h}{f(x)}\right| \approx |f'(x)| \left|\frac{h}{f(x)}\right| = \left|\frac{f'(x)\,x}{f(x)}\right| \left|\frac{h}{x}\right|$$

↑ względna zmiana wyniku przy zmianie argumentu ($x+h$ zamiast x)

$$\left|\frac{x - (x+h)}{x}\right| = \left|\frac{h}{x}\right| \leftarrow \text{względna zmiana argumentu}$$

Ta wielkość mówi nam jak względna zmiana wyniku ma się do względnej (małej) zmiany danych
⇓
Tego typu wielkości nazywamy wskaźnikami uwarunkowania

Przykład 2

Rozważmy problem uwarunkowania zadania obliczania iloczynu skalarnego dwóch wektorów

$$a := \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}, \quad b := \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \in \mathbb{R}^n \leftarrow \text{dane}$$

iloczyn skalarny

$$S(a,b) := \sum_{i=1}^{n} a_i b_i \leftarrow \text{wynik}$$

Zaburzenie danych:

$$\tilde{a} := \begin{bmatrix} a_1(1+\varepsilon_1) \\ a_2(1+\varepsilon_2) \\ \vdots \\ a_n(1+\varepsilon_n) \end{bmatrix}, \quad \tilde{b} := \begin{bmatrix} b_1(1+\delta_1) \\ b_2(1+\delta_2) \\ \vdots \\ b_n(1+\delta_n) \end{bmatrix}$$

$S(\tilde{a}, \tilde{b}) \leftarrow$ wynik dla zaburzonych danych.

Obliczmy błąd względny wyniku: (ε_i, δ_i – małe)

$$\left|\frac{S(a,b) - S(\tilde{a},\tilde{b})}{S(a,b)}\right| = \left|\frac{\sum_i a_i b_i - \sum_i a_i b_i (1+\varepsilon_i)(1+\delta_i)}{\sum_i a_i b_i}\right| \approx$$

$(1+\varepsilon_i)(1+\delta_i) = 1 + \varepsilon_i + \delta_i + \varepsilon_i\delta_i$ ← $\varepsilon_i\delta_i$ to jest zaniedbywalne

$$\approx \left|\frac{\sum_i a_i b_i - \sum_i a_i b_i - \sum_i (\varepsilon_i + \delta_i) a_i b_i}{\sum_i a_i b_i}\right| \le \max_i |\varepsilon_i + \delta_i| \cdot \frac{\sum_i |a_i b_i|}{\left|\sum_i a_i b_i\right|}$$

$\max_i |\varepsilon_i + \delta_i|$ – sumaryczny maksymalny względny błąd danych

$\frac{\sum_i |a_i b_i|}{|\sum_i a_i b_i|} = K(a,b)$ – wskaźnik uwarunkowania tego zadania

Wnioski

1° Jeśli $a_i b_i > 0$ (lub $a_i b_i < 0$) dla $i = 1,2,\dots,n$, to $K(a,b) = 1$ i zadanie jest dobrze uwarunkowane.

2° $K(a,b)$ może być jednak dowolnie duże. Będzie tak, np. $\sum |a_i b_i| = 1$, a $\sum a_i b_i \approx 0$ (wtedy $K(a,b) \to \infty$)

Algorytmy numerycznie poprawne

Definicja (algorytm numerycznie poprawny)

Algorytm nazywamy numerycznie poprawnym jeśli wynik jego działania w arytmetyce zmiennopozycyjnej <u>może być</u> zinterpretowany jako mało zaburzony wynik dokładny dla mało zaburzonych danych.

Przykład

Chcemy obliczyć wartość funkcji dwóch zmiennych $A(a,b)$.
Niech $Alg_A(a,b)$ oznacza algorytm obliczania wartości $A(a,b)$.

$$fl(Alg_A(a,b)) = A(a(1+\alpha), b(1+\beta))(1+\gamma)$$

$|\gamma|, |\alpha|, |\beta| \lesssim K \cdot 2^{-t}$, K – niewielka stała

$A(a(1+\alpha), b(1+\beta))$ – wynik dokładny dla mało zaburzonych danych; całość – mało zaburzony wynik dokładny dla mało zaburzonych danych

Uwagi podsumowujące

1° ZADANIE DOBRZE UWARUNKOWANE + ALGORYTM NUMERYCZNIE POPRAWNY = ==SUKCES==

2° ZADANIE ŹLE UWARUNKOWANE + ALGORYTM NUMERYCZNIE POPRAWNY = ==PRAWIE NA PEWNO PORAŻKA==

3° Uwarunkowanie zadania to jego cecha, której NIC NIE ZMIENI (żaden algorytm)!

Przykład

Niech dany będzie następujący algorytm sumowania liczb $x_1, x_2, \dots, x_n$:

```
S := x1
FOR i=2 TO n
   [S := S + xi
RETURN(S)
```

Pytanie: Czy jest to algorytm numerycznie poprawny?

Założenie: $x_i = rd(x_i)$ $(x_i \in X_{fl})$

$$fl(S) = \Big(\dots\big(((x_1+x_2)(1+\varepsilon_2) + x_3)(1+\varepsilon_3) + x_4\big)(1+\varepsilon_4) + \dots + x_n\Big)(1+\varepsilon_n) = \quad |\varepsilon_i| \le 2^{-t},\ \varepsilon_1 := 0$$

$$= \text{(Do domu ....)} = \sum_{i=1}^{n} x_i \prod_{j=i}^{n} (1+\varepsilon_j) =$$

Twierdzenie o kumulacji błędów:

$$1 + E_i := \prod_{j=i}^{n} (1+\varepsilon_j) \Rightarrow \begin{cases} |E_i| \lesssim (n-i+1) \cdot 2^{-t} & (i = 2,3,\dots,n) \\ |E_1| \lesssim (n-1) \cdot 2^{-t} \end{cases}$$

$$= \sum_{i=1}^{n} \underbrace{x_i(1+E_i)}_{\hat{x}_i} = \hat{x}_1 + \hat{x}_2 + \dots + \hat{x}_n$$

wynik dokładny dla (innych) nieco zaburzonych danych (przy rozsądnym założeniu, tj. dla „rozsądnych” wartości n.

Konkluzja

Ten algorytm jest numerycznie poprawny