

Metody numeryczne

<https://www.mimuw.edu.pl/~przykry/mn>

Piotr Krzyżanowski
2020/01/23,15:46:56

Algebra liniowa

- układy równań liniowych
- zadanie najmniejszych kwadratów
- zagadnienie własne

Spis treści

Układy równań liniowych
Arytmetyka zmiennopozycyjna
Wrażliwość zadania i numeryczna poprawność algorytmu
Liniowe zadanie najmniejszych kwadratów
Metody iteracyjne dla układów równań liniowych
Zagadnienie własne
Interpolacja wielomianowa
Splajny
Aproksymacja średniokwadratowa
Aproksymacja jednostajna
Równania nieliniowe
Całkowanie
Szybka transformacja Fouriera (FFT)

1

Układy równań liniowych

Układy równań liniowych

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1N}x_N = b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2N}x_N = b_2 \\ \vdots \\ a_{N1}x_1 + a_{N2}x_2 + \dots + a_{NN}x_N = b_N \end{cases}$$

Macierzowo możemy zapisać ten układ w zwartej postaci

$$Ax = b,$$

gdzie

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \vdots & \vdots & & \vdots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix}.$$

3

Tak, to GAL!

$$Ax = b$$

Układ ma dokładnie jedno rozwiązanie \iff spełniony jest którykolwiek z warunków:

- $\det(A) \neq 0$
- jedynym rozwiązaniem $Ax = 0$ jest $x = 0$
- $\ker(A) = \{0\}$
- A nie ma zerowej wartości własnej
- A jest nieosobliwa

4

Praca domowa

Powtórzyć sobie podstawowe wiadomości z GALu:

- macierze i wektory
- mnożenie macierzy przez macierz i przez wektor
- macierz odwrotna
- wartości własne i wektory własne macierzy
- rząd macierzy, macierz pełnego rzędu
- ortogonalność wektorów

Łatwe układy równań: z macierzą diagonalną

$$\begin{cases} a_{11}x_1 & & & = b_1 \\ & a_{22}x_2 & & = b_2 \\ & & \vdots & \\ & & & a_{NN}x_N = b_N \end{cases}$$

Trywialne!...

$$x_i = b_i / a_{ii}, \quad i = 1, \dots, N,$$

Koszt: $\mathcal{O}(N)$ flopów.

Koszt algorytmów numerycznych

- obliczeniowy
- pamięciowy

7

Koszt obliczeniowy algorytmów numerycznych

Tradycyjnie, jest to liczba wykonanych operacji arytmetycznych:

Czym jest *flop*?

FLOP = **FL**oating point **OP**eration

Tradycyjnie, przyjmujemy, że wszystkie operacje (+, −, ×, ÷) kosztują tyle samo: 1 flop.

Tradycyjnie, nie uwzględniamy kosztu obsługi pętli i innych instrukcji sterujących, ani kosztu dostępu do pamięci.

Zwykle przyjmujemy, że funkcje elementarne ($\sqrt{}$, sin, ...) kosztują 1 flop.

8

Koszt obliczeniowy algorytmów numerycznych

Dzisiaj żadne z tych założeń nie jest prawdziwe na typowym komputerze!

Niemniej w zdecydowanej większości przypadków ten model daje dobry wgląd w *jakościowe* zachowanie się algorytmu w rzeczywistości.

9

Łatwe układy równań: z macierzą trójkątną

$$\begin{cases} a_{11}x_1 & = b_1 \\ a_{21}x_1 + a_{22}x_2 & = b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 & = b_3 \\ \vdots & \\ a_{N1}x_1 + a_{N2}x_2 + a_{N3}x_3 + \dots + a_{NN}x_N & = b_N \end{cases}$$

Macierz układu jest *dolna trójkątna*

$$A = \begin{bmatrix} a_{11} & & & \\ a_{21} & a_{22} & & \\ \vdots & & \ddots & \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{bmatrix}$$

10

Łatwe układy równań: z macierzą trójkątną

Rozwiązanie metodą podstawienia w przód:

$$\begin{aligned}
 a_{11}x_1 &= b_1 \implies x_1 = \frac{1}{a_{11}}b_1 \\
 a_{21}x_1 + a_{22}x_2 &= b_2 \implies x_2 = \frac{1}{a_{22}}(b_2 - a_{12}x_1) \\
 a_{21}x_1 + a_{22}x_2 + a_{33}x_3 &= b_3 \implies x_3 = \frac{1}{a_{33}}(b_3 - a_{13}x_1 - a_{23}x_2) \\
 &\vdots \\
 a_{k1}x_1 + a_{k2}x_2 + a_{k3}x_3 + \dots + a_{kk}x_k &= b_k \implies x_k = \frac{1}{a_{kk}}(b_k - a_{1k}x_1 - a_{2k}x_2 - \dots - a_{k-1,k}x_{k-1}) \\
 &\vdots
 \end{aligned}$$

Łatwe układy równań: z macierzą trójkątną

Analogicznie, kosztem $\mathcal{O}(N^2)$ flopów rozwiązujemy układ z macierzą górną trójkątną:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ & a_{22} & \dots & a_{2N} \\ & & \ddots & \vdots \\ & & & a_{NN} \end{bmatrix},$$

metodą podstawienia w tył.

Łatwe układy równań: z macierzą trójkątną

Algorytm 1 Podstawienie w przód. Rozwiązanie układu $Ax = b$ z macierzą dolną trójkątną

```

 $x_1 = b_1 / a_{11}$ 
for  $i = 2 : N$ 
     $x_i = \frac{1}{a_{ii}}(b_i - \sum_{j=1}^{i-1} a_{ij}x_j)$ 
end
return  $(x_1, \dots, x_N)$ 

```

Koszt tego algorytmu jest rzędu $\mathcal{O}(N^2)$ flopów.

Łatwe układy równań: z macierzą ortogonalną

Definicja (Macierz ortogonalna)

Macierz $Q \in \mathbb{R}^{N \times N}$ nazywamy ortogonalną, jeśli

$$Q^T Q = I.$$

Inaczej: kolumny macierzy Q są wzajemnie ortogonalne i ich norma euklidesowa jest równa 1.

Wniosek

$$Q^{-1} = Q^T,$$

zatem rozwiązaniem układu

$$Qx = b$$

jest

$$x = Q^{-1}b.$$

Koszt: $\mathcal{O}(N^2)$ flopów.

A co zrobić w przypadku ogólnym?

Jak rozwiązać

$$Ax = b,$$

gdy A — macierz nieosobliwa.

Zły pomysł

Wzory Cramera ???

Koszt zaporowy: obliczenie $\det(A)$ z rozwinięcia Laplace'a kosztuje przynajmniej $N!$ flopów.

Są też inne powody, by tego nie robić.

15

Kącik porad

Jak czytać wzory?

Napisy typu $A^{-1}Y$ należy czytać jako „rozwiąż równanie $AX = Y$ ”

Wujek Dobra Rada:

Warto sprowadzać zadania do takich, które już umiemy (tanio) rozwiązać!

17

A co zrobić w przypadku ogólnym?

Jak rozwiązać

$$Ax = b,$$

gdy A — macierz nieosobliwa.

Kolejny zły pomysł

Obliczyć A^{-1} , a następnie $x = A^{-1}b$???

Pierwsze przykazanie numeryka

Nie będziesz odwracać macierzy swojej nadaremno!

16

A co zrobić w przypadku ogólnym?

Jak rozwiązać

$$Ax = b,$$

gdy A — macierz nieosobliwa.

Gdyby

$$A = XYZ,$$

gdzie X, Y, Z — „łatwe” do rozwiązania:

- diagonalne (koszt: $\mathcal{O}(N)$)
- trójkątne (koszt: $\mathcal{O}(N^2)$)
- ortogonalne (koszt: $\mathcal{O}(N^2)$)

to

$$Ax = b \iff XYZx = b.$$

18

A co zrobić w przypadku ogólnym?

$$Ax = b \iff XYZx = b.$$

$$Ax = b \iff X Y \underbrace{Zx}_{=:y} = b.$$

$$\underbrace{\quad}_{=:z}$$

Algorytm:

1. Znaleźć rozkład $A = XYZ$
2. Rozwiązać układ $Xz = b$ (max koszt $\mathcal{O}(N^2)$)
3. Rozwiązać układ $Yy = z$ (max koszt $\mathcal{O}(N^2)$)
4. Rozwiązać układ $Zx = y$ (max koszt $\mathcal{O}(N^2)$)

Łączny koszt: koszt rozkładu $+\mathcal{O}(N^2)$.

19

Rozkłady macierzy

- rozkład LU: iloczyn macierzy trójkątnych L i U :

$$A = LU \quad \text{lub} \quad PA = LU$$

(P — macierz permutacji)

- rozkład QR: iloczyn macierzy ortogonalnej Q i trójkątnej R :

$$A = QR$$

20

Rozkład LU (bez wyboru)

Na początek rozważamy algorytm bez wyboru elementu głównego (bez osiowania).

$$A = LU$$

Podzielimy macierze na bloki:

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ & U_{22} \end{bmatrix}$$

gdzie bloki $(1, 1)$ są rozmiaru $k \times k$.

- U_{ii} — górne trójkątne
- L_{ii} — dolne trójkątne (z jedynkami na diagonalu \implies jednoznaczność)

21

Rozkład LU bez wyboru — wyprowadzenie

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ & U_{22} \end{bmatrix}$$

Po wymnożeniu

$$\begin{aligned} A_{11} &= L_{11}U_{11} & A_{12} &= L_{11}U_{12} \\ A_{21} &= L_{21}U_{11} & A_{22} &= L_{21}U_{12} + L_{22}U_{22} \end{aligned}$$

skąd dostajemy algorytm rekurencyjny:

1. Wyznacz (rekurencyjnie) rozkład $A_{11} = L_{11}U_{11}$
2. $U_{12} = L_{11}^{-1}A_{12}$ (tzn. rozwiąż układ z macierzą trójkątną)
3. $L_{21} = A_{21}U_{11}^{-1}$ (tzn. rozwiąż układ z macierzą trójkątną)
4. Aktualizuj $\tilde{A}_{22} = A_{22} - L_{21}U_{12}$
5. Wyznacz (rekurencyjnie) rozkład $\tilde{A}_{22} = L_{22}U_{22}$

22

Dygresja: blokowy rozkład LU

Jeśli A_{11} — nieosobliwa, to

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} I & \\ L_{21} & I \end{bmatrix} \begin{bmatrix} A_{11} & \\ & S \end{bmatrix} \begin{bmatrix} I & U_{12} \\ & I \end{bmatrix}$$

gdzie

$$S = A_{22} - A_{21}A_{11}^{-1}A_{12}$$

— uzupełnienie Schura oraz

$$L_{21} = A_{21}A_{11}^{-1}, \quad U_{12} = A_{11}^{-1}A_{12}.$$

23

Klasyczny algorytm rozkładu LU: $k = 1$

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ & U_{22} \end{bmatrix}$$

1. Wyznacz (rekurencyjnie) rozkład $A_{11} = L_{11}U_{11}$
2. $U_{12} = L_{11}^{-1}A_{12}$ (tzn. rozwiąż układ z macierzą trójkątną)
3. $L_{21} = A_{21}U_{11}^{-1}$ (tzn. rozwiąż układ z macierzą trójkątną)
4. Aktualizuj $\tilde{A}_{22} = A_{22} - L_{21}U_{12}$
5. Wyznacz (rekurencyjnie) rozkład $\tilde{A}_{22} = L_{22}U_{22}$

$k = 1 \implies A_{11} = a_{11}$ (liczba) $\implies l_{11} = 1 \implies$ pierwsze wywołanie rekurencji jest zbędne!

25

Koszt rozkładu LU

Dwa interesujące przypadki:

- $k = 1$, tzn. macierz A_{11} jest skalar
- $k = \lceil N/2 \rceil$, tzn. dzielimy A na ćwiartki

Inny interesujący przypadek:

- Dzielimy A na $B \times B$ bloków „równego” rozmiaru $\approx N/B$ (wcześniejszy przypadek to $B = 2$)
- Nie będziemy go tu rozważać.

24

Klasyczny algorytm rozkładu LU (bez wyboru)

$$\begin{bmatrix} a_{11} & a_{12}^T \\ a_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} 1 & \\ l_{21} & L_{22} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12}^T \\ & U_{22} \end{bmatrix}$$

1. $u_{11} = a_{11}$
2. $u_{12} = a_{12}$
3. $l_{21} = a_{21}u_{11}^{-1}$ (mnożenie wektora a_{21} przez liczbę u_{11}^{-1})
4. Aktualizuj $A_{22} = A_{22} - l_{21}u_{12}^T$ (tzn. o macierz rzędu 1)
5. Wyznacz rozkład $A_{22} = L_{22}U_{22}$ — to już nie musi być rekurencja: wystarczy pętla!

Rozkład możemy wykonać *w miejscu*, zapisując U w górnym trójkącie A , a L (bez jedynek) — w dolnym trójkącie A .

26

Klasyczny algorytm rozkładu LU (bez wyboru): implementacja

Rozkład wykonujemy w miejscu, nadpisując A czynnikami L i U :

```
for  $k = 1 : N - 1$ 
  for  $i = k + 1 : N$ 
     $a_{ik} = a_{ik} / a_{kk}$  (wyznaczenie  $k$ -tej kolumny  $L$ )
  end
  for  $i = k + 1 : N$ 
    for  $j = k + 1 : N$ 
       $a_{ij} = a_{ij} - a_{ik} a_{kj}$  (aktualizacja  $\tilde{A}_{22}$ )
    end
  end
end
```

To nic innego jak pocziwa eliminacja Gaussa! Stąd inna nazwa: GE.

27

Klasyczny algorytm rozkładu LU (bez wyboru): ograniczenia

Algorytm GE nie zawsze zadziała, np.:

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

W pierwszym kroku dzielimy przez $a_{11} = 0 \dots$

Twierdzenie

Jeśli wszystkie minory główne macierzy A są niezerowe, to GE jest wykonalna, tzn. na diagonalu macierzy U nie pojawiają się zera.

Jak uniknąć dzielenia przez zero? Przez wybór elementu głównego (osiowanie).

28

Klasyczny algorytm rozkładu LU (z częściowym wyborem)

Rozkład wykonujemy w miejscu, nadpisując A czynnikami L i U :

```
for  $k = 1 : N - 1$ 
  Znajdź wiersz  $p \geq k$  t.ż.  $|a_{pk}| = \max\{|a_{ik}| : k \leq i \leq N\}$ 
  Zamień wiersze  $p$  i  $k$ 
  for  $i = k + 1 : N$ 
     $a_{ik} = a_{ik} / a_{kk}$  (wyznaczenie  $k$ -tej kolumny  $L$ )
  end
  for  $i = k + 1 : N$ 
    for  $j = k + 1 : N$ 
       $a_{ij} = a_{ij} - a_{ik} a_{kj}$  (aktualizacja  $\tilde{A}_{22}$ )
    end
  end
end
```

GEPP — GE with Partial Pivoting

29

Klasyczny algorytm rozkładu LU (z częściowym wyborem)

- Zawsze wykonalny (w arytmetyce dokładnej), gdy A jest nieosobliwa
- Daje się zinterpretować jako

$$PA = LU,$$

gdzie P — macierz permutacji

- Istnieją inne strategie
 - częściowy wybór w wierszu: $AP = LU$
 - wybór pełny (w całej $\tilde{A}_{22}^{(k)}$): $P_1 A P_2 = LU$

30

Klasyczny algorytm rozkładu LU: koszt

$$\begin{bmatrix} a_{11} & a_{12}^T \\ a_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} 1 & \\ l_{21} & L_{22} \end{bmatrix} \begin{bmatrix} a_{11} & a_{12}^T \\ & U_{22} \end{bmatrix}$$

1. $l_{21} = a_{21}u_{11}^{-1}$ (mnożenie wektora a_{21} przez liczbę u_{11}^{-1}) **Koszt:** $\mathcal{O}(N)$
2. Aktualizuj $A_{22} = A_{22} - l_{21}u_{12}^T$ (tzn. o macierz rzędu 1) **Koszt:** $\mathcal{O}(N^2)$
3. Wyznacz rozkład $A_{22} = L_{22}U_{22}$

Zatem koszt w zależności od N :

$$T(N) = \mathcal{O}(N) + \mathcal{O}(N^2) + T(N-1) = \dots = \mathcal{O}(N^3).$$

31

Koszt rozkładu LU

Koszt klasycznego rozkładu LU = $\mathcal{O}(N^3)$.

Czy można lepiej?

Rozważmy drugi przypadek, $k = \lceil N/2 \rceil$, czyli podział A na ćwiartki.

32

Rekurencyjny algorytm rozkładu LU

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ & U_{22} \end{bmatrix}$$

Macierze A_{11} i A_{22} (prawie) takiego samego rozmiaru $\approx N/2$.

1. Wyznacz (rekurencyjnie) rozkład $A_{11} = L_{11}U_{11}$
2. $U_{12} = L_{11}^{-1}A_{12}$ (tzn. rozwiąż układ z macierzą trójkątną)
3. $L_{21} = A_{21}U_{11}^{-1}$ (tzn. rozwiąż układ z macierzą trójkątną)
4. Aktualizuj $\tilde{A}_{22} = A_{22} - L_{21}U_{12}$
5. Wyznacz (rekurencyjnie) rozkład $\tilde{A}_{22} = L_{22}U_{22}$

$$\text{Koszt: } T(N) = 2T(N/2) + \underbrace{\mathcal{O}(N^\omega)}_{\text{koszt GEMM}}$$

$$\text{Wiadomo, że } 2 \leq \omega \leq 3 \implies T(N) = \mathcal{O}(N^\omega)$$

Otwarty problem

Ile dokładnie wynosi minimalna wartość ω ?

33

Koszt GEMM

GEMM = **GE**neral **M**atrix **M**ultiply

- Mnożenie macierzy przez macierz $X = A \cdot B$ tak trudne jak rozkład $A = LU$ (i tak trudne jak $Ax = b$)
- Jak wyznaczyć $A \cdot B$? Dziel i (mądrze) rządź!

$$\begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \cdot \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Naiwnie,

$$X_{11} = A_{11}B_{11} + A_{12}B_{21}$$

$$X_{12} = A_{11}B_{12} + A_{12}B_{22}$$

$$X_{21} = A_{21}B_{11} + A_{22}B_{21}$$

$$X_{22} = A_{21}B_{12} + A_{22}B_{22}.$$

34

daje algorytm rekurencyjny, ale... Koszt nadal $\mathcal{O}(N^3)$. Strassen:

- Strassen: $\mathcal{O}(N^{2.808})$
 - Coppersmith i Winograd: $\mathcal{O}(N^{2.38})$
 - Vassilevska-Williams, $\mathcal{O}(N^{2.373})$.
- Wariant metody Coppersmitha i Winograda, 2014.

To nie koniec tej historii...

W rzeczywistości liczy się nie tylko liczba flopów, ale także koszt dostępu do pamięci: intensywne wykorzystanie BLAS Level 3 (tutaj: GEMM) podnosi szybkość działania algorytmu.

35

Arytmetyka zmiennopozycyjna

$$Ax = b$$

1. GEPP — kosztem $\mathcal{O}(N^3)$ — daje nam rozkład:

$$PA = LU$$

Zatem

$$PAx = LUx = Pb$$

i stąd dalszy ciąg algorytmu:

2. $\tilde{b} = Pb$ (permutacja wektora b)
3. Rozwiąż $Ly = \tilde{b}$ (koszt $\mathcal{O}(N^2)$)
4. Rozwiąż $Ux = y$ (koszt $\mathcal{O}(N^2)$)

Koszt całkowity: $\mathcal{O}(N^3)$ (zdominowany przez rozkład $PA = LU$)

36

Arytmetyka zmiennopozycyjna

- liczby maszynowe
- standard IEEE 754
- działania w arytmetyce zmiennopozycyjnej
- ku przestrodze: historie prawdziwe

37

Zapis inżynierski liczb rzeczywistych

$$6.63 \cdot 10^{-34}$$

6.63 → „mantysa” (odpowiada za precyzję)

−34 → cecha (odpowiada za wielkość)

10 → podstawa (jaki system liczbowy wybieramy)

38

Liczby maszynowe

Standardowo $\beta = 2$.

Możemy przeznaczyć jedynie skończoną liczbę bitów na zapis m i e .

$$x = (-1)^s \cdot m \cdot 2^e$$

$m = (f_0.f_1f_2 \dots f_{p-1})_2$, $f_i \in \{0, 1\}$ cyfry binarne (razem p bitów)

e integer t. że

$$(1 - e_{\max}) \leq e \leq e_{\max}$$

$\beta = 2$

s (1 bit)

40

Zapis liczby rzeczywistej

$$x = (-1)^s \cdot m \cdot \beta^e$$

m → mantysa (odpowiada za precyzję), $0 \leq m < \beta$

e → cecha (odpowiada za wielkość), $e \in \mathbb{Z}$

β → podstawa (jaki system liczbowy wybieramy)

s → znak, $s \in \{0, 1\}$

39

Liczby maszynowe (znormalizowane) w pamięci komputera

Normalizacja: $f_0 = 1$, czyli $1 \leq m < 2$.

$$x = (-1)^s \cdot (1.f_1f_2 \dots f_{p-1})_2 \cdot 2^e$$

przechowujemy jako

s	$e + e_{\max}$	f_1	f_2	\dots	f_{p-1}
-----	----------------	-------	-------	---------	-----------

Dzięki temu m.in. $e + e_{\max} \geq 0$.

41

Przykład: 5-bitowa arytmetyka zmiennopozycyjna i

$$\beta = 2, \quad p = 3, \quad e_{\max} = 2.$$

Liczby znormalizowane:

$$x = (-1)^s \cdot (1.f_1 f_2)_2 \cdot 2^e, \quad e \in \{-1, 0, 1, 2\}.$$

Możliwe wartości mantysy to

$$(1.00)_2 = 1, \quad (1.01)_2 = 1.25, \quad (1.10)_2 = 1.5, \quad (1.11)_2 = 1.75.$$

Oto wszystkie (dodatnie, znormalizowane) liczby maszynowe:

$$0.500, 0.625, 0.750, 0.875, \quad 1.000, 1.250, 1.500, 1.750, \\ 2.000, 2.500, 3.000, 3.500, \quad 4.000, 5.000, 6.000, 7.000.$$

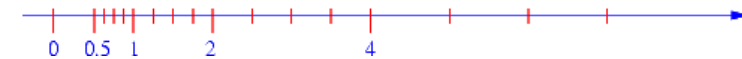
42

Liczby maszynowe w/g standardu IEEE 754

Nazwa	binary16	binary32	binary64	binary128	
„Precyzja”	poł.	poj.	podw.	pocz.	rozs.
Język C		float	double		long double
Rozmiar	16	32	64	128	80
p	11	24	53	113	64
e_{\max}	15	127	1023	16383	16383
$\nu = 2^{-p}$	$4.9 \cdot 10^{-4}$	$6.0 \cdot 10^{-8}$	$1.1 \cdot 10^{-16}$	$9.6 \cdot 10^{-35}$	$5.4 \cdot 10^{-20}$
realmax	$6.6 \cdot 10^4$	$3.4 \cdot 10^{38}$	$1.8 \cdot 10^{308}$	$1.2 \cdot 10^{4932}$	
realmin	$6.1 \cdot 10^{-5}$	$1.2 \cdot 10^{-38}$	$2.2 \cdot 10^{-308}$	$3.4 \cdot 10^{-4932}$	
subnorm	$6.0 \cdot 10^{-8}$	$1.4 \cdot 10^{-45}$	$4.9 \cdot 10^{-324}$	$6.7 \cdot 10^{-4966}$	

44

Przykład: 5-bitowa arytmetyka zmiennopozycyjna ii



Liczby subnormalne:

$$x = (-1)^s \cdot (0.f_1 f_2)_2 \cdot 2^{-1}.$$

Oto wszystkie (dodatnie) subnormalne liczby maszynowe:

$$0.125, 0.25, 0.375.$$

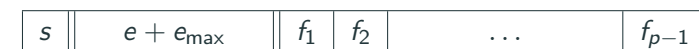


43

Liczby maszynowe w/g standardu IEEE 754

Liczby „zwyczajne”, znormalizowane

$$x = (-1)^s \cdot (1.f_1 f_2 \dots f_{p-1})_2 \cdot 2^e:$$



Liczby o specjalnym znaczeniu:

- (plus i minus) zero $x = (-1)^s \cdot 0$



- (plus i minus) nieskończoność $x = (-1)^s \cdot \infty$



- not a number $x = \text{NaN}$



- liczby subnormalne $x = (-1)^s \cdot \underbrace{(0.f_1 f_2 \dots f_{p-1})_2}_{\neq 0} \cdot 2^{1-e_{\max}}$



45

Reprezentacja liczb rzeczywistych

Niech $x \in \mathbb{R}$. Jeśli x nie jest liczbą maszynową, to jak ją reprezentować?

$fl(x)$ — reprezentacja x w arytmetyce zmiennopozycyjnej

IEEE 754 gwarantuje 4 możliwe tryby zaokrąglania x do liczby maszynowej:

RN do najbliższej (domyślnie)

RD w dół, tzn. w stronę $-\infty$

RU w górę, tzn. w stronę $+\infty$

RZ w stronę zera, tzn.

$$RZ(x) = \begin{cases} RD(x) & \text{gdy } x \geq 0 \\ RU(x) & \text{gdy } x \leq 0 \end{cases}$$

46

Precyzja arytmetyki

Jeśli $|x| \in [\text{realmin}, \text{realmax}]$, to

$$\frac{|x - RN(x)|}{|x|} \leq \frac{1}{2^p} =: \nu \quad \leftarrow \text{precyzja arytmetyki}$$

Czyli wtedy błąd względny reprezentacji x przez $RN(x)$ nigdy nie przekracza ν .

Dalej zawsze przyjmujemy $fl(x) = RN(x)$.

$$fl(x) = x \cdot (1 + \epsilon), \quad |\epsilon| \leq \nu.$$

47

Działania arytmetyczne: $\square \in \{+, -, \times, \div, \sqrt{}, FMA\}$

Jeśli a, b są liczbami maszynowymi, to $a \square b$

- może przekroczyć zakres (*overflow*):

$$fl(a \square b) = \mathbf{Inf}$$

- może być zbyt bliski zera (*underflow*):

$$fl(a \square b) = 0$$

- może nie mieć sensu:

$$fl(a \square b) = \mathbf{NaN}$$

- w przeciwnym przypadku,

$$fl(a \square b) = fl(a \square b)$$

Zatem poza skrajnościami,

$$fl(a \square b) = (a \square b) \cdot (1 + \epsilon), \quad |\epsilon| \leq \nu.$$

48

Przykłady

Zakładamy działania w podwójnej precyzji.

- overflow: $10^{308} \times 100 = \mathbf{Inf} = 1/0$
- underflow: $10^{-308} \div 10^{28} = 0$
- gradual underflow: $10^{-308} \div 100 \approx 10^{-310}$ (l. subnormalna)
- $0/0 = \mathbf{NaN}$
- $\mathbf{Inf} - \mathbf{Inf} = \mathbf{NaN}$
- przy okazji: $(\mathbf{NaN} == \mathbf{NaN}) = \text{False}$
- $0^0 = ?$

49

Dziwaczności

- działania nie są łączne (w przeciwieństwie do dokładnej arytmetyki)
- nie musi być $1+x == 1 \implies x==0$

Epsilon maszynowy

Odległość ϵ liczby 1 od następnej liczby maszynowej.

Zatem $fl(1 + \epsilon) > 1$, ale $fl(1 + \epsilon/2) = 1$.

- wynikiem $x - x + 1$ może być 1 ale też może być 0
- ryzykowne jest sprawdzanie $x == y$
- uwaga na optymalizujące kompilatory i obliczenia równoległe!

50

Redukcja cyfr przy odejmowaniu i

Wynik nawet jednego działania może być obarczony katastrofalnym błędem:

Dane: $x, y \in \mathbb{R}$. Obliczyć: $s = x + y$.

Zamiast x, y dysponujemy ich reprezentacjami:

$$\tilde{x} = fl(x), \quad \tilde{y} = fl(y).$$

Zatem obliczona wartość wyniku, \tilde{s} , spełnia $(|\delta|, |\epsilon_x|, |\epsilon_y| \leq \nu)$

$$\begin{aligned} \tilde{s} &= fl(\tilde{x} + \tilde{y}) = (\tilde{x} + \tilde{y}) \cdot (1 + \delta) \\ &= (x \cdot (1 + \epsilon_x) + y \cdot (1 + \epsilon_y)) \cdot (1 + \delta) \\ &\approx \underbrace{x + y}_{=s} + (\epsilon_x + \delta) \cdot x + (\epsilon_y + \delta) \cdot y \end{aligned}$$

52

Przesąd

Fałsz!

Najważniejszym problemem w obliczeniach zmiennopozycyjnych jest kumulacja błędów zaokrągleń.

Prawda:

Kumulacja błędów ma znaczenie.
Ale większym problemem może być drastyczne wzmocnienie wcześniejszego (nawet minimalnego) błędu.

51

Redukcja cyfr przy odejmowaniu ii

Stąd błąd względny wyniku

$$\begin{aligned} \frac{|s - \tilde{s}|}{|s|} &= \frac{|(\epsilon_x + \delta) \cdot x + (\epsilon_y + \delta) \cdot y|}{|s|} \\ &\leq \frac{(|\epsilon_x| + |\delta|) \cdot |x| + (|\epsilon_y| + |\delta|) \cdot |y|}{|s|} \leq 2 \frac{|x| + |y|}{|x + y|} \nu. \end{aligned}$$

Zatem

- gdy x, y są tego samego znaku, to

$$\frac{|s - \tilde{s}|}{|s|} \leq 2\nu \quad (\text{wysoka precyzja wyniku!})$$

53

- gdy $x \approx -y$, to

$$\frac{|s - \tilde{s}|}{|s|} \approx \infty$$

(żadna cyfra znacząca wyniku może nie być dokładna!)

54

Poza standardem IEEE 754

bfloat16:

- Używany w dedykowanych procesorach dla uczenia maszynowego, m.in. Google TPU, Intel AI (w planach).
- Coś pośredniego między binary16 a binary32, ale tylko na 16 bitach.

Nazwa	binary16	binary32	bfloat16
Rozmiar	16	32	16
p	11	24	8
e_{\max}	15	127	127
$\nu = 2^{-p}$	$4.9 \cdot 10^{-4}$	$6.0 \cdot 10^{-8}$	$3.9 \cdot 10^{-3}$
realmax	$6.6 \cdot 10^4$	$3.4 \cdot 10^{38}$	$3.4 \cdot 10^{38}$
realmin	$6.1 \cdot 10^{-5}$	$1.2 \cdot 10^{-38}$	$1.2 \cdot 10^{-38}$
subnorm	$6.0 \cdot 10^{-8}$	$1.4 \cdot 10^{-45}$?

56

- Bywa różnie.
- Wiele zależy od producenta konkretnego kompilatora.
- Nawet standardy nie są w pełni zgodne, np. C11 lub C++14 nie w pełni uwzględniają IEEE 754-2008.
- Niektóre opcje optymalizacji kodu wyłączają zgodność z IEEE 754. To jak zabawa z brzytwą.

55

Parę słów o działaniach arytmetycznych (IA-32) i

IA-32, czyli architektura x86-64.

- instrukcje **SSE/AVX**: nowoczesne, zalecane, wektorowe (128/256/512 bitów), binary32/binary64
- instrukcje **x87**: przestarzałe, ale jest też 80-bit i np. funkcje trygonometryczne
- unikać innych wyników niż znormalizowane (np. NaN, Inf, subnormal) — spowolnienie
- są instrukcje zwracające mniej dokładne wyniki \div , $\sqrt{}$ itp., ale znacznie szybsze
(Skylake+AVX: $1/\sqrt{x}$ prawie $8\times$ szybciej, gdy dokładność tylko 11 bitów zamiast 24)

57

Parę słów o działaniach arytmetycznych (IA-32) ii

Działanie	Instrukcja	Latency	Throughput	Latency	Throughput
$x + y$	ADD	4	0.5	4	0.5
$x \cdot y$	MUL	4	0.5	4	0.5
$x \cdot y + z$	FMA	–	–	4	0.5
x/y	DIV	11/14	3/4	11/14	5/8
$1/x$	RCP	4/?	1/?	4/?	1/?
\sqrt{x}	SQRT	13/18	3/6	12/18	6/12
$1/\sqrt{x}$	RSQRT	4/?	1/?	4/?	1/?

Tabela 1: Intel Skylake, 2019. Liczba cykli zegara dla wykonania (jednej — *Latency*, kolejnej — *Throughput*) instrukcji na liczbach single/double. Lewa strona: instrukcje 128-bit (SSE,SSE2) (single: xxxPS, double: xxxPD). Prawa strona: instrukcje AVX 256-bit (single: VxxxPS, double: VxxxPD).

Maksymalnie: $\underbrace{2}_{\text{potoki}} \times \underbrace{2}_{\text{wyniki/cykl}} \times \underbrace{2}_{\text{flop}} \times \underbrace{4}_{\text{w rej. AVX}} = 32 \text{ (double) flopów/cykl}$

Wrażliwość zadania i numeryczna poprawność algorytmu

Parę słów o działaniach arytmetycznych (IA-32) iii

Zegar 3.0 GHz $\rightarrow 32 \cdot 3 \cdot 10^9 \approx 100$ Gflop/s (/rdzeń).

Teoretycznie.

Więcej informacji:

- Intel® 64 and IA-32 Architectures Software Developer’s Manual Vol 1: Basic Architecture
- Intel® 64 and IA-32 Architectures Optimization Reference Manual

Wrażliwość zadania i numeryczna poprawność algorytmu

- zadanie obliczeniowe
- wrażliwość zadania na zaburzenia danych
- numeryczna poprawność algorytmu w fl
- wrażliwość zadania rozwiązywania układu równań liniowych
- numeryczna poprawność algorytmu GEPP

Zadanie obliczeniowe

Problem (zadanie obliczeniowe)

Dane jest $P : D \rightarrow W$ oraz x . Wyznaczyć $y = P(x)$.

W jest skończonego wymiaru, D — niekoniecznie.

Przykłady:

- Oblicz $y = \cos(x^2)$
- Dla danego $x \in \mathbb{R}^N$, oblicz $y = A \cdot x$
- Dla danych $A \in \mathbb{R}^{N \times N}$ oraz $b \in \mathbb{R}^N$, wyznacz $y \in \mathbb{R}^N$ t.ż. $A \cdot y = b$. Zauważmy, że A może być poza dziedziną P
- Dla danej funkcji ciągłej f znajdź wielomian p stopnia co najwyżej n , który najlepiej ją przybliży w sensie $\|\cdot\|_{[a,b]}$.

61

Błąd bezwzględny i względny

$\|\tilde{x} - x\| \leftarrow$ błąd bezwzględny

$\frac{\|\tilde{x} - x\|}{\|x\|} \leftarrow$ błąd względny ($x \neq 0$)

63

Wrażliwość

Chcemy obliczyć

$$y = P(x).$$

Z oczywistych powodów, zamiast x dysponujemy \tilde{x} .

Co można powiedzieć o

$$P(\tilde{x}) - P(x) \quad ?$$

$C = ?$

62

Wrażliwość (uwarunkowanie)

Współczynnik wzmocnienia błędu bezwzględnego na poziomie ϵ (przypadek najgorszy):

$$\text{cond}_{abs}(P, x, \epsilon) := \sup_{\|\Delta\| \leq \epsilon} \frac{\|P(x + \Delta) - P(x)\|}{\|\Delta\|}$$

Faktycznie, wtedy

$$\|P(\tilde{x}) - P(x)\| \leq \text{cond}_{abs}(P, x, \epsilon) \cdot \|\tilde{x} - x\| \quad \text{dla } \|\tilde{x} - x\| \leq \epsilon.$$

Idealizacja:

$$\text{cond}_{abs}(P, x) = \lim_{\epsilon \rightarrow 0^+} \sup_{\|\Delta\| \leq \epsilon} \frac{\|P(x + \Delta) - P(x)\|}{\|\Delta\|}$$

64

Wrażliwość (uwarunkowanie)

Gdy P różniczkowalna w x , to

$$\text{cond}_{abs}(P, x) = \|P'(x)\|.$$

Analogicznie możemy pytać o wrażliwość dla błędu względnego:

$$\text{cond}_{rel}(P, x) = \lim_{\epsilon \rightarrow 0^+} \sup_{\|\Delta\| \leq \epsilon} \frac{\frac{\|P(x + \Delta) - P(x)\|}{\|P(x)\|}}{\frac{\|\Delta\|}{\|x\|}}$$

Gdy P różniczkowalna w x , to

$$\text{cond}_{rel}(P, x) = \|P'(x)\| \frac{\|x\|}{\|P(x)\|}$$

65

Wrażliwość (uwarunkowanie)

Zatem dla $\tilde{x} \approx x$,

- $\|P(\tilde{x}) - P(x)\| \lesssim \text{cond}_{abs}(P, x) \|\tilde{x} - x\|$
- $\frac{\|P(\tilde{x}) - P(x)\|}{\|P(x)\|} \lesssim \text{cond}_{rel}(P, x) \frac{\|\tilde{x} - x\|}{\|x\|}$

Terminologia:

- Zadanie **dobrze uwarunkowane**: $\text{cond}(P, x)$ jest nieduże
- Zadanie **źle uwarunkowane**: $\text{cond}(P, x)$ jest bardzo duże

Zła wiadomość: zdarzają się zadania, których uwarunkowanie może być *patologicznie* duże, np. $\text{cond}(P, x) = 10^{23}$.

66

Uwarunkowanie układu równań liniowych

Rozwiązać układ równań:

$$Ay = b.$$

Jak wygląda zadanie obliczeniowe?

$$P : \underbrace{S \times \mathbb{R}^N}_{=D} \rightarrow \mathbb{R}^N, \quad P(A, b) = A^{-1}b$$

gdzie $S = \{A \in \mathbb{R}^{N \times N} : A \text{ jest nieosobliwa}\}$.

Zaburzenie danych:

$$\frac{\|\tilde{A} - A\|}{\|A\|} \leq \epsilon, \quad \frac{\|\tilde{b} - b\|}{\|b\|} \leq \epsilon.$$

Pytanie:

$$\frac{\|\tilde{y} - y\|}{\|y\|} \leq C \cdot \epsilon, \quad C = ?$$

67

Normy wektorowe, $x \in \mathbb{R}^N$

$$\|x\|_1 = \sum_{i=1}^N |x_i|,$$

$$\|x\|_2 = \sqrt{\sum_{i=1}^N |x_i|^2},$$

$$\|x\|_p = \left(\sum_{i=1}^N |x_i|^p \right)^{1/p},$$

$$\|x\|_\infty = \max_{i=1, \dots, N} |x_i|,$$

$$\|x\|_\infty \leq \|x\|_1 \leq N \|x\|_\infty,$$

$$\|x\|_\infty \leq \|x\|_2 \leq \sqrt{N} \|x\|_\infty,$$

$$\|x\|_2 \leq \|x\|_1 \leq \sqrt{N} \|x\|_2.$$



$$\{x : \|x\|_2 \leq 1\}$$



$$\{x : \|x\|_\infty \leq 1\}$$



$$\{x : \|x\|_1 \leq 1\}$$

68

Definicja (Norma macierzy indukowana normą wektorową)

$$\|A\| := \max_{\|x\| \neq 0} \frac{\|Ax\|}{\|x\|} = \max_{\|x\|=1} \|Ax\| = \max_{\|x\| \leq 1} \|Ax\|$$

Twierdzenie

- $\|Ax\| \leq \|A\| \cdot \|x\|$
- $\|AB\| \leq \|A\| \cdot \|B\|$
- $\|I\| = 1$
- $\|A\|_1 = \max_j \sum_i |a_{ij}|$
- $\|A\|_\infty = \max_i \sum_j |a_{ij}|$
- $\|A\|_2 = \max\{\sqrt{\mu} : \mu \text{ jest w.wł. } A^T A\}$

69

Oznaczmy

$$\text{cond}(A) := \|A\| \cdot \|A^{-1}\|.$$

Twierdzenie

Jeśli $\epsilon \text{ cond}(A) \leq 1/2$, to

$$\frac{\|\tilde{y} - y\|}{\|y\|} \leq 4 \text{ cond}(A) \cdot \epsilon.$$

Zła wiadomość: macierze *mogą* być patologicznie źle uwarunkowane, $\text{cond}(A) \gg 1$.

70

Uwarunkowanie układu równań liniowych — dowód

$$Ay = b \quad (A + \Delta)\tilde{y} = b + \delta$$

stąd

$$(A + \Delta)(\tilde{y} - y) = \delta - \Delta y,$$

więc

$$\tilde{y} - y = (A + \Delta)^{-1}(\delta - \Delta y).$$

Zatem

$$\|\tilde{y} - y\| = \|(A + \Delta)^{-1}(\delta - \Delta y)\| \leq \|(A + \Delta)^{-1}\| \|\delta - \Delta y\|.$$

71

Uwarunkowanie układu równań liniowych — dowód

Lemat

Jeśli $\|\Delta\| < 1$, to $I + \Delta$ nieosobliwa oraz

$$\frac{1}{1 + \|\Delta\|} \leq \|(I + \Delta)^{-1}\| \leq \frac{1}{1 - \|\Delta\|}$$

Na mocy lematu, jeśli A odwracalna i $\|A^{-1}\Delta\| < 1$, to $(A + \Delta)^{-1}$ istnieje oraz

$$\|(A + \Delta)^{-1}\| \leq \frac{\|A^{-1}\|}{1 - \|A^{-1}\|\|\Delta\|}$$

72

Uwarunkowanie układu równań liniowych — dowód

$$\begin{aligned}
 \|\tilde{y} - y\| &\leq \|(A + \Delta)^{-1}\| \|\delta - \Delta y\| \\
 &\leq \|(A + \Delta)^{-1}\| (\|\delta\| + \|\Delta\| \|y\|) \\
 &\leq \frac{\|A^{-1}\|}{1 - \|A^{-1}\| \|\Delta\|} (\|\delta\| + \|\Delta\| \|y\|) \\
 &\leq \frac{\|A^{-1}\|}{1 - \epsilon \|A\| \|A^{-1}\|} (\epsilon \cdot \underbrace{\|b\|}_{= \|Ay\|} + \epsilon \cdot \|A\| \|y\|) \\
 &\quad \leq \underbrace{\|A\| \|A^{-1}\|}_{\leq \|A\| \|y\|} \\
 &\leq \frac{2\|A\| \|A^{-1}\|}{1 - \frac{1}{2}} \epsilon \cdot \|y\| = 4 \operatorname{cond}(A) \epsilon \|y\|.
 \end{aligned}$$

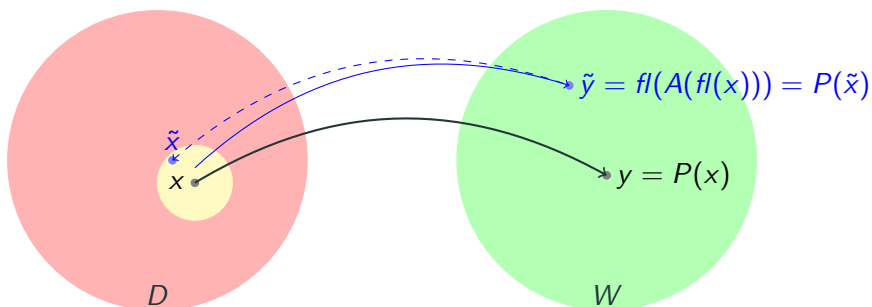
73

Algorytm numerycznie poprawny

$$y = P(x), \quad \tilde{y} = fl(A(fl(x))) = P(\tilde{x}),$$

gdzie

$$\frac{\|\tilde{x} - x\|}{\|x\|} \leq K \cdot \nu.$$



75

Algorytm

Problem (zadanie obliczeniowe)

Dane jest $P : D \rightarrow W$ oraz x . Wyznaczyć $y = P(x)$.

Definicja (algorytm dla zadania obliczeniowego)

Niech $X \subset D$. Przekształcenie $A : X \rightarrow W$, którego wartość można określić jako sekwencję operacji elementarnych, nazywamy *algorytmem* rozwiązywania zadania P w klasie X .

- Typowo algorytm będzie *skończoną* sekwencją operacji.
- Nieuniknione:
 - błąd reprezentacji w fl: informacji i wyniku
 - realizacja wszystkich działań w fl
- Naturalnie, chcielibyśmy, aby (w jakimś sensie)

$$fl(A(fl(x))) \approx P(x) \quad \forall x \in X.$$

74

Algorytm numerycznie poprawny i

$$y = P(x), \quad \tilde{y} = fl(A(fl(x))) = P(\tilde{x}).$$

Definicja (algorytm numerycznie poprawny)

Dla każdego $x \in X$ wynik algorytmu A zrealizowanego w fl $fl(A(fl(x)))$ jest dokładnym rozwiązaniem zadania dla danych x zaburzonych na poziomie błędu reprezentacji.

76

Algorytm numerycznie poprawny ii

Wniosek

Algorytm NP daje wynik, którego błąd można oszacować:

$$\frac{\|\tilde{y} - y\|}{\|y\|} = \frac{\|P(\tilde{x}) - P(x)\|}{\|P(x)\|} \lesssim \text{cond}_{rel}(P, x) \frac{\|\tilde{x} - x\|}{\|x\|} \leq K \cdot \text{cond}_{rel}(P, x) \cdot \nu.$$

Błąd algorytmu NP

- jest na poziomie nieuniknionego błędu związanego z reprezentacją danych.
- będzie mały dla zadań dobrze uwarunkowanych
- może być duży dla zadań źle uwarunkowanych

77

Przykład: suma dwóch liczb

Mamy więc

$$fl(A(fl(x))) = P(\tilde{x})$$

oraz

$$\|\tilde{x} - x\|_1 = \sum |\tilde{x}_i - x_i| = \sum |E_i x_i| \leq 2\nu \sum |x_i| = 2\nu \|x\|_1.$$

A więc jest NP! Dlaczego więc występuje redukcja cyfr?

Przykład: suma dwóch liczb

Zadanie obliczeniowe: $P(x_1, x_2) = x_1 + x_2$, gdzie $x_1, x_2 \in \mathbb{R}$.

Algorytm: $A(x_1, x_2) = x_1 + x_2$.

Czy jest NP?

$$\begin{aligned} fl(A(fl(x))) &= fl(x_1(1 + \epsilon_1) + x_2(1 + \epsilon_2)) = \\ &\quad \dots \text{gdzieś to już widzieliśmy} \dots \\ &= x_1 \underbrace{(1 + \epsilon_1)(1 + \delta)}_{=1+E_1} + x_2 \underbrace{(1 + \epsilon_2)(1 + \delta)}_{=1+E_2} \\ &= x_1(1 + E_1) + x_2(1 + E_2) =: \tilde{x}_1 + \tilde{x}_2 \end{aligned}$$

Przy czym

$$|E_i| \leq |\epsilon_i| + |\delta| \leq 2 \cdot \nu.$$

78

Przykład: suma dwóch liczb

Zbadajmy uwarunkowanie zadania $P(x_1, x_2) = x_1 + x_2$.

$$P'(x_1, x_2) = [1, 1] \implies \|P'(x)\|_1 = 1.$$

$$\text{cond}_{rel}(P, x) = \|P'(x)\|_1 \frac{\|x\|_1}{\|P(x)\|_1} = \frac{|x_1| + |x_2|}{|x_1 + x_2|}.$$

Zadanie sumy jest

- dobrze uwarunkowanie, gdy x_1, x_2 tego samego znaku,
- źle uwarunkowane, gdy $x_1 \approx -x_2$.

To złe uwarunkowanie jest przyczyną zjawiska redukcji cyfr przy odejmowaniu!

79

80

Numeryczna poprawność niektórych algorytmów

Zakładamy, że wszystkie dane, wyniki pośrednie i końcowe są liczbami maszynowymi znormalizowanymi.

- $+$, \times , \div , $\sqrt{}$, FMA — jeśli spełniają IEEE 754 — to są NP
- std algorytm rozw. układu z macierzą trójkątną $Ux = b$ jest NP: obliczony wynik \tilde{x} spełnia $(U + \Delta)\tilde{x} = b$, gdzie $|\Delta_{ij}|/|u_{ij}| \leq N \cdot \nu$.

81

Numeryczne kryterium numerycznej poprawności

Rozważmy układ równań $Ax = b$. Niech \tilde{x} — wynik obliczony w fl jakimś algorytmem. Czy jest prawdą, że

$$(A + \Delta)\tilde{x} = b + \delta, \quad \text{przy czym } \frac{\|\Delta\|}{\|A\|}, \frac{\|\delta\|}{\|b\|} \leq \epsilon?$$

Twierdzenie

Jeśli

$$\frac{\|b - A\tilde{x}\|}{\|A\|\|\tilde{x}\| + \|b\|} \leq \epsilon,$$

to istnieją Δ, δ t. że

$$(A + \Delta)\tilde{x} = b + \delta \quad \text{oraz} \quad \frac{\|\Delta\|}{\|A\|}, \frac{\|\delta\|}{\|b\|} \leq \epsilon.$$

83

Numeryczna „poprawność” (?) GEPP

Twierdzenie

Wynik \tilde{x} obliczony GEPP dla układu $Ax = b$ spełnia

$$(A + \Delta)\tilde{x} = b,$$

$$\|\Delta\|_{\infty}/\|A\|_{\infty} \leq KN^3\rho_N\nu,$$

$$\text{gdzie } \rho_N = \max_{i,j,k} |a_{ij}^{(k)}| / \max_{i,j} |a_{ij}|.$$

Dowód jest dość żmudny — pomijamy.

Niestety,

$$\rho_N \leq 2^{N-1} \quad (\text{osiągane}).$$

Dla wielu macierzy jest *znacznie lepiej*, dlatego mówimy, że GEPP jest „praktycznie” NP.

82

Czy złożenie dwóch algorytmów NP jest NP?

Zadanie: $x \mapsto P(x) = y \mapsto Q(y) = z$, tzn. $z = Q \circ P(x)$.

Założmy, że

- zadanie P rozwiązujemy algorytmem A , który jest NP w całej dziedzinie
- zadanie Q rozwiązujemy algorytmem B , który jest NP w całej dziedzinie
- istnieje P^{-1}

Czy algorytm $B \circ A$ też jest NP? Niekoniecznie.

84

Kiedy złożenie dwóch algorytmów NP jest NP?

$$\begin{aligned}x &\mapsto P(x) = y \mapsto Q(y) = z && \text{(zadanie)} \\x &\mapsto A(x) = \tilde{y} \mapsto B(\tilde{y}) = \tilde{z} && \text{(algorytm, już w fl)}\end{aligned}$$

Przy czym

- $\tilde{y} = P(\tilde{x})$, $\|\tilde{x} - x\| \leq K\nu\|x\|$, bo A jest NP
- $\tilde{z} = Q(\hat{y})$, $\|\hat{y} - \tilde{y}\| \leq K\nu\|\tilde{y}\|$, bo B jest NP

Definiując $\hat{x} = P^{-1}(\hat{y})$ mamy

$$\tilde{z} = Q(\hat{y}) = Q(P(\hat{x})) = Q \circ P(\hat{x}).$$

Wystarczy sprawdzić, czy

$$\frac{\|\hat{x} - x\|}{\|x\|} \leq \hat{K}\nu?$$

85

Liniowe zadanie najmniejszych kwadratów

Kiedy złożenie dwóch algorytmów NP jest NP?

$$x \mapsto P(x) = y \mapsto Q(y) = z$$

$$x \mapsto A(x) = \tilde{y} \mapsto B(\tilde{y}) = \tilde{z} = Q(P(\hat{x}))$$

Nietrudno policzyć, że

$$\|\hat{x} - x\| \lesssim K(1 + \text{cond}_{rel}(P^{-1}, \tilde{y}) \cdot \nu \cdot \|x\|).$$

Zatem $B \circ A$ jest NP, o ile P^{-1} jest dobrze uwarunkowane!

86

Liniowe zadanie najmniejszych kwadratów

- Rozkład QR macierzy
- Regularne LZNK
- Rozkład SVD macierzy
- Nieregularne LZNK

87

Przekształcenie Householdera

$$H = I - 2 \frac{vv^T}{v^T v}$$

Inaczej: $H = I - \gamma vv^T$, gdzie $\gamma = 2/\|v\|_2^2$. (Umowa: $v = 0 \implies H = I$)

Własności:

$$H = H^T \quad (\text{symetria})$$

Rzeczywiście:

$$H^T = (I - \gamma vv^T)^T = I^T - \gamma \underbrace{(vv^T)^T}_{(v^T)^T v^T} = I - \gamma vv^T = H.$$

$$H^T H = I \quad (\text{ortogonalność})$$

Rzeczywiście:

$$H^T H = HH = (I - \gamma vv^T)(I - \gamma vv^T) = I - 2\gamma vv^T + \gamma^2 v \underbrace{v^T v}_{=2/\gamma} v^T = I.$$

88

Przekształcenie Householdera

$$\|Hx\|_2 = \|x\|_2 \quad \forall x \quad (\text{izometria})$$

$$\text{Rzeczywiście: } \|Hx\|_2^2 = (Hx)^T (Hx) = x^T \underbrace{H^T H}_I x = x^T x = \|x\|_2^2.$$

- Reprezentacja H : wystarczy pamiętać wektor v (i kwadrat normy, $\|v\|_2^2$). **Jeśli zrobisz to źle, pamięć rośnie do N^2 !**
- Koszt wyznaczania $y = Hx$:

$$Hx = (I - 2 \frac{vv^T}{v^T v})x = x - \frac{2}{\|v\|_2^2} (v^T x) \cdot v.$$

Czyli $\mathcal{O}(4N)$. **Jeśli zrobisz to źle, koszt rośnie do $\mathcal{O}(N^2)$!**

89

Przekształcenie Householdera

Dla zadanego $b \in \mathbb{R}^N$, szukamy $v \in \mathbb{R}^N$ takiego, by

$$Hb = \alpha \cdot e_1 = \begin{bmatrix} \alpha \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Skoro $\|Hb\|_2 = \|b\|_2$, to $\alpha = \pm \|b\|_2$.

$$Hb = b - 2 \frac{vv^T}{v^T v} b = b - 2 \underbrace{\frac{v^T b}{v^T v}}_{\beta} v,$$

czyli

$$\beta v = b - \alpha e_1.$$

Stąd (dlaczego?) $v = b - \alpha e_1 = b + \text{sign}(b_1) \|b\|_2 e_1$.

Koszt: $\mathcal{O}(N)$.

90

Rozkład QR macierzy

Zakładamy, że $A \in \mathbb{R}^{M \times N}$, $M \geq N$ oraz $\text{rank}(A) = N$.

Rozkład wąski:

$$A = \hat{Q} \hat{R}$$

gdzie $\hat{Q} \in \mathbb{R}^{M \times N}$ ortogonalna $\hat{Q}^T \hat{Q} = I$ oraz $\hat{R} \in \mathbb{R}^{N \times N}$ — górna trójkątna, z niezerową diagonalą.

Rozkład pełny:

$$A = \begin{bmatrix} \hat{Q} & \tilde{Q} \end{bmatrix} \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix} = QR$$

oraz $Q \in \mathbb{R}^{M \times M}$, $Q^T Q = I$, natomiast $R \in \mathbb{R}^{M \times N}$ — górna „trójkątna” z niezerową główną diagonalą.

91

Rozkład QR macierzy: istnienie i algorytm

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad A_{11} \in \mathbb{R}^{p \times p}.$$

Niech

$$Q_1 R_1 = Q_1 \begin{bmatrix} \hat{R}_{11} \\ 0 \end{bmatrix} = \begin{bmatrix} A_{11} \\ A_{21} \end{bmatrix} \leftarrow \text{to macierz max rzędu (= } p)$$

Zatem

$$Q_1^T A = \begin{bmatrix} \hat{R}_{11} & R_{12} \\ 0 & \tilde{A}_{22} \end{bmatrix}$$

Niech dalej

$$Q_{22} R_{22} = \tilde{A}_{22}, \quad \leftarrow \text{to też macierz max rzędu (= } N - p)$$

Wtedy ostatecznie

$$A = \underbrace{Q_1 Q_2}_{Q_2} \underbrace{\begin{bmatrix} \hat{R}_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}}_R \quad \text{gdzie } Q_2 = \begin{bmatrix} I & \\ & Q_{22} \end{bmatrix}. \quad 92$$

QR iteracyjnie metodą Householdera ($p = 1$)

Algorytm 3 $A = QR$ metodą Householdera (wersja z iteracją)

for $k = 1 : N$

$$\text{Podziel } A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & A_{22} \end{bmatrix}, \quad a_{11} \in \mathbb{R}.$$

$$Q_k := \text{m. Householdera t.ż. } H \cdot \begin{bmatrix} a_{11} \\ a_{21} \end{bmatrix} = \begin{bmatrix} r_{kk} \\ 0 \end{bmatrix};$$

$$\begin{bmatrix} r_{12} \\ A_{22} \end{bmatrix} := Q_k^T \begin{bmatrix} a_{12} \\ A_{22} \end{bmatrix}$$

$$A := A_{22}$$

end

$$\text{return } Q = Q_1 \begin{bmatrix} 1 & \\ & Q_2 \end{bmatrix} \cdots \begin{bmatrix} I_{k-1} & \\ & Q_k \end{bmatrix} \cdots, R = \begin{bmatrix} r_{11} & r_{12} \\ 0 & \ddots \end{bmatrix}$$

Uwaga: czynników Q_i nie wyznaczamy!

Rozkład QR macierzy: algorytm rekurencyjny ($p \approx N/2$)

$$QR = A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad A_{11} \in \mathbb{R}^{p \times p}, \quad p \approx N/2.$$

Algorytm 2 $A = QR$ metodą Householdera (wersja rekurencyjna)

if $p = 1$ **then**

$$\text{return } Q := H \text{ m. Householdera t.ż. } H \cdot \begin{bmatrix} a_{11} \\ a_{21} \end{bmatrix} = \begin{bmatrix} r_{11} \\ 0 \end{bmatrix} =: R$$

end if

$$\text{Rozłóż } Q_1 R_1 = \begin{bmatrix} A_{11} \\ A_{21} \end{bmatrix} \text{ (rekurencja)}$$

$$\begin{bmatrix} R_{12} \\ A_{22} \end{bmatrix} := Q_1^T \begin{bmatrix} A_{12} \\ A_{22} \end{bmatrix}$$

$$\text{Rozłóż } Q_{22} R_{22} = A_{22} \text{ (rekurencja)}$$

$$\text{return } Q = Q_1 \begin{bmatrix} I & \\ & Q_{22} \end{bmatrix}, R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}$$

Uwaga: czynników Q_i nie wyznaczamy!

Koszt rozkładu QR (iteracyjnie metodą Householdera)

$$T(M, N) = \underbrace{K_1 \cdot M}_{\text{wyzn. } H} + \underbrace{4 \cdot M \cdot (N - 1)}_{\text{obl. } H \cdot \begin{bmatrix} a_{12} \\ A_{22} \end{bmatrix}} + \underbrace{T(M - 1, N - 1)}_{\text{pozost. kroki}}$$

Zatem sumarycznie

$$T(M, N) = 4 \sum_{k=1}^N (N - k)(M - k + 1) + K_1 \sum_{k=1}^N (M - k + 1) \approx 2MN^2 - \frac{2}{3}N^3$$

(z pominięciem wyrazów rzędu niższego niż 3.)

Numeryczna poprawność algorytmu rozkładu QR (iteracyjnie metodą Householdera)

Twierdzenie

Niech \tilde{R} będzie macierzą trójkątną wyznaczoną w fl algorytmem rozkładu QR (iteracyjnie, z metodą Householdera). Istnieje ortogonalna $Q = H_1 \cdots H_N$ taka, że

$$A + \Delta = Q \begin{bmatrix} \tilde{R} \\ 0 \end{bmatrix}, \quad \text{oraz } \|\Delta_j\|_2 \leq K \cdot MN \|a_j\|_2 \cdot \nu,$$

przy czym H_k — macierz Householdera wyznaczona dokładnie dla podmacierzy obliczonej w fl na k -tym kroku algorytmu.

Dowód pomijamy.

96

Liniowe zadanie najmniejszych kwadratów

Problem (LZNK)

Niech $A \in \mathbb{R}^{M \times N}$, $M \geq N$ oraz $\text{rank}(A) = N$ i niech $b \in \mathbb{R}^M$.

Znaleźć $x \in \mathbb{R}^N$ taki, że

$$\|b - Ax\|_2 \rightarrow \min!$$

tzn.

$$\|b - Ax\|_2 \leq \|b - Ay\|_2 \quad \forall y$$

98

Dygresja: jak używać wyniku

W wyniku dostajemy nie $A = QR$, ale

$$A = \underbrace{Q_1 \cdot Q_2 \cdots Q_N}_{=Q} \cdot R, \quad Q_i \in \mathbb{R}^{M \times M}.$$

Jak obliczać wyrażenia postaci

$$X = Q \cdot B?$$

Źle:

$$Q = Q_1 \cdot Q_2 \cdots Q_N \quad ???$$

$$X = Q \cdot B$$

Dobrze:

$$X = B$$

for $i = N : -1 : 1$

$$X = Q_i \cdot X$$

end

...dodatkowo wykorzystując postać Q_i !

97

Rozwiązywanie LZNK przez rozkład QR

Jeśli znamy rozkład QR macierzy,

$$A = QR = \begin{bmatrix} \hat{Q} & \tilde{Q} \end{bmatrix} \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix}$$

zadanie staje się banalne:

$$\begin{aligned} \|b - Ax\|_2^2 &= \|b - QRx\|_2^2 = \|Q(Q^T b - Rx)\|_2^2 \\ &= \|Q^T b - Rx\|_2^2 = \left\| \begin{bmatrix} \hat{Q}^T \\ \tilde{Q}^T \end{bmatrix} b - \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix} x \right\|_2^2 \\ &= \left\| \begin{bmatrix} \hat{Q}^T b - \hat{R}x \\ \tilde{Q}^T b \end{bmatrix} \right\|_2^2 = \underbrace{\|\hat{Q}^T b - \hat{R}x\|_2^2}_{\text{najmniejsze, gdy } \hat{R}x = \hat{Q}^T b} + \|\tilde{Q}^T b\|_2^2 \end{aligned}$$

Algorytm 4 Rozwiązywanie LZNK

Oblicz $\hat{b} := \hat{Q}^T b$ (koszt: $\mathcal{O}(MN)$)

Rozwiąż $\hat{R}x = \hat{b}$ (koszt: $\mathcal{O}(N^2)$)

return x

99

Dygresja: układ równań liniowych przez rozkład QR

Jeśli A — kwadratowa nieosobliwa,

$$A = QR,$$

- Q — ortogonalna, $Q^T Q = I$
- R — górna trójkątna

Układ równań:

$$Ax = b \quad \implies \quad Q \underbrace{Rx}_{=:y} = b,$$

skąd algorytm:

1. $y = Q^T b$ (koszt: $\mathcal{O}(N^2)$)
2. Rozwiąż układ z macierzą trójkątną $Rx = y$. (koszt: $\mathcal{O}(N^2)$)

Ale rozkład QR wymaga ok. 2 razy więcej flopów niż rozkład LU.

100

Liniowe zadanie najmniejszych kwadratów — układ normalny

Skoro

$$x = \hat{R}^{-1} \hat{Q}^T b,$$

to mnożąc przez (nieosobliwą) $\hat{R}^T \hat{R}$ mamy

$$\hat{R}^T \hat{R} x = \hat{R}^T \hat{Q}^T b = \begin{bmatrix} \hat{R}^T & 0 \end{bmatrix} \begin{bmatrix} \hat{Q}^T \\ \hat{Q}^T \end{bmatrix} b = R^T Q^T b = A^T b.$$

Z drugiej strony, $\hat{R}^T \hat{R} = \hat{R}^T \underbrace{\hat{Q}^T \hat{Q}}_I \hat{R} = A^T A$, więc ostatecznie x jest rozwiązaniem **układu równań normalnych**

$$A^T A x = A^T b.$$

101

Liniowe zadanie najmniejszych kwadratów — układ normalny

$$A^T A x = A^T b.$$

Własności $A^T A$:

- mała ($N \times N$)
- symetryczna
- dodatnio określona (bo A jest pełnego rzędu),

Stąd alternatywny algorytm rozwiązania LZNK:

Oblicz $B = A^T A$ (koszt MN^2)

Wyznacz rozkład Cholesky'ego $B = LL^T$ (koszt $\mathcal{O}(N^3)$)

Rozwiąż układ $LL^T x = A^T b$ (koszt $\mathcal{O}(MN)$)

Jest ok. dwa razy taniej, ale ten sposób może mieć gorsze własności numeryczne (od metody wykorzystującej rozkł. QR macierzy A)

102

Rozkład SVD

Twierdzenie (rozkład w/g wartości szczególnych)

Niech $A \in \mathbb{R}^{M \times N}$, $M \geq N$. Istnieje rozkład

$$A = U \Sigma V^T :$$

- $U \in \mathbb{R}^{M \times N}$ ortogonalna, $U^T U = I$
- $V \in \mathbb{R}^{N \times N}$ ortogonalna, $V^T V = I$
- $\Sigma \in \mathbb{R}^{N \times N}$ diagonalna, $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_N)$,
 $\sigma_1 \geq \dots \geq \sigma_N \geq 0$.

Gdy $M < N$, rozkład SVD definiujemy dla macierzy A^T .

Kolumny U , V : lewe/prawe wektory szczególne.

σ_i : wartości szczególne.

103

Rozkład SVD

- $A = U\Sigma V^T$ — rozkład wąski
- $A = \underbrace{\begin{bmatrix} U & \tilde{U} \end{bmatrix}}_{\text{ortog. } M \times M} \begin{bmatrix} \Sigma \\ 0 \end{bmatrix} V^T$ — rozkład pełny

Rozkład SVD jest kosztowny.

- Jest trudniejszy od zadania własnego, które jest trudniejsze od układu równań liniowych
- Wymaga procesu iteracyjnego, który dopiero w granicy jest zbieżny do rozwiązania
- W praktyce, ze względu na ograniczenie precyzji wyniku w fl, wykonuje się skończoną liczbę iteracji.
Koszt $\approx \mathcal{O}(MN^2) + \mathcal{O}(N^3)$ (stałe kilkakrotnie większe niż dla rozkładu QR)

104

Rozkład SVD — zastosowania

- Wyznaczenie rzędu macierzy
- PCA, używane w uczeniu maszynowym
- Rozwiązanie LZNK, gdy A jest pełnego rzędu¹: Skoro x spełnia

$$A^T A x = A^T b$$

to $A^T A = (V \Sigma U^T)(U \Sigma V^T) = V \Sigma^2 V^T$, więc

$$\begin{aligned} A^T A x = A^T b &\iff \\ V \Sigma^2 V^T x = V \Sigma U^T b &\iff V^T x = \Sigma^{-1} U^T b \iff \\ x &= V \Sigma^{-1} U^T b \end{aligned}$$

(Σ nieosobliwa, bo $\text{rank}(A) = N$.)

- Rozwiązanie LZNK, gdy A nie jest pełnego rzędu: o tym za chwilę.

¹Ale taniej użyć rozkładu QR!

106

Rozkład SVD — własności

Twierdzenie

1. Wartości własne $A^T A$ są równe $\sigma_1^2, \dots, \sigma_N^2$.
2. Jeśli $\sigma_1 \geq \dots \geq \sigma_r > \sigma_{r+1} = \dots = \sigma_N = 0$, to rząd A jest równy r .
3. $A = \sum_{i=1}^N \sigma_i u_i v_i^T$, gdzie u_i, v_i — kolumny U, V .
4. **Tw. Eckharta–Younga** Macierz rzędu $\leq k \leq N$ najbliższa A w normie $\|\cdot\|_2$ lub $\|\cdot\|_F$ to $A_k = \sum_{i=1}^k \sigma_i u_i v_i^T$.

Dowód: Na tablicy. \square

105

Uwarunkowanie LZNK

Twierdzenie

Niech $A \in \mathbb{R}^{M \times N}$, $M \geq N$ oraz $\text{rank}(A) = N$ i niech $b \in \mathbb{R}^M$.

$$\underbrace{\|b - Ax\|_2 \rightarrow \min!}_{\text{zad. dokładne}} \quad \underbrace{\|(b + \delta) - (A + \Delta)x\|_2 \rightarrow \min!}_{\text{zad. zaburzone}}$$

Niech $\epsilon = \max\left\{\frac{\|\Delta\|_2}{\|A\|_2}, \frac{\|\delta\|_2}{\|b\|_2}\right\}$, $\text{cond}_2(A)\epsilon < 1$. Wtedy zadanie zaburzone ma dokładnie jedno rozwiązanie \tilde{x} oraz

$$\frac{\|\tilde{x} - x\|_2}{\|x\|_2} \leq \underbrace{\epsilon \left(2 \frac{\text{cond}_2(A)}{\cos \theta} + \text{tg } \theta \cdot \text{cond}_2^2(A) \right)}_{=: \text{cond}_{\text{LZNK}}(A, b)} + \mathcal{O}(\epsilon^2),$$

gdzie $\sin \theta = \frac{\|b - Ax\|_2}{\|b\|_2}$, $\text{cond}_2(A) := \frac{\max \sigma_i}{\min \sigma_i}$.

107

$$\text{cond}_{LZNK}(A, b) = 2 \frac{\text{cond}_2(A)}{\cos \theta} + \text{tg } \theta \cdot \text{cond}_2^2(A)$$

$$\text{gdzie } \sin \theta = \frac{\|b - Ax\|_2}{\|b\|_2}, \quad \text{cond}_2(A) := \frac{\max \sigma_i}{\min \sigma_i}.$$

- **przypadek typowy:** $\sin \theta \approx 0$ (mała reszta)
 $\implies \text{cond}_{LZNK}(A, b) \approx \text{cond}_2(A)$
- $\sin \theta \approx 1$ (duża reszta i $x \approx 0$) $\implies \text{cond}_{LZNK}(A, b) \gg 1$
- pozostałe: $\implies \text{cond}_{LZNK}(A, b) \approx \text{cond}_2^2(A)$

Zauważmy, że $\text{cond}_2(A^T A) = \text{cond}_2^2(A)$ — jeszcze jeden powód by r-ń normalnych używać z rozwagą.

108

Nieregularne LZNK — rozwiązanie przez SVD

$$\text{Niech } A = U \begin{bmatrix} \Sigma \\ 0 \end{bmatrix} V^T, \text{ przy czym } U = \begin{bmatrix} u_1 & \dots & u_M \end{bmatrix},$$

$$V = \begin{bmatrix} v_1 & \dots & v_N \end{bmatrix}. \text{ Mamy}$$

$$\begin{aligned} \|b - Ax\|_2^2 &= \|b - U \Sigma V^T x\|_2^2 = \|U(U^T b - \Sigma V^T x)\|_2^2 \\ &= \|U^T b - \underbrace{\Sigma V^T x}_{=: y}\|_2^2 = \left\| \begin{bmatrix} u_1^T b \\ \vdots \\ u_r^T b \\ \vdots \\ u_M^T b \end{bmatrix} - \begin{bmatrix} \sigma_1 y_1 \\ \vdots \\ \sigma_r y_r \\ 0 \\ \vdots \end{bmatrix} \right\|_2^2 \\ &= \sum_{i=1}^r (\sigma_i y_i - u_i^T b)^2 + \sum_{i=r+1}^M (u_i^T b)^2 \end{aligned}$$

110

Problem (nieregularne LZNK)

$$A \in \mathbb{R}^{M \times N}, \quad M \geq N, \text{ ale } \text{rank}(A) < N.$$

Szukamy $x \in \mathbb{R}^N$ t.że

$$\|b - Ax\|_2 \rightarrow \min!$$

Skoro $\ker(A) \neq 0$, to rozwiązanie nieregularnego LZNK nie jest jednoznaczne:

x minimalizuje $\|b - Ax\|_2 \implies x + z$, gdzie $z \in \ker(A)$, też minimalizuje...

109

Nieregularne LZNK — rozwiązanie przez SVD

$$\|b - Ax\|_2^2 = \sum_{i=1}^r (\sigma_i y_i - u_i^T b)^2 + \sum_{i=r+1}^M (u_i^T b)^2$$

Czyli $x = Vy$ minimalizuje resztę wtw gdy

$$y_i = \frac{u_i^T b}{\sigma_i}, \quad i = 1, \dots, r. \quad (\text{pozostałe } y_i \text{ — dowolne})$$

Wniosek

Rozwiązanie x nieregularnego LZNK o **najmniejszej normie**:

$$\|b - Ax\|_2 \rightarrow \min!$$

t. że $\|x\|_2 \rightarrow \min!$ jest jednoznacznie określone i dane wzorem

$$x = \sum_{i=1}^r \frac{u_i^T b}{\sigma_i} v_i.$$

Dwd: Rzeczywiście, $\|x\|_2 = \|Vy\|_2 = \|y\|_2$.

111

Metody iteracyjne dla układów równań liniowych

Jak to zrobić szybciej i taniej?

Zrezygnować z wyznaczania rozwiązania „dokładnego”!

Będziemy zadowoleni, jeśli obliczone rozwiązanie (przybliżone) \tilde{x} spełnia

$$\frac{\|\tilde{x} - x\|}{\|x\|} \leq \epsilon,$$

lub nawet tylko

$$\frac{\|b - A\tilde{x}\|}{\|b\| + \|A\|\|\tilde{x}\|} \leq \epsilon$$

dla zadanego przez użytkownika kryterium tolerancji $\epsilon > 0$.

Układy wielkiego rozmiaru

Problem

Rozwiązać

$$Ax = b,$$

gdzie A jest rozmiaru $N \times N$.

Aby rozwiązać je dla $N = 10^6$ metodą **GEPP** na współczesnej stacji roboczej², potrzeba co najmniej

- 10^7 sekund tzn. ≈ 4 miesiące
- 10^{13} bajtów pamięci tzn. ≈ 9 TB

²Zakładamy szybkość 100Gflop/s.

Topowy i7 z MKL dla DGETRF wyciągał w 2017 r. ok. 200Gflop/s

Metody stacjonarne oparte na rozszczepieniu

Niech $A = M - Z$, przy czym M — nieosobliwa.

$$Ax^* = b \iff (M - Z)x^* = b \iff Mx^* = b + Zx^* \iff x^* = M^{-1}(b + Zx^*).$$

Rozwiązanie x^* to punkt stały

$$x^* = \Phi(x^*), \quad \text{gdzie } \Phi(x) = \underbrace{M^{-1}Z}_{B} x + \underbrace{M^{-1}b}_{c}$$

Możemy spróbować użyć iteracji prostej (Banacha)

$$x_{k+1} = \Phi(x_k).$$

Jak dobrać M , by

- $x_k \rightarrow x^*$ niezależnie od x_0
- krok iteracji był niedrogi?

Zbieżność metod stacjonarnych

Niech $x^* = Bx^* + c$.

Twierdzenie (o zbieżności metody stacjonarnej)

Ciąg

$$x_{k+1} = Bx_k + c$$

jest zbieżny do x^* dla każdego $x_0 \iff \rho(B) < 1$,

gdzie

$$\rho(B) = \max\{|\lambda| : \lambda \text{ jest w.wł } B\}$$

to promień spektralny macierzy B .

115

Zbieżność metod stacjonarnych

Twierdzenie (o liniowej zbieżności iteracji liniowej)

Jeśli $\|B\| < 1$, to ciąg $x_{k+1} = Bx_k + c$ jest zbieżny do x^* dla każdego x_0 oraz

$$\|x_{k+1} - x^*\| \leq \|B\| \cdot \|x_k - x^*\|.$$

Dowód:

$$x_{k+1} - x^* = (Bx_k + c) - (Bx^* + c) = B(x_k - x^*),$$

skąd

$$\|x_{k+1} - x^*\| = \|B(x_k - x^*)\| \leq \|B\| \|x_k - x^*\|.$$

Przez indukcję,

$$\|x_k - x^*\| \leq \|B\|^k \|x_0 - x^*\| \rightarrow 0.$$

□

116

Zbieżność metod stacjonarnych

Wniosek (metody oparte na rozszczepieniu)

Niech $A = M - Z$ oraz A, M — nieosobliwe. $Ax^* = b$.

- Jeśli $\rho(M^{-1}Z) < 1$, to metoda iteracyjna

$$Mx_{k+1} = Zx_k + b$$

jest zbieżna do x^* z każdego x_0 .

- Jeśli dodatkowo $\gamma := \|M^{-1}Z\| < 1$, to dodatkowo

$$\|x_{k+1} - x^*\| \leq \gamma \cdot \|x_k - x^*\|.$$

117

Przykłady metod opartych na rozszczepieniu

Niech $A = L + D + U$, $D = \text{diag}(A)$,

L (odp. U) — dolna (odp. górna) trójkątna z zerową diagonalą.

$$x_{k+1} = x_k + M^{-1}(b - Ax_k)$$

- Metoda **Jacobiego**:

$$M = D,$$

- Metoda **Gaussa–Seidela**:

$$M = D + L,$$

- Metoda **SOR**:

$$M = \frac{1}{\omega} D + L.$$

W niektórych przypadkach znamy optymalne ω .

Są to metody **algebraiczne**: wymagają dostępu do elementów macierzy.

118

Metoda Jacobiego

$$x_{k+1} = x_k + D^{-1}(b - Ax_k)$$

Twierdzenie

Jeśli A jest diagonalnie dominująca,

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}| \quad \forall i = 1, \dots, N,$$

to m. Jacobiego jest zbieżna do x^* dla każdego x_0 .

Dowód: Wyraz (i, j) macierzy $M^{-1}Z$ wynosi 0 dla $i = j$ oraz $-a_{ij}/a_{ii}$ dla $i \neq j$, więc

$$\|M^{-1}Z\|_{\infty} = \max_i \sum_{j \neq i} |a_{ij}|/|a_{ii}| < 1.$$

□

119

Metody projekcji

W naszych metodach,

$$x_{k+1} = x_k + \delta_k.$$

Jak wybierać δ_k ? Idealna poprawka, δ_k^* , spełniałaby:

$$A\delta_k^* = r_k \implies x_{k+1} = x^*$$

Wyznamy przybliżoną poprawkę idealną: $\delta_k = V_k a_k$, gdzie $V_k, U_k \in \mathbb{R}^{N \times r}$ max. rzędu t. że $a_k \in \mathbb{R}^r$ spełnia

$$\underbrace{U_k^T A V_k}_{r \times r} a_k = U_k^T r_k.$$

120

Metody projekcji — przykłady

Metoda najszybszego spadku dla $A = A^T > 0$:

$$U_k = V_k = r_k \implies a_k = \frac{r_k^T r_k}{r_k^T A r_k}$$

$$x_{k+1} = x_k + a_k r_k$$

Twierdzenie (o zbieżności metody najszybszego spadku)

Jeśli $A = A^T > 0$, to

$$\|x^* - x_{k+1}\|_A \leq \frac{\kappa - 1}{\kappa + 1} \|x^* - x_k\|_A,$$

gdzie $\kappa = \text{cond}_2(A) = \lambda_{\max}(A)/\lambda_{\min}(A)$, $\|x\|_A^2 := x^T A x$.

121

Metody projekcji — przykłady

Metoda najmniejszego residuum

$$V_k = r_k, \quad U_k = A V_k = A r_k.$$

$$a_k = \frac{r_k^T A^T r_k}{r_k^T A^T A r_k},$$

$$x_{k+1} = x_k + a_k r_k.$$

Twierdzenie

Założmy, że macierz $A_{\text{sym}} = (A + A^T)/2$ jest dodatnio określona i oznaczmy $\mu = \lambda_{\min}(A_{\text{sym}}) > 0$. Wtedy

$$\|r_{k+1}\|_2 \leq \left(1 - \frac{\mu^2}{\|A\|_2^2}\right)^{1/2} \|r_k\|_2.$$

122

Metody przestrzeni Kryłowa

Niech $r_k = b - Ax_k$.

Zdefiniujemy **przestrzeń Kryłowa** $K_k = \text{lin}\{r_0, Ar_0, \dots, A^{k-1}r_0\}$.

Oczywiście,

$$K_0 \subseteq K_1 \subseteq \dots \subseteq K_k \subseteq K_{k+1} \subseteq \dots \subseteq \mathbb{R}^N.$$

Definiujemy x_k jako rozwiązanie zadania minimalizacji

$$\|x - x^*\| = \min! \quad \text{albo} \quad \|b - Ax\| = \min!$$

dla $x \in x_0 + K_k$.

Wybór normy i minimalizowanej wielkości determinuje konkretną metodę iteracyjną (są też inne m. Kryłowa).

Będą to metody **operatorowe**: wystarczy dostęp do procedury obliczania $y := A \cdot x$.

123

Metoda CG (gradientów sprzężonych)

Dla $A = A^T > 0$.

Iteracja x_k zdefiniowana przez $\|x - x^*\|_A = \min!$, tzn.

$$\|x_k - x^*\|_A \leq \|x - x^*\|_A \quad \forall x \in x_0 + K_k.$$

Jest to LZNK: Gdy V_k zawiera bazę K_k , to $x_k = x_0 + V_k a$ i

$$\|x_k - x^*\|_A^2 = \|A^{1/2}(x_k - x^*)\|_2^2 = \|A^{1/2}(V_k a + x_0 - x^*)\|_2^2,$$

Układ normalny to $V_k^T A V_k a = V_k^T A(x^* - x_0) = V_k^T r_0$. Można go bardzo tanio rozwiązywać. (Jeśli zrobisz to źle, koszt k -tej iteracji wyniesie $\mathcal{O}(N \cdot k^2)$ flopów plus $N \times k$ pamięci.)

124

Metoda CG — implementacja

Wyprowadzenie tego algorytmu jest nietrywialne.

$$r = b - Ax;$$

$$\rho_0 = \|r\|_2^2, \beta = 0, k = 1$$

while not stop **do**

$$p = r + \beta p$$

$$w = Ap$$

$$\alpha = \rho_{k-1} / p^T w$$

$$x = x + \alpha p$$

$$r = r - \alpha w$$

$$\rho_k = \|r\|_2^2; \beta = \rho_k / \rho_{k-1}$$

$$k = k + 1$$

end while

Kosz jednej iteracji: $\underbrace{\text{SPMV}(A, x)}_{=2 \cdot \text{nnz}(A)} + \mathcal{O}(N)$. Pamięć: $\mathcal{O}(N)$.

125

Metoda CG — zbieżność

Twierdzenie (o zbieżności CG jako metody iteracyjnej)

Po k iteracjach metody CG,

$$\|x_k - x\|_A \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k \|x_0 - x\|_A,$$

gdzie $\kappa = \text{cond}_2(A) = \lambda_{\max}(A) / \lambda_{\min}(A)$.

Metoda CG

Została uznana za jeden z 20 najważniejszych algorytmów wymyślonych w XX wieku.

126

Metoda GMRES

Iteracja x_k zdefiniowana przez $\|r\|_2 = \min!$, tzn.

$$\|r_k\|_2 = \|b - Ax_k\|_2 \leq \|b - Ax\|_2 \quad \forall x \in x_0 + K_k.$$

Inteligentnie zaimplementowana m. GMRES ma koszt k -tej iteracji

- $\mathcal{O}(N \cdot k)$ flopów (trzeba rozwiązywać LZNK, aczkolwiek dość tanie)
- $\mathcal{O}(N \cdot k)$ pamięci (trzeba pamiętać bazę p-ni Kryłowa)

Uwaga: są macierze nieosobliwe, dla których GMRES może nie zadziałać lub nie być zbieżna.

127

Ściskanie macierzy

Wymagania:

- MA ma **korzystniejsze własności spektralne** z punktu używanej metody iteracyjnej,
- M jest **łatwa w „konstrukcji”**,
- M jest **tania w mnożeniu przez wektor**.

Kilka nietrafionych wyborów:

- $M = A^{-1}$??
- $M = I$??

Lepsze wybory („ $M \approx A^{-1}$, ale tania”):

- jeden (kilka?) kroków prostej metody iteracyjnej
- niepełny rozkład LU, itp.
- najlepiej: dobrać specjalnie do konkretnego zadania!

129

Ściskanie macierzy

$$A \cdot x = b$$

Przykry fakt

Szybkość zbieżności m. iteracyjnych zależy od własności spektralnych A , np. CG zależy od $\text{cond}_2(A) = \lambda_{\max}(A)/\lambda_{\min}(A)$.

Pomysł: Zadanie równoważne (M nieosobliwa):

$$\underbrace{MA}_{\text{nowa macierz}} \cdot x = Mb$$

może mieć macierz o *znacznie lepszych* własnościach!

128

Formaty macierzy rzadkich

Najbardziej popularne:

- współrzędnych
- CSR/CSC

130

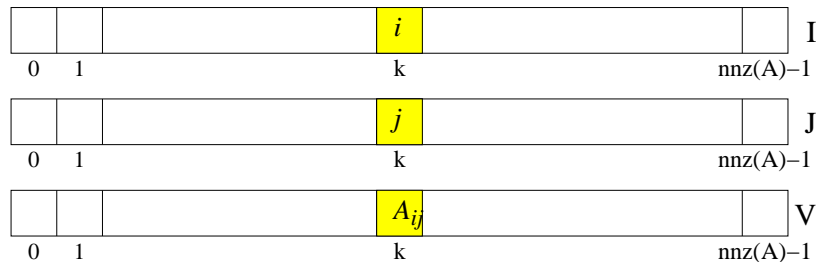
Format współrzędnych

Macierz A rozmiaru $N \times M$, licząca $\text{nnz}(A)$ niezerowych elementów.

- V — tablica typu **double**,
- I, J — tablice typu **int**;

Wszystkie o długości $\text{nnz}(A)$, przy czym zachodzi

$$A_{I[k], J[k]} = V[k], \quad k = 0, \dots, \text{nnz}(A) - 1,$$



131

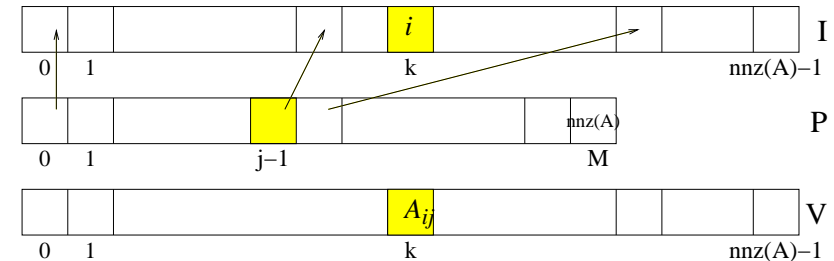
Zagadnienie własne

Formaty CSR/CSC

Zajmiemy się **CSC** = *Compressed Sparse Column*.

Analogicznie działa CSR = *Compressed Sparse Row*.

- V — tablica typu **double**, **kolejne** el-ty A **kolumnami**
- I — tablica typu **int**, długości $\text{nnz}(A)$
- P — **tablica typu int**, długości $M + 1$



$P[0]=0, P[M]=\text{nnz}(A)$.

$$A_{I[k], j} = V[k], \quad k = P[j-1], \dots, P[j] - 1.$$

132

Zagadnienie własne — zadania obliczeniowe

Problem

Znaleźć $v \in \mathbb{C}^N$ oraz $\lambda \in \mathbb{C}$ t. że

$$Av = \lambda v, \quad \|v\| = 1.$$

W praktyce spotyka się zadania:

- znaleźć **największą/najmniejszą** (co do modułu) wartość własną (i ew. odp. wektor własny)
- znaleźć wartość własną **bliską zadanej liczbie** (i ew. odp. wektor własny)
- znaleźć **wszystkie** wartości własne (i ew. odp. wektory własne)

133

Przypomnienie z GAL-u

- Każda macierz $A \in \mathbb{R}^{N \times N}$, **symetryczna** ma rozkład

$$A = Q \Lambda Q^T,$$

gdzie $Q \in \mathbb{R}^{N \times N}$ — macierz ortogonalna, $Q^T Q = I$,

$$\Lambda = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_N \end{bmatrix}$$

$\lambda_i \in \mathbb{R}$ — wartości własne A .

Kolumny Q — wektory własne A .

134

Metoda potęgowa

Problem

Zakładamy, że $|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_N|$.

Wyznaczyć parę własną (λ_1, v_1) .

Wybieramy (losowy?) x_0 .

while not stop **do**

$$x_{k+1} = Ax_k$$

$$x_{k+1} = x_{k+1} / \|x_{k+1}\|$$

$$k = k + 1$$

end while

Ta metoda leży u podstaw PageRank. Tylko skąd wiadomo, że $\lambda = 1$ jest *dominująca*?

135

Metoda potęgowa

Pomijając normowanie, mamy:

$$x_k = A \cdot x_{k-1} = A^2 \cdot x_{k-2} = \dots = A^k \cdot x_0.$$

Twierdzenie

Jeśli $x_0^T v_1 \neq 0$, to x_k dąży do kierunku v_1 , gdy $k \rightarrow \infty$.

Dowód: Tylko dla przypadku, gdy istnieje baza ortogonalna złożona z wektorów własnych A . Zatem $x_0 = \sum_i \alpha_i v_i$, $\alpha_1 \neq 0$ oraz

$$x_k = A^k x_0 = \sum_i \lambda_i^k \alpha_i v_i = \lambda_1^k \underbrace{(\alpha_1 v_1 + \sum_{i>1} \left(\frac{\lambda_i}{\lambda_1}\right)^k \alpha_i v_i)}_{\rightarrow 0}.$$

Gdy $|\lambda_2|/|\lambda_1| \approx 1$, zbieżność może być wolna. \square

Błąd zaokrągleń pomaga: nawet jeśli $x_0^T v_1 = 0$, to wskutek fl składowa v_1 ma szansę się pojawić w trakcie obliczeń.

136

Iloraz Rayleigh

Problem

Mając przybliżony wektor własny v , jak dobrać dla niego λ ?

Zminimalizować resztę,

$$\|Av - \lambda v\|_2 \rightarrow \min!$$

To jest LZNK, a rozwiązaniem jest

$$\lambda = \frac{x^H A x}{x^H x} \quad (\text{iloraz Rayleigh})$$

Dowód: Na tablicy. \square

137

Transformacje spektrum

Jeśli pary własne A to (λ_i, v_i) , to wtedy

Macierz	wart. wł.	wekt. wł.	zastrz.
A	λ_i	v_i	
$A - \mu I$	$\lambda_i - \mu$	v_i	
A^{-1}	$\frac{1}{\lambda_i}$	v_i	A nieosobl.
$(A - \mu I)^{-1}$	$\frac{1}{\lambda_i - \mu}$	v_i	$A - \mu I$ nieosobl.

Dowód: Na tablicy. \square

138

Wyznaczanie wartości własnej najbliższej zadanej liczbie

Problem

Dla zadanej liczby μ szukamy w. własnej λ^* takiej, że $|\lambda^* - \mu| \leq |\lambda_i - \mu|$ dla każdego i .

Wartości własne $(A - \mu I)^{-1}$: $\frac{1}{|\lambda_i - \mu|}$.

Metoda potęgowa dla $(A - \mu I)^{-1}$:

Wyznacz rozkład, np. $P(A - \mu I) = LU$

while not stop **do**

Rozwiąż $LUx_{k+1} = Px_k$

$x_{k+1} = x_{k+1} / \|x_{k+1}\|$

$k = k + 1$

end while

$\lambda^* = x_k^H A x_k \quad \triangleright x_k^H x_k = 1$

Jeśli zrobisz to źle, koszt jednej iteracji rośnie N -krotnie.

140

Odwrotna metoda potęgowa

Problem

Zakładamy, że $|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_{N-1}| > |\lambda_N| > 0$.

Wyznaczyć parę własną (λ_N, v_N) .

Wartości własne A^{-1} :

$$\frac{1}{|\lambda_N|} > \frac{1}{|\lambda_{N-1}|} \geq \dots \geq \frac{1}{|\lambda_1|}.$$

Metoda potęgowa dla A^{-1} :

Wyznacz rozkład, np. $PA = LU$

while not stop **do**

Rozwiąż $LUx_{k+1} = Px_k$

$x_{k+1} = x_{k+1} / \|x_{k+1}\|$

$k = k + 1$

end while

Jeśli zrobisz to źle, koszt jednej iteracji rośnie N -krotnie.

139

RQI

Zbieżność odwrotnej metody potęgowej można przyspieszyć. Zamiast: „Poprawiamy” μ :

Odwrotna metoda potęgowa

Wybieramy x_0 .

while not stop **do**

Rozwiąż $(A - \mu I)x_{k+1} = x_k$

$x_{k+1} = x_{k+1} / \|x_{k+1}\|$

$k = k + 1$

end while

$\lambda^* = x_k^H A x_k$

Metoda RQI

Wybieramy x_0 . $\lambda_0 = \mu$

while not stop **do**

Rozw. $(A - \lambda_k I)x_{k+1} = x_k$

$x_{k+1} = x_{k+1} / \|x_{k+1}\|$

$\lambda_{k+1} = x_{k+1}^H A x_{k+1}$

$k = k + 1$

end while

RQI = Rayleigh Quotient Iteration

Kłopot: nie mamy gwarancji, że zbiegniemy do najbliższej μ .

141

Pełne zagadnienie własne

Problem

Znaleźć wszystkie wartości własne (i ew. wektory własne) A .

Warto wcześniej sprowadzić A do prostszej postaci.

- Można skonstruować sekwencję przekształceń ortog. Q_i , które kosztem $\mathcal{O}(N^3)$ sprowadzą A do postaci **Hessenberga**:

$$Q_{N-1} \cdots Q_1 A Q_1^T \cdots Q_{N-1}^T = \begin{bmatrix} * & * & * & * & \cdots & * \\ * & * & * & * & \cdots & * \\ & * & * & * & \cdots & * \\ & & \ddots & \ddots & \ddots & \vdots \\ & & & \ddots & \ddots & * \\ & & & & * & * \end{bmatrix}$$

- Gdy $A = A^T$, postać Hessenberga jest trójdzielna (i symetryczna).

Dowód: Na ćwiczeniach. \square

142

Metoda dziel i rządź (idea)

$$A = \begin{bmatrix} T_1 & \\ & T_2 \end{bmatrix} + b_m u u^T$$

Dziel: założmy, że już rozwiązaliśmy zadanie dla T_i :

$Q_i^T T_i Q_i = D_i$, $i = 1, 2$. **Rządź:** Wtedy

$$\begin{bmatrix} Q_1 & Q_2 \end{bmatrix}^T \left(\begin{bmatrix} T_1 & \\ & T_2 \end{bmatrix} + b_m u u^T \right) \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} = \underbrace{\begin{bmatrix} D_1 & \\ & D_2 \end{bmatrix}}_{=D, \text{ diagonalna}} + b_m v v^T.$$

Jeśli $\lambda \notin \sigma(D)$, to wartości własne λ macierzy $D + b_m v v^T$ spełniają równanie

$$f(\lambda) \equiv 1 + b_m \sum_{j=1}^N \frac{v_j^2}{d_j - \lambda} = 0.$$

Ostateczny koszt: $\mathcal{O}(N^3)$ z małą stałą.

144

Metoda dziel i rządź

Pełne zadanie własne dla macierzy **symetrycznej** $A = Q \Lambda Q^T$

Zakładamy, że A już została sprowadzona do postaci trójdzielnej.

$$A = \begin{bmatrix} a_1 & b_1 & & & \\ b_1 & a_2 & \ddots & & \\ & \ddots & \ddots & b_{N-1} & \\ & & b_{N-1} & a_N \end{bmatrix} = \begin{bmatrix} T_1 & \\ & T_2 \end{bmatrix} + b_m u u^T,$$

gdzie T_1, T_2 trójdzielne i symetryczne, $m \approx N/2$.

$$u = e_m + e_{m+1} = [0, \dots, 0, 1, 1, 0, \dots]^T$$

$b_m u u^T$ ma tylko cztery niezerowe elementy, każdy równy b_m .

143

Deflacja

Stwierdzenie

Jeśli

$$A = \begin{bmatrix} T_{11} & T_{12} \\ & T_{22} \end{bmatrix}$$

i T_{11}, T_{22} są kwadratowe, to

$$\sigma(A) = \sigma(T_{11}) \cup \sigma(T_{22}).$$

Często T_{22} jest rozmiaru 1×1 .

Dowód: $\det(A - \lambda I) = \det(T_{11} - \lambda I) \det(T_{22} - \lambda I)$. \square

Wniosek

Jeśli znamy wartości własne T_{22} , wystarczy znaleźć wartości własne T_{11} .

145

Pełne zadanie własne dla macierzy **niesymetrycznej**

```

 $A_1 = A;$ 
for  $k = 1 : \text{stop}$ 
    wykonaj rozkład  $A_k = Q_k R_k;$ 
     $A_{k+1} = R_k \cdot Q_k;$ 
end

```

Zachodzi $\sigma(A_k) = \sigma(A)$, bo $A_{k+1} = \underbrace{R_k}_{=Q_k^T A_k} \cdot Q_k = Q_k^T A_k Q_k$.

146

Metoda QR z przesunięciem (idea)

```

 $A_1 = A;$ 
for  $k = 1 : \text{stop}$ 
    wybierz szybką przesunięcie  $\sigma_k$ 
    wykonaj rozkład  $A_k - \sigma_k I = Q_k R_k;$ 
     $A_{k+1} = R_k \cdot Q_k + \sigma_k I;$ 
end

```

Najprostsza strategia³: $\sigma_k = a_{nn}$, a potem deflacja.

Koszt wyznaczenia wszystkich wektorów i wartości własnych jest rzędu $O(N^3)$ ze stałą równą około 30.

AFAIK nie jest znana „doskonała strategia”, gwarantująca zbieżność dla każdej macierzy.

³Nie zawsze skuteczna; są lepsze.

148

Twierdzenie (O zbieżności bazowej metody QR)

Niech wartości własne $A \in \mathbb{R}^{N \times N}$ spełniają $|\lambda_1| > \dots > |\lambda_N| > 0$ oraz macierz $T = [v_1, \dots, v_N]$ o kolumnach v_i złożonych z kolejnych wektorów własnych A ma taką własność, że T^{-1} ma rozkład LU , $T^{-1} = LU$.

Wtedy w metodzie QR ciąg macierzy Q_k jest zbieżny do macierzy diagonalnej, a ciąg A_k ma podciąg zbieżny do macierzy trójkątnej, której elementy diagonalne u_{ii} są równe λ_i dla $i = 1, \dots, N$.

147

Wrażliwość wartości własnych

Twierdzenie (Bauera–Fikego)

Jeśli A jest diagonalizowalna, $X^{-1}AX = D = \text{diag}(\lambda_1, \dots, \lambda_N)$ i μ jest wartością własną $A + \Delta$, to

$$\min_i |\mu - \lambda_i| \leq \text{cond}_2(X) \|\Delta\|_2.$$

Dowód: Na tablicy. \square

Wniosek

Jeśli $A = A^T$, to w sytuacji powyżej $\min_i |\mu - \lambda_i| \leq \|\Delta\|_2$.

Twierdzenie (Weyla)

Jeśli $A = A^T$ i $\Delta = \Delta^T$ oraz

$\lambda_1 \geq \dots \geq \lambda_N - w.wt A$, $\tilde{\lambda}_1 \geq \dots \geq \tilde{\lambda}_N - w.wt A + \Delta$, to

$$|\lambda_i - \tilde{\lambda}_i| \leq \|\Delta\|_2.$$

149

Lokalizacja wartości własnych

Stwierdzenie

Jeśli $\lambda \in \mathbb{C}$ — w.wł macierzy A , to $|\lambda| \leq \|A\|$, gdzie $\|A\|$ — norma indukowana przez normę wektorową.

Twierdzenie (Gerszgorina, o lokalizacji widma macierzy)

Wartości własne macierzy A leżą w $\bigcup_{i=1}^N K_i$, gdzie

$$K_i = \{z \in \mathbb{C} : |z - a_{ii}| \leq \sum_{j \neq i} |a_{ij}|\}.$$

150

Bazy wielomianowe

Wielomian stopnia co najwyżej N :

$$p(x) = \sum_{i=0}^N a_i x^i.$$

- wszystkie wielomiany stopnia $\leq N$ tworzy przestrzeń liniową wymiaru $N + 1$, oznaczmy ją P_N .
- jeśli ϕ_0, \dots, ϕ_N są bazą P_N , to każdy $p \in P_N$ daje się zapisać

$$p(x) = \sum_{i=0}^N \alpha_i \phi_i(x).$$

151

Interpolacja wielomianowa

Bazy wielomianowe — przykłady

$$p(x) = \sum_{i=0}^N \alpha_i \phi_i(x).$$

- „naturalna”: $\phi_i(x) = x^i$;
- Newtona:

$$\phi_0(x) = 1, \quad \phi_i(x) = \prod_{j < i} (x - x_j),$$

gdzie x_0, \dots, x_{N-1} — zadane punkty;

- Lagrange'a:

$$l_i(x) = \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}$$

gdzie x_0, \dots, x_N — zadane i różne punkty;

- Są też inne **użyteczne** bazy, o nich — innym razem

Odpowiednią dać rzeczy bazę!...

152

Obliczanie wartości wielomianu

Algorytm zależy od wyboru bazy!

W bazie Newtona (i naturalnej też):

Algorytm 5 (Hornera) Obliczanie wartości wielomianu w bazie Newtona $w = p(x) = a_0 + a_1(x - x_0) + \dots + a_N(x - x_0) \cdots (x - x_{N-1})$

```
w = a_N;  
for k = N - 1 downto 0  
    w = w · (x - x_k) + a_k;  
end  
return w
```

Koszt: $\mathcal{O}(N)$

153

Obliczanie wartości wielomianu

W bazie Lagrange'a:

$$p(x) = \sum_{i=0}^N a_i \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}$$

Jeśli robisz to źle, koszt będzie kwadratowy...

Postać barycentryczna:

$$p(x) = \sum_{i=0}^N a_i \prod_{j \neq i} \frac{x - x_j}{x_i - x_j} = \begin{cases} a_i & \text{gdy } x = x_i, \\ \phi_{N+1}(x) \cdot \sum_{i=0}^N \frac{a_i}{x - x_i} \beta_i, \end{cases}$$

gdzie

$$\phi_{N+1}(x) = \prod_{i=0}^N (x - x_i), \quad \beta_i = \prod_{j \neq i} \frac{1}{x_i - x_j}$$

Współczynniki β_i obliczamy w fazie *precomputing*.

Bonus: dla niektórych ważnych typów węzłów można to zrobić całkiem tanio.

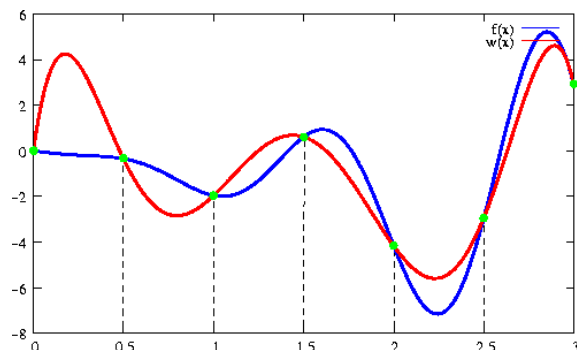
154

Interpolacja wielomianowa

Problem (interpolacja Lagrange'a)

Dla zadanych (parami różnych) węzłów interpolacji x_0, \dots, x_N i wartości $f(x_0), \dots, f(x_N)$ znaleźć wielomian $p \in P_N$ taki, że

$$p(x_i) = f(x_i), \quad i = 0, \dots, N$$



155

Interpolacja Lagrange'a

Twierdzenie

Zadanie interpolacji Lagrange'a ma jednoznaczne rozwiązanie.

Dowód: Niech $p(\cdot) = \sum_{j=0}^N c_j \varphi_j(\cdot)$, gdzie

$$P_N = \text{lin}\{\varphi_0, \varphi_1, \dots, \varphi_n\}.$$

Z warunków interpolacji,

$$p(x_i) = \sum_{j=0}^N c_j \varphi_j(x_i) = f(x_i), \quad 0 \leq i \leq N.$$

Jest to układ $N + 1$ równań liniowych z $N + 1$ niewiadomymi c_j . Ma on jednoznaczne rozwiązanie wtw wektor zerowy jest jedynym rozwiązaniem układu jednorodnego.

Ale jedyny wielomian $\text{st} \leq N$, który zeruje się w $N + 1$ punktach, musi być zerowy. \square

156

Wyznaczenie WIL

W bazie naturalnej, tzn. $w(x) = \sum_i a_i \cdot x^i$??

$$\underbrace{\begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^N \\ 1 & x_1 & x_1^2 & \cdots & x_1^N \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & x_N^2 & \cdots & x_N^N \end{bmatrix}}_{\text{macierz Vandermonde'a}} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_N) \end{bmatrix}$$

- gęsta, niesymetryczna, $N \times N$
- koszt GEPP: $\mathcal{O}(N^3)$
- patologicznie źle uwarunkowana

W innej bazie będzie lepiej!

157

Wyznaczenie WIL

W bazie Newtona, tzn.

$$p(x) = \sum_i b_i \cdot (x - x_0) \cdots (x - x_{i-1})$$

Definicja (Różnice dzielone)

$$f(t_0, t_1, \dots, t_s) = \frac{f(t_1, t_2, \dots, t_s) - f(t_0, t_1, \dots, t_{s-1})}{t_s - t_0}.$$

Twierdzenie (O różnicach dzielonych)

Jeśli p jest WIL w bazie Newtona j.w., to

$$b_i = f(x_0, x_1, \dots, x_i), \quad 0 \leq i \leq n.$$

158

Algorytm różnic dzielonych wyznaczania WIL

$$\begin{array}{ccccccc} x_0 & f(x_0) & & & & & \\ x_1 & f(x_1) & f(x_0, x_1) & & & & \\ x_2 & f(x_2) & f(x_1, x_2) & f(x_0, x_1, x_2) & & & \\ \vdots & \vdots & \vdots & \vdots & \ddots & & \\ x_N & f(x_N) & f(x_{N-1}, x_N) & f(x_{N-2}, x_{N-1}, x_N) & \cdots & f(x_0, x_1, \dots, x_N) \end{array}$$

Tabelkę wypełniamy kolejnymi kolumnami.

Odpowiadający algorytm można wykonać *in situ*, kosztem $\mathcal{O}(N^2)$

flopów.

159

Wyznaczenie WIL w bazie Lagrange'a

$$p(x) = \sum_{i=0}^N a_i \underbrace{\prod_{j \neq i} \frac{x - x_j}{x_i - x_j}}_{=l_i(x)}$$

Rozwiązaniem ZIL jest

$$p(x) = \sum_{i=0}^N f(x_i) \cdot l_i(x).$$

Koszt „zerowy”, ale pamiętajmy o koszcie obliczenia wartości p : alg. barycentr. wymaga precomputingu o koszcie $\mathcal{O}(N^2)$.

Twierdzenie (Highama)

Niech $x_i, f(x_i), x$ będą l. maszynowymi. Obliczona alg. barycentr. w fl wartość $\tilde{p}(x)$ jest dokładną wartością WIL dla $\tilde{f}(x_i) = f(x_i) \cdot (1 + \delta_i)$, gdzie $|\delta_i| \lesssim 5(N+1)\nu$.

160

Błąd interpolacji Lagrange'a

Twierdzenie (Postać błędu interpolacji)

Niech p będzie WIL dla f w punktach x_0, \dots, x_N .

- Dla dowolnego $\bar{x} \in \mathbb{R}$

$$f(\bar{x}) - p(\bar{x}) = f(x_0, x_1, \dots, x_N, \bar{x}) \cdot \phi_{N+1}(\bar{x}). \quad (1)$$

- Ponadto, jeśli $f \in C^{(N+1)}[a, b]$, gdzie $[a, b]$ zawiera x_0, \dots, x_N, \bar{x} , to istnieje $\xi = \xi(\bar{x}) \in (a, b)$ t. że

$$f(\bar{x}) - p(\bar{x}) = \frac{f^{(N+1)}(\xi)}{(N+1)!} \cdot \phi_{N+1}(\bar{x}).$$

$$\phi_{N+1}(x) = (x - x_0) \cdots (x - x_N).$$

161

Uwarunkowanie ZIL

Jak odporne jest ZIL na zaburzenie wartości w węzłach? Niech

- p_f — WIL dla f ,
- $p_{\tilde{f}}$ — WIL dla \tilde{f} ,

oba oparte na tych samych węzłach x_0, \dots, x_N .

Twierdzenie (uwarunkowanie ZIL)

Jeśli $|f(x_i) - \tilde{f}(x_i)| \leq \epsilon |f(x_i)|$ dla $i = 0, \dots, N$, to dla dowolnego x

$$\frac{|p_f(x) - p_{\tilde{f}}(x)|}{|p_f(x)|} \leq \text{cond}_{rel}(x, f) \cdot \epsilon,$$

gdzie

$$\text{cond}_{rel}(x, f) = \frac{\sum_{j=0}^N |l_j(x)| \cdot |f(x_j)|}{|p_f(x)|} \geq 1.$$

$\max_{x \in [a, b]} \sum_{j=0}^N |l_j(x)| \leftarrow$ stała Lebesgue'a dla x_0, \dots, x_N na $[a, b]$. 162

Interpolacja wielomianowa: uogólnienie

Problem (interpolacja Hermite'a)

Dla zadanych (parami różnych) węzłów interpolacji x_0, \dots, x_n i odp. wartości pochodnych f , znaleźć wielomian $p \in P_N$ taki, że

$$p^{(k)}(x_i) = f^{(k)}(x_i) \quad \text{dla} \quad k = 0 \dots m_i - 1, \quad i = 0 \dots n,$$

przy czym $N + 1 = m_0 + \dots + m_n$.

„Węzeł x_i ma krotność m_i ”

Twierdzenie

ZIH ma jednoznaczne rozwiązanie.

163

Interpolacja Hermite'a w bazie Newtona

Baza Newtona uwzględniająca krotność węzłów:

$$\phi_0(x) = 1$$

$$\phi_1(x) = (x - x_0), \quad \phi_2(x) = (x - x_0)^2, \quad \dots, \quad \phi_{m_0}(x) = (x - x_0)^{m_0},$$

$$\phi_{m_0+1}(x) = (x - x_0)^{m_0}(x - x_1), \quad \dots, \quad \phi_{m_0+m_1}(x) = (x - x_0)^{m_0}(x - x_1)^{m_1},$$

\vdots

$$\phi_{N+1}(x) = (x - x_0)^{m_0}(x - x_1)^{m_1} \cdots (x - x_n)^{m_n}$$

Twierdzenie

WIH jest postaci $p(x) = \sum_{k=0}^N f(x_0, \dots, x_k) \phi_k(x)$, gdzie

$$f(t_0, \dots, t_k) = \begin{cases} \frac{f^{(k)}(x_0)}{k!} & \text{gdy } t_0 = \dots = t_k, \\ \frac{f(t_1, \dots, t_k) - f(t_0, \dots, t_{k-1})}{t_k - t_0} & \text{w p.p.} \end{cases}$$

164

Błąd interpolacji Hermite'a

Twierdzenie (Postać błędu interpolacji)

Niech p będzie WIH dla f w punktach x_0, \dots, x_n o łącznej krotności $N+1$. Jeśli $f \in C^{(N+1)}[a, b]$, gdzie $[a, b]$ zawiera x_0, \dots, x_n, \bar{x} , to istnieje $\xi = \xi(\bar{x}) \in (a, b)$ t. że

$$f(\bar{x}) - p(\bar{x}) = \frac{f^{(N+1)}(\xi)}{(N+1)!} \cdot \phi_{N+1}(\bar{x}).$$

$$\phi_{N+1}(x) = (x - x_0)^{m_0} \cdots (x - x_N)^{m_n}.$$

165

Splajny

- Wymiar przestrzeni splajnów.
- Zadanie interpolacji.
- Warunki brzegowe dla splajnów nieparzystego stopnia: hermitowskie, periodyczne, naturalne.
- Splajny kubiczne: wyznaczanie w reprezentacji kawałkami wielomianowej w przypadku warunków brzegowych naturalnych.
- Twierdzenie o błędzie aproksymacji splajnu kubicznego interpolacyjnego (dowód $O(h^2)$ dla naturalnego; $O(h^4)$ dla hermitowskiego bez dowodu).
- B-splajny i ich wykorzystanie

166

Splajny

Splajn, czyli funkcja sklejana

Niech $a = x_0 < x_1 < \cdots < x_n = b$.

Definicja

$s : [a, b] \rightarrow \mathbb{R}$ jest splajnem rzędu r opartym na węzłach $\{x_j\}$, jeśli spełnione są następujące dwa warunki:

- (i) s jest wielomianem stopnia co najwyżej r na każdym z przedziałów $[x_{j-1}, x_j]$,
- (ii) $s \in C^{r-1}[a, b]$.

$C^k(a, b) := \{f : (a, b) \rightarrow \mathbb{R} \mid f \text{ ma przynajmniej } k \text{ pochodnych ciągłych na } (a, b)\}.$

Przestrzeń liniową funkcji sklejanych rzędu r opartych na (ustalonych) węzłach x_j będziemy oznaczać przez \mathcal{S}_r .

167

Szczegóły można znaleźć w podręczniku Kincaida i Cheney'a:

rozdz. 6.4, 6.5 (fragm.) i 6.6 (fragm.).

(Wykład o splajnach odbył się bez slajdów, a na wykładzie o B-splajnach było zastępstwo)

168

Węzły równoodległe: $x_i = a + i \cdot h$, gdzie $h = (b - a)/n$.

Twierdzenie (o błędzie aproksymacji hermitowskim kubicznym splajnem interpolacyjnym)

Niech s będzie kubicznym splajnem interpolacyjnym dla $f \in C^4[a, b]$, z hermitowskimi warunkami brzegowymi:

$$s'(a) = f'(a), \quad s'(b) = f'(b).$$

Istnieją stałe c_r niezależne od h i n takie, że

$$\|f^{(r)} - s^{(r)}\|_\infty \leq c_r \cdot h^{4-r} \cdot \|f^{(4)}\|_\infty, \quad r = 0, 1, 2, 3.$$

W szczególności,

$$\|f - s\|_\infty \leq \frac{5}{384} \cdot h^4 \cdot \|f^{(4)}\|_\infty.$$

$$\|g\|_\infty := \sup_{x \in [a, b]} |g(x)|.$$

170

Węzły równoodległe: $x_i = a + i \cdot h$, gdzie $h = (b - a)/n$.

Twierdzenie (o błędzie aproksymacji naturalnym kubicznym splajnem interpolacyjnym)

Niech s będzie naturalnym kubicznym splajnem interpolacyjnym dla $f \in C^4[a, b]$. Istnieje stała c niezależna od h i n taka, że

$$\|f - s\|_\infty \leq c \cdot h^2 \cdot \|f^{(4)}\|_\infty.$$

$$\|g\|_\infty := \sup_{x \in [a, b]} |g(x)|.$$

169

Aproksymacja średniokwadratowa

Aproksymacja

- Sformułowanie zadania najlepszej aproksymacji ogólnie i konkretniej - dla funkcji ciągłej w zadanej normie.
- Przykłady norm: 1, 2, ∞ , wersja $L^p(a, b)$ i wersja dyskretna (skupiona w węzłach).
- Przykłady pni sk.wym: wielomiany, wielomiany trygonometryczne, sumy funkcji wykładniczych.
- Aproksymacja sk.wym. w p-ni unitarnej: warunek ortogonalności i algorytm wyznaczenia ENA przez rozwiązanie układu z macierzą Grama. Przypadek bazy ortogonalnej.
- Wielomiany ortogonalne: formuła 3-członowa, przykłady (bez dowodu ortogonalności): wielomiany Legendre'a, Hermite'a, Czebyszewa.

171

Aproksymacja jednostajna

Aproksymacja średniokwadratowa — ciąg dalszy

Szczegóły można znaleźć w podręczniku Kincaida i Cheney'a:

rozd. 6.8

(Wykład odbył się bez slajdów.)

172

Przypomnienie

Twierdzenie (o istnieniu ENA)

Niech F będzie przestrzenią liniową i niech $V \subset F$ — podprzestrzeń skończonego wymiaru. Dla każdego $f \in F$ istnieje $v^* \in V$ taki, że

$$\|f - v^*\| \leq \|f - v\| \quad \forall v \in V.$$

v^* jest ENA dla f .

173

Zadanie najlepszej aproksymacji jednostajnej wielomianami

- P_N — przestrzeń wielomianów stopnia co najwyżej N .
- $\|f\|_\infty = \sup_{x \in [a,b]} |f(x)|$.

Problem

Dla zadanej funkcji $f \in C[a, b]$ znaleźć wielomian $p^* \in P_N$ taki, że

$$\|f - p^*\|_\infty \rightarrow \min!$$

(to znaczy: $\|f - p^*\|_\infty \leq \|f - p\|_\infty$ dla każdego $p \in P_N$).

Jak wiemy z ogólnego twierdzenia o istnieniu ENA, taki p^* **istnieje**.

174

Dowód twierdzenia o alternansie

- istnienie — z ogólnej teorii, bo P_N jest liniowa skończonego wymiaru
- charakteryzacja przez alternans — na tablicy
- jednoznaczność — nie wprost:

Oznaczmy $E_N = \|f - p^*\|_\infty$. Gdyby istniały dwa różne optymalne wielomiany p^* i q^* , to wtedy wielomian $w = \frac{1}{2}(p^* + q^*)$ też jest ENA dla f :

$$\|f - w\|_\infty = \left\| \frac{1}{2}(f - p^*) + \frac{1}{2}(f - q^*) \right\|_\infty \leq \frac{1}{2}\|f - p^*\|_\infty + \frac{1}{2}\|f - q^*\|_\infty = E_N.$$

176

Alternans

Twierdzenie (o alternansie)

Dla zadanej $f \in C[a, b]$ istnieje dokładnie jeden $p^* \in P_N$ będący ENA dla f w sensie normy supremum. Co więcej,

$$p^* \in P_N \text{ jest ENA dla } f \iff \text{dla } f \text{ i } p^* \text{ istnieje alternans.}$$

Definicja (alternans)

Niech $p^* \in P_N$. Zbiór $N + 2$ punktów

$a \leq \xi_0 < \xi_1 < \dots < \xi_{N+1} \leq b$ takich, że $e = f - p^*$ spełnia

- $|e(\xi_i)| = \|e\|_\infty$ dla $i = 0, \dots, N + 1$,
- $e(\xi_i) = -e(\xi_{i+1})$ dla $i = 0, \dots, N$,

nazywamy **alternansem** dla f i p^* .

175

Dowód twierdzenia o alternansie — c.d.

Skoro w jest ENA, to dla f i w istnieje alternans, więc m.in.:

$$|f(\xi_i) - w(\xi_i)| = E_N, \quad i = 0, \dots, N + 1.$$

Ale wtedy

$$\frac{1}{2} \underbrace{(f - p^*)(\xi_i)}_{|\cdot| \leq E_N} + \frac{1}{2} \underbrace{(f - q^*)(\xi_i)}_{|\cdot| \leq E_N} = E_N,$$

czyli

$$(f - p^*)(\xi_i) = (f - q^*)(\xi_i) = \pm E_N.$$

Skoro więc p^* i q^* są sobie równe w $N + 2$ różnych punktach, to musi być, że $p^* \equiv q^*$. \square

177

Zbieżność najlepszej aproksymacji jednostajnej

Twierdzenie (Jacksona)

Niech r będzie ustalone takie, że $f \in C^r[-1, 1]$ oraz

$$\exists M_r > 0 \quad |f^{(r)}(x) - f^{(r)}(y)| \leq M_r |x - y| \quad \forall x, y \in [-1, 1].$$

Jeśli $p^* \in P_N$ jest ENA dla f i $N > r$, to

$$\|f - p^*\|_\infty \leq c_r \frac{M_r}{N^{r+1}},$$

gdzie $c_r > 0$ nie zależy od f .

178

Wyznaczenie ENA jednostajnej

- W ogólnym przypadku jest to zadanie trudne (nieliniowe)
- Algorytm Remeza wyznaczania ENA: iteracyjne poprawianie alternansu. W granicy dostajemy alternans i wielomian optymalny.
(Tak, to znaczy, że liczba iteracji $\rightarrow \infty$)

Pomysł: zastąpić ENA czymś niewiele gorszym, ale za to znacznie tańszym do wyznaczenia.

179

Aproksymacja jednostajna a interpolacja wielomianowa

Przypomnienie:

Twierdzenie (Postać błędu interpolacji)

Niech p będzie WIL dla $f \in C^{(N+1)}[a, b]$ w punktach x_0, \dots, x_N w $[a, b]$. Dla $x \in [a, b]$ istnieje $\xi \in (a, b)$ t. że

$$f(x) - p(x) = \frac{f^{(N+1)}(\xi)}{(N+1)!} \cdot \underbrace{(x - x_0) \cdots (x - x_N)}_{=: \phi_{N+1}(x)},$$

Wniosek

$$\|f - p\|_\infty \leq \frac{\|f^{(N+1)}\|_\infty}{(N+1)!} \cdot \|\phi_{N+1}\|_\infty.$$

Pytanie: jak dobrać węzły interpolacji, by zminimalizować w/w oszacowanie?

180

Węzły Czebyszewa

Niech $[a, b] = [-1, 1]$. Szukamy x_0, \dots, x_N t. że

$$\|\phi_{N+1}\|_\infty \rightarrow \min!$$

czyli

$$\max_{x \in [-1, 1]} |\phi_{N+1}(x)| \rightarrow \min!$$

czyli

$$\max_{x \in [-1, 1]} |(x - x_0) \cdots (x - x_N)| \rightarrow \min!$$

Okazuje się, że rozwiązaniem tego zadania są miejsca zerowe wielomianu Czebyszewa T_{N+1} ,

$$x_i = \cos\left(\frac{2i+1}{2N+2}\pi\right), \quad i = 0, \dots, N.$$

181

Wielomiany Czebyszewa

Formuła trójkątnowa

$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x),$$

$$T_0(x) = 1, \quad T_1(x) = x.$$

$$T_0(x) = 1$$

$$T_1(x) = x$$

$$T_2(x) = 2x^2 - 1$$

$$T_3(x) = 4x^3 - 3x$$

$$T_4(x) = 8x^4 - 8x^2 + 1$$

⋮

$$T_n(x) = 2^{n-1}x^n + \dots \text{niższe potęgi} \dots$$

182

Węzły Czebyszewa

Wniosek

$$\max_{x \in [-1, 1]} |(x - x_0) \cdots (x - x_N)| \rightarrow \min!$$

wtedy i tylko wtedy, gdy $\{x_i\}$ — miejsca zerowe T_{N+1} .

Są to **węzły Czebyszewa**.

Twierdzenie

Jeśli p jest WIL dla $f \in C^{N+1}[-1, 1]$ opartym na węzłach Czebyszewa, to

$$\|f - p\|_{\infty} \leq \frac{\|f^{(N+1)}\|_{\infty}}{2^N(N+1)!}.$$

184

Wielomiany Czebyszewa — własności

- Dla $x \in [-1, 1]$, $T_n(x) = \cos n\theta$, gdzie $\theta = \arccos x$.

- (Ortogonalność) Jeśli $i \neq j$, to

$$\int_{-1}^1 T_i(x) \cdot T_j(x) \frac{1}{\sqrt{1-x^2}} dx = 0.$$

- Wszystkie miejsca zerowe T_n są jednokrotne i w $[-1, 1]$:

$$T_n(x_i) = 0 \iff x_i = \cos\left(\frac{2i-1}{2n}\pi\right), \quad i = 1, \dots, n.$$

- $|T_n(x)| \leq 1$ dla $x \in [-1, 1]$ oraz

$$T_n(y_j) = (-1)^j \iff y_j = \cos\left(\frac{j}{n}\pi\right), \quad j = 0, \dots, n.$$

- (własność minimaksu) Wśród wszystkich wielomianów postaci

$$x^n + \dots \text{niższe potęgi} \dots,$$

$$\text{najmniejszą normę } \|\cdot\|_{\infty} \text{ na } [-1, 1] \text{ ma wielomian } \frac{T_n}{2^{n-1}}.$$

183

Aproksymacja jednostajna a interpolacja wielomianowa

Twierdzenie

Niech $f \in C[a, b]$ oraz

- $p^* \in P_N$ będzie ENA dla f w normie supremum na $[a, b]$
- $p \in P_N$ będzie WIL dla f , opartym na $a \leq x_0 < \dots < x_N \leq b$.

Wtedy

$$\|f - p^*\|_{\infty} \leq \|f - p\|_{\infty} \leq (1 + \Lambda_N) \cdot \|f - p^*\|_{\infty},$$

gdzie $\Lambda_N = \|\lambda_N\|_{\infty}$ (stała Lebesgue'a), gdzie

$$\lambda_N(x) := \sum_{i=0}^N |l_i(x)| \quad \leftarrow l_i : \text{funkcje bazy Lagrange'a}.$$

185

Stała Lebesgue'a

$\Lambda_N = \|\lambda_N\|_\infty$ (stała Lebesgue'a).

Twierdzenie

- Dla dowolnego układu węzłów w $[-1, 1]$,

$$\Lambda_N \geq \frac{2}{\pi} \ln(N+1) + 0.5215 \dots$$

- Dla węzłów równoodległych w $[-1, 1]$,

$$\Lambda_N > \frac{2^{N-2}}{N^2}.$$

- Dla węzłów Czebyszewa w $[-1, 1]$,

$$\Lambda_N \leq \frac{2}{\pi} \ln(N+1) + 1.$$

186

Równania nieliniowe

Interpolacja w węzłach Czebyszewa jest prawie optymalna

Wniosek

Jeśli

- $p^* \in P_N$ jest ENA dla f w normie supremum na $[a, b]$
- $p \in P_N$ jest WIL dla f , opartym na (przeskalowanych) **węzłach Czebyszewa** w $[a, b]$,

to

$$\|f - p\|_\infty \leq C_N \cdot \|f - p^*\|_\infty,$$

gdzie $C_N < 3$ dla $N \leq 20$; $C_N < 10$ dla $N \leq 10^6$.

187

Równania nieliniowe

Problem (równanie skalarne)

Dana jest $f : \mathbb{R} \supset (a, b) \rightarrow \mathbb{R}$. Znaleźć $x \in (a, b)$ taki, że

$$f(x) = 0.$$

Problem (układ równań nieliniowych)

Dana jest $f : \mathbb{R}^N \supset D \rightarrow \mathbb{R}^N$, gdzie D jest otwarty i niepusty. Znaleźć $x \in D$ taki, że

$$f(x) = 0.$$

Źródła kłopotów:

- rozwiązanie może nie istnieć
- rozwiązań może być wiele (czy chcemy znaleźć wszystkie?)
- f może być zadana przez czarną skrzynkę (*oracle*)
- N może być duże, f może być kosztowna

188

Uwarunkowanie zadania wyznaczania miejsca zerowego

Problem

Mamy

$$f(x^*) = 0, \quad \tilde{f}(\widetilde{x^*}) = 0,$$

przy czym $\|f - \tilde{f}\|_\infty \leq \epsilon$. Jak można oszacować $\|x^* - \widetilde{x^*}\|$?

Przypomnienie (Zadanie obliczenia $x = P(y)$)

Gdy P różniczkowalna w y , to

$$\text{cond}_{abs}(P, y) = \|P'(y)\| \text{ oraz } \|x - \tilde{x}\| \lesssim \text{cond}_{abs}(P, y) \cdot \|y - \tilde{y}\|.$$

U nas $x^* = f^{-1}(0) =: P(0)$. Jeśli f jest klasy C^1 w otoczeniu x^* oraz $f'(x^*)$ jest nieosobliwa, to $P'(0) = [f'(x^*)]^{-1}$, zatem

$$\|x^* - \widetilde{x^*}\| \lesssim \underbrace{\|[f'(x^*)]^{-1}\|}_{\text{cond}(f^{-1}, 0)} \cdot \epsilon.$$

189

Metoda bisekcji (dla równania skalarnego)

Niech $f \in C[a, b]$ taka, że $f(a) \cdot f(b) < 0$. (Więc na pewno ma miejsce zerowe w (a, b))

$$x = (a + b)/2$$

while $|b - a| > \epsilon$ **do**

if $f(x) == 0$ **then return** x

▷ trafiliśmy!

end if

if $\text{sign } f(x) \neq \text{sign } f(a)$ **then**

$$b = x$$

▷ rozwiązanie w lewej połówce

else

$$a = x$$

▷ rozwiązanie w prawej połówce

end if

$$x = (a + b)/2$$

end while

return x

190

Metoda bisekcji — szybkość zbieżności

Twierdzenie

$f \in C[a, b]$ taka, że $f(a) \cdot f(b) < 0$.

Oznaczmy $[a_n, b_n]$ — przedział lokalizujący zero w n -tej iteracji.

Wtedy

$$|b_n - a_n| = \frac{1}{2} |b_{n-1} - a_{n-1}| = \frac{1}{2^n} |b - a|.$$

Ponadto $x_n = \frac{1}{2}(a_n + b_n)$ spełnia

$$|x_n - x^*| \leq \frac{1}{2} |b_n - a_n|.$$

Wniosek

Aby $|x_n - x^*| \leq \epsilon$, wystarczy, że $n + 1 \geq \log_2 \left(\frac{|b - a|}{\epsilon} \right)$.

191

Szybkość zbieżności metody iteracyjnej

Ciąg $x_n \rightarrow x^* \in \mathbb{R}^N$:

- jest zbieżny **z rzędem** (co najmniej) $p > 1$, jeśli

$$\exists C \geq 0 \quad \|x_{n+1} - x^*\| \leq C \|x_n - x^*\|^p \quad \forall n = 0, 1, \dots$$

($p = 2$: zbieżność kwadratowa, $p = 3$: zbieżność sześcienna/kubiczna)

- jest zbieżny (co najmniej) **liniowo**, jeśli

$$\exists \gamma \in [0, 1) \quad \|x_{n+1} - x^*\| \leq \gamma \|x_n - x^*\| \quad \forall n = 0, 1, \dots$$

- jest **r-zbieżny** z rzędem p (odp. liniowo), jeśli

$$\|x_n - x^*\| \leq r_n$$

dla pewnego ciągu r_n zbieżnego z rzędem p (odp. liniowo) do zera.

192

Szybkość zbieżności metody iteracyjnej

Wniosek

Metoda bisekcji jest zbieżna r -liniowo z ilorazem $\frac{1}{2}$.

Przykład (Jak wyglądają różne rodzaje zbieżności)

Startujemy z $\|x_0 - x^*\| = 0.5$ i...

Szybkość	Iteracja (n)				
	1	2	3	4	5
liniowa $\gamma = 1/2$	0.25	0.13	0.06	0.03	0.02
liniowa $\gamma = 0.9$	0.45	0.41	0.36	0.33	0.30
kwadratowa ⁴ ($p = 2$)	0.25	$6 \cdot 10^{-2}$	$4 \cdot 10^{-3}$	$2 \cdot 10^{-5}$	$2 \cdot 10^{-10}$
kubiczna ⁴ ($p = 3$)	0.13	$2 \cdot 10^{-3}$	$7 \cdot 10^{-9}$	$4 \cdot 10^{-25}$	$7 \cdot 10^{-74}$

⁴Przyjmując, że $\|x_{n+1} - x^*\| = \|x_n - x^*\|^p$

193

Zadanie punktu stałego

Problem

Niech $\Phi : \mathbb{R}^N \supset D \rightarrow \mathbb{R}^N$. Znaleźć $x^* \in D$ taki, że

$$x^* = \Phi(x^*).$$

x^* to **punkt stały** Φ .

To zadanie zawsze jest równoważne jakiemuś zadaniu wyznaczania zera funkcji, np. dla

$$f(x) = x - \Phi(x).$$

194

Metoda Banacha (iteracja prosta)

Twierdzenie (Banacha, o kontrakcji)

Niech $\Phi : D \rightarrow D$ będzie **kontrakcją**:

$$\exists L \in [0, 1) \quad \|\Phi(x) - \Phi(y)\| \leq L \|x - y\| \quad \forall x, y \in D,$$

gdzie $D \subset \mathbb{R}^N$ — domknięty, niepusty. Wtedy

- istnieje dokładnie jeden punkt stały Φ , $x^* = \Phi(x^*)$;
- iteracja Banacha

$$x_{n+1} = \Phi(x_n)$$

jest zbieżna liniowo do x^* z dowolnego $x_0 \in D$:

$$\|x_{n+1} - x^*\| \leq L \|x_n - x^*\|.$$

195

Szybkość zbieżności metody Banacha

Twierdzenie (maszynka do rzędu)

Założmy, że iteracja $x_{n+1} = \Phi(x_n)$ jest zbieżna do $x^* = \Phi(x^*) \in \text{int } D$, przy czym $\Phi \in C^p(D)$, $p \geq 1$, oraz

$$\Phi'(x^*) = \dots = \Phi^{(p-1)}(x^*) = 0.$$

Jeśli x_0 jest dostatecznie blisko x^* to

$$\exists C \quad \|x_{n+1} - x^*\| \leq C \|x_n - x^*\|^p.$$

196

Szybkość zbieżności metody Banacha — dowód

Dowód: $\underbrace{x_{n+1} - x^*}_{=e_{n+1}} = \Phi(x_n) - x^* = \underbrace{\Phi(x_n)}_{=\Phi(x^*+e_n)} - \Phi(x^*)$. Na mocy

wzoru Taylora (o ile e_n jest dostatecznie małe)

$$\Phi(x^* + e_n) = \Phi(x^*) + \sum_{k=1}^{p-1} \frac{1}{k!} \underbrace{\Phi^{(k)}(x^*)}_{=0} e_n^k + \frac{1}{p!} \Phi^{(p)}(x^*) e_n^p + R(x^*, e_n),$$

gdzie $\frac{\|R(x^*, e_n)\|}{\|e_n\|^p} \rightarrow 0$ dla $e_n \rightarrow 0$. Zatem jeśli $\|e_0\|$ jest dostatecznie mała, to

$$\|e_{n+1}\| \leq \frac{1}{p!} \|\Phi^{(p)}(x^*)\| \|e_n\|^p + C \|e_n\|^p.$$

Co więcej, dla $n \rightarrow \infty$,

$$\frac{\|e_{n+1}\|}{\|e_n\|^p} \rightarrow \frac{1}{p!} \|\Phi^{(p)}(x^*)\|.$$

□

197

Metoda Newtona

$$x_{n+1} = x_n - [f'(x_n)]^{-1} f(x_n)$$

Twierdzenie (o lokalnej zbieżności m. Newtona)

Niech $f : D \rightarrow \mathbb{R}^N$ oraz $D \subset \mathbb{R}^N$ otwarty i niepusty. Jeśli

- f jest różniczkowalna w D oraz

$$\exists L \quad \|f'(x) - f'(y)\| \leq L \|x - y\| \quad \forall x, y \in D,$$

- istnieje $x^* \in D$ t. że $f(x^*) = 0$ oraz $f'(x^*)$ jest nieosobliwa,

to dla x_0 dost. blisko x^* metoda Newtona jest zbieżna oraz

$$\exists C \quad \|x_{n+1} - x^*\| \leq C \|x_n - x^*\|^2 \quad n = 0, 1, \dots$$

198

Modyfikacje m. Newtona dla r-nia skalarnego: bez pochodnych

Idea: zamiast metody stycznych (Newtona): $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$ użyć metody przybliżonej

$$x_{n+1} = x_n - \frac{f(x_n)}{g_n}, \quad \text{gdzie } g_n \approx f'(x_n).$$

- Metoda siecznych (nie wymaga dodatk. obl. f , r-zbieżność rzędu $(1 + \sqrt{5})/2 \approx 1.618$)

$$g_n = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}$$

- Metoda Steffensena (wymaga 1 dodatk. obl. f), zb. rzędu 2)

$$g_n = \frac{f(x_n + f(x_n)) - f(x_n)}{f(x_n)}$$

- Metoda z ilorazem różnicowym (1 dod. obl. f , zb. liniowa)

$$g_n = \frac{f(x_n + h) - f(x_n)}{h}.$$

199

Implementacja wielowymiarowej metody Newtona

Metoda Newtona:

$$x_{n+1} = x_n - s,$$

wymaga obliczenia poprawki,

$$s = [f'(x_n)]^{-1} f(x_n).$$

Aby wyznaczyć poprawkę należy więc **rozwiązać układ równań** z macierzą $N \times N$,

$$f'(x_n) \cdot s = f(x_n).$$

200

Uproszczona metoda Newtona

Rozwiązywanie układu równań w każdym kroku kosztuje.

$$x_{n+1} = x_n - [f'(x_n)]^{-1} f(x_n)$$

Uproszczenie:

$$x_{n+1} = x_n - [f'(x_0)]^{-1} f(x_n)$$

Teraz wystarczy tylko raz wyznaczyć rozkład macierzy.

Metoda będzie jednak (lokalnie) zbieżna tylko liniowo.

201

Metoda z przybliżoną pochodną

Uogólnienie metody z ilorazem różnicowym dla r-nia skalarne.

$$x_{n+1} = x_n - [G_n]^{-1} f(x_n),$$

gdzie $G_n = [g_1, \dots, g_N]$ i każdą kolumnę przybliżamy ilorazem różnicowym:

$$g_i = \frac{1}{h_n} (f(x_n + h_n e_i) - f(x_n))$$

e_i — i -ty wektor jednostkowy.

h_n proporcjonalne do $\|f(x_n)\|$ pozwala zachować kwadratowy rząd (lokalnej) zbieżności.

Uwaga na dobór h_n :

- za duże — kiepskie przybliżenie pochodnej
- za małe — problemy z redukcją cyfr przy odejmowaniu

203

Przybliżona metoda Newtona

Rozwiązywanie układu równań w każdym kroku kosztuje.

$$x_{n+1} = x_n - [f'(x_n)]^{-1} f(x_n)$$

Dlaczego by nie wykorzystać metody iteracyjnej do rozwiązywania układu $f'(x_n)s = f(x_n)$?

$$x_{n+1} = x_n - s,$$

gdzie s spełnia residualne kryterium stopu

$$\|f'(x_n)s - f(x_n)\| \leq \eta_n \|f(x_n)\|.$$

η_n proporcjonalne do $\|f(x_n)\|$ pozwala zachować kwadratowy rząd (lokalnej) zbieżności.

202

Skalarne równania wielomianowe

Problem

Znaleźć $x \in \mathbb{C}$ takie, że

$$p(x) := a_N x^N + \dots a_1 x + a_0 = 0,$$

gdzie $a_i \in \mathbb{R}$ (lub \mathbb{C}) są zadane.

Dla wszystkich pierwiastków jedną z opcji jest wyznaczenie wszystkich wartości własnych macierzy stowarzyszonej:

$$C = \begin{bmatrix} -\frac{a_{N-1}}{a_N} & -\frac{a_{N-2}}{a_N} & \dots & -\frac{a_0}{a_N} \\ 1 & & & \\ & \ddots & & \\ & & 1 & \end{bmatrix}$$

Zachodzi bowiem: $\{\lambda \in \mathbb{C} : \lambda \text{ jest w. wł } C\} = \{x \in \mathbb{C} : p(x) = 0\}$.

204

Całkowanie

Kwadratury jednowymiarowe

Na wykładzie będziemy zajmować się wyłącznie całkami na odcinku:

Problem

Dla $f : [a, b] \rightarrow \mathbb{R}$ przybliżyć wartość

$$I(f) = \int_a^b f(x) dx$$

za pomocą kwadratury $Q(f)$,

$$Q(f) = \sum_{i=0}^n A_i f(x_i).$$

Kwadratury

Problem

Dla $f : \mathbb{R}^d \subset D \rightarrow \mathbb{R}$ przybliżyć wartość całki

$$I(f) = \int_D f(x) \rho(x) dx \quad (\rho \text{ to zadana waga})$$

za pomocą kwadratury $Q(f)$,

$$Q(f) = \sum_{i=0}^n A_i f(x_i).$$

Chodzi nam o wyznaczenie współczynników kwadratury A_i oraz węzłów kwadratury $x_i \in D$ takich, by $Q(f) \approx I(f)$ możliwie dobrze w pewnej klasie funkcji, $f \in F$.

Kwadratury — zamiana zmiennych

Całkę $I(f) = \int_a^b f(x) dx$ można sprowadzić do całki na zadanym odcinku (np. dla $[0, 1]$):

$$[a, b] \ni x = a + t \cdot (b - a) \text{ dla } t \in [0, 1]$$

i wtedy

$$I(f) = \int_0^1 f(a + t \cdot (b - a)) \cdot (b - a) dt = (b - a) \int_0^1 g(t) dt$$

gdzie $g(t) = f(a + t \cdot (b - a))$.

Kwadratura $\sum_i A_i g(t_i)$ dla $\int_0^1 g(t) dt$ rzędu r daje kwadraturę tego samego rzędu dla $\int_a^b f(x) dx$ z wagami $(b - a) \cdot A_i$ i węzłami $x_i = a + t_i \cdot (b - a)$.

Kwadratury interpolacyjne

Idea: funkcję f przybliżyć wielomianem interpolacyjnym p , a całkę z f — całką z p .

Niech x_0, \dots, x_n węzły interpolacji Lagrange'a oraz

$$p(x) = \sum_{i=0}^n f(x_i) \underbrace{l_i(x)}_{\text{baza Lagrange'a}}. \text{ Wtedy}$$

$$I(f) \approx Q(f) := \int_a^b p(x) = \int_a^b \sum_{i=0}^n f(x_i) l_i(x) = \sum_{i=0}^n \left(\underbrace{\int_a^b l_i(x) dx}_{=: A_i} \right) \cdot f(x_i)$$

A_i — wyznaczamy raz na zawsze (nie zależą od f) w precomputingu: umiemy obliczać całki z wielomianów.

Nie wszystkie kwadratury są interpolacyjne.

208

Kwadratury Newtona-Cotesa

To kwadratury interpolacyjne oparte na węzłach równoodległych, np.:

- prostokątów

$$Q^Q(f) = (b-a) \cdot f\left(\frac{a+b}{2}\right)$$

- trapezów

$$Q^T(f) = \frac{b-a}{2} \cdot (f(a) + f(b))$$

- parabol (Simpsona)

$$Q^S(f) = \frac{b-a}{6} \cdot \left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b) \right)$$

209

Błąd kwadratur interpolacyjnych

Twierdzenie

Jeśli $f \in C^{n+1}[a, b]$, a Q jest kwadraturą interpolacyjną opartą na węzłach $x_0, \dots, x_n \in [a, b]$, to

$$|I(f) - Q(f)| \leq \frac{\|f^{(n+1)}\|}{(n+1)!} |b-a|^{n+2}.$$

210

Błąd prostych kwadratur

Twierdzenie

- (dla kwadratury prostokątów) jeśli $f \in C^2[a, b]$, to

$$I(f) - Q^Q(f) = \frac{(b-a)^3}{24} f''(\xi), \quad \text{dla pewnego } \xi \in [a, b],$$

- (dla kwadratury trapezów) jeśli $f \in C^2[a, b]$, to

$$I(f) - Q^T(f) = -\frac{(b-a)^3}{12} f''(\xi), \quad \text{dla pewnego } \xi \in [a, b],$$

- (dla kwadratury Simpsona) jeśli $f \in C^4[a, b]$, to

$$I(f) - Q^S(f) = -\frac{(b-a)^5}{2880} f^{(4)}(\xi), \quad \text{dla pewnego } \xi \in [a, b].$$

211

Kwadratury maksymalnego rzędu

Definicja

Kwadratura Q dla całki

$$I(f) = \int_a^b f(x) \cdot \rho(x) dx \quad (\rho \text{ to funkcja wagowa})$$

jest rzędu (co najmniej) r , jeśli

$$I(p) = Q(p) \quad \forall p \in P_{r-1}.$$

Jest rzędu r , gdy jest co najmniej rzędu r , ale już nie $r+1$.

Wniosek

Każda kwadratura interpolacyjna oparta na n węzłach jest co najmniej rzędu n .

Problem

Jaki jest maksymalny rząd kwadratury? Jaka to kwadratura?

212

Kwadratury złożone

Obserwacja:

$$\int_a^b f(x) dx = \sum_{j=0}^{N-1} \int_{x_j}^{x_{j+1}} f(x) dx$$

gdzie $a = x_0 < x_1 < \dots < x_N = b$, zatem wystarczy aproksymować tylko całki na (krótszych) odcinkach $[x_i, x_{i+1}]$:

$$\int_{x_i}^{x_{i+1}} f(x) dx \approx Q_{[x_i, x_{i+1}]}(f) \implies \int_a^b f(x) dx \approx \sum_{j=0}^{N-1} Q_{[x_j, x_{j+1}]}(f).$$

214

Kwadratury Gaussa

Twierdzenie

Rząd kwadratury interpolacyjnej opartej na n węzłach nie przekracza $2n$. Realizuje go kwadratura Gaussa, oparta na zerach n -tego wielomianu ortogonalnego w $L^2_\rho(a, b)$.

Twierdzenie

Jeśli $f \in C^{2n}[a, b]$, a Q jest kwadraturą Gaussa opartą na n węzłach, to

$$\int_a^b f(x) \cdot \rho(x) dx - Q(f) = \|\phi_n\|_\rho^2 \cdot \frac{f^{(2n)}(\xi)}{(2n)!}$$

gdzie $\phi_n(x)$ — wielomian bazowy Newtona oparty na węzłach kwadratury, $\xi \in [a, b]$.

213

Kwadratury złożone: przykłady

Zakładamy jednorodny podział $[a, b]$ węzłami $x_i = a + i \cdot h$, gdzie $h = (b - a)/N$.

- prostokątów

$$\int_a^b f(x) dx \approx Q_N^Q(f) = h \sum_{i=0}^{N-1} f\left(\frac{x_i + x_{i+1}}{2}\right)$$

- trapezów

$$\int_a^b f(x) dx \approx Q_N^T(f) = \frac{h}{2} \left(f(a) + 2 \sum_{i=1}^{N-1} f(x_i) + f(b) \right)$$

- Simpsona

$$\int_a^b f(x) dx \approx Q_N^S(f) = \frac{h}{6} \left(f(a) + 2 \sum_{i=1}^{N-1} f(x_i) + f(b) + 4 \sum_{i=0}^{N-1} f\left(\frac{x_i + x_{i+1}}{2}\right) \right)$$

215

Błąd kwadratur złożonych

Szacuje się przez sumę błędów lokalnych:

$$|I(f) - Q_N(f)| \leq \sum_i |I_{[x_i, x_{i+1}]}(f) - Q_{[x_i, x_{i+1}]}(f)|.$$

Jeśli więc błędy lokalne są rzędu h^{r+1} , to błąd kwadratury będzie rzędu h^r . Ponadto na przykład:

Twierdzenie

Jeśli $f \in C^2[a, b]$, to dla złożonej kwadratury trapezów

$$I(f) - Q_N^T(f) = -\frac{h^2}{12} |b-a| f''(\xi) \quad \text{dla pewnego } \xi \in [a, b].$$

216

Podwyższanie rzędu aproksymacji. Kwadratury Romberga

Iterując ten pomysł definiujemy, oznaczając $R_j^k = R_j\left(f, \frac{h}{2^k}\right)$,

$$R_j^k = \frac{1}{4^j - 1} (4^j R_{j-1}^{k+1} - R_{j-1}^k), \quad \leftarrow \text{kwadratury Romberga.}$$

$$\begin{bmatrix} R_0^0 & & & \\ R_0^1 & R_1^0 & & \\ R_0^2 & R_1^1 & R_2^0 & \\ \vdots & \vdots & \vdots & \ddots \\ R_0^m & R_1^{m-1} & & R_m^0 \end{bmatrix}$$

R_0^k — kw. trapezów, R_1^k — kw. Simpsona

Twierdzenie

Jeśli $f \in C^{2m}[a, b]$, to dla $j \leq m$ błąd kwadratury R_j^k spełnia

$$I(f) - R_j^0 = \mathcal{O}(h^{2j+2}).$$

218

Podwyższanie rzędu aproksymacji. Kwadratury Romberga

Oznaczmy $R_0(f, h) := Q_N^T(f)$. Okazuje się, że dla $f \in C^{2m}[a, b]$ istnieją stałe α_i t. że

$$I(f) - R_0(f, h) = \alpha_2 h^2 + \alpha_4 h^4 + \dots + \alpha_{2m-2} h^{2m-2} + \mathcal{O}(h^{2m}).$$

Zatem też dla $q \in (0, 1)$

$$I(f) - R_0(f, qh) = \alpha_2 q^2 h^2 + \alpha_4 q^4 h^4 + \dots + \alpha_{2m-2} q^{2m-2} h^{2m-2} + \mathcal{O}(h^{2m}).$$

Stąd

$$R_1(f) = \frac{1}{1-q^2} (R_0(f, qh) - q^2 R_0(f, h))$$

ma wyższy rząd błędu:

$$I(f) - R_1(f, h) = \tilde{\alpha}_4 h^4 + \dots + \tilde{\alpha}_{2m-2} h^{2m-2} + \mathcal{O}(h^{2m}).$$

217

Kwadratury wielowymiarowe: tensorowe

Niech $f : [0, 1]^d = D \rightarrow \mathbb{R}$ będzie całkowna. Wtedy

$$I(f) = \int_{[0,1]^d} f(x) dx = \int_0^1 \dots \int_0^1 f(x_1, \dots, x_d) dx_1 \dots dx_d,$$

Stosując po każdej współrzędnej kwadraturę jednowymiarową Q^1 opartą na n węzłach dostajemy więc aproksymację postaci

$$\begin{aligned} I(f) \approx Q(f) &= \sum_{i_d=1}^n A_{i_d} \cdot \left(\dots \left(\sum_{i_1=1}^n A_{i_1} \cdot f(x_{i_1}, \dots, x_{i_d}) \right) \dots \right) \\ &= \sum_{i_1=1}^n \dots \sum_{i_d=1}^n A_{i_1} \dots A_{i_d} \cdot f(x_{i_1}, \dots, x_{i_d}) \end{aligned}$$

Ma ona n^d węzłów! („Przekleństwo wymiaru”)

219

$$I(f) \approx Q^{MC}(f) = \frac{1}{N} \sum_{i=1}^N f(t_i),$$

gdzie t_i są losowymi punktami z rozkładu jednostajnego na $[0, 1]^d$.

Zatem $Q^{MC}(f)$ jest zmienną losową!

Twierdzenie

Jeśli $f \in L^2([0, 1]^d)$, to

- wartość oczekiwana $\mathbb{E} Q^{MC}(f) = I(f)$
- wartość oczekiwana kwadratu błędu

$$\mathbb{E} (I(f) - Q^{MC}(f))^2 = \frac{I(f^2) - I^2(f)}{N}$$

— nie zależy od wymiaru przestrzeni d .

220

Szybka transformacja Fouriera (FFT)

Dyskretna transformacja Fouriera (DFT)

Oznaczając $\omega_N = e^{-2i\pi/N}$

$$c_k = \sum_{j=0}^{N-1} f_j \cdot e^{-ikj \cdot 2\pi/N} = \sum_{j=0}^{N-1} f_j \cdot \omega_N^{kj}, \quad k = 0, \dots, N-1.$$

Macierzowo:

$$c = F_N \cdot f,$$

gdzie

$$F_N = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_N & \omega_N^2 & \dots & \omega_N^{N-1} \\ 1 & \omega_N^2 & \omega_N^4 & \dots & \omega_N^{2(N-1)} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & \omega_N^{N-1} & \omega_N^{2(N-1)} & \dots & \omega_N^{(N-1)^2} \end{bmatrix}, \quad f = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{N-1} \end{bmatrix}, \quad c = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{N-1} \end{bmatrix}$$

Nawet jeśli byśmy wyznaczyli F_N w preprocessingu, koszt zwykłego mnożenia przez F_N byłby $\mathcal{O}(N^2)$.

221

Szybka transformacja Fouriera (FFT)

$$F_N = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_N & \omega_N^2 & \dots & \omega_N^{N-1} \\ 1 & \omega_N^2 & \omega_N^4 & \dots & \omega_N^{2(N-1)} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & \omega_N^{N-1} & \omega_N^{2(N-1)} & \dots & \omega_N^{(N-1)^2} \end{bmatrix}$$

- Koszt mnożenia przez F_N obniżymy z $\mathcal{O}(N^2)$ do $\mathcal{O}(N \log_2 N)$
- Koszt rozwiązywania układu z F_N też obniżymy do $\mathcal{O}(N \log_2 N)$

przez wykorzystanie metody *dziel i rządź*.

Stwierdzenie

$$\omega_N^N = 1.$$

222

Dla uproszczenia, $N = 2^p$ i oznaczmy $m = N/2$.

$$\begin{aligned}
 c_j &= \sum_{k=0}^{N-1} f_k \cdot \omega_N^{kj} \\
 &= f_0 \omega_N^{0 \cdot j} + f_2 \omega_N^{2 \cdot j} + \dots + f_{N-2} \omega_N^{(N-2) \cdot j} \\
 &\quad + f_1 \omega_N^{1 \cdot j} + f_3 \omega_N^{3 \cdot j} + \dots + f_{N-1} \omega_N^{(N-1) \cdot j} \\
 &= f_0 \omega_N^{0 \cdot j} + f_2 \omega_N^{2 \cdot j} + \dots + f_{N-2} \omega_N^{(N-2) \cdot j} \\
 &\quad + \omega_N^j \cdot (f_1 \omega_N^{0 \cdot j} + f_3 \omega_N^{2 \cdot j} + \dots + f_{N-1} \omega_N^{(N-2) \cdot j}) \\
 &= \sum_{k=0}^{N/2-1} f_{2k} \omega_N^{2kj} + \omega_N^j \cdot \sum_{k=0}^{N/2-1} f_{2k+1} \omega_N^{2kj} \\
 &= \sum_{k=0}^{m-1} f_{2k} \omega_N^{kj} + \omega_N^j \cdot \sum_{k=0}^{m-1} f_{2k+1} \omega_N^{kj},
 \end{aligned}$$

bo $\omega_N^2 = \omega_m$.

223

$$c_j = \sum_{k=0}^{m-1} f_{2k} \omega_m^{kj} + \omega_N^j \cdot \sum_{k=0}^{m-1} f_{2k+1} \omega_m^{kj}.$$

Zapiszmy $j = 0, \dots, m, \dots, N-1$ w postaci

$$j = \beta \cdot m + \alpha, \quad \beta \in \{0, 1\}, \quad \alpha = 0, \dots, m-1.$$

Mamy $\omega_m^j = \omega_m^\alpha$, więc

$$\begin{aligned}
 c_\alpha &= \sum_{k=0}^{m-1} f_{2k} \omega_m^{k\alpha} + \omega_N^\alpha \cdot \sum_{k=0}^{m-1} f_{2k+1} \omega_m^{k\alpha} \quad \alpha = 0, \dots, m-1, \\
 c_{\alpha+m} &= \underbrace{\sum_{k=0}^{m-1} f_{2k} \omega_m^{k\alpha}}_{=: \Phi_\alpha} - \underbrace{\omega_N^\alpha \cdot \sum_{k=0}^{m-1} f_{2k+1} \omega_m^{k\alpha}}_{=: \Psi_\alpha} \quad \alpha = 0, \dots, m-1
 \end{aligned}$$

Ostatecznie dla $\alpha = 0, \dots, m-1$,

$$c_\alpha = \Phi_\alpha + \omega_N^\alpha \Psi_\alpha,$$

$$c_{\alpha+m} = \Phi_\alpha - \omega_N^\alpha \Psi_\alpha.$$

224

Algorytm FFT

Mamy obliczyć $c_j = \sum_{k=0}^{N-1} f_k \cdot \omega_N^{kj}$ dla $j = 0, \dots, N-1$.
Dla $\alpha = 0, \dots, m-1$,

$$c_\alpha = \Phi_\alpha + \omega_N^\alpha \Psi_\alpha,$$

$$c_{\alpha+m} = \Phi_\alpha - \omega_N^\alpha \Psi_\alpha.$$

gdzie

$$\Phi_\alpha = \sum_{k=0}^{m-1} f_{2k} \omega_m^{k\alpha}, \quad \Psi_\alpha = \sum_{k=0}^{m-1} f_{2k+1} \omega_m^{k\alpha}$$

Czyli $c = F_N f$ obliczamy rekurencyjnie!

Koszt:

$$K(N) = 2K(N/2) + \mathcal{O}(N),$$

zatem na mocy TRU

$$K(N) = \mathcal{O}(N \log_2 N).$$

225

DFT — własności

$$F_N = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_N & \omega_N^2 & \dots & \omega_N^{N-1} \\ 1 & \omega_N^2 & \omega_N^4 & \dots & \omega_N^{2(N-1)} \\ & & & \dots & \\ 1 & \omega_N^{N-1} & \omega_N^{2(N-1)} & \dots & \omega_N^{(N-1)^2} \end{bmatrix}, \quad T_N = \begin{bmatrix} 1 & & & & \\ & & & & 1 \\ & & & \ddots & \\ & & \ddots & & \\ & 1 & & & \end{bmatrix}$$

Twierdzenie

- $F_N = F_N^T$
- $\frac{1}{N} F_N^H F_N = I$; $F_N^{-1} = \frac{1}{N} \bar{F}_N$; $\frac{1}{\sqrt{N}} F_N$ jest unitarna
- $\bar{F}_N = T_N F_N = F_N T_N$
- $F_N^{-1} y = \frac{1}{N} \bar{F}_N y = \frac{1}{N} \overline{F_N \bar{y}}$

226

Dyskretne transformacje trygonometryczne: DST i DCT

Dane: $f_0, \dots, f_N \in \mathbb{R}$. Obliczyć:

DST-1

$$y_k = \sum_{j=1}^{N-1} f_j \cdot \sin\left(kj \frac{\pi}{N}\right) \quad \text{dla } k = 1, \dots, N-1,$$

DCT-1

$$y_k = \sum_{j=0}^N f_j \cdot \cos\left(kj \frac{\pi}{N}\right) \quad \text{dla } k = 0, \dots, N.$$

Istnieje wiele wariantów tych transformacji (są popularne w zastosowaniach) Na przykład, kompresja JPEG wymaga użycia

DCT-2:

$$y_k = \sum_{j=0}^{N-1} f_j \cdot \cos\left(k\left(j + \frac{1}{2}\right) \frac{\pi}{N}\right) \quad \text{dla } k = 0, \dots, N-1.$$

Kompresja MP3 wymaga MDCT opartej na DCT-4.

227

Metody numeryczne!

FFT — zastosowania

- obliczanie DFT, DST i DCT różnej maści
- obliczanie cyklicznego splotu dwóch wektorów:

$$z_k = \sum_{j=0}^{N-1} u_j v_{k-j}, \quad k = 0, \dots, N-1,$$

- mnożenie wielomianów wysokiego stopnia
-

228

Specyfika metod numerycznych

Przedmiot interdyscyplinarny, na pograniczu

- matematyki teoretycznej (analiza algorytmów dla zadań ciągłych)
- informatyki teoretycznej (analiza algorytmów dla zadań dyskretnych)
- matematyki stosowanej (konstrukcja metod dla konkretnego modelu)
- informatyki praktycznej (implementacja algorytmów, dopasowanie do architektury)
- ma też swoją własną specyfikę (np. arytmetyka fl)

229

Wykłady obieralne **dostępne dla Informatyki** w stałej ofercie:

- Analiza numeryczna
- Aproksymacja i złożoność
- Grafika komputerowa
- Obliczenia naukowe
- Numeryczne równania różniczkowe
- Metody obliczeniowe w finansach

Do tego wykłady monograficzne i seminaria monograficzne z numeryki.

- seminarium magisterskie **dostępne dla informatyki i matematyki**
- prace magisterskie o charakterze
 - implementacyjnym
 - teoretycznym (analitycznym)

<https://www.mimuw.edu.pl/~przykry/mnmgr>