Algorytmy Numeryczne

Zagadnienia Egzaminacyjne

„Wada: niepewność”

Kraków

Anno Domini 2025

Spis treści

1 Błędy obliczeń **1**

1.1 Kodowanie liczb w komputerze . . . 1

1.2 Utrata precyzji . . . 1

1.2.1 Algorytm Kahana . . . 1

1.3 Uwarunkowanie zadania i stabilność algorytmu . . . 2

2 Eliminacja Gaussa **3**

2.1 Algorytm - rozkład LU . . . 3

2.2 Zastosowania . . . 3

3 Układy równań liniowych **4**

3.1 Rozwiązywanie eliminacją Gaussa . . . 4

3.2 Układy niedookreślone . . . 4

4 Ortogonalność, rozkład QR **5**

4.1 Układy nadokreślone – najmniejsze kwadraty . . . 5

4.1.1 Metoda najmniejszych kwadratów . . . 5

4.2 Rozkład Cholesky'ego . . . 5

4.3 Rozkład QR . . . 6

4.4 Algorytm Grama-Schmitta . . . 6

4.5 Odbicia Householdera . . . 6

4.6 Obroty Givensa . . . 6

5 Wektory i wartości własne **7**

5.1 Wyznaczanie wartości własnych . . . 7

5.2 Metody iteracyjne . . . 7

5.2.1 Iteracja prosta . . . 7

5.2.2 Deflacja . . . 7

5.3 Metoda QR . . . 8

5.4 PageRank . . . 8

5.5 Potęgowanie macierzy . . . 8

6 Wartości szczególne **9**

6.1 Rozkład SVD . . . 9
6.2 Wyznaczanie . . . 10
6.2.1 Algorytm Goluba-Kahana . . . 10
6.3 Rozwiązywanie układu równań . . . 10
6.4 Idea PCA . . . 10

7 Równania nieliniowe **11**
7.1 Metoda bisekcji . . . 11
7.2 Metoda Newtona . . . 11
7.3 Metoda siecznych . . . 12
7.4 Metoda punktu stałego . . . 12

8 Optymalizacja nieograniczona **13**
8.1 Ekstrema funkcji bitonicznych . . . 13
8.1.1 Wyszukiwanie ternarne . . . 13
8.1.2 Wyszukiwanie ze złotym podziałem . . . 13
8.2 Wzór Taylora . . . 14
8.3 Metoda gradient descent . . . 14
8.3.1 Metoda gradientu prostego – wariant . . . 14

9 Układy równań – metody iteracyjne **15**
9.1 Schemat działania . . . 15
9.2 Metody Richardsona . . . 15
9.3 Metody Jacobiego . . . 15
9.4 Metody Gaussa-Seidela . . . 16
9.5 Metoda najszybszego spadku . . . 16

10 Interpolacja **17**
10.1 Interpolacja Lagrange'a . . . 17
10.2 Interpolacja Newtona . . . 18
10.3 Funkcje sklejane . . . 18

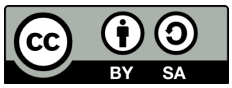
11 Optymalizacja z ograniczeniami **19**
11.1 Ekstrema warunkowe . . . 19
11.2 Programowanie liniowe . . . 19
11.3 Problem ogólny . . . 20
11.4 Metoda simplex . . . 20

12 Różniczkowanie i całkowanie numeryczne **21**
12.1 Ogólny schemat kwadratury . . . 21
12.2 Podstawowe kwadratury . . . 21
12.3 Przybliżone różniczkowanie . . . 22

13 Transformata Fouriera **23**
13.1 Rozwinięcie w szereg Fouriera . . . 23
13.2 Transformata Fouriera . . . 23
13.2.1 Dyskretna transformata Fouriera . . . 23
13.3 Kompresja fal . . . 24
13.3.1 Kompresja dźwięku . . . 24
13.3.2 Kompresja obrazu . . . 24

14 Równania różniczkowe **25**
14.1 Ogólny schemat rozwiązywania . . . 25
14.2 Równania o zmiennych rozdzielonych . . . 25
14.3 Równania jednorodne liniowe pierwszego rzędu . . . 25
14.4 Metoda Eulera . . . 26

Licencja



Rozdział 1

Błędy obliczeń

1.1 Kodowanie liczb w komputerze

- **Przez ułamki** - tylko liczby wymierne, dodawanie szybko zwiększa liczbę cyfr w liczniku i mianowniku
- **Stałoprzecinkowe** - ograniczony zakres i marne wykorzystanie pamięci
- **Zmiennoprzecinkowe**

$$x = (1 + M) \cdot 2^w,$$

gdzie w to wykładnik (cecha), a $M \in (0, 1)$ to mantysa. Wykładnik zapisuje się z przesunięciem $(2^{b_w - 1} - 1)$.

Kodowanie zera: $w = 0 \dots 0,\ M = 0 \dots 0$

Kodowanie nieskończoności: $w = 1 \dots 1,\ M = 0 \dots 0$

Kodowanie NaN: $w = 1 \dots 1,\ M \in (0, 1)$

Kodowanie nieznormalizowane: $w = 0 \dots 0,\ M \in (0, 1)$ - liczby mniejsze od 2^{-b_M}

1.2 Utrata precyzji

Przy odejmowaniu podobnych na wielkość liczb może wystąpić utrata cyfr znaczących (*catastrophic cancellation*), czyli wybuch błędu względnego. Przy dodawaniu wielu liczb, jeśli błąd pojedyńczego dodawania jest równy ε, to końcowy wynik może różnić się od poprawnego o coś rzędu $O(n\varepsilon)$.

1.2.1 Algorytm Kahana

```
sum = 0, rest = 0
for i = 1, ..., n do
    x = xi + rest
    sum1 = sum + x
    rest = x - (sum1 - sum)
    sum = sum1
```

Algorytm daje błąd bezwzględny $O(\varepsilon + n\varepsilon^2)$.

1.3 Uwarunkowanie zadania i stabilność algorytmu

Dana jest liczba t, należy wyznaczyć wartość pewnej funkcji $f(t)$.
Współczynnik uwarunkowania to taka liczba $A = A(f, t)$, dla której

$$\|f(t) - f(t^*)\| \leq A \cdot \|t - t^*\|,$$

$A \approx f'(t)$ dla rozsądnych funkcji f.

Dane jest równanie $g(x) = 0$, należy wyznaczyć x.
Współczynnik uwarunkowania to taka liczba $A = A(g, x)$, dla której

$$\|x - x^*\| \leq A \cdot \|g(x) - g(x^*)\|,$$

$A \approx \frac{1}{g'(x)}$ dla rozsądnych funkcji f.
Problem jest dobrze uwarunkowany, jeśli małe zaburzenia danych wejściowych spowodują małą zmianę wyniku.

Współczynnik uwarunkowania macierzy A to $\kappa = \|A\| \cdot \|A^{-1}\|$. Dla równania $Ax^* = b^*$ z zaburzoną wartością b

$$\frac{\|x - x^*\|}{\|x\|} \leq \kappa(A) \cdot \frac{\|b - b*\|}{\|b\|}$$

Układy równań z wysokim κ są trudne do rozwiązania numerycznie.

Dane jest x oraz algorytm obliczający $A(x)$, które przybliża $f(x)$. Algorytm jest **numerycznie poprawny**, jeśli dla każdego x istnieje x^* które dobrze przybliża x oraz $A(x^*)$ dobrze przybliża $f(x)$. Algorytm jest **numerycznie stabilny**, jeśli małe błędy na wejściu lub podczas działania algorytmu powodują małe zmiany wyniku.

Rozdział 2

Eliminacja Gaussa

2.1 Algorytm - rozkład LU

```
for i = 1, ..., n do
    m = max_j(|A[j][i]|)
    swap(A[i], A[m]) // partial pivoting
    a = A[i][i]
    for j = i+1, ..., n do
        r = A[j][i] / a
        L[i][j] = r
        A[j] -= r * A[i]
```

Złożoność: $O(n^3)$, dokładnie $\frac{2}{3}n^3$ operacji na liczbach rzeczywistych

Przy wyborze małego elementu głównego stabilność numeryczna pogarsza się przez błędy zaokrągleń - mogą pojawić się bardzo małe i bardzo duże wartości.

Pivoting (częściowy) pomaga. Wybieramy wiersz z największym na wartość bezwzględną elementem na przekątnej i to daje większą stabilność numeryczną. Przy rozkładzie LU trzeba wtedy pamiętać, że wynikiem jest rozkład $LU = PA$, gdzie P jest macierzą permutacji.

2.2 Zastosowania

Mając $A = LU$:

- $\det A = U_{1,1} \cdot \cdots \cdot U_{n,n}$
- A^{-1} wyznaczamy, eliminując wszystko poza przekątną
- można wykryć liniową zależność rzędów / kolumn
- można rozwiązać równanie $Ax = b$ w czasie $O(m \cdot n^2)$ dla m różnych b
- można rozwiązać równanie macierzowe $AX = B$, rozbijając B na kolumny

Rozdział 3

Układy równań liniowych

Twierdzenie 3.0.1 (Kroneckera-Capellego (Rouché-Capellego))**.** Niech $Ax = b$ jest układem m równań z n niewiadomymi. Oznaczamy $r_a = \operatorname{rank} A$, $r_{ab} = \operatorname{rank}[A|b]$. Układ ma rozwiązanie wtw $r_a = r_{ab}$ i jest ono jednoznaczne wtw $r_a = n$. Inaczej rozwiązania stanowią przestrzeń o rozmiarze $n - r_a$.
Jeśli układ ma dwa rozwiązania, to ma ich nieskończenie wiele. Skoro $Ax = b$ i $Ax' = b$, to $A \cdot (\alpha x + (1 - \alpha)x') = b$.

3.1 Rozwiązywanie eliminacją Gaussa

Mając $Ax = b$ (czyli $LUx = b$), najpierw rozwiązujemy $Ly = b$, a potem $Ux = y$ - wszystko w $O(n^2)$.
Układ Ux = b rozwiązuje się tak:

```
for i = n, ..., 1
    x[i] = b[i]
    for j = i + 1, ..., n
        x[i] -= U[i][j] * x[j]
```

3.2 Układy niedookreślone

Mają wiele rozwiązań, bo wiele wektorów minimalizuje $\|Ax - b\|$. Wybieramy najmniejszy, czyli ten, który dodatkowo minimalizuje $\|Ax - b\| + \alpha \|x\|^2$.

Rozdział 4

Ortogonalność, rozkład QR

4.1 Układy nadokreślone – najmniejsze kwadraty

Jeśli Ax = b jest nadokreślony i nie ma rozwiązania, to celem jest znaleźć x*, który minimalizuje $\|Ax^* - b\|^2$.

4.1.1 Metoda najmniejszych kwadratów

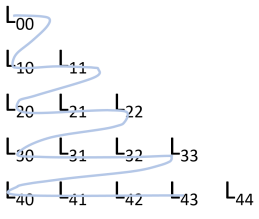
Mamy dane $x_1, \ldots, x_n$ oraz $y_1, \ldots, y_n$, a szukamy f, które zminimalizuje $\sum(f(x_i) - y_i)^2$.
Jeśli Ax = b to nadokreślony układ równań, to poszukiwane przybliżone rozwiązanie x^* spełnia $A^T Ax^* = A^T b$.

Macierz $A^T A$ jest gorzej uwarunkowana, $\kappa(A^T A) = \kappa(A)^2$

4.2 Rozkład Cholesky'ego

Macierz C jest **dodatnio określona**, jeśli $\forall_x\ x^T Cx > 0$.
Macierz C jest **dodatnio półokreślona**, jeśli $\forall_x\ x^T Cx \geq 0$.
Dla każdej macierzy A, macierz $A^T A$ jest dodatnio półokreślona, a jeśli kolumny A są liniowo niezależne, to jest dodatnio określona. Do tego jest symetryczna. Można ją więc rozłożyć na LL^T, rozkład Cholesky'ego:

Już wyliczone: $L_{1,1}, \ldots, L_{i,1}, \ldots, L_{i,j-1}$
$A_{ij} = L_i \cdot L_j = \sum_{k=1}^{j}\ L_{ik}L_{jk}$
$L_{ij} = \frac{A_{ij} - \sum_{k=1}^{j-1}\ L_{ik}L_{jk}}{L_{jj}}$, dla $i > j$
$L_{ii} = \sqrt{A_{ii} - \sum_{k=1}^{i-1}\ L_{ik}^2}$
czyli takim zygzakiem.



Rozkład Cholesky'ego jest stabilniejszy numerycznie i ma lepszą stałą niż standardowa eliminacja Gaussa.

4.3 Rozkład QR

Jeśli zbiór $\{v_1, \ldots, v_n\}$ stanowi bazę przestrzeni liniowej, to każdy wektor x wyraża się w tej bazie łatwo:

$$x = \sum_i \langle x, v_i \rangle v_i$$

Mając ortonormalną macierz Q, rozwiązaniem $Qx = b$ jest $x = Q^T b$. Pozostaje tylko rozłożyć $A = QR$, gdzie Q jest ortogonalna, a R górnotrójkątna.

4.4 Algorytm Grama-Schmitta

$\text{span}\{q_1, \ldots, q_i\} = \text{span}\{a_1, \ldots, a_i\}$

Wersja podstawowa

```
for i = 1, ..., n
    qi = ai
    for j = 1, ..., i-1
        qi = qi - <ai, qj>qj
    qi = qi / ||qi||
```

Wersja ulepszona

```
for i = 1, ..., n
    qi = ai / ||ai||
    for j = i+1, ..., n
        aj = aj - <aj, qi>qi
```

Modyfikacja daje większą stabilność numeryczną. Ostatecznie $R = Q^T A$ i $Rx = Q^T b$.

4.5 Odbicia Householdera

Szukamy takiego odbicia symetrycznego, które przeprowadzi wektor x na $\|x\|e$.

$$u = -x + \text{sgn}(x)\|x\|e, \quad v = \frac{u}{\|u\|}$$

$$Q = I - 2vv^T$$

Dalsze macierze Q_i działają na A bez pierwszych (już gotowych) wierszy i kolumn.

4.6 Obroty Givensa

Szukamy macierzy $\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$, która przekształci $\begin{bmatrix} x \\ y \end{bmatrix}$ w $\begin{bmatrix} r \\ 0 \end{bmatrix}$, gdzie $r = x^2 + y^2$. Rozwiązaniem jest $\cos\theta = \frac{x}{r}, \sin\theta = \frac{-y}{r}$. Rozszerzamy macierz do odpowiednich wymiarów jak macierz jednostkową. Metoda jest wolniejsza od odbić Householdera, ale działa lepiej dla rzadkich macierzy i dobrze się zrównolegla.

Rozdział 5

Wektory i wartości własne

5.1 Wyznaczanie wartości własnych

Wartości własne A to pierwiastki wielomianu charakterystycznego $W(t) = \det(A - tI)$.

Szukanie pierwiastków wielomianu jest bardzo źle uwarunkowane.

5.2 Metody iteracyjne

Niech $v = \alpha_1 x_1 + \cdots + \alpha_n x_n$. Wówczas:

$$A^k v = \alpha_1 \lambda_1^k x_1 + \cdots + \alpha_n \lambda_n^k x_n,$$

Jeśli $|\lambda_1| > |\lambda_i|$ dla i > 1, to $\lambda_1^k \gg \lambda_i^k$. Zatem wektor $A^k v$ ma kierunek coraz bardziej zgodny z x_1.

5.2.1 Iteracja prosta

Zaczynając od $v_0 = v$, stosujemy

$$v_{k+1} = \frac{Av_k}{\|Av_k\|}$$

Wtedy v_k zbiega do x_1, o ile A ma dominującą wartość własną.

5.2.2 Deflacja

Jeśli „wyrzucimy” wektor własny x_1 z początkowego v, to teoretycznie znajdziemy wektor odpowiadający drugiej co do wielkości wartości własnej. W praktyce musimy wyrzucać x_1 po każdym kroku:

$$v = v - \langle v, x_1 \rangle x_1$$

5.3 Metoda QR

Zaczynając od $A_0 = A$, stosujemy algorytm:

- $(Q_k, R_k) = \text{decompose}(A_k)$
- $A_{k+1} = R_k Q_k$

Kolejne A_k zmierzają do macierzy diagonalnej z wartościami własnymi na przekątnej. W teorii jest tak, jeśli A jest diagonalizowalna, symetryczna i ma różne wartości własne.
Żeby przyspieszyć obliczenia, sprowadzamy A do macierzy Hessenberga (zera pod przekątną), np odbiciami Householdera i tak unikamy wielokrotnego czasochłonnego rozkładu.
Żeby przyspieszyć zbieżność, stosujemy przesunięcie:

- $(Q_k, R_k) = \text{decompose}(A_k - I \cdot \mu)$
- $A_{k+1} = R_k Q_k + I \cdot \mu$

Za μ można przyjąć prawy-dolny element A.

5.4 PageRank

W łańcuchach Markowa stan graniczny musi spełniać $v = Pv$, czyli musi być wektorem własnym macierzy przejść. Algorytm PageRank to szczególny łańcuch Markowa. Dla każdej strony wylicza graniczne prawdopodobieństwo znalezienia się na niej.

5.5 Potęgowanie macierzy

Macierz A jest diagonalizowalna (*non-defective*), jeśli ma n liniowo niezależnych wektorów własnych. Wtedy $A = P^{-1}DP$, gdzie P jest odwracalna, D diagonalna (z wartościami własnymi). Każda macierz (nawet niediagonalizowalna) ma postać Jordana $A = P^{-1}JP$, wtedy J jest jak D + jedynki pod przekątną. Z tego wynika, że przy potęgowaniu macierz zachowuje się jak największa wartość własna – jeśli $\lambda > 1$, ciąg A^k jest rozbieżny, jeśli $\lambda < 1$ jest zbieżny.
Dygresja: Jeśli P jest odwracalna, to $P^{-1}AP$ ma te same wartości własne, co A (macierze są sprzężone).

Rozdział 6

Wartości szczególne

Prawe wektory szczególne A to $v_1, \ldots, v_n$, dla których $A^T A v_i = \lambda_i v_i$, dla $\lambda_i \in \mathbb{R}$.
Wartości szczególne A to $\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_n}$.
Lewe wektory szczególne A to $u_1, \ldots, u_n$, gdzie $u_i = \frac{1}{\sqrt{\lambda_i}} A v_i$.
Wektor szczególny to taki, który odpowiada najbardziej wydłużanemu kierunkowi.

6.1 Rozkład SVD

Przekształcamy macierz A liniowo:

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

złożeniem odwzorowań, gdzie V ma wektory v_i jako wiersze, U ma u_i jako kolumny, Σ jest diagonalna z $\sqrt{\lambda_i}$.

1. zmiana bazy (obrót przestrzeni) $(v_i \to e_i)$
2. skalowanie $(e_i \to \sqrt{\lambda_i} e_i)$
3. powrót (obrót przestrzeni) $(e_i \to u_i)$

Przypadek I: A jest osobliwa
Wtedy niektóre λ_i są zerami. Niezerowych u_i jest mniej niż n, więc uzupełniamy je dowolnymi wektorami do bazy ortonormalnej.
Przypadek II: A nie jest kwadratowa
Jeśli $m < n$, to $A^T A$ jest rozmiaru $n \times n$, ale ma mniejszy rząd (m) i część wartości szczególnych jest zerami. Mamy n prawych i m lewych wektorów szczególnych.
Jeśli $m > n$, to $A^T A$ jest rozmiaru $n \times n$. Mamy n prawych wektorów szczególnych – dopełniamy u_i wektorami $A v_i$ do bazy.
Macierz V jest $n \times n$, Σ – $m \times n$, a U jest $m \times m$.

6.2 Wyznaczanie

Szukamy wartości i wektorów własnych macierzy A^TA. Na przykład metodą QR, bo A^TA jest symetryczna i dodatnio określona.

Samo obliczanie A^TA pogarsza stabilność.

6.2.1 Algorytm Goluba-Kahana

Szukamy najpierw takich unitarnych macierzy U' i V', żeby $B = U'AV'$ była bidiagonalna (naprzemiennymi macierzami Householdera). Wtedy B^TB jest tridiagonalna (w postaci Hessenberga), więc potrzebne jest mniej dodawań i mnożeń $\rightarrow$ lepsza stabilność numeryczna.

6.3 Rozwiązywanie układu równań

Problem równoważny: zminimalizować $\|x\|^2$ dla x spełniających $A^TAx = A^Tb$.
Wstawiamy $A = U\Sigma V^T$. Wówczas $\Sigma V^Tx = U^Tb$. Oznaczając $y = V^Tx$, trzeba zminimalizować $\|y\|^2$. Rozwiązaniem jest Σ^+U^Tb, gdzie Σ^+ ma na przekątnej odwrotności wartości z Σ lub zera. To prowadzi do rozwiązania $x = V\Sigma^+U^Tb$. Macierz $A^+ = V\Sigma^+U^Tb$ to pseudoodwrotność A.

- Jeśli A jest kwadratowa i nieosobliwa, to $A^+ = A^{-1}$.
- Jeśli $Ax = b$ jest nadokreślony, to A^+b jest przybliżonym rozwiązaniem w sensie najmniejszych kwadratów.
- Jeśli $Ax = b$ jest niedookreślony, to A^+b jest najmniejszym w sensie normy rozwiązaniem

6.4 Idea PCA

Dla zbioru wektorów zestawionych w macierz szukamy pierwszej wartości szczególnej. To daje przybliżenie wszystkich wektorów jednym (współrzędną / składową). Przy szukaniu następnej usuwamy poprzednią.
Podobnie można zapominać najmniejsze wyrazy przy kompresji obrazów, po potraktowaniu ich jak macierze i rozłożeniu SVD.

Rozdział 7

Równania nieliniowe

Celem jest znaleźć pierwiastek funkcji na przedziale $[a, b]$. Metody korzystają z tego:
Twierdzenie Darboux
Jeśli funkcja $f : [a, b] \to \mathbb{R}$ jest ciągła i $f(a) \leq y \leq f(b)$ lub $f(b) \leq y \leq f(a)$, to istnieje $x \in [a, b]$, t.że $f(x) = y$.
Twierdzenie Lagrange'a
Jeśli $f : [a, b] \to \mathbb{R}$ jest różniczkowalna, to istnieje $x \in (a, b)$ t.że $f'(x) = \frac{f(b)-f(a)}{b-a}$.
Wzór Taylora

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots + \frac{f^{(k)}(a)}{k!}(x-a)^k + R_{k+1}(x, a),$$

gdzie $R_{k+1} = \frac{f'(\xi)}{(k+1)!}(x-a)^{k+1}$.

7.1 Metoda bisekcji

$p_0 = a, q_0 = b$
$x_n = \frac{p_n+q_n}{2}$

$$(p_n, q_n) = \begin{cases} (p_{n-1}, x_{n-1}) & f(p_{n-1}) \cdot f(x_{n-1}) < 0 \\ (x_{n-1}, q_{n-1}) & \text{wpp} \end{cases}$$

Zbieżność liniowa - k cyfr znaczących wymaga $O(k)$ kroków

7.2 Metoda Newtona

Działa, jeśli $f'(x^*) \neq 0$, czyli pierwiastek jest pojedyńczy i $(x_n - x^*)^2 \ll |x_n - x^*|$, jest się w jego okolicy.
$x_0 = ?,\ x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$
Zbieżność kwadratowa - k cyfr znaczących wymaga $O(\log k)$ kroków

7.3 Metoda siecznych

Pomocna, kiedy nie wiadomo, jak obliczyć pochodną.
$x_n = x_{n-1} - \frac{f(x_{n-1}) \cdot (x_{n-1} - x_{n-2})}{f(x_{n-1}) - f(x_{n-2})}$
Zbieżność ponadliniowa z $r = \frac{1+\sqrt{5}}{2}$

7.4 Metoda punktu stałego

Celem jest znaleźć punkt stały funkcji f.

Twierdzenie 7.4.1 (Banacha o punkcie stałym)**.** Niech $f : V \to V$ będzie odwzorowaniem w przestrzeni Banacha. Jeśli $\|f(x) - f(y)\| \leq \lambda \cdot \|x - y\|$ dla pewnego $0 \leq \lambda < 1$ (odzworowanie jest zwężające), to ma ono dokładnie jeden punkt stały i poniższa metoda zawsze do niego zbiega:

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

Czyli w $\mathbb{R}$ działa dla funkcji f, t.że $f' < 1$ na pewnym przedziale domkniętym - zbiega kwadratowo.

Rozdział 8

Optymalizacja nieograniczona

8.1 Ekstrema funkcji bitonicznych

8.1.1 Wyszukiwanie ternarne

Algorytm do szukania ekstremów funkcji bitonicznych, czyli ciągłych z jednym ekstremum:

(wersja dla maksimum)

```
function ternary(p, q)
    r = p + (q-p)/3
    s = q - (q-p)/3
    if f(r) < f(s)
        return ternary(r, q)
    else
        return ternary(p, s)
```

Przedział zmniejsza się do $\frac{2}{3}$ za pomocą dwóch wywołań f, czyli $\sqrt{\frac{2}{3}}$ razy na jedno obliczenie f.

8.1.2 Wyszukiwanie ze złotym podziałem

Czyli ulepszona wersja poprzedniego.

```
function golden(p, q)
    r = q - (q-p)*phi
    s = p + (q-p)*phi
    if f(r) < f(s)
        return golden(r, q)
    else
        return golden(p, s)
```

Wykorzystujemy obliczone wcześniej wartości f, więc przedział zmniejsza się o $\Phi \approx 0.618$ w jednym wywołaniu.

8.2 Wzór Taylora

Wzór Taylora dla jednej zmiennej:

$$f(x) \approx f(x_0) + f'(x_0)(x-a) + \frac{1}{2}f''(x_0)(x-x_0)^2$$

Wtedy ekstremum f jest gdzieś w okolicy $x_0 - \frac{f'(x_0)}{f''(x_0)}$.
Przybliżenie ze wzoru Taylora dla funkcji wielu zmiennych:

$$f(x) \approx f(x_0) + \nabla f(x_0) \cdot (x - x_0) + (x - x_0)^T \cdot H_f(x_0) \cdot (x - x_0)$$

$$\nabla f = \left[\frac{\partial f}{\partial x_1}, \cdots, \frac{\partial f}{\partial x_n}\right]$$

W punkcie stacjonarnym x_0 (spełniającym $\nabla f = 0$):

- jeśli $H_f(x_0)$ jest dodatnio określona, to w x_0 jest minimum.
- jeśli $H_f(x_0)$ jest ujemnie określona, to w x_0 jest maksimum.
- jeśli $H_f(x_0)$ jest nieosobliwa, ale nie ma maksimum ani minimum, to w x_0 jest punkt siodłowy.
- jeśli $H_f(x_0)$ jest osobliwa, mogą wystąpić sytuacje zdegenerowane (może też dowolna z powyższych).

8.3 Metoda gradient descent

Będąc w punkcie x_k, idziemy w kierunku wektora gradientu:

$$x_{k+1} = x_k - \nabla f(x_k) \cdot t,$$

gdzie t jest pewną stałą. Nie potrzeba wielu obliczeń f.

Można kluczyć lub przeskoczyć minimum.

8.3.1 Metoda gradientu prostego – wariant

Będąc w punkcie x_k, idziemy w kierunku wektora gradientu od razu do optimum na prostej:

$$h_k(t) = f(x_k - \nabla f(x_k) \cdot t)$$

$$x_{k+1} = \min_t \ h_k(t)$$

Szukamy minimum h_k za pomocą metod dla jednej zmiennej (Newton, wyszukiwanie ternarne). Daje to większą dokładność, ale też trzeba więcej razy obliczać f.

Rozdział 9

Układy równań – metody iteracyjne

9.1 Schemat działania

Przy ustalonej macierzy ortonormalnej Q wyznaczamy kolejne przybliżenia rozwiązania:

$$Qx_{n+1} = (Q - A)x_n + b,$$

czyli najpierw obliczamy $y = (Q - A)x_n$ + b, a potem rozwiązujemy układ równań $Qx = y$. Macierz Q powinna być niezbyt gęsta, żeby mnożenie było szybkie i łatwo odwracalna, żeby układ $Qy = c$ był rozwiązywalny:

$$x_{n+1} = Q^{-1}(Q - A)x_n + Q^{-1}b.$$

Jeśli oznaczymy $C = Q^1(Q - A)$, $b' = Q^{-1}b$, to $x_n = (I + C + C^2 + ... + C^n) \cdot b'$. Metoda jest więc zbieżna, jeśli $\rho(Q^{-1}(Q - A)) < 1$.

9.2 Metody Richardsona

Przyjmujemy Q = I, wtedy

$$x_{n+1} = x_n + \omega(b - Ax_n)$$

W wersji podstawowej $\omega = 1$. Metoda jest zbieżna, jeśli $0 < \omega < 2/\rho(A)$.

9.3 Metody Jacobiego

Przyjmujemy $Q = D$ diagonalną, taką jak główna przekątna w $A = D + R$.

$$x_{n+1} = D^{-1}(b - Rx_n)$$

Metoda jest zbieżna, jeśli $\rho(D^{-1}A) < 1$, w szczególności jeśli macierz ma dominującą przekątną,

9.4 Metody Gaussa-Seidela

Przyjmujemy za $Q = U$ dolny trójkąt macierzy A. Jeśli rozłożymy $A = L + U$, gdzie U ma zera na przekątnej:

$$x_{n+1} = L^{-1}(b - Ux_n)$$

Metoda jest zbieżna, jeśli A ma dominującą przekątną albo jeśli A jest symetryczna i dodatnio określona.

9.5 Metoda najszybszego spadku

Załóżmy, że macierz A jest symetryczna i dodatnio określona (zawsze można przejść na $A^TAx = A^Tb$). Żeby dostać rozwiązanie wystarczy znaleźć minimum funkcji

$$f(x) = \frac{1}{2}x^TAx - b^Tx$$

Gradient funkcji $\nabla f = Ax - b$ rzeczywiście zeruje się w rozwiązaniu, a hesjan to A.

$$x_{k+1} = x_k + \alpha(b - Ax_k)$$

Niech $d = b - Ax$. Szukamy minimum funkcji jednej zmiennej

$$g(\alpha) = f(x + \alpha d)$$

metodą najszybszego spadku:

```
    x = 0
    for k = 1, 2, 3, ...
        d = b - Ax
        alpha = <d,d> / <d,Ad>
        x = x + alpha * d
```

Rozdział 10

Interpolacja

Dla zadanych węzłów $x_0, \ldots, x_n$ oraz $y_0, \ldots, y_n$ chcemy znaleźć wielomian $W(x)$ stopnia n, dla którego $W(x_i) = y_i$.
Może istnieć tylko jeden taki wielomian i na pewno istnieje. Współczynniki są rozwiązaniem układu równań zadanego przez macierz Vandermonda.

Macierz Vandermonda, chociaż zawsze nieosobliwa, jest źle uwarunkowana.

Twierdzenie
Jeśli $f \in C^{n+1}[a, b]$ jest funkcją, którą interpolujemy wielomianem p w $x_0, \ldots, x_n \in [a, b]$, to:

$$|f(x) - p(x)| \leq \frac{1}{(n+1)!} \max |f(n+1)| \prod_{i=0}^{n} (x - x_i)$$

Dla niektórych funkcji przy interpolacji wielomianem w równomiernych węzłach, błąd przybliżenia rośnie, ile chce $\rightarrow$ zjawisko Rungego. Wybór węzłów ma znaczenie.

Zera wielomianów Czebyszewa $T_k(x) = \cos(k \arccos(x))$ to dobry wybór.

10.1 Interpolacja Lagrange'a

Znajdujemy wielomiany $P_0, \ldots, P_n$ t.że:

$$P_i(x_j) = 1[i = j]$$

$$P_i(x) = \frac{\prod_{s \neq i} (x - x_s)}{\prod_{s \neq i} (x_i - x_s)}$$

Wtedy $W(x) = \sum_{i=1}^{n} P_i(x) \cdot y_i$.

Dzielimy przez $\prod_{s \neq i} (x_i - x_s)$, a to może być prawie zerem.

10.2 Interpolacja Newtona

Definiujemy $R_i(x)$, takie żeby $W(x)$ można było łatwo wyrazić w bazie R_i:

$$R_i(x) = (x - x_0) \cdot \ \cdots \ \cdot (x - x_{i-1})$$

$$W_i(x) = W_{i-1}(x) + \alpha_i \cdot R_i(x) = y_0 + \alpha_1 \cdot R_1 + \cdots + \alpha_i \cdot R_i(x)$$

Współczynniki α_i dobieramy tak, żeby $W_i(x_i) = y_i$.

Algorytm Neville'a
Można ulepszyć interpolację Newtona, stosując algorytm dynamiczny - ilorazy różnicowe.

$$c[i, i] = f(x_i)$$

$$c[i, j] = \frac{c[i+1, j] - c[i, j-1]}{x_j - x_i}, \text{ dla i} < \text{j}$$

Wtedy $\alpha_i = c[0, i]$.

10.3 Funkcje sklejane

Dla ustalonych $n + 1$ węzłów interpolacji $x_0, \ldots, x_n$ i odpowiadających im wartości $y_i = f(x_i)$ szukamy funkcji g, t.że:

- g jest wielomianem k-tego stopnia na każdym przedziale $[x_i, x_{i+1}]$ (kolażem z kilku wielomianów)
- $g \in C^{k-1}$

Rozdział 11

Optymalizacja z ograniczeniami

11.1 Ekstrema warunkowe

Niech $f : \mathbb{R}^n \to \mathbb{R}$ będzie funkcją celu. Chcemy zminimalizować (zmaksymalizować) $f(x)$ na pewnym zbiorze domkniętym zadanym przez ograniczenia $g_i(x) = 0$, dla $i = 1, 2, \ldots, k$, gdzie $g_i : \mathbb{R}^n \to \mathbb{R}$. Czyli celem jest znaleźć ekstrema warunkowe. Definiujemy funkcję:

$$f_\lambda(x, \lambda) = f(x) - \sum_i \lambda_i \cdot g_i(x)$$

Ekstrema warunkowe f to ekstrema funkcji f_λ, czyli miejsca, gdzie gradient ∇f_λ się zeruje.

11.2 Programowanie liniowe

Zadanie programowania liniowego (ZPL) polega na maksymalizacji $f(x) = c^T x$, przy założeniach:

- $Ax \leq b$
- $x \geq 0$

Obszar dopuszczalny (spełniający ograniczenia) jest hiperwielościanem wypukłym. Jeśli ZPL ma rozwiązanie, to musi być któryś z jego wierzchołków.
Zadanie podstawowe przekształcamy na dualne tak:

$$\begin{array}{lcl} \max c^T x & & \min b^T y \\ Ax \leq b & \to & A^T y \geq c \\ x \geq 0 & & y \geq 0 \end{array}$$

W zadaniu dualnym:

- max przechodzi w min i odwrotnie
- równania zamieniają się na zmienne

- ograniczenia $\leq$ w warunkach przechodzą na warunki $y_i \geq 0$
  (nierówności $\leq$ tylko z dodatnim współczynnikiem)
- ograniczenia $\geq$ w warunkach przechodzą na warunki $y_i \leq 0$
  (nierówności $\geq$ z ujemnym współczynnikiem)
- ograniczenia równościowe przechodzą na zmienne nieograniczone $y_i \in \mathbb{R}$
  (równości z dowolnym współczynnikiem)

Jeśli jedno zadanie jest nieograniczone, to drugie musi być niespełnialne. Jeśli zadanie podstawowe (prymalne) jest niespełnialne, to dualne jest nieograniczone lub niespełnialne, i odwrotnie.

Silne twierdzenie o dualności
Rozwiązanie zadania dualnego jest równe rozwiązaniu zadania prymalnego.

11.3 Problem ogólny

Twierdzenie Karusha-Kuhna-Tuckera
Jeśli x^* jest minimum funkcji f z ograniczeniami równościowymi g_i i nierównościowymi h_j, to:

- $\nabla f(x^*) - \sum_i \lambda_i \nabla g_i(x^*) - \sum -j \, \mu_j \nabla h_j(x^*) = 0$
- $g_i(x^*) = 0, h_j(x^*) \geq 0$
- $\mu_j h_j(x^*) = 0$
- $\mu_j \geq 0$

11.4 Metoda simplex

Zaczynając od dowolnego wierzchołka obszaru dopuszczalnego, dopóki to możliwe, przesuwamy się do wierzchołka, który ma lepszą wartość funkcji celu. Jeśli rozwiązanie początkowe $x = 0$ nie spełnia nierówności, to dodajemy nową zmienną w, którą odejmujemy od prawej strony każdej z nierówności i próbujemy zmaksymalizować (z nadzieją, że się wyzeruje).

Metoda simplex może być wykładnicza i może się zapętlić. Własność stopu zapewnia Reguła Blanda - wybieramy zmienną bazową i niebazową o najmniejszym indeksie.

Rozdział 12

Różniczkowanie i całkowanie numeryczne

12.1 Ogólny schemat kwadratury

Mając dane wartości funkcji $f(x_1), \ldots, f(x_n)$, dla pewnych punktów z $[a, b]$, można liczyć przybliżoną wartość całki jako

$$\int_a^b f = \sum A_i \cdot f(x_i),$$

czyli (tu intuicja) interpolujemy funkcję wielomianem stopnia n w ustalonych węzłach i całkujemy ten wielomian.

12.2 Podstawowe kwadratury

Jeśli $x_1, \ldots, x_n$ są równo położone na przedziale $[a, b]$, tę metodę nazywa się kwadraturą Newtona-Cotesa.

Kwadratury Newtona-Cotesa

- zamknięte (a i b są pierwszym i ostatnim węzłem interpolacji)
- otwarte (a i b nie są węzłami interpolacji).

- Metoda prostokątów – kwadratura otwarta dla $n = 1$

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

  Błąd: $O((b - a)^3)$

- Metoda trapezów – kwadratura zamknięta dla $n = 1$

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

Błąd: $O((b-a)^3)$

- Metoda Simpsona – kwadratura zamknięta dla $n = 2$

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

Błąd: $O((b-a)^5)$

- Metoda Simpsona 3/8 – kwadratura zamknięta dla $n = 2$

$$\int_a^b f = \frac{b-a}{8} \cdot \left(3f\left(\frac{2a+b}{3}\right) + 3f\left(\frac{a+2b}{3}\right) + f(b) \right)$$

Dla dużych przedziałów błąd jest duży, więc lepiej zastosować kwadratury złożone, czyli dla podziału na mniejsze przedziały. Przykład dla prostokątów:

$$\sum \Delta x \cdot f\left(\frac{x_i + x_{i+1}}{2}\right)$$

dla przedziałów $[x_i,\ x_{i+1}]$ wielkości Δx.
Dla kwadratur złożonych błędy wynoszą $O(\Delta x^2)$ (prostokąty, trapezy) i $O(\Delta x^4)$ (Simpson).

12.3 Przybliżone różniczkowanie

- **Na zwykły użytek:** $f'(x) \approx \frac{f(x+h)-f(x)}{h}$
  $\rightarrow$ błąd $O(h)$
- **Lepiej:** $f'(x) \approx \frac{f(x+h)-f(x-h)}{2h}$
  $\rightarrow$ błąd $O(h^2)$
- **Jeszcze lepiej:** $f'(x) \approx \frac{3}{2}D\left(\frac{h}{3}\right) - \frac{1}{2}D(h)$, dla $D(h) = \frac{f(x+h)-f(x)}{h}$ (ekstr. Richardsona)
  $\rightarrow$ błąd $O(h^2)$
- **Dla wtajemniczonych:** $f'(x) \approx \frac{1}{h} \cdot \left[\frac{1}{12}f(x-2h) - \frac{2}{3}f(x-h) + \frac{2}{3}f(x+h) - \frac{1}{12}f(x+2h) \right]$
  $\rightarrow$ błąd $O(h^4)$

W niektórych z metod może się pojawić dzielenie przez $h \approx 0$.

Rozdział 13

Transformata Fouriera

13.1 Rozwinięcie w szereg Fouriera

$$f(x) = \frac{a_0}{2} + \sum_{k\geq 1} a_k \cdot \cos(kx) + \sum_{k\geq 1} b_k \cdot \sin(kx)$$

$$a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cdot \cos(kx)$$

$$b_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cdot \sin(kx)$$

13.2 Transformata Fouriera

Celem jest ustalić częstotliwość danej funkcji $f : \mathbb{R} \to \mathbb{R}$, przyjmując, że jeśli częstotliwość jest równa ξ, to wartość $\int f(x) \cdot \cos(2\pi\xi x)$ będzie raczej większa niż mniejsza. Czyli zamiast wartości funkcji w punktach, pamiętamy z jakich częstotliwości się składa. Definiujemy:

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x) \cdot e^{2\pi i \xi x} = \int_{-\infty}^{\infty} f(x) \cdot \cos(2\pi\xi x)\, dx - i \cdot \int_{-\infty}^{\infty} f(x) \cdot \sin(2\pi\xi x)\, dx$$

Transformata Fouriera jest odwracalna, czyli $\hat{\hat{f}}(x) = f(-x)$.

13.2.1 Dyskretna transformata Fouriera

Jest potrzebna, bo komputery nie operują na ciągłych funkcjach oraz znamy funkcję tylko w pewnych punktach. Przyjmujemy, że próbkowanie jest na przedziale $[0, N)$ w punktach całkowitych.

$$\hat{f}(\xi) = \sum_{k=0}^{N-1} f(k)\, e^{2\pi i \xi \cdot k}$$

Ustalamy częstotliwość bazową $\xi_0 = \frac{1}{N}$ i ograniczmy się tylko do wielokrotności ξ_0:

$$\hat{f}\left(\frac{j}{N}\right) = \sum_{k=0}^{N-1} f(k)\, e^{2\pi i \cdot jk/N}$$

Czyli zamieniamy ciąg wartości funkcji $f(0), \ldots, f(N-1)$ na ciąg wartości transformaty $\hat{f}(0), \hat{f}(1/N), \ldots, \hat{f}((N/1)/N)$. Pierwszy ciąg oznaczmy przez $(a_0, \ldots, a_{N1})$, drugi przez $(\hat{a}_0, \ldots, \hat{a}_{N1})$. Przyjmujemy, że $\omega = e^{2\pi i/N}$ to N-ty pierwiastek z jedności. Stąd wzór to:

$$\hat{a}_j = \sum_{k=0}^{N-1} a_k \omega^{jk}$$

Traktujemy $(a_0, \ldots, a_{N-1})$ jak współczynniki wielomianu A i zamieniamy na ciąg $(\hat{a}_0, \ldots, \hat{a}_{N-1})$ $= (A(1), A(\omega), \ldots, A(\omega^{N-1}))$.
Szybka transformata Fouriera (FFT) daje złożoność $O(N \log N)$.

13.3 Kompresja fal

13.3.1 Kompresja dźwięku

Pomysł: Rozbić funkcję (sygnał) na małe przedziały i zamienić za pomocą DFT na częstotliwości. Zapamiętuje się tylko najmocniejsze, po czym kompresuje wybranym standardowym, bezstratnym algorytmem (np. Huffman).

Pojawiają się problemy na granicach przedziałów.

Aby ich uniknąć, zamiast transformaty Fouriera można użyć podobnej transformaty cosinusowej, na częściowo nakładających się przedziałach. Jest obliczalna w takim samym czasie. Najbardziej znany wariant to Zmodyfikowana Transformata Cosinusowa (MDCT), używana m. in. w kompresji MP3.

$$(x_0, \ldots, x_{2n-1}) \to (X_0, \ldots, X_{n-1})$$

$$X_k = \sum x_j \cos\left(\frac{\pi}{n}\left(j + \frac{n+1}{2}\right) \cdot \left(\frac{k+1}{2}\right)\right)$$

13.3.2 Kompresja obrazu

Kompresja JPEG działa na podobnej zasadzie. Dzielimy obrazek na bloki i zamieniamy transformatą na częstotliwości. Potem zmniejszamy dokładność zapisu częstotliwości i na koniec kompresujemy bezstratnie (np. Huffman).

Rozdział 14

Równania różniczkowe

14.1 Ogólny schemat rozwiązywania

Wersja prosta (Cauchy'ego): Szukamy funkcji $y : [0, \infty) \to \mathbb{R}^n$, spełniającej $y' = F(y, t)$ oraz $y(0) = c$.
Wersja ogólna: Szukamy funkcji $y : [0, \infty) \to \mathbb{R}^n$, spełniającej $F(y^{(k)}, y^{(k-1)}, \ldots, y, t) = 0$ oraz warunki początkowe $y(0) = c_0, y'(0) = c_1, \ldots$

Nie każde równanie musi mieć rozwiązanie, a jeśli ma rozwiązanie, to niekoniecznie jedno.

Modelowe równanie
Przyjmujemy $y' = F(y)$ liniowe, czyli $y' = ay$. Wtedy rozwiązanie to $y = Ce^{at}$. Problem jest stabilny dla $a \leq 0$.

14.2 Równania o zmiennych rozdzielonych

Równanie jest postaci $y' = f(x) \cdot g(y)$.

1. Sprowadzamy do postaci $\frac{dy}{g(y)} = f(x)\, dx$.
2. Całkujemy stronami.
3. Wyznaczamy y.
4. Ustalamy wartość stałej na podstawie założeń.

14.3 Równania jednorodne liniowe pierwszego rzędu

Równanie jest postaci $y' + y \cdot p(x) = q(x)$.

1. Obliczamy czynnik $\mu(x) = e^{\int p(x)\, dx}$.

2. Przemnażamy obie strony oryginalnego równania przez $\mu(x)$.
3. Otrzymujemy równanie $\frac{d}{dx}(y \cdot \mu(x)) = \mu(x) \cdot q(x)$.
4. Całkujemy stronami.
5. Wyznaczamy y.
6. Ustalamy wartość stałej na podstawie ewentualnych założeń.

14.4 Metoda Eulera

Intuicja: Narysować strzałki wskazujące kierunek funkcji w punkcie, po czym pójść po strzałkach. Dla równania $y' = F(y)$ wyliczamy kolejne wartości y, zaczynając od $y_0 = y(0)$.

$$y_{k+1} = y_k + h \cdot F(y_k)$$

I tak rozwiązujemy równanie różniczkowe numerycznie. Dla bardzo małych h będzie stabilne, dla większych nie. Problemy, dla których potrzeba bardzo małych h do dobrych wyników są *sztywne*.

Niejawna metoda Eulera
Zaczynając od $y_0 = y(0)$

$$y_{k+1} = y_k + h \cdot F(y_{k+1})$$

Daje lepszą stabilność niż podstawowy wariant.