Algorytmy Numeryczne

Notatki

„Nie będę tego liczył bo bez jaj. Napiszę wynik."

Maciej Mikołajczak
Kraków, 2024/2025

Spis treści

1. Arytmetyka komputerowa i błędy obliczeń

2024-10-03

Najważniejsze błędy w obliczeniach to błędy zaokrągleń i ograniczenia możliwości zapisu.

Możliwe reprezentacje:

- ułamki – są dokładne, ale wolne, nie wszystko jest wymierne, nie ma pierwiastków, exp, etc. do tego trzymane liczby szybko rosną
- reprezentacja stałoprzecinkowa – działa jak wszystko jest tej samej precyzji, jak mamy inną precyzję niż zakłada typ to marnujemy miejsce, to tego łatwo stracić dokładność (jak mamy tylko 4 miejsca po przecinku to pomnożenie przez 10^{-4} psuje wszystko, nawet jak potem pomnożymy przez 10^4)
- reprezentacja zmiennoprzecinkowa – ta najlepsza

Reprezentacja zmiennoprzecinkowa: postać $\pm c.cc\ldots ccc \times B^w$, najczęściej $B = 2$. Do tego zapis znormalizowany (pierwsza w zapisie jedynka), czyli postać $2^w \cdot (1 + M)$, w to cecha, $M \in (0, 1)$ to mantysa.

Jak trzymać specjalne liczby?

- zero jako $\pm 00\ldots00000$
- nieskończoność jako $\pm 111\ldots1\ 0\ldots000$ (cecha i mantysa)
- NaN jako $11\ldots1\ x\ldots xx$
- liczby nieznormalizowane (subnormal/denormal) czyli za małe $00\ldots0\ x\ldots xx$

Liczby rzeczywiste zapisane zmiennoprzecinkowo mają następniki, im większe liczby zapisujemy tym mniejsza odległość między kolejnymi – zmiana jednej cyfry na końcu mantysy znaczy więcej przy dużej cesze.

Działania na liczbach nie są dokładne, zawsze jest przybliżenie, dlatego nie można stosować równości dla liczb rzeczywistych.

Błędy obliczeń (wartość dokładna x i przybliżona x^*):

- bezwzględny $|x - x^*|$
- względny $|\frac{x-x^*}{x}|$

Błąd względny wynikający z zaokrąglenia nie przekracza $\varepsilon = 2^{-\text{bity mantysy}}$ dla liczb mieszczących się w arytmetyce (taka jest zmiana jak ostatni bit mantysy się zmieni). Mamy $\frac{2^w(M+\varepsilon)-2^wM}{2^w} = \varepsilon$.

Dla wartości obliczonej x^* mamy $(1-\varepsilon)x \le x^* \le (1+\varepsilon)x$. Jak mnożymy z takim błędem on nie zmienia się bardzo. $x^*y^* \le xy\left(1 + 2\varepsilon + \varepsilon^2\right)$, czyli błąd w okolicy 2ε.

Dla dodawania liczb dodatnich też jest dobrze, ale przy różnych znakach jest źle. Dla $x = 1.24 \pm 0.01$ i $y = 1.20 \pm 0.01$ jest $x - y = 0.04 \pm 0.02$, czyli ogromny błąd względny – utrata cyfr znaczących (catastrophic cancellation). Dlatego liczenie $\sqrt{x^2+1} - 1$ jest gorsze niż $\frac{x^2}{\sqrt{x^2+1}+1}$, mimo że to to samo.

Dużo stabilnych obliczeń też jest problemem. $O(n)$ dodawań ma błąd $O(n\varepsilon)$. Przykład: dodanie małej liczby do dużej jej nie zmieni, powtórzenie tego wiele razy da ogromny błąd.

Algorytm Kahana dodaje w $O\left(\varepsilon + n\varepsilon^2\right)$.

Licząc poprawność rozwiązania $x^2 = 2$ możemy zastosować forward error (błąd naturalny), czyli $|\sqrt{2}-x^*|$, ale trzeba znać ten pierwiastek. Lepiej zastosować backward error $|x^{*2} - 2|$ (błąd wsteczny). Podobnie przy rozwiązywaniu równań typu $f(x) = 0$ obliczamy błąd wsteczny $|f(x^*)|$.

Definicja 1. Problem jest dobrze uwarunkowany, jeśli małe zaburzenia w danych wejściowych powodują małe zaburzenia w wyniku. Problemy źle uwarunkowana najprawdopodobniej nie mają sensownego rozwiązania numerycznego.

Przykład (wielomian Wilkinsona). $W(x) = (x-1)(x-2)\ldots(x-20)$. Lekkie zaburzenie we współczynnikach po wymnożeniu może usunąć pierwiastek (dwa się połączą). Problem liczenia pierwiastków wielomianów jest źle uwarunkowany.

Definicja 2. Dla liczby t i funkcji $f(x)$ współczynnik uwarunkowania to taka liczba $A = A(f,t)$, dla której $|f(t)-f(t^*)| \leq A\cdot|t-t^*|$. Dla rozsądnych funkcji rzeczywistych współczynnik uwarunkowania jest bliski $f'(t)$.

Definicja 3. Dla zadanego równania $g(x) = 0$ należy wyznaczyć x. Współczynnik uwarunkowania to taka liczba $A = A(g,x)$, dla której $|x-x^*| \leq A\cdot|g(x)-g(x^*)|$. Dla rozsądnych funkcji rzeczywistych współczynnik uwarunkowania jest bliski $\frac{1}{g'(x)}$.

Definicja 4. Algorytm jest numerycznie stabilny, jeśli małe błędy na wejściu powodują małe zmiany wyniku. To własność algorytmu, niestabilny algorytm można zastąpić lepszym.

Definicja 5. Algorytm $A(x)$ przybliżający wartość $f(x)$ jest numerycznie poprawny, jeśli dla każdego x istnieje takie x^*, że x^* dobrze przybliża x, a $A(x^*)$ dobrze przybliża $f(x)$.

2. Eliminacja Gaussa

2024-10-10

Twierdzenie 1 (Kronecker-Capelli)**.** Niech A będzie macierzą $m \times n$, b wektorem rozmiaru m. Niech $r = \operatorname{rank} A$, $s = \operatorname{rank}[A|b]$. Układ $Ax = b$ posiada rozwiązanie wtedy i tylko wtedy, gdy $r = s$. Rozwiązanie jest jednoznaczne, gdy $r = n$, inaczej rozwiązania stanowią przestrzeń o rozmiarze $n - r$.

Odjęcie i-tego wiersza od j-tego ze współczynnikiem c to przemnożenie z lewej przez macierz

$$T_{i,j,c} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & -c & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix},$$

gdzie $-c$ leży w j-tym wierszu i i-tej kolumnie. Łatwo zauważyć, że $T_{i,j,c}^{-1} = T_{i,j,-c}$.

Definicja 6. Rozkład LU macierzy A to takie macierze L, U, że $LU = A$ oraz L jest dolnotrójkątna, a U górnotrójkątna. Macierz U jest wyznaczana podczas eliminacji Gaussa, natomiast L to wymnożone macierze odejmowania od siebie odpowiednich wierszy (a właściwie ich odwrotności – $L_1 \cdot \ldots \cdot L_k A = U \implies A = L_k^{-1} \cdot \ldots \cdot L_1^{-1} U = LU$). Zatem macierz L można wyznaczyć dając jedynki na przekątną i wsadzając w odpowiednie miejsca współczynniki, z którymi odejmujemy wiersze podczas eliminacji Gaussa.

Standardowy algorytm eliminacji Gaussa nie zadziała dla macierzy $\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$ – mnożenie przez 0. Będzie źle działał również dla macierzy $\begin{bmatrix} \varepsilon & 1 \\ 1 & 1 \end{bmatrix}$, gdzie $\varepsilon = 10^{-20}$. Otrzymamy $LU = \begin{bmatrix} \varepsilon & 1 \\ 1 & 0 \end{bmatrix}$. Aby poradzić sobie z tymi problemami (niestabilność numeryczna i nieistnienie rozkładu LU) stosujemy pivoting.

Definicja 7 (Norma L_p)**.** Dla wektora $x \in \mathbb{R}^n$ definiujemy normę L_p jako

$$\|x\|_p = \sqrt[p]{\sum_{i=1}^{n} x_i^p x_i^p}.$$

w szczególności ważne są normy L_1 (manhattańska), L_2 (euklidesowa) i L_∞ (maksimum). Wszystkie normy na $\mathbb{R}^n$ są równoważne (ograniczają się wzajemnie przez stałą multiplikatywną).

Jeśli podczas rozwiązywania układu $Ax = b$ dostaliśmy zaburzoną wartość b^*, to wyliczymy pewne x^* spełniające $Ax^* = b^*$. Zachodzi

$$\frac{\|x - x^*\|}{\|x\|} \leq \|A\|\|A^{-1}\| \cdot \frac{\|b - b^*\|}{\|b\|}.$$

Wartość $\kappa(A) = \|A\| \cdot \|A^{-1}\|$ nazywamy wskaźnikiem uwarunkowania macierzy. Duża wartość κ czyni układ trudnym.

Uwagi: macierz odwrotna macierzy prawie osobliwej zawiera duże współczynniki (czyli ma dużą normę), nierówność we współczynniku uwarunkowania zazwyczaj jest bliska równości.

Obliczenie odwrotności macierzy $n \times n$ jest tak samo trudne, jak wymnożenie macierzy.

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} + A^{-1}BS^{-1}CA^{-1} & -A^{-1}BS^{-1} \\ -S^{-1}CA^{-1} & S^{-1} \end{bmatrix},$$

gdzie $S = D - CA^{-1}B$. Taki algorytm jest tragiczny w praktyce, ale pokazuje, że da się wykonać odwracanie macierzy za pomocą dwóch mniejszych podproblemów i kilku mnożeń, co daje nam złożoność $T(n) = 2T\left(\frac{n}{2}\right) + M(n)$, czyli $T(n) = M(n)$, gdzie $M(n)$ to złożoność mnożenia macierzy.

3. Rozkład Cholesky'ego

2024-10-17

Mając zadany nadokreślony układ równań $Ax = b$ nie możemy go rozwiązać, ale możemy poszukać takiego x^*, który minimalizuje $\|Ax^* - b\|_2$ (czyli jest najbliżej b jak się da).

Przykład. Mamy wielkość b i chcemy ją dopasować tak, żeby była liniowo zależna od $a_1, \ldots, a_n$. Robimy m odczytów i chcemy znaleźć takie parametry, które dadzą jak najlepszą wartość.

Twierdzenie 2. Jeśli $Ax = b$ to nadokreślony układ równań, to rozwiązanie x^* minimalizujące $\|Ax^* - b\|_2$ spełnia równanie $A^TAx^* = A^Tb$. Zatem do znalezienia minimum $\|Ax^* - b\|_2$ wystarczy rozwiązać równanie z macierzą A^TA, która jest kwadratowa.

Dowód. Liczymy gradient $\|Ax^* - b\|_2^2$ i wychodzi. □

Uwaga. Układ równań $A^TAx = A^Tb$ jest problematyczny, gdyż $\text{cond}\left(A^TA\right) = \text{cond}\left(A^2\right)$ (przez cond oznaczmy współczynnik uwarunkowania). Zatem nasz problem staje się znacznie gorzej uwarunkowany (gdy działamy rozkładem LU).

Definicja 8. Macierz C jest dodatnio określona, jeśli dla każdego $x \neq 0$ mamy $x^T \cdot C \cdot x > 0$.

Definicja 9. Macierz C jest dodatnio półokreślona, jeśli dla każdego x mamy $x^T \cdot C \cdot x \geq 0$.

Uwaga. Dla każdej macierzy A, macierz A^TA jest dodatnio półokreślona, bo $x^T \cdot A^TA \cdot x = (Ax)^T Ax = \langle Ax, Ax\rangle = \|Ax\| \geq 0$. Równość może zachodzić tylko, gdy A ma nietrywialne jądro, czyli jest osobliwe.

Twierdzenie 3. Dla macierzy A, która jest symetryczna i dodatnio określona, istnieje rozkład LU zwany rozkładem Cholesky'ego w którym $U = L^T$. Algorytm rozkładu Cholesky'ego jest stabilniejszy numerycznie niż standardowy rozkład LU za pomocą eliminacji Gaussa i ma lepszą stałą.

Dowód. Rozważmy macierz $A = [a_{ij}]_{i,j\in[n]}$. Mamy $a_{11} = e_1^TAe_1 > 0$ z dodatniego określenia macierzy. Zatem istnieje $\sqrt{a_{11}}$. Dzielimy pierwszą kolumnę przez $\sqrt{a_{11}}$ i robimy dla niej krok eliminacji Gaussa. Następnie robimy to samo dla pierwszego wiersza. Te operacje to macierz $E \cdot A \cdot E^T$ dla dolnotrójkątnego E. Działając indukcyjnie dostajemy $E_1 \cdot E_2 \cdot \ldots \cdot E_n \cdot A \cdot E_n^T \cdot \ldots \cdot E_2^T \cdot E_1^T = I$, a więc $A = L \cdot L^T$. □

Algorytm 1. Rozkład Cholesky'ego macierzy A wyznaczamy algorytmem dynamicznym, który liczy wartości w macierzy L. Dane są one wzorami:

$$L_{ij} = \frac{A_{ij} - \sum_{k=1}^{j-1} L_{ik}L_{jk}}{L_{jj}} \text{ dla } j < i$$

$$L_{ii} = \sqrt{A_{ii} - \sum_{k=1}^{j-1} L_{ik}^2}.$$

Dowód. Korzystamy z definicji mnożenia i warunku $A = LL^T$. □

4. Ortogonalizacja

2024-10-24

Definicja 10. Macierz Q jest ortogonalna, jeśli $Q^TQ = I$. Analogicznie, kolumny Q to wektory ortonormalne. Dla macierzy kwadratowych iloczyn macierzy ortogonalnych jest ortogonalny, a macierz ortogonalna jest izometrią, czyli zachowuje normę wektorów i kąty między nimi. Układ równań $Qx = b$ jest łatwo rozwiązać, bo $x = Q^Tb$.

Definicja 11 (Rozkład QR). Rozkładem QR macierzy A nazywamy takie dwie macierze Q, R, że $A = QR$ oraz Q jest ortogonalna, a R górnotrójkątna. W szczególności macierz A nie musi być kwadratowa.

Algorytm 2 (Grama-Schmidta). Niech macierz A ma kolumny $a_1, a_2, \ldots, a_n$. Wyliczamy kolumny macierzy Q za pomocą takiego algorytmu:

for i $= 1, \ldots, n$ **do**
  $q_i = a_i - \sum_{j=1}^{i-1} \langle a_i, q_j \rangle q_j$
  $q_i = \frac{q_i}{\|q_i\|}$
end for

Q jest ortogonalna. Zachodzi $A = QR$ dla pewnej macierzy R, którą można wyznaczać podczas wykonanie tej pętli lub pod koniec algorytmu jako $R = Q^TA$.

Dowód. Każdy kolejny wektor powstaje odejmując rzut na płaszczyznę tworzoną przez poprzednie wektory. Zatem powstałe wektory będą ortogonalne. □

Uwaga. Algorytm Grama-Schmidta jest *katastrofalnie* niestabilny numerycznie, przez to raczej niepraktyczny. Nieco lepszy jest zmodyfikowany algorytm Grama-Schmidta.

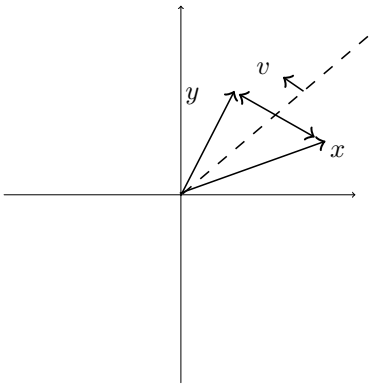
Algorytm 3 (Zmodyfikowany Grama-Schmidta). Będziemy pozbywać się części zgodnych z kolejnymi wektorami ortogonalnymi zaraz po ich wyznaczeniu.

for $i = 1, \ldots, n$ **do**
  $q_i = \frac{a_i}{\|a_i\|}$
  for $j = i + 1, \ldots, n$ **do**
    $a_j = a_j - \langle a_j, q_i \rangle q_i$
  end for
end for

Uwaga. Chcąc rozwiązać układ nadokreślony $Ax = b$ chcemy rozwiązać układ $A^TAx = A^Tb$, co po zastosowaniu rozkładu QR daje nam $R^TQ^TQRx = R^TQ^Tb$, co skraca się do $Rx = Q^Tb$, a to jest łatwo rozwiązać.

Twierdzenie 4 (Odbicie Householdera). Mając zadane wektory x, y o równej normie możemy przekształcić wektor x na y za pomocą odbicia poprzez przekształcenie $x \to x - 2v\langle v, x \rangle$, gdzie $v = \frac{u}{\|u\|}$ dla $u = y - x$. Macierz tego przekształcenia to $I - 2vv^T$ i jest ona ortogonalna.

Dowód. Wektor v jest jednostkowym wektorem leżącym wzdłuż różnicy x i y. Zatem $v\langle v, x\rangle$ jest składową x w kierunku zgodnym z $y - x$. Odejmując ją dwukrotnie zmieniamy jej znak, czyli odbijamy x na y. Macierz tego przekształcenia jest ortogonalna, bo opisuje odbicie. □



Rysunek 1: Odbicie Householdera

Algorytm 4 (Odbicia Householdera)**.** Wyznaczymy rozkład QR macierzy A za pomocą macierzy Householdera. Najpierw wyznaczamy macierz Q_1, która przekształca kolumnę a_1 na wektor $\|a_1\|e_1$. Mamy $Q_1 = I - 2v_1v_1^T$ dla pewnego v_1. Następnie działamy rekurencyjnie. Rozważamy macierz Q_1A pozbawioną pierwszej kolumny i wiersza. Powtarzamy poprzedni krok, dostajemy macierz $Q_2 = I - 2v_2v_2^T$, która ma o jeden wymiar mniej niż Q_1. Zwiększamy jej wymiar dopisując na początek wyzerowaną kolumnę i wiersz z jedynką na przekątnej (ewentualnie dopisujemy do wektora v_2 jedno zero na początek i odpowiednio powiększamy I). Zauważmy, że przez macierz typu $I-2vv^T$ można mnożyć w czasie $O\left(n^2\right)$, więc całość ma złożoność $O\left(n^3\right)$. Ostatecznie dostajemy macierz $Q = Q_n \cdot \ldots \cdot Q_1$ taką, że $QA = R$, a więc $A = Q^TR$.

Uwaga. Jeśli mamy do rozwiązania układ $Ax = b$, to wystarczy domnażać obie strony przez kolejne macierze, aż dostaniemy $Rx = b'$. Zamiast sprowadzać wektor x na $\|x\|e_1$ można równie dobrze na $-\|x\|e_1$. Stabilniejsze numerycznie jest wybranie tego samego znaku, co pierwsza współrzędna x.

Twierdzenie 5 (Obrót Givensa)**.** Mamy macierz Givensa G_{ij} z $i > j$ zdefiniowaną w następujący sposób: $(G_{ij})_{ii} = (G_{ij})_{jj} = \cos\theta$, $(G_{ij})_{ji} = -(G_{ij})_{ij} = -\sin\theta$, na reszcie przekątnej są 1 a w reszcie macierzy 0. Pomnożenie z lewej strony przez taką macierz obraca jej kolumny w płaszczyźnie ij o kąt θ w kierunku przeciwnym do ruchu wskazówek zegara.

$$G_{ij} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

Algorytm 5 (Obroty Givensa)**.** Wyznaczymy rozkład QR macierzy A za pomocą macierzy Givensa. Będziemy wykonywać kolejne obroty i zerować nimi wartości w macierzy. Jeśli chcemy wyzerować A_{ij}, to możemy obrócić w płaszczyźnie ij o kąt θ taki, że $\cos\theta = \frac{A_{jj}}{\sqrt{A_{jj}^2+A_{ij}^2}}$ oraz $\sin\theta = \frac{-A_{ij}}{\sqrt{A_{jj}^2+A_{ij}^2}}$, co obróci wartość z i-tego wiersza do j-tego.

Uwaga. Ta metoda ma gorszą stałą niż odbicia Householdera, ale działa lepiej dla macierzy, które mają dużo zer. Do tego dobrze się zrównolegla.

Dla niedookreślonego układu równań $Ax = b$ mamy wiele rozwiązań. Jest wiele wektorów minimalizujących $\|Ax - b\|$. Możemy szukać takiego minimalizującego $\|Ax - b\| + \alpha\|x\|^2$, czyli możliwie małego rozwiązania.

Definicja 12 (Regularyzacja Tichonowa)**.** Zamiast układu równań $Ax = b$ możemy rozwiązywać układ $A^TAx + \alpha Ix = A^Tb$, który dla $\alpha > 0$ jest dobrze określony (bo macierz $A^TA + \alpha I$ jest dodatnio określona). Takie przejście pomaga przy układach niedookreślonych i czasem przy trudnych numerycznie.

5. Wartości własne

2024-10-24

Definicja 13. Widmem macierzy nazywamy zbiór wszystkich jej wartości własnych.

Definicja 14. Promieniem spektralnym macierzy nazywamy moduł jej największej wartości własnej.

Uwaga. Wartości własne macierzy A to pierwiastki wielomianu $W(t) = \det(A - It)$. Z tego wynika, że wartości własne macierzy rzeczywistej mogą być zespolone i jeśli λ jest jedną z nich, to $\overline{\lambda}$ też. Każda macierz ma przynajmniej jedną wartość własną, ale nie da się wyznaczyć dokładnych wartości własnych dowolnej macierzy w skończonej liczbie kroków (nie ma wzorów na pierwiastki wielomianu stopnia większego niż 5).

Przykład. Mamy zbiór stanów $s_1, \ldots, s_n$. Układ w danej chwili jest w dokładnie jednym z nich. W każdej chwili stan układu może się zmienić. Przejście z s_i do s_j ma prawdopodobieństwo p_{ij}. Taki proces losowy nazywamy łańcuchem Markowa. Stan graniczny układu (po wielu krokach) musi spełniać $v = Pv$, gdzie $P = [p_{ij}]_{ij}$, zatem jest wektorem własnym o wartości własnej 1.

Przykład. Graf $G = (V, E)$ jest c-expanderem, jeśli dla każdego zbioru $S \subseteq V$ takiego, że $\|S\| \leq \frac{\|V\|}{2}$ zachodzi $\|N(S) \setminus S\| \geq c\|S\|$. Dołączanie sąsiadów do zbioru rozszerza go wykładniczo. Expandery mają ładne własności teoretyczne. Można pokazać, że macierz sąsiedztwa grafu d-regularnego ma dominującą wartość własną d. Jeśli druga co do wielkości wartość własna jest równa $\lambda < d$, to graf jest c-expanderem dla $c = \frac{d-\lambda}{2d}$.

Twierdzenie 6. Macierz A jest diagonalizowalna, jeśli posiada n liniowo niezależnych wektorów własnych. Wówczas $A = P^{-1}DP$, gdzie D jest diagonalna i ma wartości własne na przekątnej.

Definicja 15. Pojęcie defective matrix oznacza macierz niediagonalizowalną. Analogicznie macierz diagonalizowalna jest non-defective.

Uwaga. Zmiana bazy nie zmienia wartości własnych. Jeśli $Ax = \lambda x$, to dla $y = P^{-1}x$ mamy $P^{-1}APy = P^{-1}\lambda x = \lambda y$. Mówimy, że macierze A i $P^{-1}AP$ są sprzężone.

Uwaga. Z istnienia postaci Jordana wynika fakt, że ciąg A^k jest "rozbieżny", jeśli największa wartość własna A jest > 1. Jeśli jest < 1, to ciąg jest "zbieżny".

Twierdzenie 7. Jeśli macierz A jest hermitowska, to jej wartości własne są rzeczywiste, a wektory własne ortogonalne.

Dowód. Weźmy wektory własne x, y. Mamy

$$\langle \lambda x, y \rangle = \langle Ax, y \rangle = (Ax)^H y = x^H Ay = \langle x, Ay \rangle = \langle x, \mu y \rangle .$$

Z tego wynika $\left(\overline{\lambda} - \mu\right) \langle x, y \rangle = 0$, czyli dla $x = y$ jest $\lambda = \overline{\lambda}$, a dla $x \neq y$ jest $\langle x, y \rangle = 0$. □

Algorytm 6 (Iteracja prosta)**.** Dla wektora v i macierzy A definiujemy $v_0 = v$ oraz $v_{k+1} = \frac{Av_k}{\|Av_k\|}$. Jeśli λ_1 jest dominującą wartością własną A (czyli większą na moduł niż każda inna), to v_k zbiega do x_1 (wektora własnego odpowiadającego λ_1).

Dowód. Niech $v = \alpha_1 x_1 + \alpha_2 x_2 + \ldots + \alpha_n x_n$, gdzie x_i są kolejnymi wektorami własnymi. Mamy $A^k v = \alpha_1 \lambda_1^k x_1 + \alpha_2 \lambda_2^k x_2 + \ldots + \alpha_n \lambda_n^k x_n$. Jeśli $|\lambda_1| > |\lambda_i|$ dla $i > 1$, to $\lambda_1^k \gg \lambda_i^k$. □

Algorytm 7 (Iteracja odwrotna). Zastosowanie iteracji prostej do macierzy A^{-1} znajdzie wartość własną o najmniejszym module. Nie chcemy jednak szukać A^{-1}, więc możemy rozwiązywać wielokrotnie równanie $Av_{k+1} = v_k$ (np. za pomocą rozkładu LU).

Algorytm 8 (Iteracja odwrotna z przesunięciem). Jeśli λ jest wartością własną A, to wartością własną $A - \mu I$ jest $\lambda - \mu$. Zatem iteracja odwrotna zastosowana do $A - \mu I$ znajdzie wartość własną A najbliższą μ.

Algorytm 9 (Deflacja). Załóżmy, że macierz A jest symetryczna. Jeśli znajdziemy jej największą wartość własną λ_1 o wektorze własnym x_1, to po usunięciu x_1 z wektora v iteracja prosta da nam drugą największą wartość własną. W praktyce trzeba usuwać x_1 po każdym kroku iteracji (bo ze względu na niedokładności odejmowania będzie się pojawiał na nowo), czyli wykonywać $v = v - x_1 \langle v, x_1 \rangle$. Po wyrzuceniu s wektorów własnych rezultatem będzie wektor x_{s+1}. Ten algorytm jest mało stabilny numerycznie.

Dowód. Wektory własne macierzy symetrycznej są ortogonalne i z tego korzystamy usuwając kolejne wektory. □

Algorytm 10 (Metoda QR). Mając macierz A przyjmijmy $A_1 = A$ i zastosujmy algorytm: stosujemy rozkład QR $A_k = Q_k R_k$ i odwracamy $A_{k+1} = R_k Q_k$. Macierz A_k zbiega do macierzy diagonalnej (lub górnotrójkątnej) z wartościami własnymi A na przekątnej. Taki algorytm jest w praktyce stabilny i zbiega szybko.

Dowód. Mamy $Q^{-1}AQ = Q^{-1}QRQ = RQ$, a więc macierze powstające w kolejnych krokach mają takie same wartości własne. W ogólności nie wiemy czy ten algorytm faktycznie działa, ale w praktyce się udaje. Da się udowodnić, że działa dla diagonalizowalnej macierzy symetrycznej o różnych wartościach własnych. Definiujemy $\hat{R}_k = R_k R_{k-1} \dots R_1$ oraz $\hat{Q}_k = Q_1 Q_2 \dots Q_k$. Mamy $A^k = \hat{Q}_k \hat{R}_k$, bo

$$A^k = Q_1 R_1 Q_1 R_1 \dots Q_1 R_1 = Q_1 A_2^{k-1} R_1 = Q_1 \left(Q_2 R_2\right)^{k-1} R_1 = Q_1 Q_2 A_3^{k-2} R_2 R_1 = \dots = \hat{Q}_k \hat{R}_k.$$

Wymnożenie A^k jest właściwie zastosowaniem iteracji prostej dla pierwszej kolumny. Macierz $\hat{Q}_k$ jest właściwie zortogonalizowaną A^k, więc dąży do macierzy wektorów własnych. □

Uwaga. Aby uniknąć wielokrotnego czasochłonnego rozkładu przy wyznaczaniu wartości własnych metodą QR chcielibyśmy sprowadzić naszą macierz do trójkątnej. Nie możemy jednak tego zrobić bez zmiany wartości własnych macierzy. Zamiast tego sprowadzamy macierz do górnej macierzy Hessenberga (pod subprzekątną same zera):

$$\begin{bmatrix} x & x & \dots & x & x & x \\ x & x & \dots & x & x & x \\ 0 & x & \dots & x & x & x \\ 0 & 0 & \dots & x & x & x \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & x & x \end{bmatrix},$$

co robimy za pomocą odbić Householdera. Nie możemy po prostu mnożyć przez macierz ortogonalną, bo zmienia to wartości własne. Zamiast tego bierzemy macierz, która działa dla pierwszej kolumny bez pierwszego elementu (i całego pierwszego wiersza), domnażamy z drugiej strony przez odwrotność (transpozycję), która robi to samo dla pierwszego wiersza bez pierwszego elementu (czyli nie ruszy pierwszej kolumny). Takie przekształcenie nie zmieni wartości własnych i doprowadzi pierwszą kolumnę do postaci Hessenberga. Dalej postępujemy rekurencyjnie.

Jeśli macierz jest symetryczna, to macierz Hessenberga jest trójdiagonalna (jest też dolną macierzą

Hessenberga):

$$\begin{bmatrix} x & x & 0 & \dots & 0 & 0 & 0 \\ x & x & x & \dots & 0 & 0 & 0 \\ 0 & x & x & \dots & 0 & 0 & 0 \\ 0 & 0 & x & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & x & x & x \\ 0 & 0 & 0 & \dots & 0 & x & x \end{bmatrix}.$$

Aby przyspieszyć zbieżność algorytmu często rozkłada się macierz $A_k - \mu I = Q_k R_k$ (zmniejszamy wszystkie wartości własne o μ) i kładzie $A_{k+1} = R_k Q_k + \mu I$ (zwiększamy wartości własne o μ). Za μ można przyjąć prawy dolny element macierzy A.

6. Rozkład SVD

2024-11-07

Przykład. Mamy zbiór wektorów $a_1, a_2, \dots, a_k \in \mathbb{R}^n$. Podejrzewamy, że są prawie liniowo zależne - wszystkie są (lekko zaburzonymi) wielokrotnościami pewnego $v \in \mathbb{R}^n$. Szukamy takiego v, że $\langle v, a_i \rangle$ (długość rzutu na v) jest duże dla wszystkich a_i. Jeśli zestawimy wektory a_i w macierz A, to możemy to ująć na dwa sposoby:

- chcemy jakoś przybliżyć A przez macierz rzędu 1
- chcemy znaleźć take v, że $\|Av\|$ jest możliwie największe

Ogólniej możemy rozważać wektory będące kombinacją liniową wektorów $v_1, \dots, v_d$. Jest to przybliżenie macierzy A pewną macierzą rzędu d.

Twierdzenie 8. Szukamy maksimum $f(x) = \frac{\|Ax\|_2}{\|x\|_2}$. Okazuje się (liczymy gradient), że musi być $A^T Ax = \lambda x$, czyli rozwiązaniem jest wektor własny $A^T A$.

Definicja 16. Dla zadanej macierzy A jednostkowe wektory własne $v_1, v_2, \dots, v_n$ macierzy $A^T A$ nazywamy prawymi wektorami szczególnymi A. Tworzą one bazę ortonormalną (bo $A^T A$ jest symetryczna), więc można ich szukać znajdując wektor maksymalizujący $\|Av_1\|_2$, a później szukać kolejnych tak samo w reszcie płaszczyzny (czyli odejmując rzut na v_1).

Definicja 17. Dla zadanej macierzy A i wektorów własnych $v_1, v_2, \dots, v_n$ macierzy $A^T A$ takich, że $A^T Av_i = \lambda_i v_i$ liczby $\sqrt{\lambda_1}, \sqrt{\lambda_2}, \dots, \sqrt{\lambda_n}$ nazywamy wartościami szczególnymi A.

Twierdzenie 9. Dla prawego wektora szczególnego v_i macierzy A i odpowiadającej mu wartości szczególnej λ_i zachodzi $\|Av_i\| = \sqrt{\lambda_i}$.

Dowód.

$$\|Av_i\| = \sqrt{v_i^T A^T Av_i} = \sqrt{\lambda_i v_i^T v_i} = \sqrt{\lambda_i}\,\|v_i\| = \sqrt{\lambda_i}.$$

□

Definicja 18. Dla niezerowej wartości szczególnej $\lambda_i \neq 0$ i odpowiadającego jej prawego wektora szczególnego v_i lewym wektorem szczególnym nazywamy wektor $u_i = \frac{1}{\lambda_i} Av_i$, który jest jednostkowy.

Uwaga. W kontekście wartości szczególnych odwzorowanie liniowe A można przedstawić jako złożenie trzech odwzorowań:

- przejście do bazy kanonicznej $v_i \to e_i$, któremu odpowiada macierz V^T, której wiersze to prawe wektory szczególne v_i (jest ortonormalna, więc jest to odwrotność odwzorowania V, czyli $e_i \to v_i$)
- skalowanie $e_i \to \sqrt{\lambda_i} e_i$, któremu odpowiada macierz diagonalna z wartościami szczególnymi

na przekątnej

- powrót do bazy $e_i \to u_i$, któremu odpowiada macierz U, której kolumny to lewe wektory szczególne u_i.

Definicja 19 (Rozkład SVD)**.** Rozkładem SVD (singular value decomposition) macierzy A nazywamy rozkład na trzy macierze $A = U\Sigma V^T$, gdzie U, V są ortogonalne, a Σ diagonalna i zawiera wartości szczególne na diagonali.

Uwaga. Jeśli macierz A jest osobliwa, to część jej wartości szczególnych jest zerem i nie wszystkie jej lewe wektory szczególne są niezerowe. W rozkładzie SVD uzupełniamy je dowolnymi wektorami do bazy ortonormalnej. Podobnie, gdy $A \in \mathbb{R}^{m\times n}$ nie jest kwadratowa. Jeśli $m < n$, to A^TA ma wymiar $n \times n$, ale mniejszy rząd i część wartości szczególnych jest zerowa. Uzupełniamy U do bazy tak, by macierze U, Σ, V miały wymiary odpowiednio $m \times m, m \times n, n \times n$. Jeśli $m > n$, to A^TA na pewno ma za mały rząd i też będziemy uzupełniać U.

Najprostszym sposobem na znalezienie rozkładu SVD jest znalezienie wartości i wektorów własnych macierzy A^TA, która jest symetryczna i dodatnio określona, więc metoda QR działa dla niej bardzo dobrze. Samo obliczanie A^TA pogarsza jednak stabilność.

Algorytm 11 (Goluba-Kahana)**.** Dla kwadratowej macierzy A można usprawnić najprostszą metodę znajdowania rozkładu SVD. Robimy to, znajdując takie macierze unitarne U', V', że $B = U'AV'$ jest bidiagonalna (niezerowe wartości na przekątnej i pod nią). Można je znaleźć na przemian eliminując wiersze i kolumny macierzami Householdera. Teraz B^TB jest tridiagonalna (w postaci Hessenberga), a więc lepsza numerycznie.

Definicja 20. Dla macierzy $A = U\Sigma V^T$ o zadanym rozkładzie SVD jej pseudoodwrotnością nazywamy macierz $A^+ = V\Sigma^+U^T$, gdzie Σ^+ ma na przekątnej odwrotności elementów Σ lub 0, jeśli Σ też ma 0. Jeśli A jest kwadratowa i nieosobliwa, to $A^+ = A^{-1}$.

Twierdzenie 10. Mając zadany dowolny układ (w tym nad- lub niedookreślony) $Ax = b$ jego rozwiązanie jest równoważne zminimalizowaniu $\|x\|^2$ dla x spełniających (najlepiej jak się da) równanie $A^TAx = A^Tb$. Rozwiązaniem jest $x = A^+b$, gdzie A^+ oznacza pseudoodwrotność.

Dowód. Podstawiając rozkład SVD dostajemy $\Sigma y = d$, gdzie $y = V^Tx$ i $d = U^Tb$. Zatem chcemy zminimalizować $\|y\|^2$ przy $\Sigma y = d$, co jest spełnione przez Σ^+d. Zatem $V^Tx = \Sigma^+U^Tb$, czyli $x = A^+b$ – dla układu nadokreślonego jest to rozwiązanie w sensie najmniejszych kwadratów, a dla niedookreślonego najmniejsze w sensie normy. □

Algorytm 12. Mając zadany rozkład SVD $A = U\Sigma V^T$ możemy przybliżyć A przez macierz niższego rzędu $A^* = U\Sigma^*V^T$, gdzie Σ^* powstała z Σ poprzez zastąpienie 0 wszystkich wartości poza kilkoma największymi. Tak otrzymane przybliżenie jest najlepsze w sensie normy Frobeniusa – A^* to macierz zadanego rzędu, która minimalizuje $\|A - A^*\|_F = \sum_{i,j}\left(a_{ij} - a^*_{ij}\right)^2$.

Algorytm 13 (Principal Component Analysis)**.** Mając zadany zbiór S wektorów z $\mathbb{R}^n$ szukamy zbiorów k wektorów, który je generuje. Robimy to, ustawiając je w macierz i przybliżając ją macierzą rzędu k. Jeśli $k = 1$, to przybliżyliśmy te wektory jedną składową – principal component. W razie potrzeby szukamy też kolejnych składowych. Takie przybliżenia mają zastosowanie w analizie danych.

7. Równania nieliniowe

2024-11-14

Pomysł. Mając zadane równanie $f(x) = 0$ dla znanej funkcji f, chcemy je rozwiązać. Dla funkcji nieliniowych nie jest to łatwe. Zazwyczaj ograniczamy się do funkcji ciągłych $f : [a, b] \to \mathbb{R}$, najlepiej jeszcze różniczkowalnych.

Algorytm 14 (Bisekcja). Mając zadaną funkcję $f : [a,b] \to \mathbb{R}$ szukamy jej pierwiastka. Jeśli $f(a) \cdot f(b) < 0$, to pierwiastek istnieje i znajdzie go wyszukiwanie binarne

loop
  $s := \frac{a+b}{2}$
  if $f(a) f(s) < 0$ **then**
    $b := s$
  else
    $a := s$
  end if
end loop

które możemy kończyć, gdy wynik $f(s)$ jest odpowiednio blisko zera, przedział $[a,b]$ jest odpowiednio mały lub wykonaliśmy ustaloną liczbę iteracji. Dwie pierwsze metody mogą się zawiesić (w reprezentacji liczb może nie być takich liczb, by odpowiednie wartości zbliżyły się do siebie dostatecznie blisko). Wystarczy wykonać mniej więcej tyle iteracji, ile cyfr znaczących chcemy znać.

Dowód. W każdym kolejnym kroku szukamy wartości w przedziale $[a_n, b_n]$. Mamy $b_{n+1} - a_{n+1} \leq \frac{1}{2}(b_n - a_n)$. Zatem błąd $e_n = |a_n - x|$ spełnia $e_{n+1} \approx \frac{1}{2} e_n$. Mówimy, że metoda bisekcji ma zbieżność liniową – błąd zależy liniowo od poprzedniego, do wyznaczenia k cyfr znaczących musimy wykonać $O(k)$ kroków. □

Twierdzenie 11 (Wzór Taylora). Wielokrotnie różniczkowalną funkcję $f(x)$ można przedstawić w postaci

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

gdzie a jest pewnym punktem (w którego otoczeniu przybliżamy funkcję) oraz dla pewnego ξ jest

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

Algorytm 15 (Metoda Newtona). Szukając pierwiastka x^* równania $f(x) = 0$ mamy jego przybliżenie x_n. Otrzymujemy lepsze przybliżenie $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$, które działa, jeśli $f'(x^*) \neq 0$ oraz jesteśmy odpowiednio blisko pierwiastka.

Dowód. Ze wzoru Taylora jest $f(x^*) = f(x_n) + (x^* - x_n) f'(x_n) + R_2$, a więc

$$0 \approx f(x_n) + (x^* - x_n) f'(x_n),$$

co daje odpowiednie przybliżenie. Oczywiście metoda nie działa dla zerowej pochodnej, dla $f'(x^*) \approx 0$ dostajemy bardzo małe wartości w mianowniku, co powoduje, że algorytm przestaje zbiegać do dobrego wyniku. Do tego zakładamy $(x_n - x^*)^2 \ll |x_n - x^*|$, czyli, że jesteśmy blisko pierwiastka.

Jeśli $e_n = x_n - x^*$ jest błędem w n-tym kroku, to

$$e_{n+1} = x_{n+1} - x^* = e_n - \frac{f(x_n)}{f'(x_n)} = \frac{f'(x_n) e_n - f(x_n)}{f'(x_n)} = \frac{f''(\xi) e_n^2}{2 f'(x_n)},$$

co dla ograniczonego f'' i dalekiego od zera f' daje $e_{n+1} \leq C \cdot e_n^2$, czyli zbieżność kwadratową – do obliczenia k cyfr wystarczy $\log k$ iteracji. □

Algorytm 16 (Metoda babilońska). Szukamy pierwiastka $f(x) = x^2 - R$, mając przybliżenie x_n metoda Newtona daje nam $x_{n+1} = \frac{1}{2}\left(x_n + \frac{R}{x_n}\right)$.

Algorytm 17 (Metoda siecznych). Mając zadaną funkcję $f(x)$ nie zawsze możemy policzyć jej po-

chodną (np. gdy znamy jej wartości, a nie wzór). W takiej sytuacji stosujemy przybliżenie

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

które podstawiamy do metody Newtona. Błąd spełnia równanie $e_{n+1} \approx C \cdot e_n \cdot e_{n-1}$, a więc jest $e_{n+1} \approx e_n^{\phi}$, gdzie $\phi = \frac{1+\sqrt{5}}{2}$. Taką zbieżność nazywamy ponadliniową.

Algorytm 18 (Metoda punktu stałego)**.** Szukając rozwiązania $f(x) = x$ definiujemy kolejne przybliżenia przez $x_{n+1} = f(x_n)$. Metoda ta nie zawsze jest zbieżna. Ma zbieżność liniową, a dla $f'(x^*) = 0$ kwadratową (przez x^* oznaczamy rozwiązanie równania).

Dowód. Twierdzenie Banacha o punkcie stałym mówi, że taka metoda jest zbieżna dla odwzorowań zwężających. Na $\mathbb{R}$ są to funkcje spełniające warunek Lipschitza. Jeśli funkcja jest zwężająca ze stałą λ, to mamy

$$e_{n+1} = x_{n+1} - x^* = f(x_n) - f(x^*) \leq \lambda |x_n - x^*| = \lambda |e_n|,$$

czyli zbieżność liniową. Dla $f'(x^*) = 0$ jest

$$e_{n+1} = x_{n+1} - x^* = f(x_n) - f(x^*) = \frac{1}{2} f''(\xi)(x_n - x^*)^2 \leq C(x_n - x^*)^2.$$

□

Algorytm 19 (Metoda Dekkera)**.** Łączymy szybką zbieżność metody Newtona z pewnością metody bisekcji. Jeśli wiemy, że nasze rozwiązanie spełnia $x^* \in [p_n, q_n]$, to możemy wykonać krok metody siecznych (Newtona). Jeśli otrzymana wartość będzie w tym przedziale, to poprawiamy rozwiązanie, w przeciwnym wypadku robimy krok metody bisekcji.

Algorytm 20. Mając zadany układ równań nieliniowych $f(\overline{x}) = 0$ (gdzie f jest funkcją z $\mathbb{R}^m$ w $\mathbb{R}^n$) możemy znaleźć jego rozwiązanie $\overline{x}^*$ stosując przybliżenie

$$f(\overline{x}^*) \approx f(\overline{x}) + Df_{\overline{x}} \cdot (\overline{x}^* - \overline{x}),$$

gdzie Df_y jest jakobianem w punkcie y. Dla $m = n$ daje to wzór analogiczny do metody Newtona

$$\overline{x}_{n+1} = \overline{x}_n - (Df_{\overline{x}_n})^{-1} f(\overline{x}_n),$$

gdzie nie liczymy odwrotności, tylko rozwiązujemy odpowiedni układ równań. Możemy też przybliżać Df przez coś łatwiej obliczalnego.

Algorytm 21 (Metoda Broydena)**.** Analog metody siecznych dla układów równań. Stosujemy rekurencję

$$\overline{x}_{n+1} = \overline{x}_n - J_n^{-1} f(\overline{x}_n),$$

gdzie macierz J_n spełnia równanie

$$f(\overline{x}_n) - f(\overline{x}_{n-1}) = J_n(\overline{x}_n - \overline{x}_{n-1}).$$

Takich macierzy jest wiele, ale można szukać macierzy J_{n+1} najbliższej J_n, co daje nam wzór

$$J_{n+1} = J_n + \frac{\Delta f - J_n \Delta x}{\|\Delta x\|^2} \cdot (\Delta x)^T.$$