Metody numeryczne zadanie 2

Bartosz Kucypera

19 listopada 2023

Niech $A \in \mathbb{R}^{N \times N}$ będzie nieosobliwą macierzą trójdiagonalną.

Rozkład QR macierzy A przekształceniami Householdera

Wykorzystam zywkły algorytm znajdujący rozkład QR macierzy (o którego poprawności wiemy już z ćwiczeń) i wykorzystam specyficzną strukturę A by działał on w $O(N^2)$.

Wyzerowanie pierwszej kolumny pod diagonalą

Niech $e = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$, $\|\cdot\|$ normą euklidesową i x to zerowana kolumna.

Wyliczamy przekształcenie Householdera:

$\alpha = -\|x\| * \operatorname{sign}(x_1)$

$u = x - \alpha e$

$v = \frac{u}{\|u\|}$

$Q_1 = I - 2vv^T$

Wektor x miał co najwyżej dwie niezerowe współżędne (pierwszą i drugą), czyli macierz $2vv^T$ ma co najwyżej niezerowy kwadrat 2×2 w lewym górym rogu.
W takim razie domnożenie Q_1 do innej macierzy możemy robić liniowym kosztem.
Po domnożeniu, zmieniają się co najwyżej dwa wiersze (lub dwie kolumny w zależności z której strony domnażamy).

Macierzy Q_1 nie potrzebujemy do niczego innego niż do domnażania jej (raz do macierzy na której pracujemy by uzyskać R i raz na boku by uzyskać całe złożenie przekształceń Householdera, Q), możemy więc trzymać tylko cztery elementy macierczy $-2vv^T$ które mogą być niezerowe i przy domnażaniu odpowiednio modyfikować macierz.

Algorytm

Zerujemy pierwszą kolumnę przekształceniem Q_1.
$P = Q_1 \cdot A$

$$P = \left(\begin{array}{c|ccc} a_{11} & * & \cdots & * \\ \hline 0 & & P' & \end{array}\right)$$

P' dalej jest trójdiagonalna (zmienić mógł jej się tylko pierwszy element na diagonali), więc możemy znaleźć rekurencyjnie jej rozkład QR.

Niech $P' = Q'R'$.

Wtedy macierz

$$\left(\begin{array}{c|ccc} a_{11} & * & \cdots & * \\ \hline 0 & & R' & \end{array}\right)$$

jest szukaną macierzą R, a macierz

$$Q_1 \cdot \left(\begin{array}{c|c} 1 & 0 \\ \hline 0 & Q' \end{array}\right)^T$$

jest szukaną macierzą Q.

Macierz R możemy wyliczać w miejscu, a macierz Q możemy na początku ustawić jako identyczność i w trakcie wykonywania algorytu na bierząco domnażać do niej kolejne przekształcenia Householdera.

Wykonujemy wtedy N kroków i w każdym z nich dwa razy domnażamy macierz przekształcenia Householdera do innej macierzy w czasie $O(N)$ (dzięki jej specyficznej strukturze), co łącznie daje nam czas działania $O(N^2)$.

Kod w matlabie:

```
% QR Factorization by Householder reflections for strongly Tri Diagonal matrix
function [Q, R] = QRFHTD(A)
    N = size(A, 1);
    Q = eye(N);
    R = A;

    % zapominamy już o rekurencji, żeby łatwiej się implementowało
    % wszystkie przekształcenia wyliczamy odrazu w pełnym wymiarze i na bierząco domnażamy do Q
    % w komentarzach nomenklatura z opisu algorytmu, żeby było widać co się dzieje
    for i = 1:(N-1)
        % wyliczenie wektora v
        alf = -sqrt(R(i, i).^2 + R(i+1, i).^2)*sign(R(i,i));
        uii = R(i, i) - alf;
        uji = R(i+1, i);
        nrm_u = sqrt(uii.^2 + uji.^2);
        uii = uii/nrm_u; % pierwsza współżędna wektora v
        uji = uji/nrm_u; % druga współżędna wektora v

        % zmienia nam się tylko sześć komórek w tym jedna z macierzy P' (P' z opisu algorytmu)
        for j = i:min(N, i+2)
            % zmiana komórek wierszy
            ch1 = -2 * (R(i, j)*uii*uii + R(i+1, j)*uii*uji);
            ch2 = -2 * (R(i, j)*uji*uii + R(i+1, j)*uji*uji);
            R(i, j) += ch1;
            R(i+1, j) += ch2;
        end

        % to samo tylko tym razem domnażamy przekształcenie Householdera z drugiej strony macierzy
        % czyli zmieniają nam się kolumny
        % tym razem może zmieniać się więcej komórek ale nie więcej niż O(N)
        for k = 1:N
            ch1 = -2 * (Q(k, i)*uii*uii + Q(k, i+1)*uji*uii);
            ch2 = -2 * (Q(k, i)*uii*uji + Q(k, i+1)*uji*uji);
            Q(k, i) += ch1;
            Q(k, i+1) += ch2;
        end
    end
end
```