QR Faktoryzacja metodą obrotów Givensa

Filip Dziurdzia, Jakub Płowiec

1. Wstęp teoretyczny

Faktoryzacja QR polega na przedstawieniu macierzy $A \in \mathbb{R}^{m \times n}$ jako iloczynu ortogonalnej macierzy Q oraz górnotrójkątnej macierzy R, tzn.

$$A = QR.$$

Jednym ze sposobów realizacji faktoryzacji QR są **obroty Givensa**, które eliminują elementy pod diagonalą za pomocą macierzy obrotów 2D.

2. Obroty Givensa – wyznaczanie współczynników

Aby wyeliminować element $a_{i,j}$ z macierzy A, tworzymy macierz obrotu G, która działa tylko na wiersze $i-1$ i i, tak aby po przekształceniu:

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} a_{i-1,j} \\ a_{i,j} \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}$$

Współczynniki obrotu c (cosinus) i s (sinus) oblicza się ze wzorów:

$$r = \sqrt{a^2 + b^2}, \quad c = \frac{a}{r}, \quad s = -\frac{b}{r}$$

gdzie:

- $a = a_{i-1,j}$,
- $b = a_{i,j}$,
- r to długość wektora $[a, b]^T$.

Dzięki temu, wiersz i zostaje wyzerowany w kolumnie j.

3. Pseudokod algorytmu

```
Dla każdej kolumny j od 0 do n-1:
    Dla każdego wiersza i od m-1 do j+1 (w dół):
        Jeśli A[i, j] ≠ 0:
            a = A[i-1, j]
            b = A[i, j]
            r = sqrt(a^2 + b^2)
            c = a / r
            s = -b / r

            Skonstruuj macierz Givensa G:
                G[i-1, i-1] = c
                G[i, i]     = c
                G[i-1, i]   = s
                G[i, i-1]   = -s

            Zaktualizuj:
                R = G · R
                Q = Q · Gᵀ
```

4. Implementacja w Python

Program składa się z trzech głównych funkcji:

1. **`qr_givens(A)`** – główna funkcja implementująca faktoryzację QR za pomocą obrotów Givensa. Tworzy ortogonalną macierz Q oraz górnotrójkątną R.

```python
def qr_givens(A):
    (m, n) = A.shape
    Q = np.eye(m)
    R = A.copy()

    for j in range(n):
        for i in range(m - 1, j, -1):
            if R[i, j] != 0:
                c, s = givens_rotation(R[i - 1, j], R[i, j])
                G = np.eye(m)
                G[[i - 1, i], [i - 1, i]] = c
                G[i, i - 1] = s
                G[i - 1, i] = -s

                R = G @ R
                Q = Q @ G.T
    return Q, R
```

2. **`givens_rotation(a, b)`** – funkcja pomocnicza wykorzystywana w `qr_givens(A)`, która na podstawie elementów a i b wyznacza współczynniki cosinusa (`c`) i sinusa (`s`) dla obrotu Givensa. Dzięki temu możliwe jest wyeliminowanie składnika b w sposób numerycznie stabilny.

```python
def givens_rotation(a, b):
    r = np.hypot(a, b)
    if r == 0:
        c = 1
        s = 0
    else:
        c = a / r
        s = -b / r
    return c, s
```

3. **test_qr_givens()** – funkcja testująca, która:
    - generuje przykładową macierz A o wymiarach 32×32,
    - wykonuje dekompozycję QR przy pomocy ręcznie napisanego algorytmu Givensa,
    - weryfikuje poprawność rozkładu poprzez:
        - normę błędu rekonstrukcji $\|A - QR\|$,
        - test ortogonalności: $Q^T Q \approx I$,
        - porównanie wyniku z biblioteką `numpy.linalg.qr` przy użyciu `np.allclose` – funkcja ta sprawdza, czy dwie macierze są numerycznie zbliżone (element po elemencie, z uwzględnieniem błędów zaokrągleń).

```python
def test_qr_givens():
    np.random.seed(42)
    A = np.random.randn(32, 32)
    Q, R = qr_givens(A)

    reconstruction_error = np.linalg.norm(A - Q @ R)

    orthogonality_error = np.linalg.norm(Q.T @ Q - np.eye(32))

    Q_np, R_np = np.linalg.qr(A)
    equivalent_to_numpy = np.allclose(Q @ R, Q_np @ R_np)

    print("Błąd rekonstrukcji ||A - QR|| =", reconstruction_error)
    print("Błąd ortogonalności ||QᵀQ - I|| =", orthogonality_error)
    print("Zgodność z NumPy QR? ", equivalent_to_numpy)

    return reconstruction_error, orthogonality_error,   equivalent_to_numpy
```

5. Wyniki

Błąd rekonstrukcji $||A - QR|| = 2.4663525290012486e - 14$

Błąd ortogonalności $||Q^T Q - I|| = 4.929963396710446e - 15$

Zgodność z gotową funkcją w NumPy QR? $True$

Jak widać uzyskane wyniki są bardzo satysfakcjonujące i pokazują poprawną implementację faktoryzację QR metodą obrotów Givensa. K