Projekt 2 - Eliminacja Gaussa i LU faktoryzacja

Cyprian Neugebauer

1 Eliminacja Gaussa

1.1 Pseudokod

Algorytm eliminacji Gaussa można opisać następującymi krokami:

1. Weź macierz A i wektor b, gdzie A jest macierzą współczynników układu równań, a b jest wektorem wyrazów wolnych.
2. Dołącz wektor b do macierzy A jako dodatkową kolumnę, tworząc rozszerzoną macierz Ab.
3. Dla każdego wiersza i od 1 do n (gdzie n jest liczbą wierszy macierzy A):
   (a) Normalizuj wiersz i przez podzielenie go przez element diagonalny $Ab_{i,i}$, aby na przekątnej uzyskać 1.
   (b) Dla każdego wiersza j poniżej i (od $i+1$ do n):
      i. Odejmij od wiersza j wiersz i pomnożony przez $Ab_{j,i}$, aby wyzerować elementy poniżej przekątnej w kolumnie i.
4. Inicjalizuj wektor rozwiązań x o długości n.
5. Rozwiąż układ równań od ostatniego wiersza do pierwszego:
   (a) Dla każdego wiersza i od n do 1, oblicz $x_i = Ab_{i,n+1} - \sum_{k=i+1}^{n} Ab_{i,k} \cdot x_k$.
6. Zwróć wektor rozwiązań x.

Algorithm 1 Gaussian Elimination

procedure GaussianElimination(A, b)
    $n \leftarrow \text{rows}(A)$
    $Ab \leftarrow \text{concatenate}(A, b, \text{axis} = 1)$

    for $i \leftarrow 0$ **to** $n - 1$ **do**
        $Ab[i, i:] \leftarrow Ab[i, i:]/Ab[i, i]$
        for $j \leftarrow i + 1$ **to** $n - 1$ **do**
            $Ab[j, i:] \leftarrow Ab[j, i:] - Ab[j, i] \cdot Ab[i, i:]$
        end for
    end for

    $x \leftarrow \text{zeroes}(n)$
    for $i \leftarrow n - 1$ **down to** 0 **do**
        $x[i] \leftarrow Ab[i, n] - Ab[i, i + 1 : n] \cdot x[i + 1 :]$
    end for

    return x
end procedure

1.2 Kod algorytmu

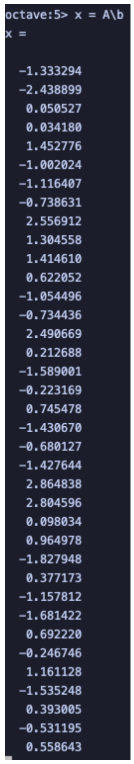
```python
def gaussian_elimination(A: np.ndarray, b: np.ndarray) -> np.ndarray:
    n = A.shape[0]

    Ab = np.column_stack((A, b))
    for i in range(n):
        Ab[i, i:] /= Ab[i, i]
        for j in range(i + 1, n):
            Ab[j, i:] -= Ab[j, i] * Ab[i, i:]

    x = np.empty((n,))
    for i in range(n - 1, -1, -1):
        x[i] = Ab[i, n] - np.dot(Ab[i, i + 1 : n], x[i + 1 :])

    return x
```

Rysunek 1: Kod eliminacji Gaussa

1.3 Porównanie wyników

```
Matrix A:
[ 2. 19. 16. 10. 10. 21.  2. 17.  5.  2. 13. 24. 18. 19. 17. 19. 12.  3. 20. 11. 12.  9.  4. 23. 19. 16. 10. 20. 13. 11. 11.  5.  2. 13. 22.  1. 21.]
[20.  6. 15.  4. 18. 17.  8.  1. 24. 11. 22. 16. 19. 18.  4.  9. 11. 12.  1. 13.  3. 18. 17. 23. 18.  9. 24. 10.  8. 22.  9.  1. 11. 19.  4. 11.  3.]
[17. 11.  8.  5. 14. 16. 23. 10.  4. 20. 15. 17.  2.  7. 19. 20. 10. 20. 21.  9. 22.  7.  5. 17. 15.  3. 20.  4. 20.  0. 19. 19. 19. 16. 11. 17.  6.]
[19. 13. 11. 12. 14.  0.  3.  6.  2. 10. 16. 16. 11. 21. 14.  1. 19. 14. 15. 14. 13.  2. 13. 19.  7. 15.  0.  8. 10. 24.  5.  6. 10. 24. 21.  0.  5.]
[20.  1. 21.  7. 22.  7. 10. 16.  3. 13. 12. 19. 24. 16. 10. 10. 10. 20.  8.  4.  8.  0.  2.  2. 19. 18. 17. 11. 17.  4. 22. 12. 23.  3. 12. 17. 12.]
[11.  4.  9.  5.  7. 17. 15. 15.  9. 23.  2.  8.  2.  8. 24.  9. 22. 12. 17. 11.  6. 19. 24.  6. 19.  6. 17. 19. 11. 18.  6.  1.  2. 11. 22.  3. 11.]
[17.  5. 18.  7. 20. 14. 13.  4. 11. 21.  0. 18. 12. 17. 16. 10.  7. 15.  3. 14.  2. 16. 15.  2. 18. 10. 19.  1.  4. 12.  4.  8. 16.  3. 16.  2.  4.]
[14. 19.  4.  7. 23. 24. 14. 11.  8. 15. 14.  7.  0.  3. 23. 10. 12. 11. 19.  1.  2.  6. 12. 16. 12. 11. 23.  4. 14. 13. 11. 20.  6. 23.  8. 13. 13.]
[21. 10.  7.  0.  9. 20.  0. 22. 13.  3.  7. 13.  7.  2. 16. 16.  2.  7.  6. 16. 18. 18. 23. 19.  8.  2. 15. 22. 24.  5.  5.  0.  3. 13. 13.  9.  7.]
[20. 11. 20. 24.  7.  5. 23. 15.  7. 15. 12. 22.  6. 24. 23.  4.  4. 13.  1.  7. 10. 24. 24.  3. 22.  2. 18. 21. 22.  5. 22.  2. 12. 11.  7.  0. 19.]
[ 8. 16. 10.  9. 22.  2. 18. 18.  9.  6.  0. 23. 23.  6.  0.  3.  1. 20. 22.  3. 18.  4.  9. 14. 17. 21.  1.  4.  7.  7. 24. 19.  8. 24. 12. 12. 10.]
[ 3. 23.  0.  4.  5.  6.  3.  0. 16. 24.  3.  7. 12. 23. 17. 19. 14. 10.  4. 21. 20.  0. 17.  2. 18. 20.  3. 10.  3. 14. 23.  2.  9. 24.  7.  3. 12.]
[12. 16.  7. 23. 13.  7. 24. 23. 12.  0. 14. 13.  3. 15. 21.  2. 18.  3. 18. 10.  0. 24.  4. 14.  4. 23. 18. 20. 10. 11. 15. 19. 21.  0. 16.  2.  7.]
[20.  6. 19. 11.  5.  8. 13. 19. 15. 23. 21. 18. 15.  7. 10.  2.  9. 23. 10.  2. 16. 24. 21.  6. 11. 13.  6.  6.  5. 24. 18. 21. 20. 23.  2. 22.  1.]
[11. 14.  8.  3.  4. 20.  7.  7. 23.  3. 22. 23. 14.  4.  3.  7. 14.  3. 11.  2. 13.  0. 17.  1.  4.  4. 20.  1. 10. 14. 12. 17. 20.  9. 20.  7. 24.]
[12. 18. 21. 19. 21. 13.  1. 12.  4. 22.  5.  5.  6. 16. 14. 23. 10.  7.  1.  0.  9. 23. 13. 22.  2.  2. 20.  2.  1. 10. 23. 17.  2. 20. 21. 20. 22.]
[ 5. 24.  8. 20. 14. 19. 15. 16. 15. 19. 18.  3. 20. 13.  8. 12. 24. 21. 20. 11. 19.  9. 17. 15.  6.  6.  1.  3. 10. 11.  4. 10. 15.  5.  8.  9. 12.]
[ 9. 18.  8. 10.  9.  4. 17.  5.  7. 19. 23. 16. 22. 21. 12. 19.  8. 11. 13. 24. 21. 12. 16.  5. 20. 13. 13. 19. 15.  1.  7.  4. 18.  0.  5. 14. 17.]
[12. 21. 22.  3. 23. 15. 15.  2. 15. 18. 14.  2. 13. 23. 18.  3. 23. 23.  4. 20. 10. 14. 11. 19. 24. 19. 23. 23.  2.  6. 18. 14.  1.  2. 13. 15.  4.]
[ 4.  3. 14.  0. 14. 11. 11. 16. 13. 18. 19. 23. 19. 12. 12.  5. 14.  3. 23. 21.  2. 19.  2. 18.  2. 12. 16. 18. 10. 23. 19.  5. 16. 18.  8. 18. 22.]
[20. 19. 15.  6. 19.  9. 13.  7.  7.  3. 24.  3.  2. 23. 13. 10. 18.  9.  3. 18.  4. 13. 23. 23.  9. 19. 15. 11.  3.  9. 16. 24.  0. 17. 21. 23. 10.]
[ 2.  6. 21. 21. 15.  3.  3. 11. 14.  0. 17. 13.  0.  2. 11. 18. 20. 10.  7. 24. 11.  7. 11.  0.  7. 12. 22. 18. 19. 13.  2.  9. 24.  1. 21. 17.  7.]
[19. 20.  1.  2. 16. 24.  0. 16.  3. 16. 15.  2. 17. 22. 21.  0.  6.  6. 11.  3. 21. 19. 20.  4.  0. 22.  7. 16. 22.  0.  0.  0.  3.  1. 20. 15. 24.]
[20. 24.  5.  7. 21.  6.  1. 16. 20. 24. 23.  3. 19. 21. 17. 20. 20. 11.  1. 11.  5.  2.  3. 17. 12. 13. 18. 13. 15.  2. 21. 11.  3. 20. 18.  1.  4.]
[ 0.  6.  2. 17. 13. 24. 18. 15. 12.  1.  1. 15.  0.  1.  5. 22. 11. 17. 10.  4.  9.  2. 19.  4.  9. 24. 14. 11. 13. 19.  6. 15. 21. 14. 11.  3.  6.]
[23. 13.  7. 10. 14. 13. 17.  6. 16. 11. 23.  3.  3. 11. 12. 17. 10. 22. 11.  3.  2. 24.  3. 23.  6.  2.  7. 15. 10. 19. 15. 10. 15. 24. 10.  3. 10.]
[21.  5. 23. 14. 15.  7. 23.  0.  7. 10. 11. 11.  3.  5.  2. 14.  8. 14. 14. 17. 13. 16. 14. 16. 15.  7.  6. 19.  8. 13.  4. 10. 13. 15. 13.  9. 24.]
[12. 20. 18. 13. 22.  3. 23. 21. 12. 18. 13.  6. 19.  6.  7. 12. 20.  9. 12.  1.  2.  1.  1. 21. 21. 20.  1.  1.  7. 14.  8. 12.  4. 18.  7.  1. 18.]
[22.  0. 10. 20. 17. 21. 10.  9. 17.  3.  5. 15. 16.  2. 12.  9. 12. 23.  1. 24. 11. 19. 10.  7. 19. 17.  3. 17. 18.  9. 23. 16.  8.  0. 16.  5. 18.]
[13. 11.  4. 17.  4. 10. 20. 23. 24. 23. 13.  5. 20. 20. 24. 16.  3. 12. 11. 21.  9.  8.  2. 20. 18.  1. 10. 11. 11. 22.  3.  7.  4.  6. 22. 23.  1.]
[15.  5.  4.  7.  4. 12. 24. 14. 18. 12.  0.  2.  6.  6. 21. 21. 21. 15. 13. 16. 10. 16. 19. 19. 15.  1. 18.  9. 18. 13. 22.  8. 23. 21.  8. 12. 13.]
[19. 17. 21.  9. 12. 10. 22.  7. 14.  3. 21.  9.  8. 14. 12. 21. 13. 12.  2.  0.  9. 24. 22. 15. 15.  9.  4.  0.  8.  7.  4.  6.  0. 15. 23. 17. 11.]
[10.  7.  3. 14. 14.  0. 10.  6.  4. 17. 13. 15. 17. 24. 20. 15. 21. 11. 24. 19. 16. 22. 12. 18.  7. 24. 15. 19. 11. 21.  1.  2. 12. 18. 20. 11. 16.]
[13. 24. 16. 10. 24.  2. 24.  3.  7. 20. 18. 24. 18. 15.  8.  0.  3.  7.  1. 12. 19. 24. 12. 21. 24. 21. 11.  8. 24.  6. 10.  1. 19. 16.  2.  5. 13.]
[13. 20.  6. 23. 11. 20.  5.  0.  1.  9. 23.  1.  9.  2. 23. 16. 19. 17. 16. 19.  0. 21. 18. 18.  9. 20.  9.  1. 22.  5.  2. 15.  5. 21. 11.  0. 23.]
[12.  6. 16. 13.  0. 18. 10. 17. 22. 15. 16.  6.  5. 10. 21. 13.  8. 19. 21. 22.  7. 17. 14. 11.  9.  2.  6. 13. 22. 21.  4. 16.  2. 16.  8. 11. 17.]
[10. 20. 13. 24. 15.  7. 19. 10. 13.  3.  2. 14.  4. 24. 21. 17. 18. 14.  1.  9. 12. 22.  5. 12.  8.  3.  2.  9.  2.  1.  8.  5. 13.  0.  7. 11. 24.]
Vector b:
 [ 9. 13.  5.  6.  6. 13. 14. 14. 11.  7.  8. 14.  4.  5. 13.  5.  3.  8.  4. 13. 13.  4.  7. 14. 11.  3.  8.  9.  9.  0.  4. 12.  3.  4.  4.  2.  2.]
```

Rysunek 2: Wylosowana macierz gęsta A_{37x37} i wektor **b**



Rysunek 3: Rozwiązanie x obliczone w programie MATLAB

Rysunek 4: Wartość normy L_2 dla różnicy rozwiązania w MATLAB z własnym rozwiązaniem eliminacją Gaussa bez pivotingu

Wartość normy wynosząca 1.8×10^{-6} jest bardzo bliska zeru, co oznacza, że rozwiązania są identyczne, a norma nie jest równa 0 przez skończoną precyzję zapisu liczb zmiennoprzecinkowych.

2 Eliminacja Gaussa z pivotingiem

2.1 Pseudokod

Algorytm eliminacji Gaussa z częściowym wyborem pivota można opisać następującymi krokami:

1. Weź macierz A i wektor b, gdzie A jest macierzą współczynników układu równań, a b jest wektorem wyrazów wolnych.
2. Dołącz wektor b do macierzy A jako dodatkową kolumnę, tworząc rozszerzoną macierz Ab.
3. Dla każdego wiersza i od 1 do n (gdzie n jest liczbą wierszy macierzy A):
   (a) Znajdź wiersz p poniżej wiersza i (włącznie z i), który ma największą wartość bezwzględną elementu w kolumnie i. Wiersz ten staje się nowym wierszem pivota.
   (b) Zamień wiersz i z wierszem pivota p.

(c) Normalizuj wiersz i przez podzielenie go przez element diagonalny $Ab_{i,i}$, aby na przekątnej uzyskać 1.

(d) Dla każdego wiersza j poniżej i (od $i+1$ do n):

i. Odejmij od wiersza j wiersz i pomnożony przez $Ab_{j,i}$, aby wyzerować elementy poniżej przekątnej w kolumnie i.

4. Inicjalizuj wektor rozwiązań x o długości n.

5. Rozwiąż układ równań od ostatniego wiersza do pierwszego:

(a) Dla każdego wiersza i od n do 1, oblicz $x_i = Ab_{i,n+1} - \sum_{k=i+1}^{n} Ab_{i,k} \cdot x_k$.

6. Zwróć wektor rozwiązań x.

Algorithm 2 Pivot Gaussian Elimination

```
procedure PivotGaussianElimination(A, b)
    n ← rows(A)
    Ab ← concatenate(A, b, axis = 1)

    for i ← 0 to n − 1 do
        pivot ← i + argmax(abs(Ab[i :, i]))
        Ab[i, i :] ← Ab[i, i :]/Ab[i, i]
        for j ← i + 1 to n − 1 do
            Ab[j, i :] ← Ab[j, i :] − Ab[j, i] · Ab[i, i :]
        end for
    end for

    x ← empty vector of length n
    for i ← n − 1 down to 0 do
        x[i] ← Ab[i, n] − Ab[i, i + 1 : n] · x[i + 1 :]
    end for

    return x
end procedure
```

2.2 Kod algorytmu

```
def pivot_gaussian_elimination(A: np.ndarray, b: np.ndarray) -> np.ndarray:
    n = A.shape[0]

    Ab = np.column_stack((A, b))
    for i in range(n):
        pivot = i + np.argmax(np.abs(Ab[i:, i]))
        Ab[[i, pivot]] = Ab[[pivot, i]]

        Ab[i, i:] /= Ab[i, i]
        for j in range(i + 1, n):
            Ab[j, i:] -= Ab[j, i] * Ab[i, i:]

    x = np.empty((n,))
    for i in range(n - 1, -1, -1):
        x[i] = Ab[i, n] - np.dot(Ab[i, i + 1 : n], x[i + 1 :])

    return x
```

Rysunek 5: Kod eliminacji Gaussa z pivotingiem

2.3 Porównanie wyników

```
2-norm: 1.8170138710430036e-06
```

Rysunek 6: Wartość normy L_2 dla różnicy rozwiązania w MATLAB z własnym rozwiązaniem eliminacją Gaussa z pivotingiem

Dla eliminacji Gaussa z pivotingiem występuje taka sama sytuacja jak bez pivotingu. Otrzymane rozwiązania są identyczne z MATLAB.

3 LU faktoryzacja

3.1 Pseudokod

Algorytm dekompozycji LU dla macierzy A można opisać następującymi krokami:

1. Weź kwadratową macierz A o wymiarach $n \times n$.
2. Inicjalizuj macierz L jako macierz jednostkową rozmiaru $n \times n$ i macierz U jako kopię macierzy A.

3. Dla każdego wiersza i od 1 do n wykonaj:
   (a) Dla każdego wiersza j od $i+1$ do n wykonaj:
      i. Oblicz współczynnik $L_{j,i} = U_{j,i}/U_{i,i}$ i zapisz go w macierzy L na pozycji (j,i).
      ii. Zaktualizuj wiersz j w macierzy U poprzez odjęcie $L_{j,i}$ razy wiersz i w macierzy U, zaczynając od kolumny i. Czyli wykonaj operację $U_{j,i:} -= L_{j,i} \cdot U_{i,i:}$.
4. Zwróć macierz L i macierz U jako wynik dekompozycji LU.

Algorithm 3 LU Decomposition

procedure LUDecomposition(A)
    $n \leftarrow \text{rows}(A)$
    $L \leftarrow \text{eye}(n)$
    $U \leftarrow A.\text{copy}()$

    for $i \leftarrow 0$ **to** $n-1$ **do**
        for $j \leftarrow i+1$ **to** $n-1$ **do**
            $L[j,i] \leftarrow U[j,i]/U[i,i]$
            $U[j,i:] \leftarrow U[j,i:] - L[j,i] \cdot U[i,i:]$
        end for
    end for

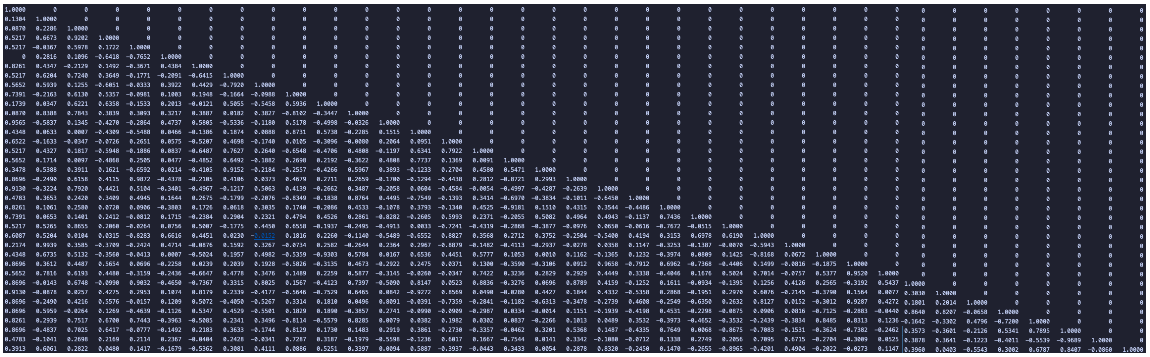
    return L, U
end procedure

3.2 Kod algorytmu

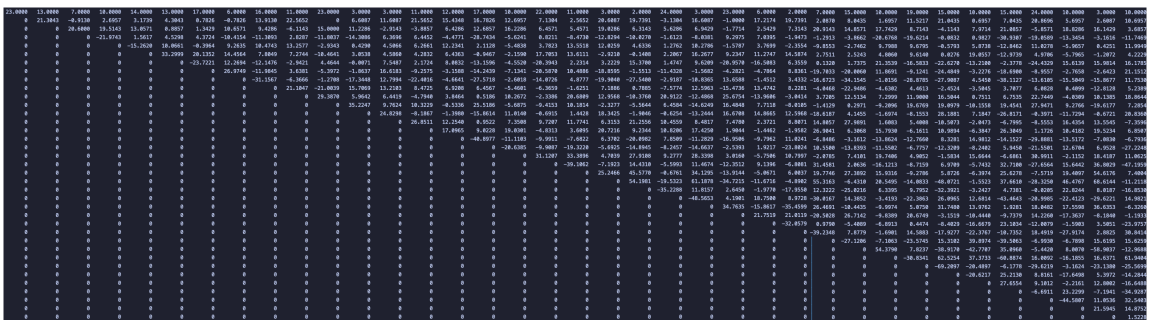
```python
def lu_decomposition(A: np.ndarray) -> tuple[np.ndarray, np.ndarray]:
    n = A.shape[0]

    L, U = np.eye(n), A.copy()
    for i in range(n):
        for j in range(i + 1, n):
            L[j, i] = U[j, i] / U[i, i]
            U[j, i:] -= L[j, i] * U[i, i:]

    return L, U
```

Rysunek 7: Kod faktoryzacji LU

3.3 Porównanie wyników



Rysunek 8: Macierz L faktoryzacji LU z MATLAB



Rysunek 9: Macierz U faktoryzacji LU z MATLAB

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Rysunek 10: Macierz permutacji P faktoryzacji LU z MATLAB

```
[[ 1.      0.      0.     ...  0.      0.      0.    ]
 [10.      1.      0.     ...  0.      0.      0.    ]
 [ 8.5     0.8179  1.     ...  0.      0.      0.    ]
 ...
 [ 6.5     0.5625  1.7488 ...  1.      0.      0.    ]
 [ 6.      0.587  -0.5435 ...  4.1187  1.      0.    ]
 [ 5.      0.4076  0.8401 ... -1.771  -0.4415  1.    ]]
```

Rysunek 11: Macierz L faktoryzacji LU z własnego rozwiązania

```
[[   2.        19.        16.     ...   22.         1.        21.    ]
 [   0.      -184.      -145.     ... -216.         1.      -207.    ]
 [   0.         0.        -9.3995 ...    0.6739     7.6821    -3.1875]
 ...
 [   0.         0.         0.     ...  -35.1402    44.97      31.449 ]
 [   0.         0.         0.     ...    0.      -180.8991  -60.8471]
 [   0.         0.        -0.     ...    0.         0.        10.1363]]
```

Rysunek 12: Macierz U faktoryzacji LU z własnego rozwiązania

Macierze L i U zwrócone z MATLAB różnią się od L i U zwróconych z własnego rozwiązania. Wynika to z faktu, że MATLAB używa częściowego wyboru pivota, co wpływa na wynik dekompozycji LU. Własne rozwiązanie nie używa pivotingu, więc wyniki mogą się różnić.

4 LU faktoryzacja z pivotingiem

4.1 Pseudokod

Algorytm dekompozycji LU z częściowym wyborem pivota dla macierzy A można opisać następującymi krokami:

1. Weź kwadratową macierz A o wymiarach $n \times n$.
2. Inicjalizuj wektor permutacji $perm$ jako ciąg liczb od 0 do $n-1$, macierz L jako macierz jednostkową rozmiaru $n \times n$ i macierz U jako kopię macierzy A.
3. Dla każdego wiersza i od 1 do n wykonaj:
   (a) Znajdź indeks $pivot$ największego (w wartości bezwzględnej) elementu w kolumnie i wierszy od i do n.
   (b) Zamień wiersze i i $pivot$ w macierzy L, w kolumnach od 1 do $i-1$.
   (c) Zamień wiersze i i $pivot$ w macierzy U.
   (d) Zamień elementy i i $pivot$ w wektorze permutacji $perm$.
   (e) Dla każdego wiersza j od $i+1$ do n wykonaj:
      i. Oblicz współczynnik $L_{j,i} = U_{j,i}/U_{i,i}$ i zapisz go w macierzy L na pozycji (j, i).
      ii. Zaktualizuj wiersz j w macierzy U poprzez odjęcie $L_{j,i}$ razy wiersz i w macierzy U, zaczynając od kolumny i. Czyli wykonaj operację $U_{j,i:} -= L_{j,i} \cdot U_{i,i:}$.
4. Inicjalizuj wektor odwrotnej permutacji inv_perm i dla każdego indeksu w $perm$ zapisz odwrotną permutację.
5. Zastosuj odwrotną permutację do wierszy macierzy L.
6. Zwróć macierz L i macierz U jako wynik dekompozycji LU.

Algorithm 4 Pivot LU Decomposition

```
procedure PivotLUDecomposition(A)
    n ← rows(A)
    perm ← range(n)
    L ← eye(n)
    U ← A.copy()

    for i ← 0 to n − 1 do
        pivot ← i + argmax(abs(U[i :, i]))
        Swap L[i, : i] and L[pivot, : i]
        Swap U[i, :] and U[pivot, :]
        Swap perm[i] and perm[pivot]

        for j ← i + 1 to n − 1 do
            L[j, i] ← U[j, i]/U[i, i]
            U[j, i :] ← U[j, i :] − L[j, i] · U[i, i :]
        end for
    end for

    inv_perm ← empty(n, dtype = int)
    inv_perm[perm] ← range(n)

    return L[inv_perm], U
end procedure
```

4.2 Kod algorytmu

```python
def lu_decomposition(A: np.ndarray) -> tuple[np.ndarray, np.ndarray]:
    n = A.shape[0]

    L, U = np.eye(n), A.copy()
    for i in range(n):
        for j in range(i + 1, n):
            L[j, i] = U[j, i] / U[i, i]
            U[j, i:] -= L[j, i] * U[i, i:]

    return L, U
```

Rysunek 13: Kod faktoryzacji LU z pivotingiem

4.3 Porównanie wyników

```
[[ 0.087   0.8388  0.7843 ...  0.      0.      0.    ]
 [ 0.8696 -0.249   0.4216 ...  0.      0.      0.    ]
 [ 0.7391  0.0653  0.1401 ...  0.      0.      0.    ]
 ...
 [ 0.5652  0.5939  0.1255 ...  0.      0.      0.    ]
 [ 0.5217 -0.0367  0.5978 ...  0.      0.      0.    ]
 [ 0.4348  0.6735  0.5132 ...  0.      0.      0.    ]]
```

Rysunek 14: Macierz L faktoryzacji LU z własnego rozwiązania

```
[[ 23.       13.        7.     ...  10.        3.       10.     ]
 [  0.       21.3043  -0.913  ...   5.6957   2.6087  10.6957]
 [  0.        0.      20.6    ...  18.8286  16.1429   3.6857]
 ...
 [  0.        0.       0.     ... -44.5807  11.0536  32.5403]
 [  0.        0.       0.     ...   0.      21.5945  14.8752]
 [  0.        0.       0.     ...   0.       0.       1.5228]]
```

Rysunek 15: Macierz U faktoryzacji LU z własnego rozwiązania

Macierz L zwrócona z własnego programu różni się od macierzy L zwróconej z MATLAB, natomiast jest to spowodowane tym, że w moim programie stosuję od razu odwrotną permutację, podczas gdy MATLAB zwraca macierz L z permutacją. Macierze U zwrócone z obu programów są identyczne.