Wydział WFiIS	Imię i nazwisko 1. Ihnatsi Yermakovich		Rok II	Grupa 03	
METODY **NUMERYCZNE**	Temat Rozkład LU, rozwiązanie układu równań liniowych				Nr ćwiczenia 01
Data wykonania 11.03.2022	Data oddania 19.03.2022	Zwrot do popr.	Data oddania	Data zaliczenia	OCENA

Rozwiązanie układu równań liniowych

Ćwiczenie nr 01

Ihnatsi Yermakovich

1 Wprowadzenie

Duża część problemów z różnych dziedzin nauki w pewnym momencie sprowadza się do rozwiązania układów równań liniowych, w związku z czym jest to problem kluczowy, więc musieli powstać wiele efektywnych algorytmów, które potrafiłyby go rozwiązać. Problem rozwiązywania układów równań należy do klasy P, a nie NP, co powoduje, że jeżeli pierwotny problem uda się sprowadzić do rozwiązania układu równań liniowych, nawet jeżeli na obecnym etapie rozwoju komputerów nie posiadamy takiej mocy obliczeniowej, to w przewidywalnej przyszłości potrafimy dostać wyniki.

1.1 Rozkład LU

Rozkład LU jest stosowany jako etap w rozwiązaniu wielu problemów, m.i. rozwiązywanie układów równań liniowych, znajdowanie wyznacznika macierzy, wyznaczanie macierzy odwrotnej. Rozkład LU polega na przestawieniu macierzy A:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & ... & a_{1n} \\ a_{21} & a_{22} & a_{23} & ... & a_{2n} \\ a_{31} & a_{32} & a_{33} & ... & a_{3n} \\ ... & ... & ... & ... & ... \\ a_{n1} & a_{n2} & a_{n3} & ... & a_{nn} \end{bmatrix} \tag{1}$$

jako macierzy trójkątnej dolnej L z jednościami na przekątnej i macierzy tkójkątnej górnej U:

$$A = \begin{bmatrix} 1 & 0 & 0 & ... & 0 \\ l_{21} & 1 & 0 & ... & 0 \\ l_{31} & l_{32} & 1 & ... & 0 \\ ... & ... & ... & ... & 0 \\ l_{n1} & l_{n2} & l_{n3} & ... & 1 \end{bmatrix} \qquad U = \begin{bmatrix} u_{11} & u_{12} & u_{13} & ... & u_{1n} \\ 0 & u_{22} & u_{23} & ... & u_{2n} \\ 0 & 0 & u_{33} & ... & u_{3n} \\ ... & ... & ... & ... & ... \\ 0 & 0 & 0 & ... & u_{nn} \end{bmatrix} \tag{2}$$

Przeprowadzenie rozkładu LU (eliminacja Gaussa) to nakład rzędu n^3.

1.2 Rozwiązanie układu równań liniowych

Sformułujemy problem jako:

$$A\mathbf{x} = \mathbf{b} \tag{3}$$

gdzie:

A - macierz współczynników.

$\mathbf{x}$ - wektor rozwiązań.

$\mathbf{b}$ - wektor wyrazów wolnych.

Wówczas dysponując macierzami L i U można rozwiązać układ równań:

$$\underbrace{LU}_{A}\mathbf{x} = \mathbf{b} \tag{4}$$

poprzez rozwiązanie 2 układów równań:

$$\begin{cases} L\mathbf{y} = \mathbf{b} \\ U\mathbf{x} = \mathbf{y} \end{cases} \tag{5}$$

2 Implementacja

Wszystkie przykłady kodu są napisane w języku C z wykorzystaniem biblioteki **gsl**.

2.1 Wczytanie danych z pliku

Wczytamy macierze A i b z pliku następująco:

```c
int n = 6;

gsl_matrix *A = read_matrix_from_file("{path_to_project_assets}/A_matrix.txt", n, n);
gsl_matrix *b_matrix = read_matrix_from_file("{path_to_project_assets}/b_matrix.txt", n, 1);
gsl_vector *b = gsl_vector_alloc_col_from_matrix(b_matrix, 0);
```

Listing 1: Wczytanie macierzy A i b z plików na dysku

2.2 Rozkład LU

```c
int n = 6;
int sig = 1;

gsl_permutation *p = gsl_permutation_calloc(n);
gsl_vector *lib_solution = gsl_vector_calloc(n);

gsl_linalg_LU_decomp(A, p, &sig);
```

Listing 2: Przeprowadzenie rozkładu LU

Po wykonaniu powyższego kodu macierz A będzie w postaci:

$$A' = \begin{bmatrix} u_{11} & u_{12} & u_{13} & \dots & u_{1n} \\ l_{21} & u_{22} & u_{23} & \dots & u_{2n} \\ l_{31} & l_{32} & u_{33} & \dots & u_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ l_{n1} & l_{n2} & l_{n3} & \dots & u_{nn} \end{bmatrix} \tag{6}$$

2.3 Rozwiązanie za pomocą funkcji biblioteki gsl

```c
gsl_linalg_LU_solve(A, p, b, lib_solution);
```

Listing 3: Krótkie rozwiązanie za pomocą biblioteki **gsl**

Po wykonaniu powyższego kodu wyniki będą zapisane w zmiennej **lib_solution**.

2.4 Rozwiązanie krok po kroku

```
gsl_permute_vector(p, b); // this gives us `pb` multiplication and saves it to b

gsl_matrix *L = extractLMatrix(A); // get L matrix
gsl_matrix *U = extractUMatrix(A); // get U matrix

gsl_permutation *inv_perm = createInversePermutation(n);   // permutation, that reorder columns 0 1 2 -> 2 1 0
gsl_matrix *inv_perm_m = get_permutation_matrix(inv_perm); // matrix, that represents permutation

/*
    Our solveGaussianEquation does not accept  matrix as
    | 1 0 0 |
    | 2 2 0 |
    | 3 4 1 |

    but only like:

    | 1 4 3 |
    | 0 2 2 |
    | 0 0 1 |

    So we have to permutate by rows and by cols one time
    Also we have to permutate b once and y after gathering results
*/

gsl_permute_vector(inv_perm, b); // permutating b

gsl_matrix *L_upper = gsl_matrix_calloc(n, n); // create copy of L
gsl_matrix_memcpy(L_upper, L);
gsl_blas_dgemm(CblasNoTrans, CblasNoTrans, 1.0, inv_perm_m, L, 0.0, L_upper); // permutate rows
gsl_permute_matrix(inv_perm, L_upper);                                         // permute columns

gsl_vector *y = solveGaussianEquation(L_upper, b);

gsl_permute_vector(inv_perm, y); // permutate y after gathering solution. So y is ready to pass to the next equation

gsl_vector *x = solveGaussianEquation(U, y); // Ux = y
```

Listing 4: Przykładowe rozwiązanie krok po kroku

W kodzie powyżej są zastosowane funkcje pomocnicze. Ich implementacja nie jest tutaj umieszczona, natomiast treść ich można zobaczyć w dołączonym kodzie źródłowym.

3 Wyniki

```
Matrix A:
5.000000 4.000000 3.000000 2.000000 1.000000 9.000000
1.000000 8.000000 9.000000 3.000000 3.000000 1.000000
7.000000 4.000000 5.000000 0.000000 4.000000 3.000000
9.000000 0.000000 6.000000 4.000000 1.000000 3.000000
7.000000 8.000000 9.000000 1.000000 2.000000 4.000000
4.000000 0.000000 2.000000 1.000000 7.000000 8.000000


Vector b:
4.000000
9.000000
2.000000
7.000000
6.000000
15.000000


Matrix A after LU decomposition:
9.000000 0.000000 6.000000 4.000000 1.000000 3.000000
0.111111 8.000000 8.333333 2.555556 2.888889 0.666667
0.555556 0.500000 -4.500000 -1.500000 -1.000000 7.000000
0.777778 1.000000 0.888889 -3.333333 -0.777778 -5.222222
0.444444 0.000000 0.148148 0.166667 6.833333 6.500000
0.777778 0.500000 0.851852 0.933333 0.491057 -3.947425


Library solution:
-1.817658
-2.903405
4.002334
-1.524784
0.293835
1.716738


Step by step solution:
-1.817658
-2.903405
4.002334
-1.524784
0.293835
1.716738
```

Listing 5: Prezentacja wyników

4 Wnioski

- Biblioteka **gsl** dostarcza wygodny interfejs który usprawnia budowania własnych rozwiązań.
- Potrafimy rozwiązać układ równań liniowych, gdzie dla n niewiadomych są podane n równań liniowo niezależnych.
- Ze względu na zaokrąglenia i ogólnie reprezentację liczb w komputerach wynik może być obdarzony błędem, który można poprawić wprowadzając wektor reszt $\mathbf{r}$ definiując go jako: $\mathbf{r} = \mathbf{b} - A\mathbf{x}$.
- Otrzymaliśmy narzędzie za pomocą którego potrafimy rozwiązać codzienne problemy, przy pracy naukowej, bez korzystania z zewnętrznych niewygodnych narzędzi.