



<b>Imię i nazwisko</b> Meg Paskowski	<b>Temat laboratorium</b> Rozwiązywanie równań nieliniowych – metoda eliminacji Gaussa	<b>Data oddania</b> 09.05.2024r.	<b>Data ćwiczeń</b> 26.04.2024r.
<b>Prowadzący</b> dr hab. inż. Marcin Hojny		<b>Grupa laboratoryjna</b> 4	

### 1. Cel ćwiczenia

Celem laboratorium nr. 8 było zapoznanie się z pojęciem rozwiązywania równań nieliniowych metodą eliminacji Gaussa oraz zaimplementowanie tego algorytmu w wybranym języku programowania.

### 2. Wstęp teoretyczny

*Metoda eliminacji Gaussa* → jest to metoda na rozwiązywanie równań nieliniowych. Jej działanie opiera się na stopniowym przekształcaniu układu równań w sposób, który ułatwia znalezienie rozwiązania.

Sam proces eliminacji Gaussa składa się z dwóch etapów:

- I. *Eliminacji w przód (postępowanie proste)* → ten etap polega na przekształceniu macierzy współczynników układu równań do postaci trójkątnej górnej. Operacje na wierszach macierzy, takie jak zamiana wierszy, mnożenie wiersza przez skalar oraz dodawanie lub odejmowanie wielokrotności jednego wiersza do innego, prowadzą do powstania zer pod główną przekątną macierzy, co upraszcza kolejne obliczenia.

W eliminacji w przód obliczamy współczynnik  $m_{ij}$  zgodnie ze wzorem

$$m_{ij} = \frac{a_{ij}^{(k)}}{a_{jj}^{(k)}} \quad \text{dla } k=1, \dots, n \quad i=2, \dots, n, \quad j=1, \dots, n-1 \quad (1)$$

Gdzie:

- $k$  to indeks obecnego kroku w eliminacji, gdzie  $a_{ij}^{(k)}$  jest elementem diagonalnym względem którego wykonujemy eliminację w  $i$ -tym wierszu.
- $i$  oraz  $j$  to indeksy elementu, który chcemy wyzerować w macierzy.

- II. *Postępowanie wsteczne (odwrotne)* → Po uzyskaniu macierzy w formie trójkątnej górnej, przystępujemy do rozwiązywania układu równań. Proces ten zaczynamy od ostatniego równania, gdzie jest najmniej niewiadomych do obliczenia i obliczamy wartości niewiadomych wstecz, aż do pierwszego równania. Podstawienie wsteczne jest opisane wzorem:

$$x_i = \frac{b_i^{(n)} - \sum_{k=i+1}^n a_{ik}^{(n)} x_k}{a_{ii}^{(n)}} \quad \text{dla } i=n, n-1, \dots, 1 \quad (2)$$

Gdzie:

- $x_i$  – niewiadoma w układzie równań,
- $b_i^{(n)}$  – kolejny ( $i$ ) element wyrazów wolnych po wykonaniu eliminacji w przód,
- $a_{ij}^{(n)}$  – element na przekątnej macierzy po wykonaniu eliminacji w przód,
- $\sum_{k=i+1}^n a_{ik}^{(n)} x_k$  – suma iloczynów elementów macierzy znajdujących się w  $i$ -tym wierszu, od kolumn  $i+1$  do ostatniej kolumny  $n$ , i odpowiadających im już obliczonych wartości niewiadomych  $x_k$ .

### 3. Implementacja

W ramach ćwiczeń zaimplementowano metodę „*gaussElimination()*” służącą do rozwiązywania równań nieliniowych metodą Gaussa w języku C++.

Na samym początku zdefiniowana jest metoda służąca do wczytania danych z pliku tekstowego do macierzy i wektora. Jako parametry przyjmują:

- nazwę pliku „*fileName*” z którego chcemy wczytać dane. Parametr ten wskazuje ścieżkę do pliku tekstowego zawierającego dane układu równań.
- macierz „*array*”, który jest dwuwymiarowym wektorem.
- wektor „*vector1*”, który będzie przechowywał wartości wyrazów wolnych.

Po uruchomieniu funkcji sprawdzane jest, czy plik został poprawnie otwarty. Jeśli nie, wyświetla komunikat o błędzie i kończy funkcję. Następnie Wczytuje rozmiar układu równań z pliku i przypisuje go do zmiennej „*size*”. Zmienia rozmiar macierzy „*array*” na „*size x size*” oraz wektora „*vector1*” na rozmiar „*size*”. Później wczytuje elementy macierzy do macierzy „*array*” za pomocą dwóch zagnieżdżonych pętli for. Pierwsza pętla iteruje po wierszach macierzy, a druga po kolumnach. Po wczytaniu macierzy, wczytuje wartości wyrazów wolnych do wektora „*vector1*”. Na koniec jest zamykany plik.

```
void readData(const string& fileName, vector<vector<double>>& array, vector<double>& vector1) {
    ifstream plik(fileName);

    if (!plik.is_open()) {
        cout << "Can't open this file :<" << endl;
        return;
    }

    int size;
    plik >> size;

    array.resize(size, vector<double>(size));
    vector1.resize(size);

    for (int i = 0; i < size; ++i) {
        for (int j = 0; j < size; ++j) {
            plik >> array[i][j];
        }
    }

    for (int i = 0; i < size; ++i) {
        plik >> vector1[i];
    }

    plik.close();
}
```

Najważniejszą utworzoną metodą jest „*gaussElimination()*”, która realizuje proces eliminacji Gaussa w celu rozwiązania układu równań liniowych. Jest to funkcja, która przyjmuje dwa argumenty „*A*” referencje do dwuwymiarowego wektora, który zawiera współczynniki macierzy układu równań oraz „*b*” referencje do jednowymiarowego wektora, który zawiera wyrazy wolne układu równań. Na początku obliczana jest wielkość układu równań, czyli liczba równań  $n$ , na podstawie rozmiaru macierzy „*A*”. Następnie wykonuje się eliminacja Gaussa w przód (1), która składa się z dwóch zagnieżdżonych pętli for. Pierwsza pętla iteruje po wierszach macierzy „*A*”, a druga po wierszach poniżej aktualnie przetwarzanego. Dla każdego kolejnego wiersza, obliczany jest współczynnik „*factor*”, który jest stosowany do eliminacji odpowiednich elementów macierzy „*A*” oraz wektora „*b*”. Po zakończeniu następuje eliminacja wsteczna (2), również realizowana za pomocą pętli for. Rozpoczyna się od ostatniego równania i przechodzi się wstecz, obliczając wartości niewiadomych  $x$ . Każda wartość  $x[i]$  jest wyliczana na podstawie już obliczonych wartości  $x[j]$  dla  $j > i$ . Ostatecznie, wyliczone wartości  $x$  są zwracane jako wektor, który zawiera rozwiązania układu równań.

```
// Funkcja do eliminacji Gaussa
vector<double> gaussElimination(vector<vector<double>>& A, vector<double>& b) {
    int n = A.size();

    // Postępowanie proste (eliminacja w przód)
    for (int i = 0; i < n - 1; ++i) {
```

```
for (int k = i + 1; k < n; ++k) {
    double factor = A[k][i] / A[i][i];

    for (int j = i; j < n; ++j) {
        A[k][j] -= factor * A[i][j];
    }
    b[k] -= factor * b[i];
}

// Postępowanie odwrotne (eliminacja wsteczna)
vector<double> x(n);

for (int i = n - 1; i >= 0; --i) {
    x[i] = b[i];

    for (int j = i + 1; j < n; ++j) {
        x[i] -= A[i][j] * x[j];
    }
    x[i] /= A[i][i];
}

return x;
}
```

Została napisana również pomocnicza do wyświetlania macierzy i wektora. Przyjmuje dwie stałe referencje „array”, która jest dwuwymiarowym wektorem przechowującym współczynniki macierzy, oraz „vector1”, który jest jednowymiarowym wektorem przechowującym wartości wektora wyrazów wolnych. Wyświetla nagłówek informujący o wyświetlanej macierzy. Pętla zagnieżdżona „for” jest używana do iteracji przez każdy element macierzy „array”. Zewnętrzna pętla przechodzi przez wiersze macierzy, a wewnętrzna pętla przechodzi przez elementy w danym wierszu. Dla każdego elementu macierzy, jego wartość jest wyświetlana na konsoli, po czym następuje przejście do kolejnego wiersza. Dalej wyświetlany jest nagłówek informujący o wektorze. Druga pętla for-each jest używana do iteracji przez wartości wektora „vector1”. Dla każdej wartości wektora, jej wartość jest wyświetlana na konsoli. Po wyświetleniu zawartości macierzy i wektora, są wyświetlane dwa znaki nowej linii w celu zapewnienia odstępu między wynikami, a ewentualnymi kolejnymi komunikatami w konsoli.

```
// Funkcja do wyświetlania macierzy i wektora
void printData(const vector<vector<double>>& array, const vector<double>& vector1) {
    cout << "Array A:" << endl;

    for (int i = 0; i < array.size(); ++i) {
        for (int j = 0; j < array[i].size(); ++j) {
            cout << array[i][j] << " ";
        }
        cout << endl;
    }

    cout << "\nVector b:" << endl;

    for (double value : vector1) {
        cout << value << " ";
    }
    cout << endl << endl;
}
```

Program główny, w którym określana jest nazwa wczytywanego pliku „fileName”, deklarowana macierz „array” i wektor „vector1”. Następnie wywoływane są funkcje do odczytu pliku, eliminacji Gaussa i wyświetlania macierzy i wektora w celu sprawdzenia. Poniżej wyświetlany zostaje wynik korzystamy do tego z pętli for.

```
//Funkcje nieliniowe - metoda eliminacji Gaussa

string fileName = "data2.txt";
vector<vector<double>> array;
vector<double> vector1;

readData(fileName, array, vector1);

// Wyświetlenie macierzy i wektora
//printData(array, vector1);

// Wywołanie eliminacji Gaussa
vector<double> result = gaussElimination(array, vector1);
```



```
// Wyświetlenie wyniku
int i = 0;
cout << "Results:" << endl;

for (double value : result) {
    cout << "x_" << i << " = " << value << endl;
    i++;
}

system("PAUSE");
return 0;
```

#### 4. Testy jednostkowe i opracowanie wyników

Testy zostały przeprowadzone dla kilku macierzy oraz porównane zostały do wyników otrzymanych przy wykorzystaniu biblioteki NumPy w języku Python.

##### I. Test dla macierzy o wymiarach 4x4

Macierz 4x4:

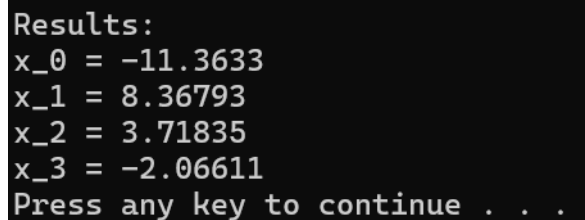
```
0.05240293 0.41151978 0.01728522 0.93023833
0.79198706 0.93952318 0.53597951 0.32951795
0.4669055 0.41167278 0.91199438 0.4975632
0.4451451 0.66918596 0.4108247 0.96776594
```

Wektor wyrazów wolnych:

[0.99039894 0.17441499 0.50235493 0.069464]

Uzyskane wyniki programu:

[-11.3633 8.36793 3.71835 -2.06611]



Rysunek 1 Uzyskane wyniki dla macierzy 4x4 dla testu 1

Uzyskane wyniki za pomocą biblioteki NumPy:

[-11.36330811 8.36793419 3.71835055 -2.06610851]

##### II. Test dla macierzy o wymiarach 8x8

Macierz 8x8:



0.35503633 0.99614184 0.90350923 0.75461324 0.50480139 0.24625487 0.65907911 0.54062691  
0.2788147 0.5208278 0.92714981 0.74548396 0.94294734 0.0376622 0.73292614 0.97799828  
0.08345718 0.14076835 0.48190074 0.69659405 0.70057489 0.50921185 0.40985468 0.06924001  
0.09553334 0.68431079 0.52900368 0.86482548 0.43585165 0.58028101 0.60475178 0.25486637  
0.16750753 0.77905995 0.81610932 0.58039289 0.06995159 0.81908688 0.4473171 0.66475249  
0.47270601 0.98645452 0.93436872 0.78584301 0.69439694 0.49607394 0.03054604 0.3029881  
0.7606576 0.04351309 0.35056603 0.74041984 0.35049311 0.65038365 0.44402514 0.65632692  
0.97468717 0.215221 0.66298968 0.8186079 0.41527323 0.80947391 0.89376695 0.24972451

Wektor wyrazów wolnych:

[0.74778131 0.17427122 0.74980459 0.64989685 0.04187962 0.699739960.07890859 0.50827138]

Uzyskane wyniki programu:

[-1.06074 -3.81676 5.60791 7.91188 -5.41981 -4.15314 -1.92092 -2.00894]

```
Results:
x_0 = -1.06074
x_1 = -3.81676
x_2 = 5.60791
x_3 = 7.91188
x_4 = -5.41981
x_5 = -4.15314
x_6 = -1.92092
x_7 = -2.00894
Press any key to continue . . . |
```

*Rysunek 2 Uzyskane wyniki dla macierzy 8x8 dla testu 2*

Uzyskane wyniki za pomocą biblioteki NumPy:

[-1.06074308 -3.81675738 5.60790788 7.91187792 -5.4198152 -4.15314033-1.92092007 -2.00894449]

III. Test dla macierzy o wymiarach 32x32

Macierz 32x32



0.16356657319607326 0.847462936326945 0.4573256888472682 0.29392890079517475 0.840896377429247  
0.14594219552620524 0.33443191969498187 0.8709361584900053 0.5155419546368221 0.5947375249580147  
0.953656172643797 0.3658407490727372 0.7254404873787123 0.8080600924723458 0.25731188302661856  
0.31805200272549894 0.48528611614190575 0.3341155424677057 0.7569350997699014 0.7403258806503755  
0.30193879644176735 0.38088702079792847 0.028034794957836007 0.10325559276255236 0.8036554386659204  
0.7066310466689163 0.8959453711353433 0.6293025350667911 0.35430813714013054 0.939468307188218  
0.32798803395709364 0.6609084385389792

[...]

0.8738925676360768 0.81269324946497 0.5826762081491521 0.7355727987890974 0.4497802011152444  
0.3712065665821239 0.7672164664444415 0.6532264247235592 0.17710714618527235 0.8295063096646208  
0.3861218549016847 0.4617277314859698 0.6818145070717463 0.3199542705457664 0.07334390842479699  
0.6454700150382776 0.15070764614859145 0.828898534288515 0.18337521055384887 0.9084676784260033  
0.7692348736248525 0.5207969763031517 0.06972102384734413 0.1339086018004626 0.6831046007812454  
0.49233951697753886 0.9296965931315834 0.20822559009150632 0.6880371647091474 0.8486919528663516  
0.3292970286461335 0.7771091652441591

0.38076097644687845 0.6696746170150198 0.6684194985194456 0.5187512121220574 0.21722724302303065  
0.9678380605349961 0.4192687452326521 0.21284851914747505 0.9008329931323095 0.060999230774417046  
0.2466548990704081 0.8289571836241469 0.7699237899274538 0.866905711867068 0.1132680912059808  
0.572844031639792 0.831731638420663 0.9372583438820155 0.02818034941494263 0.2410591329852998  
0.39313839054681754 0.5406805430007269 0.4710109720352661 0.9120904598104922 0.25343547508426245  
0.9288613859057657 0.8277348441951708 0.9187082253249683 0.7197807997282365 0.7163595331966074  
0.8432544958415059 0.41727862597556953

#### Wektor wyrazów wolnych

[0.19083872 0.77978427 0.56748051 0.85128655 0.36388905 0.383244350.59918682 0.00731203  
0.99159636 0.88012886 0.99024816 0.829506290.77623393 0.23863549 0.13637956 0.47730773  
0.07647257 0.584632060.34792354 0.59250155 0.35825446 0.10359755 0.31908355  
0.737803890.75433043 0.71373516 0.74192184 0.99028757 0.71167463 0.999602660.75837673  
0.92382883]

#### Uzyskane wyniki programu:

[0.581246 0.356766 0.263445 0.00650421 -0.0832432 -0.0296259 -0.56484 0.448898 -0.13901 1.29654  
0.198362 -0.169169 0.524392 0.310108 0.70301 -1.45027 0.552583 0.195791 -1.52003 0.16911 0.672104  
0.0271542 0.606247 0.0764653 -0.668679 0.566995 -0.526878 -0.296502 -0.353271 0.183382 0.251548 -  
0.77875]

```
Results:
x_0 = 0.581246
x_1 = 0.356766
x_2 = 0.263445
x_3 = 0.00650421
x_4 = -0.0832432
x_5 = -0.0296259
x_6 = -0.56484
x_7 = 0.448898
x_8 = -0.13901
x_9 = 1.29654
x_10 = 0.198362
x_11 = -0.169169
x_12 = 0.524392
x_13 = 0.310108
x_14 = 0.70301
x_15 = -1.45027
x_16 = 0.552583
x_17 = 0.195791
x_18 = -1.52003
x_19 = 0.16911
x_20 = 0.672104
x_21 = 0.0271542
x_22 = 0.606247
x_23 = 0.0764653
x_24 = -0.668679
x_25 = 0.566995
x_26 = -0.526878
x_27 = -0.296502
x_28 = -0.353271
x_29 = 0.183382
x_30 = 0.251548
x_31 = -0.77875
Press any key to continue . . .
```

Rysunek 3 Uzyskane wyniki dla macierzy 32x32 dla testu 3

Uzyskane wyniki za pomocą biblioteki NumPy:

```
[ 0.58124615 0.35676622 0.2634445 0.00650421 -0.08324324 -0.02962593 -0.56483981 0.44889845 -
 0.13900973 1.29653712 0.19836197 -0.16916866 0.5243924 0.31010848 0.70300966 -1.45026902
 0.55258342 0.19579118 -1.52002985 0.16911028 0.67210374 0.02715415 0.6062468 0.07646533 -
 0.66867873 0.56699451 -0.52687753 -0.29650213 -0.35327072 0.1833820 0.25154803 -0.77875003]
```

#### IV. Test dla macierzy o wymiarach 64x64

Macierz 64x64

```
[[0.07732542 0.69080602 0.56486043 ... 0.70193536 0.782256 0.80816733][0.04269135 0.34831919
 0.0524162 ... 0.95482935 0.27537107 0.12765267][0.26548567 0.76950876 0.89579761 ... 0.21357748
 0.53319964 0.55231103]...[0.77617086 0.49368094 0.17788363 ... 0.54450774 0.26721449
 0.04166852][0.48462084 0.01201553 0.75518128 ... 0.74857171 0.91111402 0.53519104][0.17160141
 0.31978045 0.23346088 ... 0.10104148 0.74020103 0.6457542 ]]
```

Wektor wyrazów wolnych

0.483742438290778 0.13715576623153858 0.06848453951354838 0.4275312598373989  
0.2005253490757979 0.081924004771192 0.6672323159126392 0.39479655927969115  
0.5548829079998544 0.7485260698633667 0.24482867198961822 0.695800285877128  
0.4786538043872831 0.5557935493807103 0.7060522557459477 0.507575033183954  
0.4581623080132363 0.4717603773147089 0.3863992196478738 0.7416797473497078  
0.7943073372861332 0.2771878985863162 0.48043828426883495 0.057298766705537685  
0.83486693116996 0.049779634056873134 0.6396551678331314 0.41437639968460704  
0.7842208965328312 0.7231601443385492 0.547787902914496 0.8543550426724618  
0.9034131118168562 0.42258367137739816 0.002912147313137714 0.9240226971886831  
0.16929502612871483 0.24419476600316514 0.3075784703283102 0.9971170757261958  
0.4651961366175401 0.16103884506465027 0.48994844085671274 0.6536633674830813  
0.904934842204449 0.19046724997583953 0.7659441525667814 0.7524583128802386  
0.3301079233107881 0.09235317612173277 0.9279251217703676 0.7859098444094023  
0.6278202885598473 0.6117634438861606 0.9439852350567384 0.2819777556187364  
0.5203551191730339 0.5602951442092619 0.634011407093944 0.015403955278417092  
0.6536895979678893 0.5841647277431327 0.30691241029508287 0.7041382673136093

Uzyskane wyniki programu:

[0.79269 0.267875 -0.865825 0.0278211 -0.660458 0.704709 -1.52899 -0.338068 1.25233 -0.203851 -  
2.25559 0.287653 0.109583 -0.345637 0.140645 0.0768109 -0.700739 -1.91267 -0.0461092 -0.354357 -  
1.20929 0.0240924 0.24666 -0.773547 0.84702 -1.18375 -0.649756 0.0859323 0.128174 0.0585711  
0.276219 1.2267 -0.211276 0.256003 1.49721 -1.04437 -0.906744 -1.29515 -0.677968 -0.598161 1.30506  
1.45681 0.202699 0.87143 0.411411 -0.422525 0.12287 -0.230055 1.21055 0.204553 1.4354 0.122996  
0.382944 0.417254 0.0337199 0.656514 0.252773 1.65892 1.52622 -0.446744 -0.248478 -1.01833 -  
0.0264551 0.952336]

```
Results:
x_0 = 0.79269
x_1 = 0.267875
x_2 = -0.865825
x_3 = 0.0278211
x_4 = -0.660458
x_5 = 0.704709
x_6 = -1.52899
x_7 = -0.338068
x_8 = 1.25233
x_9 = -0.203851
x_10 = -2.25559
x_11 = 0.287653
x_12 = 0.109583
x_13 = -0.345637
x_14 = 0.140645
x_15 = 0.0768109
x_16 = -0.700739
x_17 = -1.91267
x_18 = -0.0461092
x_19 = -0.354357
x_20 = -1.20929
x_21 = 0.0240924
x_22 = 0.24666
x_23 = -0.773547
x_24 = 0.84702
x_25 = -1.18375
x_26 = -0.649756
x_27 = 0.0859323
x_28 = 0.128174
x_29 = 0.0585711
x_30 = 0.276219
x_31 = 1.2267
x_32 = -0.211276
x_33 = 0.256003
x_34 = 1.49721
x_32 = -0.211276
x_33 = 0.256003
x_34 = 1.49721
x_35 = -1.04437
x_36 = -0.906744
x_37 = -1.29515
x_38 = -0.677968
x_39 = -0.598161
x_40 = 1.30506
x_41 = 1.45681
x_42 = 0.202699
x_43 = 0.87143
x_44 = 0.411411
x_45 = -0.422525
x_46 = 0.12287
x_47 = -0.230055
x_48 = 1.21055
x_49 = 0.204553
x_50 = 1.4354
x_51 = 0.122996
x_52 = 0.382944
x_53 = 0.417254
x_54 = 0.0337199
x_55 = 0.656514
x_56 = 0.252773
x_57 = 1.65892
x_58 = 1.52622
x_59 = -0.446744
x_60 = -0.248478
x_61 = -1.01833
x_62 = -0.0264551
x_63 = 0.952336
Press any key to continue . . .
```

Rysunek 4 Uzyskane wyniki dla macierzy 64x64 dla testu 4





Uzyskane wyniki za pomocą biblioteki NumPy:

```
[ 0.79269013 0.26787528 -0.86582521 0.0278211 -0.66045778 0.70470871 -1.52898548 -0.33806808
 1.25232504 -0.20385066 -2.25559365 0.287652980.10958343 -0.3456374 0.14064514 0.0768109 -
 0.7007388 -1.9126747-0.04610924 -0.35435717 -1.20928648 0.02409241 0.2466602 -
 0.773546560.84701979 -1.18375188 -0.64975599 0.08593231 0.12817418 0.058571130.27621918
 1.2267036 -0.21127566 0.25600293 1.49721138 -1.0443738-0.90674398 -1.29515332 -0.67796796 -
 0.59816073 1.30506104 1.456805240.20269922 0.87142973 0.41141103 -0.42252489 0.12287004 -
 0.230055071.21055376 0.20455348 1.43540375 0.12299578 0.38294412 0.417254250.03371994
 0.65651387 0.25277309 1.65892424 1.52621664 -0.44674438-0.24847828 -1.01833369 -0.02645512
 0.95233554]
```

#### V. Test dla macierzy o wymiarach 128X128

Macierz 128x128

```
[[0.99945538 0.37184501 0.96355223 ... 0.6115106 0.30059532 0.9789582 ][0.47089245 0.93273701
 0.54850406 ... 0.9018389 0.83667224 0.26702869][0.74622057 0.08974293 0.5810069 ... 0.24358955
 0.80658236 0.73955432]...[0.76138694 0.31965692 0.79327819 ... 0.00593711 0.88672335
 0.27776449][0.76090067 0.92667626 0.00875271 ... 0.67237521 0.46117282 0.1572895 ][0.84262433
 0.14224887 0.51996042 ... 0.01157398 0.1648274 0.29569602]]
```

Wektor wyrazów wolnych

```
[8.72910280e-01 4.30863170e-01 8.10221323e-01 8.98004155e-028.73282660e-01 3.59577605e-01
 1.14132191e-02 7.82970214e-015.81647791e-01 4.51016439e-01 3.76167387e-01 9.29059570e-
 011.09197057e-01 1.39997946e-01 6.78900674e-01 3.12940720e-016.18057096e-01 5.06587746e-02
 1.29228084e-01 9.13780063e-018.67197487e-01 8.07817659e-01 9.78305724e-01 2.54204363e-
 011.02634056e-01 4.83248762e-01 3.94998583e-01 1.45971238e-011.38381577e-01 7.40441996e-01
 3.95870733e-05 8.28521684e-011.65782527e-01 3.46839861e-01 6.75672193e-01 6.49760315e-
 011.10517764e-01 4.47892415e-01 5.28062676e-01 5.70294247e-019.44630819e-01 1.67649253e-01
 8.41304408e-01 9.44207227e-017.11141122e-01 6.71922662e-01 8.12010181e-01 2.87227438e-
 016.39931632e-01 2.62215260e-01 5.96058309e-01 6.13214540e-012.18144748e-01 8.87661511e-01
 9.04053555e-02 2.46704750e-014.80980387e-01 8.38143726e-01 7.44786520e-01 3.89376844e-
 019.89070035e-01 4.17191182e-01 6.42162510e-01 2.80137933e-017.86871452e-01 7.68532252e-01
 2.30107841e-01 2.44412790e-016.76059482e-01 9.29795451e-01 9.43413743e-01 1.71196746e-
 012.76625936e-01 2.84498321e-01 4.06549627e-01 1.61999900e-012.92247285e-01 3.50552906e-01
 9.11796524e-01 4.71121136e-013.70257066e-01 8.32808190e-02 5.38431475e-01 3.38584365e-
 014.32591966e-01 3.69965189e-01 9.61702356e-01 1.56105154e-014.66772181e-01 5.53968799e-01
 6.11924864e-01 6.76027222e-016.35831669e-02 5.43471090e-01 8.04940877e-01 8.35718342e-
 012.53900515e-01 3.33603317e-02 9.09236400e-01 8.25654337e-018.21380107e-01 4.48427505e-01
 2.69618479e-01 7.89044113e-017.36207512e-01 2.37556777e-01 6.81666893e-01 9.90924076e-
 024.71456916e-01 3.96163434e-01 3.71188381e-01 3.17740770e-015.05881848e-01 8.54082764e-01
 9.53079940e-01 8.01885279e-024.86386404e-01 7.09976411e-01 2.36738014e-01 5.42719415e-
 026.92837939e-01 3.76655465e-01 1.99437504e-01 7.99282925e-011.44243173e-01 4.52059696e-01
 5.13621087e-01 7.24255002e-01]
```

Uzyskane wyniki programu:

```
[6.74921 0.881665 5.19811 5.03754 9.68288 -1.13338 1.35866 -3.99841 8.28083 5.18417 3.1971 3.33936
 5.31908 0.785454 -3.66727 -1.22757 3.5303 0.798859 9.13931 -0.950532 -3.27915 5.40973 -2.15132
 8.34017 -6.59527 3.7456 2.17508 -2.36353 -2.0107 -2.06768 0.547406 -0.665161 2.91545 1.62279
```

1.18841 -0.178501 0.428088 1.66745 -1.28581 -3.4444 -3.88333 -3.36762 -0.0528909 -2.71701 2.84221  
 1.99351 1.51778 -0.973098 -2.44246 -5.84536 -1.94617 1.8749 2.21355 -2.32319 -2.07224 5.06159  
 2.91766 8.74153 -2.02098 -3.08128 1.84399 -5.28684 -0.793045 -2.0178 -7.06477 3.34778 -4.84731 -  
 2.48019 2.98659 0.794935 -1.31572 10.8529 5.41229 -3.27017 -10.4942 -11.8487 0.929647 1.9832  
 5.65298 -8.16763 -3.83895 2.67652 4.515 -2.34286 5.04484 -3.88965 -7.74793 -0.0995476 11.7776 -  
 0.385754 -2.29456 -3.85567 0.744749 4.07959 0.790174 4.14157 4.59755 0.640439 -3.92358 1.64836  
 5.29565 7.34332 -4.86623 -5.14513 0.419961 -2.5446 -2.09238 5.12673 0.49862 -6.43374 2.34178 -  
 1.09493 2.86086 -5.31949 -2.97713 -1.99181 4.29383 -3.39579 -0.457189 -2.60427 -5.62836 -9.57985  
 2.02824 -4.72996 -3.6216 -8.40785 1.7097 -8.07096]

```
Results:
x_0 = 6.74921      x_34 = 1.18841      x_69 = 0.794935
x_1 = 0.881665     x_35 = -0.178501   x_70 = -1.31572
x_2 = 5.19811      x_36 = 0.428088    x_71 = 10.8529
x_3 = 5.03754      x_37 = 1.66745     x_72 = 5.41229
x_4 = 9.68288      x_38 = -1.28581    x_73 = -3.27017
x_5 = -1.13338     x_39 = -3.4444     x_74 = -10.4942
x_6 = 1.35866      x_40 = -3.88333    x_75 = -11.8487
x_7 = -3.99841     x_41 = -3.36762    x_76 = 0.929647
x_8 = 8.28083      x_42 = -0.0528909  x_77 = 1.9832
x_9 = 5.18417      x_43 = -2.71701    x_78 = 5.65298
x_10 = 3.1971      x_44 = 2.84221     x_79 = -8.16763
x_11 = 3.33936     x_45 = 1.99351     x_80 = -3.83895
x_12 = 5.31908     x_46 = 1.51778     x_81 = 2.67652
x_13 = 0.785454    x_47 = -0.973098  x_82 = 4.515
x_14 = -3.66727    x_48 = -2.44246    x_83 = -2.34286
x_15 = -1.22757    x_49 = -5.84536    x_84 = 5.04484
x_16 = 3.5303      x_50 = -1.94617    x_85 = -3.88965
x_17 = 0.798859    x_51 = 1.8749      x_86 = -7.74793
x_18 = 9.13931     x_52 = 2.21355     x_87 = -0.0995476
x_19 = -0.950532   x_53 = -2.32319    x_88 = 11.7776
x_20 = -3.27915    x_54 = -2.07224    x_89 = -0.385754
x_21 = 5.40973     x_55 = 5.06159     x_90 = -2.29456
x_22 = -2.15132    x_56 = 2.91766     x_91 = -3.85567
x_23 = 8.34017     x_57 = 8.74153     x_92 = 0.744749
x_24 = -6.59527    x_58 = -2.02098    x_93 = 4.07959
x_25 = 3.7456      x_59 = -3.08128    x_94 = 0.790174
x_26 = 2.17508     x_60 = 1.84399     x_95 = 4.14157
x_27 = -2.36353    x_61 = -5.28684    x_96 = 4.59755
x_28 = -2.0107     x_62 = -0.793045   x_97 = 0.640439
x_29 = -2.06768    x_63 = -2.0178     x_98 = -3.92358
x_30 = 0.547406    x_64 = -7.06477    x_99 = 1.64836
x_31 = -0.665161   x_65 = 3.34778     x_100 = 5.29565
x_32 = 2.91545     x_66 = -4.84731    x_101 = 7.34332
x_33 = 1.62279     x_67 = -2.48019    x_102 = -4.86623
x_68 = 2.98659     x_103 = -5.14513   x_103 = -5.14513
x_104 = 0.419961
x_105 = -2.5446
x_106 = -2.09238
x_107 = 5.12673
x_108 = 0.49862
x_109 = -6.43374
x_110 = 2.34178
x_111 = -1.09493
x_112 = 2.86086
x_113 = -5.31949
x_114 = -2.97713
x_115 = -1.99181
x_116 = 4.29383
x_117 = -3.39579
x_118 = -0.457189
x_119 = -2.60427
x_120 = -5.62836
x_121 = -9.57985
x_122 = 2.02824
x_123 = -4.72996
x_124 = -3.6216
x_125 = -8.40785
x_126 = 1.7097
x_127 = -8.07096
Press any key to continue . . .
```

Rysunek 5 Uzyskane wyniki dla macierzy 128x128 dla testu 5

Uzyskane wyniki za pomocą biblioteki NumPy:

[ 6.74920963 0.88166534 5.19810779 5.03754037 9.68288009 -1.13337813 1.35866124 -3.99840762  
 8.28083427 5.184173083 1.9710251 3.33935526 5.3190751 0.78545401 -3.6672651 -1.22757418 3.53029922  
 0.79885869 9.13930545 -0.95053166 -3.2791475 5.40972796 -2.15132443 8.34017324 -6.59526833 7.4559773  
 2.17508103 -2.36352944 -2.01070073 -2.067675130 5.4740616 -0.66516105 2.91545353 1.62279453  
 1.1884112 -0.17850093 0.42808793 1.66745059 -1.28581478 -3.44439529 -3.88332516 -3.36761855 -  
 0.05289088 -2.71700553 2.842210331 1.9935147 1.51778415 -0.97309814 -2.44246096 -5.84535935 -1.9461672  
 1.87490059 2.21355255 -2.32319026 -2.072239415 5.06159361 2.91766057 8.74152876 -2.02097953 -  
 3.08127521 1.84398803 -5.28684463 -0.79304498 -2.01780259 -7.064767383 3.34778331 -4.84730987 -  
 2.48019376 2.98659203 0.79493459 -1.31572079 10.85293296 5.41229298 -3.27016513 -10.49421349 -  
 11.84872578 0.92964703 1.98320218 5.65297857 -8.16763307 -3.83894819 2.67651606 4.5150042 -  
 2.34286406 5.04483644 -3.88965434 -7.74793015 -0.09954757 11.77758787 -0.38575411 -2.29455879 -  
 3.85566663 0.74474918 4.07959392 0.790173934 4.14157371 4.59755495 0.6404392 -3.92357986  
 1.648360395 2.9564688 7.34332032 -4.86623235 -5.145132 0.41996118 -2.5446022 -2.09238101 5.12672654  
 0.49862037 -6.433735582 3.4177665 -1.09493228 2.86085865 -5.31948923 -2.97713041 -1.99180951  
 4.2938281 -3.39578966 -0.45718853 -2.60427289 -5.62836117 -9.57984753 2.02823901 -4.72995967 -  
 3.62160141 -8.40784942 1.70969668 -8.07096279]

Jak można zauważyć po przeprowadzonych testach w wynikach występują niewielkie różnice wynikające z założonych zaokrąglenia użytych w poszczególnych implementacjach. Mogą one prowadzić do akumulacji błędów numerycznych w trakcie wykonywania kolejnych operacji, co może wpłynąć na końcowe wyniki. Dlatego też, porównując wyniki testów, istotne jest uwzględnienie różnic w strategiach zaokrąglania. Różnice między wynikami programu, a biblioteki NumPy mogą także wynikać z różnych algorytmów, implementacji, obsługi typów danych oraz zastosowanych optymalizacji, ponieważ biblioteka NumPy może wykorzystywać zoptymalizowane implementacje algorytmów matematycznych, które mogą prowadzić do bardziej precyzyjnych wyników lub być bardziej odporne na błędy numeryczne w porównaniu do własnych implementacji.

## **5. Wnioski**

Testy przeprowadzone dla metody eliminacji Gaussa w rozwiązywaniu układów równań liniowych objęły różnorodne wielkości macierzy. Rezultaty uzyskane z zastosowaniem tej metody pokrywały się z wynikami otrzymanymi przy wykorzystaniu biblioteki NumPy w języku Python, co stanowiło potwierdzenie poprawności implementacji. Metoda eliminacji Gaussa okazała się skuteczną metodą do rozwiązywania układów równań liniowych o różnych wielkościach. Wyniki uzyskane dla macierzy o różnych wymiarach potwierdziły, że algorytm działa zgodnie z założeniami teoretycznymi. Nawet dla bardzo dużych macierzy, takich jak 128x128, uzyskane wyniki były zgodne z oczekiwaniami.

## **6. Źródła**

- Prezentacja autorstwa dr hab. inż. Marcina Hojnego „Rozwiązywanie równań nieliniowych – metoda eliminacji Gaussa”.