Rozwiązywanie układu równań liniowych AX = B, gdzie $A, B \in R^{n \times n}$, za pomocą zmodyfikowanej metody Crouta. Wyznaczenie macierzy odwrotnej A^{-1} oraz wyznacznika det(A)

Elżbieta Jowik, Filip Chrzuszcz

December 8, 2019

Contents

1 Wstęp

1.1 Opis metody

Metoda Crouta pozwala rozwiązać układ n równań z n niewiadomymi.
Polega ona ona wyznaczeniu rozkładu macierzy A na iloczyn macierzy L oraz U (A = LU), gdzie U jest macierzą trójkątną górną z jedynkami na głównej przekątnej a L jest macierzą trójkątną dolną.
Wykorzystanie tego rozkładu pozwala na rozwiązywanie układu równań Ax = b, odwracanie macierzy A czy obliczenie det(A) przy ograniczonej liczbie wykonywanych operacji. Rozważamy macierz $A \in R^{n \times n}$ postaci:

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ a_{31} & a_{32} & \dots & a_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{n,n} \end{bmatrix}$$

Wprowadzona przez nas modyfikacja klasycznej metody Crouta polega na odwróceniu kolejności wymnażania macierzy L i U. Innymi słowy, szukamy macierzy L i U, które po przeprowadzeniu rozkładu są postaci:

$$U = \begin{bmatrix} 1 & u_{12} & u_{13} & \dots & u_{1n} \\ 0 & 1 & u_{23} & \dots & u_{2n} \\ 0 & 0 & 1 & \dots & u_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix} \quad L = \begin{bmatrix} l_{11} & 0 & 0 & \dots & 0 \\ l_{21} & l_{22} & 0 & \dots & 0 \\ l_{31} & l_{32} & l_{33} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \dots & l_{n,n} \end{bmatrix}$$

takich, że A = UL.

1.2 Wzory na rozkład UL

Elementy macierzy L i U wyliczamy korzystając z następujących wzorów:

$l_{nn} = a_{nn}$

1. dla $i \geq j$

$$l_{ij} = a_{ij} - \sum_{p=i+1}^{n} u_{ip} l_{pj}$$

2. dla $i < j$

$$u_{ij} = \frac{1}{l_{jj}}(a_{ij} - \sum_{p=j+1}^{n} u_{ip} l_{pj})$$

Wzory te otrzymaliśmy mnożąc przez siebie macierze U i L i porównując współczynniki otrzymanej macierzy wynikowej ze współczynnykami macierzy A.

1.3 Obliczanie wyznacznika macierzy A z wykorzystaniem otrzymanego rozkładu UL

Otrzymany rozkład w znacznym stopniu ułatwia obliczenie wyznacznika macierzy A. Wystarczy skorzystać z własności wyznacznika dla macierzy trójkątnych.

$det(A) = det(U \cdot L) = det(U) \cdot det(L) = \prod_{i=1}^{n} u_{ii} \cdot \prod_{i=1}^{n} l_{ii} = \prod_{i=1}^{n} l_{ii}$

bo:

$\prod_{i=1}^{n} u_{ii} = \prod_{i=1}^{n} 1 = 1$

1.4 Obliczenie macierzy odwrotnej A^{-1} z wykorzystaniem rozkładu UL

Posiadając rozkład UL macierzy A jesteśmy w stanie efektywnie obliczyć macierz odwrotną do macierzy A. Robimy to korzystając ze wzoru:

$$A^{-1} = L^{-1} \cdot U^{-1}$$

Dzięki temu że obliczenie macierzy odwrotnej do macierzy trójkątnej jest tanie, to w łatwy sposób otrzymujemy macierz odwrotną do A.

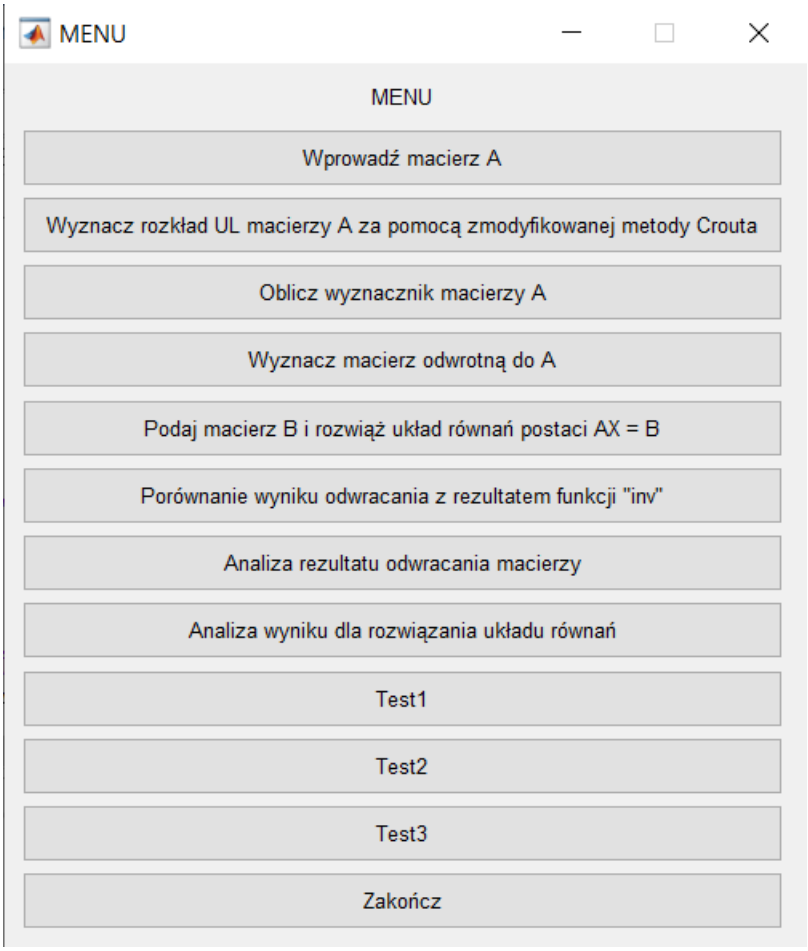
1.5 Wyznaczanie rozwiązań układu równań $AX = B$, gdzie $A, B \in R^{n \times n}$

Aby rozwiązać dany układ $A \cdot X = B$, możemy zastosować podstawienie $A = U \cdot L$. Otrzymujemy wówczas równanie

$$U \cdot L \cdot X = B$$

Aby z niego obliczyć X podstawiamy $L \cdot X = Y$. Mamy więc $U \cdot Y = B$. A więc dzięki rozkładowi UL udało nam się sprowadzić pierwotny układ do dwóch podukładów z macierzami trójkątnymi, które możemy łatwo rozwiązać poprzez forward oraz backward substition.

2 Analiza funkcjonalności oferowanych przez program obliczeniowy



2.1 Wprowadź macierz A

Funkcjonalność ta umożliwia użytkownikowi wprowadzenie macierzy i sprawdza czy możliwe jest wykonanie na niej operacji oferowanych przez program tzn. czy macierz jest kwadratowa i czy wszystkie jej minory główne są dodatnie. Jeżeli którykolwiek z warunków nie jest spełniony, program wypisuje odpowiedni komunikat o błędzie i kończy działanie, w przeciwnym przypadku oczekuje na dalsze instrukcje użytkownika.
Uwaga! Nie wprowadzając macierzy A możemy skorzystać jedynie z funkcjon-

alności związanych z testami!

2.2 Wyznacz rozkład UL macierzy A za pomocą zmodyfikowanej metody Crouta

Funkcjonalność odpowiada za wyznaczenie rozkładu UL za pomocą zmodyfikowanej metody Crouta. Jako argument wejścia przyjmuje macierz A wprowadzoną z wykorzystaniem funkcjonalności 2.1. Zwraca natomiast wektor macierzy: [U, L]. Pełny opis metody wraz z wyprowadzonymi wzorami znajdują się we wstępie dokumentacji (sekcje 1.1, 1.2).

2.3 Oblicz wyznacznik macierzy A

Ta funkcja umożliwia obliczenie wyznacznika macierzy A. Wykorzystana w tym celu procedura opisana jest w sekcji 1.3 dokumentacji.

2.4 Wyznacz macierz odwrotną do A

Funkcja umożliwia wyznaczenie macierzy odwrotnej do macierzy A.
Przyjmuje ona jedną macierz symetryczną, a zwraca macierz A^{-1}. Szczegóły działania są opisane w sekcji 1.4 dokumentacji.

2.5 Podaj macierz B i rozwiąż układ równań postaci AX = B

Funkcja przyjmuje jako argument dwie macierze A oraz B rozmiaru $n \times n$ i rozwiązuje układ równań $AB = X$ wykorzystując rozkład UL. Szczegóły działania funkcji opisane są w sekcji 1.5 dokumentacji.

2.6 Porównanie wyniku odwracania z rezultatem funkcji "inv"

Nasza metoda odwracania daje porównywalne wyniki z wbudowaną funkcją "inv". Oczywiście wraz ze wzrostem wskaźnika uwarunkowania macierzy błędy popełnianie przez naszą funkcję są coraz większe, aż z czasem, dla odpowiednio dużego wskaźnika uwarunkowania rezultaty kompletnie przestają mieć sens, jednakże dzieje się to również dla wbudowanej funkcji "inv, więc wyniki które podaje nasza funkcja są akceptowalne.

2.7 Analiza rezultatu odwracania macierzy

Pozwala sprawdzić dokładność naszego algorytmu odwracania macierzy, poprzez pokazanie macierzy różnicy powstałej z odjęcia wyniku wygenerowanego przez Matlaba od wyniku naszej metody. Uzyskaną różnicę wizualizuje w postaci mapy ciepła. Funkcjonalność ta generuje wyniki dynamicznie dla każdej macierzy wprowadzonej przez użytkownika, użycie jej nie wymaga wcześniejszego skorzystania z funkcjonalności odpowiadającej za wyznaczanie macierzy odwrotnej do A.

2.8 Analiza wyniku dla rowiązywanego układu równań

Pozwala sprawdzić dokładność naszego algorytmu rozwiązywania układu równań, poprzez pokazanie macierzy różnicy powstałej z odjęcia wyniku wygenerowanego przez Matlaba od wyniku naszej metody. Uzyskaną różnicę wizualizuje w postaci mapy ciepła. Funkcjonalność ta generuje wyniki dynamicznie dla każdej pary macierzy A i B wprowadzonej przez użytkownika, jednak użycie jej wymaga wcześniejszego skorzystania z funkcjonalności odpowiadającej za rozwiązywanie układu równań, gdyż jest to jedyny sposób na wprowadzenie do programu macierzy B.

2.9 Test 1

Stanowi sprawdzenie poprawności algorytmu odwracania losowo wygenerowanej macierzy Hilberta 5×5 sprowadzonej do macierzy Hessenberga.

2.10 Test 2

Pozwala sprawdzić poprawność rozwiązania układu równań złoznego z macierzy losowej oraz jednostkowej.

2.11 Test 3

Prezentuje rozwiązanie układu równań z macierzy Pascala oraz macierzy powstałej jako różnica macierzy magicznej oraz Pascala.

2.12 Zakończ

Ostatnia z opcji MENU opracowanego programu odpowiada za bezpieczne zakończenie jego działania.

3 Testy

Program został przetestowany zarówno na przykładach losowych, jak i na bardziej konkretnych. Naszym celem było zweryfikowanie dokładności realizacji trzech głównych założeń naszego programu jakimi są:

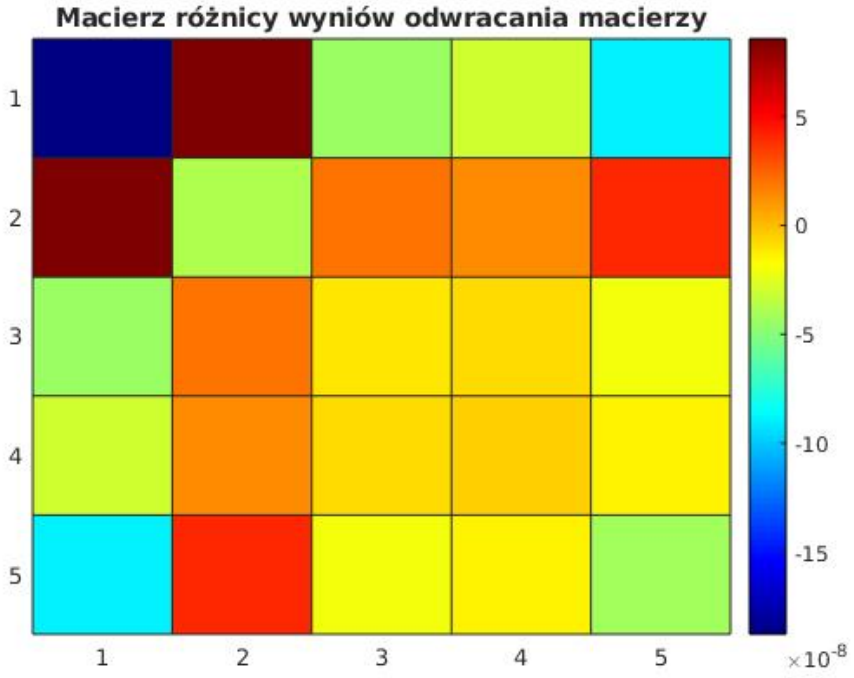
- Znajdowanie rozkładu UL
- Odwracanie macierzy
- Rozwiązanie układu równań

Wszystkie z wyżej wymienionych funkcjonalności przynosiły satysfakcjonujące rezultaty. To jak duży był błąd w sporej mierze zależało od wskaźnika uwarunkowania testowanej przez nas macierzy. Chcielibyśmy przedstawić 3 ciekawe przykłady obliczeniowe, które ukażą to co udało nam się osiągnąć projektując ten program.

3.1 Test 1: Odwracanie macierzy Hessenberga

A - losowo wygenerowana macierz Hilberta sprowadzona do macierzy Hessenberga.
A.inv - wygenerowana przez nas macierz odwrotna do A

```
A=

    0.0001    0.0002         0         0         0
    0.0002    0.0079    0.0143         0         0
         0    0.0143    0.2556   -0.3302         0
         0         0   -0.3302    1.4126   -0.3222
         0         0         0   -0.3222    0.1111


A.inv =

   1.0e+05 *

    2.0623   -0.8972    0.4643    0.3205    0.9295
   -0.8972    0.4093   -0.2118   -0.1462   -0.4240
    0.4643   -0.2118    0.1100    0.0759    0.2202
    0.3205   -0.1462    0.0759    0.0524    0.1520
    0.9295   -0.4240    0.2202    0.1520    0.4410

Elapsed time is 0.015936 seconds.

B = inv(A)

Elapsed time is 0.013200 seconds.
```


Macierz różnicy wyniów odwracania macierzy
1
2
3
4
5
1
2
3
4
5
5
0
-5
-10
-15
×10-8

3.2 Test 2: Rozwiązywanie układu równań z macierzą losową

A - macierz losowa rozmiaru 5×5 wygenerowana z rozkładu normalnego a następnie przemnożona przez jej transpozycję
B - macierz jednostkowa

```
A =

    5.9432    3.0892    2.4544    3.5887    2.7655
    3.0892    8.3168    3.7029   -0.2827    5.1115
    2.4544    3.7029    2.8951    0.9726    4.0139
    3.5887   -0.2827    0.9726    3.3089    1.3957
    2.7655    5.1115    4.0139    1.3957    8.8404


B =

     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
     0     0     0     1     0
     0     0     0     0     1


cond =

  105.6583


e.dc =

   6.0908e-17


e.rel =

   4.0215e-16

wspolczynnik.stabilności =

   3.8062e-18
```

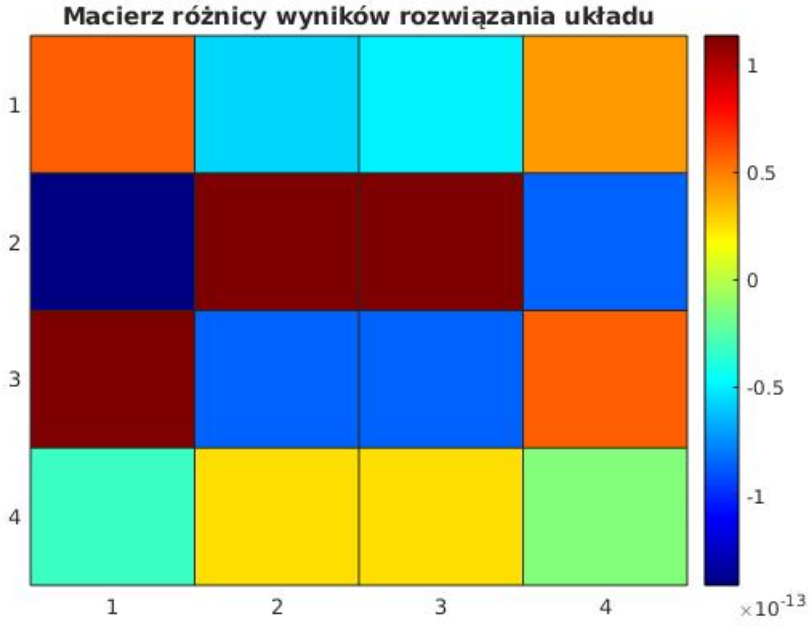
```
wspolczynnik.poprawności =

   9.9718e-16
```

3.3 Test3: Rozwiązywanie układu równań z macierzą Pascala

A - macierz Pascala 4×4
B - macierz powstała z odjęcia macierzy magicznej od macierzy Pascala
X - wynik równania $A \cdot X = B$

```
A =

     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20


B =

    15     1     2    12
     4     9     7     4
     8     4     0     2
     3    10     5   -19


X =

    65   -44   -39    51
  -113   106   101   -95
    87   -85   -84    81
   -24    24    24   -25



cond =

  691.9374


e.dc =

     0
```

```
e.rel =

   1.0749e-15


wspolczynnik.stabilnosci =

   1.5535e-18


wspolczynnik.poprawoności =

   6.7086e-13
```



4 Analiza rezultatów otrzymanych w wyniku testów

Testy w ładny i czytelny sposób prezentują dokładność wykonywanego przez nas rozkładu. Rozbicie macierzy A na iloczn macierzy $U \cdot L$ nie obniża w znaczący sposób dokładności oferowanej przez moduły Matlaba, a także daje radę wykonywać obliczenia w rozsądnie niskim czasie.

5 Kod programu

5.1 Opracowane funkcje

5.1.1 Funkcja dokonująca rozkładu UL macierzy a za pomocą zmodyfikowanej metody Crouta

Funkcja przymuje na wejściu macierz A, następnie sprawdza, czy są spełnione warunki istnienia rozkładu. Jeśli, którykolwiek z warunków nie jest spełniony, funkcja kończy działanie. W przeciwnym przypadku, zwraca macierze U oraz L.

```
function [U,L]=CroutModif(A)

n = length(A);
L = zeros(n);
U = zeros(n);
[a,b] = size(A);
assert(a==b)
assert(Minors(A) ~= 0)
L(n, n) = A(n, n);

for j = n:-1:1
    for i = n:-1:1
        if i == j
            U(i, j) = 1;
        end
        sum = 0;
        p = max(i, j) + 1;

        for k = n:-1:p
            sum = sum + (U(i, k) * L(k, j));
        end
        if i >= j
            L(i, j) = A(i, j) - sum;
        else
            U(i, j) = (1/L(j, j)) * (A(i, j) - sum);
        end
    end
end
end
```

5.1.2 Obliczanie wyznacznika macierzy A

Funkcja przyjmuje na wejściu macierz A, następnie za pomocą funkcji CroutModif wyznacza jej rozkład i na tej podstawie, wykorzystują własności wyznacznika macierzy trójkątnych oblicza wyznacznik macierzy A.

```
function[deter] = Deter(A)

[~, L] = CroutModif(A);

if norm(L-tril(L),'fro')>0
    disp('Your matrix is not lower-triangular');
    return;
end

deter = cumprod(diag(L));
deter = deter(end);
```

5.1.3 Obliczenie minorów macierzy A i weryfikacja ich znaków

Funkcja przyjmuje na wejściu macierz A i sprawdza znaki wszystkich jej minorów. Jeśli wszystkie są niezerowe zwraca 1, w przeciwnym przypadku 0.

```
function [areAllNoZeros] = Minors(A)

[n,m] = size(A);

if m~=n
    disp('m must be equal to n');
    return
end

areAllNoZeros = 1;
for i = 1:n
    if det(A(1:i, 1:i)) == 0
        areAllNoZeros = 0;
    end
end
end
```

5.1.4 Wyznaczanie macierzy odwrotnej do A na podstawie rozkładu

```
function [A_inv] = Inverse(A)
[U,L] = CroutModif(A);
[m,n] =  size(A);
b = eye(n,n);
B = zeros(n,n);
for i = 1:m
    B(1,i) = b(1,i)/L(1,1);
    for k = 2:m
        sum = 0;
            for j = k-1:-1:1
              sum = sum + L(k,j)*B(j,i);
            end
        B(k,i) = (b(k,i)- sum)/L(k,k);
    end
end
c = eye(n);
C = zeros(n,n);
for i = 1:m
      C(m,i) = c(m,i)/U(m,m);
      for k = m-1:-1:1
        sum = 0;
        for j = k+1:m
           sum = sum + U(k,j)*C(j,i);
        end
        C(k,i) = (c(k,i)- sum)/U(k,k);
      end
end
A_inv = zeros(n, n);
for i= 1 :n
  for j= 1 :n
    sum = 0;
    for k = 1 : n
      sum = sum + B(i, k) * C(k, j);
    end
    A_inv(i, j) = sum;
  end
end
end
```

5.1.5 Rozwiązywanie układu równań liniowych $\mathbf{AX = B}$, gdzie $A, B \in R^{n \times n}$

Funkcja przyjmuje na wejściu macierze A i B. Zwraca X t.że AX = B.

```
function [X] = setOfEquations(A,B)
[n, m] = size(A);
[p, r] = size(B);
if m~=n
    disp('m should be equal to n');
    return;
end
if p~=r
    disp('p should be equal to n');
    return;
end
assert(p == n)
[U, L] = CroutModif(A);
Y = zeros(n,n);
X = zeros(n,n);
for i = 1:m
      Y(m,i) = B(m,i)/U(m,m);
      for k = m-1:-1:1
        sum = 0;
        for j = k+1:m
           sum = sum + U(k,j)*Y(j,i);
        end
        Y(k,i) = (B(k,i)- sum)/U(k,k);
      end
end
for i = 1:m
    X(1,i) = Y(1,i)/L(1,1);
    for k = 2:m
        sum = 0;
            for j = k-1:-1:1
              sum = sum + L(k,j)*X(j,i);
            end
        X(k,i) = (Y(k,i)- sum)/L(k,k);
    end
end
```

5.2 Skrypt programu obliczeniowego

```
clear
clc

finish=12;
kontrol=1;
while kontrol~=finish

    kontrol=menu('MENU', 'Wprowadź macierz A',
    'Wyznacz rozkład UL macierzy A za pomocą zmodyfikowanej metody Crouta',
    'Oblicz wyznacznik macierzy A','Wyznacz macierz odwrotną do A',
    'Podaj macierz B i rozwiąż układ równań postaci AX = B',
    'Porównanie wyniku odwracania z rezultatem funkcji "inv"',
    'Analiza rezultatu odwracania macierzy',
    'Analiza wyniku dla rozwiązania układu równań', 'Test1',
    'Test2', 'Test3', 'Zako?cz');

    switch kontrol

        case 1
            A=input('Podaj macierz A ');
            [a,b] = size(A);
            if a~=b || Minors(A) == 0
                disp('Wprowadzona macierz A nie spe?nia za?o?e? rozk?adu')
                kontrol = finish;
                close all
            end

        case 2
            [U, L] = CroutModif(A)

        case 3
             detA = Det(A)
        case 4
             invA = Inverse(A)
        case 5
            B=input('Podaj macierz B ');
            [c,d] = size(B);
            if c~=d
```

```
                disp('Wprowadzona macierz B nie spe?nia za?o?e?')
                kontrol = finish;
                close all
            end
            [X] = Equation(A,B)
        case 6
            disp('Por?wnanie wyniku odwracania z rezultatem funkcji "inv"')
            disp('Rezultat otrzymany w wyniku wywo?ania na
            macierzy A naszej funkcji:')
            A1 = Inverse(A)
            disp('Rezultat otrzymany w wyniku wywo?ania na
            macierzy A funkcji "inv":')
            A2 = inv(A)
            disp('R??nica otrzymanych rezultat?w')
            Difference = A2 - A1
        case 7
            [wynik] = Error1(A)
        case 8
            [wynik] = Error2(A,B)
        case 9
             Test1
        case 10
             Test2
        case 11
             Test3
        case 12
            disp('Zako?czono')
            close all;

    end
end
```