POLITECHNIKA GDAŃSKA

WYDZIAŁ ELEKTROTECHNIKI I AUTOMATYKI
Katedra Inżynierii Systemów Sterowania

PODSTAWY AUTOMATYKI

MATLAB - komputerowe środowisko obliczeń naukowo-inżynierskich - podstawowe operacje na liczbach i macierzach.

Materiały pomocnicze do ćwiczeń laboratoryjnych - część II - termin T1

Opracowanie:
Kazimierz Duzinkiewicz, dr hab. inż.
Michał Grochowski, dr inż.
Robert Piotrowski, dr inż.
Tomasz Rutkowski, dr inż.
Rafał Łangowski, dr inż.

Gdańsk

1. Liczby w MATLAB'ie

Liczby rzeczywiste są podstawowymi danymi wykorzystywanymi w MATLAB'ie. Z liczb rzeczywistych buduje się macierze i liczby zespolone. Można stosować różne notacje dla zapisu liczb. Poniżej podano przykłady poprawnego zapisania liczb w MATLAB'ie:

```
3          3.         3.0        -3         -3.        -3.0
0.3        .3         -0.3       -.3
0.0000005578  0.5578E-6          .5578e-6
```

Liczba cyfr znaczących w liczbie jest zależna od arytmetyki komputera [patrz: część I materiału - eps].

Na każde polecenie MATLAB podaje odpowiedź, która może być wyświetlona na ekranie. Format odpowiedzi, która jest wartością numeryczną może być określony za pomocą polecenia **format** (tabela 1). Wybrane, możliwe formaty liczbowe w środowisku MATLAB'a zostały pokazane na przykładach. Zdefiniujmy wektor:

```
» a=[4/3,1.2345e-6]
a =
   1.3333    0.0000
»
```

Odpowiedź MATLAB'a została podana w formacie liczbowym **short**, w którym wyświetlane jest pięć cyfr. Format short jest formatem domyślnym MATLAB'a.

```
» format short
» a
a =
   1.3333    0.0000
»
```

Drugim dostępnym formatem liczbowym jest format **short e**. W formacie tym również wyświetlanych jest pięć cyfr. Różnice pomiędzy formatami uwidocznione są w przykładach.

```
» format short e
» a
a =
  1.3333e+000  1.2345e-006
»
```

Trzecim i czwartym formatem liczbowym są **long** i **long e**. W formatach tych wyświetlane jest 16 cyfr. Różnice pomiędzy nimi są takie same jak pomiędzy short i short e.

```
» format long
» a
a =
   1.33333333333333   0.00000123450000

» format long e
» a
a =
   1.333333333333333e+000    1.234500000000000e-006
»
```

W formatach long ostatnie wyświetlane cyfry mogą być niepoprawne [patrz: część I materiału - zmienna eps], chociaż wszystkie wyświetlane cyfry są dokładną dziesiętną reprezentacją liczby binarnej przechowywanej w komputerze.

Ciekawym formatem jest **format +**. Użycie tego formatu powoduje, że wyświetlany jest tylko znak liczby lub spacja, jeżeli liczba jest zerem. Przykłady:

```
» a=[1,-2,3,-4,5,-6;7,-8,9,-10,11,-12;13,-14,15,-16,17,-18];
» format long e
» a
a =
   1    -2     3    -4     5    -6
   7    -8     9   -10    11   -12
  13   -14    15   -16    17   -18
» format +
» a
a =
+-+-+-
+-+-+-
+-+-+-
»
```

lub:

```
» a=[1,-2,0,-4,5,0;7,0,9,-10,11,-12;13,-14,15,-16,17,-18];
» a
a =
+- -+
+ +-+-
+-+-+-
»
```

Tabela 1. Format liczb w MATLAB'ie

Format	Reprezentacja
short	0.6667; -6.0221e-07
short e	6.6667e-001; -6.0221e-007
long	0.66666666666667; -6.022100000000000e-007
long e	6.666666666666666e-001; -6.022100000000000e-007
hex	3fe5555555555555
+	znak drukowany dla liczb dodatnich
bank	format walutowy, do dwóch miejsc po przecinku
compact	wyłącza dodawanie dodatkowych pustych wierszy
loose	włącza dodawanie dodatkowych pustych wierszy
rat	przybliża liczby ułamkami małych liczb całkowitych

Poza skończonymi wartościami liczbowymi w MATLAB'ie wykorzystywane są dwie inne wartości: **Inf** i **NaN**. Wartość **Inf** oznacza nieskończoność liczbową; można ją uzyskać na przykład poleceniem:

```
» 1/0
Warning: Divide by zero
ans =
  Inf
```

Druga z podanych wartości, **NaN** oznacza nieokreśloność liczbową; można ją uzyskać na przykład poleceniem:

```
» 0/0
Warning: Divide by zero
ans =
  NaN
```

lub:

```
» Inf/Inf
ans =
  NaN
»
```

Podane dwie nazwy można traktować jako posiadające podwójne znaczenie: jako nazwy wartości lub nazwy funkcji.

W MATLAB'ie we wszystkich działaniach i funkcjach mogą występować liczby zespolone. Liczby zespolone są wprowadzane przez specjalne funkcje: **i** lub **j**. Wszystkie podane niżej zapisy są poprawne i definiujące taką samą liczbę zespoloną, co potwierdzają odpowiedzi MATLAB'a:

Zapis 1:

```
» z1=4+3i
z1 =
  4.0     .0000i
»
```

Zapis 2:

```
» z1=4+3j
z1 =
  4.0     .0000i
»
```

Zapis 3:

```
» z1=4+3*i
z1 =
  4.0     .0000i
»
```

Zapis 4:

```
» z1=4+i*3
z1 =
  4.0     .0000i
»
```

Zapis 5:

```
» z1=4+3*sqrt(-1)
```

```
z1 =
  4.0000 + 3.0000i
»
```

2. Definiowanie macierzy

MATLAB pracuje zasadniczo z jednym rodzajem obiektów, **prostokątnymi macierzami numerycznymi**. W taki sposób traktowane są skalary - macierze o rozmiarze 1x1 oraz wektory - macierze o rozmiarze nx1. Macierz w środowisku MATLAB'a można zdefiniować na kilka sposobów:

- przez wymienienie elementów macierzy,
- przez wygenerowanie elementów macierzy,
- przez zbudowanie z innych macierzy,
- przez mieszane zastosowanie wymienionych wyżej sposobów.

MATLAB nie posiada poleceń deklarujących rozmiar macierzy lub jej typ. Utworzona macierz jest pamiętana od momentu utworzenia aż do momentu jej usunięcia. Możliwa jest zmiana nie tylko wartości macierzy, ale także jej rozmiaru.

2.1 Definiowanie macierzy - Sposób I

Najprostszą metodą utworzenia macierzy jest bezpośrednie wypisanie listy jej elementów. Jeżeli chcemy utworzyć macierz o nazwie A i rozmiarze 3x3 to przez wypisanie listy jej elementów możemy to zrobić następująco:

```
A=[1,2,3;4,5,6;7,8,9]    lub    A=[1 2 3;4 5 6;7 8 9]
```

Listę elementów macierzy należy ująć w nawiasy kwadratowe; elementy wprowadzane są wierszami; poszczególne elementy wiersza należy rozdzielić przecinkami lub spacjami; elementy poszczególnych wierszy należy rozdzielić średnikami.
Jeżeli za zamykającym nawiasem kwadratowym nie umieścimy średnika to otrzymamy odpowiedź. Odpowiedni fragment okna poleceń będzie wyglądał następująco:

```
» A=[1,2,3;4,5,6;7,8,9]
A =
   1   2   3
   4   5   6
   7   8   9
»
```

lub

```
» A=[1 2 3;4 5 6;7 8 9]
A =
   1   2   3
   4   5   6
   7   8   9
»
```

Jeżeli wymienimy elementy macierzy umieszczając za zamykającym nawiasem średnik, odpowiedni fragment okna poleceń będzie wyglądał następująco:

```
» A=[1,2,3;4,5,6;7,8,9];    lub    » A=[1 2 3;4 5 6;7 8 9];
```

Możemy wprowadzić macierz nie nadając jej nazwy. Elementy wprowadzonej macierzy zostaną przypisane standardowej zmiennej roboczej **ans**. Przykładowy, odpowiedni fragment okna poleceń będzie wyglądał następująco:

```
» [1 2 3;4 5 6;7 8 9]
ans =
   1   2   3
   4   5   6
   7   8   9
»
```

Jeżeli nie możemy (lub nie chcemy) umieścić wszystkich elementów danego wiersza macierzy w jednej linii poleceń to możemy daną linię zakończyć wielokropkiem; elementy podawane w następnej linii będą traktowane jako kontynuacja wiersza z poprzedniej linii. Weźmy na przykład macierz o wymiarze 30x3:

```
» B=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,...
   21,22,23,24,25,26,27,28,29,30;...
   31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,...
   51,52,53,54,55,56,57,58,59,60;...
   61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,...
   81,82,83,84,85,86,87,88,89,90]
B =
 Columns 1 through 12
    1    2    3    4    5    6    7    8    9   10   11   12
   31   32   33   34   35   36   37   38   39   40   41   42
   61   62   63   64   65   66   67   68   69   70   71   72
 Columns 13 through 24
   13   14   15   16   17   18   19   20   21   22   23   24
   43   44   45   46   47   48   49   50   51   52   53   54
   73   74   75   76   77   78   79   80   81   82   83   84
 Columns 25 through 30
   25   26   27   28   29   30
   55   56   57   58   59   60
   85   86   87   88   89   90
```

Wprowadzając macierz można skorzystać z WINDOWS'owej funkcji **edit**. Jej wywołanie z nazwą macierzy otwiera okno edycyjne w którym możemy określić wstępnie rozmiary macierzy, a następnie wprowadzić je. Zostanie ona przekazana do MATLAB'a z nazwą jaka była użyta przy funkcji **edit**. Jeżeli nie użyto nazwy macierzy, wprowadzona macierz zostanie przekazana do MATLAB'a z nazwą **untitled**. Wygląd okna poleceń, przy wprowadzaniu macierzy 4x4 funkcją **edit** jest postaci:

```
» edit C
» C
C =
    1    2    3    4
    5    6    7    8
    9   10   11   12
   13   14   15   16
»
```

Przy wprowadzaniu macierzy użycie klawisza **Enter** jest równoważne postawieniu znaku średnika. Wprowadzenie macierzy A można też przeprowadzić następująco:

```
» A=[1,2,3
      4,5,6
      7,8,9]
A =
   1   2   3
   4   5   6
   7   8   9
»
```

Wprowadzane elementy macierzy nie muszą mieć wprost wartości liczbowych, mogą być wartościami wyrażeń. Przykładowo:

```
» x=[-1.3,sqrt(3),(1+2+3)*4/5]
x =
  -1.3000    1.7321    4.8000
»
```

oraz

```
» s1=-1.3;
» s2=3;
» s3=1;
» s4=2;
» s5=3;
» s6=4;
» s7=5;
» x=[s1,sqrt(s2),(s3+s4+s5)*s6/s7]
x =
  -1.3000    1.7321    4.8000
»
```

2.2 Definiowanie macierzy - Sposób II

Drugim sposobem wprowadzenia macierzy jest wygenerowanie jej elementów. Ten sposób można zastosować jeżeli elementy macierzy tworzą ciąg arytmetyczny. W tym celu należy użyć polecenia w postaci:

x = min : krok : max

gdzie:
min - pierwszy element ciągu, krok - różnica ciągu, max - ostatni element ciągu.

Przykład 1:

```
» x=1:0.2:2
x =
   1.0000    1.2000    1.4000    1.6000    1.8000    2.0000
»
```

Przykład 2:

```
» x=[1:0.2:2]
x =
   1.0000   1.2000   1.4000   1.6000   1.8000   2.0000
»
```

Przykład 3:

```
» B=[1:0.2:2;3:0.2:4;5:0.2:6]
B =
   1.0000   1.2000   1.4000   1.6000   1.8000   2.0000
   3.0000   3.2000   3.4000   3.6000   3.8000   4.0000
   5.0000   5.2000   5.4000   5.6000   5.8000   6.0000
»
```

Dla generowania macierzy z elementami tworzonymi inaczej niż przedstawiono wyżej można posłużyć się funkcjami wspomagającymi generowanie macierzy, które zostały zestawione w tabeli 2.

Tabela 2. Funkcje wspomagające generowanie macierzy

Funkcje	Opis
Podstawowe macierze	
eye(n)	tworzenie kwadratowej macierzy jednostkowej nxn
eye(n, m)	tworzenie prostokątnej macierzy jednostkowej nxm
ones(n)	tworzenie kwadratowej macierzy nxn o elementach równych 1
ones(n, m)	tworzenie prostokątnej macierzy nxm o elementach równych 1
zeros(n)	tworzenie kwadratowej macierzy zerowej nxn
zeros(n, m)	tworzenie prostokątnej macierzy zerowej nxm
Liczby pseudolosowe	
rand(n)	tworzenie kwadratowej macierzy nxn z liczbami pseudolosowymi o rozkładzie jednostajnym na przedziale <0,1>
rand(n, m)	tworzenie prostokątnej macierzy nxm z liczbami pseudolosowymi o rozkładzie jednostajnym na przedziale <0,1>
randn(n)	tworzenie kwadratowej macierzy nxn z liczbami pseudolosowymi o rozkładzie normalnym o wartości średniej 0 i wariancji 1
randn(n, m)	tworzenie prostokątnej macierzy nxm z liczbami pseudolosowymi o rozkładzie normalnym o wartości średniej 0 i wariancji 1
Ciągi liczb	
linspace(x_1, x_2)	generowanie wektora wierszowego 100 liczb rozmieszczonych równomiernie między wartościami x_1 i x_2
linspace(x_1, x_2, n)	generowanie wektora wierszowego n liczb rozmieszczonych równomiernie między wartościami x_1 i x_2
logspace(x_1, x_2)	generowanie wektora wierszowego 50 liczb logarytmicznie równo rozmieszczonych między wartościami x_1 i x_2
logspace(x_1, x_2, N)	generowanie wektora wierszowego n liczb logarytmicznie równo rozmieszczonych między wartościami x_1 i x_2

2.3 Definiowanie macierzy - Sposób III

Trzecim sposobem wprowadzania macierzy jest jej utworzenie z innych macierzy. Rozpoczniemy od przypadku, kiedy macierz tworzona jest z innych macierzy, które mają być jej fragmentami. Pierwsze przykłady dotyczą wektorów. Jeżeli utworzyliśmy uprzednio wektor a o wymiarze 1x3:

```
» a=[-1.3,sqrt(3),log10(2)]
a =
  -1.3000    1.7321    0.3010
»
```

Wektor ten możemy rozszerzyć wprowadzając elementy o indeksie większym niż 3. Elementom pośrednim zostanie przypisana wartość zero.

```
» a(6)=abs(a(1))
a =
  -1.3000    1.7321    0.3010         0         0    1.3000
»
```

Korzystając z wektora a możemy utworzyć nowy wektor, którego elementami będą elementy wektora a.

```
» b=[a,7,8,9]
b =
 Columns 1 through 7
  -1.3000    1.7321    0.3010         0         0    1.3000    7.0000
 Columns 8 through 9
   8.0000    9.0000
»
```

lub

```
» c=[1,2,3,a]
c =
 Columns 1 through 7
   1.0000    2.0000    3.0000   -1.3000    1.7321    0.3010         0
 Columns 8 through 9
       0    1.3000
```

Możemy utworzywszy uprzednio wektory, zdefiniować macierz:

```
» A=[b;c;31,32,33,34,35,36,37,38,39]
A =
 Columns 1 through 7
   -1.3000    1.7321    0.3010         0         0    1.3000    7.0000
    1.0000    2.0000    3.0000   -1.3000    1.7321    0.3010         0
   31.0000   32.0000   33.0000   34.0000   35.0000   36.0000   37.0000
Columns 8 through 9
    8.0000    9.0000
         0    1.3000
   38.0000   39.0000
»
```

Wykorzystując wcześniej zdefiniowane macierze dwuwymiarowe naturalnie można tworzyć inne macierze; należy tylko przestrzegać zgodności liczby kolumn w poszczególnych wierszach tworzonej macierzy.

```
» A=[1,4,1;2,0,1];
» B=[3,1;4,1];
» C=[1,2,2,0,1;2,4,7,1,0];
» W=[A,B;C]
W =
   1   4   1   3   1
   2   0   1   4   1
   1   2   2   0   1
   2   4   7   1   0
```

Użytecznym znakiem przy tworzeniu macierzy jest dwukropek. Może on być wykorzystywany przy:

- generowaniu wektorów o równomiernie rozłożonych elementach,
- wybieraniu pożądanych wierszy, kolumn lub elementów macierzy.

Generowanie wektorów, z zastosowaniem notacji dwukropkowej, odbywa się według następujących zasad:

- zapis j:k definiuje wektor [j,j+1,…,k]; jest to wektor, którego pierwszy wyraz wynosi j, ostatni jest nie większy od k, przyrost wartości kolejnych elementów wynosi 1,
- zapis j:n:k definiuje wektor [j,j+n,…,k]; jest to wektor, którego pierwszy wyraz wynosi j, ostatni jest nie większy od k, przyrost kolejnych elementów wynosi n.

Wybór pożądanych wierszy, kolumn i elementów macierzy można dokonywać za pomocą następujących zapisów:

- A(:,j) - wypisanie j-tej kolumny macierzy A,
- A(:,j:k) - wypisanie kolumn A(j), A(j+1),…,A(k) macierzy A,
- A(i,:) - wypisanie i -tego wiersza macierzy A,
- A(:) - wypisanie wszystkich elementów macierzy A w jednej kolumnie; elementom zostają przyporządkowane kolejne indeksy,
- A(j:k) - wypisanie w jednym wierszu, elementów macierzy A od elementu o indeksie j do elementu o indeksie k.

Przykład:

```
» a=[1,2,3,-4,-5,-6;-7,-8,-9,10,11,12;13,14,15,16,17,18]
a =
    1    2    3   -4   -5   -6
   -7   -8   -9   10   11   12
   13   14   15   16   17   18
»

» b=a(:,3)
b =
    3
   -9
   15

» c=a(2,:)
c =
   -7   -8   -9   10   11   12
»
```

```
» d=a(1:6)
d =
    1   -7   13    2   -8   14
»

» e=a(:)
e =
    1
   -7
   13
    2
   -8
   14
    3
   -9
   15
   -4
   10
   16
   -5
   11
   17
   -6
   12
   18
»

» f=a(2:3,:)
f =
   -7   -8   -9   10   11   12
   13   14   15   16   17   18
»

» g=a(:,4:-1:2)
g =
   -4    3    2
   10   -9   -8
   16   15   14
»
```

3. Wymiar macierzy i jej wyświetlanie

Wymiar wprowadzonej macierzy można sprawdzić wykorzystując polecenie **size**, które zwraca liczbę wierszy i kolumn macierzy. Przykładowo dla powyższej macierzy g:

```
» [w,k]=size(g)
w =
    3
k =
    3
»
```

Jeżeli chcemy uzyskać tylko liczbę wierszy lub kolumn należy użyć polecenia **size** w następujący sposób:

```
» w=size(g,1)
w =
    3
»
```

lub

```
» k=size(g,2)
k =
    3
»
```

Wyświetlenie wprowadzonej macierzy można uzyskać używając polecenia **disp** lub podając nazwę macierzy i naciskając **Enter**:

```
» disp(A)
    1    2    3
    4    5    6
    7    8    9
»

» A
A =
    1    2    3
    4    5    6
    7    8    9
»
```

Jeżeli chcemy otrzymać wymiar wektora A możemy użyć polecenia **length**. W przypadku, gdy A jest macierzą zwracany jest dłuższy z jej wymiarów. Przykładowo dla wprowadzonej macierzy A:

```
» n=length(A)
n =
    3
»
```

4. Działania arytmetyczne na macierzach

W MATLAB'ie dostępne są następujące działania na macierzach:

+ - dodawanie,
- - odejmowanie,
***** - mnożenie,
/ - prawostronne dzielenie macierzy,
**** - lewostronne dzielenie macierzy,
^ - potęgowanie.

Dodawanie i odejmowanie macierzy jest wykonalne, jeżeli każda z macierzy w tych działaniach **ma taką samą liczbę kolumn i wierszy**.

```
» B=[1:1:6;7:1:12;13:1:18]
```

```
B =
    1    2    3    4    5    6
    7    8    9   10   11   12
   13   14   15   16   17   18
»

» C=[18:-1:13;12:-1:7;6:-1:1]
C =
   18   17   16   15   14   13
   12   11   10    9    8    7
    6    5    4    3    2    1
»

» D=B+C
D =
   19   19   19   19   19   19
   19   19   19   19   19   19
   19   19   19   19   19   19
»

» E=B-C
E =
  -17  -15  -13  -11   -9   -7
   -5   -3   -1    1    3    5
    7    9   11   13   15   17
»
```

Korzystając z operacji dodawania i odejmowania można też do/od każdego elementu macierzy dodać/odjąć liczbę.

```
» F=B+2
F =
    3    4    5    6    7    8
    9   10   11   12   13   14
   15   16   17   18   19   20
»

» G=C-4
G =
   14   13   12   11   10    9
    8    7    6    5    4    3
    2    1    0   -1   -2   -3
»
```

Mnożenie macierzy jest możliwe, **jeżeli liczba kolumn macierzy mnożnej jest równa liczbie wierszy macierzy mnożnika**.

```
» H=G(1:2,1:3)
H =
   14   13   12
    8    7    6
»
```

```
» K=G(:,4:6)
K =
   11   10    9
    5    4    3
   -1   -2   -3
»

» M=H*K
M =
  207  168  129
  117   96   75
»
```

Działanie mnożenia można oczywiście wykonywać na wektorach pod warunkiem, że jeden z nich będzie wektorem wierszowym a drugi kolumnowym. W takich przypadkach bardzo często uprzednio wykonywana jest operacja transpozycji. Operacja transpozycji w MATLAB'ie oznaczana jest znakiem ` (pojedynczy apostrof).

```
» x=[1:6]
x =
   1   2   3   4   5   6
»

» y=[6:-1:1]
y =
   6   5   4   3   2   1
»

» s=x*y'
s =
   56
»
```

Taki sam wynik otrzymamy jeżeli postąpimy jak poniżej:

```
» s1=y*x'
s1 =
   56
»
```

Powyższe przykłady ilustrują obliczanie iloczynu skalarnego. Korzystając z działania mnożenia macierzy możemy również obliczyć iloczyn wektorowy.

```
» x=[-1,0,2]'
x =
   -1
    0
    2
»
```

```
» y=x-1
y =
   -2
   -1
    1
»

» w1=x*y'
w1 =
    2    1   -1
    0    0    0
   -4   -2    2
»

» w2=y*x'
w2 =
    2    0   -4
    1    0   -2
   -1    0    2
»

» w3=w1'
w3 =
    2    0   -4
    1    0   -2
   -1    0    2
»
```

Działanie mnożenia można użyć do mnożenia każdego elementu macierzy przez liczbę:

```
» sM1=0.5*w3
sM1 =
   1.0000         0   -2.0000
   0.5000         0   -1.0000
  -0.5000         0    1.0000
»

» sM2=w3*0.5
sM2 =
   1.0000         0   -2.0000
   0.5000         0   -1.0000
  -0.5000         0    1.0000
»
```

W MATLAB'ie są dostępne dwa działania dzielenia macierzy: lewostronne **** i prawostronne **/**. Jeżeli macierz A jest **nieosobliwą macierzą kwadratową**, wówczas A\B i B/A odpowiadają formalnie lewostronnemu i prawostronnemu mnożeniu macierzy B przez macierz odwrotną A, to znaczy **inv(A)*B** i **B*inv(A)**, chociaż w trakcie obliczeń w MATLAB'ie macierz odwrotna nie jest wprost obliczana. Ogólnie działania te zdefiniowane są następująco:

X = A\B jest rozwiązaniem równania A*X = B
X = B/A jest rozwiązaniem równania X*A = B

Z powyższych definicji widać, że lewostronne i prawostronne dzielenie można wykorzystać do rozwiązywania liniowych równań algebraicznych. Dzielenie lewostronne A\B jest określone, kiedy B ma taką samą liczbę wierszy jak A. Dzielenie prawostronne B/A jest określone, kiedy A ma taką sama liczbę kolumn jak B.

Potęgowanie macierzy **A^p** jest określone, jeżeli macierz jest kwadratowa i potęga jest skalarem. Jeżeli p jest liczbą całkowitą potęga macierzy jest obliczana przez jej p-krotne mnożenie. W innych przypadkach do obliczeń wykorzystywane są **wartości własne** i **wektory własne** macierzy.

5. Działania tablicowe na macierzach

W MATLAB'ie poza powszechnie znanymi działaniami arytmetycznymi na macierzach określone są tak zwane działania tablicowe na macierzach. Są to działania odnoszące się do działań arytmetycznych **element-po-elemencie** macierzy. Dostępne są następujące działania tablicowe:

+ - dodawanie,
- - odejmowanie,
.* - mnożenie,
./ - prawostronne dzielenie macierzy,
. - lewostronne dzielenie macierzy,
.^ - potęgowanie.

Działania dodawania i odejmowania są działaniami tablicowymi, bo operacje te wykonywane są element po elemencie. Działanie mnożenia tablicowego polega na mnożeniu dwóch macierzy o takich samych wymiarach, w których mnożone są odpowiadające sobie elementy:

```
» A=[1 3;2 2]
A =
  1  3
  2  2
»

» B=[2 2;1 3]
B =
  2  2
  1  3
»

» C=A.*B
C =
  2  6
  2  6
»
```

Działanie **./** powoduje dzielenie dwóch macierzy element po elemencie. Działanie **.** powoduje dzielenie elementów macierzy będącej drugim jej argumentem przez macierz stanowiącą pierwszy jej argument.

Potęgowanie tablicowe **.^** jest działaniem analogicznym do potęgowania arytmetycznego i odbywa się z wykorzystaniem poszczególnych elementów macierzy.

6. Inne wybrane operacje macierzowe

Innymi często wykorzystywanymi operacjami macierzowymi są te związane z obliczaniem **wyznacznika** macierzy, **macierzy transponowanej** i **odwrotnej**, **rzędu** macierzy, **wartości własnych** i **wektorów własnych**, **normy** wektorów i macierzy, **współczynników** i **pierwiastków równania charakterystycznego**. Do wykonania powyższych operacji służą następujące polecenia:

- **det(A)** - obliczanie wyznacznika macierzy kwadratowej A
- **trace(A)** - wyznaczanie śladu macierzy A
- **A'** - obliczanie macierzy transponowanej do macierzy A
- **inv(A)** - wyznaczanie macierzy odwrotnej do macierzy A
- **rank(A)** - obliczanie rzędu macierzy A
- **eig(A)** - wyznaczanie wartości własnych macierzy kwadratowej A (wyznaczanie pierwiastków równania charakterystycznego **det(s*I-A = 0)**).
- **[V,D]=eig(A)** - obliczenie macierzy D zawierającej na przekątnej wartości własne macierzy A oraz macierz V wektorów własnych odpowiadających tym wartościom
- **norm** - wyznaczenie norm wektorów i macierzy (tabela 3)
- **poly(A)** - jeżeli macierz kwadratowa A jest stopnia n, to budowany jest n+1 elementowy wektor współczynników równania charakterystycznego **det(s*I-A)**. Współczynniki wyświetlają się od najwyższej do najniższej potęgi s.
- **roots(A)** - obliczanie pierwiastków równania charakterystycznego:

$$\mathbf{a_n \cdot x^n + a_{n-1} \cdot x^{n-1} + ... + a_1 \cdot x + a_0 = 0}$$

którego współczynniki a_n, a_{n-1}, ...a_1, a_0 są składowymi wektora A, np.:

roots([a_1 a_2 a_3 a_4])

Tabela 3. Sposoby wywoływania funkcji **norm**

Funkcja norm()	Opis
norm(A,1)	$\|\mathbf{A}\|_1 = \max_{i=1,\dots,m}\{\|a_{1i}\| + \|a_{2i} + \cdots + \|a_{mi}\|\|\}$
norm(A); norm(A,2)	$\|\mathbf{A}\|_2$ największa wartość osobliwa macierzy A, tzn. największa wartość własna macierzy $\mathbf{A^T A}$
norm(A,inf); norm(A,'inf')	$\|\mathbf{A}\|_\infty = \max_{i,j}\|a_{ij}\|$
norm(A,'fro')	$\|\mathbf{A}\|_f = \sqrt{tr(\mathbf{A^T A})}$ tzw. Norma Frobeniusa
norm(v,p)	$\|\mathbf{V}\|_p = (\|V_1\|^p + \|V_2\|^p + \cdots + \|V_n\|^p)^{\frac{1}{p}}$ - działa tylko dla parametru V będącego wektorem, p może być dowolną liczbą rzeczywistą, zazwyczaj stosuje się wartości ze zbioru liczb naturalnych

7. Funkcje MATLAB'a

Możliwości obliczeniowe MATLAB'a wynikają w dużym stopniu z bogatego zestawu funkcji dostępnych w jego środowisku. Pewne z tych funkcji są funkcjami wewnętrznymi (wbudowanymi) MATLAB'a, zaś inne są dostępne w bibliotekach tak zwanych zewnętrznych M-plików rozprowadzanych razem z MATLAB'em (*MATLAB toolbox*). Biblioteki zewnętrzne MATLAB'a mogą być tworzone przez poszczególnych użytkowników, bądź grupy użytkowników dla swoich specjalistycznych zastosowań. Dla użytkownika jest nierozróżnialne, czy funkcja z której korzysta jest funkcją wewnętrzną czy zewnętrzną.

W materiale nie zostały omówione poszczególne funkcje dostępne w MATLAB'ie. Informacje o danej funkcji można uzyskać korzystając z polecenia **help *nazwa_funkcji***. Poniżej podano ogólne informacje o funkcjach w środowisku MATLAB'a.

Podstawowe funkcje matematyczne dostępne w MATLAB'ie to:

- funkcje trygonometryczne, cyklometryczne, hiperboliczne i odwrotne do nich (np.: **sin(A), acos(A), tanh(A), asinh(A)**)
- funkcje logarytmiczne, wykładnicze i potęgowe (np.: **sqrt(A), exp(A), log(A)**)
- funkcje związane z obliczeniami w dziedzinie liczb zespolonych (tabela 4)
- funkcje zaokrągleń, reszty z dzielenia i znaku (np.: **ceil(A), fix(A), sign(A), rem(A)**)
- funkcje zmiany układu współrzędnych (np.: **cart2sph(X,Y,Z), cart2pol(X,Y,Z), pol2cart(TH,R)**)

Tabela 4. Funkcje związane z obliczeniami w dziedzinie liczb zespolonych

Funkcja	Opis
abs(A)	macierz modułów elementów macierzy A
angle(A)	macierz argumentów elementów macierzy A
real(A)	macierz części rzeczywistych elementów macierzy A
imag(A)	macierz części urojonych elementów macierzy A
conj(A)	macierz o elementach sprzężonych z odpowiednimi elementami macierzy A

Jak wspomniano wcześniej, większość funkcji w MATLAB'ie ma charakter funkcji tablicowych, to znaczy funkcji działających na poszczególnych elementach macierzy. Jeżeli napiszemy polecenie: **exp(A)** to nastąpi obliczenie dla każdego elementu macierzy A wartości wyrażenia $e^{a_{ij}}$.

```
» A=[1:3;4:6]
A =
   1   2   3
   4   5   6
»

» B=exp(A)
B =
   2.7183    7.3891   20.0855
  54.5982  148.4132  403.4288
»
```

Pewne funkcje zwracają dwie lub więcej wielkości. Na przykład funkcja obliczająca wartości i wektory własne macierzy - **eig(A)**.

```
» A=[3,2,1;4,5,5;9,8,7]
A =
    3    2    1
    4    5    5
    9    8    7
»

» [V,D]=eig(A)
V =
   0.1719          -0.3782 + 0.3044i       -0.3782 - 0.3044i
   0.5343           0.4166 - 0.6616i        0.4166 + 0.6616i
   0.8277          -0.0155 + 0.3908i       -0.0155 - 0.3908i
D =
  14.0328                  0                       0
       0            0.4836 + 0.4401i                0
       0                   0                0.4836 - 0.4401i
»
```

W MATLAB'ie istnieją również funkcje macierzowe, których nazwy wyróżnione są dodaniem literki m, np. - **expm**.

```
» E=expm(A)
E =
  1.0e+005 *
   1.5909   1.4921   1.2870
   4.9452   4.6383   4.0008
   7.6612   7.1856   6.1980
»
```

8. Operatory relacji

W MATLAB'ie istnieje sześć operatorów relacji dla porównywania macierzy o tych samych wymiarach. Są one następujące:

< - mniejsza niż
<= - mniejsza niż lub równa (nie większa)
> - większa niż
>= - większa niż lub równa (nie mniejsza)
= = - równa
~= - nie równa

Porównanie jest wykonywane pomiędzy parą odpowiadających sobie elementów; wynik jest macierzą zer i jedynek, gdzie jedynki reprezentują **prawdę**, a zera **fałsz**.

```
» a=2+2~=4
a =
    0
»
```

lub:

```
» A1=[1:3;6:-1:4;7:9]
```

```
A1 =
   1   2   3
   6   5   4
   7   8   9
»

» A2=[1:3;4:6;7:9]
A2 =
   1   2   3
   4   5   6
   7   8   9
»

» R=A1==A2
R =
   1   1   1
   0   1   0
   1   1   1
»
```

9. Działania logiczne w MATLAB'ie

W MATLAB'ie istnieją trzy operatory logiczne działające na poszczególnych elementach macierzy, zwykle zero-jedynkowych. Są one następujące:

&	**AND**
|	**OR**
~	**NOT**

Operatory **&** oraz **|**porównuje dwa skalary lub macierze o tych samych wymiarach. Operatory logiczne traktują jakąkolwiek wartość niezerową jako **prawdę**, a zerową jako **fałsz**. Zwracają one jedynkę, jeżeli wartość wyrażenia logicznego jest **prawdą**, a zero jeżeli **fałszem**.
Operator **~** jest operatorem jednoargumentowym. Wyrażenie **~A** zwraca jedynkę, jeżeli element A jest zerowy a zero, jeżeli element A jest niezerowy.

```
» A=[0,1,2;-1,0,-2;1,2,0]
A =
   0   1   2
  -1   0  -2
   1   2   0
»

» B=[1:3;4:6;7:9]
B =
   1   2   3
   4   5   6
   7   8   9
»
```

```
» NOTA=~A
NOTA =
   1   0   0
   0   1   0
   0   0   1
»

» ILOAB=A&B
ILOAB =
   0   1   1
   1   0   1
   1   1   0
»

» SUMAB=A|B
SUMAB =
   1   1   1
   1   1   1
   1   1   1
»

» ILOANA=A&(~A)
ILOANA =
   0   0   0
   0   0   0
   0   0   0
»

» SUMANA=A|(~A)
SUMANA =
   1   1   1
   1   1   1
   1   1   1
»
```

Z operatorami relacji i działaniami logicznymi związane są pewne funkcje MATLAB'a. Wyczerpujące informacje o tych funkcjach można znaleźć korzystając z polecenia **help** lub dokumentacji MATLAB'a.

10. Bibliografia

Brzózka J. *Ćwiczenia z automatyki w Matlabie i Simulinku.* Wydawnictwo MIKOM, 1997.
Brzózka J., Dorobczyński L. *Matlab - środowisko obliczeń naukowo - technicznych.* Wydawnictwo MIKOM, 2005.
Mrozek B., Mrozek Z. *Matlab i Simulink. Poradnik użytkownika. Wydanie II.* Wydawnictwo HELION, 2004.
Pratap R. *Matlab 7 dla naukowców i inżynierów.* Wydawnictwo HELION, 2010.
Sradomski W. *Matlab. Praktyczny poradnik modelowania.* Wydawnictwo HELION, 2015.
Zalewski A., Cegieła R. *Matlab - obliczenia numeryczne i ich zastosowania.* Wydawnictwo NAKOM, 1996.