Instrukcje laboratoryjne

Instrukcje laboratoryjne do Programowania w MATLAB-ie

Instrukcja 1: Wprowadzenie do MATLAB-a

Autor: Aleksander Karolczuk
Politechnika Opolska
a.karolczuk@po.edu.pl
www.a.karolczuk.po.edu.pl

Opole, 26 marca 2010

Spis treści

1. Czym jest MATLAB?

Istnieje wiele definicji programu MATLAB. Jedna z najprostszych to: MATLAB jest to dużej wydajności język do obliczeń technicznych [1]. Inna definicja, wyrażająca możliwości MATLAB-a jest następująca: MATLAB jest jednym z najszybszych i dających wiele satysfakcji systemów do numerycznego rozwiązywania zagadnień [2]. Przy użyciu MATLAB-a wiele zadań obliczeniowych staje się dużo prostszymi niż przy próbie rozwiązania tych zadań przy zastosowaniu innych systemów oprogramowania jak: Fortran, C, Java, itp. Jest to możliwe, ponieważ MATLAB jest dedykowanym narzędziem do rozwiązywania konkretnych (technicznych) zagadnień bez konieczności spędzania dużej ilości czasu na tworzenie oprogramowania podstawowego do wizualizacji, operacji macierzowych, obliczeń statystycznych, itp. MATLAB zawiera dużą ilość narzędzi do operacji wektorowych, macierzowych, które ułatwiają i przyspieszają implementację nowych algorytmów. Wizualizacja danych wielowymiarowych nie jest już problemem. MATLAB umożliwia animowanie danych i zapis ich w formacie wideo (*.avi). Dla osób korzystających z popularnego arkusza kalkulacyjnego EXCEL możliwe jest czytanie danych zapisanych w formacie *xls bezpośrednio do pamięci MATLAB-a. MATLAB w standardowym pakiecie umożliwia czytanie danych z karty dźwiękowej komputera oraz wysyłanie danych na kartę dźwiękową, co w pewnych przypadkach można wykorzystać w prostych systemach sterowania [3]. Poza szczególnymi wymienionymi przypadkami MATLAB oferuje bardzo szeroki zestaw profesjonalnych narzędzi, który z roku na rok jest systematycznie powiększany.

Typowe zastosowania MATLAB-a:

— matematyka, obliczenia,
— tworzenie i rozwój algorytmów,
— akwizycja danych,
— modelowanie, symulacje,
— analiza i wizualizacja sygnałów,
— analiza obrazów wideo,
— tworzenie aplikacji do rozwiązywania typowych zagadnień.

Pierwsza wersja MATLAB-a została utworzona przez Cleve'a Molera w latach 70-tych [4]. Od tego czasu rozwinął się osiągając poziom profesjonalnego oprogramowania sprzedawanego przez firmę The MathWorks [5]. Nazwa MATLAB zastała zaczerpnięta początkowych liter sformułowania: MATrix LABoratory. MATLAB jest stale rozwijanych oprogramowaniem (jedna lub dwie aktualizacje w ciągu jednego roku). MATLAB składa się z wielu dedykowanych kolekcji funkcji (miedzy innymi, tzw. M-pliki, i inne). Kolekcja funkcji poświęconych jednemu zagadnieniu nosi nazwę toolbox (zestaw narzędzi). Najpopularniejsze obszary zagadnień, dla których toolbox-y są osiągalne to: przetwarzanie sygnałów (signal processing), układy sterowania (control systems), sieci neuronowe (neural networks), analiza statystyczna (statistical analysis), analiza obrazów wideo, (image analysis) i wiele innych (patrz: www.mathworks.com).

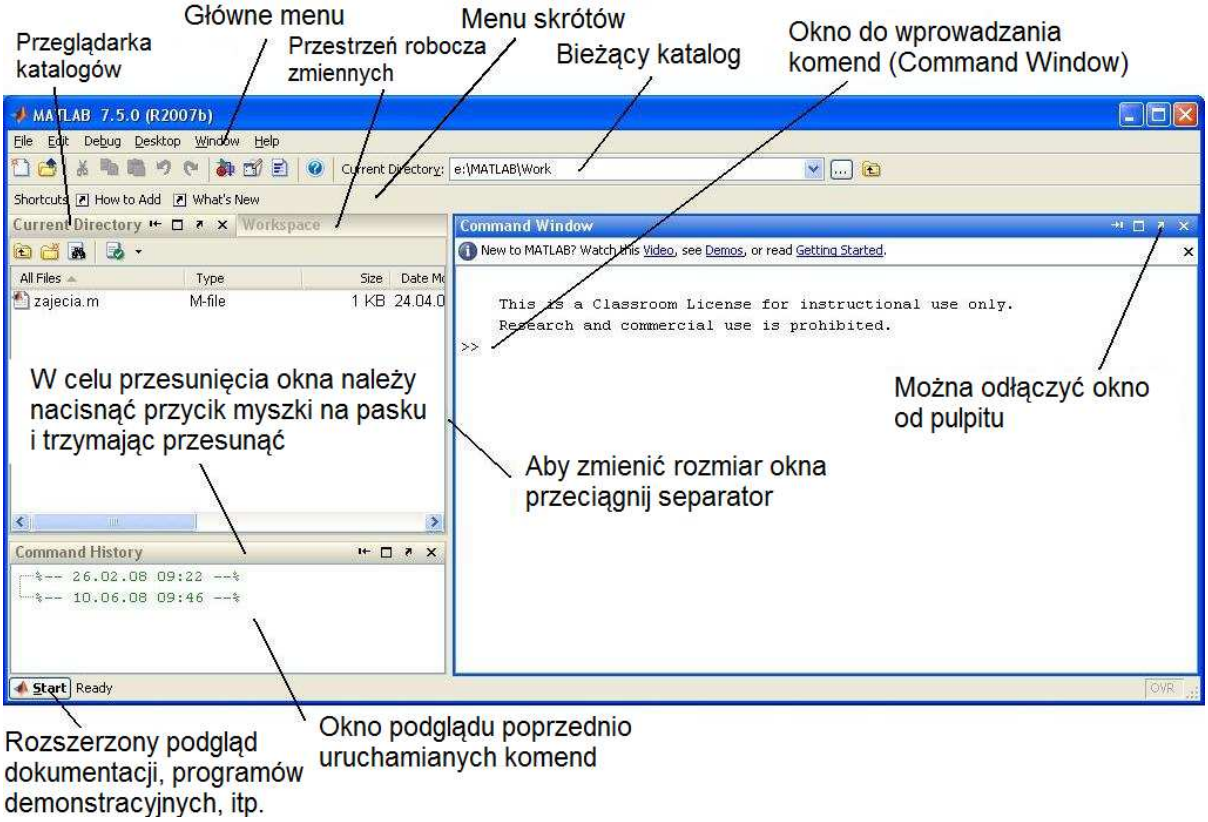
Wprowadzenie do MATLAB-a (instrukcja 1) ma na celu przedstawienie układu graficznego pulpitu MATLAB-a oraz podstawowych poleceń.

Wszystkie instrukcje laboratoryjne powstały przy założeniu posiadania podstawowych informacji o systemie Windows.

W celu uruchomienia MATLAB-a należy podwójnie kliknąć na ikonę znajdującą się na pulpicie komputera. Zamknięcie MATLAB-a następuje po wybraniu z menu MATLAB-a polecenia Exit lub napisaniu w Oknie Komend (Command Window) słowa quit i naciśnięciu klawisza enter.

2. Pulpit MATLAB-a

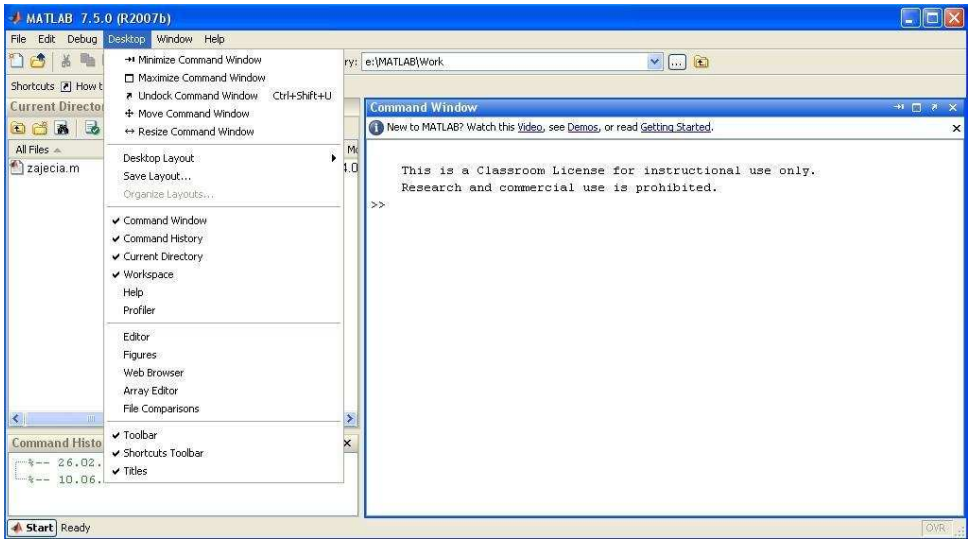
Po uruchomieniu MATLAB-a pojawia się standardowy układ okien, z jakich składa się pulpit MATLAB-a (rys. 1).



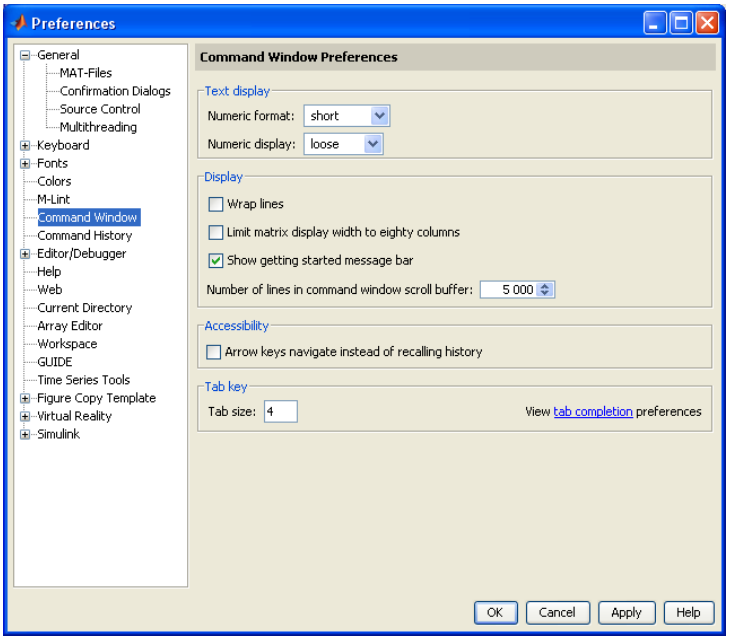
Rys. 1. Pulpit MATLABa z opisem

Każde okno to zestaw narzędzi (graficzny interfejs komunikacyjny) do zarządzania miedzy innymi plikami, zmiennymi, itp. Liczba interfejsów wzrasta prawie z każdą nową wersją MATLAB-a. Wygląd pulpitu MATLAB jest zależny od ustawień własnych oraz od wersji MATLAB-a, która się używa. W typ paragrafie jest opisany zestaw narzędzi dostępnych począwszy od wersji MATLAB-a 7.0 (R14). Pulpit MATLAB-a składa się z okien (Command Window, Command History, itp.), które może być przemieszczane, oddzielane (dołączane) od (do) pulpitu (undock, dock). Można zmieniać układ okien i ich rozmiar. Nie wszystkie narzędzia (okna) są domyślnie uruchomione. W celu zmiany liczby okien należy menu głównym pulpitu wybrać odpowiednie pozycje (rys. 2).

Niektóre właściwości okien mogą być zmieniane po wybraniu w menu głównym pulpitu MATLAB-a opcji *Preferences* (właściwości), (rys. 3). Dla przykładu można zmienić sposób wyświetlania cyfr w Oknie poleceń (Command Window), typ czcionki, rozmiar i kolor czcionek, itp.



Rys. 2. Menu pulpitu



Rys. 3. Właściwości

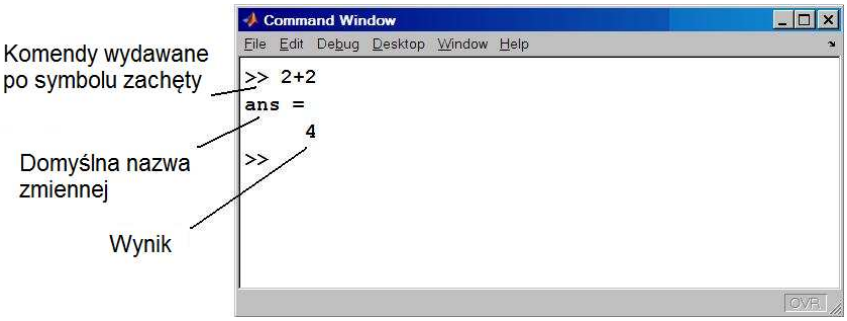
2.1. Opis wybranych narzędzi pulpitu MATLAB-a

— Okno poleceń (Command Window) - do wprowadzania zmiennych, uruchamiania funkcji i M-plików.
— Historia poleceń (Command History) - do podglądu poprzednio wydawanych poleceń (historia) oraz ich ponownego wykonywania.
— Przycisk startowy (Start Button and Launch Pad) - łatwy dostęp do narzędzi, demonstracyjnego oprogramowania, dokumentacji, itp.
— Okno pomocy (Help Browser) - do przeglądania dokumentacji (opisy funkcji, itp.).
— Podgląd bieżącego katalogu (Current Directory Browser) - do podglądu zawartości bieżącego katalogu oraz zmiany bieżącego katalogu.

— Przeglądarka przestrzeni roboczej (Workspace Browser) - do podglądu zmiennych zmagazynowanych w pamięci.
— Edytor tablic (Array Editor) - do podglądu zawartości zmiennych oraz ich edycji.
— Edytor plików (Editor/Debugger) - do tworzenia i edytowania M-plików (programy).
— Pasek skrótów (Shortcut toolbar) - do tworzenia zestawów instrukcji i ich skrótów w formie ikon.

2.2. Okno poleceń

Instrukcje mogą być bezpośrednio wydawane w oknie poleceń w MATLAB-ie. Poprzedza je symbol zachęty, >>. Napisane instrukcje są wykonywane po naciśnięciu klawisza Enter. Okno poleceń może być używane w formie zwykłego kalkulatora. Jeśli polecenie nie jest poprzedzona nazwą zmiennej MATLAB utworzy zmienną pod domyślną nazwą *ans* (rys. 4).



Rys. 4. Bezpośrednie wydawanie poleceń

▷ Ćwiczenie 2.1.

W celu zapoznania się z podstawowymi komendami MATLAB-a uruchom następujące instrukcje:

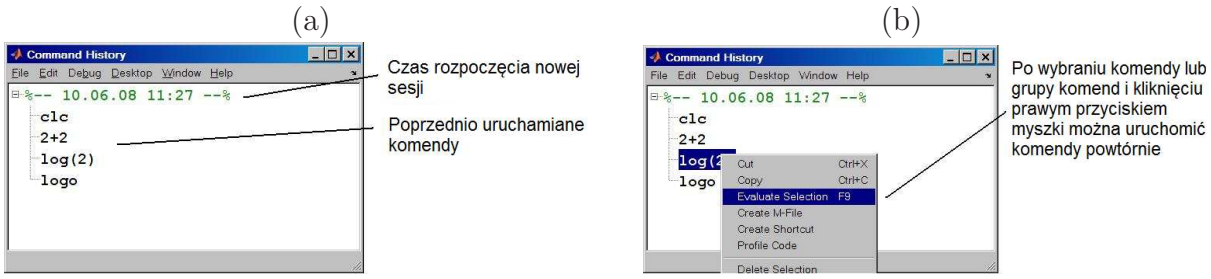
— `(-2+2/3-0.66)^2`
— `exp(1+0.442)`
— `sin(3.14/2)`
— `sin(pi/2)`
— `tan(pi)`
— `atan(inf)`
— `log(2)`
— `log(exp(1))`
— `log10(2)`
— `help sin`
— `% 3+4  opis`
— `dir`
— `clc`
— `logo`

Uwaga 2.1 Kropka, czy przecinek

Zauważ, że MATLAB używa kropki (.), jako separator w systemie dziesiętnym w przeciwieństwie do przecinka (,) używanego np. w Excelu. Zauważ również, że MATLAB rozróżnia polecenia wydawana dużymi i małymi literami, np. `Exp` to nie to samo co `exp`.

2.3. Historia poleceń (Command History)

W oknie historii poleceń można podglądnąć poprzednio uruchamiane instrukcje, uruchomić je powtórnie (Evaluate selection F9), skopiować do schowka (Copy), wyciąć (Cut), utworzyć M-plik (Create M-file), utworzyć skrót, itp. (rys. 5)



Rys. 5. (a) Okno historii poleceń, (b) edycja poleceń

▷ **Ćwiczenie 2.2.**
Pośród poprzednio uruchomionych poleceń wybrać jedną i powtórnie ją wykonać.

2.4. Okno pomocy (Help Browser)

MATLAB posiada dużą liczbę użytecznych dokumentacji dotyczących każdej funkcji zawartej w MATLAB-ie. Dokumentacja zawiera również podstawowe (encyklopedyczne) informacje dotyczące pojęć matematycznych, numerycznych, itp. Najprostszym sposobem wywołania okna pomocy jest wykonanie w oknie komend instrukcji *helpwin*. Jeśli szukamy pomocy na konkretny temat (np. funkcji *sin*) należy wywołać następujący zestaw instrukcji: `helpwin sin` lub `help sin`.

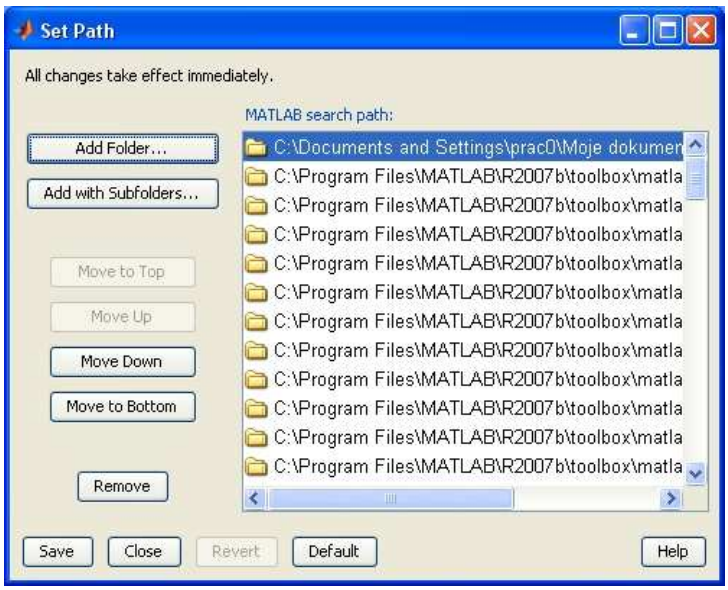
▷ **Ćwiczenie 2.3.**
Wykonaj następujące polecenia:

— Używając okna pomocy zapoznaj się z następującymi funkcjami: `log`, `exp`, `clear`, `lookfor`.
— Używając polecenia `lookfor` znajdź wszystkie M-pliki zawierające słowo *log* oraz *avi*.

Uwaga 2.2 Zatrzymanie programu
Jeśli MATLAB wykonuje zadaną instrukcje zbyt długo i chcemy ją przerwać należy jednocześnie nacisnąć dwa klawisze i je przytrzymać: Ctrl+C.

2.5. Podgląd bieżącego katalogu (Current Directory Browser)

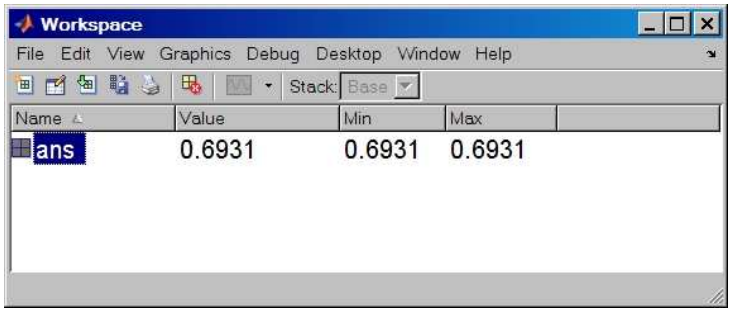
Każdy plik, który chcemy uruchomić w MATLAB-ie musi znajdować się w katalogu bieżącym (current directory) lub być na liście przeszukiwanych ścieżek dostępu. Zmiana bieżącego katalogu może nastąpić poprzez okno (Current Directory Browser). Aby sprawdzić jakie katalogi znajdują się na liście przeszukiwanych lub zmienić tę listę należy w *Main menu/File/Set Path* (rys. 6). Domyślnie, lista ścieżek dostępu zawiera wszystkie katalogi *toolboxów*.



Rys. 6. Okno do zmiany przeszukiwanych ścieżek dostępu

2.6. Okno przestrzeni roboczej (Workspace Browser)

Przestrzeń robocza MATLAB-a jest bardzo ważnym pojęciem. Przestrzeń ta zawiera wszystkie zmienne utworzone i zachowane w pamięci MATLAB-a. Okno przestrzeni roboczej dostarcza dużej liczby informacji dotyczących zachowanych zmiennych. Za pomocą tego okna można również edytować zawartość zmiennych, zapisać je w formie pliku, dokonać szybkiej wizualizacji danych. Wczytywanie zapisanych danych w formie pliku następuje np. za pomocą instrukcji importu danych (Import Data) umieszczonej w *Main menu/File*.



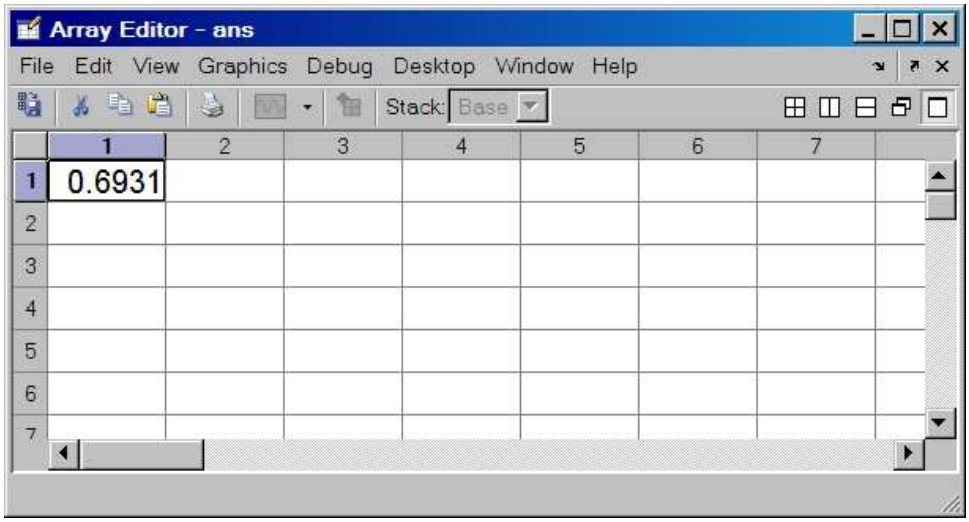
Rys. 7. Okno przestrzeni roboczej

2.7. Edytor tablic (Array Editor)

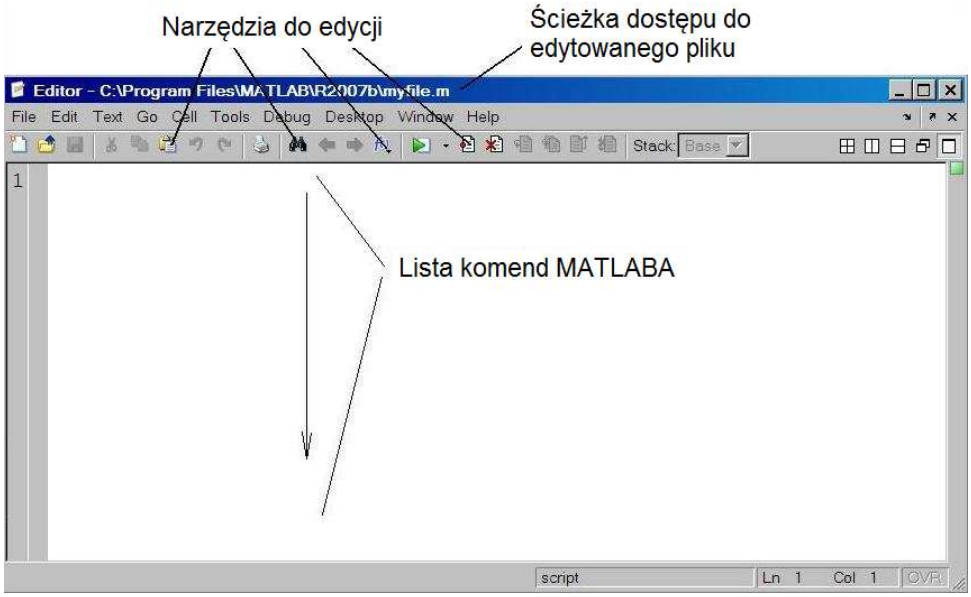
Edytor tablic może być uruchomiony, np. Poprzez podwójne kliknięcie zmiennej w przestrzeni roboczej. Używając edytora tablic można manipulować danymi.

2.8. Edytor plików (Editor/Debugger)

Edytor plików (rys. 9) jest narzędziem do tworzenia i edycji plików zawierających instrukcje zrozumiałe i uruchamiane przez MATLAB-a. Najprostszym sposobem uruchomienia edytora plików wraz z otwarciem pliku jest wydanie polecenia *edit* (w oknie poleceń).

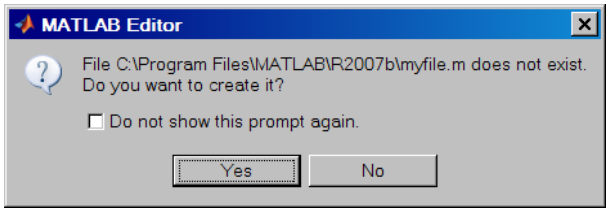


Rys. 8. Edytor tablic (Array Editor)



Rys. 9. Okno edytora plików

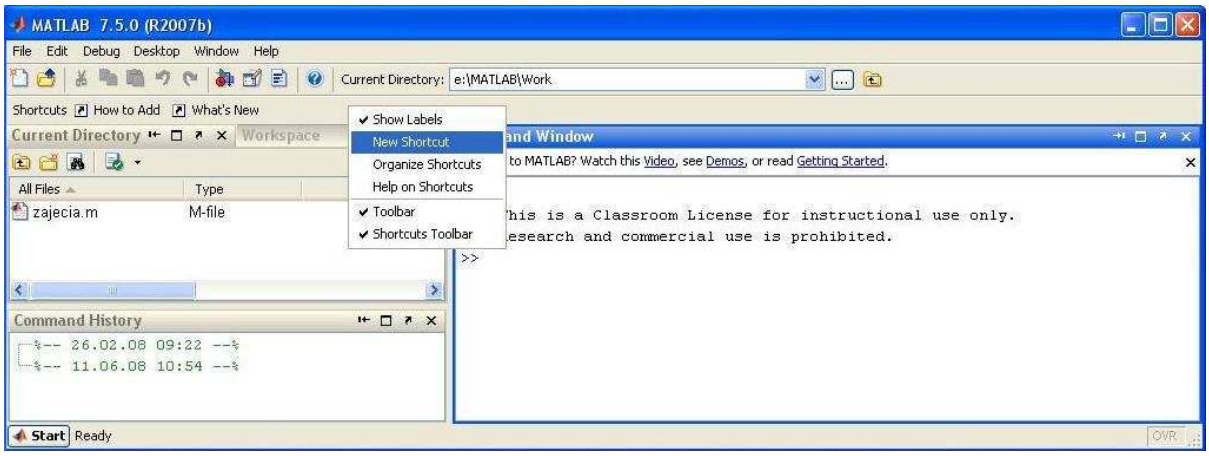
Jeśli chcemy otworzyć plik o znanej nazwie należy wydać następująca komendę: *edit 'filename'*, np. *edit var* . Jeśli wywołany plik nie istniej (na liście przeszukiwanej) MATLABzapyta czy utworzyć plik o podanej nazwie. Na przykład, przy wydaniu następującej polecenia: *edit myfile*123 pojawi się następujące okno



Rys. 10. Pytanie o utworzenie nowego pliku

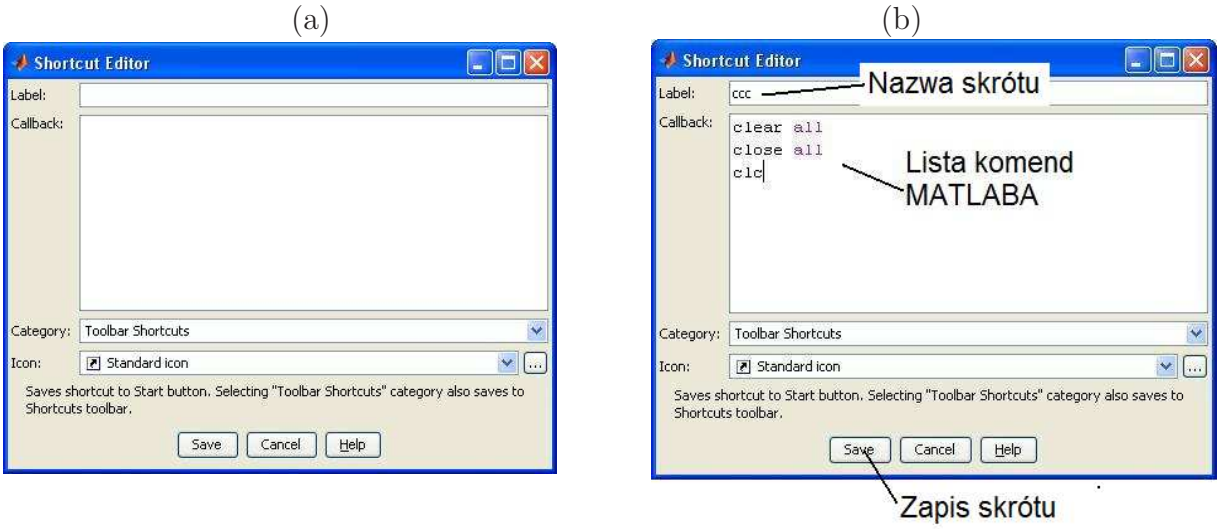
2.9. Pasek skrótów (Shortcut toolbar)

Pasek skrótów jest bardzo użytecznym narzędziem. Pozwala na utworzenie listy najczęściej używanych komend i zapisanie ich w formie skróty wywoływanego po naciśnięciu odpowiedniej ikony. Na przykład polecenia takie jak: czyszczenie pamięci (*clear all*), zamykanie rysunków (*close all*), czyszczenie ekranu (*clc*) to komendy wykonywane bardzo często i warto z nich utworzyć skrót. Aby tego dokonać należy kliknąć prawym przyciskiem myszy na pasku skrótów (rys. 11) i wybrać polecenie *New Shortcut* (nowy skrót).



Rys. 11. Tworzenie skrótów

Okno przedstawione na rysunku 12a powinno się pokazać, które pozwoli na wprowadzenie listy poleceń wywołanych po naciśnięciu ikony. W tym celu wprowadzamy listę poleceń, jak pokazano na rysunku 12b.



Rys. 12. Edycja skrótu

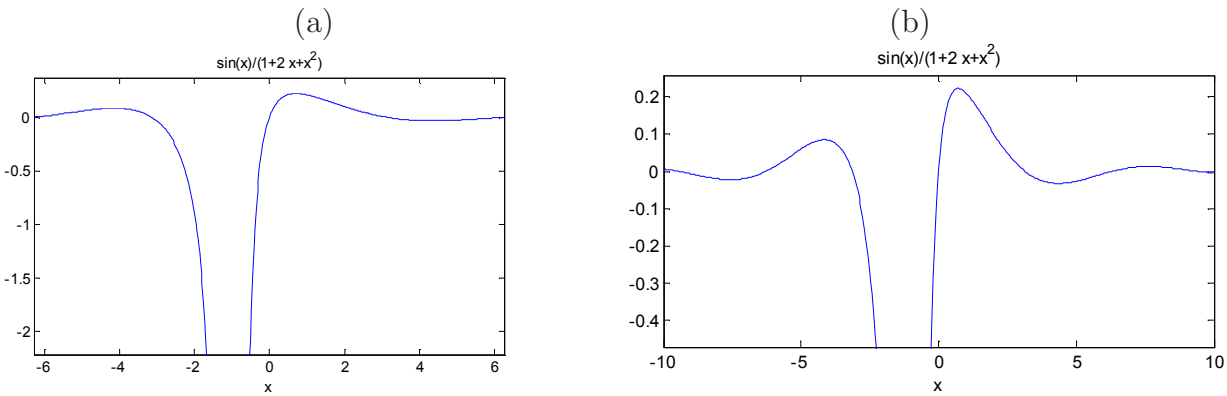
▷ **Ćwiczenie 2.4.**
Wykonaj następujące polecenia:

a) Sprawdź znaczenie komend: `clear all`, `close all`.
b) Zmodyfikuj utworzony skrót poprzez dodanie polecenia `pack`.
c) Sprawdź działanie polecenia `pack`.

3. Proste polecenia do wizualizacji równań matematycznych

MATLAB oferuje dużą liczbę funkcji graficznych do szybkiej analizy równań matematycznych zapisanych w formie symbolicznej. Do takich funkcji należą: `ezplot`, `ezsurf`, `ezmesh`. Dla przykładu, w celu dokonania wizualizacji wyrażenia: $y = \frac{\sin(x)}{1+2x+x^2}$ wydajemy następujące polecenie w MATLAB-e:

```
ezplot('sin(x)/(1+2*x+x^2)')
```



Rys. 13. Wynik wykonania poleceń: (a) $ezplot('sin(x)/(1 + 2 * x + x\hat{}2)')$, (b) $ezplot('sin(x)/(1 + 2 * x + x\hat{}2)', [-10\ 10])$

Wyrażenie `sin(x)/(1+2*x+x^2)` jest przekazywane w postaci symbolicznej jako łańcuch znaków (string) do funkcji *ezplot*. Łańcuchy znaków w MATLAB-e są reprezentowane, jako ciąg znaków w apostrofach. Zakres zmienności argumentu analizowanej funkcji jest dobierany domyślnie, ale można zadać odgórnie wymagany zakres, np.

```
ezplot('sin(x)/(1+2*x+x^2)',[-10 10])
```

lub

```
ezplot('cos(x)/(1+x+x^2)',[-10 10 -5 5])
```

a) Określ różnicę w wywołaniu następujących poleceń:

```
ezplot('cos(x)/(1+x+x^2)',[-4 4 -1 2])
```

oraz

```
ezplot('cos(x)/(1+x+x^2)-y',[-4 4 -1 2])
```

Uwaga 3.1 Nowe okno rysunku

MATLABtworząc domyślnie nowy rysunek (figure) zamyka okno z poprzednim rysunkiem. Aby tego uniknąć należy przed wywołanie nowej funkcji graficznej wywołać polecenie: *figure*.

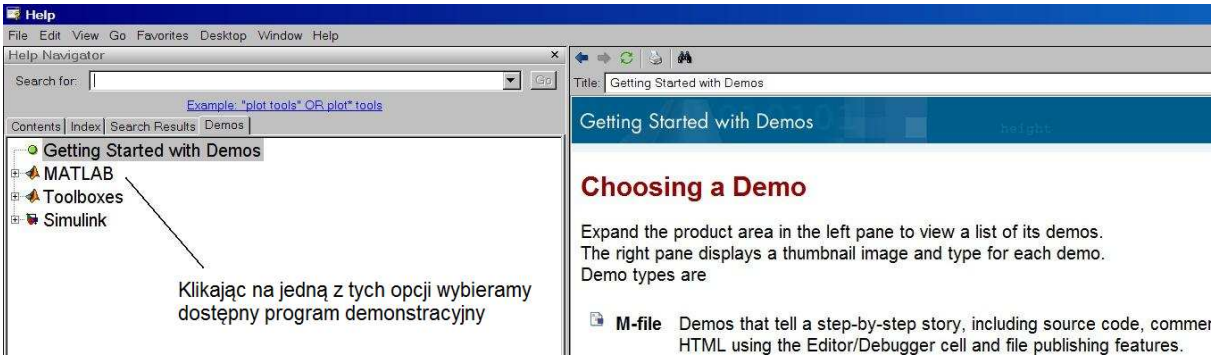
b) Dokonaj wizualizacji wyrażeń: $x^2 + y^2 - 4$, $x^2 + x + y^2 + y - 1$, $1 - e^{-x}$.
c) Sprawdź działanie następujących instrukcji:
   — `ezplot('2*cos(t)', '2*sin(t)')`
   — `ezplot3('cos(2*pi*t)','sin(2*pi*t)','t',[0 5])`
   — `ezsurf('x*y*exp(-(x^2+y^2))')`
   — `ezmeshc('y/(1 + x^2 + y^2)',[-5 5 -2*pi 2*pi])`

4. Programy demonstracyjne

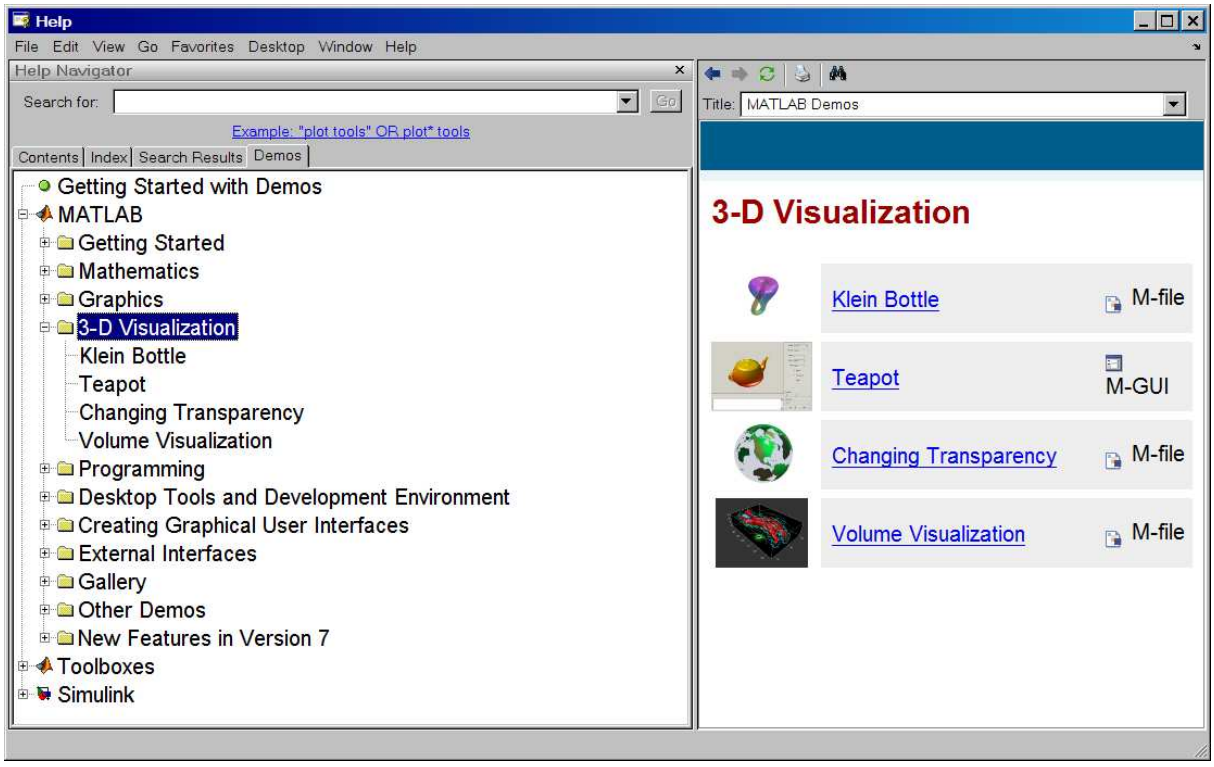
MATLAB oferuje dużą ilość programów demonstracyjnych, które ukazują jego możliwości w wielu dziedzinach nauki i techniki. W celu wywołania programów demonstracyjnych wykonujemy polecenie `demos`. Polecenie to uruchamia okno, gdzie dokonujemy wybory programu demonstracyjnego (rys. 14)

Na przykład, po wybraniu symbolu (+) przy nazwie MATLAB(rys. 15) i rozwijamy listę. Wybieramy *3-D Visualization* i klikamy na przykład zwany *Teapot* po prawej stronie okna.

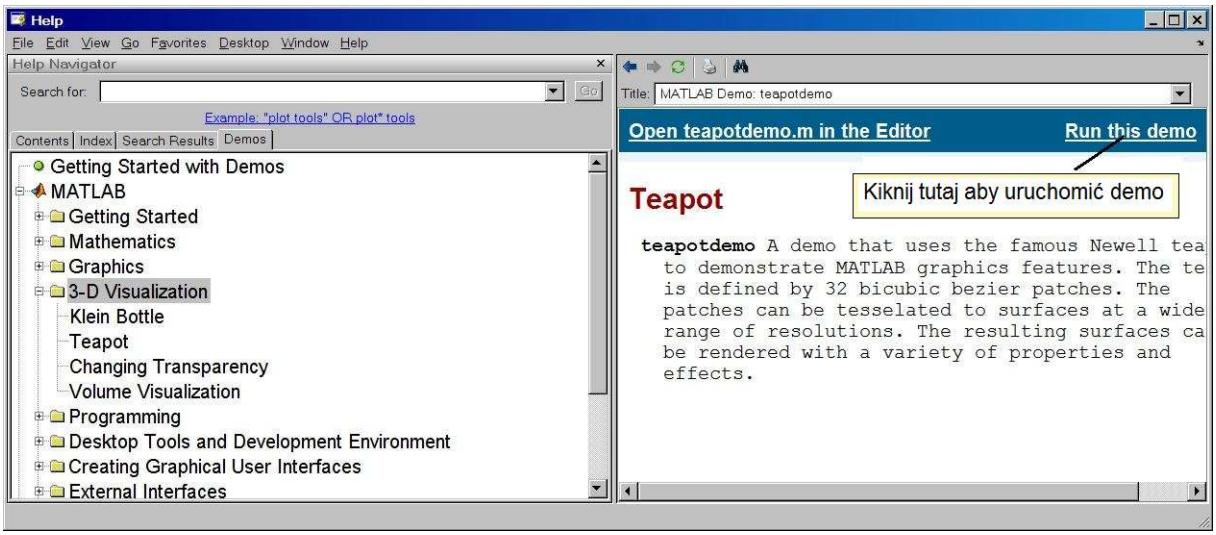
Po prawej stronie wyświetla się nowe okno, które pozwala na uruchomienie programu (rys. 16).



Rys. 14. Wybieramy program demonstracyjny



Rys. 15. Wybieramy grupę programów demonstracyjnych *3-D Visualization*



Rys. 16. Program demonstracyjny *Teapot*

▷ **Ćwiczenie 4.1.**
Uruchom przynajmniej dwa programy demonstracyjne.

5. Macierz

W MATLAB-ie zmienne numeryczne jak i nienumeryczne mogą być zapisywane w różny sposób. Jednak, podstawowym formatem zapisu danych w MATLAB-ie jest macierz. W matematyce macierz jest to układ zapisanych w postaci prostokątnej tablicy danych nazywanych elementami bądź współczynnikami będących elementami ustalonego zbioru, zwykle liczbowego [6]. W MATLAB-ie skalary są traktowane, jako macierz o wymiarach 1-na-1 (jeden wiersz, jedna kolumna), wektor, jako macierz o wymiarach 1-na-n (wierszowy) lub n-na-1 (kolumnowy). MATLAB pozwala na wykonywanie operacji macierzowych szybko i efektywnie bez konieczności tworzenia specjalnych programów.

Uwaga 5.1 Macierz

Linie poziome w macierzy to wiersze, linie pionowe to kolumny. Macierz z m wierszami i n kolumnami nazywana jest macierzą m-na-n, gdzie m i n to wymiary macierzy. Pierwszym podawanym wymiarem macierzy jest zawsze liczba wierszy a następnym jest liczba kolumn.

6. Definiowanie macierzy

MATLABpozwala na wprowadzanie macierzy do obliczeń na kilka sposobów:

— jawne wprowadzanie elementów macierzy,
— importowanie macierzy z plików zewnętrznych,
— generacja macierzy przy użyciu wbudowanych funkcji,
— tworzenie macierzy za pomocą własnych funkcji.

Jawne wprowadzanie elementów macierzy polega na wykonaniu, np. następującej operacji:

```
A=[2 3 4; 8 4 3; 2 4 5]
```

Powyższy zapis tworzy macierz o wymiarach 3-na-3 i przypisanie jej nazwy „A". Należy zauważyć, że

— użycie średnika „;", wskazuje na koniec wiersza,
— kolejne elementy w wierszu są oddzielane spacjami lub przecinkami,
— cała lista elementów jest zawarta w nawiasach kwadratowych, [ ].

Można wprowadzać cyfry w formacie wykładniczym, np. 2e3 (2000), 3e-3 (0.003), itd.

Wprowadź do MATLAB-a następującą instrukcję

```
B=[2 3 4; 8 4 3; 2 4 5];
```

Zwróć uwagę, w jaki sposób macierze A i B zostały wyświetlone na ekranie (średnik „;" postawiony na końcu polecenia oznacza, że wynik operacji nie jest wyświetlany na ekranie). Wprowadzona macierz A jest automatycznie zapamiętana w przestrzeni roboczej MATLAB-a. Można się do niej odwołać poprzez jej nazwę.

▷ **Ćwiczenie 6.1.**
Wprowadź do przestrzeni roboczej MATLAB-a następujące macierze:

$$C = \begin{bmatrix} 2 & 3 & 9 & 0 \\ -3 & 2/3 & 44 & 0.5 \end{bmatrix}, \; D = \begin{bmatrix} 0.92 & 1e-3 & -4e2 \end{bmatrix}, \; E = \begin{bmatrix} 3 \\ 3.33 \\ 3.1415 \end{bmatrix}$$

Uwaga 6.1 Usuwanie zmiennych

W celu usunięcia macierzy z przestrzeni roboczej należy użyć poleceń: `clear 'nazwazmiennej'`, np. `clear C` lub `clear C D`. Aby usunąć wszystkie zmienne wprowadź: `clear all`

MATLAB umożliwia generowanie macierzy specjalnych, np. poprzez polecenia:

`ones` - tworzy macierz gdzie wszystkie elementy to cyfra 1,
`zeros` - tworzy macierz gdzie wszystkie elementy to zera,
`eye` - tworzy macierz gdzie elementy na głównej przekątnej to cyfry 1 a pozostałe to zera,
`magic` - tworzy macierz "magiczną",
`randn` - tworzy macierz zawierającą elementy losowe o rozkładzie normalnym.

• **Przykład 6.1.**
Generacja takich samych elementów.

Polecenie `ones` jest często używane w MATLAB-ie w celu, np. generacji macierzy o elementach o takich samych wartościach:

```
>> B=3.5*ones(1,4)
B =
    3.5000    3.5000    3.5000    3.5000
>>
```

• **Przykład 6.2.**
Powielanie elementów macierzy

Komenda `ones` jest często używana do tworzenia macierzy z tablicy jednowymiarowej, np. mamy następującą tablice danych `w=[2 7 5 3]` a potrzebujemy utworzyć macierz np. `e` złożoną z trzech wierszy tablicy `w`.
Najprostszym sposobem wykonania tego zadania jest

```
>> e=[w;w;w]
e =
     2     7     5     3
     2     7     5     3
     2     7     5     3
>>
```

Nie jest to najelegantszy sposób i poza tym nie zawsze wykonalny, ponieważ liczba powtórzeń może być zmienna lub bardzo duża. Znacznie lepszym sposobem jest wykorzystanie komendy `ones` w następujący sposób:

```
>> e=ones(3,1)*w
e =
     2     7     5     3
     2     7     5     3
     2     7     5     3
>>
```

Uwaga 6.2 Mnożenie macierzy

Mnożenie macierzy *a*b* jest możliwe tylko wtedy jeśli odpowiednie wymiary macierzy *a* i *b* są zachowane, tj. macierz *a* ma wymiar *[m n]* i *b* ma wymiar *[n k]* to *c=a*b*, gdzie *c* ma wymiar *[m k]*.

7. Sprawdzanie wymiarów macierzy

Często istnieje konieczność sprawdzenia wymiarów stworzonych lub importowanych macierzy, np. dla celu prawidłowego mnożenia macierzy. Podstawową komendą do sprawdzenia wymiarów macierzy A jest

```
[m, n]=size(A)
```

Gdzie *m* jest liczbą wierszy a *n* jest liczbą kolumn macierzy A. Polecenie `size` może być również używana w następujący sposób:

`rows=size(A,1)` - tylko liczba wierszy jest zwracana `cols=size(A,2)` - tylko liczba kolumn jest zwracana.

Inne przydatne komendy to sprawdzania wymiarów macierzy to
`n = length(A)` - zwraca rozmiar wymiaru większego spośród wierszy i kolumn.
`n = numel(A)` - zwraca liczbę wszystkich elementów macierzy A.

▷ **Ćwiczenie 6.2.**
Wykonaj następujące polecenia:

— sprawdź w pomocy znaczenie następujących komend: `ones`, `zeros`, `eye`, `magic`, `randn`.
— użyj komendy `zeros` w celu stworzenie macierzy ,
— użyj komendy `ones` w celu stworzenie macierzy.

• **Przykład 7.1.**
Wymiary macierzy

```
>> [m n]=size(A)
m =
     3
n =
     5
```

8. Podstawowe operacje na macierzach

8.1. Mnożenie, dodawanie i odejmowanie macierzy

Operacje podstawowe takie jak: *, +, - są przeprowadzane w MATLAB-ie w sposób macierzowy. Oznacza to, że mnożenie macierzy jest możliwe po spełnieniu warunku równości odpowiednich wymiarów mnożonych macierzy. W operacji sumowania i odejmowania liczba elementów w macierzach musi być równa, za wyjątkiem, kiedy od dowolnej macierzy A odejmujemy lub dodajemy skalar *s*, wtedy operacja jest wykonywana dla każdego elementu macierzy A.

▷ **Ćwiczenie 8.1.**
Wykonaj polecenie

Utwórz macierze `A=[2 3 4; 3 4 5];` i `B=[4 3; 4 5; 5 6];` dla tak utworzonych macierzy wyznacz macierz `C=A*B`.

```
>> C=A*B
C =
    40    45
    53    59
```

W przypadku próby wykonania komendy `A*C` otrzymasz następujący komunikat (w kolorze czerwonym)

```
??? Error using ==> mtimes
Inner matrix dimensions must agree.
```

Co oznacza: wewnętrzne wymiary macierzy muszą być zgodne.

8.2. Notacja punktowa

Jeśli istnieje konieczność wymnożenia elementów dwóch macierzy w formie element przez element należy użyć notacji punktowej, czyli: `c=a.*b` (rozmiar macierzy a i b musi być taki sam). Inne operacje takie jak: dzielenie, potęgowanie są przeprowadzane w taki sam sposób: `c=a./b`; `c=a.^2` lub `c=a.^b`.

• Przykład 7.2.
Liczba wszystkich elementów macierzy

```
>> A=randn(3,5);
numel(A)
ans =
    15
```

▷ Ćwiczenie 7.1.
Wykonaj następujące polecenia:

— Wprowadź do MATLAB-a następującą komendę:
`A=randn(abs(round(10*randn(1,2)))+1);`
Wywołaj tą komendę kilka razy i za każdym razem sprawdź rozmiar macierzy A.
— sprawdź znaczenie funkcji `abs` i `round`,
— utwórz macierz B takiego samego rozmiaru jak macierz A ale o wszystkich elementach równych zero.

▷ Ćwiczenie 8.2.
Wykonaj następujące polecenia

— Utwórz macierz `a=[2 3 4 5 9]` i podnieś każdy element macierzy `a` do potęgi 2,
— utwórz macierz `b=[9 8 7 5 5]` i podnieś każdy element macierzy a do potęgi odpowiedniego elementu macierzy `b`,
— wykonaj operację mnożenia każdego elementu macierzy `a` przez odpowiedni element macierzy `b`.

8.3. Operacja sumowanie elementów macierzy i podobne

MATLABoferuje dużą liczbę komend operacji na elementach macierzy, np.:

`sum` - sumowanie elementów macierzy,
`diff` - odejmowanie elementów macierzy,
`prod` - wymnażanie elementów macierzy,
`det` - wyznacznik macierzy,
`trace` - suma elementów głównej przekątnej,
`eig` - wektory i wartości własne,
`inv` - macierz odwrotna,
`norm` - norma macierzy lub wektora,

i wiele innych. Kiedy potrzebujesz zdefiniować specyficzną operację na elementach macierzy należy wcześniej sprawdzić, czy MATLAB nie ma już zdefiniowanej takiej operacji !!!

▷ Ćwiczenie 8.3.
Utwórz macierz o elementach losowych:

`a=randn(5,10)` i:
— zsumuj elementy macierzy `a` w każdej kolumnie, aby utrzymać wektor 1-na-10,
— zsumuj elementy macierzy `a` w każdym wierszu, aby utrzymać wektor 5-by-1,
— zsumuj wszystkie elementy macierzy `a` aby utrzymać skalar.

▷ **Ćwiczenie 8.4.**
Utwórz macierz o elementach losowych:

`w=randn(1,12)` i wykonaj następujące polecenia:
— `dw=diff(w)`,
— `ddw=diff(dw)`,
— porównaj otrzymany wynik z wynikiem operacji `diff(w,2)`.

8.4. Operacja transponowania i podobne

Operacja transponowania (wiersze stają się kolumnami) jest wykonywana za pomocą apostrofu (').

• **Przykład 8.1.**
Wykonaj komendy:

```
>> A=[1 2 3 4; 5 6 7 8]
A =
     1     2     3     4
     5     6     7     8
>> B=A'
B =
     1     5
     2     6
     3     7
     4     8
```

MATLAB oferuje dużą liczbę operacji zmieniających kolejność wystepowania elementów w macierzy. Sprawdź następujące komendy: `fliplr`, `flipdim`, `flipud`, `rot90`.

▷ **Ćwiczenie 8.5.**
Utwórz macierz 5-na-2 o elementach losowych a następnie dokonaj manipulacji w kolejności elementów, taka by ostatni wiersz stał się pierwszym a pierwszy ostatnim.

9. Wskaźniki (indeksy)

Indeksowane w MATLAB-ie pozwala na dostęp do dowolnego elementu macierzy. Dla przykładu element w i-tym wierszu i j-tej kolumnie macierzy A jest oznaczany jako A(i,j).

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

>> A(2,3)
ans =
     3
```

W przypadku odwołania się do elementu, który nie istnieje otrzymujemy komunikat o błędzie:

```
>> t=A(5,6)
??? Index exceeds matrix dimensions.
>>
```

Jeśli umieścisz nowy element w macierzy poza jej wymiarem, MATLAB powiększy macierz, np.:

```
>> A(3,4)=1
A =
     2     3     4     0
     5     8     3     0
     9     1     2     1
>>
```

Używanie wskaźników jest pomocne przy podmienianiu danych w macierzy, np.:

```
>> A(1,1)=100
A =
   100     3     4     0
     5     8     3     0
     9     1     2     1
>>
```

▷ **Ćwiczenie 9.1.**
Utwórz macierz B z macierzy A (powyżej) tak, aby macierz B składała się z dwóch wierszy. Pierwszy wiersz macierzy B ma zawierać elementy 1 i 3 z drugiego wiersza macierzy A, wiersz drugi macierzy B składać się powinien z 3 i 2 elementu wiersza nr 3 macierzy A.

10. Dwukropek jako operator

Dwukropek : jest jednym z najczęściej wykorzystywanych operatorów w MATLAB-ie. Pozwala na efektywne manipulowanie elementami macierzy. Dwukropek może być wykorzystywana w różny sposób:

I) jako operator do tworzenia macierzy

```
>> Q=1:10
Q =
     1     2     3     4     5     6     7     8     9    10
>>
```

lub

```
>> Q=-2:0.5:1
Q =
   -2.0000   -1.5000   -1.0000   -0.5000         0    0.5000    1.0000
>>
```

Dwukropek w tworzeniu macierzy bazuje na następujących zasadach

- $j : k$ to samo co $[j, j + 1, ..., k]$
- $j : k$ pusta macierz jeśli $j > k$
- $j : i : k$ to samo co $[j, j + i, j + 2i, ..., k]$

- $j : i : k$ pusta macierz jeśli $i = 0$, lub $i > 0$ i $j > k$, lub $i < 0$ i $j < k$

  gdzie: i, j, k to skalary.

  II) jako operator wyboru wszystkich elementów danej kolumny lub wiersza, np.:

- $A(:, j)$ wszystkie elementy j-tej kolumny macierzy A
- $A(i, :)$ wszystkie elementy i-tego wiersza macierzy A
- $A(:, :)$ odwołanie do wszystkich elementów macierzy A (nie używane)
- $A(j : k)$ odwołanie się do elementów: $A(j), A(j + 1), ..., A(k)$
- $A(:, j : k)$ odwołanie się do elementów $A(:, j), A(:, j + 1), ..., A(:, k)$
- $A(:)$ wszystkie elementy macierzy A są umieszczane w jednej kolumnie.

• Przykład 10.1.
Dwukropek jako operator

```
>> A=[1 9 3 1 4; 8 0 4 2 8; 8 6 1 7 0]
A =
     1     9     3     1     4
     8     0     4     2     8
     8     6     1     7     0
```

Aby wywołać tylko pierwszy wiersz macierzy A wykonaj:

```
>> A(1,:)
ans =
     1     9     3     1     4
>>
```

• Przykład 10.2.
Dwukropek jako operator

Aby wywołać tylko drugą kolumnę macierzy A wykonaj:

```
>> A(:,2)
ans =
     9
     0
     6
>>
```

• Przykład 10.3.
Dwukropek jako operator

Aby wywołać, co drugi element wiersza drugiego wykonaj:

```
>> A(2,1:2:end)
ans =
     8     4     8
>>
```

Uwaga 10.1 Ostatni element

Symbol `end` użyty we wskaźnikach macierzy oznacza ostatni element: ostatni wiersz lub ostatnią kolumnę.

• **Przykład 10.4.**
Dwukropek jako operator

Aby wywołać, co drugi element we wszystkich wierszach wykonaj:

```
>> A(:,1:2:end)
ans =
     1     3     4
     8     4     8
     8     1     0
>>
```

• **Przykład 10.5.**
Dwukropek jako operator

Zmiana kolejności elementów macierzy na jedną kolumnę:

```
>> A(:)
ans =
     1
     8
     8
     9
     0
     6
     3
     4
     1
     1
     2
     7
     4
     8
     0
>>
```

▷ **Ćwiczenie 10.1.**
Dana jest macierz:

$$W = \begin{bmatrix} 1 & 5 & 2 & 8 & 3 \\ 2 & 3 & 5 & 6 & 1 \\ 3 & 6 & 3 & 2 & 0 \end{bmatrix}.$$

Utwórz wektor w pobierając odpowiednio elementy z macierzy W
$w = [\ 2 \quad 3 \quad 5 \quad 6 \quad 1 \quad 0 \quad 2 \quad 3 \quad 6 \quad 3\ ]$

11. Wykorzystanie operacji macierzowych; Rozwiązywanie układu równań

Aby rozwiązać następujący układ równań

$$\begin{aligned} x_1 + x_2 + x_3 &= 1 \\ 2x_1 + 4x_2 + x_3 &= 1 \\ 3x_1 + x_2 + 5x_3 &= 0 \end{aligned}$$

można wykorzystać operacje na macierzach. Powyższy układ równań zastępujemy równaniem macierzowym:

$$\mathbf{A} * \mathbf{x} = \mathbf{b}$$

gdzie: $\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 4 & 1 \\ 3 & 1 & 5 \end{bmatrix}$, $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$.

Sprawdzamy wymiary macierzy [3-na-3]*[3-na-1]=[3-na-1] - wymiary wewnętrzne mnożonych macierzy są równe. Rozwiązanie równania macierzowego następuje po prostych przekształceniach:
$\mathbf{A} * x = \mathbf{b}/\mathbf{A}^{-1}$
$\mathbf{A}^{-1} * \mathbf{A} * \mathbf{x} = \mathbf{A}^{-1} * \mathbf{b}$
$\mathbf{x} = \mathbf{A}^{-1} * \mathbf{b}$
gdzie $\mathbf{A}^{-1}$ jest macierzą odwrotną do macierzy $\mathbf{A}$.

Równanie $\mathbf{x} = \mathbf{A}^{-1} * \mathbf{b}$ jest rozwiązywane w MATLAB-ie w prosty sposób. Definiujemy macierze $\mathbf{A}$ i $\mathbf{b}$

```
>> A=[1 1 1; 2 4 1; 3 1 5]
A =
     1     1     1
     2     4     1
     3     1     5
>> b=[1 1 0]'
b =
     1
     1
     0
>>
```

i wykonujemy następującą operację

```
>> x=A^-1*b
x =
    7.5000
   -2.5000
   -4.0000
>>
```

lub

```
>> x=inv(A)*b
x =
    7.5000
```

```
   -2.5000
   -4.0000
>>
```

Sprawdzamy poprawność rozwiązania, czyli $\mathbf{A} * \mathbf{x} - \mathbf{b} = 0$

```
>> A*x-b
ans =
  1.0e-014 *
    0.0888
    0.1776
    0.3553
>>
```

Otrzymane składniki to błąd numerycznego rozwiązania naszego układu równań (*residuals*).

Literatura

[1] Getting Started with MATLAB, version 6, The MathWorks

[2] CHANDLER G.: Introduction to Matlab, Mathematics Department, The University of Queensland, February 2000, (http://www.eeng.nuim.ie/ semcloone/Teaching/Matlab/mlb.pdf)

[3] KAROLCZUK A.: Płaszczyzny krytyczne w modelach zmęczenie materiałów przy wieloosiowych obciążeniach losowych, Praca doktorska, Wydział Mechaniczny Politechniki Opolskiej, 2003, s. 146

[4] GRIFFITHS D. F.: An Introduction to Matlab, Department of Mathematics, The University Dundee, 2005, http://www.maths.dundee.ac.uk/ ftp/na-reports/MatlabNotes.pdf)

[5] www.mathworks.com

[6] http://pl.wikipedia.org/wiki/Macierz

A. Opis wybranych funkcji

ezplot

Funkcja do wykreślania symbolicznie zapisanych równań matematycznych w przestrzeni 2D.

Składnia
`ezplot(fun)` wykreślanie funkcji $fun(x)$ w domyślnym zakresie $-2\pi < x < 2\pi$.

`ezplot(fun2)` wykreślanie niejawnie zdefiniowanej funkcji $fun2(x, y) = 0$ w domyślnym zakresie $-2\pi < x < 2\pi$ oraz $-2\pi < y < 2\pi$.

`ezplot(fun,[a b])` wykreślanie funkcji $fun(x)$ w zakresie $a < x < b$.

`ezplot(fun2,[a b])` wykreślanie funkcji $fun2(x, y) = 0$ w zakresie $a < x < b$ oraz $a < y < b$.

`ezplot(fun2,[xmin, xmax, ymin, ymax])` wykreślanie funkcji $fun2(x, y) = 0$ w zakresie $xmin < x < xmax$ oraz $ymin < y < ymax$.

`ezplot(funx,funy)` wykreślanie parametrycznie zdefiniowanej planarnej krzywej $funx(t)$ i $funy(t)$ w domyślnym zakresie $0 < t < 2pi$.

`ezplot(funx,funy,[tmin, tmax])` wykreślanie parametrycznie zdefiniowanej planarnej krzywej $funx(t)$ i $funy(t)$ w zakresie $tmin < t < tmax$.

`ezplot(fun, [a b], fig)`,
`ezplot(fun2,[xmin, xmax, ymin, ymax], fig)`
lub `ezplot(funx,funy,[tmin, tmax], fig)` wykreślanie funkcji w zdefiniowanym zakresie w oknie rysunku fig (uchwyt na okno rysunku).

`ezplot(ax, )` wykreślanie funkcji na osiach wykresu określonej uchwytem ax zamiast na osiach wykresu bieżącego.

`h = ezplot()` zwrot uchwytu na wykreślany obiekt.

• Przykład A.1.
Najprostszym sposobem wykreślenia funkcji jest podanie jej postaci ciągu znaków (string), np.

```
ezplot('x^2 + 3*x + 8')

ezplot('x*y+3*x^2 + y^2 - 2')
```