WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI I INFORMATYKI

INSTYTUT AUTOMATYKI I INFORMATYKI

KIERUNEK AUTOMATYKA I ROBOTYKA

STUDIA STACJONARNE I STOPNIA

PRZEDMIOT : : LABORATORIUM PODSTAW AUTOMATYKI

1. WSTĘP DO MATLABA cz.1

MATLAB to pakiet przeznaczony do wykonywania obliczeń numerycznych oraz do graficznej prezentacji wyników. Posiada przyborniki (toolbox) z procedurami i funkcjami specyficznymi dla danej dziedziny nauki np. SIMULINK do symulacji systemów dynamicznych.

Praca w środowisku MATLAB polega na wydawaniu poleceń w wierszu poleceń okna komend MATLABA, które są wykonywane przez interpreter.

Standardowe polecenia MATLABA należy pisać małymi literami. Do nazw własnych programów i zmiennych można używać małych i dużych liter.

Wartość polecenia zakończonego średnikiem nie będzie wyświetlana na ekranie.

Nazwy zmiennych rozpoczynają się od litery, a następnie może wystąpić dowolna kombinacja liter, cyfr i znaków podkreślenia (rozróżniane 19).

Zmienna nie musi być deklarowana ani mieć określonego rozmiaru.

1. **Wprowadzanie zmiennej w oknie Command:**
   >> a=5 *% z wyświetleniem wartości*

   >> a=5; *% bez wyświetlenia wartości*

2. **Sprawdzenie wartości istniejącej zmiannej:**
   >> a

3. **Wyświetlenie listy zmiennych i ich wymiarów:**
   >> who
   Lub
   >> whos

4. **Kasowanie zmiennej:**
   >> clear a

>> clear all *%kasowanie wszystkich zmiennych*

MATLAB wykonuje operacje na macierzach. Wektory i skalary są szczególnymi przypadkami macierzy posiadającymi jeden wiersz i/lub jedna kolumnę.

Macierze występują także w roli wartości logicznych oraz łańcuchów tekstowych ('str').

Innym typem danych są dane reprezentowane przez liczby zespolone, z których również mogą być budowane macierze.

Definicji macierzy można dokonać poprzez: wymienianie elementów, wygenerowanie elementów, zbudowanie z innych macierzy, przez łączenie wymienionych technik.

Elementy w wierszu macierzy muszą być oddzielone spacją lub przecinkiem. Średnik lub znak nowego wiersza kończy wiersz macierzy i powoduje przejście do następnego.

Cała lista elementów musi być ujęta w nawiasy kwadratowe.

5. **Wprowadzenie elementów macierzy:**

   >> A=[3,1;6,4];

   lub

   >>A=[3 1;6 4];

   >>B=[1.5,2,0;2,5.8,1;7,5,4.3];

6. **Generowanie macierzy i wektorów (min:krok:max):**

   >>x=[1:9]

   >>x1=[2:2:20]

Zaleca się wcześniejsze generowanie macierzy przez rezerwację pamięci, gdy jej rozmiar jest znany.

7. **Generowanie macierzy:**

   >>y= eye(3)

   >>y1=eye(3,2)

   >>y2=ones(2)

   >>y3=ones(2,3)

   >>y4=zeros(4)

   >>y5=zeros(1,4)

   >>y6=rand(1,5) *% macierz liczb pseudolosowych z przedziału <0,1> o rozkładzie jednostajnym*

   >>y7=randn(1,6) *% macierz liczb pseudolosowych rozkładzie normalnym ze średnią 0 i wariancją 1*

8. Wybór elementów macierzy

>> x=(2:5) *% elementy wektora wierszowego od 2 do 5*

>> B(2,:) *% 2 wiersz macierzy B*

>> B(2,2:3) *%elementy o numerach od 2 do 3 w 2 wierszu macierzy B*

>> B(:,3) *% 3kolumna macierzy B*

>> B(:,2:3) *% kolumny od 2 do 3 macierzy B*

>>B(1:2,2:3) *% elementy w kolumnach od 2 do 3 wierszy od 1 do 2*

>>b=[1,2,3];

>>B(b,2:3) *%elementy w kolumnach od 2 do 3 w wierszach macierzy B o numerach określonych przez elementy wektora b*

>>B(:,:) *%cała macierz B*

>>B(:) *% cała macierz B w postaci wektora kolumnowego*

9. Rozmiary macierzy i wektorów:

>>size(A) *%wyświetla rozmiar macierzy A*

>>[n,m]=size(A) *%przypisanie zmiennej n liczby wierszy, a zmiennej m liczby kolumn*

>>n=size(A,1) *%przypisanie zmiennej n liczby wierszy*

>>m=size(A,2) *%przypisanie zmiennej m liczby kolumn*

>>length(x) *% długość wektora x lub dłuższy z wymiarów macierzy*

10. Operacje na macierzach:

>> B' *% transpozycja*

>> C=[A,[2;5];[7 8 9]]

>>D=[A,[2;5]]

>>C+B

>> C*B

>>C.*B

>>B*D *%!!!*

>>D*B

>>C^2

>>C.^2

>>B/C

>>B./C

11. Operacje na plikach

>> pwd *% aktualny katalog roboczy*

>>cd, *% aktualny katalog roboczy*

>>cd .. *% przejście do katalogu wyżej*

>>cd matlab

>>save myfile A *% zapis zmiennej w pliku *.mat*

>>save ‘my file’ B *% zapis zmiennej w pliku *.mat gdy nazwa zawiera spacje*

>>clear A B

>>A

>>B

>> load myfile A *%załadowanie zmiennej z pliku*

>>load ‘my file’ B

>>whos

12. Pomoc MATLABA jest dostępna za pomocą polecenia *help* lub *help nazwa_polecenia*:

>>help

>>help save

Funkcje graficzne zwracają wykresy w nowych oknach, których nazwy jeżeli nie zostaną zdefiniowane przyjmują nazwę *Figure No.* Z kolejnym numerem. Polecenie *figure* otwiera kolejne okno graficzne.

13. **Rysowanie funkcji:**

```
>> help plot % w celu wyświetlenia parametrów polecenia

>>t=0:0.1:4*pi;

>>y=sin(t);

>>y1=sin(4*t);

>>plot(t,y)

>>hold on

>>plot(t,y1,'r')

>> close all %zamknieciewszystkich okien z wykresami okien z wykresami
```

14. **Rysowanie funkcji ciągłej podanej wzorem:**

```
>>fplot('sin(x)',[0 4*pi])
```

15. **Umieszczanie kilku wykresów w aktywnym oknie:**

```
>>y2=sin(2*t);

>>subplot(221)

>>plot(t,y)

>>subplot(223)

>>plot(t,y1,'r-.')

>>subplot(223)

>>plot(t,y2,'r--')
```

M-plik jest plikiem dyskowym z rozszerzeniem *.m stworzonym przy użyciu edytorów (np. edytora MATLABA, notatnika Windows). M-pliki zawierają sekwencję poleceń MATLABA. Mogą zawierać wywołania do innych m-plików. Rozróżniamy m-pliki skryptowe zawierające ciągi poleceń niezbędnych do wykonania danego programu, działają na zmiennych globalnych w przestrzenie roboczej MATLABA, nie mają argumentów wejściowych i wyjściowych.

M-pliki funkcyjne zawierają funkcje tworzone przez użytkownika, akceptują parametry wejściowe i wyjściowe, działają na zmiennych lokalnych, można zdefiniować zmienne globalne poleceniem *global*, rozpoczynają się od słowa kluczowego *function*, muszą zawierać nazwę funkcji oraz listę argumentów wejściowych w nawiasach (). Nazwa funkcji powinna być taka sama jak nazwa m-pliku, w którym ja zapisano.

16. **Tworzenie skryptów:**

myfile.m

```
for i=1:4
t=0:0.1:4*pi;
if i==1
        y=funkcja(t,1);
        subplot(2,2,i)
        plot(t,y)
elseif i==2
        y=funkcja(t,2);
        subplot(2,2,i)
        plot(t,y,'r')
elseif i==3
        y=funkcja(t,4);
        subplot(2,2,i)
        plot(t,y,'g')
else
        y=funkcja(t,8);
        subplot(2,2,i)
        plot(t,y,'k')
end
end
```

funkcja.m

```
function [y]=funkcja(t,n)
```

```
y=sin(n*t)

end
```