Środowisko pracy

Po uruchomieniu programu Matlab powita on użytkownika znakiem zachęty:

```
>>
```

Od tego momentu można przystąpić do pracy, która polega na tekstowym wprowadzaniu komend. Polecenia można wydawać kolejno lub grupami, przy czym polecenia wydawane grupowo muszą być oddzielone średnikami lub przecinkami. Gdy tekst polecenia nie mieści się w jednej linijce można przejść do następnej wpisując wielokropek

```
...
```

i wciskając <ENTER>.

Zatwierdzenie komendy następuje również poprzez naciśnięcie klawisza <ENTER>.

```
>>polecenie(zmienne wejściowe) <ENTER>
```

Zmienne wejściowe oddziela się przecinkami. Jeśli komenda nie wymaga parametrów wejściowych nawiasy omija się.

Każde polecenie niezakończone średnikiem powoduje wypisanie wyniku operacji na ekranie. Może to być dosyć kłopotliwe np. przy przetwarzaniu dużych macierzy. Ładując do przestrzeni roboczej macierz o dużej liczbie elementów z pliku dyskowego „macierz.dat" poleceniem

```
>>load macierz.dat
```

musimy liczyć się z wyświetlaniem wszystkich zmiennych zapisanych w pliku.

Kluczowe znaczenie w Matlabie mają zmienne. Reprezentują one dane, które zostały im przypisane. Przypisanie wartości zmiennej odbywa się następująco:

```
>>zmienna=wyrażenie <ENTER>
```

Gdy nazwa zmiennej nie jest zdefiniowana

```
>>wyrażenie <ENTER>
```

wynik zapamiętywany jest pod zmienną standardową o nazwie „ans".

Zmiennej można przypisać wartość bezpośrednio. Napisanie

```
>>a=32
```

spowoduje przypisanie do a wartości 32.

Uwaga. Program rozróżnia wielkie i małe litery

Wielką zaletą Matlaba jest to, że zmiennych nie trzeba wcześniej deklarować.

Oczywiście zmienne nie muszą być skalarne. Matlab pozwala na proste operowanie macierzami. Na przykład generacja wektora o elementach zmieniających się od 1 do 100 co 1 wygląda następująco:

```
>>w=[1:100] <ENTER>
```

zaś wektor o elementach od 3 do 10 zmieniających się co 0.5 jest generowany poleceniem

```
>>w=[3:0.5:10] <ENTER>.
```

Uwaga. Argumenty funkcji oraz wskaźniki do elementu macierzy umieszcza się w nawiasach okrągłych, zaś elementy macierzy w nawiasach kwadratowych.

Macierze i wektory możemy definiować wpisując kolejne elementy w następującej postaci:

```
>>B=[1 3 5 9] <ENTER> (wektor 1 3 5 9)
```

```
>>B=[1 2 4; 5 3 5; 2 2 1] <ENTER>
```

(macierz o wierszach 1 2 4, 5 3 5 oraz 2 2 1).

Dostęp do elementów macierzy odbywa się poprzez określenie wiesza i kolumny wyrazu

```
>>B(2,3) <ENTER>
```

spowoduje następującą reakcję

```
>>ans = 5.
```

Notacja

```
>>B(:,3) <ENTER>
```

spowoduje wypisanie trzeciej kolumny. Podobnie można użyć notacji dwukropkowej dla określonego wiersza lub grupy wierszy

```
>>B(1:3,:) <ENTER>
```

Polecenie to wypisze elementy z wierszy od 1 do 3.

Zdefiniowana zmienna znajduje się w przestrzeni roboczej i mamy do niej dostęp poprzez jej nazwę. Aby zorientować się, jakie i ile zmiennych jest już zdefiniowanych można użyć polecenia

```
>>whos <ENTER>,
```

które wyświetli informacje o aktualnych zmiennych.

Zakończenie sesji z Matlabem powoduje utratę zmiennych. Można tego uniknąć zachowując zmienne w plikach dyskowych

```
>>save plik.dat B –ascii <ENTER>.
```

spowoduje zachowanie zmiennej B w pliku plik.dat w formacie ASCII.

Polecenie

```
>>save A A <ENTER>.
```

Zachowa zmienną A w pliku A.mat, który jest plikiem binarnym.

Zapis całej przestrzeni roboczej można przeprowadzić wpisując

```
>>save nazwa <ENTER>.
```

Skoro istnieje możliwość zapisania danych na dysku, na pewno można je wczytać do przestrzeni roboczej. Służy do tego polecenie load.

```
>>load A <ENTER>
```

wczyta zmienną A i nada jej nazwę A.

Innym użytecznym poleceniem może być polecenie diary.

```
>>diary nazwa <ENTER>
```

zapisze historię poleceń i ich rezultat w pliku nazwa. Wyłączenie „dziennika" następuje poprzez wprowadzenie komendy

```
>>diary off <ENTER>.
```

Z operacjami na plikach ściśle związane są polecenia podobne do poleceń systemowych DOS. Należą do nich:

dir wypisanie zawartości katalogu

type edycja pliku

delete usunięcie pliku

cd zmiana katalogu

pwd wyświetla nazwę aktualnego katalogu

Niezależnie od tych poleceń istnieje możliwość wykonania komendy systemowej

```
>>!komenda <ENTER>
```

spowoduje wykonanie polecenia „komenda" systemu operacyjnego.

Definiowanie własnych poleceń

Bardzo wygodnym narzędziem jest możliwość zapisywania sekwencji poleceń w pliku tekstowym o rozszerzeniu .m. Reprezentuje on polecenie, które wykonuje instrukcje zawarte w pliku. W ten sposób działają standardowe komendy i funkcje programu Matlab. Przy tworzeniu własnych poleceń lub funkcji wygodnie jest posługiwać się komentarzami. Znakiem początku komentarza jest znak %.

Standardowe funkcje, zmienne i stałe

Wybrane zmienne i stałe:

ans standardowa zmienna przypisywana do wyrażenia nieskojarzonego
pi liczba pi
i,j jednostka urojona
date aktualna data
clock aktualny czas

Wybrane funkcje:

abs	wartość bezwzględna, moduł liczby zespolonej
cos	cosinus
sin	sinus
tan	tangens
cot	cotangens
angle	kąt fazowy liczby zespolonej w radianach
ceil	zaokrąglenie w kierunku +nieskończoności
conj	liczba sprzężona
exp	funkcja ekspotencjalna
fix	zaokrąglenie w kierunku zera
floor	zaokrąglenie w kierunku -nieskończoności
imag	część urojona liczby zespolonej
log	logarytm naturalny
log10	logarytm przy podstawie 10
real	część rzeczywista liczby zespolonej
rem	reszta z dzielenia
round	zaokrąglenie
sign	znak funkcji
sqrt	pierwiastek kwadratowy
eye	tworzenie macierzy jednostkowej
ones	macierz jedynkowa
zeros	macierz zerowa
rand	macierz losowa o rozkładzie równomiernym

Więcej informacji na temat poleceń można uzyskać wpisując

```
>>help nazwa <ENTER>.
```

Operatory

W Matlabie operatory arytmetyczne można podzielić na dwie grupy: tablicowe i macierzowe. Operatory tablicowe odnoszą się do elementów macierzy, zaś macierzowe dokonują działań określonych przez algebrę liniową. Definiując 2 wektory

```
>>a=[1 2 5]; b=[2 3 1] <ENTER>
```

możemy dokonać na nich mnożenia tablicowego

```
>>a .* b <ENTER>
```

Wynikiem operacji jest wektor [2 6 5].
Mnożenie macierzowe będzie możliwe dopiero po dokonaniu transpozycji wektora b.
Poniżej przedstawione są operatory tablicowe i macierzowe.

Operacja	Symbol operacji tablicowej	Symbol operacji macierzowej
dodawanie	+	+
odejmowanie	-	-
mnożenie	.*	*
potęgowanie	.^	^
dzielenie prawostronne	./	/
dzielenie lewostronne	.\	\
sprzężenie		‘
transpozycja		.’

Operatory logiczne i relacji.

<	mniejszy od
>	większy od
<=	mniejszy lub równy
>=	większy lub równy
==	równy
~=	różny
&	AND
|	OR
~	NOT
xor	XOR

Instrukcje

W Matlabie możemy używać funkcji warunkowej oraz dwóch funkcji iteracyjnych. Instrukcja warunkowa ma postać:

```
if wyrażenie
      polecenia
elseif wyrażenie
      polecenia
else
     polecenia
end
```

Pierwsza z instrukcji iteracyjnych wygląda następująco:

```
while wyrażenie
     polecenia
end
```

Wykonuje się ona nieokreśloną ilość razy.Dla ściślejszego kontrolowania ilości iteracji można użyć instrukcji

```
for zmienna=wyrażenie
     polecenia
end
```

Definiowanie sygnałów

Ponieważ sygnał cyfrowy można traktować jako ciąg liczb Matlab jest świetnym narzędziem do symulacji cyfrowego przetwarzania sygnałów. Podstawową sprawą jest umiejętność generowania sygnałów, szczególnie tych najczęściej używanych jak impuls jednostkowy, skok jednostkowy, sinus, cosinus, przebieg prostokątny, piłokształtny itp.
Poniżej przedstawione są przykłady generowania różnych sygnałów. Nie są to jedyne możliwe metody ich generacji, tym bardziej nie muszą być one optymalne.

Impuls jednostkowy (długość obserwacji n próbek, jedynka na m.-tej pozycji)

```
>>a=zeros(n); imp=a(1,:); imp(m)=1; <ENTER>
```

Skok jednostkowy (długość obserwacji n próbek, skok od m.-tej pozycji)

```
>> a=zeros(n); skok=a(1,:); i=m; while i<=n skok(i)=1;
i=i+1; end <ENTER>
```

Przebieg piłokształtny (długość obserwacji n próbek, okres m próbek)

```
>>a=[1:n]; pila=rem(a/m) <ENTER>
```

Sinusoida o częstotliwości n i spróbkowana z częstotliwością m (liczba próbek wynosi k)

```
>>t=[1:k]; <ENTER>
>>a=sin(2*pi*n*t/m); <ENTER>
```

Przykładowe wartości liczbowe wynoszą n=300, k=100. Sprawdzić jak wygląda sygnał dla częstotliwości próbkowania m=900, m=910, m=700. Należy zwrócić uwagę, że sygnał sinusoidalny cyfrowy nie zawsze odzwierciedla oczekiwania odnośnie jego kształtu, może nawet nie być okresowy.

Sygnał prostokątny

Mając wygenerowaną sinusoidę łatwo zrobić z niej prostokąt za pomocą funkcji *sign*.

```
>>t=[1:k]; <ENTER>
>>a=sin(2*pi*n*t/m); <ENTER>
>>b=sign(a); <ENTER>
```

Parametr taki jak wypełnienie można zmieniać np. przesuwając sinusoidę wzdłuż osi y. Zmiana amplitudy sinusoidy polega na przemnożeniu sygnału przez odpowiedni czynnik.

Wizualizacja wyników

Podstawowym poleceniem tworzenia wykresu jest *plot*. Po wydaniu komendy zostanie utworzone osobne okno, w którym widoczne będą zmienne użyte jako argumenty polecenia *plot*. Polecenie to może mieć wiele różnych wywołań, w zależności od oczekiwanej postaci wykresu. Poniżej jest przedstawionych kilka przykładowych opcji polecenia.

plot(x)	wykres liniowy wektora x
plot(x,y)	wykres liniowy o współrzędnych w wektorach x,y
plot(x,y,'typ linii')	wykres liniowy o linii zdefiniowanej przez typ linii (opis poniżej)
plot(x1,y1,'typ linii1', x2,y2,'typ linii2'...)	kilka wykresów w jednym układzie współrzędnych

Gdy x jest macierzą zostanie wygenerowanych tyle wykresów, ile kolumn liczy macierz zaś y staje się zbędne.

W tabeli zestawione są typy linii dla wykresów.

Symbol	typ linii	symbol	kolor linii
.	punktowa	y	żółty
o	okręgi	m	purpurowy
x	znaki x	c	błękitny
+	znaki +	r	czerwony
*	gwiazdki	g	zielony
-	ciągła	b	niebieski
:	punktowa	w	biały
-.	Kreskowo-punktowa	k	czarny
_	kreskowa		

Oprócz wykresów liniowych istnieje cała gama innych możliwości prezentacji wyników. Niektóre z możliwych wykresów zostały przedstawione w tabeli.

Nazwa	opis	nazwa	opis
Plot	liniowy	stairs	schodkowy
Errorbars	błędy	hist	histogram
Semilogx	oś x w skali logarytmicznej	semilogy	oś y w skali logarytmicznej
Loglog	logarytmiczny	stem	dyskretny
Fplot	funkcja ciągła	fill	wypełniony prostokątny

Szczególnie przydatna z punktu widzenia wizualizacji sygnałów dyskretnych jest funkcja *stem*.

Istniejący wykres wygodnie jest odpowiednio opisać. Służą do tego polecenia zamieszczone poniżej.

grid	nałożenie siatki na wykres
gtext	umieszczenie napisu w miejscu wskazanym myszką
legend	legenda do wykresu
text	umieszczenie tekstu w wybranym miejscu
title	tytuł wykresu
xlabel	opis osi x
ylabel	opis osi y

Zmiany skali wykresu możemy dokonać poleceniem

```
>>axis([x0,x1,y0,y1])
```

Użyteczne może być tworzenie kilku wykresów na jednym rysunku. Do ich tworzenia służy polecenie

```
>>subplot(nx,my,i)
```

gdzie nx-liczba wykresów w poziomie, my-liczba wykresów w pionie, i-numer wykresu

Przykład:

>>*subplot*(1,2,1)	utworzenie 2 wykresów i wybranie pierwszego
>>*plot*(a)	wykreślenie zmiennej a na wykresie pierwszym
>>*subplot*(1,2,2)	wybranie drugiego wykresu
>>*plot*(b, '-- g')	wykreślenie zmiennej b na wykresie drugim linią kreskową, zieloną
>>*title*('tytul 2')	tytuł wykresu drugiego
>>*xlabel*('opis x')	opis osi x wykresu drugiego

Zachowywanie rysunków na dysku

Jednym ze sposobów zachowania wykresu jest jego skopiowanie poleceniem Copy z menu Edit znajdującego się w oknie graficznym programu Matlab i wklejenie np. do edytora Word. Format kopiowanego rysunku można wybrać w menu Edit – Copy options. Możliwe są dwa typy rysunków bitmapa i Windows Metafile.

Ćwiczenia

1. Zdefiniować dwa wektory o tym samym wymiarze i wykonać na nich mnożenia macierzowe (*) i tablicowe (.*). Porównać wyniki.
2. Spróbować tych samych operacji na wektorze o wymiarze 1×n i macierzy o wymiarze n×m.
3. Wygenerować sinusoidę zawierającą np. 200 próbek i próbkowaną z różnymi częstotliwościami (np. od 2.1 do 50 razy na okres). Zaobserwować przypadki próbkowania z częstotliwością nie będącą całkowitą wielokrotnością częstotliwości sinusoidy.
4. Za pomocą instrukcji iteracyjnych wygenerować skok jednostkowy o długości n i różnych „punktach skoku”
5. Utworzyć różnego rodzaju m-pliki i sprawdzić, co się dzieje po ich wywołaniu (np. utworzyć polecenie tworzące wykres zmiennej wejściowej oraz opisujące wykres zmiennymi typu string podanymi w wywołaniu polecenia). Stworzyć polecenie mające takie samo działanie jak ćwiczenie 4. Spróbować zmodyfikować polecenie tworzące wykres tak, aby rozpoznawało liczbę zmiennych wejściowych i generowało odpowiednią ilość podwykresów, dla każdej zmiennej osobny.
6. Utworzyć przebieg wykładniczy zespolony. Wykreślić jego część rzeczywistą, urojoną, fazę, moduł.
7. Zmodulować sinusoidę innym przebiegiem.