ĆWICZENIE 0 - WPROWADZENIE DO MATLABA

0.1. CELE ĆWICZENIA

- Pierwszy kontakt z pakietem matematycznym MATLAB,
- Wstępny trening w posługiwaniu się wybranymi, podstawowymi narzędziami pakietu.

0.2. O SPOSOBIE OSIĄGANIA CELÓW ĆWICZENIA

Pakiet matematyczny MATLAB jest uznanym i powszechnie stosowanym narzędziem programowania, również w dziedzinie cyfrowego przetwarzania sygnałów. Jego znajomość wśród studentów może być różna. Dlatego w ćwiczeniu wprowadzającym ćwiczący wykonają pewien prosty, ale przemyślany zestaw zadań treningowych, pozwalający odkryć lub przypomnieć zalety tego narzędzia na tyle, by je stosować w następnych ćwiczeniach:

- biernie – w ramach korzystania z przygotowanego oprogramowania tych ćwiczeń
- i czynnie – we własnym zakresie, na przykład przy rozwiązywaniu problemów w ramach przygotowania do ćwiczenia.

Aby sięganie do MATLABa ułatwić, opracowano przegląd ważniejszych instrukcji MATLABa i wybranych procedur cyfrowego przetwarzania sygnałów. Zawiera on skrócony opis tych instrukcji i procedur. Ma formę osobnego opracowania - dodatku przydatnego w realizacji ćwiczeń wykonywanych w ramach Laboratorium Podstaw Cyfrowego Przetwarzania Sygnałów.

Zakłada się, że – w ramach przygotowania się do ćwiczenia – ćwiczący:

- zapozna się z treścią niniejszej instrukcji oraz wyżej wymienionego dodatku,
- zastanowi się nad postawionymi w instrukcji problemami i spróbuje na nie odpowiedzieć, a jeżeli już miał styczność z MATLABem, sprawdzi przy jego użyciu swoje odpowiedzi,
- przepisze na dyskietkę 3.5" w formie plików tekstowych skrypty i funkcje matlabowe umieszczone w niniejszej instrukcji (i przyniesie dyskietkę z tymi plikami na zajęcia; usprawni to wykonywanie ćwiczenia)
- przyniesie kartkę *Krótkie sprawozdanie* (patrz str. 6 instrukcji)

0.3 SPOTKANIE Z MATLABEM

Poniższy materiał jest bardzo skromnym wprowadzeniem do MATLABa. Dla pełniejszego spojrzenia warto sięgnąć do wcześniej wymienionego dodatku zatytułowanego *Przegląd ważniejszych instrukcji MATLABa. Cyfrowe przetwarzanie sygnałów - wybrane procedury bibliotek MATLABa. Grafika w MATLABie* **,** bogatej literatury, bądź do dostępnego na stronach internetowych materiału zatytułowanego *MATLAB Reference Manual.*

Zamierzenie tej części jest następujące:

1. Zapoznanie się z systemem *Help* MATLABa w celu przestudiowania podstawowych komend i składni MATLABa,
2. Kontakt z funkcjami i **`m-plikami`** MATLABa,
3. Poznanie wybranych innych cech specyficznych pakietu.

0.3.1 Podstawowe polecenia

Rozpocznij poznawanie MATLABa od następujących ćwiczeń:

(a) Obejrzyj Wprowadzenie do MATLABa pisząc **`intro`** w linii zachęty programu;

(b) Zapoznaj się z możliwościami uzyskiwania pomocy od MATLABa. Wpisz każdą z poniższych linii i przeczytaj tekst pomocy na temat wyszczególnionych w nich komend:

```
help
help plot
help ones
```

Jeżeli zakres wyświetlania przekracza obszar ekranu od dołu, można zmusić MATLABa, by wyświetlał informacje pojedynczymi stronami. Do tego używamy polecenia `more on`.

(c) Użyj MATLABa jako kalkulatora. Spróbuj następujących obliczeń:

```
pi*pi-10
sin(pi/4)
ans^2      %<---"ans" przechowuje ostatni wynik
```

(d) W MATLABie pod nazwami zmiennych mogą być przechowywane wartości i macierze. Wypróbuj następujące zapisy:

```
cos(pi/3);    %<--- przypisane do czego?
```

```
ans
xx=sin(pi/3);
yy=sqrt(1-xx*xx)
```

(e) W MATLABie łatwo manipuluje się liczbami zespolonymi. Wypróbuj poniższe propozycje:

```
zz=3+4i
conj(zz)
abs(zz)
angle(zz)
real(zz)
imag(zz)
exp(sqrt(-1)*pi)
exp(j*[pi/4-pi/4])
```

(f) Rysowanie wykresów w MATLABie jest proste, zarówno dla liczb rzeczywistych, jak i zespolonych. Podstawowe polecenie tworzenia wykresu `plot(xx,yy)` kreśli `yy` uzależnione od `xx`. Sprawdź poniższe:

```
xx= [-3 -1 0 1 3];
yy= xx.*xx-3*xx;
plot(xx,yy)
zz=xx+yy*sqrt(-1);
plot(zz) %<---liczby zespolone też można wykreślać
```

Usuń średniki, jeśli chcesz wyświetlić wartości wektorów `xx`, `yy` czy `zz`. Skorzystaj z *Dodatku* lub z `help arith`, aby zorientować się, jak działa operacja `xx.*xx`; porównaj ją z mnożeniem `xx*xx` macierzy.

Jeżeli masz wątpliwości na temat polecenia, korzystaj z Dodatku lub z `help`.

0.3.2 Indeksowanie tablic i macierzy w MATLABie

(a) Upewnij się, czy rozumiesz notację dwukropkową. W szczególności wyjaśnij, co wygenerowane zostanie przez następujący kod MATLABowy:

```
jkl=2:4:17
jkl=99:-1:88
ttt=2:(1/9):4
tpi=pi*[2:(-1/9):0]
```

(b) Usuwanie i wstawianie elementów z/do wektora jest nietrudne. Zastanów się nad skutkami poniższych definicji:

```
xx=[ones(1,4),[2:2:11],zeros(1,3)]
xx(3:7)
length(xx)
xx(2:2:length(xx))
```

Objaśnij wyświetlone rezultaty trzech ostatnich linii powyższego kodu.

(c) W poprzednim punkcie wektor `xx` składał się z 12 elementów. Obejrzyj rezultat następującego przyporządkowania:

```
xx(3:7)=pi*(1:5)
```

0.3.3 Pliki skryptowe MATLABa

(a) Eksperymentuj z wektorami w MATLABie. Traktuj wektory jak listę liczb. Wypróbuj następujące zapisy:

```
kset=-3:13;
kset
cos(pi*kset/4)  %<--- komentarz: oblicza cosinus
```

Wyjaśnij, jak w ostatnim przykładzie obliczono wiele wartości cosinusa bez stosowania pętli. Tekst następujący po znaku `%` jest komentarzem i może nie być pisany. Jeżeli usuniesz średnik kończący pierwszą z instrukcji, wszystkie elementy `kset` "wysypią" się na ekran.

(b) Wykorzystaj wbudowany lub zewnętrzny edytor tekstu (np. Notatnik) w celu stworzenia pliku tekstowego nazwanego funky.m, zawierającego następujące linie:

```
tt=-2:0.05:3;
xx=sin(2*pi*0.789*tt);
plot(tt,xx), grid on %<--- rysuje sinusoidę
title('TESTOWANIE WYKRESLANIA SINUSOIDY')
xlabel('CZAS (w sekundach)')
```

W dalszej cześci tego typu pliki z ciągiem poleceń MATLABa nazywać będziemy plikami skryptowymi (MATLABa), a ich zawartość skryptem (MATLABa).

(c) Uruchom Twój skrypt z poziomu linii komend MATLABa. Aby uruchomić utworzony przez Ciebie plik funky.m spróbuj następujących poleceń:

```
funky       %<--- uruchamia polecenia z pliku funky.m
type funky  %<--- wyprowadza na ekran zawartość pliku
            %     funky.m
which funky %<--- pokazuje katalog zawierający plik
            %     funky.m
```

(d) Dodaj trzy linie kodu do Twojego skryptu tak, żeby wykreślić cosinus "na" sinusie. Użyj polecenia `hold`, aby dodać wykres funkcji

```
0.5*cos(2*pi*0.789*tt)
```

do wykresu utworzonego w punkcie (c). Skorzystaj z Dodatku lub z `help hold` MATLABa.

0.3.4 Funkcje

Poniższe przykłady są po to, by wspomnieć Tobie o łatwości pisania funkcji w MATLABie. Choć prezentowane przykłady zawierają drobne błędy, one stanowią wzór poprawnej struktury i składni dla piszących funkcje.

(a) Znajdź błąd w następującej funkcji:

```
function [tt,xx] = cosgen(f,dur)
%COSGEN Funkcja generująca falę kosinusoidalną
%     użycie:
%           [tt,xx] = cosgen(f,dur)
%           f  = pożądana częstotliwość
%           dur= czas trwania przebiegu w sekundach
%
tt = [0:1/(20*f):dur]; % daje 20 próbek na okres
yy = cos(2*pi*f*tt);
```

Dla ułatwienia podpowiadamy: "co ma yygrek do wiatraka".

(b) Popraw kod powyższej funkcji i obejrzyj rezultat jej wywołania np. następująco:

```
[tt,xx]=cosgen(10^3,5*10^-3);
plot(tt,xx);
```

0.3.5 Macierzowe operatory logiczne

Sprawdź i objaśnij działanie następujących linii kodu MATLABa:

```
A = randn(6,3);
A = A .*(A>0);
```

Na temat operatorów szukaj informacji między innymi w *Dodatku.*

0.3.6 Interfejs graficzny (czyli GUI)

Interfejsy graficzne służą do interaktywnego wykonywania obliczeń i ich wizualizacji. W ramach spotkania z prostym GUI wywołaj `graf2` (czyli wpisz w linii poleceń: `graf2` Enter). Obejrzyj oferowane w MATLABie rodzaje wykresów, klikając na kolejne przyciski ekranu, który się pojawił w wyniku tego wywołania. Skrypt innego prostego GUI, służącego obserwacji zmian kształtu paraboli przy płynnej zmianie jednego jej współczynnika, jest wypisany poniżej (takie gotowe GUI będą wykorzystywane w następnych ćwiczeniach laboratoryjnych; dla zainteresowanych załączony skrypt może być wzorcem do tworzenia własnych prostych GUI). Aby go uruchomić, należy napisać `parabola` Enter.

```
%M-plik:  parabola.m
```

```
%GUI służy do wykreślania paraboli y=x*x+b*x+1
%przy -3<x<3 i -3<b<3.
%
x=-3:0.1:3; b=0;
%Tworzenie na ekranie obiektu Figure i zapamiętanie
%jego uchwytu h_fig
h_fig=figure('NumberTitle','off','Name','Parabola');
%Tworzenie obiektu Text  (tekst statyczny)
h_text=uicontrol('style','text','units','normalized',...
                 'position',[0.025 0.8 0.21 0.1],....
                 'string',['Wykres paraboli '...
                           '  y=x*x+b*x+1   '...
                           'ze zmienianym  b']);
%Tworzenie obiektu Axes  (układ współrzędnych)
h_axes=axes('units', 'normalized',...
            'position',[0.3 0.05 0.7 0.9]);
%Tworzenie przywołania (callback), czyli łańcucha z programem
%obsługi -dokonanej przez użytkownika- edycji parametrów
parabola_clbk1=['b=str2num(get(h_b,''string''));'...
          '   y=x.*x+b*x+1            ;'...
          '   plot(x,y);    grid on     '];
%Tworzenie obiektu Edit (tekst edycyjny) - dla współczynnika b
h_b=uicontrol('style','edit','units','normalized',...
              'position',[0.08 0.72 0.1 0.05],...
              'callback',parabola_clbk1,...
              'string',num2str(b));
%Wywołanie z opcjonalnym parametrem b=0
eval(parabola_clbk1);
%Tworzenie kolejnego przywołania - tym razem dla obsługi
%zmiany parametrów dokonanej przy użyciu suwaka
parabola_clbk2=['b=get(h_slider,''value'')     ;'...
                'set(h_b,''string'',num2str(b));'...
                '  y=x.*x+b*x+1             ;'...
                '  plot(x,y); grid on         '];
%Tworzenie obiektu Slider   (suwak)
h_slider=uicontrol('style','slider','units','normalized',...
            'position',[0.005 0.63 0.24 0.05],...
            'min',-3,'max',3,'callback',parabola_clbk2);
```

0.4 KORZYSTANIE Z MATLABA - ĆWICZENIA

Należy wykonać następujące zadania. Wyniki każdego z zadań powinny zostać przekazane prowadzącemu w ramach **bardzo krótkiego** pisemnego sprawozdania.

Wygeneruj (jako wektory) dwie 3kHz sinusoidy o różnych amplitudach i fazach.

$$x_1(t)=A_1\cos(2\pi 3000t+\phi_1) \qquad x_2(t)=A_2\cos(2\pi 3000t+\phi_2)$$

(a) Wybierz następująco wartości amplitudy: A_1=13 i A_2="Twój wiek w latach". Wybierz następująco fazę (gdy jednostkami są stopnie): ϕ_1="Ostatnie dwie cyfry Twego roku urodzenia" oraz ϕ_2=-30 stopni. ***Przeprowadzając obliczenia w MATLABie, pamiętaj o zamianie jednostek na radiany.***

(b) Wykreśl oba sygnały przynajmniej w zakresie około trzech cykli (okresów). Twórz wykres tak, by na okres sygnału przypadało przynajmniej 20 próbek. Dopilnuj, by wykres zaczynał się w zakresie ujemnego czasu, a kończył w zakresie dodatniego czasu.

(c) Zweryfikuj, czy faza sygnałów $x_1(t)$ i $x_2(t)$ jest poprawna przy t=0. Sprawdź również, czy każdy z sygnałów ma poprawną amplitudę (lub maksymalną wartość).

(d) Wykorzystaj `subplot(3,1,1)` i `subplot(3,1,2)` do utworzenia trójpanelowego rysunku w jednym oknie i umieszczenia w pierwszym panelu wykresu sygnału $x_1(t)$, a w drugim panelu wykresu sygnału $x_2(t)$. Skorzystaj, w razie potrzeby z *Dodatku* lub z `help subplot`.

(e) Utwórz trzecią sinusoidę jako sumę: $x_3(t)=x_1(t)+x_2(t)$. W MATLABie oznacza to sumowanie wektorów przechowujących wartości sinusoid $x_1(t)$ i $x_2(t)$. Sporządź wykres $x_3(t)$ dla tego samego zakresu czasu jak dla poprzednich sinusoid. Włącz go jako trzeci panel okna używając polecenia `subplot(3,1,3)`.

0.5 PYTANIA UTRWALAJĄCE

1. Widziałeś, jak łatwo jest tworzyć i manipulować wektorami w MATLABie. Dla przykładu zastanów się nad:

```
yy = 0:10;
yy = zeros(1:25);
yy = 1:.25:5;
```

   (a) Jak należałoby zmodyfikować powyższe linie, aby utworzyć wektor, który kroczy od zera do 10 co ½ ?

   (b) Jak zmodyfikowałbyś jedną z linii, aby utworzyć wektor stu 100 (stu setek)?

2. Zauważyłeś, że w MATLABie nie ma trudności z operowaniem na liczbach zespolonych. Rozważ następującą linię kodu:

```
y = 3+5j;
```

   (a) Jak "wydusisz" z MATLABa moduł tej liczby zespolonej?

   (b) Co uczynisz, by MATLAB podał fazę (argument) tej liczby? W jakich jednostkach uzyskasz odpowiedź?

3. W punkcie 0.3.3 zapoznałeś się z plikami skryptowymi o rozszerzeniu .m. MATLAB wykonuje kod zawarty w takich plikach w kolejności takiej, jaka jest w pliku. Rozważ następujący plik, nazwany example.m:

```
f=200;
tt=[0:1/(20*f):1];
z=exp(j*2*pi*f*tt);
subplot(211)
plot(real(z))
title('Real part of exp(j*2*pi*f*tt)')
subplot(212)
plot(imag(z))
title('Imaginary part of exp(j*2*pi*f*tt)')
```

   (a) Jak z MATLABa uruchamiasz ciąg poleceń zawarty w pliku?

   (b) Załóżmy, że uruchamiany plik ma nazwę example.cow. Czy "ruszy" w MATLABie? Co zmieniłbyś, by ruszył?

   (c) Zakładając, że M-plik udało się uruchomić, jakich wykresów się spodziewasz? Jeżeli nie masz pewności, po prostu przepisz kod i uruchom.