PRZETWARZANIE SYGNAŁÓW – LABORATORIUM				
Ćw. 0	**MATLAB jako narzędzie w przetwarzaniu sygnałów**			
Wykonujący:	(IMIĘ NAZWISKO, nr albumu)			Punkty / Ocena
Grupa dziekańska:		Grupa laboratoryjna:		
Numer komputera:		Data i godzina wykonania ćwiczenia:		

Zadania

1. Za pomocą komendy help (p. przykład 1) zapoznaj się z opisami wybranych przez siebie działań, funkcji, skryptów.

2. Odczytaj wartość liczby pi z możliwie dużą ilością cyfr (p. przykład 2, MATLAB jako kalkulator).

3. Przećwicz działania na macierzach wzorując się na przykładzie 3.

4. Wzorem przykładu 4 zbadaj, o ile szybsze jest MATLABowe wykonanie działań na macierzach od tego samego działania wykonanego z użyciem instrukcji `for`. Przykładowo można badać szybkość obliczenia iloczynu skalarnego w zależności od długości sygnału x[1:N]. Służy temu skrypt czasilskal.m:

```
%czasilskal
%Porównanie czasów obliczenia iloczynu skalarnego <x,x>
%(mocy sygnału x[1:N]). Jeden raz obliczenia wykonano
%z użyciem matlabowej instrukcji x*x' i drugi raz z użyciem
%instrukcji for.
%Czasy obliczeń wykreślono we wspólnym układzie współrzędnych.
%Nie można dawać zbyt dużego N, bo zabraknie czasu na zajęciach,
%tak długotrwałe są obliczenia.
%Wynik silnie zależy od szybkości komputera i wersji MATLABa.
%W nowszych wersjach w zasadzie nie ma tak wielkich różnic
%(nie ma niebezpieczństwa przekroczenia czasu, a raczej pamięci),
%ale np. w starszych MATLABach 4, 5 różnica jest ogromna.
for k=1:40
   N(k)=k*5000;
   x=pi*ones(1,N(k));
   tic
   ilskal=0;
   for m=1:N(k),
      ilskal=ilskal+x(m)*x(m);
   end
   tfor(k)=toc;
   tic
   ilskal=x*x';
   tmat(k)=toc;
end
plot(N,tfor,'bo:',N,tmat,'r*:')
title('Czas obliczenia iloczynu skalarnego<x,x>:kółka-instrukcja for,gwiazdki-instrukcja matlabowa')
```

```
xlabel('Długość wektora x')
ylabel('Czas w sekundach')
clear
```

Rys. 12. Wykres czasu obliczenia iloczynu skalarnego

5. Wykreśl wybrany przez siebie sygnał lub funkcję (impuls jednostkowy, skok jednostkowy, sygnał świergotowy (chirp) $\cos(2\pi f_0 t + \pi\mu t^2)$, krzywa rezonansowa, charakterystyki częstotliwościowe i rozkłady zer i biegunów transmitancji $H(s)$, $H(z)$, itp.). Wzoruj się na przykładach 5, 6, 7, 8, 9.

6. Wygeneruj własny dźwięk pisząc własny skrypt tonszum1.m na wzór skryptu tonszum.m (przykład 10). Może to być pojedynczy ton czysty lub zaszumiony lub chirp. Ewentualnie może to być sygnał dwutonowy jak przy dwutonowym wybieraniu numeru telefonicznego. W poniższej tabeli podano pary częstotliwości tonów dla poszczególnych cyfr i znaków.

Częstotliwości	1209 Hz	1336 Hz	1477 Hz	1633 Hz
697 Hz	1	2	3	A
770 Hz	4	5	6	B
852 Hz	7	8	9	C
941 Hz	*	0	#	D

Na przykład tonowemu wybieraniu numeru telefonicznego 58 347 17 64 towarzyszy dźwięk, który usłyszymy posługując się skryptem telefon.m (posłuchaj przykładu dając >>telefon, następnie napisz własny skrypt telefon1.m):

```
% Wybieranie tonowo numeru telefonicznego
f=[697,770,852,941,1209,1336,1477];   % Zestaw częstotliwości tonów
```

```
T=0.5;   % Czas wybierania (w sekundach) jednej cyfry numeru telefonicznego
fs=8000;   % Szybkość próbkowania
t=0:1/fs:T;   % Dyskretyzacja czasu
% Dwutonowe kodowanie cyfr od 1 do 0, czyli od c1 do c0
c1=sin(2*pi*f(1)*t)+sin(2*pi*f(5)*t);
c2=sin(2*pi*f(1)*t)+sin(2*pi*f(6)*t);
c3=sin(2*pi*f(1)*t)+sin(2*pi*f(7)*t);
c4=sin(2*pi*f(2)*t)+sin(2*pi*f(5)*t);
c5=sin(2*pi*f(2)*t)+sin(2*pi*f(6)*t);
c6=sin(2*pi*f(2)*t)+sin(2*pi*f(7)*t);
c7=sin(2*pi*f(3)*t)+sin(2*pi*f(5)*t);
c8=sin(2*pi*f(3)*t)+sin(2*pi*f(6)*t);
c9=sin(2*pi*f(3)*t)+sin(2*pi*f(7)*t);
c0=sin(2*pi*f(4)*t)+sin(2*pi*f(6)*t);
% Sygnał numeru telefonicznego 58 347 17 64
x=[c5,zeros(1,300),c8,zeros(1,2000),c3,zeros(1,300),c4,zeros(1,300),c7,...
    zeros(1,2000),c1,zeros(1,300),c7,zeros(1,2000),c6,zeros(1,300),c4];
% Odsłuchanie wybieranego tonowo numeru telefonicznego
sound(x,fs)
```

Rzadko kiedy piszemy własne skrypty MATLABowe od podstaw - szkoda na to czasu. Nie inaczej jest w tym ćwiczeniu. Przekopiujmy już istniejący skrypt, zmieńmy jego nazwę i dokonajmy w nim interesujących nas zmian. Na koniec zajęć wykasujmy nasze już niepotrzebne skrypty. Wprawdzie, jeśli je pozostawimy, to nic strasznego się nie stanie, ale przeczy to dobremu obyczajowi nie zaśmiecania komputera.

7. Przetwórz cyfrowo dźwięk wzorując się na przykładach 11, 12, 13, 14. Na przykład przepuść sygnał dźwiękowy przez filtr dolnoprzepustowy o określonej odpowiedzi impulsowej:

```
>> load handel.mat
>> sound(y,Fs)   % Odsłuchanie dźwięku oryginalnego
>> h=[1,1,1]/3;   % Odpowiedź impulsowa h[n] prostego filtru dolnoprzepustowego –
>>                % uśrednianie każdych kolejnych trzech próbek
>> u=conv(y,h); % Przepuszczenie sygnału przez filtr (operacja splotu)
>> sound(u,Fs)   % Odsłuchanie przefiltrowanego dźwięku (stłumione wyższe tony)
```

7. Stwórz własny obraz, na przykład kolorową kratkę, pisząc własny skrypt kratka2.m na wzór skryptu kratka1.m (p. przykład 15).

8. Przetwórz cyfrowo obraz (p. przykłady 16, 17). Na przykład, nałóż na obraz oryginalny obraz odbity (jak na ekranie telewizora z niedopasowaną anteną), skrypt obrazodbity.m :

```
% Obraz odbity
[x]=imread('circuit.tif'); % Załadowanie obrazu do obszaru roboczego (format uint8)
[M,N]=size(x); % Wymiary obrazu, liczba wierszy i kolumn
subplot(221); imshow(x); title('Obraz oryginalny') % Podgląd oryginalnego obrazu
y=double(x); % Zmiana formatu z uint8 na double
h1=[1,zeros(1,20),1]; % Odpowiedź impulsowa pojedynczego odbicia
for k=1:M, z(k,:)=0.5*conv(h1,y(k,:)); end % Przepuszczenie sygnału obrazu przez filtr h1
subplot(222); imshow(uint8(z)); title('Pojedyncze odbicie') % Obraz z pojedynczym odbiciem
clear z   % Usunięcie zmiennej z z obszaru roboczego
h2=[1,zeros(1,20),1,zeros(1,20),1];  % Odpowiedź impulsowa podwójnego odbicia
for k=1:M, u(k,:)=0.333*conv(h2,y(k,:)); end % Przepuszczenie sygnału obrazu przez filtr h2
subplot(223); imshow(uint8(u)); title('Podwójne odbicie')% Podgląd obrazu z podwójnym %odbiciem
```

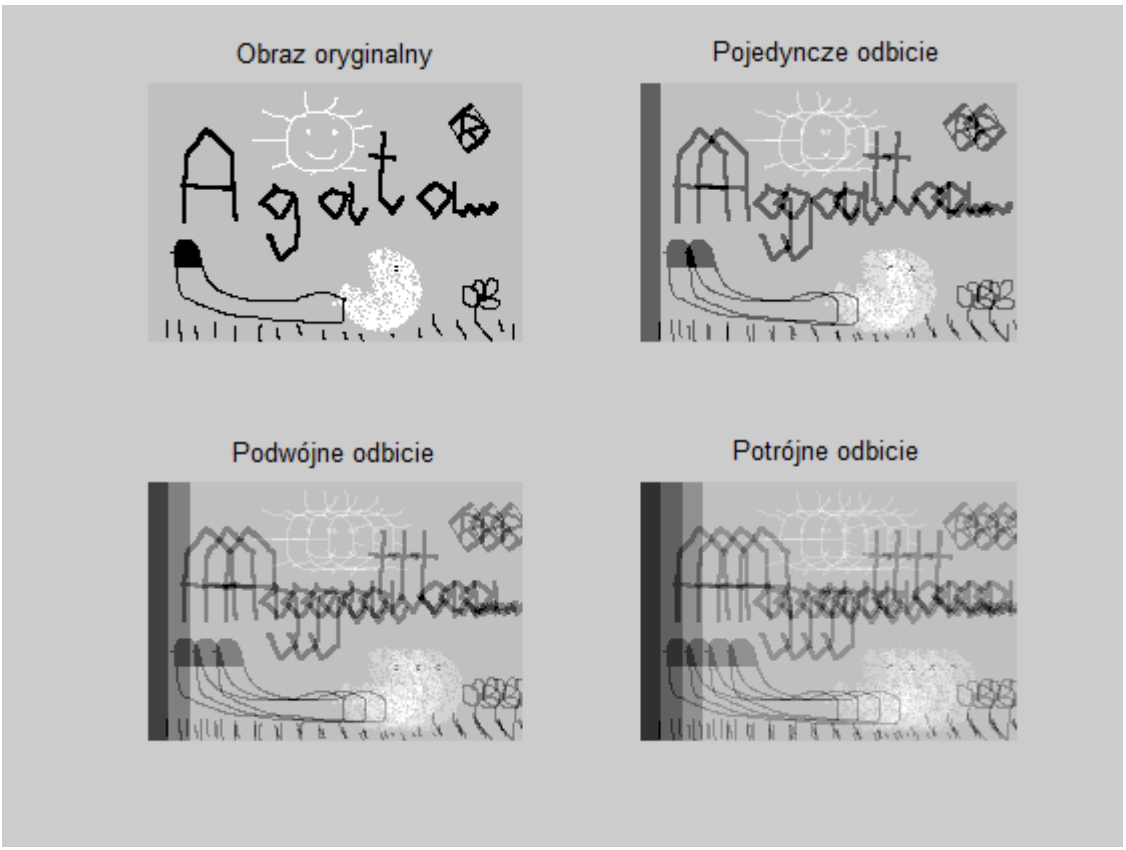
```
clear u   % Usunięcie zmiennej u
h3=[1,zeros(1,20),1,zeros(1,20),1,zeros(1,20),1]; % Odpowiedź impulsowa potrójnego dbicia
for k=1:M, v(k,:)=0.25*conv(h3,y(k,:)); end % Przepuszczenie sygnału obrazu przez filtr h3
subplot(224); imshow(uint8(v)); title('Potrójne odbicie')   % Podgląd obrazu z potrójnym
%  odbiciem
clear v x y  % Usunięcie zmiennych v, x, y
```

Inny przykład, przefiltruj sygnał obrazu przez filtr dolnoprzepustowy w celu zmniejszenia szumów (rozmycie ziarna obrazu) obrazfiltr.m :

```
[x]=imread('circuit.tif'); % Załadowanie obrazu do obszaru roboczego (format uint8)
[M,N]=size(x); % Ustalenie rozmiaru macierzy
subplot(121); imshow(x); title('Obraz oryginalny') % Podgląd oryginalnego obrazu
y=double(x); % Zmiana formatu z uint8 na double
hdp=[1,1,1,1,1]/5;   % Odpowiedź impulsowa h[n] prostego filtru dolnoprzepustowego,
                     % uśrednianie każdych kolejnych pięciu próbek
for k=1:M, v(k,:)=conv(hdp,y(k,:)); end % Przepuszczenie sygnału obrazu przez filtr hdp,
                                        % filtrowanie tylko wierszami
subplot(122); imshow(uint8(v)); title('Obraz odszumiony')
% Podgląd obrazu odszumionego, usunięcie ziarna z obrazu
clear v x y  % Usunięcie zmiennych v, x, y
```

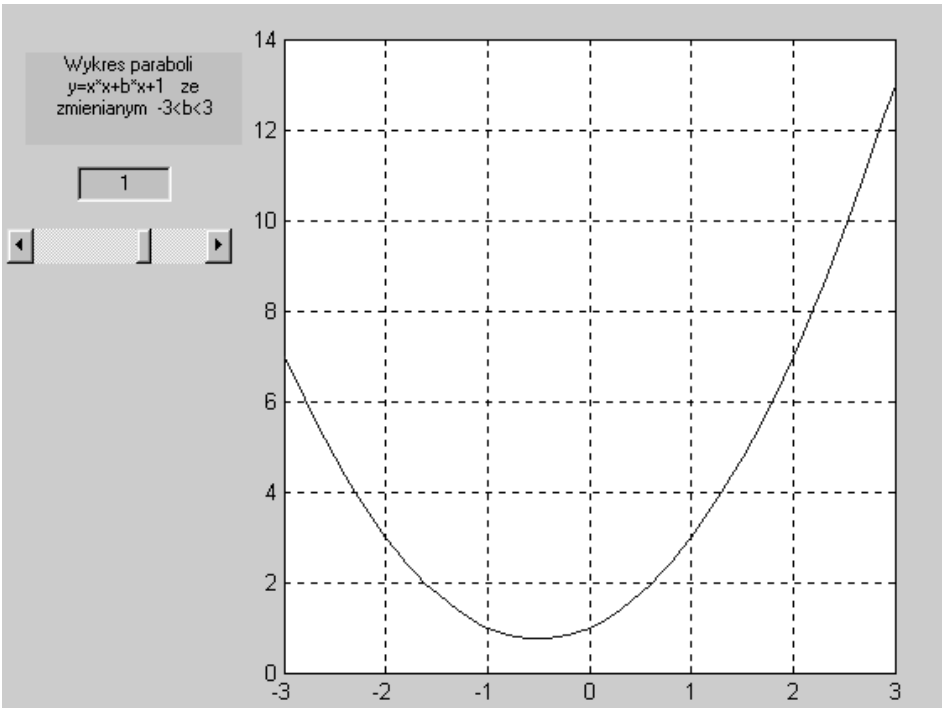
Tutaj dokonano filtracji jednowymiarowej (po liniach). Ściśle, powinno się stosować filtry dwuwymiarowe, w najprostszym przypadku należy filtrować po wierszach i po kolumnach.

Jeszcze inne przykłady przetwarzania obrazów są zawarte w skryptach obrazekbmp.m, obrazekfiltr.m, obrazekkolory.m, obrazekodbity.m, obraznotch.m.



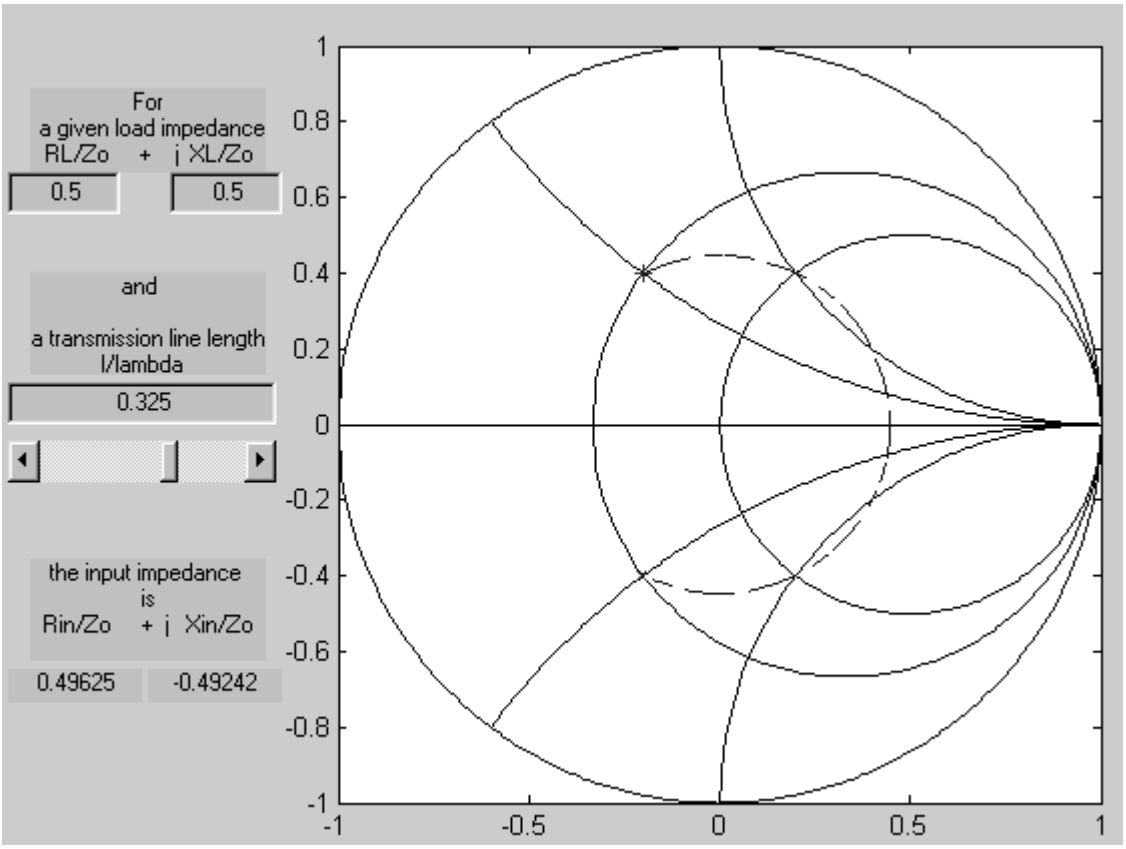
Rys. 13. Odbicia obrazu obrazekodbity.m

9. Przećwicz posługiwanie się interfejsem graficznym parabola.m (>>parabola).



Rys. 14. Zmiana kształtu paraboli w zależności od wartości współczynnika b

10. Przećwicz posługiwanie się interfejsami graficznymi SmithChart.m (>>SmithChart) i Smith.m (>>Smith).



Rys. 15. Wykres Smitha, interfejs graficzny SmithChart.m