Wykład 3

Grafika 2D

Funkcje związane z grafiką w środowisku Matlab podzielić można, przyjmując różne kryteria, na: (a) 2- lub 3-wymiarowe, (b) tworzące wykresy ciągłe i dyskretne, (c) wyświetlające grafikę rastrową i wektorową czy (d) (biorąc pod uwagę sposób programowania) funkcje wysokiego i niskiego poziomu.

Obiekty graficzne wyświetlane są w specjalnym oknie, które otwiera się poleceniem **figure**. Jednocześnie w programie może być otwartych wiele okien, każde z nich ma przypisany numer. Jedno z otwartych okien jest zawsze aktywne i do tego okna odnoszą się wszystkie polecenia wydawane w wierszu poleceń. Omówimy ważniejsze funkcje dotyczące zarządzania oknami graficznymi.

Funkcje do zarządzania oknami graficznymi

figure - tworzy nowe okno, nadając mu najmniejszy wolny numer oraz uaktywnia je

figure (n) - uaktywnia okno o numerze n (jeżeli takie istnieje) lub tworzy nowe, przyporządkowuje mu numer n i uaktywnia je

close - zamyka aktywne okno

close (n) -zamyka okno o numerze n (lub numerach, jeżeli n jest macierzą)

close all - zamyka wszystkie okna

clf - czyści aktywne okno

hold on - zachowuje bieżący widok okna graficznego, ewentualne kolejne polecenia dodają elementy do aktywnego okna (np. dodają nowe serie)

hold off - powoduje, że nowa grafika zastępuje istniejącą w aktywnym oknie

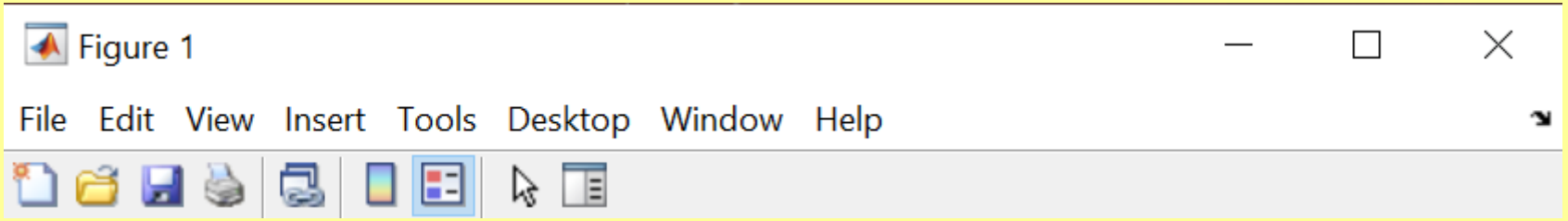
gcf - contect of figure (parametry pola na zewnątrz wykresu)

gca - contect of axes (parametry wykresu)

Wypróbować:

```
get(gcf);                    get(gca);
set(gcf,'color', 'red');     set(gca,'color','yellow');
```

Parametry wykresu można wygodnie zmieniać przy pomocy edytora i ikonek.



Wykres można **zapisać** (polecenie **save** tworzy plik z rozszerzeniem*.fig) - można go potem wczytać i edytować - lub wyeksportować (powstaje plik z rozszerzeniem *.jpg). Parametry eksportowanego pliku można zmieniać. W szczególności warto zwiększyć rozdzielczość obrazu, bo domyślna jest kiepska. Można zapisać szablon rysunku (i używać potem dla wszystkich swoich rysunków) poleceniem **generate code** – powstaje funkcja.

Wykresy funkcji podanych wzorem

Matlab daje szerokie możliwości rysowania wykresów dwu- i trójwymiarowych funkcji definiowanych za pomocą wzorów, dzięki czemu użytkownik nie musi sam definiować niektórych parametrów (np. przedziału zmienności argumentu).
Oprócz poleceń **plot, plot3** oraz **surf** (x, y, z, c), które służą do rysowania dwu- i trójwymiarowych wykresów wektorów x, y i z, Matlab umożliwia rysowanie wykresów funkcji podanych wzorem.

Podstawowym wykresem z tej rodziny jest wykres generowany przez instrukcję **ezplot**. Wykresy tego typu służą do kreślenia funkcji uwikłanych i parametrycznych. Dlatego też składnia instrukcji „ezplot" jest bardzo różnorodna. Aby narysować wykres funkcji uwikłanej, należy skorzystać z instrukcji ezplot('f(x)'), gdzie funkcję f(x) doprowadzamy wcześniej do postaci f(x) = 0.
Do instrukcji tej można dodać żądanie, aby wykres ten przebiegał w zakresach określonych dla x i y przedziałem [min max]. Instrukcja taka będzie miała składnię **ezplot('f(x)',[min max])**. Jeżeli zakresy przyjęte dla zmiennych x i y mają być różne, składnia instrukcji zmieni się na **ezplot('f(x)', [xmin xmax] [ymin ymax])**.
Przykład: >>ezplot('x.*sin(2*x)')

W powyższym przykładzie definicja funkcji, której wykres miał być rysowany, została podana jako argument w postaci łańcucha znakowego (stałej tekstowej). Nie jest to jedyny sposób przekazania definicji funkcji jako parametru wejściowego do innej funkcji.
Inne sposoby to:
- definiowanie funkcji za pomocą polecenia **inline**,
- definiowanie funkcji za pomocą m-plików funkcyjnych.

Gdy funkcja jest stosunkowo prosta jej definicję można podać „w wierszu", czyli **inline**. Ogólna postać tej definicji to:

nazwa funkcji = inline (wyrażenie)

Przykład: ezplot i plot

```
f1=inline('x.*sin(2*x)');
subplot(2,1,1); ezplot(f1,[0, pi])
x=0:pi/10:pi; y=f1(x);
subplot(2,1,2); plot(x,y)
```

Parametry poleceń **axis i grid:**

axis ([xmin xmax ymin, ymax zmin zmax]) - ustawia zakres dla wszystkich osi aktywnego wykresu w aktywnym oknie graficznym

axis equal - ustawia osie tak, aby jednostkowy odcinek był równej długości w każdym z kierunków

axis square - ustawia osie tak, aby odstępy między kolejnymi znacznikami były równej długości

axis normal - przywraca automatyczne ustawienia osi

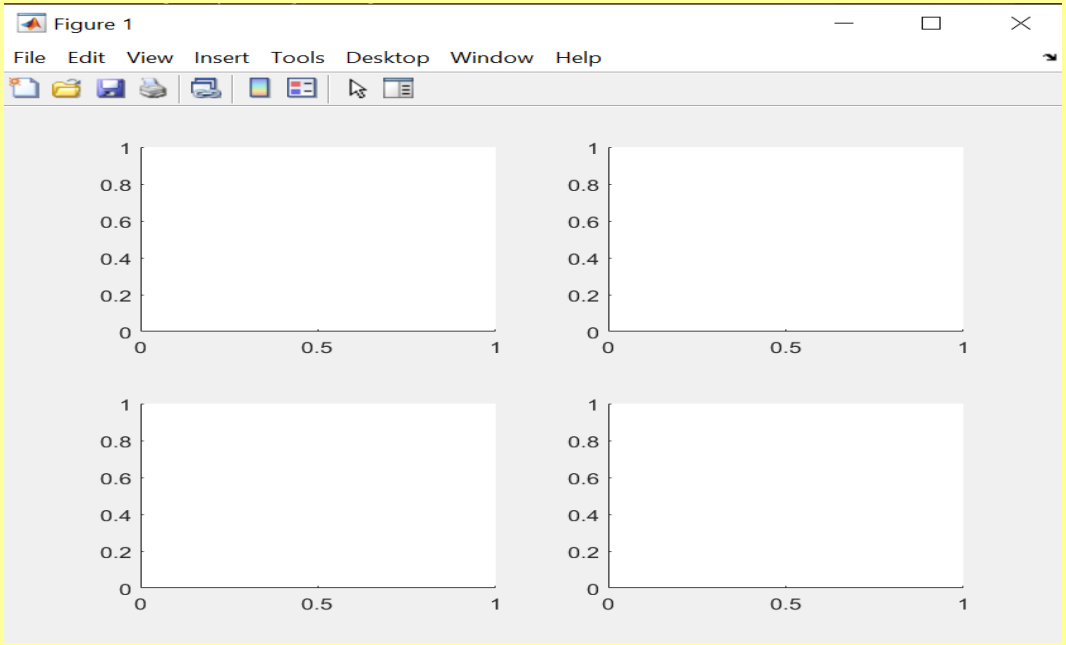
axis On/off - włącza lub wyłącza wyświetlanie osi

grid on/off - włącza lub wyłącza wyświetlanie linii siatki dla głównych znaczników

grid minor - włącza wyświetlanie dodatkowych linii siatki

W ramach jednego okna graficznego można utworzyć wiele obszarów rysowania (np. dla kilku wykresów). Służy do tego funkcja **subplot** (m, n, p). Parametry oznaczają utworzenie **m x n** podobszarów rozmieszczonych w **m** wierszach i **n** kolumnach. Podobnie jak w przypadku okien, jeden z obszarów jest zawsze aktywny; tu jest to obszar o podanym numerze p (licząc wierszami od lewej do prawej i od góry do dołu).

```
figure, subplot(2,2,1),hold on
subplot(2,2,2),subplot(2,2,3),subplot(2,2,4)
```

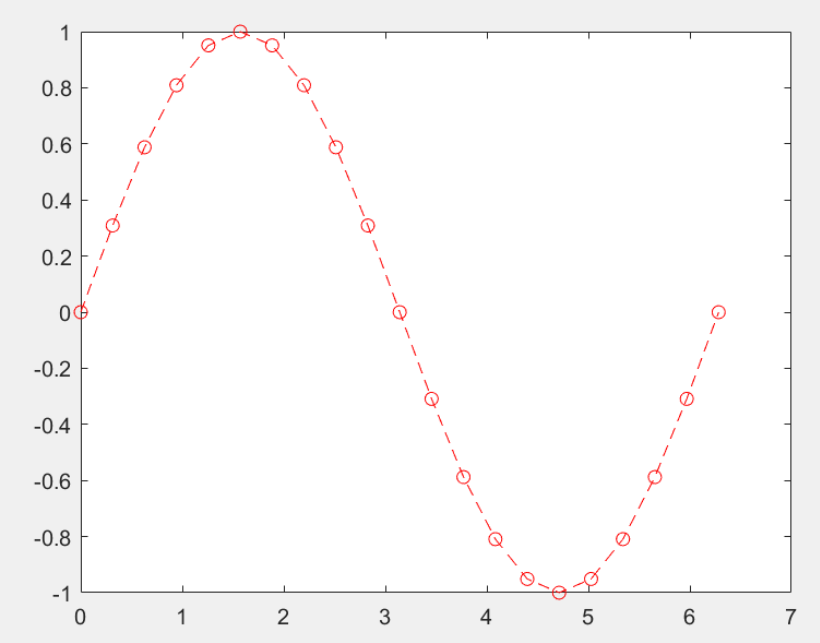


Funkcja **plot**

Funkcja plot (x, y) rysuje wykres na podstawie wartości elementów **dwóch wektorów**: x i y. Musimy zatem sami zadbać o dyskretyzację dziedziny naszych funkcji. Ważne jest, aby wektory, podawane jako argumenty funkcji plot, były tej samej długości.

Funkcja plot może być również wywołana z trzecim argumentem, który definiuje **wygląd generowanej linii**. Parametr ten jest kombinacją kilku znaków ujętych w apostrofy, które mówią o rodzaju, kolorze i znacznikach krzywej.

```
x=0:pi/10:2*pi;
plot(x,sin(x),'o--r')
```



Parametry polecenia **plot**

	Rodzaj linii		**Znaczniki**
'-'	ciągła	'.'	kropka
':'	kropkowana	'o'	kółko
'--'	kreskowana	'x'	iks
'-.'	kropka-kreska	'+'	krzyżyk
	Kolor	'*'	gwiazdka
'b'	niebieski	's'	kwadrat
'g'	zielony	'd'	romb
'r'	czerwony	'v'	trójkąt w dół
'c'	turkusowy	'^'	trójkąt w górę
'y'	żółty	'<'	trójkąt w lewo
'm'	karmazynowy	'>'	trójkąt w prawo
'y'	żółty	'p'	gwiazda pięcioramienna
'k'	czarny	'h'	gwiazda sześcioramienna

Przy wywołaniu funkcji **kolejność** symboli **nie ma znaczenia**. Nie trzeba także podawać ich wszystkich (można np. zmienić tylko kolor).

Oprócz wykresów w skalach liniowych, w Matlabie możemy również rysować wykresy w skali półlogarytmicznej i logarytmicznej.

loglog (x, y, s) - rysuje wykres z użyciem skal logarytmicznych na obu osiach,

semilogx (x, y, s) - rysuje wykres zużyciem skali logarytmicznej na osi x,

semilogy(x,y, s) - rysuje wykres zużyciem skali logarytmicznej na osi y,

linspace (xl, x2, N) - generuje wektor wierszowy N liczb rozmieszczonych równomiernie pomiędzy x1 i x2,

logspace (xl, x2, N) - generuje wektor wierszowy N liczb rozmieszczonych logarytmicznie pomiędzy 10^{x1} a 10^{x2}

Wykresy w Matlabie można opisywać dodając tytuł, opisy osi, legendę, siatkę dowolny tekst.

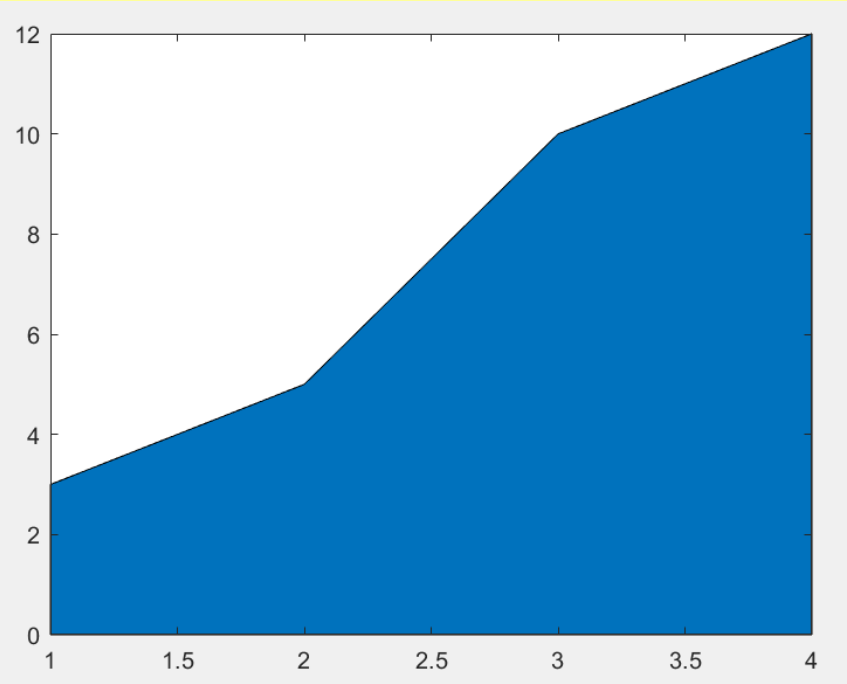
xlabel (tekst) - wyświetla tekst jako opis osi x aktywnego wykresu,
ylabel (tekst) - wyświetla tekst jako opis osi y aktywnego wykresu,
title (tekst) - wyświetla tekst jako tytuł aktywnego wykresu,
legend(s1, s2, s3...) - wyświetla legendę, s1 jest opisem dot. pierwszego wykresu, s2 jest opisem dot. drugiego wykresu, itd.,
text(x, y, tekst) - wyświetla tekst w miejscu określonym przez współrzędne
grid on/off - włącza/wyłącza wyświetlanie pomocniczej siatki współrzędnych.

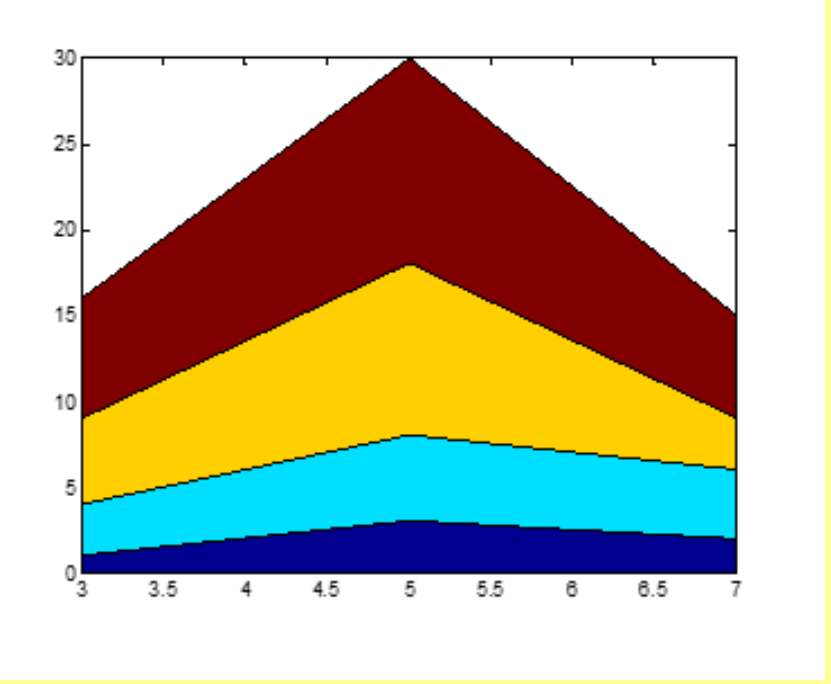
```
title('Wykres funkcji sin(x) i cos(x)');
legend('sin(x)','cos(x)');
xlabel('x-k¹t (rad)');
ylabel('sin(x),cos(x)');
grid on;
```

Wykresy danych dyskretnych

Wykres warstwowy (jedno- i dwuargumentowy)

```
y=[1 3 5 7;3 5 10 12;2 4 3 6]; x=[3 5 7];
area(x,y);
```

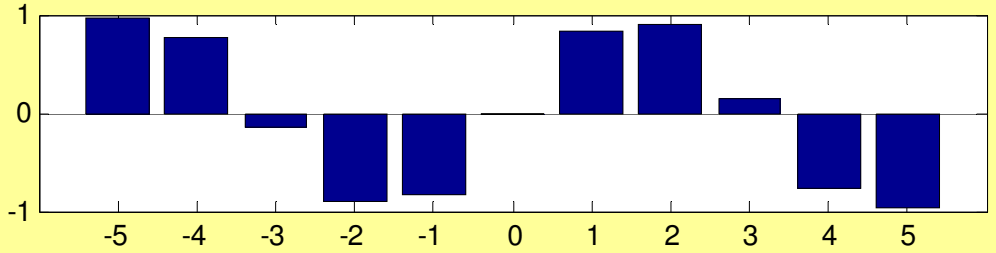




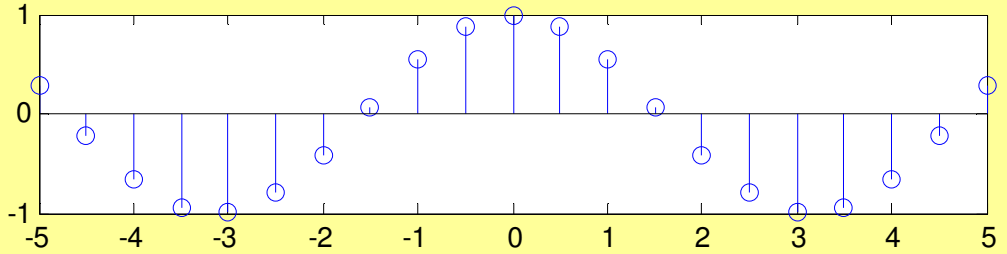
Wykresy danych dyskretnych

Wykresy: słupkowy, gałązkowy i schodkowy

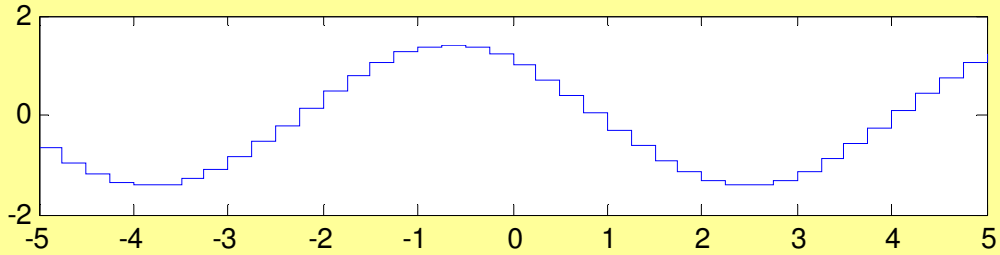
```
bar(x,y);
```




```
stem(x2,y2);
```

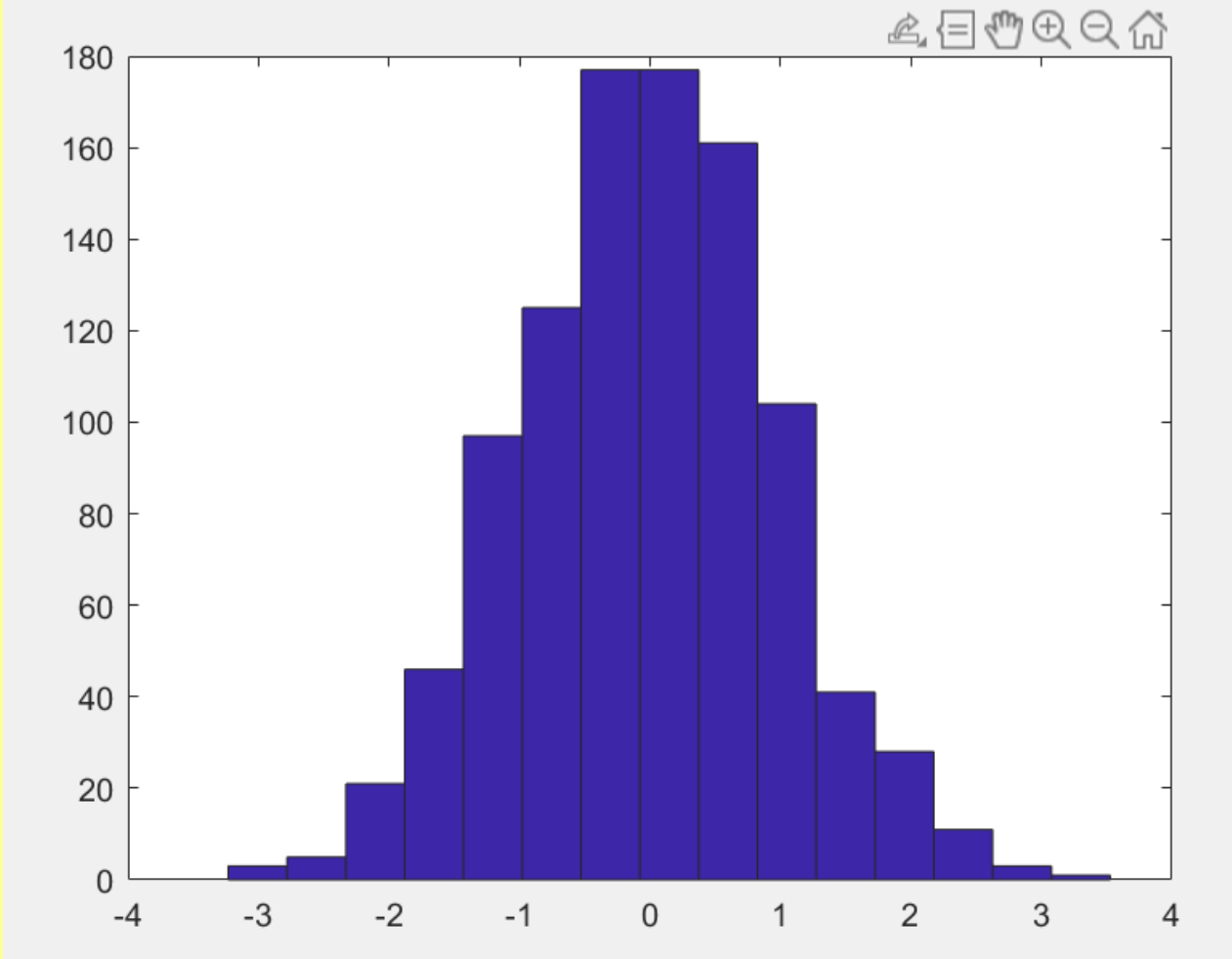



```
stairs(x3,y3);
```



Histogram

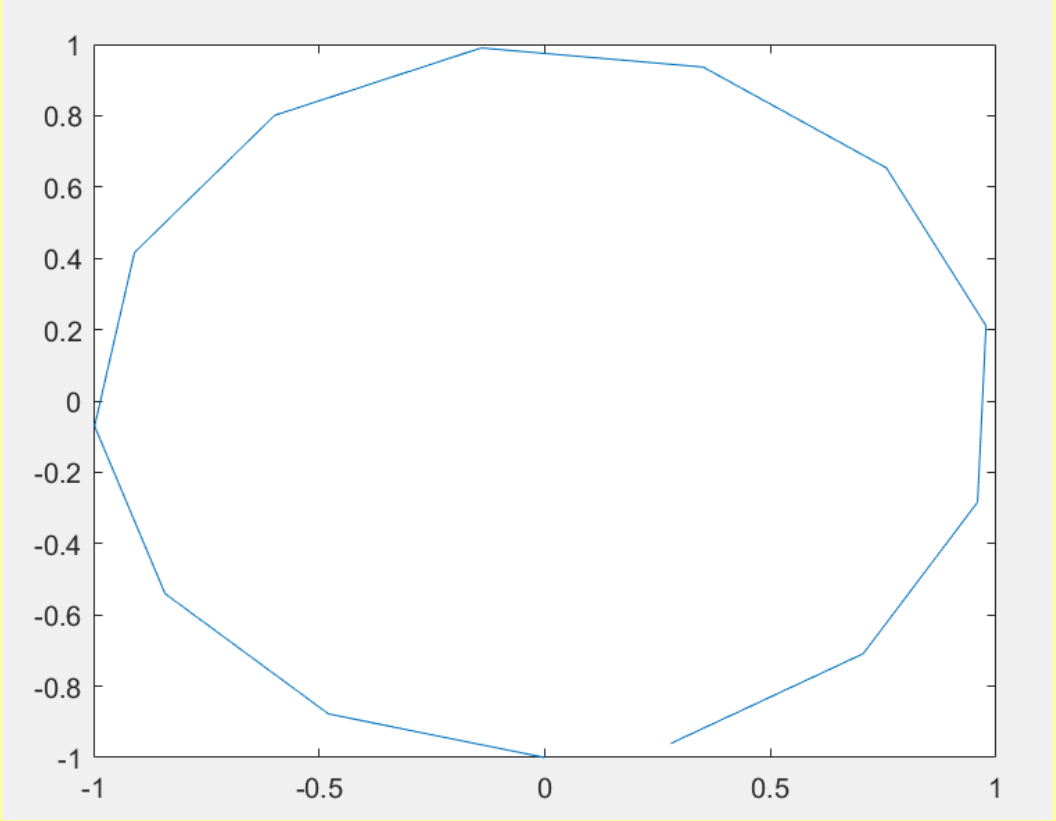
```
a=randn(1,100);
hist(a,15);
```



Wykresy na płaszczyźnie zespolonej

t= -pi:0.5:pi
x=sin(t)
y=cos(t)
z=x+i*y

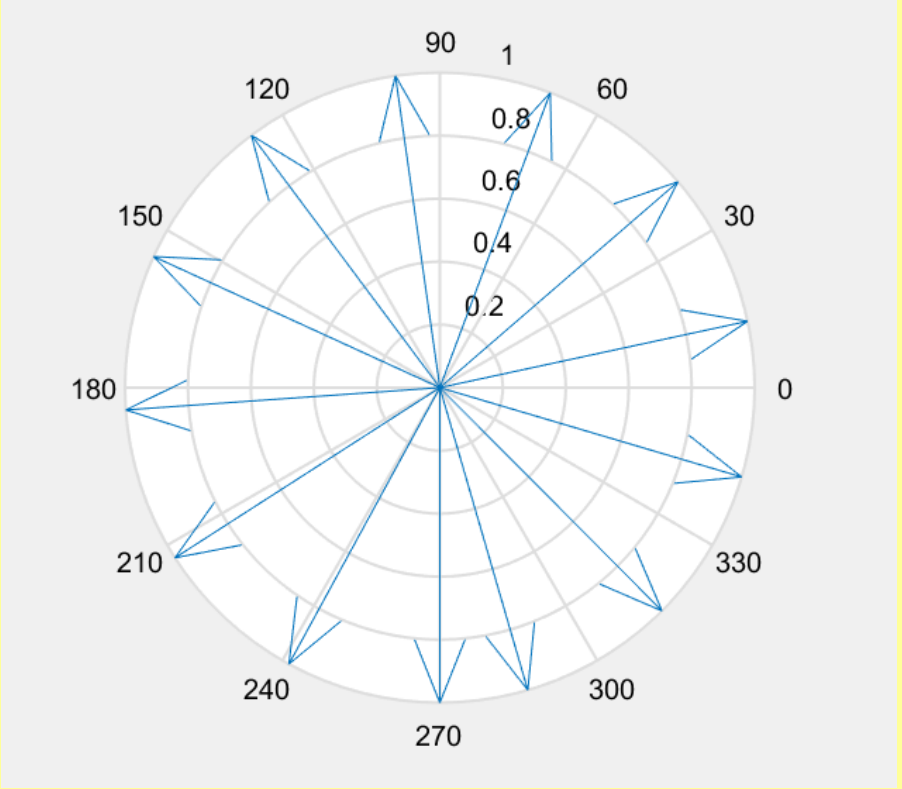
plot(z)



Wykresy na płaszczyźnie zespolonej

```
t= -pi:0.5:pi
x=sin(t)
y=cos(t)
z=x+i*y
```

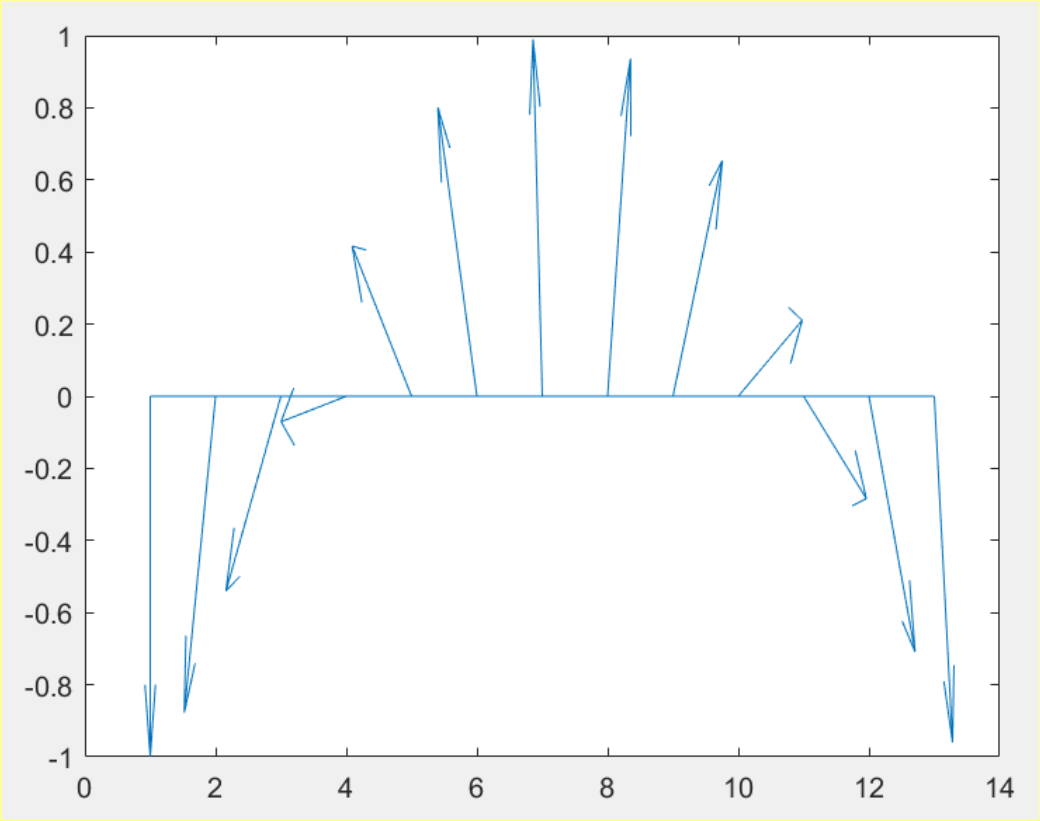
compass(z)



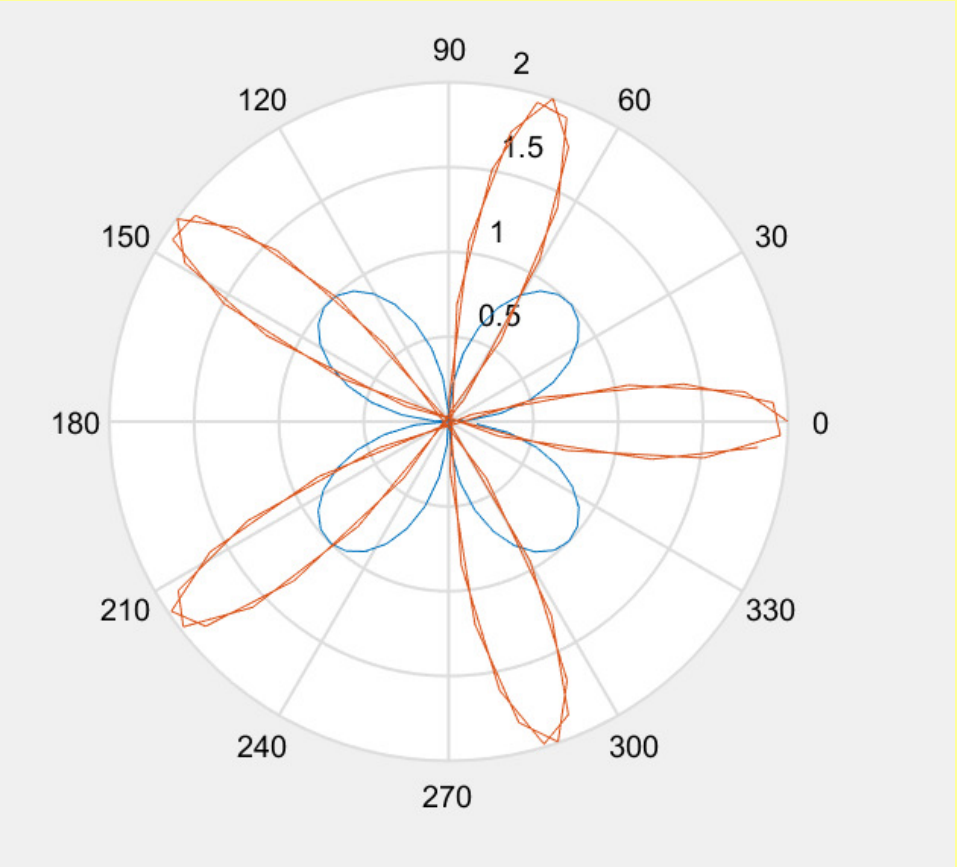
Wykresy na płaszczyźnie zespolonej

t= -pi:0.5:pi
x=sin(t)
y=cos(t)
z=x+i*y

feather(z) %pióro



Wykres w biegunowym układzie współrzędnych

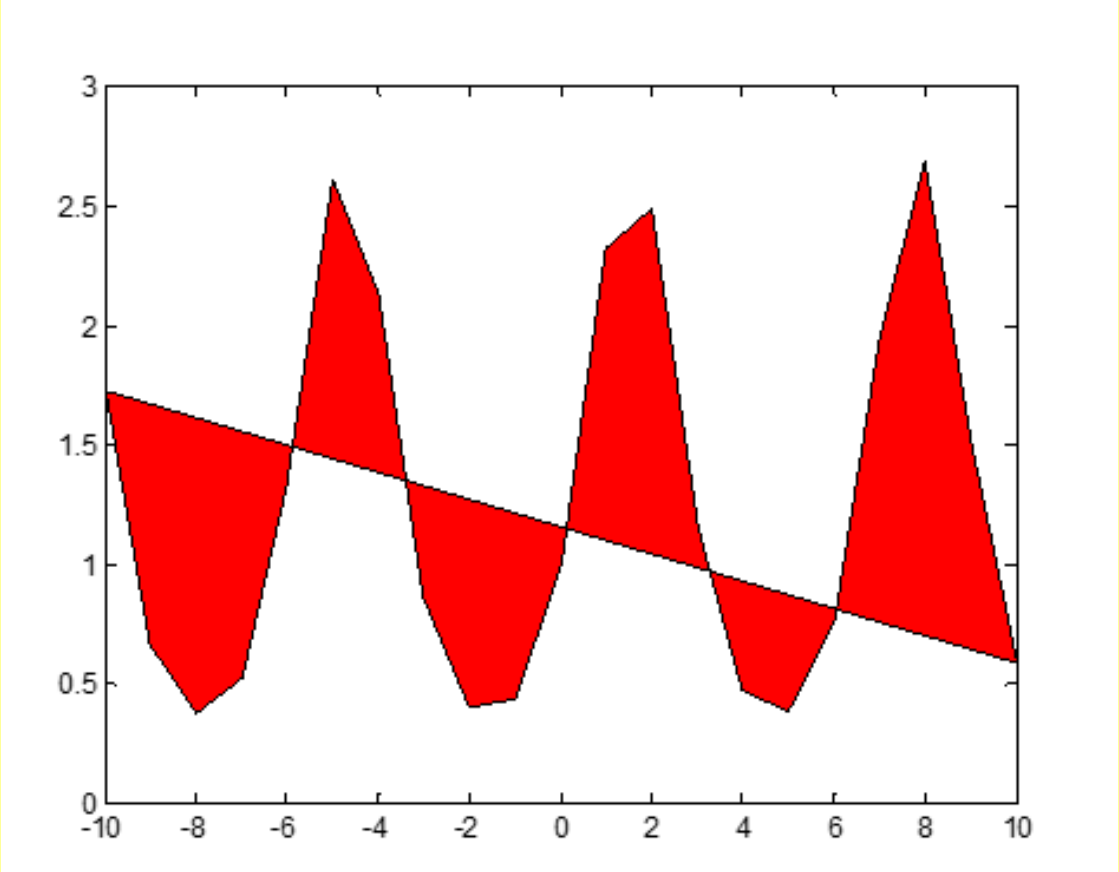



```
x=-pi:0.1:pi

polar([x;x]',[sin(2*x);2*cos(5*x)]')
```

Rysowanie linii i wypełnianie kształtów.

```
x=-10:10
y=exp(sin(x))
line(x,y)
fill(x,y,'r')
```



DZIĘKUJĘ ZA UWAGĘ