Podstawy MATLABA, cd.

1. Wielomiany

1.1. Definiowanie wielomianów

Zapiszmy przykładowy wielomian II rzędu:
$y = x^2 + 10$
można go zapisać w postaci ogólnej:
$y = \mathbf{1}x^2 + \mathbf{0}x + \mathbf{10}$
W Matlabie wpisujemy jedynie współczynniki wielomianu poczynając od najwyższej potęgi, czyli:

```
y = [1 0 10]
```

1.2. Obliczanie wartości wielomianów

Wartości wielomianu obliczamy za pomocą funkcji *polyval*. Wcześniej jednak należy zdefiniować zakres, dla którego chcemy obliczyć te wartości, np:

```
t = -10:0.1:10;
```
wektor od –10 do 10 z krokiem 0.1
```
p = polyval(y,t);
```
wektor *p* zawiera wartości wielomianu w zakresie *t*
```
plot(t,p)
```
rysowanie wielomianu

Narysuj wielomian:
$x^6 - 2x^5 + 4x^3 - 16x^2 + 15$
w zakresie $-2 \le x \le 2$

1.3. Miejsca zerowe wielomianów

Do obliczania miejsc zerowych (pierwiastków) wielomianów służy funkcja *roots*, np. dla wielomianu:
$y = x^3 – 3x^2 – 20x + 10$
należy wpisać:

```
y = [1 -3 -20 10]
z = roots(y)
```

W wyniku otrzymamy informację:

```
z =
   6.0380
  -3.5099
   0.4719
```

czyli wartości pierwiastków.

Sprawdźmy na wykresie czy są to poprawne wartości. W tym celu należy narysować ten wielomian (patrz punkt 1.2) i ustawić zakres osi na: $-4 \le x \le 7$, $-60 \le y \le 40$ za pomocą funkcji *axis*:

```
axis([-4 7 -60 40])
```

pierwsze dwie liczby oznaczają zakres osi x, a następne dwie zakres osi y.
Dla wygody można włączyć siatkę na wykresie:

```
grid
```

Po wpisaniu w linii poleceń **ginput** (kursor przybierze postać krzyża) należy wskazać myszą po kolei miejsca zerowe (wszystkie trzy) na wykresie i nacisnąć Enter:
W wyniku otrzymamy informację podobną do poniższej (zależy od dokładności trafienia w punkty):

```
ans =
  -3.5311    0.0877
   0.4482   -0.2047
   6.0495    0.0877
```

Pierwsza kolumna oznacza współrzędne *x* a druga współrzędne *y* wskazanych punktów. Sprawdź czy są zbliżone do wyników funkcji *roots*.

1.4. Obliczanie współczynników wielomianu na podstawie miejsc zerowych

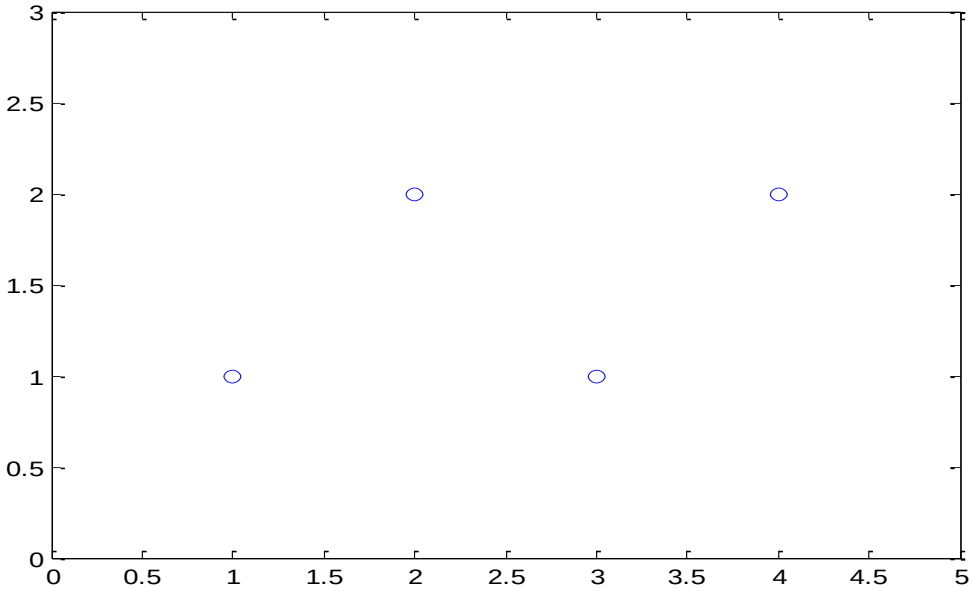
Z drugiej strony, można znaleźć współczynniki wielomianu na podstawie miejsc zerowych za pomocą funkcji *poly*:

```
y1 = poly(z)
```

```
y1 =
   1.0000   -3.0000  -20.0000   10.0000
```

1.5. Dopasowywanie wielomianu do punktów

W jaki sposób znaleźć wielomian, który przechodzi przez zadane punkty? Załóżmy, że chcemy przeprowadzić wielomian przez punkty o współrzędnych: (1,1), (2,2), (3,1), (4,2) widoczne na poniższym rysunku.



Można w tym celu wykorzystać funkcję *polyfit* o następującej składni:

```
p=polyfit([1 2 3 4],[1 2 1 2],3);
```

Funkcja ta przyjmuje 3 argumenty (dwa wektory i skalar):

- wektor współrzędnych x-owych zadanych punktów
- wektor współrzędnych y-owych zadanych punktów
- stopień wielomianu

Funkcja zwraca współczynniki znalezionego wielomianu (zobacz jakie).
Następnie na podstawie współczynników trzeba obliczyć wartości wielomianu aby go narysować. W tym celu tworzymy wektor zakresu:

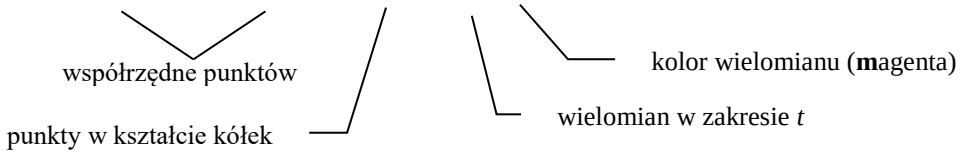
```
t=0:0.1:5;
```

oraz obliczamy wartości:

```
y=polyval(p,t);
```

i rysujemy wielomian na tle punktów:

```
plot([1 2 3 4],[1 2 1 2],'o',t,y,'m')
```



Można też nadać wykresowi tytuł:
```
title('dopasowywanie wielomianu')
```
i zakres:
```
axis([0 5 0 3])
```

2. M-pliki i m-funkcje

2.1. Skrypty

Dotychczas wpisywaliśmy wszystkie komendy w linii poleceń. W przypadku prostych obliczeń to wystarczy, ale dla bardziej skomplikowanych programów (wielu linii kodu) nie jest to wygodne. Zamiast mozolnego wpisywania linijki po linijce kodu można utworzyć tzw. m-plik czyli skrypt (program) Matlaba.
Wybierz z menu ***file → new → m-file***. Otworzy się okno edytora. Wpisz tam wszystkie komendy z punktu 1.5:

```
p=polyfit([1 2 3 4],[1 2 1 2],3);
t=0:0.1:5;
y=polyval(p,t);
plot([1 2 3 4],[1 2 1 2],'o',t,y,'m')
axis([0 5 0 3])
title('dopasowywanie wielomianu')
```

Ustaw ścieżkę na swój katalog roboczy, np. `cd f:\praca`. Zapisz powyższy program do pliku o nazwie np. **wielomian.m** (plik musi mieć koniecznie rozszerzenie **m**). Teraz możesz uruchomić ten program wpisując po prostu jego nazwę (bez rozszerzenia) w linii poleceń:

```
wielomian
```

2.2. Funkcje

Funkcje różnią się od skryptów tym, że mogą przyjmować argumenty, działają trochę szybciej i zmienne nie są widoczne w przestrzeni roboczej. Przerobimy teraz skrypt z punktu 2.1. na funkcję.
Przede wszystkim funkcja musi posiadać słowo kluczowe ***function*** zapisane w pierwszej linii kodu, po którym następuje nazwa funkcji (taka sama jak nazwa pliku, w którym jest ta funkcja zapisana).
Wybierz z menu ***file → new → m-file***. Wpisz w oknie edytora zmodyfikowany skrypt:

```
function wielomian2(n)

p=polyfit([1 2 3 4],[1 2 1 2],n);
t=0:0.1:5;
y=polyval(p,t);
plot([1 2 3 4],[1 2 1 2],'o',t,y,'m')
axis([0 5 0 3])
title('dopasowywanie wielomianu')
```

Zapisz powyższą funkcję do pliku o nazwie **wielomian2.m**. Jak widać trzeci argument funkcji *polyfit* (stopień wielomianu) nie jest jawnie podany tylko jest przekazywany poprzez argument funkcji wielomian2. Taką funkcję można wywoływać z linii poleceń z dowolnym argumentem, np.:

```
wielomian2(3)
```

lub

```
wielomian2(5)
```

3. Zapis macierzowy czy pętla „for”?

Poniższy kod pokazuje na przykładzie generowania sinusa, że wiele operacji można wykonać na różne sposoby. W szczególności operacje macierzowe zawsze można zastąpić pętlą *for*. Utwórz i wykonaj poniższy kod:

```
dt=0.1      %okres próbkowania

%macierzowo:
t1=0:dt:2*pi;
subplot(2,1,1)    %umieszcza kilka wykresów w jednym oknie
plot(t1,sin(t1)), title('Sinus generowany macierzowo')

%pętla:
for i=0:2*pi/dt
    y(i+1)=sin(i*dt);
    t2(i+1)=i*dt;
end
subplot(2,1,2)
plot(t2,y), title('Sinus generowany w petli')
```

Zwróć uwagę na składnię pętli *for* i argumenty funkcji *plot*.
Matlab jest zoptymalizowany pod względem wykonywania operacji macierzowych, więc zapis macierzowy wykonuje się dużo szybciej. Pętle są z kolei bardziej uniwersalne – nie wszystkie obliczenia można wykonać w zapisie macierzowym.
Uwaga: znak **%** oznacza komentarz – taki wiersz jest ignorowany przez program.

4. Wykresy funkcji algebraicznych

Wykonaj kilka wykresów funkcji algebraicznych za pomocą funkcji *fplot*:
$y = x^3$

```
fplot('x^3',[-10 10])
```

$y = x^3(x - 1)(x - 2)^2$

```
fplot('x^3*(x-1)*(x-2)^2',[-0.5 2.5])
```

$y = x\,e^{x} + x\,e^{-x}$

```
fplot('x*exp(x)+x*exp(-x)',[-10 10])
```

5. Wykresy trójwymiarowe

Jak wykonać wykres trójwymiarowy poniższej funkcji?
$z = x\,e^{(-x\hat{}2 - y\hat{}2)}$
w zakresie:
$-2 \le x \le 2$
$-2 \le y \le 2$
z krokiem 0.1

Najpierw należy przygotować tzw. siatkę o podanym zakresie za pomocą funkcji *meshgrid*:

```
[x,y]=meshgrid(-2:0.1:2,-2:0.1:2);
```

Następnie trzeba zapisać samą funkcję:

```
z=x.*exp(-x.^2-y.^2);
```

i użyć którejś z poniższych funkcji:

```
mesh(x,y,z)
meshc(x,y,z)
```

```
meshz(x,y,z)
surf(x,y,z)
surfl(x,y,z),shading interp,colormap(gray)
```

Oglądnij wykresy i zobacz czym się różnią.