Laboratorium 4

Obliczenia symboliczne

Będziemy korzystać z ***Symbolic Math Toolbox***.

Układy równań liniowych

Równanie liniowe ma postać:

$$a_1x_1 + a_2x_2 + a_3x_3 + \cdots + a_nx_n = b$$

lub ogólniej,

$$Ax = b.$$

Macierze

(1) **reshape**

Proszę sprawdzić, co robią polecenia.

```
>> mymat = reshape(1:16,4,4)

>> mymat_transp = reshape(1:16,4,4)'

>> mymat_niekwadratowa = reshape(1:16,2,8)
```

(2) **diag** – przekątna

```
>> diag(mymat)
```

(3) **diag** można też użyć do stworzenia macierzy diagonalnej o danej przekątnej

```
>> v = 1:4;
>> diag(v)
```

(4) **trace** – ślad macierzy

```
>> trace(mymat)
```

(5) **eye** – czyli macierz identycznościowa

```
>> eye(5)
```

(6) **triu, tril** – macierz górnotrójkątna i dolnotrójkątna.

```
>> triu(mymat)

>> tril(mymat)
```

(7) **inv** – odwrotność macierzy

```
>> a = [1 2; 2 2]

>> ainv = inv(a)

>> a*ainv
```

(8) **det** – wyznacznik macierzy

```
>> det(a)
```

Dla układu równań

$$Ax = b$$

rozwiązaniem jest

$$x = A^{-1}b.$$

```
>> A = [4 -2 1; 1 1 5; -2 3 -1];
>> b = [7;10;2];
>> x = inv(A)*b
```

Symboliczne zmienne i wyrażenia

Zmienne:

```
>> a = sym('a');
>> a + a
```

Wyrażenia:

```
>> b = sym('x^2');
>> c = sym('x^4');
```

Na takich symbolicznych wyrażeniach można wykonywać wszytskie operacje matematyczne. Proszę wypróbować poniższe przykłady:

```
>> c/b

>> b^3

>> c*b

>> b + sym('4*x^2')
```

Ale przy definiowaniu wyrażenia, nie upraszcza się ono automatycznie, tzn.:

```
>> sym('z^3 + 2*z^3')
ans =
z^3 + 2*z^3
```

Ale, jeśli z było zdefiniowane jako zmienna symboliczna, to nie potrzeba cudzysłowu wokół wyrażenia i upraszcza się ono od razu:

```
>> z = sym('z');
>> z^3 + 2*z^3
```

Jeśli chcemy od razu zdefiniować kilka zmiennych symbolicznych, używamy **syms**

```
>> syms x y z
```

To to samo, co:

```
>> x = sym('x');
>> y = sym('y');
>> z = sym('z');
```

Wielomiany

Wbudowane funkcje **sym2poly** oraz **poly2sym** służą do konwersji z wyrażeń symbolicznych na wektory i vice versa.

Na przykład:

```
>> myp = [1 2 -4 3];
>> poly2sym(myp)
ans =
x^3+2*x^2-4*x+3
>> mypoly = [2 0 -1 0 5];
>> poly2sym(mypoly)
ans =
2*x^4-x^2+5
>> sym2poly(ans)
ans =
2        0        -1        0        5
```

Upraszczanie wyrażeń

Jest bardzo fajne! Proszę się przekonać:

```
>> x = sym('x');
>> myexpr = cos(x)^2 + sin(x)^2

>> simplify(myexpr)
```

Funkcje **collect**, **expand**, **factor**.

```
>> x = sym('x');
>> collect(x^2 + 4*x^3 + 3*x^2)

>> expand((x+2)*(x-1))
ans =
x^2+x-2
>> factor(x^3 + 3*x^2 + 9*x + 27)
```

Podstawianie wartości do zmiennej – **subs**

```
>> myexp = x^3 + 3*x^2 - 2
```

```
>> subs(myexp,3)
```

Jeśli w wyrażeniu jest wiele zmiennych, zostanie ona wybrana domyślnie. Można też określić, do któ©ej zmiennej chcemy podstawiać.

```
>> syms a b x
>> varexp = a*x^2 + b*x;
>> subs(varexp,3)
ans =
9*a+3*b
>> subs(varexp,'a',3)
```

Ułamki

sym zachowuje pierwotną postać ułamka:

```
>> 1/3 + 1/2
```

```
>> sym(1/3 + 1/2)
```

```
>> double(ans)
```

numden rozbija ułamek na licznik i mianownik

```
>> [n, d] = numden(sym(1/3 + 1/2))
```

```
>> x = sym('x');
>> [n, d] = numden((x^3 + x^2)/x)
```

Wyświetlanie - **pretty**

```
>> b = sym('x^2')
```

```
>> pretty(b)
```

Wykresy

ezplot rysuje wykres z dziedziną $[-2\pi, 2\pi]$ funkcji podanej w nawiasie.

```
>> ezplot('x^3 + 3*x^2 - 2')
```

Ale dziedzinę można też określić samemu:

```
>> ezplot('cos(x)',[0 pi])
```

Funkcja **solve**

Podajemy równanie, które chcemy rozwiązać.

```
>> solve('3*x^2 + x')
```

Gdy jest wiele zmiennych, Matlab sam wybiera, względem której będzie rozwiązywać. Priorytetem jest x.

```
>>solve('a*x^2 + b*x')
```

Możemy też sami podać niewiadomą.

```
>> solve('a*x^2 + b*x','b')
```

Tyle samo równań, co niewiadomych:

```
>> rozwiazanie = solve('4*x-2*y+z=7','x+y+5*z=10','-2*x+3*
   y-z=2')
```

Aby dostać się do konkretnych składowych, dajemy kropkę:

```
>> x = rozwiazanie.x

>> y = rozwiazanie.y

>> z = rozwiazanie.z
```

double konwertuje symbole do wektora liczb:

```
>> double([x y z])
```

Zadania

(1) O funkcjach w Matlabie: `https://www.mathworks.com/help/matlab/ref/function.html`
(2) Napisz funkcję, która sprawdza, czy podana macierz jest kwadratowa i zwraca **true** lub **false**.
(3) Napisz funkcję, która przyjmuje liczbę n i zwraca górnotrójkątną macierz losowych liczb całkowitych wymiaru n.
(4) W pewnym obwodzie mamy napięcia V_1, V_2, V_3. Utwórz z poniższych równań macierz i rozwiąż je, używając: $x = A^{-1}b$.

$$V_1 = 5$$

$$-6V_1 + 10V_2 - 3V_3 = 0$$
$$-V2 + 51V_3 = 0$$

(5) Dla poniższego układu wyrysuj proste i znajdź ich przecięcie – rozwiązanie układu.

$$-3x_1 + x_2 = 2$$
$$-6x_1 + 2x_2 = 4$$

Użyj **solve**, by znaleźć rozwiązanie.
Znajdź wyznacznik macierzy układu.
Ile jest rozwiązań?

(6) Zapisz układ w symbolicznej postaci i rozwiąć go funkcją **solve**.

$$
\begin{gathered}
2x_1 + 2x_2 + x_3 = 2 \\
x_2 + 2x_3 = 1 \\
x_1 + x_2 + 3x_3 = 3
\end{gathered}
$$

Mając rozwiązanie w formie symbolicznej, utwórz wektor liczbowy, używając **double**.

(7) Przeczytaj:

`https://www.mathworks.com/help/symbolic/solve-a-single-differential-equatio`
`html?requestedDomain=www.mathworks.com`

Paulina Pełszyńska