МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Державний ВНЗ “НАЦІОНАЛЬНИЙ ГІРНИЧИЙ УНІВЕРСИТЕТ”

Кафедра електропривода

МЕТОДИЧНІ ВКАЗІВКИ

до практичних занять

з дисципліни

“Оптимізація електромеханічних систем”

для студентів спеціальності 8.05070204
“Електромеханічні системи автоматизації та електропривод”

Склав проф. М.М.Казачковський

Дніпропетровськ

2013

ЗМІСТ

.

ПЕРЕДМОВА

Даний посібник призначений для засвоєння принципів роботи з середовищем MATLAB. Він містить у собі приклади розв’язання задач, пов’язаних з формуванням масивів різного роду, роботою з елементами масивів, графічними можливостями пакету, принципами програмування на М-мові, обміном інформацією між SIMULINK та MATLAB, пошуком оптимальних рішень.

1 ФОРМУВАННЯ МАСИВІВ ТА РОБОТА З НИМИ

1.1 Формування масивів

ОДНОВИМІРНІ МАСИВИ (ВЕКТОРИ)

За допомогою конкатенації

Вектор-рядок:

```
» A=[2 3 56 12]
A =
    2    3    56    12
» F=[3  sqrt(7)  exp(4)  3/4+2]
F =
   3.0000    2.6458   54.5982    2.7500
» a=4; G=[a  3  a*2]
G =
    4    3    8
```

Вектор-стовпчик:

```
» B=[1; 3; 5; 78; 12]
B =
    1
    3
    5
   78
   12
» C=[2
3
5
1.3]
C =
   2.0000
   3.0000
   5.0000
   1.3000
```

За допомогою індексації

```
» aa(1)=2; aa(2)=4.5; aa(3)=-6
aa =
   2.0000    4.5000   -6.0000

»  as(1)=9; as(2, :)=5; as(3, :)=6.7
as =
   9.0000
   5.0000
   6.7000
```

Заданням діапазону

```
» P=5: 2:15
P =
    5    7    9    11    13    15
» P=-2: 4
P =
   -2   -1    0    1    2    3    4
```

Вектор-рядок зі 100 елементів, рівномірно розподілених у діапазоні 2...35:

```
» S=linspace(2, 35)
```

Вектор-рядок з 4 елементів, рівномірно розподілених у діапазоні 2...5:

```
» S=linspace(2, 5, 4)
S =
    2    3    4    5
```

ДВОВИМІРНІ МАСИВИ (МАТРИЦІ)

За допомогою конкатенації

```
» D=[1  4  23; 12  56  78]
D =
    1    4   23
   12   56   78

» E=[23  78  12
45  2  89]
E =
   23   78   12
   45    2   89
» DE1=[D E; E D]
DE1 =
    1    4   23   23   78   12
   12   56   78   45    2   89
   23   78   12    1    4   23
   45    2   89   12   56   78
» repmat(D,3,2)
ans =
    1    4   23    1    4   23
   12   56   78   12   56   78
    1    4   23    1    4   23
```

```
   12    56    78    12    56    78
    1     4    23     1     4    23
   12    56    78    12    56    78
```

За допомогою індексації

```
» D5(2,3)=7
D5 =
     0     0     0
     0     0     7
» D5(1,1)=5;D5(1,2)=3;D5(2,1)=0.6
D5 =
    5.0000    3.0000         0
    0.6000         0    7.0000
```

За допомогою функцій

Матриця нулів

```
» H=zeros(3,4)
H =
     0     0     0     0
     0     0     0     0
     0     0     0     0
» L=zeros(size(D))
L =
     0     0     0
     0     0     0
```

Матриця одиниць:

```
» K=ones(4, 3)
K =
     1     1     1
     1     1     1
     1     1     1
     1     1     1
» L1=eye(5,3)
L1=
     1     0     0
     0     1     0
     0     0     1
     0     0     0
     0     0     0
```

Матриця 3×4 з випадкових чисел, рівномірно розподілених у діапазоні 0...1:

```
» M=rand(3, 4)
M =
    0.4057    0.4103    0.3529    0.1389
    0.9355    0.8936    0.8132    0.2028
    0.9169    0.0579    0.0099    0.1987
```

Матриця 3×2 з випадкових чисел, розподілених за нормальним законом з μ=0, s=1:

```
Srn=randn(3, 2)
Srn =
   -0.2340    1.4435
    0.1184   -0.3510
    0.3148    0.6232
```

Те ж саме з μ=3,5, s=2, розміром 4×3:

```
Sn=normrnd(3.5, 2, 4, 3)
Sn =
    2.3809    4.6379   -0.9046
    4.3873    1.8566    5.4727
    1.6002    2.9688    2.4627
    5.0624    1.1244    4.1547
```

Магічна матриця 4×4

```
» N=magic(4)
N =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

Порожній масив:

```
» C=[ ]
C =
     [ ]
```

1.2 Розміри та розмірність масивів

Розмір матриці:

```
»  size(L1)
ans =
     5     3
```

```
»  size(B)
ans =
     5     1
```

Довжина вектора:

```
» length(B)
ans =
    5
```

Розмірність масивів:

```
»  ndims(B)
ans =
    2
» ndims(L1)
ans =
    2

» size(C)
ans =
    0     0
» ndims(C)
ans =
    2
```

1.3 Робота з елементами масивів

Елемент №4 матриці M:

```
» q=M(4)
q =
   0.4103
```

Останній елемент матриці M:

```
» M(end)
ans =
   0.1987
```

Елементи матриці M від третього до останнього як вектор-рядок:

```
» M(3:end)
ans =
  Columns 1 through 6
    0.9169    0.4103    0.8936    0.0579
0.3529    0.8132
  Columns 7 through 10
    0.0099    0.1389    0.2028    0.1987
```

Другий стовпчик матриці M:

```
» q=M(:, 2)
q =
   0.4103
   0.8936
   0.0579
```

Третій рядок матриці M:

```
» q=M(3, :)
q =
   0.9169    0.0579    0.0099    0.1987
```

Усі елементи матриці E як вектор-стовпчик:

```
» q=E(:)
q =
   23
   45
   78
    2
   12
   89
```

Елементи 2...5 матриці E як вектор-рядок:

```
» q=E(2:5)
q =
   45    78     2    12
```

Стовпчики 1...3 матриці M:

```
» q=M(:, 1:3)
q =
   0.4057    0.4103    0.3529
   0.9355    0.8936    0.8132
   0.9169    0.0579    0.0099
```

Рядки 2, 3 матриці M:

```
» q=M(2:3, :)
q =
   0.9355    0.8936    0.8132    0.2028
   0.9169    0.0579    0.0099    0.1987
```

Об'єднання матриць:

```
» R=[E  L; L  E]
R =
   23   78   12    0    0    0
   45    2   89    0    0    0
    0    0    0   23   78   12
    0    0    0   45    2   89
```

Перевизначення елементу матриці R

```
» R(2)=-34
R =
   23   78   12    0    0    0
```

```
   -34    2    89     0     0     0
    0     0    0     23    78    12
    0     0    0     45    2     89
```

```
» R(1,4)=44
R =
   23   78   12    44     0     0
  -34    2    89    0     0     0
    0     0    0     23    78    12
    0     0    0     45    2     89
```

Перевизначення першого стовпчика матриці R:

```
» R(:, 1)=[1; 2; 3; 4]
R =
    1    78   12    44     0     0
    2    2    89    0     0     0
    3    0    0     23    78    12
    4    0    0     45    2     89
```

Знищення п'ятого стовпчика матриці R:

```
» R(:,5)=[ ]
R =
    1    78   12    44    0
    2    2    89    0     0
    3    0    0     23    12
    4    0    0     45    89
```

Вибір та переставляння стовпців матриці:

```
» V3=R(:,[3 1 2 4])
V3 =
   12     1    78    44
   89     2     2     0
    0     3     0    23
    0     4     0    45
```

Вибір та переставляння елементів масива:

```
» V2=B([2 1 4 5 5])
V2 =
    3
    1
   78
   12
   12
```

Те ж за допомогою матриці індексів I:

```
» I=[1 3 4;5 7 12;16 10 1;6 8 3]
I =
    1     3     4
    5     7    12
   16    10     1
    6     8     3
» V6=V3(I)
V6 =
   12     0     0
    1     3     0
   45     2    12
    2     4     0
```

Пошук ненульових елементів вектора

```
» U=[2 0 4 1 0];
» INZ=find(U)
INZ =
    1     3     4
```

Пошук ненульових елементів матриці

```
m1=[-8 4 0; 5 0 2; 0 0 4];
[i, j, v]=find(m1)
i =
    1
    2
    1
    2
    3
j =
    1
    1
    2
    3
    3
v =
   -8
    5
    4
    2
    4
```

Пошук елементів вектора, які відповідають умові

```
» INU=find(U>1)
INU =
    1     3
```

Пошук елементів матриці, які відповідають умові

```
[inu1, inu2]=find(m1==0)
inu1 =
    3
    2
    3
    1
inu2 =
    1
    2
    2
    3
```

1.4 Табличні операції на елементах матриці

```
» E=[23  78  12
45  2  89]
» S=E.*1.5
S =
   34.5000  117.0000   18.0000
   67.5000    3.0000   133.5000
» 2.*E
ans =
    46   156    24
    90     4    178
» E.^2
ans =
        529        6084         144
       2025           4        7921
» E+2
ans =
    25    80    14
    47     4    91
» Q=sin(E)
Q =
   -0.8462    0.5140   -0.5366
    0.8509 0.9093    0.8601

» E./ Q
ans =
  -27.1797  151.7573  -22.3642
   52.8850    2.1995   103.4800
» E+Q
ans =
   22.1538   78.5140   11.4634
   45.8509    2.9093    89.8601
» E.*Q
ans =
  -19.4631   40.0903   -6.4389
   38.2907    1.8186   76.5462
```

Сума елементів стовпців

```
» sum(E)
ans =
    68    80   101
```

Мінімальний або мінімальний елемент стовпців

```
» [ME, ii]=min(E)
ME =
    23     2    12
ii =
     1     2     1
```

Сортування елементів масиву

```
» [Y, is]=sort(E)
Y =
    23     2    12
    45    78    89
is =
     1     2     1
     2     1     2
» sort(E,1)
ans =
    23     2    12
    45    78    89
» sort(E, 2)
ans =
    12    23    78
     2    45    89
```

Піднесення скаляра до матричного степеня:

```
» 2.^Q
ans =
    0.5562    1.4280    0.6894
    1.8036 1.8781    1.8152
```

Добуток елементів стовпців матриці:

```
» PRE=prod(E)
PRE =
        1035         156        1068
```

Добуток елементів вектора

```
» PRPR=prod(PRE)
PRPR =
   172439280
```

1.5 Матричні операції на елементах матриці

Транспонування матриць:

```
» X=Q'
X =
   -0.8462    0.8509
    0.5140    0.9093
   -0.5366    0.8601
```

Множення та ділення матриці на скаляр:

```
» X*2
ans =
   -1.6924    1.7018
    1.0280    1.8186
   -1.0732    1.7202
```

Множення матриць:

$$\underset{m\times r}{C} = \underset{m\times n}{A} * \underset{n\times r}{B} \text{ або } c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

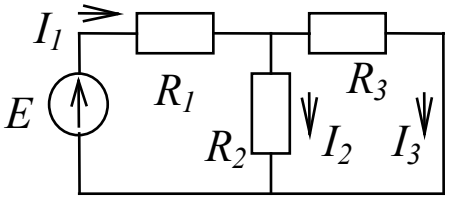
$$\begin{vmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \\ c_{31} & c_{32} \\ c_{41} & c_{42} \end{vmatrix} = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{vmatrix} \times \begin{vmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{vmatrix}; \quad \text{(m=4; n=3; r=2)}$$

$$c_{11} = a_{11}b_{11} + a_{12}b_{21} + a_{13}b_{31};$$
$$c_{12} = a_{11}b_{12} + a_{12}b_{22} + a_{13}b_{32};$$
$$c_{21} = a_{21}b_{11} + a_{22}b_{21} + a_{23}b_{31};$$
$$c_{22} = a_{21}b_{12} + a_{22}b_{22} + a_{23}b_{32};$$
$$c_{31} = a_{31}b_{11} + a_{32}b_{21} + a_{33}b_{31};$$
$$c_{32} = a_{31}b_{12} + a_{32}b_{22} + a_{33}b_{32};$$
$$c_{41} = a_{41}b_{11} + a_{42}b_{21} + a_{43}b_{31};$$
$$c_{42} = a_{41}b_{12} + a_{42}b_{22} + a_{43}b_{32}.$$

Ліве ділення матриць:

B=C \ A – розв’язок матричної системи алгебраїчних рівнянь C=A*B

Приклад. Розрахунок струмів у розгалуженому колі:



$$I_1R_1 + I_2R_2 = E;$$
$$I_1R_1 + I_3R_3 = E;$$
$$-I_1 + I_2 + I_3 = 0 \text{ або}$$

$$I = [I_1\ I_2\ I_3]^T;\ R = \begin{vmatrix} R_1 & R_2 & 0 \\ R_1 & 0 & R_3 \\ -1 & 1 & 1 \end{vmatrix};\ U = [E \quad E \quad 0]^T;\ U = R \times I;$$

$I = R \backslash U$

якщо E=12, $R_1 = 1;\ R_2 = 2;\ R_3 = 3$, то

```
R =
     1     2     0
     1     0     3
    -1     1     1
U =
    12
    12
     0
» I=R\U
I =
    5.4545
    3.2727
    2.1818
```

Піднесення до степені (тільки квадратні матриці):

```
» s=magic(4)
s =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
» s^2  % (або s*s)
ans =
   345   257   281   273
   257   313   305   281
   281   305   313   257
   273   281   257   345
```

Визначник матриці:

```
» det(R)
ans =
   -11
```

Векторний добуток векторів та матриць:

```
» V7=[1 3 5]; V8=[2 4 7]; cross(V7,V8)
ans =
     1     3    -2
» ff=magic(3);
» fy=[1 2 5; 3 6 9;2 0 5];
CR=cross(ff, fy)
CR =
    -6   -54    17
   -12    18   -20
    21    -4    19
```

Скалярний добуток векторів:

```
» sum(V7.*V8)
ans =
    49
```

1.6 Робота з комплексними числами

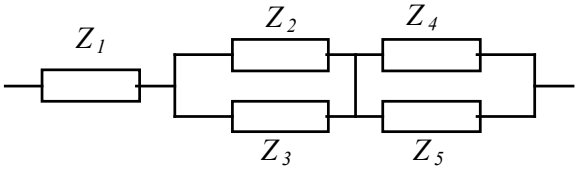
Введення комплексних чисел та операції з ними:

```
» a=4-i*5
a =
   4.0000 - 5.0000i
ka=6.*exp(i.*pi/3)
ka =
   3.0000 + 5.1962i
» b=j*6+4
b =
   4.0000 + 6.0000i
» c=a+b
c =
   8.0000 + 1.0000i
» d=a*b
d =
  46.0000 + 4.0000i
» e=a/b
e =
  -0.2692 - 0.8462i
» f=a.^b
f =
 1.0654e+005 +3.4751e+005i
» g=a*5
g =
  20.0000 -25.0000i
» s=magic(2); k=s'; r =s+k*i
r =
   1.0000 + 1.0000i   3.0000 + 4.0000i
   4.0000 + 3.0000i   2.0000 + 2.0000i
» am=abs(a)
am =
    6.4031
» rm=real(r)
rm =
     1     3
     4     2
» aalf=angle(a)
aalf =
   -0.8961
» ar=real(a)
ar =
     4
» ai=imag(a)
ai =
    -5
```

Комплексно-спряжене число

```
» acon=conj(a)
acon =
   4.0000 + 5.0000i
```

Послідовно-паралельне з'єднання комплексних опорів:



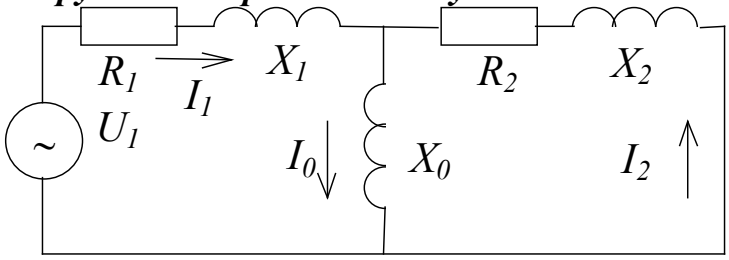

```
» z1=2+i*3; z2=4; z3=5-i*6; z4 =i; z5=6+8*i;

» z=z1+z2*z3/(z2+z3)+z5*z4/(z5+z4)
z =
   4.8205 + 3.1026i
» Z=abs(z)
Z =
    5.7326
» cosfi=real(z)./Z
cosfi =
    0.8409
```

Побудова годографа узагальненого вектора струму:

```
» Ifm=[5; 6; 7]; FI=[0 2*pi/3 4*pi/3];
» theta=linspace(0,2*pi,100);
» thetaa=(theta+FI(1))'; thetab=(theta+FI(2))'; thetac=(theta+FI(3))';
» TH=[thetaa thetab thetac];
» SS=sin(TH);
» IA=Ifm(1).*SS(:,1); IB=Ifm(2).*SS(:,2); IC=Ifm(3).*SS(:,3);
» IV=2/3*(IA+IB.*exp(2/3*pi*i)+IC.*exp(4/3*pi*i));
» plot(real(IV),imag(IV));
» figure
» plot(thetaa,IA,thetaa,IB,thetaa,IC)
```

Струми асинхронного двигуна




```
» s=0.05;
» r1=0.1; x1=0.5; r2=0.15; x2=0.6; x0=2;
» Z=[r1+i*x1   -r2./s-i*x2   0;  r1+i*x1   0  i*x0; 1   1   -1];
» U1=220; U=[U1; U1; 0];
» I=Z\U
I =
  1.0e+002 *
   0.7028  - 1.2348i
  -0.8168  + 0.4786i
  -0.1140  - 0.7562i
» E=I(3)*x0*i
E =
 1.5123e+002    -2.2792e+001i
» forcomp=[I; U1; E];
» compass(forcomp)
» gtext(' I1')
» I1m=abs(I(1))
» I1a=real(I(1))
» cosf1=cos(angle(I(1)))
```

Проходження синусоїдного сигналу з частотою 10 Гц та амплітудою 2 В через коливну ланку з ПФ 1/(a2*p^2+a1*p+1)

```
» w=10; p=i*w; a2=0.1; a1=0.2;
» WP=1/(a2*p.^2+a1*p+1); % АФХ
» A=abs(WP); FI=angle(WP); % АЧХ та ФЧХ
» X1=2; X2=X1*A %  Амплітуди вхідного та вихідного сигналів
X2 =
0.2169
```

1.7 Рядкові (текстові) вирази

```
» a ='abd'
a =
abd
```

Конкатенація рядкових масивів:

```
» st1 =['y', 't']
st1 =
yt
» st8=['7bd k';'a 77f']
st8 =
7bd k
a 77f
» st8(2,5)
ans =
f
```

Коди символів:

```
» ast=[98 'a' 'b' 97]
ast =
baba
» strcat( '78', 'hgyt')
ans =
78hgyt
» ast1=[1 2]
ast1 =
     1     2
» ast1(1)='c'
ast1 =
    99     2
>> string([192:223])
ans =
АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧ
ШЩЪЫЬЭЮЯ
>> string([224:255])
ans =
абвгдежзийклмнопрстуфхцчшщъ-
ыьэюя
```

Перетворення числа в рядок:

```
» st2= int2str(3)
st2 =
3
» t=clock;
» t=clock;z=['зараз - ',int2str(t(1)),' рік']
z =
зараз - 2002 рік
» t=clock; strcat( 'Сьогодні:',
int2str(t(3)), '.', int2str(t(2)))
ans =
Сьогодні:11.7
» st3 =[st1, st2]
st3 =
yt3
```

Пошук підрядка:

```
» findstr(z,'20')
ans =
     9
» findstr(z,'a')
ans =
     2     4
```

Зміна регістру:

```
» upper(z)
ans =
ЗАРАЗ - 2002 РІК
» lower(ans)
ans =
зараз - 2002 рік
```

Обчислення рядкових виразів:

```
» a=5; b=3; eval( 'a+b*2')
ans =
    11
» eval( 's=sin(2)')
s =
    0.9093
» t= '1+a.^2'; c=eval(t)
c =
    26
» n=5; eval([ 'St', 'ro', 'ka', '=n' ])
Stroka =
     5
```

Обчислення функції, заданої рядковим виразом:

```
» feval( 'sin', pi/2)
ans =
     1
» u='exp'; f=feval(u,1)
f =
    2.7183
```

файл extr1.m:

```
function y=extr1(x)
y=(x-2)^4+2;
```

```
» d=2; e=feval( 'extr1', d)
e =
     2
```

1.8 Вбудовані, або внутрішні (inline) функції

Inline-функція з одним параметром

```
 y=inline( '4.*x.^2+2.*x-6')
y =
     Inline function:
     y(x) = 4.*x.^2+2.*x-6
y1=y(3)
y1 =
    36


y([2 4; 6 5])
ans =
    14    66
   150   104


y(1:6)
ans =
     0    14    36    66   104   150
```

Inline-функції з кількома параметрами

```
z=inline( '2.*t+a-1', 't', 'a')
z =
     Inline function:
     z(t, a) = 2.*t+a-1
zz=z(2, 3)+7
zz =
    13
z5=inline( '(sin(x).*d-5).^2+c', 'x','d','c')
z5 =
     Inline function:
     z5(x, d, c) = (sin(x).*d-5).^2+c
d=2; c=1; z5(2, d, c)
ans =
   11.1213

g = inline( 'x^P1', 1)
g =
     Inline function:
     g(x, P1) = x^P1
g(2,3)
ans =
     8
s=inline( 'x.^a1+a2.*a3', 3)
s =
     Inline function:
     s(x, P1, P2, P3) = x.^a1+a2.*a3
s(2, 3,1, 2)
ans =
    10
```

Імена аргументів Inline-функції:

```
an=argnames(s)
an =
    'x'
    'P1'
    'P2'
    'P3'
```

1.9 Масиви записів (структури)

Створення та доповнення масиву записів:

```
» work.name='Петренко Іван'
work =
    name: 'Петренко Іван'
» work.birth=[1952 03 23]
work =
     name: 'Петренко Іван'
    birth: [1952 3 23]
» work.status='лаборант'
work =
      name: 'Петренко Іван'
     birth: [1952 3 23]
    status: 'лаборант'
» work.attest=[1985 1990 1995 2000; 4
5 4 4; 3 5 4 5; 5 5 5 4]
work =
      name: 'Петренко Іван'
     birth: [1952 3 23]
    status: 'лаборант'
    attest: [4x4 double]
» work(2).name='Дроздова Ірина'
work =
1x2 struct array with fields:
    name
    birth
    status
    attest
```

Виклик запису:

```
» work(1).birth
ans =
      1952          3          23
» work(1).attest(2,3)
ans =
    4
```

Зміна запису:

```
» work(1).name='Петренко Семен'
work =
1x2 struct array with fields:
   name
   birth
   status
   attest
```

Знищення поля:

```
» work(1).oklad=175.5
work =
1x2 struct array with fields:
   name
   birth
   status
   attest
   oklad
» work=rmfield(work,'oklad')
work =
1x2 struct array with fields:
   name
   birth
   status
   attest
```

Знищення запису:

```
» work(2)=[ ]
work =
     name: 'Петренко Семен'
    birth: [1952 3 23]
   status: 'лаборант'
   attest: [4x4 double]
```

2 ГРАФІКА MATLAB

2.1 Двовимірна графіка

Найпростіший спосіб побудови графіка функції, заданої як рядок або у вигляді символьного виразу:

```
ezplot( 'sin(x).^2')
ezplot tan(sin(x))
ezplot( 'sin(x).^2+2', [-1  4] )
```

Графік у декартовій системі координат:

```
x=linspace(0, 2*pi, 30); y=sin(x); plot(x,y)
```

або plot(x, 1+cos(x).^2)

або графік комплексного аргумента:

```
x=- pi: pi/50: pi; a=cos(x)+i*5*sin(x); plot(a)
```

Графік функції, що задана у рядковому форматі в інтервалі [Xmin Xmax]:

```
fplot( '1+sin(x)', [0  4*pi])
fplot( 'extr1', [-2  8])    -  функція задана М-файлом extr1.m
```

Графік inline-функції

```
x=1:10; z=inline( '1+x'); z =inline('1+x.^2'); plot(x, z(x))
```

Графіки у напівлогарифмічному та логарифмічному масштабі

а) масштаб осі X лінійний:

```
y3=exp(x); semilogy(x, y3)
```

б) масштаб осі Y лінійний:

```
y4=log(x); semilogx(x, y4)
```

в) масштаб обох осей логарифмічний:

```
u=logspace(-1,3); loglog(u, exp(u)./u)
```

Стовпчикові діаграми

а) другий стовпчик масиву як функція першого:

z=[1 2; 7 5; 3 2; 4 5; 5 10]; bar(z(:, 1), z(:, 2))

barh(z(:, 1), z(:, 2)) - горизонтальна гістограма

б) обидва стовпчики як функція номеру:

bar(z) або

bar(z, 'stacked')

в) тривимірна:

x=1: 5; y =sqrt(x).*5; z3=exp(x)./15; bar3(x, [z3' y']) або

bar3(x, [z3' y'], 'grouped') або

bar3(x, [z3' y'], 'stacked') або

Графік дискретних відліків (гратчаста функція)

stem(z(:,1),z(:,2))

Графіки радіус-векторів:

z1=3+5*i; compass(z1)

z2=[3+5*i -2+6*i 4-2*i]; compass(z2)

Графік руху точки на площині:

x=linspace(0, 2*pi, 500); y=sin(x); comet(x,y)

Графік у полярних координатах:

fi=0: pi/20: 4*pi; R=3.*fi;

polar(fi, R)

Кругові діаграми:

pie(z(:, 2))

pie(z(:, 2), [0 1 0 0 1]) - зсув двох секторів

pie3(z(:, 2))

Способи суміщення кількох графіків у спільному графічному вікні

а) три графіки у спільному вікні:

y1=cos(x); y2=sin(x).^2; plot(x, y, x, y1, x, y2)

б) Графік з двома осями ординат:

x1=0:20; x2=-10:10; y2=2.*x2; plotyy(x1, y1, x2, y2)

в) накладення нового графіка на вже побудований:

hold on - вмикання режиму накладення наступного графіка на попередній (осі попереднього графіка зберігаються);

hold off - вимикання цього режиму;

hold - перемикання режиму накладення (при неодноразовому повторенні команди)

Приклад 1:

plot(x, y); hold on; plot(x, y1)

hold off; plot(x, y2)

Приклад 2 (графіки різного типу в спільній системі координат):

z=[1 2; 7 5; 3 2; 4 5; 5 10]; bar(z(:, 1), z(:, 2)); hold on

x=1:.1:7; plot(x, exp(x)./20, 'r')

г) підвікна графічного вікна:

subplot(2,1,1); bar(z); subplot(2,1,2); plot(x, y, x, y1, x, y2)
x=0:.01:1; for k=1:4, subplot(2,2,k), plot(x,sin(pi*k*x)), end

Оформлення поточного графіка

а) автоматичне чергування кольорів графіка: yellow, magenta, cyan, red, green, blue, white

x=linspace(0,1,20); k=.1:.1:.8; y=k'*x; plot(x,y)

б) задавання кольору та типу лінії та точки:

plot(x, y,'--YS', x, y1, '-OC', x, y2,':') або w1=':XR'; plot(x, y, w1);

рядкові константи:

Колір лінії		Тип лінії		Тип маркера	
Y	Жовтий	-	суцільна	.	точка
M	Фіолетовий	:	пунктир	O	коло
C	Блакитний	-.	штрих-пунктир	X	хрест
R	Червоний	--	штрихова	+	плюс
G	Зелений			*	зірка
B	Синій			S	квадрат
W	Білий			D	ромб
K	Чорний			V	трикутник
				^	трикутник
				<	трикутник
				>	трикутник
				P	п’ятикутник
				H	шестикутник

в) сітка:

grid on - додавання сітки до поточного графіка

grid of - знищення сітки поточного графіка

grid - послідовне виконання команди додає та знищує сітку

г) назва графіка:

title('назва')

д) заголовки осей:

xlabel('назва')

ylabel('назва')

е) пояснювальний текст у точці з координатами Xi=2, Yi=1:

text(2,1,'текст')

є) пояснювальний текст, який надалі позиціонується за допомогою миші:

gtext('текст')

ж) креслення лінії за допомогою миші

gline(n) - у вікні за номером n; gline - у поточному вікні

з) легенда:

legend('функція y', 'функція y1', 'функція y2')

і) осі

встановлення діапазонів координат [Xmin Xmax Ymin Ymax]:

axis([-1 8 -2 2]) або e=[0 6 -1 1]; axis(e)

повернення до масштабу осей за замовчуванням*:*

axis auto

знищення позначень та маркерів осей

axis off

повернення раніше введених позначень та маркерів осей

axis on

к) можливість зміни масштабу поточного графіка за допомогою миші:

zoom - вздовж обох осей

zoom xon - вздовж осі X

zoom yon - вздовж осі Y

Зняття координат точок поточного двовимірного графіка за допомогою миші

ginput; d=ginput - будь-яка кількість точок (Enter по закінченні)

[X,Y]= ginput - окремі масиви для абсцис та ординат

d=ginput(n) - тільки *n* точок

Створення нового графічного вікна

а) створення нового вікна (старі вікна залишаються, а наступні графіки будуються в новому):

figure

б) створення нового вікна з номером i (якщо вікно з таким номером уже існує, воно стає поточним)

figure(i)

Збереження інформації з графічного вікна

а) збереження числових масивів:

меню графічного вікна File\Save

тип файла - Mat.-files (*.mat)

б) збереження числових масивів та програми побудови графіка:

меню графічного вікна File\Save

тип файла - M.-files (*.m)

в) Збереження графіка як растрового графічного файла .jpg:

[Q, map]=capture(1); (1 - номер графічного вікна)

imwrite(Q, map,'gr1. jpg');

г) Збереження поточного графіка як растрового графічного файла .bmp:

print -dbitmap gr2.bmp

д) Копіювання поточного графіка до буфера як метафайла:

меню графічного вікна Edite\Copy Figure (опцію меню MATLAB File\Preferences\CopyingOptions\ClipboardFormat\WindowsMetafile увімкнено) або

print -dmeta

Після вставлення з буферу до Word рисунок піддається редагуванню засобами Word.

е) Копіювання поточного графіка до буфера як растрового рисунка:

print -dbitmap

або

меню графічного вікна Edite\Copy Figure (опцію меню MATLAB File\Preferences\CopyingOptions\ClipboardFormat\WindowsBitmap увімкнено)

2.2 Тривимірна графіка

Тривимірна лінія:

```
t=0:pi/50:10*pi;
x=sin(t); y=cos(t);
plot3(x,y,t); grid on
```

Функція, яка створює двовимірні масиси для побудови тривимірних поверхонь

```
xx=1:5; yy=10:10:40; [XX,YY]=meshgrid(xx, yy)
XX =
     1     2     3     4     5
     1     2     3     4     5
     1     2     3     4     5
     1     2     3     4     5
YY =
    10    10    10    10    10
    20    20    20    20    20
    30    30    30    30    30
    40    40    40    40    40
```

Поверхня, задана тривимірними лініями:

```
u=-2:.1:2; v=-1:.1:1;
[X,Y]= meshgrid(u,v);
Z=exp(-X.^2-Y.^2);
plot3(X, Y, Z)
```

Поверхня у вигляді каркасу:

```
figure; mesh(X,Y,Z)
```

Те ж, с лініями рівня або у вигляді стовпців:

```
figure; meshc(X,Y, Z)
figure; meshz(X,Y, Z)
```

Поверхня з забарвленням:

```
figure; surf(X,Y, Z)
```

Те ж, з лініями рівня:

```
figure; surfc(X,Y, Z)
```

Лінії рівня тривимірної поверхні:

```
figure; contour(X,Y, Z)
```

те ж саме, з замовленням рівнів та їх цифровим позначенням:

figure; [C, H]=contour(X,Y, Z, [0.9 0.7 0.5 0.3 0.1]); clabel(C, H)

те ж саме, з вибором розташування міток рівнів за допомогою миші:

figure; C=contour(X,Y, Z, [0.9 0.7 0.5 0.3 0.1]);
clabel(C, 'manual') % Клікати лінії рівня

Лінії рівня тривимірної поверхні з забарвленням проміжків:

figure; c= contourf (X,Y, Z, [0.9 0.7 0.5 0.3 0.1]); clabel(c, 'manual')

Поверхня, задана полігонами:

figure; fill3(X,Y, Z, X)

або:

waterfall(X,Y, Z)

Властивості поточного тривимірного графіка

Характер забарвлення поверхні:

shading interp
shading faceted
shading flat

Палітри:

colormap(hsv) або замість hsv: hot, gray, bone, copper, pink, white, flag, lines, colorcube, jet, prism, cool, autumn, spring, winter, summer
colormap('default') - палітра за замовчуванням

Ракурс:

view(az, el) az - азімут (за замовчуванням -37,5°); el - кут підвищення (за замовчуванням 30°). Наприклад:
view(-15, 20)

Виведення шкали кольорів:

colorbar

Режим інтерактивного повороту поточного графіка за допомогою миші:

rotate3d on,
rotate3d off,
rotate3d

2.3 Дескриптивна графіка

Об'єкт figure (графічне вікно):

hf=figure(45) % hf - дескриптор графічного вікна №45
set(hf, 'Name', 'MyFigure', 'Color', [0.4 0.5 0.6])
set(hf, 'NumberTitle', 'off', 'MenuBar', 'none')
set(hf, 'position', [30 50   450   420])

Зміна кольору заданих вікон

figure(3); figure(6); whitebg([3 6], [.9 .85 .87])

Об'єкт axes (осі):

ha=axes('XLim', [1 6], 'YLim', [2 10], 'Pos', [.1 .1 .85 .85]);
set(ha, 'box', 'on', 'Color', [.5 .6 .7]);
set(ha, 'FontSize', [20], 'FontName', 'TimesNewRoman', 'FontWeight', 'bold', 'FontAngle', 'Ital');

grid
set(ha, 'GridLineStyle', '--', 'LineWidth', 3)
set(ha, 'XColor', [1 0 0], 'XTick', [1: 0.5: 6]);
set(ha, 'XAxisLocation', 'top', 'TickLength', [0.02 0.0])

Об'єкт line (лінія):

hl1=line([2 4], [4 8], 'Color', [0.8 0.5 .1], 'LineWidth', 4);
hl2=line([3 5], [8 2], 'Color', 'r');
set(hl2, 'Ydata', [7 4])
hl3=line([3 5], [4 2], 'Color', 'g', 'LineWidth', 3);
set(hl3, 'Marker', 'o','MarkerFaceColor', 'y', 'MarkerEdgeColor', 'r')
hl4=line([0.5 2], [2 5], 'Color', [0.5 1 0.7],'LineStyle',':');
set(hl4, 'Clipping', 'off')
hl5=line(3, 3, 'Color', [0.5 1 0.7], 'Marker', ' + ', 'MarkerSize', 20);
set(hl5, 'Color', 'r', 'LineWidth', 5)

Об'єкт patch (лоскут):

hp1=patch([3.5 4.5 4.5 5], [8 10 8 6], 'y');
set(hp1, 'EdgeColor',' m', 'LineWidth',6)
hp2=patch([5 4 5.5], [8 6 4], [.8 .9 .8], 'Marker', 's')

Об'єкт text (текст):

ht=text(1.6, 9, 'Exercize\newlineNumber1');
set(ht,'Editing','on'); % Відредагувати текст та клацнуть двічі за межами текстового поля
set(ht, 'HorizontalAlignment', 'right')
set(ht,'String','New text')
set(ht, 'Rotation',10)

Операції з дескрипторами графічних об'єктів:

Виклик можливих властивостей поточного об'єкту ({за умовчанням})

set(hf)

Виклик установлених властивостей поточного об'єкту

get(hf)

Виклик одного із установлених властивостей поточного об'єкту

get(hf, 'Color')

Виклик установлених властивостей об'єктів (лінія, лоскут, текст) створюваного графіка

hpl=plot(...)

Знищення графічних об'єктів

delete([hp1 hp2]); delete(hp2)

Пошук об'єкту із заданими властивостями

hh=findobj('Color', [0.8 0.5 .1])
hh =
  12.0015
hl1 =
  12.0015

Вилучення числового матеріалу з побудованого графика за відомим параметром (колір за умовчанням першого графика):

```
figure; plot(0: .01: pi, sin(0: .01: pi), .1, .5, 'or');
fh=findobj('Color', [0 0 1]) % Пошук об’єкта синього кольору
set(fh, 'Color', [0 1 0]); % Перефарбування знайденого в зелений
data=[(get(fh, 'XData'))' (get(fh, 'YData'))']; % Вилучення числового
```

Ідентифікація секторів колової діаграми (на прикладі першого сектора)

```
x= [3; 7;1; 2];
h= pie(x);
ht=findobj(h, 'Type', 'text'); % Пошук об’єктів типу'text' у графічному вікні з колової діаграмою
Pos= get(ht(1), 'Position'); str= get(ht(1), 'String'); % Вилучення текста та його положення
set(ht(1), 'Position', Pos+[-.1  0  0], 'String', [ '1 - '  '\newline'  str])
% Коригування текста та його положення
```

3 ОСНОВИ ПРОГРАМУВАННЯ У MATLAB

3.1 М-файли

Сценарій qu1.m:

```
%Розрахунок і побудова параболи
%при зміні x від xmin до xmax
%Змінні xmin та xmax створити у робочій області заздалегідь
x=xmin:xmax;
y=x.^2;
plot(x,y,'or')%Побудова графіка
grid on
```

Виклик: qu1

Статус змінних сценарію:

```
» xmin=-5;xmax=5; qu1
» y
y =
   25    16     9     4     1     0     1     4     9    16    25
```

Функція qu2.m:

```
function y=qu2(xmin,xmax)
%Розрахунок і побудова параболи
%при зміні x від xmin до xmax
x=xmin:xmax;
y=x.^2;
plot(x,y,'or')%Побудова графіка
grid on
```

Виклик: qu2(-5,5) або

y1=qu2(-2,12)+4 або

x1=-3;x2=4;z=qu2(a,b)

Статус змінних функції:

```
» clear y
» z=qu2(0,3)
z =
    0     1     4     9
» y
??? Undefined function or variable 'y'.
```

Функція з двома вихідними параметрами qu3.m:

```
function [y1,y2]=qu3(xmin,xmax)
%Розрахунок і побудова параболи та екпоненти
%при зміні x від xmin до xmax
x=xmin:xmax;
y1=x.^2;
y2=exp(x);
plot(x,y1,'or',x,y2,'-g')
grid on
```

Виклик:

```
» [q1, q2]=qu3(-3,3)
q1 =
    9     4     1     0     1     4     9
q2 =
   0.0498    0.1353    0.3679    1.0000    2.7183    7.3891   20.0855
```

Функція без вхідних параметрів:

```
function [x,y]=randv2
%ФормуванняЯ випадкового вектора одиничної довжини
z=rand(1,2)-0.5;
x=z(1);y=z(2);
L=sqrt(x.^2+y.^2);
x=x/L;y=y/L;
compass(x+i*y)
```

Передача даних з М-функції до інших М-функцій або до базової робочої області:

М-функція, у якій за допомогою функції assignin створюється змінна as=78:

```
function aseval
assignin('caller','as',78);
```

М-функція, у якій за допомогою функції `evalin` викликається змінна `as`:

```
function y=evalas
y=evalin('caller','as');
>> aseval
>> evalas
ans =
```

```
   78
```
Виклик змінної `as` із командного рядка або сценарію:
```
>> as
as =
   78
```
Передача даних тільки до базової робочої області
```
function aseval2
assignin('base','as2',34);
>> as2
as2 =
   34
```
Локальна змінна, значення якої зберігається до наступного виклику функції
```
function y=ff(j)
persistent k;
i=0;
k=k
while i<j
   i=i+1;
   k=i;
end
k=k
y=j;
```
Повторити виклик функції ff.m кілька разів із різними вхідними параметрами. Переконатися, що кожного разу перше значення k поточного сеансу дорівнює другому значенню k попереднього сеансу .

Передача інформації поміж функціями за допомогою глобальних змінних

Створити дві m-функції:

y3.m:
```
function y=y3(x)
global z2
z2=3;
y=x^2+z2;
```
y2.m:
```
function y=y3(x)
global z2
y=x+z2;
```
Використавши спочатку лише функцію y2, переконатися, що результатом є порожній масив (змінну z2 не визначено). Якщо ж спочатку викликати функцію y3, а потім y2, результатом буде число.

3.2 Оператори відношення

Порівняння матриці з вектором:
```
» M=[-4  0; 2  6]
```
```
M =
    -4     0
     2     6
```
```
» M==0
ans =
     0     1
```

```
    0     0

» M>0
ans =
    0     0
    1     1

» M<0
ans =
    1     0
    0     0

» M<=0
ans =
    1     1
    0     0
» M>=0
ans =
    0     1
    1     1

» M~=0
ans =
    1     0
    1     1

Порівняння матриці з матрицею:
» N=[-6 0; 2  8]
N =
   -6     0
    2     8

» N= =M
ans =
    0     1
    1     0

» N<M
ans =
    1     0
    0     0

» N>M
ans =
    0     0
    0     1
» N>=M
ans =
    0     1
    1     1
» N<=M
ans =
    1     1
    1     0
» N~=M
ans =
    1     0
    0     1
Порівняння рядків:
» 'v'= ='v'
ans =
    1

» 'v'= ='t'
ans =
    0
» 'vest'= ='vist'
ans =
    1     0     1     1
```

3.3 Інтерактивне введення/виведення даних

Діалогове введення числа:

```
» r=input('Введи радіус r = ')
Введи радіус r = 7.7
r =
   7.7000
```

Діалогове введення тексту:

```
» s=input('Введи текст ')
Введи текст 'Це текст'
s =
Це текст
```

або:

```
» name=input( 'Введи прізвище: ', 's')
Введи прізвище: Шевчук
name =
Шевчук
```

Введення за допомогою меню

```
» n=menu( 'Куди підемо далі?', 'Прямо', 'Ліворуч', 'Праворуч', 'Назад')
n =
```

4

Виведення проміжних результатів:

```
» disp( 'Довжина окружності l='); disp(2*pi*r)
Довжина окружності l=
  48.3805
```

Вибір мишею координат точок у поточному графічному вікні:

```
» figure; axis( [-1 1 -1 1] )
» a= ginput
a =
  -0.3318   -0.4211
  -0.0415    0.3918
...
  -0.1014    0.3626
  -0.0876   -0.1170
```

(Після вибору останньої точки натиснути Enter)

```
» plot(a(:,1),a(:,2),'o')
```

або:

```
» [X,Y]= ginput
```

Виведення повідомлення про помилку:

```
» error('Вхідний параметр не може бути скалЯром')
```

3.4 Керуючі структури

if

```
%Одноразовий розрахунок довжини окружності за заданим радиусом
r=input('Введи радіус r=');
if r>=0
     disp('Довжина окружності l=');disp(2*pi*r);
else
   disp('r<0!')
end
```

while

Файл dial2pir.m

```
%Багаторазовий розрахунок довжини окружності за заданим %радиусом
r=0;
while r>=0
   r=input('Введи радіус r=');
   if r>=0
            disp('Довжина окружності l=');disp(2*pi*r);
   end
end
```

while

```
function [x,y]=d2mass(x,y);
%Формування двовимірного масив размірністю [2,N]
%Відображення його в координатах XY
```

```
global x y;
N=input('Розмірність масиву N=');
k=0;
x=zeros(1,N);
y=zeros(1,N);
while k<N
   k=k+1;
   x(k)=input('Введи X=');
   y(k)=input('Введи Y=');
 end
plot(x,y,'rs')
```

for

```
%Оператор for
for k=1:2:13
   k.^2
end

%Оператор for
for i=1:3
   for j=1:4
      A(i,j)=i+j
   end
end
```

for

```
%Оператор for
%Формування чотирьох змінних Pi=i.^2
for i=1:4
   eval([ 'P', int2str(i), '=i.^2'])
end
```

switch

```
%Введення двійкових чисел
a=input('введи двійкове число=');
switch a
case 0
   disp('a=0')
case 1
   disp('a=1')
otherwise
   disp(a);disp(' - не двійкове число')
end
%Розрахунок та побудова функцій за вибором
a=input('введи ім"я функції:','s');
y=zeros(1,10);
```

```
x=1:10;
switch a
case 'sin'
   y=sin(x);
   plot(x,y)
case 'log'
   y=log(x);
   plot(x,y)
case 'exp'
   y=exp(x);
   plot(x,y)
otherwise
   disp(strcat(a,' - неіснуюча функціЯ!'))
end
%Вибір функцій
fun=['sin';'exp';'log'];
disp('1) sin')
disp('2) exp')
disp('3) log')
k=input('Вибери номер функції=');
x=input('Введи значення x=');
feval(fun(k,:),x)
```

3.5 Перевірка типу даних

» isnumeric(a) - істина, якщо a - числовий масив
» isreal(a) - істина, якщо a - масив дійсних чисел
» isempty(a) - істина, якщо a - порожній масив
» isequal(F,Y) - істина, якщо масиви F та Y s ідентичні
» islogical(d) - істина, якщо d - логічний масив
» isstruct(d) - істина, якщо d - структура
» isinf([pi NaN Inf -Inf]) % - істина, якщо параметр =±Inf
ans =
    0   0   1   1
» isnan([pi NaN Inf -Inf]) % - істина, якщо параметр =NaN
ans =
    0   1   0   0
» ischar(z) - істина, якщо z - масив символів

Перевірка типу символів

» isletter('56r iuy')
ans =
    0   0   1   0   1   1   1
» isspace('56r iuy')
ans =
    0   0   0   1   0   0   0

3.6 Розв’язання систем диференціальних рівнянь

Файл опису системи диференціальних рівнянь двигуна постійного струму
Файл dptod.m (скалярна форма):

```
function dy=dptod(t,y)
%Дифференциальные уравнения двигателя постоянного тока
%2ПБВ112S с постоянным потоком
dy=zeros(2,1);
Mc=15;
r=0.115;L=7.32e-4;J=0.034;
k=15/28;u=33;
dy(1)=-r/L*y(1)-k/L*y(2)+u/L;
dy(2)=k/J*y(1)-Mc/J;
```

Файл dptodm.m (матрична форма):

```
function dy=dptodm(t,y)
%Диференціальні рівняннЯ двигуна постійного струму 2ПБВ112S
%з поствйним потоком
dy=zeros(2,1);
Mc=15;
r=0.115;L=7.32e-4;J=0.034;
k=15/28;u=33;
A=[-r./L -k./L; k./J 0];
B=[1/L 0;0 -1/J];
U=[u;Mc];
dy=A*y+B*U;
```

Сценарій розв’язання СДР для режиму прямого пуску двигуна
Файл pusk.m:

```
%Сценарій прямого пуску двигуна постійного струму
%(модель dptod.m)
[t,y]=ode45('dptod',[0 0.2],[0 0]);
plot(t,y);
title('Прямий пуск двигуна постійного струму під навантаженняМ');
grid on;
xlabel('t, c');
gtext('i');gtext('w');
zoom
figure;
plot(y(:,1),y(:,2));
xlabel('i');
ylabel('w');
title('Динамична механічна характеристика');
```

```
grid on;
zoom
```

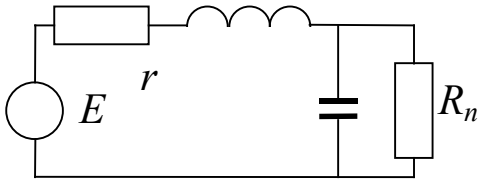
Сценарій розрахунку процесу вмикання RLC-кола до джерела синусоїдної напруги

Файл rlcsin.m:

```
%Сценарій розрахунку перехідних процесів в RLC-колі після вмиканнЯ до
%джерела синусоїдної напруги (рівняння у файлі rlcod.m)
[t,i]=ode23s('rlcod',[0 0.08],[0 0]);
E=220*sqrt(2)*sin(100*pi*t);
plot(t,i(:,1),'-r',t,i(:,2),'-g',t,E/300,'--b');
zoom;
figure;
comet(i(:,1),E)
```

Файл опису системи диференціальних рівнянь RLC-кола

Файл rlcod.m:




```
function di=rlcod(t,i)
%Диференціальні рівняння RLC кола
r=1;L=0.01;C=1e-2;Rn=3;
E=220*sqrt(2)*sin(100*pi*t);
di=zeros(2,1);
di(1)=E-i(1)*r/L-i(2)*Rn/L;
di(2)=(i(1)-i(2))/C/Rn;
```

Опції розв'язувача ДР

Створення структури опцій:

```
» ops=odeset( 'RelTol', 1e-8, 'OutputFcn', 'odephas2', 'Stats', 'on', 'Refine', 4)
```

Редагування опцій після створення:

```
» ops.Refine=2
```

Додавання нової опції:

```
» ops.Vectorized='on'
```

Виклик за замовчуванням функції odeplot:

```
» ode45( 'dptod', [0  0.2], [0  0])
```

Виклик функції odephas2:

```
» ops=odeset( 'OutputFcn', 'odephas2')
» ode45( 'dptod', [0  0.2], [0  0], ops)
```

4 SIMULINK ТА РОБОЧА ОБЛАСТЬ

4.1 Задавання параметрів блоків SIMULINK як змінних з робочої області

1. Функціональний перетворювач (блок Look-Up Table):

Параметри блока:

Vector of input values: [0; t1; t2]

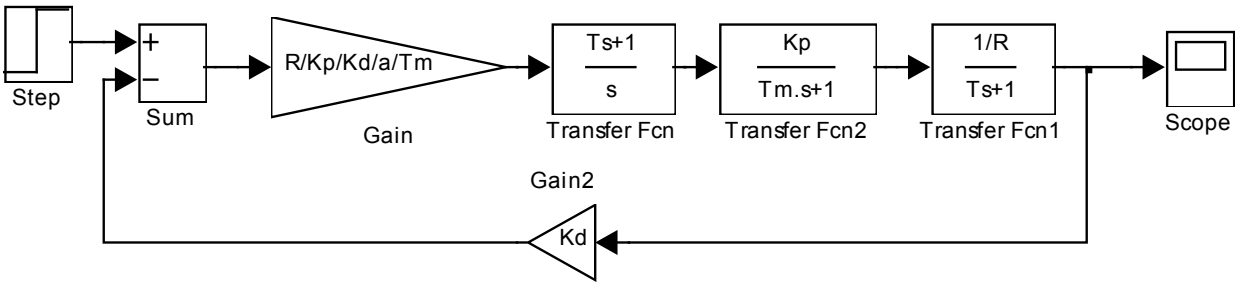
Vector of output values: [u1; u2; u3] або [u1; u2; u2-2]

Їх значення в робочій області:

» t1=0.5; t2=1; u1=1; u2=3; u3=2;

2. Генератор синусоїди (блок Sine wave):

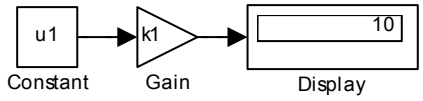
Параметри блока:
Amplitude: A
Frequency: w
Phase: fi*2
Їх значення в робочій області:
» A=2; w=1; fi=0.1;



Якірне коло: Numerator: [1/R] Denominator: [T 1]
Перетворювач енергії: Numerator: [Kp] Denominator: [Tm 1]
Датчик струму: Kd
Регулятор струму (сталі часу): Numerator: [T 1] Denominator: [1 0]
Регулятор струму (коефіцієнт підсилення): R/Kp/Kd/a/Tm
Значення параметрів у робочій області:
» Kd=4; R=0.1; Kp=20; a=2; T=0.2; Tm=0.01;
Запустити модель з різними параметрами об'єкту керування

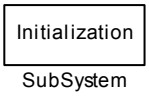
4. Задавання параметрів моделі без прямого звертання до робочої області

Створити модель та задати в ній параметри як змінні (k1 та u1). Створити сценарій, у якому цим змінним надаються значення, та зберегти його. Наприклад:



k1=2;
u1=5;

Створити підсистему (блок Subsystem) та маскувати її (Edit\MaskSubsystem…). У вікні "Mask Editor…" (вкладка Icon) у полі "Drawing commands" записати команду, яка виводить всередині підсистеми якусь підказку: disp('Initialization'). Визвати для блоку Subsystem команду "Edit\Block Properties…" та у полі "Open function:" увести ім'я створеного сценарію.



Для присвоєння параметрам k1 та u1 їхніх значень клікнути двічі блок Subsystem. Після цього модель можна запускати.

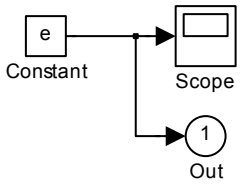
4.2 Передача даних із SIMULINK до робочої області

5. Передати до РО синусоїду з завдання 2 за допомогою блоку ***To WorkSpace*** з ім'ям ***sin2***. Параметр Decimation=2. Переглянути отриманий масив у робочій області.
6. За допомогою блоку ***Out*** передати вектор струму з завдання 3 з ім'ям ***circuit*** та вектор часу з ім'ям ***t3***. Переглянути отримані масиви у робочій області.
7. За допомогою блоку Scope передати матрицю результатів моделювання з ім'ям ***res***. Параметр Decimation=2. Переглянути отриманий масив у робочій області.
8. Запустити з командного рядка за допомогою функції ***sim*** модель із завдання 3. У переліку змінних, що передаються до РО, залишити час та струм. Переглянути отриманий масив у робочій області.

4.3 Передача до моделі SIMULINK значення змінної (скаляра) з М-функції, що її викликає, та повернення з SIMULINK до М-функції векторів часу та вихідної змінної

Передати з командного рядка через М-функцію zq.m до моделі zq1.mdl значення змінної e=12. Після виклику функції перевірити факт отримання інформації моделлю Simulink (за допомогою осцилографа) та її повернення з моделі Simulink через М-функцію до робочої області MATLAB.

Модель zq1.mdl:



М-функція zq.m:

```
function [T,X,Y]=zq(e)
op1=simget('zq1');
op1.SrcWorkspace='current';% Workspace of funciton zq.m
op1.OutputVariables='ty';
[T,X,Y]=sim('zq1',[],op1);
```

Виклик:

```
>> [T,X,Y]=zq(12);
>> Y(1)
ans =
   12
>> T(1:10)
ans =
         0
    0.2000
    0.4000
    0.6000
    0.8000
    1.0000
```

```
    1.2000
    1.4000
    1.6000
    1.8000
```

5 ПРИКЛАДИ РОЗВ'ЯЗАННЯ ОПТИМІЗАЦІЙНИХ ЗАДАЧ

Пошук мінімального значення вектора

```
q=[1, 4, 8, 5, 0, 3];
s=min(q)
s =
          0
```

Пошук мінімального значення масиву та його номера

```
       а)
z=[1 5 4; 6 2 3; 5 9 0]
z =
           1     5     4
           6     2     3
           5     9     0
         б)
[q, i]=min(z(:, 2))
q =
            2
i =
           2
```

Пошук мінімальних значень стовпців матриці

```
z=[5 1 4; 6 2 3; 4 9 0]
z =
    5     1     4
    6     2     3
    4     9     0
s=min(z)
s =
    4     1     0
```

Теж саме, із зазначенням номера мінімального елемента

```
[s, i]=min(z)
s =
    4     1     0
i =
    3     1     3
```

```
y=[0 1 2 3 4 5 4 2 1 0]; x=[2 3 4 5 6 7 8 9 10 11]; ind=find(y==max(y))
```

```
ind =
6
xm=x(ind)
xm =
     7
```

Пошук x, що відповідає мінімуму функції cos(x) у діапазоні x=[3, 5].

```
fminbnd( 'cos',  3,  5)
ans =
   3.1416
x=fminbnd( '2-cos(x)', -2, 3)
x =
 -3.7792e-006
```

Пошук x, що відповідає мінімуму функції $y = 5 + x^2 - 2x$ ***у діапазоні*** $x = [-3,2]$.

```
fminbnd( '5+x^2-2*x', -3, 2)
ans =
         1.0000
```

Те ж саме, із розрахунком значення функції в оптимальній точці

```
fminbnd( '5+x^2-2*x', -3, 2);
x=ans;
y=5+x^2-2*x
y =
         4.0000
```

Задавання опцій оптимізації (виведення звіту про перебіг пошуку)

```
options=optimset('Display','iter');
x=fminbnd( '5+x^2-2*x',-3, 2, options)

Func-count    x          f(x)         Procedure
    1      -1.09017      8.36881       initial
    2      0.0901699     4.82779       golden
    3       0.81966      4.03252       golden
    4           1              4        parabolic
    5       1.00003            4       parabolic
    6      0.999967            4       parabolic

x =
   1.0000
```

Мінімізація М-функції

Попередньо створити М-файл с ім’ям ff5.m:

```
function y=ff5(x)
%ff5 Parabole
```

```
    y=5+x^2-2*x;
```
Потім виконати команду:
```
fminbnd( 'ff5', -3, 2)
ans =
   1.0000
```

Безумовна мінімізація функції кількох змінних симплексним методом із виведенням звіту про перебіг пошуку

```
» x0=[-2, 2];
» options=optimset('Display','iter');
» x=fminsearch( '100*(x(2)-x(1)^2)^2+(1-x(1))^2', x0, options)
        Iteration   Func-count     min f(x)         Procedure
            1           3              370           initial
            2           5            126.65          expand
            3           7           33.8125          expand
            4           9            9.3125          reflect
x =
   1.0000    1.0000
```

Мінімізація Inline-функції

```
zz=inline( 'x(1).^4+x(2).^4+2.*x(1).^2.*x(2).^2-4.*x(1)+3')
zz =
     Inline function:
     zz(x) = x(1).^4+x(2).^4+2.*x(1).^2.*x(2).^2-4.*x(1)+3
xi=[1 1]; x0=fmins(zz, xi)
x0 =
   1.0000    0.0000
zz(x0)
ans =
  1.2784e-009
```

Безумовна мінімізація функції кількох змінних методом Гилла-Мюррея зі звітом про хід пошуку

```
>> x0=[-2, 2];
>> options=optimset('Display','iter','LargeScale','off','HessUpdate','gillmurray');
>> x=fminunc( '100*(x(2)-x(1)^2)^2+(1-x(1))^2', x0, options)

                                                    Directional
        Iteration  Func-count     f(x)        Step-size       derivative
            1          2            409          0.001          -2.74e+006
            2          8          6.07598     0.000334035     -1.64e+004
            3         15           6.0513      0.0802738       -9.32e-005
 ...
           25        155       0.000629285      8.37102        -4.41e-006
```

```
    26       161       4.79039e-005      1.10899       -5.56e-007
    27       167       1.07615e-006      1.06022        5.09e-007
Optimization terminated successfully:
 Current search direction is a descent direction, and magnitude of
 directional derivative in search direction less than 2*options.TolFun
x =
   0.9999    0.9997
```

Теж саме з аналітично заданим градієнтом

Створити М-функцію
rosengrad.m (функція, що мінімізується):

```
function [r,gr]=rosengrad(x)
%function of Rosenbrock with gradient
%r=100*(x(2)-x(1)^2)^2+(1-x(1))^2
r=100*(x(2)-x(1)^2)^2+(1-x(1))^2;
gr=[-400*(x(2)-x(1)^2)*x(1)-2*(1-x(1)); 200*(x(2)-x(1)^2)];
```

Основна програма:

```
>> x0=[-2, 2];
>> options=optimset('fminunc');
>> options.GradObj = 'on'; options.Display='iter'; options.HessUpdate='gillmurray';
>> options.LargeScale='off';
>> x=fminunc('rosengrad',x0,options)
                                                  Directional
 Iteration  Func-count     f(x)        Step-size      derivative
     1          0             409          0.001       -2.74e+006
     2          4           6.07598    0.000334035     -1.64e+004
     3          9           6.0513       0.080274       -9.18e-005
...
    26        109       2.01136e-005     1.50506       -5.7e-007
    27        113       5.6764e-007      1.16134       -7.79e-011
 Optimization terminated successfully:
 Current search direction is a descent direction, and magnitude of
 directional derivative in search direction less than 2*options.TolFun
x =
   1.0002    1.0003
```

Умовна мінімізація функції Розенброка

а) обмеження:

```
x(1)-x(2)<=0
```

Основна програма:

```
>> x0=[-2, 2]; A=[1 -1]; b=0;
>> x = fmincon('rosengrad',x0,A,b)
     Active Constraints:
         1
```

```
x =
    0.0102    0.0102
```

б) Теж саме в діапазоні -3<=x(1)<= 0; 1<=x(2)<= 3

Основна програма:

```
lb=[-3, 1]; ub=[0, 3]; x0=[-2, 2];
options=optimset('Display', 'iter');
x = fmincon('rosengrad',x0,A,b,[],[],lb,ub,[],options)
```

Iter	F-count	f(x)	max constraint	Step-size	Directional derivative	Procedure
1	3	409	-1	1	-3.61e+003	
2	8	229	-0.5	0.5	-353	
3	12	12.3233	0	1	-10.1	
4	16	5.03209	0	1	-4.25	
5	22	4.35763	-0.1739	0.25	-0.237	
6	26	4.1293	0	1	-0.193	Hessian modified
7	30	4.00277	0	1	-0.0263	
8	34	3.98999	0	1	-2.26e-005	
9	38	3.98997	0	1	-1.79e-009	Hessian modified

```
Optimization terminated successfully:
 Search direction less than 2*options.TolX and
  maximum constraint violation is less than options.TolCon
Active Constraints:
     2
x =
   -0.9950    1.0000
```

Оптимізація функції трьох змінних $f(x) = x_1^2 + x_2^3 + x_3^4$ ***з нелінійними обмеженнями*** $x_1 + x_2 + x_3 \le 10$; $x_3 \ge x_1 \cdot x_2 + 1{,}5$; $x_1 x_2 \ge -10$

М-файл опису функції відгуку:

```
function f=fun3(x)
f=x(1)^2+x(2)^3+x(3)^4;
```

М-файл опису нелінійних обмежень

```
function [con3,c]=con3(x)
con3(1)=x(1)+x(2)+x(3)-10;
con3(2)=x(1)*x(2)-x(3)+1.5;
con3(3)=-x(1)*x(2)-10;
c=[];
```

Основна програма:

```
>> x0=[1 1 1]; lb=[-10 -5  0]; ub=[10  5];
>> opt=optimset('fmincon'); opt.Display='iter'
>>  x = fmincon(@fun3,x0,[],[],[],[],lb,ub,@con3,opt)
```

			max		Directional	

```
 Iter  F-count      f(x)       constraint   Step-size     derivative   Procedure
    1      4          3          1.5            1           -17        dependent
    2      9         -7          3.5            1          -21.1       Hessian modified;
dependent
    3     14     -35.652            0            1          -60.5       Hessian modified
twice;   dependent
    4     19     -124.897        2.872            1         0.0784       Hessian modified;
dependent
    5     26     -124.304        2.154         0.25           4.59       dependent
…
   14     72     -124.932            0            1       -0.000767     dependent
   15     77     -124.933   6.463e-012            1        -0.00304      Hessian modified;
dependent
   16     82     -124.934   3.796e-011            1        -0.000251     dependent
   17     87     -124.934   5.422e-012            1        -5.67e-007    Hessian modified;
dependent
Optimization terminated successfully:
 Magnitude of directional derivative in search direction
  less than 2*options.TolFun and maximum constraint violation
  is less than options.TolCon
Active Constraints:
     3
     8
x =
          0.2425   -5.0000    0.2876
```

Безумовна мінімізація квадрата функції Розенброка

```
x0=[-2  2];
options=optimset('MaxFunEvals',2000,'Display','iter');
x=lsqnonlin(@rosen,x0,[],[],options)
```

Результати:

```
                                                  Directional
 Iteration  Func-count     Residual      Step-size     derivative      Lambda
     1          2           167281            1       -3.35e+005
     2          8           101.044           1          -634          100.245
     3         14           37.0117         0.745        -9.93          141.309
     4         20           36.6539         0.0204        -2.64         1269.08
…
   284       1740        3.07135e-006       14.7      -9.54e-007      0.00165256
   285       1747         3.9538e-008       1.44      -1.61e-007      0.00165257
   286       1753        1.24934e-008       0.807     -4.16e-009      0.00165258
Optimization terminated successfully:
 Gradient in the search direction less than tolFun
 Gradient less than 10*(tolFun+tolX)
```

x =
1.0105 1.0212

Найменша відстань від заданої точки до параболи

Найти чисельно найменшу відстань від точки A_0 [2; 1] до параболи $y = x^2$.

Відстань від т. A_0 з координатами [x_0; y_0] до довільної точки параболи з координатами [x; y] :

$L = \sqrt{(x - x_0)^2 + (y - y_0)^2}$. Змінна y пов'язана з x через рівняння параболи. Тому незалежною змінною є лише x.

Сценарій rasst1c.m:

```
[x,fval,exitflag,output]=fminbnd(@rasst1,0,5);
x=x
L=sqrt(fval)
iterations=output.iterations
funcCount=output.funcCount
algorithm=output.algorithm
```

Функція rasst1.m:

```
function L2=rasst1(x)
x0=2;y0=1;
L2=(x-x0)^2+(x^2-y0)^2;
```

Результат:

```
rasst1c
L2 =
   7.0171
…
L2 =
   0.8248
x =
   1.1654
L =
   0.9082
iterations =
   10
funcCount =
   10
algorithm =
golden section search, parabolic interpolation
```

Синтез оптимального фильтра

До входу фильтру третього порядку з передатною функцією $W(p) = \dfrac{1}{a_3 p^3 + a_2 p^2 + a_1 p + 1}$ подано корисний синусоїдальний сигнал частоти f_1=2 Гц з амплитудою A_1=1, а також шум з частотою f_2=50 Гц та амплітудою

A_2=0,2. За допомогою функції lsqnonlin визначити параметри фильтра (a_1, a_2, a_3), які забезпечують мінімум середнього квадрату відхилення між вихідним та вхідним корисним сигналами.
Крок часу – 1/360 періоду корисного сигналу. Початкові значення параметрів a0 =[0.03 0.001 0.0001]. Параметри сигналів передавати до цільової функції з сценарію через функцію оптимізації. Побудувати криві корисного, сумарного вхідного сигналів та похибки до та після оптімиізації.
М-функція розрахунку похибки err.m (виходячи з параметрів вхідного корисного сигнала та шуму розраховує вектор похибки вихідного сигналу по відношенню до вхідного корисного):

```
function err=err(a,A1,A2,w1,w2,n)
%Error with noise
a1=a(1);a2=a(2);a3=a(3);%Parameters of transfer function
t=linspace(0,2*pi/w1,n)';%Vector of time
W=inline('1/(a3*s^3+a2*s^2+a1*s+1)');%Transfer function
s1=i*w1;s2=i*w2;
WM1=abs(W(a1,a2,a3,s1));%Amlitude-frequency character
phase1=angle(W(a1,a2,a3,s1));%phase-frequency character
WM2=abs(W(a1,a2,a3,s2));
phase2=angle(W(a1,a2,a3,s2));
Y=A1*WM1*sin(w1*t+phase1)+A2*WM2*sin(w2*t+phase2);%Output signal
err=Y-A1*sin(w1*t);%Vector of error
```

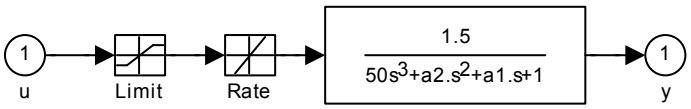
Сценарій оптимізації err_opt.m (задає параметри корисного сигналу та шуму, початкові параметри фильтра, викликає функцію оптимізації lsqnonlin та передає через неї до функції err.m ці параметри):

```
A1=1;f1=2;A2=0.2;f2=50;%Parameters of signals
a0=[0.03 0.001 0.0001];%Initial parameters of transfer function
w1=2*pi*f1;w2=2*pi*f2;
n=360;%Quantity of steps
t=linspace(0,2*pi/w1,n)';%Vector of time
u1=A1*sin(w1*t);%Input signal
u2=A2*sin(w2*t);%Noise
err0=err(a0,A1,A2,w1,w2,n);%Initial error
h11=figure(11);plot(t,u1,t,u1+u2,t,err0);
set(h11,'Name','For initial parameters');
legend('Input signal','Input signal+Noise','error Initial')
%---------------------------------Optimization--------
opterr=optimset('TolFun',1e-6);
[a,resnorm,residual,ef,output]=...
    lsqnonlin(@err,a0,[0 0 0],[],opterr,A1,A2,w1,w2,n);
disp('Optimal parameters of transfer function [a1 a2 a3]:');disp(a);
disp('Average square of error:');disp(resnorm/360);
disp('Iterations:');disp(output.iterations);
disp('FuncCount:');disp(output.funcCount);
%-----------------------------------------------------
```

```
errf=err(a,A1,A2,w1,w2,n);%Final error
h12=figure(12);plot(t,u1,t,u1+u2,t,errf);
set(h12,'Name','For final parameters');
legend('Input signal','Input signal+Noise','error Final')
>> err_opt
Optimization terminated successfully:
 Relative function value changing by less than OPTIONS.TolFun
Optimal parameters of transfer function [a1 a2 a3]:
    0.0300    0.0000    0.0002
Average square of error:
  1.3784e-009
Iterations:
    13
FuncCount:
    53
```

Синтез ПІД-регулятора для нелінійного об'єкта керування (функція fminimax)

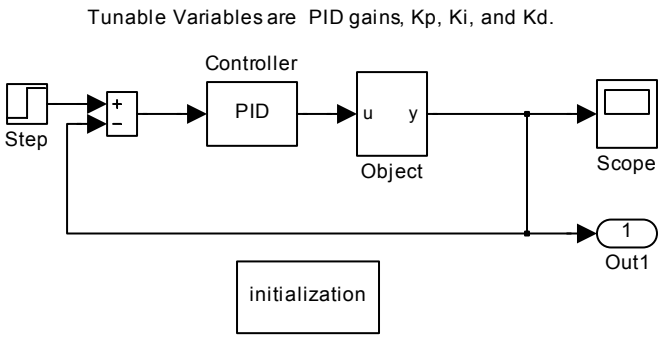
Визначити параметри ПІД-регулятора, що забезпечать під час керування об'єктом, зображеним на рисунку, перехідний процес наступної якості: досягнення рівня 0,95 не пізніше t=20 с, відсутність перерегулювання. Параметри об'єкта: a_1=3, a_2=43, обмеження рівня керування ±2, обмеження темпу зміни керування ±0,8. Початкові значення параметрів регулятора Kp = 0.63; Ki = 0.0504; Kd = 1.9688. Використати функцію fminimax.



Примітки.

1. Функції *assignin* та *evalin* призначені для передачі даних із файлу sim_minimax_obj.m до файлу sim_minimax_con.m
2. Обмеження $yout(20:100) \geq 0.95$ враховує вимоги до якості перехідного процесу.

Модель sim_minimax.mdl:



Файл цільової функції sim_minimax_obj.m:

```
function F = sim_minimax_obj(pid,a1,a2)
% Track the output of sim_minimax to a signal of 1
Kp = pid(1);
```

```
Ki = pid(2);
Kd = pid(3);
% Compute function value
opt = simset('solver','ode5','SrcWorkspace','Current');
[tout,xout,yout] = sim('sim_minimax',[0 100],opt);
F = yout;
assignin('base','F_sim_minimax_OBJ',F);
```

Файл Обмежень sim_minimax_con.m:

```
function [c,ceq] = sim_minimax_con(pid,a1,a2)
f = evalin('base','F_sim_minimax_OBJ');
% Compute constraints
c = -f(20:100)+.95;
ceq=[];
```

Файл сценарію оптимізації sim_minimax_m.m:

```
 %PID-controller
%Minimization of maximum value of signal
pid0 = [0.63 0.0504 1.9688]
a1 = 3; a2 = 43;
options = optimset('Display','iter','TolX',0.001,'TolFun',0.001);
pid = fminimax(@sim_minimax_obj,pid0,[],[],[],[],[],[],...
            'sim_minimax_con',options,a1,a2)
Kp = pid(1); Ki = pid(2); Kd = pid(3);
```

Файл ініціалізації моделі sim_minimax_init.m:

```
Kp = 0.63;
Ki = 0.0504;
Kd = 1.9688;
a2 = 43;
a1 = 3;
disp('Done initializing sim_minimax.')
```

Результат:

```
>> sim_minimax_m
pid0 =
0.6300    0.0504    1.9688
pid =
   0.5894    0.0605    5.5294
```

Пошук нуля функції (визначення ковзання АД при заданому моменті)

Файл mad.m:

```
function DM=mad(s,sk,Mk,Ms)
%Механічна характеристика АД
M=2*Mk/(s/sk+sk/s);
```

```
DM=M-Ms;
```
Головна програма:
```
>> sk=0.3; Mk=120; Ms=30;
>> op=optimset('Display','iter');
>> s=fzero( 'mad', [0.001 0.9*sk], op, sk, Mk, Ms)
 Func-count      x          f(x)             Procedure
    1          0.001        -29.2            initial
    2           0.27        89.337           initial
    3       0.0672645       21.2359          interpolation
    4       0.0393641       0.958236         interpolation
    5       0.0380875      -0.0133473        interpolation
    6        0.038105    2.14751e-005        interpolation
    7        0.038105    4.75065e-010        interpolation
    8        0.038105              0         interpolation
Zero found in the interval: [0.001, 0.27].
s =
    0.0381
```

Пошук мінімуму функції z=40*x(1)+36*x(2) (у матричному запису z=C*x) методами лінеійного програмування

за обмежень: 0<=x(1)<=8; 0<=x(2)<=10; 5*x(1)+3*x(2)>=45
(у матричному запису A*x<=b)

```
>> x0=[6; 9]; xmin=[0  0]; xmax=[8  10];
>> C=[40  36]; A=[-5  -3]; b=-45;
>> options=optimset('LargeScale','off');
>> [x,fval,exitflag,output,lambda]=linprog(C, A, b, [],[],xmin, xmax, x0,options)

Optimization terminated successfully.
x =
    8.0000
    1.6667
fval =
   380
exitflag =
     1
output =
      iterations: 2
       algorithm: 'medium-scale: activeset'
    cgiterations: []
lambda =
      lower: [2x1 double]
      upper: [2x1 double]
      eqlin: [0x1 double]
    ineqlin: 12.0000
>> lambda.lower
```

```
ans =
     0
     0
>> lambda.upper
ans =
   20.0000
         0
>> lambda.ineqlin
ans =
   12.0000
```

Оптимальне керування локомотивом. Розв'язок методами нелінійного програмування

Потяг вагою 1 000 000 Н має пройти відстань L=10 км за мінімальний час. Кут ухилу в функції пройденої відстані $\alpha = \alpha_0 + \beta x$ (α_0=0.0524 рад, β=2.0944·10^{-5}, x – пройдена відстань). Коефіцієнт тертя об рейки f=0,003. Сила в'язкого спротиву повітря $F_в=kv^2$ (k=0,01). Найти залежність швидкості руху від відстані за наявності обмежень на максимальну потужність локомотиву ($P_m = 1000$ кВт) та на максимальну швидкість руху ($v_m = 15$ м/с).

Розв'язок.

Розділимо відстань L на n інтервалів довжиною Dx=L/n та вважатимемо швидкості на кожному з інтервалів незалежними змінними. Тоді шлях, пройдений на кінець кожного i-го інтервалу, буде $x_i = \sum_{j=1}^{i\le n} Dx_j = iDx$, тривалість руху на цьому інтервалі $t_i = Dx/v_i$, сила спротиву рухові $F_i = Gf\cos\alpha_i + G\sin\alpha_i + kv_i^2$, потужність $P_i = F_i / v_i$. Сумарна тривалість руху $t = \sum_{i=1}^{n} t_i \to \min$.

Файл із ЦФ loco.m:

```
function tt=loco(v,G,f,k,n,L,a0,b)
Dx=L./n;
t=Dx./v;
tt=sum(t);
```

Файл з обмеженнями lococonstr.m:

```
function [lc,lce]=lococonstr(v,G,f,k,n,L,a0,b)
%Constraints of locomitive (for everyone interval)
Dx=L/n;
x=(1:n)*Dx;
alpha=a0+b.*x;%Angle of rise
Wm=5e10;%W*sec
Pm=1e6;%W
F=G.*f.*cos(alpha)+G.*sin(alpha)+k.*v.^2;%Force
P=F.*v;%Capacity
lc(2:n+1)=P-Pm;
```

```
lce=[];
```
Файл сценарію для 50 інтервалів:
```
%Script for optimal control of locomotive
n=50;
L=10000;
Dx=L/n;
x=(1:n)*Dx;%Current way
a0=0.0524 ;
b=2.0944e-5;
alpha=a0+b.*x;%Angle of rise
G=1e6;%Weight
k=0.01;
f=0.003;
v0=ones(1,n);
lb=zeros(1,n);
vm=15;%m/sec
ub=v0*vm;
oplo=optimset('Display','final','LargeScale','off');
[v,tt,exitflag,output]=fmincon(@loco,v0,[],[],[],[],lb,ub,@lococonstr,oplo,G,f,k,n,L,a0,b);
disp('Velocity, m/sec:');disp(v);
disp('Time, sec:');disp(tt);
disp(output);
F=G.*f.*cos(alpha)+G.*sin(alpha)+k.*v.^2;%Force
P=F.*v;
plot(x,alpha*10,x,v,x,P/100000);
legend('alpha*10, degree','v, m/sec','P*10, MW');
```

Апроксимація

Апроксимація числового матеріалу функцією довільного вигляду за допомогою функції nlinfit

Числовий матеріал:
```
x =[0; 0.0001; 0.0005; 0.0010; 0.0017; 0.0025; 0.0040; 0.0050; 0.0075; 0.0100; 0.0150;
0.0200; 0.0300; 0.0500];
y=[1.0000; 0.9882; 0.9738; 0.9631; 0.9520; 0.9420; 0.9270; 0.9186; 0.9008; 0.8859;
0.8611; 0.8404; 0.8058; 0.7516];
```
З урахуванням характеру числового матеріалу залежність $y(x)$ доцільно шукати у вигляді: $y = \dfrac{1 + b_1 x^2 + b_2 \sqrt{x}}{1 + b_3 x}$.

Ця залежність розраховується в файлі `ud.m`:
```
function f=ud(b,x)
f=(1+b(1)*x.^2+b(2)*x.^0.5)./(1+b(3)*x);
```
Пошук коефіцієнтів здійснюється у сценарії udscr.m (повторити виконання кілька разів до повторення результатів):
```
b0=[1 1 1]
```

```
k=0;
while k<20
   [B,r]=nlinfit(x,y,'ud',b0);
   yfit=ud(B,x);
   R=sqrt(sum(r.^2)/length(x));
   R1=R;
   b0=B';
   k=k+1;
end
```

disp(k)

```
disp(R*1000)
disp('B=')
disp(B)
plot(x,y,'o',x,yfit)
```

Апроксимувати залежність y=sin(x)

у діапазоні x=-3:.1:3 поліномом третього ступеня

```
>> x=-3: .1:3; y=sin(x);
p=polyfit(x,y,3) %Коефіцієнти полінома
p =
  -0.0968    0.0000    0.8712   -0.0000
>> f=polyval(p,x);
x=-4:.2:4;y=sin(x);f=polyval(p,x);plot(x,y,'o',x,f)
```

Те ж саме для зашумленої залежності з побудовою 95% довірчої зони

```
>>  x=-3: .1:3; y1=sin(x)+normrnd(0, .1, size(x)); [p1,s]=polyfit(x,y1,3)
p1 =
  -0.0949    0.0051    0.8614   -0.0149
>> [f1,d]=polyval(p1,x,s);
>> figure;plot(x, y1,'o', x, f1, x, f1+2*d, x, f1-2*d)
```

Апроксимація механічної характеристики двигуна

Знайти швидкість ідеального холостого ходу двигуна та жорсткість його механічної характеристики за результатами вимірювань:

```
M =[ 12    17    25    31    39    43];
w=[150.61  149.28  148.26  147.07  145.09  143.37].
```

Побудувати експериментальні точки та механічну характеристику

```
>> [p1,s]=polyfit(M,w,1);
>> p1
p1 =
  -0.2181  153.3510
>> beta=-1/p1(1)
beta =
  4.5846
>> f1=polyval(p1,M); plot(M, w,'o', M, f1)
```

$\omega_0 = 153.35$; $\beta = 4.5846$.

Апроксимувати експериментальну залежність y=f(x):

x = [1 2 3 4 5 6]; y = [5.5 43.1 128 290.7 498.4 700.4] поліномами другого та третього ступеня. Знайти коефіцієнти поліномів, побудувати графіки експериментальних точок та порівняти результати.

```
x = [1 2 3 4 5 6]; y = [5.5 43.1 128 290.7 498.4 700.4];
p = polyfit(x,y,2);
x2 = 1:.1:6;
y2 = polyval(p,x2);
plot(x,y,'o',x2,y2)
res2=polyval(p,x)-y;
p3 = polyfit(x,y,3);
y3 = polyval(p3,x2);
y3 = polyval(p3,x2);res3=polyval(p3,x)-y;
figure;plot(x,y,'o',x2,y3);figure;plot(x,res2,'o',x,res3,'*')
```

Багатофакторна лінійна регресія

У файлі dvig.mat збережено дані про параметри 74 електричних двигунів серії 4ПФ: змінна X – потужність, кВт (2 стовпчик), швидкість, об/хв (3 стовпчик), напруга, В (4 стовпчик); змінна m – маса двигуна, кг. З використанням функції regress знайти рівняння регресії, яке пов’язує масу двигуна з його з потужністю, швидкістю та напругою. Пояснити різні знаки коефіцієнтів. Визначити частку дисперсії маси, обумовлену зміною зазначених трьох параметрів P, n, U.

```
>>load dvig; [b,bint,r,rint,stats] = regress(m,X);
>> stats
stats =
    0.8536  137.9594         0
>> b
b =
  113.6545
    6.2655
   -0.2444
    0.7460
>> bint
bint =
  -11.4638  238.7729
    5.5548    6.9762
   -0.3186   -0.1702
    0.3877    1.1043
```

Рівняння регресії:

$$m = 113.65 + 6.2655P - 0.2444n + 0.746U$$

```
>> load dvig
>> rstool(X(:,2:4),m)
```

Однофакторна нелінійна регресія (1)

Апроксимувати залежність питомої маси (кг/кВт) двигунів 4ПФ (файл dvig.mat) від їх потужності за допомогою функції nlinfit. Ту ж саму задачу розв'язати за допомогою функції nlintool.

Функція fun1.m:

```
function fun1=fun1(b,x)
x=x(:,1);
fun1=b(1)+b(2)./x+b(3)./x.^2+b(4).*x;
```

Основна програма:

```
>> load dvig
>> P=X(:,2);
>> y1=m./P;
>> [b,R,J]=nlinfit(P,y1,@fun1,[1 1 1 1]);
>> b
b =
   11.9238
   38.4652
   60.8309
   -0.0286
```

Рівняння регресії: $11.92 + 38.46/P + 60.83/P^2 - 0.0286P$.

```
>> nlintool(P,y1,@fun1,[1 1 1 1])
```

Однофакторна нелінійна регресія (2)

За результатами вимірювань моменту та ковзання асинхронного двигуна знайти його критичний момент та критичне ковзання.

```
s =[0.0227  0.0364  0.0638  0.0832  0.0909  0.0969  0.1679  0.1739];
M=[4.9875 14.6699 23.5294 31.1804 40.0000  44.1176 48.0000 49.3469].
```

Inline-функція для розрахунку спрощеної формули Клоса:

```
fk=inline('2.*b(1)./(s./b(2)+b(2)./s)','b','s')
fk =
     Inline function:
     fk(b,s) = 2.*b(1)./(s./b(2)+b(2)./s)
>> b=nlinfit(s,M,fk,[100 .3])
b =
   51.4615
    0.2221
```

M_k=51.4615 Нм; s_k=0.2221.

Багатофакторна нелінійна регресія

Двигун постійного струму працює зі змінними напругою якоря, струмами збудження та якоря. Експериментатор, не маючи змоги задавати ці змінні, може лише вимірювати їх та швидкість. Результати вимірювань в усталеному режимі задані у матриці mes, стопці якої відповідають напрузі, струму якоря, струму збудження та швидкості. Знайти за допомогою функції nlinfit активний опір обмотки якоря та коефіцієнт $k = k_f c$, де $k_f = \Phi / I_f$ – коефіцієнт пропорційності між струмом збу-

дження та потоком, c – конструктивний коефіцієнт двигуна. Для зменшення впливу випадкових похибок кількість вимірювань замість мінімально достатніх 2 (що дорівнює кількості шуканих коефіцієнтів) була збільшена до 6:

```
mes =  [205.4800   21.5300    0.2560  838.6367
   83.3500   53.6700    0.4330  141.4928
  202.7500   81.0000    0.9120  196.6487
  106.3200   17.3000    1.0000  107.9094
   30.7700   35.8200    0.2850   47.1794
   30.7700   35.8200    0.2850   56.3044]
```

Швидкість пов'язана із зазначеними змінними через рівняння $\omega = \left(U - I_a r_a\right) \big/ kI_f$.

Воно реалізоване в inline-функції:

```
>> ww=inline('(X(:,1)-X(:,2).*b(1))./b(2)./X(:,3)', 'b','X')
ww =
     Inline function:
     ww(b,X) = (X(:,1)-X(:,2).*b(1))./b(2)./X(:,3)
>> [B,r]=nlinfit(mes(:,1:3),mes(:,4),ww,[1 1])
B =
    0.4947
    0.9078
r =
    0.2938
   -3.0000
    0.1602
    0.2195
   -3.2532
    5.8717
```

r_a =0.4947 Ом, k =0.9078 Ом.

Розкладення полігармонічної періодичної функції у ряд Фур'є

```
N=2^12;
f=100;
t=linspace(0,1/f,N);
y=27+100*sin(2*pi*f*t+pi/2)+50*sin(2*pi*5*f*t)+25*sin(2*pi*7*f*t);
figure(1);plot(t,y);grid
fy=fft(y);
c=[zeros(1,N/2)];
c(1)=fy(1)/N;
c(2:N/2)=abs(fy(2:N/2))/N*2; % Amplitudes of garmonics
ff=(0:N/2-1)/t(end); % Frequency
figure(2);h=stem(ff,c);grid % Spectrum of amplitudes
delete(h(1));
disp(' Coefficient of distortions:')
THD=c(2)/sqrt(sum(c(2:end).^2))
```

6 ЗАДАЧІ ДЛЯ САМОСТІЙНОГО РОЗВ'ЯЗАННЯ

Основи MATLAB

1. Створити вектор, який складається з цілих чисел, рівномірно розподілених в діапазоні від 5 до 12 з кроком 2. Визначити довжину отриманого вектора.

2. Створити матрицю розміром 4×3, яка складається з випадкових чисел, рівномірно розподілених в діапазоні від 5 до 10. Для цієї матриці розрахувати середні квадрати рядків, стовпців та всього масиву.

3. Створити матрицю нулів розміром 3×5.

4. У масиві m1=[8 4 0;5 0 2; 0 0 4] за допомогою функції find знайти елементи, більші від 0 и менші від 5.

5. Створити inline-функції та розрахувати їх для заданих вхідних параметрів:
$f(x)=3e^{2x}$ при x=3;
f(t,a)=a·cost при t=2; a=7;
$f(x1, x2, x3)=3(x_1-4)^2+5(x_2+3)^2+7(2x_3+1)^2$ при $x_1=1$; $x_2=2$; $x_3=3$.
6. Створити массив структур із 2-3 записів для зберігання інформації щодо результатів перепису населення. У полях розташувати інформацію:

- П.І.Б.;
- Рік народження;
- склад сім'ї у вигляді масиву [N, nd, nd18] (N- розмір сім'ї; nd – загальна кількість дітей; nd18 - кількість дітей віком до 18 років);
- загальний річний дохід;
- освіта;
- професія.

Сформувати масив із усіх значень поля «загальний річний дохід»

7. Розрахувати:

- ***x***, що змінюється у межах від 1 до 10 з кроком 1 ;
- ***y=2+4x***;
- випадковий вектор ***e*** тієї ж довжини, розподілений за нормальним законом із середнім 0 і середньоквадратичним відхиленням 1.

Побудувати залежності:

- y=f1(x) - суцільна зелена лінія;
- y+e =f2(x) - червоні зірочки.

Параметри графічного вікна: межі осі X (-1...12), Y (0...45); вивести сітку, легенду і заголовки осей.

8. Розробити програму керування положенням і кольором двох заплат. Перша заплата рухається колом у межах графічного вікна, положення другої змінюється випадковим чином довкола начального положення. Колір обох змінюється циклі-

чно (частки основних кольорів RGB міняються від 0 до 1). Закон зміни кольору різний для різних заплат. По по закінченні циклу заплаты знищити.

9. Розробити сценарій побудови ліній рівня функції двох змінних та її поверхні.
Вимоги до функції:

- функція розраховується окремому М-файлі, який повертає значення функції та текстові коментарі до неї (назву функції функции та її формулу);
- як приклад використати функції Розенброка `100*(x2-x1^2)^2+(1-x1)^2`

  та Ізона-Фентона $f(x) = 0{,}1 \cdot \left[12 + x_1^2 + \frac{1 + x_2^2}{x_1^2} + \frac{x_1^2 x_2^2 + 100}{(x_1 x_2)^4} \right]$

Вимоги до сценарию:

- им'я функції та межі її зміни X1 і X2 задаються інтерактивно з клавіатури;
- кількість значень X1 і X2 не менша від 100;
- після першої побудови ліній рівня здійснюється інтерактивний выбір потрібних ліній, перебудова графика та указівка мишшю місць для міток рівнів;
- у графічному вікні задаються сітка, заголовки осей, а також текстові коментарі функції як заголовок вікна.

10. Створити М-функцію для обробки вектора довільної довжини. Результати обробки:

- довжина вектора;
- сума елементів вектора;
- елемент з мінімальнмм значенням;
- елемент з максимальним значенням;
- розмах значень елементів;
- среднее значение элементов;
- середньоквадратичне відхилення значень елементів.

11. Розробити програму розрахунку миттєвої випрямленої напруги однофазного мостового випрямляча для заданих кута керування та амплітуди напруги живлення. Побудувати графіки напруг живлення та випрямленої у функції часу протягом періоду.

ОПТИМІЗАЦІЯ В MATLAB

12. За допомогою функції fzero знайти точку сталої рівноваги системы "асинхронний двигун – вентилятор" із механічними характеристиками

$M_a = \frac{2M_k}{\frac{s}{s_k} + \frac{s_k}{s}}$ та $M_v = M_0 + k\omega^2$. Константи M_k=120 Нм; $s_k = 0{,}3$; ω_0=157 с$^{-1}$;

M_0=10 Нм; $k = 0{,}01$ передати до m-функції через функцію fzero.

13. Знайти найменьшу відстань між параболою $y = x^2$ та від'ємною гілкою гіперболи $y = -1/x$ за допомогою функції fminsearch.

14. Знайти прямокутник, вписанный в коло радіусу r=5, який має найбільшу площу. Рекомендації:

1) площа прямокутника $S = \sqrt{(p-a_1)(p-a_2)(p-a_3)(p-a_4)}$,

де $p = (a_1 + a_2 + a_3 + a_4)/2$ – полупериметр; $a_1, \ldots a_4$ – боки прямокутника.

2) координати точок на колі зручно задавати у непрямому вигляді як $x = r\cos(t)$; $y = r\sin(t)$.

15. Апроксимація коливної ланки з передатною функцією $W(p) = \dfrac{3}{3p^2 + 3p + 1}$ аперіодичною $W(p) = \dfrac{k}{Tp+1}$. Критерій: мінімум середньоквадратичного відхилення вихідних сигналів. Фактори: k, T. Моделі обох ланок реализувати в Simulink. Тривалість моделювання – не менше 20 с. Використати функцію fminsearch.

16. Розробити програму мінімізації функцій багатьох змінних з обмеженнями типу «нерівність» із використанням методу штрафних функцій. Штрафна функція типу «квадрат зрізки». Початкове значення штрафу R=1, коэфіцієнт збільшення штрафу К=4. У процесі мінімізації перетвореної функції відгуку використати функції fminsearch або fminunc. Вихідні дані для мінімізації (вводяться інтерактивно): координати початкової точки; им'я файла, що описує цільову функцію та обмеження.

Рекомендації:

- цільова функція та обмеження розраховуються у спільному М-файлі;
- для розрахунку перетвореної з урахуванням штрафів цільової функції організувати окрему М-функцію.

ЛІТЕРАТУРА

1. Лозинський А.О., Мороз В.І., Паранчук Я.С. Розв'язання задач електромеханіки в середовищах пакетів MathCAD і MATLAB. – Львів: Видавн. Держ. універс. «Львівська Політехніка», 2000.-166 с.
2. Дьяконов В.П. MATLAB: Учебный курс. – СПб: Питер, 2001.-560 с.
3. Потемкин В.Г. Система инженерных и научных расчетов MATLAB 5.x. В 2-х т.т. – М.: ДИАЛОГ-МИФИ, 2000.
4. Потемкин В.Г., Рудаков П.И. MATLAB 5 для студентов. – М.: ДИАЛОГ-МИФИ, 1999. – 448 с.
5. Мартынов Н.Н., Иванов А.П. MATLAB 5.x. Вычисления, визуализация, программирование. – М.: КУДИЦ-ОБРАЗ, 2000.
6. Мартынов Н.Н. Введение в MATLAB 6. – М.: Кудиц-образ. 2002.
7. Рудаков И.И., Сафонов В.И. Обработка сигналов и изображений. MATLAB 5.x. – М.: ДИАЛОГ-МИФИ, 2000.
8. Дьяконов В.П. Компьютерная математика. Теория и практика. – М.: Нолидж, 2001.- 1296 с.
9. Дьяконов В., Круглов В. Математические пакеты расширения MATLAB: Специальный справочник. – СПб.: Питер, 2001. - 480 с.
10. Дьяконов В.П. Simulink 4. Специальный справочник. – СПб.: Питер, 2002. - 528 с.

11. Гультяев А. Визуальное моделирование в среде MATLAB: Учебный курс. – СПб.: Питер, 2000. - 432 с.
12. Потемкин В.Г. Инструментальные средства MATLAB 5.X. – М.: ДИАЛОГ-МИФИ, 2000 – 336 с.
13. Ануфриев И. Самоучитель MatLab 5.3/6.x. – БХВ-Петербург. 2002.