**Universitatea Tehnică "Gheorghe Asachi" din Iasi**
**Facultatea Electronică, Telecomunicaţii şi Tehnologia Informaţiei**
**Disciplina „Sisteme automate de control", Anul V / SEIII**

# LABORATORUL #1
# INTRODUCERE IN MATLAB. MODELAREA SISTEMELOR CU MATLAB.

1. Introducere în MATLAB
   a. Vectori.
   b. Funcţii.
   c. Grafice.
   d. Polinoame.
   e. Matrici.
2. Modelarea unui motor de c.c. pentru analiza în frecvenţă.
3. Modelarea unui sistem de control "*cruise control*".
4. Modelarea unei suspensii pentru un vehicul.
5. Mini-Proiect.

**Anexă** : Tabel de funcţii MATLAB.

## 1. Introducere în MATLAB

MATLAB este un program interactiv pentru calcularea şi vizualizarea unor date numerice. MATLAB este folosit foarte mult la rezolvarea unor probleme inginereşti, în special la analiza şi proiectarea sistemelor de control. In acest scop, au fost create numeroase pachete de instrucţiuni (*toolboxes*) care definesc funcţii speciale pentru diverse clase de aplicaţii. Lucrul in MATLAB este posibil:

- direct, prin scrierea secvenţiala a instrucţiunilor pe ecran urmată de execuţia lor imediată, sau
- prin intermediul unui fişier de comenzi "*.m" care este apelat din acelaşi ecran de comenzi.

*Următoarea secvenţă de instrucţiuni este disponibilă in fişierul MATLAB L1_intro.m.*

```
% _____________________________________________________
clear;
echo on
% VECTORI
% Sa incepem prin definirea unui vector. Sa alegem « a » ca numele acestei variabile in format
vectorial
% si sa definim aceasta variabila ca fiind egala cu elementele vectorului scrise intre doua paranteze
% patrate.
a=[1 2 3 4 5 6]
pause % apasati orice tasta pentru a continua.

% Acum sa cream automat un vector cu elemente intre 1 si 15, distantate in incrementi de 3 unitati.
t=0:3:15
pause    % apasati orice tasta pentru a continua.

% Operatiile cu vectori sunt la fel de simple ca si crearea lor.
% Adunarea unei constante scalare
y=t+2
pause    % apasati orice tasta pentru a continua.

% adunarea a doi vectori
```

```
c=y+a
pause    % apasati orice tasta pentru a continua.

% FUNCTII
% La fel ca in matematica, putem defini functii pe baza unui set de functii de baza, definit deja in
% MATLAB. Exemple de astfel de functii predefinit, includ functiile trigonometrice.
% Constante des folosite in problemele matematice, sau in programare sunt de asemeni pre-definite
% in MATLAB.
% Exemplul urmator ne ajuta sa intelegem utilizarea unei functii predefinite si folosirea unei
constante.
% Simpla apelare a functiei ne va da rezultatul.
sin(pi/4)
pause    % apasati orice tasta pentru a continua.

% Daca dorim sa captam rezultatul in alta variabila pentru o folosire ulterioara,
% atunci atribuim direct:
d=sin(pi/4)
pause    % apasati orice tasta pentru a continua.

% Daca dorim mai tarziu sa apelam aceeasi variabila, aceasta va deveni disponibila prin
% simpla utilizare
d
pause    % apasati orice tasta pentru a continua.


% Daca nu sunteti sigur de folosirea unei functii, puteti la orice moment sa utilizati help de la
% consola MATLAB. De exemplu :
help sin
pause    % apasati orice tasta pentru a continua.

% GRAFICE
% Probabil cel mai mare avantaj in folosirea pachetului MATLAB pentru rezolvarea
% problemelor ingineresti, il reprezinta capabilitatea grafica.
% Vom ilustra acest lucru prin folosirea catorva dintre conceptele invatate deja.
% Sa reprezentam functia sinus pe un interval egal cu 6.4 radiani (pi=3.14rad, o perioada este 2pi)
t=0:0.2:6.4;
y=sin(t);
plot(t,y)

pause    % apasati orice tasta pentru a continua.

% Impreuna cu comanda plot, avem o multitudine de posibilitati pentru definirea axelor,
% sau pentru re-aranjarea in pagina a graficelor. Observati cum se schimba graficul dupa
% executarea fiecarei comenzi.
axis([1 10 -2 2])
pause    % apasati orice tasta pentru a continua.

subplot(3,1,1); plot(t,y)
pause    % apasati orice tasta pentru a continua.

% POLINOAME
% Polinoamele sunt definite in MATLAB pe baza unor vectori ce contin coeficientii
% in ordine descrescatoare (vector al coficientilor).
```

```
% de exemplu polinomul f(x)=(s^4)+(s^3)-(s^2)-7*s+1
x=[1 1 -1 -7 1]
pause    % apasati orice tasta pentru a continua.

% Un mare avantaj al utilizarii programului MATLAB in locul calculelor manuale
% este dat de posibiliatea rezolvarii solutiilor unei ecuatii polinomiale.
roots([1 1 -1 -7 1])
pause    % apasati orice tasta pentru a continua.

% sau se poate folosi direct
roots(x)
pause    % apasati orice tasta pentru a continua.
% Putem de asemeni sa consideram operatii cu polinoame.
% De exemplu multiplicarea a doua polinoame, se face cu instructiunea conv
y=[1 2]
conv(x,y)
pause    % apasati orice tasta pentru a continua.

% aceasta proprietate de inmultire a doua polinoame se foloseste citeodata la definirea
% unor functii de transfer (Laplace), sau la multiplicarea intre doua functii de transfer
%
% Impartirea a doua polinoame se face prin instructiunea deconv, care ne va furniza catul
% si restul impartirii.
% Vom folosi aceleasi polinoame pentru exemplu :
 [Q,R] = deconv(x,y)
pause    % apasati orice tasta pentru a continua.

% operatii simple, de adunare sau scadere a polinoamelor se fac direct ca operatii intre vectori.
%
%
% MATRICI
% Matricile sunt elemente foarte importante, des utilizate in sistemele de control automat.
% Introducerea matricilor in MATLAB se face foarte asemanator cu introducerea vectorilor,
% fiecare rand fiind despartit prin “;” de celelalte.
% exemplu
A=[1 2 3; 4 5 6]
pause    % apasati orice tasta pentru a continua.
B=[7 8; 7 9; 7 1]
pause    % apasati orice tasta pentru a continua.

% operatiile cu matrici cele mai utilizate sunt
% transpusa
B'
pause    % apasati orice tasta pentru a continua.

% adunarea
A+B'
pause    % apasati orice tasta pentru a continua.

% inmultirea
A*B
pause    % apasati orice tasta pentru a continua.
```

```
% Observati cu atentie deosebirea intre AB si BA
B*A
pause    % apasati orice tasta pentru a continua.

% Inversa unei matrici
E=inv(A*B)
pause    % apasati orice tasta pentru a continua.

% valorile proprii ale unei matrici
eig(E)
pause    % apasati orice tasta pentru a continua.

% Sa ne aducem aminte ca o matrice patratica E (nxn), poate fi caracterizata printr-o ecuatie
polinomiala
% denumita ecuatie caracteristica, care se obtine prin calcularea     det(sI-E)=0
% Putem determina coeficientii unei ecuatii caracteristice ce corespunde unei matrici.
% Aceasta va fi definita sub forma unui vector, dupa cum am vazut anterior.
p=poly(E)
pause    % apasati orice tasta pentru a continua.

% Radacinile acestei ecuatii caracteristice sunt de asemeni valorile proprii ale matricei initiale E.
% Urmatoarele doua instructiuni trebuie sa dea acelasi rezultat.
roots(p)
eig(E)
pause    % apasati orice tasta pentru a continua.

%
%
% Fisierele de tip *.m
% Pentru a simplifica lucrul cu MATLAB, seturi de instructiuni se pot grupa in mici programe
% si salva ca fisiere *.m.
% Aceste fisiere pot fi apelate si executate de la consola MATLAB, prin
% simpla apelare a numelui fisierului.
% Se poate folosi un editor special pentru scrierea sau modificarea acestor instructiuni MATLAB.
% _____________________________________________________________
```

Lucru în clasă ( rezultatele se includ în raportul de laborator)

Rulaţi programul "L1_intro.m" în mediul MATLAB şi urmăriţi utilizarea fiecărei instrucţiuni. Răspundeţi la următoarele instrucţiuni după parcurgerea întregului program.

(a) Reprezentaţi un sistem trifazat sinusoidal cu ajutorul instrucţiunilor subplot şi plot, pe trei grafice diferite, în aceeaşi figură.
(b) Verificaţi existenţa în spaţiul MATLAB a variabilelor ***x*** şi ***y***. Demonstraţi că rădăcinile lui ***x*** se află printre rădăcinile produsului ***x**y***. Folosiţi instrucţiunile ***conv*** şi ***roots*** pentru aceasta.

## 2. Modelarea unui motor de c.c. pentru analiza în frecvenţă.

Multe sisteme electromecanice au în componenţă un motor de curent continuu (c.c.), capabil să producă mişcare de rotaţie şi să transmită cuplu la un arbore rotitor. Modelarea unui motor de c.c. are doua componente: una electrică şi una mecanică.

In circuitul electric avem reprezentarea înfăşurării printr-o inductanţă şi o rezistenţă de pierderi, urmată de o tensiune contra-electromotoare a cărei valoare este data de o relaţie de proporţionalitate cu turaţia (viteza unghiulară) a motorului la un moment dat. Viteza unghiulară se poate calcula prin derivata coordonatei unghiulare, notată de obicei cu ***theta***.

In circuitul mecanic avem o diagramă a forţelor, într-un sistem fără masă, caracterizat prin momentul de inerţie J, constanta de frecare dinamică b, şi constanta de dependenţă a cuplului de curentul electric aplicat.

In exemplul considerat în laborator, avem următoarele date numerice:

- moment de inerţie J = 0.01 kg.m^2/s^2
- constanta de frecare dinamică b = 0.1 Nms
- Constanta cuplului Kt = 0.01 NM/Amp
- Constanta electrica Ke = 0.01 Nm/Amp
- Rezistenţa electrică R = 1 ohm
- Inductanţa electrică L = 0.5 H
- Tensiunea de intrare V variabilă
- Poziţia sau coordonata unghiulară de rotaţie ***theta***
- Viteza si acceleraţia unghiulară vor rezulta ca derivate de primul şi al doilea ordin a acestei coordonate unghiulare.

Relaţiile matematice care caracterizează modelul motorului de c.c. sunt

- Cuplul este dat de T = Kt * i
- Tensiunea electrică este e = Ke * theta' ( ' reprezintă aici derivata)
- Circuitul electric este caracterizat de relaţia Kirchhoff

$$L \cdot \frac{di}{dt} + R \cdot i = v - K_e \cdot \frac{d\theta}{dt}$$

- Circuitul mecanic este caracterizat prin relaţia Newton de mişcare de rotaţie

$$J \cdot \ddot{\theta} + b \cdot \dot{\theta} = K_t \cdot i$$

Modelarea dinamică se face prin definirea unei funcţii de transfer Laplace, bazată pe ecuaţiile:

$$\begin{cases} s \cdot (J \cdot s + b) \cdot \Theta(s) = K_t \cdot I(s) \\ (L \cdot s + R) \cdot I(s) = V(s) - K_e \cdot s \cdot \Theta(s) \end{cases}$$

Dacă eliminăm curentul, putem scrie funcţia de transfer intrare-ieşire (tensiune electrică –> viteza).

$$\frac{s \cdot \Theta(s)}{V(s)} = \frac{K_t}{(J \cdot s + b) \cdot (L \cdot s + R) + K_t \cdot K_e}$$

Să scriem această funcţie de transfer în MATLAB.

*Lansaţi aici programul L1_motor.m*

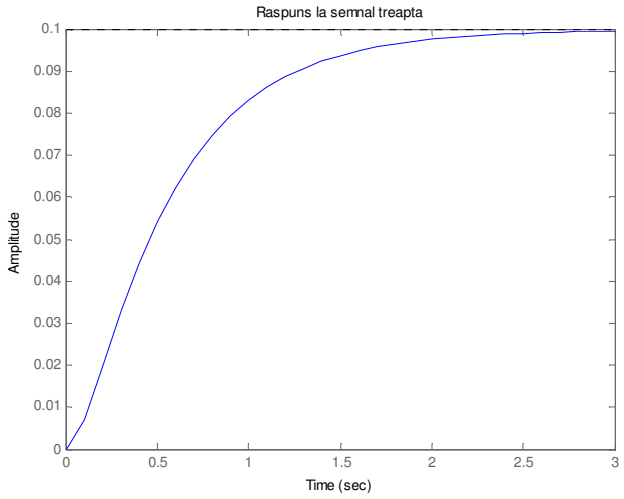
```
% ____________________________________________________
clear;
echo on;
% initializam valorile numerice
J=0.01;
b=0.1;
Kt=0.01;
Ke=0.01;
R=1
L=0.5;
% definim functia de transfer ca un raport de doua polinoame
num=Kt
den=[(J*L) ((J*R)+(L*b)) ((b*R)+Kt*Ke)]
pause    % apasati orice tasta pentru a continua.

M=tf(num,den)
pause    % apasati orice tasta pentru a continua.


% Sa observam acum functionarea acestui motor ca un sistem in bucla deschisa
% In acest scop, sa observam un raspuns la un semnal treapta
step(M, 0:0.1:3)
title('Raspuns la semnal treapta')
pause    % apasati orice tasta pentru a continua.

% Aceasta se poate interpreta ca la aplicarea unei tensiuni de 1 volt,
% se obtine o turatie maxima de 0.1rad/sec.
%
%
% ____________________________________________________
echo off;
```

Execuţia programului produce figura alăturată.



Lucru în clasă ( rezultatele se includ în raportul de laborator)

Putem salva fişierul cu un nume nou şi modifica parametrii, pentru a observa cum se schimbă funcţia de transfer şi răspunsul sistemului la semnal treaptă. In acest scop, putem folosi comanda „*hold on*" în fereastra MATLAB pentru a reţine imaginea răspunsului la semnal treaptă original, peste care vom suprapune noile rezultate.

(c) Modificaţi momentul de inerţie (J), considerând o valoare de 10 ori mai mare (J=0.1). Observaţi cum se schimbă coeficienţii funcţiei de transfer, şi cum se schimbă răspunsul sistemului la semnal treaptă.
(d) Reveniţi la valoarea iniţială a momentului de inerţie. Modificaţi acum valoarea inductanţei armăturii, considerând o inductanţă de 5 ori mai mică (L=0.1). Observaţi cum se schimbă coeficienţii funcţiei de transfer, şi cum se schimbă răspunsul sistemului la semnal treaptă.

## 3. Modelarea unui sistem de control "*cruise control*".

In vederea proiectării şi implementării unui sistem automat de menţinere constantă a vitezei unui autovehicul, trebuie să avem un model matematic capabil să descrie mişcarea vehiculului. In acest scop, se consideră o serie de ipoteze simplificatoare:

- Se considera un vehicul fără formă sau greutate distribuită,
- Se neglijează inerţia roţilor,
- Se consideră doar frecare pe direcţia de deplasare a vehiculului, ca fiind proporţională cu viteza vehiculului prin intermediul unei constante „***b***".

Vom începe modelarea prin considerarea legii de mişcare Newton pentru un vehicul de masa ***m***, ce se deplasează cu viteza ***v***, sub acţiunea unei forţe dată de motor ***F***, care se poate considera ca semnal de intrare în sistemul nostru.

$$m \cdot a = F - b \cdot v \Leftrightarrow m \cdot \dot{v} + b \cdot v = u \Leftrightarrow m \cdot s \cdot V(s) + b \cdot V(s) = U(s) \Leftrightarrow \frac{V(s)}{U(s)} = \frac{1}{m \cdot s + b}$$

In exemplul considerat în laborator, avem următoarele date numerice:

- Masa vehiculului m = 1000kg,
- Constanta de frecare dinamică b = 50Nsec/m,
- Forţa aplicată de vehicul (considerată ca mărime de intrare) u = 500N.

*Ce ar presupune o problemă de proiectare a unui sistem automat de control?*

Vom vedea prin executarea programului MATLAB că la aplicarea acestei forţe maxime de 500N, vehiculul poate ajunge la o viteză staţionară de 10m/sec. Un autoturism ar trebui să fie capabil să accelereze până la acea viteză în mai puţin de 5 secunde. Schimbarea vitezei de la starea de repaus la această viteză s-ar putea produce cu o oarecare supra-creştere (supra-reglare). Dorim să limităm aceasta la doar 10% pentru a nu produce prea mari oscilaţii în operarea vehiculului. De asemeni, s-ar putea sa nu putem menţine viteza finală la o valoarea extrem de precisă, să zicem că putem tolera o eroare staţionară de 2%.

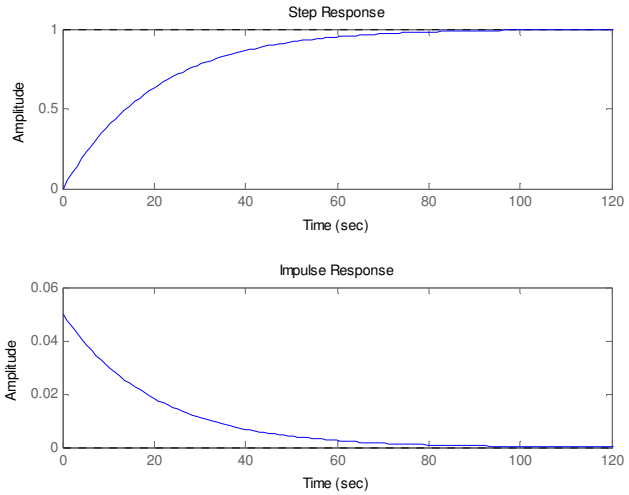
Toate aceste date numerice reprezintă cerinţe de proiectare din considerente practice.

Vom rula acum un program MATLAB să vedem cum se comportă sistemul fără nici o buclă de control şi cât de departe ar fi faţă de aceste cerinţe de proiectare.

*Lansati aici programul L1_cruise.m*

```
% ____________________________________________________
clear;
echo on;
% Initializam valorile numerice (atentie la unitatile de masura)
m=1000
b=50
u=500
pause    % apasati orice tasta pentru a continua.
%
% Definim functia de transfer
num=[1];
den=[m b];
sys=tf(num,den)
pause    % apasati orice tasta pentru a continua.
%
% Sa analizam raspunsul sistemului la un semnal treapta
% In acest caz ar corespunde aplicarii fortei u=F=500, la momentul intial
% Se observa ca functia „step” reprezinta aplicarea unei trepte unitare,
% iar noi avem nevoie de o treapta de 500N
subplot(2,1,1)
step(u*sys);
%
% Observam ca sistemul este mult mai lent decat se cere in datele de proiectare
% Acest lucru va justifica folosirea unui control in bucla inchisa, si a unei compensari dinamice.
pause    % apasati orice tasta pentru a continua.
subplot(2,1,2)
%
% De multe ori in descrierea sistemelor dinamice, ne punem problema stabilitatii sistemelor.
% Practic ne intereseaza daca sistemul nu va oscila singur din cauza unor vibratii sau oscilatii interne.
% Fara a intra in detalii aici, sa retinem ca o informatie simpla despre stabilitatea unui sistem
% poate fi data de raspunsul la impuls.
% In MATLAB, putem determina raspunsul la impuls prin instructiunea 'impulse'.
impulse(u*sys);
pause    % apasati orice tasta pentru a continua.
% ____________________________________________________
```

Prin execuţia programului, se obţine figura următoare.




Lucru în clasă (rezultatele se includ în raportul de laborator)

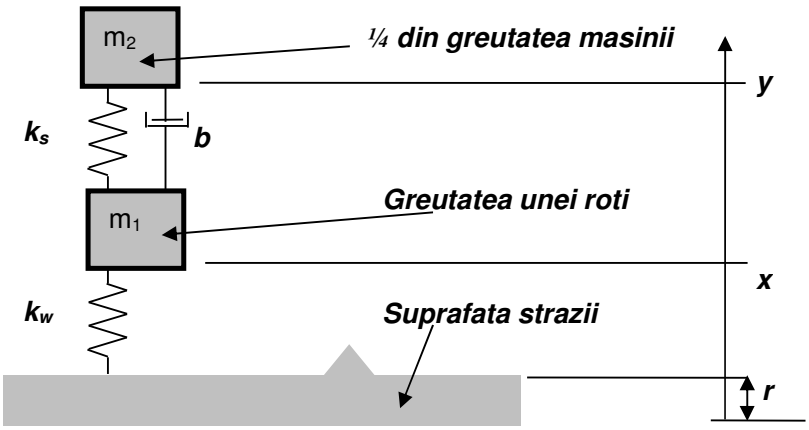
Putem salva fişierul cu un nume nou şi modifica parametrii pentru a observa cum se schimbă funcţia de transfer şi răspunsul sistemului la semnalele treaptă şi rampă. In acest caz, nu mai putem folosi comanda „*hold on*” deoarece avem două figuri la fiecare execuţie a programului. De asemeni, observăm ca nu am definit noi numele fiecărei figuri, ci am lăsat MATLAB să ne ofere nume în limba engleză.

Observaţi cum se modifică funcţia de transfer, răspunsul la treaptă şi răspunsul la impuls, dacă se modifică

(e) forţa aplicată la ***u=50***,
(f) reveniţi la o forţă aplicată de ***u=500*** şi modificaţi masa la ***m=2000***.

## 4. Modelarea unei suspensii pentru un vehicul.

Vom considera un alt exemplu de sistem dinamic: suspensia unui vehicul. Un posibil model este dat in figură.



Datele numerice ce caracterizează sistemul sunt:

- Masa maşinii m1 = 1580 kg
- Masa roţii şi cauciucului m2 = 20 kg
- Constanta arcului din componenţa sistemului de suspensie ks = 130,000 N/m
- Constanta de tip arc a cauciucului kw =1,000,000 N/m
- Constanta de amortizare (fixă) din componenţa sistemului de suspensie b = 9800 Ns/m
- Forta aplicată (mărime de intrare în sistem) u.

Ecuaţiile bazate pe legea Hooke pentru arcuri ($F = - K \cdot x$, sau cu amortizare $F = - K \cdot x - B \cdot v$) şi pe prima lege a lui Newton pentru mişcare (*"Orice corp își menține starea de repaus sau de mișcare rectilinie uniformă atât timp cât asupra sa nu acționează alte forțe sau suma forțelor care acționează asupra sa este nulă."*) sunt:

$$\begin{cases} m_1 \cdot \ddot{x} = b \cdot \left( \dot{y} - \dot{x} \right) + k_s \cdot \left( y - x \right) - k_w \cdot \left( x - r \right) \\ m_2 \cdot \ddot{y} = -k_s \cdot \left( y - x \right) - b \cdot \left( \dot{y} - \dot{x} \right) \end{cases} \Rightarrow \begin{cases} \ddot{x} + \frac{b}{m_1} \cdot \left( \dot{x} - \dot{y} \right) + \frac{k_s}{m_1} \cdot \left( x - y \right) = \frac{k_w}{m_1} \cdot \left( x - r \right) \\ \ddot{y} + \frac{b}{m_1} \cdot \left( \dot{y} - \dot{x} \right) + \frac{k_s}{m_1} \cdot \left( y - x \right) = 0 \end{cases}$$

Aceste ecuaţii pot conduce la un model bazat pe funcţia de transfer Laplace:

$$\begin{cases} \ddot{x} + \frac{b}{m_1} \cdot \left( \dot{x} - \dot{y} \right) + \frac{k_s}{m_1} \cdot \left( x - y \right) = \frac{k_w}{m_1} \cdot \left( x - r \right) \\ \ddot{y} + \frac{b}{m_1} \cdot \left( \dot{y} - \dot{x} \right) + \frac{k_s}{m_1} \cdot \left( y - x \right) = 0 \end{cases} \Rightarrow$$

$$\Rightarrow \begin{cases} s^2 \cdot X(s) + s \cdot \frac{b}{m_1} \cdot \left( X(s) - Y(s) \right) + \frac{k_s}{m_1} \cdot \left( X(s) - Y(s) \right) = \frac{k_w}{m_1} \cdot \left( X(s) - R(s) \right) \\ s^2 \cdot Y(s) + s \cdot \frac{b}{m_1} \cdot \left( Y(s) - X(s) \right) + \frac{k_s}{m_1} \cdot \left( Y(s) - X(s) \right) = 0 \end{cases} \Rightarrow$$

$$\Rightarrow \frac{Y(s)}{R(s)} = \frac{\frac{k_w \cdot b}{m_1 \cdot m_2} \cdot \left( s + \frac{k_s}{b} \right)}{s^4 + \left( \frac{b}{m_1} + \frac{b}{m_2} \right) \cdot s^3 + \left( \frac{k_s}{m_1} + \frac{k_s}{m_2} + \frac{k_w}{m_1} \right) \cdot s^2 + \left( \frac{k_w \cdot b}{m_1 \cdot m_2} \right) \cdot s + \left( \frac{k_w \cdot k_s}{m_1 \cdot m_2} \right)}$$

Aceasta reprezintă o funcţie de transfer de la o perturbaţie în suprafaţa străzii la deplasarea cabinei resimţită de pasageri (ambele mărimi sunt dimensiuni). Să vedem ce date de proiectare ar avea sens în acest caz:

- Să considerăm o supra-creştere de doar 5% pentru a nu resimţi şocuri puternice,
- Să considerăm un timp de stabilizare (atingere a regimului staţionar) mai mic de 5 secunde.

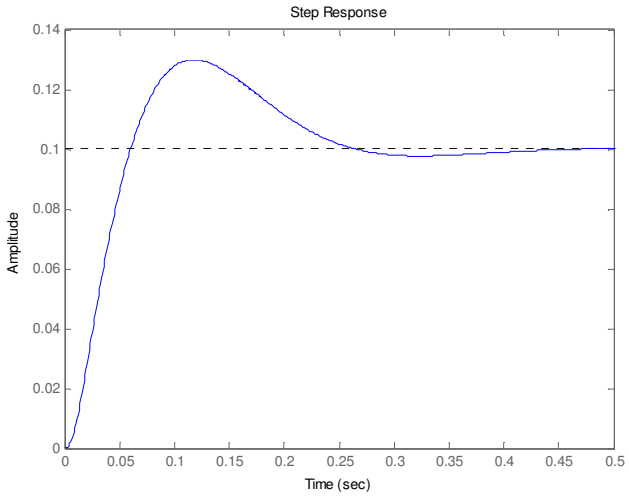
O exprimare numerică a acestor date de proiectare ar însemna că o denivelare în stradă de 10cm ar produce o mică oscilaţie de doar 5mm, urmată de o revenire la regimul staţionar în 5 secunde.

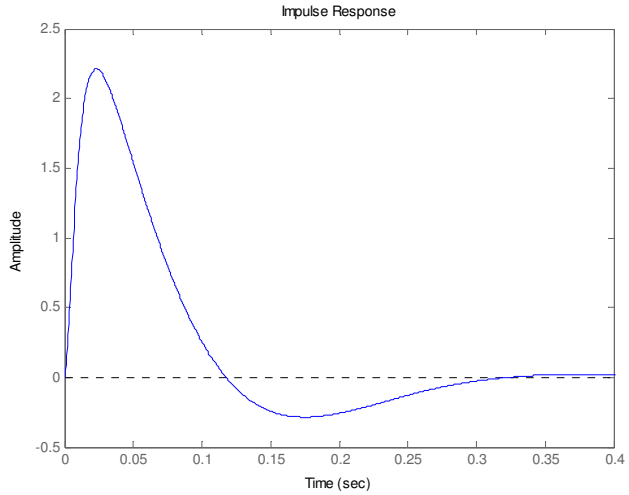
Un program scris pe baza modelului matematic ne va putea spune cât de departe suntem de aceste date de proiectare la folosirea sistemului în buclă deschisă.

*Lansati aici programul MATLAB L1_suspen.m*

```
% ____________________________________________________
clear;
echo on;
% initializam valorile numerice
m1=20;
m2=(1580-80)/4;
ks=130000;
kw=1000000;
b=9800;
%
% Sa scriem functia de transfer definita anterior pe baza unor coeficienti
% polinomiali
K=(kw*b)/(m1*m2);
a= ks/b;
b3=(b/m1)+(b/m2);
b2=(ks/m1)+(ks/m2)+(kw/m1);
b1=(kw*b)/(m1*m2);
b0=(kw*ks)/(m1*m2);
num=K*[1 a];
den=[1 b3 b2 b1 b0];
sys=tf(num,den)
pause    % apasati orice tasta pentru a continua.
%
% Sa analizam raspunsul sistemului la un semnal treapta
% Acest caz ar corespunde aplicarii unei perturbatii u=10cm=0.1m,
% la momentul initial
% Se observa ca functia „step” reprezinta aplicarea unei trepte unitare,
% iar noi avem nevoie de o treapta de 0.1m
u=0.1;
step(u*sys);
% Observam ca sistemul arata oscilatii amortizate,
% pana la atingerea regimului stationar.
% Se spune ca sistemul are raspuns oscilatoriu amortizat.
% Acest lucru va justifica folosirea unui control in bucla inchisa,
% si a unei compensari dinamice.
%
pause    % apasati orice tasta pentru a continua.
%
% De multe ori in descrierea sistemelor dinamice, ne punem problema
% stabilitatii sistemelor.
% Practic ne intereseaza daca sistemul nu va oscila singur din cauza unor
% vibratii sau oscilatii interne.
% fara a intra in detalii aici, sa retinem ca o informatie simpla despre
% stabilitatea unui sistem
% poate fi data de raspunsul la impuls.
% In MATLAB, putem determina raspunsul la impuls prin instructiunea impuls.
impulse(u*sys);
pause    % apasati orice tasta pentru a continua.
%
%____________________________________________________
```

Execuţia programului produce următoarele figuri.




Se observă că avem un răspuns oscilatoriu amortizat.

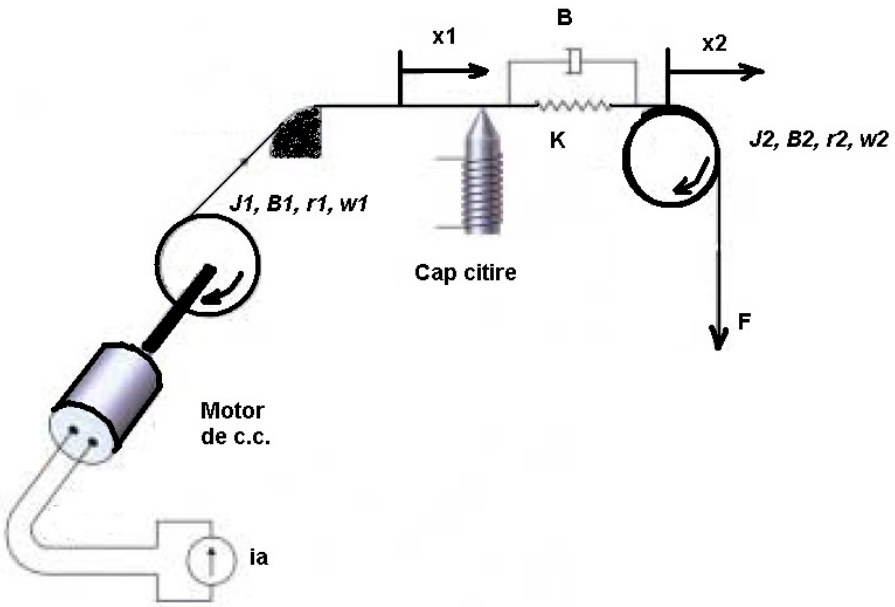
Lucru în clasă (rezultatele se includ în raportul de laborator)

Putem salva fişierul cu un nume nou şi modifica parametrii pentru a observa cum se schimbă funcţia de transfer şi răspunsul sistemului la semnalele treaptă şi rampă.

(g) micşoraţi constantele arcurilor (ks, kw) de 10 ori (F=kX, deci vom avea o deplasare mai mare la aceeaşi forţă aplicată)
(h) reveniţi la valorile originale pentru ks, kw şi modificaţi greutatea vehiculului, crescând de 2 ori masa, la m=3160kg.

## 5. Mini-Proiect.

Să considerăm cazul unui sistem de stocare de date pe banda magnetica („*tape driver*”), sistem utilizat într-un calculator (*entry-level servers sau workstations*). Acelaşi sistem poate descrie o casetă audio, cu banda magnetica. O reprezentare schematică a sistemului este dată în figura următoare.




Un motor de c.c. va furniza cuplu unui arbore capabil să mişte banda magnetică. Aplicarea unui curent pozitiv va determina rotaţia roţii ***r1*** în sensul acelor de ceasornic. Banda magnetică este întinsă de un dispozitiv caracterizat de constanta de tip arc ***K*** şi de amortizarea de tip frecare ***B***. Întinderea benzii este compensată de o forţă constantă ***F***.

Să scriem ecuaţiile de mişcare în funcţie de parametrii ce descriu sistemul practic:

- Raza primului disc r1 = $2*10^{-2}$ m
- Momentul de inerţie la primul disc J1 = $5*10^{-5}$ kg $m^2$
- Frecarea la primul disc B1 = $1*10^{-2}$ N m sec

- Constanta cuplului Kt = $3*10^{-2}$ N m /A

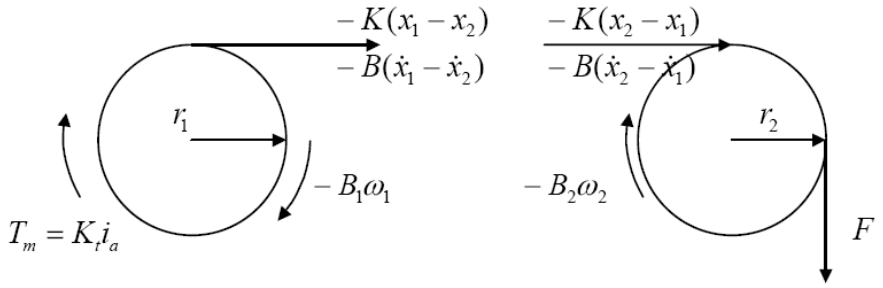
- Constanta de deformare a benzii (ca un arc de suspensie) K = $2 * 10^4$ N/m
- Dispozitiv de absorbţie a şocului aplicat benzii B = 20 N/m sec

- Raza celui de al doilea disc r2 = $2 * 10^{-2}$ m
- Momentul de inerţie la al doilea disc J2 = $2*10^{-5}$ kg $m^2$
- Frecarea la al doilea disc B2 = $2*10^{-2}$ N m sec

- Forţa constantă F = 6 N

Indicaţii pentru scrierea ecuaţiilor ce caracterizează acest sistem:

- Mărimea de intrare este curentul aplicat, iar mărimea de ieşire este deplasarea x1.
- Descompuneţi sistemul în subsisteme a căror funcţionare poate fi foarte uşor înţeleasă (figura următoare).




Ecuaţiile diferenţiale de mişcare devin:

| | |
|---|---|
| *Cuplul _ la _ roata _ r1* | $J_1 \cdot \frac{d\omega_1}{dt} = T_m - B_1 \cdot \omega_1 - \left[ B \cdot \left( \dot{x}_1 - \dot{x}_2 \right) + K \cdot \left( x_1 - x_2 \right) \right] \cdot r_1$ |
| *Cuplul _ a _ roata _ r2* | $J_2 \cdot \frac{d\omega_1}{dt} = -B_2 \cdot \omega_2 - \left[ B \cdot \left( \dot{x}_2 - \dot{x}_1 \right) + K \cdot \left( x_2 - x_1 \right) \right] \cdot r_2 + F \cdot r_2$ |
| *Cuplul _ Motor* | $T_m = K_t \cdot i_a$ |
| *Viteza _ benzii _ la _ roata _ r1* | $v = \dot{x}_1 = \frac{2 \cdot \pi \cdot r_1}{T} = r_1 \cdot \omega_1$ |
| *Viteza _ benzii _ la _ roata _ r2* | $v = \dot{x}_2 = \frac{2 \cdot \pi \cdot r_2}{T} = r_2 \cdot \omega_2$ |

Aceste ecuatii pot fi re-scrise în funcţie de mărimile de intrare (curentul $i_a$) şi ieşire ($x_1$), prin înlocuirea derivatei vitezei benzii în ecuaţiile pentru cuplu.

$$J_1 \cdot \frac{1}{r_1} \frac{d^2 x_1}{dt} = K_t \cdot i_a - B_1 \cdot \frac{1}{r_1} \dot{x}_1 - \left[ B \cdot \left( \dot{x}_1 - \dot{x}_2 \right) + K \cdot \left( x_1 - x_2 \right) \right] \cdot r_1$$

$$J_2 \cdot \frac{1}{r_2} \frac{d^2 x_2}{dt} = -B_2 \cdot \frac{1}{r_2} \dot{x}_2 - \left[ B \cdot \left( \dot{x}_2 - \dot{x}_1 \right) + K \cdot \left( x_2 - x_1 \right) \right] \cdot r_2 + F \cdot r_2$$

Mărimile de intrare care pot produce variaţii în sistem sunt curentul ***ia*** şi forţa ***F***. Determinarea unui model dinamic (de semnal mic) în jurul unui punct de echilibru, folosind acest set de ecuaţii, ar presupune înlocuirea formală a fiecărei mărimi cu termeni de genul $x=x_0+\delta x$. Deoarece forţa F se consideră constantă, aceasta nu va interveni în regimul dinamic ($\delta F=0$).

Pentru analiza de semnal mic în frecvenţă, vom cauta descrierea acestui sistem de ecuaţii cu forma Laplace, in funcţie de variabila „s". In acest sens, $i_a$ va reprezenta doar componenta dinamică a curentului. Cu mențiunea că transformata Laplace se poate aplica pentru valori inițiale nule, obţinem:

$$J_1 \cdot \frac{1}{r_1} \cdot s^2 \cdot x_1(s) = K_t \cdot i_a(s) - B_1 \cdot \frac{1}{r_1} \cdot s \cdot x_1(s) - \left[ \left( B \cdot s + K \right) \cdot \left( x_1(s) - x_2(s) \right) \right] \cdot r_1$$

$$J_2 \cdot \frac{1}{r_2} \cdot s^2 \cdot x_2(s) = -B_2 \cdot \frac{1}{r_2} \cdot s \cdot x_2(s) - \left[ \left( B \cdot s + K \right) \cdot \left( x_2(s) - x_1(s) \right) \right] \cdot r_2$$

Din care punem în evidenţă dependenţa lui ***x1*** de curentul ***ia***.

$$\left( J_1 \cdot \frac{1}{r_1} \cdot s^2 + B_1 \cdot \frac{1}{r_1} \cdot s + \left( B \cdot s + K \right) \cdot r_1 \right) \cdot x_1(s) = K_t \cdot i_a(s) + \left[ \left( B \cdot s + K \right) \cdot r_1 \right] \cdot x_2(s)$$

$$\left( J_2 \cdot \frac{1}{r_2} \cdot s^2 + B_2 \cdot \frac{1}{r_2} \cdot s + \left( B \cdot s + K \right) \cdot r_2 \right) \cdot x_2(s) = \left[ \left( B \cdot s + K \right) \cdot r_2 \right] \cdot x_1(s)$$

**=>**

$$\left(J_1\cdot\frac{1}{r_1}\cdot s^2+B_1\cdot\frac{1}{r_1}\cdot s+(B\cdot s+K)\cdot r_1\right)\cdot x_1(s)=K_t\cdot i_a(s)+\left[(B\cdot s+K)\cdot r_1\right]\cdot x_2(s)$$

$$x_2(s)=\frac{\left[(B\cdot s+K)\cdot r_2\right]\cdot x_1(s)}{\left(J_2\cdot\frac{1}{r_2}\cdot s^2+B_2\cdot\frac{1}{r_2}\cdot s+(B\cdot s+K)\cdot r_2\right)}$$

**=>**

$$\left(J_1\cdot\frac{1}{r_1}\cdot s^2+B_1\cdot\frac{1}{r_1}\cdot s+(B\cdot s+K)\cdot r_1-\frac{\left[(B\cdot s+K)\cdot r_1\right]\cdot\left[(B\cdot s+K)\cdot r_2\right]}{\left(J_2\cdot\frac{1}{r_2}\cdot s^2+B_2\cdot\frac{1}{r_2}\cdot s+(B\cdot s+K)\cdot r_2\right)}\right)\cdot x_1(s)=K_t\cdot i_a(s)$$

**=>**

$$\left(\frac{\left(J_1\cdot\frac{1}{r_1}\cdot s^2+B_1\cdot\frac{1}{r_1}\cdot s+(B\cdot s+K)\cdot r_1\right)\cdot\left(J_2\cdot\frac{1}{r_2}\cdot s^2+B_2\cdot\frac{1}{r_2}\cdot s+(B\cdot s+K)\cdot r_2\right)-\left[(B\cdot s+K)\cdot r_1\right]\cdot\left[(B\cdot s+K)\cdot r_2\right]}{\left(J_2\cdot\frac{1}{r_2}\cdot s^2+B_2\cdot\frac{1}{r_2}\cdot s+(B\cdot s+K)\cdot r_2\right)}\right)\cdot x_1(s)=K_t\cdot i_a(s)$$

**=>**

$$H(s)=\frac{x_1(s)}{i_a(s)}=\frac{K_t\cdot\left(J_2\cdot\frac{1}{r_2}\cdot s^2+B_2\cdot\frac{1}{r_2}\cdot s+(B\cdot s+K)\cdot r_2\right)}{\left(J_1\cdot\frac{1}{r_1}\cdot s^2+B_1\cdot\frac{1}{r_1}\cdot s+(B\cdot s+K)\cdot r_1\right)\cdot\left(J_2\cdot\frac{1}{r_2}\cdot s^2+B_2\cdot\frac{1}{r_2}\cdot s+(B\cdot s+K)\cdot r_2\right)-\left[(B\cdot s+K)\cdot r_1\right]\cdot\left[(B\cdot s+K)\cdot r_2\right]}$$

Lucru în clasă (rezultatele se includ în raportul de laborator):

(i) Simplificaţi (sau re-aranjaţi) această expresie cu termeni polinomiali pentru a facilita scrierea programului MATLAB corespunzător funcţiei de transfer **H(s).** Înlocuiţi valorile numerice în program.
(j) Determinaţi grafic răspunsul la semnal treaptă si la semnal impuls.
(k) Comentaţi rezultatele.

## Anexă : Tabel de funcţii/instrucţiuni MATLAB.

```
>> help elmat
Elementary matrices and matrix manipulation.

Elementary matrices.
zeros       - Zeros matrix.
ones        - Ones matrix.
eye         - Identity matrix.
rand        - Uniformly distributed random numbers.
randn       - Normally distributed random numbers.
linspace    - Linearly spaced vector.
logspace    - Logarithmically spaced vector.
meshgrid    - X and Y arrays for 3-D plots.
:           - Regularly spaced vector.

Special variables and constants.
ans         - Most recent answer.
eps         - Floating point relative accuracy.
realmax     - Largest floating point number.
realmin     - Smallest positive floating point
                number.
pi          - 3.1415926535897....
i, j        - Imaginary unit.
inf         - Infinity.
NaN         - Not-a-Number.
flops       - Count of floating point operations.
nargin      - Number of function input arguments.
nargout     - Number of function output arguments.
computer    - Computer type.
isieee      - True for computers with IEEE
                arithmetic.
isstudent   - True for the Student Edition.
why         - Succinct answer.
version     - MATLAB version number.

Time and dates.
clock       - Wall clock.
cputime     - Elapsed CPU time.
date        - Calendar.
etime       - Elapsed time function.
tic, toc    - Stopwatch timer functions.

Matrix manipulation.
diag        - Create or extract diagonals.
fliplr      - Flip matrix in the left/right
                direction.
flipud      - Flip matrix in the up/down direction.
reshape     - Change size.
rot90       - Rotate matrix 90 degrees.
tril        - Extract lower triangular part.
triu        - Extract upper triangular part.
:           - Index into matrix, rearrange matrix.


>> help specmat
Specialized matrices.

compan      - Companion matrix.
gallery     - Several small test matrices.
hadamard    - Hadamard matrix.
hankel      - Hankel matrix.
hilb        - Hilbert matrix.
invhilb     - Inverse Hilbert matrix.
kron        - Kronecker tensor product.
magic       - Magic square.
pascal      - Pascal matrix.
rosser      - Classic symmetric eigenvalue test
                problem.
toeplitz    - Toeplitz matrix.
vander      - Vandermonde matrix.
wilkinson   - Wilkinson's eigenvalue test matrix.


>> help elfun
Elementary math functions.

Trigonometric.
sin         - Sine.
sinh        - Hyperbolic sine.
asin        - Inverse sine.
asinh       - Inverse hyperbolic sine.
cos         - Cosine.
cosh        - Hyperbolic cosine.
acos        - Inverse cosine.
acosh       - Inverse hyperbolic cosine.
tan         - Tangent.
tanh        - Hyperbolic tangent.
atan        - Inverse tangent.
atan2       - Four quadrant inverse tangent.
atanh       - Inverse hyperbolic tangent.
sec         - Secant.
sech        - Hyperbolic secant.
asec        - Inverse secant.
asech       - Inverse hyperbolic secant.
csc         - Cosecant.
csch        - Hyperbolic cosecant.
acsc        - Inverse cosecant.
acsch       - Inverse hyperbolic cosecant.
cot         - Cotangent.
coth        - Hyperbolic cotangent.
acot        - Inverse cotangent.
acoth       - Inverse hyperbolic cotangent.

Exponential.
exp         - Exponential.
log         - Natural logarithm.
log10       - Common logarithm.
sqrt        - Square root.

Complex.
abs         - Absolute value.
angle       - Phase angle.
conj        - Complex conjugate.
imag        - Complex imaginary part.
real        - Complex real part.

Numeric.
fix         - Round towards zero.
floor       - Round towards minus infinity.
ceil        - Round towards plus infinity.
round       - Round towards nearest integer.
rem         - Remainder after division.
sign        - Signum function.

>> help specfun
Specialized math functions.

besselj     - Bessel function of the first kind.
bessely     - Bessel function of the second kind.
besseli     - Modified Bessel function of the first
                kind.
besselk     - Modified Bessel function of the second
                kind.
beta        - Beta function.
betainc     - Incomplete beta function.
betaln      - Logarithm of beta function.
ellipj      - Jacobi elliptic functions.
ellipke     - Complete elliptic integral.
erf         - Error function.
erfc        - Complementary error function.
erfcx       - Scaled complementary error function.
erfinv      - Inverse error function.
expint      - Exponential integral function.
gamma       - Gamma function.
gcd         - Greatest common divisor.
gammainc    - Incomplete gamma function.
lcm         - Least common multiple.
legendre    - Associated Legendre function.
gammaln     - Logarithm of gamma function.
log2        - Dissect floating point numbers.
pow2        - Scale floating point numbers.
rat         - Rational approximation.
rats        - Rational output.
cart2sph    - Transform from Cartesian to spherical
                coordinates.
cart2pol    - Transform from Cartesian to polar
                coordinates.
pol2cart    - Transform from polar to Cartesian
                coordinates.
sph2cart    - Transform from spherical to Cartesian
                coordinates.


>> help matfun
Matrix functions - numerical linear algebra.
```

```
Matrix analysis.
cond         - Matrix condition number.
norm         - Matrix or vector norm.
rcond        - LINPACK reciprocal condition estimator.
rank         - Number of linearly independent rows or
                 columns.
det          - Determinant.
trace        - Sum of diagonal elements.
null         - Null space.
orth         - Orthogonalization.
rref         - Reduced row echelon form.

Linear equations.
\ and /      - Linear equation solution; use "help
                 slash".
chol         - Cholesky factorization.
lu           - Factors from Gaussian elimination.
inv          - Matrix inverse.
qr           - Orthogonal-triangular decomposition.
qrdelete     - Delete a column from the QR
                 factorization.
qrinsert     - Insert a column in the QR
                 factorization.
nnls         - Non-negative least-squares.
pinv         - Pseudoinverse.
lscov        - Least squares in the presence of known
                 covariance.

Eigenvalues and singular values.
eig          - Eigenvalues and eigenvectors.
poly         - Characteristic polynomial.
polyeig      - Polynomial eigenvalue problem.
hess         - Hessenberg form.
qz           - Generalized eigenvalues.
rsf2csf      - Real block diagonal form to complex
                 diagonal form.
cdf2rdf      - Complex diagonal form to real block
                 diagonal form.
schur        - Schur decomposition.
balance      - Diagonal scaling to improve eigenvalue
                 accuracy.
svd          - Singular value decomposition.

Matrix functions.
expm         - Matrix exponential.
expm1        - M-file implementation of expm.
expm2        - Matrix exponential via Taylor series.
expm3        - Matrix exponential via eigenvalues and
                 eigenvectors.
logm         - Matrix logarithm.
sqrtm        - Matrix square root.
funm         - Evaluate general matrix function.


>> help general
General purpose commands.
MATLAB Toolbox  Version 4.2a 25-Jul-94

Managing commands and functions.
help         - On-line documentation.
doc          - Load hypertext documentation.
what         - Directory listing of M-, MAT- and MEX-
                 files.
type         - List M-file.
lookfor      - Keyword search through the HELP
                 entries.
which        - Locate functions and files.
demo         - Run demos.
path         - Control MATLAB's search path.

Managing variables and the workspace.
who          - List current variables.
whos         - List current variables, long form.
load         - Retrieve variables from disk.
save         - Save workspace variables to disk.
clear        - Clear variables and functions from
                 memory.
pack         - Consolidate workspace memory.
size         - Size of matrix.
length       - Length of vector.
disp         - Display matrix or text.

Working with files and the operating system.
cd           - Change current working directory.
dir          - Directory listing.
delete       - Delete file.
getenv       - Get environment value.
!            - Execute operating system command.
unix         - Execute operating system command &
                 return result.
diary        - Save text of MATLAB session.

Controlling the command window.
cedit        - Set command line edit/recall facility
                 parameters.
clc          - Clear command window.
home         - Send cursor home.
format       - Set output format.
echo         - Echo commands inside script files.
more         - Control paged output in command window.

Starting and quitting from MATLAB.
quit         - Terminate MATLAB.
startup      - M-file executed when MATLAB is invoked.
matlabrc     - Master startup M-file.

General information.
info         - Information about MATLAB and The
                 MathWorks, Inc.
subscribe    - Become subscribing user of MATLAB.
hostid       - MATLAB server host identification
                 number.
whatsnew     - Information about new features not yet
                 documented.
 ver          - MATLAB, SIMULINK, and TOOLBOX version
                 information.

>> help funfun
Function functions - nonlinear numerical methods.
ode23        - Solve differential equations, low order
                 method.
ode23p       - Solve and plot solutions.
ode45        - Solve differential equations, high
                 order method.
quad         - Numerically evaluate integral, low
                 order method.
quad8        - Numerically evaluate integral, high
                 order method.
fmin         - Minimize function of one variable.
fmins        - Minimize function of several variables.
fzero        - Find zero of function of one variable.
fplot        - Plot function.

See also The Optimization Toolbox, which has a
                 comprehensive
set of function functions for optimizing and
                 minimizing functions.

>> help polyfun
Polynomial and interpolation functions.

Polynomials.
roots        - Find polynomial roots.
poly         - Construct polynomial with specified
                 roots.
polyval      - Evaluate polynomial.
polyvalm     - Evaluate polynomial with matrix
                 argument.
residue      - Partial-fraction expansion (residues).
polyfit      - Fit polynomial to data.
polyder      - Differentiate polynomial.
conv         - Multiply polynomials.
deconv       - Divide polynomials.

Data interpolation.
interp1      - 1-D interpolation (1-D table lookup).
interp2      - 2-D interpolation (2-D table lookup).
interpft     - 1-D interpolation using FFT method.
griddata     - Data gridding.

Spline interpolation.
spline       - Cubic spline data interpolation.
ppval        - Evaluate piecewise polynomial.


>> help ops
Operators and special characters.
Char    Name                           HELP topic
   +      Plus                           arith
```

```
   -      Minus                        arith
   *      Matrix multiplication        arith
   .*     Array multiplication         arith
   ^      Matrix power                 arith
   .^     Array power                  arith

   \      Backslash or left division   slash
   /      Slash or right division      slash
   ./     Array division               slash
   kron   Kronecker tensor product     kron

   :      Colon                        colon

   ( )    Parentheses                  paren
   [ ]    Brackets                     paren

   .      Decimal point                punct
   ..     Parent directory             punct
   ...    Continuation                 punct
   ,      Comma                        punct
   ;      Semicolon                    punct
   %      Comment                      punct
   !      Exclamation point            punct
   '      Transpose and quote          punct
   =      Assignment                   punct

   ==     Equality                     relop
   <,>    Relational operators         relop
   &      Logical AND                  relop
   |      Logical OR                   relop
   ~      Logical NOT                  relop
   xor    Logical EXCLUSIVE OR         xor
```

**Logical characteristics.**

```
exist       - Check if variables or functions are
                defined.
any         - True if any element of vector is true.
all         - True if all elements of vector are
                true.
find        - Find indices of non-zero elements.
isnan       - True for Not-A-Number.
isinf       - True for infinite elements.
finite      - True for finite elements.
isempty     - True for empty matrix.
isreal      - True for real matrix.
issparse    - True for sparse matrix.
isstr       - True for text string.
isglobal    - True for global variables.
```

**>> help lang**
**Language constructs and debugging.**

**MATLAB as a programming language.**

```
script      - About MATLAB scripts and M-files.
function    - Add new function.
eval        - Execute string with MATLAB expression.
feval       - Execute function specified by string.
global      - Define global variable.
nargchk     - Validate number of input arguments.
lasterr     - Last error message.
```

**Control flow.**

```
if          - Conditionally execute statements.
else        - Used with IF.
elseif      - Used with IF.
end         - Terminate the scope of FOR, WHILE and
                IF statements.
for         - Repeat statements a specific number of
                times.
while       - Repeat statements an indefinite number
                of times.
break       - Terminate execution of loop.
return      - Return to invoking function.
error       - Display message and abort function.
```

**Interactive input.**

```
input       - Prompt for user input.
keyboard    - Invoke keyboard as if it were a Script-
                file.
menu        - Generate menu of choices for user
                input.
pause       - Wait for user response.
uimenu      - Create user interface menu.
uicontrol   - Create user interface control.
```

**Debugging commands.**

```
dbstop      - Set breakpoint.
dbclear     - Remove breakpoint.
dbcont      - Resume execution.
dbdown      - Change local workspace context.
dbstack     - List who called whom.
dbstatus    - List all breakpoints.
dbstep      - Execute one or more lines.
dbtype      - List M-file with line numbers.
dbup        - Change local workspace context.
dbquit      - Quit debug mode.
mexdebug    - Debug MEX-files.
```

**>> help plotxy**
**Two dimensional graphics.**

**Elementary X-Y graphs.**

```
plot        - Linear plot.
loglog      - Log-log scale plot.
semilogx    - Semi-log scale plot.
semilogy    - Semi-log scale plot.
fill        - Draw filled 2-D polygons.
```

**Specialized X-Y graphs.**

```
polar       - Polar coordinate plot.
bar         - Bar graph.
stem        - Discrete sequence or "stem" plot.
stairs      - Stairstep plot.
errorbar    - Error bar plot.
hist        - Histogram plot.
rose        - Angle histogram plot.
compass     - Compass plot.
feather     - Feather plot.
fplot       - Plot function.
comet       - Comet-like trajectory.
```

**Graph annotation.**

```
title       - Graph title.
xlabel      - X-axis label.
ylabel      - Y-axis label.
text        - Text annotation.
gtext       - Mouse placement of text.
grid        - Grid lines.
```

See also PLOTXYZ, GRAPHICS.

**>> help plotxyz**
**Three dimensional graphics.**

**Line and area fill commands.**

```
plot3       - Plot lines and points in 3-D space.
fill3       - Draw filled 3-D polygons in 3-D space.
comet3      - 3-D comet-like trajectories.
```

**Contour and other 2-D plots of 3-D data.**

```
contour     - Contour plot.
contour3    - 3-D contour plot.
clabel      - Contour plot elevation labels.
contourc    - Contour plot computation (used by
                contour).
pcolor      - Pseudocolor (checkerboard) plot.
quiver      - Quiver plot.
```

**Surface and mesh plots.**

```
mesh        - 3-D mesh surface.
meshc       - Combination mesh/contour plot.
meshz       - 3-D Mesh with zero plane.
surf        - 3-D shaded surface.
surfc       - Combination surf/contour plot.
surfl       - 3-D shaded surface with lighting.
waterfall   - Waterfall plot.
```

**Volume visualization.**

```
slice       - Volumetric visualization plots.
```

**Graph appearance.**

```
view        - 3-D graph viewpoint specification.
viewmtx     - View transformation matrices.
hidden      - Mesh hidden line removal mode.
shading     - Color shading mode.
axis        - Axis scaling and appearance.
caxis       - Pseudocolor axis scaling.
colormap    - Color look-up table.
```

```
Graph annotation.
title      - Graph title.
xlabel     - X-axis label.
ylabel     - Y-axis label.
zlabel     - Z-axis label for 3-D plots.
text       - Text annotation.
gtext      - Mouse placement of text.
grid       - Grid lines.

3-D objects.
cylinder   - Generate cylinder.
sphere     - Generate sphere.

See also COLOR, PLOTXY, GRAPHICS.


>> help strfun
Character string functions.

General.
strings     - About character strings in MATLAB.
abs         - Convert string to numeric values.
setstr      - Convert numeric values to string.
isstr       - True for string.
blanks      - String of blanks.
deblank     - Remove trailing blanks.
str2mat     - Form text matrix from individual
              strings.
eval        - Execute string with MATLAB expression.

String comparison.
strcmp      - Compare strings.
findstr     - Find one string within another.
upper       - Convert string to uppercase.
lower       - Convert string to lowercase.
isletter    - True for letters of the alphabet.
isspace     - True for white space characters.
strrep      - Replace a string with another.
strtok      - Find a token in a string.

String to number conversion.
num2str     - Convert number to string.
int2str     - Convert integer to string.
str2num     - Convert string to number.
mat2str     - Convert matrix to string.
sprintf     - Convert number to string under format
              control.
sscanf      - Convert string to number under format
              control.

Hexadecimal to number conversion.
hex2num     - Convert hex string to IEEE floating
              point number.
hex2dec     - Convert hex string to decimal integer.
dec2hex     - Convert decimal integer to hex string.
```