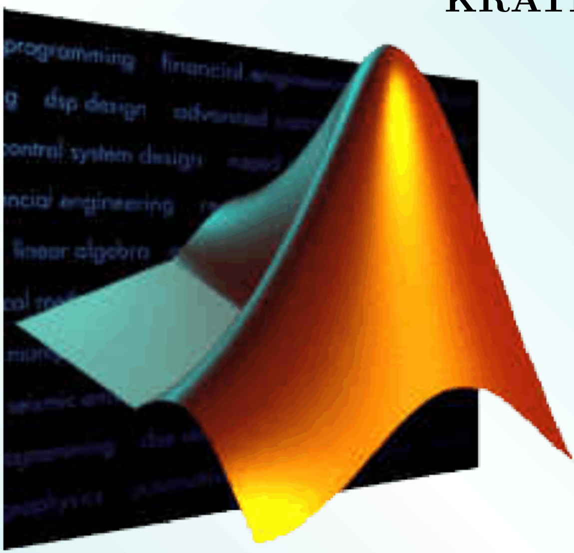
Dejan Zupan

Programski jezik MATLAB

KRATEK TEČAJ



Katedra za mehaniko
Fakulteta za gradbeništvo in geodezijo
Univerza v Ljubljani

[10.10.2007]

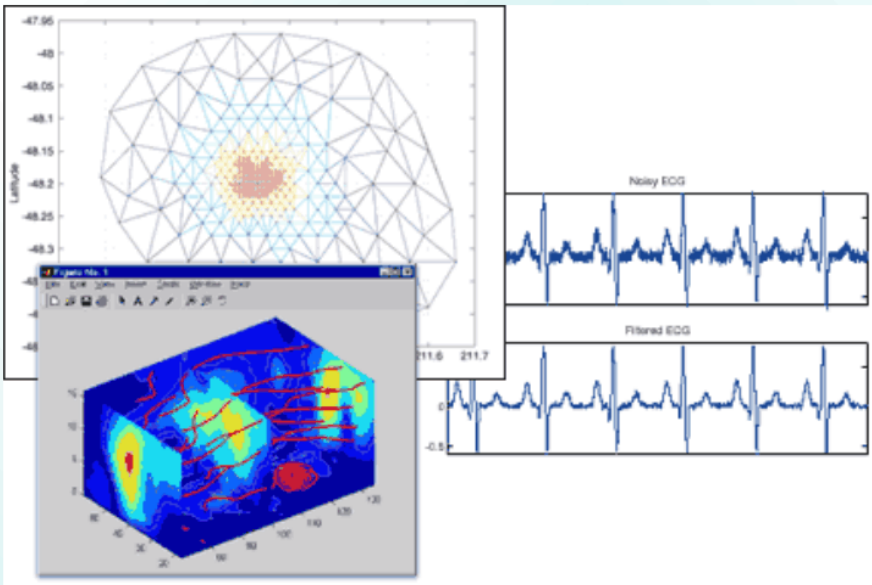
Spletna stran
Naslovnica
Kazalo
Stran 1 od 114
Nazaj
Poln zaslon
Zapri
Končaj

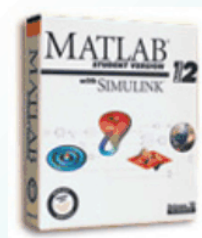
Zakaj Matlab?

- RAZŠIRJENOST IN DOSEGLJIVOST
- PRILAGOJEN JE ZA MATRIČNI RAČUN
  - osnovni element je matrika
  - skalarji in vektorji so poseben primer matrik
  - osnovna verzija zna računati le numerično
- VELIKO ŠTEVILO VGRAJENIH FUNKCIJ ZA NUMERIČNO RAČUNANJE
- ZMOGLJIVA GRAFIKA
- PROGRAMSKI JEZIK
  - preprost za uporabnika (ni potrebno: rezervacije spomina, spremenljivk....)
  - pregledno in hitro programiranje

- KVALITETNA PODPORA
  - podjetje MathWorks: http://www.mathworks.com
  - Katedra za mehaniko: http://www.km.fgg.uni-lj.si
- ŠE NEKAJ UPORABNIH NASLOVOV
  - Matlab na MathWorks: MATLAB-začetna stran
  - Kratek tečaj Matlaba: University of New Hampshire: MATLAB-tutorial
  - Osnove Matlaba: University of Bergen: MATLAB-primer
  - Priročnik Matlaba: University of Florida: MATLAB Tutorial
  - Informacijska podpora programom: Massachusetts Institute of Technology: MATLAB on Athena

- Matlabove izobraževalne strani: MATLAB Educational Sites
- Uporaba Matlaba za analizo in načrtovanje avtomatskih kontrolnih sistemov: University of Michigan: Tutorial for MATLAB
- Orodja za Matlab: MATHTools/MATLAB





Spletna stran
Naslovnica
Kazalo
Stran 4 od 114
Nazaj
Poln zaslon
Zapri
Končaj

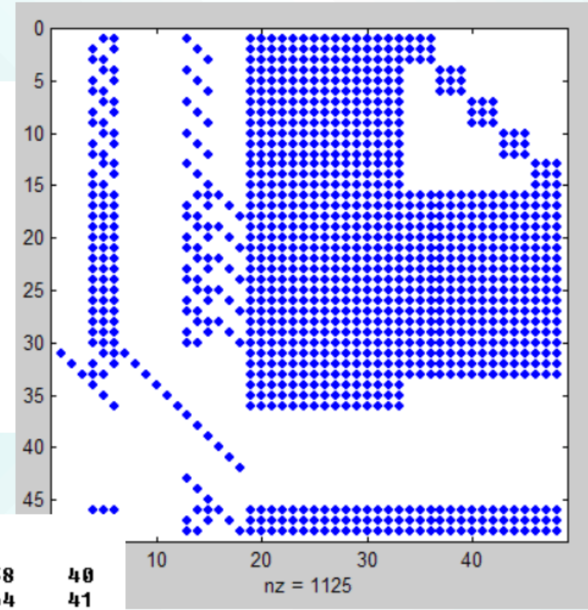
Pomoč

- Najhitrejša in najpomembnejša pomoč je ukaz help
  - oblika help ime_funkcije;
  - primera help help, help sin.
- Za iskanje neznanih funkcij je uporabnejši ukaz lookfor
  - oblika lookfor niz;
  - poišče vse MATLABove funkcije, ki imajo niz v opisu pomoči;
  - primer lookfor earth.
- Za uporabo funkcij, ki niso vgrajene v MATLAB, moramo najprej definirati pot do področja, kjer se funkcija nahaja:
  - uporabimo opcijo iz menuja File/Set Path...;
  - običajno tudi za take funkcije deluje pomoč help ime_funkcije.

I. Del

MATRIKE IN VEKTORJI

```
>> a=1:10
a =
        1         2         3         4         5         6         7         8         9
>> a'
ans =
        1
        2
        3
        4
        5
        6
        7
        8
        9
       10
```




```
>> A=magic(10)
A =
    92    99     1     8    15    67    74    51    58    40
    98    80     7    14    16    73    55    57    64    41
     4    81    88    20    22    54    56    63    70    47
    85    87    19    21     3    60    62    69    71    28
    86    93    25     2     9    61    68    75    52    34
    17    24    76    83    90    42    49    26    33    65
    23     5    82    89    91    48    30    32    39    66
    79     6    13    95    97    29    31    38    45    72
    10    12    94    96    78    35    37    44    46    53
    11    18   100    77    84    36    43    50    27    59
>>
```

Spremenljivke v Matlabu

1.– 4.

- OSNOVNA KOLIČINA JE MATRIKA
  - primer: A = [1 2 3 4; 4 5 6 7; 7 8 9 10]
  - vektor (stolpec) je poseben primer matrike v=[1; 2; 3]
  - skalar je matrika z enim elementom s=5 (pomeni isto kot s=[5])
  - Matlab pozna tudi prazno matriko B=[ ]

1.– 3.

- OSNOVNE OPERACIJE
  - ' transponiranje (A',v')
  - + in – sta operaciji, ki delujeta po komponentah (potrebujemo matriki enakih dimenzij)
  - * in / sta matrično množenje in deljenje
  - .* in ./ pomenita množenje in deljenje po komponentah
  - \ je levo deljenje matrik (sistem enačb Ax=b reši ukaz x=A\b)

Manipulacija z matrikami

- DOSTOPANJE DO KOMPONET IN PODMATRIK
  - prek indeksa (mesta) elementa: A(2,3), v(1)
  - prek nabora(ov) (vektorja) indeksov A([1 3],[1 2]), v([2,3])
  - krajše indeksiranje z uporabo :, ki pomeni 1:4=[1 2 3 4]
    npr.: A(1:3,1:3) je podmatrika 3x3 zgoraj desno, A(2,:) je druga vrstica matrike A
  - : se uporablja za indeksiranje tudi splošneje
    začetek : inkrement : konec
- ELEMENTARNE FUNKCIJE NA MATRIKAH
  - vgrajene funkcije sin, cos, tan, exp... delujejo na komponentah matrike
  - potence matrik (Aˆ3) pomenijo matrično množenje (A*A*A)
  - potence komponent dobimo z operatorjem .ˆ

5.– 10.

Vgrajene matrične funkcije

10.– 16.

FUNKCIJA	POMEN	PRIMER
size()	velikost matrike	size$\left(\begin{bmatrix} 3 & 2 & 0 \\ 4 & 0 & 6 \\ 9 & 8 & 7 \end{bmatrix}\right)$=[3,3], size([1;2;3])=[3,1]
rand()	naključna matrika	rand(7), rand(2,3), round(10*rand(5))
eye()	enotska matrika	eye(3)=$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
zeros()	matrika ničel	zeros(3)=$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$, zeros(2,3)=$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
ones()	matrika enic	
diag()	diagonala matrike diagonalna matrika	diag$\left(\begin{bmatrix} 3 & 2 & 1 \\ 4 & 5 & 6 \\ 9 & 8 & 7 \end{bmatrix}\right)$=$\begin{bmatrix} 3 \\ 5 \\ 7 \end{bmatrix}$, diag([1,2,3])=$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$
det()	determinanta matrike	
rank()	rang	
inv()	matrični inverz	
norm()	matrična/vektorska norma	
eig()	lastni prostor matrike	eig(A) . . . stolpec lastnih vrednosti [V,D]=eig(A) . . . lastni vektorji in vrednosti
cross()	vektorski produkt	cross([1 2 3],[2 1 1])

Vektorske funkcije

17.– 18.

- Matlab posebej odlikuje vektorje, to so matrike z enim samim stolpcem ali eno samo vrstico.
- Vektorske funkcije
  - delujejo na vektorjih (vrsticah ali stolpcih);
  - običajno preslikajo vektorje v realna števila;
  - uporabimo jih lahko tudi na matrikah, takrat delujejo za vsak stopec matrike posebej (rezultat je vrstica rezultatov vektorske funkcije za vsak stolpec posebej.
- Nekaj najpomembnejših vektorskih funkcij:

FUNKCIJA	POMEN
max(), min()	največja/najmanjša komponenta vektorja
sum(), prod()	vsota/produkt komponent vektorja
sort()	uredi vektor po naraščajočem vrstnem redu
any()	vrne 1, če je vsaj ena komponenta neničelna
all()	vrne 1, če so vse komponente neničelne

Ponovi, odgovori na vprašanja!

1. Kaj je vektor (stolpec/vrstica)?
2. Kaj je matrika?
3. Kako množimo matrike?
4. Kaj je linearni sistem enačb in kako ga rešimo?
5. Kaj je rang matrike (pomen)?
6. Kaj je determinata matrike (pomen)?
7. Matrični inverz.
8. Kaj so lastne vrednosti in kaj lastni vektorji?
9. Pomen matrike lastnih vektorjev.
10. Kaj če razcep na lastne podprostore ni možen?

Naloge (I.)

1. Definiraj matriki

$$\mathsf{F} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 7 & 9 \\ 1 & 4 & 8 \end{bmatrix} \quad \text{in} \quad \mathsf{G} = \begin{bmatrix} -1 & 2 & -3 \\ 1 & -2 & -5 \\ 1 & 4 & -9 \end{bmatrix}$$

   in izračunaj:

   (a) vsoto F in G;
   (b) produkt vsote F in G s F;
   (c) produkt iz točke (b) po komponentah;
   (d) kvadrat matrike G;
   (e) koren komponent matrike F – G;

2. Naj bo x = $[3\ 2\ 6\ 8]^T$ in y = $[4\ 1\ 3\ 5]^T$

   (a) seštej komponete x in y;
   (b) potenciraj komponente x z istoležnimi komponentami y;
   (c) deli vsako komponento y z istoležno komponento x;
   (d) množi vsako komponento x z istoležno komponeto y in rezultat zapiši v z;

(e) izračunaj $x^T y - z$ in interpretiraj rezultat.

3. Za matrike x = [1 4 8], y = [2 1 5] in A = [3 1 6 ; 5 2 7] ugotovi, kateri izrazi so izračunljivi (smiselni). Izračunljive izraze izvrednoti, za ostale pa utemelji, zakaj izraz ni dopusten!

   (a) x + y
   (b) x + A
   (c) x' + y
   (d) A – [x' y']
   (e) [x ; y']
   (f) [x ; y]
   (g) A – 3

4. Za dano matriko A = [2 7 9 7 ; 3 1 5 6 ; 8 1 2 5] napovej in preveri rezultate naslednjih ukazov!

   (a) A'
   (b) A(: , [1 4])
   (c) A([2 3] , [3 1])
   (d) A(:)
   (e) [A ; A(end,:)]

(f) A(1:3 , :)
(g) [A ; A(1:2 , :)]

5. Za matriko A = [2 7 9 7 ; 3 1 5 6 ; 8 1 2 5] napiši ukaze, s katerimi

   (a) prirediš lihe stolpce A matriki B;
   (b) prirediš sode vrstice matrike A matriki C;
   (c) izračunaš inverze vseh komponent matrike A.

6. Naj bo x = [3 1 5 7 9 2 6]! Napovej in preveri rezultate naslednjih ukazov!

   (a) x(3)
   (b) x(1:7)
   (c) x(1:end–1)
   (d) x(6:–2:1)
   (e) x([1 6 2 1 1])

7. Definiraj vektor x z elementi

   (a) 2, 4, 6, 8, ...;
   (b) 1, 1/2, 1/3, 1/4, 1/5, ...;
   (c) 0, 1/2, 2/3, 3/4, 4/5, ...

8. Definiraj vektor y, katerega komponente so določene s predpisom

$$y_n = -\frac{n+1}{2n-1}.$$

9. Definiraj vektor t=1:0.2:2. Za tako definiran vektor pravilno zapiši ukaz v Matlabu in izračunaj naslednje izraze!

   (a) $ln\left(2+t+t^2\right)$
   (b) $e^t\left(1+\cos\left(3t\right)\right)$
   (c) $\cos^2 t + \sin^2 t$
   (d) $\tan^{-1}\left(t\right)$ (inverz funkcije tangens)

10. V Matlab vnesi matrike

$$A_1 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2 & 1 \\ 1 & 3 & 1 & 1 & 1 \\ 2 & 5 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}, \quad A_3 = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix};$$

   (a) v matriki A_3 naj element v šesti vrstici in v petem stolpcu postane 7, element v osmi vrstici in četrtem stolpcu pa naj bo 8;
   (b) v matriki A_3 spremeni v bloku ničel v spodnjem levem kotu diagonalne elemente v enice, blok ničel v zgornjem desnem kotu pa nadomesti s števili 3;

(c) matriko A_3 prepiši v matriko B. Matriki B zbriši 4., 5. in 6. stolpec ter prve tri vrstice;

(d) izračunaj determinanto matrike B in njene lastne vrednosti.

11. Reši sistem enačb po naslednjih korakih:

   (a) konstruiraj naključno matriko A;

   (b) konstruiraj naključen vektor b;

   (c) reši sistem enačb A x = b;

   (d) preveri rezutat.

12. Za dane vektorje e1 = [1 1 -1 1], e2 = [1 -1 -1 1], e3 = [1 1 1 1] in e4 = [-1 1 1 1] dokaži, da tvorijo bazo štirirazsežnega prostora (so linearno neodvisni).

13. Za n = 50 in n = 100 vpiši matriko

$$\begin{bmatrix} -n & 0 & 0 & \cdots & 0 & 1 \\ 0 & -n & 0 & \cdots & 0 & 1 \\ 0 & 0 & -n & \cdots & 0 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -n & 1 \\ 1 & 2 & 3 & \cdots & n-1 & 1 \end{bmatrix}$$

na preprost (hiter) način v Matlab in izračunaj determinato. Rešitvi preveri še z računoma: d50=0.5*51(-50)ˆ49 in d100=0.5*101(-100)ˆ99

14. Podobni matriki. Dani sta matriki

$$\mathsf{A} = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \quad \text{in} \quad \mathsf{B} = \begin{bmatrix} 4 & -4 \\ -1 & 1 \end{bmatrix}.$$

Preveri, da sta podobni! (Matriki sta podobni natanko tedaj, ko imata enake lastne vrednosti)

15. (Rotacijska matrika) Rotacija v prostoru je določena z enotskim vektorjem $\mathsf{n} = \begin{bmatrix} n_1 \\ n_2 \\ n_3 \end{bmatrix}$ in s kotom zasuka ϑ. Če definiramo matriko

$$\mathsf{N} = \begin{bmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{bmatrix},$$

je s formulo

$$\mathsf{R} = \mathsf{I} + \sin\vartheta \mathsf{N} + (1 - \cos\vartheta)\,\mathsf{N}^2$$

določena rotacijska matrika. Za vektor rotacije $\mathsf{n} = \frac{1}{\sqrt{3}}[1\ -1\ \ 1]^T$ in kot rotacije $\vartheta = \pi/6$ izračunaj:

(a) pripadajočo rotacijsko matriko in jo shrani kot R;

(b) transponirano matriko R' in inverzno matriko k R – kaj lahko sklepaš;

(c) lastne vrednosti in lastne vektorje matrike R;

(d) lastne vrednosti in lastne vektorje matrike N.

16. Reši matrično enačbo A*X + B = 0 za matriki

$$\mathsf{A} = \begin{bmatrix} 1 & 1 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathsf{B} = \begin{bmatrix} 1 & 2 & 1 \\ 3 & 0 & 0 \\ 0 & 0 & 4 \end{bmatrix}.$$

17. Izračunaj vsoto 100 členov zaporedja

$$a_n = \frac{3n-1}{2n+3},$$

kjer je n naravno število.

18. Tvori vrstico b vsot matrike B = $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 8 \end{bmatrix}$ po vrsticah in reši linearni sistem enačb Bx = b^T.

Težje naloge

1. Poišči komponente stolpca a = [2 -3 1 5]^T v bazi štirirazsežnega prostora: e1 = [1 1 -1 1]^T, e2 = [1 -1 -1 1]^T, e3 = [1 1 1 1]^T in e4 = [-1 1 1 1]^T.
2. Izračunaj rotacijsko matriko, ki nadomesti tri rotacije za Eulerjeve kote $\pi/3$, $\pi/6$ in $\pi/4$ (vrtenja okrog osi [0 ; 0 ; 1], R$_1$[0 ; 1 ; 0] in R$_2$R$_1$[0 ; 0; 1]).

3. Za podobni matriki

$$\mathsf{A} = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \quad \text{in} \quad \mathsf{B} = \begin{bmatrix} 1 & -1 \\ -4 & 4 \end{bmatrix}.$$

izračunaj matriko P, za katero velja A = PBP^{-1}!

4. Splošno reševanje sistema enačb. Reši sistem enačb

$$\begin{aligned} 3\,x_1 - 2\,x_2 - x_3 - x_4 &= 1 \\ 6\,x_1 - 4\,x_2 + 4\,x_3 + 3\,x_4 &= 3 \\ 3\,x_1 - 2\,x_2 + 5\,x_3 + 4\,x_4 &= 2 \\ 9\,x_1 - 6\,x_2 + 3\,x_3 + 2\,x_4 &= a \end{aligned}$$

za $a = 4$ in $a = 3$. Uporabi splošni algoritem, ki deluje za poljuben sistem enačb (tudi, če je enačb premalo). Shema algoritma (glej tudi R.Flajs: MTT):

i) vnesi nalogo v obliki matrike koeficientov A in stolpca desnih strani b;

ii) Izračunaj rešitev x0 po metodi najmajših kvadratov: x0 = pinv(A) * b;

iii) Preveri, če rešitev x0, dobljena po metodi najmajših kvadratov zadošča enačbi A x0 =b. Če ji zadošča, potem obstaja vsaj ena rešitev $\Longrightarrow$ pojdi na korak (iv). Če ji ne zadošča, rešitve ni $\Longrightarrow$ KONEC.

iv) Izračunaj jedro matrike A: Ker = null(A); Če je jedro prazna matrika $\Longrightarrow$ obstaja natanko ena rešitev x = x0; Če jedro ni prazna matrika, ga napenja k vektorjev x1, x2, ..., xk in obstaja neskončno mnogo rešitev $\Longrightarrow$ splošna rešitev je oblike (za poljubno izbrane vrednosti c1, c2, ..., ck):

$$\mathsf{x} = \mathsf{x0} + \mathsf{c1} * \mathsf{x1} + \mathsf{c2} * \mathsf{x2} + \cdots + \mathsf{ck} * \mathsf{xk}, \quad \text{KONEC.}$$

5. Reši sistem enačb

$$\begin{aligned}(1+a)\,x_1 + x_2 + x_3 &= 1\\ x_1 + (1+a)\,x_2 + x_3 &= a\\ x_1 + x_2 + (1+a)\,x_3 &= a^2\end{aligned}$$

po splošnem algoritmu za $a = -3, 0, 1$. Kaj pa dobiš z običajnim reševanjem (x = A\b)?

6. Reši sistem oblike A X - XB = C za primer

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \mathsf{X} - \mathsf{X} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Uporabi prevedbo na linearen sistem enačb po naslednjem splošnem algoritmu:

i) Matrika ekvivalentnega linearnega sistema enačb je

$$\mathsf{M} = \mathsf{kron}(\mathsf{eye}(2), \mathsf{A}) - \mathsf{kron}(\mathsf{B}', \mathsf{eye}(2)).$$

ii) Desne strani ekvivalentnega linearnega sistema enačb so

$$c = C(:).$$

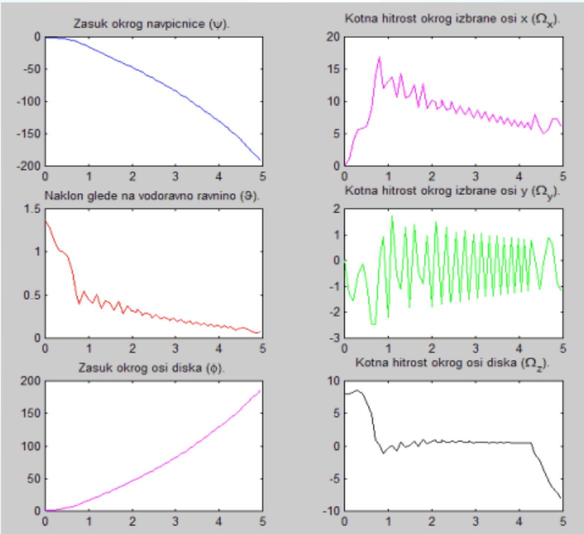
iii) Rešimo linearni sistem enačb x = M\c in rešitev preuredimo v matriko

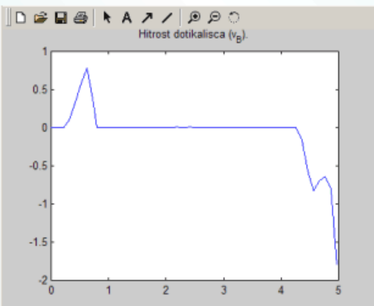
$$X = \mathrm{reshape}(x, 2, 2).$$

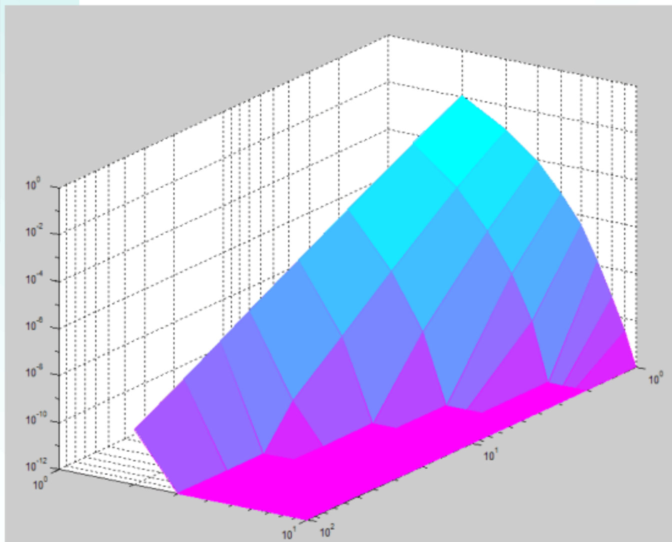
Preveri dobljene rezultate. Opomba. kron je Kroneckerjev tenzorski produkt; več o tem izveš z ukazom help kron.

II. Del

MATLABOVA GRAFIKA







Ravninski grafiki

Osnovni ukaz je plot

- plot(y) nariše odsekoma linearen graf elementov vektorja y glede na njihove indekse;
- plot(x,y) nariše odsekoma linearen graf elementov vektorja y v odvisnosti od x;
- plot(x,y,'parameter') riše graf z upoštevanjem dodatnih parametrov:

  'c', 'm', 'y', 'r', 'g', 'b', 'w', 'k' so barve;

  '-', '- -', ':', '-.' so tipi črt;

  '+', 'o', '*', 'x' so znamenja definiranih točk na grafu;
- rišemo lahko več grafov hkrati plot(x,y1,x,y2);
- y lahko zamenjamo z matriko, takrat riše skupino grafov – za vsak stolpec matrike svoj graf.

Oznake na grafu

FUNKCIJA	POMEN
title('naslov')	naslov grafa
xlabel('x')	ime osi x
ylabel('y')	ime osi y
legend('opis1','opis2',...)	pomen posameznih krivulj na grafu
text(xt,yt,'besedilo')	napiše besedilo na mesto (xt,yt)
grid on (grid off)	prikaz ali skrivanje pomožnih črt
axis on (axis off)	prikaz ali skrivanje koordinatnih osi
axis equal	enako velike enote na obeh oseh
axis square	kvadraten graf
axis([xmin xmax ymin ymax])	sami določimo meje grafa

Ostali grafični ukazi

- Kontrola grafičnega okna:
  - z risanjem slike se avtomatsko odpre grafično okno;
  - nova slika prekrije staro, razen če uporabimo ukaz hold on;
  - grafično okno lahko odpremo (aktiviramo) tudi sami: figure, figure(n).

7.– 10.

- Še nekaj funkcij, s katerimi rišemo grafe:

FUNKCIJA	POMEN
loglog	graf v logaritemskem merilu obeh osi
semilogx	graf z logaritemskim merilom na osi x
semilogy	graf z logaritemskim merilom na osi y
polar(koti,dolžine)	polarni graf
plot3(x,y,z)	graf krivulje v trirazsežnem prostoru

Trirazsežna grafika

- Ploskve v prostoru rišemo z ukazoma

  mesh(X,Y,Z) in surf(X,Y,Z).

- mesh riše ploskev s koordinatnimi črtami, surf pa z barvnimi ploskvicami.

- Vhodni podatki so trije : vrednosti koordinat X in Y ter pripadajoče točke na ploskvi Z

- Pripravo dvodimenzionalne mreže v ravnini (xy) si olajšamo z ukazom meshgrid!

- Primer:

```
[X,Y]=meshgrid(0:0.1:10,0:0.1:5);
Z=sin(X+Y);
mesh(X,Y,Z);
surf(X,Y,Z);
```

Možnosti trirazsežne grafike

11.– 12.

MOŽNOST		PRIMER
BARVNE SKALE colormap	hot cool copper pink	sphere axis equal colormap copper colormap cool
PRELIVANJA BARV 'FaceColor' 'EdgeColor'	'none' 'flat' 'interp'	set(mesh(X,Y,Z),'EdgeColor','none') set(mesh(X,Y,Z),'FaceColor','interp') set(surf(X,Y,Z),'FaceColor','flat') set(surf(X,Y,Z),'EdgeColor','interp') set(surf(X,Y,Z),'EdgeColor','g')
SENČENJE shading	flat interp faceted	set(surf(X,Y,Z),'EdgeColor','interp') shading interp shading flat shading faceted

Filmi

13.– 14.

- V novejših različicah Matlaba je ustvarjanje filmov zelo preprosto.
- Film je zaporednje slik, ki jih združimo v celoto.
- Običajno zaporedje slik generiramo znotraj zanke for, ustvarjene slike pa sestavljamo v film z ukazom getframe.
- Splošna oblika zapisovanja filma je

```
for j = 1:n %zanka po naravnih številih
    ukazi za risanje grafov;      %dobimo j-to sliko
    M(j) = getframe;      %shrani j-to sliko v M
end
```

- Pri izvajanju gornje zanke v grafičnem oknu nastaja film.
- Film M, ki smo ga tako ustvarili iz n slik, lahko ponovno prikažemo z ukazom movie(M).

Naloge (II.)

1. Nariši graf funkcije $y = \sin x$ za $x \in [0, 2\pi]$ s preizkušanjem naslednjih ukazov!
   (a) x = 0:2*pi; y = sin(x); plot(x,y)
   (b) izgled ni preveč navdušujoč, zato graf zgladimo x = 0:0.1: 2*pi; y = sin(x); plot(x,y)
   (c) še lepši graf dobimo s še manjšim korakom x = 0:0.01: 2*pi; y = sin(x); plot(x,y)
   (d) dodaj oznake osi xlabel('To je os x'), ylabel('To je os y')
   (e) dodaj naslov title('VAJE IZ RISANJA GRAFOV')
   (f) dodaj legendo legend('graf sin(x)')
2. Na graf iz prejšnje točke dodaj graf funkcije cosinus v rdeči barvi!
   (a) najprej zadrži staro sliko hold on
   (b) izračunaj točke in nariši graf z = cos(x); plot(x,z,'r')
   (c) razširi legendo z obema opisoma legend('graf sin(x)','graf cos(x)')
3. Nariši graf funkcije $f(x) = x^3$ na območju $[-4, 4]$;
   (a) izberi še majhno število točk za x (na primer –4:0.5:4) in označi točke, skozi katere poteka graf z zelenimi znaki '*';

(b) oba grafa prikaži na isti sliki.

4. Nariši zahtevnejši graf $f(x) = \mathrm{e}^{-0.4x} \sin x$, $x \in [0, 2\pi]$:

   (a) x = linspace(0,2*pi,100); y = exp(–0.4*x).*sin(x); plot(x,y);

   (b) označi osi, prikaži naslov in legendo;

   (c) z ukazom help linspace preveri, kaj je rezultat te funkcije.

   Pri tem izberi dovolj gosto mrežo za x, da bodo krivulje gladke!

5. Definiraj krivuljo v ravnini, podano v parametrični obliki kot $(\sin t, 2\cos t)$ za $t \in [0, 2\pi]$. Nariši graf te krivulje in preveri ukaze: axis normal, axis square, axis equal, axis equal tight. Namig: najprej definiraj vektor t, potem pa z ukazom plot(sin(t),2*cos(t)) nariši graf.

6. Nariši gladek (nezobat) graf funkcije $f(x) = \sin\frac{1}{x}$ za $0.01 < x < 0.1$. Kakšna izbira točk x je bila potrebna za lep graf? Odpri novo grafično okno (ukaz figure) in poskusi isti graf narisati še z ukazoma fplot('sin(1 ./ x)', [0.01 0.1]) in fplot('sin(1 ./ x)', [0.01 0.1],1e–3)! Oglej si razlago funkcije fplot: help fplot!

7. Nariši grafe funkcij x, x^3, e^x, e^{x^2} na območju $0 < x < 4$

   (a) v običajnem merilu z enakimi enotami na oseh;

   (b) v logaritemskem merilu za os y;

   (c) v logaritemskem merilu za obe osi.

8. Nariši polarni graf spirale $r = e^{0.2\varphi}$, $\varphi \in [0, 6\pi]$ z uporabo ukaza polar(fi,r)!

9. V polarnih koordinatah (r, φ) je

$$r(\varphi) = a\frac{1-\epsilon^2}{1-\epsilon\cos(\varphi)}$$

enačba elipse z enim goriščem v koordinatnem izhodišču, kjer je a dimenzija v smeri x osi in ϵ(<1) ekscentričnost elipse. Z uporabo gornje formule nariši nekaj elips. Pri tem z izbiro dovolj goste mreže za φ riši gladko krivuljo, z ukazom axis equal pa realistično razmerje osi (realistično sliko).

10. Nariši prostorsko krivuljo vijačnice $x = \sin t$, $y = \cos t$, $z = t$! Najprej pripravi vektorje t, x in y, potem pa uporabi ukaz plot3(x,y,z).

11. 'Klobuk' lahko definiramo kot ploskev $f(x, y) = \frac{\sin\left(\sqrt{x^2+y^2}\right)}{\sqrt{x^2+y^2}}$. Nariši to ploskev v Matlabu! Potek: Pripravi mrežo kot [X,Y]=meshgrid(–8:0.5:8);. Ker z nič ne smemo deliti, definiraj R kot R=sqrt(X.^2+Y.^2)+eps; in točke na ploskvi izračunaj kot Z=sin(R)./R; Z uporabo ukazov mesh in surf prikaži ploskev. Ploskev poskusi pobarvati po svojem okusu z dodatnimi možnostmi ('FaceColor', colormap, shading).

12. (Risanje parametrično podane ploskve) Enneperjevo ploskev lahko po-

damo parametrično kot

$$enn(u,v) = \{-\frac{u^3}{3} + uv^2, v - \frac{v^3}{3} + vu^2, u^2 + v^2\}.$$

Nariši Enneperjevo ploskev za $u, v \in [-5, 5]$. Najprej pripravi mrežo [u,v]=meshgrid(...); potem pa iz pripravljenih matrik u in v pravilno izračunaj matrike x, y in z ob upoštevanju formul

$$\begin{aligned} x &= -\frac{u^3}{3} + uv^2 \\ y &= v - \frac{v^3}{3} + vu^2 \\ z &= u^2 + v^2. \end{aligned}$$

Zaradi matričnega zapisa pazi na operacije po komponentah. Pripravljene matrike x, y in z predstavi z ukazom surf. Z uporabo gumba za rotacijo v grafičnem oknu si oglej samopresečišča Enneperjeve ploskve.

13. Zapiši naslednjo preprosto funkcijo ustvarjanja filma

```
function M=film
x=0:0.1:2*pi;
for i=1:10;
   plot(x,sin(i*x));
   axis([0,2*pi,–1,2])
   M(i)=getframe;
```

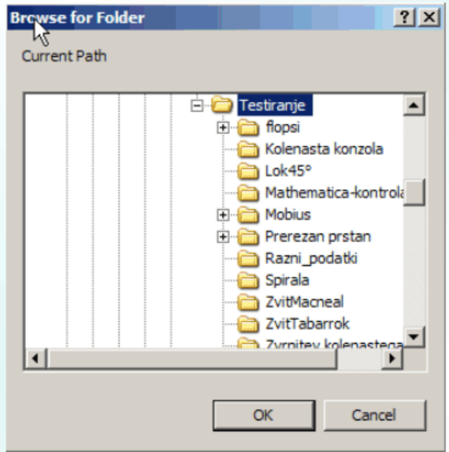
```
end
```

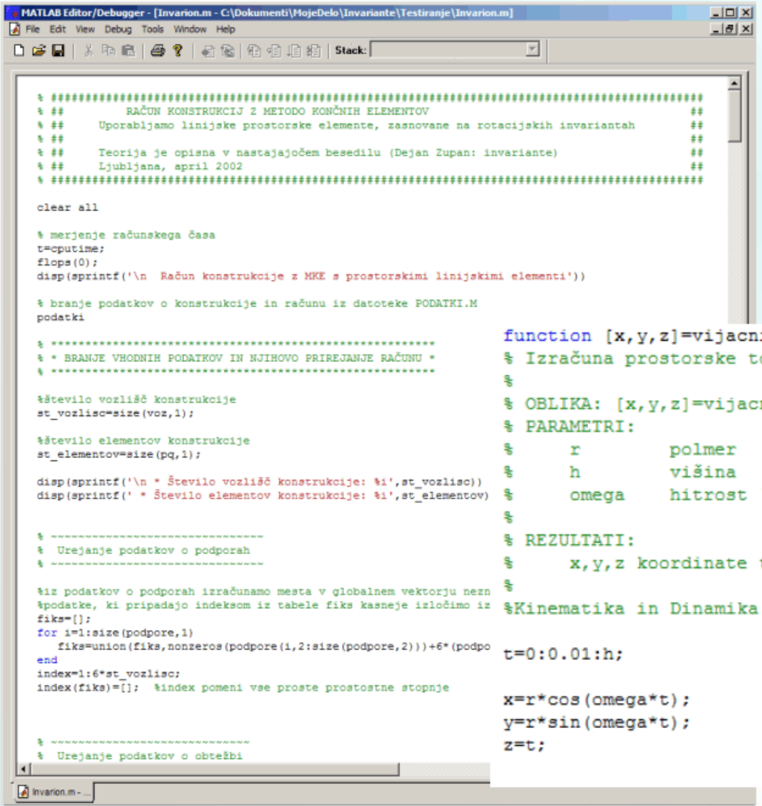
Kaj prikazuje film ?

14. (Gibanje delca po sinusoidi) Delec se giblje s časom po ravnini po krivulji $\sin(t)$. Pričetek gibanja je v koordinatnem izhodišču. Napiši funkcijo, ki animira gibanje delca. Delec prikaži kot rdeč krogec ('ro'), gibanje pa naj poteka do časa T, ki naj bo vhodni podatek. Prva verzija naj prikazuje zgolj gibanje delca v ravnini, druga pa naj riše tudi sled. Pozor! Z ukazom axis([0,T,–1,2]) omogoči, da je območje, na katerega rišeš, ves čas enako.

III. Del

PROGRAMIRANJE






```
function [x,y,z]=vijacnica(r,h,omega)
% Izračuna prostorske točke na vijačnici
%
% OBLIKA: [x,y,z]=vijacnica(r,h,omega)
% PARAMETRI:
%     r         polmer
%     h         višina
%     omega     hitrost 'navijanja'
%
% REZULTATI:
%     x,y,z koordinate točk na vijačnici, v odvisnosti vhodnih podatkov
%
%Kinematika in Dinamika 2002/03, Dejan Zupan, Katedra za mehaniko

t=0:0.01:h;

x=r*cos(omega*t);
y=r*sin(omega*t);
z=t;
```

Osnovna aritmetika

1.– 2.

- Števila (realna števila) so predstavljena v pomični piki (Matlab upošteva IEEE standard).
- Matlab pozna tudi kompleksna števila (z uporabo imaginarne enote).
- Zaradi standardov računa v pomični piki so v Matlabu definirane nekatere posebne količine:

KOLIČINA	POMEN
eps	relativna natančnost računa v pomični piki
i,j	imaginarni enoti
Inf	∞ (npr 1/0)
NaN	ni število (npr 0/0)
flops()	število operacij v pomični piki

- ZAOKROŽANJE: round(). . .zaokroži k najbližjemu celemu številu;

  zaokrožamo še s: floor(), ceil(), fix().

Spletna stran
Naslovnica
Kazalo
Stran 35 od 114
Nazaj
Poln zaslon
Zapri
Končaj

Logični operatorji

3.– 6.

Poleg aritmetičnih operatorjev (poglavje 5) pozna Matlab tudi primerjalne in logične operatorje:

OPERATOR	POMEN
$<, <=$	manjše, manjše ali enako
$>, >=$	večje, večje ali enako
$==, \tilde{}=$	je enako, ni enako
$\&$	logični IN
$\|$	logični ALI
$\tilde{}$	logični NE.

- Primerjamo lahko le matrike enake velikosti.
- Primerjanje (logična operacija) poteka po komponentah.
- V tesni zvezi z gornjimi operatorji je funkcija find:

  i=find(logični izraz)...vrne nabor ideksov komponent, pri katerih je logični izraz pravilen.

M datoteke

- Programe pišemo v obliki tekstovnih datotek s končnico .m
  - pomembna prednost datotek pred ukaznim oknom je večvrstičnost;
  - v datotekah lahko zberemo več prirejanj in ukazov skupaj – opisne m datoteke;
  - lahko pa pišemo (v skladu s pravili) tudi funkcije – funkcijske datoteke;
- novejše različice Matlaba vključujejo urejevalnik za pisanje programa – MATLAB Editor/Debbuger.

Opisne datoteke

7.– 8.

- Nimajo vhodnih in izhodnih parametrov.
- Vanje nanizamo več ukazov in prirejanj hkrati.
- Pri klicu datoteke se ukazi in prirejanja izvedejo po vrsti, vse spremenljivke pa postanejo globalne.
- Pomemben znak je ; – tako v delovnem oknu preprečimo izpis rezultatov "vmesnih" ukazov.
- Uporabljamo jih predvsem za podajanje obsežnejših (večvrstičnih) podatkov.
- Zgled:
```
A=[1 2 3
   4 5 6
   7 8 9]
B=ones(3);
C=A+B; det(C)
```

Funkcijske datoteke

- Prejmejo vhodne in vračajo izhodne parametre.
- Spremenljivke v funkciji so lokalne; le izhodni parametri vplivajo na delovno okolje.
- Tako zapišemo nove funkcije za MATLAB, ki so povsem enakovredne vgrajenim.
- Pri zapisu funkcij imamo na voljo vse možnosti, ki jih izkoriščajo vgrajene funkcije:
  - vgradnjo pomoči, dostopno z ukazom help ime_funkcije;
  - uporabo privzetih vrednosti za manjkajoče podatke;
  - možnost spremenljivega števila vhodnih in izhodnih parametrov.

Struktura funkcijske datoteke

9.– 13.

GLAVA	function ime(x). . .procedura function y = ime(x) function [y,z] = ime(x1,x2,x3)
POMOČ	vrstice, ki se pričnejo s % takoj za glavo funkcije, so pomoč dostopne so z ukazom help ime_funkcije
TELO	prireditveni ukazi operacije z vhodnimi podatki operacije z lokalnimi količinami zanke prirejanje vrednosti rezultatov

Zgled1:

```
function rezultat = ime_funkcije(vhodni_podatki)
%pomoč – opis funkcije
<< telo funkcije >>;
rezultat = izraz;
```

Zgled2:

```
function [rezultat1,rezultat2] = ime_funkcije(vhodni_podatki)
%pomoč – opis funkcije
<< telo funkcije1 >>
rezultat1 = izraz1;
<< telo funkcije2 >>
rezultat2 = izraz2;
```

Klic v ukaznem oknu:

- Funkcijo “poženemo” kot ime_funkcije(vrednosti_vhodnih_podatkov).
- Kličemo lahko le funkcije v trenutnem delovnem področju.
- Pozor! Klic funkcije kot zgoraj vrne le prvi izhodni podatek.
- Vse izhodne podatke dobimo z ukazi oblike [y1,y2,y3,...]=ime_funkcije(vrednosti_vhodnih_podatkov).

Zanki IF in CASE

14.– 15.

ZANKA	OBLIKA	LASTNOSTI
IF	if logični_izraz stavki; end	◦ izvrednoti logični izraz ◦ kadar je ta pravilen, izvede vse stavke v zanki

ELSE
ELSEIF
	if logični_izraz1
	   stavki1;
	elseif logični_izraz2
	   stavki2;
	else
	   stavki3;
	end
		◦ else NIMA logičnega pogoja
		◦ izvede se na koncu, če noben logični pogoj ni bil pravilen
		◦ elseif se izvede, če logični izraz pri if ni bil pravilen
		◦ uporabimo lahko več elseif

CASE	switch spremenljivka
	   case vrednost 1
	      stavki1;
	   case . . .
	      ⋮
	   otherwise
	      stavki;
	end
		◦ primerja vhodno vrednost z vrednostmi v stavkih
		◦ izvede se tisti sklop, kjer sta obe vrednosti enaki
		◦ otherwise pomeni neobvezen blok, ki se izvede, če noben logični pogoj ni bil pravilen

Zgled:

```
function y = pozitivno(x)
%vrne 1, če je število pozitivno, sicer pa nič
if x>=0
   y=1;
else
   y=0;
end

function vrednosti(x)
%izpiše ali je število 1, 0, -1, nič ali kaj drugega
switch x
   case 0
      disp('nič');
   case –1
      disp('minus ena');
   case 1
      disp('ena');
   otherwise
      disp('nekaj drugega');
end
```

Zanki FOR in WHILE

16.– 20.

ZANKA	OBLIKA	LASTNOSTI
FOR	for indeks=območje stavki; end	◦ indeks preteče območje začetek:inkrement:konec ◦ za vsako vrednost indeksa se izvede telo zanke ◦ pogosto zanke for nastopajo vgnezdeno
WHILE	while logični_izraz stavki; end	◦ dokler je logični izraz pravilen, se izvaja telo zanke ◦ znotraj stavkov (običajno) vplivamo na logični izraz– tako dosežemo konec zanke

BREAK	zanka for ali while
	    stavki1;
	    if logični_izraz
	        break
	    end
	    stavki2;
	end
		◦ break prekine izvajanje zanke for ali while
		◦ za ukazom break se izvedejo stavki za zanko
		◦ pri vgnezdenih zankah skoči iz najbolj notranje zanke

Zgled:

```
function C = produkt(A,B)
%izračuna produkt matrik
for i=1:size(A,1)
   for j=1:size(B,2);
      C(i,j)=sum(A(i,:).*B(:,j));
   end
end

function s = sled(A)
%izračuna sled matrike A
i=1;
s=A(1,1);
while i<min(size(A))
   i=i+1;
   s=s+A(i,i);
end
```

Dodatne možnosti funkcij

- SPREMENLJIVO ŠTEVILO PARAMETROV
  - nargin vrne število vhodnih parametrov, s katerimi smo poklicali funkcijo;
  - primer:

```
function c=produkt(a,b)
    if (nargin==1)
        c=a*a;
    else
        c=a*b;
    end
```

- GLOBALNE SPREMENLJIVKE

oblika:

```
function rezultat=ime_funkcije(vnos)
    global ALFA BETA
    telo_funkcije;
```

uporaba:

```
global ALFA BETA
    ALFA=1, BETA=0;
    ime_funkcije(parametri)
```

- VGNEZDENE (POMOŽNE) FUNKCIJE:
  - pogosto potrebujemo več krajših pomožnih funkcij;
  - take funkcije lahko dodamo v datoteko glavne funkcije;
  - prva funkcija v datoteki je glavna, ostale pa so podrejene;
  - pri klicu imajo vgnezdene funkcije prednost pred vsemi ostalimi, tudi vgrajenimi funkcijami z istim imenom;
  - primer:

```
function [rez1,rez2]=glavna(x,y)
    vsota=x+y;
    rez1=pomozna1(x,vsota)
    rez2=pomozna2(y,vsota)

function r1=pomozna1(a,vsota)
    telo_pomozne_funkcije;

function r2=pomozna2(b,vsota)
    telo_pomozne_funkcije;
```

Naloge (III.)

1. Izračunaj v Matlabu!

   (a) eps
   (b) 1/0
   (c) 1/Inf
   (d) 0/0
   (e) sin(0)/0
   (f) sin(eps)/eps
   (g) sin(10ˆ(–100))/(10ˆ(–100))

2. Izračunaj naslednje izraze in rezultate preveri v Matlabu!

   (a) round(6 / 9 + 3 * 2)
   (b) floor(6 / 9 + 3 * 2)
   (c) ceil(6 / 9 + 3 * 2)

3. Za vektorja x = [1 5 2 8 9 0 1] in y = [5 2 2 6 0 0 2] izvedi in komentiraj naslednje ukaze!

   (a) x > y

(b) x == y

(c) x <= y

(d) x | y

(e) x & y

(f) x & (~y)

4. V tej nalogi analiziraj način dostopa do komponent vektorjev z uporabo indeksa 0 oziroma 1 na primernem mestu v naboru indeksov. Naj bo x = 1:10 in y = [3 1 5 6 8 2 9 4 7 0]. Napovej in izračunaj vrednosti naslednjih izrazov!

   (a) (x > 3) & (x < 8)

   (b) x(x > 5)

   (c) y(x <= 4)

   (d) x((x < 2) | (x >=8))

   (e) y((x < 2) | (x >=8))

   (f) x(y < 0)

5. Definiraj vektor x = [1 –5 0 8 –9 0 1] in izračunaj izraze!

   (a) y=x; y(y<0)=[ ]

   (b) x(x~=0)=pi*x/10

(c) `x(x~=0)=pi*x(x~=0)/10`

(d) `x(x==0)=eps`

6. Za vektor x = [3 10 -9 12 –1 0 –12 8 5 1] napiši ukaze, s katerimi

   (a) postaviš pozitivne komponente x na nič;

   (b) nadomestiš večkratnike števila 3 s 3 (funkcija mod(a,b) vrne ostanek pri deljenju a z b);

   (c) pomnožiš lihe komponente x s številom 5;

   (d) prepišeš komponente, ki so večje kot 10, v vektor y;

   (e) poiščeš komponente x, ki so večje od nič in manjše ali enake 8.

7. Sestavi preprosto opisno datoteko, v kateri definiraš matriko A velikosti 4x4 in stolpec b s štirimi elementi! V opisni datoteki izračunaj še vektor c, ki reši sistem enačb Ax=b. Z uporabo podpičja prepreči izpis vseh prirejanj in računov! V delovnem oknu poženi opisno datoteko. Z ukazi A, b in c izpiši podatke in rezultat.

8. (Opisna datoteka ravninskega paličja) Ravninsko paličje lahko v Matlabu preprosto podamo z nekaj matrikami:

   - matrika koordinat vozlišč (npr T=[0 0; 0 2; 2 0; 2 2; 4 0; 4 2; 6 0; 6 2]);
   - matrika povezav med vozlišči za palice (npr. pq= [1 2; 1 3; 1 4; 2 3; 2 4; 3 4; 3 5; 3 6; 4 5; 4 6; 5 6; 5 7; 5 8]);

- matriki lastnosti palic (npr. EI=19E+12, A=0.000362);
- vektor fiksnih prostostnih stopenj (npr. fix=[1 2 17 18]);
- matrika obtežb (npr. f=[11 1; 12 –4]).

(a) Za navedene (ali za svoje) podatke sestavi opisno datoteko ravninskega paličja podatki_palicja.m! Datoteka naj bo čimbolj pregledna – vrstice matrik naj torej ne bodo ločene s podpičjem, temveč zapisane v različnih vrsticah datoteke.

(b) V ukaznem oknu prikliči podatke o paličju in nariši graf vozlišč paličja (to je zgolj graf točk, ne povezav, zato uporabimo parameter; npr '+' ali '*').

(c) Nariši paličje.

(d) Nariši podpore.

9. Napiši funkcijo, ki izračuna hipotenuzo pravokotnega trikotnika z znanima katetama. Poskusi zapisati funkcijo tako, da bo delovala tudi za več trikotnikov hkrati (prvi parameter je stolpec prvih katet vseh trikotnikov, drugi parameter je stolpec drugih katet vseh trikotnikov).

10. Napiši funkcijo, ki vrne dolžino tretje stranice trikotnika po kosinusnem pravilu

$$c^2 = a^2 + b^2 - 2abcos(t),$$

kjer sta a in b znani stranici in t kot med njima.

11. Napiši funkcijo, ki izračuna vrednost števila π po formuli

$$\frac{\pi^2-8}{16}=\sum_{n=1}^{\infty}\frac{1}{(2n-1)^2(2n+1)^2}.$$

Seveda ne računamo neskončne vsote, temveč seštevamo do nekega velikega naravnega števila N. Za dani veliki N naj torej funkcija vrne oceno za π. To oceno primerjaj z vgrajeno vrednostjo za različne N.

12. (Koordinatna transformacija) Eulerjeva kota ϑ in ψ določata koordinatno transformacijo iz referenčnega v lokalni koordinatni sistem. Pripadajoča transformacijska matrika je

$$\mathrm{Q}=\begin{bmatrix}\cos\psi\cos\vartheta & \sin\psi\cos\vartheta & -\sin\vartheta\\ -\sin\psi & \cos\psi & 0\\ \cos\psi\sin\vartheta & \sin\psi\sin\vartheta & \cos\vartheta\end{bmatrix}.$$

Napiši funkcijo, ki za dana Eulerjeva kota ϑ in ψ ter dani vektor X = $[X_1, X_2, X_3]^T$, zapisan v referenčni bazi, vrne pripadajoči vektor x = QX v lokalni bazi. V ukaznem oknu preveri, kam se preslikajo bazni vektorji ([1 0 0]^T, [0 1 0]^T in [0 0 1]^T) za nekaj različnih Eulerjevih kotov (npr. $\vartheta=\pi/2$ in $\psi=0$; $\vartheta=\pi$ in $\psi=-\pi$; $\vartheta=\pi/3$ in $\psi=\pi/6$...).

13. Z ukazoma who in whos preveri stanje spremenljivk v delovnem oknu. Prepričaj se, da so lokalne spremenljivke v funkcijah res neznane v delovnem okolju.

14. Napiši dve preprosti funkciji. Vhodni parameter naj bo n, izhodni pa m. Telo obeh funkcij je predpisano!

(a) Telo prve funkcije:

```
if n > 1
   m = n+1;
else
   m = n - 1;
end
```

(b) Telo druge funkcije:

```
if n < 5
   m = 2*n;
elseif n < 10
   m = 9 - n;
elseif n < 100
   m = sqrt(n);
else
   m = n;
end
```

Napovej rezultate in jih preveri za n=–10, 0, 1, 7, 80, 300.

15. Napiši funkcijo, ki za dano vrednost x izračuna

$$y(x) = \begin{cases} 2 & \text{za } x < 6 \\ x - 4 & \text{za } 6 <= x < 20 \\ 36 - x & \text{za } 20 <= x \end{cases} .$$

Spletna stran
Naslovnica
Kazalo
Stran 53 od 114
Nazaj
Poln zaslon
Zapri
Končaj

16. Napiši funkcijo, ki za dani vektor x izračuna

    (a) vsoto elementov x (uporabi zanko for);
    (b) vektor delnih vsot s (s(j) je vsota elementov v x z indeksi od 1 do j).

    Funkcijo testiraj vsaj na vektorju x = [1 8 3 9 0 1]!

17. Za dana vektorja x in y napiši funkcijo, ki vrne naslednje količine:

    (a) matriko A z elementi A(i,j)=x(i)/y(j);
    (b) matriko B z elementi B(i,j)=x(i)*y(N–j), kjer je N velikost vektorja y;
    (c) matriko C z elementi C(i,j)=x(j)*y(i)/(2+x(i)+y(j)).

    Rezultate za x = [4 1 6] in y = [6 2 7] izračunaj “peš” in z napisano funkcijo.

18. Napiši funkcijo, ki za dano matriko pregleda vse elemente (dve zanki for) in vse elemente, ki so manjši kot 0.3 postavi na 0, vse ostale pa na 1. Funkcijo testiraj za naključno matriko poljubne dimenzije! Ali znaš nalogo rešiti brez zank for? (Dodatno: uporabi funkcijo s for zankama in funkcijo brez na zelo veliki naključni matriki. Z uporabo funkcije CPUTIME (help cputime) se prepričaj o njunih hitrostih)

19. Število π lahko računamo tudi po naslednjem algoritmu http://www.netcom.com/~hjsmith/Pi/Gauss_L.html:

(a) definiraj a = 1, b = 1/sqrt(2), t = 1/4 in x = 1;

(b) ponavljaj naslednje ukaze, dokler razlika med a in b ni dovolj majhna:

y = a;

a = (a + b)/2;

b = sqrt(b*y);

t = t − x*(y − a)ˆ2;

(c) ocena za pi je moj_pi = ((a + b)ˆ2)/(4*t).

Napiši funkcijo ki za dano natančnost (razliko med a in b) oceni število π po opisanem algoritmu!

20. Napiši funkcijo, ki za dano število n izvaja naslednja ukaza, dokler je n večji od 1:

    (a) za sode n naj n postane n/2;

    (b) za lihe n pa naj n postane 3*n+1.

    Funkcija naj vrne vektor števil, ki jih je zavzel n, dokler se je zanka izvajala.

Težje naloge

1. Kreiraj vektor enajstih naključnih celih števil med 31 in 75!

2. Vemo, da je limita izraza $\frac{\sin x}{x}$, ko gre x proti nič, enaka 1. Torej lahko definiramo

$$\frac{\sin x}{x} = \begin{cases} 1, & x = 0 \\ \frac{\sin x}{x}, & \text{sicer} \end{cases}.$$

Zaradi omejene natančnosti računanja $\frac{\sin x}{x}$ ne da pravilnega rezultata za x blizu nič.

   (a) Popravi gornjo definicijo, da bo gornji izraz dobro definiran v smislu numeričnega računa!

   (b) Naj bo x=[pi/2 0 eps pi/6 10^(–100)]. Pravilno izračunaj izraz sin(x)./x.

3. (Rotacijska matrika) Rotacija v prostoru je določena z enotskim vektorjem n = $\begin{bmatrix} n_1 \\ n_2 \\ n_3 \end{bmatrix}$ in s kotom zasuka ϑ. Če definiramo matriko N = $\begin{bmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{bmatrix}$, je s formulo R $=$ I$+\sin\vartheta$N$+(1-\cos\vartheta)$ N^2 določena rotacijska matrika.

   (a) Napiši funkcijo RotM, ki za dani kot in enotski vektor na osi rotacije vrne rotacijsko matriko!

   (b) Izračunaj rotacijsko matriko za vektor rotacije n = $\frac{1}{\sqrt{3}}$[1 –1 1]^T in kot rotacije $\vartheta = \pi/6$.

(c) Funkcijo RotM izboljšaj, da bo delovala tudi za neenotske vektorje na osi rotacije. (Dodaj vrstico, kjer vektor osi rotacije normiraš tako, da postane enotski).

(d) Z izboljšano funkcijo izračunaj rotacijsko matriko za vektor rotacije n = [1 –1 1]^T in kot rotacije $\vartheta = \pi/6$ ter primerjaj rezultate s tistimi iz točke (b).

(e) Razširi funkcijo RotM tako, da bo vrnila tudi matriko N. Matrika N naj bo drugi izhodni parameter. V ukaznem oknu z uporabo razširjene funkcije izračunaj še N.

4. Permutacijski simbol e_{ijk} je definiran takole

$$
\begin{aligned}
e_{123} &= e_{231} = e_{312} = 1 \\
e_{132} &= e_{213} = e_{321} = -1,
\end{aligned}
$$

za vse preostale nabore indeksov pa je enak nič. Napiši funkcijo, ki za dano permutacijo števil 1, 2 in 3 vrne vrednost permutacijskega simbola! Namig: nabor [i j k] je enak [1 2 3], če ukaz all([i j k]==[1 2 3]) vrne 1.

IV. Del

NAVADNE DIFERENCIALNE ENAČBE

Numerične metode

- Za reševanje običajno uporabimo funkcijo ode45.
- Oblika za uporabo

  [t,X]=ode45('opis',interval,x0,lastnosti,P1,P2,...)

  - 'opis' je m datoteka, kjer zapišemo desne strani sistema diferencialnih enačb prvega reda;
  - interval je območje vrednosti parametra t, na katerem iščemo rešitev;
  - x0 je vektor začetnih vrednosti neznanke x (pri t=0);
  - lastnosti so možnosti funkcije ode45 (natančnost, masna matrika, ustavitev);
  - P1,P2,... so dodatni parametri, ki jih lahko prejme funkcija opis.
- Zelo pomebno je pravilno zapisati opisno datoteko opis.m!

Opis diferencialnih enačb

- Diferencialne enačbe opišemo s funkcijo oblike

```
function dxdt = opis(t,x,flag,P1,P2,...)
    dxdt = [ izraz1
             izraz2
             izraz3 ];
```

- Parametra t in x sta časovni parameter in vektor neznank x(t).
- t in x sta obvezna parametra funkcije.
- P1,P2,... so dodatni vhodni parametri (podane vrednosti). Pred njimi mora zaradi pravilnega delovanja obvezno nastopati parameter flag!
- dxdt je stolpec desnih strani sistema navadnih diferencianih enačb $\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x})$.

Preprost primer reševanja

1.–2.

Rešimo enačbo $\ddot{x} = 2\dot{x} - x + C\mathrm{e}^t$, kjer je C konstanta, ki jo podamo.

1. Enačbo s substitucijo $x_1 = x$ in $x_2 = \dot{x}$ prevedemo na sistem prvega reda

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= 2x_2 - x_1 + C\mathrm{e}^t. \end{aligned}$$

2. Sistem zapišemo v Matlab

```
function dxdt = primer(t,x,flag,C)
    dxdt = [x(2); 2*x(2) – x(1) + C*exp(t) ];
```

3. Poiščemo rešitev

```
[t,X]=ode45('primer',[0 1],[1 0],[ ],2).
```

4. Rešitev narišemo z ukazom

```
plot(t,X(:,1)).
```

Celovit opis začetnih problemov

- Sistem diferencialnih enačb prvega reda podamo s stolpcem desnih strani.
- V eni datoteki bi radi opisali celoten problem, ki ga rešujemo.
- Podajamo lahko:
  - območje reševanja;
  - začetne pogoje;
  - lastnosti numerične metode;
  - Jacobijevo matriko sistema;
  - masno matriko;
  - kontrolo posebnih dogodkov.
- Matlab natančno določa obliko celovitega opisa problema.

```
function varargout = naloga(t,x,flag,p1,p2)

switch flag
   case ' '   % izračunamo vrednosti desnih strani sistema enačb
      varargout{1} = f(t,x,p1,p2);
   case 'init'   % nastavimo začetne pogoje, interval in lastnosti
      [varargout{1:3}] = init(p1,p2);
   case 'jacobian'   % vrnemo vrednosti Jacobijeve matrike sistema
       varargout{1} = jacobi(t,x,p1,p2);
   case 'mass '   % vrnemo masno matriko
       varargout{1} = masa(t,x,p1,p2);
   case 'events' % vrnemo vrednosti, ustavitve in smeri
       [varargout{1:3}] = dogodki(t,x,p1,p2);
   otherwise
       error(['Neznan parameter ' ' ' flag ' ' '.']);
end
```

```
% - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
function dxdt = f(t,x,p1,p2)
    dxdt = [≪vektor desnih strani navadnih diferencialnih enačb≫];
% - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
function [interval,x0,lastnosti] = init(p1,p2)
    interval = [≪območje reševanja≫];
    x0 = [≪začetni pogoji≫];
    lastnosti = odeset(≪lastnosti metode≫);
% - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
function dfdx = jacobi(t,x,p1,p2)
    dfdx = [≪Jacobijeva matrika sistema≫];
% - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
function M = masa(t,x,p1,p2)
    M = [≪masna matrika≫];
% - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
function [vrednosti,ustavitev,smer] = dogodki(t,x,p1,p2)
    vrednosti = [≪vektor vrednosti≫]
    ustavitev = [≪vektor logičnih vrednosti (1-ustavi, 0-nadaljuj)≫];
    smer = [≪smeri bližanja komponent v 'vrednosti' proti 0≫];
% - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Jacobijeva in masna matrika

Za pravilno upoštevanje podanih matrik moramo še nastaviti lastnosti Jacobian in Mass v funkciji init:

lastnosti = odeset('Jacobian','on', ≪ostale lastnosti≫).

lastnosti = odeset('Mass', ≪'M' ali 'M(t)' ali 'M(t,y)'≫, ≪ostalo≫).

Reševanje naloge

Če opisno datoteko shranimo z imenom naloga.m, uporabimo ukaz

[t,X]=metoda('naloga'),

če nastopajo še zunanji parametri, pa

[t,X]=metoda('naloga',[ ],[ ],[ ],p1,p2).

Zaznavanje dogodkov

Za zaznavanje dogodkov 'vključimo' lastnost Events v funkciji init:

lastnosti = odeset('Events','on', ≪ostale lastnosti≫).

in pripravimo izhodne parametre funkcije dogodki:

vrednosti vsaka komponenta predstavlja en dogodek; dogodek je izpolnjen, ko komponenta postane nič

ustavitev je vektor enic in ničel, ki pripadajo dogodkom; vrednost 1 pomeni, da ob izpolnjenem dogodku ustavimo reševanje naloge

smer včasih je pomembno v kateri 'smeri' je bil dogodek izpolnjen;

- -1 pomeni, da zaznavamo le ustrezen dogodek, kjer vrednost pada k nič,
- 1 pomeni, da zaznavamo le ustrezen dogodek, kjer vrednost raste k nič,
- 0 – zaznavamo dogodek ne glede na smer.

Lastnosti metod

LASTNOST	VREDNOSTI	POMEN
AbsTol	pozitiven skalar ali vektor {1e-6}	največja absolutna lokalna napaka metode
RelTol	pozitiven skalar {1e-3}	največja dopustna relativna lokalna napaka metode
Refine	naravno število	faktor povečanja števila točk, v katerih metoda vrne rešitve
InitialStep	pozitiven skalar	zgornja meja za velikost prvega koraka
MaxStep	pozitiven skalar	zgornja meja za dolžino koraka metode
OutputFcn	niz	ime funkcije, ki se izvede po vsakem končanem koraku
MassSingular	yes | no | {maybe}	(ne)singularnost masne matrike

Primer celostnega reševanja začetne naloge

3.–5.

Rešimo začetni problem za $t \in [0, 1]$, podan z

- diferencialno enačbo drugega reda

$$\ddot{x} = 2\dot{x} - x + C\mathrm{e}^t$$

- in začetnima pogojema $x\,(0) = 1$ in $x(0) = 0$.
- C je konstanta problema, ki jo želimo spreminjati.

POSTOPEK:

1. Enačbo s substitucijo $x_1 = x$ in $x_2 = \dot{x}$ prevedemo na sistem prvega reda

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= 2x_2 - x_1 + C\mathrm{e}^t. \end{aligned}$$

2. Sistem zapišemo v Matlab, uporabimo splošno strukturo, le Jacobijeve in masne matrike ter dogodkov ne potrebujemo:

```
function varargout = preprost_primer(t,x,flag,C)

 switch flag
    case ' '
       varargout{1} = f(t,x,C);
    case 'init'
       [varargout{1:3}] = init(C);
    otherwise
        error(['Neznan parameter ''' flag '''.']);
 end
 % -----------------------------------------------
 function dxdt = f(t,x,C)
       dxdt = [x(2)
                   2*x(2) - x(1) + C*exp(t) ];
 % -----------------------------------------------
 function [interval,x0,lastnosti] = init(C)
       interval = [0 1];
       x0 = [1 0];
       lastnosti = odeset('AbsTol',1e-9,'RelTol',1e-6);
 % -----------------------------------------------
```

3. Poiščemo rešitev

   [t,X]=ode45('preprost_primer',[ ],[ ],[ ],2).

4. Rešitev narišemo z ukazom

   plot(t,X(:,1)).

Primer dogodka

Privzemimo še, da je rešitev dopustna le dokler je $x(t) < 2$. Sicer moramo reševanje ustaviti.

DODAMO:

i) case 'events'

```
[varargout{1:3}] = dogodki(t,x,C);
```

ii) lastnosti = odeset('AbsTol',1e-9,'RelTol',1e-6,'Events','on');

iii) function [vrednost,ustavitev,smer] = dogodki(t,x,C)

```
vrednost = [2-x(1)];
ustavitev = [1];
smer = [0];
```

Uporaba masne matrike.

Rešujmo sedaj sistem prvega reda, kjer so leve strani drugačne

$$C\dot{x}_1 = x_2$$
$$x_1\dot{x}_1 + 2C\dot{x}_2 = 2x_2 - x_1 + C\mathrm{e}^t.$$

Sistem lahko zapišemo v matrični obliki kot

$$\begin{bmatrix} C & 0 \\ x_1 & 2C \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ 2x_2 - x_1 + C\mathrm{e}^t \end{bmatrix}.$$

DODAMO:

i) case 'mass'

```
varargout{1} = masa(t,x,C);
```

ii) lastnosti = odeset('AbsTol',1e-9,'RelTol',1e-6,'Mass','M(t,y)');

iii) function M = masa(t,x,C)

```
M = [ C      0
      x(1)   2*C];
```

Naloge (IV.)

1. Začetno nalogo

$$y'' = \frac{x}{y} + \frac{y}{x}, \qquad y(0) = 0,\ y'(0) = -1$$

   reši z Matlabom. Uporabi preprost način, kjer opišeš le desne strani sistema dveh diferencialnih enačb prvega reda.

2. Numerično reši naslednji sistem dveh diferencialnih enačb drugega reda z uporabo Matlaba:

$$\begin{aligned} y'' - 5y' + 2y &= 0 \\ z'' + 2z' + 4z &= 0. \end{aligned}$$

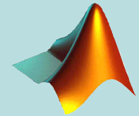
   Sistem prevedi na štiri diferencialne enačbe prvega reda. Uporabi preprost način, kjer opišeš le desne strani sistema. Pri klicu numerične metode pa podaj še $y(0) = z(0) = 0$ in $y'(0) = z'(0) = 0$. Rešitev poišči na območju $[0, 10]$.

3. (a) Diferencialno enačbo

$$y'' = \frac{x}{y} + \frac{y}{x}$$

   prevedi na sistem dveh diferencialnih enačb prvega reda.

   (b) Dobljeni sistem, skupaj z ostalimi podatki o nalogi, vključi v Matlabovo opisno datoteko problema.



(c) Časovni interval naj bo $[1, 10]$, začetni vrednosti pa $y(1) = 1$ in $y'(1) = -1$.

(d) Pri uporabi splošne sheme opisne datoteke upoštevaj, da ne potrebuješ paramerov, Jacobijeve in masne matrike.

(e) Problem reši z uporabo funkcije ode45.

4. Nihanje matematičnega nihala. Pri matematičnem nihalu je kot med nihalom in navpičnico (θ) določen z naslednjo diferencialno enačbo drugega reda

$$\ddot{\theta} = -\frac{g}{l} \sin \theta,$$

kjer je l dolžina nihala in g težnostni pospešek.

(a) Zapiši enačbe v obliki, primerni za Matlab.

(b) Dolžina nihala in težnostni pospešek naj bosta vhodna parametra naloge.

(c) Začetni odmik $\theta(0)$ naj bo tudi vhodni parameter, za $\dot{\theta}(0)$ pa naj velja $\dot{\theta}(0) = 2\theta(0)$.

(d) Rešitev mora zaznati vse čase pri katerih je nihalo prešlo ravnovesno lego $\theta(t) = 0$ (uporabi funkcijo dogodki s komponento vektorja ustavitev enako nič). Nihalo pa ustavimo, če kot θ doseže vrednost $\frac{\pi}{2}$ (v funkciji dogodki je ustrezna komponenta vektorja ustavitev enaka ena).

(e) Poišči numerične rešitve za nekaj različnih začetnih odmikov nihala od navpičnice in vsakič prikaži grafe kota θ v odvisnosti od časa.

5. Uporaba masne matrike. Vzemimo sistem dveh diferencialnih enačb drugega reda, kjer odvodi niso separirani:

$$\begin{aligned} zy'' - 5y'z'' + 2yz &= 0 \\ yz'' + 2z'y'' + 4yz &= 0. \end{aligned}$$

Odvodov ni potrebno separirati, temveč uporabi možnost zapisa naloge z masno matriko! Naloga predpisuje še $y(0) = z(0) = 0$ in $y'(0) = z'(0) = 0$, rešitev pa iščemo na območju $[0, 10]$.

(a) Enačbi najprej prevedi na sistem štirih diferencialnih enačb prvega reda. Na levi naj bodo odvodi, na desni pa preostali členi.

(b) Iz levih strani izrazi 'masno' matriko.

(c) Desne strani in masno matriko vključi v splošno shemo opisne datoteke, kjer za ta primer ne potrebujemo dodatnih parametrov.

(d) Problem reši z uporabo funkcije ode45 in nariši grafa funkcij y in z.

(e) Dodatek. Recimo, da je rešitev dopustna le, dokler produkt yz ne doseže nič z negativne strani. V opisno datoteko dodaj funkcijo, ki opiše pogoje za ustavitev.

V. Del

DELO Z DATOTEKAMI

Vrste datotek

- m datoteke
  - tekstovne datoteke, a je format vseeno predpisan
  - branje iz ukazne vrstice.
- ASCII datoteke
  - imajo enako število vrednosti v vsaki vrstici
  - števila so ponavadi ločena s presledki
- textovne datoteke različnih formatov
  - možnost prilagajanja
  - popolna kontrola datoteke
- zapisi slik (.jpg, .tif, .bmp, .png, .hdf, .pcx, .xwd)
- zapisi zvoka (.wav)
- datoteke drugih programskih jezikov (C, Fortran)

Ukazi za branje in zapisovanje

TIP	BRANJE	PISANJE
ASCII	load ime_dat dlmread('ime_dat','dlm')	save ime_dat M -ascii dlwrite('ime_dat', M,'dlm')
TEKST	fopen ukazi za formatirano branje fclose	fopen ukazi za formatirano pisanje fclose
SLIKA	A=imread('slika','format'); image(A)	export figure imwrite print
ZVOK	Y=wavread('ime_dat')	wavwrite(Y,'ime_dat')

1.– 3.

ASCII datoteke

- V vsaki vstici mora biti enako števil.
- Dovoljeni so komentarji tipa % komentar; ostali znaki niso dovoljeni.

Če so števila ločena s presledki, npr. aski.dat:

$$\begin{matrix} 1 & 2 & 3 \\ 3 & 5 & 7 \end{matrix}$$

uporabimo ukaz load aski.dat. Ime datoteke postane ime količine.
Če so števila ločena z znaki, npr. aski2.dat:

$$\begin{matrix} 1; & 2; & 3; \\ 3; & 5; & 7; \end{matrix}$$

uporabimo ukaz M=dlmread('aski2.dat',';').
Za pisanje datoteke pa bi uporabili save aski3.dat M -ascii ali dlwrite('aski4.dat', M,'~')

Textovne datoteke

4.– 5.

Odpiranje in zapiranje datotek

- Odpiranje: fid=fopen('ime_datoteke','dovoljenje');

  fid je številski indeks datoteke, z odprto datoteko pa lahko delomo le v skladu s parametrom:

DOVOLJENJE	POMEN
r	branje
w	pisanje (ustvari, če je treba)
a	dodajanje (ustvari, če je treba)
r+	branje in pisanje (ne ustvari)
w+	sprazni ali ustvari za branje in pisanje
a+	branje in dodajanje.

- Zapiranje: status=fclose(fid):
  - status=0: uspešno zapiranje datoteke;
  - status=1: zapiranje datoteke ni uspelo.
- Odprtih je lahko več datotek hkrati; vse zapremo z fclose('all').

Formatirano branje in pisanje

- Branje: prebrano=fscanf(fid,'format',dim).
- Pisanje: fprintf(fid,'format',vrednost).
- Pomeni parametrov:
  - fid je indeks datoteke
  - dim je neovezen parameter, ki predstavlja število prebranih elementov tipa '%format'

    lahko je skalar ali vektor z dvema komponentama
  - 'format' je niz znakov, ki pomenijo način zapisa in določajo:
    * poravnavo;
    * besedilo med posameznimi števili
    * število mest in število mest za decimalno piko;
    * način zapisa števil;
    * prehod v novo vrsto, itd

SIMBOL	POMEN
\b	pomik nazaj
\f	pomik naprej
\n	prehod v novo vrsto
\t	večji horizontalni presledek (tab)

FORMAT	POMEN
%c	znak
%d	decimalni zapis
%e	eksponentni zapis (3.1415e+00)
%g	krajši od zapisov %d in %e
%f	zapis v fiksni piki
%i	celo število
%s	niz znakov.

Primeri:

```
fprintf(fid,'%6.2f %12.8f\n',y);
fprintf('Obseg enotskega kroga je %g.\n',2*pi)
```

Zaslon

Ukaz za formatiran izpis v zaslonskem je povsem analogen

sprintf('format',vrednost).

za izpis nizov pa lahko uporabimo

disp('niz')

Po analigiji z fscanf uporabimo sscanf za pretvarjanje nizov v števila:

prebrano=fscanf(niz,'format').

Za branje podatkov iz ukazne vstice pa uporabljamo ukaz input

ginp=input('Podaj vrednost: ').

Naloge (V.)

1. Izračunaj magični kvadrat dimenzije 20 krat 20 in ga zapiši na dva načina v ASCII datoteko. V prvem načinu loči števila s presledki, v drugem pa z poljubnim simbolom. Z ukazom clear all pobriši vse spremenljivke. Sedaj magični kvadrat ne računaj, temveč ga preberi z datoteke. Vse skupaj poskusi še za večje dimenzije.

2. Iz dane datoteke .xls izlušči blok med stopci B in I ter vrsticami 22 in 30. Blok izriši v Matlabu v logaritemskem merilu (po vrsticah v odvisnosti od indeksa vrstice).

3. Dano sliko v png formatu prikaži v grafičnem oknu.

4. Napiši funkcijo, ki za dano matriko pregleda vse elemente (dve zanki for) in vse elemente, ki so manjši kot 0.3 postavi na 0, vse ostale pa na 1. V tekstovno datoteko vpiši naslov, matriko na začetku, opis naloge in končno matrko.

5. Število π lahko računamo tudi po naslednjem algoritmu (http://www.netcom.com/˜hjsmith/Pi/Gauss_L.html):

   (a) definiraj a = 1, b = 1/sqrt(2), t = 1/4 in x = 1;

   (b) ponavljaj naslednje ukaze, dokler razlika med a in b ni dovolj majhna:
   y = a;
   a = (a + b)/2;

b = sqrt(b*y);
t = t – x*(y – a)ˆ2;

(c) ocena za pi je moj_pi = ((a + b)ˆ2)/(4*t).

Napiši funkcijo ki za nekaj podanih natančnosti (razliko med a in b) v tekstovno datoteko izpiše po vrsticah naslednje trojice števil:

- natančnost
- oceno števila π na 10 decimalnih mest natančno
- napako v eksponentnem zapisu (decimalnih mest naj bo največ 5).

VI. Del

OSTALE PODATKOVNE STUKTURE

Razpršene matrike

1.– 3.

- Včasih ima matrika veliko ničel.
- Take matrike zapišemo ceneje s tremi nabori podatkov:
  - seznamom neničelnih elementov;
  - in pripadajočima seznamoma indeksov vrstic in stolpcev.
- Generiranje razpršene matike:
  - ukaz sparse: S=sparse(i,j,s,m,n), kjer so:
    i – indeksi vrstic, j – indeksi stolcev,
    s – neničelni elementi glede na pare (i,j),
    m,n – dimenziji matrike;
  - po komponentah in blokih:
    S=sparse(m,n),
    S(i,j)=vrednost, S(i1:i2,j1:j2)=matrika.

- Povezava s polnimi matrikami:
  - sparse(A) iz polne matrike naredi razpršeno;
  - full(S) iz razpršene matrike naredi polno;
  - spy(S) nariše grafični prikaz neničelnih elementov.
- Delo z razpršenimi matrikami:
  - večina matričnih funkcij deluje na razpršenih matrikah;
  - veliko jih zna izkoristiti razpršenost, rezultat je tudi razpršena matrika;
  - operatorji ($*$, $+$, $\backslash$, $/$) med razpršenimi matrikami ohranijo razpršenost;
  - če komponiramo polno in razpršeno matriko je rezultat polna matrika, izjemi sta $.*$ in &;
  - posebne funcije:

    sprand ustvari naključno razpršeno matriko

    speye ustvari enotsko razpršeno matriko.

Večrazsežni seznami

4.

- Matrike so dvorazsežni seznami – sistemi z dvema indeksoma A(i,j).
- Razsežnosti je lahko tudi več, ustrezno se samo poveča število indeksov:

  A(i,j,k,l,. . .).

- Generiranje večrazsežnih seznamov:

  1. po ravninah
     A(:,:,1)=matrika1, A(:,:,2)=matrika2, . . .;
  2. z vgrajenimi funkcijami na treh ali več indeksih:
     rand(i,j,k,l,. . .), zeros(i,j,k,l,. . .), ones(i,j,k,l,. . .),
     repmat(A,[i,j,k,l,. . .]);
  3. funkcija cat B=cat(dim,A1,A2,. . .)
     razporedi zaporedje seznamov A1, A2, . . . vzdolž razsežnosti dim.

- Manipulacija z večrazsežnimi seznami:
  - podatki o velikosti:
    size(A) vrne velikosti vzdolž vseh razsežnosti,
    ndims(A) vrne razsežnost seznama;
  - dostopanje do kosov je enako kot pri matrikah, le indeksov je več A(i,j,k,l), A([i1 i2],j,k1:k2);
  - preoblikovanje: B=reshape(A,[ni nj nk ...]),
    pri tem se mora ohraniti stevilo elementov;
  - stiskanje: squeeze(A) odstrani dele razsežnosti 1.
- Operacije na večrazsežnih seznamih:
  - funkcije, ki delujejo po komponentah lahko uporabimo tudi pri večjih razsežnostih (npr. sin, exp, +, .*, logični operatorji);
  - vektorske funkcije (sum, max,...) delujejo vzdolž 1. razsežnosti po celotnem seznamu;
  - matrične funkcije in operatorji ($*$, $\hat{}$, $\backslash$, $/$) ne delujejo na večrazsežnih seznamih –
    uporabiti moramo zanke po dvorazsežnih delih seznama.

Celični seznami

5.– 6.

- Celični seznami združujejo elemente različnih velikosti in tipov (števila, vektorje,matrike, nize,...) v skupen objekt z uporabo zavitih oklepajev { }.

- Generiranje celičnih seznamov:

  1. poljubna vsebina, vključena v zavite oklepaje

     C={celica1,celica2}   C={celica1 celica2; celica3 celca4};

     npr. C={A, sum(A), det(A)};

  2. po komponentah: C(i,j)={A}   ali C{i,j}=A.

- Elemente dosegamo z indeksi v zavitih oklepajih C{i}, C{i,j};

  če je v celici matrika, najprej z indeksi dosežemo celico, potem pa še komponente matrike: C{i,j}(k,l).

- Celične sezname lahko gnezdimo,

  npr. {A,{[1 2 3; 3 4 5],5},'gnezdo'}

- Razširitev na več razsežnosti je analogna kot pri matrikah.

Strukture

7.

- Strukture so seznami objektov, organizirani po ‘podatkovnih poljih’
  (npr. vsak študent ima ime, priimek, vpisno število, itd.).

- Vnašanje podatkov:

  1. po posameznih poljih
     s(indeks).polje=vsebina
     npr.:

student(1).ime=’Miha’	student(2).ime=’Janez’
student(1).priimek=’Novak’	student(2).priimek=’Kralj’
student(1).vpisno_st=78934	student(2).vpisno_st=78935
student(1).tabela=[1 2 4; 3 8 7]	student(2).tabela=[1 3 4; 3 9 7]

  2. celoten indeks hkrati z ukazom struct:
     s(indeks)=struct(’polje1’,vsebina1,’polje2’,vsebina2,. . .).

- Dostopanje do podatkov:
  - s.polje samo izpiše podatke celotnega seznama na navedenem polju;
  - [s.polje] podatke na navedenem polju v vseh elementih zlije v horizontalni smeri;
  - {s.polje} podatke na navedenem polju združi v celični seznam
  - s(indeks).polje(i,j) tako dostopamo do posamezne komponente, če je na polju matrika;
- Brisanje polj: rmfield(s,'polje').
- Gnezdenje: s.prvi_nivo.drugi_nivo

  npr. s(2).ime.geslo='test', s(2).ime.nadimek='Mihelj'
- Strukture so lahko tudi večrazsežne: s(i,j,k,...).polje

Naloge (VI.)

1. V Matlab vnesi matriki

$$A_1 = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 5 & 4 & 3 & 2 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix};$$

   kot razpršeni matriki.

   (a) Izračunaj njuno vsoto, produkt in determinati.

   (b) Z ukazom spy si oglej grafično predstavitev obeh matrik.

   (c) Ustvari razpršeno matriko B oblike

$$B = \begin{bmatrix} A_1 & A_1^2 & 0 \\ 0 & A_2 & 0 \\ [3] & 0 & A_2^2 \end{bmatrix},$$

   kjer $[3]$ pomeni matriko samih trojk ustrezne velikosti.

2. Generiraj razpršeno matriko iz 10 polnih blokov dimenzije 6x6 po naslednjem postopku:

   (a) Ustvari novo razpršeno matriko dimenzije 33x33.

(b) Vstavljani bloki naj bodo oblike round(10*rand(6)).

(c) Z zanko po blokih vstavljaj posamezne bloke v celotno matriko.

(d) i-ti blok vstaviš na mesto 3*(i-1)+1 (indeksa vrstice in stolpca sta enaka). Pri tem ne smeš prekriti starih vrednosti temveč novi blok prištej stari vsebini.

(e) Izpiši dobljeno matriko z ukazoma spy in full.

3. Vzemimo velik sistem enačb dimenzije tisoč krat tisoč. Matrika sistema ima enice po diagonali, pod diagolalo so dvojke. Na drugi poddiagonali so v vrsticah z lihimi indeksi zapisane trojke. Desne strani predstavlja stolpec števil od 1 do 1000.

   (a) Sestavi matriko sistema S (razpršena matrika).

   (b) Izračunaj rešitev sistema enačb Sx=b in preštej število operacij v pomični piki.

   (c) Razpršeno matriko S prepiši v polno matriko A.

   (d) Reši sistem Ax=b in ponovno preštej število operacij v pomični piki (pozor pred štetjem operacij je treba uporabiti ukaz flops(0)).

4. Naj bo prerez nosilca mnogokotnik z štirimi vozlišči

$$P = \begin{bmatrix} y1 & y2 & y3 & y3 \\ z1 & z2 & z3 & z3 \end{bmatrix},$$

vendar pa se prerezi z višino spreminjajo. Prereze celotnega nosilca podaj s trirazsežno matriko. Pri tem prizemimo znane zveze med ogljišči na različnih višinah. Višine naj bodo celoštevilski indeksi. Prvo vozlišče je fisno $v_1(h) = (y_1, z_1) = const$. Za drugo vozlišče velja $v_2(h) = (y_2 + h, z_2 + h)$, za tretje $v_3(h) = (h * y_3, h * z_3)$ in za četrto $v_3(h) = (h * y_4 + h, h * z_4 + h)$, $h = 1, 2, 3, ..., 7$.

5. Napiši funkcijo, ki generira celični seznam binomskih koneficientov do dane globine N. Pomoč: Binomske koeficiente določimo iz Pascalovega trikotnika

   1
   1 1
   1 2 1
   1 3 3 1

   (Vsak koeficient se tvori tako, da seštejemo dva koeficienta nad njim).

6. Izračunaj ogljišča pravilnih n-stranih enotskih likov, kjer gre n od 3 do 10. Rezultate shrani v primerni obliki. (Cilj je s preprostim ukazom dobiti podatke za izbran n).

7. Ustvari preprost seznam študentov z naslednjimi polji: zaporedna_stevilka, ime, priimek, letnik, vpisna_stevilka, povprecna_ocena, seznam ocen (ta naj bo iz več vrstic, vsaka vrstica predstavlja ocene po letu študija).

   (a) Vstavi vsaj 5 elementov seznama.

   (b) Uredi elemente po priimku. Izpiši seznam imen in priimkov po vrstnem redu priimkov.

(c) Izračunaj povprečno oceno vseh ocen vseh študentov.

(d) Izračunaj povprečno oceno po letnikih za vsakega študenta in za vse skupaj.

(e) Briši vse podatke o seznamu ocen.

VII. Del

GRAFIČNI VMESNIKI (GUI)

Vgrajeni urejevalnik

- Programi za prijetnejši za uporabo, če uporabljamo grafične vmesnike.
- Izdelava grafičnega vmesnika je lahko za začetnika zelo zahtevna.
- V Matlabu izdelavo zelo poenostavi vgrajeni urejevalnik, imenovan GUIDE (GUI editor).
  - ukaz guide nam pripravi vse potrebno za začetek izdelave novega grafičnega vmesnika (odpre se okno, v katerem narišemo novo grafično okolje).
  - ukaz guide ime datoteke omogoča popravljanje in spreminjanje obstoječih GUI-jev.

Risanje vmesnika

- Pozenemo ukaz guide in izberemo možnost Blank GUI.
- Prilagodimo velikost polja (premikamo desni spodnji vogal).
- Na osnovno polje rišemo tekstovna polja in gumbe;
  - Edit text polje za vnos podatkov;
  - Static text polje za izpis besedila;
  - Slider drsnik po območju [0,1];
  - ListBox seznam možnosti;
  - Push Button gumb za potrditev ali klic ukaza;
  - Radio Button in Check Box polji za izbiro (neizbiro);
  - Axes polje za prikaz grafike.

Programiranje gumbov

- Koda se generira avtomatsko že ob risanju polj in gumbov.
- To je le osnovna rutina (večinoma vsebuje prirejanje in le glave rutin, ki jim moramo dati e vsebino).
- Prenos med polji in gumbi poteka prek strukture handles.
- Pomembno je poimenovanje polj in gumbov.

Razumevanje grafičnih objektov

Osnovne (temeljne) rutine

- ravna črta (line)
- poligonalni lik (patch)
- ploskev v 3D (surface)
- text (text)
- rasterska slika (image)
- koordinatni sistem (axes)
- posplošeni pravokotnik (rectangle)

Sestavljene (kompleksnejše) rutine

- poligonalna črta (plot)
- ploskve (surf, mesh)
- območja (area)
- stolpični diagrami (bar, stem)
- konture (contour)
- vektorska polja (quiver)
- stopničaste funkcije (stairs)
- srednje vrednosti in odkloni (errorbar)
- točkovne vrednosti v ravnini ali prostoru (scatter)

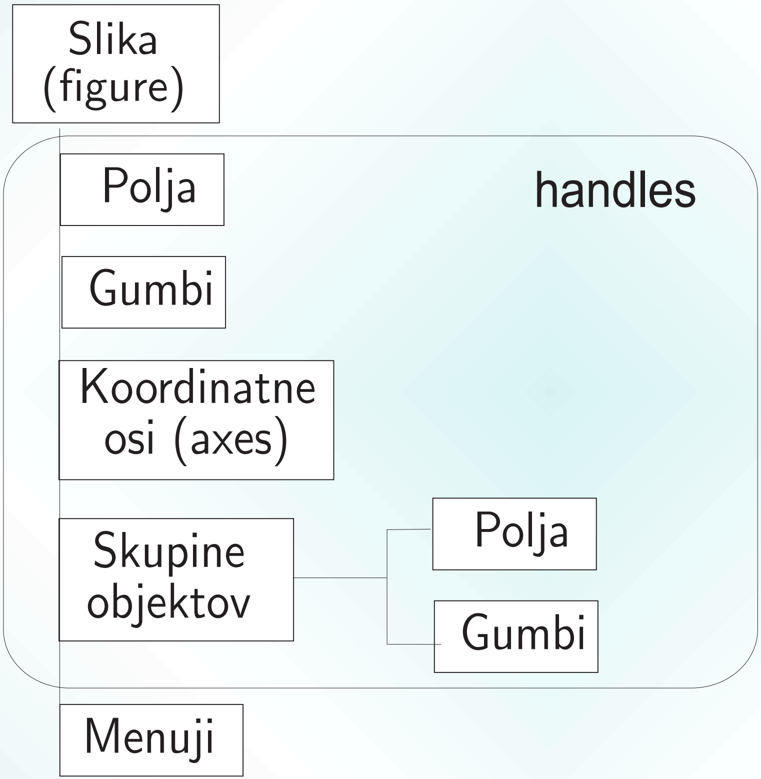
Lastnosti grafičnih objektov

- Že priklicu grafične rutine lahko navedemo barve in naštejemo ostale lastnosti.
- Lastnosti grafičnih objektov so vezane na strukturiranost grafike in jih lahko nastavljamo tudi posebej, vezane na strukturiranost grafike in kazalce na grafične objekte.
  - Z ukazom set nastavimo lastnost:
    set(GraficniObjekt,'ImeLastnosti','VrednostLastnosti');
  - Z ukazom get pa prebemo vse lastnosti ali zahtevano lastnost:
    get(GraficniObjekt);
    get(GraficniObjekt,'ImeLastnosti');

- Grafične objekte naslavljamo z vgrajenimi ali nastavljenimi naslovi:
  - 0 pomeni zaslonsko okno računalnika
  - gcf pomeni trenutno grafično okno
  - gca pomeni trenutne osi (koordinatni sistem)
  - crta=line([1 2],[2 4]) pomeni narisano linijo
- Primeri uporabe:
  - get(0)
  - get(gcf)
  - set(gcf,'color','r')
  - set(gca,'LineThickness',3)
  - crta=line([1 2],[3 3]), set(crta,'color','g')
  - vsebina=get(gca,'Children'), set(vsebina,'Visible','off')

- Še nekaj primerov:
  - ```
    zaslon = get(0,'ScreenSize');
    figure(105);
    set(105,'Position',[30 30 scrsz(3)*0.5 scrsz(4)*0.5 ]);
    set(105,'Name','Naslov','NumberTitle','off','Menubar','none');
    ```
  - ```
    set(plot(2:0.1:3,sin(2:0.1:3),'b'),'LineWidth', 1.3);
    ```
    ali
    ```
    plot(2:0.1:3,sin(2:0.1:3),'b','LineWidth', 1.3);
    ```
  - ```
    set(hp, 'LineWidth', 1.2,'Color',[0.2,0.7,0.3]);
    ```
  - ```
    text(2,2,num2str(5),.../n
        'HorizontalAlignment','left','VerticalAlignment',.../n
        'baseline','FontUnits','points','FontSize',10,.../n
        'FontWeight','bold','Margin',1.5,'Color',[1 0 0],.../n
        'BackgroundColor',[1 1 1],'EdgeColor',[1 0 0]);
    ```

Struktura grafičnega vmesnika



Upravljanje s polji in gumbi

- Polja in gumbe naslavljamo prek njihovih imen (lastnost tag).
- Struktura, ki vsebuje vsa polja in gumbe vmesnika se imenuje handles.
- Vsebino in vrednosti polj in gumbov upravljamo, kot pri ostali grafiki:
  - Z ukazom set nastavimo lastnost:
    set(handles.ImePolja,'ImeLastnosti','VrednostLastnosti');
  - Z ukazom get pa prebemo zahtevano lastnost:
    get(handles.ImePolja,'ImeLastnosti');

Poglavitne lastnosti

- 'String' ... besedilo v polju.
  - Uporaba:
    podatek1=get(handles.vnos1,'String')
    set(handles.komentarji,'String','delam.....')
  - Potrebno pretvarjanje iz nizov v števila in obratno!
- 'Value' ... Vrednost(i) v polju.
  - Uporaba:
    vkljucen=get(handles.gumb1,'Value')
    set(handles.gumb,'Value',1)
  - Posebej pomembno pri izbirah in drsnikih!
- 'Visible' ... vidno ali skrito polje.
  Uporaba: set(handles.skupina,'Visible','off')

Programi za gumbi

- Vsakemu gumbu in polju pripada rutina ImePolja_Callback.
- Taka rutina se izvede, ko kliknemo na polje ali gumb.
  - Klici rutin, ki pripadajo poljem in gumbom se izvedejo v okolju vmesnika, če ne zahtevamo drugače.
  - Kadar želimo rezultate v globalnem delovnem okolju uporabimo ukaz evalin
  - Primer: evalin('base','clc')

Kazalo

Spletna stran

Naslovnica

Kazalo

Stran 112 od 114

Nazaj

Poln zaslon

Zapri

Končaj

Literatura

[1] The MathWorks, Inc. MATLAB, Using MATLAB, Natick, 1999.

[2] The MathWorks, Inc. MATLAB, Using MATLAB Graphics, Natick, 1999.

[3] Matlab na straneh podjetja MathWorks, http://www.mathworks.com/products/matlab/

[4] M. Saje, D. Zupan, Kinematika in dinamika, http://www.km.fgg.uni-lj.si/predmeti/KiD/KD.htm

[5] MATLAB Educational Sites, http://www.eece.maine.edu/mm/matweb.html

[6] University of New Hampshire: MATLAB-tutorial, http://spicerack.sr.unh.edu/˜mathadm/tutorial/software/matlab/

[7] University of Florida: MATLAB Tutorial, http://www.math.ufl.edu/help/matlab-tutorial/

[8] Naval Postgraduate School: MATLAB-manual,
http://www.met.nps.navy.mil/manual/matlab.html

[9] Massachusetts Institute of Technology: MATLAB on Athena,
http://web.mit.edu/olh/Matlab/TOC.html

[10] Z. Bohte, Numerične metode, DMFA Slovenije, Ljubljana, 1991.