Naučni softverski alati

Rad sa stringovima i polinomima

Osnovno o stringovima

- String u MATLAB-u je niz karaktera.
- Stringovi se zadaju pod apostrofima:

```
s = 'MATLAB 2016'
```

- Pošto je string niz, pojedinačnim elementima pristupamo isto kao kod numeričkih nizova. Na primjer:

```
>> s = 'MATLAB 2016';
>> disp(s)
MATLAB 2016
>> s(1:5)
ans =
MATLA
>> disp(s(1:2:end))
MTA 06
>> disp(s(end:-1:1))
6102 BALTAM
```

ASCII tabela kodova karaktera

Kod	Taster ili simbol
32	(razmaknica)
33	!
34	"
35	#
36	$
37	%
38	&
39	'
40	(
41	)
42	*
43	+
44	,
45	-
46	.
47	/
48	0
49	1
50	2
51	3
52	4
53	5
54	6
55	7
56	8
57	9
58	:
59	;
60	<

Kod	Taster ili simbol
61	=
62	>
63	?
64	@
65	A
66	B
67	C
68	D
69	E
70	F
71	G
72	H
73	I
74	J
75	K
76	L
77	M
78	N
79	O
80	P
81	Q
82	R
83	S
84	T
85	U
86	V
87	W
88	X
89	Y

Kod	Taster ili simbol
90	Z
91	[
92	\
93	]
94	^
95	_
96	`
97	a
98	b
99	c
100	d
101	e
102	f
103	g
104	h
105	i
106	j
107	k
108	l
109	m
110	n
111	o
112	p
113	q
114	r
115	s
116	t
117	u
118	v

Kod	Taster ili simbol
119	w
120	x
121	y
122	z
123	{
124	|
125	}
126	~

Funkcije abs i char

- Svakom karakteru je dodijeljen jedinstven prirodan broj, koga nazivamo ASCII kôd karaktera. ASCII je šema za kodiranje karaktera.
- Funkcija **abs** vraća ASCII kôd karaktera argumenta, dok **char** vraća karaktere koji odgovaraju ASCII kôdu argumenta funkcije.

```
>> s = 'ABC123abc';
>> kodovi = abs(s)
kodovi =
    65   66   67   49   50   51   97   98   99
>> char(kodovi)
ans =
     ABC123abc
>> setstr(abs('ABC123abc'));%ranija verzija f-je char
ans =
     ABC123abc
```

Nadovezivanje stringova

- Stringovi se u MATLAB-u vrlo jednostavno nadovezuju, poštujući pravila za nadovezivanje nizova:

```
s = [s1, s2]
```

- Primjer:

```
>> s1 = 'Dobar dan';
>> s2 = 'dobri ljudi';
>> s3 = [s1,' ',s2]
s3 =
Dobar dan dobri ljudi
```

- s = [s1; s2] – u ovom slučaju s1 i s2 moraju imati isti broj karaktera

Poređenje stringova

- Za poređenje stringova se koristi funkcija **`strcmp(s1,s2)`**, koja poredi stringove `s1` i `s2`.
- `strcmp` vraća logičku 1 (true) ako su stringovi identični (svi karakteri isti, pri čemu se pravi razlika između velikih i malih slova), i logičku 0 (false) ako nijesu.
- Slična funkcija je **`strcmpi(s1,s2)`**, koja ne pravi razliku između malih i velikih slova.

```
>> s1 = 'Podgorica';
>> s2 = 'PODGORICA';
>> disp(strcmp(s1,s2))
     0
>> disp(strcmpi(s1,s2))
     1
```

Pretraga stringova

- Za određivanje da li string *s1* sadrži dati podstring *s2*, može poslužiti funkcija **`findstr(s1,s2)`**, koja pronalazi sve pozicije na kojima se string *s2* pojavljuje u stringu *s1*.
- Rezultat je vektor početnih pozicija ukoliko *s2* ima više karaktera.
- Ukoliko string *s2* ne postoji u *s1*, `findstr` vraća prazan string.

```
>> findstr('Banana','na')
ans =
     3     5

>> findstr('Jabuka','na')
ans =
     []
```

Funkcija strrep

- Funkcija **`strrep(s1,s2,s3)`** u stringu `s1` mijenja string `s2` stringom `s3`.

```
>> a = 'dan kao svaki drugi dan';
>> b = 'dan';
>> c = 'covjek';
>> strrep(a,b,c)

ans =

covjek kao svaki drugi covjek
```

- Obratiti pažnju da s2 i s3 ne moraju imati isti broj karaktera

Funkcije upper i lower

- Funkcija **`upper(s)`** sva mala slova u stringu `s` pretvara u velika, ostale karaktere ne mijenja.
- Funkcija **`lower(s)`** sva velika slova u stringu `s` pretvara u mala, ostale karaktere ne mijenja.

```
>> upper('Jabuka')
ans =
  JABUKA

>> lower('JABUKA')
ans =
  jabuka
```

Određivanje opsega karaktera

- Uočimo iz ASCII tabele sledeće:
  - Slova (mala i velika) su poređana u prirodno rastućem redosljedu engleskog alfabeta (A, B, C, D, ..., a, b, c, d, ...),
  - Cifre su poređane u rastućem redosljedu (0, 1, 2, 3, ...),
  - Skupovi malih i velikih slova se ne nadovezuju jedan na drugi (prvo dolaze velika slova, pa nekoliko specijalnih karaktera, pa mala slova).
- Ove osobine možemo iskoristiti da provjerimo da li karakter pripada skupu malih slova, velikih slova ili cifara.

`s(i)>='a' & s(i)<='z'` uslov da je `s(i)` malo slovo

`s(i)>='A' & s(i)<='Z'` uslov da je `s(i)` veliko slovo

`s(i)>='0' & s(i)<='9'` uslov da je `s(i)` cifra

- Za provjeru da li je karakter stringa slovo, može poslužiti funkcija **`isletter(s(i))`**, koja vraća 1 ako je `s(i)` slovo (veliko ili malo) i 0 u suprotnom.

Funkcija isstrprop

- Koristeći funkciju **`isstrprop(s,k)`**, možemo odrediti da li elementi stringa *s* pripadaju kategoriji *k*. Neke od kategorija karaktera su:
  - Slova (velika i mala)
  - Cifre
  - Slova i cifre (alfanumerički karakteri)
  - Bjeline (spejs, tab, Enter)
- Na primjer, za string `s='MATLAB 2015'`, imamo:

`isstrprop(s(1),'alpha')` vraća 1, isto kao funkcija `isletter`

`isstrprop(s(1),'digit')` vraća 0. Pita da li je s(1) cifra.

`isstrprop(s(1),'alphanum')` vraća 1

`isstrprop(s,'alpha')` vraća [1 1 1 1 1 1 0 0 0 0 0]

`isstrprop(s,'alphanum')` vraća [1 1 1 1 1 1 0 1 1 1 1]

`isstrprop(s,'wspace')` vraća [0 0 0 0 0 0 1 0 0 0 0]

Funkcija num2str

- Funkcija **num2str(x)** konvertuje broj x u odgovarajući string. Na primjer, `num2str(-34.21)` vraća string `'-34.21'`.
- Kad se primijeni na matricu, vraća matricu stringova:

```
>> num2str([2.3, 0.17; -45.01, 100])
ans =
   2.3          0.17
-45.01           100
```

- Funkcija `num2str` je pogodna za kreiranje stringova koji sadrže numeričke podatke (`title`, `xlabel`, `ylabel`, `text`) .

```
>> x = 23.21;
>> s = ['Vrijednost x-a je: ', num2str(x)];
>> disp(s)
Vrijednost x-a je: 23.21
```

- Funkcija **int2str(x)** konvertuje cijeli broj x u odgovarajući string. Ukoliko x nije cijeli broj, zaokružuje ga na najbliži cijeli broj i rezultat konvertuje u string. Na primjer, `int2str(34)` vraća string `'34'`, `int2str(34.6)` vraća string `'35'`.

Funkcija eval

- Pomoću funkcije **`eval(s)`** možemo izračunati vrijednost izraza definisanog stringom `s`. Sting *s* se se tumači i izvršava kao naredba.
- Primjeri:

`eval('147')` vraća broj 147

`eval('147+100')` vraća broj 247

`eval('2*147/67')` vraća broj 4.3881

`x=2; eval('2*x-7')` vraća broj -3

`x=2; y=5; eval('(x-y)^2')` vraća broj 9

`x=2.11; eval('sin(x)')` vraća broj 0.8581

Funkcija mat2str

- Pomoću funkcije **`str2mat(s1,s2,s3)`** možemo formirati matricu čiji su redovi proizvoljni stringovi različite dužine (s1, s2, s3 u navedenom pozivu). Izvršava se automatsko popunjavanje prazninama pojedinih vrsta.
- Primjer:

```
>> g = str2mat('x = 5','y = 4','z = x^2 + y');
>> eval(g(1,:))
x =
     5
>> eval(g(2,:))
y =
     4
>> eval(g(3,:))
z =
    29
```

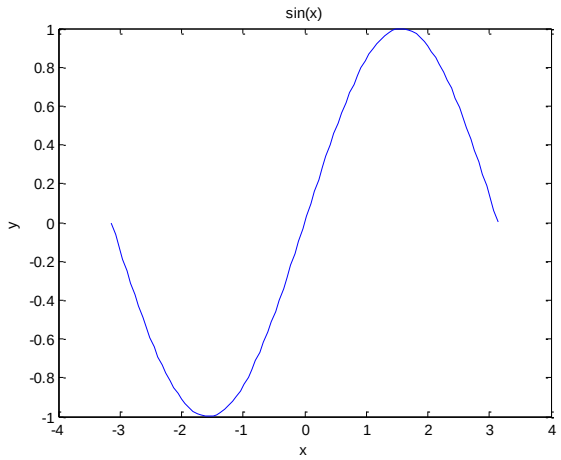
Prvi primjer sa stringom

- Napisati m-fajl u kojem će se izračunavati vrijednost funkcije zadate u tekstualnoj promjenjljivoj i crtati grafik u intervalu $-\pi \leq x \leq \pi$

```
s = input('Unesi funkciju ', 's');
% navođenjem 's' kao drugog argumenta funkcije input
% izbjegava se unošenje stringa pod apostrofima
x = linspace(-pi, pi);
plot(x, eval(s));
xlabel('x - osa')
zlabel('y - osa')
title(s)
```

Jedno izvršenje

```
Unesi funkciju sin(x)
```



Drugi primjer sa stringom

- Napisati funkcijski fajl, koji za ulazni argument ima string s, a kao rezultat daje unijeti string bez razmaka i broj obrisanih razmaka.

```
function [s1,br]=brisi_razmake(s)
s1=[];
k=0;
for i=1:length(s)
    if(s(i)~=' ')
        s1=[s1,s(i)];
        k=k+1;
    end
end
br=length(s)-length(s1);% može i br=length(s)-k;
```

```
function [s1,br] = brisi_razmake(s)
s1 = []; br = 0;
for i = 1: length(s)
    if(s(i) == ' ' )
        br = br +1;
    else
        s1 = [s1,s(i)];
    end
end
```

```
>> [a,b]=brisi_razmake('Dobar dan dobri ljudi')
a =
Dobardandobriljudi
b =
     3
```

Jedno izvršenje

Treći primjer sa stringom

- Napisati m-fajl kojim se provjerava da li se u stringu koji se unosi nalazi veliko slovo.

```
s=input('Unesi string');
ind=0;%pretpostavimo da nema velikog slova
for i=1:length(s)
    if(s(i)>='A'&s(i)<='Z')
        ind=1; % naišli smo na veliko slovo
        break
    end
end
if(ind==1)
disp(['Pojavljuje se veliko slovo, na poziciji ',...
    int2str(i)])
end
```

Jedno izvršenje

```
Unesi string'dobar Dan'
Pojavljuje se veliko slovo, na poziciji 7
```

Polinomi - koeficijenti i nule

$$y = c(1)x^n + c(2)x^{n-1} + \cdots + c(n)x + c(n+1)$$
$$y = 2x^4 - 3x^3 + x$$

- Polinom se može definisati preko svojih koeficijenata

```
>> c=[2,-3,0,1,0]; % koeficijenti polinoma
```

- Nule polinoma čiji su koeficijenti u vektoru `c` se dobijaju kao:

```
>> n = roots(c)
n =
         0
    1.0000
    1.0000
   -0.5000
```

- Dakle, prethodni polinom se može zapisati preko svojih nula, u faktorizovanom obliku $y = x(x-1)(x-1)(x+\frac{1}{2})$

Polinomi - koeficijenti i nule

- Koeficijenti polinoma čije se nule nalaze u vektoru *n* se dobijaju uz pomoć **poly(n)**

```
>> poly(n)

ans =

    1.0000   -1.5000    0.0000    0.5000         0
```

- Koeficijenti polinoma se dobijaju u obliku koji pretpostavlja jedinicu uz stepen najvišeg reda (četvrti u našem slučaju)

Polinomi – određivanje vrijednosti

- Vrijednost polinoma definisanog koeficijentima *c* za dato *x*, se može odrediti funkcijom **polyval(c,x)**

```
>> c = [2, -3, 0, 1, 0]; %koeficijenti polinoma
>> y = polyval(c,2)
y =
    10
```

$$y = 2x^4 - 3x^3 + x$$

- Ukoliko je x vektor, izračunava se vrijednost polinoma za svaki element:

```
>> y = polyval(c, [2, 4])
y =
    10   324
```

- Ukoliko je x matrica, koristi se funkcija **polyvalm(c,x)**

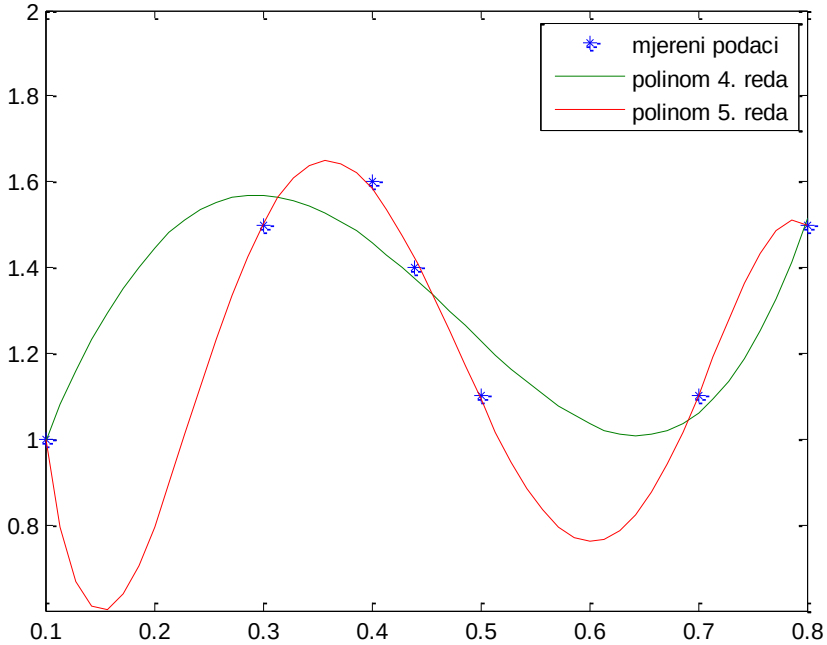
```
>> y = polyvalm(c, [2, 4; 1, 3])
y =
   222   568
   142   364
```

Polinomi – interpolacija polinomom

- Podaci zadati vektorima *x* i *y* se mogu aproksimirati polinomom *n-tog* reda funkcijom **`c = polyfit(x,y,n).`** `c` je vektor koeficijenata dobijenog polinoma

Primjer: Odrediti polinome 4. i 5. reda koji aproksimiraju podatke date vektorima *x* i *y*, a zatim nacrtati mjerene podatke i vrijednost dobijenog polinoma. x = [.1 .3 .4 .44 .5 .7 .8], y = [1 1.5 1.6 1.4 1.1 1.1 1.5]

```
x = [.1 .3 .4 .44 .5 .7 .8];
y = [1 1.5 1.6 1.4 1.1 1.1 1.5];
x1 = linspace(0.1, 0.8, 50);
c1 = polyfit(x, y, 4);
y1 = polyval(c1, x1);
c2 = polyfit(x, y, 5);
y2 = polyval(c2, x1);
plot(x, y, '*', x1, y1, x1, y2)
legend('mjereni podaci', 'polinom 4. reda', 'polinom 5. reda')
```



Polinomi – množenje i dijeljenje

$$y = \frac{c1(1)x^n + c1(2)x^{n-1} + \cdots + c1(n)x + c1(n+1)}{c2(1)x^n + c2(2)x^{n-1} + \cdots + c2(n)x + c2(n+1)}$$

- Ukoliko je stepen brojioca veći ili jednak od stepena imenioca, prethodni izraz se može napisati u obliku

$$y = \frac{y1(x)}{y2(x)} = \frac{r(x)}{y2(x)} + q(x)$$

- gdje je $q(x)$ količnik polinoma, a $r(x)$ ostatak

- Odrediti količnik polinoma $\frac{-2x^4+3x^2+5x+4}{x^3+2x^2-3x}$

```
>> c1 = [-2,0,3,5,4]; c2 = [1,2,-3,0]; [q, r] = deconv(c1, c2)
q =
    -2     4
r =
     0     0   -11    17     4
```

$$\frac{-2x^4+3x^2+5x+4}{x^3+2x^2-3x} = \frac{-11x^2+17x+4}{x^3+2x^2-3x} - 2x + 4$$

```
>> c = conv(c1, c2); % množenje (konvolucija) polinoma
```