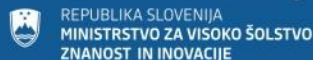


# MATLAB

izr. prof. dr. Božidar POTOČNIK, Fakulteta za elektrotehniko,  
računalništvo in informatiko Univerze v Mariboru

Ljubljana in online, Centralna tehniška knjižnica Univerze v Ljubljani,  
Usposabljanje podatkovnih strokovnjakov, 18. – 21. 11. 2024



# Vsebina

- **Kaj je Matlab?**
- Funkcionalnost in uporaba
- **Prednosti in slabosti**
- GUI okolja MATLAB
- **Urejevalnik**
- **MATLAB za telebane**
- **Še neomenjeno**
- Demo: PCA na 2D sivinski sliki

## Spremenljivke

Sklicevanje in indeksiranje

## Matrične operacije

Funkcije in skripti

## Krmiljenje toka

Vizualizacija

## Kreiranje GUI

Knjižnice/orodja



# Kaj je MATLAB?

- MATLAB = programska in numerična računalniška platforma (*po MathWorks*).
- MATLAB = skovanka iz besed „**MAT**rix“ in „**LAB**oratory“.
- Nastal leta 1984 v podjetju MathWorks.
- Implementiran v programskih jezikih C, C++ in Java.
- Programski jezik četrte generacije, a hkrati tudi programsko okolje.
- Uporablja se lahko z različnimi paradigmi programiranja:
  - Funkcionalnimi,
  - Vizualnimi, in
  - Objektno usmerjenimi



# Kaj je MATLAB?

- MATLAB ima lastno integrirano razvojno okolje (IDE) in zbirko knjižnic.
- Sestoji iz
  - Okolja (*environment*) oz. platforme (*platform*) oz. stroja (*engine*), ter
  - Knjižnic oz. orodij (*toolboxes*)

**Okolje/stroj** – osnovna platforma za izvajanje kode MATLAB.

**Orodja** – dodatni nabori funkcij, ki razširjajo osnovno funkcionalnost MATLABa za specifične naloge.



# Funkcionalnost in uporaba

## Funkcionalnosti (po MathWorks)

- Analiza podatkov
- Napredna vizualizacija
- Programiranje
- Gradnja namiznih in spletnih aplikacij
- Vmesnik do drugih programskih jezikov
- Delo s strojno opremo
- Paralelno računalništvo
- Delo v oblaku

## Uporaba/aplikacije (po MathWorks)

- Umetna inteligenca, Avtomatizirani vozni sistemi, Računalniška biologija, Nadzorni sistemi
- Podatkovna znanost, Globoko učenje, Elektrifikacija, Vgrajeni sistemi
- Podjetniški in IT sistemi, Razvoj FPGA, ASIC in SoC, Obdelava slik in računalniški vid, Internet stvari
- Strojno učenje, Mehatronika, Sistemi mešanih signalov, Prediktivno vzdrževanje, Radarski sistemi
- Robotika, Obdelava signala, Testiranje in merjenje, Brežžične komunikacije



# Prednosti in slabosti

## Prednosti

- Enostavnost uporabe
  - Višjennivojski jezik, interaktivno okolje in GUI
- Združljivost med platformami
- Od naprave neodvisna vizualizacija
- Integracija z drugimi programskimi jeziki
- Obsežna zbirka knjižnic
- Velika uporabniška skupnost

## Slabosti

- Licenčna programska oprema in stroški
- Omejena zmogljivost
  - Interpretiranje kode, velika poraba pomnilnika
- Zaprt ekosistem

# GUI okolja MATLAB

## Uporabniški vmesnik

The screenshot shows the MATLAB R2024a academic use interface. The top menu bar includes options like HOME, PLOTS, and APPS. Below it is a toolbar with icons for file operations, workspace management, and code execution. The main area is divided into several panes:

- File Explorer (left):** Shows the current folder structure, including files like `demo_GUI.m` and `demo_obdelava_podatkov.m`.
- Command Window (center):** Displays the execution of MATLAB code, showing numerical results for matrix operations and error calculations. The code includes `rmse` and `figure` commands.
- Workspace (top right):** Lists variables in the workspace, such as `img`, `img_orig`, and `score`.
- Command History (bottom right):** Shows a list of previously executed commands, including `demo_obdelava_podatkov` and `rmse(A,B)`.

Trenutni direktorij  
in info o datotekah

Delovni prostor

Ukazno okno

Zgodovina ukazov

# Urejevalnik

**Uporabniški vmesnik**

```
4 img_orig = imread('moon.tif');
5 img = im2double(img_orig); % uint8 v tip double za PCA
6 % Prikaz
7 figure; imshow(img);
8 title('Originalna slika');
9
10
11 % ANALIZA PCA
12 % koef - matrika glavnih komponent slike, PC so v stolpcih
13 % score - predstavitev slikovnih podatkov v prostoru glavnih komponent
14 % razlozena_var - polje; koliko variance razloži posamezna PC
15 [koef, score, ~, ~, razlozena_var] = pca(img);
16
17
18 % Prikaz razložene variance za posamezno glavno komponento
19 figure; plot(cumsum(razlozena_var));
20 xlabel('Število glavnih komponent');
21 ylabel('Razložena kumulativna varianca (%)');
22 title(['Razložena varianca s pomočjo ' ...
23 'glavnih komponent']);
24
25 % Izbiranje števila glavnih komponent
26 stKomponent = 20;
27 disp(sum(razlozena_var(1:stKomponent))); % 20 komponent razloži 98.78 % variance
28
29 % Rekonstruiranje slikovnih podatkov v manj dimenzionalnem prostoru
30 img_rekonstruirana = score(:, 1:stKomponent) * koef(:, 1:stKomponent)';
31
32 % Prikaz rekonstruirane slike
33 figure; imshow(img_rekonstruirana);
34 title(['Rekonstruirana slika z ', ...
35 num2str(stKomponent), ' komponentami']);
36
37 % Linearno skaliranje slikovnih podatkov in prikaz
38 img_skalirana = uint8(255.*(img_rekonstruirana - min(img_rekonstruirana(:)))/ ...
39 (max(img_rekonstruirana(:)) - min(img_rekonstruirana(:))));
40 figure; imshow(img_skalirana);
41 title('Skalirana rekonstruirana slika');
42
43
44 % Root Mean Squared Error (RMSE) med originalno
45 % in skalirano sliko
46 disp(rmse(double(img_skalirana), ...
47 double(img_orig), 'all'))
48
49 % Rekonstrukcijska napaka
50 Napaka = abs(double(img_orig) - double(img_skalirana));
51 figure; imagesc(Napaka);
52 title('Rekonstrukcijska napaka');
53 colorbar;
54
55 disp('Konec')
56
```

**Koda funkcije/skripta/aktivnega skripta**

**Integriran razhroščevalnik in profilirnik**





# Spremenljivke

- Ni deklaracij spremenljivk in rezervacije pomnilnika.
- Dinamično tipiziranje.
- Občutljivost na velikost črk (*case sensitive*).
- Možno pretvarjanje med tipi (*type casting*).
- **Primeri**
  - `stevilo = 5; % Kreiranje števila`
  - `vrstica = [1, 2, 3, 4, 5]; % Vektor`
  - `matrika = [1 2 3; 4 5 6]; % Matrika velikosti 2x3`
  - `niz = "Pozdravljen, MATLAB!"; % String`
  - `jePozitiven = (stevilo > 0); % Logična vrednost (true ali false)`
  - `celica = {1, 'tekst', [3 4 5]}; % Celica z elementi mešanih tipov`
  - `struktura = struct('Ime', 'Janez', 'Starost', 25); % Struktura s polji`



# Spremenljivke

- **Osnovne funkcije za delo s spremenljivkami**

- `disp()` – prikaz spremenljivke
- `class()` – tip spremenljivke
- `size()` – velikost/dimenzija spremenljivke
- `ndims()` – število dimenzij
- `rand()` – privzeti naključni generator

- **Pretvorba med tipi (*type casting*)**

- `int32()` – v 32-bitno celo število (integer)
- `double()` – v realno število
- `logical()` – v logično vrednost
- `complex(a,b)` – v kompleksno število  $a + bi$



# Sklicevanje in indeksiranje

- MATLAB indeksira od 1 naprej!
- **Primeri**
  - `vrstica(2)` % Sklicevanje na element na indeksu 2
  - `matrika(2,1)` % Element v vrstici 2 in stolpcu 1
  - `matrika(1,end)` % Element v vrstici 1 in zadnjem stolpcu
  - `podmatrika = matrika(1:2,2:3)` % Izlušči podmatriko velikosti 2x2
  - `celica(2)` % Celica na indeksu 2
  - `celica{2}` % Vrednost celice na indeksu 2
  - `struktura.Ime` % Vrednost v polju 'Ime'



# Matrične operacije

- MATLAB (bil) prvenstveno zasnovan za enostavno delo z matrikami!
- **Operacije na matrikami**
  - Seštevanje/odštevanje:  $A + B$ ,  $A - B$
  - Množenje matrik:  $A * B$
  - Transponiranje matrik:  $A'$
  - Determinanta matrike:  $\det(A)$
  - Inverzna matrika:  $\text{inv}(A)$
- **Operacije na nivoju istoležečih elementov (*element-wise*)**
  - Dodamo piko pred operacijo
  - $.*$  - množenje
  - $./$  - deljenje
  - $.^$  - potenciranje



# Funkcije in skripti

## Funkcije (*functions*)

- Bloki kode, ki izvajajo določeno nalogo.
- Definirani v ločenih datotekah s končnico .m.
- Uporaba rezervirane besede **function** v glavi.
- Ime datoteke se mora ujemati z imenom funkcije.
- Tipično imajo vhodne in izhodne argumente.

## Skripti (*scripts*)

- Izvajajo zaporedja ukazov.
- Shranjeni v datoteki s končnico .m.
- Ne sprejemajo vhodnih ali vračajo izhodnih argumentov.



# Funkcije in skripti

## Funkcija

- Datoteka: sestej\_stevili.m

```
function rezultat = sestej_stevili(a, b)
    rezultat = a + b; % Seštej števili in vrni rezultat
end
```

- V ukaznem oknu

```
izhod = sestej_stevili(3, 5);
disp(izhod); % Prikaže 8
```



# Funkcije in skripti

## Skript

- Datoteka: preprost\_script.m

```
a = 5;  
b = 3;  
izhod = a + b; % Izvedemo seštevanje  
disp(izhod); % Prikažemo 8
```

- V ukaznem oknu  
`preprost_script;`

# Funkcije in skripti

## Aktivni skripti (*live scripts*)

- Interaktivno izvajanje kode z vdelanim izhodom in oblikovanim besedilom (npr. markdown, enačbe ipd.).
- Definirani v datotekah s končnico .mlx.
- Dele kode lahko izvajamo interaktivno, rezultat viden takoj.
- Primerno za učne materiale, poročila ipd.

The screenshot displays the MATLAB Live Editor interface. The top toolbar includes options for file operations (New, Open, Save, Print, Export), navigation (Go To, Find, Bookmark), text formatting (Normal, Bold, Italic, Underline), code control (Code Control, Task, Refactor), and execution (Run, Step, Stop, Run and Advance, Run to End). The editor shows two sections of code:

```

1 %% Sekcija 1: Preprost izračun
2
3 a = 10;
4 b = 5;
5 rezultat = a + b;
6
7 disp("Rezultat a + b je: " + rezultat);

```

The second section of code is partially visible:

```

8 %% Sekcija 2: Izris sinusne funkcije
9
10 x = 0:0.1:10;
11 y = sin(x);
12
13 % Plot
14 plot(x, y)
15 title('Sinusna funkcija')
16 xlabel('x')
17 ylabel('sin(x)')
18 grid on

```

On the right side of the editor, the execution results are shown. The first result is the text "Rezultat a + b je: 15". The second result is a plot titled "Sinusna funkcija" showing a sine wave. The x-axis is labeled "x" and ranges from 0 to 10. The y-axis is labeled "sin(x)" and ranges from -1 to 1. The plot shows a smooth curve oscillating between 1 and -1.





# Krmiljenje toka

## IF stavek

```
a = 0;  
if a > 0  
    disp('a je pozitiven');  
elseif a < 0  
    disp('a je negativen');  
else  
    disp('a je nič');  
end
```



# Krmiljenje toka

## FOR stavek

```
% Izpis lihih števil  
for i = 1:2:10  
    disp(i);  
end
```



# Krmiljenje toka

## WHILE stavek

```
% Izpis lihih števil  
i = 1;  
while i <= 10  
    disp(i);  
    i = i + 2;  
end
```

## Posebno krmiljenje znotraj zank

- `break` – hitreje zapusti zanko ob izpolnjenem pogoju
- `continue` – preskoči trenutno ponovitev zanke



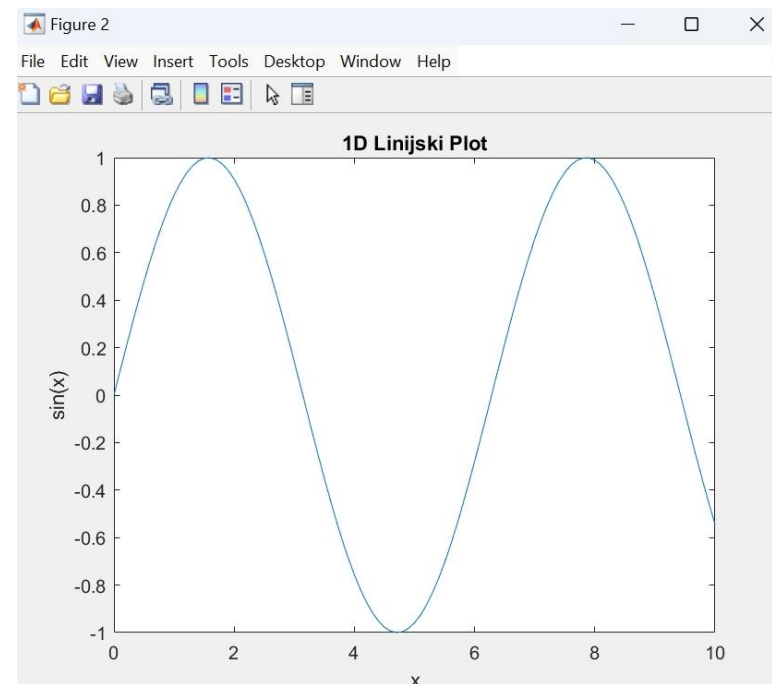
# Vizualizacija

(1/3)

- 1D podatki: `plot()`
- 2D podatki: `scatter()`, `plot()`
- 3D podatki: `surf()`, `plot3()`, `mesh()`
- Podprto shranjevanje vizualiziranih podatkov v datoteke (npr. `.jpg`, `.eps`).

## 1D vizualizacija

```
x = 0:0.1:10;  
y = sin(x);  
plot(x, y); % 1D linijski izris  
title('1D Linijski Plot');  
xlabel('x');  
ylabel('sin(x)');
```

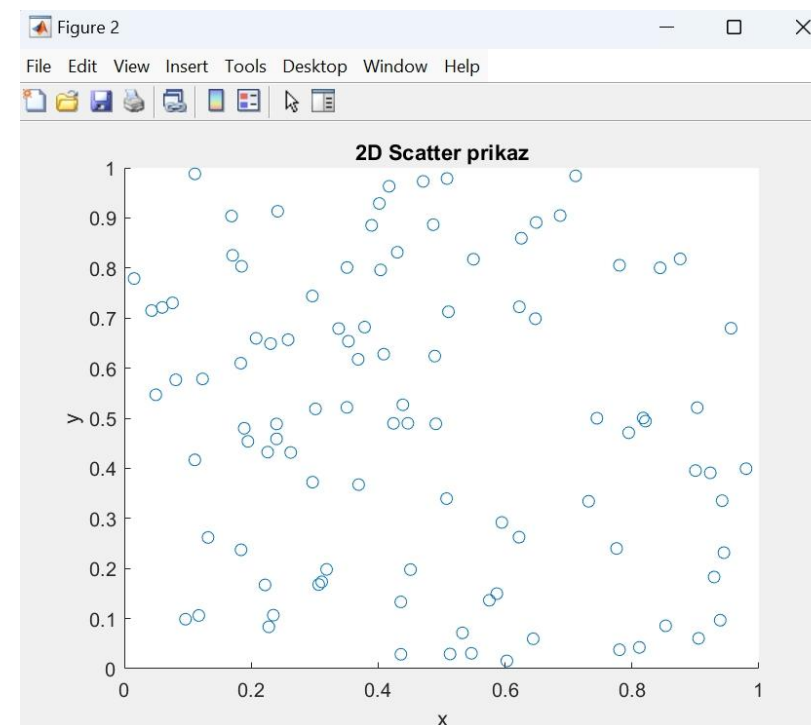




# Vizualizacija

## 2D vizualizacija

```
x = rand(1, 100);  
y = rand(1, 100);  
scatter(x, y); % Kreiranje 2D scatter plot  
title('2D Scatter prikaz');  
xlabel('x');  
ylabel('y');
```

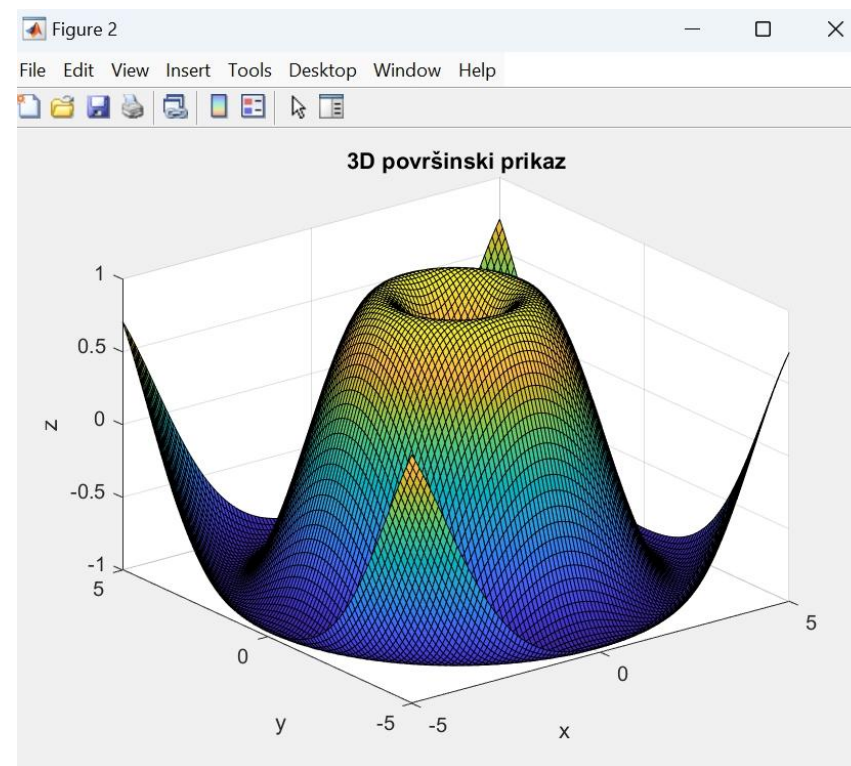




# Vizualizacija

## 3D vizualizacija

```
% Kreiranje mreže za x in y  
[x, y] = meshgrid(-5:0.1:5, -5:0.1:5);  
% z = f(x,y)  
z = sin(sqrt(x.^2 + y.^2));  
surf(x, y, z); % Kreiranje 3D prikaza  
title('3D površinski prikaz');  
xlabel('x');  
ylabel('y');  
zlabel('z');
```



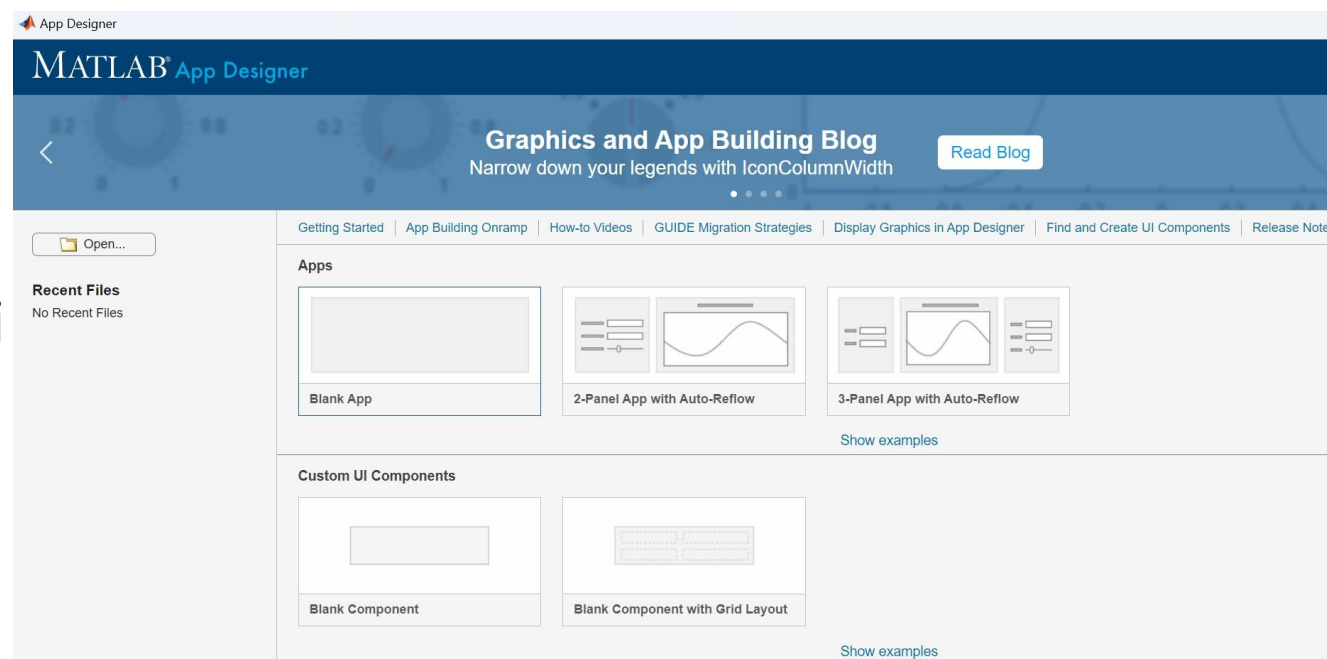
# Kreiranje GUI

- Preprosto ustvarjanje grafičnih uporabniških vmesnikov (GUI)

- Uporaba App Designer
- Programsko s pomočjo funkcije `uifigure()`

- Omogočena interakcija z aplikacijami MATLAB z uporabo grafičnih gradnikov:

- Gumbi
- Drsniki
- Meniji
- Vnosna polja
- Tabele
- ...



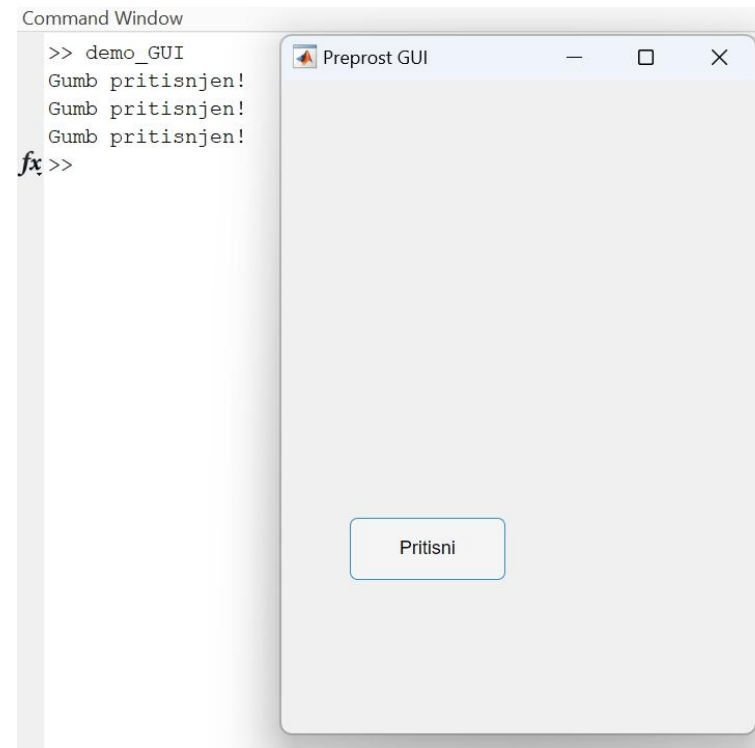


# Kreiranje GUI

(2/2)

## Uporaba gradnika gumb (*button*)

```
% Kreiranje preprostega okna  
fig = uifigure('Name', 'Preprost GUI');  
  
% Dodamo gumb k sliki  
btn = uibutton(fig, 'Position', ...  
    [100, 100, 100, 40], 'Text', 'Pritisni');  
  
% Definiranje callback funkcije za gumb  
btn.ButtonPushedFcn = @(src, ...  
    event) disp('Gumb pritisnjen!');
```







# Knjižnice/orodja

- Statistics and Machine Learning Toolbox
  - Funkcije za statistično analizo, strojno učenje in prileganje podatkov.
- Image Processing Toolbox
  - Funkcije za analizo, obdelavo in vizualizacijo slik.
- Signal Processing Toolbox
  - Orodja za analizo in obdelavo signalov.
- Optimization Toolbox
  - Funkcije za reševanje problemov optimizacije.
- Deep Learning Toolbox
  - Orodja za načrtovanje, učenje in uvajanje modelov globokega učenja.



# Knjižnice/orodja

- Symbolic Math Toolbox
  - Orodja za simbolno računanje, kar omogoča analitično reševanje matematičnih problemov namesto numerično.
- Druge knjižnice:
  - Database Toolbox
  - Parallel Computing Toolbox
  - Control System Toolbox
  - Simulink
  - ...



- Pretvorba kode v izvršljivo (*executable*) datoteko
  - Omogočeno prevajanje uporabnikove kode v samostojne izvršljive datoteke (.exe) s pomočjo prevajalnika MATLAB.
- Integracija z drugimi programskimi jeziki
  - MATLAB Engine API for C/C++
    - Omogočeno klicanje funkcij MATLAB iz programov C/C++.
  - MATLAB Java Interface
    - Omogočeno klicanje funkcij MATLAB iz Jave.
  - Integracija s Python
    - MATLAB omogoča izvajanje kode Python znotraj MATLABa in obratno z uporabo predpone py.
  - Integracija s C/C++
    - MATLAB lahko kliče funkcije C/C++ z uporabo datotek MEX (*MATLAB Executable*) za optimizacijo obdelovanja.



- File I/O
  - MATLAB lahko bere in piše podatke iz različnih formatov datotek:
    - .csv
    - .txt
    - .xls
    - .mat (MATLAB izvorni format)
    - ...
- MATLAB Online
  - MATLAB mogoče izvajati v spletnem brskalniku z uporabo MATLAB Online.



# Demo: PCA na 2D sivinski sliki

(1/7)

- **Ideja**

- Določiti glavne komponente (*principal components, PC*) v slikovnih podatkih (tj. v 2D sliki).
- Izbrati manjši nabor najpomembnejših glavnih komponent.
- Transformirati slikovne podatke v nov, manj dimenzionalen prostor.
- Rekonstruirati slikovne podatke v tem novem prostoru.

- **Cilj**

- Kompresija slikovnih podatkov: Zmanjšati velikost podatkov (oz. prostor za shranjevanje).
- Omejitev: Možna izguba podrobnosti v slikovnih podatkih.

- **Postopek**

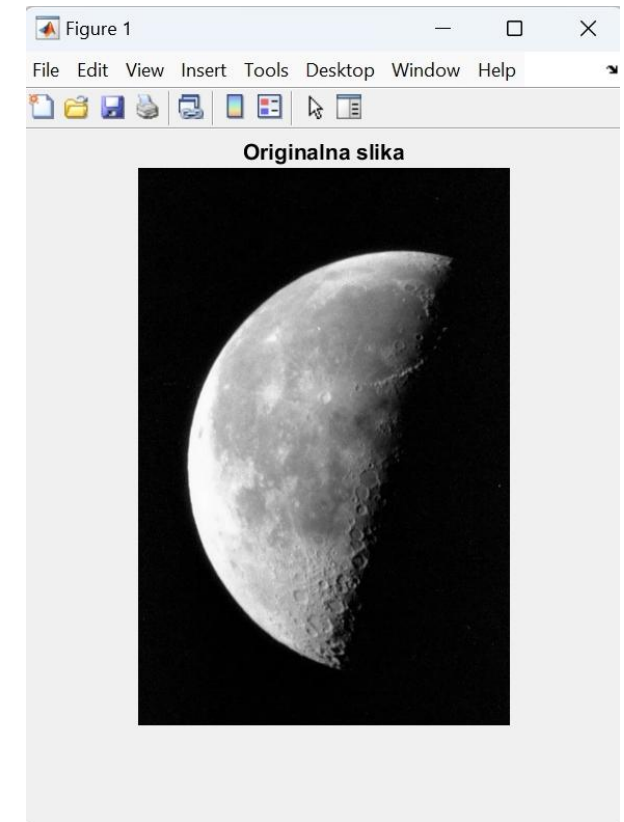
- Naložiti 2D sivinsko sliko ter podatke pretvoriti v ustrezen podatkovni tip.
- Izvesti analizo PCA (*Principal Component Analysis*).
- Na osnovi analize razložene variance v podatkih izbrati najpomembnejše glavne komponente.
- Rekonstruirati sliko zgolj z izbranimi komponentami ter ovrednotiti uspešnost.

# Demo: PCA na 2D sivinski sliki

(2/7)

## 1. Nalaganje 2D sivinske slike

```
% Nalaganje 2D sivinske slike velikosti 537x358, tipa uint8
img_orig = imread('moon.tif');
img = im2double(img_orig); % uint8 v tip double za PCA
% Prikaz
figure; imshow(img);
title('Originalna slika');
```





# Demo: PCA na 2D sivinski sliki

(3/7)

## 2. Analiza PCA

% koef - matrika glavnih komponent slike, PC so v stolpcih

% score - predstavitev slikovnih podatkov v prostoru glavnih komponent

% razlozena\_var - polje; koliko variance razloži posamezna PC

```
[koef, score, ~, ~, razlozena_var] = pca(img);
```

# Demo: PCA na 2D sivinski sliki

(4/7)

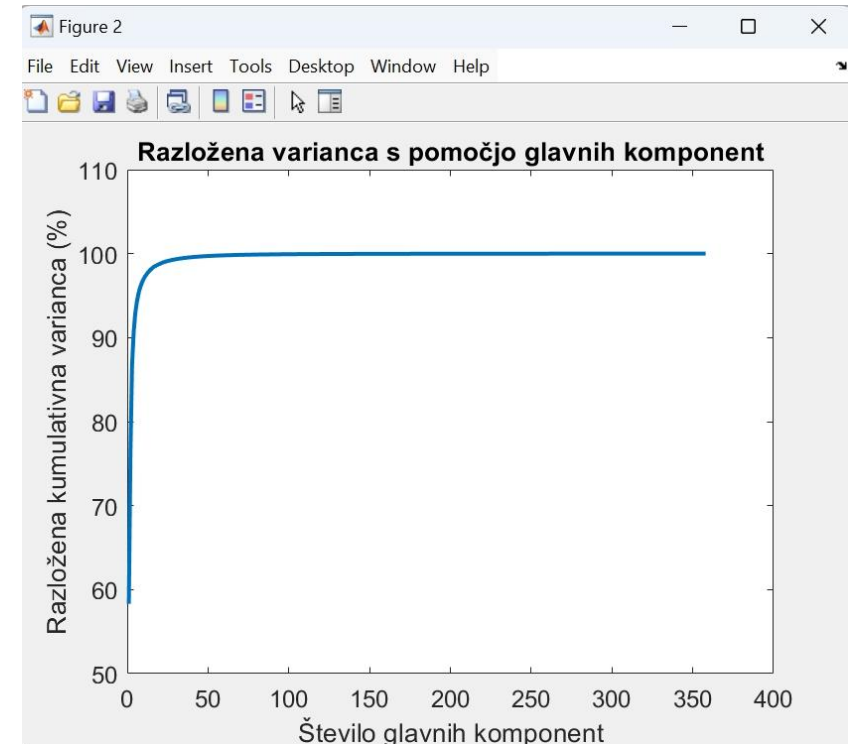
## 3. Analiza razložene variance

% Prikaz razložene variance za posamezno glavno komponento

```
figure; plot(cumsum(razlozena_var));
xlabel('Število glavnih komponent');
ylabel('Razložena kumulativna varianca (%)');
title(['Razložena varianca s pomočjo ' ...
      'glavnih komponent']);
```

% Izbiranje števila glavnih komponent

```
stKomponent = 20;
disp(sum(razlozena_var(1:stKomponent)));
% Opomba: 20 komponent razloži 98.78 % variance
```





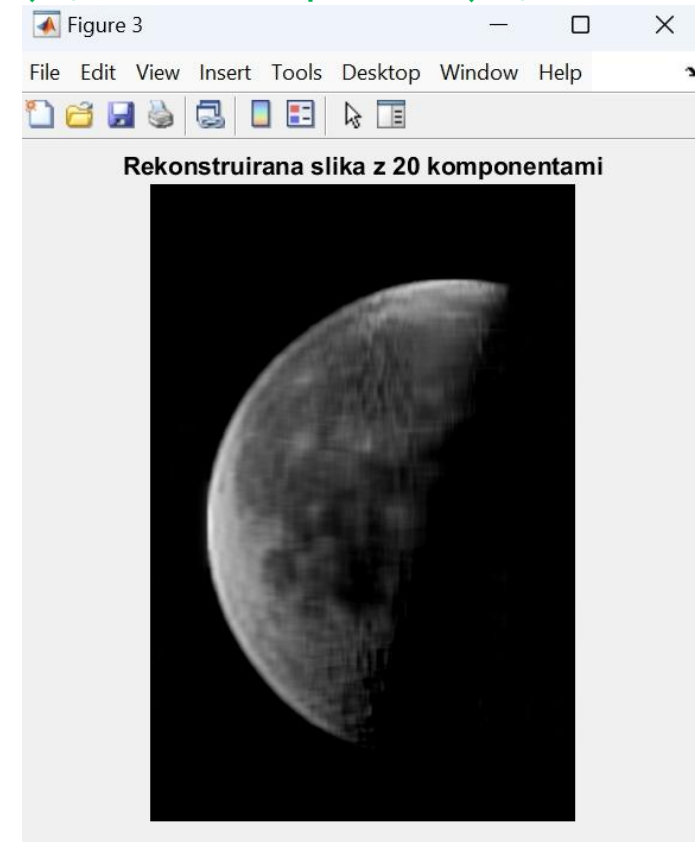
# Demo: PCA na 2D sivinski sliki

(5/7)

## 4. Rekonstruiranje slikovnih podatkov

```
% Rekonstruiranje slikovnih podatkov v manj dimenzionalnem prostoru
img_rekonstruirana = score(:, 1:stKomponent) * koef(:, 1:stKomponent)';
```

```
% Prikaz rekonstruirane slike
figure; imshow(img_rekonstruirana);
title(['Rekonstruirana slika z ', ...
       num2str(stKomponent), ' komponentami']);
```





# Demo: PCA na 2D sivinski sliki

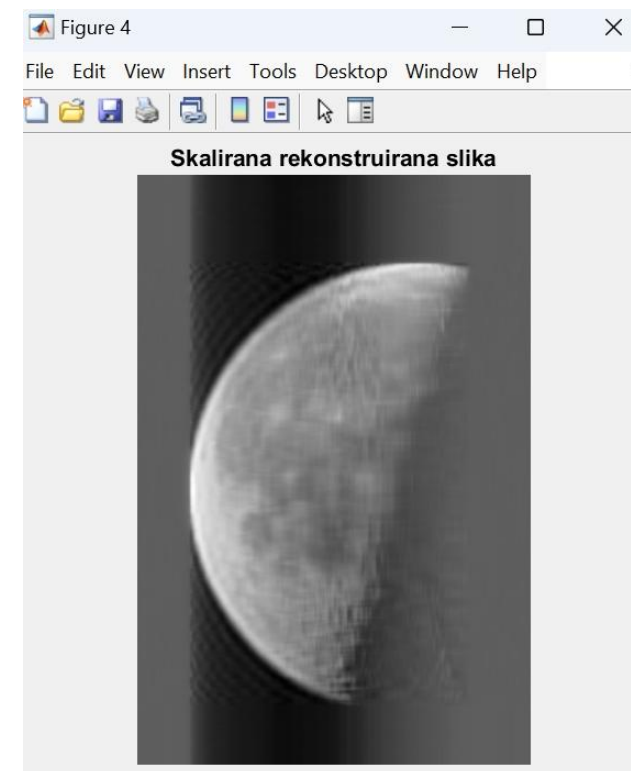
(6/7)

## 5. Ovrednotenje postopka

```
% Linearno skaliranje slikovnih podatkov in prikaz  
img_skalirana=uint8(255.*(img_rekonstruirana-min(img_rekonstruirana(:)))/ ...  
    (max(img_rekonstruirana(:))-min(img_rekonstruirana(:))));  
figure; imshow(img_skalirana);  
title('Skalirana rekonstruirana slika');
```

```
% Root Mean Squared Error (RMSE) med originalno  
% in skalirano sliko  
disp(rmse(double(img_skalirana), ...  
    double(img_orig),'all'))
```

% Opomba: RMSE = 63.8826





# Demo: PCA na 2D sivinski sliki

(7/7)

## 5. Ovrednotenje postopka

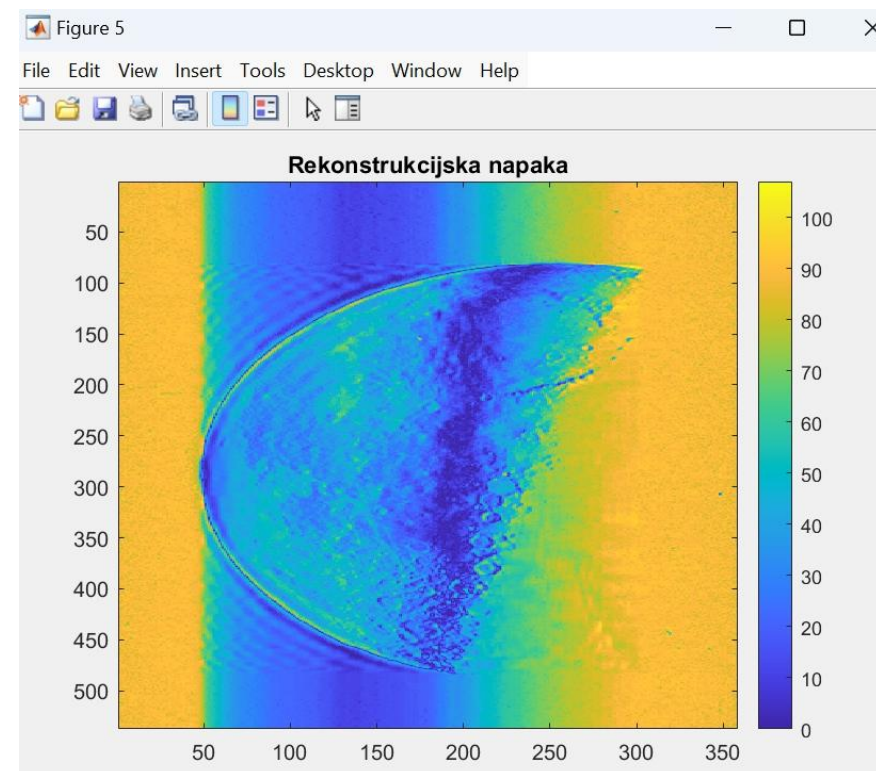
```
% Rekonstrukcijska napaka
```

```
Napaka = abs(double(img_orig)-double(img_skalirana));
```

```
figure; imagesc(Napaka);
```

```
title ('Rekonstrukcijska napaka');
```

```
colorbar;
```



Centralna tehniška knjižnica Univerze v Ljubljani  
*Central Technical Library at the University of Ljubljana*  
Trg republike 3, SI-1000 Ljubljana  
Slovenija / *Slovenia*

## Hvala za pozornost!

