

מבוא ל-MATLAB

חלק #1

אס"ט - אגודת הסטודנטים בטכניון

אסף שפניר

shpanier@gmail.com

עזרה והשראה: נעם ויסמן

- Nitai Hanani

תכני הקורס


תכני MATLAB:

- היכרות וסביבת עבודה
- ביטויים ומערכים ופעולות נומריות
- מבני מידע וארגון נתונים
- פעולות גרפיות בסיסיות
- תכנות וכתיבת פונקציות
- פעולות לוגיות ובקרת זרימה
- נושאים מתקדמים:
 - יעול וחיסכון בחישובים, Profiler
 - קריאת וכתיבת קבצים
 - הכרת toolboxes
 - גרפיקה מתקדמת
- ממשקי משתמש גרפיים (GUI)
- Simulink

יישומים הנדסיים:

- פתרון מערכות לינאריות
- גזירה ואינטגרציה
- טרנספורמציות לינאריות
- הצגת תוצאות ניסוייות
- פתרון משוואות דיפרנציאליות
- מתמטיקה עם המנוע הסימבולי
- סטטיסטיקה ושערוך פרמטרים
- ניצול מידע חזותי
- פתרון מערכות לא לינאריות
- בניית ממשקים עצמאיים
- מידול מערכות דינמיות

מבנה הקורס

- היקף הקורס הינו 10 שעות אשר יועברו בשני מפגשים ע"פ לוח הזמנים המצורף.
- כל נושא יילמד בשני שלבים:
 - הוראה פרונטלית, הכוללת הסברים ותרגילי מחשב אשר ייפתרו יחדיו.
לאורך המצגות, הסימון  מורה על תרגיל אשר יש ליישם במחשב.
- חוברות עזר: מבוא ל-MATLAB, מאת דורי פלג (ניתן להשיג דרך google)
- ← אין תחליף לתרגול והתנסות עצמאיים.
- ← מומלץ בחום להשתמש תדירות בממשק העזרה של התוכנה לביאור פעולות ו-syntax.

שיעור 1

יישומים:

- שימוש כמחשבון מספרים קומפלקסים
- פתרון מערכת משוואות לינאריות
- הפחתת רעש לבן ממדידות חוזרות
- גזירה ואינטגרציה
- אומדן השגיאה הנומרית
- טרנספורמציות לינאריות
- שליטה בתמונות

■ היכרות עם MATLAB

■ סביבת העבודה

■ ביטויים ומערכים

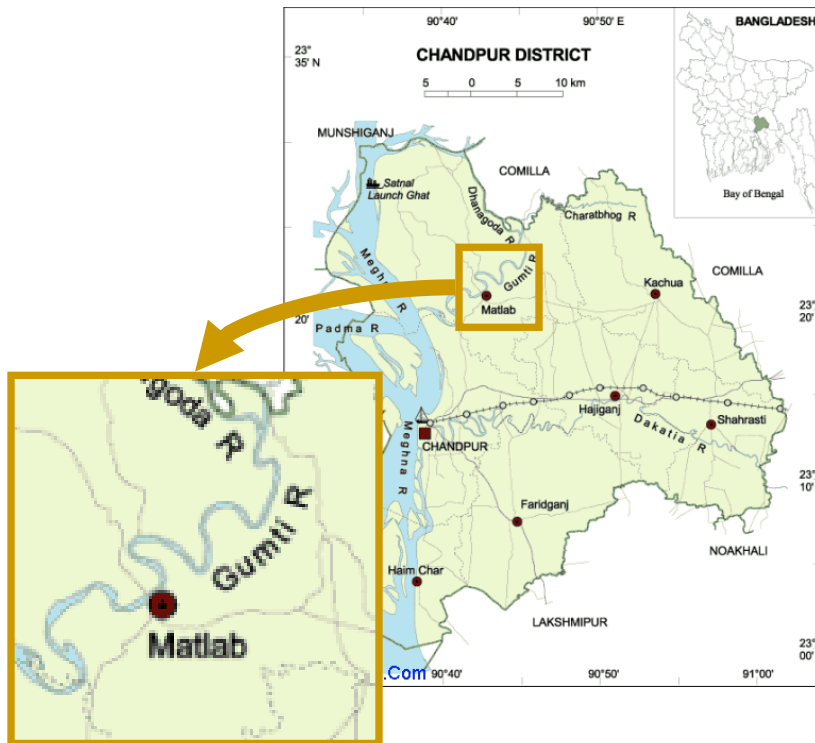
- סוגי משתנים וייצוגים מספריים
- כללים לקביעת שמות משתנים
- פעולות מתמטיות
- אתחול מטריצות
- פעולות בסיסיות על מטריצות

■ עבודה עם אינדקסים

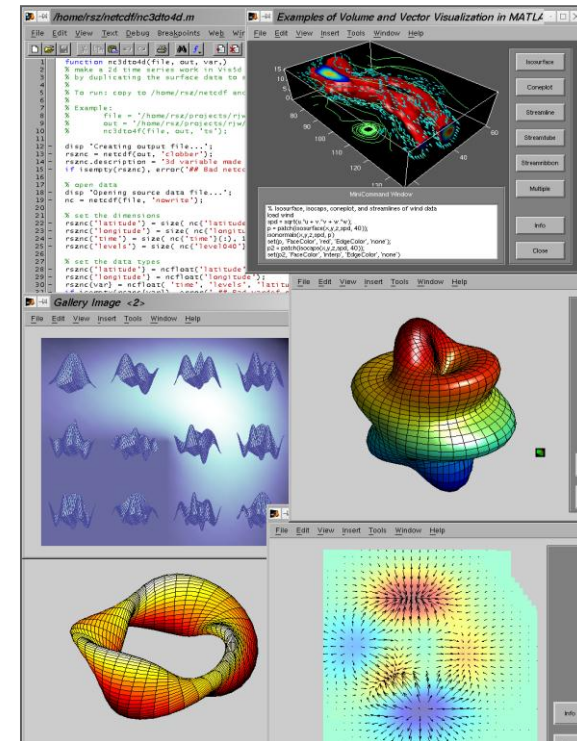
- קריאה לערכים
- מטריצות מרובות מימד
- מניפולציה של מטריצות
- שליטה בסדר מערכים: היפוך, שינוי מימד, דילול ודגימה

MATLAB – Wikipedia definitions

- Matlab – sub-district in Bangladesh



- MATLAB – MATrix LABoratory



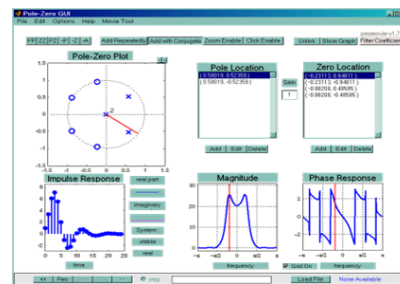
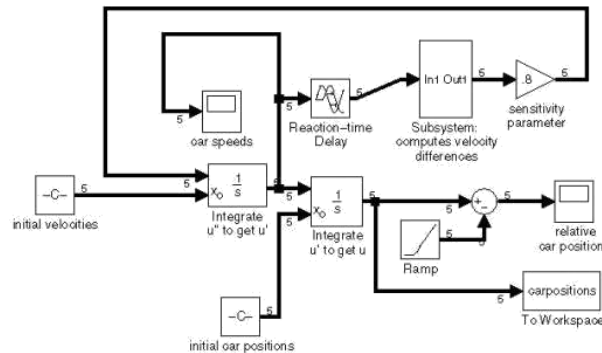
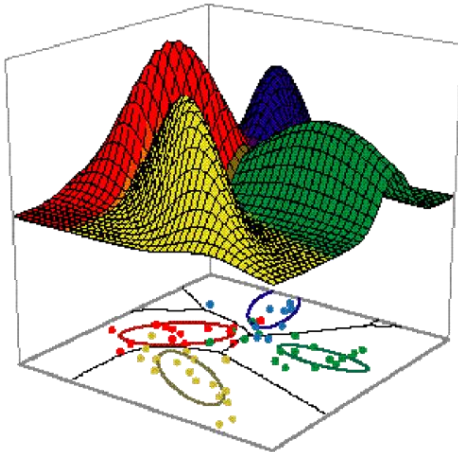
היכרות עם MATLAB - יכולות

■ מתמטיקה, עיבוד מידע ופיתוח אלגוריתמים

■ מידול וסימולציה

■ כלי פשוט לתצוגה גרפית מתקדמת

■ תכנות ופיתוח יישומים



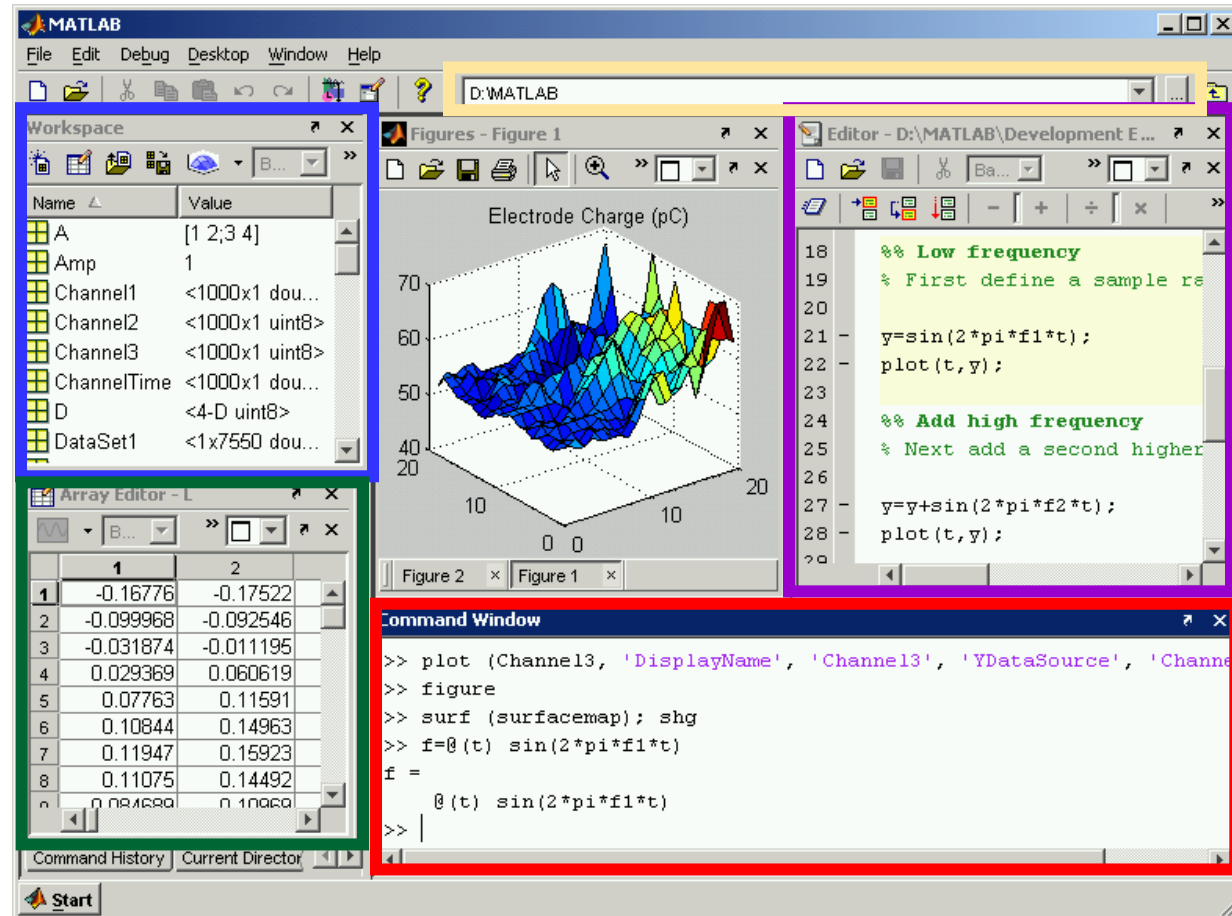
היכרות עם MATLAB - תכנות

MATLAB כשפה עילית :

- השפה מבוססת ברובה על שפת C ומודלי JAVA.
- שפת התכנות נוחה ואוטומטית ברובה, ואינה כרוכה בפעולות רגיסטרציה, הקצאות זכרון וכו'.
- קוד MATLAB אינו מצריך קומפילציה במובנה הרגיל.
- הקוד נקרא M-code וקבצי הקוד נקראים m.
- רוב הפעולות המתמטיות מתבססות על פונקציות built-in, אשר "צרובות" בבסיס התוכנה וביצוען מהיר ויעיל יותר.
- סוגי הקבצים בהם פוגשים לרוב: קבצי m, mat, ו-fig.

סביבת העבודה והפיתוח - כללי

- Command Window
 - Workspace
 - Current Directory
 - Command History
 - Array editor
 - Editor/Debugger
-
- Help Interface
 - >> help funcname
 - >> doc funcname
 - >> lookfor keyword



סביבת העבודה והפיתוח - פירוט

Workspace

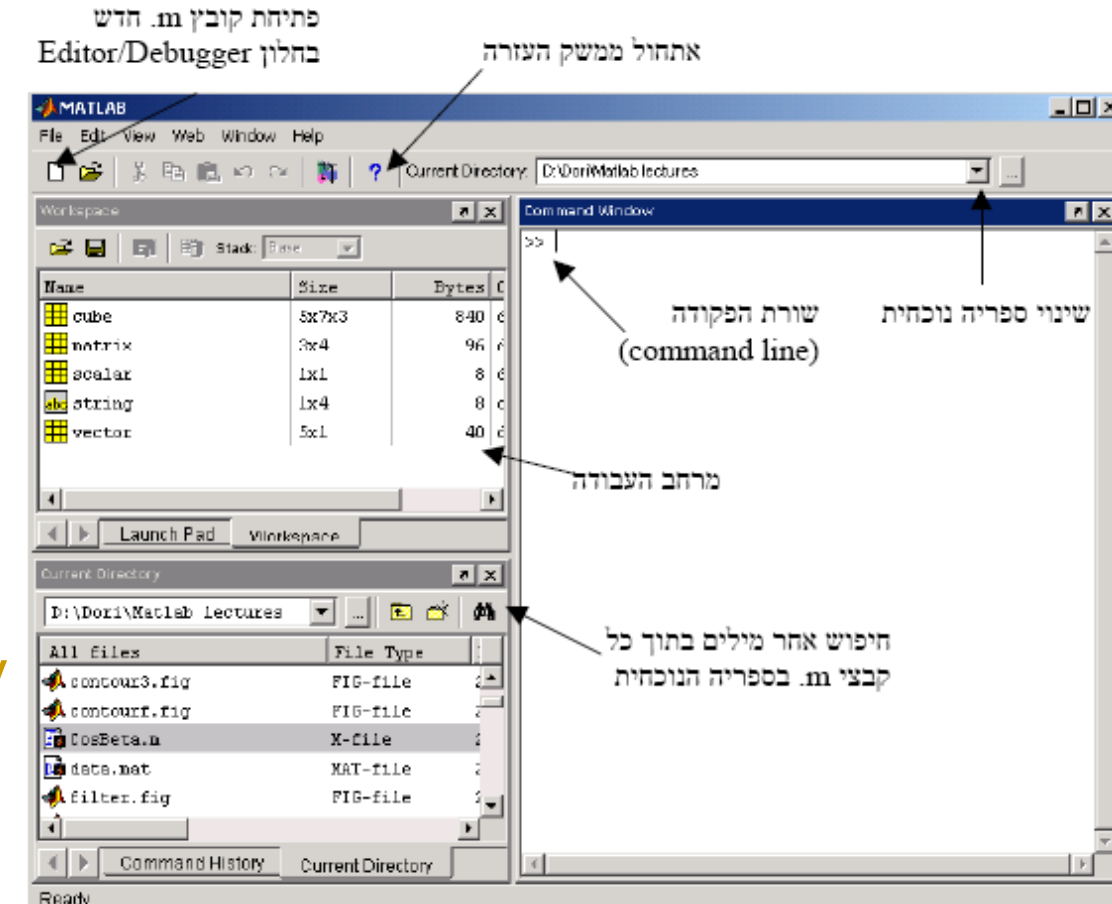
whos: List current variables and their size

clear: Clear variables and functions from memory

Current Directory

cd: Change current working directory

dir: List files in directory



Command Window

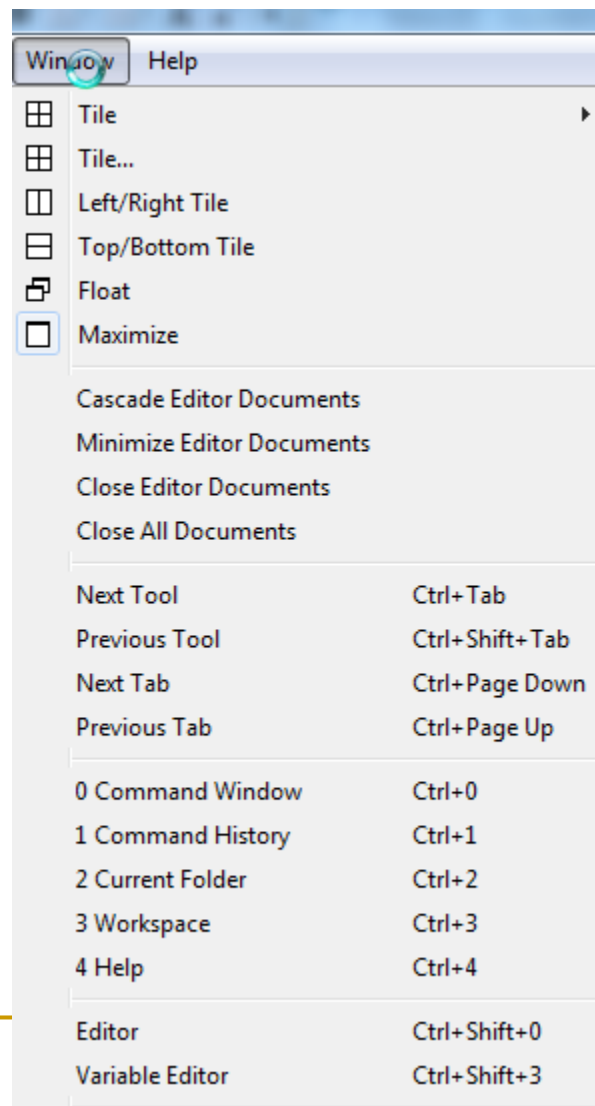
clc: clears command window

echo: Echo commands in M-

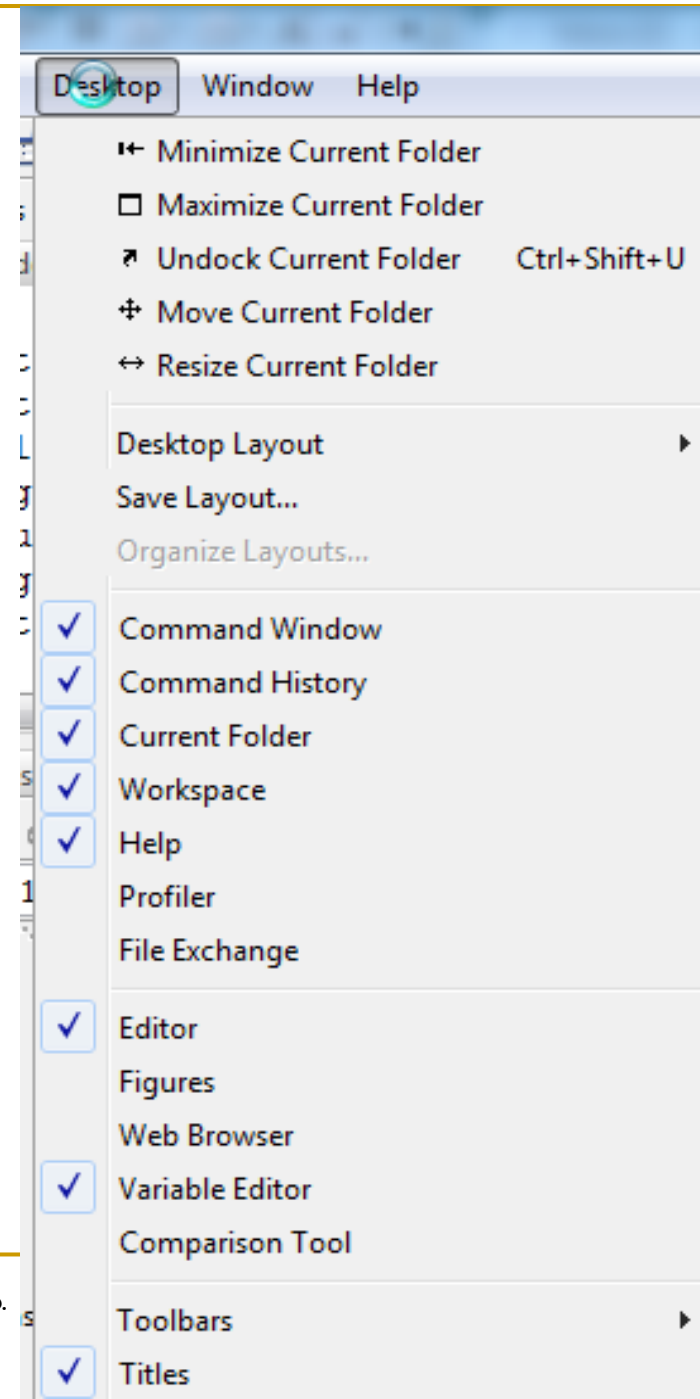
files **format:** Set output format

diary(filename): Saves all the commands you type in in a file

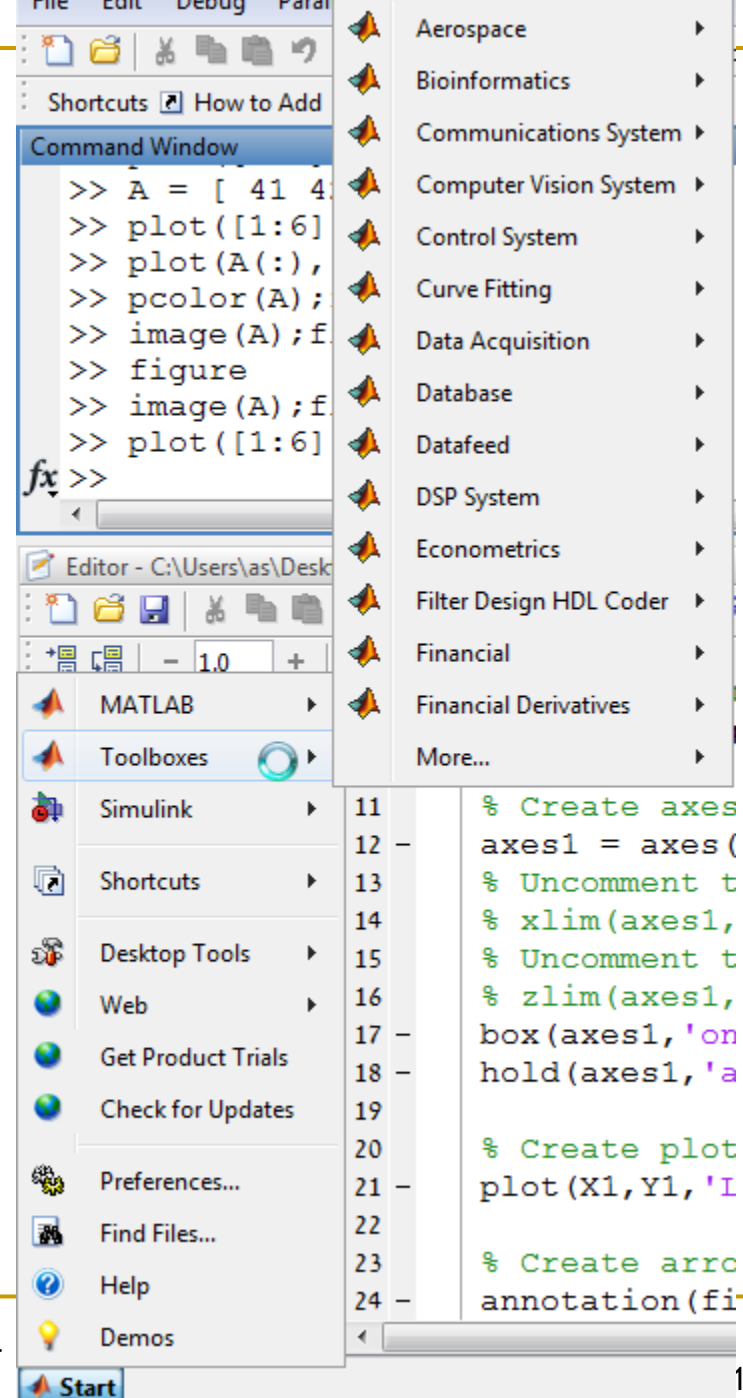
The Window menu



The Desktop menu



The Start button



סביבת העבודה והפיתוח – ממשק עזרה

חיפוש
מילת
מפתח

תצוגת
תוצאות

The screenshot shows the MATLAB Help window with the 'std' function selected in the Help Navigator. The main pane displays the 'MATLAB Function Reference' for 'std', including its syntax, definition, and description. The Help Navigator on the left lists various topics, and the main pane on the right provides detailed information about the 'std' function.

Help Navigator

Title	Section
std	MATLAB Functions
std	Financial Toolbox
std	Functions
std (timeseries)	MATLAB Functions
std2	Functions
friend std::ostrea...	C++ Utility Library Re
mwException(con...	C++ Utility Library Re
mwException& op...	C++ Utility Library Re
mapstd	Neural Network Toolb
errorbar	MATLAB Functions
Statistics	Functions -- By Cate
Time Series	Functions -- By Cate
Functions -- Alpha...	MATLAB Functions
var	MATLAB Functions
Release Note Rep...	Release Note Report
Multivariate Data	Matrices and Arrays
Logical Subscripting	Matrices and Arrays
Removing Outliers	Preparing Data for An
Functions for Calc...	Preparing Data for An
Example - Using ...	Preparing Data for An
Statistical Methods	Using Time-Series Ob
Time Plots	Using Time Series To
Using the Compon...	Programming with CC
C++ Shared Librar...	Libraries
bool operator>=(c...	C++ Utility Library Re
mwString Class	C++ Utility Library Re
mwException Class	C++ Utility Library Re
mwException(con...	C++ Utility Library Re
mwException(con...	C++ Utility Library Re

std (MATLAB Functions)

MATLAB Function Reference

std

Standard deviation

Syntax

```
s = std(X)
s = std(X, flag)
s = std(X, flag, dim)
```

Definition

There are two common textbook definitions for the standard deviation s of a data vector X .

$$(1) \quad s = \left(\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2 \right)^{\frac{1}{2}}$$
$$(2) \quad s = \left(\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2 \right)^{\frac{1}{2}}$$

where

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

and n is the number of elements in the sample. The two forms differ by a factor of $\sqrt{2}$.

Description

$s = \text{std}(X)$, where X is a vector, returns the standard deviation of the elements of X . If X is a matrix, $\text{std}(X)$ returns the standard deviation of each column. $\text{std}(X, flag)$ returns the standard deviation of each column of X using the unbiased estimator of the variance of the population from which the samples were drawn. $\text{std}(X, flag, dim)$ returns the standard deviation of each element along the dimension dim .

קישורים
לפונקציות
קרובות
המסמך

<http://www.mathworks.com>



Accelerating the pace of engineering and science

[United States](#) | [Contact Us](#) | [Store](#)

[Assaf spanier](#) | [My Account](#) | [Log Out](#)

[Products & Services](#) [Solutions](#) [Academia](#) [Support](#) [User Community](#) [Events](#)

[Company](#)

[Products & Services](#) > [MATLAB](#)

MATLAB

MAJOR UPDATE

The Language of Technical Computing

0.8147	0.0975	0.1253
0.9058	0.2785	0.9370
0.1270	0.5469	0.9573
0.9134	0.9575	0.4684
0.6324	0.9649	0.8003

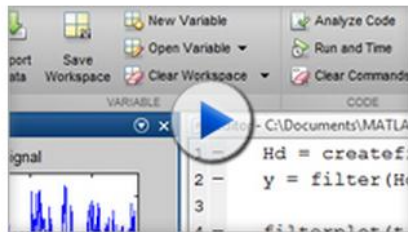
[Share](#)

[Overview](#)

[Videos & Examples](#)

[Webinars](#)

MATLAB® is a high-level language and interactive environment for numerical computation, visualization, and programming. Using MATLAB, you can analyze data, develop algorithms, and create models and applications. The language, tools, and built-in math functions enable you to explore multiple approaches and reach a solution faster than with spreadsheets or traditional programming languages, such as C/C++ or Java™.



[Product Overview](#) 2:05

You can use MATLAB for a range of applications, including signal processing and communications, image and video processing, control systems, test and measurement, computational finance, and computational biology. More than a million engineers and scientists in industry and academia use MATLAB, the language of technical computing.



TRY OR BUY

[Contact Sales](#)
[Product Trial](#)
[Pricing and Licensing](#)

R2012b

See the New MATLAB Desktop



Getting Started with MATLAB



MATLAB and Simulink
Student Version

[Documentation](#) [fx Functions](#) [Data Sheet](#) [Key Features](#)

<http://www.mathworks.com/matlabcentral/>



Search: MATLAB Central



[File Exchange](#) [Answers](#) [Newsgroup](#) [Link Exchange](#) [Blogs](#) [Trendy](#) [Cody](#) [Contest](#) [MathWorks.com](#)

Cody

Let the games begin

Now Available
R2012b
MATLAB&SIMULINK

» [Learn more](#)

Get MATLAB & Simulink Student Version

Students can purchase Student Version for just \$99.

[Learn more](#)

File Exchange

Recent Files

- magnify 2D *LY Cao*
- Basic Image processing including histogram without using hist function *Aditya*
- MIKI *Stephan Rave*
- Active contours driven by local Gaussian distribution fitting energy *li*
- a LogitBoost variant *sun peng*
- printf *Falko Schindler*

Cody

Recent Problems

- Using interp1 to interpolate NaN values in a vector *Ben*
- Plot a Figure in the command window, not in a seperate window *HITESH TRIVEDI*

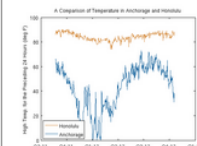
MATLAB Answers

Recent Questions

- griddedInterpolant bug *Nick Wong*
- Help with plotting shape functions? *Brian*
- Solve a system of symbolic variables *Ian Wood*
- How can i vectorize ?? *judy frost*
- Question of peak amplitude measuring *geo*
- Matrices and Line Intersection *Ping*

Trendy

Popular Plots



Temperatures in Anchorage and Honolulu (Comparison)
Ned Gulley

Blogs

Recent Updates



Loren on the Art of MATLAB
Learning to Love Regular Expressions 18 Oct 2012
[View archive](#)



Steve on Image Processing
MATLAB R2012b 16 Oct 2012
[View archive](#)

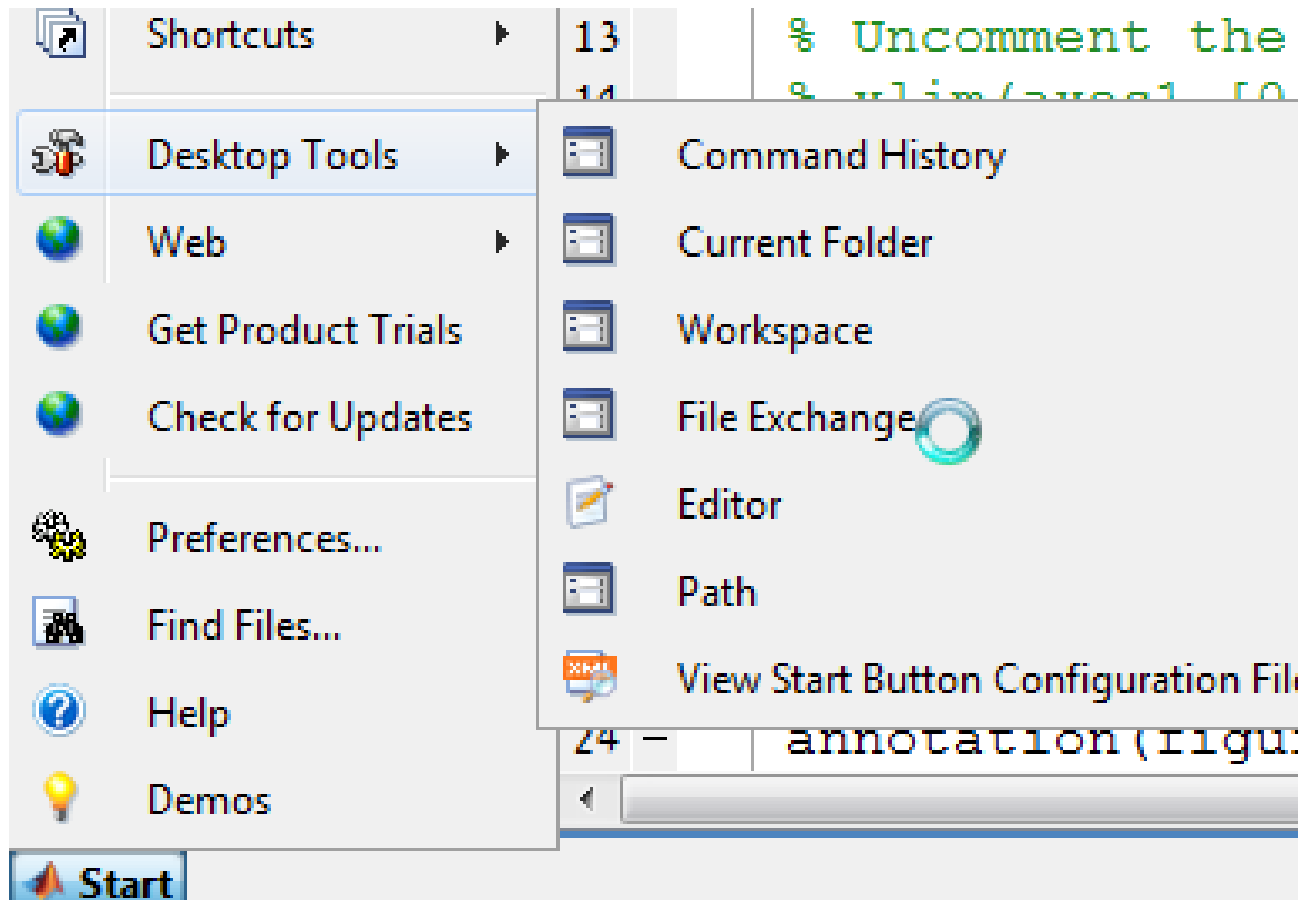


Doug's MATLAB Video Tutorials
Simple test scripts in MATLAB 16 Oct 2012
[View archive](#)



MATLAB Spoken Here
Have you heard of the MATLAB Contest? 15 Oct 2012
[View archive](#)

File Exchange



שימוש ב- Command Window כמחשבון

■ סדר פעולות מתמטיות: $+$ $-$ \rightarrow $*$ \backslash $/$ \rightarrow $^$

```
>> -5/(4.8+5.32)^2  
ans =  
-0.0488
```

■ נשתמש ב-cw לחישוב הזהויות הקומפלקסיות הבאות:

```
>> x = 3+4i      % x is stored in workspace and displayed in cw
```

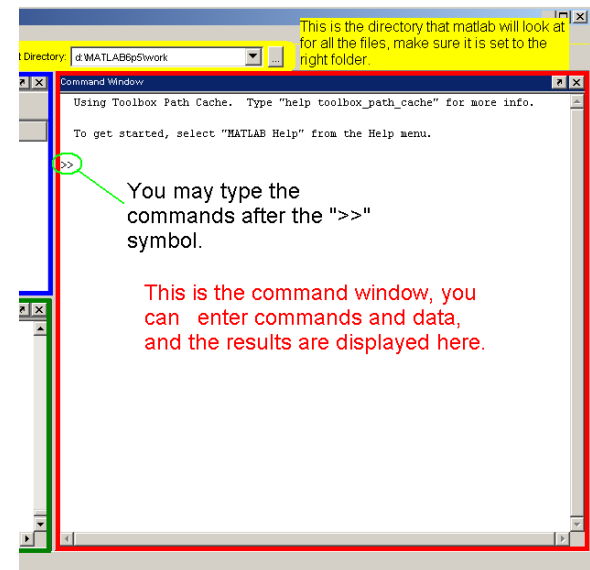
```
>> abs(x)        % Absolute value.  
ans = 5
```

```
>> angle(x)      % Phase angle (in radians).  
ans = 0.9273
```

```
>> conj(x)       % Complex conjugate.  
ans = 3-4i
```

```
>> imag(x)       % Complex imaginary part.  
ans = 4
```

```
>> real(x)       % Complex real part.  
ans = 3
```



אתחול מטריצות ידני - 1

הערות:

- כל העבודה בשלב זה תתבצע ב- command window (cw).
- שימו לב שאין צורך להגדיר את המשתנים וגודלם מראש.



נרצה להגדיר ידנית את מטריצת ה"קסם" הבאה:

נקרא למשתנה A ונאתחל אותו באופן הבא:

```
>>A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
```

התשובה המופיעה ב-workspace וב-cw:

A =

16	3	2	13
5	10	11	8
9	6	7	12
4	15	14	1

אתחול מטריצות ידני - 2

סימן " ; " מורה על סוף שורה.

סימן " ; " בסוף פקודה מונע את הצגת התשובה לפעולה ב-cw, אך לא את ביצוע הפקודה.
ניתן להציב סימני פסיק " , " בין איברי השורה של המטריצה במקום רווח (space).

נסו כעת לאתחל את המטריצה הבאה: 

```
>>A = [16 3 2 13 ; 5 10 11 8 ; 9 6 7 12 ; 4 15 14 1]
```


מה מתקבל?

צרו וקטור עמודה C ווקטור שורה R, המכילים את העמודה והשורה הראשונה של A, 
בהתאמה.

אתחול מטריצות ידני - 3

ניתן להגדיר מטריצות ע"י שמות משתנים קיימים או ביטויים:

```
>>a=2;  
>>A = [ a   R   exp(0)   j^2   3+5 ]  
A = 2   16   3   2   13   1   -1   8  
  
>>B=[A ; A]  
B = 2   16   3   2   13   1   0-1i   8  
    2   16   3   2   13   1   0-1i   8
```

איזה ביטוי מבין השניים יתקבל: ~~>>D=[R; C]~~ or >>D=[R C'] 

- שימו לב שכתובת ביטוי נומרי ב-cw ללא השמתו במשתנה מחזירה תשובה ומאחסנת אותה ב-workspace בתוך המשתנה .ans
- כעת נמחק מה-workspace את המשתנים שהוגדרו עד כה:

```
>> clear A B R C D  
>> clear all % shorter, but could be dangerous inside a script
```

אתחול מטריצות אוטומטי - 1

■ קבלת מימדי מטריצה (למקרה הכללי):

```
>>A = [ 1 2 3 ; 4 5 6];
```

```
>>[m,n] = size(A)
```

```
m = 2    n = 3
```

```
>> len = length(A)      % length(A) = max(size(A))
```

■ פונקציות להגדרת מטריצות:

>> A = zeros(m,n) מטריצת אפסים – יעיל להקצאת מקום בזכרון □

>> A = ones(m,n) מטריצת אחדים – □

>> d = [4 1 9 2.5]'; A = diag(d) diag – מטריצה אלכסונית – □

```
A = 4 0 0 0
    0 1 0 0
    0 0 9 0
    0 0 0 2.5
```

eye – מטריצת יחידה □

rand, randn – מטריצות של מספרים אקראיים □

אתחול מטריצות אוטומטי - 2

■ יצירת וקטור עוקב:

□ וקטור עולה במרווחי יחידה: $x = s : d : f \rightarrow x = [s \ s+d \ s+2d \ \dots \ s+(n-1)d]$

`>> x = 1:4` \rightarrow `x = 1 2 3 4`

□ ניתן לקבוע את גודל הקפיצות: `>> x = 0:10:100`

`x = 0 10 20 30 40 50 60 70 80 90 100`

□ כדי לקבוע וקטור יורד יש להגדיר קפיצות שליליות: `>> x = 10:-1:1`

■ נסו את הביטויים הבאים:

□ מה יהיה האיבר האחרון בוקטור: `>>0:3:10`

□ מה תהיה תוצאת הביטוי: `>>10:5`

□ הזינו בצורה הקצרה ביותר את המטריצה:

$$X = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 13 & 14 & 15 & 16 & 17 \\ 11 & 13 & 15 & 17 & 19 \\ 9 & 8 & 7 & 6 & 5 \\ 1 & 0.8 & 0.6 & 0.4 & 0.2 \end{bmatrix}$$

■ כאשר רוצים לשלוט במדויק בכמות האיברים רצוי להשתמש בפקודה `linspace(s,f,N)`.

□ משימה קשה באופן זמני: היעזרו בפקודה `round` וצרו וקטור עמודה אשר מתחיל בערך 1 ונגמר בערך 10 והינו בעל אורך (מספר איברים) אקראי בין 1 ל-100.

אתחול מטריצות אוטומטי - 3



- שימוש ב-linspace מקשה על השליטה בגודל המדויק של מרווח הסדרה, אשר הינו פרמטר מאוד חשוב בתחום עיבוד האותות והנומריקה.
- נהוג להגדיר וקטור כעמודה.

מחרוזות:

מחרוזת הינה מערך של תווים מסוג char

הגדרת פשוטה של מחרוזת:

```
>>myString = 'Mount Lu'
```

myString הוא שם מערך בגודל [1X8].

M	o	u	n	t		L	u
---	---	---	---	---	--	---	---

```
>>x=char(65) → x='A'
```

- המרה של מספר נומרי (double) ל-char:

- המרה של מחרוזת למחרוזת num2str("109")

- (שימושי לplots...)

- ניתן לאחסן מספר מחרוזות כשורות שונות של מטריצה – שימו לב, כל השורות חייבות להיות באורך זהה.

פעולות אריתמטיות על מטריצות - 1


- ניתן לבצע את כל הפעולות האלגבריות תחת שמירה על מימדים נכונים:
 - פעולת חיבור/חיסור -

```
>> A = [1 2 ; 3 4]
```

```
>> B = A + 1
```

```
>> B = A + [1 1 ; 1 1]
```

שתי הפעולות שקולות. חיבור מטריצות במימדים שונים מותר רק כשאחת מהן היא סקלר.

משימה: הוסיפו את וקטור השורה $[2+i \ 1/3]$ לשורה הראשונה במטריצה A. 



אופרטור שחלוף על מטריצות:  אם A ממטית - $A' = \text{transpose}(A)$

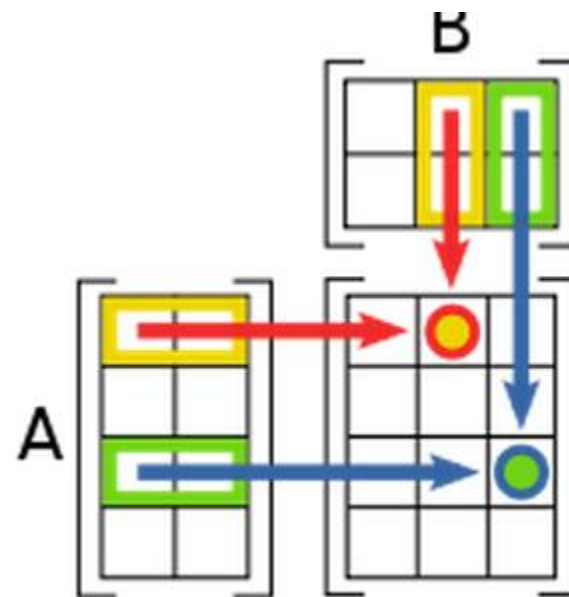
אם A מרוכבת - $A.' = \text{transpose}(A)$

מכפלת מטריצות בסקלר: 

```
>> B = 2*A      →       $B = \begin{bmatrix} 2 & 4 \\ 6 & 8 \end{bmatrix}$ 
```


נזכור כי עבור מכפלה בסקלר - $\alpha \Leftrightarrow \alpha \underline{\underline{I}}$

כפל מטריצות



פעולות אריתמטיות על מטריצות - 2

□ כפל מטריצות -

 >> A = [1 2; 3 4; 5 6]
>> B = [7 8; 9 10; 11 12]
>> C = A * B'


$$C = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \cdot \begin{bmatrix} 7 & 9 & 11 \\ 8 & 10 & 12 \end{bmatrix} = \begin{bmatrix} 23 & 29 & 35 \\ 53 & 67 & 81 \\ 83 & 105 & 127 \end{bmatrix}$$

□ כפל מטריצות / וקטורים -

>> a = [1 2 3]'; b = [10 20 30]';
>> c = a' * b

מכפלה פנימית <a,b> : 

$$C = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \cdot \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix} = 140$$

מכפלה חיצונית:  משימה- בעזרת הוקטור x=1:10 צרו מטריצה שהינה לוח הכפל.

□ היפוך מטריצה ריבועית -

>> A = [1 2; 3 4]; inv(A) or A^(-1) → $-\frac{1}{2} \begin{bmatrix} 4 & -2 \\ -3 & 1 \end{bmatrix}$ פונקצית inv או העלאת חזקה -

חלוקה ימנית - /

חלוקה שמאלית - \

חשבו את הביטוי $B^{-1}A$ בעזרת אופרטור החלוקה-  כאשר B = [5 1; 6 4]

פעולות אריתמטיות על מטריצות - 3



■ סינגולריות של מטריצות

□ מציאת דרגה של מטריצה – `rank()`

□ מציאת דטרמיננטה – `det()`

□ מספר המצב – באופן מעשי, מטריצות יכולות להיות קרובות לסינגולריות והפיכתן גוררת שגיאות נומריות גדולות. נהוג לאמוד את "חולי" המטריצות ע"י – `cond()`:
`cond(A)>>1` ← מטריצה חולה.

```
>> A = magic(4)
```

```
>> rank(A) → 3
```

הדרגה קטנה מהמימד -

```
>> det(A) → 0
```

ואכן הדטרמיננטה מתאפסת -

```
>> inv(A) → returns an answer with a warning
```

האם צריך לסמוך על התשובה שחזרה מהפונקציה?

אלגוריתם `inv` לא אמור לפעול על מטריצות סינגולריות. אם נבדוק: `>>cond(A) → ~1e17`

פעולות אריתמטיות על מטריצות - 4

■ וקטורים עצמיים וערכים עצמיים

□ למטריצה ריבועית A קיימים ו"ע וע"ע כדרגת המטריצה – $A \cdot v = \lambda \cdot v$

□ לכסון המטריצה מתבצע באמצעות – $\Lambda = V^{-1} \cdot A \cdot V$

□ ישנן מספר פונקציות שמבצעות פירוק ע"ע, הנפוצה הינה – $\text{eig}()$

```
>>A = [1 2 3 ; 0 4 5 ; 0 0 6];
```

```
>>[V,D] = eig(A)
```

מטריצה
שעמודותיה הן
ו"ע מנורמלים

$V =$

1	0.55	0.51
0	0.83	0.79
0	0	0.31

$D =$

1	0	0
0	4	0
0	0	6

מטריצה
שאלכסונה מכיל
ע"ע של A

קבלו חזרה את A מהמטריצות V ו- D . 

פתרון: 

```
>> V*D*inv(V)
```

```
ans = 1 2 3  
      0 4 5  
      0 0 6
```

פעולות אריתמטיות על מטריצות - 6

■ הכרת פונקציות מטריציות נוספות

□ קבלת מטריצה מדורגת – `rref()`

□ עכבת מטריצה – `trace()`

□ פירוק מטריצה למכפלת משולשת עליונה ומשולשת תחתונה ($A=L*U$) – `lu()`

□ היפוך מטריצה לא ריבועית – `pinv()`

$$4x_1 + 2x_2 + 25x_3 = 12$$

$$2x_1 + 6x_2 - 11x_3 = -19$$

יישום – פתרון מערכת לינארית עם דרגת חופש:

נסו לפתור ללא שימוש בפונקציות מובנות

פתרון הבעיה:



פעולות אריתמטיות על מטריצות - 7

■ פעולות חד ממדיות נפוצות

□ סכום – `sum(X,dim)`

אם `dim` לא נתון, הפונקציה פועלת על המימד הראשון שאינו באורך 1.

□ סכום מצטבר – `cumsum()`

1	2	3	4	5	6	7
1	3	6	10	15	21	28

□ ממוצע – `mean()`

□ כפל – `prod()`

מה יהיה מימד התוצאה של `prod([1 2 ; 3 4])`?

■ יישום – הפחתת רעש לבן ממדידות חוזרות 

- נדגים את יעילות השימוש במטריצה לשם טיפול במדידות חוזרות בעזרת אות א.ק.ג רועש.
- בהמשך הלימודים תגלו שהשפעת רעש לבן עם ממוצע אפס יורדת במיצוע מדידות חוזרות בעלות תבנית קבועה.

■ העתיקו את קובץ `ecg.mat` לתיקיית העבודה הנוכחית (ניתן לגלות ע"י הפקודה `cd`)

■ העלו את קובץ ה-`mat`. ע"י הפקודה `>>load ecg.mat`

■ `ecgn` מופיעה ב-`workspace` כמערך נומרי בעל 1000 שורות ו-293 עמודות.

פעולות על מערכים – אופרטור הנקודה "." (1)

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \quad B = \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \end{bmatrix}$$

פעולה איבר איבר משנה את
המשמעות האלגברית
ממטריצה למערך מספרים.

$$C = A.*B = \begin{bmatrix} a_{11} \cdot b_{11} & a_{12} \cdot b_{12} & a_{13} \cdot b_{13} & a_{14} \cdot b_{14} \\ a_{21} \cdot b_{21} & a_{22} \cdot b_{22} & a_{23} \cdot b_{23} & a_{24} \cdot b_{24} \\ a_{31} \cdot b_{31} & a_{32} \cdot b_{32} & a_{33} \cdot b_{33} & a_{34} \cdot b_{34} \end{bmatrix}$$

כפל,

$$C = A./B = \begin{bmatrix} a_{11}/b_{11} & a_{12}/b_{12} & a_{13}/b_{13} & a_{14}/b_{14} \\ a_{21}/b_{21} & a_{22}/b_{22} & a_{23}/b_{23} & a_{24}/b_{24} \\ a_{31}/b_{31} & a_{32}/b_{32} & a_{33}/b_{33} & a_{34}/b_{34} \end{bmatrix}$$

חילוק,

$$C = A.^B = \begin{bmatrix} a_{11}^{b_{11}} & a_{12}^{b_{12}} & a_{13}^{b_{13}} & a_{14}^{b_{14}} \\ a_{21}^{b_{21}} & a_{22}^{b_{22}} & a_{23}^{b_{23}} & a_{24}^{b_{24}} \\ a_{31}^{b_{31}} & a_{32}^{b_{32}} & a_{33}^{b_{33}} & a_{34}^{b_{34}} \end{bmatrix}$$

חזקה,

פעולות על מערכים – אופרטור הנקודה (2)

- מיישם פעולה אריתמטית על כל איבר במערך בנפרד.
- ממוקם לפני אופרטור הפעולה המתמטית.

```
>>A = [1 2 ; 3 4]
```

```
>>A^2
```

```
ans =  7 10  
      15 22
```

```
>>A.^2
```

```
ans =  1 4  
      9 16
```

כאשר משתמשים בפעולה איבר-איבר, חשוב להקפיד על סדר מערכים זהה.

```
>>t = [0:1e-3:1]; % time base
```

```
>>a = 5; % constant scalar
```

```
>>x = [a*t^2]; % won't work. Where shall we place the  
dot?
```

```
>>x = [a*t.^2];
```

משימה: עבור וקטור t שהוגדר קודם, צרו את סדרת מספרים שמבטאת את $\tan(2\pi t)$ 

```
>>x = sin(2*pi*t) ./cos(2*pi*t); % use dot operator
```

עבודה עם איברי מערכים - 1

■ אינדקסים של איברי מטריצה

A =

		Columns (n)				
		1	2	3	4	5
Rows (m)	1	4 ¹	10 ⁶	1 ¹¹	6 ¹⁶	2 ²¹
	2	8 ²	1.2 ⁷	9 ¹²	4 ¹⁷	25 ²²
	3	7.2 ³	5 ⁸	7 ¹³	1 ¹⁸	11 ²³
	4	0 ⁴	0.5 ⁹	4 ¹⁴	5 ¹⁹	56 ²⁴
	5	23 ⁵	83 ¹⁰	13 ¹⁵	0 ²⁰	10 ²⁵

A (2,4)

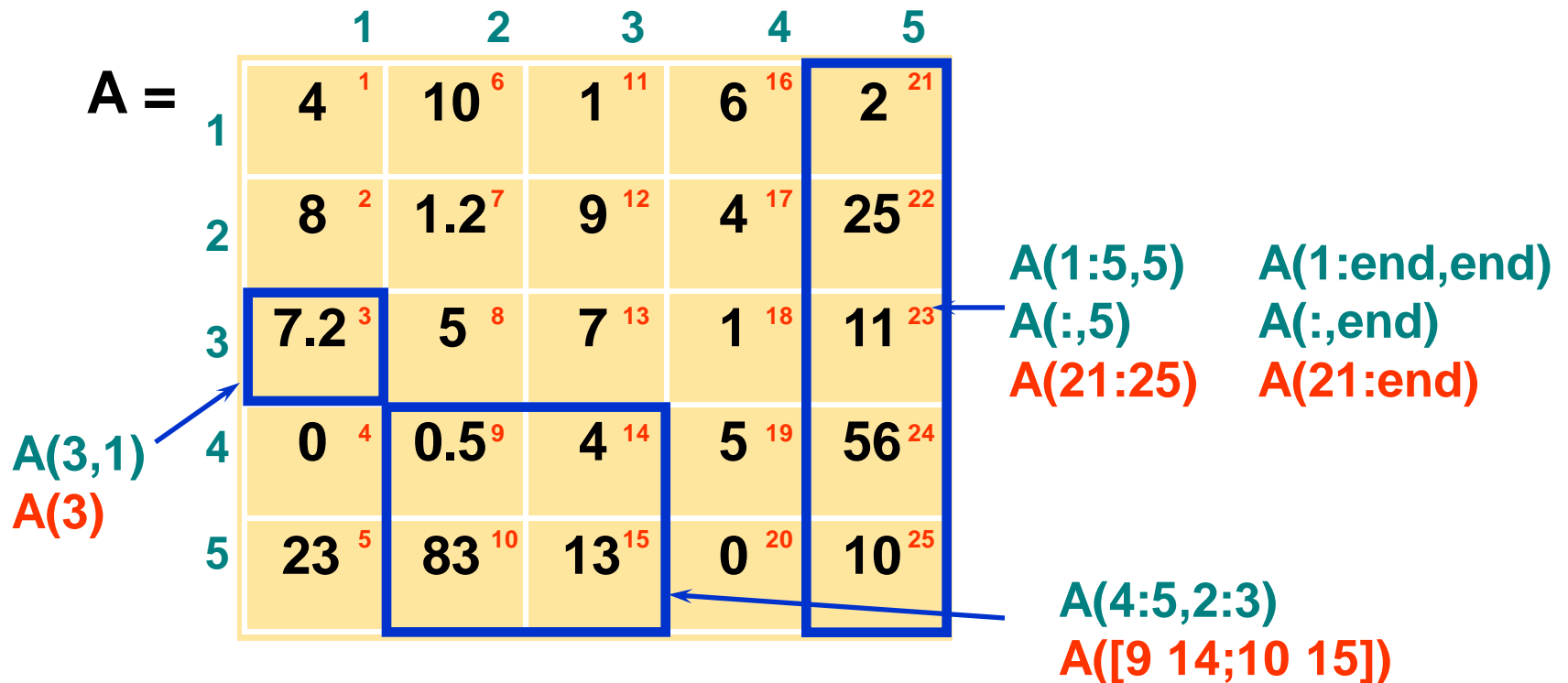
A (17)

Rectangular Matrix:
Scalar: 1-by-1 array
Vector: m-by-1 array
 1-by-n array
Matrix: m-by-n array

!!! שימו לב שהאינדקס של האיבר הראשון הוא 1 ולא 0 !!!

עבודה עם איברי מערכים - 2

■ אינדקסים של איזורים במטריצה



- אופרטור העמודה ":" פורש את כל האינדקסים הקיימים במימד הנתון.
- אופרטור end בתור אינדקס נותן את הערך המקסימלי במימד הנתון.

עבודה עם איברי מערכים - 3

■ אופרטור העמודה " : "

□ קריאה לשורה/עמודה שלמה – למשל, ניתן לקבל כל אות א.ק.ג בודד מהמטריצה:

```
second_ecg = ecg(2, :);           % second row, all columns
```

□ פורש כל מערך על פי סדר המימדים שלו לוקטור עמודה

■ מניעת שגיאות – כאשר אין וודאות בכיוון וקטורים הניתנים ככניסה לפונקציה.

למשל, אם רוצים לצרף ווקטור בתור עמודה נוספת למטריצה קיימת באותו אורך:

```
total_mat = [total_mat user_vector(:)];
```

■ מטריצה ריקה [] – סוגריים ריקים נותנים מערך בגודל 0x0 הקיים ב-workspace:

```
A = [ ]; % empty matrix
```

משימות:

נשתמש במטריצת לוח הכפל שיצרנו ע"י: `>>kefel=[1:10]'*[1:10];`

השמו את העמודה הראשונה של kefel בוקטור הנקרא c1. 

השמו את האיברים האי-זוגיים של השורה החמישית של kefel בוקטור הנקרא r2. 

אפסו ערכי kefel בעמודות הזוגיות. 

מחקו את השורה והעמודה האחרונות של kefel, כלומר – שנו אותה לגודל 9x9. 

עבודה עם איברי מערכים - 4

■ דצימציה – לקיחת כל איבר שני שביצענו היא למעשה דגימה/דצימציה בפקטור 2.

```
>>x_sampled = x(1:2:end)
```

💻 דילול – הפעולה ההפוכה לדצימציה הינה ריווח הסדרה הנתונה. **רעיונות מימוש?**

```
>>>>x_dilut(1:2:2*length(x)) = x
```

```
>>x=0.1:0.1:1;
```

■ היפוך סדר מערך -

```
>>x_rev = x(end:-1:1) → 1 0.9 0.8 0.7 0.6 0.5 0.4 0.3 0.2 0.1
```

אופרטורים לוגיים

```
>> r = 1:5
```

```
r =
```

```
1      2      3      4      5
```

```
>> ab = logical([1 0 0 0 1]);
```

```
>> r(ab)
```

עבודה עם איברי מערכים - 5

■ אופרטורים לוגיים

□ בקרת זרימה בתוכנית (נידון בנמשך)

□ קריאה לאיברי מערך:

■ דרך תנאים על איברי המערך

`==` equal to

`>` greater than

`<` less than

`>=` greater or equal

`<=` less or equal

`~` not

`&` and `(&&)`

`|` or `(||)`

`is*()`, etc. . . .

`all()`, `any()`

`find`

```
>>r = rand(100,1);
```

```
>> my_cond = [r<0.75 & r>0.25];
```

```
>>r_top = r(my_cond);
```

■ דרך תנאים על האינדקסים של המערך

```
>>ind = 1:length(r);
```

```
>> my_cond = [ind/4==round(ind/4) |  
ind/3==round(ind/3) ]
```

```
>>r( my_cond )
```

■ דרך תנאים על מערך מקביל אחר

```
>>t = [0:0.1:50]';
```

```
>>cos_t = sin(2*pi/50*t) .* (t<=max(t)/2);
```

■ פונקצית `find` - מחזירה את האינדקסים עצמם במקום מערך

עבודה עם איברי מערכים - 5

■ אופרטורים לוגיים

`==` equal to

`>` greater than

`<` less than

`>=` greater or equal

`<=` less or equal

`~` not

`&` and `(&&)`

`|` or `(||)`

`is*()`, etc. . . .

`all()`, `any()`

`find`

□ בקרת זרימה בתוכנית (נידון בנמשך)

□ קריאה לאיברי מערך:

■ דרך תנאים על איברי המערך

```
>>r = rand(100,1);
```

```
>>r_top = r(r<0.75 & r>0.25);
```

■ דרך תנאים על האינדקסים של המערך

```
>>ind = 1:length(r);
```

```
>>r(ind/4==round(ind/4) | ind/3==round(ind/3))
```

■ דרך תנאים על מערך מקביל אחר

```
>>t = [0:0.1:50]';
```

```
>>cos_t = sin(2*pi/50*t) .* (t<=max(t)/2);
```

■ פונקצית `find` - מחזירה את האינדקסים עצמם במקום מערך `.logicals`

עבודה עם איברי מערכים - 6

משימות:



התמרת אינדקסים



צרו את המטריצה הבאה:

```
>>A = [5 0 2 3 ; 0 5 8 5  
; 5 3 5 0 ; 9 5 1 1]
```

5	0	2	3
0	5	8	5
5	3	5	0
9	5	1	1

1. שנו את המטריצה כך שכל איבר שאינו שווה ל-5, יקבל את הערך NaN.

2. שנו את איברי האלכסון של A, לערך Inf, ללא שימוש בפקודה diag.

היעזרו במטריצות האינדקסים M ו-N, אותן תיצרו ע"י: `>>[N,M] = meshgrid(1:4)`

עבודה עם איברי מערכים – 7: מטריצות מרובות מימד

■ אתחול:

□ הפניית אינדקסים –

```
>>A(:, :, 1) = pascal(4);  
>>A(:, :, 2) = magic(4);  
>>A(:, :, 4) = diag(ones(1,4));
```

□ שרשור – `cat()`

```
>>A = cat(dim,A,B);
```

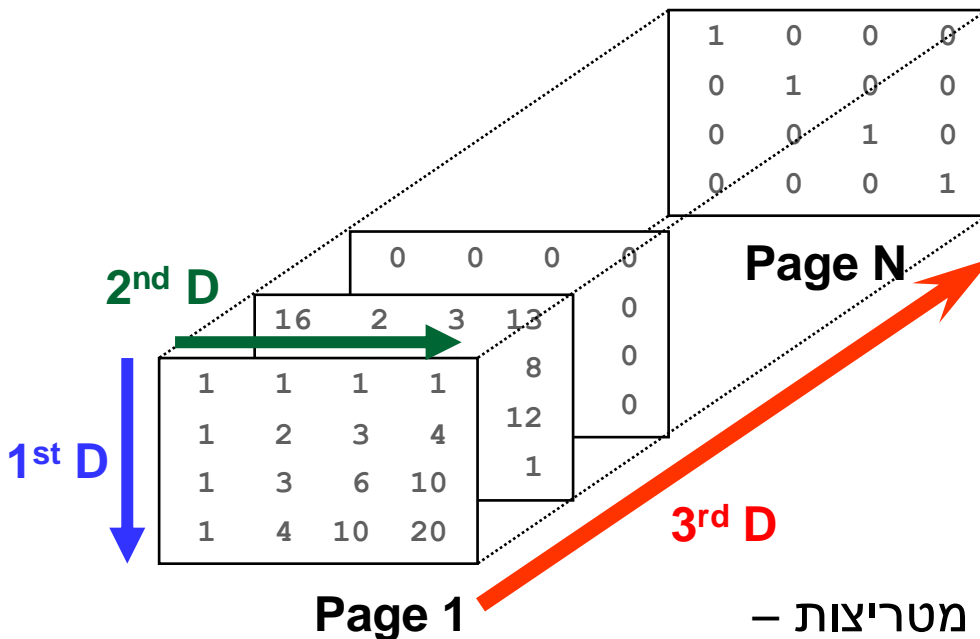
□ אתחול רגיל באמצעות פונקציות יצירת מטריצות –

```
>> ones(2,2,2,2) % 4D matrix
```

□ `reshape`, `meshgrid`, `repmat` - נלמד אותן בהמשך.

■ התמרת אינדקסים:

□ כזכור, אינדקסים מיוצגים באופן רציף, ע"פ סדר המימדים הקיים, או באופן פרטני עבור כל מימד. ניתן להיעזר בפונקציית `sub2ind` כדי לבצע את ההתמרה (על אף פשטותה).



The Cat function

```
>> A1 = pascal(4)
```

```
A1 =
```

1	1	1	1
1	2	3	4
1	3	6	10
1	4	10	20

```
>> A2 = magic(4)
```

```
A2 =
```

16	2	3	13
5	11	10	8
9	7	6	12
4	14	15	1

```
>> W = cat(3, A1,A2)
```

```
W(:, :, 1) =
```

1	1	1	1
1	2	3	4
1	3	6	10
1	4	10	20

```
W(:, :, 2) =
```

16	2	3	13
5	11	10	8
9	7	6	12
4	14	15	1

The ind2sub function

```
>> A = magic(4)
```

```
A =
```

16	2	3	13
5	11	10	8
9	7	6	12
4	14	15	1

```
>> sub2ind(size(A), 2, 2)
```

```
ans =
```

```
6
```

```
>> [a,b] = ind2sub(size(A), [6])
```

```
a =
```

```
2
```

```
b =
```

```
2
```

The Repmat function

- creates a large matrix consisting of an m-by-n tiling of copies of A.

```
>> A = [1 2 3]

A =

     1     2     3

>> repmat(A, 3 , 1)

ans =

     1     2     3
     1     2     3
     1     2     3

fx >>
```

The reshape function

- returns the m-by-n matrix B whose elements are taken column-wise from A. An error results if A does not have $m*n$ elements

```
>> A = [ 1 2 3 ; 4 5 6]
```

```
A =
```

1	2	3
4	5	6

```
>> reshape(A, 3 , 2)
```

```
ans =
```

1	5
4	3
2	6

The permute function.

Rearranges the dimensions of A so that they are in the order specified by the vector order. B has the same values of A but the order of the subscripts needed to access any particular element is rearranged as specified by order. All the elements of order must be unique.

```
>> A = [ 1 2 3 ; 4 5 6]
```

```
A =
```

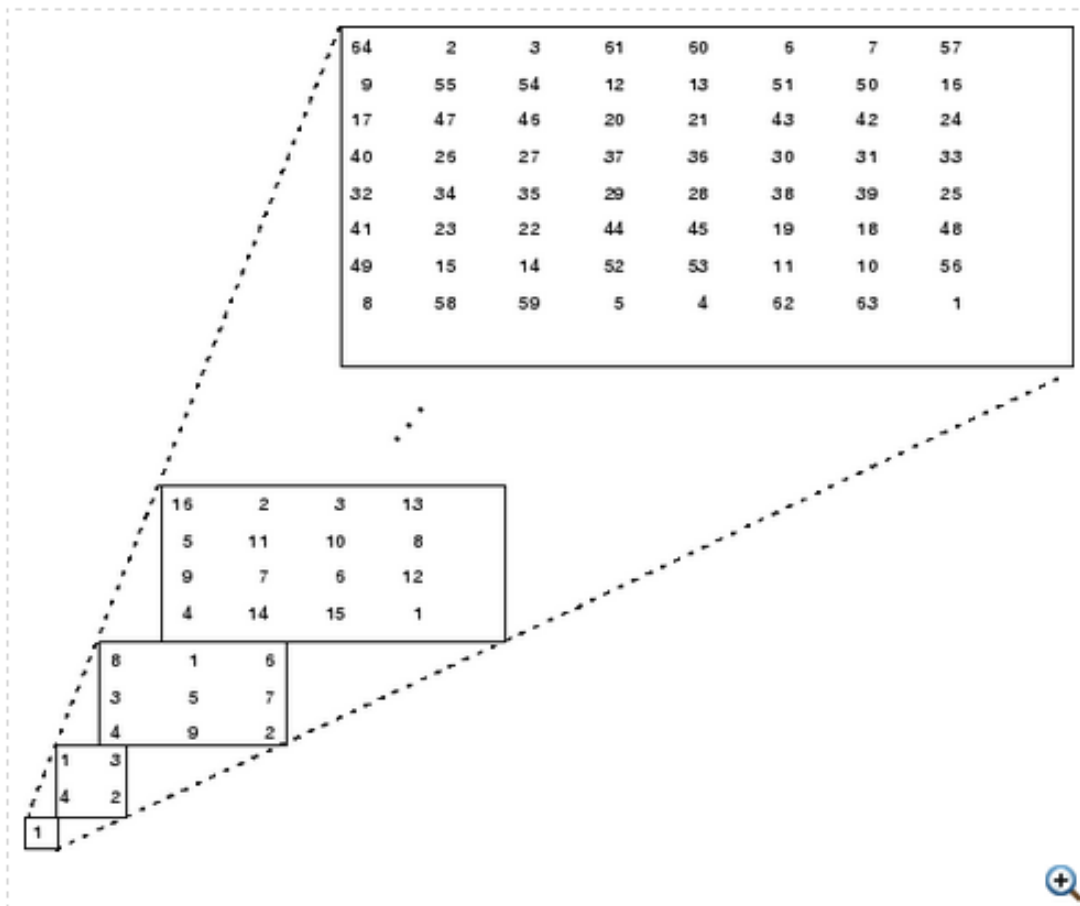
1	2	3
4	5	6

```
>> permute(A , [2 1])
```

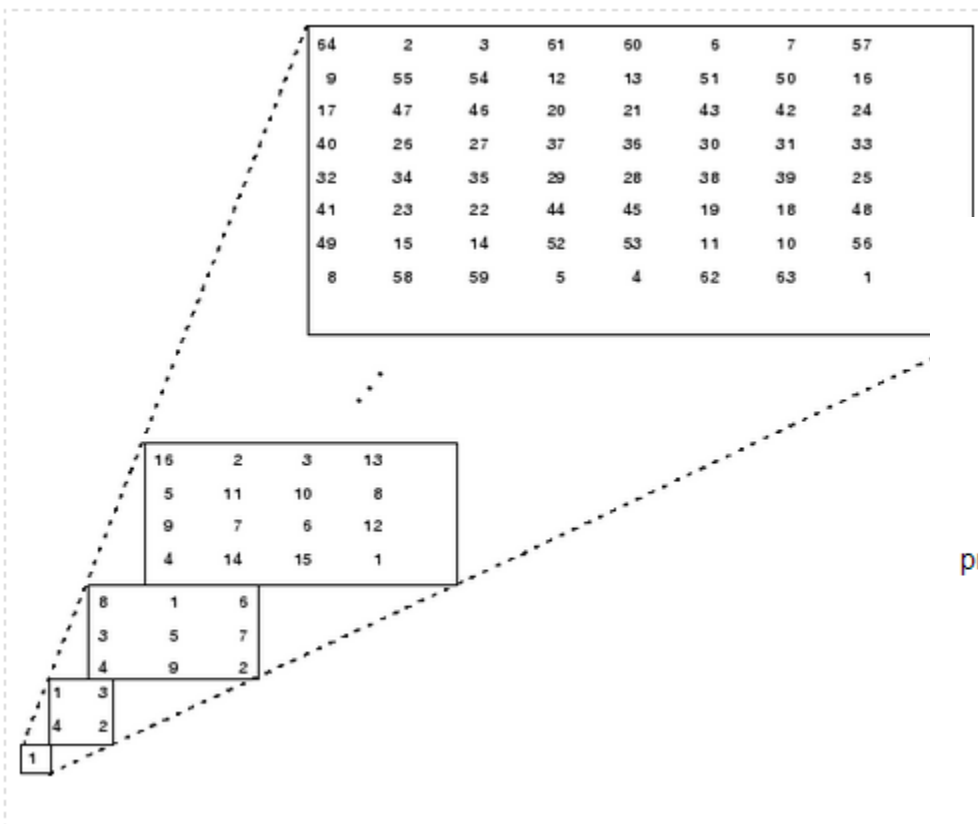
```
ans =
```

1	4
2	5
3	6

Cell Arrays



Cell Arrays



```
M = cell(8,1);  
for n = 1:8  
    M{n} = magic(n);  
end  
M
```

produces a sequence of magic squares of different order:

```
M =  
  
[          1]  
[ 2x2  double]  
[ 3x3  double]  
[ 4x4  double]  
[ 5x5  double]  
[ 6x6  double]  
[ 7x7  double]  
[ 8x8  double]
```

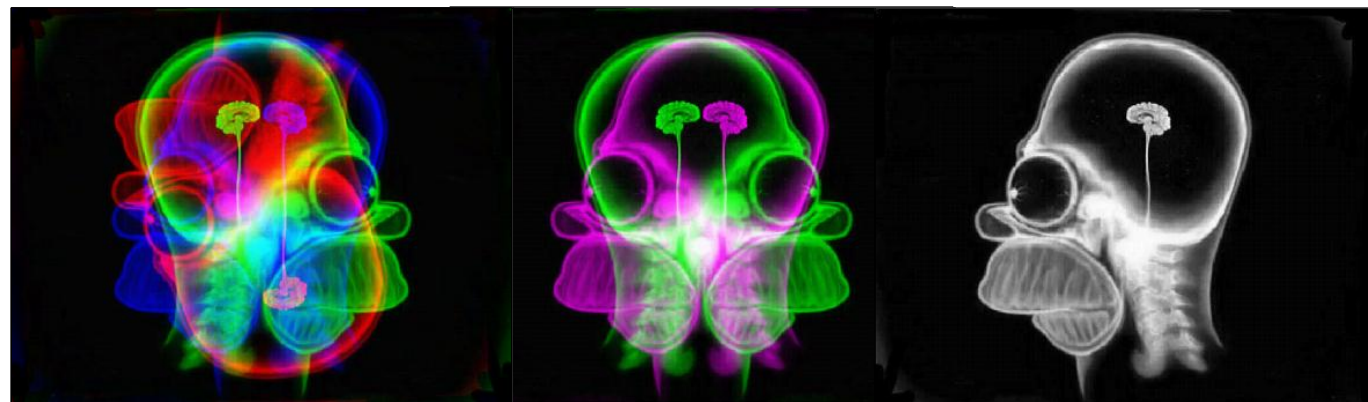
END OF LECTURE 1 ☺



עבודה עם איברי מערכים – 8: מטריצות מרובות מימד

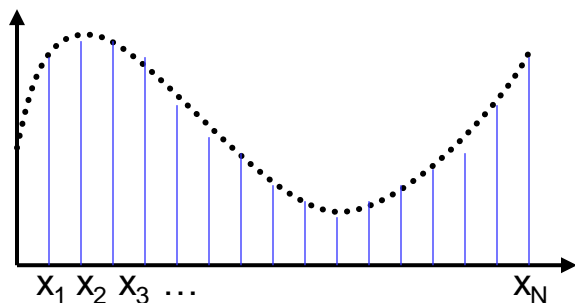
משימה: מניפולציה על תמונת RGB 🖥️

- העלו את תמונת patient.jpg מתוך ה-cd. ניתן לבצע זאת ע"י גרירת הקובץ ואישור.
- וודאו כי המטריצה patient מופיעה ב-workspace. בחנו את מימדיה ומאפייניה.
- בחנו את תמונת המטופל, ע"י הפקודה הפשוטה: `>> image(patient)`
- 1. באמצעות שימוש באינדקסים, הפכו את כיוון פני המטופל מצד שמאל לצד ימין. וודאו את התוצאה באמצעות הפקודה `image`.
- 2. באותו אופן, שנו את הכיוון האופקי של התמונה רק של תמונת הצבע הירוק - כלומר, ה-page השני מתוך השלושה. בחנו את התוצאה.

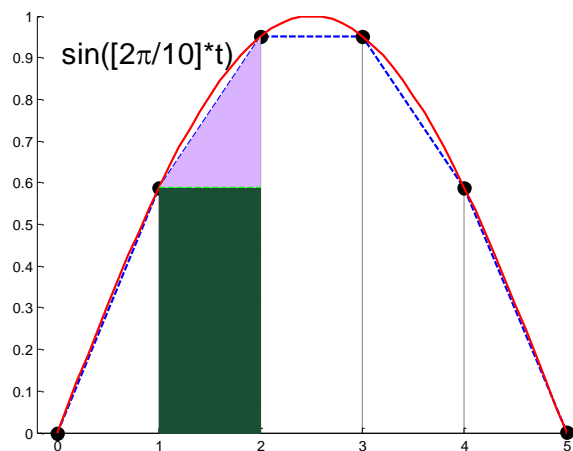


אינטגרציה נומרית - 1

- אות דיסקרטי מיוצג ע"י סדרת מספרים במיקומים נתונים אשר מהווים את כל האינפורמציה.



- אינטגרציה נומרית מקרבת את הפתרון האנליטי ע"י חישוב סכומי שטחים מקורבים הכלואים בין הנקודות.



■ אינטגרל רימן

■ שיטת הטריפז

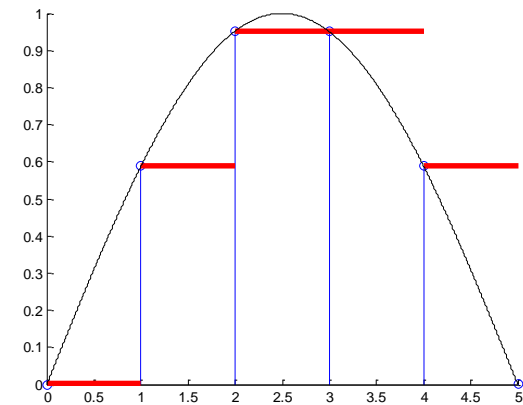
אינטגרציה נומרית - 2

■ שיטת רימן:

$$I = \sum_i f(x_i) \cdot (x_{i+1} - x_i)$$

```
>> t=0:1:5;  
>> y=sin(2*pi/10*t);  
>> I = sum(y)/pi      % dt = 1, const.
```

תוצאת החישוב, מנורמלת ב- π : $I = 0.9797$



■ שיטת הטרפז:

$$I = \sum_i \frac{f(x_{i+1}) + f(x_i)}{2} \cdot (x_{i+1} - x_i)$$

ניתן לעשות שימוש בפונקציה הספריה trapz:

```
>> I_trapz = trapz(t,y)/pi
```

תוצאת החישוב כעת היא:

$I_{\text{trapz}} = 0.9797$

מדוע שתי השיטות מניבות אותה תוצאה?

גזירה נומרית - 1

- הנגזרת הדיפרנציאלית, אותה מקבלים בהשאפת מרווח הדגימה ל- ε , נשארת, עבור סדרות דיסקרטיות, במובן הפרמיטיבי שלה:

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}, \quad \xrightarrow{\mathbf{x(t)}} \quad \frac{x_{i+1} - x_i}{t_{i+1} - t_i} \quad \text{נגזרת ימנית:}$$

- מובן כי בחישוב הפרשים מקטינים את אורך הסדרה. ניתן להימנע מיצירת פאזה בין סדרת המקור לבסדרת ההפרשים באמצעות הנגזרת המרכזית.

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x-h)}{2h}, \quad \xrightarrow{\mathbf{x(t)}} \quad \frac{x_{i+1} - x_{i-1}}{t_{i+1} - t_{i-1}} \quad \text{נגזרת מרכזית:}$$

- מכיוון שלרוב עובדים עם מדידות מעשיות, נגזרת מבליטה רעש נומרי.
- בשל ייצוג מוגבל של תחום הערכים האפשריים, הדיוק הכי טוב שניתן לקבל הוא גודל מרווח הקוונטיזציה.
- הכרת פונקציות נומריות מובנות: gradient, diff

גזירה נומרית - 2

משימה: 

עבור האות הבא:

```
>>dt=0.05; t=(0:dt:2)';
```

```
>>y=sin(2*pi*t);
```

1. הציגו את האות ע"י הפקודה: `>>plot(t,y); % t,y- same length`

2. חשבו את הנגזרת הראשונה באופן נומרי (כמשתנה dydt) והציגו אותה בעזרת:

```
>>hold on; plot(t,dydt,'r') % in red. t,dydt - same length
```

3. מהאות dydt, חשבו את האינטגרל המצטבר בעזרת cumtrapz או cumsum. הציגו את התוצאה באותו אופן. האם הגעתם בחזרה לפונקציה המקורית?

גזירה ואינטגרציה נומרית – אומדן שגיאה

ההפרש בין סדרת המקור לשחזורה: $e = y - \hat{y}$

■ נחפש פרמטר בודד שיתאר את השגיאה -

■ $\bar{e} = \text{mean}(e) \leftarrow$ מתקבלת שגיאה ממוצעת אפס בשל סימטריות

■ $\bar{e} = \text{mean}(|e|) \leftarrow$ אמד שגיאה אבסולוטית. הביטוי אינו גזיר.

■ $\bar{e} = \text{mean}(e^2) \leftarrow$ שגיאה ריבועית ממוצעת (mse).

■ במצב זה, השגיאה תלויה באמפליטודה \leftarrow נרצה לנרמל את השגיאה:

```
>> en = (y-y_int) ./ y
```

חלוקה איבר איבר \leftarrow יוצרת חלוקה באפס: $\tilde{e} = \frac{y - \hat{y}}{y}$

$$\tilde{e} = \frac{\sum (y - \hat{y})^2}{\sum y^2}$$

חלוקת mse באנרגיה של אות המקור \leftarrow מונעת חלוקה באפס:

```
>> en = norm(y-y_int)/norm(y)
```

טרנספורמציות ליניאריות ב-2D – א'

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{s}_{\text{scaling}} \cdot \underbrace{\begin{bmatrix} \cos \alpha & \sin \alpha \\ -\sin \alpha & \cos \alpha \end{bmatrix}}_{\text{מטריצת סיבוב משמרת}} \cdot \underbrace{\begin{bmatrix} x - x_0 \\ y - y_0 \end{bmatrix}}_{\text{הזזה}}$$

באופן אחר:

$$\underline{v} \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ -b & a \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} c \\ d \end{bmatrix} \quad \underline{= Au + q}$$

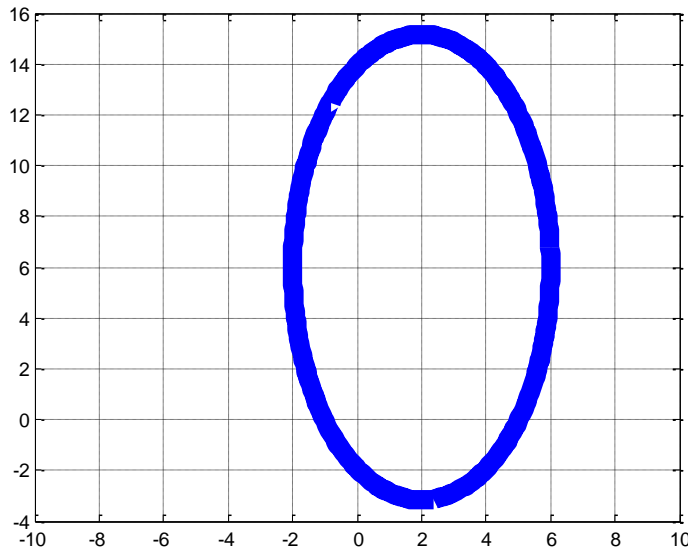
$$a = s \cos \alpha$$

$$b = s \sin \alpha$$

$$c = -s(x_0 \cos \alpha + y_0 \sin \alpha)$$

$$d = s(x_0 \sin \alpha - y_0 \cos \alpha)$$

טרנספורמציות ליניאריות ב-2D – ב'



% transformation parameters:

```
>>s = 2; alpha = pi/6;
```

```
>>x0 = -7; y0 = 5;
```

```
>>
```

```
>> % 1. define A, q.
```

```
>> % 2. calculate v with:
```

```
>> v = A*u+q;
```

```
>>plot(v(1,:),v(2,:))
```

נרצה לממש זאת באופן נומרי: 

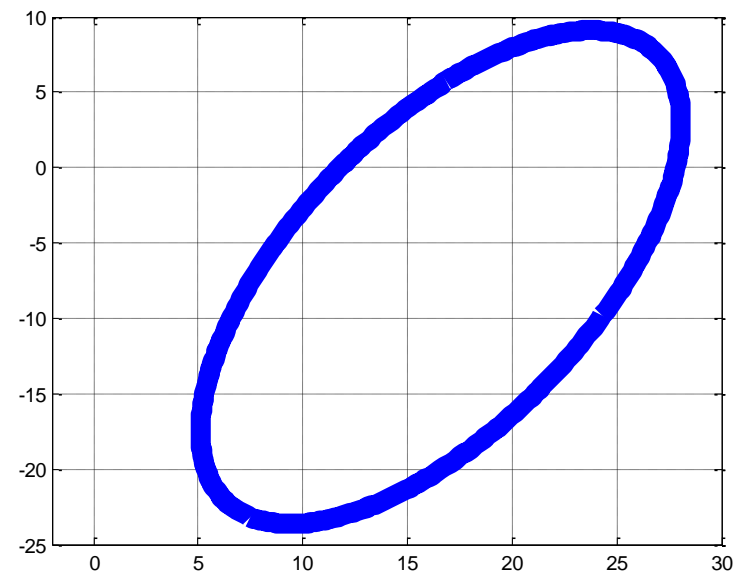
נתון סט נקודות (x_i, y_i) :

```
>>load points.mat
```

```
>>u = [X;Y]
```

```
>>plot(X,Y);
```

```
>>axis equal; grid on
```



רמז: השתמשו בפקודה `repmat`
בכדי לשכפל את q לאורך מתאים

טרנספורמציות ליניאריות ב-2D – ג'

כדאי לקחת הביתה...

■ פונקציות שימושיות:

min, max, sort, abs, sign, ceil, floor, fix - למצוא בעזרה

■ מציאת שורשי פולינום:

$$13x^3 + 25x^2 + 3x + 4$$

עבור הפולינום הבא -

```
>> C = [13 25 3 4];
```

```
>> r = roots(C)
```

$$\frac{5s+3}{s^3+3s^2-4} \rightarrow \frac{\frac{-8}{9}}{s+2} + \frac{\frac{7}{3}}{(s+2)^2} + \frac{\frac{8}{9}}{s-1} \quad \text{■ פירוק שברים חלקיים:}$$

```
>> [R,P,K] = Residue([5,3],[1 3 0 -4])
```

תרגול עצמי – דפי עבודה 1

תרגיל 1

צרו באופן הקצר ביותר סדרה הנדסית על בסיס 2, כלומר:

$$a(n+1) = 2*a(n), n=1,...,10$$

$$a(0) = 1$$

תרגיל 2

א- צרו וקטור t באורך 250, מ-0 עד 1, השתמשו בפקודה linspace. בעזרת הוקטור t צרו וקטור סינוס בשם sin6 שיכיל 6 מחזורים (אל תשכחו להכפיל ב-2*pi). הציגו את הוקטור sin6 בעזרת plot(t,sin6) כאשר ציר ה-X נע בין 0 ל-1.

ב- כעת צרו וקטור sin7 בעל 7 מחזורים. צרו מטריצה בשם sins בגודל 250X2 ש-sin6 הינו וקטור עמודה אחד שלה, ו-sin7 הינו וקטור עמודה שני. הציגו plot(t,sins).

תרגיל 3

צרו מטריצה שהיא לוח הכפל אך שכל הערכים הקטנים מ-20 והגדולים מ-70 מתאפסים. מצאו את כל המיקומים במטריצה kefel בהם הערך גדול שווה ל-81. (שימו לב שהתוצאה מתייחסת ל-kefel בתור וקטור ולא בתור מטריצה). להתמרת האינדקסים המתקבלים בזמנכן החופשי, השתמשו בפקודה ind2sub או כתבו לבד את כלל ההתמרה (פשוט מאוד).

תרגול עצמי – דפי עבודה 2

תרגיל 4

- א- צרו מטריצה נורמלית אקראית בגודל $[10 \times 5 \times 3]$ בעזרת הפקודה `randn`. מצאו את המקסימום בערך המוחלט ואת מיקומו במטריצה בעזרת הפקודות `max, abs, int2sub`.
- ב- בצעו את אותה פעולה תוך הפעלת הפונקציה `max` פעם אחת בלבד.

תרגיל 5

צרו סדרת מספרים אקראית יוניפורמית בתחום הערכים $[-1, 1]$ באורך 99 בעזרת `A=rand(99,1)`.

- א- חשבו ממוצע (mean) וסטיית תקן (std) של כל תת-סדרה המורכבת מכל איבר שלישי. נסו לבצע זאת בפעולה אחת כשהוקטורים מסודרים במטריצה אחת.
- ב- השתמשו בפקודה `sort` כדי למיין את `A` מהמספר הקטן אל הגדול.
- ג- הפכו את כיוון `A` כך שתהיה מהגדול אל הקטן וקחו את חמשת המספרים החיוביים הקטנים ביותר.
- ראינו קודם כיצד לבצע זאת בעזרת מערכי אינדקסים לוגיים. ניתן ומומלץ, לצורך התרגול, להשתמש בפונקציית `find` כדי למצוא את האיבר השלילי הראשון ולחתוך את הסדרה החל ממנו:
- ```
first_neg_index = find(A<0,1)
```

# תרגול עצמי – דפי עבודה 3

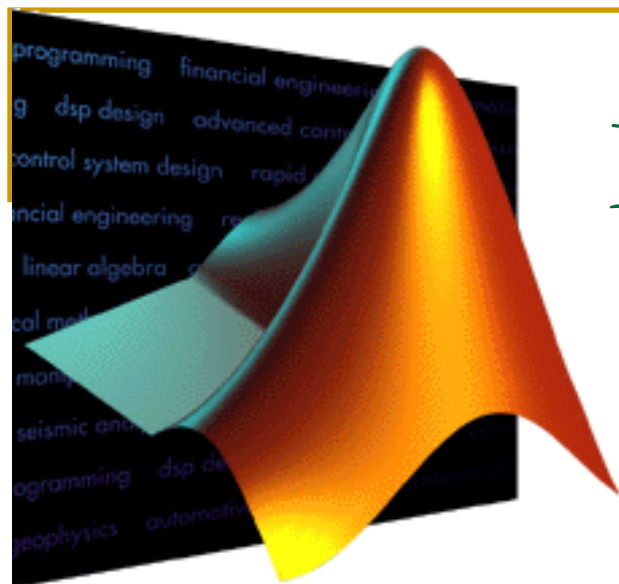
## תרגיל 6

- א- צרו מטריצת אחדים  $A$  בגודל  $[10 \times 10]$  בעזרת הפקודה `eye`. צרו וקטור עמודה  $v=1:10$ . נסו להוסיף את הוקטור לכל עמודה של  $A$ .
- ב- בעזרת הפקודה `vMat = repmat(v,_,_)` צרו מטריצת שכפולים (בשם `vMat`) של  $v$  במימדי מטריצת  $A$ . בדקו כי שתי המטריצות באותו הגודל וחברו ביניהן.

## תרגיל 7

- Define a matrix  $A$  of dimension 2 by 4 whose  $(i,j)$  entry is  $A(i,j)=i+j$
- Extract two 2 by 2 matrices  $A1$  and  $A2$  out of the matrix  $A$ .  $A1$  contains the first two columns of  $A$ ,  $A2$  contains the last two columns of  $A$
- Compute the matrix  $B$  to be the sum of  $A1$  and  $A2$
- Compute the eigenvalues and eigenvectors of  $B$
- Solve the linear system  $Bx=b$ , where  $b$  has all the entries equal to 1
- Compute the determinant of  $B$
- Compute the inverse of  $B$
- Compute the condition number of  $B$





# מבוא ל-MATLAB

## חלק #2

אס"ט - אגודת הסטודנטים בטכניון

אסף שפנייר

shpanier@gmail.com

עזרה והשראה: נעם ויסמן

# תכנית הקורס - התקדמות

## תכני MATLAB:

- היכרות וסביבת עבודה
- ביטויים ומערכים ופעולות נומריות
- מבני מידע וארגון נתונים
- פעולות גרפיות בסיסיות
- תכנות וכתיבת פונקציות
- פעולות לוגיות ובקרת זרימה
- נושאים מתקדמים:
  - ייעול וחיסכון בחישובים, Profiler
  - קריאת וכתיבת קבצים
  - הכרת toolboxes
  - גרפיקה מתקדמת
- Simulink
- ממשקי משתמש גרפיים (GUI)

## יישומים הנדסיים:

- פתרון מערכות לינאריות
- גזירה ואינטגרציה
- טרנספורמציות לינאריות
- הצגת תוצאות ניסוייות
- עבודה עם מנוע סימבולי
- פתרון משוואות דיפרנציאליות
- סטטיסטיקה ושערוך פרמטרים
- ניצול מידע חזותי
- פתרון מערכות לא לינאריות
- בניית ממשקים עצמאיים
- מידול מערכות דינמיות

# שיעור 2

## ■ מבני מידע

- עוד על מחרוזות
- מערכי תאים
- מבנים

## ■ משתנים סימבוליים

- משמעות העבודה עם מנוע סימבולי
- הגדרת משתנים והחלפתם
- יישומים

## ■ גרפיקה בסיסית

- הכרה בסיסית של אובייקטים גרפיים
- שרטוט מהיר בעזרת plot
- שרטוט מספר סדרות בגרף בודד
- hold, שרטוט עמודות מטריצה, שרטוט מאותה קריאה לפונקציה, שימוש ב-plotyy.
- עריכה של גרפים להגשה והוספת סימונים
- שרטוט מספר גרפים subplot
- אובייקטים גרפיים נוספים

## יישומים:

- ארגון נתונים
- כלי עזר מתמטיים במנוע הסימבולי:
  - פתרון משוואות
  - גזירה ואינטגרציה
  - שרטוט פונ' מהיר
  - טרנספורמציות לפלס (מכירים?)
- הצגת תוצאות ניסוייות
- בחינה גרפית של מערכת דינמית

# מחרוזות - 1


`>>myString = 'Hello Class'`  **1x9 array** ■ הגדרת מערך תווים:

`>>as = double(myString)` ■ ייצוג ascii:

`as = 72 101 108 108 111 32 67 108 97 115 115`


`>>yourString = fliplr(myString)`  $\rightarrow$  `'ssalC olleH'` ■ שרשור מחרוזות:

`matString = [myString ; yourString]`

$\rightarrow$  `matString =` `Hello Class`  
`ssalC olleH`  **1x9 array**

אם נרצה להוסיף כעת שורה שלישית באורך שונה, נשתמש ב-`strvcat`:

`matString = strvcat(matString , 'short')`


$\rightarrow$  `matString =` `Hello Class`  
`ssalC olleH`  
`short`  **1x5 array**

הפונקציה מרפדת את המחרוזות ברווחים לאורך של המחרוזת הארוכה ביותר.  
באותו אופן ניתן להשתמש בפונקציה **`str2mat`**.

## מחרוזות - 2

C1 = 'Hello'; C2 = 'hello'; C3 = 'hell'

■ השוואת מחרוזות:

אופרטור שוויון לוגי – משווה איבר איבר במערכים: 

```
>>C1 == C2
```

```
ans = 0 1 1 1 1
```

מה עושים עם תוצאה כזו?

>>C2 == C3 → הודעת שגיאה. לא ניתן להשוות מערכים באורך שונה

פונקציות להשוואת מחרוזות: 

- ❑ **strcmp**: compare whole strings → >>strcmp(C1,C2) → returns 0
- ❑ **findstr**: finds substring within a larger string → >>findstr(C3,C2) → returns 1
- ❑ **regexp**: most advanced function for string comparison and replacement.

■ התמרת ייצוגי תווים למספרים:

- ❑ **num2str**: convert from numeric to string array
- ❑ **str2num**: convert from string to numeric array (str2double is faster)

## מחרוזות - 3

עבור המשתנה `temp=25`, הציגו מחרוזת (באמצעות הפקודה `disp()`)  
אשר תציג את ערך המשתנה `temp` בתוכה ותיראה:  
`the room temperature is 25 degrees.`

פתרון: 

```
>>disp(['the room temperature is ' num2str(temp) ' degrees.'])
```

### ■ הרצת מחרוזת כביטוי:

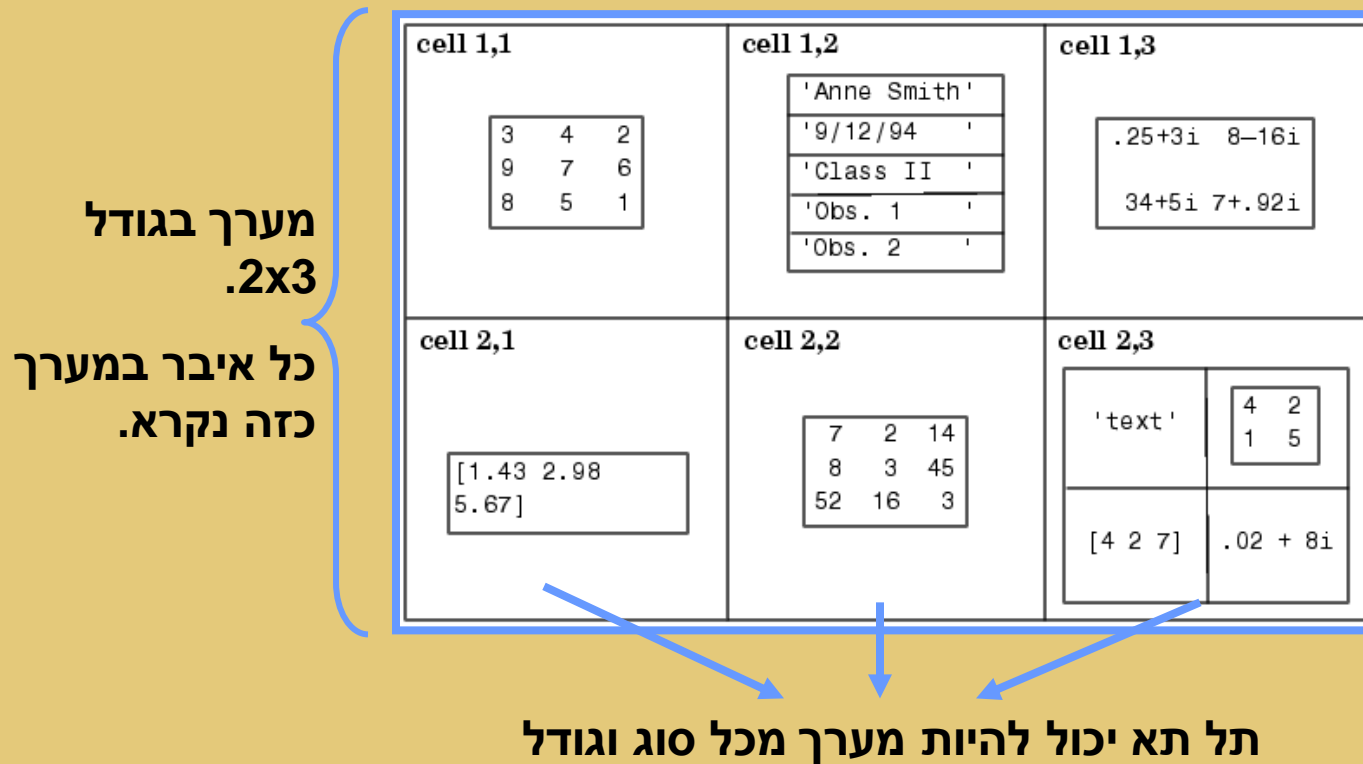
```
>>operSrting = ['x = 2^3 + exp(0)']
```

`operString`: כרגע המשתנה `x = 2^3 + exp(0)` מחזיק מחרוזת חסרת משמעות מעשית: `operString` כרגע המשתנה  
כותבים: MATLAB כדי לממש את המחרוזת כביטוי

```
>>eval(operString);
```

שאלה: האם ה-`cw` יציג את ערך `x` המוזן?

# תאים – cell arrays - 1



## תאים – cell arrays - 2

במערך תאים כל תא מכיל משתנה בגודל ומטיפוס שונה:

```
A(1,1) = {[1 4 3;
0 5 8;
7 2 9]};
A(1,2) = {'Welcome'};
A(2,1) = {patient}; % if image is loaded
A(2,2) = {-pi:pi/10:pi}
```

 **{}** מגדירים את  
המערך כתא



יצרנו cell array בגודל [2 x 2].

התשובה שחוזרת ב-cw מראה את המאפיינים השונים של כ"א מהתאים:

```
A = [3x3 double] 'Welcome'
 [600x800x3 uint8] [1x21 double]
```

■ פנייה אל תא במערך ואל איבר בתוך התא עצמו ע"י אינדקסים שונים:

```
>>A{1,1}(2,:)
```

 **{}**  **()**      ans =  
0 5 8



## תאים – cell arrays - 3

ניתן לאתחל מערך תאים ע"י הפקודה cell:

```
>>A = cell(1,3) % creates 1x3 cell array
```

```
A = [] [] []
 [] [] []
```

```
>>A(:,4) = {[]}
```

```
A = [] [] [] []
 [] [] [] []
```

```
>>A(2,:) = []
```

```
A = [] [] [] []
```

```
>>A = A(1:2)
```

```
A = [] []
```

הוספנו עמודה של איברים ריקים -

מחקנו שורה כי הצבנו "ריק" עבור כל השורה –

הזינו את הערך של גודל A (ע"י size) לתוך התא הראשון ואת המחרוזת 'hello' תא השני. 

⌘

## תאים – cell arrays - 4

### ■ תאי מחרוזות:

```
>>A = cellstr(['a';'bb';'ccc']) % creates 3x1 cell array
```

□ נוחים במיוחד בשל הבדלי אורכים של המחרוזות.

□ הרבה אפליקציות הדורשות קלט של מחרוזות שונות מקבלות cellstr.

### ■ ניתן להפעיל פונקציות מסוימות על מערך תאים:

```
>> ind = strcmpi(A,'Bb');
```

```
>> A{ind} → ans = bb
```

```
>> A(ind) → ans = 'bb'
```

} כולם מבינים את ההבדל?


### ■ ניתן להפעיל פונקציות מסוימות על התאים עצמם במקום על מערך התאים:

```
>>length(A) → ans = 3
```

מתייחס למשתנה כאל מערך רגיל -

```
>>cellfun('length',A) → ans = 1 2 3
```

פועל על כל תא בנפרד -

פונקציות להמרות תאים: הפעילו את cell2mat על מערך התאים הבא - 

```
>>C = {[1] [2 3 4]; [5; 9] [6 7 8; 10 11 12]}
```

פונקציות לתיאור תאים: cellplot(), celldisp(). הגדירו cell array ונסו אותן. 

# מבנים – structures - 1

patient

|          |            |     |    |
|----------|------------|-----|----|
| .name    | 'John Doe' |     |    |
| .billing | 135.00     |     |    |
| .test    | 120        | 39  | 78 |
|          | 98         | 154 | 12 |

מבנה הוא מערך המכיל טיפוסים משתנים שונים בשדות נפרדים, בעלי שמות מאפיינים.

שם השדה מופרד בנקודה משם המבנה:

```
» patient.name='John Doe';
» patient.billing = 135.00;
» patient.test= [120 39 78;
 98 54 12];
```

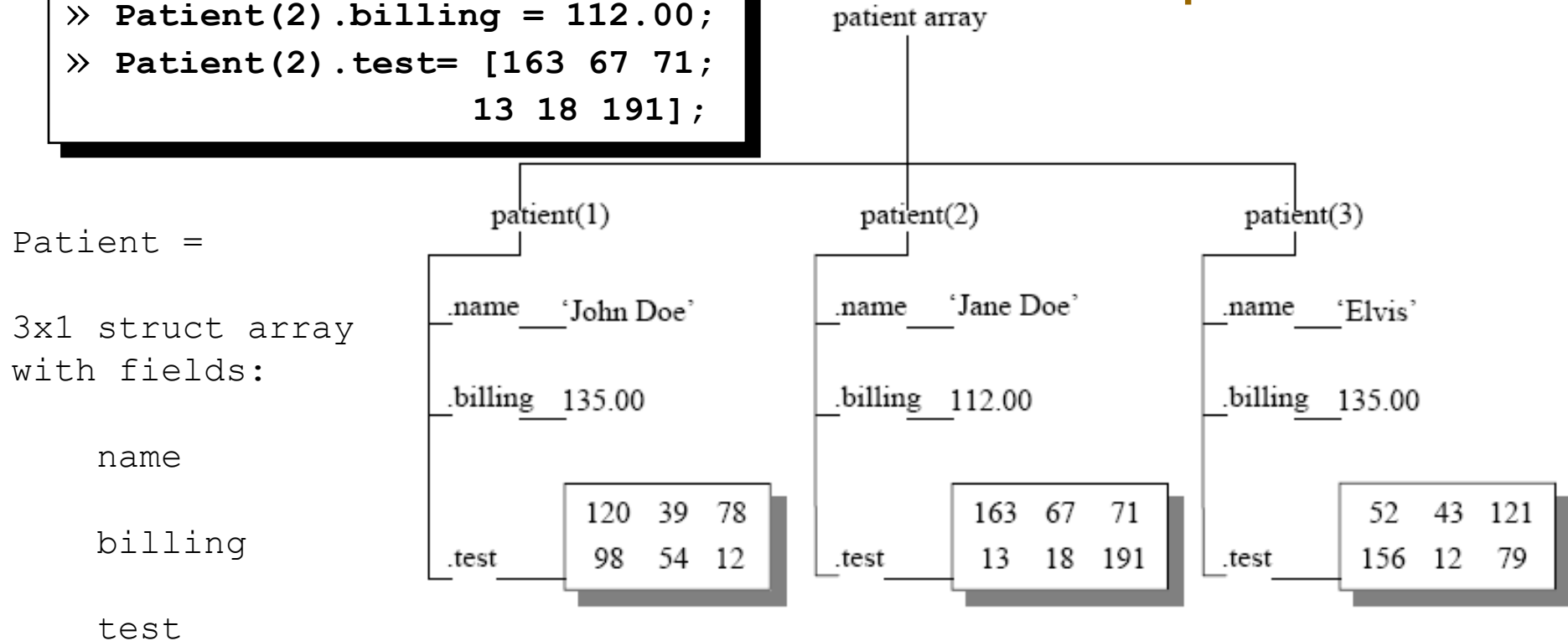
## יתרונות שימוש:

- טיפול מסודר בנתונים בעלי היררכיה.
  - ביצוע חיתוכים בין תכונות (שדות) של אלמנטים שונים במערך.
  - השמת כמות אינפורמציה רבה בתוך משתנה ראשי אחד.
- המבנה הוא מערך ומכיל יותר מאלמנט אחד בכל שדה (אם רוצים...)

## מבנים – structures - 2

```
>> Patient(2).name='Jane Doe';
>> Patient(2).billing = 112.00;
>> Patient(2).test= [163 67 71;
 13 18 191];
```

הרחבת מבנה למערך מבנים:



באופן כללי ניתן ליצור מערך מבנים באמצעות:

```
>>StrArray = struct('field1',{values1},'field2',{values2},...)
```

ערכי השדות ניתנים כתאים ולא מוגבלים לאותו הטיפוס. מימד התאים קובע את מימד המערך.

## מבנים – structures - 3

שם שדה יכול להיות **דינמי**, עבור קלט משתמש או תוצאת תוכנית: 

userField = [];                      הזינו לשורת אתחול המבנה שם שדה וערך לבחירתכם:

userVal = [];

s.(userField) = userVal                      *% note the parentheses*

קבלת פירוט השדות של מבנה:                      **>>fieldnames(structname)**

מחזיר מחרוזות של שמות השדה לתוך מערך תאים **(שאלה: מדוע לתאים?)**

מחיקת שדה ממבנה:                      **>>patient = rmfield(patient,'billing')**

מוריד את שדה ה-billing מהמבנה patient ומזין אותו חזרה לתוך משתנה patient\_new.

גישה לאיבר במערך:                      **>> patient(3)**

```
patient(3)
 .name 'Elvis'
 .billing 135.00
 .test 52 43 121
 156 12 79
```

**>> patient(3)**

ans =            name: 'Elvis'


                 billing: 135

                 test: [2x3] double

**>> patient(3).test(1,2)**

ans = 43

# תרגיל סיכום – מבני מידע (בדפים)

בתרגיל זה נסרוק ונערוך רשימת פונקציות יעודיות למטריצות בספריית MATLAB.   
המטרה: לאתר את קובץ הפונקציה הקטן ביותר בנפחו.  
עקבו אחר הפעולות הבאות. ממשו ובחנו כל שורה:

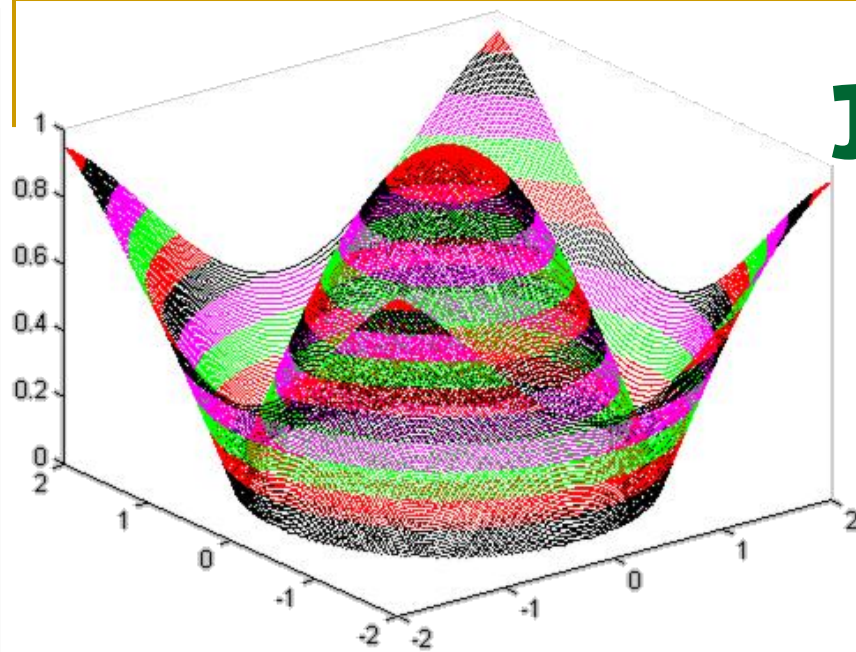
1. הספרייה הנבחרת.  
1.>>path = fullfile(matlabroot, 'toolbox\matlab')
2. פעולת dir מחזירה תשובה לתוך מבנה. בחנו את  
השדות ותחולתם. באילו שדות נמצאת  
האינפורמציה לה זקוקים?  
2.>>list = dir(path)
3. חלק מהאלמנטים הינם תיקיות ואנו מעוניינים רק  
בקבצים. צורת פקודה זו לא נוחה ויש צורך  
להעביר אותה לוקטור לוגי בינארי.  
3.>>list.isdir
4. פרישת התוצאות למערך תאים והפיכתו למטריצה.  
4.>>IsDir = cell2mat({list.isdir}) '
5. צרו מערך מבנים חדש שלא מכיל את האלמנטים  
שהינם directories.  
5.>>flist = \_\_\_\_\_
6. הגדרת וקטור המכיל את ה-bytes של הקבצים.  
6.>>bytes = cell2mat({flist.bytes}) '
7. חיפוש המינימלי מבין הקבצים.  
7.>>[mVal,mInd] = min(bytes)
8. השתמשו בתוצאת 7 והשלימו את הפקודה כדי  
לקבל את שם הקובץ המבוקש.  
8.>>minFile = flist(\_\_\_\_)\_\_\_\_
9. צרו מטריצת מחרוזות המכילה את שמות הקבצים. שלב  
זה לא הכרחי, אך טוב לאימון.  
9.>>fnames = \_\_\_\_\_({flist.name})'
10. קראו לשם הקובץ המבוקש מתוך המטריצה שיצרתם.  
10.>>minFile\_mat = .....  
**% lets see why it is so small:**  
>>open(minFile)

# תרגיל סיכום – מבני מידע (בדפים)

פתרונות: 

```
5.>>flist = list(~IsDir)
8.>>minfile = flist(minInd).name
9.>>fnames = strvcat({flist.name}')
10.>>minfile_mat = fnames(minInd,:)
```

# גרפיקה בסיסית

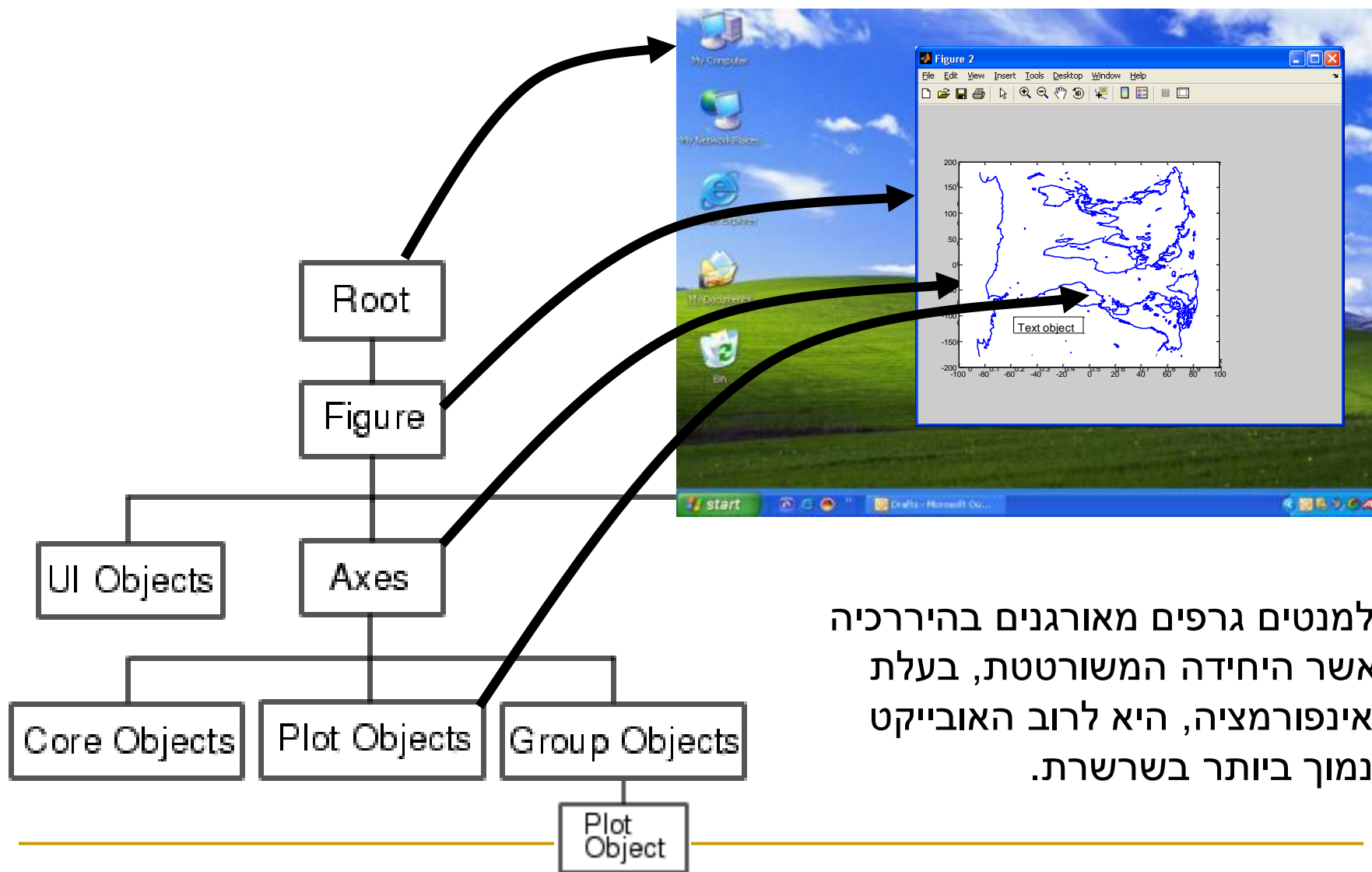


או: הדרך הכי טובה להעלות ציוני תרגילי מחשב

- הכרה בסיסית של אובייקטים גרפיים
- שרטוט מהיר בעזרת `plot`
- שרטוט מספר סדרות בגרף בודד
- עריכה של גרפים להגשה והוספת סימונים
- שרטוט מספר גרפים `subplot`
- אובייקטים גרפיים נוספים



# גרפיקה בסיסית – היררכיית אובייקטים



אלמנטים גרפיים מאורגנים בהיררכיה  
כאשר היחידה המשורטטת, בעלת  
האינפורמציה, היא לרוב האובייקט  
הנמוך ביותר בשרשרת.

# עריכת אובייקטים

## get properties list

## object's handle

|         |                                                           |   |                                                  |
|---------|-----------------------------------------------------------|---|--------------------------------------------------|
| get(0)  | 0                                                         | ← | Root                                             |
| get(hf) | hf = gcf, hf = figure(1)                                  | ← | Figure                                           |
| get(ha) | ha = gca<br>ha = axes('Parent',hf)<br>ha = subplot(m,n,k) | ← | Axes                                             |
| get(ho) | ho = gco (if selected)<br>ho = plot(...,'Parent',ha)      |   | axes-ל מתחת אובייקטים<br>line, patch, surface... |

**get:** `get(handle, 'PropertyName')` → returns property value

**set:** `set(handle, 'PropertyName', PropertyVal)`

`set(handle, 'PropertyName')` → returns property options

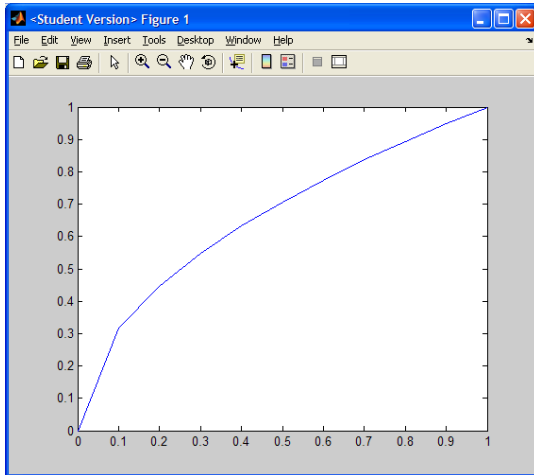
# גרפיקה בסיסית – דו מימד - 1

## ■ פונקציית Plot -

- השימושית והאוטומטית ביותר עבור שרטוט גרפים קווים דו-מימדיים.
- מייצרת את כל האובייקטים הדרושים לשרטוט הגרף.
- פועלת במספר וריאציות ועל תבניות קלט שונות:
- הצורה הכללית ביותר:

`plot(Xdata,Ydata,...,'properties',values,...)`

□ דוגמא,



```
>>x = 0:0.1:1; y = x.^.5;
>>plot(x,y)
```

- הסדרות x,y מאותו האורך.
- נוצר figure ובתוכו axes, אשר תחתיו מצויר אובייקט מסוג line.
- מכיוון שלא היה figure קיים, החדש קיבל את המספור 1.
- ברירת המחדל היא שרטוט הנקודות בקו רציף (לא חלק - הקירוב ליניארי) ובצבע כחול.
- הצירים הותאמו אוטומטית לתחום ערכי הסדרות המספריות.

# גרפיקה בסיסית – דו מימד - 2

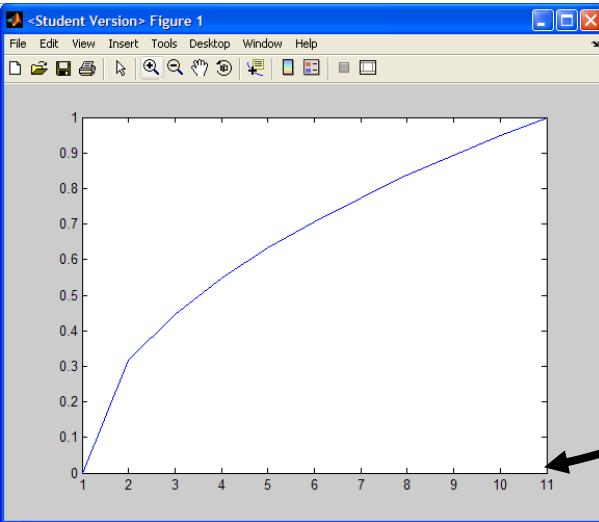
## ■ פונקציית Plot, המשך -

□ כאשר הקלט לפונקציה כולל סדרה אחת בלבד, plot מציירת את הסדרה אל מול האינדקסים התואמים:

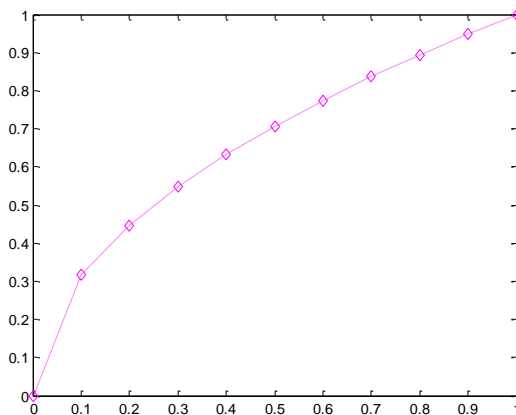
`>>plot(y)`    % *instead plot(x,y)*

□ הזנת מאפייני אובייקט - 1:

■ בצורה מקוצרת - `>>plot(x,y,'symbols')`



`>>plot(x,y,'m:d')`



| <u>Symbol</u> | <u>Color</u> |
|---------------|--------------|
| y             | yellow       |
| m             | magenta      |
| c             | cyan         |
| r             | red          |
| g             | green        |
| b             | blue         |
| w             | white        |
| k             | black        |

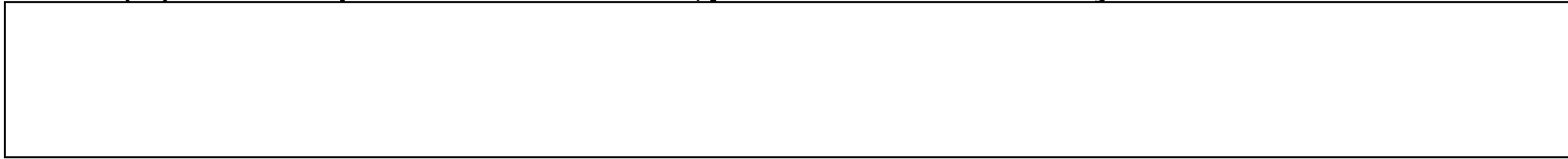
| <u>Symbol</u> | <u>Line Style</u> |
|---------------|-------------------|
| -             | solid line        |
| :             | dotted line       |
| -.            | dash-dot line     |
| --            | dashed line       |

| <u>Symbol</u> | <u>Marker</u> |
|---------------|---------------|
| .             | •             |
| o             | ◦             |
| x             | ×             |
| +             | +             |
| *             | *             |
| s             | □             |
| d             | ◇             |

# גרפיקה בסיסית – דו מימד - 3

## ■ פונקציית Plot, המשך -

נסו לשרטט את סדרה  $y$  הנתונה באותו אופן, כאשר צבע המרקר שחור (k) 



□ הזנת מאפייני אובייקט - 2:

■ קביעת תכונות מפורשת - `>>plot(x,y,'property1',val1,...)`

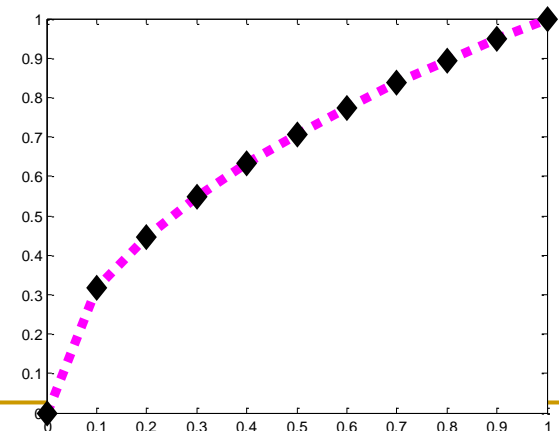
```
>>plot(x,y,'Color','m','LineStyle',':', 'LineWidth',6,...
'Marker','d','MarkerEdgeColor','k','MarkerSize',16)
```

כדי לקבל את רשימת התכונות האפשריות:

```
>>h = plot(x,y);
```

```
>>get(h)
```

נטפל בנושא זה באופן נרחב מאוחר יותר.



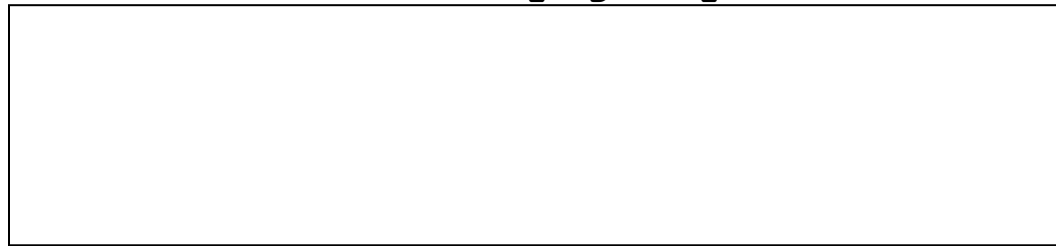
# גרפיקה בסיסית – דו מימד - 4

## ■ פונקציית Plot, המשך -

□ שרטוט מספר מרוכב

📖 נסו לשרטט את הסדרה -

```
>>t=0:.1:1; z = exp(j*2*pi*t)
```

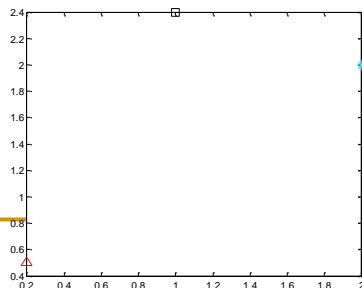


```
>>plot(z) % equals: plot(real(z),imag(z))
```

📖 משימה: שרטטו את צמדי הערכים  $\{x_i, y_i\}$ , כאשר כל צמד מיוצג ע"י נקודה

מצבע ומסוג שונה. השלימו את שורת הקוד הבאה לשם כך:

```
>>plot(2,2,__,1,2.4,__,0.2,0.5,__, 'MarkerSize', __)
```



| צבעים       | סוגי סמנים  |
|-------------|-------------|
| b r g k m c | * . ^ d s o |

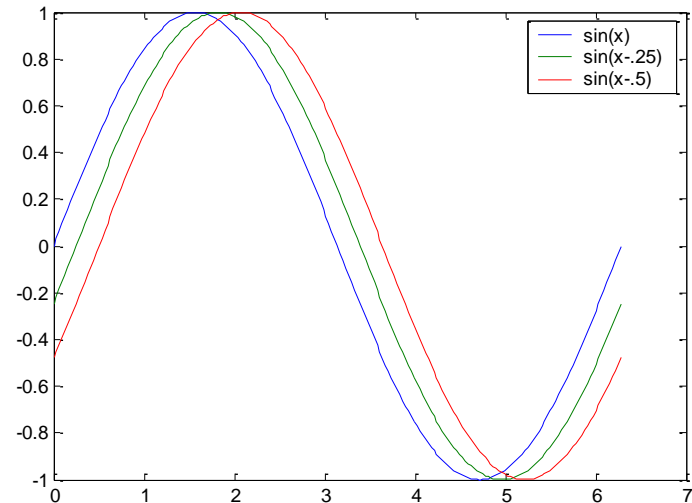
# גרפיקה בסיסית – דו מימד - 5

פונקציית Plot, שרטוט מספר אובייקטים יחדיו:


`plot(x1,y1,x2,y2,...)`

1. הזנת הסדרות בשורת הפקודה-

```
>>x = 0:pi/100:2*pi;
>>y = sin(x);
>>y2 = sin(x-.25);
>>y3 = sin(x-.5);
>>plot(x,y,x,y2,x,y3)
>>legend('sin(x)', 'sin(x-.25)' ...
, 'sin(x-.5)')
```



The **legend** command provides an easy identification.

משימה: שרטטו את הסדרות  $x=1:10$  ו-  $y=0:0.5:8$  על גבי גרף אחד. 

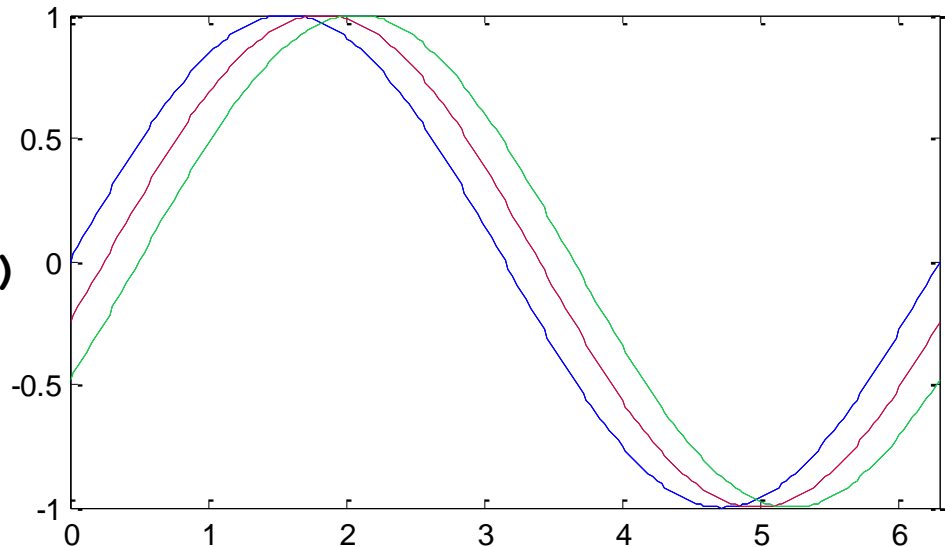


# גרפיקה בסיסית – דו מימד - 6

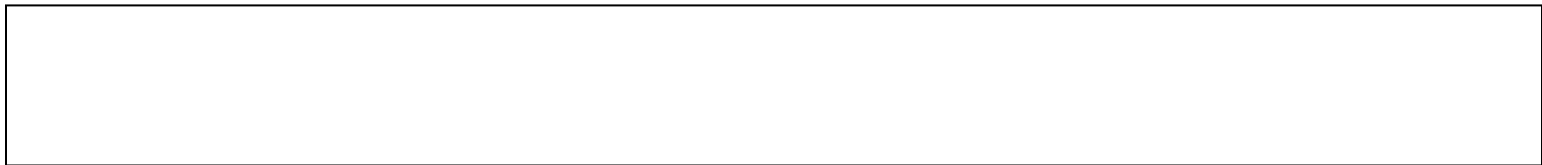
פונקציית Plot, שרטוט מספר אובייקטים יחדיו:

2. שימוש בפקודה **hold** – מונעת החלפת אובייקט גרפי באחר

```
>>x = 0: pi/100: 2*pi;
>>plot(x, sin(x), 'b')
>>hold on
>>plot(x, sin(x-0.25), 'r')
>>plot(x, sin(x-0.5), 'g')
>>axis tight
>>hold off
```



משימה: ציירו כעת, תוך שימוש בפקודה hold, את הסדרות  $x=1:10$  ו-  $y=0:0.5:8$  על גבי גרף אחד.



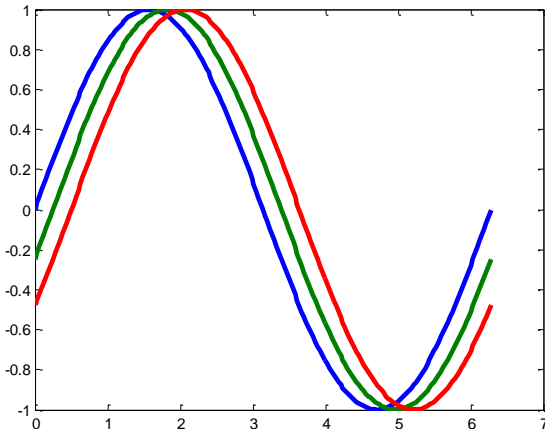


# גרפיקה בסיסית – דו מימד - 7

■ פונקציית Plot, שרטוט מספר אובייקטים יחדיו:

3. שרטוט עמודות מטריצה

```
>>x = 0: pi/100: 2*pi;
>>Mat=[sin(x)' sin(x-0.25)' sin(x-0.5)'];
>>plot(x,Mat,'LineWidth',3)
```



4. השוואה בין השיטות

| שיטה             | אורכים זהים                                                    | אורכים שונים           | סוגי גרפים שונים |
|------------------|----------------------------------------------------------------|------------------------|------------------|
| 1. צמדי וקטורים  | שליטה בצבעי וסוגי הקווים בלבד                                  | רישום קומפקטי          | לא אפשרי         |
| 2. hold          | שליטה מלאה בכל אובייקט, ל                                      | אפשרי, אך רישום מסורבל | אפשרי            |
| 3. וקטור ומטריצה | רישום קומפקטי, אך לא ניתן לשלוט בכל מאפייני אובייקט נפרד בגרף. | לא אפשרי               | לא אפשרי         |

**הערה –** ניתן לשלוט בהתנהגויות השיטות דלעיל באמצעות בקרה על ה-axes השולט.

# גרפיקה בסיסית – דו מימד - 8

שרטוט שתי סדרות בעלות יחידות שונות: 

```
>>[Ax,h1,h2] = plotyy(X1,Y1,X2,Y2)
```

Ax – handle (pointer) to axes

$h_i$  – handle to  $X_i, Y_i$  plot

modifying labels:

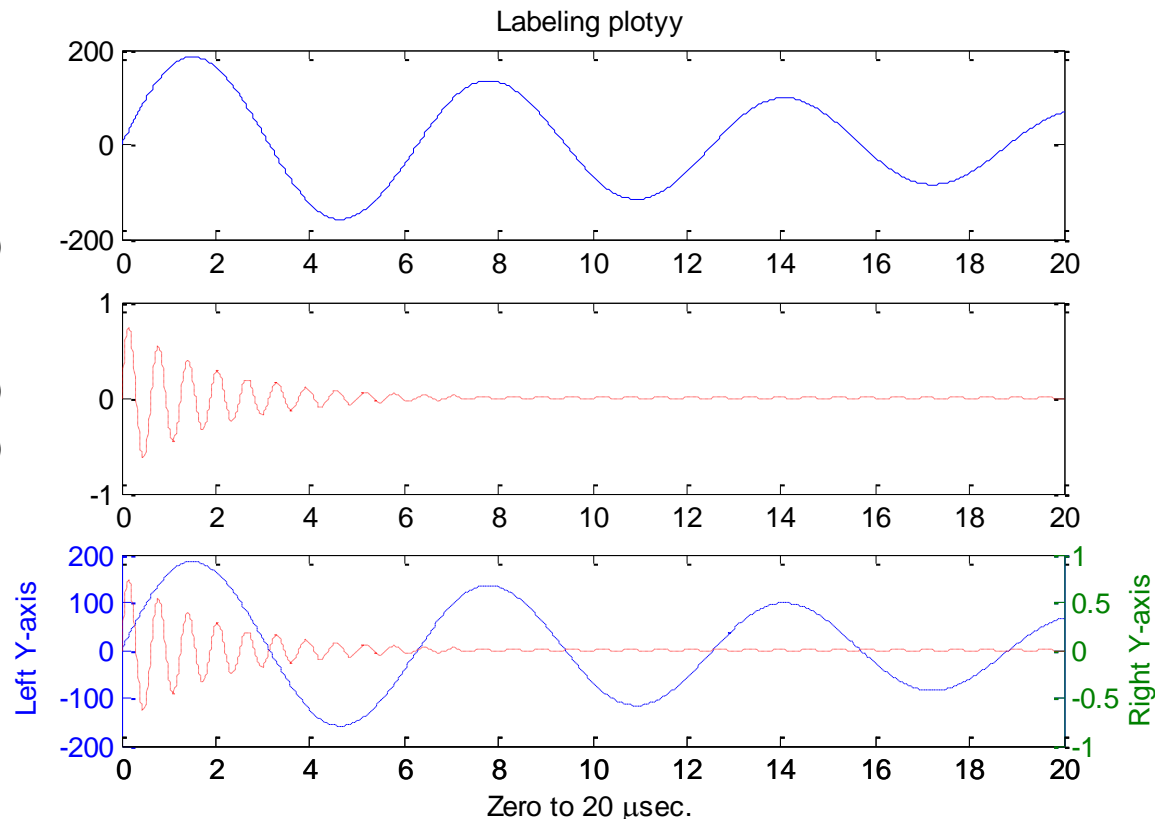
```
>>set(get(Ax(1), 'YLabel'),
'String', 'Left Y-axis')
```

```
>>set(get(Ax(2), 'YLabel'),
'String', 'Right Y-axis')
```



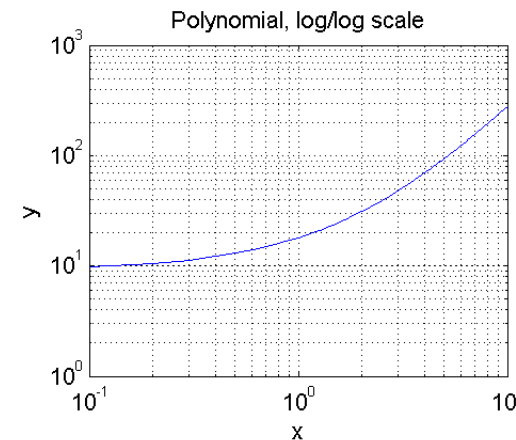
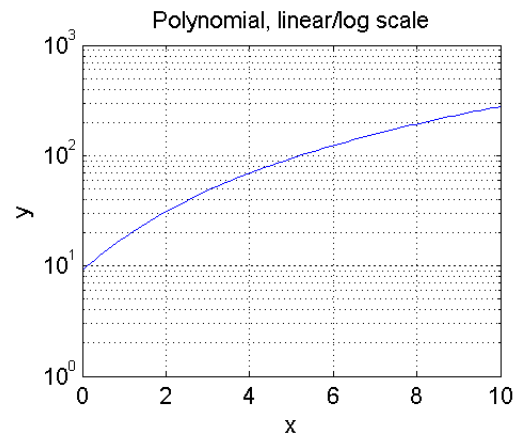
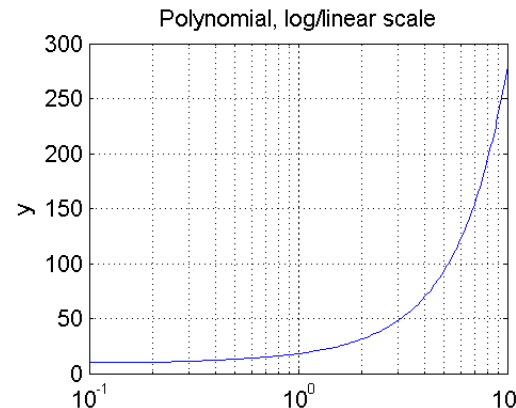
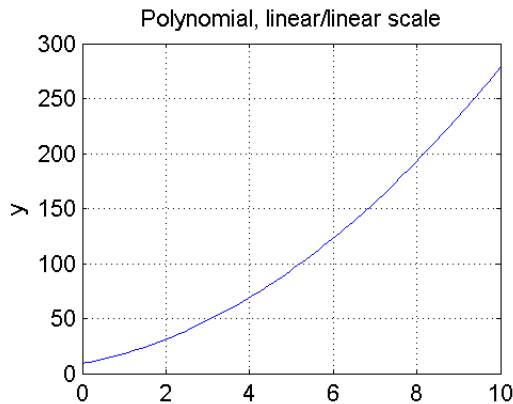
```
get(Ax(2), 'YLabel')
```

מחזיר handle לאובייקט  
YLabel הספציפי



# גרפיקה בסיסית – דו מימד - 9

□ שרטוט בסקאלות לוגריתמיות:



*% Generate the polynomial:*

```
x = linspace (0, 10, 100);
```

```
y = 2*x.^2 + 7*x + 9;
```

□ **plot (x,y)**

□ **semilogx (x,y)**

□ **semilogy (x,y)**

□ **loglog(x,y)**

# גרפיקה בסיסית – פקודות עזר - 1

## ■ פקודות עזר -

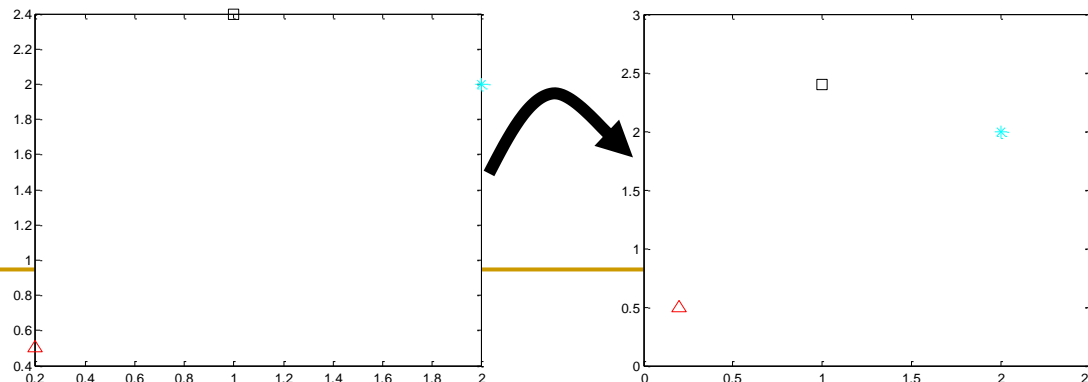
□ שליטה בתחום התצוגה - axis

| <u>command</u>               | <u>description</u>                            |
|------------------------------|-----------------------------------------------|
| axis ([xmin xmax ymin ymax]) | Define minimum and maximum values of the axes |
| axis square                  | Produce a square plot                         |
| axis equal                   | equal scaling factors for both axes           |
| axis tight                   | Scale axis to fit plot tightly                |
| axis normal                  | turn off axis square, equal                   |
| axis (auto)                  | return the axis to defaults                   |

🖥 נרצה לתקן את הגרף ה"בעייתי" מקודם ע"י שינוי תחום הצגת הצירים:

```
>>plot(2,2,'c*',1,2.4,'ks',0.2,0.5,'r^','MarkerSize',10)
```

```
>>axis([0 2.5 0 3])
```



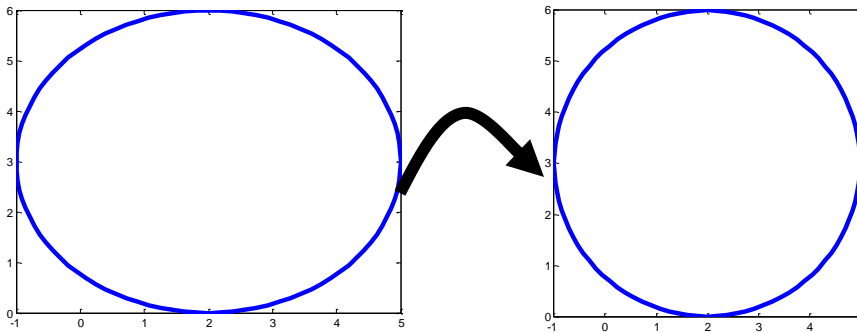
# גרפיקה בסיסית – פקודות עזר - 2

axis - דוגמא נוספת:



ציירו מעגל שמרכזו בנקודה (2,3) ורדיוסו שווה 3.  
השתמשו במשתנה  $\theta$  וחישוב את  $(x,y)$  ע"פ כלל ההתמרה –  
$$x = a + r \cdot \cos(\theta)$$
$$y = b + r \cdot \sin(\theta)$$

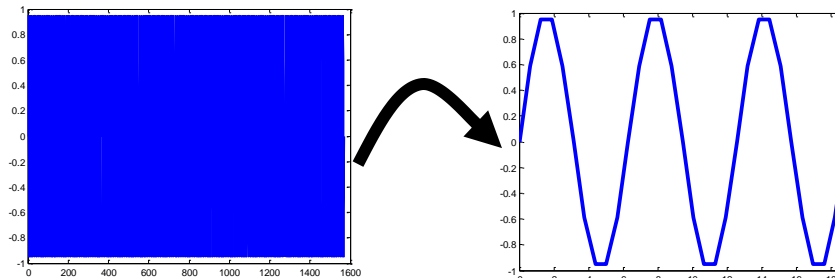
הצורה המתקבלת אינה מזכירה עיגול במבט ראשון. נסו את הפקודות `axis equal` ו-  
`axis square` לשם שיפור התצוגה.



`xlim`, `ylim` – שולט בגבולות הצירים בנפרד ■

```
>>t=0:2*pi/10:500*pi;
>>plot(t,sin(t))
>>xlim([0 6*pi])
```

**הפעולה לא חותכת את האות**



# גרפיקה בסיסית – פקודות עזר - 3

```
>>x=[0:0.1:2*pi];
>>y=sin(x); z=cos(x);
>>plot(x,y,x,z,'linewidth',2)
>>
>>title('Sample Plot','fontsize',14)
>>xlabel('X values','fontsize',14)
>>ylabel('Y values','fontsize',14)
>>legend('Y data','Z
data','fontsize',3)
>>grid on
```

■ תוספות ל-axes

■ axis labels

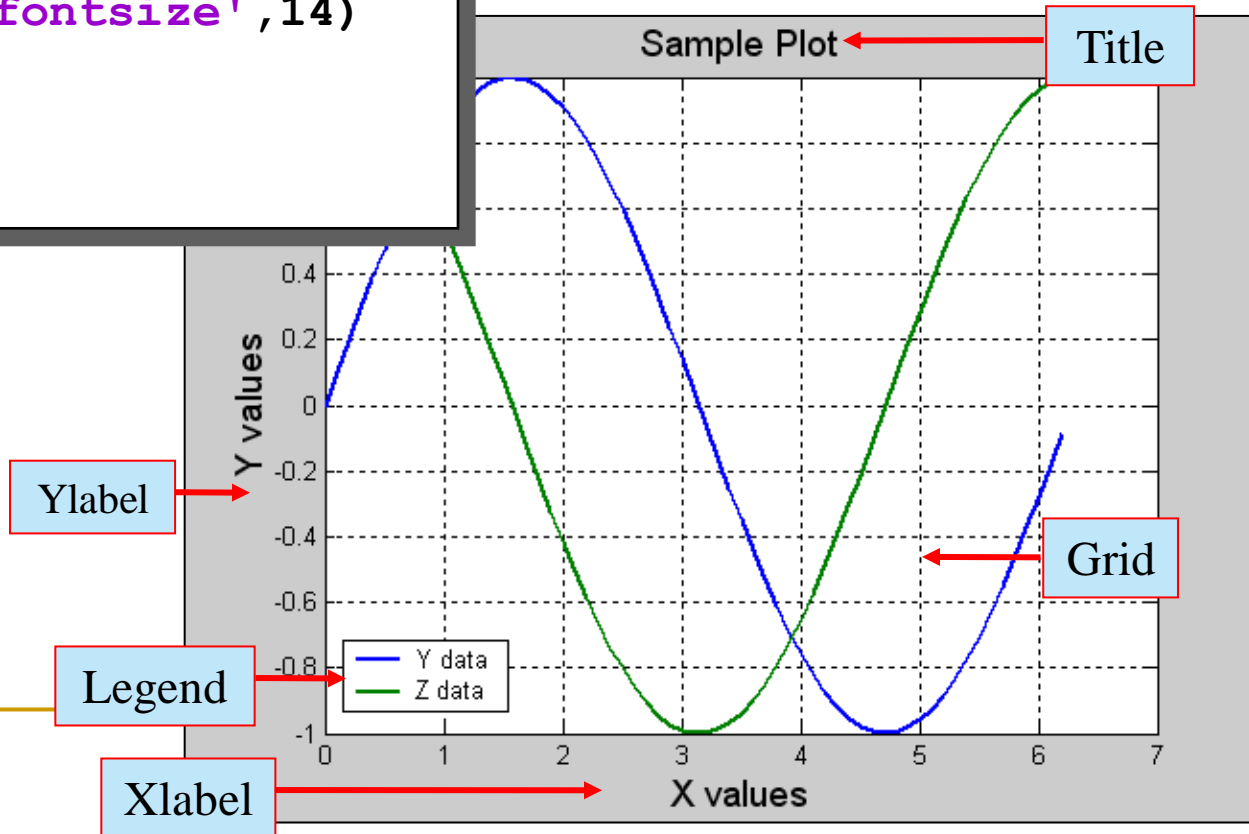
■ title

■ grid

■ ניתן להציב בטקסטים סימנים מיוחדים המפוענחים לפי קוד.

■ כל הטקסטים הם אובייקטים מסוג text והינם "ילדים" של ה-axes.

■ ה-legend יוצר אובייקט חדש מסוג axes.

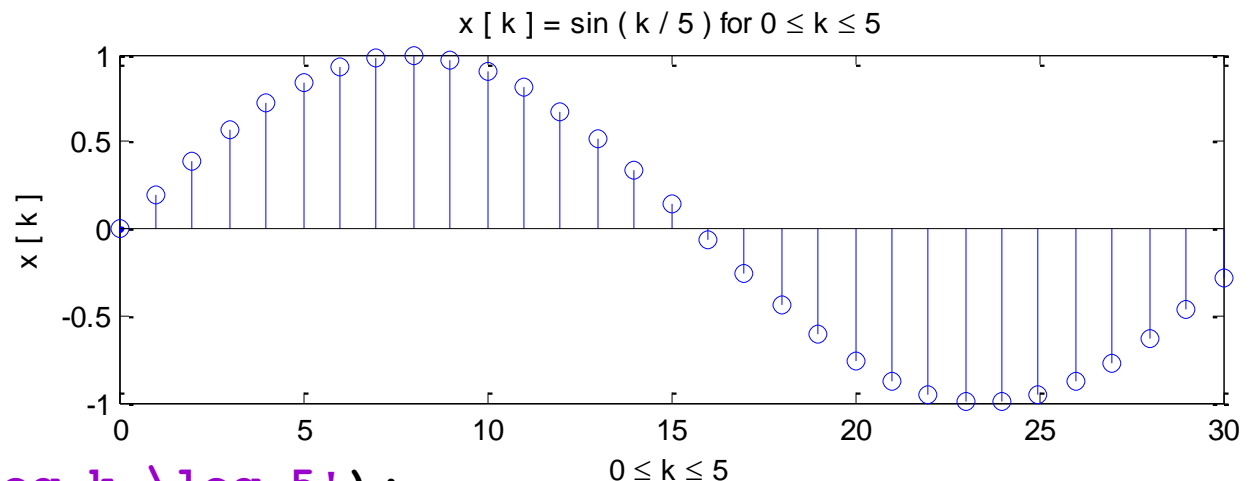


# גרפיקה בסיסית – שרטוט אות בדיד

- Stem function is very similar to plot. It is used to plot discrete time sequences. For more info: [help stem](#)

- Example:

```
>>k=[0:30];
>>x=sin(k/5);
>>stem(k,x)
```



```
>> xlabel('0 \leq k \leq 5');
>> ylabel('x [k]');
>> title('x[k] = sin(k/5) for 0 \leq k \leq 5');
```

• הסימן '\' באמצע המחרוזת מצביע על הפעלת מפענח על הטקסט שלאחריו.

• צורה נוספת לשרטוט אות דיסקרטי, ללא אינטרפולציה ליניארית של הנקודות: **stairs**

# עריכת אובייקטים - חזרה

## get properties list

## object's handle

|         |                                                           |   |                                                       |
|---------|-----------------------------------------------------------|---|-------------------------------------------------------|
| get(0)  | 0                                                         | ← | Root                                                  |
| get(hf) | hf = gcf, hf = figure(1)                                  | ← | Figure                                                |
| get(ha) | ha = gca<br>ha = axes('Parent',hf)<br>ha = subplot(m,n,k) | ← | Axes                                                  |
| get(ho) | ho = gco (if selected)<br>ho = plot(...,'Parent',ha)      |   | axes-ל מתחת :<br>אובייקטים<br>line, patch, surface... |

**get:** `get(handle, 'PropertyName')` → returns property value

**set:** `set(handle, 'PropertyName', PropertyVal)`

`set(handle, 'PropertyName')` → returns property options



# משימה בשרטוט 2D (בדפים)

שרטוט באופן רציף את הפונקציה הבאה ושתי נגזרותיה הנומריות (ראשונה ושניה) ע"ג גרף אחד. הנחיות:

אין דרישה לנגזרות מרכזיות

הקפידו על צבעים שונים

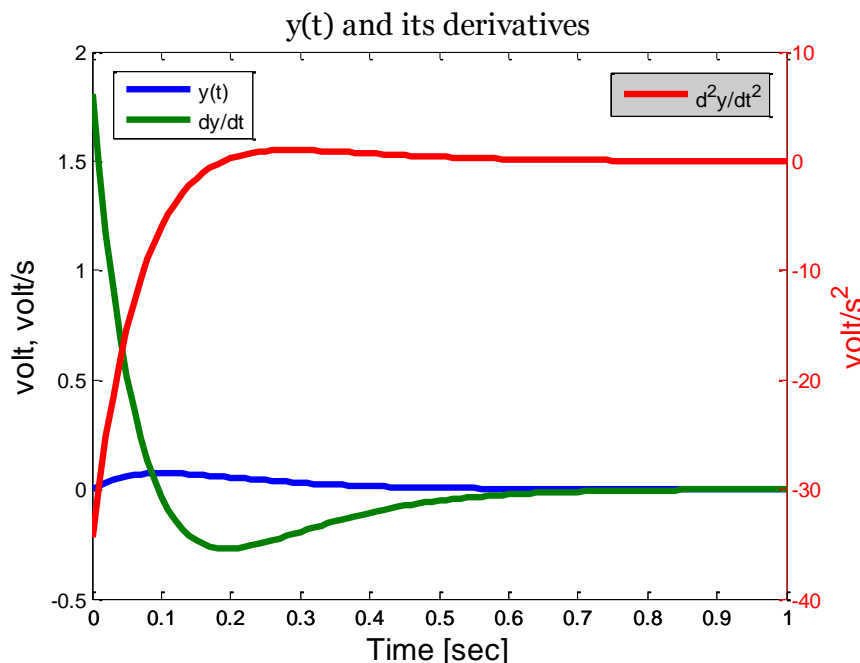
תנו כותרות מתאימות

הוסיפו מקרא (legend)

וודאו כי הפונ' מוצגות בתחום האינפורמטיבי

$$t = 0:0.01:10 \text{ [sec]}$$

$$y(t) = 2t \cdot e^{-10 \cdot t} \text{ [volts]}$$



## עזרה

• החזרת מצביע לאובייקט הציר:

`h = plot(...)`

`[hax,h1,h2] = plotyy(...)`

• "הדלקת" או קבלת תכונות של axes נתון h:

`axes(h), get(h)`

• שינוי תכונה של axes נתון h:

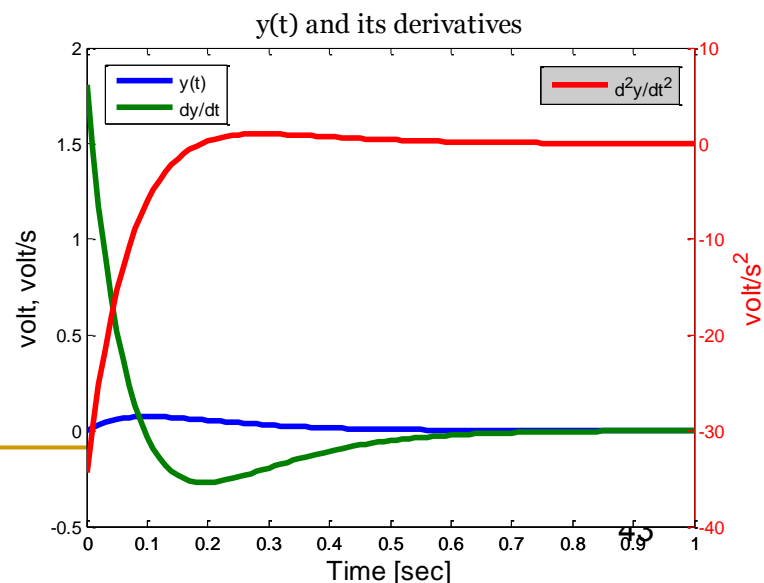
`set(h,'property',propval)`

• שינוי Xlimit של axes נתון h:

`set(h,'XLim',[xmin xmax])`

# משימה בשרטוט 2D - פתרון

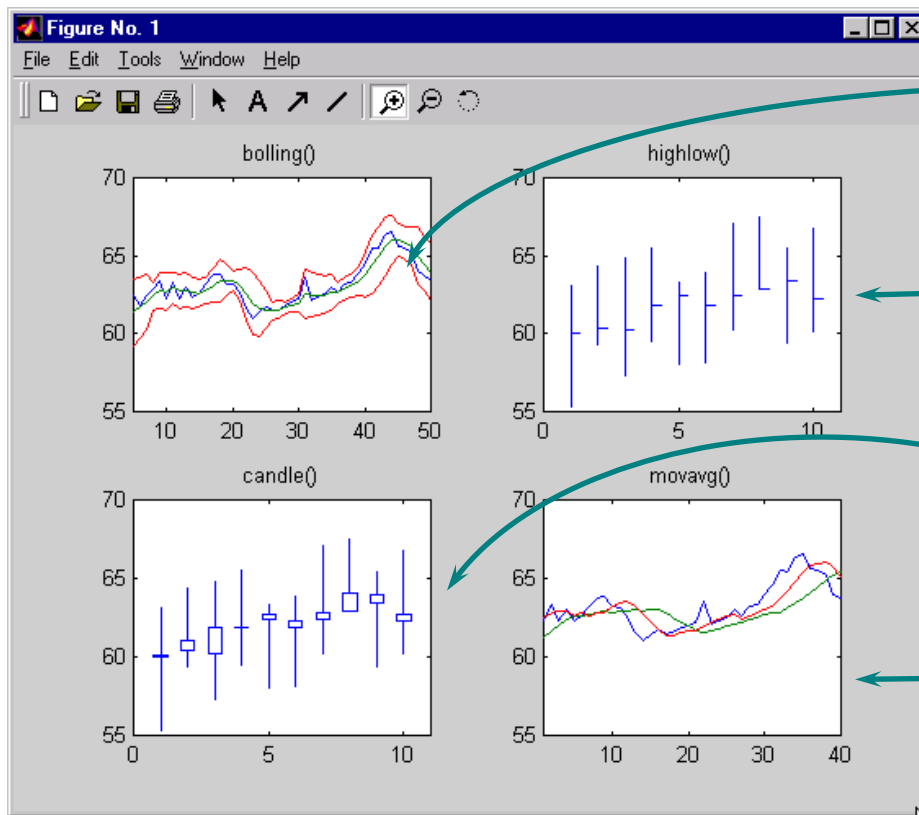
```
dt = 0.01; t = [0:dt:10]';
>>y = t*exp(-10*t);
>>dydt = diff(y)/dt; dydt2 = diff(y,2)/dt^2
>>figure; h x= plotyy(t,[y dydt; NaN],t(1:end-2),dy2dt2);
 כעת יש לנו שני axes מולבשים אחד על השני:
>>axes(hx(1)), legend('y(t)', 'dy/dt')
>>axes(hx(2)), legend('d^2y/dt^2',2)
>>set(get(hx(1), 'Ylabel'), 'String', 'volt, volt/s')
>>set(get(hx(2), 'Ylabel'), 'String', 'volt/s^2')
>>title('y(t) and its derivatives')
>>xlabel('Time [sec]')
>>set(h, 'XLim', [0 1])
```



# גרפיקה בסיסית – ארגון גרפים - 1

ניתן לצייר מספר axes ב-figure אחד באמצעות הפקודה **subplot**

Syntax: `subplot(rows,cols,index)`



» `subplot(2,2,1)`

» ...

» `subplot(2,2,2)`

» ...

» `subplot(2,2,3)`

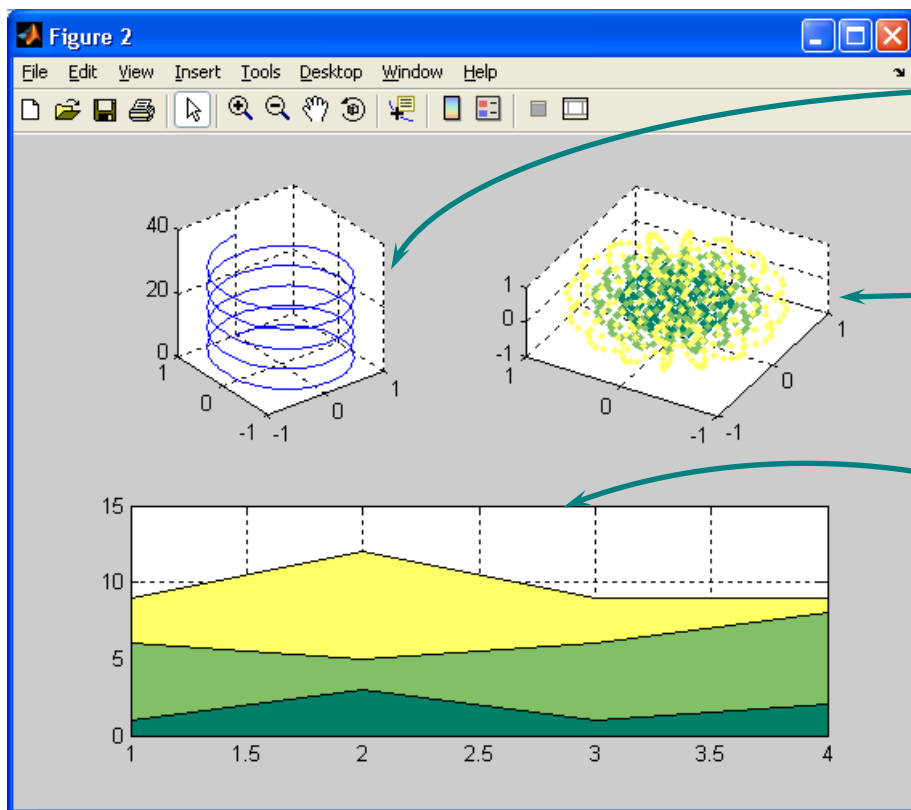
» ...

» `subplot(2,2,4)`

» ...

# גרפיקה בסיסית – ארגון גרפים - 2

באותו אופן ניתן להשיג:



```
» subplot(2,2,1)
```

```
» ...
```

```
» subplot(2,2,2)
```


```
» ...
```

```
» subplot(2,2,3:4)
```

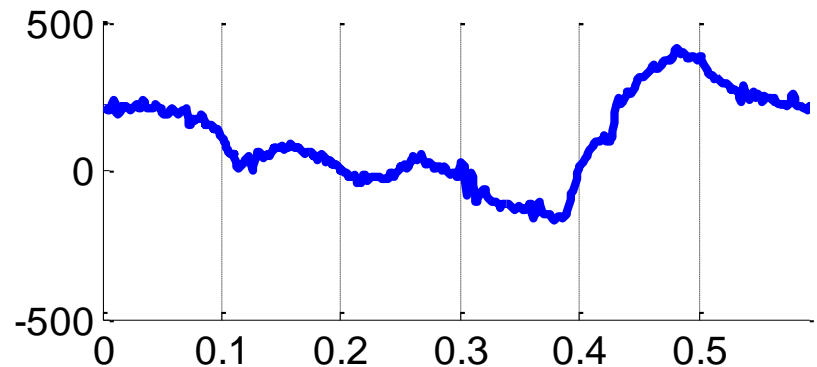
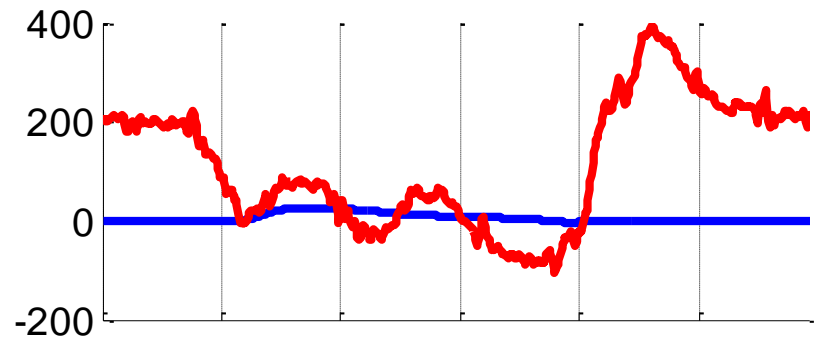
```
» ...
```

# עריכת ושיפור גרפים – דוגמא 1

נשתמש בתכונות אובייקטים לשם עיצוב גרף 

שרטוט axes אחד מתחת לשני וטיפול בצירי זמן תואמים 

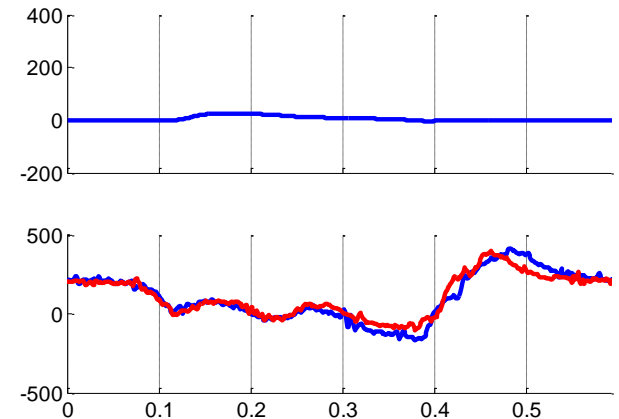
```
>>load BloodFlow % X, Y1 Y2 Y3
>>hf = figure;
>>ha1 = subplot(2,1,1);
>>hp1 = line(X,Y1,'Parent',ha1)
>>set(ha1,'NextPlot','add')
>>plot(X,Y2,'r')
>>ha2 = subplot(2,1,2);
>>hp3 = line(X,Y3,'Parent',ha2)
>>set([ha1 ha2],'Xlim',...
 [X(1) X(round(end/2))],...
 'Xgrid','on','FontSize',14)
>>set(ha1,'XTickLabel',[])
>>set([hp1 hp2 hp3],'LineWidthn',3)
→ error: hp2 doesn't exist
```



# עריכת ושיפור גרפים – דוגמא 2

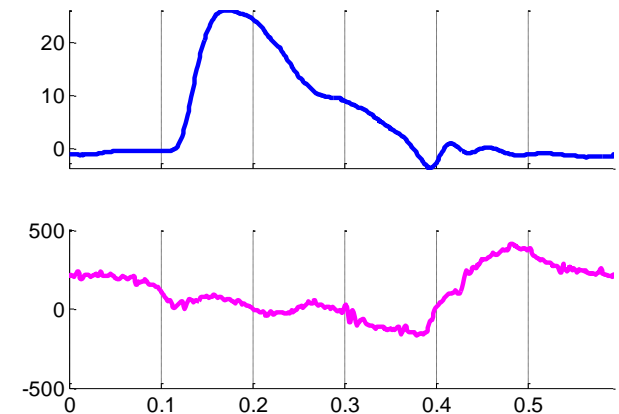
המשך, 

```
>>% find the handle:
>>hc = get(ha1,'Children')
>>IsLine = strcmpi(get(hc,'Type'),'line');
>>hp2 = hchild(IsLine & hc~=hp1);
>>set([hp1 hp2 hp3],'LineWidth',3)
>>% shift hp3 object to lower axes:
>>set(hp2,'Parent',ha2)
```




שימוש ב-XData של אובייקט קו 

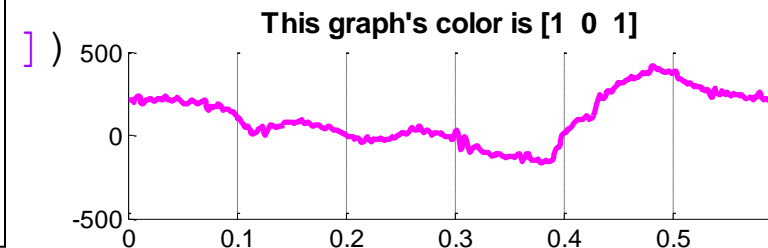
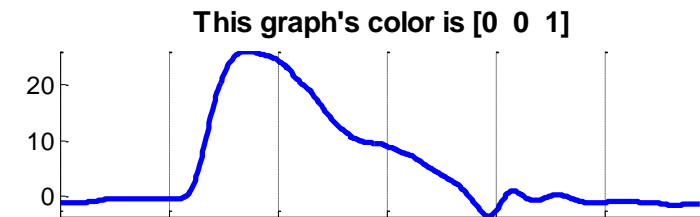
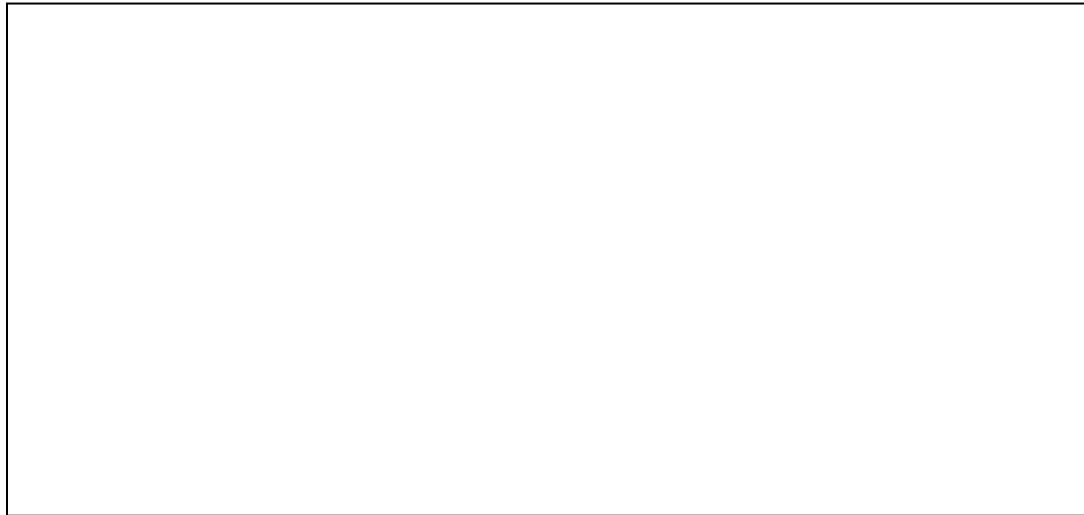
```
>>% combine the lines in ha2
>>% to the mean of Y2 and Y3:
>>clear Y2 Y3 % unavailable now
>>y2 = get(hp2,'YData');
>>y3 = get(hp3,'YData');
>>newYData = (y2(:) + y3(:))/2;
>>set(hp2,'YData',newYData,'Color','m');
>>delete(hp3)
>>set(ha1,'YLim',minmax(get(hp1,'YData')))
```



# עריכת ושיפור גרפים – דוגמא 3

## שימוש ב-`num2str` בכותרות

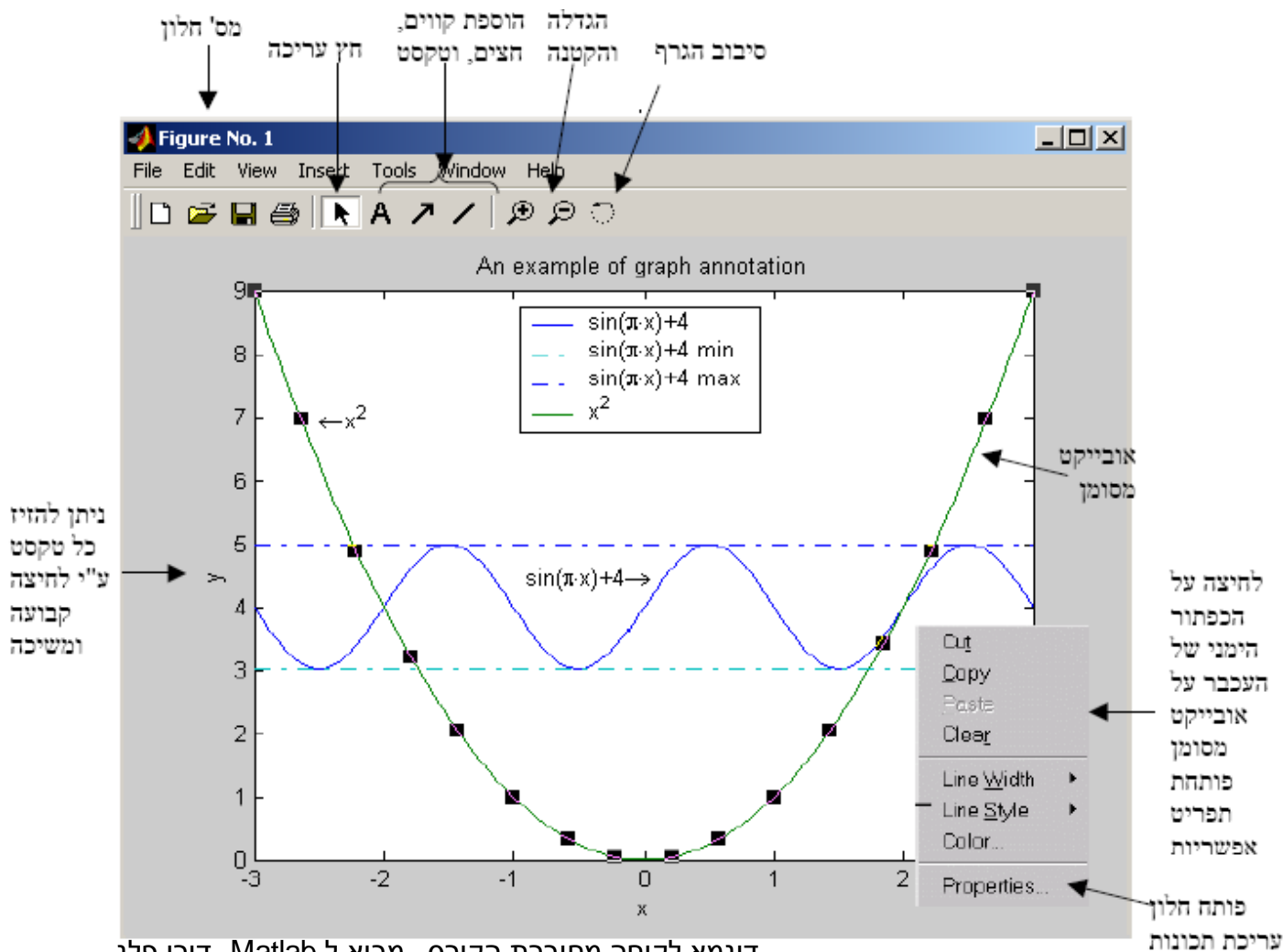
משימה: צרו לכל אחד מהגרפים כותרת המספרת מהו צבעו של הגרף   
המשורטט בתוכו. שימו לב כי צבע מיוצג ע"י שלושה מספרים `[R G B]`.



שימוש ב-`num2str` בכותרות יעיל במיוחד עבור מספור הגרף או פירוט פרמטר קובע.

קיימות עוד הרבה תכונות לכל אובייקט שניתן לבחון ע"י `get(h)` או ע"י `inspect(h)` שמפעילה ממשק משתמש לשליטה בתכונות האובייקט.

# עריכת ושיפור גרפים - שליטה ידנית



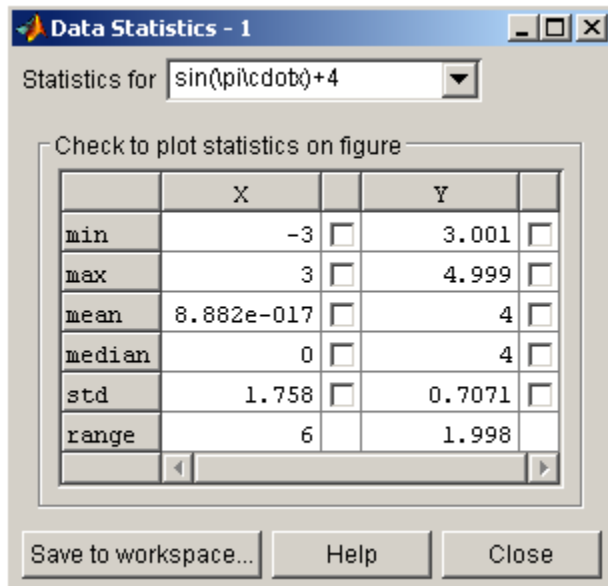
/ Generate M-File

דוגמא לקוחה מחוברת הקורס - מבוא ל-Matlab, דורי פלג

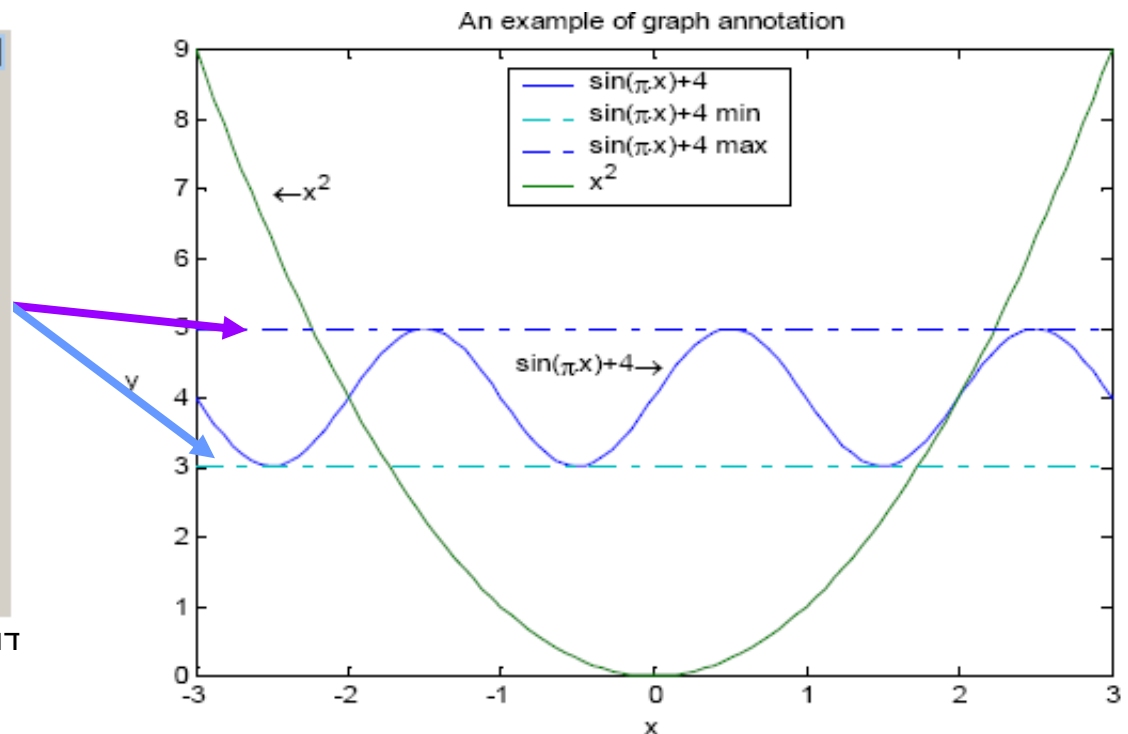


# עריכת ושיפור גרפים - הוספת סטטיסטיקה וסימונים

Tools→Data Statistics:



דוגמא מחוברת הקורס -מבוא ל-Matlab, דורי פלג



**Annotation:** `text(-2.2,7,' \leftarrow x^2','FontSize',18)`

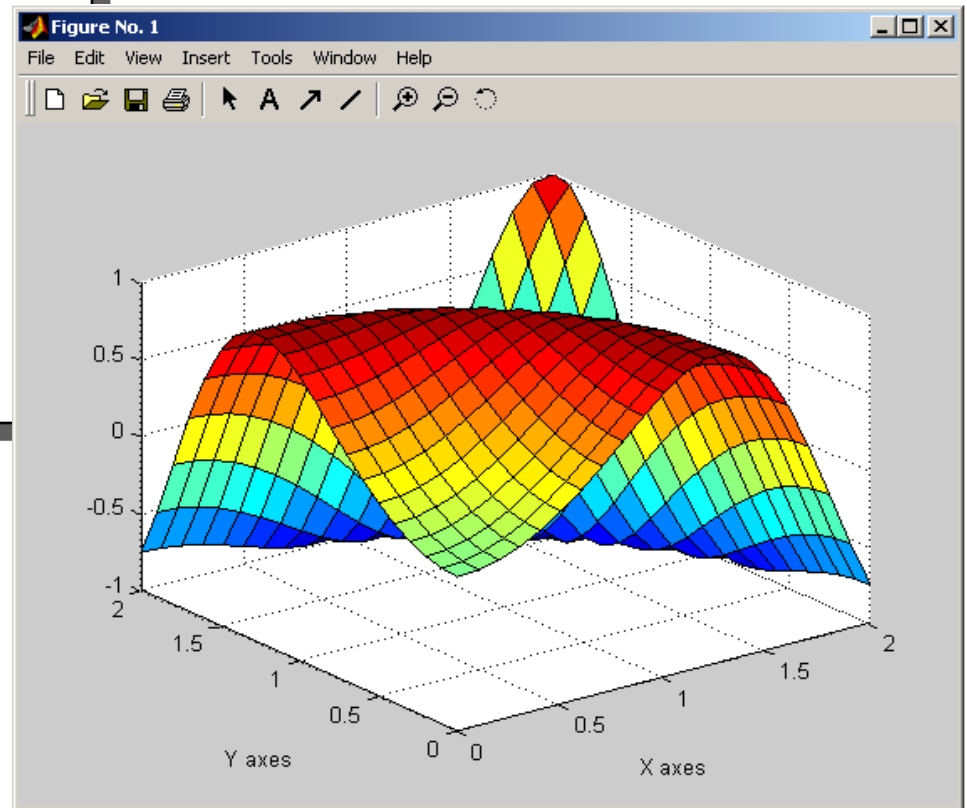
# תלת מימד - surface

נסקור מספר צורות גרפיות שכדאי להכיר:

## Surface object

```
x = 0:0.1:2;
y = 0:0.1:2;
[xx, yy] = meshgrid(x,y);
zz=sin(xx.^2+yy.^2);
surf(xx,yy,zz)
xlabel('X axes')
ylabel('Y axes')
```

מה meshgrid מבצע כאן?



# הסבר על meshgrid

```
x=1:4; y=1:5;
```

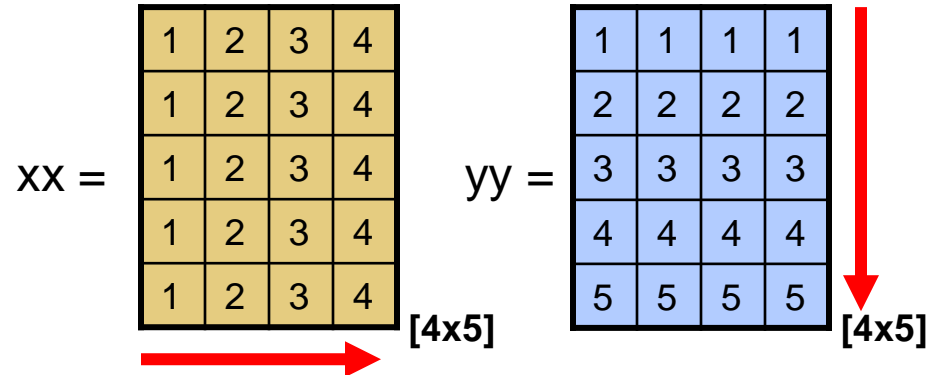
```
[xx,yy] = meshgrid(x,y)
```

עכשיו ניתן ליצור פונ' דו-ממדית של x ו-y:

```
zz = yy - xx
```

```
zz = ?
```

משתנה בכיוון אנכי (1)    משתנה בכיוון אופקי (2)



|     |     |     |     |
|-----|-----|-----|-----|
| 1-1 | 1-2 | 1-3 | 1-4 |
| 2-1 | 2-2 | 2-3 | 2-4 |
| 3-1 | 3-2 | 3-3 | 3-4 |
| 4-1 | 4-2 | 4-3 | 4-4 |
| 5-1 | 5-2 | 5-3 | 5-4 |

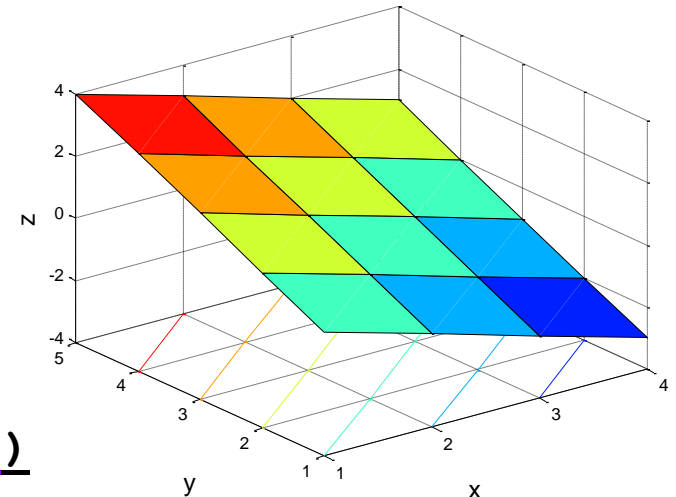
=

|   |    |    |    |
|---|----|----|----|
| 0 | -1 | -2 | -3 |
| 1 | 0  | -1 | -2 |
| 2 | 1  | 0  | -1 |
| 3 | 2  | 1  | 0  |
| 4 | 3  | 2  | 1  |

[4x5]

```
>>figure, surfc(xx,yy,zz)
```

```
>>xlabel('x'), ylabel('y'), zlabel('z')
```



# שרטוט מערכות דינמיות

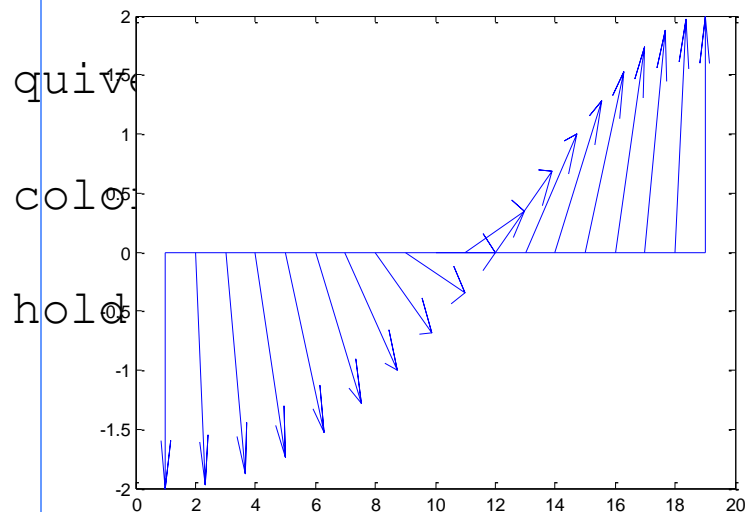
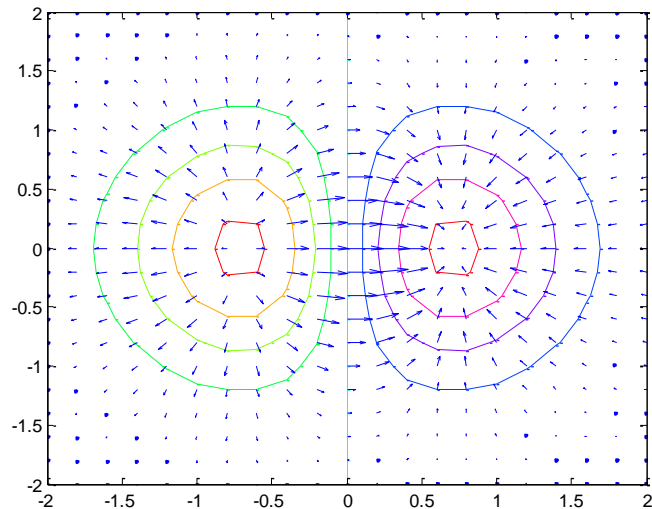
```
[X,Y] = meshgrid(-2:.2:2);

Z = X.*exp(-X.^2 - Y.^2);

[DX,DY] = gradient(Z, .2, .2);

contour(X,Y,Z)

hold on
```

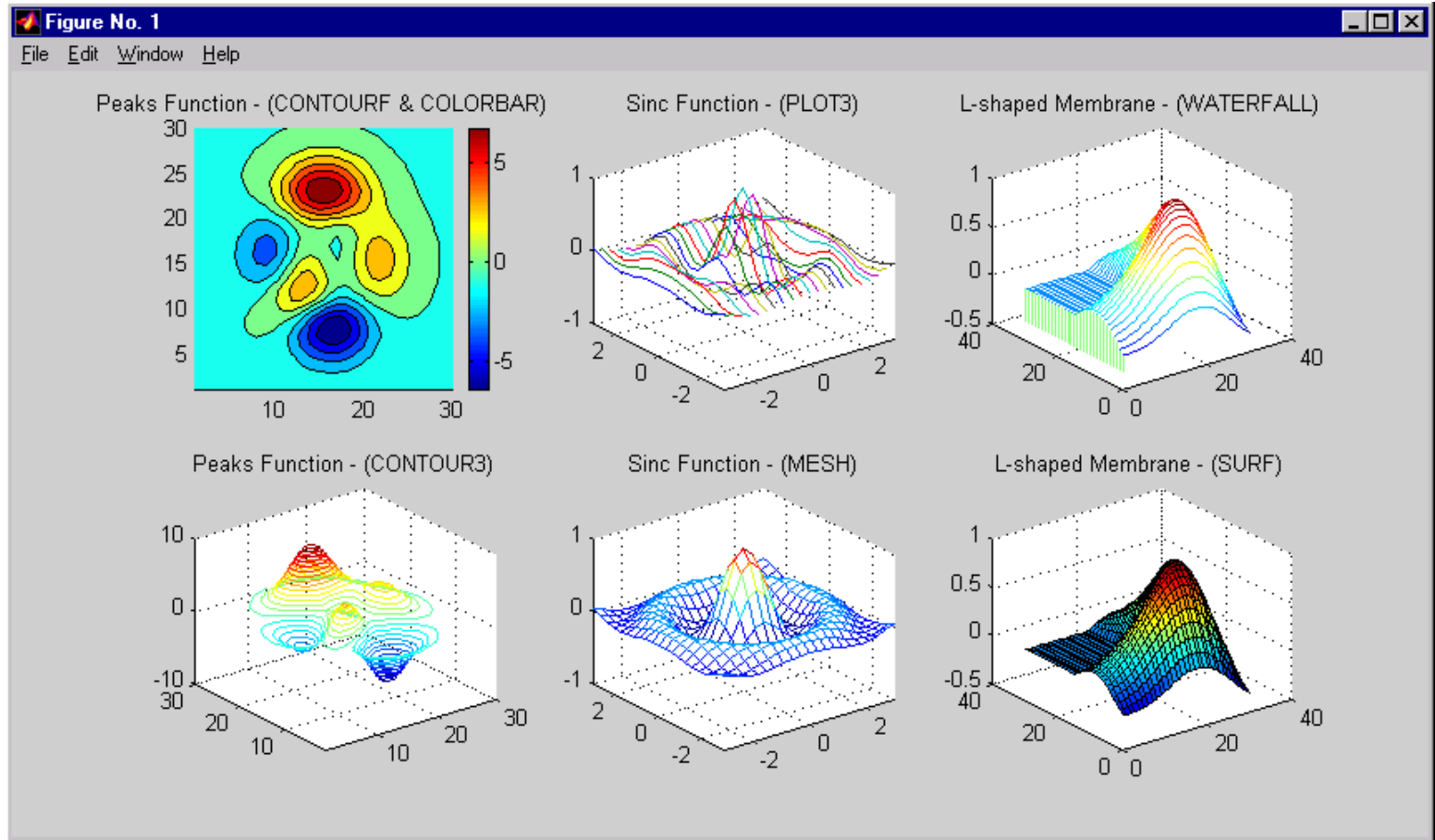


```
theta = (-90:10:90)*pi/180;
r = 2*ones(size(theta));
[u,v] = pol2cart(theta,r);
feather(u,v);
```

# פונקציות גרפיקה נוספות - 1

פונקציות תלת מימד:

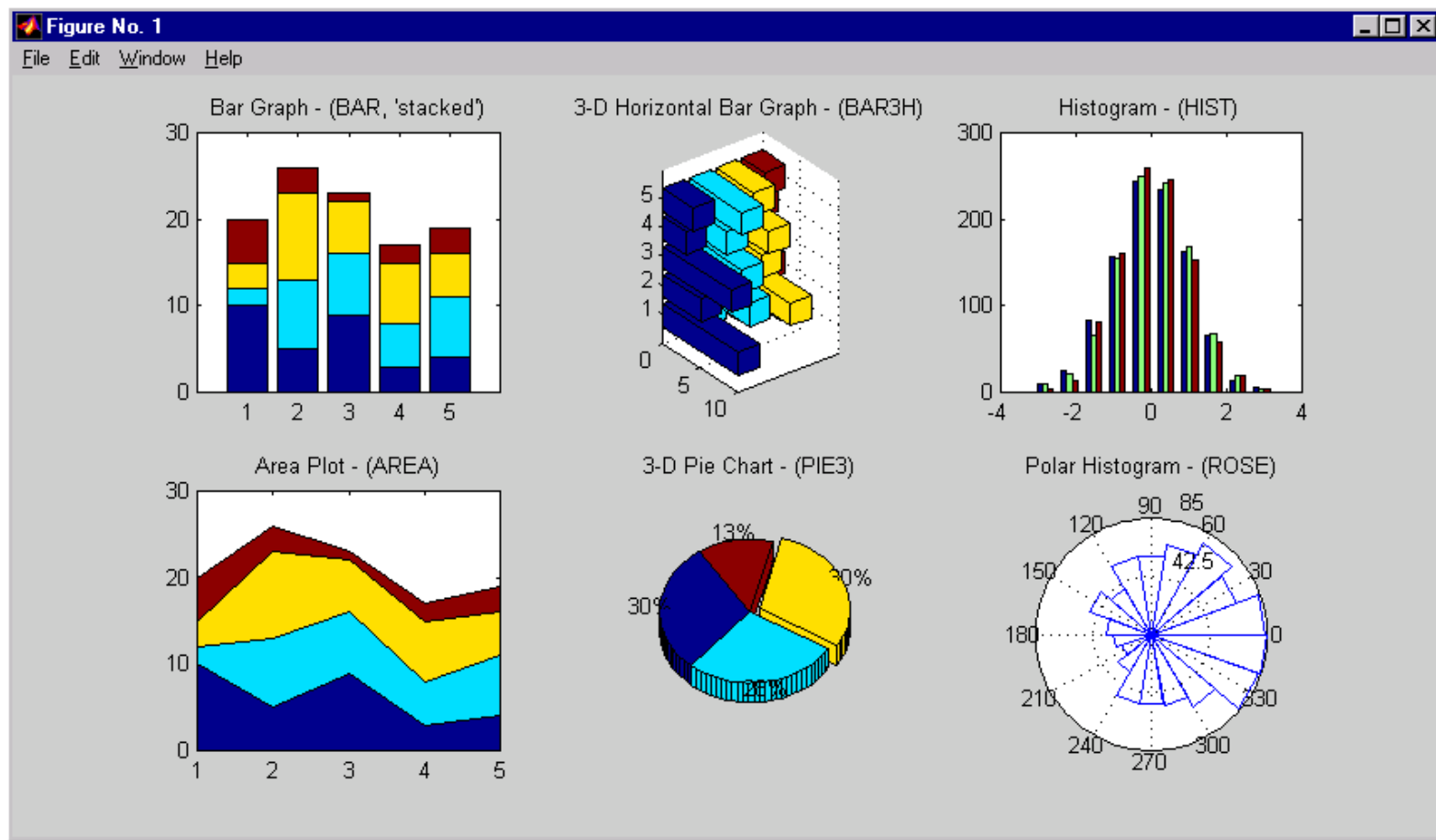
*contourf-colorbar-plot3-waterfall-contour3-mesh-surf*



# פונקציות גרפיקה נוספות - 2

שרטוטים מיוחדים:

*bar-bar3h-hist-area-pie3-rose*



A decorative graphic consisting of a thin gold circle and a horizontal bar. The bar has a gold-to-white gradient and is flanked by large black and gold brackets. The title text is centered within the white portion of the bar.

# A Quick Tutorial on MATLAB

Gowtham Bellala

# [ MATLAB ]

---

- MATLAB is a software package for doing numerical computation. It was originally designed for solving linear algebra type problems using matrices. It's name is derived from MATrix LABoratory.
- MATLAB has since been expanded and now has built-in functions for solving problems requiring data analysis, signal processing, optimization, and several other types of scientific computations. It also contains functions for 2-D and 3-D graphics and animation.



# MATLAB Variable names

- Variable names are case sensitive.
- Variable names can contain up to 63 characters ( as of MATLAB 6.5 and newer).
- Variable names must start with a letter and can be followed by letters, digits and underscores.

Examples :

```
>> x = 2;
```

```
>> abc_123 = 0.005;
```

```
>> 1ab = 2;
```

Error: Unexpected MATLAB expression

# [ MATLAB Special Variables ]

- pi Value of  $\pi$
- eps Smallest incremental number
- inf Infinity
- NaN Not a number e.g. 0/0
- i and j  $i = j = \text{square root of } -1$
- realmin The smallest usable positive real number
- realmax The largest usable positive real number

# [ MATLAB Relational operators ]

- MATLAB supports six relational operators.

|                       |                       |
|-----------------------|-----------------------|
| Less Than             | <                     |
| Less Than or Equal    | <=                    |
| Greater Than          | >                     |
| Greater Than or Equal | >=                    |
| Equal To              | ==                    |
| Not Equal To          | ~= (NOT != like in C) |

# [ MATLAB Logical Operators ]

MATLAB supports three logical operators.

|     |   |                             |
|-----|---|-----------------------------|
| not | ~ | % highest precedence        |
| and | & | % equal precedence with or  |
| or  |   | % equal precedence with and |

A decorative graphic consisting of a thin gold circle and a horizontal bar with a gold-to-white gradient. A large black left square bracket is on the left, and a gold right square bracket is on the right.

# Matrices and MATLAB

# [ MATLAB Matrices ]

---

- MATLAB treats all variables as matrices. For our purposes a matrix can be thought of as an array, in fact, that is how it is stored.
- Vectors are special forms of matrices and contain only one row OR one column.
- Scalars are matrices with only one row AND one column

# [Generating Matrices]

- A scalar can be created in MATLAB as follows:

```
>> x = 23;
```

- A matrix with only one row is called a row vector. A row vector can be created in MATLAB as follows (note the commas):

```
>> y = [12, 10, -3]
```

```
y =
```

```
 12 10 -3
```

- A matrix with only one column is called a column vector. A column vector can be created in MATLAB as follows:

```
>> z = [12; 10; -3]
```

```
z =
```

```
 12
```

```
 10
```

```
 -3
```

# [ Generating Matrices ]

- MATLAB treats row vector and column vector very differently
- A matrix can be created in MATLAB as follows (note the commas and semicolons)

```
>> X = [1,2,3;4,5,6;7,8,9]
```

```
X =
```

|   |   |   |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

Matrices must be rectangular!



# [ The Matrix in MATLAB ]

|          |   | Columns<br>(n)   |                  |                  |                 |                  |
|----------|---|------------------|------------------|------------------|-----------------|------------------|
|          |   | 1                | 2                | 3                | 4               | 5                |
| Rows (m) | 1 | 4 <sup>1</sup>   | 10 <sup>6</sup>  | 1 <sup>11</sup>  | 6 <sup>16</sup> | 2 <sup>21</sup>  |
|          | 2 | 8 <sup>2</sup>   | 1.2 <sup>7</sup> | 9 <sup>12</sup>  | 4 <sup>17</sup> | 25 <sup>22</sup> |
|          | 3 | 7.2 <sup>3</sup> | 5 <sup>8</sup>   | 7 <sup>13</sup>  | 1 <sup>18</sup> | 11 <sup>23</sup> |
|          | 4 | 0 <sup>4</sup>   | 0.5 <sup>9</sup> | 4 <sup>14</sup>  | 5 <sup>19</sup> | 56 <sup>24</sup> |
|          | 5 | 23 <sup>5</sup>  | 83 <sup>10</sup> | 13 <sup>15</sup> | 0 <sup>20</sup> | 10 <sup>25</sup> |

A(2,4)

A(17)

Note: Unlike C, MATLAB's indices start from 1

# [Extracting a Sub-matrix]

- A portion of a matrix can be extracted and stored in a smaller matrix by specifying the names of both matrices and the rows and columns to extract. The syntax is:

```
sub_matrix = matrix (r1 : r2 , c1 : c2) ;
```

where **r1** and **r2** specify the beginning and ending rows and **c1** and **c2** specify the beginning and ending columns to be extracted to make the new matrix.

# [Extracting a Sub-matrix]

- Example :

```
>> X = [1,2,3;4,5,6;7,8,9]
```

```
X =
```

|   |   |   |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

```
>> X22 = X(1:2 , 2:3)
```

```
X22 =
```

|   |   |
|---|---|
| 2 | 3 |
| 5 | 6 |

```
>> X13 = X(3,1:3)
```

```
X13 =
```

|   |   |   |
|---|---|---|
| 7 | 8 | 9 |
|---|---|---|

```
>> X21 = X(1:2,1)
```

```
X21 =
```

|   |
|---|
| 1 |
| 4 |

# [ Matrix Extension ]

- ```
>> a = [1,2i,0.56]
a =
     1     0+2i     0.56
>> a(2,4) = 0.1
a =
     1     0+2i     0.56     0
     0         0         0     0.1
```

- **repmat** – replicates and tiles a matrix

```
>> b = [1,2;3,4]
b =
     1     2
     3     4
>> b_rep = repmat(b,1,2)
b_rep =
     1     2     1     2
     3     4     3     4
```

- **Concatenation**

```
>> a = [1,2;3,4]
a =
     1     2
     3     4
>> a_cat = [a,2*a;3*a,2*a]
a_cat =
     1     2     2     4
     3     4     6     8
     3     6     2     4
     9    12     6     8
```

NOTE: The resulting matrix must be rectangular

[Matrix Addition]

- Increment all the elements of a matrix by a single value

```
>> x = [1,2;3,4]
```

```
x =
```

```
    1    2  
    3    4
```

```
>> y = x + 5
```

```
y =
```

```
    6    7  
    8    9
```

- Adding two matrices

```
>> xsy = x + y
```

```
xsy =
```

```
    7    9  
   11   13
```

```
>> z = [1,0.3]
```

```
z =
```

```
    1    0.3
```

```
>> xsz = x + z
```

??? Error using => plus
Matrix dimensions must
agree

[Matrix Multiplication]

■ Matrix multiplication

```
>> a = [1,2;3,4];      (2x2)
```

```
>> b = [1,1];          (1x2)
```

```
>> c = b*a
```

```
c =
```

```
     4     6
```

```
>> c = a*b
```

```
??? Error using ==> mtimes
Inner matrix dimensions
must agree.
```

■ Element wise multiplication

```
>> a = [1,2;3,4];
```

```
>> b = [1,1/2;1/3,1/4];
```

```
>> c = a.*b
```

```
c =
```

```
     1     1
```

```
     1     1
```

Matrix Element wise operations

- ```
>> a = [1,2;1,3];
>> b = [2,2;2,1];
```

- Element wise division

```
>> c = a./b
```

```
c =
```

```
 0.5 1
 0.5 3
```

- Element wise multiplication

```
>> c = a.*b
```

```
c =
```

```
 2 4
 2 3
```

- Element wise power operation

```
>> c = a.^2
```

```
c =
```

```
 1 4
 1 9
```

```
>> c = a.^b
```

```
c =
```

```
 1 4
 1 3
```

# [ Matrix Manipulation functions ]

- zeros : creates an array of all zeros, Ex: `x = zeros(3,2)`
- ones : creates an array of all ones, Ex: `x = ones(2)`
- eye : creates an identity matrix, Ex: `x = eye(3)`
- rand : generates uniformly distributed random numbers in [0,1]
- diag : Diagonal matrices and diagonal of a matrix
- size : returns array dimensions
- length : returns length of a vector (row or column)
- det : Matrix determinant
- inv : matrix inverse
- eig : evaluates eigenvalues and eigenvectors
- rank : rank of a matrix
- find : searches for the given values in an array/matrix.





# MATLAB inbuilt math functions

# [Elementary Math functions]

- `abs` - finds absolute value of all elements in the matrix
- `sign` - signum function
- `sin,cos,...` - Trigonometric functions
- `asin,acos...` - Inverse trigonometric functions
- `exp` - Exponential
- `log,log10` - natural logarithm, logarithm (base 10)
- `ceil,floor` - round towards +infinity, -infinity respectively
- `round` - round towards nearest integer
- `real,imag` - real and imaginary part of a complex matrix
- `sort` - sort elements in ascending order

# [ Elementary Math functions ]

- sum,prod - summation and product of elements
- max,min - maximum and minimum of arrays
- mean,median – average and median of arrays
- std,var - Standard deviation and variance

*and many more...*

A decorative graphic consisting of a thin gold circle. A horizontal bar, colored with a gradient from olive green on the left to light yellow on the right, passes through the center of the circle. A large black left square bracket is positioned on the left side of the bar, and a large gold right square bracket is on the right side.

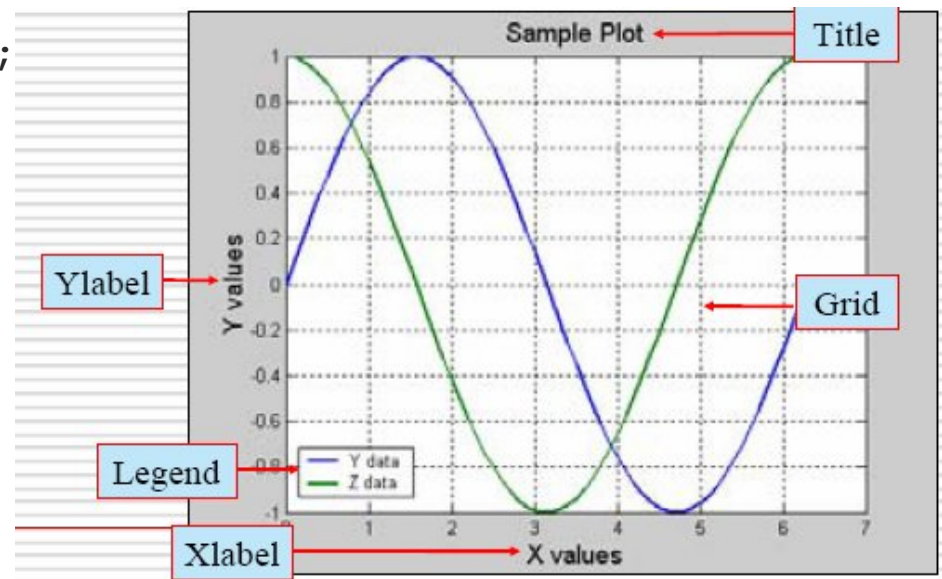
# Graphics Fundamentals

# [ 2D Plotting ]

- Example 1: Plot  $\sin(x)$  and  $\cos(x)$  over  $[0, 2\pi]$ , on the same plot with different colours

Method 1:

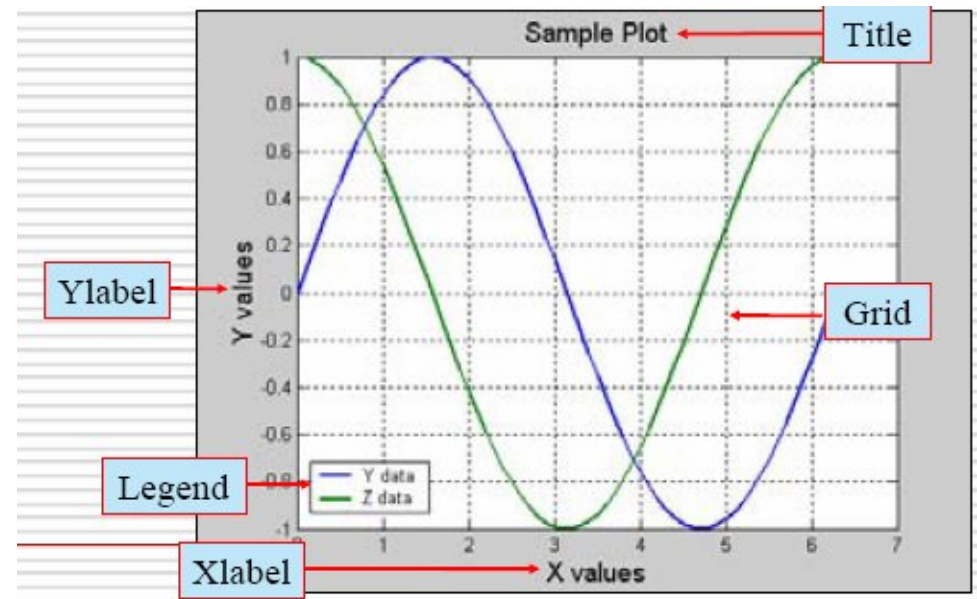
```
>> x = linspace(0,2*pi,1000);
>> y = sin(x);
>> z = cos(x);
>> hold on;
>> plot(x,y,'b');
>> plot(x,z,'g');
>> xlabel 'X values';
>> ylabel 'Y values';
>> title 'Sample Plot';
>> legend ('Y data','Z data');
>> hold off;
```



# [ 2D Plotting ]

Method 2:

```
>> x = 0:0.01:2*pi;
>> y = sin(x);
>> z = cos(x);
>> figure
>> plot (x,y,x,z);
>> xlabel 'X values';
>> ylabel 'Y values';
>> title 'Sample Plot';
>> legend ('Y data','Z data');
>> grid on;
```

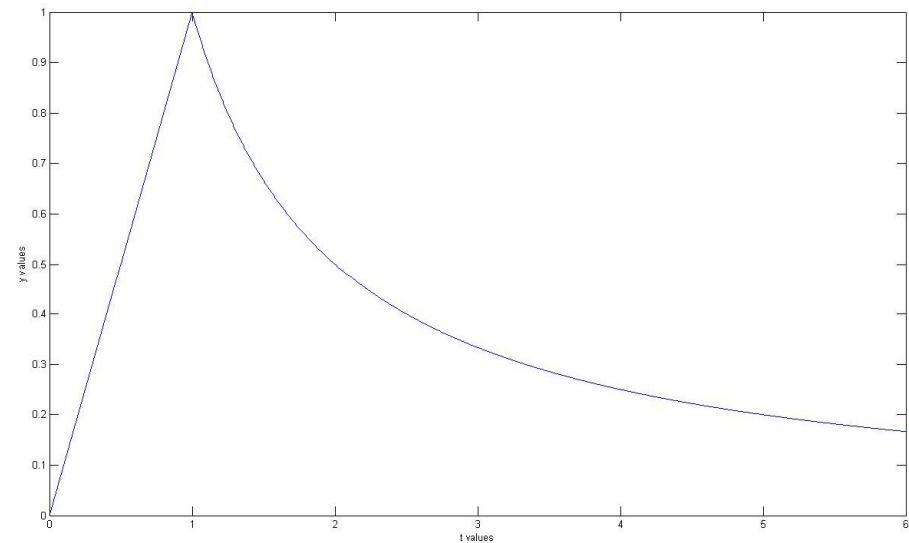


# [ 2D Plotting ]

- Example 2: Plot the following function  $y = \begin{cases} t & 0 \leq t \leq 1 \\ 1/t & 1 \leq t \leq 6 \end{cases}$

Method 1:

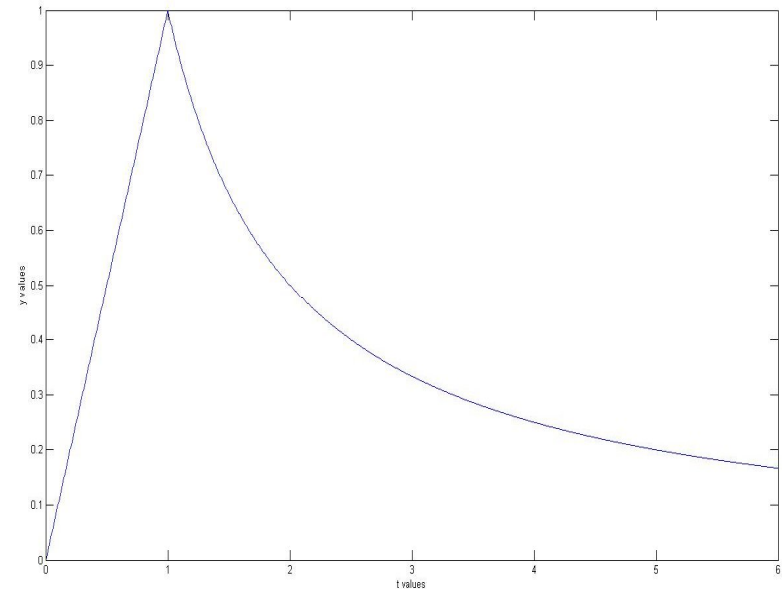
```
>> t1 = linspace(0,1,1000);
>> t2 = linspace(1,6,1000);
>> y1 = t1;
>> y2 = 1./ t2;
>> t = [t1,t2];
>> y = [y1,y2];
>> figure
>> plot(t,y);
>> xlabel 't values', ylabel 'y values';
```



# [ 2D Plotting ]

Method 2:

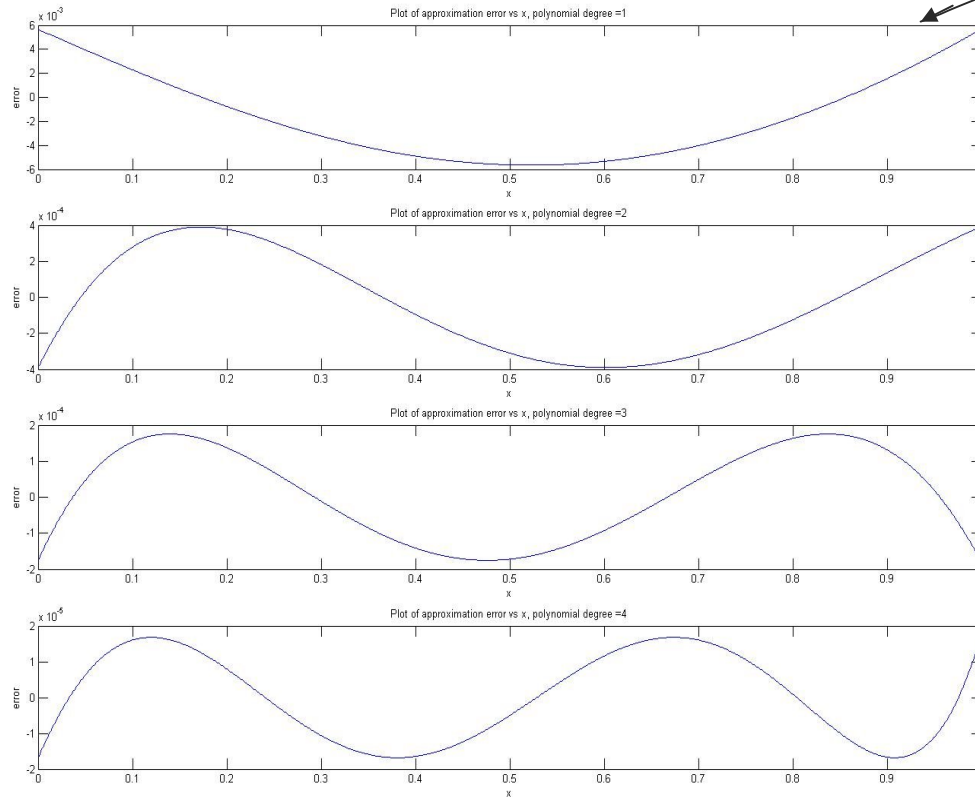
```
>> t = linspace(0,6,1000);
>> y = zeros(1,1000);
>> y(t()<=1) = t(t()<=1);
>> y(t()>1) = 1./ t(t()>1);
>> figure
>> plot(t,y);
>> xlabel't values';
>> ylabel'y values';
```





# [ Subplots ]

- Syntax: subplot (rows, columns, index)



>> subplot(4,1,1)

>> ...

>> subplot(4,1,2)

>> ...

>> subplot(4,1,3)

>> ...

>> subplot(4,1,4)

>> ...



Importing/Exporting Data

# [ Load and Save ]

- Using load and save

load filename - loads all variables from the file “filename”

load filename x - loads only the variable x from the file

load filename a\* - loads all variables starting with ‘a’

*for more information, type **help load** at command prompt*

save filename - saves all workspace variables to a binary  
.mat file named filename.mat

save filename x,y - saves variables x and y in filename.mat

*for more information, type **help save** at command prompt*

# [ Import/Export from Excel sheet ]

- Copy data from an excel sheet

```
>> x = xlsread(filename);
```

% if the file contains numeric values, text and raw data values, then

```
>> [numeric,txt,raw] = xlsread(filename);
```

- Copy data to an excel sheet

```
>>x = xlswrite('c:\matlab\work\data.xls',A,'A2:C4')
```

% will write A to the workbook file, data.xls, and attempt to fit the elements of A into the rectangular worksheet region, A2:C4. On success, 'x' will contain '1', while on failure, 'x' will contain '0'.

*for more information, type **help xlswrite** at command prompt*

# [ Read/write from a text file ]

- Writing onto a text file

```
>> fid = fopen('filename.txt','w');
>> count = fwrite(fid,x);
>> fclose(fid);
```

% creates a file named 'filename.txt' in your workspace and stores the values of variable 'x' in the file. 'count' returns the number of values successfully stored. **Do not forget to close the file at the end.**

- Read from a text file

```
>> fid = fopen('filename.txt','r');
>> X = fscanf(fid,'%5d');
>> fclose(fid);
```

% opens the file 'filename.txt' which is in your workspace and loads the values in the format '%5d' into the variable x.

*Other useful commands: fread, fprintf*



# Flow Control in MATLAB

# [ Flow control ]

---

- MATLAB has five flow control statements
  - **if** statements
  - **switch** statements
  - **for** loops
  - **while** loops
  - **break** statements

# [ 'if' statement ]

- The general form of the 'if' statement is

```
>> if expression
>> ...
>> elseif expression
>> ...
>> else
>> ...
>> end
```

- Example 1:

```
>> if i == j
>> a(i,j) = 2;
>> elseif i >= j
>> a(i,j) = 1;
>> else
>> a(i,j) = 0;
>> end
```

- Example 2:

```
>> if (attn>0.9)&(grade>60)
>> pass = 1;
>> end
```



# [ 'switch' statement ]

- **switch** Switch among several cases based on expression

- The general form of the **switch** statement is:

```
>> switch switch_expr
>> case case_expr1
>> ...
>> case case_expr2
>> ...
>> otherwise
>> ...
>> end
```

- Example :

```
>> x = 2, y = 3;
>> switch x
>> case x==y
>> disp('x and y are equal');
>> case x>y
>> disp('x is greater than y');
>> otherwise
>> disp('x is less than y');
>> end
x is less than y
```

**Note: Unlike C, MATLAB doesn't need BREAKs in each case**

# [ 'for' loop ]

- **for** Repeat statements a specific number of times

- The general form of a **for** statement is

```
>> for variable=expression
>> ...
>> ...
>> end
```

- Example 1:

```
>> for x = 0:0.05:1
>> printf('%d\n',x);
>> end
```

- Example 2:

```
>> a = zeros(n,m);
>> for i = 1:n
>> for j = 1:m
>> a(i,j) = 1/(i+j);
>> end
>> end
```

# [ 'while' loop ]

- **while** Repeat statements an indefinite number of times
- The general form of a **while** statement is

```
>> while expression
>> ...
>> ...
>> end
```

- Example 1:  

```
>> n = 1;
>> y = zeros(1,10);
>> while n <= 10
>> y(n) = 2*n/(n+1);
>> n = n+1;
>> end
```

- Example 2:  

```
>> x = 1;
>> while x
>> %execute statements
>> end
```

**Note:** In MATLAB '1' is synonymous to TRUE and '0' is synonymous to 'FALSE'

# [ 'break' statement ]

- **break** terminates the execution of **for** and **while** loops
- In nested loops, **break** terminates from the innermost loop only

- Example:

```
>> y = 3;
>> for x = 1:10
>> printf('%5d' ,x);
>> if (x>y)
>> break;
>> end
>> end
1 2 3 4
```

A decorative graphic consisting of a thin gold circle. A thick black left square bracket is positioned on the left side of the circle, and a thick gold right square bracket is on the right side. A horizontal bar with a gold-to-white gradient is placed across the middle of the circle, containing the text "Efficient Programming".

# Efficient Programming

# [ Efficient Programming in MATLAB ]

- Avoid using nested loops as far as possible
- In most cases, one can replace nested loops with efficient matrix manipulation.
- Preallocate your arrays when possible
- MATLAB comes with a huge library of in-built functions, use them when necessary
- Avoid using your own functions, MATLAB's functions are more likely to be efficient than yours.

# [ Example 1 ]

- Let  $x[n]$  be the input to a non causal FIR filter, with filter coefficients  $h[n]$ . Assume both the input values and the filter coefficients are stored in column vectors  $x, h$  and are given to you. Compute the output values  $y[n]$  for  $n = 1, 2, 3$  where

$$y[n] = \sum_{k=0}^{19} h[k]x[n+k]$$

# [Solution]

## ■ Method 1:

```
>> y = zeros(1,3);
>> for n = 1:3
>> for k = 0:19
>> y(n) = y(n) + h(k) * x(n+k);
>> end
>> end
```

## ■ Method 2 (avoids inner loop):

```
>> y = zeros(1,3);
>> for n = 1:3
>> y(n) = h' * x(n:(n+19));
>> end
```

## ■ Method 3 (avoids both the loops):

```
>> X = [x(1:20), x(2:21), x(3:22)];
>> y = h' * X;
```



# [ Example 2 ]

- Compute the value of the following function

$$y(n) = 1^3 \cdot (1^3 + 2^3) \cdot (1^3 + 2^3 + 3^3) \cdot \dots \cdot (1^3 + 2^3 + \dots + n^3)$$

for  $n = 1$  to  $20$

# [Solution]

## ■ Method 1:

```
>> y = zeros(20,1);
>> y(1) = 1;
>> for n = 2:20
>> for m = 1:n
>> temp = temp + m^3;
>> end
>> y(n) = y(n-1)*temp;
>> temp = 0
>> end
```

## ■ Method 2 (avoids inner loop):

```
>> y = zeros(20,1);
>> y(1) = 1;
>> for n = 2:20
>> temp = 1:n;
>> y(n) = y(n-1)*sum(temp.^3);
>> end
```

## ■ Method 3 (avoids both the loops):

```
>> X = tril(ones(20)*diag(1:20));
>> x = sum(X.^3,2);
>> Y = tril(ones(20)*diag(x))+ ...
 triu(ones(20)) - eye(20);
>> y = prod(Y,2);
```

# [ Getting more help ]

Where to get help?

- In MATLAB's prompt type :  
**help, lookfor, helpwin, helpdesk, demos**
- On the Web :  
<http://www.mathworks.com/support>  
<http://www.mathworks.com/products/demos/#>  
<http://www.math.siu.edu/MATLAB/tutorials.html>  
<http://math.ucsd.edu/~driver/21d-s99/MATLAB-primer.html>  
<http://www.mit.edu/~pwb/cssm/>  
<http://www.eecs.umich.edu/~aey/eecs216/.html>