Introduction to Matlab & Data Analysis

Lecture 1: Introduction

Lecture time:
Sunday 11:15-13:00
FGS C

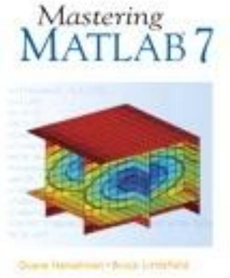
Team members

- **Lecturers:**
- Natalie Kalev-Kronik kalev001@umn.edu
- Anat Tzimmer

- **Guest Lecturers:**

- **Tutors:**
- Anat Tzimmer
- Gil Farkash
- Ayelet Sarel

- **Exercise checker:**
- Gil Farkash
- Ayelet Sarel



Tips / formalities

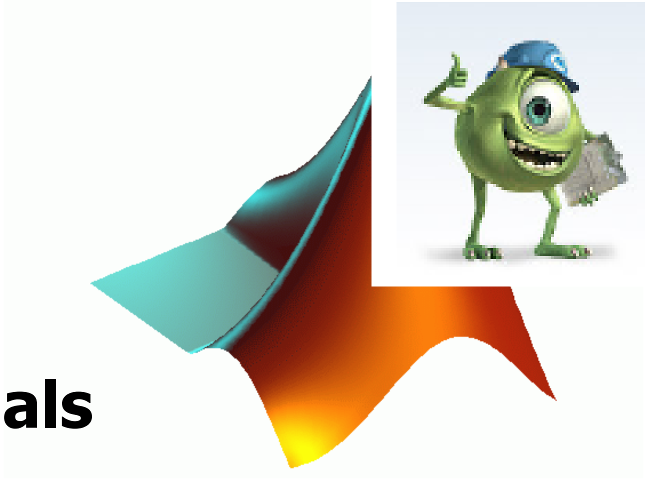
- **Course website**
  `http://www.weizmann.ac.il/midrasha/courses/MatlabIntro`

- **The website contains**
    - Course material: Lectures + tutorials + other Matlab resources
    - HW and solutions
    - News
- **Where can I do the HW?**
    - On any pc computer at Weizmann (installation of Matlab will be discussed in the first tutorial)
    - In the tutorial class

- **Grade**
    - HWs 60% + 40% Final Project

- **Course references**
    - Matlab built-in tutorials and references

Tips / formalities



- **Signing up for one of the tutorials**

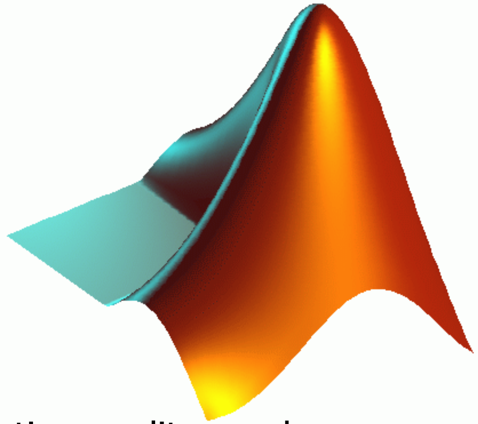
  Feinberg B
  (#1) Sunday 9:15-11
  (#2) Wednesday 9:15-11
  There may be some changes during the semester

- **HW assistance at the computer room**

  Once a week in Feinberg B

  With Ex. Checker

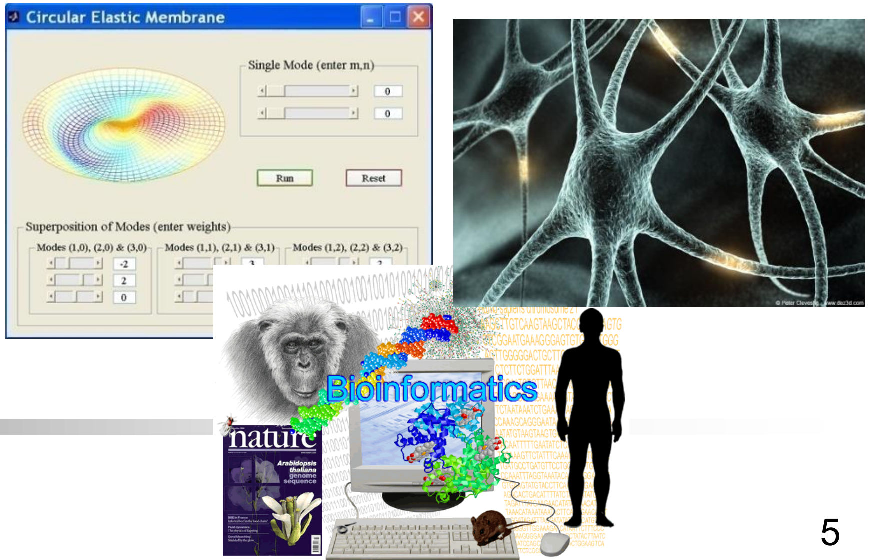
Course overview



- Introduction to Matlab
- Matlab building blocks: 1D 2D and 3D arrays
- Simple data analysis and graphics
- Control and boolean logic
- Loops
- Functions and program design
- Cells, structures and Files
- Simple algorithms and complexity
- Debugger
- GUI toolbox

- Producing publication quality graphs
- Solving ODEs for a living:

Math modeling of cancer treatment (Natalie)


Circular Elastic Membrane
Single Mode (enter m,n)
0
0
Run
Reset
Superposition of Modes (enter weights)
Modes (1,0), (2,0) & (3,0)
Modes (1,1), (2,1) & (3,1)
Modes (1,2), (2,2) & (3,2)
-2
2
0
Bioinformatics
nature

For whom is the course intended?

- For students with no or little experience of Matlab- first two thirds of the course.
- Please note that the workload is heavy and each assignment may take a few hours.
- Submit HW with a study partner.
- Some overlap or unsynchronized material may occur (lecture, tutorial, HW).

What is the course about?

(1) Programming in Matlab

(2) Tackling data analysis problems with *Matlab*

What is the course about?

Example #1 of a data analysis problem

```
CAGCATATTTGAAGCCGGGCCCACACACAATTGGGGAACGGATCCCCGCGGCCTCCCGGCA
GACCCCGTCCGGCACGACGACGAAGAAGGGGAGGATGAAGTCGAATTTGAAGCGGATGAAG
GATGAGGAGAGTGACGAAGAAGAGGACGAAGACGACGAGGTCCTTGACGAGGAAGTGAACT
ATTGAATTTGAAGCTTATTCCATCTCAGATAATGATTATGACGGAATTAAGAAATTACTAG
CAGCAGCTTTTCCTAAAGGCTCCTGTGAACACTGCAGAACTAACAGATCTCTTAATTCATA
CAGAACCATATTGGAAGTGTGAATTTGAAGCTTAAGCAAACAAATGTTTCAGAAGACAGCG
ATGATGATGATGCAGATGAAGATGAAATTTTTGGTTTCATAAGCCTTTTAAATTTAACTGA
AAGAAAGGTACCCAGTGTGCTGAACAAATTAAAGAGTTGGTATTTGAAGCGGGTGAGAAGA
ACTGTAAAGAATTTGAAGCGGCAGCTGGACAAGCTTTTAAATGACACCACCAAGCCTGTGG
GCTTTCTCCTAAGTGAAAGATTCATTAATGTCCCTCCTCAGATTGCTCTGCCCATGCACCA
GCAGCTTCAGAAAGAATTTGAAGCAATTTGAAGCCTAGTATTTGAAGCTTCTACCTTCTGA
GACCCCGTCCGGCACGACGACGAAGAAGGGGAGGATGAAGTCGAGGATGAAGACGAAGATC
GATGAGGAGAGTGACGAAGAAGAGGATTTGAAGCACGAAGACGACGAGGTCCTTGACGAGG
AAGTGAATATTGAATTTGAAGCTTATTCCATCTCAGATAATGATTATGACGGAATTAAGAA
ATTACTGCAGCAATTTGAAGCAAAGGCTCCTGTGAACACTGCAGATTTGAAGCAACTAACA
ATTCAACAGAACCATATTGGAAGTGTGATTAAGCAAACAAATGTTTCAGAAGACAGCGATG
ATGATGATGCATTTGAAGCAGATGAAGATGAAATTTTTGGTTTCATAAGCCTTTTAAATTT
CTAATAAGCCATGTGGGAAGTGCTCTTTCTACCTTATTTGAAGCACACCATTTGTGGAAGA
ATTACTGCAGCAATTTGAAGCAAAGGCTCCTGTGAACACTGCAGATTTGAAGCAACTAACA
```

What is the course about?

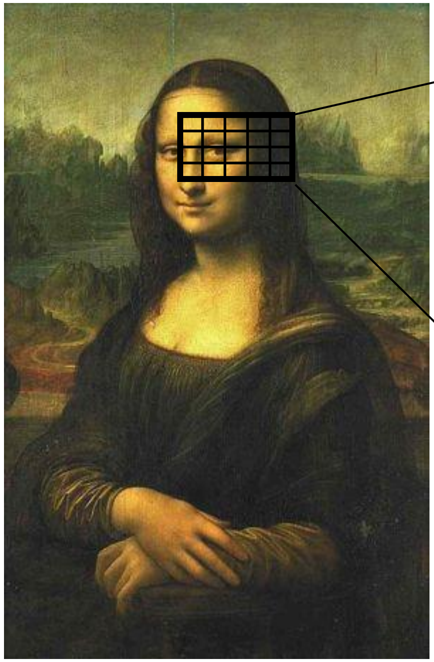
Example #1 of a data analysis problem

Identifying repeating motifs

CAGCAT**ATTTGAAGC**CGGGCCCACACACAATTGGGGAACGGATCCCCGCGGCCTCCCGGCA
GACCCCGTCCGGCACGACGACGAAGAAGGGGAGGATGAAGTCGA**ATTTGAAGC**GGATGAAG
GATGAGGAGAGTGACGAAGAAGAGGACGAAGACGACGAGGTCCTTGACGAGGAAGTGAACT
ATTGA**ATTTGAAGC**TTATTCCATCTCAGATAATGATTATGACGGAATTAAGAAATTACTAG
CAGCAGCTTTTCCTAAAGGCTCCTGTGAACACTGCAGAACTAACAGATCTCTTAATTCATA
CAGAACCATATTGGAAGTGTGA**ATTTGAAGC**TTAAGCAAACAAATGTTTCAGAAGACAGCG
ATGATGATGATGCAGATGAAGATGAAATTTTTGGTTTCATAAGCCTTTTAAATTTAACTGA
AAGAAAGGTACCCAGTGTGCTGAACAAATTAAAGAGTTGGT**ATTTGAAGCGGG**TGAGAAGA
ACTGTAAAGA**ATTTGAAGCGG**CAGCTGGACAAGCTTTTAAATGACACCACCAAGCCTGTGG
GCTTTCTCCTAAGTGAAAGATTCATTAATGTCCCTCCTCAGATTGCTCTGCCCATGCACCA
GCAGCTTCAGAAAGA**ATTTGAAGC**A**ATTTGAAGC**CTAGT**ATTTGAAGC**TTCTACCTTCTGA
GACCCCGTCCGGCACGACGACGAAGAAGGGGAGGATGAAGTCGAGGATGAAGACGAAGATC
GATGAGGAGAGTGACGAAGAAGAGG**ATTTGAAGC**ACGAAGACGACGAGGTCCTTGACGAGG
AAGTGAATATTGAATTTGAAGCTTATTCCATCTCAGATAATGATTATGACGGAATTAAGAA
ATTACTGCAGCA**ATTTGAAGC**AAAGGCTCCTGTGAACACTGCAG**ATTTGAAGC**AACTAACA
ATTCAACAGAACCATATTGGAAGTGTGATTAAGCAAACAAATGTTTCAGAAGACAGCGATG
ATGATGATGC**ATTTGAAGC**AGATGAAGATGAAATTTTTGGTTTCATAAGCCTTTTAAATTT
CTAATAAGCCATGTGGGAAGTGCTCTTTCTACCTT**ATTTGAAGC**ACACCATTTGTGGAAGA
ATTACTGCAGCA**ATTTGAAGC**AAAGGCTCCTGTGAACACTGCAG**ATTTGAAGC**AACTAACA

What is the course about?

Example #2 of a data analysis problem



10	21	10	21
73	21	18	21
10	4	8	21
3	21	10	45
8	21	2	21

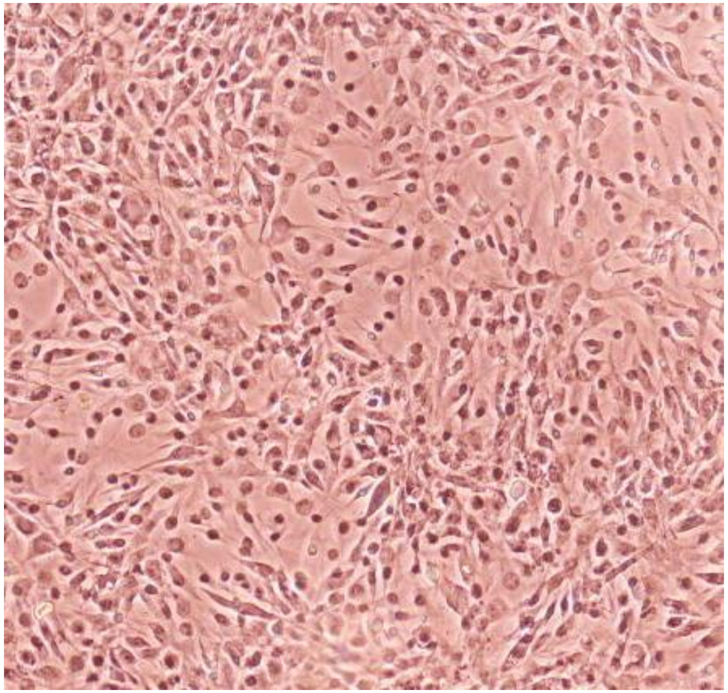
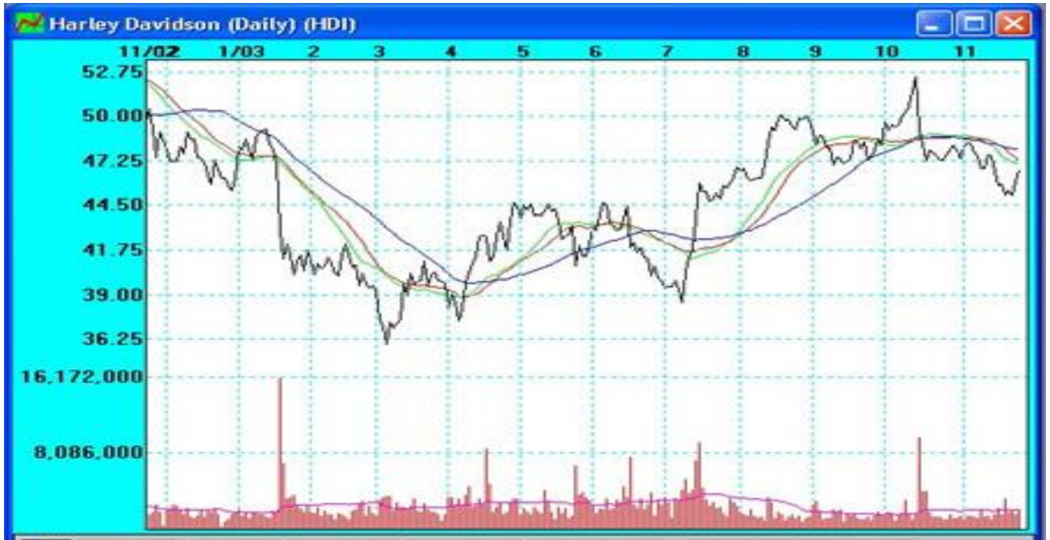


Image processing

What is the course about?

Example #3 of data analysis problems



11/25/2003 | 46.70 | 46.86 | 46.09 | 2,142,200 | TSO 153,355

Moving Avgs 25 Day 47.07 30 Day 47.24 52 Day

Signal processing

Adagio.
Beethoven, Op. 43.
Primo.
ff
ff
sf
pp
cresc.
sf
sf
p
p
sf
sf
sf
p
Allegro molto con brio.
pp
11
ff staccato

What is the course about?

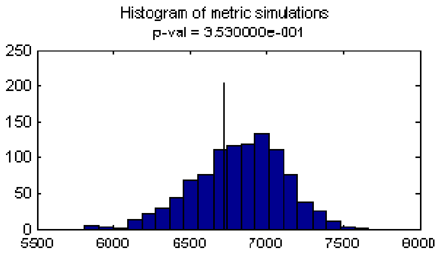
(1) Programming in Matlab

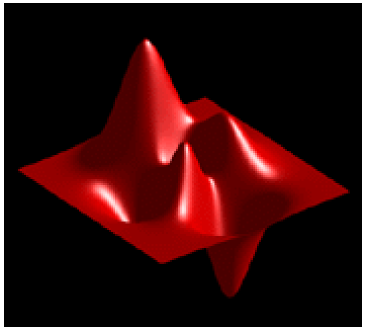
(2) Tackling data analysis problems with *Matlab*

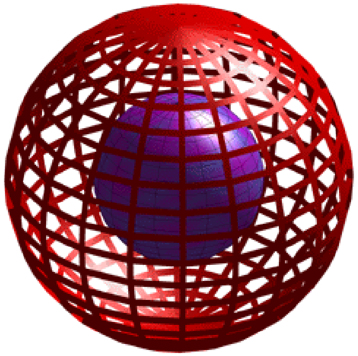
(3) Learn how to learn Matlab by yourself

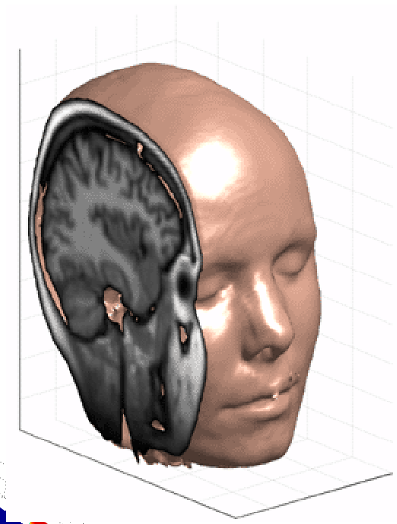
Why Matlab?

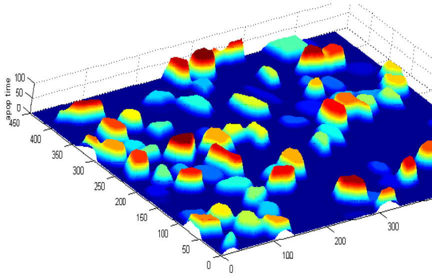
- Easy to learn
- Easy to debug
- Great tool for scientific work
  - Exploring your data
  - Visualizing your data
- Many useful "apps"







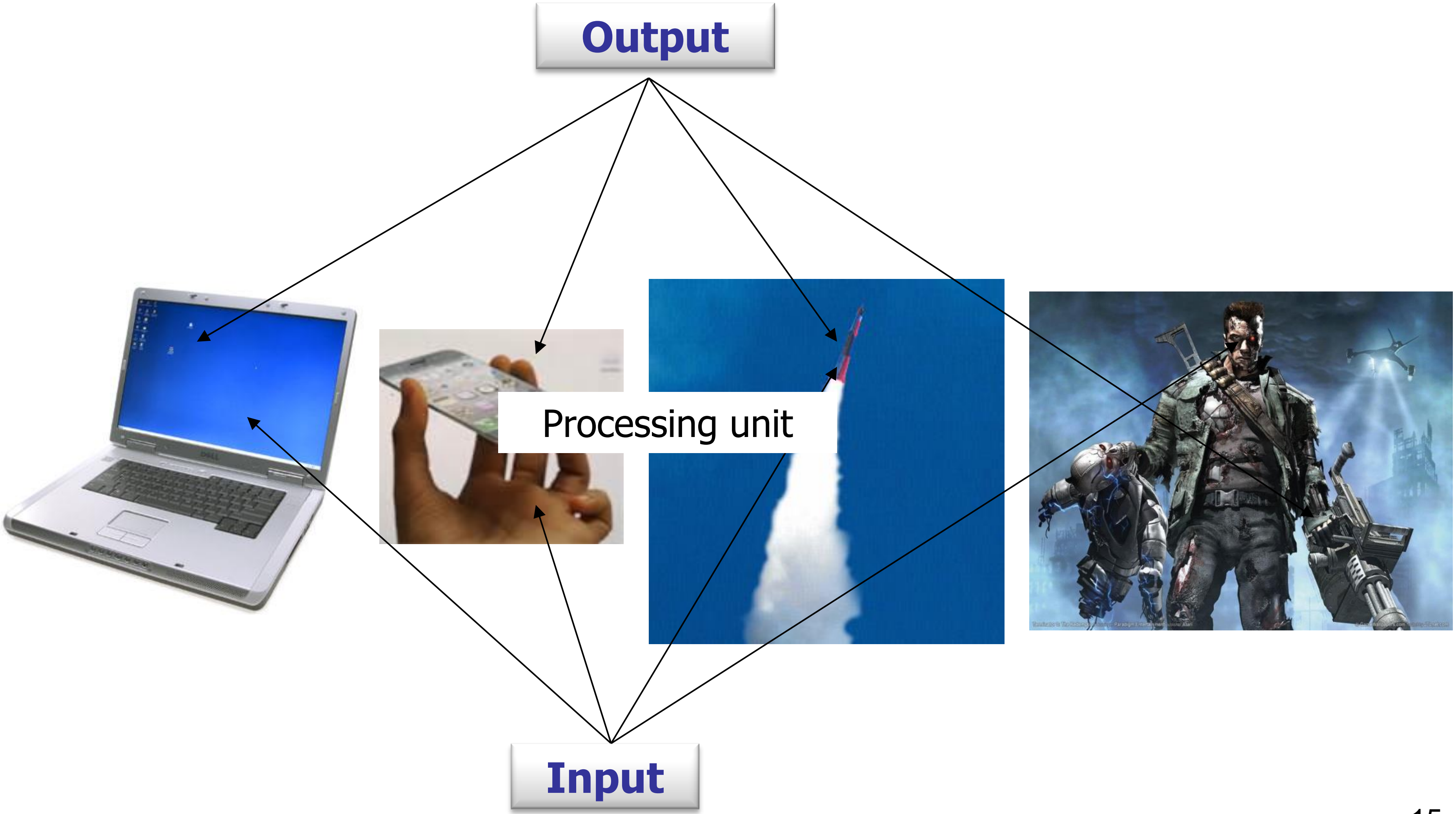




Matlab's main disadvantage…

- It's slower than other programming languages.
  - (unless you use the compiler)…

Background - computers

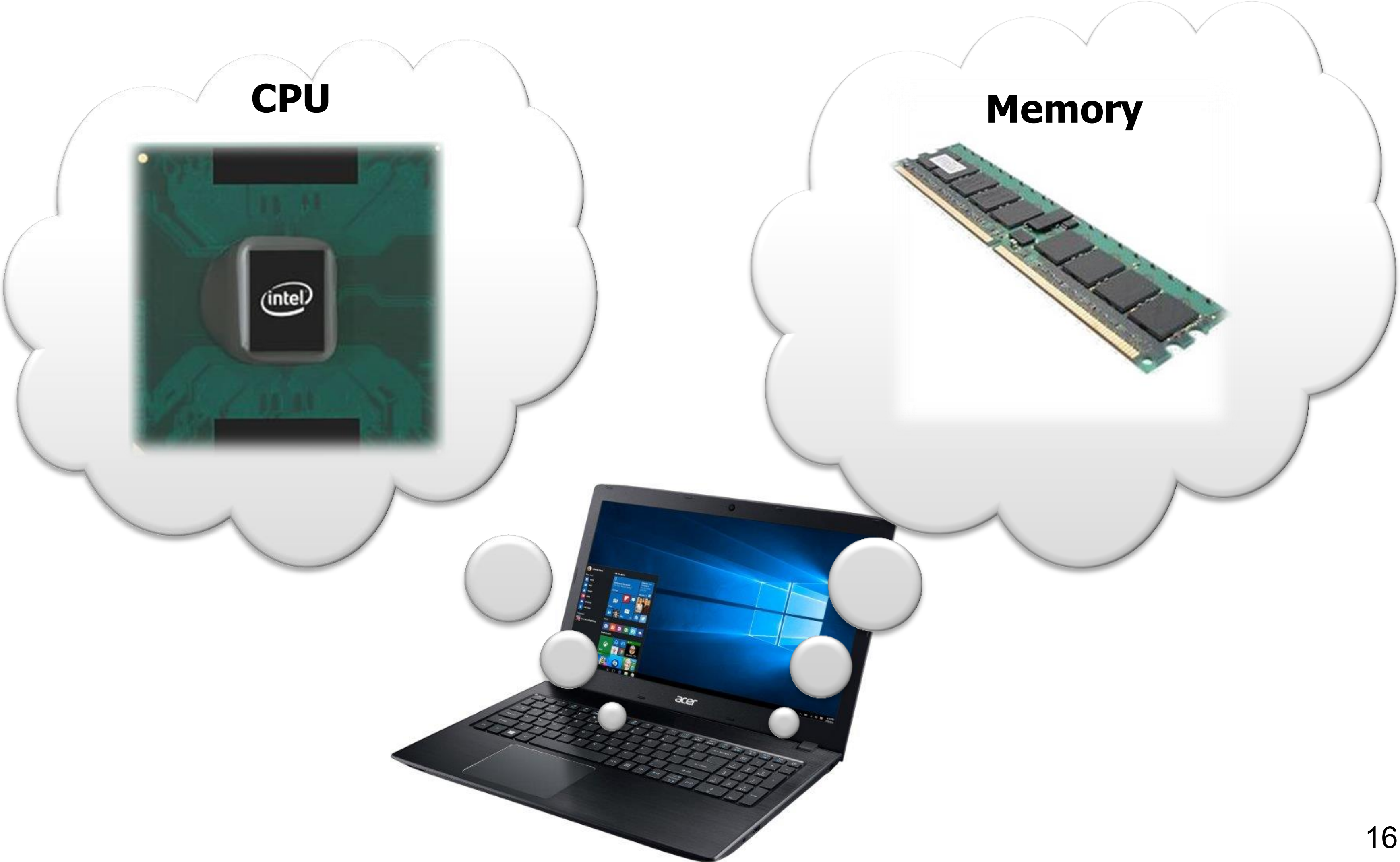


Output

Processing unit

Input

Background - hardware



CPU

Memory

Background - hardware

CPU

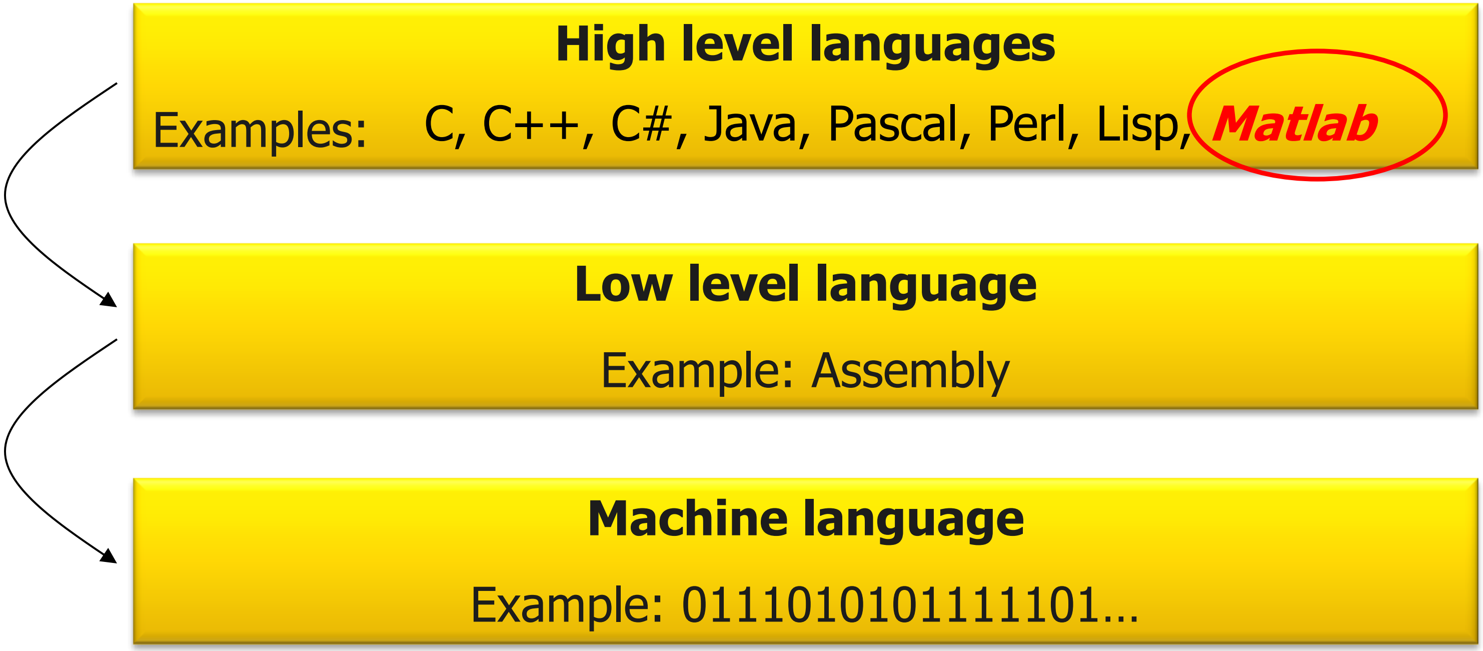
A **central processing unit** (**CPU**), is the hardware within a computer that carries out the instructions of a computer program by performing the basic arithmetical, logical, and input/output operations of the system. (Wikipedia).

Memory

In computing, **memory** refers to the physical devices used to store programs (sequences of instructions) or data (e.g. program state information) on a temporary or permanent basis for use in a computer or other digital electronic device. (Wikipedia).

Not to be confused with the data storage such as SSD and hard disk.

Background - software



High level languages

Examples: C, C++, C#, Java, Pascal, Perl, Lisp, ***Matlab***

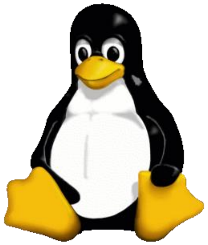
Low level language

Example: Assembly

Machine language

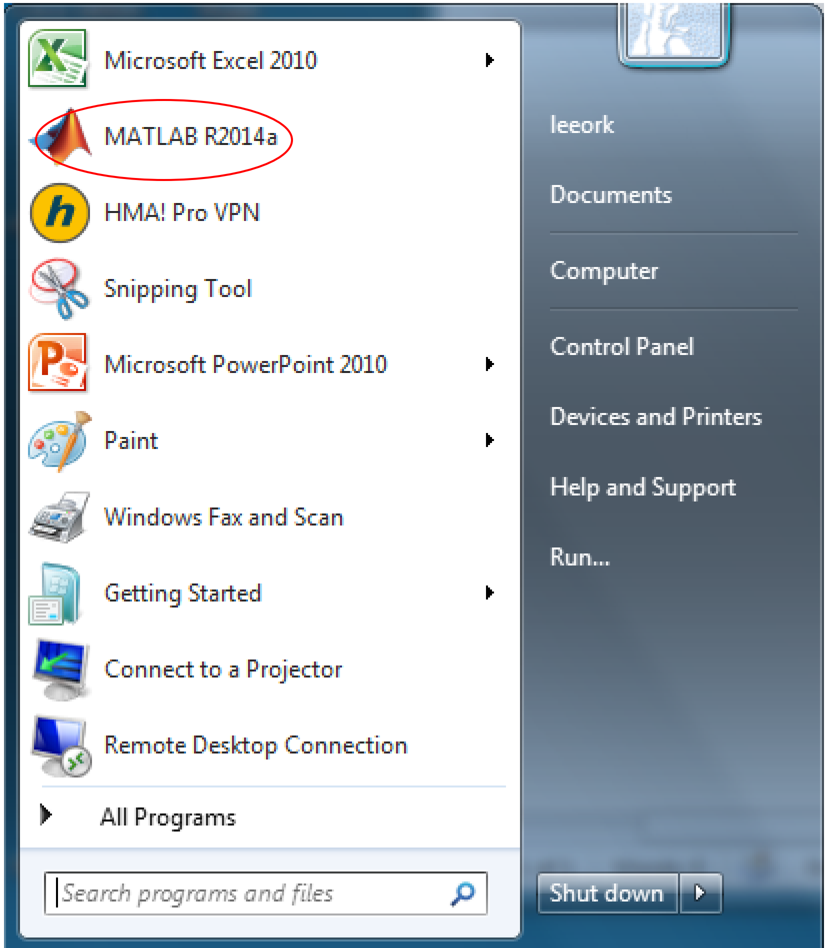
Example: 0111010101111101…

Another important player:
The operating system

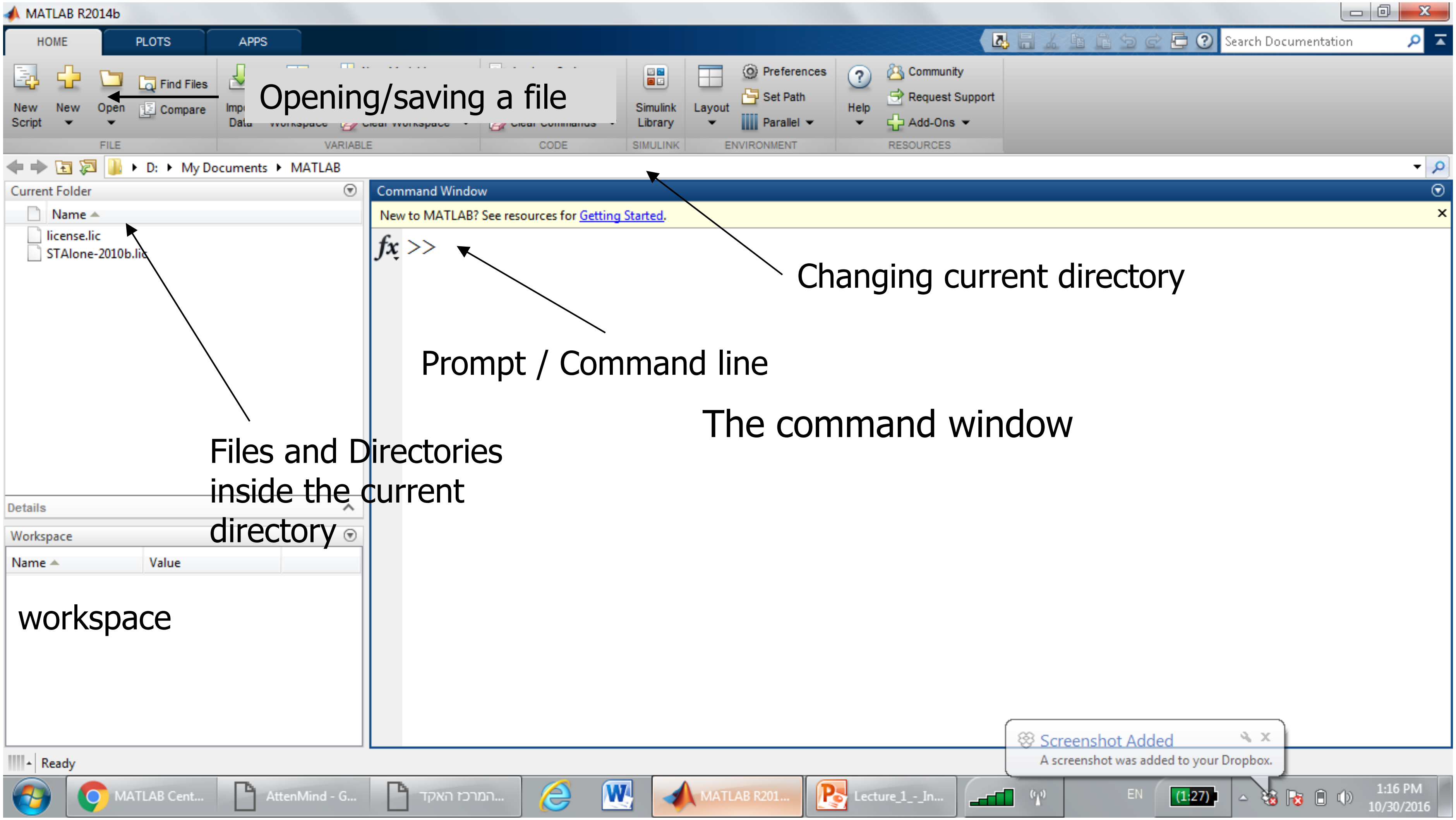


The Matlab environment

First we need to Open Matlab



The Matlab environment



New to MATLAB? See resources for Getting Started.

```
fx >>
```

Screenshot Added

A screenshot was added to your Dropbox.

Opening/saving a file

Changing current directory

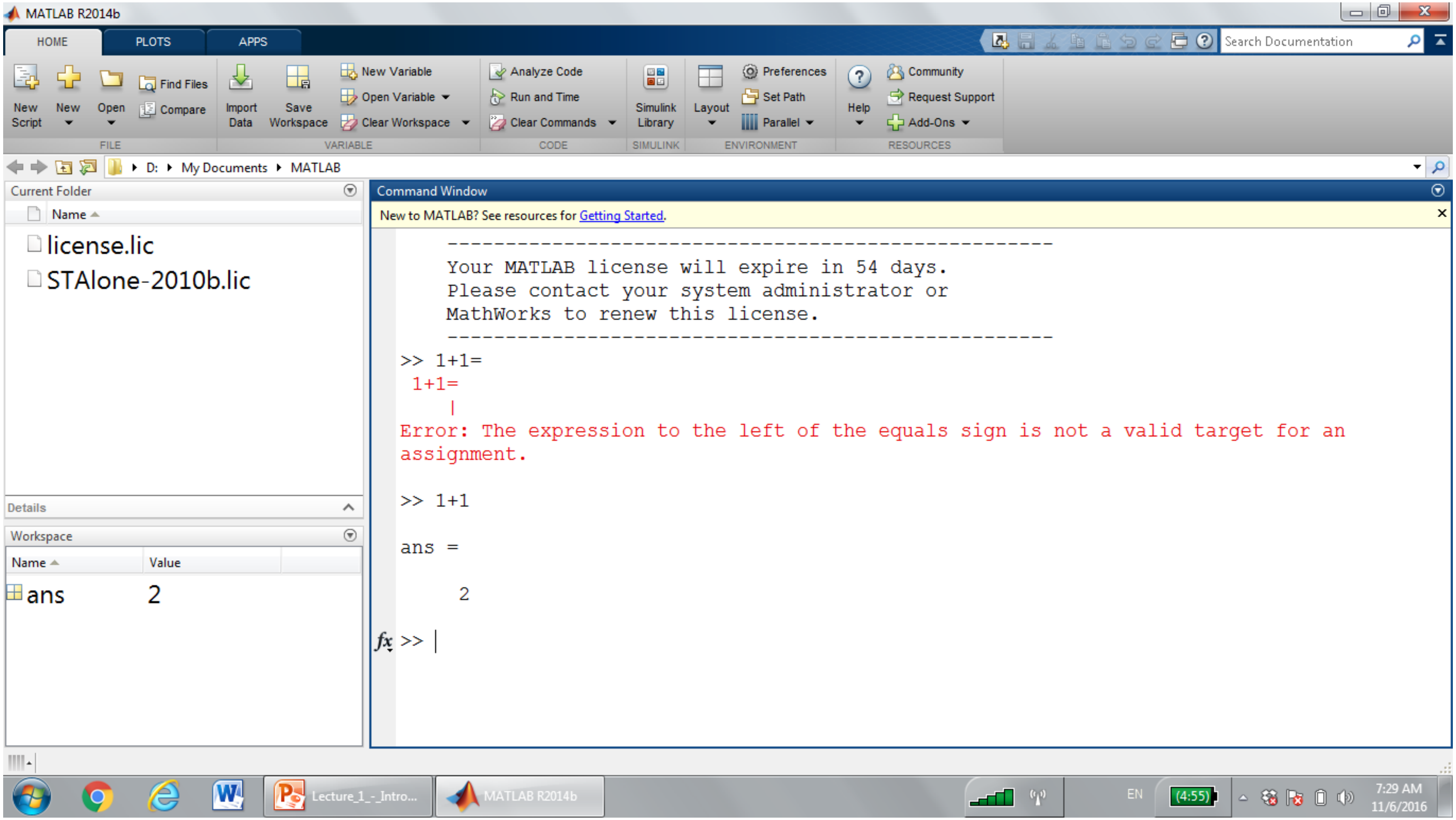
Prompt / Command line

The command window

Files and Directories inside the current directory

workspace

Matlab can be used as a calculator

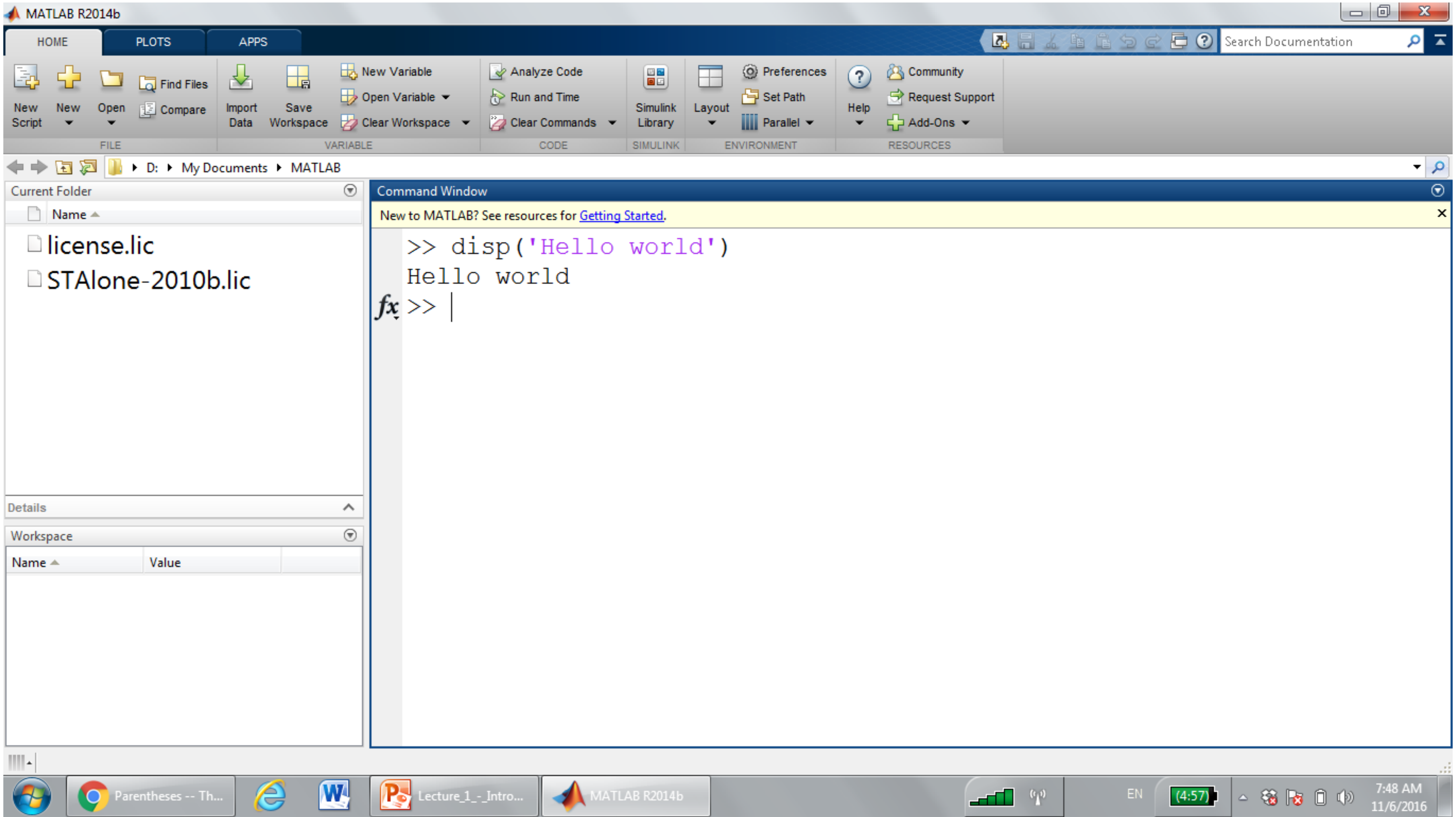


Name	Value
ans	2

```
    -----------------------------------------------------
    Your MATLAB license will expire in 54 days.
    Please contact your system administrator or
    MathWorks to renew this license.
    -----------------------------------------------------
>> 1+1=
 1+1=
     |
Error: The expression to the left of the equals sign is not a valid target for an
assignment.

>> 1+1

ans =

     2

>> |
```

Our first command

Writing a command in the command line



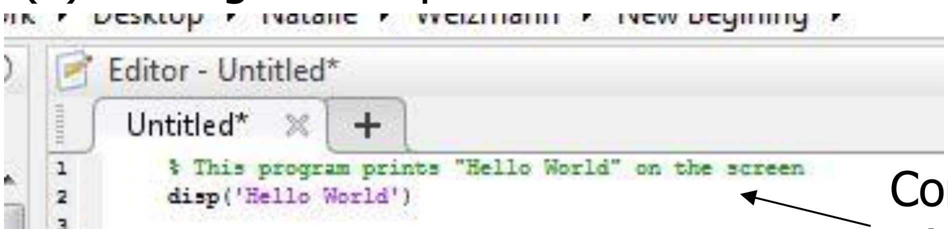
license.lic
STAlone-2010b.lic

New to MATLAB? See resources for Getting Started.

```
>> disp('Hello world')
Hello world
>> 
```

Our first script (M-file)

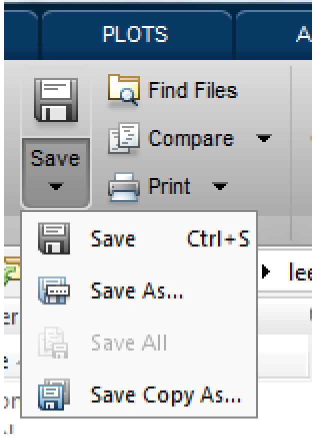
(1) Writing the script



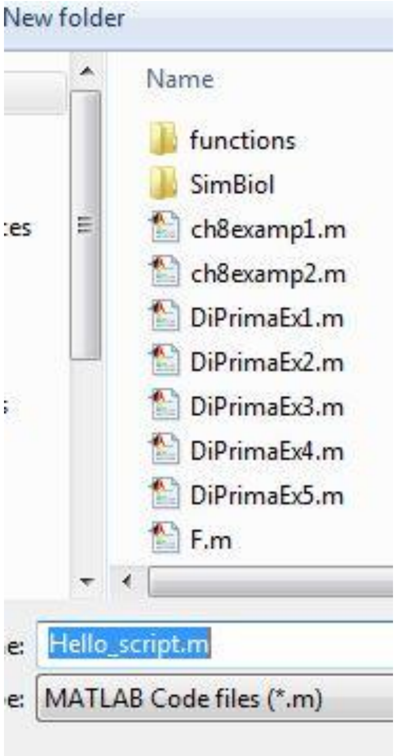
```
% This program prints "Hello World" on the screen
disp('Hello World')
```

Comments start with a %

(2) Saving the script



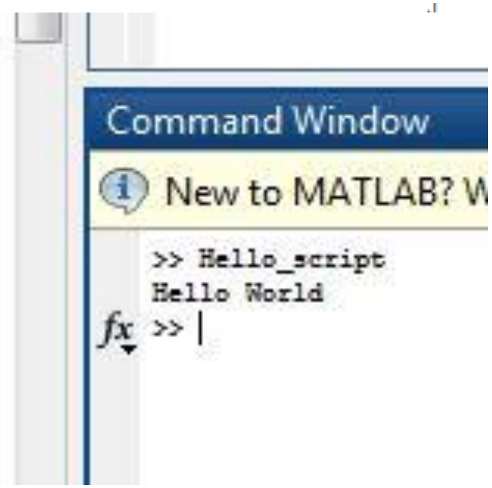
(3) Defining script name



File name: Hello_script.m

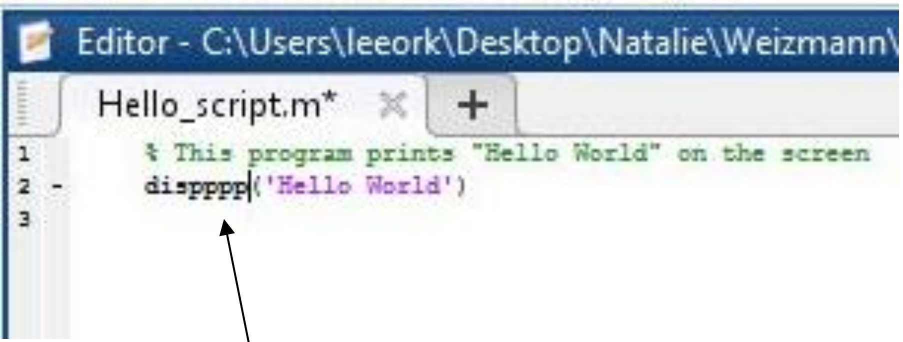
Save as type: MATLAB Code files (*.m)

(4) Running the script

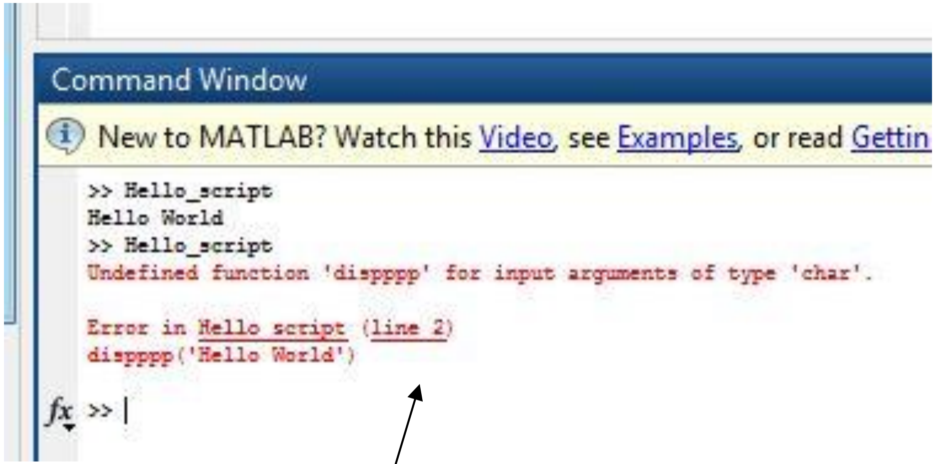


```
>> Hello_script
Hello World
>> |
```

Making errors…



```
1      % This program prints "Hello World" on the screen
2 -    dispppp('Hello World')
3
```



New to MATLAB? Watch this Video, see Examples, or read Gettin

```
>> Hello_script
Hello World
>> Hello_script
Undefined function 'dispppp' for input arguments of type 'char'.

Error in Hello_script (line 2)
dispppp('Hello World')

fx >> |
```

This command does NOT exist in Matlab!

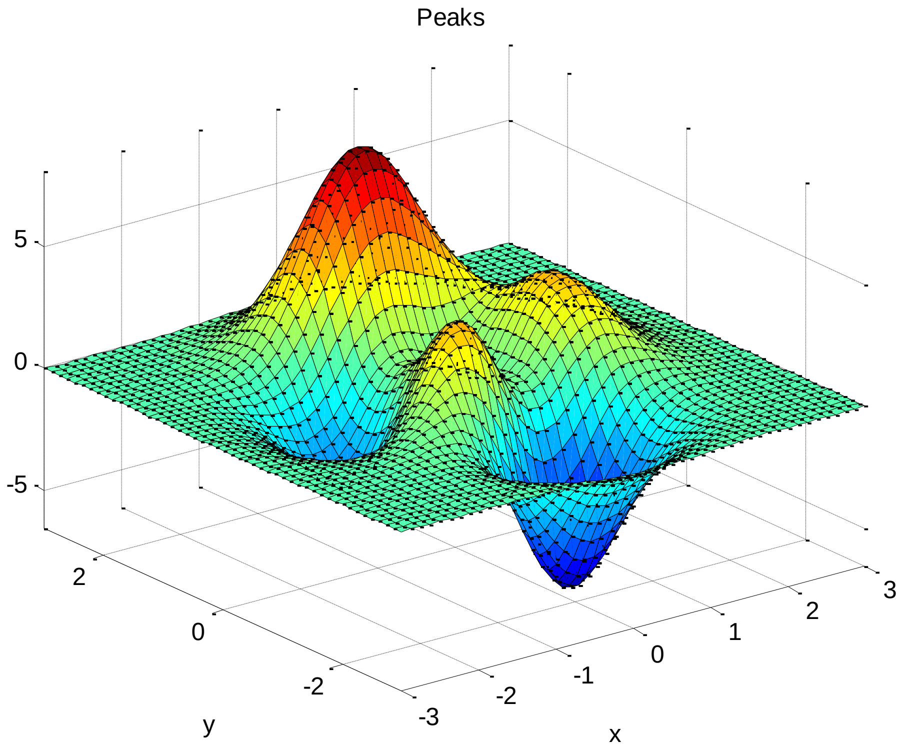
Pressing here will bring you to the line in the script where the error occurred

Another script…

Making sophisticated graphics and animation in Matlab is easy.

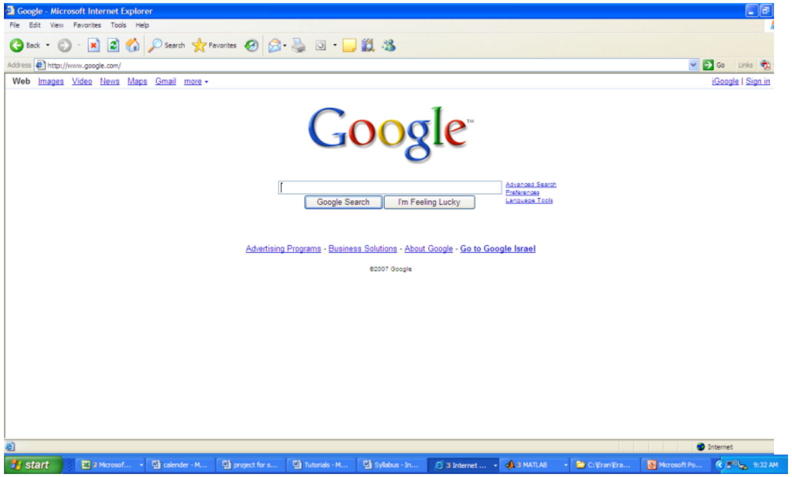
We will learn how to do this in two lectures

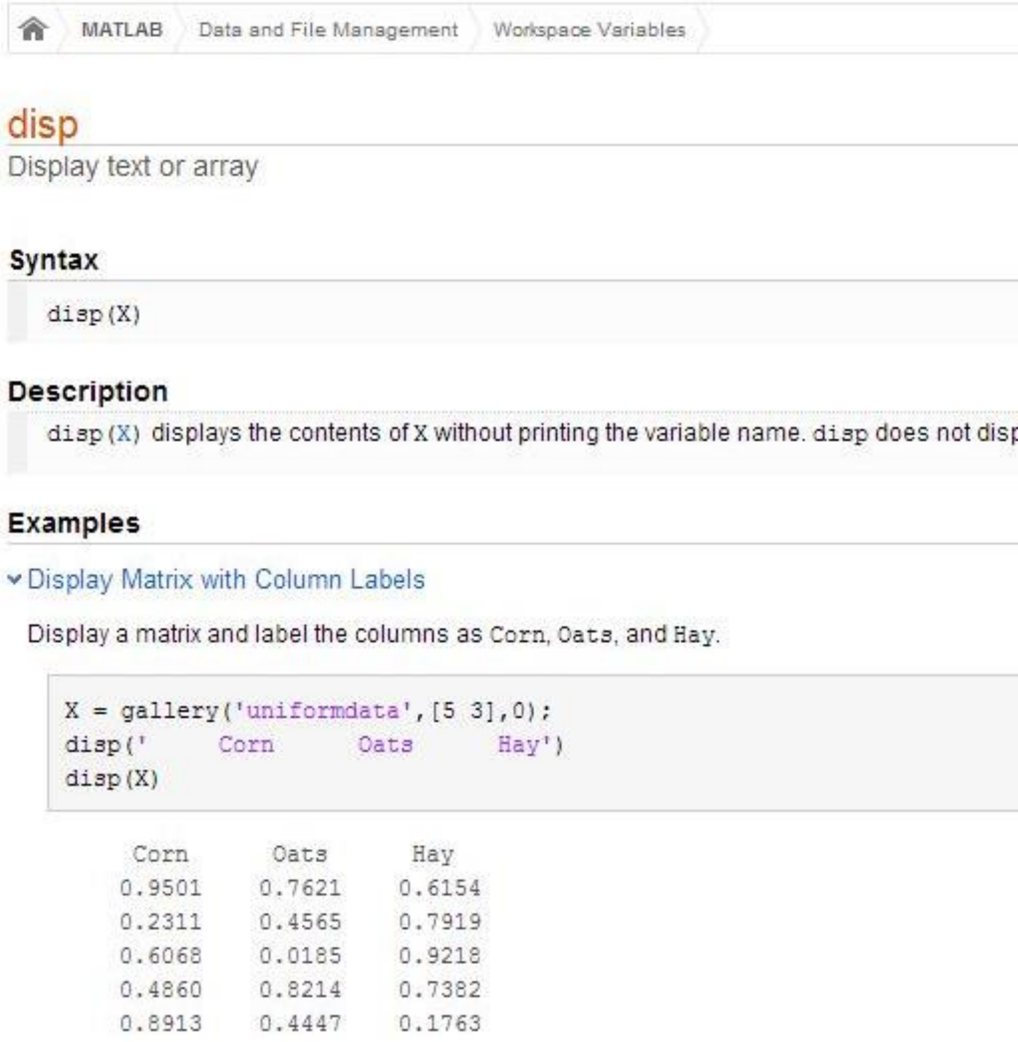
```
Z = peaks; surf(Z);
axis tight
set(gca,'nextplot','replacechildren');

% Record the movie
for j = 1:20
    surf(sin(2*pi*j/20)*Z,Z)
    F(j) = getframe;
end

% Play the movie twenty times
movie(F,20)
```



Help!!!

- help
- doc
  - Example: `doc disp`
- Google


Google - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.google.com/
Go Links
Web Images Video News Maps Gmail more
iGoogle | Sign in
Google
Google Search
I'm Feeling Lucky
Advanced Search
Preferences
Language Tools
Advertising Programs - Business Solutions - About Google - Go to Google Israel
©2007 Google
Internet
start



disp

Display text or array

Syntax

```
disp(X)
```

Description

`disp(X)` displays the contents of X without printing the variable name. `disp` does not disp

Examples

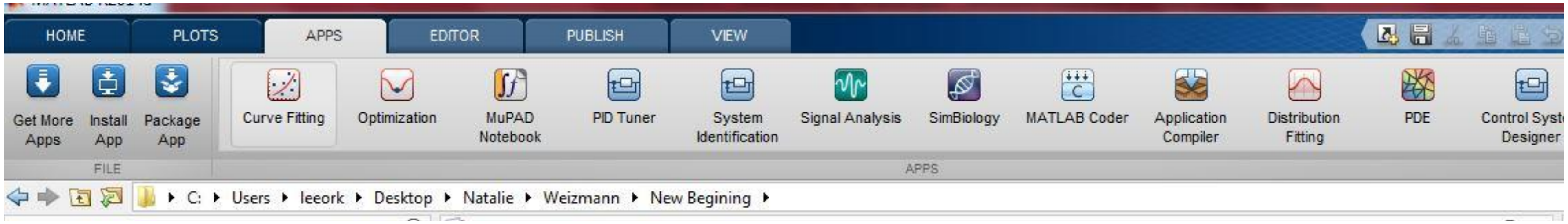
Display Matrix with Column Labels

Display a matrix and label the columns as `Corn`, `Oats`, and `Hay`.

```
X = gallery('uniformdata',[5 3],0);
disp('     Corn      Oats      Hay')
disp(X)
```

```
     Corn      Oats      Hay
    0.9501    0.7621    0.6154
    0.2311    0.4565    0.7919
    0.6068    0.0185    0.9218
    0.4860    0.8214    0.7382
    0.8913    0.4447    0.1763
```

Matlab apps


HOME
PLOTS
APPS
EDITOR
PUBLISH
VIEW
Get More Apps
Install App
Package App
FILE
Curve Fitting
Optimization
MuPAD Notebook
PID Tuner
System Identification
Signal Analysis
SimBiology
MATLAB Coder
Application Compiler
Distribution Fitting
PDE
Control Syst Designer
APPS
C: ▸ Users ▸ leeork ▸ Desktop ▸ Natalie ▸ Weizmann ▸ New Begining ▸

Introduction to Matlab
& Data Analysis

Topic #2:
The Matlab Building Blocks - Variables, Arrays and Matrices

identifiers

- Identifiers are all the words that build up the program
- An identifier is a sequence of letters, digits and underscores "_"
- Maximal length of identifiers is 63 characters
- Can't start with a digit
- Can't be a reserved word

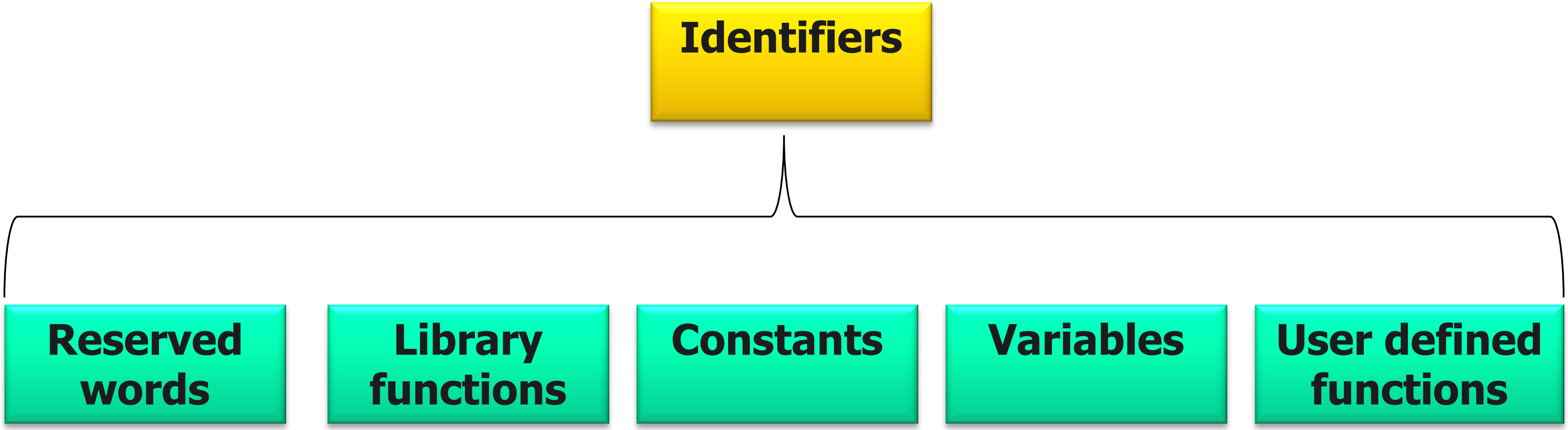
Examples of Legal identifiers:

- time
- day_of_the_week
- bond007
- findWord

Examples of illegal identifiers:

- 007bond
- #time
- ba-baluba
- if
- while

An overview of the main players in a program



Identifiers

Reserved words

Library functions

Constants

Variables

User defined functions

Reserved words (keywords)

- Words that are part of the Matlab language
  - There are 17 reserved words:
    - for
    - function
    - otherwise
    - try
    - break
    - end
    - return
    - switch
    - catch
    - if
    - elseif
    - continue
    - global
    - while
    - case
    - else
    - persistent

- Do **NOT** try to redefine their meaning!
- Don **NOT** try to redefine their library function names either!

Constants

- The value of a constant is fixed and does not change throughout the program

Numbers

100

0.3

Arrays

[1 2 3 4 5]

Matrices

[5 3

4 2]

Chars

‘c’

Strings

‘I like to eat sushi’

‘1 + 2’

Variables

- **Why do we need variables?**

- **Example:**

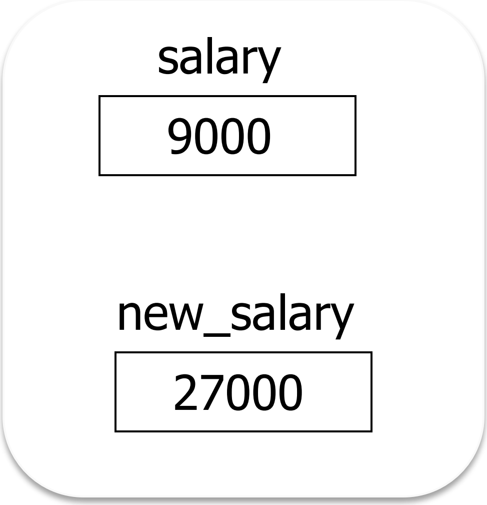
```
>> salary = 9000;
>> new_salary = salary * 3;
>> disp(new_salary);

    27000
```

constant

variable

Library functions

Computer memory



If we update salary, new_salary will NOT be updated automatically

Variables

- **Another example:**

```
price_bamba = 3
```

What happens if you omit the **‘;’ ?**

The Matlab Console

```
price_bamba =
      3
```

Variables

- **Another example:**

```
price_bamba = 3
n_bamba     = 2;
```

What happens when we add the **';' ?**

The Matlab Console

```
price_bamba =
     3
```

Variables

- **Another example:**

```
price_bamba = 3
n_bamba     = 2;
price_bisly = 5
n_bisly     = 3;
```

The Matlab Console

```
price_bamba =
     3

price_bisly =
     5

total_price =
    21

n_bamba =
     5

total_price =
    21
```

How can we fix it?

Redefine total_price

```
total_price = price_bamba * n_bamba + price_bisly * n_bisly
n_bamba     = 5
total_price
```

Variables

- **Tip #1**: Give your variables meaningful names.

```
a  = 9000
b = 100
```

are a bad choice for naming variables that store your working hours and salary!

A more meaningful choice of names would be

```
salary = 9000;
hours  = 5;
```

Variables

- **Tip #2**: Don’t make variable names too long

```
salary_I_got_for_my_work_at_the_gasoline_station = 9000;
salary_I_got_for_my_work_in_the_bakery = salary_I_got_for_my_work_at_the_gasoline_station * 3;
disp(salary_I_got_for_my_work_in_the_bakery);
```

Very bad choice of variable name!!!

- When should I use capital letters ?
- **Tip #3**: Whatever you do - be consistent.

Variables Types

- Each variable has a *type*
- Why do we need variable types?

- Different types of variable store different types of data

```
>> a = 10
a =
    10

>> class(a)
ans =
double
```

Returns the type of a variable

The default variable type in Matlab is double

Variables Types

- Double

Double-precision floating-point format is a computer number format that occupies 8 bytes (64 bits) in computer memory and represents a wide dynamic range of values by using floating point. (Wikipedia).

  - Allows representation of very large numbers (size of a galaxy) to very small numbers (subatomic particles).

Variables Types

- Each variable has a *type*

- Why do we need variable types?

- Different types of variable store different types of data

```
>> a = 10
a =
   10

>> class(a)
ans =
double
```

```
>> b = 10.56
b =
   10.5600

>> class(b)
ans =
double
```

```
>> c = 'Bush'
c =
Bush

>> class(c)
ans =
char
```

```
>> d = true
d =
     1

>> class(d)
ans =
logical
```

Variables Types

- Different variable types require different memory allocations

```
>> a = 10.4 %double requires 8 bytes
a =
   10.4
```

1	2	3		8
1 0 0 0 1 1 0 0	1 0 1 1 1 0 0 0	0 0 0 0 1 0 0 0	…	1 0 0 0 1 0 0 0

```
>> b = 'B'  %char requires 2 bytes
b =
B
```

1	2
1 0 0 0 1 1 0 0	1 0 1 1 1 0 0 0

Memory allocation and release is done automatically in Matlab

- How many bytes are required to store this variable: `c = 'Bush'` ?

Computer precision limitations

- How much is:

```
>> 0.42 + 0.08 - 0.5
ans =
    0
```

- How much is:

```
>> 0.42 - 0.5 + 0.08
ans =
 -1.3878e-017
```

Special variables

- ans

```
>> 4 * 5
ans =
    20

>> ans + 1
ans =
    21
```

Special variables

- ans
- pi
- inf

```
>> 2 * inf
ans =
   Inf

>> 1 / 0
Warning: Divide by zero.
ans =
   Inf
```

Special variables

- ans
- pi
- inf
- NaN
- In the tutorial you’ll see more…

```
>> 0 / 0
Warning: Divide by zero.
ans =
   NaN

>> NaN + 1
ans =
   NaN
```

Summary

- Matlab is a high level language
- Matlab working environment
- Variables & variable types + how to use them

Floating point

- From Wikipedia, the free encyclopedia.
- In computing, **floating point** describes a method of representing an approximation of a real number in a way that can support a wide range of values. The numbers are, in general, represented approximately to a fixed number of significant digits (the mantissa) and scaled using an exponent. The base for the scaling is normally 2, 10 or 16. The typical number that can be represented exactly is of the form:
- *Significant digits* × $\textit{base}^{\textit{exponent}}$The idea of floating-point representation over intrinsically integer fixed-point numbers, which consist purely of significand, is that expanding it with the exponent component achieves greater range. For instance, to represent large values, e.g. distances between galaxies, there is no need to keep all 39 decimal places down to femtometre-resolution (employed in particle physics).

Floating point (continued)

- Assuming that the best resolution is in light years, only the 9 most significant decimal digits matter, whereas the remaining 30 digits carry pure noise, and thus can be safely dropped. This represents a savings of 100 bits of computer data storage. Instead of these 100 bits, much fewer are used to represent the scale (the exponent), e.g. 8 bits or 2 decimal digits. Given that one number can encode both astronomic and subatomic distances with the same nine digits of accuracy, but because a 9-digit number is 100 times less accurate than the 11 digits reserved for scale, this is considered a trade-off exchanging range for precision. The example of using scaling to extend the dynamic range reveals another contrast with fixed-point numbers: Floating-point values are not uniformly spaced. Small values, close to zero, can be represented with much higher resolution (e.g. one femtometre) than large ones because a greater scale (e.g. light years) must be selected for encoding significantly larger values.[1] That is, floating-point numbers cannot represent point coordinates with atomic accuracy at galactic distances, only close to the origin.

Floating point

- The term *floating point* refers to the fact that a number's radix point (decimal point, or, more commonly in computers, binary point) can "float"; that is, it can be placed anywhere relative to the significant digits of the number. This position is indicated as the exponent component in the internal representation, and floating point can thus be thought of as a computer realization of scientific notation.