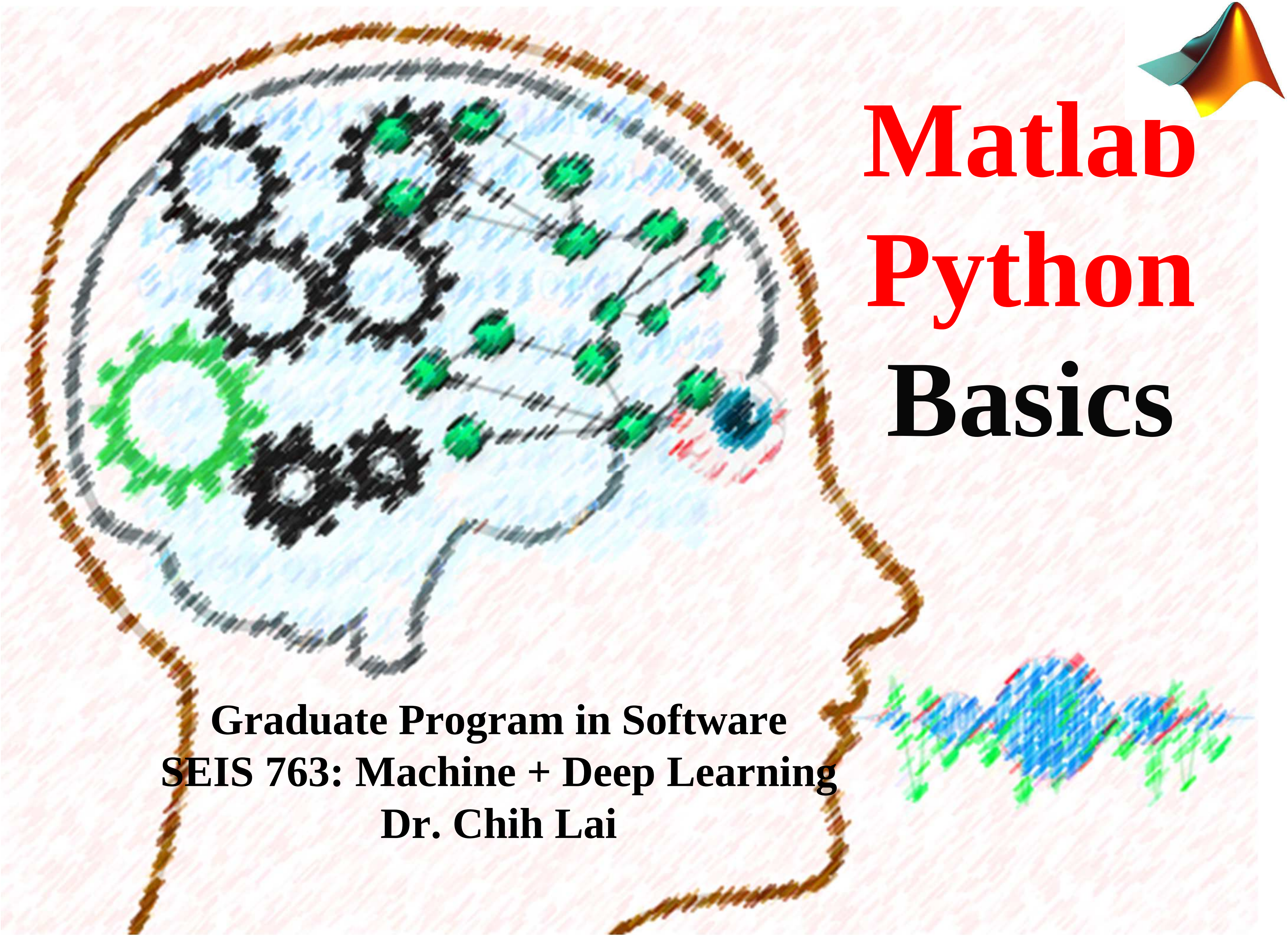
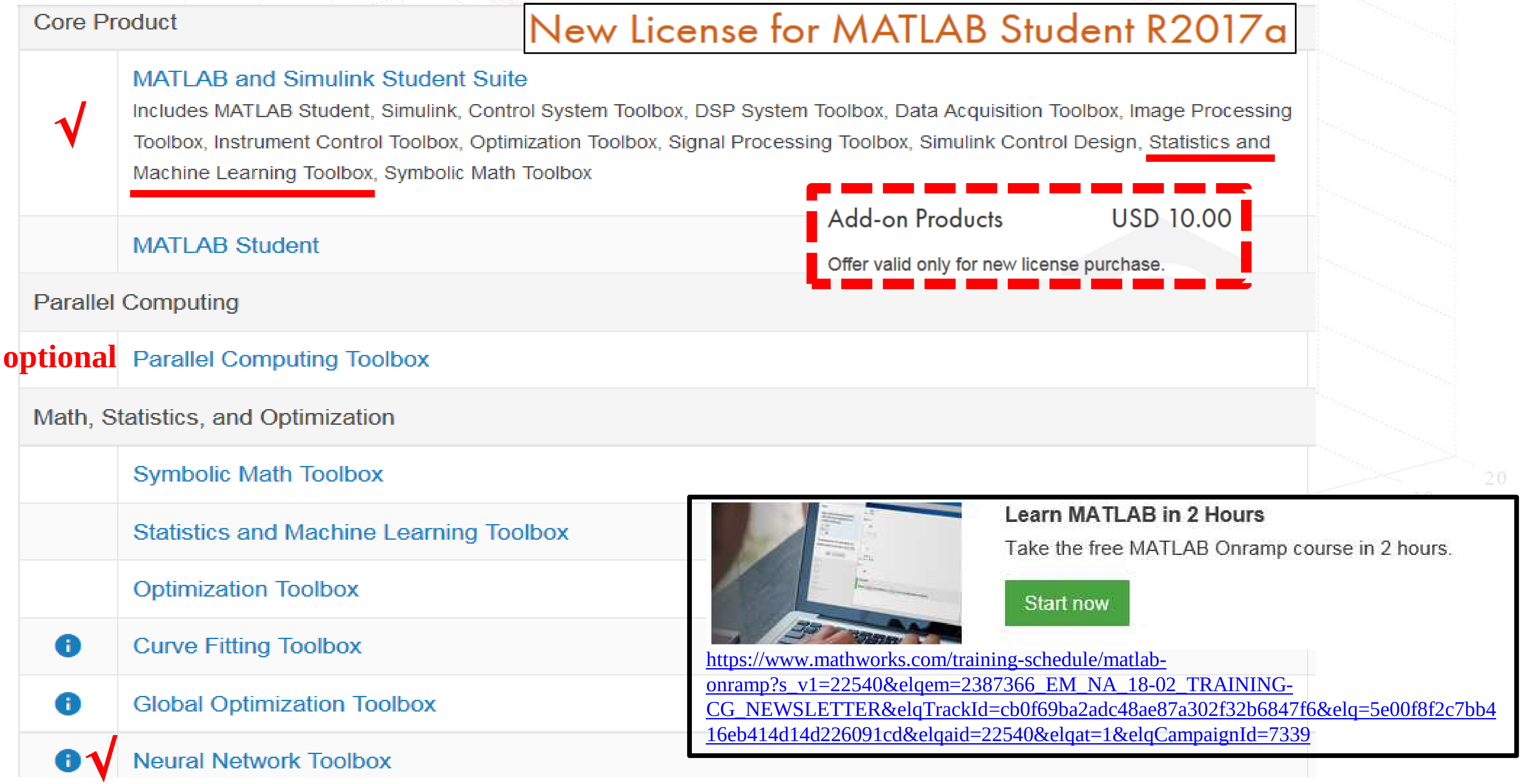


Matlab Python Basics

Graduate Program in Software
SEIS 763: Machine + Deep Learning
Dr. Chih Lai

What is Matlab

- MATLAB® is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in script notation.
- https://www.mathworks.com/store/link/products/student/new?s_iid=htb_buy_gtwy_cta3



New License for MATLAB Student R2017a

Core Product

√ MATLAB and Simulink Student Suite
Includes MATLAB Student, Simulink, Control System Toolbox, DSP System Toolbox, Data Acquisition Toolbox, Image Processing Toolbox, Instrument Control Toolbox, Optimization Toolbox, Signal Processing Toolbox, Simulink Control Design, Statistics and Machine Learning Toolbox, Symbolic Math Toolbox

MATLAB Student

Add-on Products USD 10.00
Offer valid only for new license purchase.

Parallel Computing

optional Parallel Computing Toolbox

Math, Statistics, and Optimization

Symbolic Math Toolbox

Statistics and Machine Learning Toolbox

Optimization Toolbox

ℹ Curve Fitting Toolbox

ℹ Global Optimization Toolbox

ℹ √ Neural Network Toolbox

Learn MATLAB in 2 Hours
Take the free MATLAB Onramp course in 2 hours.

https://www.mathworks.com/training-schedule/matlab-onramp?s_v1=22540&elqem=2387366_EM_NA_18-02_TRAINING-CG_NEWSLETTER&elqTrackId=cb0f69ba2adc48ae87a302f32b6847f6&elq=5e00f8f2c7bb416eb414d14d226091cd&elqaid=22540&elqat=1&elqCampaignId=7339

MatLab Introduction

- Matlab is an abbreviation of **Matrix** **Laboratory**.

- The trick behind Matlab is that
  - *everything is represented in the form of* ***arrays*** *or* ***matrices*** *or* ***tables***.

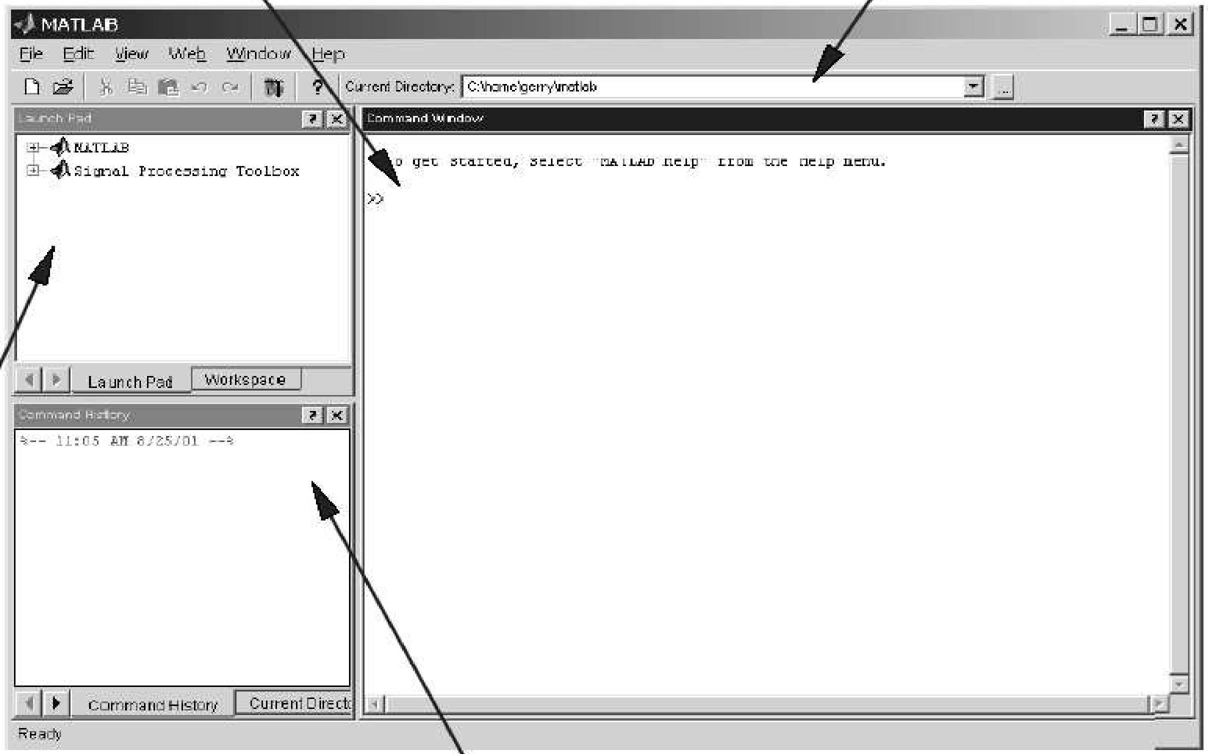
- It is a popular Mathematical Programming Environment used extensively in Education as well as in Industry.
  - Originally a user interface for numerical linear algebra routines (Lapak/Linpak)
  - Commercialized 1984 by The Mathworks
  - Since then heavily extended (**defacto-standard**)

- Code developed in Matlab can be converted into C, C++, Java

Getting Started– Matlab Screen

Command Prompt:
Enter typed commands here.
Text results are displayed here.

Recent Directory Menu:
Used to change current
working directory.



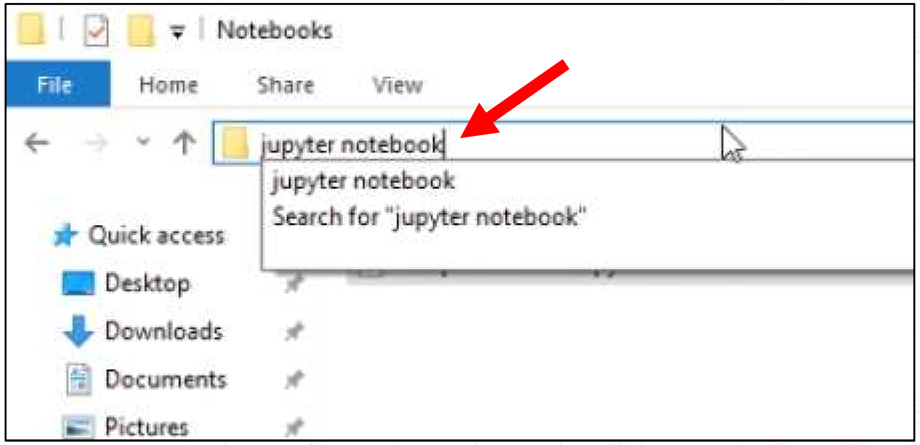
```
To get started, select "MATLAB Help" from the Help menu.

>>
```

```
%-- 11:05 AM 8/25/01 --%
```

Launch Pad/Workspace:
Used to browse documentation,
or view values of variables in
the workspace

Command History/Current Directory:
Used to view and re-enter typed commands,
or change directories

Easy Way to Run Jupyter Notebook from ANY Folder



```
jupyter notebook
```

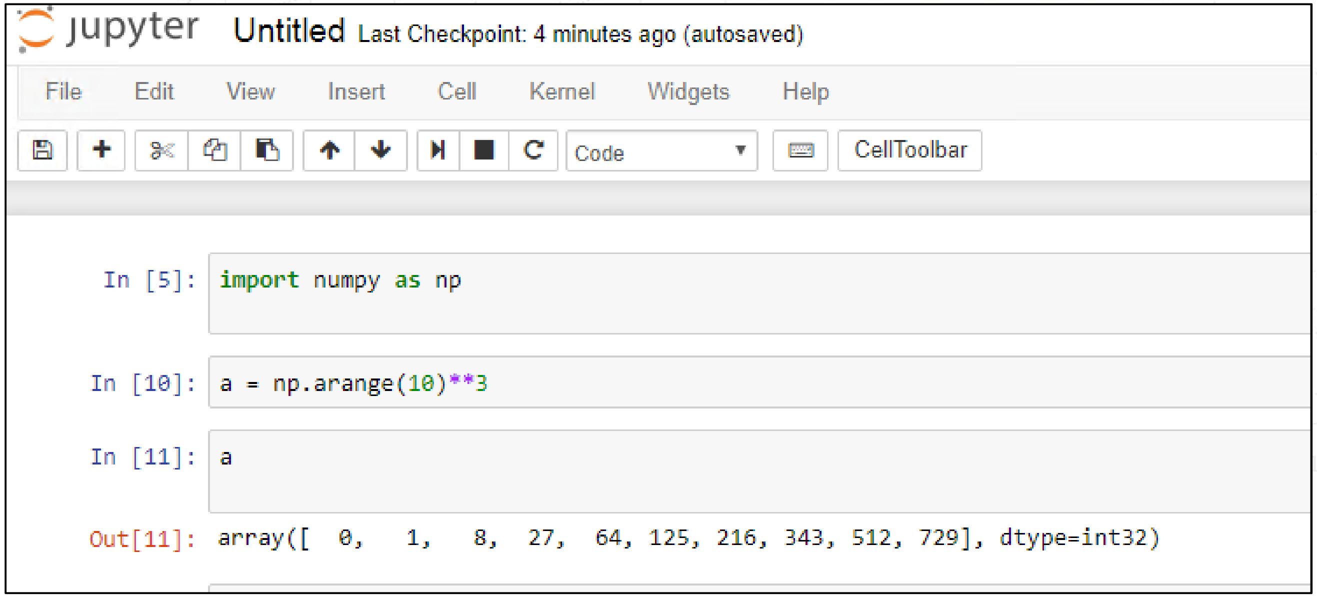
jupyter notebook

Search for "jupyter notebook"

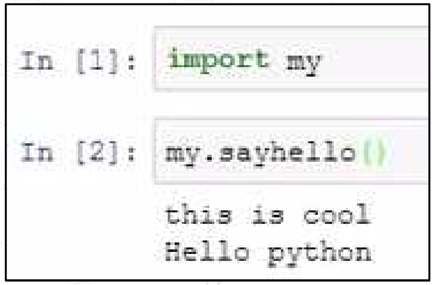
Install package under Anaconda

1 .Conda install [your_Package]

2. Pip install sklearn
   [also scipy, tensorflow, keras]

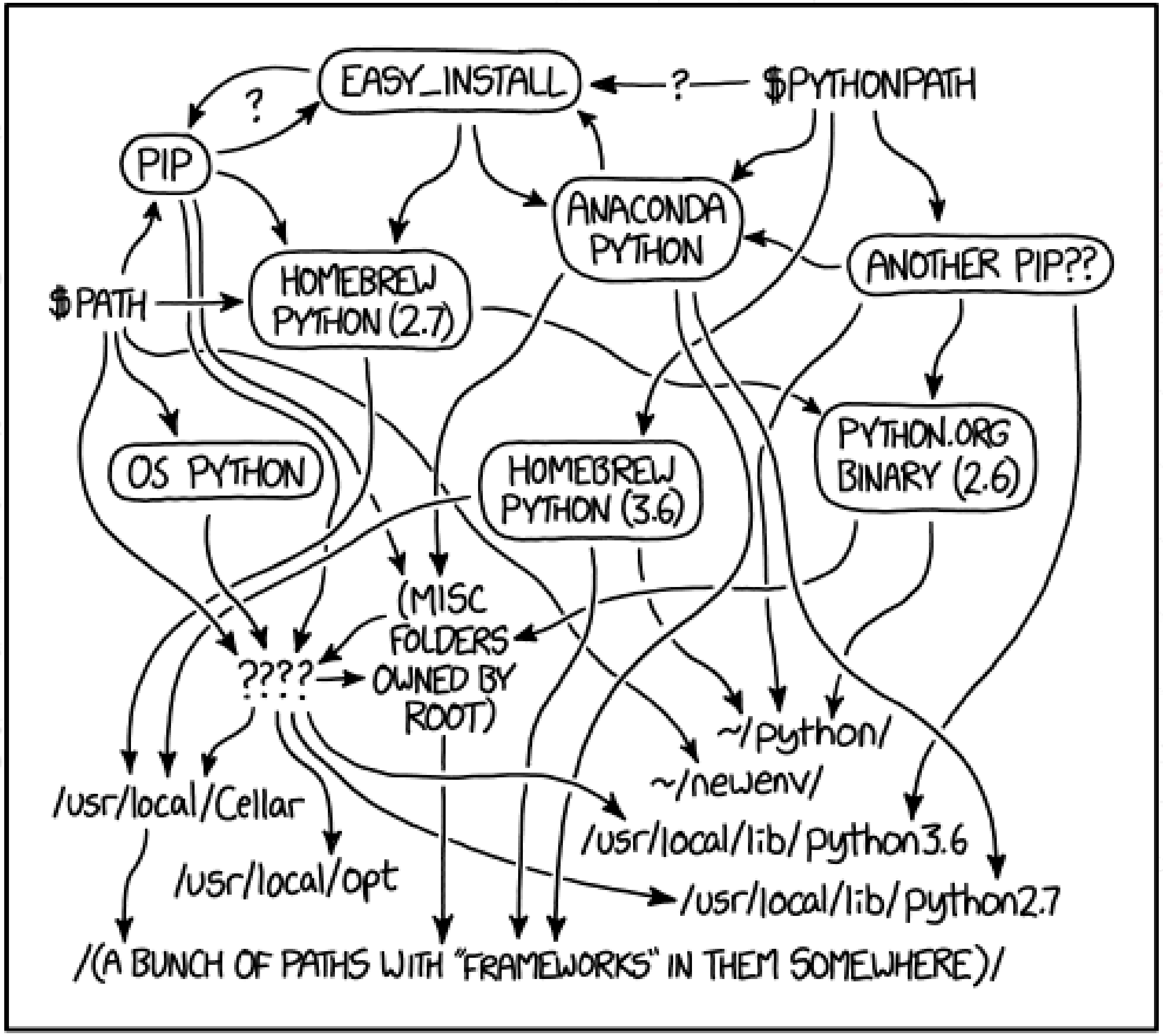


```
In [5]: import numpy as np

In [10]: a = np.arange(10)**3

In [11]: a

Out[11]: array([  0,   1,   8,  27,  64, 125, 216, 343, 512, 729], dtype=int32)
```



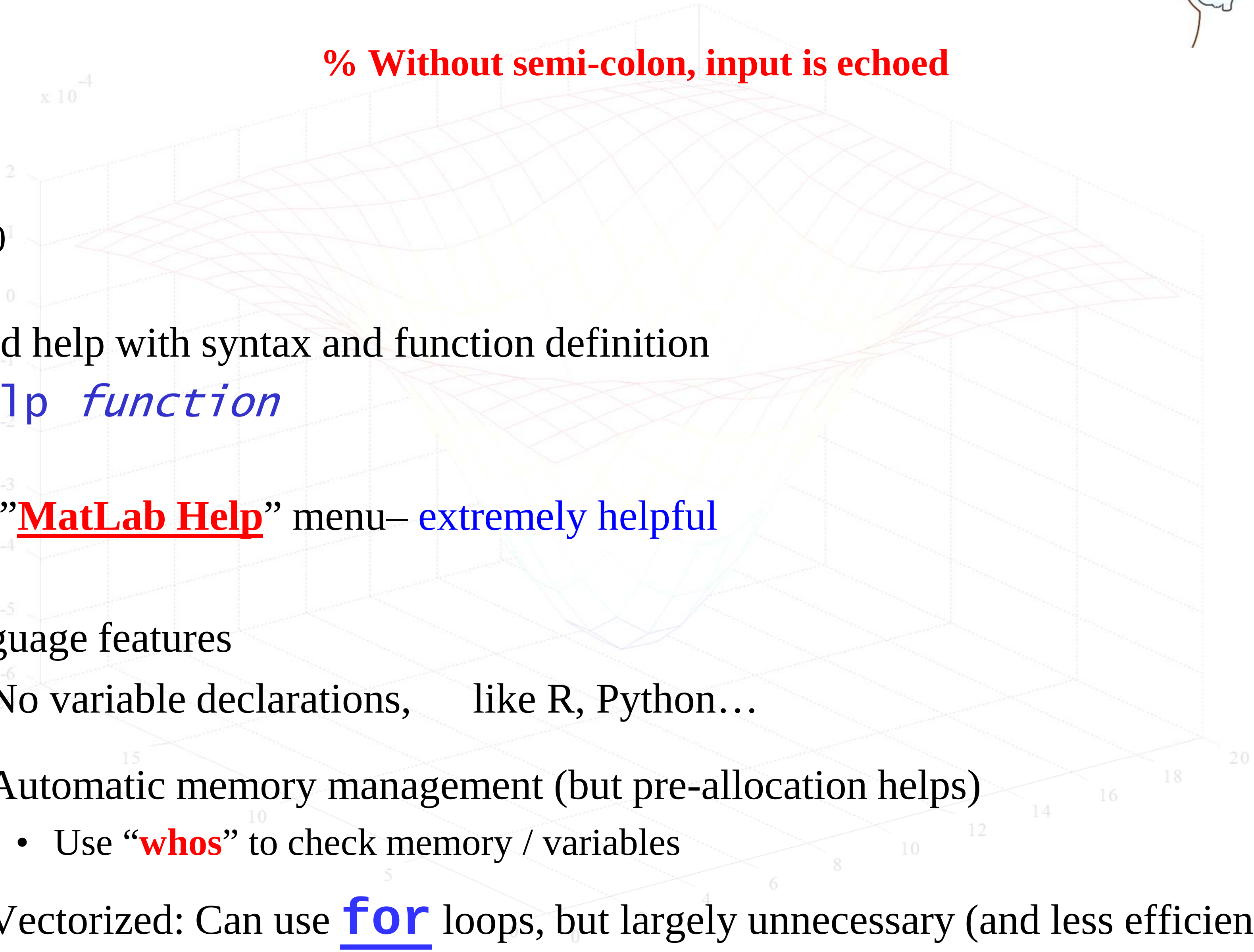
```
In [1]: import my

In [2]: my.sayhello()

        this is cool
        Hello python
```

Python Environment



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

MatLab Simple Command



```
» a=5;                          % Without semi-colon, input is echoed

» b=a/2
b =
   2.5000
```

- Rapid help with syntax and function definition

```
>> help function
```

➡ Use "**MatLab Help**" menu– extremely helpful

- Language features
  - No variable declarations,   like R, Python…
  - Automatic memory management (but pre-allocation helps)
    - Use "**whos**" to check memory / variables
  - Vectorized: Can use `for` loops, but largely unnecessary (and less efficient)

MatLab Variable Names

- Variable names ARE **case sensitive**

- Variable names can contain up to 63 characters

- Variable names must start with a letter followed by letters, digits, and underscores

- MatLab special variables

ans	Default variable name for results
pi	Value of π
inf	Infinite number
NaN	Not a number e.g. 0/0
eps	Smallest incremental number
realmin	The smallest usable positive real number
realmax	The largest usable positive real number

Matrix

- MATLAB works w/ only one kind of data– rectangular numerical matrix
  - Entered manually;
    - A = [1, 2, 3; 7, 8, 9] **% Use comma or space to separate elements in a row**
    - Use ‘ **;** ’ to indicate the **end of each row**
    - Same as A = [1 2 3

      7 8 9]
  - Generated by built-in functions;
    - A = **zeros**(2, 3); or A = **ones**(5, 8) * 2 + 4; A = rand(3,5) **% randn**
    - **np.zeros((2,3)) or A = np.ones((2,3), dtype=int) A = np.random.random((3,5)) ←Python**
  - Loaded from external disk (using “load” command)
    - **load fisheriris** **sklearn.datasets.load_iris() ← Python**
- Comma / semi-colon to separate statements on same line
  - a = 5**,** b = [3 4]**;** a = a + 7**;** b = b * 2**;**
- Some important matrix functions:
  - size(a), c = size(b); length(a), max(a), min(a), max(**a(:)**), sum(a), sum(**a(:)**)

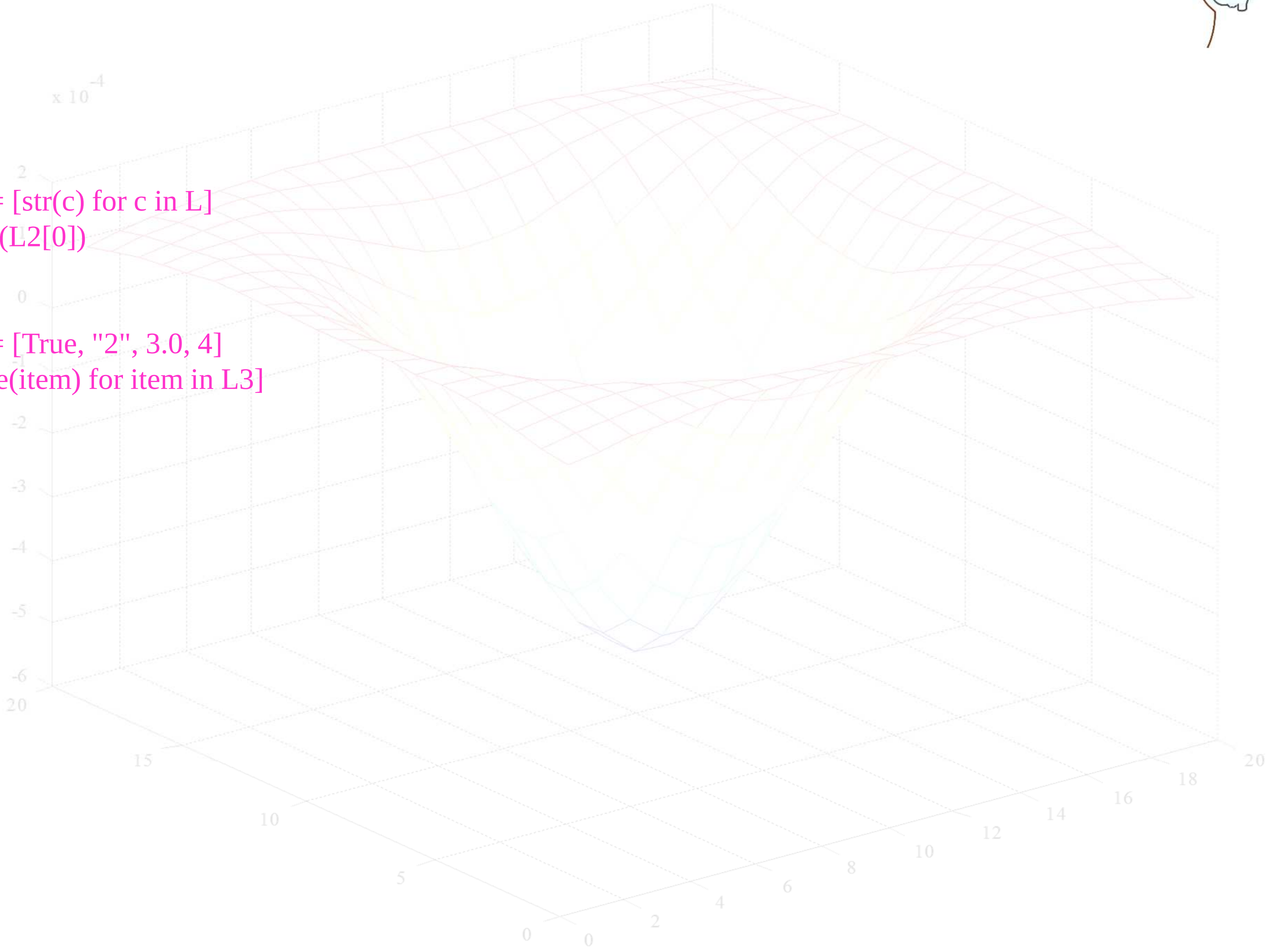
Python Matrix***

```
import numpy as np
a = np.arange(15).reshape(3, 5)
a
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14]])

a.shape
(3, 5)
a.ndim
2


np.zeros( (3,4) )
array([[ 0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.]])

np.ones( (3,4))

np.arange( 10, 30, 5 )
array([10, 15, 20, 25])
np.arange( 0, 2, 0.3 )
array([ 0. ,  0.3,  0.6,  0.9,  1.2,  1.5,  1.8 ])
```

```
a = np.ones((2,3), dtype=int)
b = np.random.random((2,3))
a *= 3
a
array([[3, 3, 3],
       [3, 3, 3]])
b += a

a.sum()
2.5718191614547998
a.min()
0.1862602113776709
a.max()


b.sum(axis=0)                         # sum of each column
array([12, 15, 18, 21])

b.min(axis=1)                         # min of each row
array([0, 4, 8])

b.cumsum(axis=1)
```

```
w, h = 3, 4; Matrix = [[0 for x in range(w)] for y in range(h)]
```

Python Matrix of Different Types



```
L2 = [str(c) for c in L]
type(L2[0])
```

```
L3 = [True, "2", 3.0, 4]
[type(item) for item in L3]
```

Matrix Subscripts

- A = [1, 2, 3; 7, 8, 9]   **A = np.array([(1, 2, 3), (7, 8, 9)]) ← Python**

- Subscripts:
  - the element in row i and column j of A is denoted by A(i, j)
  - A(1,1) + A(1,3) + A(2,2) + A(2,3) = 1 + 3 + 8 + 9 = 21

- The **Colon Operator ‘:’** is one of most important operators
  - Subscript expressions involving colons refer to **portions** of a matrix
  - A(2 , 1:3) is the first to the third elements of the second row of A
    - ans = [7, 8, 9]
    - **A[:, 1:3]   A[2, 1:3] ← Python**
    - **A[:6:2] = -1   # equivalent A[0:6:2] = -1; from start to idx 6, exclusive, set every 2nd element to -1**
  - A(end , :) is all the elements in the last row of A   **A[-1] ← Python**
  - What does A(:, 1:2) mean?
  - reshape(A, 1, [])   **A.reshape(1,A.size) A.reshape(2,20) ← Python**
  - A(:)   **A[:] = 0 ← Python**

Accessing to Outside of A Matrix

- A = [1, 2, 3; 7, 8, 9]

- Error if you try to **read** from a value of an element outside the matrix
  - e.g. A(3, 1)

- **BUT**, if you **write** to it, the matrix will **expand** to accommodate it
  - e.g. A(3, 2) = 50;
  - Now A has size 3 x 3 to

    A = [1 2 3
    7 8 9
    0 50 0]
  - You can verify it by typing size(A), size(A, 1), size(A, 2)
  - length(A) returns max(size(A))
  - numel(A) returns the total number of elements in A

Changing Matrix Size

- Deleting rows and columns
  - B(:,2) = [ ]

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

→

```
B =
    1    3
    4    6
    7    9
```

- **Concatenation**:

  A = [1 2; 3 4]

  B = [A   A-2;   A*2   A/4]

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

→

```
B =
    1.0000    2.0000   -1.0000         0
    3.0000    4.0000    1.0000    2.0000
    2.0000    4.0000    0.2500    0.5000
    6.0000    8.0000    0.7500    1.0000
```

```
# Python concatenation
# matrices must have the same size  ←←←
newX = np.concatenate((A, B), axis=0)

# Python concatenation
a = [1,2,3];                b = [5,6,7]
a+=b;                       print(a)
# [1, 2, 3, 5, 6, 7]

# Python numeric ops
a = np.array([1,2,3]);      b = np.array([5,6,7])
a+=b;                       print(a)
# [6, 8, 10]


# matrices must have the same size  ←←←
np.column_stack( (X_test_encoded, Y_train) )
# next 2 stmts are for nice print out
np.set_printoptions(precision = 3)
np.set_printoptions(suppress = True)
```

MATLAB Relational and Logical Operators

- MATLAB supports six relational operators.

Less Than	<
Less Than or Equal	<=
Greater Than	>
Greater Than or Equal	>=
Equal To	==
Not Equal To	~=

- MATLAB supports three logical operators.

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

MATLAB / Python Logical Functions

any (x) returns 1 if any element of x is nonzero

all (x) returns 1 if all elements of x are nonzero

nnz(x) returns number of non-zero elements in x

isnan (x) returns 1 at each NaN in x

X[np.isnan(X)] = 0 # python 🡐

isinf (x) returns 1 at each infinity in x

isempty(x) returns 1 if matrix is empty

finite (x) returns 1 at each finite value in x

```
# Python search "numbers"
q = np.array([])
y = np.array([0, 0, 1, 3, 4, 0, 5])
print(q.any(), y.any());  print(q.all(), y.all())
print(np.count_nonzero(q), np.count_nonzero(y))
print(np.all(qq==0), np.all(y==0))
y.isna(),      y.isna().sum()

idx1 = np.where(y <= 1);
```

find(x) %returns idx of non-zero values

findstr(x, 'Hello')

```
# Python search "String"
li = ['new', 'pilgrim', 'z', 'example', 'new', 'two', 'element']
idx = li.index("example")
print(idx)
```

Index = **find(contains(CellArry(:, col_num)**, 'MyStr')) **%🡐 search string in a matlab Cell array**

Display formats

- MATLAB supports 8 formats for outputting numerical results.

format long	16 digits
format short e	5 digits plus exponent
format long e	16 digits plus exponent
format hex	hexadecimal
format bank	two decimal digits
format +	positive, negative or zero
format rat	rational number (215/6)
format short	default display

- Check variable type
  - MATLAB class(x)
  - **Python➔** type(x)

```
# next 2 stmts are for nice print out
np.set_printoptions(precision = 3)
np.set_printoptions(suppress = True)
print(x)
```

```
>> disp(pi)
>> disp(num2str(pi, '%1.8f\n')),      disp(num2str(pi)),
```

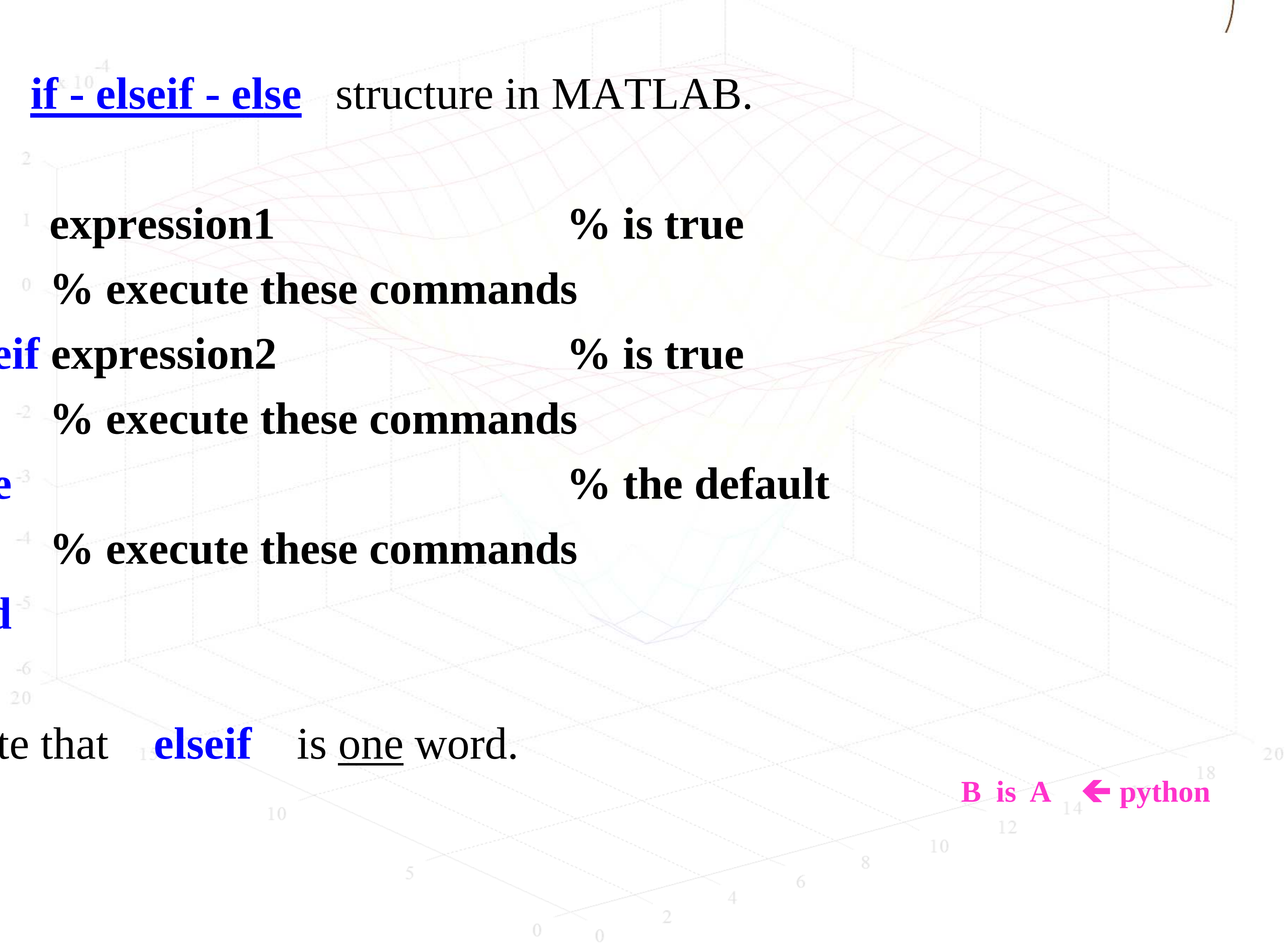
Matlab Condition Structures

- An **if - elseif - else** structure in MATLAB.

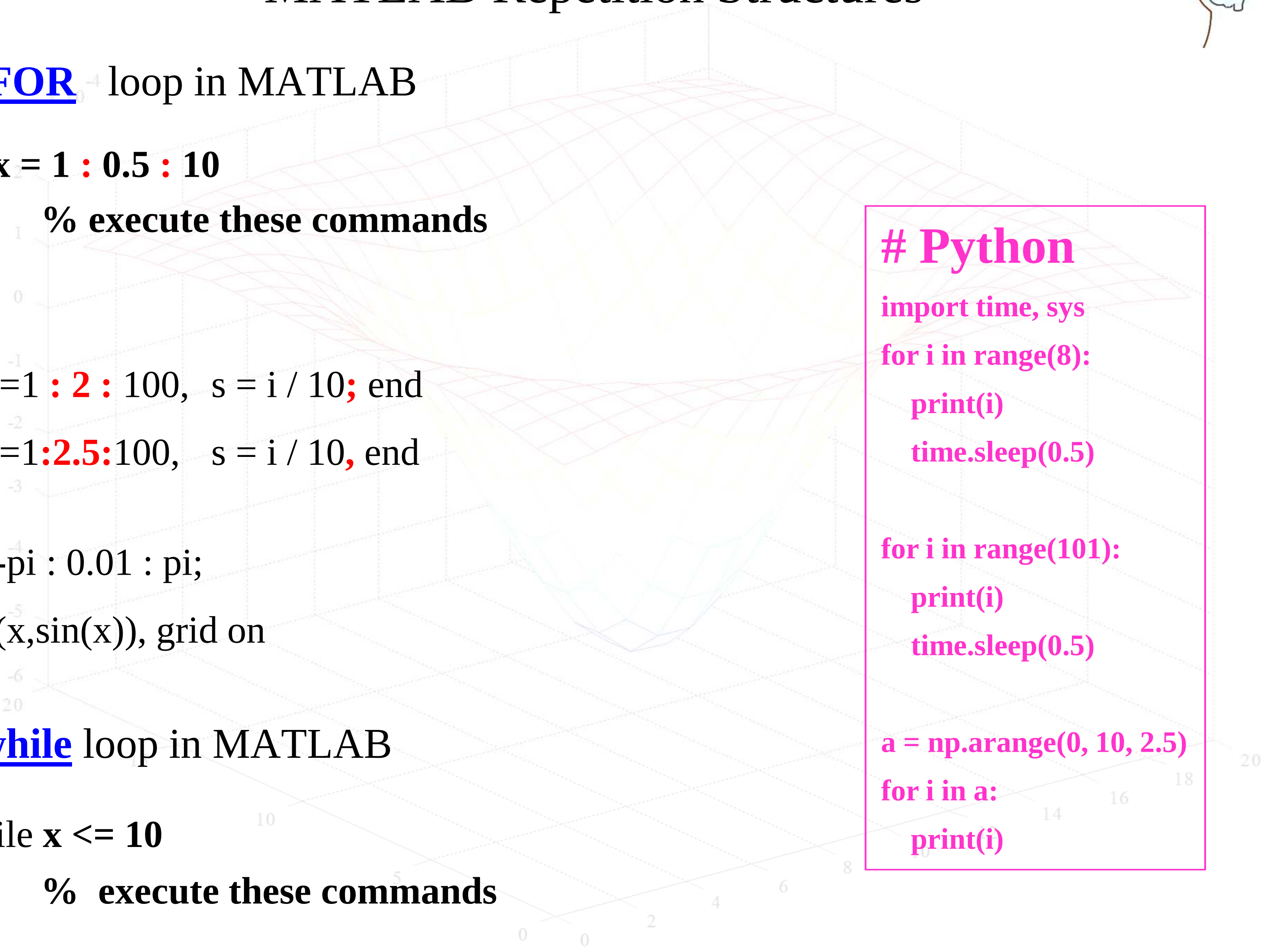
```
if      expression1                     % is true
        % execute these commands
elseif  expression2                     % is true
        % execute these commands
else                                    % the default
        % execute these commands
end
```

Note that **elseif** is one word.

B is A ← python



MATLAB Repetition Structures



- A **FOR** loop in MATLAB

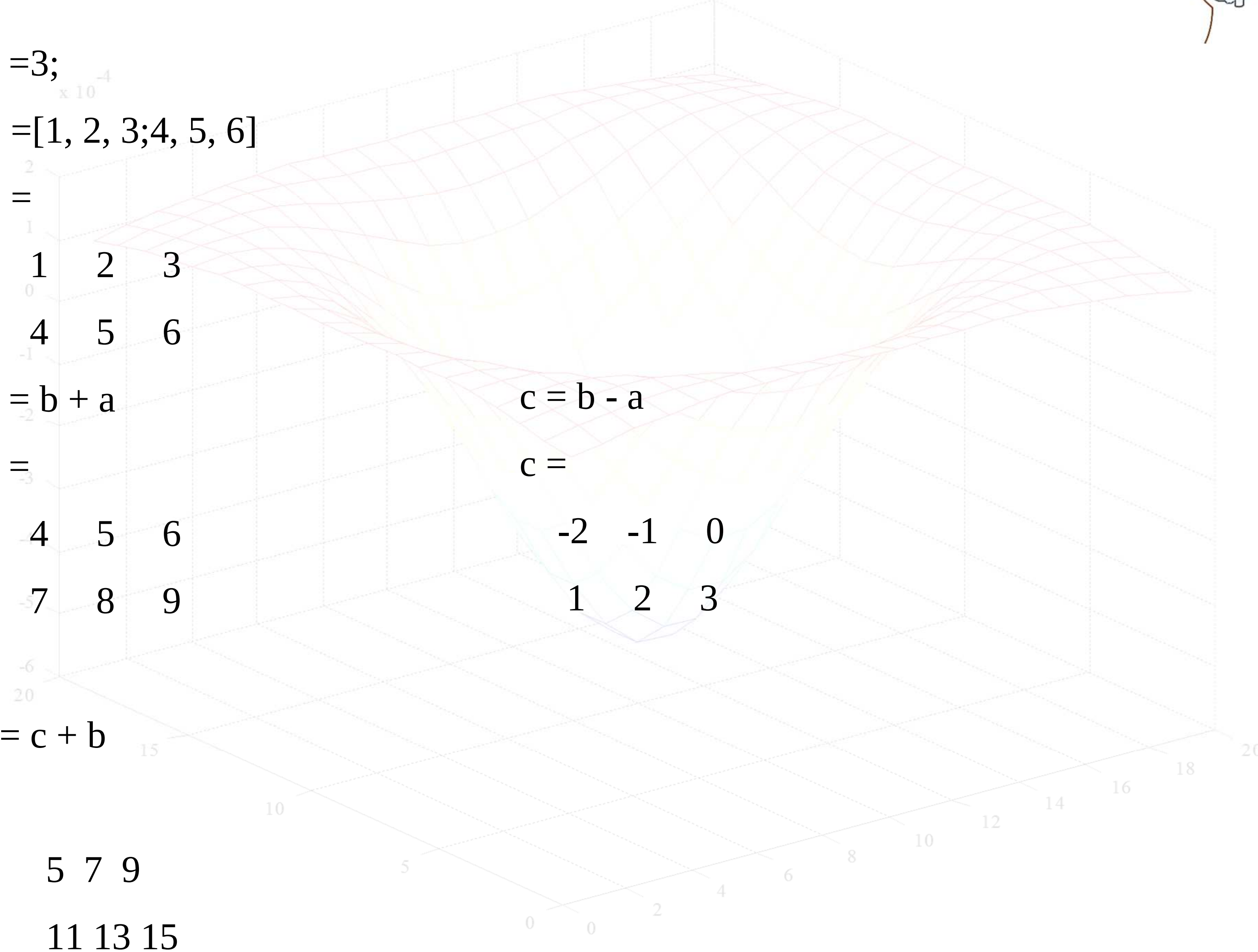
```
for x = 1 : 0.5 : 10
        % execute these commands
 end


for i=1 : 2 : 100,  s = i / 10; end

for i=1:2.5:100,   s = i / 10, end


x = -pi : 0.01 : pi;

plot(x,sin(x)), grid on
```

- A **while** loop in MATLAB

```
 while x <= 10
        %  execute these commands
 end
```

```
# Python
import time, sys
for i in range(8):
   print(i)
   time.sleep(0.5)

for i in range(101):
   print(i)
   time.sleep(0.5)

a = np.arange(0, 10, 2.5)
for i in a:
   print(i)
```

Scalar - Matrix Addition & Subtraction

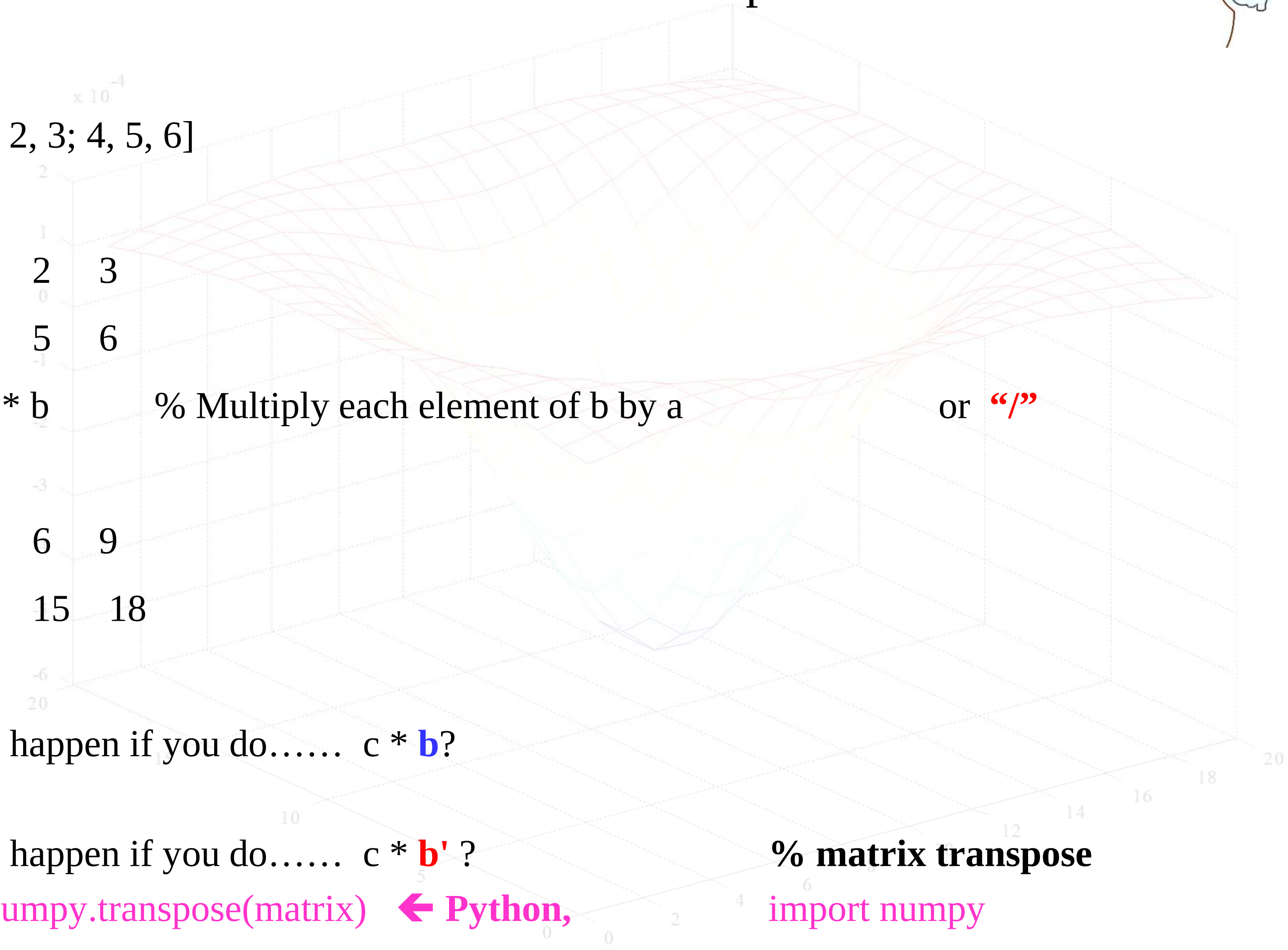


```
a =3;

b =[1, 2, 3;4, 5, 6]

b =

     1     2     3

     4     5     6

c = b + a

c =

     4     5     6

     7     8     9
```

```
c = b - a

c =

    -2    -1     0

     1     2     3
```

```
c= c + b


       5  7  9

       11 13 15
```

Scalar - Matrix Multiplication



```
a=3;

b=[1, 2, 3; 4, 5, 6]

b =

    1    2    3

    4    5    6

c = a * b          % Multiply each element of b by a                    or  “/”

c =

    3    6    9

   12   15   18
```

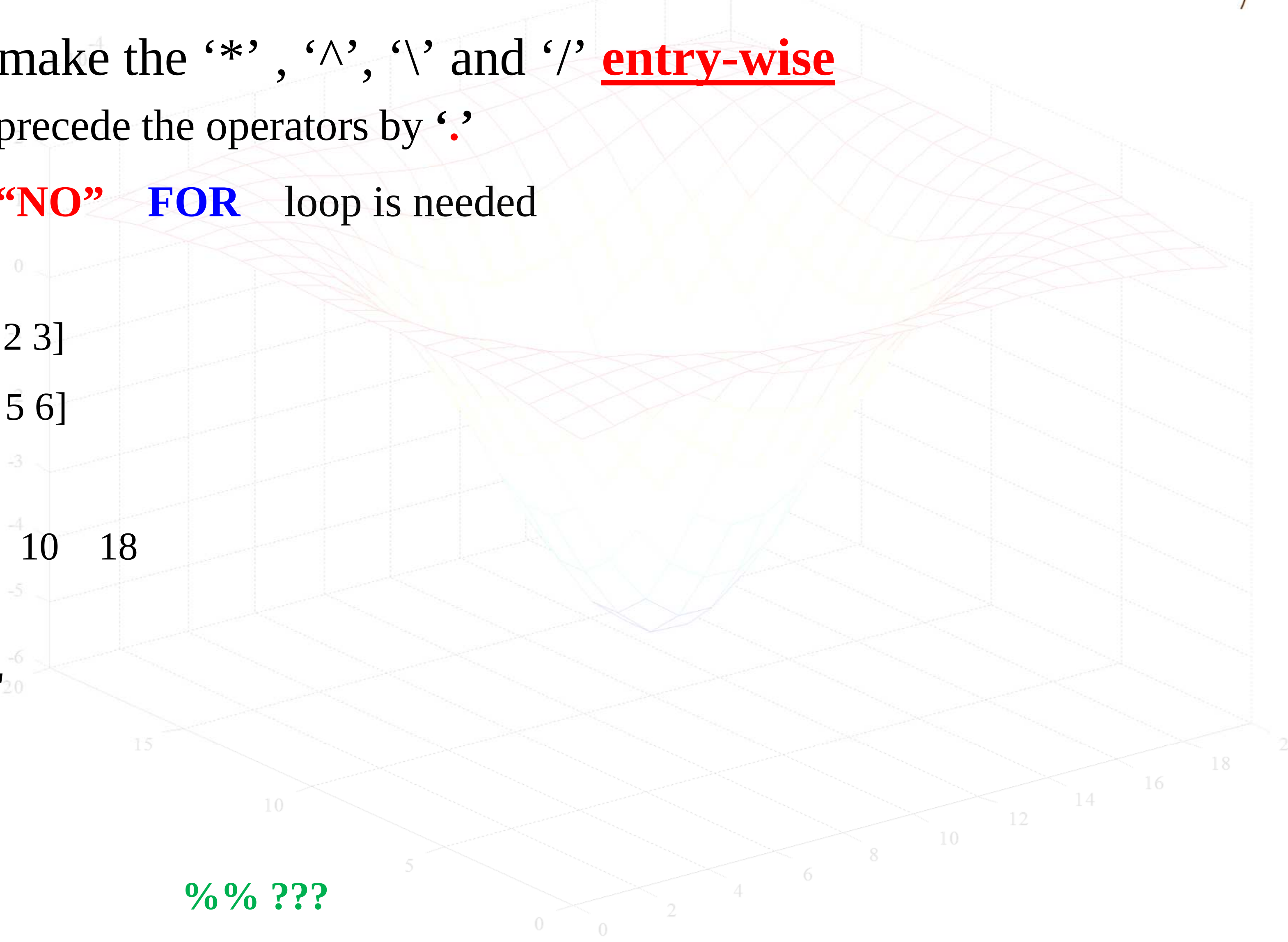
What happen if you do…… c * **b**?

What happen if you do…… c * **b'** ? **% matrix transpose**

numpy.transpose(matrix) **🡄 Python,** import numpy

Matrix Entry-Wise Operations

- To make the ‘*’ , ‘^’, ‘\’ and ‘/’ **entry-wise**
  - precede the operators by **‘.’**
  - **“NO”** **FOR** loop is needed



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

>> b=[4 5 6]

>> a.*b

ans = 4    10    18


>> a * b'

ans = 32


>> a. ^2                %% ???
```

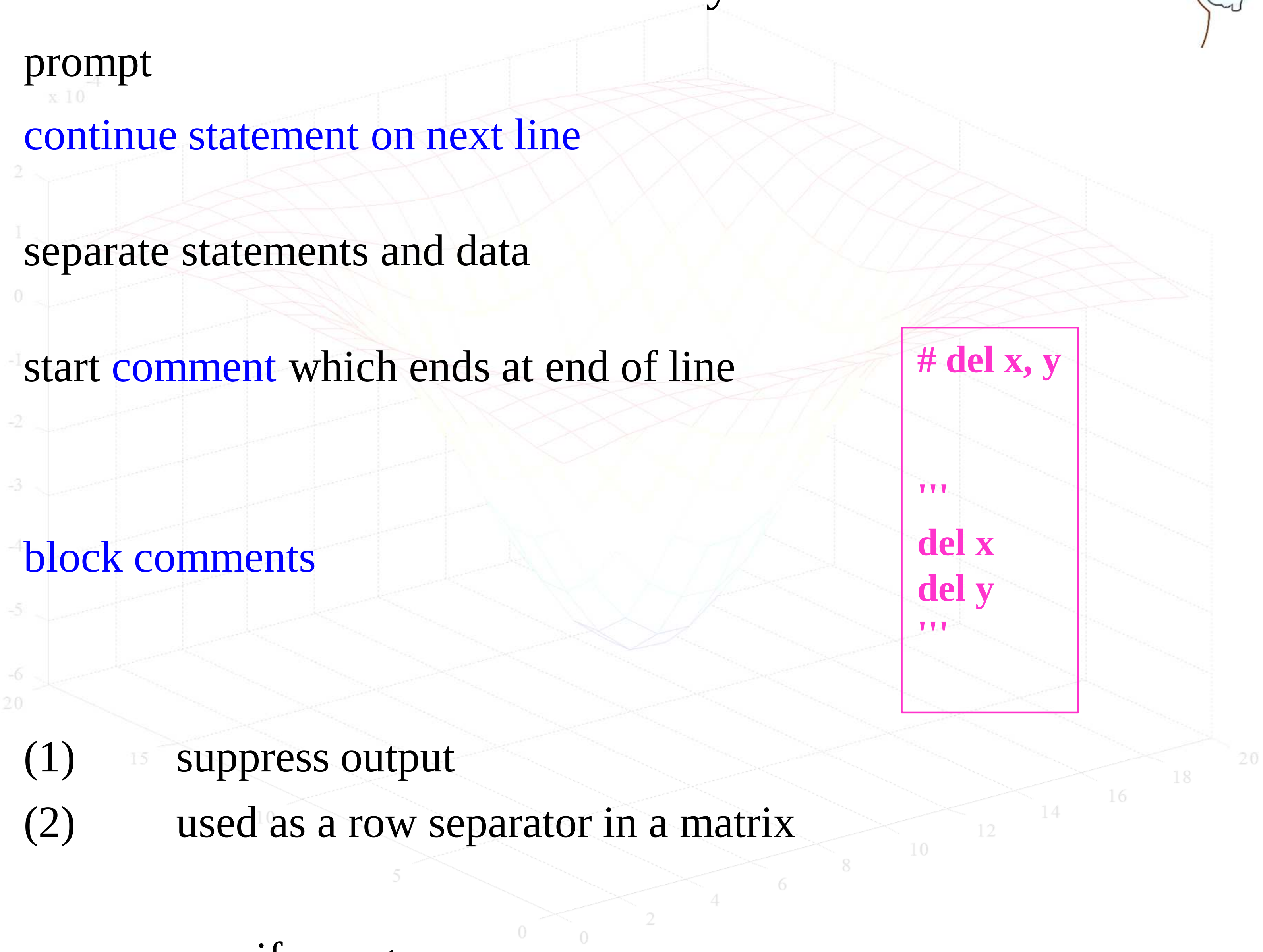
Some Useful MATLAB commands

clear	**Clear all variables from work space**
clear x y	**Clear variables x and y from work space**
clear all	
close all	
clc	Clear the command window
who	List known variables
whos	List known variables plus their size
help	Ex: >> help sqrt Help on using sqrt
lookfor	Ex: >> lookfor sqrt Search for keyword sqrt in m-files

del x, y

- what Ex:>> what a: List MATLAB files in a:

Other MATLAB symbols



>> prompt

. . . continue statement on next line

, separate statements and data

% start comment which ends at end of line

```
# del x, y

'''
del x
del y
'''
```

%{

….. block comments

%}

; (1) suppress output

(2) used as a row separator in a matrix

: specify range

Program (.m file)

- To create a m-file, choose **New** from the **File** menu and select **M-file**.
- The m-files can be run in the MATLAB workspace by typing the name of the file.
  - Note, you do not need to append the .m

```
%% ImageMatrix must be an indexed image, cannot be an RGB or HSV image
function [Rtn] = myFunc(ImageMatrix, Distance)
[MyHeight, MyWidth] = size(ImageMatrix);
Rtn = zeros(1, max(max(ImageMatrix)) + 1);
Sum = zeros(1, max(max(ImageMatrix)) + 1);
checks = zeros(1, max(max(ImageMatrix)) + 1);
Cm = zeros(1, max(max(ImageMatrix)) + 1);

for i = 1 : MyHeight
  for j = 1 : MyWidth
    Cm(ImageMatrix(i, j) + 1) = Cm(ImageMatrix(i, j) + 1) + 1;
    for KH = -Distance : Distance
      for KW = -Distance : Distance
        if ((abs(KH) ~= Distance) & (abs(KW) ~= Distance))
          continue;   %% One of (KH, KW) must be Distance
        end
        checks(ImageMatrix(i, j) + 1) = checks(ImageMatrix(i, j) + 1) + 1;
      end
  end
end
Rtn = Sum./checks;
```

```
function RtnIdx = Circular(OriginalIdx, Limit)
RtnIdx = OriginalIdx;
if OriginalIdx <= 0
  RtnIdx = OriginalIdx + Limit;
else
  if OriginalIdx > Limit
    RtnIdx = OriginalIdx - Limit;
  end
end

function RtnIdx = Reflect(OriginalIdx, Limit)
RtnIdx = OriginalIdx;
if OriginalIdx <= 0
  RtnIdx = -OriginalIdx + 1;
else
  if OriginalIdx > Limit
    RtnIdx = 2 * Limit - OriginalIdx + 1;
  end
end
```

Data Files **(.mat file)**

- Write data to a **.mat** file
- save *fileName*
- save *fileName* *variable1* *variable2 …*
- save *fileName* *variable1* *variable2 …* –asci

- Read in data stored in matrices
- load *fileName*
- load *fileName* *matrixVariable*

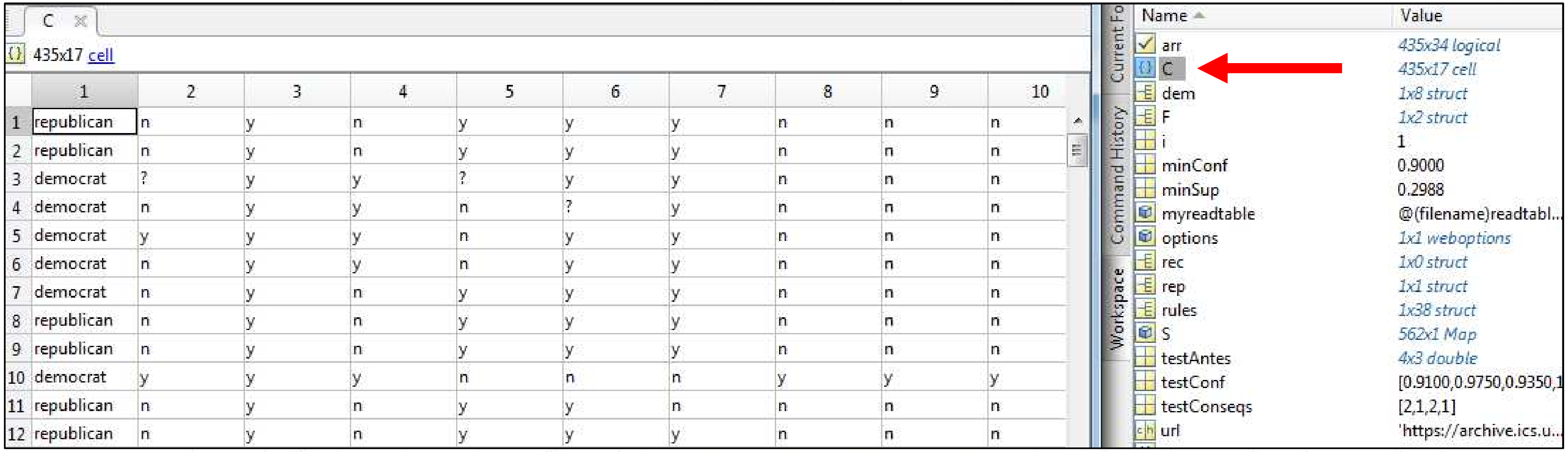
- Load data from a file and plot the data

```
a = [1 2 3];
save('myFile', 'a');
clear a;
a = load('myFile');
plot(a);
```

```
np.savetxt('D:\\folder1\\folder2\\’ + FileName, yourData, delimiter=',’)

newData = np.genfromtxt('D:\\folder\\YourFile.csv', delimiter = ',')
```

- **xlsread(), xlswrite()**

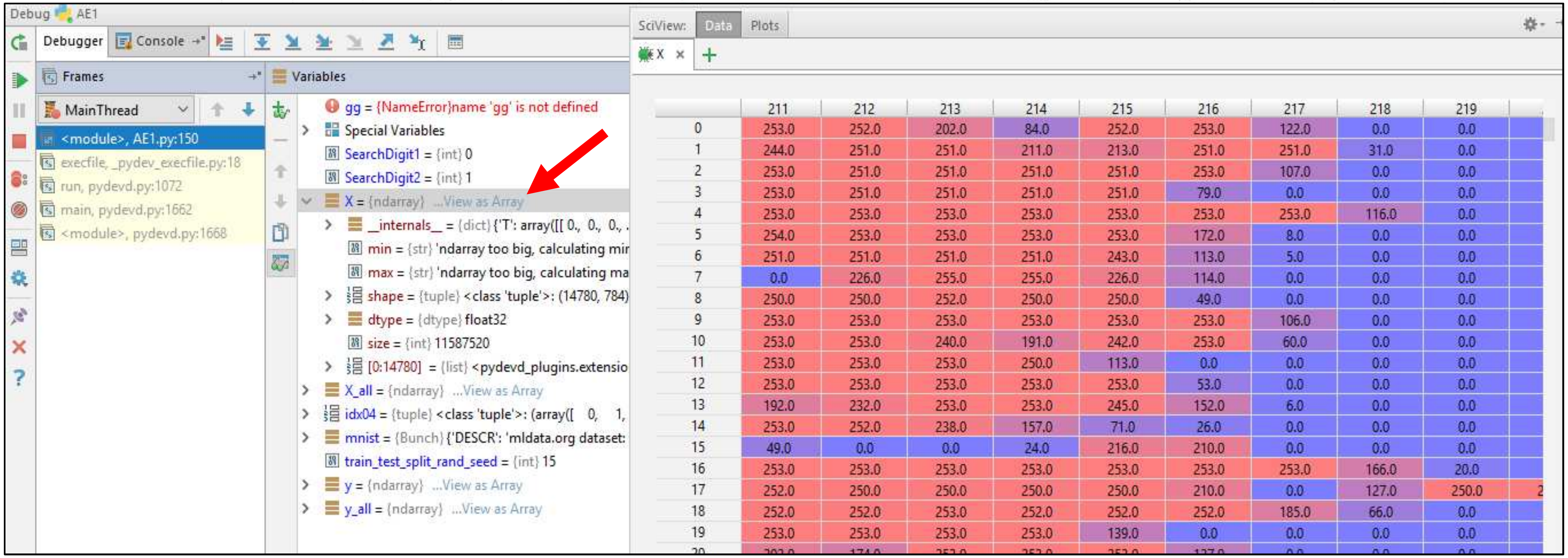
Debugging Large Datasets in Matlab & PyCharm



435x17 cell

	1	2	3	4	5	6	7	8	9	10
1	republican	n	y	n	y	y	y	n	n	n
2	republican	n	y	n	y	y	y	n	n	n
3	democrat	?	y	y	?	y	y	n	n	n
4	democrat	n	y	y	n	?	y	n	n	n
5	democrat	y	y	y	n	y	y	n	n	n
6	democrat	n	y	y	n	y	y	n	n	n
7	democrat	n	y	n	y	y	y	n	n	n
8	republican	n	y	n	y	y	y	n	n	n
9	republican	n	y	n	y	y	y	n	n	n
10	democrat	y	y	y	n	n	n	y	y	y
11	republican	n	y	n	y	y	n	n	n	n
12	republican	n	y	n	y	y	y	n	n	n

Name	Value
arr	435x34 logical
C	435x17 cell
dem	1x8 struct
F	1x2 struct
i	1
minConf	0.9000
minSup	0.2988
myreadtable	@(filename)readtabl...
options	1x1 weboptions
rec	1x0 struct
rep	1x1 struct
rules	1x38 struct
S	562x1 Map
testAntes	4x3 double
testConf	[0.9100,0.9750,0.9350,1
testConseqs	[2,1,2,1]
url	'https://archive.ics.u...



```
MainThread
<module>, AE1.py:150
execfile, _pydev_execfile.py:18
run, pydevd.py:1072
main, pydevd.py:1662
<module>, pydevd.py:1668
```

```
gg = {NameError}name 'gg' is not defined
> Special Variables
SearchDigit1 = {int} 0
SearchDigit2 = {int} 1
X = {ndarray} ...View as Array
  > _internals_ = {dict} {'T': array([[ 0.,  0.,  0., .
    min = {str} 'ndarray too big, calculating mir
    max = {str} 'ndarray too big, calculating ma
  > shape = {tuple} <class 'tuple'>: (14780, 784)
  > dtype = {dtype} float32
    size = {int} 11587520
  > [0:14780]  = {list} <pydevd_plugins.extensio
> X_all = {ndarray} ...View as Array
> idx04 = {tuple} <class 'tuple'>: (array([   0,    1,
> mnist = {Bunch} {'DESCR': 'mldata.org dataset:
train_test_split_rand_seed = {int} 15
> y = {ndarray} ...View as Array
> y_all = {ndarray} ...View as Array
```

	211	212	213	214	215	216	217	218	219
0	253.0	252.0	202.0	84.0	252.0	253.0	122.0	0.0	0.0
1	244.0	251.0	251.0	211.0	213.0	251.0	251.0	31.0	0.0
2	253.0	251.0	251.0	251.0	251.0	253.0	107.0	0.0	0.0
3	253.0	251.0	251.0	251.0	251.0	79.0	0.0	0.0	0.0
4	253.0	253.0	253.0	253.0	253.0	253.0	253.0	116.0	0.0
5	254.0	253.0	253.0	253.0	253.0	172.0	8.0	0.0	0.0
6	251.0	251.0	251.0	251.0	243.0	113.0	5.0	0.0	0.0
7	0.0	226.0	255.0	255.0	226.0	114.0	0.0	0.0	0.0
8	250.0	250.0	252.0	250.0	250.0	49.0	0.0	0.0	0.0
9	253.0	253.0	253.0	253.0	253.0	253.0	106.0	0.0	0.0
10	253.0	253.0	240.0	191.0	242.0	253.0	60.0	0.0	0.0
11	253.0	253.0	253.0	250.0	113.0	0.0	0.0	0.0	0.0
12	253.0	253.0	253.0	253.0	253.0	53.0	0.0	0.0	0.0
13	192.0	232.0	253.0	253.0	245.0	152.0	6.0	0.0	0.0
14	253.0	252.0	238.0	157.0	71.0	26.0	0.0	0.0	0.0
15	49.0	0.0	0.0	24.0	216.0	210.0	0.0	0.0	0.0
16	253.0	253.0	253.0	253.0	253.0	253.0	253.0	166.0	20.0
17	252.0	250.0	250.0	250.0	250.0	210.0	0.0	127.0	250.0
18	252.0	252.0	253.0	252.0	252.0	252.0	185.0	66.0	0.0
19	253.0	253.0	253.0	253.0	139.0	0.0	0.0	0.0	0.0

Path

- Matlab will only use those functions and data files that are in its path

- To add *c:\dm\test\hw1* to the path, type:

```
>> p = path;
>> path(p, 'c:\dm\test\hw1' );
```

- You can also use MatLab GUI to add path
  - Under the ‘File’ menu item

Shuffle and Partitioning

- Matlab CV patitioning

```
% random selection
TotRecs = size(Y, 1);
ShuffleIdx = randperm(TotRecs);
X = X(ShuffleIdx, :);
Y = Y(ShuffleIdx, :);

% random selection w/ portioning
CVP = cvpartition(Y, 'HoldOut', 0.3);
% CVP contains indices for training / test
trainX = X(CVP.training, :);
trainY = Y(CVP.training, :);
testX = X(CVP.test, :);
testY = Y(CVP.test, :);
```

sklearn.utils.shuffle

sklearn.model_selection.train_test_split

sklearn.model_selection.Kfold

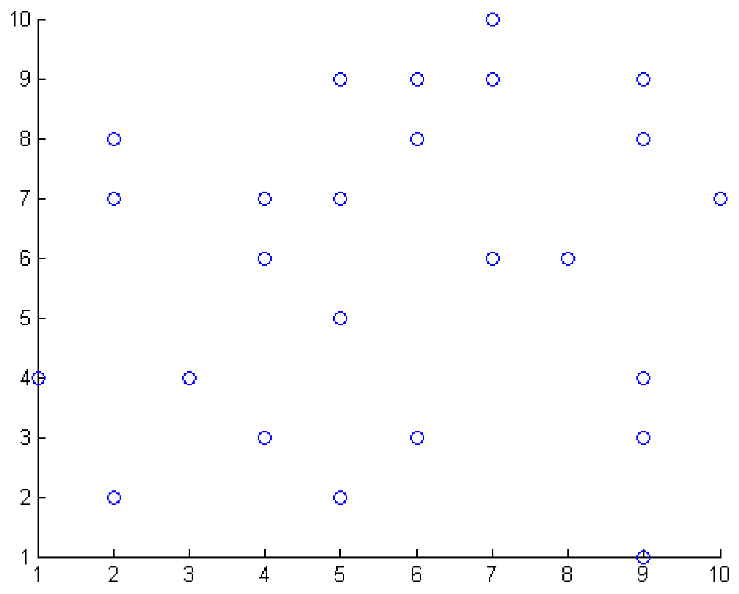
sklearn.model_selection.StratifiedShuffleSplit

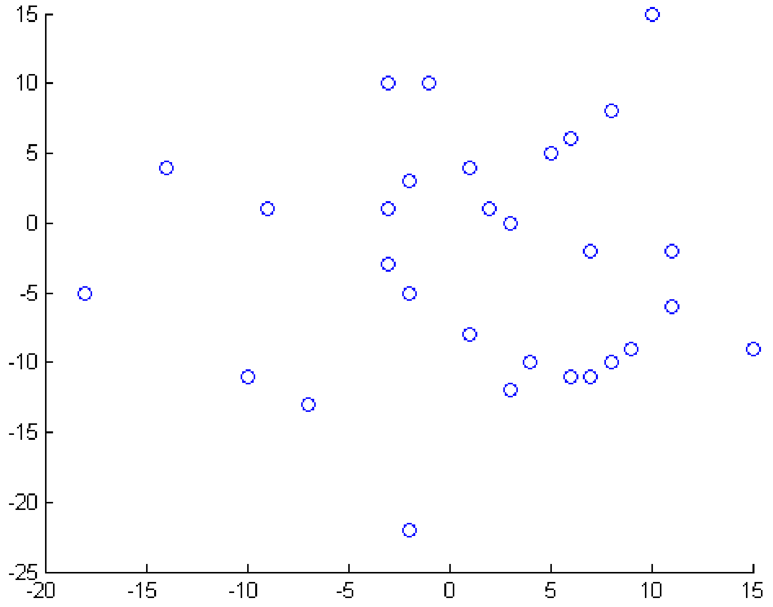
Scatter Plot of Random Data

- ❑ **scatter()** function
  - ❑ Create a scatter plot w/ circles at the locations specified by the vectors x and y

```
rng(3)

x = ceil(rand(300, 2) * 10);

scatter(x(:, 1), x(:, 2));

x1 = ceil(randn(300, 2) * 10);

figure, scatter(x1(:, 1), x1(:, 2));
```

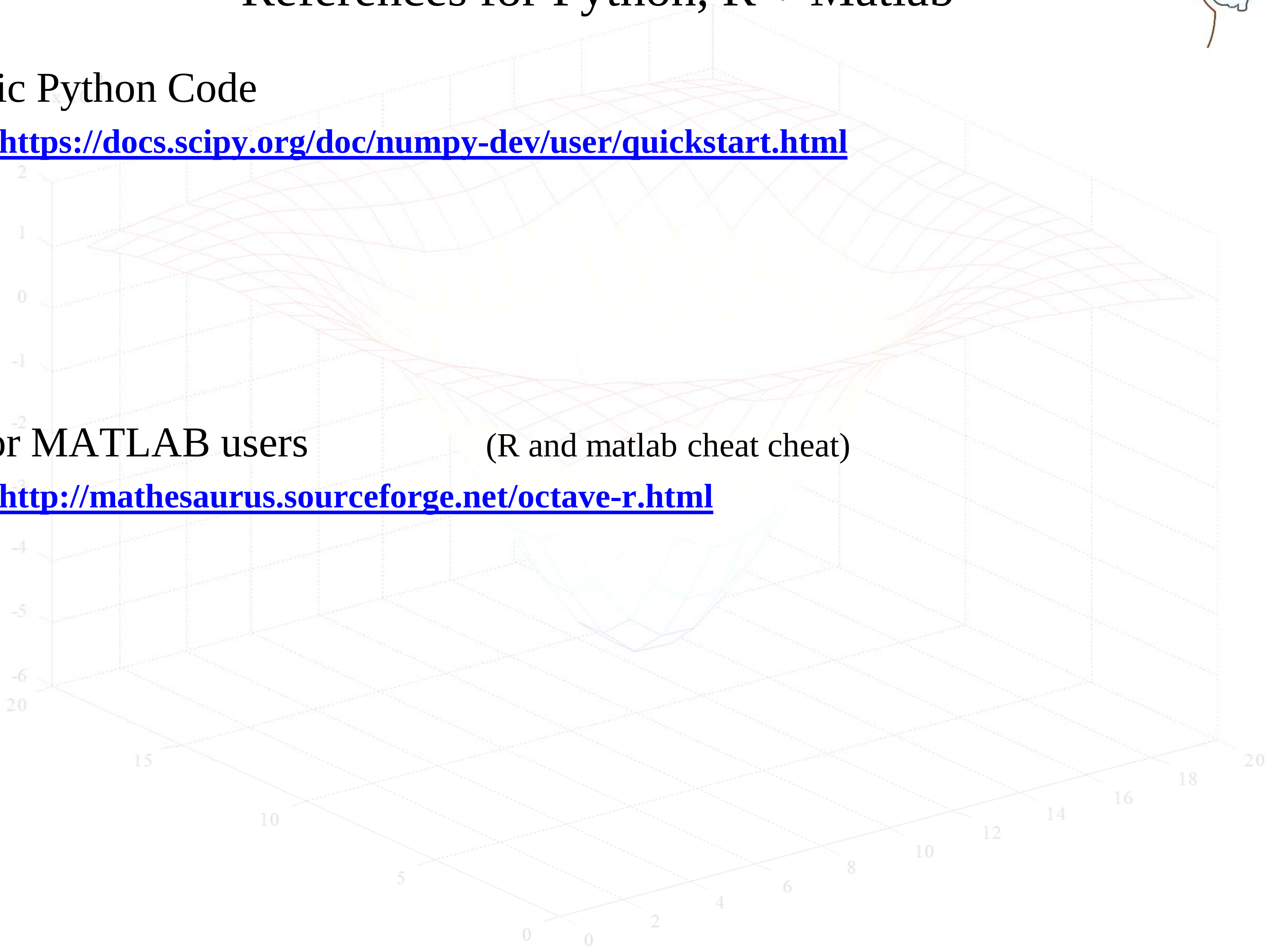
```
import matplotlib, matplotlib.pyplot as plt
plt.interactive(False)   # ←←←
plt.plot(…)
```





References for Python, R + Matlab

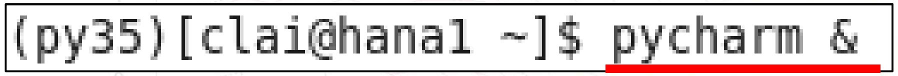
- Basic Python Code
  - **https://docs.scipy.org/doc/numpy-dev/user/quickstart.html**



- R for MATLAB users (R and matlab cheat cheat)
  - **http://mathesaurus.sourceforge.net/octave-r.html**

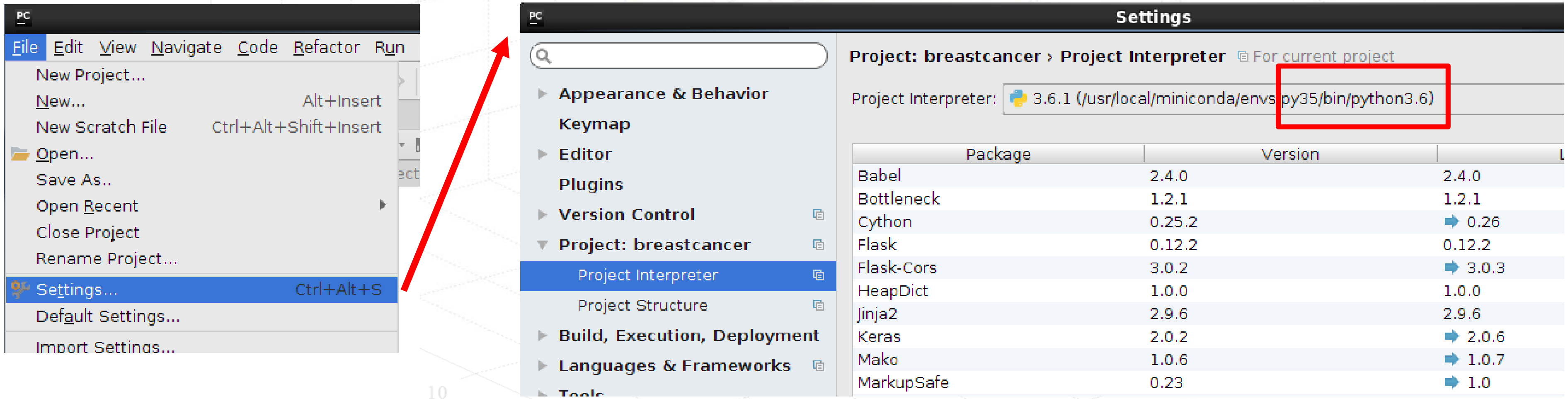
Python on PyCharm

- Basic Python Code
  - https://docs.scipy.org/doc/numpy-dev/user/quickstart.html



```
(py35)[clai@hana1 ~]$ pycharm &
```

- Change Python interpreter in PyCharm



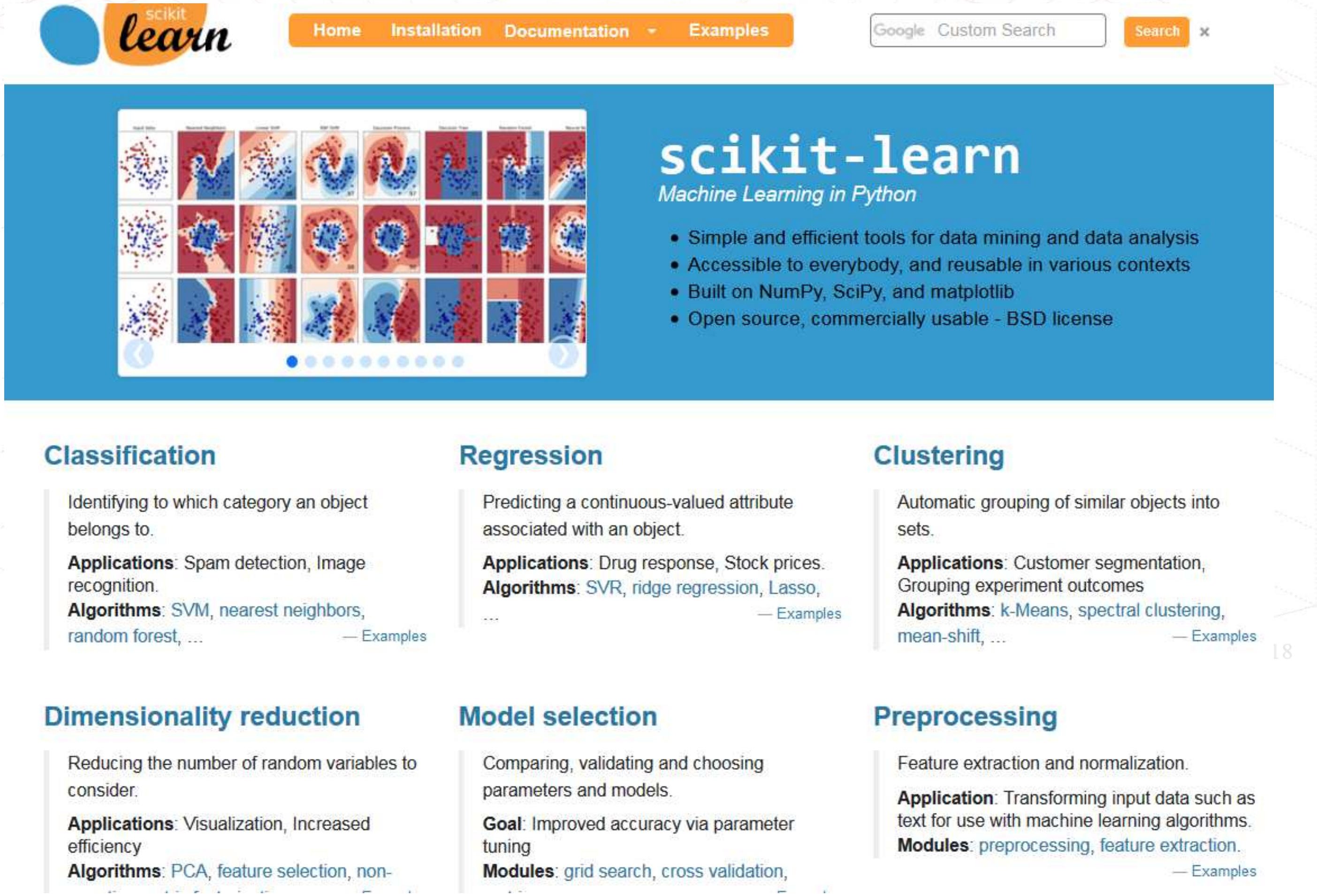
Project: breastcancer › Project Interpreter For current project

Project Interpreter: 3.6.1 (/usr/local/miniconda/envs/py35/bin/python3.6)

Package	Version	L
Babel	2.4.0	2.4.0
Bottleneck	1.2.1	1.2.1
Cython	0.25.2	➡ 0.26
Flask	0.12.2	0.12.2
Flask-Cors	3.0.2	➡ 3.0.3
HeapDict	1.0.0	1.0.0
Jinja2	2.9.6	2.9.6
Keras	2.0.2	➡ 2.0.6
Mako	1.0.6	➡ 1.0.7
MarkupSafe	0.23	➡ 1.0

Machine Learning in Python

- **scikit-learn**
  - http://scikit-learn.org/stable/



scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.

Algorithms: SVM, nearest neighbors, random forest, ...

— Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: SVR, ridge regression, Lasso, ...

— Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: k-Means, spectral clustering, mean-shift, ...

— Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Algorithms: PCA, feature selection, non-

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning

Modules: grid search, cross validation,

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.

Modules: preprocessing, feature extraction.

— Examples

Installed Packages (PyCharm)

- Install
  - tensorflow
  - keras

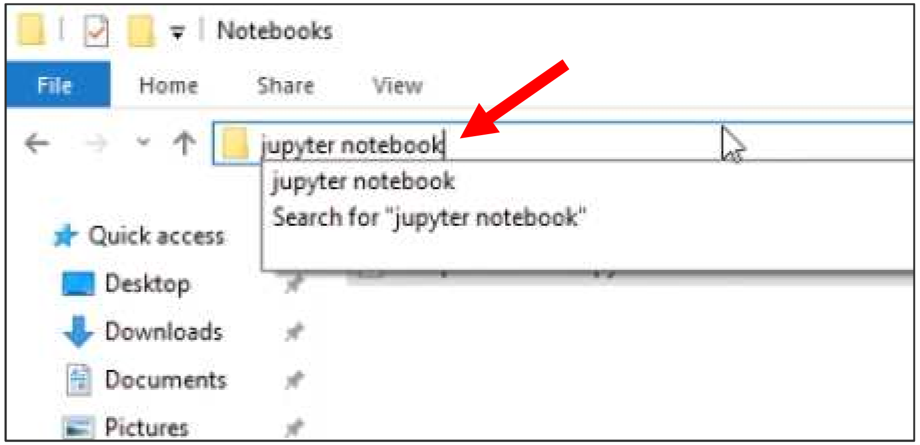
  - panda–
  - matplotlib–
  - numpy– multi-dimensional array, matrix, math functions
  - scipy– scientific computing.
  - scikit-learn– ML and data exploration.
  - pillow– for image processing.
  - h5py–data serialization by Keras for model saving.

  - ipython
  - pydot
  - Graphviz

  - mlxtend # ARs
  - xlrd # read xlsx

More Installations **(Anaconda)** + Run Jupyter Notebook

- pip install --ignore-installed --upgrade https://storage.googleapis.com/tensorflow/windows/gpu/**tensorflow**_gpu-1.0.1-cp35-cp35m-win_amd64.whl



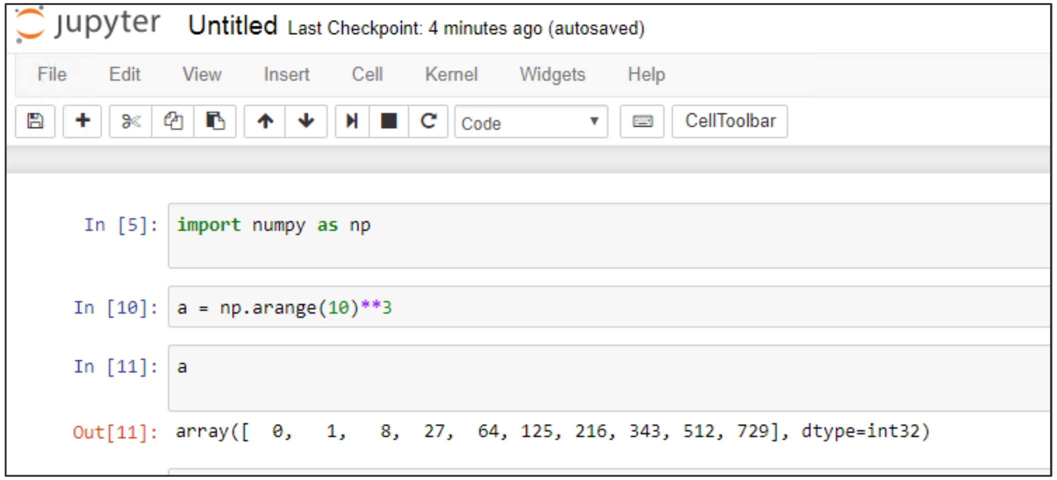
jupyter notebook

jupyter notebook
Search for "jupyter notebook"

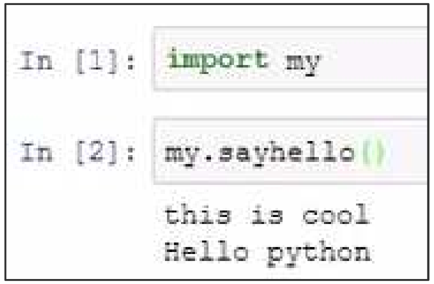
Install package under Anaconda

1 .Conda install [your_Package]

2. Pip install **sklearn**
   [also **scipy, tensorflow, keras**]



```
In [5]: import numpy as np

In [10]: a = np.arange(10)**3

In [11]: a

Out[11]: array([  0,   1,   8,  27,  64, 125, 216, 343, 512, 729], dtype=int32)
```

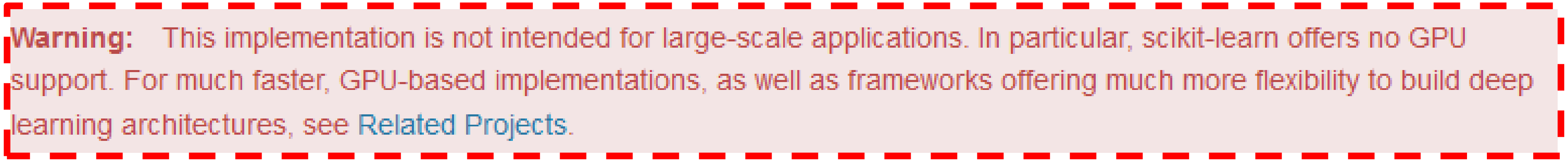


```
In [1]: import my

In [2]: my.sayhello()

        this is cool
        Hello python
```

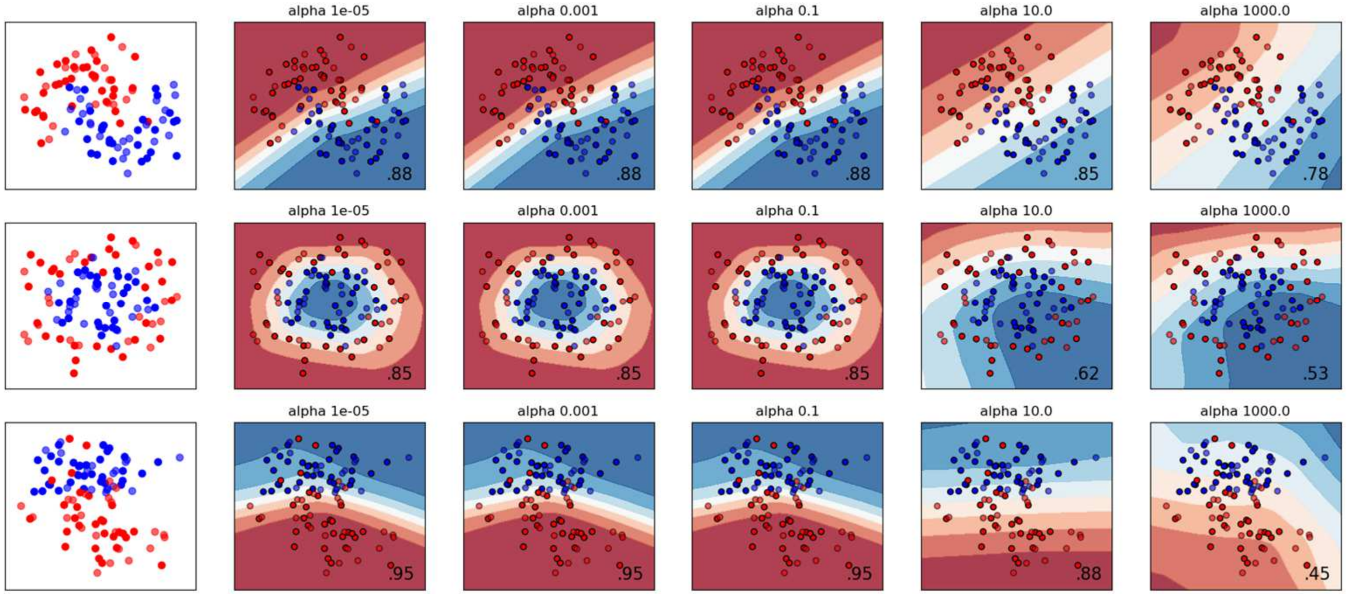
```
C:\> ipython nbconvert --to=python [YOUR_NB].ipynb
```

sklearn Neural Network Package

- http://scikit-learn.org/stable/modules/neural_networks_supervised.html



Warning: This implementation is not intended for large-scale applications. In particular, scikit-learn offers no GPU support. For much faster, GPU-based implementations, as well as frameworks offering much more flexibility to build deep learning architectures, see Related Projects.


alpha 1e-05
alpha 0.001
alpha 0.1
alpha 10.0
alpha 1000.0
alpha 1e-05
alpha 0.001
alpha 0.1
alpha 10.0
alpha 1000.0
alpha 1e-05
alpha 0.001
alpha 0.1
alpha 10.0
alpha 1000.0