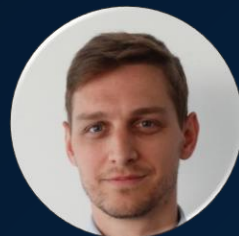




## MATLAB with Python

*Heather Gorr, PhD & Yann Debray & Shubo Chakrabarti*

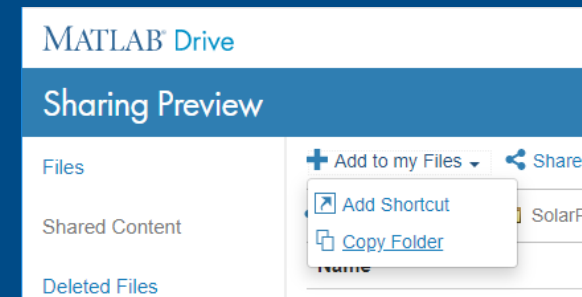


# Setup instructions

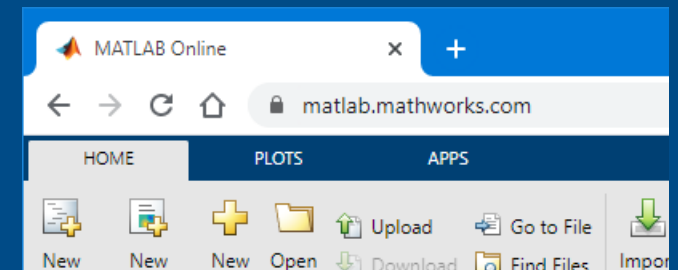
<https://drive.matlab.com/sharing/10a651a2-5e31-4408-97d6-89a22980ce46>

Step 1: Click on the link in the chat to get the files for today's session

Step 2: In MATLAB Drive, click on **Add to my Files** → **Copy Folder**



Step 3: Go to [matlab.mathworks.com](https://matlab.mathworks.com) and log into MATLAB Online using your MathWorks account credentials



Step 4: Open the folder called MATLABWithPythonWorkshop

# Sharing Preview

- Files
- Shared Content
- Deleted Files

[Go to Drive](#) | 
 [Download Shared Folder](#)

[Shared Content](#) > 
 [MATLABWithPythonWorkshop](#)

Name	Size	Date Modified
▶ <a href="#">.ipynb_checkpoints</a>		3/4/2022 00:59
▶ <a href="#">__pycache__</a>		3/24/2022 16:37
▶ <a href="#">SupportingMaterials</a>		4/13/2022 14:52
<a href="#">accessKey.txt</a>	1 KB	4/7/2019 22:11
<a href="#">airQualModel.mat</a>	133 KB	3/31/2021 14:41
<a href="#">CallMATLABfromPython.ipynb</a>	10 KB	4/13/2022 13:21
<a href="#">CallPythonFromMATLAB.mlx</a>	264 KB	4/13/2022 15:05
<a href="#">cities.mat</a>	833 KB	3/31/2021 14:41
<a href="#">large_pizza.py</a>	1 KB	3/24/2022 15:09
<a href="#">predictAirQual.m</a>	1 KB	3/24/2022 16:38
<a href="#">prepData.m</a>	3 KB	3/31/2021 14:41
<a href="#">small_pizza.py</a>	1 KB	4/13/2022 14:36

>> [MATLABWithPythonWorkshop](#)

Owned By: Heather Gorr

Access Type: Can Edit [i](#)

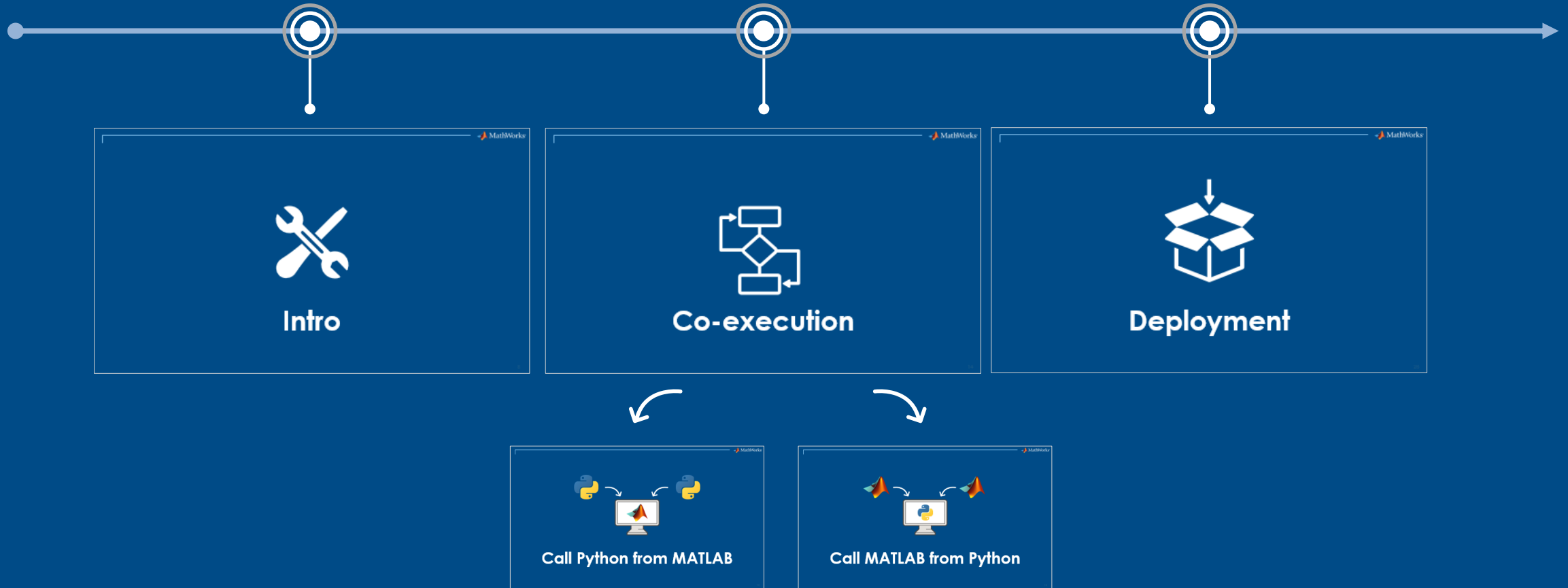
Created On: jeu. 03 mars 2022

Modified On: mer. 13 avr. 2022

Folder Size: 51.6 MB


  
 4.0 GB / 5 GB used [i](#)

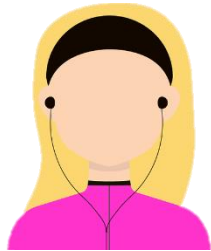
# Agenda





# Intro

## Common problematics



Engineer

I would like to **integrate my colleagues' Python codes** in my MATLAB simulations models, is it possible?

Call Python code directly from MATLAB



Data Scientist

Can I use MATLAB models in my Python code if I **don't have any MATLAB installed**?

Yes! You can build Python's libraries from your MATLAB code in one click



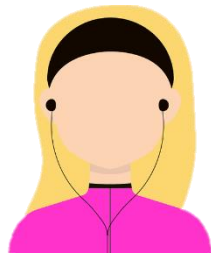
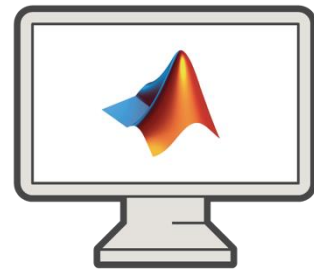
IT / DevOps

We need to put our whole process (MATLAB + Python) in production

Deploy MATLAB models as micro services and integrate with Python code



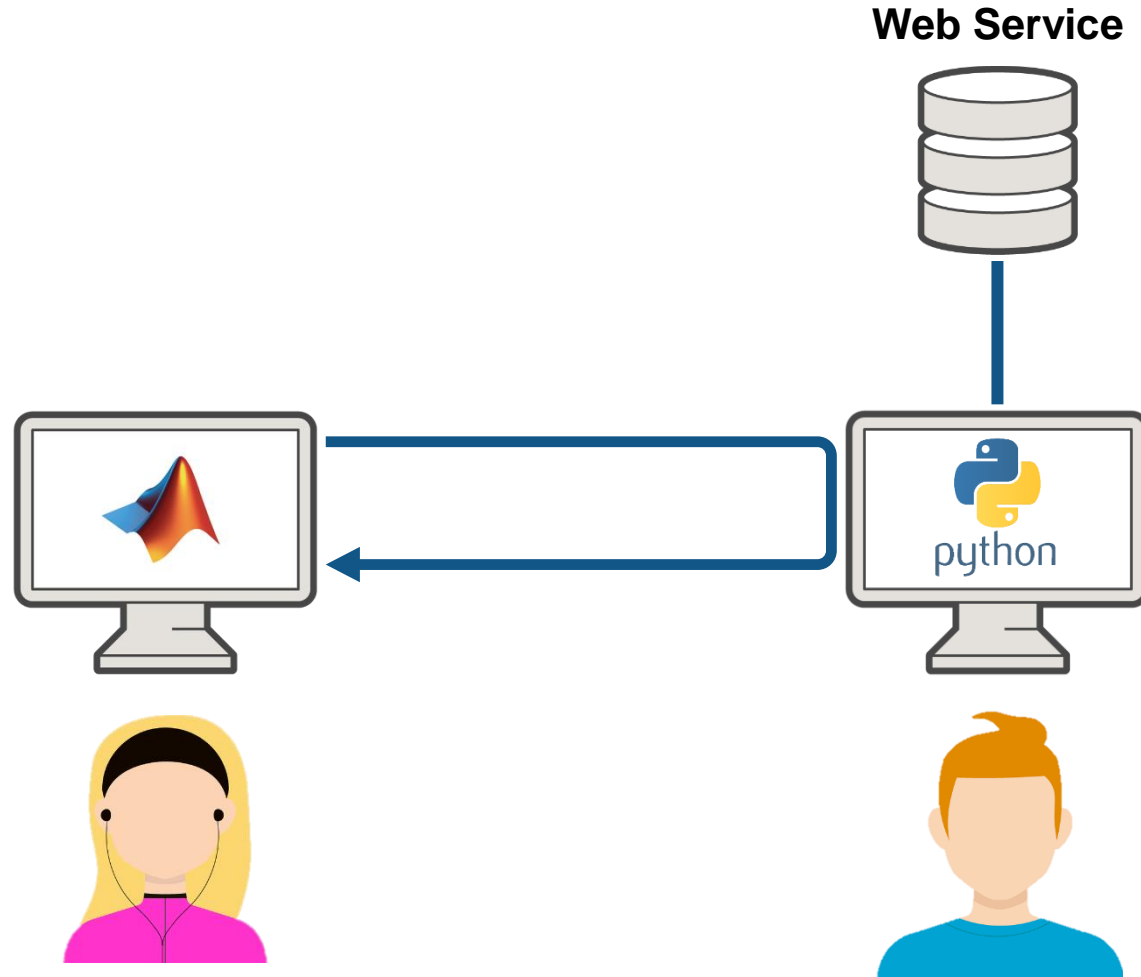
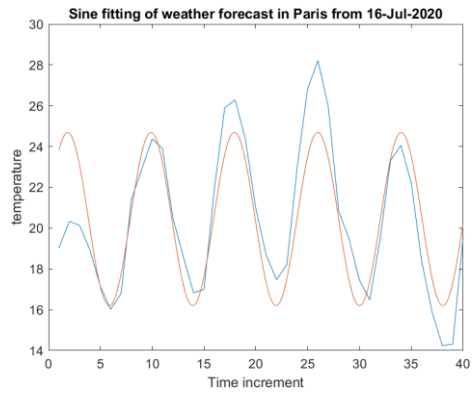
## Given: Existing Python Code accessing & preparing data



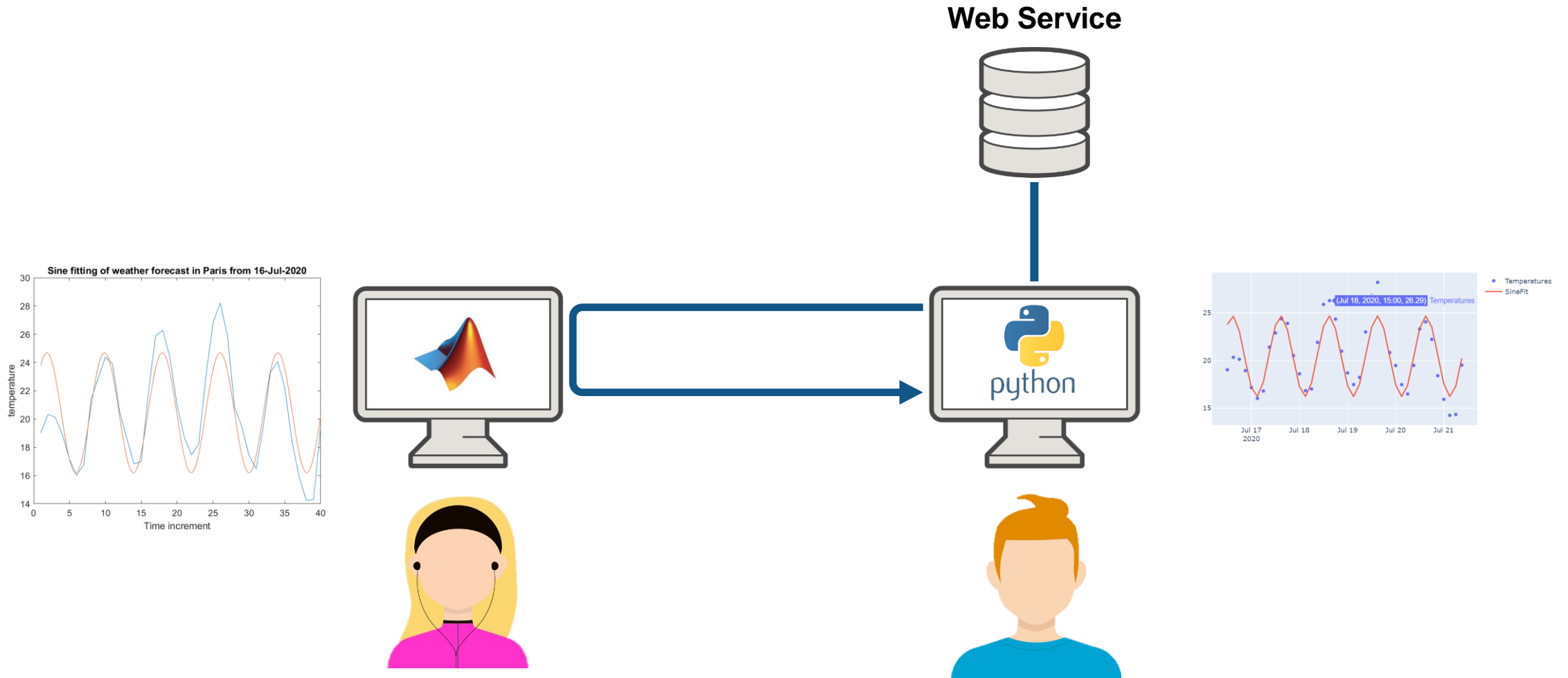
**Web Service**



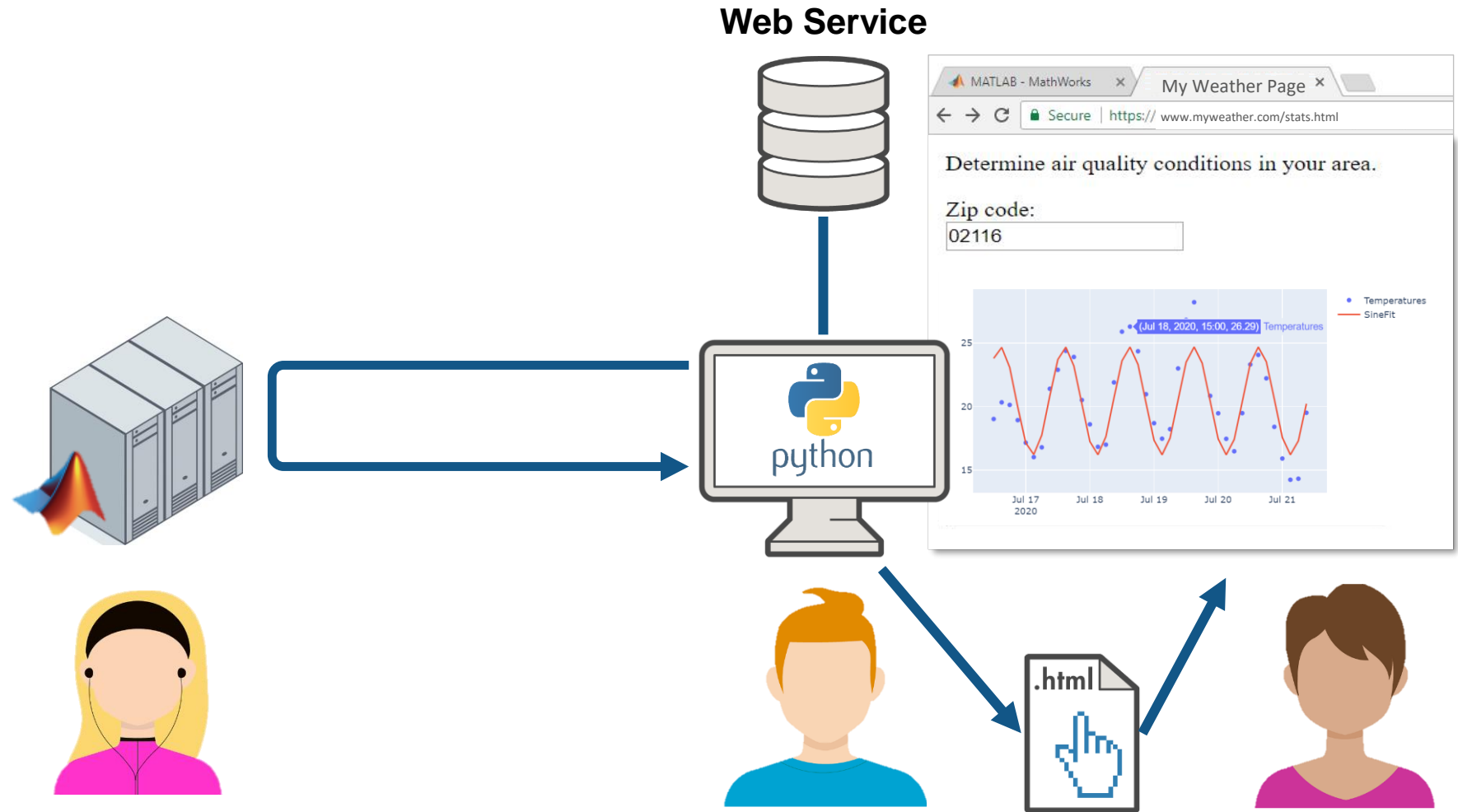
# Call Python from MATLAB

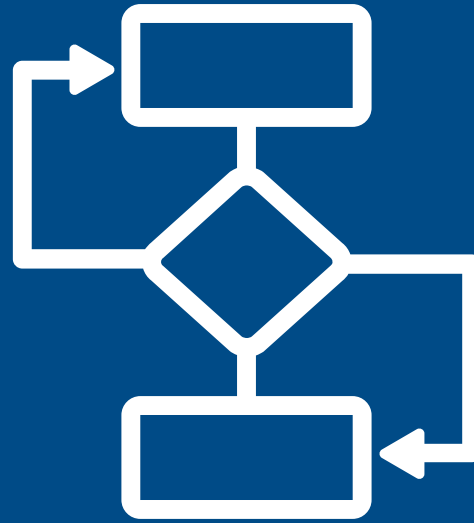


# Call MATLAB from Python



# Deploy: MATLAB Models in Python

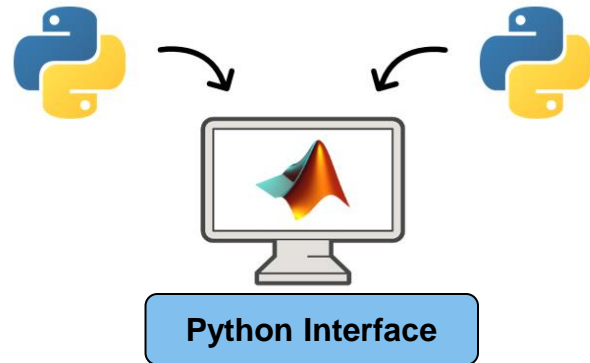




# Co-execution

# MATLAB + Python

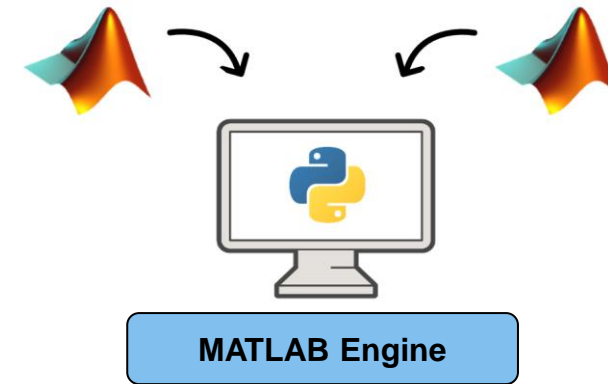
## Calling Python from MATLAB



Already working in MATLAB, and:

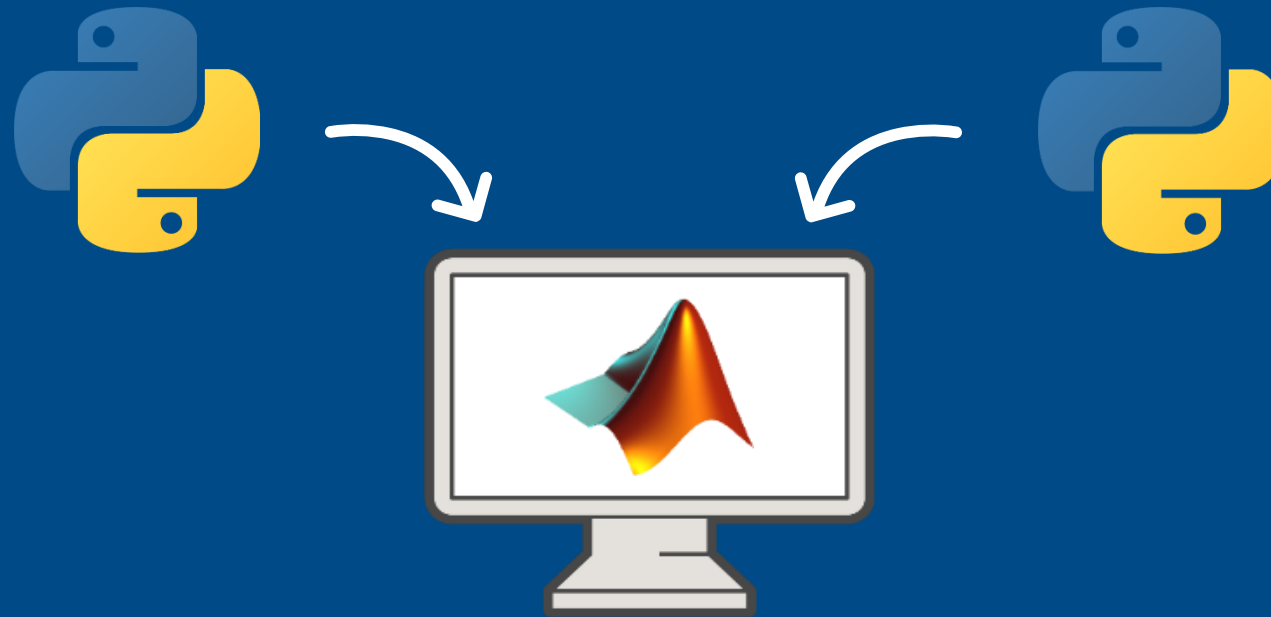
- Want to reuse existing Python code
- Need functionality that is only available in Python

## Calling MATLAB from Python



Already working in Python, and:

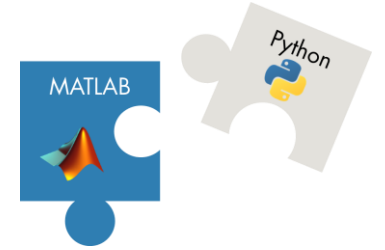
- Want to reuse existing MATLAB code
- Need functionality available in MATLAB
- Want to collaborate with MATLAB users



# Call Python from MATLAB

# Call Python from MATLAB

## Set-up your Python environment



```
pyenv
```

```
ans =
```

```
PythonEnvironment with properties:
```

```
Version: "3.9"  
Executable: "C:\Users\ydebray\AppData\Local\WPY64-39100\python-3.9.10.amd64\python.exe"  
Library: "C:\Users\ydebray\AppData\Local\WPY64-39100\python-3.9.10.amd64\python39.dll"  
Home: "C:\Users\ydebray\AppData\Local\WPY64-39100\python-3.9.10.amd64"  
Status: NotLoaded  
ExecutionMode: OutOfProcess
```



mathworks / MATLAB-Live-Task-for-Python Public

Watch 6 Fork 7 Starred 31

- Code
- Issues 2
- Pull requests
- Actions
- Projects
- Wiki
- Security
- Insights

master 1 branch 1 tag

[Go to file](#) [Add file](#) [Code](#)

Lucas Garcia Updating README		acdfebd 20 days ago	2 commits
examples	Initial commit		27 days ago
img	Initial commit		27 days ago
src	Initial commit		27 days ago
CONTRIBUTING.md	Initial commit		27 days ago
LICENSE.txt	Initial commit		27 days ago
README.md	Updating README		20 days ago
SECURITY.md	Initial commit		27 days ago
install.m	Initial commit		27 days ago
uninstall.m	Initial commit		27 days ago

### About

The MATLAB® Live Task for Python® enables you to write and execute Python code directly inside of a MATLAB Live Script.

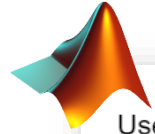
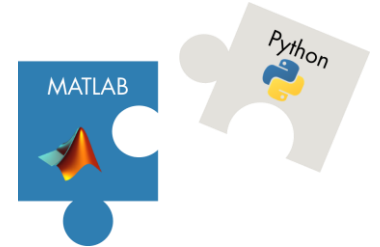
[python](#) [matlab](#) [matlab-python](#)

- Readme
- BSD-2-Clause license
- 31 stars
- 6 watching
- 7 forks

### Releases 1

Initial release [Latest](#)  
 26 days ago

# Calling Python libraries from MATLAB



Use the weather.py module to get the air quality for Paris. This is a user-defined Python module which includes functions to read and parse the current and forecasted weather data by location.

```
jsonData = py.weather.get_current_weather("Paris", "France", apikey.Key)
```

```
jsonData =  
Python dict with no properties.
```

```
{'coord': {'lon': 2.35, 'lat': 48.85}, 'weather': [{'id': 803, 'main': 'Cloudy'}
```

Parse the json data returned from the weather API.

The Python dictionary can be represented as a MATLAB struct.

```
weatherData = py.weather.parse_json(jsondata);  
struct(weatherData)
```

```
ans = struct with fields:  
    temp: 18.7100  
 feels_like: 17.3000  
    temp_min: 17.7800  
    temp_max: [1x1 py.int]
```

Use a function (prepData.m) to prepare data for machine learning (create a table with the expected variable names, preprocessing steps, etc).

```
currentData = prepData(weatherData)
```

currentData = 1x12 table

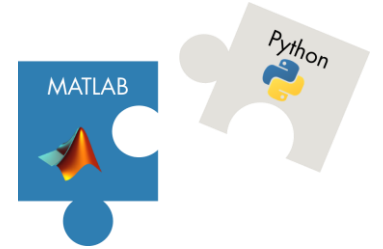
	DateLocal	city	StateName	T	P	DP	RH	WindDir	WindSpd		
1	01-Jul-2020 11:...	"Paris"	Ile de France	21.6200	20.2600	349.2200	1010	5.1000	73		

```
def get_current_weather(city, country, apikey):  
    # get current conditions in specified location  
    # get_current_weather('boston', 'us', key)  
    import urllib.request  
    import json  
    # read current conditions  
    try:  
        url = "https://api.openweathermap.org/data/2.5/weather?q="+city+", "+country+"&appid="+apikey  
        response = urllib.request.urlopen(url)  
        html = response.read()  
        json_data = json.loads(html)  
  
    except urllib.error.URLError:  
        # if weather API doesnt work, read the file  
        json_data = read_backup(city)  
  
    return json_data
```



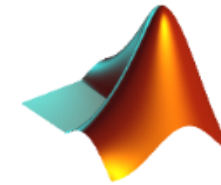
# Call Python from MATLAB

## Syntax differences



Python

```
>>> import math
>>> math.sqrt(42)
```



MATLAB

```
>> py.math.sqrt(42)
```

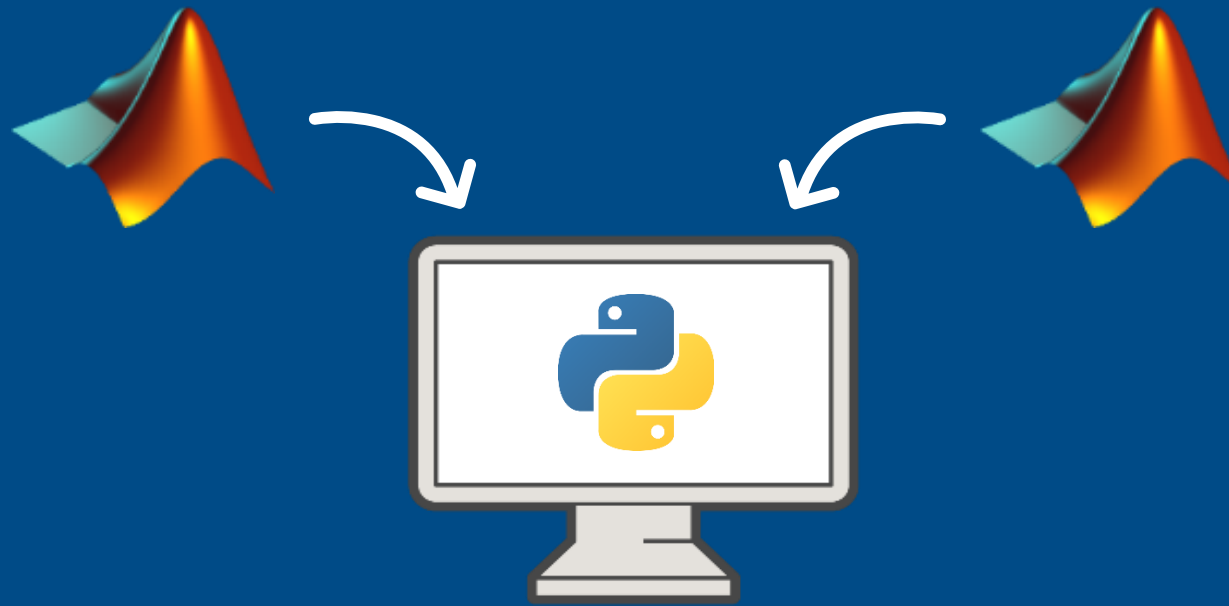
```
>>> print('hello', 'world', sep=', ')
```



```
>> py.print('hello', 'world', ...
            sep = ', ')
```

Before  
R2022a

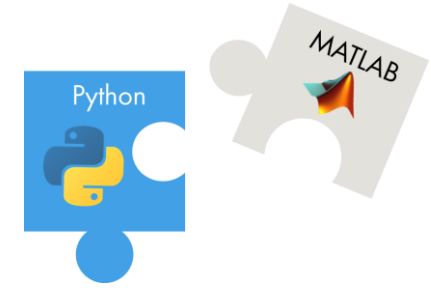
```
>> py.print('hello', 'world', ...
            pyargs('sep', ', '))
```



# Call MATLAB from Python

# Call MATLAB from Python

## MATLAB Engine API



- **NEW** Install MATLAB Engine from the Python Package Index  

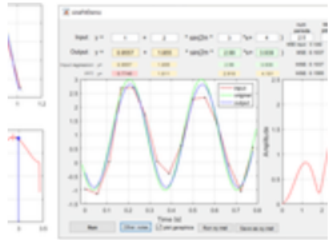
```
$ pip install matlabengine
```
- Start a MATLAB process  

```
>>> import matlab.engine  
>>> m = matlab.engine.start_matlab()
```
- Call MATLAB functions  
*(pay attention to the expected datatype and number of outputs)*  

```
>>> x = m.sqrt(42)  
>>> s = m.sort(matlab.double([1,3,2]),nargout=2)  
>>> m.workspace['s'] = s
```

# File Exchange

[File Exchange](#)

[MATLAB Central](#)
[Files](#)
[Authors](#)
[My File Exchange](#)
[Publish](#)
[About](#)
[Trial software](#)


## Sine fitting

 version 3.0.2 (1.67 MB) by [Peter Seibold](#)

Determine parameters of a noisy sine function

★★★★★ (7)

2.2K Downloads

Updated 08 Dec 2020

[View Version History](#)
[View License](#)
[+ Follow](#)
[Download](#)
[Overview](#)
[Functions](#)
[Reviews \(7\)](#)
[Discussions \(5\)](#)

sineFit is a function to detect the parameters of a noisy sine curve, even less than one period long.

It requires only x and y values and no additional parameters as input.

It is tested with R2016a and R2020a.

The mean calculation time is on my PC 13 ms with a maximum of 2400 ms.

Syntax:

```
[SineParams]=sineFit(x,y,optional)
```

optional: plot graphics if omitted. Do not plot if 0

Input:

x and y values,  $y = \text{offs} + \text{amp} * \sin(2\pi * f * x + \text{phi}) + \text{noise}$

Output:

### Requires

[MATLAB](#)

### MATLAB Release Compatibility

Created with R2016a

Compatible with R2020a and later releases

### Platform Compatibility

 Windows  macOS  Linux

### Categories

☰ README.md

## Build Instructions

---

### Get Sources

```
# Clone this repository to your machine
git clone https://github.com/mathworks-ref-arch/matlab-integration-for-jupyter.git

# Navigate to the downloaded folder
cd matlab-integration-for-jupyter/matlab
```

### Build & Run Docker Image

```
# Build container with a name and tag of your choice.
docker build -t matlab-notebook .
```

Start the container, and forward the default Jupyter web-app port (8888) to the host machine:

```
docker run -it --rm -p 8888:8888 matlab-notebook
```

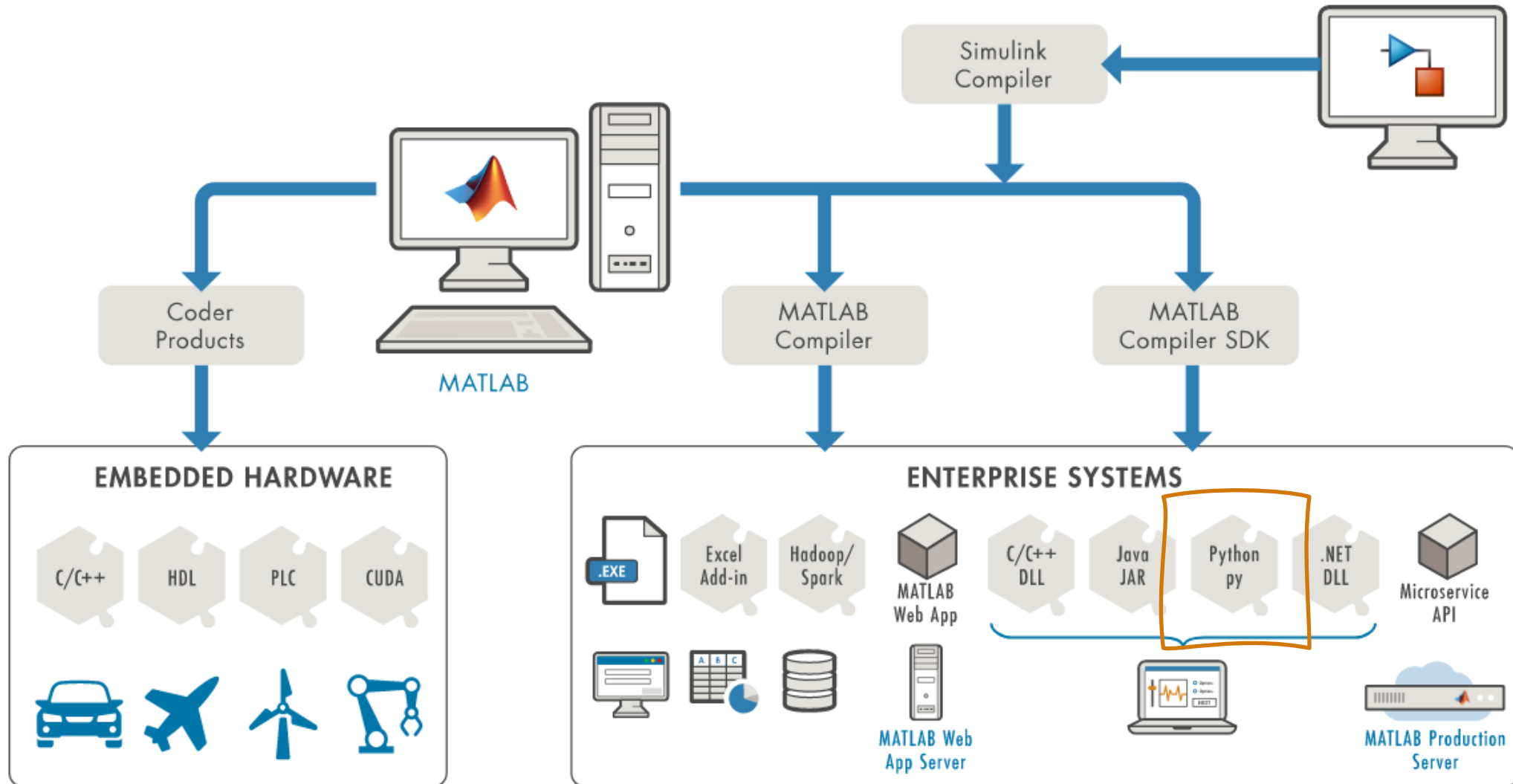
### Customize the Image

---

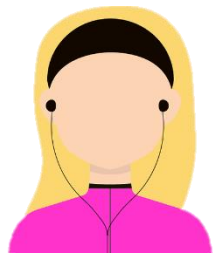


# Deployment

# Deployment of MATLAB code as Python package



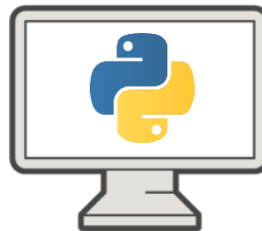
# Generate Python package



Engineer

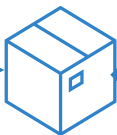


Data Scientist



Can I use MATLAB models in my Python files if I don't have any MATLAB installed?

```
Py_package =
compiler.build.
pythonPackage()
```



```
Deploy Python package with Compiler SDK

Export the SineFit MATLAB function to a Python Package to remove dependency to MATLAB in deployment https://www.mathworks.com/help/compiler_sdk/python/pass-data-to-matlab-from-python.html

cd for_redistribution_files_only
python setup.py install

import matlab
import sinefit

import urllib.request
import json

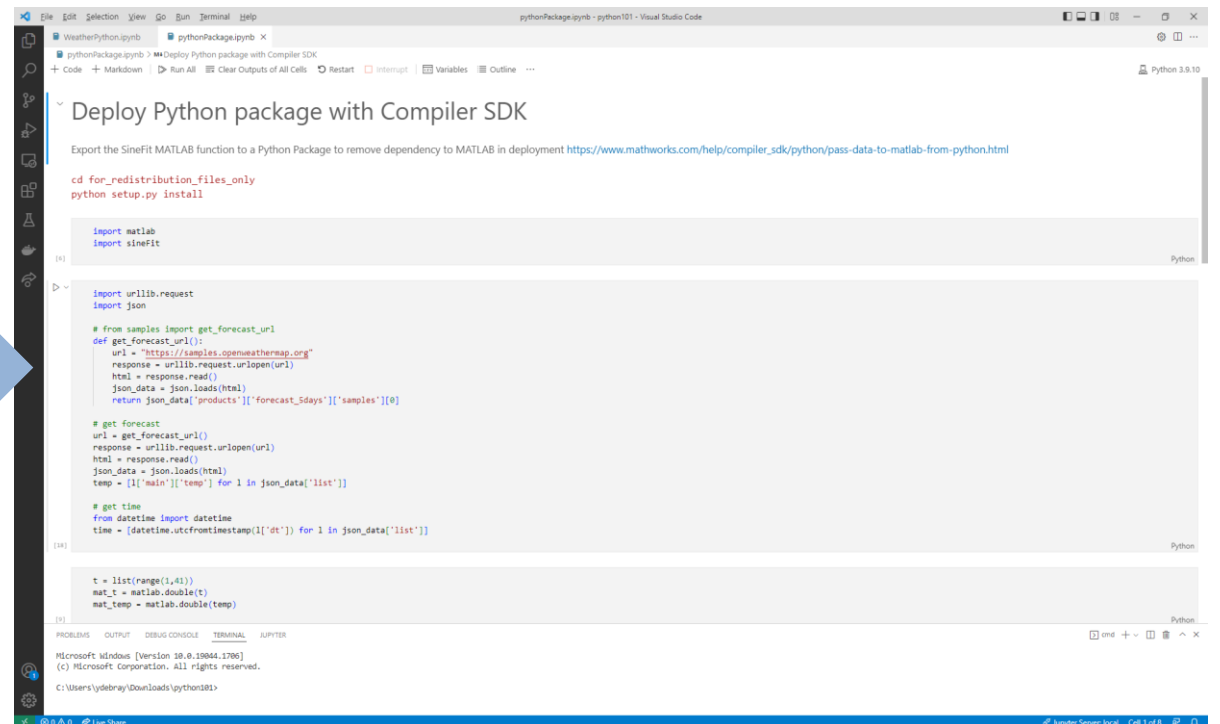
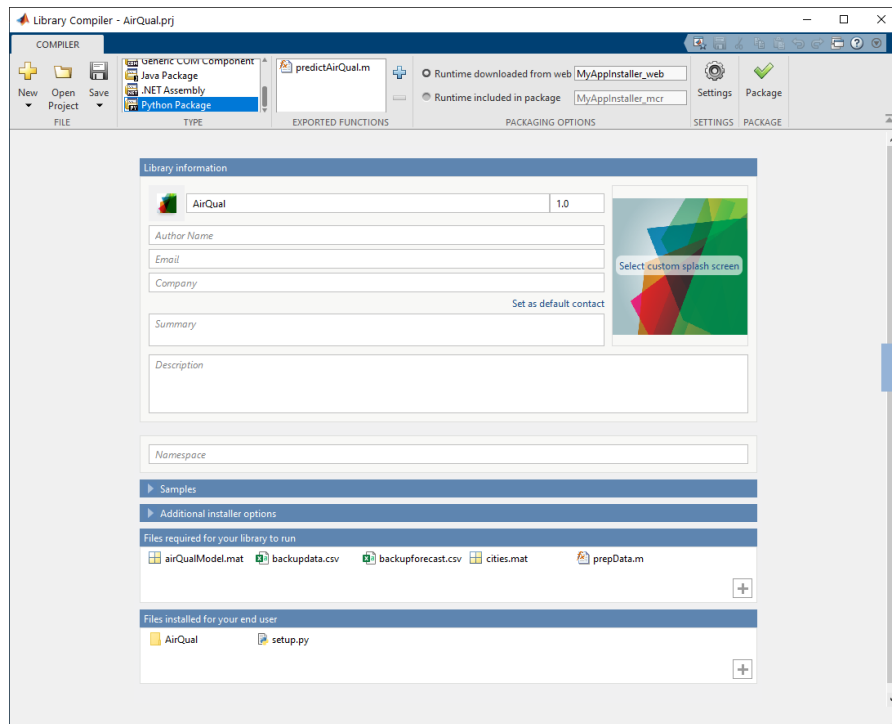
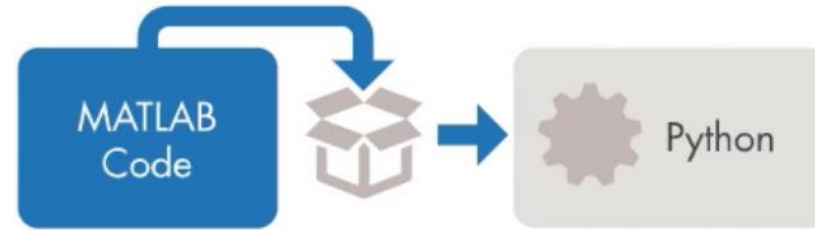
# from samples import get_forecast_url
def get_forecast_url():
    url = "https://samples.openweathermap.org"
    response = urllib.request.urlopen(url)
    html = response.read()
    json_data = json.loads(html)
    return json_data['products'][0]['forecast_5days']['samples'][0]

# get forecast
url = get_forecast_url()
response = urllib.request.urlopen(url)
html = response.read()
json_data = json.loads(html)
temp = [i['main']['temp'] for i in json_data['list']]

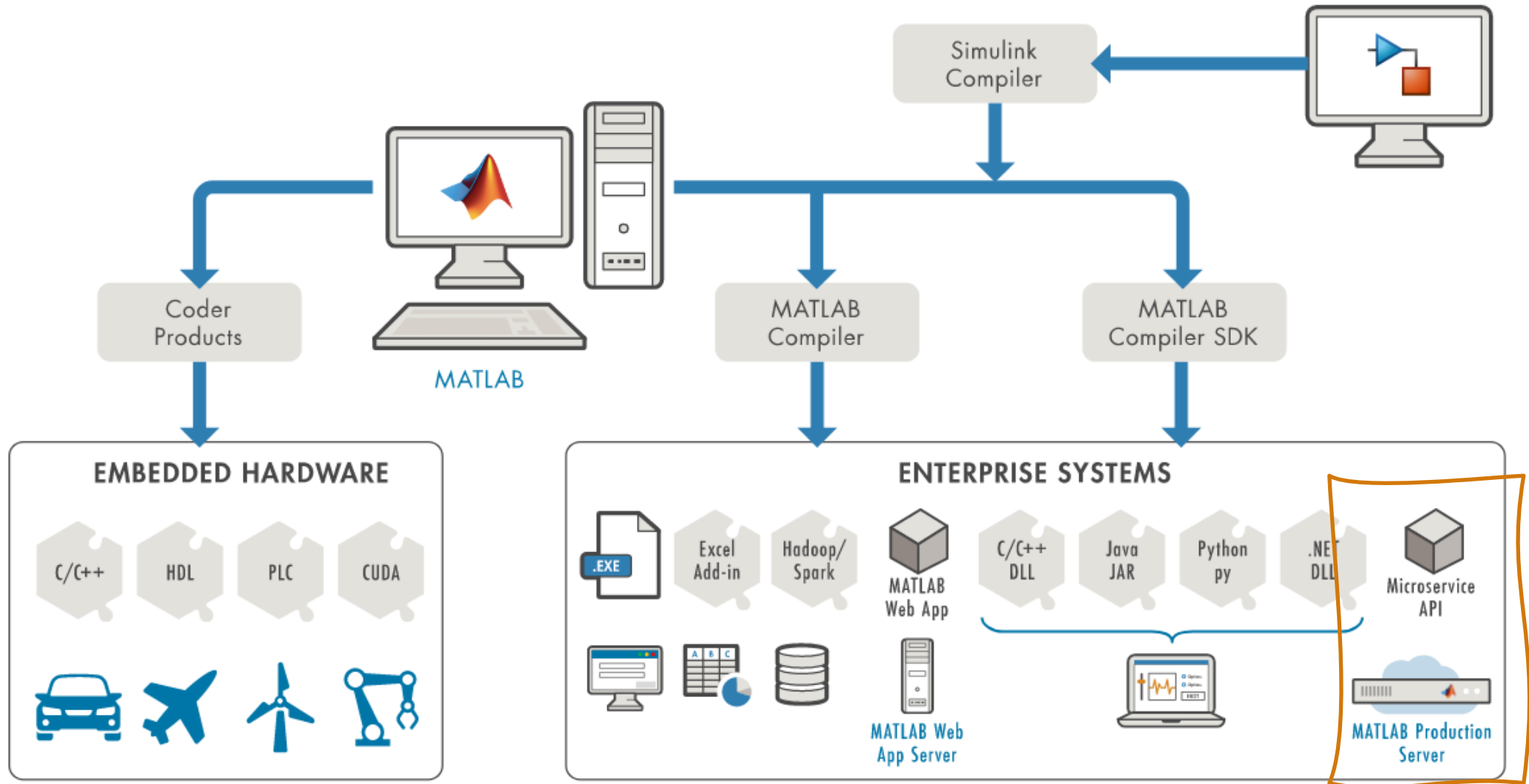
# get time
from datetime import datetime
time = [datetime.utcnow().strftime('%d-%m-%Y') for i in json_data['list']]

t = list(range(1,41))
mat_t = matlab.double(t)
mat_temp = matlab.double(temp)
```

# Generate Python package



# Deployment of MATLAB code as microservice or production server



# Share in a containerized world

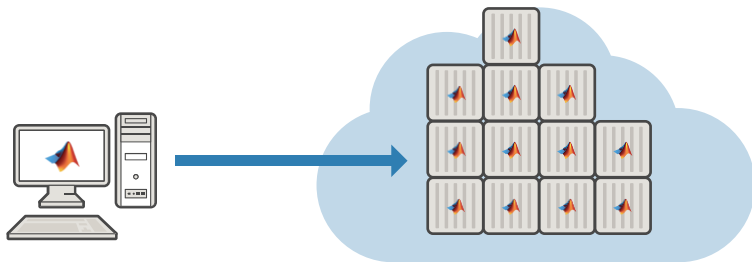


IT DevOps

We need to put our whole process (MATLAB + Python) in production

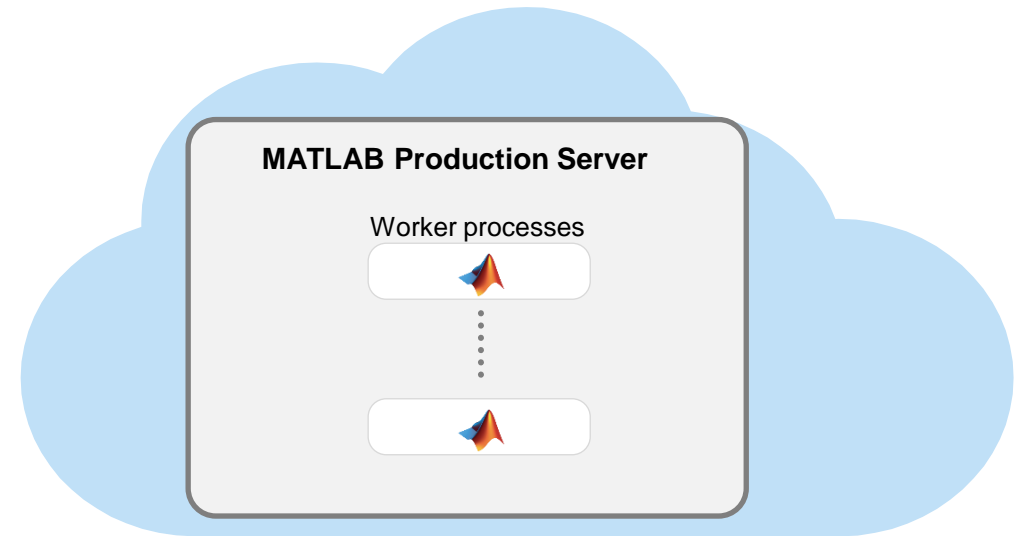
Create Microservice Docker Image:

[www.mathworks.com/help/compiler\\_sdk/mps\\_dev\\_test/create-a-microservice-docker-image.html](http://www.mathworks.com/help/compiler_sdk/mps_dev_test/create-a-microservice-docker-image.html)



# Orchestrate Web services

## MATLAB Production Server



Calling our function:

```
{"nargout":1,"rhs":["input"]}
```

Getting the result:

```
{"lhs":[{"mwdata":["output"],"mwsizе":[1,6],"mwtype":"char"}]}
```

# Call MATLAB from Python Web App

Air Quality Calculator

Archivio | C:/Users/gcarniel/MATLAB%20Drive/Python/PythonWeatherPrediction-master/4\_Call...

## Air Quality Conditions

Determine air quality conditions in your area.

Location:

The air quality is **Good**.  
The current temperature is **30.15 F**.

Copyright © 2018-2020 MathWorks, Inc.

# Deploy MATLAB App in the Web

**Air Quality Monitor**

Local Current Conditions

Location (US City):   **Air Quality: Good**

Temperature	Pressure	DewPoint	RelativeHumidity	WindDir	Wind Spd
31.1000	1033	19.9400	69	40	3.4400

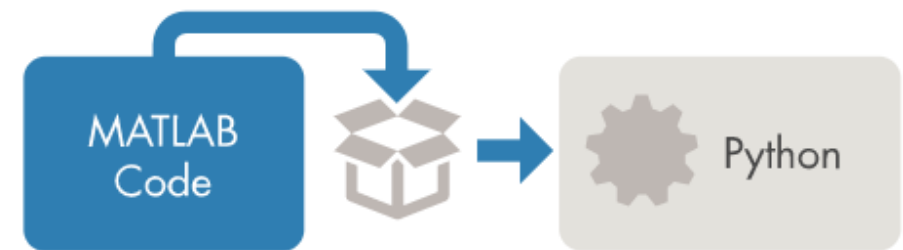
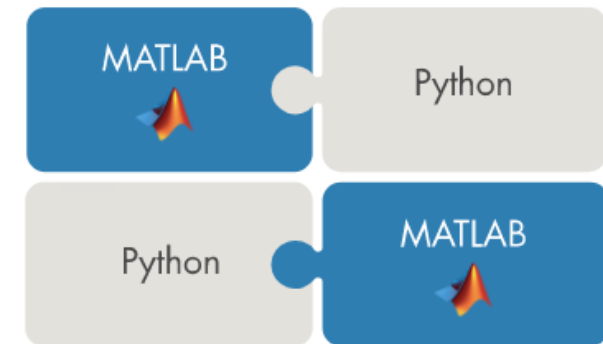
10 Days Forecast

The forecast chart displays five data series over a 10-day period. The y-axis for each series is on the left: T (0-40), P (1020), DP (20), RH (60), and WindDir (150-250). All series are plotted with green lines.

# Summary

## Using MATLAB with Python

- Access Data
  - From Python web-service
- Interoperability
  - Calling Python from MATLAB
  - Calling MATLAB from Python
- Deploy Apps & Algos
  - as Web app
  - as Web service



# Python libraries in MATLAB

## Documentation

# Python libraries in MATLAB (1)

## Directly call Python® functionality from MATLAB®

### Using Python Libraries

- **Access Python Modules from MATLAB - Getting Started**
  - How to create and use a Python object in MATLAB.
- **Configure Your System to Use Python**
  - How to verify you have installed a supported version of Python.
- **Call User-Defined Python Module**
  - Create a Python module used by examples in this documentation.
- **Understand Python Function Arguments**
  - Python method syntax which might be unfamiliar to MATLAB users.
- **Advanced Topics**
  - Code pattern differences you should be aware of.
- **Out-of-Process Execution of Python Functionality**
  - Execute Python scripts in processes that are separate from the MATLAB process.
- **Reload Out-of-Process Python Interpreter**
  - Reload out-of-process Python interpreter without restarting MATLAB.

```
tw = py.textwrap.TextWrapper(pyargs(...
    'initial_indent', '% ', ...
    'subsequent_indent', '% ', ...
    'width', int32(30));
wrapped = wrap(tw,T);
wrapped = cellfun(@char,...
    cell(wrapped),...
    'UniformOutput',false);
fprintf('%s\n', wrapped{:})
% Customize the output of the
% paragraph using keyword
% arguments.
>> |
```

 **Call Python Function in MATLAB to Wrap Paragraph Text**

Use Python language functions and modules within MATLAB. The example calls a text-formatting module from the Python standard

[Open Live Script](#)

# Python libraries in MATLAB (2)

## Directly call Python® functionality from MATLAB®

### Passing Data

- **MATLAB to Python Data Type Mapping**

- How MATLAB converts MATLAB data into compatible Python data types.

- **Access Elements in Python Container Types**

- A Python container is typically a sequence type (list or tuple) or a mapping type (dict).

- **Pass Python Function to Python map Function**

- This example shows how to display the length of each word in a list.


```
>> %Use Python array Types in MATLAB
F = py.array.array('d', 1:5)

F =

Python array:
 1  2  3  4  5

Use details function to view the properties of the Py
Use double function to convert to a MATLAB array.

>> |
```

 **Use Python Numeric Variables in MATLAB**

Use Python numeric variables with MATLAB.

[Open Live Script](#)

```
F = py.os.listdir(folder)

F =

Python list with no properties.

['5g', 'aero', 'aeroblks_produ

>> |
```

 **Use Python str Variables in MATLAB**

Use Python str variables with MATLAB.

[Open Live Script](#)


```
listVar = py.list(...
{'Name 1','Name 2','Name 3'

listVar =

Python list with no properties

['Name 1', 'Name 2', 'Name 3

>> |
```

 **Use Python list Variables in MATLAB**

Use Python list variables with MATLAB.

[Open Live Script](#)

```
tupleVar = py.tuple(...
{'Name','Subject',95})

tupleVar =

Python tuple with no properties

('Name', 'Subject', 95.0)

>> |
```

 **Use Python tuple Variables in MATLAB**

Use Python tuple variables with MATLAB.

[Open Live Script](#)

```
dict(pyargs('Name1',357,'Name2

dict with no properties.

Name2': 229.0, 'Name1': 357.0)

>> |
```

 **Use Python dict Variables in MATLAB**

Use Python dict variables with MATLAB.

[Open Live Script](#)

# Access Python Modules from MATLAB

## Getting Started

# Access Python Modules from MATLAB - Getting Started

## Learning Objectives

This tutorial explains how to:

- Check the Python version on your computer.
- Create a Python object and call a method on it.
- Display help for Python modules.
- Create specialized Python list, tuple, and dict (dictionary) types
- Call a method on a Python object with the same name as a MATLAB function.
- Call functionality from your own Python module.
- Find examples.

[https://www.mathworks.com/help/matlab/matlab\\_external/create-object-from-python-class.html](https://www.mathworks.com/help/matlab/matlab_external/create-object-from-python-class.html)

# Verify Python Configuration

To use Python in MATLAB, you must have a supported version of Python installed on your machine. To verify that you have a supported version, type:

```
pyenv
```

```
ans =
```

```
PythonEnvironment with properties:
```

```
Version: "3.6"
```

```
Executable: "C:\Users\aname\AppData\Local\Programs\Python\Python36\pythonw.exe"
```

```
Library: "C:\Users\aname\AppData\Local\Programs\Python\Python36\python36.dll"
```

```
Home: "C:\Users\aname\AppData\Local\Programs\Python\Python36"
```

```
Status: NotLoaded
```

```
ExecutionMode: InProcess
```

If the value of the Version property is empty, then you do not have a supported version available. For more information about installing Python, see [Configure Your System to Use Python](#).

# Access Python Standard Library Modules in MATLAB

- Create a Python list datatype:
- MATLAB recognizes Python objects and automatically converts the MATLAB cell array to the appropriate Python type.
- To convert the list variable to a MATLAB variable, call `cell` on the list and `char` on the elements of the list.

```
res = py.list({'Name1', 'Name2', 'Name3'})
```

```
res =
```

```
Python list with no properties.
```

```
['Name1', 'Name2', 'Name3']
```

```
res.append('Name4')
```

```
res
```

```
res =
```

```
Python list with no properties.
```

```
['Name1', 'Name2', 'Name3', 'Name4']
```

```
mylist = cellfun(@char, cell(res), 'UniformOutput', false)
```

```
mylist =
```

```
1x4 cell array
```

```
{'Name1'} {'Name2'} {'Name3'} {'Name4'}
```

## Display Python Documentation in MATLAB

You can display help text for Python functions in MATLAB. For example:

```
py.help('list.append')
```

```
Help on method_descriptor in list:
```

```
list.append = append(...)  
L.append(object) -> None -- append object to end
```

Tab completion when typing `py.` does not display available Python functionality. For more information, see [Help for Python Functions](#).

# Create List, Tuple, and Dictionary Types

This table shows the commands for creating list, tuple, and dict types. The commands on the left are run from the Python interpreter. The commands on the right are MATLAB commands.

Python list — []	MATLAB py.list
>>> ['Robert', 'Mary', 'Joseph']	>> py.list({'Robert','Mary','Joseph'})
>>> [[1,2],[3,4]]	>> py.list({py.list([1,2]),py.list([3,4])})
Python tuple — ()	MATLAB py.tuple
>>> ('Robert', 19, 'Biology')	>> py.tuple({'Robert',19,'Biology'})
Python dict — {}	MATLAB py.dict
>>> {'Robert': 357, 'Joe': 391, 'Mary': 229}	>> py.dict(pyargs(... 'Robert',357,'Mary',229,'Joe',391))
	For information about passing keyword arguments, see <a href="#">pyargs</a> .

# Call User-Defined Python Module

[https://www.mathworks.com/help/matlab/matlab\\_external/call-user-defined-custom-module.html](https://www.mathworks.com/help/matlab/matlab_external/call-user-defined-custom-module.html)

# Call User-Defined Python Module

This example shows how to call methods from the following Python<sup>®</sup> module:

```
# mymod.py
"""Python module demonstrates passing MATLAB types to Python functions"""
def search(words):
    """Return list of words containing 'son'"""
    newlist = [w for w in words if 'son' in w]
    return newlist

def theend(words):
    """Append 'The End' to list of words"""
    words.append('The End')
    return words
```

## Call User-Defined Python Module (2)

- If you create `mymod.py` in a Python editor, be sure that the module is on the [Python search path](#).
- From the MATLAB command prompt, add the current folder to the Python search path:

```
if count(py.sys.path,pwd()) == 0
    insert(py.sys.path,int32(0),pwd());
end
```

- To learn how to call the function, read the function signature for the search function in the `mymod.py` source file. The function takes one input argument, `words`.  
    `def search(words):`

## Call User-Defined Python Module (3)

- Create an input argument, a list of names, in MATLAB.

```
N = py.list({'Jones', 'Johnson', 'James'})
```

```
N =
```

```
Python list with no properties.
```

```
['Jones', 'Johnson', 'James']
```

- Call the search function. Type `py.` in front of the module name and function name.

```
names = py.mymod.search(N)
```

```
names =
```

```
Python list with no properties.
```

```
['Johnson']
```

# Reload Modified User-Defined Python Module

- **Create Python Module**

- Save the file mymod.py:

```
def myfunc():
    return 'version 1'
```

- **Modify Module**

- Modify the function:

```
    return 'version 2'
```

- **Unload Module**

```
>> clear classes
```

- **Import & Reload Modified Module**

```
>> mod = py.importlib.import_module('mymod');
```

```
>> py.importlib.reload(mod);
```

```
>> py.mymod.myfunc
```

```
ans =
```

```
Python str with no properties.
```

```
    version 1
```

```
>> py.mymod.myfunc
```

```
ans =
```

```
Python str with no properties.
```

```
    version 2
```

# Understand Python Function Arguments

[https://www.mathworks.com/help/matlab/matlab\\_external/python-function-arguments.html](https://www.mathworks.com/help/matlab/matlab_external/python-function-arguments.html)

# Positional Arguments

- A *positional* argument is passed by position. These arguments appear at the beginning of a function signature.

Python Signature	MATLAB Usage
<code>abs(X)</code> Argument X is required.	<code>&gt;&gt; py.abs(-99)</code>

- Some functions accept an arbitrary sequence of positional arguments, including no arguments. In Python, these arguments are defined by prepending the name with the \* character.

Python Signature	MATLAB Usage
<code>itertools.izip(*iterables)</code> The iterables argument is not required, in which case, the function returns a zero length iterator.	Aggregate elements from two lists. <code>&gt;&gt; py.itertools.izip(... py.list({1:10}),py.list({'a','b'}));</code>  Create zero length iterator. <code>&gt;&gt; py.itertools.izip;</code>
<code>print(*objects)</code>	<code>&gt;&gt; words = {'Hello','World!'};</code> <code>&gt;&gt; py.print(words{:})</code>

# Keyword Arguments

Use the MATLAB `pyargs` function to create keyword arguments for Python functions.

Python Signature	MATLAB Usage
<pre>print(*objects,sep=',',end='\n', file=sys.stdout) sep, end, and file are keyword arguments.</pre>	<p>Change the value of end.</p> <pre>&gt;&gt; py.print('string',pyargs('end','--'))</pre>

```
>> x1 = py.str('c:'); x2 = py.os.curdir; x3 = py.os.getenv('foo'); py.print(x1,x2,x3)
```

```
>> py.print(x1,x2,x3,pyargs('sep',sprintf('\n')))
```

```
>> py.print(x1,x2,x3,pyargs('end', sprintf(' THE END\n')), 'sep', py.str))
```

## Arbitrary Number of Keyword Arguments

Python Signature	MATLAB Usage
<pre>dict(**kwarg)</pre>	<pre>&gt;&gt; D = py.dict(pyargs('Joe',100,'Jack',101))</pre>

# Optional Arguments

Python Signature	MATLAB Usage
<code>random.randrange(start,stop[,step])</code> Argument step is optional.	<code>&gt;&gt; py.random.randrange(1,100)</code>

- Optional arguments can have default values. A default value is indicated by an equal sign = with the default value.

Python Signature	MATLAB Usage
<code>print(*objects,sep=",",end='\n', file=sys.stdout)</code> The default value for file is sys.stdout.	Print two values using default keyword values. <code>&gt;&gt; py.print(2,'2')</code>

pyrun

**R2021b**

# Copy & Paste from StackOverFlow & MATLAB Answers

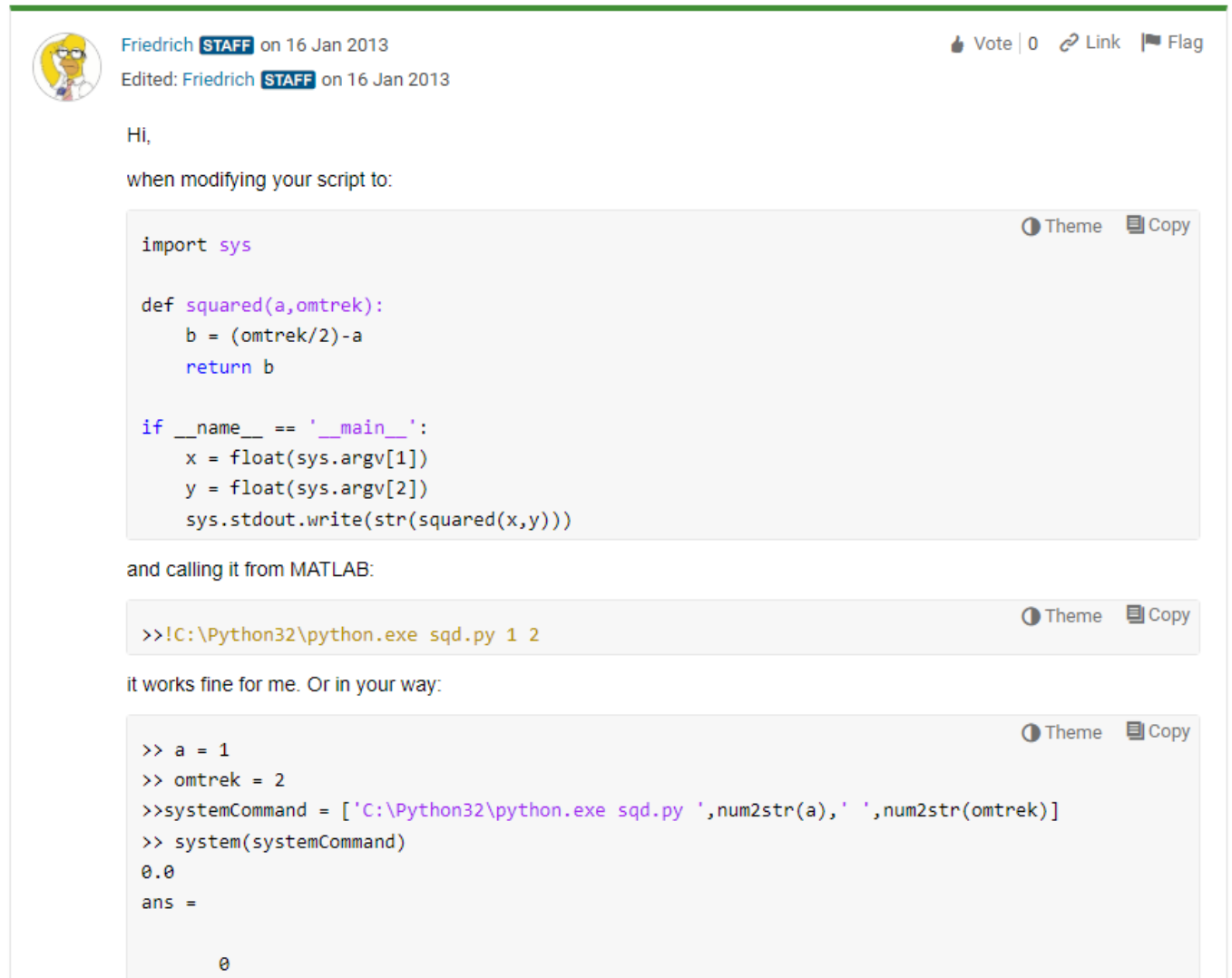
- Now with pyrunfile:

```
>> pyrunfile("sqd.py 42")  
1764.0
```

- Change .py file  
Run w/o reload:

```
>> pyrunfile("sqd.py 1 2")  
0.0
```

[Running a python script in matlab -  
MATLAB Central \(mathworks.com\)](https://www.mathworks.com/matlabcentral/answers/198888-running-a-python-script-in-matlab)



Friedrich **STAFF** on 16 Jan 2013  
Edited: Friedrich **STAFF** on 16 Jan 2013

Hi,  
when modifying your script to:

```
import sys  
  
def squared(a,omtrek):  
    b = (omtrek/2)-a  
    return b  
  
if __name__ == '__main__':  
    x = float(sys.argv[1])  
    y = float(sys.argv[2])  
    sys.stdout.write(str(squared(x,y)))
```

and calling it from MATLAB:

```
>>!C:\Python32\python.exe sqd.py 1 2
```

it works fine for me. Or in your way:

```
>> a = 1  
>> omtrek = 2  
>>systemCommand = ['C:\Python32\python.exe sqd.py ',num2str(a),' ',num2str(omtrek)]  
>> system(systemCommand)  
0.0  
ans =  
  
0
```

## “I know how to do this in Python”

```
>> pyrun("l = [1,2,3]")
```

```
>> pyrun("print(l)")  
[1, 2, 3]
```

```
>> L2 = pyrun("l2 = [k^2 for k in l]", "l2")
```

```
L2 =
```

Python [list](#) with no properties.

```
[3, 0, 1]
```

## Variables created using the `pyrun` function are persistent

```
>> pyrun("scope = dir()", "scope")
```

```
ans =
```

Python [list](#) with no properties.

```
['__builtins__', '__name__', '1', '12']
```

## Pass Arguments to Python Operator

- Executes  $a = b*c$  in the Python interpreter with the specified input values:

```
>> pyrun("a = b*c", b = 5, c = 10)
```

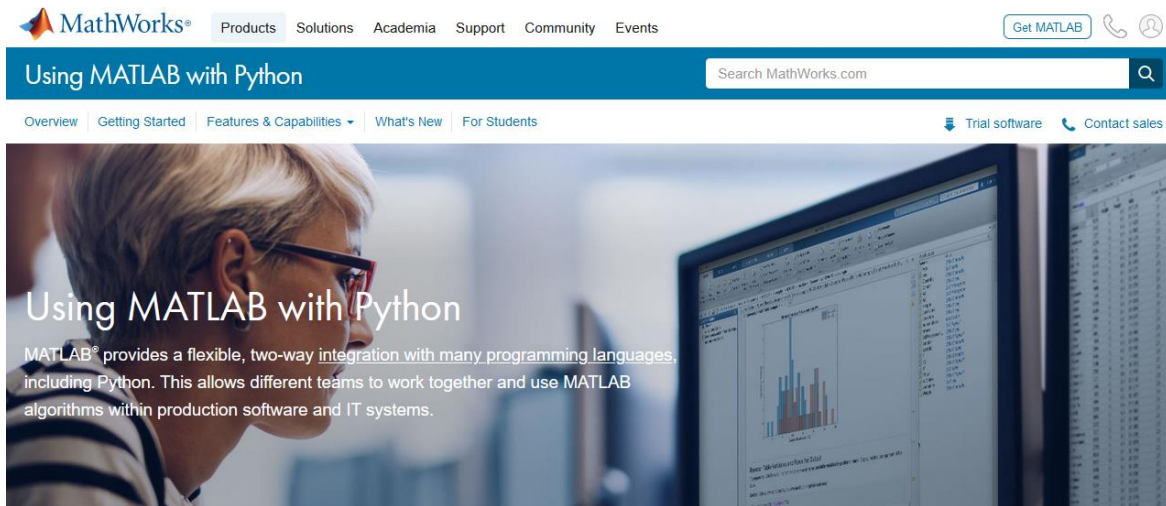
- a, b and c exist only in the Python namespace.  
However, these variables are available for further calls to pyrun

```
>> md = pyrun("d = a+c", "d")
```

```
md = 60
```

# Additional resources

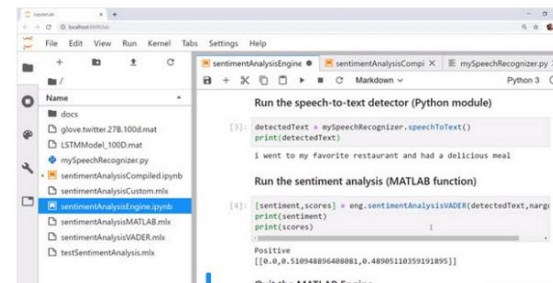
# Resources



## Calling MATLAB from Python

The MATLAB Engine API for Python allows you to call MATLAB as a computational engine from Python.

The API lets you execute MATLAB commands from within your Python environment without starting a desktop session of MATLAB. Learn more about the [MATLAB Engine API for Python](https://www.mathworks.com/products/matlab/matlab-and-python.html).



<https://www.mathworks.com/products/matlab/matlab-and-python.html>

- [Cheatsheet](#)
- [Example on GitHub](#)
- [Blog post](#)
- Videos
  - [Calling Python from MATLAB](#)
  - [Calling MATLAB from Python](#)
  - [Using MATLAB with Python + Q&A \(YouTube live stream recording\)](#)
- Documentation
  - [Calling Python from MATLAB](#)
  - Calling MATLAB from Python via:
    - [MATLAB Engine API](#)
    - [MATLAB Compiler SDK](#)
    - [MATLAB Production Server](#)
  - Data management:
    - [Data type conversions](#)
    - [Working with Parquet files](#)
    - [MATLAB library for Apache Arrow](#)
  - [Deep Learning \(TensorFlow, PyTorch, etc\)](#)

# Cheat Sheet MATLAB and Python together



## Using MATLAB® and Python® Together

The [≥](#) icon provides links to relevant sections of the MATLAB documentation to learn more.

### Call Python in MATLAB

Access settings and status of Python interpreter:

```
>> pe = pyenv
```

Specify version to use:

```
>> pe = pyenv("Version",3.7)
```

Call Python modules and functions:

```
py.module_name.function_name
```

```
>> py.math.sqrt(42)
```

**Pass keyword arguments**  
Use pyargs to pass keyword arguments

```
>>> foo(5,bar=42)
>> py.foo(5,pyargs('bar',42))
```

**Reload modules**  
Reload the module after making updates:

```
>> py.importlib.reload(module)
```

### Call MATLAB in Python

**Install MATLAB Engine API for Python** [≥](#)  
Run `setup.py` from an OS command window

```
$ cd [matlabroot]/extern/engines/python
$ python setup.py install
```

**Call MATLAB functions**  
Import the module and start the engine

```
>>> import matlab.engine
>>> eng =
    matlab.engine.start_matlab()
```

Call functions through the engine

```
>>> x = eng.sqrt(42.0)
```

Capture multiple outputs

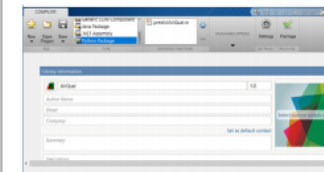
```
>>> x = eng.gcd(42.0,8.0,nargout=3)
```

Stop the engine

```
>>> eng.exit()
```

### Create Python Package

**Package MATLAB functions** [≥](#)  
Use the Library Compiler App to create a Python package for MATLAB functions



**Invoke MATLAB functions from the Python package**

```
>>> import PackageName
>>> pkg =
    PackageName.initialize()
>>> result = pkg.foo()
```

**Close package**

```
>>> pkg.terminate()
```

# Cheat Sheet MATLAB for Python Users



## MATLAB® for Python® Users

The MATLAB language is designed primarily for math-intensive scientific computing. MATLAB combines a desktop environment tuned for iterative analysis with a programming language that expresses matrix and array mathematics directly. Understanding the philosophy and API design can help while learning MATLAB.

» General Behavior			
Python Syntax	MATLAB Syntax	Purpose	MATLAB Examples
#	%	Comment	%hello
print	Do not terminate with;	Print output	x
/	...	Continue to next line	x = 1+...2;
os	!	Operating system command	! echo hi
+ - * /	+ - * /	Mathematical operators	x = 1+2
**	^	Exponent	x = y^2
* / **	.* ./ .^	Element-wise operators	x = [1 2].* [3 4]
not, and, or	~ &	NOT, AND, OR logical operators	if x<2 & x>2
del	clear	Clear variable from memory	clear x y
clear	clc	Clear command window	clc

» Referencing		
MATLAB Syntax	Purpose	Example
()	Index (copy-on-write)	x(1,1)
[]	Create array	x = [1 2 3]
	Join arrays	z = [x ; y]
{}	Create cell arrays	x = {42; "hello world"}
	Extract contents from a container	x{1,1}
.	Access class property or method	obj.Data
	Reference table or struct field	t.FieldName

- Beginning element has an index of 1.
- Indexing is left and right inclusive.
- Indexing options include N-D indexing (row,col), linear indexing (element number), and logical indexing (conditional statement).

# MATLAB Answers – tag:"python"

The screenshot shows the MATLAB Answers website interface. At the top, there is a navigation bar with the MathWorks logo and links for Products, Solutions, Academia, Support, Community, and Events. A search bar contains the text "tag:'python'" and a dropdown menu is set to "Answers". Below the navigation bar, there are links for MATLAB Central, Home, Ask, Answer, Browse, MATLAB FAQs, Contributors, More, and Help. A "Trial software" button is also visible.

The main content area displays "Results for tag:'python' (614 results)". A "Sort by" dropdown is set to "Date updated (Newest–Oldest)". There are two "0" icons for answers and a "Subscribe to this View" button. The results list includes:

- Matlab 2019a pyenv.Library empty when using Python 2.7**  
 Asked by Simon Geoffroy-Gagnon on 9 Oct 2020 at 22:29  
 Tags python, pyenv, load python
- Error 127 Details: Unable to load bundle binary C:\Program Files\MATLAB\R2020b\bin\win64\builtins\matlab\_toolbox\_general\_builtins\mwwkspintrospect\_builtinimpl.dll**  
 Asked by Mehul Garg on 1 Oct 2020 at 7:18  
 Latest activity Commented on by Mehul Garg on 9 Oct 2020 at 4:26  
 Tags bundle 117, error 127, flask, python, mcr  
 Products MATLAB Compiler, MATLAB Compiler SDK
- problem with python numpy**  
 Asked by Johann Thurn on 31 Oct 2018

On the left side, there is a "FILTER BY" section with three categories: Status, Source, and Product. The Status filter shows 370 Answered, 244 Unanswered, and 161 Answer Accepted. The Source filter shows 603 from Community and 11 from MathWorks Support. The Product filter shows 215 for MATLAB, 10 for Simulink, 1 for Audio Toolbox, 1 for Communications Toolbox, 1 for Database Toolbox, 5 for Deep Learning Toolbox, and 4 for Embedded Coder.

<https://www.mathworks.com/matlabcentral/answers/?term=tag%3A%22python%22>