https://www.halvorsen.blog

Basic Python Programming

Hans-Petter Halvorsen

Free Textbook with lots of Practical Examples

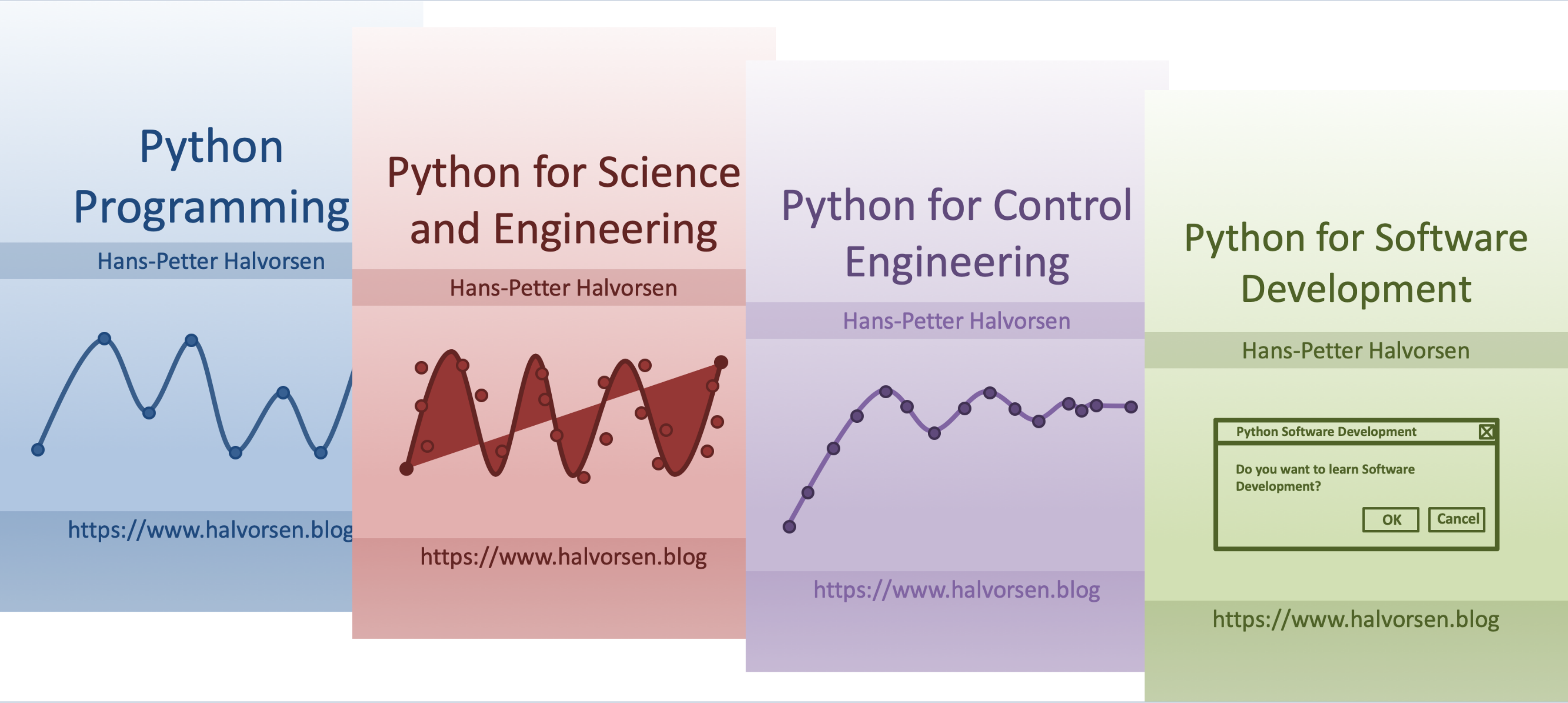
Python Programming

Hans-Petter Halvorsen

https://www.halvorsen.blog

https://www.halvorsen.blog/documents/programming/python/

Additional Python Resources



Python Programming

Hans-Petter Halvorsen

https://www.halvorsen.blog

Python for Science and Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog

Python for Control Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog

Python for Software Development

Hans-Petter Halvorsen

Python Software Development

Do you want to learn Software Development?

OK Cancel

https://www.halvorsen.blog

https://www.halvorsen.blog/documents/programming/python/

Contents

Basic Python Program

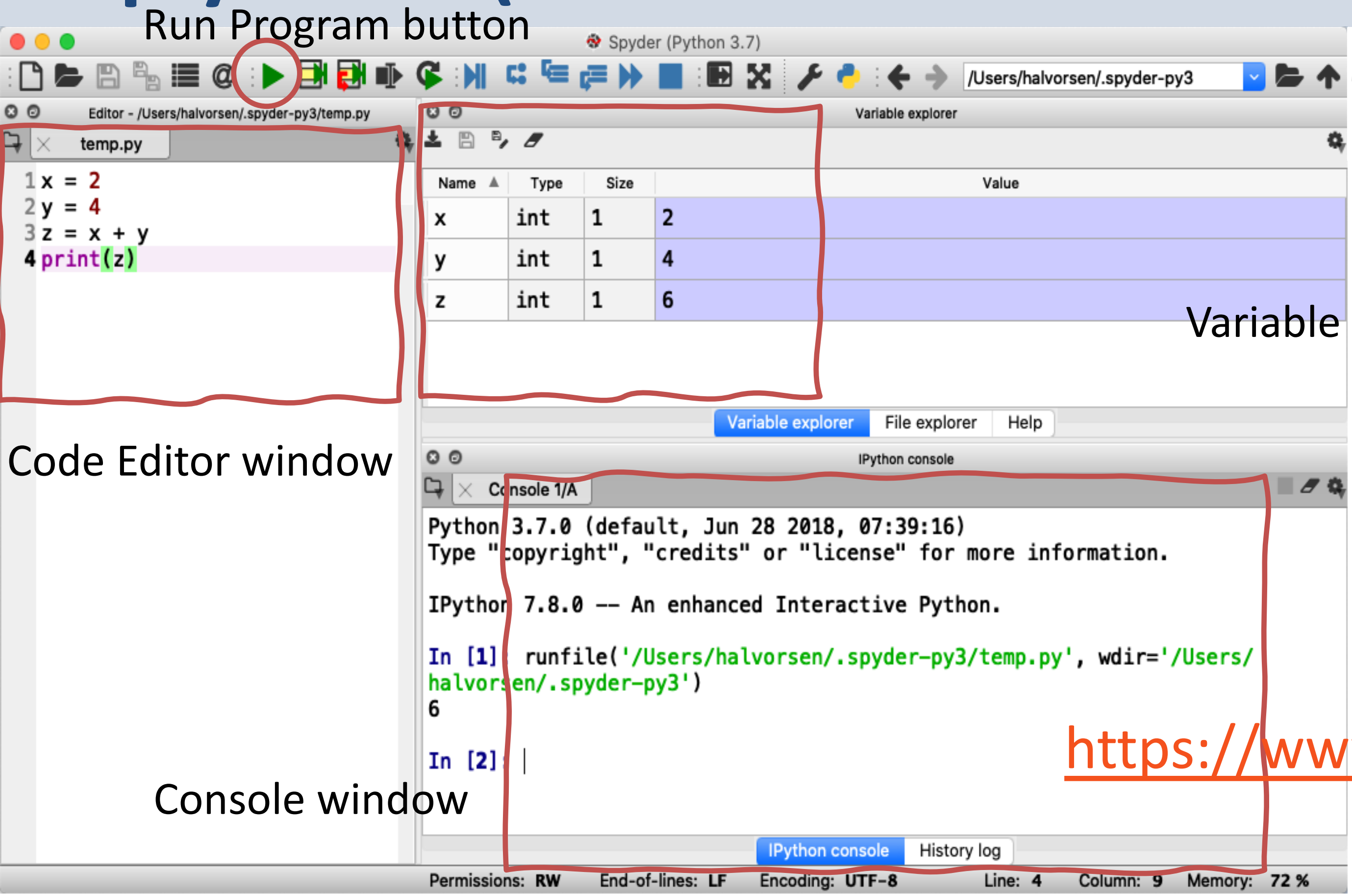
- We use the basic IDLE editor or another Python Editor like Spyder (included with Anaconda distribution) or Visual Studio Code, etc.

```
print("Hello World!")
```

Python Editors

- Python IDLE
- **Spyder** (Anaconda distribution)
- PyCharm
- **Visual Studio Code**
- Visual Studio
- Jupyter Notebook
- …

Spyder (Anaconda distribution)



```
x = 2
y = 4
z = x + y
print(z)
```

Name	Type	Size	Value
x	int	1	2
y	int	1	4
z	int	1	6

```
Python 3.7.0 (default, Jun 28 2018, 07:39:16)
Type "copyright", "credits" or "license" for more information.

IPython 7.8.0 -- An enhanced Interactive Python.

In [1]: runfile('/Users/halvorsen/.spyder-py3/temp.py', wdir='/Users/
halvorsen/.spyder-py3')
6

In [2]:
```

Permissions: RW   End-of-lines: LF   Encoding: UTF-8   Line: 4   Column: 9   Memory: 72 %

SPYDER
The Scientific Python Development Environment

https://www.anaconda.com

Variables in Python

- Variables are defined with the assignment operator, “=”.
- Python is dynamically typed, meaning that variables can be assigned without declaring their type, and that their type can change.

```
> x = 3
```

Variables in Python

Creating variables:

```
> x = 3
> x
3
```

We can implement the formula $y(x) = ax + b$ like this:

$$y(x) = 2x + 4$$

We can use variables in a calculation like this:

```
> x = 3
> y = 3*x
> print(y)
```

```
> a = 2
> b = 4

> x = 3
> y = a*x + b
> print(y)
```

A variable can have a short name (like x and y) or a more descriptive name (sum, amount, etc).
You don need to define the variables before you use them (like you need to to in, e.g., C/C++/C).

Variables in Python

Here are some basic rules for Python variables:

- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters (A-z, 0-9) and underscores
- Variable names are **case-sensitive**, e.g., amount, Amount and AMOUNT are three different variables.

Calculations in Python

We can use variables in a calculation like this:

$$y(x) = 2x + 4$$

```
> a = 2
> b = 4
```

$y(3)$ = ?

```
> x = 3
> y = a*x + b
> print(y)
```

$y(5)$ = ?

```
> x = 5
> y = a*x + b
> print(y)
```

Numbers

There are three numeric types in Python:

- int
- float
- complex

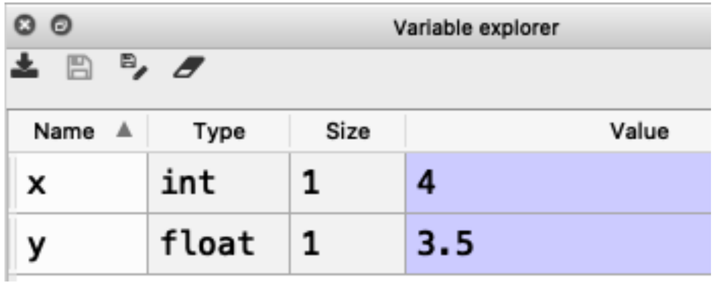
The symbol # is used for commenting the code

```
> x = 1 #int
> y = 2.8 #float
> z = 3 + 2j #complex number
```

Check the Data Type:

```
> type(x)
> type(y)
> type(z)
```

If you use the Spyder Editor, you can see the data types that a variable has using the **Variable Explorer**



Name	Type	Size	Value
x	int	1	4
y	float	1	3.5

Variables of numeric types are automatically created when you assign a value to them, so in normal coding you don't need to bother.

Strings

- Strings in Python are surrounded by either single quotation marks, or double quotation marks.
- 'Hello' is the same as "Hello".
- Strings can be output to screen using the print function. For example: print("Hello").

```
> text = ”Hello“
> print(text)
```

Manipulating Strings

There are many built-in functions form manipulating strings in Python.
The Example shows only a few of them:

```
a = "Hello World!"
print(a)
print(a[1])
print(a[2:5])
print(len(a))
print(a.lower())
print(a.upper())
print(a.replace("H", "J"))
print(a.split(" "))
```

Strings in Python are arrays of bytes, and we can use index to get a specific character within the string as shown in the example code.

String Concatenation

We can merge strings like this:

```
> a = "Hello"
> b = "World"
> c = a + b
> print(c)
```

String Input

The following example asks for the user's name, then, by using the input() method, the program prints the name to the screen:

```
> print("Enter your name:")
> x = input()
> print("Hello, " + x)
```

Built-in Functions

Python consists of lots of built-in functions.

- Some examples are the print function that we already have used (perhaps without noticing it is a built-in function) or the functions for manipulating strings.
- Python also consists of different Modules, Libraries or Packages. These Modules, Libraries or Packages consists of lots of predefined functions for different topics or areas, such as mathematics, plotting, handling database systems, etc.
- In another video we also will learn to create our own functions from scratch.

Python Standard Library

- Python allows you to split your program into modules that can be reused in other Python programs. It comes with a large collection of standard modules that you can use as the basis of your programs.
- The Python Standard Library consists of different modules for handling file I/O, basic mathematics, etc.
- You don't need to install the modules in the Python Standard Library separately, but you need to important them when you want to use some of these modules or some of the functions within these modules.

math Module

Python Standard Library

The math module has all the basic math functions you need, such as:

- Trigonometric functions: sin(x), cos(x), etc.
- Logarithmic functions: log(), log10(), etc.
- Statistics: mean(), stdev(), etc.
- Constants like pi, e, inf, nan, etc.

math Module

If we need only the sin() function, we can do like this:

```
from math import sin

x = 3.14
y = sin(x)
```

If we need many functions, we can do like this:

```
from math import *

x = pi
y = sin(x)
print(y)

y = cos(x)
print(y)

…
```

If we need a few functions, we can do like this:

```
from math import sin, cos

x = 3.14
y = sin(x)
print(y)

y = cos(x)
print(y)
```

We can also do like this:

```
import math
x = 3.14
y = math.sin(x)
print(y)
```

https://www.halvorsen.blog

Using Python Libraries, Packages and Modules

Hans-Petter Halvorsen

Python Packages/Libraries

- Rather than having all its functionality built into its core, Python was designed to be highly extensible.
- This approach has advantages and disadvantages.
- A disadvantage is that you need to install these packages separately and then later import these modules in your code.
- Some important packages are:
  - **NumPy** - NumPy is the fundamental package for scientific computing with Python
  - **Matplotlib** – With this library you can easily make plots in Python

Installing Packages/Libraries

- If you have installed Python using the **Anaconda distribution**, all the most used Python Packages/Libraries are included (NumPy, Matplotlib, +++)
- Else, you typically use **PIP** to install Python packages

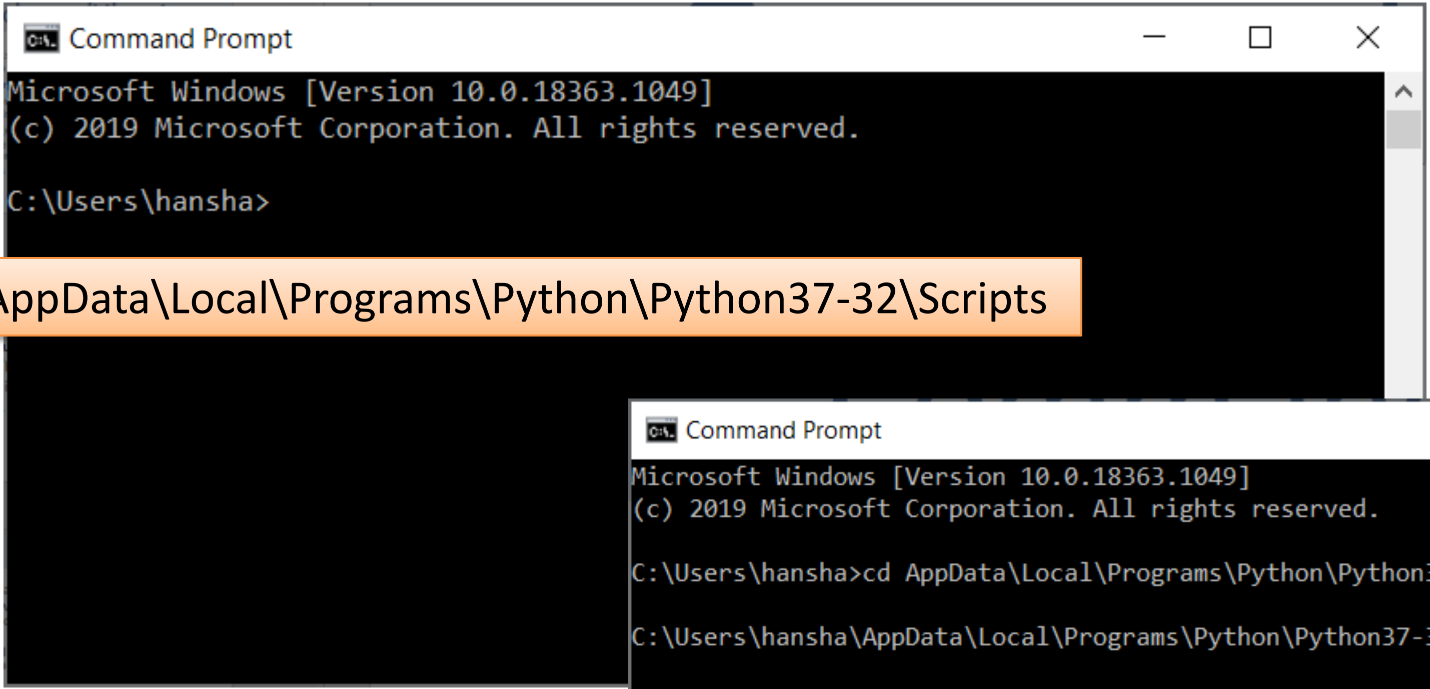
PIP

- PIP is a Package Manager for Python packages/modules
- With PIP you can download and install Python packages/modules from the Python Package Index (PyPI)
- What is a Package? A package contains all the files you need for a module. Modules are Python code libraries you can include in your project.
- The Python Package Index (PyPI) is a repository of Python packages
- Typically you just enter “**pip install <packagename>**“
- PIP uses the Python Package Index, PyPI as a source, which stores almost 200.000 Python projects

https://pypi.org

Command Prompt (cmd)

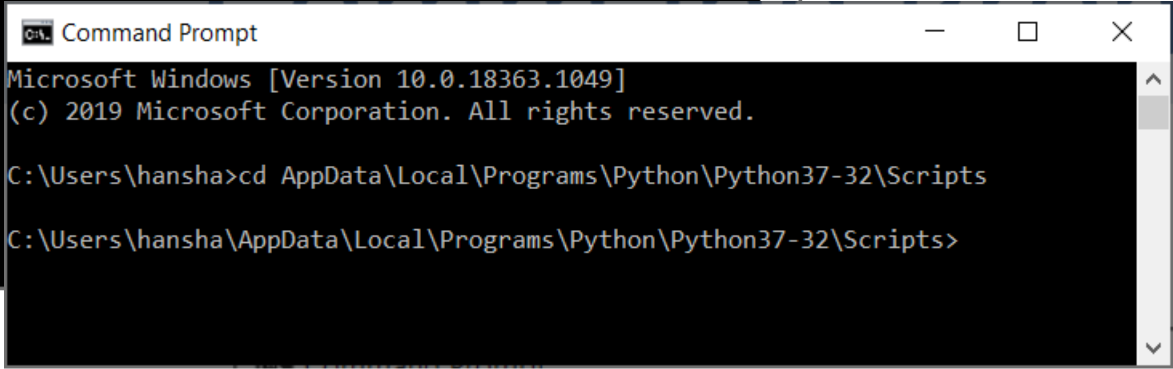
Use PIP from the Command Prompt in Windows:



Command Prompt

```
Microsoft Windows [Version 10.0.18363.1049]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hansha>
```

cd AppData\Local\Programs\Python\Python37-32\Scripts

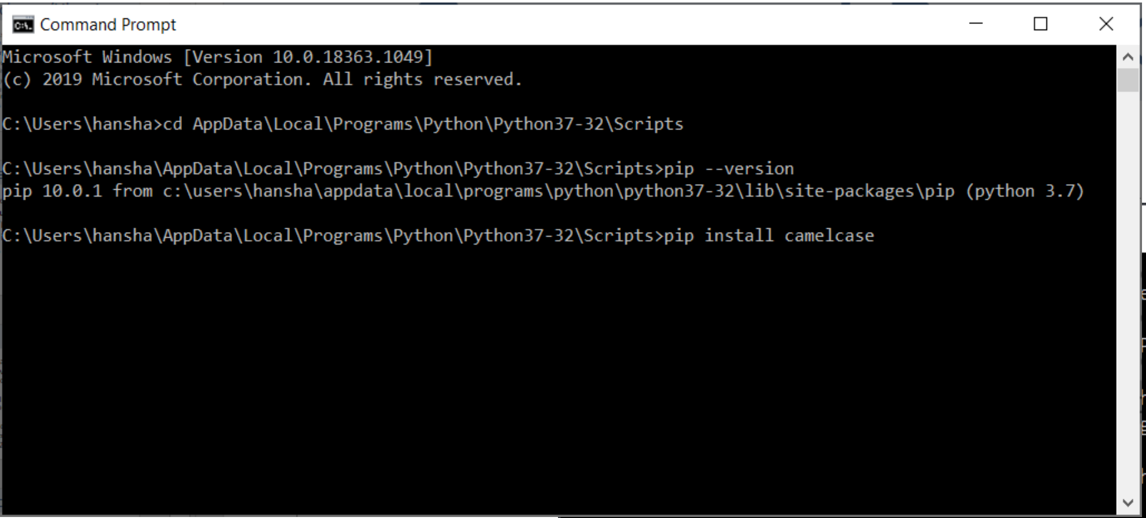


Command Prompt

```
Microsoft Windows [Version 10.0.18363.1049]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hansha>cd AppData\Local\Programs\Python\Python37-32\Scripts

C:\Users\hansha\AppData\Local\Programs\Python\Python37-32\Scripts>
```

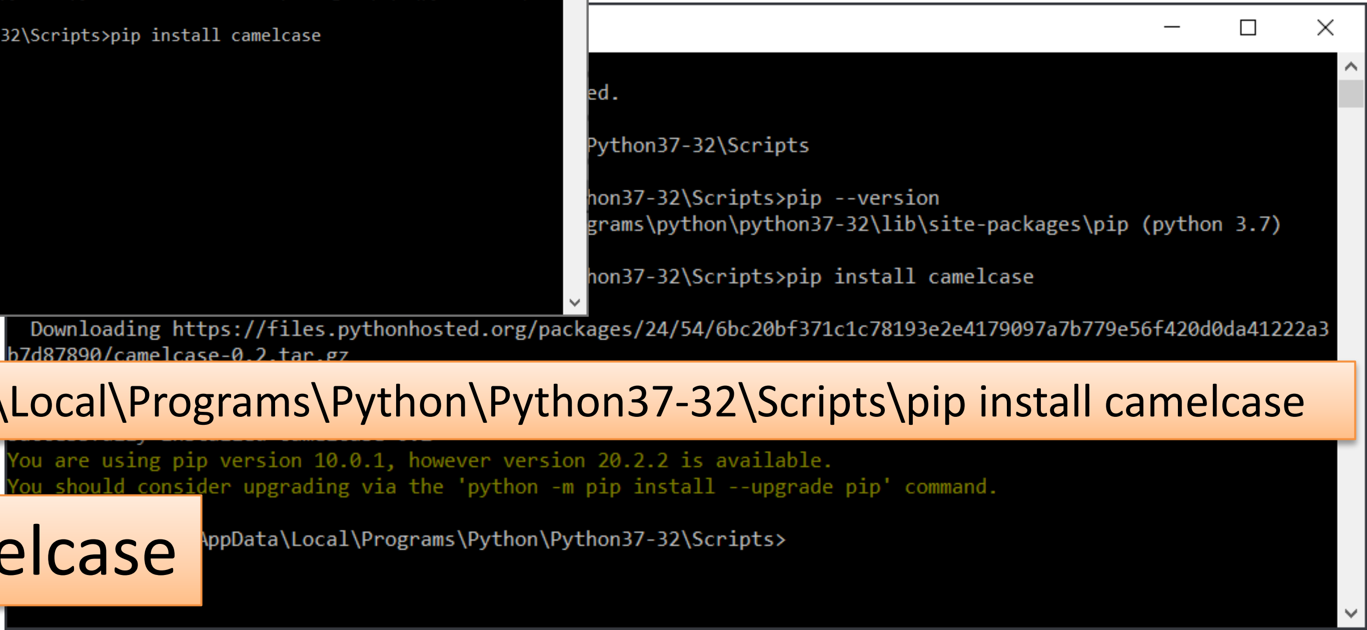
Command Prompt - PIP

Example: Install Python package “camelCase”:



```
Microsoft Windows [Version 10.0.18363.1049]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hansha>cd AppData\Local\Programs\Python\Python37-32\Scripts

C:\Users\hansha\AppData\Local\Programs\Python\Python37-32\Scripts>pip --version
pip 10.0.1 from c:\users\hansha\appdata\local\programs\python\python37-32\lib\site-packages\pip (python 3.7)

C:\Users\hansha\AppData\Local\Programs\Python\Python37-32\Scripts>pip install camelcase
```

pip install camelcase



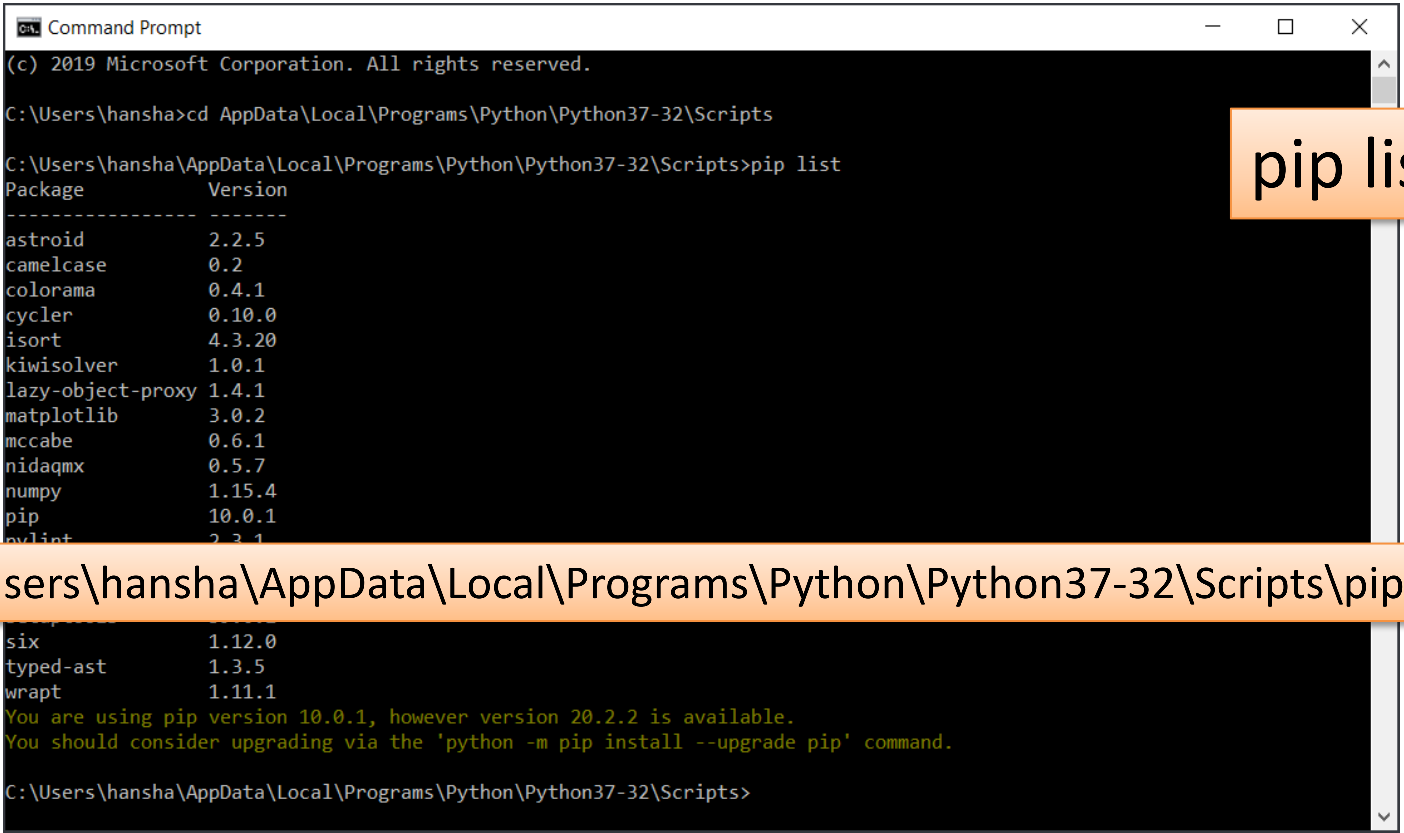
```
ed.

Python37-32\Scripts

hon37-32\Scripts>pip --version
grams\python\python37-32\lib\site-packages\pip (python 3.7)

hon37-32\Scripts>pip install camelcase

  Downloading https://files.pythonhosted.org/packages/24/54/6bc20bf371c1c78193e2e4179097a7b779e56f420d0da41222a3
b7d87890/camelcase-0.2.tar.gz

You are using pip version 10.0.1, however version 20.2.2 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

AppData\Local\Programs\Python\Python37-32\Scripts>
```

C:\Users\hansha\AppData\Local\Programs\Python\Python37-32\Scripts\pip install camelcase

pip uninstall camelcase

Command Prompt – PIP

Get overview of installed Python Packages:



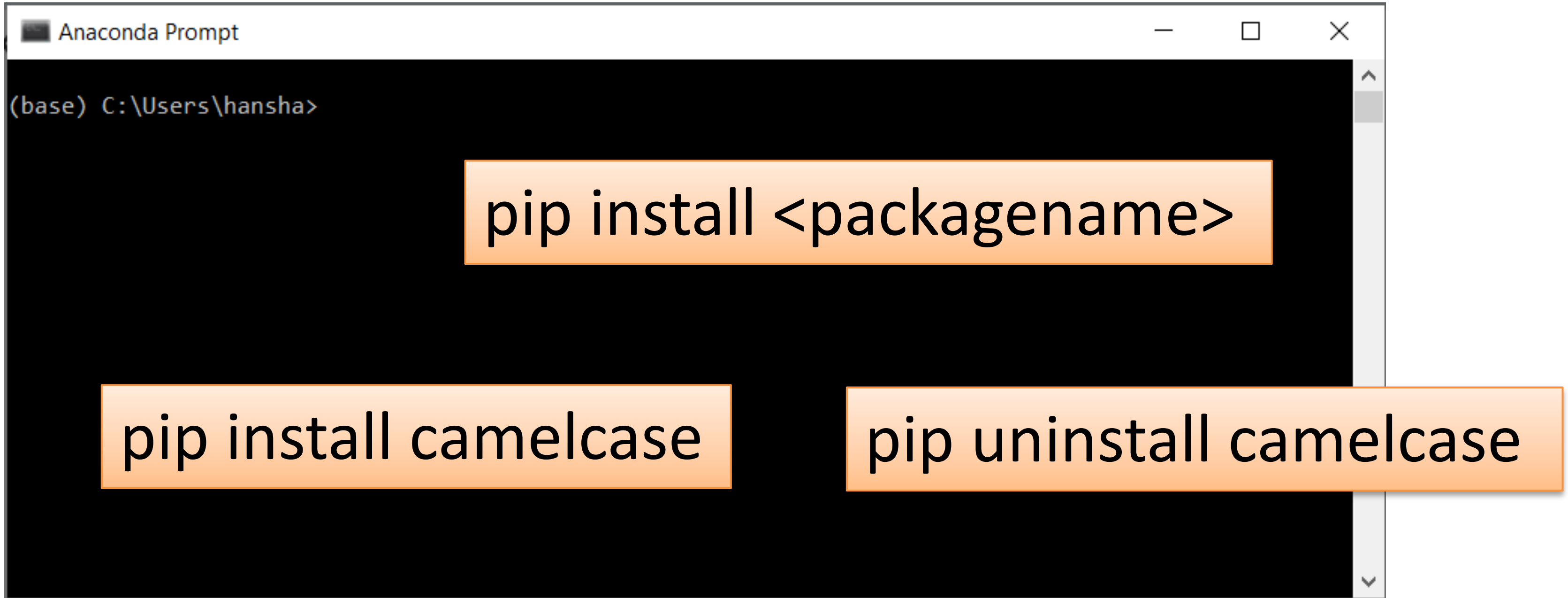
```
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hansha>cd AppData\Local\Programs\Python\Python37-32\Scripts

C:\Users\hansha\AppData\Local\Programs\Python\Python37-32\Scripts>pip list
Package           Version
----------------- -------
astroid           2.2.5
camelcase         0.2
colorama          0.4.1
cycler            0.10.0
isort             4.3.20
kiwisolver        1.0.1
lazy-object-proxy 1.4.1
matplotlib        3.0.2
mccabe            0.6.1
nidaqmx           0.5.7
numpy             1.15.4
pip               10.0.1
pylint            2.3.1
six               1.12.0
typed-ast         1.3.5
wrapt             1.11.1
You are using pip version 10.0.1, however version 20.2.2 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\Users\hansha\AppData\Local\Programs\Python\Python37-32\Scripts>
```

pip list

C:\Users\hansha\AppData\Local\Programs\Python\Python37-32\Scripts\pip list

Anaconda Prompt

If you have installed Python with Anaconda Distribution, the most popular Python Packages/Libraries have already been installed for you, and you don’t need to do anything. But if you need a package that is not installed, you can use **Anaconda Prompt** (just search for it using the Search field in Windows)



```
(base) C:\Users\hansha>
```

```
pip install <packagename>
```

```
pip install camelcase
```

```
pip uninstall camelcase
```

Using libraries

You need to use the **import** keyword on top of you Python script:

```
import packagename as alias

.. Your Python code
```

Example: Using numpy:

```
import numpy as np

x = 3

y = np.sin(x)

print(y)
```

NumPy

```
pip install numpy
```

- The only prerequisite for NumPy is Python itself.
- If you don’t have Python yet and want the simplest way to get started, you can use the **Anaconda Distribution** - it includes Python, NumPy, and other commonly used packages for scientific computing and data science.
- Or use “pip install numpy“

https://numpy.org

NumPy

Basic NumPy Example:

```
import numpy as np

x = 3

y = np.sin(x)

print(y)
```

In this example we use both the math module in the Python Standard Library and the NumPy library:

```
import math as mt
import numpy as np

x = 3

y = mt.sin(x)
print(y)

y = np.sin(x)
print(y)
```

As you see, NumPy also have also similar functions (e.g., sim(), cos(), etc.) as those who is part of the math library, but they are more powerful

Matplotlib

```
import matplotlib.pyplot as plt
```

- Typically you need to create some plots or charts. In order to make plots or charts in Python you will need an external library. The most used library is Matplotlib
- Matplotlib is a Python 2D plotting library
- Here you find an overview of the Matplotlib library: https://matplotlib.org
- Matplotlib is included with Anaconda Distribution
- If you are familiar with MATLAB and basic plotting in MATLAB, using the Matplotlib is very similar.
- The main difference from MATLAB is that you need to import the library, either the whole library or one or more functions.

Matplotlib

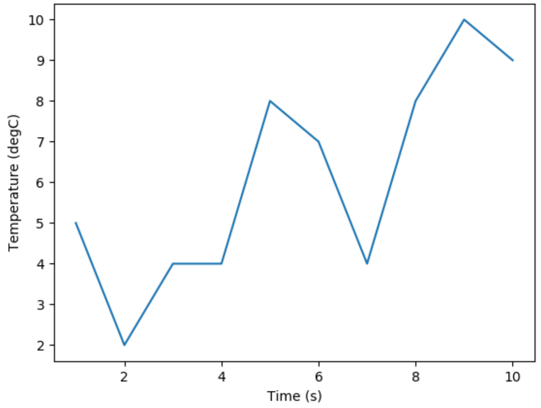
Here are some plotting functions that you will use a lot:

- plot()
- title()
- xlabel()
- ylabel()
- axis()
- grid()
- subplot()
- legend()
- show()

Matplotlib

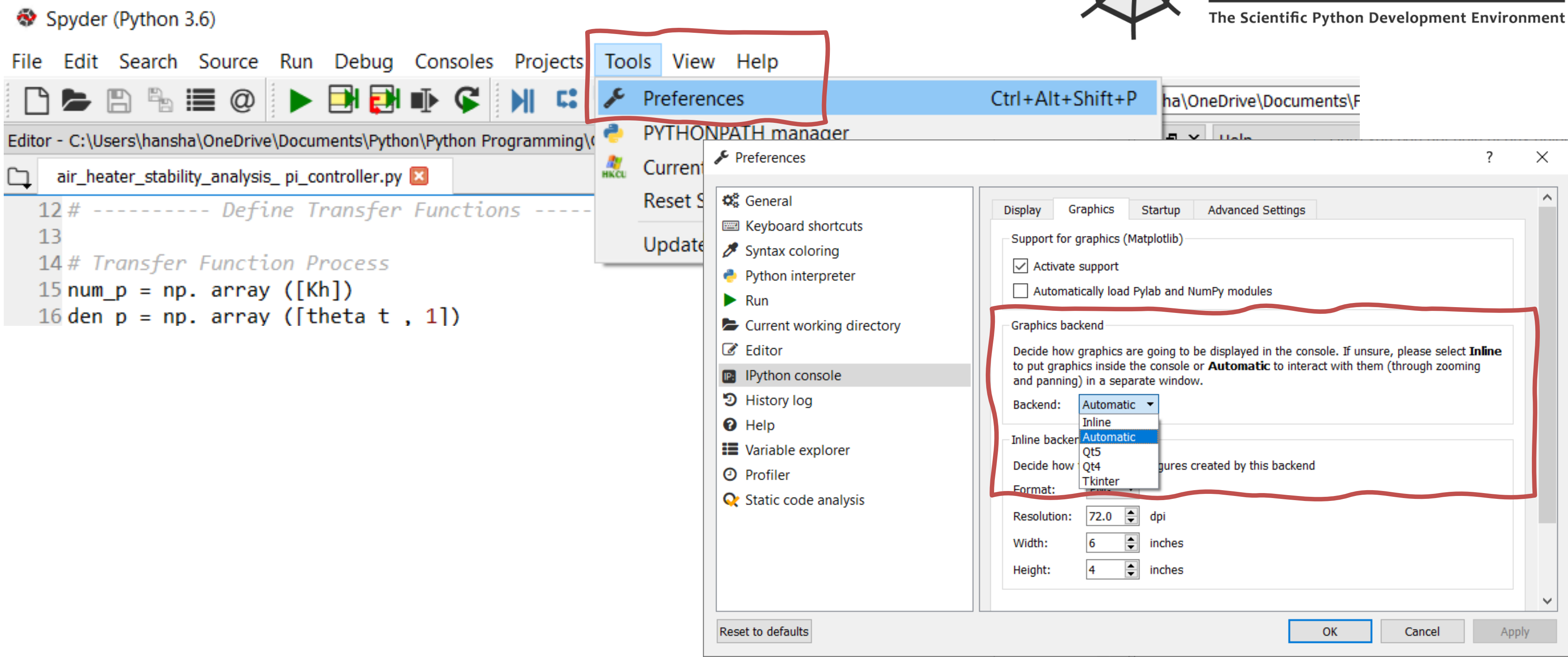
In this example we have two arrays with data. We want to plot x vs. y. We can assume x is a time series and y is the corresponding temperature in degrees Celsius.

```
import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
y = [5, 2,4, 4, 8, 7, 4, 8, 10, 9]

plt.plot(x,y)
plt.xlabel('Time (s)')
plt.ylabel('Temperature (degC)')
plt.show()
```



Matplotlib in Spyder

Typically you want to show figures and plots in separate windows



Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Preferences Ctrl+Alt+Shift+P
PYTHONPATH manager
Current
Reset S
Update
Editor - C:\Users\hansha\OneDrive\Documents\Python\Python Programming\
air_heater_stability_analysis_ pi_controller.py
Preferences
General
Keyboard shortcuts
Syntax coloring
Python interpreter
Run
Current working directory
Editor
IPython console
History log
Help
Variable explorer
Profiler
Static code analysis
Display Graphics Startup Advanced Settings
Reset to defaults
OK Cancel Apply

```
# ---------- Define Transfer Functions -----

# Transfer Function Process
num_p = np. array ([Kh])
den p = np. array ([theta t , 1])
```

Support for graphics (Matplotlib)

- [x] Activate support
- [ ] Automatically load Pylab and NumPy modules

Graphics backend

Decide how graphics are going to be displayed in the console. If unsure, please select **Inline** to put graphics inside the console or **Automatic** to interact with them (through zooming and panning) in a separate window.

Backend: Automatic (Inline, Automatic, Qt5, Qt4, Tkinter)

Inline backend

Decide how ... figures created by this backend

Format:

Resolution: 72.0 dpi

Width: 6 inches

Height: 4 inches

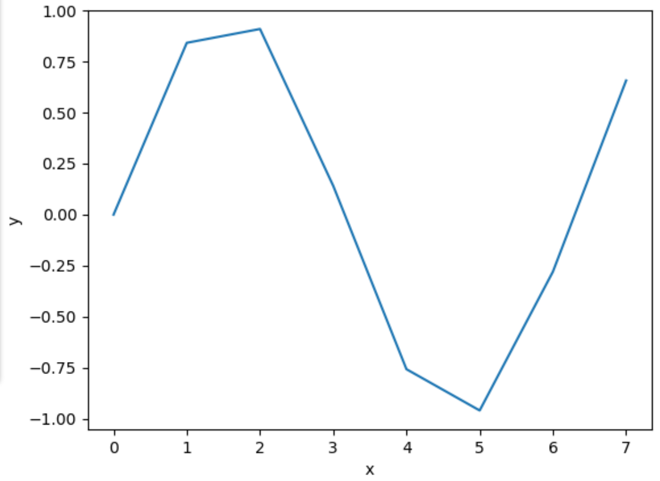
Matplotlib

Example: Plotting a Sine Curve

```
import numpy as np
import matplotlib.pyplot as plt

x = [0, 1, 2, 3, 4, 5, 6, 7]

y = np.sin(x)

plt.plot(x, y)
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

If you want grids you can use the **grid()** function

Note! The curve is not smooth due to few data points

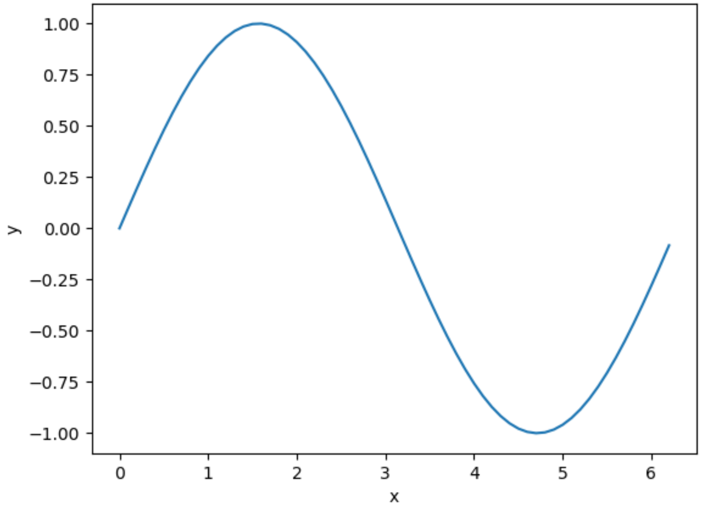


Matplotlib

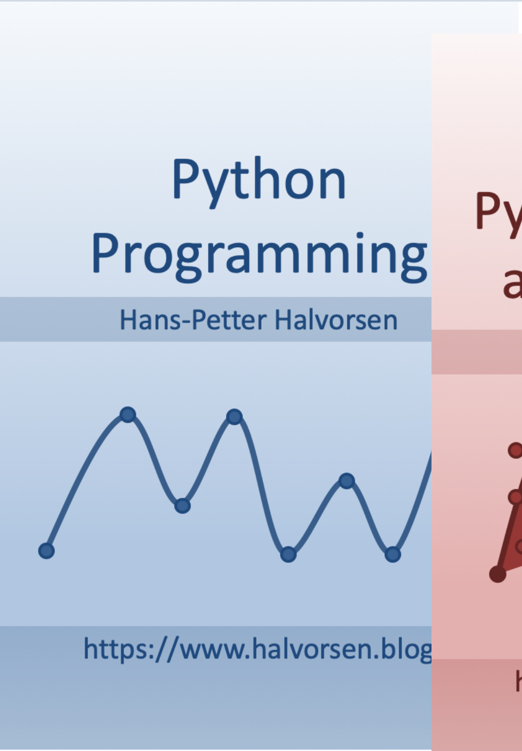
Improved Solution: Plotting a Sine Curve

```
import matplotlib.pyplot as plt
import numpy as np

xstart = 0
xstop = 2*np.pi
increment = 0.1

x = np.arange(xstart,xstop,increment)
y = np.sin(x)

plt.plot(x, y)
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

Better!

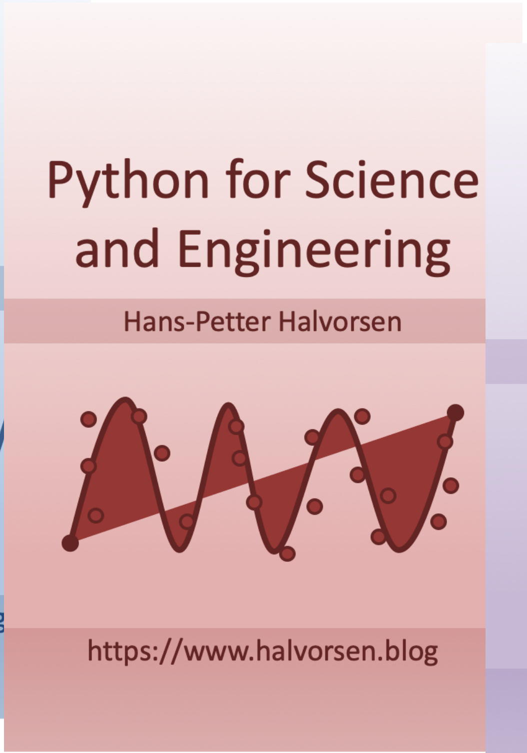


Additional Python Resources


Python Programming

Hans-Petter Halvorsen

https://www.halvorsen.blog


Python for Science and Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog

Python for Control Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog

Python for Software Development

Hans-Petter Halvorsen

Python Software Development

Do you want to learn Software Development?

OK Cancel

https://www.halvorsen.blog

https://www.halvorsen.blog/documents/programming/python/

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog