



CS-1004, Introduction to Programming for Non-Majors, A-Term 2014

Setting up Python 3.4 and numpy and matplotlib on your own Windows PC or laptop

Hugh C. Lauer[©]
Adjunct Professor
Worcester Polytechnic Institute

Programming assignments in CS-1004 will be in the programming language *Python* — specifically, version 3.4 of *Python*. In addition, you will need several *Python* packages, including one called *numpy* (meaning “Numerical Python”) and one called *matplotlib*, a *Python* version of the popular *Matlab* system. The first part of this document provides instructions for installing *Python 3.4* on *Windows 7* and *Windows 8* platforms. The second part of the document provides instructions on how to install additional *Python* packages, such as *numpy* and *matplotlib*.¹

Public laboratory computers at WPI will have the same version of *Python 3.4* and *numpy* installed on them for academic year 2014-2015. However, they will not have *matplotlib* or other additional packages that may be specified during the term.

In general, it is expected that assignments will be compatible among Windows, Macintosh, and Linux systems, provided that they all use compatible versions of *Python* and *numpy*.

Note: There are two different, incompatible versions of *Python* in general use around the world — *Python 2.7* and *Python 3.4*. Significant changes to the *Python* language were made between *Python 2.x* and *Python 3.y* (for all values of x and y). The *Python 3* language is cleaner, more self-consistent, and more user-friendly. Programs written for versions of *Python 2* will not necessarily run on *Python 3* installations; if they do run, they may get different answers to the same problem.

That being said, a lot of legacy *Python 2* code is still in use, and new *Python 2.7* code is still being written and distributed by organizations that have not yet upgraded to *Python 3*. Not all *Python 2* packages have been ported to *Python 3*.

Installing Python 3.4 on Windows Systems^{2, 3}

There are two variants of *Python 3.4* for Windows — a 32-bit version and a 64-bit version. Obviously, the 64-bit version requires a 64-bit Windows computer. Fortunately, most Win-

[©] Copyright 2014, Hugh C. Lauer. All rights reserved. Permission is given for use in courses at Worcester Polytechnic Institute, Worcester, Massachusetts.

¹ If you have a Macintosh or Linux computer or laptop, please refer to this documents instead:— [docx](#), [pdf](#)

² It is useful to print out the relevant section of this document. If you read it on-screen, the dialog boxes of the installation tend to obscure the instructions of the document, just when you need them the most!

³ These instructions have been tested on both *Windows 7* and *Windows 8*.

dows PCs sold over the past few years are 64-bit. However, the 32-bit version of *Python* can run on *both* 64-bit Windows PCs *and* 32-bit Windows PCs.

For this course, *we must use the 32-bit version of Python* on Windows. The reason is that the official release of *numpy* is currently available for Windows *only* in 32-bit format.

To obtain the correct version of *Python*, click on this link — [python-3.4.1.msi](#) — and download the resulting file to a convenient folder or directory. Alternatively, you may browse to

<http://www.cs.wpi.edu/~cs1004/a14/Resources>

and download them from there.

Note: If you go to an official *Python* download page, you will find a downloadable file named **python-3.4.1.amd64.msi**. *Do not install this!* It is the 64-bit version, and it is not compatible with the currently available version of *numpy*.

Double-click on the file **python-3.4.1.msi** to start the installation. You should be greeted by a dialog box resembling the following:–



Figure 1

If you subsequently see a dialog box resembling Figure 2 below for any version of *Python 3*, select *Remove Python* for that version.

This is important. There may already be a 64-bit *Python* installed without your knowing it, and this will cause problems with the installation of *numpy* and *matplotlib*.

Removing *Python* will take several minutes and may require you to confirm in one or more additional dialog boxes.



Figure 2

After you have removed your previous version of *Python*, click *Finish* and start over at Figure 1. After clicking *Next*, you should see a dialog box resembling the following:–

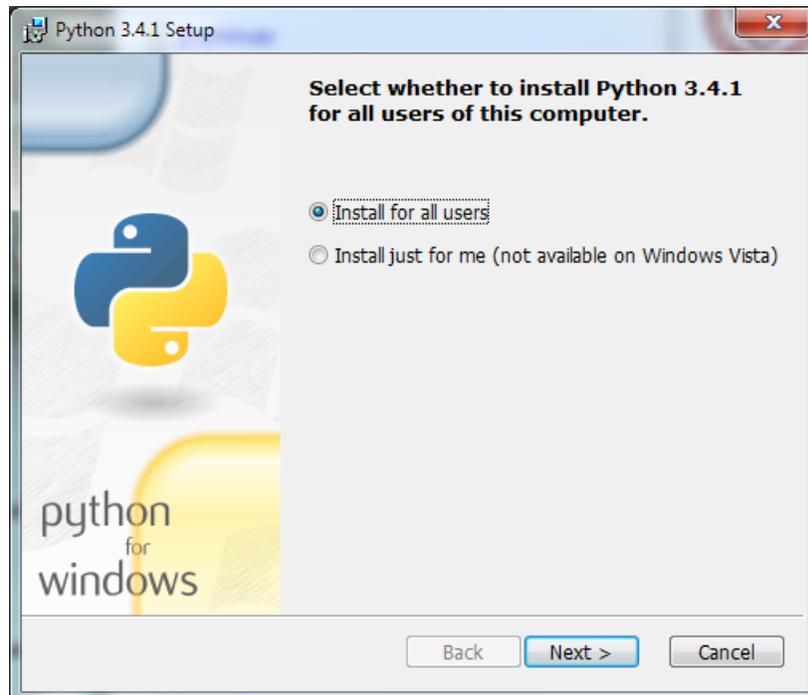


Figure 3

Whether you choose to install “for all users” or just for yourself is a matter of personal preference. Click *Next* to bring up the following dialog box.

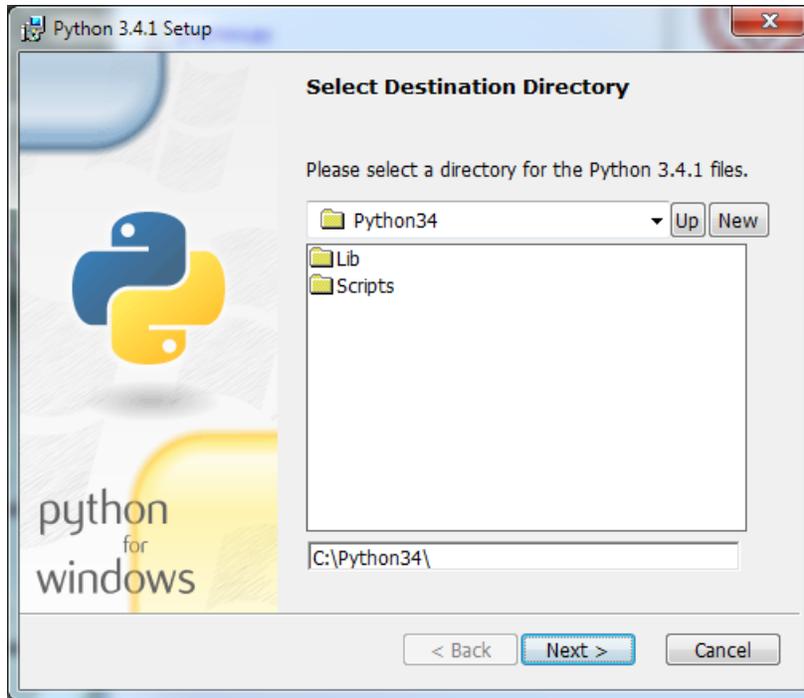


Figure 4

Click *Next* to select the default directory. If it tells you the directory already exists and asks if you are sure that you want to overwrite existing files, click *Yes*. In the next dialog box (Figure 5 below), you may customize the installation. Simply click *Next* without making any changes.

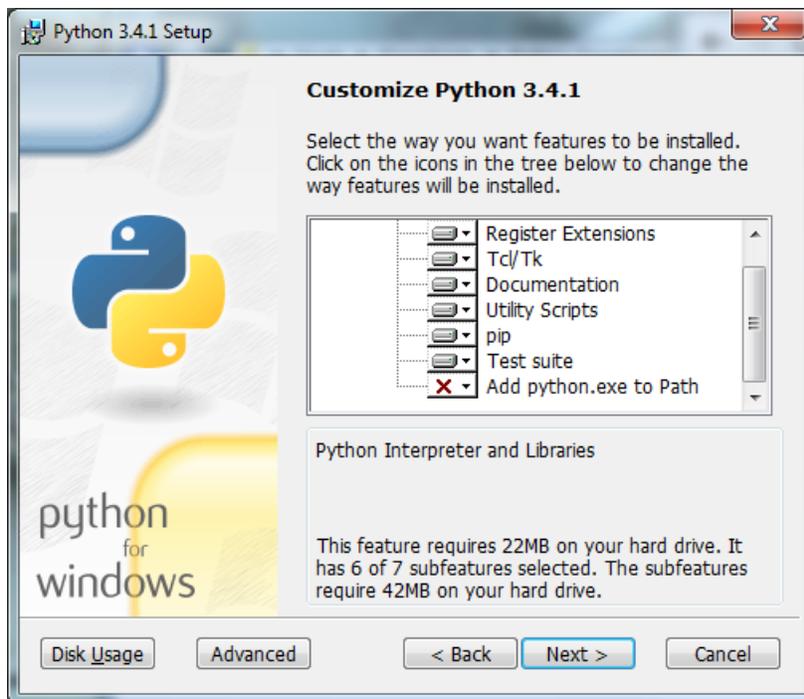


Figure 5

The installation should begin and will take several minutes and may require confirmation in additional dialog boxes.⁴ When it completes, you should see the final dialog box, below.



Figure 6

Click *Finish* to complete the installation of *Python 3.4.1*.

Testing your installation on Windows 7

If you are running *Windows 7*, you may confirm your installation by clicking the *Start* button to bring up the Windows *Start* menu. Select *All Programs* and scroll down to *Python 3.4*. Open this folder to expose the following shortcuts:–

⁴ On some computers, these additional dialog boxes are occasionally hidden behind other windows. If nothing seems to be happening, try clicking on or moving windows to look for such a dialog box.

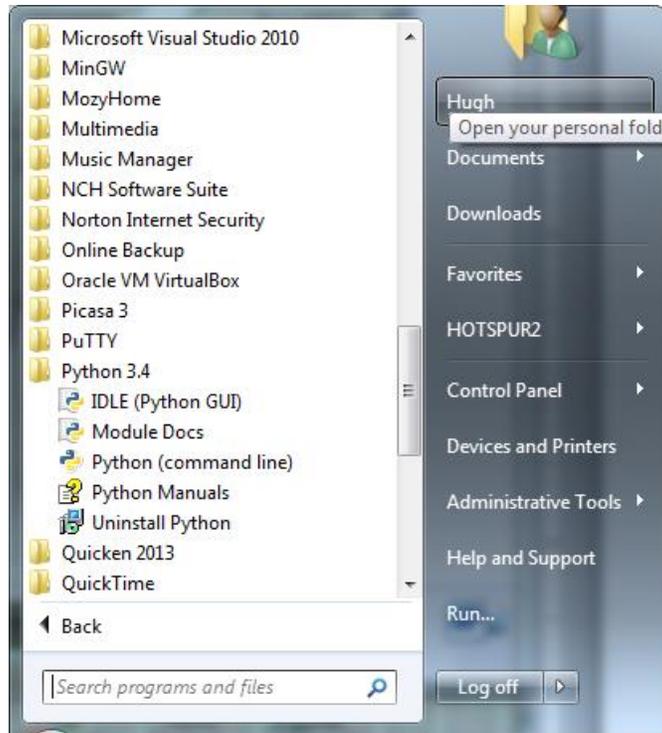


Figure 7

Click on *IDLE (Python GUI)* to bring up the following window (only the upper part of which is shown here):–

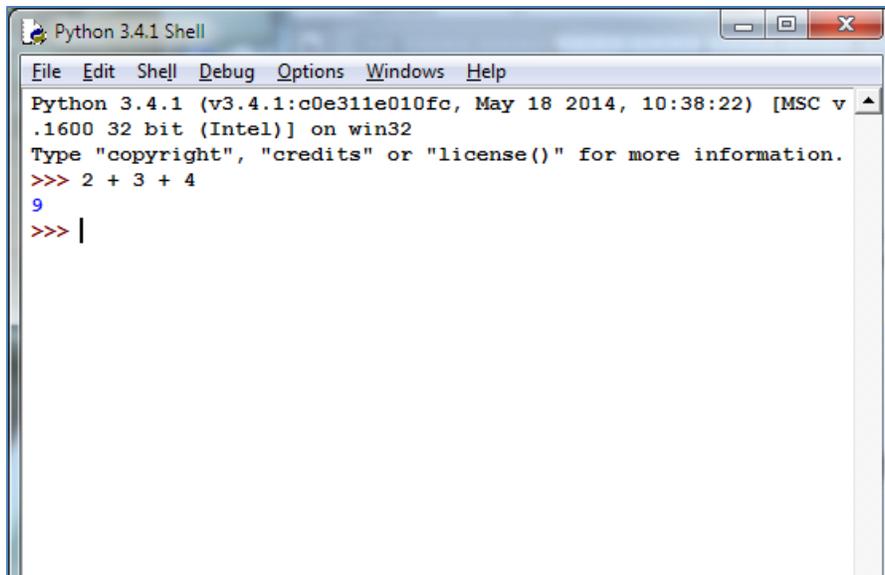


Figure 8

This is *IDLE*, the *Python* command prompt and graphical user interface. This is where we will start all programs and projects in this course. For now, simply type any *Python* statement or expression after the “>>>” prompt. For example, in Figure 8, the expression $2 + 3 + 4$ was typed and *Python* responded with the value 9. Continue testing by typing out the code on pages 10-11 of the textbook, just to make sure that your installation works as expected.

Testing your installation in Windows 8

Windows 8 does not have a *Start* button but rather a *Start* screen that is intended to make the user experience more like the smartphone experience.⁵ Unfortunately, when *Python* is installed as instructed above, its icon does not automatically appear on the *Start* screen. It also does not appear in the list of apps.

To find it, move the cursor to the upper-right or lower-right corner of the screen to expose the *Windows 8* pallet of “charms”. Select the *Search* charm to bring up a *Search* box. Type the word “Python.” This will bring up a list of matching items, shown in Figure 9 below.



Figure 9

Note that this list is similar to the *Python 3.4* folder in the Start Menu in Figure 7. *Right-click* on the item labeled *IDLE (Python GUI)*. From the menu, select “Pin to Start” to cause an icon to be added to the *Start* screen. You may also want to pin the item to the *Task bar* (i.e., the bar of tiny icons at the bottom of the screen). You may also select “Open file location,” which will bring up the following window:–

⁵ In the Professor’s opinion, this was a mistake.

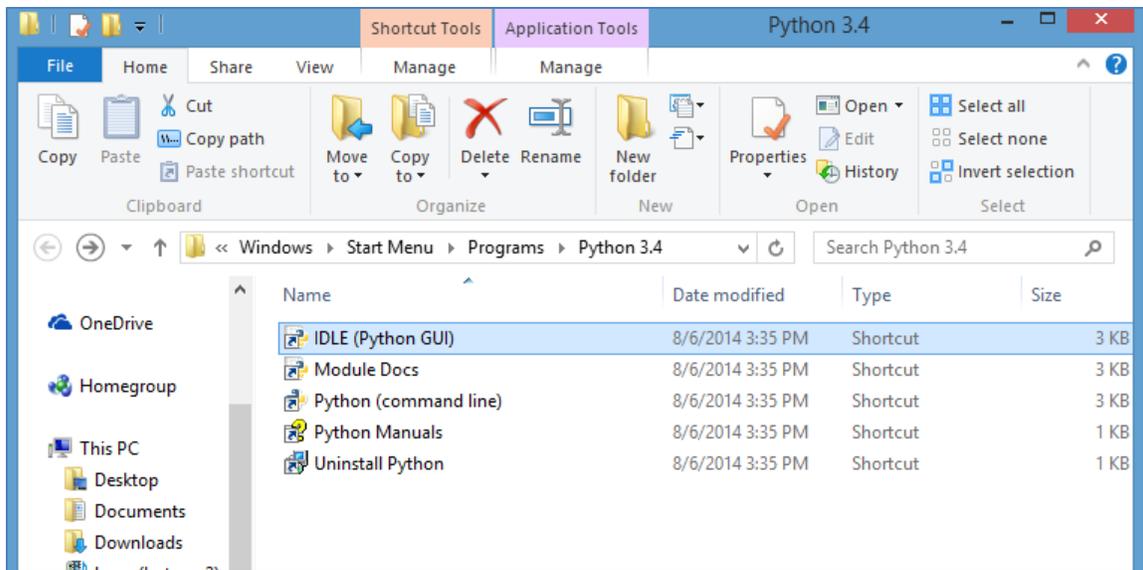


Figure 10

From this window, you can copy any or all of the *Python* links to the desktop.

To test your installation, double-click on the *IDLE (Python GUI)* icon and carry out the same tests as shown above under Figure 8.

Installing *matplotlib*, *numpy*, and other packages

One of the many benefits of *Python* is the vast number of third-party packages that can be downloaded and used by your *Python* programs. Many of these are open-source and free. For this course, we will use at least the following:—

- *matplotlib* (a package for creating 2D plots and graphs similar to *Matlab*),
- *numpy* (meaning “Numerical Python,” a package for efficient handling of large arrays of numerical data), and
- *graphics.py*, a simple tool written in *Python 3* and created by the textbook author for making simple drawings.

Installing *Graphics.py*

To install *graphics.py*, click on this link — [graphics.py](#) — and download the file to the folder where you keep your *Python* programs. Follow the instructions on p.488 of the textbook.

Installing *numpy 1.8.1*

In either *Windows 7* or *Windows 8*, click on the following link —[numpy-1.8.1-win32-superpack-python3.4.exe](#) — to download the *numpy* installer. Open this to begin the installation. After confirming that you do want to allow the system to install software, it will start the installation and show the following dialog box:—

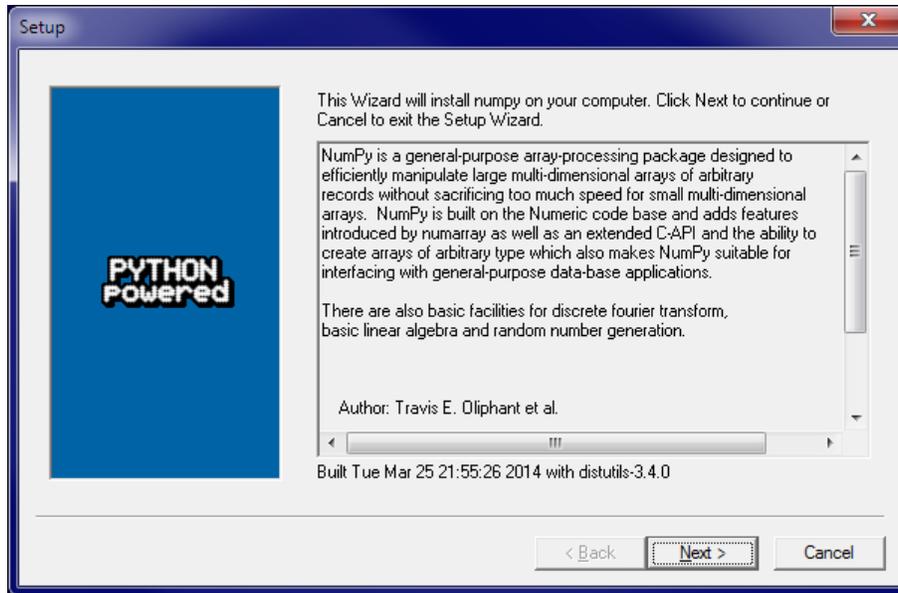


Figure 11

Click *Next*. If your installation of *Python 3.4* is correct, you should get the following dialog:–

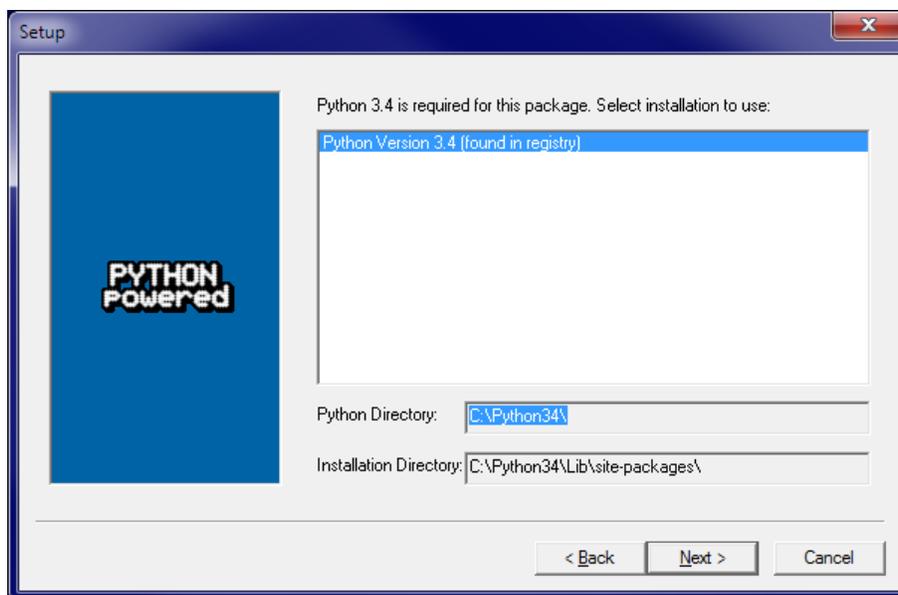


Figure 12

If instead, it complains that you do not have *Python 3.4* installed, ask for help. Such a complaint could arise if *Python* was not correctly installed or if you have the 64-bit version.

After the installation completes, click *Finish*. Note that you might have to click somewhere in some window to get the *Finish* dialog box to pop up. Note also that this installer contains all of the dependencies of *numpy* — i.e., other packages needed by *numpy* to run. They are installed silently.

You can test your installation of *numpy* by opening an *IDLE* window, as in Figure 8. Type or paste the following commands into *IDLE*, one line at a time, exactly as written:–

```

import numpy as np
np.__version__6
a = np.arange(10)
a
b = np.arange(1, 9, 2)
b
c = np.eye(3)
c
d = np.diag(np.array([1, 2, 3, 4]))
d

```

The result should resemble the following figure:–

```

Python 3.4.1 Shell
File Edit Shell Debug Options Windows Help
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v
.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import numpy as np
>>>
>>> np.__version__
'1.8.1'
>>>
>>> a = np.arange(10)
>>> a
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
>>>
>>> b = np.arange(1, 9, 2)
>>> b
array([1, 3, 5, 7])
>>>
>>> c = np.eye(3)
>>> c
array([[ 1.,  0.,  0.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.]])
>>>
>>> d = np.diag(np.array([1, 2, 3, 4]))
>>> d
array([[1, 0, 0, 0],
       [0, 2, 0, 0],
       [0, 0, 3, 0],
       [0, 0, 0, 4]])
>>>
Ln: 28 Col: 4

```

Figure 13

Congratulations! You have now installed a working versions *numpy 1.8.1*.

⁶ Note that the word “**version**” is preceded by *two* underscore characters and followed by *two* more underscore characters.

Installing Matplotlib on Windows⁷

Installing *Matplotlib* is not nearly so straightforward. There is no “officially released” installer that captures all of the dependencies. The only installer that the Professor could find at the time of this writing is an “unofficial” one that contains just *Matplotlib* itself. Therefore, you need to download and manually install each of the packages upon which *Matplotlib* depends.

Download and open [Windows/matplotlib-1.3.1.win32-py3.4.exe](#). The installer will open two dialog boxes similar to but not identical to Figure 11 and Figure 12. Allow the installation to run to completion.

Next, download and run the installers for each of the following packages:–

- [Windows/python-dateutil-2.2.win32-py3.4.exe](#)
- [Windows/pyparsing-2.0.2.win32-py3.4.exe](#)
- [Windows/pytz-2014.4.win32-py3.4.exe](#)
- [Windows/six-1.7.3.win32-py3.4.exe](#)

These should all install uneventfully, again with dialog boxes resembling Figure 11 and Figure 12.

To test your *matplotlib* installation, type or paste the following commands into IDLE, one line at a time, exactly as written:–

```
from matplotlib import pyplot
pyplot.plot([1, 2, 3, 4], [1, 4, 9, 16])
pyplot.show()
```

The IDLE window should look something like the following:–

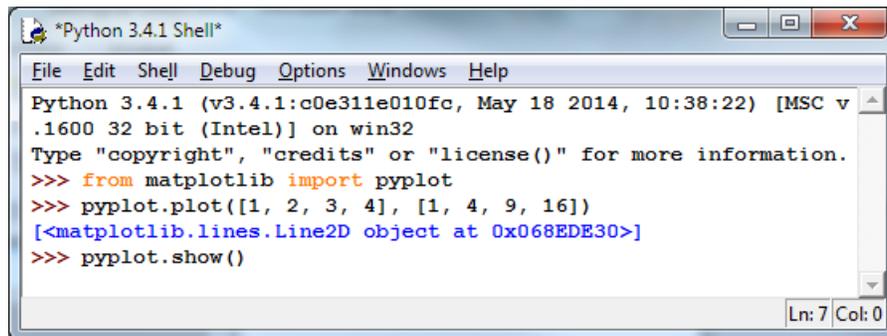


Figure 14

After you type the **ENTER** key following the last line, the following window should appear:–

⁷ Note that *numpy 1.8.1* is installed on the public laboratory computers, but *matplotlib* is not. The installation of *matplotlib* was not ready when the system image of the public computers had to be frozen.

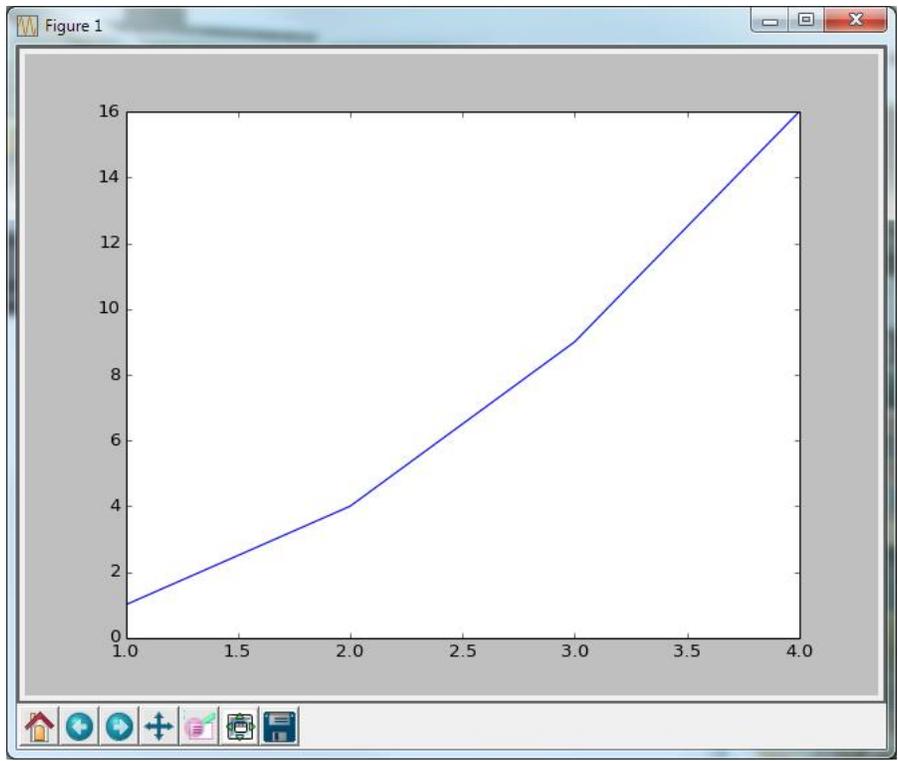


Figure 15

To close this window, click on the “close” button in the upper right.
Congratulations! You now have a working version of *matplotlib* installed.