CS-1004, Introduction to Programming for Non-Majors, A-Term 2015

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 *Python 3.4*, *numpy*, and *matplotlib* installed on them for the academic year 2015-2016.

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. Although almost all Windows PCs sold over the past few years are 64-bit systems, these in-

[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*. *Windows 10* was not yet available.

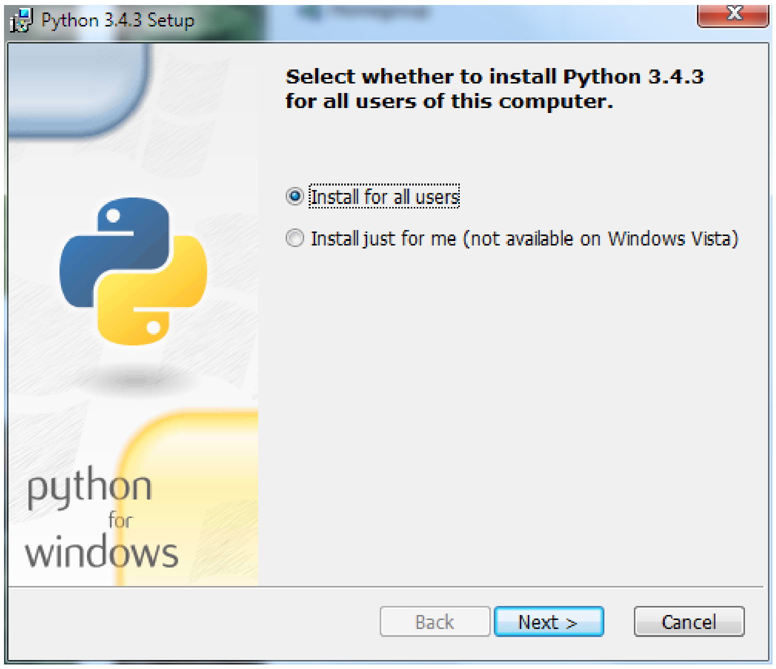
structions are for installing the 32-bit version, because we will be using a package called *numpy* that is only available in 32-bit releases for Windows.

To obtain the correct version of *Python*, click on this link — python-3.4.3.msi — and download the resulting file to a convenient folder or directory. Alternatively, you may browse to

http://www.cs.wpi.edu/~cs1004/a15/Resources

and download it from there.

Double-click on the file **`python-3.4.3.msi`** to start the installation. You should be greeted by a dialog box resembling the following:–



Select whether to install Python 3.4.3 for all users of this computer.

(●) Install for all users
( ) Install just for me (not available on Windows Vista)

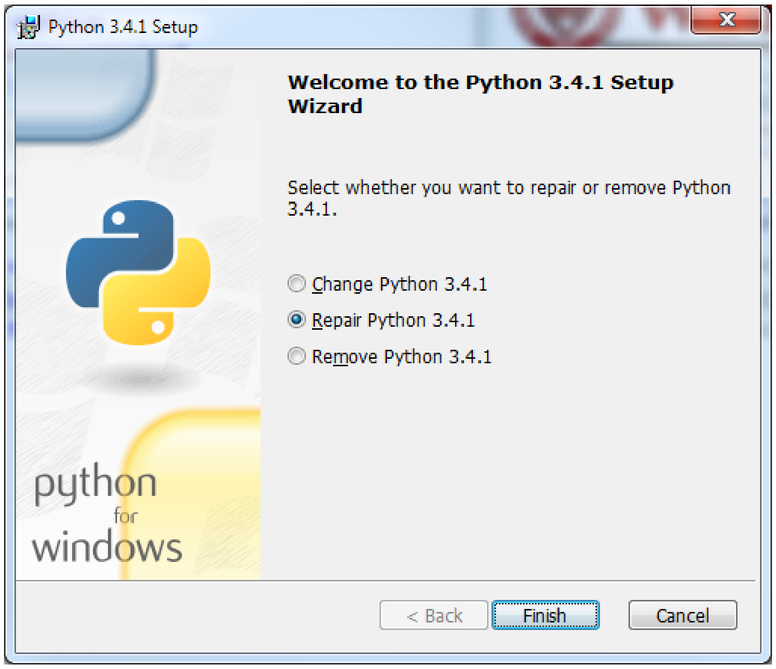
Figure 1

Whether you choose to install "for all users" or just for yourself is a personal preference.

If instead of Figure 1, you see a dialog box resembling Figure 2 below for any version of *Python*, select *Remove Python* for that version. This will remove an old, stale version, for example, an old 64-bit version of *Python 3*. Removing *Python* will take several minutes and may require you to confirm in one or more additional dialog boxes.

Note: Even if you don't see the dialog box of Figure 2, if there is a previous version of *Python* installed on your computer, you should uninstall it before continuing.

You can also remove an old version of *Python* using the **Start** menu. Select the *Python* folder and the *Remove Python x.y* menu item in that folder, where *x.y* indicates the version.



Python 3.4.1 Setup

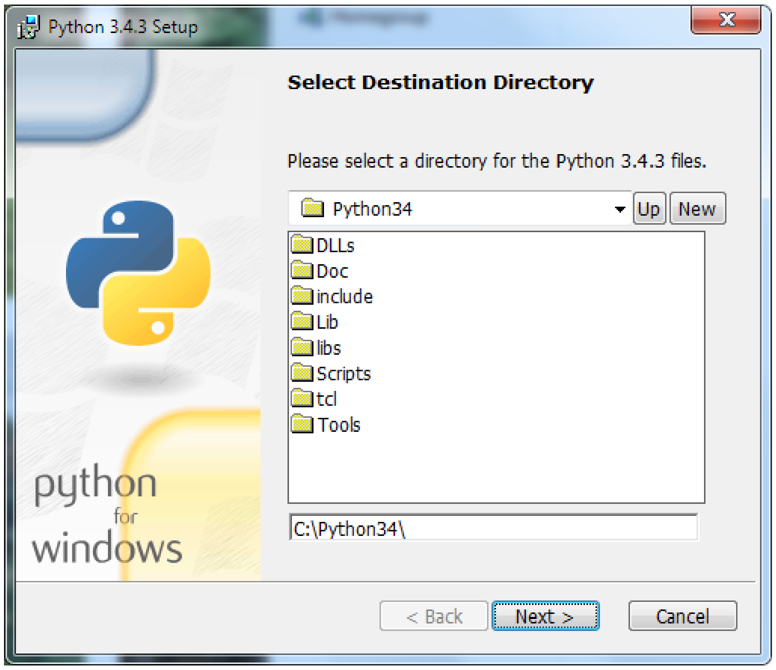
Welcome to the Python 3.4.1 Setup Wizard

Select whether you want to repair or remove Python 3.4.1.

- Change Python 3.4.1
- [✓ Repair Python 3.4.1]
- Remove Python 3.4.1

Figure 2

After removing the previous version of *Python*, click *Finish* and start over at Figure 1. Then click *Next* to bring up the following dialog box.



Python 3.4.3 Setup

Select Destination Directory

Please select a directory for the Python 3.4.3 files.

Python34

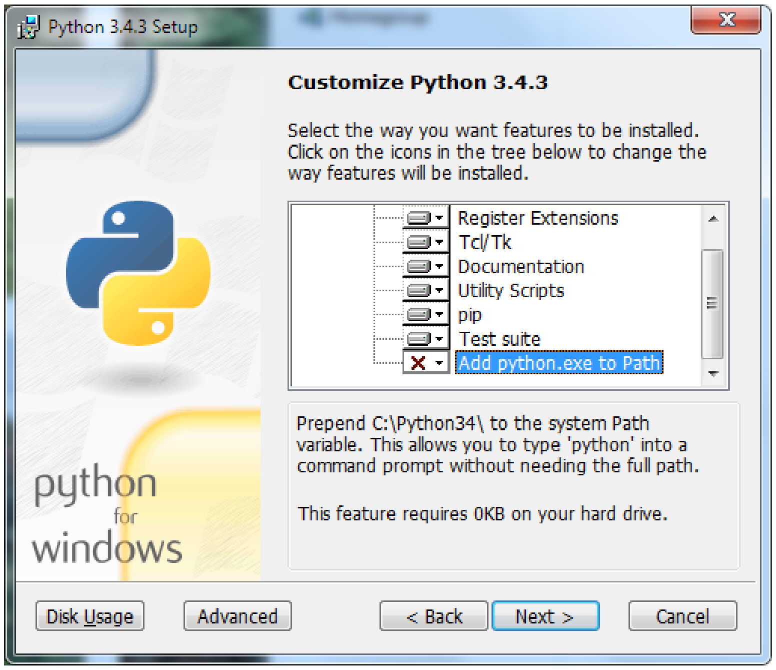
DLLs
Doc
include
Lib
libs
Scripts
tcl
Tools

C:\Python34\

Figure 3

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 4 below), you will need to customize the installation. Scroll down to the bottom and click on the "X" next to *Add python.exe to Path*. It will expand this line to several options. Select *Will be installed on local hard drive*. This facility lets you run *Python* and related programs from command prompts, something that you will need to do later on when installing other packages and also during the term.



Customize Python 3.4.3

Select the way you want features to be installed. Click on the icons in the tree below to change the way features will be installed.

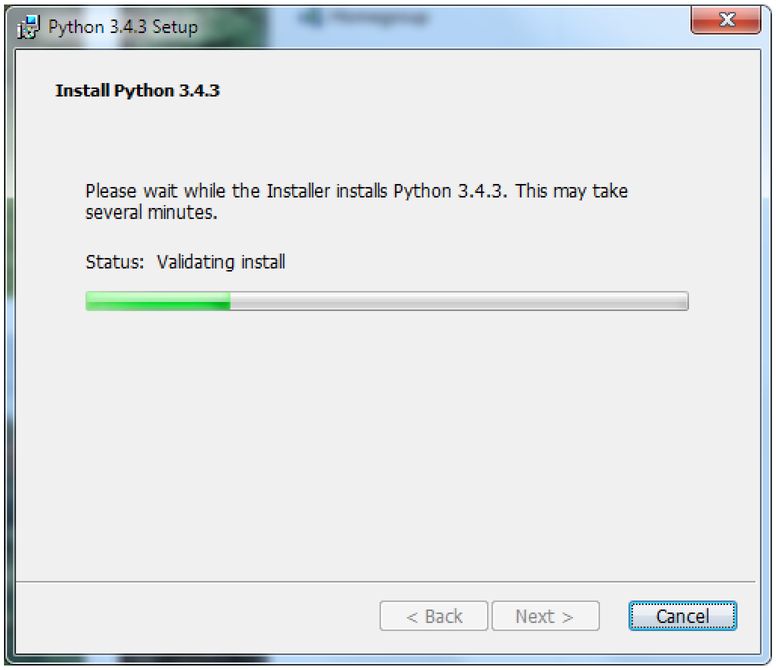
- Register Extensions
- Tcl/Tk
- Documentation
- Utility Scripts
- pip
- Test suite
- Add python.exe to Path

Prepend C:\Python34\ to the system Path variable. This allows you to type 'python' into a command prompt without needing the full path.

This feature requires 0KB on your hard drive.

Figure 4

Click *Next* to begin the installation. The progress of the installation is shown in a dialog box resembling Figure 5 below.



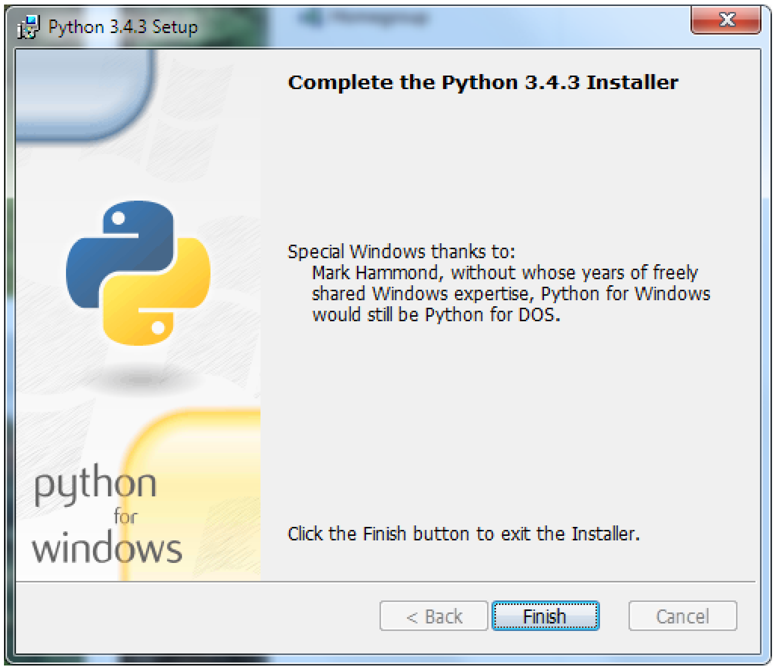
Install Python 3.4.3

Please wait while the Installer installs Python 3.4.3. This may take several minutes.

Status: Validating install

Figure 5

The installation should begin, will take several minutes, and may require confirmation in additional dialog boxes.[4] Also, a text window may appear briefly showing the status of parts of the installation. When the installation completes, you should see the final dialog box, below.



Complete the Python 3.4.3 Installer

Special Windows thanks to:
Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

Click the Finish button to exit the Installer.

Figure 6

Click *Finish* to complete the installation of *Python 3.4.3*.

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 shortcuts similar to the following:–

[4] 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.

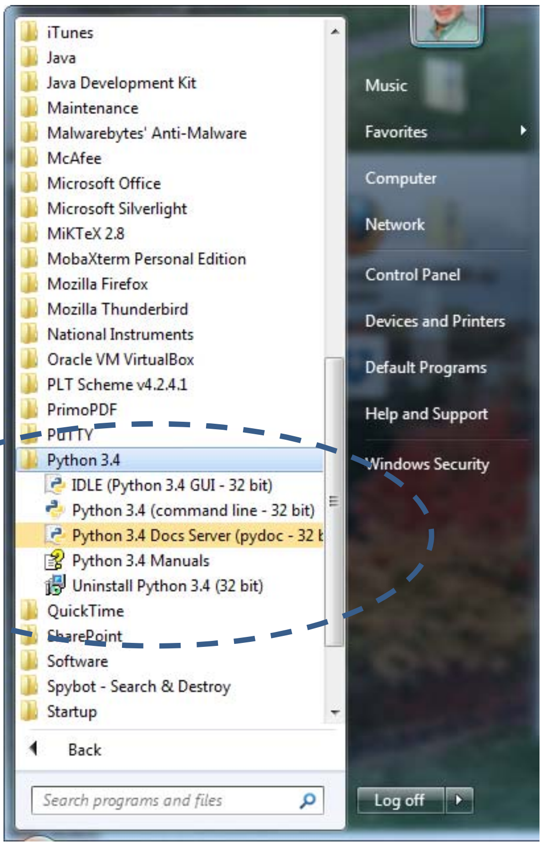
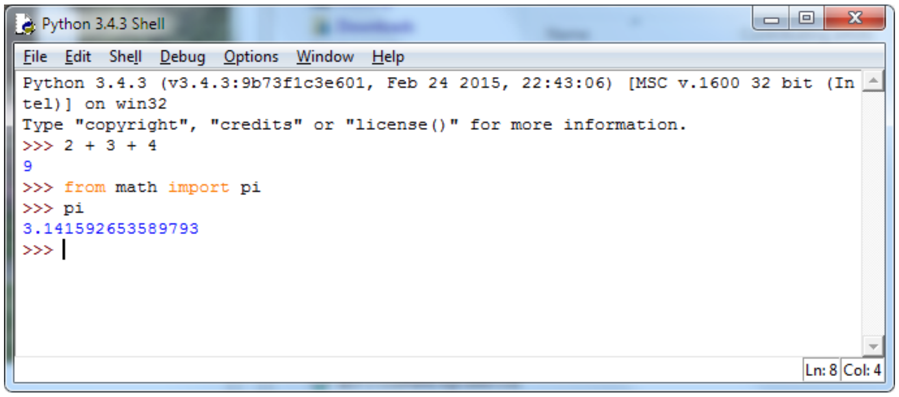


Figure 7

Click on *IDLE (Python GUI)* to bring up the following window (only a shortened version is shown here):–



```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 2 + 3 + 4
9
>>> from math import pi
>>> pi
3.141592653589793
>>> |
```

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, such as shown in Figure 9 below.

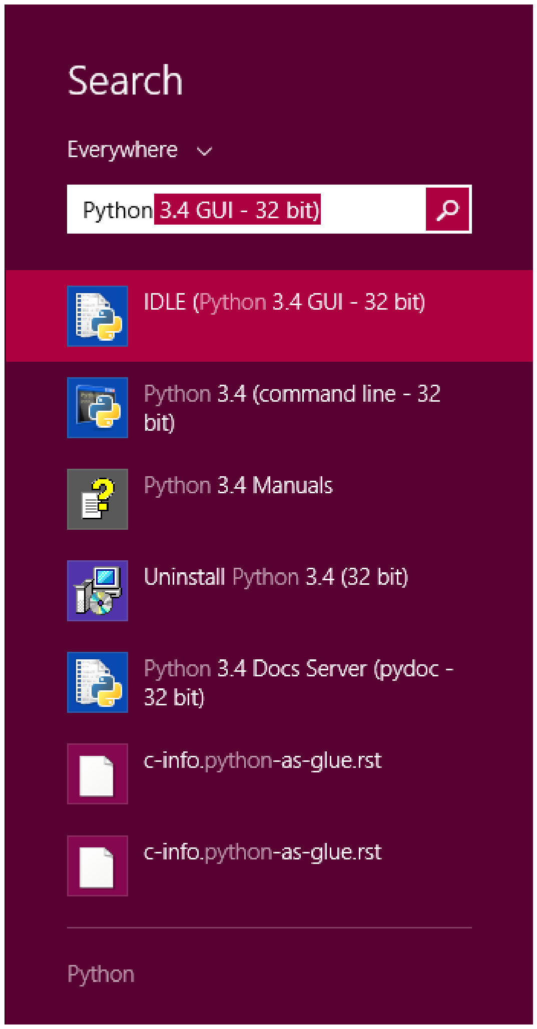
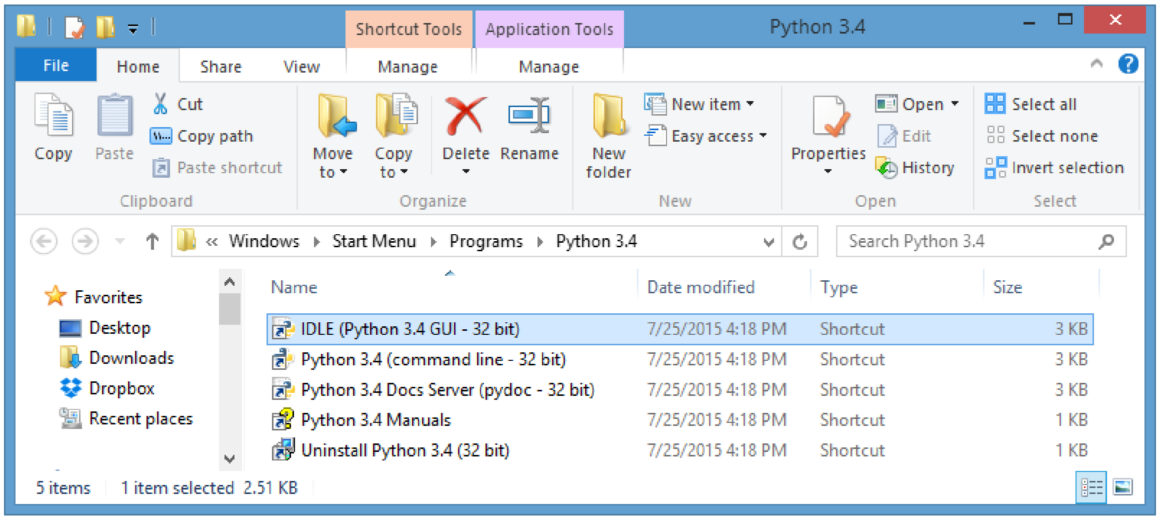

Search
Everywhere
Python 3.4 GUI - 32 bit)
IDLE (Python 3.4 GUI - 32 bit)
Python 3.4 (command line - 32 bit)
Python 3.4 Manuals
Uninstall Python 3.4 (32 bit)
Python 3.4 Docs Server (pydoc - 32 bit)
c-info.python-as-glue.rst
c-info.python-as-glue.rst
Python

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 3.4 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.



Name	Date modified	Type	Size
IDLE (Python 3.4 GUI - 32 bit)	7/25/2015 4:18 PM	Shortcut	3 KB
Python 3.4 (command line - 32 bit)	7/25/2015 4:18 PM	Shortcut	3 KB
Python 3.4 Docs Server (pydoc - 32 bit)	7/25/2015 4:18 PM	Shortcut	3 KB
Python 3.4 Manuals	7/25/2015 4:18 PM	Shortcut	1 KB
Uninstall Python 3.4 (32 bit)	7/25/2015 4:18 PM	Shortcut	1 KB

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, also needed by *matplotlib*), and
- *graphics.py*, a simple tool written in *Python 3* and created by the textbook author for making simple drawings.[6]

Click on the following links to download the respective packages to a convenient folder:–

- numpy-1.9.2-win32-superpack-python3.4.exe
- matplotlib-1.4.3.win32-py3.4.exe

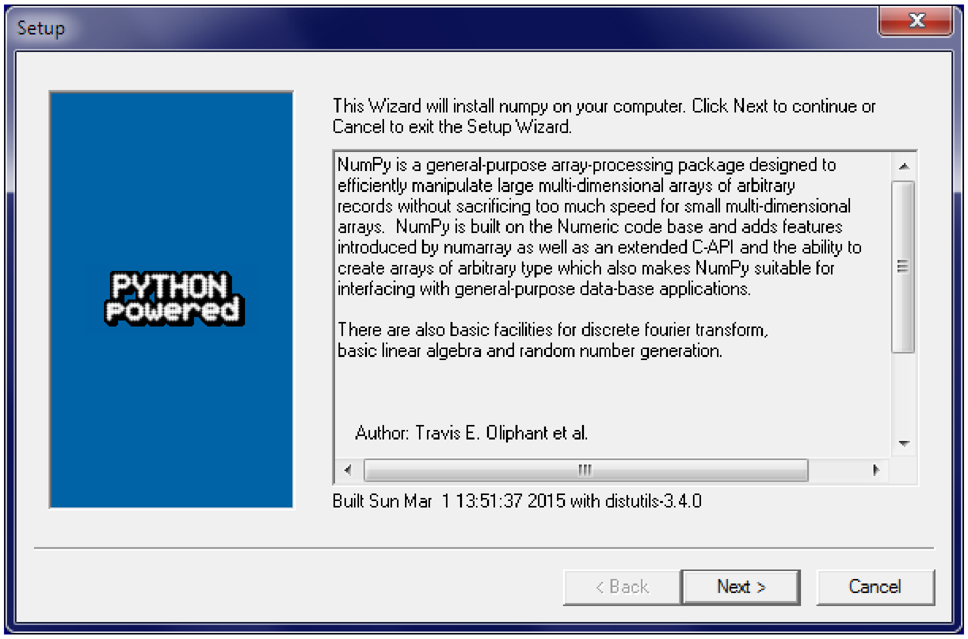
Installing Graphics.py

Graphics.py is a simple drawing package that we will use a lot in this course. To install it, 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.

[6] You can download the packages from http://www.cs.wpi.edu/~cs1004/a15/Resources/Windows, except for graphics.py, which is at http://web.cs.wpi.edu/~cs1004/a15/Resources/graphics.py.

Installing numpy 1.9.2

The numpy package needs to be installed immediately after you install *Python 3.4.2* itself. In either *Windows 7* or *Windows 8*, double-click (or open) the *numpy* installer that you downloaded above. After confirming that you do want to allow the system to install software, it will start the installation and show the following dialog box:–



Setup

This Wizard will install numpy on your computer. Click Next to continue or Cancel to exit the Setup Wizard.

NumPy is a general-purpose array-processing package designed to efficiently manipulate large multi-dimensional arrays of arbitrary records without sacrificing too much speed for small multi-dimensional arrays. NumPy is built on the Numeric code base and adds features introduced by numarray as well as an extended C-API and the ability to create arrays of arbitrary type which also makes NumPy suitable for interfacing with general-purpose data-base applications.

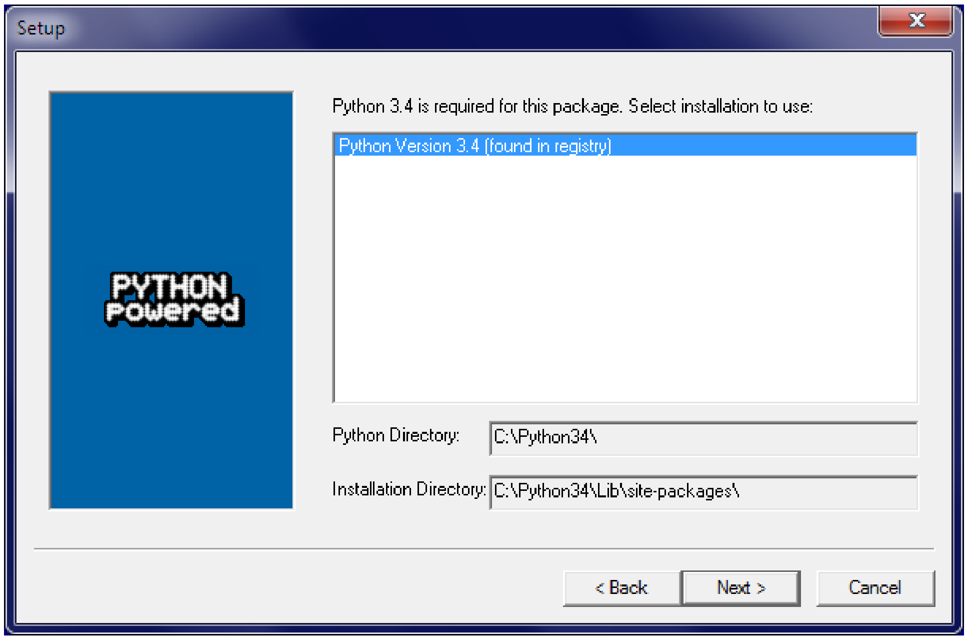
There are also basic facilities for discrete fourier transform, basic linear algebra and random number generation.

Author: Travis E. Oliphant et al.

Built Sun Mar 1 13:51:37 2015 with distutils-3.4.0

Figure 11

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



Setup

Python 3.4 is required for this package. Select installation to use:

Python Version 3.4 (found in registry)

Python Directory: C:\Python34\

Installation Directory: C:\Python34\Lib\site-packages\

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 an incompatible version.

After the installation completes, click *Finish*. Note that you might have to click somewhere in some window to get the *Close* 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. These are installed silently.

Make a quick 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:–

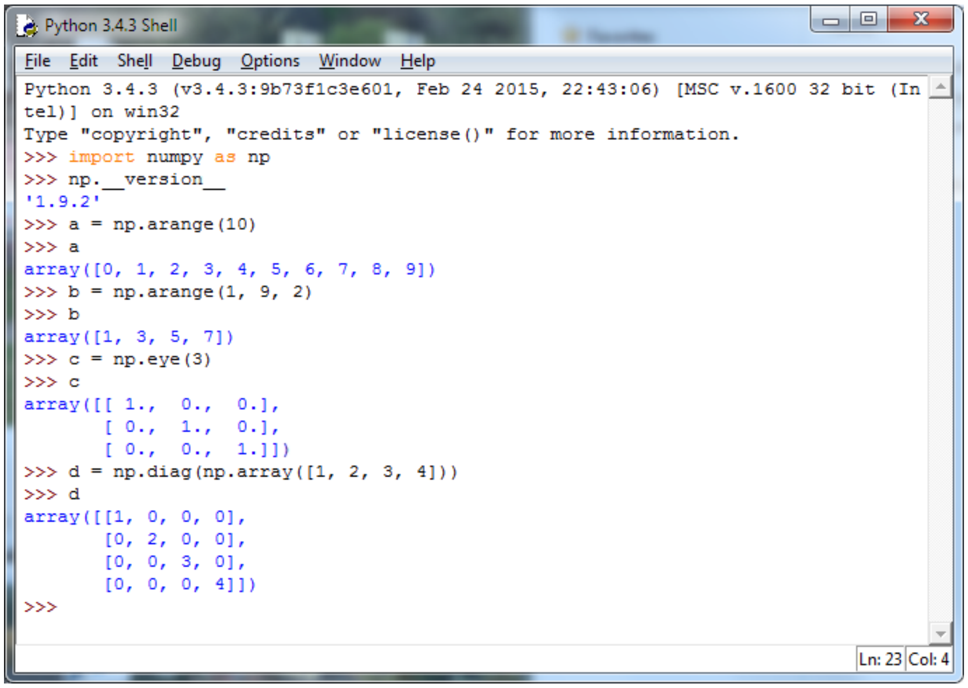
```
import numpy as np
np.__version__
```
[7]
```
a = np.arange(10)
```
[8]
```
a

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 Figure 13:–



```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import numpy as np
>>> np.__version__
'1.9.2'
>>> 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]])
>>>
```

Figure 13

A more comprehensive test of *numpy* is described later in this document.

[7] Note that the word "**`version`**" is preceded by *two* underscore characters and followed by *two* more underscore characters.

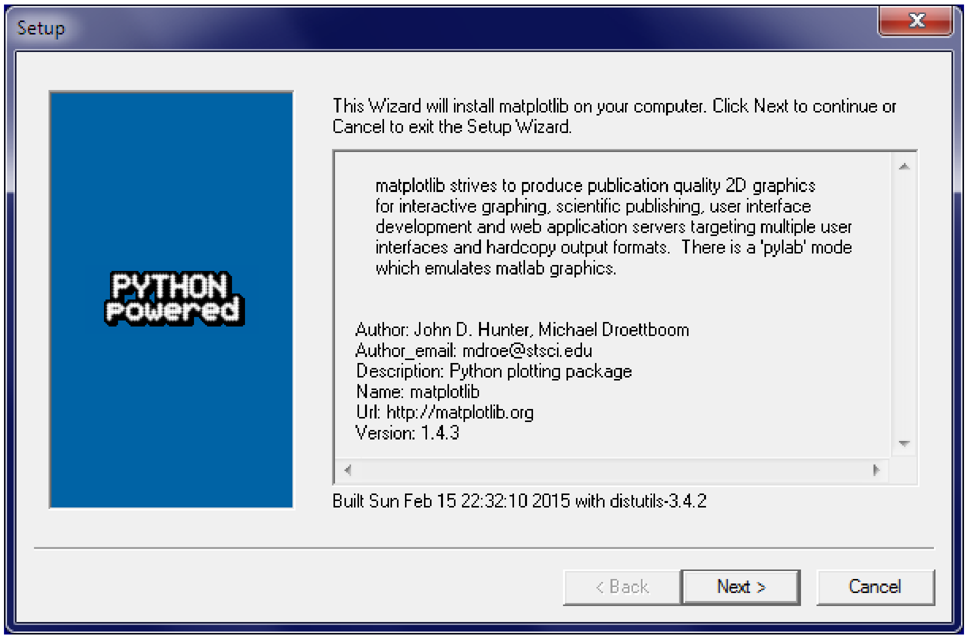
[8] Note the spelling of "**`arange`**" with only one '**`r`**'.

Installing Matplotlib on Windows

Installing *matplotlib* requires several steps. The official installer captures some, but not all, of the dependencies needed by *matplotlib*.[9] To get started with *Matplotlib*, double-click or open the installer program

matplotlib-1.4.3.win32-py3.4.exe

that you downloaded above. After asking you to verify that you really do want to do this installation, it brings up the following window:–



This Wizard will install matplotlib on your computer. Click Next to continue or Cancel to exit the Setup Wizard.

matplotlib strives to produce publication quality 2D graphics for interactive graphing, scientific publishing, user interface development and web application servers targeting multiple user interfaces and hardcopy output formats. There is a 'pylab' mode which emulates matlab graphics.

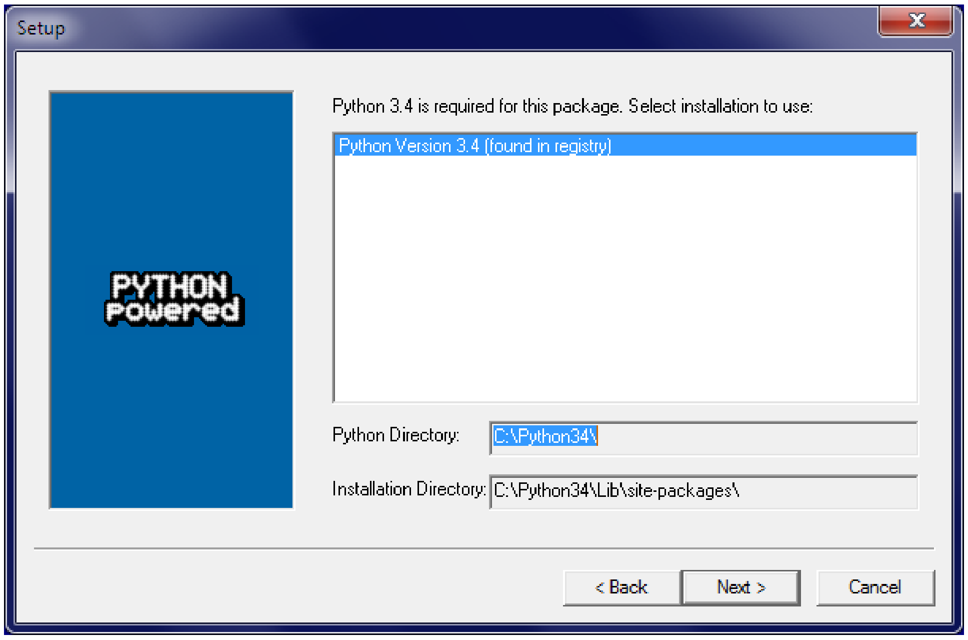
Author: John D. Hunter, Michael Droettboom
Author_email: mdroe@stsci.edu
Description: Python plotting package
Name: matplotlib
Url: http://matplotlib.org
Version: 1.4.3

Built Sun Feb 15 22:32:10 2015 with distutils-3.4.2

Figure 14

Click *Next* to bring up the following window:–

[9] A *dependency* is another program or piece of software that needed to support a particular piece of software.


Setup

Python 3.4 is required for this package. Select installation to use:

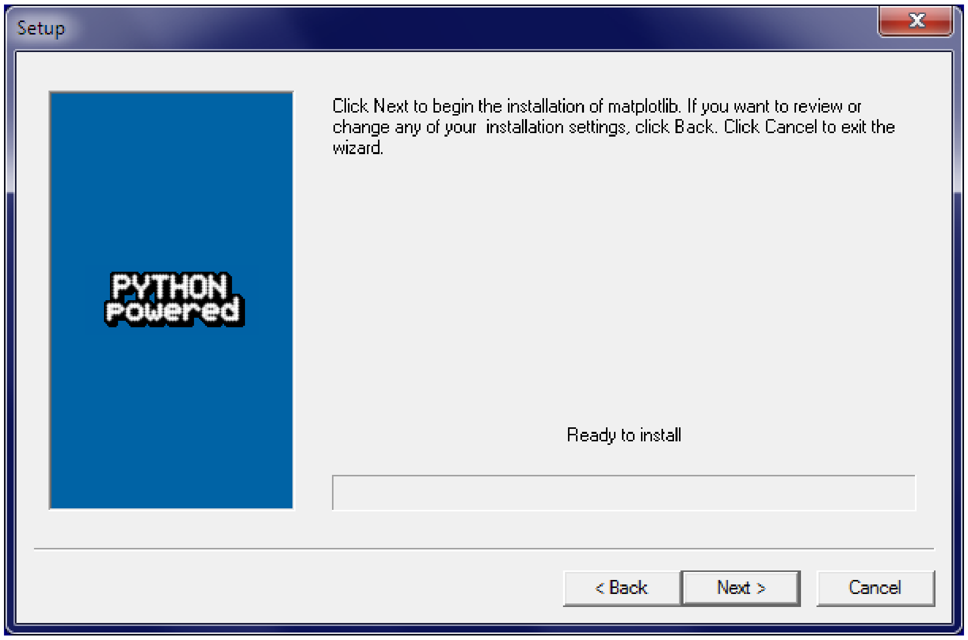
Python Version 3.4 (found in registry)

Python Directory: C:\Python34\

Installation Directory: C:\Python34\Lib\site-packages\

Figure 15

Click *Next* again to bring up a third window:–


Setup

Click Next to begin the installation of matplotlib. If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.

Ready to install

Figure 16

Clicking *Next* again starts the installation of *matplotlib*. It is probably necessary to click in the resulting screen to see the *Finish* window. Click the *Finish* button to complete this phase of installing *matplotlib*.

To install the remaining dependencies, bring up a *Command Prompt* window. This can be found in the *Accessories* folder of the *Start* menu in Windows 7. In Windows 8, you may have search for it, the same way you searched in Figure 9.

In the command prompt window, type or paste each of the following four commands. Let each command finish before typing the next one.

```
pip install --upgrade pip
```

```
pip install --upgrade matplotlib
```

```
pip install nose
```

```
pip list
```

The first of these command sometimes produces error messages, but it works anyway. The second command finds and installs the remaining dependency packages for *matplotlib*, including **`six`**, **`python-dateutil`**, **`pytz`**, and **`pyparsing`**. The third installs **`nose`**, a package used to test *numpy*. The last command lists the packages that are currently installed with your *Python 3.4* installation. The list should include the following:–

```
matplotlib (1.4.3)
nose (1.3.7)
numpy (1.9.2)
pip (7.1.1)
pyparsing (2.0.3)
python-dateutil (2.4.2)
pytz (2015.4)
setuptools (12.0.5)
six (1.9.0)
```

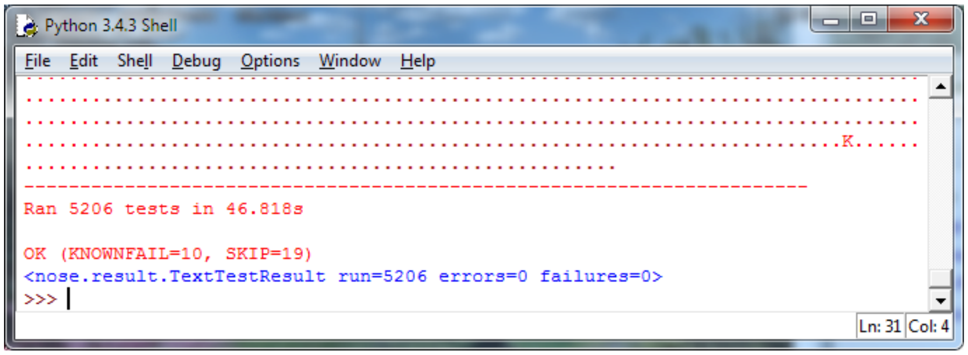
Finally, type the **`exit`** command to close the Command Prompt.

Testing Your Installation

To carry out the comprehensive test of *numpy*, open a new *IDLE* window and type the following two commands:–

```
import numpy
numpy.test()
```

This uses **`nose`** to run the standard package of *numpy* tests for a minute or three. It prints a bunch of stuff in the *IDLE* window. Although some of the output may look like error messages, these are known issues with the tests. The test should end with something resembling the following:–



```
.............................................................................
.............................................................................
.............................................................................
......................................................................K......
...................................................
----------------------------------------------------------------------
Ran 5206 tests in 46.818s

OK (KNOWNFAIL=10, SKIP=19)
<nose.result.TextTestResult run=5206 errors=0 failures=0>
>>> 
```

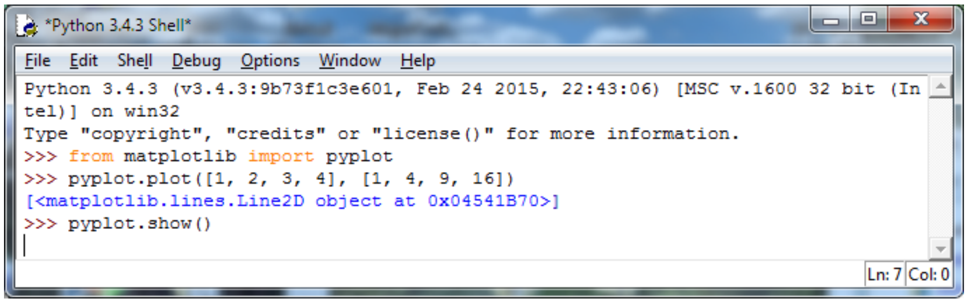
Figure 17

Congratulations! You have now installed a working versions *numpy 1.9.2.*

Finally, test your *matplotlib* installation by typing or pasting the following commands into an IDLE window, 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:–



```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> from matplotlib import pyplot
>>> pyplot.plot([1, 2, 3, 4], [1, 4, 9, 16])
[<matplotlib.lines.Line2D object at 0x04541B70>]
>>> pyplot.show()
```

Figure 18

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

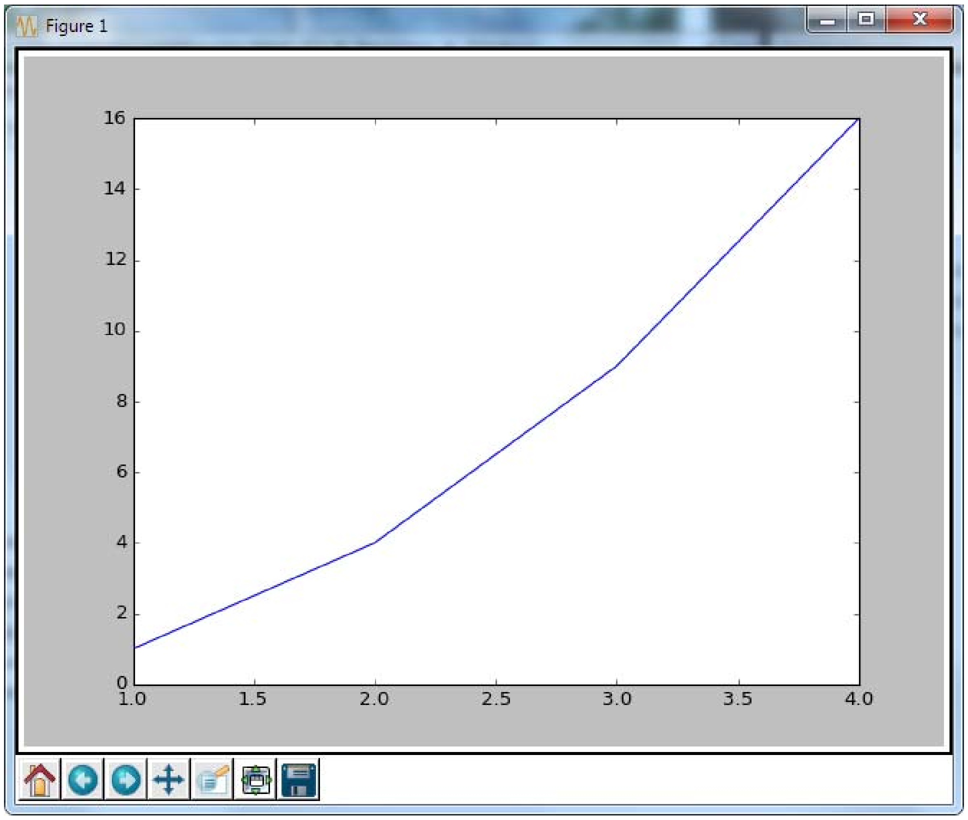


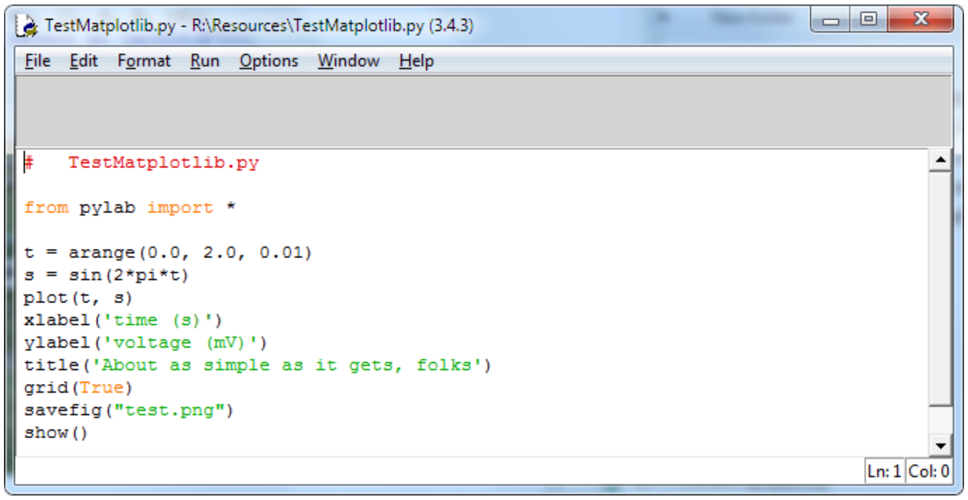
Figure 19

To close this window, click on the “close” button in the upper right.

For a more interesting test, download the following file:–

TestMatplotlib.py

Use the *File* menu in the *IDLE* window to open this file, which resembles the following:–


TestMatplotlib.py - R:\Resources\TestMatplotlib.py (3.4.3)
File Edit Format Run Options Window Help
Ln: 1 Col: 0

```
#   TestMatplotlib.py

from pylab import *

t = arange(0.0, 2.0, 0.01)
s = sin(2*pi*t)
plot(t, s)
xlabel('time (s)')
ylabel('voltage (mV)')
title('About as simple as it gets, folks')
grid(True)
savefig("test.png")
show()
```

Figure 20

Click the *Run* > *Run Module* command in the menu at the top of the window to produce the following window:–

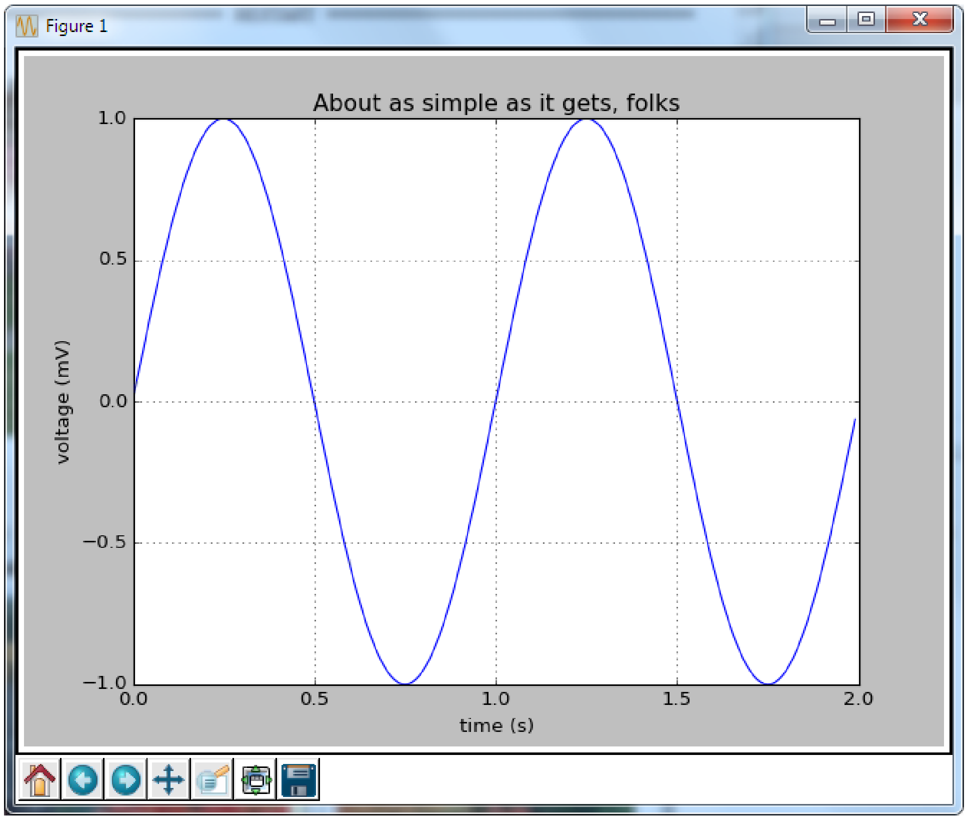


Figure 21

Congratulations! You now have a working version of *matplotlib* installed.

Note: Be sure to conduct these tests early in the term. There won't be enough time to discover problems and fix them when a homework assignment is due the next day.