Python for R Users

Shelley L. Knuth
Research Computing
University of Colorado-Boulder

Outline

- Introduction to Python
- Advantages and disadvantages
- Using Python
- Important Python packages
- Comparison of common programming commands and issues between Python and R

- Purpose of Talk
  - Geared toward researchers who readily use R
  - Some issue is pushing you away from R
  - Little to no knowledge of Python

Introduction to Python – What is it?

- Python is a programming language created in 1990 by Guido van Rossum
- Named for Monty Python
- Designed to be easy to use, learn, and understand
- Generalized programming language
  - No specific discipline use
- Open source – free!!!
- Cross-platform
- “Glue” language
  - You can call other programming language functions within Python

How do I get Python?

- If you have a Mac or a Linux system, you might already have it
  - Type **python** or **which python** at the command line to see if you do
- If you don't, I recommend installing anaconda http://continuum.io/downloads
- Anaconda is a package manager that makes it easier to get everything you need for Python
- Once downloaded, double click, follow the prompts

Introduction to Python

- Python itself is an official programming language
- The general python includes the programming language and interpreter
- Standard library
- Need additional packages to plot, to do scientific computing
- Also if want a user interface will need to pick one

Advantages and Disadvantages

- R – Advantages
  - Free!!
  - Written by statisticians, for statisticians
  - Active community generating many statistics packages
  - Lots of online support
    - Maybe more so than Python for data science
- R – Disadvantages
  - Written by statisticians, for statisticians
  - Can be complex to learn
  - Not really a programming language
    - Has a language but isn't a language
    - If it doesn't look like a traditional language can be more difficult to figure it out

Advantages and Disadvantages

- Python – Advantages
  - Free!!!!!
  - Widely used by people in all careers
  - Easy to read
  - Powerful language
- Python – Disadvantages
  - Not as nicely packaged
  - Science has been slow to catch up to using python in classes
  - Have to import libraries/packages

http://blogs.lt.vt.edu/safetyinnumbers/2014/04/23/technical-computing-wars-matlab-vs-python/

Python Context

- Python is used by scientists, non-scientists, students, non-students…everyone
- Lots of development in recent years from community
- Python along with it's vast number of libraries are its appeal
- Most recent version: 3.4.2
- Python 3 is not compatible with Python 2
- A lot of code out there is written for Python 2.7

Python and R

Let’s compare the two and see how easy it can be to transition to Python from R

Python vs. R - General Syntax

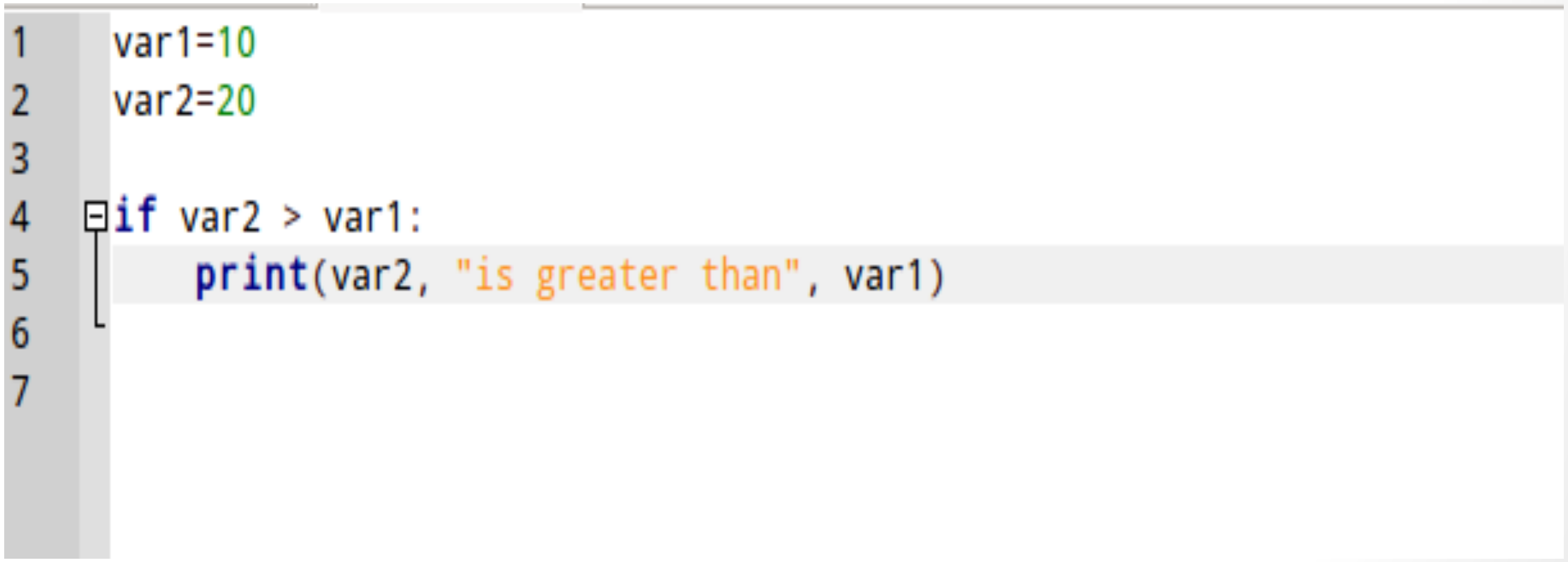
	R	Python
Element index	1	0
Comment	#	#
Print variable contents to screen	print(x)	print(x)
Print string	“Hello Everyone!”	print “hello Everyone!”
Find help on a function	help(sum)	help(func)
Script file extension	.R	.py
Import library functions	R CMD INSTALL pkg –l /dir/ Or Install.packages(“pkg”, lib=/dir/	from func import *
Random numbers	sample	Import numpy (see later) np.random.rand(row,col)
Line continuation	none	\

Number Types and Math

- In R, 21/3=7 and 23/3=7.667
- In Python, 21/3=7 and 23/3=7
- In Python you must specify the type of number or it will simply output an integer
- To get the correct answer you should type:
  - 21.0/3.0, which =7.0, and 23.0/3., which=7.6666

Syntax in python

In Python, there are no brackets or semicolons. Instead, each command is interpreted as its own “block” by indentation:



```
var1=10
var2=20

if var2 > var1:
    print(var2, "is greater than", var1)
```

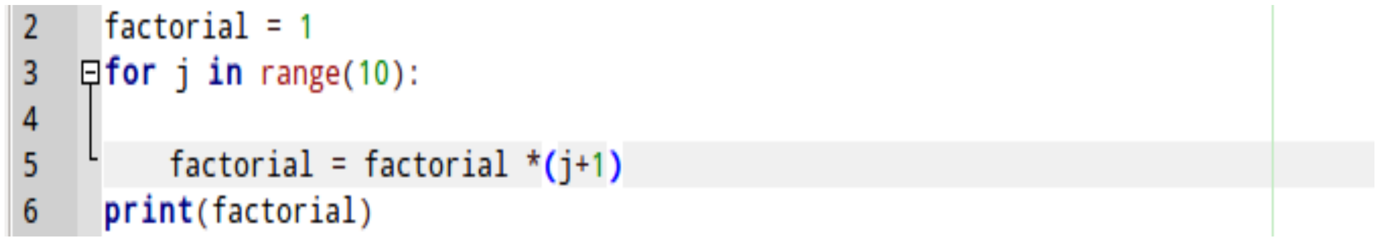
If/else statements

- If/else statements, unlike in R, don’t end
- The end is where the indentation, or block, ends

```
if (5 > 4):
        print(“The world is still sane”)
elif (5=5):
        print(“Even now it’s still sane”)
else:
        print(“You have entered another dimension”)
```

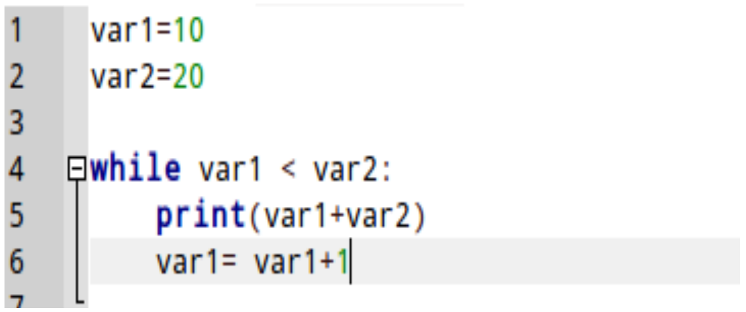
Loops

- The same is true for for and while loops:



```python
factorial = 1
for j in range(10):

    factorial = factorial *(j+1)
print(factorial)
```

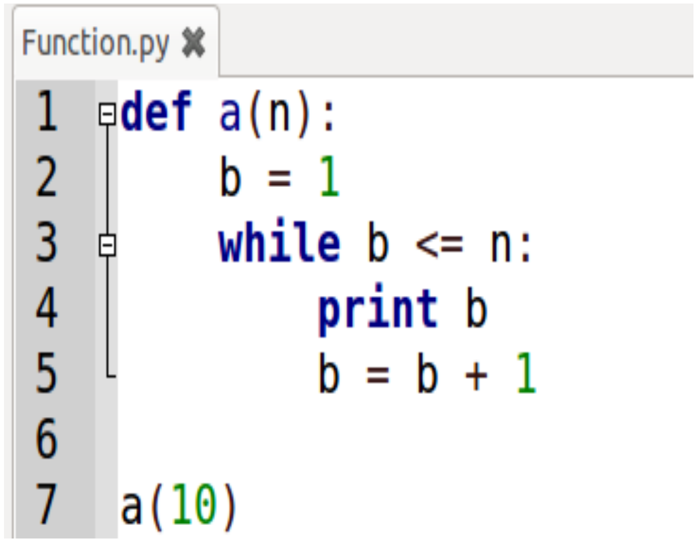


```python
var1=10
var2=20

while var1 < var2:
    print(var1+var2)
    var1= var1+1
```

Functions

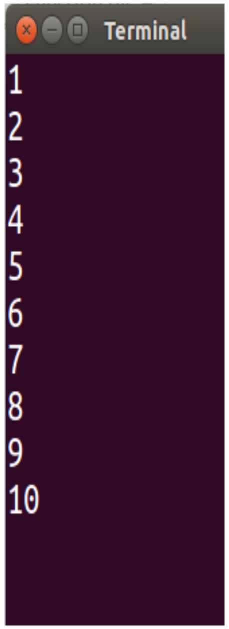
- Functions are defined using "def"



Function.py

```python
def a(n):
    b = 1
    while b <= n:
        print b
        b = b + 1

a(10)
```



Terminal

```
1
2
3
4
5
6
7
8
9
10
```

Suppose you wanted to list numbers from 1 to 10 but another time, you want to list them from 1 to 20. Instead of writing the same code twice, write a function that you can call.
In this example it's listing all numbers from 1 to 10. To list all numbers from 1 to 20, you would just call: a(20)

Calling Values in an Array

- If you have a 2D array, can call values such as:

```
test[0,]
test[1,]
```

- Remember – Python starts out at 0 index

Writing a script in Python

- Open a text window
- Type commands
- Save file to <filename>.py
- Let's try it!!

- In a text window, type the following:

```
#Our test program
print "I like test programs"
print "They are fun"
print "This is great!"
```

Then save file to test.py

Running a Script in Python

- First, you need to select your IDE or whether you'll run off the command line
- Options:
  - Type **python** at the command line
  - Type **ipython** at the command line
  - Type **IDLE** at the command line
- I use ipython typically

To Run the script…

- Depending on what you're using for your IDE it might be different for running the script
- For python or ipython type **`import test`** and your program should run

Important Python Packages

- Python is nothing without its libraries
- Many of them created and modified by the community
- Here are some additional python packages you will need to get to do any kind of scientific computing
  - Numpy – Allows you do to matrices easier/faster
  - SciPy – Allows you to manipulate the matrices
  - Pandas – Can go beyond SciPy
  - Matplotlib – graphing
  - Ipython – interactive computing environment

NumPy and SciPy

- R is useful in manipulating matrices
- Python itself cannot do that very well; very bare bones
- However, the libraries numPy and sciPy were written to make scientific computing easy
  - Provide common mathematical and numerical routines as part of functions within the libraries
- NumPy: provides basic routines for manipulating large arrays and matrices
  - Different in Python from R in that numpy doesn't differentiate between vectors and matrices
- SciPy: extends NumPy's functionality with Fourier transformation, regression, etc
- Should install both

Pandas

- Pandas built on top of R
- Pandas behaves similarly to an R data frame
- Can hold heterogeneous data
- Can label with column names and row indices
- Account for missing values
- Different from R in that it uses an object oriented interface
- Easily add/delete columns
- Merge datasets
- Time series functionality

http://www.datarobot.com/blog/introduction-to-python-for-statistical-learning/

Using NumPy, SciPy, and Pandas

- How do I get it?
  - If you’ve installed Python using Anaconda it comes with it
- How do I use it?
  - When you start up Python, you are using basic Python and whatever libraries you have imported
  - To import these libraries, at the top of your script, or on the command line, type:

```
import numpy
import scipy
import pandas
```

(continued on next slide!!)

Using NumPy, SciPy, and Pandas

- If you are using a large number of calls, however, it’s better to import the library under some shorter name so that you can access numPy, SciPy, and pandas objects

- Instead, type the following:

```
import numpy as np
import scipy as sp
import pandas as pd
```

Then you would use it in ways such as:

```
np.array([1., 2., 3., 4.])
```

Rpy2

- One of the nice things about Python is the ability to call other languages from Python
- Rpy2 allows you to call R from Python
  - Allow you to, for example, use a statistics package in Python from R

- Install:

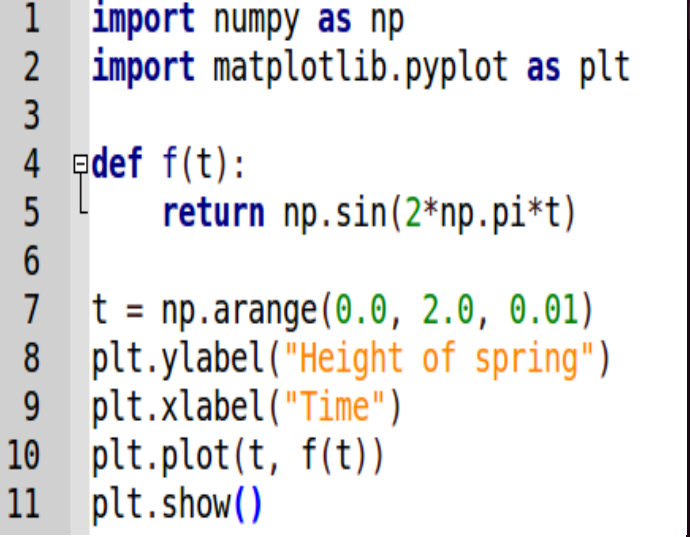
```
Pip install rpy2
```

- **More info:**

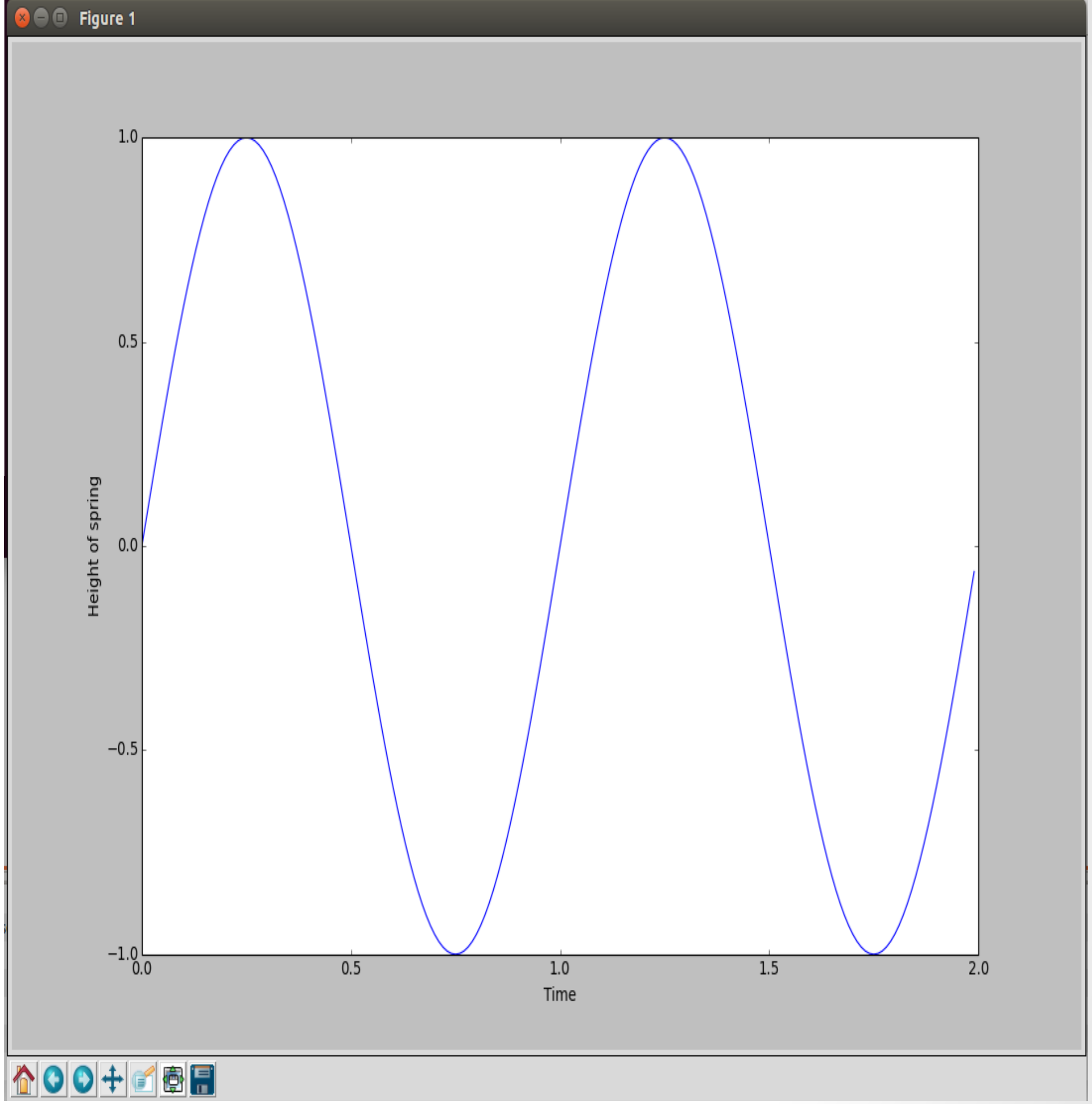
http://www.randalolson.com/2013/01/14/filling-in-pythons-gaps-in-statistics-packages-with-rmagic/

Matplotlib.pyplot

- Matplotlib is a library of functions that allows plotting
- The following example will be using matplotlib

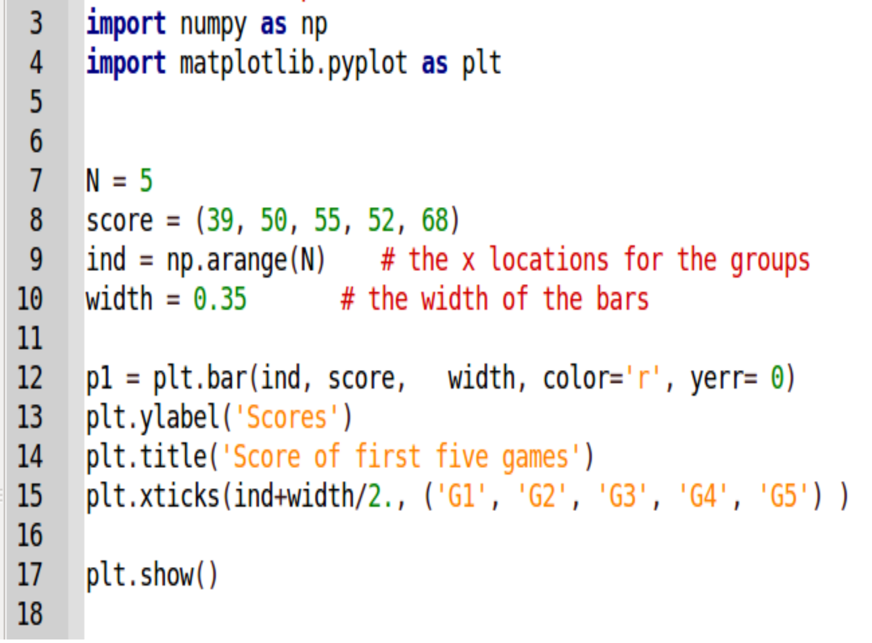


```
1   import numpy as np
2   import matplotlib.pyplot as plt
3
4   def f(t):
5       return np.sin(2*np.pi*t)
6
7   t = np.arange(0.0, 2.0, 0.01)
8   plt.ylabel("Height of spring")
9   plt.xlabel("Time")
10  plt.plot(t, f(t))
11  plt.show()
```

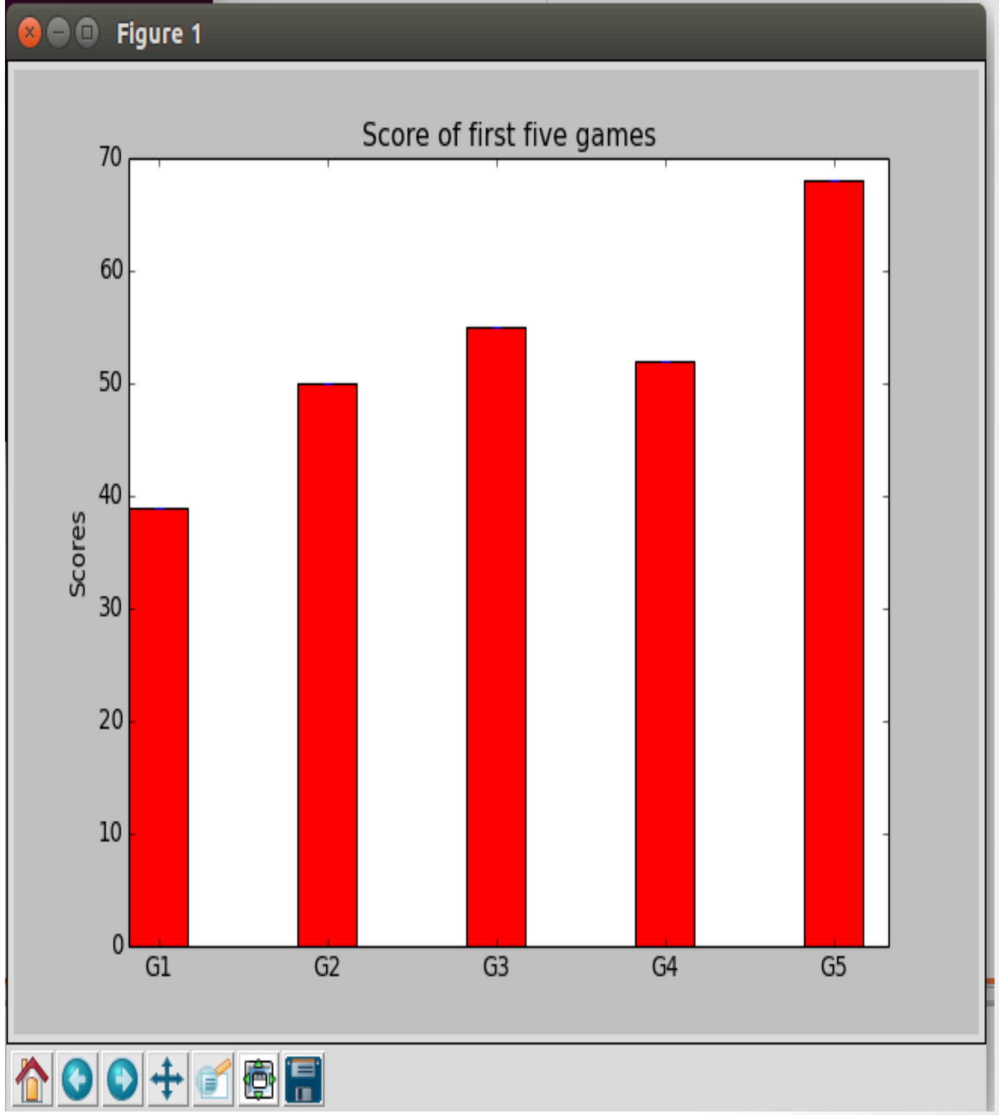


Stepping through line by line:

1. Library for math functions
2. Library for graphing
4. Create function f(t) which creates a sin wave
7. Make the x values go from 0 to 2, stepping by .01
8. Label the y-axis
9. Label the x-axis
10. Plot the graph of t and f(t)
11. Show graph on screen



```
3   import numpy as np
4   import matplotlib.pyplot as plt
5
6
7   N = 5
8   score = (39, 50, 55, 52, 68)
9   ind = np.arange(N)    # the x locations for the groups
10  width = 0.35       # the width of the bars
11
12  p1 = plt.bar(ind, score,   width, color='r', yerr= 0)
13  plt.ylabel('Scores')
14  plt.title('Score of first five games')
15  plt.xticks(ind+width/2., ('G1', 'G2', 'G3', 'G4', 'G5') )
16
17  plt.show()
18
```



Slightly different than the previous code, we utilize the function <u>plt.bar().</u>

Stepping through the code line by line:

3. Library for math functions
4. Library for graphing
7. Number of bars
8. Values of each bar
9. How far apart the bars are
10. Width of bars
12. Plotting the bars
13. Label y-axis
14. Title of graph
15. X-axis labels
17. Show graph on screen

Thanks for Attending!

- Useful documentation: docs.python.org

- Email: rc-help@colorado.edu
- Shelley.knuth@colorado.edu
- Twitter: @shelley_knuth

- Survey: http://goo.gl/forms/8VidcwOhRT

References

- https://wiki.python.org/moin/BeginnersGuide
- http://www.stat.washington.edu/~hoytak/blog/whypython.html
- http://www.sthurlow.com/python/
- http://www.engr.ucsb.edu/~shell/che210d/numpy.pdf
- www.matplotlib.org
- www.python.org