Lec 15. Modules in Python

Python has a large library of modules and packages that extends its functionality.

- The `import` statement makes reference to the modules that are Python files containing definitions and statements.
- If you write `import <module>` in a program, the Python interpreter executes the statements in the file `<module>.py` and enters the module name into the current namespace, so that the functions/procedures it defines are available with the "dotted syntax": `<module>.<function>`. Recall how you used `math` module:

  `import math` use `math.sqrt(2), math.pi`

  `from math import *` *What does this do?*

- You can also define your own module by placing a code within a file *module_name.py* somewhere in your computer (for small projects, usually in the same directory where you run your code).

  Note: DO NOT name your *module* anything that isn't a valid Python identifier (hyphen, starting with a digit, name of a built-in functions, modules, etc.)

Revisit the Bisection code

Let us organize the Bisection code (from Lab 4) using your own modules.

- First, download the bisection code (cw7.py) from your course Canvas. Put it in a new directory CW7 and make three copies of the file.

```
$ cp cw7.py bisect.py
$ cp cw7.py funct.py
$ cp cw7.py main.py
```

- In the main.py delete line 1 to 52.

```
$ vi main.py
:8,52d
```

- In the funct.py delete line 17 to end of file.

```
$ vi funct.py
:17,$d
```

- In the bisect.py delete line 8 to 17 and line 53 to end of the file.

```
$ vi bisect.py
:8,17d
:53,$d
```

- In main.py insert the following lines after the second line.

```
from funct import F
from bisect import bisection
```

- Run your code:

```
$ ./main.py
```

make sure the file has executable permission!

Revisit the Bisection code

Also try:

1. Make a copy of main.py to cw7a.py and make the following change.

   ```
   import funct as mf
   from bisect import bisection as bb
   ```

   and run your new code. What happened? How can you make it run?

2. In funct.py write the something inside the *doc_string*. For example,

   ```
   """ This is my module to keep functions:
   F(x)=x**3+x**2-3.0*x-3.0, g(x)=1-2*x """
   ```

   and in cw7a.py write the following line and run it.

   ```
   print(mf.__name__)
   print(mf.__doc__)
   print(bb.__name__)
   print(bb.__doc__)
   ```

   What did you see?

- In cw7a.py replace `if __name__ == '__main__':` to `main():` and write `main()` in the last line and run your code. What happens?

Lec 16. Introduction to NumPy

NumPy is the fundamental package for scientific computing with Python and is the base for many other packages.

- NumPy has a powerful custom N-dimensional array object for efficient and convenient representation od data.
- NumPy has tools for integration with other programming languages used for scientific programming like C/C++ and Fortran.
- NumPy is used for mathematical operations like vectors and matrix operations needed in image processing, signal processing, machine learning, etc.
- Moreover, it will introduce you to a whole other set of libraries such as SciPy and Scikit-learn, which you can use to solve almost any problem.

Introduction to vectors and matrices

- Addition and Multiplication by a scalar (by components)
- Dot product
- Matrix Vector product
- Matrix Matrix product
- Identity Matrix, Inverse Matrix, Linear System

$$AI = IA = A; \quad AA^{-1} = A^{-1}A = I; \quad Ax = b \implies x = A^{-1}b$$

NumPy

Reading and writing an array to a file

-

NumPy

Some Statistics

-