Chapter1-Preliminaries

March 28, 2021

1 Chapter Preliminaries

Scientific Computation (MKP3303)

> R.U.Gobithaasan (2021). Scientific Computing, Lectures for Undergraduate Degree Program B.Sc (Applied Mathematics), Faculty of Ocean Engineering Technology, University Malaysia Terengganu. https://sites.google.com/site/gobithaasan/LearnTeach

1. Needed Modules
2. Floating point numbers
3. Complex numbers
4. Expressions, assignment statements, equalities,
5. Functions (Python & Mathematics)
6. Overflow error, underflow error and rounding-off error
7. Developing your own module

References: - w3schools Online Materials - SciPi Lecture Notes - Robert Johansson, Numerical Python: Scientific Computing and Data Science Applications with Numpy, SciPy and Matplotlib (2019, Apress). - Donaldson Toby,Python: Visual QuickStart Guide (2008, Peachpit Press). - Tony Gaddis-Starting Out with Python,(2018,Global Edition-Pearson Education) - Robert Johansson August, Introduction to Scientific Computing in Python Continuum Analytics, (2015)

```
$ jupyter nbconvert --to html NOTEBOOK-NAME.ipynb
```

2 Using Jupyter Notebook

some shortcuts:

- b: Create a new cell below the currently selected cell.
- a: Create a new cell above the currently selected cell.
- d-d Delete the currently selected cell.
- 1 to 6: Heading cell of level 1 to 6.
- x: Cut currently selected cell.
- c: Copy currently selected cell.
- v: Paste cell from the clipboard.
- m: Convert a cell to a markdown cell.

- y: Convert a cell to a code cell.
- Up: Select previous cell.
- Down: Select next cell.
- Enter: Enter edit mode.
- Escape: Exit edit mode.
- Shift-Enter: Run the cell.
- h: Display a help window with a list of all available keyboard shortcuts.
- 0-0: Restart the kernel.
- i-i: Interrupt an executing cell.
- s: Save the notebook.

3 Markdown Cells

Summary of Markdown Syntax for Jupyter Notebook Markdown Cells

Fonts:

- italics: *text*
- bold: **text**
- stike-through: ~~text~~
- fixed-width: `text`
- url: URL text
- Vertatim(with tab):

```
def func(x):
    return x ** 2
```

- New paragraph: with an empty line.
- Types of headers:

4 Level 1 heading

4.1 Level 2 heading

4.1.1 Level 3 heading

- Block quote: > Text here is indented and offset > from the main text body.
- Unordered list (use - or *):
- Item one
- Item two
- Item three
- Ordered list:

1. Item one
2. Item two
3. Item three

Table:

A	B	C
1	2	3
4	5	6

- image: local machine

- image: internet

[ ]:
```
#![Alternative text](https://www.python.org/static/img/python-logo.png)
```

- Inline LateX equation:

$f2(x, y, z) = x^2 + y^3 + \sqrt{z}$

- Displayed LateX equation. See some examples at Latex Cookbook

$$\begin{aligned} y &= x^4 + 4 \\ &= (x^2 + 2)^2 - 4x^2 \\ &\leq (x^2 + 2)^2 \end{aligned} \tag{1}$$

5 Shift-Enter:Scientific Python Ecosystem

run command: > pip install *module*

where *module* is the name of the module you want to install. For example to install a module called **pandas**: > pip install pandas

Core numeric libraries - **Numpy**: numerical computing with powerful numerical arrays objects, and routines to manip- ulate them. Numpy

- **Scipy**: high-level numerical routines. Optimization, regression, interpolation, etc. Scipy
- **Matplotlib**: 2-D visualization, "publication-ready" plots Matplotlib

source: SciPi Lecture Notes

5.1 Needed Modules / Libraries

Pip install the modules you need. You may browse for modules at PyPI · The Python Package Index

[4]:
```
!python3 --version
!pip freeze |(grep 'matplotlib\|numpy\|jupyter\|scipy')
```

```
Python 3.8.3
jupyter==1.0.0
jupyter-client==6.1.3
jupyter-console==6.1.0
jupyter-core==4.6.3
jupyter-packaging==0.7.12
jupyter-server==1.5.1
jupyterlab==3.0.12
jupyterlab-server==2.3.0
jupyterthemes==0.20.0
matplotlib==3.2.2
numpy==1.19.4
scipy==1.5.2
```

5.1.1 Loaded Modules

[5]:
```
import sys
sys.modules.keys();
```

[19]:
```
print(dir())
```

```
['Image', 'In', 'Out', 'PythonStack', '_', '_1', '_16', '_7', '_8', '__', '___',
'__builtin__', '__builtins__', '__doc__', '__loader__', '__name__',
'__package__', '__spec__', '_dh', '_exit_code', '_i', '_i1', '_i10', '_i11',
'_i12', '_i13', '_i14', '_i15', '_i16', '_i17', '_i18', '_i19', '_i2', '_i3',
'_i4', '_i5', '_i6', '_i7', '_i8', '_i9', '_ih', '_ii', '_iii', '_oh',
'display', 'exit', 'get_ipython', 'importlib', 'quit', 'sys']
```

[20]:
```
import myModule
print(dir())
```

```
['Image', 'In', 'Out', 'PythonStack', '_', '_1', '_16', '_7', '_8', '__', '___',
'__builtin__', '__builtins__', '__doc__', '__loader__', '__name__',
'__package__', '__spec__', '_dh', '_exit_code', '_i', '_i1', '_i10', '_i11',
'_i12', '_i13', '_i14', '_i15', '_i16', '_i17', '_i18', '_i19', '_i2', '_i20',
'_i3', '_i4', '_i5', '_i6', '_i7', '_i8', '_i9', '_ih', '_ii', '_iii', '_oh',
'display', 'exit', 'get_ipython', 'importlib', 'myModule', 'quit', 'sys']
```

[21]:
```
import importlib
importlib.reload(myModule)
print(dir())
```

```
['Image', 'In', 'Out', 'PythonStack', '_', '_1', '_16', '_7', '_8', '__', '___',
'__builtin__', '__builtins__', '__doc__', '__loader__', '__name__',
'__package__', '__spec__', '_dh', '_exit_code', '_i', '_i1', '_i10', '_i11',
'_i12', '_i13', '_i14', '_i15', '_i16', '_i17', '_i18', '_i19', '_i2', '_i20',
'_i21', '_i3', '_i4', '_i5', '_i6', '_i7', '_i8', '_i9', '_ih', '_ii', '_iii',
'_oh', 'display', 'exit', 'get_ipython', 'importlib', 'myModule', 'quit', 'sys']
```

6 Documentation & Help files

[22]:
```
import math as m # importing a module
```

[ ]:
```
 m. #tab after the dot operator to see all available functions
```

[24]:
```
help(m.log)
```

```
Help on built-in function log in module math:

log(…)
    log(x, [base=math.e])
    Return the logarithm of x to the given base.

    If the base not specified, returns the natural logarithm (base e) of x.
```

[25]:
```
# import a math module
import math
```

[26]:
```
help(math.log10)
```

```
Help on built-in function log10 in module math:

log10(x, /)
    Return the base 10 logarithm of x.
```

[27]:
```
?math.log10 # (For jupyter notebook) prompt a new window to show the info
```

```
Object `math.log10 # (For jupyter notebook) prompt a new window to show the info
` not found.
```

[28]:
```
math.log10(8)
```

[28]: 0.9030899869919435

7 Input output Caching

[29]:
```
5+9
```

[29]: 14

[31]:
```
In[29] # previous input
```

[31]: '5+9'

[32]:
```
Out[29] # previous output
```

[32]: 14