ch01-Computing-python

October 23, 2024

1 Chapter 1: Computing with Python

Robert Johansson

Source code listings for Numerical Python - A Practical Techniques Approach for Industry (ISBN 978-1-484205-54-9).

The source code listings can be downloaded from http://www.apress.com/9781484205549

1.1 Interpreter

[ ]:
```
%%writefile hello.py
print("Hello from Python!")
```

[ ]:
```
!python hello.py
```

[ ]:
```
!python --version
```

1.2 Input and output caching

[ ]:
```
3 * 3
```

[ ]:
```
In[1]
```

[ ]:
```
Out[4]
```

[ ]:
```
In
```

[ ]:
```
Out
```

[ ]:
```
1+2
```

[ ]:
```
1+2;
```

[ ]:
```
x = 1
```

[ ]:
```
x = 2; x
```

1.3 Documentation

[ ]:
```
import os
```

[ ]:
```
# try os.w<TAB>
```

[ ]:
```
import math
```

[ ]:
```
math.cos?
```

1.4 Interaction with System Shell

[ ]:
```
!touch file1.py file2.py file3.py
```

[ ]:
```
!ls file*
```

[ ]:
```
files = !ls file*
```

[ ]:
```
len(files)
```

[ ]:
```
files
```

[ ]:
```
file = "file1.py"
```

[ ]:
```
!ls -l $file
```

1.5 Running scripts from the IPython console

[ ]:
```
%%writefile fib.py

def fib(N):
    """
    Return a list of the first N Fibonacci numbers.
    """
    f0, f1 = 0, 1
    f = [1] * N
    for n in range(1, N):
        f[n] = f0 + f1
        f0, f1 = f1, f[n]

    return f

print(fib(10))
```

[ ]:
```
!python fib.py
```

[ ]:
```
%run fib.py
```

[ ]:
```
fib(6)
```

1.6 Debugger

[ ]:
```
fib(1.0)
```

[ ]:
```
%debug
```

1.7 Timing and profiling code

[ ]:
```
%timeit fib(100)
```

[ ]:
```
result = %time fib(100)
```

[ ]:
```
len(result)
```

[ ]:
```
import numpy as np

def random_walker_max_distance(M, N):
    """
    Simulate N random walkers taking M steps, and return the largest distance
    from the starting point achieved by any of the random walkers.
    """
    trajectories = [np.random.randn(M).cumsum() for _ in range(N)]
    return np.max(np.abs(trajectories))
```

[ ]:
```
%prun random_walker_max_distance(400, 10000)
```

1.8 Jupyter notebook

[1]:
```
from IPython.display import display, Image, HTML, Math
```

[2]:
```
Image(url='http://python.org/images/python-logo.gif')
```

[2]:
```
<IPython.core.display.Image object>
```

[3]:
```
import scipy, numpy, matplotlib
modules = [numpy, matplotlib, scipy]
row = "<tr> <td>%s</td> <td>%s</td> </tr>"
rows = "\n".join([row % (module.__name__, module.__version__) for module in␣
 ↪modules])
s = "<table> <tr><th>Library</th><th>Version</th> </tr> %s</table>" % rows
```

[4]:
```
s
```

[4]:
```
'<table> <tr><th>Library</th><th>Version</th> </tr> <tr> <td>numpy</td>
<td>2.1.1</td> </tr>\n<tr> <td>matplotlib</td> <td>3.9.2</td> </tr>\n<tr>
```

```
<td>scipy</td> <td>1.14.1</td> </tr></table>'
```

[5]:
```
HTML(s)
```

[5]:
```
<IPython.core.display.HTML object>
```

[6]:
```
class HTMLDisplayer(object):
    def __init__(self, code):
        self.code = code

    def _repr_html_(self):
        return self.code
```

[7]:
```
HTMLDisplayer(s)
```

[7]:
```
<__main__.HTMLDisplayer at 0x16ec2b5b610>
```

[8]:
```
Math(r'\hat{H} = -\frac{1}{2}\epsilon \hat{\sigma}_z-\frac{1}{2}\delta␣
↪\hat{\sigma}_x')
```

[8]:
$$\hat{H} = -\frac{1}{2}\epsilon\hat{\sigma}_z - \frac{1}{2}\delta\hat{\sigma}_x$$

[9]:
```
class QubitHamiltonian(object):
    def __init__(self, epsilon, delta):
        self.epsilon = epsilon
        self.delta = delta

    def _repr_latex_(self):
        return "r$\hat{H} = -%.2f\hat{\sigma}_z-%.2f\hat{\sigma}_x$" % \
            (self.epsilon/2, self.delta/2)
```

[10]:
```
QubitHamiltonian(0.5, 0.25)
```

[10]:
$$\mathrm{r}\hat{H} = -0.25\hat{\sigma}_z - 0.12\hat{\sigma}_x$$

[11]:
```
import matplotlib.pyplot as plt
import numpy as np
from scipy import stats

def f(mu):
    X = stats.norm(loc=mu, scale=np.sqrt(mu))
    N = stats.poisson(mu)
    x = np.linspace(0, X.ppf(0.999))
    n = np.arange(0, x[-1])

    fig, ax = plt.subplots()
    ax.plot(x, X.pdf(x), color='black', lw=2, label="Normal($\mu=%d,␣
↪\sigma^2=%d$)" % (mu, mu))
```

```
    ax.bar(n, N.pmf(n), align='edge', label=r"Poisson($\lambda=%d$)" % mu)
    ax.set_ylim(0, X.pdf(x).max() * 1.25)
    ax.legend(loc=2, ncol=2)
    plt.close(fig)
    return fig
```

[12]:
```
from ipywidgets import interact
import ipywidgets as widgets
```

[13]:
```
interact(f, mu=widgets.FloatSlider(min=1.0, max=20.0, step=1.0));
```

```
interactive(children=(FloatSlider(value=1.0, description='mu', max=20.0, min=1.
↪0, step=1.0), Output()), _dom_c…
```

1.9 Jupyter nbconvert

[15]:
```
!jupyter nbconvert --to html ch01-Computing-python.ipynb
```

```
[NbConvertApp] Converting notebook ch01-Computing-python.ipynb to html
[NbConvertApp] WARNING | Alternative text is missing on 1 image(s).
[NbConvertApp] Writing 326771 bytes to ch01-Computing-python.html
```

[ ]:
```
!jupyter nbconvert --to pdf ch01-Computing-python.ipynb
```

[ ]:
```
%%writefile custom_template.tplx
((*- extends 'article.tplx' -*))

((* block title *)) \title{Document title} ((* endblock title *))
((* block author *)) \author{Author's Name} ((* endblock author *))
```

[ ]:
```
!ipython nbconvert ch01-code-listing.ipynb --to pdf --template custom_template.
↪tplx
```

[ ]:
```
!ipython nbconvert ch01-code-listing.ipynb --to python
```

2 Versions

[ ]:
```
%reload_ext version_information
%version_information numpy
```