ch01-code-listing

November 2, 2024

1 Chapter 1: Computing with Python

Robert Johansson

Source code listings for Numerical Python - Scientific Computing and Data Science Applications with Numpy, SciPy and Matplotlib (ISBN 979-8-8688-0412-0).

1.1 Interpreter

[1]:
```
%%writefile hello.py
print("Hello from Python!")
```

```
Overwriting hello.py
```

[2]:
```
!python hello.py
```

```
Hello from Python!
```

[3]:
```
!python --version
```

```
Python 3.10.12
```

1.2 Input and output caching

(restart the kernel for the same output and cell numbers)

[1]:
```
3 * 3
```

[1]: 9

[2]:
```
In[1]
```

[2]: '3 * 3'

[3]:
```
Out[1]
```

[3]: 9

[4]:
```
In
```

[4]:
```
['', '3 * 3', 'In[1]', 'Out[1]', 'In']
```

[5]:
```
Out
```

[5]:
```
{1: 9, 2: '3 * 3', 3: 9, 4: ['', '3 * 3', 'In[1]', 'Out[1]', 'In', 'Out']}
```

[6]:
```
1+2
```

[6]:
```
3
```

[7]:
```
1+2;
```

[8]:
```
x = 1
```

[9]:
```
x = 2; x
```

[9]:
```
2
```

1.3 Documentation

[10]:
```
import os
```

[11]:
```
# try os.w<TAB>
```

[12]:
```
import math
```

[13]:
```
math.cos?
```

```
Signature: math.cos(x, /)
Docstring: Return the cosine of x (measured in radians).
Type:      builtin_function_or_method
```

1.4 Interaction with System Shell

[14]:
```
!touch file1.py file2.py file3.py
```

[15]:
```
!ls file*
```

```
file1.py file2.py file3.py
```

[16]:
```
files = !ls file*
```

[17]:
```
len(files)
```

[17]:
```
3
```

[18]:
```
files
```

[18]:
```
['file1.py', 'file2.py', 'file3.py']
```

[19]:
```
file = "file1.py"
```

[20]:
```
!ls -l $file
```

```
-rw-------@ 1 rob  staff  0 Nov  2 17:58 file1.py
```

1.5 Running scripts from the IPython console

[21]:
```
%%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))
```

```
Overwriting fib.py
```

[22]:
```
!python fib.py
```

```
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
```

[23]:
```
%run fib.py
```

```
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
```

[24]:
```
fib(6)
```

[24]:
```
[1, 1, 2, 3, 5, 8]
```

1.6 Debugger

[25]:
```
fib(1.0)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[25], line 1
----> 1 fib(1.0)

File ~/OneDrive/Desktop/npbook-ed3/code/numerical-python-code-ed3-v2-github/fib
 ↪py:7, in fib(N)
      3 """
      4 Return a list of the first N Fibonacci numbers.
      5 """
      6 f0, f1 = 0, 1
----> 7 f = [1] * N
      8 for n in range(1, N):
      9     f[n] = f0 + f1

TypeError: can't multiply sequence by non-int of type 'float'
```

[ ]:
```
%debug
```

```
> /Users/rob/OneDrive/Desktop/npbook-ed3/code/numerical-python-code-
ed3-v2-github/fib.py(7)fib()
      5     """
      6     f0, f1 =
0, 1
----> 7     f =
[1] *
N
      8     for n in
range(1,
N):
      9         f[n]
= f0 + f1
```

1.7 Timing and profiling code

[27]:
```
%timeit fib(100)
```

```
11.5 µs ± 140 ns per loop (mean ± std. dev. of 7 runs, 100,000 loops each)
```

[28]:
```
result = %time fib(100)
```

```
CPU times: user 27 µs, sys: 1 µs, total: 28 µs
Wall time: 33.1 µs
```

[29]:
```
len(result)
```

[29]:
```
100
```

[30]:
```
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))
```

[31]:
```
%prun random_walker_max_distance(400, 10000)
```

```
         20011 function calls in 0.336 seconds

   Ordered by: internal time

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
    10000    0.189    0.000    0.189    0.000 {method 'randn' of 'numpy.random.
↪mtrand.RandomState' objects}
    10000    0.070    0.000    0.070    0.000 {method 'cumsum' of 'numpy.
↪ndarray' objects}
        1    0.047    0.047    0.330    0.330 2615584822.py:
↪3(random_walker_max_distance)
        1    0.020    0.020    0.279    0.279 2615584822.py:8(<listcomp>)
        1    0.007    0.007    0.336    0.336 <string>:1(<module>)
        1    0.004    0.004    0.004    0.004 {method 'reduce' of 'numpy.ufunc'␣
↪objects}
        1    0.000    0.000    0.336    0.336 {built-in method builtins.exec}
        1    0.000    0.000    0.004    0.004 fromnumeric.py:71(_wrapreduction)
        1    0.000    0.000    0.004    0.004 fromnumeric.py:2692(max)
        1    0.000    0.000    0.000    0.000 fromnumeric.py:72(<dictcomp>)
        1    0.000    0.000    0.000    0.000 fromnumeric.py:
↪2687(_max_dispatcher)
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.
↪Profiler' objects}
        1    0.000    0.000    0.000    0.000 {method 'items' of 'dict' objects}
```

1.8 Jupyter notebook

[32]:
```
from IPython.display import display, Image, HTML, Math
```

[33]:
```
Image(url='http://python.org/images/python-logo.gif')
```

[33]:
```
<IPython.core.display.Image object>
```

[34]:
```
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
```

[35]:
```
s
```

[35]:
```
'<table> <tr><th>Library</th><th>Version</th> </tr> <tr> <td>numpy</td>
<td>1.25.2</td> </tr>\n<tr> <td>matplotlib</td> <td>3.8.0</td> </tr>\n<tr>
<td>scipy</td> <td>1.11.3</td> </tr></table>'
```

[36]:
```
HTML(s)
```

[36]:
```
<IPython.core.display.HTML object>
```

[37]:
```
class HTMLDisplayer(object):
    def __init__(self, code):
        self.code = code

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

[38]:
```
HTMLDisplayer(s)
```

[38]:
```
<__main__.HTMLDisplayer at 0x7fb5a6f03d00>
```

[39]:
```
Math(r'\hat{H} = -\frac{1}{2}\epsilon \hat{\sigma}_z-\frac{1}{2}\delta␣
 ↪\hat{\sigma}_x')
```

[39]:
$$\hat{H} = -\frac{1}{2}\epsilon\hat{\sigma}_z - \frac{1}{2}\delta\hat{\sigma}_x$$

[40]:
```
class QubitHamiltonian(object):
    def __init__(self, epsilon, delta):
        self.epsilon = epsilon
        self.delta = delta

    def _repr_latex_(self):
```

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

[41]:
```
QubitHamiltonian(0.5, 0.25)
```

[41]:
$$\hat{H} = -0.25\hat{\sigma}_z - 0.12\hat{\sigma}_x$$

[42]:
```
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
```

[44]:
```
from ipywidgets import interact
import ipywidgets as widgets
```

[45]:
```
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

[2]:
```
!jupyter nbconvert --to html ch01-code-listing.ipynb
```

```
[NbConvertApp] Converting notebook ch01-code-listing.ipynb to html
[NbConvertApp] WARNING | Alternative text is missing on 1 image(s).
[NbConvertApp] Writing 341393 bytes to ch01-code-listing.html
```

[2]:
```
!jupyter nbconvert --to pdf ch01-code-listing.ipynb
```

```
[NbConvertApp] Converting notebook ch01-code-listing.ipynb to pdf
/Users/rob/miniconda3/envs/py3.10/lib/python3.10/site-
packages/nbconvert/utils/pandoc.py:51: RuntimeWarning: You are using an
```

```
unsupported version of pandoc (2.12).
Your version must be at least (2.14.2) but less than (4.0.0).
Refer to https://pandoc.org/installing.html.
Continuing with doubts…
  check_pandoc_version()
[NbConvertApp] Writing 61299 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-quiet']
[NbConvertApp] Running bibtex 1 time: ['bibtex', 'notebook']
[NbConvertApp] WARNING | bibtex had problems, most likely because there were no
citations
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 64876 bytes to ch01-code-listing.pdf
```

[8]:
```
%%writefile custom_template.tplx
((*- extends 'article.tplx' -*))

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

```
Overwriting custom_template.tplx
```

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

[11]:
```
!jupyter nbconvert ch01-code-listing.ipynb --to python
```

```
[NbConvertApp] Converting notebook ch01-code-listing.ipynb to python
[NbConvertApp] Writing 5142 bytes to ch01-code-listing.py
```

2 Versions

[1]:
```
%reload_ext version_information
%version_information numpy
```

[1]:

Software	Version
Python	3.10.13 64bit [Clang 14.0.6]
IPython	8.20.0
OS	macOS 10.15.7 x86_64 i386 64bit
numpy	1.24.3
Mon May 06 14:29:51 2024 JST	

[ ]:
```
```