ch01-intro

November 14, 2023

0.0.1 Chapter 1: Computing with Python

0.0.2 Overview: a typical Python-based scientific computing stack.

0.0.3 Resources:

- Intel MKL (Math Kernel Library)
- openBLAS
- ATLAS
- SciPy
- Python Numeric & Scientific topics

0.0.4 Interpreter

- The easist way to execute Python code: run the program directly.
- Use Jupyter magic command to write Python source file to disk:

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

Overwriting hello.py

- Use the ! system shell command (included in the Python Jupyter kernel) to interactively run Python with hello.py as its argument.

```
[2]: !python3 hello.py
```

Hello from Python!

```
[3]: !python3 --version
```

Python 3.11.4

0.0.5 Input and output caching

- Input & output history can be accessed using **In** (a list) & **Out** (a dictionary). Both can be indexed with a cell number.

```
[4]: 3*3
     In[1]
```

```
[4]: 'get_ipython().run_cell_magic(\'writefile\', \'hello.py\', \'print("Hello from
     Python!")\\n\')'
```

- A single underscore = the most recent output;
- A double underscore = the *next* most recent output.

```
[5]: 1+1
```

```
[5]: 2
```

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

```
[6]: 4
```

```
[7]: _, __
```

```
[7]: (4, 2)
```

```
[8]: # In = a list
     In
```

```
[8]: ['',
      'get_ipython().run_cell_magic(\'writefile\', \'hello.py\', \'print("Hello from
     Python!")\\n\')',
      "get_ipython().system('python3 hello.py')",
      "get_ipython().system('python3 --version')",
      '3*3\nIn[1]',
```

```
        '1+1',
        '2+2',
        '_, __',
        '# In = a list\nIn']
```

[9]:
```
# Out = a dictionary
Out
```

[9]:
```
{4: 'get_ipython().run_cell_magic(\'writefile\', \'hello.py\', \'print("Hello
from Python!")\\n\')',
 5: 2,
 6: 4,
 7: (4, 2),
 8: ['',
  'get_ipython().run_cell_magic(\'writefile\', \'hello.py\', \'print("Hello from
Python!")\\n\')',
  "get_ipython().system('python3 hello.py')",
  "get_ipython().system('python3 --version')",
  '3*3\nIn[1]',
  '1+1',
  '2+2',
  '_, __',
  '# In = a list\nIn',
  '# Out = a dictionary\nOut']}
```
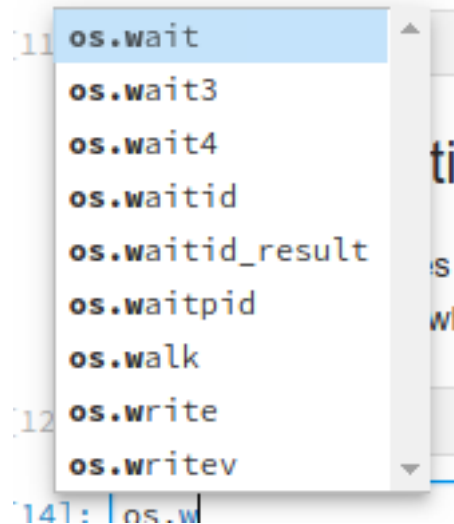
[10]:
```
# Suppress output results by ending statement with a semicolon
1+2;
```

0.0.6 Autocompletion

- The **Tab** key activates autocompletion (displays list of symbol names that are valid completions of what has been typed thus far.)

[11]:
```
import os
```

- Results of typing "os.w", followed by ⌢

0.0.7 Documentation

- "Docstrings" provide a built-in reference manual for most Python modules. Display the docstring by appending a Python object with "?".

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

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

0.0.8 Interaction with System Shell

- Anything after ! is evaluated using the system shell, such as bash.
- (I use Ubuntu Linux as my laptop OS. Your Windows equivalents will vary.)

```
[13]: !touch file1.py file2.py file3.py
      !ls file*
```

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

```
[14]: # output of a system shell command can be captured in a Python variable
      files = !ls file*
      print(len(files))
      print(files)
```

```
3
['file1.py', 'file2.py', 'file3.py']
```

```
[15]: # pass Python variable values to shell commands
      # by prefixing the variable name with $.
      file = "file1.py"
      !ls -l $file
```

```
-rw-rw-r-- 1 bjpcjp bjpcjp 0 Nov 14 08:18 file1.py
```

0.0.9   IPython Extensions

- Commands start with one or two "%" characters. A single % is used for single-line commands; dual %% is used for cells (multiple lines).

- %lsmagic returns a list of available commands.

```
[16]: %lsmagic
```

```
[16]: Available line magics:
      %alias  %alias_magic  %autoawait  %autocall  %automagic  %autosave  %bookmark
      %cat  %cd  %clear  %code_wrap  %colors  %conda  %config  %connect_info  %cp
      %debug  %dhist  %dirs  %doctest_mode  %ed  %edit  %env  %gui  %hist  %history
      %killbgscripts  %ldir  %less  %lf  %lk  %ll  %load  %load_ext  %loadpy  %logoff
      %logon  %logstart  %logstate  %logstop  %ls  %lsmagic  %lx  %macro  %magic
      %mamba  %man  %matplotlib  %micromamba  %mkdir  %more  %mv  %notebook  %page
      %pastebin  %pdb  %pdef  %pdoc  %pfile  %pinfo  %pinfo2  %pip  %popd  %pprint
      %precision  %prun  %psearch  %psource  %pushd  %pwd  %pycat  %pylab  %qtconsole
      %quickref  %recall  %rehashx  %reload_ext  %rep  %rerun  %reset
      %reset_selective  %rm  %rmdir  %run  %save  %sc  %set_env  %store  %sx  %system
      %tb  %time  %timeit  %unalias  %unload_ext  %who  %who_ls  %whos  %xdel  %xmode

      Available cell magics:
      %%!  %%HTML  %%SVG  %%bash  %%capture  %%code_wrap  %%debug  %%file  %%html
      %%javascript  %%js  %%latex  %%markdown  %%perl  %%prun  %%pypy  %%python
      %%python2  %%python3  %%ruby  %%script  %%sh  %%svg  %%sx  %%system  %%time
      %%timeit  %%writefile

      Automagic is ON, % prefix IS NOT needed for line magics.
```

0.0.10   Running scripts

- %run executes an external Python source file within an interactive IPython session.

```
[17]: %%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

[18]: `!python3 fib.py`

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

[19]: `%run fib.py`

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

[20]: `fib(6)`

[20]: [1, 1, 2, 3, 5, 8]

0.1 Listing all defined symbols

- %who lists all defined symbols
- %whos provides more detailed info.

[21]: `%who`

```
fib       file     files    math     os
```

[22]: `%whos`

```
Variable   Type        Data/Info
------------------------------
fib        function    <function fib at 0x7f39400d40e0>
file       str         file1.py
files      SList       ['file1.py', 'file2.py', 'file3.py']
math       module      <module 'math' (built-in)>
os         module      <module 'os' (frozen)>
```

0.2 Debugger

- Use %debug to step directly into the Python debugger.

[23]:
```python
# fib function fails - can't use floating point numbers.
try:
    fib(1.0)
except TypeError:
    print("nope. can't do that.")
```

nope. can't do that.

[24]:
```python
#%debug
```

0.2.1 Resetting the Python namespace

- Ensures a program is run in a pristine environment, uncluttered by existing variables and functions. Although it is necessary to reimport modules, it is important to know that even if the modules have changed since the last import, a new import after a %reset will not import the new module but rather reenable a cached version from the previous import.

- When developing Python modules, this is usually not the desired behavior. In that case, a reimport of a previously imported (and since updated) module can often be achieved by using the reload function from `IPython.lib.deepreload`. However, this method does not always work, as some libraries run code at import time that is only intended to run once. In this case, the only option might be to terminate and restart the IPython interpreter.

```
[26]: %reset
```

Once deleted, variables cannot be recovered. Proceed (y/[n])?

Nothing done.

0.3 Timing and profiling code

- `%timeit` and `%time` provide simple benchmarking utilities.

```
[27]: # first, re-define fibonacci code used above.
      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
```

```
[28]: # timeit does not return expression's resulting value.
      %timeit fib(50)
```

1.38 µs ± 12.2 ns per loop (mean ± std. dev. of 7 runs, 1,000,000 loops each)

```
[29]: # %time only runs once. less accurate estimate.
      result = %time fib(100)
```

CPU times: user 8 µs, sys: 0 ns, total: 8 µs
Wall time: 8.82 µs

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

```
[30]: 100
```

- The **cProfile** module provides **%prun** (for statements) and **%run** (for external scripts) profiling commands.

```
[31]: import numpy as np

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

```
[32]: # returns call counts, runtime & cume runtime for each function.
      %prun random_walker_max_distance(400, 10000)
```

```
        20013 function calls in 0.100 seconds

   Ordered by: internal time

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
    10000    0.058    0.000    0.058    0.000 {method 'randn' of 'numpy.random.
 ↪mtrand.RandomState' objects}
    10000    0.022    0.000    0.022    0.000 {method 'cumsum' of 'numpy.
 ↪ndarray' objects}
        1    0.013    0.013    0.099    0.099 220825864.py:
 ↪3(random_walker_max_distance)
        1    0.003    0.003    0.084    0.084 220825864.py:8(<listcomp>)
        1    0.002    0.002    0.002    0.002 {method 'reduce' of 'numpy.ufunc'␣
 ↪objects}
        1    0.001    0.001    0.100    0.100 <string>:1(<module>)
        1    0.000    0.000    0.100    0.100 {built-in method builtins.exec}
        1    0.000    0.000    0.002    0.002 fromnumeric.py:69(_wrapreduction)
        1    0.000    0.000    0.002    0.002 <__array_function__ internals>:
 ↪177(amax)
        1    0.000    0.000    0.002    0.002 fromnumeric.py:2703(amax)
        1    0.000    0.000    0.002    0.002 {built-in method numpy.core.
 ↪_multiarray_umath.implement_array_function}
        1    0.000    0.000    0.000    0.000 fromnumeric.py:70(<dictcomp>)
        1    0.000    0.000    0.000    0.000 fromnumeric.py:
 ↪2698(_amax_dispatcher)
        1    0.000    0.000    0.000    0.000 {method 'items' of 'dict' objects}
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.
 ↪Profiler' objects}
```

0.3.1 Jupyter: External image rendering

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

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

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

0.3.2 Jupyter: HTML rendering

```
[35]: 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])
      table = "<table><tr><th>Library</th><th>Version</th></tr> %s </table>" % rows
```

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

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

```
[37]: # another method
      class HTMLdisplayer(object):
          def __init__(self,code):
              self.code = code
          def _repr_html_(self):
              return self.code

      HTMLdisplayer(table)
```

```
[37]: <__main__.HTMLdisplayer at 0x7f3940100fd0>
```

0.3.3 Jupyter: Formula rendering using Latex

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

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

0.3.4 Jupyter: UI Widgets

```
[39]: 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
```

[40]:
```
from ipywidgets import interact
import ipywidgets as widgets
```

[42]:
```
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…
```

0.3.5 nbconvert to HTML

[43]:
```
!jupyter nbconvert --to html ch01-intro.ipynb
```

```
[NbConvertApp] Converting notebook ch01-intro.ipynb to html
[NbConvertApp] WARNING | Alternative text is missing on 1 image(s).
[NbConvertApp] Writing 348064 bytes to ch01-intro.html
```

0.3.6 nbconvert to PDF

- Requires a LaTeX environment to be installed.
- On this system (Ubuntu Linux): sudo apt-get install texlive-xetex

[44]:
```
!jupyter nbconvert --to pdf ch01-intro.ipynb;
```

```
[NbConvertApp] Converting notebook ch01-intro.ipynb to pdf
[NbConvertApp] ERROR | Error while converting 'ch01-intro.ipynb'
Traceback (most recent call last):
  File "/home/bjpcjp/.local/lib/python3.11/site-
packages/nbconvert/nbconvertapp.py", line 479, in export_single_notebook
    output, resources = self.exporter.from_filename(
                        ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

  File "/home/bjpcjp/.local/lib/python3.11/site-
packages/nbconvert/exporters/templateexporter.py", line 389, in from_filename
    return super().from_filename(filename, resources, **kw)  #
type:ignore[return-value]
```

```
                   ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/lib/python3.11/site-
packages/nbconvert/exporters/exporter.py", line 201, in from_filename
    return self.from_file(f, resources=resources, **kw)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/lib/python3.11/site-
packages/nbconvert/exporters/templateexporter.py", line 395, in from_file
    return super().from_file(file_stream, resources, **kw)  #
type:ignore[return-value]
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/lib/python3.11/site-
packages/nbconvert/exporters/exporter.py", line 220, in from_file
    return self.from_notebook_node(
           ^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/lib/python3.11/site-
packages/nbconvert/exporters/pdf.py", line 187, in from_notebook_node
    latex, resources = super().from_notebook_node(nb, resources=resources, **kw)
                       ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/lib/python3.11/site-
packages/nbconvert/exporters/latex.py", line 92, in from_notebook_node
    return super().from_notebook_node(nb, resources, **kw)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/lib/python3.11/site-
packages/nbconvert/exporters/templateexporter.py", line 427, in
from_notebook_node
    output = self.template.render(nb=nb_copy, resources=resources)
             ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/usr/lib/python3/dist-packages/jinja2/environment.py", line 1301, in
render
    self.environment.handle_exception()
  File "/usr/lib/python3/dist-packages/jinja2/environment.py", line 936, in
handle_exception
    raise rewrite_traceback_stack(source=source)
  File
"/home/bjpcjp/.local/share/jupyter/nbconvert/templates/latex/index.tex.j2", line
8, in top-level template code
    ((* extends cell_style *))
    ^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/share/jupyter/nbconvert/templates/latex/style_jupyte
r.tex.j2", line 176, in top-level template code
    \prompt{(((prompt)))}{(((prompt_color)))}{(((execution_count)))}{(((extra_sp
ace)))}
    ^^^^^^^^^^^^^^^^^^^^^^^^^
  File
"/home/bjpcjp/.local/share/jupyter/nbconvert/templates/latex/base.tex.j2", line
7, in top-level template code
    ((*- extends 'document_contents.tex.j2' -*))
    ^^^^^^^^^^^^^^^^^^^^^^^^^
```

11

```
File "/home/bjpcjp/.local/share/jupyter/nbconvert/templates/latex/document_con
tents.tex.j2", line 51, in top-level template code
    ((*- block figure scoped -*))
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/share/jupyter/nbconvert/templates/latex/display_prio
rity.j2", line 5, in top-level template code
    ((*- extends 'null.j2' -*))
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/share/jupyter/nbconvert/templates/latex/null.j2",
line 30, in top-level template code
    ((*- block body -*))
  File
"/home/bjpcjp/.local/share/jupyter/nbconvert/templates/latex/base.tex.j2", line
220, in block 'body'
    ((( super() )))
  File "/home/bjpcjp/.local/share/jupyter/nbconvert/templates/latex/null.j2",
line 32, in block 'body'
    ((*- block any_cell scoped -*))
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/share/jupyter/nbconvert/templates/latex/null.j2",
line 85, in block 'any_cell'
    ((*- block markdowncell scoped-*)) ((*- endblock markdowncell -*))
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/share/jupyter/nbconvert/templates/latex/document_con
tents.tex.j2", line 68, in block 'markdowncell'
    ((( cell.source | citation2latex | strip_files_prefix |
convert_pandoc('markdown+tex_math_double_backslash', 'json',extra_args=[]) |
resolve_references | convert_explicitly_relative_paths |
convert_pandoc('json','latex'))))
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/lib/python3.11/site-
packages/nbconvert/filters/pandoc.py", line 35, in convert_pandoc
    return pandoc(source, from_format, to_format, extra_args=extra_args)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/lib/python3.11/site-
packages/nbconvert/utils/pandoc.py", line 51, in pandoc
    check_pandoc_version()
  File "/home/bjpcjp/.local/lib/python3.11/site-
packages/nbconvert/utils/pandoc.py", line 99, in check_pandoc_version
    v = get_pandoc_version()
        ^^^^^^^^^^^^^^^^^^^^^
  File "/home/bjpcjp/.local/lib/python3.11/site-
packages/nbconvert/utils/pandoc.py", line 76, in get_pandoc_version
    raise PandocMissing()
nbconvert.utils.pandoc.PandocMissing: Pandoc wasn't found.
Please check that pandoc is installed:
https://pandoc.org/installing.html
```

```
[46]: !ls ch01*
```

ch01-intro.html   ch01-intro.ipynb   ch01-intro.pdf   ch01-intro.py

```
[ ]:
```