Erratalist for 4th Edition of A Primer on Scientific Programming with Python

Hans Petter Langtangen

Jan 16, 2015

Page 4. The one-line program to be written in a text editor must have `t**2` and not `t^2`.

Page 11. `%0xd` is not an integer padded with `x` leading zeros, but an integer written with a minimum field width of `x`, where any leading spaces are replaced by zeros.

Page 95. In the program `c2f.py`, the `print` statement has a wrong formatting of the `F(C)` value: `%.51f` must read `%5.1f`, i.e., the same formatting as used for the `C` value.

Page 99. Heading in Section 3.1.5 should be *Function argument or global variable?* (*or* instead of *of*).

Page 130. In Exercise 3.6, Equation (3.10), the sum must be $\sum_{i=0}^{n-1}$.

Page 158. The terminal output for the integral $\int_0^{\pi/2} \sin x \, dx$ should read `1`, not `0.583009`.

Page 189. In the `mymod.py` module, an `import sys` is needed in the test block before `print add1(float(sys.argv[1]))`.

Page 189. The last code showing the test block of the `interest` module needs two corrections: division by 365 if the expression for `years` and an `f` in the `print` statement:

```
if __name__ == '__main__':
    import sys
    p = float(sys.argv[1])
    years = days(1, 2, p)/365.0
    print 'With p=%.2f it takes %.1 years to double' % (p, years)
```

Page 216. In Exercise 4.17, the reference to the program `user_formula.py` should be `integrate.py`. The name of the resulting program is then better named `integrate2.py` than `user_formula2.py`.

Page 273. In the first code block, `x3 = mat(x).transpose()` should be `x3 = mat(x1).transpose()`.

Page 315. End of first paragraph: `Sun` is to be replaced by `Apple`.

Page 426. The argument in the `rase ValueError` call, after `if c*d <= 0`, needs a final `% other`.

Page 447. The title of the chapter should be *Random numbers and simple games.*

Page 676. The last line of the `integrate_ode.py` program should not contain `u'(t)=t**3`, but read

```
print "Numerical solution of u'(t)=%s: %.4f" % \
      (f_formula, integrate(T, n, u0))
```

The four terminal output sessions below are then also wrong: instead of `u'(t)=t**3` it should be `u'(t)=t**exp(t**2)`.

Page 765. Exercise 4.1: The exact solution is $u(t) = 0.2e^{0.1t}$.
== Page 776 ===
Exercise E.24: The first constant on the right-hand sides of equations (E.70) and (E.72) must be 3 and 4, not 2 and 3.

Page 783. Below the equation with $I(t + \Delta t)$, it must read $\Delta t \to 0$ (not ∞).

Page 784. Exercise E.42: The last element of the returned list in the `ProblemSIR.__call__` method should not have a minus sign; it should be `self.nu(t)*I`.

Page 789. Exercise E.46: A value for $I(0)$ is not given. Set $I(0) = 0$.

Page 845-846. To succesfully execute the `c2f.py` program, `cmd` must be `python c2f.y 21` or `./c2f.py 21` (if `c2f.py` is an executable file) unless `.` is in the user's `PATH` variable.

Page 846. 2. Under *Split file or folder name*, the name `user` in the text should be replaced by `/home/hpl` according to the interactive session.

Page 847. The text says “The import statements can actually be dropped since functions from `numpy` and `matplotlib` are imported by default when running the notebook in the browser or by supplying the command-line argument `--pylab` when starting notebooks locally on your machine.”. Now, the use of `--pylab` is discouraged. Also, the functions from `numpy` and `matplotlib` are not automatically imported - you have to do that explicitly.

The recommended way of using IPython notebooks with `numpy` and `matplotlib` is to do

```
import numpy as np
import matplotlib.pyplot as plt
%matplitlib inline
```

If you want the notebook to behave more as Matlab and not use the `np` and `plt` interface, you can instead write

```
%pylab
```