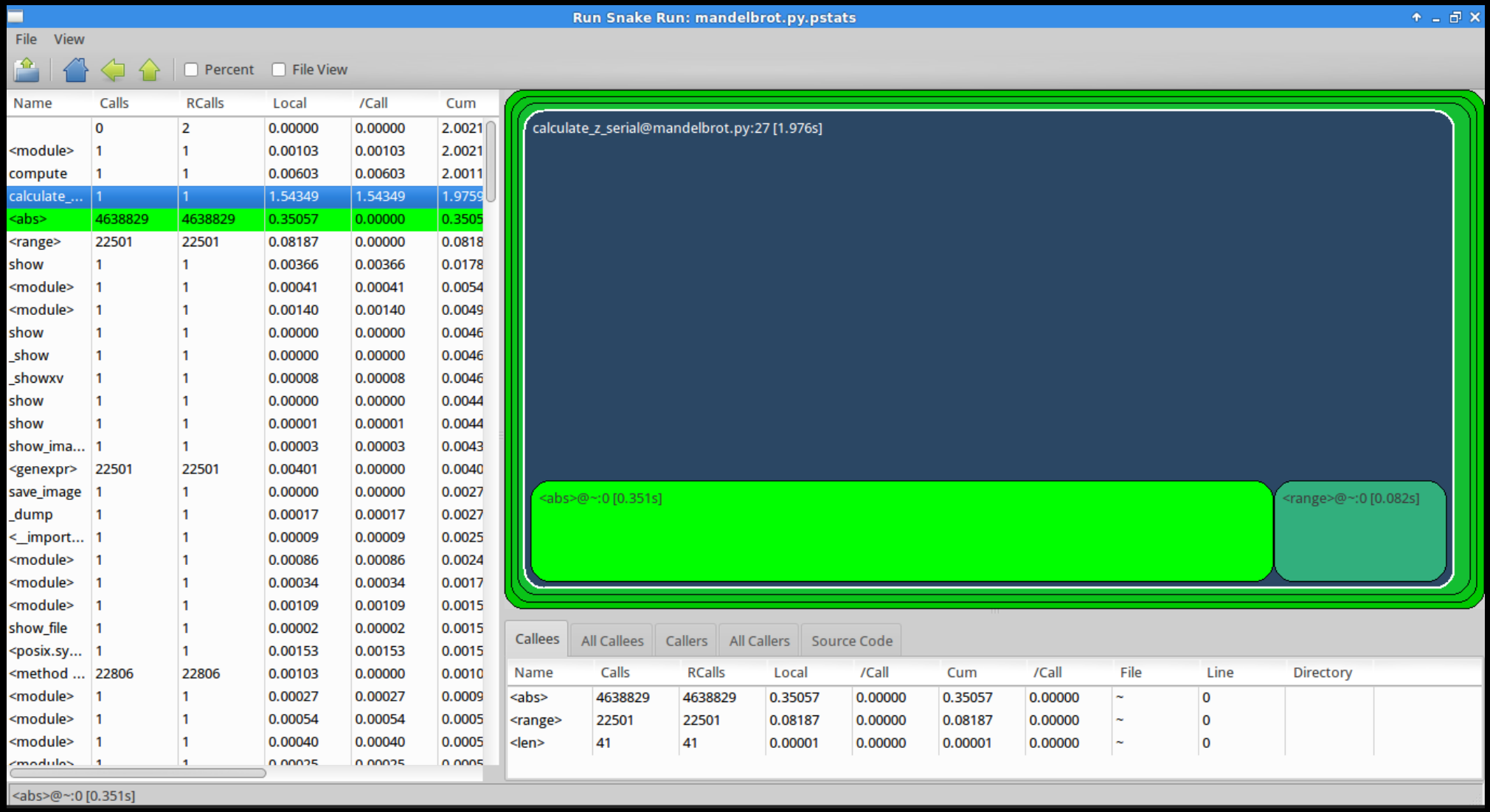
Python profiling 101

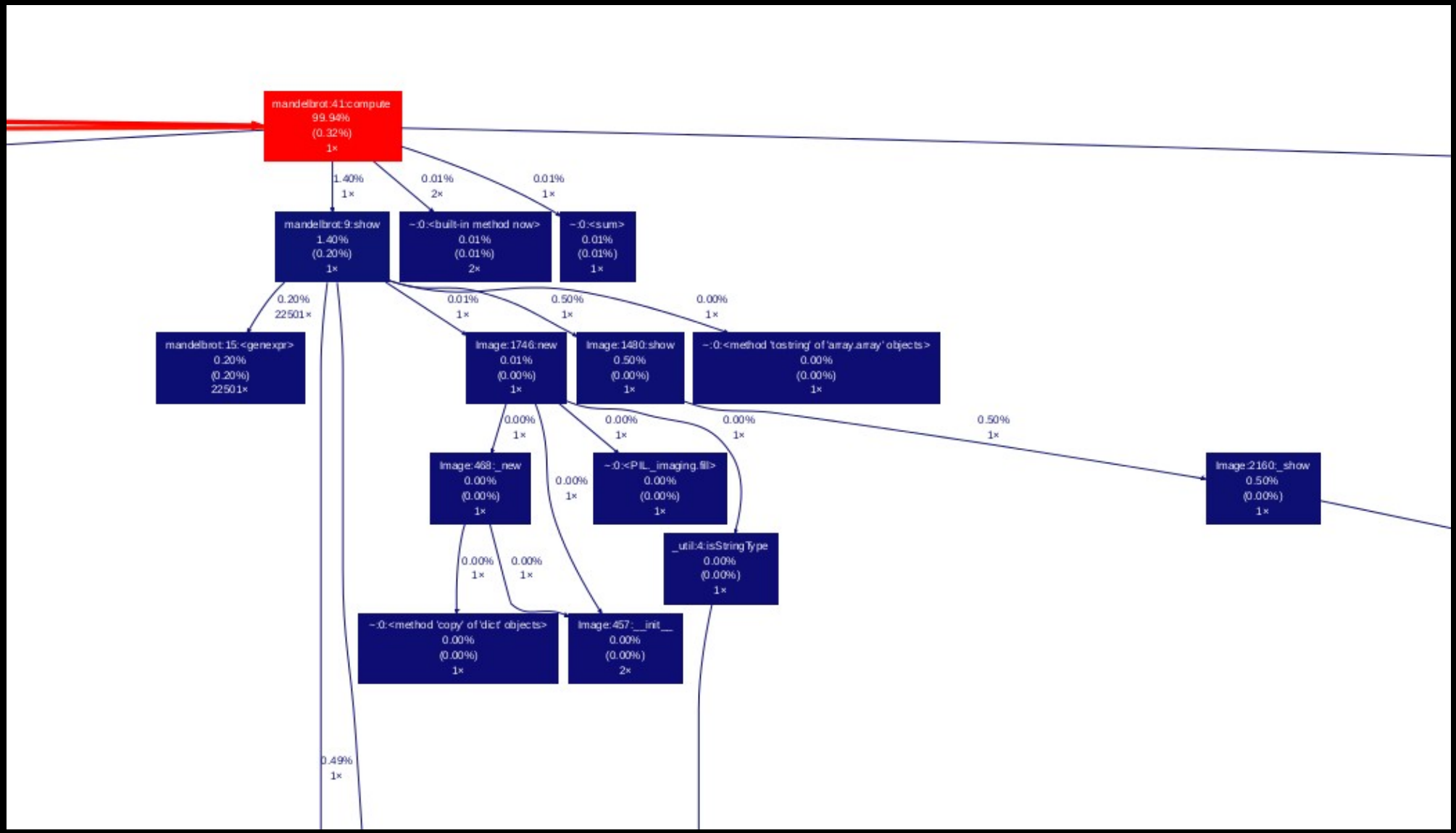
python -m cProfile mandelbrot.py

```
         4707852 function calls (4707840 primitive calls) in 2.099 seconds

   Ordered by: internal time

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    1.622    1.622    2.072    2.072 mandelbrot.py:27(calculate_z_serial)
  4638829    0.364    0.000    0.364    0.000 {abs}
    22501    0.086    0.000    0.086    0.000 {range}
        1    0.006    0.006    2.098    2.098 mandelbrot.py:41(compute)
    22501    0.004    0.000    0.004    0.000 mandelbrot.py:15(<genexpr>)
        1    0.004    0.004    0.018    0.018 mandelbrot.py:9(show)
        1    0.001    0.001    0.005    0.005 Image.py:27(<module>)
        1    0.001    0.001    0.001    0.001 {posix.system}
        1    0.001    0.001    2.099    2.099 mandelbrot.py:2(<module>)
    22806    0.001    0.000    0.001    0.000 {method 'append' of 'list' objects}
        1    0.001    0.001    0.001    0.001 __init__.py:4(<module>)
        1    0.001    0.001    0.002    0.002 FixTk.py:1(<module>)
        1    0.001    0.001    0.001    0.001 {method 'save_ppm' of 'ImagingCore' objects}
        1    0.000    0.000    0.001    0.001 hashlib.py:55(<module>)
        1    0.000    0.000    0.001    0.001 collections.py:1(<module>)
        1    0.000    0.000    0.005    0.005 Image.py:1(<module>)
        1    0.000    0.000    0.002    0.002 tempfile.py:18(<module>)
        1    0.000    0.000    0.001    0.001 io.py:34(<module>)
        1    0.000    0.000    0.001    0.001 random.py:40(<module>)
        1    0.000    0.000    0.003    0.003 Image.py:493(_dump)
        2    0.000    0.000    0.000    0.000 {built-in method now}
```

python -m cProfile -o mandelbrot.py.pstats mandelbrot.py
runsnake mandelbort.py.pstats



python -m cProfile -o ./mandelbrot.py.pstats mandelbrot.py
gprof2dot -f pstats ./mandelbrot.py.pstats -n 0 -e 0 | dot -Tpdf -o mb.profile.pdf



kernprof.py --line-by-line mandelbrot_profile.py
python -m line_profiler mandelbrot_profile.py.lprof

```
File: mandelbrot_profile.py
Function: calculate_z_serial at line 27
Total time: 11.441 s

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    27                                           @profile
    28                                           def calculate_z_serial(q, maxiter, z):
    29         1          278    278.0      0.0      output = [0] * len(q)
    30     22501        17342      0.8      0.2      for i in range(len(q)):
    31     22500        17563      0.8      0.2          if i % 1000 == 0:
    32                                                       # print out some progress info since it is so slow...
    33        23          691     30.0      0.0              print "%0.2f%% complete" % (1.0/len(q) * i * 100)
    34   4643349      3213216      0.7     28.1          for iteration in range(maxiter):
    35   4638829      4163215      0.9     36.4              z[i] = z[i]*z[i] + q[i]
    36   4638829      3940721      0.8     34.4              if abs(z[i]) > 2.0:
    37     17980        13230      0.7      0.1                  output[i] = iteration
    38     17980        74703      4.2      0.7                  break
    39         1            4      4.0      0.0      return output
```

python -c 'from dis import dis; from mandelbrot import calculate_z_serial as c; dis(c)'

```
 28           0 LOAD_CONST               1 (0)
              3 BUILD_LIST               1
              6 LOAD_GLOBAL              0 (len)
              9 LOAD_FAST                0 (q)
             12 CALL_FUNCTION            1
             15 BINARY_MULTIPLY
             16 STORE_FAST               3 (output)

 29          19 SETUP_LOOP             161 (to 183)
             22 LOAD_GLOBAL              1 (range)
             25 LOAD_GLOBAL              0 (len)
             28 LOAD_FAST                0 (q)
             31 CALL_FUNCTION            1
             34 CALL_FUNCTION            1
             37 GET_ITER
        >>   38 FOR_ITER                141 (to 182)
             41 STORE_FAST               4 (i)

 30          44 LOAD_FAST                4 (i)
             47 LOAD_CONST               2 (1000)
             50 BINARY_MODULO
             51 LOAD_CONST               1 (0)
             54 COMPARE_OP               2 (==)
             57 POP_JUMP_IF_FALSE       90

 32          60 LOAD_CONST               3 ('%0.2f%% complete')
             63 LOAD_CONST               4 (1.0)
             66 LOAD_GLOBAL              0 (len)
             69 LOAD_FAST                0 (q)
             72 CALL_FUNCTION            1
             75 BINARY_DIVIDE
             76 LOAD_FAST                4 (i)
             79 BINARY_MULTIPLY
```

http://github.com/pszostek/pyprof101