

COEN 45, MATLAB Programming
Spring Quarter, 2010

Lab assignment #4
The Collatz problem
Apr. 26-27

The subject of this assignment is the Collatz problem, a long-standing mystery of number theory. If you solve the mystery, you will be famous! See “Collatz conjecture” on wikipedia.org for more information, including a really cool fractal plot that I would like to try to figure out how to program.

The Collatz problem is based on this very simple algorithm:

1. Start with any integer $n_0 > 2$
2. Initialize a counter $i = 0$
3. Does n_i equal 1? If so, stop.
4. If n_i is even, divide it by two. Otherwise, triple it and add 1. Store the result in n_{i+1} .
5. Add 1 to i
6. Go to step 3.

Example sequence:

9, 28, 14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1

The mystery is whether every possible starting number results in convergence to 1 in a finite number of steps. As far as I know, no exception to this rule has been found, but neither has anyone proved its truth.

Normally, round-off error does not present significant difficulty, but it could in this problem. That’s because we assume integer arithmetic in the Collatz problem whereas MATLAB does all computations in double floating-point form. The largest integer that can be represented exactly in this form is 2^{53} . If your script produces a larger number, MATLAB will continue, but there will be round-off error that will very likely spoil the results. So when you code the algorithm you must watch for numbers that exceed 2^{53} and stop if you get one.

1. Write a function `collatz` that:
 - (a) Checks for correct input: a single positive integer. Hints:
 - i. `isinteger` will *not* work. Try `floor(n)==n`.
 - ii. Use `mod` to check that n is odd.
 - (b) Executes the Collatz algorithm in a loop (consider `while`)
 - (c) Checks for integer overflow

- (d) Returns two outputs:
- i. `nmax`, the maximum value of n at any step. If overflow occurs ($n > 2^{53}$), set `nmax=inf` and exit immediately.
 - ii. `nit`, the number of iterations (times through the loop) that were executed.
2. After you have checked out your function by exercising it at the command line, get the script `lab4.m` from the class web site. It's not quite complete as you will see. After you get your plots, use the magnifying class on the plot to see if you find any interesting patterns.
3. Have some fun with the script. See how high you can go. Note: if you try to compute the Collatz results for all odd integers up to a very large number like 10^9 you will probably run out of memory before you get overflow. But you can try very large numbers one at a time. `lab4.m` as written takes about 30 sec on my AMD64 box at home. If it takes too long in the lab, you may have to cut back on the number of trials. *Warning:* if you run the CPU flat out on your laptop for 20 minutes, which you can probably do with this script, it could overheat!

Demonstrate your work to the lab assistant. Submit a signed printout of your script file along with a printed copy of your plot with you name and student ID in the title.

You can find an enormous amount of math associated with the Collatz problem. In fact, an entire conference was held on just this problem:
<http://www.math.grin.edu/~chamber1/conf.html>