Siddharth Salunkhe

Dr. Mark Smith

ME 2016/ Section E

30 September 2017

Analysis of Pipe Flow for Dunetown

To determine the specs of the pipe required to supply water to Dunetown, I have run a MATLAB simulation that models the situation. Using the parameters that the pipe is steel, with roughness 0.2 mm, length 2 km, lake elevation 30 m, fluid viscosity 1.12E-6, and gravitation constant 9.807 m/s^2, I wrote programs to model the friction factor, the average velocity in the pipe as well as plot the flow rate against the pipe diameter to ultimately reveal that we need to install a 22" diameter pipe to attain a flow rate of 10000 gallons/min.

The first program, ffactor, was determining the friction factor using Newton-Raphson method to iterate the Colebrook formula until the tolerance was within 10 times machine precision. The function takes in the Reynold's number and the roughness/diameter ratio. If the Reynolds number is less than 2100, the laminar friction factor is used, otherwise, the initial value for the friction factor is assigned, then the Colebrook formula and its derivative are used to converge to the root of the equation.

The second program, pipeflow, calculates the average velocity (m/s) in the pipe as a function of the diameter and using the constants described above. The formula for velocity is dependent on the headloss in the pipe and the starting height (z1) of the water source. The headloss in the pipe is dependent on the friction factor, so the factor function is used as well with the Reynold's number argument as a function of V. This root-finding problem is also iterated to a tolerance of 10 times machine precision. The root is found using the built-in fzero function in MATLAB. A known data point, d = 0.1, V = 1.080244 was used to test this.

The third program plots the function of flow rate vs. pipe diameter and outputs the various flow rates (gal/min) for diameters from 1 to 36 inches. Using interpolation, the program determined that the pipe diameter necessary to supply 10000 gal/min of water lies between 20 and 21 inches. Analyzing the plot reveals that the flow rate increases as the diameter increases, so the pipe diameter must be greater than 20 inches. Looking up available pipe diameters online indicates that the smallest diameter steel pipe [1] that will supply this flow rate output is 22 inches.

Appendix

In order to verify the code, I tested each function with known data points. By solving certain root finding problems using a calculator and checking the result against the result from ffactor, I was able to verify that the factor function converges to the correct roots.

Here is a table of values that I tested the ffactor function:

Table 1: Values of ffactor root finding problem tested with calculator

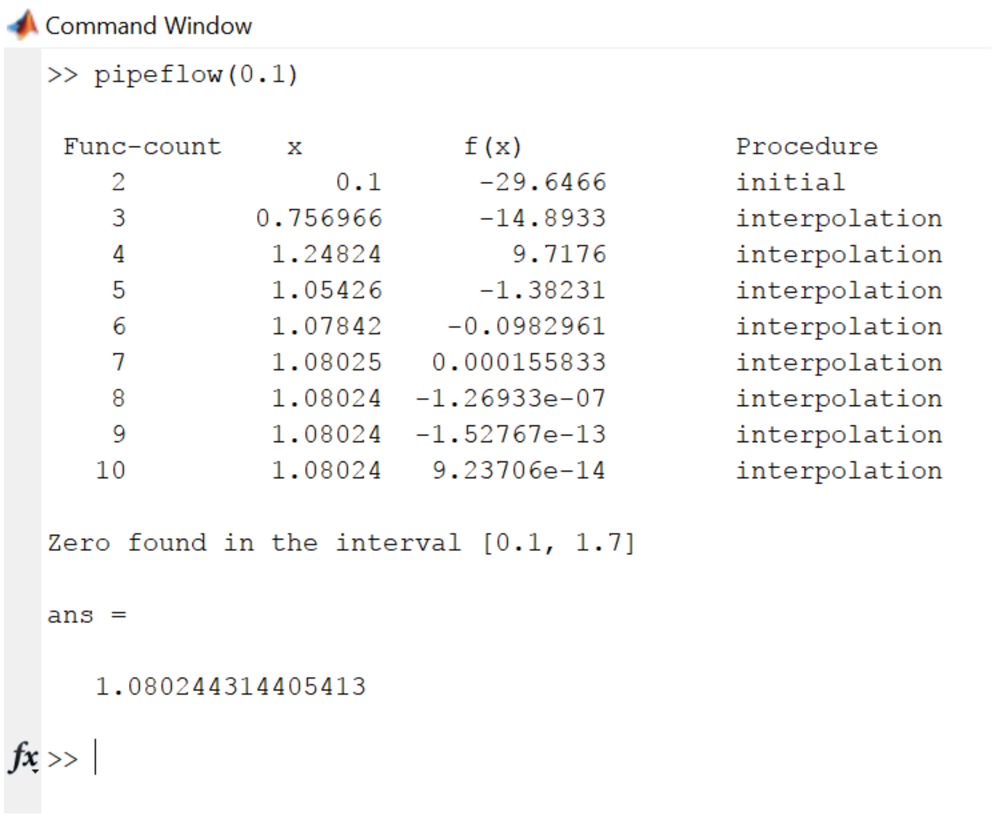
Re	e/D	Calculator	ffactor
2101	0.0067	0.05373	0.05373
21010	0.033	0.0609	0.0609
2101065	6e-9	0.001020	0.001020
100	0.44	0.64	0.64

For the pipeflow function, I tested the result against the given value. For a diameter of 0.1 m, the average velocity of the flow is 1.0802 m/s.

I plotted the pipeflow function so that I could verify if the roots were reasonable and were bounded in a reasonable range. Through some analysis I found that the pipeflow function is a quadratic function with a positive and negative root. Since the Velocity should be positive, and the function is quadratic, the bounds of 0 to a large positive number made sense to encompass the root.

I verified the iteration performance of each of these by finding which bounds or initial value would minimize the number of iterations to converge. For the ffactor function, the initial value was not decided by me, but the iteration performance was dependent on the initial value and the number of times the function and its derivative were evaluated.

The iteration performance of the pipeflow function was affected by the bounds of convergence. Larger bounds led to more iterations. I experimentally and graphically determined bounds that encompassed the root in the least number of iterations by printing the iteration summary while not erroring.



```
>> pipeflow(0.1)

 Func-count    x          f(x)             Procedure
    2             0.1      -29.6466        initial
    3        0.756966      -14.8933        interpolation
    4         1.24824        9.7176        interpolation
    5         1.05426      -1.38231        interpolation
    6         1.07842    -0.0982961        interpolation
    7         1.08025   0.000155833        interpolation
    8         1.08024  -1.26933e-07        interpolation
    9         1.08024  -1.52767e-13        interpolation
   10         1.08024   9.23706e-14        interpolation

Zero found in the interval [0.1, 1.7]

ans =

   1.080244314405413

>> 
```

Figure 1: Iteration summary for pipeflow fzero call

If the iteration summary were not printed by the fzero options, another way to find out the number of iterations would be to print a final summary after every convergence of ffactor. Since factor is called once every iteration of fzero, after ffactor converges, it signals the iteration concluding as well.

References:

[1] http://www.saginawpipe.com/steel_pipe_chart-3.htm