Isaac Huntsman

08/30/25

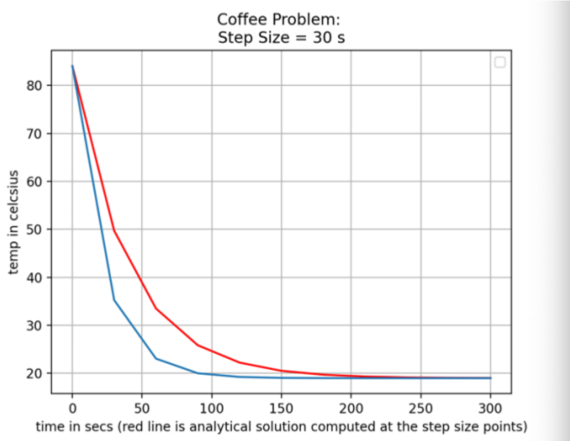
CS3200

Assignment 1

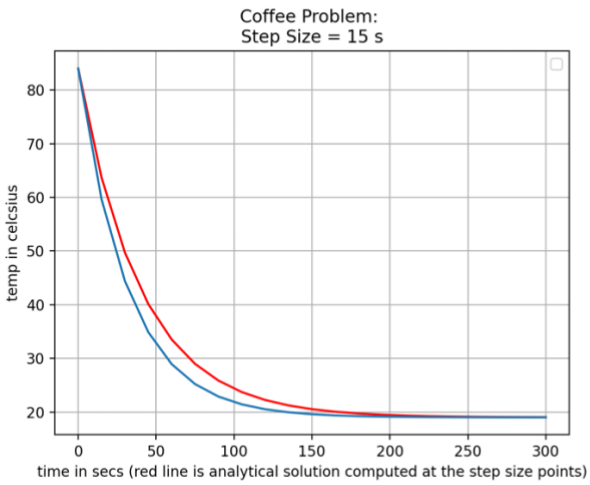
1. Solve the problem analytically

Newton's law of cooling formula (rearranged) is T(t) = Ts + (T0-Ts)e^(-rt). Plugging in our parameters, we have T(300 seconds) = 19 + (84-19)e^(-0.025*300), yielding 19.03595.

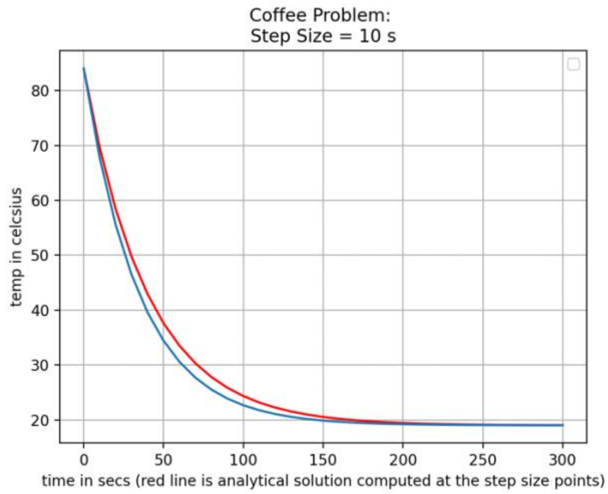
2. (Source code attached)
3. Plots
    a. Graphs using forward Euler at step sizes from 30s to 0.25s
        i. At step size of 30 seconds, biggest error is 14.45 degrees



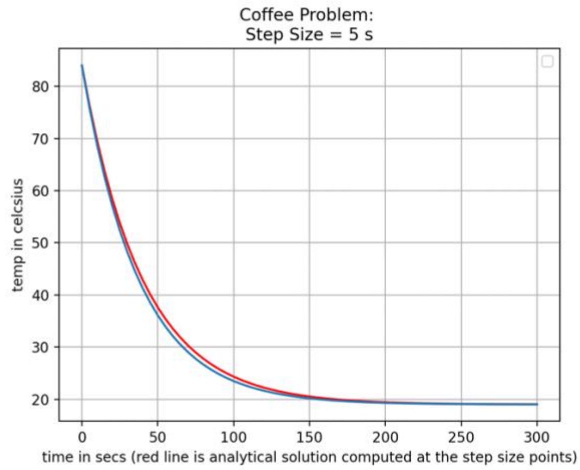
        ii. At 15s, biggest error is 5.3 degrees



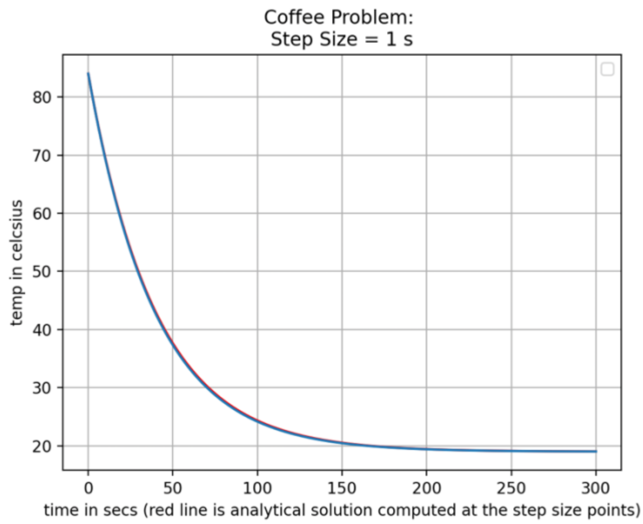
iii. At 10s, biggest error is 3.34 degrees



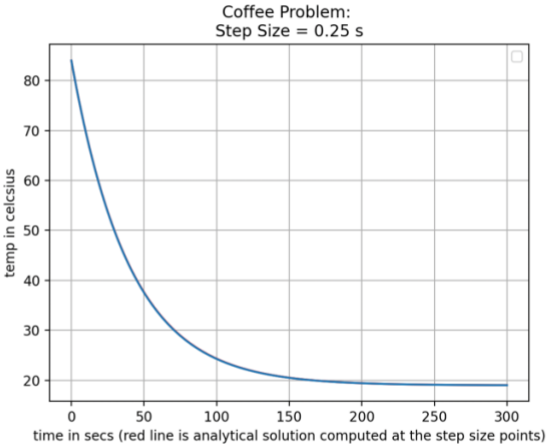
iv. At 5s, biggest error is 1.5 degrees



v. At 1s, biggest error is 0.3 degrees. At this point the lines look almost the same. I'm just going to skip 0.5 and go straight to 0.25



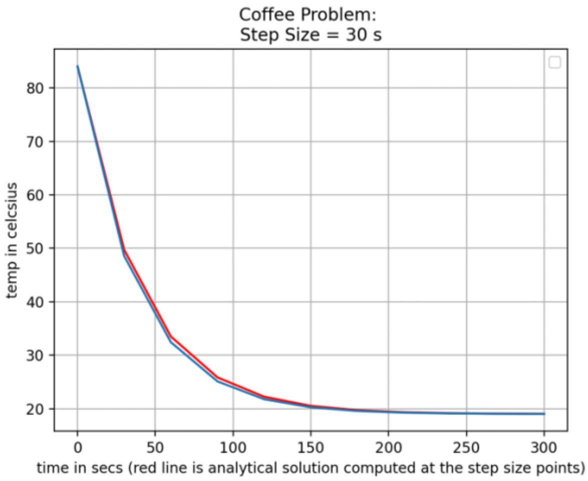
vi. At 0.25, the biggest error is .07 degrees, occurring at 40s mark



Summary: The lines look the same at a very small step size, but at big step sizes, the forward Euler method has a large error. The error starts at the steepest part of the line, but then propagates through until near the end in the bigger step size sections.

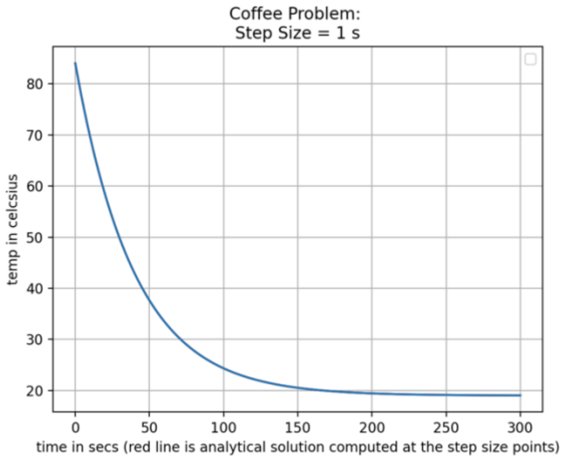
b. Graphs using trapezoidal Euler method:

i. At step size h=30, the biggest error is only 1.158, occurring at step size 30s. This is extremely better than the forward Euler already. It's kind of trivial to print all the graphs, so please don't doc me points, but I'm just going to jump to 1s steps and then 0.25s steps

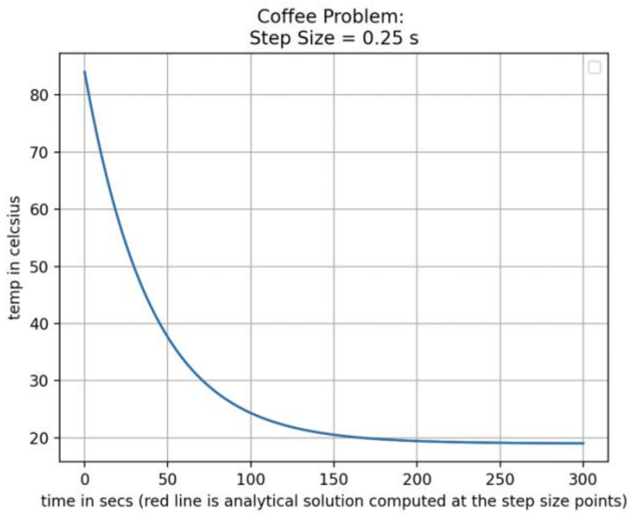


next.

ii. Skipping a few (you're welcome to run my code and see each h), at step size h=1, the biggest error is a staggering 0.0012, which is a more than twice as accurate as the smallest step size we tried on the forward Euler method.



iii. At the smallest step size of 0.25, the biggest error is 7.857143036460457e-05 found at 40 seconds



As described in the diagram-level comments, trapezoidal method is immediately better by a factor of around 13x, at the biggest step size of 30s. At the smallest step size, trapezoidal is *extremely* accurate to the 5th decimal place.

Description of methods: Forward Euler and Trapezoidal Euler implemented iteratively in python. MatPlotLib for visualization.

Sources:

https://math.libretexts.org/Bookshelves/Differential_Equations/Numerically_Solving_Ordinary_Differential_Equations_(Brorson)/01%3A_Chapters/1.02%3A_Forward_Euler_method

This class material was used as a source (https://my.eng.utah.edu/~cs3200/)

Collaborators: none