EN.510.466.01.FA25: Materials Modeling
Homework 2 Report

Manish Chandra
JHED: mkarumu1@jh.edu

Problem 1: Carbon Dating

a. Derivation of Half-Life and Decay Constant Relationship

The analytical solution for the number of radioactive nuclei $N(t)$ at a given time t is given by the exponential decay formula:

$$N(t) = N_0 e^{-t/\tau}$$

where N_0 is the initial number of nuclei and τ is the decay time constant.

The half-life, $T_{1/2}$, is defined as the time it takes for exactly half of the initial nuclei to decay. At this specific time, the number of remaining nuclei is $N_0/2$,

$$\implies N(T_{1/2}) = \frac{N_0}{2}$$

By substituting the definition of half-life into the analytical decay equation, we can solve for the relationship between $T_{1/2}$ and τ:

$$\begin{aligned}
\frac{N_0}{2} &= N_0 e^{-T_{1/2}/\tau} \\
\frac{1}{2} &= e^{-T_{1/2}/\tau} \\
\ln\left(\frac{1}{2}\right) &= -\frac{T_{1/2}}{\tau} \\
-\ln(2) &= -\frac{T_{1/2}}{\tau} \\
T_{1/2} &= \tau \ln(2) \\
T_{1/2} &= \tau \times 0.6931
\end{aligned}$$

This shows that the half-life is directly proportional to the decay time constant by a factor of $\ln(2)$, which is approximately 0.6931.

2. Numerical and Analytical Comparison of Activity

The activity of the sample was calculated numerically using the Euler method for time-step widths of 10 and 100 years. The results are plotted alongside the exact analytical solution in Figure 1. The numerical solutions closely approximate the analytical curve, basically there's no difference between 10 and 100.

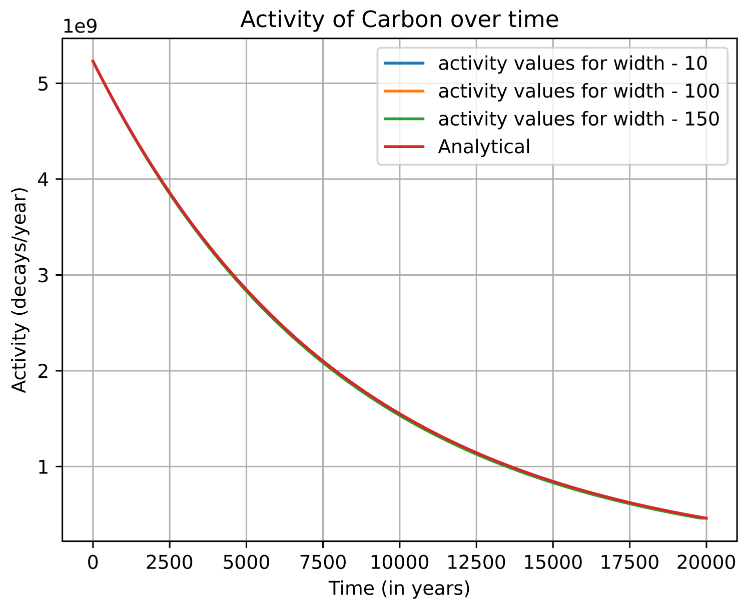

Figure 1: Comparison of numerical and analytical solutions for the activity of a ^{14}C sample over 20,000 years.

3. Discussion of Numerical Error

For the time-step width of 1000 years, the percentage deviation from the exact result after two half-lives (11,400 years) was calculated to be 3.9163%. Just to make sure, I tried calculating the error for different widths = $[2000, 3000, 4000]$ etc. and have experienced a strange phenomenon. A sawtooth-like pattern emerged (Figure 2). At first, this suggested a potential instability in the code. However, further analysis revealed the cause to be a issues with width and time step. My hypothesis for why this behavior arises is because the target time (11,400 years) is not always an integer multiple of the step width (Δt). The calculation to find the corresponding data point, **'index = int(time/width)'**, truncates the result. This leads to a comparison between the numerical value at a slightly earlier time and the analytical value at the precise target time. The plot's jaggedness is therefore a measure of how well or poorly each step width aligns with the fixed measurement point. This result powerfully demonstrates the importance of comparing numerical and analytical data at identical time points to obtain a meaningful measure of error.

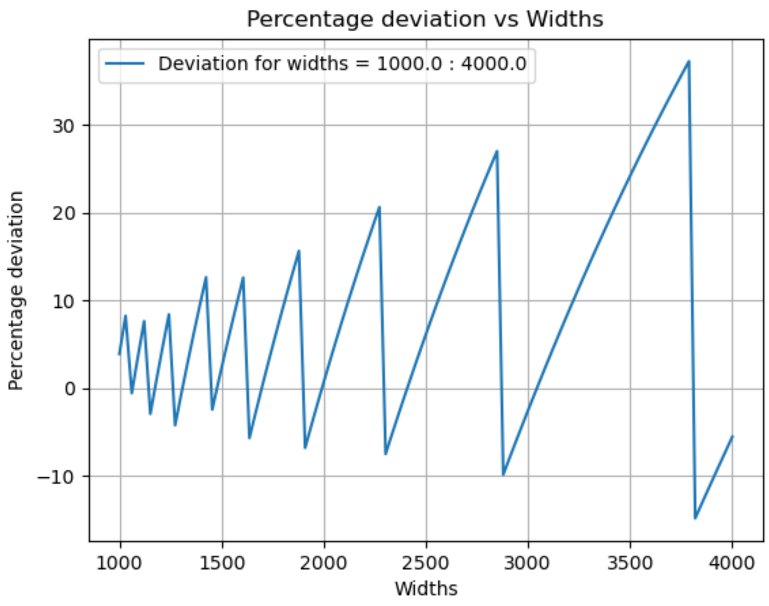

Figure 2: Percentage deviation at a fixed time point (2 half-lives) for various step widths, demonstrating a sampling artifact.

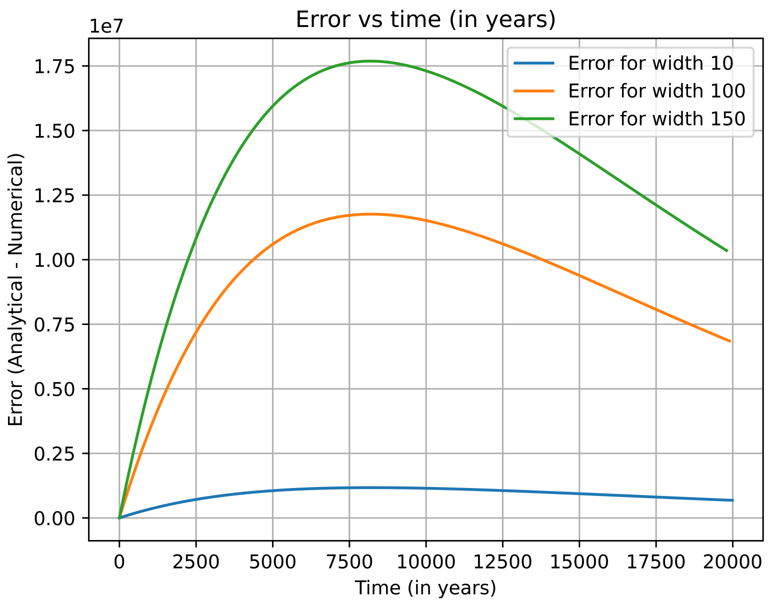

Figure 3: Error vs Time Plot

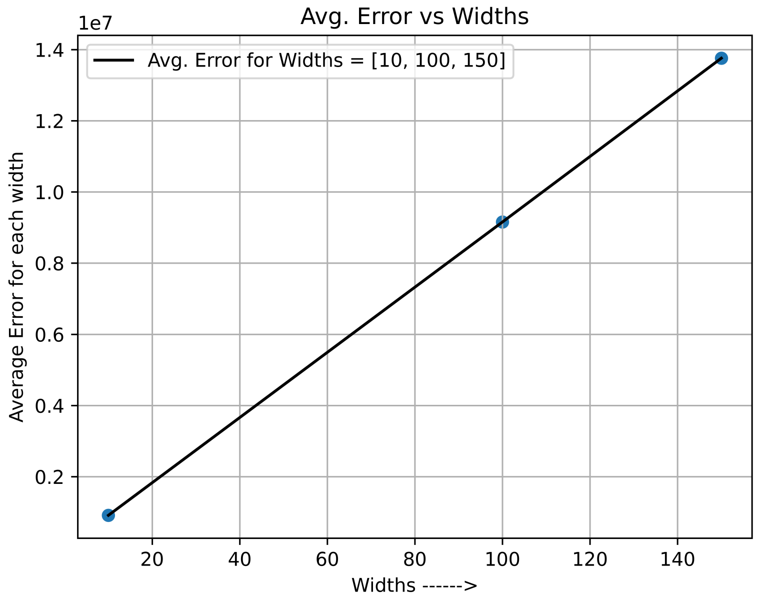

Figure 4: Average Error vs Different Widths

I've also made some extra plots to see if my algorithm is working properly. These error plots prove that the algorithm is doing calculations fine. These plots have also proved me that the width and step size play a major role in maintaining high accuracy.

Problem 2: Golf Ball Trajectory

The trajectory of a golf ball was simulated under four different physical models. The equations of motion were solved numerically using the Euler method (discritized).

In order to solve the position equations, velocity is needed and to solve for velocities, acceleration was needed. The trajectory was calculated by numerically integrating the equations of motion using the explicit Euler forward method. At each time step Δt, the position and velocity for the next step, $(t + \Delta t)$, were calculated based on the values from the current step, (t). The specific update equations for each of the four simulated models are detailed below. Note that the velocity magnitude at time t is defined as $v(t) = \sqrt{v_x(t)^2 + v_y(t)^2}$.

1. **Ideal Trajectory:** In the absence of aerodynamic forces, the acceleration is constant.

$$\begin{aligned} v_x(t+\Delta t) &= v_x(t) \\ v_y(t+\Delta t) &= v_y(t) - g\Delta t \\ x(t+\Delta t) &= x(t) + v_x(t)\Delta t \\ y(t+\Delta t) &= y(t) + v_y(t)\Delta t \end{aligned}$$

2. **Smooth Ball with Drag:** Includes a quadratic drag force with a constant drag

coefficient, $C = 0.5$.

$$
\begin{aligned}
v_x(t+\Delta t) &= v_x(t) - \frac{C\rho A}{m} v(t) v_x(t) \Delta t \\
v_y(t+\Delta t) &= v_y(t) - g\Delta t - \frac{C\rho A}{m} v(t) v_y(t) \Delta t \\
x(t+\Delta t) &= x(t) + v_x(t)\Delta t \\
y(t+\Delta t) &= y(t) + v_y(t)\Delta t
\end{aligned}
$$

3. **Dimpled Ball with Drag:** The drag coefficient C becomes velocity-dependent.

$$
C(v) = \begin{cases} 0.5 & \text{if } v(t) \leq 14 \text{ m/s} \\ 7.0/v(t) & \text{if } v(t) > 14 \text{ m/s} \end{cases}
$$

The update equations are the same as for the smooth ball, but using this velocity-dependent $C(v)$.

4. **Dimpled Ball with Drag and Spin:** Adds the Magnus force for backspin.

$$
\begin{aligned}
v_x(t+\Delta t) &= v_x(t) - \frac{C\rho A}{m} v(t) v_x(t) \Delta t - \left(\frac{S_0\omega}{m}\right) v_y(t)\Delta t \\
v_y(t+\Delta t) &= v_y(t) - g\Delta t - \frac{C\rho A}{m} v(t) v_y(t) \Delta t + \left(\frac{S_0\omega}{m}\right) v_x(t)\Delta t \\
x(t+\Delta t) &= x(t) + v_x(t)\Delta t \\
y(t+\Delta t) &= y(t) + v_y(t)\Delta t
\end{aligned}
$$

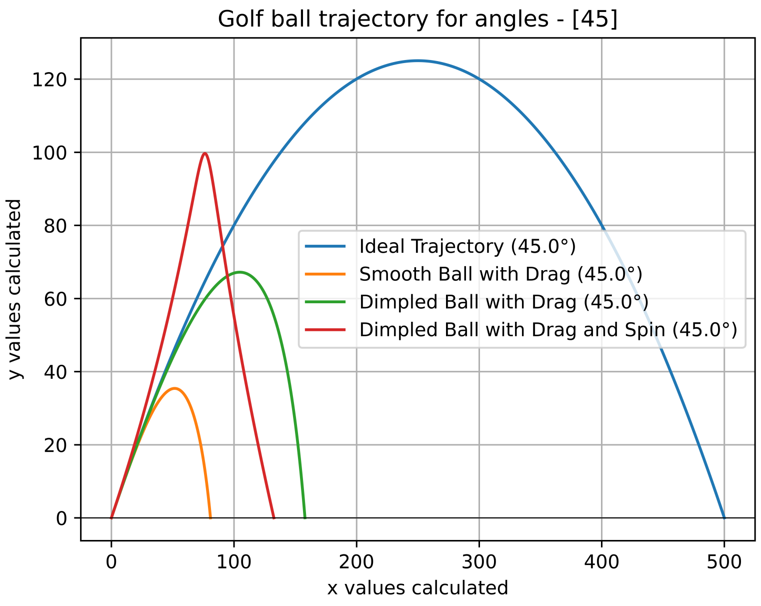

Figure 5: Comparison of the four golf ball trajectory models for an initial angle of 45°.

Approach

Both problems were solved numerically using the Euler method by discretizing the equations into forward in time form. The issue was to make sure the shapes are matching to produce the plots. First, I solved for ideal trajectory and then pasted the same block and added new acceleration term for both x and y direction and ran it through loop. The same goes on for dimpled and spin with drag. One particular issue that troubled me a lot was updation and storage of state variables that vary at each iteration. The sequence inside the loop affects the data storing. I havge used **DEBUG = True** methods to check if the data was storing properly.

Contributions :

Everyone contributed equally. Initially before solving the homework, I had several doubts about the homework, but I received help from my group members who helped me to understand the problem and code them. Yari, Ellen, Han all three of them helped me to discuss if the method I followed or their methods were correct. This has improved the grasp of the problem.

Resources used from online [1] [2]

References

[1] Python Software Foundation, "argparse — parser for command-line options, arguments and sub-commands," 2025. Official documentation for the Python argparse library, section on the 'dest' parameter.

[2] C. Oses, "Lecture notes for en.566: Numerical integration and the euler method."