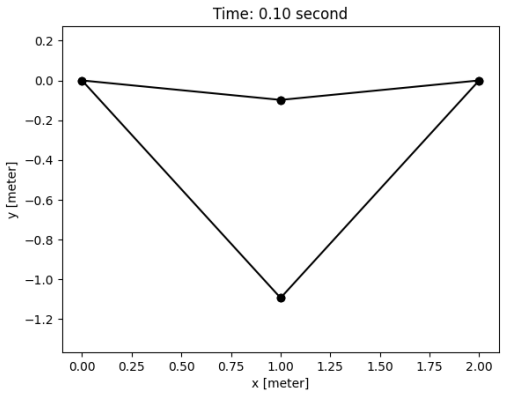
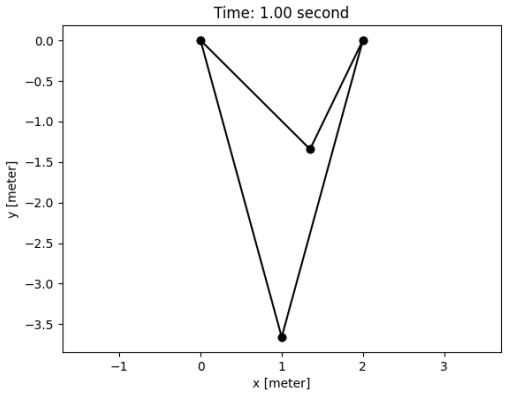
MAE 263F Fall 2025 Homework_1

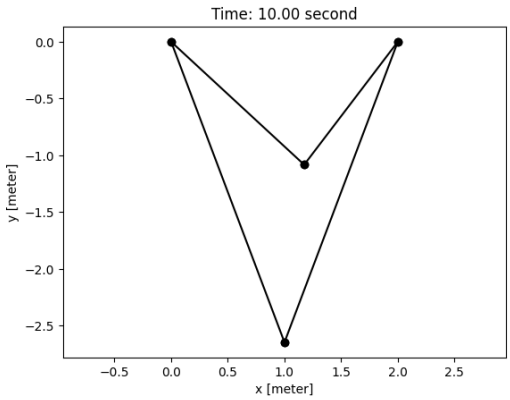
Luke Chang 905954847

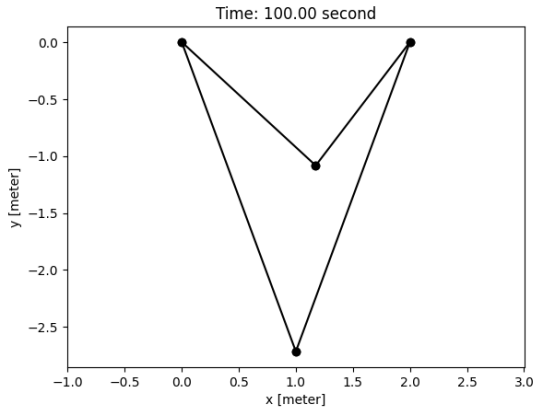
I. Task Output

Plot the shape of the spring network at t = {0, 0.1, 1, 10, 100}s. Include axis labels with units, legends if needed, and a title indicating the time.

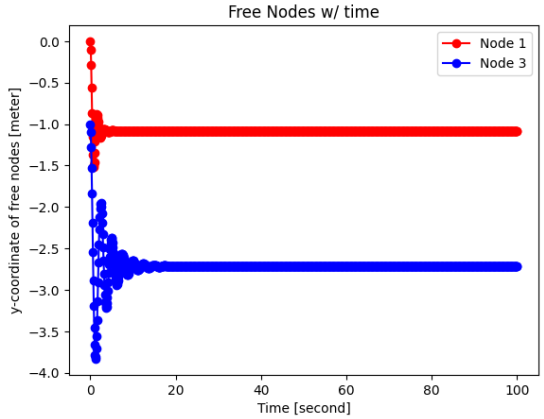








As a function of time, plot the y-coordinates of all free nodes (i.e., nodes 1 and 3). Include labels with units.



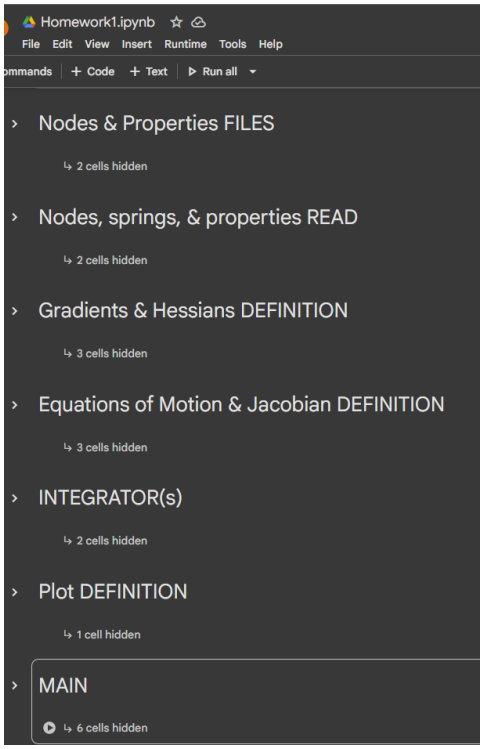
II. Report Questions

A. Pseudocode and code structure: Write pseudocode for your simulator describing its main logic and workflow. Briefly describe the main functions and scripts in your implementation (2–3 sentences each), including their inputs and outputs. Create a simple block diagram showing how the functions or scripts interact (i.e., which one calls which). There is no strict format requirement—use your best judgment to make it understandable to someone with an undergraduate-level engineering background.

All simulations for this homework can be ran through the Homework1.ipynb and can be summarized into

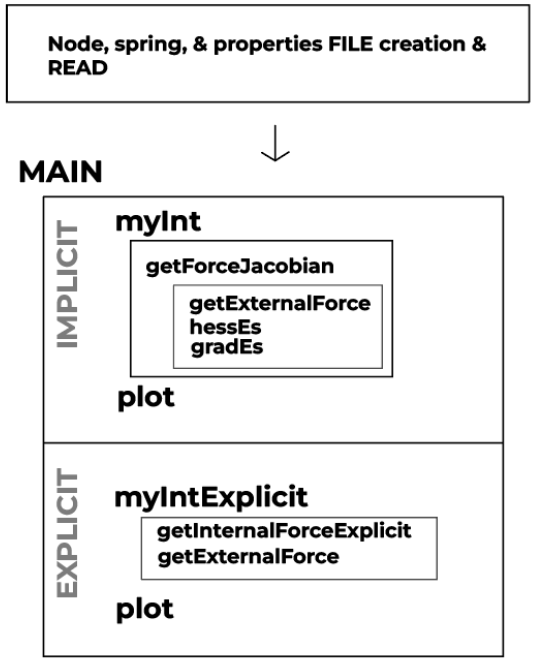
1. Node, spring, & properties FILE creation
2. DEFINITION of Gradients, Hessians, EOMs, and Jacobian get functions

3. INTEGRATOR function including both implicit and explicit integrators
4. & final plotting functions & the MAIN script.



The code is rewritten largely from the Lecture 3 example Google Colab provided by Professor Jawed, with edits to include a 4th free node as well as explicit integration.

- Node & Properties FILES includes just the initialization of the node placements and spring connections.
- Nodes, springs, & properties READ takes in the node and property files and dissects them into usable matrices.
- GradEs and hessEs definitions calculate the stretching energy gradient and hessian matrices with the given set of 2x x & y coordinates, stiffness, and unstretched reference length.
- Equations of Motion & Jacobian DEFINITIONs calculates the external forces acting on the system with the use of gradEs & hessEs defined above. Additionally, an explicit version without incorporating the intertial term is added here.
- INTEGRATORS portion does the actual calculations for new positioning with myInt being the implicit method with x_new as an input needed and the Newton Raphson method. The alternative myIntExplicit function utilizes the explicit internal force method from the previous section solving for the next positioning algebraically.
- Finally the PLOTTING section simply loops through the nodes/springs to plot the current location and with everything above are utilized in the MAIN section.
- The MAIN portion contains the actual time loop simulation, defining time steps and maximum ranges to simulate as well as the fixed and free nodes. The integrator is called here to calculate subsequent node locations and subsequently plotted. A duplicate Explicit simulation calling the explicit variants of the getForce and integrator functions are also included here.



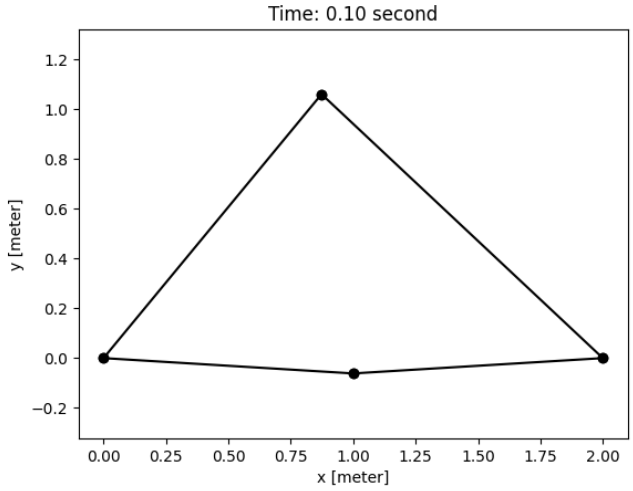
B. How do you choose an appropriate time step size Δt?

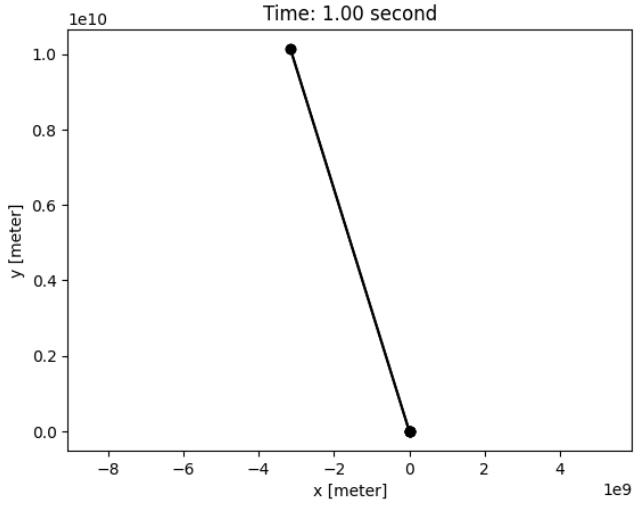
To choose an appropriate time step size Δt, convergence and accuracy must be considered. A small enough time step to capture spring oscillations is needed. This probably should be in respect to the stiffest spring and the natural frequency for such can be considered, which is crucial for the implicit Euler method with numerical damping & oscillations.

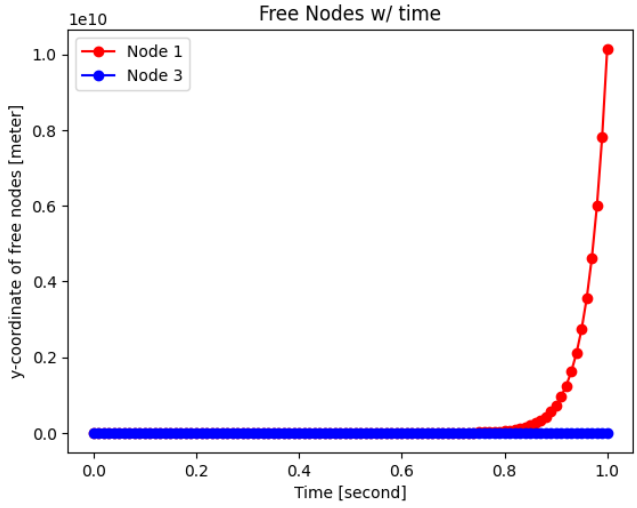
C. Simulate the spring network using Explicit Euler for t ∈ [0, 1] s. If it becomes unstable, reduce Δt and try again. If it still fails even at Δt = 10−6 s, state this in the report. Explain which method (implicit vs. explicit) is preferable for this spring network and why.

The below shows the spring network simulated using Explicit Euler with both at Δt = 1 & 10−6 s. Unfortunately the simulation does become unstable and fails even at super fine time steps of < 10−6. As such it seems for this spring network, the implicit method would be preferred as much more computational power is needed. Although the explicit method seems easier, the implicit route allows for much larger time steps in sacrifice of supposedly more accurate results.

T







D. The simulation with implicit Euler appears to reach a static state (numerical damping). Read about the Newmark–β family of time integrators and explain how such integrators can mitigate such artificial damping..

The Newmark–β family of time integrators can mitigate artificial damping by allowing one to control how much should be introduced with the use of β and γ as tuning parameters. Although there is no numerical damping with constant acceleration method, high frequency noise that arises can be suppressed with these parameters.

REFERENCES

[1] K. J. Majeed, "Spring-Mass Network Simulation Example (Implicit Euler)," course Colab notebook, MAE 263F: Mechanics of Flexible Structures and Soft Robots, University of California, Los Angeles, Fall 2025.

[2] *MOOSE Framework Documentation*, "Newmark Beta Time Integrator: Constant Average Acceleration Method," Idaho National Laboratory, 2025. [Online]. https://mooseframework.inl.gov/source/timeintegrators/NewmarkBeta.html#:~:text=When%20using%20the%20constant%20average