AE 370 Group Project 2 - 1D Beam Behavior

Philip Zolfaghari, Sidharth Nagarajan, JT Termanas, Simon Rodriguez, Jakub Mitka
University of Illinois, Urbana-Champaign - Aerospace Engineering

We will model how a beam behaves as a function of material properties and the specific application of the wave equation. We will explore two distinct numerical methods: Finite Difference Time Domain (FDTD) Method and Trapezoid Method. We will apply the latter to solving the Euler-Bernoulli Dynamic Beam equation for applications to a 1-D beam bending solution. The former will inform on the applications of the wave equation with vibrational dynamics problems.

Nomenclature

$\Delta t, \Delta x$	Time Step, Spatial Step	L	Beam Length (m)
L	Length of the beam (m)	w	Deflection of Beam (m)
E	Young Modulus (Pa)	c	Wave Speed (m/s^2)
μ	Shear Modulus (Pa)	β	Damping Coefficient
I	Moment of Inertia(kg^2 m)	g	Forcing Function

I. Introduction

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} \tag{1}$$

The wave equation is a second-order partial differential equation (PDE) that describes the dynamics of waves. The relationship can be expressed through the acceleration of the wave $\frac{\partial^2 u}{\partial t^2}$ and its spatial curve $\frac{\partial^2 u}{\partial x^2}$ shown in equation 1. This project will explore two distinct applications of the wave equation. First, to explore 1D beam bending with various materials using a forcing function to generate an expected excitation. Second, to use the Euler-Bernoulli Dynamic Beam Equation to demonstrate wave propagation within a material. It should be noted that due to time constraints, we could not implement a dampened solution modeled by equation 2.

$$\frac{\partial^2 u}{\partial t^2} + \beta \frac{\partial u}{\partial t} - c^2 \frac{\partial^2 u}{\partial x^2} = 0 \tag{2}$$

The forcing function g will be used to encode information on the following cantilever beam boundary conditions listed below in Equations 3. These conditions describe no vertical displacement as well as bending/rotation experienced on the fixed end of the beam, and at the free end they display no shear force and bending moment allowing for that end to be free to rotate.

$$\hat{u}_n = 0, \quad \frac{d\hat{u}_n}{dx} = 0 \quad \text{at } x = 0; \qquad \frac{d^2\hat{u}_n}{dx^2} = 0\,, \quad \frac{d^3\hat{u}_n}{dx^3} = 0 \quad \text{at } x = L. \tag{3}$$

The **A** matrix encodes information on our central-difference coefficients which will be used later in the spacial discretization.

No matter which case of the wave equation, there exists constants that define its behavior through a material. E defines the Elastic Modulus of the beam material, while I describes the moment of inertia; in all cases, we assume $I_{beam} = \frac{1}{3}ML^2$ with $M = \frac{3}{L^2}$.

The Euler-Bernoulli Dynamic Beam Equation expands on the 1D Wave Equation through 4th-order spacial and second-order time derivatives.

$$EI\frac{\partial^4 u}{\partial x^4} = -\mu\frac{\partial^2 u}{\partial t^2} + g \tag{4}$$

Given that the wave equation is an IBVP, we will simplify it through spatial discretization along the x position on the beam. Next, we discretize the time derivative using an explicit FDTD method. Given the explicit nature of FDTD and the inability to use an implicit method given the 4th-order nature of the dynamic wave equation, we use the Courant–Friedrichs–Lewy (CFL) conditional convergence criteria to solve the problem.

We follow a similar framework for our simplified wave equation but instead rely on the implicit Trapezoidal method to solve the IBVP. We will explore the implementation of these two methods later in the methodology section.

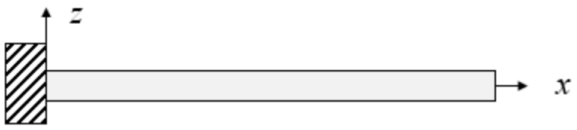


Fig. 1 Visual Concept of 1D Cantilever Beam

II. Code Design

A. Finite Difference Time Domain Method

To derive an expression used to analyze the motion of the beam, we used second order Lagrange basis functions to provide discretized approximations for the space and time derivatives. Since we are using a finite difference method [1], we can obtain these using the finite difference coefficients for our derivatives, which in this case are both 2nd order, with second-order accuracy. The resulting expressions for each are:

$$\frac{\partial^2 u_j^i}{\partial t^2} = \frac{u_j^{i-1} - 2u_j^i + u_j^{i+1}}{\Delta t^2} \qquad \text{and} \qquad \frac{\partial^2 u_j^i}{\partial x^2} = \frac{u_{j-1}^i - 2u_j^i + u_{j+1}^i}{\Delta x^2} \tag{5}$$

Where j indicates the current position and i indicates the current time. Once we have discretized expressions for both our space and time derivatives, we can plug them into equation 1 and rearrange to acquire an expression for w_j^{i+1}, the beam's deflection at the next timestep:

$$u_j^{i+1} = 2u_j^i - u_j^{i-1} + r(u_{j-1}^i - 2u_j^i + u_{j+1}^i) \qquad \text{where} \qquad r = (\frac{c\Delta t}{\Delta x})^2 \text{ and } c = \frac{EI}{\mu}. \tag{6}$$

First, we initialize **u** (variable name used in code) as a matrix with dimensions m x n (rows x columns) where m and n are the number of steps in time and space respectively. For our scenario, we have a cantilever beam beginning with no deflection (straight, no bends) and at rest, with simulated forcing and damping. Rather than adding the forcing function to the PDE, we opted to encode the forcing function into the boundary conditions of the free end of the beam. We do this by setting the free boundary conditions equal to a sine wave with respect to time. In addition to simulated forcing, we included simulated damping, which mimics the friction within the material absorbing the vibration, eventually returning the rod to its original, undeformed state. To simulate the damping, we use an exponential decay function. The free end boundary condition is as follows:

$$U = d \cdot sin(2\pi \cdot f \cdot t)e^{-kt} \tag{7}$$

In Equation 7, d is the amplitude of the vibration, f is the frequency, and k is the damping constant. In the animation we generated from our code, the d and f values are set to make the deflections in the bar over time clear

to the viewer as seen in Fig. 2, however the values are not realistic. Realistic d and f values would depend on the material and external force on the rod. This is the first of two damping functions applied to our system. The damping component of the function above serves to simulate the forcing function only being applied at $t = 0$. If the exponential decay term were not present, the forcing function would affect the rod at the same amplitude d throughout duration of the simulation. While this damping function decreases the effect of forcing over time, it does not decay the amplitude of the waves/vibrations generated by the simulated forcing function. To do that, we multiply our solution for u at the next time step by another exponential decay function, $e^{-k_2 t}$, where k_2 is another decay constant. Similarly, realistic k_2 values will be based on the materials rigidity/elasticity or its ability to absorb vibrations.

To increase the efficiency of our code, rather than looping through each individual discretized u function, we set up a matrix A, that calculates the last 3 terms (space terms) of the u^{i+1} equation for all discretized u functions. With the A matrix our equation becomes:

$$u^{i+1} = (C^2 \mathbf{A} u^i + 2u^i - u^{i-1}) e^{-k_2 t} \tag{8}$$

The rest of the boundary conditions are entered into the 1st and 2nd row as well as the first and last column to satisfy the following conditions:

1) The rod begins straight, so the first row is all zeros.
2) The rod begins at rest, so $\frac{\delta u}{\delta t} = 0$, meaning u does not change, meaning the second row is all zeros. By definition since the first derivative at t = 0 is 0, higher derivatives will also be zero.
3) the left end of the bar is fixed (Eq. 3), so the first column is all zeros.
4) the right end of the bar is the forcing function boundary condition mentioned earlier. It is applied to the last column of the u matrix.

Choosing the values for Δx is a sensitive matter in this case. Since the simulated forcing function is applied at only one end of the rod, it takes time to propagate through the full length. The higher the Δx used, the longer it takes to reach the other end of the rod, and vice versa. We can counteract the slowing down/speeding up of the propagation by changing the C constant in the above equation.

Given that this problem is not a stiff IVP, meaning there aren't any rapidly changing values or high derivatives, choosing Δt is a much less sensitive task. However, the lower it is, the more accurate the wave propagation and decay will be. A lower Δt will cause the wave to decay incredibly fast, and barely any propagation will occur.

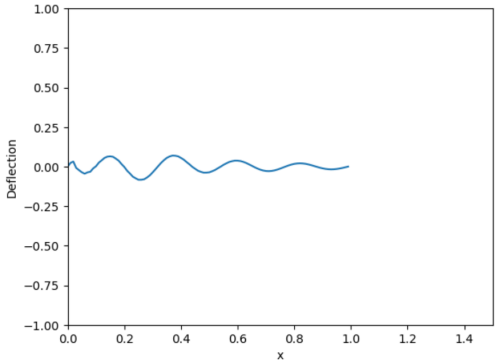


Fig. 2 Vibration Still from Animation

B. Trapezoidal Method

The second method that we will be using is the trapezoid time-stepping method, which is implicit. We will simplify the problem by making the assumption that the beam behaves like the general wave function 1. This method is beneficial because of its stability at low Δt values 9. The general formula is given by:

$$\mathbf{u}^{i+1} = \mathbf{u}^i + \frac{1}{2}\Delta t(\mathbf{f}(u^i, t^i) + \mathbf{f}(u^{i+1}, t^{i+1}) \qquad \text{where} \qquad \mathbf{f}(u^i, t^i) = \dot{\mathbf{u}} \tag{9}$$

We want to obtain some matrix $\mathbf{M}$ such that we can satisfy the following equation:

$$\dot{u}^i = \mathbf{M}\mathbf{u}^i + \mathbf{g}(t^i) \tag{10}$$

As this is the format to solve an IVP problem, we need to find a way in order to get our equation into this format despite having a $\ddot{u}$ rather than the $\dot{u}$ we need. To get around this problem, we can recast our system into the form of:

$$\dot{v} = \mathbf{M}v + g(t) \qquad \text{where, } \dot{v} = \begin{bmatrix} \dot{u} \\ \ddot{u} \end{bmatrix}, \quad v = \begin{bmatrix} u \\ \dot{u} \end{bmatrix}, \quad \text{and } \mathbf{M} = \begin{bmatrix} 0 & \mathbf{I} \\ \frac{c^2 A}{\Delta x^2} & 0 \end{bmatrix} \tag{11}$$

To satisfy our initial equations, we can write our $\mathbf{M}$ matrix in this fashion. This also allows for us consider and take care of our boundary conditions. The $\mathbf{A}$ matrix comprises the central finite difference coefficients that apply to the second derivative with an accuracy of 2. This way allows for us to take care of the boundary conditions associated with the BVP portion of our wave equation, allowing us only to be left with an IVP we have methods to deal with, such as the trapezoidal method we will implement.

This expression can be substituted into the general trapezoid method and simplified to obtain the equation for our problem.

$$\mathbf{u}^{i+1} = (\mathbf{I} - \frac{1}{2}\Delta t\mathbf{M})^{-1}(\mathbf{I} + \frac{1}{2}\Delta t\mathbf{M})\mathbf{u}^i + (\mathbf{I} - \frac{1}{2}\Delta t\mathbf{M})^{-1}(\frac{1}{2}\Delta t[\mathbf{g}(t_k) + \mathbf{g}(t_{k+1})]) \tag{12}$$

This newly found expression allows for us to find the displacement in subsequent timesteps for each spatial location along our beam. With this, we can now model our 1-D wave equation for the deflection of the beam in terms of the displacement for different materials and with different forcing functions.

III. Analysis

A. Finite Difference Time Domain

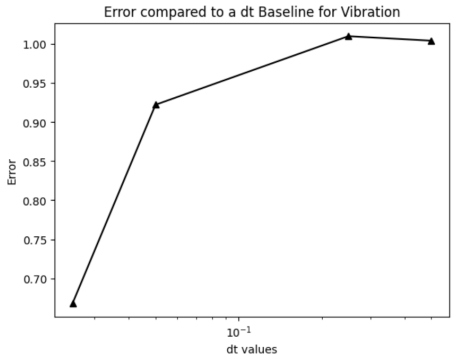


Fig. 3 Vibration Error compared to $\Delta t_{baseline}$ baseline

This error plot was generated by finding the difference between deflection values across space and time for different Δt compared to a $\Delta t_{baseline}$. The error is computed as so:

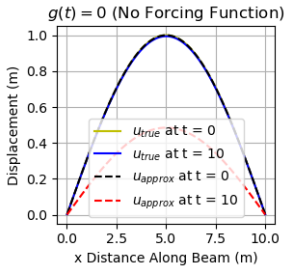
$$e = \frac{|u_{\Delta t_{baseline}} - u_{\Delta t}|}{|u_{\Delta t_{baseline}}|} \tag{13}$$

Where $\mathbf{u}$ is the matrix, including the deflection at all positions and times. The error plot may not accurately represent because the propagation varies based on Δt, and the deflections are not comparable.

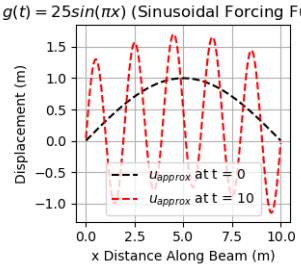
Most systems approximated with explicit methods experience divergence with low $\Delta t_{baseline}$ values, however even with $\Delta t_{baseline}$ values as low as 1 (with duration of 40 seconds), we don't experience any divergence. However for our system we immediately experienced divergence with any C value greater than 1. This is because every iteration we calculate the next time step, we multiply by C, which eventually leads to divergence. Therefore another condition for selecting Δt and Δx values is that $\frac{\Delta t^2}{\Delta x^2} c \leq 1$.

B. Trapezoidal Method

To start off, we will model our displacement under the assumption of no forcing function, and that the material of our beam is aluminum. This means that we will have a c^2 of $\frac{EI}{\mu}$ which is roughly $\frac{69\times10^9}{25\times10^9}$. These values for the E and μ were found online [2]. Plugging in a time length of 10 seconds, with a Δt of 0.01, for a 10 meter long beam and with n equal to 1000. We also set the initial displacement and velocity of our beam, which we used information we found online that provided a good set of initial values for our situation [3]. From these parameters, we get the graph seen in 4a.



(a) Displacement With No Forcing Function



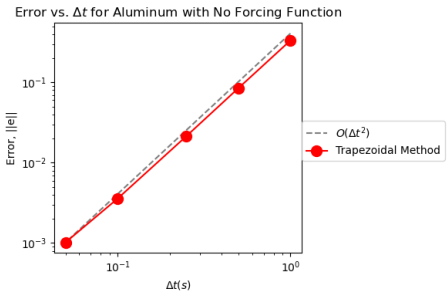
(b) Displacement With Sine Forcing Function

Fig. 4 Displacement From Trapezoidal Method

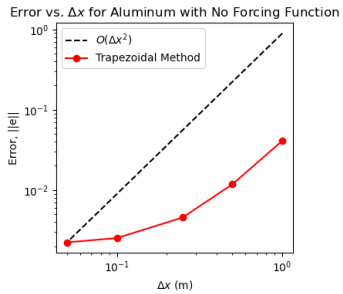
We then also plotted the same material under the same conditions, but with a sinusoidal forcing function of $g(t) = 25sin(\pi x)$. The displacement from this case can be seen in 4b. From both of these forcing functions, we can see that as time goes on, our displacements are affected by the forcing functions. We can compare the true solution, which we found online [3], to these graphs and figure out how much error each system produces using Eqn. 13. In addition to this, we can compare the maximum displacement observed for both forcing function cases between the different materials.

For the zero forcing function case, when measuring the error of each material, the error graphs ended up being very similar in the sense that they all followed our error of order $O(\Delta t^2)$ line very closely. This can be seen in Fig. 5a. In addition to this, for error on the order of $O(\Delta x^2)$, there is some discrepancy from the line but the shape it takes is still very similar as can be seen in Fig. 5b. For our sinusoidal forcing function case, the errors were not as consistent with our error of order $O(\Delta t^2)$ or our error of order $O(\Delta x^2)$ lines. A good example of this would be the errors of aluminum with the sinusoidal forcing function as seen in Figures 6a and 6b.

Comparing the maximum error throughout all the materials in the two cases for different time steps and spatial steps, we can see that the errors produced by our sinusoidal forcing function are much larger than those of our no forcing function situation. In Fig. 7a, the maximum error throughout our materials is all below a value of 1, and they are relatively consistent amongst themselves, however, if we look at the maximum error for our sinusoidal forcing function, seen in Fig. 7b, we can see that the errors are much larger, as if they have just been scaled up my a constant factor. It appears that the maximum error when taking the materials at different spatial steps also follows a similar trend as seen in Fig.8.



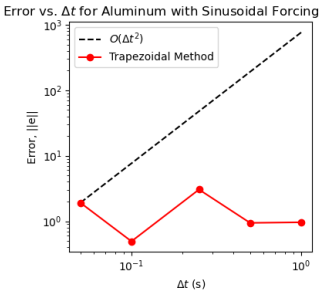
(a) Error vs. Different Time Steps



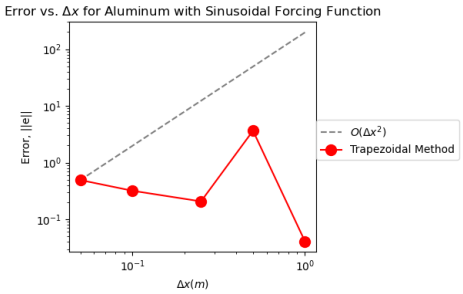
(b) Error vs. Different Spacial Steps

Fig. 5 Displacement for Aluminum With No Forcing Function

We believe there exists inconsistency between these graphs due to the size of our time and spatial steps becoming

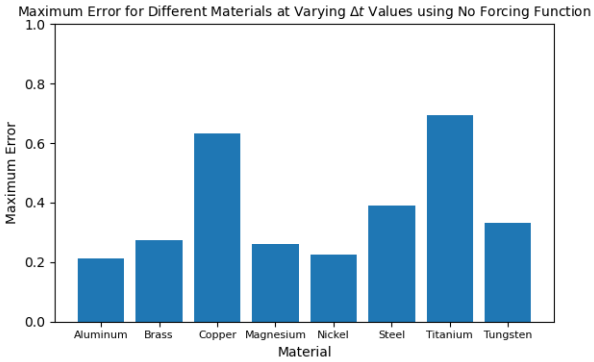


(a) Error vs. Different Time Steps

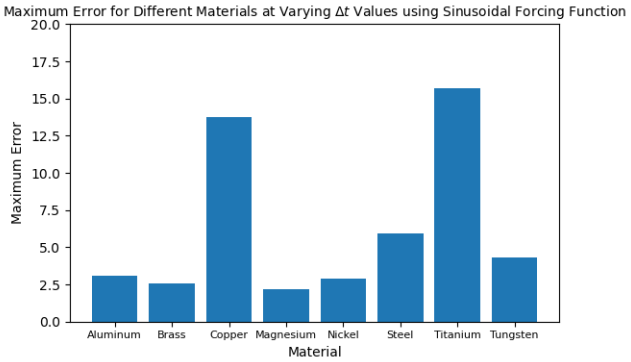


(b) Error vs. Different Spacial Steps

Fig. 6 Displacement for Aluminum With Sinusoidal Forcing Function



(a) The Error for Materials with No Forcing Function



(b) The Error for Materials with Sinusoidal Forcing Function

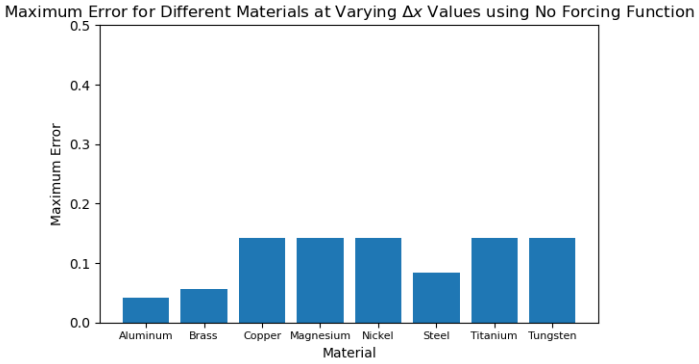
Fig. 7 Maximum Errors for Each Material Using Different Time Steps

larger. Since we are using an implicit method, as the time and spatial steps get much larger, the accuracy of our displacements will intuitively decrease. In addition to this, another factor that may be affecting the error graphs is noise in our system. It is very much possible that the discrepancies that we see in our error graphs may be a result of noise. This would help explain some of the odd phenomena we witness in our error plots.

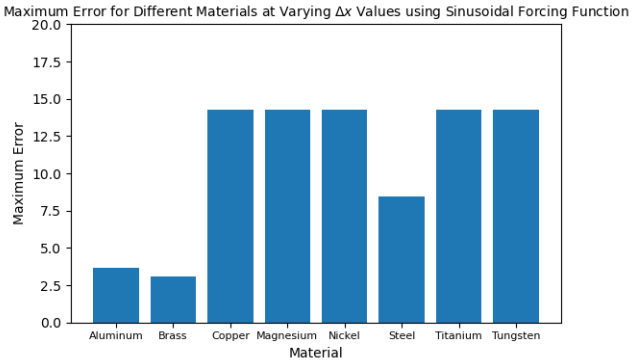
IV. Conclusion

Through our experimentation with numerical methods, we have seen that none of our methods have consistently scaling errors, although the trapezoid method with no forcing function is an exception. We have seen it is, in fact, difficult to accurately approximate fourth order partial differential equations, even if they are simplified and assumed to be second order. The trapezoidal method struggles to provide an accurate approximation of the wave equation, although it is likely that a part of this is due to random errors such as noise. With further time and resources, we could improve the FDTD method such that the approximation results in absolute convergence, since the current method results in diverging results when the constant C is higher than 1. Overall, this report incorporated the use of numerical methods in the evaluation of engineering beam behaviors. It helped to serve as a challenging approach that can be further analyzed and applied to understanding future engineering applications.

Appendix



(a) The Error for Materials with No Forcing Function



(b) The Error for Materials with Sinusoidal Forcing Function

Fig. 8 Maximum Errors for Each Material Using Different Spatial Steps

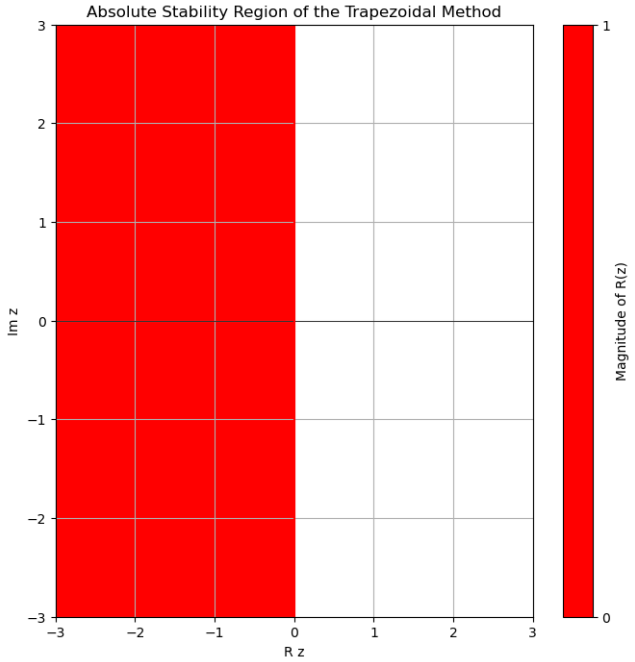


Fig. 9 Trapezoidal Method Absolute Convergence

References

[1] Wikipedia, "Solving the wave equation in 1D," *https://wiki.seg.org/wiki/Solving$_t$he$_w$ave$_e$quation$_i$n$_1$D*, $AccessedDecember6$, 2023.

[2] Çankaya Üniversitesi, "MSE 225 - INTRODUCTION TO MATERIALS SCIENCE RECITATION 5: Mechanical Properties of Materials," *http://tiny.cc/c2ugvz*, Accessed December 1, 2023.

[3] Langtangen, H. P., "Finite Difference Methods for Wave Motion," *https://hplgit.github.io/num-methods-for-PDEs/doc/pub/wave/pdf/wave-4print-A4-2up.pdf*, 2016.