Euler-Bernoulli Beam Theory

Jacob Wicher, Alexander Bonacci

MAT 460 Numerical Differential Equations – Spring 2022, Department of Mathematics & Physics

Motivation

The Euler-Bernoulli beam theory provides a basic means of calculating the deflection of a beam when a lateral load is applied to it. This theory was developed around 1750 by Leonhard Euler and Daniel Bernoulli, two very prominent mathematicians. This theory was first used on a large scale in the construction of the Eiffel Tower and the original Ferris Wheel. Today, this theory has many applications in both the civil and mechanical engineering fields. As aspiring engineers, it is our goal to use the Euler-Bernoulli beam theory to model the deflection and dynamics of a beam in Python.

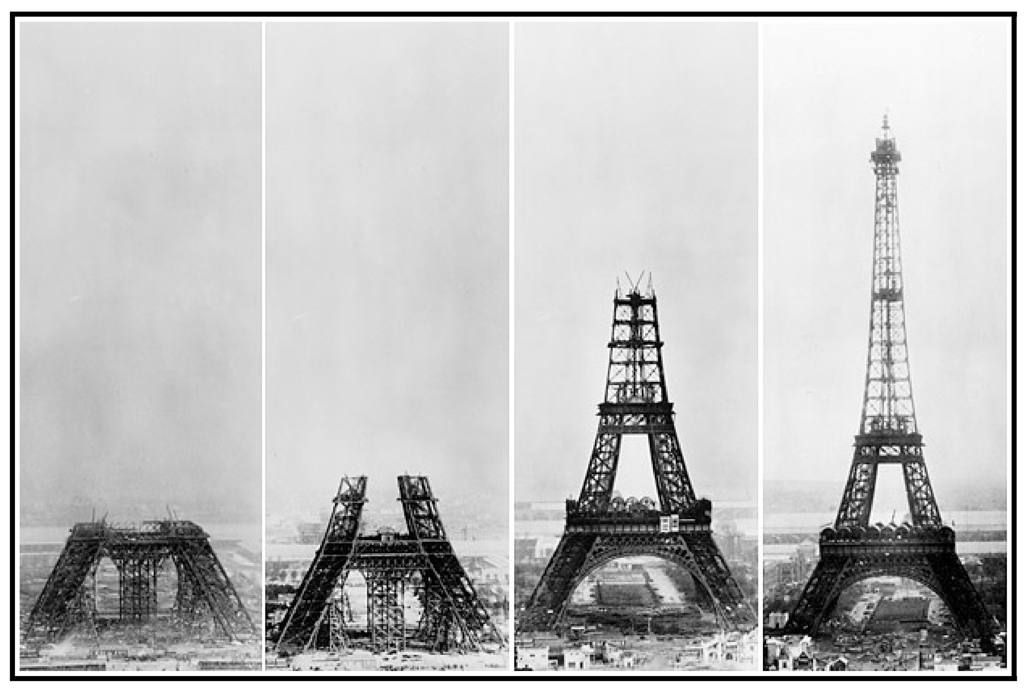

Fig. 1: The construction of the Eiffel Tower

Model

The Euler-Bernoulli beam theory describes the relationship between the beam's deflection and the applied load:

$$\mu\ddot{w}(t) + (EIw'')''(x) = q(x), x \in [0, l]$$

Where:

μ	The mass per unit length of the beam
$w = w(x)$	The vertical displacement of the beam at position x
E	Young's Modulus, a physical property of the material
$I = I(x)$	Moment of inertia of the beam
$q(x)$	Applied load distribution

This is a fourth order differential equation. Since this equation can not be easily solved in this form, a spacial discretization will be performed on the $(EIw'')''(x)$ term, making it a second order differential equation.

More notation:

$M(w) = EIw''(x)$	Bending moment at position x
$Q(w) = -(EIw'')'(x)$	Shear force at position x

Discretization of $(EIw'')''$

In order to reduce the order of the differential equation, the Taylor Series expansion for the second derivative was used and substituted in. The resulting term was then vectorized, resulting in a term Sw. The boundary conditions for a cantilever beam were then used, i.e. the displacement and slope at the fixed end is zero and the shear and bending moment at the free end is also zero. Applying these boundary conditions to Sw yielded the vector $S'w'$, shown below.

$$(EIw'')'' = S'w' = \frac{EI}{h^4}\begin{bmatrix} 6 & -4 & 1 & & & & \\ -4 & \ddots & \ddots & \ddots & & & \\ 1 & \ddots & \ddots & \ddots & \ddots & & \\ & \ddots & \ddots & \ddots & \ddots & 1 & \\ & & \ddots & \ddots & 6 & -4 & 1 \\ & & & \ddots & -4 & 5 & -2 \\ & & & & 1 & -2 & 1 \end{bmatrix}\begin{bmatrix} w_2 \\ w_3 \\ \vdots \\ \vdots \\ \vdots \\ w_{n-3} \\ w_{n-2} \end{bmatrix}$$

After discretization and application of boundary conditions, the relation becomes $\mu\ddot{w} + S'w' = q(x)$, a second order differential equation. The vector w includes the vertical displacements of all the distance elements along the beam.

Test Case

In order to test the accuracy of the model, it was plotted against the accepted equation for vertical displacement using a uniformly distributed load. The accepted equation is:

$$w(x) = \frac{-qx^2}{24EI}(x^2 - 4Lx + 6L^2)$$

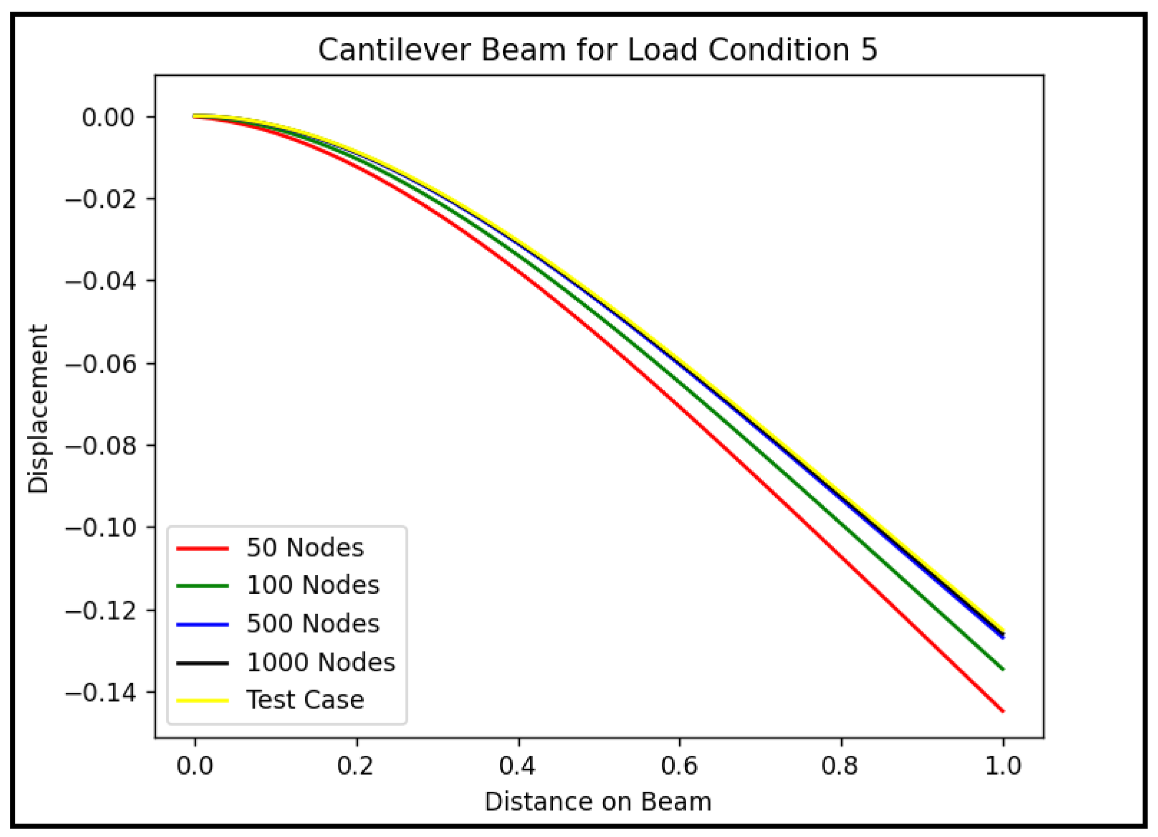

Fig. 2: Accepted Values for w compared with the model.

As shown in the figure, when the beam was split into more elements, the vertical displacement for the model approached the accepted value.

Numerical Methods

In order to model a dynamic case, a reduction of order on the Euler explicit method for differential equations was used. Below is the iterative sequence that was used in order to solve for the displacement of the beam over a specified time interval t.

$$\dot{w}_j = \dot{w}_{j-1} + \Delta t\ddot{w}_{j-1}$$
$$w_j = w_{j-1} + \Delta t\dot{w}_{j-1}$$
$$\ddot{w}_j = \frac{q_j - (S'w')_j}{\mu}$$

Results

Given an initial loading, the iteration will store the positions of the beam for every time element and plot them. The plots were used to create an animation of the beam oscillating due to the initial load conditions. Below is a few snapshots of the animation that was created from the beam.

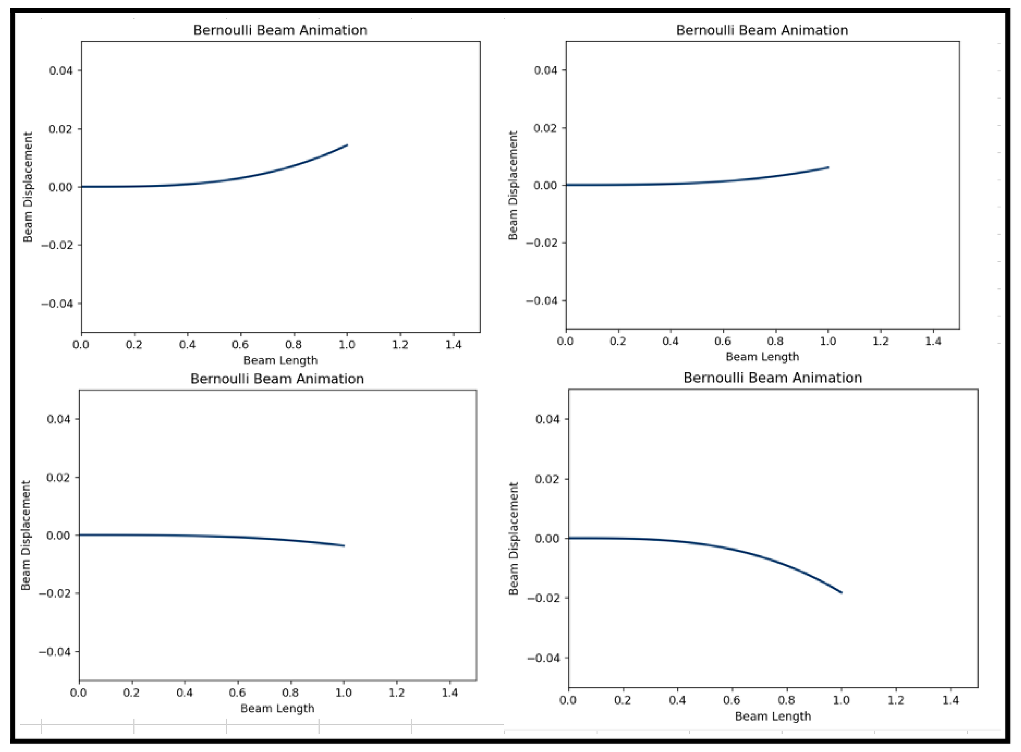

Fig. 3: The oscillating beam resulting from an initial load.

Outlook and Remarks

The Euler-Bernoulli Beam theory is modeled fairly accurately using python. It is still a work in progress, however using other numerical methods such as Runge-Kutta and the Symplectic Euler method can provide even more accurate results and animations. It is easy to see how helpful these animations and calculations can become when dealing with increasingly complicated loading conditions and situations.