489 Homework 1: Machine Learning Applied to Minimal Mass Structures

Coby Arnold

February 13, 2024

Contents

1 Introduction

1.1 Credits

The processes utilized throughout my assignment are a culmination of techniques and mathematics based on lectures given by Dr. Raktim Bhattacharya. Furthermore, a paper titled Minimal Mass Tensegrity Structures written by Kenji Nagase and Robert E. Skelton laid the groundwork for a practical application of these techniques. Python code was written and tested on the Google Colaboratory software

1.2 Problem Identification

The goal of this homework is to learn more about optimization problems and how to employ techniques learned in AERO 489 to solve simple problems. In the case of homework 1, we were tasked with applying convex optimization techniques to find a minimal mass truss structure given an applied load and fixed nodes. The following paper provides documentation to explain my thought process and approach as well as the hurdles I needed to overcome to debug my code.

2 Principle Bases

2.1 Approach

Learned in the aforementioned AERO 489 class, my approach to the problem was to define a system of equations to model the force densities of members in a truss system. By defining the problem this way, the CVXPY library could be used to perform a convex optimization of a system with infinitely many solutions to minimize a defined cost within certain restraints. Because of these reasons, I broke my code down into a few sections.

1. Parameter Input
2. Mesh Generation
3. Convex Optimization
4. Plotting

2.2 Cost and Constraints

The restraint of the system is the equilibrium condition, that is:

$$\sum F = 0$$

As discussed in the paper, the way we can represent this in Python is by defining a connection matrix for bars and strings, and then setting them equal to the applied loads and reaction forces:

$$B = NC_B^T \quad S = NC_B^T$$
$$S \times diag(\sigma_s) \times C - B \times diag(\sigma_b) \times C = F_{ext} + R_x$$

To quantify cost criteria, the 1 norm of the member's masses was taken, and the CVXPY minimize function was used to find the minimum values of total member mass that still upheld the equilibrium condition. Together, the cost and criteria formed a design space that could be represented and solved in Python.

2.3 Adjustable Parameters

I defined parameters in my Python code to make it more applicable to general truss structures. These values can be edited to change the loads and form of the problem, and I'll make a short summary here.

L: Length of the Bar
W: Width of the Bar
P: Applied Load
Spacing: Equidistant Spacing of Nodes

For testing different cases, I only needed to change these parameters to modify the structure. I recommend that any future user of my code also limit altercations to these parameters to test different cases.

3 Code

3.1 Python Modules Used

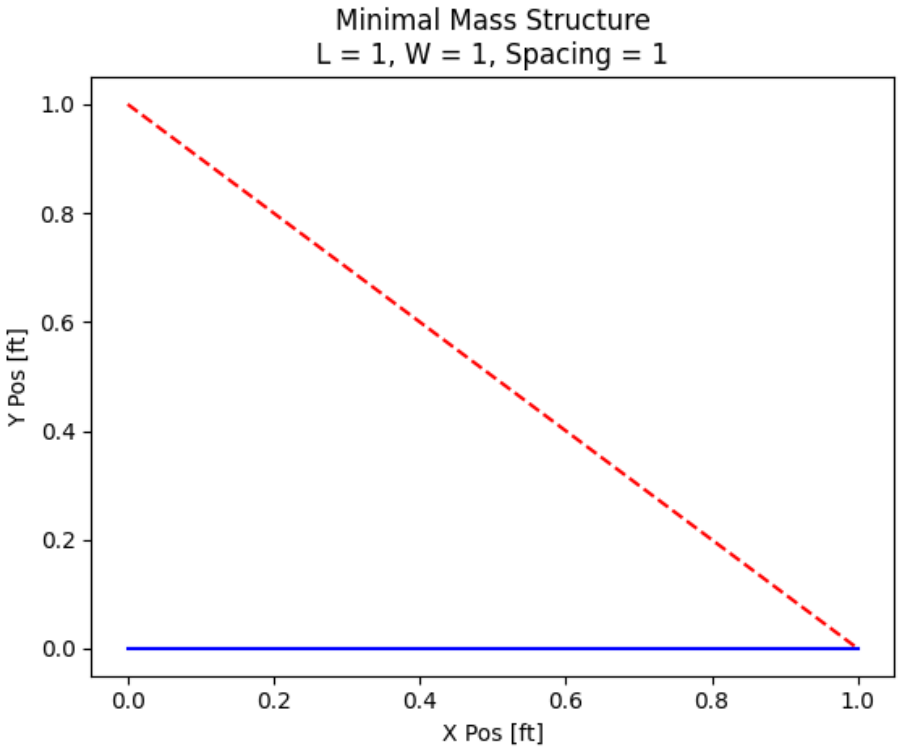
My Python code involved various capabilities of Python modules. Numpy enabled me to easily manipulate arrays and generate matrices. The CVXPY Python library is the tool used to solve the problem, generating variables and performing convex optimization. Matplotlib is used to print results and quickly validate the answers my code produces through visuals.

3.2 Test Cases

Test cases proved difficult to apply since any moderately dense meshed allowed too many possibilities to check. In all plots, **dashed red lines indicate strings** and **blue lines represent bars**. It is also important to note that I have limited my design space to the nearest neighbor node connection. Any given node can only have a member that connects it to its immediate neighbors, which indicates that the minimum found in my design space cannot be the overall minimal mass structure given the parameters.

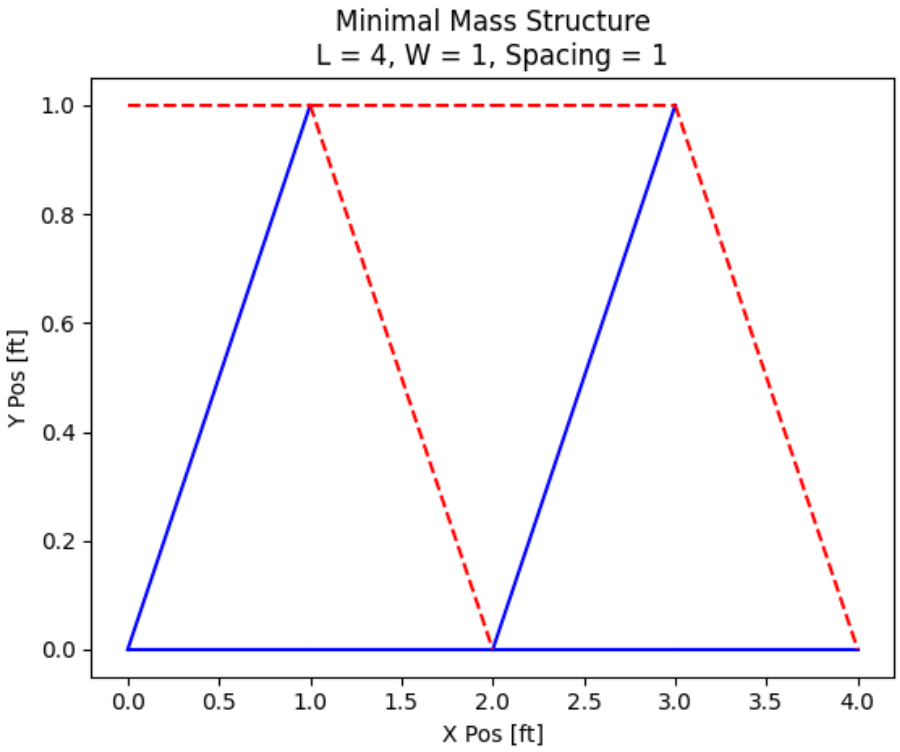
3.2.1 Test Case 1

For my first test case, I chose a very simple structure: a beam of L = 1, W = 1, and Spacing = 1. I applied a load to the bar at x = L, y = 0, and the solution seemed to match intuition.



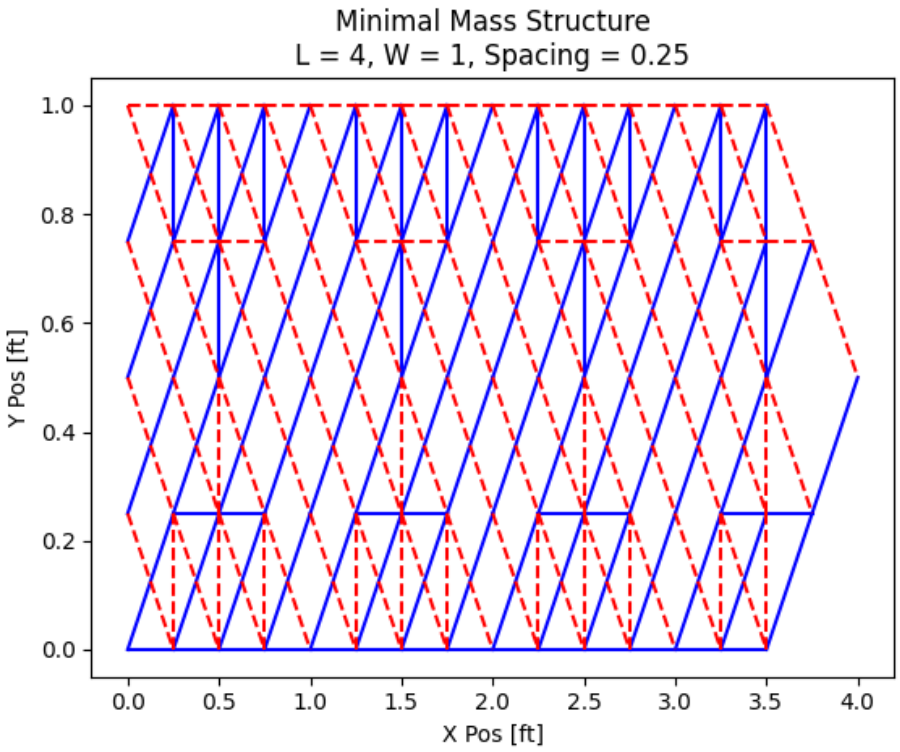
3.2.2 Test Case 2

For my second test case, I chose a longer beam, but still simple spacing of 1 unit: a beam of L = 4, W = 1, and Spacing = 1. I applied a load to the bar at x = L, y = 0, and the solution seemed physically viable meaning that strings didn't seem to be in places of compression and all beams appeared to be in compression.

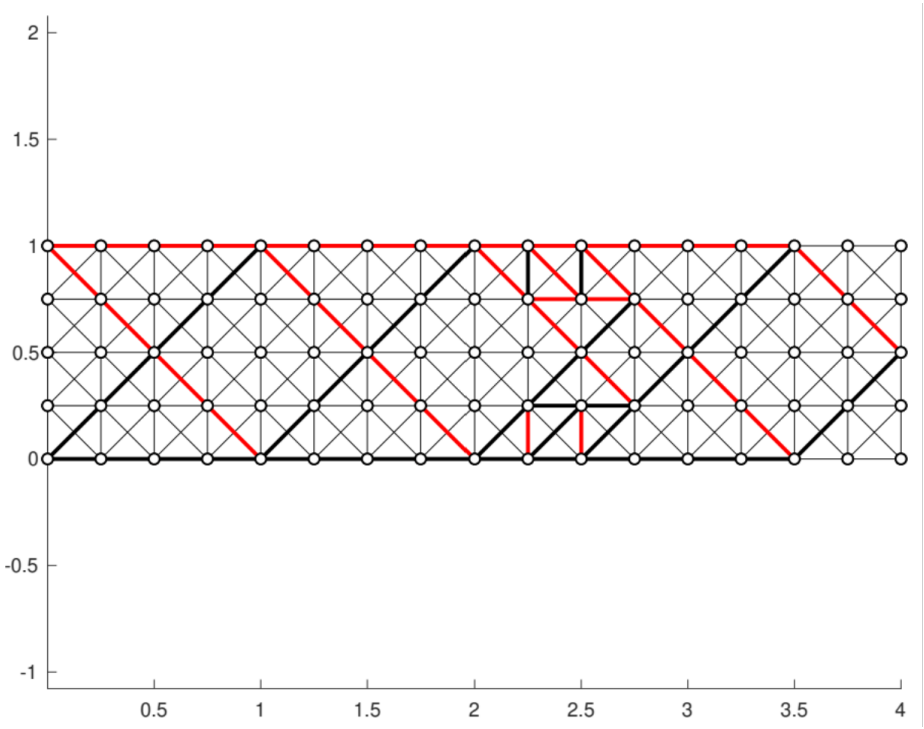


3.2.3 Test Case 3

Unfortunately, my ability to check intuition against a given solution could not extend to test case three. I chose a complicated beam of L = 4, W = 1, and Spacing = 0.25. The force was applied to the center of the beam at x = L, y = $\frac{W}{2}$.


Minimal Mass Structure
L = 4, W = 1, Spacing = 0.25
Y Pos [ft]
X Pos [ft]
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

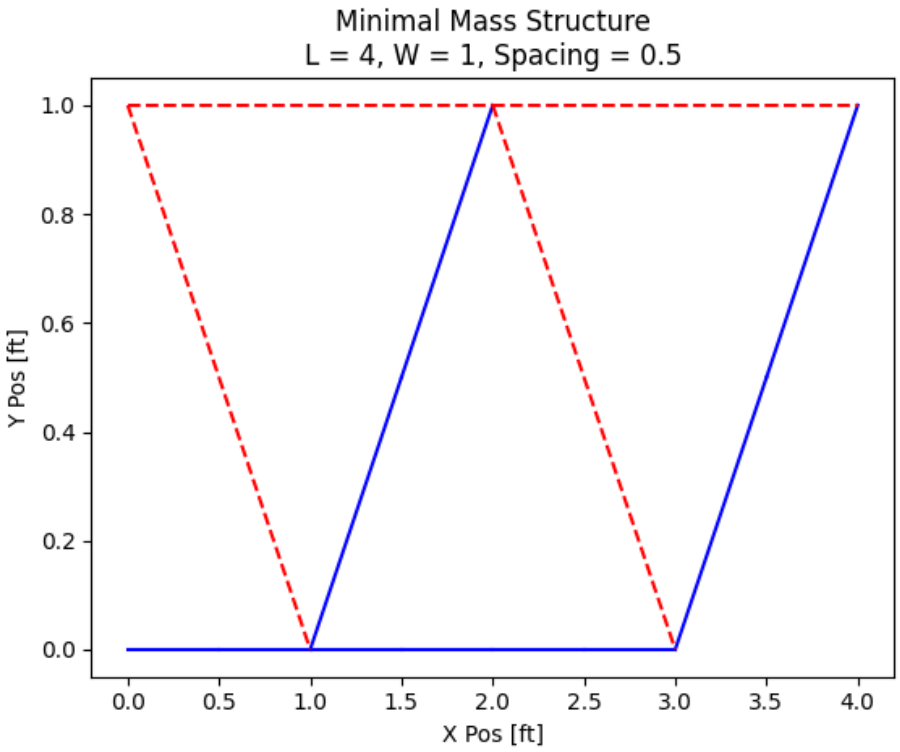
This test case was given to the AERO 489 class by Dr. Bhattacharya to test our code, and the correct configuration can be seen below.



My structure does not align with his for this test case, and for this reason, I think my code is inaccurate for more complicated structures. Specifically when the load is applied to the center of the bar.

3.2.4 Test Case 3

For thoroughness, I applied one more test case with the load applied at X = L, Y = W. I used L = 4, W = 1, Spacing = 0.5. The code seemed to produce a viable solution for this test case as well.



4 Difficulties

4.1 Code Debugging

Most of my time in this project was learning the complexities of CVXPY. When variables are introduced into the equations, certain matrices cannot be checked in a troubleshooting environment. Furthermore, I had a major problem when I tried to define my reaction force matrix. My code initially created a matrix of reaction variables and immediately set all unfixed nodes equal to zero. However, when I defined my constraints for the optimization, the code consistently delivered the wrong solutions. This was only solved by setting unfixed nodes equal to zero inside the constraints function in my code.

4.2 Validation

As previously mentioned, I was unable to check the feasibility of complex solutions by hand, having to rely on intuition to assume whether or not an answer was probably correct. I think more test cases provided to the class would fix this.

5 Citations

1. Bhattacharya, Raktim. Applied Machine Learning for Aerospace Systems, 5 Dec. 2023, isrlab.github.io/AMLAS/.

2. Skelton, Robert E, and Kenji Nagase. Minimal Mass Tensegrity Structures, Mar. 2014, www.researchgate.net/publication/269323292 Minimal Mass Design of Tensegrity Structures.