Optimization of Strike Resistant Tower Crane Truss Structure

Gabriella Garcia

SE 310 Design of Mechanisms and Structures

November 12th, 2025

Table of Contents

1. Introduction

Tower cranes are a lifting equipment, often used on construction sites of high rise buildings. Due to their shape and size, they can easily lift, transport, and lower heavy loads across the construction sites. These cranes are typically made of aluminum or steel truss components because these materials provide the necessary strength to lift the heavy loads they carry.

In structural engineering, a truss is a rigid structure made up of interconnected members that support and distribute loads efficiently. However in the real world, these members are subjected to damage and external forces. These forces or damage to the members can cause them to break, which could potentially cause instability in the rest of the structure. For this reason, truss structures are carefully engineered in a way that if one member fails, the rest of the structure remains stable.

While wanting to make sure the truss remains stable, efficiency of materials is also important. Using a provided objection function, the minimization of the volume will allow for effective use of materials. This project will focus on the truss analysis and design optimization of a tower crane truss structure that can remain stable under different member failure cases and minimize the volume function. This report will outline two designs that meet certain requirements and constraints, clearly showing the design process, structural analysis with results, and final strike resistant and optimization solutions.

2. Problem Description

The first goal of this project is to design an optimized and strike resistant tower crane truss based on a provided baseline geometry. The original truss, shown in Figure 1, has been laid out as a starting point for all design modifications, however the design modifications must meet design and geometric constraints provided.

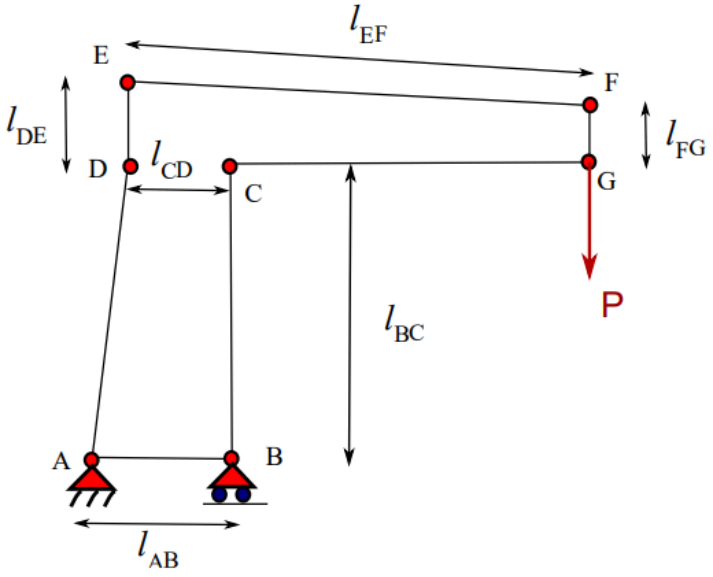


Figure 1: Original cell phone tower truss structure problem setup given in project assignment

A truss structure is considered strike resistant if the failure or damage to any singular truss member does not cause the failure or collapse of the overall truss structure. Keeping this in mind, the new truss structure must maintain global stability even when any one member is removed. To achieve this, new joints and members may be introduced to create a better strike resistant truss, given that all constraints are adhered to.

The second goal of this project is to create a truss that maximizes structural performance, as well as being strike resistant. Maximizing structural performance can be done by minimizing the quality function:

$$C = (1 + n)\sum_{i=1}^{m}(x_i^2 l_i)$$

where n is the number of joints, m is the total number of members, x_i^2 is the cross sectional area of member i, and l_i is the length of member i. By minimizing this function and creating an optimized truss, the resulting structure is one that is strike resistant and efficiently uses material without unnecessary structural complexity.

2.1. Problem Requirements and Constraints

2.1.1. Design Variables:

2.1.1.1. Number of joints, $n > 7$

2.1.1.2. Length of each element l_i

2.1.1.3. Cross sectional area x_i (assume square cross section)

2.1.2. Design Parameters:

2.1.2.1. High strength steel (600 MPa yield strength), Young's Modulus = 200 GPa, and density ρ of 8000 kg/m^3

2.1.2.2. Square cross section, $A_i = x_i^2$

2.1.2.3. Effective length of each truss element is equal to the actual length of the truss, $l_{effective} = l_i$

2.1.3. Boundary Conditions and Loading Requirements:

2.1.3.1. Joint A is a hinge, and Joint B is on a roller

2.1.3.2. The loads acting on the crane is due to the weight of the object it is lifting. The load P = 1.5 x First five digits of your UIN in N.

2.1.3.2.1. P = 1.5 x 65217 = 97825.5 N

2.1.4. Design Constraints:

2.1.4.1. General Constraints:

2.1.4.1.1. Each of the truss elements should satisfy stress constraints, buckling constraints, and the slenderness ratio

2.1.4.1.2. Factor of Safety for stress and buckling: 4 (For each truss, calculate the internal load. If the truss is in compression,

then evaluate the critical buckling load. If the critical buckling load is less than the internal load / safety factor, the truss is assumed to fail)

2.1.4.1.3. Slenderness ratio should be less than or equal to 500 in both tension and compression

2.1.4.1.4. Strike resistance: The truss structure must be stable if an arbitrary truss element is removed from the structure. Specifically, for each removal case, the damaged structure must be able to carry the same load P with all remaining members satisfying stress and buckling constraints with safety factor = 1. If any removed case causes any internal load to exceed yielding/buckling load, the design fails the strike-resistance requirement.

2.1.5. Geometric Constraints:

2.1.5.1. The lengths l_{AB}, l_{BC}, l_{CD}, l_{DE}, l_{EF} and l_{FG} indicate the distance between the joints AB, BC, CD, DE, EF and FG respectively

2.1.5.2. Joints should be separated by at least 2 m

2.1.5.3. $5m \leq l_{AB} \leq 10m$, , $2m \leq l_{FG} \leq 5m$ or l_{FG} =0m

2.1.5.4. $l_{CD} = l_{DE} = 5m$, $l_{BC} = l_{EF} = 30m$

2.1.5.5. No joints must lie outside the polygon ABCGFEDA

2.1.5.6. Truss elements can overlap

3. Quality Factor Calculation

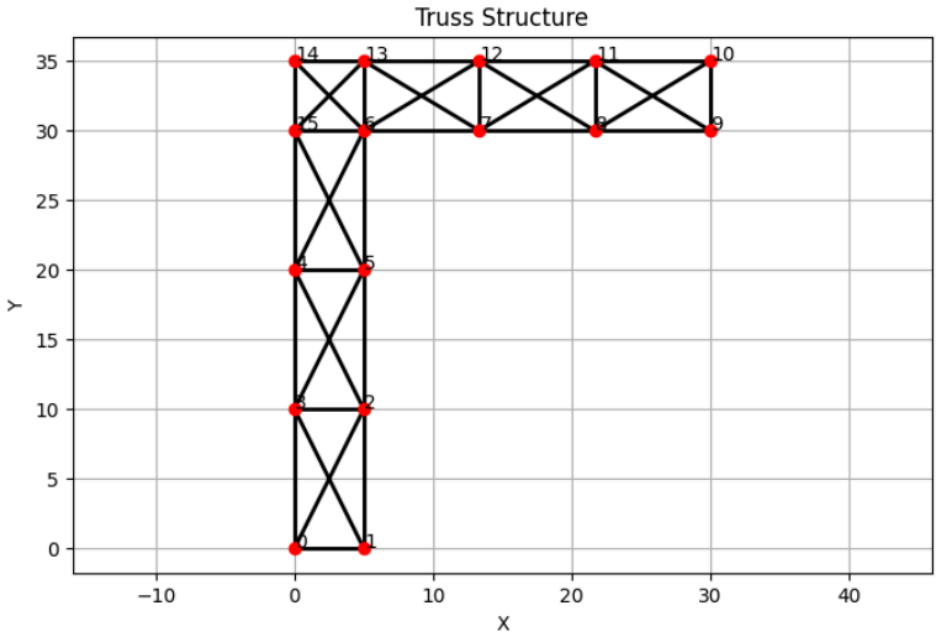
3.1. The quality factor represents a measure of how effective a truss structure is. It penalizes structures that use an excess amount of members and joints, while also accounting for the total volume of the truss, in this case, square truss members. The quality factor C is calculated from the number of joints and members, the cross sectional area of the square members, and their total lengths. While this calculates the material usage, it also represents the tradeoff between structural stability and efficiency of a truss design.

4. Strike Resistant and Damage Case Testing

4.1. To evaluate strike resistance, the truss structures were tested under all damage scenarios of a single member breaking. To test these different scenarios, a member was removed and the truss was reanalyzed with the same loading conditions. For each case, the global stiffness matrix was reassembled and all members were checked for stress, buckling, and slenderness ratio violations. A design passes if all cases pass the requirements of stability and no member fails the stress, buckling, or slenderness ratio test.

5. Initial Design Process

5.1. Truss Design



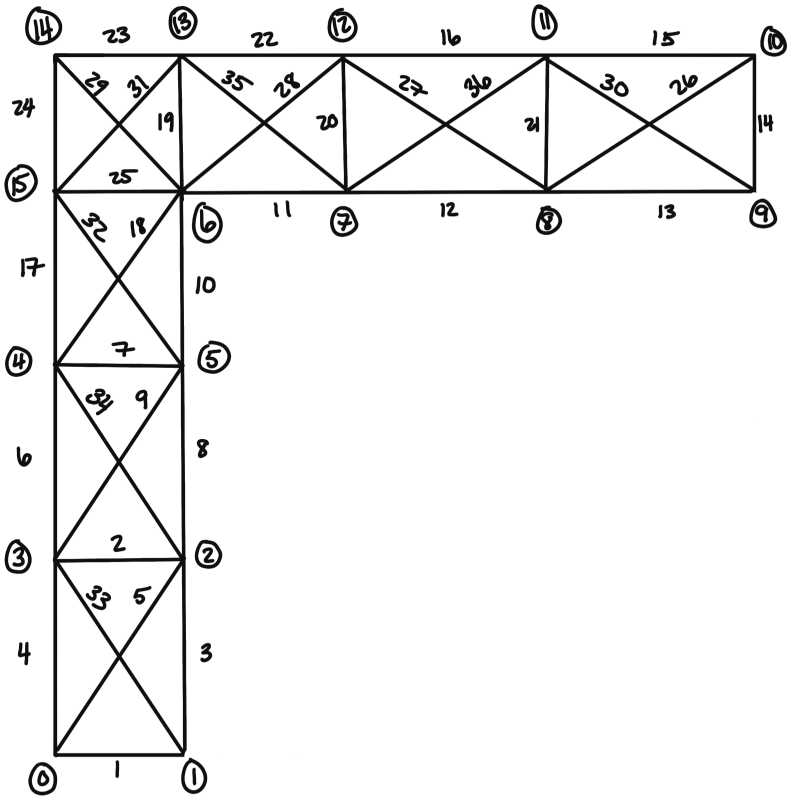


Figure 2a & 2b: Initial truss design with labeled joints and members

Joint 0 is a hinge joint with reaction forces in the x and y direction and Joint 1 is a roller with a reaction force in the y direction. Degree of determinancy: d = b + r - 2*j = 36 + 3 - 2*16 = 7. For this truss design, a cross sectional area of 0.0139 m^2 was used.

5.1.1. Joints and Members Location

5.1.1.1. Joints:

5.1.1.1.1. Chose $l_{AB} = 5m$ and $l_{FG} = 5m$

```
coords = np.array([
        [0.0, 0.0],   #(0)
        [5.0, 0.0],  #(1)
        [5.0, 10.0],  #(2)
        [0.0, 10.0], #(3)
        [0.0, 20.0], #(4)
        [5.0, 20.0],  #(5)
        [5.0, 30.0],  #(6)
        [13.333333, 30.0],  #(7)
        [21.6666667, 30.0],  #(8)
        [30.0, 30.0],  #(9)
        [30.0, 35.0],  #(10)
        [21.6666667, 35.0],  #(11)
        [13.333333, 35.0],  #(12)
        [5.0, 35.0],  #(13)
        [0.0, 35.0],  #(14)
        [0.0, 30.0]  #(15)
    ])
```

5.1.1.2. Members:

```
elements = np.array([
        [0,1], # 1
        [2,3], # 2
        [1,2], # 3
        [0,3], # 4
        [0,2], # 5
        [3,4], # 6
        [4,5], # 7
```

```
        [2,5], # 8
        [3,5], # 9
        [5,6], # 10
        [6,7], # 11
        [7,8], # 12
        [8,9], # 13
        [9,10], # 14
        [10,11], # 15
        [11,12], # 16
        [15,4], # 17
        [4,6], # 18
        [6,13], # 19
        [7,12], # 20
        [8,11], # 21
        [12,13], # 22
        [13,14], # 23
        [14,15], # 24
        [15,6], # 25
        [8,10], # 26
        [12,8], # 27
        [12,6], # 28
        [14,6], # 29
        [9,11], # 30
        [15,13], # 31
        [15,5], # 32
        [1,3], # 33
        [2,4], # 34
        [13,7], # 35
        [7,11] # 36
    ])
```

5.2. Python

5.2.1. Imports and Defining Variables:

```
import numpy as np
import pandas as pd
import os

E = 200e9             # Young's modulus (Pa)
rho = 8000            # density (kg/m3)
```

```
sigma_y = 600e6      # yield strength (Pa)
FOS_stress = 4.0     # safety factor for stress &
buckling in normal checks
FOS_buckling = 4.0   # FOS for buckling in normal
checks
K_euler = 1.0        # Effective Euler factor for
pinned-pinned
```

5.2.2. Element Stiffness

```
def element_stiffness(E, A, x1, y1, x2, y2):
    dx = x2 - x1
    dy = y2 - y1
    L = np.hypot(dx, dy)
    c = dx / L
    s = dy / L
    k_local = (E * A / L) * np.array([[ 1, -1],[-1,
1]])

    T = np.array([[ c, s, 0, 0],
                  [ 0, 0, c, s]])

    k = (E * A / L) * np.array([
        [ c*c, c*s, -c*c, -c*s],
        [ c*s, s*s, -c*s, -s*s],
        [-c*c,-c*s,  c*c,  c*s],
        [-c*s,-s*s,  c*s,  s*s]
    ])
    return k, L
```

5.2.3. Global Stiffness Matrix

```
def assemble_global(nnodes, elements, E_arr, A_arr,
coords):
    NDOF = nnodes * 2
    K = np.zeros((NDOF, NDOF))
    L_elems = np.zeros(len(elements))
    for ei, (n1, n2) in enumerate(elements):
        x1, y1 = coords[n1]
        x2, y2 = coords[n2]
        k_e, L = element_stiffness(E_arr[ei],
A_arr[ei], x1, y1, x2, y2)
```

```
            L_elems[ei] = L
            dof = np.array([2*n1, 2*n1+1, 2*n2,
2*n2+1])
            for i in range(4):
                for j in range(4):
                    K[dof[i], dof[j]] += k_e[i,j]
    return K, L_elems
```

5.2.4. Boundary Conditions and Reaction Forces

```
def solve_truss(K, F, known_disp):
    N = K.shape[0]
    u = np.zeros((N,))
    free_dofs = np.array([i for i in range(N) if i
not in known_disp])
    known_dofs =
np.array(sorted(list(known_disp.keys())))

    Kff = K[np.ix_(free_dofs, free_dofs)]
    Kfn = K[np.ix_(free_dofs, known_dofs)]
    Knf = K[np.ix_(known_dofs, free_dofs)]
    Knn = K[np.ix_(known_dofs, known_dofs)]
    Ff = F[free_dofs]
    Fn = F[known_dofs]
    un = np.array([known_disp[d] for d in
known_dofs])

    if Kff.size == 0:
        uu = np.array([])
    else:
        uu = np.linalg.solve(Kff, Ff - Kfn.dot(un))
    u[free_dofs] = uu
    u[known_dofs] = un

    Rn = Knn.dot(un) + Knf.dot(uu) - Fn if
known_dofs.size>0 else np.array([])
    R = np.zeros_like(F)
    R[known_dofs] = Rn
    return u, R
```

5.2.5. Axial Force Calculation

```
def member_axial_force(ei, elements, coords, E_arr,
A_arr, u):
    n1, n2 = elements[ei]
    x1, y1 = coords[n1]
    x2, y2 = coords[n2]
    dx = x2 - x1
    dy = y2 - y1
    L = np.hypot(dx, dy)
    c = dx / L
    s = dy / L
    dof = np.array([2*n1, 2*n1+1, 2*n2, 2*n2+1])
    ue = u[dof]

    Tbar = np.array([-c, -s, c, s])
    N = (E_arr[ei] * A_arr[ei] / L) * Tbar.dot(ue)
    return N, L
```

5.2.6. Stress, Buckling, and Slenderness Ratio

```
def check_member_constraints(N, L, A, x_square,
E_val=E, sigma_y_val=sigma_y,
                             fos_stress=FOS_stress,
fos_buckling=FOS_buckling):
    area = A
    sigma = abs(N) / area
    allowable_stress = sigma_y_val / fos_stress
    stress_ok = sigma <= allowable_stress

    I = (x_square**4) / 12.0

    Pcr = (np.pi**2 * E_val * I) / ( (K_euler *
L)**2 )
    buckling_ok = True
    if N < 0: # compression
        buckling_ok = abs(N) <= Pcr / fos_buckling
    # slenderness ratio: L/r with r = sqrt(I/A) =
x/sqrt(12)
    r = x_square / np.sqrt(12.0)
    slenderness = L / r
    slender_ok = slenderness <= 500.0
    return {
        'sigma': sigma,
```

```
            'allowable_sigma': allowable_stress,
            'stress_ok': stress_ok,
            'Pcr': Pcr,
            'buckling_ok': buckling_ok,
            'slenderness': slenderness,
            'slender_ok': slender_ok
        }
```

5.2.7. Objective Function

```
def objective_C(n_nodes, areas, L_elems):
    # C = n + sum x_i^2 * l_i  where area = x^2 =>
x = sqrt(area)
    x_vals = np.sqrt(areas)
    return n_nodes + np.sum(x_vals**2 * L_elems)
```

5.2.8. Strike Resistance Test

```
def strike_resistance_test(nodes, elements, E_arr,
A_arr, coords, F_global, known_disp,
sigma_y_val=sigma_y):
    nnodes = len(coords)
    results = []
    for ei in range(len(elements)):

        elems_dmg = [elements[j] for j in
range(len(elements)) if j != ei]
        E_dmg = np.delete(E_arr, ei)
        A_dmg = np.delete(A_arr, ei)
        K_dmg, L_elems_dmg =
assemble_global(nnodes, elems_dmg, E_dmg, A_dmg,
coords)

        try:
            u_dmg, R_dmg = solve_truss(K_dmg,
F_global.copy(), known_disp)
        except np.linalg.LinAlgError:
            results.append({'element_removed': ei,
'pass': False, 'reason': 'unstable / singular
stiffness matrix'})
            continue

        bad = False
```

```
            details = []
            for ej in range(len(elems_dmg)):
                Nj, Lj = member_axial_force(ej,
elems_dmg, coords, E_dmg, A_dmg, u_dmg)
                x_sq = np.sqrt(A_dmg[ej])
                chk = check_member_constraints(Nj, Lj,
A_dmg[ej], x_sq, fos_stress=1.0, fos_buckling=1.0)
                details.append({'ej': ej, 'N': Nj, 'L':
Lj, 'check': chk})
                if not (chk['stress_ok'] and
chk['slender_ok'] and chk['buckling_ok']):
                    bad = True
            if bad:
                results.append({'element_removed': ei,
'pass': False, 'reason': 'some member fails under
FS=1', 'details': details})
            else:
                results.append({'element_removed': ei,
'pass': True, 'details': details})
        return results
```

5.2.9. Force P and Print

```
if __name__ == "__main__":
    coords = np.array([
        [0.0, 0.0],   #(0)
        [5.0, 0.0],  #(1)
        [5.0, 10.0],  #(2)
        [0.0, 10.0], #(3)
        [0.0, 20.0], #(4)
        [5.0, 20.0],  #(5)
        [5.0, 30.0],  #(6)
        [13.333333, 30.0],  #(7)
        [21.6666667, 30.0],  #(8)
        [30.0, 30.0],  #(9)
        [30.0, 35.0],  #(10)
        [21.6666667, 35.0],  #(11)
        [13.333333, 35.0],  #(12)
        [5.0, 35.0],  #(13)
        [0.0, 35.0],  #(14)
        [0.0, 30.0]  #(15)
    ])
```

```
elements = np.array([
    [0,1], # 1
    [2,3], # 2
    [1,2], # 3
    [0,3], # 4
    [0,2], # 5
    [3,4], # 6
    [4,5], # 7
    [2,5], # 8
    [3,5], # 9
    [5,6], # 10
    [6,7], # 11
    [7,8], # 12
    [8,9], # 13
    [9,10], # 14
    [10,11], # 15
    [11,12], # 16
    [15,4], # 17
    [4,6], # 18
    [6,13], # 19
    [7,12], # 20
    [8,11], # 21
    [12,13], # 22
    [13,14], # 23
    [14,15], # 24
    [15,6], # 25
    [8,10], # 26
    [12,8], # 27
    [12,6], # 28
    [14,6], # 29
    [9,11], # 30
    [15,13], # 31
    [15,5], # 32
    [1,3], # 33
    [2,4], # 34
    [13,7], # 35
    [7,11] # 36
])
```

```
    elements = np.array(elements)
    nnodes = coords.shape[0]
    nelems = elements.shape[0]
    Area = np.full(nelems, 0.0139)
    E_arr = np.full(nelems, E)
    K_global, L_elems = assemble_global(nnodes,
elements, E_arr, Area, coords)

    NDOF = nnodes * 2
    F = np.zeros((NDOF,))

    P = 97825.5
    F[2*6 + 1] = -P

    known_disp = { 2*0: 0.0, 2*0+1: 0.0, 2*1+1: 0.0
}

    u, reactions = solve_truss(K_global, F,
known_disp)

    member_results = []
    for ei in range(nelems):
        Nval, Lval = member_axial_force(ei,
elements, coords, E_arr, Area, u)
        x_side = np.sqrt(Area[ei])
        chk = check_member_constraints(Nval, Lval,
Area[ei], x_side)
        member_results.append({
            'element': ei,
            'nodes': elements[ei],
            'length': Lval,
            'N': Nval,
            'sigma': chk['sigma'],
            'stress_ok': chk['stress_ok'],
            'Pcr': chk['Pcr'],
            'buckling_ok': chk['buckling_ok'],
            'slenderness': chk['slenderness'],
            'slender_ok': chk['slender_ok']
        })
```

```
    df_members = pd.DataFrame(member_results)
    print("Member results:")
    print(df_members)

    Cval = objective_C(nnodes, Area, L_elems)
    Cprime = Cval / abs(P)
    print(f"\nObjective C = {Cval:.4f}, C' =
{Cprime:.6f}")

    # Strike resistance test
    strike_results = strike_resistance_test(nnodes,
elements, E_arr, Area, coords, F, known_disp)
    for r in strike_results:
        print(f"Removed element
{r['element_removed']} -> pass = {r['pass']};
reason = {r.get('reason','ok')}")
```

5.2.10. Displacement

```
print("\nJoint Displacements:")
    disp_data = []

    for i in range(nnodes):
        ux = u[2*i]
        uy = u[2*i + 1]
        disp_data.append({
            'node': i,
            'ux (m)': ux,
            'uy (m)': uy
        })

    df_disp = pd.DataFrame(disp_data)
    print(df_disp)
```

5.3. Stress, Buckling, and Slenderness Results

5.3.1. Stress: >internal load / safety factor = True

	element	nodes	length	N	sigma	stress_ok
0	0	[0, 1]	5.000000	9466.554495	6.810471e+05	True
1	1	[2, 3]	5.000000	18691.979941	1.344747e+06	True
2	2	[1, 2]	10.000000	-78892.391010	5.675712e+06	True
3	3	[0, 3]	10.000000	18933.108990	1.362094e+06	True
4	4	[0, 2]	11.180340	-21167.859364	1.522868e+06	True
5	5	[3, 4]	10.000000	18450.850892	1.327399e+06	True
6	6	[4, 5]	5.000000	18716.428736	1.346506e+06	True
7	7	[2, 5]	10.000000	-79374.649108	5.710406e+06	True
8	8	[3, 5]	11.180340	-20628.678419	1.484078e+06	True
9	9	[5, 6]	10.000000	-78843.493421	5.672194e+06	True
10	10	[6, 7]	8.333333	63.483961	4.567191e+03	True
11	11	[7, 8]	8.333334	-2.450543	1.762981e+02	True
12	12	[8, 9]	8.333333	0.094453	6.795169e+00	True
13	13	[9, 10]	5.000000	0.056672	4.077101e+00	True
14	14	[10, 11]	8.333333	0.094453	6.795169e+00	True
15	15	[11, 12]	8.333334	-2.450543	1.762981e+02	True
16	16	[15, 4]	10.000000	18982.006579	1.365612e+06	True
17	17	[4, 6]	11.180340	-21222.528530	1.526801e+06	True
18	18	[6, 13]	5.000000	-948.679580	6.825033e+04	True
19	19	[7, 12]	5.000000	36.620052	2.634536e+03	True
20	20	[8, 11]	5.000000	-1.413654	1.017017e+02	True
21	21	[12, 13]	8.333333	63.483961	4.567191e+03	True
22	22	[13, 14]	5.000000	-986.769959	7.099064e+04	True
23	23	[14, 15]	5.000000	-986.769959	7.099064e+04	True
24	24	[15, 6]	5.000000	8504.233331	6.118153e+05	True
25	25	[8, 10]	9.718253	-0.110150	7.924460e+00	True
26	26	[12, 8]	9.718253	2.857800	2.055971e+02	True
27	27	[12, 6]	9.718253	-74.034386	5.326215e+03	True
28	28	[14, 6]	7.071068	1395.503458	1.003959e+05	True
29	29	[9, 11]	9.718253	-0.110150	7.924461e+00	True
30	30	[15, 13]	7.071068	1395.503458	1.003959e+05	True
31	31	[15, 5]	11.180340	-21222.528530	1.526801e+06	True
32	32	[1, 3]	11.180340	-21167.859364	1.522868e+06	True
33	33	[2, 4]	11.180340	-20628.678419	1.484078e+06	True
34	34	[13, 7]	9.718253	-74.034386	5.326215e+03	True
35	35	[7, 11]	9.718253	2.857800	2.055971e+02	True

Figure 3: Results of stress testing

5.3.2. Buckling: >internal load / safety factor = True

	Pcr	buckling_ok
0	1.271271e+06	True
1	1.271271e+06	True
2	3.178177e+05	True
3	3.178177e+05	True
4	2.542542e+05	True
5	3.178177e+05	True
6	1.271271e+06	True
7	3.178177e+05	True
8	2.542542e+05	True
9	3.178177e+05	True
10	4.576575e+05	True
11	4.576575e+05	True
12	4.576575e+05	True
13	1.271271e+06	True
14	4.576575e+05	True
15	4.576575e+05	True
16	3.178177e+05	True
17	2.542542e+05	True
18	1.271271e+06	True
19	1.271271e+06	True
20	1.271271e+06	True
21	4.576575e+05	True
22	1.271271e+06	True
23	1.271271e+06	True
24	1.271271e+06	True
25	3.365129e+05	True
26	3.365128e+05	True
27	3.365129e+05	True
28	6.356354e+05	True
29	3.365129e+05	True
30	6.356354e+05	True
31	2.542542e+05	True
32	2.542542e+05	True
33	2.542542e+05	True
34	3.365129e+05	True
35	3.365128e+05	True

Figure 4: Results of buckling testing

5.3.3. Slenderness: <500 = True

slenderness	slender_ok
146.910632	True
146.910632	True
293.821264	True
293.821264	True
328.502160	True
293.821264	True
146.910632	True
293.821264	True
328.502160	True
293.821264	True
244.851044	True
244.851064	True
244.851052	True
146.910632	True
244.851052	True
244.851064	True
293.821264	True
328.502160	True
146.910632	True
146.910632	True
146.910632	True
244.851044	True
146.910632	True
146.910632	True
146.910632	True
285.542942	True
285.542952	True
285.542934	True
207.763008	True
285.542942	True
207.763008	True
328.502160	True
328.502160	True
328.502160	True
285.542934	True
285.542952	True

Figure 5: Results of slenderness ratio testing

5.4. Quality and Normalized Quality Function

5.4.1. `C = 20.1635, C' = 0.000206`

5.5. Strike Resistance Results

5.5.1. Strike Resistance passes if when a member is removed, stress, buckling, and the slenderness ratio still hold True without that member. When all of these tests are true and every member case passes the strike resistance test, the truss solution is feasible and strike resistant.

```
Removed element 0 -> pass = True; reason = ok
Removed element 1 -> pass = True; reason = ok
Removed element 2 -> pass = True; reason = ok
Removed element 3 -> pass = True; reason = ok
Removed element 4 -> pass = True; reason = ok
Removed element 5 -> pass = True; reason = ok
Removed element 6 -> pass = True; reason = ok
Removed element 7 -> pass = True; reason = ok
Removed element 8 -> pass = True; reason = ok
Removed element 9 -> pass = True; reason = ok
Removed element 10 -> pass = True; reason = ok
Removed element 11 -> pass = True; reason = ok
Removed element 12 -> pass = True; reason = ok
Removed element 13 -> pass = True; reason = ok
Removed element 14 -> pass = True; reason = ok
Removed element 15 -> pass = True; reason = ok
Removed element 16 -> pass = True; reason = ok
Removed element 17 -> pass = True; reason = ok
Removed element 18 -> pass = True; reason = ok
Removed element 19 -> pass = True; reason = ok
Removed element 20 -> pass = True; reason = ok
Removed element 21 -> pass = True; reason = ok
Removed element 22 -> pass = True; reason = ok
Removed element 23 -> pass = True; reason = ok
Removed element 24 -> pass = True; reason = ok
Removed element 25 -> pass = True; reason = ok
Removed element 26 -> pass = True; reason = ok
Removed element 27 -> pass = True; reason = ok
Removed element 28 -> pass = True; reason = ok
Removed element 29 -> pass = True; reason = ok
Removed element 30 -> pass = True; reason = ok
Removed element 31 -> pass = True; reason = ok
Removed element 32 -> pass = True; reason = ok
Removed element 33 -> pass = True; reason = ok
Removed element 34 -> pass = True; reason = ok
Removed element 35 -> pass = True; reason = ok
```

Figure 6: Results of strike resistant testing

5.6. Displacement

	node	ux (m)	uy (m)
0	0	0.000000	0.000000
1	1	0.000017	0.000000
2	2	0.000377	-0.000284
3	3	0.000344	0.000068
4	4	0.001399	0.000134
5	5	0.001433	-0.000569
6	6	0.003183	-0.000853
7	7	0.003183	-0.002599
8	8	0.003183	-0.004344
9	9	0.003183	-0.006089
10	10	0.004230	-0.006089
11	11	0.004230	-0.004344
12	12	0.004230	-0.002599

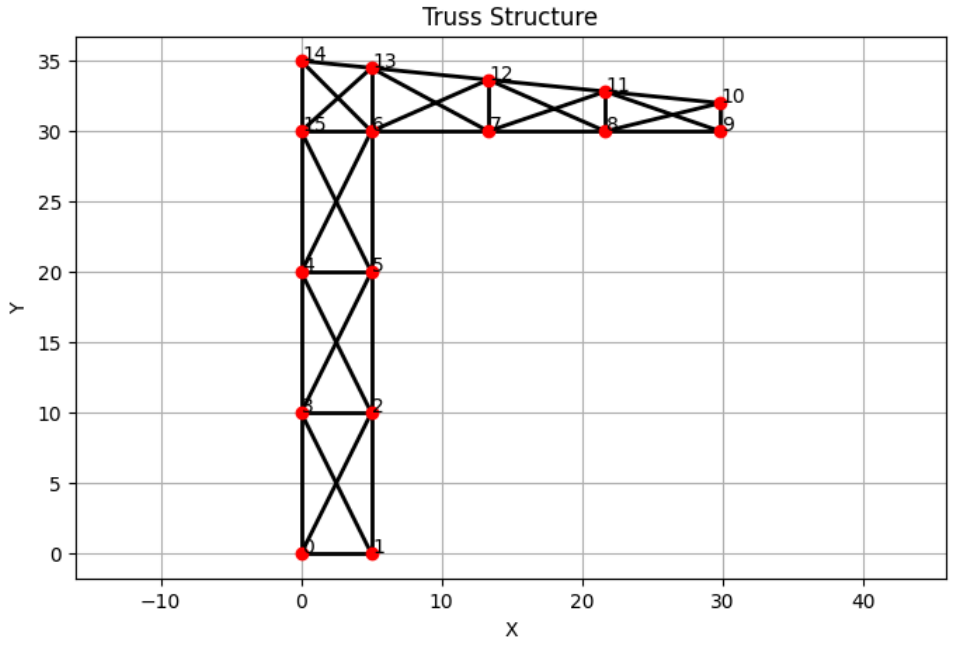
```
13     13   0.004230 -0.000855
14     14   0.004232  0.000201
15     15   0.003168  0.000203
```

6. Final Design Process

6.1. Description

After creating my initial design that met the feasibility criteria and passed all strike resistant cases, I used the design as a foundation to my final design. With each new design, I changed/got rid of/added members and joints to my foundation design and checked for stress, buckling, slenderness, and strike resistance, if all passed, I would check if C was smaller. If modifications did not improve the structure of C value, they were removed and new ones were tested. Through this process, I was able to obtain my final design that meets all requirements and constraints. Some design choses made were to add diagonal members which allowed for more bracing and support, increasing the buckling resistance. These triangular member arrangements allowed for greater resistance to deformation due to stress too. Additionally, vertical/horizontal members were connected to members that had a larger length to create stability and reduce buckling and slenderness.

6.2. Truss Design



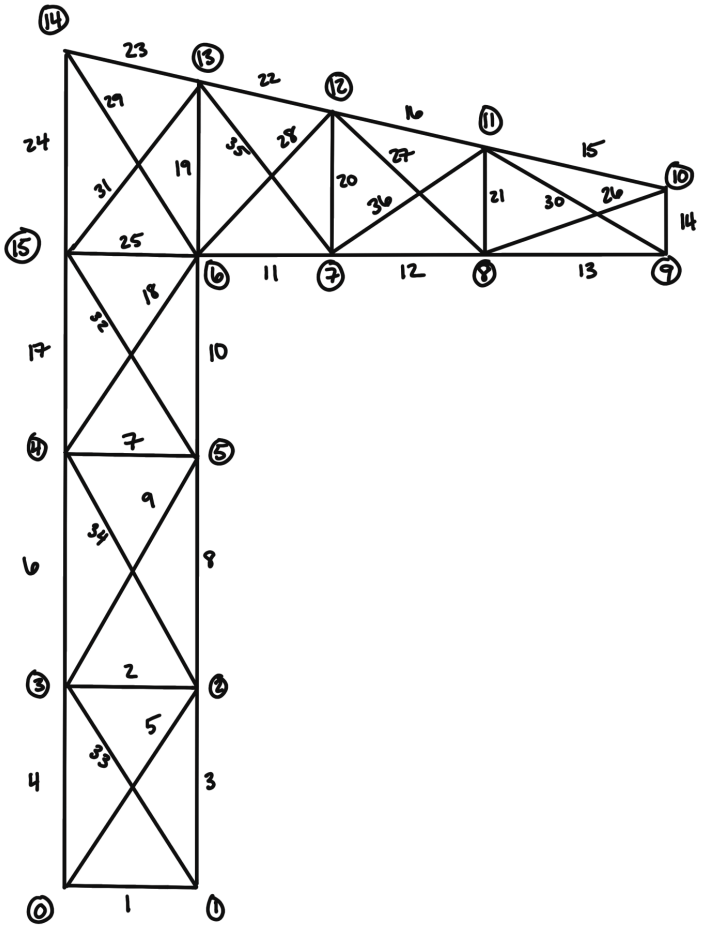


Figure 7a & 7b: Final truss design with labeled joints and members

Joint 0 is a hinge joint with reaction forces in the x and y direction and Joint 1 is a roller with a reaction force in the y direction. Degree of determinancy: d = b + r - 2*j = 36 + 3 - 2*16 = 7. For this truss design, a cross sectional area of 0.0139 m^2 was used.

6.2.1. Joints and Members Location

6.2.1.1. Joints:

6.2.1.1.1. Chose $l_{AB} = 5m$ and $l_{FG} = 2m$

```
coords = np.array([
        [0.0, 0.0],    #(0)
        [5.0, 0.0],   #(1)
        [5.0, 10.0],   #(2)
```

```
        [0.0, 10.0], #(3)
        [0.0, 20.0], #(4)
        [5.0, 20.0],  #(5)
        [5.0, 30.0],  #(6)
        [13.333333, 30.0],  #(7)
        [21.6666667, 30.0],  #(8)
        [29.8496, 30.0],  #(9)
        [29.8496, 32.0],  #(10)
        [21.6666667, 32.8224],  #(11)
        [13.333333, 33.6599],  #(12)
        [5.0, 34.4975],  #(13)
        [0.0, 35.0],  #(14)
        [0.0, 30.0]  #(15)
    ])
```

6.2.1.2. Members:

```
elements = np.array([
        [0,1], # 1
        [2,3], # 2
        [1,2], # 3
        [0,3], # 4
        [0,2], # 5
        [3,4], # 6
        [4,5], # 7
        [2,5], # 8
        [3,5], # 9
        [5,6], # 10
        [6,7], # 11
        [7,8], # 12
        [8,9], # 13
        [9,10], # 14
        [10,11], # 15
        [11,12], # 16
        [15,4], # 17
        [4,6], # 18
        [6,13], # 19
        [7,12], # 20
        [8,11], # 21
        [12,13], # 22
```

```
        [13,14], # 23
        [14,15], # 24
        [15,6], # 25
        [8,10], # 26
        [12,8], # 27
        [12,6], # 28
        [14,6], # 29
        [9,11], # 30
        [15,13], # 31
        [15,5], # 32
        [1,3], # 33
        [2,4], # 34
        [13,7], # 35
        [7,11] # 36
    ])
```

6.3. Python

6.3.1. Code for sections below is the same as Initial Design only node and element arrays are changed:

6.3.1.1. Imports and Defining Variables

6.3.1.2. Element Stiffness

6.3.1.3. Global Stiffness Matrix

6.3.1.4. Boundary Conditions and Reaction Forces

6.3.1.5. Axial Force Calculation

6.3.1.6. Stress, Buckling, and Slenderness Ratio

6.3.1.7. Objective Function

6.3.1.8. Strike Resistance Test

6.3.1.9. Force P and Print

6.3.1.10. Displacement

6.4. Stress, Buckling, and Slenderness Results

6.4.1. Stress: >internal load / safety factor = True

	element	nodes	length	N	sigma	stress_ok
0	0	[0, 1]	5.000000	9466.555590	6.810472e+05	True
1	1	[2, 3]	5.000000	18691.936853	1.344744e+06	True
2	2	[1, 2]	10.000000	-78892.388820	5.675711e+06	True
3	3	[0, 3]	10.000000	18933.111180	1.362094e+06	True
4	4	[0, 2]	11.180340	-21167.861812	1.522868e+06	True
5	5	[3, 4]	10.000000	18450.762526	1.327393e+06	True
6	6	[4, 5]	5.000000	18718.166706	1.346631e+06	True
7	7	[2, 5]	10.000000	-79374.737474	5.710413e+06	True
8	8	[3, 5]	11.180340	-20628.579623	1.484070e+06	True
9	9	[5, 6]	10.000000	-78839.929115	5.671937e+06	True
10	10	[6, 7]	8.333333	50.026302	3.599015e+03	True
11	11	[7, 8]	8.333334	-0.871097	6.266886e+01	True
12	12	[8, 9]	8.182933	0.007432	5.346847e-01	True
13	13	[9, 10]	2.000000	0.002563	1.844197e-01	True
14	14	[10, 11]	8.224156	0.007470	5.373782e-01	True
15	15	[11, 12]	8.375312	-0.875485	6.298455e+01	True
16	16	[15, 4]	10.000000	18985.570885	1.365868e+06	True
17	17	[4, 6]	11.180340	-21226.513545	1.527087e+06	True
18	18	[6, 13]	4.497500	-1036.683719	7.458156e+04	True
19	19	[7, 12]	3.659900	26.704166	1.921163e+03	True
20	20	[8, 11]	2.822400	-0.380759	2.739273e+01	True
21	21	[12, 13]	8.375322	50.278367	3.617149e+03	True
22	22	[13, 14]	5.025187	-1063.987578	7.654587e+04	True
23	23	[14, 15]	5.000000	-952.259877	6.850790e+04	True
24	24	[15, 6]	5.000000	8434.130770	6.067720e+05	True
25	25	[8, 10]	8.423799	-0.007651	5.504232e-01	True
26	26	[12, 8]	9.101611	0.951406	6.844651e+01	True
27	27	[12, 6]	9.101610	-54.638390	3.930819e+03	True
28	28	[14, 6]	7.071068	1497.163795	1.077096e+05	True
29	29	[9, 11]	8.656000	-0.007862	5.655956e-01	True
30	30	[15, 13]	6.725140	1423.920148	1.024403e+05	True
31	31	[15, 5]	11.180340	-21226.513545	1.527087e+06	True
32	32	[1, 3]	11.180340	-21167.861812	1.522868e+06	True
33	33	[2, 4]	11.180340	-20628.579623	1.484070e+06	True
34	34	[13, 7]	9.469527	-56.847054	4.089716e+03	True
35	35	[7, 11]	8.798318	0.919703	6.616566e+01	True

Figure 8: Results of stress testing

6.4.2. Buckling: >internal load / safety factor = True

	Pcr	buckling_ok
0	1.271271e+06	True
1	1.271271e+06	True
2	3.178177e+05	True
3	3.178177e+05	True
4	2.542542e+05	True
5	3.178177e+05	True
6	1.271271e+06	True
7	3.178177e+05	True
8	2.542542e+05	True
9	3.178177e+05	True
10	4.576575e+05	True
11	4.576575e+05	True
12	4.746353e+05	True
13	7.945443e+06	True
14	4.698892e+05	True
15	4.530812e+05	True
16	3.178177e+05	True
17	2.542542e+05	True
18	1.571215e+06	True
19	2.372683e+06	True
20	3.989707e+06	True
21	4.530802e+05	True
22	1.258559e+06	True
23	1.271271e+06	True
24	1.271271e+06	True
25	4.478804e+05	True
26	3.836556e+05	True
27	3.836557e+05	True
28	6.356354e+05	True
29	4.241735e+05	True
30	7.027089e+05	True
31	2.542542e+05	True
32	2.542542e+05	True
33	2.542542e+05	True
34	3.544227e+05	True
35	4.105621e+05	True

Figure 9: Results of buckling testing

6.4.3. Slenderness: <500 = True

slenderness	slender_ok
146.910632	True
146.910632	True
293.821264	True
293.821264	True
328.502160	True
293.821264	True
146.910632	True
293.821264	True
328.502160	True
293.821264	True
244.851044	True
244.851064	True
240.431981	True
58.764253	True
241.643186	True
246.084486	True
293.821264	True
328.502160	True
132.146114	True
107.535644	True
82.928114	True
246.084759	True
147.650685	True
146.910632	True
146.910632	True
247.509141	True
267.424678	True
267.424659	True
207.763008	True
254.331692	True
197.598910	True
328.502160	True
328.502160	True
328.502160	True
278.234845	True
258.513279	True

Figure 10: Results of slenderness ratio testing

6.5. Quality and Normalized Quality Function

6.5.1. `C = 19.9929, C' = 0.000204`

6.6. Strike Resistance Results

6.6.1. Strike Resistance passes if when a member is removed, stress, buckling, and the slenderness ratio still hold True without that member. When all of

these tests are true and every member case passes the strike resistance test, the truss solution is feasible and strike resistant.

```
Removed element 0 -> pass = True; reason = ok
Removed element 1 -> pass = True; reason = ok
Removed element 2 -> pass = True; reason = ok
Removed element 3 -> pass = True; reason = ok
Removed element 4 -> pass = True; reason = ok
Removed element 5 -> pass = True; reason = ok
Removed element 6 -> pass = True; reason = ok
Removed element 7 -> pass = True; reason = ok
Removed element 8 -> pass = True; reason = ok
Removed element 9 -> pass = True; reason = ok
Removed element 10 -> pass = True; reason = ok
Removed element 11 -> pass = True; reason = ok
Removed element 12 -> pass = True; reason = ok
Removed element 13 -> pass = True; reason = ok
Removed element 14 -> pass = True; reason = ok
Removed element 15 -> pass = True; reason = ok
Removed element 16 -> pass = True; reason = ok
Removed element 17 -> pass = True; reason = ok
Removed element 18 -> pass = True; reason = ok
Removed element 19 -> pass = True; reason = ok
Removed element 20 -> pass = True; reason = ok
Removed element 21 -> pass = True; reason = ok
Removed element 22 -> pass = True; reason = ok
Removed element 23 -> pass = True; reason = ok
Removed element 24 -> pass = True; reason = ok
Removed element 25 -> pass = True; reason = ok
Removed element 26 -> pass = True; reason = ok
Removed element 27 -> pass = True; reason = ok
Removed element 28 -> pass = True; reason = ok
Removed element 29 -> pass = True; reason = ok
Removed element 30 -> pass = True; reason = ok
Removed element 31 -> pass = True; reason = ok
Removed element 32 -> pass = True; reason = ok
Removed element 33 -> pass = True; reason = ok
Removed element 34 -> pass = True; reason = ok
Removed element 35 -> pass = True; reason = ok
```

Figure 11: Results of strike resistant testing

6.7. Displacement

	node	ux (m)	uy (m)
0	0	0.000000	0.000000
1	1	0.000017	0.000000
2	2	0.000377	-0.000284
3	3	0.000344	0.000068
4	4	0.001399	0.000134
5	5	0.001433	-0.000569
6	6	0.003183	-0.000853

7	7	0.003183	-0.002597
8	8	0.003183	-0.004339
9	9	0.003183	-0.006051
10	10	0.003602	-0.006051
11	11	0.003774	-0.004339
12	12	0.003949	-0.002597
13	13	0.004124	-0.000855
14	14	0.004232	0.000201
15	15	0.003168	0.000203

7. Design Comparison
   7.1. While the initial and final truss designs meet the feasibility and strike resistant requirements, including stress, buckling, and slenderness, there are a few differences that make one more optimal compared to the other.

The initial design configuration had a more typical grid-like truss design, which is optimal for stability and dealing with heavier loads. However, it did not optimize material usage, and therefore had a higher C value. The final truss design configuration used the initial design as a foundation, but changed some of the joints and members locations. This design contained a truss structure similar to a monopitch truss where a right triangle is used and then supporting members are added on the inside. Because this design shortened the lengths of members compared to the first, its optimization of material usage is better and therefore has a lower C value. This new design allowed for the redistribution of member internal loads that could support the load P, while also still meeting the strike resistant requirements.

$$C_{initial} = 20.1635 > C_{final} = 19.9929$$
$$C'_{initial} = 0.000206 > C'_{final} = 0.000204$$

8. Conclusion and Recommendations
   8.1. This project successfully developed a tower crane truss structure that is both strike resistant and optimized for material efficiency. Both solutions meet design and geometric constraints while also meeting all damage cases for strike resistance. The final design showing a lower C value proves that you can design a truss structure that meets all your needs with less materials, but that doesn't sacrifice stability or structural integrity.

For simplicity, members in this project were square beams, however most truss structures can be found to use I beams. The use of I beams could improve and create a more optimized truss structure because I beams have a higher bending

stiffness, moment of inertia, and smaller cross sectional area. Another recommendation could be to fit the cross sectional area to members based on how much internal force they experience. Allowing members that experience higher forces to have bigger cross sectional areas while others that don't experience bigger forces to not could reduce the amount of material that is required in the structure, which would reduce C and make a more optimized solution.

Appendix

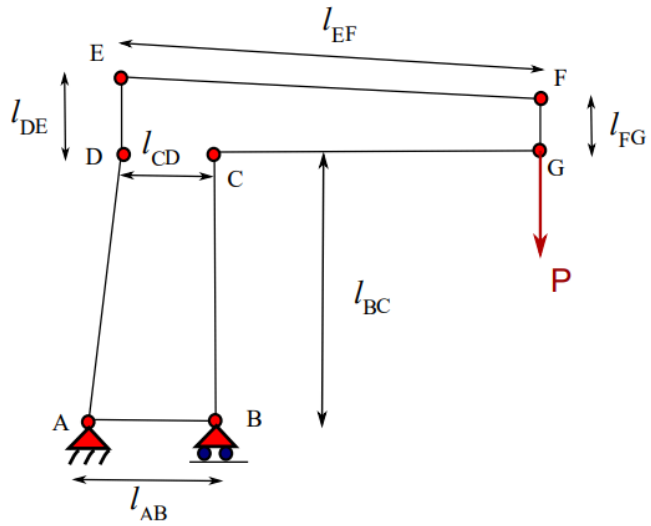
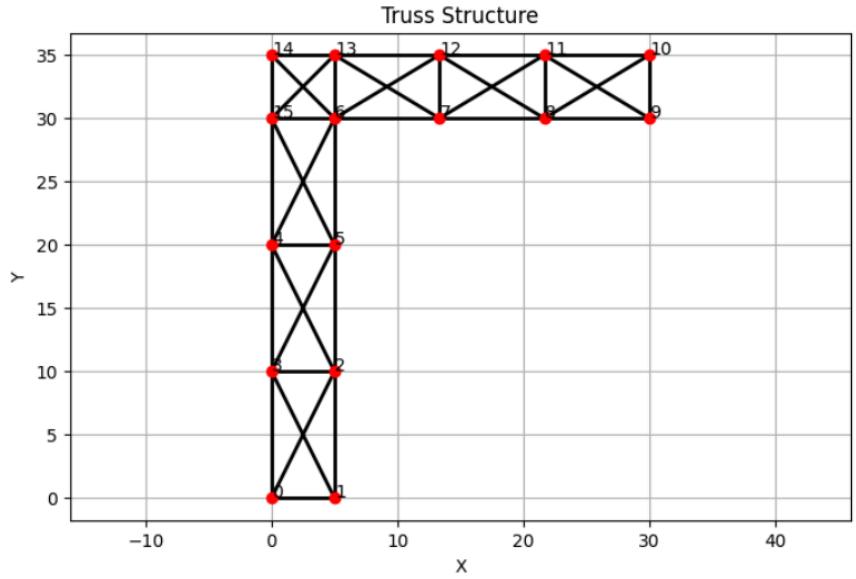


Figure 1: Original cell phone tower truss structure problem setup given in project assignment



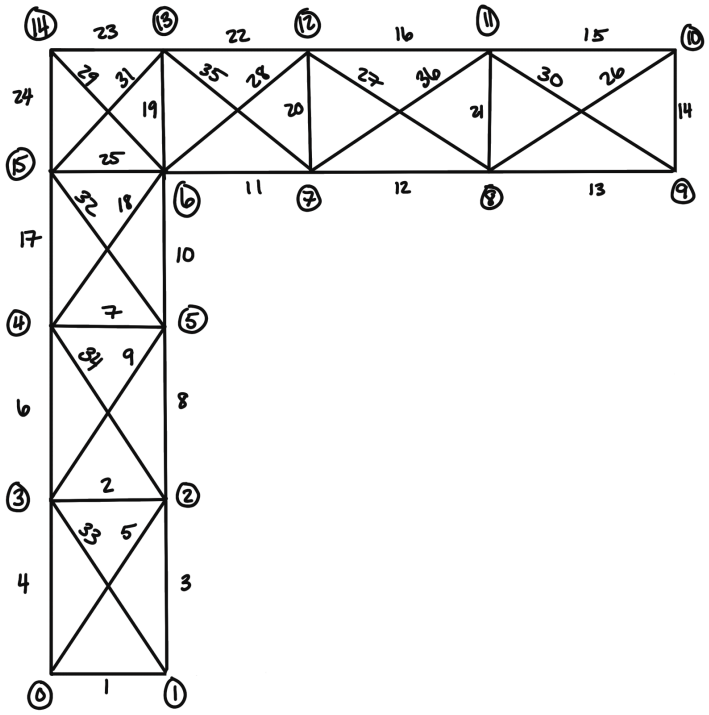


Figure 2a & 2b: Initial truss design with labeled joints and members

	element	nodes	length	N	sigma	stress_ok
0	0	[0, 1]	5.000000	9466.554495	6.810471e+05	True
1	1	[2, 3]	5.000000	18691.979941	1.344747e+06	True
2	2	[1, 2]	10.000000	-78892.391010	5.675712e+06	True
3	3	[0, 3]	10.000000	18933.108990	1.362094e+06	True
4	4	[0, 2]	11.180340	-21167.859364	1.522868e+06	True
5	5	[3, 4]	10.000000	18450.850892	1.327399e+06	True
6	6	[4, 5]	5.000000	18716.428736	1.346506e+06	True
7	7	[2, 5]	10.000000	-79374.649108	5.710406e+06	True
8	8	[3, 5]	11.180340	-20628.678419	1.484078e+06	True
9	9	[5, 6]	10.000000	-78843.493421	5.672194e+06	True
10	10	[6, 7]	8.333333	63.483961	4.567191e+03	True
11	11	[7, 8]	8.333334	-2.450543	1.762981e+02	True
12	12	[8, 9]	8.333333	0.094453	6.795169e+00	True
13	13	[9, 10]	5.000000	0.056672	4.077101e+00	True
14	14	[10, 11]	8.333333	0.094453	6.795169e+00	True
15	15	[11, 12]	8.333334	-2.450543	1.762981e+02	True
16	16	[15, 4]	10.000000	18982.006579	1.365612e+06	True
17	17	[4, 6]	11.180340	-21222.528530	1.526801e+06	True
18	18	[6, 13]	5.000000	-948.679580	6.825033e+04	True
19	19	[7, 12]	5.000000	36.620052	2.634536e+03	True
20	20	[8, 11]	5.000000	-1.413654	1.017017e+02	True
21	21	[12, 13]	8.333333	63.483961	4.567191e+03	True
22	22	[13, 14]	5.000000	-986.769959	7.099064e+04	True
23	23	[14, 15]	5.000000	-986.769959	7.099064e+04	True
24	24	[15, 6]	5.000000	8504.233331	6.118153e+05	True
25	25	[8, 10]	9.718253	-0.110150	7.924460e+00	True
26	26	[12, 8]	9.718253	2.857800	2.055971e+02	True
27	27	[12, 6]	9.718253	-74.034386	5.326215e+03	True
28	28	[14, 6]	7.071068	1395.503458	1.003959e+05	True
29	29	[9, 11]	9.718253	-0.110150	7.924461e+00	True
30	30	[15, 13]	7.071068	1395.503458	1.003959e+05	True
31	31	[15, 5]	11.180340	-21222.528530	1.526801e+06	True
32	32	[1, 3]	11.180340	-21167.859364	1.522868e+06	True
33	33	[2, 4]	11.180340	-20628.678419	1.484078e+06	True
34	34	[13, 7]	9.718253	-74.034386	5.326215e+03	True
35	35	[7, 11]	9.718253	2.857800	2.055971e+02	True

Figure 3: Results of stress testing

	Pcr	buckling_ok
0	1.271271e+06	True
1	1.271271e+06	True
2	3.178177e+05	True
3	3.178177e+05	True
4	2.542542e+05	True
5	3.178177e+05	True
6	1.271271e+06	True
7	3.178177e+05	True
8	2.542542e+05	True
9	3.178177e+05	True
10	4.576575e+05	True
11	4.576575e+05	True
12	4.576575e+05	True
13	1.271271e+06	True
14	4.576575e+05	True
15	4.576575e+05	True
16	3.178177e+05	True
17	2.542542e+05	True
18	1.271271e+06	True
19	1.271271e+06	True
20	1.271271e+06	True
21	4.576575e+05	True
22	1.271271e+06	True
23	1.271271e+06	True
24	1.271271e+06	True
25	3.365129e+05	True
26	3.365128e+05	True
27	3.365129e+05	True
28	6.356354e+05	True
29	3.365129e+05	True
30	6.356354e+05	True
31	2.542542e+05	True
32	2.542542e+05	True
33	2.542542e+05	True
34	3.365129e+05	True
35	3.365128e+05	True

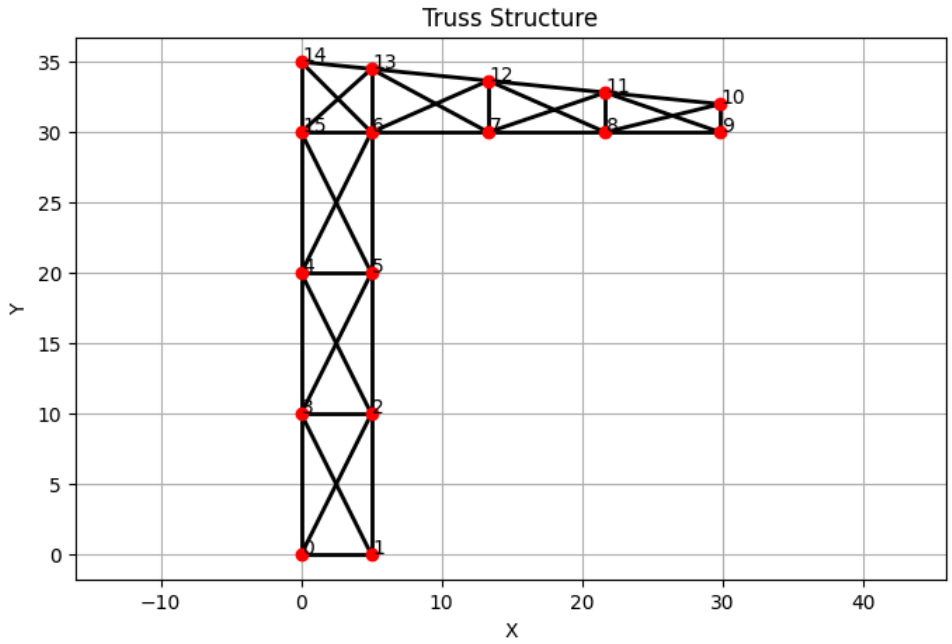
Figure 4: Results of buckling testing

slenderness	slender_ok
146.910632	True
146.910632	True
293.821264	True
293.821264	True
328.502160	True
293.821264	True
146.910632	True
293.821264	True
328.502160	True
293.821264	True
244.851044	True
244.851064	True
244.851052	True
146.910632	True
244.851052	True
244.851064	True
293.821264	True
328.502160	True
146.910632	True
146.910632	True
146.910632	True
244.851044	True
146.910632	True
146.910632	True
146.910632	True
285.542942	True
285.542952	True
285.542934	True
207.763008	True
285.542942	True
207.763008	True
328.502160	True
328.502160	True
328.502160	True
285.542934	True
285.542952	True

Figure 5: Results of slenderness ratio testing

```
Removed element 0 -> pass = True; reason = ok
Removed element 1 -> pass = True; reason = ok
Removed element 2 -> pass = True; reason = ok
Removed element 3 -> pass = True; reason = ok
Removed element 4 -> pass = True; reason = ok
Removed element 5 -> pass = True; reason = ok
Removed element 6 -> pass = True; reason = ok
Removed element 7 -> pass = True; reason = ok
Removed element 8 -> pass = True; reason = ok
Removed element 9 -> pass = True; reason = ok
Removed element 10 -> pass = True; reason = ok
Removed element 11 -> pass = True; reason = ok
Removed element 12 -> pass = True; reason = ok
Removed element 13 -> pass = True; reason = ok
Removed element 14 -> pass = True; reason = ok
Removed element 15 -> pass = True; reason = ok
Removed element 16 -> pass = True; reason = ok
Removed element 17 -> pass = True; reason = ok
Removed element 18 -> pass = True; reason = ok
Removed element 19 -> pass = True; reason = ok
Removed element 20 -> pass = True; reason = ok
Removed element 21 -> pass = True; reason = ok
Removed element 22 -> pass = True; reason = ok
Removed element 23 -> pass = True; reason = ok
Removed element 24 -> pass = True; reason = ok
Removed element 25 -> pass = True; reason = ok
Removed element 26 -> pass = True; reason = ok
Removed element 27 -> pass = True; reason = ok
Removed element 28 -> pass = True; reason = ok
Removed element 29 -> pass = True; reason = ok
Removed element 30 -> pass = True; reason = ok
Removed element 31 -> pass = True; reason = ok
Removed element 32 -> pass = True; reason = ok
Removed element 33 -> pass = True; reason = ok
Removed element 34 -> pass = True; reason = ok
Removed element 35 -> pass = True; reason = ok
```

Figure 6: Results of strike resistant testing


Truss Structure
X
Y
14
13
12
11
10
15
6
7
8
9
4
5
3
2
0
1

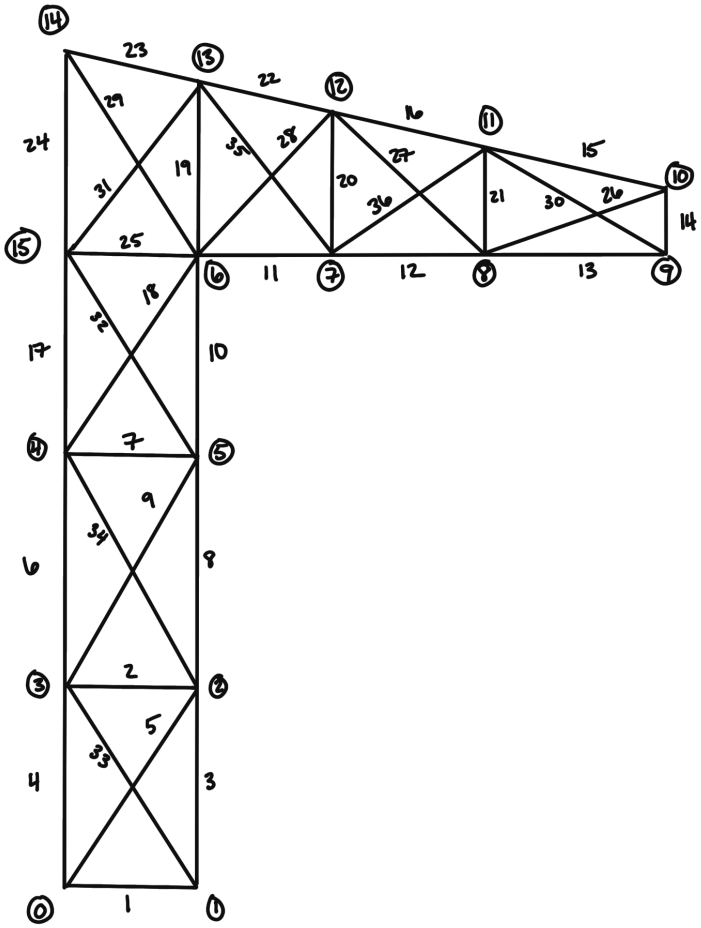

14
23
13
22
12
11
10
24
29
19
35
28
27
15
31
20
30
21
26
14
15
25
6
11
7
12
8
13
9
18
32
17
10
4
7
5
9
34
6
8
3
2
2
5
33
4
3
0
1
1

Figure 7a & 7b: Final truss design with labeled joints and members

	element	nodes	length	N	sigma	stress_ok
0	0	[0, 1]	5.000000	9466.555590	6.810472e+05	True
1	1	[2, 3]	5.000000	18691.936853	1.344744e+06	True
2	2	[1, 2]	10.000000	-78892.388820	5.675711e+06	True
3	3	[0, 3]	10.000000	18933.111180	1.362094e+06	True
4	4	[0, 2]	11.180340	-21167.861812	1.522868e+06	True
5	5	[3, 4]	10.000000	18450.762526	1.327393e+06	True
6	6	[4, 5]	5.000000	18718.166706	1.346631e+06	True
7	7	[2, 5]	10.000000	-79374.737474	5.710413e+06	True
8	8	[3, 5]	11.180340	-20628.579623	1.484070e+06	True
9	9	[5, 6]	10.000000	-78839.929115	5.671937e+06	True
10	10	[6, 7]	8.333333	50.026302	3.599015e+03	True
11	11	[7, 8]	8.333334	-0.871097	6.266886e+01	True
12	12	[8, 9]	8.182933	0.007432	5.346847e-01	True
13	13	[9, 10]	2.000000	0.002563	1.844197e-01	True
14	14	[10, 11]	8.224156	0.007470	5.373782e-01	True
15	15	[11, 12]	8.375312	-0.875485	6.298455e+01	True
16	16	[15, 4]	10.000000	18985.570885	1.365868e+06	True
17	17	[4, 6]	11.180340	-21226.513545	1.527087e+06	True
18	18	[6, 13]	4.497500	-1036.683719	7.458156e+04	True
19	19	[7, 12]	3.659900	26.704166	1.921163e+03	True
20	20	[8, 11]	2.822400	-0.380759	2.739273e+01	True
21	21	[12, 13]	8.375322	50.278367	3.617149e+03	True
22	22	[13, 14]	5.025187	-1063.987578	7.654587e+04	True
23	23	[14, 15]	5.000000	-952.259877	6.850790e+04	True
24	24	[15, 6]	5.000000	8434.130770	6.067720e+05	True
25	25	[8, 10]	8.423799	-0.007651	5.504232e-01	True
26	26	[12, 8]	9.101611	0.951406	6.844651e+01	True
27	27	[12, 6]	9.101610	-54.638390	3.930819e+03	True
28	28	[14, 6]	7.071068	1497.163795	1.077096e+05	True
29	29	[9, 11]	8.656000	-0.007862	5.655956e-01	True
30	30	[15, 13]	6.725140	1423.920148	1.024403e+05	True
31	31	[15, 5]	11.180340	-21226.513545	1.527087e+06	True
32	32	[1, 3]	11.180340	-21167.861812	1.522868e+06	True
33	33	[2, 4]	11.180340	-20628.579623	1.484070e+06	True
34	34	[13, 7]	9.469527	-56.847054	4.089716e+03	True
35	35	[7, 11]	8.798318	0.919703	6.616566e+01	True

Figure 8: Results of stress testing

	Pcr	buckling_ok
0	1.271271e+06	True
1	1.271271e+06	True
2	3.178177e+05	True
3	3.178177e+05	True
4	2.542542e+05	True
5	3.178177e+05	True
6	1.271271e+06	True
7	3.178177e+05	True
8	2.542542e+05	True
9	3.178177e+05	True
10	4.576575e+05	True
11	4.576575e+05	True
12	4.746353e+05	True
13	7.945443e+06	True
14	4.698892e+05	True
15	4.530812e+05	True
16	3.178177e+05	True
17	2.542542e+05	True
18	1.571215e+06	True
19	2.372683e+06	True
20	3.989707e+06	True
21	4.530802e+05	True
22	1.258559e+06	True
23	1.271271e+06	True
24	1.271271e+06	True
25	4.478804e+05	True
26	3.836556e+05	True
27	3.836557e+05	True
28	6.356354e+05	True
29	4.241735e+05	True
30	7.027089e+05	True
31	2.542542e+05	True
32	2.542542e+05	True
33	2.542542e+05	True
34	3.544227e+05	True
35	4.105621e+05	True

Figure 9: Results of buckling testing

slenderness	slender_ok
146.910632	True
146.910632	True
293.821264	True
293.821264	True
328.502160	True
293.821264	True
146.910632	True
293.821264	True
328.502160	True
293.821264	True
244.851044	True
244.851064	True
240.431981	True
58.764253	True
241.643186	True
246.084486	True
293.821264	True
328.502160	True
132.146114	True
107.535644	True
82.928114	True
246.084759	True
147.650685	True
146.910632	True
146.910632	True
247.509141	True
267.424678	True
267.424659	True
207.763008	True
254.331692	True
197.598910	True
328.502160	True
328.502160	True
328.502160	True
278.234845	True
258.513279	True

Figure 10: Results of slenderness ratio testing

```
Removed element 0 -> pass = True; reason = ok
Removed element 1 -> pass = True; reason = ok
Removed element 2 -> pass = True; reason = ok
Removed element 3 -> pass = True; reason = ok
Removed element 4 -> pass = True; reason = ok
Removed element 5 -> pass = True; reason = ok
Removed element 6 -> pass = True; reason = ok
Removed element 7 -> pass = True; reason = ok
Removed element 8 -> pass = True; reason = ok
Removed element 9 -> pass = True; reason = ok
Removed element 10 -> pass = True; reason = ok
Removed element 11 -> pass = True; reason = ok
Removed element 12 -> pass = True; reason = ok
Removed element 13 -> pass = True; reason = ok
Removed element 14 -> pass = True; reason = ok
Removed element 15 -> pass = True; reason = ok
Removed element 16 -> pass = True; reason = ok
Removed element 17 -> pass = True; reason = ok
Removed element 18 -> pass = True; reason = ok
Removed element 19 -> pass = True; reason = ok
Removed element 20 -> pass = True; reason = ok
Removed element 21 -> pass = True; reason = ok
Removed element 22 -> pass = True; reason = ok
Removed element 23 -> pass = True; reason = ok
Removed element 24 -> pass = True; reason = ok
Removed element 25 -> pass = True; reason = ok
Removed element 26 -> pass = True; reason = ok
Removed element 27 -> pass = True; reason = ok
Removed element 28 -> pass = True; reason = ok
Removed element 29 -> pass = True; reason = ok
Removed element 30 -> pass = True; reason = ok
Removed element 31 -> pass = True; reason = ok
Removed element 32 -> pass = True; reason = ok
Removed element 33 -> pass = True; reason = ok
Removed element 34 -> pass = True; reason = ok
Removed element 35 -> pass = True; reason = ok
```

Figure 11: Results of strike resistant testing

Code used for graphing:

```
import numpy as np
import matplotlib.pyplot as plt

nodeCords = np.array([
        [0.0, 0.0],    #(0)
        [5.0, 0.0],   #(1)
        [5.0, 10.0],   #(2)
        [0.0, 10.0], #(3)
        [0.0, 20.0], #(4)
        [5.0, 20.0],   #(5)
```

```
        [5.0, 30.0],  #(6)
        [13.333333, 30.0],  #(7)
        [21.6666667, 30.0],  #(8)
        [29.8496, 30.0],  #(9)
        [29.8496, 32.0],  #(10)
        [21.6666667, 32.8224],  #(11)
        [13.333333, 33.6599],  #(12)
        [5.0, 34.4975],  #(13)
        [0.0, 35.0],  #(14)
        [0.0, 30.0]  #(15)
    ])



elemNodes = np.array([
        [0,1], # 1
        [2,3], # 2
        [1,2], # 3
        [0,3], # 4
        [0,2], # 5
        [3,4], # 6
        [4,5], # 7
        [2,5], # 8
        [3,5], # 9
        [5,6], # 10
        [6,7], # 11
        [7,8], # 12
        [8,9], # 13
        [9,10], # 14
        [10,11], # 15
        [11,12], # 16
        [15,4], # 17
        [4,6], # 18
        [6,13], # 19
        [7,12], # 20
        [8,11], # 21
        [12,13], # 22
        [13,14], # 23
        [14,15], # 24
        [15,6], # 25
```

```
        [8,10], # 26
        [12,8], # 27
        [12,6], # 28
        [14,6], # 29
        [9,11], # 30
        [15,13], # 31
        [15,5], # 32
        [1,3], # 33
        [2,4], # 34
        [13,7], # 35
        [7,11] # 36
    ])

def plot_truss(nodeCords, elemNodes):
    plt.figure(figsize=(8, 5))

    for (i, j) in elemNodes:
        xi, yi = nodeCords[i]
        xj, yj = nodeCords[j]
        plt.plot([xi, xj], [yi, yj], 'k-', linewidth=2)

    for idx, (x, y) in enumerate(nodeCords):
        plt.plot(x, y, 'ro', markersize=6)
        plt.text(x + 0.05, y + 0.05, f'{idx}', fontsize=10)

    plt.title('Truss Structure')
    plt.xlabel('X')
    plt.ylabel('Y')
    plt.axis('equal')
    plt.grid(True)
    plt.show()

plot_truss(nodeCords, elemNodes)
```

Generative AI prompts

1. “im working on a project report but im unsure on how i should set up my report, how to split up the sections and what to include/write in each section and how to write it. i have one solution so far and i just used python to find the answers so I only have my drawing and the code so how can i use that for my report?”

    a. Our project outline stated what was to be included and graded, but I was unsure on how to break it up into sections and through this prompt I was able to get a good idea on how to do this and what I could include in each section without overlapping my data/descriptions too much. After this prompt, I asked for a sample report so I could see how it would look and what I wanted to change and not put in my report.

2. “I am getting a Linear Singular error in my code, can you tell me what is wrong: (I inserted my code here)”
    a. When trying to find potential solutions I was testing many different node and element arrays. Doing this I had to change many of my values and add/subtract values, in doing this I accidentally placed a comma instead of a period and got this error. AI helped me find this error and I was able to fix it. Additionally, I asked again when my code didn’t work, that time I made an error in my area value and needed to make it a float number.
3. “can you grade my report and tell me how to make it better if it needs” (inserted a pdf of my report)
    a. It told me my report was an A and showed a huge amount of effort and understanding of the project, but I could clean up some of my sentences in my introduction and problem description sections, add a more engineering explanation to my change in design from initial to final, and I could clean up my coding sections so it is easier to read and not repeat the code in my final design section as it is the same for the initial besides the element and node arrays. I did change the code from my final design section and tried to make my sentences better in my introduction, problem description, and final design description.