School of Mechanical Engineering

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

GRADUATE FINAL PROJECT REPORT
ME 548

Truss Bridge Design Optimization

Siddharth Palani Natarajan

School of Mechanical Engineering
Graduate Student of Mechanical Engineering
University of Wisconsin-Madison

May 2022

Statement of Ownership

The project carried out and the results presented in this report are, except where acknowledged, the original work of the author, and all project related work was conducted during the program.

Acknowledgement

I would like to extend my gratitude to my faculty **Dr. Krishnan Suresh**, School of Mechanical Engineering for his enthusiastic guidance, instruction, and encouragement at every stage of my project.

Project Goals

To design a Truss Bridge system and optimize the design in order to minimize Compliance while implementing various constraints to meet the requirements of the Problem Statement.

Section 1: Strategy

The Project requires us to design a Truss Bridge system and optimize the design such that its Compliance is minimized. The Truss Bridge design that I have chosen is a variant of the Rectangular Truss of length $100\ m$ and height $30\ m$.

The choice for this design was supported by the fact that the example 2 designs provided in the Final Project Problem Statement had the least Initial and Final Compliance. As a designer, if my goal is to obtain a better design I have two options:

Option 1: Figure out alternative designs that can produce Lower Initial and Final Compliance than the example designs provided.

Option 2: Modify the existing example designs to obtain Lower Initial and Final Compliance.

After experimenting with various alternative designs, I was unable to find designs that had Lower Initial and Final Compliance. So I decided to experiment with modifying the example designs provided. I realised that by alternatively inverting each truss member that lay inside each square formed by truss members, the Initial and Final Compliance reduced significantly. This was achieved by utilising **testProject.m** and **truss2dMinComplianceAllConstraints.m**.

In **testProject.m**, I began by first initialising a **Nodal Coordinate Array** given as **xy** that consists all the nodal points of the truss members. My Truss Bridge design consists of **147** points. I then initialise an **Element Connectivity Array** given as **connectivity** that connects all the nodes for each truss member. Material design parameters such as **Young's Modulus**, **Density** are also initialised in this script. Next I constrain my design, i.e. fixing the nodes and applying the Force on the nodes of truss members as per the Problem Statement.

This initial setup then calls various functions that is part of the class **truss2d.m** and **truss2dMinComplianceAllConstraints.m** to perform Finite Element Analysis and Design optimization. In **truss2dMinComplianceAllConstraints.m**, I setup the constraints for the problem. In order to ensure the accuracy of the solution produced by the solver (fmincon), I scale the objective function which aims to minimize the Compliance of the system. I also scale the Volume and Stress constraints for the same reason. In the end,

truss2dMinComplianceAllConstraints.m ultimately minimize the Compliance of the truss system subject to all the constraints prescribed by the Problem Statement while **truss2d.m** performs the Finite Element Analysis for this problem.

Section 2: Formulation

Objective Function:

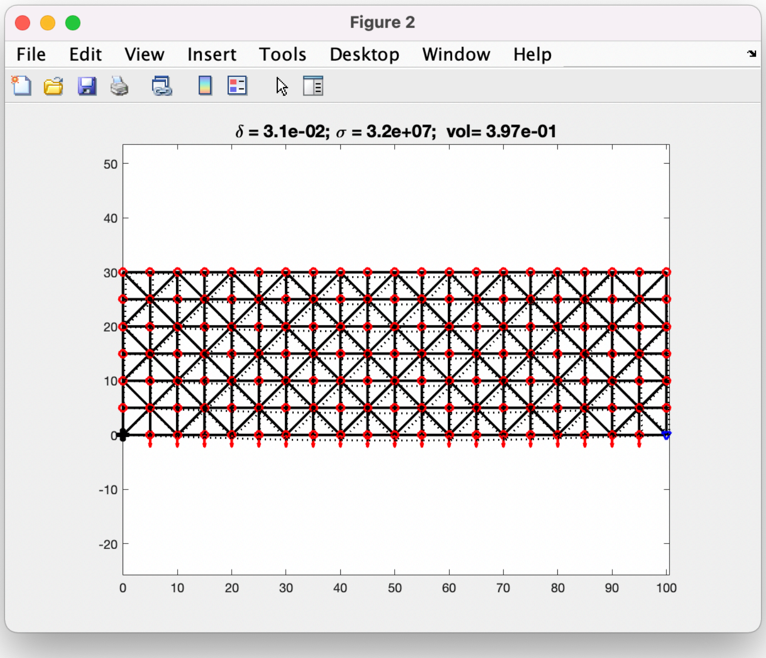
$$\underset{\{x_i\}}{Min}\ \frac{f^T d}{J_0}\ \text{(Compliance)}$$

$$\text{where } x_i = \frac{A_i}{A_0}\ \text{(Scaling Area)}$$

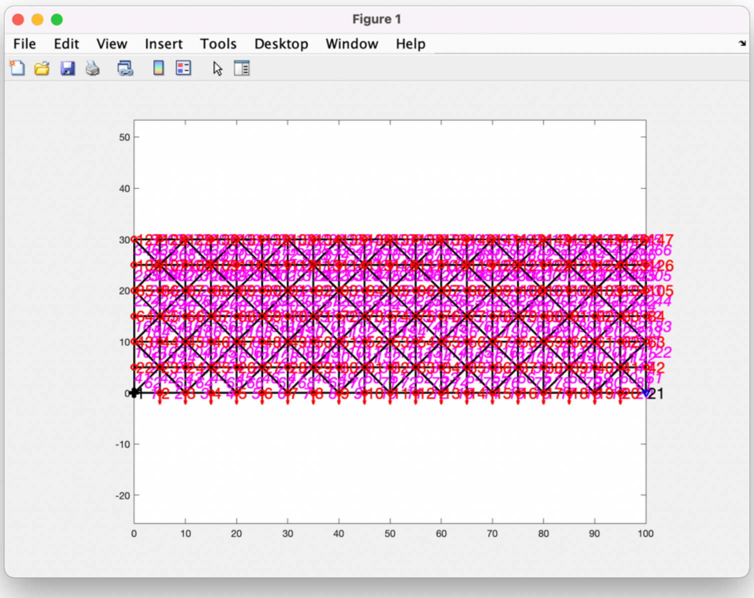
Constraints:

1. $\frac{\sigma_i}{\sigma_y} - 1 \leq 0$ (Tensile Stress)
2. $\frac{-\sigma_i}{\sigma_y} - 1 \leq 0$ (Compressive Stress)
3. $3 \leq l_i \leq 10$ (Length)
4. $0 < w_i \leq 30{,}000$ (Weight)

Section 3: Results and Conclusions



Plot of Truss Bridge



Plot of Truss Bridge with Node numbering and Force vectors

	Before Optimization	After Optimization
Min Bar Length	5	5
Max Bar Length	7.0711	7.0711
Max Mass	3060	3060
Compliance	5.5775e3	2.0639e3
Max Stress	1.3556e8	3.2237e7
Max Deflection	0.0828	0.0314

Results Table Before and After Optimization

Appendix: MATLAB Code

testProject.m

```
trussClass = 'truss2dMinComplianceAllConstraints';

xy = [0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 0 5 10
15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 0 5 10 15 20 25 30
35 40 45 50 55 60 65 70 75 80 85 90 95 100 0 5 10 15 20 25 30 35 40 45 50
55 60 65 70 75 80 85 90 95 100 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70
75 80 85 90 95 100 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90
95 100 0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100;
      0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 5 5 5 5 5 5 5 5 5 5 5 5 5
5 5 5 5 5 5 5 5 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 25 25 25 25 25 25
25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30];

connectivity = [1 2; 2 3; 3 4; 4 5; 5 6; 6 7; 7 8; 8 9; 9 10; 10 11;
                11 12; 12 13; 13 14; 14 15; 15 16; 16 17; 17 18; 18 19;
                19 20; 20 21; 22 23; 23 24; 24 25; 25 26; 26 27; 27 28;
                28 29; 29 30; 30 31; 31 32; 32 33; 33 34; 34 35; 35 36;
                36 37; 37 38; 38 39; 39 40; 40 41; 41 42;
                1 22; 2 23; 3 24; 4 25; 5 26; 6 27; 7 28; 8 29; 9 30; 10
31;
                11 32; 12 33; 13 34; 14 35; 15 36; 16 37; 17 38; 18 39; 19
40;
                20 41; 21 42; 23 1; 23 3; 25 3; 25 5; 27 5; 27 7; 29 7; 29
9;
                31 9; 31 11; 33 11; 33 13; 35 13; 35 15; 37 15; 37 17; 39
17;
                39 19; 41 19; 41 21;
                43 44; 44 45;
                45 46; 46 47; 47 48; 48 49; 49 50; 50 51; 51 52; 52 53;
                53 54; 54 55; 55 56; 56 57; 57 58; 58 59; 59 60; 60 61;
                61 62; 62 63;
                22 43; 23 44; 24 45; 25 46; 26 47; 27 48; 28 49;
                29 50; 30 51; 31 52; 32 53; 33 54; 34 55; 35 56; 36 57; 37
58;
                38 59; 39 60; 40 61; 41 62; 42 63;
                43 23; 45 23; 45 25; 47 25; 47 27; 49 27; 49 29; 51 29; 51
31;
                53 31; 53 33; 55 33; 55 35; 57 35; 57 37; 59 37; 59 39; 61
39;
                61 41; 63 41;
                64 65; 65 66; 66 67; 67 68; 68 69; 69 70;
                70 71; 71 72; 72 73; 73 74; 74 75; 75 76; 76 77; 77 78;
                78 79; 79 80; 80 81; 81 82; 82 83; 83 84;
                43 64; 44 65; 45 66; 46 67;
                47 68; 48 69; 49 70; 50 71; 51 72; 52 73; 53 74; 54 75; 55
76;
                56 77; 57 78; 58 79; 59 80; 60 81; 61 82; 62 83; 63 84;
                65 43; 65 45; 67 45; 67 47; 69 47; 69 49; 71 49; 71 51;
                73 51; 73 53; 75 53; 75 55; 77 55; 77 57; 79 57; 79 59;
                81 59; 81 61; 83 61; 83 63;
                85 86; 86 87;
                87 88; 88 89; 89 90; 90 91; 91 92; 92 93; 93 94; 94 95;
                95 96; 96 97; 97 98; 98 99; 99 100; 100 101; 101 102; 102
103;
                103 104; 104 105;
                64 85;
                65 86; 66 87; 67 88; 68 89; 69 90; 70 91; 71 92; 72 93; 73
94;
                74 95; 75 96; 76 97; 77 98; 78 99; 79 100; 80 101; 81 102;
82 103;
                83 104; 84 105;
                85 65; 87 65; 87 67; 89 67; 89 69; 91 69; 91 71; 93 71; 93
73; 95 73;
                95 75; 97 75; 97 77; 99 77; 99 79; 101 79; 101 81; 103 81;
103 83; 105 83;
                106 107; 107 108; 108 109; 109 110; 110 111;
                111 112; 112 113; 113 114; 114 115; 115 116; 116 117; 117
118;
                118 119; 119 120; 120 121; 121 122; 122 123; 123 124; 124
125;
                125 126;
                85 106; 86 107; 87 108; 88 109; 89 110; 90 111;
                91 112; 92 113; 93 114; 94 115; 95 116; 96 117; 97 118; 98
119;
                99 120; 100 121; 101 122; 102 123; 103 124; 104 125; 105
126;
                85 107; 87 107; 87 109; 89 109; 89 111; 91 111; 91 113; 93
113;
                93 115; 95 115; 95 117; 97 117; 97 119; 99 119; 99 121;
101 121;
                101 123; 103 123; 103 125; 105 125;
                127 128; 128 129; 129 130; 130 131; 131 132; 132 133;
                133 134; 134 135; 135 136; 136 137; 137 138; 138 139; 139
140;
                140 141; 141 142; 142 143; 143 144; 144 145; 145 146; 146
147;
                106 127;
                107 128; 108 129; 109 130; 110 131; 111 132; 112 133; 113
134; 114 135;
```

```
                115 136; 116 137; 117 138; 118 139; 119 140; 120 141; 121
142; 122 143;
                123 144; 124 145; 125 146; 126 147;
                127 107; 129 107; 129 109; 131 109; 131 111; 133 111; 133
113; 135 113;
                135 115; 137 115; 137 117; 139 117; 139 119; 141 119; 141
121; 143 121;
                143 123; 145 123; 145 125; 147 125]';

t = feval(trussClass,xy,connectivity); % Model Initialisation
A = 3060/(sum(t.myL)*7700);
t = t.assignE(2e11); % Young's Modulus
t = t.assignA(A);
t = t.fixXofNodes(1);
t = t.fixYofNodes([1 21]);
t = t.applyForce([2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20],[0;-
100000/19]);
t.plot(1);
tic
t = t.optimize();
toc
optimizedArea = t.myArea;

%% Results
figure; t.plot(); hold on; t.plotDeformed();
disp('Initial Compliance: '); disp(t.myInitialCompliance)
disp('Final Compliance: '); disp(t.myFinalCompliance);
disp('Minimum bar length');disp(min(t.myL))
disp('Maximum bar length');disp(max(t.myL))
disp('Total mass before Optimization');disp(sum(t.myInitialMass))
disp('Total mass after Optimization');disp(sum(t.myFinalMass))
disp('Maximum Deformation before
Optimization');disp(t.myInitialDeformation)
disp('Maximum Deformation after Optimization');disp(t.myFinalDeformation)
disp('Maximum Stress before Optimization');disp(max(t.myInitialStress))
disp('Maximum Stress after Optimization');disp(max(t.myFinalStress))
```

truss2dMinComplianceAllConstraints.m

```
classdef truss2dMinComplianceAllConstraints < truss2d
    % stress-constrained volume minimization
    properties(GetAccess = 'public', SetAccess = 'private')
        myInitialVolume;
        myInitialArea;
        myInitialCompliance;
        myInitialMass;
        myInitialDeformation;
        myInitialStress;
        myFinalArea;
        myFinalCompliance;
        myFinalVolume;
        myFinalMass;
        myFinalStress;
        myFinalDeformation;
        myYieldStress;
        myLambda;
    end
    methods
        function obj = truss2dMinComplianceAllConstraints(xy,connectivity)
            obj = obj@truss2d(xy,connectivity);
            obj.myYieldStress(1:obj.myNumTrussBars) = 100e6; % default
        end
        function obj = assignYieldStress(obj,yieldStress,members)
            % assign sMax to one or more members
            % if members is not given, then assign to all
            if (nargin == 2)
                members = 1:obj.myNumTrussBars;
            else
                assert(max(members) <= obj.myNumTrussBars);
                assert(min(members) >=  1);
            end
            obj.myYieldStress(members) = yieldStress;
        end
        function JRelative = complianceObjective(obj,x)
            Area = x.*obj.myInitialArea;
            obj = obj.assignA(Area);
            obj = obj.assemble();
            obj = obj.solve();
            J = obj.getCompliance();
            JRelative = J/obj.myInitialCompliance;
        end
%         function volRelative = volumeObjective(obj,x)
%             Area = x.*obj.myInitialArea;
%             obj = obj.assignA(Area);
%             vol = sum(obj.myArea.*obj.myL);
%             volRelative = vol/obj.myInitialVolume;
%         end
        function [cineq,ceq] = stressConstraint(obj,x)
            Area = x.*obj.myInitialArea;
            obj = obj.assignA(Area);
            obj = obj.assemble();
            obj = obj.solve();
            nConstraints = 2*obj.myNumTrussBars; % two stress constraints
per bar
            cineq = zeros(1,nConstraints);
            constraint = 1;
            for m = 1:obj.myNumTrussBars
                cineq(constraint) = obj.myStress(m)/obj.myYieldStress(m)-
1;% tension
                cineq(constraint+1) = -
obj.myStress(m)/obj.myYieldStress(m) -1;%compression
                constraint = constraint+2; % increment
            end
            ceq = [];
        end
        function obj = initialize(obj)
            obj.myInitialArea = obj.myArea;
            obj.myInitialVolume = sum(obj.myArea.*obj.myL);
            obj = obj.assemble();
            obj = obj.solve();
            obj.myInitialCompliance =  obj.getCompliance();
            obj.myInitialMass = sum((obj.myInitialArea.*obj.myL))*7700;
            obj.myInitialDeformation = obj.myDeformationMax;
            obj.myInitialStress = obj.myStress;
        end
        function processLambda(obj)
            ineqnonlin = obj.myLambda.ineqnonlin;
            maxValue = max(abs(ineqnonlin));
            ineqnonlin = ineqnonlin/maxValue; % scaled
            for m = 1:obj.myNumTrussBars
                if (ineqnonlin(2*m-1) > 0.0001)
                    disp(['Bar ' num2str(m) ': Tensile stress active']);
                elseif (ineqnonlin(2*m) > 0.0001)
                    disp(['Bar ' num2str(m) ': Compressive stress
active']);
                else
                    disp(['Bar ' num2str(m) ': Stress inactive']);
                end
            end
        end
        function obj = optimize(obj)
            options = optimset('fmincon');
            options.MaxFunEvals = 10000000;
            options.Iterations = 1000000;
            obj = obj.initialize();
            x0 = ones(1,obj.myNumTrussBars); % unitless quantities
            LB = 1e-12*ones(1,obj.myNumTrussBars); % small non-zero values
            %AinEq = (obj.myInitialArea.*obj.myL);
            %BinEq = 3060/7700;
            AinEq = (obj.myInitialArea.*obj.myL)/obj.myInitialVolume;
            BinEq = 1;
            [xMin,~,~,~,Lambda]  = fmincon(@obj.complianceObjective,x0,
...
                   AinEq,BinEq,[],[],LB,[],@obj.stressConstraint,options);
            obj = obj.assignA(xMin.*obj.myInitialArea);
            obj = obj.assemble();
            obj = obj.solve();
            obj.myFinalArea= obj.myArea;
            obj.myFinalVolume = sum(obj.myArea.*obj.myL);
            obj.myFinalCompliance =  obj.getCompliance();
            obj.myFinalMass = sum(obj.myArea.*obj.myL)*7700;
            obj.myFinalDeformation = obj.myDeformationMax;
            obj.myFinalStress = obj.myStress;
            obj.myLambda = Lambda;
        end
        end
    end
```