MATLAB Code vs. FEM Analysis of Truss

Abstract: For the roof truss below, MATLAB code was used to calculate the displacements and reaction forces at each joint, stress and strain in each truss, and results were compared to simulation results via ANSYS. Then, the MATLAB code was revised to find the maximum load capacity given a safety factor of 1.5. Optimization results were compared to ANSYS.

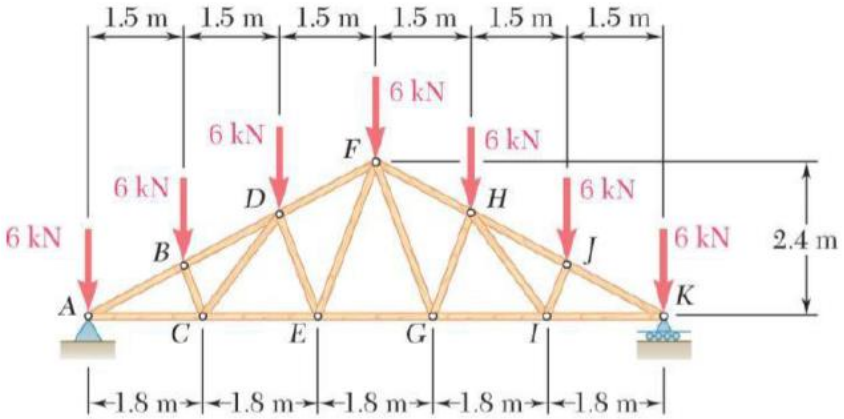


Figure 1. Double Fink Roof Truss

First, the given material properties were identified. See Table 1 for a summary of the given properties. The first part of the solution involved calculating the displacements and reaction forces at each node of the roof truss, as well as the stress and strain in each truss element. These values were calculated using MATLAB code and an ANSYS simulation of the roof truss. Before analyzing the data, it is important to describe the labeling scheme used to design the roof truss. As shown in Figure 9, the roof truss consists of 11 nodes (i.e., 22 global DOF) and 19 truss elements. In other words, for each node "i" its associated global DOF are 2i-1 and 2i. Hence, node 1 has global DOF of 1 and 2, node 2 has global DOF of 3 and 4, etc. Odd values of DOF correspond to the nodal x-direction while even values correspond to the nodal y-direction.

Table 1. Roof Truss Material and Cross-Section Properties

Cross-Section Dimensions	5cm x 5cm
Material Name	Pine Wood
Young's Modulus	9GPa
Poisson's Ratio	0.3
Ultimate Tensile Strength	96.5MPa
Compressive Yield Strength	37MPa

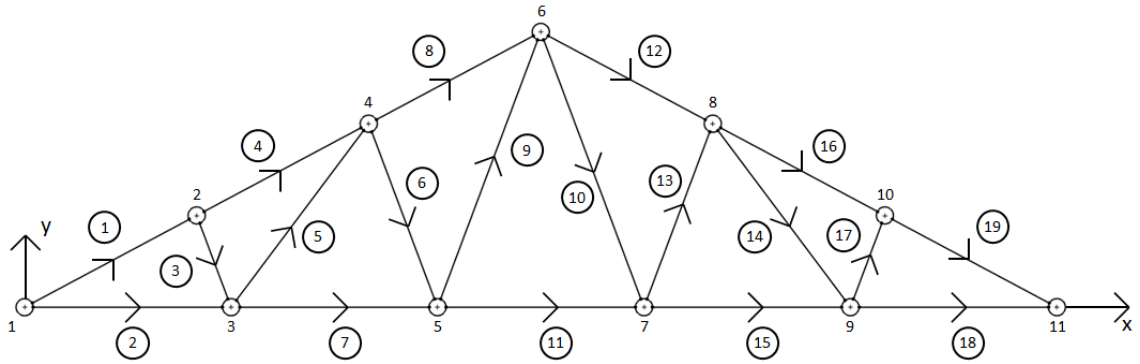


Figure 2. Roof Truss Labeling Scheme

Given the labeling scheme described above, the data outputted by the MATLAB code can be interpreted. Table 2 shows the directional displacements and total displacements at each node in units of centimeters. The directional displacement vector has 22 lines corresponding to the 22 global DOF. For instance, DOF 1 and 2 indicate the x- and y-direction displacement at node 1, respectively. On the other hand, the total displacement vector has 11 lines corresponding to 11 nodes. Similarly, Table 3 shows the stress, in units of megapascals, and strain in each truss element. These vectors have 19 lines each, corresponding to the 19 truss elements that make up the system. See Figure 2 to identify the truss element referred to by each line.

Table 2. Directional and Total Displacements for -6kN Load

Node	DOF	Directional Displacement (cm)	Total Displacement (cm)
1	1	0	0
	2	0	
2	3	0.7060	1.9666
	4	-1.8355	
3	5	0.2250	2.0069
	6	-1.9942	
4	7	0.6814	2.3673
	8	-2.2671	
5	9	0.4050	2.3410
	10	-2.3057	
6	11	0.4725	2.2830
	12	-2.2336	
7	13	0.5400	2.3681
	14	-2.3057	
8	15	0.2636	2.2823
	16	-2.2671	
9	17	0.7200	2.1202
	18	-1.9942	
10	19	0.2390	1.8510
	20	-1.8355	
11	21	0.9450	0.9450
	22	0	

Table 3. Stress and Strain in Each Truss Element for -6kN Load

Element	Stress (MPa)	Strain
1	-12.750	-0.001417
2	11.250	0.001250
3	-2.1360	-0.0002373
4	-11.900	-0.001322
5	2.5000	0.0002778
6	-3.2040	-0.0003560
7	9.0000	0.001000
8	-8.9250	-0.0009917
9	3.2040	0.0003560
10	3.2040	0.0003560
11	6.7500	0.0007500
12	-8.9250	-0.0009917
13	-3.2040	-0.0003560
14	2.5000	0.0002778
15	9.0000	0.001000
16	-11.900	-0.001322
17	-2.1360	-0.0002373
18	11.250	0.001250
19	-12.750	-0.001417

Next, the reaction forces were calculated via MATLAB code. Reaction forces for the system will only be present at nodes 1 and 11 since pinned joints exist at these nodes. Reaction forces are zero at unsupported nodes even if an external force is applied to them due to the reaction being internal. Still, the force outputs given by MATLAB at nodes 1 and 11 need to be modified to account for the external -6kN vertical load that is ignored by the code due to inactive DOF (i.e., nodes 1 and 11 contain zero displacement). The forces output by MATLAB, in units of Newtons, are shown in Table 4. The calculated reaction forces after accounting for the external vertical load are also shown.

Table 4. Displacement Reaction Forces at Node 1 and Node 11

Node	DOF	MATLAB Force Output (N)	Reaction Force (N)
1	1	-4.3688e-11	-4.3688e-11
	2	15000	21000
11	21	0	0
	22	15000	21000

Then, an ANSYS simulation of the roof truss was used to find the displacements and reaction forces at each joint, as well as the stress in each truss element. The ANSYS results for X- and Y-directional deformation, in units of centimeters, are shown in Figure 3. Additionally, the total deformation values are shown in Figure 4. Figure 5 shows the stress, in units of megapascals, in each truss element. Although graphical results are not shown for strain, given that ANSYS does

not have a solution function for this variable, it was calculated by hand and compared to MATLAB results using the following equation:

$$\varepsilon = \frac{\sigma}{E} = \frac{Direct\ Stress\ [MPa]}{9000\ MPa}$$

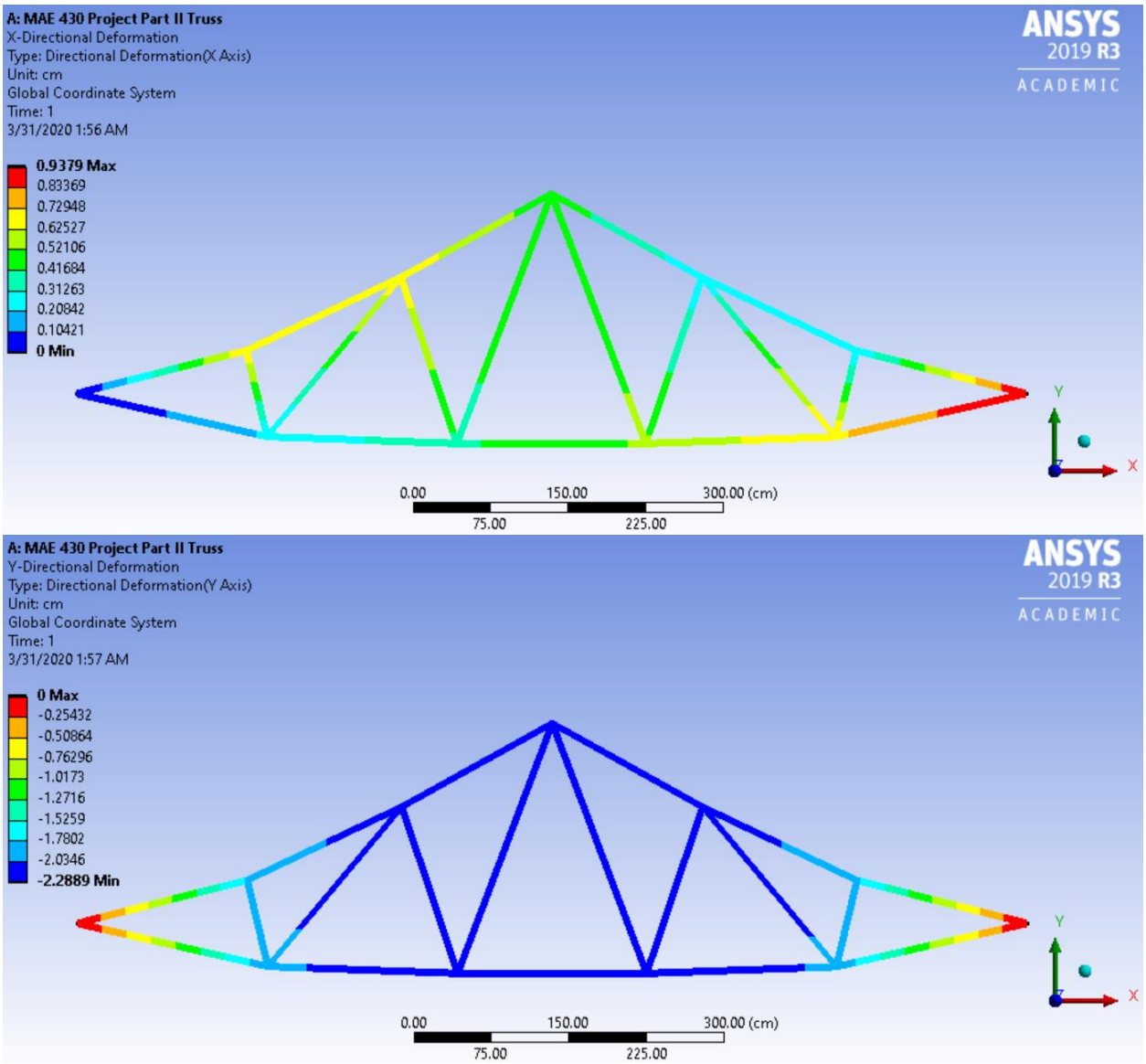


Figure 3. X- and Y-Directional Deformation for -6kN Load

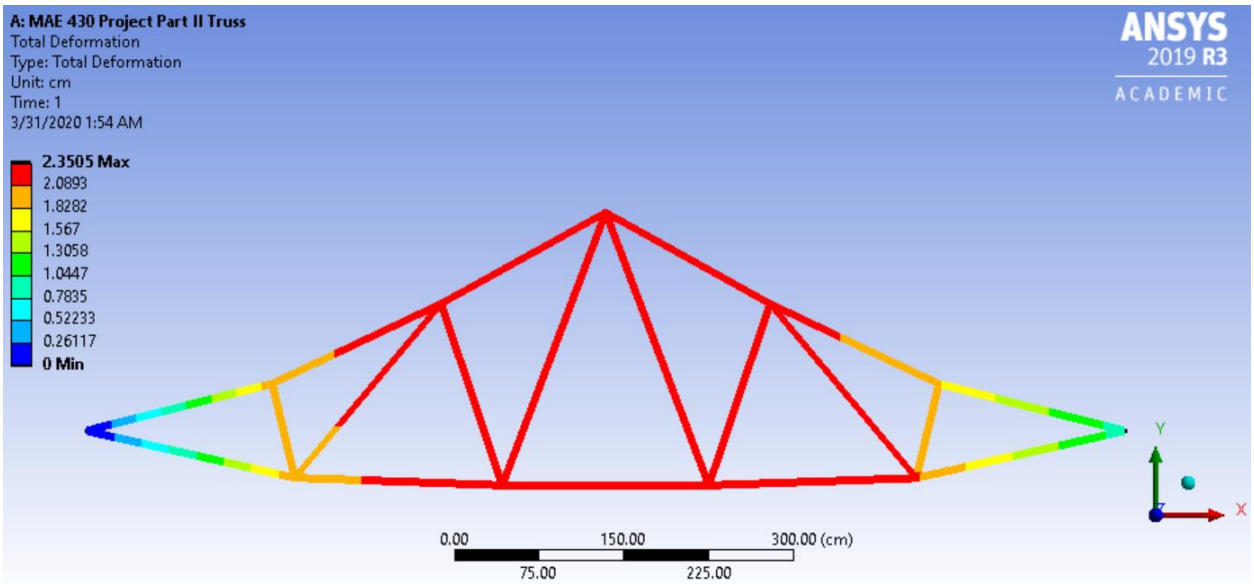


Figure 4. Total Deformation for -6kN Load

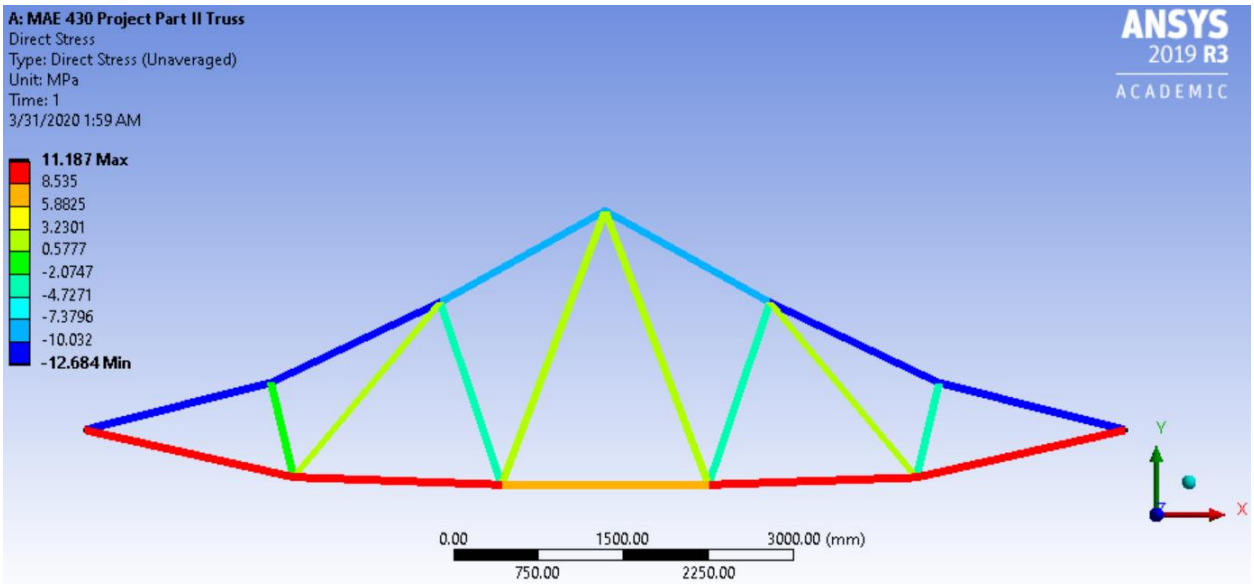


Figure 5. Stress in each Truss Element for -6kN Load

Lastly, reaction forces were computed at nodes 1 and 11 to compare with the reaction forces computed by MATLAB and shown in Table 4. Figure 6 shows the forces computed using ANSYS at both nodes. As the tables in the ANSYS results show, both methods produce similar reaction forces. Namely, nodes 1 and 11 have a vertical reaction force of approximately 21kN.

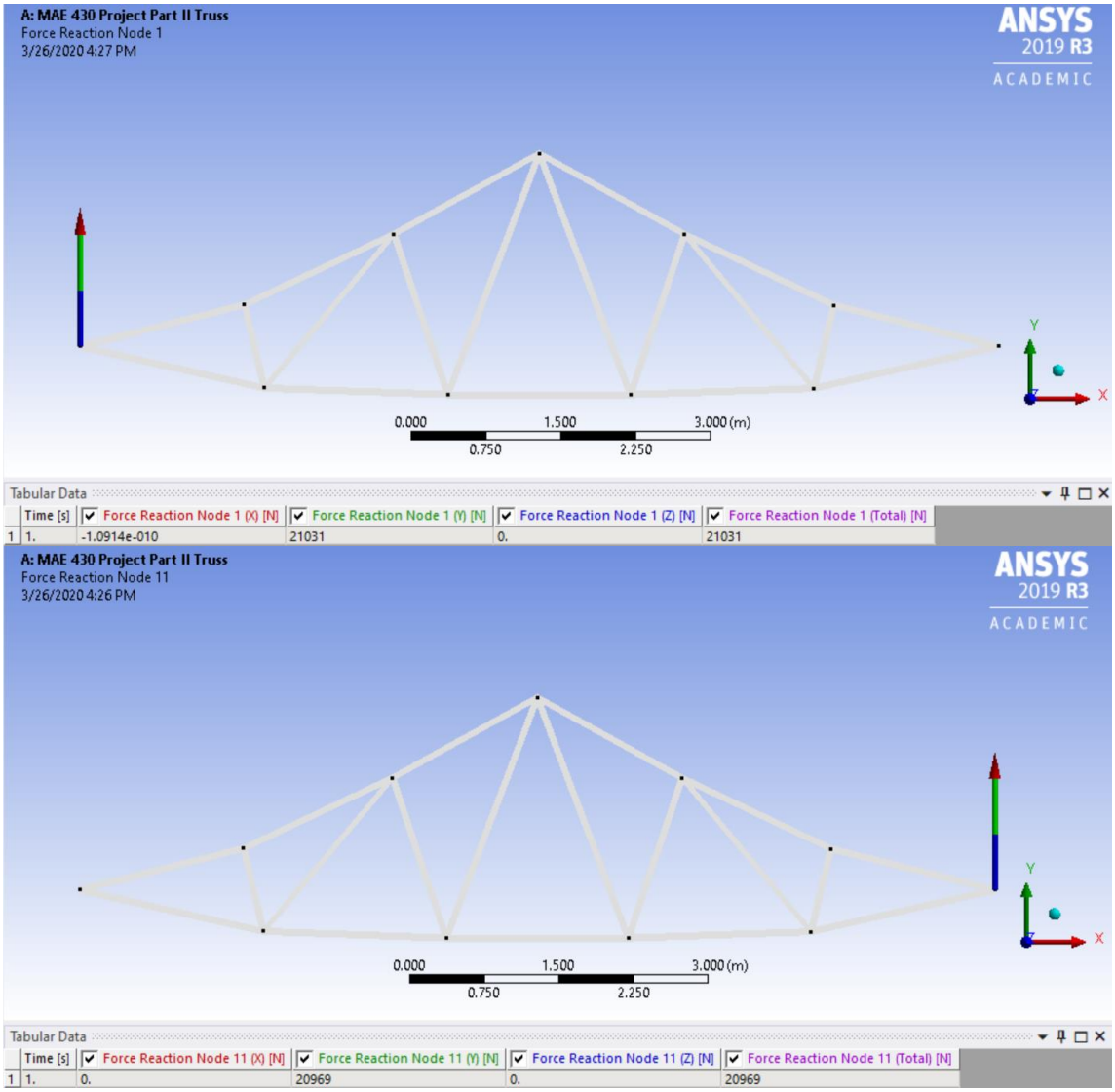


Figure 6. Reaction Forces at Node 1 and Node 11 for -6kN Load

In order to compare the results given by MATLAB and ANSYS for displacements, stress and strain, one internal truss element was chosen for comparison. Element 10 between nodes 6 and 7 was chosen. Table 5 lists results for both methods of computation along with percent differences.

Table 5. Comparison of MATLAB and ANSYS Results for Element 10

Computed Value	MATLAB Result	ANSYS Result	Percent Difference
X-Direction Disp. at Node 6	0.4725 cm	0.52106 cm	9.77%
Y-Direction Disp. at Node 6	-2.2336 cm	-2.2889 cm	2.45%
X-Direction Disp. at Node 7	0.5400 cm	0.62527 cm	14.6%
Y-Direction Disp. at Node 7	-2.3057 cm	-2.2889 cm	0.731%
Total Disp. at Node 6	2.2830 cm	2.3505 cm	2.91%
Total Disp. at Node 7	2.3681 cm	2.3505 cm	0.746%
Stress at Element 10	3.2040 MPa	3.2301 MPa	0.811%
Strain at Element 10	0.0003560	0.0003589	0.811%

The last part of this problem involves using MATLAB code to find the maximum load capacity for the truss. The current loading is -6kN as shown in Figure 1. We are given three design criteria that must be satisfied for any proposed loading for a safety factor of 1.5. The maximum tensile stress in any truss member cannot exceed the ultimate tensile strength divided by the safety factor. The maximum compressive stress in any truss member cannot exceed the compressive yield strength divided by the safety factor. Furthermore, the maximum deflection of roof truss cannot exceed 3.75cm. These conditions are summarized in Table 6.

Table 6. Design Criteria for Finding Maximum Load Capacity of Truss

Maximum Tensile Stress	$\sigma_{max,tensile} \leq 64.333MPa$
Maximum Compressive Stress	$\sigma_{max,compressive} \geq -24.667MPa$
Maximum Deflection	$\delta_{max,y} \leq 3.75cm$

A few modifications were made to the MATLAB code in order to compute the maximum load capacity. Namely, a while loop was inserted to iteratively calculate all the stress and displacement values of the truss and compare them with the design criteria values. A step was incorporated to minimize the -6kN load until one of the design criteria was exceed, at which point the code would break and MATLAB would generate the loading before then. An initial loading and step value of -3kN and -100N were specified. Once a better estimate of the maximum load capacity was obtained, the initial loading and step values were changed to -9kN and -1N, respectively. See appendix for the detailed code. *As a result, the maximum load capacity given by MATLAB was found to be -9.758kN with deflection as the limiting criterion.*

Similarly, a direct optimization simulation was created within ANSYS to minimize the loading of -6kN while satisfying the design criteria shown in Table 6. The result of this optimization code is shown in Figure 7. Since the loading is shared among the seven nodes along the top of the truss, the maximum load capacity can be calculated as follows: -68811N/7 = -9830.1N. *Hence, the maximum load capacity given by ANSYS was found to be -9.830kN.* This is a difference of approximately 70N between the two methods of optimization, or a 0.737% difference. Furthermore, the limiting criterion for both methods was found to be the deflection of the truss. Figure 8 shows the y-deformation and direct stress for the optimized loading scenario. Clearly, the truss reaches its maximum deflection of 3.75cm before exceeding the maximum stress criteria.

How significant is the proposed increase in loading? It might not seem like a lot to increase from -6kN to -9.830kN. Still, this is an increase of 26.8kN over the entire structure, which is equivalent to the weight of two small vehicles. Although the maximum tensile and compressive stresses are not exceeded by the proposed maximum load, additional analysis could be done on buckling effects to enhance the solution. Perhaps the truss will fail due to buckling before reaching its deflection limit.

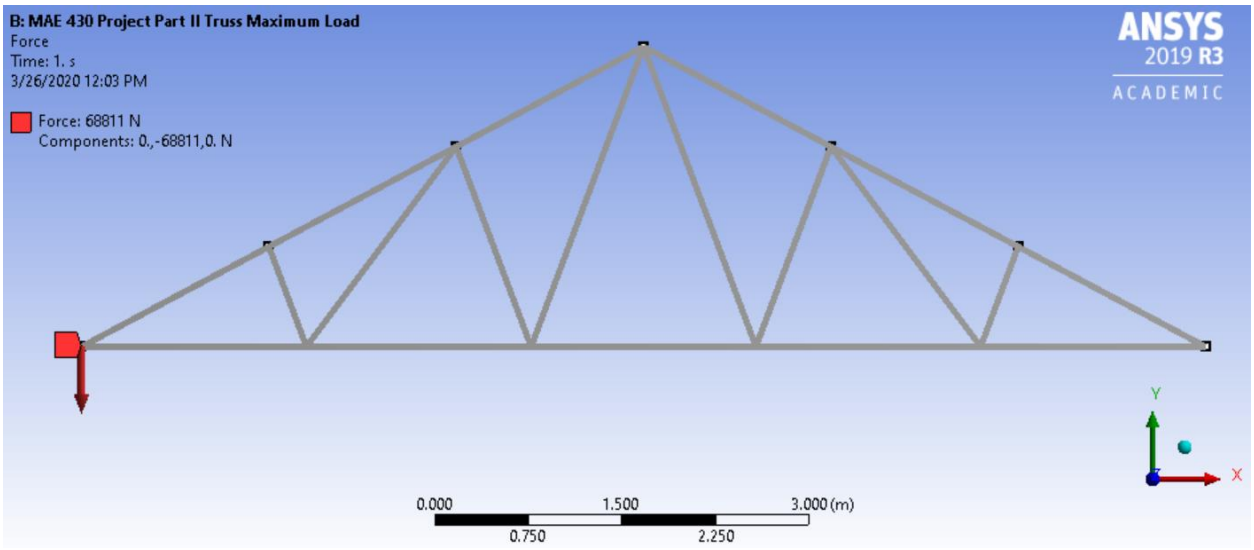
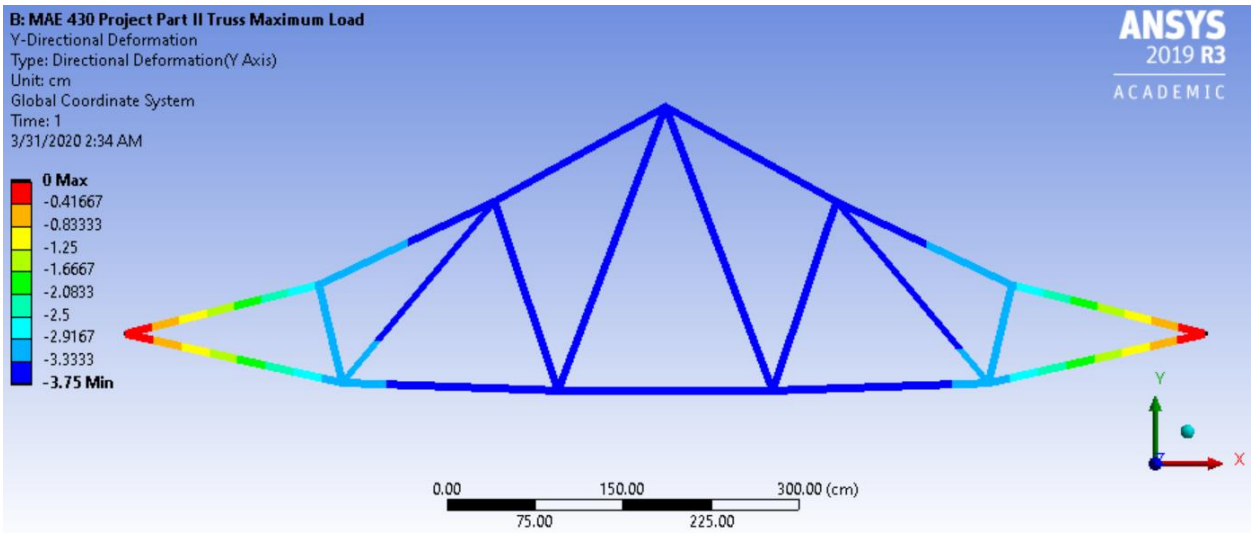


Figure 7. Maximum Load Capacity per ANSYS Optimization



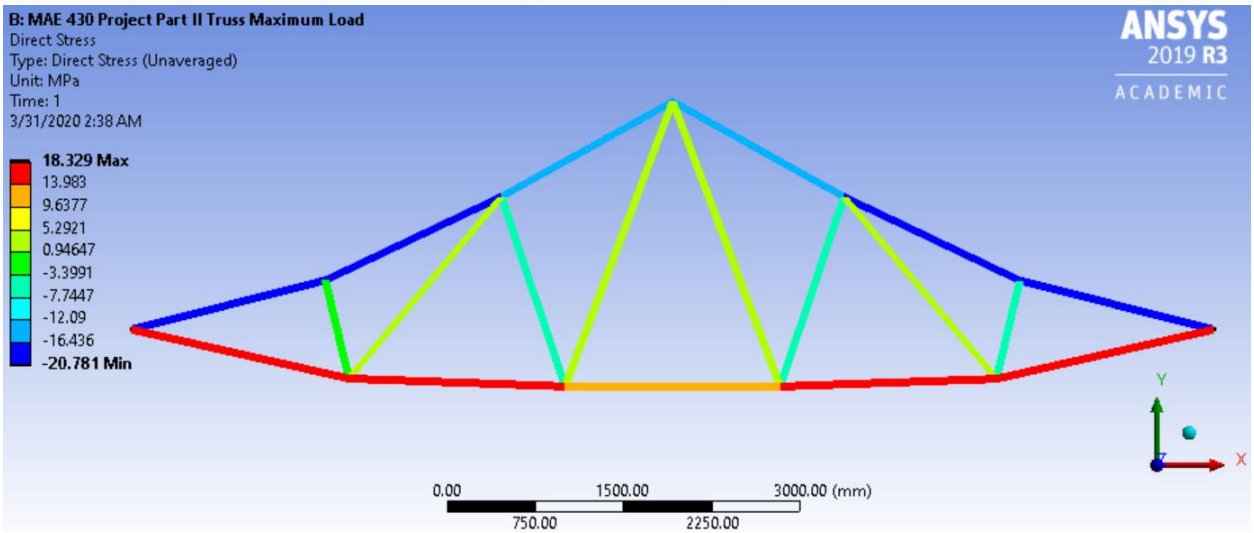


Figure 8. Y-Deformation and Direct Stress for Maximum Load Capacity

Appendix

```
%% MATLAB Codes for MAE 430 Project 1 Part II
% Code to calculate displacements and reaction forces at each joint, stress
% and strain in each truss; all functions called are found at the end of
script
clear all
clc
% E: modulus of elasticity [Pa]
% A: area of cross section [m^2]
% L: length of bar [m]
% EA: Axial force [N]
E=9e9; A=0.0025; EA=E*A;
% generation of coordinates and connectivities
numberElements=19;% number of elements
numberNodes=11; % number of nodes
elementNodes=[1 2;1 3;2 3;2 4;3 4;4 5;3 5;4 6;5 6;6 7;5 7;6 8;7 8; ...
    8 9;7 9;8 10;9 10;9 11;10 11];% Element connectivity: Each element has
% two nodes element 1 is composed of node 1 and 2; element 2 is composed
% of node 1 and 3, etc.
nodeCoordinates=[0 0;1.5 0.8;1.8 0;3 1.6;3.6 0; 4.5 2.4; ...
    5.4 0;6 1.6;7.2 0;7.5 0.8;9 0];
% x, y cooridnates for each node 1; here node 1 (0,0),
% node 2 (1.5,0.8), node 3 (1.8,0), etc.
xx=nodeCoordinates(:,1); % x-cooridnates for three nodes
yy=nodeCoordinates(:,2); % y-cooridnates for three nodes
% for structure we need to specify the following:
% displacements: displacement vector
% force : force vector
% stiffness: stiffness matrix
GDof=2*numberNodes; % GDof: total number of Global degrees of freedom
% since each node can move in the x- and y-direction
displacements=zeros(GDof,1);% displacement vectors with inital value of zero
force=zeros(GDof,1); % force vector with inital value of zero
% Loading condition
% Applied load at D.O.F 2,4,8,12,16,20,22 [N]
force([4 8 12 16 20],1)=-6000;
% computation of the global stiffness matrix for the truss system
stiffness=formStiffness2Dtruss(GDof,numberElements,...
elementNodes,numberNodes,nodeCoordinates,xx,yy,EA);
stiffness; % verify global stiffness matrix is symmetric by printing to
screen
% Call the function of formStiffness2Dtruss, the input varialbe is GDof,
% numberElements, elementNodes, numberNodes,nodeCoordinates, xx, yy, and EA
% boundary conditions and solution
prescribedDof=[1 2 22]';% prescribed B.C at D.O.F of 1,2, and 22
% solution
displacements=solution(GDof,prescribedDof,stiffness,force);% call solution
[stress, strain]=stresses2Dtruss(numberElements,elementNodes,...
xx,yy,displacements,E); % compute strain and stress in each truss
displacements % print displacements to screen
stress % print truss stresses to screen
strain % print truss strains to screen
reactionforces=outputDisplacementsReactions(displacements, ...
    stiffness,GDof,prescribedDof); % call function to compute reaction forces
% calculate total deflection at each node
for i=1:numberNodes
```

```
    totalDisplacement(i,1)=norm([displacements(2*i-1,1) displacements(2*i)]);
end
totalDisplacement % print total displament at each node to screen
reactionforces % print reaction forces to screen
% End code 1

%% Optimization of Truss
% Code to find the maximum load capacity given a safety factor of 1.5
% calls the same functions as code 1 above but incorporates a while loop
clear all
clc
% E: modulus of elasticity [Pa]
% A: area of cross section [m^2]
% L: length of bar [m]
% EA: Axial force [N]
E=9e9; A=0.0025; EA=E*A;
% generation of coordinates and connectivities
numberElements=19;% number of elements
numberNodes=11; % number of nodes
elementNodes=[1 2;1 3;2 3;2 4;3 4;4 5;3 5;4 6;5 6;6 7;5 7;6 8;7 8; ...
    8 9;7 9;8 10;9 10;9 11;10 11];% Element connectivity: Each element has
% two nodes element 1 is composed of node 1 and 2; element 2 is composed
% of node 1 and 3, etc.
nodeCoordinates=[0 0;1.5 0.8;1.8 0;3 1.6;3.6 0; 4.5 2.4; ...
    5.4 0;6 1.6;7.2 0;7.5 0.8;9 0];
% x, y cooridnates for each node 1; here node 1 (0,0),
% node 2 (1.5,0.8), node 3 (1.8,0), etc.
xx=nodeCoordinates(:,1); % x-cooridnates for three nodes
yy=nodeCoordinates(:,2); % y-cooridnates for three nodes
GDof=2*numberNodes; % GDof: total number of Global degrees of freedom
% since each node can move in the x- and y-direction
SF=1.5; % safety factor
UTS=96.5e6; % ultimate tensile strength [Pa]
CYS=37e6; % compressive yield strength [Pa]
maxTS=UTS/SF; % maximum tensile strength [Pa]
maxCS=CYS/SF; % maximum compressive strength [Pa]
% initiate while loop to find maximum load capacity
Loop=true;
load=-9000; % initializes the load
while Loop==true
    displacements=zeros(GDof,1);% displacement vectors with inital value of
zero
    force=zeros(GDof,1); % force vector with inital value of zero
    % Loading condition
    % Applied load at D.O.F 2,4,8,12,16,20,22 [N]
    force([4 8 12 16 20],1)=load;
    step=-1;
    % computation of the global stiffness matrix for the truss system
    stiffness=formStiffness2Dtruss(GDof,numberElements,...
    elementNodes,numberNodes,nodeCoordinates,xx,yy,EA);
    % boundary conditions and solution
    prescribedDof=[1 2 22]';% prescribed B.C at D.O.F of 1,2, and 22
    % solution
    displacements=solution(GDof,prescribedDof,stiffness,force);% call
solution
    % compute stress and strain in each truss
    [stress, strain]=stresses2Dtruss(numberElements,elementNodes,...
```

```
    xx,yy,displacements,E);
    % calculate total deflection at each node
    for i=1:numberNodes
        totalDisplacement(i,1)=norm([displacements(2*i-1,1)
displacements(2*i)]);
    end
    % set criteria to find maximum load capacity
    if max(stress)>maxTS % criteria 1
        fprintf('Maximum Tensile Stress Surpassed\n');
        load=load-step;
        break;
    end
    if abs(min(stress))>maxCS % criteria 2
        fprintf('Maximum Compressive Stress Surpassed\n');
        load=load-step;
        break;
    end
    if max(abs(displacements))>0.0375 % criteria 3
        fprintf('Maximum Deflection Surpassed\n');
        load=load-step;
        break;
    end
    load=load+step;
end
% End code 2
% END SCRIPTS FOR MAE 430 PROJECT 1 PART II

function [stiffness]=...
formStiffness2Dtruss(GDof,numberElements,...
elementNodes,numberNodes,nodeCoordinates,xx,yy,EA);
stiffness=zeros(GDof);
% computation of the system stiffness matrix
for i=1:numberElements
% elementDof: element degrees of freedom (Dof)
indice=elementNodes(i,:) ;
elementDof=[ indice(1)*2-1 indice(1)*2 indice(2)*2-1 indice(2)*2] ;
xa=xx(indice(2))-xx(indice(1)); % x distance from start to end node of
element
ya=yy(indice(2))-yy(indice(1)); % y distance from start to end node of
element
length_element=sqrt(xa*xa+ya*ya); % calculate length of element "i"
C=xa/length_element; % cosine value for element "i"
S=ya/length_element; % sine value for element "i"
k1=EA/length_element*...
[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]; % create 4x4 element matrix
stiffness(elementDof,elementDof)=stiffness(elementDof,elementDof)+k1;
% combine local element matrices to the global stiffness matrix
end

function [displacements]=solution(GDof,prescribedDof,stiffness,force)
% function to find solution in terms of global displacements
activeDof=setdiff([1:GDof],[prescribedDof]);
% the active D.O.F means that which D.O.F is free to move,
% in this case D.O.F 3-21 are free to move
U=stiffness( activeDof , activeDof)\force(activeDof);
displacements=zeros(GDof,1);
```

```
displacements(activeDof)=U; % assign your solved solution to your
displacements arrays

function [stress, strain]=stresses2Dtruss(numberElements,elementNodes,...
    xx,yy,displacements,E)
for i=1:numberElements
    indice=elementNodes(i,:);
    elementDof=[ indice(1)*2-1 indice(1)*2 indice(2)*2-1 indice(2)*2] ;
    xa=xx(indice(2))-xx(indice(1));
    ya=yy(indice(2))-yy(indice(1));
    length_element=sqrt(xa*xa+ya*ya);
    C=xa/length_element;
    S=ya/length_element;
    stress(i)=(E./length_element)*[-C -S C S]*displacements(elementDof,1);
    strain(i)=stress(i)/E;
end
stress=stress';
strain=strain';

function [reactionforces] = outputDisplacementsReactions(displacements, ...
    stiffness,GDof,prescribedDof)
% function to find reaction forces at nodes where displacement is zero
activeDof=setdiff([1:GDof],[prescribedDof]);
% the active D.O.F means that which D.O.F is free to move,
% in this case D.O.F 3-21 are free to move
R=stiffness(prescribedDof,activeDof)*displacements(activeDof,1);
reactionforces=zeros(GDof,1);
reactionforces(prescribedDof)=R;
% compute a column vector of reaction forces
```