Project 2 – MANE 6710 Numerical Design Optimization

Unique ID: 5284

Table of Contents

List of Figures

Introduction

The main objective of this project is to design the cross-sectional profile of a spar of an aircraft wing. The idea is to arrive at a design which leads to the least weight possible for the structure given certain design parameters. The optimal design that comes as a result should not fail under the loading conditions and in fact it must be manufacturable.

Choice of Design Variable

With the cross section fixed to be a circular annulus, the most obvious choice of design variables are the Radii of the annulus along the length of the spar. The parameterization of design variable chosen in this project is as follows: $R = \{R_{out}^1\ R_{in}^1\ R_{out}^2\ R_{in}^2\ \ldots .R_{out}^n\ R_{in}^n\}' - \{R_{out}^i, R_{in}^i\}\ i{:}\ 1(1)n$

The length of the spar can be discretized into n elements (N_{elem}) and with a pair of design variables $\{R_{out}^i, R_{in}^i\}$ at each node as shown in Figure 1.

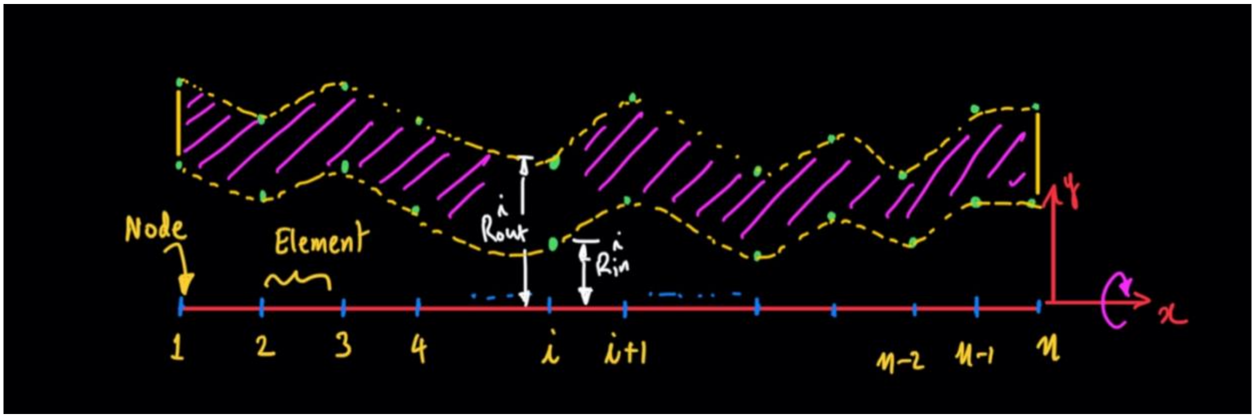

Figure 1. Cross- sectional Profile of the spar

Objective Function and Constraints

The objective of this project is to optimize the design of the spar's cross-section that has the least weight possible.
This cross-sectional profile needs to adhere to certain design parameters such as:

1. Semi Span Length – 7.5 m
2. Shape of Cross section – Circular Annulus
3. Material – Carbon Fiber composite – {Density $\rho = 1600\ kg/m^3$;Young's modulus $E = 70\ GPa$; Yield Compressive/Tensile stress $\sigma_{max} = 600\ MPa$}
4. Aircraft operational weight – $500\ kg$
5. Loading – 2.5 g with force distribution linearly varying along the length (maximum load at the root and no load at the tip)

With these parameters, the objective function will be the Mass or the Weight of the spar which can be calculated as $m = \rho V\ kg; V - volume\ (m^3), \rho - density\ (kg/m^3)$. Hence this design problem can now be formulated as:

$$\min_R \rho V(R) \qquad \text{Equation 1}$$

This minimization problem is subject to constraints that stem from the manufacturing side of the spar, and they are:

1. $c_1(R) = R_{out} \leq 5cm$ along the entire length of the spar
2. $c_2(R) = R_{in} \geq 1cm$ along the entire length of the spar
3. $c_3(R) = R_{out} - R_{in} \geq 2.5mm$ along the entire length of the spar
4. $c_4(R) = \sigma \leq \sigma_{max}$ along the entire length of the spar

MATLAB's in-built optimizer $fmincon(obj, a_0, A, b, Aeq, beq, lb, ub, Nonlinear, options)$ is used to perform the optimization of spar cross-section.

Calculation of Volume

Volume is generally calculated by the integral $\int_0^L A(x)\,dx$ where, $A(x)$ is the cross-section along the length of the spar. With very fine discretization of the length of spar into N_{elem}, this integral can be approximated as:

$$\sum_{i=1}^{N_{elem}} \frac{1}{2}\left\{\pi\left({R_{out}^{i}}^2 - {R_{in}^{i}}^2\right) + \pi\left({R_{out}^{i+1}}^2 - {R_{in}^{i+1}}^2\right)\right\}\delta x_i \qquad \text{Equation 2}$$

This calculates the average area of the cross-section on two adjacent node points and multiplies it with the distance between these two node points δx_i. This is a variational form of trapezoidal integration technique.

Calculation of Constraints

Setting up Linear Inequality constraints

Constraints $c_1(R), c_2(R), c_3(R)$ are set up as Linear Inequality constraints of the format $fmincon$ requires it in - $AR \leq b$ where A is the matrix that contains the linear mapping between the design variables R and constraint values b.

Setting up Nonlinear constraints

This is the most important aspect of this project as the computation of normal stress at each cross-section requires the use of numerical methods like Finite Element Methods to calculate it. The Area moment of Inertia for each cross-section along the length is,

$$I_{yy}^{i} = \frac{1}{4}\pi({R_{out}^{i}}^{4} - {R_{in}^{i}}^{4}) \qquad \text{Equation 3}$$

Then, the force which varies linearly along the length of the wing is calculated at each Node point as

$$\int_{0}^{L} f\, dx = \frac{2.5 * Operational\ Load(N)}{2}\ ; F^{i} = f x^{i}\ N \qquad \text{Equation 3}$$

The operational load considered in this problem is Operational Weight - $500kg \times 9.81\ m/s^2$
Using the Area Moment of Inertia and the force on each node, the vertical displacement and the angular displacement of the wing at each node is calculated using the FEM ($CalcBeamDisplacement.m$). Then, it gets easier to solve for the Normal stress σ_i along the length of the spar. This is calculated using $CalcBeamStress.m$ function. Most important aspect of any optimization problem is to perform optimization in scales that are the same for all the parameters that are involved in this process. Since, the calculated maximum σ_i at each node is at large scales compared to the value of the objective function or the design variables, this has been normalized and the final nonlinear inequality constraint $c_4(R)$ is written as

$$\frac{|\sigma_i|}{\sigma_{max}} \leq 1 \qquad \text{Equation 4}$$

Setting up Optimization Problem

With the objective and the constraints set up, it is now important to calculate accurate gradients of the objective and the constraints for successful optimization of the design. Complex step gradient method is used to compute the gradients of the objective and the inequality constraints. With a very small complex step $h = 10^{-60}$,the round off errors caused by the central difference approximation can be avoided.
Two functions, $Calc\_objGrad(X)$ and $Calc\_consJac(X, N_{nodes}, L, E, F, \sigma_{max})$ are used to compute the gradients of the objective function and the nonlinear constraint function.

Number of elements - 100
Initial point chosen - $[0.03 \quad 0.015 \quad 0.03 \quad 0.015 \quad 0.03 \cdots\cdots\cdots 0.03 \quad 0.015]'_{202\times1}$

Options chosen for $fmincon$ –

i. Choice of Algorithm – ‘Sequential Quadratic Programming (SQP)’
ii. Gradients – Provided and computed using Complex Step Algorithm

Initial weight of the spar was $\mathbf{25.446900494077372\ kg}$

Initial Design choice is shown in Figure 2

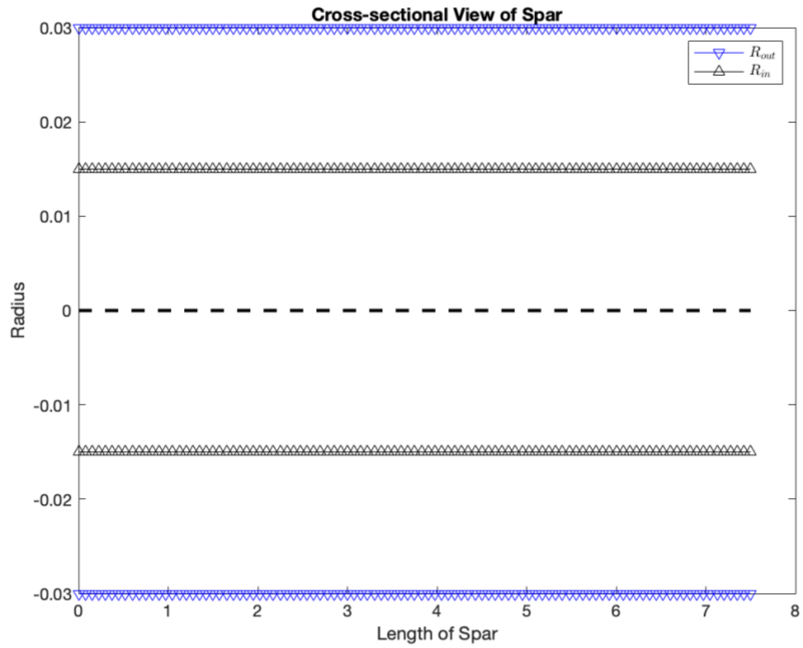

Figure 2. Initial Design Parameterization

After optimization is performed, the optimal profile of the cross-section is shown in Figure 3

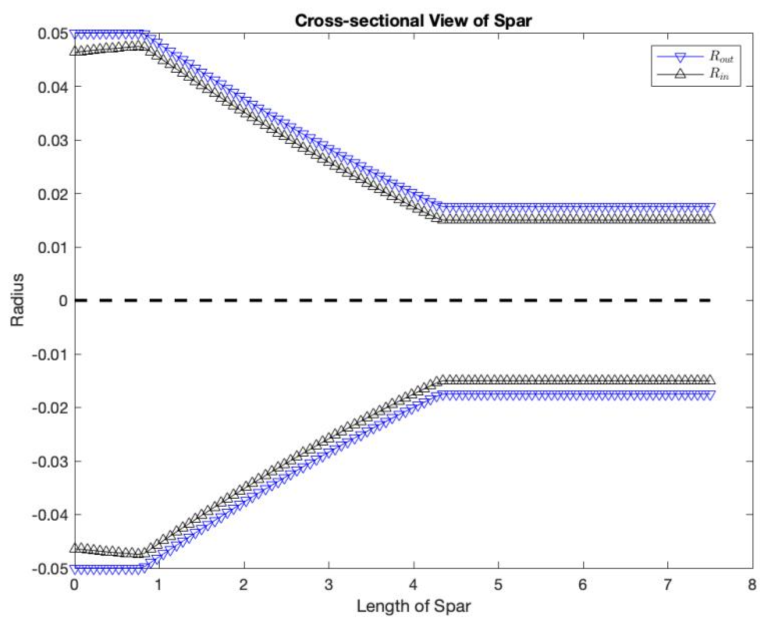

Figure 3. Optimal design of Spar Cross-section

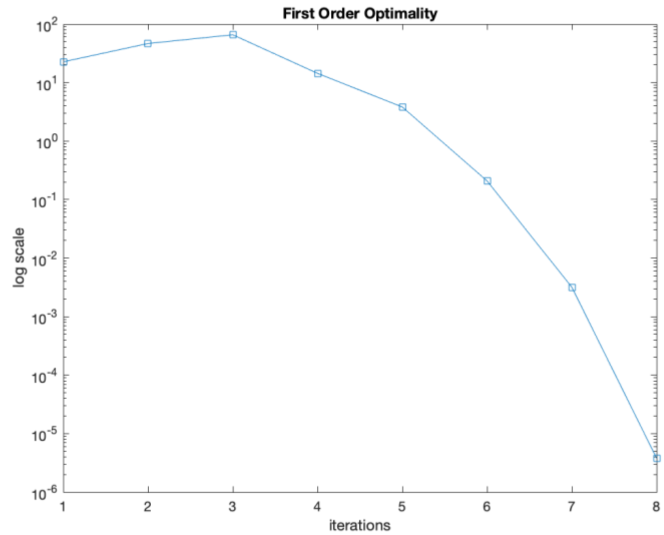

Figure 4. First Order Optimality condition

Figure 4 shows the first order optimality condition to decrease and this conveys the fact that the optimizer has converged to a local minimizer which also satisfies the constraints.

The final weight of this optimal design is $\mathbf{5.273609498808303\ kg.}$ This is a $\mathbf{79.28}$% reduction of weight from the initial design.

Results

A convergence analysis was performed to find if the mass estimate computed in this project improves as the number of elements (N_{elem}) are increased. As more cross-sections are added along the length elements, approximate volume calculated using the trapezoidal integral method improves as δx_i decreases and a small increase in the mass computed can be noticed as shown in Figure 5.

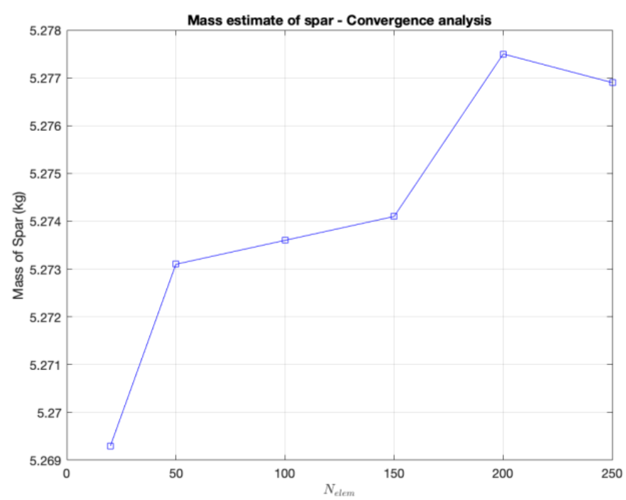

Figure 5. Mass estimates (v) number of elements

Optimized profile undergoes the following deformation, stress under the application of a linearly varying force along the length of the semi spar of the wing as shown in Figure 6.

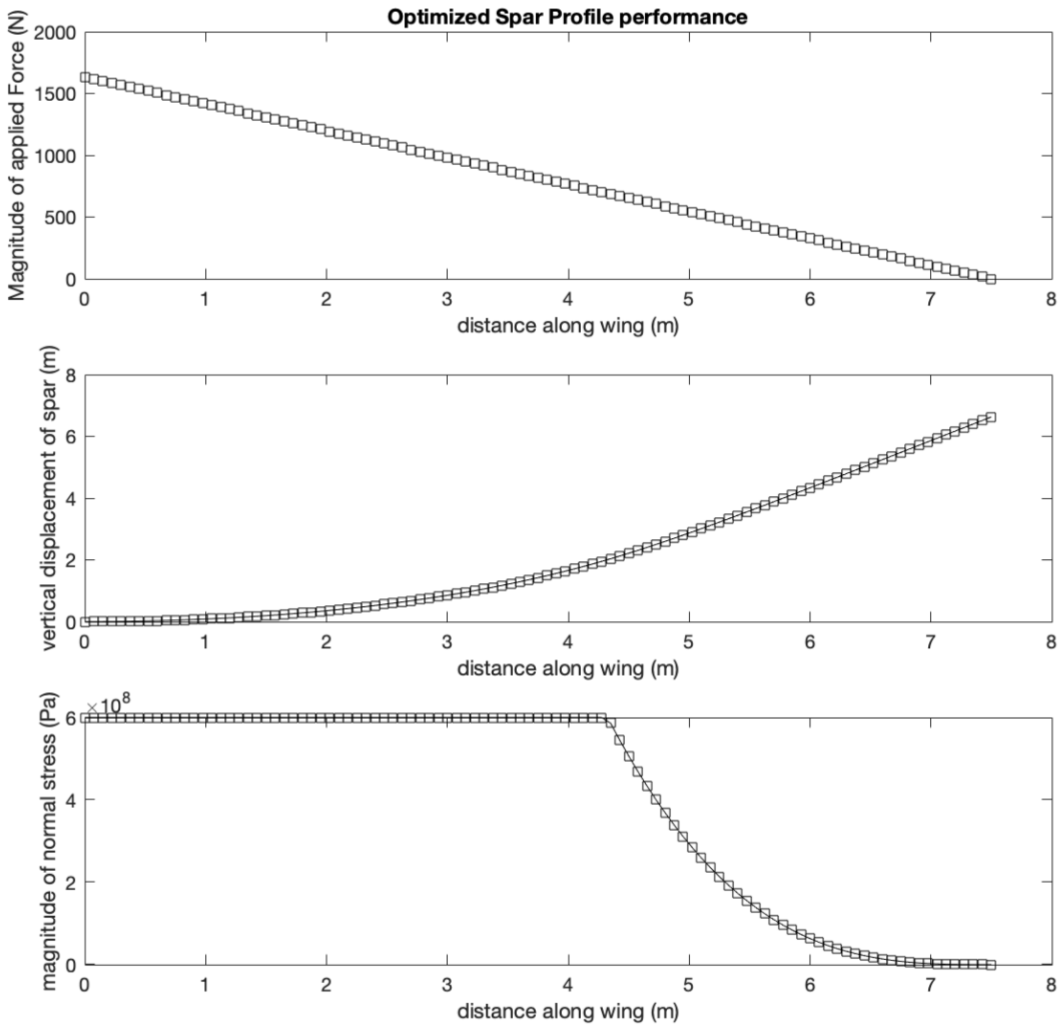

Figure 6. Optimized Spar Performance

This profile for the spar not only has decreased its mass but has also adhered to all the manufacturing constraints specified for the design of the spar. This is the final optimized profile achieved using this parameterization for the choice of design variables $\{R_{out}^{i}, R_{in}^{i}\}\ i: 1(1)n$.

Appendix

The MATLAB code written for this project has been attached in this section.

Contents


```
clc
clear all
```

Inital point of design variable

```
Nelem = 250; % number of elements along spar
r_out = 3e-2; % m - Outer radius
r_in = 1.5e-2; % m - Inner radius
L = 7.5; %m - Length of spar
x = (0:L/Nelem:L)'; % discretization of length of spar
slope1 = 0;
slope2 = 0;
X0 = ones((2*(Nelem+1)),1); % Inital design variable
k = 1;
for i=1:2:(2*Nelem+1)
    X0(i) = r_out+x(k)*slope1;
    X0(i+1) = r_in+x(k)*slope2;
    k = k+1;
end
```

Setting up Linear Inequality constraint

```
Nnodes = Nelem + 1;
% rin > 1cm ---> -rin < -1cm
A1 = zeros(Nnodes,2*Nnodes);
k=2;
for i=1:((Nnodes))
    A1(i,k) = -1;
    k = k+2;
end
b1 = -1e-2*ones(Nnodes,1);

% rout - rin > 2.5mm ---->  -rout + rin < -2.5mm
A2 = zeros((Nnodes),2*(Nnodes));
k = 1;
for i=1:((Nnodes))
    A2(i,k) = -1;
    A2(i,k+1) = 1;
    k = k+2;
end
b2 = -2.5e-3*ones(Nnodes,1);

% rout < 5cm
A3 = zeros((Nnodes),2*(Nnodes));
k=1;
for i=1:((Nnodes))
    A3(i,k) = 1;
    k = k+2;
```

```
end
b3 = 5e-2*ones(Nnodes,1);

% -rout + rin < 0 ---> rout > rin
A4 = zeros(Nnodes,2*Nnodes);
k=1;
for i=1:Nnodes
    A4(i,k) = -1;
    A4(i,k+1) = 1;
    k = k+2;
end
b4 = zeros(Nnodes,1);
A = [A1;A2;A3;A4];
b = [b1;b2;b3;b4];

lb = ones(2*Nnodes,1);
ub = lb;
lb(2:2:end) = 0.01;
lb(1:2:end) = 0.0175;
ub(2:2:end) = 0.0475;
ub(1:2:end) = 0.05;
```

Running Optimization

```
options = optimoptions('fmincon','Display','iter-detailed','Algorithm','sqp',...
    'SpecifyObjectiveGradient',true,'SpecifyConstraintGradient',true);

[X_opt,fvalue,~,op,~,grad]=fmincon(@obj_func,X0,A,b,[],[],lb,ub,@NonLnCons,options);
```

Plotting section

```
Iyy = Calc_Iyy(X_opt,Nnodes);
force = Calc_force(x,500,L);
[u] = CalcBeamDisplacement(L, 70e9,Iyy, force, Nelem);
zmax = X_opt(1:2:end);
[sigma] = CalcBeamStress(L, 70e9, zmax, u, Nnodes-1);
norm_sigma = sigma/600e6;
figure
plot(x,norm_sigma,'ks-')
xlabel('distance along wing')
ylabel('magnitude of normal stress')

figure
plot(x,sigma,'ks-')
xlabel('distance along wing')
ylabel('magnitude of normal stress')

figure
plot(x,u(1:2:2*(Nelem+1)),'ks-');
xlabel('distance along wing')
ylabel('vertical displacement of spar')

figure
plot(x,X_opt(1:2:end),'bv-');
hold on;
plot(x,X_opt(2:2:end),'k^-');
plot(x,-X_opt(1:2:end),'bv-');
plot(x,-X_opt(2:2:end),'k^-');
```

```
plot(x,0*X_opt(1:2:end),'k--','lineWidth',2);
xlabel('Length of Spar');
ylabel('Radius');
legend('$R_{out}$','$R_{in}$','Interpreter','latex');
title ('Cross-sectional View of Spar')

figure
plot(x,X0(1:2:end),'bv-');
hold on;
plot(x,X0(2:2:end),'k^-');
plot(x,-X0(1:2:end),'bv-');
plot(x,-X0(2:2:end),'k^-');
plot(x,0*X0(1:2:end),'k--','lineWidth',2);
xlabel('Length of Spar');
ylabel('Radius');
legend('$R_{out}$','$R_{in}$','Interpreter','latex');
title ('Cross-sectional View of Spar')
```


Calculate Mass of Spar

Input - X - Design Variable Output - Spar Mass, gradient of objective

```
function [Spar_mass,g] = obj_func(X)
Nnodes = (length(X)/2); % No. of nodes
L = 7.5; %m Semi-Length of spar
rho = 1600; % kg/m^3 - density

% Calculate Volume
Volume = Calc_vol(X,L,Nnodes-1);

% Compute Mass
Spar_mass = rho*Volume;

% Compute gradient
g = Calc_objGrad(X);
end
```

Calculate Volume of beam

Input - Radius profile, Lengthm # elements; Output - Volume

```
function [Volume] = Calc_vol(R,L,Nelem)
% R - vector containing the radius [Rout1 Rin1 Rout2 Rin2 ... RoutN RinN]'
Volume = 0;
k = 1;
for i=1:Nelem
    r = R(k:k+3);
    % Average CS area between 2 adjacent nodes
    avgCS = 0.5*(pi*(r(1)^2-r(2)^2)+pi*(r(3)^2-r(4)^2));
    dV = avgCS*(L/Nelem);
    k = k+2;
    Volume = Volume + dV;
end
end
```

Calculate Gradient of the Objective

Input - X - Design Variable Output - Gradient of the objective

```
function g = Calc_objGrad(X)
h = 10^-60; % complex step size

e = zeros(length(X),1);
Nnodes = (length(X)/2); % No. of elements
L = 7.5; %m - Semi-Length of spar
rho = 1600; % kg/m^3 - density of material

for j=1:length(X)
    e(j) = 1;
    x_cmplx = X + (h*e)*i;
    vol_cmplx = Calc_vol(x_cmplx,L,Nnodes-1);
    f_cmplx = vol_cmplx*rho;
    g(j,1) = imag(f_cmplx)/h;
    e(j) = 0;
end
end
```

Calculate the Nonlinear Inequality constraint

Input - X - Design Variable; Output - Nonlinear inequality {constraint, gradient} - Nonlinear equality {constraint, gradient}

```
function [c,ceq,J,Jeq] = NonLnCons(X)

Mass = 500; % total operational mass of aircraft
Nnodes = length(X)/2; % Number of nodes
L = 7.5; %m -  Semi Length of spar
x = (0:L/(Nnodes-1):L)'; % discretize the length
E = 70e9; % 70 GPa Young's modulus
Max_Tensile_Strength = 600e6; % Tensile Strength

% Calculate Iyy
Iyy = Calc_Iyy(X,Nnodes);

% Calculate force on wing
force = Calc_force(x,Mass,L);

% Calculate vertical displacement and angular displacement
[u] = CalcBeamDisplacement(L, E, Iyy, force, Nnodes-1);

% Compute normal stresses on the beam elements
zmax = X(1:2:end);
[sigma] = CalcBeamStress(L, E, zmax, u, Nnodes-1);

% Compute Nonlinear constraint and its gradient
c = sigma/Max_Tensile_Strength-1;
J = Calc_consJac(X,Nnodes,L,E,force,Max_Tensile_Strength);

ceq = [];
Jeq = [];
end
```

Calculate Iyy

```
function Iyy = Calc_Iyy(R,Nnodes)
% R - radii ordered as [Rout1 Rin1 Rout2 Rin2.... RoutN RinN]'
Iyy = zeros(Nnodes,1);
k=1;
for i=1:Nnodes
     Iyy(i) = pi*(R(k)^4 - R(k+1)^4)/4;
     if(Iyy(i)<=1e-12)
        Iyy(i) = 1e-12;
     end
    k = k+2;
end
end
```

Calculate force distribution along length of spar

Input - x (length discretization), Mass, Length of spar; Output - force at each node

```
function force = Calc_force(x,Mass,L)
% force is linear in nature - cX
% integral(cx dx)=2.5*Weight/2 - c = 2.5*Weight/L^2
g = 9.81; % m/s^2
c = 2.5*Mass*g/L^2;
force = c*x;
force = flip(force);
end
```

Compute Jacobian of Nonlinear Inequality constraint

Inputs: X - Deisgn Variable; Nnodes - # of Nodes; L - Semi Length of spar; E - Young's Modulus; force - Force on each node; Max_Tensile_Strenth - Yield tensile/compressive stress; Output - Jacobian

```
function gc = Calc_consJac(X,Nnodes,L,E,force,Max_Tensile_Strength)

h = 10^-60; % complex step size
gc = zeros(length(X),Nnodes);

for j=1:length(X)
    x_cmplx = X;
    x_cmplx(j) = x_cmplx(j) + complex(0,h);
    Iyy_cmplx = Calc_Iyy(x_cmplx,Nnodes);
    [u_cmplx] = CalcBeamDisplacement(L, E, Iyy_cmplx, force, Nnodes-1);
    zmax_cmplx = x_cmplx(1:2:end);
    [sigma_cmplx] = CalcBeamStress(L, E, zmax_cmplx, u_cmplx, Nnodes-1);
    sigma_cmplx = sigma_cmplx./Max_Tensile_Strength;
    gc(j,:) = imag(sigma_cmplx)./h;
end

end
```

Published with MATLAB® R2021a