MECH 326 Assignment 1

Prepared by:

Akshiv Bansal
35072131

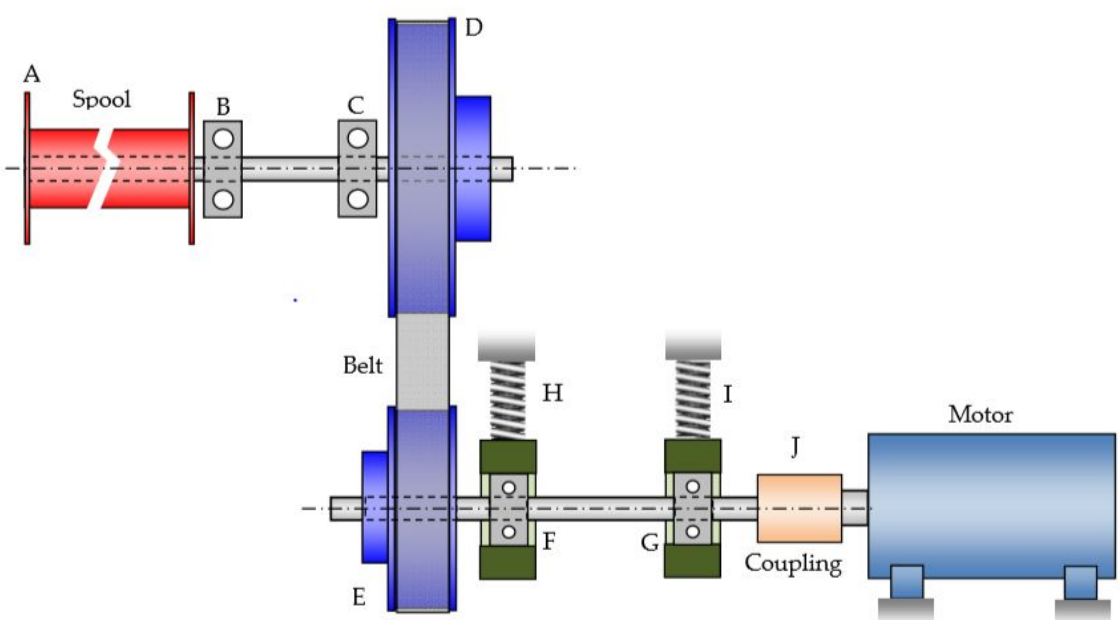
Wilhelm Gavino
31789134

Candice Ip
34435131

Doug Matthews
13843123

Stefan Sander-Green
15081136

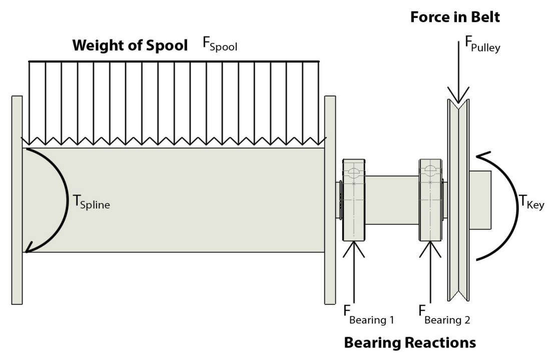
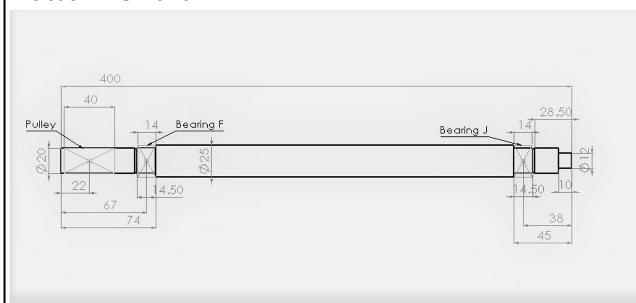
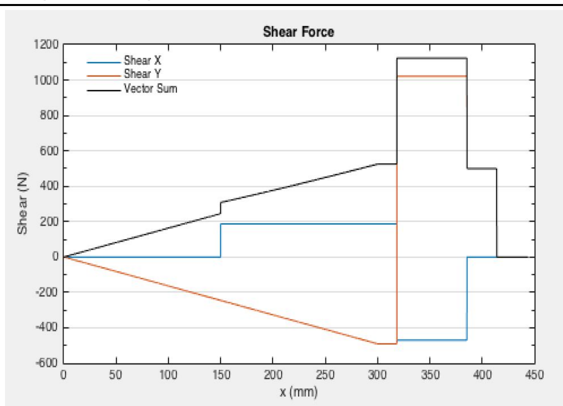
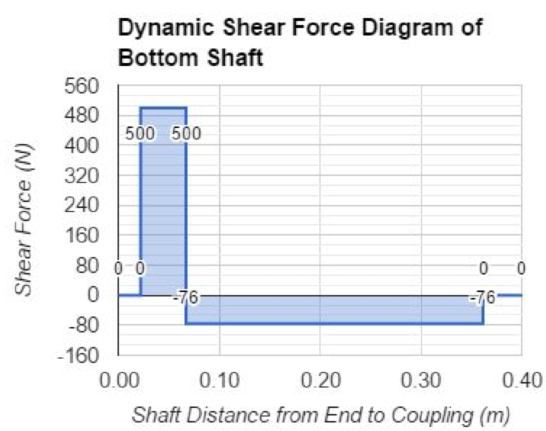
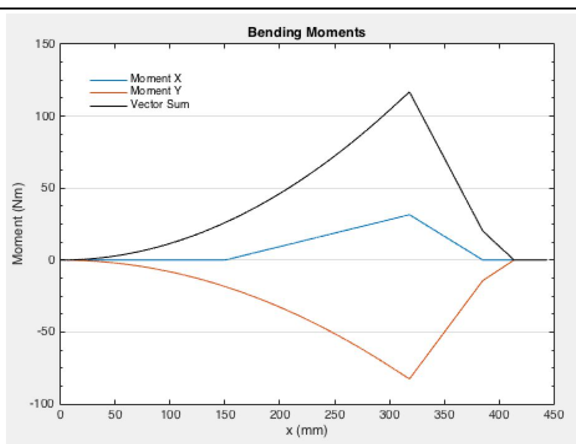
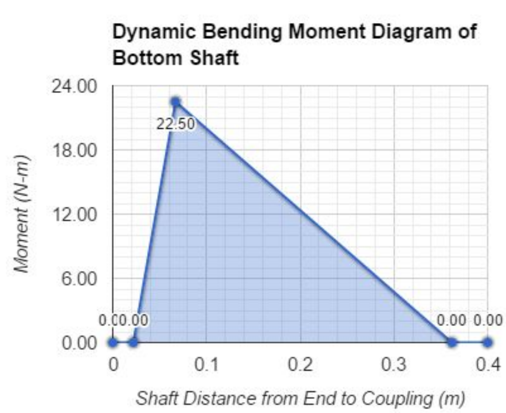
Gabriella Xiong
47625132

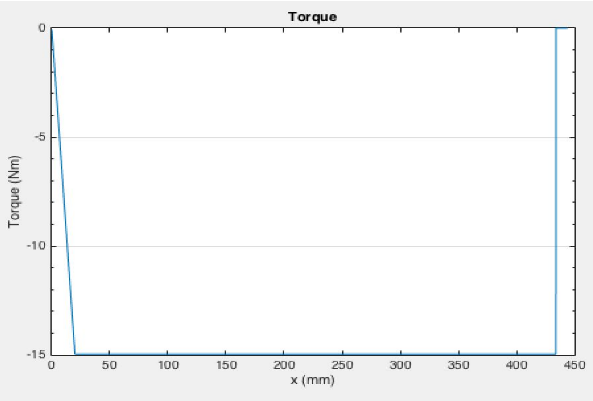


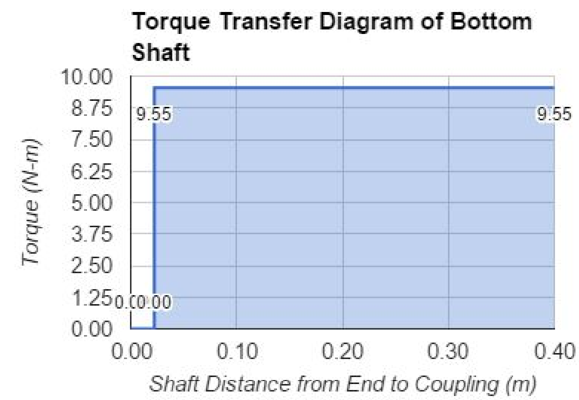
Summary of Approach

Our teams approach to this problem was to analyze each shaft individually For the top shaft we chose a geometry that would maximize our performance metric without causing unnecessarily large reactions and stresses. We then checked to make sure the shaft would satisfy the requirements for deflection, slope, infinite life, and speed. For the bottom shaft we began by symbolically determining the forces and moments along each shaft so that it would be relatively easy to change characteristics such as component spacing and shaft diameter. We used this to choose a final design which satisfies a minimum factor of safety of two.

Details of Final Design

Top Shaft:	**Bottom Shaft:**
 *Please see "**Appendix B - Top Shaft**" for enlarged image of shaft.*	 *Please see **"Appendix A - Bottom Shaft I.Detailed Bottom Shaft Diagram"** for enlarged image of shaft.*
	
	





Assumptions:

- Both springs exert the same force on the shaft when it's operating
- Assumed that weight of the shaft was insignificant compared to the loads on it
- Assumed $K_t = K_f$ and $K_{ts} = K_{fs}$ to be conservative when we did not know the notch sensitivity "q".
- Assumed transverse shear stress was small in comparison to bending stress (verified after analysis)
- Assumed deflection and slope since we are designing for the deflection at there to be zero.

Critical Locations of Design:

See appendix for all critical stress, deflection and slope results.

Maximum Stress Calculations

Critical Location	σ (MPa)	τ (MPa)	Distortion Energy	σ_m (MPa)	σ_a (MPa)	Goodman
Retaining Ring 1	26.3	1.77	24.9	8.37	113	2.00

Maximum Linear Deflection

Location	Deflection (mm)	Allowable (mm)
Spool A	0.246	1

Maximum Angular Slope:

Location	Slope (rad)	Allowable (rad)
Spool A	9.96E-04	0.008

Critical Speeds:

Shaft	Critical Speed (rpm)	Actual Speed (rpm)
Top shaft	3490	600
Bottom shaft	4520	1000

Performance:

Mass

Item	**Mass (kg)**
Mass of Raw Material for Top Shaft	5.88
Mass of Raw Material for Bottom Shaft	3.85
Mass Removed from Machining Top Shaft	-2.41
Mass Removed from Machining Bottom Shaft	-0.181
Total Mass	**7.139**

Cost

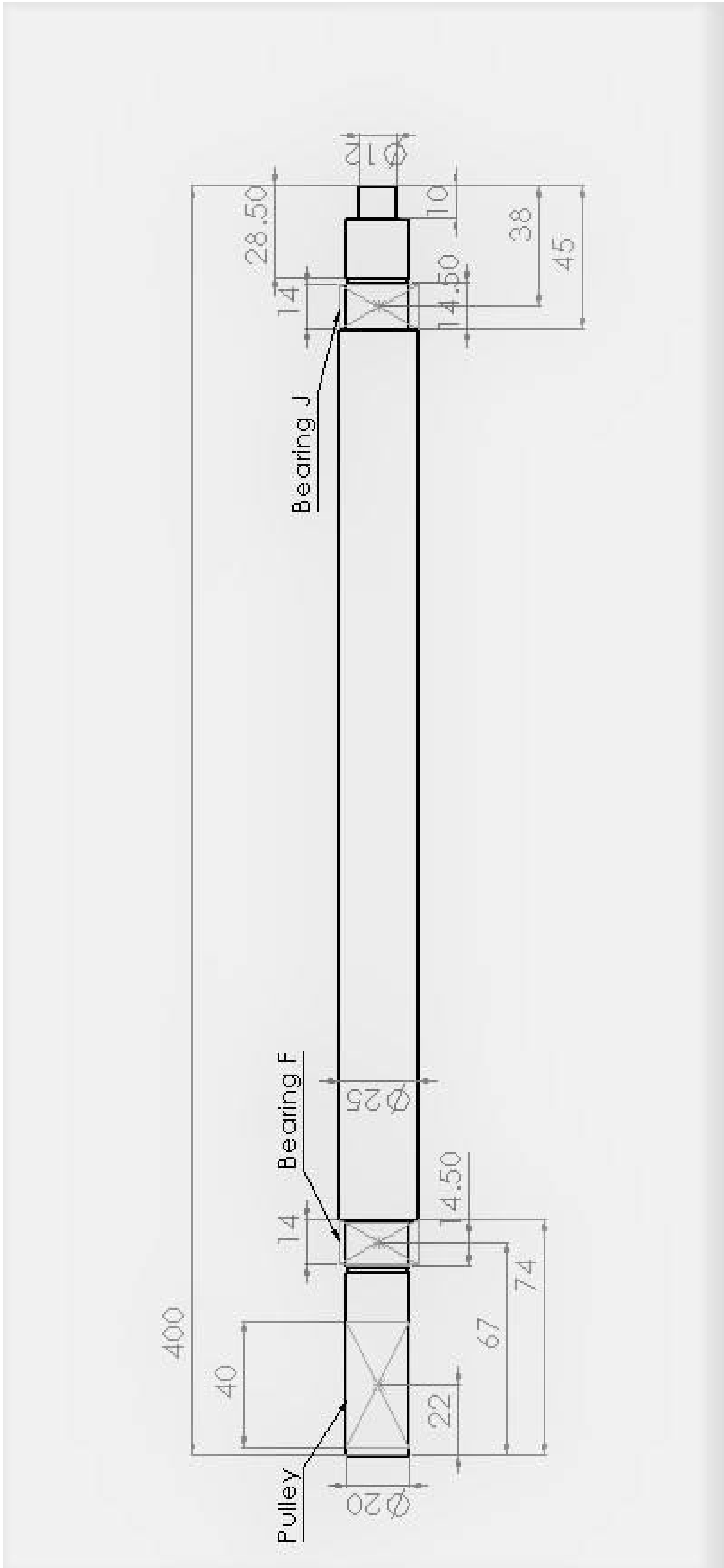
Item	Cost
Raw Material of Top Saft	$76.44
Raw Material of Bottom Shaft	$50.09
Machining Costs Top Shaft	$120.50
Machining Costs Bottom Shaft	$9.04
Shaft Additions	$240.00
Total Cost	**$496.07**

Performance Metric (cost · mass)$^{-1}$ [$ · kg]$^{-1}$	**0.000282**

Appendix A: Bottom Shaft Analysis

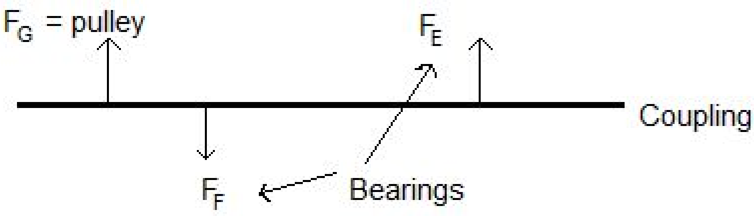
Contents:

I. Detailed Bottom Shaft Diagram



II. Force and Moment Calculations of Bottom Shaft

Free Body Diagram



Force Equations:
Equilibrium:
$\sum \mathbf{F}_y = 0 = \mathbf{F}_E + \mathbf{F}_F + \mathbf{F}_G$

Moment Equations: $\sum \mathbf{M}_J = 0 = -F_E * d_{EJ} + F_F * d_{FJ} + F_G * d_{GJ}$

Constitutive Relation: $\mathbf{F}_G = -k * \Delta y_{G0}$, $\mathbf{F}_F = -k * \Delta y_{F0}$
Using equilibrium calculations, we were able to obtain the initial compression and extension in the springs.

Outcome:
compression requirement for equilibrium
$\Delta y_{F0} = 15.245 \ mm$ (compression)
$\Delta y_{g0} = 2.018 \ mm$ (extension)

<u>III. Static Loading of Bottom Shaft Calculations</u>

A. Force Required to Change Belt

Force Equations:

$$\sum \mathbf{F}_y = 0 = \mathbf{F}_E + \mathbf{F}_F + \mathbf{F}_G + \mathbf{F}_J$$

$$\mathbf{F}_F = -k * (\Delta y_H + \Delta y_{F0})$$

$$\mathbf{F}_G = -k * (\Delta y_I + \Delta y_{G0})$$

Moment Equations:

$$\sum \mathbf{M}_J = 0 = -F_E * d_{EJ} + F_F * d_{FJ} + F_G * d_{GJ}$$

Constitutive Relation:

$$\frac{\delta_{10mm}}{d_{EJ}} = \frac{\Delta y_H + \Delta y_{F0}}{d_{FJ}} = \frac{\Delta y_I + \Delta y_{G0}}{d_{GJ}}$$

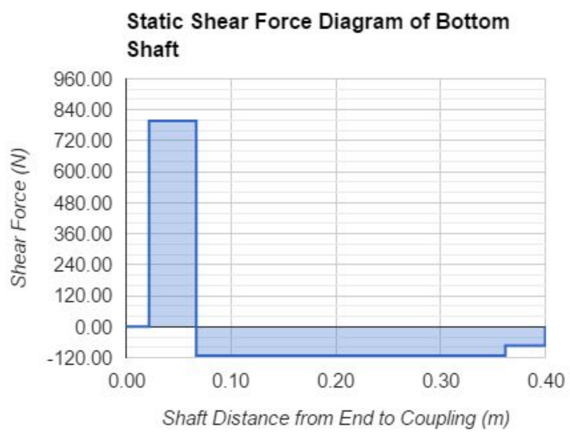
Outcome:

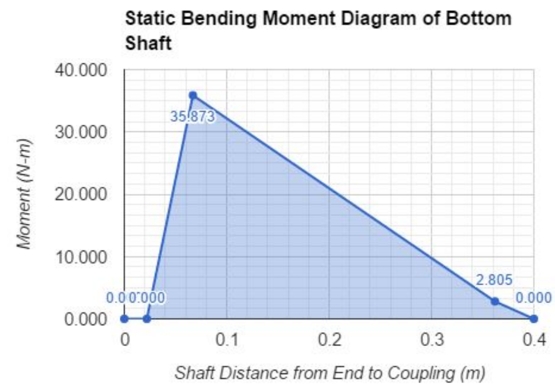
$$\Delta y_H = \frac{\delta_{10mm}}{d_{EJ}} * d_{FJ} - \Delta y_{F0}$$

$$\Delta y_I = \frac{\delta_{10mm}}{d_{EJ}} * d_{GJ} - \Delta y_{G0}$$

$$F_E = \frac{F_F * d_{FJ} + F_G * d_{GJ}}{d_{EJ}}$$ (force upwards needed to change belt)

Shear and Bending Diagrams for Bottom Shaft While Changing Belt:





Forces:	(N)
FE (pulley)	797.18
FF (bearing at F)	-909.27
FG (bearing at G)	38.27
FJ (coupling)	73.82

Moments:	(N-m)
E	0.000
F	35.873
G	2.805
J	0.000

Results:

Force Required to Change Belt (N)	Allowable Force to Change Belt (N)
797.18	800

B. Failure Due To Yielding During Belt Change

In order to calculate the safety factor of the shaft due to yielding when changing the belt we used the equation:

- Bending Stress:

$$\sigma = \frac{Mc}{I}$$

Results:

Location	Maximum Bending Stress	Safety Factor
At Bearing F	45.6 MPa	12.70

IV. Dynamic Loading of Shafts

A. Shaft Dynamic Loading Example Calculations

In order to calculate fatigue failure int the bottom shaft we first determined the midpoint and alternating stresses in the shaft. We combined normal and shear stresses using Von Mises equations. It was also important to take into consideration stress concentrations at points where the shaft diameter changed. (Because the shaft material is ductile these stress concentrations are only relevant when considering fatigue failure.) Finally, we determined the fatigue safety factor using the Modified Goodman technique as it is more conservative than the other fatigue failure methods. A similar method was used to calculate the fatigue failure of the bottom shaft.

In order to calculate the safety factor of the bottom shaft we used the following formulas:

- Torque

$$T = \frac{Power * 60}{rpm * 2 * \pi}$$

- Fatigue Equations

$$\sigma_a = K_f \frac{32M_a}{\pi d^3} \qquad \sigma_m = K_f \frac{32M_m}{\pi d^3}$$

$$\tau_a = K_{fs} \frac{16T_a}{\pi d^3} \qquad \tau_m = K_{fs} \frac{16T_m}{\pi d^3}$$

- Von Mises

$$\sigma'_a = (\sigma_a^2 + 3\tau_a^2)^{\frac{1}{2}} \qquad \sigma'_m = (\sigma_m^2 + 3\tau_m^2)^{\frac{1}{2}}$$

- Modified Goodman

$$\frac{1}{n} = \frac{\sigma'_a}{S_e} + \frac{\sigma'_m}{S_{ut}}$$

- Shear Yielding Strength from Distortion Energy Theorem

$$S_{sy} = \frac{S_y}{\sqrt{3}}$$

Values Obtained

Midrange Torque (Nm)	9.55
Alternating Torque (Nm)	0
Midrange Moment (Nm)	0
Alternating Moment (Nm)	22.50

Shaft Properties

	MPa	Pa
Ultimate Strength Sut	690	690000000
Yield Strength	580	580000000
Endurance Limit Se'	345	345000000
Corrected Se	518.7	518695050.1
Shear Strength	334.9	334863156.1

Marin Factors

ka	0.7978
kb	1.884566538
kc	1
kd	1
ke	1
kf	1

1. Safety Factor of Bottom Shaft at Critical Locations:

Notch at F

Midrange Normal Stress (Pa)	0
Alternating Normal Stress (Pa)	45836623.61
Midrange Shear Stress (Pa)	8389394.006
Alternating Shear Stress (Pa)	0
Von Mises Alt Normal Stress	159619301
Von Mises Mid Normal Stress	20052582.19
nf	**2.9692**
ny (first cycle yielding)	**5.07**

Stress at Gib Key

Est Moment at end of key (N-m)	21
Kts	3
Kt	2.14
q	1
VM Alt Stress	57219385.14
VM Midrange Stress	10529606.28
1/nf	0.1255744134
nf	**7.963405702**

Retaining Ring near F

Kt	4
Kts	2.5
q	0.8
Kf	3.4
Kfs	2.2
Von mises for alternating	38197186.34
Von mises for shearing	10529610.04
n from alternating	**2.178757678**
localized yielding n	**11.90310144**

Direct Shear From E-F

tmax (N/m)	2122065.908
n	**157.8005447**

Direct Shear From J'-J

tmax (N/m)	-6117066.296
n	**54.7424435**

2. Deflections and Slope of Bottom Shaft During Dynamic Loading

We made use of the following equations to calculate slope and deflection:

- Curvature Equation:
$$\frac{M}{EI} = \frac{\partial^2 y}{\partial x^2}$$
- Singularity Function:
$$\frac{1}{E}\frac{\partial^2 y}{\partial x^2} = a1\left\langle x-p1\right\rangle^1 + a2\left\langle x-p2\right\rangle^1 + a3\left\langle x-p3\right\rangle^1 + a4\left\langle x-p4\right\rangle^0 + a5\left\langle x-p5\right\rangle^0 + a6\left\langle x-p6\right\rangle^1$$
- Singularity Function for Slope, integrated from the last equation
$$\frac{1}{E}\frac{\partial y}{\partial x} = \frac{a1}{2}\left\langle x-p1\right\rangle^2 + \frac{a2}{2}\left\langle x-p2\right\rangle^2 + \frac{a3}{2}\left\langle x-p3\right\rangle^2 + a4\left\langle x-p4\right\rangle^1 + a5\left\langle x-p5\right\rangle^1 + \frac{a6}{2}\left\langle x-p6\right\rangle^2 + C1$$
- Singularity Function for Deflection is calculated by integrate twice the function for moment
$$\frac{1}{E}y = \frac{a1}{6}\left\langle x-p1\right\rangle^3 + \frac{a2}{6}\left\langle x-p2\right\rangle^3 + \frac{a3}{6}\left\langle x-p3\right\rangle^3 + \frac{a4}{2}\left\langle x-p4\right\rangle^2 + \frac{a5}{2}\left\langle x-p5\right\rangle^2 + \frac{a6}{6}\left\langle x-p6\right\rangle^3 + C1x + C2$$

For this case where the button shaft is operating the force from the pulley is 500N and we assume both force and moment at the coupling is zero :

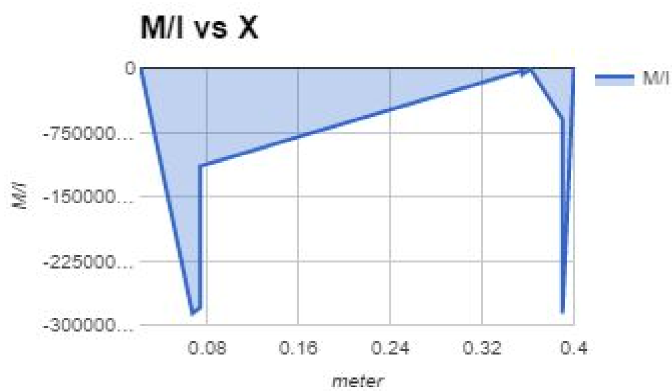
Table of forces :

location	meter	Force	Moment
Pulley E	0.022	500	-189
Bearing F	0.067	-576	+191.808
Bearing G	0.362	76	-2.808
Coupling J	0.4	0	0

With the assumption that the deflection and the slope at bearing F are both zero, we get singularity function from information below:

Location	meter	moment	M/I	step	slope	delta slope
E	0.022	0	0		-63661977237	-63661977237
F	0.067	-22.5	-2864788976		-63661977237	7333859777
F'+	0.074	-21.968	-2797052632		9676620540	
F'-	0.074	-21.968	-1145672758	1651379874	3963543773	-5713076767
G'+	0.355	-0.612	-31916957.75		3963543773	
G'-	0.355	-0.612	-77922260.14	-46005302.39	9676620540	5713076767
G	0.362	-0.08	-10185916.36		9676620540	
J'+	0.39	-0.612	-601252007.2	-523329747.1	-2419155135	-12095775675
J'-	0.39	-0.612	-2864788976		60125200724	
J	0.4	0	0		60125200724	

From the table above we are able to plot M/I in relation to X :



Results:

Location	Deflection (mm)	Allowable (mm)	Slope (rad)	Allowable (rad)
Pulley E	4.6E-06	1	3.07E-4	0.008
Bearing F	0	0	0	0.001
Bearing G	-2.69E-04	0	-8.84E-04	0.001
Coupling J	0	1	0	0.008

B. Critical Speed Calculations

The last step in our failure analysis was to ensure that the shaft is rotating below critical speed. To obtain this we divided the shaft into different sections based on their diameters, taking the weight of each section as a point force in the middle of the section. Adding the weight of the pulley at its location, we derive a deflection function using the resulting shear force diagram. Weights of the ball bearings were negligible and were not included in the shear force diagram. Using the deflection caused by the different weights, we use Rayleigh's Equation to determine the critical speed.

Comparing this result, which takes into account the deflections caused by the shaft's and the pulley's weights, with the result that uses the vector sum of the deflection caused by the shaft's weights only and the deflection caused by the pulley's weight only, we find that the first result yielded a smaller value for the critical speed.

In order to calculate critical speeds we used the following equation:

- Rayleigh's Equation

$$\omega = \sqrt{\frac{g \sum w_i y_i}{\sum w_i y_i}}$$

To combine the critical speed we calculated from the weight of the pulley alone and the weight of the shaft alone:

$$\frac{1}{\omega_{system}^2} = \frac{1}{\omega_{components}^2} + \frac{1}{\omega_{shaft}^2}$$

1. Bottom Shaft Critical Speed

Shaft	Critical Speed (rpm)	Actual Speed (rpm)
Bottom Shaft	4520	1000

Appendix B: Top Shaft Analysis (ABCD)

Contents

Given values

The setup using values given by the problem description.

```
%Geometry

r_shaft      = 0.035/2; %m
r_shoulder   = r_shaft+ 0.006; %m
r_hub        = 0.034/2; %m
r_spool      = 0.08; %m

%Simulation Parameter
dx = 1e-4;

%Material properties for ANSI 10150 CD
S_ut         = 690*10^6;  %Pa
S_y          = 580*10^6; %Pa
density      = 7860; %kg/m^3
E            = 190e9; %Pa
S_e_prime    = 0.5*S_ut;

%Known forces
F_vbelt_y    = -500; %N
F_spool_y    = -490.5; %N
F_pulley     = -20; %N
g            = -9.81; %kg*m/s

%Torque calculations
%Using the power of the motor, speeds of the shafts and efficency we can
%find the torque and froces on the shaft by the accessories
H_max                    = 1000; %W
```

```
rotation_speed_motor    = 1000; %rpm
T_motor                 = H_max/rotation_speed_motor*60/(2*pi); %Nm
rotation_speed_upshaft  = 600; %rpm
eff_vbelt               = 0.94;
T_belt                  = eff_vbelt* rotation_speed_motor /...
    rotation_speed_upshaft * T_motor;%Nm
T_spool                 = -T_belt;
F_spool_x               = -T_spool/ r_spool;

%Lengths
%The lenghts that can be changed are the space between the spoool and the
%first bearing, the space between the bearing and the space between the
%second bearing and the pulley
length_space1   = 10*10^-3; %m
length_shoulder = 50*10^-3; %m
length_space2   = 10*10^-3; %m

%Element Widths
bearingWidth    = 0.017;    %m
spoolWidth      = 0.3;      %m
splineWidth     = 0.02;     %m
pulleyWidth     = 0.02;     %m
hubWidth        = 0.02;     %m

%From the widdths of elemends and the lengths we set we can determine the
%location of all elements along the shaft
spool_position_start    = 0;            %m
spline_position_end     = splineWidth;  %m
spool_position_end      = spoolWidth;   %m
spool_position          = (spool_position_start+spool_position_end)/2;      %m
bearing1_start          = spool_position_end+length_space1;                 %m
bearing1_position       = spool_position_end+length_space1+bearingWidth/2;  %m
shoulderStart           = bearing1_position+bearingWidth/2;                 %m
shoulderEnd             = shoulderStart+length_shoulder;                    %m
bearing2_position       = shoulderEnd+bearingWidth/2;                       %m
bearing2_end            = shoulderEnd+bearingWidth;                         %m
vbelt_position_start    = shoulderEnd+bearingWidth+length_space2;           %m
vbelt_position          = vbelt_position_start+pulleyWidth/2;               %m
vbelt_hub_position      = vbelt_position_start+pulleyWidth+hubWidth/2;      %m
end_of_rod              = vbelt_position_start+pulleyWidth+hubWidth;        %m

%the geometric and physics properties of the sections of the shaft with
%different diameters are found
%region1
r1          = r_shaft;              %m
xsection1   = pi*r1^2;              %m^2
I1          = pi/4*r1^4;            %m^4
length1     = shoulderStart;        %m
volume1     = xsection1*length1;    %m^3
mass1       = density*volume1;      %kg

%region2
r2          = r_shoulder;           %m
xsection2   = pi*r2^2;              %m^2
I2          = pi/4*r2^4;            %m^4
length2     = length_shoulder;      %m
volume2     = xsection2*length2;    %m^3
```

```
mass2        = density*volume2;      %kg

%region3
r3           = r_shaft;                      %m
xsection3    = pi*r3^2;                      %m^2
I3           = pi/4*r3^4;                    %m^4
length3      = bearingWidth + length_space2; %m
volume3      = xsection3*length3;            %m^3
mass3        = density*volume3;              %kg

%region4
r4           = r_hub;                    %m
xsection4    = pi*r4^2;                  %m^2
I4           = pi/4*r4^4;                %m^4
length4      = pulleyWidth + hubWidth;   %m
volume4      = xsection4*length4;        %m^3
mass4        = density*volume4;          %kg
```

Y Direction Statics

Using the condition that the sum of forces and the sum of moments must be zero we can find the reaction in the bearings in the Y direction by using a matrix.

```
calc_forces_y        = [ 1 1 -F_vbelt_y-F_spool_y ; bearing1_position...
    bearing2_position -F_spool_y*spool_position-F_vbelt_y*vbelt_position];
rref_calc_forces_y   = rref(calc_forces_y);
F_bearing1_y         = rref_calc_forces_y(1,3);
F_bearing2_y         = rref_calc_forces_y(2,3);
```

X Direction Statics

Using the condition that the sum of forces and the sum of moments must be zero we can find the reaction in the bearings in the X direction by usign a matrix.

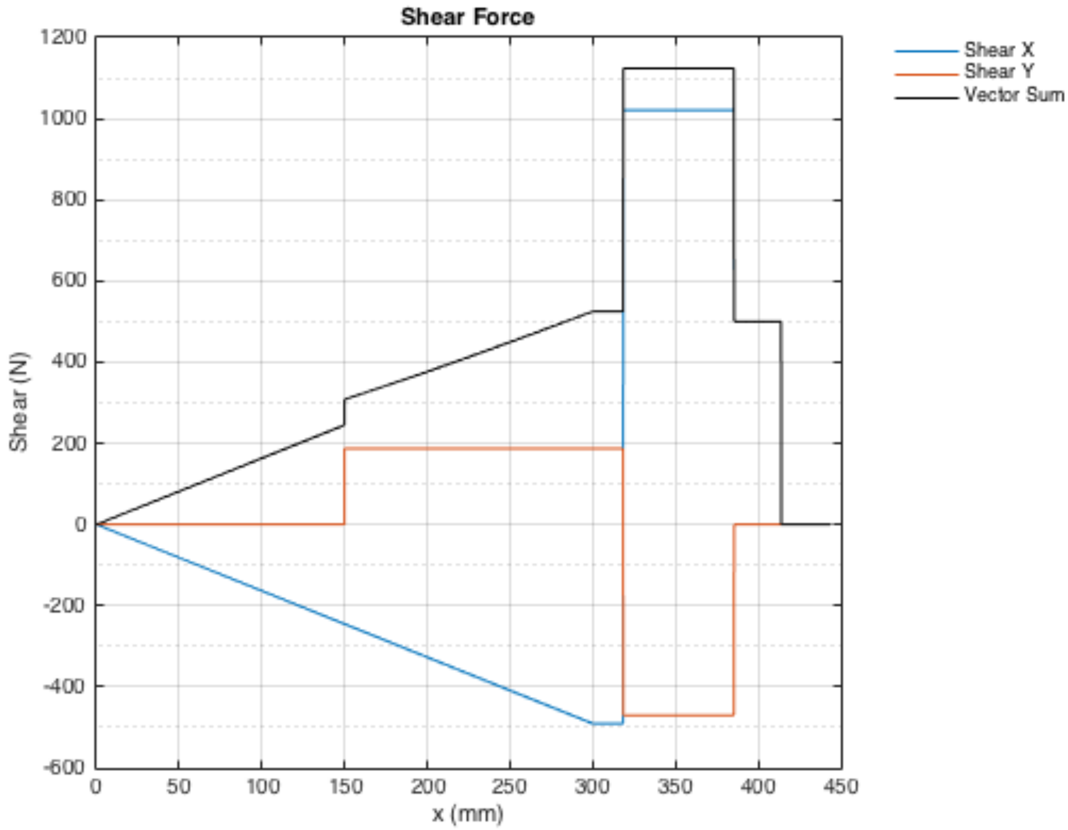
```
calc_forces_x        = [ 1 1 -F_spool_x ; bearing1_position...
    bearing2_position -F_spool_x*spool_position];
rref_calc_forces_x   = rref(calc_forces_x);
F_bearing1_x         = rref_calc_forces_x(1,3);
F_bearing2_x         = rref_calc_forces_x(2,3);

Distrub_spool_weight = F_spool_y/spoolWidth ;
```

Generating Shear Diagrams

A vector is used to represent the shear along the shaft. This vector can be plotted to obtain the shear diagram. We have seperate vectors for the shear int he X and Y direction as well as vector sum of the shear.

```
shearY = 0:dx:end_of_rod;
for i = 1:length(shearY)
    x   = i*dx;
    if (x < spool_position_end)
        shearY(i) = Distrub_spool_weight * x;
```

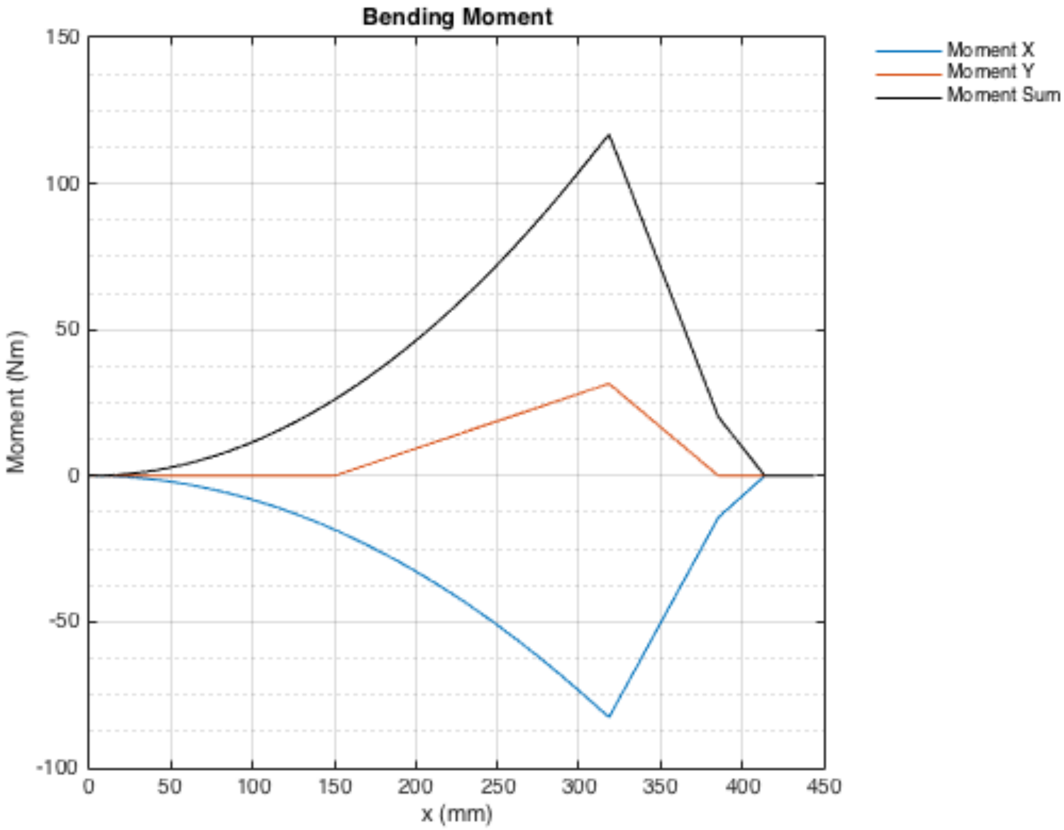
```
    elseif(x < bearing1_position)
        shearY(i) = F_spool_y;
    elseif(x < bearing2_position)
        shearY(i) = F_spool_y + F_bearing1_y;
    elseif(x < vbelt_position)
        shearY(i) = F_spool_y + F_bearing1_y + F_bearing2_y;
    else
        shearY(i) = F_spool_y + F_bearing1_y + F_bearing2_y + F_vbelt_y;
    end
end

shearX = 0:dx:end_of_rod;
for i = 1:length(shearX);
   x = i*dx;
   if(x < spool_position)
       shearX(i) = 0;
   elseif(x < bearing1_position)
       shearX(i) = F_spool_x;
   elseif(x < bearing2_position)
       shearX(i) = F_spool_x+F_bearing1_x;
   else
       shearX(i) = F_spool_x+F_bearing1_x+F_bearing2_x;
   end
end


shearVec = 0:dx:end_of_rod;
for i = 1:length(shearVec);
    shearVec(i) = sqrt( shearY(i)^2 + shearX(i)^2);
end
```



Generating Moment Diagrams

The moment diagrams are found by numerically integrating the shear digrams.

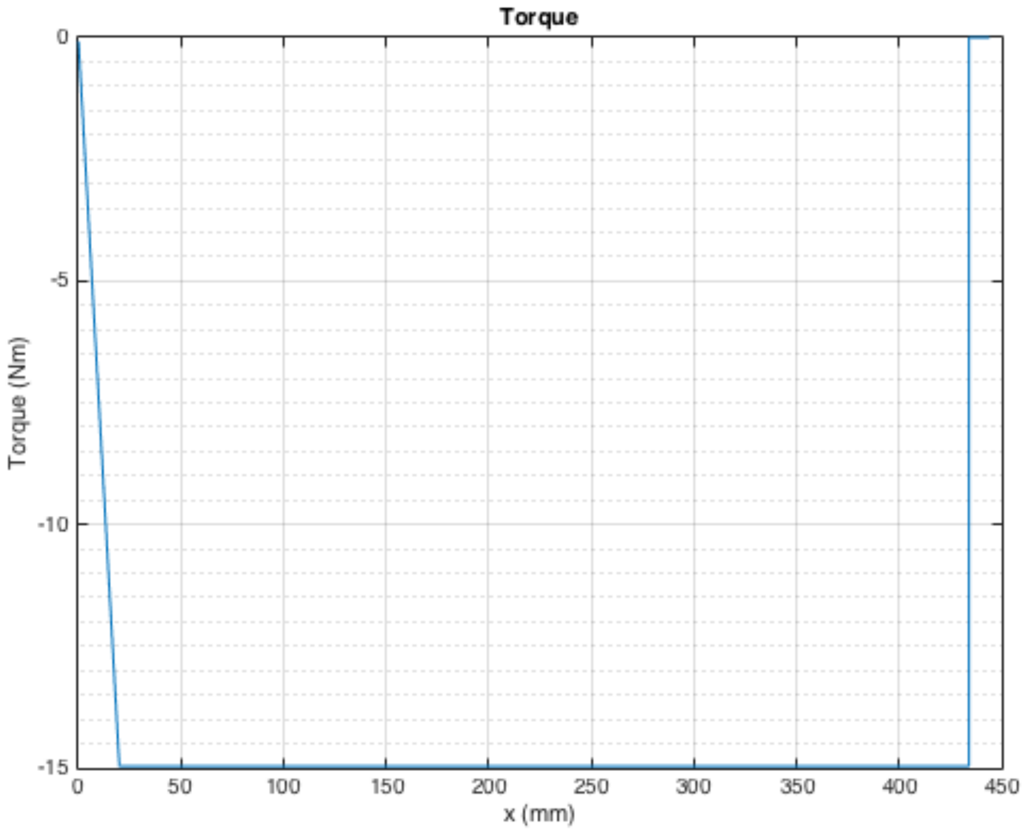
```
momentY = 0:dx:end_of_rod;
momentX = 0:dx:end_of_rod;
momentVec = 0:dx:end_of_rod;
momentY(1)   = 0;
momentX(1)   = 0;
momentVec(1) = 0;

for i = 2:length(momentY)
    momentY(i)   = momentY(i-1)+shearY(i)*dx;
    momentX(i)   = momentX(i-1)+shearX(i)*dx;
    momentVec(i) = sqrt( momentY(i)^2 + momentY(i)^2);
end
```



Generating Torque Diagrams

The torque (in the Z direction) along the shaft is also represented by a vector and plotted.

```
torsionZ = 0:dx:end_of_rod;
for i = 1:length(torsionZ)
    x = i*dx;
    if( x < spline_position_end)
        torsionZ(i)= T_spool/spline_position_end * x;
    elseif( x < vbelt_hub_position)
        torsionZ(i) = T_spool;
    else
        torsionZ(i) = T_spool + T_belt;
    end
end
```



Stress Calculations

The bending stress in the shaft are found by multiplying the elelment of the momement vector by the appropriate shaft radius and dividing by the appropriate moment of inertia. The transverse shear is any cross section is the maximum along the midplane and and is equal to:

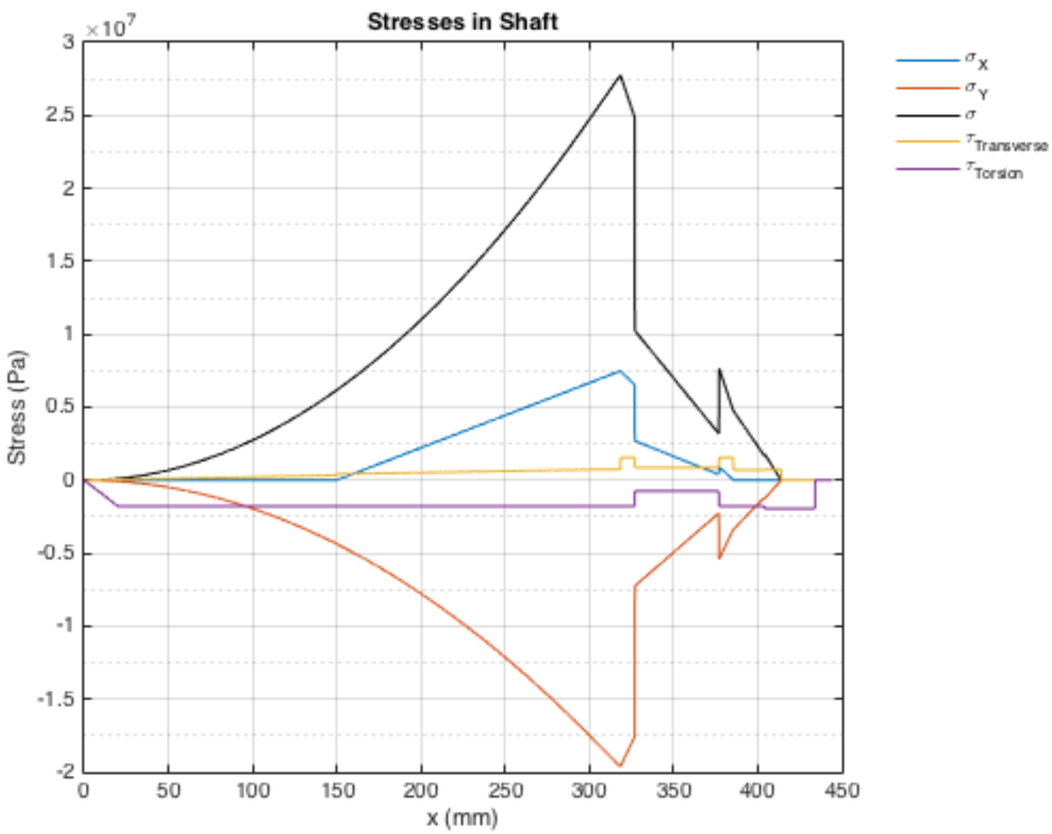
$$\tau = \frac{VQ}{It} = \frac{4V}{3A}$$

The torsional shear is maximum at the surface and is equal to:

$$\tau = \frac{rT}{J}$$

```
bendingStress             = 0:dx:end_of_rod;
bendingStressX            = 0:dx:end_of_rod;
bendingStressY            = 0:dx:end_of_rod;
transverseShearStress     = 0:dx:end_of_rod;
torsionalShearStress      = 0:dx:end_of_rod;

for i = 1:length(bendingStress)
    x = i*dx;
    if( x < shoulderStart)
        bendingStress(i)           = momentVec(i)*r1/I1;
        bendingStressX(i)          = momentX(i)*r1/I1;
        bendingStressY(i)          = momentY(i)*r1/I1;
        transverseShearStress(i)   = 4/3* shearVec(i) / xsection1;
        torsionalShearStress(i)    = torsionZ(i)*r1/(2*I1);
    elseif( x < shoulderEnd)
```

```
        bendingStress(i)          = momentVec(i)*r2/I2;
        bendingStressX(i)         = momentX(i)*r2/I2;
        bendingStressY(i)         = momentY(i)*r2/I2;
        transverseShearStress(i)  = 4/3* shearVec(i) / xsection2;
        torsionalShearStress(i)   = torsionZ(i)*r2/(2*I2);
    elseif( x < vbelt_position_start)
        bendingStress(i)          = momentVec(i)*r3/I3;
        bendingStressX(i)         = momentX(i)*r3/I3;
        bendingStressY(i)         = momentY(i)*r3/I3;
        transverseShearStress(i)  = 4/3* shearVec(i) / xsection3;
        torsionalShearStress(i)   = torsionZ(i)*r3/(2*I3);
    else
        bendingStress(i)          = momentVec(i)*r4/I4;
        bendingStressX(i)         = momentX(i)*r4/I4;
        bendingStressY(i)         = momentY(i)*r4/I4;
        transverseShearStress(i)  = 4/3* shearVec(i) / xsection4;
        torsionalShearStress(i)   = torsionZ(i)*r4/(2*I4);
    end
end

% We find that the transveres shear is insignificant conmpared to the
% bending stress and torsional sheer and is thus ignored when calculating
% stress
```



First Cycle Yield

The von Mises stress combinations is used to check for first cycle yield. We calculate the stress of each element in a vector and can then find the von Mises stress at each point along the rod. We can then find the factors of safety at each location using distrotion energy and maximum shear stress.

```
sigA     = 0:dx:end_of_rod;
sigB     = 0:dx:end_of_rod;
sig1     = 0:dx:end_of_rod;
sig2     = 0:dx:end_of_rod;
sig3     = 0:dx:end_of_rod;
sigPrime = 0:dx:end_of_rod;

%Initializing safety factors for the rod
nMSS     = 0:dx:end_of_rod;
nDE      = 0:dx:end_of_rod;

for i = 1:length(sigA)

sigA(i) = (bendingStressX(i) + bendingStressY(i))/2 +...
    sqrt( ( (bendingStressX(i)-bendingStressY(i) ) /2)^2 + (torsionalShearStress(i))^2 );

sigB(i) = (bendingStressX(i) + bendingStressY(i))/2 -...
    sqrt( ( (bendingStressX(i)-bendingStressY(i) ) /2)^2 + (torsionalShearStress(i))^2 );

sigPrime(i) = sqrt ( sigA(i)^2 - (sigA(i)*sigB(i)) + sigB(i)^2 );

    if (sigA(i) > 0 && sigB(i) > 0)
       sig1(i) = sigA(i);
       sig3(i) = 0;
    elseif (sigA(i) > 0 && sigB(i) < 0 )
       sig1(i) = sigA(i);
       sig3(i) = sigB(i);

    elseif (sigA(i) < 0 && sigB(i) < 0)
       sig1(i) = 0;
       sig3(i) = sigB(i);
    end
    nMSS(i) = S_y / (sig1(i) -sig3(i) );
    nDE(i)  = S_y / sigPrime(i);
end
```

Fatigue Analysis

We looked at each location where a stress concentration exists and found factor of safety of infinite life using Goodmann as well as checking for first cycle yield with the distortion energy theorem.

```
%Spline
location     = spline_position_end;
S_e          = 233.295*10^6; %Pa
q            = 1;
q_s          = 1;
k_t          = 1; %Bending is insignificant
k_ts         = 5; %Worst case senario cit. Peterson Stress Concentration 2008
k_f          =1+q*(k_t-1);
k_fs         =1+q_s*(k_ts-1);
midrange     = sqrt ( 3*(k_fs*torsionalShearStress(location/dx))^2 );
alternating  = abs(k_f*bendingStress(location/dx));
goodmann_spl = 1/(alternating/S_e+midrange/S_ut);
DE_spl       = nDE(location/dx);
sig_spl      = bendingStress(location/dx);
```

```
tau_spl      = torsionalShearStress(location/dx);

%Retaining Ring 1
location     = bearing1_start;
S_e          = 233.295*10^6; %Pa
q            = 0.83; %This is based on worst case senario of notch radius equal to half  groov
e width
q_s          = 0.86;
k_t          = 5; %First Iteration Approximations
k_ts         = 3;
k_f          = 1+q*(k_t-1);
k_fs         = 1+q_s*(k_ts-1);
midrange     = sqrt ( 3*(k_fs*torsionalShearStress(location/dx))^2 );
alternating  = abs(k_f*bendingStress(location/dx));
goodmann_rr1 = 1/(alternating/S_e+midrange/S_ut);
DE_rr1       = nDE(location/dx);
sig_rr1      = bendingStress(location/dx);
tau_rr1      = torsionalShearStress(location/dx);

%shoulder left
location    = shoulderStart;
S_e         = 233.295*10^6; %Pa
q           = 0.87; %Based on largest possible notch radius equal to bearing fillet radius (35
mm 02 series deep groove, table 11-2)
q_s         = 0.9;
k_t         = 2.4;
k_ts        = 2.1;
k_f         =1+q*(k_t-1);
k_fs        =1+q_s*(k_ts-1);
midrange    = sqrt ( 3*(k_fs*torsionalShearStress(location/dx))^2 );
alternating = abs(k_f*bendingStress(location/dx));
goodmann_sl = 1/(alternating/S_e+midrange/S_ut);
DE_sl       = nDE(location/dx);
sig_sl      = bendingStress(location/dx);
tau_sl      = torsionalShearStress(location/dx);

%shoulder right
location    = shoulderEnd;
S_e         = 233.295*10^6; %Pa
q           = 0.87; %Based on largest possible notch radius equal to bearing fillet radius (35
mm 02 series deep groove, table 11-2)
q_s         = 0.9;
k_t         = 2.4;
k_ts        = 2.1;
k_f         = 1+q*(k_t-1);
k_fs        = 1+q_s*(k_ts-1);
midrange    = sqrt ( 3*(k_fs*torsionalShearStress(location/dx))^2 );
alternating = abs(k_f*bendingStress(location/dx));
goodmann_sr = 1/(alternating/S_e+midrange/S_ut);
DE_sr       = nDE(location/dx);
sig_sr      = bendingStress(location/dx);
tau_sr      = torsionalShearStress(location/dx);

%retaining ring 2
location     = bearing2_end;
S_e          = 233.295*10^6; %Pa
q            = 0.83;
```

```
q_s          = 0.86;
k_t          = 5; %First Iteration Approximations
k_ts         = 3;
k_f          = 1+q*(k_t-1);
k_fs         = 1+q_s*(k_ts-1);
midrange     = sqrt ( 3*(k_fs*torsionalShearStress(location/dx))^2 );
alternating  = abs(k_f*bendingStress(location/dx));
goodmann_rr2 = 1/(alternating/S_e+midrange/S_ut);
DE_rr2       = nDE(location/dx);
sig_rr2      = bendingStress(location/dx);
tau_rr2      = torsionalShearStress(location/dx);

%gibs key
location    = vbelt_hub_position;
S_e         = 234.02*10^6; %Pa
q           = 1;
q_s         = 1;
k_t         = 2.14; %First Iteration Approximations
k_ts        = 3;
k_f         =1+q*(k_t-1);
k_fs        =1+q_s*(k_ts-1);
midrange    = sqrt ( 3*(k_fs*torsionalShearStress(location/dx-2))^2 );
alternating = abs(k_f*bendingStress(location/dx-2));
goodmann_gk = 1/(alternating/S_e+midrange/S_ut);
DE_gk       = nDE(location/dx);
sig_gk      = bendingStress(location/dx);
tau_gk      = torsionalShearStress(location/dx);
```

```
Spline Safety Factors: Distortion Energy = 1.884e+02 , Goodman = 4.390e+01
Retaning Ring 1 Safety Factors: Distortion Energy = 2.495e+01 , Goodman = 1.998e+00
Shoulder Left Safety Factors: Distortion Energy = 6.449e+01 , Goodman = 9.894e+00
Shoulder Right Safety Factors: Distortion Energy = 8.667e+01 , Goodman = 1.224e+01
Retaning Ring 2 Safety Factors: Distortion Energy = 1.496e+02 , Goodman = 1.353e+01
Gibs Key Safety Factors: Distortion Energy = 1.727e+02 , Goodman = 6.812e+01
```

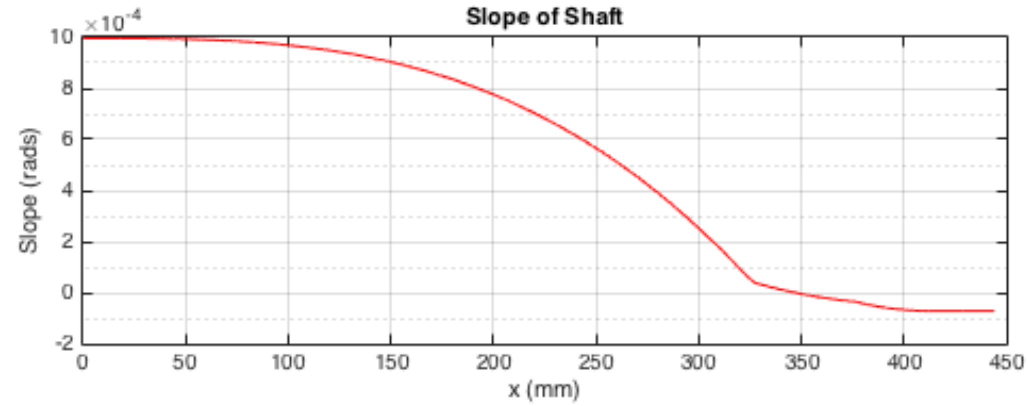
Deflection and Slope

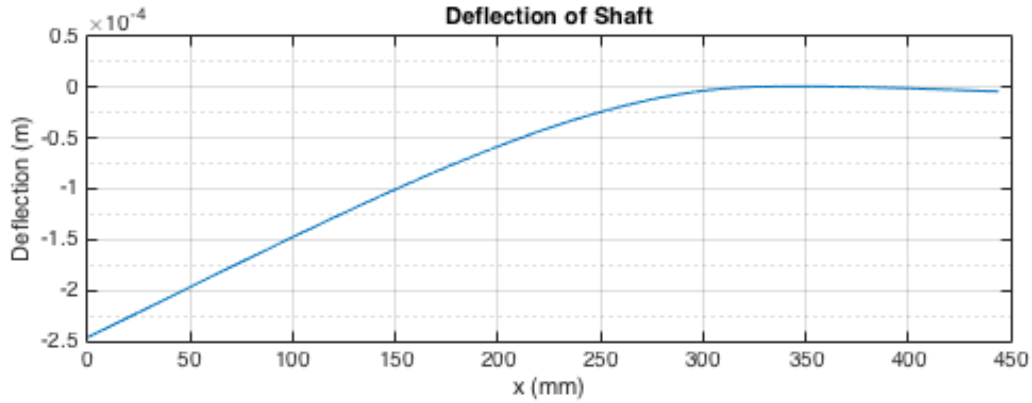
We first divide the moment by the appropraite EI for the radius of the shaft each location. We can then find the slope by intergating the result and the deflection by integrating again. The appropriate constants of integration are found by setting the deflection to be zero at the bearings. The slope is then corrected by a constant and the deflection is corrected by a linear function.

```
EI    = 0:dx:end_of_rod;
for i = 1:length(EI)
    x = i*dx;
    if(x < shoulderStart)
        EI(i) = (E*I1);
    elseif(x < shoulderEnd)
        EI(i) = (E*I2);
    elseif(x < vbelt_position_start)
        EI(i) = (E*I3);
    else
        EI(i) = (E*I4);
```

```
    end

end

slope_uncorrected            = cumtrapz(momentVec ./ EI) .* dx;
deflection_uncorrected       = cumtrapz(slope_uncorrected) .* dx;

%Constraing for zero deflection at bearings
boundary_correction = [ bearing1_position 1 -deflection_uncorrected(bearing1_position/dx);...
    bearing2_position 1 -deflection_uncorrected(bearing2_position/dx) ];
ans_boundary_correction = rref(boundary_correction);

C1 = ans_boundary_correction(1,3);
C2 = ans_boundary_correction(2,3);

slope_corrected       = slope_uncorrected + C1;
deflection_corrected  = deflection_uncorrected;

for i= 1:length(deflection_corrected)

    x = i*dx;
    deflection_corrected(i) = deflection_corrected(i) + C1*x + C2;
end
%Looking for deflection and slopes at important locations
deflection_bearing1     = deflection_corrected(bearing1_position/dx);
deflection_bearing2     = deflection_corrected(bearing2_position/dx);

slope_bearing1  = slope_corrected(bearing1_position/dx);
slope_bearing2  = slope_corrected(bearing2_position/dx);

deflection_spool    = deflection_corrected(1);
deflection_pulley   = deflection_corrected(end);
slope_spool         = slope_corrected(1);
slope_pulley        = slope_corrected(end);
```

```
Deflection at Bearing 1 = 000 m, Slope at Bearing 1  = -1.063e-04 rads
Deflection at Bearing 2 = 000 m, Slope at Bearing 2  = 5.160e-05 rads
Deflection at Spool = 2.463e-04 m, Slope at Spool = -9.962e-04 rads
Deflection at Pulley = 4.034e-06 m, Slope at Pulley = 7.229e-05 rads
```





Critical Speed

The critical speed for the shaft alone and for the shaft components alone is found using Rayleigh's method. For each case the deflection must be found only due to the weights of the elements that are being considered. This involves recalculating the reaction forces at the bearings, and finding the shear, moment, slope and deflection using the same method as above. The critical speeds are then combined using using Dunkerley's equation.

```
%Finding deflection due to the weight of the rod alone
shear_weight = 0:dx:end_of_rod;

calc_forces_y       = [ 1 1 -g*(mass1+mass2+mass3+mass4) ; bearing1_position...
    bearing2_position -g*(mass1*length1/2+mass2*(length1+length2/2)+mass3*...
    (length1+length2+length3/2)+mass4*(length1+length2+length3+length4/2))];
rref_calc_forces_y  = rref(calc_forces_y);
F_bearing1_y        = rref_calc_forces_y(1,3);
F_bearing2_y        = rref_calc_forces_y(2,3);

for i = 1:length(shear_weight);

    x = i*dx;
    if(x<bearing1_position)
        shear_weight(i) = mass1*g/length1 *x;
    elseif( x < length1)
        shear_weight(i) = mass1*g/length1 *x + F_bearing1_y;
    elseif(x < length1 + length2)
        shear_weight(i) = mass1*g + F_bearing1_y + mass2*g/length2 * (x-length1) ;
    elseif(x < bearing2_position)
        shear_weight(i) = mass1*g + F_bearing1_y + mass2*g + mass3*g/length3 * (x-length2-leng
th1);
```

```
    elseif(x < length1 + length2 + length3)
        shear_weight(i) = mass1*g + F_bearing1_y + mass2*g + mass3*g/length3 *...
            (x-length2-length1) + F_bearing2_y;
    else
        shear_weight(i) = mass1*g + F_bearing1_y + mass2*g + mass3*g+...
            F_bearing2_y + mass4*g/length4 * (x-length3-length2-length1);
    end
end

moment_weight                     = cumtrapz(shear_weight) * dx;
slope_uncorrected_weight          = cumtrapz(moment_weight ./ EI) .* dx;
deflection_uncorrected_weight     = cumtrapz(slope_uncorrected_weight) .* dx;

%Correcting for zero deflection at bearings
boundary_correction = [ bearing1_position 1 -deflection_uncorrected_weight(bearing1_position/d
x);...
    bearing2_position 1 -deflection_uncorrected_weight(bearing2_position/dx) ];
ans_boundary_correction = rref(boundary_correction);

C1 = ans_boundary_correction(1,3);
C2 = ans_boundary_correction(2,3);

slope_corrected_weight       = slope_uncorrected_weight +C1;
deflection_corrected_weight  = deflection_uncorrected_weight;

for i= 1:length(deflection_corrected_weight)

    x = i*dx;
    deflection_corrected_weight(i) = deflection_corrected_weight(i) + C1*x + C2;

end

%Calculating deflection due to the weight of the components alone
shear_components = 0:dx:end_of_rod;

calc_forces_y       = [ 1 1 -F_pulley-F_spool_y ; bearing1_position...
    bearing2_position -vbelt_position*F_pulley-spool_position*F_spool_y];
rref_calc_forces_y  = rref(calc_forces_y);
F_bearing1_y        = rref_calc_forces_y(1,3);
F_bearing2_y        = rref_calc_forces_y(2,3);

for i = 1:length(shear_components);

    x = i*dx;
    if(x<spool_position_end)
        shear_components(i) = Distrub_spool_weight * x;
    elseif( x < bearing1_position)
        shear_components(i) = F_spool_y;
    elseif(x < bearing2_position)
        shear_components(i) = F_spool_y + F_bearing1_y;
    elseif(x < vbelt_position)
        shear_components(i) = F_spool_y + F_bearing1_y + F_bearing2_y;
    else
        shear_components(i) = F_spool_y + F_bearing1_y + F_bearing2_y + F_pulley;
    end
end
```

```
moment_components                   = cumtrapz(shear_components) * dx;

slope_uncorrected_components        = cumtrapz(moment_components ./ EI) .* dx;
deflection_uncorrected_components   = cumtrapz(slope_uncorrected_components) .* dx;

%Correcting for zero deflection at bearings
boundary_correction = [ bearing1_position 1 -deflection_uncorrected_components(bearing1_positi
on/dx);...
    bearing2_position 1 -deflection_uncorrected_components(bearing2_position/dx) ];
ans_boundary_correction = rref(boundary_correction);

C1 = ans_boundary_correction(1,3);
C2 = ans_boundary_correction(2,3);

slope_corrected_components          = slope_uncorrected_components + C1;
deflection_corrected_components     = deflection_uncorrected_components;

for i= 1:length(deflection_corrected_components)

    x = i*dx;
    deflection_corrected_components(i) = deflection_corrected_components(i) + C1*x + C2;

end

%Finding Critical Speed from shaft weight
y1 = abs(deflection_corrected_weight(length1/2/dx));
y2 = abs(deflection_corrected_weight((length1+length2/2)/dx));
y3 = abs(deflection_corrected_weight((length1+length2+length3/2)/dx));
y4 = abs(deflection_corrected_weight((length1+length2+length3+length4/2)/dx));

omega_shaft = sqrt( ( -g * (mass1*y1 + mass2*y2 + mass3*y3 + mass4*y4))/...
    (mass1*y1^2 + mass2*y2^2 + mass3*y3^2 + mass4*y4^2));


%Finding Critical Speed from the component weight
y5 = deflection_corrected_components(round(spool_position/dx));
y6 = deflection_corrected_components(vbelt_position/dx);
omega_components = sqrt( (g * (F_spool_y*y5 + F_pulley*y6)) / (F_spool_y*y5^2 + F_pulley*y6^2)
 );


%Finding the overall criciitical speed
omega_crit    = sqrt( 1 / ( 1/omega_shaft^2 + 1/omega_components^2));
critical_rpm = omega_crit * 60 /2/pi;
```

```
Critical speed from weight of shaft = 1.873e+03 rad/s
Critical speed from weight of components = 3.735e+02 rad/s
Over critical speed = 3.663e+02 rad/s = 3.498e+03 RPM
```

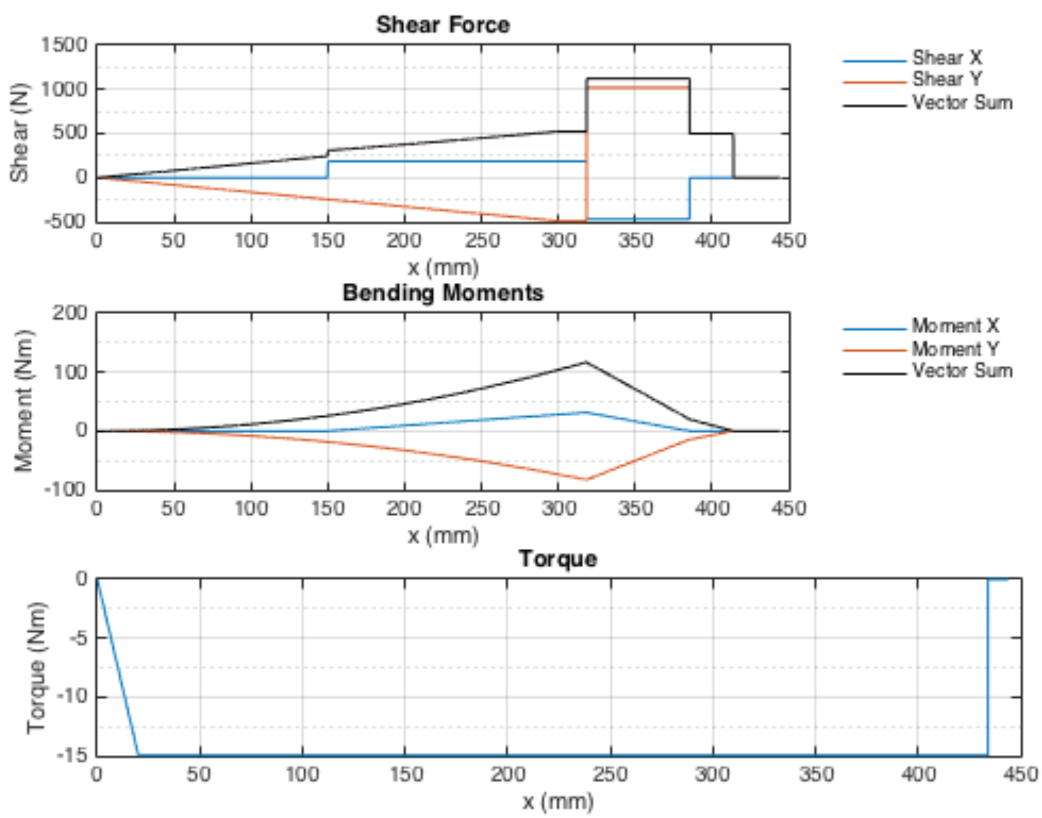
Cost and Performance

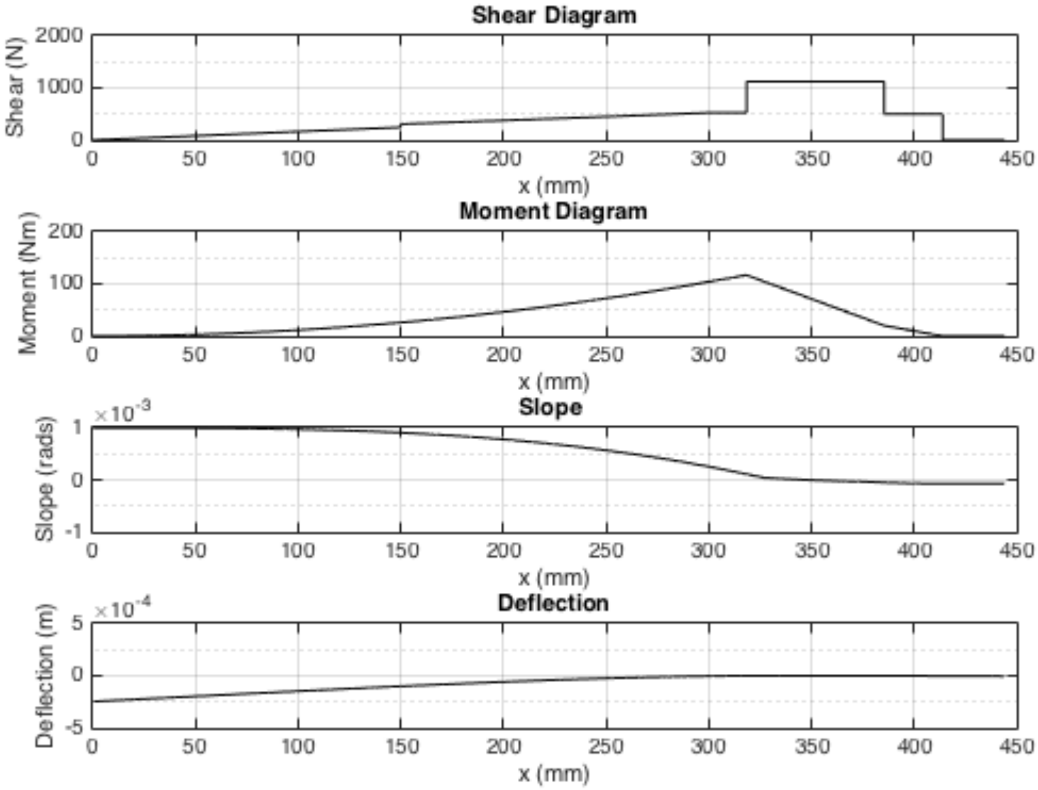
Calculating the cost and performance meteric.

```
totalMass       = mass1 + mass2 + mass3 + mass4; %kg
purchaseMass    = (length1+length2+length3+length4)*xsection2*density; %kg
removeMass      = purchaseMass-totalMass; %kg

materialCost    = 13; %$/kg
machiningCost   = 50; %$/kg

cost            = purchaseMass*materialCost+removeMass*machiningCost; %$
```

```
Total Mass = 3.644e+00 kg, Total Cost = $ 1.992e+02
```

Final Figures

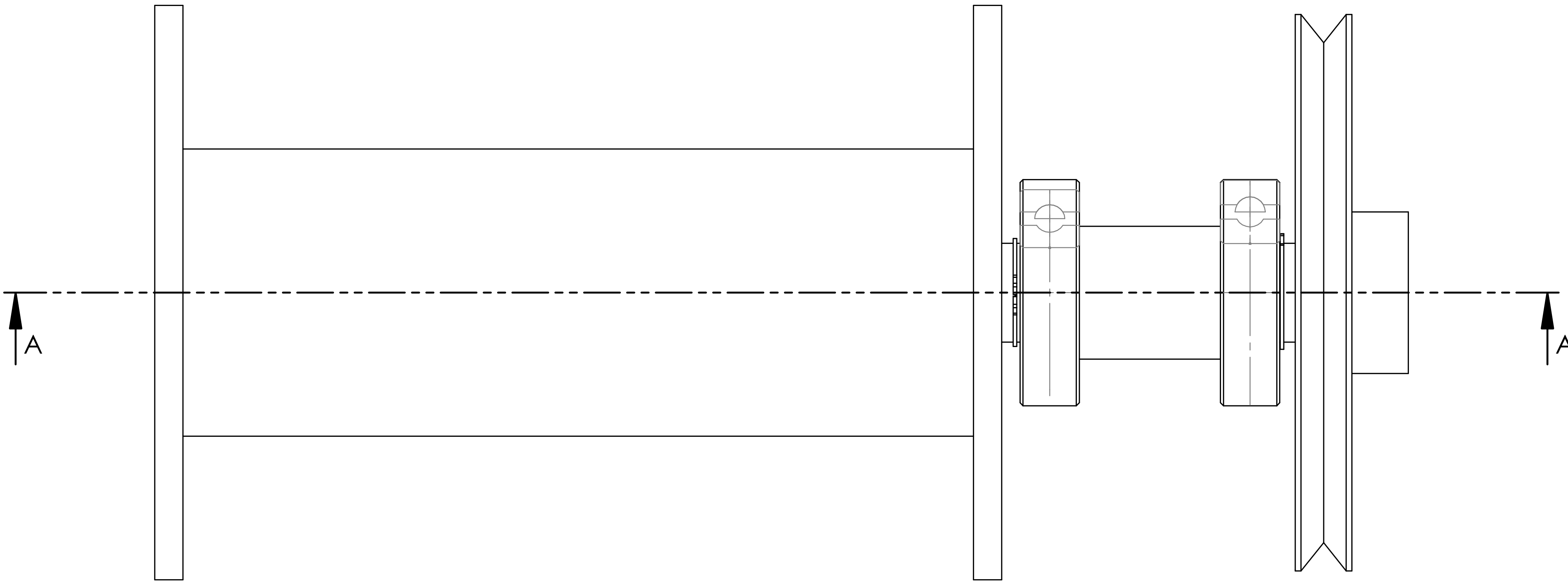




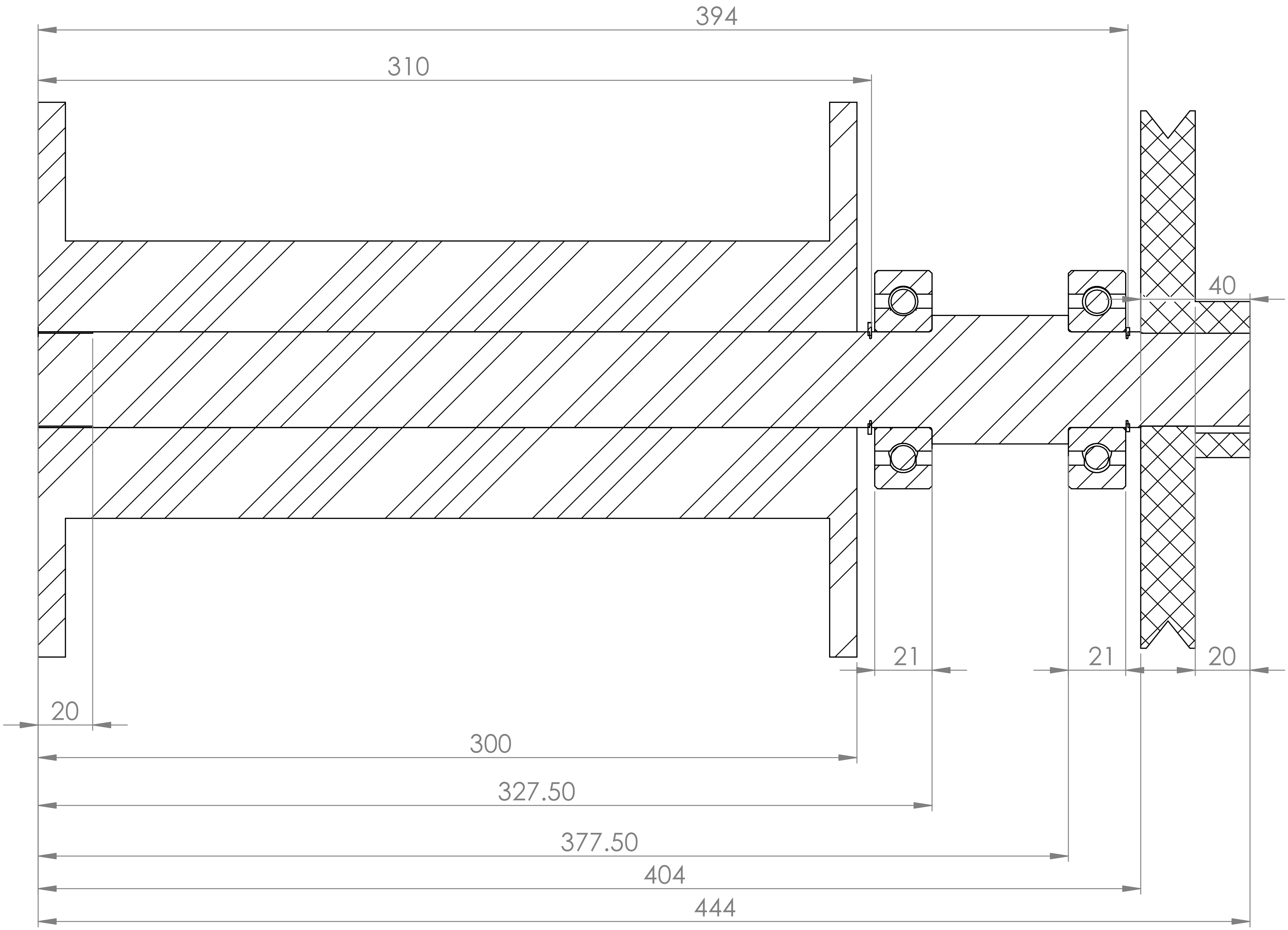
Critical Location	σ(MPa)	τ(MPa)	Distortion Energy	σ m (MPa)	σ a (MPa)	Goodman
Spline	0.11	1.77	188	15.3	0.11	43.9
Retaining Ring 1	26.3	1.77	24.9	8.37	113	2
Left Shoulder	10.2	1.77	64.5	2.52	22.7	9.89
Right Shoulder	7.66	1.77	86.7	6.13	17	12.2
Retaining Ring 2	3.33	1.77	150	8.37	14.4	13.5
Gib Head Key	8.99E-03	1.93	173	10.1	0.0195	68.1

Location	Deflection (mm)	Allowable (mm)	Slope (rad)	Allowable (rad)
Spool A	0.246	1	9.96E-04	0.008
Bearing B	0	0	1.06E-04	0.001
Bearing C	0	0	5.16E-05	0.001
Pulley D	0.00403	1	7.23E-05	0.008

Shaft	Critical Speed (rpm)	Actual Speed (rpm)
Top Shaft	3490	600



SOLIDWORKS Student License
Academic Use Only



SOLIDWORKS Student License
Academic Use Only

SECTION A-A