
Function: check_spring_condition

Find the force required to perform maintenance on the shaft. Parameters:

- B - Location of the bearing close to the pulley relative to point E
- C - Location of the bearing close to the coupling relative to E
- Fk1 - force at the first bearing due to the spring
- Fk2 - force at the second bearing due to the spring

Returns - Force Required to Lift the pulley ten centimeters for maintenance

Code

```
function force_required = check_spring_condition(B,C,Fk1,Fk2)
% Given
shaft_length = 40; % cm
angle_factor = 0.025; % sin(theta)=10mm/40cm, unitless
k = 37.8; % N/mm, spring factor

% Guide
% 1 - Bearing close to the pulley
% 2 - Bearing close to the coupling

% Find initial deflections (both in compression)
yi1 = abs(Fk1 / k); % mm
yi2 = abs(Fk2 / k); % mm

% Locations of Bearings relative to the coupling ( )
x1 = (shaft_length - B); % cm, Distance from B to D
x2 = (shaft_length - C); % cm, Distance from C to D

% Deflection on spring due to moving the shaft upwards 10mm
y1 = x1 * angle_factor * 10; % mm, in compression
y2 = x2 * angle_factor * 10; % mm, in tension

% Finding the actual deflections
y1 = y1 + yi1;
y2 = y2 - yi2;

% F = Fk1 + Fk2, where Fk = k * deflection
force_required = k*(y2 + y1);
end
```

*Error using check_spring_condition (line 25)
Not enough input arguments.*

Published with MATLAB® R2014b

Function: fatigue_stress_concentration

Finds Kfs and Kf using the Neuber Equation.

Parameters:

- Kt (or Kts) - theoretical stress concentration factor
- r - fillet radius in mm
- Sut - ultimate tensile strength of the material in MPa
- bending - 0 for torsion (Kfs), 1 for bending (Kf)

Returns - the fatigue stress concentration factor (in bending or in shear)

Code

```
function K = fatigue_stress_concentration(Kt,r,Sut,bending)
    Sut = Sut * 145.038/1000; % kpsi
    % r, notch radius
    if(bending == 1)
        % Bending case
        sqrt_a = (0.246 - 3.08*(10^-3)*Sut + 1.51*(10^-5)*Sut^2 - 2.67*(10^-8)*Sut^3) *
    else
        % Torsion case
        sqrt_a = (0.190 - 2.51*(10^-3)*Sut + 1.35*(10^-5)*Sut^2 - 2.67*(10^-8)*Sut^3) *
    end

    K = 1 + (Kt - 1) / (1 + sqrt_a/sqrt(r));
end
```

*Error using fatigue_stress_concentration (line 17)
Not enough input arguments.*

Published with MATLAB® R2014b

Function: fatigueLife

This script calculates the infinite life safety factor of a rotating cylindrical shaft using the Modified Goodman Failure Criteria. All units are in MPa. Sut is the ultimate tensile strength, Smax is the Maximum bending stress, Smin is the minimum bending stress. NO Axial Stress. TauMax is the shear stress due to torque from the motor. Kf and Kfs are fatigue stress concentrations for bending and shear. Finish should be 1 (ground), 2 (machined or CD), 3 (Hot-Rolled), 4 (As-Forged).

Code

```
function fatigueLife(Sut, Smax, Smin, TauMid, finish, diam, Kf, Kfs)

Sy = 580; % MPa

% Initialize Parameters
a = 0;
b = 0;
Se = 0;
Ka = 0;
Kb = 0;

%Midrange and Alternating Stresses

Smid = (Smax+Smin)/2;
Salt = abs(Smax-Smin)/2;

%Von Mises Alternating and Midrange stress
SaVM = Kf*Salt;
SmidVM = sqrt( (Kf*Smid)^2 + 3*(TauMid*Kfs)^2);

%Switch case takes material finish code and sets parameters for Surface
%Finish.
switch finish
    case 1
        a = 1.58;
        b = -0.085;
    case 2
        a = 4.51;
        b = -.265;
    case 3
        a = 57.7;
        b = -.718;
    case 4
        a = 272;
        b = -.995;
    otherwise
        'Invalid material finish.';
end

% Ka is the Surface Factor
Ka = a*Sut^b;
```

```
% Kb is the Size Factor
if( 2.79<= diam <=51)
    Kb = 1.24*diam^-0.107;
elseif( 51< diam <254)
    Kb = 1.51*diam^-0.157;
end

if (Sut > 1400)
    Se = 700*Ka*Kb;
else
    Se = 0.5*Sut*Ka*Kb;
end

nYield = Sy / (SaVM + SmidVM)

nGoodman = ( (SaVM / Se) + (SmidVM/Sut) ) ^ (-1)

nGerber = 0.5*((Sut/SmidVM)^2)*(SaVM/Se)*(-1 + sqrt(1+( 2*SmidVM*Se) / (Sut*SaVM)

end

Error using fatigueLife (line 19)
Not enough input arguments.
```

Published with MATLAB® R2014b

Function: plot_sf

Plots the Shear, Moment, Torque and M/I diagrams. Parameters - singularity functions for T,V,M and M/I; x is the position vector

Code

```
function plot_sf = plot_sf(T,V,M,M_over_I,x)
    % Setting up
    L = 40; % cm, shaft length

    % Shear Diagram
    figure(1)
    subplot(2,2,1)
    plot(x,V)
    axis([0 L min(V)-50 max(V)+50]) % for making the plots more appealing
    title('Shear Diagram')
    ylabel('Shear (N)')
    xlabel('Position on shaft (cm)')

    % Torque
    subplot(2,2,2)
    plot(x,T, 'g')
    axis([0 L -1 max(T)+1])
    title('Torque Diagram')
    ylabel('Torque (N*m)')
    xlabel('Position on shaft (cm)')

    % Moment Diagram
    subplot(2,2,[3,4])
    plot(x,M, 'r')
    axis([0 L -2 max(M)+2])
    title('Moment Diagram')
    ylabel('Moment (N*m)')
    xlabel('Position on shaft (cm)')

    % M over I diagram
    figure(2)
    plot(x,M_over_I, 'c')
    title('Moment over I Diagram')
    ylabel('M/I (N/m^3)')
    xlabel('Position on shaft (cm)')
end
```

*Error using plot_sf (line 15)
Not enough input arguments.*

Published with MATLAB® R2014b