```
run1 = [-0.439, 0.000, 0.670];
run2 = [-0.037, -0.213, 1.125];
run3 = [-0.026, -0.005, 0.932];
runs = [run1;run2;run3];
%input for run 1, 2, & 3. x y z change, w/ z as change in knot
 position.
%matrix w/ 3 rows & 3 columns. each column contains an x, y, or z,
 each row
%organizes them into runs.
%will need to subtract changes, in order for the knot to be the
 origin.

CW = [590.9, 730.2, 1000.8; 858.8, 699.2, 1330.4; 637.2, 770.3,
 1253.0]/1000 * 9.80665;
%counter weights in grams converted to kilograms & multiplied by
%acceleration due to gravity to get netwtons.
%[run1 W2, W3, W4; run2 W2, W3, W4, run3 W2, W3, W4]

%matrix(rows,columns)
%calculate frictions, tensions, and coefficients of friction
% for each run and and assign them to variables
[friction1, tension1, cfs1] = dorun(run1, CW(1,1:3));
[friction2, tension2, cfs2] = dorun(run2, CW(2,1:3));
[friction3, tension3, cfs3] = dorun(run3, CW(3,1:3));

%print out frictions, tensions, and coefficients of friction
disp("run 1 tensions: T2, T3, T4")
disp(tension1);
disp("run 2 tensions: T2, T3, T4");
disp(tension2);
disp("run 3 tensions: T2, T3, T4 ");
disp(tension3);
disp("run 1 frictions: T2, T3, T4")
disp(friction1);
disp("run 2 frictions: T2, T3, T4");
disp(friction2);
disp("run 3 frictions: T2, T3, T4 ");
disp(friction3);
disp("run 1 coefficifents of friction");
disp(cfs1);
disp("run 2 coefficients of friction");
disp(cfs2)
disp("run 3 coefficients of friction:")
disp(cfs3);

%calculate new tensions, frictions, and coefficients of friction (t1-
f1)
%for when each coordinate is individually altered by 2 and the weights
 are
%altered by 2 grams from the values used to calculate the tensions,
 frictions, and coefficients of friction
% for run 1.
```

```
%also calculate the magnitudes of the changes for each output (dt1,
 df1,
%dc1).
[t1, f1, c1, dt1, df1, dc1] = sensitivity(run1, CW(1,1:3));

%put those values into a spreadsheet, given cell to start at.
writematrix(t1, "Lab1 data.xlsx", 'Sheet', 1, 'Range', 'A2');
writematrix(f1, "Lab1 data.xlsx", 'Sheet', 1, 'Range', 'E2');
writematrix(c1, "Lab1 data.xlsx", 'Sheet', 1, 'Range', 'I2');

writematrix(dt1, "Lab1 data.xlsx", 'Sheet', 1, 'Range', 'A15');
writematrix(df1, "Lab1 data.xlsx", 'Sheet', 1, 'Range', 'E15');
writematrix(dc1, "Lab1 data.xlsx", 'Sheet', 1, 'Range', 'I15');
%sensitivity analysis using run 1 data:

function [t, f, c, dt, df, dc] = sensitivity(run, CW)
    %CW = [W2, W3, W4]
    [frictions, tensions, cfs] = dorun(run, CW);%calculate frictions,
 tensions, & coefficients of frictions
    w = 2/1000 * 9.80665;%2 grams in newtons
    m = 0.5/100;%0.5cm in meters.
    f = [];
    t = [];
    c = [];
    changes = [[w, 0, 0]; [-w, 0, 0]; [0, w, 0]; [0, -w, 0]; [0, 0,
 w]; [0, 0, -w]; [m, 0, 0]; [-m, 0, 0]; [0, m, 0]; [0, -m, 0]; [0, 0,
 m]; [0, 0, -m]];
    for j = [1, 2, 3, 4, 5, 6]
        [f(j,1:3), t(j,1:3), c(j,1:3)] = dorun(run, CW
+changes(j,1:3));
    end
    for g = [7, 8, 9, 10, 11, 12]
        [f(g,1:3), t(g,1:3), c(g,1:3)] = dorun(run+changes(g,1:3),
 CW);
    end
    %these loops perform the sensitivity analysis by calculating
    %the friction magnitudes, tensions, and coefficients of friction
    %for + or - 2grams for a counterweight
    %or + or - 0.5cm for one of the x, y, or z coordinates.
    %each result is stored in a row of one of the matrices.

    df = [];%magnitudes of the difference between the original and
 varied ones.
    dt = [];%each column will correspond to a difference.
    dc = [];
    for k = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
        df(k,1:3) = abs(frictions - f(k,1:3));
        dt(k,1:3) = abs(tensions - t(k,1:3));
        dc(k,1:3) = abs(cfs - c(k,1:3));
    end
end


function [friction, tensions, cfs] = dorun(run, W2_4)
```

```
    %CW = [W2, W3, W4]
    %run = change in [x, y, z]
    T2Rope = [-2.427,-1.716,2.962];%vectors for each counterweight
 rope relative to the original origin, in meters
    T3Rope = [-2.420,1.622,2.964];
    T4Rope = [2.390,-0.180,2.965];

    %RK is for w/ Respect to Knot.
    T2RK = T2Rope - run(1,:);%position vectors with respect to x, y, &
 z coordinates of the knot.
    T3RK = T3Rope - run(1,:);
    T4RK = T4Rope - run(1,:);

    r1 = 0.375/2*0.0254;
    r2 = 1.5/2*0.0254;

    AC2 = sqrt(power(T2RK(1),2) + power(T2RK(2),2));%AC =
 sqrt(x^2+y^2)
    AC3 = sqrt(power(T3RK(1),2) + power(T3RK(2),2));
    AC4 = sqrt(power(T4RK(1),2) + power(T4RK(2),2));

    AE2 = AC2 - r2;%sheave_radius is measured from center, which is
 why it = r2.
    AE3 = AC3 - r2;
    AE4 = AC4 - r2;

    z = [T2RK(3), T3RK(3), T4RK(3)];
    alpha2 = atan(z(1)/AE2);
    alpha3 = atan(z(2)/AE3);
    alpha4 = atan(z(3)/AE4);
    beta2 = asin(r2/sqrt((power(AE2,2)+power(z(1),2))));
    beta3 = asin(r2/sqrt((power(AE3,2)+power(z(2),2))));
    beta4 = asin(r2/sqrt((power(AE4,2)+power(z(3),2))));
    z_prime = [AE2*tan(alpha2 + beta2), AE3*tan(alpha3 + beta3),
 AE4*tan(alpha4 + beta4)];

%distance magnitudes with z_prime for each rope.
    magT = [sqrt(power(T2RK(1),2) + power(T2RK(2),2) +
 power(z_prime(1),2)), sqrt(power(T3RK(1),2) + power(T3RK(2),2) +
 power(z_prime(2),2)), sqrt(power(T4RK(1),2) + power(T4RK(2),2) +
 power(z_prime(3),2))];
    %this time, position vectors with z_prime
    T2 = [T2RK(1), T2RK(2), z_prime(1)];%distance vectors w/ z
 correction
    T3 = [T3RK(1), T3RK(2), z_prime(2)];%you said T3RK(3) instead of
 T3RK(2).
    T4 = [T4RK(1), T4RK(2), z_prime(3)];%used T3RK instead of T4!
    unit = [T2/magT(1); T3/magT(2); T4/magT(3)];%& divided T3 by
 magT(3) .
    %each row of unit contains the x,y,z unit vector for the tensions
 in the
    %ropes corrected for the z coordinate.
    %friction doesn't factor into calculating the unit vector because
 we
```

```
    %we're using distance, not force.
    %1 Newton moves 1 kg 1m/s^2. acceleration due to gravity is
 9.80665 m/s^2
    W1 = 1.5067 * 9.80665; %kg*acceleration due to gravity = newtons.
 downward weight should be in grams or kilograms.

    % finds item based on (row#, column#).
    A = [unit(1,1) unit(2,1) unit(3,1); unit(1,2), unit(2,2),
 unit(3,2); unit(1, 3), unit(2, 3), unit(3, 3)];
    %organizes unit vector such that each row only contains xs, ys, or
 zs.
    b = [0; 0; W1];%W1 is in newtons
    T = inv(A)*b;
    %T gives tensions [T2; T3; T4];

    Tv = [T(1)*unit(1,1:3); T(2)*unit(2,1:3); T(3)*unit(3,1:3)];
    %each column contains a vector for T.
    R = [-(Tv(1,1:3)+[0,0,W2_4(1)]); -(Tv(2,1:3)+[0,0,W2_4(2)]); -
(Tv(3,1:3)+[0,0,W2_4(3)])];
    %force that resists T & W R = -(T+W)
    %these values should give us the friction in Newtons.
    %f = (T - W)*r2/r1. this is for W2, run 1.
    friction = [abs((T(1)-W2_4(1)))*r2/r1, abs((T(2)-W2_4(2)))*r2/r1,
 abs((T(3)-W2_4(3)))*r2/r1];
    cfs = [];
    for i = [1 2 3]
        cfs(i) = [friction(i)/sqrt((norm(R(i,1:3)))^2 +
 (friction(i))^2)];
        %norm(R(i,1:3)) can be replaced w/ T(i)+W2_4(i)
    end
    %frictions for T2, T3, T4 hinges.
    tensions = [T(1), T(2), T(3)];
end
```

run 1 tensions: T2, T3, T4
5.8773   7.1405   9.6202

run 2 tensions: T2, T3, T4
7.5658   6.2331   12.0397

run 3 tensions: T2, T3, T4
5.6424   7.2297   11.3751

run 1 frictions: T2, T3, T4
0.3302   0.0813   0.7771

run 2 frictions: T2, T3, T4
3.4244   2.4949   4.0281

run 3 frictions: T2, T3, T4
2.4254   1.2974   3.6504

run 1 coefficifents of friction
0.0310   0.0062   0.0442

```
run 2 coefficients of friction
    0.2363    0.2131    0.1762

run 3 coefficients of friction:
    0.2242    0.0983    0.1675


Error using writematrix (line 134)
Unable to save the workbook to file '/MATLAB/toolbox/matlab/codetools/
Lab1 data.xlsx'. Check that write permissions are available, there is
 sufficient disk space, and the file can be written to or created.

Error in Lab1 (line 51)
writematrix(t1, "Lab1 data.xlsx", 'Sheet', 1, 'Range', 'A2');
```

Published with MATLAB® R2019b