Reginald Harris
RDH252

Hand Calculations:

Base Design:

```
M = 600;
L = 16;
h = 0.75;
b = 0.5;
c = 1.0;
a = 0.04;
E = 32.E6;
su = 370.E3;
KIC = 15.E3;
sfatigue = 115.E3;

I = (b*h^3)/12;

sig_max = (6*M)/(b*h^2);
u_max = ((M/L)*L^3)/(3*E*I);
strain = ((((M/L)*(L-c)*6)/(b*h^2))/E)*1.E6;
output = strain/1.E3;

K_I = 1.12*sig_max*sqrt(pi*a);
X_O = su/sig_max;
X_K = KIC/K_I;
X_S = sfatigue/sig_max;

disp(['Stress (psi): ' num2str(sig_max)])
disp(['Deflection (in): ' num2str(u_max)])
disp(['Strain (microstrain): ' num2str(strain)])
disp(['Output (mV/V): ' num2str(output)])
disp(['Factor of Safety - Strength XO: ' num2str(X_O)])
disp(['Factor of Safety - Fracture XK: ' num2str(X_K)])
disp(['Factor of Safety - Fatigue XS: ' num2str(X_S)])
```

M42 Steel Results:

Deflection: .091 in
Stress: 12.8 ksi
Strain: 375 microstrains
Output: .375 mV/v
Safety factor for strength = 28.9
Safety factor for crack growth = 2.95
Safety factor for fatigue = 8.98

Iterative:

```
M  = 600;
a  = 0.04;

target_vout = 1.0;
XO_min = 4.0;
XK_min = 2.0;
XS_min = 1.5;

materials(1).name     = 'M42 Steel';
materials(1).E        = 32e6;
materials(1).su       = 370e3;
materials(1).KIC      = 15e3;
materials(1).sfatigue = 115e3;

materials(2).name     = 'Aluminum Alloy';
materials(2).E        = 10.4e6;
materials(2).su       = 80e3;
materials(2).KIC      = 25e3;
materials(2).sfatigue = 20e3;

materials(3).name     = 'Titanium Alloy';
materials(3).E        = 16e6;
materials(3).su       = 130e3;
materials(3).KIC      = 55e3;
materials(3).sfatigue = 70e3;

h_vals = 0.30:0.02:1.00;
b_vals = 0.30:0.02:0.80;
L_vals = 12:0.5:20;
c_vals = 0.25:0.25:4.00;

best_found_global_safe = false;
best_vout_global_safe  = -inf;
best_safe = struct();

best_vout_global_any = -inf;
best_any = struct();

for m = 1:numel(materials)
```

```
for m = 1:numel(materials)

    E        = materials(m).E;
    su       = materials(m).su;
    KIC      = materials(m).KIC;
    sfatigue = materials(m).sfatigue;
    mname    = materials(m).name;

    best_vout_mat_safe = -inf;
    best_found_mat_safe = false;
    best_mat_safe = struct();

    best_vout_mat_any = -inf;
    best_mat_any = struct();

    for iL = 1:numel(L_vals)
        L = L_vals(iL);

        for ic = 1:numel(c_vals)
            c = c_vals(ic);

            if c >= L
                continue
            end

            for ih = 1:numel(h_vals)
                h = h_vals(ih);

                for ib = 1:numel(b_vals)
                    b = b_vals(ib);

                    Ixx    = b*h^3/12;
                    sig_max = (6*M)/(b*h^2);
                    u_max   = (M/L)*(L^3)/(3*E*Ixx);

                    eps_g = ((((M/L)*(L - c)*6)/(b*h^2))/E)*1e6;
                    vout  = eps_g/1e3;

                    KI = 1.12*sig_max*sqrt(pi*a);
                    XO = su/sig_max;
                    XK = KIC/KI;
                    XS = sfatigue/sig_max;

                    safe = (vout >= target_vout) && (XO >= XO_min) && (XK >= XK_min) && (XS >= XS_min);
```

```
                if vout > best_vout_mat_any
                    best_vout_mat_any = vout;
                    best_mat_any.material = mname;
                    best_mat_any.L        = L;
                    best_mat_any.c        = c;
                    best_mat_any.h        = h;
                    best_mat_any.b        = b;
                    best_mat_any.sig_max  = sig_max;
                    best_mat_any.u_max    = u_max;
                    best_mat_any.XO       = XO;
                    best_mat_any.XK       = XK;
                    best_mat_any.XS       = XS;
                    best_mat_any.eps_g    = eps_g;
                    best_mat_any.vout     = vout;
                end

                if vout > best_vout_global_any
                    best_vout_global_any = vout;
                    best_any = best_mat_any;
                end

                if safe && vout > best_vout_mat_safe
                    best_vout_mat_safe = vout;
                    best_found_mat_safe = true;
                    best_mat_safe.material = mname;
                    best_mat_safe.L        = L;
                    best_mat_safe.c        = c;
                    best_mat_safe.h        = h;
                    best_mat_safe.b        = b;
                    best_mat_safe.sig_max  = sig_max;
                    best_mat_safe.u_max    = u_max;
                    best_mat_safe.XO       = XO;
                    best_mat_safe.XK       = XK;
                    best_mat_safe.XS       = XS;
                    best_mat_safe.eps_g    = eps_g;
                    best_mat_safe.vout     = vout;
                end

                if safe && vout > best_vout_global_safe
                    best_vout_global_safe = vout;
                    best_found_global_safe = true;
                    best_safe = best_mat_safe;
                end
            end
        end
```

```
                    end
                end
            end
        end
    end

    fprintf('\n===== Best Design for %s =====\n', mname);
    if best_found_mat_safe
        fprintf('L = %.2f in\n', best_mat_safe.L);
        fprintf('c = %.2f in\n', best_mat_safe.c);
        fprintf('h = %.3f in\n', best_mat_safe.h);
        fprintf('b = %.3f in\n', best_mat_safe.b);
        fprintf('deflection = %.3f in\n', best_mat_safe.u_max);
        fprintf('stress = %.2f ksi\n', best_mat_safe.sig_max/1e3);
        fprintf('XO = %.2f\n', best_mat_safe.XO);
        fprintf('XK = %.2f\n', best_mat_safe.XK);
        fprintf('XS = %.2f\n', best_mat_safe.XS);
        fprintf('strain = %.0f microstrain\n', best_mat_safe.eps_g);
        fprintf('output = %.2f mV/V\n', best_mat_safe.vout);
        fprintf('meets all design requirements.\n');
    else
        fprintf('No safe design for this material. Best ANY design:\n');
        fprintf('L = %.2f in\n', best_mat_any.L);
        fprintf('c = %.2f in\n', best_mat_any.c);
        fprintf('h = %.3f in\n', best_mat_any.h);
        fprintf('b = %.3f in\n', best_mat_any.b);
        fprintf('deflection = %.3f in\n', best_mat_any.u_max);
        fprintf('stress = %.2f ksi\n', best_mat_any.sig_max/1e3);
        fprintf('XO = %.2f\n', best_mat_any.XO);
        fprintf('XK = %.2f\n', best_mat_any.XK);
        fprintf('XS = %.2f\n', best_mat_any.XS);
        fprintf('strain = %.0f microstrain\n', best_mat_any.eps_g);
        fprintf('output = %.2f mV/V\n', best_mat_any.vout);
    end
end

fprintf('\n===== OVERALL BEST SAFE DESIGN =====\n');
if best_found_global_safe
    fprintf('material = %s\n', best_safe.material);
    fprintf('L = %.2f in\n', best_safe.L);
    fprintf('c = %.2f in\n', best_safe.c);
    fprintf('h = %.3f in\n', best_safe.h);
    fprintf('b = %.3f in\n', best_safe.b);
    fprintf('deflection = %.3f in\n', best_safe.u_max);
    fprintf('stress = %.2f ksi\n', best_safe.sig_max/1e3);
    fprintf('XO = %.2f\n', best_safe.XO);
```

```
fprintf('\n===== OVERALL BEST SAFE DESIGN =====\n');
if best_found_global_safe
    fprintf('material = %s\n', best_safe.material);
    fprintf('L = %.2f in\n', best_safe.L);
    fprintf('c = %.2f in\n', best_safe.c);
    fprintf('h = %.3f in\n', best_safe.h);
    fprintf('b = %.3f in\n', best_safe.b);
    fprintf('deflection = %.3f in\n', best_safe.u_max);
    fprintf('stress = %.2f ksi\n', best_safe.sig_max/1e3);
    fprintf('X0 = %.2f\n', best_safe.X0);
    fprintf('XK = %.2f\n', best_safe.XK);
    fprintf('XS = %.2f\n', best_safe.XS);
    fprintf('strain = %.0f microstrain\n', best_safe.eps_g);
    fprintf('output = %.2f mV/V\n', best_safe.vout);
else
    fprintf('No fully safe design exists. Best ANY design overall:\n');
    fprintf('material = %s\n', best_any.material);
    fprintf('L = %.2f in\n', best_any.L);
    fprintf('c = %.2f in\n', best_any.c);
    fprintf('h = %.3f in\n', best_any.h);
    fprintf('b = %.3f in\n', best_any.b);
    fprintf('deflection = %.3f in\n', best_any.u_max);
    fprintf('stress = %.2f ksi\n', best_any.sig_max/1e3);
    fprintf('X0 = %.2f\n', best_any.X0);
    fprintf('XK = %.2f\n', best_any.XK);
    fprintf('XS = %.2f\n', best_any.XS);
    fprintf('strain = %.0f microstrain\n', best_any.eps_g);
    fprintf('output = %.2f mV/V\n', best_any.vout);
end
```

Best Overall Design:

Material: Titanium Alloy

L = 20in

C = .25

h= .54 in

b= .38 in

Deflection = 1 in

Stress = 32.49 ksi

Strain = 2005 microstains

Output = 2.01 mV/V

Safety factor for strength = 4

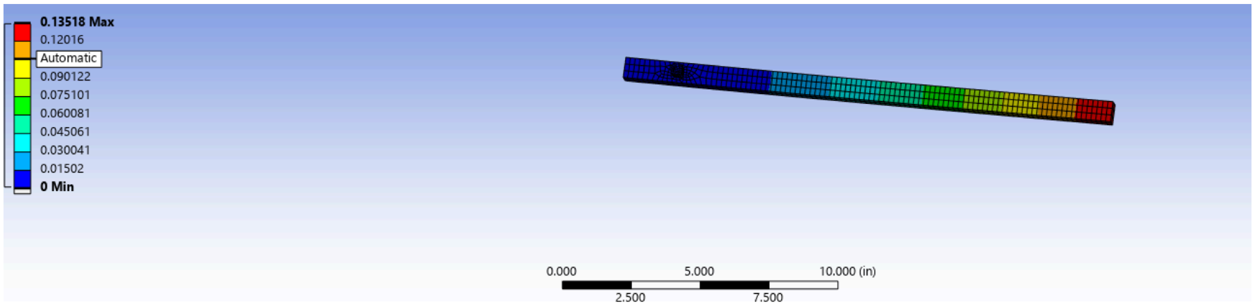
Safety factor for crack growth = 4.26

Safety factor for fatigue = 2.15

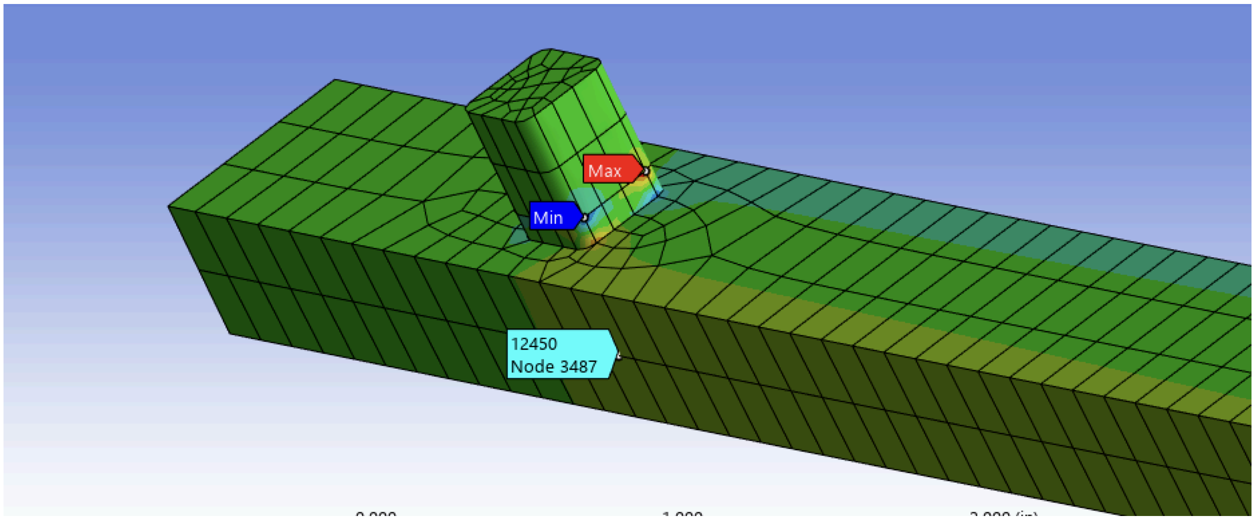
FEM Demo Results:

Material: M42 Steel

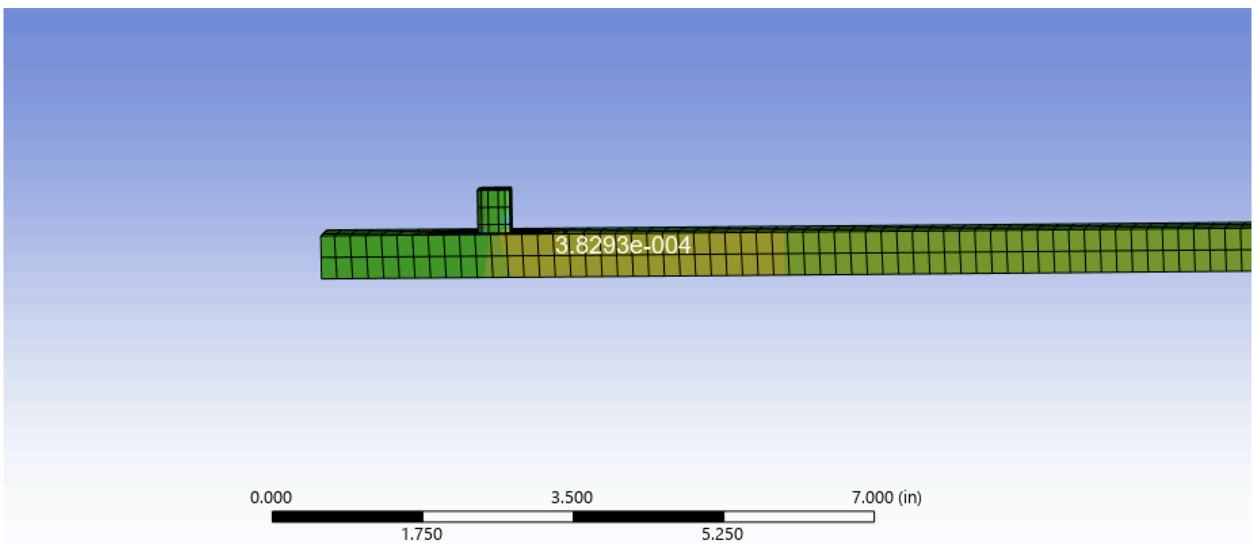
- Load Point Deflection = .13518 in

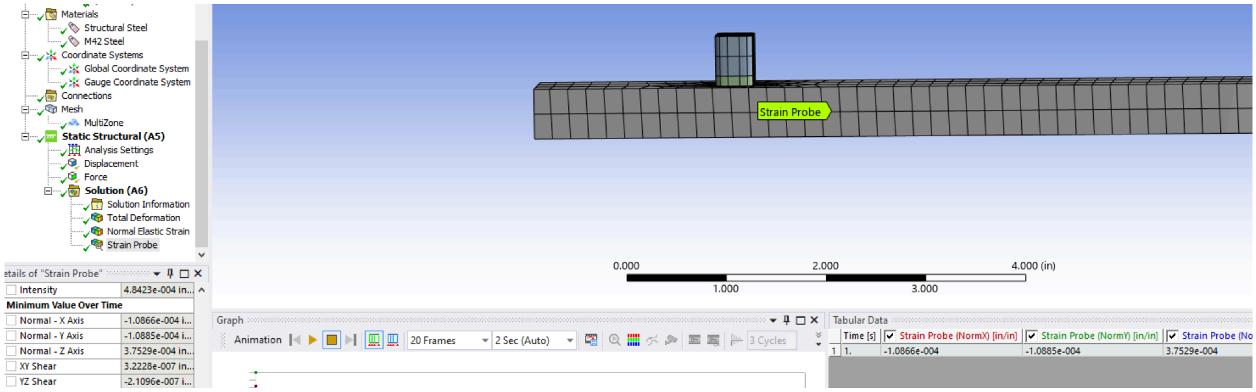


- Max Normal Stress = 12.45 ksi (along torque wrench)



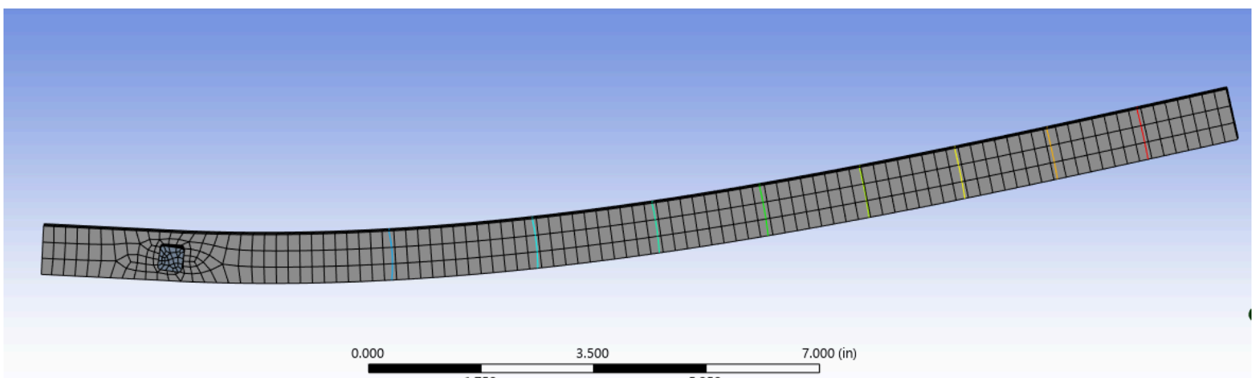
- Strain @ Strain Gauge ~ 383 microstrains/ 375 microstrains (exactly)





Discussion:

1. As pictured below, we can see that the mesh lines remain appropriately shaped across the wrench handle and only begin to deform significantly near the head. From this, it is safe to assume that plane sections remain plane even under beam deflection, and thus, beam theory is reasonably accurate.



2. The maximum normal stress of the FEM analysis was around 55 ksi, while the hand calculations (the ones that matched the dimensions of the base design) returned a normal stress of around 12.8 ksi. This difference is mainly because the hand calculations didn't take into account the stress at the wrench head, which is the exact location of the maximum normal stress. Given that, when we examine the maximum stress along the main wrench body, we find a value of 12.45 ksi, which is relatively close to our calculated value.
3. Our hand calculations generated a displacement of around .09 inches; however, our FEM generated a displacement of about .135 inches, which is reasonably accurate in respect to the scale of the wrench.