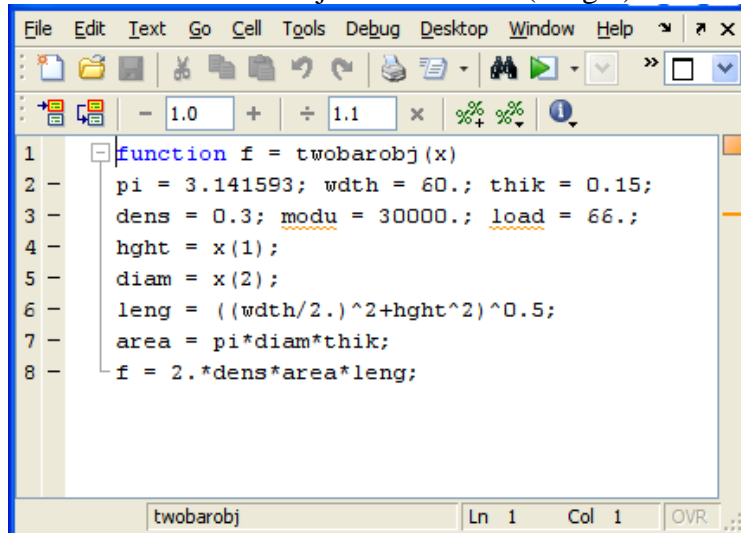


ME 575

Solving the Twobar Truss Problem Using MATLAB

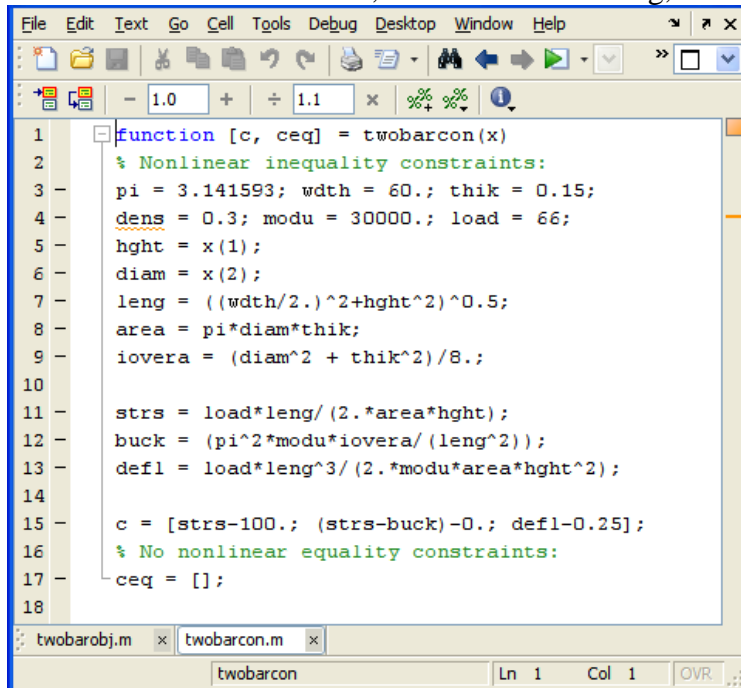
1. Create an “.m” file which has the objective function (weight) in it:



```
1 function f = twobarobj(x)
2     pi = 3.141593; width = 60.; thik = 0.15;
3     dens = 0.3; modu = 30000.; load = 66.;
4     hght = x(1);
5     diam = x(2);
6     leng = ((width/2.)^2+hght^2)^0.5;
7     area = pi*diam*thik;
8     f = 2.*dens*area*leng;
```

Recall that the file needs to have the same name as the function--in this case, “twobarobj.m”. Note in the above file that only height and diameter will be design variables (they are passed into the function).

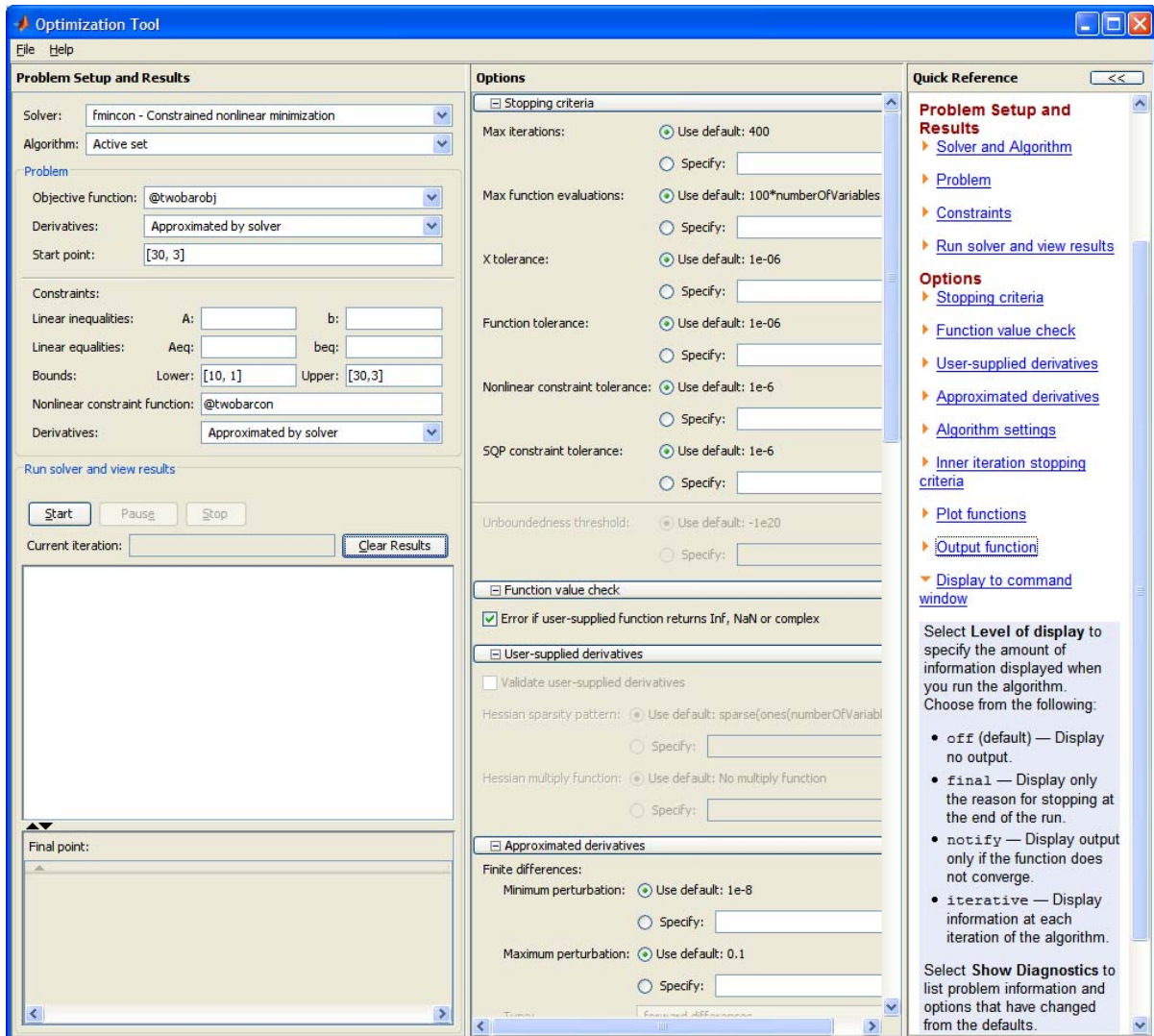
2. Create a file which has constraints stress, stress minus buckling, and deflection in it:



```
1 function [c, ceq] = twobarcon(x)
2     % Nonlinear inequality constraints:
3     pi = 3.141593; width = 60.; thik = 0.15;
4     dens = 0.3; modu = 30000.; load = 66;
5     hght = x(1);
6     diam = x(2);
7     leng = ((width/2.)^2+hght^2)^0.5;
8     area = pi*diam*thik;
9     iovera = (diam^2 + thik^2)/8.;
10
11     strs = load*leng/(2.*area*hght);
12     buck = (pi^2*modu*iovera/(leng^2));
13     defl = load*leng^3/(2.*modu*area*hght^2);
14
15     c = [strs-100.; (strs-buck)-0.; defl-0.25];
16     % No nonlinear equality constraints:
17     ceq = [];
```

In the constraint file, we note that all inequality constraints are written as ≤ 0 constraints, where the right hand side is brought over to the left.

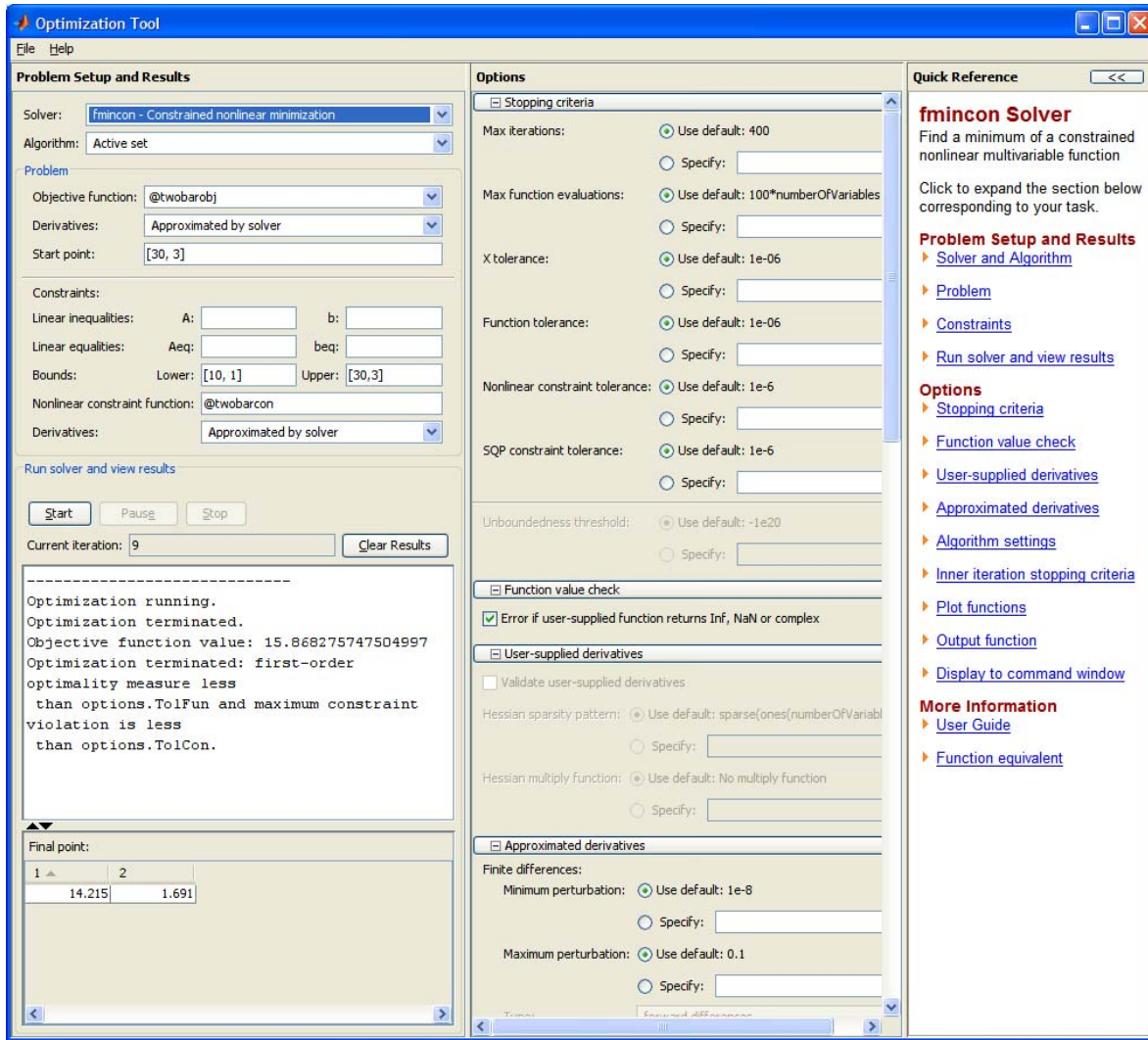
3. Open the optimization GUI by typing “optimtool” on the command line. A window opens up:



In this window, set the solver to be “fmincon”. Choose Active Set. In the Problem area, indicate the file where the objective function is coded, using the “@” symbol in front of the name. Set the starting point for the variables as a vector enclosed in brackets. In the Constraints area, indicate the lower bounds and upper bounds for variables as vectors. Indicate the file where the constraints are located.

On the right hand side of the window are numerous options, which are explained in the Quick Reference section.

4. When you are ready, push the Start button in the Solver area.

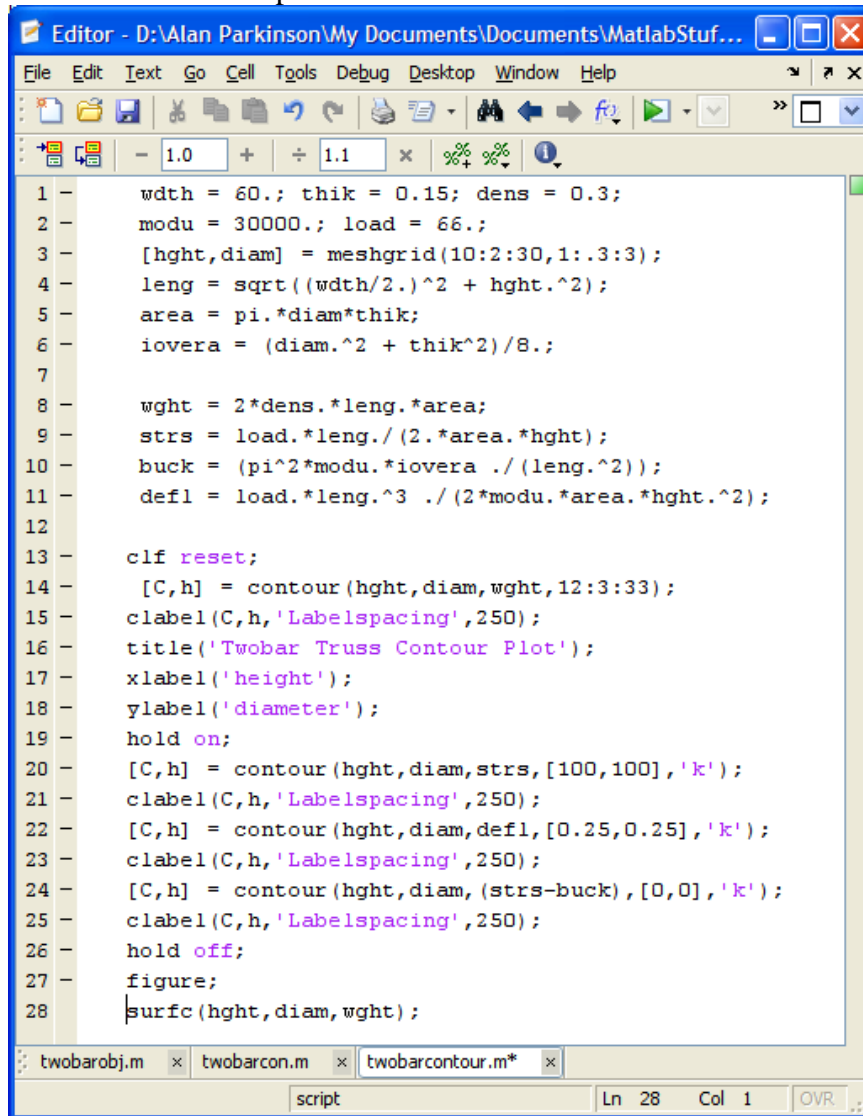


The final design point is given in the lower left window. Also, because we had the “Display to command window” option set to “Iterative”, the following was printed to the command window;

Iter	F-count	f(x)	Max constraint	Line search steplength	Directional derivative	First-order optimality	Procedure
0	3	35.9874	0				
1	6	21.4631	-0.1392	1	-14.3	3.1	
2	9	19.6565	0.01999	1	-1.8	0.449	Hessian modified
3	12	19.1516	0.06321	1	-0.498	0.524	
4	16	15.9826	6.596	0.5	-5.52	6.83	
5	20	15.6403	5.186	0.5	-0.608	0.319	
6	23	15.848	0.288	1	0.21	0.177	
7	26	15.8678	0.007528	1	0.0198	0.0302	
8	29	15.8683	1.666e-006	1	0.000481	7.77e-005	
9	32	15.8683	1.658e-008	1	7.84e-008	2.64e-008	Hessian modified

Optimization terminated: first-order optimality measure less than options.TolFun and maximum constraint violation is less than options.TolCon.
Active inequalities (to within options.TolCon = 1e-006):
lower upper ineqlin ineqnonlin
2
3

Next we show code for a contour plot:

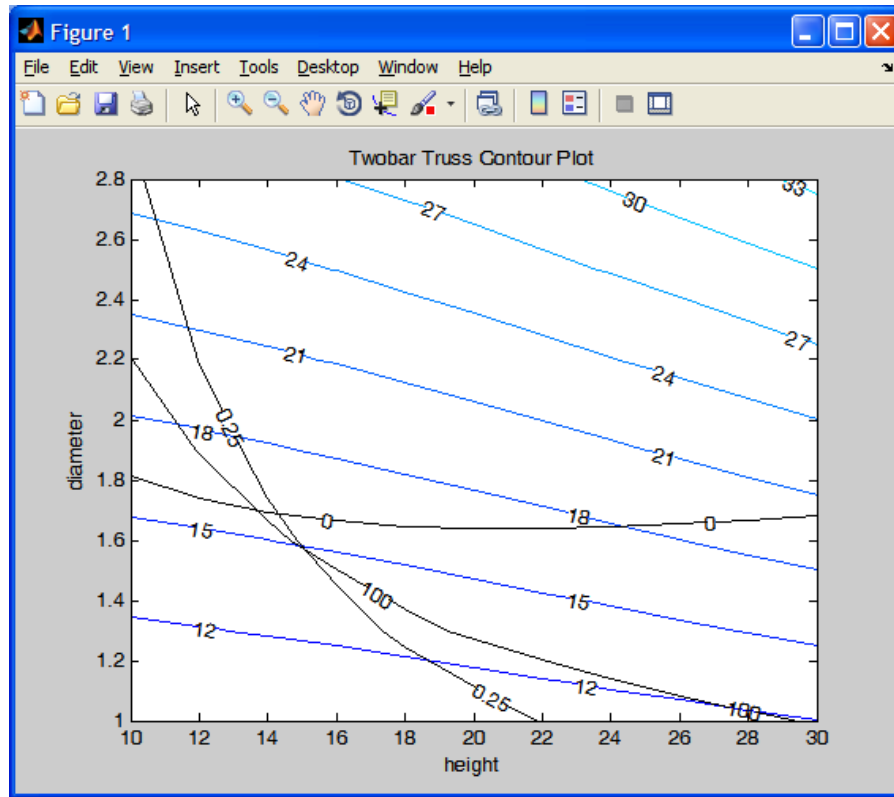


```
1 - width = 60.; thik = 0.15; dens = 0.3;
2 - modu = 30000.; load = 66.;
3 - [hght,diam] = meshgrid(10:2:30,1:.3:3);
4 - leng = sqrt((width/2.)^2 + hght.^2);
5 - area = pi.*diam*thik;
6 - iovera = (diam.^2 + thik^2)/8.;
7
8 - wght = 2*dens.*leng.*area;
9 - strs = load.*leng./(2.*area.*hght);
10 - buck = (pi^2*modu.*iovera ./ (leng.^2));
11 - defl = load.*leng.^3 ./ (2*modu.*area.*hght.^2);
12
13 - clf reset;
14 - [C,h] = contour(hght,diam,wght,12:3:33);
15 - clabel(C,h,'Labelspacing',250);
16 - title('Twobar Truss Contour Plot');
17 - xlabel('height');
18 - ylabel('diameter');
19 - hold on;
20 - [C,h] = contour(hght,diam,strs,[100,100],'k');
21 - clabel(C,h,'Labelspacing',250);
22 - [C,h] = contour(hght,diam,defl,[0.25,0.25],'k');
23 - clabel(C,h,'Labelspacing',250);
24 - [C,h] = contour(hght,diam,(strs-buck),[0,0],'k');
25 - clabel(C,h,'Labelspacing',250);
26 - hold off;
27 - figure;
28 - surf(hght,diam,wght);
```

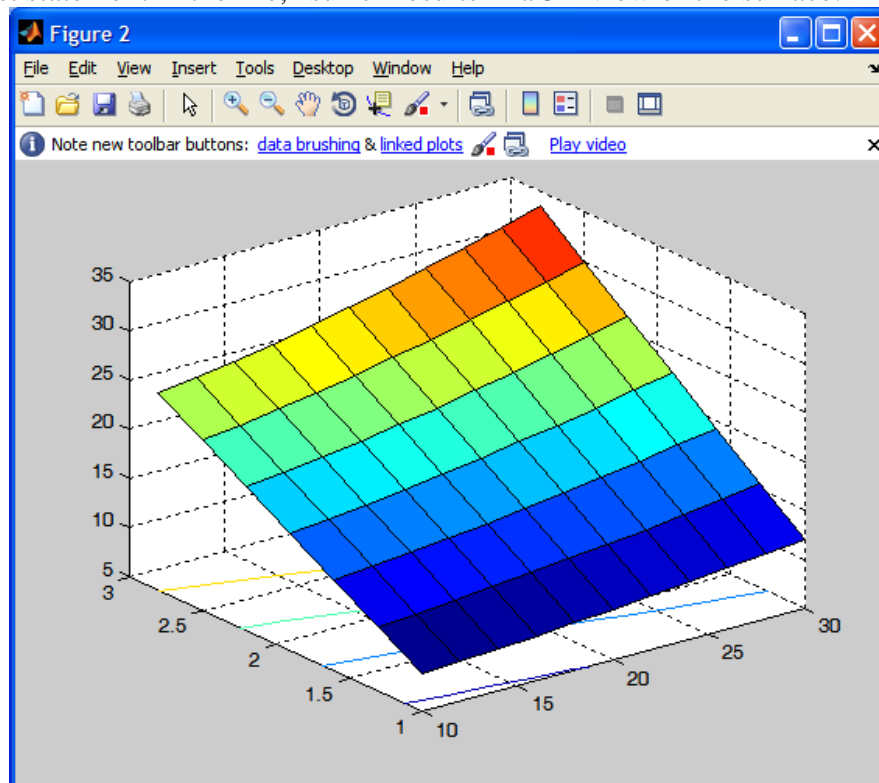
The first statement (“meshgrid”) produces a mesh and makes hght and diam arrays. Thereafter, any function that depends on these two variables will also be an array. Note that we need to use array rather than matrix notation (“.*” instead of “*”, “./” instead of “/”, and “.^” instead of “^” whenever these arrays are involved.) This notation tells Matlab to do calculations element by element.

In the statements which produce the contours, the code “[12:3:34]” indicates the levels we wish to have for the contours; in this case, they will go from 12 pounds to 34 pounds in three pound increments. Alternatively we could just give one number (without brackets), which would indicate the number of contours, and Matlab would decide on the levels.

The resulting plot is given below.



The very last statement in the file, “surf” results in a 3D view of the surface:



Ref: *An Engineer's Guide to Matlab*, 2nd Edition, Pearson Prentice Hall.