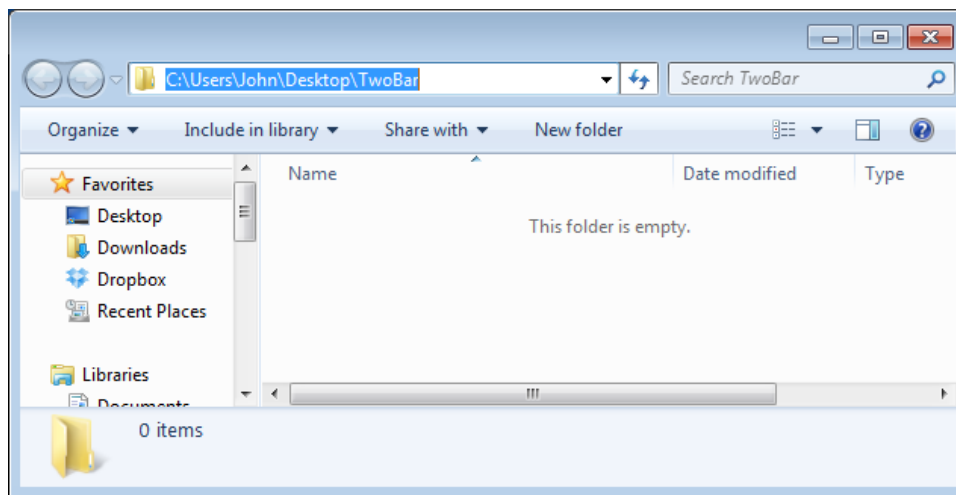


Solving a Two Bar Truss Problem Using APM MATLAB

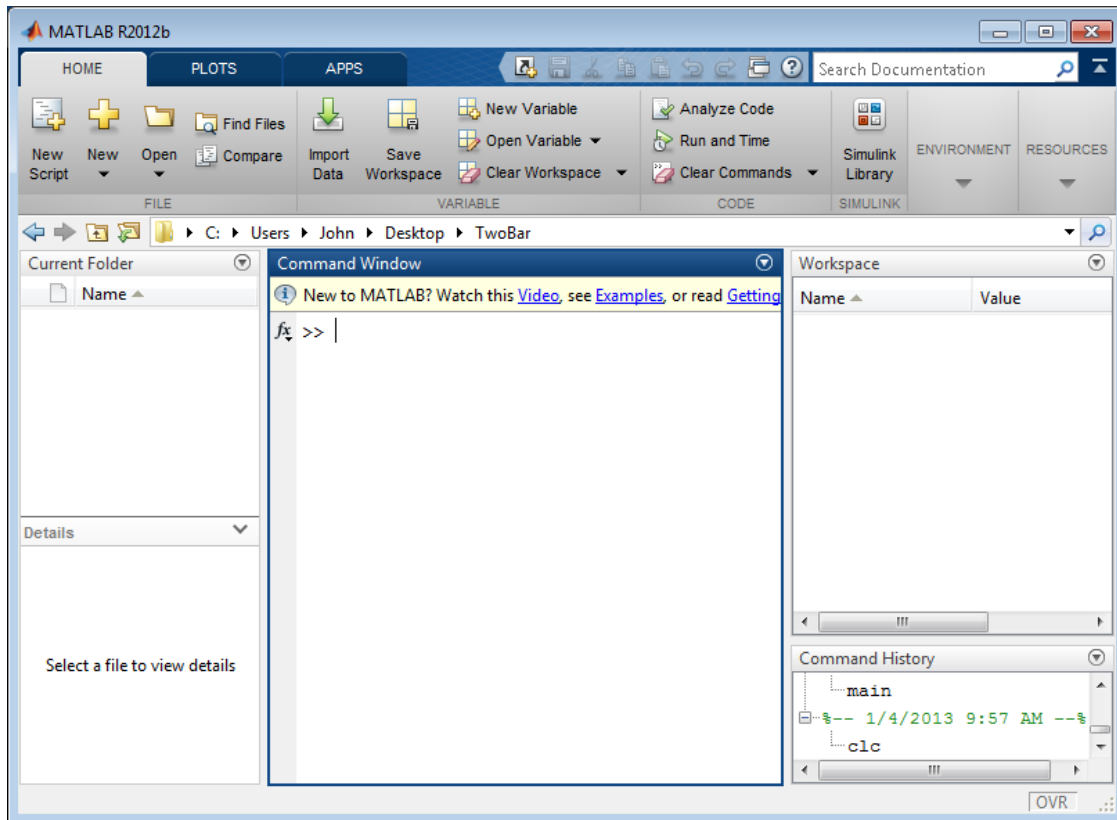
This tutorial is a step-by-step procedure for solving optimization problems with the APMonitor Toolbox for MATLAB. The tutorial assumes no prior experience with either APMonitor or MATLAB so many of the steps can be skipped by experienced users. The tutorial covers the solution to a two bar truss optimization problem with additional details [here](http://apmonitor.com/me575/index.php/Main/TwoBarTruss):

<http://apmonitor.com/me575/index.php/Main/TwoBarTruss>

1. Create a new folder or directory for this problem. This problem is demonstrated with Windows OS. Any other platform that can run MATLAB such as Linux or Mac OS is also acceptable. Create a new folder **TwoBar** on the desktop.



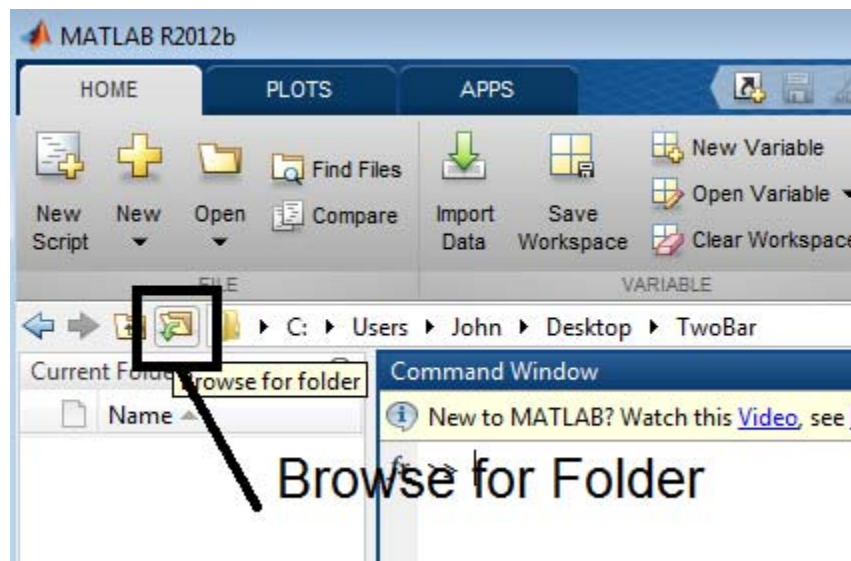
2. Open MATLAB and take note of the different workspace sections including the Command Window, Current Folder, Workspace, Command History, etc. This tutorial uses MATLAB R2012b but other versions have also been tested and should also work. Some old versions of MATLAB (pre-2005) have some known issues with web-browser actions such as the **apm_web** function but this is only for web-browser display of results and is not critical for this application.



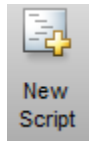
3. Browse to the TwoBar folder by either using the command

`cd C:\Users\{Your Username}\Desktop\Twobar`

in the Command Window or by using the dialog box to select the directory



4. Create a MATLAB script file (extension “.m”). A new script file can be created with the “New Script” button or by issuing the command at the Command Window:



or **edit MyNewScript.m**

For this example the script is named **MyNewScript.m** but any script name is suitable.

5. Include the following commands in your script file **MyNewScript.m**:

```
% clear everything
clear all; close all; clc

% load the APM MATLAB library
addpath('apm')

% select the server
server = 'http://byu.apmonitor.com';

% give the application a name
app = 'twobar';

% clear any previous applications by that name
apm(server,app,'clear all');

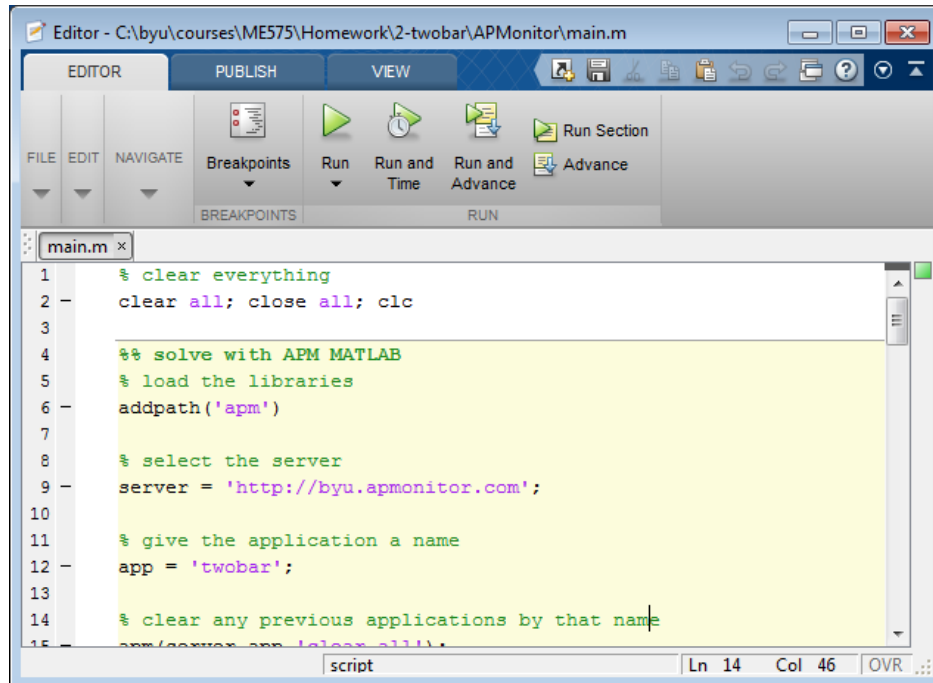
% load the model file
apm_load(server,app,'twobar.apm');

% solve the optimization problem
apm(server,app,'solve');

% retrieve the solution
y = apm_sol(server,app);
z = y.x;

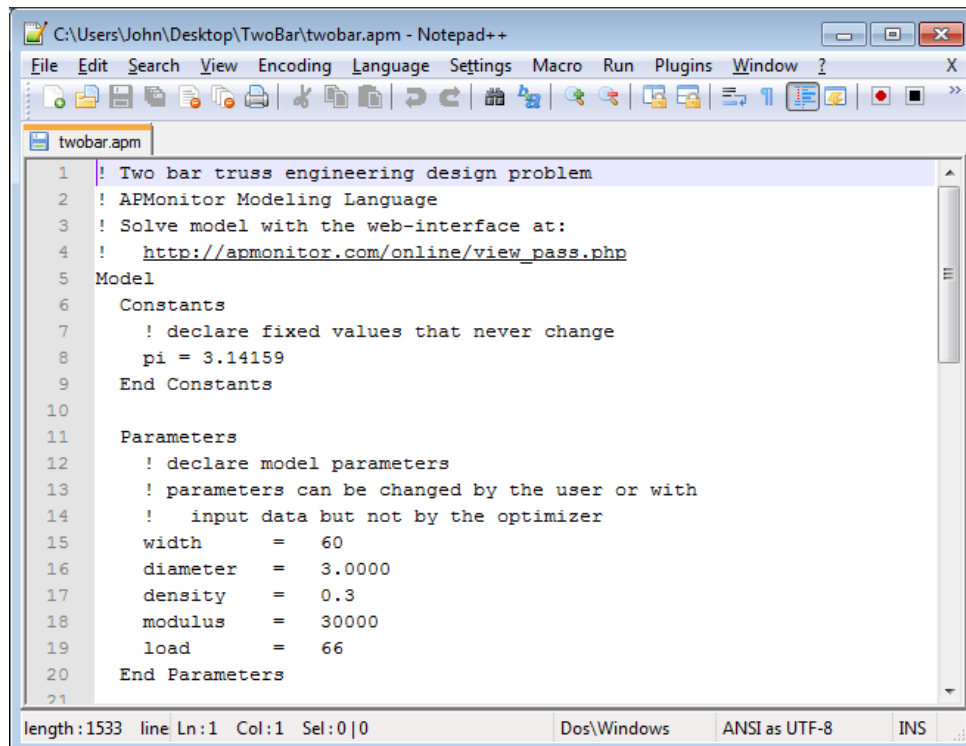
% display the solution
disp(['Height:    ' num2str(z.height)])
disp(['Diameter:  ' num2str(z.diameter)])
disp(['Weight:    ' num2str(z.weight)])
```

The MATLAB script is a text file and can be edited with a text editor like Notepad (Windows) or Emacs (Linux) or within the MATLAB editor. Note that the % sign indicates a comment character and comments are indicated by green text in MATLAB. If MATLAB is the default editor of m files, the MATLAB editor should open automatically with the script contents displayed if the file is opened.



```
1 % clear everything
2 clear all; close all; clc
3
4 %% solve with APM MATLAB
5 % load the libraries
6 addpath('apm')
7
8 % select the server
9 server = 'http://byu.apmonitor.com';
10
11 % give the application a name
12 app = 'twobar';
13
14 % clear any previous applications by that name
15 app('server', app, 'clear', all);
```

6. Create an APMonitor model file (File Extension .apm) named **twobar.apm** that will be used to configure the optimization problem. The model file is a collection of Constants, Parameters, Variables, and Equations that relate the Design Variables to the Objective Function. The APM model file can be modified by any text editor. [Notepad++ \(a free text editor\)](#) is recommended instead of the Windows default (Notepad application) if you don't have another text editor of choice.



```
1 ! Two bar truss engineering design problem
2 ! APMonitor Modeling Language
3 ! Solve model with the web-interface at:
4 ! http://apmonitor.com/online/view_pass.php
5 Model
6   Constants
7     ! declare fixed values that never change
8     pi = 3.14159
9   End Constants
10
11   Parameters
12     ! declare model parameters
13     ! parameters can be changed by the user or with
14     ! input data but not by the optimizer
15     width      = 60
16     diameter    = 3.0000
17     density     = 0.3
18     modulus     = 30000
19     load        = 66
20   End Parameters
21
```

```
! Two bar truss engineering design problem
! APMonitor Modeling Language
! Solve model with the web-interface at:
! http://apmonitor.com/online/view\_pass.php
```

Model

Constants

```
! declare fixed values that never change
pi      = 3.14159
```

End Constants

Parameters

```
! declare model parameters
! parameters can be changed by the user or with
! input data but not by the optimizer
width    = 60
diameter = 3.0000
density  = 0.3
modulus   = 30000
load     = 66
```

End Parameters

Variables

```
! declare variables and initial guesses
! variables can be changed by the optimizer
height   = 30.00, >= 10.0, <= 50.0
thickness = 0.15, >= 0.05, <= 0.20
weight
```

End Variables

Intermediates

```
! intermediate variables are explicitly determined
! with equality constraints
leng      = sqrt((width/2)^2 + height^2)
area      = pi * diameter * thickness
iovera    = (diameter^2 + thickness^2) / 8

stress    = load * leng / (2*area*height)
buckling  = pi^2 * modulus * iovera / (leng^2)
deflection = load * leng^3 / (2 * modulus * area * height^2)
```

End Intermediates

Equations

```
! objective: minimize the weight
minimize weight

! equality constraints
weight    = 2 * density * area * leng

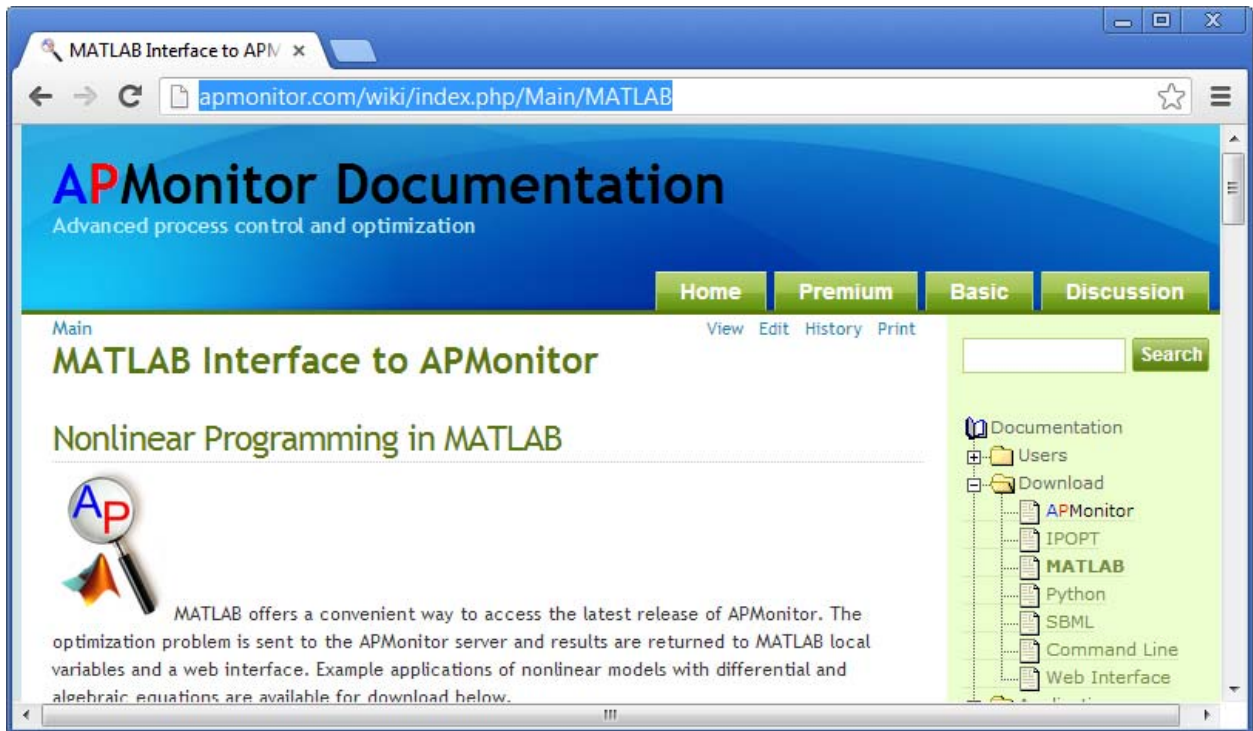
! inequality constraints
weight < 24
stress < 90
stress < buckling
deflection < 0.25
```

End Equations

End Model

7. A final requirement is to obtain the APM toolbox libraries from the APMonitor.com website. These are a collection of **.m** file functions that allow a MATLAB user to use the APM models to solve simulation and optimization problems. To obtain the APM library browse to:

<http://apmonitor.com/wiki/index.php/Main/MATLAB>



Download APM MATLAB Libraries

The latest APM MATLAB libraries are attached below. Functionality has been tested with the latest release of MATLAB.

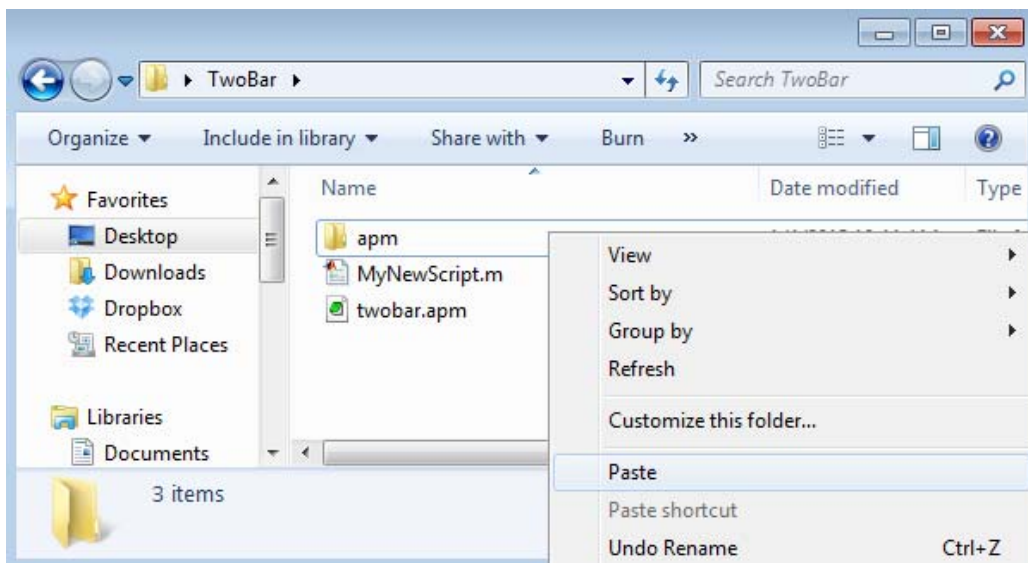
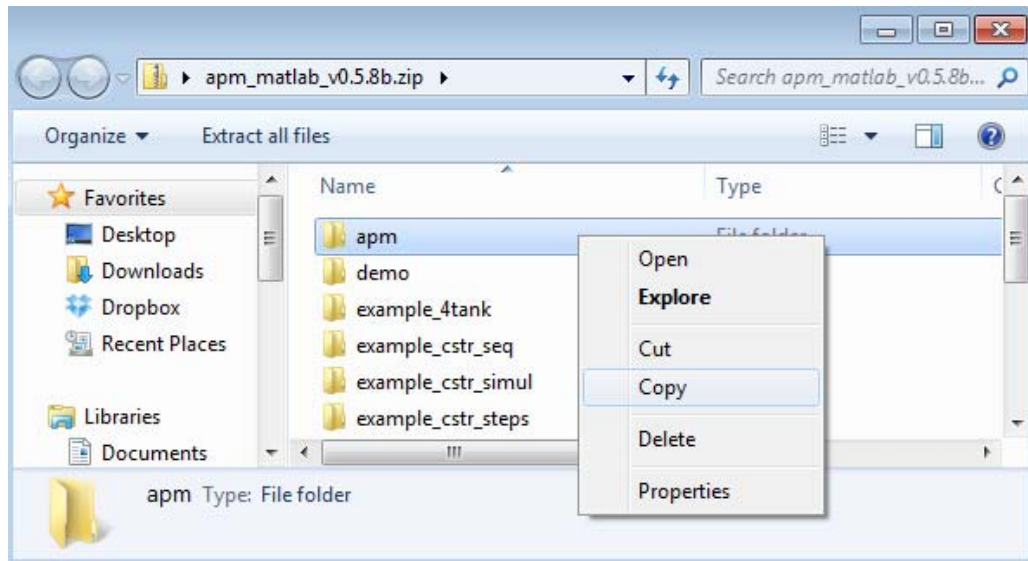


Download the zip archive

[APM MATLAB \(version 0.5.8b\)](#) - Released 26 Nov 2012

The zipped archives contain a script files such as *apm.m*. To use the APM MATLAB functions, copy the script files into the active directory or add the path with the *addpath* command.

Open the zipped archive and copy the **apm** folder into the **TwoBar** folder on your desktop.



8. The folder now contains all of the files necessary to solve the optimization problem. Once the **apm** folder is copied into the **TwoBar** folder, the following files should appear:

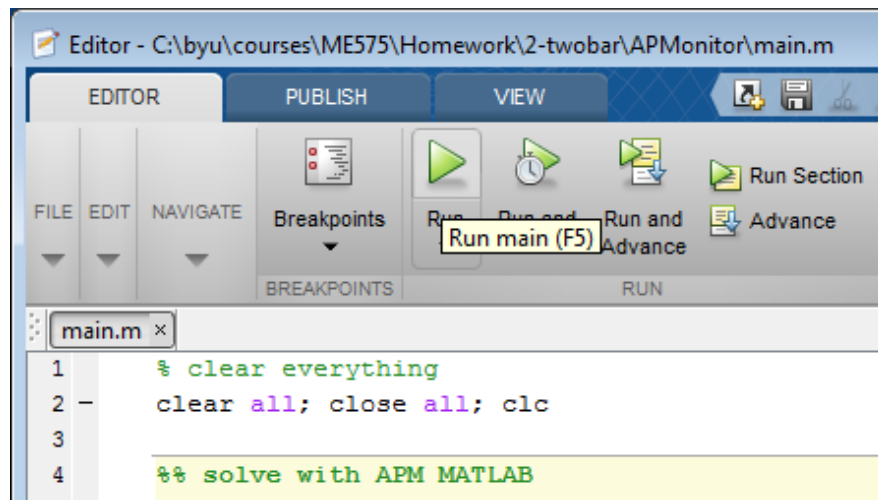
- **apm** – folder that allows users to work with the APM MATLAB toolbox
- **MyNewScript.m** – MATLAB driver script file for solving the Two Bar problem
- **twobar.apm** – Two Bar problem in the APMonitor Modeling Language

Name	Date modified	Type
apm	1/4/2013 10:44 AM	File folder
MyNewScript.m	1/4/2013 10:44 AM	MATLAB Code
twobar.apm	1/4/2013 10:44 AM	APM File

9. Solve the optimization problem by running the MATLAB script **MyNewScript.m**. The script can be run by clicking the **Run button**, Pressing **F5** with the MATLAB editor in focus, or by issuing the statement from the Command Window:

>> MyNewScript

(note, don't include the >> or the .m file extension)



10. The results will be accessible through a solution.csv file downloaded to your run directory or displayed in the Command Window as:

```
Height:    29.9999
Diameter:  3
Weight:    13.2
```

11. A web-interface can also be used to view the results by issuing the statement at the Command Window. The semi-colon “;” at the end of the function call is used to suppress any output from the command and is optional.

>> apm_web_var(server,app);

Name	Lower	Value	Upper
pi	---	3.1416E+00	---
leng	---	4.2426E+01	---
area	---	5.1855E-01	---
iovera	---	1.1254E+00	---
stress	---	9.0000E+01	---
buckling	---	1.8512E+02	---
deflection	---	1.8000E-01	---
ss.width	---	6.0000E+01	---
ss.diameter	---	3.0000E+00	---
ss.density	---	3.0000E-01	---
ss.modulus	---	3.0000E+04	---
ss.load	---	6.6000E+01	---
ss.height	1.0000E+01	3.0000E+01	5.0000E+01
ss.thickness	5.0000E-02	5.5019E-02	2.0000E-01
ss.weight	---	1.3200E+01	---

12. Next is code for contour plots that can be added to the end of the **MyNewScript.m** file:

```
% constants
pi = 3.14159;
dens = 0.3;
modu = 30000;
load = 66.0;

% analysis variables
wdth = 60;
thik = 0.15;

% design variables at mesh points
[hght,diam] = meshgrid(10:2:30,1:.3:3);

% equations
leng = ((wdth/2)^2 + hght.^2).^0.5;
area = pi * diam .* thik;
iovera = (diam.^2 + thik^2)/8;
wght = 2 * dens * leng .* area;
strs = load * leng ./ (2 * area .* hght);
buck = pi^2 * modu * iovera ./ (leng.^2);
defl = load * leng.^3 ./ (2*modu * area .* hght.^2);

figure(1)
contour(hght,diam,wght,12:3:33,'k');
title('Two Bar Truss Contour Plot');
xlabel('Height');
ylabel('Diameter');
hold on;
% solid lines to show constraint boundaries
contour(hght,diam,strs,[100,100],'g-','LineWidth',3);
contour(hght,diam,defl,[0.25,0.25],'r-','LineWidth',3);
contour(hght,diam,(strs-buck),[0.0,0.0],'b-','LineWidth',3);
% show a legend
legend('Weight','Stress<100','Deflection<0.25','Stress-Buckling<0')

figure(2)
[C,h] = contour(hght,diam,wght,12:3:33,'k. ');
clabel(C,h,'Labelspacing',250);
title('Two Bar Truss Contour Plot');
xlabel('Height');
ylabel('Diameter');
hold on;
[C,h] = contour(hght,diam,strs,[85,90,95,100],'g.-');
clabel(C,h,'Labelspacing',250);
[C,h] = contour(hght,diam,defl,[0.19,0.21,0.23,0.25],'r:');
clabel(C,h,'Labelspacing',250);
[C,h] = contour(hght,diam,(strs-buck),[-30.0,-20.0,-10.0,0.0],'b--');
clabel(C,h,'Labelspacing',250);
% show a legend
legend('Weight','Stress<100','Deflection<0.25','Stress-Buckling<0')
% solid lines to show constraint boundaries
contour(hght,diam,strs,[100,100],'g-','LineWidth',3);
contour(hght,diam,defl,[0.25,0.25],'r-','LineWidth',3);
contour(hght,diam,(strs-buck),[0.0,0.0],'b-','LineWidth',3);

figure(3)
surfc(hght,diam,wght);
```

13. The statement **meshgrid** produces a mesh and makes **hght** and **diam** arrays.

```
% design variables at mesh points  
[hght,diam] = meshgrid(10:2:30,1:.3:3);
```

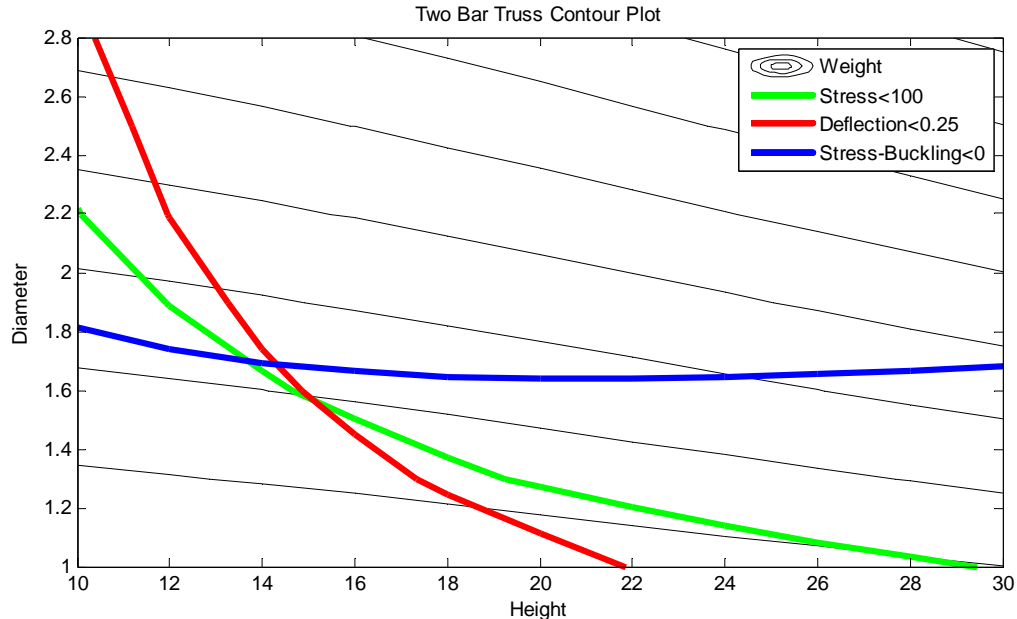
Thereafter, any function that depends on these two variables will also be an array. Note that we need to use array rather than matrix notation (“.” instead of “*”, “./” instead of “/”, and “.^” instead of “^” whenever these arrays are involved.) This notation tells MATLAB to do calculations element by element instead of regular matrix multiplication.

In the statements which produce the contours, the code **[12:3:34]** indicates the levels we wish to have for the contours

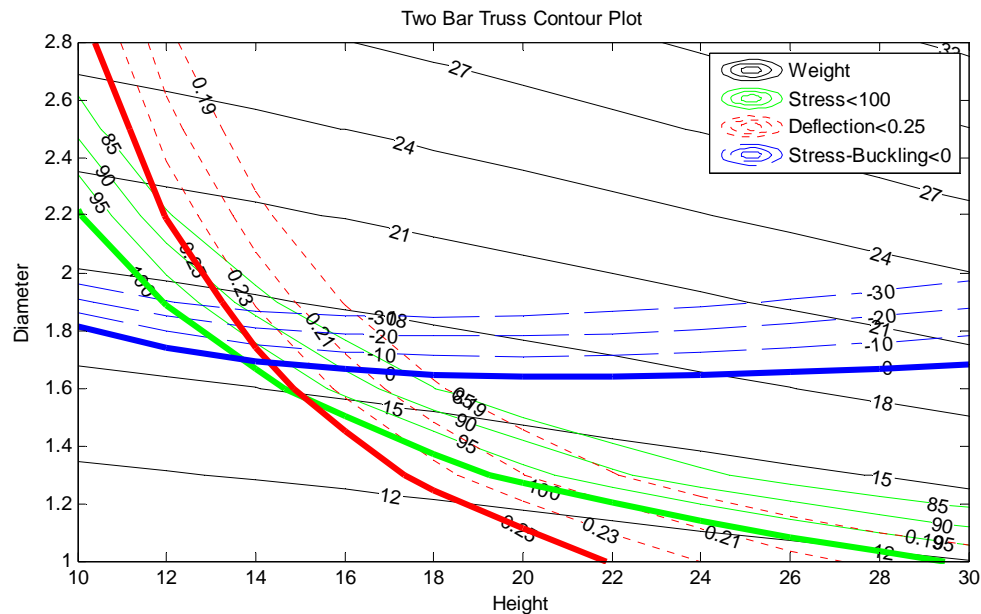
```
figure(1)  
contour(hght,diam,wght,12:3:33,'k');
```

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.

14. The resulting **Figure 1** contour plot is given below. The plot file can be saved in a variety of file formats including JPEG, PNG, EMF, EPS, etc. by using the **Save As** selection under the **File** toolbar menu or by using the **saveas** MATLAB command.

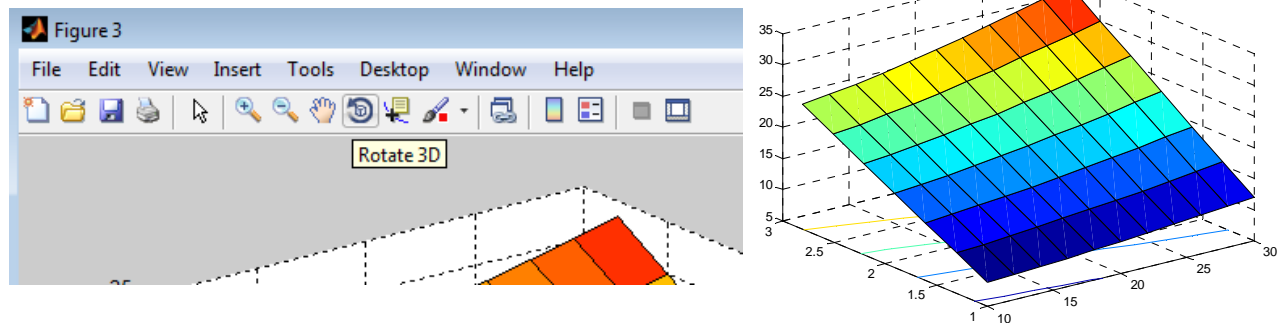


15. **Figure 2** is a more detailed contour plot that shows specific values and the region of feasible designs. Annotations and other mark-up items can be added within MATLAB or afterwards in an image editor.



16. The very last statement in the file, **surf** results in **Figure 3** for a 3D view of the surface. The 3D plot can be rotated after selecting the **Rotate 3D** button at the top. The figure can be enhanced with the **xlabel**, **ylabel**, **zlabel**, and **legend** commands such as

```
xlabel('Height');
ylabel('Diameter');
```



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