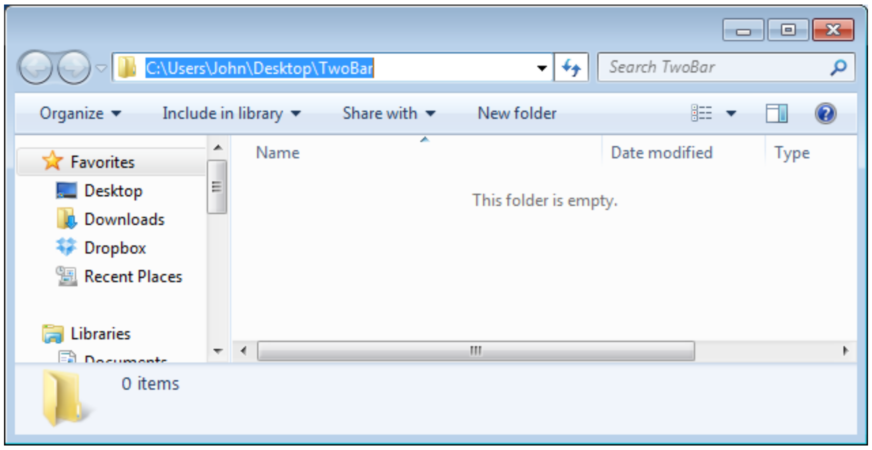
Solving a Two Bar Truss Problem Using APM Python

This tutorial is a step-by-step procedure for solving optimization problems with the APMonitor Toolbox for Python. It has been tested with Python 2.7 with Numpy and Matlibplot as additional packages. The tutorial assumes no prior experience with either APMonitor or Python 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:

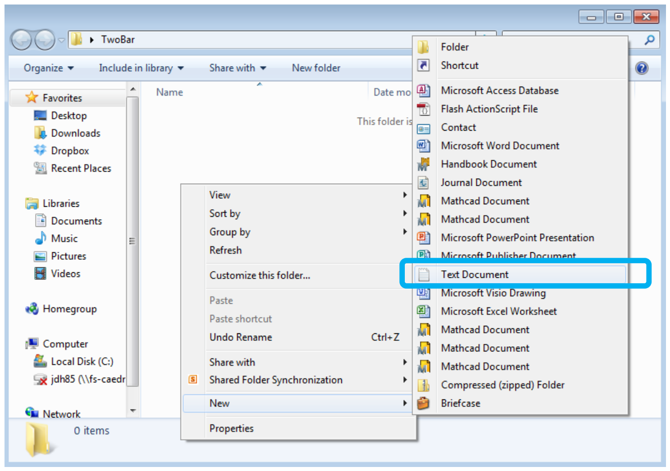
http://apmonitor.com/me575/index.php/Main/TwoBarTruss

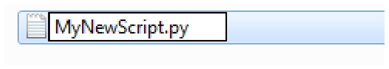
1. Create a new folder or directory for this problem. This problem is demonstrated with Windows OS. Any other platform that can run Python 2.7 such as Linux or Mac OS is also acceptable. Create a new folder **TwoBar** on the desktop. It will be located at:

   C:\Users\**{Your Username}**\Desktop\Twobar

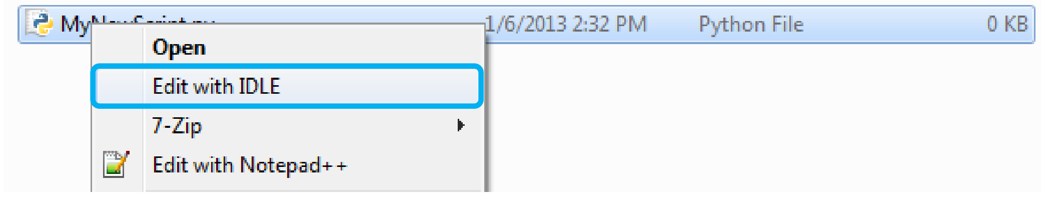


2. Create a Python script file (extension ".py"). A new script file can be created by right-clicking in the new TwoBar folder and selecting **New**…**Text Document**. In this case the script name of **MyNewScript.py** is given, but any script name can be chosen.





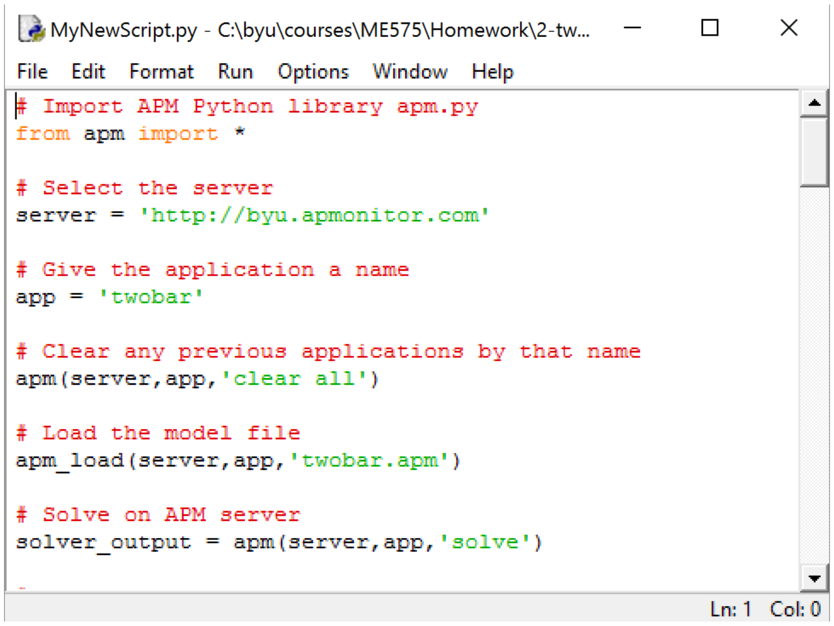
3. Edit the Python script with IDLE, a Python script editor and development environment installed with Python. To edit with IDLE, right-click **MyNewScript.py** and select “Edit with IDLE”.



4. Include the following commands in your script file **MyNewScript.py**:

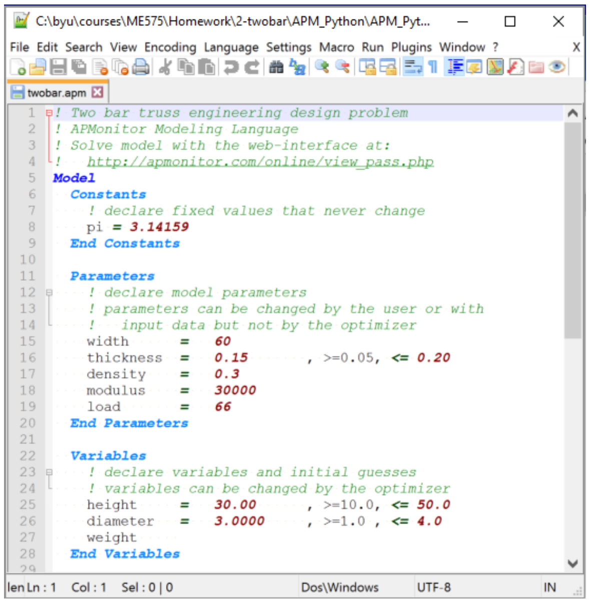
```
# Import APM Python library apm.py
from apm import *

# 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 on APM server
solver_output = apm(server,app,'solve')

# Display solver output
print(solver_output)

# Retrieve results
sol = apm_sol(server,app)

print ('')
print ('--- Results of the Optimization Problem ---')
print ('Height: ' + str(sol['height']))
print ('Diameter: ' + str(sol['diameter']))
print ('Weight: ' + str(sol['weight']))

print ('')
print ('--- All available variables ---')
print (sol.keys())

# Display Results in Web Viewer
url = apm_web_var(server,app)
```

The Python script is a text file and can be edited with a text editor like Notepad++, Emacs, or within the IDLE editor. Note that the # sign indicates a comment character and comments are indicated by gray text in the above script or as red in the IDLE editor.


MyNewScript.py - C:\byu\courses\ME575\Homework\2-tw...
File Edit Format Run Options Window Help
Ln: 1 Col: 0

```python
# Import APM Python library apm.py
from apm import *

# 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 on APM server
solver_output = apm(server,app,'solve')
```

5. 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++ (http://notepad-plus-plus.org) is recommended instead of the Windows default (Notepad application) if you don't have another text editor of choice.

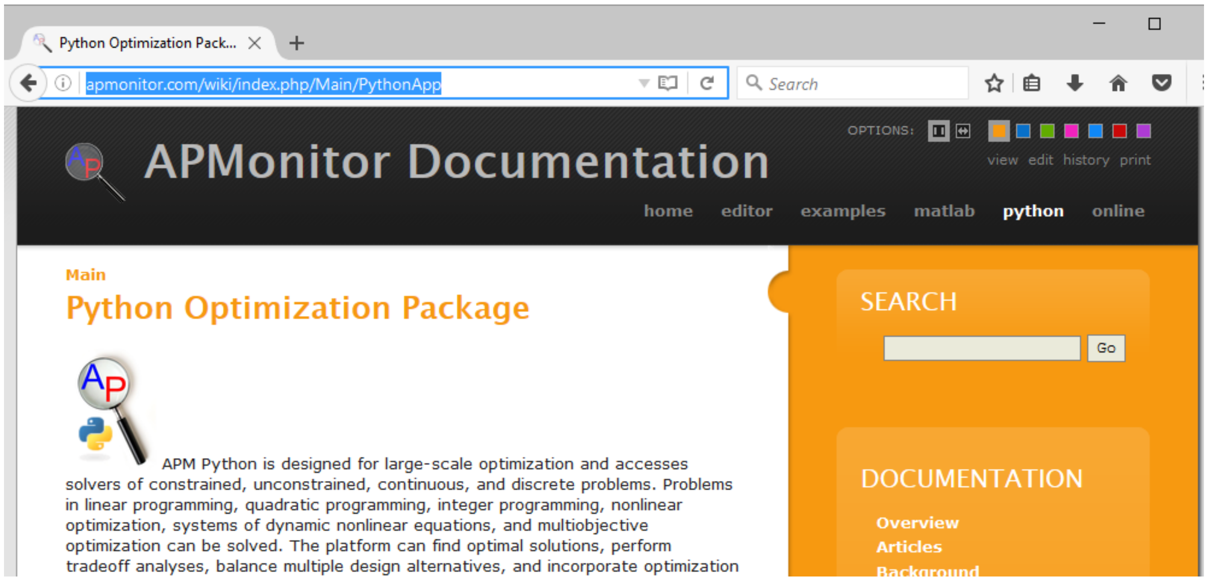

C:\byu\courses\ME575\Homework\2-twobar\APM_Python\APM_Pyt...
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
twobar.apm
len Ln : 1 Col : 1 Sel : 0 | 0 Dos\Windows UTF-8 IN

```
! 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
    thickness  =   0.15        , >=0.05, <= 0.20
    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
    diameter   =   3.0000      , >=1.0 , <= 4.0
    weight
  End Variables
```

```
! 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
    thickness   = 0.15
    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
    diameter    = 3.000, >= 1.00, <= 4.00
    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 < 100
    stress < buckling
    deflection < 0.25
  End Equations
End Model
```

6. A final requirement is to obtain the APM package libraries from the APMonitor.com web-site. These are a collection of .py file functions that allow a PYTHON user to use the APM models to solve simulation and optimization problems. To obtain the APM library browse to:

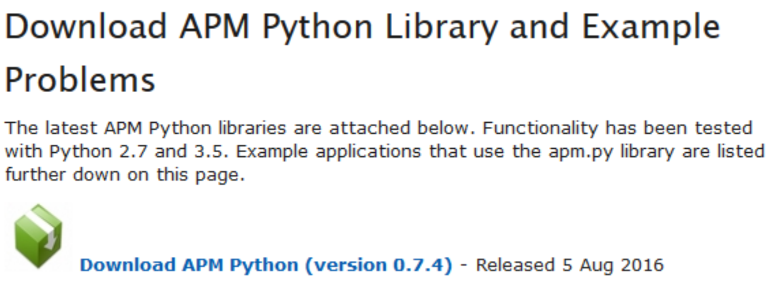
   http://apmonitor.com/wiki/index.php/Main/PythonApp


Python Optimization Pack...
apmonitor.com/wiki/index.php/Main/PythonApp
Search
OPTIONS:
view edit history print
APMonitor Documentation
home editor examples matlab python online
SEARCH
Go
DOCUMENTATION
Overview
Articles
Background

Main

Python Optimization Package

APM Python is designed for large-scale optimization and accesses solvers of constrained, unconstrained, continuous, and discrete problems. Problems in linear programming, quadratic programming, integer programming, nonlinear optimization, systems of dynamic nonlinear equations, and multiobjective optimization can be solved. The platform can find optimal solutions, perform tradeoff analyses, balance multiple design alternatives, and incorporate optimization

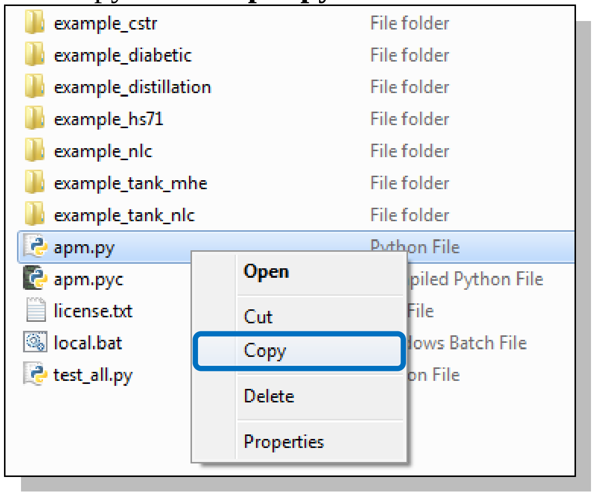


Download APM Python Library and Example Problems

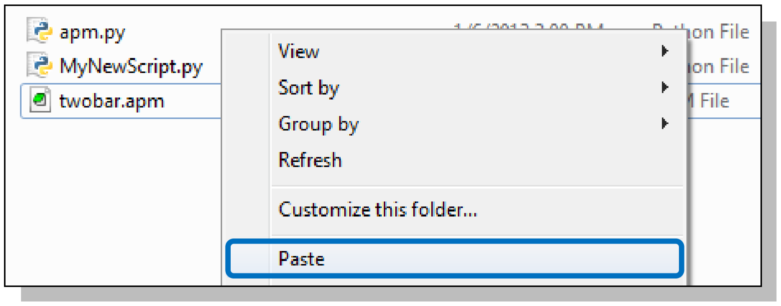
The latest APM Python libraries are attached below. Functionality has been tested with Python 2.7 and 3.5. Example applications that use the apm.py library are listed further down on this page.

Download APM Python (version 0.7.4) - Released 5 Aug 2016

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

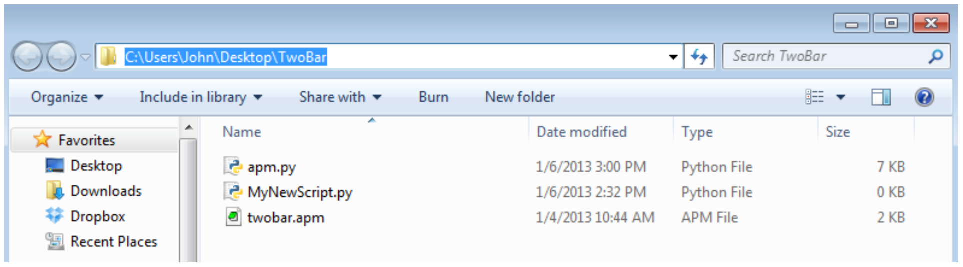


example_cstr	File folder
example_diabetic	File folder
example_distillation	File folder
example_hs71	File folder
example_nlc	File folder
example_tank_mhe	File folder
example_tank_nlc	File folder
apm.py	Python File
apm.pyc	piled Python File
license.txt	File
local.bat	lows Batch File
test_all.py	on File



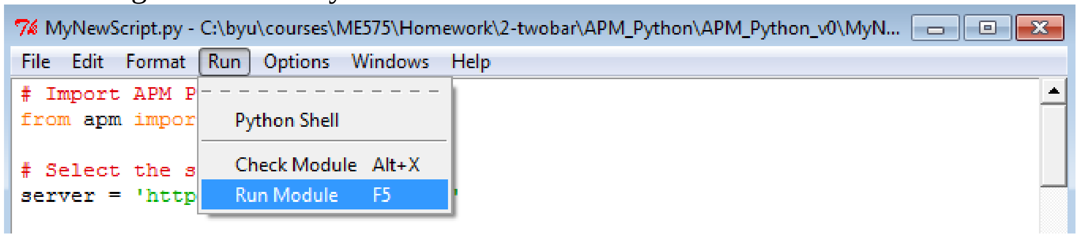
7. The folder now contains all of the files necessary to solve the optimization problem. Once the **apm.py** file is copied into the **TwoBar** folder, the following files should appear:

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



Name	Date modified	Type	Size
apm.py	1/6/2013 3:00 PM	Python File	7 KB
MyNewScript.py	1/6/2013 2:32 PM	Python File	0 KB
twobar.apm	1/4/2013 10:44 AM	APM File	2 KB

8. Solve the optimization problem by running the Python script **MyNewScript.py**. The script can be run from the IDLE editor by clicking **Run…Run Module** from the toolbar or Pressing **F5** with the Python editor IDLE in focus


MyNewScript.py - C:\byu\courses\ME575\Homework\2-twobar\APM_Python\APM_Python_v0\MyN...
File
Edit
Format
Run
Options
Windows
Help
Python Shell
Check Module Alt+X
Run Module F5

```
# Import APM P
from apm impor

# Select the s
server = 'http
```

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

10. A web-interface can also be used to view the results by issuing the statement at the Command Window.

    >> **apm_web_var(server,app);**

Name	Lower	Value	Upper
pi	---	3.1416E+00	---
leng	---	3.3926E+01	---
area	---	7.8520E-01	---

iovera	---	3.4986E-01	---
stress	---	9.0000E+01	---
buckling	---	9.0000E+01	---
deflection	---	2.1795E-01	---
width	---	6.0000E+01	---
thickness	5.0000E-02	1.5000E-01	2.0000E-01
density	---	3.0000E-01	---
modulus	---	3.0000E+04	---
load	---	6.6000E+01	---
height	1.0000E+01	1.4215E+01	5.0000E+01
diameter	1.0000E+00	1.6906E+00	4.0000E+00
weight	---	1.5868E+01	---
slk_3	0.0000E+00	8.1317E+00	---
slk_4	0.0000E+00	3.2617E+00	---
slk_5	0.0000E+00	0.0000E+00	---
slk_6	0.0000E+00	0.0000E+00	---

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

```
## Generate a contour plot
# Import some other libraries that we'll need
# matplotlib and numpy packages must also be installed
import matplotlib
import numpy as np
import matplotlib.pyplot as plt

# Constants
pi = 3.14159
dens = 0.3
modu = 30000.0
load = 66.0

# Analysis variables
wdth = 60.0
thik = 0.15

# Design variables at mesh points
x = np.arange(10.0, 30.0, 2.0)
y = np.arange(1.0, 3.0, 0.3)
hght, diam = np.meshgrid(x, y)

# Equations and Constraints
leng = ((wdth/2.0)**2.0 + hght**2)**0.5
area = pi * diam * thik
iovera = (diam**2.0 + thik**2.0)/8.0
wght = 2.0 * dens * leng * area
strs = load * leng / (2.0 * area * hght)
buck = pi**2.0 * modu * iovera / (leng**2.0)
defl = load * leng**3.0 / (2.0*modu * area * hght**2.0)

# Create a contour plot
# Visit http://matplotlib.org/examples/pylab_examples/contour_demo.html
#   for more examples and options for contour plots
plt.figure()
# Weight contours
CS = plt.contour(hght, diam, wght)
plt.clabel(CS, inline=1, fontsize=10)
# Stress<100
CS = plt.contour(hght, diam, strs,[100.0],colors='k',linewidths=[4.0])
plt.clabel(CS, inline=1, fontsize=10)
# Deflection<0.25
CS = plt.contour(hght, diam, defl,[0.25],colors='b',linewidths=[4.0])
plt.clabel(CS, inline=1, fontsize=10)
# Stress-Buckling<0
CS = plt.contour(hght, diam, strs-buck,[0.0],colors='r',linewidths=[4.0])
plt.clabel(CS, inline=1, fontsize=10)
# Add some labels
plt.title('Two Bar Optimization Problem')
plt.xlabel('Height')
plt.ylabel('Diameter')
# Save the figure as a PNG
plt.savefig('contour1.png')

# Create a new figure to see more detail
plt.figure()
# Weight contours
CS = plt.contour(hght, diam, wght)
plt.clabel(CS, inline=1, fontsize=10)
# Stress<100
CS = plt.contour(hght, diam, strs,[90.0,100.0],colors='k',linewidths=[0.5, 4.0])
plt.clabel(CS, inline=1, fontsize=10)
# Deflection<0.25
CS = plt.contour(hght, diam, defl,[0.22,0.25],colors='b',linewidths=[0.5, 4.0])
plt.clabel(CS, inline=1, fontsize=10)
# Stress-Buckling<0
CS = plt.contour(hght, diam, strs-buck,[-5.0,0.0],colors='r',linewidths=[0.5, 4.0])
plt.clabel(CS, inline=1, fontsize=10)
# Add some labels
plt.title('Two Bar Optimization Problem')
plt.xlabel('Height')
plt.ylabel('Diameter')
# Save the figure as a PNG
plt.savefig('contour2.png')

# Show the plots
plt.show()
```

12. The statement **np.meshgrid(x, y)** produces a mesh and makes hght and diam into arrays.

```
# Design variables at mesh points
x = np.arange(10.0, 30.0, 2.0)
y = np.arange(1.0, 3.0, 0.3)
hght, diam = np.meshgrid(x, y)
```

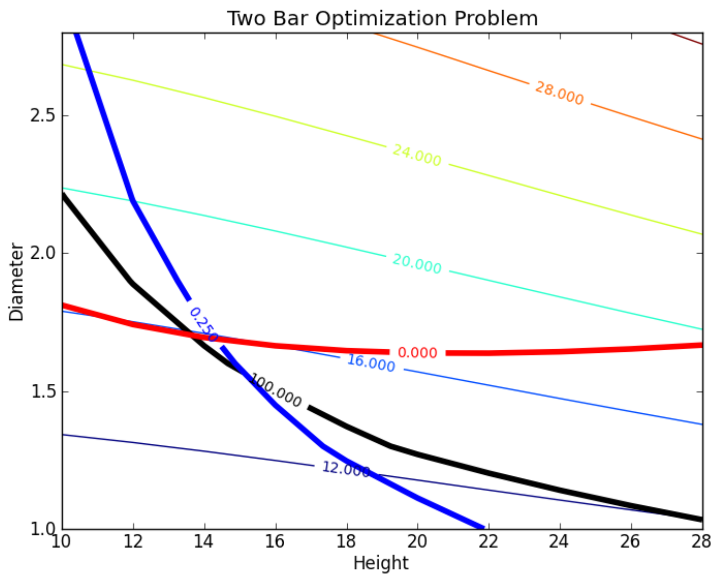
Thereafter, any function that depends on these two variables will also be an array. Note that we use array rather than matrices so that calculations are performed element by element instead of regular matrix operations.

In the statements which produce the contours, the code **[90.0, 100.0]** indicates the levels we wish to have for the contours. The statement colors='k' makes all of the contours the same color (black). Other colors are 'b'=blue, 'r'=red, 'g'=green, etc. The statement linewidths=[0.5, 4.0] adjusts the line widths for each of the contour lines. Modifying the clabel property places labels on the contour line for the values.

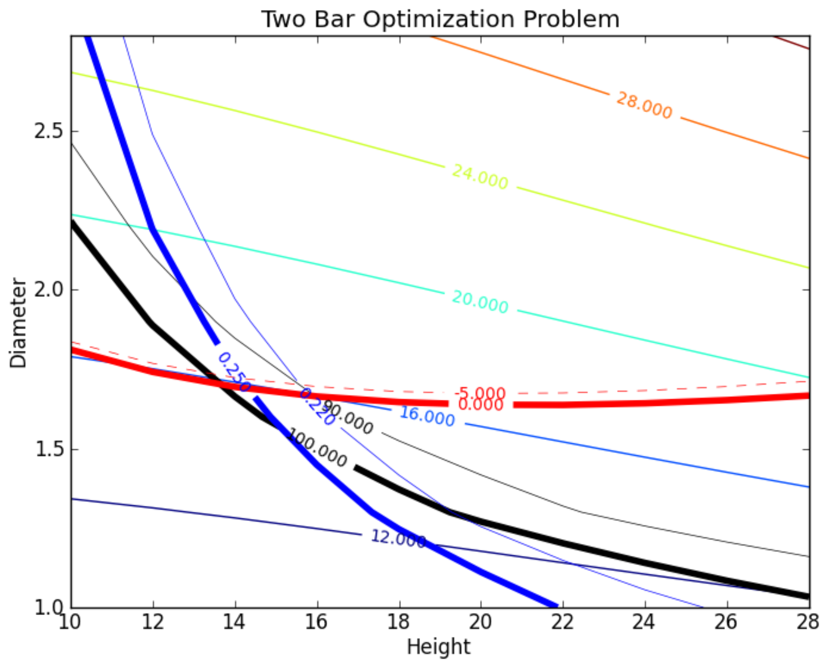
```
CS = plt.contour(hght, diam, strs,[90.0,100.0],colors='k',linewidths=[0.5, 4.0])
plt.clabel(CS, inline=1, fontsize=10)
```

In this case, the stress contour goes from 90 to 100 in just two increments. Alternatively we could just give one number (without brackets), which would indicate the number of contours, and Python would decide on the levels.

13. 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** button on the bottom toolbar menu or by using the **plt.savefig('myFile.png')** Python command.

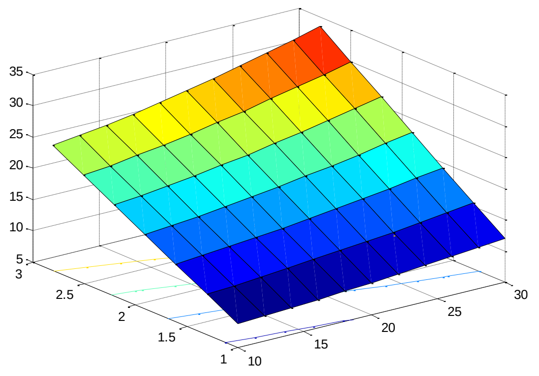


14. **Figure 2** is a more detailed contour plot that shows specific values and the region of feasible designs. Annotations and other mark-up items can be added within Python or afterwards in an image editor.



15. For a 3-D plot of the objective function, the MatPlotLib package can be used. Other 3-D plotting capabilities are also available in Python. The 3-D trending is not included in this tutorial.

http://matplotlib.org/examples/mplot3d/surface3d_demo.html


35
30
25
20
15
10
5
3
2.5
2
1.5
1
10
15
20
25
30