Chapter 3: Functions and Files

Topics Covered:

- Built-In Functions
  - Mathematical Functions
- User-Defined Functions
  - Function Files
  - Anonymous Functions
- Function Functions
  - Function Handles
- Working with Data Files

Built-In Functions:

Table 3.1–1 Some common mathematical functions

Exponential	
`exp(x)`	Exponential; e^x.
`sqrt(x)`	Square root; $\sqrt{x}$.
Logarithmic	
`log(x)`	Natural logarithm; $\ln x$.
`log10(x)`	Common (base-10) logarithm; $\log x = \log_{10} x$.
Complex	
`abs(x)`	Absolute value; x.
`angle(x)`	Angle of a complex number x.
`conj(x)`	Complex conjugate.
`imag(x)`	Imaginary part of a complex number x.
`real(x)`	Real part of a complex number x.
Numeric	
`ceil(x)`	Round to the nearest integer toward ∞.
`fix(x)`	Round to the nearest integer toward zero.
`floor(x)`	Round to the nearest integer toward $-\infty$.
`round(x)`	Round toward the nearest integer.
`sign(x)`	Signum function: $+1$ if $x > 0$; 0 if $x = 0$; -1 if $x < 0$.

Built-In Functions:

Table 3.1–2 Trigonometric functions

Trigonometric*	
`cos(x)`	Cosine; $\cos x$.
`cot(x)`	Cotangent; $\cot x$.
`csc(x)`	Cosecant; $\csc x$.
`sec(x)`	Secant; $\sec x$.
`sin(x)`	Sine; $\sin x$.
`tan(x)`	Tangent; $\tan x$.
Inverse trigonometric†	
`acos(x)`	Inverse cosine; $\arccos x = \cos^{-1} x$.
`acot(x)`	Inverse cotangent; $\operatorname{arccot} x = \cot^{-1} x$.
`acsc(x)`	Inverse cosecant; $\operatorname{arccsc} x = \csc^{-1} x$.
`asec(x)`	Inverse secant; $\operatorname{arcsec} x = \sec^{-1} x$.
`asin(x)`	Inverse sine; $\arcsin x = \sin^{-1} x$.
`atan(x)`	Inverse tangent; $\arctan x = \tan^{-1} x$.
`atan2(y,x)`	Four-quadrant inverse tangent.

*These functions accept x in radians.

†These functions return a value in radians.

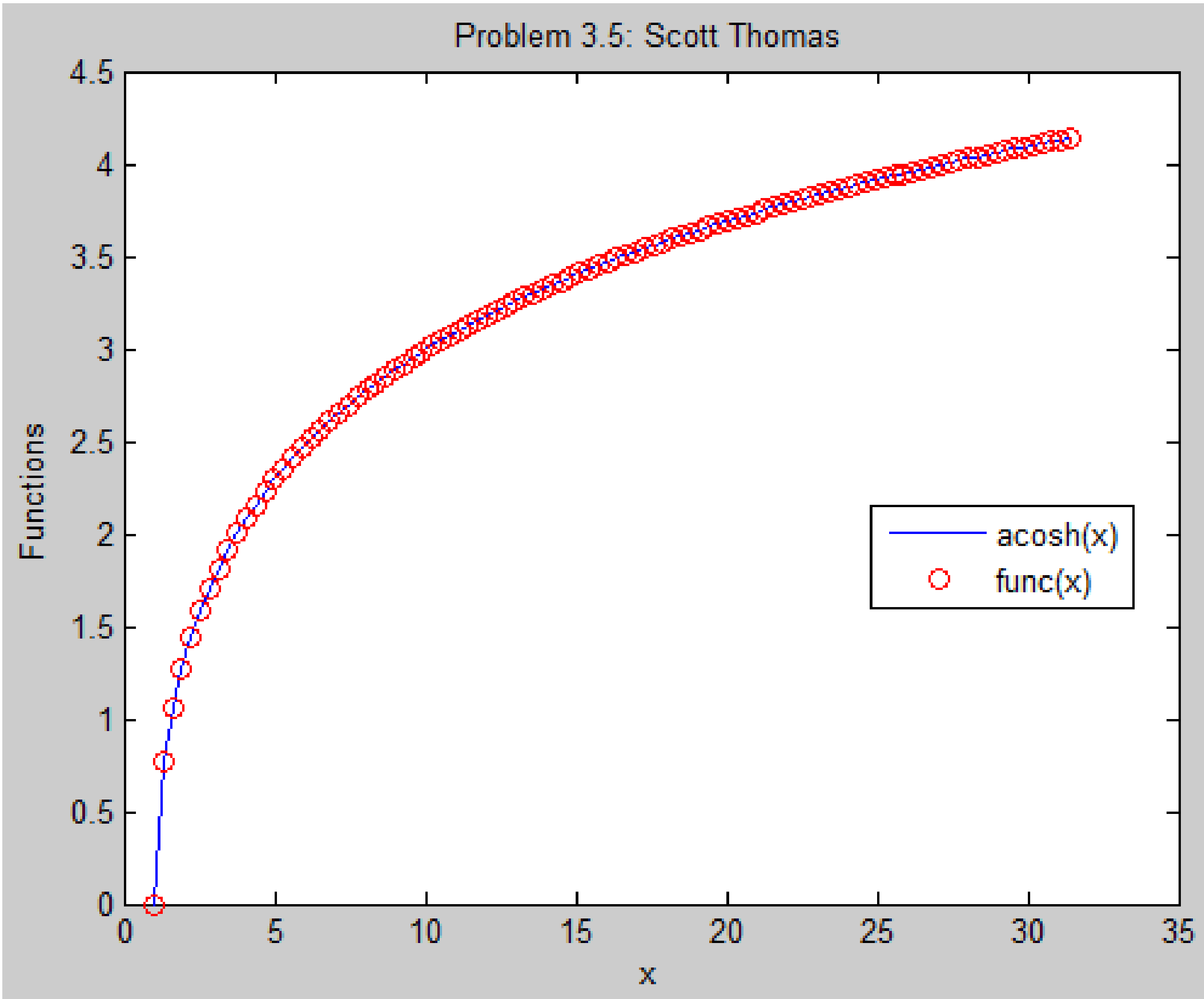
Built-In Functions:

Table 3.1–3 Hyperbolic functions

Hyperbolic	
`cosh(x)`	Hyperbolic cosine; $\cosh x = (e^x + e^{-x})/2$.
`coth(x)`	Hyperbolic cotangent; $\cosh x/\sinh x$.
`csch(x)`	Hyperbolic cosecant; $1/\sinh x$.
`sech(x)`	Hyperbolic secant; $1/\cosh x$.
`sinh(x)`	Hyperbolic sine; $\sinh x = (e^x - e^{-x})/2$.
`tanh(x)`	Hyperbolic tangent; $\sinh x/\cosh x$.
Inverse hyperbolic	
`acosh(x)`	Inverse hyperbolic cosine
`acoth(x)`	Inverse hyperbolic cotangent
`acsch(x)`	Inverse hyperbolic cosecant
`asech(x)`	Inverse hyperbolic secant
`asinh(x)`	Inverse hyperbolic sine
`atanh(x)`	Inverse hyperbolic tangent

Problem 3.5:

Create a vector for x over the range $1 \leq x \leq 10\pi$. Use a MATLAB script file to plot both $\cosh^{-1}(x)$ and $\ln\left(x + \sqrt{x^2 - 1}\right)$ to show that they are the same function.

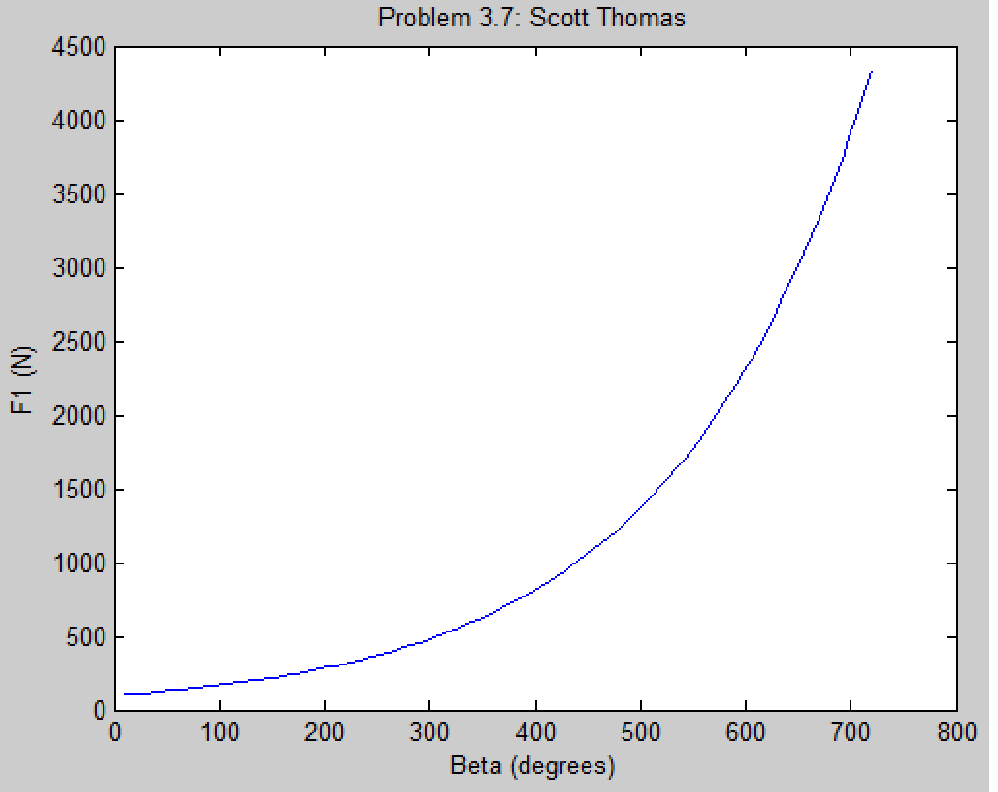


Problem 3.7:

When a belt is wrapped around a cylinder, the relation between the belt forces on each side of the cylinder is

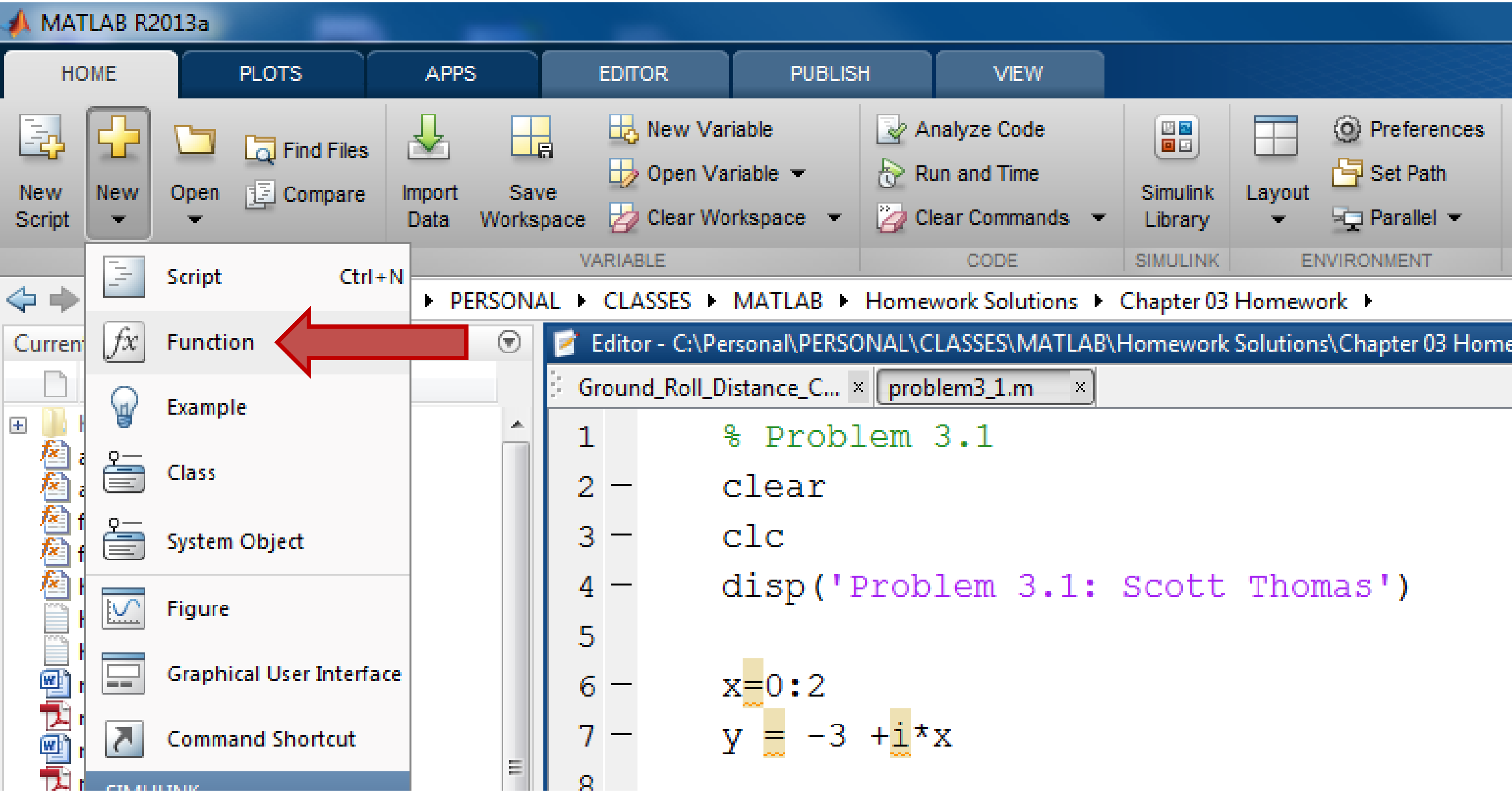
$$F_1 = F_2 e^{\mu\beta}$$

where β is the angle of wrap of the belt in radians and μ is the friction coefficient. Create a vector for β over the range $10 \leq \beta \leq 720°$. Use a MATLAB script file to plot the force F_1 over the range of β for $\mu = 0.3$ and $F_2 = 100$ N. (Hint: Be careful with β!)



User-Defined Functions:

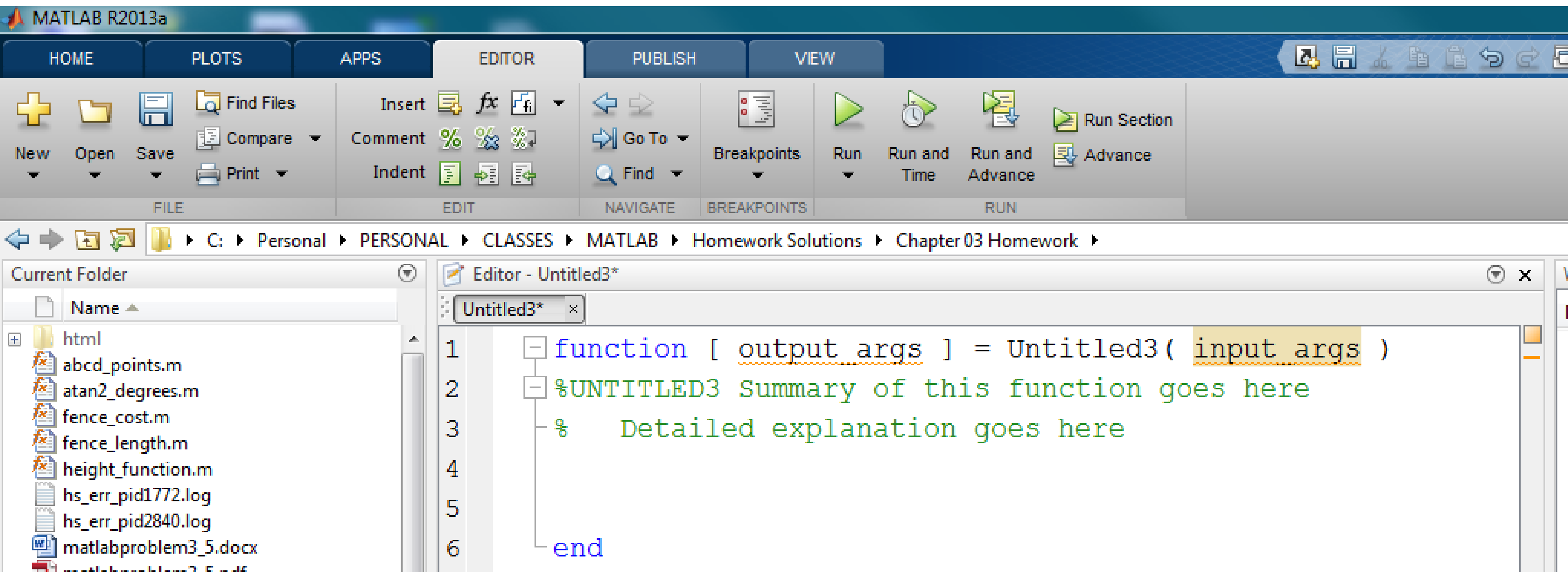
Open MATLAB. Open a new **Function File**.



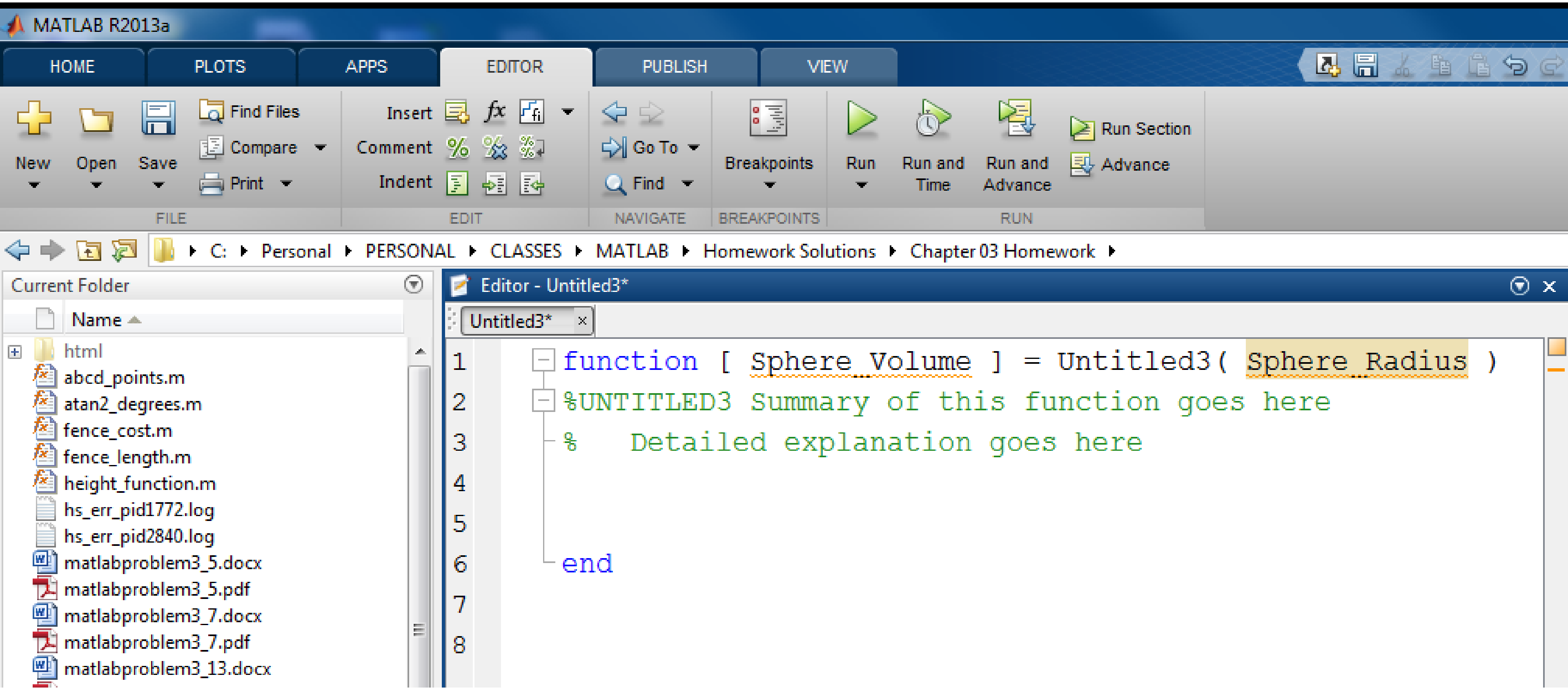
The **Function File** consists of a beginning and end which is outlined on the left of the function. This function will be used to calculate the volume of a sphere:

$$V = \frac{4\pi}{3} r^3$$

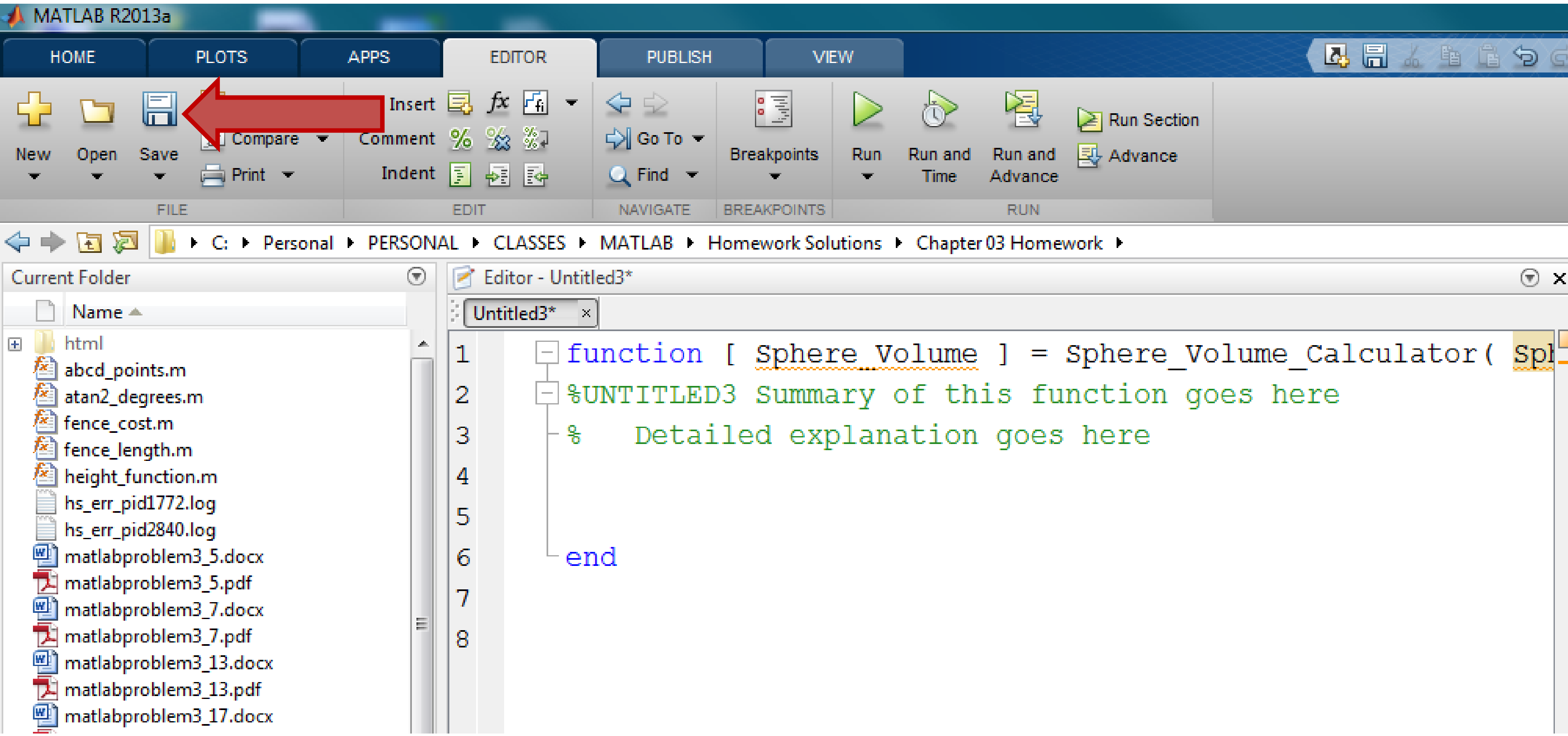
The radius of the sphere will be an input argument, and the volume of the sphere will be an output variable. Double-click on **input args** and replace it with **Sphere_Radius**. Double-click on **output args** and replace it with **Sphere_Volume**.

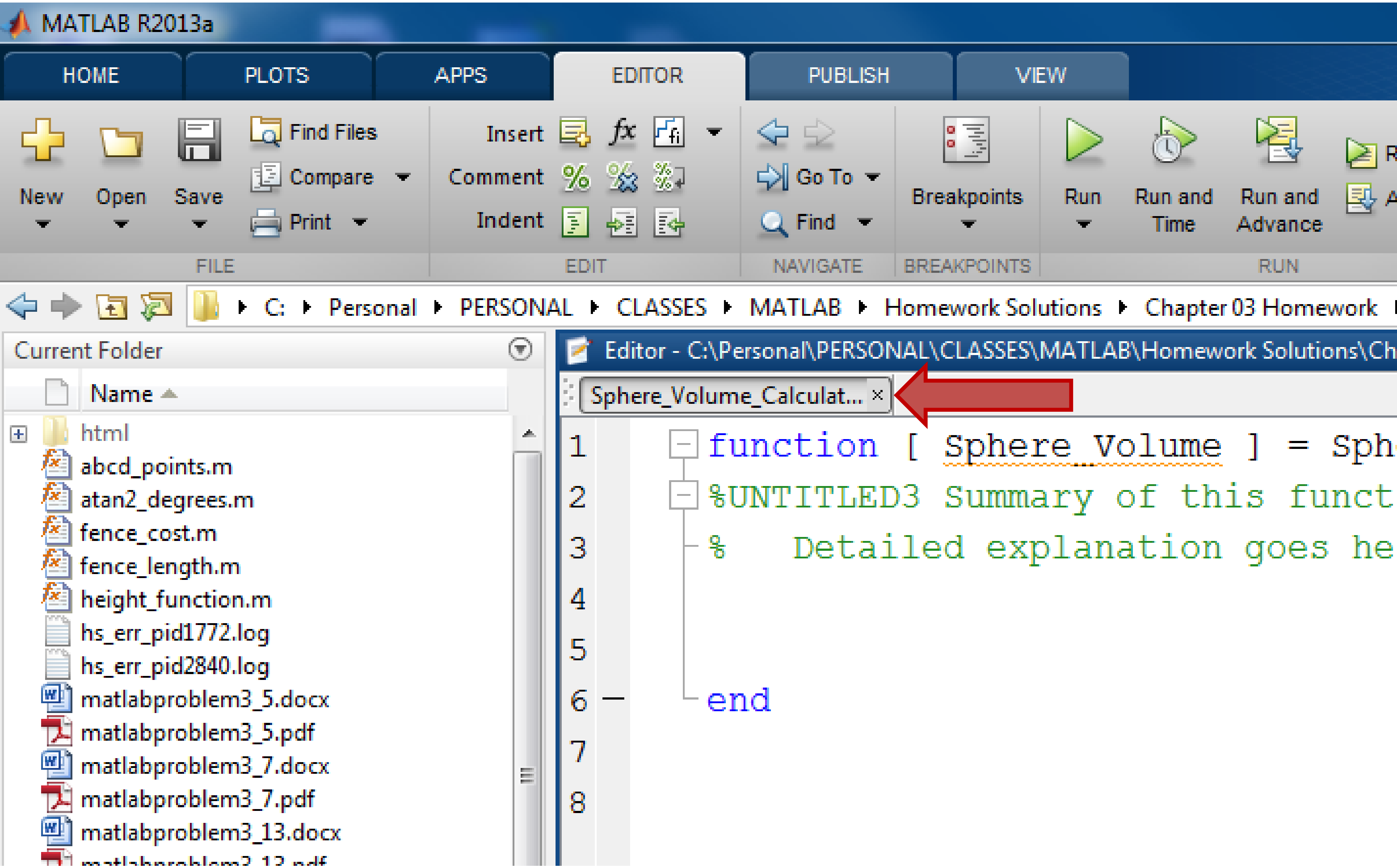


Double-click on **Untitled** and replace it with **Sphere_Volume_Calculator**.



Press the Save button. This will automatically name the **Function File** appropriately.

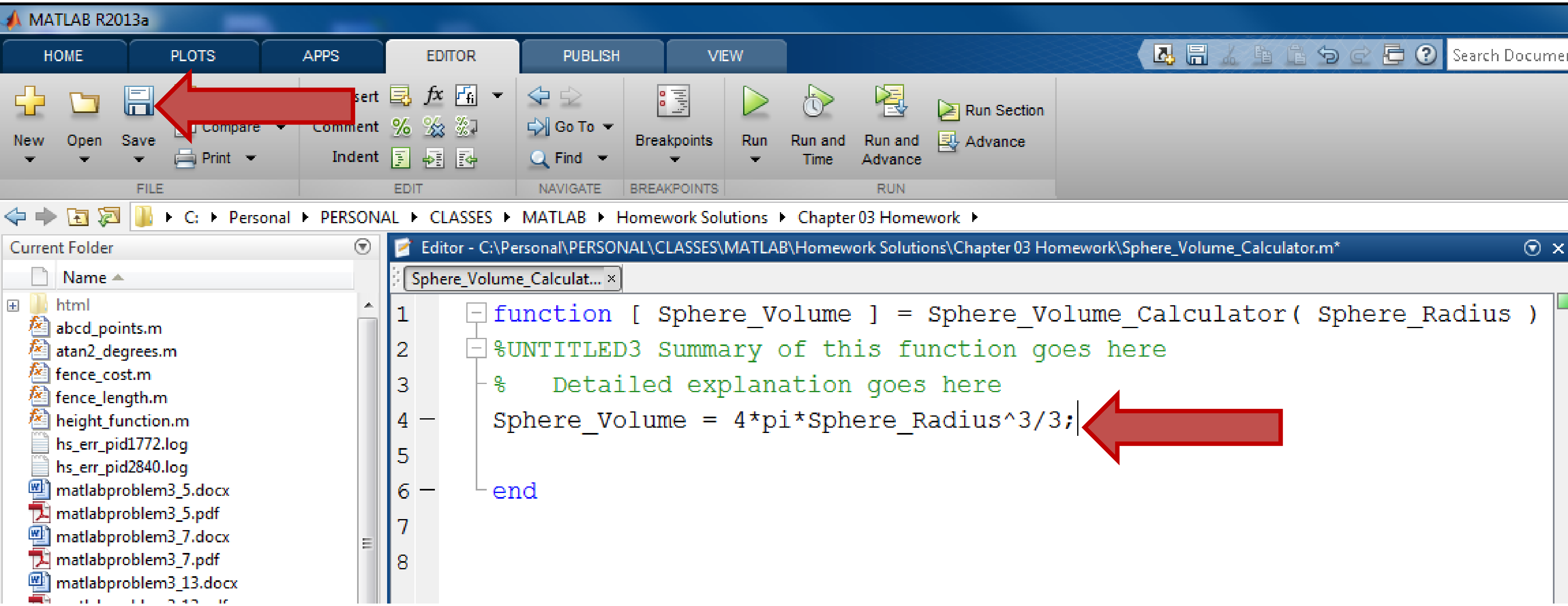




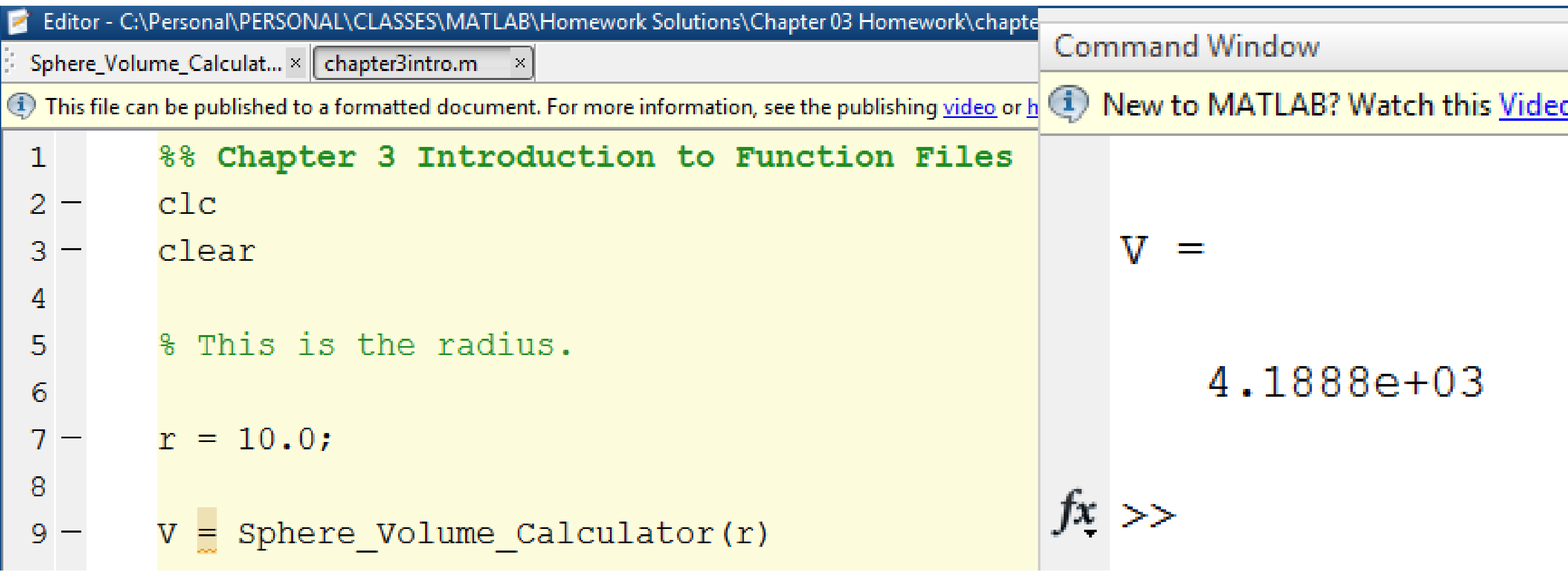
MATLAB R2013a
HOME
PLOTS
APPS
EDITOR
PUBLISH
VIEW
New
Open
Save
Find Files
Compare
Print
Insert
Comment
Indent
Go To
Find
Breakpoints
Run
Run and Time
Run and Advance
FILE
EDIT
NAVIGATE
BREAKPOINTS
RUN
C: ▸ Personal ▸ PERSONAL ▸ CLASSES ▸ MATLAB ▸ Homework Solutions ▸ Chapter 03 Homework
Current Folder
Name
html
abcd_points.m
atan2_degrees.m
fence_cost.m
fence_length.m
height_function.m
hs_err_pid1772.log
hs_err_pid2840.log
matlabproblem3_5.docx
matlabproblem3_5.pdf
matlabproblem3_7.docx
matlabproblem3_7.pdf
matlabproblem3_13.docx
Editor - C:\Personal\PERSONAL\CLASSES\MATLAB\Homework Solutions\Ch
Sphere_Volume_Calculat...
1 function [Sphere_Volume] = Sph
2 %UNTITLED3 Summary of this funct
3 % Detailed explanation goes he
6 end

Type in the equation for the volume of a sphere as shown below. You must use the input variable name **Sphere_Radius** and the output variable name **Sphere_Volume** as specified in the **Function Statement**. Make sure to Save the **Function File** whenever you make changes to it.

$$V = \frac{4\pi}{3} r^3$$



Open a new MATLAB Script File. This will be the **Calling Script File**. Save it into the same folder location as the **Function File** that you just created. Use the radius value and submit it as an input to the **Function File**. Notice that the input argument name (**r**) does not have to match the name in the function file (**Sphere_Radius**). The names of variables inside the **Function File** are called **Local Variables** (**Local** to the **Function**). Run this script file and check the result by hand.



Anonymous Functions are used to create a simple function within a script file. We will create an **Anonymous Function** to calculate the surface area of a sphere as shown below. Again, the input argument name in the calling statement does not have to be the same as that in the defining statement. Run the script file again and check the result by hand.

```
%% Chapter 3 Introduction to Function Files
clc
clear
% This is the radius.
r = 10.0;
V = Sphere_Volume_Calculator(r)

% Surface area of a sphere:
Surface_Area_Sphere = @(Sphere_Radius) 4*pi*Sphere_Radius^2;

S = Surface_Area_Sphere(r)
```

```
%% Chapter 3 Introduction to Function Files
clc
clear
% This is the radius.
r = 10.0;
V = Sphere_Volume_Calculator(r)

% Surface area of a sphere:
Surface_Area_Sphere = @(Sphere_Radius) 4*pi*Sphere_Radius^2;

S = Surface_Area_Sphere(r)
```

Command Window

New to MATLAB? Watch this Vid

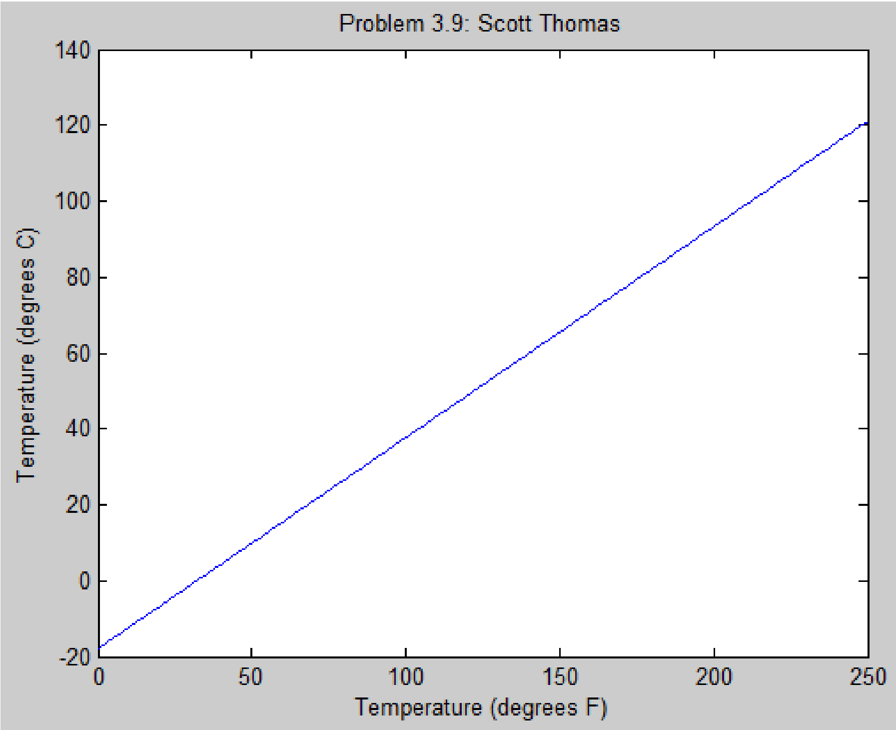
```
V =

   4.1888e+03


S =

   1.2566e+03
```

Problem 3.9:

Create a **Function File** that accepts temperature in degrees Fahrenheit (°F) and computes the corresponding value in degrees Celsius (°C). The relation between the two is

$$T°\mathrm{C} = \frac{5}{9}(T°\mathrm{F} - 32)$$

Create a **Calling Script File** to plot the temperature in °C versus the temperature in °F over the range $0 \leq T \leq 250$°F.



Function Functions:

Functions that act on other functions are called **Function Functions**. The **@function** is called a **Function Handle**, which is used to reference a **User-Defined Function**.

Table 3.2–1 Minimization and root-finding functions

Function	Description
`fminbnd(@function,x1,x2)`	Returns a value of `x` in the interval `x1` ≤ `x` ≤ `x2` that corresponds to a minimum of the single-variable function described by the handle `@function`.
`fminsearch(@function,x0)`	Uses the starting vector `x0` to find a minimum of the multivariable function described by the handle `@function`.
`fzero(@function,x0)`	Uses the starting value `x0` to find a zero of the single-variable function described by the handle `@function`.

Search Documentation: function handle

function_handle (@)

Handle used in calling functions indirectly

Syntax

handle = @functionname

handle = @(arglist)anonymous_function

Description

handle = @functionname returns a handle to the specified MATLAB function.

A function handle is a MATLAB value that provides a means of calling a function indirectly. You can pass function handles in calls to other functions (often called *function functions*).

At the time you create a function handle, the function you specify must be on the MATLAB path.

Problem 3.11:

11. A water tank consists of a cylindrical part of radius r and height h and a hemispherical top. The tank is to be constructed to hold 600 m^3 when filled. The surface area of the cylindrical part is $2\pi rh$, and its volume is $\pi r^2 h$. The surface area of the hemispherical top is given by $2\pi r^2$, and its volume is given by $2\pi r^3/3$. The cost to construct the cylindrical part of the tank is \$400 per square meter of surface area; the hemispherical part costs \$600 per square meter. Use the `fminbnd` function to compute the radius that results in the least cost. Compute the corresponding height h.

$$A_{\text{cyl}} = 2\pi rh;\ A_{\text{hemi}} = 2\pi r^2;\ V_{\text{cyl}} = \pi r^2 h;\ V_{\text{hemi}} = \frac{2\pi}{3} r^3$$

$$V = V_{\text{cyl}} + V_{\text{hemi}} = \pi r^2 h + \frac{2\pi}{3} r^3 = 600\ \text{m}^3$$

$$h = \frac{1}{\pi r^2}\left(600 - \frac{2\pi}{3} r^3\right)$$

$$\text{Cost} = \left(400 \frac{\$}{\text{m}^2}\right) A_{\text{cyl}} + \left(600 \frac{\$}{\text{m}^2}\right) A_{\text{hemi}}$$

$$\text{Cost} = \left[400(2\pi rh) + 600[2\pi r^2]\right]\ \$$$

<u>Problem 3.11:</u>

Search Documentation: fminbnd

fminbnd

Find minimum of single-variable function on fixed interval

Syntax

x = fminbnd(fun,x1,x2)

Description

fminbnd finds the minimum of a function of one variable within a fixed interval.

x = fminbnd(fun,x1,x2) returns a value x that is a local minimizer of the function that is described in fun in the interval x1< x < x2. fun is a function_handle.

Problem 3.11:

Create a new **Function File** that calculates the following in terms of the **Radius**:

- Water Heater Height
- Cost of the Water Heater

Input: Radius;   **Output**: Water Heater Cost

Test your **Function File** using the **Calling File** for a radius of r = 1.0 m.

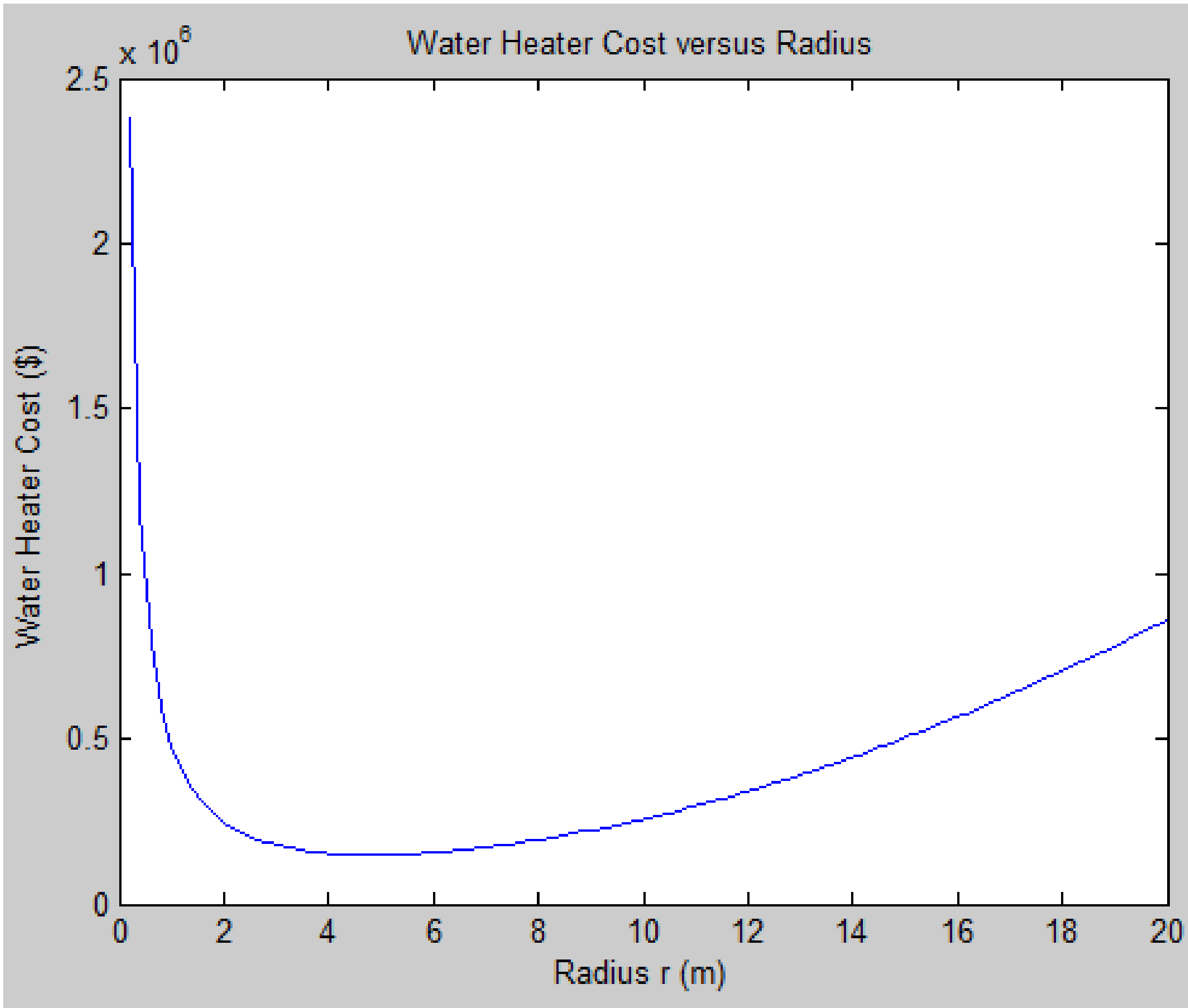
$$h = \frac{1}{\pi r^2}\left(600 - \frac{2\pi}{3} r^3\right)$$

$$\text{Cost} = [400(2\pi r h) + 600(2\pi r^2)]\ \$$$

Cost = WH_Calculator(1.0) h = 190.3193 cost = 4.8209e+05

Problem 3.11:

Use the **Calling Script File** to plot the cost of the water heater. Make sure the **Calling Script File** is in the same folder as the **Function File**.



Problem 3.11:

Calculate the minimum cost of the water heater using the Built-In Function **fminbnd**. Then compute the corresponding height of the water heater.

```
fminbnd(@function,x1,x2)
```

Returns a value of `x` in the interval `x1` $\le$ `x` $\le$ `x2` that corresponds to a minimum of the single-variable function described by the handle `@function`.

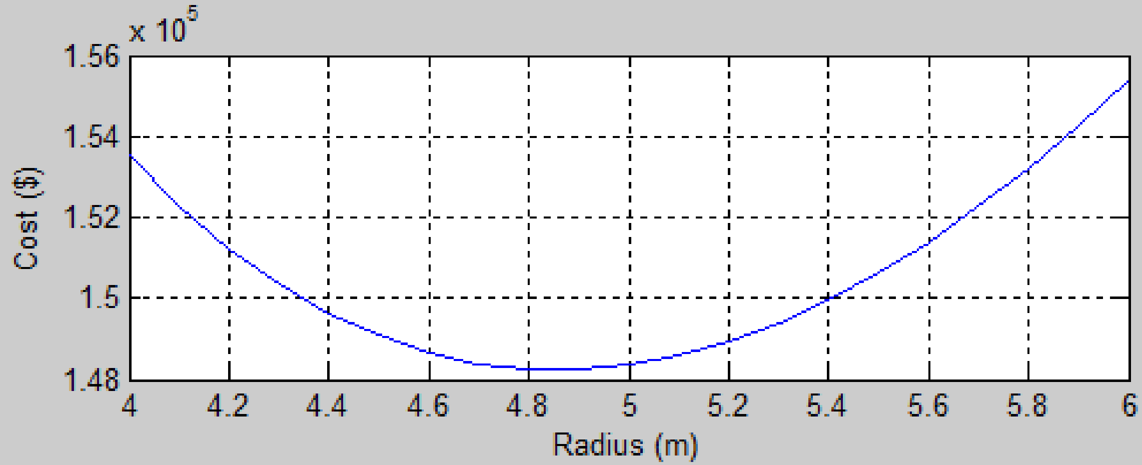
$$h = \frac{1}{\pi r^2}\left(600 - \frac{2\pi}{3} r^3\right)$$

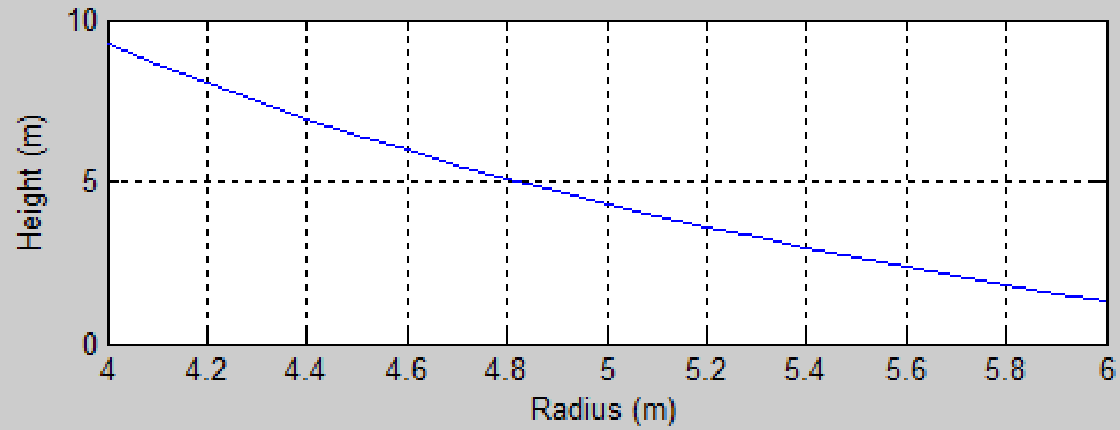
$$\text{Cost} = [400(2\pi r h) + 600(2\pi r^2)]\ \$$$

Problem 3.11:

```
N = 100;
radius = linspace(0,20,N);
cost = prob3(radius);
plot(radius,cost)

rmin = fminbnd(@prob3, 0, 20)
hmin = 1/pi/rmin^2.*(600 - 2*pi/3*rmin^3)
costmin = prob3(rmin)
```

```
rmin =

    4.8572

hmin =

    4.8571

costmin =

   1.4823e+05
```





Problem 3.19:

Create an anonymous function for $20x^2 - 200x + 3$ and use it

a. To plot the function to determine the approximate location of its minimum

b. With the `fminbnd` function to precisely determine the location of the minimum

The argument `fun` can also be a function handle for an anonymous function. For example, to find the minimum of the function $f(x) = x^3 - 2x - 5$ on the interval `(0,2)`, create an anonymous function `f`

```
f = @(x)x.^3-2*x-5;
```

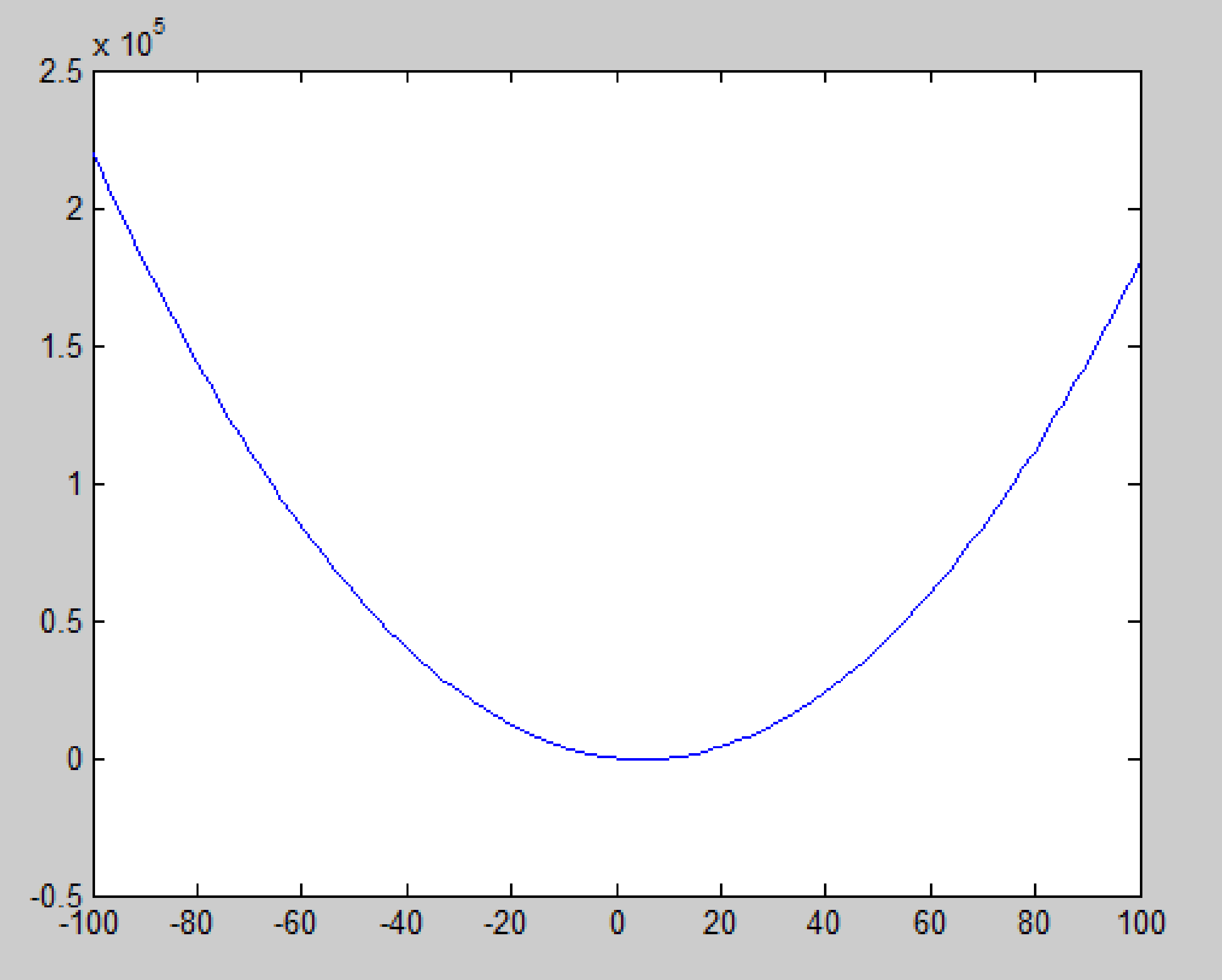
Then invoke `fminbnd` with

```
x = fminbnd(f, 0, 2)
```

Problem 3.19:

```
N = 10000
x = linspace(-100,100,N);
y = @(x)  20*x.^2 - 200*x + 3;
yplot = y(x);
plot(x,yplot)
xmin = fminbnd(y,-100,100)
```

```
ymin =

    5.0000
```



Working with Data Files

xlsread

Read Microsoft Excel spreadsheet file

Syntax

num = xlsread(filename)

Read data from the first worksheet.

filename = 'myExample.xlsx'; A = xlsread(filename)

xlsread returns the numeric data in array A.

load

Load data from MAT-file into workspace

Syntax

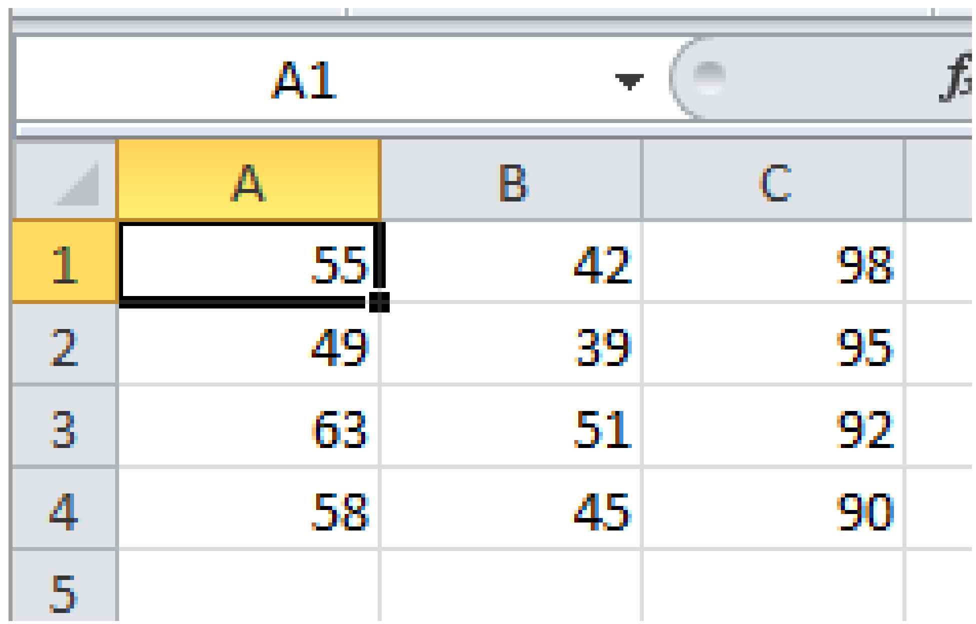
S = load(filename, '-ascii')

Forces load to treat the file as an ASCII file, regardless of the extension.

Problem 3.23:

Use Microsoft Excel to create a file containing the following data. Then use the `load` function to load the file into MATLAB, and use the `mean` function to compute the mean value of each column.

55	42	98
49	39	95
63	51	92
58	45	90



A1

	A	B	C
1	55	42	98
2	49	39	95
3	63	51	92
4	58	45	90
5			

Problem 3.23:

```
filename = 'prob323.xlsx';
A = xlsread(filename)
col_mean_1 = mean(A(:,1))
col_mean_2 = mean(A(:,2))
col_mean_3 = mean(A(:,3))
```

Command Window

```
A =

    55    42    98
    49    39    95
    63    51    92
    58    45    90
```

```
col_mean_1 =

   56.2500


col_mean_2 =

   44.2500


col_mean_3 =

   93.7500
```