King Abdulaziz University
Faculty of Engineering

Mechanical Engineering Department

MEP460
Heat Exchanger Design

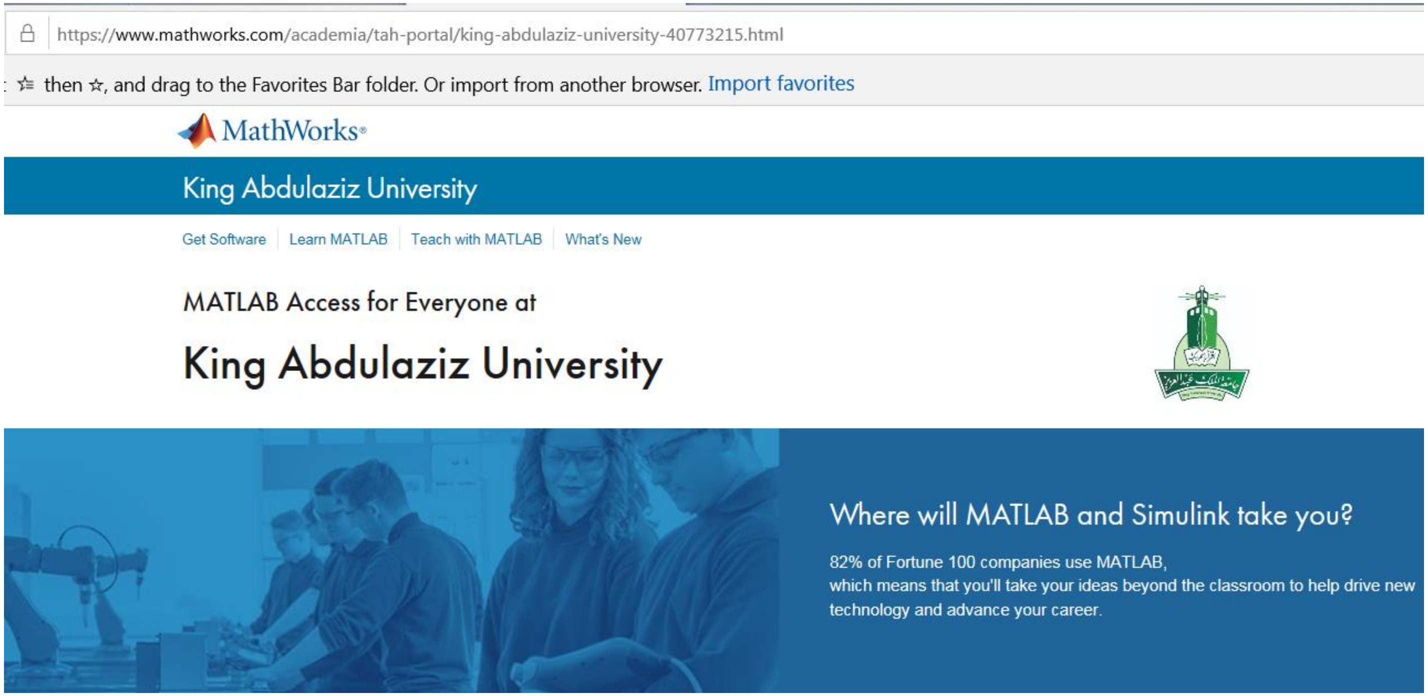
MATLAB Introduction

Sept. 2020

Contents

1-Introduction

- MATLAB is a very powerful programming software used by engineers and scientists
- The software is available for students and faculty members at KAU
- MATLAB has several toolboxes for different fields



https://www.mathworks.com/academia/tah-portal/king-abdulaziz-university-40773215.html

Some of MATLAB toolboxes

What Toolboxes are available to you?

1.MATLAB
2.Simulink
3.MATLAB Distributed Computing Server (MDCS)
4.Aerospace Blockset
5.Aerospace Toolbox
6.Antenna Toolbox
7.Audio System Toolbox
8.Automated Driving System Toolbox
9.Bioinformatics Toolbox
10.Communications System Toolbox
11.Computer Vision System Toolbox
12.Control System Toolbox
13.Curve Fitting Toolbox
14.Data Acquisition Toolbox
15.Database Toolbox
16.Datafeed Toolbox
17.DSP System Toolbox
18.Econometrics Toolbox
19.Embedded Coder
20.Filter Design HDL Coder
21.Financial Toolbox
22.Financial Instruments Toolbox
23.Fixed-Point Designer
24.Fuzzy Logic Toolbox
25.Global Optimization Toolbox
26. GPU Coder
27. HDL Coder
28. HDL Verifier
29. Image Acquisition Toolbox
30. Image Processing Toolbox
31. Instrument Control Toolbox
32. LTE HDL Toolbox
33. LTE System Toolbox
34. Mapping Toolbox
35. MATLAB Compiler SDK
36. MATLAB Coder
37. MATLAB Compiler
38. MATLAB Report Generator
39. Model Predictive Control Toolbox
40. Model-Based Calibration Toolbox
41. Neural Network Toolbox
42. OPC Toolbox
43. Optimization Toolbox
44. Parallel Computing Toolbox
45. Partial Differential Equation Toolbox
46. Phased Array System Toolbox
47. Polyspace Bug Finder
48. Polyspace Code Prover
49. Powertrain Blockset
50. Predictive Maintenance Blockset
51. RF Blockset
52. RF Toolbox
53. Risk Management Toolbox
54. Robotics System Toolbox
55. Robust Control Toolbox
56. Signal Processing Toolbox
57. SimBiology
58. Simscape Driveline
59. Simscape Electronics
60. SimEvents
61. Simscape Fluids
62. Simscape Power Systems
63. Simscape Multibody
64. Simscape
65. Simulink 3D Animation
66. Simulink Check
67. Simulink Code Inspector
68. Simulink Coder
69. Simulink Coverage
70. Simulink Control Design
71. Simulink Design Optimization
72. Simulink Design Verifier
73. Simulink Desktop Real-Time
74. Simulink PLC Coder
75. Simulink Real-Time
76. Simulink Report Generator
77. Simulink Test
78. Spreadsheet Link
79. Stateflow
80. Statistics and Machine Learning Toolbox
81. Symbolic Math Toolbox
82. System Identification Toolbox
83. Text Analytics Toolbox
84. Trading Toolbox
85. Thingspeak
86. Trading Toolbox
87. Vehicle Dynamics Blockset
88. Vehicle Network Toolbox
89. Vision HDL Toolbox
90. Wavelet Toolbox
91. WLAN System Toolbox

Free MATLAB alternative programs

- GNU octave
  https://www.gnu.org/software/octave/index

- Scilab

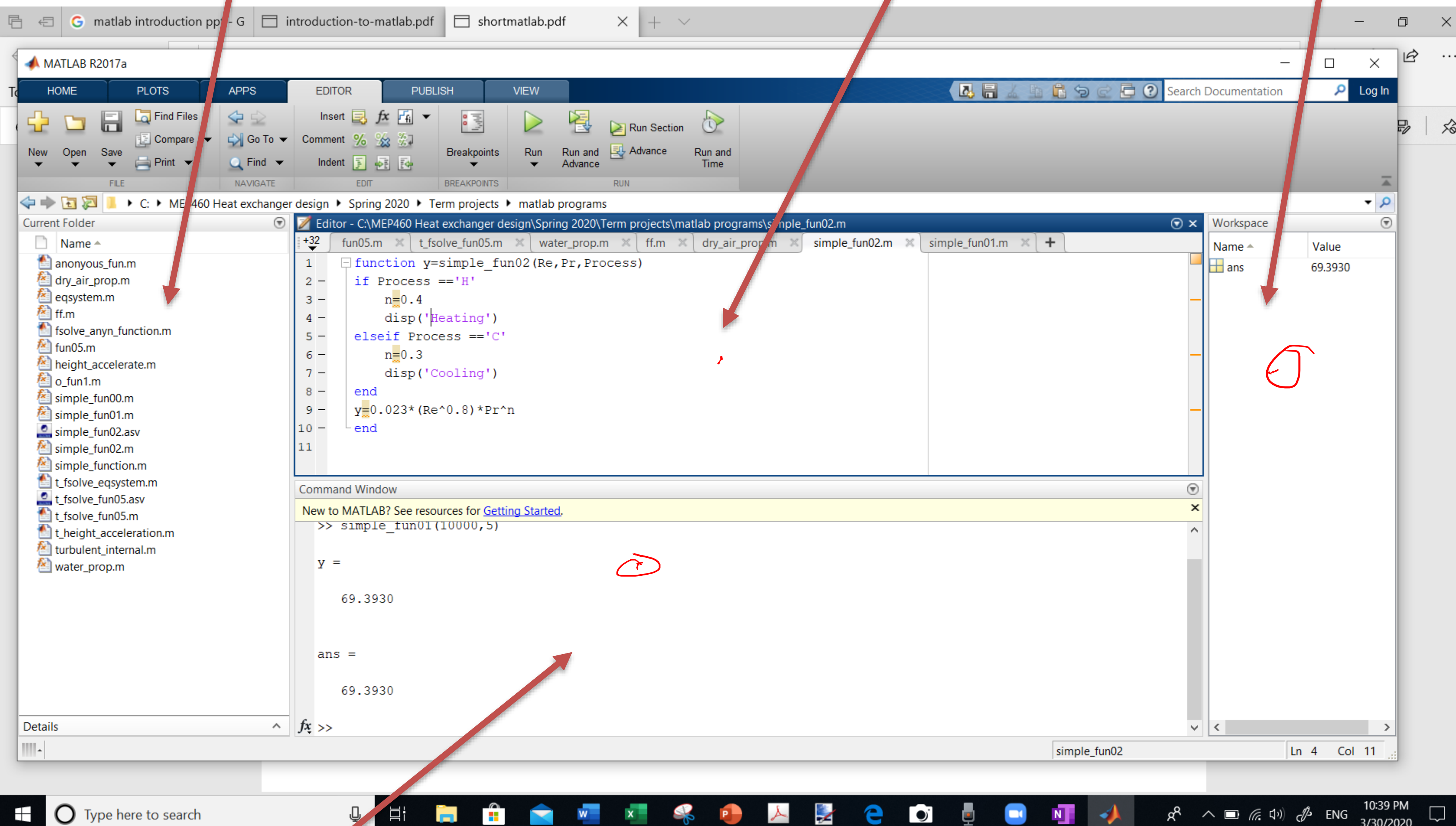
  https://www.scilab.org/

User interface

Current directory

Editor Window

Work space



Command window

2-How to get help in MATLAB

A- Using MATLAB environment

write in the command screen:

- help general [General information]
- demo [visual help for MATLAB]
- help fun_name [for example help plot]
- help lang [programming language construct]
- help elmat [help on elementary matrices]
- help graphics [handle graphics]
- help funfun [help on functions]

You can press of any outcome from the above help for more help in each topic

B-Utilizing the internet

- Lots of sites and courses free
- Youtube just write MATLAB help, MATLAB tutorials or MATLAB your choice of topic
- Lots of courses notes and pdf files
- pdf books

3-Basics

>> a=10

a =

10

>> b=1:10 % b is a column vector of length 10

b =

Columns 1 through 10

1 2 3 4 5 6 7 8 9 10

c=[1;2;3;4;5]

% c is a row vector of length 5

z=[1 2 3 10; 4 5 6 20 ; 7 8 9 30] % z a 2D vector 3 rows and 4 columns

z =

1	2	3	10
4	5	6	20
7	8	9	30

>> z(2,3)

% gives 6

w=1:3:20 % w is a vector starting with 1 to 20 and increment is 3

% is for comment statement

size(b) will give

1 10

size(c) will give

5 1

3-Basics

Some useful commands

W = 10

w = 5

% used for comment statement

clc to clear the screen

clear to clear all the variables in workspace

who to show the variables in the workspace

whos to show more information about the variables

ls list of files in the current directory

save fname.mat $\rightarrow$ save the variables in workspace and values in fname.mat file

load fname.mat $\rightarrow$ to load the variables back to workspace

Variables in MATLAB are case sensitive i.e. x is different than X

3-Basics

```
% for log base 10 use
y=log10(x)

% for ln use
y=log(x)

% for examples
% if x=10 then
y1=log10(10)
% will give 1.0 and
y2=log(10)
Wil give 2.3026
```

4-m-functions

Function y=myfun(x)
y=output arguments.
myfun =function name
x = input arguments

```
function y=simple_fun01(Re,Pr)
% file name fun01
% this is a simple function to
calculate
% Nu for internal turbulent
% Nu=0.023 (Re^ 0.8)*Pr^0.4
% See Incropera Heat transfer
book
y=0.023*(Re^0.8)*Pr^0.4
end
```

% if you type in the command window
W=simple_fun01(10000,5)
% To get
W=69.393

4-m-functions

```
function y=simple_fun02(Re,Pr,Process)
% insert information about the function
% input:
% Re, Pr and Process
% Process is character either H for heating
or
% C cold process
% output Nu
if Process =='H'
    n=0.4
    disp('Heating')
elseif Process =='C'
    n=0.3
    disp('Cooling')
end
y=0.023*(Re^0.8)*Pr^n
end
```

Nu=simple_fun02(10000,5,'H')

Nu=69.393

4-m-functions

Array input and output

```
function y=fun07(x1,x2)
% write information about the function
% so that you can access the help
% by writing help name of the function
% i.e. for our example write help fun07
% filename=fun07.m
% another example of simple function
% y=x1^2+x2^2
% notice if x1 and x2 are vectors then the above
equation
% should be written as
 y=x1.^2+x2.^2
 end
```

Example:

s1=[1:3], s2=[7:9]

Q=fun07(s1,s2)

Q=[50 68 90]

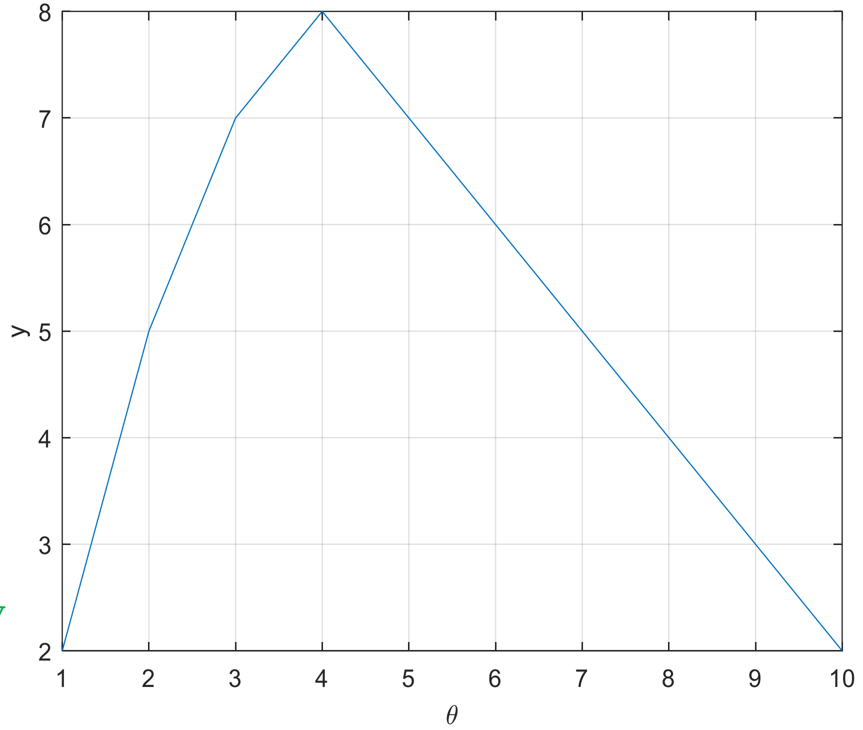
5-FOR & IF statements

```
clc
clear
n=1:10
for i=1:10
    if i <=5
        b(i)=100
    else
        b(i)=b(i-1)*2
    end
end
% to exit for loop or while loop
use break
```

B(1) … B(5)=100
B(6)=200, B(7)=400, B(8)=800, B(9)=1600, B(10)=3200

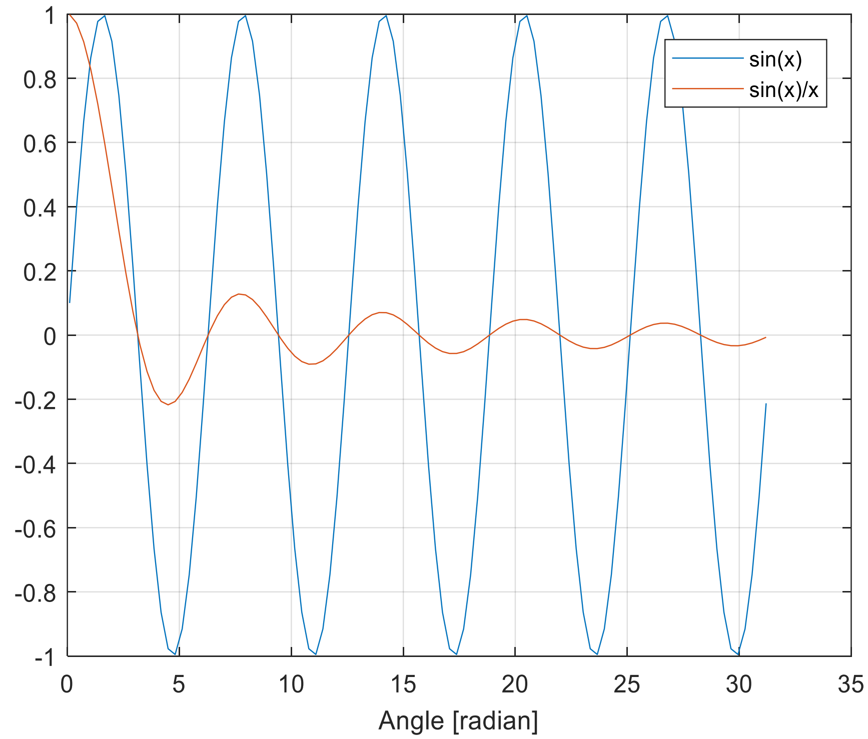
6-Simple Plotting

```
clc
clear
% simple plot example
% filename t_plot_00
x=[1:10]; y=[2 5 7 8 7 6 5 4 3
2];
plot(x,y)
xlabel('\theta’) % use \ for
Greek letters
ylabel('y')
grid on  % show grids in x and y
coordinate
```



6-Simple plotting

```
clc
clear
x=0.1:pi/10:10*pi
y=sin(x)
z=y./x
plot(x,y,x,z)
%plot(x,z)
grid on
legend('sin(x)', 'sin(x)/x')
xlabel('Angle [radian]')
```



7-Logical operators

==	Equal to
~=	Not equal to
<	Less than
<=	Less than or equal
>	Greater than
>=	Greater than or equal
&	And
|	Or
~	Not
xor	Exclusive or

8- Interpolation functions- (interp1 function)

If you have two vectors of values and you want to interpolate values, you can use the MATLAB function interp1

Given the saturated T & P for water, find the saturated pressure at T=283 K

T=[273.15 275.0 280.0 290.0 300.0 305.0]
P=[0.0061 0.00697 00099 0.01387 0.01917 0.02617]

Water saturation data
(Ref: Incropera)

T [K]	P [bar]
273.15	0.00611
275.0	0.00697
280.0	0.00990
290.0	0.01387
300.0	0.01917
305.0	0.02617

Format: xp=interp1(T,P,xt)

xp=interp1(T,P,283)

Will give
Xp=0.0123 bar

8- Interpolation functions- (interp1 function)

Interpolation functions General use

Vq = interp1(X,V,Xq,METHOD) specifies the interpolation method.
The available methods are:

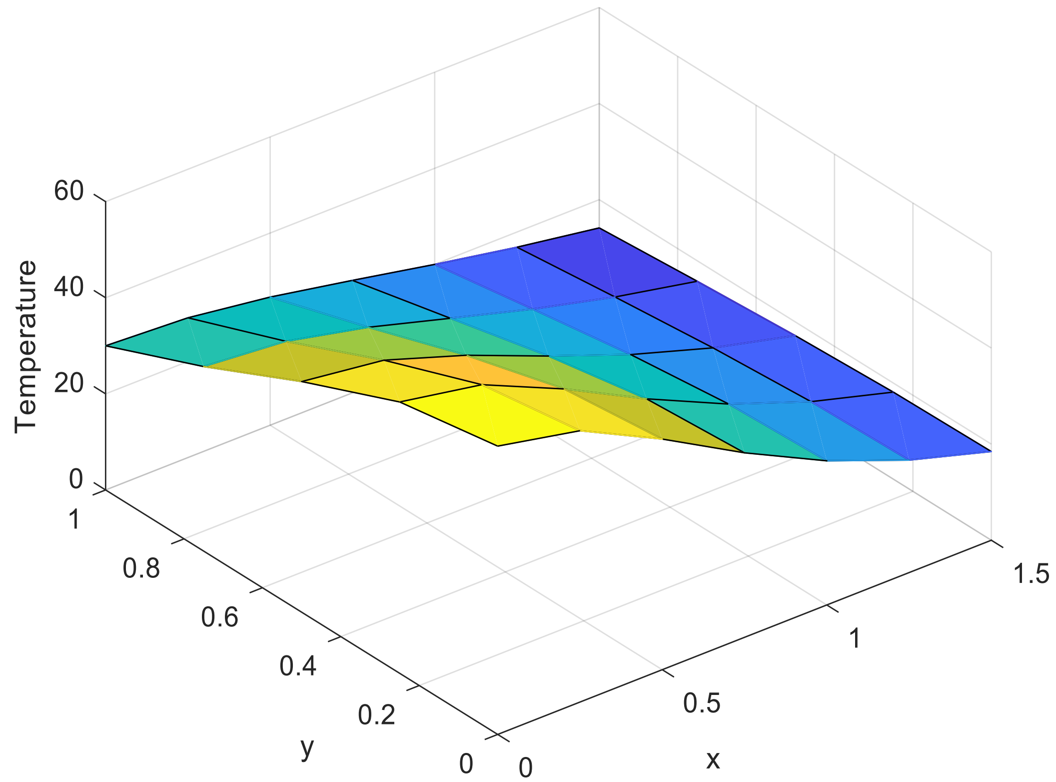
'linear' - (default) linear interpolation
'nearest' - nearest neighbor interpolation
'next' - next neighbor interpolation
'previous' - previous neighbor interpolation
'spline' - piecewise cubic spline interpolation (SPLINE)
'pchip' - shape-preserving piecewise cubic interpolation
'cubic' - same as 'pchip'
'v5cubic' - the cubic interpolation from MATLAB 5, which does not extrapolate and uses 'spline' if X is not equally spaced.

9- 2 D interpolation
use of interp2 Command

+24 | t_interp2b.m | tfprintf.m | tfprintf2.m | fprintf_example_00.m | t_interp2a.m | t_interp1a.m

```
% File name t_interp2b.m
% test for 2d intrpolation  i.e. interp2.m function
% Vq = interp2(X,Y,V,Xq,Yq) interpolates to find Vq, the values of the
%     underlying 2-D function V at the query points in matrices Xq and Yq.
%     Matrices X and Y specify the points at which the data V is given.
% q = interp2(...,METHOD) specifies alternate methods.  The default
%     is linear interpolation.  Available methods are:
%
%       'nearest' - nearest neighbor interpolation
%       'linear'  - bilinear interpolation
%       'spline'  - spline interpolation
%       'cubic'   - bicubic interpolation as long as the data is
%                   uniformly spaced, otherwise the same as 'spline'

% Example from youtube. See the video which spanish
x=[0.0 0.25 0.5 0.75 1.0 1.25 1.5]
y=[0.0 0.25 0.5 0.75 1.0]
t=[60.00 56.47 48.00 38.40 30.00 23.41 18.46;
  56.47 53.33 45.71 36.92 29.09 22.86 18.11;
  48.00 45.71 40.00 33.10 26.67 21.33 17.14;
  38.40 36.92 33.10 28.24 23.41 19.20 15.74;
  30.00 29.09 26.67 23.41 20.00 16.84 14.12]
vq=interp2(x,y,t,0.6,0.3)
mesh(x,y,t)
surface(x,y,t)
xlabel('x'); ylabel('y'); zlabel('Temperature')
```

9- 2 D interpolation
use of interp2 Command



use of mesh and surface command to plot 2D plot

See the file t_interp2b.m

IO Input data into MATLAB

- Use the command **load** to load numerical data file
- Use **import Data** built in app
- Use the **fopen** and **fscanf** commands to open a file and read the data
- You can read (copy) from **Excel** file and paste into working space

10- Input data for MATLAB
Use of load command

If the data is numerical values, then one can use the command load

For example the file load oil4_num_only.dat contains oil property data

load oil4_num_only.dat

then you will get a matrix of size (17,6) with the name oil4_num_only

oil4_num_only.dat

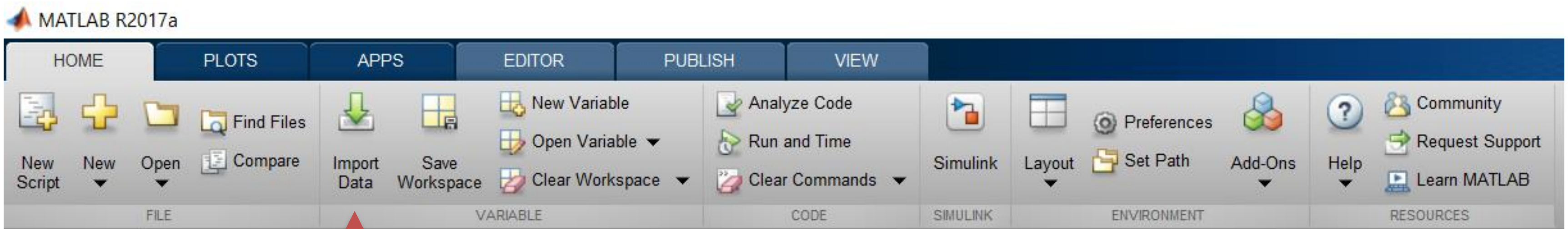
0273.0	899.1	1796.0	385.000e-2	147.0e-3	47000.0
0280.0	895.3	1827.0	217.000e-2	144.0e-3	27000.0
0290.0	890.0	1868.0	099.900e-2	145.0e-3	12900.0
0300.0	884.1	1909.0	048.600e-2	145.0e-3	06400.0
0310.0	877.9	1951.0	025.300e-2	145.0e-3	03400.0
0320.0	871.8	1993.0	014.100e-2	141.0e-3	01965.0
0330.0	865.8	2035.0	008.360e-2	143.0e-3	01205.0
0340.0	859.9	2076.0	005.310e-2	139.0e-3	00793.0
0350.0	853.9	2118.0	003.560e-2	138.0e-3	00546.0
0360.0	847.8	2161.0	002.560e-2	138.0e-3	00395.0
0370.0	841.8	2206.0	001.860e-2	137.0e-3	00300.0
0380.0	836.0	2250.0	001.410e-2	136.0e-3	00233.0
0390.0	830.6	2294.0	001.100e-2	135.0e-3	00187.0
0400.0	825.1	2337.0	000.874e-2	134.0e-3	00152.0
0410.0	818.9	2381.0	000.698e-2	133.0e-3	00125.0
0420.0	812.1	2427.0	000.564e-2	133.0e-3	00103.0
0430.0	806.5	2471.0	000.470e-2	132.0e-3	00088.0

then you can write each variable in a separate variables

```
T=oil4_num_only(:,1)    % Temperature
rho=oil4_num_only(:,2)  % density
Cp=oil4_num_only(:,3)   % Cp
```

11- Read data file using the built-in app called import Data

You can also use import file on the menu bar under home of the MATLAB



There are several options to select from which that meet your requirement.
Then you can either load the data into the workspace or generate m file

See the generated script fie read_oil_data when importing the file : oil3_with_source.dat

11- Read data fie using the built-in app called import Data

Reading the fie name oil3_with_source.dat using **import**

```
0273.0    899.1    1796.0    385.000e-2    147.0e-3    47000.0
0280.0    895.3    1827.0    217.000e-2    144.0e-3    27000.0
0290.0    890.0    1868.0    099.900e-2    145.0e-3    12900.0
0300.0    884.1    1909.0    048.600e-2    145.0e-3    06400.0
0310.0    877.9    1951.0    025.300e-2    145.0e-3    03400.0
0320.0    871.8    1993.0    014.100e-2    141.0e-3    01965.0
0330.0    865.8    2035.0    008.360e-2    143.0e-3    01205.0
0340.0    859.9    2076.0    005.310e-2    139.0e-3    00793.0
0350.0    853.9    2118.0    003.560e-2    138.0e-3    00546.0
0360.0    847.8    2161.0    002.560e-2    138.0e-3    00395.0
0370.0    841.8    2206.0    001.860e-2    137.0e-3    00300.0
0380.0    836.0    2250.0    001.410e-2    136.0e-3    00233.0
0390.0    830.6    2294.0    001.100e-2    135.0e-3    00187.0
0400.0    825.1    2337.0    000.874e-2    134.0e-3    00152.0
0410.0    818.9    2381.0    000.698e-2    133.0e-3    00125.0
0420.0    812.1    2427.0    000.564e-2    133.0e-3    00103.0
0430.0    806.5    2471.0    000.470e-2    132.0e-3    00088.0
 T        Rho        Cp          Mu            K           Pr
 (K)      (kg/m3)  (J/kg.K)     (Pa.s)        (W/m.K)      (-)
 oil properties    ref: Incropera Principle of heat and mass transfer, 7th edition 2013
```

12- Use the fopen and fscanf commands to read data from a file

```
% Example of using fscanf command
clear
fid=fopen('filetest.dat’)
% fid is file identification given by MATAB
ll=fgetl(fid)
% fgetl is just to read the first line

b=fscanf(fid,'%g',[7 inf])
%  %g is the format of the numbers, [7 inf]
seven columns
b'  % this is now as like the original data
fclose(fid)   % to close the file
```

filename: fietest.dat

st	Year	Month	Day	Hour	Tdry	Tdew
00010	1990	01	01	01	50.01	30.9
00010	1990	01	01	02	70.90	40.17
00010	1990	01	01	03	50.01	30.9
00010	1990	01	01	04	70.90	40.17
00010	1990	01	01	05	50.01	30.9
00010	1990	01	01	06	70.90	40.17
00010	1990	01	01	07	50.01	30.9
00010	1990	01	01	10	70.90	40.17
00010	1990	01	01	11	50.01	30.9
00010	1990	01	01	12	70.90	40.17
00010	1990	01	01	13	50.01	30.9
00010	1990	01	01	14	70.90	40.17

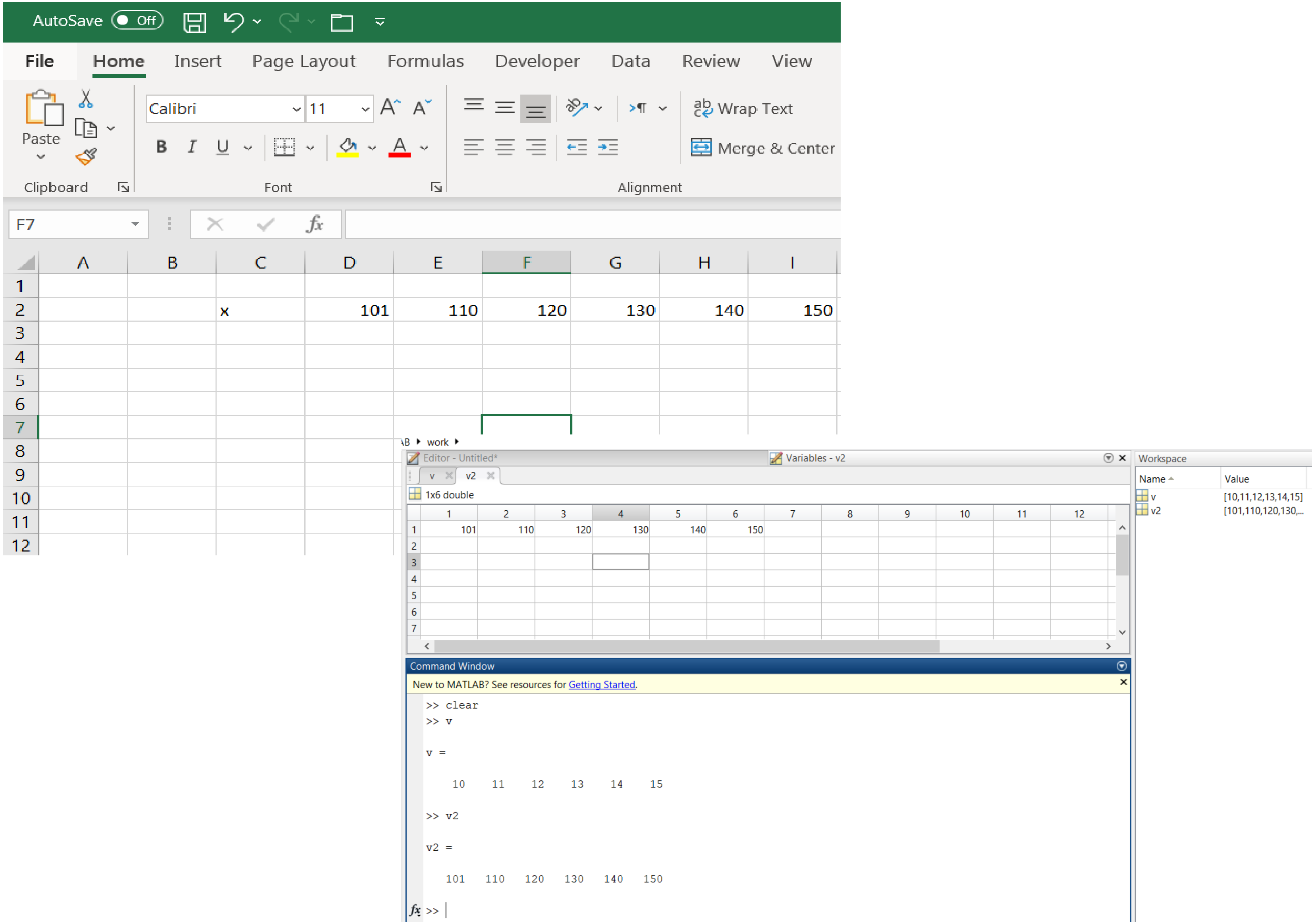
12- Use the fopen and fscanf commands to read data from a fie

Editor - C:\Users\ASUS\Documents\MATLAB\work\tfscanf.m

+11 | t_water_tpp_incrop.m | interplate_00.m | t_interp1.m | water_tpp_Incrop.m | fun07.m | read_oil_dat

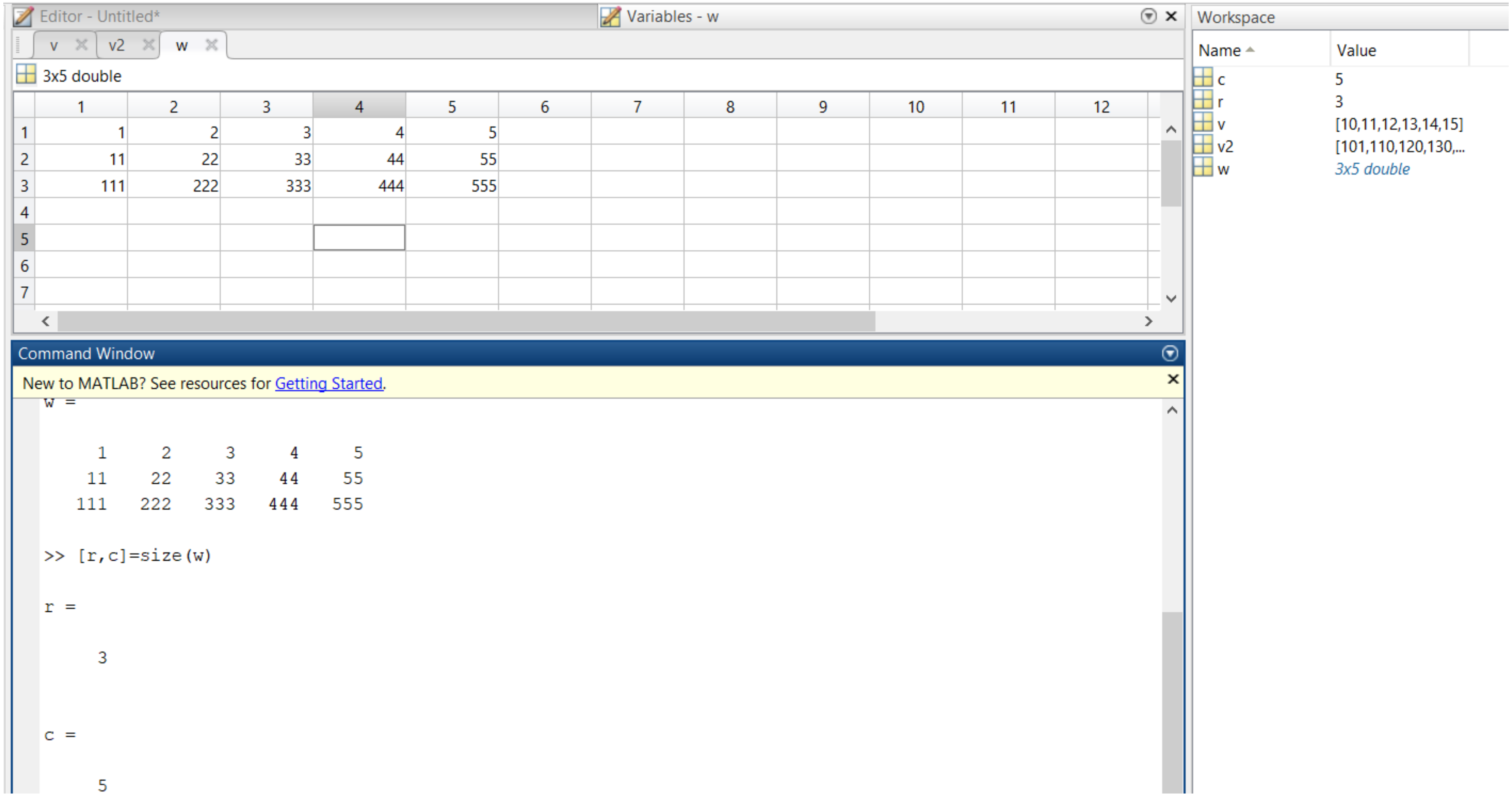
```
% test read file
% tfscanf.m
clear
clc
 fid=fopen('filetest.dat')
%[t1 t2 t3 t4 t5 t6 t7]=textread('filetest.dat','%s %s  %s  %s  %s  %s  %s\n',1)
%[a1,cr]=fscanf(fid,'%s %s')

ll=fgetl(fid)
% fgetl is just to read the first line

%b=fscanf(fid,'%g %g %g %g %g %g  %g')
b=fscanf(fid,'%g',[7 inf])
%  %g is the format of the numbers, [7 inf] seven cloumns and up to the end
%  of the file
c=b'  % c vector now is the same as the original data in the file fietext.dat
% b=textread('filetest.dat','\n %g')
 fclose(fid);  % this is to close the file
```

13- Import data from Excel



13- Import data from Excel

Import 2D variable from Excel



You can change the unnamed created variable to any name you want

Outputs from MATLAB

1-use disp command to show your results on the screen
2-vector and matrices can be copied to Excel or Microsoft Word from the workspace
3-Plotting the data using the plotting command then the figures can be copied to Word or save as an image in different format
4-Using the function fprintf (results can be shown on screen or save on a file

14-Using disp command to display results on the screen

Example using disp command

```
%fprintf

x = 1:10;
y = x.^2 +2;
z = sqrt(x);

disp('Table using disp command'), disp('x         y         z');
disp([x',y',z'])
```

15- Outputs from MATLAB into Excel or Word

Reading the file filetext.txt and move the data into Excel and Word

```
% test read file
% tfscanf.m
clear
clc
 fid=fopen('filetest.dat')
%[t1 t2 t3 t4 t5 t6 t7]=textread('filetest.dat','%s %s  %s  %s  %s  %s  %s\n',1
%[a1,cr]=fscanf(fid,'%s %s')

l1=fgetl(fid)
% fgetl is just to read the first line

%b=fscanf(fid,'%g %g %g %g %g %g  %g')
b=fscanf(fid,'%g',[7 inf])
%  %g is the format of the numbers, [7 inf] seven cloumns and up to the end
%  of the file
c=b'  % c vector now is the same as the original data in the file fietext.dat
% b=textread('filetest.dat','\n %g')
 fclose(fid);  % this is to close the file
```

Excel

AutoSave Off

File Home Insert Page Layout Formulas Developer Data

J8

	A	B	C	D	E	F	G
1							
2	10	1990	1	1	1	50.01	30.9
3	10	1990	1	1	2	70.9	40.17
4	10	1990	1	1	3	50.01	30.9
5	10	1990	1	1	4	70.9	40.17
6	10	1990	1	1	5	50.01	30.9
7	10	1990	1	1	6	70.9	40.17
8	10	1990	1	1	7	50.01	30.9
9	10	1990	1	1	10	70.9	40.17
10	10	1990	1	1	11	50.01	30.9
11	10	1990	1	1	12	70.9	40.17
12	10	1990	1	1	13	50.01	30.9
13	10	1990	1	1	14	70.9	40.17
14							

Word

copying output data from MTLAB into WORD

10	1990	1	1	1	50.0100000000000	30.9000000000000
10	1990	1	1	2	70.9000000000000	40.1700000000000
10	1990	1	1	3	50.0100000000000	30.9000000000000
10	1990	1	1	4	70.9000000000000	40.1700000000000
10	1990	1	1	5	50.0100000000000	30.9000000000000
10	1990	1	1	6	70.9000000000000	40.1700000000000
10	1990	1	1	7	50.0100000000000	30.9000000000000
10	1990	1	1	10	70.9000000000000	40.1700000000000
10	1990	1	1	11	50.0100000000000	30.9000000000000
10	1990	1	1	12	70.9000000000000	40.1700000000000
10	1990	1	1	13	50.0100000000000	30.9000000000000
10	1990	1	1	14	70.9000000000000	40.1700000000000

10	1990	1	1	1	50.0100000000000	30.9000000000000
10	1990	1	1	2	70.9000000000000	40.1700000000000
10	1990	1	1	3	50.0100000000000	30.9000000000000
10	1990	1	1	4	70.9000000000000	40.1700000000000
10	1990	1	1	5	50.0100000000000	30.9000000000000
10	1990	1	1	6	70.9000000000000	40.1700000000000
10	1990	1	1	7	50.0100000000000	30.9000000000000
10	1990	1	1	10	70.9000000000000	40.1700000000000
10	1990	1	1	11	50.0100000000000	30.9000000000000
10	1990	1	1	12	70.9000000000000	40.1700000000000
10	1990	1	1	13	50.0100000000000	30.9000000000000
10	1990	1	1	14	70.9000000000000	40.1700000000000

See the program **tfscanf.m**

16- Output the results in term of graphs

Function example: Planck's black body radiation

Editor - C:\Users\ASUS\Documents\MATLAB\work\eblam.m

+12 | interplate_00.m | t_interp1.m | water_tpp_Incrop.m | fun07.m | read_oil_data.m | tfscanf.m | eblam.m

```
% spectral emissive power from a black body at temperature T (K)
% file_name =eblam.m
function y=eblam(lam,T)
% this function calcuate the emissive power emitted from a black body kept
% at T [Kilven]. Planck's law
% lam is the wavelength in mico-meter
c1=3.7405e-16
c2=0.0143879
lamm=lam*1.e-6
y=c1./(lamm.^5.*(exp(c2./(lamm.*T))-1))
% Notice the vector multiplication, division and power
end
```

$$E_{b,\lambda} = \frac{C_1}{\lambda^5\left(e^{(C_2/\lambda T)} - 1\right)}$$

If lam is a vector, then the output of the function is a vector of the same length

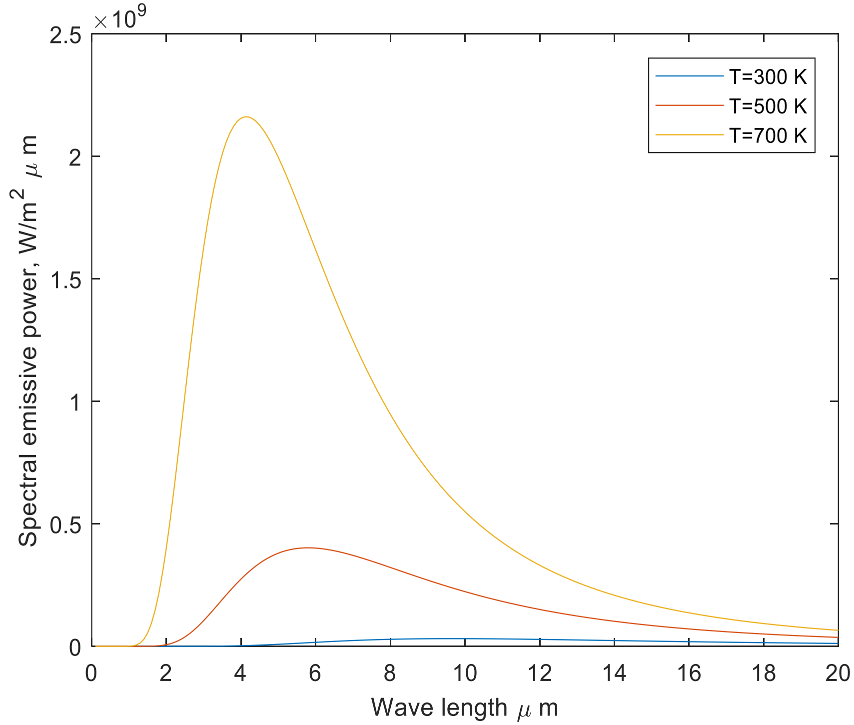
16- Output the results in term of graphs

Function example: Planck's black body radiation

Editor - C:\Users\ASUS\Documents\MATLAB\work\t_eblam.m

t_eblam.m | eblam.m | oil_tpp_Incrop.m | t_water_tpp_incr

```matlab
% This is atest for the function eblam.m
% file name t_eblam
clc
clear
T=[300 500 700] % body temp in [K]
x=0:0.1:20  % wavelength in [micro-meter]

q1=eblam(x,T(1))
q2=eblam(x,T(2))
q3=eblam(x,T(3))

plot(x,q1)
hold on
plot(x,q2)
hold on
plot(x,q3)
legend('T=300 K','T=500 K','T=700 K')
xlabel('Wave length \mu m')
ylabel('Spectral emissive power, W/m^2 \mu m')
```



Utilizing the function eblam to generate the behavior of the spectral emissive power with wavelength

17- Use of fopen and fprintf commands to output data

fprintf command can be used to output data to either the screen or a file using fopen command

Example File name

fprintf_example_00

```
clc
clear
% filename: fprintf_example_00
x = 0:.1:1;
        y = [x; exp(x)]; % y has 2 rows. Columns is 11
        [r,c]=size(y) % rows and columns of the vector y
        fid = fopen('exp1.txt','w');
        fprintf(fid,'%6.2f  %12.8f\n',y) % write in columns.
        %  Here each column has two values
        z=[1 2 3 4 5; 6 7 8 9 10; 11 12 13 14 15]
        zt=z'   % z transpose
        [rz,cz]=size(zt)
        % here the number of columns is 3 (i.e. cz=5
        fprintf(fid,'________________________\n')
        fprintf(fid,'%6.2f %6.2f %6.2f %6.2f %6.2f \n',zt)
        fclose(fid);
```

You can write
Type exp1.txt to see the file on the screen

Output file: exp1.txt

```
  0.00     1.00000000
  0.10     1.10517092
  0.20     1.22140276
  0.30     1.34985881
  0.40     1.49182470
  0.50     1.64872127
  0.60     1.82211880
  0.70     2.01375271
  0.80     2.22554093
  0.90     2.45960311
  1.00     2.71828183
________________________
  1.00   2.00   3.00   4.00   5.00
  6.00   7.00   8.00   9.00  10.00
 11.00  12.00  13.00  14.00  15.00
```

18-Using MATLAB fsolve function

If you have several equations in several unknowns, you can use fsolve built in MATLAB function to solve these equations

procedure

- Create an m-function where you calculate the values of the function at any given value of the variables say x
- Assume guess values for the solution in a vector say x0
- issue the commands (assuming we have two equations in two unknows

```
x0=[1 3] % initial guess for the solution
x= fsolve(@(x) fun_name(x),x0,options)
```

x is the solution vector for the given equations

18-Using MATLAB fsolve function

simple example

Two equations in two unknowns to be solved for x1 and x2

$$x_1 + 2\,x_1 x_2 + x_2^2 = 23 \qquad 5\,x_1 + x_1^2 - 3\,x_2 = 5$$

create an m-function with function equal zeros in the right hand side i.e.

$$F(1) = x_1 + 2x_1 x_2 + x_2^2 - 23 \qquad F(2) = 5\,x_1 + x_1^2 - 3x_2 - 5$$

```
function F=fun05(x)
% Here we have two functions F(1) and F(2)
% functions of x1 and x2 to be solved together
% to find x1 and x2 when F(1) and F(2) are zeros
% Notice that if one operates this function with % two
different values of x1&x2,the function calculate F(1) and F(2)
 F(1)=x(1)+2*x(1)*x(2)+x(2)^2-23
 F(2)=5*x(1)+x(1)^2-3*x(2)-5
end
```

```
x0=[3 5]
Q=fun05(x0)
Q=[35 4]  % This means that F(1)=35 and F(2)=4 when
x1=3 and x2=5
```

18-Using MATLAB fsolve function

```
function F=fun05(x)
% Here we have two functions F(1) and F(2)
% functions of x1 and x2 to be solved together
% to find x1 and x2 when F(1) and F(2) are zeros
% Notice that if one operates this function with two
different values of x1 and
% x2, the function calculate F(1) and F(2)
 F(1)=x(1)+2*x(1)*x(2)+x(2)^2-23
F(2)=5*x(1)+x(1)^2-3*x(2)-5
end

% test for fsolve with function func05
clc
clear
x0=[1 5]
F0=fun05(x0)
[x F1]= fsolve(@(x) fun05(x),x0)
% if you do not want to see the function values
% just write x=fsolve (@(x)fun05(x),x0)
```

x=[2 3] % this the solution of the above two equations

19-Additional topics

- More about Input and output for MATLAB
- More about graphics
- More useful functions such as ezplot, fzero, feval, interpt, spline, quard, etc
- In-line functions
- Anonymous functions

Water properties at saturation

```
function [V]=water_tpp_Incrop(tempC)
% [V]=water_tpp_Incrop(tempC)
% tempC in deg. C
% V is a vector of properties
% Water prpoerties at saturation from Incropera heat transder book 7th
% edition
% V(1)=P [in bar], V(2)=v_f, V(3)=v_g,V(4)=h_fg_V(5)=Cp_f,V(6)=Cp_g,V(7)=mu_f,
%V(8)=mu_g, V(9)=k_f, V(10)=k_g,V(11)=pr_f,V(12)=Pr_g,V(13)=sigmma,
%V(14)=beta
tempK=tempC+273.15


T=[273.15 275 280 285 290 295 300 305 310 315 320 325 330 335 340 345 350 ...
    355 360 365 370 373.15 375 380 385 390 400 410 420 430];
% T in Kelvin
P=[0.00611 0.00697 0.00990 0.01387 0.01917 0.02617 0.03531 0.04712 0.06221 ...
    0.08132 0.1053 0.1351 0.1719 0.2167 0.2713 0.3372 0.4163 0.5100 0.6209 ...
    0.7514 0.9040 1.0133 1.0815 1.2869 1.5233 1.794 2.455 3.302 4.370 5.699];
% P in bars, saturated pressure [bar]
vf=1.0E-3*[1.000 1.000 1.000 1.000 1.001 1.002 1.003 1.005 1.007 1.009 ...
    1.011 1.013 1.016 1.018 1.021 1.024 1.027 1.030 1.034 1.038 1.041 ...
```

Q=water_tpp_Incrop(30)

Q is a vector of length 14 for al water properties at saturation

For example (liquid water specific heat)
Cp_l=Q(5)=4.1783 kJ/kg.K

Dry air properties

```
function [V]=dryair_tpp_Incrop(tempC)
% [V]=dryair_tpp_Incrop(tempC)
% dryair prpoerties at atmospheric pressure from Incropera heat transder book 7t
% edition
% [V]=dryair_tpp_Incrop(tempC)
% V(1)=rhoi;
% V(2)=Cpi;
% V(3)=mui;
% V(4)=ki;
% V(5)=alphai;
% V(6)=Pri;
tempK=tempC+273.15

T=[100 150 200 250 300 350 400 450 500 550 600 650 700 750 800 850 900 950 ...
    1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 ...
    2300 2400 2500 3000];
% T in Kelvin
```

Example:
A=dryair_cpp_Incrop(30)
Will give a vector of length 6

The specific heat of air at 30 C
is rho_a=A(2)=1.0071 kJ/kg.K

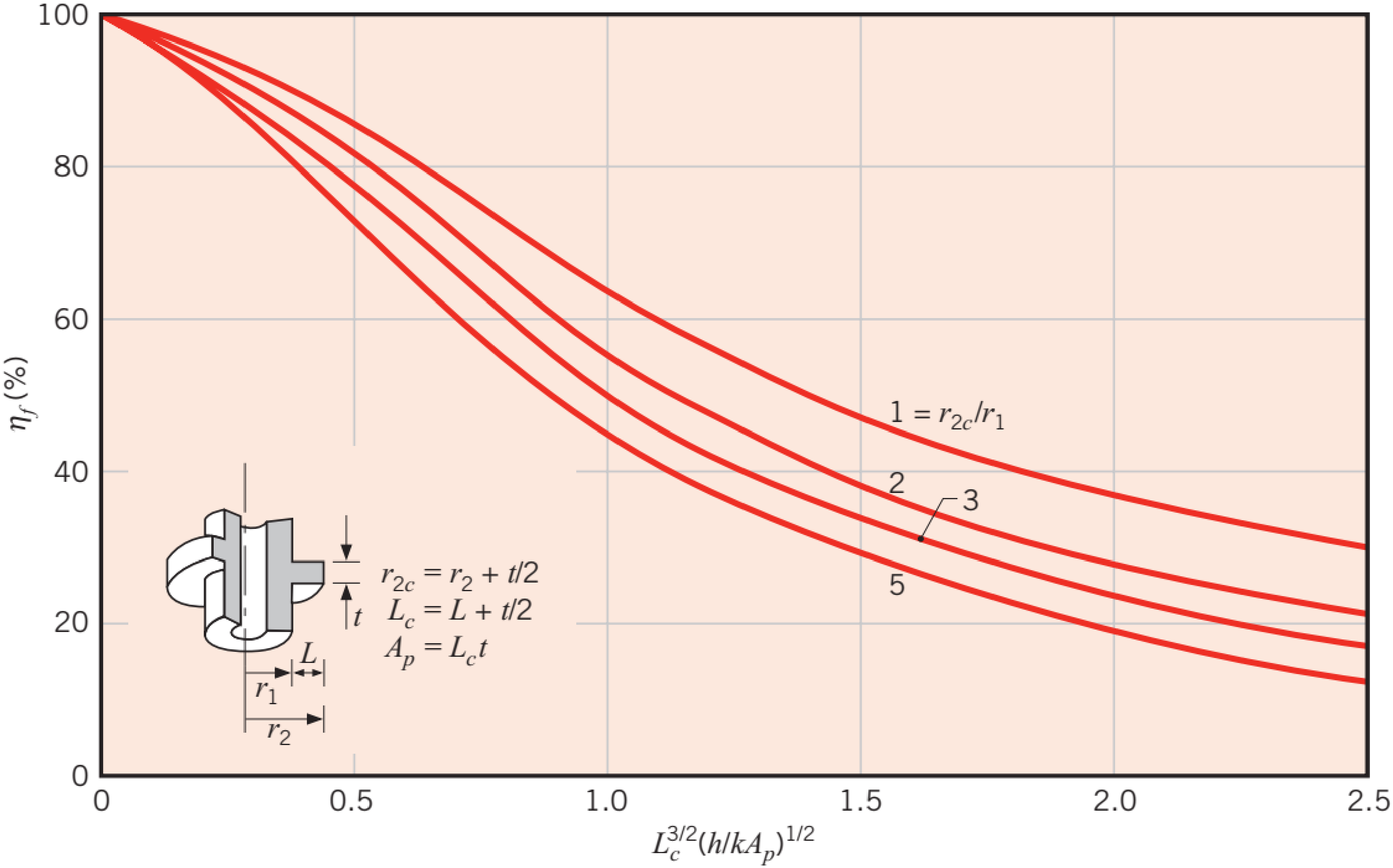
Friction factor

```
function y=frictionf_2020(Re)
% y=frictionf_2020(Re)
% with Re as input this function calculate the friction factor for
% either laminar or turbulent flows
% using the information from Incropera
% for laminar and turbulent in smooth pipe

if Re < 2300

disp([' from frictionf_2020 Flow is laminar Re= ',num2str(Re)])
    ff=64/Re
elseif Re >= 2300
    disp([' from frictionf_2020 Flow is turbulent  Re= ',num2str(Re)])

ff=(0.79*log(Re)-1.64)^-2
end
y=ff
end
```

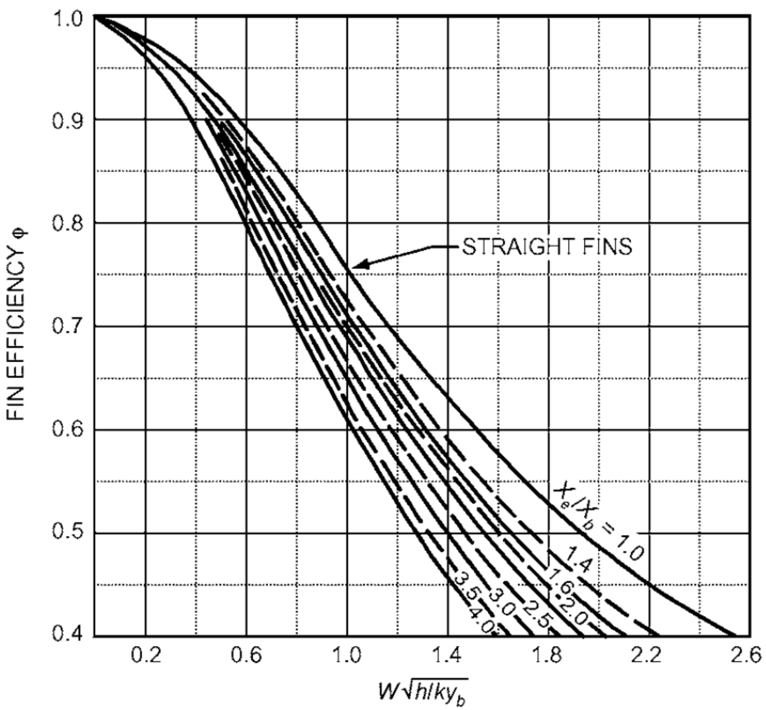
f1=friction_2020(1500) will give
f1=0.0427

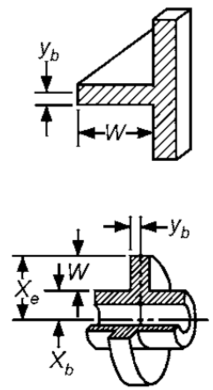
f2=friction_2020(5000) will give
f2= 0.0386

Circular fin efficiency



Ref: ASHRAE
Book of Fundamentals 2013



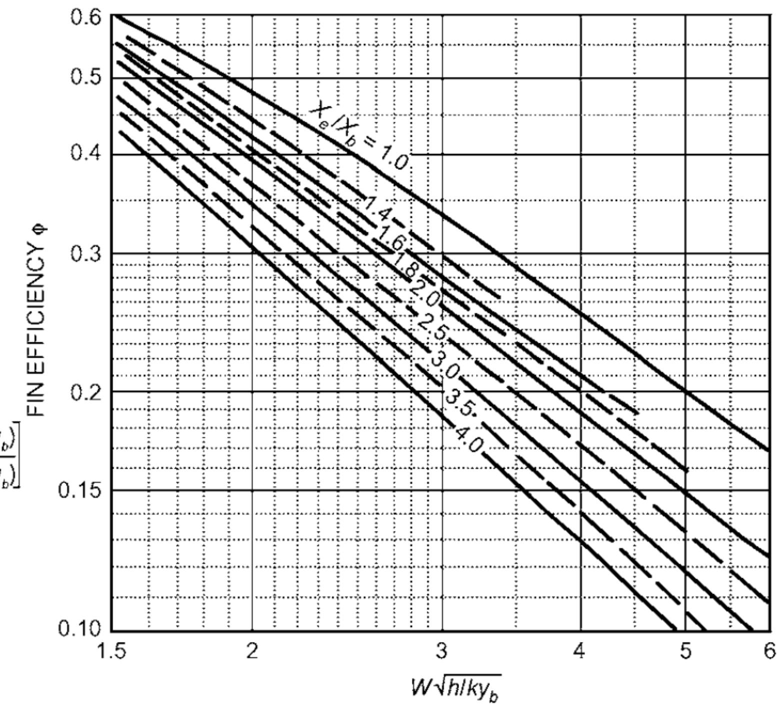


$$\varphi = \frac{2}{u_b[1-(u_e/u_b)^2]}\left[\frac{I_1(u_b)-\beta K_1(u_b)}{I_0(u_b)+\beta K_0(u_b)}\right]$$

$$\beta = I_1(u_e)/K_1(u_e)$$

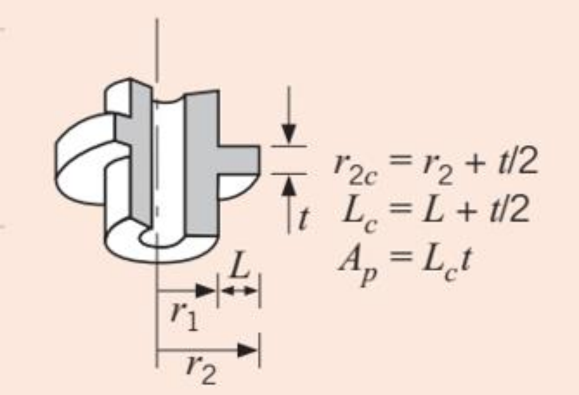
$$u_b = \frac{W\sqrt{h/ky_b}}{(X_e/X_b - 1)}$$

$$u_e = u_b(X_e/X_b)$$



MATLAB program for circular fin efficiency

```
function y=fin_eff_cir(rratio,x)
% filenme fin_eff_cir(rratio,x)
% function to calculate the fin efficiency for
% a circular fin on circular pipe
% the input parameter are % r2_o_r1 ratio of r2/r1
% the paramter W*sqrt(h/ky)=x
% where
% W is r2-r1
% y is half of the fin thickness i.e. t/2
% h is the heat transfer coeffcient
% y is half the fin thickness
% k fin thermal conductivity
% all units are in SI
% the reference for this is ASHRAE Handbook of fundamentals
% the heat transfer chapter
% The function uses the Bessel functions I and K
%
```



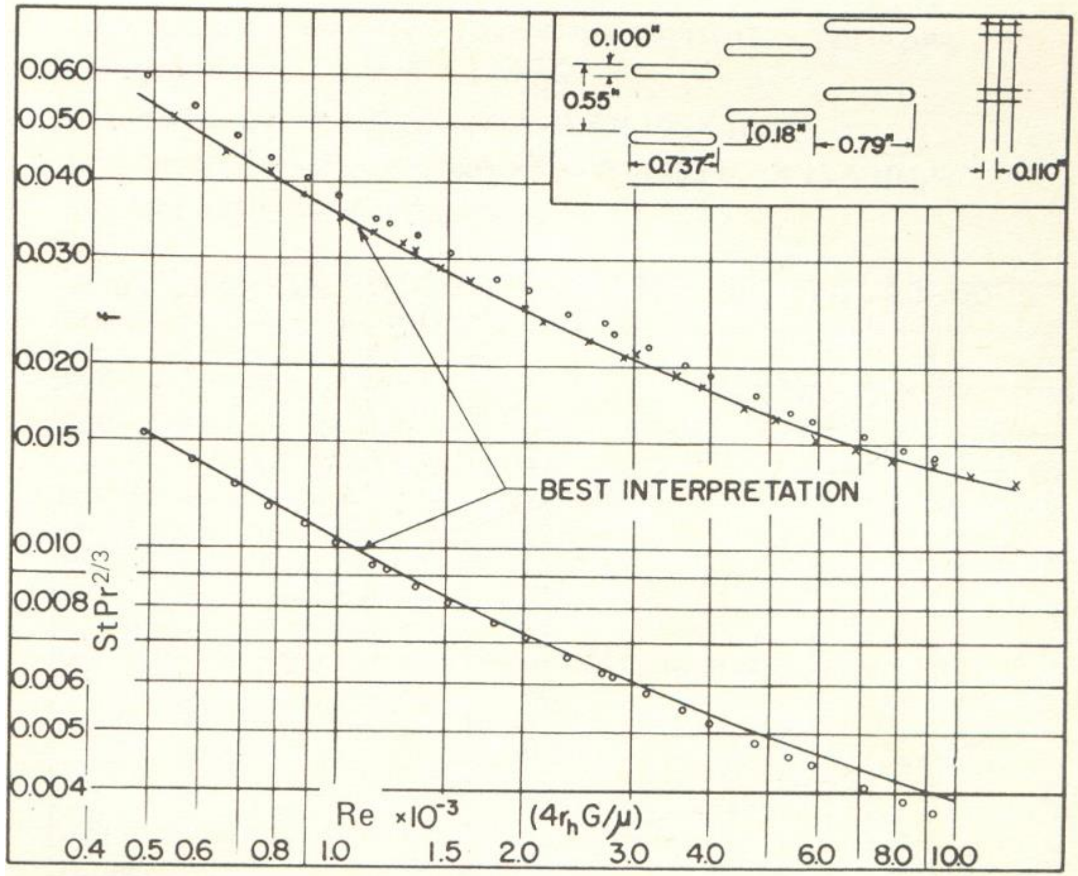
$$x = L\sqrt{\frac{2h}{kt}}$$

$$rratio = \frac{r_2}{r_1}$$

eta_f=fin_eff_cir(1.5,0.5)=0.91

Interpolation example

Fig. 10-94 Finned flat tubes, surface 9.1 – 0.737-S.



Fin pitch = 9.1 per in = 358 per m

Flow passage hydraulic diameter, $4r_h$ = 0.01380 ft = 4.206×10^{-3} m

Fin metal thickness = 0.004 in, copper = 0.102×10^{-3} m

Free-flow area/frontal area, σ = 0.788

Total heat transfer area/total volume, α = 224 ft^2/ft^3 = 735 m^2/m^3

Fin area/total area = 0.813

Data from Kays & London for tube-fin compact heat exchanger

Compact HX	continuous fins on flat tube	
Surface	9.1-737S	

Re	j_H	f
500	0.015	0.054
600	0.014	0.048
800	0.012	0.04
1000	0.0105	0.036
1500	0.0082	0.028
2000	0.0072	0.025
3000	0.006	0.0205
4000	0.0054	0.018
6000	0.0047	0.0151
8000	0.0042	0.0145
10000	0.0039	0.014

MATLAB program for interpolation

```
Re=[500 600 800 1000 1500 2000 3000 4000 6000 8000 10000];
j_H=[0.015 0.014 0.012 0.0105 0.0082 0.0072 0.006 0.0054 0.0047 0.0042 0.0039];
f=[0.054 0.048 0.04 0.036 0.028 0.025 0.0205 0.018 0.0151 0.0145 0.014];

% Notice that the data is limited by Re 400 and Re 10,000
% you can use 'spline' or 'anymehod' and e'extrap' to make an extrapoltion
% if you omit method, no resuls will be given for Re <500 and Re > 10000

j_1=interp1(Re,j_H,300,'spline')  % spline will do extrapolation
j_2=interp1(Re,j_H,12000,'linear','extrap')  %  extrapolate
j_3=interp1(Re,j_H,300,'pchip') % will do extrapolation
```