MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY

(AUTONOMOUS Institution – UGC Govt of India)
Permanently Affiliated to JNTUH, ACCREDITED by AICTE-NBA, NAAC
Maisammaguda, Dhulapally post, Secunderabad – 500014 (A. P.)

DEPARTMENT OF AERONAUTICAL ENGINEERING

MTECH AEROSPACE ENGINEERING

I Year I Sem

Aerodynamics and Simulation Lab

LAB MANUAL

MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY

(UGC Autonomous Institution)
Maisammaguda, Dhulapally post, Secunderabad – 500100

CERTIFICATE

Department of Aeronautical Engineering certified that this is a bonafide record of work done in this Laboratory during the year 2024 – 2025 by Mr. /Miss : _____ Reg .No: _____ of M.Tech. (AEROSPACE ENGINEERING) I Year I Semester in the _____.

Staff Incharge HOD/ANE

Internal Examiner External Examiner

INDEX

EXPERIMENT -1

INTRODUCTION TO MATLAB SOFTWARE

Aim: To study and solve following types of problems using MATLAB

(a) Simple arithmetic operations using command window.

(b) Creating arrays and mathematical operations with arrays(dot and cross product, inverse, transpose, Eigen values, solutions of linear equations).

(c) Creating simple script file.

Equipment and material needed: Core 2 duo processor with 1GB RAM and MATLAB software Version 2008a or higher. In addition, thorough understanding of chapter 1, 2 and 3 of Reference book 1(MATLAB: An Introduction with Applications by Amos Gilat).

Brief introduction about MATLAB and algorithm: The MATLAB® is a very popular language for technical computing by students, engineers, and scientists in universities, research institutes, and industries all over the world. The software is popular and easy to use. A brief on few features are explained below. For complete coverage you should go through all the chapters of Reference 1. These will also be explained to you during the LAB.

It is utmost important that you bring this reference book during the LAB for ease of understanding and quick reference. It will be of help if the concerned faculty keeps the copy of the book in the Lab for their reference and demonstration.

Command Window: Command Window is used to enter variables and to run functions and M-file scripts.

Command History: Statements entered in the Command Window are logged in the Command History. From the Command History, one can view and search for previously run statements, as well as copy and execute selected statements. One can also create an M-file from selected statements.

Current Directory: A quick way to view or change the current directory is by using the current directory field in the desktop toolbar

Workspace: The MATLAB® workspace consists of the set of variables built up during a MATLAB session and stored in memory. Variables are added to the workspace by using functions, running M-files, and loading saved workspaces.

Editor: Editor is used to create and debug M-files, which are programs we write to run MATLAB® functions. The Editor provides a graphical user interface for text editing, as well as for M-file debugging. To create or edit an M-file one use File > New or File > Open, or use the edit function.

Keep the following points in mind when working with MATLAB.

- Upper and lower-case characters are not equivalent (MATLAB is case sensitive).
- Typing the name of a variable will cause MATLAB to display its current value.
- A semicolon at the end of a command suppresses the screen output.
- MATLAB uses both parentheses, (), and square brackets, [], and these are not interchangeable.
- The up arrow and down arrow keys can be used to scroll through previous commands. Also, an old command can be recalled by typing the first few characters followed by up arrow.
- One can type help topic to access online help on the command, function or symbol topic.
- You can quit MATLAB by typing exit or quit.
- First character of the saved M-file should be a letter and not a number. So do not save file using your roll no in the beginning of the file name.
- Having entered MATLAB, you should work through this tutorial by typing in the text that appears after the MATLAB prompt, >>, in the Command Window. After showing you what to type, we display the output that is produced. Faculty will demonstrate and help you in thefollowing tutorial.

  Defining Scalar Variable: Do not use reserved key words as variables (e.g. break,else, while etc.).

```
>> x=15;
X=15
>> x=3*x-12
X=
33
>>E=sin(x)^2+cos(x)^2
E=
1
>>
```

Creating Arrays and Mathematical Operations with Arrays: The array is a fundamental form that MATLAB uses to store and manipulate data. It is a list of numbers arranged in rows and/or columns. The simplest array is a row or a column of numbers(one-dimensional). A more complex array (2-D) is a collection of numbers arranged in rows and columns. In science and engineering 1-D arrays frequently represents vectors, and two-dimensional arrays often represent **matrices**. Faculty will demonstrate to you how to create and perform mathematical operations on arrays.

-Create an array from given population data in Table 2-1: (Referene-1 text book chapter-2).

Table2-1: Population data

Year	1984	1986	1988	1990	1992	1994	1996
Population(millions)	127	130	136	145	158	178	211

>>yr= [1984,1986,1988,1990,1992,1994,1996]

yr=

1984 1986 1988 1990 1992 1994 1996

pop=[127,130,136,145,158,178,211]

-create a vector by specifying first term,the spacing and the last term.

Example:

> X=[1:2:13]

X

1 3 5 7 9 11 13

-create a vector using linear spacing by specifying the first and last terms, and the number of terms:

Va= linspace(0,8,6)

Va=

0 1.6000 3.2000 4.8000 6.4000 8.0000

Some of the useful commands for matrix manipulations are tabulated below with examples.

Table-1: Useful functions on Matrix operations

S.No	**Command**	**Description**
1	A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]	Matrix
2	A'	Transpose of Matrix
3	diag(A)	Diagonal elements of matrix
4	A(4,2)	the number in the fourth row and second column
5	100:-7:50	a row vector containing the integers from 100 to 50 with decrement of 7

6	sum(A(1:4,4))	computes the sum of the fourth column
7	Z=zeros(2,4)	2x4 matrix with all zeros
8	C=ones(1,3)	1x3 matrix with all ones
9	A(:,2) = []	Deleting second row
10	E = A([1,1,1],:)	copies the first row of *A three* times to create a new matrix
11	det(A)	Determinant of matrix
12	X = inv(A)	Inverse of a matrix
13	e = eig(A)	Eigen values of a matrix
14	eye(3x3)	Creates 3x3 Identity matrix
15	linspace(a,b,n)	creates a row vector of *n* regularly spaced elements between *a* and *b*
16	function [out1, out2, ...] = funname(in1, in2, ...)	*out1, out2, ...,* are the function outputs, *in1, in2, ...* are its inputs and *funname* is the function name then the function can be called in the command window or in other m-files.

Creating script file (M-file): A script file is a list of MATLAB commands, called a program that is saved in a file. When the script file is executed(run), MATLAB executes the commands. You will be demonstrated by the faculty, how to create, save and run a simple script file in which commands are executed in order in which they are listed, and in which all the variables are defined within the script file. On similar line we can write functions for various applications.

Result: -Students should be able to formulate and solve simple problems using various functions including Arrays and Matrices available in MATLAB.

EXERCISES

Solve the flowing problems given in Chapter4 of Reference book 1.

1. Problem 8

2. Problem 16

3. Problem 13

4. Problem 25

5. Problem 28

6. Problem 29

7. Compute the array and matrix product of $A = \begin{bmatrix} 8 & 7 & 11 \\ 6 & 5 & -1 \\ 0 & 2 & -8 \end{bmatrix}$, $B = \begin{bmatrix} 2 & 1 & 2 \\ -1 & 6 & 4 \\ 2 & 2 & 2 \end{bmatrix}$

8. Find a solution to the following set of equations:

$x + 2y + 3z = 12$
$-4x + y + 2z = 13$
$9y - 8z = -1$

What is the determinant of the coefficient matrix?

9. Write a function ‘altitude’ which takes static pressure in millibar as input argument and computes the pressure altitude in meters using standard atmosphere. (Kindly refer to any book on aerodynamics for relation between pressure and altitude in standard atmosphere.

EXPERIMENT -2

PROGRAMS FOR TWO-DIMENSIONAL (2-D) AND THREE-DIMENSIONAL (3-D)PLOTTING

Aim: To study the programs for creating 2D & 3D Plots.

Equipment needed: Core 2 duo processor with 1GB RAM and MATLAB software Version 2008a or higher

Theory:

Plotting: Plots are very useful tool for presenting information. This is true in any field, but specially in science and engineering, where MATLAB is mostly used. MATLAB has many commands that can be used for creating different types of plots. These include standard plots with linear axes, plots with logarithmic and semi-logarithmic axes, polar plots, 3-D contour surfaces and mesh plots, and many more. In this experiment you will learn how MATLAB can be used to create and format many types of 2-D and 3-D plots.

2-D Plot of a Function. In many situations there is a need to plot a given function. This can be done by using the 'plot' or the 'fplot' command. In order to plot a function y =f(x) with the plot command, the user needs to first create a vector of x for the domain over which the function will be plotted. Then a vector y is created with the corresponding values of f(x) by using element-by-element calculations as explained in Experiment number 1. Once the two vectors are defined, they can be used in the plot command. The fplot command plots a function with the form y=f(x) between specified limits. The command has the form

fplot('function',limits,'line specifiers')

'function' can be typed directly as a string inside the command. For example if the function that is being plotted is f(x) = 8x^2+5 cos(x),it is typed as:'8*x^2+5*cos(x)'. the function can include MATLAB built-in functions and functions that are created by the users.

Plotting multiple graphs in the same plot. In many situations there is a need to make several graphs in the same plot. There are three methods to plot multiple graphs in one figure. One is by using the plot command, the second by using the hold on and hold off and the third is by using the line command.

Three-Dimensional Graphics: MATLAB provides a variety of functions to display 3-D data. Some functions plot lines in 3-D, while others draw surfaces and wire frames.In addition, color can be used to represent a fourth dimension. When color is used in this manner, it is called

pseudo color, since color is is not inherent or natural property of the underlying data in the way that color in a photograph is natural characteristic of the image.

(a) **LINE PLOTS**. General format of the command is plot3(x1,y1,z1,S1,x2,y2,z2,S2,…) where xn,yn and zn are vectors or matrices and Sn are optional character strings specifying color, marker symbol,and or line style.

(b) **MESH PLOT**. MATLAB defines a mesh surface by the z-coordinates of points above a rectangular grid in the x-y plane. It formats a mesh plot by joining adjacent points with straight lines. The result looks like a fishing net with knots at the data points.

(c) **SURFACE PLOTS**. A surface plot is like a mesh plot, except that the spaces between the lines called patches are filled in. Plots of this type are generated using the surf function.

All the above features will be demonstrated to you by the faculty with examples discussed below. Some important commands for plotting are tabulated below for quick reference.

Table 2: Important plot commands in MATLAB

S.No	Command	Description
1	plot(*x, y*)	Plots the variation of y with respect to x
2	xlabel('x')	Gives the label for x axis
3	ylabel('cos(x)')	Gives the label for y axis
	title('plot name')	Gives the name of plot
4	fplot ('function string,' [*x*start, *x*end])	The function fplot gets around our choice of interval used to generate the plot, and instead decides the number of plotting points to use for us.
5	plot(t,f,t,g,'--')	To plot multiple functions, call the plot(*x, y*) command with multiple pairs *x, y* defining the independent and dependent variables used in the plot in pairs. This is followed by a character string enclosed in single quotes to tell us what kindof line to use to generate the second curve
6	'Linewidth'2	Increases thickness of curve
7	legend('sinh(x)','cosh(x)')	To name the curves
8	plot(x,y,'r',x,z,'b')	Differentiates the curves with colors r-red, b-blue, g-green, k-black, w-white, y-yellow, m-magenta, c-cyan.
9	axis([xmin xmax ymin ymax])	Plot range
10	subplot(1,2,1)	Creates the plot with 2 panes and 1 row, and that this particular plot will appear in the first pane
11	polar (theta, r)	Creates Polar plots
12	bar(x,y)	Creates bar chart
13	[x,y] = meshgrid(–5:0.1:5,–3:0.1:3);	Generate a matrix of elements that give the range over *x* and *y* we want to use along with the specification of increment in each case.

14	mesh(x,y,z)	3D plots
15	surf(x,y,z)	Shaded surface plots

Example 1: Plotting multiple graphs in the same plot using fplot and hold command.

Plot three sine waves with different phases. For the first, use a line width of 2 points. For the second, specify a dashed red line style with circle markers. For the third, specify a cyan, dash-dotted line style with asterisk markers.

Code:

```
fplot(@(x) sin(x+pi/5),'Linewidth',2);
holdon
fplot(@(x) sin(x-pi/5),'--or');
fplot(@(x) sin(x),'-.*c')
holdoff
```

Result: -The plot is shown in Fig 2.1**.** The default limit for x : -5,5

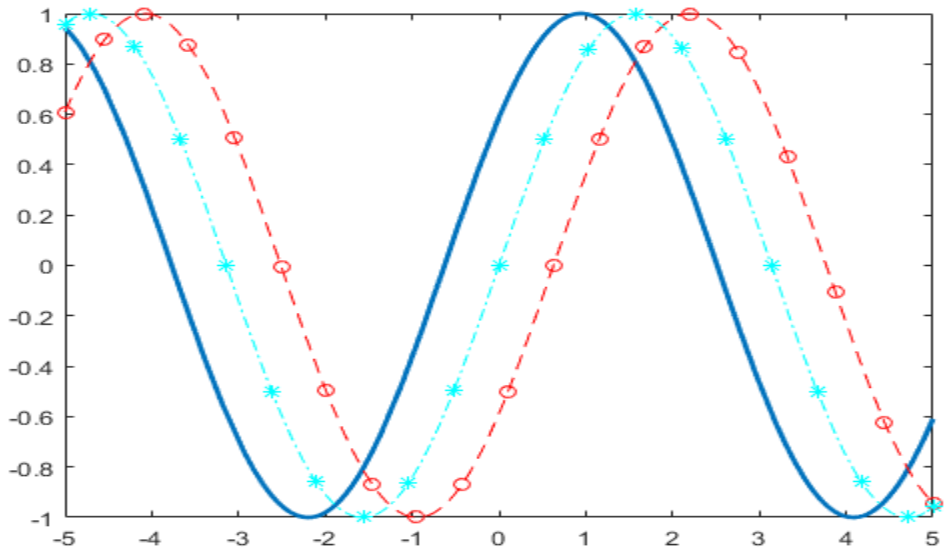


Fig 2.1

Example 2: Polar plot of a function r=3 cos^2(0.5θ) + θ for 0<= θ<= 2*pi

```
t= linspace (0,2*pi,200);
r=3*cos(0.5*t).^2+t;
polar(t, r)
```

Result: polar plot is shown in Fig 2.2

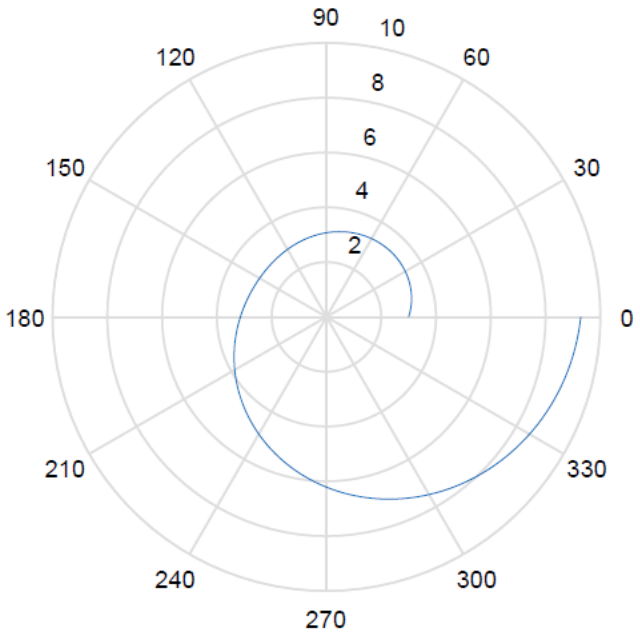
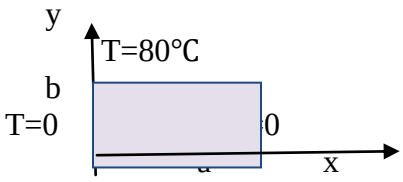


Fig 2.2 Polar plot

Example 3 for 3-D plot. Three sides of a rectangular plate (a=5m, b=4m) are kept at temperature of 0 deg C and one side is kept at a temperature T1=80 deg C, as shown in the figure. Determine and plot the temperature distribution T(x,y) in the plate.



T=0

Solution. The temperature distribution, T(x,y) in the plate can be determined by solving the 2-D heat equation. For given boundary conditions T9x,y) can be expressed analytically by a Fourier series:

$$T(x,y)=\frac{4T_1}{\pi}\sum_{n=1}^{\infty}\frac{\sin\left[(2n-1)\frac{\pi x}{a}\right]}{(2n-1)}\frac{\sinh\left[(2n-1)\frac{\pi y}{a}\right]}{\sinh\left[(2n-1)\frac{\pi b}{a}\right]}$$

A program in a script file that solves the problem is listed below. The program follows these steps:

(a) Create an X,Y grid in the domain $0 \leq x \leq a$ and $0 \leq y \leq b$. The length of the plate, a, is divided into 20 segments, and the width of the plate, b, is divided into 16 segments.
(b) Calculate the temperature at each point of the mesh. The calculation are done point by point using a double loop. At each point the temperature is determined by adding k terms of the Fourier series.
(c) Make a surface plot of T.

```
% 3-D plot for heat equation PLMM lab
%
% script file
 a=5;
 b=4;na=20;nb=16;T0=80;k=5;
x=linspace(0,a,na);
y=linspace(0,b,nb);
[X,Y]=meshgrid(x,y);
for i=1:nb
for j=1:na
   T(i,j)=0;
for n=1:k
   ns=2*n-1;      % third loop, n, is the
% term of the Fourier series,% k is the number of  %terms
T(i,j)=T(i,j)+sin(ns*pi*X(i,j)/a).*sinh(ns*pi*Y(i,j)/a)/(sinh(ns*pi*b/a)...
*ns);
end
  T(i,j)=T(i,j)*4*T0/pi;
end
end
mesh(X,Y,T)
xlabel( 'x  (m)'); ylabel('y  (m) ');
zlabel(' T  ( ^0C) ')
```

Program was executed with two different values of k (5 &50). The mesh plots are shown in each case in the figure 2.3 and 2.4. the temperature should be uniformly at y=4 m. Note the effect of k on the accuracy at y=4m.

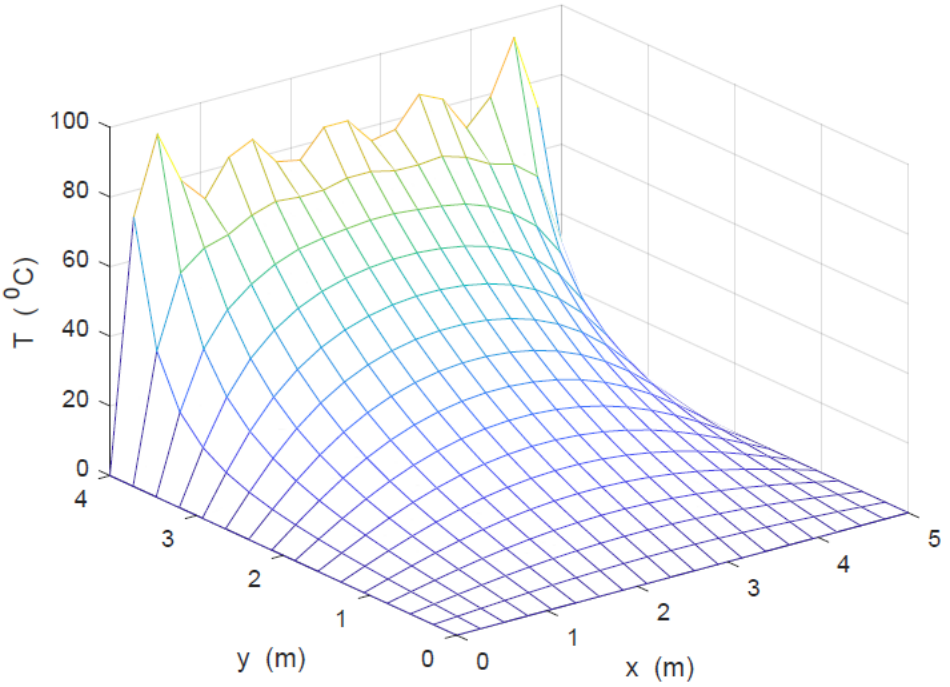


Fig 2.3: for k=5;

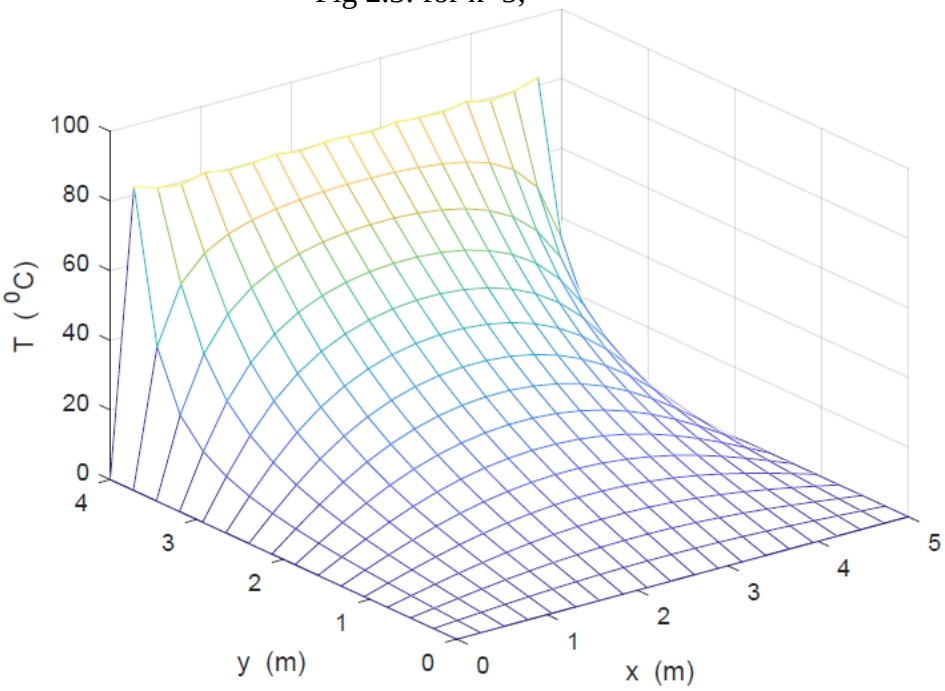


Fig 2.4: For k=50

EXERCISES

1. Acycloid is a curve traced by a point on a circle that rolls along a line. The parametric equation of a cycloid is given by

x=r(t-sint) and y=r(1-cost)

Plot a cycloid with r=1.5 and $0 \leq t \leq 4\pi$

2. Solve the problem No 24 relating to NACA airfoil given in chapter 5 of reference book 1.

3. Solve the problem no 31 relating to simply supported beam given in Chapter 5 of reference 1

4. Solve the problem no 26 relating to vibrations of chapter 5 of reference 1.

5. Solve the problem no 19 of chapter 5 of reference 1 relating to tensile strength**.**

6. Solve the problem no 14 of chapter 10 of reference 1 relating to defect in crystal lattice.

7. Solve problem No 19 of chapter 10 of reference book 1.

8. In a study of the effect of various factors on the growth performance of activated sludge, the oxygen uptake rate was measured at various temperatures.

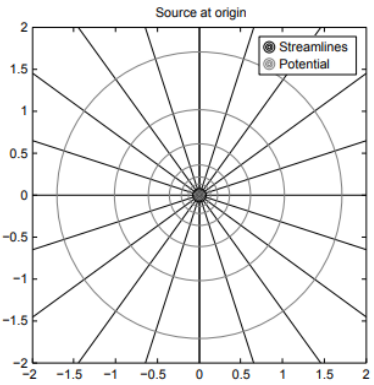
Temperature °C	Oxygen uptake rate grams oxygen per gram dry Weight
5	0.01
10	0.04
15	0.10
20	0.20
25	0.25
30	0.28
35	0.30
40	0.25
45	0.02

Write a script M-file that generates a plot of these data, including title, labels, and grid. Print the generated graph.

EXPERIMENT 3

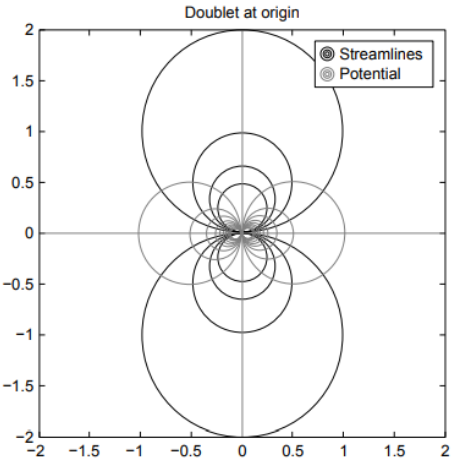
Development of numerical solver code to generate basic potential flows: source, sink and doublet.

```
sig = 1; % Source strength
% GRID:
x = -2:.02:2; y = -2:.02:2;
for m = 1:length(x)
for n = 1:length(y)
xx(m,n) = x(m);
yy(m,n) = y(n); % Velocity potential function:
phi Source(m,n) = (sig/4/pi) * log(x(m)^2+(y(n)+.01)^2); % Stream function:
psi Source(m,n) = (sig/2/pi) * atan2(y(n),x(m));
end
end % Plots
% Source at origin of coordinate system:
contour(xx,yy,psi Source,[-.5:.05:.5],'k'),hold on
contour(xx,yy,phi Source,10,'r')
legend('streamlines' ,'potential')
title(' Source at origin')
axis image,hold off
```




```
mu = 1; % Doublet strength
% GRID:
x = -2:.02:2;
y = -2:.02:2;
for m = 1:length(x)
for n = 1:length(y)
xx(m,n) = x(m);
yy(m,n) = y(n); % Velocity potential function:
```

```
phi Doublet(m,n) = mu * x(m)/(x(m)^2+(y(n)+.01)^2); % Stream function:
psi Doublet(m,n) = - mu * y(n)/(x(m)^2+(y(n)+.01)^2);
end
end
% Plots
% Doublet at origin of coordinate system:\ figure(4)
contour(xx,yy,psi Doublet,[-2:.5:2],'k'),hold on
contour(xx,yy,phi Doublet,[-10:1:10],'r')
legend('streamlines' ,'potential')
title(' Doublet at origin')
axis image,hold off
```



EXERCISE – 4

Development of potential flow solver code to simulate flow over cylinder with source-panels.

```
clear all;
%Circular cylinder example, radius 2, 35 panels
npanels=35;r=2;winf=1;
z=r*exp(i*[2*pi/npanels:2*pi/npanels:2*pi]);
a=[1:npanels];b=[2:npanels 1];
dzds=(z(b)-z(a))./abs(z(b)-z(a));

eps=0.0001;
zc=(z(a)+z(b))/2-i*eps*(z(b)-z(a)); %control points

cm=zeros(npanels);
for m=1:npanels
    cm(:,m)=log((zc(m)-z(a))./(zc(m)-z(b)))/2/pi./dzds(a)*dzds(m);
end
res=imag(-winf*dzds);
q=res/imag(cm);

ut=real(q*cm+winf*dzds);
cp=1-ut.^2/abs(winf).^2;
figure
plot(angle(zc)*180/pi,cp);
```

EXPERIMENT -5

PROGRAM TO GENERATE AIRFOIL COORDINATES

Aim: Write a code for generating airfoil coordinates.

Equipment needed: Core 2 duo processor with 1GB RAM and MATLAB software Version 2008a or higher or Sci-lab.

Theory:

NACA 4- digit airfoil specification

This NACA airfoil series is controlled by 4 digits e.g. NACA 2412, which designate the camber, position of the maximum camber and thickness. If an airfoil number is NACA MPXX e.g.NACA 2412
then:

- M is the maximum camber divided by 100. In the example M=2 so the camber is 0.02 or 2% of the chord
- P is the position of the maximum camber divided by 10. In the example P=4 so the maximum camber is at 0.4 or 40% of the chord.
- XX is the thickness divided by 100. In the example XX=12 so the thickness is 0.12 or 12% of the chord.

The NACA airfoil section is created from a camber line and a thickness distribution plotted perpendicular to the camber line.

The equation for the camber line is split into sections either side of the point of maximum camber position (P). In order to calculate the position of the final airfoil envelope later the gradient of the camber line is also required. The equations are:

	Front ($0 \le x < p$)	Back ($p \le x \le 1$)
Camber	$y_c = \frac{M}{P^2}\left(2Px - x^2\right)$	$y_c = \frac{M}{(1-P)^2}\left(1 - 2P + 2Px - x^2\right)$
Gradient	$\frac{dy_c}{dx} = \frac{2M}{P^2}(P - x)$	$\frac{dy_c}{dx} = \frac{2M}{(1-P)^2}(P - x)$

The thickness distribution is given by the equation:

$$y_t = \frac{T}{0.2}\left(a_0 x^{0.5} + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4\right)$$

Where:

$a_0 = 0.2969$ $a_1 = -0.126$ $a_2 = -0.3516$ $a_3 = 0.2843$

$a_4 = -0.1015$ or -0.1036 for a closed trailing edge

- The constants a0 to a4 are for a 20% thick airfoil. The expression T/0.2 adjusts the constants to the required thickness.
- At the trailing edge (x=1) there is a finite thickness of 0.0021 chord width for a 20% airfoil. If a closed trailing edge is required the value of a4 can be adjusted.
- The value of yt is a half thickness and needs to be applied both sides of the camber line.

Using the equations above, for a given value of x it is possible to calculate the camber line position Yc, the gradient of the camber line and the thickness. The position of the upper and lower surface can then be calculated perpendicular to the camber line.

$$\theta = \operatorname{atan}\left(\frac{dy_c}{dx}\right)$$

Upper Surface $x_u = x_c - y_t \sin(\theta)$ $y_u = y_c + y_t \cos(\theta)$

Lower Surface $x_l = x_c + y_t \sin(\theta)$ $y_l = y_c - y_t \cos(\theta)$

The most obvious way to to plot the airfoil is to iterate through equally spaced values of x calculating the upper and lower surface coordinates. While this works, the points are more widely spaced around the leading edge where the curvature is greatest and flat sections can be seen on the plots. To group the points at the ends of the airfoil sections a cosine spacing is used with uniform increments of β

$$x = \frac{(1-\cos(\beta))}{2} \quad \text{where: } 0 \le \beta \le \pi$$

Algorithm:

i. Give the specifications of airfoil.
ii. Give the number of points required.
iii. Write code for camber and gradient equations
iv. Equations for theta and thickness distribution
v. Equations to generate upper curve and lower curve coordinates separately

MATLAB code for generating 4-digit airfoil: Airfoil number is required to be entered as vector as [x x x x]. Then program asks for chord length. Co-ordinates are computed and plotted.

```
    % code for 4-digit NACA airfoil
    % Notes:
    % 1) That this code Plots NACA 4 Digit Series ONLY.

    % Airfoil Equation
        AirfoilAsk = 'Enter The Airfoil Number in Row Vector as [x x x x]:  ';

Airfoil = input(AirfoilAsk);

NACA = Airfoil;

ChordAsk = 'Enter The Airfoil Chord Length:  ';    % Airfoil Chord

Chord = input(ChordAsk);

x = 0:0.0001:Chord;

if length(NACA) == 4

   disp(['NACA 4 Digit Series:  NACA ',  num2str(NACA(1)) num2str(NACA(2)) num2str(NACA(3)) num2str(NACA(4))])

   if NACA(1) == 0 && NACA(2) == 0

      Symm = 1;

      disp('Symmetric Airfoil')

   else

      Symm = 0;

      disp('Cambered Airfoil')

   end

end

if Symm == 1

   t = str2num([num2str(NACA(3)),num2str(NACA(4))])/100;
```

```
    y_upper = 5*t*Chord*(0.2969*sqrt(x/Chord)-0.126*(x/Chord)-
0.3516*(x/Chord).^2+0.2843*(x/Chord).^3-0.1015*(x/Chord).^4);
    y_lower = -y_upper;
    x_upper = x;
    x_lower = x;
else
    m = NACA(1)/100;
    p = NACA(2)*Chord/10;
    t = str2num([num2str(NACA(3)),num2str(NACA(4))])/100;
    for i = 1:length(x)
        if x(i)/Chord<=p
            y_camber(i) = m*x(i)/p^2*(2*p-x(i)/Chord);
            dy_camber(i) = 2*m/p^2*(p-x(i)/Chord);
        else
            y_camber(i) = m*(Chord-x(i))/(1-p)^2*(1+x(i)/Chord-2*p);
            dy_camber(i) = 2*m/(1-p)^2*(p-x(i)/Chord);
        end
    end
    y_t = 5*t*Chord*(0.2969*sqrt(x/Chord)-0.126*(x/Chord)-
0.3516*(x/Chord).^2+0.2843*(x/Chord).^3-0.1015*(x/Chord).^4);
    theta = atan(dy_camber);
    x_upper = x-y_t.*sin(theta);
    x_lower = x+y_t.*sin(theta);
    y_upper = y_camber+y_t.*cos(theta);
    y_lower = y_camber-y_t.*cos(theta);
```

```
end
% Plots
figure
hold on
grid on
axis equal
plot(x_upper,y_upper,x_lower,y_lower,'LineWidth',1.5,'color','b')
plot(x,y_camber,'--','LineWidth',1,'color','r')
title(['NACA 4 Digit Series: NACA ', num2str(NACA(1)) num2str(NACA(2)) num2str(NACA(3)) num2str(NACA(4))])
xlabel('x')
ylabel('y')
```

Result: The output is shown in figure 5.1

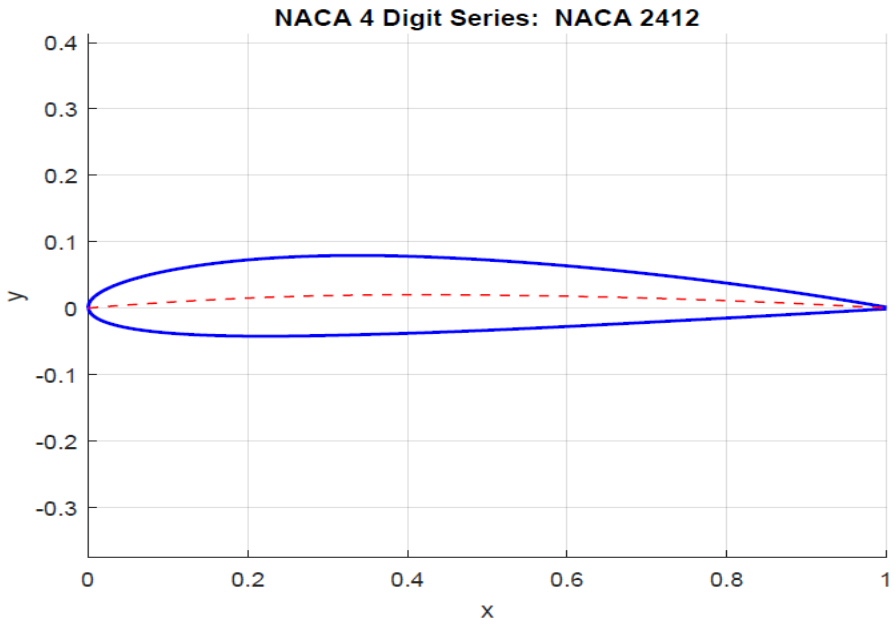


Fig 5

EXPERIMENT 6

PROGRAM TO FIND CRITICAL MACH NUMBER AND DRAG POLAR

Aim: Write a code to find critical Mach number and to draw drag polar of an airfoil.

Equipment needed: Core 2 duo processor with 1GB RAM and MATLAB software Version 2008a or higher or Sci-lab.

Theory :

Critical Mach number: In aerodynamics the critical Mach number (Mcr or M*) of an airfoil is the lowest Mac Number at which the airflow over some point of the airfoil reaches the speed of sound but does not exceed it. This creates a weak shock wave as aircraft exceeds the critical Mach number, its drag coefficient increases suddenly, causing dramatically increased drag.

Plot the variation of Coefficient of Pressure with respect to Mach number using Prandtl-Gluaret compressibility correction for given airfoil

$$Cp = \frac{Cp_0}{\sqrt{M_\infty^2 - 1}} \text{ ------- (1)}$$

Where Cp_0 is incompressible pressure coefficient of given airfoil and M_∞ is freestream Mach number.

Similarly plot the variation of Critical Coefficient of Pressure with respect to Mach number for given airfoil using equation

$$Cp_{cr} = \frac{2}{\gamma M_\infty^2}\left[\left(\frac{1+[(\gamma-1)/2]M_\infty^2}{1+(\gamma-1)/2}\right)^{\gamma/(\gamma-1)} - 1\right] \text{ --------- (2)}$$

The intersection of these two curves will give critical Mach number of respective airfoil.

Drag Polar: The Drag Polar is the relationship between the lift on an airfoil and its drag, expressed in terms of coefficients. It may be described by an equation or displayed in a diagram called a polar plot.

Algorithm

Critical Mach number

i. Give the incompressible pressure coefficient of airfoil.
ii. Give the range of Mach number.
iii. Calculate coefficient of pressure using equation (1).
iv. Calculate Critical coefficient of pressure using equation (2)
v. Plot the graph between coefficient of pressure and Mach number using two equations.

Drag Polar:

i. Load pressure data for various angle of attack
ii. Calculate lift and drag coefficient of airfoil

Result: The intersection of curve 1 and curve 2 gives critical Mach number (Refer fig 6).

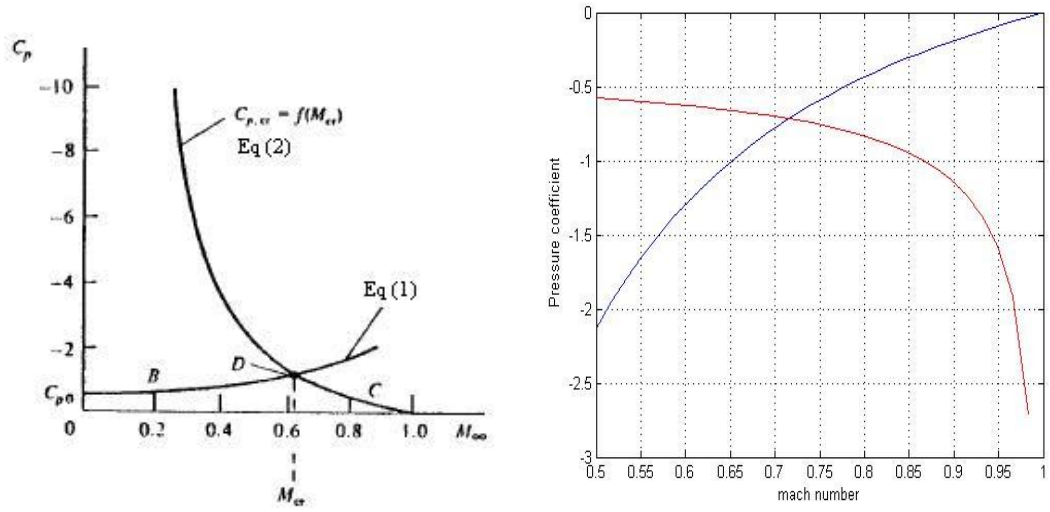


Fig 6: Critical Mach Number

EXERCISES

1. The minimum pressure coefficient for an NACA 0009 airfoil in low-speed flow is -0.25. Calculate the critical Mach number for this airfoil using Prandtl - Glauert rule and Karman-Tsien rule using Matlab code.

2. Plot drag polar plots of symmetric and cambered airfoils using Matlab code.

EXPERIMENT 7

FLOW CHARACTERISTICS ACROSS SHOCK WAVES

Aim: Write a code to find flow characteristics across shock waves

Equipment needed: Core 2 duo processor with 1GB RAM and MATLAB software Version 2008a or higher or Sci-lab.

Theory: Shock waves are formed when a pressure front moves at supersonic speeds and pushes on the surrounding air. At the region where this occurs, sound waves travelling against the flow reach a point where they cannot travel any further upstream and the pressure progressively builds in that region; a high-pressure shock wave rapidly forms.

Normal Shock: If the shock is perpendicular to flow direction then it is known as normal shock. As the flow passes shock wave Mach number decreases with the relation

$$M_2^2 = \frac{1 + [(\gamma - 1)/2]M_1^2}{\gamma M_1^2 - (\gamma - 1)/2}$$

The rise in pressure, density, and temperature after normal shock can be calculated as follows:

$$\frac{\rho_2}{\rho_1} = \left[\frac{(\gamma + 1)M_1^2}{2 + (\gamma - 1)M_1^2}\right]$$

$$\frac{P_2}{P_1} = \left[1 + \frac{2\gamma}{\gamma + 1}(M_1^2 - 1)\right]$$

$$\frac{T_2}{T_1} = \frac{P_1}{P_2}\frac{\rho_2}{\rho_1}$$

Oblique Shock: When analyzing shock waves in a flow field, which are still attached to the body, the shock wave which is deviating at some arbitrary angle from the flow direction is termed oblique shock. For a given Mach number, M_1, and corner angle, θ, the oblique shock angle, β, and the downstream Mach number, M_2 based on M_{n2}, can be calculated.

$$tan\theta = 2cot\beta\left[\frac{sin^2\beta - 1}{M_1^2(\gamma + cos2\beta) + 2}\right]$$

$$M_{n1} = M_1 sin\beta$$

$$M_2 = \frac{M_{n2}}{\sin(\beta - \theta)}$$

Algorithm:

i. Enter the range of Mach number.

ii. Enter the value of specific heat ratio.

iii. Write the equations for pressure, density and temperature variation across shocks.

iv. Plot the variation

Result: Shown in Fig 7.

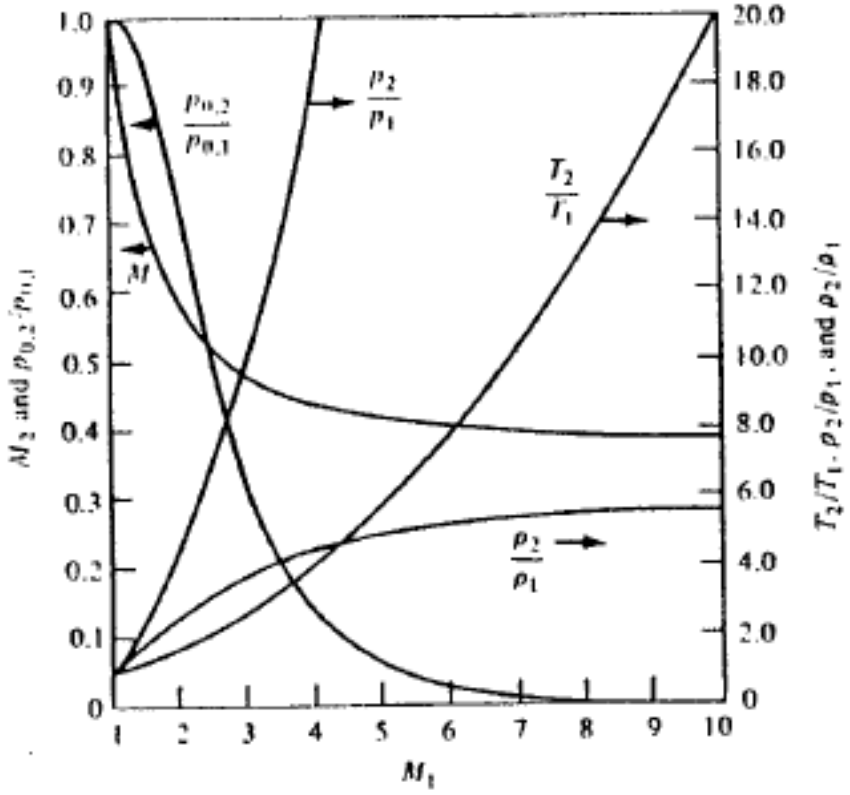


Fig 7: Variation of flow properties across normal shock wave

EXERCISES

1. Write a Matlab code to solve oblique shock problems.
2. Write a Matlab code to solve expansion wave problems.

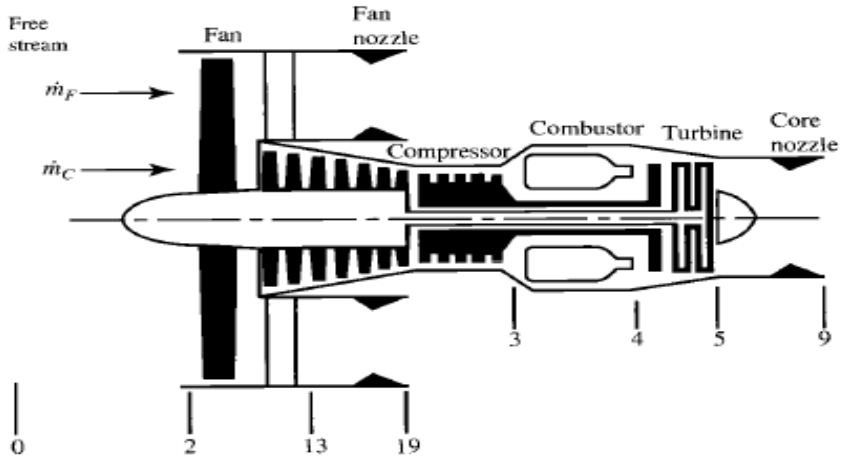
Experiment 8

PERFORMANCE OF TURBOFAN ENGINE

Aim: Write a Matlab code to find the performance of turbofan engine.

Equipment needed: Core 2 duo processor with 1GB RAM and MATLAB software Version 2008a or higher or Sci-lab.

Theory: The turbofan is a type of airbreathing jet engine that is widely used in aircraft propulsion. The *turbo* portion refers to a gas turbine engine which achieves mechanical energy from combustion, and the *fan*, a ducted fan that uses the mechanical energy from the gas turbine to accelerate air rearwards.



For clarity we adopt the following notations

M_0 – Mach number
T_0 – Initial Temperature
γ – Ratio of specific heats
T_4 – Combustor entry temperature
a_0 – speed of sound

π_c – Compressor pressure ratio
π_f – Fan pressure ratio
F – Thrust
f – fuel to air ratio

The performance of Ideal turbofan engine is plotted in terms of fuel to air ratio vs compressor pressure ratio and Thrust vs compressor pressure ratio using following equations:

$$F = a_0 \frac{1}{1+\alpha}\left[\frac{V_9}{a_0} - M_0 + \alpha\left(\frac{V_{19}}{a_0} - M_0\right)\right]$$

$$\frac{V_9}{a_0} = \sqrt{\frac{2}{\gamma-1}\left\{\tau_\lambda - \tau_r\left[\tau_c - 1 + \alpha(\tau_f - 1)\right] - \frac{\tau_\lambda}{\tau_f \tau_c}\right\}}$$

$$\frac{V_{19}}{a_0} = \sqrt{\frac{2}{\gamma-1}(\tau_r\tau_f - 1)}$$

Ram temperature ratio $\tau_r = 1 + \frac{\gamma-1}{2}M_0^2$

Burner to Exit Temperature ratio$\tau_\lambda = \frac{T_{t4}}{T_0}$

Compressor temperature ratio $\tau_c = (\pi_c)^{(\gamma-1)/\gamma}$

Fan temperature ratio $\tau_f = (\pi_f)^{(\gamma-1)/\gamma}$

Fuel to air ratio $f = \frac{C_p T_0}{h_{pr}}(\tau_\lambda - \tau_r\tau_c)$

Specific Fuel Consumption $S = \frac{f}{(1+\alpha)F}$

Thermal Efficiency $\eta_T = 1 - \frac{1}{\tau_r\tau_c}$

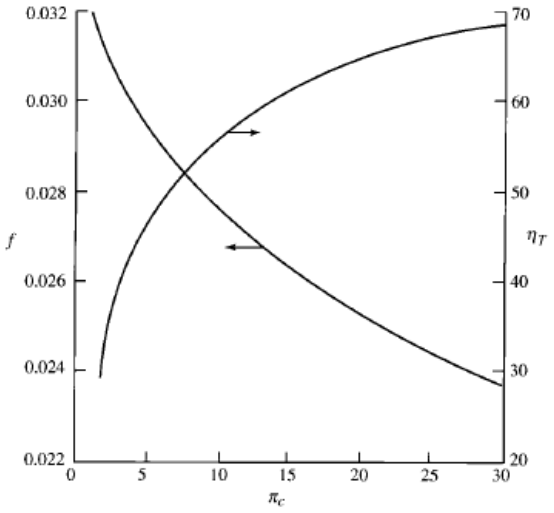
Algorithm:

i. Give the range of compressor pressure ratio.
ii. Enter all the required data like Mach number, Initial Temperature, Ratio of specific heats, Combustor entry temperature.

iii. Write code for above equations to find performance of turbofan engine.

Result: Shown in the figure 8.

i. Plot compressor pressure ratio vs Thrust,
ii. Compressor pressure ratio vs fuel to air ratio and
iii. Compressor pressure ratio vs Thermal efficiency.



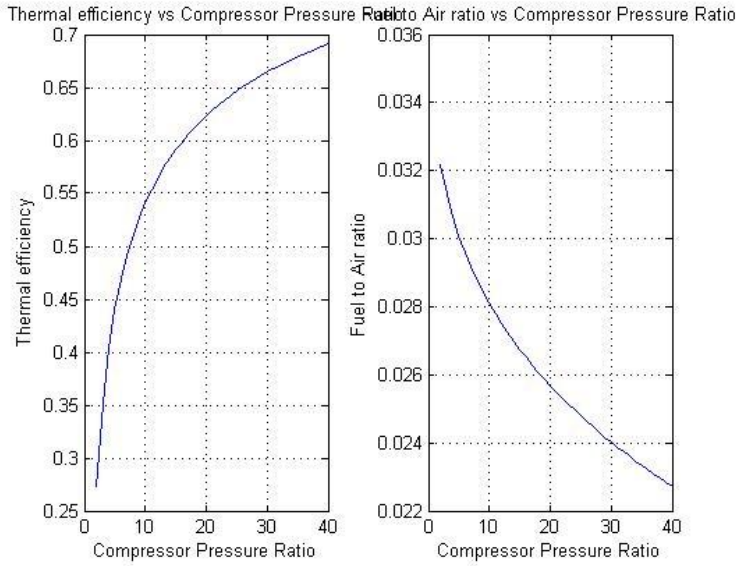


Fig 8: Compressor ratio vs fuel air ratio and Thermal efficiency

EXERCISES

1. Write Matlab code to analyze turbojet performance for varying compressor pressure

EXPERIMENT 9

FLOW CHARACTERISTICS OF A CD NOZZLE

Aim: Write a code to find the variation of flow characteristics in a CD nozzle.

Equipment needed: Core 2 duo processor with 1GB RAM and MATLAB software Version 2008a or higher or Sci-lab.

Theory: For a supersonic flow to develop from a reservoir where the velocity is zero, the subsonic flow must first accelerate through a converging area to a throat, followed by continued acceleration through an enlarging area. The nozzles on a rocket designed to place satellites in orbit are constructed using such converging-diverging geometry.

The following relation is used to find the variation of Mach number with respect to area ratio.

$$\frac{A}{A^*} = \frac{1}{M^2}\left[\frac{2}{\gamma+1}\left(1+\frac{\gamma-1}{2}M^2\right)\right]^{(\gamma+1)/(\gamma-1)}$$

Then use isentropic relation to find pressure, density and temperature variation across nozzle.

Algorithm:

i. Give the range of Area ratio.
ii. Enter all the required data like Mach number, Ratio of specific heats
iii. Write code for above equations to find Mach number.
iv. Write code for isentropic relations.

Result: Plot the variation of flow characteristics along nozzle from inlet to outlet.

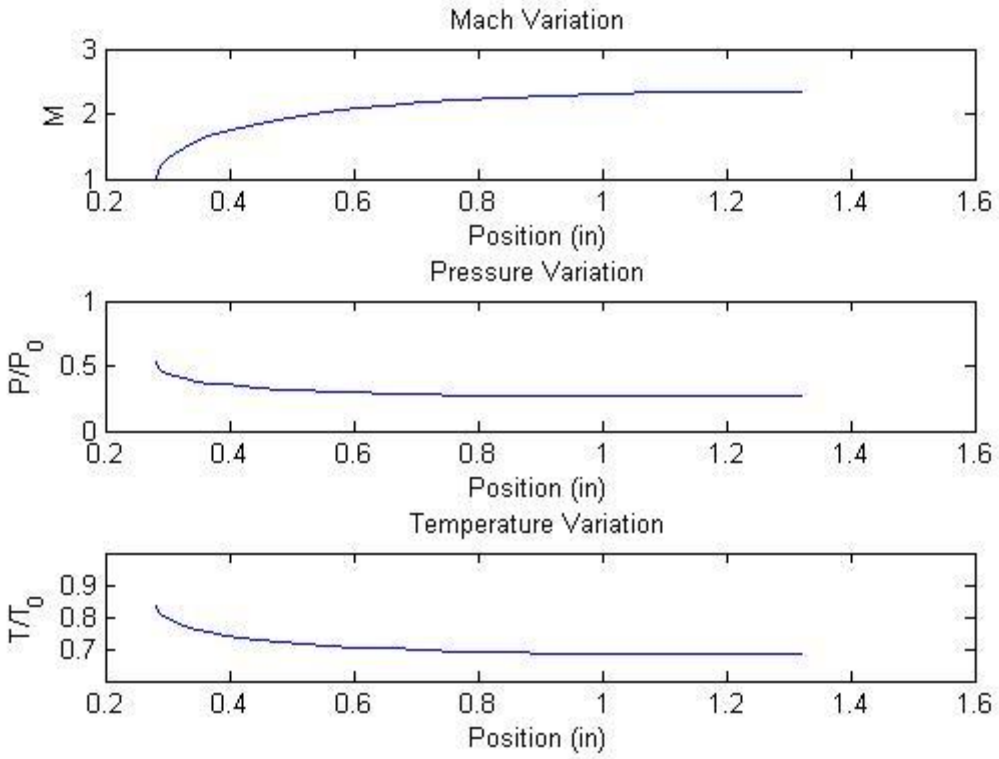


Fig 9. Variation of flow characteristics along nozzle

EXERCISES

1. Write a Matlab code for a supersonic wind tunnel to produce Mach 2.4 at standard sea level conditions to calculate exit to throat area ratio of nozzle and reservoir pressure and temperature.
2. The reservoir pressure of a supersonic wind tunnel is 5 atm. A static pressure probe measures 4 atm, 2.64 atm and 0.5 atm along centerline of nozzle. Write a matlab code to calculate local Mach number and area ratio.

EXPERIMENT 10

Aim: Program to design contour of nozzle.

Software: Matlab

```
clear all;
clc;
%   Problem parameters
T_c = 2000;     % Temperature in the combustion chamber (K)
P_c = 1.2e6;    % Pressure in the combustion chamber (Pa)
P_amb = 101e3;  % Ambient pressure (Pa)
T_amb = 300;    % Ambient temperature (K)
gamma = 1.25;   % Ratio of Specific Heats Cp/Cv (Gamma)
W = 25.4;       % Molecular weight of gas (kg/kmol)
width = .1;     % Nozzle width (meters)
h_th = .025;    % Throat  height (meters)
%   Method of Characteristics
num = 15;       % Number of Characteristic lines
theta_i = .03;  % Initial step in theta
plotter = 1;    % Set to '1' to plot nozzle

dh = h_th/100;
max_iter = 10000;
R = 8314/W;
%   Method of Characteristics
M_e = Ma(b);       %Mach number at ideal exit
%Find theta_max by using equation 11.33
theta_max = (180/pi)*(sqrt((gamma+1)/(gamma-1))*atan((sqrt((gamma-1)*(M_e^2-1)/(gamma+1))))-
atan(sqrt(M_e^2-1)))/2;
%  D_theta for each char line
del_theta = (theta_max - theta_i)/(num-1);
% Find
for i=1:num
    %   Initialize mach numeber

    for j=1:num
        if i==1
            %Theta for each line (first lines)
            theta(i,j) = theta_i + del_theta*(j-1);
            nu(i,j) = theta(i,j);
            K_m(i,j) = theta(i,j) + nu(i,j);
            K_p(i,j) = theta(i,j) - nu(i,j);

        elseif i > 1

            K_p(i,j) = -K_m(1,i);

            % Find Thetas
            if j >= i
                theta(i,j) = del_theta*(j-i);
            else
                %theta(i,j) = theta(j,i-1);
                theta(i,j) = theta(j,i);

            end
            nu(i,j) = theta(i,j) - K_p(i,j);
            K_m(i,j) = theta(i,j) + nu(i,j);
        end
```

```
    % Prandtl-Meyer function (using Newton Rhapson)
    dM = .1; % Leave at about .1
    if j == 1
        M_ex(i,j) = 1.00;
    else
        M_ex(i,j) = M_ex(i,j-1);
    end
    M = M_ex(i,j);

    res = 1;
    while res > .01
        M2 = M + dM;
        funv1 = (-nu(i,j)*(pi/180)+(sqrt((gamma+1)/(gamma-1))*atan((sqrt((gamma-1)*(M^2-
1)/(gamma+1))))-atan(sqrt(M^2-1))));
        funv2 = (-nu(i,j)*(pi/180)+(sqrt((gamma+1)/(gamma-1))*atan((sqrt((gamma-1)*(M2^2-
1)/(gamma+1))))-atan(sqrt(M2^2-1))));
        dv_dm = (funv2-funv1)/dM;

        M = M - funv1/dv_dm;
        res = abs(funv1);

    end
    M_ex(i,j) = M;

    % Find the angle mu
    mu(i,j) = (180/pi)*asin(1/M_ex(i,j));

    end

    % Add last point to char line
    theta(i,num+1) = theta(i,num);
    nu(i,num+1) = nu(i,num);
    K_m(i,num+1) = K_m(i,num);
    K_p(i,num+1) = K_p(i,num);
end
char = zeros(num,num+1,2);
for i=1:num

    for j=1:num+1

% Draw points of intersection
        %   Point 1 of all char lines
        if j == 1
            char(i,j,1) = 0;
            char(i,j,2) = h_th/2;
        end

        %   Where first line hits the symmetry line
        if i == 1 & j==2
            char(i,j,1) = (-h_th/2)/tan((pi/180)*(theta(1,j-1)-mu(1,j-1)));
            char(i,j,2) = 0;
        end
        %   Where all other lines hit the symmetry line
        if j == i+1 & j>2
              char(i,j,1) = -char(i-1,j,2)/tan((pi/180)*(.5*theta(i,j-2)-.5*(mu(i,j-2)+mu(i,j-1))))
+ char(i-1,j,1);
              char(i,j,2) = 0;
              test(i,j) = (theta(i,j-2)-.5*(mu(i,j-2)+mu(i,j-1)));
              testpty(i,j) = char(i-1,j,2);
              testptx(i,j) = char(i-1,j,1);
        end
```

```
        %   All other data points for char 1 calculated
        if i ==1 & j>2 & j ~= i+1
            C_p = tan((pi/180)*(.5*(theta(i,j-2)+theta(i,j-1))+.5*(mu(i,j-2)+mu(i,j-1))));
            C_m = tan((pi/180)*(.5*(theta(j-1,1)+theta(i,j-1))-.5*(mu(j-1,1)+mu(i,j-1))));
            A = [1,-C_m;1,-C_p];
            B = [char(1,1,2) - char(1,1,1)*C_m;
                char(1,j-1,2) - char(1,j-1,1)*C_p];
                iterm(1,:)=inv(A)*B;
                char(i,j,1) = iterm(1,2);
                char(i,j,2) = iterm(1,1);
        end

        %   All other points for all char lines calculated
        if i > 1 & j~=i+1 & j>2
            C_p = tan((pi/180)*(.5*(theta(i,j-2)+theta(i,j-1))+.5*(mu(i,j-2)+mu(i,j-1))));
            C_m = tan((pi/180)*(.5*(theta(i-1,j-1)+theta(i,j-1))-.5*(mu(i-1,j-1)+mu(i,j-1))));
            A = [1,-C_m;1,-C_p];
            B = [char(i-1,j,2) - char(i-1,j,1)*C_m; char(i,j-1,2) - char(i,j-1,1)*C_p];

            iterm(1,:) = inv(A)*B;
            char(i,j,1) = iterm(1,2);
            char(i,j,2) = iterm(1,1);
        end
    end
end

%  Fill in similar points (where char lines share points)
for i = 2:num
    for j=2:num
        char(j,i,1) = char(i-1,j+1,1);
        char(j,i,2) = char(i-1,j+1,2);
    end
end

% ******Make the nozzle shape and extend the char lines to wall******
%   Initial start point of the nozzle (at throat)
noz(1,1) = 0;
noz(1,2) = h_th/2;
%   Find all the points of the nozzle
for i = 2 : num
    %   Find different slopes and points to intersect
    m1 = tan((pi/180)*(theta(i-1,num)+mu(i-1,num)));
    if i ==2
        m2 = (pi/180)*theta_max;
    else
        m2 = ((pi/180)*(theta(i-1,num+1)));
    end
    m3 = ((pi/180)*(theta(i-1,num)));
    m4 = tan((m2+m3)/2);

    A = [1,-m4; 1,-m1];
    B = [noz(i-1,2) - noz(i-1,1)*m4; char(i-1,num+1,2) - char(i-1,num+1,1)*m1];

    iterm(1,:) = inv(A)*B;
    noz(i,1) = iterm(1,2);
    noz(i,2) = iterm(1,1);

    %   Extend char lines to wall
    char(i-1,num+2,1)= noz(i,1);
    char(i-1,num+2,2)= noz(i,2);
```

```
end
%Last line
m1 = tan((pi/180)*(theta(num,num)+ mu(num,num)));
m2 = ((pi/180)*(theta(num-1,num)));
m3 = ((pi/180)*(theta(num,num+1)));
m4 = tan((m2+m3)/2);
A = [1,-m4; 1,-m1];
B = [noz(num,2) - noz(num,1)*m4; char(num,num+1,2) - char(num,num+1,1)*m1];

iterm(1,:) = inv(A)*B;
noz(num+1,1) = iterm(1,2);
noz(num+1,2) = iterm(1,1);

%   Extend char lines to wall
char(num,num+2,1)= noz(num+1,1);
char(num,num+2,2)= noz(num+1,2);
if plotter ==1
%   Plot the nozzle shape
figure(1);clf;
subplot(2,1,1);
plot(noz(:,1),noz(:,2),'k','LineWidth',3)
hold on;
[a,b] = max(noz);
plot(a(1),A_max/width/2,'g*')
%   Plot for loop for char lines
for i = 1 : num
    figure(1)
    hold on;
    plot(char(i,:,1),char(i,:,2))
    hold on;
    plot(char(i,:,1),-char(i,:,2))
end
%   Plot the nozzle shape (bottom side)
figure(1)
subplot(2,1,1)
hold on;
plot(noz(:,1),-noz(:,2),'k','LineWidth',3)
hold on;
plot(a(1),-A_max/width/2,'g*')
title('Max Thrust (minimum length) Nozzle Design')
xlabel('Nozzle length (m)')
ylabel('Nozzle height (m)')
legend('Nozzle shape','Area_e_x_i_t(predicted)','Char. Lines')
else
end
```