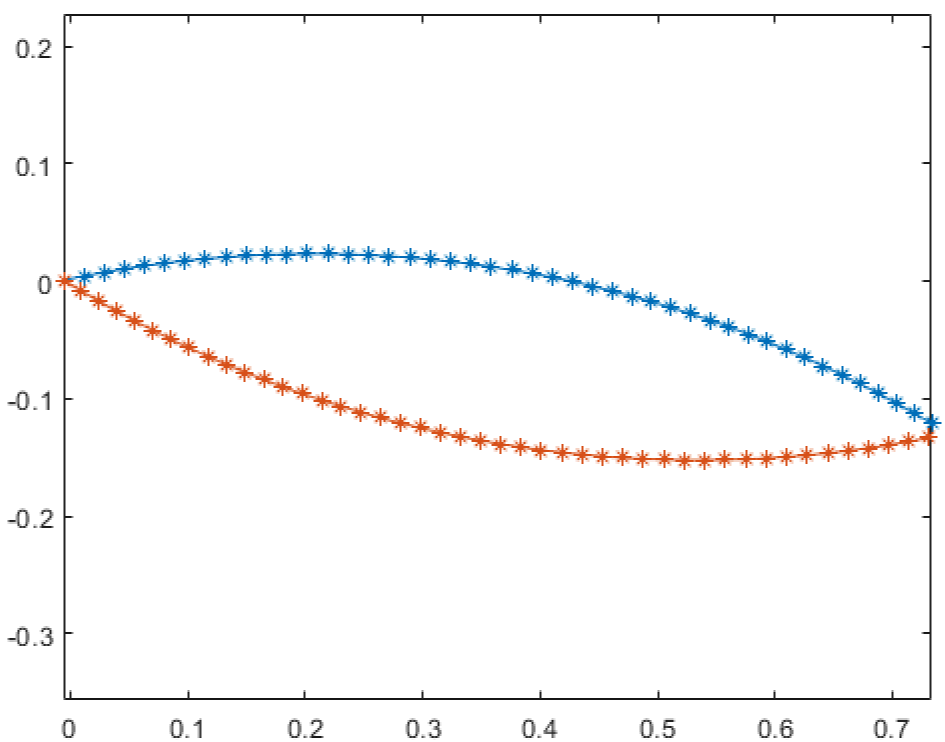
Table of Contents


```
%HYPERSONIC FLOWS AE 625
%HOMEWORK 4

close all
clear all


%geometry
up=[112.5:-1:68.5];
xt=cosd(up)+0.3769;
xb=xt;
yt=sind(up)-0.9239;
yb=-yt;
k=length(xt);
chord=abs(xt(1)-xt(k)); %chord length
S=0.03832131784768; %span (half of plot)
aoa=10*pi/180;% angle of attack


%rotating coordinates based on AoA
xt=xt.*cos(aoa)+yt.*sin(aoa);
yt=-xt.*sin(aoa)+yt.*cos(aoa);
xb=xb.*cos(aoa)+yb.*sin(aoa);
yb=-xb.*sin(aoa)+yb.*cos(aoa);

plot(xt,yt,'-*')
hold on
plot(xb,yb,'-*')
axis equal

%empirical constants for the M equation, given prandtl-meyer angles
%source: http://www.pdas.com/pm.pdf
A=1.3604;
B=0.0962;
C=-0.5127;
D=-0.6722;
E=-0.3278;




n=length(up);
R=8134; %gas constant
T=273.15; %atmospheric characteristic temperature (K)
```

```
P=101325; %atmospheric pressure at SL (Pa)
rho=1.225; %atmospheric density at SL (kg/m^3)
M=zeros(n,n);
M(1,:)=linspace(4,10,n);%Mach numbers
g=1.4; %heat air coefficient
muInf=(pi/2)*(sqrt((g+1)/(g-1))-1); %prandtl-meyer max angle
mu1=zeros(1,n);
mu2=zeros(1,n);
p1s=zeros(1,n);

p1s(1)=P;

sound=sqrt(g*R*T);
m=zeros(1,n);
theta=zeros(1,n);
M2=zeros(1,n);
p2b=zeros(n,n);
p2t=zeros(n,n);
```



BOTTOM PART

```
for j=1:n
%First slope: oblique shock

m(1)=atan((yb(2)-yb(1))/(xb(2)-xb(1)));
theta(1)=aoa+m(1);
```

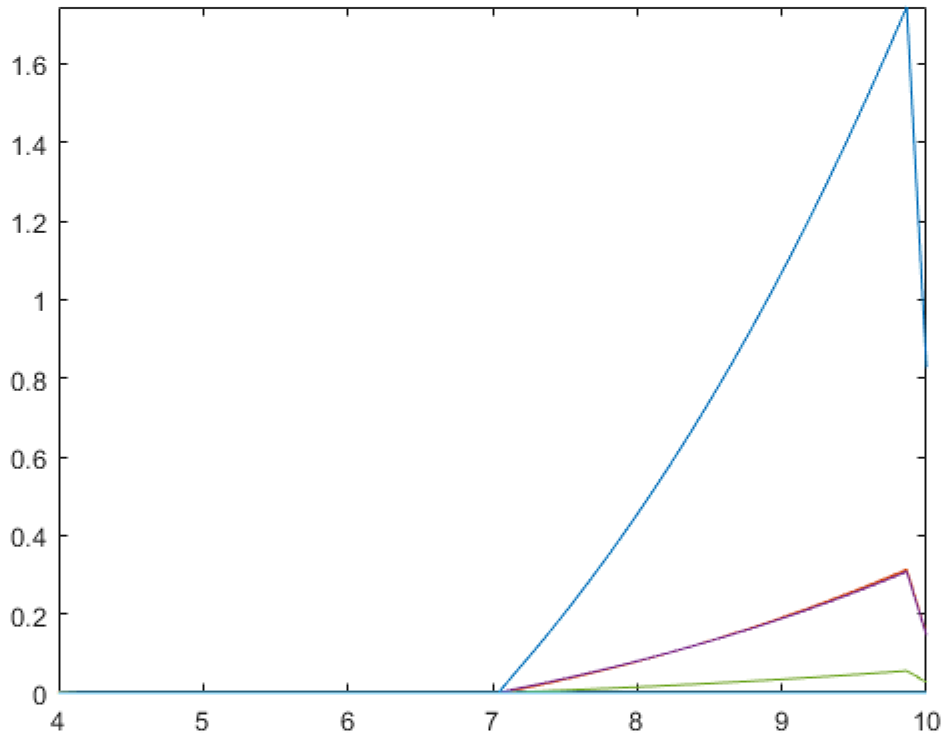
```
mu(j)=asin(1/M(1,j));  % Mach wave angle
c(j)=tan(mu(j))^2;
a(j)=((g-1)/2+(g+1)*c(j)/2)*tan(theta(1));
b(j)=((g+1)/2+(g+3)*c(j)/2)*tan(theta(1));
d(j)=sqrt(4*(1-3*a(j)*b(j))^3/((27*a(j)^2*c(j)+9*a(j)*b(j)-2)^2)-1);
beta(j)=atan((b(j)+9*a(j)*c(j))/(2*(1-3*a(j)*b(j)))-
(d(j)*(27*a(j)^2*c(j)+9*a(j)*b(j)-2))/
(6*a(j)*(1-3*a(j)*b(j)))*tan(n*pi/3+1/3*atan(1/d(j))));


%equation M_2 for oblique shock relations
M(j+1,1)=(1/(sin(beta(j)-
theta(1))))*sqrt((1+((g-1)/2)*M(1,j)^2*sin(beta(j))^2)/
((g*M(1,j)^2*sin(beta(j))^2)-((g-1)/2)));

%obtaining the pressure according to shock relations
p2b(1,j)=(1+((2*g)/(g+1))*(M(1,j)^2*sin(beta(j))^2-1))*p1s(j);
p1s(j+1)=p2b(1,j);
M(j+1,2)=M(j+1,1);

for i=2:n
    %EXPANSION
    if i<n
        m(i)=atan(((yb(i+1)-yb(i))/(xb(i+1)-xb(i))));
        theta(i)=aoa+m(i);
    end

        %calculating prandtl-meyer angles
        mu1(i)=sqrt((g+1)/(g-1))*atan(sqrt(((g-1)/(g+1))*(M(j
+1,i)^2-1)))-atan(sqrt(M(j+1,i)^2-1));
        mu2(i)=mu1(i)+theta(i);

        y(i)=mu2(i)/muInf;
        M(j+1,i+1)=(1+A*y(i)+B*y(i)^2+C*y(i)^3)/(1+D*y(i)+E*y(i)^2);

        %pressure relation for expansion waves
        p2b(i,j)=((1+(g-1)*M(j+1,i)^2/2)/(1+(g-1)*M(j+1,i+1)^2/2))^(g/
(g-1))*p1s(i);
        p1s(i+1)=p2b(i,j);


end

mu1=zeros(1,n);
mu2=zeros(1,n);
p1s=zeros(1,n);

p1s(j+1)=P;

m=zeros(1,n);

end
```

```
M=zeros(n,n);
M(1,:)=linspace(4,10,n);
mu1=zeros(1,n);
mu2=zeros(1,n);
p1s=zeros(1,n);

p1s(1)=P;

m=zeros(1,n);
```

UPPER PART

```
for j=1:n
%First slope: oblique shock

m(1)=atan((yt(2)-yt(1))/(xt(2)-xt(1)));
theta(1)=aoa+m(1);


mu(j)=asin(1/M(1,j));  % Mach wave angle
c(j)=tan(mu(j))^2;
a(j)=((g-1)/2+(g+1)*c(j)/2)*tan(theta(1));
b(j)=((g+1)/2+(g+3)*c(j)/2)*tan(theta(1));
d(j)=sqrt(4*(1-3*a(j)*b(j))^3/((27*a(j)^2*c(j)+9*a(j)*b(j)-2)^2)-1);
beta(j)=atan((b(j)+9*a(j)*c(j))/(2*(1-3*a(j)*b(j)))-
(d(j)*(27*a(j)^2*c(j)+9*a(j)*b(j)-2))/
(6*a(j)*(1-3*a(j)*b(j)))*tan(n*pi/3+1/3*atan(1/d(j))));


%equation M_2 for oblique shock relations
M(j+1,1)=(1/(sin(beta(j)-
theta(1))))*sqrt((1+((g-1)/2)*M(1,j)^2*sin(beta(j))^2)/
((g*M(1,j)^2*sin(beta(j))^2)-((g-1)/2)));

%obtaining the pressure according to shock relations
p2t(1,j)=(1+((2*g)/(g+1))*(M(1,j)^2*sin(beta(j))^2-1))*p1s(j);
p1s(j+1)=p2t(1,j);
M(j+1,2)=M(j+1,1);

for i=2:n
    %EXPANSION
    if i<n
        m(i)=atan(((yt(i+1)-yt(i))/(xt(i+1)-xt(i))));
        theta(i)=aoa+m(i);
    end

        %calculating prandtl-meyer angles
        mu1(i)=sqrt((g+1)/(g-1))*atan(sqrt(((g-1)/(g+1))*(M(j
+1,i)^2-1)))-atan(sqrt(M(j+1,i)^2-1));
        mu2(i)=mu1(i)+theta(i);

        y(i)=mu2(i)/muInf;
```

```
        M(j+1,i+1)=(1+A*y(i)+B*y(i)^2+C*y(i)^3)/(1+D*y(i)+E*y(i)^2);

        %pressure relation for expansion waves
        p2t(i,j)=((1+(g-1)*M(j+1,i)^2/2)/(1+(g-1)*M(j+1,i+1)^2/2))^(g/
(g-1))*p1s(i);
        p1s(i+1)=p2t(i,j);


end

mu1=zeros(1,n);
mu2=zeros(1,n);
p1s=zeros(1,n);

p1s(j+1)=P;

m=zeros(1,n);

end

M1=linspace(4,10,n);

for j=1:n
    for i=1:n
         V(i)=M1(i)*sound;
         %calculating Lift and Drag
         L(j,i)=(p2t(j,i)-p2b(j,i))*chord*sin(aoa);
         D(j,i)=(p2t(j,i)-p2b(j,i))*chord*cos(aoa);
         %calculating lift and drag coefficients based on Q_Inf
         cl(j,i)=2*L(j,i)/(rho*V(i)^2*S*2);
         cd(j,i)=2*D(j,i)/(rho*V(i)^2*S*2);
    end
end

figure(2)
plot (M1,cd)
hold on
plot(M1,cl)
ylim([0 Inf])

Warning: Imaginary parts of complex X and/or Y arguments ignored
Warning: Imaginary parts of complex X and/or Y arguments ignored
```



COMMENTS

```
%{
 My approach was to consider the first "panel" as an oblique shock and
the rest of the geometry as a gradual expansion until the trailing
 edge. So
I would calculate the slope of the panels given the coordinates of two
subsequent points and add that to our initial angle of attack (10°).
 The
sign of the slope should automatically take care of the fact that
 wether we
are analyzing a shock or an expansion wave the slope should be either
subtracted or added to the initial angle.

Then I created a Mach number matrix in which the first row is the free
stream Mach no. (4-10) and the others will be the analysis of each
 panel at
every initial mach no.
So in the end after all the calculations I ended up having a 45x45
 matrix
for the upper surface pressure and the bottom surface pressure, and
 every
row represents every initial mach number, while the columns represent
 the
value of the pressure along the geometry.
```

```
For some reason I ended up having imaginary values at some point and
 could
not find the bug.
%}
```

Published with MATLAB® R2019b