Scilab Textbook Companion for Fundamentals Of Aerodynamics by J. D. Anderson Jr.[1]

Created by
Prateek Bhandari
B.Tech
Others
IIT Bombay
College Teacher
Iit Bombay
Cross-Checked by

May 21, 2016

[1]Funded by a grant from the National Mission on Education through ICT, http://spoken-tutorial.org/NMEICT-Intro. This Textbook Companion and Scilab codes written in it can be downloaded from the "Textbook Companion Project" section at the website http://scilab.in

Book Description

Title: Fundamentals Of Aerodynamics

Author: J. D. Anderson Jr.

Publisher: McGraw - Hill

Edition: 3

Year: 2001

ISBN: 0072373350

Scilab numbering policy used in this document and the relation to the above book.

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

AP Appendix to Example(Scilab Code that is an Appednix to a particular Example of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means a scilab code whose theory is explained in Section 2.3 of the book.

Contents

List of Scilab Codes

Chapter 1

Aerodynamics Some Introductory Thoughts

Scilab code Exa 1.1 Calculation of drag coefficient over a wedge

```
//All the quantities are in SI units
M_inf = 2; //freestream mach number
p_inf = 101000; //freestream static pressure
rho_inf = 1.23; //freestream density
T_inf = 288; //freestream temperature
R = 287; //gas constant of air
a = 5; //angle of wedge in degrees
p_upper = 131000; //pressure on upper surface
p_lower = p_upper; //pressure on lower surface is equal to upper surface
c = 2; //chord length of the wedge
c_tw = 431; //shear drag constant

//SOLVING BY FIRST METHOD
//According to equation 1.8, the drag is given by D = I1 + I2 + I3 + I4
//Where the integrals I1, I2, I3 and I4 are given as

I1 = (-p_upper*sind(-a)*c/cosd(a))+(-p_inf*sind(90)*
```

```
c*tand(a)); //pressure drag on upper surface
I2 = (p_lower*sind(a)*c/cosd(a))+(p_inf*sind(-90)*c*tand(a));  //pressure drag on lower surface
I3 = c_tw*cosd(-a)/0.8*((c/cosd(a))^0.8);                //skin friction drag on upper surface
I4 = c_tw*cosd(-a)/0.8*((c/cosd(a))^0.8);                //skin friction drag on lower surface

D = I1 + I2 + I3 + I4; //Total Drag

a_inf = sqrt(1.4*R*T_inf); //freestream velocity of sound
v_inf = M_inf*a_inf; //freestream velocity
q_inf = 1/2*rho_inf*(v_inf^2); //freestream dynamic pressure
S = c*1; //reference area of the wedge

c_d1 = D/q_inf/S; //Drag Coefficient by first method

printf("\nRESULT\n--------\nThe Drag coefficient by first method is: %1.3f\n", c_d1)

//SOLVING BY SECOND METHOD
C_p_upper = (p_upper-p_inf)/q_inf; //pressure coefficient for upper surface
C_p_lower = (p_lower-p_inf)/q_inf; //pressure coefficent for lower surface

c_d2 = (1/c*2*((C_p_upper*tand(a))-(C_p_lower*tand(-a)))) + (2*c_tw/q_inf/cosd(a)*(2^0.8)/0.8/c);

printf("\nThe Drag coefficient by second method is: %1.3f\n\n", c_d2)
```

Scilab code Exa 1.3 Calculation of center of pressure for a NACA 4412 airfoil

```
//All the quantities are expressed in SI units

alpha = 4; //angle of attack in degrees
c_l = 0.85; //lift coefficient
c_m_c4 = -0.09; //coefficient of moment about the
    quarter chord
x_cp = 1/4 - (c_m_c4/c_l); //the location centre of
    pressure with respect to chord

printf("\n\nRESULTS\n---------\nXcp/C = %1.3f\n\n",
    x_cp)
```

Scilab code Exa 1.5 Calculation of parametres for wind tunnel testing

```
V1 = 550; //velocity of Boeing 747 in mi/h
h1 = 38000; //altitude of Boeing 747 in ft
P1 = 432.6; //Freestream pressure in lb/sq.ft
T1 = 390; //ambient temperature in R
T2 = 430; //ambient temperature in the wind tunnel
    in R
c = 50; //scaling factor

//Calculations
//By equating the Mach numbers we get
V2 = V1*sqrt(T2/T1); //Velocity required in the wind
    tunnel
//By equating the Reynold's numbers we get
P2 = c*T2/T1*P1; //Pressure required in the wind
    tunnel
```

```
P2_atm = P2/2116; //Pressure expressed in atm
printf("\nRESULTS\n--------\nThe velocity required in
    the wind tunnel is: %3.1f mi/h\n\n",V2)
printf("The pressure required in the wind tunnel is:
    %5.0f lb/sq.ft or %2.2f atm\n\n",P2,P2_atm)
```

Scilab code Exa 1.6 Calculation of cruise lift coefficient and lift to drag ratio of a Cesna 560

```
v_inf_mph = 492; //freestream velocity in miles per
   hour
rho = 0.00079656; //aimbient air density in slugs
   per cubic feet
W = 15000; //weight of the airplane in lbs
S = 342.6; //wing planform area in sq.ft
C_d = 0.015; //Drag coefficient

//Calculations
v_inf_fps = v_inf_mph*(88/60); //freestream velocity
    in feet per second

C_l = 2*W/rho/(v_inf_fps^2)/S; //lift coefficient

//The Lift by Drag ratio is calculated as
L_by_D = C_l/C_d;

printf("\nRESULTS\n---------\nThe lift to drag ratio
   L/D is equal to: %2.0f\n",L_by_D)
```

Scilab code Exa 1.7 Calculation of maximum lift coefficient for Cesna 560

```
v_stall_mph = 100; //stalling speed in miles per
   hour
```

```
rho = 0.002377; //aimbient air density in slugs per
    cubic feet
W = 15900; //weight of the airplane in lbs
S = 342.6; //wing planform area in sq.ft

//Calculations
v_stall_fps = v_stall_mph*(88/60); //converting
    stalling speed in feet per second

//The maximum lift coefficient C_l_max is given by
    the relation
C_l_max = 2*W/rho/(v_stall_fps^2)/S;

printf("\nRESULTS\n---------\nThe maximum value of
    lift coefficient is\n                    Cl_max = %1.3f
    \n",C_l_max)
```

Scilab code Exa 1.8.a calculation of upward acceleration of a hot air balloon

```
d = 30; //inflated diameter of ballon in feet
W = 800; //weight of the balloon in lb
g = 32.2; //acceleration due to gravity
//part (a)
rho_0 = 0.002377; //density at zero altitude

//Assuming the balloon to be spherical , the Volume
    can be given as
V = 4/3*%pi*((d/2)^3);

//The Buoyancry force is given as
B = g*rho_0*V;

//The net upward force F is given as
F = B - W;
```

```

m = W/g; //Mass of the balloon

//Thus the upward acceleration of the ballon can be
    related to F as
a = F/m;

printf("\nRESULTS\n----------\nThe initial upward
    acceleration is:\n              a = %2.1f ft/s2",a)
```

Scilab code Exa 1.8.b Calculation of maximum altitude for the hot air balloon

```
d = 30; //inflated diameter of ballon in feet
W = 800; //weight of the balloon in lb
g = 32.2; //acceleration due to gravity
rho_0 = 0.002377; //density at sea level (h=0)
//part (b)
//Assuming the balloon to be spherical, the Volume
    can be given as
V = 4/3*%pi*((d/2)^3);
//Assuming the weight of balloon does not change,
    the density at maximum altitude can be given as
rho_max_alt = W/g/V;

//Thus from the given variation of density with
    altitude, we obtain the maximum altitude as

h_max = 1/0.000007*(1-((rho_max_alt/rho_0)^(1/4.21))
    )

printf("\nRESULTS\n----------\nThe maximum altitude
    that can be reached is:\n              h = %4.0f ft",
    h_max)
```

Chapter 2

Aerodynamics Some Fundamental Principles and Equations

Scilab code Exa 2.1 Calculation of time rate of change of volume of the fluid element per unit volume for the given velocity field

```
//All the quantities are in SI units
v_inf = 240; //freestream velocity
l = 1;       //wavelength of the wall
h = 0.01;    //amplitude of the wall
M_inf = 0.7; //freestream mach number
b = sqrt(1-(M_inf^2));
x = l/4;
y = l;

function temp = u(x,y)
temp = v_inf*(1 + (h/b*2*%pi/l*cos(2*%pi*x/l)*exp
   (-2*%pi*b*y/l)));
endfunction

function temp = v(x,y)
temp = -v_inf*h*2*%pi/l*sin(2*%pi*x/l)*exp(-2*%pi*b*
```

```
    y/l);
endfunction

d = 1e-10;

du = derivative(u,x,d);

dv = derivative(v,y,d);

grad_V = du + dv;

test = (b-(1/b))*v_inf*h*((2*%pi/l)^2)*exp(-2*%pi*b);

printf("\nRESULT\n--------\nThe time rate of change of the volume of the fluid element per unit volume is: %1.4f s-1\n", grad_V)
```

Chapter 3

Fundamentals of Inviscid Incompressible Flow

Scilab code Exa 3.1 Calculation of velocity on a point on the airfoil

```
//All the quantities are expressed in SI units

rho_inf = 1.23;     //freestream density of air at sea level
p_inf = 101000;     //freestream static pressure
v_inf = 50;         //freestream velocity
p = 90000;          //pressure at given point

//The velocity at the given point can be expressed as
v = sqrt((2*(p_inf-p)/rho_inf) + (v_inf^2));

printf("\nRESULTS\n---------\nThe velocity at the given point is\n              V = %3.1f m/s\n",v)
```

Scilab code Exa 3.2 Calculation of pressure on a point on the airfoil

```
//All the quantities are expressed in SI units

rho = 1.225;        //freestream density of air along the streamline
p_1 = 101314.1;     //pressure at point 1
v_1 = 3.05;         //velocity at point 1
v_2 = 57.91;        //velocity at point 2

//The pressure at point 2 on the given streamline can be given as
p_2 = p_1 + 1/2*rho*((v_1^2) - (v_2^2));

printf("\nRESULTS\n---------\nThe pressure at point 2 is\n                    p2 = %5.2f Pa\n",p_2)
```

Scilab code Exa 3.3 Calculation of velocity at the inlet of a venturimeter for a given pressure difference

```
//All the quantities are expressed in SI units

rho = 1.225;        //freestream density of air along the streamline
delta_p = 335.16;    //pressure difference between inlet and throat
ratio = 0.8;        //throat-to-inlet area ratio

//The velocity at the inlet can be given as
v_1 = sqrt(2*delta_p/rho/(((1/ratio)^2)-1));

printf("\nRESULTS\n---------\nThe value of velocity at the inlet is\n                    V1 = %3.1f m/s\n",v_1)
```

Scilab code Exa 3.4 Calculation of height difference in a U tube mercury manometer

```
//All the quantities are expressed in SI units

rho = 1.23;             //freestream density of air
    along the streamline
v = 50;                 //operating velocity inside
    wind tunnel
rho_hg = 13600;         //density of mercury
ratio = 12;             //contraction ratio of the
    nozzle
g = 9.8;                //acceleration due to gravity
w = rho_hg*g;           //weight per unit volume of
    mercury

//The pressure difference delta_p between the inlet
    and the test section is given as
delta_p = 1/2*rho*v*v*(1-(1/ratio^2));

//Thus the height difference in a U-tube mercury
    manometer would be
delta_h = delta_p/w;

printf("\nRESULTS\n---------\nThe height difference
    in a U-tube mercury manometer is\n
    delta_h = %1.5f m\n",delta_h)
```

Scilab code Exa 3.5 Calculation of the maximum allowable pressure difference between the wind tunnel settling chamber and test section

```
//all the quantities are expressed in SI units

ratio = 12;          //contraction ratio of wind
    tunnel nozzle
```

```
Cl_max = 1.3;        //maximum lift coefficient of the model
S = 0.56;            //wing planform area of the model
L_max = 4448.22;     //maximum lift force that can be measured by the mechanical balance
rho_inf = 1.225;     //free-stream density of air

//the maximum allowable freestream velocity can be given as
V_inf = sqrt(2*L_max/rho_inf/S/Cl_max);

//thus the maximum allowable pressure difference is given by
delta_p = 1/2*rho_inf*(V_inf^2)*(1-(ratio^-2));

printf("\nRESULTS\n---------\nThe maximum allowable pressure difference between the wind tunnel setling chamber and the test section is\n                    delta_p = %4.2f Pa",delta_p)
```

Scilab code Exa 3.6.a Calculation of reservoir pressure in a nozzle

```
//all the quantities are expressed in SI units

V2 = 100*1609/3600;          //test section flow velocity converted from miles per hour to meters per second
p_atm = 101000;              //atmospheric pressure
p2 = p_atm;                  //pressure of the test section which is vented to atmosphere
rho = 1.23;                  //air density at sea level
ratio = 10;                  //contraction ratio of the nozzle

```

```
//the pressure difference in the wind tunnel can be calculated as
delta_p = rho/2*(V2^2)*(1-(1/ratio^2));

//thus the reservoir pressure can be given as
p1 = p2 + delta_p;

p1_atm = p1/p_atm;          //reservoir pressure expressed in units of atm

printf("\nRESULTS\n---------\nThe reservoir pressure is\n                    p1 = %1.2f atm",p1_atm)
```

Scilab code Exa 3.6.b Calculation of increment in the reservoir pressure

```
//all the quantities are expressed in SI units

V2 = 89.4;          //test section flow velocity converted from miles per hour to meters per second
p_atm = 101000;             //atmospheric pressure
p2 = p_atm;                 //pressure of the test section which is vented to atmosphere
rho = 1.23;                 //air density at sea level
ratio = 10;                 //contraction ratio of the nozzle

//the pressure difference in the wind tunnel can be calculated as
delta_p = rho/2*(V2^2)*(1-(1/ratio^2));

//thus the reservoir pressure can be given as
p1 = p2 + delta_p;

```

```
p1_atm = p1/p_atm;          //reservoir pressure expressed in units of atm

printf("\nRESULTS\n---------\nThe new reservoir pressure is\n                 p1 = %1.3f atm",p1_atm)
```

Scilab code Exa 3.7 Calculation of airplane velocity from pitot tube measurement

```
//all the quantities are expressed in SI units

p0 = 104857.2;              //total pressure as measured by the pitot tube
p1 = 101314.1;              //standard sea level pressure
rho = 1.225;                //density of air at sea level

//thus the velocity of the airplane can be given as
V1 = sqrt(2*(p0-p1)/rho);

printf("\nRESULTS\n---------\nThe velocity of the airplane is\n                 V1 = %2.2f atm",V1)
```

Scilab code Exa 3.8 Calculation of pressure measured by the pitot tube for a given velocity

```
//all the quantities are expressed in SI units

V_inf = 100.1;                //freestream velocity
p_inf = 101314.1;             //standard sea level pressure
```

```
rho_inf = 1.225;                    //density of air at sea level

//the dynamic pressure can be calculated as
q_inf = 1/2*rho_inf*(V_inf^2);

//thus the total pressure is given as
p0 = p_inf + q_inf;

printf("\nRESULTS\n---------\nThe total pressure measured by pitot tube is\n                    p0 = %6.2f Pa",p0)
```

Scilab code Exa 3.9 Calculation of airplane velocity from pitot tube measurement

```
//all the quantities are expressed in SI units

p0 = 6.7e4;                   //total pressure as measured by the pitot tube
p1 = 6.166e4;                 //ambient pressure at 4km altitude
rho = 0.81935;                //density of air at 4km altitude

//thus the velocity of the airplane can be given as
V1 = sqrt(2*(p0-p1)/rho);

printf("\nRESULTS\n---------\nThe velocity of the airplane is\n                    V1 = %3.1f m/s = %3.0f mph",V1,V1/0.447)
```

Scilab code Exa 3.10 Calculation of equivallent air speed for an aircraft flying at a certain altitude

```
//all the quantities are expressed in SI units

V1 =114.2;                   //velocity of airplane at 4km altitude
rho = 0.81935;               //density of air at 4km altitude
q1 = 1/2*rho*(V1^2)          //dynamic pressure experienced by the aircraft at 4km altitude
rho_sl = 1.23;               //density of air at sea level

//according to the question
q_sl = q1;                   //sealevel dynamic pressure

//thus the equivallent air speed at sea level is given by
Ve = sqrt(2*q_sl/rho_sl);

printf("\nRESULTS\n---------\nThe equivallent airspeed of the airplane is\n            Ve = %2.1f m/s",Ve)
```

Scilab code Exa 3.11 Calculation of pressure coefficient on a point on an airfoil

```
//all the quantities are expressed in SI units

V_inf = 45.72;          //freestream velocity
V = 68.58;              //velocity at the given point

//the coeeficient of pressure at the given point is
```

```
given as
Cp = 1 - (V/V_inf)^2;

printf("\nRESULTS\n---------\nThe coefficient of pressure at the given point is\n                    Cp = %1.2f",Cp)
```

Scilab code Exa 3.12.a Calculation of velocity on a point on the airfoil for a given pressure coefficient

```
//all the quantities are expressed in SI units

Cp = -5.3;              //peak negative pressure coefficient
V_inf = 24.38;          //freestream velocity

//the velocity at the given point can be calculated as
V = sqrt(V_inf^2*(1-Cp));

printf("\nRESULTS\n---------\nThe velocity at the given point is\n                    V = %2.1f m/s",V)
```

Scilab code Exa 3.12.b Calculation of velocity on a point on the airfoil for a given pressure coefficient

```
//all the quantities are expressed in SI units

Cp = -5.3;              //peak negative pressure coefficient
V_inf = 91.44;          //freestream velocity

```

```
//the velocity at the given point can be calculated as
V = sqrt(V_inf^2*(1-Cp));

printf("\nRESULTS\n---------\nThe velocity at the given point is\n                    V = %3.1f m/s",V)
```

Scilab code Exa 3.13 Calculation of locations on cylinder where the surface pressure equals the freestream pressure

```
//all the quantities are expressed in SI units

//When p = p_inf, Cp = 0, thus
//1-4*(sin(theta)^2) = 0
//thus theta can be given as

theta = [asind(1/2) 180-asind(1/2) 180-asind(-1/2) 360+asind(-1/2)];              //sine inverse of 1/2 and -1/2 where theta varies from 0 to 360 degrees

printf("\nRESULTS\n---------\nThe angular locations where surface pressure equals freestream pressure are\n                    theta = %2.0f, %2.0f, %2.0f, %2.0f degrees",theta(1),theta(2),theta(3),theta(4))
```

Scilab code Exa 3.14 Calculation of the peak negative pressure coefficient for a given lift coefficient

```
//All the quantities are expressed in SI units

```

```
Cl = 5;                                   //lift coefficient of the cylinder
V_by_Vinf = -2 - Cl/2/%pi;                //ratio of maximum to freestream velocity

//thus the pressure coefficient can be calculated as
Cp = 1 - (V_by_Vinf^2);

printf("\nRESULTS\n---------\nThe peak negative pressure coefficient is\n                    Cp = %1.2f",Cp)
```

Scilab code Exa 3.15 Calculation of stagnation points and locations on cylinder where the surface pressure equals the freestream pressure

```
//All the quantities are expressed in SI units

theta = [180-asind(-5/4/%pi) 360+asind(-5/4/%pi)];         //location of the stagnation points

printf("\nRESULTS\n---------\nThe angular location of the stagnation points are\n                    theta = %3.1f, %3.1f degrees",theta(1),theta(2))

function temp = Cp(thet)
    temp = 0.367 -3.183*sind(thet) - 4*(sind(thet)^2);     //Cp written as a function of theta
endfunction

printf("\nRESULTS\n---------\nThe value of Cp on top of the cylinder is\n                    Cp = %1.2f",Cp(90))

[k] = roots([-4 -3.183 0.367]);

```

```
theta_2 = 180/%pi*[%pi-asin(k(1)) 2*%pi+asin(k(1)) asin(k(2)) %pi-asin(k(2))];

printf("\nRESULTS\n---------\nThe angular location of points on the cylinder where p = p_inf is\n                    theta = %3.1f, %3.2f, %1.2f, %3.1f",theta_2(1),theta_2(2),theta_2(3),theta_2(4))

printf("\nRESULTS\n---------\nThe value of Cp at the bottom of the cylinder is\n                    Cp = %1.2f",Cp(270))
```

Scilab code Exa 3.16 Calculation of lift per unit span of the cylinder

```
//All the quantities are expressed in SI units

rho_inf = 0.90926;          //density of air at 3km altitude
V_theta = -75;              //maximum velocity on the surface of the cylinder
V_inf = 25;                 //freestream velocity
R = 0.25;                   //radius of the cylinder

//thus the circulation can be calculated as
tow = -2*%pi*R*(V_theta+2*V_inf);

//and the lift per unit span is given as
L = rho_inf*V_inf*tow;

printf("\nRESULTS\n---------\nThe Lift per unit span for the given cylinder is\n                    L'' = %3.1f N",L)
```

Chapter 4

Incompressible Flow over Airfoils

Scilab code Exa 4.1 Calculation of angle of attack and drag per unit span of a NACA 2412 airfoil

```
//All the quantities are expressed in SI units

c = 0.64;                                   //chord
   length of the airfoil
V_inf = 70;                                 //
   freestream velocity
L_dash = 1254;                              //lift per
   unit span L'
rho_inf = 1.23;                             //density
   of air
mu_inf = 1.789e-5;                          //
   freestream coefficient of viscosity
q_inf = 1/2*rho_inf*V_inf*V_inf;            //
   freestream dynamic pressure

//thus the lift coefficient can be calculated as
c_l = L_dash/q_inf/c;

```

```
//for this value of C_l, from fig. 4.10
alpha = 4;

//the Reynold's number is given as
Re = rho_inf*V_inf*c/mu_inf;

//for the above Re and alpha values, from fig. 4.11
c_d = 0.0068;

//thus the drag per unit span can be calculated as
D_dash = q_inf*c*c_d;

printf("\nRESULTS\n---------\n\nc_l = %1.2f , for
    this c_l value, from fig. 4.10we get\nalpha = %1
    .0f\n------------\n\nRe = %1.2f x 10^6, for this
    value of Re, from fig. 4.11 we get\nc_d = %1.4f\
    nD'' = %2.1f N/m\n",c_l,alpha,Re/1000000,c_d,
    D_dash)
```

Scilab code Exa 4.2 Calculation of moment per uint span about the aerodynamic center of a NACA 2412 airfoil

```
//All the quantities are expressed in SI units

c = 0.64;                                   //chord
    length of the airfoil
V_inf = 70;                                 //
    freestream velocity
rho_inf = 1.23;                             //density
    of air
q_inf = 1/2*rho_inf*V_inf*V_inf;            //
    freestream dynamic pressure
c_m_ac = -0.05                              //moment
    coefficient about the aerodynamic center as seen
    from fig. 4.11
```

```

//thus moment per unit span about the aerodynamic
    center is given as
M_dash = q_inf*c*c*c_m_ac;

printf("\nRESULTS\n---------\nThe Moment per unit
    span about the aerodynamic center is is\n
                 M'' = %2.1f Nm\n",M_dash)
```

Scilab code Exa 4.3 Compare lift to drag ratios at different angle of attacks for a NACA 2412 airfoil for a given Reynolds number

```
//All the quantities are expressed in SI units

alpha = [0 4 8 12];
c_l = [0.25 0.65 1.08 1.44];
c_d = [0.0065 0.0070 0.0112 0.017];

for i = 1:4
    L_by_D(i) = c_l(i)/c_d(i);
end

temp = [alpha' c_l' c_d' L_by_D];
printf("\nRESULTS\n---------\n alpha     Cl      Cd
           Cl/Cd")
disp(temp)
```

Scilab code Exa 4.4 Calculation of lift and moment coefficients for a thin flat plate at a given angle of attack

```
//All the quantities are expressed in SI units

```

```
alpha = 5*%pi/180;              //angle of attack in radians

//from eq.(4.33)according to the thin plate theory, the lift coefficient is given by
c_l = 2*%pi*alpha;

//from eq.(4.39) the coefficient of moment about the  leading edge is given by
c_m_le = -c_l/4;

//from eq.(4.41)
c_m_qc = 0;

//thus the coefficient of moment about the trailing can be calculated as
c_m_te = 3/4*c_l;

printf("\nRESULTS\n---------\n(a)\n  Cl = %1.4f\n(b)\n  Cm_le = %1.3f\n(c)\n  Cm_c/4 = %1.0f\n(d)\n  Cm_te = %1.3f\n",c_l,c_m_le,c_m_qc,c_m_te)
```

Scilab code Exa 4.5 Calculation of diiferent attributes of an airfoil using thin airfoil theory for a cambered airfoil

```
//all the quantities are expressed in SI units

//(a)
//the slope function in terms of theta is given as
function temp = dz_by_dx(theta)
    if (theta>=0) & (theta<=0.9335) then
        temp = 0.684 - 2.3736*cos(theta)+1.995*(cos(theta)^2);
    elseif (theta<=%pi) & (theta>0.9335) then
        temp = -0.02208;
```

```
    else
        temp = 0;
    end
endfunction

//the integration function fot alpha,L=0 is thus
   given as
function temp = integ1(theta)
    temp = dz_by_dx(theta)*(cos(theta)-1);
endfunction

//from eq.(4.61)
alpha_L0 = -1/%pi*intg(0,%pi,integ1);

//(b)
alpha = 4*%pi/180;

//from eq.(4.60)
c_l = 2*%pi*(alpha-alpha_L0);

//(c)
//the integration function for A1 is given by
function temp = integ2(theta)
    temp = dz_by_dx(theta)*cos(theta);
endfunction

//thus
A1 = 2/%pi*intg(0,%pi,integ2);

//the integration function for A2 is given by
function temp = integ3(theta)
    temp = dz_by_dx(theta)*cos(2*theta);
endfunction

//thus
A2 = 2/%pi*intg(0,%pi,integ3);

//from eq.(4.64), the moment coefficient about the
```

```
quarter chord (c/4) is given as
c_m_qc = %pi/4*(A2-A1);

//(d)
//from eq.(4.66)
x_cp_by_c = 1/4*(1+%pi/c_l*(A1-A2));

printf("\nRESULTS\n--------\n(a)\n        alpha_L=0 =
    %1.2f degrees\n(b)\n          cl = %1.3f\n(c)\n
          cm,c/4 = %1.4f\n(d)\n          x_cp/c = %1.3
    f",alpha_L0*180/%pi,c_l,c_m_qc,x_cp_by_c)
```

Scilab code Exa 4.6 Calculation of location of aerodynamic center for a NACA 23012 airfoil

```
//All the quantities are expressed in SI units

alpha1 = 4;
alpha2 = -1.1;
alpha3 = -4;
cl_1 = 0.55;                //cl at alpha1
cl_2 = 0;                   //cl at alpha2
c_m_qc1 = -0.005;           //c_m_qc at alpha1
c_m_qc3 = -0.0125;          //c_m_qc at alpha3

//the lift slope is given by
a0 = (cl_1 - cl_2)/(alpha1-alpha2);

//the slope of moment coefficient curve is given by
m0 = (c_m_qc1 - c_m_qc3)/(alpha1-alpha3);

//from eq.4.71
x_ac = -m0/a0 + 0.25;

printf("\nRESULTS\n---------\nThe location of the
```

```
aerodynamic center is\n          x_ac = %1.3f\n
",x_ac)
```

Scilab code Exa 4.7 Calculation of laminar boundary layer thickness and the net laminar skin friction drag coefficient for a NACA 2412 airfoil

```
//All the quantities are expressed in SI units

c = 1.5;                //airfoil chord
Re_c = 3.1e6;           //Reynolds number at trailing edge

//from eq.(4.84), the laminar boundary layer thickness at trailing edge is given by
delta = 5*c/sqrt(Re_c);

//from eq(4.86)
Cf = 1.328/sqrt(Re_c);

//the net Cf for both surfaces is given by
Net_Cf = 2*Cf;

printf("\nRESULTS\n---------\n(a)\n    delta = %1.5f m\n----------\n(b)\n    Cf = %1.2f x 10^-4\n    Net Cf = %1.4f",delta,Cf*10000,Net_Cf)
```

Scilab code Exa 4.8 Calculation of turbulent boundary layer thickness and the net turbulent skin friction drag coefficient for a NACA 2412 airfoil

```
//All the quantities are expressed in SI units

c = 1.5;                //airfoil chord
```

```
Re_c = 3.1e6;          //Reynolds number at trailing edge

//from eq.(4.87), the turbulent boundary layer thickness at trailing edge is given by
delta = 0.37*c/(Re_c^0.2);

//from eq(4.86)
Cf = 0.074/(Re_c^0.2);

//the net Cf for both surfaces is given by
Net_Cf = 2*Cf;

printf("\nRESULTS\n---------\n(a)\n    delta = %1.4f m\n----------\n(b)\n    Cf = %1.5f\n    Net Cf = %1.5f",delta,Cf,Net_Cf)
```

Scilab code Exa 4.9 Calculation of net skin friction drag coefficient for NACA 2412 airfoil

```
//All the quantities are expressed in SI units

c = 1.5;                   //airfoil chord length
Rex_cr = 5e5;              //critical Reynold's number
Re_c = 3.1e6;              //Reynold's number at the trailing edge

//the point of transition is given by
x1 = Rex_cr/Re_c*c;

//the various skin friction coefficients are given as
Cf1_laminar = 1.328/sqrt(Rex_cr);
Cfc_turbulent = 0.074/(Re_c^0.2);
Cf1_turbulent = 0.074/(Rex_cr^0.2);
```

```

//thus the total skin friction coefficient is given by
Cf = x1/c*Cf1_laminar + Cfc_turbulent - x1/c*Cf1_turbulent;

//taking both sides of plate into account
Net_Cf = 2*Cf;

printf("\nRESULTS\n---------\nThe net skin friction coefficient is\n          Net Cf = %1.4f",Net_Cf)
```

Scilab code Exa 4.10 Calculation of net skin friction drag coefficient for NACA 2412 airfoil

```
//All the quantities are expressed in SI units

c = 1.5;                    //airfoil chord length
Rex_cr = 1e6;               //critical Reynold's number
Re_c = 3.1e6;               //Reynold's number at the trailing edge

//the point of transition is given by
x1 = Rex_cr/Re_c*c;

//the various skin friction coefficients are given as
Cf1_laminar = 1.328/sqrt(Rex_cr);                    //this is a mistake in the book in calulation of this quantity thus the answer in book is wrong
Cfc_turbulent = 0.074/(Re_c^0.2);
Cf1_turbulent = 0.074/(Rex_cr^0.2);

//thus the total skin friction coefficient is given by
```

```
Cf = x1/c*Cf1_laminar + Cfc_turbulent - x1/c*
    Cf1_turbulent;

//taking both sides of plate into account
Net_Cf = 2*Cf;

printf("\nRESULTS\n---------\nThe net skin friction
    coefficient is\n          Net Cf = %1.5f",Net_Cf)
```

Chapter 5

Incompressible Flow over Finite Wings

Scilab code Exa 5.1 Calculation of lift and induced drag coefficients for a finite wing

```
//All the quantities are expressed in SI units

AR = 8;                    //Aspect ratio of the wing
alpha = 5*%pi/180;                  //Angle of attack
   experienced by the wing
a0 = 2*%pi                 //airfoil lift curve slope
alpha_L0 = 0;              //zero lift angle of attack
   is zero since airfoil is symmetric

//from fig. 5.20, for AR = 8 and taper ratio of 0.8
delta = 0.055;
tow = delta;               //given assumption

//thus the lift curve slope for wing is given by
a = a0/(1+(a0/%pi/AR/(1+tow)));

//thus C_l can be calculated as
C_l = a*alpha;
```

```

//from eq.(5.61)
C_Di = C_l^2/%pi/AR*(1+delta);

printf("\nRESULTS\n---------\n          Cl = %1.4f\n\n          CD,i = %1.5f",C_l,C_Di)
```

Scilab code Exa 5.2 Calculation of induced drag coefficient for a finite wing

```
//All the quantities are expressed in SI units

CDi1 = 0.01;                        //induced drag coefficient for first wing
delta = 0.055;                      //induced drag factor for both wings
tow = delta;
alpha_L0 = -2*%pi/180;              //zero lift angle of attack
alpha = 3.4*%pi/180;                //angle of attack
AR1 = 6;                            //Aspect ratio of the first wing
AR2 = 10;                           //Aspect ratio of the second wing

//from eq.(5.61), lift coefficient can be calculated as
C_l1 = sqrt(%pi*AR1*CDi1/(1+delta));

//the lift slope for the first wing can be calculated as
a1 = C_l1/(alpha-alpha_L0);

//the airfoil lift coefficient can be given as
```

```
a0 = a1/(1-(a1/%pi/AR1*(1+tow)));

//thus the list coefficient for the second wing which has the same airfoil is given by
a2 = a0/(1+(a0/%pi/AR2*(1+tow)));
C_l2 = a2*(alpha-alpha_L0);
CDi2 = C_l2^2/%pi/AR2*(1+delta);

printf("\nRESULTS\n---------\nThe induced drag coefficient of the second wing is\n      CD,i = %1.4f",CDi2)
```

Scilab code Exa 5.3 Calculation of angle of attack of an airplane at cruising conditions

```
//all the quantities are expressed in SI units

a0 = 0.1*180/%pi;                    //airfoil lift curve slope
AR = 7.96;                    //Wing aspect ratio
alpha_L0 = -2*%pi/180;                    //zero lift angle of attack
tow = 0.04;                    //lift efficiency factor
C_l = 0.21;                    //lift coefficient of the wing

//the lift curve slope of the wing is given by
a = a0/(1+(a0/%pi/AR/(1+tow)));

//thus angle of attack can be calculated as
alpha = C_l/a + alpha_L0;

printf("\nRESULTS\n---------\n      alpha = %1.1f degrees\n",alpha*180/%pi)
```

Scilab code Exa 5.4 Calculation of lift and drag coefficients for a Beechcraft Baron 58 aircraft wing

```
//All the qunatities are expressed in SI units

alpha_L0 = -1*%pi/180;                      //zero lift
   angle of attack
alpha1 = 7*%pi/180;                         //reference
   angle of attack
C_l1 = 0.9;                                 //wing lift
   coefficient at alpha1
alpha2 = 4*%pi/180;
AR = 7.61;                                  //aspect
   ratio of the wing
taper = 0.45;                               //taper ratio
    of the wing
delta = 0.01;                               //delta as
   calculated from fig. 5.20
tow = delta;

//the lift curve slope of the wing/airfoil can be
   calculated as
a0 = C_l1/(alpha1-alpha_L0);

e = 1/(1+delta);

//from eq. (5.70)
a = a0/(1+(a0/%pi/AR/(1+tow)));

//lift coefficient at alpha2 is given as
C_l2 = a*(alpha2 - alpha_L0);

//from eq.(5.42), the induced angle of attack can be
    calculated as
```

```
alpha_i = C_l2/%pi/AR;

//which gives the effective angle of attack as
alpha_eff = alpha2 - alpha_i;

//Thus the airfoil lift coefficient is given as
c_l = a0*(alpha_eff-alpha_L0);

c_d = 0.0065;                              //section drag
    coefficient for calculated c_l as seen from fig.
    5.2b

//Thus the wing drag coefficient can be calculated
   as
C_D = c_d + ((C_l2^2)/%pi/e/AR);

printf("\nRESULTS\n---------\nThe drag coefficient of
    the wing is\n          C_D = %1.4f\n",C_D)
```

Chapter 7

Compressible Flow Some Preliminary Aspects

Scilab code Exa 7.1 Calculation of internal energy and enthalpy of air in a room

```
//All the quantities are expressed in SI units

l = 5;                  //dimensions of the room
b = 7;
h = 3.3;
V = l*b*h;              //volume of the room
p = 101000;             //ambient pressure
T = 273 + 25;           //ambient temperature
R = 287;                //gas constant
gam = 1.4;              //ratio of specific heats
cv = R/(gam-1);
cp = gam*R/(gam-1);

//the density can by calculaled by the ideal gas law
rho = p/R/T;

//thus the mass is given by
M = rho*V;
```

```

//from eq.(7.6a), the internal energy per unit mass
    is
e = cv*T;

//thus internal energy in the room is
E = e*M;

//from eq.(7.6b), the enthalpy per unit mass is
    given by
h = cp*T;

//Thus the enthalpy in the room is
H = M*h;

printf("\nRESULTS\n---------\nThe internal energy in
    the room is:\n         E = %1.2f x 10^7 J\n\nThe
    Enthalpy in the room is:\n         H = %1.2f x
    10^7 J\n",E/10^7,H/10^7 )
```

Scilab code Exa 7.2 Calculation of temperature at a point on the Boeing 747 wing

```
//All the quantities are expressed in SI units

p_inf = 22790.9;            //ambient pressure at
    36000 ft
T_inf = 217.2;              //ambient temperature at
     36000 ft
p = 19152;                  //pressure at the given
    point
gam = 1.4;

//thus the temperature at the given point can be
    calculated by eq.(7.32) as
```

```
T = T_inf*((p/p_inf)^((gam-1)/gam));

printf("\nRESULTS\n---------\nThe temperature at the given point is:\n          T = %3.1f K\n",T)
```

Scilab code Exa 7.3 Calculation of total temperature and total pressure at a point in the flow

```
//All the quantities are expressed in SI units

p =101000;                  //static pressure
T = 320;                    //static temperature
v = 1000;                   //velocity
gam = 1.4;                  //ratio of specific heats
R = 287;                    //universal gas constant
cp = gam*R/(gam-1);         //specific heat at constant pressure

//from eq.(7.54), the total temperature is given by
T0 = T + (v^2)/2/cp;

//from eq.(7.32),the total pressure is given by
p0 = p*((T0/T)^(gam/(gam-1)));

p0_atm = p0/101000;


printf("\nRESULTS\n---------\nThe total temperature and pressure are given by:\n          T0 = %3.1f K\n\n          P0 = %2.1f atm\n",T0,p0_atm)
```

Chapter 8

Normal Shock Waves and Related Topics

Scilab code Exa 8.1 Calculation of Mach number at different flying altitudes

```
//All the quantities are expressed in SI units

R = 287;
gam = 1.4;
V_inf = 250;

//(a)
//At sea level
T_inf = 288;

//the velocity of sound is given by
a_inf = sqrt(gam*R*T_inf);

//thus the mach number can be calculated as
M_inf = V_inf/a_inf;

printf("\n(a)\nThe Mach number at sea level is:\n
        M_inf = %1.3f\n",M_inf)
```

```
//similarly for (b) and (c)
//(b)
//at 5km
T_inf = 255.7;

a_inf = sqrt(gam*R*T_inf);

M_inf = V_inf/a_inf;

printf("\n(b)\nThe Mach number at 5 km is:\n M_inf = %1.2f\n",M_inf)

//(c)
//at 10km
T_inf = 223.3;

a_inf = sqrt(gam*R*T_inf);

M_inf = V_inf/a_inf;

printf("\n(c)\nThe Mach number at 10 km is:\n M_inf = %1.3f\n",M_inf)
```

Scilab code Exa 8.2 Calculation of Mach number at a given point

```
//All the quantities are expressed in SI units

T = 320;               //static temperature
V = 1000;              //velocity
gam = 1.4;             //ratio of specific heats
R = 287;               //universal gas constant

//the speed of sound is given by
a = sqrt(gam*R*T);
```

```
//the mach number can be calculated as
M = V/a;

printf("\nRESULTS\n----------\nThe Mach number is:\n M = %1.2f\n",M)
```

Scilab code Exa 8.3 Calculation of ratio of kinetic energy to internal energy at a point in an airflow for given mach numbers

```
//All the quantities are expressed in SI units

gam = 1.4;                    //ratio of specific heats

//(a)
M = 2;                        //Mach number

//the ratio of kinetic energy to internal energy is given by
ratio = gam*(gam-1)*M*M/2;

printf("\n(a)\nThe ratio of kinetic energy to internal energy is:\n\n                    %1.2f\n",ratio)

//similarly for (b)
//(b)
M = 20;

ratio = gam*(gam-1)*M*M/2;

printf("\n(b)\nThe ratio of kinetic energy to internal energy is:\n\n                    %3.0f\n",ratio)
```

Scilab code Exa 8.4 Calculation of total temperature and total pressure at a point in the flow

```
//All the quantities are expressed in SI units

M = 2.79;          //Mach number
T = 320;           //static temperature from ex. 7.3
p = 1;             //static pressure in atm
gam = 1.4;

//from eq. (8.40)
T0 = T*(1+((gam-1)/2*M*M));

//from eq. (8.42)
p0 = p*((1+((gam-1)/2*M*M))^(gam/(gam-1)));

printf("\nRESULTS\n----------\nThe total temperature
   and pressure are:\n        T0 = %3.0f K\n
   P0 = %2.1f atm\n",T0,p0)
```

Scilab code Exa 8.5 Calculation of local stagnation temperature and pressure speed of sound and mach number at the given point

```
//All the quantities are expressed in SI units

M = 3.5;          //Mach number
T = 180;           //static temperature from ex. 7.3
p = 0.3;             //static pressure in atm
gam = 1.4;
R = 287;

```

```
//from eq. (8.40)
T0 = T*(1+((gam-1)/2*M*M));

//from eq. (8.42)
p0 = p*((1+((gam-1)/2*M*M))^(gam/(gam-1)));

a = sqrt(gam*R*T);
V = a*M;

//the values at local sonic point are given by
T_star = T0*2/(gam+1);
a_star = sqrt(gam*R*T_star);
M_star = V/a_star;

printf("\nRESULTS\n----------\n          T0 = %3.0f K\n
          P0 = %2.1f atm\n          T* = %3.1f k\n
          a* = %3.0f m/s\n          M* = %1.2f",T0,p0,
   T_star,a_star,M_star)
```

Scilab code Exa 8.6 Calculation of local mach number at the given point on the airfoil

```
//All the quantities are expressed in SI units

p_inf = 1;
p1 = 0.7545;
M_inf = 0.6;
gam = 1.4;

//from eq. (8.42)
p0_inf = p_inf*((1+((gam-1)/2*M_inf*M_inf))^(gam/(
   gam-1)));

p0_1 = p0_inf;

```

```
//from eq. (8.42)
ratio = p0_1/p1;

//from appendix A, for this ratio, the Mach number is
M1 = 0.9;

printf("\nRESULTS\n---------\nThe mach number at the given point is:\n          M1 = %1.1f\n",M1)
```

Scilab code Exa 8.7 Calculation of velocity on a point on the airfoil for compressible flow

```
//All the quantities are expressed in SI units

T_inf = 288;          //freestream temperature
p_inf = 1;            //freestream pressure
p1 = 0.7545;          //pressure at point 1
M = 0.9;              //mach number at point 1
gam = 1.4;            //ratio of specific heats

//for isentropic flow, from eq. (7.32)
T1 = T_inf*((p1/p_inf)^((gam-1)/gam));

//the speed of sound at that point is thus
a1 = sqrt(gam*R*T1);

//thus, the velocity can be given as
V1 = M*a1;

```

```
printf("\nRESULTS\n----------\nThe velocity at the given point is:\n          V1 = %3.0f m/s\n",V1)
```

Scilab code Exa 8.8 Calculation of velocity temperature and pressure downstream of a shock

```
//All the quantities are expressed in SI units

u1 = 680;                                 //velocity upstream of shock
T1 = 288;                                 //temperature upstream of shock
p1 = 1;                                   //pressure upstream of shock
gam = 1.4;                                //ratio of specific heats
R = 287;                                  //universal gas constant

//the speed of sound is given by
a1 = sqrt(gam*R*T1)

//thus the mach number is
M1 = 2;

//from Appendix B, for M = 2, the relations between pressure and temperature are given by
pressure_ratio = 4.5;                     //ratio of pressure accross shock
temperature_ratio = 1.687;                //ratio of temperature accross shock
M2 = 0.5774;                              //mach number downstream of shock

//thus the values downstream of the shock can be
```

```
    calculated as
p2 = pressure_ratio*p1;
T2 = temperature_ratio*T1;
a2 = sqrt(gam*R*T2);
u2 = M2*a2;

printf("\nRESULTS\n----------\n          p2 = %1.1f atm
    \n          T2 = %3.0f K\n          u2 = %3.0f m/s",
    p2,T2,u2)
```

Scilab code Exa 8.9 Calculation of loss of total pressure across a shock wave for given values of mach number

```
//All the quantities are expressed in SI units

p1 = 1;
    //ambient pressure upstream of shock


//(a)
//for M = 2;
p0_1 = 7.824*p1;
    //total pressure upstream of shock
pressure_ratio = 0.7209;
    //ratio of total pressure accross the shock
p0_2 = pressure_ratio*p0_1;
    //total pressure downstream of shock

//thus the total loss of pressure is given by
pressure_loss = p0_1 - p0_2;

printf("\nRESULTS\n----------\nThe total pressure
    loss is:\n(a)          P0_loss = %1.3f atm\n",
    pressure_loss)

```

```
//similarly
//(b)
//for M = 4;
p0_1 = 151.8*p1;
pressure_ratio = 0.1388;
p0_2 = pressure_ratio*p0_1;

//thus the total loss of pressure is given by
pressure_loss = p0_1 - p0_2;

printf("\n(b)          P0_loss = %3.1f atm\n", pressure_loss)
```

Scilab code Exa 8.10 Calculation of air temperature and pressure for a given value of local mach number

```
//All the quantities are expressed in SI units

M_inf = 2;                    //freestream mach number
p_inf = 2.65e4;               //freestream pressure
T_inf = 223.3;                //freestream temperature

//from Appendix A, for M = 2
p0_inf = 7.824*p_inf;         //freestream total pressure
T0_inf = 1.8*T_inf;           //freestream total temperature

//from Appendix B, for M = 2
p0_1 = 0.7209*p0_inf;         //total pressure downstream of the shock
T0_1 = T0_inf;                //total temperature accross the shock is conserved
```

```

//since the flow downstream of the shock is isentropic
p0_2 = p0_1;
T0_2 = T0_1;

//from Appendix A, for M = 0.2 at point 2
p2 = p0_2/1.028;
T2 = T0_2/1.008;

p2_atm = p2/102000;

printf("\nRESULTS\n----------\nThe pressure at point 2 is:\n          p2 = %1.2f atm\n",p2_atm)
```

Scilab code Exa 8.11 Calculation of air temperature and pressure for a given value of local mach number

```
//All the quantities are expressed in SI units

M_inf = 10;                      //freestream mach number
p_inf = 2.65e4;                  //freestream pressure
T_inf = 223.3;                   //freestream temperature

//from Appendix A, for M = 2
p0_inf = 0.4244e5*p_inf;         //freestream total pressure
T0_inf = 21*T_inf;               //freestream total temperature

//from Appendix B, for M = 2
p0_1 = 0.003045*p0_inf;          //total pressure downstream of shock
```

```
T0_1 = T0_inf;                          //total temperature downstream of shock is conserved

//since the flow downstream of the shock is isentropic
p0_2 = p0_1;
T0_2 = T0_1;

//from Appendix A, for M = 0.2 at point 2
p2 = p0_2/1.028;
T2 = T0_2/1.008;

p2_atm = p2/102000;


printf("\nRESULTS\n----------\nThe pressure at point 2 is:\n          p2 = %2.1f atm\n",p2_atm)
```

Scilab code Exa 8.13 Calculation of stagnation pressure at the stagnation point on the nose for a hypersonic missile

```
//All the quantities are expressed in SI units

p1 = 4.66e4;                                        //ambient pressure
M = 8;                                              //mach number

//from Appendix B, for M = 8
p0_2 = 82.87*p1;                                    //total pressure downstream of the shock

//since the flow is isentropic downstream of the shock, total pressure is conserved
ps_atm = p0_2/101300;                               //
```

```
pressure at the stagnation point

printf("\nRESULTS\n----------\nThe pressure at the nose is:\n        p_s = %2.1f atm\n",ps_atm)
```

Scilab code Exa 8.14 Calculation of velocity of a Lockheed SR71 Blackbird at given flight conditions

```
//All the quantities are expressedin SI units

p1 = 2527.3;                        //ambient pressure at the altitude of 25 km
T1 = 216.66;                        //ambient temperature at the altitude of 25 km
p0_1 = 38800;                       //total pressure
gam = 1.4;                          //ratio of specific heats
R = 287;                            //universal gas constant
pressure_ratio = p0_1/p1;           //ratio of total to static pressure

//for this value of pressure ratio, mach number is
M1 = 3.4;

//the speed of sound is given by
a1 = sqrt(gam*R*T1)

//thus the velocity can be calculated as
V1 = M1*a1;

printf("\nRESULTS\n----------\nThe Velocity of the airplane is:\n        V1 = %4.0f m/s\n",V1)
```

Chapter 9

Oblique Shock and Expansion Waves

Scilab code Exa 9.1 Calculation of the horizontal distance between a supersonic aircraft from a bystander at the instant he hears the sonic boom from the aircraft

```
//All the quantities are expressed in SI units

M = 2;                          //mach number
h = 16000;                          //altitude of the plane

//the mach angle can be calculated from eq.(9.1) as
mue = asin(1/M);                //mach angle

d = h/tan(mue);

printf("\nRESULTS\n----------\nThe plane is ahead of the bystander by a distance of:\n        d = %2.1f km\n",d/1000)
```

Scilab code Exa 9.2 Calculation of flow mach number pressure temperature and stagnation pressure and temperature just behind an oblique shock wave

```
//All the quantities are expressed in SI units

M1 = 2;                                   //mach number
p1 = 1;                                    //ambient pressure
T1 = 288;                                  //ambient temperature
theta = 20*%pi/180;                       //flow deflection

//from figure 9.9, for M = 2, theta = 20
b = 53.4*%pi/180;                         //beta
Mn_1 = M1*sin(b);                         //upstream mach number normal to shock

//for this value of Mn,1 = 1.60, from Appendix B we have
Mn_2 = 0.6684;                            //downstream mach number normal to shock
M2 = Mn_2/sin(b-theta);                   //mach number downstream of shock
p2 = 2.82*p1;
T2 = 1.388*T1;

//for M = 2, from appendix A we have
p0_2 = 0.8952*7.824*p1;
T0_1 = 1.8*T1;
T0_2 = T0_1;

printf("\nRESULTS\n----------\n          M2 = %1.2f\n          p2 = %1.2f atm\n          T2 = %3.1f K\n          p0,2 = %1.2f atm\n          T0,2 = %3.1f K", M2,p2,T2,p0_2,T0_2)
```

Scilab code Exa 9.3 Calculation of deflection angle of the flow and the pressure and temperature ratios accross the shock wave and the mach number the wave

```
//All the quantities are expressed in SI units

b = 30*%pi/180;                                  //oblique
   shock wave angle
M1 = 2.4;                                        //upstream
   mach number

//from figure 9.9, for these value of M and beta, we
    have
theta = 6.5*%pi/180;

Mn_1 = M1*sin(b);                                //upstream
   mach number normal to shock

//from Appendix B
pressure_ratio = 1.513;
temperature_ratio = 1.128;
Mn_2 = 0.8422;

M2 = Mn_2/sin(b-theta);

printf("\nRESULTS\n---------\n        theta = %1.1f
   degrees\n         p2/p1 = %1.3f\n          T2/T1 =
   %1.3f\n          M2 = %1.2f\n",theta*180/%pi,
   pressure_ratio,temperature_ratio,M2)
```

Scilab code Exa 9.4 Calculation of mach number upstream of an oblique shock

```
//All the quantities are expressed in SI units

b = 35*%pi/180;                        //oblique shock wave angle
pressure_ratio = 3;                    //upstream and downstream pressure ratio

//from appendix B
Mn_1 = 1.64;
M1 = Mn_1/sin(b);

printf("\nRESULTS\n----------\nThe upstream mach number is:\n          M = %1.2f\n",M1)
```

Scilab code Exa 9.5 Calculation of the final total pressure values for the two given cases

```
//All the quantities are expressed in SI units

M1 = 3;
b = 40*%pi/180;

//for case 1, for M = 3, from Appendix B, we have
p0_ratio_case1 = 0.3283;

//for case 2
Mn_1 = M1*sin(b);

//from Appendix B
p0_ratio1 = 0.7535;
Mn_2 = 0.588;

//from fig. 9.9, for M1 = 3 and beta = 40, we have
theta = 22*%pi/180;
M2 = Mn_2/sin(b-theta);
```

```

//from appendix B for M = 1.9; we have
p0_ratio2 = 0.7674;
p0_ratio_case2 = p0_ratio1*p0_ratio2;

ratio = p0_ratio_case2/p0_ratio_case1;

printf("\nRESULTS\n----------\n          Ans = %1.2f\n"
    ,ratio)
```

Scilab code Exa 9.6 Calculation of the drag coefficient of a wedge in a hypersonic flow

```
//All the quantities are expressed in SI units

M1 = 5;
theta = 15*%pi/180;
gam = 1.4;

//for these values of M and theta, from fig. 9.9
b = 24.2*%pi/180;
Mn_1 = M1*sin(b);

//from Appendix B, for Mn,1 = 2.05, we have
p_ratio = 4.736;

//hence
c_d = 4*tan(theta)/gam/(M1^2)*(p_ratio-1);

printf("\nRESULTS\n----------\nThe drag coefficient
    is given by:\n          cd = %1.3f\n",c_d)
```

Scilab code Exa 9.7 Calculation of the angle of deflected shock wave related to the straight wall and the pressure temperature and mach number behind the reflected wave

```
//All the quantities are expressed in SI units

M1 = 3.5;
theta1 = 10*%pi/180;
gam = 1.4;
p1 = 101300;
T1 = 288;

//for these values of M and theta, from fig. 9.9
b1 = 24*%pi/180;
Mn_1 = M1*sin(b);

//from Appendix B, for Mn,1 = 2.05, we have
Mn_2 = 0.7157;
p_ratio1 = 2.32;
T_ratio1 = 1.294;
M2 = Mn_2/sin(b1-theta1);

//now
theta2 = 10*%pi/180;

//from fig. 9.9
b2 = 27.3*%pi/180;
phi = b2 - theta2;

//from Appendix B
p_ratio2 = 1.991;
T_ratio2 = 1.229;
Mn_3 = 0.7572;
M3 = Mn_3/sin(b2-theta2);

//thus
p3 = p_ratio1*p_ratio2*p1;
T3 = T_ratio1*T_ratio2*T1;
```

```

printf("\nRESULTS\n----------\n        p3 = %1.2f x 10^5 N/m2\n        T3 = %3.0f K\n",p3/1e5,T3)
```

Scilab code Exa 9.8 Calculation of mach number pressure temperature and stagnation pressure temperature and mach line angles behind an expansion wave

```
//All the quantities are expressed in SI units

M1 = 1.5;                          //upstream mach number
theta = 15*%pi/180;                //deflection angle
p1 = 1;                            //ambient pressure in atm
T1 = 288;                          //ambient temperature

//from appendix C, for M1 = 1.5 we have
v1 = 11.91*%pi/180;

//from eq.(9.43)
v2 = v1 + theta;

//for this value of v2, from appendix C
M2 = 2;

//from Appendix A for M1 = 1.5 and M2 = 2.0, we have
p2 = 1/7.824*1*3.671*p1;
T2 = 1/1.8*1*1.45*T1;
p0_1 = 3.671*p1;
p0_2 = p0_1;
T0_1 = 1.45*T1;
T0_2 = T0_1;

```

```
//from fig. 9.25, we have
fml = 41.81;                          //Angle of forward Mach line
rml = 30 - 15;                        //Angle of rear Mach line

printf("\nRESULTS\n----------\n     p2 = %1.3f atm\n     T2 = %3.0f K\n     p0,2 = %1.3f atm\n     T0,2 = %3.1f K\n     Angle of forward Mach line = %2.2f degrees\n     Angle of rear Mach line = %2.0f degrees",p2,T2,p0_2,T0_2,fml,rml)
```

Scilab code Exa 9.9 Calculation of mach number and pressure behind a compression wave

```
//All the quantities are expressed in SI units

M1 = 10;                              //upstream mach number
theta = 15*%pi/180;                    //deflection angle
p1 = 1;                                //ambient pressure in atm

//from appendix C, for M1 = 10 we have
v1 = 102.3*%pi/180;

//in region 2
v2 = v1 - theta;

//for this value of v2, from appendix C
M2 = 6.4;

//from Appendix A for M1 = 10 and M2 = 6.4, we have
p2 = 1/(2355)*1*42440*p1;
```

```

printf("\nRESULTS\n----------\n            M2 = %1.1f\n            p2 = %2.2f  atm\n",M2,p2)
```

Scilab code Exa 9.10 Calculation of mach number static pressure and stagnation pressure behind an oblique shock wave

```
//All the quantities are expressed in SI units

M1 = 10;                              //upstream mach number
theta = 15*%pi/180;                    //deflection angle
p1 = 1;                                //ambient pressure in atm

//from fig 9.9, for M1 = 10 and theta = 15 we have
b = 20*%pi/180;
Mn_1 = M1*sin(b);

//from Appendix B, for Mn,1 = 3.42
Mn_2 = 0.4552;
M2 = Mn_2/sin(b-theta);
p2 = 13.32*p1;

//from Appendix A, for M1 = 10
p0_2 = 0.2322*42440*p1;

printf("\nRESULTS\n----------\n            M2 = %1.2f\n            p2 = %2.2f  atm\n            p0,2 = %1.2f x 10^3 atm\n",M2,p2,p0_2/1e3)
```

Scilab code Exa 9.11 Calculation of the lift and drag coefficients of a flat plate in a supersonic flow

```
//All the quantities are expressed in SI units

M1 = 3;                              //upstream mach number
theta = 5*%pi/180;                   //deflection angle
alpha = theta;                       //angle of attack
gam = 1.4;

//from appendix C, for M1 = 3 we have
v1 = 49.76*%pi/180;

//from eq.(9.43)
v2 = v1 + theta;

//for this value of v2, from appendix C
M2 = 3.27;

//from Appendix A for M1 = 3 and M2 = 3.27, we have
p_ratio1 = 36.73/55;

//from fig. 9.9, for M1 = 3 and theta = 5
b = 23.1*%pi/180;
Mn_1 = M1*sin(b);

//from Appendix B
p_ratio2 = 1.458;

//thus
c_l = 2/gam/(M1^2)*(p_ratio2-p_ratio1)*cos(alpha);

c_d = 2/gam/(M1^2)*(p_ratio2-p_ratio1)*sin(alpha);

printf("\nRESULTS\n----------\nThe lift and drag coefficients are given by:\n        cl = %1.3f\n        cd = %1.3f\n",c_l,c_d)
```

Chapter 10

Compressible Flow Through Nozzles Diffusers and Wind Tunnels

Scilab code Exa 10.1 Calculation of mach number pressure and temperature at the nozzle exit

```
//All the quantities are expressed in Si units

area_ratio = 10.25;                               //exit to
    throat area ratio
p0 = 5;                                           //
   reservoir pressure in atm
T0 = 333.3;                                       //
   reservoir temperature

//from appendix A, for an area ratio of 10.25
Me = 3.95;                                        //exit
   mach number
pe = 0.007*p0;                                    //exit
   pressure
Te = 0.2427*T0;                                   //exit
   temperature
```

```
printf("\nRESULTS\n----------\n             Me = %1.2f\n             pe = %1.3f atm\n             Te = %2.1f K",Me,pe,Te)
```

Scilab code Exa 10.2 Calculation of isentropic flow conditions through a CD nozzle for a supersonic and subsonic flow

```
//All the quantities are expressed in Si units

area_ratio = 2;                          //exit to throat area ratio
p0 = 1;                                  //reservoir pressure in atm
T0 = 288;                                //reservoir temperature

//(a)
//since M = 1 at the throat
Mt = 1;
pt = 0.528*p0;                           //pressure at throat
Tt = 0.833*T0;                           //temperature at throat

//from appendix A for supersonic flow, for an area ratio of 2
Me = 2.2;                                //exit mach number
pe = 1/10.69*p0;                         //exit pressure
Te = 1/1.968*T0;                         //exit temperature

printf("\nRESULTS\n----------\nAt throat:\n             Mt
```

```
 = %1.1f\n             pt = %1.3f atm\n             Tt = %3.0f K\n\nFor supersonic exit:\n             Me = %1.1f\n             pe = %1.4f atm\n             Te = %3.0f K\n",Mt,pt,Tt,Me,pe,Te)

//(b)
//from appendix A for subonic flow, for an area ratio of 2
Me = 0.3;                                  //exit mach number
pe = 1/1.064*p0;                           //exit pressure
Te = 1/1.018*T0;                           //exit temperature

printf("\nFor subrsonic exit:\n             Me = %1.1f\n             pe = %1.2f atm\n             Te = %3.1f K",Me,pe,Te)
```

Scilab code Exa 10.3 Calculation of throat and exit mach numbers for the nozzle used in previous example for the given exit pressure

```
//All the quantities are expressed in Si units

area_ratio = 2;                            //exit to throat area ratio
p0 = 1;                                    //reservoir pressure in atm
T0 = 288;                                  //reservoir temperature
pe = 0.973;                                //exit pressure in atm

p_ratio = p0/pe;                           //ratio of reservoir to exit pressure
```

```

//from appendix A for subsonic flow, for an pressure ratio of 1.028
Me = 0.2;                                    //exit mach number
area_ratio_exit_to_star = 2.964;             //A_exit/A_star

//thus
area_ratio_throat_to_star = area_ratio_exit_to_star/area_ratio;             //A_exit/A_star

//from appendix A for subsonic flow, for an area ratio of 1.482
Mt = 0.44;                                   //throat mach number

printf("\nRESULTS\n----------\n          Me = %1.1f\n          Mt = %1.2f\n",Me,Mt)
```

Scilab code Exa 10.4 Calculation of thrust for the given rocket engine and the nozzle exit area

```
//All the quantities are expressed in SI units

p0 = 30*101000;                              //reservoir pressure
T0 = 3500;                                   //reservoir temperature
R = 520;                                     //specific gas constant
gam = 1.22;                                  //ratio of specific heats
A_star = 0.4;                                //rocket nozzle throat area
```

```
pe = 5529;                                    //rocket nozzle exit pressure equal to ambient pressure at 20 km altitude

//(a)
//the density of air in the reservoir can be calculated as
rho0 = p0/R/T0;

//from eq.(8.46)
rho_star = rho0*(2/(gam+1))^(1/(gam-1));

//from eq.(8.44)
T_star = T0*2/(gam+1);
a_star = sqrt(gam*R*T_star);
u_star = a_star;
m_dot = rho_star*u_star*A_star;

//rearranging eq.(8.42)
Me = sqrt(2/(gam-1)*(((p0/pe)^((gam-1)/gam)) - 1));
Te = T0/(1+(gam-1)/2*Me*Me);
ae = sqrt(gam*R*Te);
ue = Me*ae;

//thus the thrust can be calculated as
T = m_dot*ue;
T_lb = T*0.2247;

//(b)
//rearranging eq.(10.32)
Ae = A_star/Me*((2/(gam+1)*(1+(gam-1)/2*Me*Me))^((gam+1)/(gam-1)/2));

printf("\nRESULTS\n----------\n(a)The thrust of the rocket is:\n        T = %1.2f x 10^6 N = %6.0f lb\n\n(b)\nThe nozzle exit area is:\n        Ae = %2.1f m2\n",T/1e6,T_lb,Ae)
```

Scilab code Exa 10.5 Calculation of mass flow through the rocket engine used in the previous example

```
//All the quantities are expressed in SI units

p0 = 30*101000;                          //reservoir pressure
T0 = 3500;                               //reservoir temperature
R = 520;                                 //specific gas constant
gam = 1.22;                              //ratio of specific heats
A_star = 0.4;                            //rocket nozzle throat area

//the mass flow rate using the closed form analytical expression
//from problem 10.5 can be given as
m_dot = p0*A_star*sqrt(gam/R/T0*((2/(gam+1))^((gam+1)/(gam-1))));

printf("\nRESULTS\n----------\nThe mass flow rate is :\n        m_dot = %3.1f kg/s\n",m_dot)
```

Scilab code Exa 10.6 Calculation of the ratio of diffuser throat area to the nozzle throat area for a supersonic wind tunnel

```
//All the quantities are expressed in SI units

M = 2;                          //Mach number

```

```
//for this value M, for a normal shock, from
   Appendix B
p0_ratio = 0.7209;

//thus
area_ratio = 1/p0_ratio;

printf("\nRESULTS\n---------\nThe diffuser throat to
    nozzle throat area ratio is:\n        At,2/At,1
   = %1.3f",area_ratio)
```

Chapter 11

Subsonic Compressible Flow over Airfoils Linear Theory

Scilab code Exa 11.1 Calculation of pressure coefficient on a point on an airfoil with compressibilty corrections

```
//All the quantities are expressed in SI units

Cp_incompressible = -0.3;                          //Cp for incompressible flow
M = 0.6;                                           //Mach number

//Thus from eq.(11.52)
Cp_compressible = Cp_incompressible/sqrt(1-M^2);

printf("\nRESULTS\n----------\n(a)\The Cp after compressibility corrections is:\n         Cp = %1.3f\n",Cp_compressible)
```

Scilab code Exa 11.2 Calculatiom of the lift coefficient for an airfoil with compressibility corrections

```
//All the quantities are expressed in SI units

cl_incompressible = 2*%pi;                              //lift curve slope
M_inf = 0.7;                                            //Mach number

//from eq.(11.52)
cl_compressible = cl_incompressible/sqrt(1-M_inf^2);        //compressible lift curve slope

printf("\nRESULTS\n----------\n(a)\The cl after compressibility corrections is:\n        cl = %1.1falpha\n",cl_compressible)
```

Chapter 12

Linearized Supersonic Flow

Scilab code Exa 12.1 Calculation of lift and drag coefficients for a flat plate in a supersonic flow using linearized theory

```
//All the quantities are expressed in SI units

alpha = 5*%pi/180;                          //angle of attack
M_inf = 3;                                  //freestream mach number

//from eq.(12.23)
c_l = 4*alpha/sqrt(M_inf^2 - 1);

//from eq.(12.24)
c_d = 4*alpha^2/sqrt(M_inf^2 - 1);

printf("\nRESULTS\n----------\nThe cl and cd according to the linearized theory are:\n cl = %1.3f\n          cd = %1.3f\n",c_l,c_d)
```

Scilab code Exa 12.2 Calculation of angle of attack of a Lockheed F104 wing in a supersonic flow

```
//All the quantities are expressed in SI units

M_inf = 2;                          //freestream mach number
rho_inf = 0.3648;                   //freestream density at 11 km altitude
T_inf = 216.78;                     //freestream temperature at 11 km altitude
gam = 1.4;                          //ratio of specific heats
R = 287;                            //specific gas constant
m = 9400;                           //mass of the aircraft
g = 9.8;                            //acceleratio due to gravity
W = m*g;                            //weight of the aircraft
S = 18.21;                          //wing planform area

//thus
a_inf = sqrt(gam*R*T_inf);
V_inf = M_inf*a_inf;
q_inf = 1/2*rho_inf*V_inf^2;

//thus the aircraft lift coefficient is given as
C_l = W/q_inf/S;

alpha = 180/%pi*C_l/4*sqrt(M_inf^2 - 1);

printf("\nRESULTS\n---------\nThe angle of attack of the wing is:\n         alpha = %1.2f degrees\n", alpha)
```

Scilab code Exa 12.3 Calculation of the airfoil skin friction drag coefficient and the airfoil drag coefficient for the wing used in the previous example

```
//All the quantities are expressed in SI units
//All the quantities are expressed in SI units

//(a)
M_inf = 2;                          //freestream mach number
rho_inf = 0.3648;                   //freestream density at 11 km altitude
T_inf = 216.78;                     //freestream temperature at 11 km altitude
gam = 1.4;                          //ratio of specific heats
R = 287;                            //specific gas constant
m = 9400;                           //mass of the aircraft
g = 9.8;                            //acceleratio due to gravity
W = m*g;                            //weight of the aircraft
S = 18.21;                          //wing planform area
c = 2.2;                            //chord length of the airfoil
alpha = 0.035;                      //angle of attack as calculated in ex. 12.2
T0 = 288.16;                        //ambient temperature at sea level
mue0 = 1.7894e-5;                   //reference viscosity at sea level

```

```
//thus
a_inf = sqrt(gam*R*T_inf);
V_inf = M_inf*a_inf;

//according to eq.(15.3), the viscosity at the given
    temperature is
mue_inf = mue0*(T_inf/T0)^1.5*(T0+110)/(T_inf+110);

//thus the Reynolds number can be given by
Re = rho_inf*V_inf*c/mue_inf;

//from fig.(19.1), for these values of Re and M, the
    skin friction coefficient is
Cf = 2.15e-3;

//thus, considering both sides of the flat plate
net_Cf = 2*Cf;

//(b)
c_d = 4*alpha^2/sqrt(M_inf^2 - 1);

printf("\nRESULTS\n----------\n(a)\n          Net Cf =
   %1.1f x 10^-3\n(b)\n          cd = %1.2f x 10^-3\n"
   ,net_Cf*1e3,c_d*1e3)
```

Chapter 14

Elements of Hypersonic Flow

Scilab code Exa 14.1 Calculation of the pressure coefficients on the top and bottom surface the lift and drag coefficients and the lift to drag ratio using the exact shock expansion theory and the newtonian theory for an infinitely thin flat plate in a hypersonic flow

```
//All the quantities are expressed in SI units

M1 = 8;                          //mach number
alpha = 15*%pi/180;              //anlge of attack
theta= alpha;
gam = 1.4;

//(a)
//for M = 8
v1 = 95.62*%pi/180;
v2 = v1 + theta;

//from Appendix C
M2 = 14.32;

//from Appendix A, for M1 = 8 and M2 = 14.32
p_ratio = 0.9763e4/0.4808e6;

```

```
//from eq.(11.22)
Cp2 = 2/gam/M1^2*(p_ratio - 1);

//for M1 = 8 and theta = 15
b = 21*%pi/180;
Mn_1 = M1*sin(b);

//for this value of Mn,1, from appendix B
p_ratio2 = 9.443;

//thus
Cp3 = 2/gam/M1^2*(p_ratio2 - 1);

c_n = Cp3 - Cp2;

c_l = c_n*cos(alpha);

c_d = c_n*sin(alpha);

L_by_D = c_l/c_d;

printf("\nRESULTS\n----------\n(a) The exact results from the shock-expansion theory are:\n          Cp2 = %1.4f\n          Cp3 = %1.4f\n          cl = %1.4f\n          cd = %1.4f\n          L/D = %1.2f\n",Cp2,Cp3,c_l,c_d,L_by_D)

//(b)
//from Newtonian theory, by eq.(14.9)
Cp3 = 2*sin(alpha)^2;
Cp2 = 0;
c_l = (Cp3 - Cp2)*cos(alpha);
c_d = (Cp3 - Cp2)*sin(alpha);
L_by_D = c_l/c_d;

printf("\n(b) The results from Newtonian theory are:\n          Cp2 = %1.4f\n          Cp3 = %1.4f\n          cl = %1.4f\n          cd = %1.4f\n          L/
```

```
D = %1.2f\n",Cp2,Cp3,c_l,c_d,L_by_D)
```

Chapter 16

Some Special Cases Couette and Poiseuille Flows

Scilab code Exa 16.1 Calculation of the velocity in the middle of the flow the shear stress the maximum temperature in the flow the heat transfer to either wall and the temperature of the lower wall if it is suddenly made adiabatic

```
//All the quantities are expressed in SI units

mue = 1.7894e-5;                    //coefficient of
    viscosity
ue = 60.96;                         //velocity of
   upper plate
D = 2.54e-4;                        //distance
   between the 2 plates
T_w = 288.3;                        //temperature of
    the plates
Pr = 0.71;                          //Prandlt number
cp = 1004.5;                        //specific heat
   at constant pressure

//(a)
//from eq.(16.6)
```

```
u = ue/2;

//(b)
//from eq.(16.9)
tow_w = mue*ue/D;

//(c)
//from eq.(16.34)
T = T_w + Pr*ue^2/8/cp;

//(d)
//from eq.(16.35)
q_w_dot = mue/2*ue^2/D;

//(e)
//from eq.(16.40)
T_aw = T_w + Pr/cp*ue^2/2;

printf("\nRESULTS\n----------\n(a)\n        u = %2.2f m/s\n(b)\n        tow_w = %1.1f N/m2\n(c)\n        T = %3.1f K\n(d)\n        q_w_dot = %3.1f Nm-1s-1\n(e)\n        Taw = %3.1f K",u,tow_w,T,q_w_dot,T_aw)
```

Scilab code Exa 16.2 Calculation of the heat transfer to either plate for the given geometry

```
//All the quantities are expressed in SI units

mue = 1.7894e-5;                    //coefficient of viscosity
Me = 3;                             //mach number of upper plate
D = 2.54e-4;                        //distance between the 2 plates
```

```
pe = 101000;                               //ambient pressure
Te = 288;                                  //temperature of the plates
Tw = Te;
gam = 1.4;                                 //ratio of specific heats
R = 287;                                   //specific gas constant
Pr = 0.71;                                 //Prandlt number
cp = 1004.5;                               //specific heat at constant pressure
tow_w = 72;                                //shear stress on the lower wall

//the velocity of the upper plate is given by
ue = Me*sqrt(gam*R*Te);

//the density at both plates is
rho_e = pe/R/Te;

//the coefficient of skin friction is given by
cf = 2*tow_w/rho_e/ue^2;

//from eq.(16.92)
C_H = cf/2/Pr;

//from eq.(16.82)
h_aw = cp*Te + Pr*ue^2/2;

h_w = cp*Tw;

q_w_dot = rho_e*ue*(h_aw-h_w)*C_H;

printf("\nRESULTS\n---------\nThe heat transfer is given by:\n         q_w_dot = %1.2f x 10^4 W/m2\n",q_w_dot/1e4)
```

Chapter 18

Laminar Boundary Layers

Scilab code Exa 18.1 Calculation of the friction drag on a flat plate for the given velocities

```
//All the quantities are expressed in SI units

p_inf = 101000;                          //freestream
   pressure
T_inf = 288;                             //freestream
   temperature
c = 2;                                   //chord length of
    the plate
S = 40;                                  //planform area
   of the plate
mue_inf = 1.7894e-5;                     //coefficient of
   viscosity at sea level
gam = 1.4;                               //ratio of
   specific heats
R = 287;                                 //specific gas
   constant

//the freestream density is
rho_inf = p_inf/R/T_inf;

```

```
//the speed of sound is
a_inf = sqrt(gam*R*T_inf);

//(a)
V_inf = 100;

//thus the mach number can be calculated as
M_inf = V_inf/a_inf;

//the Reynolds number at the trailing is given as
Re_c = rho_inf*V_inf*c/mue_inf;

//from eq.(18.22)
Cf = 1.328/sqrt(Re_c);

//the friction drag on one surface of the plate is
    given by
D_f = 1/2*rho_inf*V_inf^2*S*Cf;

//the total drag generated due to both surfaces is
D = 2*D_f;

printf("\nRESULTS\n---------\nThe total frictional
    drag is:\n(a)\n         D = %3.1f N\n",D)

//(b)
V_inf = 1000;

//thus the mach number can be calculated as
M_inf = V_inf/a_inf;

//the Reynolds number at the trailing is given as
Re_c = rho_inf*V_inf*c/mue_inf;

//from eq.(18.22)
Cf = 1.2/sqrt(Re_c);

//the friction drag on one surface of the plate is
```

```
given by
D_f = 1/2*rho_inf*V_inf^2*S*Cf;

//the total drag generated due to both surfaces is
D = 2*D_f;

printf("\n(b)\n          D = %4.0f N\n",D)
```

Scilab code Exa 18.2 Calculation of the friction drag on a flat plate using the reference temperature method

```
//All the quantities are expressed in SI units

Pr = 0.71;                          //Prandlt number of air at standard conditions
Pr_star = Pr;
Te = 288;                           //temperature of the upper plate
ue = 1000;                          //velocity of the upper plate
Me = 2.94;                          //Mach number of flow on the upper plate
p_star = 101000;
R = 287;                            //specific gas constant
T0 = 288;                           //reference temperature at sea level
mue0 = 1.7894e-5;                   //reference viscosity at sea level
c = 2;                              //chord length of the plate
S = 40;                             //plate planform area

//recovery factor for a boundary layer is given by eq.(18.47) as
```

```
r = sqrt(Pr);

//rearranging eq.(16.49), we get for M = 2.94
T_aw = Te*(1+r*(2.74-1));

//from eq.(18.53)
T_star = Te*(1 + 0.032*Me^2 + 0.58*(T_aw/Te-1));

//from the equation of state
rho_star = p_star/R/T_star;

//from eq.(15.3)
mue_star = mue0*(T_star/T0)^1.5*(T0+110)/(T_star+110);

//thus
Re_c_star = rho_star*ue*c/mue_star;

//from eq.(18.22)
Cf_star = 1.328/sqrt(Re_c_star);

//hence, the frictional drag on one surface of the plate is
D_f = 1/2*rho_star*ue^2*S*Cf_star;

//thus, the total frictional drag is given by
D = 2*D_f;

printf("\nRESULTS\n----------\nThe total frictional drag is:\n        D = %4.0f N\n",D)
```

Scilab code Exa 18.3 Calculation of the friction drag on a flat plate using the Meador Smart equation for the reference temperature

```
//All the quantities are expressed in SI units
```

```

Pr = 0.71;                    //Prandlt number of air at standard conditions
Pr_star = Pr;
Te = 288;                     //temperature of the upper plate
ue = 1000;                    //velocity of the upper plate
Me = 2.94;                    //Mach number of flow on the upper plate
p_star = 101000;
R = 287;                      //specific gas constant
gam = 1.4;                    //ratio of specific heats
T0 = 288;                     //reference temperature at sea level
mue0 = 1.7894e-5;             //reference viscosity at sea level
c = 2;                        //chord length of the plate
S = 40;                       //plate planform area

//recovery factor for a boundary layer is given by eq.(18.47) as
r = sqrt(Pr);

//from ex.(8.2)
T_aw = Te*2.467;
T_w = T_aw;

//from the Meador-Smart equation
T_star = Te*(0.45 + 0.55*T_w/Te + 0.16*r*(gam-1)/2*Me^2);

//from the equation of state
rho_star = p_star/R/T_star;

```

```
//from eq.(15.3)
mue_star = mue0*(T_star/T0)^1.5*(T0+110)/(T_star+110);

//thus
Re_c_star = rho_star*ue*c/mue_star;

//from eq.(18.22)
Cf_star = 1.328/sqrt(Re_c_star);

//hence, the frictional drag on one surface of the plate is
D_f = 1/2*rho_star*ue^2*S*Cf_star;

//thus, the total frictional drag is given by
D = 2*D_f;

printf("\nRESULTS\n----------\nThe total frictional drag is:\n          D = %4.0f N\n",D)
```

Chapter 19

Turbulent Boundary Layers

Scilab code Exa 19.1 Calculation of the friction drag on a flat plate assuming turbulent boundary layer for the given velocities

```
//All the quantities are expressed in SI units

//(a)
Re_c = 1.36e7;                    //as obtained from ex. 18.1a
rho_inf = 1.22;                   //freestream air denstiy
S = 40;                           //plate planform area

//hence, from eq.(19.2)
Cf = 0.074/Re_c^0.2;

V_inf = 100;

//hence, for one side of the plate
D_f = 1/2*rho_inf*V_inf^2*S*Cf;

//the total drag on both the surfaces is
D = 2*D_f;
```

```

printf("\nRESULTS\n----------\nThe total frictional drag is:\n(a)\n          D = %4.0f N\n",D)

//(b)
Re_c = 1.36e8;                    //as obtained from ex. 18.1b

//hence, from fig 19.1 we have
Cf = 1.34e-3;

V_inf = 1000;

//hence, for one side of the plate
D_f = 1/2*rho_inf*V_inf^2*S*Cf;

//the total drag on both the surfaces is
D = 2*D_f;

printf("\n(b)\n          D = %5.0f N\n",D)
```

Scilab code Exa 19.2 Calculation of the friction drag on a flat plate assuming turbulent boundary layer using reference temperature method

```
//All the quantities are expressed in SI units

//from ex 18.2
Re_c_star = 3.754e7;                 //Reynolds number at the trailing edge of the plate
rho_star = 0.574;
ue = 1000;                           //velocity of the upper plate
S = 40;                              //plate planform area

```

```
//from eq.(19.3) we have
Cf_star = 0.074/Re_c_star^0.2;

//hence, for one side of the plate
D_f = 1/2*rho_star*ue^2*S*Cf_star;

//the total drag on both the surfaces is
D = 2*D_f;

printf("\nRESULTS\n----------\nThe total frictional drag is:\n          D = %5.0f N\n",D)
```

Scilab code Exa 19.3 Calculation of the friction drag on a flat plate for a turbulent boundary layer using the Meador Smart reference temperature method

```
//All the quantities are expressed in SI units

Me = 2.94;                          //mach number of the flow over the upper plate
ue = 1000;
Te = 288;                           //temperature of the upper plate
ue = 1000;                          //velocity of the upper plate
S = 40;                             //plate planform area
Pr = 0.71;                          //Prandlt number of air at standard condition
gam = 1.4;                          //ratio of specific heats

//the recovery factor is given as
r = Pr^(1/3);

```

```
//for M = 2.94
T_aw = Te*(1+r*(2.74-1));
T_w = T_aw;                              //since the flat plate has an adiabatic wall

//from the Meador-Smart equation
T_star = Te*(0.5*(1+T_w/Te) + 0.16*r*(gam-1)/2*Me^2);

//from the equation of state
rho_star = p_star/R/T_star;

//from eq.(15.3)
mue_star = mue0*(T_star/T0)^1.5*(T0+110)/(T_star+110);

//thus
Re_c_star = rho_star*ue*c/mue_star;

//from eq.(18.22)
Cf_star = 0.02667/Re_c_star^0.139;

//hence, the frictional drag on one surface of the plate is
D_f = 1/2*rho_star*ue^2*S*Cf_star;

//thus, the total frictional drag is given by
D = 2*D_f;

printf("\nRESULTS\n----------\nThe total frictional drag is:\n         D = %5.0f N\n",D)
```