Scilab Textbook Companion for Introduction To Fluid Mechanics by R. W. Fox And A. T. McDonald[1]

Created by
Eswar Prasad
B.Tech
Electrical Engineering
National Institute of Technology Trichy
College Teacher
Prof Shiv Raj Deshmukh
Cross-Checked by
Prof Shiv Raj Deshmukh

December 6, 2013

[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: Introduction To Fluid Mechanics

Author: R. W. Fox And A. T. McDonald

Publisher: Wiley India, New Delhi

Edition: 5

Year: 2001

ISBN: 9971-51-355-2

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

List of Figures

Chapter 1

Introduction

Scilab code Exa 1.01 Head addition

```
//Heat addition//
filename=pathname+filesep()+'1.01-data.sci'
exec(filename)
//Heat added during the process(in kJ):
Q12=m*cp*(T2-T1)
printf("\n\nRESULTS\n\n")
printf("\n\nHeat added during the process: %f kJ\n\n",Q12/1000)
```

Scilab code Exa 1.02 speed and actual speed

```
//speed and actual speed//
pathname=get_absolute_file_path('1.02.sce')
filename=pathname+filesep()+'1.02-data.sci'
exec(filename)
//Speed at which the ball hits the ground(in m/sec):
V=sqrt(m*g/k*(1-%e^(2*k/m*(-y0))))
//Terminal speed(in m/sec):
```

```
Vt=sqrt(m*g/k)
//Ratio of actual speed to the terminal speed:
r=V/Vt;
printf("\n\nRESULTS\n\n")
printf("\n\nSpeed at which the ball hits he ground:
    %f m/sec\n\n",V)
printf("\n\nRatio of actual speed to the terminal
    speed: %f\n\n",r)
```

Chapter 2

Fundamental Concepts

Scilab code Exa 2.02 Viscosity and stress

```
pathname=get_absolute_file_path('2.02.sce')
filename=pathname+filesep()+'2.02-data.sci'
exec(filename)

//Viscosity in units of lbf-s/ft^2:
u1=u/100/454/32.2*30.5
//Kinematic viscosity (in m/sec^2):
v=u1/SG/d*(0.305)^2
//Shear stress on the upper plate(lbf/ft^2):
tu=u1*U/D*1000
//Shear stress on the lower plate(in Pa)
tl=tu*4.45/0.305^2
printf("\n\nRESULTS\n\n")
printf("\n\nViscosity in units of lbf-s/ft^2: %1.8f
   $lbf-s/ft^2\n\n",u1)
printf("\n\nKinematic viscosity: %1.8f m/sec^2\n\n",
   v)
printf("\n\nShear stres on the upeer plate: %f lbf/
   ft^2\n\n",tu)
printf("\n\nSear stress on the lower plate: %f Pa\n\
   n",tl)
```

Chapter 3

Fluid Statics

Scilab code Exa 3.01 liquid level

```
//liquid level//
pathname=get_absolute_file_path('3.01.sce')
filename=pathname+filesep()+'3.01-data.sci'
exec(filename)
//Tube diameter(in mm):
D=1:25;
D1=D/1000
[m n]=size(D1)
for i=1:n
//Change in liquid level for water(in mm):
dhw(i)=4*STw*cosd(thetaw)/dw/g/D1(i);
//Change in liquid level for mercury(in mm):
dhm(i)=4*STm*cosd(thetam)/dm/g/D1(i);
end;

//Plotting tube daimeter and water level:
plot(D1*1000,dhw,'-o')
//Plotting tube daimeter and mercury level:
plot(D1*1000,dhm,'-*')
```

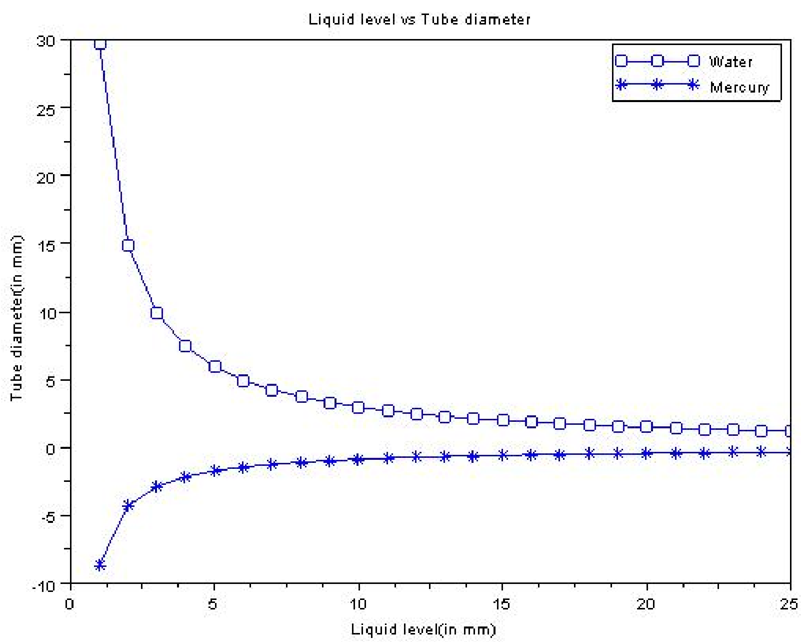

Figure 3.1: liquid level

```
legend(['Water';'Mercury']);
xtitle('Liquid level vs Tube diameter','Liquid level (in mm)','Tube diameter(in mm)')
```

Scilab code Exa 3.03 pressure difference

```
//pressure difference//
pathname=get_absolute_file_path('3.03.sce')
filename=pathname+filesep()+'3.03-data.sci'
exec(filename)
//Pressure difference(in lbf/in^2):
dp=g*d*(-d1+SGm*d2-SGo*d3+SGm*d4+d5)/12/144
printf("\n\nRESULTS\n\n")
printf("\n\nPressure difference between A and B: %f lbf/in^2\n\n",dp)
```

Scilab code Exa 3.04 temperature and pressure

```
//temperature and pressure//
pathname=get_absolute_file_path('3.04.sce')
filename=pathname+filesep()+'3.04-data.sci'
exec(filename)
//Assuming temperature varies linearly with altitude:
//Temperature gradient(in F/ft):
m=(T1-T2)/(z2-z1)
//Value of g/(m*R):
v=g/m/R/32.2
//Pressure at Vail Pass(in inches of Hg):
p12=p1*((T2+460)/(T1+460))^v
//Percentage change in density:
pc1=(p12/p1*(T1+460)/(T2+460)-1)*100
//Assuming density is constant:
```

```
//Pressure at Vail Pass(in inches of Hg):
p22=p1*(1-(g*(z2-z1)/(R*32.2)/(T1+460)))
//Percentage change in density:
pc2=0;
//Assuming temperature is constant:
//Pressure at Vail Pass(in inches of Hg):
p32=p1*%e^(-g*(z2-z1)/(R*32.2)/(T2+460))
//Percentage change in density:
pc3=(p32/p1*(T1+460)/(T1+460)-1)*100
//For an adiabatic atmosphere:
p42=p1*((62+460)/(80+460))^(k/(k-1))
//Percentage change in density:
pc4=(p42/p1*(T1+460)/(T2+460)-1)*100
printf("\n\nRESULTS\n\n")
printf("\n\n1) If temperature varies linearly with
   altitude\n\n")
printf("\n\n\tAtmospheric pressure at Vail Pass: %f
   inches of Hg\n\n",p12)
printf("\n\n\tPercentage change in density wrt
   Denver: %f percent\n\n",pc1)
printf("\n\n2) If density is constant\n\n")
printf("\n\n\tAtmospheric pressure at Vail Pass: %f
   inches of Hg\n\n",p22)
printf("\n\n\tPercentage change in density wrt
   Denver: %f percent\n\n",pc2)
printf("\n\n3) If temperature is constant\n\n")
printf("\n\n\tAtmospheric pressure at Vail Pass: %f
   inches of Hg\n\n",p32)
printf("\n\n\tPercentage change in density wrt
   Denver: %f percent\n\n",pc3)
printf("\n\n4) For an adiabatic atmosphere\n\n")
printf("\n\n\tAtmospheric pressure at Vail Pass: %f
   inches of Hg\n\n",p42)
printf("\n\n\tPercentage change in density wrt
   Denver: %f percent\n\n",pc4)
```

Scilab code Exa 3.05 force and pressure

```
//force and pressure//
pathname=get_absolute_file_path('3.05.sce')
filename=pathname+filesep()+'3.05-data.sci'
exec(filename)
//Net force on the gate(in kN):
Fr=d*g*w*(D*L+L^2/2*sind(theta))
//Centre of pressure:
//Calculation for y coordinate:
   yc=D/sind(theta)+L/2
   //Area(in m^2):
   A=L*w
   //Moment of inertia of rectangular gate(in m^4):
   Ixx=w*L^3/12
   //y coordinate(in m):
   y=yc+Ixx/A/yc
//Calculation for x coordinate:
   Ixy=0
   xc=w/2
   //x coordinate(in m):
   x=xc+Ixy/A/xc
printf("\n\nRESULTS\n\n")
printf("\n\nNet force on the gate: %f kN\n\n",Fr
   /1000)
printf("\n\nCoordinate of centre of pressure:(%0.1f,
   %0.1f)\n\n",x,y)
```

Scilab code Exa 3.06 force

```
//force//
pathname=get_absolute_file_path('3.06.sce')
```

```
filename=pathname+filesep()+'3.06-data.sci'
exec(filename)
//Force required to keep the door shut(in lbf):
function y=f(z),y=b/L*p0*z+d*b/L*(L*z-z^2),
    endfunction
Ft=intg(0,L,f)
printf("\n\nRESULTS\n\n")
printf("\n\nForce requiredto kep the door shut: %.1f
     lbf\n\n",Ft)
```

Scilab code Exa 3.07 force at equilibrium

```
//force at equilibrium//
pathname=get_absolute_file_path('3.07.sce')
filename=pathname+filesep()+'3.07-data.sci'
exec(filename)
//Horizontal component of resultant force(in kN):
Frh=0.5*d*g*w*D^2
//Line of action of Frh(in m):
y1=0.5*D+w*D^3/12/(0.5*D)/(w*D)
//Vertical component of resultant force(in kN):
function y=q(x), y=d*g*w*(D-sqrt(a*x)),endfunction
Frv=intg(0,D^2/a,q)
//Line of acion of Frv(in m):
function k=f(x), k=d*g*w/Frv*x*(D-sqrt(a*x)),
    endfunction
xa=intg(0,D^2/a,f)
//Force required to keep the gate in equilibrium(in
    kN):
Fa=1/l*(xa*Frv+(D-y1)*Frh)
printf("\n\nRESULTS\n\n")
printf("\n\nForce required to keep the gate at
    equilibrium: %f kN\n\n",Fa/1000)
```

Chapter 4

Basic Equations in Integral form for a Control Volume

Scilab code Exa 4.01 Velocity

```
//Velocity//
pathname=get_absolute_file_path('4.01.sce')
filename=pathname+filesep()+'4.01-data.sci'
exec(filename)
//If I=integral of(pV.dA):
//For system: Ics=IA1+IA2+IA3+IA4.
//For area 1
IA1=-d*V1*A1
//For area 3: IA2=d*V3*A3=m3
IA3=m3
//For area 4: IA4=-d*V4*A4=-d*Q4
IA4=-d*Q4
//For area 2:
IA2=-IA1-IA3-IA4
//Velocity at section 2(in ft/sec):
V2=IA2/d/A2
//V2 is in the negative y direction
printf("\n\nRESULTS\n\n")
printf("\n\nVelocity at section 2: -%.0fj ft/sec\n\n
```

```
   ",V2)
```

Scilab code Exa 4.02 Mass flow

```
//Mass flow//
pathname=get_absolute_file_path('4.02.sce')
filename=pathname+filesep()+'4.02-data.sci'
exec(filename)
//If I=integral of(pV.dA):
//For system: ICS=Iab+Ibc+Icd+Ida
//But ICS=0

//For Aab:
function p=f(y),p=-d*U*w*y^0,endfunction
IAab=intg(0,t,f)

//For Acd:
function q=g(y),q=d*U*w*(2*y/t-(y/t)^2),endfunction
IAcd=intg(0,t,g)

//Mass flow rate across surface bc(in kg/sec):
mbc=(-IAab-IAcd)/1000
printf("\n\nRESULTS\n\n")
printf("\n\nMass flow rate across surface bc: %.4f
   kg/sec\n\n",mbc)
```

Scilab code Exa 4.03 density

```
//density//
pathname=get_absolute_file_path('4.03.sce')
filename=pathname+filesep()+'4.03-data.sci'
exec(filename)
```

```
//Rate of change of air density in tank(in (kg/m^3)/s):
r=-d*v*A/V/10^6
printf("\n\nRESULTS\n\n")
printf("\n\nRate of change of air density in tank: %.3f kg/m^3\n\n",r)
printf("\n\nThe density decreases as is indicated by the negative sign\n\n")
```

Scilab code Exa 4.04 Horizontal force

```
//Horizontal force//
pathname=get_absolute_file_path('4.04.sce')
filename=pathname+filesep()+'4.04-data.sci'
exec(filename)
//1) Control Volume selected so that area of left surface is equal to the area of the right surface
u1=15;
//Force of support on control volum(in kN):
function y=f(A),y=-u1*d*V,endfunction
Rx1=intg(0,0.01,f)
//Horizontal force on support(in kN):
Kx=-Rx1
//2) Control volumes are selected do that the area of the left and right surfaces are equial to the area of the plate

function z=g(A),z=-u1*d*V,endfunction
Fsx=intg(0,0.01,g)
//Net force on plate:Fx=0=-Bx-pa*Ap+Rx
//                    Rx=pa*Ap+Bx
//From the above, it is obtained that:
Rx2=-2.25
//Horizontal force on support(in kN):
Kx2=-Rx2
```

```
printf("\n\nRESULTS\n\n")
printf("\n\nHorizontal force on support: %.3f kN\n\n",Kx/1000)
```

Scilab code Exa 4.05 Scale

```
//Scale//
pathname=get_absolute_file_path('4.05.sce')
filename=pathname+filesep()+'4.05-data.sci'
exec(filename)
//Weight of water in the tank(in lbf):
d1=62.4;
WH2O=d1*A*h
v=-5;
//Total body force in negative y direction(lbf):
function y=f(A),y=-v*d2*V1,endfunction
F=intg(0,A1,f)
//Force of scale on control volume(in kN):
Ry=W+WH2O-F
printf("\n\nRESULTS\n\n")
printf("\n\nScale Reading: %.3f lbf\n\n",Ry)
```

Scilab code Exa 4.06 force exerted per unt

```
//force exerted per unt//
pathname=get_absolute_file_path('4.06.sce')
filename=pathname+filesep()+'4.06-data.sci'
exec(filename)
//X-component of reaction force per unit width of the gate(in N/m):
Rxw=(d*(V2^2*D2-V1^2*D1))-(d*g/2*(D1^2-D2^2))
//Horizontal force exerted per unt width on the gate (in N/m):
```

```
Kxw=-Rxw
printf("\n\nRESULTS\n\n")
printf("\n\nHorizontal force exerted per unt width on the gate: %.3f kN/m\n\n",Kxw/1000)
```

Scilab code Exa 4.07 Force to hold

```
//Force to hold//
pathname=get_absolute_file_path('4.07.sce')
filename=pathname+filesep()+'4.07-data.sci'
exec(filename)
//Velocity at section 1(in m/sec):
V1=V2*A2/A1
//Gauge pressure(in kPa):
p1g=p1-patm
u1=V1;u2=-V2;
//Reaction force component in the x direction(in N):
Rx=-p1g*A1-u1*d*V1*A1
//Reaction force component in the y direction(in N):
Ry=u2*d*V2*A2
printf("\n\nRESULTS\n\n")
printf("\n\nForce to hold elbow acting to the left: %.3f kN\n\n",Rx/1000)
printf("\n\nForce to hold elbow acting downwards: %.3f N\n\n",Ry)
```

Scilab code Exa 4.08 Tension

```
//Tension//
filename=pathname+filesep()+'4.08-data.sci'
exec(filename)
//Tension required to pull the belt(in lbf):
T=Vbelt*m/32.2
```

```
printf("\n\nRESULTS\n\n")
printf("\n\nTension required to pull the belt: %.3f
    lbf\n\n",T)
```

Scilab code Exa 4.09 pressure required

```
//pressure required//
pathname=get_absolute_file_path('4.09.sce')
filename=pathname+filesep()+'4.09-data.sci'
exec(filename)
//Minimum gauge pressure required(in lbf/in^2):
pg=8/%pi^2*d/D1^4*Q^2*((D1/D2)^4-1)*144
printf("\n\nRESULTS\n\n")
printf("Minimum gauge pressure required: %.3f lbf/in
    ^2",pg)
```

Scilab code Exa 4.10 Net force

```
//Net force//
pathname=get_absolute_file_path('4.10.sce')
filename=pathname+filesep()+'4.10-data.sci'
exec(filename)
u1=V-U
u2=(V-U)*cosd(theta)
v2=(V-U)*sind(theta)
V1=V-U
V2=V1
//X component of moment equation(in N):
function y=f(A),y=u1*-(d*V1),endfunction
function z=g(A),z=u2*d*V2,endfunction
Rx=intg(0,A,f)+intg(0,A,g)

//Y component of moment equation(in N):
```

```
function a=h(A),a=v2*d*V1,endfunction
Ry=intg(0,A,h) //This is after neglecting weight of
   vane and the water.
printf("\n\nRESULTS\n\n")
printf("\n\nNet force on the vane: %.3f i+%.2f j kN\
   n\n",Rx/1000,Ry/1000)
```

Scilab code Exa 4.11 PLOTTING

```
//PLOTTING//
pathname=get_absolute_file_path('4.11.sce')
filename=pathname+filesep()+'4.11-data.sci'
exec(filename)
//Evaluating the value of Vb:
Vb=V*(1-cosd(theta))*d*A/M
//Value of U/V for various values of t
t=0:20;
[m n]=size(t)
for i=1:n
  U_V(i)=Vb*t(i)/(1+Vb*t(i));
end

//Plotting U/V vs t:
plot(t,U_V)
xtitle('U/V vs t','t (in sec)','U/V')
```

Scilab code Exa 4.12 Velocity of rocket

```
//Velocity of rocket//
pathname=get_absolute_file_path('4.12.sce')
filename=pathname+filesep()+'4.12-data.sci'
exec(filename)
//Acceleration of rocket at t=0(in m/sec^2):
```

```
Ve*me/M0-g
//Velocity of rocket at t=10 (in m/sec):
function y=f(t),y=Ve*me/(M0-me*t)-g,endfunction
Vcv=intg(0,t,f)
printf("\n\nRESULTS\n\n")
printf("\n\nVelocity of rocket at t=10: %.1f m/sec\n\n",Vcv)
```

Scilab code Exa 4.14 Relative speed and friction

```
//Relative speed and friction//
pathname=get_absolute_file_path('4.14.sce')
filename=pathname+filesep()+'4.14-data.sci'
exec(filename)
//Area of jet(in mm^2):
Ajet=%pi/4*D^2
//Jet speed relative to the nozzle(in m/sec):
Vrel=Q/2/Ajet*10^6/60/1000
//Value of w*R in m/sec:
wR=w*R*2*%pi/60/1000
//Friction torque at pivot(in N-m):
Tf=R*(Vrel*cosd(alpha)-wR)*d*Q/1000/60/1000
printf("\n\nRESULTS\n\n")
printf("\n\nJet speed relative to each nozzle: %.2f m/sec\n\n",Vrel)
printf("\n\nFriction torque at pivot: %.5f N-m\n\n",Tf)
```

Scilab code Exa 4.16 Rate of heat

```
//Rate of heat//
pathname=get_absolute_file_path('4.16.sce')
filename=pathname+filesep()+'4.16-data.sci'
```

```
exec(filename)
//Velocity at exit(in ft/sec):
V2=m*R*(T2+460)/A2/p2/144
//As power input is to CV, Ws=-600
//Rate of heat transfer(in Btu/sec):
Q=Ws*550/778+m*cp*(T2-T1)+m*V2^2/2/32.2/778
printf("\n\nRESULTS\n\n")
printf("\n\nRate of heat transfer: %.3f Btu/sec\n\n"
    ,Q)
```

Scilab code Exa 4.17 Mass flow rate

```
//Mass flow rate//
pathname=get_absolute_file_path('4.17.sce')
filename=pathname+filesep()+'4.17-data.sci'
exec(filename)
//Density of tank(in kg/m^3):
d=(p1+patm)/R/T
//Mass flow rate of air into the tank(in kg/sec):
m=d*V*cv*r/R/T*1000
printf("\n\nRESULTS\n\n")
printf("\n\nMass flow rate of air into the tank: %.3
    f g/sec\n\n",m)
```

Chapter 5

Introducton to Differential Analysis of Fluid Motion

Scilab code Exa 5.02 Rate of change

```
//Rate of change//
pathname=get_absolute_file_path('5.02.sce')
filename=pathname+filesep()+'5.02-data.sci'
exec(filename)
//Rate of change of density with time(in kg/m^3-s):
r=-d*V/L
printf("\n\nRESULTS\n\n")
printf("\n\nRate of change of density with time: %.1
    f kg/m^3-s\n\n",r)
```

Scilab code Exa 5.07 angular and rotation

```
//angular and rotation//
pathname=get_absolute_file_path('5.07.sce')
filename=pathname+filesep()+'5.07-data.sci'
exec(filename)
```

```
//At point b, u=3 mm/sec
u=3;
//Displacemet of b(in mm):
xb=u*t
//Rate of angular deformation(in s^-1):
def=U/h
//Rate of rotation(in s^-1):
rot=-0.5*U/h
printf("\n\nRSULTS\n\n")
printf("\n\nRate of angular deformation: %.1f /sec\n
   \n",def)
printf("\n\nRate of rotation: %.1f /sec\n\n",rot)
```

Scilab code Exa 5.08 Rates and area

```
//Rates and area//
pathname=get_absolute_file_path('5.08.sce')
filename=pathname+filesep()+'5.08-data.sci'
exec(filename)
//Value of T:
T=log(3/2)/A
x0=1:2;
y0=1:2;
for i=1:2
  for j=1:2
  //For X coordinate:
  X(i)(j)=x0(i)*%e^(A*T)
  //For Y coordinate:
  Y(i)(j)=y0(j)*%e^(-A*T)
  end
end
plot(X,Y)
//Rates of linear deformation in X direction:
Ax=0.3;
//Rate of linear deformation in the y direction:
```

```
Ay=-0.3;
//Rate of volume dilation(s^-1):
v=A-A
//Area of abcd:
A1=1;
//Area of a'b'c'd':
A2=(3-3/2)*(4/3-2/3)
printf("\n\nRESULTS\n\n")
printf("\n\nRates of linear deformation in X and Y direction: %.1f /s , %.1f /s\n\n",Ax,Ay)
printf("\n\nRate of volume dilation: %.0f /sec\n\n",v)
printf("\n\nArea of abcd and a,b,c,d:%.1f m^2, %.1f m^\n\n",A1,A2)
```

Scilab code Exa 5.09 Volume flow rate

```
//Volume flow rate//
pathname=get_absolute_file_path('5.09.sce')
filename=pathname+filesep()+'5.09-data.sci'
exec(filename)
//Volume flow rate(in m^3/sec):
Q=d*g*sind(theta)*b*(h/1000)^3*1000/u/3
printf("RESULTS")
printf("\n\nVolume flow rate: %.4f m^3/sec\n\n",Q)
```

Chapter 6

Incompressible Inviscid Flow

Scilab code Exa 6.01 Volume flow rate

```
//Volume flow rate//
pathname=get_absolute_file_path('06.01.sce')
filename=pathname+filesep()+'06.01-data.sci'
exec(filename)
//Velocity of flow(in m/sec):
V=sqrt(dw/log((r+w)/r)*g/da*p/1000)
//Volume flow rate(in m^3/sec):
Q=V*(d*w)
printf("\n\nRESULTS\n\n")
printf("\n\nVolume flow rate: %.3f m^3/sec\n\n",Q)
```

Scilab code Exa 6.02 Velocity of flow

```
//Velocity of flow//
pathname=get_absolute_file_path('06.02.sce')
filename=pathname+filesep()+'06.02-data.sci'
exec(filename)
//Velocity of flow(in m/sec):
```

```
V=sqrt(2*dw*g*p/1000*SG/da)
printf("\n\nRESULTS\n\n")
printf("\n\nVelocity of flow: %.3f m/sec\n\n",V)
```

Scilab code Exa 6.03 prssure required

```
//prssure required//
filename=pathname+filesep()+'06.03-data.sci'
exec(filename)
//Velocity of flwat the inlet(in m/sec):
V1=Ae/Ai*V2
//Gauge pressure required at the inlet(in kPa):
p=0.5*da*(V2^2-V1^2)
printf("\n\nRESULTS\n\n")
printf("\n\nGauge prssure required at the inlet: %.3
    f kPa\n\n",p/1000)
```

Scilab code Exa 6.04 Speed and pressure

```
//Speed and pressure//
pathname=get_absolute_file_path('06.04.sce')
filename=pathname+filesep()+'06.04-data.sci'
exec(filename)
//Speed of water at exit(in m/sec):
V2=sqrt(2*g*z)
//Pressure at point A in the flow(kPa):
pA=p1+d*g*(0-l)-0.5*d*V2^2
printf("\n\nRESULTS\n\n")
printf("\n\nSpeed of water at exit: %.3f m/sec\n\n",
    V2)
printf("\n\nPressure at point A in the flow: %3f kPa
    \n\n",pA/1000)
```

Scilab code Exa 6.05 flow

```
//flow//
pathname=get_absolute_file_path('06.05.sce')
filename=pathname+filesep()+'06.05-data.sci'
exec(filename)
//Velocity of flow at the exit(in ft/sec):
V2=sqrt(2*g*(Du-Dd/12))
//Volume flow rate/width(ft^2/sec):
Q=V2*Dd/12
printf("\n\nRESULTS\n\n")
printf("\n\nVelocity of flow at the exit: %.3f ft/
    sec\n\n",V2)
printf("\n\nVolume flow rate/width: %.3f ft^2/sec\n\
    n",Q)
```

Scilab code Exa 6.06 pressure

```
//pressure//
pathname=get_absolute_file_path('06.06.sce')
filename=pathname+filesep()+'06.06-data.sci'
exec(filename)
//Pressure of air at 1000 m(in N/m^2):
p=P1*pa
//Density of air at 1000m(in kg/m^3):
d=D1*da
//Stagnation pressure at A(in kPa):
p0A=p+0.5*d*(V*1000/3600)^2
//Static pressure at B(in kPa):
pB=p+d/2*((V*1000/3600)^2-Vb^2)
printf("\n\nRESULTS\n\n")
```

```
printf("\n\nStagnation pressure at A: %.3f kPa\n\n",
    p0A/1000)
printf("\n\nStatic pressure at B: %.3f kPa\n\n",pB
    /1000)
```

Scilab code Exa 6.08 temperature

```
//temperature//
pathname=get_absolute_file_path('06.08.sce')
filename=pathname+filesep()+'06.08-data.sci'
exec(filename)
//Velocity of flow at exit(in ft/sec):
V4=sqrt(2*g*(z3-0))
//Mass flow rate of water(in slug/sec):
m=d*V4*A4/144
//Rise in temperature between points 1 and 2(in R):
T=Q*3413/3600/m/32.2
printf("\n\nRESULTS\n\n")
printf("\n\nRise in temperature between points 1 and
     2: %.3f R\n\n",T)
```

Scilab code Exa 6.09 Streamline flow

```
//Streamline flow//
pathname=get_absolute_file_path('06.09.sce')
filename=pathname+filesep()+'06.09-data.sci'
exec(filename)
t=0:5
//Value of sqrt(2gh):
x=sqrt(2*g*h)
//Value of 1/2L*sqrt(2gh):
y=1/2/L*x
[m n]=size(t)
```

```
i=1:n;
//Velocity(in m/sec):
V2=x*tanh(y*t(i))
plot(t,V2);
xtitle('Streamline flow from 1 to 2','Time(in s)','
   V2(in m/sec)')
```

Chapter 7

Dimensional Analysis and Simlitude

Scilab code Exa 7.04 speed and force

```
//speed and force//
pathname=get_absolute_file_path('7.04.sce')
filename=pathname+filesep()+'7.04-data.sci'
exec(filename)
//Velocity of prototype in ft/sec
Vp1=Vp*6080/3600
//Reynolds number of prototype:
Rep=Vp1*Dp/vp
//Rep=Rem
//Therefore:
Rem=Rep;
//Velocity of air for wind tunnel(in ft/sec):
Vm=Rem*vm/(Dm/12)
//Drag force on prototype(in lbf):
Fp=Fm*(dp/dm)*(Vp1/Vm)^2*(Dp/(Dm/12))^2
printf("\n\nRESULTS\n\n")
printf("\n\nTest speed in air: %.3f ft/sec\n\n",Vm)
printf("\n\nDrag force on prototype: %.3f lbf\n\n",
    Fp)
```

Scilab code Exa 7.05 speed force and power

```
//speed force and power//
pathname=get_absolute_file_path('7.05.sce')
filename=pathname+filesep()+'7.05-data.sci'
exec(filename)
//Width of the model(in m):
wm=S*wp*0.3048
//Area of model (in m^2):
Am=S^2*Ap*0.305^2
[m n]=size(V)
i=1:n
//Aerodynamic drag coefficient():
Cd=2.*Fd(i)/d./(V(i))^2/0.0305
//Reynolds number:
Re=V(i)*wm/v
plot(Re,Cd);
a=gca()
a.data_bounds=[100000,0.4;500000,0.6]
xtitle('Aerodynamic drag coefficient vs drag force','Reynolds number','Model Drag Coeff.')
//It is seen that drag coefficient becomes constant at CD=0.46above Re=4*10^5 at which speed of air is 40m/s
CDc=0.46;
Va=40;
//Drag force (in N):
FDp=CDc/2*d*(Vp*5/18)^2*Ap*0.305^2
//Power required to pull prototype at 100 kmph(in W)
Pp=FDp*Vp*5/18
printf("\n\nRESULTS\n\n")
printf("\n\nSpeed above which Cd is constant: %.3f m/sec\n\n",Va)
printf("\n\nDrag Force: %.3f kN\n\n",FDp/1000)
```

```
printf("\n\nPower required to pull prototype at 100
   kmph: %.3f kW\n\n",Pp/1000)
```

Scilab code Exa 7.06 power and speed

```
//power and speed//
pathname=get_absolute_file_path('7.06.sce')
filename=pathname+filesep()+'7.06-data.sci'
exec(filename)
//The same pump is used for both the conditions.
   Hence:
D2=D1;
//The same water is used for both the conditions.
   Hence:
d2=d1;
//Flow rate at condition 2(in gpm):
Q2=Q1*N2/N1*(D2/D1)^3
//Head at condition 1(in ft):
H1=(N1*sqrt(Q1)/Nscu1)^(4/3)
//Head at condition 1(in ft):
H2=H1*(N2/N1)^2*(D2/D1)^2
//Pump output power at condition 1(in hp):
P1=d1*g*Q1*H1/7.48/60/550
//Pump output power at condition 2(in hp):
P2=P1*(d2/d1)*(N2/N1)^3*(D2/D1)^5
//Required input power(in hp):
Pin=P2/Effp
//Specific speed at condition 2:
Nscu2=N2*sqrt(Q2)/H2^(3/4)
printf("\n\nRESULTS\n\n\n")
printf("\n\nVolume flow rate at condition 2: %.3f
   gpm\n\n\n",Q2)
printf("\n\nHead at condition: %.3f ft\n\n\n",H2)
printf("\n\nPump output power at condition: %.3f hp\
   n\n\n",P2)
```

```
printf("\n\nRequired input power: %.3f hp\n\n\n",Pin
   )
printf("\n\nSpecific speed at condition 2: %.3f\n\n\
   n",Nscu2)
```

Chapter 8

Internal Incompressible Viscous Flow

Scilab code Exa 8.01 Leakage flow rate

```
//Leakage flow rate//
pathname=get_absolute_file_path('8.01.sce')
filename=pathname+filesep()+'8.01-data.sci'
exec(filename)
//Leakage flow rate (in mm^3/sec):
Q=%pi/12*D*a^3*(p1-p2)*10^3/u/L
//Velocity of flow(in m/sec):
V=Q/%pi/D/a/1000
//Specific gravity of SAE 10W oil:
SG=0.92;
//Reynolds Number:
Re=SG*dw*V*a/u/1000
//As Re<1400, flow is laminar.
printf("\n\nRESULTS\n\n")
printf("\n\nLeakage flow rate: %.3f mm^3/sec\n\n",Q)
```

Scilab code Exa 8.02 Torque and power

```
//Torque and power//
pathname=get_absolute_file_path('8.02.sce')
filename=pathname+filesep()+'8.02-data.sci'
exec(filename)
//Shear stres (in lbf/ft^2):
Tyx=u*N*2*%pi/60*D/2/(a/2)
//Torqe(in inches-lbf):
T=%pi/2*Tyx*D^2*L/144
//Power dissipated in the bearing(in hp):
P=T*N/60*2*%pi/12/550
//Reynolds number:
Re=SG*p*N*2*%pi/60*1.5*a/2/u/144
printf("\n\nRESULTS\n\n")
printf("\n\nTorque: %.3f inches-lbf\n\n",T)
printf("\n\nPower dissipated in the bearing: %.3f hp
    \n\n",P)
```

Scilab code Exa 8.04 Viscosity of fluid

```
//Viscosity of fluid//
pathname=get_absolute_file_path('8.04.sce')
filename=pathname+filesep()+'8.04-data.sci'
exec(filename)
//Viscosity of the liquid(in N-s/m^2):
u=%pi/128*p*1000*D^4/Q/L/1000
//Velocity(in m/sec)
V=Q/(%pi/4*D^2)/1000
//Reynolds number:
Re=d*V*D/u/1000
printf("\n\nRESULTS\n\n")
printf("\n\nViscosity of fluid %.3f N-s/m^2\n\n",u)
```

Scilab code Exa 8.05 required

```
//required//
pathname=get_absolute_file_path('8.05.sce')
filename=pathname+filesep()+'8.05-data.sci'
exec(filename)
//Reservoir depth required to maintain flow(in m):
D1=8*Q^2/(%pi)^2/D^4/g*(f*L/D+K+1)
//Reynolds number:
Re=4*d*Q/((%pi)*u*D)
printf("\n\nRESULTS\n\n")
printf("\n\nReservoir depth required to maintain
    flow: %.3f m\n\n",D1)
```

Scilab code Exa 8.06 Maximum and power

```
//Maximum and power//
pathname=get_absolute_file_path('8.06.sce')
filename=pathname+filesep()+'8.06-data.sci'
exec(filename)
//Velocity of flow(in ft/sec):
V=Q/24/3600/(%pi/4*(D/12)^2)*42/7.48
//Maximum spacing(in ft):
L=2/f*D/12*(p2-p1)/(SG*d)/V^2*144
//Power needed at each pump(in hp):
Win=1/Effp*V*%pi/4*(D/12)^2*(p2-p1)/550*144
printf("\n\nRESULTS\n\n")
printf("\n\nMaximum spacing: %.3f feet\n\n",L)
printf("\n\nPower needed at each pump: %.3f hp\n\n",
    Win)
```

Scilab code Exa 8.07 Volume low

```
//Volume low//
pathname=get_absolute_file_path('8.07.sce')
filename=pathname+filesep()+'8.07-data.sci'
exec(filename)
//Velocity(in ft/sec):
V2=sqrt(2*g*l/(f*((L+l)/D*12+8)+1))
//Volume flow rate(in gpm):
Q=V2*%pi*(D/12)^2/4*7.48*60
printf("\n\nRESULTS\n\n")
printf("\n\nVolume low rate: %.3f\n\n",Q)
```

Scilab code Exa 8.08 Minimum diameter

```
//Minimum diameter//
pathname=get_absolute_file_path('8.08.sce')
filename=pathname+filesep()+'8.08-data.sci'
exec(filename)
//Value of dPmax(in psi):
dPmax=p1-p2
//Q in cubic feet/sec:
Q1=1500/60/7.48;
//Initially assume diameter to be 4inches:
D=4;
//Reynolds number:
Re=4*Q1/%pi/v/D*12
//For this value,
f=0.012;
 dP=8*f*L*p*Q1^2/(%pi)^2/D^5*1728;
while(dP>dPmax)
   dP=8*f*L*p*Q1^2/(%pi)^2/D^5*1728;
```

```
    if(dP<dPmax)
    break
    else
    D=D+1;
 end
end
printf("\n\nRESULTS\n\n")
printf("Minimum diameter that can be used:%.1f
   inches\n\n",D)
```

Scilab code Exa 8.09 Loss Coefficient

```
//Loss Coefficient//
pathname=get_absolute_file_path('8.09.sce')
filename=pathname+filesep()+'8.09-data.sci'
exec(filename)
//Average velocity (in ft/s):
V2=4/%pi*Q/D^2*144
//Reynolds number:
Re=V2*D/v/12
//For this value,
f=0.013;
//Power law exponent:
n=-1.7+1.8*log10(Re)
//Value of V/U:
v_u=2*n^2/(n+1)/(2*n+1)
//Value of alpha:
alpha=(1/v_u)^3*2*n^2/(3+n)/(3+2*n)
//Loss Coefficient for a square edged entrance:
K=2*g*h/V2^2-f*L/D*12-alpha;
printf("\n\nRESULTS\n\n")
printf("\n\nLoss Coefficient for a square edged
   entrance: %.3f \n\n",K)
```

Scilab code Exa 8.10 Volume and increase

```
//Volume and increase//
pathname=get_absolute_file_path('8.10.sce')
filename=pathname+filesep()+'8.10-data.sci'
exec(filename)
//Velocity V1(in m/s):
V1=sqrt(2*g*z0/1.04)
//Volume flow rate(in m^3/sec):
Q=V1*%pi*D^2/4
Kdiff=1-1/A_R^2-Cp
//For 2nd case:
//Velocity(in m/s):
V1=sqrt(2*g*z0/0.59)
//Volume flow rate(in m^3/s):
Qd=V1*%pi*D^2/4
//Increase in discharge after addition of diffuser
    is:
dQ=(Qd-Q)/Q*100
printf("\n\nRESULTS\n\n")
printf("\n\nVolume flow rate in case1: %.3f m^3/sec\
    n\n",Q)
printf("\n\nVolume flow rate in case 2: %.3f m^3/sec
    \n\n",Qd)
printf("\n\nIncrease in discharge after addition of
    diffuser is: %.3f percent\n\n",dQ)
```

Scilab code Exa 8.11 Diameter and head

```
//Diameter and head//
pathname=get_absolute_file_path('8.11.sce')
filename=pathname+filesep()+'8.11-data.sci'
```

```
exec(filename)

//Value of K*B^2:
K_B=Q/(%pi/4*D^2)*sqrt(0.5*d1/g/d2/h)
//Reynods number:
ReD1=4/%pi*Q/D/v
//By trial and error method, the value of beta is fixed at:
betta=0.66;
//K is then:
K=K_B/betta^2
//Diameter of orifice plate(in m):
Dt=betta*D
//Value of p3-p2(in N/m^2):
P1=d1*Q^2/(%pi/4*D^2)^2*(1/0.65/betta^2-1)
//Value of p1-p2(in N/m^2):
P2=d2*g*h
//Head loss between sections 1 and 3(in N-m/kg):
hLT=(P2-P1)/d1
//Expressing the permanent pressure as a fractio of the meter differential:
C=(P2-P1)/P2
printf("\n\n\nRESULTS\n\n")
printf("\n\nDiameter of the orifice: %.3f m\n\n",Dt)
printf("\n\nHead loss between secions 1 and 3: %.3f N-m/kg\n\n",hLT)
```

Chapter 9

External Incompressible Viscous Flow

Scilab code Exa 9.01 static pressure

```
//static pressure//
pathname=get_absolute_file_path('9.01.sce')
filename=pathname+filesep()+'9.01-data.sci'
exec(filename)
//Change in static pressure between sections 1 and
    2:
C=(((L-2*d1)/(L-2*d2))^4-1)*100;
printf("\n\nRESULTS\n\n")
printf("\n\nChange in static pressure between the
    sections 1 and 2: %.3f percent \n\n",C)
```

Scilab code Exa 9.04 Displacement thickness and stress

```
//Displacement thickness and stress//
pathname=get_absolute_file_path('9.04.sce')
filename=pathname+filesep()+'9.04-data.sci'
```

```
exec(filename)
//Reynolds number:
ReL=U*L/v
//FOR TURBULENT FLOW
//Disturbance thickness(in m):
dL1=0.382/ReL^0.2*L
//Displacement thickness(in m):
function y=f(n),y=dL1*(1-n^(1/7))
endfunction
dl1=intg(0,1,f)
//Skin friction coefficient:
Cf1=0.0594/ReL^0.2
//Wall shear stress(in N/m^2):
tw1=Cf1*0.5*d*U^2
//For LAMINAR FLOW:
//Disturbance thickness(in m)
dL2=5/sqrt(ReL)*L
//Displacement thickness(in m):
dl2=0.344*dL2
//Skin friction coefficient:
Cf2=0.664/sqrt(ReL)
//Wall shear stress(in N/m^2):
tw2=Cf2*0.5*d*U^2
//COMPARISON OF VALUES WITH LAMINAR FLOW
//Disturbance thickness
D=dL1/dL2
//Displacement thickness
DS=dl1/dl2
//Wall shear stress
WSS=tw1/tw2
printf("\n\nRESULTS\n\n")
printf("\n\nDisturbace thickness: %.3f m\n\n",dL1)
printf("\n\nDisplacement thickness: %.3f m\n\n",dl1)
printf("\n\nWall shear stress: %f N/m^2\n\n",tw1)
printf("\n\nCOMPARISON WIH LAMINAR FLOW\n\n\n")
printf("\n\n Disturbance thicknes: %.3f \n\n",D)
printf("\n\nDisplacement thickness: %.3f\n\n",DS)
printf("\n\nWall shear stress: %.3f \n\n",WSS)
```

Scilab code Exa 9.05 force and power

```
//force and power//
pathname=get_absolute_file_path('9.05.sce')
filename=pathname+filesep()+'9.05-data.sci'
exec(filename)
//Speed in m/s:
U=s*6076*0.305/3600
//Reynolds number:
Re=U*L/v
//Drag coefficient:
Cd=0.455/log10(Re)^2.58-1610/Re
//Area(in m^2):
A=L*(W+D)
//Drag force(in N)
Fd=Cd*A*0.5*d*U^2
//Power required to overcome skin friction drag(in W
    ):
P=Fd*U
printf("\n\nRESULTS\n\n")
printf("\n\nDrag force: %f N\n\n",Fd)
printf("\n\nPower required to overcome skin friction
    drag: %.3f W\n\n",P)
```

Scilab code Exa 9.06 Bending moment

```
//Bending moment//
pathname=get_absolute_file_path('9.06.sce')
filename=pathname+filesep()+'9.06-data.sci'
exec(filename)
//Velocity in m/sec:
```

```
V=s*5/18
//Reynolds number:
Re=d*V*D/u
//Value of Cd is obtained as:
Cd=0.35;
//Area(in m^2):
A=L^2;
//Moment about the chimney base(in N-m):
M0=Cd*A*D/4*d*V^2
printf("\n\nRESULTS\n\n")
printf("\n\nBending moment at the bottom of the
    chimney: %.3f N-m\n\n",M0)
```

Scilab code Exa 9.07 Time required

```
//Time required//
pathname=get_absolute_file_path('9.07.sce')
filename=pathname+filesep()+'9.07-data.sci'
exec(filename)
//Time required to decelerate to 100 mph(in seconds)
    :
t=(s1-s2)*2*w/(s1*s2)/Cd/d/A/g*3600/5280
printf("\n\nRESULTS\n\n")
printf("\n\nTime required to decelerate to 100 mph:
    %.3f seconds\n\n",t)
```

Scilab code Exa 9.08 Optimum cruise speed

```
//Optimum cruise speed//
pathname=get_absolute_file_path('9.08.sce')
filename=pathname+filesep()+'9.08-data.sci'
exec(filename)
//Plotting velocity with drag force
```

```
V=175:25:455;

[m n]=size(V);
for i=1:n
  CL(i)=2*W/p*(3600/V(i)/5280)^2/A;
  Cd(i)=Cd0+CL(i)^2/%pi/ar;
  Fd(i)=Cd(i)/CL(i)*W;
  FD(i)=Fd(i)/1000;
end
plot(V,FD)
xtitle('Flight speed vs thrust','Flight Speed(in mph)','Drag Force(in 1000lbf)')
//Optimum cuise speed at speed level is obtained to be 320 mph from the graph.
Vosl=320;
//Ratio of speeds at 30000 ft and at sea level is given by:
r=sqrt(1/0.375);
//Stall speed at 30000ft is(in mph):
Vs3=Vssl*r;
//Optimum Cruise speed at 30000ft(in mph):
Vo3=Vosl*r;
printf("\n\nRESULTS\n\n")
printf("\n\nOptimum cruise speed at sea level: %.3f mph\n\n",Vosl)
printf("\n\nStall speed at 30000 ft: %.3f mph\n\n",Vs3)
printf("\n\nOptimum cruise speed at 30000 ft: %.3f\n\n",Vo3)
```

Scilab code Exa 9.09 Aerodynamic and Radius

```
//Aerodynamic and Radius//
pathname=get_absolute_file_path('9.09.sce')
filename=pathname+filesep()+'9.09-data.sci'
```

```
exec(filename)
//Reynolds number:
//Value of wD/2V:
W=0.5*N*D/1000/V*2*%pi/60
Red=V*D/v;
//For this value, CL is obtained as:
CL=0.3;
//Aerodynamic lift(in N):
FL=%pi/8*CL*(D/1000)^2*d*V^2;
//Radius of curvature of the path in the vertical
    plane(in m) with topspin:
Rts=V^2/(g+FL/(m/1000));
//Radius of curvature without topspin(in m):
Rwts=V^2/g;
printf("\n\nRESULTS\n\n")
printf("\n\nAerodynamic lift acting on the ball:%.3f
    N\n\n",FL)
printf("\n\nRadius of curvature of the path when
    ball has topspin:%.3f m\n\n",Rts)
printf("\n\nRadius of curvature of the path when
    ball has topspin: %.3f m\n\n",Rwts)
```

Chapter 10

Fluid Machinery

Scilab code Exa 10.01 input and power

```
//input and power//
pathname=get_absolute_file_path('10.01.sce')
filename=pathname+filesep()+'10.01-data.sci'
exec(filename)
//Impeller exit width b2(in feet):
b2=Q*12/(2*%pi*R2*Vrb2*7.48*60)
//Torque of the Shaft, Tshaft(in ft-lbf):
Tshaft=w*R2^2*p*Q*2*%pi/3600/7.48/144
//Power, Wm(in hp):
Wm=w*Tshaft*2*%pi/60/550
printf("\n\nRESULTS\n\n")
printf("\n\nImpeller exit width: %.3f feet\n\n",b2)
printf("\n\Torque input: %.3f ft-lbf\n\n",Tshaft)
printf("\n\nPower: %.3f hp\n\n",Wm)
```

Scilab code Exa 10.02 volume and power

```
//volume and power//
```

```
pathname=get_absolute_file_path('10.02.sce')
filename=pathname+filesep()+'10.02-data.sci'
exec(filename)
U=0.5*(Dh+Dt)/2*1200*2*%pi/60
k=tand(alpha1)+cotd(betta1)
Vn1=U/k
V1=Vn1/cosd(alpha1)
Vt1=V1*sind(alpha1)
Vrb1=Vn1/sind(betta1)
//Volume flow rate (in m^3/sec):
Q=%pi/4*Vn1*(Dt^2-Dh^2)
k=(U-Vn1*cotd(betta2))/Vn1
alpha2= atand(k)
V2=Vn1/cosd(alpha2)
Vt2=V2*sind(alpha2)
//Rotor Torque (in N-m):
Tz=p*Q*(Dh+Dt)/4*(Vt2-Vt1)
//Power required (in W):
Wm=w*2*%pi/60*Tz
printf("\n\nRESULTS\n\n")
printf("\n\nVolume flow rate: %.3f m^3/sec\n\n",Q)
printf("\n\nRotor Torque: %.3f N-m\n\n",Tz)
printf("\n\nPower required: %.3f W\n\n",Wm)
```

Scilab code Exa 10.03 Pump Power

```
//Pump Power//
pathname=get_absolute_file_path('10.03.sce')
filename=pathname+filesep()+'10.03-data.sci'
exec(filename)
[nQ mQ]= size(Q);
[nps mps]=size(ps);
[npd mpd]= size(pd);
[nI mI]= size(I);
//Correct measured static pressures to he pump
```

```
   centreline p1, p2(in psig):
j=1:mps;
 p1=ps(j)+px*g*zs/144
  j=1:mpd;
 p2=pd(j)+px*g*zd/144
 //The value of Pump head(in feet):
 j=1:mps;
 Hp=(p2(j)-p1(j))/(px*g)*144
 //Values of Hydraulic Power delivered(in hp):
  j=1:mps;
  Wh=Q(j).*(p2(j)-p1(j))/7.48/60*144/550
  //Values of motor power output(in hp):
 j=1:mI;
 Pin=Effm*sqrt(3)*PF*E*I(j)/746
 //Values of Pump Efficiecy:
  j=1:mI;
 Effp= Wh(j)./Pin(j)*100
 //Plotting pump characteristics:
  plot(Q,Hp,"-o")
 plot(Q,Pin,"-+")
 plot(Q,Effp,"-*")
 xtitle('Pump Characteristics','Volume flow rate(in
     gpm)',['Pump Efficincy(%)      ','    Pump Head(
    in feet)    ','     Pump Power input(in hp)    '])
 legend('Hp','Pin','Effp')
```

Scilab code Exa 10.06 Specific and relation

```
//Specific and relation//
pathname=get_absolute_file_path('10.06.sce')
filename=pathname+filesep()+'10.06-data.sci'
exec(filename)
//Specific speed in Us customary units:
Nscu=N*Qus^0.5/Hus^0.75
//Conversion to SI units:
```

```
w=1170*2*%pi/60;
Qsi=Qus/7.48/60*0.305^3;
Hsi=Hus*0.305;
//Energy per unit mass is:
h=g*Hsi;
//Specific speed in SI units:
Nssi=w*Qsi^0.5/h^0.75
//Conversion to hertz:
whz=N/60;
//Specific speed in European units:
Nseu=whz*Qsi^0.5/65.5^0.75
//Relation between specific speeds in Us customary
    units and European units:
Conversionfactor1=Nscu/Nseu
//Relation between specific speeds in Us customary
    units and SI units:
Conversionfactor2=Nscu/Nssi
printf("\n\nRESULTS\n\n")
printf("\n\nSpecific speed in US customary units: %
    .3f \n\n",Nscu)
printf("\n\nSpecific speed in SI units: %.3f \n\n",
    Nssi)
printf("\n\nSpecific speedin European units: %.3f \n
    \n",Nseu)
printf("\n\nRelation between specific speeds in Us
    customary units and European units: %.3f \n\n",
    Conversionfactor1)
printf("\n\nRelation between specific speeds in Us
    customary units and SI units: %.3f \n\n",
    Conversionfactor2)
```

Scilab code Exa 10.07 Comparison of head

```
//Comparison of head//
pathname=get_absolute_file_path('10.07.sce')
```

```
filename=pathname+filesep()+'10.07-data.sci'
exec(filename)
//Volume flow rate(in gpm) at shut off condition for
    N2:
Q2so=N2/N1*Q1so
//Volume flow(in gpm) rate at best efficiency for N2
    :
Q2be=N2/N1*Q1be
//Relation between pump heads:
head_relation=(N2/N1)^2
//Head(in feet) at shut off condition for N2:
H2so=(N2/N1)^2*H1so
//Head(in feet) at best efficiency condition for N2:
H2be=(N2/N1)^2*H1be
Q1=[Q1so Q1be];
Q2=[Q2so Q2be];
H1=[H1so H1be];
H2=[H2so H2be];
plot(Q1,H1,"-o")
plot(Q2,H2,"-*")
xtitle('Comparison of head for both conditions','
    Volume Flow Rate','Head')
legend('1170','1750')
```

Scilab code Exa 10.08 NPSHA and NPSHR

```
//NPSHA and NPSHR//
pathname=get_absolute_file_path('10.08.sce')
filename=pathname+filesep()+'10.08-data.sci'
exec(filename)
//Diameter of pipe (in feet):
Df= Di/12
//Area of crossection of pipe(in ft^2):
A=%pi/4*Df^2
//Velocity of flow(in ft/sec):
```

```
V=Q/7.48/A/60
//For water at T=80F, viscosity=0.927e-5 ft^2/sec, Reynolds number:
Re=V*Df/v
//Friction loss Coefficient for this value of Re:
f=0.0237;
//For cast iron, roughness(in feet):
e=0.00085
//e/D is:
e/Df
//Total head loss(in feet):
HL=K+f*(SE+OGV)+f*(L/Df)+1
//The heads are(in feet):
H1=patm*144/(p*g)
Vh=V^2/2/g
//Suction head(in feet):
Hs=H1+h-HL*Vh
//NPSHA(in feet):
NPSHA=Hs+Vh-Hv1
//For a flow rate of 1000 gpm,NPSHR(in feet) for water at 80 F
NPSHR=10
//PLOTTING NPSHA AND NPSHR VERSUS VOLUME FLOW RATE:
//For 80 F
Qp=0:100:1500;
[nQp mQp]=size(Qp);
 for j=1:mQp;
 Vp(j)=Qp(j)/(7.48*A*60);
 Vhp(j)=(Vp(j))^2/2/g;
 Hs(j)=H1+h-HL*Vhp(j);
 end

for j=1:mQp;
  NPSHAp1(j)=Hs(j)+(Vhp(j))-Hv1;
end

plot(Qp,NPSHAp1,"-+")
plot(Qh,NPSHRp,"-o")
```

```
xtitle('Suction head vs Flow rate','Volume flow rate
    (gpm)','Suction Head(feet)');
printf("\n\nType (Resume) to continue or (abort) to
    end\n\n")
legend('NPSHA','NPSHR')
pause
clf

//For 180 F
for j=1:mQp;
  NPSHAp2(j)=Hs(j)+(Vhp(j))-Hv2;
end
plot(Qp,NPSHAp2,"-+")
plot(Qh,NPSHRp,"-o")
xtitle('Suction head vs Flow rate','Volume flow rate
    (gpm)','Suction Head(feet)');
legend('NPSHA','NPSHR')
printf("\n\nRESULTS\n\n")
printf("\n\nNPSHA at Q=1000 gpm of water at 80 F: %
    .2f ft\n\n",NPSHA)
printf("\n\nNPSHR at Q=1000 gpm of water at 80 F: %
    .1f ft\n\n",NPSHR)
```

Scilab code Exa 10.11 Performance curves

```
//Performance curves//
pathname=get_absolute_file_path('10.11.sce')
filename=pathname+filesep()+'10.11-data.sci'
exec(filename)
[nQ mQ]= size(Q1);
[np mp]= size(p1);
[nP mP]= size(P1);
//Volume flow rate for fan 2(in cfm):
```

```
j=1:mQ;
Q2=Q1(j)*(N2/N1)*(D2/D1)^3
//Pressure values for fan 2(in inches of H2O):
j=1:mp;
p2=p1(j)*(d2/d1)*((N2/N1)^2)*((D2/D1)^2)
//Power values for fan 2(in hp):
j=1:mP;
P2=P1(j)*(d2/d1)*((N2/N1)^3)*((D2/D1)^5)
plot(Q2,p2)
xtitle('Performance curves','Volume flow rate(in cfm
    )','Pressure head(in inches of water)')
printf("\n\nType (resume) to continue or (abort) to
    exit\n\n")
pause
clf
plot(Q2,P2)
xtitle('Performance curves','Volume flow rate(in cfm
    )','Power(in hp)')
printf("\n\nType (resume) to continue or (abort) to
    exit\n\n")
pause
clf
plot(Q2,Eff)
xtitle('Performance curves','Volume flow rate(in cfm
    )','Eficiency(in percentage)')
//Specific speed of fan(in US customary units) at
    operating point:
Nscu= 1150*110000^0.50*0.045^0.75/7.4^0.75
//Specific speed of fan (in SI units) at operating
    point:
Nssi=120*3110^0.5*0.721^0.75/1.84e3^0.75
```

Scilab code Exa 10.12 Power required

```
//Power required//
```

```
pathname=get_absolute_file_path('10.12.sce')
filename=pathname+filesep()+'10.12-data.sci'
exec(filename)
//From given graph, for maximum delivery condition, Q=48.5gpm.
//Volume of oil per revolution delivered by the pump (in in^3/rev):
vc=Qe/N*231
//Volumetric Effciency of pump at max flow:
Effv=vc/va
//Operating point of the pump is found to be at 1500 psig ,Q=46.5gpm
//Power delivered by the fluid(in hp):
Pf=Qo*po1/7.48/60*144/550
//Input power(in hp):
Pi=Pf/Effp
//The power delivered to the load(in hp):
Pl=Q*(po1)/7.48/60*144/550
//Power dissipated by throttling(in hp):
Pd=Pf-Pl
//The dissipation with the variable displacement pump(in hp):
Pvd=Q*(po2-po1)/7.48/60*144/550
//Power required for te load sensing pump if pump pressure is 100psi above that required by the load(in hp):
Pls=Q*100/7.48/60*144/550
printf("\n\nRESULTS\n\n")
printf("\n\nVolume of oil per revolution delivered by the pump: %.3f in^3/rev\n\n",vc)
printf("\n\nRequired pump power input: %.3f hp\n\n",Pi)
printf("\n\nPower deliverd to the load: %.3f hp\n\n",Pl)
printf("\n\nPower dissipated by throttling: %.3f hp\n\n",Pd)
printf("\n\nThe dissipation with the variable displacement pump: %.3f hp\n\n",Pvd)
```

```
printf("\n\nPower required for te load sensing pump if pump pressure is 100psi above that required by the load: %.3f hp\n\n",Pls)
```

Scilab code Exa 10.14 propeller

```
//propeller//
pathname=get_absolute_file_path('10.14.sce')
filename=pathname+filesep()+'10.14-data.sci'
exec(filename)
//Propeller Thrust(in MN) :
Ft=P/V
//Required power input  to the propeller(in MW):
Pin=P/Eff
//Calculating value of D(in m):
nD=V/J
D=(Ft*10^6/p/(nD)^2/Cf)^0.5
//Operating speed (in rpm) is given by:
n=nD/D*60
printf("\n\nRESULTS\n\n")
printf("\n\nDiameter of the single propeller required to pwer the ship:%.3f m\n\n",D)
printf("\n\nOperating speed of the propeller: %.3f rpm\n\n",n)
```

Scilab code Exa 10.16 Actual

```
//Actual//
pathname=get_absolute_file_path('10.16.sce')
filename=pathname+filesep()+'10.16-data.sci'
exec(filename)
//Tip speed ratio of windmill:
X=N*2*%pi/60*D/2/(V*5/18)
```

```
//Accounting for whirl,max attainable efficiency is:
Efw=0.53;
//Kinetic energy flux(in W) is given by:
KEF=0.5*p*(V*5/18)^3*%pi*(D/2)^2
//Actual Efficiency:
Effa=Po/KEF
//The maximum possible thrust occurs for an
   interference factor of:
amax=0.5;
//Thrust(in W):
Kx=p*(V*5/18)^2*%pi*(D/2)^2*2*amax*(1-amax)
printf("\n\nRESULTS\n\n")
printf("\n\nTip speed ratio of windmill:%.3f\n\n",X)
printf("\n\nActual Efficiency: %.3f\n\n",Effa)
printf("\n\nActual Thrust: %.3f W\n\n",Kx)
```

Chapter 11

Introduction to Compressible Flow

Scilab code Exa 11.01 Change

```
//Change//
pathname=get_absolute_file_path('11.01.sce')
filename=pathname+filesep()+'11.01-data.sci'
exec(filename)
//Density of air at entry:
d1=p1*10^3/R/T1
//Area(in m^2):
A=m/d1/V1
//Change in enthalpy of air(in kJ/kg):
dh=cp*(T2-T1)
//Change in internal energy of air(in kJ/kg):
du=cv*(T2-T1)
//Change in entropy(in kJ/(kg-K)):
ds=cp*log(T2/T1)-R/1000*log(p2/p1)
printf("\n\nRESULTS\n\n")
printf("\n\nDuct Area: %.3f m^2\n\n",A)
printf("\n\nChange in enthalpy of air: %.3f kJ/kg\n\
    n",dh)
printf("\n\nChange in internal energy of air:%.3f kJ
```

```
/kg\n\n",du)
printf("\n\nChange in entropy: %.3f kg-K\n\n",ds)
```

Scilab code Exa 11.03 Speed of sound

```
//Speed of sound//
pathname=get_absolute_file_path('11.03.sce')
filename=pathname+filesep()+'11.03-data.sci'
exec(filename)
//Values of altitude(in m):
Al=0:1000:15000
[nAl mAl]=size(Al);
//Values of temperature at given altitudes(in K):
T=[288.2 281.7 275.2 268.7 262.2 255.7 249.2 242.7 236.2 229.7 223.3 216.8 216.7 216.7 216.7 216.7];
[nT mT]=size(T);
//Values of speed of sound at these altitudes(in m/sec):
j=1:mT;
c=sqrt(k*R*T(j))
//Speed of sound at sea level(in m/sec):
c1=sqrt(k*R*T(1))
plot(c,Al)
xtitle('Variation of sound speed with altitude','Speed of sound(m/sec)','Altitude(m)')
printf("\n\nRESULTS\n\n")
printf("\n\nSpeed of sound at sea level: %.3f m/sec\n\n",c1)
```

Scilab code Exa 11.04 pressure and change

```
//pressure and change//
pathname=get_absolute_file_path('11.04.sce')
```

```
filename=pathname+filesep()+'11.04-data.sci'
exec(filename)
//Mach number at entry:
M1=V1/sqrt(k*R*T1)
//Stagnation pressure at entry(in kPa):
p01=p1*(1+(k-1)/2*M1^2)^(k/(k-1))
//Stagnation temperature at entry(in K):
T01=T1*(1+(k-1)/2*M1^2)
//Static pressure at exit(in kPa):
p2=p02/(1+(k-1)/2*M2^2)^(k/(k-1))
//Temperature at exit(in K):
T2=T02/(1+(k-1)/2*M2^2)
//Change in entropy(in kJ/kg-K):
ds=cp*log(T2/T1)-R/1000*log(p2/p1)
printf("\n\nRESULTS\n\n")
printf("\n\nStagnation pressure at entry: %.3f kPa\n
    \n",p01)
printf("\n\nStagnation temperature at entry: %.3f K\
    n\n",T01)
printf("\n\nStatic pressure at exit: %.3f kPa\n\n",
    p2)
printf("\n\nTemperature at exit: %.3f K\n\n",T2)
printf("\n\nChange in entropy: %.3f kJ/kg-K\n\n",ds)
```

Chapter 12

Steady One Dimensional Compressible Flow

Scilab code Exa 12.01 pressure and area

```
//pressure and area//
pathname=get_absolute_file_path('12.01.sce')
filename=pathname+filesep()+'12.01-data.sci'
exec(filename)
//Here the stagnation quantities are constant.
// Stagnation temperature(in K):
T0=T1*(1+(k-1)/2*M1^2)
//Stagnation pressure(in kPa):
p0=p1*((1+(k-1)/2*M1^2)^(k/(k-1)))
//Finding T2/T1:
T=t2/t1
//Temperature at exit(in K):
T2=T*T1
//Finding p2/p1:
P=P2/P1
//Pressure at exit(in kPa):
p2=P2*p1
//Density of air at exit(in kg/m^3):
d2=p2*10^3/R/T2
```

```
//Velocity of air at exit(in m/sec):
V2=M2*sqrt(k*R*T2)
//Finding A2/A1:
a=a2/a1
//Area at exit(in m^2):
A2=a*A1
printf("\n\nRESULTS\n\n")
printf("\n\nStagnation temperature: %.3f K\n\n",T0)
printf("\n\nStagantion pressure: %.3f kPa\n\n",p0)
printf("\n\nTemperature a exit %.3f K\n\n",T2)
printf("\n\nPressure at exit: %.3f kPa\n\n",p2)
printf("\n\nDensity of air at exit: %.3f kg/m^3\n\n"
    ,d2)
printf("\n\nVelocity of air at exit: %.3f m/sec\n\n"
    ,V2)
printf("\n\nArea at exit: %.3f \n\n",A2)
```

Scilab code Exa 12.02 Mass flow

```
//Mass flow//
pathname=get_absolute_file_path('12.02.sce')
filename=pathname+filesep()+'12.02-data.sci'
exec(filename)
//Checking for chocking:
c=pb/p0;
if(c<=0.528)
  //choked
else
  //Not choked
  //Therefore pressure at exit = back pressure
  pe=pb;
  //Mach number at exit:
  Me=(((p0/pe)^((k-1)/k)-1)*(2/(k-1)))^0.5
  //Temperature at exit(in K):
  Te=T0/(1+(k-1)/2*Me^2)
```

```
  //Velocity at exit(in m/sec):
  Ve=Me*sqrt(k*R*Te)
  //Density at exit(in kg/m^3):
  de=pe*10^3/R/Te
  //Mass flow rate of air(kg/sec):
  m=de*Ve*Ae
end;
printf("\n\nRESULTS\n\n")
printf("\n\nMach number at exit: %.3f\n\n",Me)
printf("\n\nMass flow rate of air: %.3f kg/sec\n\n",
   m)
```

Scilab code Exa 12.03 mass and area

```
//mass and area//
pathname=get_absolute_file_path('12.03.sce')
filename=pathname+filesep()+'12.03-data.sci'
exec(filename)
//Saturation pressure(in psia):
p0=p1*(1+(k-1)/2*M1^2)^(k/(k-1))
//Checking for choking:
x=pb/p0;
if(x>0.528)
  //Not choked
else
  //choked
end
//As there is choking:
Mt=1;
//Velocity at entry:
V1=M1*sqrt(k*R*(T1+460)*32.2)
//Density at the entry(in lbm/ft^3):
d1=p1/(R*(T1+460))*144
//Mass flow rate(in lbm/sec):
m=d1*V1*A1
```

```
//Finding the valueof A1/A*;
A=1/M1*((1+(k-1)/2*M1^2)/(1+(k-1)/2))^((k+1)/(2*(k-1)))
//For choked flow, At=A*
At=A1/A
printf("\n\nRESULTS\n\n")
printf("\n\nMach number at throat: %.3f\n\n",Mt)
printf("\n\nMass flow rate: %.3f lbm/sec\n\n",m)
printf("\n\nArea at throat: %.3f ft^2\n\n",At)
```

Scilab code Exa 12.04 throat

```
//throat//
pathname=get_absolute_file_path('12.04.sce')
filename=pathname+filesep()+'12.04-data.sci'
exec(filename)
//Temperature at the throat(in K):
Tt=T0/(1+(k-1)/2*Mt^2)
//Pressure at throat(in kPa):
pt=p0*(Tt/T0)^(k/(k-1))
//Density at throat(in kg/m^3):
dt=pt*1000/R/Tt
//Velocity at the throat(in m/s):
Vt=Mt*sqrt(k*R*Tt)
//Value of At/A*:
Ax=1/Mt*((1+(k-1)/2*Mt^2)/(1+(k-1)/2))^((k+1)/(2*(k-1)))
//Stagnation properties are constant
//As a result pressure at exit,
pe=pb;
//The Mach number at the exit is therefore given by
Me=sqrt(((p0/pe)^((k-1)/k)-1)*2/(k-1))
//Calculating the value of Ae/A*:
Ay=1/Me*((1+(k-1)/2*Me^2)/(1+(k-1)/2))^((k+1)/(2*(k-1)))
```

```
//Value of A*(in m^2):
A_star=Ae/Ay
//Area at throat(in m^2):
At=Ax*A_star
printf("\n\nRESULTS\n\n")
printf("\n\nTemperature at the throat: %.3f K\n\n",
   Tt)
printf("\n\nPressure at throat: %.3f kPa\n\n",pt)
printf("\n\nDensity at throat: %.3f kg/m^3\n\n",dt)
printf("\n\nVelocity at the throat: %.3f m/sec\n\n",
   Vt)
printf("\n\nMach number at the exit: %.3f\n\n",Me)
printf("\n\nArea at throat: %.3f m^2\n\n",At)
```

Scilab code Exa 12.05 number and flow

```
//number and flow//
pathname=get_absolute_file_path('12.05.sce')
filename=pathname+filesep()+'12.05-data.sci'
exec(filename)
//Mach number at the exit:
Me=sqrt(((p0/pe)^((k-1)/k)-1)*2/(k-1))
//Temperature at exit(in K):
Te=T0/(1+(k-1)/2*Me^2)
//Mass flow rate(in kg/s):
m=pe*1000*Me*sqrt(k/R/Te)*Ae
printf("\n\nRESULTS\n\n")
printf("\n\nMach number at the exit: %.3f \n\n",Me)
printf("\n\nMass flow rate: %.3f kg/sec\n\n",m)
```

Scilab code Exa 12.06 mass and volume

```
//mass and volume//
```

```
pathname=get_absolute_file_path('12.06.sce')
filename=pathname+filesep()+'12.06-data.sci'
exec(filename)
//Mach umber at section 1:
M1=sqrt((2/(k-1)*((p0/p1)^((k-1)/k)-1)))
//Temperature at section 1(in K):
T1=T0/(1+(k-1)/2*M1^2)
//Density at section 1(in kg/m^3):
d1=p1*1000/R/T1
//Velocity at section1(in m/sec):
V1=M1*sqrt(k*R*T1)
//Area at section 1(in m^2):
A1=%pi/4*D^2
//Mass flow rate(in kg/sec):
m=d1*A1*V1
//Mach number at section 2:
M2=sqrt((2/(k-1))*((T0/T2)-1))
//Velocity at section 2(in m/sec):
V2=M2*sqrt(k*R*T2)
//Density at section 2(in kg/m^3):
d2=d1*V1/V2
//Pressure at section 2(in kPa):
p2=d2/1000*R*T2
//Stagnation pressure at section 2(in kPa):
p02=p2*(1+(k-1)/2*M2^2)^(k/(k-1))
//Force exerted on control volume by duct wall(in N):
F=(p2-p1)*1000*A1+m*(V2-V1)
printf("\n\nRESULTS\n\n")
printf("\n\nMass flow rate: %.3f kg/sec\n\n",m)
printf("\n\nLocal isentropic stagnation pressure at section 2:%.3f kPa\n\n",p02)
printf("\n\nForce exerted on control volume by duct wall:%.3f N\n\n",F)
```

Scilab code Exa 12.07 length

```
//length//
pathname=get_absolute_file_path('12.07.sce')
filename=pathname+filesep()+'12.07-data.sci'
exec(filename)
//Mach number at section 1:
M1= sqrt(2/(k-1)*((p0/(p0+p1))^((k-1)/k)-1))
//Temperature at section 1(in K):
T1=T0/(1+(k-1)/2*(M1)^2)
V1=M1*sqrt(k*R*T1)
//Pressure at section 1(in kPa):
p1=g*dHg*(760-18.9)*10^-3
//Density at section 1(in kg/m^3):
d1=p1/R/T1
//At M1=0.190,
//(p/p*)1:
P1=5.745
// (fLmax/Dh)1:
F1=16.38
//Value of L13(in m):
L13=F1*D/f
//Value of (p/p*)2:
P2=p2/p1*P1
//For this value, Value of M2 is obtained as 0.4
M2=0.4;
//For M=0.4, fLmX/D=2.309
F2=2.309
//Value of L23(in m):
L23=F2*D/f
//Length of duct between section 1 and 2(in m):
L12=L13-L23
printf("\n\nRESULTS\n\n")
printf("\n\nLength of duct required for choking from
    section 1: %3f m\n\n",L13)
printf("\n\nMach number section 2: %.3f \n\n",M2)
printf("\n\Length of duct between section 1 and 2: %
    .3f m\n\n",L12)
```

Scilab code Exa 12.08 velocity and entropy

```
//velocity and entropy//
pathname=get_absolute_file_path('12.08.sce')
filename=pathname+filesep()+'12.08-data.sci'
exec(filename)
//Density at section 1(in lbm/ft^3):
d1=p1*144/R/T1
//Velocity at section 2(in ft/sec):
V2=(p1-p2)*144/d1/V1*32.2+V1
//Density at section 2(in lbm/ft3):
d2=d1*V1/V2
//Temperature at section 2(in R):
T2=p2/d2/R*144
//Mach number at section 2:
M2=V2/sqrt(k*R*32.16*T2)
//Stagnation Temperature at section 2(in R):
T02=T2*(1+(k-1)/2*M2^2)
//Stagnation pressure at section 2 (in psia):
p02=p2*(T02/T2)^(k/(k-1))
//Mach Number at section 1:
M1=V1/sqrt(k*R*32.16*T1)
//Stagnation temperature at section 1(in R):
T01=T1*(1+(k-1)/2*M1^2)
//Energy added(in Btu/lbm):
E=Cp*(T02-T01)
//Change in entropy(in Btu/(lbm-R)):
dS=Cp*log(T2/T1)-(Cp-Cv)*log(p2/p1)
printf("\n\nRESULTS\n\n")
printf("\n\nVelocity at section 2: %.3f ft/sec\n\n",
    V2)
printf("\n\nDensity at section 2: %.3f lbm/ft^3\n\n"
    ,d2)
printf("\n\nTemperature at section 2: %.3f R\n\n",T2
```

```
)
printf("\n\nStagnation Temperature at section 2: %.3f R\n\n",T02)
printf("\n\nStagnation pressure at section 2: %.3f psia\n\n",p02)
printf("\n\nEnergy added: %.3f Btu/lbm\n\n",E)
printf("\n\nChange in entropy: %.3f Btu/(lbm-R)\n\n",dS)
```

Scilab code Exa 12.09 Temperature and entropy

```
//Temperature and entropy//
pathname=get_absolute_file_path('12.09.sce')
filename=pathname+filesep()+'12.09-data.sci'
exec(filename)
//Mach nuber at section 1:
M1=V1/sqrt(k*R*T1)
//For these value of M1 and M2, the following values are obtained:
//(To/T0*)1:
t01=0.7934;
//(T0/T0*)2:
t02=0.9787;
//(p0/p0*)1:
P01=1.503;
//(p0/p0*)2:
P02=1.019;
//(T/T*)1:
t1=0.5289;
//(T/T*)2:
t2=0.9119;
//(p/p*)1:
P1=0.3636;
//(p/p*)2:
P2=0.7958;
```

```
//(V/V*)1:
v1=1.455;
//(V/V*)2:
v2=1.146;
//Value of T2/T1:
t=t2/t1
//Temperature at section 2(in K):
T2=t*T1
//Value of p2/p1:
p=P2/P1
//Pressure at section 2(in kPa):
p2=p*p1
//Value of V2/V1:
v=v2/v1
//Velocity at section 2(in m/sec):
V2=v*V1
//Density at section 2(in kg/m^3):
d2=p2*1000/R/T2
//At M1, T/T0=0.5556
T01=T1/0.5556
//At M2, T/T0=0.7764
T02=T2/0.7764
//Heat added(in kJ/kg):
E=Cp*(T02-T01)
//Change in entropy(kJ/kg-K):
dS=Cp*log(T2/T1)-R*log(p2/p1)/1000
printf("\n\nRESULTS\n\n")
printf("\n\nTemperature at section 2: %.3f K\n\n",T2)
printf("\n\nPressure at section 2: %.3f kPa\n\n",p2)
printf("\n\nVelocity at section 2: %.3f m/sec\n\n",V2)
printf("\n\nDensity at section 2: %.3f kg/m^3\n\n",d2)
printf("\n\nStagnation temperature at section 2: %.3f K\n\n",T02)
printf("\n\nHeat added: %.3f kJ/kg\n\n",E)
printf("\n\nChange in entropy: %.3f kJ/kg\n\n",dS)
```

Scilab code Exa 12.10 Temperature

```
//Temperature//
pathname=get_absolute_file_path('12.10.sce')
filename=pathname+filesep()+'12.10-data.sci'
exec(filename)
//Density at section 1(in kg/m^3):
d1=p1*1000/R/T1
//Mach number at section 1:
M1=V1/sqrt(k*R*T1)
//Stagnation temperature at section 1(in K):
T01=T1*(1+(k-1)/2*M1^2)
//Stagnation pressure at section 1(in kPa):
p01=p1*(1+(k-1)/2*M1^2)^(k/(k-1))
//The following values are obtained from the
    appendix:
//po2/p01:
p0=0.7209;
//T2/T1:
T=1.687;
//p2/p1:
p=4.5;
//V2/V1:
V=0.3750;
//Temperature at section 2 (in K):
T2=T*T1
//Pressure at section 2(in kPa):
p2=p*p1
//Velocity at section 2(in m/sec):
V2=V*V1
//Density at section 2 (in kg/m^3):
d2=p2*1000/R/T2
//Stagnation pressure at section 2(in kPa):
p02=p0*p01
```

```
//Stagnation temperature at section 2(in K):
T02=T01;
//Change in entropy(in kJ/(kg-K)):
dS=-R/1000*log(p0)
printf("\n\nRESULTS\n\n")
printf("\n\nTemperature at section 2 : %.3f K\n\n",
    T2)
printf("\n\nPressure at section 2: %.3f kPa\n\n",p2)
printf("\n\nVelocity at section 2: %.3f m/sec\n\n",
    V2)
printf("\n\nDensity at section 2 : %.3f kg/m^3\n\n",
    d2)
printf("\n\nStagnation pressure at section 2: %.3f
    kPa\n\n",p02)
printf("\n\nChange in entropy: %.3f kg-K\n\n",dS)
printf("\n\nStagnation temperature at section 2: %.3
    f K\n\n",T02)
```