Scilab Textbook Companion for Transport Phenomena by R. S. Brodkey And H. C. Hershey[1]

Created by
Ankit Vijay
engineering
Chemical Engineering
IIT-BHU
College Teacher
Self
Cross-Checked by
Ganesh R

August 10, 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: Transport Phenomena

Author: R. S. Brodkey And H. C. Hershey

Publisher: McGraw - Hill Book Company, New York

Edition: 1

Year: 1988

ISBN: 0-07-007963-3

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

introduction to transport phenomena

Scilab code Exa 1.1 fundamental variables and units

```
clc;
warning('off');
printf("\n\n example1.1 - pg6");
// given
v=0.01283;  //[m^3] - volume of tank in m^3
v=0.4531;  //[ft^3] - volume of tank in ft^3
p=2;  //[atm] - pressure
T=1.8*300;  //[degR] - temperature
R=0.73;  //[(atm*ft^3)/(lbmol*degR)] - gas constant
// using the equation of state for an ideal gas pv=
   nRT
n=(p*v)/(R*T);
disp(n,"no. of moles,n=");
xN2=0.5;  // fractiom of N2 in tank
nN2=xN2*n;
Ca=nN2/v;
printf("\n\n Ca=%elb*mol/ft^3",Ca);
```

Scilab code Exa 1.2 The role of intermolecular forces

```
clc;
warning("off");
printf("\n\n example1.2 - pg9");
// given
// the three unknowns are x,y,z
// the three equations are-
// x+y+z=1500
// (1) 0.05*x+0.15*y+0.40*z=1500*0.25
// (2) 0.95*x+0.00*y+0.452*z=1500*0.50
a=[1 1 1;0.05 0.15 0.40;0.95 0 0.452];
d=[1500;1500*0.25;1500*0.50];
ainv=inv(a);
sol=ainv*d;
printf("\n\n the amount of concentrated HNO3 is %fkg
    \n the amount of concentrated H2SO4 is %fkg\n the
     amount of waste acids is %fkg",sol(2),sol(1),sol
    (3));
```

Chapter 2

molecular transport mechanisms

Scilab code Exa 2.1 the analogous form

```
clc;
warning('off');
printf("\n\n example2.1 - pg28");
// given
deltax=0.1;  //[m] - thickness of copper block
T2=100;  //[degC] - temp on one side of copper block
T1=0;  //[degC] - temp on other side of the copper
   block
k=380;  //[W/mK] - thermal conductivity
// using the formula (q/A)*deltax=-k*(T2-T1)
g=-k*(T2-T1)/deltax;
g1=(g/(4.184*10000));
printf("\n\n The steady state heat flux across the
   copper block is\n q/A=%fW/m^2 \n or in alternate
   units is \n q/A=%fcal/cm*sec",g,g1);
```

Scilab code Exa 2.2 the analogous form

```
clc;
warning('off');
printf("\n\n example2.2 - pg29");
// given
dely=0.1;   //[m] - distance between two parralel plates
delUx=0.3;   //[m/sec] - velocity of a plate
mu=0.001;   //[kg/m*sec] - viscosity
// using the formula tauyx=F/A=-mu*(delUx/dely)
tauyx=-mu*(delUx/dely);
printf("\n\n the momentum flux and the the force per unit area,(which are the same thing) is\n tauyx=F/A=%fN/m^2",tauyx);
```

Scilab code Exa 2.3 heat transfer

```
clc;
warning('off');
printf("\n\n example2.3 - pg30");
// given
tauyx=-0.003;   //[N/m^2] - momentum flux
dely=0.1;   //[m] - distance between two parralel plates
mu=0.01;   //[kg/m*sec] - viscosity
// using the formula tauyx=F/A=-mu*(delUx/dely)
delUx=-((tauyx*dely)/mu)*100;
printf("\n\n Velocity of the top plate is \n deltaUx=%fcm/sec",delUx);
```

Scilab code Exa 2.5 heat transfer

```
clc;
warning('off');
printf("\n\n example2.5 - pg31");
// given
d=0.0013;  //[m] - diameter of the tube
delx=1;  //[m] - length of the glass tube
T2=110.6;  //[degC] - temperature on one end of the
    rod
T1=0;  //[degC] - temperature on other side of the
    rod
k=0.86;  //[W/m*K] - thermal conductivity
Hf=333.5;  //[J/g] - heat of fusion of ice
// (a)using the equation (q/A)=-k*(delt/delx)
A=(%pi*d^2)/4;
q=A*(-k*(T2-T1)/delx);
printf("\n\n (a) the heat flow is \n q=%fJ/sec",q);
// (b) dividing the total heat transfer in 30minutes
     by the amount of heat required to melt 1g of ice
a=abs((q*30*60)/333.5);
printf("\n\n (b)the amount or grams of ice melted in
     30minutes is %fg",a);
```

Scilab code Exa 2.6 mass transfer

```
clc;
warning('off');
printf("\n\n example2.6 - pg36");
// given
d=1.2*10^-2;  //[m] - diameter of the hole
Ca1=0.083;  //[kmol/m^3]
Ca2=0;  //[kmol/m^3]
L=0.04;  //[m] - thickness of the iron piece
Dab=1.56*10^-3;  //[m^2/sec] - diffusion coefficient
     of CO2
A=(%pi*d^2)/4;  //area
```

```
// (a)using the formula (Na/)A=(Ja/A)=-Dab(delCa/delx)
intdCa=integrate('1','Ca',Ca2,Ca1);
intdx=integrate('1','x',0,0.04);
g=(intdCa/intdx)*Dab;
printf("\n\n (a) The molar flux with respect to stationary coordinates is\n (Na/A)=%fkmol/m^2*sec",g);
// using the formula na/A=(Na/A)*Ma
Ma=44.01; //[kg/mol] - molcular weight of co2
na=(intdCa/intdx)*Dab*Ma*A*(3600/0.4539);
printf("\n\n The mass flow rate is %flb/hr",na);
```

Scilab code Exa 2.7 mass transfer

```
clc;
warning('off');
printf("\n\n example2.7 - pg38");
// given
T=30+273.15; //[K] temperature
pA=3; //[atm] partial pressure of the component A
R=0.082057; //[atm*m^3*/kmol*K] gas constant
// (a) using the equation Ca=n/V=pA/(R*T)
Cco2=pA/(R*T);
Cco2=Cco2*(44.01);
printf("\n\n (a) The concentarion of Co2 entering is %fkg/m^3",Cco2);
// (b) using the same equation as above
pN2=(0.79)*3; //[atm] partial pressure of mitrogen(as nitrogen is 79% in air)
R=0.7302; //[atm*ft^3*lb/mol*R] - gas constant
T=T*(1.8); //[R] temperature
CN2=pN2/(R*T);
printf("\n\n (b) The concentration of N2 entering is %flbmol/ft^3",CN2);
```

```
// (c) using the same equation as above
nt=6;
nCo2=4;
nO2=2*(0.21);
nN2=2*(0.79);
yCo2=nCo2/nt;
yO2=nO2/nt;
yN2=nN2/nt;
R=82.057;  //[atm*cm^3/mol*K] - gas constant
T=30+273.15;  //[K] - temperature
pCo2=3*yCo2;
Cco2=pCo2/(R*T);
printf("\n\n (c) The concentartion of Co2 in the exit is %fmol/cm^3",Cco2);
// (d) using the same equation as above
R=8.3143;  //[kPa*m^3/kmol*K] - gas constant
pO2=3*(yO2)*(101.325);  //[kPa] - partial pressure
CO2=pO2/(R*T);
printf("\n\n (d) The concentration of O2 in the exit stream is %fkmol/m^3",CO2);
```

Scilab code Exa 2.8 mass transfer

```
clc;
warning('off');
printf("\n\n example2.8 - pg39");
// given
delx=0.3-0;  //[m] - length
d=0.05-0;  //[m] - diameter
A=(%pi*d^2)/4;  //[m^2] - area;
R=8.314*10^3;  //[N*m/kmol*K] - gas constant
xco1=0.15;  // mole prcent of co in one tank
xco2=0;  // mole percent of co in other tank
p2=1;  //[atm] - pressure in one tank
p1=p2;  //[atm] - pressure in other tank
```

```
D=0.164*10^-4;  //[m^2/sec] - diffusion coefficient
T=298.15;  //[K] - temperature
// using the formula (Na/A)=(Ja/A)=-D*(delca/delx)=-(D/R*T)*(delpa/delx);
delpa=(p2*xco2-p1*xco1)*10^5;  //[N/m^2] - pressure difference
Na=-((D*A)/(R*T))*(delpa/delx);
disp(Na)
printf("\n\n The initial rate of mass transfer of co2 is %ekmol/sec",Na);
printf("\n\n In order for the pressure to remain at 1atm, a diffusion of air must occur which is in the opposite direction and equal to %ekmol/sce",Na);
```

Scilab code Exa 2.9 momentum transfer

```
clc;
warning("off");
printf("\n\n example2.9 - pg44");
// given
A=5;  //[m^2] - area of the plates
Ft=0.083  //[N] - force on the top plate
Fb=-0.027;  //[N] - force on the bottom plate
ut=-0.3;  //[m/sec] - velocity of the top plate
ub=0.1;  //[m/sec] - velocity of the bottom plate
dely=0.01;  //[m]
delux=ut-ub;  //[m/sec]
// using the formula tauyx=F/A=-mu(delux/dely)
tauyx=(Ft-Fb)/A;
mu=tauyx/(-delux/dely);  //[Ns/m^2]
mu=mu*10^3;  //[cP]
printf("\n\n The viscosity of toulene in centipose is %fcP",mu);
```

Scilab code Exa 2.11 diffusion coefficient

```
clc;
warning('off');
printf("\n\n example2.11 - pg51");
// given
po=1;  //[atm] - pressure
p=2;  //[atm] - pressure
To=0+273.15;  //[K] - temperature
T=75+273.15;  //[K] - temperature
Do=0.219*10^-4;  //[m^2/sec];
n=1.75;
// using the formula D=Do*(po/p)*(T/To)^n
D=Do*(po/p)*(T/To)^n;
printf("\n\n The diffusion coefficient of water
    vapour in air at %fatm and %fdegC is \n D=%em^2/
    sec",p,T-273.15,D);
```

Scilab code Exa 2.12 viscosity

```
clc;
warning('off');
printf("\n\n example2.12 - pg52");
// given
T=53+273.15;  //[K] - temperature
mu1=1.91*10^-5;
mu2=2.10*10^-5;
T1=313.15;  //[K] - temperature
T2=347.15;  //[K] - temperature
// for air
// using linear interpolation of the values in table
    2.2
```

```
function b=f(a)
    b=log(mu1/a)/log(T1);
endfunction
function y=g(a)
    y=log(mu2)-log(a)-f(a)*log(T2);
endfunction
a1=10^-7;
A=fsolve(a1,g);
B=f(A);
// using the formula ln(mu)=lnA+Bln(t)
mu=%e^(log(A)+B*log(T))*10^3;  //[cP]
printf("\n\n the viscosity of air at %fdegC is %fcP",T-273.15,mu);
// similarly for water
BdivR=1646;
A=3.336*10^-8;
mu=A*%e^(BdivR/T)*10^5 //[cP]
printf("\n\n the viscosity of water at %fdegC is %fcP",T-273.15,mu);
```

Chapter 3

molecular transport and the general property balance

Scilab code Exa 3.1 balance or conservation concept

```
clc;
warning("off");
printf("\n\n example3.1 - pg65");
// given
a=0.0006;  //[m^2] - area
l=0.1;  //[m] - length
// (a) using the fourier law
deltax=0.1;  //[m] - thickness of copper block
T2=100;  //[degC] - temp on one side of copper block
T1=0;  //[degC] - temp on other side of the copper block
k=380;  //[W/mK] - thermal conductivity
// using the formula (q/A)*deltax=-k*(T2-T1)
g=-k*(T2-T1)/deltax;
printf("\n\n (a) The steady state heat flux across the copper block is\n q/A=%5eJ*m^-2*sec^-1 ",g);
// (b)
V=a*l; //[m^3] - volume
// using the overall balance equation with the
```

```
accumulation and generation term
Qgen=1.5*10^6;   //[j*m^-3*sec^-1]
SIx=(g*a-Qgen*V)/a;
printf("\n\n (b) the flux at face 1 is %5ej*m^-2*sec^-1;\nthe negative sign indicates taht the heat flux is from right to left(negative x direction",SIx);
```

Scilab code Exa 3.2 the balance equation in differential form

```
clc;
warning('off');
printf("\n\n example3.2 - pg68");
// given
syms x;
SIx2=-3.8*10^5;   //[j*m^-2*sec^-1] - flux at x=0.1,i.e through face2
Qgen=1.5*10^6;   //[j*m^-3*sec^-1] - uniform generation in the volume
T2=100+273.15;   //[K] temperature at face 2
x2=0.1;   //[m]
k=380;   //[W/mK] - thermal conductivity
// using the equation der(SIx)*x=SIx+c1;where c1 is tyhe constant of integration
c1=(Qgen*x2)-SIx2;
disp(c1)
SIx=Qgen*x-c1;
disp(SIx,"SIx=");
printf("\n where SIx is in units of j m^-2 sec^-1 and x is in units of m");
// using the equation -k*T=der(SIx)*x^2-c1*x+c2;where c2 is the constant of integration
c2=-k*T2-(Qgen*(x2)^2)/2+c1*x2;
T=-(Qgen/k)*x^2+(c1/k)*x-(c2/k);
disp(T,"T=");
```

```
printf("\n where T is in units of kelvin(K)");
```

Scilab code Exa 3.3 the balance equation in differential form

```
clc;
warning("off");
printf("\n\n example3.3 - pg69");
// given
syms t x;
hf1=-270;   //[J/sec] - heat flow at face 1
hf2=-228;   //[J/sec] - heat flow at face2
Qgen=1.5*10^6;   //[J*m^-3*sec^-1] generation per unit volume per unit time
v=6*10^-5;   //[m^3] volume
Cp=0.093;   //[cal*g^-1*K^-1] heat capacity of copper
sp=8.91;   //specific gravity of copper
a=0.0006;   //[m^2] - area
// (a) using the overall balance
acc=hf1-hf2+Qgen*v;
printf("\n\n (a) the rate of accumulation is %fJ/sec\n\n ",acc);
// (b)
SIx1=hf1/a;
SIx2=hf2/a;
x1=0;
// solving for the constant of integration c1 in the equation [del(p*cp*T)/delt-der(SIx)]*x=-SIx+c1;
c1=0+SIx1;
x2=0.1;
g=(-(SIx2)+c1)/x2+Qgen;
SIx=c1-(g-Qgen)*x;
disp(SIx,"SI(x)=","(b)");
// solving for constant of integration c3 in the equation p*cp*T=g*t+c3
T2=100+273.15;
```

```
t2=0;
p=sp*10^3;  //[kg/m^3] - density
cp=Cp*4.1840;  //[J*kg^-1*K^-1]
c3=p*cp*T2-g*t2;
T=(g*(10^-3)/(p*cp))*t+c3/(p*cp);
disp(T,"T=");
// solving for constant of integration c2 in the
   equation -k*T=der(SIx)*x^2-c1*x+c2
k=380;  //[w/m^1*K^1]
x2=0.1;
c2=k*T+(3.5*10^5)*x2^2-(4.5*10^5)*x2;
function y=T(t,x)
    y=(-(3.5*10^5)*x^2+(4.5*10^5)*x+87.7*t
       +1.00297*10^5)/k;
endfunction
// at face 1;
x1=0;
t1=60;  //[sec]
T1=T(t1,x1);
disp(T1,"T=","at face 1");
// at face 2
x2=0.1;
t2=60;  // [sec]
T2=T(t2,x2);
disp(T2,"T=","at face 2");
```

Chapter 4

molecular transport and the general property balance

Scilab code Exa 4.1 variable area transport

```
clc;
warning('off');
printf("\n\n example4.1 - pg99");
// given
id=2.067;  //[in] - inside diameter
t=0.154;  //[in] - wall thickness
od=id+2*t;  //[in] - outer diameter
a=1.075;  //[in^2] - wall sectional area of metal
A=a*(1/144);  //[ft^2] - wall sectional area of metal in ft^2
deltaz=5/12;  //[ft] - length of transfer in z direction
T2=10+273.15;  //[K] - temperature at the top
T1=0+273.15;  //[K] - temperature at the bottom
q=-3.2;  //[Btu/hr] - heat transferred
deltaT=(T2-T1)+8;  //[degF]
k=-(q/A)/(deltaT/deltaz);
printf("\n\n korrect=%fBtu h^-1 ft^-1 degF^-1=17.17 W m^-1 K^-1",k);
```

```
Alm=(2*%pi*deltaz*((od-id)/(2*12)))/log(od/id);  //[ft^2] log mean area
disp(Alm)
kincorrect=k*(A/Alm);
printf("\n\n kincorrect=%fBtu h^-1 ft^-1 degF^-1=0.529 W m^-1 K^-1",kincorrect);
errorf=(k-kincorrect)/kincorrect;
disp(errorf,"error factor is-");
```

Scilab code Exa 4.2 variable area transport

```
clc;
warning('off');
printf("\n\n example4.2 - pg100");
// given
T1=0;  //[degC]
T2=10;  //[degC]
km=17.17;  //[W/m*K]
l=1;  //[m]
r2=1.1875;
r1=1.0335;
deltaT=T1-T2;
// using the formula Qr=-km*((2*pi*l)/ln(r2/r1))*deltaT;
Qr=-km*((2*%pi*l)/log(r2/r1))*deltaT;
printf("\n\n qr=%fW\n the plus sign indicates that the heat flow is radially out from the center",Qr);
```

Scilab code Exa 4.3 variable area transport

```
clc;
warning('off');
```

```
printf("\n\n example4.3 - pg100");
// given
km=9.92;   //[Btu/h*ft*degF]
Alm=0.242*(12/5);   //[ft^2]
T1=0;   //[degC]
T2=10;   //[degC]
deltaT=(T1-T2)*1.8;   //[degF]
r2=1.1875;
r1=1.0335;
deltar=(r2-r1)/12;   //[ft]
// using the formula Qr/Alm=-km*(deltaT/deltar)
Qr=(-km*(deltaT/deltar))*Alm;
printf("\n\n qr=%fBtu/h",Qr);
// in SI units
Alm=0.177;   //[m^2]
T1=0;   //[degC]
T2=10;   //[degC]
km=17.17;   //[W/m*K]
r2=1.1875;
r1=1.0335;
deltaT=T1-T2;
deltar=(r2-r1)*0.0254;   //[m]
// using the same formula
Qr=(-km*(deltaT/deltar))*Alm;
printf("\n\n qr=%fW",Qr);
```

Scilab code Exa 4.4 variable area transport

```
clc;
warning("off");
printf("\n\n example4.4 - pg101");
// given
x1=0;   //[cm]
x2=30;   //[cm]
p1=0.3;   //[atm]
```

```
p2=0.03;   //[atm]
D=0.164;   //[am^2/sec]
R=82.057;   //[cm^3*atm/mol*K]
T=298.15;   //[K]
// using the formula Nax*int(dx/Ax)=-(D/RT)*int(1*
   dpa)
a=integrate("1/((%pi/4)*(10-(x/6))^2)","x",x1,x2);
b=integrate("1","p",p1,p2);
Nax=-((D/(R*T))*b)/a;
printf("\n\n Nax=%6emol/sec=%3emol/h \n the plus
   sign indicates diffusion to the right",Nax,Nax
   *3600);
```

Scilab code Exa 4.5 heat or mass transport with constant generation

```
clc;
warning("off");
printf("\n\n example4.5 - pg105");
// given
syms r;
ro=0.5;  //[inch] - outside radius
ro=0.0127;  //[m] - outside radius in m
Tg=2*10^7;  //[J/m^3*sec] - heat generated by
   electric current
Tw=30;  //[degC] - outside surface temperature
km=17.3;  //[W/m*K] - mean conductivity
// using the formula T=Tw+(Tg/4*km)*(ro^2-r^2)
T=Tw+(Tg/(4*km))*(ro^2-r^2);
disp(T,"T=");
printf("\n where r is in meters and T is in degC");
function y=t(r)
    y=Tw+(Tg/(4*km))*(ro^2-r^2);
endfunction
printf("\n\n at the centre line (r=0),the maximum
   temperature is %fdegC.At the outside ,the
```

```
    temperature reduces to the boundary condition
    value of %fdegC.The distribution i parabolic
    between these 2 limits",t(0),t(0.0127));
```

Scilab code Exa 4.7 laminar flow in a tube

```
clc;
warning("off");
printf("\n\n example4.7 - pg119");
// given
r=10^-3;  //[m] - radius
l=1;  //[m] - length
Q=10^-7;  //[m^3/s] - flow rate
deltap=-10^6;  //[N/m^2=Pa] - pressure difference
spg=1.1;
pwater=1000;  //[kg/m^3] - density of water at 4degC
pfluid=spg*pwater;
mu=(r*-(deltap)*(%pi*r^3))/((4*Q)*(2*l));
mupoise=mu*10;
mucentipoise=mupoise*100;
printf("\n\n mu=%fNsM^-2=%fpoise=%fcP",mu,mupoise,
    mucentipoise);
```

Chapter 5

transport with a net convective flux

Scilab code Exa 5.9 mass fluxes in stationary and convected coordinates

```
clc;
warning("off");
printf("\n\n example5.9 - pg166");
// given
v=1;  //[cm/sec] - volume velocity or bulk velocity
vol=1;  //[cm^3] - volume
na=2;  // moles of a
nb=3;  // moles of b
nc=4;  // moles of c
mma=2;  //molecular weight of a
mmb=3;  //molecular weight of b
mmc=4;  //molecular weight of c
ma=na*mma;  //[g] weight of a
mb=nb*mmb;  //[g] weight of b
mc=nc*mmc;  //[g] weight of c
NabyA=2+2;  //[mol/cm^2*s] - molar flux = diffusing
   flux +convected flux
NbbyA=-1+3;  //[mol/cm^2*s] - molar flux = diffusing
    flux +convected flux
```

```
NcbyA=0+4;  //[mol/cm^2*s] - molar flux = diffusing flux +convected flux
NtbyA=NabyA+NbbyA+NcbyA;  //[mol/cm^2*s] - total molar flux
// on a mass basis, these corresponds to
nabyA=4+4;  //[g/cm^2*s]; - mass flux = diffusing flux +convected flux
nbbyA=-3+9;  //[g/cm^2*s]; - mass flux = diffusing flux +convected flux
ncbyA=0+16;  //[g/cm^2*s]; - mass flux = diffusing flux +convected flux
ntbyA=nabyA+nbbyA+ncbyA;  //[g/cm^2*s] - total mass flux
// concentrations are expressed in molar basis
CA=na/vol;  //[mol/cm^3]
CB=nb/vol;  //[mol/cm^3]
CC=nc/vol;  //[mol/cm^3]
CT=CA+CB+CC;  //[mol/cm^3] - total concentration
// densities are on a mass basis
pa=ma/vol;  //[g/cm^3]
pb=mb/vol;  //[g/cm^3]
pc=mc/vol;  //[g/cm^3]
pt=pa+pb+pc;  //[g/cm^3]
Ua=NabyA/CA;  //[cm/sec];
Ub=NbbyA/CB;  //[cm/sec];
Uc=NcbyA/CC;  //[cm/sec];
// the same result will be obtained from dividing mass flux by density
Uz=(pa*Ua+pb*Ub+pc*Uc)/(pa+pb+pc);
printf("\n\n Uz=%fcm/sec",Uz);
Uzstar=(NtbyA/CT);
printf("\n\n Uz*=%fcm/sec",Uzstar);
printf("\n\n for this example both Uz and Uz* are slightly greater than the volume velocity of 1cm/sec, because there is a net molar and mass diffusion in the positive direction.");
```

Scilab code Exa 5.10 total flux and ficks law

```
clc;
warning("off");
printf("\n\n example5.10 - pg171");
// given (from example 5.9)
na=2;  // moles of a
nb=3;  // moles of b
nc=4;  // moles of c
mma=2;  //molecular weight of a
mmb=3;  //molecular weight of b
mmc=4;  //molecular weight of c
ma=na*mma;  //[g] weight of a
mb=nb*mmb;  //[g] weight of b
mc=nc*mmc;  //[g] weight of c
NabyA=2+2;  //[mol/cm^2*s] - molar flux = diffusing flux +convected flux
NbbyA=-1+3;  //[mol/cm^2*s] - molar flux = diffusing flux +convected flux
NcbyA=0+4;  //[mol/cm^2*s] - molar flux = diffusing flux +convected flux
NtbyA=NabyA+NbbyA+NcbyA;  //[mol/cm^2*s] - total molar flux
// on a mass basis,these corresponds to
nabyA=4+4;  //[g/cm^2*s]; - mass flux = diffusing flux +convected flux
nbbyA=-3+9;  //[g/cm^2*s]; - mass flux = diffusing flux +convected flux
ncbyA=0+16;  //[g/cm^2*s]; - mass flux = diffusing flux +convected flux
// concentrations are expressed in molar basis
CA=na/vol;  //[mol/cm^3]
CB=nb/vol;  //[mol/cm^3]
CC=nc/vol;  //[mol/cm^3]
```

```
CT=CA+CB+CC;  //[mol/cm^3] - total concentration
// densities are on a mass basis
pa=ma/vol;  //[g/cm^3]
pb=mb/vol;  //[g/cm^3]
pc=mc/vol;  //[g/cm^3]
Ua=NabyA/CA;  //[cm/sec];
Ub=NbbyA/CB;  //[cm/sec];
Uc=NcbyA/CC;  //[cm/sec];
U=(pa*Ua+pb*Ub+pc*Uc)/(pa+pb+pc);
Ustar=(NtbyA/CT);
// the fluxes relative to mass average velocities
    are found as follows
JabyA=CA*(Ua-U);  //[mol/cm^2*sec]
JbbyA=CB*(Ub-U);  //[mol/cm^2*sec]
JcbyA=CC*(Uc-U);  //[mol/cm^2*sec]
printf("\n\n fluxes relative to mass average
    velocities are-");
printf("\n\n Ja/A=%fmol/cm^2*sec",JabyA);
printf("\n Jb/A=%fmol/cm^2*sec",JbbyA);
printf("\n Jc/A=%fmol/cm^2*sec",JcbyA);
jabyA=pa*(Ua-U);  //[g/cm^2*sec]
jbbyA=pb*(Ub-U);  //[g/cm^2*sec]
jcbyA=pc*(Uc-U);  //[g/cm^2*sec]
printf("\n\n ja/A=%fg/cm^2*sec",jabyA);
printf("\n jb/A=%fg/cm^2*sec",jbbyA);
printf("\n jc/A=%fg/cm^2*sec",jcbyA);
// the fluxes relative to molar average velocity are
    found as follows
JastarbyA=CA*(Ua-Ustar);  //[mol/cm^2*sec]
JbstarbyA=CB*(Ub-Ustar);  //[mol/cm^2*sec]
JcstarbyA=CC*(Uc-Ustar);  //[mol/cm^2*sec]
printf("\n\n fluxes relative to molar average
    velocities are-");
printf("\n\n Ja*/A=%fmol/cm^2*sec",JastarbyA);
printf("\n Jb*/A=%fmol/cm^2*sec",JbstarbyA);
printf("\n Jc*/A=%fmol/cm^2*sec",JcstarbyA);
jastarbyA=pa*(Ua-Ustar);  //[g/cm^2*sec]
jbstarbyA=pb*(Ub-Ustar);  //[g/cm^2*sec]
```

```
jcstarbyA=pc*(Uc-Ustar);   //[g/cm^2*sec]
printf("\n\n ja*/A=%fg/cm^2*sec",jastarbyA);
printf("\n jb*/A=%fg/cm^2*sec",jbstarbyA);
printf("\n jc*/A=%fg/cm^2*sec",jcstarbyA);
```

Scilab code Exa 5.11 binary mass diffusion in gases

```
clc;
warning("off");
printf("\n\n example5.11 - pg176");
// given
T=0+273.15;   //[K] - temperature in Kelvins
pa2=1.5;   //[atm] - partial presuure of a at point2
pa1=0.5;   //[atm] - partial pressure of a at point 1
z2=20;   //[cm] - position of point 2 from reference point
z1=0;   //[cm] - position of point1 from reference point
p=2;   //[atm] - total pressure
d=1;   //[cm] - diameter
D=0.275;   //[cm^2/sec] - diffusion coefficient
A=(%pi*((d)^2))/4;
R=0.082057;   //[atm*m^3*kmol^-1*K^-1] - gas constant
// (a) using the formula Na/A=-(D/(R*T))*((pa2-pa1)/(z2-z1))
Na=(-(D/(R*T))*((pa2-pa1)/(z2-z1)))*(A)/(10^6);
printf("\n\n Na=%ekmol/sec\n The negative sign indicates diffusion from point 2 to point 1",Na);
pb2=p-pa2;
pb1=p-pa1;
// (b) using the formula Na/A=((D*p)/(R*T*(z2-z1)))*ln(pb2/pb1)
Na=(((D*p)/(R*T*(z2-z1)))*log(pb2/pb1))*(A)/(10^6);
printf("\n\n Na=%ekmol/sec",Na);
printf("\n The induced velocity increases the net
```

```
    transport of A by the ratio of 10.6*10^-10 to
    4.82*10^-10 or 2.2 times.This increse is
    equivalent to 120 percent");
```

Scilab code Exa 5.12 binary mass diffusion in gases

```
clc;
warning("off");
printf("\n\n example5.12 - pg178");
// given
T=0+273.15;  //[K] - temperature in Kelvins
pa2=1.5;  //[atm] - partial presuure of a at point2
pa1=0.5;  //[atm] - partial pressure of a at point 1
z2=20;  //[cm] - position of point 2 from reference
    point
z1=0;  //[cm] - position of point1 from reference
    point
p=2;  //[atm] - total pressure
d=1;  //[cm] - diameter
D=0.275;  //[cm^2/sec] - diffusion coefficient
A=(%pi*((d)^2))/4;
R=0.082057;  //[atm*m^3*kmol^-1*K^-1] - gas constant
k=0.75;
// using the formula (Na/A)=-(D/(R*T*(z2-z1)))*ln
    ((1-(pa2/p)*(1-k))/(1-(pa1/p)*(1-k)))
NabyA=-(D/(R*T*(z2-z1)))*(2*0.7854)*log((1-(pa2/p)
    *(1-k))/(1-(pa1/p)*(1-k)))/(10^6);
printf("\n\n (Na/A)=%ekmol/sec",NabyA);
printf("\n Note that this answer is larger than the
    rate for equimolar counter diffusion but smaller
    tahn the rate for diffusion through a stagnant
    film.Sometimes the rate for diffusin through a
    stagnant film can be considered as an upper bound
    , if k ties between zero and one");
```

Scilab code Exa 5.13 diffusion due to pressure gradient

```
clc;
warning("off");
printf("\n\n example5.13 - pg184");
// given
l=4;  //[m] - length of the tube
id=1.6*10^-3;  //[m] - inside diameter
Nkn=10;  // - knudsen no.
Ma=92;  // - molecular weight of gas
mu=6.5*10^-4;  //[kg/m*sec] - viscosity
T=300;  //[K] - temperature
R=8314;  //[kPa*m^3*kmol^-1*K^-1] - gas constant
lambdaA=Nkn*id;  //[m] mean free path
// for calculating pressure using the formula lamdaA
    =32*(mu/p)*((R*T)/(2*pi*Ma))^(1/2)
p=32*(mu/lambdaA)*((R*T)/(2*%pi*Ma))^(1/2);
patm=p/(1.01325*10^5);
printf("\n\n p=%fkg/m*sec^2=%fPa=%eatm",p,p,patm);
printf("\n The value of 10 for the knudsen number is
     on the border between Knudsen diffusion and
    transition flow");
```

Chapter 6

flow turbulence

Scilab code Exa 6.1 the reynolds experiment

```
clc;
warning("off");
printf("\n\n example6.1 - pg200");
// given
q=50;  //[gal/min] - volumetric flow rate
d=2.067/12;  //[ft] - diameter
A=0.02330;  //[ft^2] - flow area
p=0.99568*62.43;  //[lb/ft^3] - density of water at
   86degF
mu=0.8007*6.72*10^-4;  //[lb/ft*sec] - viscosity of
   water at 86degF
u=q/(60*7.48*A);
// using the formula Nre=d*u*p/mu;
Nre=(d*u*p)/mu;
disp(Nre,"Nre=");
printf("\n Hence the flow is turbulent.Note also
   that Nre is dimensionless");
```

Scilab code Exa 6.2 transitional flow

```
clc;
warning("off");
printf("\n\n example6.2 - pg202");
// given
p=0.99568*62.43;  //[lb/ft^3] - density of water at 86degF
mu=0.8007*6.72*10^-4;  //[lb/ft*sec] - viscosity of water at 86degF
u=4.78;  //[ft/sec] - free stream velocity
Nre=5*10^5;  // the lower limit for the transition reynolds number range is substituted
x=(Nre*mu)/(p*u);
disp(x,"x");
printf("\nThus the transition could star at about %fft.The reynolds number at the upper end of the transition range is %e.The value of x at this location is ten times then the value obtained above i.e %fft",x,Nre*10,x*10);
```

Scilab code Exa 6.3 the equations for transport under turbulent conditions

```
clc;
warning("off");
printf("\n\n example6.3 - pg212");
// given
t=[0 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.10 0.11 0.12];
Ux=[3.84 3.50 3.80 3.60 4.20 4.00 3.00 3.20 3.40 3.00 3.50 4.30 3.80];
Uy=[0.43 0.21 0.18 0.30 0.36 0.28 0.35 0.27 0.21 0.22 0.23 0.36 0.35];
Uz=[0.19 0.16 0.17 0.13 0.09 0.10 0.16 0.15 0.13 0.18 0.17 0.18 0.17];
// using the formula AREA=(deltat/2)*(U1+U13+2*(U2+
```

```
   U3+U4+U5+U6+U7+U8+U9+U10+U11+U12))
// for Uxmean
deltat=0.01;
T=t(13)-t(1);
AREA=(deltat/2)*(Ux(1)+Ux(13)+2*(Ux(2)+Ux(3)+Ux(4)+
   Ux(5)+Ux(6)+Ux(7)+Ux(8)+Ux(9)+Ux(10)+Ux(11)+Ux
   (12)));
Uxmean=AREA/T;
disp(Uxmean,"Uxmean=");
// for Uymean
deltat=0.01;
T=t(13)-t(1);
AREA=(deltat/2)*(Uy(1)+Uy(13)+2*(Uy(2)+Uy(3)+Uy(4)+
   Uy(5)+Uy(6)+Uy(7)+Uy(8)+Uy(9)+Uy(10)+Uy(11)+Uy
   (12)));
Uymean=AREA/T;
disp(Uymean,"Uymean=");
// for Uzmean
deltat=0.01;
T=t(13)-t(1);
AREA=(deltat/2)*(Uz(1)+Uz(13)+2*(Uz(2)+Uz(3)+Uz(4)+
   Uz(5)+Uz(6)+Uz(7)+Uz(8)+Uz(9)+Uz(10)+Uz(11)+Uz
   (12)));
Uzmean=AREA/T;
disp(Uzmean,"Uzmean=");
U=(Uxmean^2+Uymean^2+Uzmean^2)^(1/2);
disp(U,"U=");
```

Scilab code Exa 6.5 the prandtl mixing theory

```
clc;
warning('off');
printf("\n\n example6.5 - pg232");
// given
UzmaxbyU=24.83;
```

```
roUbyv=2312;
Re=100000;
// using the formula Et/v=95.5*((r/ro)/slope)-1
// from fig 6.6 at Re=100000
rbyro=[0 0.040 0.100 0.200 0.300 0.4 0.5 0.6 0.7 0.8
    0.9 0.960 1];
slope=[0 0.105 0.112 0.126 0.144 0.168 0.201 0.252
   0.336 0.503 1.007 2.517 94.59];
for i=2:13
    Etbyv(i)=95.5*((rbyro(i))/slope(i))-1;
end
clf;
xtitle("eddy viscosity ratio versus dimensionless
   radius","r/ro","Et/v");
plot(rbyro,Etbyv);
```

Scilab code Exa 6.9 friction factor

```
clc;
warning("off");
printf("\n\n example6.9 - pg258");
// given
spg=0.84;
p=0.84*62.4;  //[lbf/ft^3] - density
dP=80*144;  //[lbf/ft^2] - pressure
dz=2000;  //[ft] - length of pipe
gc=32.174;  //[(lbm*ft)/(lbf*sec^2)] - gravitational
    conversion constant
dpbydz=-dP/dz;
do=2.067/12;  //[ft]
U=2000*(1/24)*(1/3600)*(42)*(1/7.48)*(1/0.02330);
// using the formula f=((do/2)*(-dp/dz)*gc)/(p*(U)
   ^2)
f=((do/2)*(-dpbydz)*gc)/(p*(U)^2)
disp(f,"f=");
```

Chapter 7

integral methods of analysis

Scilab code Exa 7.2 the integral mass balance

```
clc;
warning("off");
printf("\n\n example7.2 - pg273");
// given
id=4;  //[m] - inside diameter
h=2;  //[m] - water level
ro=0.03;  //[m] - radius of exit hole
rt=id/2;  //[m] - inside radius
g=9.80665;  //[m/sec^2] - gravitational acceleration
// using the equation dh/h^(1/2)=-((ro^2)/(rt^2))*(2*g)^(1/2)dt and integrating between h=2 and h=1
t1=integrate('(1/h^(1/2))*(1/(-((ro^2)/(rt^2))*(2*g)^(1/2)))','h',2,1);
printf("\n\n Time required to remove half of the contents of the tank is \n t=%fsec=%fmin",t1,t1/60);
//integrating between h=2 and h=0
t2=integrate('(1/h^(1/2))*(1/(-((ro^2)/(rt^2))*(2*g)^(1/2)))','h',2,0);
printf("\n\n Time required to empty the tank fully
```

```
is \n t=%fsec=%fmin",t2,t2/60);
```

Scilab code Exa 7.3 integral balance on an individual species

```
clc;
warning("off");
printf("\n\n example7.3 - pg274");
// given
// composition of fuel gas
nH2=24;
nN2=0.5;
nCO=5.9;
nH2S=1.5;
nC2H4=0.1;
nC2H6=1;
nCH4=64;
nCO2=3.0;
// calculating the theoritical amount of O2 required
nO2theoreq=12+2.95+2.25+0.30+3.50+128;
// since fuel gas is burned with 40% excess O2,then O2 required is
nO2req=1.4*nO2theoreq;
nair=nO2req/0.21;  // as amount of O2 in air is 21%
nN2air=nair*(0.79);  // as amount of N2 in air is 79 %
nN2=nN2+nN2air;
nO2=nO2req-nO2theoreq;
nH2O=24+0+0.2+3.0+128;
nCO2formed=72.1;
nCO2=nCO2+nCO2formed;
nSO2=1.5;
ntotal=nSO2+nCO2+nO2+nN2+nH2O;
mpSO2=(nSO2/ntotal)*100;
mpCO2=(nCO2/ntotal)*100;
mpO2=(nO2/ntotal)*100;
```

```
mpN2=(nN2/ntotal)*100;
mpH2O=(nH2O/ntotal)*100;
printf("\n\n gas                                    N2                    O2                    H2O              CO2                    SO2");
printf("\n\n moles                                 %f       %f       %f       %f       %f",nN2,nO2,nH2O,nCO2,nSO2);
printf("\n\n mole percent                          %f       %f       %f       %f       %f",mpN2,mpO2,mpH2O,mpCO2,mpSO2);
```

Scilab code Exa 7.4 integral momentum balance

```
clc;
warning("off");
printf("\n\n example7.4 - pg280");
// given
id=6;  //[inch] - inlet diameter
od=4;  //[inch] - outlet diameter
Q=10;  //[ft^3/sec] - water flow rate
alpha2=%pi/3;  //[radians] - angle of reduction of elbow
alpha1=0;
p1=100;  //[psi] - absolute inlet pressure
p2=29;  //[psi] - absolute outlet pressure
S1=(%pi*((id/12)^2))/4;
S2=(%pi*((od/12)^2))/4;
U1=Q/S1;
U2=Q/S2;
mu=6.72*10^-4;  //[lb*ft^-1*sec^-1]
p=62.4;  //[lb/ft^3]
Nrei=((id/12)*U1*p)/(mu);
disp(Nrei,"Nre(inlet)=");
Nreo=((od/12)*U2*p)/(mu);
disp(Nreo,"Nre(outlet)=");
```

```
// thus
b=1;
w1=p*Q;  //[lb/sec] - mass flow rate
w2=w1;
gc=32.174;
// using the equation (w/gc)*((U1)*(cos(alpha1))-(U2)*(cos(alpha2)))+p1*S1*cos(alpha1)-p2*S2*cos(alpha2)+Fextx=0;
Fextx=-(w1/gc)*((U1)*(cos(alpha1))-(U2)*(cos(alpha2)))-p1*144*S1*cos(alpha1)+p2*144*S2*cos(alpha2);
disp(Fextx,"Fext,x=");
Fexty=-(w1/gc)*((U1)*(sin(alpha1))-(U2)*(sin(alpha2)))-p1*144*S1*sin(alpha1)+p2*144*S2*sin(alpha2);
disp(Fexty,"Fext,y=");
printf("\n\n the forces Fxt,x and Fext,y are the forces exerted on the fluid by the elbow.Fext,x acts to the left and Fext,y acts in the positive y direction.Note that the elbow is horizantal,and gravity acts in the z direction");
```

Scilab code Exa 7.5 integral momentum balance

```
clc;
warning("off");
printf("\n\n example7.5 - pg 282");
// given
Fextx=-2522;  //[lb] - force in x direction
Fexty=2240;  //[lb] - force in y direction
// the force exerted by the elbow on the fluid is the resolution of Fext,x and Fext,y , therefore
Fext=((Fextx)^2+(Fexty)^2)^(1/2);
alpha=180+(atan(Fexty/Fextx))*(180/%pi);
printf("\n\n the force has a magnitude of %flb and a direction of %f from the positive x direction(in the second quadrant",Fext,alpha);
```

Scilab code Exa 7.6 integral momentum balance

```
clc;
warning("off");
printf("\n\n example7.6 - pg283");
// given
id=6;  //[inch] - inlet diameter
od=4;  //[inch] - outlet diameter
Q=10;  //[ft^3/sec] - water flow rate
alpha2=%pi/3;  //[radians] - angle of reduction of elbow
alpha1=0;
p1=100;  //[psi] - absolute inlet pressure
p2=29;  //[psi] - absolute outlet pressure
patm=14.7;  //[psi] - atmospheric pressure
p1gauge=p1-patm;
p2gauge=p2-patm;
S1=(%pi*((id/12)^2))/4;
S2=(%pi*((od/12)^2))/4;
U1=Q/S1;
U2=Q/S2;
p=62.4;  //[lb/ft^3]
b=1;
w1=p*Q;  //[lb/sec] - mass flow rate
w2=w1;
gc=32.174;
// using the equation Fpress=p1gauge*S1-p2gauge*S2*cos(alpha2);
Fpressx=p1gauge*144*S1-p2gauge*144*S2*cos(alpha2);
Fpressy=p1gauge*144*S1*sin(alpha1)-p2gauge*144*S2*sin(alpha2);
wdeltaUx=(w1/gc)*((U2)*(cos(alpha2))-(U1)*(cos(alpha1)));
wdeltaUy=(w1/gc)*((U2)*(sin(alpha2))-(U1)*(sin(
```

```
    alpha1)));
Fextx=wdeltaUx-Fpressx;
Fexty=wdeltaUy-Fpressy;
Fext=((Fextx)^2+(Fexty)^2)^(1/2);
alpha=180+(atan(Fexty/Fextx))*(180/%pi);
printf("\n\n The force has a magnitude of %flb and a
     direction of %f from the positive x direction(in
     the second quadrant",Fext,alpha);
printf("\n\n Also there is a force on the elbow in
    the z direction owing to the weight of the elbow
    plus the weight of the fluid inside");
```

Scilab code Exa 7.7 integral energy balance

```
clc;
warning("off");
printf("\n\n example7.7 - pg293");
// given
Uo=1;   //[m/sec]
// using Ux/Uo=y/yo
// assuming any particular value of yo will not
    change the answer,therefore
yo=1;
Uxavg=integrate('(Uo*y)/yo','y',0,yo);
Ux3avg=integrate('((Uo*y)/yo)^3','y',0,yo);
// using the formula alpha=(Uxavg)^3/Ux3avg
alpha=(Uxavg)^3/Ux3avg;
disp(alpha,"alpha=");
printf("\n\n Note that the kinetic correction factor
      alpha has the same final value for laminar pipe
     flow as it has for laminar flow between parallel
     plates.");
```

Scilab code Exa 7.8 integral energy balance

```
clc;
warning("off");
printf("\n\n example7.8 - pg293");
// given
Q=0.03;  //[m^3/sec] - volumetric flow rate
id=7;  //[cm] - inside diameter
deltaz=-7;  //[m] - length of pipe
T1=25;  //[degC] - lowere side temperature
T2=45;  //[degC] - higher side temperature
g=9.81;  //[m/sec^2] - acceleration due to gravity
deltaP=4*10^4;  //[N/m^2] - pressure loss due to
    friction
p=1000;  //[kg/m^3] - density of water
w=Q*p;
C=4184;  //[J/kg*K) - heat capacity of water
deltaH=w*C*(T2-T1);
// using the formula Qh=deltaH+w*g*deltaz
Qh=deltaH+w*g*deltaz;
printf("\n\n the duty on heat exchanger is \n Q=%6eJ
    /sec",Qh);
```

Scilab code Exa 7.10 the energy equation and the engineering bernoulli equation

```
clc;
warning("off");
printf("\n\n example7.10 - pg298");
// given
d=0.03;  //[m] - diameter
g=9.784;  //[m/sec] - acceleration due to gravity
deltaz=-1;
// using the equation (1/2)*(U3^2/alpha3-U1^2/alpha1
    )+g*deltaz=0
```

```
// assuming
alpha1=1;
alpha3=1;
// also since the diameter of the tank far exceeds the diameter of the hose , the velocity at point 1 must be negligible when compared to the velocity at point 3
U1=0;
U3=(-2*g*deltaz+(U1^2)/alpha1)^(1/2);
p=1000;  //[kg/m^3] - density of water
S3=(%pi/4)*(d)^2
w=p*U3*S3;
printf("\n\n the mass flow rate is \n w=%fkg/sec",w);
// the minimum pressure in the siphon tube is at the point 2. Before the result of 3.13 kg/sec is accepted as the final value , the pressure at point 2 must be calcilated in order to see if the water might boil at this point
// using deltap=p*((U3^2)/2+g*deltaz)
deltap=p*((U3^2)/2+g*deltaz);
p1=1.01325*10^5;  //[N/m^2] - is equal to atmospheric pressure
p2=p1+deltap;
vp=0.02336*10^5;
if p2>vp then
    printf("\n\n the siphon can operate since the pressure at point 2 is greater than the value at which the liquid boils");
else
    printf("\n\n the siphon cant operate since the pressuer at point 2 is less than the value at which the liquid boils");

end
```

Scilab code Exa 7.11 the energy equation and the engineering bernoulli equation

```
clc;
warning("off");
printf("\n\n example7.11 - pg300");
// given
sp=1.45;  // specific gravity of trichloroethylene
pwater=62.4;  //[lb/ft^3] - density of water
p=sp*pwater;
d1=1.049;  //[inch] - density of pipe at point 1
d2=0.6;  //[inch] - density of pipe at point 2
d3=1.049;  //[inch] - density of pipe at point 3
// using the formula U1*S1=U2*S2; we get U1=U2*(d2/d1);
// then using the bernoulli equation deltap/p=(1/2)*(U2^2-U1^2);
deltap=4.2*(144);  //[lb/ft^2] - pressure difference
U2=((2*(deltap/p)*(1/(1-(d2/d1)^4)))^(1/2))*(32.174)^(1/2);
// using the formula w=p*U2*S
w=p*U2*((%pi/4)*(0.6/12)^2);
w1=w/(2.20462);
printf("\n\n the mass flow rate is \n w=%flb/sec\n or in SI units \n w=%fkg/sec",w,w1);
```

Scilab code Exa 7.12 the mechanical energy equation and the engineering bernoulli equation

```
clc;
warning("off");
printf("\n\n example7.12 - pg301");
```

```
// given
Q=50/(7.48*60);  //[ft/sec] - volumetric flow rate of water
d1=1;  //[inch] - diameter of pipe
deltaz=-5;  //[ft] - distance between end of pipe and tank
g=32.1;  //[ft/sec] - acceleration due to gravity
Cp=1;  //[Btu/lb*F] - heat capacity of water
p=62.4;  //[lb/ft^3] - density of water
S1=(%pi/4)*(d1/12)^2;
U1=Q/S1;
w=p*Q;
U2=0;
gc=32.174;
// using the formula deltaH=(w/2)*((U2)^2-(U1)^2)+w*g*deltaz
deltaH=-(w/(2*gc))*((U2)^2-(U1)^2)-w*(g/gc)*deltaz;
disp(deltaH);
deltaH=deltaH/778;  // converting from ftlb/sec to Btu/sec
deltaT=deltaH/(w*Cp);
printf("\n\n The rise in temperature is %fdegF",deltaT);
```

Scilab code Exa 7.13 the mechanical energy equation and the engineering bernoulli equation

```
clc;
warning("off");
printf("\n\n example7.13 - pg303");
// given
deltaz=30;  //[ft] - distance between process and the holding tank
Q=100;  //[gpm] - volumetric flow rate of water
p1=100;  //[psig]
```

```
p2=0;  //[psig]
g=32.1;  //[ft/sec] - acceleration due to gravity
sv=0.0161;  //[ft^3/lb] - specific volume of water
p=1/sv;  //[lb/ft^3] - density of water
e=0.77;  // efficiency of centrifugal pump
deltap=(p1-p2)*(144);  //[lbf/ft^2]
gc=32.174;
// using the equation deltap/p+g*(deltaz)+Ws=0;
Wst=-deltap/p-(g/gc)*(deltaz);
// using the formula for efficiency e=Ws(theoritical
   )/Ws(actual)
// therefore
Wsa=Wst/e;
// the calulated shaft work is for a unit mass flow
   rate of water,therfore for given flow rate
   multiply it by the flow rate
w=(Q*p)/(7.48*60);
Wsactual=Wsa*w;
power=-Wsactual/(778*0.7070);
printf("\n\n the required horsepower is %fhp",power)
   ;
```

Scilab code Exa 7.14 the mechanical energy equation and the engineering bernoulli equation

```
clc;
warning("off");
printf("\n\n example7.14 - pg304");
// given
p1=5;  //[atm] - initial pressure
p2=0.75;  //[atm] - final pressure after expansion
   through turbine
T=450;  //[K] - temperature
y=1.4;  // cp/cv for nitrogen
// using the equation Ws=-(y/(y-1))*(p1/density1)*((
```

```
   p2/p1)^((y-1)/y)-1)
R=8314;  // gas constant
p1bydensity=R*T;
Ws=-(y/(y-1))*(p1bydensity)*((p2/p1)^((y-1)/y)-1);
printf("\n\n the shaft work of the gas as it expands
    through the turbine and transmits its molecular
   energy to the rotating blades is \n Ws=%eJ/kmol",
   Ws);
```

Scilab code Exa 7.15 manometers

```
clc;
warning("off");
printf("\n\n example 7.15 - pg311");
// given
T=273.15+25;   //[K] - temperature
R=8.314;   //[kPa*m^3/kmol*K] - gas constant
p=101.325;   //[kPa] - pressure
M=29;   // molecular weight of gas
pa=(p*M)/(R*T);
sg=13.45;   // specific gravity
pm=sg*1000;
g=9.807;   //[m/sec^2] - acceleration due to gravity
deltaz=15/100;   //[m]
// using the equation p2-p1=deltap=(pm-pa)*g*deltaz
deltap=-(pm-pa)*g*deltaz;
printf("\n\n the pressure drop is %eN/m^2",deltap);
printf("\n the minus sign means the upstream
   pressure p1 is greater than p2, i.e ther is a
   pressure drop.");
```

Scilab code Exa 7.16 manometers

```
clc;
warning("off");
printf("\n\n example7.16 - pg312");
// given
T=536.67;   //[degR]; - temperature
R=10.73;   //[(lbf/in^2*ft^3)*lb*mol^-1*degR] - gas constant
p=14.696;   //[lbf/in^2];
g=9.807*3.2808;   //[ft/sec^2] - acceleration due to gravity
M=29;   // molecular weight of gas
pa=(p*M)/(R*T);
sg=13.45;   // specific gravity
pm=sg*62.4;
deltaz=15/(2.54*12);   //[ft]
gc=32.174;
// using the equation p2-p1=deltap=(pm-pa)*g*deltaz
deltap=(pm-pa)*(g/gc)*deltaz;
printf("\n\n the pressure drop is %flbf/ft^2",deltap);
```

Scilab code Exa 7.18 manometers

```
clc;
warning("off");
printf("\n\n example7.18 - pg315");
// given
at=0.049;   //[in^2] - cross sectional area of the manometer tubing
aw=15.5;   //[in^2] - cross sectional area of the well
g=32.174;   //[ft/sec^2] - acceleration due to gravity
gc=32.174;
sg=13.45;   //[ specific garvity of mercury
```

```
p=62.4;  //[lb/ft^3] - density of water;
pm=sg*p;
deltaz_waterleg=45.2213;
// using the equation A(well)*deltaz(well)=A(tube)*
   deltaz(tube)
deltazt=70;  //[cm]
deltazw=deltazt*(at/aw);
deltaz=deltazt+deltazw;
deltap_Hg=-pm*(g/gc)*(deltaz/(2.54*12));
disp(deltap_Hg);
deltazw=45.2213;  //[cm]
deltap_tap=deltap_Hg+p*(g/gc)*(deltazw/(12*2.54));
printf("\n\n deltap_tap=%f lbf/ft^2",deltap_tap);
printf("\ndeltap is negative and therefore p1 is
   greater than p2");
```

Scilab code Exa 7.19 buoyant forces

```
clc;
warning("off");
printf("\n\n example7_19 - pg317");
// given
p=749/760;  //[atm]
T=21+273.15;  //[K]
R=82.06;  //[atm*cm^3/K] - gas constant
v=(R*T)/p;  //[cm^3/mole] - molar volume
M=29;  //[g/mole] - molecular weight
pair=M/v;
m_air=53.32;  //[g]
m_h2o=50.22;  //[g]
ph2o=0.998;  //[g/cm^3] - density of water
V=(m_air-m_h2o)/(ph2o-pair);  //[cm^3]
density=m_air/V;
printf("\n\n The density of coin is \n density=%fg/
   cm^3",density);
```

```
printf("\n\n Consulting a handbook it is seen that
    this result is correct density for gold");
```

Scilab code Exa 7.20 buoyant forces

```
clc;
warning("off");
printf("\n\n example7.20 - pg318");
// given
P=749/760;  //[atm] - pressure
T=21+273.15;  //[K] - temperature
poak=38*(1/62.4);  //[g/cm^3] - density of oak
pbrass=534/62.4;  //[g/cm^3] - density of brass
m_brass=6.7348;  //[g]
pair=0.001184;  //[g/cm^3] - density of air
// using the formula m_oak=m_brass*((1-(pair/pbrass)
    )/(1-(pair/poak)))
m_oak=m_brass*((1-(pair/pbrass))/(1-(pair/poak)));
printf("\n\n m_oak=%fg",m_oak);
```

Scilab code Exa 7.21 variation of pressure with depth

```
clc;
warning("off");
printf("\n\n example7.21 - pg320");
// given
T=545.67;  //[degR] - temperature
R=1545;  //[Torr*ft^3/degR*mole] - gas constant
M=29;  //[g/mole] - molecular weight
g=9.807;  //[m/sec^2] - acceleration due to gravity
gc=9.807;
po=760;  //[Torr] - pressure
deltaz=50;  //[ft]
```

```
// using the equation p=po*exp(-(g/gc)*M*(deltaz/R*T))
p=po*%e^(-(g/gc)*M*(deltaz/(R*T)));
printf("\n\n p=%fTorr\n Thus, the pressure decrease for an elevation of 50ft is very small",p);
```

Scilab code Exa 7.22 variation of pressure with depth

```
clc;
warning("off");
printf("\n\n example7.22 - pg321");
// given
To=545.67;   //[degR] - air temperature at beach level
betaa=-0.00357;   //[degR/ft] - constant
R=1545;   //[Torr*ft^3/degR*mole] - gas constant
M=29;
deltaz=25000;   //[ft]
// using the equation ln(p/po)=((M)/(R*betaa))*ln(To/(To+betaa*deltaz)
p=po*exp(((M)/(R*betaa))*log(To/(To+betaa*deltaz)));
printf("\n\n p=%fTorr",p);
// using the equation T=To+betaa*deltaz
T=To+betaa*deltaz;
printf("\n\n T=%fdegR",T);
```

Chapter 9

agitation

Scilab code Exa 9.3 scale up procedures for turbulent flow with a single test volume

```
clc;
warning("off");
printf("\n\n example9.3 - pg389");
Nblades=4;  // no. of blades
d=9/12;  //[ft] - diameter of the impeller
dt=30/12;  //[ft] - diameter of the tank
Nbaffles=4;  //  no. of baffles
h=30;  // [inch]  - height of unit
mu=10;  //[cP] - viscosity of fluid
sg=1.1;  // specific gravity of fluid
s=300;  //[rpm] - speed of agitator
CbyT=0.3;
V=(%pi*dt^3)/4; //volume of tank in ft^3
V1=V*7.48;  //[gal] - volume of tank in gallons
mu=mu*(6.72*10^-4);  //[lb/ft*sec]
p=sg*62.4;  //[lb/ft^3] - density of fluid
N=s/60;  //[rps] - impeller speed in revolutions per
    second
Nre=((d^2)*N*p)/mu;
disp(Nre,"Nre=");
```

```
printf("\n\n Therefore the agitator operates in the turbulent region");
Npo=1.62;
gc=32.174;
P=(Npo*(p*(N^3)*(d^5)))/(gc*550);
Cf=63025;
Tq=(P/s)*Cf;
PbyV=P/V;
PbyV1=P/V1;
TqbyV=Tq/V;
TqbyV1=Tq/V1;
printf("\n\n The power per unit volume and the torque per unit volume is \n P/V=%f hp/ft^3=%f hp/gal\n Tq/V=%f in*lb/ft^3=%f in*lb/gal",PbyV,PbyV1,TqbyV,TqbyV1);
```

Scilab code Exa 9.4 scale up procedures for turbulent flow with a single test volume

```
clc;
warning("off");
printf("\n\n example9.4 - pg391");
// given
Tpilot=30;
Tlab=10;
N1=690;
N2=271;
D2=3;
D1=1;
n=(log(N1/N2))/(log(D2/D1));
V=12000/7.48;   //[ft^3]
T=((4*V)/%pi)^(1/3);   //[ft]
R=12.69/(30/12);
N3=N2*(1/R)^n;   //[rpm] - impeller speed in the reactor
```

```
disp(N3,"impeller speed in rpm=");
D3=0.75*R;  //[ft] - reactor impeller diameter
disp(D3,"reactor impeller diameter in ft=");
P=0.1374*((N3/N2)^3)*(R^5);
disp(P,"power in hp=");
Cf=63025;
Tq=(P/N3)*Cf;  //[inch*lb]
disp(Tq,"torque in inch*lb=");
printf("\n\n At this point, the design is complete. A standard size impeller would be chosen as well as a standard size motor(7.5 hp or 10 hp)");
```

Scilab code Exa 9.5 scale up procedures for turbulent flow with a single test volume

```
clc;
warning("off");
printf("\n\n example9.5 - pg 393");
// given
n=[0.5 0.6 0.7 0.8 0.9 1.0];
D2=3.806;
D1=0.25;
R=D2/D1;
N1=690;
N2=N1*((D1/D2)^n);
P1=9.33*10^-3;  //[hp]
P2=P1*R^(5-3*n);
printf("\n\n n                              N,rpm                     P,hp");
for i=1:6
printf("\n %f                    %f                  %f",n(i),N2(i),P2(i));
end
```

Chapter 10

transport in ducts

Scilab code Exa 10.1 laminar pipe flow

```
clc;
warning("off");
printf("\n\n example10.1 - pg405");
T=30;  //[degC] - temperature
d=8.265*10^-4;  //[m] - diameter of the capillary viscometer
deltapbyL=-0.9364;  //[psi/ft] - pressure drop per unit length
deltapbyL=deltapbyL*(2.2631*10^4);  //[kg/m^2*sec^2] - pressure drop per unit length
Q=28.36*(10^-6)*(1/60);
p=(0.88412-(0.92248*10^-3)*T)*10^3;  //[kg/m^3] - density
s=(%pi*(d^2))/4;
U=Q/s;
tauw=(d/4)*(-deltapbyL);
shearrate=(8*U)/d;
mu=tauw/(shearrate);
printf("\n\n The viscosity is \n mu=%f kg/m*sec=%f cP",mu,mu*10^3);
printf("\n\n Finally, it is important to check the
```

```
reynolds number to make sure the above equation applies");
Nre=(d*U*p)/(mu);
disp(Nre,"Nre=");
printf("\n\n The flow is well within the laminar region and therefore the above equation applies");
```

Scilab code Exa 10.2 turbulent pipe flow

```
clc;
warning("off");
printf("\n\n example10.2 - pg407");
Nreold=1214;
Uold=0.8810;
Nre=13700;
U=Uold*(Nre/Nreold);
Lbyd=744;
// using the newton raphson method to calculate the value of f from the equation - 1/(f^(1/2))=4*log(Nre*(f^(1/2)))-0.4
f=0.007119;
p=(0.88412-(0.92248*10^-3)*T)*10^3;  //[kg/m^3] - density
tauw=(1/2)*p*(U^2)*f;
deltap=tauw*(4)*(Lbyd);
d=0.03254/12;  //[ft]
L=Lbyd*d;
printf("\n\n Pressure drop is \n -deltap=%e N/m^2=%f kpa=130 psi",deltap,deltap*10^-3);
printf("\n\n A pressure drop of 130 psi on a tube of length of %f ft is high and shows the impracticality of flows at high reynolds number in smaller tubes",L);
```

Scilab code Exa 10.3 pressure drop in rough pipes

```
clc;
warning("off");
printf("\n\n example10.3 - pg414");
// given
u=1/60;  //[m/sec] - velocity
p=1000;  //[kg/m^3] - density
mu=1*10^-3;  //[kg/m*sec] - viscosity
d=6*10^-2;  //[m] - inside diameter of tube
L=300;  //[m] - length of the tube
Nre=(d*u*p)/(mu);
disp("therefore the flow is laminar",Nre,"Nre=");
f=16/Nre;
disp(f);
deltap=(4*f)*(L/d)*((p*(u^2))/2);
printf("\n\n -deltap=%f N/m^2 = %f kPa = %e psi",
   deltap,deltap*10^-3,deltap*1.453*10^-4);
```

Scilab code Exa 10.4 pressure drop in rough pipes

```
clc;
warning("off");
printf("\n\n example10.4 - pg415");
// given
d=6*10^-2;  //[m] - inside diameter of tube
p=1000;  //[kg/m^3] - density
// for smooth pipe
Nre=[10^4 10^5];
f=[0.0076 0.0045];
mu=10^-3;  //[kg/m^2*s]
U=(Nre*mu)/(d*p);
```

```
L=300;  //[m] - length of the tube
for i=1:2
deltap(i)=(4*f(i))*(L/d)*((p*(U(i)^2))/2);
end
disp("for smooth pipe");
printf(" Nre                              -deltap");
printf("\n %f                         %f",Nre(1),deltap(1));
printf("\n %f                       %f \n",Nre(2),deltap(2));
// for commercial steel
Nre=[10^4 10^5];
f=[0.008 0.0053];
U=(Nre*mu)/(d*p);
L=300;  //[m] - length of the tube
for i=1:2
deltap(i)=(4*f(i))*(L/d)*((p*(U(i)^2))/2);
end
disp("for commercial steel pipe");
printf(" Nre                              -deltap");
printf("\n %f                         %f",Nre(1),deltap(1));
printf("\n %f                       %f \n",Nre(2),deltap(2));
// for cast iron pipe
Nre=[10^4 10^5];
f=[0.009 0.0073];
U=(Nre*mu)/(d*p);
L=300;  //[m] - length of the tube
for i=1:2
deltap(i)=(4*f(i))*(L/d)*((p*(U(i)^2))/2);
end
disp("for cast iron pipe");
printf(" Nre                              -deltap");
printf("\n %f                         %f",Nre(1),deltap(1));
printf("\n %f                       %f",Nre(2),deltap(2));
```

Scilab code Exa 10.5 von karman number

```
clc;
warning("off");
printf("\n\n example10.5 - pg417");
// given
L=300;   //[m] - length of pipe
d=0.06;   //[m] - inside diameter
deltap=147*10^3;   //[Pa] - pressure the pump can
   supply
ebyd=0.000762;   // relative roughness
p=1000;   //[kg/m^3] - density
mu=1*10^-3;   //[kg/m*sec] - viscosity
tauw=(d*(deltap))/(4*L);
// using the hit and trial method for estimation of
   flow velocity
// let
f=0.005;
U=((2*tauw)/(p*f))^(1/2);
Nre=(d*U*p)/mu;
// from the graph value of f at the above calculated
    reynolds no. and the given relative roughness(e/
   d)
f=0.0054;
U=((2*tauw)/(p*f))^(1/2);
Nre=(d*U*p)/mu;
// from the graph value of f at the above calculated
    reynolds no. and the given relative roughness(e/
   d)
f=0.0053;
U=((2*tauw)/(p*f))^(1/2);
Nre=(d*U*p)/mu;
// from the graph value of f at the above calculated
    reynolds no. and the given relative roughness(e/
```

```
    d)
f=0.0053;
// At this point the value of f is deemed unchanged
    from the last iteration .Hence, the values
    obtained after the third iteration are the
    converged values
printf("\n\n The maximum flow velocity is \n U=%f m/
    sec",U);
```

Scilab code Exa 10.6 von karman number

```
clc;
warning("off");
printf("\n\n example10.6 - pg419");
// given
L=300;  //[m] - length of pipe
d=0.06;  //[m] - inside diameter
deltap=147*10^3;  //[Pa] - pressure the pump can
    supply
ebyd=0.000762;  // relative roughness
p=1000;  //[kg/m^3] - density
mu=1*10^-3;  //[kg/m*sec] - viscosity
Nvk=((d*p)/mu)*((d*(deltap))/(2*L*p))^(1/2);
disp(Nvk,"von karman no.-");
// From the fig  at given von karman no and relative
     roughness the value of f is-
f=0.0055;
Nre=Nvk/(f^(1/2))
U=(Nre*mu)/(d*p);
printf("\n\n U=%f m/sec",U);
```

Scilab code Exa 10.7 the velocity head concept

```
clc;
warning("off");
printf("\n\n example10.7 - pg422");
// given
L=300;  //[m] - length of pipe
d=0.06;  //[m] - inside diameter
p=1000;  //[kg/m^3] - density
mu=1*10^-3;  //[kg/m*sec] - viscosity
Nre=[10^4 10^5];
U=(Nre*mu)/(d*p);
velocityhead=(U^2)/2;
N=(L/d)/45;  // no of velocity heads
deltap=p*N*(velocityhead);
for i=1:2
    disp(Nre(i),"Nre=");
    printf("\n\n velocity head =%f m^2/sec^2",
       velocityhead(i));
    printf("\n\n -deltap = %f kPa = %f psi",deltap(i
       )*10^-3,deltap(i)*1.453*10^-4);
end
```

Scilab code Exa 10.8 pipe fittings and valves

```
clc;
warning("off");
printf("\n\n example10.8 - pg439");
// given
mu=6.72*10^-4;  //[lb/ft*sec] - viscosity
p=62.4;  //[lb/ft^3] - density
S=0.03322;  //[ft^2] - flow area
d=0.206;  //[ft]
e=1.5*10^-4;  // absolute roughness for steel pipe
ebyd=e/d;
Nre=10^5;
// friction factor as read from fig in book for the
```

```
given reynolds no. and relative roughness is-
f=0.0053;
U=(Nre*mu)/(p*d);
Q=U*S;
gc=32.174;
// (a) equivalent length method
deltapbyL=f*(4/d)*(p*(U^2))*(1/(2*gc))*(6.93*10^-3);
// using L=Lpipe+Lfittings+Lloss;
Lfittings=2342.1*d;
kc=0.50;  //  due to contraction loss
ke=1;  // due to enlargement loss
Lloss=(kc+ke)*(1/(4*f))*d;
Lpipe=137;
L=Lpipe+Lfittings+Lloss;
deltap=deltapbyL*L;
patm=14.696;  //[psi] - atmospheric pressure
p1=patm+deltap;
printf("\n\n (a)The inlet pressure is\n p1=%f psi",p1);
// (b) loss coefficient method
// using the equation deltap/p=-(Fpipe+Ffittings+Floss)
L=137;
kfittings=52.39;
sigmaF=((4*f*(L/d))+kc+ke+kfittings)*((U^2)/(2*gc));
deltap=(p*sigmaF)/(144);
p1=patm+deltap;
printf("\n\n (b)The inlet pressure is \n p1=%f psi",p1);
printf("\n\n Computation of the pressure drop by the loss coefficient method differs from the equivalent length method by less than 1 psi");
```

Scilab code Exa 10.9 gases

```
clc;
warning("off");
printf("\n\n example10.9 - pg443");
// given
L1=50;  //[m] - length of first pipe
L2=150;  //[m] - length of second pipe
L3=100;  //[m] - length of third pipe
d1=0.04;  //[m] - diameter of first pipe
d2=0.06;  //[m] - diameter of second pipe
d3=0.08;  //[m] - diameter of third pipe
deltap=-1.47*10^5;  //[kg/m*sec] - pressure drop
mu=1*10^-3;  //[kg/m*sec] - viscosity
p=1000;  //[kg/m^3] - density
// for branch 1
S=(%pi*(d1^2))/4;
Nvk=((d1*p)/mu)*(-(d1*deltap)/(2*L1*p))^(1/2);
f=(1/(4*log10(Nvk)-0.4))^2;
U=(((-deltap)/p)*(d1/L1)*(2/4)*(1/f))^(1/2);
w1=p*U*S;
printf("\n\n For first branch w1=%f kg/sec",w1);
// for branch 2
S=(%pi*(d2^2))/4;
Nvk=((d2*p)/mu)*(-(d2*deltap)/(2*L2*p))^(1/2);
f=(1/(4*log10(Nvk)-0.4))^2;
U=(((-deltap)/p)*(d2/L2)*(2/4)*(1/f))^(1/2);
w2=p*U*S;
printf("\n\n For second branch w2=%f kg/sec",w2);
// for branch 3
S=(%pi*(d3^2))/4;
Nvk=((d3*p)/mu)*(-(d3*deltap)/(2*L3*p))^(1/2);
f=(1/(4*log10(Nvk)-0.4))^2;
U=(((-deltap)/p)*(d3/L3)*(2/4)*(1/f))^(1/2);
w3=p*U*S;
printf("\n\n For third branch w3=%f kg/sec",w3);
// total flow rate w=w1+w2+w3
w=w1+w2+w3;
printf("\n\n total flow rate is w=%f kg/sec",w);
```

Scilab code Exa 10.11 complex fluid flow systems

```
clc;
warning("off");
printf("\n\n example10.11 - pg 447");
// given
sp=1.1;
p=sp*62.4;  //[lb/ft^3] - density
mu=2*6.72*10^-4;  //[lb/ft*sec] - viscosity
Q=400;  //[gpm] - volumetric flow rate
e=1.5*10^4;  //roughness of steel pipe
gc=32.174;
kexit=1;
kentrance=0.5;
// 4 in schedule pipe
d=4.026/12;  //[ft]
U4=Q/39.6;  //[ft/sec]
Lgv=13.08;
Lglv=114.1;
Le=40.26;
Lpipe_4=22;
Lfittings_4=Lgv+Lglv+Le;
Lloss=0;
L_4=Lpipe_4+Lfittings_4+Lloss;
Nre_4=(d*U4*p)/mu;
f=0.00475;
Fpipe_4=((4*f*L_4)/d)*(U4^2)*(1/(2*gc));
Floss_4=((kentrance+0)*(U4^2))/(2*gc);
// 5 in schedule pipe
d=5.047/12;
U5=Q/62.3;
Lgv=10.94;
Le=75.71;
Lpipe_5=100;
```

```
Lfittings_5=Lgv+Le;
Lloss=0;
L_5=Lpipe_5+Lfittings_5+Lloss;
Nre=(d*U5*p)/mu;
f=0.00470;
Fpipe_5=((4*f*L_5)/d)*(U5^2)*(1/(2*gc));
Floss_5=((kexit+0)*(U5^2))/(2*gc);
// 6 in schedule pipe
d=6.065/12;
U6=Q/90;
Lgv=6.570;
Le=30.36;
Lpipe_6=4;
Lfittings_6=Lgv+Le;
Lloss=0;
L_6=Lpipe_6+Lfittings_6+Lloss;
Nre=(d*U6*p)/mu;
f=0.00487;
Fpipe_6=((4*f*L_6)/d)*(U6^2)*(1/(2*gc));
kc=0.50;
Floss_6=kc*((U6^2)/(2*gc));
Ffittings=0;
deltap_6=p*(Fpipe_6+Ffittings+Floss_6);
// 3/4 in 18 gauge tube
d=0.652112/12;
L_3by4=15;
U_3by4=(Q*0.962)/100;
Floss_3by4=100*(kexit+kentrance)*((U_3by4^2)/2);
Nre=d*U_3by4*p*(1/mu);
// clearly the flow is turbulent
f=0.08*((Nre)^(-1/4))+0.012*((d)^(1/2));
deltap_3by4=((4*f*p*L_3by4)/d)*((U_3by4^2)/(2*gc));
Fpipe_3by4=100*((4*f*L_3by4)/d)*((U_3by4^2)/(2*gc));
deltap_spraysystem=25; //[psi]
Fspraysystem=(deltap_spraysystem/p)*(144);
delta_p=[p*(kexit+kentrance)]*[(U_3by4^2)/(2*gc)];
Fpipe=Fpipe_4+Fpipe_5+Fpipe_6;
Floss=Floss_4+Floss_5+Floss_6+Floss_3by4;
```

```
ws=0+([(15^2)-0]/[2*gc])+38.9+382.5;
w=(Q*p)/(7.48);
Ws=(ws*w)/(33000);
efficiency=0.6;
Ws_actual=Ws/efficiency
printf("\n\n The power supplied to th pump is\n W_actual = %f",Ws_actual);
```

Scilab code Exa 10.12 complex fluid flow systems

```
clc;
warning("off");
printf("\n\n example10.12 - pg454");
// given
kexit=1;
kentrance=0.5;
Q=400;  //[gpm] - volumetric flow rate
gc=32.174;
// for 4 inch pipe
d=4.026;  //[inch]
L=22;  //[ft]
Lbyd=(L*12)/(d);
// adding the contributions due to fittings
Lbyd=Lbyd+3*13+340+4*30;
N=Lbyd/45;
N=N+kentrance+0;
U4=Q/39.6;  //[ft/sec]
Fpipe_4=(N*(U4^2))/(2*gc);
printf("\n\n F(4 in.pipes) = %f ft*lbf/lbm",Fpipe_4);
// for 5 inch pipe
L=100;  //[ft]
d=5.047;  //[inch]
Lbyd=(L*12)/(d);
// valves contributes 26 diameters and six elbows
```

```
   contribute 30 diameters ecah;therefore
Lbyd=Lbyd+26+6*30;
N=Lbyd/45;  // no. of velocity heads
N=N+kexit+kentrance;
U5=Q/62.3;
Fpipe_5=(N*(U5^2))/(2*gc);
printf("\n\n F(5 in.pipes) = %f ft*lbf/lbm",Fpipe_5)
   ;
// for 6 inch pipe
d=6.065;  //[inch]
L=5;  //[ft]
Lbyd=(L*12)/(d);
// adding the contributions due to fittings
Lbyd=Lbyd+1*13+2*30;
N=Lbyd/45;
N=N+0+kentrance;
U6=Q/90;
Fpipe_6=(N*(U6^2))/(2*gc);
printf("\n\n F(6 in.pipes) = %f ft*lbf/lbm",Fpipe_6)
   ;
F_largepipes=Fpipe_4+Fpipe_5+Fpipe_6;
printf("\n\n F(large pipes) = %f ft*lbf/lbm",
   F_largepipes);
```

Scilab code Exa 10.14 non circular conduits

```
clc;
warning("off");
printf("\n\n example10.14 - pg459");
// given
l=0.09238;
rh=0.1624*l;
L=300;
de=4*rh;
p=1000;  //[kg/m^3]
```

```
mu=10^-3;  //[kg/m*sec]
Uavg=1.667;
Nre=(de*Uavg*p)/mu;
f=0.0053;
deltap=((4*f*L)/de)*(p*(Uavg^2)*(1/2));
printf("\n\n -deltap = %e kg/m*s = %e N/m^2 = %f kPa
    ",deltap,deltap,deltap*10^-3);
```

Scilab code Exa 10.15 orifice meter

```
clc;
warning("off");
printf("\n\n example10.15 - pg466");
// given
Q=400;  //[gpm]
p=1.1*62.4;  //[lbm/ft^3]
mu=2*(6.72*10^-4);  //[lb/ft*sec]
e=1.5*10^4;
// 4 inch schedule pipe
d=0.3355;
S=(%pi*(d^2))/4;
U4=Q/39.6;
ebyd=e/d;
w=3671/60;
pm=13.45*62.4;
g=32.1;
gc=32.174;
deltaz=2.5;
deltap=(g/gc)*(pm-p)*(deltaz);
betaa=((1)/(1+[(2*p*gc)*(deltap)]*(((0.61*S)/w)^2)))
    ^(1/4);
d2=betaa*d;
Nre2=(4*w)/(%pi*d2*mu);
a=(1/30)*4.026;
b=(1/4)*(2.013-1.21);
```

```
c=(1/8)*(2.42);
if a<b then
    if a<c then
        opt=a;
    else
        opt=c;
    end
else
    if b<c then
        opt=b;
    else
        opt=c;
    end
end
printf("\n\n The pertinent orifice details are \n
   orifice diameter = %f in \n corner taps, square
   edge\n orifice plate not over %f in thick",d2*12,
   opt);
```

Scilab code Exa 10.16 venturi and nozzle

```
clc;
warning("off");
printf("\n\n example10.16 - pg470");
// given
Q=400;  //[gpm]
p=1.1*62.4;  //[lbm/ft^3]
mu=2*(6.72*10^-4);  //[lb/ft*sec]
e=1.5*10^4;
// 4 inch schedule pipe
d=0.3355;
S=(%pi*(d^2))/4;
U4=Q/39.6;
ebyd=e/d;
w=3671/60;
```

```
pm=13.45*62.4;
g=32.1;
gc=32.174;
Nre=(d*U4*p)/mu;
if Nre>10^4 then
    c=0.98;
end
deltaz=2.5;
deltap=(g/gc)*(pm-p)*(deltaz);
betaa=((1)/(1+[(2*p*gc)*(deltap)]*(((c*S)/w)^2)))
   ^(1/4);
d2=betaa*d;
printf("\n\n The pertinentr details of the venturi
   design are\n Throat diameter = %f inch\n Approach
    angle = 25\n Divergence angle = 7",d2*12);
```

Scilab code Exa 10.17 pitot tube

```
clc;
warning("off");
printf("\n\n example10.17 - pg477");
// given
Uzmax=3.455;  //[ft/sec]
m=32;
a1=-0.3527;
a2=-0.6473;
rbyro=0.880;
UzbyUzmax=1+a1*(rbyro^2)+a2*(rbyro^(2*m));
Uz=Uzmax*(UzbyUzmax);
Uzavg=(4/9)*Uzmax+(5/18)*(Uz+Uz);
printf("\n\n the average velocity is \n Uzavg = %f
   ft/sec \n\n Thus, in this example there is an
   inherent error of 5.5 percent, even before any
   experimental errors are introduced",Uzavg);
```

Chapter 11

heat and mass transfer in duct flow

Scilab code Exa 11.1 conduction

```
clc;
warning("off");
printf("\n\n example11.1 - pg497");
// given
K_drywall=0.28;  //[Btu/ft*degF] - thermal conductivity of dry wall
K_fibreglass=0.024;  //[Btu/ft*degF] - thermal conductivity of fibre glass
K_concrete=0.5;  //[Btu/ft*degF] - thermal conductivity of concrete
T4=0;  //[degF]
T1=65;  //[degF]
deltaT=T4-T1;  //[degF]
a=1;  //[ft^2] - assuming area of 1 ft^2
deltax1=0.5/12;  //[ft]
deltax2=3.625/12;  //[ft]
deltax3=6/12;  //[ft]
R1=deltax1/(K_drywall*a);  //[h*degF/Btu]
R2=deltax2/(K_fibreglass*a);  //[h*degF/Btu]
```

```
R3=deltax3/(K_concrete*a);   //[h*degF/Btu]
qx=deltaT/(R1+R2+R3);
q12=-qx;
q23=-qx;
q34=-qx;
deltaT1=(-q12)*deltax1*(1/(K_drywall*a));
T2=T1+deltaT1;
deltaT2=(-q23)*deltax2*(1/(K_fibreglass*a));
T3=T2+deltaT2;
deltaT3=(-q34)*deltax3*(1/(K_concrete*a));
T4=T3+deltaT3;
printf("\n\n T1 = %f degF\n T2 = %f degF\n T3 = %f
    degF\n T4 = %f degF",T1,T2,T3,T4);
```

Scilab code Exa 11.2 the resistance concept

```
clc;
warning("off");
printf("\n\n example11.2 - pg501");
// given
r1=(2.067/2)/(12);   //[ft]
r2=r1+0.154/12;   //[ft]
r3=r2+3/12;   //[ft]
L=1;   //[ft]
Ka=26;   //[Btu/h*ft*degF]
Kb=0.04;   //[Btu/h*ft*degF]
T1=50;   //[degF]
Ra=(log(r2/r1))/(2*%pi*L*Ka);
Rb=(log(r3/r2))/(2*%pi*L*Kb);
R=Ra+Rb;
deltaT=-18;   //[degF] - driving force
Qr=-(deltaT/(R));
disp(Qr);
deltaT1=(-Qr)*(Ra);
T2=T1+deltaT1;
```

```
printf("\n\n The interface temperature is \n T2 = %f
    degF",T2);
```

Scilab code Exa 11.3 the resistance concept

```
clc;
warning("off");
printf("\n\n example11.3 - pg502");
// given
Ra=8.502*10^-4;  //[h*degF*Btu^-1]
Rb=5.014;  //[h*degF*Btu^-1]
r1=(2.067/2)/(12);  //[ft]
r2=r1+0.154/12;  //[ft]
r3=r2+3/12;  //[ft]
d1=2*r1;
d0=2*r3;
h0=25;  //[Btu/h*ft^2*degF]
h1=840;  //[Btu/h*ft^2*degF]
L=1;  //[ft] - considering 1 feet length
R0=1/(h0*%pi*d0*L);
R1=1/(h1*%pi*d1*L);
R=R0+R1+Ra+Rb;
disp(R);
deltaT=-400;  //[degF]
Qr=-(deltaT)/R;
disp(Qr);
// the heat loss calculated above is the heat loss
   per foot.therefore for 500 ft
L=500;
Qr=Qr*L;
printf("\n\n the heat loss for a 500 feet pipe is \n
    qr = %e Btu/h",Qr);
```

Scilab code Exa 11.5 heat and mass transfer during turbulent flow

```
clc;
warning("off");
printf("\n\n example11.5 - pg521");
// given
Nre=50000;
d=0.04;  //[m] - diameter of pipe
// physical properties of water
T1=293.15;  //[K]
T2=303.15;  //[K]
T3=313.15;  //[K]
p1=999;  //[kg/m^3] - density of water at temperature T1
p2=996.0;  //[kg/m^3] - density of water at temperature T2
p3=992.1;  //[kg/m^3] - density of water at temperature T3
mu1=1.001;  //[cP] - viscosity of water at temperature T1
mu2=0.800;  //[cP] - viscosity of water at temperature T2
mu3=0.654;  //[cP] - viscosity of water at temperature T3
k1=0.63;  //[W/m*K] - thermal conductivity of water at temperature T1
k2=0.618;  //[W/m*K] - thermal conductivity of water at temperature T2
k3=0.632;  //[W/m*K] - thermal conductivity of water at temperature T3
cp1=4182;  //[J/kg*K] - heat capacity of water at temperature T1
cp2=4178;  //[J/kg*K] - heat capacity of water at temperature T2
cp3=4179;  //[J/kg*K] - heat capacity of water at temperature T3
Npr1=6.94;  // prandtl no. at temperature T1
Npr2=5.41;  // prandtl no. at temperature T2
```

```
Npr3=4.32;  // prandtl no. at temperature T3
// (a)  Dittus -Boelter-this correction evalutes all properties at the mean bulk temperature,which is T1
kmb=0.603
h=(kmb/d)*0.023*((Nre)^(0.8))*((Npr1)^0.4);
printf("\n\n (a) Dittus -Boelter\n the heat transfer coefficient is \n h = %f W/m^2*K = %f Btu/ft^2*h^-1*degF",h,h*0.17611);
// (b) Seider Tate-this correlation evaluates all the properties save muw at the mean bulk temperature
h=(kmb/d)*(0.027)*((Nre)^0.8)*((Npr1)^(1/3))*((mu1/mu3)^0.14);
printf("\n\n (b) Seider Tate\n the heat transfer coefficient is \n h = %f W/m^2*K = %f Btu/ft^2*h^-1*degF",h,h*0.17611);
// (c) Sleicher-Rouse equation
a=0.88-(0.24/(4+Npr3));
b=(1/3)+0.5*exp((-0.6)*Npr3);
Nref=Nre*(mu1/mu2)*(p2/p1);
Nnu=5+0.015*((Nref)^a)*((Npr3)^b);
h=Nnu*(kmb/d);
printf("\n\n (c) Sleicher-Rouse equation\n the heat transfer coefficient is \n h = %f W/m^2*K = %f Btu/ft^2*h^-1*degF",h,h*0.17611);
// (d) Colbum Analogy- the j factor for heat transfer is calculated
jh=0.023*((Nref)^(-0.2));
Nst=jh*((Npr2)^(-2/3));
U=(Nre*mu1*10^-3)/(d*p1);
h=Nst*(p1*cp1*U);
printf("\n\n (d) Colbum Analogy\n the heat transfer coefficient is \n h = %f W/m^2*K = %f Btu/ft^2*h^-1*degF",h,h*0.17611);
// (e) Friend-Metzner
f=0.005227;
Nnu=((Nre)*(Npr1)*(f/2)*((mu1/mu3)^0.14))
```

```
    /(1.20+((11.8)*((f/2)^(1/2))*(Npr1-1)*((Npr1)
    ^(-1/3))));
h=Nnu*(kmb/d);
printf("\n\n (e) Friend-Metzner\n the heat transfer
    coefficient is \n h = %f W/m^2*K = %f Btu/ft^2*h
    ^-1*degF",h,h*0.17611);
// (f) Numerical analysis
Nnu=320;
h=Nnu*(kmb/d);
printf("\n\n (f) Numerical analysis\n the heat
    transfer coefficient is \n h = %f W/m^2*K = %f
    Btu/ft^2*h^-1*degF",h,h*0.17611);
```

Scilab code Exa 11.6 heat and mass transfer during turbulent flow

```
clc;
warning("off");
printf("\n\n example11.6 - pg525");
// given
Tw=680;  //[K] - temperature at the wall
Tb=640;  //[K] - temperature at the bulk
Tf=(Tw+Tb)/2;  //[K]
Nre=50000;
vmb=2.88*10^-7;
vf=2.84*10^-7;
Nref=Nre*(vmb/vf);
k=27.48;
d=0.04;
// from table 11.3 the prandtl no. is
Npr=8.74*10^-3
// constant heat flow
Nnu=6.3+(0.0167)*((Nref)^0.85)*((Npr)^0.93);
h=Nnu*(k/d);
printf("\n\n constant heat flow\n h = %f W/m^2*K =
    %f Btu/ft^2*h*degF",h,h*0.17611);
```

```
// constant wall temperature
Nnu=4.8+0.0156*((Nref)^0.85)*((Npr)^0.93);
h=Nnu*(k/d);
printf("\n\n constant wall temperature\n h = %f W/m
    ^2*K = %f Btu/ft^2*h*degF",h,h*0.17611);
```

Scilab code Exa 11.7 double pipe heat exchangers simple solutions

```
clc;
warning("off");
printf("\n\n example11.7 - pg536");
// given
di=0.620;   //[inch] - internal diameter
d0=0.750;   //[inch] - outer diameter
Ai=0.1623;   //[ft^2/ft]
Ao=0.1963;   //[ft^2/ft]
wc=12*(471.3/0.9425);
cp=1;   //[Btu/lbm*degF] - heat capacity of water
Tco=110;
Tci=50;
qtotal=wc*cp*(Tco-Tci);
deltaH_coldwater=3.6*10^5;
deltaH_vapourization=1179.7-269.59;
wh=deltaH_coldwater/deltaH_vapourization;
hi=80;   //[Btu/h*ft^2*degF]
ho=500;   //[Btu/h*ft^2*degF]
km=26;   //[Btu/h*ft*degF]
Ui=1/((1/hi)+((Ai*log(d0/di))/(2*%pi*km))+(Ai/(Ao*ho
    )));
disp(Ui)
deltaT1=300-50;
deltaT2=300-110;
LMTD=(deltaT1-deltaT2)/(log(deltaT1/deltaT2));
A=qtotal/(Ui*LMTD);
L=A/Ai;
```

```
printf("\n\n the length of the heat exchanger is \n L = %f ft",L);
```

Scilab code Exa 11.8 double pipe heat exchangers simple solutions

```
clc;
warning("off");
printf("\n\n example11.8 - pg537");
// given
L=30;  //[ft] - length
Ai=0.1623*L;
di=0.620;  //[inch] - internal diameter
d0=0.750;  //[inch] - outer diameter
Ao=0.1963*L;  //[ft^2/ft]
wc=12*(471.3/0.9425);
cp=1;  //[Btu/lbm*degF] - heat capacity of water
deltaH_coldwater=3.6*10^5;
deltaH_vapourization=1179.7-269.59;
wh=deltaH_coldwater/deltaH_vapourization;
hi=80;  //[Btu/h*ft^2*degF]
ho=500;  //[Btu/h*ft^2*degF]
km=26;  //[Btu/h*ft*degF]
Ui=1/((1/hi)+(((Ai/L)*log(d0/di))/(2*%pi*km))+(Ai/(Ao*ho)));
deltaT1=300-50;
deltaT=deltaT1/(exp((Ui*Ai)/(wc*cp)));
Tsat=300;
Tc2=Tsat-deltaT;
printf("\n\n Therefore, the outlet temperature of the cold fluid is \n Tc2 = %f degF",Tc2);
```

Scilab code Exa 11.9 double pipe heat exchangers simple solutions

```
clc;
warning("off");
printf("\n\n example11.9 - pg538");
// given
Ai=4.869;
wc=6000;
cp=1;
Rf=0.002;
Uclean=69.685;
Udirty=1/(Rf+(1/Uclean));
deltaT1=300-50;
deltaT2=deltaT1/(exp((Udirty*Ai)/(wc*cp)));
Th2=300;
Tc2=Th2-deltaT2;
printf("\n\n the outlet temperature is \n Tc2 = %f
   degF",Tc2);
```

Scilab code Exa 11.10 multipass heat exchangers equipment

```
clc;
warning("off");
printf("\n\n example11.10 - pg544");
// given
Ui=325;  //[W/m^2*K] - overall heat transfer
   coefficient
Thi=120;  //[degC] - inlet temperature of
   hydrocarbon
Tho=65;  //[degC] - outlet temperature of
   hydrocarbon
Tci=15;  //[degC] - inlet temperature of water
Tco=50;  //[degC] - outlet temperture of water
cp=4184;  //[J/kg*K] - heat capacity of water
ch=4184*0.45;  //[J/kg*K] - heat capacity of
   hydrocarbon
wc=1.2;  //[kg/sec] - mass flow rate of water
```

```
wh=((wc*cp)*(Tco-Tci))/((ch)*(Thi-Tho));
qtotal=wc*cp*(Tco-Tci);
// (a) - parallel double pipe
F=1;
Thi=120;  //[degC] - inlet temperature of hydrocarbon
Tho=65;  //[degC] - outlet temperature of hydrocarbon
Tci=15;  //[degC] - inlet temperature of water
Tco=50;  //[degC] - outlet temperture of water
deltaT1=Thi-Tci;
deltaT2=Tho-Tco;
LMTD=(deltaT2-deltaT1)/(log(deltaT2/deltaT1));
Ai=qtotal/((Ui*LMTD));
printf("\n\n (a) parallel double pipe \n Ai = %f m^2",Ai);
// (b) - counter flow
F=1;
Thi=120;  //[degC] - inlet temperature of hydrocarbon
Tho=65;  //[degC] - outlet temperature of hydrocarbon
Tco=15;  //[degC] - inlet temperature of water
Tci=50;  //[degC] - outlet temperture of water
deltaT1=Thi-Tci;
deltaT2=Tho-Tco;
LMTD=(deltaT2-deltaT1)/(log(deltaT2/deltaT1));
Ai=qtotal/((Ui*LMTD));
printf("\n\n (b) counter flow \n  Ai = %f m^2",Ai);
// (c) - 1-2 shell and tube
Thi=120;  //[degC] - inlet temperature of hydrocarbon
Tho=65;  //[degC] - outlet temperature of hydrocarbon
Tci=15;  //[degC] - inlet temperature of water
Tco=50;  //[degC] - outlet temperture of water
Z=(Thi-Tho)/(Tco-Tci);
nh=(Tco-Tci)/(Thi-Tci);
```

```
deltaT1=Thi-Tco;
deltaT2=Tho-Tci;
F=0.92;
LMTD=(F*(deltaT2-deltaT1))/(log(deltaT2/deltaT1));
Ai=qtotal/((Ui*LMTD));
printf("\n\n (c) 1-2 shell and tube \n  Ai = %f m^2"
   ,Ai);
// (d) - 2-4 shell and tube
Thi=120;  //[degC] - inlet temperature of
   hydrocarbon
Tho=65;  //[degC] - outlet temperature of
   hydrocarbon
Tci=15;  //[degC] - inlet temperature of water
Tco=50;  //[degC] - outlet temperture of water
Z=(Thi-Tho)/(Tco-Tci);
nh=(Tco-Tci)/(Thi-Tci);
F=0.975;
LMTD=(F*(deltaT2-deltaT1))/(log(deltaT2/deltaT1));
Ai=qtotal/((Ui*LMTD));
printf("\n\n (d) 2-4 shell and tube \n  Ai = %f m^2"
   ,Ai);
```

Chapter 12

transport past immersed bodies

Scilab code Exa 12.2 the laminar boundary layer

```
clc;
warning("off");
printf("\n\n example12.2 - pg562");
p=1.2047*0.06243;  //[lb/ft^3]
mu=(18.17*10^-6)*(0.6720);  //[lb/ft*sec]
v=mu/p;
x=2;  //[ft]
U=6;  //[ft/sec]
Nre=(x*U)/v;
disp("The Reynolds number is well within the laminar region",Nre,"Nre=");
del=5*x*(Nre)^(-1/2);
C1=0.33206;
Cd=2*C1*(Nre)^(-1/2);
L2=2;  //[ft]
L1=1;  //[ft]
b=1;
F=((2*(C1)*U*b))*((mu*p*U)^(1/2))*(((L2)^(1/2))-((L1)^(1/2)));
gc=32.174;
F=F/gc;
```

```
printf("\n\n The value of F properly expressed in
    force units is \n F=%e lbf",F);
```

Scilab code Exa 12.3 turbulent boundary layer

```
clc;
warning("off");
printf("\n\n example12.3 - pg569");
U=3;   //[m/sec]
x1=1;   //[m]
x2=2;   //[m]
p=1/(1.001*10^-3);   //[kg/m^3];
mu=1*10^-3;   //[kg/m*sec]
Nre1=(x1*U*p)/(mu);
Nre2=(x2*p*U)/(mu);
tauw=(1/2)*(p*(U^2))*((2*log10(Nre1)-0.65)^(-2.3));
B=1700;
Cd=(0.455*(log10(Nre2))^-2.58)-(B/(Nre2));
Lb=2.0;
F=(1/2)*(p*(U^2))*(Lb)*(Cd);
printf("\n\n the drag on the plate is \n F = %f kg*m
    /sec^2 = %f N",F,F);
```

Scilab code Exa 12.5 heat and mass transfer during boundary layer flow past a flat plate

```
clc;
warning("off");
printf("\n\n example12.5 - pg576");
T=290;   //[K] - temperature of flowing water
U=3;   //[m/sec] - free stream velocity
Tfs=285;   //[K] - temperature of free stream
vr=10^-3;   //[m^3/kg] - volume per unit mass
```

```
p=1/vr;  //[kg/m^3] - density of water at Tfs
mu=1225*10^-6;  //[N*sec/m^2]
k=0.590;  //[W/m*K]
Npr=8.70;
//  (a) The length of laminar boundary
Nre=5*10^5;
xc=(Nre)*(mu/(p*U));
printf("\n\n (a) The length of laminar boundary is \
   n xc = %f m",xc);
//  (b) Thickness of the momentum boundary layer and
    thermal boundary layer
del=5*xc*((Nre)^(-1/2));
delh=del*((Npr)^(-1/3));
printf("\n\n (b) The thickness of momentum boundary
   layer is \n del = %e m\n The thickness of the
   hydryodynamic layer is \n delh = %e m",del,delh);
// (c) Local heat transfer coefficient
x=0.2042;  //[ft]
hx=((0.33206*k)/(x))*((Nre)^(1/2))*((Npr)^(1/3));
printf("\n\n (c) The local heat transfer coefficient
    is \n h = %f W/m^2*K = %f Btu/hr*ft^2*degF",hx,
   hx*0.17611);
// (d) Mean heat transfer coefficient
hm=2*hx;
printf("\n\n (d) The mean heat transfer coefficient
   is \n h = %f W/m^2*K = %f Btu/hr*ft^2*degF",hm,hm
   *0.17611);
```

Scilab code Exa 12.10 stokes flow past a sphere

```
clc;
warning("off");
printf("\n\n example12.10 - pg590");
// given
T=293.15;  //[K]
```

```
pp=999;  //[kg/m^3] - density of water
mu=0.01817*10^-3;  //[kg/m*sec] - viscosity of air
p=1.205;  //[kg/m^3] - density of air
d=5*10^-6;  //[m] - particle diameter
g=9.80;  //[m/sec^2]
rp=d/2;
Ut=((2*g*(rp^2))*(pp-p))/(9*mu);
Nre=(d*Ut*p)/(mu);
// clearly the flow is in the stokes law region at this low reynolds number;therefore , the drag force is
Fp=6*%pi*mu*rp*Ut;
printf("\n\n The drag force is \n Fp = %e N",Fp);
```

Scilab code Exa 12.11 drag coefficient correlations

```
clc;
warning("off");
printf("\n\n example12.11 - pg591");
// given
T=293.15;  //[K]
pp=999;  //[kg/m^3] - density of water
mu=0.01817*10^-3;  //[kg/m*sec] - viscosity of air
p=1.205;  //[kg/m^3] - density of air
d=5*10^-6;  //[m] - particle diameter
g=9.80;  //[m/sec^2]
rp=d/2;
Ut=((2*g*(rp^2))*(pp-p))/(9*mu);
Nre=(d*Ut*p)/(mu);
t=((-2*(rp^2)*pp))/(9*mu)*(log(1-0.99));
printf("\n\n Time for the drop of water in previous example from an initial velocity of zero to 0.99*Ut is \n t = %e sec",t);
printf("\n\n In other words, the drop accelerates almost instantaneously to its terminal velocity")
```

```
;
```

Scilab code Exa 12.12 drag coefficient correlations

```
clc;
warning("off");
printf("\n\n example12.12 - pg 594");
// given
pp=1.13*10^4;   //[kg/m^3] - density of lead particle
p=1.22;   //[kg/m^3] - density of air
g=9.80;   //[m/sec^2] - acceleration due to gravity
d=2*10^-3;   //[m] - diameter of particle
mu=1.81*10^-5;   //[kg/m*sec] - viscosity of air
// let us assume
Cd=0.44;
Ut=((4*d*g*(pp-p))/(3*p*Cd))^(1/2);
disp(Ut)
Nre=(Ut*d*p)/(mu);
// from fig 12,16 value of Cd is
Cd=0.4;
Ut=((4*d*g*(pp-p))/(3*p*Cd))^(1/2);
Nre=(Ut*d*p)/(mu);
// Within the readibility of the chart Cd is
   unchanged and therefore the above obtained Cd is
   the final answer
printf("\n\n The terminal velocity is \n Ut = %f m/
   sec",Ut);
```

Scilab code Exa 12.13 drag coefficient correlations

```
clc;
warning("off");
printf("\n\n example12.13 - pg595");
```

```
// given
distance=1/12;  //[ft]
time=60;  //[sec]
Ut=distance/time;
mu=1.68;  //[lb/ft*sec] - viscosity
pp=58;  //[lb/ft^3] - density of sphere
p=50;  //[lb/ft^3] - density of polymer solution
g=32;  //[ft/sec] - acceleration due to gravity
rp=((9*mu)*(Ut)*((2*g)^(-1))*((pp-p)^(-1)))^(1/2);
printf("\n\n The required particle diameter would be about %f inch",rp*2*12);
Nre=(rp*2*Ut*p)/(mu);
disp(Nre,"Nre=");
printf("\n\n This reynolds number is well within the stokes law region ; thus the design is reasonable");
```

Scilab code Exa 12.14 liquid solid fluidization

```
clc;
warning("off");
printf("\n\n example12.14 - pg616");
// given
T=842;  //[degF] - temperature
P=14.6;  //[psia] - pressure
p=0.487;  //[kg/m^3] - density of air
mu=3.431*10^-5;  //[kg/m*sec] - viscosity of air
k=0.05379;  //[W/m*K] - thermal conductivity
Npr=0.7025;  //prandtl no.
// (a) static void fraction
mcoal=15*2000;  //[lb] - mass of coal
pcoal=94;  //[lbm/ft^3] - density of coal
d=10;  //[ft]
L=7;  //[ft]
area=((%pi*(d^2))/4);
```

```
Vcoal=mcoal/pcoal;
Vtotal=area*L;
e=(Vtotal-Vcoal)/(Vtotal);
disp(e,"(a) The void fraction is E=");
// (b) minimum void fraction and bed height
d=200;  //[um] - particle diameter
Emf=1-0.356*((log10(d))-1);
// this value seems to be a lottle low and therefore 0.58 will be used
Emf=0.58;
Lmf=((L)*(1-e))/(1-Emf);
printf("\n\n (b) The bed height is \n Lmf = %f ft",Lmf);
// (c) Minimum fluidization velocity
P1=20;  //[psia]
P2=14.696;  //[psia]
p1=(p*P1)/(P2);
// the archimides no. is
g=9.78;  //[m/sec^2]
Nar=p1*g*((d*10^-6)^3)*(1506-p1)*((1/(mu)^2));
C1=27.2;
C2=0.0408;
Nremf=(((C1^2)+C2*Nar)^(1/2))-C1;
Umf=(Nremf*mu)/((d*10^-6)*p1);
printf("\n\n (c) The minimum fluidization velocity is \n Umf = %f m/sec",Umf);
// (d) Minimum pressure
deltapmf=(1506-p1)*(g)*(1-Emf)*((Lmf*12*2.54)/(100))+p1*g*Lmf;
printf("\n\n (c) The minimum pressure drop for fluidization is \n -deltapmf = %e Pa",deltapmf);
// (e) Particle settling velocity
Cd=0.44;
Ut=(((8*((d*10^-6)/2)*g)*(1506-p1))/(3*p1*Cd))^(1/2);
Nrep=(Ut*d*10^-6*p1)/(mu);
disp(Nrep,"Nrep=");
// clearly at the point of minimum velocity for fast
```

```
fluidization , the terminal settling velocity is not in the range of Newtons law.Therefore the eq. for the transition region will be tried
Ut=((5.923/18.5)*((((d*10^-6)*p1)/(mu))^(0.6))^(1/(2-0.6))
printf("\n\n (e) The particle settling velocity is \n Ut = %f m/sec",Ut);
// (f) Bed to wall heat transfer coefficient
Nrefb=(d*10^-6)*2.5*Umf*p1*(1/mu);
Nnufb=0.6*Npr*((Nrefb)^(0.3));
hw=Nnufb*(k/(d*10^-6));
printf("\n\n (f) The bed to wall heat transfer coefficient is \n hw = %f W/m^2*K",hw);
```

Scilab code Exa 12.15 liquid solid fluidization

```
clc;
warning("off");
printf("\n\n example12.5 - pg618");
// given
pp=249.6;  //[lb/ft^3] - density of catalyst
p=58;  //[lb/ft^3] - density of liquid
g=32.174;  //[ft/sec^2]
gc=32.174;
Lmf=5;  //[ft] - height of bed
mu=6.72*10^-3;  //[lbm/ft*sec] - viscosity of liquid
dp=0.0157/12;  //[ft] - diameter of particle
emf=0.45;
deltapmf=(pp-p)*(g/gc)*(1-emf)*(Lmf);
Nar=(p*g*dp^3)*(pp-p)*(1/(mu)^2);
C1=27.2;
C2=0.0408;
Nremf=(((C1^2)+C2*Nar)^(1/2))-C1;
Umf=Nremf*(mu/(dp*p));
printf("\n\n Minimum fluidization velocity is \n Umf
```

```
    = %e ft/sec",Umf);
```

Scilab code Exa 12.16 single cyclinder heat transfer

```
clc;
warning("off");
printf("\n\n example12.16 - pg624");
// given
d=24*10^-6;  //[m] - diameter of wire
T=415;  //[K] - operating temperature of hot wire
   anemometer
P=0.1;  //[W] - power consumption
L=250*d;
Tair=385;  //[K] - temperature of air in duct
A=%pi*d*L;
Tfilm=(T+Tair)/2;
// properties of air at Tfilm
p=0.8825;  //[kg/m^3]
mu=2.294*10^-5;  //[kg/m*s]
cpf=1013;  //[J*kg/K]
kf=0.03305;  //[W/m*K]
Npr=0.703;
h=P/(A*(T-Tair));
Nnu=(h*d)/kf;
function y=func(x)
    y=Nnu-0.3-((0.62*(x^(1/2))*(Npr^(1/3)))
       /((1+((0.4/Npr)^(2/3)))^(1/4)))*((1+((x
       /(2.82*(10^5)))^(5/8)))^(4/5));
endfunction
// on solving the above function for x by using some
    root solver technique like Newton raphson method
    , we get
x=107.7;
// or
Nre=107.7;
```

```
y=func(x);
Um=(Nre*mu)/(d*p);
printf("\n\n The velocity is \n Um = %f m/sec = %f
    ft/sec",Um,Um*3.28);
```

Scilab code Exa 12.17 single cyclinder heat transfer

```
clc;
warning("off");
printf("\n\n example12.17 - pg630");
// given
dt=0.75;
St=1.5*dt;
Sl=3*dt;
Lw=1;   //[m]
N=12;
Stotalarea=N*(St/12)*Lw;
Sminarea=N*((St-dt)/12)*Lw*0.3048;
// properties of air at 293.15 K
p=1.204;   //[kg/m^3]
mu=1.818*10^-5;   //[kg/m*s]
cp=1005;   //[J*kg/K];
k=0.02560;   //[J/s*m*K]
Npr=(cp*mu)/k;
U_inf=7;   //[m/sec]
Umax=U_inf*(St/(St-dt));
w=p*Umax*Sminarea;
C_tubes=0.05983;   //[m^2/m] - circumference of the
    tubes
N_tubes=96;
Atubes=N_tubes*C_tubes*Lw;
Tw=328.15;   //[K]
Tinf=293.15; //[K]
Tin=293.15;   //[K]
Tout=293.15;   //[K]
```

```
u=100;
while u>10^-1
    T=(Tin+Tout)/2
    Told=Tout;
    p=-(0.208*(10^-3))+(353.044/T);
    mu=-(9.810*(10^-6))+(1.6347*(10^-6)*(T^(1/2)));
    cp=989.85+(0.05*T);
    k=0.003975+7.378*(10^-5)*T;
    Npr=(cp*mu)/k;
    dt=0.75*0.0254;
    Gmax=w/Sminarea;
    Nre=(dt*Gmax)/mu;
    h=0.27*(k/dt)*(Npr^0.36)*(Nre^0.63);
    h=h*0.98;
    deltaT=(h*Atubes*(Tw-Tinf))/(w*cp);
    Tout=Tin+deltaT;
    u=abs(Tout-Told);
end
T=(Tin+Tout)/2
p=-(0.208*(10^-3))+(353.044/T);
mu=-(9.810*(10^-6))+(1.6347*(10^-6)*(T^(1/2)));
dt=0.75;
dv=(4*(St*Sl-(%pi*(dt^2)*(1/4))))/(%pi*dt)
   *(0.09010/3.547);
de=dv;
Nre=(dv*24.72)/mu;
dv=dv/(0.09010/3.547);
ftb=1.92*(Nre^(-0.145));
Zt=Sl;
Ltb=8*Sl;
deltap=(ftb*(24.72^2))/(2*p*(dv/Ltb)*((St/dv)^0.4)
   *((St/Zt)^0.6));
printf("\n\n -deltap = %f kg/m*s = %f N/m^2 = %f
   psia",deltap,deltap,deltap*(0.1614/1113));
```

Chapter 13

unsteady state transport

Scilab code Exa 13.1 heat transfer with negligible internal resistance

```
clc;
warning("off");
printf("\n\n example13.1 - pg651");
// given
h=12;  //[W/m^2*K] - heat transfer coefficeint
k=400;  //[W/m*K] - thermal conductivity
// (a) for sphere
r=5*10^-2;  //[m] - radius of copper sphere
Lc=((4*%pi*((r)^3))/3)/(4*%pi*((r)^2));
Nbi=h*Lc*(1/k);
printf("\n\n (a) The biot no. is \n Nbi = %e",Nbi);
// (b) for cyclinder
r=0.05;  //[m] - radius of cyclinder
L=0.3;  //[m] - height of cyclinder
Lc=(%pi*((r)^2)*L)/(2*%pi*r*L);
Nbi=h*Lc*(1/k);
printf("\n\n (b) The biot no. is \n Nbi = %e",Nbi);
// (c) for a long square rod
L=.4;  //[m] - length of copper rod
r=0.05;  //[m] - radius of a cyclinder havimg same cross sectional area as that of square
```

```
x=((%pi*r^2)^(1/2));
Lc=((x^2)*L)/(4*x*L);
Nbi=h*Lc*(1/k);
printf("\n\n (c) The biot no. is \n Nbi = %e",Nbi);
```

Scilab code Exa 13.6 generalized chart solution for finite slab and cyclinder

```
clc;
warning("off");
printf("\n\n example13_6 - pg684");
// given
d=1*0.0254;   //[m]
Lr=d/2;   //[m];
Lz=(1.2/2)*(0.0254);
x=Lz;
r=Lr;
k=0.481;
h=20;
mr=k/(h*Lr);
mz=k/(h*Lz);
nr=r/Lr;
nz=x/Lz;
t=1.2;   //[sec]
alpha=1.454*10^-4;
Xr=(alpha*t)/(Lr^2);
Xz=(alpha*t)/(Lz^2);
// using the above value of m,n,X the value for Ycz
   and Ycr from fig 13.14 is
Ycr=0.42;
Ycz=0.75;
Yc=Ycr*Ycz;
T_infinity=400;   //[K]
To=295;
Tc=T_infinity-(Yc*(T_infinity-To));
```

```
printf("\n\n The temperature t the centre is \n Tc =
    %f K",Tc);
```

Scilab code Exa 13.7 generalized chart solution for finite slab and cyclinder

```
clc;
warning("off");
printf("\n\n example13_7 - pg684");
// given
T_x0=300;   //[K]
Tw=400;   //[K]
L=0.013;   //[m]
alpha=2.476*(10^-5);   //[m^/sec]
h=600;   //[W/m^2*K]
pcp=3.393*(10^6);   //[J/m^3*K]
L=0.013;   //[m]
deltax=L/10;
betaa=0.5;
deltat=0.03;
deltat=betaa*((deltax)^2)*(1/alpha);
T_infinity=400;   //[K]
// to be sure that the solution is stable, it is
    customary to truncate this number
deltat=0.03;   //[sec]
// betaa=alpha*deltat*((1/deltax)^2);
    for i=1:11
    Told(i)=300;
end
a=((2*h*deltat)/(pcp*deltax));
b=((2*alpha*deltat)/(pcp*((deltax)^2)));
for j=1:11
Tnew(1)=(T_infinity*0.08162)+(Told(1)
    *(1-0.08162-0.8791))+(Told(2)*0.8791)
for k=1:9
```

```
    Tnew(k+1)=(betaa*Told(k+2))+((1-2*betaa)*(Told(k
        +1)))+(betaa*Told(k));
end
Tnew(11)=((2*betaa)*(Told(10)))
Told=Tnew;
end
disp(Told);
```

Scilab code Exa 13.9 semi infinite slab

```
clc;
warning("off");
printf("\n\n example 13_9 - pg700");
// given
p=2050;   //[kg/m^3] - density of soil
cp=1840;   //[J/kg*K] - heat cpapacity of soil
k=0.52;   //[W/m*K] - thermal conductivity of soil
alpha=0.138*10^-6;   //[m^2/sec]
t=4*30*24*3600;   //[sec] - no. of seconds in 4
    months
Tx=-5;   //[degC]
Tinf=-20;   //[degC]
T0=20;   //[degC]
// from the fig 13.24 the dimensionless distance Z
    is
Z=0.46;
// then the depth is
x=2*((alpha*t)^(1/2))*Z
printf("\n\n the depth is \n x = %f m = %f ft",x,x
    *(3.6/1.10));
```

Scilab code Exa 13.10 cyclinder

```
clc;
warning("off");
printf("\n\n example13.10 - pg701");
// given
d=0.01;   //[m] - diameter of cyclindrical porous
    plug
D=2*10^-9;   //[m^2/sec] - diffusion coefficient
t=60*60;   //[sec]
r=d/2;
m=0;
Ca_inf=0;
Ca_0=10;
X=(D*t)/((r)^2);
// from fig 13.14 the ordinate is
Y=0.7;
Ca_c=Ca_inf-Y*(Ca_inf-Ca_0);
printf("\n\n the concentration of KCL at the centre
    after 60 min is \n Ca = %f kg/m^3",Ca_c);
```

Chapter 14

estimation of transport coefficients

Scilab code Exa 14.1 kinetic theory of gases

```
clc;
warning("off");
printf("\n\n example14.1 - pg726");
// given
T=40+273.15;  //[K] - temperature
P=1;  //[atm] - pressure
sigma=3.711*10^-10;  //[m]
etadivkb=78.6;  //[K]
A=1.16145;
B=0.14874;
C=0.52487;
D=0.77320;
E=2.16178;
F=2.43787;
Tstar=T/(etadivkb);
// using the formula si=(A/(Tstar^B))+(C/exp(D*Tstar))+(E/exp(F*Tstar)
si=(A/(Tstar^B))+(C/exp(D*Tstar))+(E/exp(F*Tstar));
M=28.966;  //[kg/mole] - molecular weight
```

```
// using the formula mu=(2.6693*(10^-26))*(((M*T)^(1/2))/((sigma^2)*si))
mu=(2.6693*(10^-26))*(((M*T)^(1/2))/((sigma^2)*si));
printf("\n\n The viscosity of air is \n mu=%eNs/m^2=%fcp",mu,mu*10^3);
```

Scilab code Exa 14.2 non uniform gas theory

```
clc;
warning("off");
printf("\n\n example14.2.sce - pg726");
T=40+273.15;  //[K] - temperature
P=1;  //[atm] - pressure
// thermal conductivit of air
sigma=3.711*10^-10;  //[m]
etadivkb=78.6;  //[K]
A=1.16145;
B=0.14874;
C=0.52487;
D=0.77320;
E=2.16178;
F=2.43787;
Tstar=T/(etadivkb);
// using the formula si=(A/(Tstar^B))+(C/exp(D*Tstar))+(E/exp(F*Tstar)
si=(A/(Tstar^B))+(C/exp(D*Tstar))+(E/exp(F*Tstar));
// using the formula K=(8.3224*(10^-22))*(((T/M)^(1/2))/((sigma^2)*si))
M=28.966;  //[kg/mole] - molecular weight of air
k=(8.3224*(10^-22))*(((T/M)^(1/2))/((sigma^2)*si));
printf("\n\n Thermal conductivity of air is \n k=%fW/m*K",k);
printf("\n\n Agreement between this value and original value is p[oor;the Chapman-Enskog theory is in erreo when applied to thermal conductivity
```

```
of polyatomic gases");
// thermal conductivity of argon
sigma=3.542*10^-10;  // [m]
etadivkb=93.3;  // [K]
A=1.16145;
B=0.14874;
C=0.52487;
D=0.77320;
E=2.16178;
F=2.43787;
Tstar=T/(etadivkb);
// using the formula si=(A/(Tstar^B))+(C/exp(D*Tstar))+(E/exp(F*Tstar)
si=(A/(Tstar^B))+(C/exp(D*Tstar))+(E/exp(F*Tstar));
// using the formula K=(8.3224*(10^-22))*(((T/M)^(1/2))/((sigma^2)*si))
M=39.948;  // [kg/mole] - molecular weight of argon
k=(8.3224*(10^-22))*(((T/M)^(1/2))/((sigma^2)*si));
printf("\n\n Thermal conductivity of argon is \n k=%fW/m*K",k);
printf("\n\n The thermal conductivity from Chapman-Enskog theory agrees closely with the experimental value of 0.0185; note that argon is a monoatomic gas");
```

Scilab code Exa 14.3 non uniform gas theory

```
clc;
warning("off");
printf("\n\n example14.3 - pg727");
T=40+273.15;  // [K] - temperature
P=1;  // [atm] - pressure
Cp=1005;  // [J/kg*K] - heat capacity
M=28.966;  // [kg/mole] - molecular weight
R=8314.3;  // [atm*m^3/K*mole] - gas constant
```

```
// using the formula Cv=Cp-R/M
Cv=Cp-R/M;
y=Cp/Cv;
mu=19.11*10^-6;  //[kg/m*sec] - viscosity of air
// using the original Eucken correlation
k_original=mu*(Cp+(5/4)*(R/M));
printf("\n\n From the original Eucken correlation\n k=%fW/m*K",k_original);
// using the modified Eucken correlation
k_modified=mu*(1.32*(Cp/y)+(1.4728*10^4)/M);
printf("\n\n From the modified Eucken correlation \n k=%fW/m*K",k_modified);
printf("\n\n As discussed , the value from the modified Eucken equation is highre than the experimental value(0.02709) , and the value predicted by the original Eucken equation is lower than the experimental value , each being about 3 percent different , in this case");
```

Scilab code Exa 14.4 non uniform gas theory

```
clc;
warning("off");
printf("\n\n example14.4 - pg728");
// given
D=7.66*10^-5;  //[m^2/sec] - diffusion coefficient of the helium nitrogen
P=1;  //[atm] - pressure
// (a) using the Chapman-Enskog
T(1)=323;  //[K]
T(2)=413;  //[K]
T(3)=600;  //[K]
T(4)=900;  //[K]
T(5)=1200;  //[K]
Ma=4.0026;
```

```
sigma_a=2.551*10^-10;   //[m]
etaabykb=10.22;   //[K]
Mb=28.016;
sigma_b=3.798*10^-10;   //[m]
etabbykb=71.4;   //[K]
sigma_ab=(1/2)*(sigma_a+sigma_b);
etaabbykb=(etaabykb*etabbykb)^(1/2);
Tstar=T/(etaabbykb);
siD=[0.7205;0.6929;0.6535;0.6134;0.5865];
patm=1;
// using the formula Dab=1.8583*10^-27*(((T^3)*((1/Ma)+(1/Mb)))^(1/2))/(patm*sigma_ab*siD)
Dab=(1.8583*(10^-(27))*(((T^3)*((1/Ma)+(1/Mb)))^(1/2)))/(patm*(sigma_ab^(2))*siD)
printf("\n\n (a)");
for i=1:5;
    printf("\n at T=%fK;Dab=%em^2/sec",T(i),Dab(i));
end
// (b) using experimental diffusion coefficient and Chapman-Enskog equation
for i=1:4
    D(i+1)=D(1)*((T(i+1)/T(1))^(3/2))*(siD(1)/(siD(i+1)));
end
printf("\n\n (b)");
for i=1:5;
    printf("\n at T=%fK;Dab=%em^2/sec",T(i),Dab(i));
end
// (c)
for i=1:4
    Dab(i+1)=D(1)*(T(i+1)/T(1))^(1.75);
end
printf("\n\n (c)");
for i=1:5;
    printf("\n at T=%fK;Dab=%em^2/sec",T(i),Dab(i));
end
```

Scilab code Exa 14.5 non uniform gas theory

```
clc;
warning("off");
printf("\n\n example14.5 - pg730");
// given
T=323;  //[K] - temperature
P=1;  //[atm] - pressure
Dab_experimental=7.7*10^-6;  //[m^2/sec]
DPM_A=1.9;  // dipole moment of methyl chloride
DPM_B=1.6;  // dipole moment of sulphur dioxide
Vb_A=5.06*10^-2;  // liquid molar volume of methyl chloride
Vb_B=4.38*10^-2
Tb_A=249;  // normal boiling point of methyl chloride
Tb_B=263;  // normal boiling point of sulphur dioxide
del_A=((1.94)*(DPM_A)^2)/(Vb_A*Tb_A);
del_B=((1.94)*(DPM_B)^2)/(Vb_B*Tb_B);
del_AB=(del_A*del_B)^(1/2);
sigma_A=(1.166*10^-9)*(((Vb_A)/(1+1.3*(del_A)^2))^(1/3));
sigma_B=(1.166*10^-9)*(((Vb_B)/(1+1.3*(del_B)^2))^(1/3));
etaabykb=(1.18)*(1+1.3*(del_A^2))*(Tb_A);
etabbykb=(1.18)*(1+1.3*(del_B^2))*(Tb_B);
sigma_AB=(1/2)*(sigma_A+sigma_B);
etaabbykb=(etaabykb*etabbykb)^(1/2);
Tstar=T/(etaabbykb);
sigmaDnonpolar=1.602;
sigmaDpolar=sigmaDnonpolar+(0.19*(del_AB^2))/Tstar;
patm=1;
Ma=50.488;  //[kg/mole] - molecular weight of methyl
```

```
chloride
Mb=64.063;  //[kg/mole] - molecular weight of sulphur dioxide
D_AB=(1.8583*(10^-(27))*(((T^3)*((1/Ma)+(1/Mb)))^(1/2)))/(patm*(sigma_AB^(2))*sigmaDpolar);
printf("\n\n Dab=%em^2/sec",D_AB);
printf("\n\n The Chapman Enskog prediction is about 8 percent higher");
```

Scilab code Exa 14.6 empirical correlations for gases

```
clc;
warning("off");
printf("\n\n example14.6 - pg732");
// given
T=423.2;  //[K] - temperature
P=5;  //[atm] - pressure
Ma=4.0026;  //[kg/mole] - molecular weight of helium
Mb=60.09121;  //[kg/mole] - molecular weight of propanol
Dab_experimental=1.352*10^-5;  //[m^2/sec] - experimental value of diffusion coefficient of helium-proponal system
// the diffusion volumes for carbon , hydrogen and oxygen are-
Vc=16.5;
Vh=1.98;
Vo=5.48;
V_A=3*Vc+8*Vh+Vo;
V_B=2.88;
patm=5;
// using the FSG correlation
Dab=(10^-7)*(((T^1.75)*((1/Ma)+(1/Mb))^(1/2))/(patm*((V_A)^(1/3)+(V_B)^(1/3))^2));
printf("\n\n Dab=%em^2/sec",Dab);
```

```
printf("\n\n The FSG correlation agrees to about 2
    percent with the experimental value");
```

Scilab code Exa 14.7 viscosity

```
clc;
warning("off");
printf("\n\n example14.7 - pg736");
// given
beta0=-6.301289;
beta1=1853.374;
clf;
xtitle("Temperature variation of the viscosity of
    water","(1/T)*10^3,K^-1","viscosity,cP");
x=[2.2,0.2,3.8]';
y=[(beta0+beta1*x)];
plot2d(x,y);
// at T=420;
T=420;   //[K]
x=1/T;
y=beta0+beta1*x;
mu=exp(y);
printf("\n\n mu=%fcP",mu);
printf("\n\n The error is seen to be 18 percent.AT
    midrange 320(K), the error is approximately 4
    percent");
```

Scilab code Exa 14.8 thermal conductivity

```
clc;
warning("off");
printf("\n\n example14.8 - pg737");
// given
```

```
M=153.82;   //[kg/mole] - molecular weight of ccl4
T1=349.90;   //[K] - temperature1
T2=293.15;   //[K] - temperature 2
cp1=0.9205;   //[KJ/kg*K] - heat capacity at
    temperature T1
cp2=0.8368;   //[KJ/kg*K] - heat capacity at
    temperature T2
p1=1480;   //[kg/m^3] - density at temperature T1
p2=1590;   //[kg/m^3] - density at temperature T2
Tb=349.90;   //[K] - normal boiling point
pb=1480;   //[kg/m^3] - density at normal boiling
    point
cpb=0.9205;   //[KJ/kg*K] - heat capacity at normal
    boiling point
k1=(1.105/(M^(1/2)))*(cp1/cpb)*((p1/pb)^(4/3))*(Tb/
    T1);
printf("\n\n The estimated thermal conductivity at
    normal boiling point is \n k=%f W*m^-1*K^-1",k1);
k2=(1.105/(M^(1/2)))*(cp2/cpb)*((p2/pb)^(4/3))*(Tb/
    T2);
printf("\n\n The estimated thermal conductivity at
    temperature %f K is \n k=%f W*m^-1*K^-1",T2,k2);
printf("\n\n The estimated value is 3.4 percent
    higher than the experimental value of 0.1029 W*m
    ^-1*K^-1");
```

Scilab code Exa 14.9 diffusion coefficient

```
clc;
warning("off");
printf("\n\n example14.9 - pg743");
// given
T=288;   //[K] - temperature
M1=60.09;   //[kg/mole] - molecular weight of
    proponal
```

```
M2=18.015;  //[kg/mole] - molecular weight of water
mu1=2.6*10^-3;  //[kg/m*sec] - viscosity of proponal
mu2=1.14*10^-3;  //[kg/m*sec] - viscosity of water
Vc=14.8*10^-3;  //[m^3/kmol] - molar volume of carbon
Vh=3.7*10^-3;  //[m^3/kmol] - mlar volume of hydrogen
Vo=7.4*10^-3;  //[m^3/kmol] - molar volume of oxygen
Vp=3*Vc+8*Vh+Vo;  // molar volume of proponal
phi=2.26;  // association factor for diffusion of proponal through water
Dab=(1.17*10^-16*(T)*(phi*M2)^(1/2))/(mu2*(Vp^0.6));
printf("\n\n The diffusion coefficient of proponal through water is \n Dab=%e m^2/sec",Dab);
phi=1.5;  // association factor for diffusion of water through proponal
Vw=2*Vh+Vo;  //[molar volume of water
Dab=(1.17*10^-16*(T)*(phi*M1)^(1/2))/(mu1*(Vw^0.6));
printf("\n\n The diffusion coefficient of water through propanol is \n Dab=%e m^2/sec",Dab);
```

Chapter 15

non newtonial phenomena

Scilab code Exa 15.1 rheological characteristics of material time independant behaviour

```
clc;
warning("off");
printf("\n\n example15.1 - pg760");
// given
r=[10 20 50 100 200 400 600 1000 2000]
tau=[2.2 3.1 4.4 5.8 7.4 9.8 11.1 13.9 17.0]
tau=tau*(10^-4);
clf;
xtitle("basic shear diagram for the fluid","shear
   rate","shear stress");
plot2d("ll",r,tau);
// the data falls nearly on a straight line
// from the graph the slope and the intercept are
slope=0.3841;
intercept=9.17046;
// from the relation tau=K*(-r)^n;
K=exp(intercept);
n=slope
disp(K,"K=",n,"n=");
printf("\n\n The fluid is pseudo plastic , since the
```

```
slope is less than 1 ");
```

Scilab code Exa 15.2 capillary viscometer

```
clc;
warning("off");
printf("\n\n example15.2 - pg774");
// given
a=[651 1361 2086 5089 7575 11140 19270 25030]
tau=[3.71 7.49 11.41 24.08 -35.21 46.25 77.50 96.68]
clf;
xtitle("capillary shear diagram for polyisobutylene L-80 in cyclohexane","pseudoshear rate","wall shear stress");
plot2d("ll",a,tau);
// from the graph
betao=-4.3790154;
beta1=0.8851;
K'=exp(betao);
n'=beta1;
printf("\n\n The final rheological model is \n tauw = %f*(8*Uz,avg/do)^%f",K',n');
```

Scilab code Exa 15.3 capillary viscometer

```
clc;
warning("off");
printf("\n\n example15.3 - pg774");
// given
// from example 15.2
n'=0.8851;
K'=0.01254;
n=n';
```

```
K=K'/((3*n+1)/(4*n));
disp(n,"n=");
printf("\n K = %f N/m^2",K);
```

Scilab code Exa 15.4 capillary viscometer

```
clc;
warning("off");
printf("\n\n example15.4 - pg775");
// given
a=[10 20 50 100 200 400 600 1000 2000];
tau=[2.24 3.10 4.35 5.77 7.50 9.13 11.0 13.52 16.40]
tau=tau*10^-4;
clf;
xtitle("capillary shear diagram for a commercial
    polyethylene melt at 190 degC","pseudoshear rate"
    ,"wall shear stress");
plot2d("ll",a,tau);
// such a plot suggests a second order polynomila
    of the type y=betao+beta1*x+beta2*x^2;
// where y=ln(tauw) and x=ln(8*Uz,avg/do)=ln(a);
// from the graph
betao=8.96694;
beta1=0.48452520;
beta2=0.010923041;
n=beta1+2*beta2*a;
phiw=((3*n+1)/(4*n))*(a);
mu=tau/phiw;
printf("\n\n 8*Uz,avg/do          n          (3*n+1)/(4*n)
          phiw                  mu");
for i=1:9
    printf("\n %f          %f          %f          %f          %f"
        ,a(i),n(i),(3*n(i)+1)/(4*n(i)),phiw(i),mu);
end
```