Scilab Textbook Companion for Engineering Heat Transfer by W. S. Janna[1]

Created by
Sintu Rongpipi
Bachelor of Technology
Chemical Engineering
Indian Institute of Technology Guwahati
College Teacher
Dr. Prakash Kotecha
Cross-Checked by
Mukul R. Kulkarni

September 27, 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: Engineering Heat Transfer

Author: W. S. Janna

Publisher: CRC Press

Edition: 3

Year: 2009

ISBN: 978-1420072020

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

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

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

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

Contents

List of Scilab Codes

List of Figures

Chapter 1

Fundamental Concepts

Scilab code Exa 1.1 The Surface Temperature of firewall

```
clear;
clc;
printf("\t\t\tChapter1_Example1\n\n\n");
// determination of surface temperature on one side of a firewall
k=9.4; // thermal conductivity in [BTU/hr.ft.Rankine]
q=6.3; // heat flux in [BTU/s. sq.ft]
T1=350; // the outside surface temperature of one aide of the wall [ F]
// converting heat flux into BTU/hr  sq.ft
Q=6.3*3600 // [BTU/hr.sq.ft]
printf("\nThe heat flux is %.2f BTU/hr. sq.ft",Q);
dx=0.5; // thickness in [inch]
//converting distance into ft
Dx=0.5/12 // thickness in [ft]
printf("\nThe thickness of firewall is %.2f ft",Dx);
// solving for temeprature T2
T2=T1-(Q*Dx/k); // [ F]
```

```
printf("\nThe required temperature on the other side of the firewall is %.1f degree Fahrenheit",T2);
```

Scilab code Exa 1.2 Thermal Conductivity of Aluminium

```
clear;
clc;
printf("\t\t\tchapter1_example2\n\n\n");
// determination of thermal conductivity of aluminium
k_ss=14.4; // thermal conductivity of stainless steel in [W/m.K]
printf("\nThe thermal conductivity of stainless steel is %.1f W/m.K",k_ss);
dt_ss=40; // [K]
dt_al=8.65; // [K]
dz_ss=1; // [cm]
dz_al=3; // [cm]
k_al=k_ss*dt_ss*dz_al/(dt_al*dz_ss);// thermal conductivity of Al in [W/m.K]
printf("\nThe thermal conductivity of aluminium is %d W/m.K",k_al);
```

Scilab code Exa 1.3 Convective Heat Transfer

```
clear;
clc;
printf("\t\t\tchapter1_example3\n\n\n");
// determination of heat transferred by convection
```

```
h_c=3; // convective coefficient in [BTU/hr.ft^2
A=30*18; // Cross sectional area in ft^2
T_w=140; // Roof surface temperature in degree
    Fahrenheit
T_inf=85; // Ambient temperature in degree
    Fahrenheit
dT= (T_w-T_inf);
Q_c=h_c*A*dT; // Convective heat transfer in BTU/hr
printf("\nThe heat transferred by convection is %d
    BTU/hr",Q_c);
```

Scilab code Exa 1.4 Average Film Conductance

```
clear;
clc;
printf("\t\t\tchapter1_example4\n\n\n");
// determining average film conductance
D=2.43/100; // diameter in meter
L=20/100; // length in meter
A=3.14*D*L; // cross-sectional area in sq.m
cp=4200; // specific heat of water in J/kg.K
T_b2=21.4; // temperature of bulk fluid in degree
    celsius
T_in=20; // temperature of inlet water in degree
    celsius
T_w=75; // temperature of wall in degree celsius
Q=500; // volumetric flow rate in cc/s
density=1000; // density of water in kg/cu.m
m=Q*density/10^6; // mass flowa rate in kg/s
printf("\nThe mass flow rate is %.1f kg/s",m);
// using definition of specific heat and Newton's
    law of cooling
hc=m*cp*(T_b2-T_in)/(A*(T_w-T_in));
```

```
printf("\nThe average film conductance is %d W/sq.m.
    K",hc);
```

Scilab code Exa 1.5 Instantaneous Heat loss by radiation

```
clear;
clc;
printf("\t\t\tchapter1_example5\n\n\n");
// determination of heat loss rate by radiation
W=14; // width in ft
L=30; // length in ft
A=W*L; // area in ft^2
F_12=1; // view factor assumed to be 1
T1=120+460; // driveway surface temperature  in
    degree Rankine
printf("\nThe driveway surface temperature is %d
    degree Rankine",T1);
T2=0; // space temperature assumed to be 0 degree
    Rankine
sigma=0.1714*10^(-8); // value of Stefan-Boltzmann's
     constant in BTU/(hr.ft^2.(degree Rankine)^4)
e=0.9; // surface emissivity
q=sigma*A*e*F_12*((T1)^4-(T2)^4);
printf("\nThe heat loss rate by radiation is %d BTU/
    hr",q);
```

Scilab code Exa 1.6 Radiation thermal conductance

```
clear;
```

```
clc;
printf("\t\t\tchapter1_example6\n\n\n");
// determination of radiation thermal conductance
A=14*30; // area in sq.ft
T1=120+460; // driveway surface temperature in degree Rankine
T2=0; // surface temperature assumed to be 0 degree Rankine
Qr=73320; // heat loss rate in BTU/hr
hr=Qr/(A*(T1-T2)); // radiation thermal conductance in BTU/(hr.ft^2.(degree Rankine)
printf("\nthe radiation thermal conductance is %.2f BTU/(hr. sq.ft.(degree Rankine))",hr);
```

Scilab code Exa 1.7 Thermal Circuit

```
clear;
clc;
printf("\t\t\tchapter1_example7\n\n\n");
// Identification of all resistances and their values
// Estimation of heat transfer per unit area
// Determination of the inside and outside wall temperatures
printf("\n\t\t\tSolution to part (b)\n");
A=1; // assuming A=1 m^2 for convenience
hc1_avg=(5+25)/2; // taking average of extreme values for hc [W/m^2.K]
Rc1=1/(hc1_avg*A); // resistance on left side of wall [K/W]
printf("\nThe resistance on left side of wall is %.3f K/W",Rc1);
k=(0.38+0.52)/2; // thermal conductivity of common
```

```
brick in W/M.k
L=0.1; //10 cm converted into m
Rk=(L/(k*A));// resistance of construction material, assume common brick
printf("\nThe resistance of construction material of wall is %.3f K/W",Rk);
Rc2=Rc1;
printf("\nThe resistance on right side of wall is %.3f K/W",Rc2);
printf("\n\n\t\t\tSolution to part (c)\n");
T_inf1=1000; // temperature of exhaust gases in K
T_inf2=283; // temperature of ambient air in K
q=(T_inf1-T_inf2)/(Rc1+Rk+Rc2); // heat transferred per unit area
printf("\nThe Heat transferred per unit area is %d W = %.3f kW",q,q/1000);
printf("\n\n\t\t\tSolution to part (d)\n");
T_in=T_inf1-Rc1*q; //
T_out=T_inf2+Rc2*q;
printf("\nThe inside wall temperature is %d K",T_in);
printf("\nThe outside wall temperature is %d K",T_out);
```

Scilab code Exa 1.8 Combined Heat Transfer Mechanisms

```
clear;
clc;
printf("\t\t\tchapter1_example8\n\n\n");
// determination of surface temperature
k=0.604; // [BTU/(hr.ft.degree Rankine)]
hc=3; // average value for natural convection in BTU/(hr.ft^2.degree Rankine)
```

```
ew=0.93;
f_wr=1; // shape factor
sigma= 0.1714*10^(-8) // BTU/(hr.ft^2.degree Rankine
    ).
L=4/12; // length in ft
T1=80+460; // temperature of side-walk in degree
    Rankine
T_inf=20+460; // temperature of ambient air in
    degree Rankine
T_r=0; // assuming space temperature to be 0 degree
    Rankine
// LHS of the form a*Tw+b*Tw^4=c
a=((k/L)+hc);
b=(sigma*ew*f_wr);
c=(k*T1/L)+(hc*T_inf)+(sigma*f_wr*ew*T_r^4);
printf("\nRHS=%d",c);
    Tw=[470;480;490;485;484.5];
for i=1:5
    LHS(i)=a*Tw(i)+b*Tw(i)^4;
end
printf("\nSolving by trial and error yields the
    following, where LHS is the left-hand side of the
     equation");
printf("\n\tTw\tLHS");
for i=1:5
    printf("\n\t%.1f\t%d",Tw(i),LHS(i));
end
printf("\nThe Surface temperature is %.1f degree R =
    %.1f degree F",Tw(5),Tw(i)-460);
```

Chapter 2

Steady State Conduction in One Dimension

Scilab code Exa 2.1 Materials in Series

```
clc;
clear;
printf("\t\t\tChapter2_example1\n\n\n");
//  determination of the heat flow through a composite wall
T3=-10; // temperature of inside wall in degree Fahrenheit
T0=70; // temperature of outside wall in degree Fahrenheit
dT=T0-T3; // overall temperature difference
// values of thermal conductivity in BTU/(hr.ft.degree Rankine) from appendix table B3
k1=0.38; // brick masonry
k2=0.02; // glass fibre
k3=0.063; // plywood
dx1=4/12; // thickness of brick layer in ft
dx2=3.5/12; // thickness of glass fibre layer in ft
```

```
dx3=0.5/12; // thickness of plywood layer in ft
A=1; // cross sectional area taken as 1 ft^2
R1=dx1/(k1*A); // resistance of brick layer in (hr.
    degree Rankine)/BTU
R2=dx2/(k2*A); // resistance of glass fibre layer in
     (hr.degree Rankine)/BTU
R3=dx3/(k3*A); // resistance of plywood layer in (hr
    .degree Rankine)/BTU
printf("\nResistance of brick layer is %.3f (hr.
    degree Rankine)/BTU",R1);
printf("\nResistance of glass fibre layer is %.1f (
    hr.degree Rankine)/BTU",R2);
printf("\nResistance of plywood layer is %.3f (hr.
    degree Rankine)/BTU",R3);
qx=(T0-T3)/(R1+R2+R3);
printf("\nHeat transfer through the composite wall
    is %.2f BTU/hr",qx);
```

Scilab code Exa 2.2 Materials in Parallel

```
clc;
clear;
printf("\t\t\tChapter2_example2\n\n\n");
// determination of heat transfer through composite
    wall for materials in parallel
// values of thermal conductivities in W/(m.K) from
    appendix table B3
k1=0.45;// thermal conductivity of brick
k2a=0.15; // thermal conductivity of pine
k3=0.814; // thermal conductivity of plaster board
k2b=0.025; // thermal conductivity of air from
    appendix table D1
// Areas needed fpor evaluating heat transfer in sq.
```

```
m
A1=0.41*3; // cross sectional area of brick layer
A2a=0.038*3; // cross sectional area of wall stud
A2b=(41-3.8)*0.01*3; // cross sectional area of air layer
A3=0.41*3; // cross sectional area of plastic layer
dx1=0.1; // thickness of brick layer in m
dx2=0.089; // thickness of wall stud and air layer in m
dx3=0.013; // thickness of plastic layer in m
R1=dx1/(k1*A1); // Resistance of brick layer in K/W
R2=dx2/(k2a*A2a+k2b*A2b); // Resistance of wall stud and air layer in K/W
R3=dx3/(k3*A3); // Resistance of plastic layer in K/W
printf("\nResistance of brick layer is %.3f K/W",R1);
printf("\nResistance of wall stud and air layer is %.2f K/W",R2);
printf("\nResistance of plastic layer is %.3f K/W",R3);
T1=25; // temperature of inside wall in degree celsius
T0=0; // temperature of outside wall in degree celsius
qx=(T1-T0)/(R1+R2+R3); // heat transfer through the composite wall in W
printf("\nHeat transfer through the composite wall is %.1f W",qx);
```

Scilab code Exa 2.3 Overall Heat Transfer Coefficient

```
clc;
```

```
clear;
printf("\t\t\tChapter2_example3\n\n\n");
// determination of heat transfer rate and overall
    heat transfer coefficient
k1=24.8; // thermal conductivity of 1C steel in BTU
    /(hr.ft.degree Rankine)from appendix table B2
k2=0.02; // thermal conductivity of styrofoam steel
    in BTU/(hr.ft.degree Rankine)
k3=0.09; // thermal conductivity of fibreglass in
    BTU/(hr.ft.degree Rankine)
hc1=0.79; // convection coefficient between the air
    and the vertical steel wall in BTU/(hr.ft^2.
    degree Rankine)
hc2=150; // the convection coefficient between the
    ice water and the fiberglass
A=1; // calculation based on per square foot
dx1=0.04/12; // thickness of steel in ft
dx2=0.75/12; // thickness of styrofoam in ft
dx3=0.25/12; // thickness of fiberglass in ft
// Resistances in (degree Fahrenheit.hr)/BTU
disp('Resistances in (degree Fahrenheit.hr)/BTU:');
Rc1=1/(hc1*A); // Resistance from air to sheet metal
printf("\nResistance from air to sheet metal: %.3f
    degree F.hr/BTU",Rc1);
Rk1=dx1/(k1*A); // Resistance of steel layer
printf("\nResistance of steel layer: %.4f degree F.
    hr/BTU",Rk1);
Rk2=dx2/(k2*A); // Resistance of styrofoam layer
printf("\nResistance of styrofoam layer: %.3f degree
     F.hr/BTU",Rk2);
Rk3=dx3/(k3*A); // Resistance of fiberglass layer
printf("\nResistance of fiberglass layer: %.3f
    degree F.hr/BTU",Rk3);
Rc2=1/(hc2*A); // Resistance from ice water to
    fiberglass
printf("\nResistance from ice water to fiberglass: %
    .4f degree F.hr/BTU",Rc2);
U=1/(Rc1+Rk1+Rk2+Rk3+Rc2); // overall heat transfer
```

```
   coefficient in BTU/(hr.ft^2.degree Rankine)
printf("\nThe overall heat transfer coefficient is %
   .3f BTU/(hr. sq.ft.degree Rankine)",U);
T_inf1=90;// temperature of air in degree F
T_inf2=32; // temperature of mixture of ice and
   water in degree F
q=U*A*(T_inf1-T_inf2);
printf("\nThe heat transfer rate is %.1f BTU/hr",q);
```

Scilab code Exa 2.4 Pipe and Tube Specifications

```
clc;
clear;
printf("\t\t\tChapter2_example4\n\n\n");
// determination of the heat transfer through the
   pipe wall per unit length of pipe.
k=14.4; // thermal conductivity of 304 stainless
   steel in W/(m.K) from appendix table B2
// dimensions of steel pipes in cm from appendix
   table F1
D2=32.39;
D1=29.53;
T1=40;
T2=38;
Qr_per_length=(2*3.14*k)*(T1-T2)/log(D2/D1);
format(6);
printf("\nThe heat transfer through the pipe wall
   per unit length of pipe is %.1f W/m = %.2f kW/m",
   Qr_per_length,Qr_per_length/1000);
```

Scilab code Exa 2.5 Materials in Series in tubular arrangement

```
clc;
clear;
printf("\t\t\tChapter2_example5\n\n\n");
// determination of the heat gain per unit length
k1=231; // thermal conductivity of copper in BTU/(hr
    .ft.degree Rankine)from appendix table B1
k2=0.02; // thermal conductivity of insuLtion in BTU
    /(hr.ft.degree Rankine)
// Specifications of 1 standard type M copper tubing
     from appendix table F2 are as follows
D2=1.125/12; // outer diameter in ft
D1=0.08792; // inner diameter in ft
R2=D2/2;// outer radius
printf("\nOuter radius is %.4f ft",R2);
R1=D1/2; // inner radius
printf("\nOuter radius is %.3f ft",R1);
t=0.5/12; // wall thickness of insulation in ft
R3=R2+t;
printf("\nRadius including thickness is %.4f ft",R3)
    ;
LRk1=(log(R2/R1))/(2*3.14*k1); // product of length
    and copper layer resistance
printf("\nProduct of length and copper layer
    resistance is: %.1e",LRk1);
LRk2=(log(R3/R2))/(2*3.14*k2); // product of length
    and insulation layer resistance
printf("\nProduct of length and insulation layer
    resistance is: %.2f",LRk2);
T1=40; // temperature of inside wall of tubing in
    degree fahrenheit
```

```
T3=70; // temperature of surface temperature of
    insulation degree fahrenheit
q_per_L=(T1-T3)/(LRk1+LRk2); // heat transferred per
     unit length in BTU/(hr.ft)
printf("\nThe heat transferred per unit length is %
    .2f BTU/(hr.ft)",q_per_L);
```

Scilab code Exa 2.6 Overall Heat Transfer Coefficient in pipe

```
clc;
clear;
printf("\t\t\tChapter2_example6\n\n\n");
// Determination of the overall heat transfer
    coefficient
k12=24.8; // thermal conductivity of 1C steel in BTU
    /(hr.ft.degree Rankine)from appendix table B2
k23=.023; // // thermal conductivity of glass wool
    insulation in BTU/(hr.ft.degree Rankine)from
    appendix table B3
// Specifications of 6 nominal, schedule 40 pipe (no
     schedule was specified , so the standard is
    assumed) from appendix table F1 are as follows
D2=6.625/12; // outer diameter in ft
D1=0.5054; // inner diameter in ft
printf("\nOuter diameter is %.3f ft",D2);
printf("\nInner diameter is %.4f ft",D1);
t=2/12; // wall thickness of insulation in ft
D3=D2+t;
printf("\nDiameter including thickness is %.5f ft",
    D3);
hc1=12; // convection coefficient between the air
    and the pipe wall in BTU/(hr. sq.ft.degree
    Rankine).
```

```
hc2=1.5; // convection coefficient between the glass
    wool and the ambient air in  BTU/(hr. sq.ft.
    degree Rankine).
U=1/((1/hc1)+(D1*log(D2/D1)/k12)+(D1*log(D3/D2)/k23)
    +(D1/(hc2*D3)));
printf("\nOverall heat transfer coefficient is %.3f
    BTU/(hr. sq.ft.degree Fahrenheit)",U);
```

Scilab code Exa 2.7 Thermal Contact Resistance

```
clc;
clear;
printf("\t\t\tChapter2_example7\n\n\n");
// Determination of the thermal contact resistance
k=14.4; // thermal conductivity of 304 stainless
    steel in  W/(m.K)from appendix table B2
T1=543; // temperature in K at point 1
T2=460; // temperature in K at point 2
dT=T1-T2; // temperature difference between point 1
    and 2
dz12=0.035; // distance between thermocouple 1 and 2
    in cm
qz_per_A=k*dT/dz12; // heat flow calculated in W/m^2
    calculated using Fourier's law
printf("\nHeat flow calculated is %.2f kW/sq.m",
    qz_per_A/1000);
dz56=4.45; // distance between thermocouple 5 and 6
    in cm
dz6i=3.81; // distance between thermocouple 6 and
    interface in cm
dz5i=dz56+dz6i; // distance between thermocouple 5
    and interface in cm
T5=374; // temperature in K at point 5
```

```
T6=366; // temperature in K at point 6
T_ial=T5-(dz5i*(T5-T6)/dz56); // temperature of
    aluminium interface in K
printf("\nTemperature of aluminium interface is %.1f
    K",T_ial);
dzi7=2.45; // distance between thermocouple 7 and
    interface in cm
dz78=4.45; // distance between thermocouple 7 and 8
    in cm
dzi8=dzi7+dz78; // distance between thermocouple 8
    and interface in cm
T7=349; // temperature in K at point 7
T8=337; // temperature in K at point 8
T_img=dzi8*(T7-T8)/dz78+T8; // temperature of
    magnesium interface in K
printf("\nTemperature of magnesium interface is %.1f
    K",T_img);
Rtc=(T_ial-T_img)/qz_per_A;
printf("\nThe required thermal contact resistance is
    %.2e K. sq.m/W",Rtc);
```

Scilab code Exa 2.8 Analysis of a Pin Fin

```
clc;
clear;
printf("\t\t\tChapter2_example8\n\n\n");
// determination of temperature profile, heat
    transferred, efficiency, effectiveness.
printf("\n\t\t\tSolution to part (a)");
k=24.8; // thermal conductivity of 1C steel in BTU/(
    hr.ft.degree Rankine)from appendix table B2
```

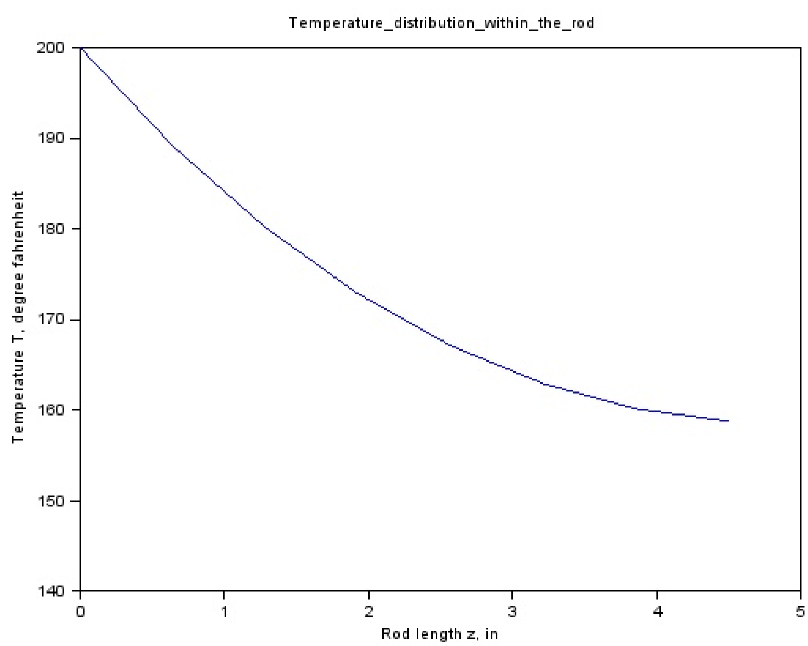

Figure 2.1: Analysis of a Pin Fin

```
D=(5/16)/12; // diameter of the rod in ft
P=(3.14*D); // Circumference of the rod in ft
printf("\nThe perimeter is %.4f ft",P);
A=(3.14/4)*D^2; // Cross sectional area of the rod
    in sq.ft
printf("\nThe Cross sectional area is %.6f sq.ft",A)
    ;
hc=1; // assuming the convective heat transfer
    coefficient as 1 BTU/(hr. sq.ft. degree Rankine)
m=sqrt(hc*P/(k*A));
printf("\nThe value of parameter m is: %.3f/ft",m);
L=(9/2)/12; // length of rod in ft
// using the equation (T-T_inf)/(T_w-T_inf)=(cosh[m(
    L-z)])/(cosh(mL)) for temperature profile
T_inf=70;
T_w=200;
dT=T_w-T_inf;
const=dT/cosh(m*L);
printf("\nThe temperature profile is:\t");
printf("T=%d+%.2fcosh[%.3f(%.3f-z)]",T_inf,const,m,L
    );
z=0:.05:L;
T=T_inf+const*cosh(m*(L-z));
x=linspace(0,4.5,8);
plot(x,T);
a=gca();
a.data_bounds=[0,140;5,200];
newticks=a.x_ticks;
newticks(2)=[0;1;2;3;4;5];
newticks(3)=['0';'1';'2';'3';'4';'5'];
a.x_ticks=newticks;
newticks1=a.y_ticks;
newticks1(2)=[140;150;160;170;180;190;200];
newticks1(3)=['140';'150';'160';'170';'180';'190';'
    200'];
a.y_ticks=newticks1;
xlabel('Rod length z, in');
ylabel('Temperature T, degree fahrenheit');
```

```
title('Temperature_distribution_within_the_rod');
printf("\n\n\t\t\tSolution to part (b)\n");
// the heat transferred can be calculated using the
    equation qz=k*A*m*(T_w-T_inf)*tanh(m*L)
qz=k*A*m*dT*tanh(m*L);
printf("\nThe heat transferred is %.2f BTU/hr",qz);
printf("\n\n\t\t\tSolution to part (c)\n");
mL=m*L;
printf("\nThe value of mL is: %.3f",mL);
efficiency=0.78;
printf("\nThe efficiency found from the graph in
    figure 2.30 is: %.2f",efficiency);
printf('\n\n\t\t\tSolution to part (d)\n');
// the effectiveness can be found using the equation
    effectiveness=sqrt(k*P/h*A)*tanh(mL)
effectiveness=sqrt(k*P/(hc*A))*tanh(mL);
printf("\nThe effectiveness is found to be: %.1f",
    effectiveness);
```

Scilab code Exa 2.9 Corrected Length Solution

```
clc;
clear;
printf("\t\t\tChapter2_example9\n\n\n");
// determination of heat transferred
k=136; // thermal conductivity of aluminium in BTU/(
    hr.ft.degree Rankine)from appendix table B1
L=9/(8*12);
W=9/(4*12);
delta=1/(32*12);
printf("\nLength=%.5f ft, Width=%.4f ft, Delta=%.6f
    ft",L,W,delta);
hc=0.8; // the convective heat transfer coefficient
```

```
    estimated as 1 BTU/(hr.ft^2. degree Rankine)
T_w=1000;// the root temperature in degree
    fahrenheit
T_inf=90; // the ambient temperature in degree
    fahrenheit
m=sqrt(hc/(k*delta));
printf("\nThe value of m is %.3f",m);
P=2*W;
A=2*delta*W;
printf("\n\t\t\tSolution to part (a)\n");
qz1=sqrt(hc*P*k*A)*(T_w-T_inf)*(sinh(m*L)+(hc/(m*k)*
    cosh(m*L)))/(cosh(m*L)+(hc/(m*k)*sinh(m*L)));
printf("\nThe heat transferred is %.2f BTU/hr",qz1);
printf("\n\n\t\t\tSolution to part (b)\n");
qz2=sqrt(k*A*hc*P)*(T_w-T_inf)*tanh(m*L);
printf("\nThe heat transferred is %.2f BTU/hr\n",qz2
    );
printf("\n\t\t\tSolution to part (c)\n");
Lc=L+delta;
qz3=k*A*m*(T_w-T_inf)*tanh(m*L*(1+delta/Lc));
printf("\nThe heat transferred is %.2f BTU/hr\n",qz3
    );
```

Scilab code Exa 2.10 Straight fins of triangular profile

```
clc;
clear;
printf("\t\t\tChapter2_example10\n\n\n");
// determination of optimum fin length and heat
    transferred by fin
k=8.32; // thermal conductivity of Type 304
    stainless steel in BTU/(hr.ft.degree Rankine)from
     appendix table B2
```

```
hc=400; // the convective heat transfer coefficient given in BTU/(hr.ft^2. degree Rankine)
printf("\n\t\t\tSolution to part (a)\n");
delta_opt=0.55/(12*2);
// determination of dimension of one fin using the equation delta_opt=0.583*hc*Lc^2/k
Lc=sqrt(delta_opt*k/(0.583*hc));
printf("\nThe optimum length is %.3f in",Lc*12);
printf("\n\n\t\t\tSolution to part (b)\n");
A=Lc*delta_opt;
// determination of parameter for finding out efficiency from graph
parameter=Lc^1.5*sqrt(hc/(k*A));
printf("\nThe parameter value for finding the efficiency is: %.2f",parameter);
efficiency=0.6;
printf("\nThe efficiency found from the graph in figure 2.36 is %.1f", efficiency);
W=1/(2*12);// width in ft
T_w=190; // wall temperature in degree fahrenheit
T_inf=58; // ambient temperature in degree fahrenheit
L=1; // length in ft
delta=W/2;
q_ac=efficiency*hc*2*W*sqrt(L^2+delta^2)*(T_w-T_inf);
printf("\nThe actual heat transferred is %d BTU/hr",q_ac);
```

Scilab code Exa 2.11 Circular fin of rectangular profile

```
clc;
clear;
```

```
printf("\t\t\tChapter2_example11\n\n\n");
// determination of heat transferred and fin effectiveness
printf("\t\t\tSolution to part (a)\n");
//parameters of the problem are
N=9; // number of fins
delta=0.003/2;
L=0.025;
Lc=L+delta;
R=0.219/2;
R2c=R+delta;
R1=R-L;
T_w=260; // root wall temperature in degree celsius
T_inf=27; // ambient temperature in degree celsius
hc=15;
k=52; // thermal conductivity of cast iron in W/(m.K)from appendix table B2
Ap=2*delta*Lc;
As=2*3.14*(R2c^2-R1^2);
radius_ratio=R2c/R1; // for finding efficiency from figure 2.38
variable=Lc^1.5*sqrt(hc/(k*Ap));
printf("\n\nThe value of R2c/R1 is %.2f",radius_ratio);
printf("\n\nThe value of Lc^(3/2)(hc/kAp)^(1/2) is %.2f",variable);
efficiency=0.93; // efficiency from figure 2.38
printf("\n\nThe efficiency of the fin from figure 2.38 is %.2f",efficiency);
qf=N*efficiency*As*hc*(T_w-T_inf);
printf("\n\nThe heat transferred by the nine fins is %.1f w",qf);
Sp=0.0127; // fin spacing
Asw=2*3.14*R1*Sp*N; // exposed surface area
qw=hc*Asw*(T_w-T_inf);
printf("\n\nThe heat transferred by exposed surface of the cylinder is %d W",qw);
q=qf+qw;
```

```
printf("\n\nThe total heat transferred from the
    cylinder is %d W",q)
printf("\n\n\t\t\tSolution to part (b)\n");
H=N*(Sp+2*delta);// height of cylinder
Aso=2*3.14*R1*H; // surface area without fins
qo=hc*Aso*(T_w-T_inf);
printf("\n\nThe Heat transferred without fins is %d
    W",qo)
printf("\n\n\t\t\tSolution to part (c)\n");
effectiveness=q/qo; // effectiveness defined as
    ratio of heat transferred with fins to heat
    transferred without fins
printf("\nThe fin effectiveness is %.2f",
    effectiveness);
```

Chapter 3

Steady State Conduction in Multiple Dimensions

Scilab code Exa 3.1 Flow Net method of solution

```
clc;
clear;
printf("\t\t\tChapter3_example1\n\n\n");
// Determination of the heat-flow rate from one tube
// specifications of 1 standard type K from table F2
OD=0.02858; // outer diameter in m
// from figure 3.11
M=8; // total number of heat-flow lanes
N=6; // number of squares per lane
S_L=M/N; // conduction shape factor
printf("\nThe Conduction shape factor is %.3f",S_L);
k=0.128; // thermal conductivity in W/(m.K) for concrete from appendix table B3
T1=85; // temperature of tube surface
T2=0; // temperature of ground beneath the slab
q_half=k*S_L*(T1-T2);
printf("\nThe heat flow per unit length from one
```

```
half of one tube is %.1f W/m",q_half);
q=2*q_half;
printf("\nThe total heat flow per tube is %.1f W/m",
    q);
```

Scilab code Exa 3.2 Conduction Shape Factor

```
clc;
clear;
printf("\t\t\tChapter3_example2\n\n\n");
//  Determination of the heat transferred from the
    buried pipe per unit length
// shape factor number 8 is selected from table 3.1
// specifications of 10 nominal, schedule 80 pipe
    from table F1
OD=10.74/12; // diameter in ft
R=OD/2;
T1=140;
T2=65;
k=0.072; // thermal conductivity in BTU/(hr-ft.
    degree R)
d=18/12; // distance from centre-line
S_L=(2*%pi)/(acosh(d/R));
q_L=k*S_L*(T1-T2);
printf("\nThe heat transferred from the buried pipe
    per unit length is %.1f BTU/(hr.ft)",q_L);
```

Scilab code Exa 3.3 Heat lost through shape factor method

```
clc;
clear;
printf("\t\t\tChapter3_example3\n\n\n");
// Determination of the heat lost through the walls, using the shape-factor method. (b) Repeat the calculations but neglect the effects of the corners; that is, assume only one-dimensional effects through all the walls.
k = 1.07; // thermal conductivity of silica brick from appendix table B3 in W/(m.K)
// Calculation of total shape factor
// From figure 3.12, for component A
S1_A=0.138*0.138/0.006;
nA=2;
St_A=nA*S1_A; // Total shape factor of component A
printf("\nThe Total shape factor of component A is %.3f ",St_A);
// For component B
S1_B=0.138*0.188/0.006;
nB=4;
St_B=nB*S1_B; // Total shape factor of component B
printf("\nThe Total shape factor of component B is %.3f ",St_B);
// For component C
S3_C=0.15*0.006;
nC=8;
St_C=nC*S3_C; // Total shape factor of component C
printf("\nThe Total shape factor of component C is %.4f ",St_C);
// For component D
S2_D=0.54*0.188;
nD=4;
St_D=nD*S2_D; // Total shape factor of component D
printf("\nThe Total shape factor of component D is %.5f ",St_D);
// For component E
S2_E=0.138*0.54;
nE=8;
```

```
St_E=nE*S2_E; // Total shape factor of component E
printf("\nThe Total shape factor of component E is %.5f ",St_E);
S=St_A+St_B+St_C+St_D+St_E;
printf("\nThe Total shape factor is %.2f",S);
printf("\n\t\t\tSolution to part (a)\n");
T1=550;
T2=30;
q=k*S*(T1-T2);
printf("\nThe heat transferred through the walls of the furnace is %d W = %.1f kW",q,q/1000);
printf("\n\n\t\t\tSolution to part (b)\n");
// Neglecting the effects of the edges and corners, the shape factor for all walls is found as
S=St_A+St_B;
printf("\nNeglecting the effects of the edges and corners, the shape factor for all walls is %.2f",S);
q_1=k*S*(T1-T2);
printf("\nNeglecting the effects of the edges and corners, the heat transferred is %d W = %.1f kW",q_1,q_1/1000);
Error=(q-q_1)/q;
printf("\nThe error introduced by neglecting heat flow through the edges and corners is %.1f percent",Error*100);
```

Scilab code Exa 3.4 Shape factor for given pipe

```
clc;
clear;
printf("\t\t\tChapter3_example4\n\n\n");
// Determination of the conduction shape factor for
```

```
   the underground portion of the configuration
// specifications of 4 nominal, schedule 40 pipe
   from table F1
OD=4.5/12; // diameter in ft
R=OD/2;
// For pipe A
L_A=4.5; // length in ft
// shape factor number 9 is selected from table 3.1
S_A=(2*%pi*L_A)/(log(2*(L_A)/R));
printf("\nThe Shape Factor of pipe A is %.1f",S_A);
// For pipe B
L_B=18; // length in ft
// shape factor number 9 is selected from table 3.1
S_B=(2*%pi*L_B)/(acosh(L_A/R));
printf("\nThe Shape Factor of pipe B is %.1f",S_B);
S=2*S_A+S_B;
printf("\nThe total conduction shape factor for the
   system is %.1f",S);
```

Scilab code Exa 3.5 Numerical Method for temperature distribution

```
clc;
clear;
printf("\t\t\tChapter3_example5\n\n\n");
//  (a) Using the pin-fin equations for the case
   where the exposed tip is assumed insulated, graph
    the temperature distribution existing within the
    rod. (b) Use the numerical formulation of this
   section to obtain the temperature distribution. (
   c) Compare the two models to determine how well
   the numerical results approximate the exact
```

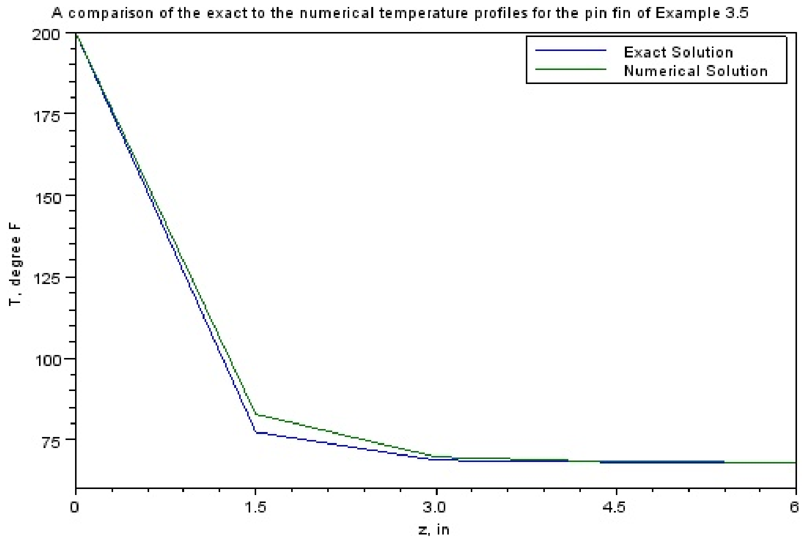

Figure 3.1: Numerical Method for temperature distribution

```
results
h=1.1; // convective coefficient in BTU/(hr.ft^2.degree R)
Tw=200;
T_inf=68; // ambient temperature
printf("\n\t\t\tSolution to part (a)\n");
k=0.47; // thermal conductivity in BTU/(hr.ft.degree R) from table B3
D=0.25/12; // diameter in ft
A=%pi*D^2/4; // cross sectional area in ft^2
P=%pi*D; // perimeter in ft
printf("\nThe cross sectional area is %.3e sq.ft and Perimeter is %.3e ft",A,P);
L=6/12; // length in ft
mL=L*((h*P)/(k*A))^0.5;
printf("\nThe value of Product mL is %.2f",mL);
z=0:1.5:6;
[n m]=size(z);
for i=1:m
```

```
    T(i)=T_inf+(Tw-T_inf)*(cosh(mL*(1-(z(i)/6)))/(cosh(mL)));
end
printf("\n\n\t\t\tSolution to part (b)\n");
d_zeta=1/4;
K=2+(mL*d_zeta)^2;
printf("\nThe value of K is %.4f",K);
T_(5)=T_inf+(Tw-T_inf)*(2/(K^4-4*K^2+2));
T_(4)=T_inf+(Tw-T_inf)*(K/(K^4-4*K^2+2));
T_(3)=T_inf+(Tw-T_inf)*((K^2-1)/(K^4-4*K^2+2));
T_(2)=T_inf+(Tw-T_inf)*((K^3-3*K)/(K^4-4*K^2+2));
T_(1)=200;
printf("\n\nA Comparison of Exact to Numerical Results for the Data of Example 3.5");
printf("\nz,in\tExact (e) T\tNumerical (n) T\tPercent error (e - n)/e");
for i=1:m
err(i)=(T(i)-T_(i))/T(i);
printf("\n%.1f\t%.2f\t\t%.2f\t\t%.2f\n",z(i),T(i),T_(i),err(i)*100);
end
plot(z,T,z,T_);
a=gca();
newticks1=a.x_ticks;
newticks1(2)=[0;1.5;3.0;4.5;6];
newticks1(3)=['0';'1.5';'3.0';'4.5';'6'];
a.x_ticks=newticks1;
newticks2=a.y_ticks;
newticks2(2)=[75;100;125;150;175;200];
newticks2(3)=['75';'100';'125';'150';'175';'200'];
a.y_ticks=newticks2;
title('A comparison of the exact to the numerical temperature profiles for the pin fin of Example 3.5');
xlabel("z, in");
ylabel("T, degree F");
hl=legend(['Exact Solution';'Numerical Solution']);
```

Chapter 4

Unsteady State Heat Conduction

Scilab code Exa 4.1 Response Time of thermocouple junction

```
clc;
clear;
printf("\t\t\tChapter4_example1\n\n\n");
// determination of response time
k=12; // thermal conductivity in BTU/(hr.ft.degree
    Rankine)
c=0.1; // specific heat in BTU/(lbm.degree Rankine)
D=0.025/12; // diameter in ft
density=525; // density in lbm/cu.ft
hc=80; // convective coefficient in BTU/(hr. sq.ft.
    degree Rankine)
T_i=65; // intial temperature in degree fahrenheit
T_inf=140; // fluid temperature in degree fahrenheit
As=3.14*D^2; // surface area in sq.ft
Vs=3.14*D^3/6; // volume in cu.ft
reciprocal_timeconstant=(hc*As)/(density*Vs*c);
printf("\nThe reciprocal of time constant is %.1f /
```

```
hr",reciprocal_timeconstant);
// selecting T=139 degree fahrenheit as T=140 gives an infinite time through the equation (T-T_inf)/(T_i-T_inf)=exp(-hc*As/density*Vs*c)t
T=139;
t=log((T-T_inf)/(T_i-T_inf))/(-reciprocal_timeconstant);
printf('\n\nThe response time of the junction is %.1f s",t*3600);
```

Scilab code Exa 4.2 Lumped Capacitance Approach

```
clc;
clear;
printf("\t\t\tChapter4_example2\n\n\n");
// Determination of temperature of metal and cumulative heat rate
// properties of aluminium from appendix table B1
k=236; // thermal conductivity in W/(m.K)
Cp=896;// specific heat in J/(kg.K)
sp_gr=2.702; // specific gravity
density=2702; // density in kg/cu.m
D=0.05; // Diameter in m
L=0.60; // length in m
hc=550; // unit surface conductance between the metal and the bath in W/(K.sq.m)
Vs=(3.14*D^2*L)/4; // Volume in cu.m
As=(2*3.14*D^2/4)+(3.14*D*L); // surface area in sq.m
printf("\n\nThe volume of cylinder is %.5f cu.m",Vs);
printf("\n\nThe surface area of cylinder is %.3f sq.m",As);
Bi=(hc*Vs)/(k*As); // Biot Number
printf("\n\nThe Biot number is %.3f",Bi);
```

```
// Biot number is less than 1 hence lump capacitance
    equations apply
printf("\n\n\t\t\tSolution to part (a)\n");
T_i=50; // initial temperature in degree celsius
T_inf=2; // temperature of ice water bath in degree
   celsius
t=60; // time=1 minute=60 s
T=T_inf+(T_i-T_inf)*exp(-(hc*As)/(density*Vs*Cp)*t);
printf("\nThe temperature of aluminium is %.1f
   degree celsius",T);
printf("\n\n\t\t\tSolution to part (b)\n");
Q=density*Vs*Cp*(T_inf-T_i)*[1-exp(-(hc*As)/(density
   *Vs*Cp)*t)];
printf("\nThe cumulative heat transferred is %d J =%
   .1f kJ",abs(Q),abs(-Q/1000));
```

Scilab code Exa 4.3 Temperature profile through Infinite plate chart

```
clc;
clear;
printf("\t\t\tChapter4_example3\n\n\n");
hc=30;
L=0.24;
k=1.25;
c=890;
rou=550;
Bi=hc*L/k;
alpha=k/(rou*c);
printf("The value of diffusivity is %.2e sq.m/s",
   alpha);
Tc=150;
```

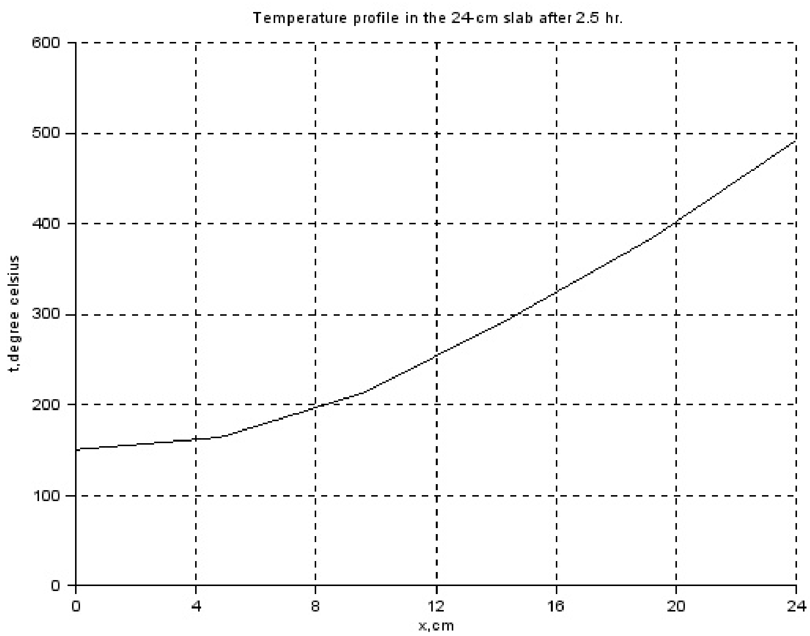

Figure 4.1: Temperature profile through Infinite plate chart

```
T_inf=600;
T_i=50;
printf("\nThe Biot number is %.2f,",Bi);
if Bi<0.1 then
    n=0;
else if  Bi>0.1 then
        n=1;
    end
end
select n
case 0 then
    disp('The Lumped capacity approach is applicable
       ');
case 1 then
    disp('Since value of Biot number is greater than
        0.1, Lumped capacity approach would not give
        accurate results, so figure 4.6 is to be
       used');
    reciprocal_Bi=1/Bi;
    dimensionless_temp=(Tc-T_inf)/(T_i-T_inf);
    Fo=0.4; //the value of Fourier Number from
       figure 4.6(a)
    t=L^2*Fo/alpha;
    printf("The required time is %d s = %.1f hr",t,t
       /3600);
end
// reading values of dimensionless temperature from
   figure 4.6(b) using reciprocal of Biot number
x_per_L=[0 0.2 0.4 0.6 0.8 0.9 1.0];
[n,m]=size(x_per_L);
printf("\nThe choosen values of x/L are: \n");
disp(x_per_L);
printf("\n Values for dimensionless temperature for
   corresponding values of x/L:")
dim_T=[1.0 .97 .86 .68 .48 .36 .24]; // value for
   dimensionless temperature for corresponding value
    of x/L
disp(dim_T);
```

```
printf("the temperature profile with distance is\n");
printf("\tx/L\t\t");
for j=1:m
    printf("%.2f\t",x_per_L(1,j));

end
printf("\n");
printf("(T-T_inf)/T_i-T_inf)\t");
for i=1:m
    printf("%.2f\t",dim_T(i));
end
T=zeros(1,m);
x=zeros(1,m);
for i=1:m
    T(1,i)=dim_T(1,i)*dimensionless_temp*(T_i-T_inf)+T_inf;
    x(1,i)=x_per_L(1,i)*L;
end
printf("\n\tx,cm\t\t");
for i=1:m
    X(1,i)=x(1,i)*100;
    printf("%.1f\t",X(1,i));
end
printf("\nT, degree celsius\t");
for i=1:m
    printf("%d\t",T(1,i));
end
plot2d(X,T,rect=[0,0,24,600]);
a=gca();
newticks=a.x_ticks;
newticks(2)=[0;4;8;12;16;20;24];
newticks(3)=['0';'4';'8';'12';'16';'20';'24'];
a.x_ticks=newticks;
newticks1=a.y_ticks;
newticks1(2)=[0;100;200;300;400;500;600];
newticks1(3)=['0';'100';'200';'300';'400';'500';'600'];
```

```
a.y_ticks=newticks1;
xlabel('x,cm');
ylabel('t,degree celsius');
title('Temperature profile in the 24-cm slab after
    2.5 hr.');
filename='Temperature profile in the 24-cm slab
    after 2.5 hr.';
xgrid(1);
xs2jpg(0,filename);
```

Scilab code Exa 4.4 Dimensionless temperature history of a sphere

```
clc;
clear;
printf("\t\t\tChapter4_example4\n\n\n");
hc=6;
D=0.105;
k=0.431;
c=2000;
rou=998;
Vs=%pi*D^3/6;
As=%pi*D^2;
// calculating Biot Number for lumped capacitance
    approach
Bi_lumped=hc*Vs/(k*As);
printf("\nThe Biot number is %.3f,",Bi_lumped);
alpha=k/(rou*c);
printf("\nThe value of diffusivity is %.2e sq.m/s",
    alpha);
Tc=20;
T_inf=23;
T_i=4;
if Bi_lumped<0.1 then
```

```
    n=0;
else if  Bi_lumped>0.1 then
        n=1;
    end
end
select n
case 0 then
    disp('The Lumped capacity approach is applicable');
case 1 then
    printf("\n\nSince value of Biot number is greater than 0.1,\nLumped capacity approach would not give accurate results, so figure 4.8 is to be used\n");
    // calculating Biot Number for using figure 4.8
    Bi_figure=hc*D/(2*k);
    printf("\nThe Biot Number for using figure 4.8 is %.3f",Bi_figure);
    reciprocal_Bi=1/Bi_figure;
    dimensionless_temp=(Tc-T_inf)/(T_i-T_inf);
    printf("\nThe dimensionless temperature is %.3f",dimensionless_temp);
    Fo=1.05;//The corresponding value of Fourier Number from figure 4.8a
    t=(D/2)^2*Fo/alpha;
    printf("\nThe required time is %.2e s = %.1f hr",t,t/3600);
end
Bi2Fo=Bi_figure^2*Fo;
printf("\nBi^2Fo=%.1e",Bi2Fo);
Dimensionless_HeatFlow=0.7; // The corresponding dimensionless heat flow ratio from figure 4.8c
Q=Dimensionless_HeatFlow*rou*c*Vs*(T_i-T_inf);
printf("\nThe heat transferred is %.3e J",Q);
```

Scilab code Exa 4.5 Estimation of the depth of Freeze line

```
clc;
clear;
printf("\t\t\tChapter4_example5\n\n\n");
hc=6;
D=0.105;
k=0.3;
c=0.41;
sp_gr=2.1;
rou_water=62.4;
alpha=k/(sp_gr*rou_water*c);
printf("\nThe diffusivity of the soil is %.2e sq.ft/
    hr",alpha);
t=3*30*24;
printf("\nTime in hours is %d hr",t);
// Bi_sqrt(Fo) is infinite
T_inf=10;
Ts=10;
T=32;
T_i=70;
dimensionless_temp=(T-T_i)/(T_inf-T_i);
printf("\nThe dimensionless temperature is %.4f",
    dimensionless_temp);
variable_fig4_12=0.38; //The value of x/(2*(alpha*t)
    ^0.5) from figure 4.12
x=2*sqrt(alpha*t)*variable_fig4_12;
printf("\nThe depth of the freeze line in soil is %
    .2f ft",x);
```

Scilab code Exa 4.6 Determination of temperature using kpc product

```
clc;
clear;
printf("\t\t\tChapter4_example6\n\n\n");
// properties of aluminium from appendix table B1
k_al=236;
p_al=2.7*1000;
c_al=896;
// properties of oak from appendix table B3
k_oak=0.19;
p_oak=0.705*1000;
c_oak=2390;
sqrt_kpc_al=sqrt(k_al*p_al*c_al);
printf("\nThe square root of kpc product of
    aluminium is %.2e sq.W.s/(m^4.sq.K)",sqrt_kpc_al)
    ;
kpc_R=4;
T_Li=20;
T_Ri=37.3;
T_al=(T_Li*(sqrt_kpc_al)+T_Ri*sqrt(kpc_R))/(
    sqrt_kpc_al+sqrt(kpc_R));
printf("\nThe temperature of aluminium is felt as %
    .1f degree celsius",T_al);
sqrt_kpc_oak=sqrt(k_oak*p_oak*c_oak);
printf("\nThe square root of kpc product of oak is %
    .2e sq.W.s/(m^4.sq.K)",sqrt_kpc_oak);
T_oak=(T_Li*(sqrt_kpc_oak)+T_Ri*sqrt(kpc_R))/(
    sqrt_kpc_oak+sqrt(kpc_R));
printf("\nThe temperature of oak is felt as %.1f
    degree celsius",T_oak);
if (T_al>T_oak) then
```

```
    printf("\nThe aluminium will feel warmer.");
elseif (T_al<T_oak) then
        printf("\nThe oak will feel warmer.");
else
    printf("\nBoth will be felt equally warm.")
end
```

Scilab code Exa 4.7 Combination of 1D Transient Systems for beer can

```
clc;
clear;
printf("\t\t\tChapter4_example7\n\n\n");
// properties of water at 68 degree fahrenheit from appendix table C11
rou=62.46;
cp=0.9988;
k=0.345;
alpha=k/(rou*cp);
printf("\nThe diffusivity at 68 degree fahrenheit is %.2e sq.ft/hr",alpha);
D=2.5/12;
L=4.75/12;
Vs=%pi*D^2*L/4;
As=(%pi*D*L)+(%pi*D^2)/2;
Lc=Vs/As;
printf("\nThe volume of the can is %.4f cu.ft",Vs);
printf("\nThe surface area of the can is %.3f sq.ft",As);
printf("\nThe characteristic length is %.3f ft",Lc);
hc=1.7;
Bi=hc*Lc/k;
printf("\nThe Biot number is %.3f",Bi);
t=4;
```

```
// for the cylinder solution
Fo_cylinder=alpha*t/(D/2)^2;
Bi_cylinder=hc*(D/2)/k;
printf("\nFor the cylinder , The Fourier number is %
    .2f and Biot Number is %.3f",Fo_cylinder,
    Bi_cylinder);
reciprocal_Bi_cylinder=1/Bi_cylinder;
printf("\nThe reciprocal for Biot number for
    cylinder is %.2f",reciprocal_Bi_cylinder);
dim_T_cylinder=0.175; //The value of dimensionless
    temperature of cylinder from figure 4.7a at
    corresponding values of Fo and 1/Bi
// for the infinite plate solution
Fo_plate=alpha*t/(L/2)^2;
Bi_plate=hc*L/(2*k);
printf("\nFor the infinite plate , The Fourier number
     is %.3f and Biot Number is %.2f",Fo_plate,
    Bi_plate);
reciprocal_Bi_plate=1/Bi_plate;
printf("\nThe reciprocal for Biot number for
    infinite plate is %.2f",reciprocal_Bi_plate);
dim_T_plate=0.55; //The value of dimensionless
    temperature of infinite plate from figure 4.7a at
     corresponding values of Fo and 1/Bi
// Table 4. I , for the short-cylinder problem ,
    indicates that the solution is the product of the
     infinite-cylinder problem (Figure 4.7) and the
    infinite-plate problem (Figure 4.6).
// For short cylinder problem
dim_T_shortcylinder=dim_T_cylinder*dim_T_plate;
printf("\nThe value of dimensionless temperature for
     short cylinder is %.3f ",dim_T_shortcylinder);
T_inf=30;
T_i=72;
Tc=dim_T_shortcylinder*(T_i-T_inf)+T_inf;
printf("\nThe temperature at centre of can is %.1f
    degree celsius",Tc);
dim_Tw_cylinder=0.77; //The dimensionless
```

```
    temperature from figure 4.7b corresponding to the
     value of 1/Bi and r/R=1
dim_Tw_plate=0.65; //The dimensionless temperature
    from figure 4.6b corresponding to the value of 1/
    Bi and x/L=1
dim_Tw_shortcylinder=dim_Tw_cylinder*dim_Tw_plate;
printf("\nThe value of dimensionless temperature  at
     the wall for short cylinder is %.2f ",
    dim_Tw_shortcylinder);
Tw=dim_Tw_shortcylinder*(Tc-T_inf)+T_inf;
printf("\nThe wall temperature is %.1f degree F",Tw)
    ;
```

Scilab code Exa 4.8 Combination of 1D Transient Systems for rectangular bar

```
clc;
clear;
printf("\t\t\tChapter4_example8\n\n\n");
rou=7817;
c=461;
k=14.4;
alpha=.387e-5;
L1=.03;
L2=0.03;
L3=0.04;
x=0.04;
T_i=95;
T_inf=17;
// for infinite plate
L=L1/2;
hc=50;
reciprocal_Bi_plate=k/(hc*L);
```

```
printf("\nThe value of 1/Bi for infinite plate is %
    .1f",reciprocal_Bi_plate);
T=50;
n=1;
t=[3000 1500 700 400 200 300 350];
[n m]=size(t);
// parameter for infinite plate Fourier Number,Fo is
     named as parameter1
for i=1:m
    parameter1(i)=alpha*t(i)/L^2;
// parameters for semi-infinite solid Bi(Fo)^0.5 and
     x/(2*(alpha*t)^0.5) are named as parameter2 and
    parameter3
parameter2(i)=hc*((alpha*t(i))^0.5)/k;
parameter3(i)=x/(2*(alpha*t(i))^0.5);
dim_T_plate=[0.085 0.34 0.55 0.7 0.8 0.8 0.7]; //the
     corresponding values of dimensionless
    temperature for infinite plate from figure 4.6a
dim_T_solid=[0.225 0.14 0.075 0.046 0.02 0.035
    0.042]; // the corresponding values of
    dimensionless temperature for semi-infinite solid
     from figure 4.12
dim_T_bar(i)=dim_T_plate(i)*dim_T_plate(i)*(1-
    dim_T_solid(i));
T(i)=dim_T_plate(i)*dim_T_plate(i)*(1-dim_T_solid(i)
    )*(T_i-T_inf)+T_inf;
end
printf("\nThe Results for different time instances:\
    n");
printf("\n\tInfinite Plate\t\t\t\t\t\tSemi-Infinite
    Solid\t\t\t\tDimensionless Temperature\
    tTemperature");
printf("\ntime t, s\t1/Bi\tFo\t(T-Tinf)/(Ti-Tinf)\
    tBi(Fo)^0.5\tx/(2*(at)^0.5)\t(T-Tinf)/(Ti-Tinf)\t
    (T-Tinf)/(Ti-Tinf)\t\tT");
for i=1:m
    printf("\n%d\t\t%.1f\t%.2f\t\t%.2f\t\t%.3f\t\t%
        .3f\t\t%.3f\t\t\t%.3f\t\t\t\t%.1f",t(i),
```

```
        reciprocal_Bi_plate,parameter1(i),dim_T_plate
        (i),parameter2(i),parameter3(i),dim_T_solid(i
        ),dim_T_bar(i),T(i));
end
```

Scilab code Exa 4.9 Numerical Method for transient conduction system

```
clc;
clear;
printf("\t\t\tChapter4_example9\n\n\n");
rou=.5*1000;
cp=837;
k=0.128;
alpha=0.049e-5;
// let Fo=0.5 and dx=0.05
dt=0.5*(0.05)^2/alpha;
printf("\nThe time increment is %.3f hr",dt/3600);
p=1;
m=6;
A=2*eye(6,6);
n=1;
N=1;
for j=1:n
    for i=1:6
        T(i,j)=20;
    end
end
for n=1:7
    for i=1:4
        B(i+1,n)=T(i+2,n)+T(i,n);
        B(1,n)=T(i+1,n)+200;
```

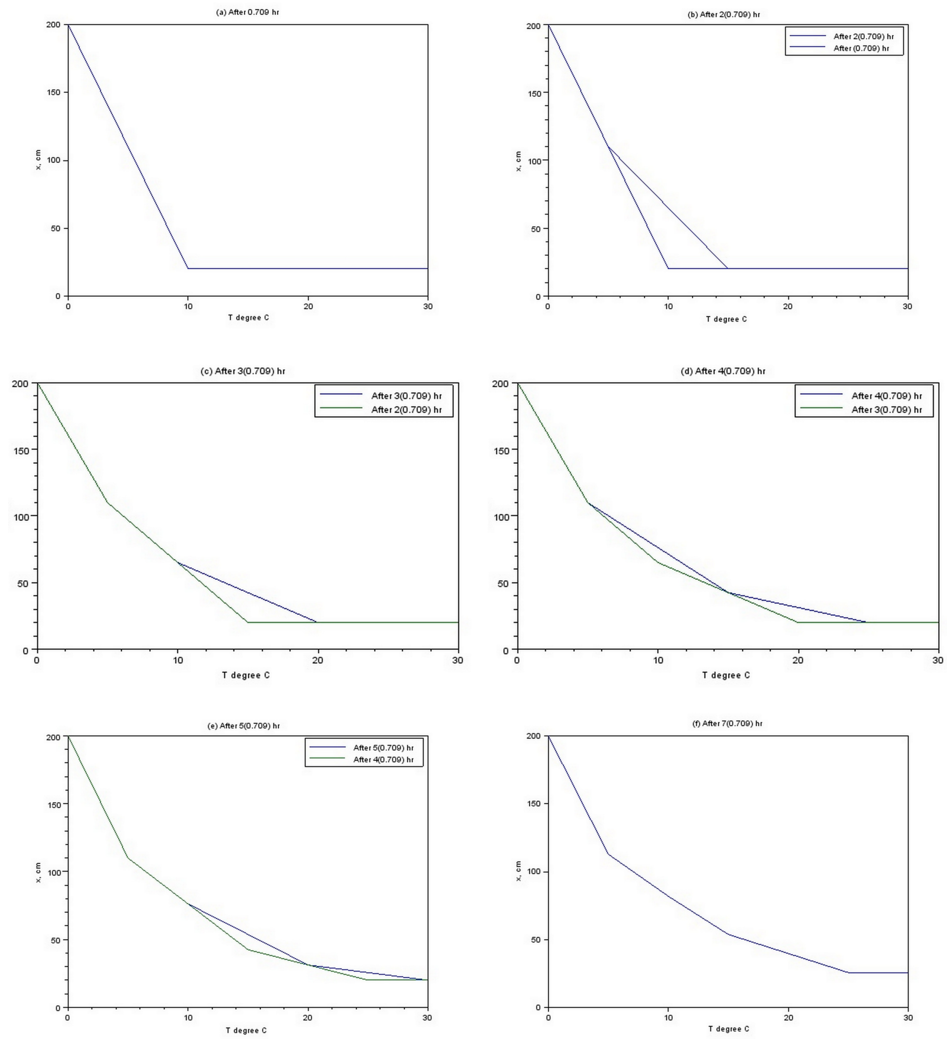

Figure 4.2: Numerical Method for transient conduction system

```
        B(6,n)=2*T(i+1,n);
    end
Temp=inv(A)*B(:,n); // temperature at the different
    points
printf("\nThe temperature at different points after
    %d time interval are:",n);
T(:,n+1)=Temp;
disp(T(:,n+1));
end
time=n*dt;
printf("\nThe required time is %.2f hr",time/3600);
x=0:5:30;
plot(x,[200;T(:,2)]);
a1=gca();
a1.data_bounds=[0,0;30,200];
xtitle('(a) After 0.709 hr','T degree C','x, cm');
newticks=a1.x_ticks;
newticks(2)=[0;10;20;30];
newticks(3)=['0';'10';'20';'30'];
a1.x_ticks=newticks;
newticks1=a1.y_ticks;
newticks1(2)=[0;50;100;150;200];
newticks1(3)=['0';'50';'100';'150';'200'];
a1.y_ticks=newticks1;
plot(x,[200;T(:,3)]);
a2=gca();
hl=legend(['After 2(0.709) hr ';'After (0.709) hr '
    ]);
a2.data_bounds=[0,0;30,200];
xtitle('(b) After 2(0.709) hr ','T degree C','x, cm'
    );
newticks=a2.x_ticks;
newticks(2)=[0;10;20;30];
newticks(3)=['0';'10';'20';'30'];
a2.x_ticks=newticks;
newticks1=a2.y_ticks;
newticks1(2)=[0;50;100;150;200];
newticks1(3)=['0';'50';'100';'150';'200'];
```

```
a2.y_ticks=newticks1;
filename='(b) After 2(0.709) hr ';
clf();
plot(x,[200;T(:,4)],x,[200;T(:,3)]);
a3=gca();
hl=legend(['After 3(0.709) hr ';'After 2(0.709) hr ']);
a3.data_bounds=[0,0;30,200];
xtitle('(c) After 3(0.709) hr ','T degree C','x, cm');
newticks=a3.x_ticks;
newticks(2)=[0;10;20;30];
newticks(3)=['0';'10';'20';'30'];
a3.x_ticks=newticks;
newticks1=a3.y_ticks;
newticks1(2)=[0;50;100;150;200];
newticks1(3)=['0';'50';'100';'150';'200'];
a3.y_ticks=newticks1;
clf();
plot(x,[200;T(:,5)],x,[200;T(:,4)]);
a4=gca();
hl=legend(['After 4(0.709) hr ';'After 3(0.709) hr ']);
a4.data_bounds=[0,0;30,200];
xtitle('(d) After 4(0.709) hr ','T degree C','x, cm');
newticks=a4.x_ticks;
newticks(2)=[0;10;20;30];
newticks(3)=['0';'10';'20';'30'];
a4.x_ticks=newticks;
newticks1=a4.y_ticks;
newticks1(2)=[0;50;100;150;200];
newticks1(3)=['0';'50';'100';'150';'200'];
a4.y_ticks=newticks1;
clf();
plot(x,[200;T(:,6)],x,[200;T(:,5)]);
a5=gca();
hl=legend(['After 5(0.709) hr ';'After 4(0.709) hr '
```

```
]);
a5.data_bounds=[0,0;30,200];
xtitle('(e) After 5(0.709) hr ','T degree C','x, cm');
newticks=a5.x_ticks;
newticks(2)=[0;10;20;30];
newticks(3)=['0';'10';'20';'30'];
a5.x_ticks=newticks;
newticks1=a5.y_ticks;
newticks1(2)=[0;50;100;150;200];
newticks1(3)=['0';'50';'100';'150';'200'];
a5.y_ticks=newticks1;
clf();
plot(x,[200;T(:,7)]);
a6=gca();
a6.data_bounds=[0,0;30,200];
xtitle('(f) After 7(0.709) hr ','T degree C','x, cm');
newticks=a6.x_ticks;
newticks(2)=[0;10;20;30];
newticks(3)=['0';'10';'20';'30'];
a6.x_ticks=newticks;
newticks1=a6.y_ticks;
newticks1(2)=[0;50;100;150;200];
newticks1(3)=['0';'50';'100';'150';'200'];
a6.y_ticks=newticks1;
```

Scilab code Exa 4.10 Graphical Method for transient conduction system

```
clc;
clear;
printf("\t\t\tChapter4_example10\n\n\n");
// determination of time required to cool to a
```

```
   certain temperature
rou=7.817*62.4;
c=.110;
k=8.32;
alpha=0.417e-4;
dx=1/12;
// taking Fo=1
Fo=1;
dt=Fo*dx^2/alpha;
printf("\nThe time increments is %.1f s",dt);
// We have to draw the Saul'ev plot to determine the
    number of time intervals
n=8; //Enter the number of time intervals from
   Saulev plot
time=n*dt;
printf("\nThe required time is %.2f hr",time/3600);
```

Chapter 5

Introduction to Convection

Scilab code Exa 5.1 Heat transferred using specific heat

```
clc;
clear;
printf("\t\t\tChapter5_example1\n\n\n");
// properties of CO at 300K from appendix table D2
Cp=871;
Gamma=1.3;
Cv=Cp/Gamma;
printf("\nThe specific heat at constant volume is %d
    J/(kg.K)",Cv);
dT=20;
m=5;
Qp=m*Cp*dT;
Qv=m*Cv*dT;
printf("\n The heat required at constant pressure is
    %.1f kJ",Qp/1000);
printf("\nThe heat required at constant volume is %d
    kJ",Qv/1000);
```

Scilab code Exa 5.2 Volumetric thermal expansion coefficient

```
clc;
clear;
printf("\t\t\tChapter5_example2\n\n\n");
// properties of Freon-12 from appendix table C3
T1_Fr=-50;
T2_Fr=-40;
rou1_Fr=1.546*1000;
rou2_Fr=1.518*1000;
beta_Fr=-(rou1_Fr-rou2_Fr)/(rou1_Fr*(T1_Fr-T2_Fr));
printf("\nThe volumetric thermal expansion
    coefficient calculated for Freon-12 is %.3e /K",
    beta_Fr);
beta_acc_Fr=2.63e-3; // the accurate value of
    volumetric thermal expansion coefficient for
    Freon-12
error_Fr=(beta_acc_Fr-beta_Fr)*100/beta_acc_Fr;
printf("\nThe error introduced in the case of Freon
    -12 is %d percent",error_Fr);
// properties of helium from appendix table D3
T1_He=366;
T2_He=477;
rou1_He=0.13280;
rou2_He=0.10204;
beta_He=-(rou1_He-rou2_He)/(rou1_He*(T1_He-T2_He));
printf("\nThe volumetric thermal expansion
    coefficient calculated for Freon-12 is %.3e /K",
    beta_He);
```

Chapter 6

Convection Heat Transfer in a Closed Conduit

Scilab code Exa 6.1 Constant heat flux at the wall

```
clc;
clear;
printf("\t\t\tChapter6_example1\n\n\n");
// Determination of the fluid outlet tetnperature
   and the tube-wall temperature at the outlet.
// properties of ethylene glycol at 20 degree
   celsius from appendix table C5
Cp_20=2382;
rou_20=1.116*1000;
v_20=19.18e-6;
kf_20=.249;
a_20=.939e-7;
Pr_20=204;
// specifications of 1/2 standard type M seamless
   copper water tubing from appendix table F2
OD=1.588/100;
ID=1.446/100;
```

```
A=1.642e-4;
Q=3.25e-6;
V=Q/A;
printf("\nThe average flow velocity is %.1f m/s",V
    *100);
// calculation of Reynold's Number to check flow
    regime
Re=V*ID/v_20;
printf("\nThe Reynolds Number is %.1f",Re);
// since Re>he 2100, the flow regime is laminar and
    the hydrodynamic length can be calculated as
Z_h=0.05*ID*Re;
printf("\nThe hydrodynamic length is %.1f cm",Z_h
    *100);
Tbi=20; // bulk-fluid inlet temperature in degree
    celsius
qw=2200; // incident heat flux in W/m^2
L=3; // Length of copper tube in m
R=ID/2; // inner radius in m
Tbo=Tbi+(2*qw*a_20*L)/(V*kf_20*R);
printf("\nThe bulk-fluid outlet temperature is %.1f
    degree celsius",Tbo);
// This result is based on fluid properties
    evaluated at 20 C. taken as a first
    approximation
Z_t=0.05*ID*Re*Pr_20;
printf("\nThe thermal entry length is %.1f m",Z_t);
Two=Tbo+(11*qw*ID)/(48*kf_20); // The wall
    temperature at outlet in degree celsius
printf("\nThe wall temperature at outlet is %.1f
    degree celsius",Two);
//The result is based on first approximation based
    on flow properties evaluated at the fluid inlet
    temperature.
```

Scilab code Exa 6.2 Constant wall temperature

```
clc;
clear;
printf("\t\t\tChapter6_example2\n\n\n");
// determination of average convection coefficient
T_avg=(140+70)/2;
printf("\nThe average bulk temperature is %d degree
    celsius",T_avg);
// properties of water at average bulk temperature
    from appendix table C11
rou=.994*62.4;
kf=.363;
cp=.9980;
a=5.86e-3;
v=0.708e-5;
Pr=4.34;
// specifications of 1 standard type M copper tube
    from appendix table F2
OD=1.125/12; // outer diameter in ft
ID=0.8792; // inner diameter in ft
A=0.006071 // cross sectional area in sq.ft
m_flow=1.5; // mass flow rate in lbm/s
V=m_flow*3600/(rou*A); // velocity in ft/hr
printf("\nThe velocity is %d ft/hr",V);
L=20;
Tw=240;
Tbo=140;
Tbi=70;
hL=-(rou*V*ID*cp*log((Tw-Tbo)/(Tw-Tbi)))/(4*L);
printf("\nThe average convective coefficient is %d
    BTU/(hr. sq.ft.degree Rankine)",hL);
```

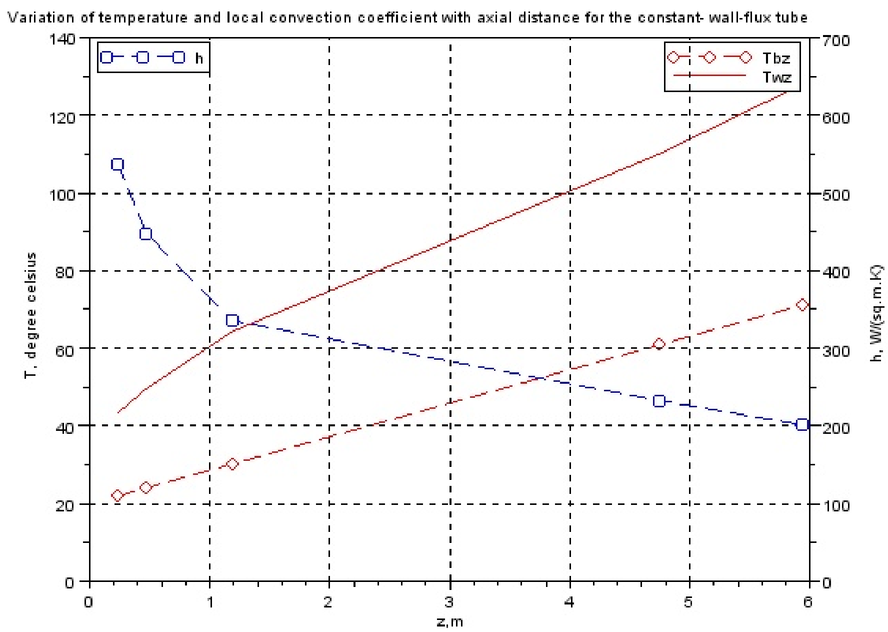

Figure 6.1: Hydrodynamic and Thermal Entry Length

Scilab code Exa 6.3 Hydrodynamic and Thermal Entry Length

```
clc;
clear;
printf("\t\t\tChapter6_example3\n\n\n");
// Determination of the variation of wall temperature with length up to the point where the flow becomes fully developed.
```

```
// properties of milk
kf=0.6; // thermal conductivity in W/(m-K)
cp=3.85*1000; // specific heat in J/(kg*K)
rou=1030; // density in kg/m^3
mu=2.12e3; // viscosity in N s/m^2
// specifications of 1/2 standard type K tubing from
    appendix table F2
OD=1.588/100; // outer diameter in m
ID=1.340/100; // inner diameter in m
A=1.410e-4 // cross sectional area in m^2
rou=1030;
V=0.1;
mu=2.12e-3
// determination of flow regime
Re=rou*V*ID/(mu);
printf("\nThe Reynolds Number is %d",Re);
// The flow being laminar, the hydrodynamic entry
   length is calculated as follows
ze=0.05*ID*Re;
printf("\nThe hydrodynamic entry length is %.1f cm",
   ze*100);
Tbo=71.7; // final temperature in degree celsius
Tbi=20; // initial temperature in degree celsius
L=6; // heating length in m
qw=rou*V*ID*cp*(Tbo-Tbi)/(4*L);
printf("\nThe heat flux is %d W/sq.m",qw);
q=qw*%pi*ID*L;
printf("\nThe power required is %.1f W",q);
printf("\nA 3000 W heater would suffice");
Pr=(cp*mu)/kf; // Prandtl Number
printf("\nThe Prandtl Number is %.1f",Pr);
zf=0.05*ID*Re*Pr;
printf("\nThe length required for flow to be
   thermally developed is %.1f m",zf);
// calculations of wall temperature of the tube
reciprocal_Gz=[0.002 0.004 0.01 0.04 0.05];// values
    of 1/Gz taken
[n m]=size(reciprocal_Gz);
```

```
Nu=[12 10 7.5 5.2 4.5]; //Enter the corresponding value of Nusselts Number from figure 6.8
for i=1:m
    z(i)=ID*Re*Pr*reciprocal_Gz(i);
    h(i)=kf*Nu(i)/ID;
    Tbz(i)=20+(8.617*z(i));
    Twz(i)=Tbz(i)+(11447/h(i));
end
printf("\nSummary of Calculations to Find the Wall Temperature of the Tube");
printf("\n\t1/Gz\t\tNu\t\tz (m)\t\th W/(sq.m.K)\t\tTbz (degree celsius)\t\tTwz (degree celsius)");
for i=1:m
printf("\n\t%.3f\t\t%.1f\t\t%.3f\t\t%d\t\t\t%.1f\t\t\t\t%.1f",reciprocal_Gz(i),Nu(i),z(i),h(i),Tbz(i),Twz(i));
end
subplot(211);
plot(z,Tbz,'r--d',z,Twz,'r-');  // our first figure
a1 = gca();
h1=legend(["Tbz";"Twz"]);
subplot(212)
plot(z,h, 'o--');  // our second figure
hl=legend(['h'],2);
title('Variation of temperature and local convection coefficient with axial distance for the constant-wall-flux tube');
a2 = gca();
a2.axes_visible = ["off", "on","on"];
a2.y_location ="right";

a1.axes_bounds=[0 0 1 1];  // modify the first figure to occupy the whole area
a2.axes_bounds=[0 0 1 1]; // modify the second figure to occupy the whole area too

a1.data_bounds=[0,0;6,140];
a2.data_bounds=[0,0;6,700];
```

```
a1.x_ticks = tlist(["ticks", "locations", "labels"], (0:6)', ["0";"1";"2";"3";"4";"5";"6"]);
a1.x_label
a1.y_label
x_label=a1.x_label;
x_label.text=" z,m"
a2.x_label
a2.y_label
y_label=a1.y_label;
y_label.text="T, degree celsius"
y_label=a2.y_label;
y_label.text="h, W/(sq.m.K)"
xgrid(1);
a2.filled = "off";
```

Scilab code Exa 6.4 Variation of temperature with length of tube

```
clc;
clear;
printf("\t\t\tChapter6_example4\n\n\n");
// The average bulk temperature of the Freon-12 is [-4O +(-4)]/2 = -22 F
// properties of Freon-12 at average bulk temperature
kf=0.04; // thermal conductivity in BTU/(hr.ft. R)
cp=0.2139; // specific heat in BTU/(lbm- R)
rou= 1.489*(62.4); // density in lbm/cu.ft
v=0.272e-5; // viscosity in sq.ft/s
a=2.04e-3; // diffusivity in sq.ft/hr
Pr=4.8; // Prandtl Number
```

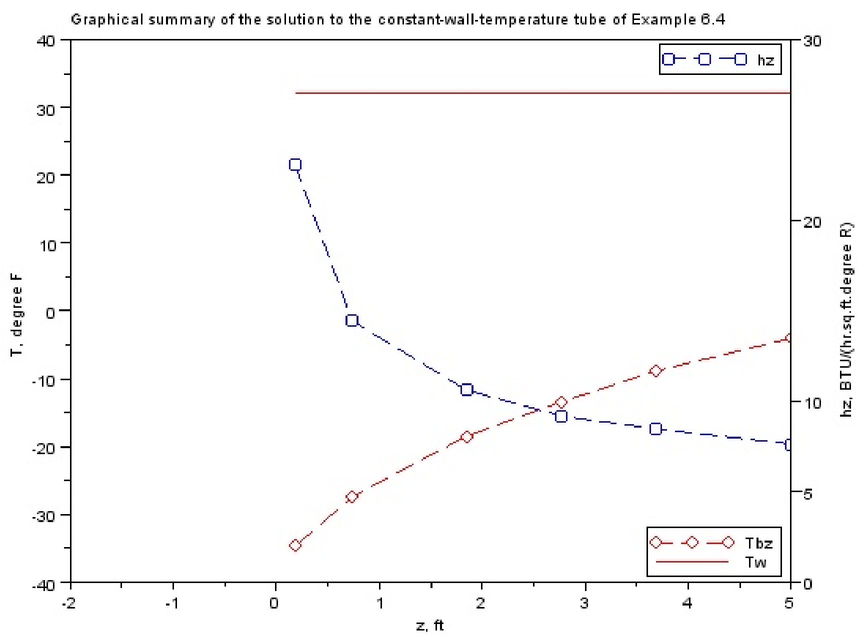

Figure 6.2: Variation of temperature with length of tube

```
// specifications of 3/8 standard type K copper
   tubing from appendix table F2
OD=0.5/12; // outer diameter in ft
ID=0.03350; // inner diameter in ft
A=0.0008814 // cross sectional area in sq.ft
// Laminar conditions are asssumed
z=5;
Tw=32;
Tbo=-4;
Tbi=-40;
L=5;
i=1;
V_assumed(i)=100; //assumed value for velocity
for i=1:6
    inv_Gz(i)=(z*a)/(V_assumed(i)*ID^2);
    Nu=[4.7 5.8 6.2 6.3 6.4 6.4]; // corresponding
       Nusselt numbers from fig. 8.8:
    hL(i)=Nu(i)*kf/ID;
    V(i)=-(2*a*L*hL(i))/((kf*ID/2)*log((Tw-Tbo)/(Tw-
       Tbi)));
        V_assumed(i+1)=V(i);
end
printf("\nSummary of Results\n");
printf("Assmued V (ft/hr)\t1/Gz\tNu(fig 8.8)\thL BTU
   /(hr. sq.ft. degree R)\tV (ft/hr)\n");
for j=1:6
printf("\t%d\t\t%.4f\t%.1f\t\t%.2f\t\t\t\t%d\n",
   V_assumed(j),inv_Gz(j),Nu(j),hL(j),V(j));
end
V_final=V(i-1);
hL_final=hL(i-1);
printf("\nThe final velocity is %d ft/hr = %.4f ft/s
   ",V_final,V_final/3600);
printf("\nThe final convective coefficient is %.2f
   BTU/(hr. sq.ft. degree R)",hL_final);
// checking the laminar-flow assumption by
   calculating the Reynolds number
Re=(V_final/3600)*ID/v;
```

```
printf("\nThe Reynolds number is %d",Re);
// The flow is laminar
m_Fr=rou*A*V_final/3600;
printf("\nThe mass flow rate of Freon-12 is %.2e lbm/s = %.2f lbm/hr",m_Fr,m_Fr*3600);
As=%pi*ID*L;
q=hL_final*As*[(Tw-Tbo)-(Tw-Tbi)]/(log((Tw-Tbo)/(Tw-Tbi)));
printf("\nThe heat gained by Freon-12 is %.1f BTU/hr",q);
q_check=m_Fr*cp*(Tbo-Tbi);
printf("\nOn checking the heat transferred we find almost equal to the heat gained by Freon-12");
rou_water=1.002*62.4; // density of water in lbm/ft^3 from appendix table C11
m_water=rou_water*L*(2/12)*(3/12);
printf("\nThe mass of water in the prescribed volume is %.1f lbm",m_water);
// to remove 144 BTU/lbm of water, the time required is caalculated as below
t=144*m_water/q;
printf("\nThe required time is %.1f hr",t);
inv_Gz1=[0.001 0.004 0.01 0.015 0.02 0.0271]; // guess values of 1/Gz
Nu_D=[19.3 12.1 8.9 7.7 7.1 6.4]; //corresponding Nusselt number from fig. 6.8
[n m]=size(inv_Gz1);
for j=1:m
    Z(j)=ID*Re*Pr*(inv_Gz1(j));
    hz(j)=Nu_D(j)*kf/ID;
    Tbz(j)=32-72*exp(-0.01812*Z(j)*hz(j));
end
printf("\nSummary of Data for Example 6.4 ");
printf("\n\t1/Gz\tNu_D\tz (ft)\thz, BTU/(hr. sq.ft.degree R)\tTbz,degree F\n");
for p=1:m
    printf("\t%.4f\t%.1f\t%.2f\t%.2f\t\t\t%.1f\n",inv_Gz1(p),Nu_D(p),Z(p),hz(p),Tbz(p));
```

```
end
subplot(211);
plot(Z,Tbz,'r--d',Z,Tw,'r-');  // your first figure
a1 = gca();
hl=legend(['Tbz';'Tw'],4);
subplot(212)
plot(Z,hz, 'o--');  // your second figure
a2 = gca();
hl=legend(['hz'],1);
a2.axes_visible = ["off", "on","on"];
a2.y_location ="right";

a1.axes_bounds=[0 0 1 1];  // modify the first figure to occupy the whole area
a2.axes_bounds=[0 0 1 1]; // modify the second figure to occupy the whole area too
a2.filled = "off";
a1.data_bounds=[-2,-40;5,40];
a2.data_bounds=[-2,0;5,30];
x_label1=a1.x_label;
x_label1.text="z, ft";
y_label2=a2.y_label;
y_label2.text="hz, BTU/(hr.sq.ft.degree R)";
y_label=a1.y_label;
y_label.text="T, degree F";
newticks1=a1.y_ticks;
newticks1(2)=[-40;-30;-20;-10;0;10;20;30;40];
newticks1(3)=['-40';'-30';'-20';'-10';'0';'10';'20';'30';'40'];
a1.y_ticks=newticks1;
newticks2=a2.y_ticks;
newticks2(2)=[0;5;10;20;30];
newticks2(3)=['0';'5';'10';'20';'30'];
a2.y_ticks=newticks2;
newticks=a1.x_ticks;
newticks(2)=[-2;-1;0;1;2;3;4;5];
newticks(3)=['-2';'-1';'0';'1';'2';'3';'4';'5'];
a1.x_ticks=newticks;
```

```
title('Graphical summary of the solution to the constant−wall−temperature tube of Example 6.4');
```

Scilab code Exa 6.5 Combined Entry Length Problem

```
clc;
clear;
printf("\t\t\tChapter6_example5\n\n\n");
// Determination for the power required for heating and the wall temperature at the outlet.
// The liquid properties are evaluated at the mean temperature of (80 + 20)/2 = 50 C.
// specifications of 1 standard type K copper water tubing from appendix table F2
OD = 2.858/100; // outer diameter in m
ID = 2.528/100; // inner diameter in m
A = 5.019e-4; // cross sectional area in sq.m
// 1 oz = 2.957e−5 m^3
Q=80*2.957e-5/120; // The volume flow rate of water (at 20 C) in cu.m/s
printf("\nThe volume flow rate of water (at 20 C) is %.2e cu.m/s",Q);
p_20= 1.000*1000; // density of water at 20 C in kg/cu.m
// properties of water at 50 C from appendix table C11
p_50= 0.990*(1000); // density in kg/m3
cp= 4181; // specific heat in J/(kg*K)
v = 0.586e-6; // viscosity in sq.m/s
kf = 0.640; // thermal conductivity in W/(m.K)
a = 1.533e-7; // diffusivity in sq.m/s
Pr = 3.68; // Prandtl number
```

```
mass_flow=p_20*Q; // mass flow rate through the tube in kg/s
printf("\nmass flow rate through the tube is %.4f kg/s",mass_flow);
L=3; //  length of tube in m
As=%pi*ID*L;
Tbo=80; // final temperature in  C
Tbi=20; // initial temperature in  C
qw=mass_flow*cp*(Tbo-Tbi)/(As);
q=qw*As;
A=%pi*(ID/2)^2;
printf("\nThe power required in %.3e W/sq.m = %d W",qw,q);
V=mass_flow/(p_50*A); // average velocity at 50  C
printf("\nThe average velocity at 50 C is %.2e m/s",V);
Re=(V*ID)/v; // Reynold's Number
printf("\nThe Reynolds Number for the flow is %d",Re);
// The flow is laminar so we can use  Figure 6.12 to obtain the information needed on Nusselt number and to find hz
inv_Gz=L/(Re*ID*Pr); // The inverse Graetz number at tube end, based on 50 C conditions
printf("\nThe inverse Graetz number at tube end, based on 50 C conditions is %.4f",inv_Gz);
Nu=6.9; //value of corresponding Nusselts Number from figure 6.12
hz=(Nu*kf)/ID;
printf("\nThe local convection coefficient is %.1f W/(sq.m.K)",hz);
Two=(qw/hz)+Tbo; // The outlet wall temperature in  C
printf("\nThe outlet wall temperature is %d  C",Two);
```

Scilab code Exa 6.6 Heat transfer to and from turbulent flow

```
clc;
clear;
printf("\t\t\tChapter6_example6\n\n\n");
// determibation of heat gained
// air properties to be calculated at T=(72+45)/2=58.5 degree Fahrenheit
// properties at T=58.5 degree fahrenheit from appendix table D1
p = 0.077; // density in lbm/ft^3
cp = 0.240; // specific heat in BTU/(lbm.degree Rankine)
v = 15.28e-5; // viscosity in ft^2/s
kf = 0.0146; // thermal conductivity in BTU/(hr.ft."R)
a = 0.776; // diffusivity in ft^2/hr
Pr = 0.711; // prandtl number
D=7/12; // diameter in ft
L=40; // length in ft
Tbo=72; // outlet temperature in degree Fahrenheit
Tbi=45; // inlet temperature in degree Fahrenheit
A=%pi*(D^2)/4; // cross sectional area of duct in ft^2
// density at outlet temperature in lbm/ft^3
rou_o=.0748;
V=10; // average velocity in ft/s
mass_flow=rou_o*A*V;
printf("\nThe mass flow rate is %.1f lbm/s",mass_flow);
// average velocity evaluated by using the average bulk temperature
```

```
V_avg=mass_flow/(p*A);
printf("\nThe average velocity evaluated by using
    the average bulk temperature is %.2f ft/s",V_avg)
    ;
Re=(V_avg*D)/v;
printf("\nThe Reynolds number for the flow is %.3e "
    ,Re);
// the flow is in turbulent regime
q=mass_flow*cp*(Tbo-Tbi);
printf("\nThe heat gained by air is %.3f BTU",q);
hc=1; // convection coefficient between the outside
    duct wall and the attic air in BTU/(hr. sq.ft.
    degree Rankine).
T_inf=105; // The temperature of attic air
    surrounding the duct in degree Fahrenheit
hz=(0.023*Re^(4/5)*Pr^0.4)*kf/D; // The local
    coefficient at the duct end is %.2f BTU/(hr. sq.
    ft.degree Rankine)
printf("\nThe local coefficient at the duct end is %
    .2f BTU/(hr. sq.ft.degree Rankine)",hz);
qw=(T_inf-Tbo)/((1/hc)+(1/hz)); // wall flux in BTU
    /(hr. sq.ft.degree Rankine)
printf("\nThe wall flux is %.1f BTU/(hr. sq.ft.
    degree Rankine)",qw);
Two=qw*(1/hz)+Tbo; // The wall temperature at exit
    in degree Fahrenheit
printf("\nThe wall temperature at exit is %.1f
    degree Fahrenheit",Two);
```

Chapter 7

Convection Heat Transfer in Flows Past Immersed Bodies

Scilab code Exa 7.1 Laminar Boundary Layer flow over flat plate

```
clc;
clear;
printf("\t\t\tChapter7_example1\n\n\n");
printf("\t\t\tSolution to part (a)\n");
// determination of boundary layer growth with length
// properties of air at 27 degree celsius from appendix table D.1
rou=1.177; // density in kg/cu.m
v=15.68e-6; // viscosity in sq.m/s
L=0.5; // length in m
V_inf=1; // air velocity in m/s
Re= (V_inf*L)/v; // Reynolds Number
printf("The Reynolds Number is %.2e ",Re);
// Reynolds Number is less than 5e5 hence the flow
```

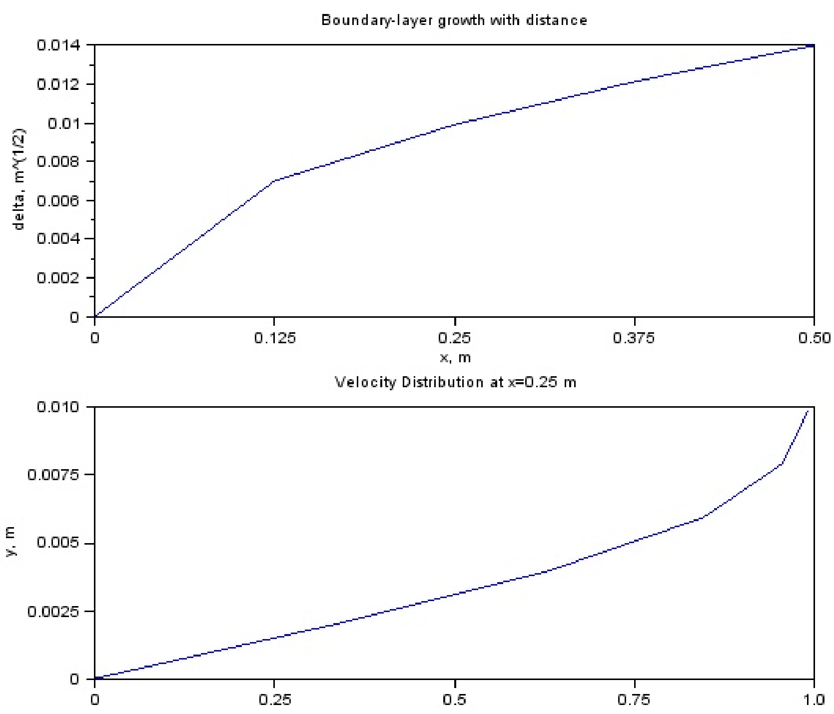

Figure 7.1: Laminar Boundary Layer flow over flat plate

```
is laminar and Blasius Solution applies
x=[0 0.125 0.25 0.375 0.5]; // distances in m where boundary layer growth is determined
[n,m]=size(x);
for i=1:m
    delta(i)=5*x(i)^0.5/(V_inf/v)^0.5;
end
subplot(211);
plot(x,delta);
a=gca();
newTicks=a.x_ticks;
newTicks(2)=[0;0.125;0.25;0.375;0.5];
newTicks(3)=['0';'0.125';'0.25';'0.375';'0.50'];
a.x_ticks=newTicks;
title('Boundary-layer growth with distance');
xlabel('x, m');
ylabel('delta, m^(1/2)');
printf("\n\t\t\tSolution to part (b)\n");
// produce graph of velocity distribution at x=0.25 m
eta=0:5;
[p,q]=size(eta);
f=[0 0.32979 0.62977 0.84605 0.95552 0.99155];// value for f for corresponding eta value from Table 7.1
for j=1:q
    y(j)=eta(j)*(v*0.25)^0.5;
end
printf("\n\t\t\tResults of Calculations for Example 7.1\n");
printf("\teta\t\ty,m\t\t\tf=vx, m/s\n");
for i=1:q
printf("\t%d\t\t%.2e\t\t%.5f\n",eta(i),y(i),f(i));
end
subplot(212);
plot(f,y);
b=gca();
newTicks1=b.x_ticks;
```

```
newTicks1(2)=[0;0.25;0.5;0.75;1.0];
newTicks1(3)=['0';'0.25';'0.5';'0.75';'1.0'];
b.x_ticks=newTicks1;
newTicks2=b.y_ticks;
newTicks2(2)=[0;0.0025;0.005;0.0075;0.010];
newTicks2(3)=['0';'0.0025';'0.005';'0.0075';'0.010'
    ];
b.y_ticks=newTicks2;
title('Velocity Distribution at x=0.25 m');
xlabel('Vx, m/s');
ylabel('y, m');
printf("\t\t\tSolution to part (c)\n");
// calculation of absolute viscosity
gc=1;
mu=rou*v/gc;
printf("\nThe absolute viscosity is %.3e N.s/sq.m",
    mu);
b=1; // width in m
Df=0.664*V_inf*mu*b*(Re)^0.5;
printf("\nThe skin-drag is %.2e N",Df);
printf("\nThe skin-drag including both sides of
    plate is %.2e N",2*Df);
```

Scilab code Exa 7.2 Flow over plate at constant temperature

```
clc;
clear;
printf("\t\t\tChapter7_example2\n\n\n");
// determination of temperature profile
// properties of water at (40 + 100)/2 = 70 F = 68
    F from appendix table C11
```

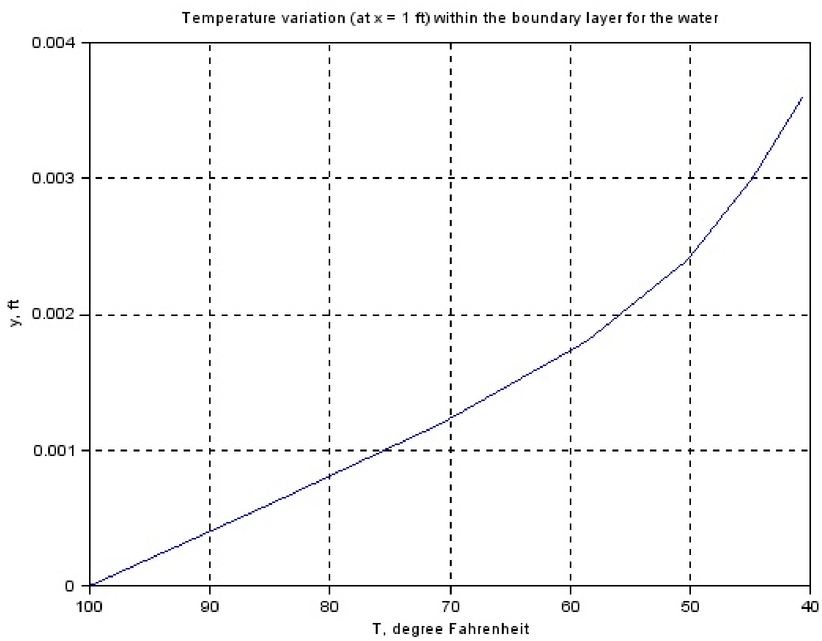

Figure 7.2: Flow over plate at constant temperature

```
rou= 62.4; // density in Ibm/ft^3
cp=0.9988; // specific heat BTU/(lbm-degree Rankine)
v= 1.083e-5; // viscosity in sq.ft/s
kf = 0.345 ; // thermal conductivity in BTU/(hr.ft.degree Rankine)
a = 5.54e-3; // diffusivity in sq.ft/hr
Pr = 7.02; // Prandtl Number
V=1.2; // velocity in ft/s
x=[1 2]; // distances from plate entry in ft
for i=1:2
Re(i)=(V*x(i))/v; // Reynolds Number at x=1 ft
printf("\nThe Reynolds Number at x=%d ft is %.3e",i,Re(i));
// since Reynolds Number is less than 5*10^5, the flow is laminar
hL(i)=0.664*Pr^(1/3)*Re(i)^0.5*kf/x(i);
printf("\nThe average convection coefficient at x=%d is %.1f BTU/(hr. sq.ft. degree Rankine)",i,hL(i));
Tw=100; // temperature of metal plate in degree fahrenheit
T_inf=40; // temperature of water in degree fahrenheit
A(i)=x(i)*18/12; // cross sectional area for 1 ft length
q(i)=hL(i)*A(i)*(Tw-T_inf);
printf("\nThe heat transferred to water over the x=%d ft is %.3e BTU/hr",i,q(i));
end
eta=0:0.2:1.2;
[n m]=size(eta);
theta=[1 .75 .51 .31 .17 .08 0.01]; // values of dimensionless temperature from figure 7.7 corresponding to eta value taken
for i=1:m
y(i)=eta(i)*(v*x(1)/V(1))^0.5;
T(i)=theta(i)*(Tw-T_inf)+T_inf;
end
```

```
printf("\nSolution Chart for example 7.2\n");
printf("\teta\t\ttheta\t\ty, ft\t\t\tT, degree F\n")
   ;
for i=1:m
printf("\t%.1f\t\t%.2f\t\t%.1e\t\t\t%.1f\n",eta(i),
   theta(i),y(i),T(i));
end
plot(T,y);
a=gca();
newTicks=a.x_ticks;
newTicks(2)=[100; 90; 80; 70; 60;50; 40];
newTicks(3)=['100'; '90'; '80'; '70'; '60';'50'; '40
   '];
a.x_ticks=newTicks;
newTicks1=a.y_ticks;
newTicks1(2)=[0; 0.001; 0.002; 0.003; 0.004];
newTicks1(3)=['0'; '0.001'; '0.002'; '0.003'; '0.004
   '];
a.y_ticks=newTicks1;
a.axes_reverse=["on","off"];
xgrid(1);
title('Temperature variation (at x = 1 ft) within
   the boundary layer for the water');
xlabel('T, degree Fahrenheit');
ylabel('y, ft');
```

Scilab code Exa 7.3 Flow over Isothermal flat plate

```
clc;
clear;
printf("\t\t\tChapter7_example3\n\n\n");
// Determination of the average convection
   coefficient and the total drag force exerted on
```

```
the plate.
// properties of air at (300 + 400)/2 = 350 K from appendix table D1
rou= 0.998; // density in kg/cu.m
cp= 1009; // specific heat in J/(kg*K)
v= 20.76e-6; // viscosity in sq.m/s
Pr = 0.697; // Prandtl Number
k= 0.03003; // thermal conductivity in W/(m.K)
a = 0.2983e-4; // diffusivity in sq.m/s
L=1; // Length of plate in m
V=5; // velocity of air in m/s
b=0.5; // width in m
Re=V*L/v; // Reynolds number at plate end
printf("\nThe Reynolds number is %.2e",Re);
// Since the flow is laminar at plate end, The average convection coefficient is calculated with Equation Nu=h*L/k= 0.664 Re^(1/2)Pr^(1/3)
h=k*0.664*Re^(1/2)*Pr^(1/3)/L; // The average convection coefficient in W/(sq.m.K)
printf("\nThe average convection coefficient is %.2f W/(sq.m.K)",h);
Df=0.664*V*rou*v*b*(Re)^0.5; // drag force in N
printf("\nThe drag force is %.2e N",Df);
hx=(1/2)*h; // local convective coefficient
printf("\nThe local convective coefficient is %.2f W/(sq.m.K)",hx);
delta=5*L/(Re)^0.5; // The boundary-layer thickness at plate end
printf("\nThe boundary-layer thickness at plate end is %.2f cm",delta*100);
delta_t=delta/(Pr)^(1/3);
printf("\nThe thermal-boundary-layer thickness is %.2f cm",delta_t*100);
```

Scilab code Exa 7.4 Maximum heater surface temperature

```
clc;
clear;
printf("\t\t\tChapter7_example4\n\n\n");
//  Determination of the maximum heater-surface
    temperature for given conditions
// fluid properties at (300 degree R + 800 degree R)
    /2 = 550 degree R=540degree R from Appendix Table
     D.6
rou= 0.0812; // density in Ibm/ft^3
cp=0.2918; // specific heat BTU/(lbm-degree Rankine)
v= 17.07e-5; // viscosity in ft^2/s
kf = 0.01546 ; // thermal conductivity in BTU/(hr.ft
    .degree Rankine)
a = 0.8862; // diffusivity in ft^2/hr
Pr = 0.709; // Prandtl Number
qw=10/(1.5*10.125)*(1/.2918)*144; // The wall flux
printf("\nThe wall flux is %d BTU/(hr. sq.ft)",qw);
V_inf=20; // velocity in ft/s
L=1.5/12; // length in ft
Re_L=V_inf*10*L/v; // Reynolds number at plate end
printf("\nThe Reynolds number at plate end is %.2e",
    Re_L);
// So the flow is laminar and we can find the wall
    temperature at plate end as follows
T_inf=300; // free stream temperature in degree
    Rankine
Tw=T_inf+(qw*L*10/(kf*0.453*(Re_L)^0.5*(Pr)^(1/3)));
printf("\nThe maximum heater surface temperature is
    %d degree Rankine",Tw);
```

Scilab code Exa 7.5 Reynolds Colburn Analogy

```
clc;
clear;
printf("\t\t\tChapter7_example5\n\n\n");
// validation of the equation st.(Pr)^(2/3)=Cd/2 where St: Stanton Number Pr:Prandtl Number Cd: Drag Coefficient
// values of parameters from example 7.4
rou= 0.0812; // density in Ibm/ft^3
cp=0.2918; // specific heat BTU/(lbm-degree Rankine)
v= 17.07e-5; // viscosity in ft^2/s
kf = 0.01546 ; // thermal conductivity in BTU/(hr.ft.degree Rankine)
a = 0.8862; // diffusivity in ft^2/hr
Pr = 0.709; // Prandtl Number
Tw=469; // maximum heater temperature in degree Rankine
T_inf=300; // free-stream temperature in degree Rankine
qw=324; // The wall flux in BTU/(hr.ft^2)
V_inf=20; // velocity in ft/s
hx=qw/(Tw-T_inf); //  The convection coefficient
printf("\nThe convection coefficient is %.2f BTU/(hr.sq.ft.degree R)",hx);
LHS=(hx/3600)*(Pr)^(2/3)/(rou*cp*V_inf);
printf("\nThe value of left hand side of the equation is %.2e",LHS);
Re_L=1.46e+005; // Reynolds number at plate end
RHS=0.332*(Re_L)^(-0.5);
printf("\nThe value of left hand side of the equation is %.2e",RHS);
err=(LHS-RHS)*100/LHS;
printf("\nThe error is %d percent",err);
printf("\nSince the error is only %d percent, the agreement is quite good",err);
```

Scilab code Exa 7.6 Drag due to skin friction

```
clc;
clear;
printf("\t\t\tChapter7_example6\n\n\n");
//  Estimation of the drag due to skin friction
// properties of water at 68 F from Appendix Table
    C.11
rou= 62.4; // density in Ibm/cu.ft
v= 1.083e-5; // viscosity in sq.ft/s
V_inf=5*.5144/.3048; // barge velocity in ft/s using
     conversion factors from appendix table A1
printf("\nThe barge velocity is %.2f ft/s",V_inf);
L=20; // Length of barge in ft
Re_L=V_inf*L/v; // Reynolds number at plate end
printf("\nThe Reynolds number at plate end is %.2e",
    Re_L);
Cd=0.003; //value of Cd corresponding to the
    Reynolds number from figure 7.11
gc=32.2;
b=12; // width in ft
Df=(Cd*rou*V_inf^2*b*L)/(2*gc);
printf("\nThe drag force is %d lbf",Df);
```

Scilab code Exa 7.7 Wattage requirement of heater

```
clc;
clear;
```

```
printf("\t\t\tChapter7_example7\n\n\n");
// Determination of wattage requirement
// properties of carbon dioxide at a film temperature of (400+600)/2 = 500 K from appendix table D2
rou= 1.0732; // density in kg/m^3
cp= 1013; // specific heat in J/(kg*K)
v= 21.67e-6; // viscosity in m^2/s
Pr = 0.702; // Prandtl Number
k= 0.03352; // thermal conductivity in W/(m.K)
a = 0.3084e-4; // diffusivity in m^2/s
V_inf=60; // carbon dioxide velocity in m/s
x_cr=(5e5)*v/V_inf; // The transition length in m
printf("\nThe transition length is %.1f cm",x_cr*100);
w=4; // width of each heater in cm
b=.16; // effective heating length in m
Tw=600; // temperature of heater surface in K
T_inf=400; // temperature of carbon dioxide in K
r=pmodulo(x_cr*100,w);
n=(x_cr*100+r)/w; // number of heater where transition occurs
printf("\nThe transition thus occur at %dth heater",n);
m=6; // number of heater strips
q=zeros(m+1,1);
x=[0.04 0.08 0.12 0.16 0.20 0.24];
for i=1:n-1 // transition occurs at 5th heater, so laminar zone equation is followed till then
    h(i)=(0.664*k)*(V_inf/v)^0.5*(Pr)^(1/3)/x(i)^0.5;
    printf("\n\nThe convective coefficient for heater no. %d is %d W/(sq.m.K)",i,h(i));
    q(i+1)=h(i)*x(i)*b*(Tw-T_inf);
    dq(i)=q(i+1)-q(i);
    printf("\nThe heat transferred by heater no. %d is %d W",i,dq(i));
end
```

```
// Turbulent zone exists from 5th heater onwards so
   the following equation is followed Nu=h*x/kf
   =[0.0359*(Re_L)^(4/5)-830]*(Pr)^(1/3)
for i=5:6
    Re_L(i)=V_inf*x(i)/v;
    h(i)=(k/x(i))*[0.0359*(Re_L(i))^(4/5)-830]*(Pr)
       ^(1/3)
    printf("\n\nThe Reynolds number for heater no.
       %d is %.2e",i,Re_L(i));
    printf("\nThe convective coefficient for heater
       no. %d is %.1f W/(sq.m.K)",i,h(i));
    q(i+1)=h(i)*x(i)*b*(Tw-T_inf);
    dq(i)=q(i+1)-q(i);
    printf("\nThe heat transferred by heater no. %d
       is %d W",i,dq(i));
end
```

Scilab code Exa 7.8 Flow past a telephone pole

```
clc;
clear;
printf("\t\t\tChapter7_example8\n\n\n");
// Estimation of force exerted on the pole
// properties of air at given conditions from
   appendix table D1
rou= 0.0735; // density in Ibm/ft^3
v= 16.88e-5; // viscosity in ft^2/s
V=20*5280/3600; // flow velocity in ft/s
printf("\nThe flow velocity is %.1f ft/s",V);
D=12/12; // diameter of pole in ft
L=30;// length of pole in ft
gc=32.2;
Re_D=V*D/v; // Reynolds Number for flow past the
```

```
pole
printf("\nThe Reynolds Number for flow past the pole is %.2e ",Re_D);
Cd_cylinder=1.1; // value of Cd for smooth cylinder from figure 7.22
A_cylinder=D*L; // frontal area of pole
printf("\nThe frontal area of pole is %d sq.ft",A_cylinder);
Df_cylinder=Cd_cylinder*(1/2)*rou*V^2*A_cylinder/gc;
printf("\nThe Drag force exerted on only the pole is %.1f lbf",Df_cylinder);
D_square=2/12; // length of square part of pole
L_square=4;
Re_square=V*D_square/v; // Reynolds Number for square part of pole
printf("\nThe  Reynolds Number for square part of pole is %.1e",Re_square);
Cd_square=2; // Corresponding value of Cd for square part from figure 7.23
A_square=D_square*L_square; // projected frontal area of square part
printf("\nThe frontal area of square part of pole is %.3f sq.ft",A_square);
Df_square=Cd_square*(1/2)*rou*V^2*A_square/gc;
printf("\nThe Drag force exerted on cross piece of the pole is %.2f lbf",Df_square);
Df_total=Df_cylinder+Df_square;
printf("\nThe total drag force on the pole is %.1f lbf",Df_total);
```

Scilab code Exa 7.9 Current through Hot Wire Anemometer

```
clc;
```

```
clear;
printf("\t\t\tChapter7_example9\n\n\n");
// determination of required current
// properties of air at film temperature (300 + 500)/2 = 400 K from appendix table D1
rou= 0.883; // density in kg/cu.m
cp= 1014; // specific heat in J/(kg*K)
v= 25.90e-6; // viscosity in sq.m/s
Pr = 0.689; // Prandtl Number
kf= 0.03365; // thermal conductivity in W/(m.K)
a = 0.376e-4; // diffusivity in sq.m/s
V_inf=1; // velocity in m/s
D=0.00013; // diameter in m
L=1/100; // length of wire in cm
Re_D=V_inf*D/v; // The Reynolds number of flow past the wire
printf("\nThe Reynolds number of flow past the wire is %.3f",Re_D);
C=0.911; //value of C for cylinder from table 7.4
m=0.385; //value of m for cylinder from table 7.4
hc=kf*C*(Re_D)^m*(Pr)^(1/3)/D; // the convection coefficient in W/(m^2.K)
printf("\nThe convection coefficient is %d W/(sq.m.K)",hc);
Tw=500; // air stream temperature in K
T_inf=300; // wire surface temperature in K
As=%pi*D*L; // cross sectional area in sq.m
qw=hc*As*(Tw-T_inf); // The heat transferred to the air from the wire
printf("\nThe heat transferred to the air from the wire is %.3f W",qw);
resistivity=17e-6; // resistivity in ohm cm
Resistance=resistivity*(L/(%pi*D^2)); // resistance in ohm
printf("\nThe resistance is %.3f ohm",Resistance/100);
i=(qw*100/Resistance)^0.5; // current in ampere
printf("\nThe current is %.1f A",i);
```

Scilab code Exa 7.10 Pressure Drop for flow over tubes

```
clc;
clear;
printf("\t\t\tChapter7_example10\n\n\n");
// Calculation of the pressure drop for the air
   passing over the tubes and the heat transferred
   to the air.
// properties of air at  70 + 460 = 530 degree R =
   540 degree R from appendix table D1
rou= 0.0735; // density in Ibm/cu.ft
cp=0.240; // specific heat BTU/(lbm-degree Rankine)
v= 16.88e-5; // viscosity in sq.ft/s
kf = 0.01516 ; // thermal conductivity in BTU/(hr.ft
   .degree Rankine)
a = 0.859; // diffusivity in sq.ft/hr
Pr = 0.708; // Prandtl Number
// specifications of 3/4 standard type K copper
   tubing from appendix table F2
OD=0.875/12; // outer diameter in ft
ID=0.06208; // inner diameter in ft
A=0.003027; // cross sectional area in sq.ft
L=2;
sL=1.5/12;
sT=1.3/12;
V_inf=12; // velocity of air in ft/s
V1=(sT*V_inf)/(sT-OD); // velocity at area A1 in ft/
   s
printf("\nVelocity at area A1 is %.1f ft/s",V1);
sD=((sL)^2+(sT/2)^2)^0.5; // diagonal pitch in inch
printf("\nThe diagonal pitch is %.2f in",sD*12);
V2=(sT*V_inf)/(2*(sD-OD));
```

```
printf("\nVelocity at area A2 is %.1f ft/s",V2);
if V1>V2 then
    Vmax=V1;
    else Vmax=V2;
end
Re_D=Vmax*OD/v; // Reynolds Number
printf("\nThe Reynolds number is %.2e ",Re_D);
sT_OD=1.3/0.875;
sT_sL=1.3/1.5;
printf("\nThe values of parameters are sT/Do=%.2f and sT/sL=%.2f",sT_OD,sT_sL);
f1=0.35; //value of f1 for above values of sT/Do and Re
f2=1.05; //Corresponding value of f2 for above values of sT/sL and Re
gc=32.2;
N=7;
dP=N*f1*f2*(rou*Vmax^2/(2*gc));
printf("\nThe pressure drop is %.2f lbf/ft^2 = %.4f psi",dP, dP/147);
sL_Do=sL/OD;
C1=0.438; //value of C1 for above values of sT/Do and sL/Do
C2=0.97; //value of C2 for above values of sT/Do and sL/Do
m=0.565; //value of m for above values of sT/Do and sL/Do
hc=kf*1.13*C1*C2*(Re_D)^m*(Pr)^(1/3)/OD; // The convection coefficient
printf("\nThe convection coefficient is %.1f BTU/(hr.sq.ft.degree Rankine)",hc);
As=70*%pi*OD*L; // outside surface area of 70 tubes
printf("\nThe outside surface area of 70 tubes is %.1f sq.ft",As);
Tw=200; // outside surface temeperature in degree F
T_inf=70; // air temperature in degree F
q=hc*As*(Tw-T_inf);// heat transferred
printf("\nThe heat transferred is %.2e BTU/hr",q);
```

Chapter 8

Natural Convection Systems

Scilab code Exa 8.1 Natural Convection on a vertical surface

```
clc;
clear;
printf("\t\t\tChapter8_example1\n\n\n");
// Determination of the heat transferred to the wall
   .
// air properties at (400+120)/2 =260 degree F = 720
    degree R from Appendix Table D1
rou= 0.0551; // density in Ibm/cu.ft
cp=0.2420; // specific heat BTU/(lbm-degree Rankine)
v= 27.88e-5; // viscosity in sq.ft/s
kf = 0.01944 ; // thermal conductivity in BTU/(hr.ft
   .degree Rankine)
a = 1.457; // diffusivity in sq.ft/hr
Pr = 0.689; // Prandtl Number
T_inf=120+460; // wall temperature in degree R
Tw=400+460; // inside wall temperature in degree R
Beta=1/T_inf;
printf("\nThe volumetric thermal expansion
   coefficient is %.5f/degree R",Beta);
```

```
gc=32.2;
L=1; // length of wall in ft
W=2; // width in ft
Gr=(gc*Beta*(Tw-T_inf)*L^3)/v^2;// Grashof Number
printf("\nThe Grashof number is %.2e",Gr);
temperature_slope=0.505; //value of temperature slope from table 8.1 corresponding to Pr=.72
hL=(kf/L)*(4/3)*(Gr/4)^(1/4)*temperature_slope; // The convection coefficient in BTU/(hr.ft^2.degree R)
printf("\nThe convection coefficient is %.2f BTU/(hr.sq.ft.degree R)",hL);
A=L*W; // cross sectional area in sq.ft
qw=hL*A*(Tw-T_inf);
printf("\nThe heat transferred is %d BTU/hr",qw);
```

Scilab code Exa 8.2 1D heat flow through a window glass

```
clc;
clear;
printf("\t\t\tChapter8_example2\n\n\n");
// Determination of heat lost through the glass per unit area
// properties of air at  22 + 273 = 295 K = 300 K(approx) and 273 K from appendix table D1
rou= [1.177 1.295]; // density in kg/cu.m
cp= [1005 1005.5]; // specific heat in J/(kg*K)
v= [15.68e-6 12.59e-6]; // viscosity in sq.m/s
Pr = [0.708 0.713]; // Prandtl Number
kf= [0.02624 0.02426]; // thermal conductivity in W/(m.K)
a = [0.22160e-4 0.17661e-4]; // diffusivity in sq.m/s
```

```
T_inf=[22 0]// inside and outside temperature in K
Beta=[1/(T_inf(1)+273) 1/(T_inf(2)+273)]; // volumetric thermal expansion coefficient at 295 K and 273 K
printf("\nThe volumetric thermal expansion coefficients at 295 K and 273 K are respectively %.5f and %.5f",Beta(1),Beta(2));
g=9.81;
t=0.005; // thickness of glass
L=0.60; // window length in m
k=0.81; // thermal conductivity of glass from appendix table B3
// for first guess
Tw=[18 4];
printf("\nFor first guess, the results are:\n");
for i=1:2
    Ra(i)=(g*Beta(i)*(Tw(i)-T_inf(i))*L^3)/(v(i)*a(i));
    hL(i)=(kf(i)/L)*(0.68+((0.67*(abs(Ra(i)))^(1/4))/(1+(0.492/Pr(i))^(9/16))^(4/9)));
end
printf("\nThe Rayleigh Numbers are %.3e and %.3e",-Ra(1),Ra(2));
printf("\nThe convective coefficients are %.2f W/(sq.m.K) and %.2f W/(sq.m.K)",hL(1),hL(2));
q=(T_inf(1)-T_inf(2))/((1/hL(2))+(t/k)+(1/hL(1)));
printf("\nThe heat flux is %.1f W/sq.m",q);
for i=1:2
    Tw_final(i)=T_inf(i)-q*(1/hL(i));
    printf("\nThe wall temperature calculated is %.1f",abs(Tw_final(i)));
    Tw(i)=abs(Tw_final(i)); // second guess
end
printf("\nFor second guess, the results are:\n");
for i=1:2
    Ra(i)=(g*Beta(i)*(Tw(i)-T_inf(i))*L^3)/(v(i)*a(i));
    hL(i)=(kf(i)/L)*(0.68+((0.67*(abs(Ra(i)))^(1/4))
```

```
        /(1+(0.492/Pr(i))^(9/16))^(4/9)));
end
printf("\nThe Rayleigh Numbers are %.3e and %.3e",-
   Ra(1),Ra(2));
printf("\nThe convective coefficients are %.2f W/(sq
   .m.K) and %.2f W/(sq.m.K)",hL(1),hL(2));
q=(T_inf(1)-T_inf(2))/((1/hL(2))+(t/k)+(1/hL(1)));
printf("\nThe heat flux is %.1f W/sq.m",q);
for i=1:2
    Tw_final(i)=T_inf(i)-q*(1/hL(i));
    printf("\nThe wall temperature calculated is %.1
       f degree celsius",abs(Tw_final(i)));
end
```

Scilab code Exa 8.3 The cooling unit of a refrigerator

```
clc;
clear;
printf("\t\t\tChapter8_example3\n\n\n");
// determination of heat loss through the side.
rou= 0.0735; // density in Ibm/cu.ft
cp=0.240; // specific heat BTU/(lbm-degree Rankine)
v= 16.88e-5; // viscosity in sq.ft/s
kf = 0.01516 ; // thermal conductivity in BTU/(hr.ft
   .degree Rankine)
a = 0.859; // diffusivity in sq.ft/hr
Pr = 0.708; // Prandtl Number
Tw=90;
T_inf=70;
g=32.2;
L=5.5; // length in ft
W=2+(4/12); // width in ft
Beta=1/(Tw+460); // volumetric thermal expansion
```

```
coefficient in per degree Rankine
printf("\nThe volumetric thermal expansion coefficient is %.5f /degree R",Beta);
Ra=(g*Beta*(Tw-T_inf)*L^3)/(v*a/3600);
printf("\nThe Rayleigh Number is %.2e ",Ra);
hc=(kf/L)*(0.825+((0.387*(Ra)^(1/6))/(1+(0.492/Pr)^(9/16))^(8/27)))^2;
printf("\nThe value of convection coefficient is %.3f BTU/(hr.sq.ft.degree R)",hc);
q=hc*L*W*(Tw-T_inf);
printf("\nThe heat gained is %d BTU/hr",q);
```

Scilab code Exa 8.4 Natural convection on an inclined flat plate

```
clc;
clear;
printf("\t\t\tChapter8_example4\n\n\n");
// Determination of the variation of average convection coefficient with distance
// properties of air at (65 + 20)/2 = 42.5 degree C =315 K. from appendix table D1
rou= 1123; // density in kg/m^3
cp= 1006.7; // specific heat in J/(kg*K)
v= 17.204e-6; // viscosity in m^2/s
Pr =0.703; // Prandtl Number
kf= 0.02738; // thermal conductivity in W/(m.K)
a = 0.2446e-4; // diffusivity in m^2/s
g=9.81;
L=5;
theta=45;
T_inf=20; // ambient air temperature in degree C
Tw=65; // roof surface temperature in degree C
Beta=1/(T_inf+273); // volumetric thermal expansion
```

```
coefficient in per K
printf("\nThe volumetric thermal expansion coefficient is %.5f /K",Beta);
// determination of Laminar-turbulent  transition length by Vliet equation Ra=3x10^5xexp(0.1368cos(90-theta))
x=((3e5*exp(0.1368*cos(90-theta))*v*a)/(g*cos(theta)*Beta*(Tw-T_inf)))^(1/3);
printf("\nThe Laminar-turbulent  transition length by Vliet equation is %.3f m",x);
i=1;
N=1;
n=0;
X=[0.02 0.04 0.05 0.051 0.1 1.0 3.0 5.0]; //entering values for length(m)
[n m]=size(X);
for i=1:m
    if X(i)<=x then
        // Laminar Flow regime exists
        Ra(i)=(g*cos(%pi*45/180)*Beta*(Tw-T_inf)*X(i)^3)/(v*a);
        hc(i)=(kf/X(i))*(0.68+(0.670*Ra(i)^(1/4))/(1+(0.492/Pr)^(9/16))^(4/9));
    else
        // Turbulent Flow regime exists
        Ra(i)=(g*Beta*(Tw-T_inf)*X(i)^3)/(v*a);
        hc(i)=(0.02738/X(i))*(0.825+0.324*Ra(i)^(1/6))^2;
    end
end
printf("\n\tx,m\t\tRa\t\thc,W/(sq.m.K)\n");
for i=1:m
    printf("\t%.2f\t\t%.2e\t%.2f\n",X(i),Ra(i),hc(i));
end
```

Scilab code Exa 8.5 Natural convection on a horizontal flat surface

```
clc;
clear;
printf("\t\t\tChapter8_example5\n\n\n");
// determine if heat is lost lose more heat through
   its upper surface or one of its vertical sides
// properties of air at (100 + 60)/2 = 80 F = 540
   degree R from appendix table D1
rou= 0.0735; // density in lbm/cu.ft
cp=0.240; // specific heat BTU/(lbm-degree Rankine)
v= 16.88e-5; // viscosity in sq.ft/s
kf = 0.01516 ; // thermal conductivity in BTU/(hr.ft
   .degree Rankine)
a = 0.859; // diffusivity in sq.ft/hr
Pr = 0.708; // Prandtl Number
Tw=100; // temperature of outside surface
   temperature of oven in degree F
T_inf=60; // ambient temperature  in degree F
g=32.2;
L=2; // length in ft
W=2; // width in ft
Beta=1/(T_inf+460); // volumetric thermal expansion
   coefficient in per degree Rankine
printf("\nThe volumetric thermal expansion
   coefficient is %.5f /degree R",Beta);
Ra=(g*Beta*(Tw-T_inf)*L^3)/(v*a/3600);
printf("\nThe Rayleigh Number is %.2e ",Ra);
hc=(kf/L)*(0.68+(0.670*Ra^(1/4))/(1+(0.492/Pr)
   ^(9/16))^(4/9));
printf("\nThe value of convection coefficient is %.3
   f BTU/(hr.sq.ft.degree R)",hc);
```

```
q1side=hc*L*W*(Tw-T_inf);
printf("\nThe heat transferred from one side is %.1f BTU/hr",q1side);
// For the top, we have a heated  surface facing upward, The characteristic length is determined as follows
Lc=(2*2)/(2+2+2+2);
Ra_L=(g*Beta*(Tw-T_inf)*Lc^3)/(v*a/3600); // Rayleigh number based on characteristic length
printf("\nThe Rayleigh Number based on characteristic length is %.2e ",Ra_L);
hc_L=(kf/Lc)*0.54*(Ra_L)^(1/4);
printf("\nThe convective coefficient based on characteristic length is %.3f BTU/(hr.sq.ft.degree R)",hc_L);
qtop=hc_L*L*W*(Tw-T_inf);
printf("\nThe heat transferred from top is %d BTU/hr",qtop);
if qtop>q1side then
    printf("\nThe top transfers heat at a higher rate");
elseif qtop<q1side
    printf("\nThe side transfers heat at a higher rate");
    else printf("\nThe top and side transfer heat at equal rates");
end
```

Scilab code Exa 8.6 Natural convection on cylinders

```
clc;
clear;
printf("\t\t\tChapter8_example6\n\n\n");
```

```
// determination of heat lost from the insulation by
    convection
// properties of air at (50 + 5)/2 = 27.5 degree C =
    300 K from appendix table D1
rou= 1.177; // density in kg/cu.m
cp= 1005.7; // specific heat in J/(kg*K)
v= 15.68e-6; // viscosity in sq.m/s
Pr =0.708; // Prandtl Number
kf= 0.02624; // thermal conductivity in W/(m.K)
a = 0.22160e-4; // diffusivity in sq.m/s
g=9.81;
L=4; // length in m
D=15/100; // diameter in m
T_inf=5; // ambient air temperature in degree C
Tw=50; // outside surface temperature in degree C
Beta=1/(T_inf+273); // volumetric thermal expansion
    coefficient in per K
printf("\nThe volumetric thermal expansion
    coefficient is %.5f /K",Beta);
Ra=(g*Beta*(Tw-T_inf)*D^3)/(v*a);
printf("\nThe Rayleigh Number is %.2e ",Ra);
// for horizontal pipe, the convective coefficient
    is determined as follows
hc_h=(kf/D)*(0.60+(0.387*Ra^(1/6))/(1+(0.559/Pr)
    ^(9/16))^(8/27))^2;
printf("\nThe convection coefficient for horizontal
    length is %.2f W/(sq.m.K)",hc_h);
As=%pi*D*L;
q_hor=hc_h*As*(Tw-T_inf);
printf("\nThe heat transferred from the horizontal
    length of 4 m is %d W",q_hor);
// for vertical pipe, the convective coefficient is
    determined as follows
hc_v=(kf/D)*0.6*(Ra*(D/L))^(1/4);
printf("\nThe convection coefficient for vertical
    length is %.2f W/(sq.m.K)",hc_v);
q_ver=hc_v*As*(Tw-T_inf);
printf("\nThe heat transferred from the vertical
```

```
   length of 4 m is %d W",q_ver);
q=q_ver+q_hor;
printf("\nThe total heat lost from the pipe is %d W"
   ,q);
```

Scilab code Exa 8.7 Natural convection around blocks

```
clc;
clear;
printf("\t\t\tChapter8_example7\n\n\n");
//  Determinion of the convection coefficient about
   the ice cube
// properties of air at (0 + 70)/2 = 35  F  == 495
   degree R from appendix table D1
rou= 0.0809; // density in lbm/cu.ft
cp=0.240; // specific heat BTU/(lbm-degree Rankine)
v= 13.54e-5; // viscosity in sq.ft/s
kf = 0.01402 ; // thermal conductivity in BTU/(hr.ft
   .degree Rankine)
a = 0.685; // diffusivity in sq.ft/hr
Pr = 0.712; // Prandtl Number
Tw=0; // temperature of outside surface temperature
   of oven in degree F
T_inf=70; // ambient temperature  in degree F
g=32.2;
Beta=1/(T_inf+460); // volumetric thermal expansion
   coefficient in per degree Rankine
printf("\nThe volumetric thermal expansion
   coefficient is %.5f /degree R",Beta);
//  The characteristic length is found by using King
    Equation
Lc=1/((1/1)+(1/1.2));
printf("\nThe characteristic length is %.3f ft",Lc);
```

```
Ra=(g*Beta*abs(Tw-T_inf)*Lc^3)/(v*a/3600);
printf("\nThe Rayleigh Number is %.2e ",Ra);
hc=(kf/Lc)*0.6*(Ra)^(1/4);
printf("\nThe value of convection coefficient is %.2
    f BTU/(hr.sq.ft.degree R)",hc);
```

Scilab code Exa 8.8 Natural convection about an array of fins

```
clc;
clear;
printf("\t\t\tChapter8_example8\n\n\n");
// determination of the maximum amount of heat that
     fins can transfer
// properties of air at (100 + 35)/2 = 67.5 degree C
     from appendix table D1
rou= 0.998; // density in kg/cu.m
cp= 1009.0; // specific heat in J/(kg*K)
v= 20.76e-6; // viscosity in sq.m/s
Pr =0.697; // Prandtl Number
kf= 0.03003; // thermal conductivity in W/(m.K)
a = 0.2983e-4; // diffusivity in sq.m/s
g=9.81;
T_inf=35; // ambient air temperature in degree C
Tw=100; // surface temperature in degree C
Beta=1/(T_inf+273); // volumetric thermal expansion
    coefficient in per K
printf("\nThe volumetric thermal expansion
    coefficient is %.5f /K",Beta);
// properties of aluminium from appendix table B1
rou_Al=2702; // density in kg/cu.m
k_Al=236; // thermal conductivity in W/(m.K)
cp_Al=896;// specific heat in J/(kg*K)
a_Al=97.5e-6; // diffusivity in sq.m/s
```

```
b=46/100;
w=24/100;
// Applying the Bar-Cohen Equations
zeta=((w*v^2)/(g*Beta*(Tw-T_inf)*Pr))^(1/4);
printf("\nThe value of zeta is %.2e ",zeta);
L=1.54*(k_Al/kf)^(1/2)*zeta;
printf("\nThe fin length is %.3f m",L);
S=2.89*zeta;
printf("\nThe fin spacing is %.5f m",S);
q=(b*w*(Tw-T_inf)*1.3*(k_Al*kf)^(1/2))/(6*zeta);
printf("\nThe heat transfer rate is %d W",q);
N=b/(2*S);
printf("\nThe number of fins can be atmost %d",N);
```

Chapter 9

Heat Exchangers

Scilab code Exa 9.1 Log Mean Temperature Difference

```
clc;
clear;
printf("\t\t\tChapter9_example1\n\n\n");
// determination of counterflow and parallel-flow configurations.
// temperatures of hot fluid in degree C
T1=100;
T2=75;
// temperatures of cold fluid in degree C
t1=5;
t2=50;
// for counterflow
LMTD_counter=((T1-t2)-(T2-t1))/(log((T1-t2)/(T2-t1)));
printf("\nThe LMTD for counter flow configuration is %.1f degree C",LMTD_counter);
// for parallel flow
LMTD_parallel=((T1-t1)-(T2-t2))/(log((T1-t1)/(T2-t2)));
```

```
printf("\nThe LMTD for parallel flow configuration
   is %.1f degree C",LMTD_parallel);
```

Scilab code Exa 9.2 LMTD for equal outlet temperatures

```
clc;
clear;
printf("\t\t\tChapter9_example2\n\n\n");
// Determination of the LMTD for both counterflow
   and parallel-flow configurations.
// temperatures of hot fluid in degree F
T1=250;
T2=150;
// temperatures of cold fluid in degree F
t1=100;
t2=150;
// for counterflow
LMTD_counter=((T1-t2)-(T2-t1))/(log((T1-t2)/(T2-t1))
   );
printf("\nThe LMTD for counter flow configuration is
    %.1f degree C",LMTD_counter);
// for parallel flow
printf("\nFor a finite heat-transfer rate and a
   finite overall heat-transfer coefficient,\nif
   parallel flow is to give equal outlet
   temperatures,\nthen the area needed must be
   infinite which is not feasible economically.");
```

Scilab code Exa 9.3 Double Pipe Heat Exchanger

```
clc;
clear;
printf("\t\t\tChapter9_example3\n\n\n");
//  Determination of the  outlet  temperature of the  ethylene glycol  for counterflow.
// properties of air at (195 + 85)/2 = 140  F. from appendix table CII
rou_1= 0.985*62.4; // density in lbm/ft^3
cp_1=0.9994; // specific heat BTU/(lbm-degree Rankine)
v_1= 0.514e-5; // viscosity in ft^2/s
kf_1 = 0.376 ; // thermal conductivity in BTU/(hr.ft.degree Rankine)
a_1 = 6.02e-3; // diffusivity in ft^2/hr
Pr_1 = 3.02; // Prandtl Number
m_1=5000; // mass flow rate in lbm/hr
T_1=195; // temperature in degree F
// properties of ethylene glycol at 140 degree F from Appendix Table C.5
rou_2= 1.087*62.4; // density in lbm/ft^3
cp_2=0.612; // specific heat BTU/(lbm-degree Rankine)
v_2= 5.11e-5; // viscosity in ft^2/s
kf_2 = 0.150 ; // thermal conductivity in BTU/(hr.ft.degree Rankine)
a_2 = 3.61e-3; // diffusivity in ft^2/hr
Pr_2 = 51; // Prandtl Number
m_2=12000; // mass flow rate in lbm/hr
T_2=85; // temperature in degree F
// specifications of seamless copper  water tubing (subscripts: a = annulus, p = inner pipe or tube)  from appendix table F2
ID_a=0.1674;
ID_p=0.1076;
OD_p=1.375/12;
// Flow Areas
A_p=%pi*ID_p^2/4;
A_a=%pi*((ID_a)^2-(OD_p)^2)/4;
```

```
printf("\nThe area of annulus is %.5f sq.ft",A_a);
printf("\nThe area of inner pipe is %.5f sq.ft",A_p)
    ;
if A_a>A_p then
    printf("\nAir flows through annulus");
    else printf("\ncarbon dioxide flows through
        annulus");
end
// Annulus Equivalent Diameters
D_h=ID_a-OD_p;
D_e=(ID_a^2-OD_p^2)/(OD_p);
printf("\nThe Annulus Equivalent Diameter for
    friction is %.4f ft",D_h);
printf("\nThe Annulus Equivalent Diameter for heat
    transfer is %.4f ft",D_e);
// Reynolds Numbers
Re_1=(m_1/3600)*(ID_p)/(v_1*rou_1*A_p);
printf("\nThe Reynolds Number for water is %.1e",
    Re_1);
Re_2=(m_2/3600)*(D_e)/(v_2*rou_2*A_a);
printf("\nThe Reynolds Number for ethylene glycol is
     %.2e",Re_2);
// Nusselt numbers
Nu_1=0.023*(Re_1)^(4/5)*(Pr_1)^0.3;
Nu_2=0.023*(Re_2)^(4/5)*(Pr_2)^0.4;
printf("\nThe Nusselt number for water is %d",Nu_1);
printf("\nThe Nusselt number for ethylene glycol is
    %d",Nu_2);
// Convection Coefficients
h_1i=Nu_1*kf_1/ID_p;
h_1o=h_1i*ID_p/OD_p;
h_2=Nu_2*kf_2/D_e;
printf("\nThe convective coefficient for water based
     on inner diameter is %d BTU/(hr.ft^2.degree R)",
    h_1i);
printf("\nThe convective coefficient for water based
     on outer diameter is %d BTU/(hr.sq.ft.degree R)"
    ,h_1o);
```

```
printf("\nThe convective coefficient for ethylene
    glycol is %d BTU/(hr.sq.ft.degree R)",h_2);
// Exchanger Coefficient
Uo=1/((1/h_1o)+(1/h_2));
printf("\nThe overall exchanger coefficient is %d
    BTU/(hr.sq.ft.degree R)",Uo);
R=(m_2*cp_2)/(m_1*cp_1);
L=20;
A=%pi*OD_p*L;
printf("\nThe ratio is %.2f and area is %.1f sq.ft",
    R,A);
T1=195;
t1=85;
T2=((T1*(R-1))-(R*t1*(1-exp((Uo*A*(R-1))/(m_2*cp_2))
    )))/(R*exp(Uo*A*(R-1)/(m_2*cp_2))-1);
printf("\nThe temperature T2=%d degree F",T2);
t2=t1+(T1-T2)/R;
printf("\nThe outlet temperature of Ethylene glycol
    is %.1f degree F",t2);
```

Scilab code Exa 9.4 Fouling Factors in Double Pipe Heat Exchangers

```
clc;
clear;
printf("\t\t\tChapter9_example4\n\n\n");
// Determination of (a) no. of exchangers required,
    (b) the overall coefficient of (all) the
    exchanger(s), and (c) the pressure drop for each
    stream.
// assuming counterflow arrangement
// properties of air at 323 K. from appendix table
    D1
rou_1= 1.088; // density in kg/m^3
```

```
cp_1= 1007; // specific heat in J/(kg*K)
v_1= 18.2e-6; // viscosity in m^2/s
Pr_1 =0.703; // Prandtl Number
kf_1= 0.02814; // thermal conductivity in W/(m.K)
a_1 = 0.26e-4; // diffusivity in m^2/s
m_1=100; // mass flow rate in kg/hr
// temperatures in K
t1_air=20+273;
t2_air=80+273;
// properties of carbon dioxide at [600 + (20 + 273)]/2 = 480 = 500 K. from appendix table D2
rou_2= 1.0732; // density in kg/m^3
cp_2= 1013; // specific heat in J/(kg*K)
v_2= 21.67e-6; // viscosity in m^2/s
Pr_2 =0.702; // Prandtl Number
kf_2= 0.03352; // thermal conductivity in W/(m.K)
a_2 = 0.3084e-4; // diffusivity in m^2/s
m_2=90; // mass flow rate in kg/hr
// temperatures in K
T1_CO2=600;
// specifications of seamless copper tubing from appendix table F2
ID_a=.098;
ID_p=.07384;
OD_p=.07938;
// Flow Areas
A_p=%pi*ID_p^2/4;
A_a=%pi*((ID_a)^2-(OD_p)^2)/4;
printf("\nThe area of annulus is %.2e sq.m",A_a);
printf("\nThe area of inner pipe is %.2e sq.m",A_p);
if A_a>A_p then
    printf("\nAir flows through annulus");
    else printf("\nair flows through inner pipe");
end
// Heat Balance
q_air=(m_1/3600)*(cp_1)*(t2_air-t1_air);
printf("\nThe heat transferred is %.2e W",q_air);
T2_CO2=T1_CO2-(q_air/(m_2*cp_2/3600));
```

```
printf("\nThe low temperature of carbon dioxide is %d K",T2_CO2);
// Log-Mean Temperature Difference
LMTD_counter=((T1_CO2-t2_air)-(T2_CO2-t1_air))/(log((T1_CO2-t2_air)/(T2_CO2-t1_air)));
printf("\nThe LMTD for counter flow configuration is %d degree C",LMTD_counter);
// Annulus Equivalent Diameters
D_h=ID_a-OD_p;
D_e=(ID_a^2-OD_p^2)/(OD_p);
printf("\nThe Annulus Equivalent Diameter for friction is %.5f m",D_h);
printf("\nThe Annulus Equivalent Diameter for heat transfer is %.4f m",D_e);
// Reynolds Numbers
Re_1=(m_1/3600)*(ID_p)/(v_1*rou_1*A_p);
printf("\nThe Reynolds Number for air is %.2e",Re_1);
Re_2=(m_2/3600)*(D_e)/(v_2*rou_2*A_a);
printf("\nThe Reynolds Number for carbon dioxide is %.2e",Re_2);
// Nusselt numbers
Nu_1=0.023*(Re_1)^(4/5)*(Pr_1)^0.3;
Nu_2=0.023*(Re_2)^(4/5)*(Pr_2)^0.4;
printf("\nThe Nusselt number for air is %.1f",Nu_1);
printf("\nThe Nusselt number for carbon dioxide is %.1f",Nu_2);
// Convection Coefficients
h_1i=Nu_1*kf_1/ID_p;
h_1o=h_1i*ID_p/OD_p;
h_2=Nu_2*kf_2/D_e;
printf("\nThe convective coefficient for air based on inner diameter is %.1f W/(sq.m.K)",h_1i);
printf("\nThe convective coefficient for air based on outer diameter is %.1f W/(sq.m.K)",h_1o);
printf("\nThe convective coefficient for carbon dioxide is %.1f W/(sq.m.K)",h_2);
// Fouling Factors in (m^2.K)/W
```

```
Rd_air=.0004;
Rd_CO2=0.002;
// exchanger coefficients
Uo=1/((1/h_1o)+(1/h_2));
Uo=1/((1/Uo)+Rd_air+Rd_CO2);
printf("\nThe overall exchanger coefficient is %.1f
   W/(sq.m.K)",Uo);
// area required
A=q_air/(Uo*LMTD_counter);
printf("\nThe area required is %.2f sq.m",A);
// surface area of one exchanger is A=%pi*OD*L, so
L=(A/(%pi*OD_p)); // length of each exchanger
L_available=2; // available exchanger length
N=L_available/L; // no. of exchangers
printf("\nThe number of exchangers is %d",N);
//friction factors
fp=0.0245; //friction factor for air fom figure 6.14
     corresponding to Reynolds Number calculated
    above
fa=0.033; //friction factor for carbon dioxide fom
    figure 6.14 corresponding to Reynolds Number
    calculated above
// Velocities
V_air=(m_1/3600)/(rou_1*A_p);
V_CO2=(m_2/3600)/(rou_2*A_a);
printf("\nThe velocity of air is %.2f m/s",V_air);
printf("\nThe velocity of carbon dioxide is %.2f m/s
    ",V_CO2);
// pressure drops
dP_p=(fp*L_available*rou_1*V_air^2)/(ID_p*2);
dP_a=((rou_2*V_CO2^2)/2)*((fa*L_available/D_h)+1);
printf("\nThe pressure drop for tube side is %.2f Pa
    ",dP_p);
printf("\nThe pressure drop for shell side is %d Pa"
    ,dP_a);
printf("\n\t\t\tSummary of Requested Information\n")
    ;
printf("(a) Exchanger required: %d\n(b)Overall
```

```
exchanger coefficient = %.1f W/(sq.m.K)\n(c)Air
pressure drop = %.2f Pa\nDiesel exhaust pressure
drop = %d Pa",N,Uo,dP_p,dP_a);
```

Scilab code Exa 9.5 Shell and Tube heat exchangers

```
clc;
clear;
printf("\t\t\tChapter9_example5\n\n\n");
//  Determination of the outlet temperature of the
    distilled  water and the pressure drop for each
    stream.
// properties of (distilled) water at 104 F from
    appendix table CII
rou_1= 0.994*62.4; // density in lbm/ft^3
cp_1=0.998; // specific heat BTU/(lbm-degree Rankine
    )
v_1= 0.708e-5; // viscosity in ft^2/s
kf_1 = 0.363 ; // thermal conductivity in BTU/(hr.ft
    .degree Rankine)
a_1 = 5.86e-3; // diffusivity in ft^2/hr
Pr_1 = 4.34; // Prandtl Number
m_1=170000; // mass flow rate in lbm/hr
T1=110; // temperature in degree F
// properties of (raw) water at 68 F from Appendix
    Table C11
rou_2= 62.4; // density in lbm/ft^3
cp_2=0.9988; // specific heat BTU/(lbm-degree
    Rankine)
v_2= 1.083e-5; // viscosity in ft^2/s
kf_2 = 0.345 ; // thermal conductivity in BTU/(hr.ft
    .degree Rankine)
a_2 = 5.54e-3; // diffusivity in ft^2/hr
```

```
Pr_2 = 7.02; // Prandtl Number
m_2=150000; // mass flow rate in lbm/hr
t1=65; // temperature in degree F
// specifications of 3/4-in-OD, 18-BWG tubes, from table 9.2
OD=3/(4*12);
ID=0.652/12;
OD_p=1.375/12;
Nt=224; // from table 9.3
Np=2; // no. of tube passes
// Shell dimensions and other miscellaneous data
Ds=17.25/12;
Nb=15; // no. of baffles
B=1;
sT=15/(16*12);
C=sT-OD;
// flow areas
At=(Nt*%pi*ID^2)/(4*Np);
As=(Ds*C*B)/sT;
printf("\nThe areas are %.3f sq.ft and %.3f sq.ft",At,As);
if At>As then
    printf("\nThe distilled water flows through the tubes");
    else printf("\nThe raw water flows through the tubes");
end
// Shell Equivalent Diameter
De=4*[(sT/2)*(0.86*sT)-(%pi*OD^2/8)]/(%pi*OD/2);
printf("\nThe equivalent diameter is %.4f ft",De);
// Reynolds Numbers
Re_s=(m_1/3600)*(De)/(v_1*rou_1*As);
printf("\nThe Reynolds Number for raw water is %.2e",Re_s);
Re_t=(m_2/3600)*(ID)/(v_2*rou_2*At);
printf("\nThe Reynolds Number for distilled water is %.2e",Re_t);
// Nusselt numbers
```

```
Nu_t=0.023*(Re_t)^(4/5)*(Pr_2)^0.4;
Nu_s=0.36*(Re_s)^(.55)*(Pr_1)^(1/3);
printf("\nThe Nusselt number for raw water is %.1f",Nu_t);
printf("\nThe Nusselt number for distilled water is %.1f",Nu_s);
h_ti=Nu_t*kf_2/ID;
h_to=h_ti*ID/OD;
h_s=Nu_s*kf_1/De;
printf("\nThe convective coefficient for raw water based on inner diameter is %d BTU/(hr.sq.ft.degree R)",h_ti);
printf("\nThe convective coefficient for raw water based on outer diameter is %d BTU/(hr.sq.ft.degree R)",h_to);
printf("\nThe convective coefficient for distilled water is %d BTU/(hr.sq.ft.degree R)",h_s);
// Exchanger Coefficient
Uo=1/((1/h_to)+(1/h_s));
printf("\nThe overall exchanger coefficient is %d BTU/(hr.sq.ft.degree R)",Uo);
R=(m_2*cp_2)/(m_1*cp_1);
L=16;
Ao=Nt*%pi*OD*L;
printf("\nThe ratio is %.3f and area is %.1f sq.ft",R,Ao);
UoAo_mccp=(Uo*Ao)/(m_2*cp_2);
printf("\n(UoAo)/(McCpc)=%.2f",UoAo_mccp);
S=0.58; //value of S from fig. 9.13 Ten Broeck graph corresponding to the value of (UoAo)/(McCpc)
t2=S*(T1-t1)+t1;
T2=T1-R*(t2-t1);
printf("\nt2=%.1f degree F\nT2=%.1f degree F",t2,T2);
//friction factors
ft=0.029; //friction factor for raw water fom figure 6.14 corresponding to Reynolds Number calculated above
```

```
printf("\nFriction factor for raw water fom figure 6.14 corresponding to Reynolds Number calculated above is %.3f",ft);
fs=0.281; //friction factor for distilled water fom figure 6.14 corresponding to Reynolds Number calculated above
printf("\nFriction factor for distilled water fom figure 6.14 corresponding to Reynolds Number calculated above is %.3f",fs);
// Velocities
V_t=(m_2/3600)/(rou_2*At);
V_s=(m_1/3600)/(rou_1*As);
printf("\nThe velocity of raw water is %.2f ft/s",V_t);
printf("\nThe velocity of distilled water is %.2f ft/s",V_s);
// pressure drops
gc=32.2;
dP_t=(rou_2*V_t^2)*((ft*L*Np/ID)+4*Np)/(2*gc);
dP_s=((rou_1*V_s^2)*(fs*Ds*(Nb+1)))/(2*gc*De);
printf("\nThe pressure drop for tube side is %.1f lbf/sq.ft = %.1f psi",dP_t,dP_t/147);
printf("\nThe pressure drop for shell side is %.1f lbf/sq.ft = %.1f psi",dP_s,dP_s/147);
printf("\n\t\t\tSummary of Requested Information\n");
printf("\nOutlet Temperatures:\n\tRaw Water: %.1f degree F\n\tDistilled Water: %.1f degree F\n",t2,T2);
printf("\nPressure Drops:\n\tRaw Water: %.1f ddegree F\n\tDistilled Water: %.1f degree F\n",dP_t/147,dP_s/147);
```

Scilab code Exa 9.6 Effectiveness NTU method of analysis

```
clc;
clear;
printf("\t\t\tChapter9_example6\n\n\n");
// Using the effectiveness-NTU method to calculate the outlet temperatures of the fluids
// Data from Example 9.5
// properties of (distilled) water at 104 F
m_1=170000; // mass flow rate in lbm/hr
T1=110; // temperature in degree F
cp_1=0.998; // specific heat BTU/(lbm-degree Rankine)
// properties of (raw) water at 68 F
m_2=150000; // mass flow rate in lbm/hr
t1=65; // temperature in degree F
cp_2=0.9988; // specific heat BTU/(lbm-degree Rankine)
Uo=350; // exchanger coefficient
Ao=703.7;
// The effectiveness-NTU approach is used  when the overall heat transfer coefficient is known
// determining the capacitances
mcp_raw=m_2*cp_2;
mcp_distilled=m_1*cp_1;
printf("\nThe capacitance value of raw water is %d BTU/(hr. degree R)",mcp_raw);
printf("\nThe capacitance value of distilled water is %d BTU/(hr. degree R)",mcp_distilled);
if mcp_raw>mcp_distilled then
    mcp_max=mcp_raw;
    mcp_min=mcp_distilled;
    printf("\nDistilled water has minimum capacitance");
    else mcp_max=mcp_distilled;
    mcp_min=mcp_raw;
    printf("\nRaw water has minimum capacitance");
end
```

```
// determination of parameters for determining effectiveness
mcp_min_max=mcp_min/mcp_max;
UA_mcpmin=(Uo*Ao)/(mcp_min);
printf("\nThe required parameters are mcp_min/mcp_max=%.3f and (UoAo/mcp_min)=%.2f",mcp_min_max,UA_mcpmin);
effectiveness=0.58; //value of effectiveness from figure 9.15 corresponding to the above calculated values of capacitance ratio and (UoAo/mcp_min)
qmax=mcp_min*(T1-t1);
printf("\nThe maximum heat transfer is %.2e BTU/hr",qmax);
q=effectiveness*qmax; // actual heat transfer
printf("\nThe actual heat transfer is %.2e BTU/hr",q);
t2=(q/mcp_raw)+t1;
T2=T1-(q/mcp_distilled);
printf("\nThe Outlet temperatures are:\n\tRaw Water:%.1f degree F\n\tDistilled Water:%.1f degree F\n",t2,T2);
```

Scilab code Exa 9.7 Crossflow heat exchangers

```
clc;
clear;
printf("\t\t\tChapter9_example7\n\n\n");
// (a) Determine the UA product for the exchanger. (b) Calculate the exit temperatures for the exchanger, assuming that only the inlet temperatures are known
// properties of engine oil at (190 + 158)/2 = 174 F = 176 degree F from appendix table C4
```

```
rou_1= 0.852*62.4; // density in lbm/ft^3
cp_1=0.509; // specific heat BTU/(lbm-degree Rankine
    )
v_1= 0.404e-3; // viscosity in ft^2/s
kf_1 = 0.08; // thermal conductivity in BTU/(hr.ft.
    degree Rankine)
a_1 = 2.98e-3; // diffusivity in ft^2/hr
Pr_1 = 490; // Prandtl Number
m_1=39.8; // mass flow rate in lbm/min
// temperatures in degree F
T1=190;
T2=158;
// properties of air at (126 + 166)/2 = 146 F = 606
     degree R from appendix table D1
rou_2= 0.0653; // density in lbm/ft^3
cp_2=0.241; // specific heat BTU/(lbm-degree Rankine
    )
v_2= 20.98e-5; // viscosity in ft^2/s
kf_2 = 0.01677 ; // thermal conductivity in BTU/(hr.
    ft.degree Rankine)
a_2 = 1.066; // diffusivity in ft^2/hr
Pr_2 = 0.706; // Prandtl Number
m_2=67; // mass flow rate in lbm/min
// temperatures in degree F
t1=126;
t2=166;
// Heat Balance
q_air=m_2*cp_2*60*(t2-t1);
q_oil=m_1*cp_1*60*(T1-T2);
printf("\nThe heat gained by air is %.2e BTU/hr",
    q_air);
printf("\nThe heat lost by oil is %.2e BTU/hr",q_oil
    );
// for counterflow
LMTD=((T1-t2)-(T2-t1))/(log((T1-t2)/(T2-t1)));
printf("\nThe LMTD for counter flow configuration is
     %.1f degree F",LMTD);
// Frontal Areas for Each Fluid Stream
```

```
Area_air=(9.82*8)/144;
Area_oil=(3.25*9.82)/144;
printf("\nThe Core frontal area on the air side is %.3f sq.ft\nThe Core frontal area on the oil side is %.3f sq.ft ",Area_air,Area_oil);
// Correction Factors (parameters calculated first)
S=(t2-t1)/(T1-t1);
R=(T1-T2)/(t2-t1);
F=0.87; //value of correction factor from figure 9.21a corresponding to above calculated values of S and R
// Overall Coefficient (q = U*A*F*LMTD)
UA=q_air/(F*LMTD);
printf("\nThe Overall Coefficient is %.2e BTU/(hr. degree R)",UA);
// determining the capacitances
mcp_air=m_2*cp_2*60;
mcp_oil=m_1*cp_1*60;
printf("\nThe capacitance value of air is %d BTU/(hr. degree R)",mcp_air);
printf("\nThe capacitance value of engine oil is %d BTU/(hr. degree R)",mcp_oil);
if mcp_air>mcp_oil then
    mcp_max=mcp_air;
    mcp_min=mcp_oil;
    printf("\nEngine Oil has minimum capacitance");
    else mcp_max=mcp_oil;
    mcp_min=mcp_air;
    printf("\nAir has minimum capacitance");
end
// determination of parameters for determining effectiveness
mcp_min_max=mcp_min/mcp_max;
NTU=(UA/mcp_min);
printf("\nThe required parameters are mcp_min/mcp_max=%.3f and (UoAo/mcp_min)=%.2f",mcp_min_max,NTU);
effectiveness=0.62; //value of effectiveness from
```

```
    figure 9.21b corresponding to the above
    calculated values of capacitance ratio and (UoAo/
    mcp_min):');
t2_c=(T1-t1)*effectiveness+t1;
T2_c=T1-(mcp_min_max)*(t2_c-t1);
printf("\n\t\t\tSummary of Requested Information\n")
    ;
printf("\n(a) UA = %.2e BTU/(hr. degree R)",UA);
printf("\n(b) The Outlet temperatures (degree F)");
printf("\n\tCalculated\tGiven in Problem Statement")
    ;
printf("\nAir\t\t%d\t%d",t2_c,t2);
printf("\nEngine Oil\t%d\t%d",T2_c,T2);
```

Chapter 10

Condensation and Vaporization Heat Transfer

Scilab code Exa 10.1 Laminar film condensation on a vertical flat surface

```
clc;
clear;
printf("\t\t\tChapter10_example1\n\n\n");
// Calculation of the heat-transfer rate and the
   amount of steam condensed
// properties of engine oil at (328 + 325)/2 = 326.5
    degree F = 320 F from appendix table C11
rou_f= 0.909*62.4; // density in lbm/ft^3
cp=1.037; // specific heat BTU/(lbm-degree Rankine)
v_f= 0.204e-5; // viscosity in ft^2/s
kf = 0.393; // thermal conductivity in BTU/(lbm.ft.
   degree Rankine)
a = 6.70e-3; // diffusivity in ft^2/hr
Pr = 1.099; // Prandtl Number
V_v=4.937; // specific volume in ft^3/lbm from
```

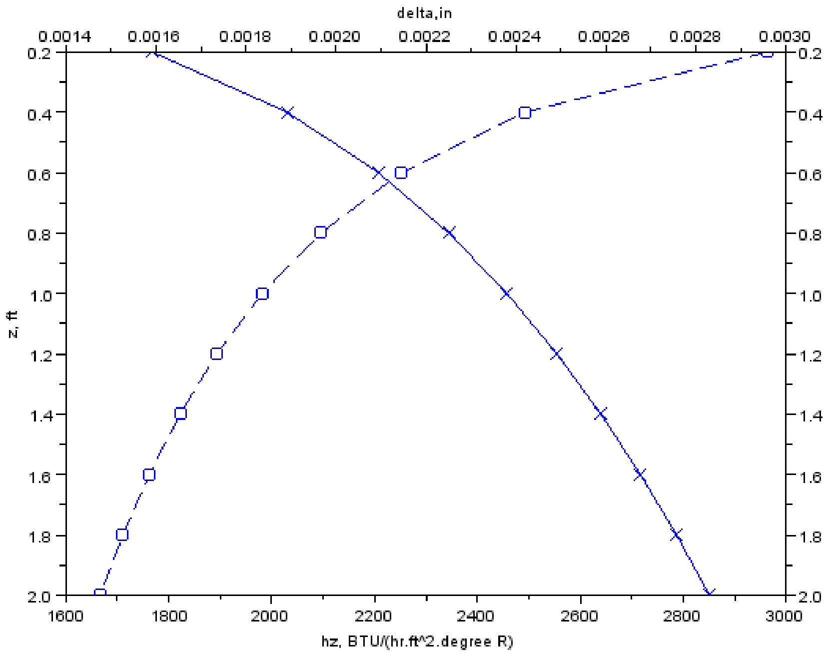

Figure 10.1: Laminar film condensation on a vertical flat surface

```
superheated steam tables
rou_v=1/V_v; // vapour density
g=32.2;
hfg=888.8; // from saturated steam tables
Tg=327.81;
Tw=325;
L=2; // length in ft
W=3; // width in ft
z=0.2:0.2:2; // distance from entry of plate in ft
[n m]=size(z);
// film thickness is given as follows
for i=1:m
delta(i)=[(4*kf*v_f*z(i)*(Tg-Tw)/3600)/(rou_f*g*hfg*(1-(rou_v/rou_f)))]^(1/4);
hz(i)=(kf/delta(i));
end
printf("\nGrowth of and Heat-Transfer Coefficient for the Condensate Film of Example 10.1 ");
printf("\nz, ft\tdelta, ft\tdelta, in\thz, BTU/(hr.sq.ft.degree Rankine)");
for i=1:m
printf("\n%.1f\t%.2e\t%.4f\t\t%d\n",z(i),delta(i),12*delta(i),hz(i));
end
hL=(4/3)*hz(m); // at plate end
mf=(hL*L*W*(Tg-Tw))/hfg;
printf("\nThe convective coefficient at the plate end is %d BTU/(hr.sq.ft. degree Rankine)",hL);
printf("\nThe amount of steam condensed is %.1f lbm/hr",mf);
q=mf*hfg;
printf("\nThe heat transfer rate is %.2e BTU/hr",q);
Re=(4*mf/3600)/(W*rou_f*v_f);
printf("\nThe Reynolds Number is %d",Re);
if Re<1800 then
    printf("\nThe film is laminar and above equations apply");
    else printf("\nThe film is not laminar and above
```

```
        assumption is wrong");
end
subplot(211);
plot(delta*12,z,'x-');  // our first figure
a1 = gca();
a1.x_location="top";
a1.axes_reverse=["off","on"];
subplot(212)
plot(hz,z, 'o--');  // our second figure
a2 = gca();
a2.axes_reverse=["off","on"];
a2.x_location="bottom";
a2.axes_visible = ["on", "on","on"];
a2.y_location ="right";
x_label1=a1.x_label;
x_label1.text="delta ,in";
x_label2=a2.x_label;
x_label2.text="hz, BTU/(hr.sq.ft.degree R)";
y_label=a1.y_label;
y_label.text="z, ft";
a1.axes_bounds=[0 0 1 1];  // modify the first
   figure to occupy the whole area
a2.axes_bounds=[0 0 1 1]; // modify the second
   figure to occupy the whole area too
a2.filled="off";
```

Scilab code Exa 10.2 Film condensation on a horizontal tube

```
clc;
clear;
printf("\t\t\tChapter10_example2\n\n\n");
// Determination of both the heat that the cooling
   fluid must remove and the condensation rate.
```

```
// properties of water at (100 + 60)/2 = 80 C from
    appendix table C11
rou_f= 947; // density in kg/m^3
cp_1= 4196; // specific heat in J/(kg*K)
v_1= 0.364e-6; // viscosity in m^2/s
Pr_1 =2.22; // Prandtl Number
kf= 0.668; // thermal conductivity in W/(m.K)
a_1 =1.636e-7; // diffusivity in m^2/s
Vv=1.9364; // specific volume in m^3/kg
rou_v=1/Vv; // vapor density;
g=9.81;
hfg=2257.06*1000;
Tg=100;
Tw=60;
L=1;
printf("\nThe vapor density is %.3f kg/cu.m",rou_v);
// specifications of 1 nominal schedule 40 pipe from
     appendix F1
OD=.03340;
hD=0.782*[(g*rou_f*(1-(rou_v/rou_f))*(kf^3)*hfg)/(
    v_1*OD*(Tg-Tw))]^(1/4);
printf("\nThe average heat transfer coefficient is %
    .3e W/(sq.m.K)",hD);
q=hD*%pi*OD*L*(Tg-Tw);
printf("\nThe heat flow rate is %.1e W",q);
mf=q/hfg;
printf("\nThe rate at which steam condenses is %.2f
    kg/s = %d kg/hr",mf,.02*3600);
```

Scilab code Exa 10.3 Nucleate pool boiling critical heat flux

```
clc;
clear;
```

```
printf("\t\t\tChapter10_example3\n\n\n");
// Calculation of (a) the power input to the water
   for boiling to occur, (b) the evaporation rate of
    water, and (c) the critical heat flux.
// properties of water at 100 C = 373 K from
   appendix table 10.3
rou_f=958; // density in kg/m^3
cp_f= 4217; // specific heat in J/(kg*K)
v_f= 2.91e-7; // viscosity in m^2/s
Pr_f =1.76; // Prandtl Number
rou_g=0.596;
sigma=0.0589; // surface tension in N/m
hfg=2257000;
Tw=120
Tg=100;
D=.141; // diameter of pan in m
g=9.81;
gc=1;
// nucleate boiling regime
Cw=0.0132; // formechanically polished stainless
   steel from table 10.2
q_A=(rou_f*v_f*hfg)*[(g*rou_f*(1-(rou_g/rou_f)))/(
   sigma*gc)]^(1/2)*[(cp_f*(Tw-Tg))/(Cw*hfg*Pr_f
   ^1.7)]^3;
printf("\nThe heat transferred per unit area is %.2e
    W/sq.m",q_A);
A=%pi*D^2/4;
printf("\nThe area of the pan inside-bottom surface
   in contact with liquid is %.2e sq.m",A);
printf("\n\n\t\t\tSolution to part (a)");
q=q_A*A; // power delivered to the water in W
printf("\nThe power delivered to the water is %.2f
   kW",q/1000);
printf("\n\n\t\t\tSolution to part (b)");
mf=q/hfg; // water evaporation rate
printf("\nThe water evaporation rate is %.2e kg/s =
   %.2f kg/hr",mf,mf*3600);
printf("\n\n\t\t\tSolution to part (c)");
```

```
q_cr=0.18*hfg*[sigma*g*gc*rou_f*rou_g^2]^(1/4);
printf("\nThe critical heat flux is %.2e W/sq.m",
    q_cr);
```

Chapter 11

Introduction to Radiation Heat Transfer

Scilab code Exa 11.1 Radiation Intensity

```
clc;
clear;
printf("\t\t\tChapter11_example1\n\n\n");
// Calculation of the value of the solid angle subtended by surfaces dA2 and dA3 with respect to dA1 (b) the intensity of emission from dA, in the direction of the other areas (c) the rate at which radiation emitted by dA, is intercepted by the other areas
printf("\t\t\tSolution to Part (a)\n");
// solid angle is calculate using the equation dw=dA*cos(Beta)/r^2
// Beta is the angle between the surface normal of a receiver surface and the line connecting the two surfaces
// For area A2
// dimensions are 1X1 in, so
```

```
dA2=(1*1)/144;
Beta1=40*%pi/180;
r=4;
dw2_1=dA2*cos(Beta1)/r^2;
printf("\nThe solid angle subtended by area dA2 with
    respect to dA1 is %.2e sr",dw2_1);
dA3=dA2;
Beta2=0;
dw3_1=dA3*cos(Beta2)/r^2;
printf("\nThe solid angle subtended by area dA3 with
    respect to dA1 is %.2e sr",dw3_1);
printf("\n\n\t\t\tSolution to Part (b)\n");
theta2=%pi*50/180;
theta3=%pi*60/180;
I_theta2=2000*(1-0.4*(sin(theta2))^2);
I_theta3=2000*(1-0.4*(sin(theta3))^2);
printf("\n The intensity of radiation emitted from
   dA1 in the direction of dA2 is %d BTU/(hr.sq.ft.
   sr)",I_theta2);
printf("\n The intensity of radiation emitted from
   dA1 in the direction of dA3 is %d BTU/(hr.sq.ft.
   sr)",I_theta3);
printf("\n\n\t\t\tSolution to Part (c)\n");
dA1=1/144;
dq1_2=I_theta2*dA1*cos(theta2)*dw2_1;
dq1_3=I_theta3*dA1*cos(theta3)*dw3_1;
printf("\nThe rate at which radiation emitted by dA1
    is intercepted by dA2 is %.2e BTU/hr",dq1_2);
printf("\nThe rate at which radiation emitted by dA1
    is intercepted by dA3 is %.2e BTU/hr",dq1_3);
```

Scilab code Exa 11.2 Irradiation and Radiosity

```
clc;
clear;
printf("\t\t\tChapter11_example2\n\n\n");
// Calculation of the value of the solid angle subtended by surfaces dA2 with respect to dA1 (b) the rate at which radiation emitted by dA1 is intercepted by dA2 (c)  the irradiation associated with dA2
printf("\t\t\tSolution to Part (a)\n");
// solid angle is calculate using the equation dw=dA*cos(Beta)/r^2
// The angle Beta is 0 because the surface normal of dA2 is directed at dA1
dA2=0.02*0.02;
Beta=0;
r=1;
dw2_1=dA2*cos(Beta)/r^2;
printf("\nThe solid angle subtended by area dA2 with respect to dA1 is %.2e sr",dw2_1);
printf("\n\n\t\t\tSolution to Part (b)\n");
dA1=dA2;
theta=%pi*30/180;
I_theta=1000;// The intensity of radiation leaving dA1 in any direction is 1 000 W/(m^2.sr
dq1_2=I_theta*dA1*cos(theta)*dw2_1;
printf("\nThe rate at which radiation emitted by dA1 is intercepted by dA2 is %.2e W",dq1_2);
printf("\n\n\t\t\tSolution to Part (c)\n");
// The irradiation associated with dA2 can be found by dividing the incident radiation by the receiver area
dQ1_2=dq1_2/dA2;
printf("\nThe irradiation associated with dA2 is %.2e W/sq.m",dQ1_2);
```

Scilab code Exa 11.3 Emissivity and Rate of radiant emission

```
clc;
clear;
printf("\t\t\tChapter11_example3\n\n\n");
// (a) Calculation of the emissivity of the hole.(b)
    the rate of radiant emission from the hole
D=2.5/12; // diameter in ft
L=4.5/12; // length in ft
A=(2*%pi*D^2/4)+(%pi*D*L);
printf("\nThe inside surface area is %.3f sq.ft ",A)
    ;
A_hole=%pi*(1/(8*12))^2/4;
printf("\nThe area of a 1/8 inch hole is %.3e sq.ft"
    ,A_hole);
f=A_hole/A; // fraction of area removed
printf("\nThe fraction of area removed is %.3e ",f);
printf("\n\n\t\t\tSolution to Part (a)\n");
// for rolled and polished aluminum, that emissivity
    = 0.039 from appendix table E1
emissivity=0.039;
emissivity_hole=emissivity/(emissivity+(1-emissivity
    )*f);
printf("\nThe emissivity of the hole is %.4f",
    emissivity_hole);
printf("\n\n\t\t\tSolution to Part (b)\n");
sigma=0.1714e-8; // stefan Boltzmann constant in BTU
    /(hr~ft^2 degree R)
T=150+460; // temperature in degree R
qe=emissivity_hole*sigma*T^4;
printf("\nThe heat lost per unit area of the hole is
    %d BTU/hr",qe);
```

```
Qe=A_hole*qe;
printf("\nThe heat lost by the hole is %.2e BTU/hr",
    Qe);
```

Scilab code Exa 11.4 Plancks distribution law

```
clc;
clear;
printf("\t\t\tChapter11_example4\n\n\n");
// Determination of the percentage of total emitted
    energy that lies in the visible range.
T=2800;
lambda1=4e-7;
lambda2=7e-7;
hT=lambda1*T;
lambdaT=lambda2*T;
printf("\nhT=%.2e m.K and lambda2_T=%.2e m.K",hT,
    lambdaT);
I1=0.0051; //Fraction of Total Radiation Emitted for
     lower Wavelength-Temperature Product from Table
    11.1
I2=0.065; //Fraction of Total Radiation Emitted for
    upper Wavelength-Temperature Product from Table
    11.1
dI=I2-I1;
printf("\nThe percentage of total emitted energy
    that lies in the visible range is %.1f percent",
    dI*100);
```

Scilab code Exa 11.5 Wiens displacement law

```
clc;
clear;
printf("\t\t\tChapter11_example5\n\n\n");
// Estimation of the surface temperature of the sun
   and the emitted heat flux
lambda_max=0.5e-6; // maximum wavelength in m
// From Wien s Displacement Law we can write
   lambda_max*T=2.898e-3 m.K
T=2.898e-3/lambda_max;
printf("\nThe Surface Temperature of the Sun is %d K
   ",T);
// The heat flux is given by the Stefan-Boltzmann
   Equation as q=sigma*T^4
sigma=5.675e-8; // value of Stefan-Boltzmann
   constant in W/(m^2.K^4)
q=sigma*T^4;
printf("\nThe heat flux emitted is %.3e W/sq.m",q);
```

Scilab code Exa 11.6 Transmission through glass windshield

```
clc;
clear;
printf("\t\t\tChapter11_example6\n\n\n");
// (a) Calculation of the rate at which the sun s
   radiant energy is transmitted through the glass
   windshield. The interior of the car is considered
    to be a black body that radiates at 100 F. (b)
   Calculation of the rate at which radiant energy
   from the car interior is transmitted through the
   glass windshield.
printf("\t\t\tSolution to Part (a)\n");
```

```
lambda1=300e-9; // lower limit of wavelength
lambda2=380e-9; // upper limit of wavelength
T=5800;
lambda1_T=lambda1*T;
lambda2_T=lambda2*T;
printf("\nThe Lower and Upper limits of Wavelength-
    Temperature Products are %.2e m.K and %.3e m.K
    respectively",lambda1_T,lambda2_T);
I1=0.101; //Fraction of Total Radiation Emitted for
    lower Wavelength-Temperature Product from Table
    11.1
I2=0.0334; //Fraction of Total Radiation Emitted for
     upper Wavelength-Temperature Product from Table
    11.1
dI=abs(I2-I1);
t=dI*0.68; // transmissivity
printf("\nThe Transmittivity is %.4f",t);
q=1100; // radiation received by car in W/sq.m
q_in=t*q; // energy transmitted from the sun through
     the glass
printf("\nThe energy transmitted from the sun
    through the glass is %.1f W/sq.m",q_in);
printf("\n\t\t\tSolution to Part (b)\n");
Tb=311; // temperature of black body source in K
lambda1_Tb=lambda1*Tb;
lambda2_Tb=lambda2*Tb;
printf("\nThe Lower and Upper limits of Wavelength-
    Temperature Products are %.2e m.K and %.2e m.K
    respectively",lambda1_Tb,lambda2_Tb);
dI_b=0; // Table 11.1 gives negligibly small values
    of the corresponding integrals.
t_b=dI_b*0.68; // transmissivity
q_out=t_b*q;
printf("\nthe rate at which radiant energy from the
    car interior is transmitted through the glass
    windshield is %d W/sq.m",q_out);
```

Chapter 12

Radiation Heat Transfer between Surfaces

Scilab code Exa 12.3 View Factor algebra for pairs of surfaces

```
clc;
clear;
printf("\t\t\tChapter12_example3\n\n\n");
//  Determination of the heat transferred by radiation from dA1 to A.
// The view factor Fd1_2 can be calculated as Fd1_2=Fd1_3+Fd1_4+Fd1_5
// For Fd1_3
a_13=100;
b_13=250;
c_13=100;
X_13=a_13/c_13;
Y_13=b_13/c_13;
printf("\nFor Fd1_3, the values of a/c=%.1f and b/c=%.1f",X_13,Y_13);
Fd1_3=0.17; // value for Fd1_3 corresponding to above calculated values of a/c and b/c
```

```
// For Fd1_4
a_14=300;
b_14=50;
c_14=100;
X_14=a_14/c_14;
Y_14=b_14/c_14;
printf("\nFor Fd1_4, the values of a/c=%.1f and b/c=
   %.1f",X_14,Y_14);
Fd1_4=0.11; //value for Fd1_4 corresponding to above
    calculated values of a/c and b/c
// For Fd1_5
a_15=100;
b_15=50;
c_15=100;
X_15=a_15/c_15;
Y_15=b_15/c_15;
printf("\nFor Fd1_5, the values of a/c=%.1f and b/c=
   %.1f",X_15,Y_15);
Fd1_5=0.09; //value for Fd1_3 corresponding to above
    calculated values of a/c and b/c
Fd1_2=Fd1_3+Fd1_4-Fd1_5;
printf("\nFd1_2=%.2f",Fd1_2);
printf("\n%d percent of the energy leaving dA1
   reaches A",100*Fd1_2);
sigma=0.1714e-8; // Stefan-Boltzmann constant
T1=660;
T2=560;
q12_A1=sigma*Fd1_2*(T1^4-T2^4);
printf("\nThe net heat transferred is %.1f BTU/(hr.
   sq.ft)",q12_A1);
```

Scilab code Exa 12.4 View Factor algebra for enclosures

```
clc;
clear;
printf("\t\t\tChapter12_example4\n\n\n");
// Determination of the heat transferred to the
   conveyed parts for the conditions given
L1=1;
angle=%pi*45/180;
L2=L1*sin(angle);
L3=L2;
printf("\nThe Widths are L1=%d m, L2=%.3f m and L3=%
   .3f m",L1,L2,L3);
T1=303;
T2=473;
sigma=5.67e-8; // Stefan-Boltzmann constant
q21_A2=sigma*(T2^4-T1^4)*((L1/L2)+1-(L3/L2))/2;
q31_A3=sigma*(T2^4-T1^4)*((L1/L2)-1+(L3/L2))/2;
printf("\nThe heat transferred from A2 to A1 is %.2e
    W/sq.m",q21_A2);
printf("\nThe heat transferred from A3 to A1 is %.2e
    W/sq.m",q31_A3);
```

Scilab code Exa 12.5 Crossed String method

```
clc;
clear;
printf("\t\t\tChapter12_example5\n\n\n");
//  Determination of the heat exchanged between the
   two plates
// The view factor can be found with the crossed-
   string method
// from figure 12.13(b)
ac=1;
bd=1;
```

```
ad=(9+1)^0.5;
bc=ad;
crossed_strings=ad+bc;
uncrossed_strings=ac+bd;
L1_F12=(1/2)*(crossed_strings-uncrossed_strings);
printf("\nThe Product L1F12=%.2f ft",L1_F12);
L1=3;
F12=L1_F12/L1;
printf("\nThe view factor F12=%.2f",F12);
sigma=5.67e-8; // Stefan-Boltzmann constant
T1=560;
T2=460;
q12_A1=sigma*(T1^4-T2^4)*F12;
printf("\nThe heat transfer rate is %.2e W/sq.m",
   q12_A1);
```

Scilab code Exa 12.6 Radiation heat transfer within a broiler

```
clc;
clear;
printf("\t\t\tChapter12_example6\n\n\n");
//  Determination of the heat that must be supplied
   to each of the isothermal surfaces, and also the
   temperature of the insulated surface.
// we can apply the equations as follows
// q1=sigma*A1*[(T1^4-T2^4)F12+(T1^4-T3^4)F13].....
   (1)
// q2=sigma*A2*[(T2^4-T1^4)F21+(T2^4-T3^4)F23].....
   (2)
// q3=sigma*A3*[(T3^4-T1^4)F31+(T3^4-T2^4)F32].....
   (3)
// given data:
T1=1000;
```

```
T3=500;
q2=0;
F12=1/2;
F13=1/2;
F21=1/2;
F23=1/2;
F31=1/2;
F32=1/2;
T2=[(T1^4+T3^4)/2]^(1/4); // using equation (2)
printf("\nThe temperature T2=%.1f degree R",T2);
sigma=0.1714e-8; // Stefan-Boltzmann constant
q1_A1=sigma*[(T1^4-T2^4)*F12+(T1^4-T3^4)*F13]; // using equation (1)
printf("\nThe heat flux through area A1 is %d BTU/(hr.sq.ft)",q1_A1);
q3_A3=sigma*[(T3^4-T1^4)*F31+(T3^4-T2^4)*F32]; // using equation (3)
printf("\nThe heat flux through area A3 is %d BTU/(hr.sq.ft)",q3_A3);
printf("\nThe results are logical in that the heat entering the system (the oven itself) must equal that which leaves under steady-state conditions.");
```

Scilab code Exa 12.7 Radiation heat transfer within an enclosure of diffuse gray surfaces

```
clc;
clear;
printf("\t\t\tChapter12_example7\n\n\n");
// Determination of the heat lost by the oven through its top surface.
// all energy leaving A1 is intercepted by A2 and
```

```
    vice versa
F12=1;
F21=1;
F11=0; // the surfaces are flat
F22=0;
emissivity1=0.94; // for oxidized steel from
    appendix table E1
emissivity2=0.94
T1=533;
T2=323;
sigma=5.67e-8; // Stefan-Boltzmann constant
q1=(sigma*(T1^4-T2^4))/((1/emissivity1)+(1/
    emissivity2)-1);
printf("\nThe heat lost through bottom surface is %d
    W/sq.m",q1);
q2=-q1;
printf("\nThe heat lost through top surface is %d W/
    sq.m",q2);
```

Scilab code Exa 12.8 Radiation heat transfer within an enclosure of black surfaces and diffuse gray surfaces

```
clc;
clear;
printf("\t\t\tChapter12_example8\n\n\n");
// Determination of the net heat exchanged between
    the dish and the surroundings by radiation at the
     instant the dish is removed from the oven.
    Perform the calculations (a) if the dish and
    surroundings behave like black bodies, and again
    (b) if the dish has an emissivity of 0.82 and the
     surroundings have an emissivity of 0.93.
D=12/12; // diameter in ft
```

```
L=6/12; // length in ft
A=2*%pi*D^2/4+%pi*D*L;
printf("\nThe Surface area is %.2f sq.ft",A);
printf("\n\t\t\tSolution to part (a)\n");
F12=1; // the view factor between the dish and the
    surroundings is unity
T1=810;
T2=530;
sigma=0.1714e-8; // Stefan-Boltzmann constant
q1=sigma*A*(T1^4-T2^4)*F12;
printf("\nThe heat exchanged between the dish and
    the surroundings is %d BTU/hr",q1);
printf("\n\t\t\tSolution to part (b)\n");
// For gray-surface behavior, we can apply the
    following Equation
// q1/(A1e1)-[F11*(q1/A1)*(1-e1)/e1+F12*(q2/A2)*(1-
    e2)/e2]=sigma*T1^4-(F11*sigma*T1^4+F12*sigma*T2
    ^4)... equation (1)
F11=0;
e1=0.82;
e2=0.93;
// putting q2/A2=0 in equation (1) as A2 tends to
    infinity
q1_=A*e1*[sigma*T1^4-F12*sigma*T2^4];
printf("\nThe heat exchanged between the dish and
    the surroundings for the second case is %d BTU/hr
    ",q1_);
```