Scilab Textbook Companion for Heat Transfer (In SI Units) by J. P. Holman[1]

Created by
Shivam Singh
B.Tech
Chemical Engineering
IIT BHU
College Teacher
Prakash Kotecha
Cross-Checked by

May 20, 2016

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

Book Description

Title: Heat Transfer (In SI Units)

Author: J. P. Holman

Publisher: Tata McGraw - Hill Education, New Delhi

Edition: 9

Year: 2002

ISBN: 0-07-063451-3

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

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

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

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

Contents

List of Scilab Codes

Chapter 1

Introduction

Scilab code Exa 1.1 conduction through copper plate

```
clear;
clc;
printf("\t\t\tExample Number 1.1\n\n\n");
// conduction through copper plate
// illustration1.1
// solution

k = 370; // [W/m] at 250 degree celsius
dt = 100-400;//[degree celsius] temperature
   difference
dx = 3*10^(-2);//[m] thickness of plate
//calculating heat transfer per unit area from
   fourier's law
q = -k*dt/dx;//[MW/square meter]
printf("rate of heat transfer per unit area is %f MW
   /square meter",q/1000000);
```

Scilab code Exa 1.2 convection calculation

```
clear;
clc;
printf("\t\t\tExample Number 1.2\n\n\n");
// convection calculation
// illustration1.2
// solution

Twall = 250;//[degree celsius] wall temperature
Tair = 20;//[degree celsius] air temperature
h = 25;//[W/square meter] heat transfer coefficient
l = 75*10^(-2);//[m] length of plate
b = 50*10^(-2);//[m] width of plate
area = l*b;//[square meter] area of plate
dt = 250-20;//[degree celsius]
// from newton's law of cooling
q = h*area*dt;// [W]
printf("rate of heat transfer is %f kW",q/1000);
```

Scilab code Exa 1.3 multimode heat transfer

```
clear;
clc;
printf("\t\t\tExample Number 1.3\n\n\n");
// multimode heat transfer
// illustration1.3
// solution

Qconv = 2156;// [W] from previous problem
Qrad = 300;// [W] given
dx = 0.02;// [m] plate thicknesss
l = 0.75;// [m] length of plate
w = 0.5;// [m] width of plate
k = 43;//[W/m] from table 1.1
area = l*w;//[square meter] area of plate
Qcond = Qconv+Qrad;// [W]
```

```
dt = Qcond*dx/(k*area);// [degree celsius]
   temperature difference
Ti = 250+dt;// inside temperature
printf("the inside plate temperature is therefore %f
    degree celsius",Ti);
```

Scilab code Exa 1.4 heat source and convection

```
clear;
clc;
printf("\t\t\tExample Number 1.4\n\n\n");
// heat source and convection
// illustration1.4
// solution

d = 1*10^(-3);//[m] diameter of wire
l = 10*10^(-2);//[m] length of wire
Sarea = 22*d*l/7;//[square meter] surface area of
   wire
h = 5000;//[W/square meter] heat transfer
   coefficient
Twall = 114;// [degree celsius]
Twater = 100;// [degree celsius]
//total convection loss is given by equation(1-8)
Q = h*Sarea*(Twall-Twater);// [W]
printf("heat transfer is therefore %f W",Q);
printf(" this is equal to the electric power which
   must be applied");
```

Scilab code Exa 1.5 radiation heat transfer

```
clear;
clc;
```

```
printf("\t\t\tExample Number 1.5\n\n\n");
// radiation heat transfer
// illustration1.5
// solution

sigma = 5.699*10^(-8);//[W/square meter*k^(4)] universal constant
T1 = 273.15+800;// [k] first plate temperature
T2 = 273.15+300;// [k] second plate temperature
//equation(1-10) may be employed for this problem
Q = sigma*((T1^(4))-(T2^(4)));// [W/square meter]
printf("heat transfer per unit area is %f kW/square meter",Q/1000);
```

Scilab code Exa 1.6 total heat loss by convection and radiation

```
clear;
clc;
printf("\t\t\tExample Number 1.6\n\n\n");
// total heat loss by convection and radiation
// illustration1.6
// solution

d = 0.05;//[m] diameter of pipe
Twall = 50;//[degree celsius]
Tair = 20;//[degree celsius]
emi = 0.8;//emissivity
h = 6.5;//[W/square meter] heat transfer coefficient for free convection
Q1 = h*22*d*(Twall-Tair)/7;//[W/m] convection loss per unit length
sigma = 5.669*10^(-8);// [W/square meter*k^(4)] universal constant
T1 = 273.15+Twall;// [k]
T2 = 273.15+Tair;// [k]
```

```
Q2 = emi*22*d*sigma*((T1^(4))-(T2^(4)))/7;// [W/m]
    heat loss due to radiation per unit length
Qtotal = Q1+Q2;// [W/m] total heat loss per unit
    length
printf("total heat loss is therefore %f W/m",Qtotal)
    ;
```

Chapter 2

Steady State Conduction One Dimension

Scilab code Exa 2.1 multilayer conduction

```
clear;
clc;
printf("\t\t\tExample Number 2.1\n\n\n");
// multilayer conduction
// illustration2.1
// solution

dx1 = 0.1;// [m] thickness of layer of common brick
k1 = 0.7;// [W/m degree celsius] heat transfer
    coefficient of common brick
dx2 = 0.0375;// [m] thickness of layer of gypsum
    plaster
k2 = 0.48;// [W/m degree celsius] heat transfer
    coefficient gypsum plaster
Rb = dx1/k1;// [square meter degree celsius /W]
    thermal resistance of brick
Rp = dx2/k2;// [square meter degree celsius /W]
    thermal resistance of gypsum plaster
R = Rb+Rp;// [square meter degree celsius /W]
```

```
    thermal resistance without insulation
R1 = R/0.2;// [square meter degree celsius /W] with
    insulation
// heat loss with the rock-wool insulation is 20
    percent
Rrw = R1-R;// [square meter degree celsius /W]
k3 = 0.065;// [W/m degree celsius] heat transfer
    coefficient
dx3 = Rrw*k3;// [m]
printf("length of thickness is %f cm added to reduce
     the heat loss(or gain) through wall by 80
    percent",dx3*100);
```

Scilab code Exa 2.2 multilayer cylindrical system

```
clear;
clc;
printf("\t\t\tExample Number 2.2\n\n\n");
// multilayer cylindrical system
// illustration2.2
// solution

ID = 0.02;// [m] inner diameter of steel
OD = 0.04;//[m] outer diameter of steel
t = 0.03;//[m] thickness of asbestos insulation
// system is like three concentric cylinders
T1 = 600;// [degree celsius] inside wall temperature
T2 = 100;// [degree celsius] outside insulation
    temperature
Ks = 19;//[W/m degree celsius] heat transfer
    coefficient of steel
Ka = 0.2;// [W/m degree celsius] heat transfer
    coefficient of asbestos
// heat flow is given by per unit length
Q_l = ((2*22*(T1-T2)/7)/((log(OD/ID)/Ks)+(log(0.1/OD
```

```
    )/Ka)));// [W/m]
// above calculated heat flow is used to calculate
    the interface temperature
// between the outside wall and the insulation
Ta = Q_l*(log(0.1/OD)/(2*3.14*Ka))+T2;// [degree
    celsius] Ta is interface temperature
printf("heat flow is given by %f W/m",Q_l);
printf("\n the interface temperature is %f degree
    celsius ",Ta);
```

Scilab code Exa 2.3 heat transfer through a composite wall

```
clear;
clc;
printf("\t\t\tExample Number 2.3\n\n\n");
// heat transfer through a composite wall
// illustration2.3
// solution

// 1. heat transfer through studs for unit depth
l = 0.0413;// [m] length of wood studs
b = 1.0;// [m] unit depth
A = l*b;// [square meter] area of studs for unit
    depth
hi = 7.5;// [W/square meter per degree celsius]
    convectional heat transfer coefficient
ho = 15;// [W/square meter per degree celsius]
    convectional heat transfer coefficient
Kb = 0.69;// [W/m per degree celsius] heat transfer
    coefficient of brick
Kgi = 0.96;// [W/m per degree celsius] heat transfer
     coefficient of gypsum inner sheath
Ki = 0.04;// [W/m per degree celsius] heat transfer
    coefficient of insulation
Kws = 0.1;// [W/m per degree celsius] heat transfer
```

```
coefficient of wood stud
Kgo = 0.48;// [W/m per degree celsius] heat transfer coefficient of gypsum outer sheath
Rair = 1/(ho*A);// [degree celsius /W] convection resistance outside of brick
dx_b = 0.08;// [m] thickness of brick
dx_os = 0.019;//[m] thickness of outer sheet
dx_ws = 0.0921;// [m] thickness of wood stud
dx_is = 0.019;// [m] thickness of inner sheet
Rb = dx_b/(Kb*A);// [degree celsius /W] conduction resistance in brick
Ros = dx_os/(Kgi*A);// [degree celsius /W] conduction resistance through outer sheet
Rws = dx_ws/(Kws*A);// [degree celsius /W] conduction resistance through wood stud
Ris = dx_is/(Kgo*A);// [degree celsius /W] conduction resistance through inner sheet
Ri = 1/(hi*A);// [degree celsius /W] convection resistance on inside
Rt = Rair+Rb+Ros+Rws+Ris+Ri;// [degree celsius /W] total thermal resistance through the wood stud section
printf("total thermal resistance through the wood stud section is %f degree celsius /W",Rt);
// 2. heat transfer through insulation section
A1 = 0.406-A;// [square meter] area of insulation section for unit depth
dx_ins = 0.0921;// [m] thickness of insulation
Rins = dx_ins/(Ki*A1);// [degree celsius /W] conduction resistance through insulation section
// five of the materials are same but resistance involve different area
// i.e. (40.6-4.13) cm instead of 4.13 cm
// so that each of the previous must be multiplied by a factor of (4.13/(40.6-4.13)) = 0.113
Rt_ins = (Rair+Rb+Ros+Ris+Ri)*0.113+Rins;// [degree celsius /W] total resistance through insulation section
```

```
printf("\n total thermal resistance through the
    insulation section is %f degree celsius /W",
    Rt_ins);
R_overall = 1/((1/Rt)+(1/Rt_ins));// [degree celsius
     /W] overall resistance for the section
// the value is related to overall heat transfer
    coefficient by
// Q = U*A*dt = dt/R_overall
// where A is area of total section
A_ = 0.406;// [square meter] area of total section
U = 1/(R_overall*A_);// [W/square meter degree
    celsius] overall heat transfer coefficient
// R value is somewhat different from thermal
    resistance and is given by
R_value = 1/U;// [degree celsius square meter/W] R
    value of system
printf("\n overall heat transfer coefficient is %f W
    /square meter per degree celsius",U);
printf("\n R value is %f square meter/W",R_value);
```

Scilab code Exa 2.4 overall heat transfer coefficient for a tube

```
clear;
clc;
printf("\t\t\tExample Number 2.4\n\n\n");
// overall heat transfer coefficient for a tube
// illustration2.3
// solution

ID = 0.025;// [m] inner diameter of steel
OD = ID+2*0.0008;//[m] outer diameter of steel
hi = 3500;// [W/square meter per degree celsius]
    convectional heat transfer coefficient of inside
ho = 7.6;// [W/square meter per degree celsius]
    convectional heat transfer coefficient of outside
```

```
L = 1.0;// [m] tube length
Ai = %pi*ID*L;// [square meter] inside crossectional area
Ao = %pi*OD*L;// [square meter] outside crossectional area
k = 16;// [W/square meter per degree celsius] thermal conductivity of tube
Ri = 1/(hi*Ai);// [degree celsius /W] convection resistance inside tube
Rt = log(OD/ID)/(2*3.14*k*L);// [degree celsius /W] thermal resistance
Ro = 1/(ho*Ao);// [degree celsius /W] convection resistance outside tube
R_total = Ri+Rt+Ro;// [degree celsius /W] total thermal and convection  resistance
Uo = 1/(Ao*R_total);// [W/square meter degree celsius] overall heat transfer coefficient
printf("overall heat transfer coefficient is %f W/square meter degree celsius",Uo);
Tw = 50;// [degree celsius] water temperature
Ta = 20;// [degree celsius] surrounding air temperature
dt = Tw-Ta;// [degree celsius] temperature difference
q = Uo*Ao*dt;// [W] heat transfer
printf("\n heat loss per unit length is %f W(for 1m length)",q);
```

Scilab code Exa 2.5 critical insulation thickness

```
clear;
clc;
printf("\t\t\tExample Number 2.5\n\n\n");
// critical insulation thickness
// illustration2.5
```

```
// solution

k = 0.17;// [W/m per degree celsius] heat transfer coefficient of asbestos
Tr = 20;// [degree celsius] temperature of room air
h = 3;// [W/square meter per degree celsius] convectional heat transfer coefficient
Tp = 200;// [degree celsius] temperature of pipe
d = 0.05;// [m] diameter of pipe
// from equation (2-18) we calculate r_o as
r_o = k/h;// [m] critical radius of insulation
printf("critical radius of insulation for asbestos is %f cm ",r_o*100);
Ri = d/2;// [m] inside radius of insulation
// heat transfer is calculated from equation (2-17)
q_by_L = (2*3.14*(Tp-Tr))/(((log(r_o/Ri))/0.17)+(1/(h*r_o)));// [W/m] heat transfer per unit length
printf("\n heat loss when covered with critical radius of insulation is %f W/m",q_by_L);
// without insulation the convection from the outer surface of pipe is
q_by_L1 = h*2*3.14*Ri*(Tp-Tr);// [W/m] convection from outer surface without insulation
printf("\n heat loss without  insulation is %f W/m",q_by_L1);
per_inc = ((q_by_L-q_by_L1)/q_by_L1)*100;// percentage increase in heat transfer
printf("\n so the addition of 3.17 of insulation actually increases the heat transfer by %f percent",per_inc);
```

Scilab code Exa 2.6 heat source with convection

```
clear;
clc;
```

```
printf("\t\t\tExample Number 2.6\n\n\n");
// heat source with convection
// illustration2.6
// solution

// all the power generated in the wire must be
    dissipated by convection to the liquid
// P = i^(2)*R = q = h*A*dt
L = 100;// [cm] length of the wire
k = 19;// [W/m per degree celsius] heat transfer
    coefficient of steel wire
A = %pi*(0.15)^(2);// [square meter] crossectional
    area of wire
rho = 70*10^(-6);// [micro ohm cm] resistivity of
    steel
R = rho*L/A;// [ohm] resistance of wire
i = 200;// [ampere] current in the wire
P = i^(2)*R;// [W] power generated in the wire
Tl = 110;// [degree celsius] liquid temperature
d = 0.003;// [m] diameter of wire
l = 1;// [m] length of wire
Tw = (P/(4000*3.14*d*l))+110;// [degree celsius]
    wire temperature
// heat generated per unit volume q_dot is
    calculated as
// P = q_dot*V = q_dot*3.14*r^(2)*l
r = d/2;// [m] radius of wire
q_dot = P/(%pi*r^(2)*l);// [W/m^(3)]
// finally the center temperature of the wire is
    calculated from equation (2-26)
To = ((q_dot*(r^(2)))/(4*k))+Tw;// [degree celsius]
printf("center temperature of the wire is %f degree
    celsius",To);
```

Scilab code Exa 2.7 influence of thermal conductivity on fin temperature profiles

```
clear;
clc;
printf("\t\t\tExample Number 2.7\n\n\n");
// influence of thermal conductivity on fin temperature profiles
// illustration2.7
// solution

d = 0.02;// [m] diameter of rod
L = 0.1;// [m] length of rod
A = %pi*d^(2)/4;// [square meter] crossectional area of rod
h = 25;// [W/square meter per degree celsius] convectional heat transfer coefficient
k_c = 385;// [W/m per degree celsius] heat transfer coefficient of copper
k_s = 17;// [W/m per degree celsius] heat transfer coefficient of steel
k_g = 0.8;// [W/m per degree celsius] heat transfer coefficient of glass
// calculating (h*P/(k*A)) and m and m*L for three different rod
P = %pi*d;// [m] perimeter of rod
printf("Material\t(hP/kA)\t\tm\t\tmL");
printf("\ncopper\t\t%f\t%f\t\t%f",(h*P/(k_c*A)),((h*P/(k_c*A)))^(1/2),((h*P/(k_c*A)))^(1/2)*L);
printf("\nstainless steel\t%f\t%f\t%f",(h*P/(k_s*A)),((h*P/(k_s*A)))^(1/2),((h*P/(k_s*A)))^(1/2)*L);
printf("\nglass\t\t%f\t%f\t%f",(h*P/(k_g*A)),((h*P/(k_g*A)))^(1/2),((h*P/(k_g*A)))^(1/2)*L);
//
Lc = L+d/4;// [m] corrected length
// the parameters of interest for the heat flow and efficiency comparisons are now tabulated as
printf("\nthe parameters of interest for the heat
```

```
flow and efficiency comparisons are now tabulated as");
printf("\nMaterial\t\t(hPkA)\t\tmLc");
printf("\ncopper\t\t%f\t\t%f",(h*P*k_c*A),((h*P/(k_c*A)))^(1/2)*Lc);
printf("\nstainless steel\t%f\t\t%f",(h*P*k_s*A),((h*P/(k_s*A)))^(1/2)*Lc);
printf("\nglass\t\t%f\t\t%f",(h*P*k_g*A),((h*P/(k_g*A)))^(1/2)*Lc);
// efficiency is calculated using equation(2-38) by using the above values of mLc
// to compare the heat flows we could either calculate the values from equation (2-36) for a unit value of theta_o
printf("\nMaterial\t\tefficiency\tq relative to copper percentage");
printf("\ncopper\t\t%f\t\t%f",tanh(((h*P/(k_c*A)))^(1/2)*Lc)/(((h*P/(k_c*A)))^(1/2)*Lc),100);
printf("\nstainless steel\t%f\t\t%f",tanh(((h*P/(k_s*A)))^(1/2)*Lc)/(((h*P/(k_s*A)))^(1/2)*Lc),((tanh(((h*P/(k_s*A)))^(1/2)*Lc)/(((h*P/(k_s*A)))^(1/2)*Lc))/(tanh(((h*P/(k_c*A)))^(1/2)*Lc)/(((h*P/(k_c*A)))^(1/2)*Lc)))*100);
printf("\nglass\t\t%f\t\t%f",tanh(((h*P/(k_g*A)))^(1/2)*Lc)/(((h*P/(k_g*A)))^(1/2)*Lc),((tanh(((h*P/(k_g*A)))^(1/2)*Lc)/(((h*P/(k_g*A)))^(1/2)*Lc))/(tanh(((h*P/(k_c*A)))^(1/2)*Lc)/(((h*P/(k_c*A)))^(1/2)*Lc)))*100);
deff('[y]=f1(x)','y=exp(-((h*P/(k_c*A)))^(1/2)*x)/(1+exp(-2*((h*P/(k_c*A)))^(1/2)*L))+exp(((h*P/(k_c*A)))^(1/2)*x)/(1+exp(2*((h*P/(k_c*A)))^(1/2)*L))');
deff('[y]=f2(x)','y=exp(-((h*P/(k_s*A)))^(1/2)*x)/(1+exp(-2*((h*P/(k_s*A)))^(1/2)*L))+exp(((h*P/(k_s*A)))^(1/2)*x)/(1+exp(2*((h*P/(k_s*A)))^(1/2)*L))');
deff('[y]=f3(x)','y=exp(-((h*P/(k_g*A)))^(1/2)*x)/(1+exp(-2*((h*P/(k_g*A)))^(1/2)*L))+exp(((h*P/(
```

```
    k_g*A)))^(1/2)*x)/(1+exp(2*((h*P/(k_g*A)))^(1/2)*
    L))');
x=0:0.01:0.1;
plot(x,f1,x,f2,x,f3);
legend("copper, k = 385 W/m degree celsius","
    stainless steel k = 17 W/m degree celsius","glass
     k = 0.8 W/m degree celsius");
xlabel("x,m");
ylabel("theta / theta_O");
xgrid();
```

Scilab code Exa 2.8 straight aluminium fin

```
clear;
clc;
printf("\t\t\tExample Number 2.8\n\n\n");
// straight aluminium fin
// illustration2.8
// solution

t = 0.003;// [m] thickness of fin
L = 0.075;// [m] length of fin
Tb = 300;// [degree celsius] base temperature
Tair = 50;// [degree celsius] ambient temperature
k = 200;// [W/m per degree celsius] heat transfer
    coefficient of aluminium fin
h = 10;// [W/square meter per degree celsius]
    convectional heat transfer coefficient
// We Will use the approximate method of solution by
     extending the fin
// With a fictitious length t/2
// using equation(2-36)
Lc = L+t/2;// [m] corrected length
z = 1;// [m] unit depth
p = (2*z+2*t);// [m] perimeter of fin
```

```
A = z*t;// [square meter] crossectional area of fin
m = ((h*p)/(k*A))^(0.5);
// from equation(2-36)
dt = Tb-Tair;// [degree celsius] temperature
    difference
q = tanh(m*Lc)*((h*p*k*A)^(0.5))*dt;// [W/m] heat
    transfer per unit length
printf("heat loss from the fin per unit length is %f
    W/m",q);
```

Scilab code Exa 2.9 circumferential aluminium fin

```
clear;
clc;
printf("\t\t\tExample Number 2.9\n\n\n");
// circumferential aluminium fin
// illustration2.9
// solution

t = 0.001;// [m] thickness of fin
L = 0.015;// [m] length of fin
Ts = 170;// [degree celsius] surface temperature
Tfluid = 25;// [degree celsius] fluid temperature
k = 200;// [W/m per degree celsius] heat transfer
    coefficient of aluminium fin
h = 130;// [W/square meter per degree celsius]
    convectional heat transfer coefficient
d = 0.025;// [m] tube diameter
Lc = L+t/2;// [m] corrected length
r1 = d/2;// [m] radius of tube
r2_c = r1+Lc;// [m] corrected radius
Am = t*(r2_c-r1);// [square meter] profile area
c = r2_c/r1;// constant to determine efficiency of
    fin from curve
c1 = ((Lc)^(1.5))*((h/(k*Am))^(0.5));// constant to
```

```
determine efficiency of fin from curve
// using c and c1 to determine the efficiency of the fin from figure (2-12)
// we get nf = 82 percent
// heat would be transferred if the entire fin were at the base temperature
// both sides of fin exchanging heat
q_max = 2*%pi*(r2_c^(2)-r1^(2))*h*(Ts-Tfluid);// [W] maximum heat transfer
q_act = 0.82*q_max;//[W] actual heat transfer
printf("the actual heat transferred is %f W",q_act);
```

Scilab code Exa 2.10 rod with heat sources

```
clear;
clc;
printf("\t\t\tExample Number 2.10\n\n\n");
// rod with heat sources
// illustration2.10
// solution

// q_dot is uniform heat source per unit volume
// h is convection coefficient
// k is heat transfer coefficient
// A is area of crossection
// P is perimeter
// Tinf is environment temperature
// we first make an energy balance on the element of the rod shown in figure(2-10)
// energy in left place + heat generated in element = energy out right face + energy lost by convection
// or
// -(k*A*dT_by_dx)+(q_dot*A*dx) = -(k*A(dT_by_dx+(d2T_by_dx2)*dx))+h*P*dx*(T-Tinf)
```

```
// simlifying we have
// d2T_by_dx2-((h*P)/(k*A))*(T-Tinf)+q_dot/k = 0
// replacing theta = (T-Tinf) and (square meter) = ((h*P)/(k*A))
// d2theta_by_dx2-(square meter)*theta+q_dot/k = 0
// we can make a further substitution as theta` = theta-(q_dot/(k*(square meter)))
// so that our differential equation becomes
// d2theta`_by_dx2-(square meter)*theta`
// which has the general solution theta` = C1*exp^(-m*x)+C2*exp^(m*x)
// the two end temperatures are used to establish the boundary conditions:
// theta` = theta1` = T1-Tinf-q_dot/(k*(square meter)) = C1+C2
// theta` = theta2` = T2-Tinf-q_dot/(k*(square meter)) = C1*exp^(-m*L)+C2*exp^(m*L)
// solving for the constants C1 and C2 gives
// (((theta1`*exp^(2*m*L)-theta2`*exp^(m*L))*exp^(-m*x))+((theta2`exp^(m*L)-theta1`)exp^(m*x))/(exp^(2*m*L)-1))
printf("the expression for the temperature distribution in the rod is ");
printf("\n theta` = (((theta1`*exp^(2*m*L)-theta2`*exp^(m*L))*exp^(-m*x))+((theta2`exp^(m*L)-theta1`)exp^(m*x))/(exp^(2*m*L)-1))");
printf("\n for an infinitely long heat generating fin with the left end maintained at T1, the temperature distribution becomes ");
printf("\n theta`/theta1 = exp^(-m*x)");
```

Scilab code Exa 2.11 influence of contact conductance on heat transfer

```
clear;
clc;
```

```
printf("\t\t\tExample Number 2.11\n\n\n");
// influence of contact conductance on heat transfer
// illustration2.11
// solution

d = 0.03;//[m] diameter of steel bar
l = 0.1;//[m] length of steel bar
A = (%pi*d^(2))/4;// [square meter] crossectional area of bar
k = 16.3;// [W/square meter per degree celsius] thermal conductivity of tube
hc = 1893.93;// [W/square meter per degree celsius] contact coefficient
// the overall heat flow is subjected to three thermal resistances
// one conduction resistance for each bar
// contact resistance
Rth = l/(k*A);// [degree celsius /W]
// from table(2-2) the contact resistance is
Rc = 1/(hc*A);// [degree celsius /W]
Rt = 2*Rth+Rc;// [degree celsius /W] total resistance
dt = 100;// [degree celsius] temperature difference
q = dt/Rt;// [W] overall heat flow
printf("overall heat flow is %f W",q);
// temperature drop across the contact is found by taking the ratio
// of the contact resistance to the total thermal resistance
dt_c = (Rc/(2*Rth))*dt;// [degree celsius]
printf("\nthe temperature drop across the contact is %f degree celsius",dt_c);
```

Chapter 3

Steady State Conduction Multiple Dimension

Scilab code Exa 3.1 buried pipe

```
clear;
clc;
printf("\t\t\tExample Number 3.1\n\n\n");
// buried pipe
// illustration3.1
// solution

d = 0.15;// [m] diameter of pipe
r = d/2;// [m] radius of pipe
L = 4;// [m] length of pipe
Tp = 75;// [degree celsius] pipe wall temperature
Tes = 5;// [degree celsius] earth surface temperature
k = 0.8;// [W/m per degree celsius] thermal conductivity of earth
D = 0.20;// [m] depth of pipe inside earth
// We may calculate the shape factor for this situation using equation given in table 3-1
// since D<3*r
```

```
S = (2*%pi*L)/acosh(D/r);// [m] shape factor
// the heat flow is calculated from
q = k*S*(Tp-Tes);// [W]
printf("heat lost by the pipe is %f W",q);
```

Scilab code Exa 3.2 cubical furnace

```
clear;
clc;
printf("\t\t\tExample Number 3.2\n\n\n");
// cubical furnace
// illustration3.2
// solution

a = 0.5;// [m] length of side of cubical furnace
Ti = 500;// [degree celsius] inside furnace
    temperature
To = 50;// [degree celsius] outside temperature
k = 1.04;// [W/m per degree celsius] thermal
    conductivity of fireclay brick
t = 0.10;// [m] wall thickness
A = a*a;// [square meter] area of one face
// we compute the total shape factor by adding the
    shape factors for the walls, edges and corners
Sw = A/t;// [m] shape factor for wall
Se = 0.54*a;// [m] shape factor for edges
Sc = 0.15*t;// [m] shape factor for corners
// there are six wall sections, twelve edges and
    eight corners, so the total shape factor S is
S = 6*Sw+12*Se+8*Sc;// [m]
// the heat flow is calculated as
q = k*S*(Ti-To);// [W]
printf("heat lost through the walls is %f kW",q
    /1000);
```

Scilab code Exa 3.3 buried disk

```
clear;
clc;
printf("\t\t\tExample Number 3.3\n\n\n");
// buried disk
// illustration3.3
// solution

d = 0.30;// [m] diameter of disk
r = d/2;// [m] radius of disk
Td = 95;// [degree celsius] disk temperature
Ts = 20;// [degree celsius] isothermal surface
    temperature
k = 2.1;// [W/m per degree celsius] thermal
    conductivity of medium
D = 1.0;// [m] depth of disk in a semi-infinite
    medium
// We have to calculate shape factor using relation
    given in table (3-1)
// We select the relation for the shape factor is
    for the case D/(2*r)>1
S = (4*%pi*r)/((%pi/2)-atan(r/(2*D)));// [m] shape
    factor
// heat lost by the disk is
q = k*S*(Td-Ts);// [W]
printf("heat lost by disk is %f W",q);
```

Scilab code Exa 3.4 buried parallel disk

```
clear;
clc;
```

```
printf("\t\t\tExample Number 3.4\n\n\n");
// buried parallel disk
// illustration3.4
// solution

d = 0.50;// [m] diameter of both disk
r = d/2;// [m] radius of disk
Td1 = 80;// [degree celsius] first disk temperature
Td2 = 20;// [degree celsius] second disk temperature
k = 2.3;// [W/m per degree celsius] thermal conductivity of medium
D = 1.5;// [m] seperation of disk in a infinite medium
// We have to calculate shape factor using relation given in table (3-1)
// We select the relation for the shape factor is for the case D>5*r
S = (4*%pi*r)/((%pi/2)-atan(r/D));// [m] shape factor
q = k*S*(Td1-Td2);// [W]
printf("heat transfer between the disks is %f W",q);
```

Scilab code Exa 3.5 Nine node problem

```
clear;
clc;
printf("\t\t\tExample Number 3.5\n\n\n");
// Example 3.5 (page no.-86-88)
// Nine-node problem
// solution

h = 10;// [W/square meter per degree celsius] convectional heat transfer coefficient
k = 10;// [W/m per degree celsius] heat transfer coefficient
```

```
dx = 1/3;// [m] length of small squares in x
    direction
dy = 1/3;// [m] length of small squares in y
    direction
y = h*dx/(k);// to use in equation (3-25) and (3-26)
// the nodal equation for nodes is following
// T2+T4-4*T1 = -600 FOR NODE 1
// T3+T1+T5-4*T2 = -500 FOR NODE 2
// 2*T2+T6-4.67*T3 = -567 FOR NODE 3
// T5+T7+T1-4*T4 = -100 FOR NODE 4
// T6+T4+T8+T2-4*T5 = 0 FOR NODE 5
// 2*T5+T3+T9-4.67*T6 = -67 FOR NODE 6
// 2*T4+T8-4.67*T7 = -167 FOR NODE 7
// 2*T5+T7+T9-4.67*T8 = -67 FOR NODE 8
// T6+T8-2.67*T9 = -67 FOR NODE 9
A = [-4 1 0 1 0 0 0 0 0;1 -4 1 0 1 0 0 0 0;0 2 -4.67
     0 0 1 0 0 0;1 0 0 -4 1 0 1 0 0;0 1 0 1 -4 1 0 1
    0;0 0 1 0 2 -4.67 0 0 1;0 0 0 2 0 0 -4.67 1 0;0 0
     0 0 2 0 1 -4.67 1;0 0 0 0 0 1 0 1 -2.67];
C = [-600;-500;-567;-100;0;-67;-167;-67;-67];
T = A^(-1)*C;// [degree celsius]
printf("The nodal temperature of node 1 to 9 is
    shown below respectively");
disp(T);
// the heat flows at the boundaries are computed in
    two ways:
// as conduction flows for the 100 and 500 degree
    celsius faces and
// as convection flows for the other two faces
// for the 500 degree face the heat flow into the
    face is  q = sigma(k*dx*dT/dy)
// where dt is temperature difference and dy is
    length of small squares in y direction
q = k*dx*[500-T(1)+500-T(2)+(500-T(3))/2]/dy;// [W/m
    ]
// the heat flow out of the 100 degree face is  q =
    sigma(k*dy*dT/dx)
q1 = k*dy*[T(1)-100+T(4)-100+(T(7)-100)/2]/dx;// [W/
```

```
   m]
// the convection heat flow out the right face is
   given by the convection relation q = sigma(h*dy*(
   T-Tinf))
q2 = h*dy*[T(3)-100+T(6)-100+(T(9)-100)/2];// [W/m]
// the convection heat flow out the bottom face is
   given by the convection relation q = sigma(h*dx*(
   T-Tinf))
q3 = h*dx*[(100-100)/2+T(7)-100+T(8)-100+(T(9)-100)
   /2];// [W/m]
// total heat flow out is
qt = q1+q2+q3;
printf(" heat conducted into the top face is %f W/m"
   ,q);
printf("\n total heat flow out is %f W/m",qt);
printf("\n this compares that heat flow into the
   system is equal to the heat flow out of the
   system ");
```

Scilab code Exa 3.6 Gauss seidal calculation

```
clear;
clc;
printf("\t\t\tExample Number 3.6\n\n\n");
// Gauss-Seidal calculation
// Example 3.6 (page no.-97-98)
// solution

// it is useful to think in terms of a resistance
   formulation for this problem because all the
   connecting resistances between the nodes in
   figure 3-6(page no.-83) are equal; that is
// R = dy/(k*dy) = dx/(k*dy) = 1/k
                                (a)
// therefore, when we apply equation(3-32) to each
```

```
    node, we obtain(qi = 0)
// Ti = (sum Kj*Tj)/(sum Kj)
                                    (b)
// because each node has four resistances connected
    to it and k is assumed constant,
// sum Kj = 4*k
// and
// Ti = (1/4)*(sum Tj)
                                        (c)
// we are now making four nadal equations for
    iteration
// node 1 : T1 = (1/4)*(100+500+T2+T3)
// node 2 : T2 = (1/4)*(500+100+T1+T4)
// node 3 : T3 = (1/4)*(100+100+T1+T4)
// node 3 : T4 = (1/4)*(T3+T2+100+100)
// we now set up an iteration table as shown in
    output
A=[4 -1 -1 0;-1 4 0 -1;-1 0 4 -1;0 -1 -1 4];
b=[600;600;200;200];
x=[300;300;200;200];
NumIters=6;
D=diag(A);
A=A-diag(D);
for i=1:4
    D(i)=1/D(i);
end
n=length(x);
x=x(:);
y=zeros(n,NumIters);
for j=1:NumIters
    for k=1:n
        x(k)=(b(k)-A(k,:)*x)*D(k);
    end
    y(:,j)=x;
end
printf("the iteration table is shown as : \n\n");
disp(y);
printf("\n\n after five iterations the solution
```

```
converges and the final temperatures are \n");
disp(y(1,6),"T1=");
disp(y(2,6),"T2=");
disp(y(3,6),"T3=");
disp(y(4,6),"T4=");
```

Scilab code Exa 3.7 numerical formulation with heat generation

```
clear;
clc;
printf("\t\t\tExample Number 3.7\n\n\n");
// numerical formulation with heat generation
// Example 3.7 (page no.-99-100)
// solution

d = 4;// [mm] diameter of wire
Q = 500;// [MW/cubic meter] heat generation
Tos = 200;// [degree celsius] outside surface
    temperature of wire
k = 19;// [W/m degree celsius] thermal conductivity
// we shall make the calculations per unit length
dz = 1;
// because the system is one-dimensional, we take
dphai = 2*%pi;
dr = 0.5;// [mm]
// a summary of values for different nodes are
    following

// node 1.

rm1 = 0.25;// [mm]
Rmplus1 = (dr/2)/((rm1+dr/4)*dphai*dz*k);// [degree
    celsius/W]
// Rmminus1 = infinity
dV1 = rm1*dr*dphai*dz;// [cubic micro meter]
```

```
q1 = Q*dV1;// [W]

// node 2.

rm2 = 0.75;// [mm]
Rmplus2 = (dr/2)/((rm2+dr/4)*dphai*dz*k);// [degree
    celsius/W]
// Rmminus2 = infinity
dV2 = rm2*dr*dphai*dz;// [cubic micro meter]
q2 = Q*dV2;// [W]

// node 3.

rm3 = 1.25;// [mm]
Rmplus3 = (dr/2)/((rm3+dr/4)*dphai*dz*k);// [degree
    celsius/W]
// Rmminus3 = infinity
dV3 = rm3*dr*dphai*dz;// [cubic micro meter]
q3 = Q*dV3;// [W]

// node 4.

rm4 = 1.75;// [mm]
Rmplus4 = (dr/2)/((rm4+dr/4)*dphai*dz*k);// [degree
    celsius/W]
// Rmminus1 = infinity
dV4 = rm4*dr*dphai*dz;// [cubic micro meter]
q4 = Q*dV4;// [W]

// a summary of values of sum_one_by_Rij and Ti
    according to equation (3-32) is now given to be
    used in gauss seidal iteration scheme

// node 1

sum_one_by_Rij1 = (1/Rmplus1);// [degree celsius/W]
// the equations formed after putting values are
// T1 = 3.288+T2
```

```
// node 2

sum_one_by_Rij2 = (1/Rmplus2);// [degree celsius/W]
// the equations formed after putting values are
// T2 = 3.289+(1/3)*T1+(2/3)*T3

// node 3

sum_one_by_Rij3 = (1/Rmplus3);// [degree celsius/W]
// the equations formed after putting values are
// T3 = 3.290+ 0.4*T2+06*T4

// node 4

sum_one_by_Rij4 = (1/Rmplus4);// [degree celsius/W]
// the equations formed after putting values are
// T4 = 2.193+(2/7)*T3+142.857

// now we will solve these equations by iteration
A=[1 -1 0 0;-(1/3) 1 -(2/3) 0;0 -0.4 1 -0.6;0 0
    -(2/7) 1];
b=[3.288;3.289;3.290;142.857+2.193];
x=[240;230;220;210];
NumIters=13;
D=diag(A);
A=A-diag(D);
n=length(x);
x=x(:);
y=zeros(n,NumIters);
for j=1:NumIters
    for z=1:n
        x(z)=(b(z)-A(z,:)*x)*D(z);
    end
    y(:,j)=x;
end
printf("thirteen iterations are now tabulated :\n");
disp(y);
```

```
// the total heat loss from the wire may be
   calculated as the conduction through Rmplus at
   node 4. then
T4 = y(4,13);// [degree celsius]
q = (T4-Tos)/(Rmplus4);// [W/m]
// this must equal the heat generated in the wire,
   or
V = %pi*(d*10^(-3)/2)^(2);// [square meter]
q_exact = Q*10^(6)*V;// [W/m]
printf("\n\n the total heat loss from the wire by
   the conduction through Rmplus at node 4 is %f kW/
   m",q/1000);
printf("\n\n heat generated in the wire is %f kW/m",
   q_exact/1000);
printf("\n\n the difference between the two values
   results from the inaccuracy in determination of
   T4");
```

Scilab code Exa 3.8 heat generation with non uniform nodal elements

```
clear;
clc;
printf("\t\t\tExample Number 3.8\n\n\n");
// heat generation with non uniform nodal elements
// Example 3.8 (page no.-100-103)
// solution

k = 0.8;// [W/m degree celsius] thermal conductivity
    of glass
d = 0.003;// [m] thickness of layer of glass
x = 0.001;// [m] thickness of electric conducting
   strip
Tinf = 30;// [degree celsius] environment
   temperature
h = 100;// [W/square meter degree celsius]
```

```
q1 = 40;// [W] heat generated by strips
q2 = 20;// [W] heat generated by strips
// the nodal network for a typical section of the glass is shown in figure. In this example we have not chosen dx = dy.
// because of symmetry T1 = T7 ,T2 = T6, etc, and we only need to solve the temperatures of 16 nodes. we employ the resistance formulation. As shown, we have chosen
dx = 0.005;// [m]
dy = 0.001;// [m]
A = 0.005;// [square meter]
// various resistances may now be calculated:

// for nodes 1,2,3,4:
one_by_Rm_p1 = k*dy/(2*dx);
one_by_Rm_m1 = one_by_Rm_p1;
one_by_Rn_p1 = h*A;
one_by_Rn_m1 = k*dx/dy;

// for nodes 8,9,10,11,15,16,17,18:
one_by_Rm_p2 = k*dy/(dx);
one_by_Rm_m2 = one_by_Rm_p2;
one_by_Rn_m2 = k*dx/dy;
one_by_Rn_p2 = one_by_Rn_m2;

// for nodes 22,23,24,25:
one_by_Rm_p3 = k*dy/(2*dx);
one_by_Rm_m3 = one_by_Rm_p3;
one_by_Rn_p3 = k*dx/dy;
one_by_Rn_m3 = 0;// [insulated surface]

// from the above resistances we may calculate the sum_one_by_Rij as
// nodes : 1,2,3,4:
sum_one_by_Rij1 = 4.66;
// nodes : 8,9,10,11,15,16,17,18:
sum_one_by_Rij2 = 8.32;
```

```
// nodes : 22,23,24,25:
sum_one_by_Rij3 = 4.16;
// the nodal equations are obtained from equation (3-31)
// only node 4 has a heat generation term, Qi = 0 for all other nodes.
// the equations are listed below
// for node 1 : T8*one_by_Rn_m1+T2*one_by_Rm_p1+30*one_by_Rn_m1-sum_one_by_Rij1*T1 = 0;
// for node 4 : T5*one_by_Rm_p1+T3*one_by_Rm_m1+30*one_by_Rn_p1+T11*one_by_Rn_m1+Q-sum_one_by_Rij1*T4 = 0
// similar equations are obtained and we solve it by matrix method
Z = [-4.58 0.08 0 0 4 0 0 0 0 0 0 0 0 0 0 0;
     0.08 -4.66 0.08 0 0 4 0 0 0 0 0 0 0 0 0 0;
     0 0.08 -4.66 0.08 0 0 4 0 0 0 0 0 0 0 0 0;
     0 0 0.16 -4.66 0 0 0 4 0 0 0 0 0 0 0 0;
     4 0 0 0 -8.16 0.16 0 0 4 0 0 0 0 0 0 0;
     0 4 0 0 0.16 -8.32 0.16 0 0 4 0 0 0 0 0 0;
     0 0 4 0 0 0.16 -8.32 0.16 0 0 4 0 0 0 0 0;
     0 0 0 4 0 0 0.32 -8.32 0 0 0 4 0 0 0 0;
     0 0 0 0 4 0 0 0 -8.16 0.16 0 0 4 0 0 0;
     0 0 0 0 0 4 0 0 0.16 -8.32 0.16 0 0 4 0 0;
     0 0 0 0 0 0 4 0 0 0.16 -8.32 0.16 0 0 4 0;
     0 0 0 0 0 0 0 4 0 0 0.32 -8.32 0 0 0 4;
     0 0 0 0 0 0 0 0 4 0 0 0 -4.08 0.08 0 0;
     0 0 0 0 0 0 0 0 0 4 0 0 0.08 -4.16 0.08 0;
     0 0 0 0 0 0 0 0 0 0 4 0 0 0.08 -4.16 0.08;
     0 0 0 0 0 0 0 0 0 0 0 4 0 0 0.16 -4.16];
C = [-15;-15;-15;-15-q2;0;0;0;0;0;0;0;0;0;0;0;0];
T1 = Z^(-1)*C;
printf("Nodes (1,2,3,4,8,9,10,11,15,16,17,18,22,23,24,25) temperature at 20 W/m respectively");
disp(T1);
Z1 = [-4.58 0.08 0 0 4 0 0 0 0 0 0 0 0 0 0 0;
     0.08 -4.66 0.08 0 0 4 0 0 0 0 0 0 0 0 0 0;
```

```
      0 0.08 -4.66 0.08 0 0 4 0 0 0 0 0 0 0 0 0;
      0 0 0.16 -4.66 0 0 0 4 0 0 0 0 0 0 0 0;
      4 0 0 0 -8.16 0.16 0 0 4 0 0 0 0 0 0 0;
      0 4 0 0 0.16 -8.32 0.16 0 0 4 0 0 0 0 0 0;
      0 0 4 0 0 0.16 -8.32 0.16 0 0 4 0 0 0 0 0;
      0 0 0 4 0 0 0.32 -8.32 0 0 0 4 0 0 0 0;
      0 0 0 0 4 0 0 0 -8.16 0.16 0 0 4 0 0 0;
      0 0 0 0 0 4 0 0 0.16 -8.32 0.16 0 0 4 0 0;
      0 0 0 0 0 0 4 0 0 0.16 -8.32 0.16 0 0 4 0;
      0 0 0 0 0 0 0 4 0 0 0.32 -8.32 0 0 0 4;
      0 0 0 0 0 0 0 0 4 0 0 0 -4.08 0.08 0 0;
      0 0 0 0 0 0 0 0 0 4 0 0 0.08 -4.16 0.08 0;
      0 0 0 0 0 0 0 0 0 0 4 0 0 0.08 -4.16 0.08;
      0 0 0 0 0 0 0 0 0 0 0 4 0 0 0.16 -4.16];
C1 = [-15;-15;-15;-15-q1;0;0;0;0;0;0;0;0;0;0;0;0];
T2 = Z1^(-1)*C1;
printf("\n\n Nodes
   (1,2,3,4,8,9,10,11,15,16,17,18,22,23,24,25)
   temperature at 40 W/m respectively");
disp(T2);
// we know the numerical value that the convection
   should have
// the convection losss at the top surface is given
   by
qc1 = 2*h*[(dx/2)*(T1(1)-Tinf)+dx*(T1(2)+T1(3)-2*
   Tinf)+(dx/2)*(T1(4)-Tinf)];// [W] for 20W/m, the
   factor of 2 accounts for both sides of section
qc2 = 2*h*[(dx/2)*(T2(1)-Tinf)+dx*(T2(2)+T2(3)-2*
   Tinf)+(dx/2)*(T2(4)-Tinf)];// [W] for 40W/m
printf("\n\n the convection loss at the top surface
   is given by (for 20 W/m heat generation) %f W",
   qc1);
printf("\n\n the convection loss at the top surface
   is given by (for 40 W/m heat generation) %f W",
   qc2);
```

Scilab code Exa 3.11 use of variable mesh size

```
clear;
clc;
printf("\t\t\tExample Number 3.11\n\n\n");
// use of variable mesh size
// Example 3.11 (page no.-108-110)
// solution

// using data given in figure example 3-11(page no
   .-109)
// nodes 5,6,8, and 9 are internal nodes with dx =
   dy and have nodal equations in the form of
   equation(3-24). Thus,
// 600+T6+T8-4*T5 = 0
// 500+T5+T7+T9-4*T6 = 0
// 100+T5+T9+T11-4*T8 = 0
// T8+T6+T10+T12-4*T9 = 0
// For node 7 we can use a resistance formulation
   and obtain
// (1/R_7_6) = k
// (1/R_7_500_degree) = k*(dx/6+dx/2)/(dy/3) = 2*k
// (1/R_7_10) = 2*k
// and we find
// 1000+T6+2*T10-5*T7 = 0
// similar resistance are obtained for node 10
// (1/R_10_9) = k
// (1/R_10_7) = 2*k = (1/R_10_1)
// so that
// 2*T7+T9+2*T1-5*T10 = 0
// for node 1,
// (1/R_1_12) = k*(dy/6+dy/2)/(dx/3) = 2*k
// (1/R_1_3) = k*(dx/6+dx/2)/(dy) = 2*k/3
// (1/R_1_10) = 2*k
```

```
// and the nodal equation becomes
// 3*T12+3*T10+T3-7*T1 = 0
// for node 11,
// (1/R_11_100_degree) = (1/R_11_12) = k*(dy/6+dy/2)/(dx/3) = 2*k
// (1/R_11_8) = k
// (1/R_11_13) = k*(dx/3)/dy = k/3
// and the nodal equation becomes
// 600+6*T12+3*T8+T13-16*T11 = 0
// Similarly, the equation for node 12 is
// 3*T9+6*T11+6*T1+T14-16*T12 = 0
// for node 13,
// (1/R_13_100_degree) = k*(dy)/(dx/3) = 3*k = 1/R_13_14
// (1/R_13_11) = (1/R_13_100) = k/3
// and we obtain
// 1000+9*T14+T11-20*T13 = 0
// similarly for node 14,
// 100+9*T13+9*T3+T12-20*T14 = 0
// finally, from resistances already found, the nodal equation for node 3 is
// 200+9*T14+2*T1-13*T3 = 0
// we choose to solve the set of equations by the gauss-seidel iteration technique
A=[1 -1 0 0 0 0 0 0 0 0 0 0 0 0;0 0 1 -1 0 0 0 0 0 0 0 0 0 0;0 0 0 0 -4 1 0 1 0 0 0 0 0 0;0 0 0 0 1 -4 1 0 1 0 0 0 0 0;0 0 0 0 1 0 0 -4 1 0 1 0 0 0;0 0 0 0 0 1 0 1 -4 1 0 1 0 0;0 0 0 0 0 1 -5 0 0 2 0 0 0 0;2 0 0 0 0 0 2 0 1 -5 0 0 0 0;-7 0 1 0 0 0 0 0 0 3 0 3 0 0;0 0 0 0 0 0 0 3 0 0 -16 6 1 0;6 0 0 0 0 0 0 0 3 0 6 -16 0 1;0 0 0 0 0 0 0 0 0 0 1 0 -20 9;0 0 9 0 0 0 0 0 0 0 0 1 9 -20;2 0 -13 0 0 0 0 0 0 0 0 0 0 9];
b
=[0;0;-600;-500;-100;0;-1000;0;0;-600;0;-1000;-100;-200];

T = A^(-1)*b;
printf("Nodal temperatures for node
```

```
(1,2,3,4,5,6,7,8,9,10,11,12,13,14) are respectively as follows in degree celsius");
disp(T);
```

Scilab code Exa 3.12 Three dimensional numerical formulation

```
clear;
clc;
printf("\t\t\tExample Number 3.12\n\n\n");
// Three-dimensional numerical formulation
// Example 3.12 (page no.-110-113)
// solution

Tinf = 10;// [degree celsius] environment temperature
h = 500;// [W/square meter degree celsius]
Ts = 100;// [degree celsius] four side temperature
k = 2;// [W/m degree celsius]
dx = 0.01;// [m]
dy = 0.01;// [m]
dz = 0.01;// [m]
// all of the interior nodes for Z-planes 2,3,4 have resistances of
A = dy*dz;// [square meter]
one_by_R = k*A/dx;
one_by_R_11_21 = one_by_R;
one_by_R_21_22 = one_by_R;
// the surface conduction resistances for surface Z-plane are
one_by_R_11_12 = k*A/dx;
one_by_R_11_14 = one_by_R_11_12;
// the surface convection resistances are
one_by_R_11_inf = h*A;
// for surfaces nodes like 11 the sum_one_by_R_ij term in equation (3-32) becomes
```

```
sum_one_by_R_11_j = 4*one_by_R_11_12+one_by_R+one_by_R_11_inf;
// while, for interior nodes, we have
sum_one_by_R_21_j = 6*one_by_R;
// for the insulated black surface nodes
sum_one_by_R_51_j = 4*one_by_R_11_12+one_by_R;
// there are 30 nodes in total; 6 in each z-plane. we could write the equations for all of them but prefer to take advantage of the symmetry of the problem as indicated in figure. thus,
// T11 = T13 = T14 = T16 And T12 = T15, etc
// we may then write the surface nodal equations as
// T11 = [0.05*Tinf+0.02*T21+(0.01)*(100+100+T14+T12)]/0.11
// T12 = [0.05*Tinf+0.02*T22+(0.01)*(100+T11+T15+T13)]/0.11
// inserting
Tinf = 10;// [degree celsius]
// following the same procedure for the other z-planes we obtain
// T21 = (200+T11+T31+T22)/5
// T22 = (100+T12+T32+2*T21)/5
// T31 = (200+T21+T41+T32)/5
// T32 = (100+T22+T42+2*T31)/5
// T41 = (200+T31+T51+T42)/5
// T42 = (100+T32+T52+2*T41)/5
// T51 = (2+0.02*T41+0.01*T52)/0.05
// T52 = (1+0.02*T42+0.02*T51)/0.05
// Solving the 10 equations
Z = [-0.1 0.01 0.02 0 0 0 0 0 0 0;
     0.02 -0.1 0 0.02 0 0 0 0 0 0;
     1 0 -5 1 1 0 0 0 0 0;
     0 1 2 -5 0 1 0 0 0 0;
     0 0 1 0 -5 1 1 0 0 0;
     0 0 0 1 2 -5 0 1 0 0;
     0 0 0 0 1 0 -5 1 1 0;
     0 0 0 0 0 1 2 -5 0 1;
     0 0 0 0 0 0 0.02 0 -0.05 0.01;
```

```
     0 0 0 0 0 0 0 0.02 0.02 -0.05];
C = [-2.5;-1.5;-200;-100;-200;-100;-200;-100;-2;-1];
T = Z^(-1)*C;
T11 = T(1);
T12 = T(2);
T21 = T(3);
T22 = T(4);
T31 = T(5);
T32 = T(6);
T41 = T(7);
T42 = T(8);
T51 = T(9);
T52 = T(10);
printf("the following results for the temperature in each z-plane is ;");
printf("\n\t\t z-plane\t\tNode 1\t\t\tNode 2");
printf("\n\t\t%f\t\t%f\t\t%f",1,T11,T12);
printf("\n\t\t%f\t\t%f\t\t%f",2,T21,T22);
printf("\n\t\t%f\t\t%f\t\t%f",3,T31,T32);
printf("\n\t\t%f\t\t%f\t\t%f",4,T41,T42);
printf("\n\t\t%f\t\t%f\t\t%f",5,T51,T52);
val = [1 2 3 4 5];
val1 = [T11 T21 T31 T41 T51];
val2 = [T12 T22 T32 T42 T52];
plot(val,val1,val,val2);
legend("T11","T22");
xgrid();
xlabel("z-plane");
ylabel("Temperature (degree celsius)");
```

Chapter 4

Unsteady State Conduction

Scilab code Exa 4.1 steel ball cooling in air

```
clear;
clc;
printf("\t\t\tExample Number 4.1\n\n\n");
// steel ball cooling in air
// illustration4.1
// solution

h = 10;// [W/square meter per degree celsius]
    convectional heat transfer coefficient
k = 35;// [W/m per degree celsius] heat transfer
    coefficient
c = 460;// [kJ/kg]
r = 0.05/2;// [m] diameter of ball
Tb = 450;// [degree celsius] ball temperature
Te = 100;// [degree celsius] environment temperature
A = 4*%pi*r^(2);
V = 4*%pi*r^(3)/3;
// We anticipate that the lumped capacity method
    will apply because of the low value of h and high
     value of k
// we check by using equation (4-6)
```

```
K = h*(V/A)/k;
// since the value of K is less than 0.1 so we will
   use equation (4-5)
T = 150;// [degree celsius] attained temperature by
   the ball
rho = 7800;// [kg/cubic meter] density of the ball
a = (h*A)/(rho*c*V);
t = log((T-Te)/(Tb-Te))/(-a);// [s] time required to
    attain the temperature of 150 degree celsius
printf("time required to attain the temperature of
   150 by degree celsius by the ball is %f h",t
   /3600);
```

Scilab code Exa 4.2 Semi infinite solid with sudden change in surface conditions

```
clear;
clc;
printf("\t\t\tExample Number 4.2\n\n\n");
// Semi-infinite solid with sudden change in surface
    conditions
// illustration4.2
// solution

k = 45;// [W/m per degree celsius] thermal
   conductivity of steel block
alpha = 1.4*10^(-5);// [square meter/s] constant
Tb = 35;// [degree celsius] block temperature
x = 0.025;// [m] depth at which temperature is
   calculated
t = 30;// [s] time after which temperature is to be
   calculated
// we can make use of the solutions for the semi-
   infinite solid given as equation (4-8) and (4-13a
   )
```

```
// for case A (by suddenly raising the  surface temperature to 250 degree celsius)
To = 250;// [degree celsius]
T_x_t = To+(Tb-To)*(erf(x/(2*(alpha*t)^(1/2))));
printf("temperature at depth of 0.025 m after 30 second for case 1 is %f degree celsius",T_x_t);
// for the constant heat flux case B we make use of equation (4-13a)
// since qo/A is given
q_by_A = 3.2*10^(5);// [W/square meter]
T_x_t1 = Tb+(2*q_by_A*(alpha*t/%pi)^(1/2)*exp(-x^(2)/(4*alpha*t))/k)-(q_by_A*x*(1-erf(x/(2*(alpha*t)^(1/2))))/k);// [degree celsius]
printf("\n temperature at depth of 0.025 m after 30 second for case 2 is %f degree celsius",T_x_t1)
// for the constant heat flux case the surface temperature after 30 s would be evaluated with x = 0 in equation(4-13a)
x = 0;// [m] at the surface
T_x_o = Tb+(2*q_by_A*(alpha*t/%pi)^(1/2)*exp(-x^(2)/(4*alpha*t))/k)-(q_by_A*x*(1-erf(x/(2*(alpha*t)^(1/2))))/k);// [degree celsius]
printf("\n surface temperature after 30 second is %f degree celsius",T_x_o);
```

Scilab code Exa 4.3 pulsed energy at surface of semi infinite solid

```
clear;
clc;
printf("\t\t\tExample Number 4.3\n\n\n");
// pulsed energy at surface of semi-infinite solid
// illustration4.3
// solution

rho = 7800;// [kg/cubic meter] density of slab
```

```
c = 460;// [J/kg degree celsius] constant
alpha = 0.44*10^(-5);// [square meter/s] constant
Ts = 40;// [degree celsius] initial temperature of of slab
x = 0.0;// [m] depth at which temperature is calculated
t = 2;// [s] time after which temperature is calculated
// this problem is a direct application of equation (4-13b)
// we have
Qo_by_A = 10^(7);// [J/square meter] heat flux
To = Ts+(Qo_by_A/(rho*c*(%pi*alpha*t)^(1/2)))*exp(-x^(2)/(4*alpha*t));// [degree celsius] surface temperature at x = 0
printf("surface temperature at x = 0 and at t = 2 second is %f degree celsius",To);
x = 0.002;// [m] depth at which temperature is calculated
T = Ts+(Qo_by_A/(rho*c*(%pi*alpha*t)^(1/2)))*exp(-x^(2)/(4*alpha*t));// [degree celsius] temperature at depth x = 0.002
printf("\n temperature at depth 0.002 m and after 2 second is %f degree celsius",T);
```

Scilab code Exa 4.4 heat removal from semi infinite solid

```
clear;
clc;
printf("\t\t\tExample Number 4.4\n\n\n");
// heat removal from semi-infinite solid
// illustration4.4
// solution

alpha = 8.4*10^(-5);// [square meter/s] constant
```

```
Ts = 200;// [degree celsius] initial temperature of
   of slab
x = 0.04;// [m] depth at which temperature is
   calculated
T_x_t = 120;// [degree celsius] temperature at depth
    0.04 m
To = 70;// [degree celsius] surface temperature
   after lowering
k = 215;// [W/m degree celsius] heat transfer
   coefficient
// We first find the time required to attain the 120
    degree celsius temperature
// and then integrate equation(4-12) to find the
   total heat removed during this interval
t = (x/(erfinv(((T_x_t-To)/(Ts-To)))*2*sqrt(alpha)))
   ^(2);// [s]
printf("time taken to attain the temperature of 120
   degree celsius %f s",t);
// the total heat removed at the surface is obtained
    by integrating equation(4-12):
// Qo_by_A = integrate('qo_by_A','dt',0,t)
// or Qo_by_A = integrate('k*(To-Ts)/(sqrt(%pi*alpha
   *t))','dt',0,t)
Qo_by_A = integrate('k*(To-Ts)/(sqrt(%pi*alpha*t))',
   't',0,t);
printf("\n the total heat removed from the surface
   is %e J/square meter",Qo_by_A);
```

Scilab code Exa 4.5 sudden exposure of semi infinite solid slab to convection

```
clear;
clc;
printf("\t\t\tExample Number 4.5\n\n\n");
// sudden exposure of semi-infinite solid slab to
```

```
convection
// illustration4.5
// solution

alpha = 8.4*10^(-5);// [square meter/s] constant
Ts = 200;// [degree celsius] initial temperature of of slab
Te = 70;// [degree celsius] environment temperature
k = 215;// [W/m degree celsius] heat transfer coefficient of slab
h = 525;// [W/square meter degree celsius] heat transfer coefficient
x = 0.04;// [m] depth at which temperature is calculated
T_x_t = 120;// [degree celsius] temperature at depth 0.04 m
// we can use equation (4-15) or figure (4-5) for solution of this problem
// by using figure it is easier to calculate  it involves iterative method to solve because time appeares in both the variables
// h*sqrt(alpha*t)/k and x/(2*sqrt(alpha*t))
K = (T_x_t-Ts)/(Te-Ts);
// we seek the values of t such that the above value of K is equal to the value of K which comes out from graph
// we therfore try values of t and obtain other readings
printf("The iteration are listed below\n");
// at t = 1000s
t = 1000;// [s] time
A =  h*sqrt(alpha*t)/k;
B = x/(2*sqrt(alpha*t));
printf(" t\t\th*sqrt(alpha*t)/k \t x/(2*sqrt(alpha*t)) \t (T_x_t-Ts)/(Te-Ts)");
printf("\n %f\t\t %f \t %f \t\t 0.41",t,A,B);
t = 3000;// [s] time
A =  h*sqrt(alpha*t)/k;
```

```
B = x/(2*sqrt(alpha*t));
printf("\n %f\t\t %f \t %f \t\t 0.61",t,A,B);
t = 4000;// [s] time
A =  h*sqrt(alpha*t)/k;
B = x/(2*sqrt(alpha*t));
printf("\n %f\t\t %f \t %f \t\t 0.68",t,A,B);
printf("\n consequently the time required is
   approximately 3000 second");
```

Scilab code Exa 4.6 aluminium plate suddenly exposed to convection

```
clear;
clc;
printf("\t\t\tExample Number 4.6\n\n\n");
// aluminium plate suddenly  exposed to convection
// illustration4.6
// solution

alpha = 8.4*10^(-5);// [square meter/s] constant
Ts = 200;// [degree celsius] initial temperature of
   of plate
Te = 70;// [degree celsius] environment temperature
k = 215;// [W/m degree celsius] heat transfer
   coefficient of plate
h = 525;// [W/square meter degree celsius] heat
   transfer coefficient
x = 0.0125;// [m] depth at which temperature is
   calculated
t = 60;// [s] time after which plate temperature is
   calculated
L = 0.025;// [m] thickness of plate
theta_i = Ts-Te;// [degree celsius]
// then
Z = alpha*t/L^2;
X = k/(h*L);
```

```
x_by_L = x/L;
// from figure 4-7(page no.-144-145)
theta_o_by_theta_i = 0.61;
theta_o = theta_o_by_theta_i*theta_i;// [degree
    celsius]
// from figure 4-10(page no.-149) at x/L = 0.5,
theta_by_theta_o = 0.98;
theta = theta_by_theta_o*theta_o;// [degree celsius]
T = Te+theta;// [degree celsius]
// we compute the energy lost by the slab by using
    Figure 4-14(page no.-152). For this calculation
    we require the following properties of aluminium:
rho = 2700;// [kg/cubic meter]
C = 900;// [J/kg degree celsius]
// for figure 4-14(page no.-152) we need
V = h^2*alpha*t/(k^2);
B = h*L/k;
// from figure 4-14(page no.-152)
Q_by_Qo = 0.41;
// for unit area
Qo_by_A = rho*C*2*L*theta_i;// [J/square meter]
// so that the heat removed per unit surface area is
Q_by_A = Qo_by_A*Q_by_Qo;// [J/square meter]
printf("\n\n temperature at a depth of 1.25 cm from
    one of faces after 1 min of exposure of plate to
    the environment is %f degree celsius",T);
printf("\n\n energy removed per unit area from the
    plate in this time is %e J/square meter",Q_by_A);
```

Scilab code Exa 4.7 long cylinder suddenly exposed to convection

```
clear;
clc;
printf("\t\t\tExample Number 4.7(Page no.-154-155)\n
    \n\n");
```

```
// long cylinder suddenly exposed to convection
// Example 4.7
// solution

d = 0.05;// [m] diameter of cylinder
Ti = 200;// [degree celsius] initial temperature of
   aluminium cylinder
Tinf = 70;// [degree celsius] temperature of
   environment
h = 525;// [W/square meter degree celsius] heat
   transfer coefficient
// we have
theta_i = Ti-Tinf;// [degree celsius]
alpha = 8.4*10^(-5);// [square meter/s]
ro = d/2;// [m]
t = 60;// [s]
k = 215;// [W/m degree celsius]
r = 0.0125;// [m]
rho = 2700;// [kg/cubic meter]
C = 900;// [J/kg degree celsius]
// we compute
Z = alpha*t/ro^2;
X = k/(h*ro);
r_by_ro = r/ro;
// from figure 4-8(page no.-146)
theta_o_by_theta_i = 0.38;
// and from figure 4-11(page no.-150) at r/ro = 0.5
theta_by_theta_o = 0.98;
// so that
theta_by_theta_i = theta_o_by_theta_i*
   theta_by_theta_o;
theta = theta_i*theta_by_theta_i;// [degree celsius]
T = Tinf+theta;// [degree celsius]
// to compute the heat lost , we determine
V = h^2*alpha*t/k^2;
B = h*ro/k;
// then from figure 4-15(page no.-153)
Q_by_Qo = 0.65;
```

```
// for unit length
Qo_by_L = rho*C*%pi*ro^2*theta_i;// [J/m]
// and the actual heat lost per unit length is
Q_by_L = Qo_by_L*Q_by_Qo;// [J/m]
printf("temperature at a radius of 1.25 cm is %f
   degree celsius",T);
printf("\n\nheat lost per unit length 1 minute after
    the cylinder is exposed to the environment is %e
    J/m",Q_by_L);
```

Scilab code Exa 4.8 semi infinite cylinder suddenly exposed to convection

```
clear;
clc;
printf("\t\t\tExample Number 4.8\n\n\n");
// semi-infinite cylinder suddenly exposed to
   convection
// illustration4.8
// solution

d = 0.05;// [m] diameter of aluminium cylinder
Ti = 200;// [degree celsius] initial temperature of
   of cylinder
Te = 70;// [degree celsius] environment temperature
k = 215;// [W/m degree celsius] heat transfer
   coefficient of plate
h = 525;// [W/square meter degree celsius]
   convection heat transfer coefficient
alpha = 8.4*10^(-5);// [square meter/s] constant
x = 0.10;// [m] distance at which temperature is
   calculated from end
t = 60;// [s] time after which temperature is
   measured
// so that the parameters for use with figure(4-5)
A = h*sqrt(alpha*t)/k;
```

```
B = x/(2*sqrt(alpha*t));
// from figure (4-5)
z = 1-0.036;
S_of_X = z;
// for the infinite cylinder we seek both the axis-
    and surface-temperature ratios.
// the parameters for use with fig(4-8) are
r_o = d/2;// [m] radius of aluminium cylinder
r = d/2;// [m]  for surface temperature ratio
C = k/(h*r_o);
D = (alpha*t/r_o^(2));
y = 0.38;
// this is the axis temperature ratio.
// to find the surface-temperature ratio,we enter
    figure (4-11),using
R = r/r_o;
u = 0.97;
// thus
w = y;// at r = 0
v = y*u;// at r = r_o
C_of_O_axis = w;// at r = 0
C_of_O_r_o = v;// at r = r_o
// combining the solutions for the semi-infinite
    slab and infinite cylinder, we have
t = S_of_X*C_of_O_axis;
s = S_of_X*C_of_O_r_o;
// the corresponding temperatures are
T_axis = Te+t*(Ti-Te);
T_r_o = Te+s*(Ti-Te);
printf("the temperature at the axis is %f degree
    celsius",T_axis);
printf("\n the temperature at the surface is %f
    degree celsius",T_r_o);
```

Scilab code Exa 4.9 finite length cylinder suddenly exposed to convection

```
clear;
clc;
printf("\t\t\tExample Number 4.9\n\n\n");
// finite length cylinder suddenly exposed to
   convection
// illustration4.9
// solution

d = 0.05;// [m] diameter of aluminium cylinder
Ti = 200;// [degree celsius] initial temperature of
   of cylinder
Te = 70;// [degree celsius] environment temperature
k = 215;// [W/m degree celsius] heat transfer
   coefficient of plate
h = 525;// [W/square meter degree celsius]
   convection heat transfer coefficient
alpha = 8.4*10^(-5);// [square meter/s] constant
x1 = 0.00625;// [m] distance at which temperature is
    calculated from end
t = 60;// [s] time after which temperature is
   measured
r = 0.0125;// [m] radius at which temperature is
   calculated
// to solve this problem we combine the solutions
   from heisler charts for an infinite cylinder and
   an infinite plate in accordance with the
   combination shown in fig (4-18f)
// for the infinite plate problem
L = 0.05;// [m]
// the x position is measured fromthe center of the
   plate so that
x = L-x1;// [m]
A = k/(h*L);
B = (alpha*t/L^(2));
// from figures (4-17) and (4-10) respectively
thetha_o_by_i = 0.75;
thetha_by_i = 0.95;
// so that
```

```
thetha_by_i_plate = thetha_o_by_i*thetha_by_i;
// for the cylinder
r_o = d/2;// [m] radius of the cylinder
R = r/r_o;
C = k/(h*r_o);
D = (alpha*t/r_o^(2));
// and from figures (4-8) and (4-11), respectively
thetha_o_by_i_cyl = 0.38;
thetha_by_o = 0.98;
// so that
thetha_by_i_cyl = thetha_o_by_i_cyl*thetha_by_o;
// combibing the solutions for the plate and
   cylinder gives
thetha_by_i_short_cyl = thetha_by_i_plate*
   thetha_by_i_cyl;
// thus
T = Te+thetha_by_i_short_cyl*(Ti-Te);
printf("the temperature at a radial position of
   0.0125 m and a distance of 0.00625m from one end
   of cylinder 60 second after exposure to
   environment is %f degree celsius",T);
```

Scilab code Exa 4.10 heat loss for finite length cylinder

```
clear;
clc;
printf("\t\t\tExample Number 4.10\n\n\n");
// heat loss for finite-length cylinder
// illustration4.10
// solution

d = 0.05;// [m] diameter of aluminium cylinder
l = 0.1;// [m] length of aluminium cylinder
Ti = 200;// [degree celsius] initial temperature of
   of cylinder
```

```
Te = 70;// [degree celsius] environment temperature
k = 215;// [W/m degree celsius] heat transfer
    coefficient of plate
h = 525;// [W/square meter degree celsius]
    convection heat transfer coefficient
alpha = 8.4*10^(-5);// [square meter/s] constant
x1 = 0.00625;// [m] distance at which temperature is
     calculated from end
t = 60;// [s] time after which temperature is
    measured
r = 0.0125;// [m] radius at which temperature is
    calculated
// we first calculate the dimensionless heat-loss
    ratio for the infinite plate and infinite
    cylinder which make up the multidimensional body
// for the plate we have
L = 0.05;// [m]
A = h*L/k;
B = h^(2)*alpha*t/k^(2);
// from figure (4-14), for the plate, we read
Q_by_Q_o_plate = 0.22;
// for the cylinder
r_o = 0.025;// [m]
// so we calculate
C = h*r_o/k;
// and from figure(4-15) we have
Q_by_Q_o_cyl = 0.55;
// the two heat ratios may be inserted in equation
    (4-22) to give
Q_by_Q_o_tot = Q_by_Q_o_plate+Q_by_Q_o_cyl*(1-
    Q_by_Q_o_plate);
c = 896;// [J/kg degree celsius] specific heat of
    aluminium
rho = 2707;// [kg/cubic meter] density of aluminium
V = %pi*r_o^(2)*1;// [cubic meter]
Qo = rho*c*V*(Ti-Te);// [J]
Q = Qo*Q_by_Q_o_tot;// [J] the actual heat loss in
    the 1-minute
```

```
printf(" the actual heat loss in the 1-minute  is %f kJ",Q/1000);
```

Scilab code Exa 4.12 implicit formulation

```
clear;
clc;
printf("\t\t\tExample Number 4.12\n\n\n");
// implicit formulation
// Example 4.12 (page no.-173-174)
// solution

// we are using the data of example 4.11 for this question
// we are inserting the value of Rij in equation (4-43) to write the nodal equations for the end of the first time increment, taking all T1^(p) = 200 degree celsius
// we use underscore to designate the temperatures at the end of the time increment. for node 1
// 0.05302*T1_ = 200/70.731+T2_/70.731+40/84.833+0.01296*200
// for node 2
// 0.05302*T2_ = T1_/70.731+T3_/70.731+40/84.833+0.01296*200
// for node 3 and 4,
// 0.05302*T3_ = T2_/70.731+T4_/70.731+40/84.833+0.01296*200
// 0.02686*T4_ = T3_/70.731+40/2829+40/169.77+0.00648*200
// these equations can then be reduced to
// 0.05302*T1_-0.01414*T2_ = 5.8911
// -0.01414*T1_+0.05302*T2_-0.01414*T3_ = 3.0635
// -0.01414*T2_+0.05302*T3_-0.01414*T4_ = 3.0635
// -0.01414*T3_+0.02686*T4_ = 1.5457
```

```
// These equations can be solved by matrix method
Z = [0.05302 -0.01414 0 0;-0.01414 0.05302 -0.01414
    0;0 -0.01414 0.05302 -0.01414;0 0 -0.01414
    0.02686];
C = [5.8911;3.0635;3.0635;1.5457];
T_ = Z^(-1)*C;
T1_ = T_(1);// [degree celsius]
T2_ = T_(2);// [degree celsius]
T3_ = T_(3);// [degree celsius]
T4_ = T_(4);// [degree celsius]
// we can now apply the backward-difference
    formulation a second time using the double
    underscore to designate the temperatures at the
    end of the second time increment:
// 0.05302*T1__ = 200/70.731+T2__
    /70.731+40/84.833+0.01296*145.81
// 0.05302*T2__ = T1__/70.731+T3__
    /70.731+40/84.833+0.01296*130.12
// 0.05302*T3__ = T2__/70.731+T4__
    /70.731+40/84.833+0.01296*125.43
// 0.02686*T4__ = T3__
    /70.731+40/2829+40/169.77+0.00648*123.56
// These equations can be solved by matrix method
X = [0.05302 -0.01414 0 0;-0.01414 0.05302 -0.01414
    0;0 -0.01414 0.05302 -0.01414;0 0 -0.01414
    0.02686];
V = [5.1888;2.1578;2.0970;1.0504];
T__ = X^(-1)*V;
T1__ = T__(1);// [degree celsius]
T2__ = T__(2);// [degree celsius]
T3__ = T__(3);// [degree celsius]
T4__ = T__(4);// [degree celsius]
printf(" temperatures after time increment 1 are:");
printf("\n\n\t\t T1'' == %f",T1_);
printf("\n\n\t\t T2'' == %f",T2_);
printf("\n\n\t\t T3'' == %f",T3_);
printf("\n\n\t\t T4'' == %f",T4_);
printf("\n\n temperatures after time increment 2 are
```

```
    :");
printf("\n\n\t\t T1'''' == %f",T1__);
printf("\n\n\t\t T2'''' == %f",T2__);
printf("\n\n\t\t T3'''' == %f",T3__);
printf("\n\n\t\t T4'''' == %f",T4__);
```

Chapter 5

Principles of Convection

Scilab code Exa 5.1 water flow in a diffuser

```
clear;
clc;
printf("\t\t\tExample Number 5.1\n\n\n");
// water flow in a diffuser
// illustration5.1
// solution

Tw = 20;// [degree celcius] water temperature
m_dot = 8;// [kg/s] water flow rate
d1 = 0.03;// [m] diameter at section 1
d2 = 0.07;// [m] diameter at section 2
A1 = %pi*d1^(2)/4;// [square meter] cross-sectional area at section 1
A2 = %pi*d2^(2)/4;// [square meter] cross-sectional area at section 2
gc = 1;// [m/s^(2)] acceleration due to gravity
rho = 1000;// [kg/cubic meter] density of water at 20 degree celcius
// we may calculate the velocities from the mass-continuity relation
u1 = m_dot/(rho*A1);// [m/s]
```

```
u2 = m_dot/(rho*A2);// [m/s]
// the pressure difference is obtained by Bernoulli
   equation(5-7a)
p2_minus_p1 = rho*(u1^(2)-u2^(2))/(2*gc);// [Pa]
printf("the increase in static pressure between
   sections 1 and 2 is %f kPa",p2_minus_p1/1000);
```

Scilab code Exa 5.2 isentropic expansion of air

```
clear;
clc;
printf("\t\t\tExample Number 5.2\n\n\n");
// isentropic expansion of air
// illustration5.2
// solution

Ta = 300+273.15;// [K] air temperature
Pa = 0.7;// [MPa] pressure of air
u2 = 300;// [m/s] final velocity
gc = 1;// [m/s^(2)] acceleration due to gravity
Y = 1.4;// gama value for air
Cp = 1005;// [J/kg degree celsius]
// the initial velocity is small and the process is
   adiabatic. in terms of temperature
T2 = Ta-u2^(2)/(2*gc*Cp);
printf("the static temperature is %f K",T2);
// we may calculate the pressure difference from the
    isentropic relation
p2 = Pa*((T2/Ta)^(Y/(Y-1)));
printf("\n\n static pressure is %f MPa",p2);
// the velocity of sound at condition 2 is
a2 = 20.045*T2^(1/2);// [m/s]
// so that the mach no. is
M2 = u2/a2;
printf("\n\n Mach number is %f",M2);
```

Scilab code Exa 5.3 mass flow and boundary layer thickness

```
clear;
clc;
printf("\t\t\tExample Number 5.3\n\n\n");
// mass flow and boundary-layer thickness
// illustration5.3
// solution

Ta = 27+273.15;// [K] air temperature
Pa = 101325;// [Pa] pressure of air
u = 2;// [m/s] air velocity
x1 = 0.2;// [m] distance from the leading edge of
    plate
x2 = 0.4;// [m] distance from the leading edge of
    plate
R = 287;// []
mu = 1.85*10^(-5);// [kg/m s] viscosity of air
// the density of air is calculated from
rho = Pa/(R*Ta);// [kg/cubic meter]
// the reynolds number is calculated as
Re_x1 = rho*u*x1/mu;
Re_x2 = rho*u*x2/mu;
// the boundary layer thickness is calculated from
    equation(5-21)
del_x1 = 4.64*x1/Re_x1^(1/2);// [m]
del_x2 = 4.64*x2/Re_x2^(1/2);// [m]
// to calculate the mass flow which enters the
    boundary layer from the free stream between x =
    0.2 m and x = 0.4 m
// we simply take the difference between the mass
    flow in the boundary layer at these two x
    positions.
// at any x position the mass flow in the boundary
```

```
    layer is given by the integral dm = integrate('
    rho*u_y','y',0,del);
// velocity is given by equation (5-19) u_y = u
    *[1.5*(y/del)-0.5*(y/del)^(3)]
// after integration we get dm = 5*rho*u*del/8
dm = 5*rho*u*(del_x2-del_x1)/8
printf(" mass flow entering the boundary layer is %e
     kg/s",dm);
```

Scilab code Exa 5.4 isothermal flat plate heated over entire length

```
clear;
clc;
printf("\t\t\tExample Number 5.4\n\n\n");
// isothermal flat plate heated over entire length
// illustration5.4
// solution

// total heat transfer over a certain length of the
    plate is desired, so we wish to calculate average
     heat transfer coefficients.
// for this purpose we use equations (5-44) and
    (5-45), evaluating the properties at the film
    temperature :
Tp = 60+273.15;// [K] plate temperature
Ta = 27+273.15;// [K] air temperature
Tf = (Tp+Ta)/2;// [K]
u = 2;// [m/s] air velocity
// from appendix A the properties are
v = 17.36*10^(-6);// [square meter/s] kinematic
    viscosity
x1 = 0.2;// [m] distance from the leading edge of
    plate
x2 = 0.4;// [m] distance from the leading edge of
    plate
```

```
k = 0.02749;// [W/m K] heat transfer coefficient
Pr = 0.7;// prandtl number
Cp = 1006;// [J/kg K]
// at x = 0.2m
Re_x1 = u*x1/v;// reynolds number
Nu_x1 = 0.332*Re_x1^(1/2)*Pr^(1/3);// nusselt number
hx1 = Nu_x1*k/x1;// [W/square meter K]
// the average value of the heat transfer
   coefficient is twice this value, or
h_bar1 = 2*hx1;// [W/square meter K]
// the heat flow is
A1 = x1*1;// [square meter] area for unit depth
q1 = h_bar1*A1*(Tp-Ta);// [W]
// at x = 0.4m
Re_x2 = u*x2/v;// reynolds number
Nu_x2 = 0.332*Re_x2^(1/2)*Pr^(1/3);// nusselt number
hx2 = Nu_x2*k/x2;// [W/square meter K]
// the average value of the heat transfer
   coefficient is twice this value, or
h_bar2 = 2*hx2;// [W/square meter K]
// the heat flow is
A2 = x2*1;// [square meter] area for unit depth
q2 = h_bar2*A2*(Tp-Ta);// [W]
printf("the heat transfered in first case of the
   plate is %f W",q1);
printf("\n\n and the heat transfered in second case
   of the plate is %f W",q2);
```

Scilab code Exa 5.5 flat plate with constant heat flux

```
clear;
clc;
printf("\t\t\tExample Number 5.5\n\n\n");
// flat plate with constant heat flux
// illustration5.5
```

```
// solution

u = 5;// [m/s] air velocity
l = 0.6;// [m] plate length
Ta = 27+273.15;// [K] temperature of airstream
// properties should be evaluated at the film temperature, but we do not know the plate temperature so for an initial calculation we take the properties at the free-stream conditions of
v = 15.69*10^(-6);// [square meter/s] kinematic viscosity
k = 0.02624;// [W/m degree celsius] heat transfer coefficient
Pr = 0.7;// prandtl number
Re_l = l*u/v;// reynolds number
P = 1000;// [W] power of heater
qw = P/l^(2);// [W/square meter] heat flux per unit area
// from equation (5-50) the average temperature difference is
Tw_minus_Tinf_bar = qw*l/(0.6795*k*(Re_l)^(1/2)*(Pr)^(1/3));// [degree celsius]
// now, we go back and evaluate properties at
Tf = (Tw_minus_Tinf_bar+Ta+Ta)/2;// [degree celsius]
// and obtain
v1 = 28.22*10^(-6);// [square meter/s] kinematic viscosity
k1 = 0.035;// [W/m degree celsius] heat transfer coefficient
Pr1 = 0.687;// prandtl number
Re_l1 = l*u/v1;// reynolds number
Tw_minus_Tinf_bar1 = qw*l/(0.6795*k1*(Re_l1)^(1/2)*(Pr1)^(1/3));// [degree celsius]
// at the end of the plate(x = l = 0.6m) the temperature difference is obtained from equation (5-48) and (5-50) with the constant of 0.453
Tw_minus_Tinf_x_equal_l = Tw_minus_Tinf_bar1*0.6795/0.453;// [degree celsius]
```

```
printf("average temperature difference along the
   plate is %f degree celsius",Tw_minus_Tinf_bar);
printf("\n\n temperature difference at the trailing
   edge is %f degree celsius",
   Tw_minus_Tinf_x_equal_l);
```

Scilab code Exa 5.6 plate with unheated starting length

```
clear;
clc;
printf("\t\t\tExample Number 5.6\n\n\n");
// plate with unheated starting length
// illustration5.6
// solution

u = 20;// [m/s] air velocity
l = 0.2;// [m] plate length as well as width (square
   )
p = 101325;// [Pa] air pressure
Ta = 300;// [K] temperature of airstream
Tw = 350;// [K] temperature of last half of plate
// First we evaluate the air properties at the film
   temperature
Tf = (Tw+Ta)/2;// [K]
// and obtain
v = 18.23*10^(-6);// [square meter/s] kinematic
   viscosity
k = 0.02814;// [W/m degree celsius] heat transfer
   coefficient
Pr = 0.7;// prandtl number
// at the trailing edge of the plate the reynolds
   number is
Re_l = l*u/v;// reynolds number
// or laminar flow over the length of the plate
// heating does not start until the last half of the
```

```
    plate, or at position xo = 0.1m.
xo = 0.1;// [m]
// the local heat transfer coefficient is given by
   equation (5-41)
// hx = 0.332*k*Pr^(1/3)*(u/v)^(1/2)*x^(-1/2)*[1-(xo
   /x)^(0.75)]^(-1/3);
// the plate is 0.2 m wide so the heat transfer is
   obtained by integrating over the heated length xo
   <x<l
q = l*(Tw-Ta)*integrate('(0.332*k*Pr^(1/3)*(u/v)
   ^(1/2)*x^(-1/2)*[1-(xo/x)^(0.75)]^(-1/3))','x',xo
   ,l);
printf("the heat lost by the plate is %f W",q);
// the average value of the heat transfer
   coefficient over the heated length is given by
h = q*(Tw-Ta)*(l-xo)*l;// [W/square meter degree
   celsius]
printf("\n\n the average value of heat transfer
   coefficient over the heated length is given by %f
    W/square meter degree celsius",h);
```

Scilab code Exa 5.7 oil flow over heated flat plate

```
clear;
clc;
printf("\t\t\tExample Number 5.7\n\n\n");
// oil flow over heated flat plate
// illustration5.7
// solution

u = 1.2;// [m/s] oil velocity
l = 0.2;// [m] plate length as well as width (square
   )
To = 20+273.15;// [K] temperature of engine oil
Tu = 60+273.15;// [K] uniform temperature of plate
```

```
// First we evaluate the film temperature
T = (To+Tu)/2;// [K]
// and obtain the properties of engine oil are
rho = 876;// [kg/cubic meter] density of oil
v = 0.00024;// [square meter/s] kinematic viscosity
k = 0.144;// [W/m degree celsius] heat transfer coefficient
Pr = 2870;// prandtl number
// at the trailing edge of the plate the reynolds number is
Re = l*u/v;// reynolds number
// because the prandtl no. is so large we will employ equation(5-51) for the solution.
// we see that hx varies with x in the same fashion as in equation(5-44) , i.e. hx is inversely proportional to the square root of x ,
// so that we get the same solution as in equation (5-45) for the average heat transfer coefficient.
// evaluating equation(5-51) at x = 0.2m gives
Nux = 0.3387*Re^(1/2)*Pr^(1/3)/[1+(0.0468/Pr)^(2/3)]^(1/4);
hx = Nux*k/l;// [W/square meter degree celsius] heat transfer coefficient
// the average value of the convection coefficient is
h = 2*hx;// [W/square meter degree celsius]
// so that total heat transfer is
A = l^(2);// [square meter] area of the plate
q = h*A*(Tu-To);// [W]
printf("average value of the convection coefficient is %f W/square meter degree celsius",h);
printf("\n\n and the heat lost by the plate is %f W",q);
```

Scilab code Exa 5.8 drag force on a flat plate

```
clear;
clc;
printf("\t\t\tExample Number 5.8\n\n\n");
// drag force on a flat plate
// illustration5.8
// solution

// data is used from example 5.4
// we use equation (5-56) to compute the friction
    coefficient and then calculate the drag force .
// an average friction coefficient is desired , so
    st_bar*pr^(2/3) = Cf_bar/2
p = 101325;// [Pa] pressure of air
x = 0.4;// [m] drag force is computed on first 0.4 m
    of the plate
R = 287;// []
Tf = 316.5;// [K]
u = 2;// [m/s] air velocity
Cp = 1006;// [J/kg K]
Pr = 0.7;// prandtl no.
rho = p/(R*Tf);// [kg/cubic meter] density at 316.5
    K
h_bar = 8.698;// [W/square meter K] heat transfer
    coefficient
// for the 0.4m length
st_bar = h_bar/(rho*Cp*u);
// then from equation (5-56)
Cf_bar = st_bar*Pr^(2/3)*2;
// the average shear stress at the wall is computed
    from equation(5-52)
tau_w_bar = Cf_bar*rho*u^(2)/2;// [N/square meter]
A = x*1;// [square meter] area per unit length
// the drag force is the product of this shear
    stress and the area ,
D = tau_w_bar*A;// [N]
printf("Drag force exerted on the first 0.4 m of the
    plate is %f mN",D*1000);
```

Scilab code Exa 5.9 turbulent heat transfer from isothermal flat plate

```
clear;
clc;
printf("\t\t\tExample Number 5.9\n\n\n");
// turbulent heat transfer from isothermal flat plate
// illustration5.9
// solution

p = 101325;// [Pa] pressure of air
R = 287;// []
Ta = 20+273.15;// [K] temperature of air
u = 35;// [m/s] air velocity
L = 0.75;// [m] length of plate
Tp = 60+273.15;// [K] plate temperature
// we evaluate properties at the film temperature
Tf = (Ta+Tp)/2;// [K]
rho = p/(R*Tf);// [kg/cubic meter]
mu = 1.906*10^(-5);// [kg/m s] viscosity
k = 0.02723;// [W/m degree celsius]
Cp = 1007;// [J/kg K]
Pr = 0.7;// prandtl no.
 // the reynolds number is
 Rel = rho*u*L/mu;
 // and the boundary layer is turbulent because the reynolds number is greater than 5*10^(5).
 // therefore, we use equation(5-85) to calculate the average heat transfer over the plate:
 Nul_bar = Pr^(1/3)*(0.037*Rel^(0.8)-871);
 A = L*1;// [square meter] area of plate per unit depth
h_bar = Nul_bar*k/L; // [W/square meter degree celsius]
```

```
q = h_bar*A*(Tp-Ta);// [W] heat transfer from plate
 printf("heat transfer from plate is %f W",q);
```

Scilab code Exa 5.10 turbulent boundary layer thickness

```
clear;
clc;
printf("\t\t\tExample Number 5.10\n\n\n");
// turbulent-boundary-layer thickness
// illustration5.10
// solution

// we have to use the data from example 5.8 and 5.9
Rel = 1.553*10^6;// from previous example
L = 0.75;// [m] length of plate
// it is a simple matter to insert this value in
   equations(5-91) and (5-95) along with
x = L;// [m]
// turbulent-boundary-layer thickness are
// part a. from the leading edge of the plate
del_a = x*0.381*Rel^(-0.2);// [m]
// part b  from the transition point at Recrit =
   5*10^(5)
del_b = x*0.381*Rel^(-0.2)-10256*Rel^(-1);// [m]
printf("turbulent-boundary-layer thickness at the
   end of the plate from the leading edge of the
   plate is %f mm",del_a*1000);
printf("\n\n turbulent-boundary-layer thickness at
   the end of the plate from the transition point at
    Re_crit = 5*10^(5) is %f mm",del_b*1000);
```

Scilab code Exa 5.11 high speed heat transfer for a flat plate

```
clear;
clc;
printf("\t\t\tExample Number 5.11\n\n\n");
// high speed heat transfer for a flat plate
// Example 5.11 (page no.-257-259)
// solution

L = 0.7;// [m] length of flat plate
W = 1;// [m] width of plate
// flow conditions are
M = 3;
p = 101325/20;// [Pa]
T = -40+273;// [degree celsius]
Tw = 35;// [degree celsius] temperature at which
    plate is maintained
Gamma = 1.4;
g_c = 1;// []
R = 287;// [] universal gas costant
// we have to consider laminar and turbulent
    portions of the boundary layer seperately
// the free-stream acoustic velocity is calculated
    from
a = sqrt(Gamma*g_c*R*T);// [m/s]
// so that free stream velocity is
u = M*a;// [m/s]
// the maximum reynolds number is estimated by
    making a computation based on properties
    evaluated at free stream conditions:
rho = p/(R*T);// [Kg/m^(3)]
mu = 1.434*10^(-5);// [Kg/m s]
Re_L = rho*u*L/mu;
// thus we conclude that both laminar and turbulent
    boundary layer heat transfer must be considered.
// we first determine the reference temperature for
    the two regimes and then evaluate properties at
    these temperatures.

// LAMINAR PORTION
```

```
T_o = T*(1+((Gamma-1)/2)*M^(2));// [K]
Pr = 0.7// prandtl number(assuming)
// we have
r = sqrt(Pr);
T_aw = r*(T_o-T)+T;// [K]
// then the reference temperature from equation
   (5-124) is
T_star = T+0.5*(Tw-(T-273))+0.22*(T_aw-T);// [K]
// checking the prandtl number at this temperature
Pr_star = 0.697;
// so that the calculation is valid.because Pr_star
   and the value of Pr used to determine the
   recovery factor are almost same
// the other properties to be used in the laminar
   heat transfer analysis are
rho_star = p/(R*T_star);// [Kg/m^(3)]
mu_star = 2.07*10^(-5);// [Kg/m s]
k_star = 0.03;// [W/m degree celsius]
Cp_star = 1009;// [J/Kg degree celsius]

// TURBULENT PORTION

// Assuming
Pr = 0.7;
r = Pr^(1/3);
T_aw1 = r*(T_o-T)+T;// [K]
// then the reference temperature from equation
   (5-124) is
T_star = T+0.5*(Tw-(T-273))+0.22*(T_aw-T);// [K]
// checking the prandtl number at this temperature
Pr_star1 = 0.695;
// the agreement between Pr_star and the assumed
   value is sufficiently close.
// the other properties to be used in the turbulent
   heat transfer analysis are
rho_star1 = p/(R*T_star);// [Kg/m^(3)]
mu_star1 = 2.09*10^(-5);// [Kg/m s]
```

```
k_star1 = 0.0302;// [W/m degree celsius]
Cp_star1 = 1009;// [J/Kg degree celsius]

// LAMINAR HEAT TRANSFER

// we assume
Re_star_crit = 5*10^(5);
x_c = Re_star_crit*mu_star/(rho_star*u);// [m]
Nu_bar = 0.664*(Re_star_crit)^(1/2)*(Pr_star)^(1/3);
h_bar = Nu_bar*k_star/x_c;// [W/m^(2) degree celsius
    ] average heat transfer coefficient
// heat transfer is calculated as
A = x_c*1;// [m^(2)]
q = h_bar*A*(Tw-(T_aw-273));// [W]

// TURBULENT HEAT TRANSFER

// to determine the turbulent heat transfer we must
    obtain an expression for the local heat transfer
    coefficient from
// St_x*Pr_star1^(2/3) = 0.0296*Re_star_x^(-1/5)
// and then integrate from x = 0.222m to x = 0.7m to
    determine the total heat transfer
h_x = integrate('Pr_star1^(-2/3)*rho_star1*u*
    Cp_star1*0.0296*(rho_star1*u*x/mu_star1)^(-1/5)',
    'x',0.222,0.7);// [W/m^(2) degree celsius]
// the average heat transfer coefficient in the
    turbulent region is determined from integral h_x
    dx divided by integral dx limit from 0.222 to 0.7
h_bar1 = h_x/(integrate('1','x',0.222,0.7));//[W/m
    ^(2) degree celsius]
// using this value we may calculate the heat
    transfer in the turbulent region of the flat
    plate:
A1 = (L-0.222);// [m^(2)]
q1 = h_bar1*A1*(Tw-(T_aw1-273));// W

// the total amount of cooling required is the sum
```

```
   of the heat transfers for the laminar and
   turbulent portions
 Total_cooling = abs(q)+abs(q1);// [W]
printf("the total amount of cooling required is the
   sum of the heat transfers for the laminar and
   turbulent portions is %f W",Total_cooling);
```

Chapter 6

Empirical and Practical Relations for Forced Convection Heat Transfer

Scilab code Exa 6.1 turbulent heat transfer in a tube

```
clear;
clc;
printf("\t\t\tExample Number 6.1\n\n\n");
// turbulent heat transfer in a tube
// illustration6.1
// solution

p = 2*101325;// [Pa] pressure of air
Ta = 200+273.15;// [K] temperature of air
d = 0.0254;// [m] diameter of tube
R = 287;// [] gas constant
u = 10;// [m/s] velocity of air
dT = 20;// [degree celsius] temperature difference
   between wall and air
// we first calculate the reynolds number to
   determine if the flow is laminar or turbulent,
   and then select the appropriate empirical
```

```
correlation to calculate the heat transfer
// the properties of air at a bulk temperature of 473 K are
rho = p/(R*Ta);// [kg/cubic meter] density of gas
mu = 2.57*10^(-5);// [kg/m s] viscosity
k = 0.0386;// [W/m degree celsius]
Cp = 1025;// [J/kg K]
Pr = 0.681;// prandtl no.
Re_d = rho*u*d/mu;// reynolds number
disp(Re_d, "reynolds number is" );
disp("so that the flow is turbulent");
// we therefore use equation (6-4a) to calculate the heat transfer coefficient
Nu_d = 0.023*Re_d^(0.8)*Pr^(0.4);// nusselt no.
h = Nu_d*k/d;// [W/m^2 degree celsius] heat transfer coefficient
// the heat transfer per unit length is then
q_by_L = h*%pi*d*(dT);// [W/m]
L = 3;// [m]
// we can now make an energy balance to calculate the increase in bulk temperature in a 3 m length of tube :
// q = m_dot*Cp*dT_b = L*(q_byL)
m_dot = rho*u*%pi*d^(2)/4;// [kg/s] gas flow rate
// so that we insert the numerical values in the energy balance to obtain
dT_b = L*q_by_L/(m_dot*Cp);// [degree celsius]
printf("\n heat transfer per unit length is %f W/m", q_by_L);
printf("\n\n bulk temperature increase over the length of 3 m on tube is %f degree celsius ",dT_b);
```

Scilab code Exa 6.2 heating of water in laminar tube flow

```
clear;
clc;
printf("\t\t\tExample Number 6.2\n\n\n");
// heating of water in laminar tube flow
// illustration6.2
// solution

Tw = 60;// [degree celsius] temperature of water
d = 0.0254;// [m] diameter of tube
R = 287;// [] gas constant
u = 0.02;// [m/s] velocity of water
Tw = 80;// [degree celsius] temperature of wall
L = 3;// [m] length of the tube
// we first calculate the reynolds number at the
   inlet bulk temperature to determine the flow
   regime
// the properties of water at  temperature of 333.15
    K are
rho = 985;// [kg/cubic meter] density of gas
mu = 4.71*10^(-4);// [kg/m s] viscosity
k = 0.651;// [W/m degree celsius]
Cp = 4.18*10^3;// [J/kg K]
Pr = 3.02;// prandtl no.
Re_d = rho*u*d/mu;// reynolds number
disp(Re_d, "reynolds number is" );
disp("so that the flow is laminar");
// so the flow is laminar, calculating the
   additional parameter, we have
B = Re_d*Pr*d/L ;
// since the value of B is greater than 10, so
   equation(6-10) is applicable.
// firstly making the calculation on the basis of 60
    degree celsius, determine the exit bulk
   temperature
// the energy balance becomes q = h*pi*d*L(Tw-(Tb1+
   Tb2)/2) = m_dot*Cp*(Tb2-Tb1)   say equation a
// at the wall temperature of 80 degree celsius
mu_w = 3.55*10^(-4);// [kg/m s]
```

```
// from equation (6-10)
Nu_d = 1.86*(B)^(1/3)*(mu/mu_w)^(0.14);
h = k*Nu_d/d;
// the mass flow rate is
m_dot = rho*%pi*d^(2)*u/4;// [kg/s]
// inserting the values in equation a
Tb1 = 60;// [degree celsius]
deff('[y] = f(Tb2)','y = (h*%pi*d*L*(Tw-(Tb1+Tb2)/2)-m_dot*Cp*(Tb2-Tb1))')
Tb2 = fsolve(1,f);
Tb_mean = (Tb1+Tb2)/2;// [degree celsius]
// we obtain the properties at Tb_mean
rho1 = 982;// [kg/m^(3)] density of gas
mu1 = 4.36*10^(-4);// [kg/m s] viscosity
k1 = 0.656;// [W/m degree celsius]
Cp1 = 4.185*10^3;// [J/kg K]
Pr1 = 2.78;// prandtl no.
Re_d1 = Re_d*mu/mu1;
C = Re_d1*Pr1*d/L ;
Nu_d1 = 1.86*(C)^(1/3)*(mu1/mu_w)^(0.14);
h = k1*Nu_d1/d;
// we insert this value of h back into equation a to get
deff('[y] = f(Tb2)','y = (h*%pi*d*L*(Tw-(Tb1+Tb2)/2)-m_dot*Cp*(Tb2-Tb1))')
Tb2 = fsolve(1,f);
printf("\n the exit water temperature is %f degree celsius",Tb2);
```

Scilab code Exa 6.3 heating of air in laminar tube flow for constant heat flux

```
clear;
clc;
printf("\t\t\tExample Number 6.3\n\n\n");
```

```
// heating of air in laminar tube flow for constant
    heat flux
// illustration6.3
// solution

p = 101325;// [Pa] pressure of air
Ta = 27;// [degree celsius] temperature of air
d = 0.005;// [m] diameter of tube
R = 287;// [] gas constant
u = 3;// [m/s] velocity of air
L = 0.1;// [m] length of tube
Tb = 77;// [degree celsius] exit bulk temperature
// we first must evaluate the flow regime and do so
    by taking properties at the average bulk
    temperature
Tb_bar = (Ta+Tb)/2;// [degree celsius]
v = 18.22*10^(-6);// [square meter/s] kinematic
    viscosity
k = 0.02814;// [W/m degree celsius]
Cp = 1006;// [J/kg K]
Pr = 0.703;// prandtl no.
Re_d = u*d/v;// reynolds number
disp(Re_d, "reynolds number is" );
disp("so that the flow is laminar");
// so that the flow is laminar
//the tube length is short, so we expect a thermal
    entrance effect and shall consult figure(6-5)
// the inverse Graetz number is computed as
Gz_inverse = L/(Re_d*Pr*d);
// therefore, for qw = constant, we obtain the
    nusselt number at exit from figure (6-5) as
Nu = 4.7;
// the total heat transfer is obtained in terms of
    the overall energy balance
// at entrance
rho = 1.1774;// [kg/cubic meter] density
// mass flow is
m_dot = rho*%pi*d^(2)*u/4;// [kg/s]
```

```
q = m_dot*Cp*(Tb-Ta);// [W]
// thus we may find the heat transfer without the
   actually determining wall temperatures or values
   of h. However, to determine Tw we must compute qw
    for insertion in equation(b). we have
qw = q/(%pi*d*L);// [W]
// now
Tw = Tb+(qw*d/(Nu*k));// [degree celsius]
// and the heat transfer coefficient is
h = qw/(Tw-Tb);// [W/square meter degree celsius]
printf("\n total heat transfer is %f W",q);
printf("\n\n exit wall temperature is %f degree
   celsius",Tw);
printf("\n\n heat transfer coefficient is %f W/
   square meter degree celsius",h);
```

Scilab code Exa 6.4 heating of air with isothermal tube wall

```
clear;
clc;
printf("\t\t\tExample Number 6.4\n\n\n");
// heating of air with isothermal tube wall
// illustration6.4
// solution

p = 101325;// [Pa] pressure of air
Ta = 27;// [degree celsius] temperature of air
d = 0.005;// [m] diameter of tube
R = 287;// [] gas constant
u = 3;// [m/s] velocity of air
L = 0.1;// [m] length of tube
Tb = 77;// [degree celsius] exit bulk temperature
// we first must evaluate the flow regime and do so
   by taking properties at the average bulk
   temperature
```

```
Tb_bar = (Ta+Tb)/2;// [degree celsius]
v = 18.22*10^(-6);// [square meter/s] kinematic viscosity
k = 0.02814;// [W/m degree celsius]
Cp = 1006;// [J/kg K]
Pr = 0.703;// prandtl no.
Re_d = u*d/v;// reynolds number
disp(Re_d, "reynolds number is" );
disp("so that the flow is laminar");
// so that the flow is laminar
// now we determine Nu_d_bar for Tw = constant. for Gz_inverse = 0.0346 we read
Nu_d = 5.15;
// we thus calculate the average heat transfer coefficient as
h_bar = Nu_d*k/d;// [W/square meter degree celsius]
// we base the heat transfer on a mean bulk temperature of Tb_bar, so that
Tw = 3.49/(h_bar*%pi*d*L)+Tb_bar;// [degree celsius]
printf("\n exit wall temperature is %f degree celsius",Tw);
```

Scilab code Exa 6.5 heat transfer in a rough tube

```
clear;
clc;
printf("\t\t\tExample Number 6.5\n\n\n");
// heat transfer in a rough tube
// illustration6.5
// solution

Tw = 90;// [degree celsius] temperature of tube wall
d = 0.02;// [m] diameter of tube
u = 3;// [m/s] velocity of air
Tw_i = 40;// [degree celsius] entering water
```

```
temperature
Tw_f = 60;// [degree celsius] leaving water temperature
Cp = 4.174*10^3;// [J/kg K]
Tb_bar = (Tw_i+Tw_f)/2;// [degree celsius]
// we first calculate the heat transfer from q = m_dot*Cp*dTb
q = 989*3*%pi*0.01^(2)*4174*(Tw_f-Tw_i);// [W]
// for the rough tube condition , we may employ the Petukhov relation , equation (6-7) The mean film temperaturee is
Tf = (Tw+Tb_bar)/2;// [degree celsius]
// and the fluid properties are
rho = 978;// [kg/cubic meter] density of gas
mu = 4.0*10^(-4);// [kg/m s] viscosity
k = 0.664;// [W/m degree celsius]
Pr = 2.54;// prandtl no.
// also
mu_b = 5.55*10^(-4);// [kg/m s] viscosity
mu_w = 2.81*10^(-4);// [kg/m s] viscosity
// the reynolds number is thus
Re_d = rho*u*d/mu;
// consulting figure(6-14), we find the friction factor as
f_f = 0.0218;
// because Tw>Tb, we take
n = 0.11;
// and obtain
Nu_d = ((f_f*Re_d*2.54)/((1.07+12.7*(f_f/8)^(0.5)*(2.54^(2/3)-1))*8))*(mu_b/mu_w)^(n);
h = Nu_d*k/d;// [W/square meter degree celsius]
// the tube length is obtained from energy balance
L = q/(h*%pi*d*(Tw-Tb_bar));// [m]
printf("the length of tube necessary to accomplish the heating is %f m",L);
```

Scilab code Exa 6.6 turbulent heat transfer in a short tube

```
clear;
clc;
printf("\t\t\tExample Number 6.6\n\n\n");
// turbulent heat transfer in a short tube
// illustration6.6
// solution

p = 101325;// [Pa] pressure of air
Ta = 300;// [K] temperature of air
d = 0.02;// [m] diameter of tube
u = 40;// [m/s] velocity of air
L = 0.1;// [m] length of tube
dT = 5;// [degree celsius] rise in temperature
// we first must evaluate the air properties at 300 K
v = 15.69*10^(-6);// [square meter/s] kinematic viscosity
k = 0.02624;// [W/m degree celsius]
Cp = 1006;// [J/kg K]
Pr = 0.70;// prandtl no.
rho = 1.18;// [kg/cubic meter]
Re_d = u*d/v;// reynolds number
disp(Re_d,"reynolds number is ");
disp("so the flow is turbulent");
// consulting figure (6-6) for this value of Re_d and L/d = 5 we find
Nu_x_by_Nu_inf = 1.15;
// or the heat transfer coefficient is about 15 percent higher that it would be for thermally developed flow.
// we calculate heat-transfer for developed flow using
```

```
Nu_d = 0.023*Re_d^(0.8)*Pr^(0.4);
// and
h = k*Nu_d/d;// [W/square meter degree celsius]
// increasing this value by 15 percent
h = 1.15*h;// [W/square meter degree celsius]
// the mass flow is
Ac = %pi*d^(2)/4;// [square meter]
m_dot = rho*u*Ac;// [kg/s]
// so the total heat transfer is
A = %pi*d*L;// [square meter]
q_by_A = m_dot*Cp*dT/A;// [W/square meter]
printf("\n\n the constant heat flux that must be applied at the tube surface to result in an air temperature rise of 5 degree celsius is %f W/square meter",q_by_A);
Tb_bar = (Ta+(Ta+dT))/2;// [K]
Tw_bar = Tb_bar+q_by_A/h;// [K]
printf("\n\n average wall temperature is %f K",Tw_bar);
```

Scilab code Exa 6.7 airflow across isothermal cylinder

```
clear;
clc;
printf("\t\t\tExample Number 6.7\n\n\n");
// airflow across isothermal cylinder
// illustration6.7
// solution

p = 101325;// [Pa] pressure of air
Ta = 35+273.15;// [K] temperature of air
d = 0.05;// [m] diameter of tube
R = 287;// [] gas constant
u = 50;// [m/s] velocity of air
Tc = 150+273.15;// [degree celsius] cylinder
```

```
temperature
// we first find the reynolds number and then find
   the applicable constants from table(6-2) for use
   with equation (6-17)
// the properties of air are evaluated at the film
   temperature:
Tf = (Ta+Tc)/2;// [K]
rho_f = p/(R*Tf);// [kg/cubic meter]
mu_f = 2.14*10^(-5);// [kg/m s]
k_f = 0.0312;// [W/m degree celsius]
Pr_f = 0.695;// prandtl number
Re_f = rho_f*u*d/mu_f;// reynolds number
// from table (6-2) table
C = 0.0266;
n = 0.805;
// so from equation (6-17)
h = C*(Re_f)^(n)*(Pr_f)^(1/3)*k_f/d;// [W/square
   meter degree celsius] heat transfer coefficient
// the heat transfer per unit length is
q_by_L = h*%pi*d*(Tc-Ta);// [W/m]
printf("heat loss per unit length of cylinder is %f
   W/m",q_by_L);
```

Scilab code Exa 6.8 heat transfer from electrically heated

```
clear;
clc;
printf("\t\t\tExample Number 6.8\n\n\n");
// heat transfer from electrically heated
// illustration6.8
// solution

p = 101325;// [Pa] pressure of air
Tw = 25+273.15;// [K] temperature of air
d = 3.94*10^(-5);// [m] diameter of wire
```

```
R = 287;// [] gas constant
u = 50;// [m/s] velocity of air perpendicular to the air
Tr = 50+273.15;// [degree celsius] rise in surface temperature
// we first obtain the properties at the film temperature :
Tf = (Tw+Tr)/2;// [K]
v_f = 16.7*10^(-6);// [square meter/s]
k = 0.02704;// [W/m degree celsius]
Pr_f = 0.706;// prandtl number
Re_d = u*d/v_f;// reynolds number
// the Peclet number is
Pe = Re_d*Pr_f;
// and we find that equations (6-17),(6-21), or (6-19) apply.
// let us make the calculation with both the simplest expression, (6-17),and the most complex ,(6-21), and compare results.
// using equation (6-17) with
C = 0.683;
n = 0.466;
// we have
Nu_d = 0.683*Re_d^(n)*Pr_f^(1/3);
// the value of heat transfer coefficient is
h = Nu_d*k/d;// [W/square meter degree celsius]
// the heat transfer per unit length is then
q_by_L = %pi*d*h*(Tr-Tw);// [W/m]
// using equation (6-21), we calculate the nusselt no as
Nu_d1 = 0.3+((0.62*Re_d^(1/2)*Pr_f^(1/3))/((1+(0.4/Pr_f)^(2/3))^(1/4))*((1+(Re_d/282000)^(5/8))^(4/5)));
h1 = Nu_d1*k/d;// [W/square meter degree celsius]
// and
q_by_L1 = h1*%pi*d*(Tr-Tw);// [W/m]
printf("heat lost per unit length by the wire is %f W/m",q_by_L1);
```

Scilab code Exa 6.9 heat transfer from sphere

```
clear;
clc;
printf("\t\t\tExample Number 6.9\n\n\n");
// heat transfer from sphere
// illustration6.9
// solution

p = 101325;// [Pa] pressure of air
Ta = 27+273.15;// [K] temperature of air
d = 0.012;// [m] diameter of sphere
u = 4;// [m/s] velocity of air
Ts = 77+273.15;// [degree celsius] surface
    temperature of sphere
// consulting equation (6-30) we find that the
    reynolds number is evaluated at the free-stream
    temperature.
// we therefore need the following properties at Ta
    = 300.15K
v = 15.69*10^(-6);// [square meter/s]
k = 0.02624;// [W/m degree celsius]
Pr = 0.708;// prandtl number
mu_inf = 1.8462*10^(-5);// [kg/m s]
// at Ts = 350K
mu_w = 2.075*10^(-5);// [kg/m s]
Re_d = u*d/v;// reynolds number
// from equation (6-30),
Nu_bar = 2+((0.4)*(Re_d)^(1/2)+0.06*(Re_d)^(2/3))*(
    Pr^(0.4))*((mu_inf/mu_w)^(1/4));
// and
h_bar = Nu_bar*k/d;// [W/square meter degree celsius
    ] heat transfer coefficient
// the heat transfer is then
```

```
A = 4*%pi*d^(2)/4;// [square meter] area of sphere
q = h_bar*A*(Ts-Ta);// [W]
// for comparison purposes let us also calculate the heat-transfer coefficient using equation(6-25). the film temperature is
Tf = (Ta+Ts)/2;// [K]
v_f = 18.23*10^(-6);// [square meter/s]
k_f = 0.02814;// [W/m degree celsius]
// reynolds number is
Re_d1 = u*d/v_f;
// from equation (6-25)
Nu_f = 0.37*(u*d/v_f)^(0.6);
// and h_bar is calculated as
h_bar = Nu_f*k_f/d;// [W/square meter degree celsius]
printf("heat lost by the sphere is %f W",q);
```

Scilab code Exa 6.10 heating of air with in line tube bank

```
clear;
clc;
printf("\t\t\tExample Number 6.10\n\n\n");
// heating of air with in-line tube bank
// Example 6.10(page number-300-302)
// solution

p = 101325;// [Pa] pressure of air
Ta = 10+273.15;// [K] temperature of air
d = 0.0254;// [m] diameter of tubes
Sp = 0.0381;// [m] spacing between tubes in normal and parallel direction
Sn = Sp;
R = 287;// [] universal gas constant
u = 7;// [m/s] velocity of air
Ts = 65+273.15;// [K] surface temperature of tubes
```

```
// the constants for use with equation (6-17) may be obtained from table 6-4(page no.-298),using
Sp_by_d = Sp/d;
Sn_by_d = Sn/d;
// so that
C = 0.278;
n = 0.620;
// the properties of air are evaluated at the film temperature, which at entrance to the tube bank is
Tf = (Ta+Ts)/2;// [K]
rho_f = p/(R*Tf);// [kg/cubic meter]
mu_f = 1.894*10^(-5);// [kg/m s]
k_f = 0.027;// [W/m degree celsius]
Pr_f = 0.706;// prandtl number
Cp = 1007;// [J/Kg degree celsius]
// the maximum velocity is therefore
u_max = u*Sn/(Sn-d);// [m/s]
// the reynolds number is computed by using the maximum velocity
Re = rho_f*u_max*d/mu_f;
// the heat transfer coefficient is calculated by using equation(6-17)
h = C*Re^(n)*Pr_f^(1/3)*k_f/d;// [W/square meter degree celsius]
// multiplying by 0.92 from table 6-5 (page no.-298) to correct for only five tube rows gives
h = 0.92*h;// [W/square meter degree celsius]
// the total surface area for heat transfer, considering unit length of tubes is
N = 15*5;// ttal no. of tubes
A = N*%pi*d*1;// [square meter/m]
// befre calculating the heat transfer, we must recognize thet the air temperature increases as the air flows thrugh the tube bank.
// therefore, this must be taken into account when using q=h*A*(Ts-Ta)
// as a good approximatin, we can use an arithmetic
```

```
average value of Tinf and write for the energy balance
// say the equation A is       q = h*A*(Ts-(Tinf1+Tinf2)/2) = m_dot*Cp*(Tinf2-Tinf1)
// where now the subscripts 1 and 2 designate entrance and exit to the tube bank.
// the mass flow to the entrance to the 15 tubes is
rho_inf = p/(R*Ta);// [Kg/m^(3)]
m_dot=rho_inf*u*15*Sn;// [kg/s]
// so that equation A becomes after inserting the values and solving
Tinf1 = Ta;// [K]
deff('[y] = f1(Tinf2)','y = (h*A*(Ts-(Tinf1+Tinf2)/2)-m_dot*Cp*(Tinf2-Tinf1))')
Tinf2=fsolve(1,f1);
// the heat transfer is then obtained from the right side of equation A
q = m_dot*Cp*(Tinf2-Ta);// [W/m]
printf("the exit air temperature is %f degree celsius",Tinf2-273);
printf("\n\n heat transfer per unit length for the tube bank is %f kW/m",q/1000);
```

Scilab code Exa 6.11 alternate calculation method

```
clear;
clc;
printf("\t\t\tExample Number 6.11\n\n\n");
// alternate calculation method
// example 6.10 (page no.-302)
// solution

// data for this example is taken from previous example (6-10)
// properties for use in equation (6-34) are
```

```
evaluated at free-atream conditions of 10 degree celsius
v = 14.2*10^(-6);// [square meter/s]
k = 0.0249;// [W/m degree celsius]
Pr = 0.712;// prandtl number
Pr_w = 0.70;// prandtl number
u = 7;// [m/s] velocity of air
Sp = 0.0381;// [m] spacing between normal and parallel direction to the flow
Sn = 0.0381;// spacing between normal and parallel direction to the flow
d = 0.0254;// [m] diameter of tube
//maximum velocity is
u_max = u*(Sn/(Sn-d));// [m/s]
// the reynolds number is
Re_d_max = u_max*d/v;
// so that the constants for equation (6-34) are
C = 0.27;
n = 0.63;
// inserting values we obtain
h = C*Re_d_max^(n)*(Pr/Pr_w)^(1/4)*k/d;// [W/square meter degree celsius] heat transfer coefficient
// multiplying by 0.92 from table 6-7 (page no.-300) to correct for only five tube rows gives
h = 0.92*h;// [W/square meter degree celsius]
printf("the value of heat transfer coefficient is %f W/square meter degree celsius",h);
h_in = 163.46432;// [W/square meter degree celsius] from previous example
printf("\n\n the value of heat transfer coefficient for previous problem is %f W/square meter degree celsius",h_in);
P = (h-h_in)*100/h_in;
printf("\n\n percentage increase in value of h is %f ",P);
```

Scilab code Exa 6.12 heating of liquid bismuth in tube

```
clear;
clc;
printf("\t\t\tExample Number 6.12\n\n\n");
// heating of liquid bismuth in tube
// example 6.11 (page no.-305-6)
// solution

m_dot = 4.5;// [Kg/s] flow rate of bismuth
d = 0.05;// [m] diameter of steel tube
Ti = 415;// [degree celsius] initial temperature of
    bismuth
Tf = 440;// [degree celsius] final temperature of
    bismuth
// because a constant heat flux is maintained, we
    may use equation 6-47 to calculate the heat
    transfer coefficient.
// the properties of bismuth are evaluated at the
    average bulk temperature of
Ta = (Ti+Tf)/2;// [degree celsius]
mu = 1.34*10^(-3);// [Kg/m s] viscosity
Cp = 149;// [J/Kg degree celsius] heat
k = 15.6;// [W/m degree celsius]
Pr = 0.013;// prandtl number
// the total transfer is calculated from
q = m_dot*Cp*(Tf-Ti);// [W]
// we calculate reynolds and peclet number as
G = m_dot/(%pi*d^(2)/4);
Re_d = d*G/mu;
Pe = Re_d*Pr;
// the heat transfer coefficient is calculated from
    equation 6-47
Nu_d = 4.82+0.0185*Pe^(0.827);
```

```
h = Nu_d*k/d;// [W/square meter degree celsius]
// the total required surface area of the tube may
   now be computed from q=h*A*DT
// where we use the temperature difference of
DT = 20;// [degree celsius]
A = q/(h*DT);// [square meter]
// the area in turn can be expressed in terms of
   tube length
L = A/(%pi*d);// [m]
printf("Length of tube required to effect the heat
   transfer is %f m",L);
```

Chapter 7

Natural Convection Systems

Scilab code Exa 7.1 constant heat flux from vertical plate

```
clear;
clc;
printf("\t\t\tExample Number 7.1\n\n\n");
// constant heat flux from vertical plate
// Example 7.1 (page no.-330-331)
// solution

q_w = 800;// [W/square meter] radiant energy flux
H = 3.5;// [m] height of metal plate surface
W = 2;// [m] width of metal plate
Ta = 30;// [degree celsius] surrounding air
    temperature
// we treat this problem as one with constant heat
    flux on the surface since we do not know the
    surface temperature, we must make an estimate for
     determining Tf and the air properties.
// an approximate value of h for free convection
    problems is
h = 10;// [W/square meter degree celsius]
dT = q_w/h;// [degree celsius]
// then
```

```
Tf = (dT/2)+Ta;// [degree celsius] approximately
// at Tf the properties of air are
v = 2.005*10^(-5);// [square meter/s]
k = 0.0295;// [W/m degree celsius]
Pr = 0.7;// prandtl number
Beta = 1/(Tf+273);// [K^(-1)]
// from equation (7-30), with
x = 3.5;// [m]
g = 9.8;// [square meter/s] acceleration due to
    gravity
Gr_x = (g*Beta*q_w*x^(4))/(k*v^(2));
// we may therefore use equation (7-32) to evaluate
    h_x
h_x = (k*0.17*(Gr_x*Pr)^(1/4))/x;// [W/square meter
    degree celsius]
// in the turbulent heat transfer governed by
    equation (7-32), we note that
// Nu_x = h*x/k ~ (Gr_x)^(1/4) ~ x
// or h_x doest noy vary with x, and we may take
    this as the average value. the value of h
h = 5.41;// [W/square meter degree celsius]
// is less than the approximate value we used to
    estimate Tf, recalculating dT, we obtain
dT1 = q_w/h_x;// [degree celsius]
// our new film temperature would be
Tf1 = Ta+dT1/2;// [degree celsius]
// at Tf the properties of air are
v1 = 2.354*10^(-5);// [square meter/s]
k1 = 0.0320;// [W/m degree celsius]
Pr1 = 0.695;// prandtl number
Beta1 = 1/(Tf1+273);// [K^(-1)]
// then
Gr_x1 = (g*Beta1*q_w*x^(4))/(k1*v1^(2));
// and h_x is caalculated from
h_x1 = (k1*0.17*(Gr_x1*Pr1)^(1/4))/x;// [W/square
    meter degree celsius]
// our new temperature difference is calculated as
dT2 = q_w/h_x1;// [degree celsius]
```

```
// the average wall temperature is therefore
T_w_avg = dT2+Ta;// [degree celsius]
printf("the average wall temperature is therefore %f
    degree celsius",T_w_avg);
```

Scilab code Exa 7.2 heat transfer from isothermal vertical plate

```
clear;
clc;
printf("\t\t\tExample Number 7.2\n\n\n");
// heat transfer from isothermal vertical plate
// Example 7.2 (page no.-332)
// solution

H = 4;// [m] height of vertical plate
Tp = 60;// [degree celsius] plate temperature
Ta = 10;// [degree celsius] atmospheric temperature
// we first determine the film temperature as
Tf = (Tp+Ta)/2;// [degree celsius]
// the properties of interest are thus
v = 16.5*10^(-6);// [square meter/s]
k = 0.02685;// [W/m degree celsius]
Pr = 0.7;// prandtl number
Beta = 1/(Tf+273);// [K^(-1)]
g = 9.8;// [square meter/s] acceleration due to
   gravity
// and
Gr_into_Pr = (g*Beta*(Tp-Ta)*H^(3)*Pr)/(v^(2));
// we then may use equation (7-29) to obtain
Nu_bar_root = 0.825+(0.387*(Gr_into_Pr)^(1/6))
   /(1+(0.492/Pr)^(9/16))^(8/27);
Nu_bar = (Nu_bar_root)^(2);
// the heat transfer coefficient is
h_bar = Nu_bar*k/H;// [W/square meter degree celsius
   ]
```

```
// the heat transfer is
A = H*10;// [square meter] for 10 m wide plate
q = h_bar*A*(Tp-Ta);// [W]
// as an alternative, we could employ the simpler
   relation
Nu = 0.1*(Gr_into_Pr)^(1/3);
printf("heat transfer if the plate is 10 m wide is
   %f W",q);
```

Scilab code Exa 7.3 heat transfer from horizontal tube in water

```
clear;
clc;
printf("\t\t\tExample Number 7.3\n\n\n");
// heat transfer from horizontal tube in water
// Example 7.3 (page no.-333)
// solution

d = 0.02;// [m] diameter of heater
Ts = 38;// [degree celsius] surface temperature of
   heater
Tw = 27;// [degree celsius] water temperature
// the film temperature is
Tf = (Ts+Tw)/2;// [degree celsius]
// from appendix A the properties of water are
k = 0.630;// [W/m degree celsius] thermal
   conductivity
// and the following term is particularly useful in
   obtaining the product GrPr product when it is
   multiplied by d^(3)*DT
// g*Beta*rho^(2)*Cp/(mu*k) = 2.48*10^(10) [1/m^(3)
   degree celsius]
K = 2.48*10^(10);// [1/m^(3) degree celsius]
Gr_into_Pr = K*(Ts-Tw)*d^(3);
// using table 7-1 (page number -328), we get
```

```
C = 0.53;
m = 1/4;
// so that
Nu = C*(Gr_into_Pr)^(1/4);
h = Nu*k/d;// [W/square meter degree celsius]
    convection heat transfer coefficient
// the heat transfer is thus
q_by_L = h*%pi*d*(Ts-Tw);// [W/m]
printf("free-convection heat loss per unit length of
     heater is %f W/m",q_by_L);
```

Scilab code Exa 7.4 heat transfer from fine wire in air

```
clear;
clc;
printf("\t\t\tExample Number 7.4\n\n\n");
// heat transfer from fine wire in air
// Example 7.4 (page no.-333-334)
// solution

d = 0.00002;// [m] diameter of wire
L = 0.5;// [m] length of wire whose temperature is
    maintained
Ts = 54;// [degree celsius] surface temperature of
    wire
Pa = 101325;// [Pa] pressure of air
Ta = 0;// [degree celsius] temperature of air
// we first determine the film temperature as
Tf = (Ts+Ta)/2;// [degree celsius]
// the properties of interest are thus
v = 15.69*10^(-6);// [square meter/s]
k = 0.02624;// [W/m degree celsius]
Pr = 0.708;// prandtl number
Beta = 1/(Tf+273);// [K^(-1)]
g = 9.8;// [square meter/s] acceleration due to
```

```
    gravity
// and
Gr_into_Pr = (g*Beta*(Ts-Ta)*d^(3)*Pr)/(v^(2));
// from table 7-1 we find
C = 0.675;
m = 0.058;
// so that
Nu_bar = C*(Gr_into_Pr)^(m);
h_bar = Nu_bar*k/d;// [W/square meter degree celsius
    ]
// the heat required is
A = %pi*d*L;// [square meter] surface area of wire
q = h_bar*A*(Ts-Ta);// [W]
printf("electric power necessary to maintain the the
     wire temperature if the length is 0.5 m is %f W"
    ,q);
```

Scilab code Exa 7.5 heated horizontal pipe in air

```
clear;
clc;
printf("\t\t\tExample Number 7.5\n\n\n");
// heated horizontal pipe in air
// Example 7.5 (page no.-334-335)
// solution

d = 0.3048;// [m] diameter of pipe
Ts = 250;// [degree celsius] surface temperature of
    pipe
Ta = 15;// [degree celsius] temperature of air
// we first determine the Grashof-prandtl number
    product and then select the appropriate constants
     from table 7-1(page no.-328) for use with
                  equation (7-25)
// the properties of air are evaluated at the film
```

```
    temperature:
Tf = (Ts+Ta)/2;// [degree celsius]
// the properties of interest are thus
v = 26.54*10^(-6);// [square meter/s]
k = 0.03406;// [W/m degree celsius]
Pr = 0.687;// prandtl number
Beta = 1/(Tf+273);// [K^(-1)]
g = 9.8;// [square meter/s] acceleration due to
    gravity
Gr_d_into_Pr = g*Beta*(Ts-Ta)*d^(3)*Pr/(v^(2));
// from table 7-1
C = 0.53;
m = 1/4;
Nu_d = C*(Gr_d_into_Pr)^(m);
h = Nu_d*k/d;// [W/square meter degree celsius]
// the heat transfer per unit length is then
    calculated from
q_by_L = h*%pi*d*(Ts-Ta);// [W/m]
printf("free-convection heat loss per unit length is
     %f kW/m",q_by_L/1000);
```

Scilab code Exa 7.6 cube cooling in air

```
clear;
clc;
printf("\t\t\tExample Number 7.6\n\n\n");
// cube cooling in air
// Example 7.6 (page no.-336)
// solution

L = 0.2;// [m] side length of cube
Ts = 60;// [degree celsius] surface temperature of
    cube
Ta = 10;// [degree celsius] air temperature
// this is an irregular solid so we use the
```

```
    information in the last entry of table 7-1(page
    no.-328) in the absence of a specific correlation
     for this   geometry.
// the properties were evaluated as
v = 17.47*10^(-6);// [square meter/s]
k = 0.02685;// [W/m degree celsius]
Pr = 0.70;// prandtl number
Beta = 3.25*10^(-3);// [K^(-1)]
g = 9.8;// [square meter/s] acceleration due to
    gravity
// the characteristic length is the distance a
    particle travels in the boundary layer, which is
    L/2 along the bottom plus L along the side plus L
    /2 on  the top or
Gr_into_Pr = (g*Beta*(Ts-Ta)*(2*L)^(3)*Pr)/(v^(2));
// from the last entry in table 7-1 we find
C = 0.52;
n = 1/4;
// so that
Nu = C*(Gr_into_Pr)^(n);
h_bar = Nu*k/(2*L);// [W/square meter degree celsius
    ]
// the cube has six sides so the area is
A = 6*L^(2);// [square meter]
// the heat required is
q = h_bar*A*(Ts-Ta);// [W]
printf("heat transfer is %f W",q);
```

Scilab code Exa 7.7 calculation with simplified relations

```
clear;
clc;
printf("\t\t\tExample Number 7.7\n\n\n");
// calculation with simplified relations
// Example 7.7 (page no.-338-339)
```

```
// solution

// this example is calculation of heat transfer with
    simplified relations for example (7.5) so we use
    the data of example 7.5

d = 0.3048;// [m] diameter of pipe
Ts = 250;// [degree celsius] surface temperature of
   pipe
Ta = 15;// [degree celsius] temperature of air
// we first determine the Grashof-prandtl number
   product and then select the appropriate constants
    from table 7-1(page no.-328) for use with
               equation (7-25)
// the properties of air are evaluated at the film
   temperature:
Tf = (Ts+Ta)/2;// [degree celsius]
// the properties of interest are thus
v = 26.54*10^(-6);// [square meter/s]
k = 0.03406;// [W/m degree celsius]
Pr = 0.687;// prandtl number
Beta = 1/(Tf+273);// [K^(-1)]
g = 9.8;// [square meter/s] acceleration due to
   gravity
// in example (7.5) we found that a rather large
   pipe with a substantial temperature difference
   between the surface and air still had a GrPr
   product of  1.57*10^(8)<10^(9), so laminar
   equation is selected from table 7-2(page no.-339)
   . the heat transfer coefficient is given by
h = 1.32*((Ts-Ta)/d)^(1/4);// [W/square meter degree
    celsius]
// the heat transfer is then
q_by_L = h*%pi*d*(Ts-Ta);// [W/m]
printf("heat transfer is %f kW/m",q_by_L/1000);
```

Scilab code Exa 7.8 heat transfer across vertical air gap

```
clear;
clc;
printf("\t\t\tExample Number 7.8\n\n\n");
// heat transfer across vertical air gap
// Example 7.8 (page no.-345)
// solution

L = 0.5;// [m] side length vertical square plate
d = 0.015;// [m] distance between plates
p = 101325;// [Pa] pressure of air
R = 287;// [] universal gas constant
T1 = 100;// [degree celsius] temperature of first plate
T2 = 40;// [degree celsius] temperature of second plate
E = 0.2;// emissivity of both surfaces
// the properties of air is evaluated at the mean temperature
Tf = (T1+T2)/2;// [degree celsius]
rho = p/(R*(Tf+273));// [Kg/m^(3)] density
k = 0.0295;// [W/m degree celsius]
Pr = 0.70;// prandtl number
Beta = 1/(Tf+273);// [K^(-1)]
mu = 2.043*10^(-5);// [Kg/m s] viscosity
g = 9.8;// [square meter/s] acceleration due to gravity
// the Grashof-prandtl number product is now calculated as
Gr_into_Pr = (g*rho^(2)*Beta*(T1-T2)*(d)^(3)*Pr)/(mu^(2));
// we may now use equation (7-64) to calculate the effective thermal conductivity, with
```

```
L = 0.5;// [m]
del = 0.015;// [m]
// and the constants taken from table 7-3(page no
    .-344):
Ke_by_K = 0.197*(Gr_into_Pr)^(1/4)*(L/del)^(-1/9);
// the heat transfer may now be calculated with
    equation (7-54). the area is
A = L^(2);// [square meter]
q = Ke_by_K*k*A*(T1-T2)/del;// [W]
 // the radiation flux is calculated with equation
     (7-67), taking
T1 = 373;// [K]
T2 = 313;// [K]
E1 = E;
E2 = E;
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
q_A = sigma*(T1^(4)-T2^(4))/((1/E1)+(1/E2)-1);// [W/
    square meter]
q_rad = A*q_A;// [W]
printf("free-convection heat transfer across the air
     space is %f W",q);
printf("\n\nradiation heat transfer across the air
    space is %f W",q_rad);
```

Scilab code Exa 7.9 heat transfer across horizontal air gap

```
clear;
clc;
printf("\t\t\tExample Number 7.9\n\n\n");
// heat transfer across horizontal air gap
// Example 7.9 (page no.-346)
// solution

a = 0.2;// [m] side length of plate
d = 0.01;// [m] seperation between two plates
```

```
p = 101325;// [Pa] pressure of air
R = 287;// [] universal gas constant
T1 = 100;// [degree celsius] temperature of first
    plate
T2 = 40;// [degree celsius] temperature of second
    plate
// the properties are the same as given in example
    (7.8)
Tf = (T1+T2)/2;// [degree celsius]
rho = p/(R*(Tf+273));// [Kg/m^(3)] density
k = 0.0295;// [W/m degree celsius]
Pr = 0.70;// prandtl number
Beta = 1/(Tf+273);// [K^(-1)]
mu = 2.043*10^(-5);// [Kg/m s] viscosity
g = 9.8;// [square meter/s] acceleration due to
    gravity
// the GrPr product is evaluated on the basis of the
     separating distance, so we have
Gr_into_Pr = (g*rho^(2)*Beta*(T1-T2)*(d)^(3)*Pr)/(mu
    ^(2));
// consulting table 7-3(page no.-344) we find
C = 0.059;
n = 0.4;
m = 0;
Ke_by_K = C*(Gr_into_Pr)^(n)*(a/d)^(m);
A = a^(2);// [square meter] area of plate
q = Ke_by_K*k*A*(T1-T2)/d;// [W]
printf("heat transfer across the air space is %f W",
    q);
```

Scilab code Exa 7.10 heat transfer across water layer

```
clear;
clc;
printf("\t\t\tExample Number 7.10\n\n\n");
```

```
// heat transfer across water layer
// Example 7.10 (page no.-346-347)
// solution

L = 0.5;// [m] length of square plate
d = 0.01;// [m] seperation between square plates
T1 = 100;// [degree F] temperature of lower plate
T2 = 80;// [degree F] temperature of upper plate
// we evaluate properties at mean temperature of 90
    degree F and obtain , for water
k = 0.623;// [W/m degree celsus]
// and the following term is particularly useful in
    obtaining the product GrPr
// g*Beta*rho^(2)*Cp/(mu*k) = 2.48*10^(10) [1/m^(3)
    degree celsius]
// the Grashof-prandtl number product is now
    evaluated using the plate spacing of 0.01 m as
    the characterstic dimension
K = 2.48*10^(10);// [1/m^(3) degree celsius]
Gr_into_Pr = K*(T1-T2)*(5/9)*d^(3);
// now, using equation 7-64 and consulting table
    7-3(page no.-344) we obtain
C = 0.13;
n = 0.3;
m = 0;
// therefore , equation (7-64) becomes
Ke_by_K = C*Gr_into_Pr^(n);
// the effectve thermal conductivity is thus
ke = k*Ke_by_K;// [W/m degree celsius]
// and the heat transfer is
A = L^(2);// [square meter] area of plate
q = ke*A*(T1-T2)*(5/9)/d;// [W]
printf("heat lost by the lower plate is %f W",q);
```

Scilab code Exa 7.11 reduction of convection in ar gap

```
clear;
clc;
printf("\t\t\tExample Number 7.11 \n\n\n");
// reduction of convection in ar gap
// Example 7.11 (page no.-347)
// solution

Tm = 300;// [K] mean temperature of air
dT = 20;// [degree celsius] temperature difference
R = 287;// [] universal gas constant
g = 9.81;// [m/s^(2)] acceleration due to gravity
p_atm = 101325;// [Pa] atmospheric pressure
// consulting table 7-13(page no.-344), we find that
    for gases, a value Grdel_into_Pr <2000 is
   necessary to reduce the system to one of pure
   conduction.
// at 300 K the properties of air are
k = 0.02624;// [W/m degree celsius]
Pr = 0.7;// prandtl no.
mu = 1.846*10^(-5);// [Kg/m s]
Beta = 1/300;
// we have
Grdel_into_Pr = 2000;

// Part A for spacing of 1cm

del = 0.01;// [m] spacing between plate
p = sqrt((Grdel_into_Pr*((R*Tm)^(2))*mu^(2))/(g*Beta
   *dT*del^(3)*Pr));// [Pa]
// or vacuum
vacuum = p_atm-p;// [Pa]
printf("vacuum necessary for glass spacings of 1 cm
   is %f Pa",vacuum);

// Part B for spacing of 2cm

del1 = 0.02;// [m] spacing between plate
p1 = sqrt(Grdel_into_Pr*(R*Tm)^(2)*mu^(2)/(g*Beta*dT
```

```
    *del1^(3)*Pr));// [Pa]
// or vacuum
vacuum1 = p_atm-p1;// [Pa]
printf("\n\n vacuum necessary for glass spacings of
    2 cm is %f Pa",vacuum1);
```

Scilab code Exa 7.12 heat transfer across evacuated space

```
clear;
clc;
printf("\t\t\tExample Number 7.12\n\n\n");
// heat transfer across evacuated space
// Example 7.12 (page no.-351-352)
// solution

E = 0.06;// emmisvity of polished aluminium plate
d = 0.025;// [m] seperation between plates
p = 101325*10^(-6);// [Pa] pressure of air between
    plates
T1 = 100;// [degree celsius] temperature of plate 1
T2 = 30;// [degree celsius] temperature of plate 2
// we first calculate the mean free path to
    determine if low-density effects to be important.
// evaluating properties at the mean air temperature
     of 65 degree celsius , we have
lambda = (2.27*10^(-5)*((T1+T2)/2+273))/(p);// [m]
// since the plate spacing is only 2.5 cm, we should
     expect low-density effects to be important.
// evaluating properties at the mean temperature of
    65 degree celsius , we have
k = 0.0291;// [W/m degree celsius]
Gamma = 1.40;
Pr = 0.7;
alpha = 0.9;// from table 7-4(page no.-350)
// combining equations (7-75)with the central
```

```
   temperature gradient relation gives
// inserting the appropriate properties gives
deff('y = f(dT)','y = dT-((2-alpha)/alpha)*(2*Gamma
   /(Gamma+1))*(lambda/Pr)*((T1-T2-2*dT)/d)');
dT = fsolve(1,f);
// the conduction heat transfer is thus
q_by_A = k*((T1-T2-2*dT)/d);// [W/square meter]
printf("conduction heat transfer through the air gap
    is %f W/square meter",q_by_A);
// at normal atmospheric pressure the conduction
   would be
q_by_A1 = k*((T1-T2)/d);// [W/square meter]
// the radiation heat transfer is calculated with
   equation (8-42), taking E1=E2=0.06 for polished
   aluminium:
sigma = 5.669*10^(-8);// []
q_by_A_rad = sigma*(((T1+273)^(4)-(T2+273)^(4))/((2/
   E)-1));// [W/square meter]
printf("\n\n thus, at the low density condition the
   radiation heat transfer is almost %f times as
   large as the conduction",q_by_A_rad/q_by_A);
```

Scilab code Exa 7.13 combined free and forced convection with air

```
clear;
clc;
printf("\t\t\tExample Number 7.13\n\n\n");
// combined free and forced convection with air
// Example 7.12 (page no.-353-355)
// solution

p = 101325;// [Pa] pressure of air
Ta = 27;// [degree celsius] temperature of air
d = 0.025;// [m] diameter of tube
u = 0.3;// [m/s] velocity of air
```

```
Tw = 140;// [degree celcius] temperature of tube wall
L = 0.4;// [m] length of tube
R = 287;// [] universal gas constant
// the properties of air are evaluated at the film temperature:
Tf = (Tw+Ta)/2;// [degree celcius]
// the properties of interest are thus
kf = 0.0305;// [W/m degree celcius]
Pr = 0.695;// prandtl number
Beta = 1/(Tf+273);// [K^(-1)]
g = 9.8;// [square meter/s] acceleration due to gravity
mu_f = 2.102*10^(-5);// [Kg/m s]
mu_w = 2.337*10^(-5);// [Kg/m s]
rho_f = p/(R*(Tf+273));// [Kg/cubic meter]
// let us take the bulk temperature as 27 degree celsius for evaluating mu_b;then
mu_b = 1.8462*10^(-5);// [Kg/m s]
// the significant parameters are calculated as
Re_f = rho_f*u*d/mu_f;
Gr = rho_f^(2)*g*Beta*(Tw-Ta)*d^(3)/mu_f^(2);
Z = Gr*Pr*d/L;// constant
// according to figure(7-14)(page no.-354), the mixed convection flow regime is encountered. thus we must use equation(7-77).
// The graetz number is calculated as
Gz = Re_f*Pr*d/L;
// and the numerical calculation for equation(7-77) becomes
Nu = 1.75*(mu_b/mu_w)^(0.14)*[Gz+0.012*(Gz*Gr^(1/3))^(4/3)]^(1/3);
// the average heat transfer coefficient is calculated as
h_bar = Nu*kf/d;// [W/square meter degree celsius]
printf("heat transfer coefficient is %f W/square meter degree celsius",h_bar);
```

Chapter 8

Radiation Heat Transfer

Scilab code Exa 8.1 transmission and absorption in a gas plate

```
clear;
clc;
printf("\t\t\tExample Number 8.1\n\n\n");
// transmission and absorption in a gas plate
// Example 8.1 (page no.-381)
// solution

T = 2000+273;// [K] furnace temperature
L = 0.3;// [m] side length of glass plate
t1 = 0.5;// transmissivity of glass between lambda1
   to lambda2
lambda1 = 0.2;// [micro m]
lambda2 = 3.5;// [micro m]
E1 = 0.3;// emissivity of glass upto lambda2
E2 = 0.9;// emissivity of glass above lambda2
t2 = 0;// transmissivity of glass except in the
   range of lambda1 to lambda2
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
A = L^(2);// [square meter] area of glass plate
// calculating constants to use table 8-1(page no
   .-379-380)
```

```
K1 = lambda1*T;// [micro m K]
K2 = lambda2*T;// [micro m K]
// from table 8-1
Eb_0_lam1_by_sigmaT4 = 0;
Eb_0_lam2_by_sigmaT4 = 0.85443;
Eb = sigma*T^(4);// [W/square meter]
// total incident radiation is
// for 0.2 micro m to 3.5 micro m
TIR = Eb*(Eb_0_lam2_by_sigmaT4-Eb_0_lam1_by_sigmaT4)
   *A;// [W]
TRT = t1*TIR;// [W]
RA1 = E1*TIR;// [W] for 0<lambda<3.5 micro m
RA2 = E2*(1-Eb_0_lam2_by_sigmaT4)*Eb*A;// [W] for
   3.5 micro m <lambda< infinity
TRA = RA1+RA2;// [W]
printf("total energy absorbed in the glass is %f kW"
   ,TRA/1000);
printf("\n\n total energy transmitted by the glass
   is %f kW",TRT/1000);
```

Scilab code Exa 8.2 heat transfer between black surfaces

```
clear;
clc;
printf("\t\t\tExample Number 8.2\n\n\n");
// heat transfer between black surfaces
// Example 8.2 (page no.-389-390)
// solution

L = 1;// [m] length of black plate
W = 0.5;// [m] width of black plate
T1 = 1000+273;// [K] first plate temperature
T2 = 500+273;// [K] second plate temperature
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
// the ratios for use with figure 8-12(page no.-386)
```

```
    are
Y_by_D = W/W;
X_by_D = L/W;
// so that
F12 = 0.285;// radiation shape factor
// the heat transfer is calculated from
q = sigma*L*W*F12*(T1^(4)-T2^(4));
printf("net radiant heat exchange between the two
   plates is %f kW",q/1000);
```

Scilab code Exa 8.3 shape factor algebra for open ends of cylinder

```
clear;
clc;
printf("\t\t\tExample Number 8.3\n\n\n");
// shape-factor algebra for open ends of cylinder
// Example 8.3 (page no.-395)
// solution

d1 = 0.1;// [m] diameter of first cylinder
d2 = 0.2;// [m] diameter of second cylinder
L = 0.2;// [m] length of cylinder
// we use the nomenclature of figure 8-15(page no
   .-388) for this problem and designate the open
   ends as surfaces 3 and 4.
// we have
L_by_r2 = L/(d2/2);
r1_by_r2 = 0.5;
// so from figure 8-15 or table 8-2(page no.-389) we
    obtain
F21 = 0.4126;
F22 = 0.3286;
// using the reciprocity relation (equation 8-18) we
    have
F12 = (d2/d1)*F21;
```

```
// for surface 2 we have F12+F22+F23+F24 = 1.0
// and from symmetry F23 = F24 so that
F23 = (1-F21-F22)/2;
F24 = F23;
// using reciprocity again,
A2 = %pi*d2*L;// [m^2]
A3 = %pi*(d2^2-d1^2)/4;// [m^2]
F32 = A2*F23/A3;
// we observe that F11 = F33 = F44 = 0 and for surface 3 F31+F32+F34 = 1.0
// so, if F31 can be determined, we can calculate the desired quantity F34. for surface 1 F12+F13+F14 = 1.0
// and from symmetry F13 = F14 so that
F13 = (1-F12)/2;
F14 = F13;
// using reciprocity gives
A1 = %pi*d1*L;// [square meter]
F31 = (A1/A3)*F13;
// then
F34 = 1-F31-F32;
printf("shape factor between the open ends of the cylinder is %f ",F34);
```

Scilab code Exa 8.4 shape factor algebra for truncated cone

```
clear;
clc;
printf("\t\t\tExample Number 8.4\n\n\n");
// shape-factor algebra for truncated cone
// Example 8.4 (page no.-396)
// solution

d1 = 0.1;// [m] diameter of top of cone
d2 = 0.2;// [m] diameter of bottom of cone
```

```
L = 0.1;// [m] height of cone
// we employ figure 8-16(page no.-390) for solution
   of this problem and take the nomenclature as
   shown, designating the top as surface 2,
// the bottom as surface 1, and the side as surface
   3. thus the desired quantities are F23 and F33.
   we have
Z = L/(d2/2);
Y = (d1/2)/L;
// thus from figure 8-16(page no.-390)
F12 = 0.12;
// from reciprcity(equatin 8-18)
A1 = %pi*d2^(2)/4;// [square meter]
A2 = %pi*d1^(2)/4;// [square meter]
F21 = A1*F12/A2;
//and
F22 = 0;
// so that
F23 = 1-F21;
// for surface 3 F31+F32+F33 = 1,  so we must find
   F31 and F32 in order to evaluate F33. since F11 =
    0 we have
F13 = 1-F12;
// and from reciprocity
A3 = %pi*((d1+d2)/2)*[(d1/2-d2/2)^(2)+L^(2)]^(1/2);
   // [square meter]
// so from above equation
F31 = A1*F13/A3;
// a similar procedure is applies with surface 2 so
   that
F32 = A2*F23/A3;
// finally from above equation
F33 = 1-F32-F31;
printf("shape factor between the top surface and the
    side is %f ",F23);
printf("\nshape factor between the side and itself
   is %f ",F33);
```

Scilab code Exa 8.5 shape factor algebra for cylindrical reflactor

```
clear;
clc;
printf("\t\t\tExample Number 8.5\n\n\n");
// shape-factor algebra for cylindrical reflactor
// Example 8.5 (page no.-397-398)
// solution

d = 0.6;// [m] diameter of long half-circular
    cylinder
L = 0.2;// [m] length of square rod
// we have given figure example 8-5(page no.-397)
    for solution of this problem and take the
    nomenclature as shown,
// from symmetry we have
F21 = 0.5;
F23 = F21;
// in general, F11+F12+F13 = 1. to aid in the
    analysis we create the fictious surface 4 shown
    in figure example 8-5 as dashed line.
// for this surface
F41 = 1.0;
// now, all radiation leaving surface 1 will arrive
    either at 2 or at 3. likewise,this radiation will
     arrive at the imaginary surface 4, so that F41 =
     F12+F13 say eqn a
// from reciprocity
A1 = %pi*d/2;// [square meter]
A4 = L+2*sqrt(0.1^(2)+L^(2));// [square meter]
A2 = 4*L;// [square meter]
// so that
F14 = A4*F41/A1;// say eqn b
// we also have from reciprocity
```

```
F12 = A2*F21/A1;// say eqn c
// combining a,b,c, gives
F13 = F14-F12;
// finally
F11 = 1-F12-F13;
printf("value of F12 is %f ",F12);
printf("\nvalue of F13 is %f ",F13);
printf("\nvalue of F11 is %f ",F11);
```

Scilab code Exa 8.6 hot plates enclosed by a room

```
clear;
clc;
printf("\t\t\tExample Number 8.6\n\n\n");
// hot plates enclosed by a room
// Example 8.6 (page no.-402-404)
// solution

w = 0.5;// [m] width of plate
L = 1;// [m] length of plate
t = 0.5;// [m] seperation between two plates
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
// this is a three-body problem, the two plates and the room, so the radiation network is shown in figure(8-27) page no.-401.
// from the data of the problem
T1 = 1000+273;// [K] temperature of first plate
T2 = 500+273;// [K] temperature of second plate
T3 = 27+273;// [K] temperature of walls of plates
A1 = w*L;// [square meter] area of plate
A2 = A1;// [square meter] area of plate
E1 = 0.2;// emissivity of plate 1
E2 = 0.5;// emissivity of plate 2
// because the area of the room A3 is very large, the resistance (1-E3)/(E3*A3) may be taken as
```

```
   zero and we obtain Eb3 = J3.
// the shape factor F12 was given in example 8-2:
F12 = 0.285;
F21 = F12;
F13 = 1-F12;
F23 = 1-F21;
// the resistance in the network are calculated as
R1 = (1-E1)/(E1*A1);
R2 = (1-E2)/(E2*A2);
R3 = 1/(A1*F12);
R4 = 1/(A1*F13);
R5 = 1/(A2*F23);
// taking the resistance (1-E3)/(E3*A3) as zero, we have the network (as shown in figure example 8-6(page no.-403)).
// to calculate the heat flows at each surface we must determine the radiosities J1 and J2. the network is solved by setting the sum of the heat currents entering nodes J1 and J2 to zero

// node J1:
// (Eb1-J1)/R1+(J2-J1)/R3+(Eb3-J1)/R4 = 0                                  (a)

// node J2:
// (J1-J2)/R3+(Eb3-J2)/R5+(Eb2-J2)/R2 = 0                                  (b)

// now
Eb1 = sigma*T1^(4);// [W/square meter]
Eb2 = sigma*T2^(4);// [W/square meter]
Eb3 = sigma*T3^(4);// [W/square meter]
J3 = Eb3;// [W/square meter]
// inserting the values of Eb1,Eb2, and Eb3 into equations (a) and (b), we have two equations and two unknowns J1 and J2 that may be solved simultaneously to give
// on simplifying we get J1 = (J2-R3*[(Eb3-J2)/R5+(
```

```
Eb2-J2)/R2])
// putting this value in equation (a) and solve for J2
deff('[y] = f3(J2)','y = (Eb1-(J2-R3*[(Eb3-J2)/R5+(Eb2-J2)/R2]))/R1+(J2-(J2-R3*[(Eb3-J2)/R5+(Eb2-J2)/R2]))/R3+(Eb3-(J2-R3*[(Eb3-J2)/R5+(Eb2-J2)/R2]))/R4');
J2 = fsolve(1,f3);// [W/square meter]
J1 = (J2-R3*[(Eb3-J2)/R5+(Eb2-J2)/R2]);// [W/square meter]
// the total heat lost by plate 1 is
q1 = (Eb1-J1)/[(1-E1)/(E1*A1)];// [W]
// and the heat lost by plate 2 is
q2 = (Eb2-J2)/[(1-E2)/(E2*A2)];// [W]
// the total heat received by the room is
q3 = [(J1-J3)/(1/(A1*F13))]+[(J2-J3)/(1/(A2*F23))];// [W]
printf("the net heat transfer for plate 1 is %f kW",q1/1000)
printf("\n\n the net heat transfer for plate 2 is %f kW",q2/1000)
printf("\n\n the net heat transfer to the room  is %f kW",q3/1000)
```

Scilab code Exa 8.7 surface in radiant balance

```
clear;
clc;
printf("\t\t\tExample Number 8.7\n\n\n");
// surface in radiant balance
// Example 8.7 (page no.-404-405)
// solution

w = 0.5;// [m] width of plate
L = 0.5;// [m] length of plate
```

```
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
// from the data of the problem
T1 = 1000;// [K] temperature of first surface
T2 = 27+273;// [K] temperature of room
A1 = w*L;// [square meter] area of rectangle
A2 = A1;// [square meter] area of rectangle
E1 = 0.6;// emissivity of surface 1
// although this problems involves two surfaces which exchange heat and one which is insulated or re-radiating, equation (8-41) may not be used for the calculation because one of the heat-exchanging surfaces(the room) is not convex. The radiation network is shown in figure example 8-7(page no.-404) where surface 3 is the room and surface 2 is the insulated surface. note that J3 = Eb3 because the room is large and (1-E3)/(E3*A3) approaches zero.Because surface 2 is insulated it has zero heat transfer and J2 = Eb2. J2 "floats" in the network and is determined from the overall radiant balance.
// from figure 8-14(page no.-387) the shape factors are
F12 = 0.2;
F21 = F12;
// because
F11 = 0;
F22 = 0;
F13 = 1-F12;
F23 = F13;
// the resistances are
R1 = (1-E1)/(E1*A1);
R2 = 1/(A1*F13);
R3 = 1/(A2*F23);
R4 = 1/(A1*F12);
// we also have
Eb1 = sigma*T1^(4);// [W/square meter]
Eb3 = sigma*T2^(4);// [W/square meter]
J3 = Eb3;// [W/square meter]
```

```
// the overall circuit is a series parallel arrangement and the heat transfer is
R_equiv = R1+(1/[(1/R2)+1/(R3+R4)]);
q = (Eb1-Eb3)/R_equiv;// [W]
// this heat transfer can also be written as q = (Eb1-J1)/((1-E1)/(E1*A1))
// inserting the values
J1 = Eb1-q*((1-E1)/(E1*A1));// [W/square meter]
// the value of J2 is determined from proportioning the resistances between J1 and J3, so that
// (J1-J2)/R4 = (J1-J3)/(R4+R2)
J2 = J1-((J1-J3)/(R4+R2))*R4;// [W/square meter]
Eb2 = J2;// [W/square meter]
// finally , we obtain the temperature of the insulated surface as
T2 = (Eb2/sigma)^(1/4);// [K]
printf("temperature of the insulated surface is %f K ",T2);
printf("\n\n heat lost by the surface at 1000K is %f kW",q/1000);
```

Scilab code Exa 8.8 open hemisphere in large room

```
clear;
clc;
printf("\t\t\tExample Number 8.8\n\n\n");
// open hemisphere in large room
// Example 8.8 (page no.-406-408)
// solution

d = 0.3;// [m] diameter of hemisphere
T1 = 500+273;// [degree celsius] temperature of hemisphere
T2 = 30+273;// [degree celsius] temperature of enclosure
```

```
E = 0.4;// surface emissivity of hemisphere
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
    constant
// the object is completely surrounded by a large
    enclosure but the inside surface of the sphere is
     not convex.
// in the given figure example 8-8(page no.-407) we
    take the inside of the sphere as surface 1 and
    the enclosure as surface 2.
// we also create an imaginary surface 3 covering
    the opening.
// then the heat transfer is given by
Eb1 = sigma*T1^(4);// [W/square meter]
Eb2 = sigma*T2^(4);// [W/square meter]
A1 = 2*%pi*(d/2)^(2);// [square meter] area of
    surface 1
// calculating the surface resistance
R1 = (1-E)/(E*A1);
// since A2 tends to 0 so R2 also tends to 0
R2 = 0;
// now at this point we recognize that all of the
    radiation leaving surface 1 which will eventually
     arrive at enclosure 2 will also hit the
    imaginary  surface 3(F12 = F13). we also
    recognize that A1*F13 = A3*F31. but
F31 = 1.0;
A3 = %pi*(d/2)^(2);// [square meter]
F13 = (A3/A1)*F31;
F12 = F13;
// then calculating space resistance
R3 = 1/(A1*F12);
// we can claculate heat transfer by inserting the
    quantities in equation (8-40):
q = (Eb1-Eb2)/(R1+R2+R3);// [W]
printf("net radiant exchange is %f W",q);
```

Scilab code Exa 8.9 effective emissivity of finned surface

```
clear;
clc;
printf("\t\t\tExample Number 8.9\n\n\n");
// effective emissivity of finned surface
// Example 8.9 (page no.-409-410)
// solution

// for unit depth in the z-dimension we have
A1 = 10;// [square meter]
A2 = 5;// [square meter]
A3 = 60;// [square meter]
// the apparent emissivity of the open cavity area A1 is given by equation(8-47) as
// Ea1 = E*A3/[A1+E*(A3-A1)]
// for constant surface emissivity the emitted energy from the total area A1+A2 is
// e1 = Ea1*A1+E*A2*Eb
// and the energy emitted per unit area for that total area is
// e_t = [(Ea1*A1+E*A2)/(A1+A2)]*Eb
// the coefficient of Eb is the effective emissivity, E_eff of the combination of the surface and open cavity. inserting
// above equations gives the following values

// for E = 0.2

E = 0.2;
Ea1 = E*A3/[A1+E*(A3-A1)];
E_eff = [(Ea1*A1+E*A2)/(A1+A2)];
printf("For emissivity of 0.2 the value of effective emissivity is %f ",E_eff);
```

```

// for E = 0.5

E = 0.5;
Ea1 = E*A3/[A1+E*(A3-A1)];
E_eff = [(Ea1*A1+E*A2)/(A1+A2)];
printf("\n\n For emissivity of 0.5 the value of effective emissivity is %f ",E_eff);

// for E = 0.8

E = 0.8;
Ea1 = E*A3/[A1+E*(A3-A1)];
E_eff = [(Ea1*A1+E*A2)/(A1+A2)];
printf("\n\n For emissivity of 0.8 the value of effective emissivity is %f ",E_eff);
```

Scilab code Exa 8.10 heat transfer reduction with parallel plate shield

```
clear;
clc;
printf("\t\t\tExample Number 8.10\n\n\n");
// heat transfer reduction with parallel plate shield
// Example 8.10 (page no.-413)
// solution

E1 = 0.3;// emissivity of first plane
E2 = 0.8;// emissivity of second plane
E3 = 0.04;// emissivity of shield
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
// the heat transfer without the shield is given by
// q_by_A = sigma*(T1^4-T2^4)/((1/E1)+(1/E2)-1) = 0.279*sigma*(T1^4-T2^4)
// where T1 is temperature of first plane and T2 is
```

```
temperature of second plane
// the radiation network for the problem with the
   shield in place is shown in figure (8-32) (page
   no.-410).
// the resistances are
R1 = (1-E1)/E1;
R2 = (1-E2)/E2;
R3 = (1-E3)/E3;
// the total resistance with the shield is
R = R1+R2+R3;
// and the heat transfer is
// q_by_A = sigma*(T1^4-T2^4)/R = 0.01902*sigma*(T1
   ^4-T2^4)
printf("so the heat tranfer is reduced by %f percent
   ",((0.279-0.01902)/0.279)*100);
```

Scilab code Exa 8.11 open cylindrical shield in large room

```
clear;
clc;
printf("\t\t\tExample Number 8.11\n\n\n");
// open cylindrical shield in large room
// Example 8.11 (page no.-413-415)
// solution

// two concentric cylinders of example(8.3) have
T1 = 1000;// [K]
E1 = 0.8;
E2 = 0.2;
T3 = 300;// [K] room temperature
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
// please refer to figure example 8-11(page no.-413)
    for radiation network
// the room is designed as surface 3 and J3 = Eb3,
   because the room is very large,(i.e. its surface
```

```
    is very small)
// in this problem we must consider the inside and outside of surface 2 and thus have subscripts i and o to designate the respective quantities.
// the shape factor can be obtained from example 8-3 as
F12 = 0.8253;
F13 = 0.1747;
F23i = 0.2588;
F23o = 1.0;
// also
A1 = %pi*0.1*0.2;// [square meter] area of first cylinder
A2 = %pi*0.2*0.2;// [square meter] area of second cylinder
Eb1 = sigma*T1^4;// [W/square meter]
Eb3 = sigma*T3^4;// [W/square meter]
// the resistances may be calculated as
R1 = (1-E1)/(E1*A1);
R2 = (1-E2)/(E2*A2);
R3 = 1/(A1*F12);
R4 = 1/(A2*F23i);
R5 = 1/(A2*F23o);
R6 = 1/(A1*F13);
// the network could be solved as a series-parallel circuit to obtain the heat transfer, butwe will need the radiosities anyway, so we setup three nodal equations to solve for J1,J2i, and J2o.
// we sum the currents into each node and set them equal to zero:

// node J1: (Eb1-J1)/R1+(Eb3-J3)/R6+(J2i-J1)/R3 = 0
// node J2i: (J1-J2i)/R3+(Eb3-J2i)/R4+(J2o-J2i)/(2*R2) = 0
// node J2o: (Eb3-J2o)/R5+(J2i-J2o)/(2*R2) = 0
// these equations can be solved by matrix method and the solution is
J1 = 49732;// [W/square meter]
```

```
J2i = 26444;// [W/square meter]
J2o = 3346;// [W/square meter]
// the heat transfer is then calculated from
q = (Eb1-J1)/((1-E1)/(E1*A1));// [W]
// from the network we see that
Eb2 = (J2i+J2o)/2;// [W/square meter]
// and
T2 = (Eb2/sigma)^(1/4);// [K]
// if the outer cylinder had not been in place
   acting as a "shield" the heat loss from cylinder
   1 could have been calculated from equation(8-43a)
    as
q1 = E1*A1*(Eb1-Eb3);// [W]
printf("temperature of the outer cylinder is %f K",
   T2);
printf("\n\ntotal heat lost by inner cylinder is %f
   W",q1);
```

Scilab code Exa 8.12 network for gas radiation between parallel plates

```
clear;
clc;
printf("\t\t\tExample Number 8.12\n\n\n");
// network for gas radiation between parallel plates
// Example 8.12 (page no.-422-423)
// solution

T1 = 800;// [K] temperature of first plate
E1 = 0.3;// emissivity
T2 = 400;// [K] temperature of second plate
E2 = 0.7;// emissivity
Eg = 0.2;// emissivity of gray gas
tg = 0.8;// transmissivity of gray gas
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
// the network shown in figure 8-39(page no.-419)
```

```
   applies to this problem. all the shape factors
   are unity for large planes and the various
   resistors can be computed on a unit area basis as
F12 = 1;
F1g = 1;
F2g = F1g;
R1 = (1-E1)/E1;
R2 = (1-E2)/E2;
R3 = 1/(F12*(1-Eg));
R4 = 1/(F1g*Eg);
R5 = 1/(F2g*Eg);
Eb1 = sigma*T1^(4);// [W/square meter]
Eb2 = sigma*T2^(4);// [W/square meter]
// the equivalent resistance of the center "triangle" is
R = 1/[(1/R3)+(1/(R4+R5))];
// the total heat transfer is then
q_by_A = (Eb1-Eb2)/(R1+R2+R);// [W/square meter]
// if there were no gas present the heat transfer would be given by equation (8-42):
q_by_A1 = (Eb1-Eb2)/[(1/E1)+(1/E2)-1];// [W/square meter]
// the radiosities may be computed from q_by_A = (Eb1-J1)*(E1/(1-E1)) = (J2-Eb2)*(E2/(1-E2))
J1 = Eb1-q_by_A*((1-E1)/E1);// [W/square meter]
J2 = Eb2+q_by_A*((1-E2)/E2);// [W/square meter]
// for the network Ebg is just the mean of these values
Ebg = (J1+J2)/2;// [W/square meter]
// so that the temperature of the gas is
Tg = (Ebg/sigma)^(1/4);// [K]
printf("the heat-transfer rate between the two planes is %f W/square meter",q_by_A);
printf("\n\n the temperature of the gas is %f K",Tg);
printf("\n\n the ratio of heat-transfer with presence of gas to without presence of gas is %f",q_by_A/q_by_A1);
```

Scilab code Exa 8.13 cavity with transparent cover

```
clear;
clc;
printf("\t\t\tExample Number 8.13\n\n\n");
// cavity with transparent cover
// Example 8.13 (page no.-433-434)
// solution

E1 = 0.5;// emissivity of rectangular cavity
t2 = 0.5;// transmissivity
rho2 = 0.1;// reflectivity
E2 = 0.4;// emissivity
// from example 8-9 we have
// per unit depth in the z direction we have
A1 = 25+25+10;
A2 = 10;
// we may evaluate K from equation(8-96a)
K = E1/(t2+(E2/2));
// the value of Ea is then computed from equation
    (8-96) as
Ea = (t2+(E2/2))*K/[(A2/A1)*(1-E1)+K];
printf("apparent emissivity of covered opening is %f
    ",Ea);
// if there were no cover present, the value of Ea
    would be given by equation (8-47) as
Ea1 = E1*A1/[A2+E1*(A1-A2)];
printf("\n\n if there were no cover present, the
    value of Ea(apparent emissivity) would be %f",Ea1
    );
```

Scilab code Exa 8.14 Transmitting and reflecting system for furnace opening

```
clear;
clc;
printf("\t\t\tExample Number 8.14\n\n\n");
// Transmitting and reflecting system for furnace
   opening
// Example 8.14 (page no.-434-435)
// solution

T1 = 1000+273;// [K] temperature of furnace
lambda = 4.0;// [micro meter]
//for  0 < lambda < 4 micro meter
t1 = 0.9;
E1 = 0.1;
rho1 = 0;
//for  4 micro meter < lambda < infinity
t2 = 0;
E2 = 0.8;
rho2 = 0.2;
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
T3 = 30+273;// [K] room temperature
// the diagram of this problem is shown in figure
   example 8-14(page no.-434). because the room is
   large it may be treated as a blackbody also.
// we shall analyze the problem by calculating the
   heat transfer for each wavelength band and then
   adding them together to obtain the total. the
   network for each band is a modification of figure
    8-57(page no.-430), as shown here for black
   furnace and room. we shall make the calculation
   for unit area; then
A1 = 1.0;// [square meter]
A2 = 1.0;// [square meter]
A3 = 1.0;// [square meter]
F12 = 1.0;
F13 = 1.0;
```

```
F32 = 1.0;
// the total emissive powers are
Eb1 = sigma*T1^(4);// [W/square meter]
Eb3 = sigma*T3^(4);// [W/square meter]
// to determine the fraction of radiation in each
   wavelength band, we calculate
lamba_into_T1 = lambda*T1;// [micro meter K]
lamba_into_T3 = lambda*T3;// [micro meter K]
// consulting table 8-1(page no.-379-380), we find
Eb1_0_to_4 = 0.6450*Eb1;// [W/square meter]
Eb3_0_to_4 = 0.00235*Eb3;// [W/square meter]
Eb1_4_to_inf = (1-0.6450)*Eb1;// [W/square meter]
Eb3_4_to_inf = (1-0.00235)*Eb3;// [W/square meter]
// we now apply these numbers to the network for the
    two wavelengths bands, with unit areas.

// 0 < lambda < 4 micro meter band:
R1 = 1/(F13*t1);
R2 = 1/(F32*(1-t1));
R3 = 1/(F12*(1-t1));
R4 = rho1/(E1*(1-t1));
// the net heat transfer from the network is then
R_equiv_1 = 1/(1/R1+1/(R2+R3+R4));
q1 = (Eb1_0_to_4-Eb3_0_to_4)/R_equiv_1;// [W/square
   meter]

// 4 micro meter < lambda < infinity band:
R2 = 1/(F32*(1-t2));
R3 = 1/(F12*(1-t2));
R4 = rho2/(E2*(1-t2));
// the net heat transfer from the network is then
// R1 is infinity
R_equiv_2 = R2+R3+R4*2;
q2 = (Eb1_4_to_inf-Eb3_4_to_inf)/R_equiv_2;// [W/
   square meter]

// the total heat loss is then
q_total = q1+q2;// [W/square meter]
```

```
// with no windows at all, the heat transfer would have been the difference in blackbody emissive powers,
Q = Eb1-Eb3;// [W/square meter]
printf("radiation lost through the quartz window to a room temperature of 30 degree celsius is %f W/square meter",q_total);
printf("\n\n with no windows at all, the heat transfer would be %e W/square meter",Q);
```

Scilab code Exa 8.15 numerical solution for enclosure

```
clear;
clc;
printf("\t\t\tExample Number 8.15\n\n\n");
// numerical solution for enclosure
// Example 8.15 (page no.-440)
// solution

// the geometry of example 8-5 is used
d = 0.6;// [m] diameter of long half-circular cylinder
L = 0.2;// [m] length of square rod
E2 = 0.5;
T2 = 1000;// [K] temperature of body 2
T3 = 300;// [K] temperature of body 3
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
// for unit length we have:
Eb2 = sigma*T2^(4);// [W/square meter]
Eb3 = sigma*T3^(4);
A1 = 4*L;// [square meter]
A2 = %pi*d/2;// [square meter/meter]
// we will use the numerical formulation. we find from example 8-5, using the nomenclature of the figure
```

```
F11 = 0.314;
F12 = 0.425;
F13 = 0.261;
F21 = 0.5;
F22 = 0;
F23 = 0.5;
// F31, F32 tends to zero so
F33 = 1;
// we now write the equations.
// surface 1 is insulated so we use equation(8-107a):
// J1*(1-F11)-F12*J2-F13*J3 = 0
// surface 2 is constant temperature so we use equation (8-106a):
// J2*(1-F22*(1-E2))-(1-E2)*[F21*J1+F23*J3] = E2*Eb2
// because surface 3 is so large
J3 = Eb3;// [W/square meter]
// rearranging the equation gives
// J1*(1-F11)-J2*F12 = F13*J3
// J1*(-1)*(1-E2)*F21+J2*(1-F22*(1-E2)) = E2*Eb2+(1-E2)*(F23*J3)
// solving the above two equations using matrix
X = [(1-F11) -F12;(-1)*(1-E2)*F21 (1-F22*(1-E2))];
Y = [F13*J3;E2*Eb2+(1-E2)*(F23*J3)];
J = X^(-1)*Y;
J1 = J(1);// [W/square meter]
J2 = J(2);// [W/square meter]
// the heat transfer is thus
q = (Eb2-J2)/((1-E2)/(E2*A1));// [W/m] length
// because surface 1 is insulated
Eb1 = J1;// [W/square meter]
// we could calculate the temperature as
T1 = (Eb1/sigma)^(1/4);// [K]
printf("heat lost to the large room per unit length of surface 2 is %f W/m",q);
printf("\n\n temperature of the insulated surface is %f K",T1);
```

Scilab code Exa 8.16 numerical solution for parallel plates

```
clear;
clc;
printf("\t\t\tExample Number 8.16\n\n\n");
// numerical solution for parallel plates
// Example 8.16 (page no.-440-443)
// solution

T1 = 1000;// [K]
T2 = 400;// [K]
E1 = 0.8;//
E2 = 0.5;//
// consulting figure 8-12, we obtain
F12 = 0.2;
F21 = 0.2;
F11 = 0;
F22 = 0;
F13 = 0.8;
F23 = 0.8;
A1 = 1;// [square meter]
A2 = 1;// [square meter]
// surface 3 is the surrounding or insulated surface. For part A(the plates are surrounded by a large room at 300K)
T3 = 300;// [K]
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
Eb1 = sigma*T1^(4);// [W/square meter]
Eb2 = sigma*T2^(4);// [W/square meter]
Eb3 = sigma*T3^(4);// [W/square meter]
// because A3 tends to infinity, F31 and F32 must approach zero since A1*F13 = A3*F31 and A2*F23 = A3*F32. the nodal equations are written in the form of equations (8-107):
```

```
// surface 1        J1-(1-E1)*(F11*J1+F12*J2+F13*J3) = E1*Eb1
// surface 2        J2-(1-E2)*(F21*J1+F22*J2+F23*J3) = E2*Eb2
// surface 3        J3-(1-E3)*(F31*J1+F32*J2+F33*J3) = E3*Eb3

// because F31 and F32 approach zero, F33 must be 1.0.
F33 = 1;
// inserting the various numerical values for the various terms and solving the third equation we get
// the third equation as:  J3*E3 = E3*Eb3  so we get the value of J3  as
J3 = Eb3;// [W/square meter]
// finally the equations are written in compact form after getting the value of J3 we solve for J2 and J1 by matrix method
Z = [1-(1-E1)*F11 -(1-E1)*F12;-(1-E2)*F21 1-(1-E2)*F22];
C = [E1*Eb1+(1-E1)*F13*J3;E2*Eb2+(1-E2)*F23*J3];
J = Z^(-1)*C;
J1 = J(1);// [W/square meter]
J2 = J(2);// [W/square meter]
// the heat transfers are obtained from equation (8-104):
q1 = A1*E1*(Eb1-J1)/(1-E1);// [W]
q2 = A2*E2*(Eb2-J2)/(1-E2);// [W]
// the net heat absorbed by the room is algebric sum of q1 and q2
q3_absorbed = q1+q2;// [W]
printf("\t\t CASE(A)");
printf("\n\n the heat transfers are \n\n\t\t q1 = %f kW",q1/1000);
printf("\n\t\t q1 = %f kW",q2/1000)
printf("\n\n the net heat absorbed by the room in part (a) is %f kW",q3_absorbed/1000);
```

```
// for part(b), A3 for the enclosing wall is 4.0 square meter

A3 = 4;// [square meter]
// and we set
J3 = Eb3;// [W/square meter], because surface 3 is insulated.
// from reciprocity we have
F31 = A1*F13/A3;
F32 = A2*F23/A3;
// then, we have
F33 = 1-F31-F32;
// the set of equations are same with J3 = Eb3
// surface 1        J1-(1-E1)*(F11*J1+F12*J2+F13*J3) = E1*Eb1
// surface 2        J2-(1-E2)*(F21*J1+F22*J2+F23*J3) = E2*Eb2
// surface 3        J3-(1-E3)*(F31*J1+F32*J2+F33*J3) = E3*J3
// the third equation of set can be written as
// J3(1-E3)-(1-E3)*(F31*J1+F32*J2+F33*J3) = 0
// so that (1-E3) term drops out, and we obtain three equation in three variable which can be solved by matrix method
Z = [1-(1-E1)*F11 -(1-E1)*F12 -(1-E1)*F13;-(1-E2)*F21 1-(1-E2)*F22 -(1-E2)*F23;-F31 -F32 1-F33];
C = [E1*Eb1;E2*Eb2;0];
J = Z^(-1)*C;
J1n = J(1);// [W/square meter]
J2n = J(2);// [W/square meter]
J3n = J(3);// [W/square meter]
// the heat transfers are
q1n = A1*E1*(Eb1-J1n)/(1-E1);// [W]
q2n = A2*E2*(Eb2-J2n)/(1-E2);// [W]
// of course these heat transfers should be equal in magnitude with opposite sign because the insulated wall exchanges no heat.
```

```
// the temperature of the insulated wall is obtained
   from
T3 = (J3n/sigma)^(1/4);// [degree celsius]
printf("\n\n \t\tCASE(B)");
printf("\n\n the heat transfers are \n\n\t\t q1 = %f
   kW",q1n/1000);
printf("\n\t\t q2 = %f kW",q2n/1000);
printf("\n\n the temperature of the insulated wall
   is %f K",T3);
```

Scilab code Exa 8.17 radiation from a hole with variable radiosity

```
clear;
clc;
printf("\t\t\tExample Number 8.17\n\n\n");
// radiation from a hole with variable radiosity
// Example 8.17 (page no.-443-446)
// solution

T1 = 1273;// [K]
T5 = 293;// [K]
E1 = 0.6;
// all the shape factors can be obtained with the
   aid of figure 8-13(page no.-387) and the
   imaginary disk surfaces 6 and 7. we have
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
Eb1 = sigma*T1^(4);// [W/square meter]
Eb2 = Eb1;// [W/square meter]
Eb3 = Eb2;// [W/square meter]
Eb4 = Eb3;// [W/square meter]
Eb5 = sigma*T5^(4);// [W/square meter]
E2 = E1;
E3 = E2;
E4 = E3;
E5 = 1.0;
```

```
r = 0.01;// [m]
A1 = %pi*r^(2);// [square m]
A5 = A1;// [square m]
A6 = A1;// [square m]
A7 = A1;// [square m]
A2 = %pi*2*r*0.01;// [square m]
A3 = A2;// [square m]
A4 = A2;// [square m]
F11 = 0;
F55 = 0;
F16 = 0.37;
F17 = 0.175;
F15 = 0.1;
F12 = 1-F16;
F54 = F12;
F13 = F16-F17;
F53 = F13;
F14 = F17-F15;
F52 = F14;
F21 = F16*A1/A2;
F26 = F21;
F45 = F21;
F36 = F45;
F37 = F36;
F22 = 1-F21-F26;
F33 = F22;
F44 = F22;
F31 = F13*A1/A3;
F32 = F36-F31;
F34 = F32;
F43 = F34;
F23 = F34;
F27 = F26-F23;
F46 = F27;
F41 = F14*A1/A4;
F25 = F41;
F42 = F46-F41;
F24 = F42;
```

```
// the equations for the radiosities are now written
   in the form of equation 8-106, noting that
F11 = 0;
J5 = Eb5;// [W/square meter]
//  J1 = (1-E1)*(F12*J2+F13*J3+F14*J4+F15*Eb5)+E1*
   Eb1
//  J2 = [(1-E2)*(F21*J1+F23*J3+F24*J4+F25*Eb5)+E2*
   Eb2]/(1-F22*(1-E2))
//  J3 = [(1-E3)*(F31*J1+F32*J2+F34*J4+F35*Eb5)+E3*
   Eb3]/(1-F33*(1-E3))
//  J4 = [(1-E2)*(F41*J1+F42*J2+F43*J3+F45*Eb5)+E4*
   Eb4]/(1-F44*(1-E4))
// we have 4 equations with 4 variables which can be
   solved by matrix method
Z = [1 -(1-E1)*F12 -(1-E1)*F13 -(1-E1)*F14;-F21*(1-
   E2)/(1-F22*(1-E2)) 1 -F23*(1-E2)/(1-F22*(1-E2)) -
   F24*(1-E2)/(1-F22*(1-E2));-F31*(1-E3)/(1-F33*(1-
   E3)) -F32*(1-E3)/(1-F33*(1-E3)) 1 -F34*(1-E3)/(1-
   F33*(1-E3));-F41*(1-E4)/(1-F44*(1-E4)) -F42*(1-E4
   )/(1-F44*(1-E4)) -F43*(1-E4)/(1-F44*(1-E4)) 1];
C = [E1*Eb1+(1-E1)*F15*Eb5;(E2*Eb2+F25*Eb5*(1-E2))
   /(1-F22*(1-E2));104859;(E4*Eb4+F45*Eb5*(1-E4))
   /(1-F44*(1-E4))];
J = Z^(-1)*C;
J1 = J(1);// [W/square meter]
J2 = J(2);// [W/square meter]
J3 = J(3);// [W/square meter]
J4 = J(4);// [W/square meter]
// the heat transfer can be calculated from equation
   (8-104):
q1 = E1*A1*(Eb1-J1)/(1-E1);// [W]
q2 = E2*A2*(Eb2-J2)/(1-E2);// [W]
q3 = E3*A3*(Eb3-J3)/(1-E3);// [W]
q4 = E4*A4*(Eb4-J4)/(1-E4);// [W]
// THE TOTAL HEAT TRANSFER
q = q1+q2+q3+q4;// [W]
printf("the heat transfer rate is %f W",q);
// It is of interest to compare this heat transfer
```

```
   with the value we would obtain by assuming
   uniform radiosity on the hot surface. we would
   then have a two-body problem with
A1 = %pi+3*(2*%pi);// [square cm]
A5 = %pi*10^(-4);// [square cm]
F51 = 1.0;
E1 = 0.6;
E5 = 1.0;
// the heat transfer is then calculated from
   equation(8-43), with appropriate change of
   nomenclature:
q_new = (Eb1-Eb5)*A5/((1/E5)+(A5/A1)*((1/E1)-1));//
   [w]
printf("\n\nthus the assumption of uniform radiosity
    gives a heat transfer that is %f percent below
   the value obtained by breaking the hot surface
   into four parts for the calculations",(q-q_new)
   *100/q);
// let us now consider the case where surface 1 is
   still radiating at 1000 degree celsius
E = 0.6;
// the nodal equations for J1 is the same as before
   but now the equations for J2, J3, and J4 must be
   written in the form of equation(8-107). when that
    is done and the numerical values are inserted,
   we obtain
//  J1 = 0.252*J2+0.078*J3+0.03*J4+89341
//  J2 = 0.5*J1+0.3452*J3+0.09524*J4+24.869
//  J3 = 0.1548*J1+0.3452*J2+0.3452*J4+64.66
//  J4 = 0.05952*J1+0.0952*J2+0.3452*J3+208.9
// when these equations are solved, we obtain
J1 = 1.1532*10^(5);// [W/square meter]
J2 = 0.81019*10^(5);// [W/square meter]
J3 = 0.57885*10^(5);// [W/square meter]
J4 = 0.34767*10^(5);// [W/square meter]
// the heat transfer at surface 1 is
A1 = %pi*10^(-4);// [square cm]
A5 = %pi*10^(-4);// [square cm]
```

```
A2 = %pi*10^(-4);// [square cm]
q1 = (E1*A1)*(Eb1-J1)/(1-E1);// [W]
// the temperatures of the insulated surface
   elements are obtained from
T2 = 820;// [degree celsius]
T3 = 732;// [degree celsius]
T4 = 612;// [degree celsius]
// it is of interest to compare the heat transfer
   calculated above with that obtained by assuming
   surfaces 2,3 and 4 uniform in temperature and
   radiosity. equation(8-41) applies for this case:
q2 = A1*(Eb1-Eb5)/[((A1+A2+2*A1*F15)/(A5-A1*F15^2))
   +(1/E1-1)+(A1/A5)*(1/E5-1)];// [w]
```

Scilab code Exa 8.18 heater with constant heat flux and surrounding shields

```
clear;
clc;
printf("\t\t\tExample Number 8.18\n\n\n");
// heater with constant heat flux and surrounding
   shields
// Example 8.18 (page no.-446-449)
// solution

sigma = 5.669*10^(-8);// [W/square meter K^(4)]
T6 = 293;// [K] temperature of room
E1 = 0.8;
E2 = 0.4;
E3 = 0.4;
E4 = 0.4;
E5 = 0.4;
// in reality, surfaces 2,3,4, and 5 have two
   surfaces each; an inside and an outside surface.
   we thus have nine surfaces plus the room, so a 10
    body problem is involved. of course, from
```

```
    symmetry we can see that T2 = T4 and T3 = T5.
// we designate the large room as surface 6 and it
    behaves as E6 = 1.0.
// the shape factors of the inside surfaces are
    obtained from figure 8-12 and 8-14:
F16 = 0.285;
F61 = F16;
F13 = 0.24;
F15 = 0.24;
F31 = 0.24;
F51 = 0.24;
F12 = 0.115;
F14 = 0.115;
F24 = 0.068;
F42 = 0.068;
F35 = 0.285;
F53 = 0.285;
F32 = 0.115;
F52 = 0.115;
F34 = 0.115;
F25 = 0.23;
F23 = 0.23;
F45 = 0.23;
F43 = 0.23;
F21 = 0.23;
F41 = 0.23;
F26 = 0.23;
F46 = 0.23;
F11 = 0;
F22 = 0;
F33 = 0;
F44 = 0;
F55 = 0;
// for the outside surfaces,
F_26 = 1;
F_36 = 1;
F_46 = 1;
F_56 = 1;
```

```
// Where the underscore indicate the outside surfaces.
// for the room
Eb6 = sigma*T6^(4);// [W/square meter]
// for surface 1 with constant heat flux, we use equation (8-108a) and write
// J1-(F12*J2+F13*J3+F14*J4+F15*J5+F16*J6) = 1.0*10^(5)                GIVEN                (a)
// because of the radiant balance condition we have
// (J2-Eb2)*E2*A2/(1-E2) = (Eb2-J_2)*E2*A2/(1-E2)
// and      Eb2 = (J2+J_2)/2                                                    (b)
// Where underscore indicates the outside radiosity. a similar relation applies for surfaces 3,4, and 5. thus we can use equation (8-106a) for inside surface 2
// J2-(1-E2)*(F21*J1+F23*J3+F24*J4+F25*J5+F26*J6) = E2*(J2+J_2)/2                (c)
// and for outside surface 2
// J_2-(1-E2)*(F_26*J6) = E2*(J2+J_2)/2                                          (d)
// Equations like (c) and (d) are written for surfaces 3,4, and 5 also, and with the shape factors and emmissivities inserted the following set of equations is obtained
// J1-0.115*J2-0.24*J3-0.115*J4-0.24*J5 = 1.0012*10^(5)                  1
// -0.138*J1+0.8*J2-0.2*J_2-0.138*J3-0.0408*J4-0.138*J5 = 57.66          2
// 0.2*J2-0.8*J_2 = -250.68                                              3
// -0.144*J1-0.069*J2+0.8*J3-0.2*J_3-0.069*J4-0.05*J5 = 60.16            4
// 0.2*J3-0.8*J_3 = -250.68
```

```
    5
//  -0.138*J1-0.0408*J2-0.138*J3+0.8*J4-0.2*J_4 -0.138*J5 = 57.66                6
//  0.2*J4-0.8*J_4 = -250.68

    7
//  -0.144*J1-0.069*J2-0.057*J3-0.069*J4+0.8*J5-0.2*J_5 = 60.16                8
//  0.2*J5-0.8*J_5 = -250.68

    9
// We thus have nine equations and nine unknowns, which may be solved by matrix method
Z = [1 -0.115 -0.24 -0.115 -0.24 0 0 0 0;-0.138 0.8 -0.138 -0.0408 -0.138 -0.2 0 0 0;0 0.2 0 0 0 -0.8 0 0 0;-0.144 -0.069 0.8 -0.069 -0.05 0 -0.2 0 0;0 0 0.2 0 0 0 -0.8 0 0;-0.138 -0.0408 -0.138 0.8 -0.138 0 0 -0.2 0;0 0 0 0.2 0 0 0 -0.8 0;-0.144 -0.069 -0.057 -0.069 0.8 0 0 0 -0.2;0 0 0 0 0.2 0 0 0 -0.8];
C = [1.0012*10^(5);57.66;-250.68;60.16;-250.68;57.66;-250.68;60.16;-250.68];

J = Z^(-1)*C;
J1 = J(1);// [W/square meter]
J2 = J(2);// [W/square meter]
J3 = J(3);// [W/square meter]
J4 = J(4);// [W/square meter]
J5 = J(5);// [W/square meter]
J_2 = J(6);// [W/square meter]
J_3 = J(7);// [W/square meter]
J_4 = J(8);// [W/square meter]
J_5 = J(9);// [W/square meter]
// the temperatures are thus computed from equation (b):
Eb2 = (J2+J_2)/2;// [W/square meter]
T2 = (Eb2/sigma)^(1/4);// [K]
T4 = T2;// [K]
```

```
Eb3 = (J3+J_3)/2;// [W/square meter]
T3 = (Eb3/sigma)^(1/4);// [K]
T5 = T3;// [K]
// for surface 1 we observed that
q = 1*10^(5);// [W/square meter]
Eb1 = q*(1-E1)/E1+J1;// [W/square meter]
// and
T1 = (Eb1/sigma)^(1/4);// [K]
printf("temperature of all surfaces are following ")
    ;
printf("\n\n\t T1 = %f K",T1);
printf("\n\n\t T2 = %f K",T2);
printf("\n\n\t T3 = %f K",T3);
printf("\n\n\t T4 = %f K",T4);
printf("\n\n\t T5 = %f K",T5);

// surfaces 2,3,4, and 5 as one surface
// we now go back and take surfaces 2,3,4, and 5 as
    one surface, which we choose to call surface 7.
    the shape factors are then
F16 = 0.285;
F61 = 0.285;
F17 = 1-0.285;
A1 = 2.0;
A7 = 6.0;
// THUS
F71 = A1*F17/A7;
F77 = 1-2*F71;
F76 = F71;
F_76 = 1.0;
// then for surface 1 we use equation(8-109a) to
    obtain
// J1-(F17*J7+F16*J6) = 1.0*10^(5)
// using Eb7 = (J7+J_7)/2, we have for the inside of
     surface 7
// J7*[1-F77*(1-E7)]-(1-E7)*(F71*J1+F76*J6) = (J7+
    J_7)*E7/2
// while for the outside we have
```

```
// J_7-(1-E7)*F_76*J6 = (J7+J_7)*E7/2
// when the numerical values are inserted , we obtain
    the set of three equations:
// J1-0.715J7 = 1.0012*10^(5)
// -0.143*J1+0.486*J7-0.2*J_7 = 59.74
// 0.2*J7-0.8*J_7 = -250.68
// Solving above three equations by matrix method
Z = [1 -0.715 0;-0.143 0.486 -0.2;0 0.2 -0.8];
C = [1.0012*10^(5);59.74;-250.68];
J = Z^(-1)*C;
J1 = J(1);// [W/square meter]
J7 = J(2);// [W/square meter]
J_7 = J(3);// [W/square meter]
// the temperatures are thus computed as before
Eb7 = (J7+J_7)/2;// [W/square meter]
T7 = (Eb7/sigma)^(1/4);// [K]
Eb1 = q*(1-E1)/E1+J1;// [W/square meter]
T11 = (Eb1/sigma)^(1/4);// [K]
printf("\n\n from second method T1 = %f K",T11);
printf("\n\n so there is a difference of %f K
   between the two methods",T11-T1);
```

Scilab code Exa 8.19 numerical solution for combined convection and radiation non linear system

```
clear;
clc;
printf("\t\t\tExample Number 8.19\n\n\n");
// numerical solution for combined convection and
   radiation(non-linear system)
// Example 8.19 (page no.-449-452)
// solution

l = 0.5;// [m] length of plate
b = 0.5;// [m] breadth of plate
```

```
T1 = 1300;// [K] temperature of plate
Tinf = 300;// [K] temperature of surrounding
T4 = Tinf;// [degree celsius]
h = 50;// [W/square meter] convection heat transfer
    coefficient
E1 = 0.8;
E2 = 0.3;
E3 = 0.3;
// using figures 8-12(page no.-386) and 8-14(page no
    .-387), we can evaluate the shape factors as
F12 = 0.2;
F13 = 0.2;
F23 = 0.2;
F32 = 0.2;
F14 = 1-0.2-0.2;
F24_L = 1;
F34_R = 1;
F21 = F12;
F31 = F12;
F24_R = 0.6;
F34_L = 0.6;
F11 = 0;
F22 = 0;
F33 = 0;
// J2R = J3L
// J2L = J3R       From symmetry
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
Eb4 = sigma*T4^(4);// [W/square meter]
Eb1 = sigma*T1^(4);// [W/square meter]
J4 = Eb4;// [W/square meter]
// we now use equation(8-107) to obtain a relation
    for J1:
// J1 = (1-E1)*[F12*J2R+F13*J3L+F14*J4]+E1*Eb1
// but J2R = J3L and F12 = F13 so that
// J1 = (1-E1)*[2*F13*J2R+F14*J4]+E1*Eb1
                                            (a)
// we use equation (8-108) for the overall energy
    balance on surface 2:
```

```
// 2*h*(Tinf-T2) = E2*(Eb2-J2R)/(1-E2)+E2*(Eb2-J2L)/(1-E2)
// 2*h*(Tinf-T2) = E2*(2*Eb2-J2R-J2L)/(1-E2)                    (b)
// equation (8-105) is used for surface J2R.
// J2R = (1-E2)*(F21*J1+F23*J3L+F24_R*J4)+E2*Eb2
//  But J2R = J3L so that
// J2R = [(1-E2)*(F21*J1+F24_R*J4)+E2*Eb2]/(1-(1-E2)*F23)          (c)
// for surface J2L the equation is
// J2L = (1-E2)*(F24_L*J4)+E2*Eb2                                  (d)
// we now have four equations with four unknowns, J1,J2R,J2L,Eb2, with T2 = (Eb2/sigma)^(1/4).
// however equation (b) is nonlinear in Eb so we must use a special procedure to solve the set.
for T2 = 300:0.1:400
    Z = [1 -(1-E1)*2*F13 0 0;0 -E2/(1-E2) -E2/(1-E2) 2*E2/(1-E2);(1-E2)*F21/(1-(1-E2)*F23) -1 0 E2/(1-(1-E2)*F23);0 0 1 -E2];
    C = [E1*Eb1;2*h*(Tinf-T2);-F24_R/(1-(1-E2)*F23);(1-E2)*F24_L];
    S = Z^(-1)*C;
    Eb2_E = S(4);
    Eb2_T = sigma*T2^(4);
    dEb2 = Eb2_E-Eb2_T;
    if (dEb2>0 & dEb2<5) then
        J1 = S(1);// [W/square meter]
        J2R = S(2);// [W/square meter]
        J2L = S(3);// [W/square meter]
        Eb2 = S(4);// [W/square meter]
        T2new = T2;// [K]
    end
end
// the total heat flux lost by surface 1 is
q1_by_A1 = h*(T1-Tinf)+(Eb1-J1)*E1/(1-E1);// [W/square meter]
// for a 0.5 by 0.5 m surface the heat lost is thus
```

```
q1 = q1_by_A1*l*b;// [W]
printf("\n\n the heat lost by plate is %f W",q1);
```

Scilab code Exa 8.20 solar environment equilibrium temperatures

```
clear;
clc;
printf("\t\t\tExample Number 8.20\n\n\n");
// solar-environment equilibrium temperatures
// Example 8.20 (page no.-454)
// solution

q_by_A_sun = 700;// [W/m^(2)] solar flux
T_surr = 25+273;// [K] surrounding temperature
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
// at radiation equilibrium the netenergy absorbed
   from sun must equal the long-wavelength radiation
    exchange with the surroundings,or
// (q_by_A_sun)*alpha_sun = alpha_low_temp*sigma*(T
   ^4-T_surr^4)          (a)

// case (a) for white paint

// for white paint we obtain from table 8-4
alpha_sun = 0.12;
alpha_low_temp = 0.9;
// so that equation (a) becomes
T = [(q_by_A_sun)*alpha_sun/(alpha_low_temp*sigma)+
   T_surr^(4)]^(1/4);// [K]
printf("radiation equilibrium temperature for the
   plate exposed to solar flux if the surface is
   coated with white paint is %f degree celsius",T
   -273);

// case (b) for flat black lacquer we obtain
```

```
alpha_sun = 0.96;
alpha_low_temp = 0.95;
// so that equation (a) becomes
T = [(q_by_A_sun)*alpha_sun/(alpha_low_temp*sigma)+
   T_surr^(4)]^(1/4);// [K]
printf("\n\nradiation equilibrium temperature for
   the plate exposed to solar flux if the surface is
    coated with flat black lacquer is %f degree
   celsius",T-273);
```

Scilab code Exa 8.21 influence of convection on solar equilibrium temperature

```
clear;
clc;
printf("\t\t\tExample Number 8.21\n\n\n");
// influence of convection on solar equilibrium
   temperature
// Example 8.21 (page no.-455)
// solution

T_surr = 25+273;// [K] surrounding temperature
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
h = 10;// [W/square meter] heat transfer coefficient
// in this case the solar energy absorbed must equal
    the sum of the radiation and convection
   transfers to the surroundings
// (q_by_A_sun)*alpha_sun = alpha_low_temp*sigma*(T
   ^4-T_surr^4)+h*(T-T_surr)          (a)
q_by_A_sun = 700;// [W/m^(2)] solar flux

// for the white paint, using the same surface
   properties as in example 8-20 gives

```

```
alpha_sun = 0.12;
alpha_low_temp = 0.9;
// so that equation (a) becomes
deff('[y] = f(T)','y = (q_by_A_sun)*alpha_sun-alpha_low_temp*sigma*(T^4-T_surr^4)-h*(T-T_surr)');
T = fsolve(1,f);
printf("the radiation-convection equillibrium temperatures for case (a) is %f degree celsius",T-273);

//for flat black lacquer we obtain

alpha_sun = 0.96;
alpha_low_temp = 0.95;
// so that equation (a) becomes
deff('[y] = f2(T1)','y = (q_by_A_sun)*alpha_sun - alpha_low_temp*sigma*(T1^4-T_surr^4)-h*(T1-T_surr)');
T1 = fsolve(1,f2);
printf("\n\n the radiation-convection equillibrium temperatures for case (b) is %f degree celsius",T1-273);
printf("\n\n where case (a)     surface is coated with white paint");
printf("\n\n       case (b)     surface is coated with flat black lacquer");
```

Scilab code Exa 8.23 temperature measurement error caused by radiation

```
clear;
clc;
printf("\t\t\tExample Number 8.23\n\n\n");
// temperature measurement error caused by radiation
// Example 8.23 (page no.-460)
```

```
// solution

E = 0.9;// emissivity of mercury-in-glass
   thermometer
Tt = 20+273;// [K] temperature indicated by
   thermometer
Ts = 5+273;// [K] temperature of walls
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
h = 8.3;// [W/square meter] heat transfer
   coefficient for thermometer
// we employ equation(8-113) for the solution: h*(
   Tinf-Tt) = sigma*E*(Tt^4-Ts^4)
// inserting the values in above equation
Tinf = sigma*E*(Tt^4-Ts^4)/h+Tt;// [K]
printf("the true air temperature is %f degree
   celsius",Tinf-273);
```

Chapter 9

Condensation and Boiling Heat Transfer

Scilab code Exa 9.1 condensation on vertical plate

```
clear;
clc;
printf("\t\t\tExample Number 9.1\n\n\n");
// condensation on vertical plate
// Example 9.1(page no.-492)
// solution

// we have to check the reynolds no. to that film is
    laminar or turbulent
Tf = (100+98)/2;// [degree celsius]
Tw = 98;// [degree celsius]
RHOf=960;// [kg/cubic meter]
MUf=2.82*10^(-4);// [kg/m s]
Kf=0.68;// [W/m degree celsius]
g=9.81;// [m/s^(2)]
L=0.3;// [m]
// RHOf(RHOf-RHOv)~RHOf^(2)
// let us assume laminar film condensate
Tsat=100;// [degree celsius]
```

```
Tg=100;// [degree celsius]
Hfg=2255*10^(3);// [J/kg]
hbar=0.943*((RHOf^(2)*g*Hfg*Kf^(3)/(L*MUf*(Tg-Tw)))
   ^(0.25));// [W/square meter degree celsius]
h=hbar;// [W/square meter degree celsius]
// checking reynolds no. with equation(9-17)
Ref=4*h*L*(Tsat-Tw)/(Hfg*MUf);
printf("value of reynolds no. is %f \n\n so the
   laminar assumption was correct ",Ref);
// the heat transfer is now calculated from
A=0.3*0.3;// [square meter]
q=hbar*A*(Tsat-Tw);// [W]
mdot=q/Hfg;// [kg/h]
printf("\n\n the heat transfer is %f w",q);
mdot=mdot*3600;// [kg/h]
printf("\n\n total mass flow condensate is %f kg/h",
   mdot);
```

Scilab code Exa 9.2 condensation on tube tank

```
clear;
clc;
printf("\t\t\tExample Number 9.2\n\n\n");
// condensation on tube tank
// Example 9.2(page no.-493)
// solution

// the condensate properties are obtained from
   previous example
// replacing L by n*d
Tw=98;// [degree celsius]
RHOf=960;// [kg/cubic meter]
MUf=2.82*10^(-4);// [kg/m s]
Kf=0.68;// [W/m degree celsius]
g=9.81;// [m/s^(2)]
```

```
Tsat=100;// [degree celsius]
Tg=100;// [degree celsius]
Hfg=2255*10^(3);// [J/kg]
d=0.0127;// [m]
n=10;
hbar=0.725*((RHOf^(2)*g*Hfg*Kf^(3)/(n*d*MUf*(Tg-Tw))
   )^(0.25));// [W/square meter degree celsius]
// total surface area is
n=100;
Al=n*22*d/7;// [square meter]
printf("total surface area is %f square meter/m",Al)
   ;
// so the heat transfer is
Ql=hbar*Al*(Tg-Tw);// [W]
printf("\n\n heat transfer is %f kW/m",Ql/1000);
// total mass flow of condensate is then
mdotl=Ql/Hfg;// [kg/h]
mdotl=mdotl*3600;// [kg/h]
printf("\n\n total mass flow of condensate is %f kg/
   h",mdotl);
```

Scilab code Exa 9.3 boiling on brass plate

```
clear;
clc;
printf("\t\t\tExample Number 9.3\n\n\n");
// boiling on brass plate
// Example 9.3(page no.-501-502)
// solution

Qawater_platinum=946.1;//[kw/square meter] from
   figure (9-8) heat flux for water platinum
   combination
Tw=117;// [degree celsius]
Tsat=100;// [degree celsius]
```

```
// from table (9-2)
Csfwater_platinum=0.013;// for water platinum
Csfwater_brass=0.006;// for water brass
deff('y = G(Qawater_brass)','y = (((Qawater_brass)/((Qawater_platinum)))-((Csfwater_platinum/Csfwater_brass)^(3)))');
Qawater_brass = fsolve(0,G);
printf("heat transfer for water brass system is %f W/square meter",Qawater_brass);
```

Scilab code Exa 9.4 Flow boiling

```
clear;
clc;
printf("\t\t\tExample Number 9.4\n\n\n");
// Flow boiling
// Example 9.4(page no.-506)
// solution

p = 5*101325/10^(6);// [MPa] pressure of water
d = 0.0254;// [m] diameter of tube
Tw = 10;// [degree celsius]
// for calculation we use equation (9-45), noting that
dT = 10;// [degree celsius]
// the heat transfer coefficient is calculated as
h = 2.54*Tw^(3)*exp(p/1.551);// [W/square meter degree celsius]
// the surface area for a 1-m length of tube is
L = 1;// [m]
A = %pi*d*L;// [square meter]
// so the heat transfer is
q = h*A*dT;// [W/m]
printf("the heat transfer in a 1.0 m length of tube is %f W/m",q);
```

Scilab code Exa 9.5 water boiling in a pan

```
clear;
clc;
printf("\t\t\tExample Number 9.5\n\n\n");
// water boiling in a pan
// Example 9.5(page no.-506-507)
// solution

p = 101325/10^(6);// [MPa] pressure of water
dT_x = 8;// [degree celsius]
p1 = 0.17;// [MPa] given operating pressure
// we will use the simplified relation of table
   9-13(page no.-506) for the estimates.we do not
   know the value of q_by_A and so must choose one
   of the two relation for a horizontal surface from
    the table
// we anticipate nucleate boiling , so choose
h = 5.56*dT_x^(3);// [W/square meter degree celsius]
// and the heat flux is
q_by_A = h*dT_x;// [W/square meter]
// for operation as a pressure cooker we obtain the
   value of h from equation(9-44)
hp = h*(p1/p)^(0.4);// [W/square meter degree
   celsius]
// the corresponding heat flux is
q_by_A1 = hp*dT_x;// [W/square meter]
printf("heat flux obtained is %f kW/square meter",
   q_by_A/1000);
per_inc = 100*(q_by_A1-q_by_A)/q_by_A;
printf("\n\n if the pan operates as a pressure
   cooker at 0.17 MPa the increase in heat flux is
   %f percent",per_inc);
```

Scilab code Exa 9.6 heat flux comparisons

```
clear;
clc;
printf("\t\t\tExample Number 9.6\n\n\n");
// heat-flux comparisons
// Example 9.6(page no.-509)
// solution

Tw = 200;// [degree celsius] water temperature
L = 0.08;// [m] length of solid copper bar
dT = 100;// [degree celsius] temperature
   differential in copper bar
//using the data of table 9-4(page no.-508)
// the heat flux per unit area is expressed as
   q_by_A = -k*del_T/dx
// from table A-2(page no.-) the thermal
   conductivity of copper is
k = 374;// [W/m degree celsius]
q_by_A = -k*(-dT)/L;// [W/square meter]
// from table 9-4(page no.-508) the typical axial
   heat flux for a water heat flux for a water heat
   pipe is
q_by_A_axial = 0.67;// [kW/csquare meter]
q_by_A = q_by_A/(1000*10^(4));// [kW/csquare meter]
printf("thus the heat transfers more than %f times
   the heat of a pure copper rod with a substantial
   temperature gradient.",q_by_A_axial/q_by_A);
```

Chapter 10

Heat Exchangers

Scilab code Exa 10.1 overall heat transfer coefficient for pipe in air

```
clear;
clc;
printf("\t\t\tExample Number 10.1\n\n\n");
// overall heat transfer coefficient for pipe in air
// Example 10.1 (page no.-520-522)
// solution

Tw = 98;// [degree celsius] temperature of hot water
k_p = 54;// [W/m degree celsius] heat transfer coefficient of pipe
Ta = 20;// [degree celsius] atmospheric air temperature
u = 0.25;// [m/s] water velocity
// from appendix A the dimensions of 2-in schedule 40 pipe are
ID = 0.0525;// [m]
OD = 0.06033;// [m]
// the properties of water at 98 degree celsius are
rho = 960;// [kg/cubic meter]
mu = 2.82*10^(-4);// [kg/m s]
k_w = 0.68;// [W/m degree celsius]
```

```
Pr = 1.76;// prandtl number
// the reynolds number is
Re = rho*u*ID/mu;
// and since turbulent flow is encountered, we may use equation(6-4):
Nu = 0.023*Re^(0.8)*Pr^(0.4);
hi = Nu*k_w/ID;// [W/square meter degree celsius]
// for unit length of pipe the thermal resistance of the steel is
Rs = log(OD/ID)/(2*%pi*k_p);
// again, on a unit length basis the thermal resistance on the inside is
Ai = %pi*ID;// [square meter]
Ri = 1/(hi*Ai);
Ao = %pi*OD;// [square meter]
// the thermal resistance for outer surface is as yet unknown but is written, for unit lengths, is Ro = 1/(ho*Ao)                    (a)
// from table 7-2(page no.-339), for laminar flow, the simplified relation for ho is
// ho = 1.32*(dT/d)^(1/4) = 1.32*((To-Ta)/OD)^(1/4)

(b)
// where To is the unknown outside pipe surface temperature. we designate the inner pipe surface as Ti and the water temperature as Tw; then the energy balance requires
// (Tw-Ti)/Ri = (Ti-To)/Rs = (To-Ta)/Ro

(c)
// combining equations (a) and (b) gives
// (To-Ta)/Ro = %pi*OD*1.32*(To-Ta)^(5/4)/OD^(1/4)
// this relation may be introduced into equation (c) to yield two equations with the two unknowns Ti and To:

// (Tw-Ti)/Ri = (Ti-To)/Rs                    (1)
// (Ti-To)/Rs = %pi*OD*1.32*(To-Ta)^(5/4)/OD^(1/4)
```

```
                 (2)
// this is a non-linear equation which can be solved
   as
for Ti = 50:0.001:100
    Q = ((Ti-(Ti-(Tw-Ti)*(Rs/Ri)))/Rs)-(%pi*OD
       *1.32*((Ti-(Tw-Ti)*(Rs/Ri))-Ta)^(5/4)/OD
       ^(1/4));
    if Q>0 & Q<6 then
        Tinew = Ti;
    else
        Ti = Ti;
    end
end
Ti = Tinew;// [degree celsius]
To = (Ti-(Tw-Ti)*(Rs/Ri));// [Degree celsius]
// as a result , the outside heat transfer
   coefficient and thermal resistance are
ho = 1.32*((To-Ta)/OD)^(1/4);// [W/square meter
   degree celsius]
Ro = 1/(OD*7.91*%pi);//
// the overall heat transfer coefficient based on
   the outer area is written in terms of these
   resistances as
Uo = 1/(Ao*(Ri+Ro+Rs));// [W/area degree celsius]
// in this calculation we used the outside area for
   1.0 m length as  Ao
// so
Uo = Uo;// [W/square meter degree celsius]
printf("overall heat transfer coefficient is %f W/
   square meter degree celsius",Uo);
```

Scilab code Exa 10.2 overall heat transfer coefficient for pipe exposed to steam

```
clear;
```

```
clc;
printf("\t\t\tExample Number 10.2\n\n\n");
// overall heat transfer coefficient for pipe exposed to steam
// Example 10.2 (page no.-523-524)
// solution

p = 101325;// [Pa] pressure of steam
Tg = 100;// [degree celsius] temperature of steam
// we have already determined the inside convection heat-transfer coefficient in example(10.1) as
hi = 1961;// [W/square meter]
// the water film properties are
rho = 960;// [kg/cubic meter] density
mu_f = 2.82*10^(-4);// [kg/m s]
kf = 0.68;// [W/m degree celsius]
hfg = 2255*10^(3);// [J/kg]
g = 9.8;// [m/s^(2)] acceleration due to gravity
d = 0.06033;// [m] diameter of the pipe
// the convection coefficient for condensation on the outside of the pipe is obtained by using equation(9-12)
// h_o = 0.725*[(rho^(2)*g*hfg*kf^(3))/(mu_f*d*(Tg-To))]^(1/4)                    (a)
Ao = %pi*d;// [square meter] outside area
// outside thermal resistance per unit length is
// R_o = 1/(h_o*A_o)                                                              (b)
// the energy balance requires
// [Tg-To]/R_o = [To-Ti]/R_s = [Ti-Tw]/R_i                                        (c)
// from example 10.1 we have
Ri = 3.092*10^(-3);
Rs = 4.097*10^(-4);
Tw = 98;// [degree celsius]
// equation (b) and (c) may be combined to give
```

```
// (Tg-To)^(3/4)/3403 = (To-Ti)/Rs                (1)
// (To-Ti)/Rs = (Ti-Tw)/Ri              (2)
// this is a non-linear equation which can be solved
    as
for Ti = 98.1:0.01:99.75
    P = ((Tg-(Ti+Rs*(Ti-Tw)/Ri))^(3/4))*3403-(((Ti+
       Rs*(Ti-Tw)/Ri)-Ti)/Rs);
    if P>(-10) & P<0 then
        Tinew = Ti;
    else
        Ti = Ti;
    end

end
Ti = Tinew;// [degree celsius]
To = (Ti+Rs*(Ti-Tw)/Ri);// [degree celsius]
// the exterior heat-transfer coefficient and
   thermal resistance then become
ho = 0.725*[(rho^(2)*g*hfg*kf^(3))/(mu_f*d*(Tg-To))
   ]^(1/4);// [W/square meter degree celsius]
Ro = 1/(ho*Ao);
// based on unit length of pipe, the overall heat
   transfer coefficient is
Uo = 1/(Ao*(Ri+Ro+Rs));// [W/area degree celsius]
// since Ao and the R's were per unit length
// so
Uo = Uo;// [W/square meter degree celsius]
printf("overall heat transfer coefficient is %f W/
   square meter degree celsius",Uo);
```

Scilab code Exa 10.3 influence of fouling factor

```
clear;
clc;
printf("\t\t\tExample Number 10.3\n\n\n");
```

```
// influence of fouling factor
// Example 10.2 (page no.-524-525)
// solution

// the fouling factor influences the heat transfer
   coefficient on the inside of the pipe. we have
Rf = 0.0002;
// using
h_clean = 1961;// [W/square meter degree celsius]
// we obtain
hi = 1/[Rf+(1/h_clean)];// [W/square meter degree
   celsius]
printf("the percent reduction because of fouling
   factor is %f ",(h_clean-hi)*100/h_clean);
```

Scilab code Exa 10.4 calculation of heat exchanger size from known temperatures

```
clear;
clc;
printf("\t\t\tExample Number 10.4\n\n\n");
// calculation of heat exchanger size from known
   temperatures
// Example 10.4 (page no.-532-533)
// solution

m_dot = 68;// [kg/min] water flow rate
U = 320;// [W/square meter degree celsius] overall
   heat transfer coefficient
T1 = 35;// [degree celsius] initial temperature
T2 = 75;// [degree celsius] final temperature
Toe = 110;// [degree celsius] oil entering
   temperature
Tol = 75;// [degree celsius] oil leaving temperature
Cw = 4180;// [J/kg degree celsius] water specific
```

```
heat capacity
// the total heat transfer is determined from the energy absorbed by the water:
q = m_dot*Cw*(T2-T1);// [J/min]
q = q/60;// [W]
// since all the fluid temperatures are known, the LMTD can be calculated by using the temperature scheme in figure 10-7b(page no.-530)
dT_m = ((Toe-Tol)-(T2-T1))/log((Toe-Tol)/(T2-T1));// [degree celsius]
// then, since  q = U*A*dT_m
A = q/(U*dT_m);// [square meter] area of heat-exchanger
printf("area of heat-exchanger is %f square meter ",A);
```

Scilab code Exa 10.5 shell and tube heat exchanger

```
clear;
clc;
printf("\t\t\tExample Number 10.5\n\n\n");
// shell-and-tube heat exchanger
// Example 10.5 (page no.-533-534)
// solution

// to solve this problem, we determine a correction factor from figure 10-8 to be used with the LMTD calculated on the basis of counterflow exchanger.
// the parameters according to the nomenclature of figure 10-8(page no.-532) are
T1 = 35;// [degree celsius]
T2 = 75;// [degree celsius]
t1 = 110;// [degree celsius]
t2 = 75;// [degree celsius]
P = (t2-t1)/(T1-t1);
```

```
R = (T1-T2)/(t2-t1);
// so the correction factor is
F = 0.81;// from figure 10-10(page no.-534)
// and the heat transfer is q = U*A*F*dT_m
// so that. from example 10-4 we have
U = 320;// [W/square meter degree celsius] overall heat transfer coefficient
q = 189493.33;// [W]
dT_m = 37.44;// [degree celsius]
A = q/(U*F*dT_m);// [square meter]
printf("area required for this exchanger is %f square meter",A)
```

Scilab code Exa 10.6 design of shell and tube heat exchanger

```
clear;
clc;
printf("\t\t\tExample Number 10.6\n\n\n");
// design of shell-and-tube heat exchanger
// Example 10.5 (page no.-534-536)
// solution

m_dot_c = 3.8;// [kg/s] water flow rate
Ti = 38;// [degree celsius] initial temperature of water
Tf = 55;// [degree celsius] final temperature of water
m_dot_h = 1.9;// [kg/s] water flow rate entering the exchanger
Te = 93;// [degree celsius] entering water temperature
U = 1419;// [W/square meter degree celsius] overall heat transfer coefficient
d = 0.019;// [m] diameter of tube
v_avg = 0.366;// [m/s] average water velocity in
```

```
    exchanger
Cc = 4180;// [] specific heat of water
Ch = Cc;// [] specific heat
rho = 1000;// [kg/cubic meter] density of water
// we first assume one tube pass and check to see if
     it satisfies the conditions of this problem. the
     exit temperature of the hot water is calculated
    from
dTh = m_dot_c*Cc*(Tf-Ti)/(m_dot_h*Ch);// [degree
    celsius]
Th_exit = Te-dTh;// [degree celsius]
// the total required heat transfer is obtained for
    the cold fluid is
q = m_dot_c*Cc*(Tf-Ti);// [W]
// for a counterflow exchanger, with the required
    temperature
LMTD = ((Te-Tf)-(Th_exit-Ti))/log((Te-Tf)/(Th_exit-
    Ti));// [degree celsius]
dTm = LMTD;// [degree celsius]
A = q/(U*dTm);// [square meter]
// using the average water velocity in the tubes and
     the flow rate, we calculate the total area with
A1 = m_dot_c/(rho*v_avg);// [square meter]
// this area is the product of number of tubes and
    the flow area per tube:
n = A1*4/(%pi*d^(2));// no. of tubes
n = ceil(n);// rounding of value of n because no. of
     pipe is an integer value
// the surface area per tube per meter of length is
S = %pi*d;// [square meter/tube meter]
// we recall that the total surface area required
    for a one tube pass exchanger was calculated
    above .
// we may thus compute the length of tube for this
    type of exchanger from
L = A/(S*n);// [m]
// this length is greater than the allowable 2.438 m
    , so we must use more than one tube pass. when we
```

```
    increase the number of passes , we
    correspondingly increase the total surface area
    required because of the reduction in LMTD caused
    by the correction factor F.
// we next try two tube passes. from figure 10-8(
    page no.-532)
F = 0.88;
A_total = q/(U*F*dTm);// [square meter]
// the number of tubes per pass is still 37 because
    of the velocity requirement. for the two pass
    exchanger the total surface area is now related
    to the length by
L1 = A_total/(2*S*n);// [m]
// this length is within the 2.438 m requirement , so
    the final design choice is
printf("number of tubes per pass = %f",n);
printf("\n\n number of passes = 2");
printf("\n\n length of tube per pass = %f m",L1);
```

Scilab code Exa 10.7 cross flow exchanger with one fluid mixed

```
clear;
clc;
printf("\t\t\tExample Number 10.7\n\n\n");
// cross flow exchanger with one fluid mixed
// Example 10.7 (page no.-537)
// solution

m_dot = 5.2;// [kg/s] mass flow rate
T1 = 130;// [degree celsius] temperature of entering
    steam
T2 = 110;// [degree celsius] temperature of leaving
    steam
t1 = 15;// [degree celsius] temperature of entering
    oil
```

```
t2 = 85;// [degree celsius] temperature of leaving
    oil
c_oil = 1900;// [J/kg degree celsius] heat capacity
    of oil
c_steam = 1860;// [J/kg degree celsius] heat
    capacity of steam
U = 275;// [W/square meter degree celsius] overall
    heat transfer coefficient
//the total heat transfer may be obtained from an
    energy balance on the steam
q = m_dot*c_steam*(T1-T2);// [W]
// we can solve for the area from equation (10-13).
    the value of dT_m is calculated as if the
    exchanger were counterflow double pipe,thus
dT_m = ((T1-t2)-(T2-t1))/log((T1-t2)/(T2-t1));// [
    degree celsius]
// t1,t2 is representing the unmixed fluid(oil) and
    T1,T2 is representing the mixed fluid(steam) so
    that:
// we calculate
R = (T1-T2)/(t2-t1);
P = (t2-t1)/(T1-t1);
// consulting figure 10-11(page no.-534) we find
F = 0.97;
// so the area is calculated from
A = q/(U*F*dT_m);// [square meter]
printf("surface area of heat exchanger is %f square
    meter",A);
```

Scilab code Exa 10.8 effects of off design flow rates for exchanger in previous example

```
clear;
clc;
printf("\t\t\tExample Number 10.8\n\n\n");
```

```
// effects of off-design flow rates for exchanger in example 10-7
// Example 10.8 (page no.-537-538)
// solution

// we did not calculate the oil flow in example 10-7 but can do so now from
q = 193;// [kW]
c_oil = 1.9;// [J/kg degree celsius] heat capacity of oil
t1 = 15;// [degree celsius] temperature of entering oil
t2 = 85;// [degree celsius] temperature of leaving oil
m_dot_o = q/(c_oil*(t2-t1));// [kg/s]
// the new flow rate will be half this value
m_dot_o = m_dot_o/2;// [kg/s]
// we are assuming the inlet temperatures remain the same at 130 degree celsius for the steam and 15 degree celsius for the oil.
// the new relation for the heat transfer is q = m_dot_o*c_oil*(Teo-15) = m_dot_s*cp*(130-Tes)                                (a)
// but the exit temperatures, Teo and Tes are unknown. furthermore, dT_m is unknown without these temperatures, as are the values of R and P from figure  10-11(page no.-535). this means we must use an iterative procedure to solve for the exit temperatures using equation (a) and   q = U*A*F*dT_m             (b)
// the general procedure is to assume values of the exit temperatures until the q's agree between equations(a) and (b).
printf("the objective of this example is to show that an iterative procedure is required when the inlet and outlet temperatures are not known or easily calculated");
printf("\n\n there is no need to go through this
```

iteration because it can be avoided by using the techniques described in section 10−6");
```

**Scilab code Exa 10.9** off design calculation using E NTU method

```
clear;
clc;
printf("\t\t\tExample Number 10.9\n\n\n");
// off−design calculation using E−NTU method
// Example 10.9 (page no.−542−544)
// solution

m_dot_o = 0.725;// [kg/s] oil flow rate
m_dot_s = 5.2;// [kg/s] steam flow rate
t1 = 15;// [degree celsius] temperature of entering
    oil
T1 = 130;// [degree celsius] temperature of entering
     steam
c_oil = 1900;// [J/kg degree celsius] heat capacity
    of oil
c_steam = 1860;// [J/kg degree celsius] heat
    capacity of steam
// for the steam
Cs = m_dot_s*c_steam;// [W/degree celsius]
// for the oil
Co = m_dot_o*c_oil;// [W/degree celsius]
// so the oil is minium fluid. we thus have
C_min_by_C_max = Co/Cs;
U = 275;// [W/square meter degree celsius] overall
    heat transfer coefficient
A = 10.83;// [square meter] surface area of heat
    exchanger
NTU = U*A/Co;
// we choose to use the table and note that Co(
    minimum) is unmixed and Cs(maximum) is mixed so
```
```

```
that the first relation in the table 10-3(page no.-543) applies.
// we therfore calculate E(effectiveness) as
E = (1/C_min_by_C_max)*{1-exp(-C_min_by_C_max*(1-exp(-NTU)))};
// if we were using figure 10-14(page no.-544) we would have to evaluate
C_mixed_by_C_unmixed = Cs/Co;
// and would still determine
E = 0.8;// approximately
// now, using the effectiveness we can determine the temperature difference of the minimum fluid(oil as)
dT_o = E*(T1-t1);// [degree celsius]
// so that heat transfer is
q = m_dot_o*c_oil*(dT_o);// [W]
q_initial = 193440;// [W] heat transfer when oil flow rate is 100 %
printf("we find a reduction in the oil flow rate of 50 %% causes a reduction in heat transfer of only %f %%",(q_initial-q)*100/q_initial);
```

Scilab code Exa 10.10 off design calculation of exchanger in example 10 4

```
clear;
clc;
printf("\t\t\tExample Number 10.10\n\n\n");
// off-design calculation of exchanger in example 10-4
// Example 10.10 (page no.-544-546)
// solution

m_dot_c = 68;// [kg/min] water flow rate
T1 = 35;// [degree celsius] initial temperature
```

```
T2 = 75;// [degree celsius] final temperature
Toe = 110;// [degree celsius] oil entering temperature
Tol = 75;// [degree celsius] oil leaving temperature
Cc = 4180;// [J/kg degree celsius] water specific heat capacity
Ch = 1900;// [J/kg degree celsius] heat capacity of oil
U = 320;// [W/square meter degree celsius] overall heat transfer coefficient
A = 15.814568;// [square meter] area of heat exchanger (from example 10-4)
// the flow rate of oil is calculated from the energy balance for the original problem:
m_dot_h = m_dot_c*Cc*(T2-T1)/(Ch*(Toe-Tol));// [kg/min]
// the capacity rates for the new conditions are calculated as
C_h = m_dot_h*Ch/60;// [W/degree celsius]
C_c = m_dot_c*Cc/60;// [W/degree celsius]
// so that the water (cold fluid) is the minimum fluid , and
C_min_by_C_max = C_c/C_h;
NTU_max = U*A/C_c;
// from figure 10-13(page no.-542) or table 10-3(page no.-543) the effectiveness is
E = 0.744;
// and because the cold fluid is the minimum, we can write
dT_cold = E*(Toe-T1);// [degree celsius]
// and the exit water temperature is
Tw_exit = T1+dT_cold;// [degree celsius]
// the total heat transfer under the new flow conditions is calculated as
m_dot_c = 40;// [kg/min]
q = m_dot_c*Cc*dT_cold/60;// [W]
printf("exit water temperature is %f degree celcius",Tw_exit);
```

```
printf("\n\n the total heat transfer under the new
    flow conditions is %f kW",q/1000);
```

Scilab code Exa 10.11 cross flow exchanger with both fluid unmixed

```
clear;
clc;
printf("\t\t\tExample Number 10.11\n\n\n");
// cross-flow exchanger with both fluid unmixed
// Example 10.11 (page no.-547-549)
// solution

pa = 101325;// [Pa] pressure of air
Ti = 15.55;// [degree celsius] initial temperature
    of air
Tf = 29.44;// [degree celsius] final temperature of
    air
Thw = 82.22;// [degree celsius] hot water
    temperature
U = 227;// [W/square meter degree celsius] overall
    heat transfer coefficient
S = 9.29;// [square meter] total surface area of
    heat exchanger
R = 287;// [] universal gas constant
Cc = 1006;// [J/kg degree celsius] specific heat of
    air
Ch = 4180;// [J/kg degree celsius] specific heat of
    water
// the heat transfer is calculated from the energy
    balance on the air. first , the inlet air density
    is
rho = pa/(R*(Ti+273.15));// [kg/cubic meter]
// so the mass flow of air (the cold fluid) is
mdot_c = 2.36*rho;// [kg/s]
// the heat transfer is then
```

```
q = mdot_c*Cc*(Tf-Ti);// [W]
// from the statement of the problem we do not know whether the air or water is the minimum fluid. a trial and error procedur must be used with figure 10-15(page no.-545) or table 10-3(page no.-543).
// we assume that the air is the minimum fluid and then check out our assumption. then
Cmin = mdot_c*Cc;// [W/degree celsius]
NTU_max = U*S/Cmin;
// and the effectiveness based on the air as the minimum fluid is
E = (Tf-Ti)/(Thw-Ti);
// entering figure 10-15, we are unable to match these quantities with the curves. this require that the hot fluid be the minimum. we must therefore assume values for the water flow rate until we able to match the performance as given by figure 10-15 or table 10-3. we first note that
Cmax = mdot_c*Cc;// [W/degree celsius]                    (a)
// NTU_max = U*S/Cmin;                                     (b)
// E = dT_h/(Thw-Ti)                                       (c)
// dT_h = q/Cmin                                           (d)

// now we assume different values for Cmin abd calculate different-different values for NTU_max, dT_h, and E

// for
Cmin_by_Cmax1 = 0.5;
Cmin1 = Cmin_by_Cmax1*Cmax;// [W/degree celsius]
NTU_max1 = U*S/Cmin1;
dT_h1 = q/Cmin1;// [degree celsius]
E1_c1 = dT_h1/(Thw-Ti);// calculated
E1_t1 = 0.65;// from table

```

```
// for
Cmin_by_Cmax2 = 0.25;
Cmin2 = Cmin_by_Cmax2*Cmax;// [W/degree celsius]
NTU_max2 = U*S/Cmin2;
dT_h2 = q/Cmin2;// [degree celsius]
E1_c2 = dT_h2/(Thw-Ti);// calculated
E1_t2 = 0.89;// from table

// for
Cmin_by_Cmax3 = 0.22;
Cmin3 = Cmin_by_Cmax3*Cmax;// [W/degree celsius]
NTU_max3 = U*S/Cmin3;
dT_h3 = q/Cmin3;// [degree celsius]
E1_c3 = dT_h3/(Thw-Ti);// calculated
E1_t3 = 0.92;// from table

// we estimate the water-flow rate as about
Cmin = 660;// [W/degree celsius]
mdot_h = Cmin/Ch;// [kg/s]
// the exit water temperature is accordingly
Tw_exit = Thw-q/Cmin;// [degree celsius]
printf("the exit water temperature is %f degree
   celsius",Tw_exit);
printf("\n\n the heat transfer is %f kW",q/1000);
```

Scilab code Exa 10.12 comparison of single or two exchanger options

```
clear;
clc;
printf("\t\t\tExample Number 10.12\n\n\n");
// comparison of single- or two-exchanger options
// Example 10.12 (page no.-549-551)
// solution

mdot_c = 1.25;// [kg/s] water flow rate
```

```
Ti = 35;// [degree celsius] initial temperature of water
Tf = 80;// [degree celsius] final temperature of water
Toi = 150;// [degree celsius] initial temperature of oil
Tof = 85;// [degree celsius] final temperature of oil
U = 850;// [W/square meter degree celsius] overall heat transfer coefficient
Cp_water = 4180;// [] specific heat of water
Cp_oil = 2000;// [J/kg degree celsius]
// we calculate the surface area required for both alternatives and then compare costs. for the one large exchanger
q = mdot_c*Cp_water*(Tf-Ti);// [W]
mdot_c_into_Cp_water = mdot_c*Cp_water;// [W/degree celsius]
mdot_h_into_Cp_oil = q/(Toi-Tof);// [W/degree celsius]
Cmin = mdot_h_into_Cp_oil;// [W/degree celsius]
Cmax = mdot_c_into_Cp_water;// [W/degree celsius]
// so that oil is the minimum fluid:
Eh = (Toi-Tof)/(Toi-Ti);
Cmin_by_Cmax = Cmin/Cmax;
// from figure 10-13(page no.-542),
NTU_max = 1.09;
A = NTU_max*Cmin/U;// [square meter]
// we now wish to calculate the surface-area requirement for the two small exchanger because U*A and Cmin are the same for each exchanger.
// this requires that the effectiveness be the same for each exchanger. thus,
// E1 = (Toi-Toe_1)/(Toi-Ti) = E2 = (Toi-Toe_2)/(Toi-Tw2)
   (a)
// where the nomenclature for the temperatures is indicated in the sketch. because the oil flow is
```

```
   the same in each exchanger and the average exit
   oil temperature must be 85 degree celsius , we may
    write
// (Toe_1+Toe_2)/2 = 85

   (b)
// an energy balance on the second heat exchanger
   gives
// mdot_c_into_Cp_water*(Tf-Tw2) =
   mdot_h_into_Cp_oil*(Toi-Toe_2)/2
                                             (c)
// we now have three equations (a),(b), and (c)
   which may be solved for the three unknowns Toe_1,
    Toe_2, and Tw2.
// eliminating Tw2, and Toe_1 from equation (a) by
   the help of equation (b) and (c)
deff('[y] = H(Toe_2)','y = (Toi-(170-Toe_2))/(Toi-Ti
   ) - (Toi-Toe_2)/(Toi-(Tf-(mdot_h_into_Cp_oil*(Toi
   -Toe_2)/(mdot_c_into_Cp_water*2))))');
Toe_2 = fsolve(1,H);// [degree celsius]
Toe_1 = (170-Toe_2);// [degree celsius]
Tw2 = (Tf-(mdot_h_into_Cp_oil*(Toi-Toe_2)/(
   mdot_c_into_Cp_water*2)));// [degree celsius]
// the effectiveness can then be calculated as
E1 = (Toi-Toe_1)/(Toi-Ti);
E2 = E1;
// from figure 10-13(page no.-542), we obtain
NTU_max = 1.16;
// so that
A1 = NTU_max*Cmin/(U*2);// [square meter]
printf("we have find that %f square meter of area is
    required for each of small exchangers, or a
   total of %f square meter",A1,2*A1);
printf("\n\n the area required in the one larger
   exchanger is %f square meter",A);
printf("\n\n the cost per unit area is greater so
   that the most economical choice would be the
   single larger exchanger ");
```

Scilab code Exa 10.13 shell and tube exchangeras air heater

```
clear;
clc;
printf("\t\t\tExample Number 10.13\n\n\n");
// shell and tube exchangeras air heater
// Example 10.13 (page no.-551-552)
// solution

To = 100;// [degree celsius] temperature of hot oil
m_dot_a = 2;// [kg/s] flow rate of air
T1 = 20;// [degree celsius] initial temperature of air
T2 = 80;// [degree celsius] final temperature of air
Cp_o = 2100;// [J/kg degree celsius] specific heat of the oil
Cp_a = 1009;// [J/kg degree celsius] specific heat of the air
m_dot_o = 3;// [kg/s] flow rate of oil
U = 200;// [W/square meter] overall heat transfer coefficient
// the basic energy balance is m_dot_o*Cp_o*(To-Toe) = m_dot_a*Cp_a*(T2-T1)
Toe = To-m_dot_a*Cp_a*(T2-T1)/(m_dot_o*Cp_o);// [degree celsius]
// we have
m_dot_h_into_Ch = m_dot_o*Cp_o;// [W/degree celsius]
m_dot_c_into_Cc = m_dot_a*Cp_a;// [W/degree celsius]
// so the air is minimum fluid
C = m_dot_c_into_Cc/m_dot_h_into_Ch;
// the effectiveness is
E = (T2-T1)/(To-T1);
// now we may use either figure 10-16(page no.-546) or the analytical relation from table 10-4(page
```

```
   no.-543) to obtain NTU.
// for this problem we choose to use the table
NTU = -(1+C^(2))^(-1/2)*log((2/E-1-C-(1+C^2)^(1/2))
   /(2/E-1-C+(1+C^2)^(1/2)));
// now, we calcuate the area as
A = NTU*m_dot_c_into_Cc/U;// [square meter]
printf("area required for the heat exchanger is %f
   square meter",A);
```

Scilab code Exa 10.14 ammonia condenser

```
clear;
clc;
printf("\t\t\tExample Number 10.14\n\n\n");
// ammonia condenser
// Example 10.14 (page no.-552-553)
// solution

Ta = 50;// [degree celsius] temperature of entering
   ammonia vapour
Tw1 = 20;// [degree celsius] temperature of entering
    water
q = 200;// [kW] total heat transfer required
U = 1;// [kW/square meter degree celsius] overall
   heat transfer coefficient
Tw2 = 40;// [degree celsius] temperature of exiting
   water
Cw = 4.18;// [kJ/kg degree celsius] specific heat of
    water
// the mass flow can be calculated from the heat
   transfer with
m_dot_w = q/(Cw*(Tw2-Tw1));// [kg/s]
// because this is the condenser the water is the
   minimum fluid and
C_min = m_dot_w*Cw;// [kW/degree celsius]
```

```
// the value of NTU is obtained from the last entry
   of table 10-4(page no.-543), with
E = 0.6;// effectiveness
NTU = -log(1-E);
// so that area is calculated as
A = C_min*NTU/U;// [square meter]
// when the flow rate is reduced in half the new
   value of NTU is
NTU1 = U*A/(C_min/2);
// and the effectiveness is computed from the last
   entry of table 10-3(page no.-543):
E1 = 1-exp(-NTU1);
// the new water temperature difference is computed
   as
dT_w = E1*(Ta-Tw1);// [degree celsius]
// so that the heat transfer is
q1 = C_min*dT_w/2;// [kW]
printf("the area to achieve a heat exchanger
   effectiveness of 60%% with an exit water
   temperature of 40 degree celsius is %f square
   meter",A);
printf("\n\n by reducing the flow rate we have
   lowered the heat transfer by %d percent",(q-q1)
   *100/q);
```

Scilab code Exa 10.15 crossflow exchanger as energy conservation device

```
clear;
clc;
printf("\t\t\tExample Number 10.15\n\n\n");
// crossflow exchanger as energy conservation device
// Example 10.15 (page no.-553-555)
// solution

q = 210000;// [W] heat to be removed from
```

```
atmospheric air
m_dot_h = 1200/60;// [kg/s] hot air flow rate
m_dot_c = m_dot_h;// [kg/s] cold air flow rate
Ta1 = 25;// [degree celsius] atmospheric air temperature
Ta2 = 0;// [degree celsius] temperature of air entering from out-door conditions
U = 30;// [W/m degree celsius] overall heat transfer coefficient
Cp = 1005;// [J/kg degree celsius] specific heat of air

//*************calculation 1. the design value for the area of the heat exchanger **************//

// the hot and cold fluids have the same flow rate
// and
Ch = m_dot_h*Cp;// [W/degree celsius]
Cc = m_dot_c*Cp;// [W/degree cslsius]
Cmin_by_Cmax = 1;// for use in table 10-3(page no.-543)
// the energy balance gives q = Ch*dT_h = Cc*dT_c
// and
dT_h = q/Ch;// [degree celsius]
dT_c = q/Cc;// [degree celsius]
// the heat exchanger effectiveness is
E = dT_h/(Ta1-Ta2);
// consulting table 10-3(page no.-543) for a cross flow exchanger with both fluids unmixed, and inserting the value
C = 1;
// we have
deff('[y] = f(N)','y = E-1+exp(N^(0.22)*(exp(-N^(0.78))-1))');
N = fsolve(1,f);
// solving above to get the value of NTU
// area is
A = N*Ch/U;// [square meter]
```

```
printf("the design value for the area of heat
    exchanger is %f square meter",A);

//*************calculation 2. the percent reduction
    in heat transfer rate if the flow rate is reduced
     by 50% while keeping the inlet temperatures and
      value of U constant *****************//

// we now examine the effect of reducing the flow
    rate by half, while keeping the inlet
    temperatures and value of U the same.
// note that the flow rate of both fluids is reduced
     because they are physically the same fluid. this
     means that the value of Cmin_by_Cmax will remain
     the same at a value of 1.0.
// the new value of Cmin is
Cmin = Cc/2;// [W/degree celsius]
// so that NTU is
N = U*A/Cmin;
// equation (b) may be used for the calculation of
    effectiveness
E = 1-exp(N^(0.22)*(exp(-N^(0.78))-1));
// the temperature difference for each fluid is then
dT = E*(Ta1-Ta2);// [degree celsius]
// the resulting heat transfer is then
q_dot = m_dot_c*Cp*dT/2;// [W]
printf("\n\nthe percent reduction in heat transfer
    rate if the flow rate is reduced by 50%% is %f "
    ,(q-q_dot)*100/q);

//*************calculation 3. the percent reduction
    in heat transfer rate if the flow rate is reduced
     by 50% and the value of U varies as mass flow to
       the 0.8 power, with the same inlet temperature
     conditions

// finally, we examine the effect of reducing the
    flow rate by 50 percent coupled with reduction in
```

```
overall heat−transfer coefficient under the
   assumption that U varies as m_dot^(0.8) or,
   correspondingly, as Cmin^(0.8)
// still keeping the area constant, we would find
   that NTU varies as N = U*A/Cmin ~ C^(0.8)*C^(−1)
   = C^(−0.2)
// our new value of N under these conditions would
   be
N1 = 0.8*(Cmin/Cc)^(-0.2);
// inserting this value in equation (b) above for
   the effectiveness
E1 = 1-exp(N1^(0.22)*(exp(-N1^(0.78))-1));
// the corresponding temperature difference in each
   fluid is
dT = E1*(Ta1-Ta2);// [degree celsius]
// the heat transfer is calculated as
q1 = Cmin*dT;// [W]
printf("\n\n the percent reduction in heat transfer
   is %f ",(q-q1)*100/q);
```

Scilab code Exa 10.16 heat transfer coefficient in compact exchanger

```
clear;
clc;
printf("\t\t\tExample Number 10.16\n\n\n");
// heat−transfer coefficient in compact exchanger
// Example 10.16 (page no.−556−557)
// solution

p = 101325;// [Pa] pressure of air
T = 300;// [K] temperature of entering air
u = 15;// [m/s] velocity of air
// we obtain the air properties from table A−5(page
   no.−607)
rho = 1.1774;// [kg/cubic meter] density of air
```

```
Cp = 1005.7;// [J/kg degree celsius] specific heat
    of air
mu = 1.983*10^(-5);// [kg/m s] viscosity of air
Pr = 0.708;// prandtl number
// from figure 10-19(page no.-557) we have
Ac_by_A = 0.697;
sigma = Ac_by_A;
Dh = 3.597*10^(-3);// [m]
// the mass velocity is thus
G = rho*u/sigma;// [kg/square meter s]
// and the reynolds number is
Re = Dh*G/mu;
// from figure 10-19(page no.-557) we can read
St_into_Pr_exp_2_by_3 = 0.0036;
// and the heat transfer coefficient is
h = St_into_Pr_exp_2_by_3*G*Cp*(Pr)^(-2/3);// [W/
    square meter degree celsius]
printf("heat-transfer coefficient is %f W/square
    meter degree celsius",h);
```

Scilab code Exa 10.17 transient response of thermal energy storage system

```
clear;
clc;
printf("\t\t\tExample Number 10.17\n\n\n");
// transient response of thermal-energy storage
    system
// Example 10.17 (page no.-559-562)
// solution

Rinf = 0.176;// [degree celsius square meter/W]
    overall R value of material
A = 2.25;// [square meter] inlet flow area
l = 3;// [m] rock bed length
```

```
// properties of the rock are:
rho_r = 1281.4;// [kg/cubic meter]
Cr = 0.880;// [kJ/kg degree celsius]
kr = 0.87;// [W/m degree celsius]
Ti = 5;// [degree celsius] initial temperature of
   rock bed
Ta = 40;// [degree celsius] air temperature
Tinf = Ta;// [degree celsius]
p = 101.325;// [kPa] pressure of air
Ts = 5;// [degree celsius] surrounding temperature
v1 = 0.3;// [m/s] inlet velocity 1
v2 = 0.9;// [m/s] inlet velocity 2
Cpa = 1.004;// [kJ/kg degree celsius]
R = 0.287;// [kJ/kg K] universal gas constant
// it can be seen that the axial energy conduction
   is small compared to the mass energy transport.
// for a 35 degree celsius temperature difference
   over a 0.6 length
dx = 1/5;// [m]
q_cond = kr*A*(Ta-Ti)/dx;// [W]
                                                     (a
   )
// the density of air at 40 degree celsius
rho_a = p/(R*(Ta+273));// [kg/cubic meter]
                                            (b)
// and the mass flow rate at 0.3 m/s is
mdot_a = rho_a*A*v1;// [kg/s]

   (c)
// the corresponding energy transport for a
   temperature difference of 35 degree celsius is
q = mdot_a*Cpa*(Ta-Ti);// [kW]

   (d)
// and this is much larger than the value in
   equation (a).
// we now write an energy balance for one of the
   axial nodes as
```

```
// energy transported in - energy transported out - energy lost to surroundings = rate of energy accumulation of node
// or mdot_a*Cpa*(Tm_o^(t)-Tm^(t)) - (Tm^(t)-Tinf)*P*dx/Rinf = rho_r*Cr*dVr*(Tm^(t+1)-Tm^(t))/dt                    (e)
// where the exit temperature from node m is assumed to be the rock temperatre of that node(Tm^(t)). equation (e) may be solved to give
// Tm^(t+1) = F*mdot_a*Cpa*Tm_o^(t) + [1-F*(mdot_a*Cpa-P*dx/Rinf)]*Tm^(t) + F*P*dx*Tinf/Rinf                    (f)
// where
//          F = dt/(rho_r*Cr*dVr)
// here P is perimeter and dx is the increment.
P = 4*1.5;// [m]
// the stability requirement is such that the coefficient on the Tm^(t) terms cannot be negative. using dx = 0.6m, we find that the maximum value of
dx = 0.6;// [m]
Fmax = 6.4495*10^(-4);
// which yields a maximum time increment of
tmax = 0.54176;// [h]
// with a velocity of 0.9 m/s the maximum time increment for stability is
tmax_v2 = 0.1922;// [h]
// for the calculations we select the following values of dt with the resultant values of F:

// for v1
dt1 = 0.2;// [h]
F1 = 2.38095*10^(-4);
// for v2
dt2 = 0.1;// [h]
F2 = 1.190476*10^(-4);

// with the appropriate properties and these values
```

```
   inserted into equation(f) there results
// for v1
// Tm^(t+1) = F1*mdot_a*Cpa*Tm_o^(t) + [1-F1*(mdot_a
   *Cpa+P*dx/Rinf)]*Tm^(t) + F1*P*dx*Tinf/Rinf
                          (g)
// for v2
// Tm^(t+1) = F2*mdot_a*Cpa*Tm_o^(t) + [1-F2*(mdot_a
   *Cpa+P*dx/Rinf)]*Tm^(t) + F2*P*dx*Tinf/Rinf
                          (h)

// the energy storage relative to 5 degree celsius
   can then be calculated from
E_t = 0;
i = 1;
T1 = 40;
T2 = 5;
T3 = 5;
T4 = 5;
T5 = 5;
    for i = 1:100
    T2 = (F2*mdot_a*Cpa*1000*T1 + [1-F2*(mdot_a*Cpa
       *1000-P*dx/Rinf)]*T2 + F2*P*dx*Tinf/Rinf);
    T3 = (F2*mdot_a*Cpa*1000*T2 + [1-F2*(mdot_a*Cpa
       *1000-P*dx/Rinf)]*T3 + F2*P*dx*Tinf/Rinf);
    T4 = (F2*mdot_a*Cpa*1000*T3 + [1-F2*(mdot_a*Cpa
       *1000-P*dx/Rinf)]*T4 + F2*P*dx*Tinf/Rinf);
    T5 = (F2*mdot_a*Cpa*1000*T4 + [1-F2*(mdot_a*Cpa
       *1000-P*dx/Rinf)]*T5 + F2*P*dx*Tinf/Rinf);
    Temp(i,:) = [T1 T2 T3 T4 T5];
    E_t = (dt1/F1)*[(T1-5)+(T2-5)+(T3-5)+(T4-5)+(T5
       -5)];
    val(i) = i;
    val1(i) = E_t;
    end

E_t = 0;
i = 1;
T1 = 40;
```

```
T2 = 5;
T3 = 5;
T4 = 5;
T5 = 5;
    for i = 1:100
    T2 = (F1*mdot_a*Cpa*1000*T1 + [1-F1*(mdot_a*Cpa
       *1000-P*dx/Rinf)]*T2 + F1*P*dx*Tinf/Rinf);
    T3 = (F1*mdot_a*Cpa*1000*T2 + [1-F1*(mdot_a*Cpa
       *1000-P*dx/Rinf)]*T3 + F1*P*dx*Tinf/Rinf);
    T4 = (F1*mdot_a*Cpa*1000*T3 + [1-F1*(mdot_a*Cpa
       *1000-P*dx/Rinf)]*T4 + F1*P*dx*Tinf/Rinf);
    T5 = (F1*mdot_a*Cpa*1000*T4 + [1-F1*(mdot_a*Cpa
       *1000-P*dx/Rinf)]*T5 + F1*P*dx*Tinf/Rinf);
    Temp(i,:) = [T1 T2 T3 T4 T5];
    E_t = (dt1/F1)*[(T1-5)+(T2-5)+(T3-5)+(T4-5)+(T5
       -5)];
    val2(i) = i;
    val3(i) = E_t;
    end
plot(val,val1,val2,val3);
legend("v = 0.3m/s","v = 0.9m/s");
xlabel("time(h)");
ylabel("E(t) kJ ");
printf("the result of the calculations are shown in
   the accompanying figure");
```

Scilab code Exa 10.18 variable properties analysis of a duct heater

```
clear;
clc;
printf("\t\t\tExample Number 10.18\n\n\n");
// variable-properties analysis of a duct heater
// Example 10.18 (page no.-562-564)
// solution

```

```
d = 0.3;// [m] diameter of duct
Tma = 700;// [K] temperature of hot air
E = 0.6;// emissivity of outside duct surface
Tinf = 20+273;// [K] room temperature
// air properties at 700 K
rho = 0.5030;// [kg/cubic meter] density of air
mu = 3.332*10^(-5);// [kg/m s] viscosity of air
k = 0.05230;// [W/m degree celsius] heat transfer
    coefficient
Pr = 0.684;// prandtl no. of air
A = %pi*d^(2)/4;// [square meter] area of duct
sigma = 5.669*10^(-8);// [W/square meter K^(4)]
P = %pi*d;// [m]
Cp = 1083.5;// [J/kg degree celsius]
// this is a problem where a numerical solution must
     be employed.
// we choose a typical section of the duct with
    length dx and perimeter P as shown inn figure
    example 10-18A(page no.-562) and make the energy
    balances.
// we assume that resistance of the duct wall is
    negligible.
// inside the duct the energy balance is
// mdot_a*Cp*Tma = hi*P*dx*(Tma-Tmw)+mdot_a*Cp*
    Tm_po_a                    (a)
// where hi is the convection heat transfer
    coefficient on the inside which may be calculated
     from(the flow is turbulent)
// Nu = hi*d/k = 0.023*Re_d^(0.8)*Pr^(0.3)
                                    (b)
// with the properties evaluated at the bulk
    temperature of air(Tma). the energy balance for
    the heat flow through the wall is
// qconv_i = qconv_o+qrad_o
// or, by using convection coefficients and
    radiation terms per unit area,
// hi*(Tma-Tmw) = hc*(Tmw-Tinf)+sigma*E*(Tmw^(4)-
    Tinf^(4))                  (c)
```

```
// where the outside convection coefficient can be calculated from the free convection relation
// hc = 0.27*((Tmw-Tinf)/d)^(1/4)                                        (d)
// inserting this relation in equation (c) gives
// hi*(Tma-Tmw) = 0.27*(Tmw-Tinf)^(5/4)/d^(1/4)+sigma*E*(Tmw^(4)-Tinf^(4))          (e)
// equation (a) may be solved for Tm_po_a to give
// Tm_po_a = (1-hi*P*dx/(mdot_a*Cp))_m*Tma + (hi*P*dx/(mdot_a*Cp))_m*Tmw          (f)

// for
x=180;
mdot_a = [0.14 0.45 0.68];// [kg/s]
for i = 1:3

v = mdot_a(i)/(A*rho);// [m/s]
Re_d = d*v*rho/mu;
hi = k*0.023*Re_d^(0.8)*Pr^(0.3)/d;// [W/square meter degree celsius]


for dx = 1:1:179
    for Tmw = 295:1:715
        Z = (hi/dx)*(Tma-Tmw)-0.27*(Tmw-Tinf)^(5/4)/d^(1/4)-sigma*E*(Tmw^(4)-Tinf^(4));
        if (Z>0 & Z<40) then
            Tmw_new = Tmw;
        end
    end
    for Tm_po_a = 275:1:715
        X = Tm_po_a-(1-(hi/dx)*P*dx/(mdot_a(i)*Cp))*Tmw_new + ((hi/dx)*P*dx/(mdot_a(i)*Cp))*Tmw_new;
        if (X>0 & X<5) then
            Tm_po_a_new = Tm_po_a;
        end
    end
```

```
    q_by_A = (hi/dx)*(Tma-Tmw_new);// [W/square meter]
    val1(dx,i) = q_by_A;
    val(dx) = dx;
    val2(dx,i) = Tmw_new;
    val3(dx,i) = Tm_po_a_new;
end
end
scf(1);
plot(val,val1(:,1),val,val1(:,2),val,val1(:,3));
legend("mdot_a=0.14","mdot_a=0.45","mdot_a=0.68");
xlabel("Duct Length x,m");
ylabel("Local Heat Flux q / A,W / m^2");
xgrid();
title("Heat Flux");

scf(2);
plot(val,val2(:,1),val,val2(:,2),val,val2(:,3));
legend("Tw=0.14","Tw=0.45","Tw=0.68");
xlabel("Duct Length x,m");
ylabel("Local Wall Temperature Tw K");
xgrid();
title("Temperature Profile");

scf(3);
plot(val,val3(:,1),val,val3(:,2),val,val3(:,3));
legend("Ta=0.14","Ta=0.45","Ta=0.68");
xlabel("Duct Length x,m");
ylabel("Local Air Temperature Ta K");
xgrid();
title("Temperature Profile");
printf("plots are shown as :");
```

Chapter 11

Mass Transfer

Scilab code Exa 11.1 diffusion coefficient for co2

```
clear;
clc;
printf("\t\t\tExample Number 11.1\n\n\n");
// diffusion coefficient for co2
// Example 11.1(page no.-583)
// solution

T = 25+273.15;// [K] temperature of air
Vco2 = 34.0;// molecular volume of co2
Vair = 29.0;// molecular volume of air
Mco2 = 44;// molecular weight of co2
Mair = 28.9;// molecular weight of air
P = 1.01325*10^(5);// [Pa] atmospheric pressure
// using equation (11-2)
D = 435.7*T^(1.5)*(((1/Mco2)+(1/Mair))^(1/2))/(P*(
    Vco2^(1/3)+Vair^(1/3))^(2));
printf("value of diffusion coefficient for co2 in
    air is %f square centimeter/s",D);
```

Scilab code Exa 11.2 diffusion coefficient for co2

```
clear;
clc;
printf("\t\t\tExample Number 11.2\n\n\n");
// diffusion coefficient for co2
// Example 11.2(page no.-586-587)
// solution

T = 25+273.15;// [K] temperature of air
p = 1.01325*10^(5);// [Pa] atmospheric pressure
pw1 = 3166.7618901;// [Pa] partial pressure at the
    bottom of test tube
pw2 = 0;// [Pa] partial pressure at the top of test
    tube
pa1 = p-pw1;// [Pa]
pa2 = p-pw2;// [Pa]
D = .256*10^(-4);// [square meter/s] diffusion
    coefficient
Mw = 18;// [g] molecular weight of water
A = 22*((5*10^(-3))^(2))/7;// [square meter] area of
    test tube
R = 8314;// [J/mol K]gas constant
// using equation(11-16)
mw = (D*p*Mw*A/(R*T*0.15))*log(pa2/pa1);// mass flow
    rate
printf(" mass flow rate is %e kg/s",mw);
```

Scilab code Exa 11.3 Wet bulb temperature

```
clear;
clc;
printf("\t\t\tExample Number 11.3\n\n\n");
// Wet-bulb temperature
// Example 11.3(page no.-590-591)
```

```
// solution

Pg = 2107;// [Pa] from steam table at 18.3 degree
    celcius
Pw = Pg*18;// [Pa]
Rw = 8315;// [J/mol K] gas constant
Tw = 273.15+18.3;// [K]
RHOw = Pw/(Rw*Tw);// [kg/cubic meter]
Cw = RHOw;// [kg/cubic meter]
RHOinf = 0;// since the free stream is dry air
Cinf = 0;
P = 1.01325*10^(5);// [Pa]
R = 287;// [ J /kg K ]
T = Tw;// [K]
RHO = P/(R*T);// [kg/cubic meter]
Cp = 1004;// [J/kg degree celsius]
Le = 0.845;
Hfg = 2.456*10^(6);// [J/kg]
// now using equation(11-31)
Tinf = (((Cw-Cinf)*Hfg)/(RHO*Cp*(Le^(2/3))))+Tw;// [
   K]
Tin = Tinf-273.15;// [degree celsius]
printf("temperature of dry air is %f degree celsius"
    ,Tin);
printf("\n\n these calculations are now recalculated
     the density at the arithmetic-average
    temperature between wall and free-stream
    conditions");
printf("\n\n with this adjustments these results are
     RHO = 1.143 kg/m^(3) and Tinf = 55.8 degree
    celcius");
```

Scilab code Exa 11.4 relative humidity of air stream

```
clear;
```

```
clc;
printf("\t\t\tExample Number 11.4\n\n\n");
// relative humidity of air stream
// Example 11.4(page no.-591)
// solution

// these data were taken from previous example
Rho = 1.212;// [kg/cubic meter]
Cp = 1004;// [J/kg]
Le = 0.845;
Tw = 18.3;// [degree celsius]
Tinf = 32.2;// [degree celsius]
Rhow = 0.015666;// [kg/cubic meter]
Cw = Rhow;// [kg/cubic meter]
Hfg = 2.456*10^(6);// [J/kg]
// we use eqn 11-31
Cinf = Cw-(Rho*Cp*Le^(2/3)*(Tinf-Tw)/Hfg);// [kg/cubic meter]
Rhoinf = Cinf;// [kg/cubic meter]
Rhog = 0.0342;// [kg/cubic meter]
RH = (Rhoinf/Rhog)*100;
printf(" relative humidity is therefore %f percentage",RH);
```

Scilab code Exa 11.5 water evaporation rate

```
clear;
clc;
printf("\t\t\tExample Number 11.5\n\n\n");
// water evaporation rate
// Example 11.5(page no.-593-594)
// solution

Ta = 38+273;// [K] temperature of atmospheric air
RH = 0.30;// relative humidity
```

```
u = 10;// [mi/h] mean wind speed
R = 0.287;// universal gas constant
Dw = 0.256*10^(-4);// [square meter/s] from table A
    -8(page no.-610)
rho_w = 1000;// [kg/cubic meter]
// for this calculation we can make use of equation
    (11-36). from thermodynamic steam tables
p_g = 6.545;// [kPa] at 38 degree celsius
p_s = p_g;// [kPa]
p_w = RH*p_s;// [kPa]
p_s = 1.933;// [in Hg]
p_w = 0.580;// [in Hg]
// also
u_bar = u*24;// [mi/day]
// equation(11-36) yields, with the application of
    the 0.7 factor
E_lp = 0.7*(0.37+0.0041*u_bar)*(p_s-p_w)^(0.88);// [
    in/day]
E_lp = E_lp*2.54/100;// [m/day]
// noting that standard pan has the diameter of 1.2m
    , we can use the figure to calculate the mass
    evaporation rate per unit area as
m_dot_w_by_A = E_lp*rho_w/24;// [kg/h square meter]
// as a matter of interest, we might calculate the
    molecular-diffusion rate of water vapour from
    equation(11-35), taking z1 as the 1.5m dimension
    above the standard pan.
z1 = 1.5;// [m]
// since    rho = p/(R*T)
// equation(11-35) can be written as
m_dot_w_by_A1 = 0.622*Dw*p_g*3600/(R*Ta*z1);// [kg/h
     square meter]
printf("evaporation rate on the land under these
    conditions is %e kg/h square meter",m_dot_w_by_A1
    );
```