Scilab Textbook Companion for Fundamentals Of Heat And Mass Transfer by F. P. Incropera, D. P. Dewitt, T. L. Bergman And A. S. Lavine[1]

Created by
Abhishek J Jain
B. Tech.
Chemical Engineering
Indian Institute of Technology (BHU) , Varanasi
College Teacher
Dr. Prakash Kotecha
Cross-Checked by

August 10, 2013

---

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

Book Description

Title: Fundamentals Of Heat And Mass Transfer

Author: F. P. Incropera, D. P. Dewitt, T. L. Bergman And A. S. Lavine

Publisher: Wiley India Pvt. Ltd., New Delhi

Edition: 6

Year: 2010

ISBN: 9788126527649

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 Heat Loss Through Wall

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 1.1    Page 5 ')//Example 1.1
// Find Wall Heat Loss - Problem of Pure Conduction Unidimensional Heat

L=.15;     //[m] - Thickness of conducting wall
delT = 1400 - 1150;   //[K] - Temperature Difference across the Wall
A=.5*1.2; //[m^2] - Cross sectional Area of wall = H*W
k=1.7;     //[W/m.k] - Thermal Conductivity of Wall Material

//Using Fourier's Law eq 1.2
Q = k*delT/L;     //[W/m^2] - Heat Flux

q = A*Q;            //[W] - Rate of Heat Transfer

```

```
printf("\n \n Heat Loss through the Wall = %.2f W",q);
//END
```

Scilab code Exa 1.2 Surface Emissive Power and Irradiation

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 1.2   Page 11 \n')// Example 1.2
// Find a) Emissive Power & Irradiation b)Total Heat Loss per unit length

d=.07;     //[m] - Outside Diameter of Pipe
Ts = 200+273.15;   //[K] - Surface Temperature of Steam
Tsurr = 25+273.15; //[K] - Temperature outside the pipe
e=.8; //  Emissivity of Surface
h=15;     //[W/m^2.k] - Thermal Convectivity from surface to air
stfncnstt=5.67*10^(-8);     // [W/m^2.K^4] - Stefan Boltzmann Constant
//Using Eq 1.5
E = e*stfncnstt*Ts^4;     //[W/m^2] - Emissive Power
G = stfncnstt*Tsurr^4;     //[W/m^2] - Irradiation falling on surface

printf("\n (a) Surface Emissive Power = %.2f W/m^2",E);
printf("\n     Irradiation Falling on Surface = %.2f W/m^2",G);

//Using Eq 1.10 Total Rate of Heat Transfer Q  = Q
```

```
by convection + Q by radiation
q = h*(%pi*d)*(Ts-Tsurr)+e*(%pi*d)*stfncnstt*(Ts^4-Tsurr^4);            //[W]

printf("\n\n (b) Total Heat Loss per unit Length of Pipe= %.2f W",q);
//END
```

Scilab code Exa 1.3 Theoretical Problem

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 1.3    Page 18 \n')// Example 1.3
//Theoretical Problem

printf('\n The given example is theoretical and does not involve any numerical computation')

//End
```

Scilab code Exa 1.4 Coolant Fluid Velocity

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 1.4    Page 20 \n')// Example 1.4
// Find Velocity of Coolant Fluid

Ts = 56.4+273.15;   //[K] - Surface Temperature of Steam
```

```
Tsurr = 25+273.15; //[K] - Temperature of Surroundings
e=.88; //  Emissivity of Surface

//As h=(10.9*V^.8) [W/m^2.k] - Thermal Convectivity from surface to air
stfncnstt=5.67*10^(-8);     // [W/m^2.K^4] - Stefan Boltzmann Constant

A=2*.05*.05;     // [m^2] Area for Heat transfer i.e. both surfaces

E = 11.25;     //[W] Net heat to be removed by cooling air

Qrad = e*stfncnstt*A*(Ts^4-Tsurr^4);

//Using Eq 1.10 Total Rate of Heat Transfer Q  = Q by convection + Q by radiation
Qconv = E - Qrad;//[W]

//As Qconv = h*A*(Ts-Tsurr) & h=10.9 Ws^(.8)/m^(-.8) K.V^(.8)

V = [Qconv/(10.9*A*(Ts-Tsurr))]^(1/0.8);

printf("\n\n Velocity of Cooling Air flowing= %.2f m/s",V);
//END
```

Scilab code Exa 1.5 Theoretical Problem

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
```

```
Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 1.5    Page 23 \n')// Example 1.5
//Theoretical Problem

printf('\n The given example is theoretical and does not involve any numerical computation')

//End
```

Scilab code Exa 1.6 Human Body Heat Loss

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 1.6    Page 26 ')// Example 1.6
// Find Skin Temperature & Heat loss rate

A=1.8;     // [m^2] Area for Heat transfer i.e. both surfaces
Ti = 35+273;   //[K] - Inside Surface Temperature of Body
Tsurr = 297; //[K] - Temperature of surrounding
Tf = 297; //[K] - Temperature of Fluid Flow
e=.95; //  Emissivity of Surface
L=.003;     //[m] - Thickness of Skin
k=.3;  //  Effective Thermal Conductivity
h=2;     //[W/m^2.k] - Natural Thermal Convectivity from body to air
stfncnstt=5.67*10^(-8);     // [W/m^2.K^4] - Stefan Boltzmann Constant
//Using Eq 1.5

Tsa=305;             //[K]  Body Temperature Assumed

```

```
i=-1;
while(i==-1)
   hr = e*stfncnstt*(Tsa+Tsurr)*(Tsa^2+Tsurr^2);    //[W/m^2.K] - Radiative Heat transfer Coeff on assumption

   //Using Eq 1.8 & Eq 1.9 k(Ti-Ts)/L = h(Ts - Tf) + hr(Ts - Tsurr)
Ts = (k*Ti/L + (h+hr)*Tf)/(k/L +(h+hr));
   c=abs(Ts-Tsa);
   if(c<=0.0001)
     i=1;
     break;
   end
   Tsa=Ts;
end

q = k*A*(Ti-Ts)/L;          //[W]

printf("\n\n (I) In presence of Air")
printf("\n (a) Temperature of Skin = %.2f K",Ts);
printf("\n (b) Total Heat Loss = %.2f W",q);

//When person is in Water
h = 200;    //[W/m^2.k] - Thermal Convectivity from body to water
hr = 0;    // As Water is Opaque for Thermal Radiation
Ts = (k*Ti/L + (h+hr)*Tf)/(k/L +(h+hr));   //[K] Body Temperature
q = k*A*(Ti-Ts)/L;          //[W]
printf("\n\n (II) In presence of Water")
printf("\n (a) Temperature of Skin = %.2f K",Ts);
printf("\n (b) Total Heat Loss = %.2f W",q);

//END
```

Scilab code Exa 1.7 Cure Temperature

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 1.7    Page 30 \n')//Example 1.7
// (a) Cure Temperature for h = 15 W/m^2
// (b) Value of h for cure temp = 50 deg C

Tsurr = 30+273; //[K] - Temperature of surrounding
Tf = 20+273; //[K] - Temperature of Fluid Flow
e=.5; //  Emissivity of Surface
a = .8;    // Absorptivity of Surface
G = 2000;    //[W/m^2] - Irradiation falling on surface
h=15;    //[W/m^2.k] - Thermal Convectivity from plate to air
stfncnstt=5.67*10^(-8);    // [W/m^2.K^4] - Stefan Boltzmann Constant
T=375;    //[K] Value initially assumed for trial-error approach
//Using Eq 1.3a & 1.7 and trial-and error approach of Newton Raphson
while(1>0)
f=((a*G)-(h*(T-Tf)+e*stfncnstt*(T^4 - Tsurr^4)));
fd=(-h*T-4*e*stfncnstt*T^3);
Tn=T-f/fd;
if(((a*G)-(h*(Tn-Tf)+e*stfncnstt*(Tn^4 - Tsurr^4)))<=.01)
   break;
end;
T=Tn;
end
```

```

printf("\n (a) Cure Temperature of Plate = %i degC\n",T-273);
//solution (b)
Treq=50+273;
function[T]=Tvalue(h)
    T=240;
    while(1>0)
        f=((a*G)-(h*(T-Tf)+e*stfncnstt*(T^4 - Tsurr^4)));
        fd=(-h*T-4*e*stfncnstt*T^3);
        Tn=T-f/fd;
        if(((a*G)-(h*(Tn-Tf)+e*stfncnstt*(Tn^4 - Tsurr^4)))<=.01)
            break;
        end;
        T=Tn;
    end
    funcprot(0)
endfunction

h = [2:.5:100];
Tm = [1:1:197];
for i=1:1:197;
    Tm(i)=Tvalue(h(i));
end

T=Treq;
hnew=((a*G)-(e*stfncnstt*(T^4 - Tsurr^4)))/(T-Tf);
clf()
xtitle("Graph Temp vs Convection Coeff", "h (W/m^2/K)", "T (degC)");
x=[0 hnew hnew];
y=[Treq-273 Treq-273 0];
plot(h,Tm-273,x,y);
legend("Plot","h at T = 50 degC");
printf("\n (b) Air flow must provide a convection of = %i W/m^2.K", hnew);
```

```
//END
```

Scilab code Exa 1.8 Theoretical Problem

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   1.8    Page 40 \n')// Example 1.8
//Theoretical Problem

printf('\n The given example is theoretical and does
    not involve any numerical computation')

//End
```

Chapter 2

Introduction to Conduction

Scilab code Exa 2.1 Thermal Diffusivity

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 2.1    Page 68 \n')//Example 2.1
// Find Value for Thermal Diffusivity

function a=alpha(p, Cp, k)
    a=k/(p*Cp); //[m^2/s]
    funcprot(0);
endfunction

//(a) Pure Aluminium at 300K
// From Appendix A, Table A.1

p = 2702;   //[Kg/m^3] - Density Of Material
Cp = 903; //[J/kg.K] - Specific heat of Material
k = 237;     //[W/m.k] - Thermal Conductivity of Material

printf("\n (a) Thermal Diffuisivity of Pure
```

```
Aluminium at 300K = %.2e m^2/s\n",alpha(p, Cp, k));

//(b) Pure Aluminium at 700K
// From Appendix A, Table A.1

p = 2702;  //[Kg/m^3] - Density Of Material
Cp = 1090; //[J/kg.K] - Specific heat of Material
k = 225;     //[W/m.k] - Thermal Conductivity of Material

printf("\n (b) Thermal Diffuisivity of Pure Aluminium at 700K = %.2e m^2/s\n",alpha(p, Cp, k));

//(c) Silicon Carbide at 1000K
// From Appendix A, Table A.2

p = 3160;  //[Kg/m^3] - Density Of Material
Cp = 1195; //[J/kg.K] - Specific heat of Material
k = 87;    //[W/m.k] - Thermal Conductivity of Material

printf("\n (c) Thermal Diffuisivity of Silicon Carbide at 1000K = %.2e m^2/s\n",alpha(p, Cp, k));

//(d) Paraffin at 300K
// From Appendix A, Table A.3

p = 900;  //[Kg/m^3] - Density Of Material
Cp = 2890; //[J/kg.K] - Specific heat of Material
k = .24;     //[W/m.k] - Thermal Conductivity of Material

printf("\n (d) Thermal Diffuisivity of Paraffin at 300K = %.2e m^2/s",alpha(p, Cp, k));
//END
```

Scilab code Exa 2.2 Non Uniform Temperature Distribution

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 2.2   Page 75 \n')//Example 2.2
// Analyze a Situation of Non-Uniform Temperature Distribution
//T(x) = a + bx +cx^2   T-degC & x-meter

a = 900;    //[degC]
b = -300;   //[degC/m]
c = -50;    //[degC/m^2]

q = 1000;    //[W/m^2.K] - Unifrom heat Generation
A = 10 ;      //[m^2]    - Wall Area
//Properties of Wall
p = 1600;    //[kg/m^3]  - Density
k = 40;      //[W/m]    - Thermal Conductivity
Cp = 4000;   //[J/kg.K]  - Specific Heat
L = 1;      //[m]    - Length of wall

//(i) Rate of Heat Transfer entering the wall and leaving the wall
// From Eqn 2.1
// qin = -kA(dT/dx)|x=0 = -kA(b)

qin= - b*k*A;

// Similarly
// qout = -kA(dT/dx)|x=L = -kA(b+2cx)|x=L

qout= - k*A*(b+2*c*L);
```

```

printf("\n (i) Rate of Heat Transfer entering the wall = %i W \n      And leaving the wall = %i W \n", qin, qout);

//(ii) Rate of change Of Energy Storage in Wall E`st
// Applying Overall Energy Balance across the Wall
//E`st = E`in + E`g + E`out = qin + q`AL - qout
Est = qin + q*A*L - qout;

printf("\n (ii) Rate of change Of Energy Storage in Wall = %i W\n",Est);

//(iii) Time rate of Temperature change at x= 0, 0.25 and .5m
//Using Eqn 2.19
// T`= dT/dt = (k/p*Cp)*d(dT/dx)/dx + q`/p*Cp
//As d(dT/dx)/dx = d(b + 2cx)/dx = 2c - Independent of x
T = (k/(p*Cp))*(2*c)+ q/(p*Cp);
printf("\n (iii) Time rate of Temperature change independent of x = %f degC/s\n",T);

//END
```

Scilab code Exa 2.3 Theoretical Problem

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 2.3    Page 79 \n')// Example 2.3
//Theoretical Problem

printf('\n The given example is theoretical and does
```

```
    not involve any numerical computation')

//End
```

Chapter 3

One Dimensional Steady State Conduction

Scilab code Exa 3.1 Human Heat Loss

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 3.1    Page 104 \n') //Example 3.1
// Find Skin Temperature & Aerogel Insulation Thickness

A=1.8;     // [m^2] Area for Heat transfer i.e. both surfaces
Ti = 35+273;   //[K] - Inside Surface Temperature of Body
Tsurr = 10+273; //[K] - Temperature of surrounding
Tf = 283; //[K] - Temperature of Fluid Flow
e=.95; //  Emissivity of Surface
Lst=.003;     //[m] - Thickness of Skin
kst=.3;  //   [W/m.K] Effective Thermal Conductivity of Body
kins = .014;   //   [W/m.K] Effective Thermal
```

```
Conductivity of Aerogel Insulation
hr = 5.9;     //[W/m^2.k] - Natural Thermal Convectivity from body to air
stfncnstt=5.67*10^(-8);     // [W/m^2.K^4] - Stefan Boltzmann Constant
q = 100;           //[W] Given Heat rate

//Using Conducion Basic Eq 3.19
Rtot = (Ti-Tsurr)/q;
//Also
//Rtot=Lst/(kst*A) + Lins/(kins*A)+(h*A + hr*A)^-1
//Rtot = 1/A*(Lst/kst + Lins/kins +(1/(h+hr)))

//Thus
//For Air,
h=2;     //[W/m^2.k] - Natural Thermal Convectivity from body to air
Lins1 =  kins * (A*Rtot - Lst/kst - 1/(h+hr));

//For Water,
h=200;     //[W/m^2.k] - Natural Thermal Convectivity  from body to air
Lins2 =  kins * (A*Rtot - Lst/kst - 1/(h+hr));

Tsa=305;            //[K]  Body Temperature Assumed

//Temperature of Skin is same in both cases as Heat Rate is same
//q=(kst*A*(Ti-Ts))/Lst
Ts = Ti - q*Lst/(kst*A);

//Also from eqn of effective resistance Rtot F
printf("\n\n (I) In presence of Air, Insulation Thickness = %.1f mm",Lins1*1000)

printf("\n (II) In presence of Water, Insulation Thickness = %.1f mm",Lins2*1000);
printf("\n\n  Temperature of Skin = %.2f degC",Ts
```

```
-273);
//END
```

Scilab code Exa 3.2 Chip Operating Temperature

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   3.2    Page 107 \n'); //Example 3.2
// Chip Operating Temperature

Tf = 25+273; //[K] - Temperature of Fluid Flow

L=.008;     //[m] - Thickness of Aluminium
k=239;  //  [W/m.K]  Effective Thermal Conductivity
   of Aluminium
Rc=.9*10^-4;     //[K.m^2/W]     Maximum permeasible
   Resistane of Epoxy Joint
q=10^4;     //[W/m^2]     Heat dissipated by Chip
h=100;     //[W/m^2.k] - Thermal Convectivity from
   chip to air

//Temperature of Chip
//q=(Tc-Tf)/(1/h)+(Tc-Tf)/(Rc+(L/k)+(1/h))

Tc = Tf + q*(h+1/(Rc+(L/k)+(1/h)))^-1;

printf("\n\n Temperature of Chip = %.2f degC",Tc
   -273);
printf("\n Chip will Work well below its maximum
   allowable Temperature ie 85 degC")
//END
```

Scilab code Exa 3.3 Carbon Nanotube

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 3.3   Page 109 \n'); //Example 3.3
// Find Thermal conductivity of Carbon Nanotube

D = 14 * 10^-9;     // [m] Dia of Nanotube
s = 5*10^-6;        // [m] Distance between the islands
Ts = 308.4;         //[K] Temp of sensing island
Tsurr = 300;        //[K] Temp of surrounding
q = 11.3*10^-6;     //[W] Total Rate of Heat flow

//Dimension of platinum line
wpt = 10^-6;        //[m]
tpt = 0.2*10^-6;    //[m]
Lpt = 250*10^-6;    //[m]
//Dimension of Silicon nitride line
wsn = 3*10^-6;        //[m]
tsn = 0.5*10^-6;    //[m]
Lsn = 250*10^-6;    //[m]
//From Table A.1 Platinum Temp Assumed = 325K
kpt = 71.6;     //[W/m.K]
//From Table A.2, Silicon Nitride Temp Assumed = 325 K
ksn = 15.5;     //[W/m.K]

Apt = wpt*tpt;         //Cross sectional area of platinum support beam
Asn = wsn*tsn-Apt;     //Cross sectional area of Silicon Nitride support beam
```

```
Acn = %pi*D^2/4;        //Cross sectional Area of Carbon nanotube

Rtsupp = [kpt*Apt/Lpt + ksn*Asn/Lsn]^-1;     //[K/W] Thermal Resistance of each support

qs = 2*(Ts-Tsurr)/Rtsupp;     //[W] Heat loss through sensing island support
qh = q - qs;     //[W] Heat loss through heating island support

Th = Tsurr + qh*Rtsupp/2;     //[K] Temp of Heating island

//For portion Through Carbon Nanotube
//qs = (Th-Ts)/(s/(kcn*Acn));

kcn = qs*s/(Acn*(Th-Ts));

printf("\n\n Thermal Conductivity of Carbon nanotube = %.2f W/m.K",kcn);
//END
```

Scilab code Exa 3.4 Conical Section

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 3.4    Page 113 \n'); //Example 3.4
// Temperature Distribution And Heat rate

a = 0.25;
x1 = .05;       //[m] Distance of smaller end
x2 = .25;       //[m] Distance of larger end
```

```
T1 = 400;       //[K] Temperature of smaller end
T2 = 600;       //[K] Temperature of larger end
k = 3.46;     //[W/m.K] From Table A.2, Pyroceram at Temp 285K

x = linspace(0.05,.25,100);
T=(T1 + (T1-T2)*[(x^-1 - x1^-1)/(x1^-1 - x2^-1)]);
clf();
plot(x,T);
xtitle(" Temp vs distance x", "x (m)", "T (K)");

qx = %pi*a^2*k*(T1-T2)/(4*[1/x1 - 1/x2]);      //[W]
printf("\n\n Heat Transfer rate = %.2f W",qx);
//END
```

Scilab code Exa 3.5 Critical Thickness

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 3.5    Page 119 \n'); //Example 3.5
// Critical Thickness

k = .055;     //[W/m.K] From Table A.3, Cellular glass at Temp 285K
h = 5;        //[W/m^2.K]
ri = 5*10^-3;       //[m]  radius of tube

rct = k/h;      // [m] Critical Thickness of Insulation for maximum Heat loss or minimum resistance

x = linspace(0,.07,100);
```

```
ycond=(2.30*log10((x+ri)/ri)/(2*%pi*k));
yconv=(2*%pi*(x+ri)*h)^-1;
ytot=yconv+ycond;
clf();
plot(x,ycond,x,yconv,x,ytot);
xtitle("Resistance vs Radii", "r-ri (m)", "R (m.K/W)");
legend ("Rcond", "Rconv", "Rtotal");

printf("\n\n Critical Radius is = %.3f m \n Heat transfer will increase with the addition of insulation up to a thickness of %.3f m",rct,rct-ri);
//END
```

Scilab code Exa 3.6 Spherical Composite

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 3.6    Page 122 \n'); //Example 3.6
// Heat conduction through Spherical Container

k = .0017;     //[W/m.K] From Table A.3, Silica Powder at Temp 300K
h = 5;         //[W/m^2.K]
r1 = 25*10^-2;       //[m]  Radius of sphere
r2 = .275;           //[m]  Radius including Insulation thickness

//Liquid Nitrogen Properties
T = 77;         //[K] Temperature
rho = 804;      //[kg/m^3] Density
hfg = 2*10^5;   //[J/kg] latent heat of vaporisation
```

```

//Air Properties
Tsurr = 300;    //[K] Temperature
h = 20         ;//[W/m^2.K]  convection coefficient

Rcond = (1/r1-1/r2)/(4*%pi*k);    //Using Eq 3.36
Rconv = 1/(h*4*%pi*r2^2);
q = (Tsurr-T)/(Rcond+Rconv);

printf("\n\n (a)Rate of Heat transfer to Liquid Nitrogen %.2f W",q);

//Using Energy Balance q - m*hfg = 0
m=q/hfg;    //[kg/s] mass of nirtogen lost per second
mc = m/rho*3600*24*10^3;
printf("\n\n (b)Mass rate of nitrogen boil off %.2f Litres/day",mc);
//END
```

Scilab code Exa 3.7 Composite Plane Wall

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 3.7   Page 129 \n'); //Example 3.7
// Composite Plane wall

Tsurr = 30+273;    //[K] Temperature of surrounding Water
h = 1000;      //[W/m^2.K] Heat Convection Coeff of Water
kb = 150;    //[W/m.K] Material B
Lb = .02;    //[m] Thickness Material B
```

```
ka = 75;     //[W/m.K] Material A
La = .05;     //[m] Thickness Material A
qa = 1.5*10^6;  //[W/m^3] Heat generation at wall A
qb = 0;  //[W/m^3] Heat generation at wall B

T2 = Tsurr + qa*La/h;

Rcondb = Lb/kb;
Rconv = 1/h;
T1 = Tsurr +(Rcondb + Rconv)*(qa*La);
//From Eqn 3.43
T0 = qa*La^2/(2*ka) + T1;

printf("\n\n (a) Inner Temperature of Composite To = %i degC \n (b) Outer Temperature of the Composite T2 = %i degC",T0-273,T2-273);
//END
```

Scilab code Exa 3.8 Theoretical Problem

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 3.8    Page 134 \n')// Example 3.8
//Theoretical Problem

printf('\n The given example is theoretical and does not involve any numerical computation')

//End
```

Scilab code Exa 3.9 Rod Fin Heat Transfer

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 3.9   Page 145 \n'); //Example 3.9
// Heat conduction through Rod

kc = 398;     //[W/m.K] From Table A.1, Copper at Temp 335K
kal = 180;     //[W/m.K] From Table A.1, Aluminium at Temp 335K
kst = 14;     //[W/m.K] From Table A.1, Stainless Steel at Temp 335K
h = 100;       //[W/m^2.K] Heat Convection Coeff of Air
Tsurr = 25+273;     //[K] Temperature of surrounding Air
D = 5*10^-3;     //[m] Dia of rod
To = 100+273.15;     //[K] Temp of opposite end of rod

//For infintely long fin m = h*P/(k*A)
mc = (4*h/(kc*D))^.5;
mal = (4*h/(kal*D))^.5;
mst = (4*h/(kst*D))^.5;
x = linspace(0,.300,100);
Tc = Tsurr + (To - Tsurr)*2.73^(-mc*x) - 273;
Tal = Tsurr + (To - Tsurr)*2.73^(-mal*x) -273;
Tst = Tsurr + (To - Tsurr)*2.73^(-mst*x) -273;
clf();
plot(x,Tc,x,Tal,x,Tst);
xtitle("Temp vs Distance", "x (m)", "T (degC)");
legend ("Cu", "2024 Al", "316 SS");

//Using eqn 3.80
qfc = (h*%pi*D*kc*%pi/4*D^2)^.5*(To-Tsurr);
qfal = (h*%pi*D*kal*%pi/4*D^2)^.5*(To-Tsurr);
qfst = (h*%pi*D*kst*%pi/4*D^2)^.5*(To-Tsurr);
```

```

printf("\n\n (a) Heat rate \n          For Copper = %.2f W \n          For Aluminium = %.2f W \n          For Stainless steel = %.2f W",qfc,qfal,qfst);

//Using eqn 3.76 for satisfactory approx
Linfc = 2.65/mc;
Linfal = 2.65/mal;
Linfst = 2.65/mst;

printf("\n\n (a) Rods may be assumed to be infinite Long if it is greater than equal to \n          For Copper = %.2f m \n          For Aluminium = %.2f m \n          For Stainless steel = %.2f m",Linfc,Linfal,Linfst);
//END
```

Scilab code Exa 3.10 Finned Cylinder Heat Transfer

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 3.10    Page 156 \n'); //Example 3.10
// Study of motorcycle finned cylinder

H = .15;     //[m] height
k = 186;      //[W/m.K] alumunium at 400K
h = 50;        //[W/m^2.K] Heat convection coefficient
Tsurr = 300;     //[K] Temperature of surrounding air
To = 500;      //[K] Temp inside

//Dimensions of Fin
N = 5;
t = .006;      //[m] Thickness
```

```
L = .020;      //[m] Length
r2c = .048;         //[m]
r1 = .025;             //[m]

Af = 2*%pi*(r2c^2-r1^2);
At = N*Af + 2*%pi*r1*(H-N*t);

//Using fig 3.19
nf = .95;

qt = h*At*[1-N*Af*(1-nf)/At]*(To-Tsurr);
qwo = h*(2*%pi*r1*H)*(To-Tsurr);

printf("\n\n  Heat Transfer Rate with the fins =%i W
    \n  Heat Transfer Rate without the fins =%i W \n
    Thus Increase in Heat transfer rate of %i W is
   observed with fins",qt,qwo,qt-qwo);
//END
```

Scilab code Exa 3.11 Study of Fuel Cell Fan System

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   3.11    Page 158 \n'); //Example 3.11
// Study of Fuel-cell fan system

Wc =.05;     //[m] width
H = .026;    //[m] height
tc = .006;   //[m] thickness of cell
V = 9.4;     //[m/sec] vel of cooling air
P = 9;       //[W] Power generated
C = 1000;     //[W/(m^3/s)] Ratio of fan power
   consumption to vol flow rate
```

```
k = 200;     //[W/m.K] alumunium
Tsurr = 25+273.15;     //[K] Temperature of surrounding air
Tc = 56.4+273.15;     //[K] Temp of fuel cell
Rtcy = 10^-3;          //[K/W]   Contact thermal resistance
tb = .002;             //[m] thickness of base of heat sink
Lc = .05;          //[m] length of fuel cell
//Dimensions of Fin
tf = .001;     //[m] Thickness
Lf = .008;      //[m] Length

Vf = V*[Wc*(H-tc)];     //[m^3/sec] Volumetric flow rate
Pnet = P - C*Vf;


P = 2*(Lc+tf);
Ac = Lc*tf;
N = 22;
a=(2*Wc - N*tf)/N;
h = 19.1;              ///[W/m^2.K]
q = 11.25;              //[W]
m = (h*P/(k*Ac))^.5;
Rtf = (h*P*k*Ac)^(-.5)/ tanh(m*Lf);
Rtc = Rtcy/(2*Lc*Wc);
Rtbase = tb/(2*k*Lc*Wc);
Rtb = 1/[h*(2*Wc-N*tf)*Lc];
Rtfn = Rtf/N;
Requiv = [Rtb^-1 + Rtfn^-1]^-1;
Rtot = Rtc + Rtbase + Requiv;

Tc2 = Tsurr +q*(Rtot);

printf("\n\n (a) Power consumed by fan is more than the generated power of fuel cell, and hence system cannot produce net power = %.2f W \n\n (b)
```

```
 Actual fuel cell Temp is close enough to %.1f degC for reducing the fan power consumption by half ie Pnet = %.1f W, we require 22 fins , 11 on top and 11 on bottom.",Pnet, Tc2-273, C*Vf/2);

//END
```

Scilab code Exa 3.12 Heat Loss From Body and Temp at Inner Surface

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 3.12    Page 163 \n'); //Example 3.12
// Heat loss from body & temp at inner surface

hair = 2;       //[W/m^2.K] Heat convection coefficient air
hwater = 200;       //[W/m^2.K] Heat convection coefficient water
hr = 5.9 ;      //[W/m^2.K] Heat radiation coefficient
Tsurr = 297;     //[K] Temperature of surrounding air
Tc = 37+273;     //[K] Temp inside
e = .95;
A = 1.8 ;        //[m^2] area
//Prop of blood
w = .0005 ;       //[s^-1] perfusion rate
pb = 1000;         //[kg/m^3] blood density
cb = 3600;         //[J/kg] specific heat
//Dimensions & properties of muscle & skin/fat
Lm = .03 ;        //[m]
Lsf = .003 ;      //[m]
km = .5 ;        //[W/m.K]
ksf = .3;         //[W/m.K]
```

```
q = 700;          //[W/m^3]   Metabolic heat generation rate

Rtotair = (Lsf/ksf + 1/(hair + hr))/A;
Rtotwater = (Lsf/ksf + 1/(hwater))/A;

m = (w*pb*cb/km)^.5;
Theta = -q/(w*pb*cb);

Tiair = (Tsurr*sinh(m*Lm) + km*A*m*Rtotair*[Theta + (Tc + q/(w*pb*cb))*cosh(m*Lm)])/(sinh(m*Lm)+km*A*m*Rtotair*cosh(m*Lm));
qair = (Tiair - Tsurr)/Rtotair;

Tiwater = (Tsurr*sinh(m*Lm) + km*A*m*Rtotwater*[Theta + (Tc + q/(w*pb*cb))*cosh(m*Lm)])/(sinh(m*Lm)+km*A*m*Rtotwater*cosh(m*Lm));
qwater = (Tiwater - Tsurr)/Rtotwater;

printf("\n\n For Air \n Temp excess Ti = %.1f degC and Heat loss rate =%.1f W \n\n For Water \n Temp excess Ti = %.1f degC and Heat loss rate =%.1f W",Tiair-273,qair,Tiwater-273,qwater);
//END
```

Chapter 4

Two Dimensional Steady State Conduction

Scilab code Exa 4.1 Thermal Resistance of Eccentric Wire

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 4.1    Page 211 \n'); //Example 4.1
// Thermal resistance of wire coating associated with peripheral variations in coating thickness

d = .005;          //[m] Diameter of wire
k = .35;           //[W/m.K] Thermal Conductivity
h = 15;            //[W/m^2.K] Total coeff with Convection n Radiation

rcr = k/h;          // [m] critical insulation radius
tcr = rcr - d/2;    // [m] critical insulation Thickness

Rtcond = 2.302*log10(rcr/(d/2))/(2*%pi*k);       //[K/W] Thermal resistance
```

```
//Using Table 4.1 Case 7
z = .5*tcr;
D=2*rcr;
Rtcond2D = (acosh((D^2 + d^2 - 4*z^2)/(2*D*d)))/(2*
    %pi*k);

printf("\n\n The reduction in thermal resistance of
    the insulation is %.2f K/W ", Rtcond-Rtcond2D);
//END
```

Scilab code Exa 4.2 Theoretical Problem

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
    Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
    4.2    Page 218 \n')// Example 4.2
//Theoretical Problem

printf('\n The given example is theoretical and does
     not involve any numerical computation')

//End
```

Scilab code Exa 4.3 Temperature Distribution in Column and Heat Rate per Unit Length

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
    Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
    4.3    Page 224 \n'); //Example 4.2
```

```
// Temperature Distribution and Heat rate per unit length

Ts = 500;           //[K] Temp of surface
Tsurr = 300;        //[K] Temp of surrounding Air
h = 10;             //[W/m^2.K] Heat Convection soefficient
//Support Column
delx = .25;         //[m]
dely = .25;         //[m]
k = 1;            //[W/m.K] From Table A.3, Fireclay Brick at T = 478K

//Applying Eqn 4.42 and 4.48
A = [-4 1 1 0 0 0 0 0;
     2 -4 0 1 0 0 0 0;
     1 0 -4 1 1 0 0 0;
     0 1 2 -4 0 1 0 0;
     0 0 1 0 -4 1 1 0;
     0 0 0 1 2 -4 0 1;
     0 0 0 0 2 0 -9 1;
     0 0 0 0 0 2 2 -9 ];

C = [-1000; -500; -500; 0; -500; 0; -2000; -1500 ];

T = inv(A)*C;

printf("\n Temp Distribution = ");
printf("\n    %.2f K ", T);

q = 2*h*[(delx/2)*(Ts-Tsurr)+delx*(T(7)-Tsurr)+delx*(T(8)-Tsurr)/2];
printf("\n\n Heat rate from column to the airstream %.1f W/m ", q);
//END
```

Scilab code Exa 4.4 Temperature Field of Channel and Rate of Heat Transfer

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   4.4   Page 230 \n'); //Example 4.4
// Temperature Field and Rate of Heat Transfer

//Operating Conditions

ho = 1000;          //[W/m^2.K] Heat Convection
   coefficient
hi = 200;           //[W/m^2.K] Heat Convection
   coefficient
Ti = 400;         //[K] Temp of Air
Tg = 1700;      //[K] Temp of Gas
h = 10 ;          //[W/m^2.K] Heat Convection
   coefficient

A = 2*6*10^-6 ;    //[m^2] Cross section of each
   Channel
x = .004 ;      //[m] Spacing between joints
t = .006;       //[m] Thickness
k = 25;          //[W/m.K] Thermal Conductivity of
   Blade
delx = .001 ;      //[m]
dely = .001 ;      //[m]

//Applying Eqn 4.42 and 4.48
A = [-(2+ho*delx/k) 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
   0 0 0 0;
     1 -2*(2+ho*delx/k) 1 0 0 0 0 2 0 0 0 0 0 0 0 0
```

```
       0 0 0 0 0;
     0 1 -2*(2+ho*delx/k) 1 0 0 0 0 2 0 0 0 0 0 0 0
       0 0 0 0 0;
     0 0 1 -2*(2+ho*delx/k) 1 0 0 0 0 2 0 0 0 0 0 0
       0 0 0 0 0;
     0 0 0 1 -2*(2+ho*delx/k) 1 0 0 0 0 2 0 0 0 0 0
       0 0 0 0 0;
     0 0 0 0 1 -(2+ho*delx/k) 0 0 0 0 0 1 0 0 0 0 0
       0 0 0 0;
     1 0 0 0 0 0 -4 2 0 0 0 0 1 0 0 0 0 0 0 0 0;
     0 1 0 0 0 0 1 -4 1 0 0 0 0 1 0 0 0 0 0 0 0;
     0 0 1 0 0 0 0 1 -4 1 0 0 0 0 1 0 0 0 0 0 0;
     0 0 0 1 0 0 0 0 1 -4 1 0 0 0 0 1 0 0 0 0 0;
     0 0 0 0 1 0 0 0 0 1 -4 1 0 0 0 0 1 0 0 0 0;
     0 0 0 0 0 1 0 0 0 0 2 -4 0 0 0 0 0 1 0 0 0;
     0 0 0 0 0 0 1 0 0 0 0 0 -4 2 0 0 0 0 1 0 0;
     0 0 0 0 0 0 0 1 0 0 0 0 1 -4 1 0 0 0 0 1 0;
     0 0 0 0 0 0 0 0 2 0 0 0 0 2 -2*(3+hi*delx/k) 1
       0 0 0 0 1;
     0 0 0 0 0 0 0 0 0 2 0 0 0 0 1 -2*(2+hi*delx/k)
       1 0 0 0 0;
     0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 1 -2*(2+hi*delx/k
       ) 1 0 0 0;
     0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 -(2+hi*delx/k
       ) 0 0 0;
     0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 -2 1 0;
     0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 1 -4 1;
     0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 -(2+hi*
       delx/k)];

C = [-ho*delx*Tg/k;
     -2*ho*delx*Tg/k;
     -2*ho*delx*Tg/k;
     -2*ho*delx*Tg/k;
     -2*ho*delx*Tg/k;
     -ho*delx*Tg/k;
     0;
     0;
```

```
      0;
      0;
      0;
      0;
      0;
      0;
      -2*hi*delx*Ti/k;
      -2*hi*delx*Ti/k;
      -2*hi*delx*Ti/k;
      -hi*delx*Ti/k;
      0;
      0;
      -hi*delx*Ti/k];

T = inv(A)*C;

printf("\n Temp Distribution = ");
printf("\n    %.1f K ", T);

q = 4*ho*[(delx/2)*(Tg-T(1))+delx*(Tg-T(2))+delx*(Tg-T(3))+ delx*(Tg-T(4))+delx*(Tg-T(5))+delx*(Tg-T(6))/2];
printf("\n\n Heat rate Transfer %.1f W/m ", q);
//END
```

Chapter 5

Transient Conduction

Scilab code Exa 5.1 Thermo Couple Junction

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
    Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
    5.1    Page 261 \n'); //Example 5.1
// Junction Diameter and Time Calculation to attain
    certain temp

//Operating Conditions

h = 400;           //[W/m^2.K] Heat Convection
    coefficient
k = 20;          //[W/m.K] Thermal Conductivity of
    Blade
c = 400;         //[J/kg.K] Specific Heat
rho = 8500;       //[kg/m^3] Density
Ti = 25+273;         //[K] Temp of Air
Tsurr = 200+273;      //[K] Temp of Gas Stream
TimeConstt = 1;       //[sec]

//From Eqn 5.7
```

```
D = 6*h*TimeConstt/(rho*c);
Lc = D/6;
Bi = h*Lc/k;

//From eqn 5.5 for time to reach
T = 199+273;     //[K] Required temperature

t = rho*D*c*2.30*log10((Ti-Tsurr)/(T-Tsurr))/(h*6);

printf("\n\n Junction Diameter needed for a time
    constant of 1 s = %.2e m \n\n Time Required to
    reach 199degC in a gas stream = %.1f sec ", D, t)
    ;
//END
```

Scilab code Exa 5.2 Steady State Temperature of Junction

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
    Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
    5.2   Page 265 \n'); //Example 5.2
// Steady State Temperature of junction
// Time Required for thermocouple to reach a temp
    that is within 1 degc of its steady-state value

//Operating Conditions

h = 400;            //[W/m^2.K] Heat Convection
    coefficient
k = 20;           //[W/m.K] Thermal Conductivity of
    Blade
c = 400;          //[J/kg.K] Specific Heat
e = .9;          //Absorptivity
rho = 8500;        //[kg/m^3] Density
```

```
Ti = 25+273;         //[K] Temp of Air
Tsurr = 400+273;      //[K] Temp of duct wall
Tg = 200+273;      //[K] Temp of Gas Stream
TimeConstt = 1;       //[sec]
stfncnstt=5.67*10^(-8);     // [W/m^2.K^4] - Stefan Boltzmann Constant

//From Eqn 5.7
D = 6*h*TimeConstt/(rho*c);
As = %pi*D^2;
V = %pi*D^3/6;

//Balancing Energy on thermocouple Junction
//Newton Raphson method for 4th order eqn
T=500;
while(1>0)
f=(e*stfncnstt*(Tsurr^4-T^4)-(h*(T-Tg)));
fd=(-3*e*stfncnstt*T^3)-h;
Tn=T-f/fd;
if((e*stfncnstt*(Tsurr^4-Tn^4)-(h*(Tn-Tg)))<=.01)
    break;
end;
T=Tn;
end
printf("\n (a) Steady State Temperature of junction = %.2f degC\n",T-273);

//Using Eqn 5.15 and Integrating the ODE
// Integration of the differential equation
// dT/dt=-A*[h*(T-Tg)+e*stefncnstt*(T^4-Tsurr^4)]/(rho*V*c) , T(0)=25+273, and finds the minimum time t such that T(t)=217.7+273.15
deff("[Tdot]=f(t,T)","Tdot=-As*[h*(T-Tg)+e*stfncnstt*(T^4-Tsurr^4)]/(rho*V*c)");
deff("[z]=g(t,T)","z=T-217.7-273");

T0=25+273;ng=1;
[T,rd]=ode("roots",T0,0,217.7+273,f,ng,g);
```

```
printf("\n (b) Time Required for thermocouple to
   reach a temp that is within 1 degc of its steady-
   state value = %.2f s\n",rd(1));

//END
```

Scilab code Exa 5.3 Total Time Required for Two Step Process

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   5.2    Page 267 \n'); //Example 5.3
// Total Time t required for two step process

//Operating Conditions

ho = 40;           //[W/m^2.K] Heat Convection
   coefficient
hc = 10;           //[W/m^2.K] Heat Convection
   coefficient
k = 177;          //[W/m.K] Thermal Conductivity
e = .8;         //Absorptivity
L = 3*10^-3/2;       //[m] Metre
Ti = 25+273;         //[K] Temp of Aluminium
Tsurro = 175+273;      //[K] Temp of duct wall
   heating
Tsurrc = 25+273;      //[K] Temp of duct wall
Tit = 37+273;      //[K] Temp at cooling
Tc = 150+273;         //[K] Temp critical

stfncnstt=5.67*10^(-8);    // [W/m^2.K^4] - Stefan
   Boltzmann Constant
p = 2770;         //[kg/m^3] density of aluminium
c = 875;         //[J/kg.K] Specific Heat
```

```

//To assess the validity of the lumped capacitance approximation
Bih = ho*L/k;
Bic = hc*L/k;
printf("\n Lumped capacitance approximation is valid as Bih = %f and Bic = %f", Bih, Bic);

//Eqn 1.9
hro = e*stfncnstt*(Tc+Tsurro)*(Tc^2+Tsurro^2);
hrc = e*stfncnstt*(Tc+Tsurrc)*(Tc^2+Tsurrc^2);
printf("\n Since The values of hro = %.1f and hrc = %.1f are comparable to those of ho and hc, respectively radiation effects must be considered", hro,hrc);

// Integration of the differential equation
// dy/dt=-1/(p*c*L)*[ho*(y-Tsurro)+e*stfncnstt*(y^4 - Tsurro^4)] , y(0)=Ti, and finds the minimum time t such that y(t)=150 degC
deff("[ydot]=f1(t,y)","ydot=-1/(p*c*L)*[ho*(y-Tsurro)+e*stfncnstt*(y^4 - Tsurro^4)]");
deff("[z]=g1(t,y)","z=y-150-273");
y0=Ti;
[y,tc]=ode("root",y0,0,150+273,f1,1,g1);
te = tc(1) + 300;

//From equation 5.15 and solving the two step process using integration
function Tydot=f(t,T)
    Tydot=-1/(p*c*L)*[ho*(T-Tsurro)+e*stfncnstt*(T^4 - Tsurro^4)];
    funcprot(0)
endfunction
Ty0=Ti;
t0=0;
t=0:10:te;
Ty=ode("rk",Ty0,t0,t,f);
```

```

// solution of integration of the differential equation
// dy/dt=-1/(p*c*L)*[hc*(y-Tsurrc)+e*stfncnstt*(y^4 - Tsurrc^4)] , y(rd(1))=Ty(43), and finds the minimum time t such that y(t)=37 degC=Tit
deff("[Tdot]=f2(t,T)","Tdot=-1/(p*c*L)*[hc*(T-Tsurrc)+e*stfncnstt*(T^4 - Tsurrc^4)]");
for(tt=0:1:900)
    tq=ode(Ty(43),0,tt,f2);
    if(tq-Tit<=10^-2)
        break;
    end
end

function Ty2dot=f2(t,T)
    Ty2dot=-1/(p*c*L)*[hc*(T-Tsurrc)+e*stfncnstt*(T^4 - Tsurrc^4)];
    funcprot(0)
endfunction
Ty20=Ty(43);
t20=te;
t2=te:10:1200;
Ty2=ode("rk",Ty20,t20,t2,f2);
clf();
plot(t,Ty-273,t2,Ty2-273,[tc(1) tc(1)],[0 Tc-273],[te te],[0 Ty(43)-273],[tt+te tt+te],[0 tq-273]);
xtitle('Plot of the Two-Step Process','t (s)','T (degC)');
legend('Heating','Cooling','tc','te','tt');

printf('\n\n Total time for the two-step process is t = %i s with intermediate times of tc = %i s and te = %i s.',tt+te,tc(1),te);
//END
```

Scilab code Exa 5.4 Radial System with Convection

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 5.4    Page 278 \n'); //Example 5.4
// Radial System with Convection

//Operating Conditions

h = 500;            //[W/m^2.K] Heat Convection coefficientat inner surface
k = 63.9;           //[W/m.K] Thermal Conductivity
rho = 7832;          //[kg/m^3] Density
c = 434;              //[J/kg.K]  Specific Heat
alpha = 18.8*10^-6;              //[m^2/s]
L = 40*10^-3;        //[m] Metre
Ti = -20+273;          //[K] Initial Temp
Tsurr = 60+273;      //[K] Temp of oil
t = 8*60 ;            //[sec] time
D = 1 ;         //[m] Diameter of pipe

//Using eqn 5.10 and 5.12
Bi = h*L/k;
Fo = alpha*t/L^2;

//From Table 5.1 at this Bi
C1 = 1.047;
eta = 0.531;
theta0=C1*exp(-eta^2*Fo);
T = Tsurr+theta0*(Ti-Tsurr);

//Using eqn 5.40b
```

```
x=1;
theta = theta0*cos(eta);
T1 = Tsurr + (Ti-Tsurr)*theta;
q = h*[T1 - Tsurr];

//Using Eqn 5.44, 5.46 and Vol per unit length V = pi*D*L
Q = [1-(sin(eta)/eta)*theta0]*rho*c*%pi*D*L*(Ti-Tsurr);

printf("\n (a) After 8 min Biot number = %.2f and Fourier Numer = %.2f \n\n (b) Temperature of exterior pipe surface after 8 min = %i degC \n\n (c) Heat Flux to the wall at 8 min = %i W/m^2 \n\n (d) Energy transferred to pipe per unit length after 8 min = %.2e J/m",Bi,Fo, T-273,q,Q);

//END
```

Scilab code Exa 5.5 Two Step Cooling Process Of Sphere

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 5.5    Page 280 \n'); //Example 5.5
// Two step cooling process of Sphere

//Operating Conditions

ha = 10;            //[W/m^2.K] Heat Convection coefficientat air
hw = 6000;            //[W/m^2.K] Heat Convection coefficientat water
k = 20;           //[W/m.K] Thermal Conductivity
```

```
rho = 3000;         //[kg/m^3] Density
c = 1000;              //[J/kg.K]  Specific Heat
alpha = 6.66*10^-6;             //[m^2/s]
Tiw = 335+273;         //[K] Initial Temp
Tia = 400+273;         //[K] Initial Temp
Tsurr = 20+273;      //[K] Temp of surrounding
T = 50+273;         //[K] Temp of center
ro = .005;         //[m] radius of sphere

//Using eqn 5.10 and
Lc = ro/3;
Bi = ha*Lc/k;
ta = rho*ro*c*2.30*(log10((Tia-Tsurr)/(Tiw-Tsurr)))/(3*ha);

//From Table 5.1 at this Bi
C1 = 1.367;
eta = 1.8;
Fo = -1*2.30*log10((T-Tsurr)/((Tiw-Tsurr)*C1))/eta^2;

tw = Fo*ro^2/alpha;

printf("\n (a) Time required to accomplish desired cooling in air ta = %.1f s\n\n (b) Time required to accomplish desired cooling in water bath tw = %.2f s",ta,tw);

//END
```

Scilab code Exa 5.6 Burial Depth

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
```

```
    Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
    5.6   Page 288 \n'); //Example 5.6
// Burial Depth

//Operating Conditions

k = .52;          //[W/m.K] Thermal Conductivity
rho = 2050;         //[kg/m^3] Density
c = 1840;             //[J/kg.K]  Specific Heat
Ti = 20+273;        //[K] Initial Temp
Ts = -15+273;     //[K] Temp of surrounding
T = 0+273;        //[K] Temp at depth xm after 60
    days
t = 60*24*3600;          //[sec] time perod

alpha = k/(rho*c);              //[m^2/s]
//Using eqn 5.57
xm = erfinv((T-Ts)/(Ti-Ts))*2*(alpha*t)^.5;

printf("\n Depth at which after 60 days soil freeze
    = %.2f m",xm);

//END
```

Scilab code Exa 5.7 Spherical Tumor

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
    Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
    5.7   Page 293 \n'); //Example 5.7
// Spherical Tumor

//Operating Conditions

```

```
k = .5;           //[W/m.K] Thermal Conductivity Healthy Tissue
kappa = .02*10^3;     //[m] extinction coefficient
p = .05;             // reflectivity of skin
D = .005;             //[m] Laser beam Dia
rho = 989.1  ;       //[kg/m^3] Density
c = 4180 ;            //[J/kg.K]  Specific Heat
Tb = 37+273;         //[K] Temp of healthy tissue
Dt = .003 ;          //[m] Dia of tissue
d = .02  ;           //[m] depth beneath the skin
Ttss = 55+273 ;        //[K] Steady State Temperature
Tb = 37+273 ;        //[K] Body Temperature
Tt = 52+273 ;        //[K] Tissue Temperature
q = .170 ;            //[W]

//Case 12 of Table 4.1
q = 2*%pi*k*Dt*(Ttss-Tb);

//Energy Balancing
P = q*(D^2)*exp(kappa*d)/((1-p)*Dt^2);

//Using Eqn 5.14
t = rho*(%pi*Dt^3/6)*c*(Tt-Tb)/q;

alpha=k/(rho*c);
Fo = 10.3;
//Using Eqn 5.68
t2 = Fo*Dt^2/(4*alpha);

printf("\n (a) Heat transferred from the tumor to maintain its surface temperature at Ttss = 55 degC is %.2f W \n\n (b) Laser power needed to sustain the tumor surface temperautre at Ttss = 55 degC is %.2f W \n\n (c) Time for tumor to reach Tt = 52 degC when heat transfer to the surrounding tissue is neglected is %.2f sec \n\n (d) Time for tumor to reach Tt = 52 degC when Heat transfer to thesurrounding tissue is
```

```
   considered and teh thermal mass of tumor is
   neglected is %.2f sec” ,q,P,t,t2);

//END
```

Scilab code Exa 5.8 Thermal Conductivity of Nanostructured Material

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   5.8    Page 300 \n'); //Example 5.8
// Thermal Conductivity of Nanostructured material

//Operating Conditions

k = 1.11 ;          //[W/m.K] Thermal Conductivity
rho = 3100;          //[kg/m^3] Density
c = 820 ;            //[J/kg.K] Specific Heat
//Dimensions of Strip
w = 100*10^-6;       //[m] Width
L = .0035 ;          //[m] Long
d = 3000*10^-10;     //[m] Thickness
delq = 3.5*10^-3;    //[W] heating Rate
delT1 =1.37 ;       //[K] Temperature 1
f1 = 2*%pi ;        //[rad/s] Frequency 1
delT2 =.71 ;       //[K] Temperature 2
f2 = 200*%pi;          //[rad/s] Frequency 2

A = [delT1 -delq/(L*%pi);
     delT2 -delq/(L*%pi)] ;

C= [delq*-2.30*log10(f1/2)/(2*L*%pi);
    delq*-2.30*log10(f2/2)/(2*L*%pi)] ;

```

```
B = inv(A)*C;

alpha = k/(rho*c);
delp = [(alpha/f1)^.5 (alpha/f2)^.5];
printf("\n C2 = %.2f    k = %.2f W/m.K \n\n Thermal Penetration depths are %.2e m and %.2e m at frequency 2*pi rad/s and 200*pi rad/s",B(2),B(1), delp);

//END
```

Scilab code Exa 5.9 Temperature Distribution Using Finite Difference Method

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 5.9    Page 305 \n'); //Example 5.9
// Temperature distribution 1.5s after a change in operating power

//Operating Conditions

L = .01;                  //[m] Metre
Tsurr = 250+273;           //[K] Temperature
h = 1100;                  //[W/m^2.K] Heat Convective Coefficient
q1 = 10^7;                 //[W/m^3] Volumetric Rate
q2 = 2*10^7;                 //[W/m^3] Volumetric Rate
k = 30;                    //[W/m.K] Conductivity
a = 5*10^-6;                //[m^2/s]

delx = L/5;         //Space increment for numerical solution
Bi = h*delx/k;            //Biot  Number
```

```
//By using stability criterion for Fourier Number
Fo = (2*(1+Bi))^-1;
//By definition
t = Fo*delx^2/a;
printf('\n As per stability criterion delt = %.3f s, hence setting stability limit as .3 s.',t)
// Using Finite time increment of .3s
delt = 1*.3;
Fo1 = a*delt/delx^2;
x = [0 delx delx*2 delx*3 delx*4 delx*5];

//At p=0 Using equation 3.46
for i = 1: length(x)
T(1,i) = q1*L^2/(2*k)*(1-x(i)^2/L^2)+Tsurr + q1*L/h -273 ;
end
//System of Equation in Finite Difference method
for j = 2:6
    T(j,1)=Fo1*(2*T(j-1,2)+q2*delx^2/k) + (1 -2*Fo1)*T(j-1,1);
    T(j,2)=Fo1*(T(j-1,1)+T(j-1,3)+q2*delx^2/k) + (1 -2*Fo1)*T(j-1,2);
    T(j,3)=Fo1*(T(j-1,2)+T(j-1,4)+q2*delx^2/k) + (1 -2*Fo1)*T(j-1,3);
    T(j,4)=Fo1*(T(j-1,3)+T(j-1,5)+q2*delx^2/k) + (1 -2*Fo1)*T(j-1,4);
    T(j,5)=Fo1*(T(j-1,4)+T(j-1,6)+q2*delx^2/k) + (1 -2*Fo1)*T(j-1,5);
    T(j,6)=2*Fo1*(T(j-1,5)+Bi*(Tsurr-273)+q2*delx^2/(2*k)) + (1 -2*Fo1-2*Bi*Fo1)*T(j-1,6);
end
//At p=infinity Using equation 3.46
x = [0 delx delx*2 delx*3 delx*4 delx*5];
for i = 1:length(x)
T(7,i) = q2*L^2/(2*k)*(1-x(i)^2/L^2)+Tsurr+q2*L/h -273;
end

```

```
for j= 1:6
Tans(j,:) = [j-1 delt*(j-1) T(j,:)];
end

printf(”\n\n Tabulated Nodal Temperatures \n\n    p      t(s)     T0        T1          T2         T3          T4        T5\n”);
format(’v’,6);
disp(Tans);
printf(”    inf     inf     %.1f     %.1f     %.1f     %.1f     %.1f     %.1f”,T(7,1),T(7,2),T(7,3),T(7,4),T(7,5),T(7,6));

//END
```

Scilab code Exa 5.10 Temperature Distribution Analytical and Explicit and Implicit Finite Difference

```
clear;
clc;
printf(’FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 5.10    Page 311 \n’); //Example 5.10
// Using Explicit Finite Difference method, determine temperatures at the surface and 150 mm from the surface after an elapsed time of 2 min
// Repeat the calculations using the Implicit Finite Difference Method
// Determine the same temperatures analytically

//Operating Conditions

delx = .075;                //[m] Metre
T = 20+273;         //[K] Temperature
q = 3*10^5;               //[W/m^3] Volumetric Rate
```

```
//From Table A.1 copper 300 K
k = 401;              //[W/m.K] Conductivity
a = 117*10^-6;        //[m^2/s]

//By using stability criterion reducing further Fourier Number
Fo = (2)^-1;
//By definition
delt = Fo*delx^2/a;
format('v',5);

//System of Equation for Explicit Finite difference Fo = 1/2
Tv1(1,:) = [20 20 20 20 20];                    //At p=0 Initial Temperature t - 20 degC
for i = 2:6
    Tv1(i,1) = 56.1 + Tv1(i-1,2);
    Tv1(i,2) = (Tv1(i-1,3) + Tv1(i-1,1))/2;
    Tv1(i,3) = (Tv1(i-1,4) + Tv1(i-1,2))/2;
    Tv1(i,4) = (Tv1(i-1,5) + Tv1(i-1,3))/2;
    Tv1(i,5) = Tv1(i-1,5);
end
for j=1:6
    T1(j,:)=[j-1 delt*(j-1) Tv1(j,:)];
end
printf("\n\n EXPLICIT FINITE-DIFFERENCE SOLUTION FOR Fo = 1/2\n    p      t(s)     T0       T1       T2       T3       T4\n");
disp(T1);
printf('\n Hence after 2 min, the surface and the desirde interior temperature T0 = %.2f degC and T2 = %.1f degC',T1(6,3),T1(6,5));

//By using stability criterion reducing further Fourier Number
Fo = (4)^-1;
//By definition
```

```
delt = Fo*delx^2/a;
//System of Equation for Explicit Finite difference for Fo = 1/4
Tv2(1,:) = [20    20    20    20    20    20    20    20    20];           //At p=0 Initial Temperature t - 20 degC
for i=2:11
    Tv2(i,1)=1/2*(q*delx/k  + Tv2(i-1,2)) +Tv2(i-1,1)/2;
    Tv2(i,2)=(Tv2(i-1,1)+Tv2(i-1,3))/4 + Tv2(i-1,2)/2;
    Tv2(i,3)=(Tv2(i-1,2)+Tv2(i-1,4))/4 + Tv2(i-1,3)/2;
    Tv2(i,4)=(Tv2(i-1,3)+Tv2(i-1,5))/4 + Tv2(i-1,4)/2;
    Tv2(i,5)=(Tv2(i-1,4)+Tv2(i-1,6))/4 + Tv2(i-1,5)/2;
    Tv2(i,6)=(Tv2(i-1,5)+Tv2(i-1,7))/4 + Tv2(i-1,6)/2;
    Tv2(i,7)=(Tv2(i-1,6)+Tv2(i-1,8))/4 + Tv2(i-1,7)/2;
    Tv2(i,8)=(Tv2(i-1,7)+Tv2(i-1,9))/4 + Tv2(i-1,8)/2;
    Tv2(i,9)= Tv2(i-1,9);
end
for j=1:11
    T2(j,:)=[j-1 delt*(j-1) Tv2(j,:)];
end
printf("\n\n  EXPLICIT FINITE-DIFFERENCE SOLUTION FOR Fo = 1/4\n     p       t(s)       T0        T1        T2        T3       T4          T5        T6        T7       T8        \n")
disp(T2)
printf('\n Hence after 2 min, the surface and the desirde interior temperature T0 = %.2f degC and T2 = %.1f degC',T2(11,3),T2(11,5))


```

```
//(b)Implicit Finite Difference solution
Fo = (4)^-1;
//By definition
delt = Fo*delx^2/a;

T3 = rand(6,11);              //Random Initital
   Distribution
function[Tm]=Tvalue(i)
function[f]=F(x)
    f(1)= 2*x(1) - x(2) - q*delx/k - T3(i,3);
    f(2)= -x(1)+4*x(2)-x(3)-2*T3(i,4);
    f(3)= -x(2)+4*x(3)-x(4)-2*T3(i,5);
    f(4)= -x(3)+4*x(4)-x(5)-2*T3(i,6);
    f(5)= -x(4)+4*x(5)-x(6)-2*T3(i,7);
    f(6)= -x(5)+4*x(6)-x(7)-2*T3(i,8);
    f(7)= -x(6)+4*x(7)-x(8)-2*T3(i,9);
    f(8)= -x(7)+4*x(8)-x(9)-2*T3(i,10);
    f(9)= -x(9)+T3(i,11);
    funcprot(0);
endfunction
x = [30 30 30 30 30 30 30 30 30];
Tm = fsolve(x,F);
    funcprot(0)
endfunction

//At p=0 Initial Temperature t – 20 degC
T3(1,:) = [0 delt*0 20    20    20    20    20    20
       20    20    20];
for j=1:5
    T3(j+1,:)=[j delt*j Tvalue(j)];
end
printf("\n\n  IMPLICIT FINITE–DIFFERENCE SOLUTION
   FOR Fo = 1/4\n    p      t(s)      T0       T1
   T2       T3      T4        T5       T6       T7      T8
   \n");
disp(T3);
printf('\n Hence after 2 min, the surface and the
   desirde interior temperature T0 = %.2f degC and
```

```
   T2 = %.1f degC',T3(6,3),T3(6,5));

t = 120;        //[seconds]
//(c) Approximating slab as semi-infinte medium
Tc = T -273 + 2*q*(a*t/%pi)^.5/k;

//At interior point x=0.15 m
x =.15;         //[metre]
//Analytical Expression
Tc2 = T -273 + 2*q*(a*t/%pi)^.5/k*exp(-x^2/(4*a*t))-
   q*x/k*[1-erf(.15/(2*sqrt(a*t)))];

printf('\n\n (c) Approximating slab as a semi
   infinte medium, Analytical epression yields \n At
    surface after 120 seconds = %.1f degC \n At x
   =.15 m after 120 seconds = %.1f degC',Tc,Tc2);
//END
```

Chapter 6

Introduction to Convection

Scilab code Exa 6.1 Theroetical Problem

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   6.1    Page 355 \n')// Example 6.1
//Theoretical Problem

printf('\n The given example is theoretical and does
    not involve any numerical computation')

//End
```

Scilab code Exa 6.2 Napthalene Sublimation

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   6.2    Page 356 \n'); //Example 6.2
```

```
// Napthalene Sublimation rate per unit length

//Operating Conditions

h = .05;            //[W/m^2.K] Heat Convection coefficient
D = .02;            //[m] Diameter of cylinder
Cas = 5*10^-6;      //[kmol/m^3] Surface molar Conc
Casurr = 0;         //[kmol/m^3] Surrounding molar Conc
Ma = 128;           //[Kg/kmol] Molecular weight

//From Eqn 6.15
Na = h*(%pi*D)*(Cas-Casurr);
na = Ma*Na;

printf("\n\n Mass sublimation Rate is = %.2e kg/s.m", na);
//END
```

Scilab code Exa 6.3 Convection Mass Transfer Coefficient

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 6.3    Page 357 \n'); //Example 6.3
// Convection Mass Transfer coefficient

//Operating Conditions

Dab = .288*10^-4;          //[m^2/s] Table A.8 water vapor-air (319K)
pas = .1;              //[atm] Partial pressure at surface
```

```
pasurr = .02;              //[atm] Partial pressure at infinity
y0 = .003;              //[m] Tangent at y = 0 intercepts y axis at 3 mm

//From Measured Vapor Pressure Distribution
delp = (0 - pas)/(y0 - 0);              //[atm/m]
hmx = -Dab*delp/(pas - pasurr);         //[m/s]

printf("\n\n Convection Mass Transfer coefficient at prescribed location = %.4f m/s", hmx);
//END
```

Scilab code Exa 6.4 Convection Mass Transfer coefficient of Plate

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 6.4    Page 362 \n'); //Example 6.4
// Convection Mass Transfer coefficient

//Operating Conditions
v = 1;          //[m/s]  Velocity of water
L = 0.6;        //[m] Plate length
Tw1 = 300;      //[K]
Tw2 = 350;      //[K]
//Coefficients [W/m^1.5 . K]
Clam1 = 395;
Cturb1 = 2330;
Clam2 = 477;
Cturb2 = 3600;

//Water Properties at T = 300K
p1 = 997;     //[kg/m^3]  Density
```

```
u1 = 855*10^-6;     //[N.s/m^2] Viscosity
//Water Properties at T = 350K
p2 = 974;     //[kg/m^3]  Density
u2 = 365*10^-6;     //[N.s/m^2] Viscosity


Rec = 5*10^5;         //Transititon Reynolds Number
xc1 = Rec*u1/(p1*v); //[m]Transition length at 300K
xc2 = Rec*u2/(p2*v); //[m]Transition length at 350K

//Integrating eqn 6.14
//At 300 K
h1 = [Clam1*xc1^.5/.5 + Cturb1*(L^.8-xc1^.8)/.8]/L;

//At 350 K
h2 = [Clam2*xc2^.5/.5 + Cturb2*(L^.8-xc2^.8)/.8]/L;

printf("\n\n Average Convection Coefficient over the entire plate for the two temperatures at 300K = %.2f W/m^2.K and at 350K = %.2f W/m^2.K", h1,h2);
//END
```

Scilab code Exa 6.5 Heat Flux of Plate

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 6.5    Page 372 \n'); //Example 6.5
// Heat Flux to blade when surface temp is reduced
// Heat flux to a larger turbine blade

//Operating Conditions
v = 160;          //[m/s]  Velocity of air
L = 0.04;        //[m] Blade length
```

```
Tsurr = 1150+273;     //[K]
Ts = 800+273;     //[K] Surface Temp
q = 95000;         //[W/m^2] Original heat flux

//Case 1
Ts1 = 700+273;     //[K] Surface Temp
q1 = q*(Tsurr-Ts1)/(Tsurr-Ts);

//Case 2
L2 = .08;              //[m] Length
q2 = q*L/L2;          //[W/m^2] Heat flux


printf("\n\n (a) Heat Flux to blade when surface
   temp is reduced = %i KW/m^2 \n (b) Heat flux to a
    larger turbine blade = %.2f KW/m^2", q1/1000,q2
   /1000);
//END
```

Scilab code Exa 6.6 Molar Flux over Plate

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   6.6    Page 379 \n'); //Example 6.6
// Water vapor conc and flux associated with the
   same location on larger surface of the same shape

//Operating Conditions
v = 100;          //[m/s]  Velocity of air
Tsurr = 20+273;     //[K] Surrounding Air Temperature
L1 = 1;       //[m] solid length
Ts = 80+273;     //[K] Surface Temp
qx = 10000;         //[W/m^2] heat flux at a point x
```

```
Txy = 60+273;         //[K] Temp in boundary layer
   above the point

//Table A.4 Air Properties at T = 323K
v = 18.2*10^-6;     //[m^2/s] Viscosity
k = 28*10^-3;       //[W/m.K] Conductivity
Pr = 0.7;           //Prandttl Number
//Table A.6 Saturated Water Vapor at T = 323K
pasat = 0.082;      //[kg/m^3]
Ma = 18;            //[kg/kmol] Molecular mass of
   water vapor
//Table A.8 Water Vapor-air at T = 323K
Dab = .26*10^-4;     //[m^2/s]

//Case 1
Casurr = 0;
Cas = pasat/Ma;         //[kmol/m^3] Molar conc of
   saturated water vapor at surface
Caxy = Cas + (Casurr - Cas)*(Txy - Ts)/(Tsurr - Ts);

//Case 2
L2 = 2;
hm = L1/L2*Dab/k*qx/(Ts-Tsurr);
Na = hm * (Cas - Casurr);


printf("\n (a) Water vapor Concentration above the
   point = %.4f Kmol/m^3 \n (b) Molar flux to a
   larger surface = %.2e Kmol/s.m^2", Caxy,Na);
//END
```

Scilab code Exa 6.7 Evaporative Cooling

```
clear;
clc;
```

```
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 6.7    Page 383 \n'); //Example 6.7
// Steady State Temperature of Beverage

//Operating Conditions
Tsurr = 40+273;     //[K] Surrounding Air Temperature
//Volatile Wetting Agent A
hfg = 100;          //[kJ/kg]
Ma = 200;           //[kg/kmol] Molecular mass
pasat = 5000;       //[N/m^2] Saturate pressure
Dab = .2*10^-4;     //[m^2/s] Diffusion coefficient

//Table A.4 Air Properties at T = 300K
p = 1.16;                   //[kg/m^3] Density
cp = 1.007;                 //[kJ/kg.K] Specific Heat
alpha = 22.5*10^-6;         //[m^2/s]
R = 8.314;                  //[kJ/kmol] Universal Gas Constt

//Applying Eqn 6.65 and setting pasurr = 0
// Ts^2 - Tsurr*Ts + B = 0      , where the coefficient B is
B = Ma*hfg*pasat*10^-3/[R*p*cp*(alpha/Dab)^(2/3)];
Ts = [Tsurr + sqrt(Tsurr^2 - 4*B)]/2;

printf("\n Steady State Surface Temperature of Beverage = %.1f degC", Ts-273);
//END
```

Chapter 7

External Flow

Scilab code Exa 7.1 Cooling Rate per Unit Width of the Plate

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   7.1    Page 415 \n'); //Example 7.1
// Cooling rate per Unit Width of the Plate

//Operating Conditions
v = 10;                 //[m/s] Air velocity
p = 6000;               //[N/m^2] Air pressure
Tsurr = 300+273;         //[K] Surrounding Air
   Temperature
L = .5;                 //[m] Length of plate
Ts = 27+273;            //[K] Surface Temp

//Table A.4 Air Properties at T = 437K
uv = 30.84*10^-6*(101325/6000);          //[m^2/s]
   Kinematic Viscosity at P = 6000 N/m^2
k = 36.4*10^-3;                //[W/m.K] Thermal
   COnductivity
Pr = .687;                     //Prandtl number
```

```

Re = v*L/uv;          //Reynolds number
printf("\n Since Reynolds Number is %i, The flow is laminar over the entire plate",Re);

//Correlation 7.30
NuL = .664*Re^.5*Pr^.3334;     //Nusselt Number over entire plate length
hL = NuL*k/L;                  // Average Convection Coefficient
//Required cooling rate per unit width of plate
q = hL*L*(Tsurr-Ts);

printf("\n\n Required cooling rate per unit width of plate = %i W/m", q);
//END
```

Scilab code Exa 7.2 Maximum Heater Power Requirement

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 7.2    Page 417 \n'); //Example 7.2
// Maximum Heater Power Requirement

//Operating Conditions
v = 60;               //[m/s] Air velocity
Tsurr = 25+273;       //[K] Surrounding Air Temperature
w = 1;              //[m] Width of plate
L = .05;            //[m] Length of stripper
Ts = 230+273;         //[K] Surface Temp

//Table A.4 Air Properties at T = 400K
uv = 26.41*10^-6;           //[m^2/s] Kinematic
```

```
Viscosity
k = .0338;                  //[W/m.K]  Thermal COnductivity
Pr = .690;                  //Prandtl number

Re = v*L/uv;         //Reynolds number

Rexc = 5*10^5;         //Transition Reynolds Number
xc = uv*Rexc/v;       //Transition Length
printf("\n Reynolds Number based on length L = .05m is %i. \n And the transition occur at xc = %.2f m ie fifth plate",Re,xc);

//For first heater
//Correlation 7.30
Nu1 = .664*Re^.5*Pr^.3334;     //Nusselt Number
h1 = Nu1*k/L;                    // Average Convection Coefficient
q1 = h1*(L*w)*(Ts-Tsurr);    // Convective Heat exchange

//For first four heaters
Re4 = 4*Re;
L4 = 4*L;
Nu4 = .664*Re4^.5*Pr^.3334;     //Nusselt Number
h4 = Nu4*k/L4;                    // Average Convection Coefficient

//For Fifth heater from Eqn 7.38
Re5 = 5*Re;
A = 871;
L5 = 5*L;
Nu5 = (.037*Re5^.8-A)*Pr^.3334;     //Nusselt Number
h5 = Nu5*k/L5;                    // Average Convection Coefficient
q5 = (h5*L5-h4*L4)*w*(Ts-Tsurr);

//For Sixth heater from Eqn 7.38
```

```
Re6 = 6*Re;
L6 = 6*L;
Nu6 = (.037*Re6^.8-A)*Pr^.3334 ;    //Nusselt Number
h6 = Nu6*k/L6 ;                    // Average Convection Coefficient
q6 = (h6*L6-h5*L5)*w*(Ts-Tsurr);

printf(”\n\n Power requirement are \n qconv1 = %i W  qconv5 = %i W  qconv6 = %i W”, q1,q5,q6);
printf(”\n Hence %i > %i > %i and the sixth plate has largest power requirement”, q6,q1,q5);
//END
```

Scilab code Exa 7.3 Daily Water Loss

```
clear;
clc;
printf(’FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 7.3    Page 417 \n’); //Example 7.2
// Daily Water Loss

//Operating Conditions
v = 2;                  //[m/s] Air velocity
Tsurr = 25+273;          //[K] Surrounding Air Temperature
H = .5;                  // Humidity
w = 6;                  //[m] Width of pool
L1 = 12;                //[m] Length of pool
e = 1.5;               //[m] Deck Wide
Ts = 25+273;             //[K] Surface Temp of water

//Table A.4 Air Properties at T = 298K
uv = 15.7*10^-6;            //[m^2/s] Kinematic Viscosity
//Table A.8 Water vapor-Air Properties at T = 298K
```

```
Dab = .26*10^-4;          //[m^2/s] Diffusion Coefficient
Sc = uv/Dab;
//Table A.6 Air Properties at T = 298K
rho = .0226;                    //[kg/m^3]

L = L1+e;
Re = v*L/uv;         //Reynolds number

//Equation 7.41 yields
ShLe = .037*Re^.8*Sc^.3334;
//Equation 7.44
p = 8;         //Turbulent Flow
ShL = (L/(L-e))*ShLe*[1-(e/L)^((p+1)/(p+2))]^(p/(p+1));

hmL = ShL*(Dab/L);
n = hmL*(L1*w)*rho*(1-H);

printf(”\n Reynolds Number is %.2e. Hence for turbulent Flow p = 8 in Equation 7.44.\n Daily Water Loss due to evaporation is %i kg/day”,Re,n*86400);

//END
```

Scilab code Exa 7.4 Convection Coefficient Using Zukauskas Relation

```
clear;
clc;
printf(’FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 7.4    Page 428 \n’); //Example 7.4
// Convection Coefficient associated with operating conditions
```

```
// Convection Coefficient from an appropriate correlation

//Operating Conditions
v = 10;                 //[m/s] Air velocity
Tsurr = 26.2+273;         //[K] Surrounding Air Temperature
P = 46;                 // [W] Power dissipation
L = .094;               //[m] Length of cylinder
D = .0127;              //[m] Diameter of cylinder
Ts = 128.4+273;            //[K] Surface Temp of water
q = 46-.15*46;             //[W] Actual power dissipation without the 15% loss

//Table A.4 Air Properties at T = 300K
uv = 15.89*10^-6;            //[m^2/s] Kinematic Viscosity
k = 26.3*10^-3;              //[W/m.K] Thermal conductivity
Pr = .707;                   //Prandtl Number
//Table A.4 Air Properties at T = 401K
Prs = .690;                   //Prandtl Number

A = %pi*D*L;
h = q/(A*(Ts-Tsurr));

Re = v*D/uv;          //Reynolds number
//Using Zukauskas Relation, Equation 7.53
C = .26;
m = .6;
n = .37;
Nu = C*Re^m*Pr^n*(Pr/Prs)^.25;
havg = Nu*k/D;

printf("\n Convection Coefficient associated with operating conditions %i W/m^2.K. \n Reynolds Number is %i. Hence taking suitable corresponding data from Table 7.4.\n Convection Coefficient
```

```
from an appropriate Zukauskas correlation %i W/m^2.K",h,Re,havg);

//END
```

Scilab code Exa 7.5 Convective Heat transfer to the Canister

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 7.5    Page 431 \n'); //Example 7.5
// Convective Heat transfer to the canister and the additional heating needed

//Operating Conditions
v = 23;               //[m/s] Air velocity
Tsurr = 296;      //[K] Surrounding Air Temperature
L = .8;             //[m] Length of cylinder
Di = .1;            //[m] Diameter of cylinder
t = .005;        //[m] Thickness of cylinder

//Table A.4 Air Properties at T = 285K
uv = 14.56*10^-6;          //[m^2/s] Kinematic Viscosity
k = 25.2*10^-3;            //[W/m.K] Thermal conductivity
Pr = .712;                 //Prandtl Number
//Table A.1 AISI 316 Stainless steel Properties at T = 300K
kss = 13.4;                 //[W/m.K]Conductivity

pH2 = 1.01;         //[N]
Ti = -3550/(2.30*log10(pH2) - 12.9);
Eg = -(1.35*10^-4)*(29.5*10^6);
```

```

Re = v*(Di+2*t)/uv;         //Reynolds number
// Equation 7.54
Nu = .3+.62*Re^.5*Pr^.3334/[1+(.4/Pr)^.6668]^.25*[1+(Re/282000)^(5/8)]^.8;
h = Nu*k/(Di+2*t);

qconv = (Tsurr-Ti)/[(1/(%pi*L*(Di+2*t)*h))+(2.30*log10((Di+2*t)/Di)/(2*%pi*kss*L))];
printf("\n Additional Thermal Energy must be supplied to canister to mainatin steady-state operating temperatue %i W",-qconv-Eg);

//END
```

Scilab code Exa 7.6 Time required to Cool on Plastic Film

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 7.6   Page 434 \n'); //Example 7.6
// Time required to cool from Ti = 75 degC to 35 degC

//Operating Conditions
v = 10;               //[m/s] Air velocity
Tsurr = 23+273;       //[K] Surrounding Air Temperature
D = .01;            //[m] Diameter of sphere
Ti = 75+273;          //[K] Initial temp
Tt = 35+273;          //[K] Temperature after time t
p = 1;                  //[atm]

//Table A.1 Copper at T = 328K
rho = 8933;           //[kg/m^3] Density
```

```
k = 399;             //[W/m.K] Conductivity
cp = 388;            //[J/kg.K] specific
//Table A.4 Air Properties T = 296 K
u = 182.6*10^-7;           //[N.s/m^2] Viscosity
uv = 15.53*10^-6;           //[m^2/s] Kinematic Viscosity
k = 25.1*10^-3;             //[W/m.K] Thermal conductivity
Pr = .708;                  //Prandtl Number
//Table A.4 Air Properties T = 328 K
u2 = 197.8*10^-7;           //[N.s/m^2] Viscosity

Re = v*D/uv;          //Reynolds number
//Using Equation 7.56
Nu = 2+(0.4*Re^.5 + 0.06*Re^.668)*Pr^.4*(u/u2)^.25;
h = Nu*k/D;
//From equation 5.4 and 5.5
t = rho*cp*D*2.30*log10((Ti-Tsurr)/(Tt-Tsurr))/(6*h);

printf("\nTime required for cooling is %.1f sec",t);

//END
```

Scilab code Exa 7.7 Air side Convection coefficient and Heat Rate for Staggered Arrangement

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 7.7    Page 443 \n'); //Example 7.7
// Air side Convection coefficient and Heat rate
// pressure Drop

```

```
//Operating Conditions
v = 6;              //[m/s] Air velocity
Tsurr = 15+273;     //[K] Surrounding Air Temperature
D = .0164;          //[m] Diameter of tube
Ts = 70+273;         //[K] Temp of tube
//Staggered arrangement dimensions
St = .0313;         //[m]
Sl = .0343;         //[m]

//Table A.4 Air Properties T = 288 K
rho = 1.217;         //[kg/m^3] Density
cp = 1007;           //[J/kg.K] specific heat
uv = 14.82*10^-6;          //[m^2/s] Kinematic Viscosity
k = 25.3*10^-3;            //[W/m.K] Thermal conductivity
Pr = .71;                 //Prandtl Number
//Table A.4 Air Properties T = 343 K
Pr2 = .701;                 //Prandtl Number
//Table A.4 Air Properties T = 316 K
uv3 = 17.4*10^-6;          //[m^2/s] Kinematic Viscosity
k3 = 27.4*10^-3;            //[W/m.K] Thermal conductivity
Pr3 = .705;                 //Prandtl Number

Sd = [Sl^2 + (St/2)^2]^.5;
Vmax = St*v/(St-D);

Re = Vmax*D/uv;        //Reynolds number

C = .35*(St/Sl)^.2;
m = .6;
C2 = .95;
N = 56;
Nt = 8;
//Using Equation 7.64 & 7.65
Nu = C2*C*Re^m*Pr^.36*(Pr/Pr2)^.25;
```

```
h = Nu*k/D;

//From Eqnn 7.67
Tso = (Ts-Tsurr)*exp(-(%pi*D*N*h)/(rho*v*Nt*St*cp));
Tlm = ((Ts-Tsurr) - Tso)/(2.30*log10((Ts-Tsurr)/Tso));
q = N*(h*%pi*D*Tlm);

Pt = St/D;
//From Fig 7.14
X = 1.04;
f = .35;
NL = 7;
press = NL*X*(rho*Vmax^2/2)*f;

printf("\n Air side Convection coefficient h = %.1f W/m^2.k and Heat rate q = %.1f kW/m \n Pressure Drop = %.2e bars",h,q/1000,press/100000);

//END
```

Chapter 8

Internal Flow

Scilab code Exa 8.1 Theoretical Problem

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   8.1    Page 494 \n')// Example 8.1
//Theoretical Problem

printf('\n The given example is theoretical and does
    not involve any numerical computation')

//End
```

Scilab code Exa 8.2 Length of Tube and Local Convection Coefficient at the Outlet

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
```

```
Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 8.2   Page 499 \n'); //Example 8.2
// Length of tube needed to achieve the desired outlet temperature
//Local convection coefficient at the outlet

//Operating Conditions
m = .1;                //[kg/s] mass flow rate of water
Ti = 20+273;           //[K] Inlet temp
To = 60+273;           //[K] Outlet temperature
Di = .02;              //[m] Inner Diameter
Do = .04;              //[m] Outer Diameter
q = 10^6;              //[w/m^3]  Heat generation Rate
Tsi = 70+273;          //[K] Inner Surface Temp
//Table A.4 Air Properties T = 313 K
cp = 4179;              //[J/kg.K] specific heat

L = 4*m*cp*(To-Ti)/(%pi*(Do^2-Di^2)*q);

//From Newtons Law of cooling, Equation 8.27, local heat convection coefficient is
h = q*(Do^2-Di^2)/(Di*4*(Tsi-To));

printf("\n Length of tube needed to achieve the desired outlet temperature = %.1f m \n Local convection coefficient at the outlet = %i W/m^2.K ",L,h);

//END
```

Scilab code Exa 8.3 Average Convection Coefficient of Stream

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
```

```
Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 8.3    Page 503 \n'); //Example 8.3
// average convection coefficient

//Operating Conditions
m = .25;             //[kg/s] mass flow rate of water
Ti = 15+273;        //[K] Inlet temp
To = 57+273;        //[K] Outlet temperature
D = .05;           //[m] Diameter
L = 6;           //[m] Length of tube
Ts = 100+273;       //[K] outer Surface Temp

//Table A.4 Air Properties T = 309 K
cp = 4178;           //[J/kg.K] specific heat

Tlm = ((Ts-To)-(Ts-Ti))/(2.30*log10((100-57)/(100-15)));

h = m*cp*(To-Ti)/(%pi*D*L*Tlm);

printf("\n Average Heat transfer Convection Coefficient = %i W/m^2.K",h);

//END
```

Scilab code Exa 8.4 Solar Energy

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 8.4    Page 506 \n'); //Example 8.4
// Length of tube for required heating
// Surface temperature Ts at outlet section

```

```
//Operating Conditions
m = .01;                //[kg/s] mass flow rate of water
Ti = 20+273;          //[K] Inlet temp
To = 80+273;          //[K] Outlet temperature
D = .06;             //[m] Diameter
q = 2000;             //[W/m^2] Heat flux to fluid

//Table A.4 Air Properties T = 323 K
cp = 4178;              //[J/kg.K] specific heat
//Table A.4 Air Properties T = 353 K
k = .670;               //[W/m] Thermal Conductivity
u = 352*10^-6;        //[N.s/m^2] Viscosity
Pr = 2.2;               //Prandtl Number
cp = 4178;              //[J/kg.K] specific heat

L = m*cp*(To-Ti)/(%pi*D*q);

//Using equation 8.6
Re = m*4/(%pi*D*u);
printf("\n (a) Length of tube for required heating = %.2f m\n\n (b)As Reynolds Number is %i. The flow is laminar.",L,Re);

Nu = 4.364;           //Nusselt Number
h = Nu*k/D;           //[W/m^2.K] Heat convection Coefficient

Ts = q/h+To;            //[K]

printf("\n Surface Temperature at tube outlet = %i degC",Ts-273);

//END
```

Scilab code Exa 8.5 Length of Blood Vessel Artery

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   8.5    Page 509 \n'); //Example 8.5
// Length of Blood Vessel

//Operating Conditions
um1 = .13;            //[m/s] Blood stream
um2 = 3*10^-3;            //[m/s] Blood stream
um3 = .7*10^-3;            //[m/s] Blood stream
D1 = .003;           //[m] Diameter
D2 = .02*10^-3;           //[m] Diameter
D3 =  .008*10^-3;           //[m] Diameter
Tlm = .05;
kf = .5;           //[W/m.K] Conductivity
//Table A. Water Properties T = 310 K
rho = 993;         //[kg/m^3] density
cp = 4178;          //[J/kg.K] specific heat
u = 695*10^-6;      //[N.s/m^2] Viscosity
kb = .628;           //[W/m.K] Conductivity
Pr = 4.62;          //Prandtl Number
i=1;
//Using equation 8.6
    Re1 = rho*um1*D1/u;
    Nu = 4;
    hb = Nu*kb/D1;
    hf = kf/D1;
    U1 = (1/hb + 1/hf)^-1;
    L1 = -rho*um1*D1/U1*cp*2.303*log10(Tlm)/4;
    xfdh1 = .05*Re1*D1;
    xfdr1 = xfdh1*Pr;

    Re2 = rho*um2*D2/u;
    Nu = 4;
    hb = Nu*kb/D2;
    hf = kf/D2;
    U2 = (1/hb + 1/hf)^-1;
```

```
    L2 = -rho*um2*D2/U2*cp*2.303*log10(Tlm)/4;
    xfdh2 = .05*Re2*D2;
    xfdr2 = xfdh2*Pr;

    Re3 = rho*um3*D3/u;
    Nu = 4;
    hb = Nu*kb/D3;
    hf = kf/D3;
    U3 = (1/hb + 1/hf)^-1;
    L3 = -rho*um3*D3/U3*cp*2.303*log10(Tlm)/4;
    xfdh3 = .05*Re3*D3;
    xfdr3 = xfdh3*Pr;

printf("\n Vessel          Re         U(W/m^2.K)      L(m)        xfdh(m)      xfdr(m)\n Artery            %i           %i            %.1f            %.2f          %.1f \n Anteriole         %.3f       %i       %.1e       %.1e       %.1e \n Capillary         %.3f       %i       %.1e       %.1e       %.1e",Re1,U1,L1,xfdh1,xfdr1,Re2,U2,L2,xfdh2,xfdr2,Re3,U3,L3,xfdh3,xfdr3);

//END
```

Scilab code Exa 8.6 Heat Loss from the Metal Duct over the Length

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 8.6    Page 516 \n'); //Example 8.6
// Heat Loss from the Duct over the Length L, q
// Heat flux and suface temperature at x=L

//Operating Conditions
m = .05;             //[kg/s] mass flow rate of water
```

```
Ti = 103+273;        //[K] Inlet temp
To = 77+273;         //[K] Outlet temperature
D = .15;            //[m] Diameter
L = 5;              //[m] length
ho = 6;              //[W/m^2.K] Heat transfer convective coefficient
Tsurr = 0+273;      //[K] Temperature of surrounding

//Table A.4 Air Properties T = 363 K
cp = 1010;            //[J/kg.K] specific heat
//Table A.4 Air Properties T = 350 K
k = .030;             //[W/m] Thermal Conductivity
u = 20.82*10^-6;       //[N.s/m^2] Viscosity
Pr = .7;             //Prandtl Number

q = m*cp*(To-Ti);

Re = m*4/(%pi*D*u);
printf(”\n As Reynolds Number is %i. The flow is Turbulent.”,Re);

//Equation 8.6
n = 0.3;
Nu = .023*Re^.8*Pr^.3;
h = Nu*k/D;
q2 = (To-Tsurr)/[1/h + 1/ho];
Ts = -q2/h+To;

printf(”\n\n Heat Loss from the Duct over the Length L, q = %i W \n Heat flux and suface temperature at x=L is %.1f W/m^2 & %.1f degC respectively”,q,q2,Ts-273);

//END
```

Scilab code Exa 8.7 Micro Channel

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 8.7    Page 525 \n'); //Example 8.5
// Time needed to bring the reactants to within 1 degC of processing temperature

//Operating Conditions
T1 = 125+273;     //[K] Chip Temperature 1
T2 = 25+273;     //[K] Chip Temperature 2
Ti = 5+273;     //[K] Inlet Temperature
D = .01;            //[m] Diameter
L = .02;              //[m] length
delP = 500*10^3;     //[N/m^2] Pressure drop
//Dimensions
a = 40*10^-6;
b = 160*10^-6;
s = 40*10^-6;

//Table A.5 Ethylene Glycol Properties T = 288 K
rho = 1120.2;                //[kg/m^3]     Density
cp = 2359;                   //[J/kg.K]     Specific Heat
u = 2.82*10^-2;            //[N.s/m^2] Viscosity
k = 247*10^-3;             //[W/m.K]   Thermal Conductivity
Pr = 269;                  //Prandtl number
//Table A.5 Ethylene Glycol Properties T = 338 K
rho2 = 1085;               //[kg/m^3]     Density
cp2 = 2583;                  //[J/kg.K]     Specific Heat
u2 = .427*10^-2;           //[N.s/m^2] Viscosity
k2 = 261*10^-3;            //[W/m.K]   Thermal Conductivity
Pr2 = 45.2;                  //Prandtl number

```

```
P = 2*a+2*b;                    //Perimeter of microchannel
Dh = 4*a*b/P;                   //Hydraulic Diameter

um2 = 2/73*Dh^2/u2*delP/L;         //[[m/s] Equation 8.22a
Re2 = um2*Dh*rho2/u2;          //Reynolds Number
xfdh2 = .05*Dh*Re2;          //[m] From Equation 8.3
xfdr2 = xfdh2*Pr2;          //[m] From Equation 8.23
m2 = rho2*a*b*um2;          //[kg/s]
Nu2 = 4.44;             //Nusselt Number from Table 8.1
h2 = Nu2*k2/Dh;          //[W/m^2.K] Convection Coeff
Tc2 = 124+273;          //[K]
xc2 = m2/P*cp2/h2*2.303*log10((T1-Ti)/(T1-Tc2));
tc2 = xc2/um2;

um = 2/73*Dh^2/u*delP/L;         //[[m/s] Equation 8.22a
Re = um*Dh*rho/u;          //Reynolds Number
xfdh = .05*Dh*Re;          //[m] From Equation 8.3
xfdr = xfdh*Pr;          //[m] From Equation 8.23
m = rho2*a*b*um;          //[kg/s]
Nu = 4.44;             //Nusselt Number from Table 8.1
h = Nu*k/Dh;          //[W/m^2.K] Convection Coeff
Tc = 24+273;          //[K]
xc = m/P*cp/h*2.303*log10((T2-Ti)/(T2-Tc));
tc = xc/um;

printf("\n Temp [degC]                                  %i
                 %i\n\n Flow rate [m/s]
                      %.3f                 %.3f\n
   Reynolds Number                           %.1f
                 %.1f\n Hydrodynamic entrance Length
    [m]    %.1e           %.1e\n Thermal entrance
   Length [m]          %.1e               %.1e\n Mass Flow
   rate [kg/s]                  %.2e            %.2e\n
```

```
Convective Coeff [W/m^2.K]          %.2e          %.2e\n Transition Length [m]          %.2e          %.2e\n Required Time [s]          %.3f          %.3f",T2-273,T1-273,um,um2,Re,Re2,xfdh,xfdh2,xfdr,xfdr2,m,m2,h,h2,xc,xc2,tc,tc2);
//END
```

Scilab code Exa 8.8 Average mass trasnfer Convection Coefficient for the Tube

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 8.8    Page 529 \n'); //Example 8.8
// Average mass trasnfer convection coefficient for the tube

//Operating Conditions
m = .0003;              //[kg/s] mass flow rate of water
T = 25+273;      //[K] Temperature of surrounding and tube
D = .01;            //[m] Diameter
L = 1;              //[m] length

//Table A.4 Air Properties T = 298 K
uv = 15.7*10^-6;            //[m^2/s] Kinematic Viscosity
u = 18.36*10^-6;     //[N.s/m^2] Viscosity
//Table A.8 Ammonia-Air Properties T = 298 K
Dab = .28*10^-4;        //[m^2/s] Diffusion coeff
Sc = .56;

```

```
Re = m*4/(%pi*D*u);
printf("\n As Reynolds Number is %i. The flow is Laminar.",Re);

//Using Equation 8.57
Sh = 1.86*(Re*Sc*D/L)^.3334;
h = Sh*Dab/D;
printf("\n Average mass trasnfer convection coefficient for the tube %.3f m/s",h);

//END
```

Chapter 9

Free Convection

Scilab code Exa 9.1 Vertical Plate

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   9.1    Page 569 \n'); //Example 9.1
// Boundary Layer thickness at trailing edge.

//Operating Conditions
Ts = 70+273;     //[K] Surface Temperature
Tsurr = 25+273;     //[K] Surrounding Temperature
v1 = 0;             //[m/s] Velocity of free air
v2 = 5;             //[m/s] Velocity of free air
L = .25;             //[m] length

//Table A.4 Air Properties T = 320 K
uv = 17.95*10^-6;          //[m^2/s] Kinematic
   Viscosity
be = 3.12*10^-3;           //[K^-1]  Tf^-1
Pr = 269;                 // Prandtl number
g = 9.81;         //[m^2/s]gravitational constt

```

```
Gr = g*be*(Ts-Tsurr)*L^3/uv^2;
del = 6*L/(Gr/4)^.25;
printf("\n Boundary Layer thickness at trailing edge for no air stream %.3f m",del);

Re = v2*L/uv;
printf("\n\n For air stream at 5 m/s As the Reynolds Number is %.2e the free convection boundary layer is Laminar",Re);
del2 = 5*L/(Re)^.5;
printf("\n Boundary Layer thickness at trailing edge for air stream at 5 m/s is %.4f m",del2);
//END
```

Scilab code Exa 9.2 Heat Transfer by Convection Between Screen and Room air

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 9.2    Page 572 \n'); //Example 9.2
// Heat transfer by convection between screen and room air.

//Operating Conditions
Ts = 232+273;     //[K] Surface Temperature
Tsurr = 23+273;     //[K] Surrounding Temperature
L = .71;             //[m] length
w = 1.02;            //[m] Width

//Table A.4 Air Properties T = 400 K
k = 33.8*10^-3              ;//[W/m.K]
uv = 26.4*10^-6          ;//[m^2/s] Kinematic Viscosity
```

```
al = 38.3*10^-6          ;//[m^2/s]
be = 2.5*10^-3            ;//[K^-1]  Tf^-1
Pr = .69                 ;// Prandtl number
g = 9.81                  ;//[m^2/s] gravitational constt

Ra = g*be*(Ts-Tsurr)/al*L^3/uv;
printf("\n\n As the Rayleigh Number is %.2e the free convection boundary layer is turbulent",Ra);
//From equatiom 9.23
Nu = [.825 + .387*Ra^.16667/[1+(.492/Pr)^(9/16)]^(8/27)]^2;
h = Nu*k/L;
q = h*L*w*(Ts-Tsurr);

printf("\n Heat transfer by convection between screen and room air is %i W",q);
//END
```

Scilab code Exa 9.3 Heat Loss from Duct per Meter of Length

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 9.3    Page 577 \n'); //Example 9.3
// Heat Loss from duct per meter of length

//Operating Conditions
Ts = 45+273;     //[K] Surface Temperature
Tsurr = 15+273     ;//[K] Surrounding Temperature
H = .3             ;//[m] Height
w = .75            ;//[m] Width

//Table A.4 Air Properties T = 303 K
```

```
k = 26.5*10^-3             ;//[W/m.K]
uv = 16.2*10^-6          ;//[m^2/s] Kinematic Viscosity
al = 22.9*10^-6         ;//[m^2/s] alpha
be = 3.3*10^-3           ;//[K^-1]  Tf^-1
Pr = .71                ;// Prandtl number
g = 9.81                 ;//[m^2/s] gravitational constt

Ra = g*be*(Ts-Tsurr)/al*H^3/uv;    //Length = Height
//From equatiom 9.27
Nu = [.68 + .67*Ra^.25/[1+(.492/Pr)^(9/16)]^(4/9)];
//for Sides
hs = Nu*k/H;

Ra2 = g*be*(Ts-Tsurr)/al*(w/2)^3/uv;       //Length = w/2
//For top eq 9.31
ht = [k/(w/2)]*.15*Ra2^.3334;
//For bottom Eq 9.32
hb = [k/(w/2)]*.27*Ra2^.25;

q = (2*hs*H+ht*w+hb*w)*(Ts-Tsurr);

printf("\n Rate of heat loss per unit length of duct is %i W/m",q);
//END
```

Scilab code Exa 9.4 Heat Loss from Pipe per Meter of Length

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 9.4   Page 580 \n'); //Example 9.4
```

```
// Heat Loss from pipe per meter of length

//Operating Conditions
Ts = 165+273;     //[K] Surface Temperature
Tsurr = 23+273;     //[K] Surrounding Temperature
D = .1              ;//[m] Diameter
e = .85             ;// emissivity
stfncnstt=5.67*10^(-8)    ;// [W/m^2.K^4] - Stefan Boltzmann Constant

//Table A.4 Air Properties T = 303 K
k = 31.3*10^-3              ;//[W/m.K] Conductivity
uv = 22.8*10^-6           ;//[m^2/s] Kinematic Viscosity
al = 32.8*10^-6          ;//[m^2/s] alpha
be = 2.725*10^-3            ;//[K^-1]  Tf^-1
Pr = .697                  ;// Prandtl number
g = 9.81                   ;//[m^2/s] gravitational constt

Ra = g*be*(Ts-Tsurr)/al*D^3/uv;
//From equatiom 9.34
Nu = [.60 + .387*Ra^(1/6)/[1+(.559/Pr)^(9/16)]^(8/27)]^2;
h = Nu*k/D;

qconv = h*%pi*D*(Ts-Tsurr);
qrad = e*%pi*D*stfncnstt*(Ts^4-Tsurr^4);

printf("\n Rate of heat loss per unit length of pipe is %i W/m",qconv+qrad);
//END
```

Scilab code Exa 9.5 Radiation Shield

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 9.5   Page 592 \n'); //Example 9.5
// Heat Loss from pipe per unit of length
// Heat Loss if air is filled with glass-fiber blanket insulation

//Operating Conditions
To = 35+273     ;//[K] Shield Temperature
Ti = 120+273     ;//[K] Tube Temperature
Di = .1              ;//[m] Diameter inner
Do = .12              ;//[m] Diameter outer
L = .01              ;//[m] air gap insulation

//Table A.4 Air Properties T = 350 K
k = 30*10^-3              ;//[W/m.K] Conductivity
uv = 20.92*10^-6           ;//[m^2/s] Kinematic Viscosity
al = 29.9*10^-6          ;//[m^2/s] alpha
be = 2.85*10^-3            ;//[K^-1] Tf^-1
Pr = .7                  ;// Prandtl number
g = 9.81                  ;//[m^2/s] gravitational constt
//Table A.3 Insulation glass fiber T=300K
kins = .038                    ;//[W/m.K] Conductivity

Lc = 2*[2.303*log10(Do/Di)]^(4/3)/((Di/2)^-(3/5)+(Do/2)^-(3/5))^(5/3);
Ra = g*be*(Ti-To)/al*Lc^3/uv;
keff = .386*k*(Pr/(.861+Pr))^.25*Ra^.25;
q = 2*%pi*keff*(Ti-To)/(2.303*log10(Do/Di));

//From equatiom 9.58 and 3.27
qin = q*kins/keff;

printf("\n Heat Loss from pipe per unit of length is
```

```
%i W/m \n Heat Loss if air is filled with glass-
fiber blanket insulation %i W/m",q,qin);
//END
```

Chapter 10

Boiling and Condensation

Scilab code Exa 10.1 Boiling Water Pan

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 10.1    Page 632 \n'); //Example 10.1
// Power Required by electruc heater to cause boiling
// Rate of water evaporation due to boiling
// Critical Heat flux corresponding to the burnout point

//Operating Conditions
Ts = 118+273     ;//[K] Surface Temperature
Tsat = 100+273    ;//[K] Saturated Temperature
D = .3            ;//[m] Diameter of pan
g = 9.81          ;//[m^2/s] gravitaional constant
//Table A.6 Saturated water Liquid Properties T = 373 K
rhol = 957.9              ;//[kg/m^3] Density
cp = 4.217*10^3               ;//[J/kg] Specific Heat
u = 279*10^-6              ;//[N.s/m^2] Viscosity
```

```
Pr = 1.76                ;// Prandtl Number
hfg = 2257*10^3          ;//[J/kg] Specific Heat
si = 58.9*10^-3      ;//[N/m]
//Table A.6 Saturated water Vapor Properties T = 373 K
rhov = .5956             ;//[kg/m^3] Density

Te = Ts-Tsat;
//From Table 10.1
C = .0128;
n = 1;
q = u*hfg*[g*(rhol-rhov)/si]^.5*(cp*Te/(C*hfg*Pr^n))^3;
qs = q*%pi*D^2/4;

m = qs/hfg;

qmax = .149*hfg*rhov*[si*g*(rhol-rhov)/rhov^2]^.25;

printf("\n Boiling Heat transfer rate = %.1f kW \n Rate of water evaporation due to boiling = %i kg/h \n Critical Heat flux corresponding to the burnout point = %.2f MW/m^2",qs/1000,m*3600,qmax/10^6);
//END
```

Scilab code Exa 10.2 Power Dissipation per unith Length for the Horizontal Cylinder

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 10.2    Page 635 \n'); //Example 10.2
// Power Dissipation per unith length for the
```

```
cylinder, qs

//Operating Conditions
Ts = 255+273     ;//[K] Surface Temperature
Tsat = 100+273     ;//[K] Saturated Temperature
D = 6*10^-3              ;//[m] Diameter of pan
e = 1              ;// eimssivity
stfncnstt=5.67*10^(-8)     ;// [W/m^2.K^4] - Stefan Boltzmann Constant
g = 9.81           ;//[m^2/s] gravitaional constant
//Table A.6 Saturated water Liquid Properties T = 373 K
rhol = 957.9              ;//[kg/m^3] Density
hfg = 2257*10^3          ;//[J/kg] Specific Heat
//Table A.4 Water Vapor Properties T = 450 K
rhov = .4902              ;//[kg/m^3] Density
cpv = 1.98*10^3                ;//[J/kg.K] Specific Heat
kv = 0.0299                  ;//[W/m.K] Conductivity
uv = 15.25*10^-6               ;//[N.s/m^2] Viscosity

Te = Ts-Tsat;

hconv = .62*[kv^3*rhov*(rhol-rhov)*g*(hfg+.8*cpv*Te)/(uv*D*Te)]^.25;
hrad = e*stfncnstt*(Ts^4-Tsat^4)/(Ts-Tsat);

//From eqn 10.9 h^(4/3) = hconv^(4/3) + hrad*h^(1/3)
//Newton Raphson
h=250;         //Initial Assumption
while(1>0)
f = h^(4/3) - [hconv^(4/3) + hrad*h^(1/3)];
fd = (4/3)*h^(1/3) - [(1/3)*hrad*h^(-2/3)];
hn=h-f/fd;
if((hn^(4/3) - [hconv^(4/3) + hrad*hn^(1/3)])<=.01)
    break;
end;
h=hn;
```

```
end

q = h*%pi*D*Te;

printf("\n Power Dissipation per unith length for the cylinder , qs= %i W/m",q);
//END
```

Scilab code Exa 10.3 Heat Transfer and Condensation Rates

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 10.3    Page 648 \n'); //Example 10.3
// Heat Transfer and Condensation Rates

//Operating Conditions
Ts = 50+273     ;//[K] Surface Temperature
Tsat = 100+273     ;//[K] Saturated Temperature
D = .08              ;//[m] Diameter of pan
g = 9.81             ;//[m^2/s] gravitaional constant
L = 1                   //[m] Length
//Table A.6 Saturated Vapor Properties p = 1.0133 bars
rhov = .596               ;//[kg/m^3] Density
hfg = 2257*10^3           ;//[J/kg] Specific Heat
//Table A.6 Saturated water Liquid Properties T = 348 K
rhol = 975               ;//[kg/m^3] Density
cpl = 4193                ; //[J/kg.K] Specific Heat
kl = 0.668                   ;//[W/m.K] Conductivity
ul = 375*10^-6               ;//[N.s/m^2] Viscosity
uvl = ul/rhol;               ;//[N.s.m/Kg] Kinematic viscosity
```

```
Ja = cpl*(Tsat-Ts)/hfg;
hfg2 = hfg*(1+.68*Ja);
//Equation 10.43
Re = [3.70*kl*L*(Tsat-Ts)/(ul*hfg2*(uvl^2/g)^.33334)
   +4.8]^.82;

//From equation 10.41
hL = Re*ul*hfg2/(4*L*(Tsat-Ts));
q = hL*(%pi*D*L)*(Tsat-Ts);

m = q/hfg;
//Using Equation 10.26
del = [4*kl*ul*(Tsat-Ts)*L/(g*rhol*(rhol-rhov)*hfg2)
   ]^.25;


printf("\n Heat Transfer Rate = %.1f kW and
   Condensation Rates= %.4f kg/s \n And as del(L) %
   .3f mm << (D/2) %.2f m use of vertical cylinder
   correlation is justified",q/1000,m,del*1000,D/2);
//END
```

Scilab code Exa 10.4 Condensation Rate per unit Length of Tubes

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   10.4    Page 652 \n'); //Example 10.4
// Condensation rate per unit length of tubes

//Operating Conditions
Ts = 25+273     ;//[K] Surface Temperature
Tsat = 54+273     ;//[K] Saturated Temperature
D = .006          ;  //[m] Diameter of pan
```

```
g = 9.81          ;//[m^2/s] gravitaional constant
N = 20                // No of tubes

//Table A.6 Saturated Vapor Properties p = 1.015 bar
rhov = .098             ;//[kg/m^3] Density
hfg = 2373*10^3         ;//[J/kg] Specific Heat
//Table A.6 Saturated water Liquid Properties Tf = 312.5 K
rhol = 992             ;//[kg/m^3] Density
cpl = 4178               ;//[J/kg.K] Specific Heat
kl = 0.631                ; //[W/m.K] Conductivity
ul = 663*10^-6            ; //[N.s/m^2] Viscosity

Ja = cpl*(Tsat-Ts)/hfg;
hfg2 = hfg*(1+.68*Ja);
//Equation 10.46
h = .729*[g*rhol*(rhol-rhov)*kl^3*hfg2/(N*ul*(Tsat-Ts)*D)]^.25;
//Equation 10.34
m1 = h*(%pi*D)*(Tsat-Ts)/hfg2;

m = N^2*m1;

printf("\n For the complete array of tubes, the condensation per unit length is %.3f kg/s.m",m);
//END
```

Chapter 11

Heat Exchangers

Scilab code Exa 11.1 Tube Length to Achieve a Desired Hot Fluid Temperature in a Counter Flow Tube Heat Exchanger

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   11.1    Page 680 \n'); //Example 11.1
// Tube Length to achieve a desired hot fluid
   temperature

//Operating Conditions
Tho = 60+273     ;//[K] Hot Fluid outlet Temperature
Thi = 100+273    ; //[K] Hot Fluid intlet Temperature
Tci = 30+273     ;//[K] Cold Fluid intlet Temperature
mh = .1           ;//[kg/s] Hot Fluid flow rate
mc = .2           ;//[kg/s] Cold Fluid flow rate
Do = .045         ;//[m] Outer annulus
Di = .025         ;//[m] Inner tube

//Table A.5 Engine Oil Properties T = 353 K
cph = 2131                  ;//[J/kg.K] Specific Heat
kh = .138                   ; //[W/m.K] Conductivity
```

```
uh = 3.25*10^-2            ; //[N.s/m^2] Viscosity
//Table A.6 Saturated water Liquid Properties Tc = 308 K
cpc = 4178               ;//[J/kg.K] Specific Heat
kc = 0.625                ; //[W/m.K] Conductivity
uc = 725*10^-6            ; //[N.s/m^2] Viscosity
Pr = 4.85                 ;//Prandtl Number

q = mh*cph*(Thi-Tho);

Tco = q/(mc*cpc)+Tci;

T1 = Thi-Tco;
T2 = Tho-Tci;
Tlm = (T1-T2)/(2.30*log10(T1/T2));

//Through Tube
Ret = 4*mc/(%pi*Di*uc);
printf(”\n Flow through Tube has Reynolds Number as %i. Thus the flow is Turbulent”, Ret);
//Equation 8.60
Nut = .023*Ret^.8*Pr^.4;
hi = Nut*kc/Di;

//Through Shell
Reo = 4*mh*(Do-Di)/(%pi*uh*(Do^2-Di^2));
printf(”\n Flow through Tube has Reynolds Number as %i. Thus the flow is Laminar”, Reo);
//Table 8.2
Nuo = 5.63;
ho = Nuo*kh/(Do-Di);

U = 1/[1/hi+1/ho];
L = q/(U*%pi*Di*Tlm);

printf(”\n Tube Length to achieve a desired hot fluid temperature is %.1f m”,L);
//END
```

Scilab code Exa 11.2 Exterior Dimensions of Counter Flow Plate Heat Exchanger

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   11.2   Page 683 \n'); //Example 11.2
// Exterior Dimensions of heat Exchanger
// Pressure drops within the plate-type Heat
   exchanger with N=60 gaps

//Operating Conditions
Tho = 60+273    ;//[K] Hot Fluid outlet Temperature
Thi = 100+273    ;//[K] Hot Fluid intlet Temperature
Tci = 30+273    ;//[K] Cold Fluid intlet Temperature
mh = .1          ;//[kg/s] Hot Fluid flow rate
mc = .2          ;//[kg/s] Cold Fluid flow rate
Do = .045        ;//[m] Outer annulus
Di = .025        ;//[m] Inner tube

//Table A.5 Engine Oil Properties T = 353 K
cph = 2131                 ;//[J/kg.K] Specific Heat
kh = .138                   ;//[W/m.K] Conductivity
uh = 3.25*10^-2              ; //[N.s/m^2] Viscosity
rhoh = 852.1               ;//[kg/m^3] Density
//Table A.6 Saturated water Liquid Properties Tc =
   308 K
cpc = 4178                 ;//[J/kg.K] Specific Heat
kc = 0.625                   ;//[W/m.K] Conductivity
uc = 725*10^-6               ;//[N.s/m^2] Viscosity
Pr = 4.85                   ;//Prandtl Number
rhoc = 994                 ;//[kg/m^3] Density

```

```
q = mh*cph*(Thi-Tho);

Tco = q/(mc*cpc)+Tci;

T1 = Thi-Tco;
T2 = Tho-Tci;
Tlm = (T1-T2)/(2.30*log10(T1/T2));

N = linspace(20,80,100);
L = q/Tlm*[1/(7.54*kc/2)+1/(7.54*kh/2)]*(N^2-N)^-1;
clf();
plot(N,L);
xtitle("Size of Heat Xchanger vs Number of gaps", "Number of Gaps (N)", "L (m)");

N2 = 60;
L = q/((N2-1)*N2*Tlm)*[1/(7.54*kc/2)+1/(7.54*kh/2)];
a = L/N2;
Dh = 2*a          ;//Hydraulic Diameter [m]
//For water filled gaps
umc = mc/(rhoc*L^2/2);
Rec = rhoc*umc*Dh/uc;
//For oil filled gaps
umh = mh/(rhoh*L^2/2);
Reh = rhoh*umh*Dh/uh;
printf("\n Flow of the fluids has Reynolds Number as %.2f & %i. Thus the flow is Laminar for both", Reh,Rec);

//Equations 8.19 and 8.22a
delpc = 64/Rec*rhoc/2*umc^2/Dh*L         ;//For water
delph = 64/Reh*rhoh/2*umh^2/Dh*L         ;//For oil

//For example 11.1
L1 = 65.9;
Dh1c = .025;
Dh1h = .02;
Ret = 4*mc/(%pi*Di*uc);
```

```
f = (.790*2.30*log10(Ret)-1.64)^-2        ;//friction factor through tube Eqn 8.21
umc1 = 4*mc/(rhoc*%pi*Di^2);
delpc1 = f*rhoc/2*umc1^2/Dh1c*L1;
Reo = 4*mh*(Do-Di)/(%pi*uh*(Do^2-Di^2));
umh1 = 4*mh/(rhoh*%pi*(Do^2-Di^2));
delph1 = 64/Reo*rhoh/2*umh1^2/Dh1h*L1;

printf("\n Exterior Dimensions of heat Exchanger L = %.3f m \n Pressure drops within the plate-type Heat exchanger with N=60 gaps\n For water = %.2f N/m^2     For oil = %.2f N/m^2\n Pressure drops tube Heat exchanger of example 11.1\n For water = %.1f kN/m^2     For oil = %.1f kN/m^2",L,delpc,delph,delpc1/1000,delph1/1000);
//END
```

Scilab code Exa 11.3 Required Gas Side Surface Area in CrossFlow Finned Heat Exchanger

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 11.3    Page 692 \n'); //Example 11.3
// Required gas side surface area

//Operating Conditions
Tho = 100+273     ;//[K] Hot Fluid outlet Temperature
Thi = 300+273     ;//[K] Hot Fluid intlet Temperature
Tci = 35+273     ;//[K] Cold Fluid intlet Temperature
Tco = 125+273    ; //[K] Cold Fluid outlet Temperature
mc = 1          ;//[kg/s] Cold Fluid flow rate
Uh = 100        ;//[W/m^2.K] Coefficient of heat
```

```
transfer
//Table A.5 Water Properties T = 353 K
cph = 1000       ;       //[J/kg.K] Specific Heat
//Table A.6 Saturated water Liquid Properties Tc = 308 K
cpc = 4197        ;      //[J/kg.K] Specific Heat

Cc = mc*cpc;
//Equation 11.6b and 11.7b
Ch  = Cc*(Tco-Tci)/(Thi-Tho);
// Equation 11.18
qmax = Ch*(Thi-Tci);
//Equation 11.7b
q = mc*cpc*(Tco-Tci);

e = q/qmax;
ratio = Ch/Cc;

printf("\n As effectiveness is %.2f with Ratio Cmin/Cmax = %.2f, It follows from figure 11.14 that NTU = 2.1",e,ratio);
NTU = 2.1;
A = 2.1*Ch/Uh;

printf("\n Required gas side surface area = %.1f m^2",A);
//END
```

Scilab code Exa 11.4 Heat Transfer Rate and Fluid Outlet Temperatures of Cross Flow Finned Heat Exchanger

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
```

```
11.4    Page 695 \n’); //Example 11.4
// Heat Transfer Rate and Fluid Outlet Temperatures

//Operating Conditions
Thi = 250+273     ;//[K] Hot Fluid intlet Temperature
Tci = 35+273     ;//[K] Cold Fluid intlet Temperature
mc = 1           ;//[kg/s] Cold Fluid flow rate
mh = 1.5         ;  //[kg/s] Hot Fluid flow rate
Uh = 100         ;//[W/m^2.K] Coefficient of heat transfer
Ah = 40          ; //[m^2] Area
//Table A.5 Water Properties T = 353 K
cph = 1000        ;       //[J/kg.K] Specific Heat
//Table A.6 Saturated water Liquid Properties Tc = 308 K
cpc = 4197         ;      //[J/kg.K] Specific Heat

Cc = mc*cpc;
Ch  = mh*cph;
Cmin = Ch;
Cmax = Cc;

NTU = Uh*Ah/Cmin;
ratio = Cmin/Cmax;

printf(”\n As Ratio Cmin/Cmax = %.2f and Number of transfer units NTU = %.2f, It follows from figure 11.14 that e = .82”,ratio,NTU);
e = 0.82;
qmax = Cmin*(Thi-Tci);
q = e*qmax;

//Equation 11.6b
Tco = q/(mc*cpc) + Tci;
//Equation 11.7b
Tho = -q/(mh*cph) + Thi;
printf(”\n Heat Transfer Rate  = %.2e W \n Fluid Outlet Temperatures Hot Fluid (Tho) = %.1f degC
```

```
    Cold Fluid (Tco) = %.1f degC",q,Tho-273,Tco
    -273);
//END
```

Scilab code Exa 11.5 Study of Shell n Tube Heat Exchanger

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
    Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
    11.5    Page 696 \n'); //Example 11.5
// Outlet Temperature of cooling Water
// Tube length per pass to achieve required heat
    transfer

//Operating Conditions
q = 2*10^9          ;//[W] Heat transfer Rate
ho = 11000          ;//[W/m^2.K] Coefficient of heat
    transfer for outer surface
Thi = 50+273      ;//[K] Hot Fluid Condensing
    Temperature
Tho = Thi      ;//[K] Hot Fluid Condensing Temperature
Tci = 20+273      ;//[K] Cold Fluid intlet Temperature
mc = 3*10^4      ;      //[kg/s] Cold Fluid flow rate
m =  1            ;//[kg/s] Cold Fluid flow rate per
    tube
D = .025          ;//[m] diameter of tube
//Table A.6 Saturated water Liquid Properties Tf =
    300 K
rho = 997          ;      //[kg/m^3] Density
cp = 4179          ;        //[J/kg.K] Specific Heat
k = 0.613          ;          //[W/m.K] Conductivity
u = 855*10^-6      ;          //[N.s/m^2] Viscosity
Pr = 5.83          ;          // Prandtl number

```

```
//Equation 11.6b
Tco = q/(mc*cp) + Tci;

Re = 4*m/(%pi*D*u);
printf("\n As the Reynolds number of tube fluid is %i. Hence the flow is turbulent. Hence using Diettus-Boetllor Equation 8.60", Re);
Nu = .023*Re^.8*Pr^.4;
hi = Nu*k/D;
U = 1/[1/ho + 1/hi];
N = 30000             ;//No of tubes
T1 = Thi-Tco;
T2 = Tho-Tci;
Tlm = (T1-T2)/(2.30*log10(T1/T2));
L2 = q/(U*N*2*%pi*D*Tlm);


printf("\n Outlet Temperature of cooling Water = %.1f degC\n Tube length per pass to achieve required heat transfer = %.2f m",Tco-273,L2);
//END
```

Scilab code Exa 11.6 Finned Compact Heat Exchanger

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 11.6    Page 702 \n'); //Example 11.6
// Gas-side overall heat transfer coefficient. Heat exchanger Volume

//Operating Conditions
hc = 1500        ;//[W/m^2.K] Coefficient of heat transfer for outer surface
```

```
hi = hc;
Th = 825    ;//[K] Hot Fluid Temperature
Tci = 290    ;//[K] Cold Fluid intlet Temperature
Tco = 370    ;//[K] Cold Fluid outlet Temperature
mc = 1          ;//[kg/s] Cold Fluid flow rate
mh =  1.25          ;//[kg/s] Hot Fluid flow rate
Ah = .20            ;//[m^2] Area of tubes
Di = .0138        ;//[m] diameter of tube
Do = .0164        ;//[m] Diameter
//Table A.6 Saturated water Liquid Properties Tf = 330 K
cpw = 4184          ;     //[J/kg.K] Specific Heat
//Table A.1 Aluminium Properties T = 300 K
k = 237               ;    //[W/m.K] Conductivity
//Table A.4 Air Properties Tf = 700 K
cpa = 1075          ;     //[J/kg.K] Specific Heat
u = 33.88*10^-6     ;        //[N.s/m^2] Viscosity
Pr = .695           ;      // Prandtl number

//Geometric Considerations
si = .449;
Dh = 6.68*10^-3         ;//[m] hydraulic diameter
G = mh/si/Ah;
Re = G*Dh/u;
//From Figure 11.16
jh = .01;
hh = jh*G*cpa/Pr^.66667;

AR = Di*2.303*log10(Do/Di)/(2*k*(.143));
//Figure 11.16
AcAh = Di/Do*(1-.830);
//From figure 3.19
nf = .89;
noh = 1-(1-.89)*.83;

U = [1/(hc*AcAh) + AR + 1/(noh*hh)]^-1;

Cc = mc*cpw;
```

```
q = Cc*(Tco-Tci);
Ch = mh*cpa;
qmax = Ch*(Th-Tci);
e = q/qmax;
ratio = Ch/Cc;

printf("\n As effectiveness is %.2f with Ratio Cmin/Cmax = %.2f, It follows from figure 11.14 that NTU = .65",e,ratio);
NTU = .65;
A = NTU*Ch/U;
//From Fig 11.16
al = 269;             //[m^-1] gas side area per unit heat wxchanger volume
V = A/al;

printf("\n Gas-side overall heat transfer coefficient.r = %i W/m^2.K\n Heat exchanger Volume = %.3f m^3",U,V);
//END;
```

Chapter 12

Radiation Processes and Properties

Scilab code Exa 12.1 Plate Surface Emission Study

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   12.1    Page 731 \n')// Example 12.1

// a) Intensity of emission in each of the three
   directions
// b) Solid angles subtended by the three surfaces
// c) Rate at which radiation is intercepted by the
   three surfaces

A1 = .001        ;//[m^2] Area of emitter
In  = 7000       ;//[W/m^2.Sr] Intensity of radiation
    in normal direction
A2 = .001        ;//[m^2] Area of other intercepting
   plates
A3 = A2          ;//[m^2] Area of other intercepting
   plates
```

```
A4 = A2          ;//[m^2] Area of other intercepting plates
r = .5           ;//[m] Distance of each plate from emitter
theta1 = 60      ;//[deg] Angle between surface 1 normal & direction of radiation to surface 2
theta2 = 30      ;//[deg] Angle between surface 2 normal & direction of radiation to surface 1
theta3 = 45      ;//[deg] Angle between surface 1 normal & direction of radiation to surface 4

//From equation 12.2
w31 = A3/r^2;
w41 = w31;
w21 = A2*cos(theta2*0.0174532925)/r^2;


//From equation 12.6
q12 = In*A1*cos(theta1*0.0174532925)*w21;
q13 = In*A1*cos(0)*w31;
q14 = In*A1*cos(theta3*0.0174532925)*w41;

printf("\n (a) As Intensity of emitted radiation is independent of direction , for each of the three directions I = %i W/m^2.sr \n\n (b) By the Three Surfaces\n          Solid angles subtended          Rate at which radiation is intercepted \n          w4-1 = %.2e sr          q1-4 = %.1e W \n          w3-1 = %.2e sr          q1-3 = %.1e W\n          w2-1 = %.2e sr          q1-2 = %.1e W     ",In, w41,q14,w31,q13,w21,q12);
//END
```

Scilab code Exa 12.2 Total Irradiation of Spectral Distribution

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   12.2    Page 734\n')// Example 12.2

// Total Irradiation
x=[0 5 20 25];
y=[0 1000 1000 0];
clf();
plot2d(x,y,style=5,rect=[0,0,30,1100]);
xtitle("Spectral Distribution", "wavelength (micro-m
   )", "G (W/m^2.micro-m)");

//By Equation 12.4
G = 1000*(5-0)/2+1000*(20-5)+1000*(25-20)/2;

printf("\n G = %i W/m^2",G);
//END
```

Scilab code Exa 12.3 Blackbody Radiation

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   12.3    Page 741 \n')// Example 12.3

// Spectral Emissive Power of a small aperture on
   the enclosure
```

```
// wavelengths below which and above which 10% of the radiation is concentrated
// Spectral emissive power and wavelength associated with maximum emission
// Irradiation on a small object inside the enclosure

T = 2000            ;//[K] temperature of surface
stfncnstt = 5.67*10^-8      ;//[W/m^2.K^4] Stefan-Boltzmann constant
E = stfncnstt*T^4;        //[W/m^2]

//From Table 12.1
constt1 = 2195    ;    //[micro-m.K]
wl1 = constt1/T;
//From Table 12.1
constt2 = 9382   ;     //[micro-m.K]
wl2 = constt2/T;

//From Weins Law, wlmax*T = consttmax = 2898 micro-m.K
consttmax = 2898          ;//micro-m.K
wlmax = consttmax/T;
//from Table 12.1 at wlmax = 1.45 micro-m.K and T = 2000 K
I = .722*10^-4*stfncnstt*T^5;
Eb = %pi*I;

G = E;         //[W/m^2] Irradiation of any small object inside the enclosure is equal to emission from blackbody at enclosure temperature

printf("\n (a) Spectral Emissive Power of a small aperture on the enclosure = %.2e W/m^2.Sr for each of the three directions \n (b) Wavelength below which 10percent of the radiation is concentrated = %.1f micro-m \n      Wavelength above which 10percent of the radiation is
```

```
   concentrated = %.2f micro-m \n (c) Spectral
   emissive power and wavelength associated with
   maximum emission is %.2e micro-m and %.2e W/m^2.
   micro-m respectively \n (d) Irradiation on a
   small object inside the enclosure = %.2e W/m^2",E
   ,wl1,wl2,Eb,wlmax,G);
//END
```

Scilab code Exa 12.4 Blackbody Angular Radiation

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   12.4    Page 743 \n')// Example 12.4

// Rate of emission per unit area over all
   directions between 0 degC and 60 degC and over
   all wavelengths between wavelengths 2 and 4 micro
   -m

T = 1500            ;//[K] temperature of surface
stfncnstt = 5.67*10^-8      ;//[W/m^2.K^4] Stefan-
   Boltzmann constant

//From Equation 12.26 Black Body Radiation
Eb = stfncnstt*T^4;        //[W/m^2]

//From Table 12.1 as wl1*T = 2*1500 (micro-m.K)
F02 = .273;
//From Table 12.1 as wl2*T = 4*1500 (micro-m.K)
F04 = .738;

//From equation 12.10 and 12.11
i1 = integrate('2*cos(x)*sin(x)','x',0,%pi/3);
```

```
delE = i1*(F04-F02)*Eb;

printf("\n Rate of emission per unit area over all directions between 0 degC and 60 degC and over all wavelengths between wavelengths 2 micro-m and 4 micro-m = %.1e W/m^2",delE);
//END
```

Scilab code Exa 12.5 Diffuse Emitter

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 12.5    Page 748 \n')// Example 12.5

// Total hemispherical emissivity
// Total emissive Power
// Wavelength at which spectral emissive power will be maximum

T = 1600             ;//[K] temperature of surface
wl1 = 2             ;//[micro-m] wavelength 1
wl2 = 5             ;//[micro-m] wavelength 2
stfncnstt = 5.67*10^-8;      //[W/m^2.K^4] Stefan-Boltzmann constant
// From the given graph of emissivities
e1 = .4;
e2 = .8;
//From Equation 12.26 Black Body Radiation
Eb = stfncnstt*T^4;        //[W/m^2]

//Solution (A)
//From Table 12.1 as wl1*T = 2*1600 (micro-m.K)
F02 = .318;
```

```
//From Table 12.1 as wl2*T = 5*1600 (micro-m.K)
F05 = .856;
//From Equation 12.36
e = e1*F02 + e2*[F05 - F02];

//Solution (B)
//From equation 12.35
E = e*Eb;

//Solution (C)
//For maximum condition Using Weins Law
consttmax = 2898          ;//[micro-m.K]
wlmax = consttmax/T;

//equation 12.32 with Table 12.1
E1 = %pi*e1*.722*10^-4*stfncnstt*T^5;

E2 = %pi*e2*.706*10^-4*stfncnstt*T^5;

printf("\n (a) Total hemispherical emissivity = %.3f \n (b) Total emissive Power = %i kW/m^2 \n (c) Emissive Power at wavelength 2micro-m is greater than Emissive power at maximum wavelength \n i.e. %.1f kW/m^2 > %.1f kW/m^2 \n      Thus, Peak emission occurs at %i micro-m",e,E/1000,E2/1000,E1/1000,wl1);
//END
```

Scilab code Exa 12.6 Metallic Surface Irradiation

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 12.6    Page 751 \n')// Example 12.6
```

```

// Spectral , Normal emissivity en and spectral hemispherical emissivity e
// Spectral normal intensity In and Spectral emissive power

T = 2000              ;//[K] temperature of surface
wl = 1              ;//[micro-m] wavelength
stfncnstt = 5.67*10^-8;      //[W/m^2.K^4] Stefan-Boltzmann constant

// From the given graph of emissivities
e1 = .3;
e2 = .6;
//From Equation 12.26 Black Body Radiation
Eb = stfncnstt*T^4;        //[W/m^2]

//Equation 12.34
i1 = integrate('e1*cos(x)*sin(x)','x',0,%pi/3);
i2 = integrate('e2*cos(x)*sin(x)','x',%pi/3,4*%pi/9);
e = 2*[i1+i2];

// From Table 12.1 at wl = 1 micro-m and T = 2000 K.

I = .493*10^-4 * stfncnstt*T^5        ;//[W/m^2.micro-m.sr]

In = e1*I;

//Using Equation 12.32 for wl = 1 micro-m and T = 2000 K
E = e*%pi*I;

printf('\n Spectral Normal emissivity en = %.1f and spectral hemispherical emissivity e = %.2f \n Spectral normal intensity In = %.2e W/m^2.micro-m.sr and Spectral emissive power = %.1e W/m^2.
```

```
    micro-m.sr  ', e1, e,In,E);
```

Scilab code Exa 12.7 Study of Radiation on Opaque Surface

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
    Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
    12.7    Page 759 \n')// Example 12.7

// Spectral distribution of reflectivity
// Total, hemispherical absorptivity
// Nature of surface temperature change

T = 500          ;//[K] temperature of surface
e = .8;
stfncnstt = 5.67*10^-8;      //[W/m^2.K^4] Stefan-
    Boltzmann constant

x=[0 6 8 16];
y=[.8 .8 0 0];
clf();
plot2d(x,y,style=5,rect=[0,0,20,1]);


xtitle("Spectral Distribution of reflectivity", "
    wavelength (micro-m)", "reflectivity");

//From equation 12.43 and 12.44
Gabs = {.2*500/2*(6-2)+500*[.2*(8-6)+(1-.2)*(8-6)
    /2]+1*500*(12-8)+500*(16-12)/2}            ;//[w/
   m^2]
G = {500*(6-2)/2+500*(12-6)+500*(16-12)/2}
                ;//[w/m^2]
a = Gabs/G;
```

```

//Neglecting convection effects net het flux to the surface
qnet = a*G - e*stfncnstt*T^4;

printf('\n Total, hemispherical absorptivity %.2f \n Nature of surface temperature change = %i W/m^2 \n Since qnet > 0, the sirface temperature will increase with the time', a,qnet);
```

Scilab code Exa 12.8 Total Emissivity of Cover Glass to Solar Radiation

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 12.8    Page 761 \n')// Example 12.8

// Total emissivity of cover glass to solar radiation

T = 5800            ;//[K] temperature of surface
e = .8;
stfncnstt = 5.67*10^-8;      //[W/m^2.K^4] Stefan-Boltzmann constant

//From Table 12.1
//For wl1 = .3 micro-m and T = 5800 K, At wl1*T = 1740 micro-m.K
F0wl1 = .0335;
//For wl1 = .3 micro-m and T = 5800 K, At wl2*T = 14500 micro-m.K
F0wl2 = .9664;

//Hence from equation 12.29
```

```
t = .90*[F0wl2 - F0wl1];

printf('\n Total emissivity of cover glass to solar radiation = %.2f',t);
```

Scilab code Exa 12.9 Total Hemispherical Emissivity of Fire Brick Wall

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 12.9    Page 766 \n')// Example 12.9

// Total hemispherical emissivity of fire brick wall
// Total emissive power of brick wall
// Absorptivity of the wall to irradiation from coals

Ts = 500                ;//[K] temperature of brick surface
Tc = 2000               ;//[K] Temperature of coal exposed
stfncnstt = 5.67*10^-8;      //[W/m^2.K^4] Stefan-Boltzmann constant
// From the given graph of emissivities
e1 = .1;    //between wavelength 0 micro-m- 1.5 micro-m
e2 = .5;    //between wavelength 1.5 micro-m- 10 micro-m
e3 = .8;    //greater than wavelength 10 micro-m

//From Table 12.1
//For wl1 = 1.5 micro-m and T = 500 K, At wl1*T = 750 micro-m.K
F0wl1 = 0;
```

```
//For wl2 = 10 micro-m and T = 500 K, At wl2*T = 5000 micro-m.K
F0wl2 = .634;
//From equation 12.36
e = e1*F0wl1 + e2*F0wl2 + e3*(1-F0wl1-F0wl2);

//Equation 12.26 and 12.35
E = e*stfncnstt*Ts^4;

//From Table 12.1
//For wl1 = 1.5 micro-m and T = 2000 K, At wl1*T = 3000 micro-m.K
F0wl1c = 0.273;
//For wl2 = 10 micro-m and T = 2000 K, At wl2*T = 20000 micro-m.K
F0wl2c = .986;
ac = e1*F0wl1c + e2*[F0wl2c-F0wl1c] + e3*(1-F0wl2c);

printf('\n Total hemispherical emissivity of fire brick wall = %.3f \n Total emissive power of brick wall = %i W/m^2.\n Absorptivity of the wall to irradiation from coals = %.3f',e,E,ac);
```

Scilab code Exa 12.10 Total Hemispherical Absorptivity and Emissivity of Metallic Sphere

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 12.10    Page 768 \n')// Example 12.10

// Total hemispherical absorptivity and emissivity of sphere for initial condition
// values of absoprtivity and emissivity after
```

```
sphere has been in furnace a long time

Ts = 300;           //[K] temperature of surface
Tf = 1200;           //[K] Temperature of Furnace
stfncnstt = 5.67*10^-8;     //[W/m^2.K^4] Stefan-Boltzmann constant
// From the given graph of absorptivities
a1 = .8;    //between wavelength 0 micro-m- 5 micro-m
a2 = .1;    //greater than wavelength 5 micro-m

//From Table 12.1
//For wl1 = 5 micro-m and T = 1200 K, At wl1*T = 6000 micro-m.K
F0wl1 = 0.738;
//From equation 12.44
a = a1*F0wl1 + a2*(1-F0wl1);
//From Table 12.1
//For wl1 = 5 micro-m and T = 300 K, At wl1*T = 1500 micro-m.K
F0wl1s = 0.014;
//From equation 12.36
e = a1*F0wl1s + a2*(1-F0wl1s);

printf('\n For Initial Condition \n Total hemispherical absorptivity = %.2f     Emissivity of sphere = %.2f \n\n Beacuase the spectral characteristics of the coating and the furnace temeprature remain fixed, there is no change in the value of absorptivity with increasing time. \n Hence, After a sufficiently long time, Ts = Tf = %i K and emissivity equals absorptivity e = a = %.2f',a,e,Tf,a);
```

Scilab code Exa 12.11 Heat Removal Rate per Unit Area of Solar Collector

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 12.11    Page 774 \n')// Example 12.11

// Useful heat removal rate per unit area
// Efficiency of the collector

Ts = 120+273;              //[K] temperature of surface
Gs = 750;                    //[W/m^2] Solar irradiation
Tsky = -10+273;            //[K] Temperature of Sky
Tsurr = 30+273;            //[K] Temperature os surrounding Air
e = .1                     ;// emissivity
as = .95                   ;// Absorptivity of Surface
asky = e                   ;// Absorptivity of Sky
stfncnstt = 5.67*10^-8;      //[W/m^2.K^4] Stefan-Boltzmann constant
h = 0.22*(Ts - Tsurr)^.3334    ;//[W/m^2.K] Convective Heat transfer Coeff
//From equation 12.67
Gsky = stfncnstt*Tsky^4;       //[W/m^2] Irradiadtion from sky
qconv = h*(Ts-Tsurr);        //[W/m^2]  Convective Heat transfer
E = e*stfncnstt*Ts^4;        //[W/m^2]  Irradiadtion from Surface

//From energy Balance
q = as*Gs + asky*Gsky - qconv - E;

//Collector efficiency
eff = q/Gs;
```

```

printf('\n Useful heat removal rate per unit area by Energy Conservation = %i W/m^2 \n Collector efficiency defined as the fraction of solar irradiation extracted as useful energy is %.2f',q,eff);
```

Chapter 13

Radiation Exchange between the Surface

Scilab code Exa 13.1 Theoretical Problem

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   13.1   Page 820 \n')// Example 13.1
//Theoretical Problem

printf('\n The given example is theoretical and does
    not involve any numerical computation')

//End
```

Scilab code Exa 13.2 View Factor of Different Geometries

```
clear;
clc;
```

```
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 13.2    Page 821 \n')// Example 13.2

// View Factors of known surface Geometries

// (1) Sphere within Cube
F12a = 1                        ;//By Inspection
F21a = (%pi/6)*F12a             ; //By Reciprocity

// (2) Partition within a Square Duct
F11b = 0                        ;//By Inspection
//By Symmetry F12 = F13
F12b = (1-F11b)/2        ;       //By Summation Rule
F21b = sqrt(2)*F12b       ;      //By Reciprocity

// (3) Circular Tube
//From Table 13.2 or 13.5, with r3/L = 0.5 and L/r1 = 2
F13c = .172;
F11c = 0;                        //By Inspection
F12c = 1 - F11c - F13c          ;//By Summation Rule
F21c = F12c/4                   ;//By Reciprocity

printf('\n Desired View Factors may be obtained from inspection, the reciprocity rule, the summation rule and/or use of charts \n (1) Sphere within Cube F21 = %.3f \n (2) Partition within a Square Duct F21 = %.3f \n (3) Circular Tube F21 = %.3f',F21a,F21b,F21c);
```

Scilab code Exa 13.3 Net rate of Heat transfer to the absorber surface

```
clear;
clc;
```

```
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 13.3    Page 826 \n')// Example 13.3

// Net rate of Heat transfer to the absorber surface

L = 10        ;//[m] Collector length = Heater Length
T2 = 600      ;//[K] Temperature of curved surface
A2 = 15      ;//[m^2] Area of curved surface
e2 = .5      ;// emissivity of curved surface
stfncnstt = 5.67*10^-8;     //[W/m^2.K^4] Stefan-Boltzmann constant
T1 = 1000        ;//[K] Temperature of heater
A1 = 10          ;//[m^2] area of heater
e1 = .9          ;// emissivity of heater
W = 1            ;//[m] Width of heater
H = 1            ;//[m] Height
T3 = 300         ;//[K] Temperature of surrounding
e3 = 1           ;// emissivity of surrounding

J3 = stfncnstt*T3^4;        //[W/m^2]
//From Figure 13.4 or Table 13.2, with Y/L = 10 and X/L =1
F12 = .39;
F13 = 1 - F12;        //By Summation Rule
//For a hypothetical surface A2h
A2h = L*W;
F2h3 = F13;        //By Symmetry
F23 = A2h/A2*F13;         //By reciprocity
Eb1 = stfncnstt*T1^4;    //[W/m^2]
Eb2 = stfncnstt*T2^4;    //[W/m^2]
//Radiation network analysis at Node corresponding 1
//-10J1 + 0.39J2 = -510582
//.26J1 - 1.67J2 = -7536
//Solving above equations
A = [-10 .39;
     .26 -1.67];
```

```
B = [-510582;
     -7536];

X = inv(A)*B;

q2 = (Eb2 - X(2))/(1-e2)*(e2*A2);

printf('\n Net Heat transfer rate to the absorber is = %.1f kW',q2/1000);
```

Scilab code Exa 13.4 Power Required to Maintain Prescribed Temperatures in Cylindrical Furnace

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 13.4    Page 830 \n')// Example 13.4

// Power required to maintain prescribed temperatures

T3 = 300          ;//[K] Temperature of surrounding
L = .15         ;//[m] Furnace Length
T2 = 1650+273       ;//[K] Temperature of bottom surface
T1 = 1350+273       ;//[K] Temperature of sides of furnace
D = .075            ;//[m] Diameter of furnace
stfncnstt = 5.670*10^-8;        //[W/m^2.K^4] Stefan Boltzman Constant
A2 = %pi*D^2/4         ;//[m] Area of bottom surface
A1 = %pi*D*L         ;//[m] Area of curved sides
//From Figure 13.5 or Table 13.2, with ri/L = .25
F23 = .056;
```

```
F21 = 1 - F23;          //By Summation Rule
F12 = A2/A1*F21;            //By reciprocity
F13 = F12               ;//By Symmetry
//From Equation 13.17 Heat balance
q = A1*F13*stfncnstt*(T1^4 - T3^4) + A2*F23*stfncnstt*(T2^4 - T3^4);

printf('\n Power required to maintain prescribed temperatures is = %i W',q);
```

Scilab code Exa 13.5 Concentric Tube Arrangement

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 13.5    Page 834 \n')// Example 13.5

// Heat gain by the fluid passing through the inner tube
// Percentage change in heat gain with radiation shield inserted midway between inner and outer tubes

T2 = 300        ;//[K] Temperature of inner surface
D2 = .05        ;//[m] Diameter of Inner Surface
e2 = .05        ;// emissivity of Inner Surface
T1 = 77        ;//[K] Temperature of Outer Surface
D1 = .02              ;//[m] Diameter of Inner Surface
e1 = .02        ;// emissivity of Outer Surface
D3 = .035          ;//[m] Diameter of Shield
e3 = .02          ;// emissivity of Shield
stfncnstt = 5.670*10^-8          ;//[W/m^2.K^4] Stefan Boltzman Constant

```

```
//From Equation 13.20 Heat balance
q = stfncnstt*(%pi*D1)*(T1^4-T2^4)/(1/e1 + (1-e2)/e2*D1/D2) ;// [W/m]

RtotL = (1-e1)/(e1*%pi*D1) + 1/(%pi*D1*1) + 2*[(1-e3)/(e3*%pi*D3)] + 1/(%pi*D3*1) + (1-e2)/(e2*%pi*D2)    ;// [m^-2]
q2 = stfncnstt*(T1^4 - T2^4)/RtotL;    // [W/m]

printf('\n Heat gain by the fluid passing through the inner tube = %.2f W/m \n Percentage change in heat gain with radiation shield inserted midway between inner and outer tubes is = %.2f percent',q,(q2-q)*100/q);
```

Scilab code Exa 13.6 Rate at which Heat must be Supplied per Unit Length of Triangular Duct

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 13.6    Page 836 \n')// Example 13.6

// Rate at which heat must be supplied per unit length of duct
// Temperature of the insulated surface

T2 = 500      ;//[K] Temperature of Painted surface
e2 = .4      ;// emissivity of Painted Surface
T1 = 1200      ;//[K] Temperature of Heated Surface
W = 1            ; //[m] Width of Painted Surface
e1 = .8      ;// emissivity of Heated Surface
er = .8        ;// emissivity of Insulated Surface
stfncnstt = 5.670*10^-8        ;//[W/m^2.K^4] Stefan
```

```
    Boltzman Constant

//By Symmetry Rule
F2R = .5;
F12 = .5;
F1R = .5;

//From Equation 13.20 Heat balance
q = stfncnstt*(T1^4-T2^4)/((1-e1)/e1*W+ 1/(W*F12+[(1/W/F1R) + (1/W/F2R)]^-1) + (1-e2)/e2*W) ;//[W/m]

//Surface Energy Balance 13.13
J1 = stfncnstt*T1^4 - (1-e1)*q/(e1*W)        ;// [W/m^2] Surface 1
J2 = stfncnstt*T2^4 - (1-e2)*(-q)/(e2*W)     ;// [W/m^2] Surface 2
//From Equation 13.26 Heat balance
JR = (J1+J2)/2;
TR = (JR/stfncnstt)^.25;

printf('\n Rate at which heat must be supplied per unit length of duct = %.2f kW/m \n Temperature of the insulated surface = %i K',q/1000,TR);
```

Scilab code Exa 13.7 Semi Circular Tube

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 13.7    Page 841 \n')// Example 13.7

// Rate at which heat must be supplied
// Temperature of the insulated surface
```

```

T1 = 1000       ;//[K] Temperature of Heated Surface
e1 = .8      ;// emissivity of Heated Surface
e2 = .8       ; // emissivity of Insulated Surface
r = .02         ;//[m] Radius of surface
Tm = 400         ;//[K] Temperature of surrounding air
m = .01         ;//[kg/s] Flow rate of surrounding air
p = 101325      ;//[Pa] Pressure of surrounding air
stfncnstt = 5.670*10^-8         ;//[W/m^2.K^4] Stefan Boltzman Constant
//Table A.4 Air Properties at 1 atm, 400 K
k = .0338         ;//[W/m.K] conductivity
u = 230*10^-7     ;//[kg/s.m] Viscosity
cp = 1014         ;//[J/kg] Specific heat
Pr = .69          ;// Prandtl Number

//Hydraulic Diameter
Dh = 2*%pi*r/(%pi+2)        ;//[m]
//Reynolds number
Re = m*Dh/(%pi*r^2/2)/u;
//View Factor
F12 = 1 ;

printf("\n As Reynolds Number is %i, Hence it is Turbulent flow inside a cylinder. Hence we will use Dittus-Boelter Equation",Re);

//From Dittus-Boelter Equation
Nu = .023*Re^.8*Pr^.4;
h = Nu*k/Dh;            //[W/m^2.K]

//From Equation 13.18 Heat Energy balance
//Newton Raphson
T2=600;         //Initial Assumption
while(1>0)
f=(stfncnstt*(T1^4 - T2^4)/((1-e1)/(e1*2*r)+1/(2*r*
```

```
   F12)+(1-e2)/(e2*%pi*r)) - h*%pi*r*(T2-Tm));
fd=(4*stfncnstt*( - T2^3)/((1-e1)/(e1*2*r)+1/(2*r*
   F12)+(1-e2)/(e2*%pi*r)) - h*%pi*r*(T2));
T2n=T2-f/fd;
if(stfncnstt*(T1^4 - T2n^4)/((1-e1)/(e1*2*r)+1/(2*r*
   F12)+(1-e2)/(e2*%pi*r)) - h*%pi*r*(T2n-Tm))<=.01
    break;
end;
T2=T2n;
end

//From energy Balance
q = h*%pi*r*(T2-Tm) + h*2*r*(T1-Tm)        ;// [W/m]

printf('\n Rate at which heat must be supplied per
   unit length of duct = %.2f W/m & Temperature of
   the insulated surface = %i K',q,T2);
```

Chapter 14

Diffusion Mass Transfer

Scilab code Exa 14.1 Molar and Mass Fluxes of Hydrogen

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   14.1    Page 884 \n')// Example 14.1

// Molar and mass fluxes of hydrogen and the
   relative values of the mass and thermal
   diffusivities for the three cases

T = 293       ;//[K] Temperature
Ma = 2        ;//[kg/kmol] Molecular Mass
//Table A.8 Hydrogen-Air Properties at 298 K
Dab1 = .41*10^-4;         //[m^2/s] diffusion
   coefficient
//Table A.8 Hydrogen-Water Properties at 298 K
Dab2 = .63*10^-8;         //[m^2/s] diffusion
   coefficient
//Table A.8 Hydrogen-iron Properties at 293 K
Dab3 = .26*10^-12;         //[m^2/s] diffusion
   coefficient
```

```
//Table A.4 Air properties at 293 K
a1 = 21.6*10^-6;         //[m^2/s] Thermal Diffusivity
//Table A.6 Water properties at 293 K
k = .603        ;//[W/m.K] conductivity
rho = 998       ;//[kg/m^3] Density
cp = 4182       ;//[J/kg] specific Heat
//Table A.1 Iron Properties at 300 K
a3 = 23.1 * 10^-6;     //[m^2/s]

//Equation 14.14
//Hydrogen-air Mixture
DabT1 = Dab1*(T/298)^1.5;      // [m^2/s] mass diffusivity
J1 = -DabT1*1;                //[kmol/s.m^2]  Total molar concentration
j1 = Ma*J1;                   //[kg/s.m^2] mass Flux of Hydrogen
Le1 = a1/DabT1;               // Lewis Number Equation 6.50

//Hydrogen-water Mixture
DabT2 = Dab2*(T/298)^1.5;      // [m^2/s] mass diffusivity
a2 = k/(rho*cp)               ;//[m^2/s] thermal diffusivity
J2 = -DabT2*1                ;//[kmol/s.m^2]  Total molar concentration
j2 = Ma*J2                   ;//[kg/s.m^2] mass Flux of Hydrogen
Le2 = a2/DabT2                ;// Lewis Number Equation 6.50

//Hydrogen-iron Mixture
DabT3 = Dab3*(T/298)^1.5;      // [m^2/s] mass diffusivity
J3 = -DabT3*1;                //[kmol/s.m^2]  Total molar concentration
```

```
j3 = Ma*J3;                    //[kg/s.m^2] mass Flux of
   Hydrogen
Le3 = a3/DabT3                   ;// Lewis Number Equation
    6.50

printf('\n Species     a (m^2/s)       Dab (m^2/s)
        Le          ja (kg/s.m^2) \n Air          %.1e
         %.1e          %.2f           %.1e \n Water
       %.1e          %.1e           %i             %.1e \n
    Iron         %.1e          %.1e          %.1e      %.1e
   ',a1,DabT1,Le1,j1,a2,DabT2,Le2,j2,a3,DabT3,Le3,j3
   );
```

Scilab code Exa 14.2 Evaporation Rate Through a Single Pore

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n
   Incropera / Dewitt / Bergman / Lavine \n EXAMPLE
   14.2    Page 898 \n')// Example 14.2

// Evaporation rate through a single pore

T = 298           ;//[K] Temperature
D = 10*10^-6      ;//[m]
L = 100*10^-6;           //[m]
H = .5             ;// Moist Air Humidity
p = 1.01325       ;//[bar]
//Table A.6 Saturated Water vapor Properties at 298
   K
psat = .03165;              //[bar] saturated Pressure
//Table A.8 Water vapor-air Properties at 298 K
Dab = .26*10^-4;          //[m^2/s] diffusion
   coefficient

```

```
C = p/(8.314*10^-2*298)        ;//Total Concentration
//From section 6.7.2, the mole fraction at x = 0 is
xa0 = psat/p;
//the mole fraction at x = L is
xaL = H*psat/p;

//Evaporation rate per pore Using Equation 14.41 with advection
N = (%pi*D^2)*C*Dab/(4*L)*2.303*log10((1-xaL)/(1-xa0))    ;//[kmol/s]

//Neglecting effects of molar averaged velocity Equation 14.32
//Species transfer rate per pore
Nh = (%pi*D^2)*C*Dab/(4*L)*(xa0-xaL)        ;//[kmol/s]

printf('\n Evaporation rate per pore Without advection effects %.2e kmol/s and With Advection effects %.2e kmol/s',Nh,N)

clf();
x = linspace(300,800,100);
y1 = N*x^1.5/298^1.5*10^15;
y2 = Nh*x^1.5/298^1.5*10^15;
plot(x,y1,x,y2);
xtitle("Evaporation Temp vs Temp", "T (K)", "Na*10^15(kmol/s)");
legend ("Without Advection", "With Advection");
```

Scilab code Exa 14.3 Polymer Sheet and Trough Geometry

```
clear;
clc;
```

```
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 14.3    Page 898 \n')// Example 14.3

// Rate of water vapor molar diffusive ttansfer through the trough wall

D = .005      ;//[m] Diameter
L = 50*10^-6;            //[m] Length
h = .003               ;//[m] Depth
Dab = 6*10^-14          ;//[m^2/s] Diffusion coefficient
Cas1 = 4.5*10^-3         ;//[kmol/m^3] Molar concentrations of water vapor at outer surface
Cas2 = 0.5*10^-3         ;//[kmol/m^3] Molar concentrations of water vapor at inner surface

//Transfer Rate through cylindrical wall Equation 14.54
Na = Dab/L*(%pi*D^2/4 + %pi*D*h)*(Cas1-Cas2);    //[kmol/s]

printf('\n Rate of water vapor molar diffusive ttansfer through the trough wall %.2e kmol/s',Na);
//END
```

Scilab code Exa 14.4 Helium Gas Spherical Container

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 14.4    Page 902 \n')// Example 14.4

```

```
// The rate of change of the helium pressure dp/dt

D = .2              ;//[m] Diameter
L = 2*10^-3         ;//[m] Thickness
p = 4               ;//[bars] Helium Pressure
T = 20+273          ;//[K] Temperature
//Table A.8 helium-fused silica (293K) Page 952
Dab = .4*10^-13         ;//[m^2/s] Diffusion coefficient
//Table A.10 helium-fused silica (293K)
S = .45*10^-3         ;//[kmol/m^3.bar] Solubility

// By applying the species conservation Equation 14.43 and 14.62
dpt = -6*(.08314)*T*(Dab)*S*p/(L*D);

printf('\n The rate of change of the helium pressure dp/dt %.2e bar/s',dpt);
//END
```

Scilab code Exa 14.5 Hydrogen Plastic Diffusion

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 14.5   Page 904 \n')// Example 14.5

// The Hydrogen mass diffusive flux nA (kg/s.m^2)
//A -> Hydrogen
//B -> Plastic

Dab = 8.7*10^-8         ;//[m^2/s] Diffusion coefficient
Sab = 1.5*10^-3         ;//[kmol/m^3.bar] Solubility
```

```
L = .0003              ;//[m] thickness of bar
p1 = 3                ;//[bar] pressure on one side
p2 = 1                ;//[bar] pressure on other side
Ma = 2                ;//[kg/mol] molecular mass of Hydrogen
//Surface molar concentrations of hydrogen from Equation 14.62
Ca1 = Sab*p1        ; //[kmol/m^3]
Ca2 = Sab*p2        ; //[kmol/m^3]
//From equation 14.42 to 14.53 for obtaining mass flux
N = Dab/L*(Ca1-Ca2) ;       //[kmol/s.m^2]
n = Ma*N            ;       //[kg/s.m^2] on Mass basis

printf('\n The Hydrogen mass diffusive flux n = %.2e (kg/s.m^2)',n);
//END
```

Scilab code Exa 14.6 Bacteria BioFilm

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 14.6    Page 909 \n')// Example 14.6

// Maximum Thickness of a bacteria laden biofilm, that may be siccessfully treated

Dab = 2*10^-12         ;//[m^2/s] Diffusion coefficient
Ca0 = 4*10^-3         ;//[kmol/m^3] Fixed Concentration of medication
```

```
Na = -.2*10^-3          ;//[kmol/m^3.s] Minimum consumption rate of antibiotic
k1 = .1                 ;//[s^-1] Reaction Coefficient

//For firsst order kinetic reaction Equation 14.74
m = (k1/Dab)^.5;
L = m^-1*acosh(-k1*Ca0/Na);

printf('\n Maximum Thickness of a bacteria laden biofilm, that may be siccessfully treated is %.1f pico-m',L*10^6);
//END
```

Scilab code Exa 14.7 Drug Medication

```
clear;
clc;
printf('FUNDAMENTALS OF HEAT AND MASS TRANSFER \n Incropera / Dewitt / Bergman / Lavine \n EXAMPLE 14.7    Page 913 \n')// Example 14.7

// Total dosage of medicine delivered to the patient over a one-week time period, sensivity of the dosage to the mass duffusivity of the patch and skin

Dap = .1*10^-12         ;//[m^2/s] Diffusion coefficient of medication with patch
Das = .2*10^-12         ;//[m^2/s] Diffusion coefficient of medication with skin
L = .05                 ;//[m] patch Length
rhop = 100              ;//[kg/m^3] Density of medication on patch
rho2 = 0                ;//[kg/m^3] Density of medication on skin
```

```
K = .5                    ;//Partition Coefficient
t = 3600*24*7             ;//[s] Treatment time

//Applying Conservation of species equation 14.47b
//By analogy to equation 5.62, 5.26 and 5.58
D = 2*rhop*L^2/(sqrt(%pi))*sqrt(Das*Dap*t)/(sqrt(Das
   )+sqrt(Dap)/K);

printf('\n Total dosage of medicine delivered to the
    patient over a one-week time period is %.1f mg',
   D*10^6);

//Senstivity of dosage to the patch and skin
clf();
//Subplot 1
Dap1 = .1*10^-12              ;//[m^2/s]
Das1 = .1*10^-12              ;//[m^2/s]
Das2 = .2*10^-12              ;//[m^2/s]
Das3 = .4*10^-12             ;//[m^2/s]
x = linspace(0,7,50);
y1 = 2*rhop*L^2/(sqrt(%pi))*sqrt(Das1*Dap1*3600*24*x
   )/(sqrt(Das1)+sqrt(Dap1)/K)*10^6;
y2 = 2*rhop*L^2/(sqrt(%pi))*sqrt(Das2*Dap1*3600*24*x
   )/(sqrt(Das2)+sqrt(Dap1)/K)*10^6;
y3 = 2*rhop*L^2/(sqrt(%pi))*sqrt(Das3*Dap1*3600*24*x
   )/(sqrt(Das3)+sqrt(Dap1)/K)*10^6;
subplot(1,2,1);
plot(x,y1,x,y2,x,y3);
xtitle("Dosage vs Time-period at Dap = .1*10^ -12 (m
   ^2/s)", "Day", "Dosage (mg)");
legend (".1*10^12", ".2*10^12", ".4*10^12");

//Subplot 2
Dap2 = .01*10^-12              ;//[m^2/s]
yn1 = 2*rhop*L^2/(sqrt(%pi))*sqrt(Das1*Dap2*3600*24*
   x)/(sqrt(Das1)+sqrt(Dap2)/K)*10^6;
yn2 = 2*rhop*L^2/(sqrt(%pi))*sqrt(Das2*Dap2*3600*24*
   x)/(sqrt(Das2)+sqrt(Dap2)/K)*10^6;
```

```
yn3 = 2*rhop*L^2/(sqrt(%pi))*sqrt(Das3*Dap2*3600*24*
    x)/(sqrt(Das3)+sqrt(Dap2)/K)*10^6;
subplot(1,2,2);
plot(x,yn1,x,yn2,x,yn3);
xtitle("Dosage vs Time-period at Dap = .01*10^ -12 (
    m^2/s)", "Day", "Dosage (mg)");
legend (".1*10^12", ".2*10^12", ".4*10^12");
//END
```