Scilab Textbook Companion for Introduction to Nuclear Engineering by J. R. Lamarsh and A. J. Baratta[1]

Created by
Dhaivat Udayan Mandavia
NUCLEAR SCIENCE AND ENGINEERING
Physics
DELHI TECHNOLOGICAL UNIVERSITY
College Teacher
None
Cross-Checked by
Spandana

June 8, 2016

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

Book Description

Title: Introduction to Nuclear Engineering

Author: J. R. Lamarsh and A. J. Baratta

Publisher: Prentice Hall, New Jersey

Edition: 3

Year: 2001

ISBN: 0-201-82498-1

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

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

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

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

Contents

List of Scilab Codes

List of Figures

Chapter 2

Atomic and Nuclear Physics

Scilab code Exa 2.1 Number of deuterium atoms

```
// Example 2.1
clear all;
clc;
// Given data
atom_h = 6.6*10^24;                         // Number of atoms in Hydrogen
// Using the data given in Table II.2, Appendix II for isotropic abundance of deuterium
isoab_H2 = 0.015;                           // Isotropic abundance of deuterium
// Calculation
totatom_d=(isoab_H2*atom_h)/100;
// Result
printf('\n Number of deuterium atoms = %2.1E \n', totatom_d);
```

Scilab code Exa 2.2 Atomic and Molecular weight

```
// Example 2.2
clear all;
clc;
// Given data
// Using the data given in the example 2.2
atwt_O16 = 15.99492;                    // Atomic weight of O-16 isotope
isoab_O16 = 99.759;                     // Abundance of O-16 isotope
atwt_O17 = 16.99913;                    // Atomic weight of O-17 isotope
isoab_O17 = 0.037;                      // Abundance of O-17 isotope
atwt_O18 = 17.99916;                    // Atomic weight of O-18 isotope
isoab_O18 = 0.204;                      // Abundance of O-18 isotope
// Calculation
atwt_O=(isoab_O16*atwt_O16 + isoab_O17*atwt_O17 + isoab_O18*atwt_O18)/100;
// Result
printf('\n Atomic Weight of Oxygen = %5.5f \n',atwt_O);
```

Scilab code Exa 2.3 Mass and Energy

```
// Example 2.3
clear all;
clc;
// Given data
me = 9.1095*10^(-28);                   // Mass of electron in grams
c = 2.9979*10^10;                       // Speed of light in vacuum in cm/sec
// Calculation
```

```
rest_mass = me*c^2;
// Result
printf('\n Rest mass energy of electron = %5.4E ergs\n',rest_mass);
disp('Expressing the result in joules')
// 1 Joule = 10^(-7)ergs
rest_mass_j = rest_mass*10^(-7);
printf('\n Rest mass energy of electron = %5.4E joules\n',rest_mass_j);
disp('Expressing the result in MeV')
// 1 MeV = 1.6022*10^(-13)joules
rest_mass_mev = rest_mass_j/(1.6022*10^(-13));
printf('\n Rest mass energy of electron = %5.4f MeV\n',rest_mass_mev);
```

Scilab code Exa 2.4 Mass and Energy

```
// Example 2.4
clear all;
clc;

// From the result of Example 2.3
// Rest mass energy of electron = 0.5110 MeV
rest_mass_mev = 0.5110;
me = 9.1095*10^(-28);                    // Mass of electron in grams
// From standard data table
// 1 amu = 1.6606*10^(-24)g
amu = 1.6606*10^(-24);
// Calculation
en_eq = (amu/me)*rest_mass_mev;
// Result
printf('\n Energy equivalent of one amu = %3.1f MeV\n',en_eq);
```

Scilab code Exa 2.5 Excited states and radiation

```
// Example 2.5
clear all;
clc;

// From the standard data table
h = 6.626*10^(-34);                                     // Planck's constant in J-s
c = 3*10^8;                                              // Speed of light in vacuum in m/sec
// Given data
disp('The ionization energy of K shell electron in Lead atom is 88keV');
E = 88*10^3;                                             // Ionization energy  in keV
// Expressing the result in joules by using 1 eV = 1.6022*10^(-19) J
E = E*1.6022*10^(-19);
printf("From Planck\''s law of photoelectric effect \n Energy = (h*c)/lambda\n");
// Calculation
lambda = (h*c)/E;
// Result
printf('\n Wavelength of radiation = %4.3E m\n', lambda);
```

Scilab code Exa 2.6 Activity of radioactive foil

```
// Example 2.6
clear all;
clc;
```

```

// Given data
T12 = 64.8;                                                     // Half life = 64.8 hour
lambda = 0.693/T12;                                             // Decay constant in hour^(-1)
t = 12;                                                         // Analysis time of gold sample in hours
alpha = 0.9;                                                    // Activity of gold sample after analysis time

// 1.
// Calculation
R = alpha/(1-exp(-lambda*t));
// Result
printf('\n Theoretical maximum activity = %3.1f curie (Ci) \n',R);

// 2.
// Calculation
// The expression to calculate 80 percent of maximum activity is \n 0.8R = R*(1-exp(-lambda*t))
t = -log(0.2)/lambda;
// Result
printf('\n Time to reach 80 percent of maximum activity = %d hours \n',t);
```

Scilab code Exa 2.7 Nuclear Reaction

```
// Example 2.7
clear all;
```

```
clc;

disp('The reactants are Nitrogen and neutron')
// The total atomic number of reactants
Z_reactant = 7+0;
// The total atomic mass number of reactants
A_reactant = 14+1;
disp('One of the known product is Hydrogen')
Z_H = 1;                                   // The atomic number of Hydrogen
A_H = 1;                                   // The atomic mass number of Hydrogen
// The atomic number of unknown element
Z_unknown = Z_reactant-Z_H;
// The atomic mass number of unknown element
A_unknown = A_reactant-A_H;
// Result
printf(" \n For unknown element the atomic number is %d and atomic mass number is %d \n",Z_unknown,A_unknown);
// From periodic table
disp('The element corresponds to Carbon-14');
```

Scilab code Exa 2.8 Q value of nuclear reaction

```
// Example 2.8
clear all;
clc;

disp('The reaction is Tritium(d,n)Helium-4');
// Using standard data table of mass in amu
M_H3 = 3.016049;                           // Atomic mass of Tritium
M_He4 = 4.002604;                          // Atomic mass of Helium
```

```
M_d = 2.014102;                                    // Atomic mass of Deuterium
M_n = 1.008665;                                    // Atomic mass of neutron
// Calculation of total mass of reactants
tot_reac = M_H3+M_d;
// Calculation of total mass of products
tot_prod = M_He4+M_n;
// Calculation
Q = tot_reac-tot_prod;
// Expressing in MeV by using 1 amu = 931.5 MeV
Q_mev = Q*931.5;
// Result
printf(" \n Q value for the reaction = %5.3f MeV", Q_mev);
if Q_mev > 0 then
    printf("\n The reaction is exothermic. \n");
else
    printf("\n The reaction is endothermic. \n");
end
```

Scilab code Exa 2.9 Binding energy

```
// Example 2.9
clear all;
clc;

// Using standard data table of mass in amu
M_C12 = 12;                                  // Atomic mass of Carbon-12
M_n = 1.00866;                               // Atomic mass of neutron
M_C13 = 13.00335;                            // Atomic mass of Carbon-13
//If one neutron is removed from carbon-13, carbon
```

```
-12 is obtained
tot = M_C12+M_n;
dm = tot-M_C13;                          // Mass defect
// Converting to energy equivalent from mass by using 1 amu = 931.5 MeV
Es = dm*931.5;
// Result
printf(" \n The binding Energy of the last neutron in Carbon-13 atom = %4.2f MeV",Es);
```

Scilab code Exa 2.10 Mass equation

```
// Example 2.10
clear all;
clc;

// Using standard data table of mass and the coefficients of mass equation for Silver-107
N = 60;                                  // Number of neutrons
Z = 47;                                  // Atomic number
A = 107;                                 // Atomic mass number
// The coefficients used in mass equation are
alpha = 15.56;
bet = 17.23;
gam = 0.697;
zeta = 23.285;
mn = 939.573;                            // Mass of neutron in terms of energy
mH = 938.791;                            // Mass of proton in terms of energy
// Calculation
```

```
disp('Using mass equation');
M = (N*mn)+(Z*mH)-(alpha*A)+(bet*(A^(2/3)))+(gam*Z^2/A^(1/3))+(zeta*(A-2*Z)^2/A);
// Expressing in amu by using 1 amu = 931.5 MeV
M_amu = M/931.5;
printf(" Mass = %5.5f MeV = %5.5f u \n",M,M_amu);
disp('Actual mass = 106.905092 u');
// Calculation
BE = (alpha*A)-(bet*(A^(2/3)))-(gam*Z^2/A^(1/3))-((zeta*(A-(2*Z))^2)/A);
// Result
printf("\n Binding Energy = %4.2f MeV or %3.1f MeV/nucleon \n",BE,BE/107);
// The value is different from the answer given in the textbook. The textbook answer is wrong.
```

Scilab code Exa 2.11 Energy of Maxwellian distribution

```
// Example 2.11
clear all;
clc;

// Given data
T_C = 38;                                    // Given temeperature in celsius
//The temperature in Kelvin
T_K = T_C+273.15;
T_0 = 293.61;                                // The temperature in kelvin equivalent to 0 deg celsius
kT = 0.0253;                                 // The term 'kT' in eV at temperature T0
// Calculation
Ep = 0.5*kT*(T_K/T_0);
Ebar = 3*Ep;
// Result
```

```
printf(" Most probable energy of air molecules = %5.5f eV \n",Ep);
printf(" Average energy of air molecules = %5.5f eV \n",Ebar);
```

Scilab code Exa 2.12 Atom density

```
// Example 2.12
clear all;
clc;

// Given data
rho = 0.97;                              // Density of Sodium in gram/cm^3
// From standard data table
NA = 0.6022*10^24;                       // Avagodro number
M = 22.99;                               // Atomic weight of Sodium
// Calculation
N = rho*NA/M;
// Result
printf("Atom density of sodium = %5.5E atoms/cm^3 \n",N);
```

Scilab code Exa 2.13 Atom density

```
// Example 2.13
clear all;
clc;

// Given data
```

```
rho_NaCl = 2.17;                              // Density of Sodium Chloride(NaCl) in gram/cm^3
// From standard data table
NA = 0.6022*10^24;                            // Avogodro number
M_Na = 22.99;                                 // Atomic weight of Sodium(Na)
M_Cl = 35.453;                                // Atomic weight of Chlorine(Cl)
M_NaCl = M_Na+M_Cl;                           // Molecular weight of Sodium Chloride(NaCl)
// Calculation
N = rho_NaCl*NA/M_NaCl;
// As in NaCl, there is one atom of Na and Cl
N_Na = N;
N_Cl = N;
// Result
printf(" Atom density of Sodium and Chlorine = %5.4E molecules/cm^3 \n",N);
```

Scilab code Exa 2.14 Atom density

```
// Example 2.14
clear all;
clc;

// Given data
rho = 1;                                                  // Density of water in gram/cm^3

// 1.
M_H = 1.00797;                                            // Atomic weight of Hydrogen(H)
M_O = 15.9994;                                            // Atomic weight of Oxygen(O)
```

```
// As in water, there is two atoms of Hydrogen(H) and one atom of Oxygen(O)
M = (2*M_H)+M_O;                                 // Molecular weight of water
// From standard data table
NA = 0.6022*10^24;                               // Avagodro number
// Calculation
N = rho*NA/M;
// Result
printf("Atom density of water = %5.5E molecules/cm^3 \n",N);

// 2.
// As in water, there is two atoms of Hydrogen(H) and one atom of Oxygen(O)
N_H = 2*N;                                       // Atom density of Hydrogen
N_O = N;                                         // Atom density of Oxygen
// Result
printf("Atom density of Hydrogen(H) = %5.4E atoms/cm^3 \n",N_H);
printf("Atom density of Oxygen(O)= %5.4E atoms/cm^3 \n",N_O);

// 3.
// Using the data given in Table II.2, Appendix II for isotropic abundance of deuterium
isoab_H2 = 0.015;
// Calculation
N_H2 = isoab_H2*N_H/100;
// Result
printf("Atom density of Deuterium(H-2)= %5.4E atoms/cm^3 \n",N_H2);
```

Scilab code Exa 2.15 Atom density

```
// Example 2.15
clear all;
clc;

// Given data
rho = 19.1;                                     // Density of Uranium-235 in gram/cm^3
wt = 1500;                                      // Weight of uranium rods in a reactor in kg
nr = 0.2;                                       // Enrichment(w/o) of Uranium-235

// 1.
// As Enrichment is 20(w/o)
wt_U235 = nr*wt;                                // Amount of Uranium-235
// Result
printf("Amount of Uranium-235 in the reactor = %d kg \n",wt_U235);

// 2.
// From standard data table
NA = 0.6022*10^24;                              // Avagodro number
M_U235 = 235.0439;                              // Atomic weight of Uranium-235
M_U238 = 238.0508;                              // Atomic weight of Uranium-238
// Calculation
N_U235 = nr*rho*NA/M_U235;                      // Atom density of Uranium-235
N_U238 = (1-nr)*rho*NA/M_U238;                  // Atom
```

```
density of Uranium-238
// Result
printf("Atom density of Uranium-235 = %5.2E atoms/cm^3 \n",N_U235);
printf("Atom density of Uranium-238 = %5.2E atoms/cm^3 \n",N_U238);
```

Scilab code Exa 2.16 Atom density

```
// Example 2.16
clear all;
clc;

// Given data
rho_UO2 = 10.5;                    // Density of UO2 pellets in gram/cm^3
nr = 0.3;                          // Enrichment(w/o) of Uranium-235
// From standard data table
M_U235 = 235.0439;                 // Atomic weight of Uranium-235
M_U238 = 238.0508;                 // Atomic weight of Uranium-238
M_O = 15.999;                      // Atomic weight of Oxygen
NA = 0.6022*10^24;                 // Avogodro number

M = 1/((nr/M_U235)+((1-nr)/M_U238));
M_UO2 = M+(2*M_O);                 // Molecular weight of UO2
nr_U = M/M_UO2*100;                // The percent(w/o) of Uranium in UO2 pellet
rho_U = nr_U*rho_UO2/100           // Density of Uranium in g/cm^3
```

```
rho_U235 = nr*rho_U                               //
   Density of Uranium-235 in g/cm^3
// Calculation
N_U235=rho_U235*NA/M_U235;
// Result
printf("Atom density of Uranium-235 = %5.2E atoms/cm
   ^3 \n",N_U235);
```

Chapter 3

Interaction of Radiation with Matter

Scilab code Exa 3.1 Neutron collsion

```
// Example 3.1
clear all;
clc;

// Given data
// 1 barn = 10^(-24) cm^2
sigma = 2.6*10^(-24);                       // Cross section of carbon-12 in cm^2
I = 5*10^8;                                 // Intensity of neutron beam in neutrons/cm^2-sec
A = 0.1;                                    // Cross sectional area of the beam in cm^2;
X = 0.05;                                   // Thickness of the target in cm

// 1.
// Using the data given in Table I.3, Appendix II for carbon-12
N = 0.08*10^(24);                           // Atom
```

```
density in atoms/cm^3
// Calculation
IR = sigma*I*N*A*X;
// Result
printf('\n Total interaction rate = %2.1E interactions/sec \n',IR);

// 2.
no = I*A;                                    // Neutron rate in neutrons/sec
// Calculation
p = IR/no;
printf('\n Probability of collision = %3.2E \n',p);
```

Scilab code Exa 3.2 Probability of nuclear reaction

```
// Example 3.2
clear all;
clc;

// Given data
sigmaf = 582;             // Fission cross section of U-235 on bombardment of neutron in barn
sigmay = 99;              // Radiative capture cross section of U-235 on bombardment of neutron in barn
// Calculation
pf = sigmaf/(sigmaf+sigmay);
// Result
printf('\n Probability of fission = %.3f = %3.1f percent\n',pf,pf*100);
```

Scilab code Exa 3.3 Neutron collsion

```
// Example 3.3
clear all;
clc;

// Given data
// Using the data given in the example 3.1
N = 0.08*10^(24);                    // Atom density of Carbon-12 in atoms/cm^3
// 1 barn = 10^(-24) cm^2
sigma = 2.6*10^(-24);                // Cross section of carbon-12 in cm^2
I = 5*10^8;                          // Intensity of neutron beam in neutrons/cm^2-sec

// 1.
// Calculation
SIGMAt = N*sigma;
// Result
printf('\n Macroscopic cross section of carbon-12 = %3.2f cm^(-1)\n',SIGMAt);

//2.
// Calculation
 F= I*SIGMAt;
// Result
printf('\n Collision density in the carbon-12 target = %3.2E collisions/cm^(3)-sec\n',F);
```

Scilab code Exa 3.4 Mean free path

```
// Example 3.4
clear all;
clc;

// Given data
```

```
E = 100;                                // Neutron energy in keV
// Using the data given in Table II.3, for E = 100 keV
atom_density = 0.0254*10^(24);          // Atom density of sodium in atoms/cm^3
// 1 barn = 10^(-24) cm^2
sigma = 3.4*10^(-24);                   // Microscopic cross section of sodium in cm^2
// Calculation
SIGMA = atom_density*sigma;
lambda = 1/SIGMA;
// Result
printf('\n Macroscopic cross section = %5.4f cm^(-1)\n',SIGMA);
printf('\n Mean Free Path = %3.2f cm\n',lambda);
```

Scilab code Exa 3.5 Absorption cross section

```
// Example 3.5
clear all;
clc;

// Given data
atom_density_U235 = 3.48*10^(-4)*10^(24);   // Atom density of Uranium-235 in atoms/cm^3
atom_density_U238 = 0.0483*10^(24);         // Atom density of Uranium-238 in atoms/cm^3
// 1 barn = 10^(-24) cm^2
sigmaa_U235 = 680.8*10^(-24);               // Absorption cross section of Uranium-235 incm^2
sigmaa_U238 = 2.7*10^(-24);                 // Absorption cross section of Uranium-238 incm^2
// Calculation
SIGMAA=(atom_density_U235*sigmaa_U235)+(
```

```
atom_density_U238*sigmaa_U238);
// Result
printf('\n Macroscopic absorption cross section = %4.3f cm^(-1)\n',SIGMAA);
```

Scilab code Exa 3.6 Scattering cross section

```
// Example 3.6
clear all;
clc;

// Given data
sigmas_H_1 = 3;                    // Scattering cross section of Hydrogen in barn at 1 MeV
sigmas_O_1 = 8;                    // Scattering cross section of Oxygen in barn at 1 MeV
sigmas_H_th = 21;                  // Scattering cross section of Hydrogen in barn at 0.0253 eV
sigmas_O_th = 4;                   // Scattering cross section of Oxygen in barn at 0.0253 eV
// Calculation
sigmas_H2O_1 = (2*sigmas_H_1)+(1*sigmas_O_1);
// Result
printf('\n Scattering cross section of Water at 1 MeV = %d b \n',sigmas_H2O_1);
// The equation used to calculate the scattering cross section at 1 MeV cannot be used at thermal energy.
printf(' Experimental value of scattering cross section of Water at 0.0253 eV = %d b \n',103);
```

Scilab code Exa 3.7 Reactor power

```
// Example 3.7
clear all;
clc;

// Given data
phi = 1*10^(13);                              // Neutron flux in neutrons/cm^3
v = 64000;                                    // Volume of research reactor in cm^3
sigmaf = 0.1;                                 // Macroscopic fission cross section in cm^(-1)
// The energy released per fission reaction is 200 MeV
// 1 MeV = 1.6*10^(-13) joule
E = 200*1.6*10^(-13);
// Calculation
fiss_rate = sigmaf*phi;                       // Fission rate in neutrons/cm^2-sec
power_cc = E*fiss_rate/10^6;                  // Reactor power/cc
power = power_cc*v;
printf('\n Reactor power of a research reactor = %d MW\n',power);
```

Scilab code Exa 3.8 Elastic scattering

```
// Example 3.8
clear all;
clc;

// 1 barn = 10^(-24) cm^2
// From the Figure 3.4 given in the textbook
sigmae = 4.8*10^(-24);                  // Experimental cross section of carbon from 0.02eV to 0.01MeV
```

```
// Assuming spherical shape and elstic scattering
R = sqrt(sigmae/(4*%pi));
// Result
printf('\n Radius of carbon nucleus = %3.1E cm\n',R);
```

Scilab code Exa 3.9 Radiative capture reaction

```
// Example 3.9
clear all;
clc;

// Given data
E0 = 0.0253;                                   // Thermal energy in eV
// 1 barn = 10^(-24) cm^2
sigmay_E0 = 0.332*10^(-24);                    // Radiative capture cross section at 0.0253 eV in cm^2
E = 1;                                         // Energy in eV at which radiative cross section is to be found
// Calculation
sigmay_E = sigmay_E0*sqrt(E0/E);
// Result
// Expressing the result in barn
printf('\n Radiative capture cross section of hydrogen at 1 eV = %5.4f b\n',sigmay_E*10^(24));
```

Scilab code Exa 3.10 Energy loss in scattering reactions

```
// Example 3.10
clear all;
```

```
clc;

// Given data
E = 1;                                          // Energy of neutron in MeV
A = 2;                                          // Atomic mass number of deuterium
v = 45;                                         // Scattering angle in degree

//  1.
// Calculation
E_dash = E/(A+1)^2 *((cosd (v)+sqrt(A^2-(sind(v))^2))^2);
// Result
printf('\n Energy of scattered neutron = %4.3f MeV \n',E_dash);

// 2.
// Calculation
E_A = E-E_dash;
// Result
printf('\n Energy of recoil nucleus = %4.3f MeV \n',E_A);

// 3.
// Calculation
deltau = log(E/E_dash);
// Result
printf('\n Change in lethargy of neutron on collision = %4.3f \n',deltau);
```

Scilab code Exa 3.11 Neutron absorption rate

```
// Example 3.11
```

```
clear all;
clc;

// Given data
phi = 5*10^(12);                                  // Neutron flux in neutrons/cm^2-sec
T = 600;                                          // Temperature of neutron in degree
// Using the data given in Table II.3, Appendix II for indium
N = 0.0383*10^(24);                               // Atom density in atoms/cm^3
// 1 barn = 10^(-24) cm^2
sigmaa_E0 = 194*10^(-24);                         // Microscopic absorption cross section in cm^2
SIGMA_E0 = N*sigmaa_E0;                           // Macroscopic absorption cross section in cm^(-1)
// From Table 3.2
ga_600 = 1.15;                                    // Non 1/v factor at 600 degree celsius
// Calculation
F_a = ga_600*SIGMA_E0*phi;
// Result
printf('\n Absorption rate of neutrons per cc in indium foil = %4.2E neutrons/cm^3-sec \n',F_a);
```

Scilab code Exa 3.12 Activity estimation

```
// Example 3.12
clear all;
clc;

// Given data
N = 120;                                  // Number of fuel rods
```

```
P = 100;                                    // Reactor power in MW
t = 1;                                      // Estimation time of fuel rod after removal in days
T = 365;                                    // Time of reactor operation
// Estimation
Activity_total = 1.4*10^6*P*[t^(-0.2)-(t+T)^(-0.2)];
Activity_one = Activity_total/N;            // For one fuel rod
// Result
printf('\n The activity of a fuel rod = %2.1E Ci \n',Activity_one);
```

Scilab code Exa 3.13 Fission neutron estimation

```
// Example 3.13
clear all;
clc;

// Using the data given in Table 3.4 and Table II.2 for uranium
v_235 = 2.418;              // Average number of neutrons released per fission
y_235 = 0.72;               // Isotropic abundance of Uranium-235 on the earth
sigmaf_235 = 582.2;         // Fission cross section of Uranium-235
sigmaa_235 = 680.8;         // Absorption cross section of Uranium-235
N_235 = y_235;
y_238 = 99.26;              // Isotropic abundance of Uranium-238 on the earth
sigmaa_238 = 2.7;           // Absorption cross section of Uranium-238
```

```
// Calculation
n = (v_235*y_235*sigmaf_235)/((y_235*sigmaa_235)+(
    y_238*sigmaa_238));
// Result
printf('\n Eta for natural uranium = %3.2f \n',n);
```

Scilab code Exa 3.14 Energy released in fission reaction

```
// Example 3.14
clear all;
clc;

// Fission of 1 g of Uranium-235 releases
    approximately 1 MW/day of energy.
// 1 MW/day = 8.64*10^(10) J
energy_uranium = 8.64*10^10;

// 1. Coal
h_coal = 3*10^7;        // Heat contenet of coal in J/
    kg
// Calculation
amt_coal = energy_uranium/h_coal;
// Result
printf('\n Amount of coal required for energy
    equivalent of fission = %3.2E kg \n or %3.2f
    metric tons or %3.2f short tons\n',amt_coal,
    amt_coal/10^3,amt_coal*1.10231/10^3);
// The result is expressed in all units of
    commercial importance.

// 2. Oil
h_oil = 4.3*10^7;        // Heat contenet of oil in J/
    kg
// Calculation
amt_oil = energy_uranium/h_oil;
```

```
// Result
printf('\n Amount of oil required for energy equivalent of fission = %3.2E kg \n or %3.2f tons or %3.1f barrels\n',amt_oil,amt_oil/10^3,amt_oil*6.3/10^3);
// The result is expressed in all units of commercial importance.
```

Scilab code Exa 3.15 Gamma ray attenuation in materials

```
// Example 3.15
clear all;
clc;

// Given data
rho = 10;                                  // Density of UO2 in g/cm^3
mol_wt_UO2 = 238+(16*2);                   // Molecular weight of UO2
per_U   =  (238/mol_wt_UO2)*100;           // Percent by weight of Uranium
per_O = 100-per_U;                         // Percent by weight of Oxygen

// Calculation
//Using the data given in Table II.4 for uranium and oxygen
mup_U = 0.0757;                            // Ratio of mass attenuation coefficient to density of uranium in cm^2/g
mup_O = 0.0636;                            // Ratio of mass attenuation coefficient to density of oxygen in cm^2/g
mup = (per_U/100*mup_U)+(per_O/100*mup_O);     // The total ratio of mass attenuation coefficient in
```

```
cm^2/g
mu = mup*rho;
// Calculation
lambda = 1/mu;
// Result
printf('\n Mass attenuation coefficient of Uranium dioxide (UO2) = %5.3f cm^(-1) \n',mu);
printf('\n Mean free path = %3.2f cm \n',lambda);
// The answer is marked wrongly in the textbook. But the solution is correctly evaluated.
```

Scilab code Exa 3.16 Energy deposition of radioactive samples

```
// Example 3.16
clear all;
clc;

// Given data
E = 0.8;                        // Average gamma ray energy in MeV
I = 3*10^(11);                  // Intensity of gamma rays incident on the container in gamma rays/cm^2-sec
// Using the data given in Table II.5 for iron at 0.8 MeV
mup_iron = 0.0274;              // Ratio of mass attenuation coefficient to density of iron in cm^2/g
// Calculation
dep_rate = E*I*mup_iron;
// Expressing the result in SI units
// 1 MeV = 1.6*10^(-13) J
// 1 kg = 1000 g
dep_rate_SI = dep_rate*(1.6*10^(-13)*1000);
printf('\n Rate of energy deposited = %3.2E MeV/g-
```

```
sec or %.2f J/kg-sec \n',dep_rate,dep_rate_SI);
```

Scilab code Exa 3.17 Range of charged particles

```
// Example 3.17
clear all;
clc;

// Given data
E_max = 1.39;                        // Maximum energy of beta rays in MeV
// Calculation
R_max = 0.412*E_max^(1.265-(0.0954*log(E_max)));
// Result
printf('\n Maximum distance of beta rays traversed = %4.3f cm \n',R_max);
```

Chapter 4

Nuclear Reactors and Nuclear Power

Scilab code Exa 4.1 Conversion and Breeding of nuclear fuels

```
// Example 4.1
clear all;
clc;

// Given data
// Number of neutrons absorbed by Uranium-238 in
   resonances for every neutron absorbed in Uranium
   -235
n_resonance = 0.254;
// Number of neutrons absorbed by Uranium-238 at
   thermal energy for every neutron absorbed in
   Uranium-235
n_th = 0.64;
m = 1;                    // Amount of Uranium-235
   consumed in kg
A_U = 235;                // Atomic mass number of
   Uranium-235
A_Pu = 239;               // Atomic mass number of
   Plutonium-239
```

```
// 1.
// Calculation
C = n_resonance+n_th;
// Result
printf('\n Conversion ratio of the reactor = %4.3f \
   n',C);

// 2.
// Calculation
amt_Pu = m*C*A_Pu/A_U;
// Result
printf('\n Amount of Plutonium-239 produced in the
   reactor = %4.3f kg \n',amt_Pu);
```

Scilab code Exa 4.2 Production and doubling time of nuclear fuels

```
// Example 4.2
clear all;
clc;

// Given data
wP0 = 1;                  // Total fuel consumption
   rate in terms of kg/day
M = 500;                  // Amount of Plutonium-239
   in kg at startup of the reactor
breeding_gain = 0.15;     // Breeding gain of the
   reactor

// 1.
printf(" The Fast breeder reactor produces %.2f kg
   of plutonium-239 more for every kilogram consumed
    \n",breeding_gain);
// Calculation
// 1 year = 365 days
```

```
production_rate = ceil(breeding_gain*365);
// Result
printf("\n Production rate of plutonium-239 = %3.2f kg/day = %d kg/year",breeding_gain,production_rate);

// 2.
// Calculation
t_Dl = M/production_rate;
t_De = log(2)*t_Dl;
// Result
printf(" \n Linear doubling time of plutonium fuel in the reactor = %2.1f years \n",t_Dl);
printf(" \n Exponential doubling time of plutonium fuel in the reactor = %2.1f years \n",t_De);
```

Scilab code Exa 4.3 Nuclear fuel performance

```
// Example 4.3
clear all;
clc;

// Given data
power = 3300;           // Reactor power in MW
time = 750;             // Reactor operation time in days
amt_UO2 = 98;           // Amount of Uranium dioxide (UO2) in metric tons
atwt_U = 238;           // As the enrichment of Uranium-235 is 3 w/o the majority portion is Uranium-238
molwt_O = 16;            // Molecular weight of Oxygen


// 1.
```

```
amt_U = amt_UO2*atwt_U/(atwt_U+2*molwt_O);   // Amount of uranium in tonne
total_burnup = power*time;                    // Total burnup in MWd
// Calculation
specific_burnup = total_burnup/amt_U;
// Result
printf(" \n Specific burnup = %3.2f MWd/tonne \n", specific_burnup);

// 2.
// Due to fission of 1.05 g of Uranium-235, 1 MWd of energy is released.
m = 1.05;
P = 10^6;
maxth_burnup = P/m;                           // Theoritical maximum burnup
// Calculation of Fractional burnup
bet = specific_burnup/maxth_burnup;
// Result
printf(" \n Fractional burnup = %3.2f percent \n", bet*100);
// Due to approximation of specific burnup value, there is a slight change in fractional burnup value as compared to the textbook value.
```

Scilab code Exa 4.4 Plant availability factor and capacity factor

```
// Example 4.4
clear all;
clc;

// Given data
ratpower = 1075;                   // Output rated electrical power in MWe of the reactor
```

```
delpower_yr = 255000;            // Net output power delivered in one year in terms of MWd
time_refuel = 28;                // Number of days the plant was shutdown for refuelling
time_repairs = 45;               // Number of days the plant was shutdown for repairs
time_convrepairs = 18;           // Number of days the plant was shutdown for conventional repairs

// 1.
// 1 year = 365 days
ratpower_yr = ratpower*365;      // Net output rated power in one year in terms of MWd
// Calculation
cap_factor = delpower_yr/ratpower_yr;
// Result
printf("\n Plant capacity factor = %d percent\n",ceil(cap_factor*100));

// 2.
// Number of days the plant was shutdown in one year
total_shutdown = time_refuel+time_repairs+time_convrepairs;
// Number of days the plant was operable in one year
total_operation = 365-total_shutdown;
// Calculation
ava_factor = total_operation/365;
// Result
printf("\n Plant availability factor = %d percent\n",ava_factor*100);
```

Scilab code Exa 4.5 Nuclear fuel utlization

```
// Example 4.5
clear all;
```

```
clc;

// Given data
t = 30;                              // Time of uranium sufficiency in years
// Assuming once through Light Water Reactor (LWR) fuel cycle
U_LWR = 0.0055;                      // Uranium Utilization factor for LWR
// Assuming once through Liquid Metal cooled Fast Breeder Reactor (LMFBR) fuel cycle
U_LMFBR = 0.67;                      // Uranium Utilization factor for LMFBR
// Estimation
est_time = 30*U_LMFBR/U_LWR;
// Result
printf("The time for which Uranium would fuel LMFBR = %d years \n",ceil(est_time));
```

Scilab code Exa 4.6 Separative work

```
// Example 4.6
clear all;
clc;

// Given data
A_U = 238;                  // Atomic Mass number of Uranium
A_O = 16;                   // Atomic Mass number of Oxygen
amt_UO2 = 33000;            // Amount of Uranium dioxide (UO2) present in kilogram(kg)
x_P = 0.032;                // Enrichment of 3.2 w/o uranium product
x_T = 0.002;                // Enrichemnt of 0.2 w/o
```

```
residual tails
// From Figure 4.45
x_F = 0.00711;           // Enrichemnt of 0.711 w/o feed

// 1.
// Estimation of enriched uranium in kg
M_P = A_U*amt_UO2/(A_U+2*A_O);
// Estimation of amount of Uranium feed in kg
M_F = ((x_P-x_T)/(x_F-x_T))*M_P;
// Result
printf(" \n The amount of uranium feed required per reload = %d kg \n",ceil(M_F));

// 2.
V_x_P = (1-2*x_P)*log((1-x_P)/x_P);          // Value function of uranium product with enrichemnt of 3.2 w/o
V_x_F = (1-2*x_F)*log((1-x_F)/x_F);          // Value function of feed with enrichemnt of 0.711 w/o
V_x_T = (1-2*x_T)*log((1-x_T)/x_T);          // Value function of tallings with enrichemnt of 0.2 w/o
rate_SWU = 130.75;                           // Enrichment cost in dollars per SWU
// Calculation
SWU = M_P*(V_x_P-V_x_T)-M_F*(V_x_F-V_x_T);   // Separative Work (SWU) in kg
enrich_cost = ceil(SWU)*rate_SWU;            // Enrichment cost in dollars
// Result
printf("\n The enrichment cost = $ %d \n",ceil(enrich_cost));
// Due to approximation of Separative Work Unit(SWU), there is a difference in the value of enrichment cost on comparison with the textbook value.
```

Chapter 5

Neutron Diffusion and Moderation

Scilab code Exa 5.2 Neutron diffusion

```
// Example 5.2
clear all;
clc;

// Given data
// 1 barn = 10^(-24) cm^2
sigma_s = 4.8*10^(-24)           // Scattering cross section of carbon in cm^2
A_C = 12;                        // Atomic Mass number for carbon-12
E = 1;                           // Energy of carbon-12 atom in eV
// Using the data given in Table II.3, for carbon (graphite) at energy 1 eV
N = 0.08023*10^(24);             // Atom density in terms of atom/cm^3
mu_bar = 2/(3*A_C);              // Average value of the cosine of the angle at which neutrons are scattered in the med/ium
```

```
SIGMA_s = N*sigma_s                // Macroscopic scattering cross section of carbon-12
// Calculation
D = 1/(3*SIGMA_s*(1-mu_bar));
// Result
printf('\n Diffusion coefficeint of graphite at 1 eV = %4.3f cm \n',D);
```

Scilab code Exa 5.5 Multigroup diffusion theory

```
// Example 5.5
clear all;
clc;

// Given data
phi1 = 6*10^(14);                  // Neutron flux of Group 1
phi2 = 1*10^(15);                  // Neutron flux of Group 2
phi3 = 3*10^(15);                  // Neutron flux of Group 3

// 1.
// Using the data given in Table II.3, for atom density of sodium
N = 0.02541*10^(24);               // Atom density in terms of atom/cm^3
// Using the data given for sigmay (Microscopic radiative capture cross section) in Table II.3,
// 1 barn = 10^(-24) cm^2
sigmay1 = 0.0005*10^(-24);         // Microscopic gamma cross section of Group 1
sigmay2 = 0.001*10^(-24);          // Microscopic gamma cross section of Group 2
sigmay3 = 0.001*10^(-24);          // Microscopic gamma
```

```
      cross section of Group 3
// Calculation
F_a = N*((sigmay1*phi1)+(sigmay2*phi2)+(sigmay3*phi3));
// Result
printf('\n Total absorption rate for three groups = %3.2E neutrons/cm^3-sec \n',F_a);

// 2.
// Calculation
sigmag_12 = 0.24*10^(-24);          // Microscopic scattereing cross section of neutrons from Group 1 to Group 2
F_12 = N*sigmag_12*phi1;
// Result
printf('\n Neutron scattering rate from the first to second group = %3.2E neutrons/cm^3-sec \n',F_12);
```

Scilab code Exa 5.6 Neutron diffusion

```
// Example 5.6
clear all;
clc;

// Given data
S = 10^7;                  // Strength of neutron source in neutrons/sec
r = 15;                    // Distance over which neutron flux is to be calculated in cm
// Using the data given in Table 5.2,
L_T = 2.85;                // Thermal diffusion length in cm
D_bar = 0.16;              // Diffusion coefficient in cm
// Calculation
```

```
phi_T = S*exp(-r/L_T)/(4*%pi*D_bar*r);
// Result
printf('\n Neutron flux = %3.2E neutrons/cm^2-sec \n',phi_T);
```

Scilab code Exa 5.7 Neutron diffusion

```
// Example 5.7
clear all;
clc;

// Given data
T_F = 500;                  // Temeperature in Fahrenheit
P = 2000;                   // Pressure in psi
rho = 49.6;                 // Density in terms of lb/ft^3
// Converting the given temperature from Fahrenheit to Celsius
T_C = (5/9)*(T_F-32);
// Converting the temperature from Celsius to Kelvin scale
T_K = 273+T_C;

// Using the data given in Table 5.2,
D_bar_0 = 0.16;             // Diffusion coefficient at 293 K
rho_0 = 62.4;               // Density at 293 K in terms of lb/ft^3
L_T2_0 = 8.1;               // Diffusion area at 293 K in cm^2
T_0 = 293;                  // Standard Temperature in kelvin
m = 0.47;                   // Material specific constant
```

```
// Calculation
D_bar = D_bar_0*(rho_0/rho)*(T_K/T_0)^m;
L_T2 = L_T2_0*(rho_0/rho)^2*(T_K/T_0)^(m+1/2);
// Result
printf('\n Diffusion coefficient of ordinary water = %4.3f cm \n',D_bar);
printf('\n Diffusion area of ordinary water = %3.1f cm^2 \n',L_T2);
```

Chapter 6

Neutron Reactor Theory

Scilab code Exa 6.1 Critical neutron parameters

```
// Example 6.1
clear all;
clc;

// Given data
M_F = 235;                    // Atomic mass of Uranium-235
M_S = 23;                     // Atomic mass of Sodium-23
rho_F_S = 1;                  // Ratio of densities of  Uranium fuel to Sodium
// Using the data given in Table 5.2,
sigmaa_S=0.0008;              // Absorption cross section of Sodium
sigmaa_F=1.65;                // Absorption cross section of Uranium

rho_S_F = 100-rho_F_S;
N_S_F = rho_S_F*(M_F/M_S);      // Ratio of atomic densities of Uranium and Sodium
// Using the data in Table 6.1 for Uranium-235
```

```
// The value of average number of neutrons produced for a neutron absorbed n(eta) for Uranium-235 is 2.2
eta = 2.2;

// Calculation
f = 1/(1+(N_S_F*(sigmaa_S/sigmaa_F)));
k_inf = eta*f;
// Result
printf('\n Thermal Utilization factor = %.3f \n',f);
printf('\n Infinite Multiplication factor = %3.2f \n',k_inf);
```

Scilab code Exa 6.2 Maximum and average flux

```
// Example 6.2
clear all;
clc;

// Given data
R = 50;                     // Radius of reactor core in cm
P = 100*10^6;               // Power level of the reactor in watt
SIGMA_f = 0.0047;           // Macroscopic fission cross section in cm^(-1)
E_R = 3.2*10^(-11);         // Energy released per fission in joules/second
// Using the data in Table 6.2 for spherical geometry
OMEGA = 3.29;               // Measure of the variation of flux in the reactor
// Calculation
phi_max = (%pi*P)/(4*E_R*SIGMA_f*R^3);
phi_av = phi_max/OMEGA;
```

```
// Result
printf('\n Maximum flux in the spherical reactor = %3.2E neutrons/cm^2-sec \n',phi_max);
printf('\n Average flux in the spherical reactor = %3.2E neutrons/cm^2-sec \n',phi_av);
```

Scilab code Exa 6.3 Critical radius

```
// Example 6.3
clear all;
clc;

// Given data
N_F = 0.00395*10^(24);                    // Atom density of Plutonium-239 fuel in atom/cm^3
N_S = 0.0234*10^(24);                     // Atom density of Sodium-23 in atom/cm^3
// Using the data given in Table 6.1,
// 1 barn = 10^(-24) cm^2
sigmaa_S = 0.0008*10^(-24);               // Microscopic absorption cross section of Sodium in cm^2
sigmaa_F = 2.11*10^(-24);                 // Microscopic absorption cross section of Plutonium in cm^2
sigmatr_F = 6.8*10^(-24);                 // Microscopic transport cross section of Plutonium
sigmatr_S = 3.3*10^(-24);                 // Microscopic transport cross section of Sodium
// The value of average number of neutrons produced for a neutron absorbed n(eta) for Plutonium-239 is 2.61
eta = 2.61;

SIGMAA_S = sigmaa_S*N_S;                 // Macroscopic absorption cross section of Sodium in cm^(-1)
SIGMAA_F = sigmaa_F*N_F;                // Macroscopic
```

```
absorption cross section of Plutonium in cm^(-1)
SIGMAA = SIGMAA_S+SIGMAA_F;           // Total macroscopic absorption cross section in cm^(-1)
SIGMA_tr = (sigmatr_F*N_F)+(sigmatr_S*N_S); // Macroscopic transport cross section
f = SIGMAA_F/SIGMAA;                  // Calculation of Thermal Utilization factor(f)
f = ceil(f);
k_inf = eta*f;                         // Calculation of Infinite Multiplication factor(k_inf)

D = 1/(3*SIGMA_tr);                   // Calculation of Diffusion coefficient
L2 = D/SIGMAA;                        // Diffusion area
d = 2.13*D;                           // Extrapolated distance
R_ctil = %pi*sqrt(L2/(k_inf-1));      // Critical Radius for an extrapolated boundary
// Calculation
R_c = R_ctil-d;
// Result
printf('\n Critical Radius = %2.1f cm \n',R_c);
// The answer given in the textbook is wrong.
```

Scilab code Exa 6.4 Non leakage probability

```
// Example 6.4
clear all;
clc;

// Using the result from Example 6.3
R_c = 48.5;                        // Critical Radius for an extrapolated boundary in cm
L2 = 384;                          // Diffusion area in
```

```
   cm^2
// Calculation
P_L = 1/(1+((%pi/R_c)^2*L2));
// Result
printf('\n Nonleakage probability of a fission neutron = %3.2f \n',P_L);
```

Scilab code Exa 6.5 Critical parameter calculations

```
// Example 6.5
clear all;
clc;

// Given data
R = 100;                          // Radius of a spherical reactor in cm
P = 10^5;                         // Power of the reactor in watt

// 1.
// Calculation
B = sqrt((%pi/R)^2);
// Result
printf(" \n Buckling = %3.2E \n",B);

// 2.
// Using the data from Tables 3.2, 5.2, 5.3 and 6.3
L_TM2 = 3500;                     // Diffusion area of moderator (Sodium) in cm^2
n_T = 2.065;                      // Average number of fission neutrons emitted per neutron absorbed
t_TM = 368;                       // Diffusion time of moderator (Sodium) in cm^2
// 1 barn = 10^(-24) cm^2
sigma_aM = 0.0034*10^(-24);    // Microscopic
```

```
absorption cross section of Sodium in cm^2
sigma_aF = 681*10^(-24);        // Microscopic absorption cross section of Uranium-235 in cm^2
g_a = 0.978;                    // Non 1/v factor
M_F = 235;                      // Molecular weight of Uranium-235
M_M = 12;                       // Molecular weight of Carbon-12
Z = (1+B^2*(L_TM2+t_TM))/(n_T-1-(B^2*t_TM)); // An intermediate factor
// Calculation
rho_M = 1.6;                    // Density of Graphite in g/cm^3
m_M = (4/3*%pi*R^3)*rho_M;      // Mass of moderator
// Calculation
m_F = ((Z*sigma_aM*M_F)/(g_a*sigma_aF*M_M))*m_M/1000;
// Result
printf("\n Critical mass = %2.1f kg \n",m_F);

// 3.
f = Z/(Z+1);                    // Thermal utilization factor
// Calculation
k_inf = n_T*f;
// Result
printf('\n Infinite Multiplication factor (k_inf) = %.2f \n',k_inf);

// 4.
// Calculation
L_T2 = (1-f)*L_TM2
// Result
printf("\n Thermal Diffusion area = %d cm^2 \n",L_T2);

// 5.
E_R = 3.2*10^(-11);             // Energy per fission
```

```
    reaction in joules/second
N_A = 6.02*10^(23);            // Avogadro number (
    constant)
V = (4/3*%pi*R^3);             // Volume of the
    spherical reactor in cm^3
// Using the data from Tables 3.2
g_fF = 0.976;                  // Non 1/v factor
    Uranium-235 fuel
// Using the data from Tables II.2 for Uranium-235
sigma_f = 582*10^(-24);        // Microscopic fission
     cross section for Uranium-235 in cm^2
// Macroscopic fission cross section is calculated
    as follows
SIGMA_f = m_F*N_A*0.886*g_fF*sigma_f*1000/(V*M_F);

// From Table 6.2, the constant A can be calculated
    as
A = P/(4*(R^2)*E_R*SIGMA_f);

// The expression for thermal flux is
printf(" \n The expression for thermal flux = %4.3E
    sin (Br)/r \n",A);
// The maximum value of thermal flux is given at
    distance equal to zero
phi_T0 = A*B;
// Result
printf(" The maximum thermal flux = %4.3E neutrons/
    cm^2-sec \n",phi_T0);
// There is a slight variation in the values of
    diffusion area and constant A as compared from
    the textbook. This is due to approximation of
    values in textbook.
```

Scilab code Exa 6.6 Critical mass

```
// Example 6.6
clear all;
clc;

// Given data
rho_F = 0.0145;             // Density of Uranium-235 in the mixture in g/cm^3
rho_M = 1;                  // Density of Water in the mixture in g/cm^3
M_M = 18;                   // Molecular weight of water
M_F = 235;                  // Molecular weight of Uranium-235

// 1.
// The ratio of number of atoms of Uranium-235 to water per cc is
NF_NM = (rho_F*M_M)/(rho_M*M_F);
// Using the data from Tables 3.2
g_aF = 0.978;               // Non 1/v factor of Uranium-235 fuel
g_aM = 2;                   // Non 1/v factor of Water
// Using the data from Table II.2 for Uranium-235
// 1 barn = 10^(-24) cm^2
sigma_aF = 681*10^(-24);  // Microscopic absorption cross section of Uranium-235 in cm^2
sigma_aM=0.333*10^(-24);  // Microscopic absorption cross section of Hydrogen in cm^2
// Using the data form Table 6.3 at temperature = 20 deg
n_T = 2.065;                // Average number of neutrons produced per neutron absorbed in fission
phisig_aF = 0.886*g_aF*sigma_aF;     // Average thermal absorption cross-section of fuel
phisig_aM = 0.886*g_aM*sigma_aM;     // Average thermal absorption cross-sections of moderator
Z = (NF_NM)*(phisig_aF/phisig_aM);   // Parameter Z
f = Z/(Z+1);                          // Thermal
```

```
utilization factor of the fuel
k_inf = n_T*f;                        // Infinite multiplication factor

// From Table 5.2 and 5.3
L_TM2 = 8.1;                        // Diffusion area in cm^2
t_T = 27;                           // Neutron age in cm^2
L_T2 = (1-f)*L_TM2;                 // Diffusion area of fuel moderator mixture
M_T2 = L_T2+t_T;                    // Migration area of fuel moderator mixture
// Buckling can be found as
B2 = (k_inf-1)/M_T2;
printf(" \n Using the buckling formula from Table 6.2 \n B^2 = (2.405/R)^2+(pi/H)^2 \n For minumum critical mass H = 1.82R \n");
// On solving for R in B^2 = 8.763/R^2
R = sqrt(8.763/B2);
H = 1.82*R;
// Result
printf(" \n The dimensions of the cylinder are");
printf(" \n Radius of cylinder = %2.1f cm \t Height of cylinder = %3.1f cm \n",R,H);

// 2.
V = %pi*R^2*H;                        // Reactor volume (in cc) assuming cylindrical geometry
// Calculation
m_F = rho_F*V;
printf(" \n The critical fuel mass = %2.1f kg \n",m_F/1000);
// There is a slight variation in the values of dimensions of cylinder and critical fuel mass as compared from the textbook. This is due to approximation of values in textbook.
```

Scilab code Exa 6.7 Critical concentration

```
// Example 6.7
clear all;
clc;

// Given data
R = 300;                          // Radius of the sphere
   in cm
M_M = 20;                         // Molecular weight of
   heavy water
M_F = 235;                        // Molecular weight of
   Uranium-235

// 1.
// Using the data from Table 5.2
Dbar_r = 0.84;                     // Diffusion coefficient
    of graphite in cm
Dbar_c = 0.87;                     // Diffusion coefficient
    of heavy water in cm
L_TM2 = 9400;                      // Diffusion area of
   heavy water in cm^2
L_r = 59;                          // Diffusion length of
   graphite in cm
// Using the data from Table 3.2
g_aF = 0.978;                      // Non 1/v factor
   Uranium-235 fuel
// Using the data from Table II.2 for Uranium-235
// 1 barn = 10^(-24) cm^2
sigma_aF = 681*10^(-24);                   // Microscopic
   absorption cross section of Uranium-235 in cm^2
SIGMA_aM = 9.3*10^(-5)*10^(-24);     // Macroscopic
   absorption coefficient of Heavy water in cm^(-1)
N = 0.03323;                               // Atomic
```

```
    density of heavy water
// Let BRcot(BR)= y
y = 1-((Dbar_r/Dbar_c)*((R/L_r)+1));
// Considering only the first solution , B*R=2.64
B = 2.64/R;
// Using the data form Table 6.3 at temperature = 20 deg
n_T = 2.065;                          // Average number of neutrons produced per neutron absorbed in fission
Z = (1+(B^2*L_TM2))/(n_T-1);          // A parameter
sigma_aM = sqrt(4/%pi)*SIGMA_aM/N;    // Microscopic absorption cross section of Heavy water in cm^2
// The ratio of densities of fuel to moderator
rho_FM = Z*(M_F*sigma_aM)/(M_M*g_aF*sigma_aF)
rho_M = 1.1;                          // Density of Heavy water in g/cm^3
// Calculation
rho_F = rho_FM*rho_M;
// Result
printf(" \n The critical concentration = %.4f g/litre \n",rho_F*1000);

// 2.
V = (4/3)*%pi*R^3;                    // Reactor volume (in cc) assuming spherical geometry
// Calculation
m_F = rho_F*V;
// Result
printf(" \n The critical fuel mass = %3.2f kg \n",m_F/1000);
```

Scilab code Exa 6.8 Critical radius

```
// Example 6.8
```

```
clear all;
clc;

// Given data
rho_F = 2*10^(-4);                // Concentration of Uranium-235 fuel in g/cm^3
rho_M = 1.6;                      // Concentration of graphite moderator in g/cm^3
M_F = 235;                        // Molecular mass of Uranium-235 fuel
M_M = 12;                         // Molecular mass of Graphite(Carbon) moderator

// 1.
// Using the data from Tables 3.2
g_aF = 0.978;                     // Non 1/v factor Uranium-235 fuel
// Using the data from Table II.2 for Uranium-235 and Carbon
// 1 barn = 10^(-24) cm^2
sigma_aF = 681*10^(-24);          // Microscopic absorption cross section of Uranium-235 in cm^2
sigma_aM = 3.4*10^(-3)*10^(-24); // Microscopic absorption cross section of Graphite in cm^2
Z = (rho_F*M_M*g_aF*sigma_aF)/(rho_M*M_F*sigma_aM);     // Parameter Z
f = Z/(Z+1);                        // Thermal utilization factor of the fuel
// Using the data form Table 6.3 at temperature = 20 deg
n_T = 2.065;                        // Average number of neutrons produced per neutron absorbed in fission
k_inf = n_T*f;                      // The infinite multiplication factor
// From Table 5.2
L_TM2 = 3500;                       // Diffusion area of Graphite in cm^2
```

```
L_r = 59;                         // Diffusion length of graphite in cm
L_T2 = (1-f)*L_TM2;               // Diffusion area of fuel moderator mixture
// Buckling can be found as
B = sqrt((k_inf-1)/L_T2);
// Calculation
R = acot(-1/(B*L_r))/B;
// Result
printf(" \n The critical radius of fuel loaded thermal reactor = %3.2f cm \n",R);

// 2.
// Reactor is bare or reflector is not present
// Calculation
R0 = %pi/B;
// Result
printf(" \n The critical radius of bare thermal reactor = %d cm \n",R0);
// There is a slight variation in the value of critical radius as compared from the textbook. This is due to approximation of the thermal utilization factor value in textbook.
```

Scilab code Exa 6.9 Critical radius and mass

```
// Example 6.9
clear all;
clc;

// Given data
rho_F = 0.0145;                   // Concentration of Uranium-235 fuel in g/cm^3
// Using the result of Example 6.6
M_T2 = 30.8;                      // Migration area in
```

```
    cm^2
B = 0.0529;                      // Buckling factor
delta = 7.2+0.1*(M_T2-40);       // Empirical formula for reflector savings
R0 = %pi/B;                      // The radius of the bare reactor
// Calculation
R = R0-delta;
m_F=rho_F*4/3*%pi*R^3;
// Result
printf(" \n The critical radius of reflected reactor = %3.2f cm \n",R);
printf(" \n The critical mass of reflected reactor = %3.2f kg \n",m_F/1000);
```

Scilab code Exa 6.10 Critical parameters for heterogenous reactor

```
// Example 6.10
clear all;
clc;

// Given data
N = 150;                    // Number of zirconium atoms for every uranium atom

// 1.
// Using the data of atom density of zirconium from Table II.3
N_Z = 0.0429;                   // Atom density of zirconium in terms of 10^(24)
sigma_tZ = 6.6;                 // Total cross section of zirconium in barns
// Using the data of cross section of uranium-235 from Table II.3
sigma_tU = 690;                 // Total cross section
```

```
of uranium in barns
N_25 = N_Z/N;                        // Atom concentration of uranium-235
// Calculation
lambda = 1/((sigma_tZ*N_Z)+(sigma_tU*N_25));
// Result
printf(" \n The mean free path of thermal neutrons = %3.1f cm \n",lambda);

// 2.
// Using the data of atom density of water from Table II.3
N_W = 0.0334;                        // Atom density of water in terms of 10^(24)
// As the water and zirconium occupy half of the volume
N_W = 0.5*0.0334;
N_Z = 0.5*0.0429;
// From the Figure 6.6
// Uranium is present in one third of the sandwich or \n one sixth of the entire area
N_25 = 2.86*10^(-4)/6;
// Using the data from Table 3.2
g_aF = 0.978;                   // Non 1/v factor Uranium-235 fuel
// Using the data from Table II.3 for microscopic absorption cross section
sigma_aU = 681;                 // Microscopic absorption cross section of Uranium-235 in barns
sigma_aZ = 0.185;               // Microscopic absorption cross section of Zirconium in barns
sigma_aW = 0.664;               // Microscopic absorption cross section of Water in barns
f = (N_25*g_aF*sigma_aU)/((N_25*g_aF*sigma_aU)+(N_Z*sigma_aZ)+(N_W*sigma_aW));      // Thermal utilization factor
// Using the data form Table 6.3 at temperature = 20 deg
```

```
n_T = 2.065;                    // Average number of
    neutrons produced per neutron absorbed in fission
// Calculation
k_inf = n_T*f;
// Result
printf("\n Infinite multiplication factor = %4.3f \n
    ",k_inf);
```

Scilab code Exa 6.11 Critical parameter for heterogenous reactor

```
// Example 6.11
clear all;
clc;

// Using the data from Table 3.2
g_a25=0.978;                   // Non 1/v factor Uranium
    -235 fuel for absorption
g_f25=0.976;                   // Non 1/v factor Uranium
    -235 fuel for fission
g_a28=1.0017;                  // Non 1/v factor Uranium
    -238 fuel for absorption

v_25=2.42;                     // Average number of
    neutrons in one fission of Uranium-235
// Using the data from Table II.3 for microscopic
    absorption and fission cross section
sigma_a25=681;                 // Microscopic absorption
    cross section of Uranium-235 in barns
sigma_a28=2.7;                 // Microscopic absorption
    cross section of Uranium-238 in barns
sigma_f25=582;                 // Microscopic fission
    cross section of Uranium-235 in barns

// Using the data of atom density of uranium and let
    N_28/N_25= N
```

```
N = 138;
// Calculation
n_T = (v_25*sigma_f25*g_f25)/((sigma_a25*g_a25)+(N*
    sigma_a28*g_a28));
// Result
printf("\n Average number of neutrons produced per
    neutron absorbed in fission = %3.2f \n",n_T);
```

Scilab code Exa 6.12 Critical parameter for heterogenous reactor

```
// Example 6.12
clear all;
clc;

// Given data
rdist = 25.4;                           // Distance
    between the rods in cm
a = 1.02;                               // Radius of a rod
     in cm
// From the Figure 6.9
b = rdist/sqrt(%pi);                   // Radius of
    equivalent cell in cm
// Using the data from Table 5.2
L_F = 1.55;                            // Diffusion length
     of uranium fuel in cm
L_M = 59;                              // Diffusion length
     of graphite moderator in cm
// Using the data from Table II.3 at thermal energy
SIGMA_aM = 0.0002728;                 // Macroscopic
    absorption cross section of graphite moderator in
     barns
SIGMA_aF = 0.3668;                    // Macroscopic
    absorption cross section of uranium fuel in barns
// Let
x = a/L_F;
```

```
y = a/L_M;
z = b/L_M;
// The series expansion relations are
F = 1+(0.5*(x/2)^2)-((1/12)*(x/2)^4)+((1/48)*(x/2)^6);
E = 1+(z^2/2)*(((z^2*log(z/y))/(z^2-y^2))-(3/4)+(y^2/(4*z^2)));
// Let the ratio of volumes of moderator to fuel is denoted by V
V = (b^2-a^2)/a^2;
// Calculation
f = 1/((SIGMA_aM*V*F/SIGMA_aF)+E);
// Result
printf("\n The thermal utilization factor = %.4f \n",f);
// There is a slight variation in the value as compared from the textbook. This is due to approximation of the parameters value in textbook.
```

Scilab code Exa 6.13 Critical parameter for heterogenous reactor

```
// Example 6.13
clear all;
clc;

// Using the data given in the problem 6.12
rdist = 25.4;                              // Distance between the rods in cm
a = 1.02;                                  // Radius of the rod in cm
b = rdist/sqrt(%pi);                       // Radius of equivalent cell
V = (b^2-a^2)/a^2;                         // Ratio of volumes of moderator to fuel
```

```
// Using the data from Table II.3 for Uranium-238 density and atom density
rho = 19.1;                                    // Uranium-238 density in g/cm^3
N_F = 0.0483;                                  // Atom density in terms of 10^(24)
// Using Table 6.5 for Uranium-238
A = 2.8;
C = 38.3;
// Using Table 6.6 for graphite
// Let zeta_M*SIGMA_sM = s
s = 0.0608;
I = A+C/sqrt(a*rho);                           // Empirical expression of resonance integral parameter
// Calculation
p = exp(-(N_F*I)/(s*V));
// Result
printf("\n Resonance escape probability = %.4f \n",p);
```

Chapter 7

The Time Dependent Reactor

Scilab code Exa 7.1 Reactor kinetics

```
// Example 7.1
clear all;
clc;

// Using the data form Table 6.3 at temperature = 20
    deg
n_T = 2.065;                  // Average number of
    neutrons produced per neutron absorbed in fission
// Using the data from Table 7.1
t_dM = 2.1e-4;                // The mean diffusion time
    of the moderator in seconds
k_inf = 1;                    // The reactor is critical
f = k_inf/n_T;                // Thermal utilization
    factor
// Calculation
t_d = t_dM*(1-f);
l_p = t_d;
// Result
printf(" \n The prompt neutron lifetime = %3.2E
    seconds \n",l_p);
```

Scilab code Exa 7.2 Reactor period

```
// Example 7.2
clear all;
clc;

// Given data
k_inf = 1.001;                          // Infinite
    multiplication factor
// From the Example 7.1
l_p = 1e-4;                             // Prompt
    neutron lifetime
// Calculation
T = l_p/(k_inf-1);
// Result
printf(" \n The response time of the reactor = %2.1f
     sec \n",T);
printf(" \n The reactor power will increase as exp(t
    /%2.1f), where ''t'' denotes the time in seconds
    \n",T);
```

Scilab code Exa 7.3 Reactivity

```
// Example 7.3
clear all;
clc;

// Given data
k_inf = 1.001;                          // Infinite
    multiplication factor
// Calculation
rho = (k_inf-1)/k_inf;
```

```
// Result
printf(" \n The reactivity = %.1E or %2.1f percent \n",rho,rho*100);
```

Scilab code Exa 7.4 Reactor period

```
// Example 7.4
clear all;
clc;

// Using the result of Example 7.1
lp = 1e-4;                              // Prompt neutron lifetime in seconds
// Using the result of Example 7.3
rho = 1e-3;                             // Reactivity
// By referring to Figure 7.2
printf(" \n Reactor period = 57 seconds \n");
```

Scilab code Exa 7.5 Reactivity

```
// Example 7.5
clear all;
clc;

// Using the result of Example 7.3
reactivity = 0.001;
// As the reactor is fueled with Uranium-235
bet = 0.0065;                           // Total delayed neutron fraction of all groups denoted by 'beta'
printf(" \n A dollar is worth 0.0065 in reactivity for Uranium-235 reactor. \n");
// Calculation
rho = reactivity/bet;
```

```
// Result
printf(" \n Reactivity = %4.3f dollars or %2.1f cents \n",rho,rho*100);
```

Scilab code Exa 7.6 Prompt drop

```
// Example 7.6
clear all;
clc;

// Given data
P0 = 500;                          // Reactor power in MW
rho = -0.1;                        // 10% in reactivity (Insertion of control rods correspond to negative reactivity)
// As the reactor is fueled with Uranium-235
bet = 0.0065;                      // Total delayed neutron fraction of all groups denoted by 'beta'

P1 = (bet*(1-rho)*P0)/(bet-rho);   // The drop in power level in terms of MW
// Assuming that negative reactivity is greater than 4%
T = 80;                            // Reactor period obtained from Figure 7.2 in seconds
t = 600;                           // Analysis time in seconds
// Calculation
P = P1*exp(-t/T);                  // Power level drop in MW
// Result
printf(" \n The power level drop after 10 minutes = %5.4f MW \n",P);
```

Scilab code Exa 7.7 Control worth of a center rod

```
// Example 7.7
clear all;
clc;

// Given data
H = 70;                                                   // Height of the cylinder in cm
R = H/2;                                                  // Diameter of the cylinder in cm
a = 1.9;                                                  // Radius of black control rod in cm
// From Table 6.2, Buckling can be found by
B0 = sqrt((2.405/R)^2+(%pi/H)^2);
// Using the data from Table 5.2 and 5.3
L_TM2 = 8.1;                                              // Diffusion area of water moderator in cm^2
t_TM = 27;                                                // Neutron age of water moderator in cm^2
// Using the data form Table 6.3 at temperature = 20 deg
n_T = 2.065;                                              // Average number of neutrons produced per neutron absorbed in fission
// Using the data from Table 5.2 and Table II.3
D_bar = 0.16;                                             // Thermal neutron diffusion coefficient in cm
SIGMA_t = 3.443;                                          // Total macroscopic cross section in cm^(-1)
f = (1+B0^2*(L_TM2+t_TM))/(n_T+B0^2*L_TM2);               // Thermal utilization factor
M_T2 = (1-f)*L_TM2+t_TM;                                  // Thermal migration area in cm^2
```

```
d = 2.131*D_bar*(a*SIGMA_t+0.9354)/(a*SIGMA_t+0.5098);  // Extrapolation distance
// Calculation
rho_w = (7.43*M_T2*(0.116+log(R/(2.405*a))+(d/a))^(-1))/((1+B0^2*M_T2)*R^2);
// Result
printf(" \n The worth of a black control rod = %.3f or %2.1f percent \n",rho_w,rho_w*100);
```

Scilab code Exa 7.8 Total control rod worth

```
// Example 7.8
clear all;
clc;

// Using the data and result from Example 7.7
f = 0.583;                                    // Thermal Utilization factor
L_TM2 = 8.1;                                  // Diffusion area of water moderator in cm^2
R = 35;                                       // Radius of the cylinder of the core in cm
a = 0.508;                                    // Radius of control rod in cm
Rc = sqrt(R^2/100);                           // Critical radius in cm
L_T = sqrt((1-f)*L_TM2);                      // Thermal diffusion length in cm
// The points of estimation are chosen as follows
y = a/L_T;
z = Rc/L_T;
// Using the data given in Table V.I for modified Bessel functions
I0_275 = 1.019;                               // I0 at 0.275
```

```
I1_275 = 0.1389;                                // I1 at 0.275
I1_189 = 1.435;                                 // I1 at 1.89
K0_275 = 1.453;                                 // K0 at 0.275
K1_275 = 3.371;                                 // K1 at 0.275
K1_189 = 0.1618;                                // K1 at 1.89
E = ((z^2-y^2)/(2*y))*(((I0_275*K1_189)+(K0_275*I1_189))/((I1_189*K1_275)-(K1_189*I1_275)));                // The lattice function
// Using the data from Table 5.2 and Table II.3
D_bar = 0.16;                                   // Thermal neutron diffusion coefficient in cm
SIGMA_t = 3.443;                                // Total macroscopic cross section in cm^(-1)
d = 2.131*D_bar*(a*SIGMA_t+0.9354)/(a*SIGMA_t+0.5098);  // Extrapolation distance
f_R = 1/((((z^2-y^2)*d)/(2*a))+E);              // Rod utilization parameter
// Calculation
rho_w = f_R/(1-f_R);
// Result
printf(" \n The total worth of the control rods = %.3f or %2.1f percent \n",rho_w,rho_w*100);
// There is a deviation in the value computed on comparison with the value given in the textbook. This is due to approximation of thermal diffusion area in the textbook.
```

Scilab code Exa 7.9 Total control rod worth

```
// Example 7.9
clear all;
clc;

// Given data
SIGMAa_bar = 0.2;                                   // Average macroscopic absorption cross section in cm^(-1)
L_T = 1.2;                                          // Thermal diffusion length in cm
// Converting the given dimensions from inches to centimeters
// 1 inch = 2.54 cm
// From Figure 7.9
l = 9.75*(2.54/2);                                 // Length of the half rod
a = 0.312*(2.54/2);                                // Thickness of the half rod
m = 44.5/sqrt(2);                                  // Closest distance between two rods

D_bar = SIGMAa_bar*L_T^2;                           // Thermal neutron diffusion coefficient in cm
d = 2.131*D_bar;                                    // Extrapolation distance in cm which is obtained for bare planar surface
f_R = ((4*(l-a)*L_T)/(m-(2*a))^2*(1/((d/L_T)+coth((m-(2*a))/(2*L_T)))));     // Rod utilization parameter
// Calculation
rho_w = f_R/(1-f_R);
// Result
printf(" \n The total worth of the control rods = %.3f or %.1f percent \n",rho_w,rho_w*100);
// There is a slight deviation in the value computed on comparison with the value given in the textbook. This is due to approximation of rod utilization parameter in the textbook.
```

Scilab code Exa 7.10 Total control rod worth

```
// Example 7.10
clear all;
clc;

// Given data
d = 5;                                    // Inner diameter of the tube in cm
a = d/2;                                  // Inner radius of the tube in cm
l = 76;                                   // Length of the tube in cm
rho = 2;                                  // Density of B4C in g/cm^3
n = 5;                                    // Number of rods in tbe reactor
m_B4C = 2*(n*%pi*(a^2)*l);                // Mass of B4C in all the rods
// Using the data from standard periodic table
molwt_B = 10.8;                           // Molecular weight of Boron(B)
molwt_C = 12;                             // Molecular weight of Carbon(C)
molwt_B4C = (4*molwt_B)+molwt_C;          // Molecular weight of B4C
N_A = 0.6*10^(24);                        // Avogadro number
// From Table II.3
sigma_a = 0.27*10^(-24);                  // Microscopic absorption cross section of boron in cm^2
n_B = (4*m_B4C*N_A)/molwt_B4C;            // Number of boron atoms
```

```
// Using the result of Example 6.3
SIGMA_aF = 0.00833;                              //
    Macroscopic absorption cross section of plutonium
     fuel in cm^(-1)
SIGMA_aC = 0.000019;                             //
    Macroscopic absorption cross section of sodium
    coolant in cm^(-1)
R_c = 41.7;                                      //
    Critical radius in cm
N_B = n_B/((4/3)*%pi*R_c^3);                     // Atom
    density of boron over an entire reactor assuming
    spherical shape
SIGMA_aB = sigma_a*N_B;                          //
    Macroscopic absorption cross section of boron
// Calculation
rho_w = SIGMA_aB/(SIGMA_aF+SIGMA_aC);
// Result
printf(" \n The worth of the control rods using one
    group theory = %.4f or %.2f percent \n",rho_w,
    rho_w*100);
// In textbook, the final answer of total worth of
    control rods in percentage is wrong.
```

Scilab code Exa 7.11 Differential control rod worth

```
// Example 7.11
clear all;
clc;

// Given data
H = 70;                                          //
    Height of square cylindrical reactor in cm
rho_wH = 0.065;                                  //
    Total worth of a control rod at full height
rho_wx = 0.01;                                   //
```

```
   Total worth of a control rod to be achieved
// Let y-sin(y) = t
t = 2*%pi*(rho_wx/rho_wH);
// Using Newton Raphson method for solving the
   transcendental equation y - sin(y) -0.966 = 0
deff('y=f(y)','y = y-sin(y)-0.966')
deff('y=f1(y)','y = 1-cos(y)')
y0=0.5;                // Initial value
e = 0.00001;           // Relative error tolerance
for i=1:4
    y1 = y0-f(y0)/f1(y0)
    e1 = abs(y0-y1)
    y0 = y1;
    if abs(y0)<e then
        break;
    end
end
y = y1;                                          //
   The solution of transcendental equation
// Calculation
x = (y*H)/(2*%pi);
// Result
printf('\n The length of control rod to be inserted
   = %2.1f cm \n',x);
```

Scilab code Exa 7.12 Chemical Shim

```
// Example 7.12
clear all;
clc;

// Given data
f0 = 0.93;                                  // Thermal
    utilization factor
rho = 0.205;                                // Total
```

```
excess reactivity
rho_w = 0.085;                                  // Total worth of control rods
rho_sh = rho-rho_w;                             // Total worth of shim control
C = (rho_sh*10^3)/(1.92*(1-f0));                // Concentration of boric acid in ppm
printf('\n The minimum concentration of boric acid = %d ppm \n',ceil(C));
// Expressing in gram/litre
// Using the data from standard periodic table
molwt_B = 10.8;                                 // Molecular weight of Boron(B)
molwt_O = 16;                                   // Molecular weight of Oxygen(O)
molwt_H = 1;                                    // Molecular weight of Hydrogen(H)
molwt_H3BO3 = (3*molwt_H)+molwt_B+(3*molwt_O);          // Molecular weight of Boric acid
// Calculation
amt_H3BO3 = (molwt_H3BO3/molwt_B)*C/1000;
// Result
printf("\n The shim system must contain %3.2f g/litre of boric acid to hold down the reactor. \n",amt_H3BO3);
```

Scilab code Exa 7.13 Temperature coefficient of reactivity

```
// Example 7.13
clear all;
clc;

// Given data
p = 0.878;                                      // Resonance escape probability
```

```
T = 273+350;                                    // Given temeprature converted in Kelvin
d = 2.8;                                        // Diameter of rod in cm
a = d/2;                                        // Radius of rod in cm
rho = 19.1;                                     // Density of uranium in g/cm^3
// Using data from Table 7.4 for Uranium-238
A = 48*10^(-4);                                 // Constant value
C = 1.28*10^(-2);                               // Constant value
beta_I = A+C/(a*rho);                           // A parameter

// Calculation
alpha_prompt = -(beta_I/(2*sqrt(T)))*log(1/p);
// Result
printf('\n The prompt temperature coefficient = %.2E per K \n',alpha_prompt);
```

Scilab code Exa 7.14 Fission product poisons

```
// Example 7.14
clear all;
clc;

// Assuming that the fission product poisoning results in 12 barns per original Uranium-235 atom in a time frame of one year
sigma_p = 12;                                   // Microscopic poison cross section in barns
v = 2.42;                                       // Average number of neutrons produced in fission
```

```
// Using Table II.2 for fission cross section of
   Uranium-235 at thermal energy
sigma_f = 587;                                  //
   Microscopic fission cross section in barns
// Calculation
rho = -sigma_p/(v*sigma_f);
// Result
printf(" \n The reactivity due to poisons = %.5f or
   %.3f percent \n",rho,rho*100);
```

Chapter 8

Heat Removal from Nuclear Reactors

Scilab code Exa 8.1 Coolant temperature

```
// Example 8.1
clear all;
clc;

// Given data
P = 3025;                                   // Reactor thermal power in MW
w = 136.3*10^6;                             // Coolant flow rate in lb/hr
// According to Table 1.9
// 1 kW = 3412 Btu/hr
q = P*1000*3412;                            // Converting into Btu/hr
delh = q/w;                                 // Rise in enthalpy
// Using the data from Table IV.1 for temperature 542.6 F
hin = 539.7;                                // Enthalpy of input water in Btu/lb
```

```
// Calculation
hout = hin+delh;                                 // Enthalpy of released water in Btu/lb
// From Table IV.1
// Result
printf(" \n %3.1f Btu/lb corresponds to 599 F coolant water temperature. \n",hout);
```

Scilab code Exa 8.2 Steam temperature

```
// Example 8.2
clear all;
clc;

// Given data
P = 6.895;                                       // Pressure of steam in MPa
w = 2.93*10^6;                                   // Steam flow rate in kg/hr
Tin = 190.6+273;                                 // Inlet temperature in Kelvin

// 1.
// Using the data from Table IV.2
// Result
printf(" \n At a pressure of 6.895 MPa the steam temeperature is 284.86 C \n");

// 2.
// Using the data from Table IV.2
hout = 2773.2;                                   // Enthalpy of spent steam in kJ/kg
// Using the data from Table IV.1
hin = 807.8;                                     // Enthalpy of inlet steam at Tin in kJ/kg
```

```
// Calculation
q = w*(hout-hin);
// Result
printf(" \n Reactor power is %.3E J/hr or %d MW \n", q,q/(3600*1000));
```

Scilab code Exa 8.3 Heat production in fuel rods

```
// Example 8.3
clear all;
clc;

// Given data
n = 193*204;                                    // Total number of fuel rods in the reactor
// 1 feet = 12 inches
R = 67/12;                                      // Outer radius of the cylinder in feet(ft)
H = 144/12;                                     // Outer radius of the cylinder in ft
d = 0.42/12;                                    // Diameter of the fuel rod in ft
a = d/2;                                        // Radius of the fuel rod in ft
P = 1893;                                       // Reactor thermal power in MW
Ed = 180;                                       // Energy deposited locally in the fuel per fission in MW(Assumption)
ER = 200;                                       // Recoverable energy per fission in MW(Assumption)

// 1.
// Calculation
// According to Table 1.9
```

```
// 1 kW=3412 Btu/hr
q_r = (2.32*P*Ed)/(n*ER);
q_max=(q_r*3412*1000)/(2*H*a^2);
// Result
printf(" \n Total energy production at the axis = %.2E Btu/hr",(q_r*3412*1000));
printf(" \n Maximum energy production at the axis = %.2E Btu/hr-ft^3 \n",q_max);

// 2.
r = 20/12;                                   // Distance from the axis in ft
j0 = besselj(0,((2.405*r)/R));               // Bessel function
// Calculation
// According to Table 1.9
// 1 kW=3412 Btu/hr
q_r20 = q_r*j0;
q_max20 = (q_r20*3412*1000)/(2*H*a^2);
// Result
printf(" \n Total energy production at a distance of 20 inches = %.2E Btu/hr",(q_r20*3412*1000));
printf(" \n Maximum energy production at a distance of 20 inches = %.2E Btu/hr-ft^3 \n",q_max20);
```

Scilab code Exa 8.4 Decay energy

```
// Example 8.4
clear all;
clc;

// Given data
P0 = 825;                                    // Reactor thermal power in MW
t0 = 1.5*3.16*10^7;                          //
```

```
Reactor operation time in seconds
ts = 0.1;                                   // Reactor shutdown time in seconds

// 1.
// Let P/P0 = q
// From Figure 8.3
q_ts = 0.07;                                // Fission product to decay power during shutdown time
q_t0 = 0.0007;                              // Fission product to decay power after operating time
q = q_ts-q_t0;                              // Net fission product to decay power
// Calculation
P = q*P0;
// Result
printf("\n Decay energy at shutdown = %2.1f MW",P);

// One hour after shutdown
ts1 = 3.6*10^3;                             // Reactor shutdown time in seconds
// Let P/P0=q
// From Figure 8.3
q_ts1 = 0.014;                              // Fission product to decay power at shutdown time
q_t0 = 0.0007;                              // Fission product to decay power after operating time
q1 = q_ts1-q_t0;
// Calculation
P1 = q1*P0;
// Result
printf("\n Decay energy one hour after shutdown = %2.1f MW",P1);

// One year after shutdown
```

```
ts2 = 3.16*10^7;                                    // Reactor shutdown time in seconds
// Let P/P0=q
// From Figure 8.3
q_ts2 = 0.00079;                                    // Fission product to decay power at shutdown time
// Now the operating time is t0+ts2 which can be denoted by t01
q_t01 = 0.00063;                                    // Fission product to decay power after operating time
q2 = q_ts2-q_t01;
// Calculation
P2 = q2*P0;
// Result
printf(" \n Decay energy one year after shutdown = %.3f MW \n",P2);

// 2.
C = 0.88;                                           // Conversion factor
// Using data from Table II.2
sigma_a25 = 681;                                    // Microscopic absorption cross section in barns
sigma_f25 = 582;                                    // Microscopic fission cross section in barns
// At shutdown time
P_29 = (2.28*10^(-3)*C*(sigma_a25/sigma_f25))*P0;
P_39 = (2.17*10^(-3)*C*(sigma_a25/sigma_f25))*P0;
printf(" \n Decay energy at shutdown with effect of Uranium-239 and Neptunium-239 decay = %2.2f MW and %2.2f MW respectively",P_29,P_39);

// One hour after shutdown
ts1 = 3600;                                         // TIme in seconds
P_291 = P_29*exp(-4.9*10^(-4)*ts1);
P_391 = P_39*(exp(-3.41*10^(-6)*ts1)-(7*10^(-3)*exp
```

```
(-4.9*10^(-4)*ts1)));
printf("\n Decay energy one hour after shutdown with effect of Uranium-239 and Neptunium-239 decay = %2.2f MW and %2.2f MW respectively",P_291,P_391);

// One year after shutdown
P_292 = 0;                                          // Half life of Uranium-239 is 23.5 minutes
P_392 = 0;                                          // Half life of Neptunium-239 is 2.35 days
printf("\n Decay energy one year after shutdown with effect of Uranium-239 and Neptunium-239 decay = %d MW and %d MW respectively",P_292,P_392);
// There is a slight deviation in the values as compared with the texbook. This is because of approximation of difference values in the textbook.
```

Scilab code Exa 8.5 Fuel rod temperature parameters

```
// Example 8.5
clear all;
clc;

// Given data
// 1 feet = 12 inches
d = 0.42/12;                                  // Diameter of the fuel rod in feet(ft)
a = d/2;                                      // Radius of the fuel rod in ft
b = 0.024/12;                                 // Thickness of Zircaloy-4 clad in ft
H = 12;                                       // Length
```

```
of fuel rod in ft
T_m = 3970;                              // Center temperature of fuel in F

// 1.
// Using the result of Example 8.3
q_max = 4.66*10^7;                       // Maximum heat flux at the center of the rod in Btu/hr-ft^3
// Calculation
q_bar = (a^2*q_max)/(2*(a+b));
// According to Table 1.9
// 1 kW=3412 Btu/hr
// Result
printf(" \n Heat flux of the fuel rod = %.2E Btu/hr-ft^2 or %3.1f W/cm^2 \n",q_bar,(q_bar*1000)/(3412*30.48^2));

// 2.
// Using the data from Table IV.6
k_f = 1.1;                               // Thermal conductivity of fuel rod in Btu/hr-ft-F
k_c = 10;                                // Thermal conductivity of cladding in Btu/hr-ft-F
R_f = 1/(4*%pi*H*k_f);                   // Thermal resistance of fuel in F-hour/Btu
R_c = log(1+(b/a))/(2*%pi*H*k_c);        // Thermal resistance of cladding in F-hour/Btu
// Calculation
T_c = T_m-(q_bar*2*%pi*(a+b)*H*(R_f+R_c));
// Result
printf(" \n Outer temperature of cladding = %d F \n",ceil(T_c));
```

Scilab code Exa 8.6 Coolant temperature

```
// Example 8.6
clear all;
clc;

// Given data
h = 7500;                                              // Heat transfer coefficient in Btu/hr-ft^2-F
// Using the result of Example 8.5
q_bar = 3.66*10^5;                                     // Heat flux of the fuel rod in Btu/hr-ft^2
T_c = 650;                                             // Outer temperature of cladding in F
// Calculation
T_b = T_c-(q_bar/h);
// Result
printf(" \n Temperature of water with respect to the midpoint of the hottest fuel rod = %d F \n",T_b);
```

Scilab code Exa 8.7 Fuel rod temperature parameters and coolant temperature

```
// Example 8.7
clear all;
clc;

// Given data
T_b0 = 543;                                         // Temperature of inlet coolant in F
w = 3148;                                           // Coolant rate per channel in lb/hr

// 1.
V_f = 1.15*10^(-2);                                 // Volume of fueled portion in ft^2
```

```
// Using the result of Example 8.3
q_max = 4.66*10^7;                                  // Maximum heat flux at the center of the rod in Btu/hr-ft^3
// From Table IV.3
c_p = 1.3;                                          // Specific heat at constant pressure in Btu/lb-F
// Calculation
T_bmax = T_b0+((2*q_max*V_f)/(%pi*w*c_p));
// Result
printf(" \n Exit temperature of the coolant = %d F \n",T_bmax);

// 2.
// Using the data of Example 8.6
h = 7500;                                           // Heat transfer coefficient in Btu/hr-ft^2-F
// Using the data of Example 8.5
d = 0.42/12;                                        // Diameter of the fuel rod in feet(ft)
a = d/2;                                            // Radius of the fuel rod in ft
b = 0.024/12;                                       // Thickness of Zircaloy-4 clad in ft
H = 12;                                             // Length of fuel rod in ft
A = 2*%pi*(a+b)*H;                                  // Area of the assumed cylinder in ft^2
R_h = 1/(h*A);                                      // Convective resistance in F-hour/Btu
alpha = %pi*w*c_p*R_h;                              // A parameter

// Using the result from Example 8.5
R_f = 6.03*10^(-3);                                 // Thermal resistance of fuel
R_c = 1.43*10^(-4);                                 // Thermal resistance of cladding
```

```
R = R_f+R_c+R_h;                                    // Total resistance
bet = %pi*w*c_p*R;                                  // A parameter denoted by 'beta'
// Calculation
T_cmax = T_b0+((q_max*V_f*R_h)*((1+sqrt(1+alpha^2))/alpha));
T_mmax = T_b0+((q_max*V_f*R)*((1+sqrt(1+bet^2))/bet));
// Result
printf("\n Maximum temperature of cladding and fuel = %d F and %d F respectively. \n",ceil(T_cmax),T_mmax);
```

Scilab code Exa 8.8 Reynolds number

```
// Example 8.8
clear all;
clc;

// Given data
d = 0.42;                                           // Diameter of the fuel rod in inches
b = 0.024;                                          // Thickness of Zircaloy-4 clad in inches
v = 15.6*3600;                                      // Speed of fluid in feet/hour
a = (d/2)+b;                                        // Radius of fuel rods in inches
P = 2000;                                           // Pressure of water in psi
T = 600;                                            // Water temperature in F
// Using the data from example 8.5
s = 0.6;                                            // Pitch
```

```
of square array in inches
D_e = 2*((s^2-(%pi*a^2))/(%pi*a));          // Equivalent diameter in inches
// Converting the units in terms of feet
D_e = D_e/12;
// Using tha Table IV.3 at given T and P value
rho = 42.9;                                  // Density of fluid in ft/hr
mu = 0.212;                                  // Viscosity of fluid in lb/hr-ft
// Calculation
Re = (D_e*v*rho)/mu;
// Result
printf(" \n Reylonds number = %d \n",Re);
if Re >= 10000 then
    printf(" \n As the reylonds number is greater than 10000, the flow is turbulent. \n");
end
// The value is different as compared to the textbook value. This is due to approximation of Reynolds number in the textbook.
```

Scilab code Exa 8.9 Heat transfer coefficient

```
// Example 8.9
clear all;
clc;

// Using the data from Example 8.8
s = 0.6;                                     // Pitch of square lattice in inches
d = 0.42;                                    // Diameter of the fuel rod in inches
b = 0.024;                                   // Thickness of Zircaloy-4 clad in inches
```

```
a = (d/2)+b;                                    // Radius of fuel rods in inches
D_e = 0.0427;                                   // Equivalent diameter in feet
Re = 484329;                                    // Reynolds number
PD = s/(2*a);                                   // The ratio of pitch to diameter of fuel rod
// For a square lattice
C = 0.042*PD-0.024;                             // A constant

// According to Table IV.3
c_p = 1.45;                                     // Specific heat at constant pressure in Btu/lb-F
mu = 0.212;                                     // Viscosity of fluid in lb/hr-ft
k = 0.296;                                      // Conductivity of fluid in Btu/hr-ft F
Pr=(c_p*mu)/k;                                 // Prandtl number
// The constants are assumed as
m = 0.8;
n = 1/3;
// Calculation
h = C*(k/D_e)*(Re)^m*Pr^n;
// Result
printf(" \n Heat transfer coefficient = %d Btu/hr-ft^2-F\n",h);
// The value is different as compared to the textbook value. This is due to approximation of Reynolds number in the textbook and in this problem actual value is considered.
```

Scilab code Exa 8.10 Boiling crisis

```
// Example 8.10
clear all;
clc;

// Using the data from Example 8.3 to 8.8
P = 2000;                                          // Pressure in psi
v = 15.6;                                          // Coolant velocity in ft/sec
D_e = 0.0427;                                      // Equivalent diameter in ft
d = 0.42;                                          // Diameter of the fuel rod in inches
b = 0.024;                                         // Thickness of Zircaloy-4 clad in inches
a = (d/2)+b;                                       // Radius of fuel rods in inches
T_b = 600;                                         // Bulk temeperature in F

// 1.
// Using Bernath correlation
// Calculation
T_wc = 102.6*log(P)-((97.2*P)/(P+15))-(0.45*v)+32;
// Result
printf(" \n Cladding temeperature = %d F\n",T_wc);

// 2.
D_i = (2*%pi*a)/(%pi*12);                           // Heated perimeter is (2*%pi*a)/12 in feet
// Calculation
h_c = 10890*((D_e)/(D_e+D_i))+((48*v)/D_e^0.6);
// Result
printf(" \n Heat transfer coefficient = %d Btu/hr-ft^2-F\n",h_c);

// 3.
// Calculation
```

```
q_c = h_c*(T_wc-T_b);
// Result
printf(" \n Critical heat flux = %.2E Btu/hr-ft^2\n",q_c);
// In the textbook, the unit of critical heat flux is wrong.
```

Scilab code Exa 8.11 Engineering sub factor

```
// Example 8.11
clear all;
clc;

// Given data
m_lbar = 0.457;                        // Average linear density of UO2 in lb/ft
sigma = 0.0122;                        // Standard deviation of set of measured linear densities of UO2
// Calculation
F_Eml = 1+(3*sigma)/m_lbar;
printf(" \n Engineering subfactor for the amount of fissile material = %.2f \n",F_Eml);
```

Scilab code Exa 8.12 Fuel rod calculation

```
// Example 8.12
clear all;
clc;

q_max = 539000;                               // Maximum heat flux  Btu/hr-ft^2
```

```
F = 2.8;                                          // Hot channel factor
P = 3000;                                         // Reactor thermal power in MW
// Expressing in Btu/hr
// According to Table 1.9, 1 kW = 3412 Btu/hr
P = P*3.412*10^6;                                 // Reactor thermal power in Btu/hr
l = 12;                                           // Length of fuel rod in ft
d = 0.5/12;                                       // Diameter of fuel rod in ft
r = d/2;                                          // Radius of fuel rod in ft

// 1.
q_av = q_max/F;
// Calculation
A = P/q_av;
// Result
printf("\n Total heat transfer area = %d ft^2\n",A);

// 2.
A_one = 2*%pi*r;                                  // The total surface area of one fuel rod
// Calculation
n = A/A_one;
// Result
printf("\n Number of fuel rods = %d \n",n);
// The value is different as compared to the textbook value. This is due to approximation of total heat transfer area in the textbook and in this problem actual value is considered. As total heat transfer area is further used to calculate number of fuel rods, therefore the difference in value exists.
```

Chapter 9

Radiation Protection

Scilab code Exa 9.1 Gamma ray interaction

```
// Example 9.1
clear all;
clc;

// Given data
e = 1.6*10^(-19);                                    //
   Electronic charge in couloumb(coul)
X = 1*10^(-3)/3600;                                  //
   Exposure rate in terms of R/sec
// According to the definition of Roentgen, 1 R =
   2.58*10^(-7) coul/g
R = 2.58*10^(-7);
// From standard table
// There is 0.001293 g of air per 1 cm^3 at 1
   atmospheric pressure at 0 C
density_air = 0.001293;
// Calculation
IR = (X*R*density_air)/e;
// Result
printf(" \n Rate of ions produced from gamma ray
   interaction = %d ions/cm^3-sec",IR);
```

Scilab code Exa 9.2 Absorbed dose rate and dose equivalent rate

```
// Example 9.2
clear all;
clc;

// According to the definition of radiation absorbed
    dose(rad), 1 rad/sec = 100 ergs/g-sec
// Given data
D = 5*10^(-3)/100;                                    //
   Absorbed dose in terms of rad/sec
// Expressing absorbed dose rate in SI units
// 1 Gray(Gy) = 100 rad
D_dot = D*3600/100;
// Using data from Table 9.2
Q = 1;                                                //
   Quality factor for gamma rays for tissue
// Calculation
H_dot = D_dot*Q;
printf(' \n Absorbed dose rate in a tissue = %.1f
   mGy/hr \n',D_dot*1000);
printf(' \n Dose equivalent rate in a tissue = %.1f
   mSv/hr \n',H_dot*1000);
```

Scilab code Exa 9.3 Acute radiation exposure

```
// Example 9.3
clear all;
clc;

// Given data
```

```
H = 25;                                 // Equivalent dose in rem
age = 30;                               // Age of worker in years
exp_age = 77;                           // Average age upto which a person lives in years
// Using data from Table 9.6
// Bone cancer
rc_bone = 0.2;                          // Risk coefficient per 10^6 rem/year
lp_bone = 10;                           // Latent period in years
// Probability of dying from bone cancer
p_bone=(H*rc_bone*(exp_age-(lp_bone+age)))/10^6;

// Breast cancer
rc_breast = 1.5;                         // Risk coefficient per 10^6 rem/year
lp_breast = 15;                          // Latent period in years
// Probability of dying from breast cancer
p_breast = (H*rc_breast*(exp_age-(lp_breast+age)))/10^6;

// Leukemia
rc_leukemia = 1;                        // Risk coefficient per 10^6 rem/year
lp_leukemia = 2;                        // Latent period in years
// Probability of dying from leukemia
p_leukemia = (H*rc_leukemia*(exp_age-(lp_leukemia+age)))/10^6;

// Lung cancer
rc_lung = 1;                            // Risk coefficient per 10^6 rem/year
lp_lung = 15;                           // Latent period in years
```

```
// Probability of dying from lung cancer
p_lung = (H*rc_lung*(exp_age-(lp_lung+age)))/10^6;

// Pancreatic cancer
rc_pancreas = 0.2;                                  // Risk coefficient per 10^6 rem/year
lp_pancreas = 15;                                   // Latent period in years
// Probability of dying from Pancreatic cancer
p_pancreas = (H*rc_pancreas*(exp_age-(lp_pancreas+age)))/10^6;

// Stomach cancer
rc_stomach = 0.6;                                   // Risk coefficient per 10^6 rem/year
lp_stomach = 15;                                    // Latent period in years
// Probability of dying from stomach cancer
p_stomach = (H*rc_stomach*(exp_age-(lp_stomach+age)))/10^6;

// Rest of alimentary cancer
rc_ali = 0.2;                                       // Risk coefficient per 10^6 rem/year
lp_ali = 15;                                        // Latent period in years
// Probability of dying from rest of alimentary cancer
p_ali = (H*rc_ali*(exp_age-(lp_ali+age)))/10^6;

// Thyroid cancer
rc_thy = 0.43;                                       // Risk coefficient per 10^6 rem/year
lp_thy = 10;                                         // Latent period in years
// Probability of dying from thyroid cancer
p_thy = (H*rc_thy*(exp_age-(lp_thy+age)))/10^6;
```

```
// All other type of cancer
rc_other = 1;                                   // Risk coefficient per 10^6 rem/year
lp_other = 15;                                  // Latent period in years
// Probability of dying from all other type of cancer
p_other = (H*rc_other*(exp_age-(lp_other+age)))/10^6;

// Calculation
p = p_bone+p_breast+p_leukemia+p_lung+p_pancreas+p_stomach+p_ali+p_thy+p_other;
// Result
printf('\n Probability that the worker will die from cancer = %.1E \n',p);

// The value obtained is different from the value given in the textbook. This is because of approximation of individual probabilities in the textbook.
```

Scilab code Exa 9.4 Acute radiation exposure

```
// Example 9.4
clear all;
clc;
H = 1;                          // Equivalent dos in rem
n = 10^6;                       // Population
// Given data

// Using the data of number of expected deaths of leukemia per 10^6 people from Table 9.9
// In utero age group
```

```
frac_utero = 0.011;                    // Fraction of population
riskyr_utero = 10;                     // Risk years
riskcoef_utero = 15;                   // Risk coefficient
// Number of deaths in utero is given by
deaths_utero = frac_utero*riskyr_utero*riskcoef_utero;

// In 0-0.99 age group
frac_0_099 = 0.014;                    // Fraction of population
riskyr_0_099 = 25;                     // Risk years
riskcoef_0_099 = 2;                    // Risk coefficient
// Number of deaths in 0-0.99 age group is given by
deaths_0_099 = frac_0_099*riskyr_0_099*riskcoef_0_099;

// In 1-10 age group
frac_1_10 = 0.146;                     // Fraction of population
riskyr_1_10 = 25;                      // Risk years
riskcoef_1_10 = 2;                     // Risk coefficient
// Number of deaths in 1-10 age group is given by
deaths_1_10=frac_1_10*riskyr_1_10*riskcoef_1_10;

// In 11-20 age group
frac_11_20 = 0.196;                    // Fraction of population
riskyr_11_20 = 25;                     // Risk years
riskcoef_11_20 = 1;                    // Risk coefficient
// Number of deaths in 11-20 age group is given by
deaths_11_20=frac_11_20*riskyr_11_20*riskcoef_11_20;

// In 21-30 age group
```

```
frac_21_30 = 0.164;                    // Fraction of population
riskyr_21_30 = 25;                     // Risk years
riskcoef_21_30 = 1;                    // Risk coefficient
// Number of deaths in 21-30 age group is given by
deaths_21_30=frac_21_30*riskyr_21_30*riskcoef_21_30;

// In 31-40 age group
frac_31_40 = 0.118;                    // Fraction of population
riskyr_31_40 = 25;                     // Risk years
riskcoef_31_40 = 1;                    // Risk coefficient
// Number of deaths in 31-40 age group is given by
deaths_31_40=frac_31_40*riskyr_31_40*riskcoef_31_40;

// In 41-50 age group
frac_41_50 = 0.109;                    // Fraction of population
riskyr_41_50 = 25;                     // Risk years
riskcoef_41_50 = 1;                    // Risk coefficient
// Number of deaths in 41-50 age group is given by
deaths_41_50 = frac_41_50*riskyr_41_50*riskcoef_41_50;

// In 51-60 age group
frac_51_60 = 0.104;                    // Fraction of population
riskyr_51_60 = 22.5;                   // Risk years
riskcoef_51_60 = 1;                    // Risk coefficient
// Number of deaths in 51-50 age group is given by
deaths_51_60 = frac_51_60*riskyr_51_60*riskcoef_51_60;

// In 61-70 age group
```

```
frac_61_70 = 0.08;
riskyr_61_70 = 15.1;
riskcoef_61_70 = 1;                    // Risk
   coefficient
// Number of deaths in 61-70 age group is given by
deaths_61_70=frac_61_70*riskyr_61_70*riskcoef_61_70;

// In 71-80 age group
frac_71_80 = 0.044;                    // Fraction of
   population
riskyr_71_80 = 9.1;                    // Risk years
riskcoef_71_80 = 1;                    // Risk
   coefficient
// Number of deaths in 71-80 age group is given by
deaths_71_80 = frac_71_80*riskyr_71_80*
   riskcoef_71_80;

// Age greater than 80
frac_80 = 0.02;                        // Fraction of
   population
riskyr_80 = 4.5;                       // Risk years
riskcoef_80 = 1;                       // Risk
   coefficient
// Number of deaths with age greater than 80 years
   is given by
deaths_80=frac_80*riskyr_80*riskcoef_80;

// Calculation
total_deaths = deaths_utero+deaths_0_099+deaths_1_10
   +deaths_11_20+deaths_21_30+deaths_31_40+
   deaths_41_50+deaths_51_60+deaths_61_70+
   deaths_71_80+deaths_80;
// Result
printf(" \n Number of cases or deaths expected from
   leukemia = %.1f \n",total_deaths);
```

Scilab code Exa 9.5 Chronic radiation exposure

```
// Example 9.5
clear all;
clc;

// Given data
H_year = 5;                                          // Equiavelnt dose per year in rem
start_age = 18;                                      // Initial age of the worker in years
ret_age = 68;                                        // Retirement age of the worker in years
// Using data from Table 9.6 with respect to Bone cancer
latent_period = 10;                                  // Latent period in years
plateau_period = 30;                                 // Plateau period in years
rc_bone = 0.2;                                       // Risk coefficient per 10^6 rem/year

n = ret_age-(start_age+latent_period);               // Number of years of accumulated dose
H = n*H_year;                                        // Total equivalent dose in rem
// Calculation
p_bone = (H*rc_bone*plateau_period)/10^6;
// Result
printf("\n The probability of dying from bone cancer = %.1E \n",p_bone);
```

Scilab code Exa 9.6 External exposure due to gamma rays

```
// Example 9.6
clear all;
clc;

// Given data
E =2 ;                                          // Energy of gamma radiation in MeV
X_dot = 1;                                      // Exposure rate in mR/hour
// Using the data from Table II.5
// Let mu_a/rho of air at 2 Mev be denoted as mu_rho
mu_rho = 0.0238;                                // Ratio of absorption coefficient to sensity of air in cm^2/g
// Calculation
I = X_dot/(E*mu_rho*0.0659);
printf(" \n The beam intensity of gamma radiation required = %d gamma rays/cm^2-sec \n",ceil(I));
```

Scilab code Exa 9.7 External exposure due to X rays

```
// Example 9.7
clear all;
clc;

// Given data
phi = 2.4*10^5;                                 // Flux in x-rays/cm^2-sec
// From Figure 9.9
// To receive an exposure rate of 1 mR/hr at 50 keV, the flux is 8*10^3 x-rays/cm^2-sec
phi_eq = 8*10^3;                                // Equivalent flux in x-rays/cm^2-sec
```

```
X_dot_eq = 1;                                    // Equivalent Exposure rate in mR/hr
X_dot = (phi*X_dot_eq)/phi_eq;                   // Exposure rate of the operator in mR/hr
// From Figure 9.10 at 50 kV energy, the energy dependent function is
f_bone = 3.3;
f_muscle = 0.93;
f_fat = 0.9;
// Using data from Table 9.2
Q = 1;                                           // Quality factor for x-rays
// Calculation
D_dot_bone = X_dot*f_bone*Q;                     // Dose equivalent rate in bone
D_dot_muscle = X_dot*f_muscle*Q;                 // Dose equivalent rate in muscle
D_dot_fat = X_dot*f_fat*Q;                       // Dose equivalent rate in fat
// Result
printf(" \n Dose equivalent rate in bone = %d mrem/hour \n",ceil(D_dot_bone));
printf(" \n Dose equivalent rate in muscle = %d mrem/hour \n",ceil(D_dot_muscle));
printf(" \n Dose equivalent rate in fat = %d mrem/hour \n",ceil(D_dot_fat));
```

Scilab code Exa 9.8 External exposure due to neutrons

```
// Example 9.8
clear all;
clc;

// Given data
phi_n = 20;                                   // Given
```

```
neutron flux in neutrons/cm^2-sec
// From Figure 9.12
// To receive an dose equivalent rate of 1 mrem/hr, the fast neutron flux is 7 neutrons/cm^2-sec
phi_n_eq = 7;
D_dot_eq = 1;
D_dot_n = (phi_n*D_dot_eq)/phi_n_eq;       // Dose rate due to fast neutron flux in mrem/hr
phi_th = 300;                               // Given thermal flux in neutrons/cm^2-sec
// From Figure 9.12
// To receive an dose equivalent rate of 1 mrem/hr, the thermal flux is 260 neutrons/cm^2-sec
phi_th_eq = 260;
D_dot_th = (phi_th*D_dot_eq)/phi_th_eq;   // Dose rate due to thermal neutron flux in mrem/hr
D_dot = D_dot_n+D_dot_th;                  // Total dose rate in mrem/hr
printf("\n The permitted weekly dose is 100 mrem \n");
D_dot_perm = 100;
// Calculation
t = D_dot_perm/D_dot;
printf(" \n The time of exposure upto a safe level = %.1f hour \n",t);
// The answer given in the textbook is wrong. This is because of wrong computation of total dose rate
```

Scilab code Exa 9.9 External exposure due to neutrons

```
// Example 9.9
clear all;
clc;

```

```
// Given data
fluence = 10^8;                                  // Given fluence neutrons/cm^2
// From Figure 9.12
// To receive an dose equivalent rate of 1 mrem/hr, the fast neutron flux is 7 neutrons/cm^2-sec
phi_eq = 7;                                     // Equivalent flux in neutrons/cm^2-sec
D_eq = 1;                                       // Equivalent dose rate in mrem/hr
// 1 hour = 3600 seconds
fluence_eq = phi_eq*3600;                       // Equivalent fluence in neutrons/cm^2
// Calculation
D = (fluence*D_eq)/fluence_eq;
// Result
printf(" \n Dose received due to exposure of accelerator source = %d mrem \n",D);
// The answer given in textbook is approximated to a nearest value.
```

Scilab code Exa 9.10 Internal exposure due to radionuclide

```
// Example 9.10
clear all;
clc;

// Given data
M = 20;                               // Mass of organ in grams

// a)
// Using the data from Table 9.15
T_12 = 8.04;                          // Radiological half life of Iodine-131 in days
```

```
T_12_b = 138;                         // Biological half life of Iodine-131 in days
lambda = 0.693/T_12;                  // Radiological decay constant of Iodine-131 in days^-1
lambda_b = 0.693/T_12_b;              // Biological decay constant of Iodine-131 in days^-1
lambda_e = lambda+lambda_b;           // Equivalent decay constant in days^-1
// Using the data from Table 9.15
zeta = 0.23;                          // Effective energy equivalent in MeV
q = 0.23;                             // The fraction of Iodine-131 that goes by inhalation
// Calculation
DCF = (51.1*zeta*q)/(M*lambda_e);
// Result
printf("\n The dose commitment factor by inhalation = %.2f rem/ucurie \n",DCF);

// b)
breathing_rate = 2.32*10^(-4);        // Normal breathing rate in m^3/sec
time = 2*3600;                        // Time of radiation exposure in seconds
I_conc = 2*10^(-9);                   // Iodine-131 concentration
C0 = breathing_rate*time*I_conc;      // Total intake of Iodine-131 by inhalation
// Calculation
H = C0*(DCF*10^6);                    // Using DCF in micro-curie
// Result
printf("\n The dose commitment to thyroid = %.2E rem = %.2f mrem \n",H,H*1000);
```

Scilab code Exa 9.11 Maximum Permissible Concentration

```
// Example 9.11
clear all;
clc;

// Given data
V_W = 2200;                                    // Volume of water inatke in terms of cm^3/day
// 1 litre = 1000 gram(g)
M = 43*1000;                                   // Mass of water present in standard man according to standards
// Using the data from Table 9.13
MPD = 0.1/7;                                   // Maximum Permissible Dose (MPD) in rem/day
// Using the data from Table 9.15
zeta = 0.01;                                   // Effective energy equivalent in MeV
q = 1;                                         // The fraction of Tritium that goes inside by ingestion
T_b = 11.9;                                    // Biological Half life of Tritium in years
lambda_b = 0.693/T_b;                          // Biological decay constant of Tritium in years^-1

// As biological and radiological half lives are less than 50 year intake period, the exponential term (exp(-lambda_e*50)) is neglected
// Maximum Permissible Concentration(MPC) for a 7 day or 168 hour week tritium dose
MPC_w_168 = (lambda_b*M*MPD)/(51.1*V_W*zeta*q);
printf("\n Maximum Permissible Concentration(MPC) for a 7 day or 168 hour week tritium dose for
```

```
occupational purpose = %.2f uCi/cm^3 \n",MPC_w_168);
// The exposure at work is doubled for 40 hour week as compared to 168 hour week
// For 40 hour week, with work of 5 days out of 7 day week according to a study
MPC_w_40 = MPC_w_168*2*(7/5);
printf("\n Maximum Permissible Concentration(MPC) for a 40 hour week tritium dose for occupational purpose = %.3f uCi/cm^3 \n",MPC_w_40);

// By analyzing the data of Table 9.13
// The whole body dose of general public is one tenth of the occupational purpose.
MPC_w_168_gp = MPC_w_168*0.1;
printf("\n Maximum Permissible Concentration(MPC) for a 7 day or 168 hour week tritium dose for general public = %.3f uCi/cm^3 \n",MPC_w_168_gp);
// The answer of Maximum Permissible Concentration(MPC) for a 168 hour week tritium dose for general public is given wrong in the textbook.
```

Scilab code Exa 9.12 Acute radiation exposure

```
// Example 9.12
clear all;
clc;

// Given data
no_home = 10^6;                          // Number of houses
no_resident = 4;                         // Number of residents in a home
total_time = 50;                         // Number of years the analysis is carried out
```

```
radon_concn_old = 1;                                    // Radon concentration in older uninsulated homes in terms of pCi/l
radon_concn_new = 6;                                    // Radon concentration in modern insulated homes in terms of pCi/l
time = 3500;                                            // Time spent in home by a person per year in hours
eq_concn = 0.5;                                         // Equilibrium concentration of 50%
// 1 year = 24*365 hours
X_increased = eq_concn*(radon_concn_new-radon_concn_old)*(time/(24*365));    // The increased exposure of radon per person

// Using the data of Radon risk assessment of United States of America
// There are nearly 100 cases of cancer per 10^6 persons at 1 pCi-year dose.
// Calculation
no_cancer = (no_home*no_resident)*total_time*(100/10^6)*X_increased;
// Result
printf("\n Number of additional cases of cancer from insulation of home = %d \n",no_cancer);
// There is a slight deviation in the value given in the textbook. This is because of approximation of the number of additional cases of cancer in the textbook.
```

Scilab code Exa 9.13 Acute radiation exposure

```
// Example 9.13
clear all;
clc;
```

```

// Given data
H_ext = 3;                                              // External dose in rem
H_wbL = 5;                                              // Annual whole body dose limit in rem
// Using the data from Table 9.17
// Annual Limit Intake (ALI) for inhalation of Iodine-131 is 54uCurie (Ci)
ALI = 54;
// Calculation
I = ALI*(1-(H_ext/H_wbL));
// Result
printf("\n Amount of Iodine-131 intake within safety limits = %d uCi \n",ceil(I));
```

Chapter 10

Radiation Shielding

Scilab code Exa 10.1 Gamma ray buildup factor

```
// Example 10.1
clear all;
clc;

// Given data
E0 = 2;                                   // Energy of gamma rays in MeV
a = 10;                                   // Thickeness of lead shield in cm
phi0 = 10^6;                              // Intensity of gamma rays in gamma-rays/cm^2-sec

// 1.
// Using the data from Table II.4 for E0 = 2 MeV
mu_rho = 0.0457;                          // The ratio of total attenuation coefficient to density in cm^2/g
// From standard data tables for lead
rho = 11.34;                              // Density of lead in g/cm^3
// Calculation
```

```
phi_u = phi0*exp(-(mu_rho*rho*a));
// Result
printf("\n Uncollided flux at the rear side of lead shield = %.2E gamma-rays/cm^2-sec \n",phi_u)

// 2.
// Using the data from Table 10.1 for 2 MeV of lead material
mua = mu_rho*rho*a;
B_4 = 2.41;                                        // Buildup factor if mu*a = 4
B_7 = 3.36;                                        // Buildup factor if mu*a = 7
// Using two point method of straight line for calculating buildup factor at mu*a
B_m = B_4+((mua-4)*((B_7-B_4)/(7-4)));
// Calculation
phi_b = phi_u*B_m;
// Result
printf("\n Buildup flux at the rear side of lead shield = %.2E gamma-rays/cm^2-sec \n",phi_b);

// 3.
// Using the data from Table II.5 for 2 MeV
mua_rho_air = 0.0238;                              // The ratio of total attenuation coefficient to density of air in cm^2/g
// Calculation
X_dot = 0.0659*E0*mua_rho_air*phi_b;
// Result
printf("\n Exposure rate at the rear side of lead shield = %.1f mR/hour \n",X_dot);
```

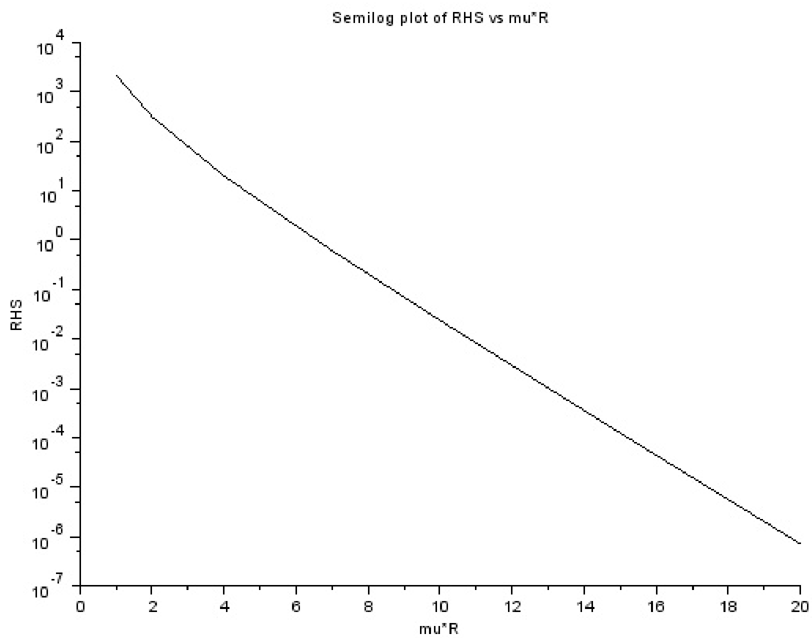

Figure 10.1: Gamma ray buildup factor

Scilab code Exa 10.2 Gamma ray buildup factor

```
// Example 10.2
clear all;
clc;

// Given data
E = 1;                                      // Energy of gamma rays in MeV
X_dot = 1;                                  // Exposure rate in mR/hour
phi0 = 10^8;                                // Intensity of gamma rays in gamma-rays/cm^2-sec from isotropic point source
// Using the data from Table II.5 for 1 MeV
mua_rho_air = 0.028;                        // The ratio of total attenuation coefficient to density of air in cm^2/g
phi_b = X_dot/(0.0659*E*mua_rho_air);       // Buildup flux in gamma-rays/cm^2-sec
// Using Eq 10.14
printf(" \n The equation to calculate radius is \n %.2E = %E * Bp*exp(-mu*R)/(4*pi*R^2) \n",phi_b,phi0);
// Using the data from Table II.4 for E = 1 MeV for Iron
mu_rho = 0.0595;                            // The ratio of total attenuation coefficient to density in cm^2/g
// From standard data tables for iron
rho = 7.864;                                // Density of iron in g/cm^3
mu = mu_rho*rho;
// On solving the right hand side of equation
// RHS = 3.22*10^3*Bp*exp(-mu*R)/(mu*R)^2
// Let mu*R = x
// Using the data from Table 10.2 for isotropic point source of 1 MeV incident on iron material
```

```
Bp = [1.87 2.89 5.39 10.2 16.2 28.3 42.7];
x = [1 2 4 7 10 15 20];
for i = 1:7
    RHS(i) = (3.22*10^3*Bp(i)*exp(-x(i))/x(i)^2);
end
plot2d("nl",x(:),RHS(:));
xlabel("mu*R");
ylabel("RHS");
title("Semilog plot of RHS vs mu*R")
// From the graph
muR = 6.55;                                      // This is the value when RHS = 1
// Calculation
R = muR/mu;
// Result
printf("\n The shield radius required = %d cm \n", ceil(R));
```

Scilab code Exa 10.3 Shielding of infinite planar source

```
// Example 10.3
clear all;
clc;

// Given data
E = 2;                                           // Energy of gamma rays in MeV
X_dot = 2.5;                                     // Exposure rate in mR/hour
phi0 = 10^9;                                     // Intensity of gamma rays in gamma-rays/cm^2-sec from isotropic point source
// Using the data from Table II.5 for 1 MeV
```

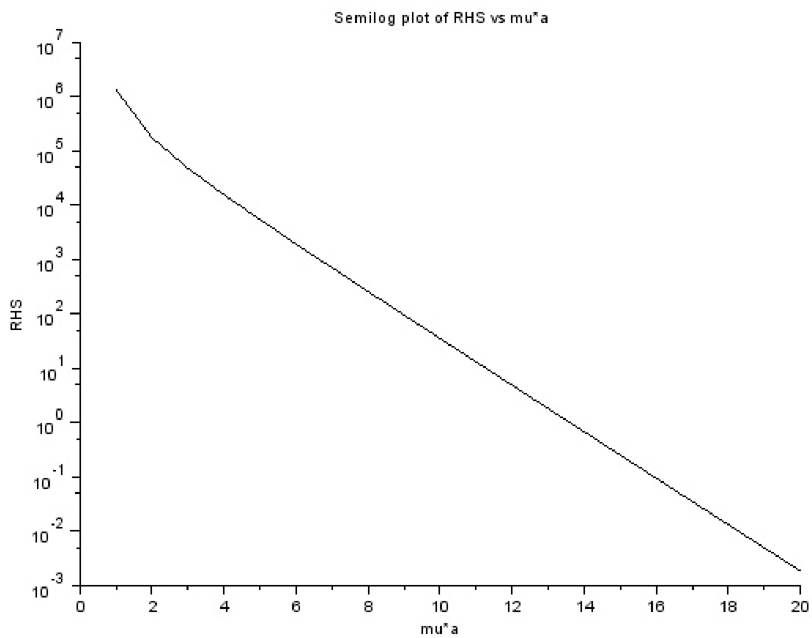

Figure 10.2: Shielding of infinite planar source

```
mua_rho_air = 0.0238;                                  // The ratio of total attenuation coefficient to density of air in cm^2/g
phi_b = X_dot/(0.0659*E*mua_rho_air);                 // Buildup flux in gamma-rays/cm^2-sec

// From standard data tables for concrete
rho = 2.35;                                            // Density of concrete in g/cm^3
// Using the data from Table 10.3 for concrete at 2 MeV
A1 = 18.089;
A2 = 1-A1;
alpha1 = -0.0425;
alpha2 = 0.00849;
// Using Eq 10.26
printf("\n The equation to calculate thickness is \n %.2E = (%E/2) *(%4.3f*E1(%4.3f*mu*a) %4.3f*E1(%4.3f*mu*a)) \n",phi_b,phi0,A1,(1+alpha1),A2,(1+alpha2));
// Using the data from Table II.4 for E = 1 MeV for concrete
mu_rho = 0.0445;                                      // The ratio of total attenuation coefficient to density in cm^2/g
mu = mu_rho*rho;
// On solving the right hand side of equation
// RHS = 1.13*10^7*(E1(0.9575*mu*a)-0.94*E1(1.00849*mu*a))
// Let mu*a = x
x = 1:20
for i = 1:20
    RHS(i) = 1.13*10^7*(exp(-0.9575*x(i))*((1/(0.9575*x(i))+(1/(0.9575*x(i))^3))) - exp(-1.00849*x(i))*((1/(1.00849*x(i))+(1/(1.00849*x(i))^3))));
end
plot2d("nl",x(:),RHS(:));
```

```
xlabel("mu*a");
ylabel("RHS");
title("Semilog plot of RHS vs mu*a")
// From the graph
mua = 13.6;                                        // This is the value when RHS = 1
// Calculation
a = mua/mu;
// Result
printf("\n The concrete thickness = %d cm \n",a);
```

Scilab code Exa 10.4 Multilayered shielding

```
// Example 10.4
clear all;
clc;

// Given data
E = 6;                                             // Energy of gamma rays in MeV
phi0 = 10^2;                                       // Intensity of gamma rays in gamma-rays/cm^2-sec from mono-directional beam
x_w = 100;                                         // Thickness of water in cm
x_Pb = 8;                                          // Thickness of lead in cm
// Using data from Table II.4 at 6 MeV
mu_w = 0.0275;                                     // Total attenuation coefficient of water in cm^(-1)
mu_Pb = 0.4944;                                    // Total attenuation coefficient of lead in cm^(-1)

mua_w = x_w*mu_w;                                  // Attenuation due to thickness of water
```

```
mua_Pb = x_Pb*mu_Pb;                              // Attenuation due to thickness of lead

// Case (a) - Water is placed before the lead
printf(" \n In case (a), Buildup factor is only due to lead measured at %.2f",mua_Pb);
// Using the data from Table 10.1 at 6 MeV
B_Pb = 1.86;
phi_b_a = phi0*B_Pb*exp(-(mua_w+mua_Pb));

// Using the data from Table II.5 for 6 MeV
mua_rho_air = 0.0172;                              // The ratio of total attenuation coefficient to density of air in cm^2/g
// Calculation
X_dot_a = 0.0659*E*mua_rho_air*phi_b_a;
// Result
printf("\n Exposure rate if water is placed before lead shield = %.2f uR/hour \n",X_dot_a*1000);

// Case (b) - Lead is placed before water
printf(" \n In case (b), Buildup factor is due to water and lead measured at %.2f and %.2f respectively",mua_w,mua_Pb);
// Using the data from Table 10.1 for water at 3.2 MeV,, which is the minimum point of mu_Pb curve
B_w = 2.72;
B_m = B_Pb*B_w;
phi_b_b = phi0*B_m*exp(-(mua_w+mua_Pb));

// Calculation
X_dot_b = 0.0659*E*mua_rho_air*phi_b_b;
// Result
printf("\n Exposure rate if lead is placed before water = %.2f uR/hour \n",X_dot_b*1000);
// The answer given in the textbook is wrong. This is because the intensity of gamma rays is wrongly taken for calculation.
```

Scilab code Exa 10.5 Removal cross section

```
// Example 10.5
clear all;
clc;

// Given data
fission_density = 4*10^7;                    // Fission density in fissions/cm^2-sec
// 1 inches = 2.54 cm
d = 28*2.54;                                 // Diamaeter of plate in cm
R = d/2;                                     // Radius of plate in cm
v = 2.42;                                    // Number of fission neutrons emitted per fission
x = 75;                                      // Distance of point from center of plate in cm
// Using the data from Table 10.4 for removal macroscopic cross section of water
sigma_RW = 0.103;                            // Removal macroscopic cross section of water in cm^-1
S = v*fission_density;                       // Strength of neutron source in terms of neutrons/cm^2-sec
A = 0.12;                                    // A constant
// From Figure 10.19
tan_theta = R/x;
theta = atan(R/x);
sec_theta = sec(theta);

// 1.
```

```
x11 = sigma_RW*x;                                   // Exponential integral function
x21 = sigma_RW*x*sec_theta;                         // Exponential integral function
// Let the upper limit of integral be 20
x_limit = 20;
function y=f(x),y=exp(-x)/x,endfunction
 E1_x11 = intg(x11,x_limit,f);
 E1_x21 = intg(x21,x_limit,f);
// Calculation
phi_1 = S*A/2*(E1_x11-E1_x21);
// Result
printf(" \n The fast flux without iron shield = %d neutrons/cm^2-sec \n",phi_1);

// 2. Iron slab is inserted in front of the fission plate
// Using the data from Table 10.4 for removal macroscopic cross section of iron
sigma_R = 0.168;                                   // Removal macroscopic cross section of iron in cm^-1
t = 3*2.54;                                         // Thickness of iron slab in cm
// Now the analysis is similar to multi layered shielding
x12 = sigma_RW*x+sigma_R*t;                         // Exponential integral function
x22 = sigma_RW*x*sec_theta+sigma_R*t*sec_theta; // Exponential integral function
// Let the upper limit of integral be 20
x_limit = 20;
function y=f(x),y=exp(-x)/x,endfunction
 E1_x12 = intg(x12,x_limit,f);
 E1_x22 = intg(x22,x_limit,f);
// Calculation
phi_2 = S*A/2*(E1_x12-E1_x22);
// Result
```

```
printf(" \n The fast flux with iron shield = %.1f neutrons/cm^2-sec \n",phi_2);
```

Scilab code Exa 10.6 Removal cross section

```
// Example 10.6
clear all;
clc;

// Given data
// Assuming average energy produced per fission reaction is 200 MeV
P = 250*10^3;                                  // Power of research reactor in Watts
P_fission = 200*10^6*1.6*10^(-19);             // Energy produced in a fission reaction in terms of joule
f = 0.75;                                      // Metal volume fraction
// In this problem, both reflector and shield act as a composite shield
a = 150+15;                                    // Net shield distance in cm
// 1 litre = 1000 grams
V = 32*1000;                                   // Core volume in gram

fission_density = (P/P_fission)*(1/V);
v = 2.42;                                      // Number of fission neutrons emitted per fission
S = fission_density*v;                         // Neutron source strength in neutrons/cm^3-sec
// Assuming spherical shape
// Volume of sphere = (4/3)*pi*(radius)^3
R = ((3*V)/(4*%pi))^(1/3);
```

```
// Using the data from Table 10.4 for removal macroscopic cross section
sigma_R = 0.174;                                    // Removal macroscopic cross section of uranium in cm^-1
sigma_RW = 0.103;                                   // Removal macroscopic cross section of water in cm^-1
A = 0.12;                                           // A constant
alpha = ((1-f)*sigma_RW)+(f*sigma_R);               // A parameter
// Calculation
theta = (S*A/(4*alpha))*(ceil(R)/(ceil(R)+a))^2*exp(-sigma_RW*a)*(1-exp(-2*alpha*ceil(R)));
// Converting into equivalent dose rate by referring Figure 9.12
// Fast neutron flux of 6.8 neutrons/cm^2-sec is equivalent to 1 mrem/hr
H_dot = theta/6.8;
// Result
printf(" \n Fast neutron dose rate near the surface of the shield = %.1f mrem/hour \n ",H_dot);
```

Scilab code Exa 10.7 Dose equivalent rate and thickness

```
// Example 10.7
clear all;
clc;

// Given data
E = 14;                                             // Energy of neutrons in MeV
phi0 = 10^9;                                        // Intensity of neutrons in neutrons/cm^2-sec from
```

```
isotropic point source
// 1 feet  = 30.48 cm
d = 10*30.48;                                        // Distance of concrete wall from a point source in cm
// As Intensity obeys inverse square law
I = phi0/(4*%pi*d^2);                                // Intensity of neutron beam in terms of neutrons/cm^2-sec
H_dot = 1;                                           // The required dose equivalent rate in mrem/hour
// From Figure 10.23(b)
H0_dot = H_dot/I;                                     // The dose equivalent rate
// Result
printf(" \n The reduced dose equivalent rate due to concrete wall is = %.2E mrem/hr \n",H0_dot);
// By looking into Figure 10.23(b) on thickness axis
printf(" \n The concrete slab thickness is = 70 cm \n");
```

Scilab code Exa 10.8 Shielding of prompt fission gamma rays

```
// Example 10.8
clear all;
clc;

// Given data
R = 7*30.48;                                         // Distance of core from the center of shield in cm
// Assuming average energy produced per fission reaction is 200 MeV
P = 10;                                              // Power of teaching reactor in Watts
P_fission = 200*10^6*1.6*10^(-19);                   //
```

```
Energy produced in a fission reaction in terms of joule
fission_rate = P/P_fission;                          // Number of fission reactions

// By assuming that the gammma rays are of equal energy of 1 MeV (Group 1) and looking into Table 10.5
E1 = 1;                                              // Energy of gamma rays in MeV (Assumed)
chi_pn1 = 5.2;                                       // Number of prompt gamma rays emitted per fission with energy 2 MeV
S1 = chi_pn1*fission_rate;                           // Source strength in gamma rays/sec
// Using the data from Table II.4 for E = 1 MeV for water
mu1 = 0.0996;                                        // Mass attenuation coefficient at 1 MeV in cm^-1
printf(" \n Buildup factor is due to water measured at %.2f",mu1*R);
// Using the data from Table 10.2 at 1 MeV
B_p1 = 373;
phi_b1 = (S1/(4*%pi*R^2))*B_p1*exp(-mu1*R);          // Buildup flux
// Using the data from Table II.5 for 1 MeV
mua_rho_air1 = 0.028;                                // The ratio of total attenuation coefficient to density of air in cm^2/g
// Calculation
X_dot1 = 0.0659*E1*mua_rho_air1*phi_b1;
printf("\n Exposure rate due to group 1 = %.4f mR/hour \n",X_dot1);

// By assuming that the gammma rays are of equal energy of 2 MeV (Group 2) and looking into Table 10.5
E2 = 2;                                              //
```

```
Energy of gamma rays in MeV (Assumed)
chi_pn2 = 1.8;          // Number of prompt gamma rays emitted per fission with energy 2 MeV
S2 = chi_pn2*fission_rate;          // Source strength in gamma rays/sec
// Using the data from Table II.4 for E = 2 MeV for water
mu2 = 0.0493;          // Mass attenuation coefficient at 2 MeV in cm^-1
printf(" \n Buildup factor is due to water measured at %.2f",mu2*R);
// Using the data from Table 10.2 at 2 MeV
B_p2 = 13.1;
phi_b2 = (S2/(4*%pi*R^2))*B_p2*exp(-mu2*R);          // Buildup flux
// Using the data from Table II.5 for 2 MeV
mua_rho_air2 = 0.0238;          // The ratio of total attenuation coefficient to density of air in cm^2/g
// Calculation
X_dot2 = 0.0659*E2*mua_rho_air2*phi_b2;
printf("\n Exposure rate due to group 2 = %.1f mR/hour \n",X_dot2);

// By assuming that the gammma rays are of equal energy of 4 MeV (Group 3) and looking into Table 10.5
E3 = 4;          // Energy of gamma rays in MeV (Assumed)
chi_pn3 = 0.22;          // Number of prompt gamma rays emitted per fission with energy 4 MeV
S3 = chi_pn3*fission_rate;          // Source strength in gamma rays/sec
// Using the data from Table II.4 for E = 4 MeV for water
mu3 = 0.0339;          //
```

```
Mass attenuation coefficient at 4 MeV in cm^-1
printf(" \n Buildup factor is due to water measured at %.1f",mu3*R);
// Using the data from Table 10.2 at 4 MeV
B_p3 = 5.27;
phi_b3 = (S3/(4*%pi*R^2))*B_p3*exp(-mu3*R);      // Buildup flux
// Using the data from Table II.5 for 4 MeV
mua_rho_air3=0.0194;                             // The ratio of total attenuation coefficient to density of air in cm^2/g
// Calculation
X_dot3 = 0.0659*E3*mua_rho_air3*phi_b3;
printf("\n Exposure rate due to group 3 = %.1f mR/hour \n",X_dot3);

// By assuming that the gammma rays are of equal energy of 6 MeV (Group 4) and looking into Table 10.5
E4 = 6;                                          // Energy of gamma rays in MeV (Assumed)
chi_pn4 = 0.025;                                 // Number of prompt gamma rays emitted per fission with energy 4 MeV
S4 = chi_pn4*fission_rate;                       // Source strength in gamma rays/sec
// Using the data from Table II.4 for E = 6 MeV for water
mu4 = 0.0275;                                    // Mass attenuation coefficient at 6 MeV in cm^-1
printf(" \n Buildup factor is due to water measured at %.2f",mu4*R);
// Using the data from Table 10.2 at 6 MeV
B_p4 = 3.53;
phi_b4 = (S4/(4*%pi*R^2))*B_p4*exp(-mu4*R);       // Buildup flux
// Using the data from Table II.5 for 4 MeV
mua_rho_air4=0.0172;                             //
```

```
   The ratio of total attenuation coefficient to density of air in cm^2/g
// Calculation
X_dot4 = 0.0659*E4*mua_rho_air4*phi_b4;
printf("\n Exposure rate due to group 3 = %.2f mR/hour \n",X_dot4);

//Calculation
X_dot = X_dot1+X_dot2+X_dot3+X_dot4;
// Result
printf("\n The total exposure rate due to all groups = %.2f mR/hour \n",X_dot);
```

Scilab code Exa 10.9 Coolant activity

```
// Example 10.9
clear all;
clc;

// Given data
// Assuming average energy produced per fission reaction is 200 MeV
P = 55;                                 // Power density of reactor in watts/cm^3
rho_eff_U = 0.33;                       // Effective density of uranium in g/cm^3
rho_eff_W = 1-rho_eff_U;                // Effective density of water in g/cm^3
t_i = 3;                                // Average time spent by water in the reactor in seconds
t_0 = 2;                                // Average time spent by water in the external coolant circuit in seconds
// 1 eV = 1.6*10^(-19) J
```

```
P_fission = 200*10^6*1.6*10^(-19);                   // Energy produced in a fission reaction in terms of joule
fission_density = P/P_fission;                       // Number of fission reactions
v = 2.42;                                            // Number of fission neutrons emitted per fission
S = v*fission_density;                               // Strength of neutron source in terms of neutrons/cm^2-sec
// Atom density of oxygen at normal density of 1 g/cm^3 is
rho = 1;                                             // Density of water in g/cm^3
N_A = 6.02*10^(23);                                  // Avogadro's constant
M = 18;                                              // Molecular weight of water
atom_density = (rho*N_A)/M;

// Using the data from Table 10.7
sigma_r = 1.9*10^(-5)*10^(-24);                      // Reaction cross section in cm^2
T_12 = 7.1;                                          // Half life of the given reaction in seconds
lambda = 0.693/T_12;                                 // Decay constant of the given reaction in seconds^(-1)
sigma_act = rho_eff_W*atom_density*sigma_r;          // Average macroscopic activation cross section
// Using the data from Table 10.4
sigma_RW = 0.103;                                    // Removal cross section of water in cm^-1
sigma_RU = 0.174;                                    // Removal cross section of Uranium in cm^-1
sigma_R = (sigma_RU*rho_eff_U)+(sigma_RW*rho_eff_W);        // Removal cross section of mixture
// Let activation rate given by (sigma_act*phi_av)
```

```
be denoted as AR
AR = (sigma_act*S)/sigma_R;
// Calculation
alpha = AR*(1-exp(-t_i*lambda))/(1-exp(-(t_i+t_0)*lambda));
// 1 curie = 3.7*10^(10) disintegrations/sec
// Result
printf("\n Equilibrium activity of water due to neutron capture of oxygen = %.2E disintegrations/cm^3-sec or %d uCi/cm^3 \n",alpha,ceil(alpha*10^6/(3.7*10^10)));
```

Chapter 11

Reactor Licensing Safety and the Environment

Scilab code Exa 11.1 Diffusion of radioactive effluents

```
// Example 11.1
clear all;
clc;

// Given data
h = 30;                                    // Height at which the effluent is relaesed
// Calculation of maxima location
sigma_z = h/sqrt(2);                       // Vertical dispersion coefficient
// Using the plot given in Figure 11.12 for Type F condition
// The corresponding value with calculated maximum location is noted.
h_max = 1900;
// Result
printf(" \n The point of maximum concentration of non-radioactive effluent = %d m \n",h_max);
```

Scilab code Exa 11.2 External dose from gamma rays

```
// Example 11.2
clear all;
clc;

// Given data
A = 2*10^3;                                    // Amount of radioactivity release due to Xenon-133 in curie
t = 365*24*3600;                               // Time in seconds
Q_dash = A/t;                                  // Average emission rate of Xenon-133
h = 100;                                       // Location of radioactivity release through vent
v_bar = 1;                                     // Wind speed in m/sec
// Using the plot given in Figure 11.11 and 11.12 for Type F condition at 100 m
sigma_y = 275;                                 // Horizontal dispersion coefficient
sigma_z = 46;                                  // Vertical dispersion coefficient
chi = (Q_dash*exp(-h^2/(2*sigma_z^2)))/(%pi*v_bar*sigma_y*sigma_z);      // Radionuclide concentration in terms of Ci/cm^3
// Using data from Table 11.3
Eg_bar = 0.03;                                 // Average gamma decay energy per disintegration in MeV
// Calculation
H_dot = 0.262*chi*Eg_bar*t*10^3;               // The units are in mrem/year
```

```
// Expressing the dose rate in SI units
H_dot_SI = 2.62*chi*Eg_bar*t*10^3;
// Result
printf(" \n The external gamma dose rate due to
    xenon release under type F condition = %.4f mrem/
    year or %.3f mSv/year \n",H_dot,H_dot_SI);
```

Scilab code Exa 11.3 External dose from gamma rays

```
// Example 11.3
clear all;
clc;

// Using data from Table 11.3
Eg_bar = 0.00211;                              //
    Average gamma decay energy per disintegration in
    MeV
// Calculation
C_y = 0.262*Eg_bar;
// Result
printf(" \n The dose rate factor due to krypton
    exposure = %.2E rem*m^3/sec-Ci \n",C_y);
```

Scilab code Exa 11.4 External dose from gamma rays

```
// Example 11.4
clear all;
clc;

// The results given in Example 11.2 are to be used
    in this problem
chi = 1.5*10^(-10);                                 //
    Radionuclide concentration in terms of Ci/cm^3
```

```
t = 365*24*3600;                                    // Time in seconds
// Using data from Table 11.3
Ebeta_bar = 0.146;                                  // Average gamma decay energy per disintegration in MeV
f = 1;                                              // Experimentally detemined factor
// Calculation
H_dot = 0.229*Ebeta_bar*chi*f*t;
// Expressing the result in milli-rem
printf(" \n The external beta dose rate due to xenon exposure for a year = %.3f mrem/year \n",H_dot*10^3);
```

Scilab code Exa 11.5 Dose rate to thyroid

```
// Example 11.5
clear all;
clc;

// Given data
A = 1.23;                                   // Amount of radioactivity release due to I-131 in curie in one year
h = 30;                                     // Location of radioactivity release through vent in meter
v_bar = 1.2;                                // Wind speed in m/sec
T_12 = 8.04;                                // Half life of Iodine 131 in days
T_12b = 138;                                // Biological half life of Iodine 131 in days
zeta = 0.23;                                //
```

```
Fraction of core inventory in MeV
q = 0.23;                                        // Fraction of Iodine-131 in thyroid
M = 20;                                          // Mass of an adult thyroid in grams

// 1.
t = 365*24*3600;                                 // Time in seconds
Q_dash = A/t;                                    // Average emission rate of Iodine-131
// Converting days into seconds by using 1 day = 86400 seconds
lambda = 0.693/(T_12*86400);                     // Decay constant of Iodine-131
lambda_b = 0.693/(T_12b*86400);                  // Biological decay constant of Iodine-131
// Let the quantity chi*v_bar/Q_bar = x
// Using the plot given in Figure 11.13 for Type E condition at 2000 m
x = 6*10^(-5);
// Solving for chi
chi = (x*Q_dash)/v_bar;
// As per the standards of International Commission on Radiolgical Protection (ICRP)
B = 2.32*10^(-4);                                // Normal breathing rate in m^3/sec
// Calculation
H_dot = (592*B*zeta*q*chi)/(M*(lambda+lambda_b));
// Result
printf(" \n The equilibrium dose rate to an adult thyroid = %.2E rem/sec \n",H_dot);

// 2.
// Calculation
H = H_dot*t;
// Expressing the result in milli-rem
// Result
```

```
printf(" \n The annual dose to an adult thyroid = %.2f mrem \n",H*10^3);
```

Scilab code Exa 11.6 Ground exposure from gamma rays

```
// Example 11.6
clear all;
clc;

// Given data
E = 0.66;                       // Energy of gamma ray emitted by caesium in MeV
x = 100;                        // Height of exposure in cm
// Using the data from Table II.5 for air at E = 0.66 MeV
mua_rho = 0.0294;               // The ratio of absorption coefficient to density of air in cm^2/g
// Using the data from Table II.4 for air at E = 0.66 MeV
mu_rho = 0.0775;                // The ratio of attenuation coefficient to density of air in cm^2/g
// Using standard value for density of air
rho = 1.293*10^(-3);
mu = mu_rho*rho;
mux = mu*x;
gamma = 0.57722;                // Euler's constant
E1 = -gamma+log(1/mux)+mux;     // Conversion factor
// Using parameter data from Table 11.16
C = 1.41;                       // A constant
```

```
beta = 0.0857;                                    // A constant
// Calculation
H_dot_S = 3.39*10^(-2)*E*mua_rho*(E1+(C*exp(-(1-beta)*mux)/(1-beta)));
// Converting time in hours by 1 hour = 3600 seconds
// Result
printf(" \n The gamma ray dose rate conversion factor due to caesium-137 = %.2E rem*m^2/sec-Ci or %.2f rem*m^2/hour-Ci\n",H_dot_S,H_dot_S*3600);
```

Scilab code Exa 11.7 External dose from gamma rays

```
// Example 11.7
clear all;
clc;

// Given data
C0 = 6.25*10^6;                                 // Amount of initial radioactivity release due to I-131 in curie
p = 0.1;                                        // Leakage rate in percent
t0 = 2*3600;                                    // Analysis time in seconds
v_bar = 1;                                      // Wind speed in m/sec

// 1.
lambda1 = 0.01*p/86400;                         // Decay constant corresponding to leakage rate in seconds (1 day = 86400 seconds)
// Using the data from Example 11.5 for the half life of Iodine-131
T_12 = 8.04;                                    // Half
```

```
life of Iodine 131 in days
lambdac = 0.693/(T_12*86400);                    // Decay constant of Iodine-131 (I-131) in seconds
// Using data from Table 11.3
Eg_bar = 0.371;                                  // Average gamma decay energy per disintegration of I-131 in MeV
// Using the plot given in Figure 11.11 and 11.12 for Type F condition at 100 m
sigma_y = 21;                                    // Horizontal dispersion coefficient
sigma_z = 70;                                    // Vertical dispersion coefficient
// As lambdac*t << 1,
// Calculation
H = (0.262*Eg_bar*lambdal*C0*t0)/(%pi*v_bar*sigma_y*sigma_z);
// Result
printf(" \n The total external dose due to gamma ray exposure = %.2E rem\n",H)

// 2.
// Using the data given in Example 11.5
B = 2.32*10^(-4);                                // Normal breathing rate in m^3/sec
zeta = 0.23;                                     // Fraction of core inventory in MeV
q = 0.23;                                        // Fraction of Iodine-131 in thyroid
M = 20;                                          // Mass of an adult thyroid in grams
// Calculation
H_dot = (592*B*zeta*q*lambdal*C0*t0)/(%pi*v_bar*sigma_y*sigma_z*M);
// Converting the units from rem/sec to milli-rem/hour by multiplying by (1000*3600)
// Result
printf(" \n The dose rate to an adult thyroid after
```

```
2 hours = %.2E rem/sec or %d mrem/hour\n",H_dot,ceil(H_dot*(1000*3600)));

// 3.
// Using the data given in Example 11.5
T_12 = 8.04;                                    // Half life of Iodine 131 in days
T_12b = 138;                                    // Biological half life of Iodine 131 in days
lambda = 0.693/(T_12*86400);                    // Decay constant of Iodine-131 in sec^(-1)
lambda_b = 0.693/(T_12b*86400);                 // Biological decay constant of Iodine-131 in sec^(-1)
// Calculation
H = H_dot/(lambda+lambda_b);
// Result
printf(" \n The dose commitment to thyroid by Iodine-131 exposure after 2 hours = %.2f rem \n",H);
```

Scilab code Exa 11.8 Direct dose from gamma rays

```
// Example 11.8
clear all;
clc;

// Given
E = 2.4;                                        // Energy of gamma rays in MeV
r = 1000*100;                                   // Distance from the building where radiation is exposed in cm
t0 = 2*3600;                                    // Time of exposure in seconds
A = 3*10^7;                                     //
```

```
   Amount of initial radioactivity release due to Kr-88 in curie
f = 0.4;                                        // Fraction of disintegrations which release 2.4 MeV gamma rays
C0 = A*f;                                       // Effective number of curies
T_12 = 2.79;                                    // Half life of Iodine 131 in hours

lambda = 0.693/(T_12*3600);                     // Decay constant of Iodine-131 in sec^(-1)
// Using the result given in Example 11.7
lambdal = 1.16*10^(-8);                         // Decay constant corresponding to fission prouduct release from building
lambdac = lambda+lambdal;                       // Total decay constant in sec^(-1)
// Using the data from Table II.4 for air at E = 2.4 MeV
mu_rho = 0.041;                                 // The attenuation coefficient in cm^2/g
// Using standard value for density of air in g/cm^3
rho = 1.293*10^(-3);
mu = mu_rho*rho;
// Using the data from Table II.5 for air at E = 2.4 MeV
mua_rho = 0.0227;                               // The ratio of absorption coefficient to density of air in cm^2/g
printf(" \n Buildup factor is measured at %.2f",mu*r);
// Using Berger's form in Problem 11.9
B_p = 4.7;                                      // Buildup factor due to a point source
// Calculation
H = (54*C0*(1-exp(-lambdac*t0))/lambdac)*(E*mua_rho*B_p*exp(-mu*r)/r^2);
```

```
// Result
printf(" \n The direct dose due to gamma ray exposure = %.4f rem \n",H)
// There is a slight deviation in the answer due to approximation of value in the textbook.
```

Scilab code Exa 11.9 Activity of radionuclide release into atmosphere

```
// Example 11.9
clear all;
clc;

// Given data
gammai = 0.0277;                                  // Fission yield of Iodine-131 in fraction
P = 3200;                                         // Thermal power of the plant in MW
// Calculation
alphai = 8.46*10^5*P*gammai;
// Result
printf(" \n The saturation activity of Iodine-131 during reactor operation = %.2E curie \n",alphai)

// Using assumption 2 of Nuclear Regulatory Commission (NRC) in calculation of radii of exclusion zone and Low Population Zone (LPZ)
// Due to core meltdown, 25 percent of iodine inventory is released and out of which 91 percent is in elemental form.
Fp = 0.25*0.91;                                   // Fraction of Iodine-131 released from the fuel into the reactor containment
// As entire iodine escapes through air
Fb = 1;                                           // Fraction of 'Fp' which remains airborne and is
```

```
    capable of escaping from the building
// Calculation
C0 = alphai*Fp*Fb;
// Result
printf("\n The activity of Iodine-131 in elemental form due to core meltdown = %.2E curie \n",C0);
```

Scilab code Exa 11.10 Concentration of radionuclide

```
// Example 11.10
clear all;
clc;

// Given data
P = 1000;                     // Electrical power of the plant in Mwe
eta = 0.38;                   // Plant efficiency
P_th = P/eta;                 // Thermal power of the plant in MW
h = 100;                      // Height of stack in metre
t = 24*365;                   // The number of hours in a year
m0 = 13000;                   // Amount of coal in the plant in Btu/lb
m0_ash = 0.09;                // Fraction of ash in the coal

// 1.
E = P_th*t;                   // Energy released in a year in MW-hour
// Converting the units in Btu by using 1 MW-hour = 3.412*10^6 Btu
m = (E*3.412*10^6)/m0;
```

```
// Converting the units in g/year by using 1 lb/year = 453.59237 g/year
m = m*453.59237;
// Assuming the fly ash equipment has an efficiency of 97.5% of fly ash removal
eta_flyash = 0.025;                    // Only (1-efficiency) is exhausted
m_ash = eta_flyash*m0_ash*m;
// A typical power plant contains 3pCi/g of each nuclide (Radium-226) in one year
A = 3*10^(-12);
// Calculation
A_total = A*m_ash;
// Result
printf("\n Total activity of Radium-226 emitted = %.4f curie \n",A_total)

// 2.
v_bar = 1;                    // Wind speed in m/sec
t = 24*365*3600;                    // Analysis time of one year equivalently in seconds
MPC = 3*10^(-12);                    // Maximum Permissible Concentration in micro-curie/cm^3
Q_bar = A_total/t;                    // Emission rate for one year in curie/year
// Let the quantity chi*v_bar/Q_bar = x
// Using the plot for Pasquill F, given in Fig. A.7, Pg no 413 from Slade, D. H., Editor, Meteorology and Atomic Energy-1968. U. S. Atomic Energy Commission Report TID-24190, 1968.
x = 2.5*10^(-6);                    // Maximum value of x at 10^4 m
// Solving for chi
chi = (x*Q_bar)/v_bar;
// Converting the units from Ci/m^3 to micro-curie/cm^3 by multiplying by 10^6/10^6  = 1
```

```
printf(" \n Concentration of Radium-226 present = %
    .2E micro-curie/cm^3 \n",chi)
// Let c be the comparison factor
// Calculation
c = MPC/chi;
// Result
printf(" \n On comparison, the total concentration
    of Radium-226 is %d times smaller than Maximum
    Permissible Concentration (MPC) \n",c)
// The comparison factor is slightly different from
    the value in the textbook. This is because of
    approximation used in the textbook.
```

Scilab code Exa 11.11 Activity of radioactive gas effluent

```
// Example 11.11
clear all;
clc;

// Given data
Qy_bar = 1.04*10^(-2);                          //
    Emission rate for one year in curie/year
// Let (chi/Q_bar) = d which is called 'Dilution
    factor'
d = 4*10^(-8);                                  //
    Dilution factor in year/cm^3
vd = 0.01;                                      //
    Experimentally determined constant

// 1.
T_12 = 8.04;                                    // Half
    life of Iodine 131 in days
T_12f = 14;                                     // First
     order half life of Iodine 131 in days
// Converting days into years by using 1 year = 365
```

```
days
lambda = 0.693/(T_12/365);                    // Decay constant of Iodine-131
lambdaf = 0.693/(T_12f/365);                  // First order decay constant of Iodine-131
// Calculation
Cf = (Qy_bar*d*vd)/(lambda+lambdaf);
// Expressing the result in micro-curie
Cf = Cf*10^6;
// Result
printf(" \n The activity of I-131 on the vegetation = %.2E micro-curie/m^2 \n",Cf);

// 2.
// The proportionality factor has a value 9*10^(-5) Ci/cm^3 of milk per Ci/m^2 of grass
// Calculation
Cm = 9*10^(-5)*Cf;
// Result
printf(" \n The concentration of I-131 in the milk = %.2E micro-curie/m^2 \n",Cm);

// 3.
MPC = 3*10^(-7);                              // Maximum Permissible Concentration in micro-curie/cm^3
// Calculation
H_dot = (2270*Cm)/MPC;
// Result
printf(" \n The annual dose rate to an infant thyroid by consuming radiated milk = %.2E mrem/year \n",H_dot);
```

Scilab code Exa 11.12 Activity of radioactive liquid effluent

```
// Example 11.12
clear all;
clc;

// Given data
Qy_bar = 0.197;                                    // Emission rate for one year in micro-curie/year
// Let (chi/Q_bar) = d which is called 'Dilution factor'
d = 6.29*10^(-16);                                 // Dilution factor in year/cm^3
MPC_w = 6*10^(-5);                                 // Maximum Permissible Concentration (MPC) of iron in micro-curie/cm^3

Cw = Qy_bar*d;                                     // The concentration of Fe-59
// For fish
Rs_fish = 110;                                     // Consumption rate in g/day
// Using the data from Table 11.15 for saltwater concentration of fish for iron
CF_fish = 1800;                                    // Concentration Factor of fish
Cs_fish = CF_fish*Cw;                              // Activity of fish
H_dot_fish = (Cs_fish*Rs_fish*500)/(MPC_w*2200);   // Dose rate for fish

// For mollusks
Rs_mollusk = 10;                                   // Consumption rate in g/day
// Using the data from Table 11.15 for saltwater concentration of mollusk for iron
CF_mollusk = 7600;                                 // Concentration Factor of mollusk
Cs_mollusk = CF_mollusk*Cw;                        // Activity of mollusk
```

```
H_dot_mollusk = (Cs_mollusk*Rs_mollusk*500)/(MPC_w*2200);    // Dose rate for mollusk

// For crustaceans
Rs_crustacean = 10;                                // Consumption rate in g/day
// Using the data from Table 11.15 for saltwater concentration of crustacean for iron
CF_crustacean = 2000;                              // Concentration Factor of crustacean
Cs_crustacean = CF_crustacean*Cw;                  // Activity of crustacean
H_dot_crustacean = (Cs_crustacean*Rs_crustacean*500)/(MPC_w*2200);  // Dose rate for crustacean

// Calculation
H_dot = H_dot_fish+H_dot_mollusk+H_dot_crustacean;
// Result
printf(" \n The annual dose rate to GI tract by consuming fish = %.2E mrem/year",H_dot_fish);
printf(" \n The annual dose rate to GI tract by consuming mollusk = %.2E mrem/year",H_dot_mollusk);
printf(" \n The annual dose rate to GI tract by consuming crustaceans = %.2E mrem/year",H_dot_crustacean);
printf(" \n The annual dose rate to GI tract by consuming seafood = %.2E mrem/year \n",H_dot);
// The answer for annual dose rate to GI tract by consuming fish is wrong in the textbook. This is because the value of fish consumption rate is wrongly considered.
```