Scilab Textbook Companion for
Irrigation and Water Power Engineering
by B. C. Punmia[1]

Created by
Ahmed Belal
B.Tech
Civil Engineering
Aligarh Muslim University
College Teacher
Dr Javed Alam
Cross-Checked by
K. V. P. Pradeep

December 16, 2014

Book Description

Title: Irrigation and Water Power Engineering

Author: B. C. Punmia

Publisher: Laxmi Publications Ltd., New Delhi

Edition: 16

Year: 2009

ISBN: 978-81-318-0763-7

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

7

List of Figures

Chapter 2

METHODS OF IRRIGATION

Scilab code Exa 2.1 EX2 1

```
1
2
3  //example 2.1
4  //calculate time required to cover 0.1 hectare area
       by tubewell
5  clc;
6  //Given
7  Q=0.0108,    //discharge through well
8  y=0.075,    //average depth of flow
9  I=0.05,     //average infiltration rate
10 A=0.1,     //area to cover
11 t=(60*2.303*y*log10(Q/(Q-I*A)))/I,
12 t=round(t);
13 mprintf("Time required to cover given area=%f
       minutes.",t);
```

Scilab code Exa 2.2 EX2 2

14

```
 1
 2
 3  //example 2.2
 4  //calculate maximum area that can be irrigated
 5  clc;
 6  //Given
 7  Q=0.0108,    //discharge through well
 8  y=0.075,    //average depth of flow
 9  I=0.05,    //average infiltration rate
10  A=0.1,    //area to cover
11  Amax=Q/I;
12  mprintf("Maximum area that can be irrigated =%f
        hectare.",Amax);
```

Scilab code Exa 2.3 EX2 3

```
 1  //example 2.3
 2
 3  //calculate
 4  //time of water application
 5  //optimum length of each border strip
 6  //dischrge for each border strip
 7
 8  clc;
 9  //Given
10  d=0.05;//depth of root zone
11  I=1.25D-5;//average infiltration rate
12  s=0.0035//slope of border strip
13  t=d/(I*3600);
14  t=round(t*1000)/1000;
15  mprintf("Time of water application=%f hours.",t);
16
17  //Part (a)
18  q=2D-3;//discharge entering water source
19  qdash=q*100^2*60;
```

```
20  n=0.55425-(0.0001386*qdash);
21  yo=(n*q/(s^0.5))^0.6;
22  y=0.665*yo;
23  L=(q/I)*(1-%e^(-d/y));
24  L=round(10*L)/10;
25  mprintf("\nPart (a):");
26  mprintf("\nOptimum length of each border strip=%f m.
        ",L);
27
28  //Part (b)
29  Lgiven=150//given value of length
30  //First Trial
31  q=3D-3;
32  qdash=q*100^2*60;
33  n=0.55425-(0.0001386*qdash);
34  yo=(n*q/(s^0.5))^0.6;
35  y=0.665*yo;
36  L=(q/I)*(1-%e^(-d/y));
37  //second trial
38  q=3.15D-3;
39  qdash=q*100^2*60;
40  n=0.55425-(0.0001386*qdash);
41  yo=(n*q/(s^0.5))^0.6;
42  y=0.665*yo;
43  L=(q/I)*(1-%e^(-d/y));
44  q=9*Lgiven*q*1000/L;
45  q=round(q*10)/10;
46  mprintf("\nPart (b):");
47  mprintf("\nDischarge for each border strip=%f lps.",
        q);
```

Scilab code Exa 2.4 EX2 4

```
1
2
```

```
3  //example 2.4
4  //calculate
5  //deep percolation loss
6  //water application efficiency and time of
        irrigation.
7
8  clc;
9  //Given
10 B=12;//breadth of basin
11 L=36//length of basin
12 d=70//depth of irrigation
13 Ic=70//cumulative infiltration
14 kdash=9;
15 ndash=0.42;
16 //Part (a)
17 a=5;
18 b=0.6;
19 q=1.5;//stream size
20 Q=(q*B)/1000;
21 tl=(L/a)^(1/b);
22 td=(Ic/kdash)^(1/ndash);
23 T=tl+td;
24 p=(1-(td/T)^(ndash))*100;
25 eita=(1-p/100)*100;
26 Tdash=(d*L*B)/(10*eita*Q*60);
27 p=round(p*100)/100;
28 eita=round(eita*100)/100;
29 Tdash=round(Tdash*10)/10;
30 mprintf("Part (a):")
31 mprintf("\nDeep percolation loss= %f percent.",p);
32 mprintf("\nWater application efficiency= %f percent.
        ",eita);
33 mprintf("\nTime of irrigation= %f minutes.",Tdash);
34 //part (b)
35 a=8;
36 b=0.6;
37 q=3;
38 Q=(q*B)/1000;
```

```
39  tl=(L/a)^(1/b);
40  td=(Ic/kdash)^(1/ndash);
41  T=tl+td;
42  p=(1-(td/T)^(ndash))*100;
43  eita=(1-p/100)*100;
44  Tdash=(d*L*B)/(10*eita*Q*60);
45  p=round(p*100)/100;
46  eita=round(eita*100)/100;
47  Tdash=round(Tdash*10)/10;
48  mprintf("\nPart (b):")
49  mprintf("\nDeep percolation loss= %f percent.",p);
50  mprintf("\nWater application efficiency= %f percent.
        ",eita);
51  mprintf("\nTime of irrigation= %f minutes.",Tdash);
```

Scilab code Exa 2.5 EX2 5

```
1  //example 2.5
2  //calculate
3  //size of cut-back stream.
4  //time required for putting 37.5 mm depth of water
5  //average depth of water applied
6
7  clc;
8  //given
9  d=37.5//crop water requirement
10  W=1//furrow spacing
11  L=120//length of furrow
12  n=-0.49;
13  k=38;
14  Ttotal=143;//Total time of irrigation
15  A=[0 23 52 88 127]//given values of time of advance
16
17  for i=1:5//loop to find respective values of time of
        ponding
```

```
18      B(i)=143-A(i);
19  end
20
21
22  for j=1:5//loop to find respective furrow
        infiltration
23      C(j)=B(j)^(n)*k;
24  end
25
26
27  for K=1:4//loop to find respective average
        infiltration
28
29      D(K)=(C(K)+C(K+1))/2;
30  end
31
32  E(1)=D(1);
33  for l=2:4//loop to determine cumulative infiltration
34      E(l)=D(l)+E(l-1);
35  end
36  I=E(4);
37
38  T=(30*d*W*(n+1)/k)^(1/(n+1));
39  dav=((24.5*Ttotal)+(I*(T-Ttotal)))/L;
40  q=((120*37.5)-(24.5*143))/62;
41  T=round(T);
42  dav=round(dav*10)/10;
43  q=round(q*100)/100;
44  I=round(I*100)/100;
45  mprintf("Maximum size of cut-back stream=%f lpm.",I)
        ;
46  mprintf("\nMinimum size of cut-back stream=%f lpm.",
        q);
47  mprintf("\nTime required for putting 37.5mm depth of
         water=%f minutes.",T);
48  mprintf("\nAverage depth of water required=%f mm.",
        dav);
```

Scilab code Exa 2.6 EX2 6

```
1
2  //example 2.6
3  //calculate Average depth of water applied
4  clc;
5  //Given
6  L=100;//length of furrow
7  W=1;//furrow spacing
8  s=0.3//longitudnal slope of furrow
9  t1=80//initial time flow of stream
10 t2=35//final time flow of stream
11 qm=0.6/s;
12 q=qm*0.4;
13 dav=((q*t2*60)+(2*t1*60))/100;
14 mprintf("Average depth of water applied=%f mm.",dav)
       ;
```

Scilab code Exa 2.7 EX2 7

```
1
2
3  //example 2.7
4  //calculate
5  //time required to irrigate
6  //maximum area that can be irrigated
7  clc;
8  //Given
9  Q=0.0072;//discharge through well
10 y=0.1;//average depth of flow
11 I=0.05//infiltration capacity of soil
12 A=0.04//area of land
```

```
13  t =(2.303* y *60/ I )* log10 ( Q /( Q - I * A ));
14  Amax = Q / I ;
15  t = round ( t *100)/100;
16  mprintf ("Time  required  to  irrigate=%f  minutes.", t );
17  mprintf ("\nMaximum  area  that  can  be  irrigated=%f  ha.
        ", Amax );
```

Chapter 3

WATER REQUIREMENTS OF CROPS

Scilab code Exa 3.1 EX3 1

```
1
2
3  //example 3.1
4  //classify the irrigation water
5  clc;
6  //Given
7  Na=24;                              //
       concentration of sodium ion
8  Ca=3.6;                             //
       concentration of calcium ion
9  Mg=2;                               //
       concentration of magnesium ion
10 EC=180;                             //
       electrical conductivity
11 SAR=Na/((((Ca+Mg)/2)^(0.5));        //
       Sodium absorption ratio
12 SAR=round(SAR*100)/100;
13 mprintf("SAR=%f.",SAR);
14 mprintf("\nWater falls under S2 class.");    //from
```

```
           table  3.2
15  mprintf("\nFor  EC=180,");
16  mprintf("\nwater  falls  under  C1  class.");      //from
           table  3.1
17  mprintf("\nWater  is  medium  sodium  and  low  saline
           water.");
```

Scilab code Exa 3.2 EX3 2

```
1
2
3  //example  3.2
4  //calculate
5  //Depth  of  moisture  in  root  zone  at  field  capacity
6  //Depth  of  moisture  in  root  zone  at  permanent
           wilting  point
7  //Depth  of  moisture  available  in  root  zone
8  clc;
9  //Given
10  gammad=15;          //dry  weigth  of  soil
11  gammaw=9.81;         //unit  weigth  of  water
12  Fc=0.3;            //field  capacity
13  pwp=0.08;          //permanent  wilting  point
14  d=0.8;            //root  zone  depth
15  d1=gammad*Fc*1000/gammaw;
16  d2=gammad*pwp*1000/gammaw;
17  d3=gammad*d*(Fc-pwp)*1000/gammaw;
18  d1=round(d1);
19  d2=round(d2);
20  d3=round(d3);
21  mprintf("Depth  of  moisture  in  root  zone  at  field
           capacity=%f  mm/m.",d1);
22  mprintf("\nDepth  of  moisture  in  root  zone  at
           permanent  wilting  point=%f  mm/m.",d2);
23  mprintf("\nDepth  of  moisture  available  in  root  zone=
```

23

```
   %f mm/m." ,d3);
```

Scilab code Exa 3.3 EX3 3

```
1
2
3  //example 3.3
4  //calculate
5  //depth upto which soil profile is wetted
6  clc;
7  //Given
8  gammad =15.3;        //dry weigth of soil
9  gammaw =9.81;        //unit weigth of water
10 Fc =0.15;            //field capacity
11 Mc =0.08;            //moisture content before
       irrigation
12 D=60;               //Depth of water applied
13 d=(gammaw*D)/(gammad*(Fc-Mc));
14 d=round(d);
15 mprintf("Depth upto which soil profile is wetted=%f
       mm." ,d);
```

Scilab code Exa 3.4 EX3 4

```
1
2
3  //example 3.4
4  //calculate
5  //Depth of water required to irrigate the soil
6  clc;
7  //Given
8  Sg =1.6;      //Apparent specific gravity
9  Fc =0.2;          //Field capacity
```

```
10  M1 =150;            //mass of sample soil
11  M2 =136;            //mass of sample after drying
12  d =0.9;             //depth of soil to be irrigated
13  Mc =( M1 - M2 )/ M2 ;
14  D=Sg*d*1000*(Fc-Mc);
15  D=round(D);
16  mprintf("Depth of water required to irrigate the
        soil=%f mm.",D);
```

Scilab code Exa 3.5 EX3 5

```
1
2
3  //example 3.5
4  //calculate Field Capacity
5  clc;
6  //Given
7  d=2;                //root zone depth
8  Wc=0.05;            //existing water content
9  gammad=15;          //dry density of soil
10  gammaw=9.81;       //unit weigth of water
11  Vw=500             //water applied to the soil
12  Wl=0.1;            //water loss
13  A=1000;            //area of plot
14  Vu=Vw*0.9;         //volume of water used in soil
15  Wu=Vu*gammaw;      //weigth of water used in soil
16  Ws=A*d*gammad;     //total dry weigth of soil
17  Wa=Wu*100/Ws;      //percent water added
18  Fc=Wc*100+Wa;
19  Fc=round(Fc*100)/100;
20  mprintf("The Field Capacity of soil is=%f percent.",
        Fc);
```

Scilab code Exa 3.6 EX3 6

```scilab
1
2
3  //example 3.6
4  //calculate
5  //storage capcity of soil
6  //Water depth required to be applied
7  clc;
8  //Given
9  Fc=0.22;
                                            //
      Field Capacity
10 wc=0.1;
      //wilting coefficient
11 gammad=15;
      //dry unit weigth of soil
12 gammaw=9.81;
      //unit wiegth of water
13 d=0.7;
      //root zone depth
14 w=0.14;
      //falled moisture content
15 E=0.75;                                  //
      water application efficiency
16 SC=gammad*d*(Fc-wc)*100/gammaw;
17 D=gammad*d*(Fc-w)*1000/gammaw;
18 FIR=D/E;                                 //
      Field irrigation requirement
19 SC=round(SC*10)/10;
20 D=round(D);
21 FIR=round(FIR)+1;
22 mprintf("Maximum storage capacity of soil=%f cm.",SC
      );
23 mprintf("\nWater depth required to be applied=%f mm"
      ,D);
24 mprintf("\nField Irrigation Requirement=%f mm",FIR);
```

Scilab code Exa 3.7 EX3 7

```
1
2
3  //example 3.7
4  //calculate watering frequency
5  clc;
6  //Given
7  Fc=0.27;            //Field capacity
8  pwp=0.14;           //permanent wilting point
9  gammad=15;          //dry density of soil
10 gammaw=9.81;        //unit weigth of water
11 d=0.75;             //effective depth of root zone
12 Du=11;              //daily consumptive use of water
13 Am=Fc-pwp;          //Available moisture
14 //let readily available moisture be 80 percent of
        available moisture
15 RAm=0.8*Am;
16 Mo=Fc-RAm;
17 D=gammad*d*(Fc-Mo)*100/gammaw;
18 WF=D*10/Du;
19 mprintf("Watering Frequency=%i days.",WF);
```

Scilab code Exa 3.8 EX3 8

```
1
2
3  //example 3.8
4  //calculate
5  //net depth of irrigation water reqiured
6  //time required to irrigate field
7  clc;
```

```
 8  //given
 9  Fc=0.22;          //Field capacity
10  Sg=1.56;          //Apparent specific gravity
11  d=0.6;            //root zone depth
12  //irrigation is started when 70 percent of moisture
       is used
13  l=250;            //length of field
14  b=40;             //width of field
15  q=20;             //Discharge
16
17
18  m=(1-0.7)*Fc;
19  D=Sg*d*(Fc-m)*1000;
20  A=l*b;
21  t=A*D/(q*3600);
22  D=round(D);
23  t=round(t);
24  mprintf("Net depth of irrigation water required=%f
       mm.",D);
25  mprintf("\nTime required to irrigate field=%f hours.
       ",t);
```

Scilab code Exa 3.9 EX3 9

```
 1
 2
 3  //example 3.9
 4  //calculate delta for crop
 5  clc;
 6  //Given
 7  B=110;        //Base period
 8  D=1400;       //Duty of water
 9
10  delta=8.64*B*100/D;
11  delta=round(delta);
```

```
12  mprintf("Delta for crop is=%f cm.",delta);
```

Scilab code Exa 3.10 EX3 10

```
1
2
3  //example 3.10
4  //calculate Duty of water
5  clc;
6  //Given
7  B=120;          //Base period
8  delta=92;       //total depth requirement of crop
9
10  D=8.64*B*100/delta;
11  D=round(D);
12  mprintf("Duty of water=%f hectares/cumec.",D);
```

Scilab code Exa 3.11 EX3 11

```
1
2
3  //example 3.11
4  //calculate Dischage required at head of canal
5  clc;
6  //Given
7  Cr=2;                //crop ratio
8  A=80000;             //Area of field
9  CI=85;               //percent field culturable
        irrigable
10  IK=30;               //irrigation intensity during
        kharif season
11  IR=60;               //irrigation intensity for rabi
        season
```

```
12  DuK =800;                 //Duty of water for kharif season
13  DuR =1700;                //Duty of water for rabi season
14
15  CIA = A * CI /100;    // Culturable irrigable area
16  AK = CIA * IK /100;   // Area under kharif season
17  AR = CIA * IR /100;   // Area under rabi season
18  DK = AK / DuK ;
19  DR = AR / DuR ;
20  mprintf (" Dischage required at head of canal during
        Kharif season=%f cumecs.", DK);
21  mprintf ("\nDischage required at head of canal during
         Rabi season=%f cumecs.", DR);
22  mprintf ("\nWater requirement during kharif is
        greater than during rabi season");
23  mprintf ("\nHence, canal should be designed to carry
        discharge of %f cumecs.", DK);
```

Scilab code Exa 3.12 EX3 12

```
1
2
3  //example 3.12
4  // calculate Dischage required at head of canal
5  clc ;
6  // Given
7  CA =2600;          // culturable area
8  IS =20;            // irrigation intensity for sugarcane
9  IR =40;            // irrigation intensity for rice
10  DuS =750;          //Duty of water for sugarcane
11  DuR =1800;         //Duty of water for rice
12  PK =1.2;           //Peak demand
13
14  AS = CA * IS /100;    //Area under sugarcane
15  AR = CA * IR /100;    //Area under rice
16  DS = AS / DuS ;
```

```
17  DR=AR/DuR;
18  DT=DS+DR;
19  DD=PK*DT-0.005333+0.01;
20  DR=round(DR*1000)/1000;
21  DT=round(DT*1000)/1000;
22  mprintf("Water required for Rice=%f cumecs.",DR);
23  mprintf("\n Sugarcane is a perennial crop.");
24  mprintf("\nHence,Water required for Sugarcane=%f
        cumecs.",DT);
25  mprintf("\nDesign dischage to meet the peak demand=
        %f cumecs.",DD);
```

Scilab code Exa 3.13 EX3 13

```
1
2
3  //example 3.13
4  //compare the efficiency
5  clc;funcprot(0);
6  //given
7  ql=20;              //discharge in left branch
8  Al=20000;           //culturable area in left branch
9  Bl=120;             //Base period in left branch
10 Il=0.8;             //intensity of rabi in left branch
11 qr=8;               //discharge in rigth branch
12 Ar=12000;           //culturable area in rigth branch
13 Br=120;             //Base period in rigth branch
14 Ir=0.5;             //intensity of rabi in rigth branch
15
16 //for left canal
17 ARl=Al*Il;
18 Dl=ARl/ql;
19 mprintf("Duty for left canal is=%i hectares/cumecs."
        ,Dl);
20
```

```
21  // for rigth canal
22  ARr = Ar * Ir;
23  Dr = ARr / qr;
24  mprintf("\nDuty for left canal is=%i hectares/cumecs
        .", Dr);
25  mprintf("\nSince,left canal has higher duty,it is
        more efficient");
```

Scilab code Exa 3.14 EX3 14

```
1
2
3  // example 3.14
4  // calculate Discharge for water course
5  clc;
6  // Given
7  CA=1200;          // culturable area
8  IA=0.4;           // intensity of irrigation of crop A
9  IB=0.35;          // intensity of irrigation of crop B
10 bA=20;            // kor period of crop A
11 bB=15;            // kor period of crop B
12 deltaA=0.1;         // kor depth of crop A
13 deltaB=0.16;          // kor depth of crop B
14
15 // crop A
16 A=CA*IA;
17 Du=8.64*bA/deltaA;
18 qA=A/Du;
19 qA=round(qA*1000)/1000;
20 mprintf("Discharge required for crop A=%f cumec.",qA
        );
21
22 // crop B
23 A=CA*IB;
24 Du=8.64*bB/deltaB;
```

```
25  qB=A/Du;
26  qB=round(qB*1000)/1000;
27  mprintf("\nDischarge required for crop B=%f cumec.",
        qB);
28  D=qA+qB;
29  D=round(D*10)/10;
30  mprintf("\nDesign discharge of water course=%f cumec
        .",D);
```

Scilab code Exa 3.15 EX3 15

```
1
2
3  //example 3.15
4  //calculate
5  //duty of irrigation water
6  //discharge required
7  clc;
8  //Given
9  B=12;           //transplantaion period
10 D=0.5;          //total depth of water required by the
        crop
11 R=0.1;          //rain falling on field
12 L=0.2;          //loss of water
13 A=600;          //irrigated area
14 I=0.6;          //intensity of irrigation
15 delta=D-R;
16 Dui=8.64*B/delta;
17 //since water loss is 20 percent
18 Du=(1-L)*Dui;
19 mprintf("Duty of water required=%f hectares/cumec.",
        Du);
20
21 TA=I*A;
22 q=TA/Du;
```

```
23  q=round(q*100)/100;
24  mprintf("\nDischarge  at  head  of  water  course=%f
         cumecs.",q);
```

Scilab code Exa 3.16 EX3 16

```
 1
 2
 3  //example 3.16
 4  //calculate
 5  //discharge required at the head
 6  //design discharge
 7  clc;
 8  //Given
 9  CF=0.8;                       //Capacity factory
10  Tf=13/20;                     //time factor
11  A=[850 120 600 500 360];      //given values of area
12  B=[320 90 120 120 120];       //given values of Base
         period
13  D=[580 580 1600 2000 600];    //given values of duty
         at head canal
14
15  DS=A(1)/D(1);                 //discharge for
         sugarcane
16  DOS=A(2)/D(2);                //discharge for
         overlap sugarcane
17  DW=A(3)/D(3);                 //discharge for wheat
18  DB=A(4)/D(4);                 //discharge for bajri
19  DV=A(5)/D(5);                 //discharge for
         vegetables
20  DR=DS+DW;
21  DM=DS+DB;
22  DH=DS+DOS+DV;
23  mprintf("Maximum demand is in hot weather");
24  q=DH/Tf;
```

```
25 D=q/CF ;
26 q=round (1000* q) /1000;
27 D=round (100*D) /100;
28 mprintf (" \nFull supply discharge at head=%f cumecs" ,
      q) ;
29 mprintf (" \nDesign discharge=%f cumecs . " ,D) ;
```

Scilab code Exa 3.17 EX3 17

```
 1
 2
 3 //example 3.17
 4 //calculate resvior capacity
 5 clc ;
 6 //Given
 7 CL =0.2;                      //Canal loss
 8 RL =0.12;                     //Reservior loss
 9 A=[4800 5600 2400 3200 1400];      //given values of
      area under crop
10 D=[1800 800 1400 900 700];       //given values of
      duty at field
11 B=[120 360 200 120 120];         //given values of
      base period
12
13 //(a) Wheat
14 d=A(1) /D(1) ;
15 V1=d*B(1) ;
16 //(b) Sugarcane
17 d=A(2) /D(2) ;
18 V2=d*B(2) ;
19 //(c) Cotton
20 d=A(3) /D(3) ;
21 V3=round (d*B(3) ) ;
22 //(d) Rice
23 d=A(4) /D(4) ;
```

```scilab
24  V4=round(d*B(4));
25  //(e) vegetables
26  d=A(5)/D(5);
27  V5=d*B(5);
28
29  Vd=(V1+V2+V3+V4+V5)*8.64;
30  SC=Vd/((1-CL)*(1-RL));
31  mprintf("Reservior capacity=%f hectare-metres.",SC);
32
33  //Alternative method
34
35  for i=1:5
36      delta(i)=8.64*B(i)/D(i);
37  end
38
39  for j=1:5
40      V(j)=A(j)*delta(j);
41  end
42  s=0;
43  for k=1:5
44      s=s+V(k);
45  end
46  SC=s/((1-CL)*(1-RL));
47  SC=round(SC);
48  mprintf("\n By Alternative method.\nStorage capacity
        =%f hectare-metres.",SC);
```

Scilab code Exa 3.18 EX3 18

```scilab
1
2
3  //example 3.18
4  //Calculate
5  //consumptive use
6  //consumptive irrigatin requirement
```

36

```scilab
7  //field irrigatio reqiurement
8  clc;
9  //Given
10 eita=0.7;            //water application efficiency
11 k=0.75;              //crop factor
12 T=[19 16 12.5 13];   //given values of temperature
13 p=[7.19 7.15 7.30 7.03];   //daytime hours of the
      year
14 RD=1.2;                   //rainfall in december
15 RJ=0.8;                   //rainfall in january
16 for i=1:4
17     f(i)=p(i)*(1.8*T(i)+32)/40;
18 end
19 s=0;
20 for i=1:4
21     s=s+f(i);
22 end
23 C=k*s;
24 R=RD+RJ;
25 CIR=C-R;
26 FIR=CIR/eita;
27 C=round(10*C)/10;
28 CIR=round(CIR*10)/10;
29 FIR=round(FIR*10)/10;
30 mprintf("Consumptive use=%f cm.",C);
31 mprintf("\nconsumptive irrigatin requirement=%f cm."
      ,CIR);
32 mprintf("\nfield irrigatio reqiurement=%f cm.",FIR);
```

Scilab code Exa 3.19 EX3 19

```scilab
1
2
3  //example 3.19
4  //calculate
```

```
5  //consumptive use of rice using penman's formula in
       january
6  clc;
7  //Given
8  L=20;                 //latitude of place(degree North)
9  T=15;                 //mean monthly temperature(degree
       celcius)
10 RH=0.5;               //relative humidity
11 E=250;                //elevation of area
12 V=25;                 //wind velocity at 2 m heigth
13 //from table 3.10
14 VP=12.79;             //saturation vapour pressure
15 s=0.8;                //slope of curve between vapur
       pressure and temperature
16 //from table 3.11
17 R=10.8;
18 //from table 3.12
19 N=11.1;
20 //from table 3.9
21 n=7.74;
22 p=n/N;
23 e=VP*RH;
24 Ea=0.002187*(160+V)*(VP-e);
25 r=0.2;
26 alpha=0.49;
27 sigma=2.01D-9;
28 Ta=293;
29 H=R*(1-r)*(0.29*cos(%pi/9)+0.55*p)-sigma*Ta
       ^4*(0.56-0.092*e^0.5)*(0.10+0.9*p);
30 Et=(s*H+alpha*Ea)*31/(s+alpha);
31 Et=round(Et*10)/10;
32 mprintf("consumptive use of rice in january=%f mm of
        water.",Et);
```

Scilab code Exa 3.20 EX3 20

```scilab
1
2
3  //example 3.20
4  //calculate
5  //maximum storage capacity;depth of irrigation water
6  //field irrigation requirement;water required at
      canal outlet
7  clc;
8  //Given
9  Fc=0.27;                  //Field capacity
10 pwp=0.13;                 //permanent wilting point
11 d=80;                     //depth of soil(cm)
12 gammad=1.5;               //dry unit weigth of soil(g/cc
      )
13 gammaw=1;                 //unit weigth of water(g/cc)
14 M=0.18;                   //avearge soil moisture
15 eita=0.8;                 //field efficiency
16 FC=0.15;                  //field channel
17 SC=gammad*d*(Fc-pwp)/gammaw;
18 D=gammad*d*(Fc-M)/gammaw;
19 FIR=D/eita;
20 W=FIR/(1-FC);
21 W=round(W*10)/10;
22 mprintf("maximum storage capacity=%f cm",SC);
23 mprintf("\ndepth of irrigation water=%f cm",D);
24 mprintf("\nfield irrigation requirement=%f cm",FIR);
25 mprintf("\nwater required at canal outlet=%f cm",W);
```

Scilab code Exa 3.21 EX3 21

```scilab
1
2
3  //example 3.21
4  //calculate reservior capacity
5  clc;
```

```
6  //given
7  W=0.4;                              //amount of water
       available from precipitation
8  Cl=0.15;                           //Channel loss
9  RL=0.1;                            //reservior loss
10 B=[120 320 120 200 100];    //Base period
11 D=[1800 800 900 1400 1200];//Duty at field
12 A=[500 600 300 1200 500];   //Area under crop
13
14 for i=1:5
15     delta(i)=8.64*B(i)/D(i);
16 end
17
18 for i=1:5
19     V(i)=delta(i)*A(i);
20
21 end
22 s=0;
23 for i=1:5
24     s=s+V(i);
25 end
26 C=s*(1-W)/((1-Cl)*(1-RL));
27
28 mprintf("Reservior capacity=%i ha-m.",C);
```

Scilab code Exa 3.22 EX3 22

```
1
2
3  //example 3.22
4  //calculate
5  //discharge required at head of distributory
6  clc;
7  //given
8  GCA=10000;                         //gross commanded area
```

```
9  CCA =0.75* GCA ;                          // Culturable  commanded
        area
10  IR =0.6;                              // intensity  of
      irrigation  during  rabi  season
11  IK =0.3;                              // intensity  of
      irrigation  during  kharif  season
12  DuR =2500;                            // duty  during  rabi
        season
13  DuK =1000;                            // duty  during  kharif
        season
14  AR = IR * CCA ;                        // area  under
      irrigation  in  rabi  season
15  AK = IK * CCA ;                        // area  under
      irrigation  in  kharif  season
16  DR = AR / DuR ;
17  DK = AK / DuK ;
18  mprintf (" discharge  required  at  head  of  distributory =
      %f  cumecs .", DK );
```

Scilab code Exa 3.23 EX3 23

```
1
2
3  // example  3.23
4  // calculate  irrigation  schedule
5  clc ;
6  // given
7  Fc =0.18;          // field  capacity
8  wc =0.07;          // wilting  cofficient
9  Sg =1.35;          // bulk  density  of  soil
10  d =1.2;           // root  zone  depth
11  m = Fc - wc ;
12  mo = wc + m /3;
13  dw =100* Sg * d *( Fc - mo );
14  mprintf (" Depth  of  water  required =%f  cm ", dw );
```

```
15  ev1 =1.1;              // average  evapotranspiration  rates
        in  1 NOV-30 NOV
16  ev2 =1.7;              // average  evapotranspiration  rates
        in  1 DEC-31 DEC
17  ev3 =2.4;              // average  evapotranspiration  rates
        in  1 JAN-31 JAN
18  ev4 =1.5;              // average  evapotranspiration  rates
        in  1 FEB-28 FEB
19  ev5 =3.5;              // average  evapotranspiration  rates
        in  1 MAR-25 MAR
20  // irrigation  requirement  from  1 NOV to  3 JAn
21  dev =( ev1 *30+ ev2 *31+ ev3 *3) /10;
22  mprintf (" \n\nWater  consumed  by  evapotranspiration=%f
        cm.", dev );
23  mprintf (" \nNo  water  is  required  during  1 NOV-3 JAN")
        ;
24
25  // irrigation  requirement  after  3rd JAN
26  ws =( ev3 -1.5) *16/10;            // water  consumed
        from  soil  from  4 JAN-19 JAN
27  ts=ws+dev ;                 // water  withdrawn  from
        soil  from  1 NOV-19 JAN
28  s =( dw -ts ) *10;
29  day=s/ev3 ;
30  depth=ts +(4* ev3 ) /10+(2* ev3 ) /10;
31  mprintf (" \n\ndepth  of  water  required  in  first
        irrigation=%f  cm.", depth );
32  /// irrigation  requirement  from  26 JAn to  25 MAR
33  w1=ev3 *6;
34  w2=ev4 *28;
35  w3=ev5 *25;
36  W=w1+w2+w3 ;
37  x =( dw *10 -(14.4+42) )/ ev5 ;
38  mprintf (" \n\nHence  second  irrigation  is  required
        after  %f  days  i.e on  18th  March.", x );
39  depth1 =(W -( dw *10) ) /10;
40  mprintf (" \nrequired  water  depth=%f  cm", depth1 );
41  mprintf (" \n\nFirst  Watering  on  29 JAn and  30 JAN=%f
```

42

```
        cm.\ nSecond watering required on 18th March=%f cm
        ." , depth , depth1 );
```

Scilab code Exa 3.24 EX3 24

```
 1
 2
 3  //example 3.24
 4  //calculate daily consumptive
 5  //discharge in canal
 6  clc;
 7  //given
 8  Fc=0.26;                //Field capacity of soil
 9  A=3000;                 //Area of field
10  OM=0.12;                //optimum moisture
11  pwp=0.1;                //permanent wilting point
12  d=80;                  //depth of root zone
13  RD=1.4;                 //relative density of soil
14  f=10;                   //frequency of irrigation
15  eita=0.23;              //overall efficiency
16  D=RD*d*(Fc-OM);
17  U=D*10/f;
18  Wr=A*D*100;
19  q=Wr/(f*24*3600);
20  q=round(q*100)/100;
21  mprintf("daily consumptive=%f mm.",U);
22  mprintf("\ndischarge in canal=%f q cumecs.",q);
```

Scilab code Exa 3.25 EX3 25

```
 1
 2
 3  //example 3.25
```

43

```scilab
4  //calculate total water to be delivered
5  clc;
6  //given
7  C1=0.2;                  //consumptive requirement of
      crop for 1 to 15 days
8  C2=0.3;                  //consumptive requirement of
      crop for 16 to 40 days
9  C3=0.5;                  //consumptive requirement of
      crop for 41 to 50 days
10 C4=0.1;                  //consumptive requirement of
      crop for 51 to 55 days
11 A=50;                    //area of land
12 wr=5;                    //presowing water requirement
13 R=3.5;                   //rainfall during 36th and 45th
       day
14 w1=15*C1*100;
15 w2=25*C2*100;
16 w3=10*C3*100;
17 w4=5*C4*100;
18 w5=5*100;
19 W=w1+w2+w3+w4+w5;
20 ER=3.5*100;
21 q=(W-ER)*A;
22 mprintf("total water to be delivered=%i cubic metre.
      ",q);
```

Scilab code Exa 3.26 EX3 26

```scilab
1
2
3  //example 3.26
4  //calculate watering interval
5  clc;
6  //given
7  Fc=0.3;              //field capacity
```

```
8  pwp=0.11;              //permanent wilting percent
9  gammad=1300;           //density of soil
10 gammaw=1000;           //density of water
11 d=700;                 //root zone depth
12 CW=12;                 //daily consumptive use of water
13 WHC=Fc-pwp;
14 mo=Fc-(0.75*WHC);
15 D=gammad*d*(Fc-mo)/gammaw;
16 I=D/CW;
17 mprintf(" watering interval=%i days",I);
```

Scilab code Exa 3.27 EX3 27

```
1
2
3  //example 3.27
4  //calculate Duty of water
5  //discharge required in water course
6  clc;
7  //given
8  A=1000;                    //total area
9  AI=0.7*A;                  //area under irrigation
10 B=15;                      //Base period
11 d=500;                     //depth of water required
      during transplantation
12 R=120;                     //useful rain falling
13 Wl=0.2;                     //water loss
14 delta=d-R;
15 Du=8.64*B*1000/delta;
16 DuH=Du*(1-Wl);
17 q=AI/DuH;
18 q=round(q*100)/100;
19 mprintf("Duty of water=%i hec/cumec.",Du);
20 mprintf("\ndischarge required in water course=%f
      cumecs.",q);
```

Scilab code Exa 3.28 EX3 28

```scilab
1
2
3 //example 3.28
4 //calculate reservior capacity
5 clc;
6 //given
7 Ar=4000;                    //culturable commanded area
8 CL=0.25;                    //canal loss
9 RL=0.15;                    //reservior loss
10 B=[120 360 180 120 120];   //base period
11 D=[1800 1700 1400 800 700];//duty of water
12 I=[20 20 10 15 15];        //intensity of irrigation
13 for i=1:5
14     A(i)=Ar*I(i)/10;       //area under crop
15 end
16 for i=1:5
17     Q(i)=A(i)/D(i);        //discharge required
18 end
19 for i=1:5
20     V(i)=8.64D4*Q(i)*B(i); //quantity of water
21 end
22 s=0;
23 for i=1:5
24     s=s+V(i);
25 end
26 SC=round(s/((1-CL)*(1-RL)*1000000));
27 mprintf("Storage capacity=%iD+06 cubic metre.",SC);
```

Chapter 4

HYDROLOGY

Scilab code Exa 4.1 EX4 1

```
1
2
3  //example  4.1
4  //calculate  mean  rainfall ; additional  guages  needed
5  clc ; funcprot (0) ;
6  //given
7  p =[78.8  90.2  98.6  102.4  70.4];    //rain  guage
       readings  at  respective  stations
8  s =0;
9  for  i =1:5
10      s=s+p(i) ;
11  end
12  pavg=s /5;
13  u =0;
14  for  i =1:5
15      u=u+(p(i)-pavg)^2;
16  end
17  sx =(u/4)^0.5;
18  Cv=sx *100/ pavg ;
19  N=(Cv/6)^2;
20  N=round(N*100) /100;
```

```
21  mprintf("mean rainfall=%f cm.",pavg);
22  mprintf("\ntotal stations needed=%f.",N);
23  //taking N=7
24  N=7;
25  n=N-5;
26  mprintf("\nadditional guages needed=%i.",n);
```

Scilab code Exa 4.2 EX4 2

```
1
2
3  //example 4.2
4  //calculate precipitation at A using arithmatic mean
        method
5  clc;funcprot(0);
6  //given
7  pB=48;            //precipitation at B
8  pC=51;            //precpitation at C
9  pD=45;            //precipitation at D
10
11  pA=(pB+pC+pD)/3;
12  mprintf("precipitation at A=%i mm.",pA);
```

Scilab code Exa 4.3 EX4 3

```
1
2
3  //example 4.3
4  //calculate precipitation at x
5  clc;funcprot(0);
6  //given
7  pA=6.6;           //precipitation at A
8  pB=4.8;           //precpitation at B
```

```
9  pC=3.7;              // precipitation at C
10 nA=72.6;             // normal precipitation at A
11 nB=51.8;             // normal precipitation at B
12 nC=38.2;             // normal precipitation at C
13 nX=65.6;             // normal precipitation at X
14
15 pX=(nX*pA/nA+nX*pB/nB+nX*pC/nC)/3;
16 pX=round(pX*100)/100;
17 mprintf("precipitation at x=%f cm.",pX);
```

Scilab code Exa 4.4 EX4 4

```
1
2
3  //example 4.4
4  //calculate precipitation at A by inverse distance
      method
5  clc;funcprot(0);
6  //given
7  pB=74;           // precipitation at B
8  pC=88;           // precpitation at C
9  pD=71;           // precipitation at D
10 pE=80;           // precipitation at E
11 Bx=9;By=6;
12 Cx=12;Cy=-9;
13 Dx=-11;Dy=-6;
14 Ex=-7;Ey=7;
15 Ax=0;Ay=0;
16 Db=(Bx^2+By^2);
17 Dc=(Cx^2+Cy^2);
18 Dd=(Dx^2+Dy^2);
19 De=(Ex^2+Ey^2);
20 Wb=1/Db;
21 Wc=1/Dc;
22 Wd=1/Dd;
```

49

```
23  We =1/ De ;
24  s = pB * Wb + pC * Wc + pD * Wd + pE * We ;
25  pA = s /( Wb + Wc + Wd + We ) ;
26  pA = round ( pA *10) /10;
27  mprintf ( " p r e c i p i t a t i o n  at  A=%f mm. " ,pA ) ;
```

Scilab code Exa 4.5 EX4 5

```
1
2
3  // example  4 . 5
4  // c a l c u l a t e  average  r a i n f a l l  u s i n g
5  // a r i t h m a t i c  average  method
6  // i s o h y t e l  method
7  // t h i e s s o n  polygon  method
8  clc ; funcprot (0) ;
9  // g i v e n
10  p =[58  61  69  56  84  86  69  79  71];   // v a l u e s  of
        p r e c i p i t a t i o n
11  s =0;
12  for  i =1:9
13      s = s + p ( i ) ;
14  end
15  ar = s /9;
16  ar = round ( ar *10) /10;
17  mprintf ( " u s i n g  a r i t h m a t i c  average  method : " )
18  mprintf ( " \ nAverage  r a i n f a l l=%f cm. " ,ar ) ;
19
20  I =[86  85  80  75  70  65  60  55  50];            // i s p h y t e s
21  A =[0.43  5.20  4.0  5.04  5.85  4.53  4.09  1.27];   // area
        between  i s o h y t e s
22  for  i =1:8
23      a ( i ) =( I ( i ) + I ( i +1) ) /2;
24  end
25  for  i =1:8
```

```
26      P(i)=A(i)*a(i);
27  end
28  s=0;
29  for i=1:8
30      s=s+P(i);
31  end
32  t=0;
33  for i=1:8
34      t=t+A(i);
35  end
36  ar=s/t;
37  ar=round(ar*10)/10;
38  mprintf("\n\nisohytel method:")
39  mprintf("\nAverage rainfall=%f cm.",ar);
40
41  A=[3.26 0.39 1.61 2.04 2.46 0.84 3.91 5.09 0.41 3.94
        2.06 4.40];    //thiessen area
42  p=[58 63 71 69 86 81 84 56 53 69 61 79];
        //observed precipitation
43  for i=1:12
44      P(i)=A(i)*p(i);
45  end
46  s=0;
47  for i=1:12
48      s=s+P(i);
49  end
50  disp(s);
51  t=0;
52  for i=1:12
53      t=t+A(i);
54  end
55  ar=s/t;
56  ar=round(ar*10)/10;
57  mprintf("\n\nthiesson polygon method:")
58  mprintf("\nAverage rainfall=%f cm.",ar);
59  //mean rainfall obtained by thiesson polygon method
        is different from book as product(A*P) is round
        offed in book.
```
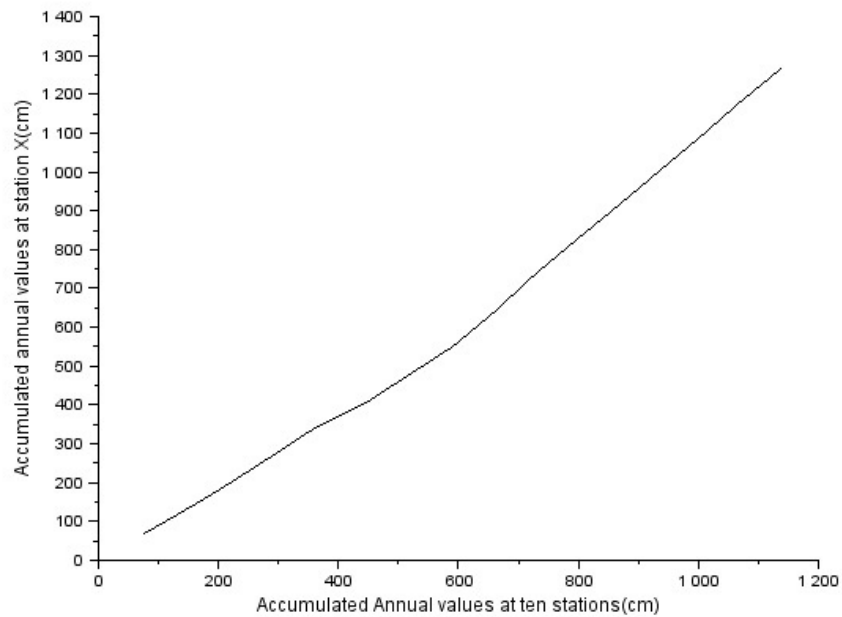
Figure 4.1: EX4 6

Scilab code Exa 4.6 EX4 6

```
1
2
3 //example 4.6
4 //ceck whether data at station X is consistence
5 //year in which regime is indicated
6 //compute the adjusted rainfall atX
7 clc;funcprot(0);
8 //given
```

```scilab
 9  X =[69 55 62 67 87 70 65 75 90 100 90 95 85 90 75
       95];    // annual rainfall at X
10  Y =[77 62 67 68 86 90 65 75 70 70 70 75 65 70 55 75];
            // average rainfall at 10 base stations
11  cx (1) =69;                      // accumulated annual values
       at station X
12  for i =2:16
13      cx (i)= cx (i -1)+ X (i);
14  end
15  cy (1)=77;
16  for i =2:16
17      cy (i)= cy (i -1)+ Y (i);   // accumulated annual values
            at ten stations
18
19  end
20
21  // since curve is not having unform slope
22  mprintf (" Record at X is not consistent .");
23  mprintf ("\ nFrom the curve regime is observed in the
       year 1978.")
24
25  Q =[1970 1971 1972 1973 1974 1975 1976 1977];
26  O =[95 75 90 85 95 90 100 90];
27  for i =1:8
28      A (i)=0.7051* O (i);
29  end
30  mprintf ("\ n \ nYear              Observed rainfall
                Adjusted rainfall");
31  for i =1:8
32      mprintf ("\ n%i                    %i
                                 %i",Q (i),O (i),A (i));
33  end
34  // graph is plotted between cx and cy
```
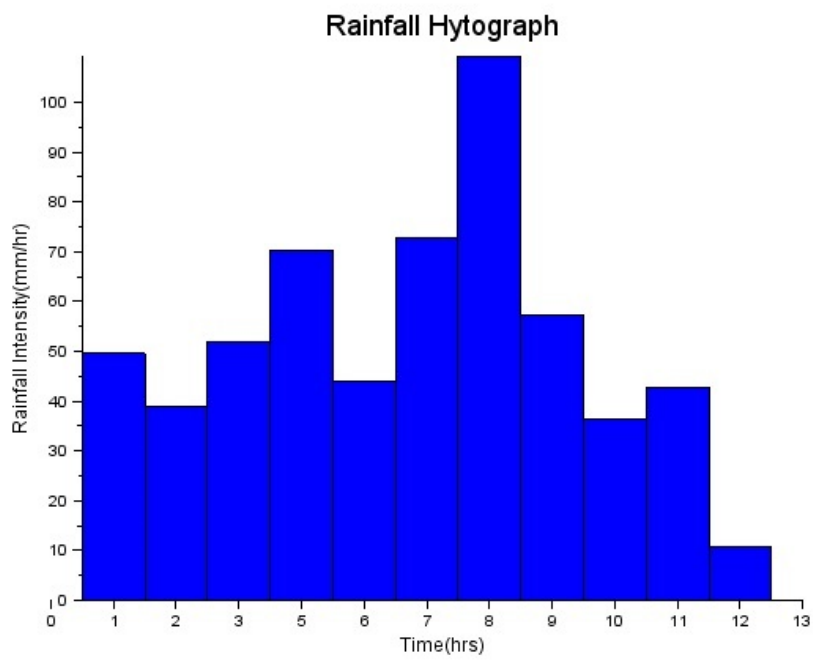
Figure 4.2: EX4 7

Scilab code Exa 4.7 EX4 7

```scilab
1
2
3  //example  4.7
4  //construct  hyetograph
5  clc;funcprot(0);
6  //given
7  c=[0  12.4  22.1  35.1  52.7  63.7  81.9  109.2  123.5  132.6
        143.3  146  146];   //cumulative  rainfall
8  T=[0:1:13];            //Time
9  t=15/60;               //time  interval
10 r(1)=0;
11 mprintf("Rainfall  intensity:");
12 I(1)=0;
13 for  i=2:13
14    r(i)=c(i)-c(i-1);
15     I(i)=r(i)/t;              //Rainfall  intensity
16 mprintf("\n%f",I(i));
17 end
18
19 //graph  is  plotted  between  I  and  T
```

Scilab code Exa 4.8 EX4 8

```scilab
1
2
3  //example  4.8
4  //compute  maximum  rainfall  intensities  for
        15,30,45,60,90,120  minutes
5  //plot  intensity  duration  graph
6  clc;funcprot(0);
7  //given
```
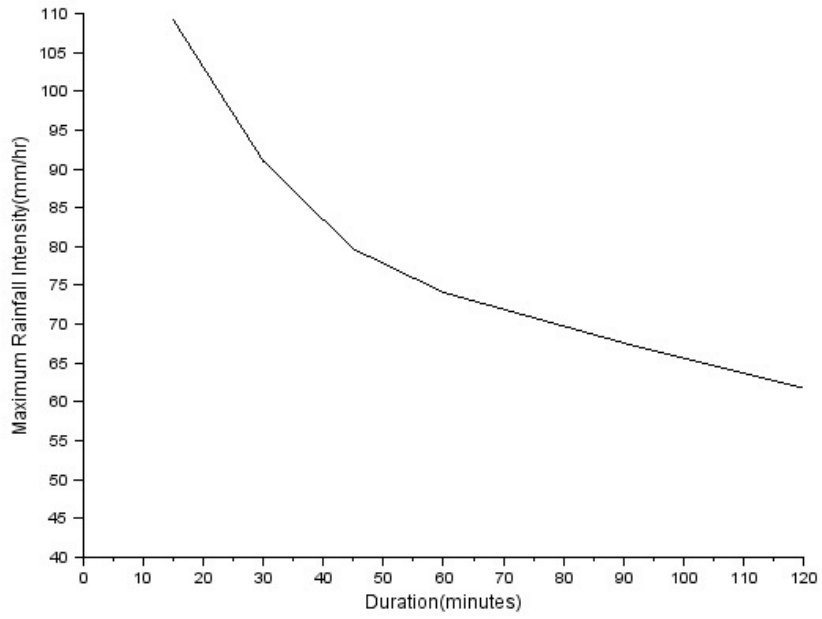
Figure 4.3: EX4 8

```scilab
 8  CR =[0 12.4 22.1 35.1 52.7 63.7 81.9 109.2 123.5
       132.6 143.3 146.0 146.0];  //cumulative rainfall
 9
10  c15(2)=12.4;
11  c30(3)=22.1;
12  c45(4)=35.1;
13  c60(5)=52.7;
14  c90(7)=81.9;
15  c120(9)=123.5;
16  for i=3:13
17      c15(i)=CR(i)-CR(i-1);
18  end
19  for i=4:13
20      c30(i)=CR(i)-CR(i-2);
21  end
22  for i=5:13
23      c45(i)=CR(i)-CR(i-3);
24  end
25  for i=6:13
26      c60(i)=CR(i)-CR(i-4);
27  end
28  for i=8:13
29      c90(i)=CR(i)-CR(i-6);
30  end
31  for i=10:13
32      c120(i)=CR(i)-CR(i-8);
33  end
34  mprintf("15min                    30min                    45
       min                  60min                90min
       120min");
35  for i=1:13
36      mprintf("\n%f              %f              %f
           %f              %f                %f",c15(i),c30(i),c45
           (i),c60(i),c90(i),c120(i));
37  end
38  I=[109.2 91 79.7 74.1 67.6 61.75];      //maximum
       intensity at respective durations
39  D=[15 30 45 60 90 120];                 //durations
```

57

```
40  //greph is plotted between I and D
```

Scilab code Exa 4.9 EX4 9

```
 1
 2
 3  //example 4.9
 4  //calculate preciptation value which has recurrence
        period of 6 years
 5  clc;funcprot(0);
 6  //given
 7  p=[475 377 731 1066 361 305 926 628 409 236 337
        853];    //precipitation value
 8  N=12;                          //total number of years
 9  T=6;                           //recurrence interval
10  m=N/T;
11  mprintf("Ranking of storm=%i.",m);
12  //hence pick 2nd severest storm
13  mprintf("\npreciptation value which has recurrence
        period of 6 years=%i mm.",p(7));
```

Scilab code Exa 4.10 EX4 10

```
 1
 2
 3  //example 4.10
 4  //calculate average depth of precipitation using
        depth area curve
 5  clc;funcprot(0);
 6  //given
 7  I=[25:-1:16];            //isohytes
```
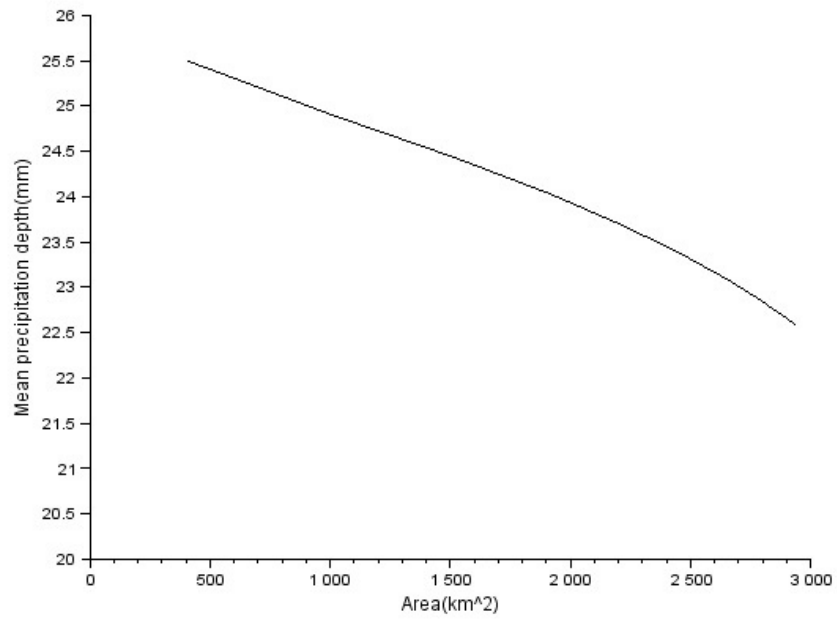
Figure 4.4: EX4 10

```
 8  a=[407 1008 1522 1909 2216 2460 2651 2782 2910
       2936];   //enclosed area
 9  ia(1)=407;
10  for i=2:10
11      ia(i)=a(i)-a(i-1);
12  end
13  r=[25.5:-1:16.5]
14  for i=1:10
15      rv(i)=r(i)*ia(i);
16  end
17  cv(1)=10378;
18  for i=2:10
19      cv(i)=cv(i-1)+rv(i);
20  end
21  for i=1:10
22      eud(i)=cv(i)/a(i);                    //mean
            precipitation
23  end
24
25  mprintf("From depth area curve we obtain average
       depth of precipitation=24.1 mm for an \narea of
       1800 sq. km.");
26  //graph is plotted between eud and a.
```

Scilab code Exa 4.11 EX4 11

```
1
2
3  //example 8.11
4  //calculate
5  //24h max. rainfall with return period of 8,15 and
       25.
6  //24h max rainfall with 40%,24% and 8% probability.
```
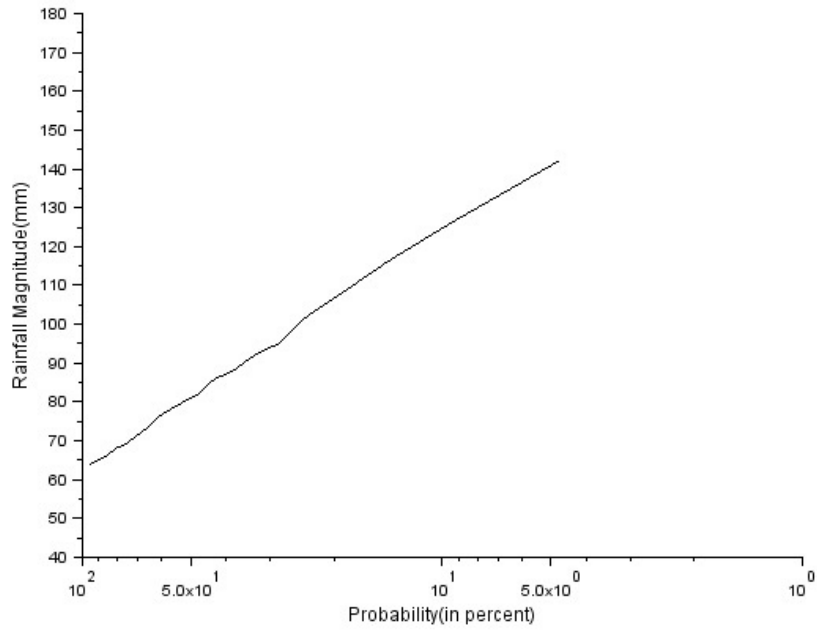
Figure 4.5: EX4 11

```scilab
 7  //probabilty of rainfall of magnitude equal to or
       exceeding 100 mm.
 8  clc;funcprot(0);
 9  //given
10  N=20;
11  r=[142 126 116 108 102 95 92 88 86 82 80 78 76 73 71
       69 68 66 65 64];    //rainfall in respective
     years
12  m=[1:1:20];                  //ranking of storm
13  for i=1:20
14      p(i)=m(i)*100/(N+1);          //probability(
           percent)
15      T(i)=100/p(i);                //recurrence
           interval
16  end
17  //from frequency curve obtained we get
18  //Part (a)
19  T1=[8 15 25];
20  r1=[119 134 149];
21  mprintf("T(years)            Rainfall(mm)");
22  for i=1:3
23      mprintf("\n%i                         %i",T1(i),
         r1(i));
24  end
25
26  //Part (b)
27  p1=[40 24 8];
28  r2=[87 101 130];
29  mprintf("\n\nprobability(percent)          Rainfall(
     mm)");
30  for i=1:3
31      mprintf("\n%i                          %i"
         ,p1(i),r2(i));
32  end
33  //graph is plotted on semi-log graph between r and p
34
35  mprintf("\n\nFor rainfall=100 m.\nT=4 years.\
     nProbability=25 percent.");
```

Scilab code Exa 4.12 EX4 12

```scilab
1
2
3  //example 4.12
4  //plot IDF curve for return period of 10,2 and 1
       years using california formula
5  clc;funcprot(0);
6  //given
7  t=[5 10 20 30 60 90 120];            //duration
8  //value of P for respective return period is
9  p10=[10.6 14.7 19.3 20.8 25.5 29 34.7];  //rainfall
       for T=10 years
10 p2=[8.2 10.3 13.2 14.2 16.6 19.4 21.4];  //rainfall
       for T=2 years
11 p1=[3.5 6.2 8.9 10 13.2 15 16.5];        //rainfall
       for T=1 year
12 for i=1:7
13     i1(i)=p10(i)*60/t(i);                    //intensity
           of rainfall with return period of 10 years
14     i2(i)=p2(i)*60/t(i);                     //intensity
           of rainfall with return period of 2 years
15     i3(i)=p1(i)*60/t(i);                     //intensity
           of rainfall with return period of 1 year
16 end
17 //graph is plotted between
18 //t and i1
19 //t and i2
20 //t and i3
```
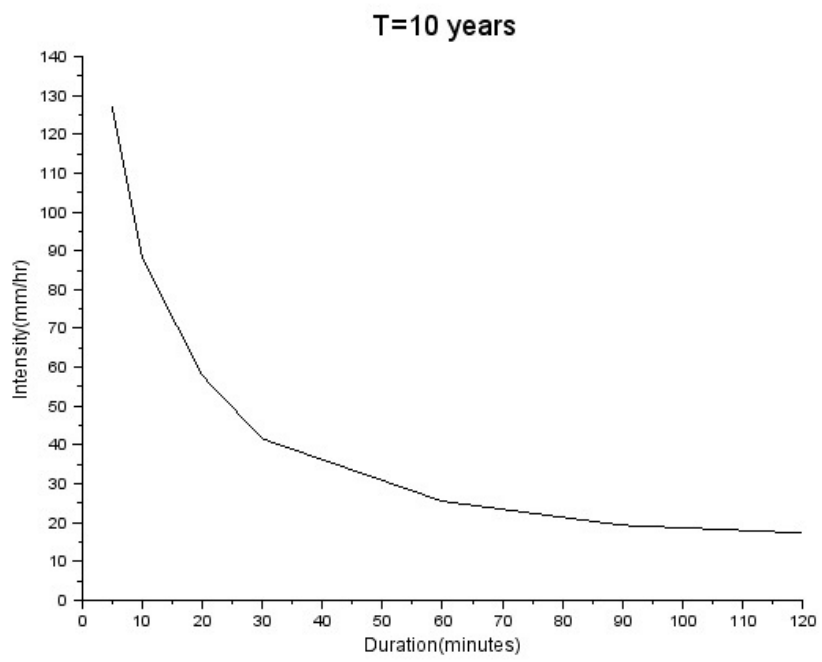
Figure 4.6: EX4 12
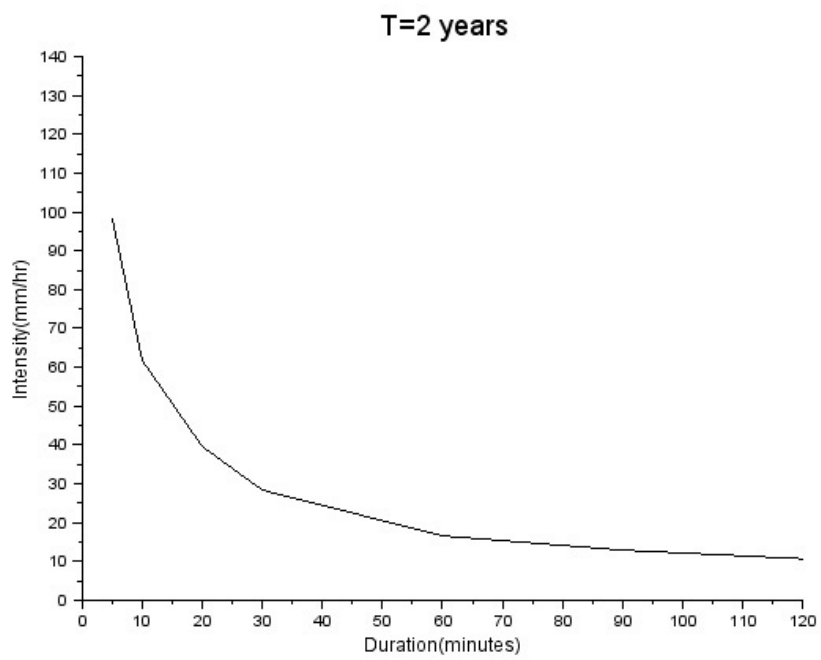
Figure 4.7: EX4 12

Scilab code Exa 4.13 EX4 13

```
1
2
3  //example  4.13
4  //calculate  maximum  and  minimum  rainfall
5  clc;funcprot(0);
6  //given
7  N=20;
8  m=[1:1:20];                //rank  number
9  rd=[82  78  75  72  70  68  65  63  61  58  56  54  52  50  46  40
       36  34  32  30];     //rainfall  in  decresing  order
10 for  i=1:20
11     ri(i)=rd(21-i);
12 end
13 for  i=1:20
14     T(i)=N/(m(i)-0.5);
15 end
16 //from  the  curves
17 mprintf("maximum  rainfall=79cm  for  T=15  years.");
18 mprintf("\nminimum  rainfall  =31  cm  for  T=15  years.")
       ;
19 //graph  is  plotted  between  rd  and  T;ri  and  T
```

Scilab code Exa 4.14 EX4 14

```
1
2
```
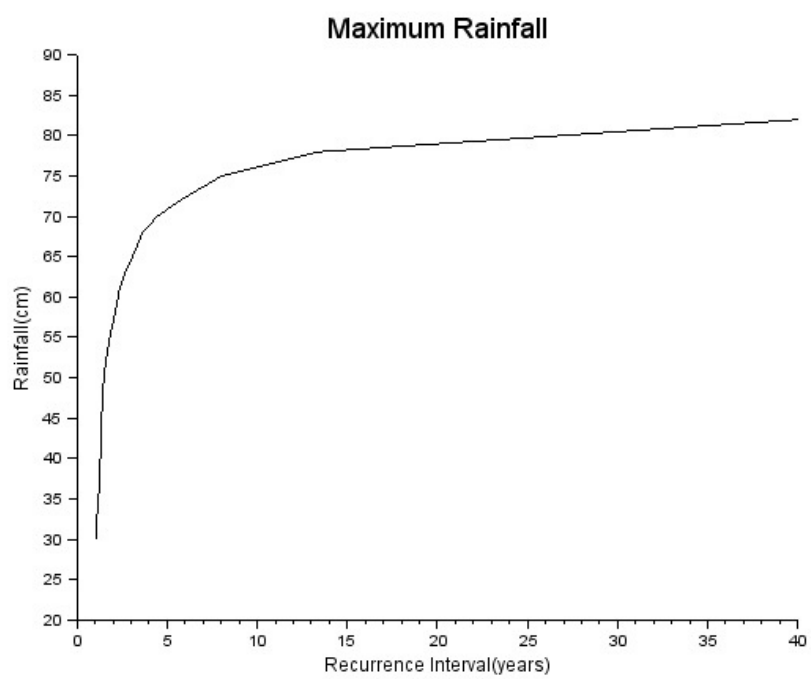
Figure 4.8: EX4 13
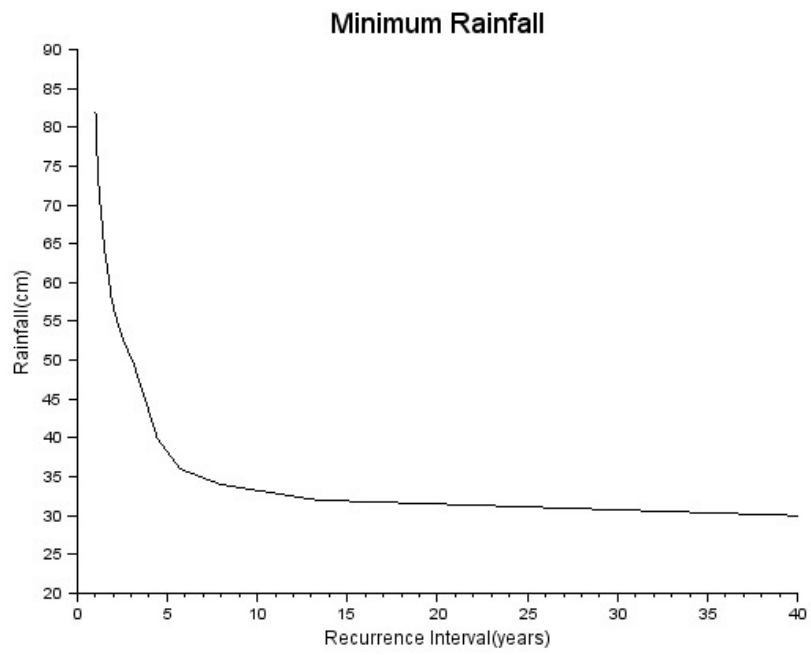
Figure 4.9: EX4 13

```
3  //example 4.14
4  //calculate daily lake evaporation
5  //average evaporation for one week
6  clc;funcprot(0);
7  //given
8  w=[12 5 2 -3 1 6 11];        //water added or taken out
9  r=[0 6 8 12 9 5 0];          //rainfall
10 for i=1:7
11     pan(i)=w(i)+r(i);        //Pan evaporation
12     le(i)=0.8*pan(i);        //lake evaporation
13 end
14
15 s=0;
16 for i=1:7
17     s=s+le(i);
18 end
19 mprintf("daily lake evaporation(mm):");
20 for i=1:7
21     mprintf("\n%f",le(i));
22 end
23 av=s/7;
24 av=round(av*100)/100;
25 mprintf("\n\naverage evaporation for one week=%f mm.
        ",av);
```

Scilab code Exa 4.15 EX4 15

```
1
2
3  //example 4.15
4  //calculate average evaporation loss from reservior
5  //total depth and volume of evaporation loss
6  clc;funcprot(0);
7  //given
8  Rh=0.4;              //relative humidity
```

```scilab
 9  A =4.8;              // average  surface  spread  of
        reservior
10  v3 =18;              // wind  velocity  at  3m above  ground
11  es =31.81;           // saturated  vapour  pressure
12  Km =0.36;            // for  large  deep  waters
13
14  // using  Meyer 's  formula
15  ea=es*Rh;
16  v9=v3*(9/3)^(1/7);
17  E=Km*(es-ea)*(1+v9/16);
18  d=7*E;
19  v=d*A*100/1000;
20  E=round(E*10)/10;
21  d=round(d*10)/10;
22  v=round(v*100)/100;
23  mprintf("using  Meyers  formula:");
24  mprintf("\naverage  evaporation  loss  from  reservior=
        %f  mm/day.",E);
25  mprintf("\ntotal  depth=%f  mm",d);
26  mprintf("\ntotal  volume=%f  hectare-m.",v);
27
28  // using  Rohwer 's  formula
29  Pa=760;
30  vdash=(0.6/2)^(1/7)*18;
31  E=0.771*(1.465-0.000732*Pa)*(0.44+0.0733*vdash)*(es-
        ea);
32  d=7*E;
33  v=d*A*100/1000;
34  E=round(E*10)/10;
35  d=round(d*10)/10;
36  v=round(v*10)/10;
37  mprintf("\n\nusing  Rohwers  formula:");
38  mprintf("\naverage  evaporation  loss  from  reservior=
        %f  mm/day.",E);
39  mprintf("\ntotal  depth=%f  mm",d);
40  mprintf("\ntotal  volume=%f  hectare-m.",v);
```
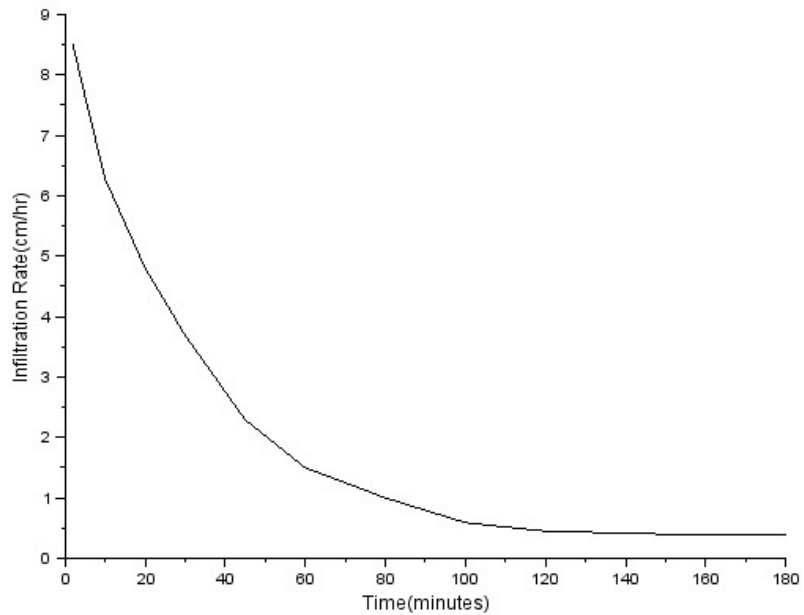
Figure 4.10: EX4 16

Scilab code Exa 4.16 EX4 16

```
1
2
3 //example 4.16
4 //plot infiltration capacity curve
5 //calculate constant rate of infiltration
6 clc;funcprot(0);
7 //given
8 D=30;            //diameter of inside ring of
```

```
      infiltrometer
 9  A=%pi*D^2/4;
10  V=[0 200 470 840 1405 1840 2245 2510 2745 2885 2990
       3130 3270];  //cumulative volume;
11  t=[0 2 5 10 20 30 45 60 80 100 120 150 180];
                         //Time(minutes)
12  dt(1)=0;
13  for i=2:13
14      dt(i)=(t(i)-t(i-1))/60;
15  end
16  for i=1:13
17      F(i)=V(i)/A;
18  end
19  Fd(1)=F(1);
20  for i=2:13
21      Fd(i)=F(i)-F(i-1);
22  end
23  for i=2:13
24      ft(i)=Fd(i)/dt(i);      //infirltration rate
25  end
26  //from the graph
27  mprintf("constant rate of infiltration=0.40 cm/hr.")
       ;
28  avg10=F(4)*60/10;
29  avg30=F(6)*60/30;
30  avg10=round(avg10*100)/100;
31  avg30=round(avg30*100)/100;
32  mprintf("\naverage rate of infiltration for first 10
       min=%f cm/hr.",avg10);
33  mprintf("\naverage rate of infiltration for first 30
       min=%f cm/hr.",avg30);
34  //graph is plotted between ft and t
```

Scilab code Exa 4.17 EX4 17

```scilab
1
2
3 //example 4.17
4 //calculate
5 //drainage desity
6 //form factor
7 //channel slope
8 //average overland flow length
9 clc;funcprot(0);
10 //given
11 A=82;           //area of watershed
12 d=12.6;         //distance between outlet and farther
        most point
13 l=440;          //total length of channel
14 e=656;          //elevation differnce between outlet
      and further most point
15
16 Dd=l/A;
17 ff=A/d^2;
18 cs=e/(d*1000);
19 lo=1000/(2*Dd);
20 Dd=round(Dd*100)/100;
21 ff=round(ff*1000)/1000;
22 mprintf("drainage desity=%f km/square.km.",Dd);
23 mprintf("\nform factor=%f.",ff);
24 mprintf("\nchannel slope=%f.",cs);
25 mprintf("\naverage overland flow length=%i m.",lo);
```

Scilab code Exa 4.18 EX4 18

```scilab
1
2
3 //example 4.18
4 //compute fi and W index
5 clc;funcprot(0);
```

```
 6  // given
 7  R =3.6;              // surface runoff
 8  r =[0  1.3  2.8  4.1  3.9  2.8  2.0  1.8  0.9];    // rainfall
        at  respective  time
 9  t =4;                    // total  time
10  s =0;
11  for  i =3:8
12      s = s + r ( i )
13  end
14  fi =( s - R *2)/6;
15  // since  fi  >1.3  and  <1.8
16  mprintf (" fi  index=%f  cm.",fi);
17  mprintf ("\ncomputations  are  correct.");
18
19  s =0;
20  for  i =1:9
21      s = s + r ( i );
22  end
23  P = s /2;
24  Sr =0;
25  W =( P - R - Sr )/ t;
26  mprintf ("\nW  index=%f  cm/hr.",W);
```

Scilab code Exa 4.19 EX4 19

```
 1
 2
 3  // example4 .19
 4  // calculate  fi  index  and  time  of  rainfall  excess
 5  clc ; funcprot (0);
 6  // given
 7  T =[1:1:9];        // time  from  start
 8  r =[0.7  1.4  2.4  3.7  2.9  2.6  1.7  0.8  0.5];    //
        increamental  rainfall
 9  R =9.3;            // total  run−off
```

```scilab
10  s=0;
11  for i=1:9
12      s=s+r(i);
13  end
14  ti=s-R;
15
16  //first trial
17  tr=9;    //assumed
18  fi1=ti/tr;
19  //this makes 1st,8th and 9th hour ineffective
20
21  //second trial
22  tr=6;
23  ti=s-R-r(1)-r(8)-r(9);
24  fi=ti/tr;
25  for i=1:9
26      P(i)=r(i)-fi;
27      if (P(i)<0) then
28          P(i)=0;
29      end
30  end
31  mprintf("Time(h)           rainfall excess.");
32  for i=1:9
33      mprintf("\n%f            %f",T(i),P(i));
34  end
35  mprintf("\n\nfi index=%f cm/hr.",fi);
36  mprintf("\n\ntime of rainfall excess=%i hours..",tr)
       ;
```

Scilab code Exa 4.20 EX4 20

```scilab
1
2
3  //example 4.20
4  //calculate relation between R and P
```

```
5  clc;funcprot(0);
6  //given
7  P=[72.2 70.1 73.3 42.5 81.3 50.6 52.9 59.4 60.3 64.3
        68.8 56.7 77.2 40.5 44.1 65.5];  //Precipitation
8  R=[24.1 22.7 25.6 11.3 28.4 12.7 13.4 15.7 16.2 17.7
        19.2 14.9 25.4 10.6 11.7 17.9];  //runoff
9  for i=1:16
10     Ps(i)=P(i)^2;
11     Rs(i)=R(i)^2;
12     PR(i)=P(i)*R(i);
13 end
14
15 s=0;t=0;u=0;q=0;w=0;
16 for i=1:16
17     s=s+Ps(i);
18     t=t+Rs(i);
19     u=u+PR(i);
20     q=q+P(i);
21     w=w+R(i);
22 end
23 N=16;
24 a=(N*u-q*w)/(N*s-q^2);
25 b=(w-a*q)/N;
26 b=round(b*1000)/1000;
27 a=round(a*10000)/10000;
28 mprintf("Equation is:\n%fP%f.",a,b);
```

Scilab code Exa 4.21 EX4 21

```
1
2
3  //example4.21
4  //calculateeffective  rainfall  hyetograph  and  volume
```
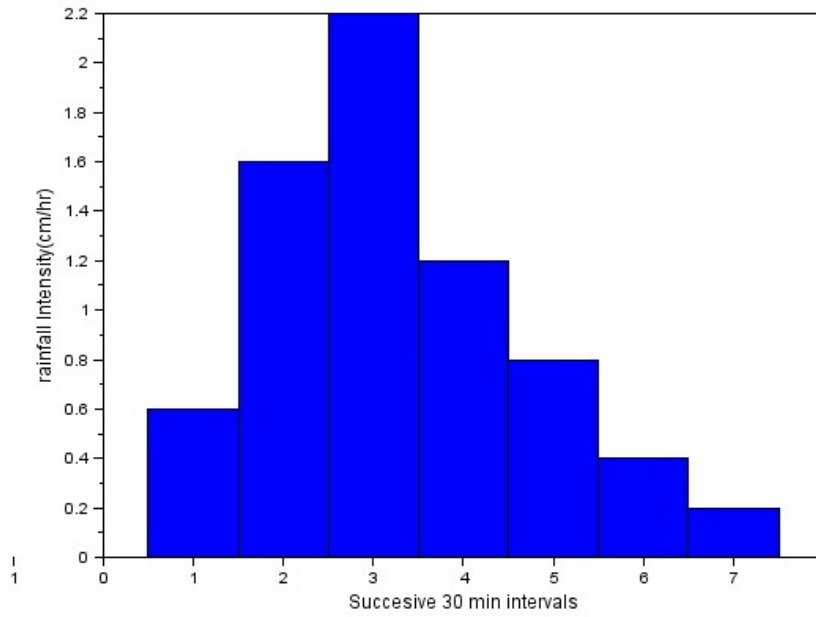
Figure 4.11: EX4 21

```scilab
      of diret run-off
 5  clc;funcprot(0);
 6  //given
 7  A=8.6;                                      //
        catchment area
 8  T=[0:0.5:4];                                //time
 9  r=[0 0.4 1.1 2.3 3.8 4.8 5.6 6.2 6.7];      //
        accumulated rainfall
10  fi=0.4;                                     //fi index
11  dt=0.5;                                     //time
        interval
12  d(1)=0;
13  for i=2:9
14      d(i)=r(i)-r(i-1);                       //accumulated
            rainfall
15  end
16  mprintf("Intensity of effective Rainfall:");
17  I(1)=0;
18  s=0;
19  for i=2:9
20      p(i)=d(i)-fi;                           //effective
            rainfall
21      I(i)=p(i)/dt;                           //Intensity of
            effective Rainfall
22      s=s+I(i);
23      mprintf("\n%f",I(i));
24  end
25  //graph is plotted between I and T
26  run=s*dt;
27  V=run*A*10000;
28  mprintf("\nVolume of direct run-off=%f cubic metre."
        ,V);
```
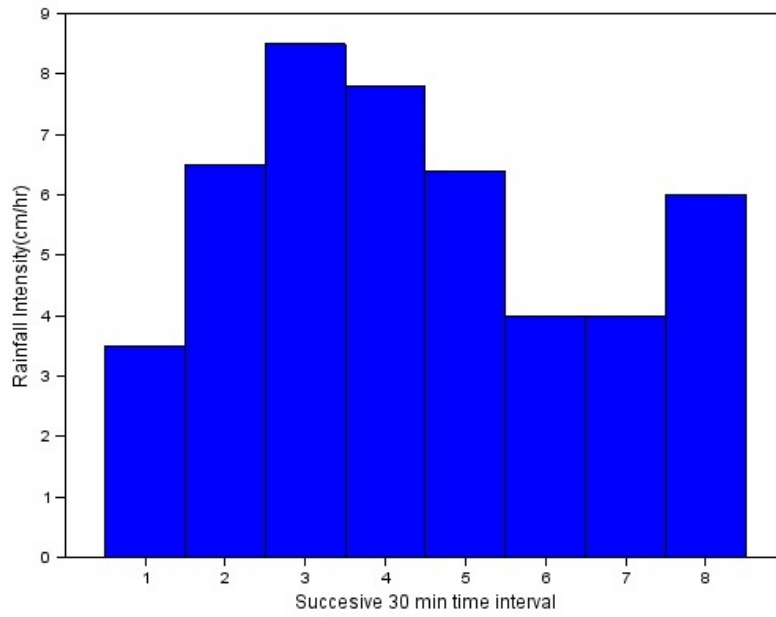
Figure 4.12: EX4 22

Scilab code Exa 4.22 EX4 22

```
1
2
3  //example 4.22
4  //calculate
5  //total rainfall
6  //total rainfall excess
7  //W index
8  clc;funcprot(0);
9  //given
10 r=[3.5 6.5 8.5 7.8 6.4 4 4 6];          //rainfall
        intensity
11 T=[0:30:240];                            //time
12 dt=30;                                   //time interval
13 //graph is plotted between r and T
14 s=0;
15 for i=1:8
16     s=s+r(i);
17 end
18 P=s*dt/60;
19 Pe=((6.5-4.5)+(8.5-4.5)+(7.8-4.5)+(6.4-4.5)+(6-4.5))
        *dt/60;       //area of graph above r=4.5.
20 w=(P-Pe)/4;
21 mprintf("total rainfall=%f cm.",P);
22 mprintf("\ntotal rainfall excess=%f cm.",Pe);
23 mprintf("\nW index=%f cm/hr.",w);
```

Scilab code Exa 4.23 EX4 23

```
1
2
3  //example 4.23
4  //calculate fi index
5  clc;funcprot(0);
```

```
 6  //given
 7  r=[0 8 22 74 92 105 114 120];        //raccumulated
        rainfall
 8  T=[0 2 4 6 8 10 12 14];              //time for start
        of rainfall
 9  V=2D6;                               //volume of run−
        off
10  A=40;                                //catchment area
11  tr=14;                               //duration of
        rainfall
12
13  d=V*1000/(40*1000000);
14
15  l=r(8)-d;
16  W=l/tr;
17  for i=2:8
18      I(i)=r(i)-r(i-1);               //incremental
            rainfall
19  end
20
21  //rainfall excess is available in 4 time intervals
        of 2 hrs
22  tre=8;
23  fi=(l-I(2)-I(7)-I(8))/tre;
24  fi=round(fi*100)/100;
25  mprintf("fi index=%f mm/hr.",fi);
```

Scilab code Exa 4.24 EX4 24

```
1
2
3  //example 4.24
4  //calculate average infiltration index
5  clc;funcprot(0);
6  //given
```

```
7  r=[2.0 2.5 7.6 3.8 10.6 5.0 7.0 10.0 6.4 3.8 1.4
       1.4];        //rainfall depths
8  R=25.5;

       //total rum-off
9  s=0;
10 for i=1:12
11     s=s+r(i);
12 end
13 tf=s-R;
14 af=tf/12;
15 //rainfall is less than average infiltration in1st,2
       nd,11th and 12th hours
16
17 f=(tf-r(1)-r(2)-r(11)-r(12))/8;
18 f=round(f*10)/10;
19 mprintf("average infiltration index=%f cm/hour.",f);
```

Scilab code Exa 4.25 EX4 25

```
1
2
3  //example 4.25
4  //calculate average depth of hourly rainfall excess
5  clc;funcprot(0);
6  //given
7  r=[2.0 2.5 7.6 3.8 10.6 5.0 7.0 10.0 6.4 3.8 1.4
       1.4];        //rainfall depths
8  A1=20;
9  A2=40;
10 A3=60;
11 A=A1+A2+A3;
12 fi1=7.6;
13 fi2=3.8;
14 fi3=1.0;
```

```
15  for i=1:12
16      R1(i)=r(i)-fi1;                    //rainfall excess
17      R2(i)=r(i)-fi2;
18      R3(i)=r(i)-fi3;
19      if (R1(i)<0)   then
20      R1(i)=0;
21  end
22  if (R2(i)<0)   then
23      R2(i)=0;
24  end
25  if (R3(i)<0)   then
26      R3(i)=0;
27  end
28  end
29  mprintf("average depth of hourly rainfall excess(cm/
        hr)");
30  for i=1:12
31      a1(i)=R1(i)*A1/A;                      //average
            rainfall excess
32      a2(i)=R2(i)*A2/A;
33  a3(i)=R3(i)*A3/A;
34  T(i)=a1(i)+a2(i)+a3(i);                 //total hourly
        rainfall excess
35  T(i)=round(T(i)*100)/100;
36  mprintf("\n%f",T(i));
37  end
```

Scilab code Exa 4.26 EX4 26

```
1
2
3  //example4.26
4  //derive the unit hydrograph
5  clc;funcprot(0);
6  //given
```

```
7  A=92;                              //area of drainage basin
8  t=[6 8 10 12 14 16 18 20 22 24 2 4 6 8 10 12 14 16];
         //time
9  r=[10.6 9.7 107.8 175.6 193.9 150.3 126.2 106.9 90
      72.8 58.2 48 36.2 28.4 20.2 14 10.2 10.4];   //
      total run-off
10 B=[10.6 9.7 9.73 9.77 9.8 9.83 9.87 9.9 9.93 9.97 10
       10.03 10.07 10.10 10.13 10.16 10.20 10.40];   //
      base flow
11 s=0;
12 for i=1:18
13     d(i)=r(i)-B(i);                   //direct run-off
          ordinate
14 s=s+d(i);
15 end
16 n=0.36*s*2/A;
17 mprintf("ordinates of unit hydrograph:");
18 for i=1:18
19     u(i)=d(i)/n;                      //ordinates of
          unit hydrograph
20     u(i)=round(u(i)*100)/100;
21      mprintf("\n%f",u(i));
22 end
23 mprintf("\nHydograph is 4-hr unit hydrograph");
24 //graph is plotted between:
25 //r and t
26 //u and t
```

Scilab code Exa 4.27 EX4 27

1
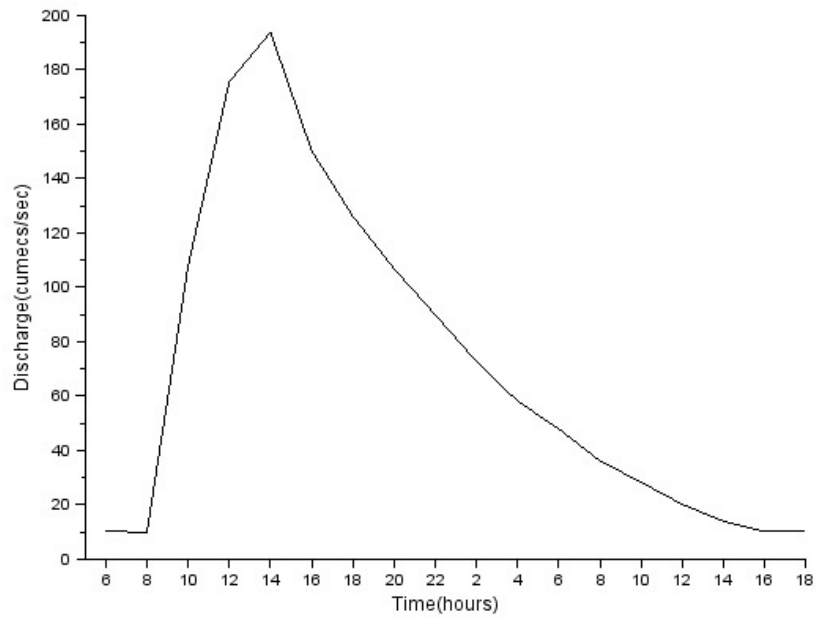
Figure 4.13: EX4 26
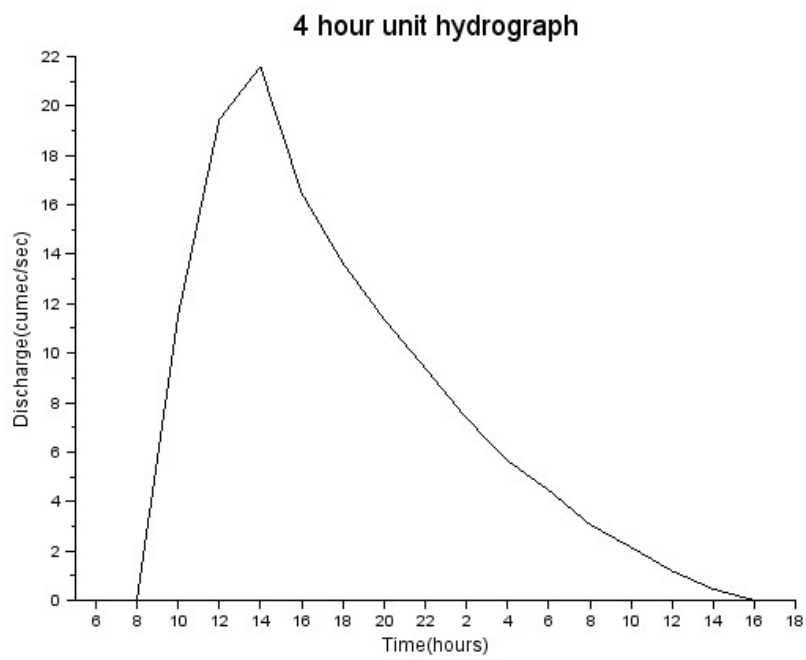
Figure 4.14: EX4 26

```
2
3  //example 4.27
4  //calculate rainfall excess
5  clc;funcprot(0);
6  //given
7  A=316;           //drainage area
8  B=17;            //base flow
9  t=6;
10 O=[17.0 113.2 254.5 198.0 150.0 113.2 87.7 67.9 53.8
        42.5 31.1 22.6 17.0];     //ordinates of storm
      hydrograph
11 for i=1:13
12     Or(i)=O(i)-B;                    //ordinates of
          direct run−off
13     Oh(i)=Or(i)/6.477;               //ordinates of
          unit hydrograph
14 end
15 s=0;
16 for i=1:13
17     s=s+Or(i);
18 end
19 re=s*60*60*t/(A*10000);
20 re=round(re*1000)/1000;
21 mprintf("rainfall excess=%f cm.",re);
```

Scilab code Exa 4.28 EX4 28

```
1
2
3  //example 4.28
4  //calculate ordinates of storm hydrograph
5  clc;funcprot(0);
6  //given
7  fi=2.5;          //infiltration index
8  B=10;            //Base flow
```

```
 9  O = [0  110  365  500  390  310  250  235  175  130  95  65  40  22
        10  0  0  0];    //ordinates  of  unit  hydrograph
10  R1 =2; R2 =6.75; R3 =3.75;
11  r1 =( R1 *10 -( fi *3) -5) /10;                 // rainfall
        excess  in  first  three  hour
12  r2 =( R2 *10 -( fi *3) ) /10;                 // rainfall
        excess  in  second  three  hour
13  r3 =( R3 *10 -( fi *3) ) /10;                 // rainfall
        excess  in  third  three  hour
14
15  for  i =1:18
16      s1 ( i )= r1 * O ( i );
17  end
18  for  i =2:18
19      s2 ( i )= r2 * O (i -1) ;
20  end
21  for  i =3:18
22      s3 ( i )= r3 * O (i -2) ;
23  end                                       // surface  run −
        off  from  rainfall  excess  during  succesive  unit
        periods
24  mprintf (" ordinates  of  storm  hydrograph ");
25  for  i =1:18
26      T ( i )= s1 ( i )+ s2 ( i )+ s3 ( i );
27      t ( i )= T ( i )+ B ;
28      t ( i )= round ( t ( i ) *10) /10;
29      mprintf (" \n%f " , t ( i ) );
30  end
```

Scilab code Exa 4.29 EX4 29

```
1
2
3  //example4.29
4  //derive  and  plot  6  hr  unit  hydrograph
```

```
5  clc; funcprot (0) ;
6  //given
7  A=103.4;         //area  of  basin
8  t =[0:3:36];     //time
9  q=[0  21  80  82  189  123  184  87  55.5  25.25  9  6  0];   //
       flow
10  mprintf ("ordinates  of  unit  hydrograph  are :") ;
11  u (1) =0;
12  u (2) =q (2) /2;
13  u (3) =( q (3) -4*u (1) )/2;
14  u (4) =( q (4) -4*u (2) )/2;
15  for  i =5:9
16      u(i) =( q(i) -3*u(i -4) -4*u(i -2) )/2;        //
            ordinates  of  unit  hydrograph
17  end
18  for  i =1:9
19      mprintf ("\n%f",u(i)) ;
20  end
21  mprintf ("\n\nThe  succesive  unit  hydrograph  will  have
        same  ordinates  but  will  be  shifted\nlaterally  by
        6  hrs .") ;
22  //graph  is  plotted  between  u  and  t .
```

Scilab code Exa 4.30 EX4 30

```
1
2
3  //example  4.30
4  //derive  ordinates  of  6  hrs  unit  hydrograph
5  clc; funcprot (0) ;
6  //given
```
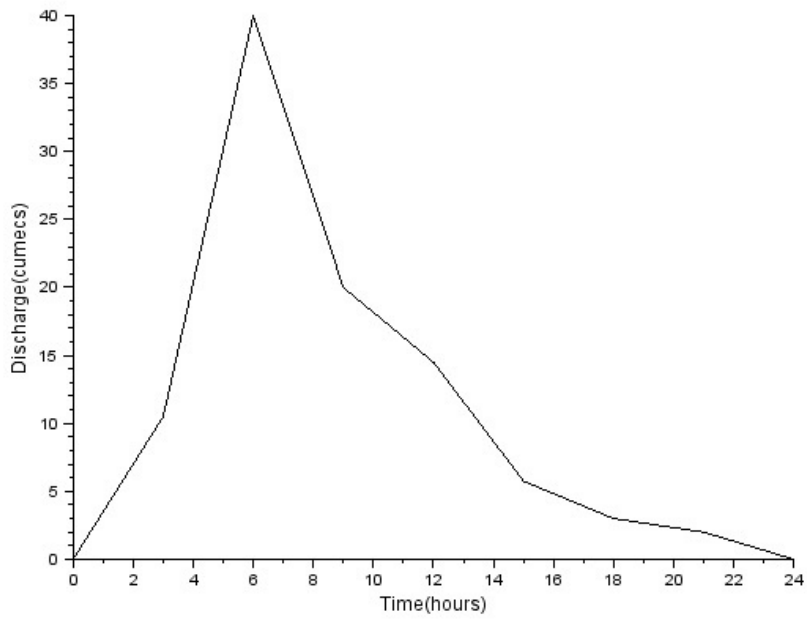
Figure 4.15: EX4 29
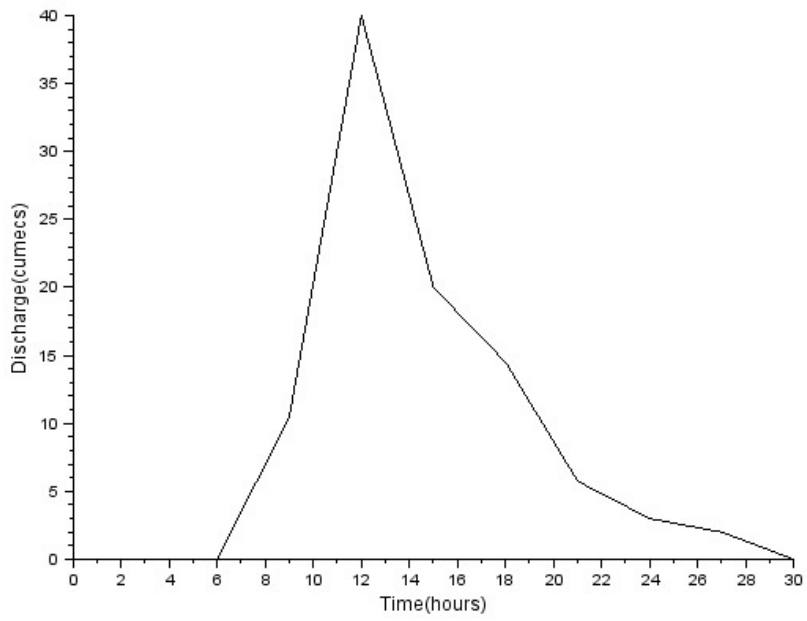
Figure 4.16: EX4 29

```
7  R=[0 1 2.7 5 8 9.8 9 7.5 6.3 5 4 2.9 2.1 1.3 0.5 0 0
       0 0 0];   //2hrs unit hydrograph
8  mprintf("ordinates of 6 hrs unit hydrograph");
9  for i=1:18
10     O1(i+2)=R(i);
11
12
13 end
14 for i=1:16
15     O2(i+4)=R(i);
16 end                                    //offset unit
      hydrograph
17 for i=1:20
18     S(i)=O1(i)+O2(i)+R(i);             //sum
19     f(i)=S(i)/3;                       //ordinates of 6
          hrs unit hydrograph
20     f(i)=round(f(i)*10)/10;
21     mprintf("\n%f",f(i));
22 end
```

Scilab code Exa 4.31 EX4 31

```
1
2
3  //example 4.31
4  //calculate ordinate of 9 hr unit hydrograph
5  clc;funcprot(0);
6  //given
7  t=[0:3:45];       //time
8  O=[0 9 20 35 49 43 35 28 22 17 12 9 6 3 0 0];
                //ordinate of 2 hr unit hydrograph
9  of(1)=0;
10 of(2)=0;
11 for i=3:16
12     of(i)=O(i-2)+of(i-2);                   //offset
```

```
                 ordinate
13  end
14  for  i =1:16
15     s(i)=O(i)+of(i);                               // ordinate
             of s−curve
16  end
17  of1(3)=0;
18  for  i =4:16
19      of1(i)=s(i−3);                                // offset
             of s−curve
20  end
21  mprintf("ordinates  of  9  hrs  unit  hydrograph:");
22  for  i =1:16
23      y(i)=s(i)-of1(i);
24      u(i)=2*y(i)/3;                                //
             ordinate  of  9  hrs  unit  hydrograph
25      u(i)=round(u(i)*10)/10;
26      mprintf("\n%f",u(i));
27  end
```

Scilab code Exa 4.32 EX4 32

```
 1
 2
 3  //example  4.32
 4  //calculate  recurrence  interval  of  flood  using
 5  //california  method
 6  //Hazens  method
 7  //gumbels  method
 8  clc;funcprot(0);
 9  //given
10  q=[9200  7800  6600  5800  5260  4980  4525  3810  3630  3250
        3110  3090  2380  2390  1723];  //Discharge  arranged
       in  decreasing  order
11  N=15;
```

```
12  C=0.3;
13  m=[1:1:15];
14  C=[0.3 0.44 0.52 0.57 0.61 0.66 0.7 0.74 0.78 0.82
        0.86 0.88 0.94 0.96 1];    //from table 4.25
15  mprintf("California          Hazen          Gumbel")
        ;
16  for i=1:15
17      Ca(i)=N/m(i);
18      H(i)=2*N/(2*m(i)-1);
19      G(i)=N/(m(i)+C(i)-1);
20      Ca(i)=round(Ca(i)*100)/100;
21      G(i)=round(G(i)*100)/100;
22      H(i)=round(H(i)*100)/100;
23      mprintf("\n%f          %f          %f",Ca(i),H(i
            ),G(i));
24  end
```

Scilab code Exa 4.33 EX4 33

```
1
2
3  //example 4.33
4  //calculate flood magnitude with return period of
        240 years
5  clc;funcprot(0);
6  //given
7  T1=40;T2=80;              //Return period
8  F1=27000;F2=31000;        //Peak flood
9  y80=-(2.303*log10(2.303*log10(T2/(T2-1))));
10  y40=-(2.303*log10(2.303*log10(T1/(T1-1))));
11  y=(F2-F1)/(y80-y40);
12  T=240;
13  y240=-(2.303*log10(2.303*log10(T/(T-1))));
14  x240=F2+(y240-y80)*y;
15  mprintf("flood magnitude with return period of 240
```

```
      years=%i cumec.",x240);
```

Scilab code Exa 4.34 EX4 34

```
1
2
3  //example 4.34
4  //calculate flood discharge with recurrence period
       of 100 years and 200 years
5  clc;funcprot(0);
6  //given
7  N=40;
8  Sn=1.1413;yn=0.5436;              //from table 4.21 (a)
       and(b)
9  q=[1330 1095 1030 980 975 950 945 940 925 855 853
       840 835 825 810 795 756 710 708 705 700 670 625
       620 610 605 595 585 570 550 530 505 500 495 485
       465 460 420 390 380];  //discharge
10 s=0;
11 for i=1:40
12     s=s+q(i);
13     end
14 xavg=s/N;
15 w=0;
16 for i=1:40
17     t(i)=(q(i)-xavg)^2;
18     w=w+t(i);
19 end
20 sigma=(w/(N-1))^0.5;
21 N=10;
22 y10=-(2.303*log10(2.303*log10(N/(N-1))));
23 K10=(y10-yn)/Sn;
24 x10=xavg+K10*sigma;
25 N=20;
26 y20=-(2.303*log10(2.303*log10(N/(N-1))));
```

95

```
27  K20=(y20-yn)/Sn;
28  x20=xavg+K20*sigma;
29  N=5;
30  y5=-(2.303*log10(2.303*log10(N/(N-1))));
31  K5=(y5-yn)/Sn;
32  x5=xavg+K5*sigma;
33
34  T=100;
35  y100=-(2.303*log10(2.303*log10(T/(T-1))));
36  K100=(y100-yn)/Sn;
37  x100=xavg+K100*sigma;
38
39  T=200;
40  y200=-(2.303*log10(2.303*log10(T/(T-1))));
41  K200=(y200-yn)/Sn;
42  x200=xavg+K200*sigma;
43  x100=round(x100);
44  mprintf("For T=100 years:\nflood discharge=%f cumecs
        .\n\nFor T=200 years:\nflood discharge=%i cumecs.
        ",x100,x200);
```

Scilab code Exa 4.35 EX4 35

```
1
2
3  //example 4.35
4  //calculate 200 year flood for stream using gumbel's
        method
5  clc;funcprot(0);
6  //given
7  sigma=1.1413;          //standard deviation
8  yn=0.5436;
9  T=50;
10 y50=-2.303*log10(2.303*log10(T/(T-1)));
11 K50=(y50-yn)/sigma;
```

96

```
12  T =100;
13  y100 = -2.303* log10 (2.303* log10 (T /(T -1)));
14  K100 =( y100 - yn )/ sigma ;
15  x50 =878;   x100 =970;      // given  peak  flood
16  A =[ K50  1; K100  1];
17  B =[ x50 ; x100 ];
18  C = A \B ;
19  xavg = C (2);
20  sigmad = C (1);
21  T =200;
22  y200 = -2.303* log10 (2.303* log10 (T /(T -1)));
23  K200 =( y200 - yn )/ sigma ;
24  x200 = xavg + K200 * sigmad ;
25  x200 = round ( x200 );
26  mprintf ("200  year  flood  for  stream=%f  cumecs .", x200 )
        ;
```

Scilab code Exa 4.36 EX4 36

```
1
2
3   // example  4.36
4   // calculate
5   // risk  of  failure  of  cofferdam
6   // return  period
7   clc ; funcprot (0);
8   // given
9   T =30;               // deign  for  period
10  n =6;                // period  of  construction
11  R =(1 -(1 -(1/ T ))^n )*100;
12  R1 =0.1;             // reduced  risk
13  T1 =1/(1 -(1 - R1 )^(1/6));
14  R = round ( R *10)/10;
15  T1 = round ( T1 *100)/100;
16  mprintf (" risk  of  failure  of  cofferdam=%f  percent .", R
```

```
        ) ;
17  mprintf ( " \ nreturn  period=%f  years . " , T1 ) ;
```

Scilab code Exa 4.37 EX4 37

```
 1
 2
 3  // example  4.37
 4  // calculate
 5  // probability  of  excedence
 6  // probability  of  flood  magnitude  occuring  at :
 7  // at  least  once  in  10  years
 8  // two  times  in  10  succesive  years
 9  // once  in  10  succesive  years
10  clc ; funcprot (0) ;
11  // given
12  T=40;                    // return  period
13  P=1/T;
14  n=10;
15  Rsk=1-(1-P)^n;
16  s=1;t=1;
17  for  i =1:n
18      s=s*i;
19  end
20  for  i =1:(n-2)
21      t=t*i;
22  end
23  P2n=s*P^2*(1-P)^8/(t*2);
24  P1n=n*P*(1-P)^(n-1);
25  Rsk=round(Rsk*1000)/1000;
26  P2n=round(P2n*10000)/10000;
27  P1n=round(P1n*1000)/1000;
28  mprintf ( " probability  of  excedence=%f . " ,P ) ;
29  mprintf ( " \ nprobability  of  flood  magnitude  occuring
        at  least  once  in  10  years=%f " ,Rsk ) ;
```

```
30  mprintf("\nprobability of flood magnitude occuring
        at two times in 10 succesive years=%f",P2n);
31  mprintf("\nprobability of flood magnitude occuring
        at once in 10 succesive years=%f",P1n);
```

Scilab code Exa 4.38 EX4 38

```
1
2
3  //example 4.38
4  //calculate peak rate of run off
5  clc;funcprot(0);
6  //given
7  C1=0.22;C2=0.12;C3=0.32;      //run-off coefficient
8  A1=3.2;A2=4.8;A3=1.8;         //calculated area
9  L=2.4;                        //length of water course
10  H=30;                         //fall
11  T=30;                         //frequency
12  t=60*0.000323*(L*1000)^0.77*(H/(L*1000))^(-0.385);
13  i=78*T^0.22/(t+12)^0.45;
14  q=2.778*i*(C1*A1+C2*A2+C3*A3);
15  q=round(q*10)/10;
16  mprintf("peak rate of run off=%f cumecs.",q);
```

Scilab code Exa 4.39 EX4 39

```
1
2
3  //example 4.39
4  //calculate peak flow rate
5  clc;funcprot(0);
6  //given
7  T=30;              //return period
```

```
8  A=2.4;                 //area of watershed
9  s=1/200;               //slope oof catchment
10 L=1.8;                 //length of travel of water
11 C=0.25;                //average run−off coefficient
12 r=[2.5 3.8 4.8 5.9 6.7 7.4 8.4 8.7 9.2];    //
      rsinfall depth
13 t=60*0.000323*(L*1000)^0.77*(s)^(-0.385);
14 rmax=r(7)+(r(8)-r(7))*7.84/10;
15 i=rmax*60/t;
16 q=2.778*C*A*i;
17 q=round(q*100)/100;
18 mprintf("peak flow rate=%f cumecs.",q);
```

Scilab code Exa 4.40 EX4 40

```
1
2
3  //example 4.40
4  //calculate precipitation at x
5  clc;funcprot(0);
6  //given
7  pA=75;          //precipitation at A
8  pB=58;          //precpitation at B
9  pC=47;          //precipitation at C
10 nA=826;         //normal precipitation at A
11 nB=618;         //normal precipitation at B
12 nC=482;         //normal precipitation at C
13 nX=757;         //normal precipitation at X
14
15 pX=(nX*pA/nA+nX*pB/nB+nX*pC/nC)/3;
16 pX=round(pX*10)/10;
17 mprintf("precipitation at x=%f cm.",pX);
```

Scilab code Exa 4.41 EX4 41

```
1
2
3 //example 4.1
4 //calculate mean rainfall;additional guages needed
5 clc;funcprot(0);
6 //given
7 p=[41 51 32 55 50 68];    //rain guage readings at
       respective stations
8 s=0;
9 for i=1:6
10     s=s+p(i);
11 end
12 pavg=s/6;
13 u=0;
14 for i=1:6
15     u=u+(p(i)-pavg)^2;
16 end
17 sx=(u/5)^0.5;
18 Cv=sx*100/pavg;
19 N=(Cv/8)^2;
20 N=round(N*100)/100;
21 mprintf("mean rainfall=%f cm.",pavg);
22 mprintf("\ntotal stations needed=%f.",N);
```

Scilab code Exa 4.42 EX4 42

```
1
2
3 //example 4.42
4 //calculate mean precipitaion using thiesson polygon
       method
5 clc;funcprot(0);
6 //given
```

```
 7  a=4;                        //dimension of plot sides
 8  P1=4.8;P2=13;P3=8;P4=5.4;P5=3.2;P6=9.4;        //
        precipitaion at respective stations
 9  A1=a^2/8+a^2/(4*1.73);
10  A2=a^2/8;
11  A3=A2;A4=A1;
12  A5=a^2/(4*1.73);
13  A6=a^2/2;
14  A=A1+A2+A3+A4+A5+A6;
15  Pavg=(P1*A1+P2*A2+P3*A3+P4*A4+P5*A5+P6*A6)/A;
16  mprintf("Mean precipitaion=%f cm.",Pavg);
```

Scilab code Exa 4.43 EX4 43

```
 1
 2
 3  //example 4.43
 4  //calculate average depth of precipitation
 5  clc;funcprot(0);
 6  //given
 7  A=[90 140 125 140 85 40 20];   //area of isohytes
 8  I=[13:-2:1];                      //average isohytel
        interval
 9  s=0;t=0;
10  for i=1:7
11      s=s+A(i)*I(i);
12      t=t+A(i);
13  end
14  Pavg=s/t;
15  Pavg=round(Pavg*10)/10;
16  mprintf(" average depth of precipitation=%f cm.",
        Pavg);
```

Scilab code Exa 4.44 EX4 44

```
1
2
3 //example 4.44
4 //calculate mean rainfall;additional guages needed
5 clc;funcprot(0);
6 //given
7 p=[120 95 96 60 65 70 45 21];    //rain guage
      readings at respective stations
8 s=0;
9 for i=1:8
10     s=s+p(i);
11 end
12 pavg=s/8;
13 u=0;
14 for i=1:8
15     u=u+(p(i)-pavg)^2;
16 end
17 sx=(u/7)^0.5;
18 Cv=sx*100/pavg;
19 N=(Cv/13.99)^2;
20 N=round(N*100)/100;
21 mprintf("mean rainfall=%f cm.",pavg);
22 mprintf("\ntotal stations needed=%f.",N);
23 //taking N=10
24 N=10;
25 n=N-8;
26 mprintf("\nadditional guages needed=%i.",n);
```

Scilab code Exa 4.45 EX4 45

```
1
```
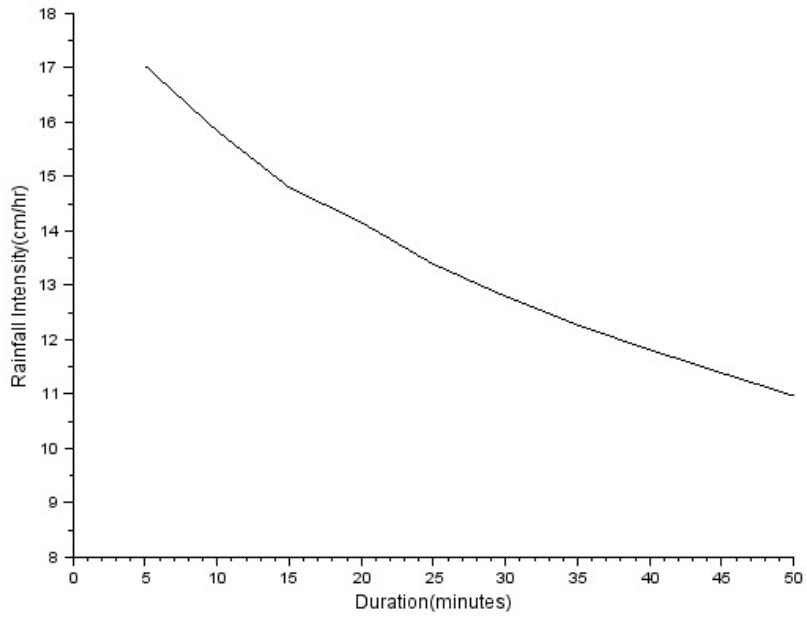
Figure 4.17: EX4 45

```
 2
 3  //example 4.45
 4  //compute maximum rainfall intensities for
        5,10,15,20,25,30,35,40,45,50 minutes
 5  //plot intensity duration graph
 6  clc;funcprot(0);
 7  //given
 8  CR=[0 1.02 2.08 3.30 4.72 5.58 6.40 7.16 7.88 8.54
        9.14];  //cumulative rainfall
 9
10  c5(2)=CR(2);
11  c10(3)=CR(3);
12  c15(4)=CR(4);
13  c20(5)=CR(5);
14  c25(6)=CR(6);
15  c30(7)=CR(7);
16  c35(8)=CR(8);
17  c40(9)=CR(9);
18  c45(10)=CR(10);
19  c50(11)=CR(11);
20  for i=3:11
21      c5(i)=CR(i)-CR(i-1);
22  end
23  for i=4:11
24      c10(i)=CR(i)-CR(i-2);
25  end
26  for i=5:11
27      c15(i)=CR(i)-CR(i-3);
28  end
29  for i=6:11
30      c20(i)=CR(i)-CR(i-4);
31  end
32  for i=7:11
33      c25(i)=CR(i)-CR(i-5);
34  end
35  for i=8:11
36      c30(i)=CR(i)-CR(i-6);
37  end
```

```
38  for i=9:11
39      c35(i)=CR(i)-CR(i-7);
40  end
41  for i=10:11
42      c40(i)=CR(i)-CR(i-8);
43  end
44  for i=11:11
45      c45(i)=CR(i)-CR(i-9);
46  end                                    //rainfall in any
        possible time interval
47
48  mprintf("5min               10min              15min
                   20min              25min
        30min             35min               40min
                   45min             50min");
49  for i=1:11
50      mprintf("\n%f            %f            %f
            %f          %f          %f          %f
                    %f          %f          %f",c5(i),
        c10(i),c15(i),c20(i),c25(i),c30(i),c35(i),c40
        (i),c45(i),c50(i));
51  end
52  I=[17.04 15.84 14.80 14.16 13.39 12.80 12.27 11.82
        11.39 10.97];      //maximum intensity at
        respective durations
53  D=[5:5:50];                      //durations
54  //graph is plotted between I and D
```

Scilab code Exa 4.46 EX4 46

```
1
2
3  //example 4.46
4  //draw storm hyetograph and intensity duration curve
5  clc;funcprot(0);
```
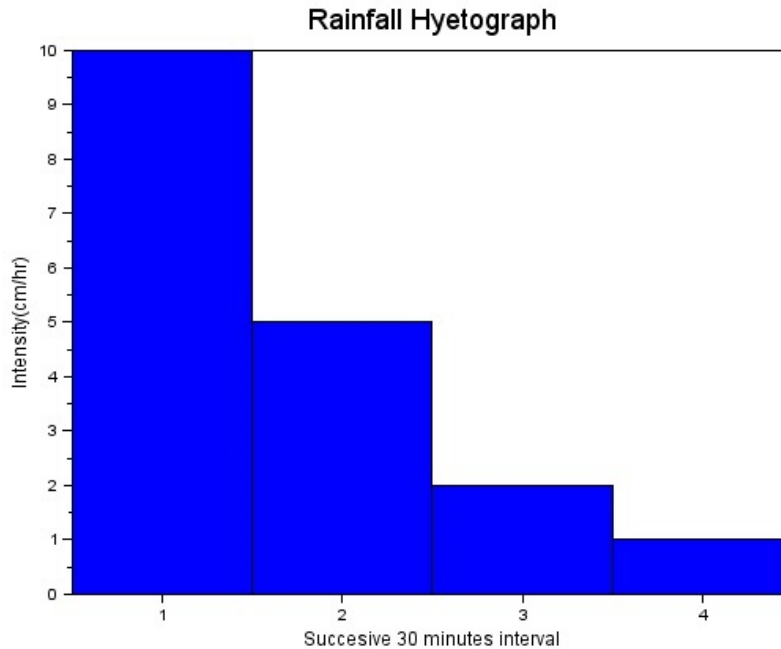
Figure 4.18: EX4 46

```
 6  //given
 7  p=[0 5 7.5 8.5 9];           //accumulated precipitation
 8  t=[0 30 60 90 120];          //time
 9  r(1)=0;
10  mprintf("Rainfall intensity:");
11  for i=2:5
12      r(i)=p(i)-p(i-1);            //rainfall in
             succesive 30 min interval
13      I(i)=r(i)*60/30;            //rainfall intensity
14      mprintf("\n%f",I(i));
15  end
16  //graph is plotted between I and t.
```
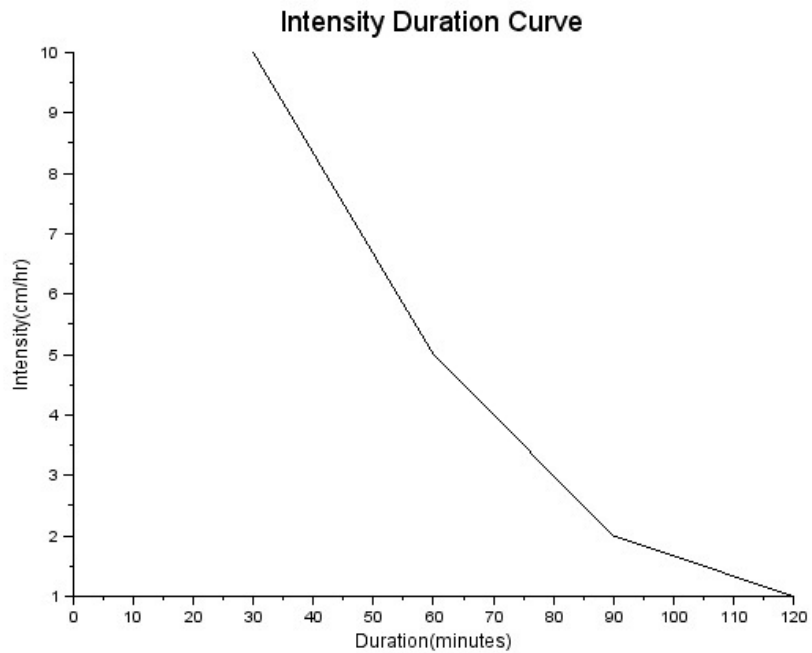
Figure 4.19: EX4 46

Scilab code Exa 4.47 EX4 47

```
1
2
3 //example 4.47
4 //calculate average depth of precipitation using
      depth area curve
5 clc;funcprot(0);
```
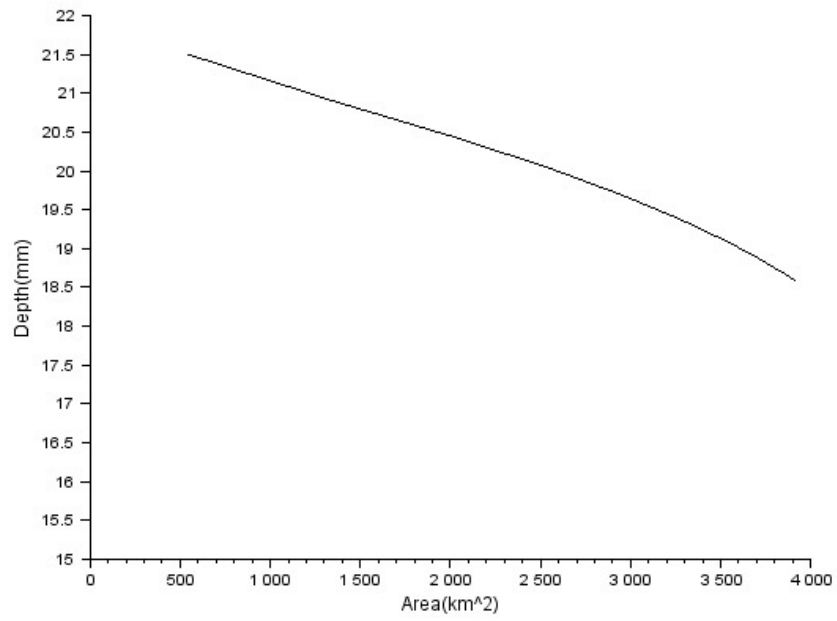
Figure 4.20: EX4 47

```scilab
 6  //given
 7  I=[21:-1:12];                  //isohytes
 8  a=[543 1345 2030 2545 2955 3280 3535 3710 3880
        3915];   //enclosed area
 9  ia(1)=543;
10  for i=2:10
11      ia(i)=a(i)-a(i-1);                    //net
           incremental area between isohytes
12  end
13  r=[21.5:-1:12.5]
14  for i=1:10
15      rv(i)=r(i)*ia(i);                 //rainfall
           volume
16  end
17  cv(1)=11675;
18  for i=2:10
19      cv(i)=cv(i-1)+rv(i);              //cumulative
           volume
20  end
21  for i=1:10
22      eud(i)=cv(i)/a(i);                   //depth(mm)
23  end
24
25  mprintf("From depth area curve we obtain average
        depth of precipitation=20.15 mm for an\narea of
        2400 sq. km.");
26  //graph is plotted between eud and a
```

Scilab code Exa 4.48 EX4 48

```scilab
1
2
3  //example 4.48
4  //calculate evaporation from reservior surface
        during the week
```

```
5  clc;funcprot(0);
6  //given
7  h1=7.75;           //initial  depth  of  water
8  r=3.80;            //rainfall  during  the  week
9  hr=2.50;           //depth  of  water  removed
10 C=0.7;             //pan  coefficient
11 ha=r-hr;
12 hl=ha+h1;
13 h2=8.32;
14 ev=hl-h2;
15 evs=ev*C;
16 evs=round(evs*100)/100;
17 mprintf("evaporation  from  reservior  surface  during
       the  week=%f  cm.",evs);
```

Scilab code Exa 4.49 EX4 49

```
1
2
3  //example4.49
4  //calculate  fi  index  and  time  of  rainfall  excess
5  clc;funcprot(0);
6  //given
7  T=[1:1:12];         //time  from  start
8  r=[1.8  2.6  7.8  3.9  10.6  5.4  7.8  9.2  6.5  4.4  1.8
       1.6];   //increamental  rainfall
9  R=24.4;            //total  run-off
10 s=0;
11 for  i=1:12
12     s=s+r(i);
13 end
14 ti=s-R;
15
16 //first  trial
17 tr=7;    //assumed
```

```
18  ti=s-R-r(1)-r(2)-r(4)-r(11)-r(12);
19  fi=ti/tr;
20  for i=1:12
21      P(i)=r(i)-fi;
22      if (P(i)<0) then
23          P(i)=0;
24      end
25  end
26  mprintf("Time(h)           rainfall  excess .");
27  for i=1:12
28      mprintf("\n%f           %f",T(i),P(i));
29  end
30  mprintf("\n\nfi  index=%f cm/hr .",fi);
```

Scilab code Exa 4.50 EX4 50

```
1
2
3  //example  4.50
4  //calculate  fi  index
5  clc;funcprot(0);
6  //given
7  r=[0.6  1.35  2.25  3.45  2.7  2.4  1.5  0.75];        //
        incremental  rainfall
8  T=[1:1:8];                    //time  from  start  of  rainfal
9  t=8;
10 P=15;                     //total  rainfall
11 R=8.7;                    //direct  run-off
12 W=(P-R)/t;
13 //since  fi  wil  be  more  than W
14 tre=6;
15 fi=((P-R)-r(1)-r(8))/tre;
16 mprintf("fi  index=%f cm/hr .",fi);
```
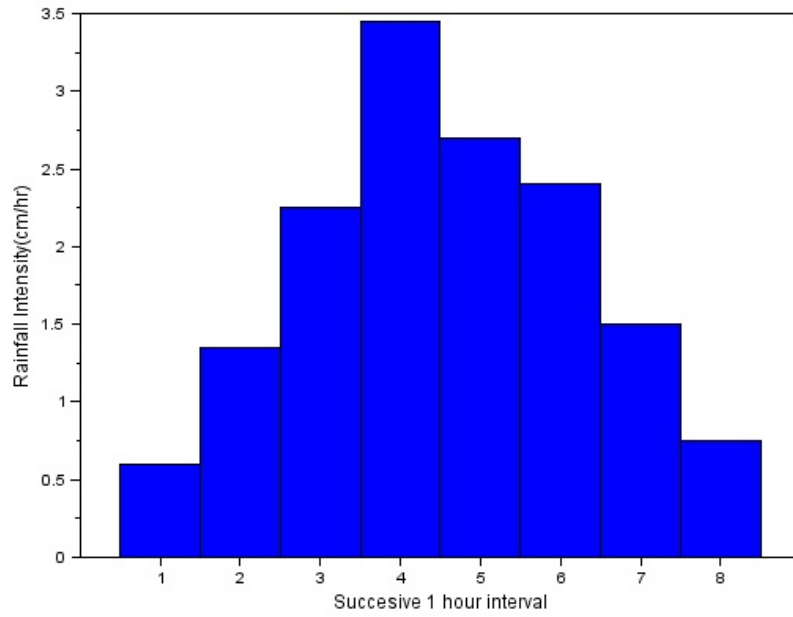
Figure 4.21: EX4 50

Scilab code Exa 4.51 EX4 51

```
1  //example 4.51
2  //calculate total infiltration depth lasting 6 hrs
3  clc;funcprot(0);
4  //given
5  I=10;              //total infiltration rate
6  fI=5;              //final infiltration rate
7  k=0.95;            //rate of decay of difference
       between final and initial infiltration rate
8
9  q=integrate('fI+(I-fI)*%e^(-k*t)','t',0,6);
10 q=round(q*100)/100;
11 mprintf("total infiltration depth=%f mm.",q);
```

Scilab code Exa 4.52 EX4 52

```
1  //example 4.52
2  //find the equation of infiltration capacity
3  clc;funcprot(0);
4  //given
5  fc=1;              //constant infiltration rate
6  ft=[10.4 5.6 3.2 2.1 1.5 1.2 1.1 1 1];   //
       infiltration capacity
7  f=ft(1)-fc;
8  t=[0:0.25:2];
9
10 for i=1:9
11     r(i)=ft(i)-fc;
```

Figure 4.22: EX4 52

115
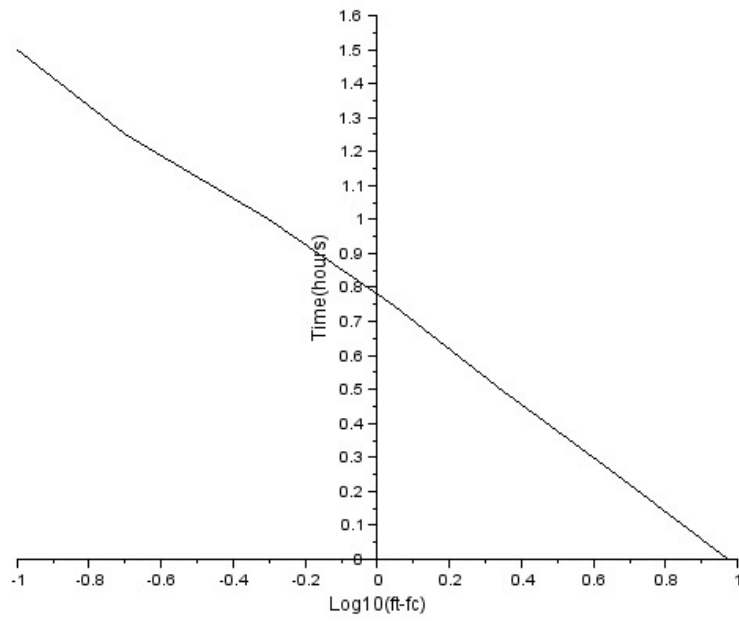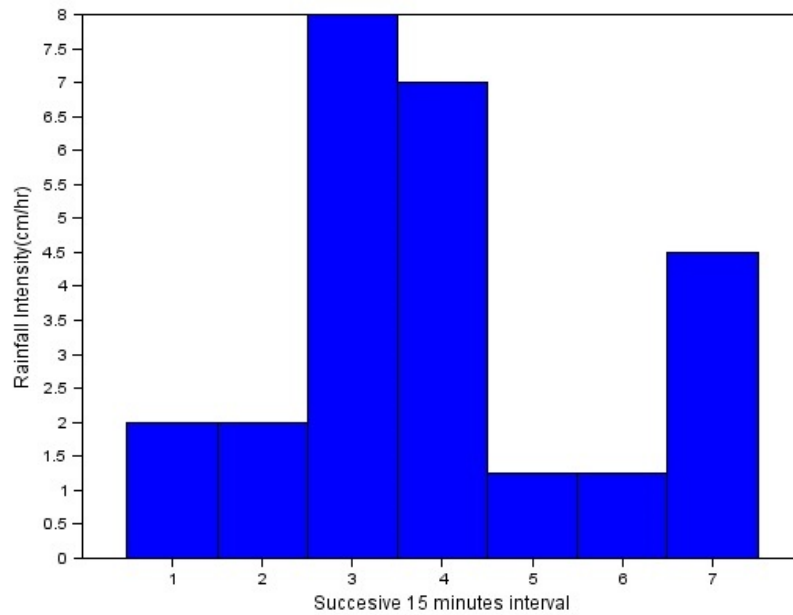
Figure 4.23: EX4 53

```
12
13  end
14  for i=1:7
15      h(i)=log10(r(i));
16  end
17  s=0.775;          //from graph
18  k=1/(log10(%e)*s);
19  k=round(k*100)/100;
20  mprintf("Equation is :\ nft=fc+%fe^(-%ft)",f,k);
```

Scilab code Exa 4.53 EX4 53

```scilab
1
2
3  //example 4.53
4  //calculate
5  //total rainfall
6  //net run-off
7  //W index
8  clc;funcprot(0);
9  //given
10 r=[2 2 8 7 1.25 1.25 4.5];          //rainfall
      intensity
11 T=[15 30 45 60 70 90 105];                    //
      time
12 dt=15;                                  //time interval
13 fi=3;                                   //fi index
14 //graph is plotted between r and T
15 s=0;
16 for i=1:7
17     s=s+r(i);
18 end
19 P=s*dt/60;
20 Pe=((8-3)+(7-3)+(4.5-3))*dt/60;   //area of graph
      above r=3.0.
21 w=(P-Pe)/(105/60);
22 w=round(w*1000)/1000;
23 mprintf("total rainfall=%f cm.",P);
24 mprintf("\nnet run-off=%f cm.",Pe);
25 mprintf("\nW index=%f cm/hr.",w);
```

Scilab code Exa 4.54 EX4 54

```scilab
1
2
3  //example 4.54
4  //calculate Total rainfall in catchment
```

```scilab
5  //run-off by rainfall of 3.3cm in 3hrs
6  clc;funcprot(0);
7  //given
8  A=[36 18 66];                //area of catchment
9  fi=[0.9 1.1 0.5];            //fi index
10 r1=[0.6 0.9 1.0];            //rainfall in first hour
11 r2=[2.4 2.1 2.0];            //rainfall in second hour
12 r3=[1.3 1.5 0.9];            //rainfall in third hour
13
14 t36=r1(1)+r2(1)+r3(1);
15 t18=r1(2)+r2(2)+r3(2);
16 t66=r1(3)+r2(3)+r3(3);
17
18 p=(t36*A(1)+t18*A(2)+t66*A(3))/(A(1)+A(2)+A(3));
19 mprintf("Total rainfall in catchment=%f cm.",p);
20
21 ro1=[0 0 0.5];ro2=[1.5 1.0 1.5];ro3=[0.4 0.4 0.4];
          //rainfall-fi
22 t1=ro1(1)+ro2(1)+ro3(1);
23 t2=ro1(2)+ro2(2)+ro3(2);
24 t3=ro1(3)+ro2(3)+ro3(3);
25 run=(A(1)*t1+A(2)*t2+A(3)*t3)/(A(1)+A(2)+A(3));
             //run-off from entire catchment
26 mprintf("\nrun-off by rainfall of 3.3cm in 3hrs=%f
     cm.",run);
27 fia=(fi(1)*A(1)+fi(2)*A(2)+fi(3)*A(3))/(A(1)+A(2)+A
     (3));
28 tr=(1.1-fia)*3;
29 mprintf("\nTotal run-off=%f cm.",tr);
```

Scilab code Exa 4.55 EX4 55

```scilab
1
2
3  //example 4.55
```

```
4  //calculate relation between R and P
5  clc;funcprot(0);
6  //given
7  P=[4 22 28 15 12 8 4 15 10 5];   //Precipitation
8  R=[0.2 7.1 10.9 4.0 3.0 1.3 0.4 4.1 2.0 0.3];   //
       runoff
9  for i=1:10
10     Ps(i)=P(i)^2;
11     Rs(i)=R(i)^2;
12     PR(i)=P(i)*R(i);
13 end
14
15 s=0;t=0;u=0;q=0;w=0;
16 for i=1:10
17     s=s+Ps(i);
18     t=t+Rs(i);
19     u=u+PR(i);
20     q=q+P(i);
21     w=w+R(i);
22 end
23 N=10;
24 a=(N*u-q*w)/(N*s-q^2);
25 b=(w-a*q)/N;
26 a=round(a*10000)/10000;
27 b=round(b*10000)/10000;
28 mprintf("Equation is:\n%fP%f.",a,b);
```

Scilab code Exa 4.56 EX4 56

```
1
2
3  //example 4.56
4  //calculate peak discharge of 6 hrs unit hydrograph
5  clc;funcprot(0);
6  //given
```

```
 7  Q=470;                    //peak  discharge  of  flood
       hydrograph
 8  B=15;                     //base  flow
 9  l=0.25;                   //infiltration  loss
10  Qr=Q-B;
11  d=8;                      //average  depth  of  rainfall
12  re=d-l*6;                 //rainfall  excess
13  q=Qr/re;
14  mprintf("peak  discharge  of  6  hrs  unit  hydrograph=%i
       cumecs.",q);
```

Scilab code Exa 4.57 EX4 57

```
 1
 2
 3  //example  4.57
 4  //calculate  ordinates  of  storm  hydrograph
 5  clc;funcprot(0);
 6  //given
 7  fi=0.25;          //infiltration  index
 8  B=20;             //Base  flow
 9  O=[0  20  60  150  120  90  70  50  30  20  10  0  0  0];    //
       ordinates  of  unit  hydrograph
10  R1=5;R2=0.8;R3=3;
11  r1=R1-(fi*4);                                    //
       rainfall  excess  in  first  four  hour
12  r2=R2-(fi*4);                                    //
       rainfall  excess  in  second  four  hour
13  r3=R3-(fi*4);                                    //
       rainfall  excess  in  third  four  hour
14  if  r2<0
15      r2=0;
16      end
17
18  for  i=1:14
```

120

```
19      s1(i)=r1*O(i);
20  end
21  for i=2:14
22      s2(i)=r2*O(i-1);
23  end
24  for i=3:14
25      s3(i)=r3*O(i-2);
26  end                                          //
        surface run-off from rainfall excess during
        succesive unit periods
27  mprintf("ordinates of storm hydrograph");
28  for i=1:14
29      T(i)=s1(i)+s2(i)+s3(i);                  //sub-total
30      t(i)=T(i)+B;                             //ordinate
            of flood hydrograph
31      mprintf("\n%i",t(i));
32  end
```

Scilab code Exa 4.58 EX4 58

```
1
2
3  //example 4.58
4  //calculate ordinates of discharge hydrograph and
        peak discharge
5  clc;funcprot(0);
6  //given
7  fi=2.5;              //fi index
8  t=24;
9  A=200;               //area of catchment
10 R1=7.5;R2=2.0;R3=5;  //rainfall
11 r1=R1-fi;r2=R2-fi;r3=R3-fi;
12 r2=0;
13 r=[5 0 2.5];         //excess rainfall
14 D=[5 15 40 25 10 5 0 0 0];  //distribution
```

```
15  for i=1:9
16      d1(i)=D(i)*r(1)/100;
17  end
18  for i=1:8
19       d2(i+1)=D(i)*r(2)/100;
20  end
21  for i=1:7
22       d3(i+2)=D(i)*r(3)/100;
23  end                                            //
        distribution run-off for rainfall excess
24
25  for i=1:9
26      tr1(i)=d1(i)+d2(i)+d3(i);              //total run-
            off as depth
27      tr2(i)=23.148*tr1(i);                 //total run-
            off as discharge
28      tr2(i)=round(tr2(i)*1000)/1000;
29  end
30  s=0;
31  for i=1:9
32      s=s+tr2(i);
33  end
34  mprintf("Total run-off:");
35  mprintf("\nas depth        as discharge");
36  for i=1:9
37      mprintf("\n%f           %f",tr1(i),tr2(i));
38  end
39  r=0.36*s*t/A;
40  r=round(r*10)/10;
41  mprintf("\ntotal run-off=%f cm.",r);
```

Scilab code Exa 4.59 EX4 59

```
1
2
```

```
3  // example 4.59
4  // calculate ordinate of 6 hr unit hydrograph
5  clc ; funcprot (0) ;
6  // given
7  O = [10 30 90 220 280 220 166 126 92 62 40 20 10];
            // ordinates of 6 hr flood hydrograph
8  B =10;              // Base flow
9  for i =1:13
10     r ( i ) = O ( i ) -B ;              // ordinates of direct run
          -off
11 end
12 mprintf ( " Ordinates of 6 hr unit hydrograph " ) ;
13 u (1) =0;
14 for i =2:13
15     u ( i ) = r ( i ) -u ( i -1) ;              // ordinates of 6 hrs
          unit hydrograph
16 end
17 for i =1:13
18     mprintf ( " \n%i " ,u ( i ) ) ;
19 end
```

Scilab code Exa 4.60 EX4 60

```
1
2
3  // example 4.60
4  // determine the ordinates of 1 cm-6 hour hydrograph
5  clc ; funcprot (0) ;
6  // given
7  t =6;
8  A =450;         // catchment area
9  O = [5 15 40 80 60 50 25 15 5];      // ordinates of flood
      hydrograph
10 B =5;           // base flow assumed
11 s =0;
```

```
12  for  i =1:9
13      r(i)=O(i)-B;            // ordinates  of  direct  run−
            off
14  s=s+r(i);
15  end
16  n=s*0.36*12/A;
17  mprintf(" ordinates  of  unit  hydrograph");
18  for  i =1:9
19      u(i)=r(i)/n;
20      u(i)=round(u(i)*100)/100;
21      mprintf("\n%f",u(i));
22  end
```

Scilab code Exa 4.61 EX4 61

```
 1
 2
 3  //example  4.61
 4  //obtain  ordinates  24  hr  unit  hydrograph
 5  clc; funcprot(0);
 6  //given
 7  O=[0  5.5  13.5  26.5  45  82  162  240  231  165  112  79  57
      42  31  22  14  9.5  6.6  4  2  1  0  0  0  0  0];  //
      ordinates  of   1st  8  hrs  unit  hydrograph
 8  for  i =1:25
 9      o1(i+2)=O(i);                          //
          ordinates  of   2nd  8  hrs  unit  hydrograph
10      o2(i+4)=O(i);                          //
          ordinates  of   3rd  8  hrs  unit  hydrograph
11  end
12  mprintf(" ordinates  24  hr  unit  hydrograph:");
13  for  i =1:27
14      o3(i)=o1(i)+o2(i)+O(i);            // total  24  hr
          hydrograph  of  3  cm  run−off
15      t(i)=o3(i)/3;
```
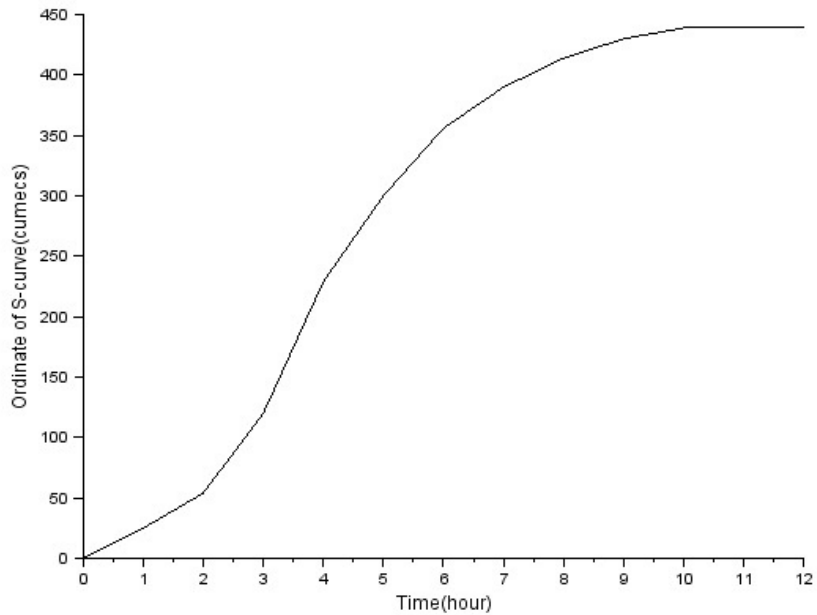
Figure 4.24: EX4 62

```
16        t(i)=round(t(i)*10)/10;
17        mprintf("\n%f",t(i));
18 end
```

Scilab code Exa 4.62 EX4 62

```
1 //example 4.62
2 //ordinates of 1 hr unit hydrograph
3 clc;funcprot(0);
4 //given
5 t=[0:1:12];        //time
```

```
6  O=[0 0 54 0 175 0 127 0 58 0 25 0 0 0];                //
       ordinate  of 2 hr unit  hydrograph
7  of(1)=0;
8  of(2)=0;
9  for i=3:13
10     if modulo(i,2)==0;
11          of(i)=0;
12
13  else
14      of(i)=O(i-2)+of(i-2);
15  end
16  end
17  s=[0 25 54 120 229 300 356 390 414 430 439 439 439];
              //Ordinates of S-curve
18  for i=2:13
19      of1(i)=s(i-1);
20  end
21  mprintf("ordinates of 1 hr unit hydrograph:");
22  for i=1:13
23      y(i)=s(i)-of1(i);
24  u(i)=y(i)*2;
25  mprintf("\n%i",u(i));
26  end
27  //graph is plotted between u and t
```

Scilab code Exa 4.63 EX4 63

```
1
2
3  //example 4.63
4  //calculate  design disharge
5  clc;funcprot(0);
6  //given
7  xavg=1200;          //sample mean
8  n=50;               //assurance year
```

126

```
 9  A=0.95;               // assurance  percent
10  Rsk=1-A;
11  sigma=650;            // standard  deviation
12  yn=0.53622;           // mean  of  reduced  variate
13  sigma30=1.11238;      // standard  deviation  of  reduced
        variate
14
15  T=1/(1-(1-Rsk)^(1/n));
16  yt=-2.303*log10(2.303*log10(T/(T-1)));
17  K=(yt-yn)/sigma30;
18  xt=xavg+K*sigma;
19  mprintf(" design  disharge=%i  cumecs.",xt);
```

Scilab code Exa 4.64 EX4 64

```
 1
 2
 3  // example  4.64
 4  // calculate  flood  magnitude  with  return  period  of
        500  years
 5  clc; funcprot (0);
 6  // given
 7  T1=50; T2=100;                 // Return  period
 8  F1=20600; F2=22150;            // Peak  flood
 9  y100=-(2.303*log10(2.303*log10(T2/(T2-1))));
10  y50=-(2.303*log10(2.303*log10(T1/(T1-1))));
11  y=(F2-F1)/(y100-y50);
12  T=500;
13  y500=-(2.303*log10(2.303*log10(T/(T-1))));
14  x500=F2+(y500-y100)*y;
15  x500=round(x500);
16  mprintf("flood  magnitude  with  return  period  of  240
        years=%f  cumec.",x500);
```

Scilab code Exa 4.65 EX4 65

```
1
2
3  //example  4.65
4  //calculate  recurrence  interval  of  10  minutes  storm
       using  Gumbel's  method
5  clc;funcprot(0);
6  //given
7  xavg=1.65;          //mean  of  data
8  sigma=0.45;         //standard  deviation
9  x=3;
10 y=1.2825*(x-xavg)/sigma+0.577;
11 l=%e^(%e^(-y));
12 T=l/(l-1);
13 T=round(T*10)/10;
14 mprintf("recurrence  interval  of  10  minutes  storm=%f
       years.",T);
```

Scilab code Exa 4.66 EX4 66

```
1
2
3  //example  4.66
4  //calculate  flood  magnitude  with  return  period  of
       200  years
5  clc;funcprot(0);
6  //given
7  T1=50;T2=100;                  //Return  period
8  F1=30800;F2=36300;         //Peak  flood
9  y100=-(2.303*log10(2.303*log10(T2/(T2-1))));
10 y50=-(2.303*log10(2.303*log10(T1/(T1-1))));
```

```
11 y=(F2-F1)/(y100-y50);
12 T=200;
13 y200=-(2.303*log10(2.303*log10(T/(T-1))));
14 x200=F2+(y200-y100)*y;
15 x200=round(x200);
16 mprintf("flood magnitude with return period of 240
       years=%f cumecs.",x200);
```

Scilab code Exa 4.67 EX4 67

```
1
2
3 //example 4.67
4 //calculate return period of flood of 9950 cumec/s
5 clc;funcprot(0);
6 //given
7 xavg=4200;        //mean
8 sigma=1705;       //standard deviation
9 xt=9550;          //flood value
10 K=(xt-xavg)/sigma;
11 yt=1.2825*K+0.577;
12 l=%e^(%e^(-yt));
13 T=l/(l-1);
14 T=round(T*100)/100;
15 mprintf("Return period of flood of 9950 cumec/s=%f
       years.",T);
```

Scilab code Exa 4.68 EX4 68

```
1
2
3 //example 4.68
```

```
 4  // calculate  flood  magnitude  with  return  period  of
        1000  years
 5  clc ; funcprot (0) ;
 6  // given
 7  T1 =100; T2 =50;                    // Return  period
 8  F1 =485; F2 =445;          // Peak  flood
 9  y50 = -(2.303* log10 (2.303* log10 (T2 /( T2 -1)))) ;
10  y100 = -(2.303* log10 (2.303* log10 (T1 /( T1 -1)))) ;
11  y =( F2 -F1 ) /( y50 - y100 ) ;
12  T =1000;
13  y1000 = -(2.303* log10 (2.303* log10 (T /(T -1)))) ;
14  x1000 = F2 +( y1000 - y50 ) *y;
15  x1000 = round ( x1000 *10) /10;
16  mprintf ("flood  magnitude  with  return  period  of  240
        years =%f  cumecs ." , x1000 ) ;
```

Scilab code Exa 4.69 EX4 69

```
 1
 2
 3  // example  4.69
 4  // calculate
 5  // probability  of  exceedence
 6  // probability  of  occurence  in  next  12  years
 7  clc ; funcprot (0) ;
 8  // given
 9  T =25;          // return  period
10  n =12;
11  P =1/ T;
12  Rsk =1 -(1 -P)^n;
13  P = round (P *100) /100;
14  Rsk = round ( Rsk *10000) /10000;
15  mprintf (" probability  of  exceedence =%f." ,P);
16  mprintf ("\ nprobability  of  occurence  in  next  12  years
        =%f." , Rsk ) ;
```

Chapter 5

GROUND WATER WELL IRRIGATION

Scilab code Exa 5.1 EX5 1

```
1
2
3  //example 5.1
4  //design an open wellin fine sand
5  clc;
6  //given
7  Q=0.003;          //required discharge
8  H=2.5;            //depression head
9  A=Q*3600/(0.5*H);
10 d=(4*A/%pi)^0.5;
11 d=round(d*100)/100
12 mprintf("Well diameter=%f m.",d);
13
14 //Alternative solution
15 C=7.5D-5;    //permeability constant from table 5.2
16 A=Q/(C*H);
17 d=(16*3/%pi)^0.5;
18 d=round(d*10)/10;
19 mprintf("\nBy alternative solution:")
```

```
20 mprintf("\nWell diameter=%f m",d);
```

Scilab code Exa 5.2 EX5 2

```
1
2
3 //example 5.2
4 //calculate
5 //yield from well
6 //diameter of well
7 clc;
8 //given
9 h1=2.5;                    //initial pumping
      depression
10 h=1.8;                    //heigth after recuperation
11 t=80;                     //time
12 h2=h1-h;
13 KbyA=2.303*60*log10(h1/h2)/t;
14
15
16 //Part (a)
17 d=4;           //diameter of well
18 H=3;           //depression head
19 A=%pi*d^2/4;
20 Q=(KbyA)*A*H/3.6;
21 mprintf("Part (a)");
22 Q=round(Q);
23 mprintf("\nYield from well=%f lit/sec.",Q);
24
25 //Part (b)
26 Q=8;           //yield(lit/sec)
27 H=2;
28 A=Q*3.6/(H*(KbyA));
29 d=(4*A/%pi)^0.5;
30 d=round(d*10)/10;
```

```
31  mprintf("\nPart (b)");
32  mprintf("\nDaimeter of well=%f m",d);
```

Scilab code Exa 5.3 EX5 3

```
1
2
3  //example 5.3
4  //calculate yield from well
5  clc;
6  //given
7  d=30;              //well diameter
8  L=15;              //strainer length
9  P=50;              //coefficient of permeability
10 s=0.2;             //effective size of sand
11 b=3;               //drawdown
12 r=150;             //radius of drawdown
13
14 Q=2.72*L*P*b/(log10(r*2*100/d)*24*3.6);
15 Q=round(Q*10)/10;
16 mprintf(" yield from well=%f lit/sec.",Q);
```

Scilab code Exa 5.4 EX5 4

```
1
2
3  //example 5.4
4  //calculate discharge from tubewell
5  clc;
6  //given
7  d=30;                //diameter of well
8  s=2;                 //drawdown
9  L=10;                //length of stainer
```

```
10  k=0.05;            //coefficient of permeability
11  r=300;             //radius of zero drawdown
12  Q=2.72*k*s*(L+s/2)/(100*log10(2*100*r/d));
13  Q=round(Q*10000)/10000;
14  mprintf(" discharge from tubewell=%f cumec.",Q);
```

Scilab code Exa 5.5 EX5 5

```
1
2
3  //example 5.5
4  //design tube well
5  clc;
6  //given
7  Q=0.08;                //yield required
8  b=30;                  //thickness of acquifer
9  R=300;                 //Radius of circle of influence
10  k=60;                  //permeability coefficient
11  s=5;                   //Drawdown
12  r=R/(10^(2.72*b*s*k/(3600*24*Q)));
13  r=round(r*10000)/10000;
14  mprintf("Radius of well=%f m",r);
```

Scilab code Exa 5.6 EX5 6

```
1
2
3  //example 5.6
4  //calculate yield from well
5  clc;
6  //given
7  b=30;                    //thickness of acquifer
8  s=4;                     //drawdown
```

135

```
 9  r =0.1;                        // well radius
10  k =36;                         // permeability coefficient
11  R=3000*s*(k/(24*3600))^0.5;
12
13  Q=2.72*b*k*s/(log10(R/r)*24*3.6);
14  Q=round(Q*10)/10;
15  mprintf("yield from well=%f lit/sec.",Q);
```

Scilab code Exa 5.7 EX5 7

```
 1
 2
 3  //example 5.7
 4  //calculate discharge and percent increase in
        discharge
 5  clc;
 6  //given
 7  k =0.005;                // coefficient of permeability
 8  r =0.1;                  // well radius
 9  s =4;                    // drawdown
10  b =10;                   // thickness
11  R =300;                  // radius of circle of influence
12  //Part(a)
13  Q1=2.72*b*k*s/log10(R/r);
14  Q1=round(Q1*10000)/10000;
15  mprintf("Discharge=%f cumec",Q1);
16
17  //Part (b)
18  r =0.2;
19  Q2=2.72*b*k*s/log10(R/r);
20  I=(Q2-Q1)*100/Q1;
21  I=round(I*10)/10;
22  mprintf("\npercent increase in discharge=%f percent.
        ",I);
```

Scilab code Exa 5.8 EX5 8

```
1
2
3  //example 5.8
4  //calculate coefficient of permeability
5  //percentage error
6  //actual radius of influence
7  clc;
8  //given
9  d=0.2;                  //diameter of well
10 Q=240;                  //discharge
11 RL1=240.5;              //reduce level of original water
        surface
12 RL2=235.6;              //reduced level of water at pumping
13 RL3=210;                //reduced level of impervious layer
14 RL4=239.8;              //reduced level of water in well
15 D=50;                   //radial distance of well from
        tube well
16 //Part(a)
17 h1=RL2-RL3;
18 h2=RL4-RL3;
19 k1=Q*24*log10(D*2/d)/(1.36*(h2^2-h1^2));
20 k1=round(k1*100)/100;
21 mprintf("Part(a)");
22 mprintf("\ncoefficient of permeability=%f m/day.",k1
    );
23 //Part (b)
24 R=300;                  //radius of influence
25 H=RL1-RL3;
26 h=RL2-RL3;
27 k2=Q*24*log10(R*2/d)/(1.36*(H^2-h^2));
28 PE=(k2-k1)*100/k1;
29 mprintf("\nPart(b)");
```

```
30  mprintf("\npercentage error=%i percent.",PE);
31  //Part (b)
32  R=(d/2)*10^(1.36*k1*(H^2-h^2)/(24*Q));
33  mprintf("\nPart(c)");
34  mprintf("\nActual radius of influence=%i m.",R);
```

Scilab code Exa 5.9 EX5 9

```
1
2
3  //example 5.9
4  //calculate input h.p of pump
5  clc;
6  //given
7  A=20;                //area of field
8  H=129;               //level to the highest land
9  h1=120.2;            //water level in well during
        discharge
10 Du=800;              //duty for rise;
11 eita=0.6;            //efficiency of the pump
12 Q=A/Du;
13 w=Q*1000;
14 lift=H-h1;
15 //design lift is taken as 9m
16 wd=w*9;
17 o=wd/75;
18 i=o/eita;
19 mprintf("Input h.p of pump=%i h.p",i);
```

Scilab code Exa 5.10 EX5 10

```
1
2
```

```
 3  //example  5.10
 4  //calculate  culturable  area
 5  clc;
 6  //given
 7  Q=150;              //discharge  from  tubewell
 8  t=4000;             //working  period  of  tubewell
 9  I=0.45;             //intensity  of  irrigation
10  d=0.38;             //average  depth  of  rabi  and  kharif
        crop
11  V=Q*t;
12  A=V/d;
13  CA=A/(I*10000);
14  CA=round(CA);
15  mprintf("culturable  area=%f  hectares.",CA);
```

Scilab code Exa 5.11 EX5 11

```
 1
 2
 3  //example  5.11
 4  //calculate  discharge  if  one  well  discharges
 5  //percent  decrease  when  two  well  discharges
 6  clc;
 7  //given
 8  d=0.2;              //diameter  of  well
 9  r=d/2;
10  B=100;               //distance  between  wells
11  b=12;               //thickness  of  acquifer
12  k=60;               //coefficient  of  permeability
13  s=3;                //dispersion  head
14  R=250;              //radius  of  influence
15  Q=2.72*b*k*s/(24*log10(R/r));
16  mprintf("discharge  if  one  well  discharges=%i  cubic
        metre/hour.",Q);
17  //when  both  well  are  discharging
```

```
18  Q1=2.72*k*b*s/(24*log10(R^2/(r*B)));
19  Q1=round(Q1*10)/10;
20  mprintf("\ndischarge if both wells discharges=%f
        cubic metre/hour.",Q1);
21  PE=(Q-Q1)*100/Q;
22  PE=round(PE*100)/100;
23  mprintf("\npercentage decrease in discharge=%f
        percent.",PE);
```

Scilab code Exa 5.12 EX5 12

```
1
2
3  //example 5.12
4  //calculate radius of zero drawdown
5  //coefficient of permeability
6  //drawdown in well
7  //specific capacity
8  //maximum rate at which water can be pumped
9  clc;
10  //given
11  d=0.6;                //diameter of well;
12  rw=d/2;
13  H=40;                 //depth of water in well before
        pumping
14  Q=2000;               //discharge from well
15  s1=4;                 //drawdown in well
16  B1=10;                //distance between well
17  s2=2;
18  B2=20;
19  //Part (a)
20  h1=H-s1;
21  h2=H-s2;
22  t=(H^2-h2^2)/(H^2-h1^2);
23  R=(B2/(B1^t))^(1/(1-t));
```

```scilab
24  R=round(R*100)/100;
25  mprintf(" radius of zero drawdown=%f m",R);
26  //Part (b)
27  r=10;
28  k=Q*log10(R/r)*60*24/(1.36*(H^2-h1^2)*1000);
29  k=round(k*100)/100;
30  mprintf("\ncoefficient of permeability=%f m/day.",k)
        ;
31
32  //part (c)
33  Ho=(H^2-(Q*log10(R/rw)*24*60/(1000*1.36*k)))^0.5;
34  D=H-Ho;
35  D=round(D*100)/100;
36  mprintf("\ndrawdown in well=%f m.",D);
37
38  //part (d)
39  C=Q/(1000*R);
40  //for R=1 m;Q=Sc
41  //hence on putting the values in discharge equation
        we get
42  //Sc*log10(61.2*Sc)=0.3223.
43  //on solving this by trial and error method we get
        Sc=0.266 m^2/min.
44  mprintf("\nSpecific capacity=0.266 cubic metre/
        minutes/metre.");
45
46  //part (e)
47  //this is obtained when Q=H
48  //hence from equation of discharge ,we get
49  //Q*log10(69.2*Q)=6.528.
50  //solving it by trial and error method we get Q=2.85
        m^3/min.
51  mprintf("\nmaximum rate at which water can be pumped
        =2.85 cubic metre/min");
```

Scilab code Exa 5.13 EX5 13

```
1
2
3  //example 5.13
4  //calculate formation constant of acquifer using
       theis method
5  clc;funcprot(0);
6  //given
7  Q=2500;        //discharge(l/min)
8  r=60;          //distance of observation well from
       acquifer
9  tmin=[1 1.5 2 2.5 3 4 5 6 8 10 12 14 18 24 30 40 50
       60 80 100 120 150 180 210 240];   //time in
       minutes
10 s=[0.2 0.26 0.3 0.33 0.36 0.41 0.45 0.48 0.53 0.56
       0.59 0.62 0.66 0.71 0.75 0.80 0.83 0.86 0.91 0.95
        0.98 1.03 1.05 1.08 1.10];   //drawdown
11 u=[1:1:9];
12 Wu=[0.2194 0.04891 0.01315 0.003779 0.001148
       0.000360 0.000116 0.0000377 0.0000125];
13 for i=1:25
14     tday(i)=tmin(i)/(60*24);
15 end
16
17 for i=1:25
18     rt(i)=r^2/tday(i);
19 end
20 //graph is plotted between s and r^2/t and W(u) and
       u and they are superimposed.
21 //from which we get
22 s1=0.52;
23 Wu1=2.96;
24 rt1=700000;  u1=0.03;
25 Q=3600;                 //discharge in cumec/day
26 T=Q*Wu1/(4*%pi*s1);
27 S=4*u1*T/rt1;
28 T=round(T);
```

142
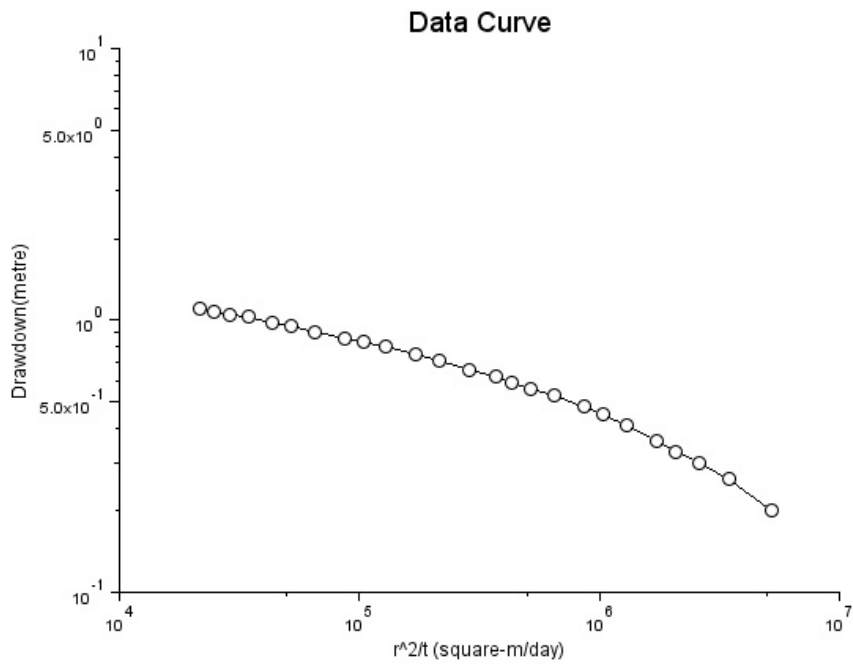
Figure 5.1: EX5 13

```
29  mprintf("formation  constant  of  acquifer :");
30  mprintf("\nT=%f  cubic  metre/day/m.\nS=%f.",T,S);
```

Scilab code Exa 5.14 EX5 14

```
1
2
```

Figure 5.2: EX5 13

144
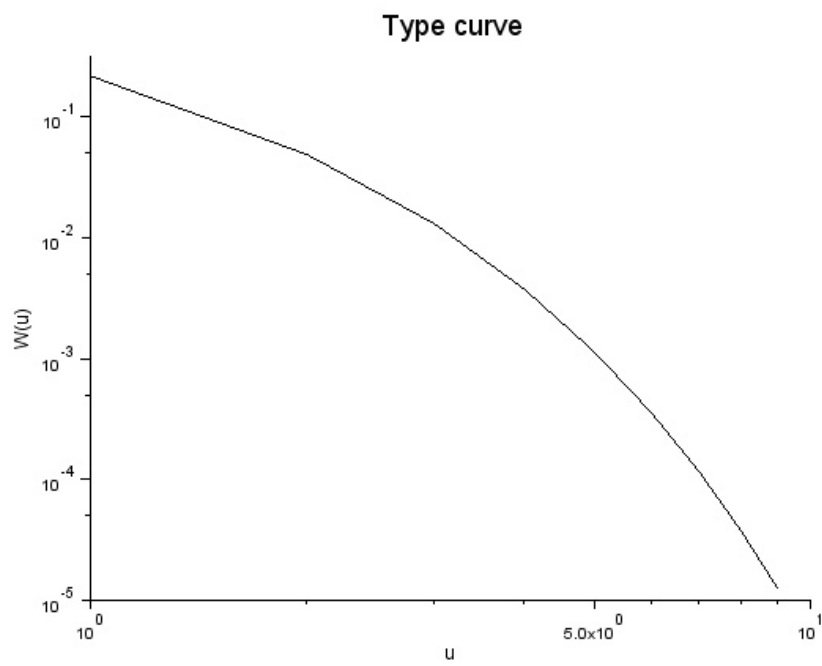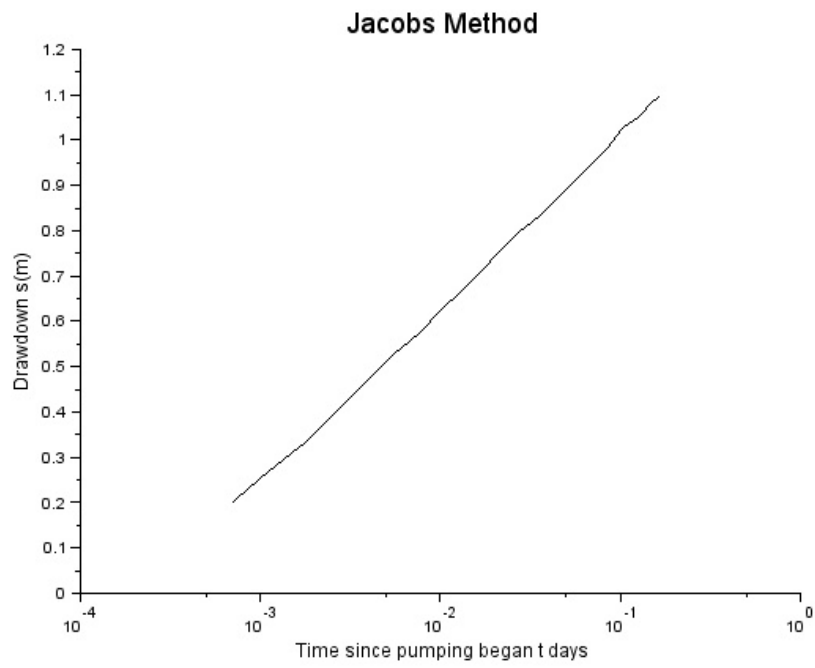
Figure 5.3: EX5 14

```
3  //example 5.14
4  //calculate formation constant of acquifer using
        Jacob's method
5  clc;funcprot(0);
6  //given
7  Q=2500;         //discharge(l/min)
8  r=60;           //distance of observation well from
        acquifer
9  tmin=[1 1.5 2 2.5 3 4 5 6 8 10 12 14 18 24 30 40 50
        60 80 100 120 150 180 210 240];   //time in
        minutes
10  s=[0.2 0.26 0.3 0.33 0.36 0.41 0.45 0.48 0.53 0.56
        0.59 0.62 0.66 0.71 0.75 0.80 0.83 0.86 0.91 0.95
         0.98 1.03 1.05 1.08 1.10];   //drawdown
11  for i=1:25
12      tday(i)=tmin(i)/(60*24);
13  end
14  //from the graph between s and t we get
15  ds=0.38;
16  Q=3600;         //discharge in cumec/day
17  T=2.303*Q/(4*%pi*ds);
18  //extending the straight line we get
19  to=0.00024;
20  S=2.25*T*to/r^2;
21  mprintf("formation constant of acquifer:");
22  mprintf("\nT=%i cubic metre/day/m.\nS=%f.",T,S);
```

Scilab code Exa 5.15 EX5 15

```
1  //example 5.15
2  //calculate formation constant of acquifer using
        Chow's method
3  clc;funcprot(0);
```

Figure 5.4: EX5 15

147
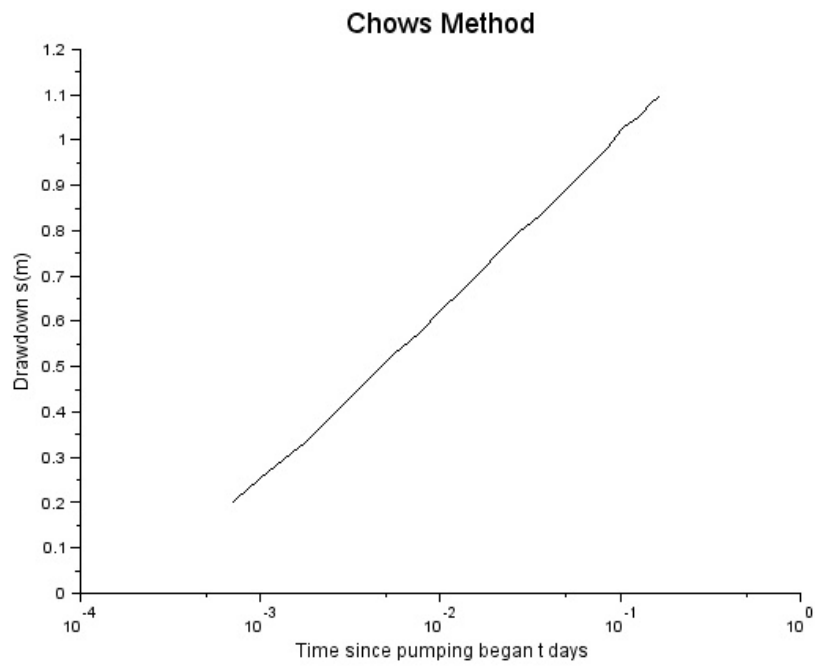
```
4  // given
5  Q=2500;            // discharge ( l / min )
6  r=60;              // distance of observation well from
        acquifer
7  tmin =[1 1.5 2 2.5 3 4 5 6 8 10 12 14 18 24 30 40 50
        60 80 100 120 150 180 210 240];   // time in
        minutes
8  s=[0.2 0.26 0.3 0.33 0.36 0.41 0.45 0.48 0.53 0.56
        0.59 0.62 0.66 0.71 0.75 0.80 0.83 0.86 0.91 0.95
         0.98 1.03 1.05 1.08 1.10];   // drawdown
9  for i=1:25
10     tday(i)=tmin(i)/(60*24);
11 end
12 // graph is plotted between s and t
13 // point P is choosen on it whose ordinate is:
14 s1=0.45;
15 t=0.00347;
16 ds=0.38;                // for one log cycle of time
17 Fu=s1/ds;
18 // from fig 5.43
19 // or using relation
20 Wu=2.303*Fu;
21 u=0.035;        // from table 5.2
22 Q=3600;              // discharge in cumec/day
23 T=Q*Wu/(4*%pi*s1);
24 S=4*u*t*T/r^2;
25 mprintf(" formation constant of acquifer:");
26 mprintf("\nT=%i cubic metre/day/m.\nS=%f.",T,S);
```

Scilab code Exa 5.16 EX5 16

```
1
2
3  // example 5.16
4  // calculate transmissibility of acquifer
```

148

```
5  //draw daown in main well
6  clc;
7  //given
8  H=25;              //static water level
9  rw=0.15;           //radius of well
10 Q=5400;            //discharge(litre/min)
11 t=24;              //time of discharge
12 r1=30;              //distance of first well
13 s1=1.11;           //drawdown
14 h1=H-s1;
15 r2=90;             //distance of second well
16 s2=0.53;           //drawdown
17 h2=H-s2;
18 k=(Q*2.303*log10(r2/r1))/(%pi*(h2^2-h1^2)*60000);
19 T=k*H;
20 T=round(T*10000)/10000;
21 mprintf("transmissibility of acquifer=%f cumec/sec."
       ,T);
22 hw=(h2^2-(Q*2.303*log10(r2/rw))/(%pi*k*60000))^0.5;
23 sw=H-hw;
24 sw=round(sw*100)/100;
25 printf("\ndraw daown in main well=%f m.",sw);
```

Scilab code Exa 5.17 EX5 17

```
1
2
3  //example 5.17
4  //calculate discharge at 18m drawdown
5  clc;
6  //given
7  Q=250;              //discharge(lit/min)
8  H=100;              //water level in acquifer
9  s1=12;              //drawdown
10 h1=H-s1;
```

```
11  // let  t=ln(R/r)/(pi*k)
12  t=(H^2-h1^2)/Q;
13  h2=H-18;
14  Q1=(H^2-h2^2)/t;
15  mprintf(" discharge  at  18m  drawdown=%i  lpm",Q1);
```

Scilab code Exa 5.18 EX5 18

```
1
2
3  // example  5.18
4  // calculate  effective  well  diameter
5  clc;
6  // given
7  b=10;                    // thickness  of  acquifer
8  k=48;                    // permeability  coefficient
9  R=500;                   // radius  of  influence
10 s=12;                    // drawdown
11 Q=5000;                  // discharge ( cumec/day )
12 r=R/%e^(2*%pi*b*k*s/Q);
13 D=2*r;
14 D=round(D*100)/100;
15 mprintf(" effective  well  diameter=%f m.",D);
```

Scilab code Exa 5.19 EX5 19

```
1
2
3  // example  5.19
4  // calculate  drawdown  at  40m
5  clc;
6  // given
7  Q=1500;                  // discharge ( lit /min)
```

```
 8  S=0.004;            //storage coefficient
 9  k=35;               //permeability
10  t=20;               //time of pumping
11  b=30;               //thickness of acquifer
12  r=40;               //distance of observation well
13  T=k*b;
14  s=Q*(2.303*log10(4*T*t*3600/(60*60*24*r^2*S))
       -0.5772)*60*60*24/(4*%pi*T*60000);    //Jacob's
       equation
15  s=round(s*100)/100;
16  mprintf("drawdown at 40m=%f m.",s);
```

Scilab code Exa 5.20 EX5 20

```
 1
 2
 3  //example 5.20
 4  ////calculate
 5  //yield from well
 6  clc;
 7  //given
 8  h1=2.5;                   //initial pumping
        depression
 9  h=1.8;                    //heigth after recuperation
10  t=80;                     //time
11  h2=h1-h;
12  KbyA=2.303*60*log10(h1/h2)/t;
13  d=4;           //diameter of well
14  H=3;           //depression head
15  A=%pi*d^2/4;
16  Q=(KbyA)*A*H/3.6;
17  Q=round(Q);
18  mprintf("\nYield from well=%f lit/sec.",Q);
```

Scilab code Exa 5.21 EX5 21

```scilab
1
2
3  //example 5.21
4  //calculate transmissibility
5  //drawdown at pumping well
6  clc;
7  //given
8  rw=0.15;              //radius of well
9  b=40;                //depth of acquifer
10 Q=1500;              //discharge(lpm)
11 s1=3.5;              //drawdown of first well
12 s2=2;                //drawdown of second well
13 H=40;
14 r1=25;               //distance of first well
15 r2=75;               //distance of second well
16 h1=H-s1;
17 h2=H-s2;
18 k=Q*2.303*log10(r2/r1)/(%pi*1000*60*(h2^2-h1^2));
19 T=b*k*1000;
20 mprintf("transmissibility=%fD-3 square metre/sec",T)
       ;
21
22 hw=(h2^2-(Q*2.303*log10(r2/rw)/(%pi*k*60000)))^0.5;
23 sw=H-hw;
24 sw=round(sw*100)/100;
25 mprintf("\ndrawdown at pumping well=%f m.",sw);
```

Scilab code Exa 5.22 EX5 22

```scilab
1
```

```
 2
 3  //example 5.22
 4  //calculate coefficient of permeability
 5  //drawdown in test well
 6  clc;
 7  //given
 8  r=0.25;          //radius of test well
 9  r1=10;           //distance of first well
10  r2=60;           //distance of second well
11  Q=0.1;           //discharge(cumec/sec)
12  s1=4;            //drawdown of first well
13  s2=3;            //drawdown of second well
14  b=20;            //thickness of well
15  k=1000*Q*log10(r2/r1)/(2.72*b*(s1-s2));
16  mprintf("coefficient of permeability=%fD-3 m/sec",k)
       ;
17  s=s2+Q*log10(r2/r)/(2.72*b*k);
18  s=round(s*100)/100;
19  mprintf("\ndrawdown in test well=%f m.",s);
```

Scilab code Exa 5.23 EX5 23

```
 1
 2
 3  //example 5.23
 4  //calculate
 5  //diameter of well
 6  clc;
 7  //given
 8  h1=2.1;                    //initial pumping
       depression
 9  h=1.6;                     //heigth after recuperation
10  t=90;                      //time
11  h2=h1-h;
12  KbyA=2.303*60*log10(h1/h2)/t;
```

```
13  Q=10;                  //yield(lit/sec)
14  H=2;
15  A=Q*3.6/(H*(KbyA));
16  d=(4*A/%pi)^0.5;
17  d=round(d*10)/10;
18  mprintf("\nDaimeter of well=%f m",d);
```

Scilab code Exa 5.24 EX5 24

```
1
2
3  //example 5.24
4  //calculate yield from well
5  clc;
6  //given
7  h1=2.5;                       //initial pumping
        depression
8  h=1;                      //heigth after recuperation
9  t=60;                        //time
10 h2=h1-h;
11 KbyA=2.303*60*log10(h1/h2)/t;
12 d=2;           //diameter of well
13 H=3;           //depression head
14 A=%pi*d^2/4;
15 Q=(KbyA)*A*H;
16 Q=round(Q*1000)/1000;
17 mprintf("\nYield from well=%f cubic metre/hour.",Q);
```

Chapter 6

RESERVIOR PLANNING

Scilab code Exa 6.1 EX6 1

```
1
2
3  //exampple 6.1
4  //determine maximum reservior level
5  //maximum discharge over spillway
6  //plot inflow and routed hydrograph and find peak
       flow and peak lag
7  clc;funcprot(0);
8  //given
9  e=[100 100.3 100.6 100.9 101.2 101.5 101.8 102.1
       102.4 102.7];      //elevation(km)
10 A=[405 412 420 425 428 436 445 453 460 469];
                //area
11 o=[0 14.9 42.2 77.3 119 169 217 272 334 405];
                //outflow
12 c(1)=0;
13 for i=2:10
14     dh(i)=e(i)-e(i-1);
15     s(i)=dh(i)/3*(A(i-1)+A(i)+(A(i-1)*A(i))^0.5);
                //storage between contours
16     c(i)=c(i-1)+s(i);                          //
```

```
                cumulative storage
17      h(i)=c(i)/1.08;                                //2s/t
18      h1(i)=h(i)-o(i);                               //2s/t-
            o
19      h2(i)=h(i)+o(i);                               //2s/t+
            o
20  end
21  T=[0:6:102];
22  I=[42 45 57 88 147 210 272 340 350 338 314 288 263
        240 198 170 143 120];  //inflow
23  h4=[0 0 60 122 185 266 362 455 545 605 623 620 600
        575 550 515 470 430];    //2s/t-0 obtained from
        curve a
24  O=[0 10 24 42 74 130 194 260 316 334 328 312 286 264
        236 204 177 150];        //outflow read from curve
        a
25  re=[100.2 100.39 100.58 100.86 101.26 101.65 102.03
        102.31 102.4 102.37 102.3 102.18 102.06 101.9
        101.72 101.56 102.4];   //reservior elevation
        read from curve b
26  for i=2:17
27      t(i)=I(i-1)+I(i);                             //I1+I2
28      h3(i)=t(i)+h4(i);                             //2s/t+
            O
29  end
30  pt=T(10)-T(9);
31  d=I(9)-O(10);
32  //results
33  mprintf(" maximum reservior level=%f m.",re(10));
34  mprintf("\nmaximum discharge over spillway=%f cumecs
        .",O(10));
35  mprintf("\nreduction in peak discharge=%f cumecs.",d
        );
36  mprintf("\npeak lag=%f hours.",pt);
```
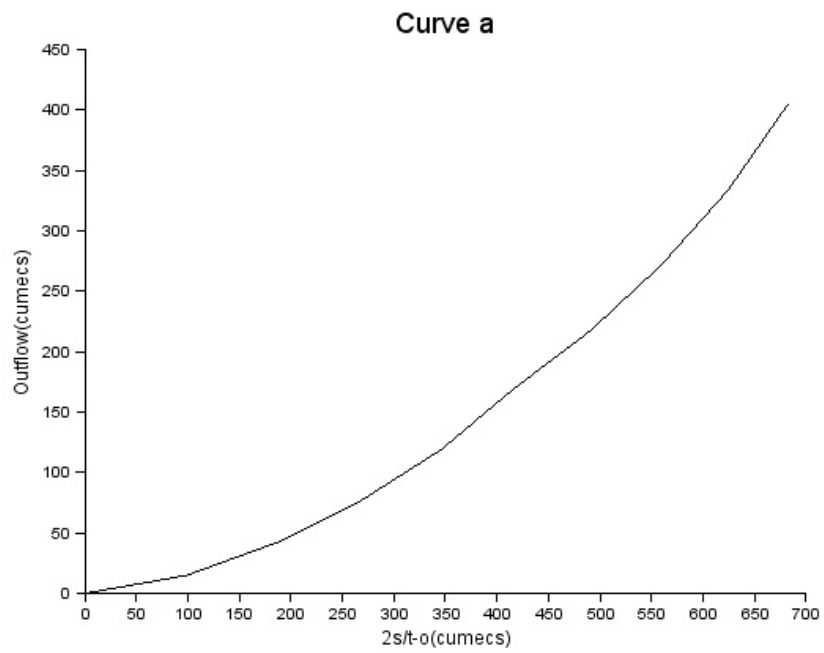
Figure 6.1: EX6 1
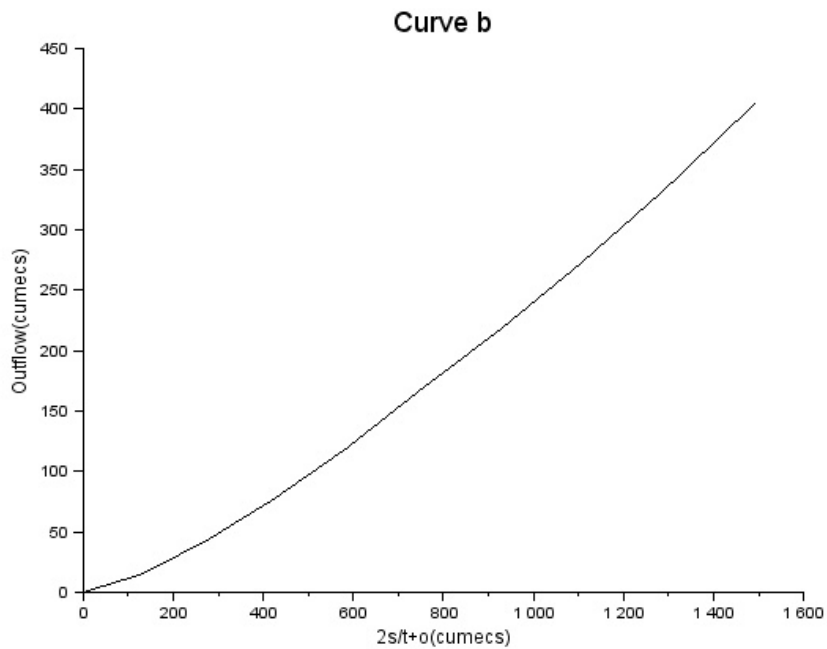
Figure 6.2: EX6 1

Scilab code Exa 6.2 EX6 2

```
1
2
3  //example6.2
4  //calculate required useful storage
5  clc;funcprot(0);
6  //given
7  in=[8.6 2.2 1.8 0 0 13.5 280.6 510.2 136 52.5 20.6
        12.3];         //inflow(ha-m)
```

```
 8  pan=[2.2 2.3 3.1 8.6 12.8 15.6 12.3 10.6 10 8.2 5.8
        3];           //pan evaporation
 9  p=[0.8 1.2 0 0 0 4.8 12.2 18.6 8.6 1.5 0 0]
                              //precipitation
10  D=[14.5 15.8 16.2 16.8 17.5 18 18 17 16.5 16 15.8
        15];            //Demand
11  s=0;
12  for i=1:12
13      if in(i)<10 then
14          r(i)=in(i);
                                                              //D
              /S requirement
15      else
16          r(i)=10;
17      end
18      E(i)=3.6*pan(i);
                                                      //
            Evaporation over reservior area
19      P(i)=3.5*p(i);
                                                        //
            Precipitation
20      I(i)=in(i)-r(i)-E(i)+P(i);
                                          //Adjusted inflow
21      S(i)=D(i)-I(i);
                                                        //Water
          required from storage
22      if S(i)<0 then
23          S(i)=0;
24      end
25      s=s+S(i);
26  end
27  mprintf("required useful storage=%f ha-m.",s);
```
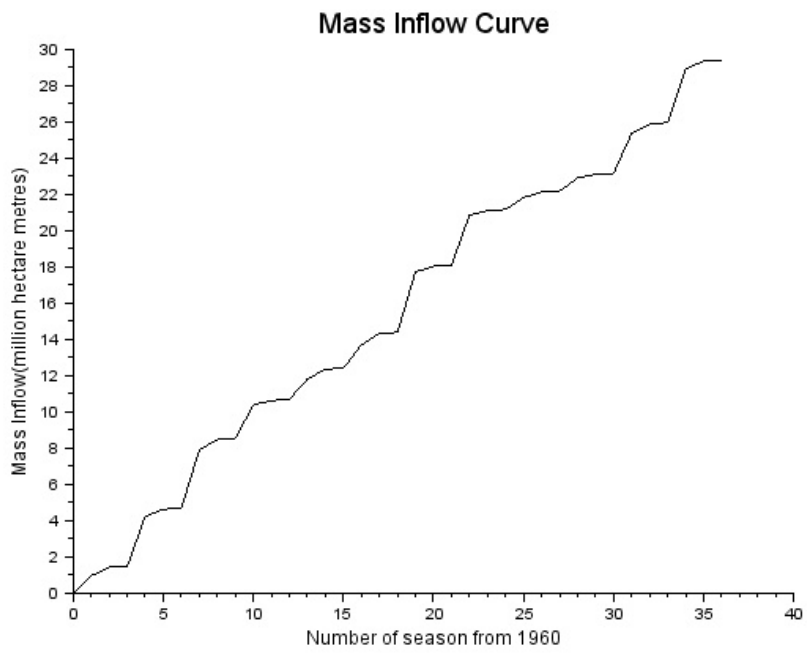
Figure 6.3: EX6 3

Scilab code Exa 6.3 EX6 3

```
1
2
3  //example  6.3
4  //calculate  storage  capacity  of  reservior
5  clc;funcprot(0);
6  //given
7  V=475;              //flow  required  to  be  maintained
       throughout  the  year
8  Y=V*365*8.64;       //yearly  demand
9  //yearly  demand  gives  the  slope  of  demand  curve
10 t=[0:1:36];              //number  of  season  startin
       from  1960;each  year  is  diveded  into  3  seasons.
11 q=[0  1050  300  50  3000  250  40  3500  370  90  2000  150
       120  1200  350  65  1400  400  100  3600  200  80  3000  200
        80  3000  150  120  700  210  50  800  120  80  2400  320
       120  3200  280  80];    //average  discharge
12 v=[0  0.9707  0.4717  0.0328  2.7734  0.3981  0.0263
       3.2357  0.5818  0.0591  1.8490  0.2356  0.0788  1.1094
       0.5504  0.0427  1.2943  0.6290  0.0657  3.3281  0.3145
       0.0525  2.7734  0.2359  0.0788  0.6441  0.3302  0.028
       0.7396  0.1887  0.0525  2.2188  0.5032  0.0788  2.9583
       0.4403  0.0525];          //voloume
13 cv(1)=v(1);
14 for  i=2:37
15     cv(i)=cv(i-1)+v(i);
16 end
17 //each  year  is  divided  into  three  seasons(monsoon,
       winter  and  summer).and  readings  are  taken  for  12
       years
18 //mass  inflow  curve  is  plotted  and  tangent  are  drawn
        at  the  apexes  and  parellel  to  demand  curve  slope
       ;
19 //the  respectiv  ordinates  represent  the  deficiency
       during  dry  period
20 //maximum  of  these  ordinates  gives  the  desired
       reservior  capacity
```

```
21  mprintf ( " storage  capacity  of  reservior =1.6  million
        ha−m . " ) ;
```

Scilab code Exa 6.4 EX6 4

```
 1
 2
 3  // example  6.4
 4  // calculate  probable  life  of  reservior
 5  clc ; funcprot (0) ;
 6  // given
 7  asi =3.6;                                    // annual
        sediment  inflow ( x10 ^6)
 8  gamma_s =12;                                 // specific
        weigth  of  sediment
 9  vs = asi /12;
10  ir =30;                                      // initial
        reservior  capacity
11  fr =60;                                      // final
        reservior  capacity
12  r = ir / fr ;                                // initial
        capacity / inflow  ratio
13  // r =0.5;  hence  we  start  capacity / inflow  ratio  from
        0.5
14  c = [0.5: -0.1:0.1];                         // capacity
        inflow  ratio
15  e = [0.96  0.955  0.95  0.93  0.87];         // trap
        efficiency
16  for  i =1:4
17      ae ( i ) = ( e ( i ) + e ( i +1) ) /2;   // average
            efficiency  for  interval
18  end
19  as = [0.2872  0.2857  0.2820  0.2700];       // annual
        sediment  trapped
20  s =0;
```

162

```
21 for i=1:4
22     y(i)=6/as(i);                        //year to fill
23     s=s+y(i);
24 end
25 mprintf(" probable life of reservior=%i years.",s);
```

Scilab code Exa 6.5 EX6 5

```
1
2
3 //example 6.5
4 //calculate maximum outflow discharge over spillway
5 //corresponding maximum level of water above
       spillway crest
6 clc;funcprot(0);
7 //given
8 I=[60 480 900 470 270 160 110 80 60];    //inflow
9 //for the first time interval 0 hours to 3 hours
10 I1=I(1);
11 I2=I(2);
12 t=3*3600;
13 ti=(I1+I2)*t/2;                 //total inflow
14 //outflow=1.62*h1^1.5;
15 //storage change=(30+3h1)h1
16 //from the basic equation i.e total inflow=total
       outflow+change in storage
17 //on solving we get
18 //h1^2+0.54h1^1.5+10h1-0.972=0;
19 //solving it by trial and error method;we get
20 h1=0.0954;
21 //for the second time interval 3 hours to 6 hours
22 I1=I(2);
23 I2=I(3);
24 t=3*3600;
25 ti=(I1+I2)*t/2;                     //total inflow
```

```
26  // outflow =0.0477+1.62* h2 ^ 1.5;
27  // storage  change=(30+3h2)h2
28  // from  the  basic  equation  i.e  total  inflow=total
        outflow+change  in  storage
29  // on  solving  we  get
30  // h2 ^2+0.54 h2 ^1.5+10h2 −3.4312=0;
31  // solving  it  by  trial  and  error  method ; we  get
32  h2 =0.323;
33  // for  the  third  time  interval  6  hours  to  9  hours
34  I1=I (3);
35  I2=I (4);
36  t =3*3600;
37  ti =(I1+I2)*t/2;                      // total  inflow
38  // outflow =0.2974+1.62* h3 ^ 1.5;
39  // storage  change=(30+3h3)h3
40  // from  the  basic  equation  i.e  total  inflow=total
        outflow+change  in  storage
41  // on  solving  we  get
42  // h3 ^2+0.54 h3 ^1.5+10h3 −5.7012=0;
43  // solving  it  by  trial  and  error  method ; we  get
44  h3 =0.522;
45  // for  the  fourth  time  interval  9  hours  to  12  hours
46  I1=I (4);
47  I2=I (5);
48  t =3*3600;
49  ti =(I1+I2)*t/2;                      // total  inflow
50  // outflow =0.611+1.62* h4 ^ 1.5;
51  // storage  change=(30+3h4)h4
52  // from  the  basic  equation  i.e  total  inflow=total
        outflow+change  in  storage
53  // on  solving  we  get
54  // h4 ^2+0.54 h4 ^1.5+10h4 −6.6208=0;
55  // solving  it  by  trial  and  error  method ; we  get
56  h4 =0.601;
57  // for  the  fifth  time  interval  12  hours  to  15  hours
58  I1=I (5);
59  I2=I (6);
60  t =3*3600;
```

```
61  ti=(I1+I2)*t/2;                    //total  inflow
62  //outflow=0.7548+1.62*h5^1.5;
63  //storage  change=(30+3h5)h5
64  //from  the  basic  equation  i.e  total  inflow=total
       outflow+change  in  storage
65  //on  solving  we  get
66  //h5^2+0.54h5^1.5+10h5-6.8936=0;
67  //solving  it  by  trial  and  error  method;we  get
68  h5=0.624;
69  //for  the  sixth  time  interval  12  hours  to  15  hours
70  I1=I(6);
71  I2=I(7);
72  t=3*3600;
73  ti=(I1+I2)*t/2;                    //total  inflow
74  //outflow=0.7985.62*h6^1.5;
75  //storage  change=(30+3h6)h6
76  //from  the  basic  equation  i.e  total  inflow=total
       outflow+change  in  storage
77  //on  solving  we  get
78  //h6^2+0.54h6^1.5+10h6-6.8492=0;
79  //solving  it  by  trial  and  error  method;we  get
80  h6=0.620;
81  hmax=h5;
82  q=300*(h5)^1.5;           //equation  given
83  q=round(q*100)/100;
84  mprintf("maximum  outflow  discharge  over  spillway=%f
       cumecs.",q);
85  mprintf("\nmaximum  level  of  water  above  spillway
       crest=%f m.",h5);
```

Scilab code Exa 6.6 EX6 6

```
1
2
3  //example  6.6
```

```
4  //calculate the allocations to each project purpose
5  clc;funcprot(0);
6  //given
7  t=240;                 //total cost of project(million
      rupees)
8  s=[32 88 72];      //separable cost
9  eb=[40 138 112];    //estimated benifit
10 sp=[47 104 101];    //alternate single purpose cost
11 //using remaining benifit method
12 ts=s(1)+s(2)+s(3); //total separable cost
13 tj=t-ts;             //total joint cost
14 w=0;
15 for i=1:3
16     if eb(i)<sp(i) then
17         b(i)=eb(i);               //benifit limited by
                alternate cost
18     else
19         b(i)=sp(i);
20     end
21    rb(i)=b(i)-s(i);               //remaining benifit
22    w=w+rb(i);
23
24 end
25 y=0;
26 for i=1:3
27     aj(i)=tj*rb(i)/w;            //allocated joint
            cost
28     ta(i)=s(i)+aj(i);            //total allocations
29     y=y+ta(i);
30 end
31 mprintf("Using remaining benifit method.");
32 mprintf("\n\nallocations to each project purpose(
      percent):");
33 for i=1:3
34    per(i)=ta(i)*100/y;               //total
            allocation percent
35    mprintf("\n%f",per(i));
36 end
```

```
37
38
39  //using alternate justifiable method
40  w=0;
41  for i=1:3
42      ac(i)=sp(i)-s(i);              //alternate cost
            less separable cost
43      w=w+ac(i);
44
45  end
46  y=0;
47  for i=1:3
48      ajc(i)=tj*ac(i)/w;            //allocated joint
            cost
49      ta(i)=s(i)+ajc(i);           //total allocation
50    y=y+ta(i);
51  end
52  mprintf("\n\nUsing alternate justifiable expenditure
        method method.");
53  mprintf("\n\nallocations to each project purpose(
        percent):");
54  for i=1:3
55      pr(i)=ta(i)*100/y;           //total allocation
            percent
56  mprintf("\n%f",pr(i));
57  end
```

Scilab code Exa 6.8 EX6 8

```
1
2
3  //example 6.8
4  //calculate outflow hydrograph
5  clc;funcprot(0);
6  //given
```

```scilab
7  I =[35 55 92 130 160 140];    //inflow (cumec/sec)
8  x=0.28;K=1.6;                  //studied value
9  t=6;
10 K=K*24;                        //in hours
11 co=(-K*x+0.5*t)/(K-K*x+0.5*t);
12 c1=(K*x+0.5*t)/(K-K*x+0.5*t);
13 c2=(K-K*x-0.5*t)/(K-K*x+0.5*t);
14 c=co+c1+c2;
15 //c=1; which implies (OK)
16 //from Muskingum equation
17 O(1)=35;
18 mprintf("outflow hydrograph:\n%f",O(1));
19 for i=2:6
20     p1(i)=co*I(i);
21     p2(i)=c1*I(i-1);
22     p3(i)=c2*O(i-1);
23     O(i)=p1(i)+p2(i)+p3(i);
24     O(i)=round(O(i)*100)/100;
25    mprintf("\n%f",O(i));
26 end
```

Scilab code Exa 6.9 EX6 9

```scilab
1
2
3  //example 6.9
4  //calculate minimum storage to meet the demand
5  clc;funcprot(0);
6  //given
7  md=[50 75 80 85 130 120 25 25 40 45 50 60];      //
       monthly demand
8  e=[6 8 13 17 22 22 14 11 13 12 7 5];             //
       evaporation
9  r=[1 0 0 0 0 19 43 39 22 6 2 1];                 //
       rainfall
```

```
10  in =[50 40 30 25 20 30 200 225 150 90 70 60];           //
        monthly  inflow
11  A =30;                                                    //
        area  of  reservior
12  Cr =0.4;                                                  //
        run−off  coefficient
13  for  i =1:12
14      er(i)=0.4*r(i);                                       //
            effective  rainfall
15      ni(i)=er(i)-e(i);                                     //
            net  inflow
16      niv(i)=ni(i)*0.01*A;                                  //
            net  inflow  volume
17      nd(i)=md(i)-niv(i);                                   //
            net  demand
18  end
19  cnd(1)=nd(1);                                             //
        cumulative  demand
20  ci(1)=in(1);                                              //
        cumulative  inflow
21  for  i =2:12
22      cnd(i)=cnd(i-1)+nd(i);
23      ci(i)=ci(i-1)+in(i);
24  end
25  mprintf("Excess  demand:");
26  for  i =1:12
27      ed(i)=cnd(i)-ci(i);                                   //
            excess  demand
28      if  ed(i)<0  then
29          es(i)=ed(i);                                      //
                excess  supply
30          ed(i)=0;
31      end
32      mprintf("\n%f",ed(i));
33  end
34  mprintf("\nminimum  storage  required=Maximum  of
        excess  demand=%f Mm^3.",ed(6));
```

Scilab code Exa 6.10 EX6 10

```
1
2
3  //example 6.10
4  //calculate
5  //minimum capacity of reservior
6  //the initial storage storage required to maintain
       uniform demand
7  clc;funcprot(0);
8  //given
9  in=[2.83 4.25 5.66 18.4 22.64 22.64 19.81 8.49 7.1
       7.1 5.66 5.66];      //inflow(x10^5)
10 s=0;
11 for i=1:12
12     s=s+in(i);
13 end
14 avd=s/12;

       //average demand(x10^5)
15 s=0;t=0;
16 for i=1:12
17     e(i)=avd-in(i);
18     if e(i)<0 then
19          S(i)=-e(i);

              //surplus(x10^5)
20          s=s+S(i);
21     else
22          D(i)=e(i);

              //Deficit(x10^5)
23          t=t+D(i);
24     end
```

170

```
25  end
26
27  d=(s-(t-D(1)-D(2)-D(3)));
28  s=s;
29
30  mprintf("minimum  capacity  of  reservior=%fD+5  cumec."
         ,s);
31  mprintf("\nstorage  required  to  maintain  uniform
         demand=%fD+5  cumec",d);
```

Chapter 8

GRAVITY DAMS

Scilab code Exa 8.1 EX8 1

```
1
2
3  // example  8.1
4  // calculate  forces  induced  due  to  earthquake
5  clc ; funcprot (0) ;
6  // given
7  H =100;                  // heigth  of  dam
8  wb =70;                  // width  of  base  of  dam
9  wt =7;                   // width  of  top  of  dam
10 l =1;                    // length  of  dam
11 hw =98;                  // heigth  of  water  in  dam
12 hsu =90;                 // heigth  of  slope  on  downstream
       side
13 s =1/0.7;                // slope  on  downstream  side
14 gammad =24;              // unit  weigth  of  dam
15 gammaw =9.81;            // unit  weigth  of  water
16 E =2.05 D7 ;             // modulus  of  elasticity
17
18 // ( a )  inertial  forces  and  moments
19 alpha0 =0.05;            // from  table  8.1
20 alphah =2* alpha0 ;
```

```
21  //at 10m from top
22  F10=integrate('25.2-0.25*y','y',0,10);
23  M10=integrate('25.2*(1-0.01*y)*(10-y)','y',0,10);
24  //at 100m below top
25  F100=F10+integrate('0.15*(1-0.01*y)*16.8*y','y'
        ,10,100);
26  M100=M10+90*F10+integrate('0.15*(1-0.01*y)*16.8*y
        *(100-y)','y',10,100);
27  mprintf("Inertial forces:\nAt 10m from top: F=%f kn;
        M=%ikn-m\nAt 100m from top: F=%f kn;M=%ikn-m.",
        F10,M10,F100,M100);
28
29  //(b) hydrodynamic pressure and moment
30  //at 10m from top
31  y=8;
32  W10=1680;
33  alphah=F10/W10;
34  Cm=0.735;
35  Cy=(Cm/2)*((y*(2-y/hw)/hw)+(y*(2-y/hw)/hw)^0.5);
36  p=Cy*alphah*gammaw*hw;
37  P10=0.726*p*y;
38  Mp10=0.299*p*y^2;
39  P10=round(P10*100)/100;
40  Mp10=round(Mp10*100)/100;
41  //at 100m from top
42  y=98;
43  W100=84840;
44  alphah=F100/W100;
45  Cm=0.735;
46  Cy=(Cm/2)*(y*(2-y/hw)/hw+(y*(2-y/hw)/hw)^0.5);
47  p=Cy*alphah*gammaw*hw;
48  P100=0.726*p*y;
49  Mp100=0.299*p*y^2;
50  mprintf("\nHydrodynamic forces:\nAt 10m from top: F=
        %f kn;M=%fkn-m\nAt 100m from top: F=%i kn;M=%ikn-
        m.",P10,Mp10,P100,Mp100);
```

Scilab code Exa 8.2 EX8 2

```
1
2
3  //example 8.2
4  //calculate forces induced due to earthquake by
      responce spectrum method
5  clc;funcprot(0);
6  //given
7  H=100;                  //heigth of dam
8  wb=70;                  //width of base of dam
9  wt=7;                   //width of top of dam
10 l=1;                    //length of dam
11 hw=98;                  //heigth of water in dam
12 hsu=90;                 //heigth of slope on downstream
      side
13 s=1/0.7;                //slope on downstream side
14 gammad=24;              //unit weigth of dam
15 gammaw=9.81;            //unit weigth of water
16 E=2.05D7;               //modulus of elasticity
17 beta=1;
18 I=2;
19 Fo=0.25;                //from table 8.2
20                         //t=Sa/g;
21 t=0.19;                 //from fig. 8.4
22 alphah=beta*I*Fo*t;
23 T=5.55*H^2/wb*(gammad/(gammaw*E))^0.5;
24 //(a) Base shear
25 W=l*gammad*(wt*H+((hsu/s)*hsu)/2);
26 Fb=0.6*W*alphah;
27 mprintf("Base shear=%f KN.",Fb);
28
29 //(b) Base moment
30 hbar=((wt*H^2/2)+((hsu/s)*hsu^2/6))/((wt*H)+(hsu/s)*
```

174

```
        hsu/2);
31  Mb=0.9*W*hbar*alphah;
32  mprintf("\nBase moment=%f KN−m.",Mb);
33
34  //(c) shear at 10m from top
35  Cv=0.08;
36  F10=Cv*Fb;
37  F10=round(F10);
38  mprintf("\nshear at 10m from top=%f KN.",F10);
39
40  //(d) Moment at 10m from top
41  Cm=0.02;
42  M10=Cm*Mb;
43  M10=round(M10);
44  mprintf("\nmoment at 10m from top=%f KN.",M10);
45  //(e) Hydrodynamic pressure
46  //at 10m from top
47  y=8;
48  W10=1680;
49  Cm=0.735;
50  Cy=(Cm/2)*((y*(2-y/hw)/hw)+(y*(2-y/hw)/hw)^0.5);
51  p=Cy*alphah*gammaw*hw;
52  P10=0.726*p*y;
53  Mp10=0.299*p*y^2;
54  P10=round(P10*100)/100;
55  Mp10=round(Mp10*100)/100;
56  //at 100m from top
57  y=98;
58  W100=84840;
59  Cm=0.735;
60  Cy=(Cm/2)*(y*(2-y/hw)/hw+(y*(2-y/hw)/hw)^0.5);
61  p=Cy*alphah*gammaw*hw;
62  P100=0.726*p*y;
63  Mp100=0.299*p*y^2;
64  mprintf("\nHydrodynamic forces:\nAt 10m from top: F=
        %f kn;M=%fkn−m\nAt 100m from top: F=%i kn;M=%ikn−
        m.",P10,Mp10,P100,Mp100);
```

Scilab code Exa 8.3 EX8 3

```scilab
1
2
3  //example 8.3
4  //calculate forces induced due to earthquake
5  clc;funcprot(0);
6  //given
7  H=100;                    //heigth of dam
8  wb=70;                    //width of base of dam
9  wt=7;                     //width of top of dam
10 l=1;                      //length of dam
11 hw=98;                    //heigth of water in dam
12 hsu=90;                   //heigth of slope on downstream
       side
13 s=1/0.7;                  //slope on downstream side
14 gammad=24;                //unit weigth of dam
15 gammaw=9.81;              //unit weigth of water
16 E=2.05D7;                 //modulus of elasticity
17 //(a) Seismic coefficient method
18 alpha0=0.05;              //from table 8.1
19 alphah=2*alpha0;
20 alphav=0.75*alphah;
21 //at 10m from top
22 F10=integrate('alphav*168*(1-0.01*y)','y',0,10);
23 //at 100m below top
24 F100=F10+integrate('alphav*(1-0.01*y)*16.8*y','y'
       ,10,100);
25 mprintf("Part(a):\nAt 10m from top: F=%f kn\nAt 100m
       from top: F=%f kn.",F10,F100);
26
27 //(b)Response spectrum method
28 beta=1;
29 I=2;
```

176

```
30  Fo=0.25;                 //from table 8.2
31                           //t=Sa/g;
32  t=0.19;                  //from fig. 8.4
33  alphah=beta*I*Fo*t;
34  alphav=0.75*alphah;
35  //at 10m from top
36  F10=integrate('alphav*168*(1-0.01*y)','y',0,10);
37  //at 100m below top
38  F100=F10+integrate('alphav*(1-0.01*y)*16.8*y','y'
       ,10,100);
39  F100=round(F100*100)/100;
40  mprintf("\nPart(b):\nAt 10m from top: F=%f kn\nAt
       100m from top: F=%f kn.",F10,F100);
```

Scilab code Exa 8.4 EX8 4

```
1
2
3  //example 8.4
4  //calculate hydrodynamic pressure on10m,40m and 100m
       from top
5  clc;funcprot(0);
6  //given
7  H=100;                  //heigth of dam
8  wb=73;                  //width of base of dam
9  wt=7;                   //width of top of dam
10  l=1;                    //length of dam
11  hw=98;                  //heigth of water in dam
12  hsu=90;                 //heigth of slope on downstream
       side
13  s=1/0.7;                //slope on downstream side
14  gammad=24;              //unit weigth of dam
15  gammaw=9.81;            //unit weigth of water
16  E=2.05D7;               //modulus of elasticity
17
```

```
18  //at 10m from top
19  y=8;
20  alphah=0.1;
21  Cm=0.72;
22  Cy=(Cm/2)*((y*(2-y/hw)/hw)+(y*(2-y/hw)/hw)^0.5);
23  p10=Cy*alphah*gammaw*hw;
24  F10=0.726*p10*y;
25  Mp10=0.299*p10*y^2;
26
27  //at 40m from top
28  y=38;
29  alphah=0.1;
30  Cm=0.72;
31  Cy=(Cm/2)*((y*(2-y/hw)/hw)+(y*(2-y/hw)/hw)^0.5);
32  p40=Cy*alphah*gammaw*hw;
33  F40=0.726*p40*y;
34  Mp40=0.299*p40*y^2;
35
36  //at 100m from top
37  y=98;
38  alphah=0.1;
39  Cm=0.72;
40  Cy=(Cm/2)*((y*(2-y/hw)/hw)+(y*(2-y/hw)/hw)^0.5);
41  p100=Cy*alphah*gammaw*hw;
42  F100=0.726*p100*y;
43  Mp100=0.299*p100*y^2;
44  p10=round(p10*1000)/1000;
45  F10=round(F10*1000)/1000;
46  Mp10=round(Mp10*10)/10;
47  p40=round(p40*1000)/1000;
48  F40=round(F40*1000)/1000;
49  Mp40=round(Mp40*10)/10;
50  p100=round(p100*100)/100;
51  F100=round(F100*1000)/1000;
52  Mp100=round(Mp100*10)/10;
53  mprintf("\nHydrodynamic Forces:\nAt 10m from top: P=
        %f KN/square m;F=%f KN;M=%f KN-m.\nAt 40m from
        top: P=%f KN/square m.;F=%f KN;M=%f KN-m.\nAt 100
```

```
        m from top: P=%f KN/square m;F=%f KN;M=%f KN–m.",
        p10,F10,Mp10,p40,F40,Mp40,p100,F100,Mp100);
54
55  //vertical component of reservior water on
        horizontal section
56  s1=3/60;
57  Wh=(F100-F40)*s1;
58  Wh=round(Wh*100)/100;
59  mprintf("\n\nvertical component of reservior water
        on horizontal section=%f kN/m.",Wh);
```

Scilab code Exa 8.8 EX8 8

```
 1
 2
 3  //example 8.8
 4  //calculate Heigth of dam when
 5  //no tension is permissible
 6  //factor of safety against slidingis 1.5
 7  clc;funcprot(0);
 8  //given
 9
10  wb=3;            //width of dam;
11  miu=0.5;         //coefficient of friction
12  Sg=2.4;          //specific gravity of masonary
13  gamma_w=9.81;    //unit weigth of water
14  c=1;
15
16  //when uplift is considered
17  //when no tension is permissible then e=wb/6;
18
19  p1=wb*Sg*gamma_w;
20  p2=c*wb*gamma_w/2;
21  p3=p1-p2;
22  p4=p1*wb/2-p2*2;
```

179

```
23  p5=gamma_w/6;
24  d1=p4/p3; d2=p5/p3;
25  d3=1.5-d1;
26  H=((0.5-d3)/d2)^0.5;
27  H=round(H*100)/100;
28  mprintf("when uplift is considered:")
29  mprintf("\nHeigth of dam when no tension is
        permissible=%f m.",H);
30  H=p3*0.5/(1.5*p5*3);
31  mprintf("\nHeigth of dam when factor of safety
        against sliding is 1.5=%f m.",H);
32
33  //when uplift is not considered
34  p1=wb*Sg*gamma_w;
35  p4=p1*wb/2;
36  p5=gamma_w/6;
37  d1=p4/p1;
38  d2=p5/p1;
39  H=(0.5/d2)^0.5;
40  H=round(H*100)/100;
41  mprintf("\n\nwhen uplift is not considered:")
42  mprintf("\nHeigth of dam when no tension is
        permissible=%f m.",H);
43  H=p1*0.5/(1.5*p5*3);
44  mprintf("\nHeigth of dam when factor of safety
        against sliding is 1.5=%f m.",H);
```

Scilab code Exa 8.9 EX8 9

```
1
2
3  //example 8.9
4  //calculate streeses at heel and toe of dam
5  clc;funcprot(0);
6  //given
```

```
 7  c=1;
 8  hw=6;                 //heigth  of  water  in  reservior
 9  Bt=1.5;                //width  of  top  of  dam
10  H=6;                 //heigth  of  the  dam
11  wb=4.5;                //width  of  base  of  dam
12  Sg=2.4;                //specific  gravity  of  masonary
13  gamma_w=9.81;          //weigth  density  of  water
14
15  W1=Bt*gamma_w*Sg*H;
16  W2=gamma_w*Sg*H*(wb-Bt)/2;
17  L1=(wb-Bt)+(Bt/2);
18  L2=(2*(wb-Bt))/3,
19  M1=W1*L1,M2=W2*L2,
20
21  //Reaervior  empty
22  SumW=W1+W2;
23  SumM=M1+M2;
24  x=SumM/SumW;
25  e=wb/2-x;
26  pnt=(SumW/wb)*(1+(6*e/wb));
27  pnh=(SumW/wb)*(1-(6*e/wb));
28  pnt=round(pnt*10)/10;
29  pnh=round(pnh*10)/10;
30  mprintf("Reservior  empty:");
31  mprintf("  \nNormal  stress  at  toe=%f  kN/square.m.",
        pnt);
32  mprintf("\nNormal  stress  at  heel=%f  kN/square.m.",
        pnh);
33
34  //Reservior  full
35  W3=gamma_w*H^2/2;
36  U=gamma_w*H*c*wb/2;
37  SumV=SumW-U;
38  L3=hw/3;
39  L4=2*wb/3;                //lever  arm
40  M3=W3*L3;
41  M4=U*L4;                  //moment  about  toe
42  SumM1=SumM-M4-M3;
```

```
43  x=SumM1/SumV;
44  e=wb/2-x;
45  pnt=(SumV/wb)*(1+(6*e/wb));
46  pnh=(SumV/wb)*(1-(6*e/wb));
47  pnt=round(pnt*10)/10;
48  pnh=round(pnh*10)/10;
49  mprintf("\n\nReservior full:");
50  mprintf(" \nNormal stress at toe=%f kN/square.m.",
        pnt);
51  mprintf("\nNormal stress at heel=%f kN/square.m.",
        pnh);
```

Scilab code Exa 8.10 EX8 10

```
1
2
3  //example 8.10
4  //calculate width of base if no tension is to
        develop
5  //check the stability
6  clc;funcprot(0);
7  //given
8  c=1;
9  hw=6;                  //heigth of water in reservior
10 Bt=1.5;                //width of top of dam
11 H=6;                   //heigth of the dam
12 gamma_m=20;            //unit weigth of masonary
13 gamma_w=9.81;          //weigth density of water
14 f=1800;                //compressive strength
15 miu=0.6;               //coefficient of friction
16
17 //to develop no tension e=b/6;x=b/3.
18 //hence on solving the relations we get
19
20 P=poly([-39.074 2.944 1],'b','c');     //equation is
```

```
         written  wrong  in  book
21  wb=roots(P);                              //sign  of
         coefficient  is  2.944  is  not  taken  correctly  in
         book
22
23
24  //roots  are  4.94  and  −7.89
25  //since  negative  value  cannot  be  taken
26
27  wb=4.94;
28  mprintf("Neglecting  the  negative  value.\nWidth  of
         base  is=4.94  m.");
29  W1=Bt*gamma_m*H;
30  W2=gamma_m*H*(wb-Bt)/2;
31  L1=(wb-Bt)+(Bt/2);
32  L2=(2*(wb-Bt))/3;
33  M1=W1*L1,
34  M2=W2*L2;
35  U=gamma_w*H*c*wb/2;
36  L4=2*wb/3;
37  M4=U*L4;
38  W3=gamma_w*H^2/2;
39  L3=hw/3;
40  M3=W3*L3;
41  SumW=W1+W2-U;
42  SumM=M1+M2-M4-M3;
43  pn=2*SumW/wb;
44  pn=round(pn*10)/10;
45  mprintf("\nMaximum  stress=%f  kN/square.m.",pn);
46  mprintf("\nDam  is  safe  against  compression");
47  FOS=miu*SumW/W3;
48  FOS=round(FOS*100)/100;
49  mprintf("\nFactor  of  safety  against  sliding=%f.  <1",
         FOS);
50  mprintf("\nDam  is  unsafe  against  sliding.");
```

183

Scilab code Exa 8.11 EX8 11

```
1
2
3 //example 8.11
4 //calculate width of base if no tension is to
       develop
5 //check the stability if uplift is neglected
6 clc;funcprot(0);
7 //given
8 c=1;
9 hw=6;                   //heigth of water in reservior
10 Bt=1.5;                //width of top of dam
11 H=6;                   //heigth of the dam
12 gamma_m=20;            //unit weigth of masonary
13 gamma_w=9.81;          //weigth density of water
14 f=1800;                //compressive strength
15 miu=0.6;               //coefficient of friction
16
17 //to develop no tension e=b/6;x=b/3.
18 //hence on solving the relations we get
19
20 P=poly([-19.908 1.5 1],'b','c')
21 wb=roots(P);
22
23 //roots are 3.774 and -5.27
24 //since negative value cannot be taken
25
26 wb=3.77;
27 mprintf("Neglecting the negative value.\nWidth of
       base is =3.77 m.");
28
29 W1=Bt*gamma_m*H;
30 W2=gamma_m*H*(wb-Bt)/2;
```

184

```
31  L1=(wb-Bt)+(Bt/2);
32  L2=(2*(wb-Bt))/3;
33  M1=W1*L1,
34  M2=W2*L2;
35  W3=gamma_w*H^2/2;
36  L3=hw/3;
37  M3=W3*L3;
38  SumW=W1+W2;
39  SumM=M1+M2-M3;
40  pn=2*SumW/wb;
41  pn=round(pn*10)/10;
42  mprintf("\nMaximum stress=%f kN/square.m.",pn);
43  mprintf("\nDam is safe against compression");
44
45  FOS=miu*SumW/W3;
46  FOS=round(FOS*1000)/1000;
47  mprintf("\nFactor of safety against sliding=%f. > 1"
        ,FOS);
48  mprintf("\nDam is safe against sliding.");
```

Scilab code Exa 8.12 EX8 12

```
1
2
3  //example8.12
4  // calculate maximum permissible heigth of shutter
        so that no tension develops
5  clc;funcprot(0);
6  //given
7  Bt=3;              //width of top of dam
8  H=12;             //heigth of the dam
9  wb=9;             //width of base of dam
10 gamma_m=21;          //unit weigth of masonary
11 gamma_w=9.81;    //weigth density of water
12
```

```
13  W1=Bt*gamma_m*H;
14  W2=gamma_m*H*(wb-Bt)/2;
15
16  //taking moment about a point on base at 3m from toe
17  L1=3+Bt/2;
18  L2=(2*(wb-Bt)/3)-3;
19  M1=W1*L1,
20  M2=W2*L2;
21  M=M1+M2;
22
23  //net moment about this point should be zero for
        equilibrium
24  s=(M*6/gamma_w)^(1/3)-12;
25  s=round(s*100)/100;
26  mprintf("maximum permissible heigth of shutter=%f m.
        ",s);
```

Scilab code Exa 8.13 EX8 13

```
1
2
3  //example 8.13
4  //calculate hydrodynamic earthquake pressure
5  //moment at 50m below water surface
6  clc;funcprot(0);
7  //given
8  c=1;
9  H=100;              //heigth of dam
10 hw=100;             //heigth of water in reservior
11 FB=1;               //free board
12 s=0.15;             //slope of upstream face
13 gamma_w=9.81;       //unit weigth of water
14 alphah=0.1;
15
16 theta=atan(s);
```

```
17  y=50;
18  Cm=0.735*(1-(theta*2/%pi));
19  Cy=(Cm/2)*((y*(2-y/hw)/hw)+(y*(2-y/hw)/hw)^0.5);
20  pe=Cy*alphah*gamma_w*hw;
21  F=0.726*pe*y;
22  M=0.299*pe*y^2;
23  pe=round(pe*1000)/1000;
24  F=round(F*10)/10;
25  M=round(M*10)/10;
26  mprintf("hydrodynamic  earthquake  pressure=%f kN/
        square.m\nshear=%f kN/m.\nMoment=%f kN-m/m.",pe,F
        ,M);
```

Scilab code Exa 8.14 EX8 14

```
 1
 2
 3  //example  8.14
 4  //check  stability
 5  //calculate  stresses  at   toe  and  heel
 6  clc;funcprot(0);
 7  //given
 8  c=1;
 9  H=10;                  //heigth  of  dam
10  hw=10;                 //heigth  of  water  in  reservior
11  wb=8.25;               //bottom  width
12  Bt=1;                  //top  width
13  Hs1=0.1;               //slope  on  upstream  side
14  gamma_w=9.81;          //unit  weigth  of  water
15  gamma_m=22.4;          //unit  weigth  of  masonary
16  f=1400;                //permissible  shear  stress  at
        joint
17  miu=0.75;              //coefficient  of  friction
18  fi=atan(0.625);
19  theta=atan(0.1);
```

```
20
21  W1=Bt*H*gamma_m;
22  W2=H*H*Hs1*gamma_m/2;
23  W3=H*6.25*gamma_m/2;
24  W4=hw*gamma_w*H*Hs1/2;
25  P=gamma_w*hw^2/2;
26  U=wb*gamma_w*hw*c/2;
27  SumV=W1+W2+W3+W4-U;
28  L3=2*(wb-(Hs1*H)-Bt)/3;
29  L1=(wb-(Hs1*H)-Bt)+Bt/2;
30  L2=(wb-(Hs1*H)-Bt)+Bt+(Hs1*H/3);
31  L4=(wb-(Hs1*H)-Bt)+Bt+(2*Hs1*H/3);
32  L5=2*wb/3;L6=hw/3;
33  M1=W1*L1;M2=W2*L2;M3=W3*L3;M4=W4*L4;
34  M5=U*L5;M6=P*L6;
35  SumM=M1+M2+M3+M4-M5-M6;
36  Mplus=M1+M2+M3+M4;
37  Mminus=M5+M6;
38  FOS=miu*SumV/P;
39  SFF=(miu*SumV+wb*1400)/P;
40  FOO=Mplus/Mminus;
41  FOS=round(FOS*100)/100;
42  SFF=round(SFF*10)/10;
43  FOO=round(FOO*100)/100;
44  mprintf("Factor of safety against sliding=%f. >1 ",
        FOS);
45  mprintf("\nShear friction factor=%f.",SFF);
46  mprintf("\nFactor of safety against overturning=%f.
        <1.5",FOO);
47  mprintf("\nDam is unsafe against overturning");
48
49  x=SumM/SumV;
50  e=wb/2-x;
51  p=hw*gamma_w;
52  pnt=(SumV/wb)*(1+(6*e/wb));        //calculation is
        done wrong in book;value of b is not taken
        correctly
53  pnh=(SumV/wb)*(1-(6*e/wb));
```

```
54  sigmat=pnt*sec(fi)^2;
55  sigmah=pnh*sec(theta)^2-p*tan(theta)^2;
56  taut=pnt*tan(fi);
57  tauh=-(pnh-p)*tan(theta);
58  pnt=round(pnt*10)/10;
59  pnh=round(pnh*10)/10;
60  sigmat=round(sigmat*10)/10;
61  sigmah=round(sigmah*10)/10;
62  taut=round(taut*10)/10;
63  tauh=round(tauh*10)/10;
64  mprintf("\n\nNormal stress at toe=%f kN/square.m.",
        pnt);
65  mprintf("\nNormal stress at heel=%f kN/square.m.",
        pnh);
66  mprintf("\nPrincipal stress at toe=%f kN/square.m.",
        sigmat);
67  mprintf("\nPrincipal stress at heel=%f kN/square.m."
        ,sigmah);
68  mprintf("\nShear stress at toe=%f kN/square.m.",taut
        );
69  mprintf("\nShear stress at heel=%f kN/square.m.",
        tauh);
```

Scilab code Exa 8.15 EX8 15

```
1
2
3  //example 8.15
4  //Check the stability and determine sliding factor
        and shear factor
5  clc; funcprot(0);
6  //Given
7  c=1;
8  miu=0.75;              //coefficient of friction
9  H=90;                  //heigth of dam
```

```
10  wb=73.1;                  //width of base
11  Bt=7;                     //width of top of dam
12  hw=89;                    //heigth of water in reservior
13  Hs1=28;                   //heigth of slope on upstream
        side
14  Hs2=83;                   //heigth of slope on downstream
        side
15  Cm=0.735;
16  alphah=0.1;
17  gamma_m=23.5;             //unit weigth of concrete
18  gamma_w=9.81;             //unit weigth of water
19  theta=atan(8/28);
20  fi=atan(0.7);
21  //self weigth of dam
22  W1=(Hs1*8*gamma_m)/2,
23  W2=(Bt*H*gamma_m),
24  W3=(Hs2^2*0.7*gamma_m)/2,
25  //weigth of superimposed water
26  W4=(Hs1*8*gamma_w)/2,
27  W5=(hw-Hs1)*8*gamma_w,
28  U=hw*wb*2*gamma_w/6;                      //uplift force
29  wp=hw^2*gamma_w/2;                        //water
        pressure
30  hp=0.726*Cm*alphah*gamma_w*hw^2;        //hydrodynamic
        pressure
31  Mhp=0.299*Cm*alphah*gamma_w*hw^3;      //moment due to
        hydrodynamic pressure
32  //inertial load due to horizontal acceleration
33  I1=W2/10;
34  I2=W3/10;
35  I3=W1/10;
36  SumV=W1+W2+W3+W4+W5-U;
37  SumH=wp+hp+I1+I2+I3;
38  L1=(wb-8)+8/3,
39  L2=(0.7*Hs2)+(Bt/2),
40  L3=(2*Hs2*0.7)/3,
41  L4=(wb-8)+(2*8)/3,
42  L5=(wb-8)+(8/2),
```

```
43  L6=hw/3;
44  L7=2*wb/3;
45  M1=W1*L1,M2=W2*L2,M3=W3*L3,M4=W4*L4;
46  M5=W5*L5;
47  M6=wp*L6;
48  M7=U*L7;
49  M8=I1*45;
50  M9=I2*83/3;
51  M10=I3*28/3;
52  Mplus=M1+M2+M3+M4+M5;
53  Mminus=M6+M7+M8+M9+M10+Mhp;
54  SumM=Mplus-Mminus;
55  x=SumM/SumV;
56  e=wb/2-x;
57  pnt=(SumV/wb)*(1+(6*e/wb));
58  pnh=(SumV/wb)*(1-(6*e/wb));
59  sigmat=pnt*sec(fi)^2;
60  p=hw*gamma_w;
61  pe=Cm*alphah*gamma_w*hw;
62  sigmah=pnh*sec(theta)^2-(p+pe)*tan(theta)^2;
63  taut=pnt*tan(fi);
64  tauh=-(-pnh-(p+pe))*tan(theta);
65  mprintf("Normal stress at toe=%i kN/square.m.",pnt);
66  mprintf("\nNormal stress at heel=%i kN/square.m.",
        pnh);
67  mprintf("\nPrincipal stress at toe=%i kN/square.m.",
        sigmat);
68  mprintf("\nPrincipal stress at heel=%i kN/square.m."
        ,sigmah);
69  mprintf("\nShear stress at toe=%i kN/square.m.",taut
        );
70  mprintf("\nShear stress at heel=%i kN/square.m.",
        tauh);
71
72  FOS=miu*SumV/SumH;
73  SFF=(miu*SumV+wb*1400)/SumH;
74  FOO=Mplus/Mminus;
75  Ffi=1.2;Fc=2.4;
```

```
76  F=(miu*SumV/Ffi+1400*wb/Fc)/SumH;
77  FOS=round(FOS*100)/100;
78  F=round(F*100)/100;
79  SFF=round(SFF*100)/100;
80  FOO=round(FOO*100)/100;
81  mprintf("\n\nFactor of safety against sliding as per
        IS:6512−1972=%f. <1.5",FOS);
82  mprintf("\nFactor of safety against sliding as per
        IS:6512−1984=%f. >1",F);
83  mprintf("\nShear friction factor=%f. <6",SFF);
84  mprintf("\nFactor of safety against overturning=%f.
        <1.5",FOO);
85  mprintf("\n\nDam is unsafe for given loading
        conditions");
```

Scilab code Exa 8.16 EX8 16

```
1
2
3  //example 8.16
4  //Check the stability and determine principal and
        shear stress at toe and heel
5  clc; funcprot(0);
6  //Given
7  c=1;
8  miu=0.7;              //coefficient of friction
9  H=70;                  //heigth of dam
10 ht=0;                  //heigth of tail water
11 Lf=6.5;               //location of foundation gallery
        from heel
12 wb=52.5;              //width of base
13 Bt=7;                  //width of top of dam
14 hw=70;                 //heigth of water in reservior
15 Hs1=35;                //heigth of slope on upstream
        side
```

```scilab
16  Hs2=60;                     //heigth  of  slope  on  downstream
        side
17  gamma_m=24;          //unit  weigth  of  concrete
18  gamma_w=9.81;            //unit  weigth  of  water
19  theta=atan(0.1);
20  fi=atan(0.7);
21  //self  weigth  of  dam
22  W1=(Hs1*3.5*gamma_m)/2,
23  W2=(Bt*H*gamma_m),
24  W3=(Hs2^2*0.7*gamma_m)/2,
25  //weigth  of  superimposed  water
26  W4=(Hs1*3.5*gamma_w)/2,
27  W5=(hw-Hs1)*3.5*gamma_w,
28  wp=hw^2*gamma_w/2;                       //water
        pressure
29  Pt=gamma_w*ht,
30  Ph=gamma_w*hw,
31  Pg=(ht+(hw-ht)/3)*gamma_w,
32  U=(Pt*(wb-Lf))+(Pg*Lf)+((Ph-Pg)*Lf/2)+((Pg-Pt)*(wb-
        Lf)/2)*c,
33  l1=(wb-Lf)/2,l2=(2*(wb-Lf))/3,l3=(wb-Lf)+(Lf/2),l4=(
        wb-Lf)+((2*Lf)/3),
34  L7=(((Pt*(wb-Lf))*l1)+((Pg-Pt)*(wb-Lf)*l2/2)+((Pg*Lf
        )*l3)+((Ph-Pg)*Lf*l4/2))/U,
35  L1=(wb-3.5)+3.5/3,
36  L2=(0.7*Hs2)+(Bt/2),
37  L3=(2*Hs2*0.7)/3,
38  L4=(wb-3.5)+(2*3.5)/3,
39  L5=(wb-3.5)+(3.5/2),
40  L6=hw/3;
41  M1=W1*L1,M2=W2*L2,M3=W3*L3,M4=W4*L4;
42  M5=W5*L5;
43  M6=wp*L6;
44  M7=U*L7;
45  SumV1=W1+W2+W3;
46  SumM1=M1+M2+M3;
47  SumV2=SumV1+W4+W5;
48  SumM2=SumM1+M4+M5-M6;
```

```scilab
49  SumV3=SumV2-U;
50  SumM3=SumM2-M7;
51  Mplus=1547377;
52  Mminus=870421;
53  SumH=wp;
54
55  //case 1. Reservior empty
56  x=SumM1/SumV1;
57  e=wb/2-x;
58  pnt=(SumV1/wb)*(1+(6*e/wb));
59  pnh=(SumV1/wb)*(1-(6*e/wb));
60  sigmat=pnt*sec(fi)^2;
61  sigmah=pnh*sec(theta)^2;
62  taut=pnt*tan(fi);
63  tauh=pnh*tan(theta);
64  pnt=round(pnt*10)/10;
65  pnh=round(pnh*10)/10;
66  sigmat=round(sigmat*10)/10;
67  sigmah=round(sigmah*10)/10;
68  taut=round(taut*10)/10;
69  tauh=round(tauh*10)/10;
70  mprintf("case 1. Reservior empty:");
71  mprintf("\nNormal stress at toe=%f kN/square.m.",pnt
        );
72  mprintf("\nNormal stress at heel=%f kN/square.m.",
        pnh);
73  mprintf("\nPrincipal stress at toe=%f kN/square.m.",
        sigmat);
74  mprintf("\nPrincipal stress at heel=%f kN/square.m."
        ,sigmah);
75  mprintf("\nShear stress at toe=%f kN/square.m.",taut
        );
76  mprintf("\nShear stress at heel=%f kN/square.m.",
        tauh);
77
78  //case2. reservior full without uplift
79  x=SumM2/SumV2;
80  e=wb/2-x;
```

```
81  p=hw*gamma_w;
82  pnt=(SumV2/wb)*(1+(6*e/wb));
83  pnh=(SumV2/wb)*(1-(6*e/wb));
84  sigmat=pnt*sec(fi)^2;
85  sigmah=pnh*sec(theta)^2-p*tan(theta)^2;
86  taut=pnt*tan(fi);
87  tauh=-(pnh-p)*tan(theta);
88  pnt=round(pnt*10)/10;
89  pnh=round(pnh*10)/10;
90  sigmat=round(sigmat*10)/10;
91  sigmah=round(sigmah*10)/10;
92  taut=round(taut*10)/10;
93  tauh=round(tauh*10)/10;
94  mprintf("\n\ncase 2. reservior full without uplift:"
       );
95  mprintf("\nNormal stress at toe=%f kN/square.m.",pnt
       );
96  mprintf("\nNormal stress at heel=%f kN/square.m.",
       pnh);
97  mprintf("\nPrincipal stress at toe=%f kN/square.m.",
       sigmat);
98  mprintf("\nPrincipal stress at heel=%f kN/square.m."
       ,sigmah);
99  mprintf("\nShear stress at toe=%f kN/square.m.",taut
       );
100 mprintf("\nShear stress at heel=%f kN/square.m.",
       tauh);
101
102 //case3. reservior full with uplift
103 x=SumM3/SumV3;
104 e=wb/2-x;
105 p=hw*gamma_w;
106 pnt=(SumV3/wb)*(1+(6*e/wb));
107 pnh=(SumV3/wb)*(1-(6*e/wb));
108 sigmat=pnt*sec(fi)^2;
109 sigmah=pnh*sec(theta)^2-p*tan(theta)^2;
110 taut=pnt*tan(fi);
111 tauh=-(pnh-p)*tan(theta);
```

```scilab
112  pnt=round(pnt);
113  pnh=round(pnh);
114  sigmat=round(sigmat);
115  sigmah=round(sigmah);
116  taut=round(taut);
117  tauh=round(tauh);
118  mprintf("\n\ncase 3. reservior full with uplift:");
119  mprintf("\nNormal stress at toe=%f kN/square.m.",pnt
        );
120  mprintf("\nNormal stress at heel=%f kN/square.m.",
        pnh);
121  mprintf("\nPrincipal stress at toe=%f kN/square.m.",
        sigmat);
122  mprintf("\nPrincipal stress at heel=%f kN/square.m."
        ,sigmah);
123  mprintf("\nShear stress at toe=%f kN/square.m.",taut
        );
124  mprintf("\nShear stress at heel=%f kN/square.m.",
        tauh);
125
126  FOS=miu*SumV3/SumH;
127  SFF=(miu*SumV3+wb*1400)/SumH;
128  FOO=Mplus/Mminus;
129  Ffi=1.5;Fc=3.6;
130  F=(miu*SumV3/Ffi+1400*wb/Fc)/SumH;
131  FOS=round(FOS*1000)/1000;
132  SFF=round(SFF*100)/100;
133  FOO=round(FOO*100)/100;
134  F=round(F*1000)/1000;
135  mprintf("\n\nFactor of safety against sliding=%f.",
        FOS);
136  mprintf("\nShear friction factor=%f.",SFF);
137  mprintf("\nFactor of safety against overturning=%f."
        ,FOO);
138  mprintf("\nFactor of safety for load combination B=
        %f. > 1",F);
139  mprintf("\nDam is safe ");
140
```

```matlab
141  //Case4.considering seismic forces
142  Cm=0.712;
143  alphah=0.1;
144  alphav=0.08;
145  hp=0.726*Cm*alphah*gamma_w*hw^2;        //hydrodynamic
         pressure
146  Mhp=0.299*Cm*alphah*gamma_w*hw^3;        //moment due to
         hydrodynamic pressure
147  //inertial load due to horizontal acceleration
148  I1=W2/10;
149  I2=W3/10;
150  I3=W1/10;
151  v=SumV1*alphav;
152  Mv=116444;
153  SumV4=SumV3-v;
154  SumH1=SumH+I1+I2+I3+hp;
155  M8=I1*35;
156  M9=I2*20;
157  M10=I3*35/3;
158  Mminus1=1161849;
159  SumM4=SumM3-M8-M9-M10-Mhp-Mv;
160
161  x=SumM4/SumV4;
162  e=wb/2-x;
163  p=hw*gamma_w;
164  pe=Cm*alphah*gamma_w*hw;
165  pnt=(SumV4/wb)*(1+(6*e/wb));
166  pnh=(SumV4/wb)*(1-(6*e/wb));
167  sigmat=pnt*sec(fi)^2;
168  sigmah=pnh*sec(theta)^2-(p+pe)*tan(theta)^2;
169  taut=pnt*tan(fi);
170  tauh=(-pnh+(p+pe))*tan(theta);
171  pnt=round(pnt);
172  pnh=round(pnh);
173  sigmat=round(sigmat);
174  sigmah=round(sigmah);
175  taut=round(taut);
176  tauh=round(tauh);
```

```
177  mprintf("\n\ncase 4.considering seismic forces");
178  mprintf("\nNormal stress at toe=%f kN/square.m.",pnt
        );
179  mprintf("\nNormal stress at heel=%f kN/square.m.",
        pnh);
180  mprintf("\nPrincipal stress at toe=%f kN/square.m.",
        sigmat);
181  mprintf("\nPrincipal stress at heel=%f kN/square.m."
        ,sigmah);
182  mprintf("\nShear stress at toe=%f kN/square.m.",taut
        );
183  mprintf("\nShear stress at heel=%f kN/square.m.",
        tauh);    //answer is wrong in book
184
185  FOS=miu*SumV4/SumH1;
186  SFF=(miu*SumV4+wb*1400)/SumH1;
187  FOO=Mplus/Mminus1;
188  Ffi=1.2;Fc=2.7;
189  F=(miu*SumV4/Ffi+1400*wb/Fc)/SumH1;
190  FOS=round(FOS*1000)/1000;
191  SFF=round(SFF*100)/100;
192  FOO=round(FOO*100)/100;
193  F=round(F*100)/100;
194  mprintf("\n\nFactor of safety against sliding=%f.",
        FOS);
195  mprintf("\nShear friction factor=%f.",SFF);
196  mprintf("\nFactor of safety against overturning=%f."
        ,FOO);
197  mprintf("\nFactor of safety for load combination E=
        %f. > 1",F);
198  mprintf("\nDam is safe ");
```

Scilab code Exa 8.17 EX8 17

```
1
```

```
 2
 3  //example 8.17
 4  //design practical profile of gravity dam
 5  clc;funcprot(0);
 6  //given
 7  c=1;
 8  rlb=1450;              //R.L of base of dam
 9  rlw=1480.5;            //R.L of water level
10  Sg=2.4;                //specific gravity of masonary
11  gamma_w=9.81;          //unit weigth of water
12  w=1;                   //heigth of waves
13  f=1200;                //safe compressive stress for
        masonary
14  FB=1.5*w;
15  rlt=FB+rlw;            //R.L of top of dam
16  H=rlt-rlb;             //heigth of dam
17  LH=f/(gamma_w*(Sg+1))
18  LH=round(LH*100)/100;
19  mprintf("Heigth of dam=%f m.",H);
20  mprintf("\nlimiting heigth of dam=%f m.",LH);
21  mprintf("\nDam is low gravity dam");
22  hw=rlw-rlb;
23  //keep top width,a=4.5.
24  a=4.5;
25  P=hw/(Sg^0.5);
26  P=round(P*10)/10;
27  mprintf("\nBase width of elementary profile=%f m.",P
        );
28  uo=a/16;
29  wb=uo+P;
30  wb=round(wb);
31  mprintf("\nBase width=%f m.",wb);
32  D=2*a*(Sg^0.5);
33  D=round(D);
34  mprintf("\nDistance upto which u/s slope is vertical
         from water level=%f m.",D);
```

Scilab code Exa 8.18 EX8 18

```
1
2
3  //example 8.18
4  //determine if dam is safe against sliding
5  clc;funcprot(0);
6  //given
7  hw=97;              //heigth of water in reservior
8  Bt=7;               //width of top of dam
9  H=100;              //heigth of the dam
10 Hs2=90;             //heigth of slope on downstream
       side
11 wb=75;              //width of base of dam
12 miu=0.75;           //coefficient of friction
13 gamma_d=2.4;        //weigth density of concrete
14 gamma_w=1000;       //weigth density of water
15
16 P=gamma_w*hw^2/(2*1000);
17 W1=Bt*gamma_d*H;
18 W2=(wb-Bt)*Hs2*gamma_d/2;
19 W=W1+W2;
20 FOS=miu*W/P;
21 FOS=round(FOS*1000)/1000;
22 mprintf("Factor of safety against sliding=%f.",FOS);
23 mprintf("\nDam is safe against sliding");
```

Scilab code Exa 8.19 EX8 19

```
1
2
3  //example 8.19
```

```scilab
 4  //calculate
 5  //Factor of safety against overturning
 6  //Factor of safety against sliding
 7  //Shear friction factor
 8  clc;funcprot(0);
 9  //given
10  c=1;
11  H=10;                //heigth of dam
12  hw=10;               //heigth of water in reservior
13  wb=8.25;             //bottom width
14  Bt=1;                //top width
15  Hs1=0.1;             //slope on upstream side
16  gamma_w=9.81;        //unit weigth of water
17  gamma_m=22.4;        //unit weigth of masonary
18  f=1400;              //permissible shear stress at
        joint
19  miu=0.75;            //coefficient of friction
20  fi=atan(0.625);
21  theta=atan(0.1);
22
23  W1=Bt*H*gamma_m;
24  W2=H*H*Hs1*gamma_m/2;
25  W3=H*6.25*gamma_m/2;
26  W4=hw*gamma_w*H*Hs1/2;
27  P=gamma_w*hw^2/2;
28  U=wb*gamma_w*hw*c/2;
29  SumV=W1+W2+W3+W4-U;
30  L3=2*(wb-(Hs1*H)-Bt)/3;
31  L1=(wb-(Hs1*H)-Bt)+Bt/2;
32  L2=(wb-(Hs1*H)-Bt)+Bt+(Hs1*H/3);
33  L4=(wb-(Hs1*H)-Bt)+Bt+(2*Hs1*H/3);
34  L5=2*wb/3;L6=hw/3;
35  M1=W1*L1;M2=W2*L2;M3=W3*L3;M4=W4*L4;
36  M5=U*L5;M6=P*L6;
37  SumM=M1+M2+M3+M4-M5-M6;
38  Mplus=M1+M2+M3+M4;
39  Mminus=M5+M6;
40  FOS=miu*SumV/P;
```

```
41  SFF =( miu * SumV + wb *1400)/ P ;
42  FOO = Mplus / Mminus ;
43  FOS = round ( FOS *100)/100;
44  SFF = round ( SFF *10)/10;
45  FOO = round ( FOO *100)/100;
46  mprintf (" Factor  of  safety  against  sliding=%f.", FOS );
47  mprintf ("\ nShear  friction  factor=%f.", SFF );
48  mprintf ("\ nFactor  of  safety  against  overturning=%f."
        , FOO );
49  mprintf ("\ nDam  is  unsafe  against  overturning");
```

Scilab code Exa 8.20 EX8 20

```
 1
 2
 3  // example  8.20
 4  // calculate  streeses  at  heel  and  toe  of  dam
 5  clc ; funcprot (0);
 6  // given
 7  c =1;
 8  hw =80;              // heigth  of  water  in  reservior
 9  Bt =6;               // width  of  top  of  dam
10  H =84;               // heigth  of  the  dam
11  Hs2 =75;             // heigth  of  slope  on  downstream
        side
12  wb =56;              // width  of  base  of  dam
13  Lf =8;               // distance  of  foundation  gallery
        from  heel
14  gamma_d =23.5;       // weigth  density  of  concrete
15  gamma_w =9.81;       // weigth  density  of  water
16  ht =6;               // heigth  of  tail  water
17
18  W1 = Bt * gamma_d * H ;
19  W2 = gamma_d * Hs2 *( wb - Bt )/2;
20  W3 = gamma_w * ht *4/2;
```

```
21  W4=gamma_w*hw^2/2;
22  W5=gamma_w*ht^2/2;
23  Pt=gamma_w*ht,
24  Ph=gamma_w*hw,
25  Pg=(ht+(hw-ht)/3)*gamma_w,
26  U=(Pt*(wb-Lf))+(Pg*Lf)+((Ph-Pg)*Lf/2)+((Pg-Pt)*(wb-
      Lf)/2)*c,
27  l1=(wb-Lf)/2,l2=(2*(wb-Lf))/3,l3=(wb-Lf)+(Lf/2),l4=(
      wb-Lf)+((2*Lf)/3),
28  L6=(((Pt*(wb-Lf))*l1)+((Pg-Pt)*(wb-Lf)*l2/2)+((Pg*Lf
      )*l3)+((Ph-Pg)*Lf*l4/2))/U,
29  L1=(wb-Bt)+(Bt/2),
30  L2=(2*(wb-Bt))/3,
31  L3=4/3;
32  L4=hw/3;
33  L5=ht/3;
34  M1=W1*L1,M2=W2*L2,M3=W3*L3,M4=W4*L4,M5=W5*L5,M6=U*L6
      ;
35  SumV=W1+W2+W3-U;
36  SumH=W4-W5;
37  SumM=M1+M2+M3-M4+M5-M6;
38  x=SumM/SumV;
39  e=wb/2-x;
40  pnt=(SumV/wb)*(1+(6*e/wb));
41  pnh=(SumV/wb)*(1-(6*e/wb));
42  pnt=round(pnt*10)/10;
43  pnh=round(pnh*10)/10;
44  mprintf("Maximum Normal stress at toe=%f kN/square.m
      .",pnt);
45  mprintf("\nMaximum Normal stress at heel=%f kN/
      square.m.",pnh);
```

Chapter 10

EARTH AND ROCKFILL DAM

Scilab code Exa 10.1 EX10 1

```
1
2
3  // example 10.1
4  // calculate  seepage  flow  per  unit  length  of  dam
5  clc ; funcprot (0) ;
6  // given
7  K =5D -4;                 // coefficient  of  permeability  of
           soil
8  Bt =6;                    // width  of  top  of  dam
9  wb =146;                  // width  of  base  of  dam
10 H =20;                    // heigth  of  dam
11 hw =2;                    // heigth  of  water  in  reservior
12 hs1 =4;                   // slope  on  upstream  side
13 hs2 =3;                   // slope  on  downstream  side
14 df =30;                   // length  of  drainage  filter
15
16 x = wb - df -72+72*0.3;
17 y =18;
18 s =( x ^2+ y ^2) ^0.5 - x ;
```

```
19
20  x =[0  10  20  30  40  50  60  65.6];
21  for  i =1:8
22       y(i)=(4.849*x(i)+5.879)^0.5;
23       y(i)=round(y(i)*1000)/1000;
24  end
25
26  mprintf("\nx                            y");
27  for  i =1:8
28       mprintf("\n%f            %f",x(i),y(i));
29  end
30
31  sf=K*s*10000;
32  sf=round(sf*1000)/1000;
33  mprintf("\nSeepage  flow  per  unit  length  of  dam=%fD-6
            cumecs/metre  length  of  dam.",sf);
```

Scilab code Exa 10.2 EX10 2

```
1
2
3  //example  10.2
4  //calculate  discharge  per  meter  length  of  dam
5  clc;funcprot(0);
6  //given
7  K=3D-3;                //coefficient  of  permeability
8  nd=25;                  //number  of  potential  drops
9  nf=4;                   //number  of  flow  channels
10 lf=40;                 //filter  length
11 H=52;                  //heigth  of  dam
12 fb=2;                  //free  board
13
14 q=K*(H-fb)*nf/(nd*100);
15 mprintf("Discharge  per  meter  length  of  dam=%f cumec/
          metre  length.",q);
```

Scilab code Exa 10.3 EX10 3

```scilab
1
2
3  //example10.3
4  //calculate factor of safety for slope
5  clc;funcprot(0);
6  //given
7  x=4;                //given scale
8  An=14.4;            //area of N rectangle
9  At=6.4;             //area of T rectangle
10 Au=4.9;             //area of U rectangle
11 L=12.6;             //length of arc;
12 gamma_m=19;         //unit weigth of soil
13 gamma_w=9.81;       //unit weigth of water
14 fi=26;              //effective angle(degree)
15 co=19.5;            //cohesion value
16
17 //consider 1m length of dam
18 SumN=An*x^2*gamma_m;
19 SumT=At*x^2*gamma_m;
20 SumU=Au*x^2*gamma_w;
21 Le=x*L;
22 F=((Le*co)+(SumN-SumU)*tand(fi))/SumT;
23 F=round(F*100)/100;
24 mprintf("Factor of safety for slope=%f.",F);
```

Scilab code Exa 10.4 EX10 4

```scilab
1
2
```

```
3  //example 10.4
4  //check section for:
5  //Stability of d/s slope against steady seepage
6  //Sloughing of u/s slope against sudden drawdown
7  //Stability of the foundation against shear
8  //Seepage through body of dam
9  clc;funcprot(0);
10 //given
11 //Dimensions
12 H=20;              //Heigth of dam
13 Bt=6;              //top width of dam
14 s1=4;              //u/s slope
15 s2=3;              //d/s slope
16 fb=2;              //free board
17 //Properties of materials of dam
18 gamma_d=17.27;         //dry density
19 wc=0.15;               //optimum water content
20 gamma_s=21.19;         //saturated density
21 gamma_w=9.81;          //unit weigth of water
22 wavg=19.62;            //average unit weigth under
       seepage
23 theta=26;              //average angle of internal
       friction(degree)
24 co=19.13;              //average cohesion
25 K=5D-4;                //coefficient of permeability
26 //properties of foundation materials
27 gamma_f=17.27;        //average unit weigth
28 cof=47.87;            //average cohesion
29 fi=8;                 //average angle internal
       friction
30 t=6;                  //thickness of clay
31 FOSp=1.5;             //permissible factor of safety
       of slope
32 PS=8D-6;              //permissible seepage
33
34
35 //(a) Stability of d/s slope against steady seepage
36 An=302.4;             //area of N diagram
```

```
37  At=91.2;                    //area  of T diagram
38  Au=98.4;                    //area of U diagram
39  Le=60;                      //length of arc
40  SumN=An*gamma_s;
41  SumT=At*gamma_s;
42  SumU=Au*gamma_w;
43  F=((Le*co)+(SumN-SumU)*tand(theta))/SumT;
44  F=round(F*100)/100;
45  mprintf("Part(a):")
46  mprintf("\nFactor of safety for slope=%f.",F);
47  mprintf("\nSafe");
48
49  //(b) Sloughing of u/s slope against sudden drawdown
50  h1=15;
51  b=80;
52  P=gamma_s*H^2*tand(45-(theta/2))^2/2+gamma_w*h1^2/2;
53  sav=P/b;
54  smax=2*sav;
55  Ne=(gamma_s-gamma_w)*b*H/2;
56  R=Ne*tand(theta)+co*b;
57  fs=R/P;
58  fs=round(fs*100)/100;
59  mprintf("\n\nPart(b):")
60  mprintf("\nFactor of safety w.r.t average shear=%f."
        ,fs);
61  mprintf("\nSafe");
62  sr=0.6*H*(gamma_s-gamma_w)*tand(theta)+co;
63  FS=sr/smax;
64  FS=round(FS*100)/100;
65  mprintf("\n\nFactor of safety w.r.t maximum shear=%f
        .",FS);
66  mprintf("\nSafe");
67
68  //(c) Stability of the foundation against shear
69  h1=26;
70  h2=6;
71  gamma_m=(wavg*(h1-h2)+gamma_f*h2)/h1;
72  l=(gamma_m*h1*tand(fi)+cof)/(gamma_m*h1);
```

```
73  fi1=atand(l);
74  P=(h1^2-h2^2)/2*gamma_m*tand(45-(fi1/2))^2;
75  sav=P/b;
76  smax=2*sav;
77  s1=cof+gamma_f*h2*tand(fi);
78  s2=cof+gamma_m*h1*tand(fi);
79  as=(s1+s2)/2;
80  fs=as/sav;
81  fs=round(fs*100)/100;
82  mprintf("\n\nPart(c):")
83  mprintf("\nFactor of safety w.r.t overall shear=%f."
        ,fs);
84  mprintf("\nSafe");
85
86  gamma_av=(wavg*0.6*H+gamma_f*h2)/((0.6*H)+h2);
87  s=cof+gamma_av*0.6*H*tand(fi);
88  fs=s/smax;
89  fs=round(fs*100)/100;
90  mprintf("\n\nFactor of safety w.r.t overall shear=%f
        .",fs);
91  mprintf("\nUnsafe");
92
93  //(d) Seepage through body of dam
94  s=2;          //measured
95  q=K*s*100000/100;
96  mprintf("\n\nPart(d):")
97  mprintf("\n Seepage through body of dam=%fD-5 cumecs
        /m length of dam",q);
```

Scilab code Exa 10.5 EX10 5

```
1
2
```
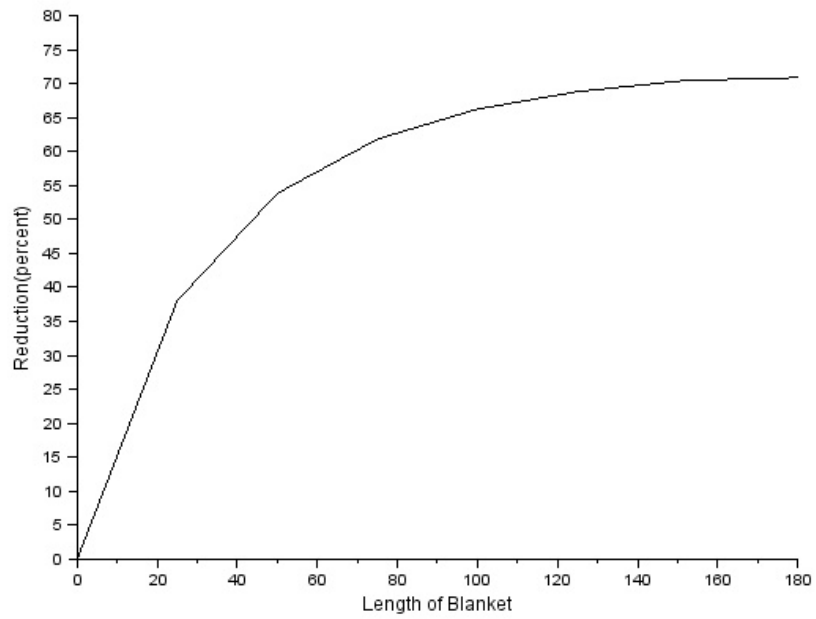
Figure 10.1: EX10 5

```scilab
3  //example 10.5
4  //design upstream impervious blanket
5  clc;funcprot(0);
6  //given
7  Zb=1.2;          //thickness of blanket
8  Zf=8;            //distance of blanket from foundation
9  kb=0.06;         //coefficient of permeability of
       blanket material
10 kf=72;           //coefficient of permeability of
       foundation soil
11 Hw=10;              //heigth of water in reservior
12 Xd=40;
13
14 a=(kb/(kf*Zb*Zf))^0.5;
15 Xo=1.414/a;
16
17 //we vary value of x
18 x=[0 25 50 75 100 125 151.8 300]
19 for i=1:8
20     e=exp(2*a*x(i));
21     Xr(i)=(e-1)/(a*(e+1));
22     ho(i)=Xr(i)*Hw/(Xr(i)+Xd);
23     r(i)=Xr(i)*100/(Xr(i)+Xd);
24 end
25 mprintf("\nx                       Xr                  ho
              reduction q(percent)");
26 for i=1:8
27     mprintf("\n%f          %f          %f          %f",x(i
          ),Xr(i),ho(i),r(i));
28 end
29 //graph is plotted between r and x.
30 //after around 130m length there is only slight
      increase in head dissipated(ho)
31 L=130;
32 mprintf("\nThickness of blanket=%f m",Zb);
33 mprintf("\nLength of blanket=%i m.",L);
```

Chapter 11

SPILLWAYS

Scilab code Exa 11.1 EX11 1

```
1
2
3  //example 11.1
4  //calculate compute the dynamic force on curved
       section
5  clc; funcprot (0);
6  //given
7  h=1.2;          //head of water
8  Cd=2.2;         //coefficient of discharge
9  rho=1;          //density of water
10 gamma_w=9.81;   //unit weigth of water
11
12 q=Cd*h^1.5;
13
14 //applying bernaulli's equation at u/s water surface
       at section A and B
15 //solving it by error and trial method we get
16 v1=13.7; v2=14.7;
17 d1=0.212; d2=0.197;
18
19 F1=gamma_w*d1^2*cosd(60)/2;
```

```
20  F2=gamma_w*d2^2/2;
21  W=gamma_w*60*2*%pi*3*((d1+d2)/2)/360;
22  Fx=rho*q*(v2-v1*cosd(60))-F1/2+F2;
23  Fy=rho*q*(v1*sind(60))+F1*sind(60)+W;
24  F=(Fx^2+Fy^2)^0.5;
25  F=round(F*100)/100;
26  mprintf("Resultant force=%f kN/m.",F);
```

Scilab code Exa 11.2 EX11 2

```
1
2
3  //example 11.2
4  //calculate discharge over oggy weir
5  clc;funcprot(0);
6  //given
7  C=2.4;              //coefficient of discharge
8  H=2;                //head
9  L=100;              //length of spillway
10 wc=8;               //heigth of weir crest above bottom
11 g=9.81;             //acceleration due to gravity
12 h=H+wc;
13 Q1=C*L*H^(1.5); //neglecting approach velocity and
       end contractions
14 va=Q1/(h*L);
15 ha=va^2/(2*g);
16 Ha=ha+H;
17 Q=C*L*Ha^1.5;
18 Q=round(Q*10)/10;
19 mprintf("discharge over oggy weir=%f cumecs.",Q);
```

Scilab code Exa 11.3 EX11 3

```
1
2
3  //example 11.3
4  //calculate
5  //capacity of siphon
6  //head required in oggy spillway
7  //length of oggy weir required
8  clc;funcprot(0);
9  //given
10 t=6;             //tail water elevation
11 h=1;             //heigth of siphon spillway
12 w=4;             //width of siphon spillway
13 hw=1.5;          //head water elevation
14 C=0.6;           //coefficient of discharge
15 Co=2.25;         //coefficient of discharge of oggy
       spillway
16 lo=4;            //length of oggy spillway
17 hc=1.5;          //head on weir crest
18 g=9.81;          //acceleration due to gravity
19
20 //part (a)
21 Q=C*h*w*(2*g*(t+hw))^0.5;
22 Q=round(Q*10)/10;
23 mprintf("capacity of siphon=%f cumecs.",Q);
24
25 //part (b)
26 h1=(Q/(Co*lo))^(2/3);
27 h1=round(h1*100)/100;
28 mprintf("\nhead required in oggy spillway=%f m",h1);
29
30 //part (c)
31 L=Q/(Co*(hc)^1.5);
32 L=round(L*100)/100;
33 mprintf("\nlength of oggy weir required=%f m.",L);
```

Scilab code Exa 11.4 EX11 4

```
1
2
3  //example 11.4
4  //calculate number of siphons units required
5  clc;funcprot(0);
6  //given
7  rl=435;            //full reservior level
8  cl=429.6;          //level of centre of siphon
9  hfl=435.85;        //high flood level
10 hfd=600;           //high flood discharge
11 w=4;               //width of throat
12 h=2;               //heigth of throat
13 C=0.65;            //coefficient of discharge
14 g=9.81;            //acceleration due to gravity
15
16 H=hfl-cl;
17 Q=C*w*h*(2*g*H)^0.5;
18 n=hfd/Q;
19 n=round(n*100)/100;
20 mprintf(" number of siphons units required=%f.\
       nhence provide 11 siphons units.",n);
```

Scilab code Exa 11.5 EX11 5

```
1
2
3  //example 11.5
4  //design oggy spillway for concrete gravity dam
5  clc;funcprot(0);
6  //given
7  rbl=250;           //avarage river bed level
8  rlc=350;           //R.L of spillway crest
9  s=0.75;            //slope on downstream side
```

215

```
10   Q=6500;            //discharge
11   L=5*9;             //length of spillway
12   Cd=2.2;            //coefficient of discharge
13   t=2;               //thickness of each pier
14
15   //step 1. computation of design head
16   H=(Q/(Cd*L))^(2/3);
17   P=rlc-rbl;
18
19   //P/H=6.15,which is <1.33;it is a high overflow
         spillway
20
21   //H+P/H=7.15>1.7; hence discharge coefficient is not
          affected by downstream apron interface
22
23   Kp=0.01;Ka=0.1;N=4;
24   He=17.5;                    //assumed
25   Le=L-2*(N*Kp+Ka)*He;
26   He1=(Q/(Cd*Le))^(2/3);
27   He1=round(He1*100)/100;
28   //He1 is almost equal to He
29   mprintf("crest profile will be designed for Hd=%f m.
         ",He1);
30
31   //step 2. determination of d/s profile
32
33   //equating the slope of d/s side and derivative of
         profile equation suggested by WES
34   x=27.03;
35   y=0.04372*x^1.85;
36   mprintf("\n\ndownstream profile:");
37   x=[1:1:26]
38   for i=1:26
39       y(i)=0.04372*x(i)^1.85;
40       y(i)=round(y(i)*1000)/1000;
41   end
42   mprintf("\nx            y");
43   for i=1:26
```

216

```
44         mprintf(" \n%i            %f",x(i),y(i));
45 end
46 mprintf(" \n27.03            19.48");
47
48
49 //step 3. determination of u/s profile
50 // cosidering equation for vertical u/s face and Hd
       =17.58
51
52 mprintf(" \n\nupstream profile :");
53 x=[-0.5 -0.1 -1.5 -2.0 -3.0 -4.0 -4.75];
54 for i=1:7
55     y(i)=0.0633*(x(i)+4.7466)^1.85+2.2151-1.2643*(x(
           i)+4.7466)^0.625;
56     y(i)=round(y(i)*1000)/1000;
57 end
58 mprintf(" \nx                  y");
59 for i=1:7
60     mprintf(" \n%f            %f",x(i),y(i));
61 end
62
63 //step 4.design of d/s bucket
64
65 R=P/4;
66 mprintf(" \n\nradius of bucket=%i m.",R);
67 mprintf(" \nbucket will subtend angle of 60 degree at
        the centre .");
```

Scilab code Exa 11.6 EX11 6

```
1
2
3 //example 11.6
4 //design length and depth of stilling basin
5 clc;funcprot(0);
```

```
 6  //given
 7  q=1;                 //discharge of spillway
 8  Cd=0.7;              //coefficient of discharge
 9  h1=10;                //heigth of crest above downstream
        silting basin
10  g=9.81;                //acceleration due to gravity
11  Cv=0.9;               //coefficient of velocity
12
13  h=(3*q/(2*Cd*(2*g)^0.5))^(2/3);
14  H=h1+h/2;
15  vt=(2*g*H)^0.5;
16  v1=Cv*vt;
17  y1=q/v1;
18  F1=v1/(g*y1)^0.5;
19  //F>1, flow is super-critical
20  y2=1;
21  v2=q/y2;
22  F2=v2/(g*y2)^0.5;                //<1
23  y2=(y1/2)*((1+8*F1^2)^0.5-1);
24  de=y2-1;
25  le=5*(y2-y1);
26  de=round(de*1000)/1000;
27  le=round(le*10)/10;
28  mprintf("stilling basin should be depressed by %f m.
        ",de);
29  mprintf("\nlength of stilling basin=%f m.",le);
```

Scilab code Exa 11.7 EX11 7

```
1
2
3  //example 11.7
4  //calculate leading dimension of hydraulic jump
        stilling basin
5  clc;funcprot(0);
```

```
6  //given
7  q=7.83;            //discharge through spillway
8  w=12.5;            //width of fall
9  d=2;               //depth of water in downstream
10 g=9.8;

11
12 y1=0.5;
13 v1=q/y1;
14 F1=v1/(g*y1)^0.5;

15
16 //F>1,flow is super-critical
17 v2=q/d;
18 F2=v2/(g*d)^0.5;
19 y2=(y1/2)*((1+8*F1^2)^0.5-1);
20 de=y2-d;
21 le=5*(y2-y1);
22 de=round(de*100)/100;
23 le=round(le*10)/10;
24 mprintf("stilling basin should be depressed by %f m.
       ",de);
25 mprintf("\nlength of stilling basin=%f m.",le);
```

Scilab code Exa 11.8 EX11 8

```
1
2
3  //example 11.8
4  //calculate force to be exerted to lift the gate
5  clc;funcprot(0);
6  //given
7  Ag=5*2.5;          //area of gate
8  miu=0.25;          //coefficient of friction
9  w=0.5;             //weigth of gate
10 h=2;               //head of water over crest
11 g=9.81;            //acceleration due to gravity
```

```
12  gamma_w =1000;        //unit  weigth  of  water
13
14  m=w*g*1000;
15  F=gamma_w*Ag*h*h*g/10;
16  ff=miu*F;
17  tf=(m+ff)/1000;
18  mprintf("force  to  be  exerted  to  lift  the  gate=%f  kN.
        ",tf);
```

Scilab code Exa 11.9 EX11 9

```
1
2
3   //example  11.9
4   //calculate  depth  of  flow  at  both  end  of  jumps
5   clc;funcprot(0);
6   //given
7   q=19;            //dischrge  through  spillway
8   E=1;             //energy  loss
9
10  //from  energy  loss  equation;E=(y2-y1)^3/4y2y1;  and
        solving  it  we  get
11  //x=0.5*(-1+(1+294.39*(x-1)^9/64*x^3))
12  //by  trial  and  error  method  x=2.806
13  x=2.806;
14  y1=4*x/(x-1)^3;
15  y2=x*y1;
16  y1=round(y1*1000)/1000;
17  y2=round(y2*1000)/1000;
18  mprintf("depth  of  flow  at  both  end  of  jumps=%f  m  and
        %f  m.  respectively.",y1,y2);
```

Chapter 12

DIVERSION HEADWORKS

Scilab code Exa 12.1 EX12 1

```
1
2
3  //example 12.1
4  //calculate average hydraulic gradient
5  //uplift presuures and thickness of floor at 6m, 12m
       and 18m from u/s
6  clc;funcprot(0);
7  //given
8  rho=2.24;                //relative density of material
9  gamma_w=9.81;            //unit weigth of water
10 L=22;                    //total length
11 lc=(2*6)+L+(2*8);        //length of creep
12 hg=4/lc;                 //hydraulic gradient
13 mprintf("avearge hydraulic gradient=%f.",hg);
14 //at 6 m from u/s
15 x=6;
16 lg=(6*2)+x;
17 h1=4*(1-lg/50);          //unbalanced head
18 up=gamma_w*h1;
19 t=4*h1/(3*(rho-1));
20 up=round(up*100)/100;
```

```
21  t=round(t*100)/100;
22  mprintf("\n\nuplift at 6 m from u/s=%f kN/square
        metre.",up);
23  mprintf("\nthickness at 6 m from u/s=%f m.",t);
24
25  //at 12 m from u/s
26  x=12;
27  lg=(6*2)+x;
28  h1=4*(1-lg/50);          //unbalanced head
29  up=gamma_w*h1;
30  t=4*h1/(3*(rho-1));
31  up=round(up*100)/100;
32  t=round(t*100)/100;
33  mprintf("\n\nuplift at 12 m from u/s=%f kN/square
        metre.",up);
34  mprintf("\nthickness at 12 m from u/s=%f m.",t);
35
36  //at 18m from u/s
37  x=18;
38  lg=(6*2)+x;
39  h1=4*(1-lg/50);          //unbalanced head
40  up=gamma_w*h1;
41  t=4*h1/(3*(rho-1));
42  up=round(up*10)/10;
43  t=round(t*100)/100;
44  mprintf("\n\nuplift at 18 m from u/s=%f kN/square
        metre.",up);
45  mprintf("\nthickness at 18 m from u/s=%f m.",t);
```

Scilab code Exa 12.2 EX12 2

```
1
2
3  //example 12.2
4  //calculate uplift pressure and exit gradient
```

```scilab
5  //check whether section is safe against overturning
      and piping
6  clc;funcprot(0);
7  //given
8  b=54;                   //width of section
9  D1D2=16;                //distance between points D1 and D2
10 D2D3=37;                //distance between points D2 and D3
11
12 //first pipe line
13 //taking data from figure
14 d=105-97;
15 b1=0.5;
16 alpha=b/d;
17 //from the curves we get
18 fic1=0.665;
19 fid1=0.76;
20 fie1=1;
21 t=105-104;                              //floor thickness
22 corec=(fid1-fic1)*100*t/d;             //correction for
      floor thickness
23 //for pile no. 2
24 D=104-97;
25 d=104-97;
26 bdash=16;
27 C=19*(D/bdash)^0.5*(d+D)/b;    //correction for pile
      no. 2
28 fic1=fic1*100+corec+C;           //corrected pressures
29
30 //intermedite pipe line
31 d=105-97;
32 b1=16.5;
33 alpha=b/d;
34 r=b1/b;                    //ratio b1/b
35 //from the curves we get
36 fic2=0.52;
37 fie2=0.725;
38 fid2=0.615;
39 corec_c1=(fid2-fic2)*100*t/d;
```

```
40  corec_e1 =( fie2 - fid2 ) *100/ d ;
41
42  // for  pile  no .  1
43  C1 = C ;
44  d =104 -97;
45  bdash =37;
46  D =104 -95;
47  C2 =19*( D / bdash ) ^0.5*( d + D ) / b ;
48  // correction  due  to  slope
49  corec_e2 =3.3;                      // from  table  12.4
50  // correction  is  negative  due  to  upwrd  slope
51  l =4;                // horizontal  length  of  slope
52  corec_c2 = corec_e2 * l / bdash ;
53
54  fie2 = fie2 *100 - corec_e1 - corec_e2 ;
55  fic2 = fic2 *100+ corec_c1 + C2 - corec_c2 ;
56
57  // pile  no .  3  at  d / s  end
58  d =103.5 -95;
59  alpha_ = d / b ;
60  // for  curves
61  fie3 =0.35; fid3 =0.242;
62  corec_t =( fie3 - fid3 ) *100*(103.5 -102) / d ;
63
64  // correction  for  interference  at  pile  no .  2
65  d =102 -95;
66  D =102 -97;
67  C3 =19*( D / bdash ) ^0.5*( d + D ) / b ;
68  fie3 = fie3 *100 - corec_t - C3 ;
69
70  point =[ 'C1'  'C2'  'E2'  'E3' ];            // Point
71  P =[ fic1  fic2  fie2  fie3 ];                // pressure
        percent
72  P_ =[3.55  2.78  3.39  1.58];                // pressure
        head
73  mprintf (" Points          Pressure  percent
        Pressure  head ") ;
74  for  i =1:4
```

```
75      P(i)=round(P(i)*10)/10;
76      mprintf("\n%s                    %f                    %f
            ",point(i),P(i),P_(i));
77  end
78
79  //check for floor thickness
80  Pa=P_(2)-((P_(2)-P_(4))*6.5/37);
81  Pb=P_(2)-((P_(2)-P_(4))*24/37);
82  Pc=P_(2)-((P_(2)-P_(4))*30/37);
83  rho=2.24;                              //specific
        gravity of concrete
84  ta=Pa/(rho-1);
85  tb=Pb/(rho-1);
86  tc=Pc/(rho-1);
87  ta=round(ta*100)/100;
88  tb=round(tb*100)/100;
89  tc=round(tc*100)/100;
90  mprintf("\n\nThickness required at A=%f m.",ta);
91  mprintf("\nThickness required at B=%f m.",tb);
92  mprintf("\nThickness required at C=%f m.",tc);
93  t=103.5-102;
94  mprintf("\nThickness provided=%f m.",t);
95  mprintf("\nFloor thickness at B and C are adequate")
        ;
96
97  //exit gradient
98  H=108.5-103.5;                    //seepage head
99  d=103.5-95;                       //depth cut-off
100 //from exit gradient curve
101 alpha=6.35;
102 lambda=(1+(1+alpha^2)^0.5)/2;
103 Ge=H/(d*%pi*lambda^0.5);
104 mprintf("\n\nexit gradient=%f.",Ge);
105 mprintf("\n it is less than permissible exit
        gradient  < 1/6\nHence safe..");
```

Scilab code Exa 12.3 EX12 3

```
1
2
3  //example 12.3
4  //design a vertical drop weir on Bligh's theory
5  //test floor by Khosla's theory
6  clc;funcprot(0);
7  //given
8  Q=2800;                      //maximum flood discharge
9  hfl=285;                     //H.F.L before
       construction
10 hw=278;                      //minimum water level
11 fsl=284;                     //F.S.L of canal
12 c=12;                        //coefficient of creep
13 flux=1;                      //allowable afflux
14 Ge=1/6;                      //permissible exit
       gradient
15 rho=2.24;                    //specific gravity of
       concrete
16
17 //Hydraulic calculation
18 L=4.75*Q^0.5;
19 q=Q/L;
20 q=round(q*10)/10;
21 mprintf("Hydraulic calculation:");
22 mprintf("\ndischarge per unit width of river=%f
       cumecs.",q);
23 f=1;
24 R=1.35*(q^2/f)^(1/3);
25 R=round(R*100)/100;
26 mprintf("\nregime scour depth=%f m.",R);
27 V=q/R;                       //regime velocity
28 vh=V^2/(2*9.81);             //velocity head
```

```scilab
29  l_down=hfl+vh;
30  l_up=l_down+flux;
31  hfl_up=l_up-vh;
32  hfl_down=hfl-0.5;
33  hfl_down=round(hfl_down*100)/100;
34  mprintf("\nactual d/s H.F.L allowing 0.5 m for
        retrogation=%f m.",hfl_down);
35  K=(q/1.7)^(2/3);
36  cl=l_up-K;                    //crest level
37  cl=round(cl*100)/100;
38  mprintf("\ncrest level=%f m.",cl);
39  pl=fsl+0.5;                   //pond level
40  s=hfl_down-cl;                    //heigth of shutter
41  mprintf("\nheigth of shutter=%f m.",s);
42  rl_up_pile=hfl_up-1.5*R;      //R.L of bottom u/s pile
43  d_up_cut=hw-276;              //depth of upstream cut-
        off
44  mprintf("\ndepth of upstream cut-off=%f m.",d_up_cut
        );
45  mprintf("\n provide concrete cut off 2 m depth.");
46  rl_bot_ds=hfl_down-2*R;
47  Hs=hfl_down-hw;              //seepage head
48  Hc=cl-hw;                    //heigth of crest
49  mprintf("\nR.L of gates crest=%f m.",Hs);
50  mprintf("\nHeigth of crest=%f m.",Hc);
51
52  //design of weir wall
53  d=hfl_up-cl;
54  a=d/(rho)^0.5;
55  a=3*d/(2*rho);               //from sliding
        consideration
56  a=s+1;                       //from practical
        consieration
57  a=a+1;
58  mprintf("\n\ndesign of weir wall:")
59  mprintf("\nprovide top width of %i m.",a);
60  Mo=9.81*Hs^3/6;                    //overtirning moment
61  //equating the moment of resistance to overturning
```

```
   moment and putting the values we get
62 y=poly([-1.084,0.020,0.039],'x','c');
63 b=roots(y);
64 //we get b= − 5.5347261 and 5.0219056
65 //taking
66 b=5;
67 //when weir is submerged
68 C=0.58;
69 d=(q^2/((2*C/3)^2*2*9.81))^(1/3);
70 Mo=9.81*d*Hc^2/2;
71 //from equation of moment of resistence we get
72 y=poly([-77.55,3,1],'x','c');
73 b=roots(y);
74 //we get b= − 10.433085 and 7.4330846
75 //taking
76 b=8;
77 mprintf("\nbottom width=%i m.",b);
78
79 //design of impervious and pervious aprons
80 C=12;
81 L=C*Hs;
82 mprintf("\n\ndesign of impervious and pervious
      aprons:");
83 mprintf("\ntotal creep length=%i m.",L);
84 l1=2.21*C*(Hs/13)^0.5;
85 l1_=l1+1;
86 mprintf("\nlength of downstream impervious apron=%i
      m.",l1_);
87 d1=hw-276;
88 d2=hw-271;
89 l2=L-l1-(b+2*d1+2*d2);
90 mprintf("\nlength of upstream impervious apron=%i m.
      ",l2);
91 l3=18*C*(Hs*q/975)^0.5;
92 mprintf("\ntotal length of d/s apron=%i m.",l3);
                      //calculation is wrong in book
93 l=l3-l1;
94 le=l/2;
```

```
95  le=round(le*100)/100;
96  mprintf('\nprovide filter of length %f m. and
        launching apron of length %f m.',le,le);
97  t=d2*10^0.5/le;
98  mprintf("\nthickness of launching apron in
        horizontal position=%f m.",t);
99  mprintf("\nprovide launching apron of thickness 1.5
        m.");
100 T=2*d1;
101 V=d1*10^0.5;
102 ta=V/T;
103 ta=round(ta*10)/10;
104 mprintf("\nthickness of apron in horizontal position
        =%f m.",ta);
105 Hr=Hs-Hs*(4+33+8)/L;
106 t=4*Hr/(3*(rho-1));
107 t=round(t*10)/10;
108 mprintf('\nprovide thickness of %f m from d/s of
        weir wall to point 6 m from it.',t);
109 Hr=Hs-Hs*(4+33+8+6)/L;
110 t=4*Hr/(3*(rho-1));
111 t=round(t*10)/10;
112 mprintf("\nprovide thickness of %f m from 6 m to 12
        m from d/s end of weir wall.",t);
113 Hr=Hs-Hs*(4+33+8+12)/L;
114 t=4*Hr/(3*(rho-1));
115 t=round(t*10)/10;
116 mprintf("\nprovide thickness of %f m for rest of
        length of weir floor.",t);
117
118 //check by khosla's theory
119 b=33+8+19;                  //total horizontal length of
        impervious floor
120 d=7;                        //depth of downstream pile
121 alpha=b/d;
122 n=0.14;                     //n=1/%pi*(lambda)^0.5;
123 Ge=Hs*n/d;
124 mprintf("\n\ncheck by Khosla theory:");
```

```
125  mprintf("\nexit gradient=%f. < 1/6\n hence safe",Ge)
       ;
126  alpha_=d/b;
127  fic1=0.83;fid1=0.88;
128  corec_c1=(fid1-fic1)*100/2;
129  bdash=b;
130  d=2;D=7;
131  C1=19*(D/bdash)^0.5*(d+D)/b;
132  fic1=fic1*100+corec_c1+C1;
133  Pc=Hs*fic1/100;                         //pressure
       head at C
134  alpha_=d/b;
135  fie2=0.31;fid2=0.21;
136  corec_e1=(fie2-fid2)*1.7*100/7;
137  bdash=b;
138  d=7;D=2;
139  C1=19*(D/bdash)^0.5*(d+D)/b;
140  fie2=fie2*100-corec_e1-C1;              //in book
       3.53 value is wrong
141  Pe=Hs*fie2/100;                         //pressue
       head at E
142  //assuming linear variation of pressure for
       intermediate points
143  Pa=Pc-(Pc-Pe)*(33+8)/b;
144  t=Pa/1.24;
145  Pa=round(Pa*100)/100;
146  t=round(t*100)/100;
147  mprintf("\npressure at d/s of weir wall=%f m.",Pa);
148  mprintf("\nthickness at d/s of weir wall=%f m. <
       thickness by Bligh theory;\nhence safe.",t);
149  Pb=Pc-(Pc-Pe)*(33+8+6)/b;
150  t=Pb/1.24;
151  Pa=round(Pa*100)/100;
152  t=round(t*100)/100;
153  mprintf("\npressure at 6 m from d/s of weir wall=%f
       m.",Pb);
154  mprintf("\nthickness at 6m from d/s of weir wall=%f
       m. < thickness by Bligh theory;\nhence safe.",t);
```

```
155  Pc=Pc-(Pc-Pe)*(33+8+12)/b;
156  t=Pc/1.24;
157  Pa=round(Pa*100)/100;
158  t=round(t*100)/100;
159  mprintf("\npressure at 12 m from d/s of weir wall=%f
          m.",Pc);
160  mprintf("\nthickness at 12m from d/s of weir wall=%f
          m. > thickness by Bligh theory;\nhence unsafe.",
          t);
161  mprintf("\nhence increase th ethickness to 1.9 m for
          a length of 7 m of impervious floor.");
```

Scilab code Exa 12.4 EX12 4

```
 1
 2
 3  //example 12.4
 4  //design a slopeing glacis
 5  clc;funcprot(0);
 6  //given
 7  q=10;                          //maximum discharge
          intensity on weir crest
 8  hfl=255;                       //H.F.L before
          construction of weir
 9  rb=249.5;                      //R.L of river bed
10  pl=254;                        //pond level
11  s=1;                           //heigth of crest shutter
12  dhw=251.5;                     //anticipated downstream
          water level in river when water is dischrging
          with pond level upstream
13  br=0.5;                        //bed retrogression
14  f=0.9;                         //Laecey silt factor
15  Ge=1/7;                        //permissible exit
          gradient
16  flux=1;                        //permissible afflux
```

```
17
18  cl=pl-s;                          //crest level
19  mprintf("crest level=%f m.",cl);
20  K=(q/1.7)^(2/3);
21  tel_up=cl+K;
22  tel_up=round(tel_up*100)/100;
23  mprintf("\nelevation of u/s T.E.L=%f m.",tel_up);
24  R=1.35*(q^2/f)^(1/3);
25  R=round(R*10)/10;
26  mprintf("\nregime scour depth=%f m.",R);
27  V=q/R;                            //regime velocity
28  vh=V^2/(2*9.81);                  //velocity head
29  hfl_up=tel_up-vh;
30  tel_down=hfl+vh;
31  flux=hfl_up-hfl;
32  flux=round(flux*100)/100;
33  mprintf("\nafflux=%f. which is near to permissible",
        flux);
34  hfl_down=hfl-br;                  //downstream H.F.L
        after retrogression
35  tel_down=tel_down-br;            //downstream T.F.L
        after retrogression
36  Hl=tel_up-tel_down;              //loss of head in flood
37  Hl=round(Hl*100)/100;
38  mprintf("\nloss of head in at high flood=%f m.",Hl);
39  K=pl-cl;                        //head over crest
40  q_=1.7*(K)^1.5;
41  Hl_=pl-dhw;                      //loss of head
42  mprintf("\nloss of head=%f m.",Hl_);
43  Ef2=4.3;
44  Ef2_=1.7;                        //from Blench curve
45  jump=tel_down-Ef2;
46  jump_=251.5-Ef2_;       //level at which jump will
        form
47  Ef1=Ef2+Hl;
48  Ef1_=Ef2_+Hl_;
49  D1=1.03;
50  D1_=0.15;                        //calculated from Ef1 and
```

```scilab
      Ef1_ respectively
51  D2=3.96;D2_=1.68;           //calculated from Ef2 and
      Ef2_ respectively
52  hj=D2-D1;
53  hj_=D2_-D1_;                //heigth of jump
54  concrete=5*hj;
55  concrete_=5*hj_;            //length of concrete floor
56  mprintf("\n\nHydraulic jump calculation:");
57  mprintf("\nheigth of jump for high flood condition=
      %f m.",hj);
58  mprintf("\nlength of concrete floor for high flood
      condition=%f m.",concrete);
59  mprintf("\nheigth of jump for pond level condition=
      %f m.",hj_);
60  mprintf("\nlength of concrete floor for high pond
      level condition=%f m.",concrete_);
61
62  cw=2;                       //crets width
63  us=2;                       //upstream slope
64  ds=3;                       //downstream slope
65  l=15;
66  mprintf("\n\n upstream slope of glacis=%i:1.",us);
67  mprintf("\ndownstream slope of glacis=%i:1.",ds);
68  mprintf("\nhorizontal length of floor beyond the toe
      =%i m..",l);
69
70  R=6.5;
71  sh_up=hfl_up-1.5*R;
72  sh_down=hfl_down-2*R;
73  sh_up=round(sh_up*100)/100;
74  mprintf("\nR.L of bottom of upstream sheet pile=%f m
      .",sh_up);
75  mprintf("\nR.L of downstream sheet pile=%f m.",
      sh_down);
76  mprintf("\nprovide intermediate sheet pile at d/s
      toe of glacis.");
77  Hs=pl-249.6;                          //maximum
      percolation head
```

233

```scilab
78  d=249.6-sh_down;                              //depth of d/s
        cut-off
79  n=Ge*d/Hs;                                    //n=1/(%pi*
        lambda^0.5);
80  //from khosla exit gradient curve
81  alpha=1.5;
82  b=alpha*d;
83  mprintf("\n\nlength of impervious floor=%f m.",b);
84  fl=(2*(253-249.5))+2+(3*(253-249.6))+15;
85  us=36-fl;
86  mprintf("\nlength of floor already provide=%f m.",fl
        );
87  mprintf("\nwhich is more than required from
        permissible exit gradient.\nno upstream floor is
        required.");
88  mprintf("\nprovide %f m upstream floor so that total
         length becomes 36 m.",us);
89  alpha_1=0.089;
90  alpha_2=0.225;                    //alpha_=1/alpha
91  b1=21;
92  alpha=4.44;
93  mprintf("\n\nPressure percent at points:");
94  point=['C1' 'D1' 'C2' 'E2' 'D2' 'D3' 'E3'];
95  bc=[72 82 31.5 45.5 58.5 29 44];
96  crt=[3.1 0 3.5 0 -3.2 0 0 -3.6];
97  crs=[0 0 0 0 2.3 0 0 0];
98  cri=[3.7 0 6.4 0 -2.4 0 -6.4];
99  mprintf("\nPoints        Before correction
                After correction");
100 for i=1:7
101     after(i)=bc(i)+crt(i)+crs(i)+cri(i);
102     mprintf("\n%s                      %i
                               %f",point(i),bc(i),
        after(i));
103 end
104 Hs=254-249.6;                    //no flow condition
105 Hs_=256.13-254.5;                //high flood condition
106 Hs__=254-251.5;                  //flow at pond level
```

```
107  mprintf("\n\nelevation of subsoil H.G above datum:")
     ;
108  mprintf("\nno flow condition:");
109  fie1=1*Hs;
110  fid1=0.82*Hs;
111  fic1=0.788*Hs;
112  fie2=0.552*Hs;
113  fid2=0.455*Hs;
114  fic2=0.414*Hs;
115  fie3=0.34*Hs;
116  fid3=0.29*Hs;
117  fic3=0;
118  fie1=round(fie1*100)/100;fid1=round(fid1*100)/100;
     fic1=round(fic1*100)/100;
119  fie2=round(fie2*100)/100;fid2=round(fid2*100)/100;
     fic2=round(fic2*100)/100;
120  fie3=round(fie3*100)/100;fid3=round(fid3*100)/100;
     fic3=round(fic3*100)/100;
121  mprintf("\nfie1=%f.;fid1=%f.;fic1=%f.\nfie2=%f.;fid2
     =%f.;fic2=%f.\nfie3=%f.;fid3=%f.;fic3=%f.",fie1,
     fid1,fic1,fie2,fid2,fic2,fie3,fid3,fic3);
122  mprintf("\nhigh flood condition:");
123  fie1=1*Hs_;
124  fid1=0.82*Hs_;
125  fic1=0.788*Hs_;
126  fie2=0.552*Hs_;
127  fid2=0.455*Hs_;
128  fic2=0.414*Hs_;
129  fie3=0.34*Hs_;
130  fid3=0.29*Hs_;
131  fic3=0;
132  fie1=round(fie1*100)/100;fid1=round(fid1*100)/100;
     fic1=round(fic1*100)/100;
133  fie2=round(fie2*100)/100;fid2=round(fid2*100)/100;
     fic2=round(fic2*100)/100;
134  fie3=round(fie3*100)/100;fid3=round(fid3*100)/100;
     fic3=round(fic3*100)/100;
135  mprintf("\nfie1=%f.;fid1=%f.;fic1=%f.\nfie2=%f.;fid2
```

```
         =%f .; fic2=%f.\ nfie3=%f.; fid3=%f.; fic3=%f.",fie1,
         fid1,fic1,fie2,fid2,fic2,fie3,fid3,fic3);
136  mprintf("\nflow at pond level:");
137  fie1=1*Hs__;
138  fid1=0.82*Hs__;
139  fic1=0.788*Hs__;
140  fie2=0.552*Hs__;
141  fid2=0.455*Hs__;
142  fic2=0.414*Hs__;
143  fie3=0.34*Hs__;
144  fid3=0.29*Hs__;
145  fic3=0;
146  fie1=round(fie1*100)/100;fid1=round(fid1*100)/100;
         fic1=round(fic1*100)/100;
147  fie2=round(fie2*100)/100;fid2=round(fid2*100)/100;
         fic2=round(fic2*100)/100;
148  fie3=round(fie3*100)/100;fid3=round(fid3*100)/100;
         fic3=round(fic3*100)/100;
149  mprintf("\nfie1=%f.; fid1=%f.; fic1=%f.\ nfie2=%f.; fid2
         =%f .; fic2=%f.\ nfie3=%f.; fid3=%f.; fic3=%f.",fie1,
         fid1,fic1,fie2,fid2,fic2,fie3,fid3,fic3);
150
151  mprintf("\n\nPrejump profile:");
152  mprintf("\nhigh flood condition:");
153  dist=[3 6 8.4];                    //distance
154  glacis=[252 251 250.32];           //R.L of glacis
155  D1=[1.3 1.15 1.03];
156  mprintf("\nEf1              D1");
157  for i=1:3
158      Ef1(i)=256.25-glacis(i);
159      mprintf("\n%f          %f",Ef1(i),D1(i));
160  end
161  mprintf("\npond level flow:");
162  dist=[3 6 9 9.6];                  //distance
163  glacis=[252 251 250 249.9];        //R.Lof glacis
164  D1=[0.31 0.23 0.16 0.15];
165  mprintf("\nEf1              D1");
166  for i=1:4
```

```scilab
167      Ef1(i)=254-glacis(i);
168        mprintf("\n%f          %f",Ef1(i),D1(i));
169 end
170
171
172 rho=2.24;
173 Uf=4;                                  //unbalanced head
       for high flood condtion
174 Us=2.56;                               //unbalanced static
       head
175 Hf=2*Uf/3;
176 t=Hf/(rho-1);
177 t=round(t*10)/10;
178 mprintf("\n\nfloor thickness at the point of
       formation of hydraulic jump=%f m.",t);
179 Uf=2.9;                                //unbalanced head
       for high flood condtion
180 Us=2.2;                                //unbalanced static
       head
181 Hf=2*Uf/3;
182 t=Us/(rho-1);
183 t=round(t*10)/10;
184 mprintf("\nfloor thickness at the point of formation
        of hydraulic jump at the pond level condition=%f
       m.",t);
185 P=1.5;                                 //pressure head at d/s
        end of floor
186 t=P/(rho-1);
187 t=round(t*10)/10;
188 mprintf("\n\nfloor thickness at downstream side of
       sloping glacis=%f m.",t);
189 D=rb-sh_up;                            //depth of u/s scour
       hole above bed level
190 a=1.5*D;
191 a=round(a*10)/10;
192 mprintf("\n\nminimum length of upstream launching
       apron=%f m.",a);
193 mprintf("\nprovide 1.5 m thick apron for length of 5
```

```
          m." ) ;
194  D=249.6-241.5;
195  a=1.5*D;
196  mprintf(" \n\nminimum length of downstream launching
         apron=%f m." ,a);
197  mprintf(" \nprovide 1.5 m thick apron for length of
         12 m." );
```

Scilab code Exa 12.5 EX12 5

```
 1
 2
 3  //example 12.5
 4  //calculate uplift pressure at the junction of inner
         faces of pile with weir floor using Khosla
         theory
 5  clc;funcprot(0);
 6  //given
 7  b=16;           //total length of floor
 8  d=5;            //depth of downstream pile
 9  D=4;            //depth of upstream pile
10  H=2.5;          //head created by weir
11
12  //pressure at E
13  alpha=b/d;
14  lambda=(1+(1+alpha^2)^0.5)/2;
15  fie=acos((lambda-2)/lambda)/%pi;
16  C=19*(D/b)^0.5*((d+D)/b);
17  fie=fie*100-C;
18  P=H*fie/100;
19  P=round(P*1000)/1000;
20  mprintf("Pressure at E=%f m." ,P);
21
22  //pressure at C1
23  alpha=b/D;
```

```
24  lambda=(1+(1+alpha^2)^0.5)/2;
25  fie=acos((lambda-2)/lambda)/%pi;
26  fic=1-fie;                //by principle reversibility
        of flow
27  C=19*(d/b)^0.5*((d+D)/b);
28  fic=fic*100+C;
29  P=fic*H/100;
30  P=round(P*1000)/1000;
31  mprintf("\n Pressure at C=%f m.",P);
```

Scilab code Exa 12.6 EX12 6

```
1
2
3  //example 12.6
4  //calculate floor thickness at mid length and at
        junction with u/s and d/s cut-off walls
5  clc;funcprot(0);
6  //given
7  b=13;          //length of floor
8  d=2;           //depth of downstream wall
9  D=1.5;         //depth of upstream cut-off
10 rho=2.24;      //relative density
11 H=1.5;
12
13 //at junction of d/s cut-off with floor
14 alpha=b/d;
15 lambda=(1+(1+alpha^2)^0.5)/2;
16 fie=acos((lambda-2)/lambda)/%pi;
17 C=19*(D/b)^0.5*((d+D)/b);
18 fie=fie*100-C;
19 P=H*fie/100;
20 t=P/(rho-1);
21 t=round(t*10)/10;
22 mprintf("floor thickness at junction of d/s cut-off
```

```
        with floor=%f m.",t);
23
24  //at junction of u/s cut−off with floor
25  alpha=b/D;
26  lambda1=(1+(1+alpha^2)^0.5)/2;
27  fie=acos((lambda1-2)/lambda1)/%pi;
28  fic=1-fie;                    //by principle reversibility
        of flow
29  C=19*(D/b)^0.5*((d+D)/b);
30  fiec=fic*100+C;
31  P=fiec*H/100;
32  t=0.3;                        //this the uplift will be
        counter balanced by downward weigth of impounded
        water
33  mprintf("\nfloor thickness at junction of u/s cut−
        off with floor=%f m.",t);
34
35  //at mid−length
36  P=(1.08+0.489)/2;                    //assuming linear
        variation
37  t=P/(rho-1);
38  t=round(t*100)/100;
39  mprintf("\nfloor thickness at mid−length=%f m.",t);
40
41  //exit gradient
42  G=H/(d*%pi*(lambda)^0.5);
43  G=round(G*1000)/1000;
44  //since G<0.18
45  mprintf("\n G=%f. <0.18./nfloor is safe against
        failure by piping.",G);
```

Scilab code Exa 12.7 EX12 7

```
1
2
```

```
3  // example12.7
4  // calculate heigth of weir to be built
5  clc; funcprot (0);
6  // given
7  B=30;               // stream width
8  D=3;                // stream depth
9  V=1.25;             // mean velocity
10 Cd=0.95;            // discharge coefficient
11 Q=B*D*V;
12 C=2*Cd*(2*9.81)^0.5/3;
13 x=4-(Q/(C*B))^(2/3);
14 x=round(x*1000)/1000;
15 mprintf("heigth of weir to be built=%f m.",x);
```

Scilab code Exa 12.8 EX12 8

```
1
2
3  // example 12.8
4  // calculate uplift pressure at two cut−off
5  clc; funcprot (0);
6  // given
7  b=50;          // length of floor
8  d=8;           // depth of downstream pile
9  D=8;           // depth of upstream pile
10 H=5;           // effective head
11 tu=1;          // floor thickness at upstream
12 td=2;          // floor thickness at downstream
13
14 // downstream cut−off
15 alpha=b/d;
16 lambda=(1+(1+alpha^2)^0.5)/2;
17 fie=acos((lambda-2)/lambda)/%pi;
18 fid=acos((lambda-1)/lambda)/%pi;
19 Ct=(fie-fid)*td/d;
```

```
20  C=19*(D/b)^0.5*((d+D)/b);
21  fie=fie*100-C-Ct*100;
22  P=H*fie/100;
23  P=round(P*100)/100;
24  mprintf("Pressure at downstream cut-off=%f m.",P);
25
26  //upstream cut-off
27  fie=acos((lambda-2)/lambda)/%pi;
28  fid=acos((lambda-1)/lambda)/%pi;
29  fic1=1-fie;
30  fid1=1-fid;
31  Ct=(fic1-fid1)*td/d;
32  C=-19*(D/b)^0.5*((d+D)/b);
33  fic1=fic1*100-C-Ct*100;
34  P=H*fic1/100;
35  P=round(P*100)/100;
36  mprintf("\nPressure at upstream cut-off=%f m.",P);
37  G=H/(d*%pi*(lambda)^0.5);
38  mprintf("\nExit Gradient=%f.",G);
```

Scilab code Exa 12.9 EX12 9

```
1
2
3  //example 12.9
4  //calculate depth of downstream cut-off
5  clc;funcprot(0);
6  //given
7  Q=1000;          //discharge of river
8  L=256;           //crest length of diversion
9  f=1.1;           //silt factor
10 seg=1/6;         //safe exit gradient
11 hfl=103;         //high flood level
12 cf=100;          //reduced level of downstream
       concrete floor
```

```
13  H=2.4;              //maximum  static  head  of  weir
14  b=40;               //length  of  concrete  floor
15
16  q=Q/L;
17  R=1.35*(q^2/f)^(1/3);
18  rld=hfl-1.5*R;
19  d=cf-rld;
20  d=round(d*100)/100;
21  mprintf("depth  of  downstream  cut-off=%f m.",d);
22
23  alpha=b/d;
24  lambda=(1+(1+alpha^2)^0.5)/2;
25  G=H/(d*%pi*(lambda)^0.5);
26  //since  G<seg
27  mprintf("\n  G=%f.  <1/6./nfloor  is  safe  against
        failure  by  piping.",G);
```

Scilab code Exa 12.10 EX12 10

```
1
2
3  //example  12.10
4  //calculate  critical  exit  gradient  and  factor  of
       safety  of  system
5  clc;funcprot(0);
6  //given
7  b=60;          //length  of  floor
8  H=6;           //static  head  of  weir
9  d=6;           //downstream  depth  of  pile
10 n=0.3;         //porousity  of  soil  particles
11 G=2.7;         //relative  density  of  soil  particles
12
13 alpha=b/d;
14 lambda=(1+(1+alpha^2)^0.5)/2;
15 Ge=H/(d*%pi*(lambda)^0.5);
```

```
16  e=n/(1-n);
17  chg=(G-1)/(1+e);
18  f=chg/Ge;
19  f=round(f*100)/100;
20  mprintf("critical exit gradient=%f.\nfactor of
        safety of system=%f.",chg,f);
```

Scilab code Exa 12.11 EX12 11

```
1
2
3  //example 12.11
4  //design a vertical drop weir on Bligh's theory
5  //test floor by Khosla's theory
6  clc;funcprot(0);
7  //given
8  Q=2800;                      //maximum flood discharge
9  hfl=285;                     //H.F.L before
        construction
10  hw=278;                     //minimum water level
11  fsl=284;                    //F.S.L of canal
12  c=12;                       //coefficient of creep
13  flux=1;                     //allowable afflux
14  Ge=1/6;                     //permissible exit
        gradient
15  rho=2.24;                   //specific gravity of
        concrete
16
17  //Hydraulic calculation
18  L=4.75*Q^0.5;
19  q=Q/L;
20  q=round(q*10)/10;
21  mprintf("Hydraulic calculation:");
22  mprintf("\ndischarge per unit width of river=%f
        cumecs.",q);
```

```
23  f=1;
24  R=1.35*(q^2/f)^(1/3);
25  R=round(R*100)/100;
26  mprintf("\nregime scour depth=%f m.",R);
27  V=q/R;                          //regime velocity
28  vh=V^2/(2*9.81);           //velocity head
29  l_down=hfl+vh;
30  l_up=l_down+flux;
31  hfl_up=l_up-vh;
32  hfl_down=hfl-0.5;
33  hfl_down=round(hfl_down*100)/100;
34  mprintf("\nactual d/s H.F.L allowing 0.5 m for
        retrogation=%f m.",hfl_down);
35  K=(q/1.7)^(2/3);
36  cl=l_up-K;                      //crest level
37  cl=round(cl*100)/100;
38  mprintf("\ncrest level=%f m.",cl);
39  pl=fsl+0.5;                     //pond level
40  s=hfl_down-cl;                    //heigth of shutter
41  mprintf("\nheigth of shutter=%f m.",s);
42  rl_up_pile=hfl_up-1.5*R;     //R.L of bottom u/s pile
43  d_up_cut=hw-276;              //depth of upstream cut-
        off
44  mprintf("\ndepth of upstream cut-off=%f m.",d_up_cut
        );
45  mprintf("\n provide concrete cut off 2 m depth.");
46  rl_bot_ds=hfl_down-2*R;
47  Hs=hfl_down-hw;                 //seepage head
48  Hc=cl-hw;                       //heigth of crest
49  mprintf("\nR.L of gates crest=%f m.",Hs);
50  mprintf("\nHeigth of crest=%f m.",Hc);
51
52  //design of weir wall
53  d=hfl_up-cl;
54  a=d/(rho)^0.5;
55  a=3*d/(2*rho);                  //from sliding
        consideration
56  a=s+1;                          //from practical
```

```
        consieration
57  a=a+1;
58  mprintf("\n\ndesign of weir wall:")
59  mprintf("\nprovide top width of %i m.",a);
60  Mo=9.81*Hs^3/6;                    //overtirning moment
61  //equating the moment of resistance to overturning
        moment and putting the values we get
62  y=poly([-1.084,0.020,0.039],'x','c');
63  b=roots(y);
64  //we get b= - 5.5347261 and 5.0219056
65  //taking
66  b=5;
67  //when weir is submerged
68  C=0.58;
69  d=(q^2/((2*C/3)^2*2*9.81))^(1/3);
70  Mo=9.81*d*Hc^2/2;
71  //from equation of moment of resistence we get
72  y=poly([-77.55,3,1],'x','c');
73  b=roots(y);
74  //we get b= - 10.433085 and 7.4330846
75  //taking
76  b=8;
77  mprintf("\nbottom width=%i m.",b);
78
79  //design of impervious and pervious aprons
80  C=12;
81  L=C*Hs;
82  mprintf("\n\ndesign of impervious and pervious
        aprons:");
83  mprintf("\ntotal creep length=%i m.",L);
84  l1=2.21*C*(Hs/13)^0.5;
85  l1_=l1+1;
86  mprintf("\nlength of downstream impervious apron=%i
        m.",l1_);
87  d1=hw-276;
88  d2=hw-271;
89  l2=L-l1-(b+2*d1+2*d2);
90  mprintf("\nlength of upstream impervious apron=%i m.
```

246

```
      ",l2);
91  l3=18*C*(Hs*q/975)^0.5;
92  mprintf("\ntotal length of d/s apron=%i m.",l3);
                      //calculation is wrong in book
93  l=l3-l1;
94  le=l/2;
95  le=round(le*100)/100;
96  mprintf('\nprovide filter of length %f m. and
        launching apron of length %f m.',le,le);
97  t=d2*10^0.5/le;
98  mprintf("\nthickness of launching apron in
        horizontal position=%f m.",t);
99  mprintf("\nprovide launching apron of thickness 1.5
        m.");
100 T=2*d1;
101 V=d1*10^0.5;
102 ta=V/T;
103 ta=round(ta*10)/10;
104 mprintf("\nthickness of apron in horizontal position
        =%f m.",ta);
105 Hr=Hs-Hs*(4+33+8)/L;
106 t=4*Hr/(3*(rho-1));
107 t=round(t*10)/10;
108 mprintf('\nprovide thickness of %f m from d/s of
        weir wall to point 6 m from it.',t);
109 Hr=Hs-Hs*(4+33+8+6)/L;
110 t=4*Hr/(3*(rho-1));
111 t=round(t*10)/10;
112 mprintf("\nprovide thickness of %f m from 6 m to 12
        m from d/s end of weir wall.",t);
113 Hr=Hs-Hs*(4+33+8+12)/L;
114 t=4*Hr/(3*(rho-1));
115 t=round(t*10)/10;
116 mprintf("\nprovide thickness of %f m for rest of
        length of weir floor.",t);
117
118 //check by khosla's theory
119 b=33+8+19;                  //total horizontal length of
```

```
        impervious  floor
120 d=7;                          //depth  of  downstream  pile
121 alpha=b/d;
122 n=0.14;                       //n=1/%pi*(lambda)^0.5;
123 Ge=Hs*n/d;
124 mprintf("\n\ncheck  by  Khosla  theory:");
125 mprintf("\nexit  gradient=%f. < 1/6\n  hence  safe",Ge)
        ;
126 alpha_=d/b;
127 fic1=0.83; fid1=0.88;
128 corec_c1=(fid1-fic1)*100/2;
129 bdash=b;
130 d=2;D=7;
131 C1=19*(D/bdash)^0.5*(d+D)/b;
132 fic1=fic1*100+corec_c1+C1;
133 Pc=Hs*fic1/100;                            //pressure
        head  at  C
134 alpha_=d/b;
135 fie2=0.31; fid2=0.21;
136 corec_e1=(fie2-fid2)*1.7*100/7;
137 bdash=b;
138 d=7;D=2;
139 C1=19*(D/bdash)^0.5*(d+D)/b;
140 fie2=fie2*100-corec_e1-C1;                 //in  book
        3.53  value  is  wrong
141 Pe=Hs*fie2/100;                            //pressue
        head  at  E
142 //assuming  linear  variation  of  pressure  for
        intermediate  points
143 Pa=Pc-(Pc-Pe)*(33+8)/b;
144 t=Pa/1.24;
145 Pa=round(Pa*100)/100;
146 t=round(t*100)/100;
147 mprintf("\npressure  at  d/s  of  weir  wall=%f m.",Pa);
148 mprintf("\nthickness  at  d/s  of  weir  wall=%f m. <
        thickness  by  Bligh  theory;\nhence  safe.",t);
149 Pb=Pc-(Pc-Pe)*(33+8+6)/b;
150 t=Pb/1.24;
```

```
151  Pa=round(Pa*100)/100;
152  t=round(t*100)/100;
153  mprintf("\npressure at 6 m from d/s of weir wall=%f
         m.",Pb);
154  mprintf("\nthickness at 6m from d/s of weir wall=%f
         m. < thickness by Bligh theory;\nhence safe.",t);
155  Pc=Pc-(Pc-Pe)*(33+8+12)/b;
156  t=Pc/1.24;
157  Pa=round(Pa*100)/100;
158  t=round(t*100)/100;
159  mprintf("\npressure at 12 m from d/s of weir wall=%f
         m.",Pc);
160  mprintf("\nthickness at 12m from d/s of weir wall=%f
         m. > thickness by Bligh theory;\nhence unsafe.",
         t);
161  mprintf("\nhence increase th ethickness to 1.9 m for
          a length of 7 m of impervious floor.");
```

Scilab code Exa 12.12 EX12 12

```
 1
 2
 3  //example 12.12
 4  //calculate
 5  //number of gates required for the barrage
 6  //head regulator if each gate has 10 m clear span(
        neglect end contractions and approach velocity)
 7  //length and R.L of basin floor if silting basin is
        provided downstream of barrage
 8  clc;funcprot(0);
 9  //given
10  Lmax=212;              //maximum reservior level
11  Lp=211;                //pond level
12  hfl=210;               //downstream high flood level in
        the river
```

```scilab
13  Qmax =3500;                  //maximum design flood discharge
14  Lcrest =207;                 //crest level of the barrage
15  Lcrest_r =208;               //crest level of head regulator
16  Cd =2.1;                     //coefficient of discharge for
        barrage
17  Cd_r =1.5;                   //coefficient of discharge for
        head regulator
18  rbl =205;                    //river bed level
19  Q =500;                      //design discharge of main canal
20
21  //design of water way for barrage during flood
22  H=Lmax - Lcrest;
23  L=Qmax/(Cd*H^1.5);
24  //which gives L=149.07.
25  //provide 15 bays of 10m clear span
26  mprintf("nunmber of gates for the barrage=15.");
27
28  //design of waterway for canal head regulator
29  H=Lp - Lcrest_r;
30  L1=Q/(Cd_r*H^1.5);
31  //which gives L=64.2
32  //hence provide 7 bays of 10 m each
33  mprintf("\n\nnunmber of gates for the head regulator
        =7.");
34
35  //design of stilling basin
36  Hl=Lmax - hfl;
37  q=Qmax/L;
38  yc=(q^2/9.81)^(1/3);
39  Z=Hl/yc;
40  //since Z<1
41  Y=1+0.93556*Z^0.368;
42  y2=Y*yc;
43  Lc=5*y2;
44  Lc=round(Lc*10)/10;
45  mprintf("\n\nLength of cistern=%f m.",Lc);
46  Ef2=yc*(Y+1/(2*Y^2));
47  j=hfl - Ef2;
```

```
48  j=round(j*10)/10;
49  mprintf("\nR.L of cistern=%f m.",j);
```

Chapter 14

IRRIGATION CHANNEL 1 SILT THEORIES

Scilab code Exa 14.1 EX14 1

```
 1
 2
 3  //example 14.1
 4  //design irrigation channel on Kennedy's theory
 5  clc;funcprot(0);
 6  //given
 7  Q=45;          //discharge
 8  N=0.0225;      //rogosity coefficient
 9  m=1.05;        //critical velocity ratio
10  S=1/5000;      //bed slope
11
12  D=2;   //assume
13  Vo=0.55*m*D^0.64;
14  A=Q/Vo;
15  //for a trapezoidal section
16  B=(A-0.5*D^2)/2;
17  P=B+D*5^0.5;
18  R=A/P;
19  C=(23+1/N+0.00155/S)*(R*S)^0.5/(1+(23+0.00155/S)*N/R
```

```
        ^0.5);
20  V=C*(R*S)^0.5;
21  //Vo<V
22
23  //assume D=2.2
24  D=2.2;
25  Vo=0.55*m*D^0.64;
26  A=Q/Vo;
27  B=(A-0.5*D^2)/D;
28  P=B+D*5^0.5;
29  R=A/P;
30  C=(23+1/N+0.00155/S)*(R*S)^0.5/(1+(23+0.00155/S)*N/R
        ^0.5);
31  V=C*(R*S)^0.5;
32
33  //ratio of V and Vo is almost equal to 1
34  B=round(B*10)/10;
35   mprintf("Width of channel section=%f m.",B);
36  mprintf("\nDepth of channel section=%f m.",D);
```

Scilab code Exa 14.2 EX14 2

```
1
2
3  //example 14.2
4  //design an irrigation canal for given data
5  clc;funcprot(0);
6  //given
7  Q=14;          //discharge
8  m=1;           //critical velocity ratio
9  r=5.7;         //B/D
10
11  D=(Q/(0.55*6.2))^(1/2.64);
12  B=D*r;
13  R=(B*D+D^2/2)/(B+D*5^0.5);
```

```
14  Vo =0.55* m*D ^0.64;
15  //applying kutters formula; V=C(RS)^0.5
16  //where C=(23+1/N+0.00155/S)*(R*S)
        ^0.5/(1+(23+0.00155/S)*N/R^0.5);
17  //assuming S^0.5=y
18  y=poly([-1.98D-5,1.55D-3,-0.954,67.5],'x','c');
19  roots(y);
20  //taking real values of y
21  S=0.0139906^2;
22  B=round(B*100)/100;
23  D=round(D*100)/100;
24  mprintf("Width of channel section=%f m.",B);
25  mprintf("\nDepth of channel section=%f m.",D);
26  mprintf("\nBed slope=%f.",S);
```

Scilab code Exa 14.3 EX14 3

```
 1
 2
 3  //example 14.3
 4  //design a channel on Kennedy's theory
 5  clc;funcprot(0);
 6  //given
 7  Q=45;                //discharge
 8  m=1.05;              //critical velocity ratio
 9  N=0.025;             //rugosity coefficient
10  S=1/5000;            //bed slope
11
12  l=S*Q^0.02/(N^2*m^2.02);
13  //from fig.14.3 we get r=10
14  //solving the equation by trial and error method we
        get
15  r=9.7;
16  D=(1.818*Q/(m*(r+0.5)))^(1/2.64);
17  B=r*D;
```

254

```
18  V=Q/(D^2*(r+0.5));
19  Vo=0.55*D^0.64*m;
20  B=round(B);
21  D=round(D*100)/100;
22  V=round(V*1000)/1000;
23  Vo=round(Vo*1000)/1000;
24  mprintf("Width of channel section=%f m.",B);
25  mprintf("\nDepth of channel section=%f m.",D);
26  mprintf("\nVelocity through the channel section=%f m
        /s.",V);
27  mprintf("\nVo=%f m/s.\nHence Safe",Vo);
```

Scilab code Exa 14.4 EX14 4

```
 1
 2
 3  //example 14.4
 4  //design channel using method of curve fitting based
        onKennedy's theory
 5  clc;funcprot(0);
 6  //given
 7  Q=45;            //discharge
 8  N=0.0225;        //rugosity coefficient
 9  m=1.05;          //critical velocity ratio
10  S=1/5000;        //Bed slope
11
12  r=(1.607*S^1.63*Q^0.033/(N^3.26*m^3.293)-0.258)
        ^(-0.915);
13  D=(1.818*Q/(m*(r+0.5)))^(1/2.64);
14  B=r*D;
15  B=round(B);
16  D=round(D*100)/100;
17  mprintf("Width of channel section=%f m.",B);
18  mprintf("\nDepth of channel section=%f m.",D);
```

Scilab code Exa 14.5 EX14 5

```
1
2  //example 14.5
3  //design channel using curve of CWPC for B/D ratio
4  clc;funcprot(0);
5  //given
6  Q=45;           //discharge
7  N=0.0225;       //rugosity coefficient
8  m=1.05;         //critical velocity ratio
9
10 r=(15+6.44*Q)^0.382;
11 S=(N^2/1.338*Q^0.02)*(0.258+(15+6.44*Q)^(-0.417))
      ^0.6135;
12 D=(1.818*Q/(m*(r+0.5)))^(1/2.64);
13 B=r*D;
14 B=round(B);
15 D=round(D*100)/100;
16 mprintf("Bed slope=%f.",S);
17 mprintf("\nWidth of channel section=%f m.",B);
18 mprintf("\nDepth of channel section=%f m.",D);
```

Scilab code Exa 14.6 EX14 6

```
1
2
3  //example 14.6
4  //design the channel section using the following
      data and calculate logitudnal section
5  clc;funcprot(0);
6  //given
7  Q=30;        //discharge
```

```
 8  f =1;          // silt  factor
 9  s =1/2;        // side  slope
10
11  V =( Q * f /140) ^(1/6) ;
12  A = Q / V ;
13  P =4.75* Q ^0.5;
14  D =( P -( P ^2 -6.944* A ) ^0.5) /3.472;
15  B = P -2.236* D ;
16
17  R =5* V ^2/(2* f ) ;
18  S = f ^(5/3) /(3340* Q ^(1/6) ) ;
19  B = round ( B *100) /100;
20  D = round ( D *100) /100;
21  mprintf (" Bed  slope=%f." , S ) ;
22  mprintf ("\ nWidth  of  channel  section=%f  m." , B ) ;
23  mprintf ("\ nDepth  of  channel  section=%f  m." , D ) ;
```

Scilab code Exa 14.7 EX14 7

```
 1
 2
 3  // example  14.7
 4  // design  a  channel  in  alluvial  soil  using  tractive
         force  approach
 5  clc ; funcprot (0) ;
 6  // given
 7  Q =45;             // discharge
 8  S =1/4800;         // bed  slope
 9  N =0.0225;         // rogosity  coefficient
10  sigma =0.0035;     // permissible  tractive  stress
11  s =1/2;            // side  slope
12  gamma_w =9.81;     // unit  weigth  of  water
13
14  R = sigma /( gamma_w * S ) ;
15  V = R ^(2/3) * S ^0.5/ N ;
```

```
16  A = Q / V ;
17  P = A / R ;
18  y = poly ([ -49 ,28.61 , -1.736] , 'x ', 'c ') ;
19  D = roots ( y ) ;
20  // we get D=14.539034 and 1.9413812
21  // taking D=1.9413812
22  D =1.9413812;
23  B =28.61 -2.23* D ;
24  B = round ( B *100) /100;
25  D = round ( D *100) /100;
26  mprintf (" Width of channel section=%f m." ,B ) ;
27  mprintf ("\ nDepth of channel section=%f m." ,D ) ;
```

Scilab code Exa 14.8 EX14 8

```
 1
 2
 3  // example 14.8
 4  // designa channel section by Kennedy theory
 5  clc ; funcprot (0) ;
 6  // given
 7  Q =28;          // discharge
 8  m =1;           // critical velocity ratio
 9  r =7.6;         //B/D
10
11  D =( Q /4.46) ^(1/2.64) ;
12  B = r * D ;
13  R =0.823* D ;
14  V =0.55*( D ) ^0.64;
15
16  // applying kutters formula ; V=C(RS) ^0.5
17  // where C=(23+1/N+0.00155/S) *(R*S)
        ^0.5/(1+(23+0.00155/S) *N/R^0.5) ;
18  // we get equation in S
19  // assuming S^0.5=y
```

258

```
20  y=poly ([ -1.42D -5 ,1.55D -3 , -0.885 ,67.4] , 'x ', 'c ') ;
21  roots (y) ;
22  // taking  real  values  of  y
23  S =0.0126305^2;
24  B= round (B *10) /10;
25  D= round (D *100) /100;
26  mprintf (" Width  of  channel  section=%f m." ,B) ;
27  mprintf ("\nDepth  of  channel  section=%f m." ,D) ;
28  mprintf ("\nBed  slope=%f." ,S) ;
```

Scilab code Exa 14.9 EX14 9

```
 1
 2
 3  // example  14.9
 4  // design  the  channel  section  and  calculate  discharge
 5  clc ; funcprot (0) ;
 6  // given
 7  r =5.7;           //B/D
 8  S =1/5000;        // bed  slope
 9  N =0.0225;        // rogosity  coefficient
10  m =1;             // critical  velocity  ratio (assumed )
11
12  // applying  kutters  formula ;  V=C(RS) ^0.5
13  // where  C=(23+1/N+0.00155/S) *(R*S)
        ^0.5 /(1+(23+0.00155/S) *N/R^0.5) ;
14  // we  get  equation  in  d  as
15  // 38.88*D^0.64 -66.5*D^0.5+30.37*D^0.14=0
16  // solving  it  by  trial  and  error  method
17  // we  get  D=1.7 m.
18  D =1.7;
19  B=r*D;
20  V =0.55*m*(D) ^0.64;
21  A=B*D+D^2/2;
22  Q=A*V;
```

259

```
23  Q=round(Q*100)/100;
24  mprintf("Width of channel section=%f m.",B);
25  mprintf("\nDepth of channel section=%f m.",D);
26  mprintf("\n Discharge=%f cumecs.",Q);
```

Scilab code Exa 14.10 EX14 10

```
1
2
3  //example 14.10
4  //design irrigation channel  according to Laecy silt
        theory
5  clc;funcprot(0);
6  //given
7  Q=15;                    //discharge
8  f=1;                     //laecy silt factor
9  s=1/2;                   //channel side slope
10
11  V=(Q*f^2/140);
12  A=Q/V;
13  R=5*V^2/(2*f);
14  //using the value of A in equations we get,
15  //equation in D as
16  y=poly([-21.765,18.336,-1.73],'x','c');
17  D=roots(y);
18  //we get D=9.2368003 and 1.3620436.
19  //taking
20  D=1.3620436;
21  B=18.336-D*2.23;
22  P=4.75*Q^0.5;
23  S=1/(3340*Q^(1/6));
24  B=round(B*10)/10;
25  D=round(D*100)/100;
26  mprintf("Width of channel section=%f m.",B);
27  mprintf("\nDepth of channel section=%f m.",D);
```

```
28  mprintf ("\nBed  slope=%f.",S);
```

Scilab code Exa 14.11 EX14 11

```
 1
 2
 3  //example  14.11
 4  //find  channel  section  and  discharge
 5  clc;funcprot(0);
 6  //given
 7  S=1/5000;                  //bed  slope
 8  s=1/2;                     //side  slope
 9  f=0.9;                          //laecy  silt  factor
10
11  Q=(f^(5/3)/(3340*S))^6;
12  R=f^3/(4980*S)^2;
13  P=4.75*Q^0.5;
14  A=P*R;
15  //using  the  value  of  A and  P  in  equations  we  get,
16  //equation  in  D as
17  y=poly([-6.961,9.41,-1.73],'x','c');
18  D=roots(y);
19  //we  get  D=4.5561754  and  0.8831309.
20  //taking
21  D=0.8831309;
22  B=9.41-D*2.23;
23  B=round(B*100)*100;
24  D=round(D*100)/100;
25  Q=round(Q*1000)/1000;
26  mprintf("Width  of  channel  section=%f m.",B);
27  mprintf("\nDepth  of  channel  section=%f m.",D);
28  mprintf("\n Discharge=%f cumecs.",Q);
```

Scilab code Exa 14.12 EX14 12

```scilab
1
2
3 //example 14.12
4 //calculate quantity of bed load moved by the Meyer-
      Peter equation
5 clc;funcprot(0);
6 //given
7 gamma_w=9.81;            //unit weigth of water
8 D=3;                     //depth of channel
9 d=0.3;                   //grain size
10 k=1.5;                  //size of roughness of channel
       bed
11 S=1/4400;               //bed slope
12 G=2.65;                 //specific gravity
13 tau_b=gamma_w*D*S;
14 N1=d^(1/6)/24;
15 N=k^(1/6)/24;
16 gamma_s=gamma_w*G;
17 tau_c=0.047*(gamma_s-gamma_w)*d/1000;
18 r=(N1/N)^1.5;
19 q=47450*(tau_b*r-tau_c)^1.5;
20 q=round(q*100)/100;
21 mprintf("quantity of bed load moved=%f kN/m/hr.",q);
```

Scilab code Exa 14.13 EX14 13

```scilab
1
2
3 //example 14.13
4 //calculate bed load transported by channel by
      einstein equation
5 clc;funcprot(0);
6 //given
```

262

```
7  gamma_w =9.81;              //unit  weigth  of  water
8  D =3;                       //depth  of  channel
9  d =0.3;                     //grain  size
10 k =1.5;                     //size  of  roughness  of  channel
        bed
11 S =1/4400;                  //bed  slope
12 G =2.65;                    //specific  gravity
13
14 N1 =d ^(1/6) /24;
15 N =k ^(1/6) /24;
16 r =(N1/N) ^1.5;
17 R1 =3*r;
18 si =(G -1) *d /(1000* R1*S);
19 //hence  we  get
20 fi =7;
21 q =3600* fi *G* gamma_w *(G -1) ^0.5*( gamma_w ) ^0.5*(d /1000)
        ^1.5;
22 q = round (q *10) /10;
23 mprintf (" quantity  of  bed  load  moved=%f  kN/m/ hr . " ,q) ;
```

Scilab code Exa 14.14 EX14 14

```
1
2
3  //example  14.14
4  //calculate  concentration  of  suspended  load
5  clc ; funcprot (0) ;
6  //given
7  gamma_w =9.81;              //unit  weigth  of  water
8  D =3;                       //depth  of  channel
9  d =0.3;                     //grain  size
10 k =1.5;                     //size  of  roughness  of  channel
        bed
11 S =1/4400;                  //bed  slope
12 G =2.65;                    //specific  gravity
```

```
13  V=0.03;                    //fall velocity
14  c_=400;                        //concentration at 0.3 m above
        bed
15  a=0.3;
16  y=1;
17  k_=0.4;                   //van karman's constant
18
19  N1=d^(1/6)/24;
20  N=k^(1/6)/24;
21  r=(N1/N)^1.5;
22  R1=3*r;
23  V_=(gamma_w*R1*S)^0.5;
24  c=c_*((a/y)*(D-y)/(D-a))^(V/(V_*k_));
25  c=round(c*10)/10;
26  mprintf("concentration of suspended load=%f ppm.",c)
        ;
```

Scilab code Exa 14.15 EX14 15

```
1
2
3  //example 14.15
4  //design an irrigation channel by Meyer peter
        equation
5  clc;funcprot(0);
6  //given
7  Q=45;                //discharge
8  c=55;                //bed load concentraion
9  d=0.3;               //average grain diameter
10  gamma_w=9.81;        //unit weigth of water
11  G=2.67;
12  f=0.964;
13
14  c=c*Q*gamma_w*3600/1000000;
15  P=4.75*Q^0.5;
```

```
16 //taking channel width as B=28 m(slightly less than
       P)
17 B=28;
18 qs=c/B;
19 //assuming effective grain diameter k=0.4 mm
20 ks=0.4D-3;
21 N1=ks^(1/6)/24;
22 sf=1.76*d^0.5;
23 N=0.0225*sf^0.25;
24 r=N1/N;
25 tau_c=0.047*gamma_w*(G-1)*d/1000;
26 tau_b=r^1.5*((qs/47450)^(2/3)+tau_c);
27 //from Manning's formula we get on simplification
28 R=(0.000992*1000/0.525)^(3/7);
29 S=0.525/(1000*R);
30 //solving equation of R for trapezoidal section of
       side slope 1/2 we get
31 y=poly([-36.792,25.06,0.5],'x','c');
32 D=roots(y);
33 //we get D= -51.547499  and 1.4274989
34 //taking
35 D=1.4274989;
36 D=round(D*100)/100;
37 mprintf("Width of channel section=%i m.",B);
38 mprintf("\nDepth of channel section=%f m.",D);
39 mprintf("\nBed slope=%f.",S);
```

Scilab code Exa 14.16 EX14 16

```
1
2
3 //example 14.16
4 //design an irrigation channel by Einstein equation
5 clc;funcprot(0);
6 //given
```

```
 7  Q =45;                    // discharge
 8  c =55;                    // bed load concentraion
 9  d =0.3;                   // average grain diameter
10  gamma_w =9.81;            // unit weigth of water
11  G =2.67;                  // specific gravity of soil
12  f =0.964;                 // silt factor
13
14  // taking channel width as B=28 m( slightly less than
       P)
15  B =28;
16  qs = c / B ;
17
18  fi =( qs /( gamma_w * G ) ) *(1/( G -1) ) ^0.5*(1000000000/(
       gamma_w * d ^3) ) ^0.5;
19  // from fig. 14.6 we get value of sci
20  // using the sci equation and Manning formula and on
       simplifications we get
21  R =(2.4296) ^(3/7) ;
22  S =0.4083/(1000*1.463) ;
23  // solving equation of R for trapezoidal section of
       side slope 1/2 we get
24  y = poly ([ -40.96 ,24.73 ,0.5] , 'x ', 'c ') ;
25  D = roots ( y ) ;
26  // we get D= -51.064253    and 1.6042534
27  // taking
28  D =1.6042534;
29  D = round ( D *10) /10;
30  mprintf (" Width of channel section=%i m. ",B ) ;
31  mprintf (" \nDepth of channel section=%f m. ",D ) ;
32  mprintf (" \nBed slope=%f. ",S ) ;
```

Scilab code Exa 14.17 EX14 17

```
1
2
```

266

```
3  //example 14.17
4  //design a channel for non−alluvial deposites
5  clc;funcprot(0);
6  //given
7  Q=45;                    //discharge
8  S=1/4000;                //bed slope
9  v=0.9;                   //permissible velocity
10 N=0.025;                 //rogosity coefficient
11
12 A=Q/v;
13 R=(v*N/S^0.5)^1.5;
14 P=A/R;
15 //let us provide a trapezoidal section
16 //from equation of Area and Perimeter of trapezoid
17 y=poly([50,-29.45,1.828],'x','c');
18 D=roots(y);
19 //from which we get D=14.181815 and 1.9286881
20 //taking
21 D=1.9286881;
22 B=P-2*1.41*D;
23 D=round(D*100)/100;
24 mprintf("Width of channel section=%i m.",B);
25 mprintf("\nDepth of channel section=%f m.",D);
```

Scilab code Exa 14.18 EX14 18

```
1
2
3  //example14.18
4  //design non−allvial channel using Bazin's formula
5  clc;funcprot(0);
6  //given
7  Q=15;                    //discharge
8  V=0.75;                  //mean velocity
9  s=1;                     //side slope
```

```
10  K =1.3;                    // bazin 's  coefficient
11  // width  is  five  times  its  depth
12
13  A = Q / V ;
14  D =( A /6) ^0.5;
15  B =5* D ;
16  P = B +2* D *1.41;
17  R = A / P ;
18  C =87/(1+ K /( R ) ^0.5) ;
19  S =( V / C ) ^2/ R ;
20  B = round ( B *10) /10;
21  D = round ( D *100) /100;
22  mprintf (" Width  of  channel  section=%f m." , B );
23  mprintf ("\nDepth  of  channel  section=%f m." , D );
24  mprintf ("\nBed  slope=%f." , S );
```

Scilab code Exa 14.19 EX14 19

```
1
2
3
4  // example14 .19
5  // determine  dimension  of  channel  using  chezy 's
        equation
6  // calculate  the  value  of  manning  n
7  clc ; funcprot (0) ;
8  // given
9  Q =21.5;            // discharge
10  S =1/2500;          // slope  of  bottom
11  C =70;
12  r =1/1.73;
13  // taking  R=0.5*D
14  // and  keeping  it  in  Q=V*A; where  V=C(RS) ^0.5  and  A=D
        ^2(2*(4/3) ^0.5 −1/3^0.5) ;
15  D =(21.5/1.7146) ^(1/2.5) ;
```

```
16  B=2*D*((4/3)^0.5-(1/3)^0.5);
17  B=round(B*100)/100;
18  D=round(D*100)/100;
19  mprintf("side slope=%f.",r);
20  mprintf("\nWidth of channel section=%f m.",B);
21  mprintf("\nDepth of channel section=%f m.",D);
22
23  R=0.5*D;
24  V=C*(R*S)^0.5;
25  n=R^(2/3)*S^0.5/V;
26  n=round(n*1000)/1000;
27  mprintf("\n\nvalue of manning n=%f.",n);
```

Scilab code Exa 14.20 EX14 20

```
1
2
3  //example14.20
4  //design a regime channel
5  clc;funcprot(0);
6  //given
7  Q=100;              //discharge
8  f=1.1;              //silt factor
9  s=1/2;              //side slope
10
11  V=(Q*f^2/140)^(1/6);
12  A=Q/V;
13  P=4.75*Q^0.5;
14  D=(P-(P^2-6.944*A)^0.5)/3.472;
15  B=P-2.236*D;
16  R=5*V^2/(2*f);
17  S=f^(5/3)/(3340*Q^(1/6));
18  B=round(B*10)/10;
19  D=round(D*100)/100;
20  mprintf("Width of channel section=%f m.",B);
```

```
21  mprintf("\nDepth of channel section=%f m.",D);
22  mprintf("\nBed slope=%f.",S);
```

Scilab code Exa 14.21 EX14 21

```
1
2
3  //example 14.21
4  //design a channel using Laecy theory
5  clc;funcprot(0);
6  //given
7  Q=40;                    //discharge
8  s=1;                     //side slope
9  md=0.8;                  //average size of base material
10
11  f=1.76*(md)^0.5;
12  V=(Q*f^2/140)^(1/6);
13  A=Q/V;
14  P=4.75*Q^0.5;
15  //from equations of Area and perimeter of
        trapezoidal section;we get
16  y=poly([42.41,-30.04,1.828],'x','c');
17  D=roots(y);
18  //we get D=14.873416 and 1.5598447
19  //taking
20  D=1.5598447;
21  B=A/D-D;
22  R=5*V^2/(2*f);
23  S=f^(5/3)/(3340*Q^(1/6));
24  B=round(B*100)/100;
25  D=round(D*100)/100;
26  mprintf("Width of channel section=%f m.",B);
27  mprintf("\nDepth of channel section=%f m.",D);
28  mprintf("\nBed slope=%f.",S);
```

Scilab code Exa 14.22 EX14 22

```
1
2
3  //example14.22
4  //calculate bed width and floor depth
5  clc;funcprot(0);
6  //given
7  Q=30;                    //discharge
8  V=1;                     //velocity of flow
9
10 A=Q/V;
11 //perimeter of section=30/D--D/2
12 //taking its derivative w.r.t to D
13 D=1/((1.914/30)^0.5);
14 //from equation of area
15 B=30/D-D/2;
16 B=round(B*10)/10;
17 D=round(D*100)/100;
18 mprintf("Width of channel section=%f m.",B);
19 mprintf("\nDepth of channel section=%f m.",D);
```

Scilab code Exa 14.23 EX14 23

```
1
2  //example 14.23
3  //determine whether flow is critical or sub-critical
4  clc;funcprot(0);
5  //given
6  Q=17;           //discharge
7  B=6;            //base of channel
8  s=1/2;          //side slope
```

```
 9  D=1.5;                // depth  of  channel
10
11  A=D*((B+B/s)/2);
12  V=Q/A;
13  P=B+2*((D/s)^2+D^2)^0.5;
14  R=A/P;
15  F=V/(9.81*R)^0.5;           // froud  number
16  F=round(F*100)/100;
17  // since  F<1;
18  mprintf("Froud  number=%f.\nF<1.\nFlow  is  sub-
        critical",F);
```

Scilab code Exa 14.24 EX14 24

```
 1
 2
 3  // example14.24
 4  // calculate  normal  depth  and  average  shear  stress  at
        channel  bed
 5  clc;funcprot(0);
 6  // given
 7  B=3.5;              // bottom  width  of  channel
 8  n=0.016;            // manning  n
 9  S=2.6/10000;        // bed  slope
10  Q=8;                // discharge
11  lfs=1;              // left  side  slope
12  rhs=1.5;            // rigth  side  slope
13  gamma_w=9.81;       // unit  weigth  of  water
14
15  // using  the  equation  of  area  and  perimeter  of
        trapezoidal  section;Manning's  formula  and  V=Q/A
        we  get  D  as
16  // Manning  formula:  V=R^(2/3)*S^0.5/n
17  // (D*(3.5+1.25*D))^2.5=78.281+71.951*D
18  // solving  it  by  trial  and  error  method;we  get
```

272

```
19  D =1.5;
20  R=(D*(3.5+1.25*D))/(3.5+3.217*D);
21  tau = gamma_w * R * S *1000;
22  tau = round ( tau *100) /100;
23  mprintf ( " Depth  of  section =%f m. " , D );
24  mprintf ( " \ nAverage  shear  stress  at  channel  bed =%f N/
        square-mm. " , tau );
```

Scilab code Exa 14.25 EX14 25

```
1
2
3  // example  14.25
4  // calculate  bed  load  transported  by  the  channel  in
        tonnes  per  day
5  clc ; funcprot (0) ;
6  // given
7  S =1/5000;            // bed  slope
8  B =40;                // width  of  channel
9  D =2.6;               // depth  of  channel
10 d =0.38;              // mean  diameter  of  bed  material
11 n =0.021;             // Manning  n
12 D65 =0.64D -3;        // bed  material  size (m)
13 w =1000;              // density  of  water
14 //B/D  as  large  tau_c =0.075*d;
15 tau_c =0.075* d;
16 tau_b = w * D * S;
17 N1 =( D65 ) ^(1/6) /24;
18 r = N1 /n;
19 qs =4700*24*( tau_b * r ^1.5 - tau_c ) ^1.5/1000;
20 qs40 = qs *40;
21 mprintf ( " bed  load  transported  by  the  channel  =%i t/m
        / day . " , qs40 );
```

Scilab code Exa 14.26 EX14 26

```scilab
1
2
3  //example 14.26
4  //calculate bed width of channel;also check depth
       using Kutter equation
5  clc;funcprot(0);
6  //given
7  Q=5;                    //discharge
8  S=0.2/1000;             //bed slope
9  m=0.8;                  //critical velocity ratio
10 s=1/2;                  //side slope of chanel
11 C=30;
12 //assuming
13 D=1;
14 Vo=0.55*m*D^0.64;
15 A=Q/Vo;
16 B=A/D-(s*D);
17 P=B+2.43*D;
18 R=A/P;
19 V=C*(R*S)^0.5;
20 //Vo>V
21 //hence take second trial
22 D=0.8;       //assume
23 Vo=0.55*m*D^0.64;
24 A=Q/Vo;
25 B=A/D-(s*D);
26 P=B+2.43*D;
27 R=A/P;
28 V=C*(R*S)^0.5;
29 //again Vo>V
30 //hence we take third trial
31 D=0.7;
```

```
32  Vo =0.55* m*D ^0.64;
33  A = Q / Vo ;
34  B = A / D +( s * D ) ;
35  P = B +2.43* D ;
36  R = A / P ;
37  V = C *( R * S ) ^0.5;
38  B = round ( B *100) /100;
39  // Vo  is  almost  equal  to  V ;
40  mprintf ( " Width  of  channel  section =% f m . " , B ) ;
41  mprintf ( " \ nDepth  of  channel  section =% f m . " , D ) ;
```

Scilab code Exa 14.27 EX14 27

```
 1
 2
 3  // example  14.27
 4  // design  irrigation  channel  by  Kennedy  method
 5  clc ; funcprot (0) ;
 6  // given
 7  Q =50;            // discharge
 8  r =2.5;           // B / D  ratio
 9  m =1.1;           // critical  velocity  ratio
10  N =0.025;         // rogosity  coefficient
11  s =0.5;           // side  slope  of  channel
12
13  // using  the  equation  of  Vo  and  Q = A * V ; we  get
14  D =( Q /1.815) ^(1/2.64) ;
15  B = r * D ;
16  R =( B * D +0.5* D ^2) /( B +2.236* D ) ;
17  Vo =0.55* m * D ^0.64;
18
19  // applying  kutters  formula ;  V = C ( RS ) ^0.5
20  // where  C =(23+1/ N +0.00155/ S ) *( R * S )
        ^0.5/(1+(23+0.00155/ S ) * N / R ^0.5) ;
21  // assuming  S ^0.5= y
```

275

```
22  y=poly ([ -3.737D-7 ,2.46D-5 ,-0.0199 ,1] ,'x','c');
23  roots(y);
24  // taking  real  values  of  y
25  S=0.0196171 ^2;
26  B=round(B*100)/100;
27  D=round(D*100)/100;
28  mprintf("Width  of  channel  section=%f m.",B);
29  mprintf("\nDepth  of  channel  section=%f m.",D);
30  mprintf("\nBed  slope=%f.",S);
```

Scilab code Exa 14.28 EX14 28

```
 1
 2
 3  // example  14.28
 4  // design  a  regime  channel  using  Laecy's  theory
 5  clc;funcprot(0);
 6  // given
 7  Q=35;              // discharge
 8  f=0.9;             // silt  factor
 9  s=1/2;             // side  slope
10
11  V=(Q*f/140)^(1/6);
12  A=Q/V;
13  P=4.75*Q^0.5;
14  D=(P-(P^2-6.944*A)^0.5)/3.472;
15  B=P-2.236*D;
16
17  R=5*V^2/(2*f);
18  S=f^(5/3)/(3340*Q^(1/6));
19  D=round(D*100)/100;
20  mprintf("Bed  slope=%f.",S);
21  mprintf("\nWidth  of  channel  section=%i m.",B);
22  mprintf("\nDepth  of  channel  section=%f m.",D);
```

Scilab code Exa 14.29 EX14 29

```
1
2
3  //example 14.29
4  //design an irrigation canal for given data
5  clc;funcprot(0);
6  //given
7  Q=15;           //discharge
8  m=1;            //critical velocity ratio
9  r=5.7;          //B/D
10
11 D=(Q/(0.55*6.2))^(1/2.64);
12 B=D*r;
13 R=(B*D+D^2/2)/(B+D*5^0.5);
14 Vo=0.55*m*D^0.64;
15 //applying kutters formula; V=C(RS)^0.5
16 //where C=(23+1/N+0.00155/S)*(R*S)
      ^0.5/(1+(23+0.00155/S)*N/R^0.5);
17 //assuming S^0.5=y
18 y=poly([-2D-5,1.55D-3,-0.968,67.5],'x','c');
19 roots(y);
20 //taking real values of y
21 S=0.0141937^2;
22 B=round(B*100)/100;
23 D=round(D*100)/100;
24 mprintf("Width of channel section=%f m.",B);
25 mprintf("\nDepth of channel section=%f m.",D);
26 mprintf("\nBed slope=%f.",S);
```

Scilab code Exa 14.30 EX14 30

```
 1
 2
 3  //example 14.30
 4  //Design a section of unlined canal in a loomy soil
 5  clc;funcprot(0);
 6  //given
 7  Q=50;                    //discharge
 8  V=1;                     //permissible velocity
 9  s=2;                     //side slope
10  r=6;                     //B/D ratio
11  N=0.0225;                //rogosity coefficient
12
13  A=Q/V;
14  D=(A/(r+2))^0.5;
15  B=r*D;
16  P=B+2*(5*D^2)^0.5;
17  R=A/P;
18  S=(V*N/R^(2/3))^2;
19  mprintf("Width of channel section=%i m.",B);
20  mprintf("\nDepth of channel section=%f m.",D);
21  mprintf("\nBed slope=%f.",S);
```

Scilab code Exa 14.31 EX14 31

```
 1
 2
 3  //example 14.31
 4  //calculate concentration of suspended load at depth
 5  clc;funcprot(0);
 6  //given
 7  gamma_w=9.81;            //unit weigth of water
 8  D=5;                     //depth of channel
 9  d=0.3;                   //grain size
10  k=1.5;                   //size of roughness of channel
        bed
```

```
11  S=1/4000;                    //bed slope
12  G=2.65;                      //specific gravity
13  V=0.02;                      //fall velocity
14  c_=1000;                        //concentration at 0.3 m
       above bed
15  a=0.3;
16  y=2.5;
17  k_=0.4;                    //van karman's constant
18
19  R=5;                       //R=D for wide channel
20  V_=(gamma_w*R*S)^0.5;
21  c=c_*((a/y)*(D-y)/(D-a))^(V/(V_*k_));
22  mprintf("concentration of suspended load=%i ppm.",c)
       ;
```

Scilab code Exa 14.32 EX14 32

```
 1
 2
 3  //example 14.32
 4  //calculate dimension of channel if it is design on
      the basis of Laecy theory and Kennedy's theory
 5  clc;funcprot(0);
 6  //given
 7  Q=40;         //discharge
 8  f=1;          //silt factor
 9
10  //Laecey's theory
11  V=(Q*f/140)^(1/6);
12  A=Q/V;
13  P=4.75*Q^0.5;
14  D=(P-(P^2-6.944*A)^0.5)/3.472;
15  B=P-2.236*D;
16
17  R=5*V^2/(2*f);
```

```
18  S=f^(5/3)/(3340*Q^(1/6));
19  B=round(B);
20  D=round(D*100)/100;
21  mprintf("\n\nBy Laecey theory:");
22  mprintf("\nBed slope=%f.",S);
23  mprintf("\nWidth of channel section=%f m.",B);
24  mprintf("\nDepth of channel section=%f m.",D);
25
26  //Kennedy's theory
27  r=B/D;
28  m=1;            //critical velocity ratio
29  N=0.0225;     //rogosity coefficient
30  //using equation of area of trapezoidal section;Vo
        =0.55mD^0.64 and Q=A*Vo
31
32  D=(Q/8.058)^(1/2.64);
33  B=r*D;
34  B=round(B);
35  D=round(D*100)/100;
36  mprintf("\n\nBy Kennedy theory:");
37  mprintf("\nWidth of channel section=%f m.",B);
38  mprintf("\nDepth of channel section=%f m.",D);
```

Scilab code Exa 14.33 EX14 33

```
1
2
3  //example 14.33
4  //design Laecey regime channel
5  clc;funcprot(0);
6  //given
7  A=100000;              //culturable area(hectare)
8  IR=0.4;                //intensity of irrigation in
        kharif season
9  IK=0.3;                //intensity of irrigation in
```

```
        rabi  season
10  OR =1800;                  // outlet  discharge  factor  in
        kharif  season
11  OK =800;                   // outlet  discharge  factor  in
        kharif  season
12  l =0.1;                    // conveyance  loss
13  md =0.328;                 // average  diameter  of  material
14
15  AR = A * IR ;              // area  under  rabi
16  AK = A * IK ;              // area  under  kharif
17  Qr = AR / OR ;
18  Qk = AK / OK ;
19  Q =1.1* Qk ;
20  f =1.76*( md ) ^0.5;
21  V =( Q * f ^2/144) ^(1/6) ;
22  A = Q / V ;
23  P =4.75*( Q ) ^0.5;
24  D =( P -( P ^2 -6.944* A ) ^0.5) /3.472;
25  B = P -2.236* D ;
26  S = f ^(5/3) /(3340* Q ^(1/6) ) ;
27  B = round ( B *10) /10;
28  D = round ( D *100) /100;
29  mprintf ( " \ nBed  slope =% f . " , S ) ;
30  mprintf ( " \ nWidth  of  channel  section =% f  m . " , B ) ;
31  mprintf ( " \ nDepth  of  channel  section =% f  m . " , D ) ;
```

Scilab code Exa 14.34 EX14 34

```
1
2
3  // example  14.34
4  // calculate  concentration  at  point  10  cm  above  the
        bed
5  clc ; funcprot (0) ;
6  // given
```

281

```
 7  D=2.8;                  //depth of flow
 8  c_=700;                 //concentration at 30 cm below water
        surface
 9  y=0.1;
10  a=D-0.3;
11  e=0.4;                  //exponent in rouse equation;
12
13  c=c_*(a*(D-y)/(y*(D-a)))^e;
14  mprintf("concentration at point 10 cm above the bed=
        %i ppm.",c);
```

Scilab code Exa 14.35 EX14 35

```
 1
 2
 3  //example 14.35
 4  //design the distributory using Laecey theory
 5  clc;funcprot(0);
 6  //given
 7  f=0.85;                 //silt factor
 8  AR=3600;                //area for rabi
 9  AK=1400;                //area for kharif
10  delta_r=0.135;          //kor depth for rabi
11  delta_k=0.19;           //kor depth for kharif
12  tr=4;                   //kor period for rabi
13  tk=2.5;                 //kor period for kharif
14  Du_r=8.64*tr*7/delta_r;     //duty for rabi
15  Du_k=8.64*tk*7/delta_k;     //duty for kharif
16  q_r=AR/Du_r;                //discharge for rabi
17  q_k=AK/Du_k;                //discharge for kharif
18  Q=q_r;                      //since q_r>q_k
19  V=(Q*f^2/144)^(1/6);
20  A=Q/V;
21  P=4.75*(Q)^0.5;
22  D=(P-(P^2-6.944*A)^0.5)/3.472;
```

```
23  S=f^(5/3)/(3340*Q^(1/6));
24  P=round(P*100)/100;
25  D=round(D*100)/100;
26  mprintf("\nBed slope=%f.",S);
27  mprintf("\nPerimeter of channel section=%f m.",P);
28  mprintf("\nDepth of channel section=%f m.",D);
```

Chapter 15

IRRIGATION CHANNEL 2 DESIGN PROCEDURE

Scilab code Exa 15.1 EX15 1

```
 1
 2
 3  //example  15.1
 4  //design  a  channel  by  Kennedy  theory  using  Garret's
        diagram
 5  clc;funcprot(0);
 6  //given
 7  Q=7;                    //full  supply  discharge
 8  N=0.0225;               //rogosity  coefficient
 9  S=1/4444;               //bed  slope
10  m=1;                    //critical  velocity  ratio
11  s=1/2;                  //side  slope
12
13  //Values  of  B  and  D  are  obtained  by  Garret's  diagram
        fig.  15.3(b)  and  tabulated  as  below
14  B=[6 7 6.75];           //width  of  bed  from  Garret
        diagram
15  D=[1.5 1.35 1.38];      //depth  of  bed  from  Garret
        diagram
```

```scilab
16  Vo=[0.72 0.673 0.685];    //from Garret diagram
17
18  mprintf("Bed width          Depth              Ratio of
        V/Vo:              Remarks");
19  for i=1:3
20      A(i)=B(i)*D(i)+D(i)^2/2;          //Area
21      V(i)=Q/A(i);                      //Velocity
22      r(i)=V(i)/Vo(i);                      //ratio V/
            Vo
23      r(i)=round(r(i)*1000)/1000;
24      if i==1 then
25      s='small';
26  else
27      if (i==2) then
28      s='more';
29
30  else
31      s='satisfactory';
32  end
33  end
34      mprintf("\n%f              %f            %f
                                %s",B(i),D(i),r(i),s);
35  end
36  mprintf("\nHence, B=%f m; D=%f m.",B(3),D(3));
```

Scilab code Exa 15.2 EX15 2

```scilab
1
2
3  //example 15.2
4  //design an irrigation channel in alluvial soil by
        Laecy's theory
5  clc;funcprot(0);
6  //given
7  Q=15;                    //Full supply discharge
```

```
 8  f=1;                     //silt  factor
 9  s=1/2;                   //side  slope  of  channel
10
11  //from Laecey  regime  channel  (Fig.15.4(b))  B and  D
        is  obtained  as;
12  B=15.1;
13  D=1.38;
14  //also  from  Fig.15.5  we  get  slope  as
15  S=0.19/1000;
16  mprintf("Width  of  channel  section=%f m.",B);
17  mprintf("\nDepth  of  channel  section=%f m.",D);
18  mprintf("\nBed  slope=%f.",S);
```

Scilab code Exa 15.3 EX15 3

```
 1
 2
 3  //example  15.3
 4  //design  and  prepare  the  longitudnal  section;
        schedule  of  area  statistics  and  channel  dimension
         of  irrigation  channel
 5  clc;funcprot(0);
 6  //given
 7  dl=157.7;                    //datum  level
 8  fsl=157;                     //full  supply  level  of  parent
        channel
 9  bl=156;                  //bed  level  of  parent  channel
10  kor_r=4;                 //kor  period  of  rabi
11  kor_k=2.5;               //kor  period  of  kharif
12  kord_r=13.4;             //kor  depth  of  rabi
13  kord_k=19;               //kor  depth  of  kharif
14  s=0.5;                   //side  slope
15  m=1;                     //critical  velocity  ratio
16  N=0.0225;                //Kutter  n
17  qo_r=8.64*7*kor_r*100/kord_r;   //outlet  discharge
```

```scilab
        for rabi(calculation is wrong in book)
18  qo_k=8.64*7*kor_k*100/kord_k;   //outlet discharge
        for kharif(calculation is wrong in  book)
19  ca=16000;                      //culturable commanded area
20  Ir=0.3;                        //intensity of irrigation in
        rabi
21  Ik=0.125;                      //intensity of irrigation in
        rabi
22  Ar=Ir*ca;                      //area under rabi
23  Ak=ca*Ik;                      //area under kharif
24  q_r=Ar/qo_r;
25  q_k=Ak/qo_k;
26  q_r=round(q_r*100)/100;
27  q_k=round(q_k*100)/100;
28  mprintf("discharge neede for rabi crop=%f cumecs.",
        q_r);
29  mprintf("\ndischarge neede for kharif crop=%f cumecs
        .",q_k);
30  mprintf("\noutlet discharge factor adopted=%i
        hectares per cumecs.",qo_r);
31  //at km 5
32  ca=8000;           //culturable area
33  Ar=Ir*ca;              //area under rabi
34  q_r=Ar/qo_r;
35  l=0.5               //total loss after 5 km
36  q=q_r+l;             //total discharge
37  dq=1.1*q;            //desigm discharge
38  S=1/4000;           //slope
39  B=[5.5 4.9 4.55];   //Bed width
40  D=[0.73 0.79 0.84]; //water depth
41  Vo=[0.448 0.472 0.488];   //critical velocity
42  mprintf("\n\nBed width     water depth      area
            velocity      critical velocity    C.V.R")
        ;
43  for i=1:3
44      A(i)=B(i)*D(i)+D(i)^2/2;
45      V(i)=dq/A(i);
46      m(i)=V(i)/Vo(i);
```

```
47      A(i)=round(A(i)*100)/100;
48      V(i)=round(V(i)*1000)/1000;
49      m(i)=round(m(i)*100)/100;
50      mprintf("\n%f        %f        %f        %f        %f
                    %f",B(i),D(i),A(i),V(i),Vo(i),m(i))
          ;
51  end
52  B=4.55;D=0.84;
53  mprintf("\nhence take B=%f .; D=%f m.",B,D);
54  //at km 4
55  q=round(q*100)/100;
56  mprintf("\ndischarge at 5 km=%f cumecs.",q);
57  ca=10000;          //culturable area
58  Ar=Ir*ca;               //area under rabi
59  q_r=Ar/qo_r;
60  l=0.5              //total loss below 5 km
61  P=B+D*5^0.5;          //wetted perimeter
62  l1=P*1000*2/1000000;  //loss between 5 km and 4km
63  l2=l1+l;
64  q=q_r+l2;
65  dq=1.1*q;
66  q=round(q*1000)/1000;
67  mprintf("\ndischarge at 4 km =%f cumecs",q);
68  mprintf("\nother discharge are calculated and are
        tabulated as:");
69  x=[0:1:5];
70  A1=[4800 4200 3600 3300 3000 2400];
71  A2=[2000 1750 1500 1375 1250 1000];
72  S=[22.5 22.5 22.5 24 24 25];
73  B=[5.5 5.2 4.85 4.7 4.55 4.55];
74  D=[1.04 1.007 0.975 0.945 0.915 0.840];
75  dq=[3.56 3.17 2.8 2.6 2.4 2.02];
76  V=[0.570 0.555 0.538 0.530 0.521 0.484];
77  m=[1.015 1 1 1 1 0.992];
78  mprintf("\n\nBelow km        area to irrigate rabi
            area to irrigate kharif      bed slope
            bed width            water depth
        design discharge        velocity        C.V.R");
```

```
79  for i =1:6
80      mprintf("\n%i                      %i
                                    %i
          %f                %f                %f                %f
                      %f            %f",x(i),A1(i),A2(i),S(i)
          ,B(i),D(i),dq(i),V(i),m(i));
81  end
```

Scilab code Exa 15.4 EX15 4

```
1
2
3  //example 15.4
4  //calculate the economical depth of cutting for
       cross section of channel
5  clc;funcprot(0);
6  //given
7  B=5;                      //bed width
8  t=2;                      //top width of banks
9  h=2.92;                   //heigth of banks from bed
10  n=1.5;
11
12  //sectional area of digging=sectional area of two
        banks
13  //By+zy^2=2(h-y)+2n(h-y)^2
14  //substituting the values and on simplificatio we
        get
15  s=poly([18.59,-13.26,1],'x','c');
16  y=roots(s);
17  //from this we get y=11.666556 and 1.5934436.
18  //taking
19  y=1.5934436;
20  y=round(y*10)/10;
21  mprintf("economical depth of cutting=%f m.",y);
```

Chapter 16

WATERLOGGING AND CANAL LINING

Scilab code Exa 16.1 EX16 1

```
1
2
3  //example 16.1
4  //design a trapezoidal concrete lined channel
5  clc;funcprot(0);
6  //given
7  Q=100;                    //discharge
8  S=25/100000;                //bed slope
9  N=0.016;                  //rogsity coefficient
10 s=1.5;                    //side slope
11 V=1.5;                    //limiting velocity
12
13 //using manning's equation   V=(R^2/3*S^1/2)/N;
14 R=(V*N/(S^0.5))^(1.5);        //hydraulic mean depth
15
16 //for s=1.5;
17 theta=acot(1.5);
18 A=Q/V;
19 P=A/R;
```

```
20  //using equation of area and perimeter of trapezium
21  //perimeter of trapezium=b+2d(theta+cot(theta));
22  //area of trapezium=bd+d^2(theta+cot(theta));
23  //we get
24  y=poly([31.9,-17.1,1],'x','c');
25  d=roots(y);
26  //we get D=14.968917 and 2.1310826.
27  //taking
28  d=2.1310826;
29  b=P-4.18*d;
30  b=round(b*100)/100;
31  d=round(d*100)/100;
32  mprintf("required bed width=%f m.",b);
33  mprintf("\nrequired bed depth=%f m",d);
```

Scilab code Exa 16.2 EX16 2

```
1
2
3  //example 16.2
4  //design a trapezoidal concrete lined channel
5  clc;funcprot(0);
6  //given
7  Q=100;                      //discharge
8  S=25/100000;                //bed slope
9  N=0.016;                    //rogsity coefficient
10 s=1.5;                      //side slope
11 r=8;                        //b/d ratio
12
13 //using manning equation V=(R^2/3*S^1/2)/N;
14 //Perimeter=A/R
15 //V=Q/A and on simplification we get
16 d=((101/10.09)*(12.18/10.09)^(2/3))^(3/8);
17 b=r*d;
18 b=round(b);
```

```
19  d=round(d*100)/100;
20  mprintf("required bed width=%f m.",b);
21  mprintf("\nrequired bed depth=%f m",d);
```

Scilab code Exa 16.3 EX16 3

```
1
2
3  //example 16.3
4  //design a concrete lined channel
5  clc;funcprot(0);
6  //given
7  Q=45;                  //discharge
8  S=1/10000;             //bed slope
9  s=5/4;                 //side slope
10 N=0.018;               //rogosity coefficient(manning N)
11
12 //channel is assumed to be of triangular section
13 theta=acot(s);
14 //using manning equation V=(R^2/3*S^1/2)/N;
15 //V=Q/A;
16 //perimeter of trapezium=b+2d(theta+cot(theta));
17 //area of trapezium=bd+d^2(theta+cot(theta));
18 //we get
19 d=(Q*2.86/1.925)^(3/8);
20 d=round(d*100)/100;
21 mprintf("\nrequired depth of triangular channel=%f m
        ",d);
```

Scilab code Exa 16.4 EX16 4

```
1
2
```

```
3  //example 16.4
4  //design a concrete lined channel of trapezoidal
       section
5  clc;funcprot(0);
6  //given
7  Q=250;                //discharge
8  S=1/6000;             //bed slope
9  s=1.5;                //side slope
10 d=3;                  //limiting depth
11 N=0.015;              //rogosity coefficient
12
13 //using Perimeter=A/R;
14 //perimeter of trapezium=b+2d(theta+cot(theta));
15 //area of trapezium=bd+d^2(theta+cot(theta));
16 //Q=A*V; and on simplification
17 //we get
18 //(3b+18.81)^5/3/(b+12.54)^2/3=290.47;
19 //solving it by trial and error method we get
20 b=44.6;
21 mprintf("required bed width=%f m.",b);
22 mprintf("\nrequired bed depth=%i m",d);
```

Scilab code Exa 16.5 EX16 5

```
1
2
3  //example 16.5
4  //calculate spacing of drains
5  clc;funcprot(0);
6  //given
7  H=10;                  //depth of impervious stratum from
       top soil
8  D=1.8;                 //position of drain below top soil
     surface
9  Hw=1.5;                //depth of highest point of water
```

```
10  k=1D-4;                // permeability  constant
11
12  // since  water  has  to  be  removed  in  24  hours
13  b=H-Hw;
14  a=H-D;
15  L=(4*k*(b^2-a^2)*100*24*3600/0.8)^0.5;
16  mprintf(" drains  should  be  spaced  at  %i m c/c.",L);
```

Scilab code Exa 16.6 EX16 6

```
 1
 2
 3  // example  16.6
 4  // calculate  permeability  coefficient
 5  clc;funcprot(0);
 6  // given
 7  L=30;              // spacing  between  drans
 8  Q=4D-6;            // discharge
 9  a=8;
10  b=8.3;
11
12  k=1000000*Q*L/(4*(b^2-a^2));
13  k=round(k*100)/100;
14  mprintf(" permeability  coefficient=%fD-6 m/sec.",k);
```

Scilab code Exa 16.7 EX16 7

```
 1
 2
 3  // example  16.7
 4  // calculate  annual  average  rainfall
 5  clc;funcprot(0);
 6  // given
```

```
7  L=50;              //spacing between drains
8  k=1D-5;            //permeability coefficient
9  a=10;
10 b=10.3;
11
12 Q=4*k*(b^2-a^2)/L;
13 Pav=Q*24*3600*100*100/L;
14 mprintf("annual average rainfall=%i cm",Pav);
```

Scilab code Exa 16.8 EX16 8

```
1
2
3  //example16.8
4  //calculate ratio of discharge at A and B;ratio of
       average rainfall at A and B
5  clc;funcprot(0);
6  //given
7  r1=2;          //ka/kb
8  r2=1/1.5;      //La/Lb
9  r3=5/6;        //(b^2-a^2)a/((b^2-a^2)b)
10
11 Rq=r1*r3/r2;
12 Rp=Rq/r2;
13 mprintf("ratio of discharge at A and B=%f.",Rq);
14 mprintf("\nratio of average rainfall at A and B=%f."
       ,Rp);
```

Scilab code Exa 16.9 EX16 9

```
1
2
3  //example 16.9
```

295

```
 4  // decide  whether  it  is  economically  feasible  to
        provide  canal  lining
 5  clc ; funcprot (0);
 6  // given
 7  li =2.5;                        // seepage  loss  for  lined
        channel
 8  p1 =25;                         // wetted  perimeter  for  lined
        channel
 9  t =12;                          // thickness  of  concrete  lining
10  lf =0.02;                       // seepage  loss  for  unlined
        channel
11  p2 =20;                         // wetted  perimeter  for  unlined
         channel
12
13  // assume 1 km length  of  canal
14  // annual  benifit
15
16  //(1). seepage
17  A1=p1 *1000;                    // area  of  wetted  perimeter
18  li=li*p1 /1000;                 // seepage  loss
19  A2=p2 *1000;                    // area  of  wetted  perimmeter
        for  unlined  channel
20  lf=p2*lf /1000;                 // seepage  loss  for  unlined
        channel
21  s=li-lf ;                       // saving  in  water  loss
22  a1=s*p1 *100000;                // annual  revenue  saved
23
24  //(2) maintainence
25  a2 =0.4*25000;                  // saving  in  maintainance  cost
26  ts=a1+a2 ;                      // total  annual  benifit
27
28  // annual  cost
29  A1=p2 *1000;                    // area  of  lining  for  unlinrd
        canal
30  C=100* A1;                      // cost  of  lining
31  // interest  rate  is  6%
32  i =0.06;
33  N=50;
```

```
34  a=(C*i*(i+1)^N)/((1+i)^N-1);   //annual cost of
        lining  or  capital  recovery  factor
35  bcr=ts/a;                      //benifit  cost  ratio
36  bcr=round(bcr*1000)/1000;
37  mprintf("\nBenifit  cost  ratio=%f.",bcr);
38  //as  bcr>1
39  mprintf("  ;Since  it  is  more  than  1.\nHence,  it  is
        economically  justifiable.  ");
```

Scilab code Exa 16.10 EX16 10

```
1
2
3  //example  16.10
4  //calculate  the  required  depth  of  water  to  be
        applied
5  clc;funcprot(0);
6  //given
7  Ecd=20;               //electrical  conductivity  of
        drainage  water
8  Eci=1.5;          //m mho/cm
9  Dc=55.5;          //consumptive  use
10
11  Lr=Eci/Ecd;
12  D=Dc/(1-Lr);
13  mprintf("required  depth  of  water  to  be  applied=%i mm
        .",D);
```

Scilab code Exa 16.11 EX16 11

```
1
2
3  //example  16.11
```

297

```
 4 //calculate the required depth of water to be
      applied
 5 clc;funcprot(0);
 6 //given
 7 Eci=1.4;           // m mho/cm
 8 Ece=11;            //saturated extract of soil
 9 Dc=85;             //consumptive use requirement of
      crop
10
11 //let us assume Ecd=2Ece
12 Lr=Eci/(2*Ece);
13 Di=Dc/(1-Lr);
14 Di=round(Di*10)/10;
15 mprintf("required depth of water to be applied=%f mm
      .",Di);
```

Scilab code Exa 16.12 EX16 12

```
 1
 2
 3 //example 16.12
 4 //calculate average boundary shear stress;
 5 //percentage of earth work is saved in lined section
 6 clc;funcprot(0);
 7 //given
 8 s=1.5;          //side slope
 9 Q=15;           //discharge
10 S=1/4000;       //bed slope
11 Nl=0.014;        //manning n for lined channel
12 Nu=0.028;        //manning n for ulined channel
13 fb=0.75;          //free board
14
15 //considering the perimeter of trapezoidal section
16 //taking minimum perimeter for given area
17 //i.e dP/dD=0
```

```
18  //we get
19  //A=2.1D^2; R=D/2; and P=4.2D
20
21  //for linrd channel
22  //Q=AR^(2/3)*S^0.5
23  //substituting above values we get
24  D=(10.0396)^(3/8);
25  B=0.6*D;
26  R=D/2;
27  tau=9.81*R*S*1000;
28  tau=round(tau*1000)/1000;
29  mprintf("for lined canal:");
30  mprintf("\naverage boundary shear stress=%f N/square
        m.",tau);
31  Dc=D+fb;                //total depth of cutting
32  A1=(B+1.5*Dc)*Dc;
33
34  //for unlined channel
35  //Q=AR^(2/3)*S^0.5
36  //substituting above values we get
37  D=3.08;
38  B=0.6*D;
39  R=D/2;
40  tau=9.81*R*S*1000;
41  tau=round(tau*100)/100;
42  mprintf("\n\nfor unlined canal:");
43  mprintf("\naverage boundary shear stress=%f N/square
        m.",tau);
44  Dc=D+fb;                //total depth of cutting
45  A2=(B+1.5*Dc)*Dc;
46  per=(A2-A1)*100/A2;
47  per=round(per*100)/100;
48  mprintf("\n\npercent saving of earth=%f percent.",
      per);
```

Scilab code Exa 16.13 EX16 13

```
 1
 2
 3  //example 16.13
 4  //design a lined canal
 5  clc;funcprot(0);
 6  //given
 7  Q=100;                   //discharge
 8  S=1/2500;                //bed slope
 9  V=2;                     //maximum permissible velocity
10  n=0.013;                 //manning n
11  s=1.25;                  //side slope
12
13  A=Q/V;
14  //from manning formula   V=(R^2/3*S^1/2)/N;
15  R=(V*n/S^0.5)^1.5;
16  P=A/R;
17
18  //now using the equation of area and perimeter of
        trapezoid
19  //area=D(B+2.5D)
20  //perimeter=B+3.2D;
21  //we get
22  y=poly([50,-33.73,1.95],'x','c');
23  D=roots(y);
24  //we get D=15.660087 and 1.6373489
25  //taking
26  D=1.6373489;
27  B=P-3.2*D;
28  B=round(B*10)/10;
29  D=round(D*100)/100;
30  mprintf("required bed width=%f m.",B);
31  mprintf("\nrequired bed depth=%f m",D);
```

Scilab code Exa 16.14 EX16 14

```
1
2
3  //example 16.14
4  //calculate to what extent discharge can be
       increased without changing bed slope
5  clc;funcprot(0);
6  //given
7  B=5;            //bed width
8  D=2;            //bed depth
9  S=1/1600;       //bed slope
10 n=0.015;        //manning n
11
12 A=B+2*D;        //area of lining
13 //let B1 and D1 be new  width and depth of bed
14 //for getting maximum discharge we diffrentiate Q
       and equating it to zero
15 //Q=S^0.5*B1D1^5/3/n
16 //we get
17 D1=45/16;
18 B1=9-2*D1;
19 Q1=S^0.5*B1*D1^5/3/n;
20 D1=round(D1*10000)/10000;
21 mprintf("new width of bed=%f m.",B1);
22 mprintf("\nnew depth of bed=%f m.",D1);
23 mprintf("\n maximum discharge=%f cumec.",Q1);
24 R=D;
25 V=R^(2/3)*S^0.5/n;
26 F=V/(9.81*D)^0.5;         //froud number
27 R=D1;
28 V=R^(2/3)*S^0.5/n;
29 F=V/(9.81*D1)^0.5;        //froud number
30 mprintf("\nFroud number is less than 1 in both case
       .\nHence,flow doesnot change from sub-critical to
        super critical.");
```

Scilab code Exa 16.15 EX16 15

```
1
2
3  //example 16.15
4  //calculate maximum carrying capacity of canal
5  //area to be irrigated
6  clc;funcprot(0);
7  //given
8  B=5;            //bed width
9  D=2.5;          //bed depth
10 s=1.5;          //side slope
11 S=1/1000;       //bed slope
12 n=0.016;         //manning n
13 k=10;           //kor period
14 d=150;          //field irrigation requirement
15
16 theta=acot(s);
17 A=B*D+D^2*(theta+1/tan(theta));
18 P=B+2*D*(theta+1/tan(theta));
19 R=A/P;
20 Q=A*R^(2/3)*S^0.5/n;
21 V=Q*k*24*3600;    //volum of water supply by channel
22 A=V*10/(d*10000);
23 Q=round(Q*100)/100;
24 A=round(A)*100;
25 mprintf("maximum carrying capacity of canal=%f cumec
       .",Q);
26 mprintf("\nArea to be irrigated=%f hectares.",A);
```

Chapter 17

CANAL OUTLETS

Scilab code Exa 17.1 EX17 1

```
1
2
3  //example 17.1
4  //calculate discharge through the outlet
5  clc;funcprot(0);
6  //given
7  D=100.0;          //F.S.L of distributory
8  wc=99.90;         //F.S.L of water course
9  L=9;              //length of pipe
10 d=20;             //diameter of pipe
11 f=0.005;          //coefficient of friction
12 g=9.81;           //acceleration due to gravity
13
14 H=D-wc;           //working head
15 C=(d/((1.5*d/(400*f)+L)*f))^0.5/20;
16 A=%pi*d^2/(4*10000);
17 q=C*A*(2*g*H)^0.5;
18 q=round(q*10000)/10000;
19 mprintf("discharge through the outlet=%f cumec.",q);
```

Scilab code Exa 17.2 EX17 2

```
1
2
3  //example 17.2
4  //design a submerged pipe
5  clc;funcprot(0);
6  //given
7  q=0.04;           //discharge through outlet
8  D=100.0;            //F.S.L of distributing canal
9  wc=99.90;         //F.S.L of water course
10 dep=1.1;          //full supply depth distributing
       canal
11 C=0.7;             //average value of coefficient of
       discharge
12 g=9.81;          //acceleration due to gravity
13
14 H=D-wc;              //available head
15 A=q/(C*(2*g*H)^0.5);
16 d=(4*A/%pi)^0.5*100;
17 d=round(d*10)/10;
18 mprintf("diameter of pipe required=%f cm.",d);
19 mprintf("\nuse pipe of diameter 25 cm.");
```

Scilab code Exa 17.3 EX17 3

```
1
2  //example 17.3
3  //design submerged pipe
4  clc;funcprot(0);
5  //given
6  q=0.04;             //discharge through outlet
```

304

```
 7  D=100.0;              //F.S.L  of  distributing  canal
 8  wc=99.90;         //F.S.L  of  water  course
 9  dep=1.1;           //full  supply  depth  distributing
        canal
10  f=0.01;              //coefficient  of  friction
11  g=9.81;         //acceleration  due  to  gravity
12  L=9;             //Length  of  pipe
13
14  H=D-wc;            //working  head
15  //first  trial
16  //taking  d=22.8  cm
17  d=22.8;
18  C=(d/((1.5*d/(400*f)+L)*f))^0.5/20;
19  A=q/(C*(2*g*H)^0.5);
20  d=(4*A/%pi)^0.5*100;
21  //second  trial
22  C=(d/((1.5*d/(400*f)+L)*f))^0.5/20;
23  A=q/(C*(2*g*H)^0.5);
24  d=(4*A/%pi)^0.5*100;
25  d=round(d*100)/100;
26  mprintf("diameter  of  pipe  required=%f  cm.",d);
27  mprintf("\nprovide  diameter  of  pipe  as  25  cm.");
```

Scilab code Exa 17.4 EX17 4

```
 1
 2
 3  //example  17.4
 4  //design  an  open  flume  outlet
 5  clc;funcprot(0);
 6  //given
 7  Q=0.06;              //discharge
 8  D=0.85;              //full  supply  depth
 9  Hw=15;                //available  working  head
10  Bt=7;C=1.6;         //let  us  choose
```

```
11  H=(Q*100/(C*Bt))^(2/3);
12  mh=0.2*H;              //minimum modular head
13  mh=round(mh*1000)/1000;
14  mprintf("minimum modular head=%f m. < available
          working head.\nhemce,design is safe.",mh);
15  o=H/D;
16  o=round(o*1000)/1000;
17  mprintf("\nsetting of outlet=%f. <0.9.\nhence,outlet
          will work as hyper propotional outlet.",o);
```

Chapter 18

CANAL REGULATION WORKS

Scilab code Exa 18.1 EX18 1

```
1
2
3  //example  18.1
4  //design  Sarda  type  fall
5  clc ; funcprot (0) ;
6  // given
7  Q =40;                    // full  supply  discharge
8  sl_u =218.3;              // supply  level  at  upstream
9  sl_d =216.8;              // supply  level  at  downstream
10 D =1.8;                   // suplly  depth
11 L =26;                    // bed  width
12 bl_u =216.5;              // bed  level  upstream
13 bl_d =215;                // bed  level  downstream
14 drop =1.5;
15
16 //from  the  eqauation ;  Q=1.99LH^1.5*(H/B) ^ (1/6) ;
17 //B=0.55*(H+d) ^0.5;
18 //H+d=drop+D;
19 //we  get
```

```
20  H=(0.774)^0.6;
21  d=3.3-H;
22  Hc=D-H;
23  d=round(d*100)/100;
24  H=round(H*100)/100;
25  Hc=round(Hc*100)/100;
26  mprintf("H=%f m.\nd=%f m.",H,d);
27  mprintf("\ncrest height above bed=%f m.",Hc);
28
29  //adopt trapezoidal crest
30  B=1;                        //top width
31  mprintf("\n\nD/S batter=1:3; U/S batter=1:8.");
32  Va=Q/((27+D)*D);
33  vh=Va^2/(2*9.81);
34  tel_up=sl_u+vh;
35  crest=sl_u-H;
36  E=sl_u-crest;
37  mprintf("\nR.L of crest=%f m.",crest);
38  mprintf('\nE=%f m.',E);
39  //design of cistern
40  x=(E*drop)^(2/3)/4;                   //depth of cistern
41  lc=5*(E*drop)^0.5;                    //length of
        cistern
42  cb=bl_d-x;
43  x=round(x*100)/100;
44  cb=round(cb*1000)/1000;
45  lc=round(lc*10)/10;
46  mprintf("\n\ndepth of cistern=%f m.",x);
47  mprintf("\nlength of cistern=%f m.",lc);
48  mprintf("\nR.L of bed of cistern=%f m.",cb);
49  mprintf("\nkeep cistern at R.L 214.69.");
50  //design of impervious floor
51  Hs=2.44;            //seepage head
52  c=8;               //Bligh's coefficient
53  li=Hs*c;
54  d1=1;d2=1.6;
55  vl=2*(d1+d2);
56  lh=li-vl;
```

```
57  mprintf("\n\ndesign of impervious floor:");
58  mprintf("\nprovide upstream cut−off=%i m.;
        downstream cut−off=%f m.",d1,d2);
59  mprintf("\nlength of horizontal impervious floor=%f
        m.",lh);
60  mprintf("\nprovide 15 m length impervious floor.");
61  ld=2*(D+1.2)+drop;
62  mprintf("\nminimum length of impervious floor to the
         d/s of toe of crest wall=%f m.",ld);
63  mprintf("\nprovide ld=8 m.");
64  bl=15-8;
65  mprintf("\nthe balance of the length %i m is to be
        provided under and u/s of the crest.",bl);
66
67  tcl=15+2*(1+16);
68  mprintf("\n\nuplift pressure is counter balanced by
        weigth of water.\n hence provide thickness of 0.4
        m.");
69  rho=2.24;
70  static=2.44*(1-0.446)+x;
71  t=static/(rho-1);
72  t=round(t*100)/100;
73  mprintf("\nfor other points; thickness required =%f
        m.",t);
74  mprintf("\nprovide thickness of 1.40 m.");
75  mprintf("\nat downstream end of floor provide
        thickness of 0.6 m overlaid by 0.2 m brick
        pitching.");
76
77  n=d2/(Hs*5);                    //n=1/%pi*(lambda)^0.5
78  //from khosla exit curve we get
79  alpha=10.5;
80  lambda=(1/(%pi*n))^2;
81  alpha=((2*lambda-1)^2-1)^0.5;
82  b=alpha*d2;
83  b=round(b*100)/100;
84  mprintf("\n\nchecking of floor thickness by khosla
        theory:");
```

```
85  mprintf("\nlength of floor provided=%f m. > length
        by Bligh theory.",b);
86  b=15;
87  d2=1.8;
88  alpha=b/d2;
89  n=0.145;
90  Ge=Hs*n/d2;
91  Ge=round(Ge*10)/10;
92  mprintf("\nexit gradient after increase in depth cut
        -off=%f. which is in permissible limit",Ge);
93  mprintf('\nprovide depth cut-off to 1.8 m.');
94  //calculation of pressure
95  mprintf("\n\ncalculation of pressure:");
96  mprintf("\nU/S cut-off:");
97  d1=1;
98  b=15;
99  alpha_=d1/b;
100 fic1=100-24;
101 fid1=100-17;
102 t=0.4;
103 fic1=fic1+(fid1-fic1)*t/d1;
104 mprintf("\ncorrected fic1=%f percent.",fic1);
105 mprintf("\nD/S cut-off wall:");
106 d2=1.8;
107 b=15;
108 alpha_=d1/b;
109 fie2=31;
110 fid2=21.5;
111 t=0.6;
112 fie2=fie2-(fie2-fid2)*t/1.8;
113 fie2=round(fie2*10)/10;
114 mprintf("\ncorrecte fie2=%f percent.",fie2);
115 //calculation of thickness
116 mprintf("\n\nprovide a minimum thickness of 0.4 m
        for u/s floor.");
117 pre=fie2+(fic1-fie2)*8/b;
118 static=pre*Hs/100+x;
119 t=static/(rho-1);
```

```
120  t=round(t*100)/100;
121  mprintf("\nthickness  at  d/s  toe  of  crest=%f m.",t);
122  mprintf("\nprovide  thickness  of  1.4  m  thick  concrete
         overlaid  by  0.2  m  brick  pitching.");
123  pre=fie2+(fic1-fie2)*5/b;
124  static=pre*Hs/100+x;
125  t=static/(rho-1);
126  t=round(t*100)/100;
127  mprintf("\nthickness  at  3  m  from  d/s  toe  of  crest=%f
         m.",t);
128  mprintf("\nprovide  thickness  of  1.2  m  thick  concrete
         overlaid  by  0.2  m  brick  pitching.");
129  pre=fie2+(fic1-fie2)*2/b;
130  static=pre*Hs/100;                    //calculation  is
         wrong  in  book
131  t=static/(rho-1);
132  t=round(t*100)/100;
133  mprintf("\nthickness  at  6m  from  d/s  toe  of  crest=%f
         m.",t);
134  mprintf("\nprovide  thickness  of  0.7  m  thick  concrete
         overlaid  by  0.2  m  brick  pitching.");
135  //design  of  downstream  wings
136  wing=6*(E*drop)^0.5;
137  hw=D+0.5;
138  mprintf("\n\nheigth  of  top  of  downstream  wings  above
         the  bed=%f m.",hw);
139  projec=hw*3;
140  mprintf("\nlength  of  warped  wing  measured  along
         centre  line  of  canal=%f m.",projec);
141  //downstream  pitching
142  l=9+2*1.5;
143  mprintf("\n\nlength  of  bed  pitching=%f m.",l);
144  mprintf("\nlength  of  sloping  pitching=7 m.\nlength
         of  horizontal  pitching=6 m.");
145  mprintf("\nprovide  one  toe  wall  of  1  m  depth  and  0.4
         m  width.");
146  mprintf("\nside  pitching  is  curtailed  at  45  degree
         from  the  end  of  bed  pitching  in  plan.\nsupprot
```

```
        the side pitching on toe wall 0.4 m thick and 1 m
         deep. ");
147 //energy dissipators
148 q=Q/L;
149 dc=(q^2/9.81)^(1/3);
150 mprintf("\n\nsize and position of friction blocks:")
        ;
151 L=2*dc;
152 w=dc;
153 h=dc;
154 di=1.5*dc;
155 L=round(L*10)/10;
156 w=round(w*10)/10;
157 h=round(h*10)/10;
158 di=round(di);
159 mprintf("\nlength of block=%f m.\nwidth of block=%f
       m.\nheight of block=%f m.\ndistance from toe of
       crest=%f m.",L,w,h,di);
160 mprintf("\nprovide two rows staggered ata distance
       of 1 m from toe of crest.");
161 mprintf("\nsize and position of cube blocks:");
162 L=D/10;
163 w=D/10;
164 h=w;
165 L=round(L*10)/10;
166 w=round(w*10)/10;
167 h=round(h*10)/10;
168 mprintf("\nlength of block=%f m.\nwidth of block=%f
       m.\nheight of block=%f m.",L,w,h);
169 mprintf("\nprovide two rows staggered at the end of
       impervious floor.");
170 //u/s approach
171 r=6*H;
172 mprintf("\n\nprovide wing wall segmental with 5 m
       radius subtending angle of 60 degree at the
       centre.");
```

312

Scilab code Exa 18.2 EX18 2

```
1
2
3  //example 18.2
4  //design an unflumed straight glacis non−meter fall
5  clc;funcprot(0);
6  //given
7  Q=40;                  //full supply discharge
8  sl_u=218.3;            //supply level at upstream
9  sl_d=216.8;            //supply level at downstream
10 D=1.8;                 //suplly depth
11 L=26;                  //bed width
12 bl_u=216.5;            //bed level upstream
13 bl_d=215;              //bed level downstream
14 drop=1.5;
15 Ge=1/6;                //permissible exit gradient
16
17 //design of crest
18 mprintf("design of crest:");
19 E=(Q/(1.84*L))^(2/3);
20 V=Q/((L+D)*D);
21 vh=V^2/(2*9.81);
22 tel_up=sl_u+vh;
23 cl=tel_up-E;
24 w=2*E/3;
25 w=round(w*10)/10;
26 mprintf("\nlength of crest=%f m.",L);
27 mprintf("\nwidth of crest=%f m.",w);
28 //design of cistern
29 q=Q/L;
30 Hl=1.5;
31 //from blench curve
32 Ef2=1.44;
```

313

```
33  cistern=sl_d+0.03-1.25*Ef2;
34  mprintf("\n\nR.L of cistern=%f m. > d/s bed level.",
        cistern);
35  mprintf("\nkeep R.L of cistern at 214.5 m.");
36  l=6*Ef2;
37  mprintf("\nlength of cistern=%f m.",l);
38  mprintf("\nprovide cistern of 9 m length ");
39  d=bl_d-214.5;
40  mprintf("\ndepth of cistern=%f m.",d);
41
42  //design of impervious floor
43  d1=D/3;
44  mprintf("\n\ndesign of impervious floor:");
45  mprintf("\nprovide 0.4 m wide and 1 m deep curtain
        wall at u/s.");
46  d2=D/2;
47  mprintf("\nprovide 0.4 m wide and 1 m deep curtain
        wall at d/s.\nthe curtain wall will project the
        above the d/s bed by 0.18 m.");
48  Hs=cl-bl_d;
49  d2=1;
50  n=d2*Ge/Hs;        //n=1/(%pi*(lambda)^0.5)
51  //from khosla exit curves we get
52  alpha=40;
53  lambda=(1/(%pi*n))^2;
54  alpha=((2*lambda-1)^2-1)^0.5;
55  b=alpha*d2;
56  //since length is to excessive
57  d2=2;
58  n=d2*Ge/Hs;        //n=1/(%pi*(lambda)^0.5)
59  //from khosla exit curves we get
60  alpha=10;
61  lambda=(1/(%pi*n))^2;
62  alpha=((2*lambda-1)^2-1)^0.5;
63  b=alpha*d2+1;
64  mprintf("\ntotal length=%i m.\nlength of cistern=9 m
        .\nlength of d/s glacis=5.88 m.\nwidth of crest
        =0.6 m.\nlength of u/s glacis=0.47 m.\nbalance to
```

```
      be provided to u/s of the u/s glacis=4.05 m.",b)
      ;
65
66 //pressure calculations
67 mprintf("\n\npressure calculations:");
68 mprintf("\nupstream curtain wall:");
69 d1=1;b=20;
70 alpha_=d1/b;
71 t=0.3;
72 fic1=100-22;
73 fid1=100-15;
74 corec=(fid1-fic1)*t/d1
75 fic1=fic1+corec;
76 mprintf("\ncorrected fi_c1=%f percent.",fic1);
77 mprintf("\ndownstream curtain wall:");
78 d2=2;b=20;
79 alpha_=d2/b;
80 t=0.5;
81 fie=29;
82 fid=21;
83 corec=(fie-fid)*t/d2
84 fie=fie-corec;
85 mprintf("\ncorrected fi_e=%f percent.",fie);
86 mprintf("\ntoe of glacis:");
87 //assuming linear variation of pressure
88 p=fie+(80-fie)*9/20;
89 mprintf("\npressure at downstream of the glacis=%f
      percent.",p);
90
91 //floor thickness
92 rho=2.24;
93 mprintf("\n\nfloor thickness:\nprovide minimum
      thickness of 0.3 m at the u/s floor.");
94 static=p*2.44/100+(bl_d-214.5);
95 t=static/(rho-1);
96 t=round(t*100)/100;
97 mprintf("\nfloor thickness required at toe of glacis
      =%f m.\nprovide 1.5 m thick floor for length of 3
```

```
        m." , t ) ;
 98  p=fie +(80- fie )*6/20;
 99  static=p*2.44/100+( bl_d -214.5) ;
100  t=static /( rho -1) ;
101  t=round ( t *100) /100;
102  mprintf (" \ nfloor  thickness  required  at  3m from toe
         of  glacis=%f m. \ nprovide  1.3 m thick  floor  from 3
          m to  6.5 m from toe  of  glacis ." ,t ) ;
103  t=0.27*2.44/( rho -1) ;
104  t=round ( t *100) /100;
105  mprintf (" \ nthickness  of  d/s end  of  cistern=%f m. \
         nprovide  thickness  of  0.6 m at  d/s end  of  floor ."
         ,t ) ;
106
107  // design  of  d/s protection
108  mprintf (" \ n \ nno bed  protection  is  needed  as
         deflector  wall  is  provided ." ) ;
109  sp=3*D;
110  mprintf (" \ nlength  of  side  protection=%f m. \ nprovide
         5.5 m length  of  20 cm thick  brick  pitching  beyond
          impervious  floor . \ npitching  will  rest  on toe
         wall  0.4 m wide  and  0.9 m deep . \ nprovide  0.4 m
         wide  profile  at  the  end  of  pitching " ,sp ) ;
111  // design  of  u/s approach
112  mprintf (" \ n \ nu/s wing  wall  is  splayed  at  45 degree
         from u/s end  of  impervious  floor . \ nextend  1 m
         into  earthen  banks  from line  of  F.S.L." ) ;
```

Scilab code Exa 18.3 EX18 3

```
1
2  // example  18.3
3  // design  a  cross −regulator  and  head  regulatorfor  a
        distributory  channel
4  clc ; funcprot (0) ;
```

```scilab
 5  // givrn
 6  Q=100;                  // discharge of parent channel
 7  Qd=15;                  // discharge ofdistributory
 8  fsl_u=218.1;            //F.S.L of upstream parent channel
 9  fsl_d=217.9;            //F.S.L of downstream of parent
        channel
10  bw_u=42;                // bed width of parent channel
        upstream
11  bw_d=38;                // bed width of parent channel
        downstream
12  hw=2.5;                 // depth of water in parent channel
13  fsl_dis=217.1;          //F.S.L of distributory
14  hw_dis=1.5;             // depth of water in distributory
15  Ge=1/5;                 // permissible exit gradient
16
17  // design of cross regulator
18  mprintf("DESIGN OF CROSS-REGULATOR::");
19  // design of crest and waterway
20  mprintf("\n\ndesign of crest and waterway:");
21  cl=fsl_u-hw;
22  h=fsl_u-fsl_d;
23  d=fsl_d-cl;
24  C1=0.557;C2=0.8;
25  L=Q/(2*C1*(2*9.81)^0.5*h^1.5/3+C2*d*(2*9.81*h)^0.5);
26  L=round(L*10)/10;
27  mprintf("\ncrest level=%f m.",cl);
28  mprintf("\nlength of crest=%f m.",L);
29  mprintf("\nprovide 4 bays of 7 m each with a clear
        water-way.");
30  tw=28+4.5;
31  mprintf("\nprovide 3 piers of 1.5 m width each.\
        ntotal width of cross regulator=%f m.",tw);
32  // design of d/s floor
33  L=28;
34  q=Q/L;
35  Hl=fsl_u-fsl_d;
36  Ef2=1.89;               //from blench curve
37  fl_d=fsl_d-Ef2;
```

317

```
38  mprintf("\n\ndesign of d/s floor :");
39  mprintf("\nd/s floor level=%f m.; which is higher
        than d/s bed level.\nadopt floor level =d/s bed
        level=215.40 m.",fl_d);
40  Ef1=Ef2+Hl;
41  //from specific energy curve
42  D1=0.7;D2=1.65;
43  cil=5*(D2-D1);        //cistern length
44  tl=2*16/3;
45  tl=round(tl*10)/10;
46  mprintf("\ncistern length =%f m.\nlength of d/s
        floor=%f m.",cil,tl);
47  //design of impervious floor
48  d1=hw/3+0.6;              //depth of u/s cut-off
49  w=0.5;                    //width of cut-off
50  d2=hw/2+0.6;              //deth of d/s cut-off
51  d2=2;                     //keep
52  Hs=fsl_u-(fsl_d-hw);   //maximum static head
53  n=Ge*d2/Hs;              //n=1/%pi*(lambda)^0.5;
54  //from exit gradient curves we get
55  alpha=8;n=0.148;
56  b=alpha*d2;
57  mprintf("\n\ndesign of impervious floor :");
58  mprintf("\ntotal length of impervious floor=%i m.;
        which is divided as-",b);
59  mprintf("\nd/s floor length=10.6 m.\nd/s glacis
        length with 2:1 slope=0.4 m.\nbalance to be
        provided upstream=5 m.");
60  d1=1.5;b=16;
61  alpha_=d1/b;
62  //hence
63  fic1=100-28;
64  fid1=100-19;
65  t=0.5;
66  fic1=fic1+(fid1-fic1)*t/d1;
67  mprintf("\n\npressure calculation :\nupstream cut-off
        :\npressure =%f percent.",fic1);
68  d2=2;b=16;
```

```scilab
69  alpha_=d2/b;
70  //hence
71  t=0.6;
72  fie2=31;fid2=22;
73  fie2=fie2-(fie2-fid2)*t/d2;
74  mprintf("\ndownstream cut-off:\npressure=%f percent.
        ",fie2);
75  t=10.6;
76  p=fie2+(fic1-fie2)*t/b;
77  p=round(p*10)/10;
78  mprintf("\ntoe of glacis:\npressure=%f percent.",p);
79  mprintf("\n\nthickness of floor:\nminimu thickness
        for u/s floor=0.5 m.");
80  rho=2.24;
81  t=fie2*2.7/(100*(rho-1));
82  t=round(t*100)/100;
83  mprintf("\nthickness of floor near d/s cut-off=%f m
        .\nprovide 0.7 m thick floor for last 2.1 m
        length.",t);
84  t=1.6/(rho-1);
85  t=round(t*100)/100;
86  mprintf("\nthickness of floor at toe of glacis=%f m.
        ",t);
87  t=6.6;
88  p=fie2+(fic1-fie2)*t/b;
89  t=p*2.7/(100*(rho-1));
90  t=round(t*100)/100;
91  mprintf("\nthickness of floor at 4 m from toe of
        glais=%f m.\nprovide 1.1 m thick floor for next 2
         m length",t);
92  t=4.6;
93  p=fie2+(fic1-fie2)*t/b;
94  t=p*2.7/(100*(rho-1));
95  t=round(t*100)/100;
96  mprintf("\nthickness of floor at 6 m from toe of
        glais=%f m.\nprovide 0.9 m thick floor for next
        2.5 m length",t);
97
```

```
 98  //design of u/s protection
 99  d1=hw/3+0.6;
100  v=d1;
101  v=round(v*100)/100;
102  mprintf("\n\ndesign of u/s protection:\nvolume of
        block protection=%f cubic metre/metre.",v);
103  mprintf("\nkeep thickness of protection=1 m.\
        nprovide 0.8mx0.8mx0.6m thick concret blocks over
         0.4 m thick apron in length of 0.6 m.");
104  cu=2.25*d1;
105  cu=round(cu*100)/100;
106  mprintf("\ncubic content of launching apron=%f cubic
         metre/metre.\nprovide 1 m thick and 3.5 m long
        launching apron.",cu);
107  //design of d/s protection
108  d2=hw/2+0.6;
109  v=d2;
110  v=round(v*100)/100;
111  mprintf("\n\ndesign of d/s protection:\nvolume of
        inverted filter=%f cubic metre/metre.",v);
112  mprintf("\nkeep thickness of concrete block=0.6 m.\
        nprovide 2 rows of 0.8mx0.8mx0.6m thick concret
        blocks over 0.6 m graded filter for length of 1.6
         m.");
113  cu=2.25*d2;
114  cu=round(cu*100)/100;
115  mprintf("\nlaunching apron volume=%f cubic metre/
        metre.\nprovide 1 m thick launching apron for
        length of 4.5 m.\nprovide a toe wall 0.4 m wide
        and 1.5 m deep between filter and launching apron
        .",cu);
116
117  //design of head regulator
118  mprintf("\n\n\nDESIGN OF DISTRIBUTORY HEAD REGULATOR
        ::");
119  //design of crest and waterway
120  mprintf("\n\ndesign of crest and waterway:");
121  cl=fsl_u-hw+0.5;
```

```scilab
122 h=fsl_u-fsl_dis;
123 d=fsl_dis-cl;
124 C1=0.557;C2=0.8;
125 L=Qd/(2*C1*(2*9.81)^0.5*h^1.5/3+C2*d*(2*9.81*h)^0.5)
      ;
126 L=round(L*100)/100;
127 mprintf("\ncrest level=%f m.",cl);
128 mprintf("\nlength of crest=%f m.",L);
129 mprintf("\nprovide 2 bays of 3.5 m each with a 1 m
       thick pier in between.");
130 tw=8;
131 mprintf("\ntotal width of cross regulator=%f m.",tw)
      ;
132 //design of d/s floor
133 L=7.5;
134 q=Q/L;
135 Hl=fsl_u-fsl_dis;
136 Ef2=1.58;          //from blench curve
137 fl_d=fsl_dis-Ef2;
138 mprintf("\n\ndesign of d/s floor:");
139 mprintf("\nd/s floor level=%f m.;\nkeepR.L of d/s
       floor=215.50 m.",fl_d);
140 Ef1=Ef2+Hl;
141 //from specific energy curve
142 D1=0.42;D2=2.55;
143 cil=5*(D2-D1);      //cistern length
144 tl=2*14/3;
145 mprintf("\ncistern length =%f m.",cil);
146
147 //design of impervious floor
148 d1=hw/3+0.6;             //depth of u/s cut-off
149 w=0.5;                   //width of cut-off
150 d2=hw_dis/2+0.6;          //deth of d/s cut-off
151 d2=2;                    //keep
152 Hs=fsl_u-215.5;       //maximum static head
153 n=Ge*d2/Hs;             //n=1/%pi*(lambda)^0.5;
154 //from exit gradient curves we get
155 alpha=7;n=0.154;
```

```
156  b=alpha*d2;
157  mprintf("\n\ndesign of impervious floor:");
158  mprintf("\ntotal length of impervious floor=%i m.;
         which is divided as-",b);
159  mprintf("\nlength below the toe of glacis=10.5 m\
         nlength of d/s glacis at 2:1 slope=1.2 m.\nwidth
         of crest=1 m.\nlength of u/s glacis at 1:1 slope
         =0.5 m.\nu/s floor:balnce=0.8 m.");
160  d1=1.5;b=16;
161  alpha_=d1/b;
162  //hence
163  fic1=100-28;
164  fid1=100-19;
165  t=0.5;
166  fic1=fic1+(fid1-fic1)*t/d1;
167  mprintf("\n\npressure calculation:\nupstream cut-off
         :\npressure =%f percent.",fic1);
168  d2=2;b=16;
169  alpha_=d2/b;
170  //hence
171  t=0.6;
172  fie2=31;fid2=22;
173  fie2=fie2-(fie2-fid2)*t/d2;
174  mprintf("\ndownstream cut-off:\npressure=%f percent.
         ",fie2);
175  t=10.6;
176  p=fie2+(fic1-fie2)*t/b;
177  p=round(p*100)/100;
178  mprintf("\ntoe of glacis:\npressure=%f percent.",p);
179  mprintf("\n\nthickness of floor:\nminimu thickness
         for u/s floor=0.5 m.");
180  rho=2.24;
181  t=p*2.6/(100*(rho-1));
182  t=round(t*100)/100;
183  mprintf("\nthickness under the crest=1 m.");
184  mprintf("\nthickness of floor at toe of glacis=%f m.
         ",t);
185  t=9.5;
```

```
186  p=fie2 +( fic1 - fie2 ) * t /b ;
187  t=p *2.7/(100*( rho -1) ) ;
188  t=round ( t *100) /100;
189  mprintf (" \ nthickness  of  floor  at  2 m from  toe  of
         glais=%f m. \ nprovide  1.1  m thick  floor  for  next  4
          m length ",t ) ;
190  t=4.5;
191  p=fie2 +( fic1 - fie2 ) * t /b ;
192  t=p *2.7/(100*( rho -1) ) ;
193  t=round ( t *100) /100;
194  mprintf (" \ nthickness  of  floor  at  6 m from  toe  of
         glais=%f m. \ nprovide  0.9  m thick  floor  for  next
         2.5  m length ",t ) ;
195  t=2;
196  p=fie2 +( fic1 - fie2 ) * t /b ;
197  t=p *2.7/(100*( rho -1) ) ;
198  t=round ( t *100) /100;
199  mprintf (" \ nthickness  of  floor  at  8.5  m from  toe  of
         glais=%f m. \ nprovide  0.7  m thick  floor  for  next  2
          m length ",t ) ;
200
201  // design  of  upstream  protection
202  d=hw /3+0.6;
203  d=round ( d *10) /10;
204  mprintf (" \ n \ ndesign  of  u/s  protection : \ nu/s  scour
         depth=%f m. \ nprovide  same  protection  as  in  cross
         regulator ",d ) ;
205
206  // design  of  d/s  protection
207  d2=hw_dis /2+0.6;
208  v=d2 ;
209  mprintf (" \ n \ ndesign  of  d/s  protection : \ nvolume  of
         inverted  filter=%f cubic  metre / metre .",v ) ;
210  mprintf (" \ nkeep  thickness  of  concrete  block =0.5  m. \
         nprovide  2 rows  of  0.8 mx0.8mx0.5m thick  concret
         blocks  over  0.5  m thick  graded  filter .") ;
211  cu =2.25* d2 ;
212  mprintf (" \ nlaunching  apron  volume=%f cubic  metre /
```

metre.\nprovide 1 m thick launching apron for
length of 3.5 m.\nprovide a masonary toe wall 0.4
 m wide and 1.2 m deep between filter and
launching apron.",cu);

Chapter 19

CROSS DRAINAGE WORKS

Scilab code Exa 19.1 EX19 1

```
1
2
3  //example 19.1
4  //design an expansion transition for canal by Mitra '
       s method
5  clc;funcprot(0)
6  //given
7  Lf=16;                //length of flume
8  Bf=9;                 //width of throat
9  Bo=15;                //width of canal
10
11 //width at any distance x from flumed section is
       given by
12 //Bx=Bo*Bf*Lf/(Lf*Bo-(Bo-Bf)x)
13 //on solving we get
14 //Bx=2160/(240-6x)
15
16 x=[2:2:16];          //distance
17 mprintf("width at any distance x from flumed section
       :");
18 for i=1:8
```

```scilab
19      Bx(i)=2160/(240-6*x(i));
20      Bx(i)=round(Bx(i)*100)/100;
21      mprintf('\n%f',Bx(i));
22  end
```

Scilab code Exa 19.2 EX19 2

```scilab
1
2
3  //example 19.2
4  //design an expansion transition for canal by
       Chaturvedi's method
5  clc;funcprot(0);
6  //given;
7  Lf=16;              //length of flume
8  Bf=9;               //width of throat
9  Bo=15;              //width of canal
10
11 x=[2:2:16];       //distance
12
13 //distance x is related as x=Lf*Bo^(2/3)(1-(Bf/Bx)
       ^1.5)/(Bo^1.5-Bf^1.5)
14 //on solving we get
15 //(Bf/Bx)^1.5=1-(x/29.893)    (relation is misprinted
        in book)
16 //let (Bf/Bx)^1.5=r
17
18 mprintf("width at any distance x from flumed section
       :");
19 for i=1:8
20     r(i)=1-(x(i)/29.893);         //Bf/Bx^(1.5)
21     R(i)=r(i)^(2/3);              //Bf/Bx
22     Bx(i)=Bf/R(i);
23     Bx(i)=round(Bx(i)*100)/100;
24     mprintf("\n%f.",Bx(i));
```

Scilab code Exa 19.3 EX19 3

```
1
2
3  //example 19.3
4  //design a syphon aqueduct
5  clc;funcprot(0);
6  //given
7  Q=25;                  //design discharge of canal
8  B=20;                  //bed width of canal
9  D=1.5;                 //depth of water in canal
10 bl=160;                //bed level of canal
11 hfq=400;               //high flood discharge of drainage
12 hfl=160.5;             //high flood level of drainage
13 bl_drain=158;          //bed level of drainage
14 gl=160;                //general ground level
15
16 //desing of drainage water-way
17 P=4.75*(hfq)^0.5;      //laecey P-Q formula
18 mprintf("design of drainage water-way:\nwetted
       perimeter of river=%i m.\nprovide 13 spans of 6 m
        each,separated by 12 piers each of 1.25 m thick.
       ",P);
19 t=78+15;
20 mprintf("\ntotal length of water-way=%i m.",t);
21 v=2;                   //velocity through syphon
22 hb=hfq/(78*v);
23 ac=hfq/(6*2.5*1.3);    //calculation is wrong in book
24 hb=round(hb*100)/100;
25 ac=round(ac*100)/100;
26 mprintf("\nheight of barrels=%f m.\nprovide
       rectangular barrels 6 m wide and 2.5 m high.\
       nactual velocity through barrels=%f m/sec.",hb,ac
```

327

```
      ) ;
27
28  // design of canal waterway
29  mprintf ( "\n\ndesign of canal waterway :\ nType 3
         aqueduct is adopted ." ) ;
30  l1=B-10;
31  l2 =(20-10) *3/2;
32  mprintf ( "\nproviding a splay 2:1 in expansion , length
          of contraction transition=%i m.\ nproviding a
         splay of 3:1 in expansion , length of expansion
         transition=%i m." ,l1 ,l2 ) ;
33  mprintf ( '\nIn transition side slopes are warped from
          original slope of 1.5:1 to vertical .' ) ;
34
35  // design of levels of different sectionn
36  mprintf ( "\n\ndesign of levels of different sectionn
         :\ nat section 4-4:" ) ;
37  A=(B+1.5*D) ;          // area
38  V=Q/A;                // velocity of flow
39  vh=V^2/(2*9.81) ;     // velocity head
40  ws=gl+D;              //R.L of water surface
41  tel=ws+vh ;
42  tel=round (tel *1000) /1000;
43  mprintf ( "\nR.L of T.E.L=%f m.\n at section 3-3:" ,tel
         ) ;
44  A=10*D;                        // area of trough
45  V=Q/A;                        // velocity
46  vh1=V^2/(2*9.81) ;            // velocity head
47  le=0.3*(vh1-vh) ;   // loss of head in expansion from
         section 3-3 to 4-4
48  tel=tel+le ;
49  rlw=tel -vh1 ;
50  rlb=rlw -D;
51  tel=round (tel *1000) /1000;
52  rlb=round (rlb *1000) /1000;
53  mprintf ( "\nelevation of T.E.L=%f m.\nR.L of bed to
         maintain constant water depth=%f m." ,tel ,rlb ) ;
54
```

```
55 //at section 2−2
56 R=A/P;
57 N=0.016;
58 S=V^2*N^2/R^(4/3);                //from manning's
      formula
59 L=93;                            //length of trough
60 hl=L*S;                          //head loss
61 tel=tel+hl;
62 rlw=tel-vh1;
63 rlb=rlw-D;
64 tel=round(tel*1000)/1000;
65 rlb=round(rlb*1000)/1000;
66 mprintf("\nat section 2−2:\nR.L of T.E.L=%f m.\nR.L
      of bed to maintain constant water depth=%f m.",
      tel,rlb);
67
68 //at section 1−1
69 hl=0.2*(vh1-vh);    //loss of hed in contraction
      transition
70 tel=tel+hl;
71 rlw=tel-vh;
72 rlb=tel-D;
73 tel=round(tel*1000)/1000;
74 rlb=round(rlb*1000)/1000;
75 mprintf("\nat section 1−1:\nR.L of T.E.L=%f m.\nR.L
      of bed to maintain constant water depth=%f m.",
      tel,rlb);
76
77 //design of contraction transition
78 //it is designed on the basis of chaturvedi's
      formula
79 Bo=20;
80 Bf=10;
81 L=10;
82 //from chaturvedi formula we get relation between x
      and Bx as: x=15.45(1−(10/Bx)^1.5);
83 Bx=[10:1:20];
84 mprintf("\n\ndesign of contraction transition on the
```

329

```scilab
           basis  of  chaturvedi  formula:\nBx                    x");
85  for  i=1:11
86      x(i)=15.45*(1-(10/Bx(i))^1.5);
87      x(i)=round(x(i)*100)/100;
88      mprintf("\n%i            %f",Bx(i),x(i));
89  end
90
91  //design  of  expansion  transition  on  the  basis  of
        chaturvedi  formula
92  L=15;
93  Bf=10;Bo=20;
94  //from  chaturvedi  formula  we  get  relation  between  x
        and  Bx  as:  x=23.15(1-(10/Bx)^1.5);
95  mprintf("\n\ndesign  of  expansion  transition  on  the
        basis  of  chaturvedi  formula:\nBx                x");
96  for  i=1:11
97       x(i)=23.15*(1-(10/Bx(i))^1.5);
98       x(i)=round(x(i)*100)/100;
99      mprintf("\n%i            %f",Bx(i),x(i));
100  end
101
102  //design  of  trough
103  mprintf("\n\ndesign  of  the  trough:");
104  mprintf("\nflumed  water  way  of  canal=10  m.\ntrough
        carrying  canal  will  divide  into  two  compartments
        each  5  m  wide  an  dseparated  by  0.3  m  thick
        partiions.\nheigth  of  trough  will  be  =  2  m.\
        ntrough  iss  constructed  using  monolithic
        reinforced  concrete.\nthe  outer  and  inner  walls
        ca  be  kept  0.4  m  thick.\nthus,outer  width  of
        trough  =  11.1  m.");
105
106  //head  loss  through  syphon  barrels
107  V=2.05;          //velocity  through  barrels
108  f1=0.505;        //coefficient  of  loss  of  head  at
        entry
109  a=0.00316;b=0.030;
110  R=(6*2.5)/(2*(6+2.5));
```

```scilab
111  f2=a(1+b/R);
112  L=11.1;              //length of barrel
113  h=(1+f1+f2*L/R)*V^2/(2*9.81);
114  hfl_up=hfl+h;
115  h=round(h*1000)/1000;
116  hfl_up=round(hfl_up*1000)/1000;
117  mprintf("\n\nhead loss through syphon barrels=%f m.\
         nupstream H.F.L=%f m.",h,hfl_up)
118
119  //uplift pressure on the roof
120  bt=gl-0.4;              //R.L of bottom of the trough
121  hl=0.505*V^2/(2*9.81);
122  u=hfl_up-hl-159.6;
123  up=u*9.81;
124  mprintf("\n\nuplift pressure on the roof=%f kN/
         square m.\ntrough slab is 0.4 m thick and exert a
          downward load of 9.42 kN.",up);
125  mprintf("\nth ebalance of the uplift pressure has to
          be resisted by bending action of trough slab.\
         nso,reinforcement has to be provided at the top
         of the slab.");
126
127  //uplift on the floor of the barrel and its design
128  //(a) static head
129  mprintf("\n\nuplift on the floor of the barrel and
         its design:\n(a) static head:");
130  bf=bt-2.5;              //R.L of barrel floor
131  t=0.8;                  //tentative thickness of floor
132  bot=bf-t;
133  static=bl_drain-bot;
134  static=round(static*100)/100;
135  mprintf("\nstatic uplift on the floor=%f m.",static)
         ;
136
137  //(b) seepage head
138  L=10;                 //length of u/s transition
139  bs=3;                 //half the barrel span
140  df=11;                //end drainage floor
```

```
141  tcl=24;                    //total creep length
142  tsh=161.5-bl_drain;    //total seepage head
143  rs=tsh*(1-13/tcl);       //residual seepage at B
144  tu=(static+rs)*9.81;
145  tu=round(tu*100)/100;
146  mprintf("\n(b) seepage head:\ntotal uplift=%f kN/
        square m.\nprovide thickness of floor 0.8 m",tu);
147  bending=tu-17.58;
148  bending=round(bending*100)/100;
149  mprintf("\nuplift to be resisted by bending action
        of floor=%f kN/square m.",bending);
150
151  //design of cut-off and protection works for
        drainage floor
152  mprintf("\n\ndesign of cut-off and protection works
        for drainage floor:");
153  Q=400;f=1;
154  R=0.47*(Q/f)^(1/3);
155  d_up=1.5*R;                          //depth of u/s cut-
        off
156  bot_up=hfl_up-d_up;                  //R.L of bottom of u
        /s cut-off
157  d_down=1.5*R;                        //depth of d/s cut-
        off
158  bot_down=hfl-d_down;                 //R.L of bottom of d/
        s cut-off
159  l_down=2.5*(bl_drain-bot_down);
160  l_down1=2*(bl_drain-bot_up);
161  bot_up=round(bot_up*100)/100;
162  bot_down=round(bot_down*100)/100;
163  l_down=round(l_down);
164  l_down1=round(l_down1);
165  mprintf("\nR.L of bottom of u/s cut-off=%f m.\nR.L
        of bottom of d/s cut-off=%f m.",bot_up,bot_down);
166  mprintf("\nlength of d/s protection consisting of 40
         cm brick pritching=%f m.\npitching is supported
        by toe wall 0.4 m wide and 1.5 m deep at its d/s
        end.\nlength of d/s protection consisting of 0.4
```

```
cm brick pritching=%f m.\npitching is supported
by toe wall 0.4 m wide and 1 m deep at its u/s
end.",l_down,l_down1);
```

Chapter 20

RIVER ENGINEERING

Scilab code Exa 20.1 EX20 1

```
1
2
3  //example 20.1
4  //design   a guide bank required for a bridge in a
       river
5  //calculate volume of stone required per m length of
        guide bank
6  clc;funcprot(0);
7  //given
8  Q=50000;            //discharge
9  f=1.1;              //silt factor
10 bl=130;             //bed level of river
11 hfl=140;            //high flood level
12 L=4.75*(Q)^0.5;
13 L=L+212;            //providing 20 percent more length
14 L_up=5*L/4;         //upstream length of guide bund
15 L_down=L/4;         //downstream length of guide bund
16 r_up=0.45*L;        //radius of upstream curved head
17 mprintf("upstream length of guide bund=%i m.",L_up);
18 mprintf("\ndownstream length of guide bund=%i m.",
     L_down);
```

334

```
19  mprintf("\nupstream radius of curved head=%i m.; it
        can be carved at 145 degrees.",r_up);
20  mprintf("\ndownstream radius of curved head=287m.; it
         can be carved at 60 degrees.");
21
22  fb=1.5;          //free board
23  ltop=fb+hfl;   //level of top of guide bund
24  mprintf("\n\nlevel of top of guide bund=%f m.",ltop)
        ;
25  mprintf("\nadopt top level=142 m.");
26  ltop=142;
27  Hr=ltop-bl;
28  mprintf("\nkeep top width=4 m. and side slope as
        2:1.");
29  T=0.06*(Q)^(1/3);              //thickness of stone
        pitching
30  T=round(T*100)/100;
31  mprintf("\n\nThickness of stone pitching=%f m.",T);
32  R=0.47*(Q/f)^(1/3);          //depth of scour
33  Rmax=1.25*R;                 //maximum scour
34  rl=hfl-Rmax;                 //R.L at maximum anticipated
         cover
35  D=bl-rl;                     //depth of maximum scour
36  Lapron=1.5*D;
37  R=round(R*100)/100;
38  Lapron=round(Lapron*100)/100;
39  mprintf("\ndepth of scour=%f m.",R);
40  mprintf("\n\nfor straigtht reach of guide band:");
41  mprintf("\nlength of apron=%f m.",Lapron);
42  Rmax=1.5*R;
43  rl=hfl-Rmax;
44  D1=bl-rl;
45  Lapron=1.5*D1;
46  R=round(R*100)/100;
47  mprintf("\n\nfor curvilinear transition portion of
        guide band:");
48  mprintf("\nlength of apron=%f m.",Lapron);
49  T1=1.9*T;
```

```
50  T1=round(T1*10)/10;
51  mprintf("\nthickness of apron=%f m.",T1);
52  mprintf("\n\nvolume of stones:");
53  ss=5^0.5*(141-130)*T;
54  as=5^0.5*D*1.25*T;
55  ss=round(ss*100)/100;
56  as=round(as*100)/100;
57  mprintf("\nat shank:");
58  mprintf("\non slope=%f cubic metre/m.",ss);
59  mprintf("\non apron with a slope 2:1 =%f cubic metre
        /m.",as);
60
61  va=5^0.5*D1*1.25*T;
62  vs=ss;
63  vs=round(vs*100)/100;
64  va=round(va*100)/100;
65  mprintf("\nU/S andD/S curved portion:");
66  mprintf("\non slope=%f cubic metre/m.",vs);
67  mprintf("\non apron =%f cubic metre/m.",va);
68
69  ta=va/(1.5*D1);
70  ta=round(ta*10)/10;
71  mprintf("\n\nthickness of launching apron=%f m.",ta)
        ;
```

Chapter 21

WATER POWER ENGINEERING

Scilab code Exa 21.1 EX21 1

```
1
2
3  //example 21.1
4  //calculate
5  //total installed capacity
6  //load factor
7  //plant factor
8  //utilization factor
9  clc;funcprot(0);
10 //given
11 c=10000;                    //capacity of each generator;
12 n=3;                  //number of generator
13 l1=12000;               //initial load on plant
14 l2=26000;                //final load on plant
15
16 tc=n*c;
17 mprintf("Total installed capacity=%i kW.",tc);
18
19 avg=(l1+l2)/2;          //average load
```

```
20  pk=12;                        //peak load
21  lf=avg*100/pk;
22  lf=round(lf*10)/10;
23  mprintf("\nload factor=%f percent.",lf);
24
25  //take any time duration t hours
26  pf=avg*100/tc;
27  pf=round(pf*10)/10;
28  mprintf("\nplant factor=%f percent.",pf);
29
30  uf=pk*100/tc;
31  uf=round(uf*10)/10;
32  mprintf("\nutilization ratio=%f percent.",uf);
```

Scilab code Exa 21.2 EX21 2

```
1
2
3  //example 21.2
4  //calculate maximum generation capacity of generator
5  //pondage to be provided
6  clc;funcprot(0);
7  //given
8  Q=40;            // minimum flow in river
9  H=30;            //net head
10  lf=0.73;          //load factor
11  eita=0.6;        //plant efficiency
12
13  P=9.81*Q*H*eita;
14  pk=P/lf;
15  pk=round(pk*10)/10;
16  mprintf("maximum generation capacity of generator=%f
       kW.",pk);
17
18  pp=pk-P;                  //power develop from pondage
```

```
19  Q=pp/(9.81*H*eita);
20  pr=Q*4*3600/10000;
21  pr=round(pr*10)/10;
22  mprintf("\nPondage required=%fD+4 cubic metre.",pr);
```

Scilab code Exa 21.3 EX21 3

```
 1
 2
 3  //example 21.3
 4  //calculate minimum discharge required in the stream
 5  //maximum load factor
 6  clc;funcprot(0);
 7  //given
 8  c=15000;              //installed capacity of plant
 9  lf=0.3;               //load factor
10  eita=0.82;            //plant efficiency
11  H=25;                 //working head
12
13  avg=c*lf;         //average power developed
14  Q=avg/(9.81*H*eita);
15  Q=round(Q*100)/100;
16  mprintf("minimum discharge required in the stream=%f
        cumecs.",Q);
17
18  Q=32;             //for second case
19  P=9.81*H*Q*eita;
20  lf=P*100/c;
21  lf=round(lf*10)/10;
22  mprintf("\nmaximum load factor=%f percent.",lf);
```